

dreamweaver mx: building on solid

foundations

AL SPARBER - GERRY JACOBSEN

```
body {  
  foundation: solid;  
  content: accessible;  
  layout: cool;  
}
```

PVII PRESS

PVII Visions

Publishing Information

DREAMWEAVER MX: BUILDING ON SOLID FOUNDATIONS

Copyright © 2002 by PVII PRESS

FIRST EDITION: SEPTEMBER, 2002

All rights reserved. You may print one copy of this E-Book for personal use. Beyond that, no part of this book may be reproduced or transmitted in any form or by any means, or photocopied, recorded, or electronically copied or distributed without the express written permission of PVII PRESS—except for minor inclusions for the purposes of quoting in a review.

TRADEMARKS

All references in this book to items, products, or services that are known to be trademarks have been capitalized. PVII PRESS cannot confirm the accuracy of this information. While every precaution has been taken, neither PVII PRESS nor PVII (Project Seven Development) assume responsibility for errors or omissions, or for damages resulting from the use of the information contained in this book.

Authors: Al Sparber and Gerry Jacobsen

Co-Publishers: Al Sparber and Gerry Jacobsen

Contributing Writers: Linda Rathgeber and Patricia Ayers

Acknowledgements

This book would not have been possible without the help of our testing and technical editing team: Ken Ford, Karen Boyle, Corey Eiges, David Powers, Patty Ayers, Linda Rathgeber, Robert Boughner, and David Nicholls.

We would be remiss if we didn't mention our friends, customers, and associates who hang out on the PVII Newsgroups. You are our greatest assets.

Thanks to Macromedia for making such a wonderful program to write about. And a special thanks to Macromedia's community voices: Matt Brown, John Dowdell, and Vern Viehe. Without you gentlemen, there would be no Dreamweaver community.

THANK YOU ONE AND ALL

Introduction

So, we're sitting around talking about writing a new Dreamweaver book and it occurs to us that the best way to teach somebody how to do something is to do it step by step—together. In a sense, this **book** is a collaborative project. We're going to work together as a team—*building a solid foundation*.

Dreamweaver MX: Building on Solid Foundations is an invitation to work with us as we plan and code Web pages, design images, grapple with cross-browser issues, and identify *and solve* any problem that stands between us and our goal.

WHAT YOU CAN EXPECT

You can expect to learn how to build structurally sound, efficient, accessible, and engaging Web pages. You'll start with a plan, erect an HTML structure, and then style it with CSS—*as you go*. The approach is linear, with an emphasis on cause and effect. You'll learn by doing—and as you *do*, you'll learn both how *and* why things work.

You'll get up close and personal with the source code. You'll understand the pieces and parts that give shape to a Web page—to *your* page. You'll become CSS authors. You'll learn how to use every weapon in Dreamweaver's CSS arsenal, and you'll learn how CSS works in harmony with HTML.

You'll also learn how to separate a page's structure (HTML) from its presentation (CSS). If you think you don't need to know about HTML or CSS because you only make *data-base-driven* sites, think again. A data-driven site merely *pours* content into a page's structure that would otherwise be typed manually. If the underlying structure isn't sound, database technology offers no extra protection. Web browsing software neither knows nor cares whether you typed content on your keyboard or retrieved it from a database—but it does care if your markup is well-formed and correct.

Mastering **Foundations** is all about your ability to follow steps—and has nothing to do with your current skill level. As you follow along, remember to keep your eyes open because there's lots going on between the lines—such as tips and notes. Pause and reflect as you go. Explore all the nuances and *read each footnote*.

SUPPORT AND COMMUNITY

When you buy a PVII book or product, you also buy into a worldwide community that starts on our Web site (www.projectseven.com) and continues on our newsgroup (<news://forums.projectseven.com/pviiwebdev>)¹. The PVII newsgroup is a wonderful place to be—a place where you can interact with hundreds of Dreamweaver users, as well as the PVII team.

ERRATA

You'll want to check our **Foundations** Web page on a regular basis for news, corrections, and updates at <http://www.projectseven.com/foundations>.

FOUNDATIONS

While the official title of this work is **Dreamweaver MX: Building on Solid Foundations**, the emphasis is definitely on the word *foundation*. You'll be seeing that word often throughout the book.

WHY DID WE MAKE FOUNDATIONS AN E-BOOK?

- It was an easy decision. This format keeps control of the book's content and tone where it belongs—in the hands of the authors.
- It also gives us complete freedom to issue revisions when (or if) the need arises.
- It takes far less time to deliver the finished work.
- It gives you, the reader, more flexibility. You can read it on your screen with either Acrobat Reader or Adobe E-Book Reader free software. You can even have it read aloud to you.
- It travels well. You can read it on your laptop while waiting for the train.
- It never gets *dog-ears*. A fresh copy is always available.
- You can print parts of it, or the whole thing on your desktop or office printer.
- You can have one copy printed and bound at a chain print shop like Kinko's®.
- A small forest has been saved from the axe.

¹ If you are not familiar with Newsgroups, all you need is a **Newsreader** such as Microsoft Outlook Express (PC and Mac), Microsoft Entourage (Mac), or Netscape Communicator. If you have trouble connecting, send us mail (support@projectseven.com) and we'll get you fixed up in short order.

Conventions Used in this Book

MENU CONVENTIONS

Menus and their options are referred to using arrows, as in:

File→ **Open**, **Edit**→ **Preferences**, **Insert**→ **Table**, etc.

PC / MAC EQUIVALENTS

In the course of the book, we refer to PC keyboard sequences. Here are the Mac equivalents:

PC	Mac
Press Enter	Press Return
Right-Click	Control-Click
Ctrl +	Command +

CODE

When illustrating code that exists on a page, we use a monospace font:

```
<p>This is a paragraph.</p>
```

CURSORS

The pipe character | indicates your cursor's position. For example: `<td |>` indicates that your cursor is inside the `<td>` tag just before the right angle bracket.

SEQUENTIAL STEPS

Steps that you must complete sequentially are prefaced by a leading dash:

-Place the jack under the right corner of the front bumper.

Be sure to complete each step in order to successfully complete the task.

BROWSER PREVIEWS

To preview a page in your default browser, press F12. To set up your other browsers to work with Dreamweaver...

-Open the Preferences window by pressing Ctrl + U (Command + U – Mac).

-Select the *Preview in Browser* Category.

Your default system browser should be present. If not, click the plus (+) sign to add a new browser.

To set a browser as your *Secondary Browser*, select it in the *Browser* box and put a check in the *Secondary Browser* box.

Preview Using Temporary File is a default setting and requires that you close and reopen your browser to see the effects of any page edits made in Dreamweaver. Your page does not have to be saved first in Dreamweaver to see the changes in your browser. If you'd like to leave your browser open and use its refresh or reload feature to see your changes, you need to uncheck the *Preview Using Temporary File* option. However, you will need to save your Dreamweaver page before refreshing your browser.

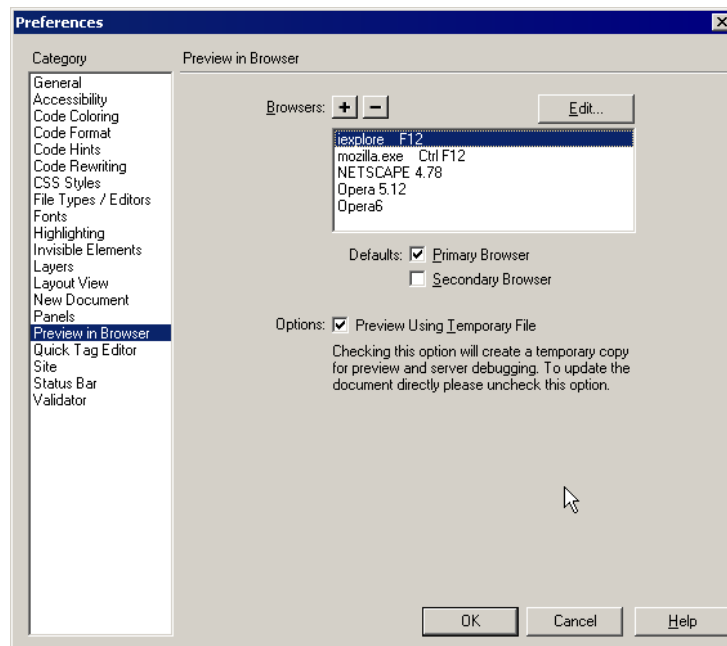


FIGURE 1

SAVING FILES

To prevent losing your work, please save often as you edit. Ctrl + S will save the file you are working on (Command + S on the Mac).

Whenever you edit a CSS file, you should save the file. When switching from a CSS file to an HTML document that file is linked to, it's a good idea to close the CSS file¹.

SWITCHING VIEWS

Throughout the book's course, we'll ask you to switch to Code View, Design View, or Code and Design (Split View). Switching views is accomplished by using the main View menu or the three view icons located just to the left of the Title Box and just above your edit window.

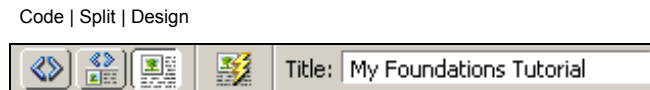


FIGURE 2

FOLDER LISTS

Folder lists in Windows are opened and closed (expanded and collapsed) by clicking plus (+) and minus (-) icons. Mac OS uses triangle icons instead.

¹ When working inside an HTML document and editing a style with the CSS Editor, Dreamweaver is accessing the CSS file behind the scenes, as it were. If the CSS file is open, the changes will be made—but you'll need to save the file before seeing the changes reflected in the HTML document. Worse yet, Dreamweaver may become totally confused when you switch back to the CSS file and may not prompt you to save it when you finally do close it.

Table of Contents

PUBLISHING INFORMATION	2
Dreamweaver MX: Building on Solid Foundations	2
First Edition: September, 2002	2
Trademarks.....	2
 ACKNOWLEDGEMENTS	 3
Thank You One and All	3
 INTRODUCTION.....	 4
What You Can Expect.....	4
Support and Community	5
Errata.....	5
Foundations	5
Why Did We Make Foundations an E-Book?	5
 CONVENTIONS USED IN THIS BOOK	 6
Menu Conventions.....	6
PC / Mac Equivalents.....	6
Code	6
Cursors.....	6
Sequential Steps.....	6
Browser Previews.....	7
Saving Files	8
Switching Views	8
Folder Lists	8
 TABLE OF CONTENTS	 9
 CHAPTER 1—SETTING UP.....	 15
Moving the Unzipped Folders	15
The Files Included with the Book	16
The Included Files	16
Define a Local Site.....	17
Open Dreamweaver MX	17
Install The Filler Text Snippet and SwapClass Extension.....	21
 CHAPTER 2—PANEL MANAGEMENT	 23
The MX Workspace.....	23
Before Continuing.....	23
Mac Users.....	23
Close a Panel Group.....	24
Modify a Panel Group.....	25

Group a New Panel on an Existing Panel.....	26
Add the Tag Inspector to the Files Group.....	27
Add the Snippets panel to the Files Group.....	27
Adjusting Panel Group Height.....	28
Changing the Panel Bar Width.....	28
Changing to Dreamweaver 4 Workspace (PC Only).....	29

CHAPTER 3—SETTING UP A NEW PAGE.....30

Setting Preferences	30
Create a New Document.....	31
Adding a Page Title.....	32

CHAPTER 4—ADDING A DOCTYPE & A SNIPPET 33

The Doctype.....	33
Creating a Snippet	34
Inserting a Snippet.....	36

CHAPTER 5—DREAMWEAVER ACCESSIBILITY AIDS 38

Enabling Dreamweaver MX Accessibility Features.....	38
Declaring the default page language.....	39

CHAPTER 6—PLANNING THE PAGE LAYOUT.....40

The design strategy	40
Assembling an Assets List.....	43

CHAPTER 7—CREATING THE FOUNDATIONS IMAGES 44

Create a New Document and Configure the Canvas.....	44
Rulers and Pixels and Guides	47
Setting Up The Fireworks MX Workspace.....	48

Drawing the Foundations Images 50

Creating the Button Background.....	50
The Table Corners and Borders Images	54
Creating the Logo.....	58
The Home Icon	61
The Shim (transparent spacer GIF).....	65
The Main table cell background	66
The Foundations Background Tile.....	67

Slicing, Optimizing & Exporting 68

Slicing the Button Background	68
Slicing the Logo.....	69
Slicing the Shim.....	69
Slicing the Main Table Cell Background Image	69
Slicing the Table Corners and Borders	70
Slicing the Striped Background Tile.....	72

Optimizing the Foundations Images.....72

Optimizing the Button Background	73
Optimizing the Logo.....	73
Optimizing the Shim.....	74
Optimizing the Main Table Cell Background Image	75
Optimizing the Table Corners and Borders	75
Optimizing the Striped Background Tile.....	75

Exporting the Foundations Images	76
In Closing	80

CHAPTER 8—BUILDING THE FOUNDATIONS PAGE 81

Inserting a Table	81
The Accessibility Dialog	82
Centering the Table	83
Checking the Source Code	84
Code Formatting	85
Inserting the Logo	87
Blueprinting our Table and Assigning IDs	89
Reviewing the CSS ID Selectors	90
Creating a Style Sheet	90
A CSS Rule Defined	91
CSS Efficiencies	91
Setting CSS Preferences	92
Enabling CSS in the Properties Inspector	93
Creating a Style Sheet and Linking it	94
Copy and Paste: the CSS Code Warrior's Ally	98
Building Up The Table	101
Insert Shim Images and IDs Where Needed	101
Code Check	107
Find and Replace	107
If Non-Breaking Spaces are Breaking Your Heart	109
Styling the Body Element	111
Collapsing the Table	116
Creating the #mn ID Selector	117
Adding a Level 1 Heading	122
Adding a Paragraph	122
The Vertical Alignment Compromise	123
Styling the H1	124
Time for a Browser Check	126
Creating a Remedial CSS Rule	128
Styling the <p>	129
Save and Switch	130
Styling The Footer ID Selector	131
ID the Footer <td>	134
Add an Address	135
Creating an <address> Selector	136
The Sidebar ID Selector	138
ID the Sidebar Cell	139
Adding Hyperlinks to the Sidebar	140
Code Check	142
Making Hyperlink Styles	143
#mn Hyperlinks	143
Make a test link	144
Creating the #mn Link Styles	145
The footer link styles	148
Preview	149
The sidebar navigation styles	150
Assign Classes to the Main and Sub-Section Links	155
using the #nbutspacer ID	157
Making Buttons from Text Part 1: The Main Section Links	158
Making Buttons from Text Part 2: The Sub-Section Links	164

CHAPTER 9—MAKING A STYLE SHEET FOR VERSION 4 BROWSERS 168

Take Two Style Sheets and Call Me in the Morning	168
Planning the Version 4 Style Sheet	169
Making a Copy of mfv5.css	171
Reset the Linked Style Sheet	172
Importing the Version 5 Style Sheet	173
Editing the Version 4 Style Sheet.....	174
Setting Font-Family for Version 4 Browsers	174
Editing The Sidebar Link Styles	175
Editing the <h1> Selector	179
Editing the #mn and #mn p Selectors	180
Editing the #footer Selector	182
Editing the Address Selector	183
Previewing the Finished Style Sheet in Navigator 4	184
Code Check	185
The Dreamweaver MX @Import Issue.....	187
About Duplicate Rules and Redundant Properties	188
The Navigator 4 URL Issue	188
Solving the Navigator 4 CSS URL Issue	191
Saved Queries	192
Using the Saved Query	195
Optimizing Our Two Style Sheets	197
A Hypothetical Example	197
The Optimization Process	198
CSS Code Check—mfv4.....	201
CSS Code Check—mfv5.....	203

CHAPTER 10—EXTRA INGREDIENTS 204

How do I turn temporary links into real links—and make more pages?	204
Can I mark the navigation link that relates to the current page, so users know where they are?	205
Can I add images inside the main content table cell?.....	206
Can I add a table inside the main content cell so I can list my company's widgets?	207

CHAPTER 11—ADDING INTERACTIVITY TO THE MENU.....212

Dressing up the Main Buttons	212
Editing the Navbg Image.....	212
Making a Second Navbg Image for the Hover Class	213
Slicing it	214
Exporting to Foundations	215
And Now For a Little CSS Wizardry	216
Editing the p.navbut a:hover, p.navbut a:active Rule.....	216
Preview in a Version 5 (or higher) Browser	218

CHAPTER 12—STEPPING SMARTLY OUTSIDE THE BOX.....219

How the Menu Will Work	219
Browser Limitations and Accessibility	220
Specific Browsers	220
The Menu Mechanics	220
The Styles SwapClass Will Use	221
Creating the CSS Styles for our SwapClass Menu	222
The Hacks	223
Adding More Menu Sublinks.....	224
The Great Divide	226
Code Check	231

Editing the Tag Library	231
Applying the SwapClass JavaScript Behavior	232
Menu Modifications	237

CHAPTER 13-EXTERNALIZING THE SWAPCLASS JAVASCRIPT 241

Move the Script	241
Create the JavaScript File	242
Link the JavaScript File to Your Page	243

CHAPTER 14—ALTERNATE IMAGE SETS.....245

Overview	245
Changing the Images	246
Matching Style Sheet Sets for Each Color Set	250
Backing Up the Style Sheets	250
Setting Design Time Style Sheets	252
Editing the Yellow Style Sheets	254
Turning off Design Time Style Sheets	258
Resetting the Default Style Sheets to the Yellow Color Theme	259
Managing Multiple Color Schemes	262

CHAPTER 15-THE TABLE LOBOTOMY 263

Look Mom—No Table	263
What are the Benefits of Table-Less Designs?	263
Planning the Structure	263
Create the Lobotomy Page	264
Creating the Base Structure	264

The Base Structure Defined 265

The Logo DIV	265
The Main DIV	267
The Evolving Structure	270
The Navbar DIV	271

Accessibility Check..... 273

Code Check 274

Writing a Style Sheet for Version 4 Browsers 275

Create the Version 4 Style Sheet and the Body Selector	275
Identify and Create the Remaining Selectors	277
Code Check—mfv4_lobo.css	286
Apply the Netscape Resize Fix	287

CSS Unleashed—The Lobotomy Style Sheet for Version 5 Browsers 288

Making a Copy of mfv4_lobo.css	288
Importing the Version 5 Style Sheet	289

Editing and Optimizing the Version 5 Style Sheet 290

The Plan	290
Let's do the math	290
Editing the Version 5 Style Sheet	291
Preview and Behold a CSS Miracle	292
Code Check—mfv5_lobo.css	299
The Finished Page	300
Does it Print?	301
A Print Media Style Sheet to the Rescue	302
Creating the Print Style Sheet	302
Linking the Print Media Style Sheet	305
Testing the Print Style Sheet	307

CHAPTER 16—AFTERTHOUGHTS.....308
We Wrote Foundations because—..... 308

Chapter 1—Setting Up

Before we begin the lesson, please make a backup copy of the **PVII_FoundationsMX.zip** archive for safekeeping.

MOVING THE UNZIPPED FOLDERS

Create a new folder on your main hard disk drive and name it Studio VII. Place it at the drive's top level (or root) so you'll be able to locate it easily.

Copy the unzipped (or unstuffed) **PVII_FoundationsMX** archive inside the Studio VII folder. The Foundations folder (inside your Studio VII folder) should look like this:

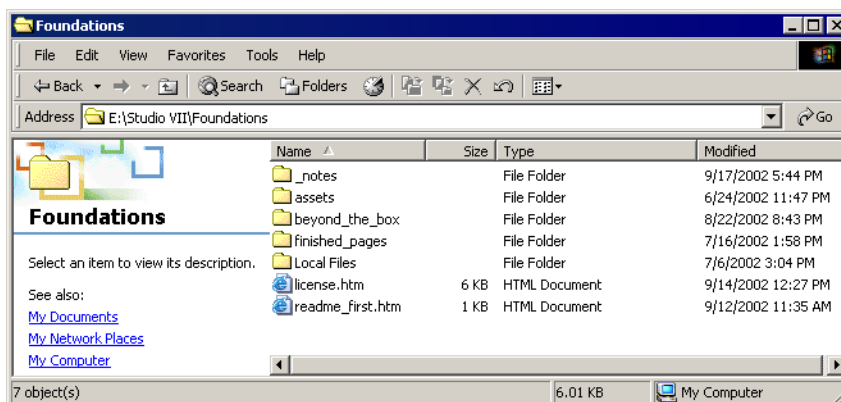


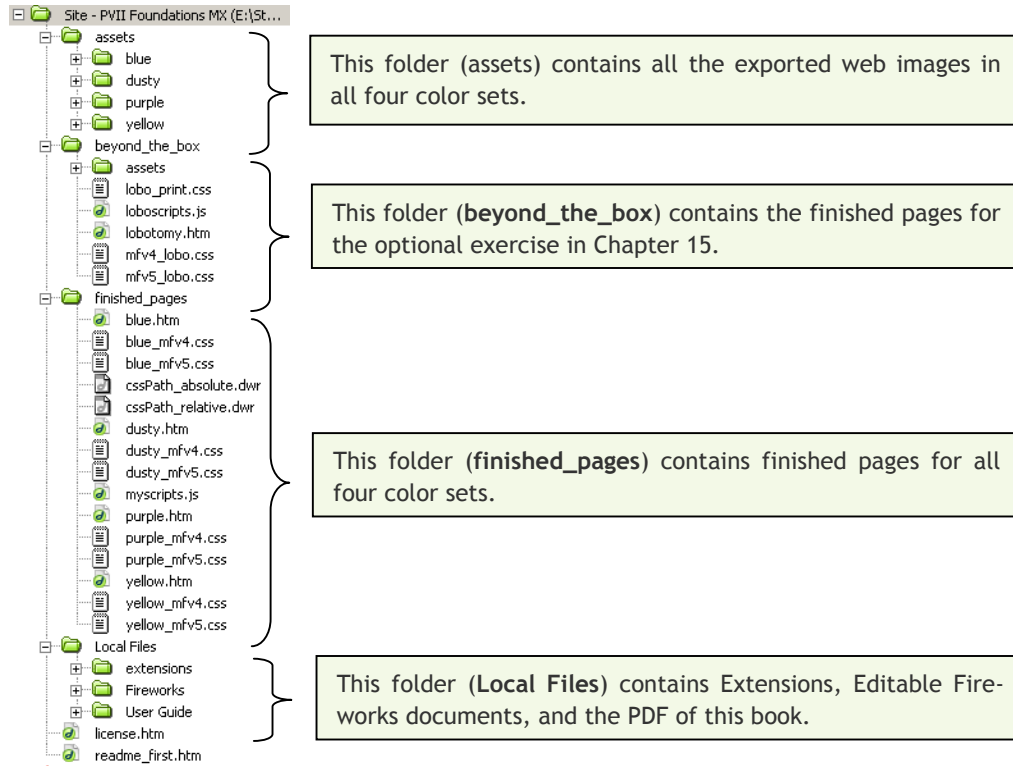
FIGURE 3- THE FOUNDATIONS DIRECTORY STRUCTURE AS SEEN IN WINDOWS 2000

We use a second Hard Drive (E) for data. If you have a single drive on your machine, then you will probably be unzipping to C:\Studio VII.

THE FILES INCLUDED WITH THE BOOK

THE INCLUDED FILES

These are the files and folders you'll find inside the zipped archive.



license.htm and readme_first.htm are in the site root.

DEFINE A LOCAL SITE

Dreamweaver needs to know all about the files you will be using and creating. You need to define a site.

OPEN DREAMWEAVER MX

- Choose Site→ New Site. The Site Definition dialog box will open.
- Make sure the **Basic Tab** is selected. The Basic Tab presents you with a “Wizard-driven” series of dialogs and is appropriate for this lesson. The Advanced Tab can be used at any time if you’d like more control over the options¹.
- Enter a name for your new site. We chose **PVII Foundations MX**. You can use any name you want.

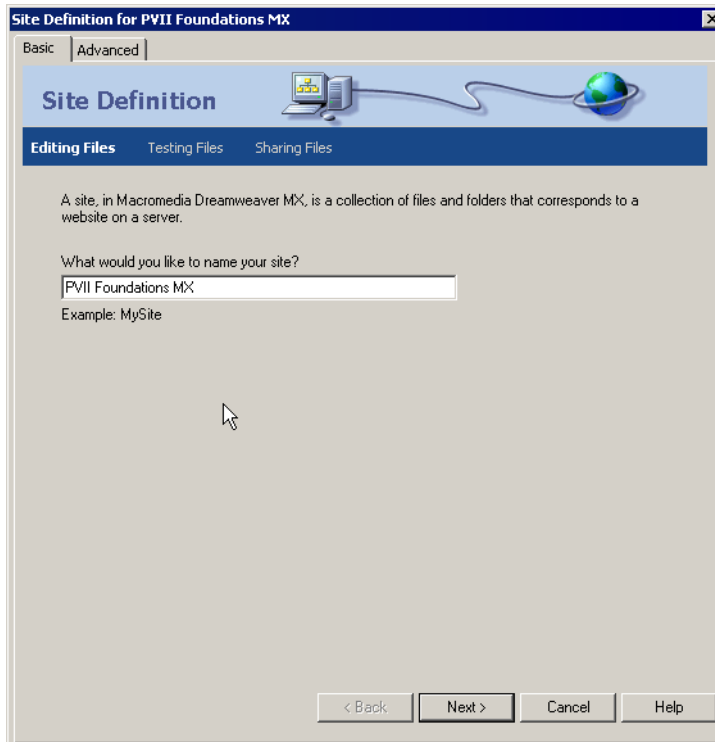


FIGURE 4- NAME THAT SITE

- Click **Next**. The Server Technology screen will appear.

¹ Macromedia does an excellent job of documenting the Site Definition window. To access Help at any time, press F1.

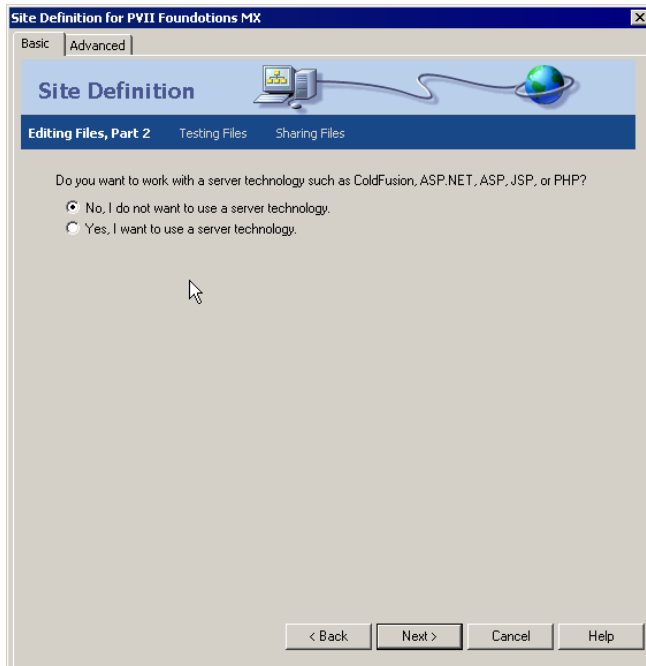


FIGURE 5- SERVER TECHNOLOGY

-Select **No**. We won't be using database technology for this site.

-Click **Next**. The Editing Files screen opens...

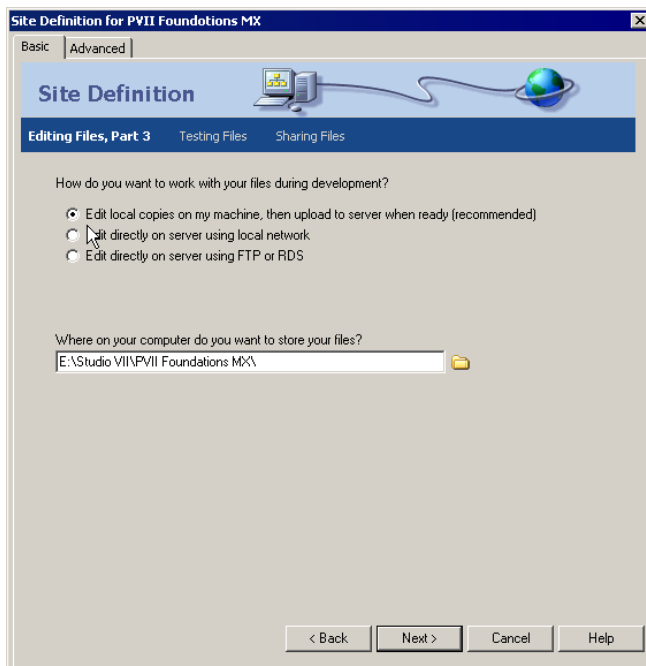


FIGURE 6- WORKING WITH SITE FILES

-Select **Edit local copies on my machine, and then upload to server when ready**.

-In the text box labeled *Where on your computer do you want to store your file*, click the little file folder icon to the right of the text box to open up the *Choose Local Root Folder* window, from which you can navigate your computer's file system to your newly created **Foundations** folder.

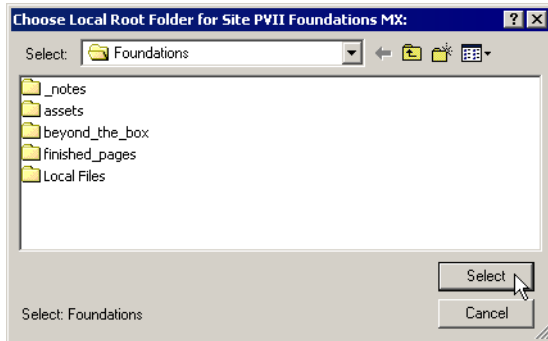


FIGURE 7- SELECTING THE ROOT FOLDER

-Click **Select** to choose *Foundations* as your new site's root folder. The Choose Local Root Folder window will close and you'll be returned to the main Site Definition window.

Click **Next**. The Connection dialog will open.

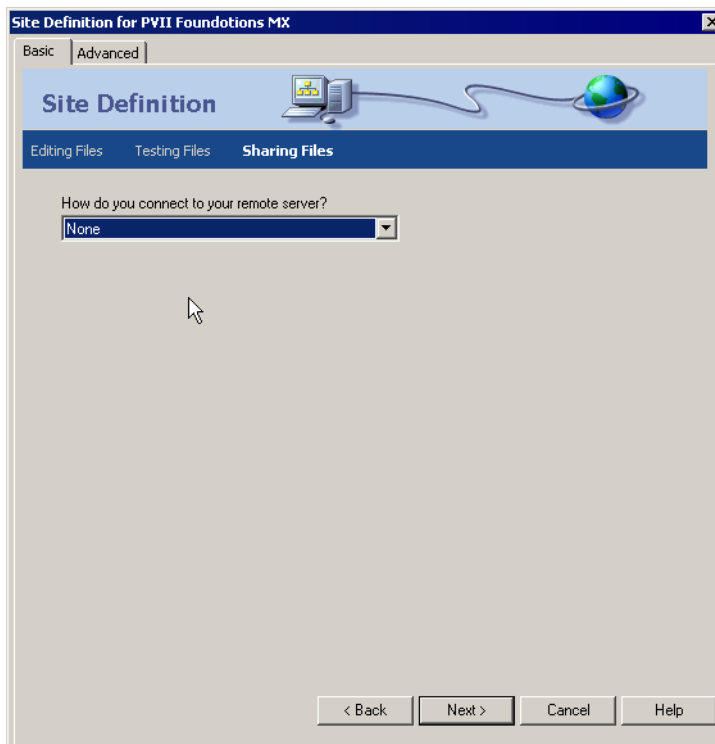


FIGURE 8- SERVER CONNECTION

-Choose **None** from the pop-up menu.

-Click **Next**. You will see a summary of all the settings you have chosen.

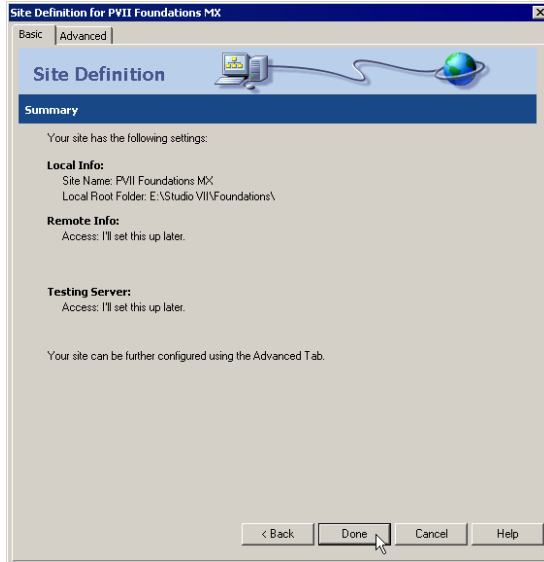


FIGURE 9- SITE SUMMARY

-Click **Done**.

Dreamweaver may prompt you to allow it to create a site cache. This allows Dreamweaver to track and store site information and helps to speed things up.

-Click **OK** to create the cache.

When the Site Definition window closes, you'll see your new site listed in the Site panel.

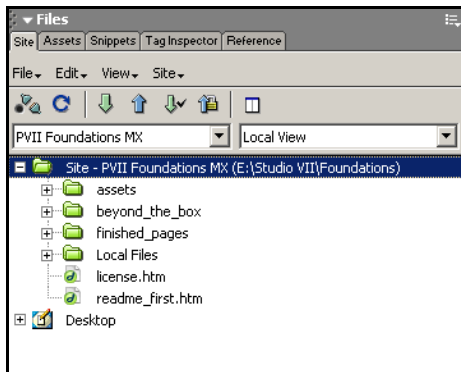


FIGURE 10- THE SITE PANEL

Our Site panel and its parent **Files** Panel Group probably looks a bit different from yours! We'll show you how to manage your panels in Chapter 2.

INSTALL THE FILLER TEXT SNIPPET AND SWAPCLASS EXTENSION

We've included a pre-made Snippet that will allow you to quickly enter filler or placeholder text in your documents. We've also included a JavaScript extension that we'll be using in Chapter 13.

-Choose **Commands**→ **Manage Extensions...**

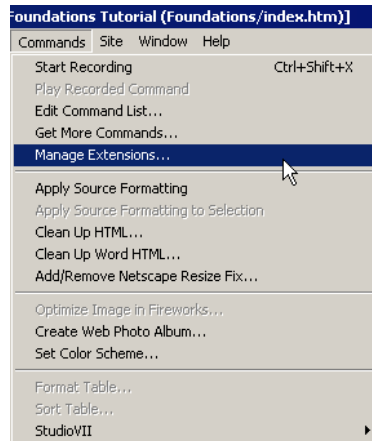


FIGURE 11- OPENING THE EXTENSION MANAGER

The Extension Manager will open.

-Choose **File**→ **Install Extension**

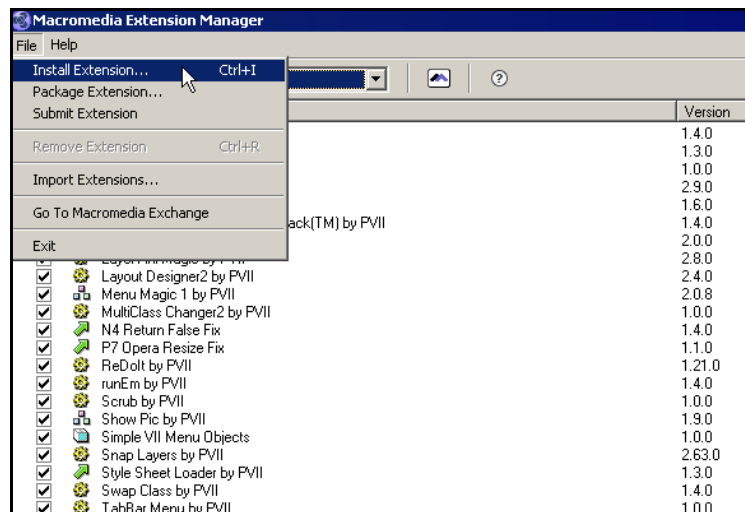


FIGURE 12- THE EXTENSION MANAGER APPLICATION WINDOW

The *Select Extension to Install* window will open.

-Browse to the Local Files/extensions folder of your Foundations site and select **PVII-Filler.mxp**

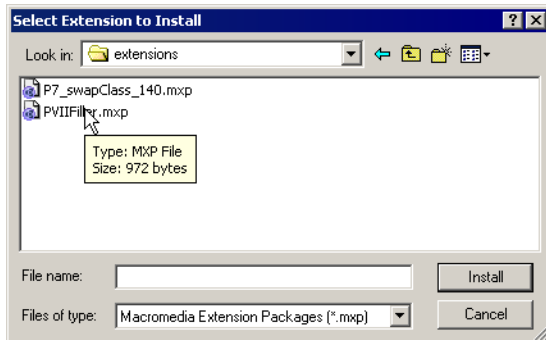


FIGURE 13

-Click **Install**

-Repeat to install **P7_swapClass_140.mxp**

Dreamweaver will inform you that you must restart the program to have access to your new extensions.

-Close the Extension Manager.

-Close Dreamweaver

-Restart Dreamweaver

MAC OSX ISSUE

Sometimes Mac OSX will have file association problems if you attempt to install an extension via the Extension Manager's File Menu. If you experience problems, use Dreamweaver's Site panel to locate the extensions folder and double-click the files. The Extension Manager should open and installation should begin.

Chapter 2—Panel Management

Dreamweaver MX is a very powerful program. Its power and flexibility necessitate many panels and toolbars. This section may prove very useful and offers a practical insight into managing (or taming) your Dreamweaver workspace.

THE MX WORKSPACE

The first time you open Dreamweaver MX, you may be a bit overwhelmed with all of the panels and how everything is arranged. After a little work, your Workspace can look tidier—and very efficient.

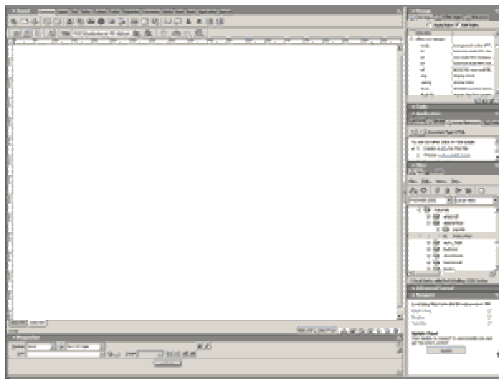


FIGURE 14- THE DEFAULT WORKSPACE

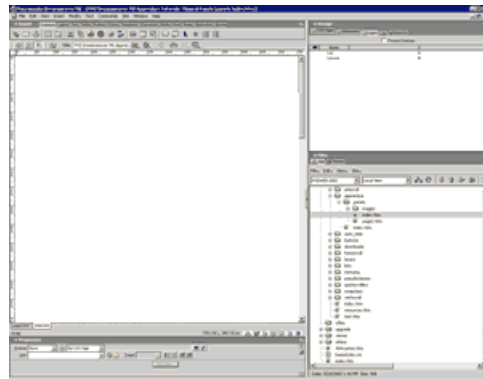


FIGURE 15- THE WORKSPACE AFTER TWEAKING

The manner in which you set up your panels should reflect your own work habits and needs. The techniques you'll learn will enable you to create your ideal Dreamweaver MX Workspace.

BEFORE CONTINUING

If you work at a screen size less than 1024x768, you'll probably be best served by setting your Dreamweaver Workspace to [Dreamweaver 4 mode](#) (floating panels). You can use the same techniques discussed in this tutorial to consolidate and arrange your Panel Groups, even if they are all floating.

MAC USERS

You can use the techniques in this Chapter to regroup your panels, but the new workspace options (docked toolbars on either the right or left) are not available.

CLOSE A PANEL GROUP

If you use Dreamweaver MX for static web page creation, you won't need the Application Panel Group—and although *you* may find it useful, I seldom use the Answers Group. So let's get rid of them (but only if you feel you won't need them!).

Note: If you use the Applications or Answers Panels you can use the techniques explained below to manage them as well.

-Click the white arrow to the left of the word *Application* to expand the panel group.

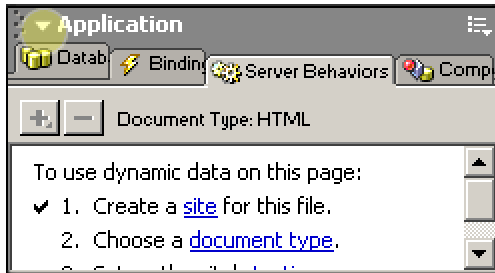


FIGURE 16

-Once the panel is expanded, open the panel Options menu

To open the panel Options menu, click the icon at the panel's top right corner- the one that has 3 squiggly lines and a little down arrow.

-A little pop-up menu will appear. Choose **Close Panel Group** and the Application Panel will disappear.

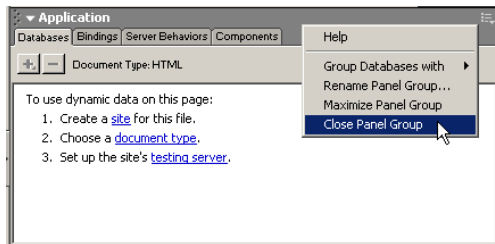


FIGURE 17

Note: Closing a panel is not a permanent action. All panels can be reopened from the Dreamweaver MX "Window" menu.

-Close all panels you *don't* need in this manner.

MODIFY A PANEL GROUP

I like to work with two active panels: Design and Files. My Design panel, however, is modified to reflect my work habits. The default Dreamweaver MX Design Panel Group contains:

- ⇒ CSS Styles
- ⇒ HTML Styles
- ⇒ Behaviors

My Design Panel Group contains:

- ⇒ CSS Styles
- ⇒ Behaviors
- ⇒ Layers

Let's start by ridding the Design Panel group of the HTML Styles panel.

- Expand the **Design panel**.
- Click the **HTML Styles** tab.
- Open the **panel Options Menu**.

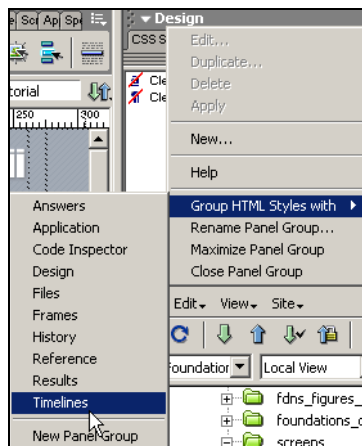


FIGURE 18

- Choose **Group HTML Styles with**
- Click **Timelines**

We've now banished HTML styles to a Panel Group (Timelines) that we never use. But wait! The huge Timelines Panel Group is now open and taking up valuable real estate next to our Properties Inspector. What now? Close it! Open the Timelines Panel menu and choose: **Close Panel Group**

GROUP A NEW PANEL ON AN EXISTING PANEL

Now our Design panel has CSS Styles and Behaviors. We need to add the Layers panel to the group.

-Select **Window**→**Others**→**Layers** (or press F2).

Notice that a new Panel Group has appeared called: *Advanced Layout*. It contains the Layers panel (which we want) and Frames (unwanted).

-Select the **Layers** tab

-Open the group's **panel Options Menu**.

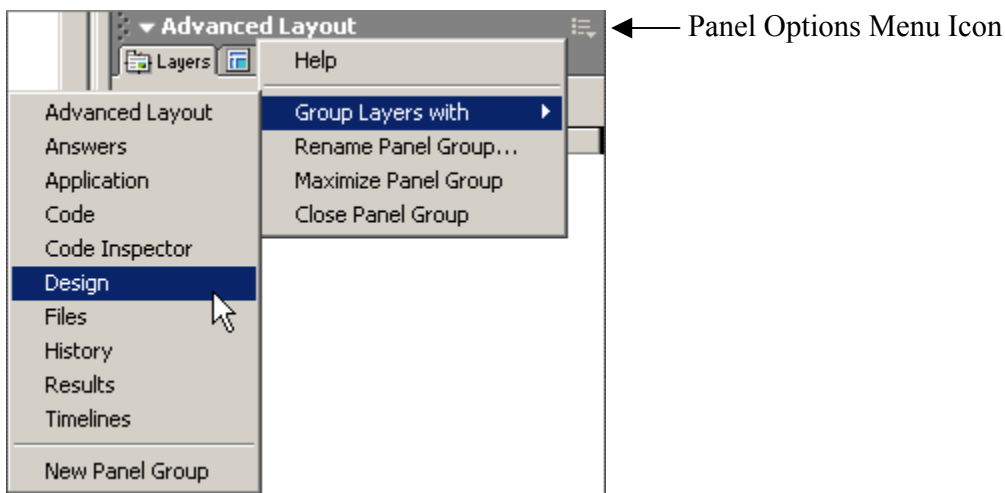


FIGURE 19

-Choose **Group Layers with**, then choose **Design**.

-The Layers panel will appear in the Design Group. Now you can close the Advanced Layout Panel Group (choose *Close Panel Group*), which now contains only the unwanted Frames panel.

Now we have a Design Panel that looks like this:

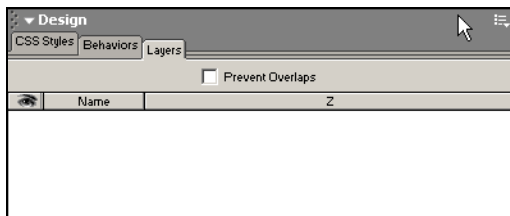


FIGURE 20

ADD THE TAG INSPECTOR TO THE FILES GROUP

-Choose **Window→Tag Inspector** (or press F9)

A new Panel Group called "Code" will open.

-Select the **Tag Inspector** Tab

-Open the **Panel Options Menu**

-Choose **Group Tag Inspector with→Files**

-Close the Code Panel.

ADD THE SNIPPETS PANEL TO THE FILES GROUP

-Open the **Snippets panel** by pressing Shift + F9. The Code panel group will expand.

-Select the *Snippets tab*

-Open the **Panel Options Menu**

-Choose **Group Snippets Panel with→Files**

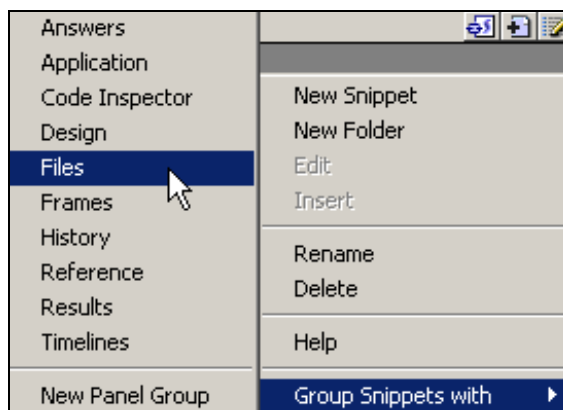


FIGURE 21

-Close the *Code panel group*.

The Files Panel now looks like this:



FIGURE 22

ADJUSTING PANEL GROUP HEIGHT

We've reduced our active Panel Groups to two: Design and Files. We have lots of room now, so let's set the heights of these two Panel Groups optimally.

To change the height of a Panel Group, at least two Groups must be expanded. Position your cursor at the top of the Group you wish to resize. When positioned properly, your cursor will change to a double-pointing vertical arrow.

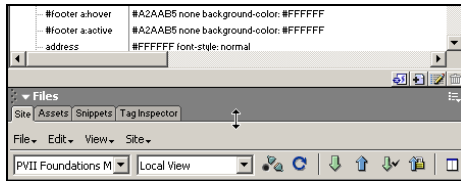


FIGURE 23- RESIZING A PANEL GROUP

Drag up or down to resize the Panel Group.

CHANGING THE PANEL BAR WIDTH

The default Dreamweaver MX Workspace gives you a rather narrow Panel Bar. It almost makes the convenient **Files** panel useless. The PVII web site is 770 pixels wide. To maximize our workspace, we increased our Panel Bar's width until the remaining space to its left was just wide enough to fit our pages.

To widen the Panel Bar, position your cursor over the border between the Panel Bar and the Editing Space.

When your cursor changes to a horizontal double-sided arrow, drag left to widen the Bar. We set our Panel Bar width to achieve an optimal editing space for our web site pages: 770 pixels. Your **Foundations** web pages will also fit comfortably within this space.

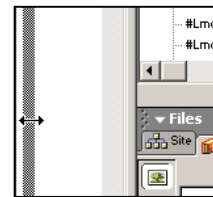


FIGURE 23

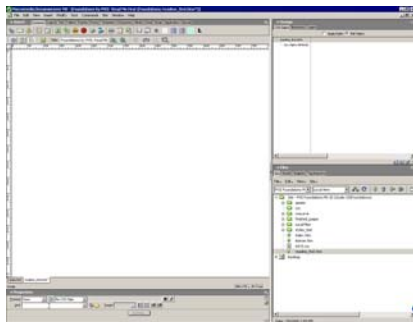


FIGURE 24- OUR FINISHED WORKSPACE

Note: Dreamweaver MX will remember your Panel setup each time the program is started.

Now we have a Workspace perfectly sized for efficient editing and a Panel Bar that's wide enough to be truly useful—but the available real estate will vary depending on your monitor size and screen depth. My screen size is usually set to 1280x1024 pixels. While this screen size is not typical of the average computer user, it is most definitely typical of professional web designers. If your screen is larger or smaller, adjust your Workspace width proportionally.

If you work at a screen size less than 1024x768, you'll probably be best served by setting your Dreamweaver Workspace to Dreamweaver 4 mode (floating panels). You can use the same techniques discussed in this tutorial to consolidate and arrange your Panel Groups, even if they are all floating.

CHANGING TO DREAMWEAVER 4 WORKSPACE (PC ONLY)

- Choose **Edit**→**Preferences**
- Select the *General* Tab
- Click the **Change Workspace** button

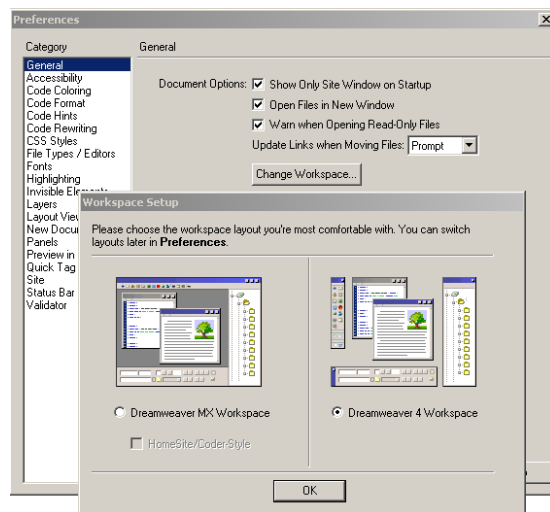


FIGURE 25

- Enable the *Dreamweaver 4 Workspace* radio button.
- Click **OK**. Your changes will appear after restarting Dreamweaver.

Note: Changing your Workspace will reset all Panels to the default for the Workspace mode you choose. You can use the techniques discussed in this tutorial to reconfigure your panels after making a Workspace change.

Chapter 3—Setting up a New Page

Before we begin the building process, we need to create a new document.

SETTING PREFERENCES

When you started Dreamweaver, a blank document was automatically created. Before moving on, please close that document now (choose **File**→ **Close**).

If you find Dreamweaver's creation of a blank document at startup annoying, here's how to prevent it in the future:

-Choose **Edit**→ **Preferences**

The Preferences window will open.

-If it's not already selected, select the *General* category

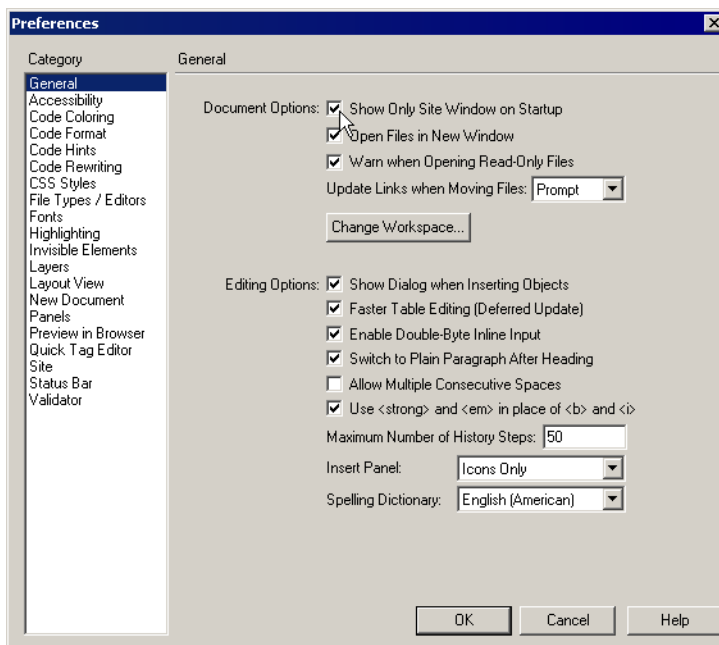


FIGURE 26- PLACE A CHECK IN THE TOP BOX

-Place a check in the box labeled: **Show Only Site Window on Startup**

-Click **OK**

CREATE A NEW DOCUMENT

Choosing **File→New...** from the main Dreamweaver menu (at the very top of your window), will open a dialog window called: **New Document**. It sounds innocent and clear enough—but it's not as straightforward a process as you might think. You'll be presented with a rather overwhelming choice of sample pages and preformatted pages that are (in our opinion) a bit *underwhelming*. If you create a document using this window, the page you create will be unsaved. When you attempt to save this page, the odds are that Dreamweaver will have no idea that you want to save the file in the same site you have open. Here is a much better way to create a new file:

-Open the **Site panel** (if it isn't already open)

-**Select** (highlight) the site root folder¹—the one at the very top of the list

-Choose **File→New File** (I know what you're thinking, but this File menu is totally different from the main File menu)

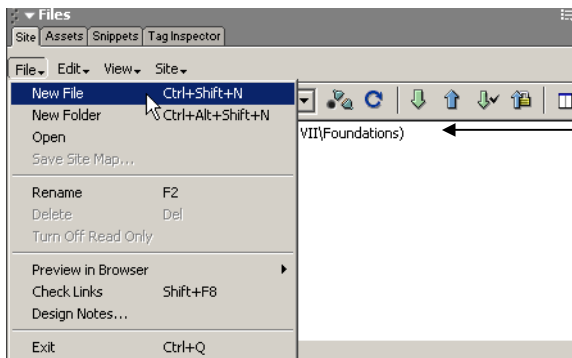


FIGURE 27

A new file called: **untitled.htm** will appear in the Site window ready to edit.

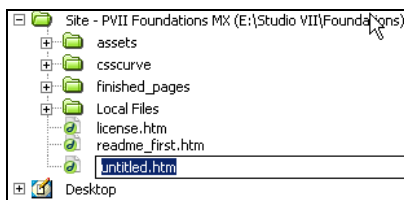


FIGURE 28

-Type over (replace) **untitled.htm** with **index.htm**

-Double-click the file icon just to the left of **index.htm** to open it

¹ When you create a new file from within the Site panel, the new file will be placed within whichever folder you happen to have selected. For instance, if you want to create a new file in the `finished_pages` folder, select (highlight) that folder in the panel before you create the new file. In addition to using the panel's File Menu, you can right-click on a folder to open a context menu which includes the New File command. **Mac users must Ctrl-click the root folder because Dreamweaver's Mac Site panel does not have its own File menu.**

ADDING A PAGE TITLE

With modern Search Engines (like Google), properly phrased page titles are very important. A good title is concise and descriptive:

```
<title>PVII Dreamweaver Extensions, Tutorials, Templates, and FAQs</title>
```

A poor title is confusing or silly:

```
<title>::::::::::The best Site in the World for Woozles::::::::::</title>
```

This title would be a good choice:

```
<title>My Foundations Tutorial</title>
```

-Enter your title into the *Title* box on Dreamweaver's toolbar:

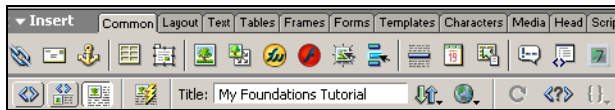


FIGURE 29

Chapter 4—Adding a Doctype & a Snippet

Let's have a look at the default HTML that Dreamweaver uses for a new page.

-Click the Code View icon

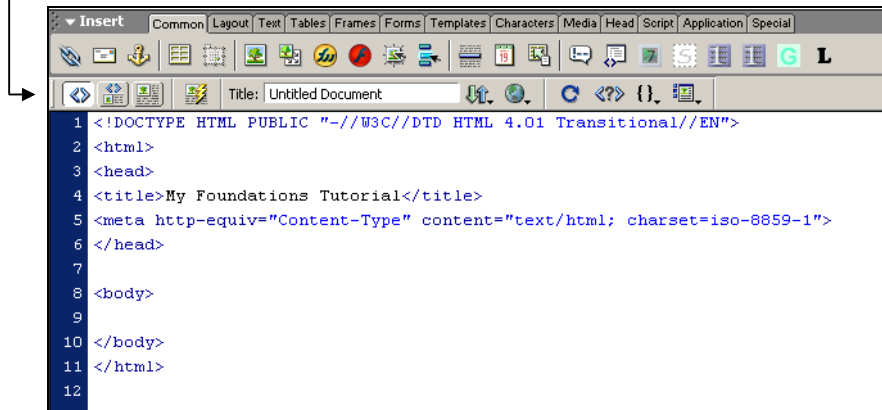


FIGURE 30

THE DOCTYPE

The first line is the DOCTYPE¹ (Document Type). Browsers use the doctype to determine how strictly to interpret certain parts of the code. This is especially important for pages that contain JavaScript and/or CSS. For some reason, Macromedia chose not to include a full doctype. A DOCTYPE that includes a full URL (a complete web address) tells modern browsers to render your page in standards-compliant mode thus treating your HTML, CSS, and JavaScript in a predictable way.

By default, Dreamweaver inserts² an incomplete DOCTYPE (no URL), which puts modern browsers into *Quirks* mode. The browser will attempt to analyze your page in a backwards-compatible way—rendering CSS and HTML as if for an older browser. Obviously, this is not a good thing.

Let's fix that now by correcting the DOCTYPE.

¹ Doctype is short for Document Type, which in turn is short for: Document Type Declaration (DTD). We are using an HTML 4.01 *Transitional* Doctype. The “Transitional” part means that certain tag properties are allowed in order to achieve compatibility with older (version 4) browsers. You can copy the Doctype from within Acrobat Reader by clicking the “T” icon on Acrobat’s toolbar. If the icon is not present, select **Window**→**Toolbars**→**Basic Tools** (Acrobat 5 or higher).

² If you set up a Server Model (Cold Fusion, ASP, etc) in your Site Definition, Dreamweaver will not enter any Doctype at all.

It would be good if we could store our new DOCTYPE inside Dreamweaver so it could be easily accessed whenever needed. Dreamweaver MX has a feature called “Snippets” that is tailor-made for this task.

Copy the text below:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Note: If you are not able to copy the text in your Acrobat reader, then open one of the pages in the finished_pages folder, go into code view, and copy the 2-line doctype.

CREATING A SNIPPET

-Open the Snippets panel (Shift + F9) or click the Snippets tab on your Files panel group. Click the New Folder icon on the panel’s lower right border. It’s the first icon in the row and resembles a file folder.

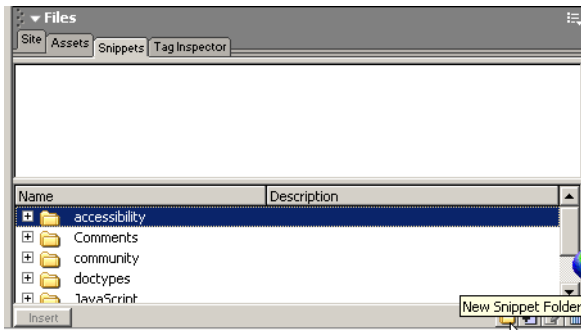


FIGURE 31

If you’re wondering why we have different Snippets folders than Dreamweaver MX installs by default, it’s because we removed the ones we don’t use and added some new ones.

-A new untitled folder will appear¹. Overwrite the word *untitled* with **doctype**.

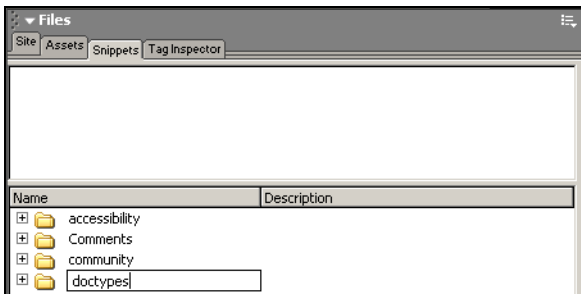


FIGURE 32- WE’VE NAMED OUR NEW FOLDER

¹ We want the new folder to be in the root of your Snippets directory. Make sure that no folder in the Snippets panel is selected (highlighted). If one of the folders is highlighted (even if the highlight appears grayed out), click the **Name Title Bar** above the folders list in the Snippets panel twice to deselect the folder before creating a new folder.

- With the doctype folder selected (highlighted), **click** the *New Snippet* icon (it's the one next to the New Folder icon) on the bottom right border of the panel.
- The Snippet window will open.
- In the Name box, type: **401 Transitional**
- In the Description Box, type: **Complete 4.01 transitional doctype**
- For Snippet Type, enable (click) the *Insert Block* radio button
- Paste your copied doctype into the *Insert Code* box

Mac OSX users will have to type the Snippet code, as Dreamweaver for OSX seems not to have support for pasting in the Snippets dialog.

- For Preview Type, enable the *Code* radio button

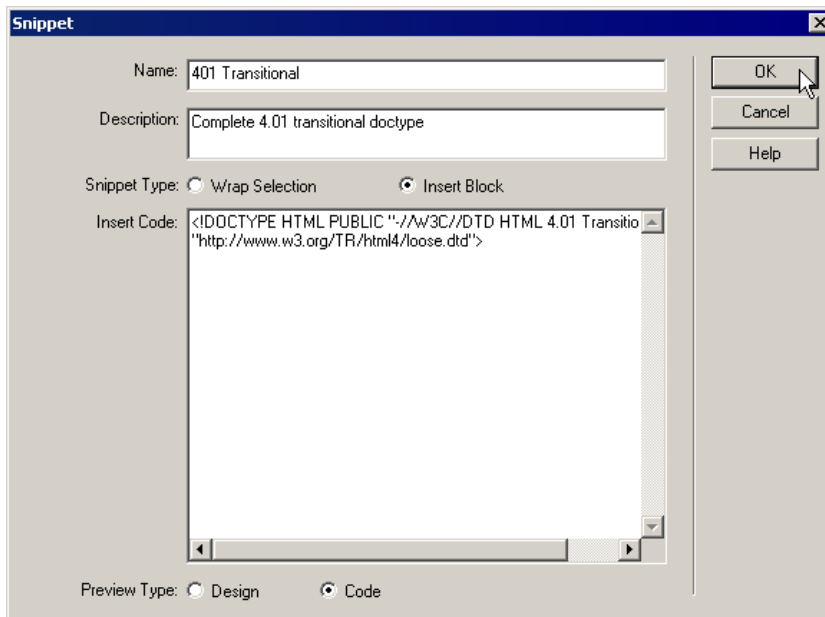


FIGURE 33

- Click **OK**

INSERTING A SNIPPET

Now that we've created our Snippet, let's learn how to add a code Snippet to our page. Index.htm should still be open. If it's not, please reopen it now.

-Switch to **Code View** by clicking the Code View icon on the main toolbar or by choosing View→ Code.

-Delete the existing Doctype, then position your cursor on the now-blank line at the very top of the code window, just above the opening <html> tag¹.

```
1 |
2 <html>
3 <head>
```

FIGURE 34- CURSOR ON LINE 1

-Open the Snippets panel

-Click the **plus (+)** sign to the left of the new doctypes folder you created

-Select the *401 transitional doctype* Snippet

-Click the **Insert** button on the lower left edge of the Snippets panel

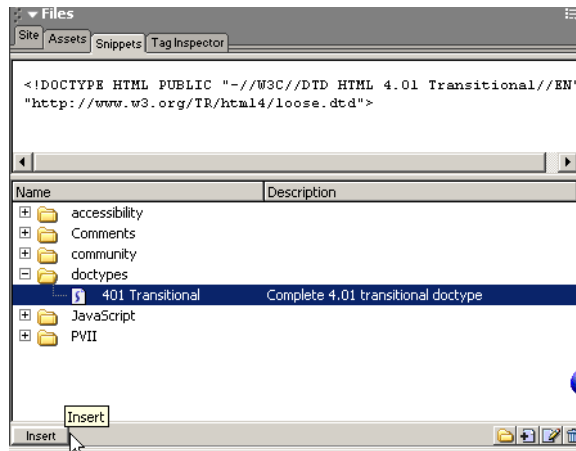


FIGURE 35- INSERTING THE SNIPPET

¹ A Doctype must go at the top of the page as its purpose is to inform browsers how the page that follows should be interpreted.

The code (doctype) will appear on the new line you created in your source code.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>My Foundations Tutorial</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>

</body>
</html>
```

Congratulations! Now you're a *Snippet Maker first class*.

For more information on this powerful productivity tool, see your Dreamweaver MX documentation or tune in to the PVII Webdev Newsgroup¹ for cogent and informative discussions on all things Dreamweaver.

SAVE YOUR PAGE

Save your page by choosing File→ Save or by pressing Ctrl + S

**As you go through the lessons—
please remember to save your work often.**

¹ Our news server is: forums.projectseven.com

Chapter 5—Dreamweaver Accessibility Aids

**“ THE POWER OF THE WEB IS IN ITS UNIVERSALITY.
ACCESS BY EVERYONE REGARDLESS OF
DISABILITY IS AN ESSENTIAL ASPECT.” —TIM BERNERS-LEE**

Nothing is ever perfect, but we’re going to make our web pages as accessible as we possibly can. We’ll point you along the right path to good basic accessibility, but keep in mind that there is no single standard.

Depending upon the specific project you are working on, you may need to adhere to one or more specific accessibility standards. A good example would be [Section 508](http://www.section508.gov/) (<http://www.section508.gov/>). You should also keep abreast of the [W3C](http://www.w3.org/WAI/) (<http://www.w3.org/WAI/>) accessibility specifications and guidelines. The W3C site also has a fine listing of reference resources.

ENABLING DREAMWEAVER MX ACCESSIBILITY FEATURES

-Choose Edit→ Preferences, and then select the Accessibility category

-Put a mark in all five check boxes at the top of the window. You can leave *Make Dreamweaver use Large Fonts* unchecked

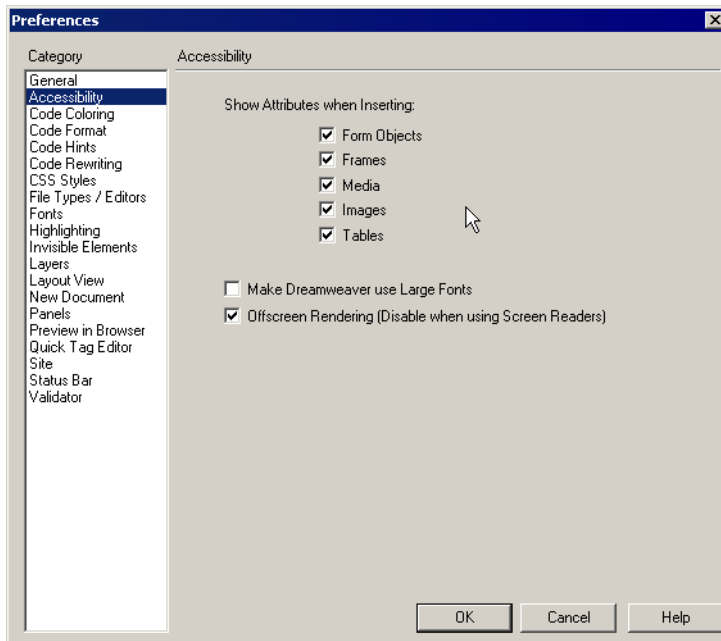


FIGURE 36

-Click **OK**

Dreamweaver will now prompt you to enter accessibility attributes when you insert any of the elements listed next to the check boxes (Figure 38). As an example, whenever you insert an image you will get this prompt:

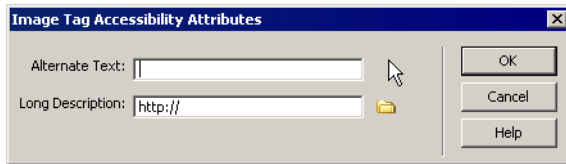


FIGURE 37

The prompt is a reminder that Alternate Text¹ is required and that a URL to a file that contains an expanded description of a complicated image (a map, for example) is required.

Now let's get back to our index.htm page.

DECLARING THE DEFAULT PAGE LANGUAGE

Software that audibly “reads” pages needs to know the default language used. Therefore, we will modify our opening HTML tag to note that our page is English.

-Open Code View

-Stick your cursor inside the opening `<html>` tag, just before the closing angle bracket, and press your spacebar. A little box will pop up containing *hints* of the possible attributes you can use on this tag. You've discovered Dreamweaver Code Hints!

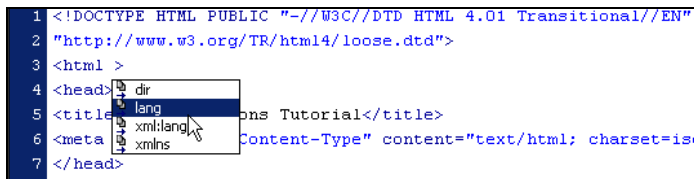


FIGURE 38

-Double-click “lang” and the attribute will be inserted. Your cursor will be positioned between the opening and closing quotation marks

-Type the letters “en”² inside the quotes for **English**

```
<html lang="en">
```

¹ Alternate Text refers to the Alt property for the `<img>` tag: `<img alt="This is Image Description">`. Alt text is used by browsers that cannot display images or as placeholder text while the image is downloading. Some browsers pop up the Alt text in a yellow box when you hover your mouse over an image (which is an optional browser feature and not a required standard).

² Examples of other Language codes are: fr (French), de (German), it (Italian), nl (Dutch), el (Greek), es (Spanish), pt (Portuguese), ar (Arabic), he (Hebrew), ru (Russian), zh (Chinese), ja (Japanese), hi (Hindi), ur (Urdu), and sa (Sanskrit).

Chapter 6—Planning the Page Layout

“I LOVE IT WHEN A PLAN COMES TOGETHER.”

—HANNIBAL SMITH (THE A TEAM)

Having a visual plan makes web page building a much more pleasant and organized task. Before this book saw its first word, I spent time sketching an interface. I started with pencil and paper and ended up in Fireworks.

With sketch in hand, it was decision time. Having a logo, a main content area, and a sidebar is straightforward. Creating the curved borders around the main content area is not ☺. I decided that if you opened this book and discovered that learning its lessons would enable you to build something that was not only structurally sound, but pleasing to look at, then you'd be happier and a whole lot more eager to learn. Therefore, the curve was *in*.

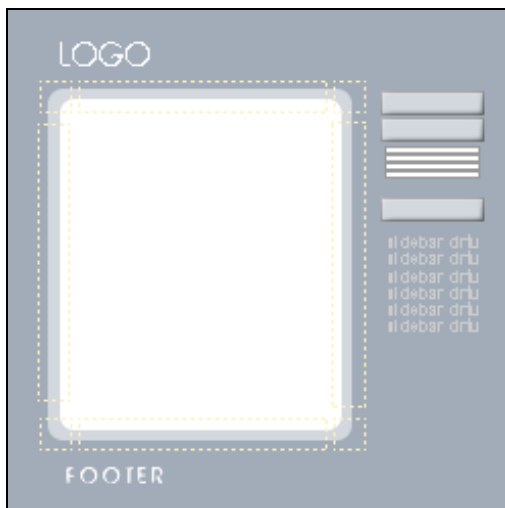


FIGURE 39- THE SKETCH IN FIREWORKS

THE DESIGN STRATEGY

The interface is sketched out and the curve is in. Now we need a Design Strategy...

Our sketch shows a navigation bar that has buttons for the main links and for the sub links. We could take a traditional path and use images, but wouldn't it be cool if we could make a navigation bar that looked like image buttons, but was just text? We could use CSS to make it happen. We could even use CSS to use a single image applied as a background to all of the main links. Imagine that... a single image downloaded a single time making an unlimited number of text links look like *images*.

Now what about **browser support**? This is where a web designer needs to make some choices. So let's see what our choices are:

Using a table to structure a rounded box will work in all version 4 and higher browsers. Making text links look and act like images requires CSS that will work in all version 5 and higher browsers. Version 4 browsers will not be able to render this level of CSS, but we'll make it so that these older browsers see nicely colored and styled text links. The technique is so cool and will save us so much bandwidth, and you'll learn so much—that we just have to do it! Here's how the finished page will look in version 5 and higher browsers...



FIGURE 40- THE FINISHED PAGE IN MSIE6

Here's how the finished page will look in version 4 browsers...



FIGURE 41: THE FINISHED PAGE IN NETSCAPE NAVIGATOR 4

All things considered, the page looks pretty darn good in Navigator 4. The trick is to use two style sheets: one for version 5 and higher browsers, the other for version 4 browsers. Each contains styles that can be used safely and effectively in their target browsers.

RECESS TIME

At this point, we'll let you go off and test the pages we have included for you in the **finished_pages** folder of your Foundations site. They are:

dusty.htm, **purple.htm**, **blue.htm**, and **yellow.htm**

Preview them in all of your browsers, get a feel for how they are set up, and note how each browser displays them. We'll meet you back here in a moment.

Take your time!

ASSEMBLING AN ASSETS LIST

Having planned a design strategy, we'll proceed with the task of identifying the image assets needed to construct our page:

- We need a background image for the links.
- We need a logo.
- We need to construct the curves and borders that will serve to create our rounded main content box.
- And I think we should make a chic little background tile to give our page some texture.
- A background tile for the main content table cell.
- We'll also need a transparent spacer image for making spaces and holding them open.

The next logical step is to create the images.

-Open up **Fireworks MX**—we have some images to design.

Chapter 7—Creating the Foundations Images

“THERE IS NOTHING REMARKABLE ABOUT IT.
ALL ONE HAS TO DO IS HIT THE RIGHT KEYS AT THE
RIGHT TIME AND THE INSTRUMENT PLAYS ITSELF.”—J.S. BACH

We’ll start at the beginning—by creating a new document.

CREATE A NEW DOCUMENT AND CONFIGURE THE CANVAS

-Choose **File**→**New**

In the New Document dialog box, we’ll set the canvas size, resolution, and color.

-Set *Width*: **600 Pixels**

-Set *Height*: **600 Pixels**

-Set *Resolution*: **96 Pixels/Inch**¹

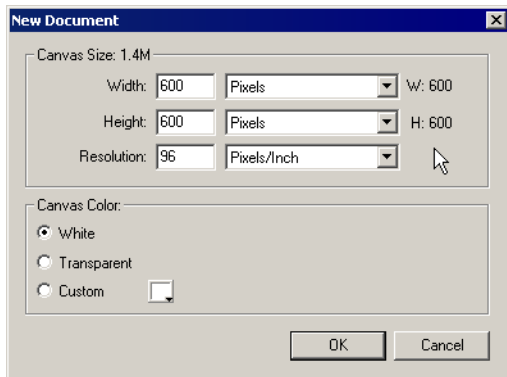


FIGURE 42

-Select the *Canvas Color*: **Custom** radio button

-Click the **color square** next to the Custom button

-Click on the **right pointing arrow** (top, right edge of the pop-up) to open its menu

¹ When working with web images, there are basically two choices we can make: 96ppi or 72ppi. 96ppi is the default screen resolution for Windows operating systems and also the default image rendering resolution for most modern browsers on both Windows and Mac systems. Choosing either one is more or less a personal preference and will not affect your images in any measurable way.

-If there is a check mark next to *Snap to Web Safe*¹ (as shown in the figure below), click to disable it². If the check mark is not there, just press your **ESC** key and move on to the next step without making any changes.

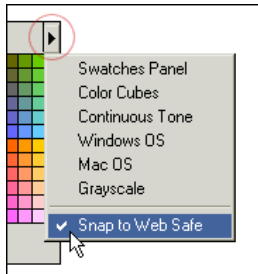
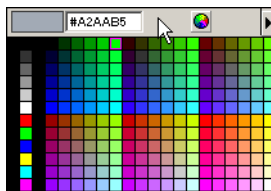


FIGURE 43

-Double-click to highlight the existing color code in the color entry box and Type **#A2AAB5** into the box



TIP: You can also change the existing value in the swatches pop-up by slowly dragging your cursor over it from left to right to highlight it, then typing in a new value.

FIGURE 44

-Press **Enter** to accept the new color and close the swatches pop-up³

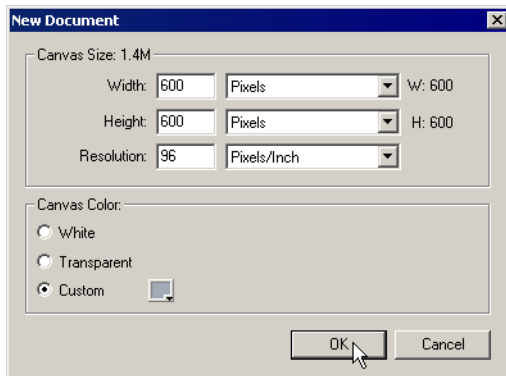


FIGURE 45

-Click **OK**

¹ Fireworks MX is preset to operate with Web safe colors. We tend to use web safe colors when working with complex JPEGs that need to blend into background colors that are neither pure white nor pure black. Working with GIFs, or with JPEGs that have clearly defined borders, really should not cause us to limit our palette.

² You won't actually see it, but when you disable Snap to Web Safe, the check mark is removed. Should you open the Swatches pop-up menu again, you'll see that the check has been removed.

³ A **Swatches Pop-up** is a color palette containing individual cubes from which you can choose (by clicking one).

SAVE YOUR DOCUMENT

Although there's nothing on the canvas yet, now is a good time to save your document for the first time.

-Choose **File**→**Save As**

-Browse to the location of the Foundations Fireworks folder (..\Foundations\Local Files\Fireworks)

-Name your new file **myFoundations**

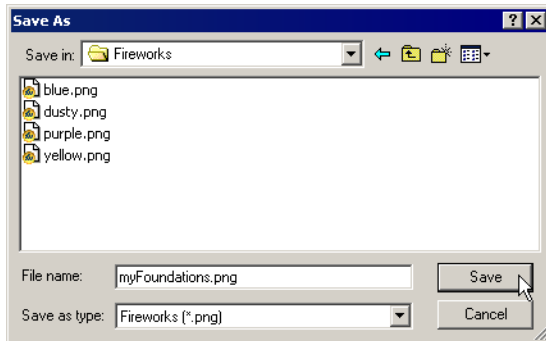


FIGURE 46

-Click on the **Save** button

Note: As you proceed through this Chapter, you may find it useful to open and study the included Fireworks document: **blue.png**

Please remember to save your file often as you work!

RULERS AND PIXELS AND GUIDES

When they are visible, rulers appear along the top and left sides of the document window. The rulers always measure in pixels instead of DPI (dots per inch), lines, centimeters, rods or furlongs¹. Why is that? Fireworks is for making art with pixels, not paper—and pixels are the only relevant unit of measure for Web images.

Guides are lines that float over the canvas. You can add and remove guides, and you can lock guides into place to avoid moving them by mistake. Rulers and Guides assist you to lay out objects on the canvas with precision.

-Choose **View→Rulers**

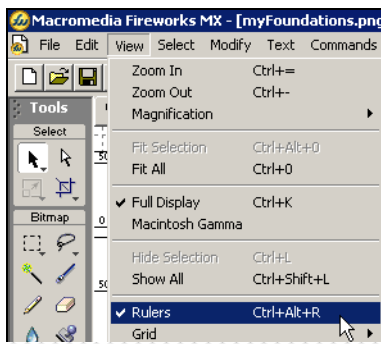


FIGURE 47

-Choose **View→Guides→Show Guides**.

It's usually handy to have **Snap to Guides** turned on as well.

¹ A furlong is equal to 220 yards, or approximately 200 meters. Use this measure only for very large web pages. For normal pages, we highly recommend sticking with pixels.

SETTING UP THE FIREWORKS MX WORKSPACE

Just like Dreamweaver, Fireworks MX has a customizable user interface. Unlike Dreamweaver, not all of the Fireworks panels and toolbars open by default. Windows users can add *Main* and *Modify* toolbars, for example.

-Choose **Window**→ **Toolbars**→ **Main**

-Choose **Window**→ **Toolbars**→ **Modify**

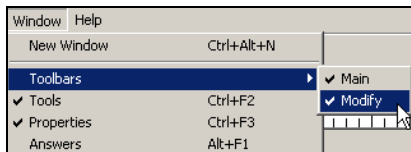


FIGURE 48

Two especially useful panels to add to the default set are the Info and Align panels. As well as showing the dimensions of selected objects, the Info panel provides live feedback of the cursor coordinates and local color in Hex numbers as you move the cursor across the canvas. The Align panel is new to Fireworks MX and enables you to align selected objects to each other or to the canvas.

TO OPEN THE INFO AND ALIGN PANELS:

-Choose **Window**→ **Info**

Note that the Info panel may not open to its full size. We'll fix that in a later step.

-Choose **Window**→ **Align**

It's useful to group the new set of panels together before inserting them into the docking area. To group the panels, click on the *Expander* icon at the top right of the Info panel to open the panel's *Options* menu. When the menu pops up:

-Choose **Group Info with**→ **Align**

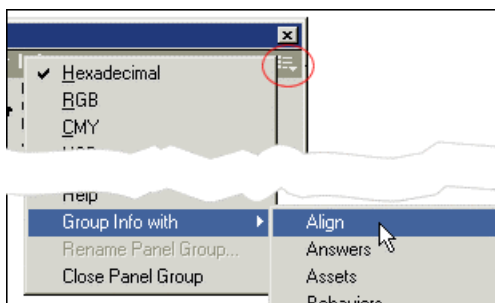


FIGURE 49

The Info panel groups with the Align panel and expands to its full size.

Now that you have the Info and Align panels grouped, you need to decide where to dock your new panel group. My personal preference is to place it between the Optimize and Layers panels. If you'd like to try it out, click on the panel gripper (the cluster of dots at the far left of the panel title bar) and drag slowly toward the docking area.

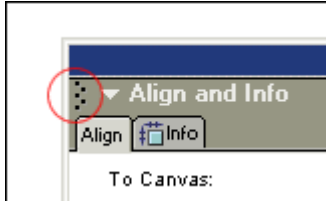


FIGURE 50

As you drag the panel group toward the divider bar that separates the workspace from the docking area, a bright, turquoise colored placement preview rectangle appears. When you reach the area between the title bars of the Optimize and Layers panels, the turquoise (gray in OSX) rectangle changes to a thick line.



FIGURE 51

As soon as you see the line form, release your cursor.

One other customization we'll suggest is grouping the Frames and Layers panels together.

You'll find other customization options, such as *Show Tab icons*, by choosing **Edit→Preferences** (On Mac OS X, choose **Fireworks→Preferences**).

To save your custom layout:

-Choose **Commands→Panel Layout Sets→Save Panel Layout**

-Give the panel layout a name and click **OK**

Important

If the content of one or two of your panels should disappear (or refuse to load) the next time you start Fireworks, close Fireworks down and then use your Operating System's file search feature to locate the following two files:

Fireworks MX Preferences.txt
Fireworks.ini

Delete the two files and restart the program.

Now let's create some images!

DRAWING THE FOUNDATIONS IMAGES

The images we'll be creating for Foundations are:

- A button background
- Table corners and borders
- A logo comprised of three elements:
 1. Header text
 2. Home icon
 3. Subheading text
- A shim
- A background for the main table cell
- A background tile for our Foundations Web page

CREATING THE BUTTON BACKGROUND

The button background is the gradient that will appear for most viewers (those using a version 5 or higher browser). Until popular browsers and CSS have advanced to the level of enabling coded gradient backgrounds for tables we'll need to use images for this purpose.

A warning to those of you used to working exclusively with Bitmap editing software, like Photoshop or Paint Shop Pro: avoid the temptation to use the Marquee tool to draw a rectangle! 

While Fireworks does allow you to work that way, it's far more powerful when used in vector editing mode. It's in Fireworks vector mode that your images remain editable throughout every stage of development. We'll be working in vector mode for this project.

-Select the Rectangle tool 

-Click on the canvas and drag diagonally to create a **178 pixel wide** by **96 pixel high** rectangle. Don't worry if you can't get the size exactly right. You can adjust the size by changing the dimensions in the Property inspector.

TO ALTER THE DIMENSIONS:

-Type the width in *W box* in the lower left-hand corner of the Property inspector

-Type the height in the *H box* in the lower left-hand corner of the Property inspector

-Press **Enter**

TIP: You can also change the dimensions and position of an object in the Info panel.

With the rectangle still selected, use the Property inspector to:

-Change the Stroke category¹ to **None**

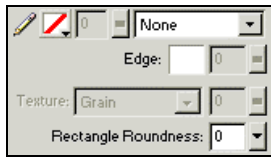


FIGURE 52

-Change the Fill category from *Solid* to **Linear**

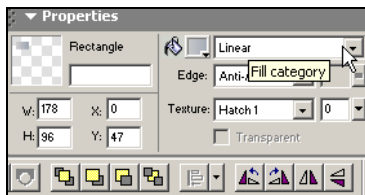


FIGURE 53

-Click inside the Fill Color box in the Property inspector to open the Edit Gradient pop-up window.

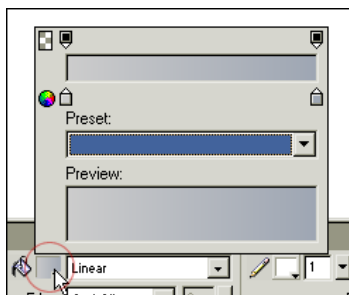


FIGURE 54

¹ Fireworks has a rather odd naming convention. Many of the property boxes on the *Property* inspector are defined as *Categories*, while they really would seem to be **properties**.

-In the Edit Gradient pop-up window, click on the left, lower color chip to open a swatches pop-up¹

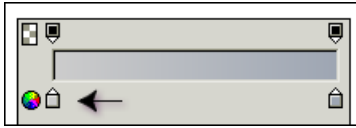


FIGURE 55

-Use the built-in color picker to select the #CCCCCC (light gray) color swatch.

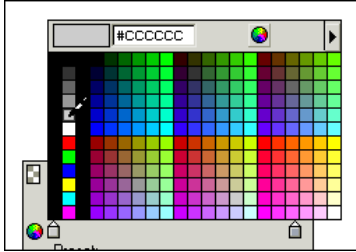


FIGURE 56

The swatches pop-up will close as soon as you pick the color.

The swatches pop-up interface is used in all color selection operations in Fireworks.

-Click on the right, lower color chip to open the swatches pop-up again.

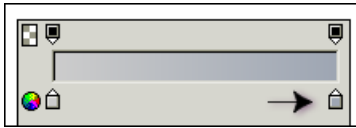


FIGURE 57

-Use the swatches pop-up color picker to click on an empty spot on the canvas²

-When the swatches pop-up closes, click on an empty spot on the canvas to close the Edit Gradient pop-up window

TIP: Choosing the Linear fill option automatically changes the existing Fill Color to a gradient that blends the currently selected Stroke and Fill Colors together. If you like, you can set up the two colors for the gradient by first selecting in the Stroke and Fill Color boxes on the Tools panel

¹ Make sure you click the existing color chip. Do not click between the two gradient color chips or Fireworks will insert a new color chip where you click. If you do accidentally create a new color chip, drag it all the way up and off the panel.

² The Fireworks color picker can pick colors from any pixel on the canvas, from another open document, or even from the Fireworks application window itself—toolbars, menus, and all.

Since we will create several more images, let's move this one out of the way. You'll do this by changing the values in the *X* and *Y* boxes in the lower left-hand corner of the Property inspector.

-Enter **0** in the *X box* to set the horizontal distance of the rectangle from the left edge of the canvas

-Enter **47** in the *Y box* to set the vertical distance of the rectangle from the top edge of the canvas.

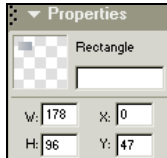


FIGURE 58

-Press **Enter** to set the position

THE TABLE CORNERS AND BORDERS IMAGES

One of the more frequently asked questions in the Dreamweaver forum is how to create table borders with sleek, rounded corners. The answer is to build them with images, of course. Here you'll go beyond the usual technique and learn to create a handsome, double border to wind around the main content area of your Foundations Web pages. We'll begin by drawing some rounded rectangles.

-Click and hold on the *Rectangle* tool until the shape tool menu flies out

-Select the **Rounded Rectangle** tool

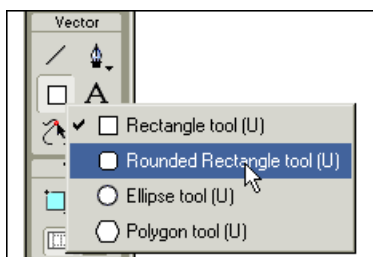


FIGURE 59

-Drag out a rectangle to make a **48x48** pixel square¹.

-Select the pointer tool

The Rounded Rectangle tool creates a 30° corner radius by default. By coincidence, that's just the radius we want our corners to be.

-Correct the size, if necessary, by entering the new dimensions into the *W* and *H* boxes in the Info panel (or Property inspector).

-Change the *Fill* category in the Property inspector back to **Solid**.

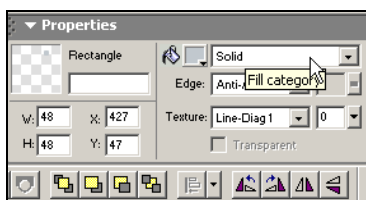


FIGURE 60

-Click in the *Fill Color* box on the Property inspector (or on the Tools panel)

-Double-click in the text box at the top of the swatches pop-up and enter the Hex number **#C1C8CE**

-Press **Enter** to close the swatches pop-up

¹ If all four sides of a rectangle are the same length, it is said to be square (just in case you were wondering ☺)

Your image should look like this one:

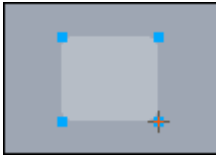


FIGURE 61

You'll be creating the second color of the border by making a clone of the first rectangle and then re-coloring it and changing its dimensions. Cloning in Fireworks creates a copy of an object—be it a vector shape, a text object, or an imported bitmap—directly on top of the original object.

-With the first rectangle selected, choose **Edit→Clone**

It will appear as if nothing happened because the clone is the same color as the original image it was cloned from. Proceed to the next step to change the clone's fill and you will see it.

-Click in the *Fill Color* box of the Property inspector to open the swatches pop-up

-Set the fill to **white** (#FFFFFF)

It's obvious that something has to change if you want to make a two-tone border out of two images stacked on top of each other. We need to shrink the white rectangle. Instead of entering specific values in the Property inspector to transform an object, you can let Fireworks do the scaling for you by using the *Numeric Transform* feature.

-Choose **Modify→Transform→Numeric Transform**.

The Numeric Transform dialog box opens.

-Choose the *Scale* option

-Enter **70** in the width box

-Leave *Scale Attributes* and *Constrain Proportions* checked

The *Scale Attributes* option transforms the fill, stroke, and effects of the object with the object itself. The *Constrain Proportions* option preserves the horizontal and vertical proportions when scaling or resizing a selection.

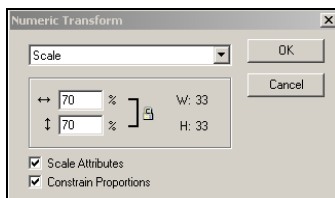


FIGURE 62

-Click **OK**

Fireworks resizes the clone to 34 x 34¹ pixels and leaves it sitting squarely in the middle of the original rectangle.

To finish creating the Foundations *table corners and borders* image, we'll make one more rectangle on top of the other two, then group all three.

-Clone the smaller (white) rectangle.

-Choose **Modify**→ **Transform**→ **Numeric Transform**.

The Numeric Transform dialog box opens.

-Choose the *Scale* option

-Enter **70** in the width box (it should have carried over from your last session).

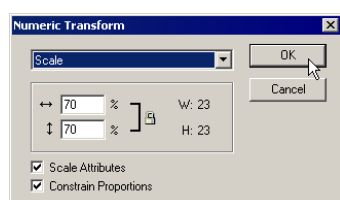


FIGURE 63

Fireworks resizes the clone to 24 x 24² pixels and leaves it sitting squarely in the middle of the *white* square.

To smooth jagged edges that may occur on curved objects, Fireworks *Anti-Aliases* all vectors by default. Anti-Aliasing works by adding extra pixels to an object's edges. The extra pixels render somewhere between the color of the object and the color of the canvas, softly blending the colors together. Turning Anti-Aliasing off will make the edges of your 24 x 24 pixel rectangle³ appear crisp and clean, which will help us later when we slice the image. To turn off Anti-Aliasing:

-Locate the *Edge* selector box in the Property inspector and change the Edge type from Anti-Alias to **Hard**



FIGURE 64

¹ Although the Transform box indicates that the rectangle will be resized to 33x33, Fireworks actually resizes it to 34x34 as reported in the Property inspector. Fireworks is simply rounding a fractional number. 70% of 48 is actually 33.6.

² The Transform box indicates that the rectangle will be resized to 23x23. The actual size is 24x24 as reported in the Property inspector. Again, Fireworks is simply rounding a fractional number.

³ Only the innermost rectangle needs to have Anti-Aliasing turned off. The outer rectangles will comprise our actual borders and we want those to be as smooth as possible.

To finish the table corners and borders image:

- Click inside the *color box* on Property inspector to open a swatches pop-up.
- Click on the canvas with the color picker to fill the rectangle with the same color you used for the canvas (#A2AAB5)



FIGURE 65

- Switch to the *Pointer Tool* before continuing
- Place your cursor on the canvas above and to the left of the top left corner of the stacked rectangles. Drag diagonally to a point past their bottom right. This draws a cursor box around the rectangle group, leaving all three rectangles selected once you release your mouse button.

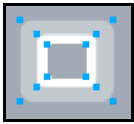


FIGURE 66

- Choose **Modify**→ **Group** and move it to a convenient spot towards the top right corner of your document—such as **X: 427** and **Y: 47**

There are other ways to make our round border images. *There are always other ways!* For instance, you could employ mathematics to extrapolate the size and number of rectangles needed, then simply draw them on the canvas. We've chosen a method that should be manageable by most users from beginner to advanced levels.

CREATING THE LOGO

The word *logo* derives from the Greek word *Logos*—meaning word, thought, or speech. We wanted the Foundations logo to evoke strength and stability, *two components upon which to build a good foundation*. We'll use Impact¹, a heavy, condensed, and strong looking typeface for the main element of the logo—the header text.

-Click on the Text tool to select it 

In the Property inspector:

-Select the **Impact** font.

-Set the *size* to **28** points

-Choose **Smooth Anti-Alias**

-Make sure that *Auto Kern* is **checked**

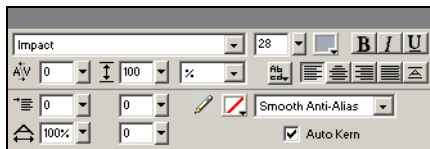


FIGURE 67

-Click on an empty spot on the canvas

-Type the words **foundations by pvii** (in lowercase letters)

-Click on the Pointer tool to exit text editing mode

Let's change the color to a gradient.

¹ The fonts you'll be using are Impact and Verdana. Both install with most modern Windows and Mac systems: Win 98, ME, 2000, and XP; as well as Mac OS9x, and OSX.

-Click inside the Property inspector Fill Color box (make sure you've clicked the Pointer tool to exit text editing mode)

-Click the **Fill Options** button

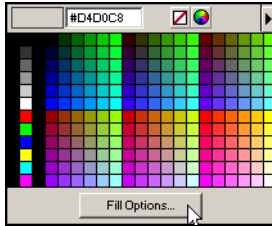


FIGURE 68

-Select **Linear**

-Click the **Edit** button inside the Fill Options dialog

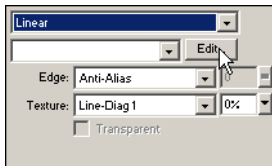


FIGURE 69

-Click the left color chip and when the swatch pop-up opens, use the eyedropper to select the canvas color (#A2AAB5)

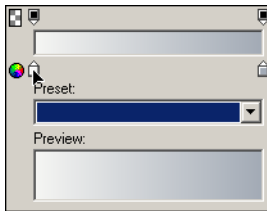


FIGURE 70

-Click the right color chip and set its color to white (#FFFFFF)

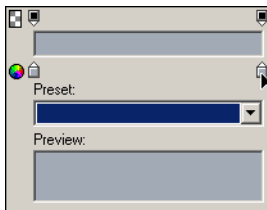


FIGURE 71

-Press **Enter** to close the Edit Gradient dialog

-Click on an empty part of your canvas to close the Fill Options dialog

The left end of the text all but disappears into the canvas. Adjusting the placement of the gradient is an easy way to fix that.

-Click on the round handle of the gradient rotation bar (the black line with round and square handles on its ends¹) and drag it to the left until you can clearly see the letter **f**.



FIGURE 72

Next, you'll add a drop shadow to the text. With the header text selected, click on the Add Effects button [+] on the Property inspector.

-Choose **Shadow & Glow**→ **Drop Shadow**



FIGURE 73

For the sake of readability, the distance between the shadow and the text should be small.

-Use the slider to adjust the distance setting to **2**

Dark shadow colors contribute to the illusion that the light falling on the object is strong. Conversely, shadow colors that are only a little darker than the background color of the image suggest diffused lighting.

The color you'll use is darker than the canvas color by around six shades.

-Click in the Fill Color box of the Drop Shadow dialog box to open a swatches pop-up

-Use the built-in color picker to select the **#666666** swatch

-Set the opacity of the shadow to **65%**

-Set the degree of scatter (Softness) to **0**

-Leave the angle set at the default **315°**

-Click on an empty part of the Property inspector or canvas to close the shadow dialog box

-Move the finished text to convenient position by typing **24** into the *X box* of the Property inspector and **199** into the *Y box*.

-Press **Enter** to accept the changes.

¹ The Gradient drag handles will remain present at all times.

THE HOME ICON

This tiny house image goes immediately to the left of the logo header text.

-Drag out a **22 pixel wide** by **21 pixel high** rectangle. Make any necessary correction to the dimensions in the Property inspector.

This will serve as the background and frame for the icon.

The Fill Category selector and Fill Color boxes in the Property inspector remain set up for the gradient you used for the logo header text. You'll change the gradient colors to the same ones you used for the button background, but with the two colors reversed.

-Click inside the Fill Color box on the Property inspector to open the *Edit Gradient* pop-up window

-Click on the left lower color chip to open a swatches pop-up

-Use the color picker to click on an empty spot on the canvas (to pick up the canvas background color)

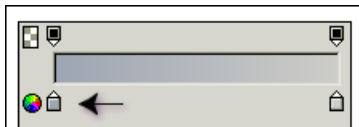


FIGURE 74

-Click on the right lower color chip to open a swatches pop-up

-Use the color picker to choose the #CCCCCC swatch

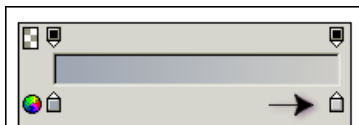


FIGURE 75

-Click on an empty spot on the canvas to close the Edit Gradient pop-up window

We'll add a stroke to the Home icon's background rectangle to give it depth. To add the stroke:

-Click in the *Stroke category* box in the Property inspector.

-Choose **Pencil**→**1-Pixel Soft**

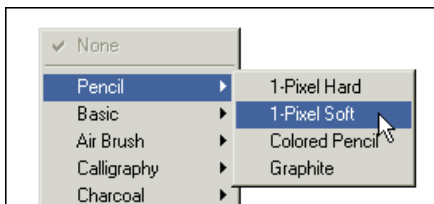


FIGURE 76

To color the stroke:

- Click inside the Stroke Color box to open a swatches pop-up
- Enter the Hex number **#FFFFFF** (white) into the text box of the swatches pop-up
- Press **Enter** to accept the changes and close the pop-up
- Use **0** for the remaining stroke settings

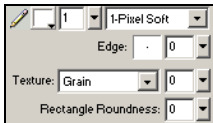


FIGURE 77

You'll be working with tiny shapes for the rest of the Home icon creation. Use the zoom pop-up menu at the bottom of the document window to increase magnification to **400%**.



FIGURE 78

To draw the house:

- While pressing the Shift key to constrain the proportions of the shape, drag out a **10 x 10** white square with the Rectangle tool. If necessary, adjust the dimensions in the Property inspector.
- Change the Fill Category to **Solid**
- Change the Stroke Category to **None**
- Click in the Fill Color box in the Properties inspector. Select the **white** swatch in the swatches pop-up.

This square will serve as the base for your house. Move it atop the icon background, towards the center. You'll draw the roof, the door and the window next.

- Press and hold on the Rectangle tool until the Shape menu flies out. Click on the Polygon tool to select it.

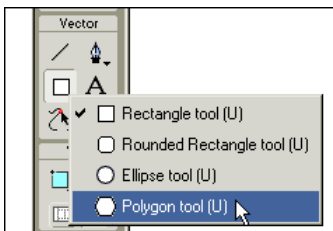


FIGURE 79

-In the Property inspector, set the polygon shape to **Star**

-Change the number of sides to **3** and the angle to **50**

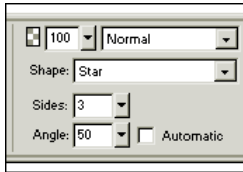


FIGURE 80

-While pressing the Shift key, drag upward to create a triangle (if you drag downward, the apex of the triangle will be on the bottom).

-Adjust the triangle dimensions to **17 pixels wide** and **9 pixels high**.

-Click in the Fill Color box to open the swatches pop-up and select the **#666666** swatch to change the triangle color to dark gray.

-Move the triangle to the top of the house to form a roof.

-Press **Shift** and select the house, the roof, and the rectangle. Use the *Align panel* to center these elements horizontally.

-Adjust the roof if necessary so that it covers the house.



FIGURE 81

-Draw two tiny rectangles to represent a door and window.

-Fill them black (**#000000**) so you can easily see them (the Fill Color doesn't matter because we will be punching these shapes out from the larger rectangle)

-Make sure no stroke is set.

Cutting the door and window shapes out of the white square that forms the body of the house gives the image more character and depth. Fireworks MX makes this operation simple by enabling you to punch one vector shape through another.



FIGURE 82

-Press Shift and select both the *door shape* and the *main body* of the house.

-Choose **Modify**→ **Combine Paths**→ **Punch**

-Press Shift and select both the *window shape* and the *main body* of the house.

-Choose **Modify**→ **Combine Paths**→ **Punch**

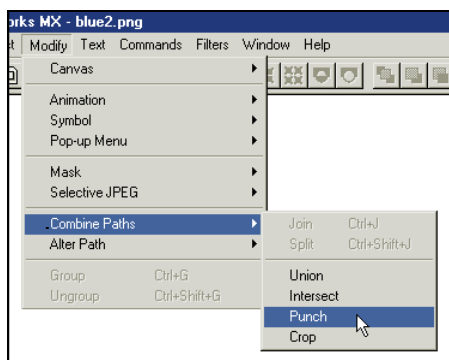


FIGURE 83

Your home icon should look something like this: . The next step is to add some drop shadows to make the shapes of the icon pop out from the background.

Select the house roof and then click on the add effects button [+] in the Property inspector.

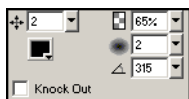


FIGURE 84

-Choose **Shadow and Glow**→ **Drop Shadow**

-Set the shadow distance to 2

-Set the color to #666666

-Set the shadow opacity to 65%

-Set the shadow softness to 2

-Set the shadow angle to 315°

-Apply a shadow to the lower part of the house **and then to the icon's background rectangle** using the same settings

Draw a cursor box around the icon to select all of its elements, then choose **Modify**→ **Group**. Move the completed Home icon to the left of the logo header text (X:1, Y: 205).



FIGURE 85

The final part of the logo is the subheading text.

- Select the Text tool and click on an empty spot on the canvas.
- Click in the Fill Color box in the Property inspector (or the Tools panel) to open a swatches pop-up
- Set the color to **white** (#FFFFFF)
- Type the words *the future is now* in lower case characters
- Click on the Pointer tool to exit text editing mode

In the Property inspector:

- Change the font from Impact to Verdana
- Set the point size to **10**

- Set the Range Kerning to **9**

Range kerning increases or decreases the space between the characters.

- Set the Anti-Aliasing type to **none**. Leave the other text settings at their defaults

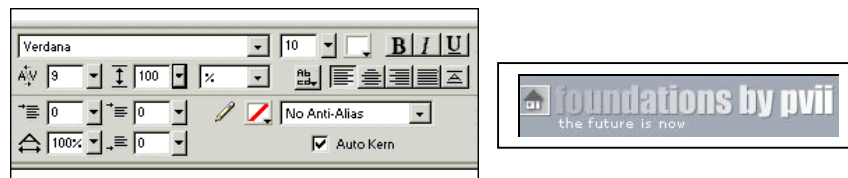


FIGURE 86

- Move the subheader text to just below the word *foundations*, or about the X: **27**, Y: **224** position

THE SHIM (TRANSPARENT SPACER GIF)

This is the shim you'll use to control the width and height of the table cells for the border decoration of the Foundations pages.

- Drag out a **12 x 12 pixel**, *white* square
- Move the square to a convenient location below the logo: X:**56**; Y:**315**. That's it. Honest!



FIGURE 87

THE MAIN TABLE CELL BACKGROUND

This image tiles a subtle checkerboard pattern across the top of the main PVII Foundations table cell.

- Drag out a **72 pixel wide x 58 pixel high**, white rectangle. Adjust the dimensions in the Property inspector if necessary
- Drag out a **60 pixel wide by 46 pixel high** rectangle
- Click in the Fill Color box in the Tools panel. Type the Hex number **#EDEFF1** into the text box of the swatches pop-up and press **Enter**
- With the rectangle still selected, click the Fill *Texture* category in the Property inspector to open the texture menu
- Choose **Hatch 1**

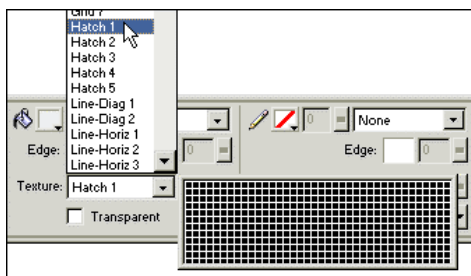


FIGURE 88

- Move the slider to **100%** intensity
- Switch to the Select tool
- Click on the background color rectangle to select it
- While pressing the *Shift* key, select the textured rectangle
- Choose **Modify→Align→Center Vertical**
- Choose **Modify→Align→Center Horizontal**
- With both rectangles still selected, choose **Modify→Group**
- Move the rectangle group below the shim image, to about the X: **2**, Y: **407** position

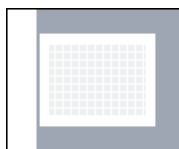


FIGURE 89

- Ungroup the rectangle group after moving it

THE FOUNDATIONS BACKGROUND TILE

The last image you need to create is a slate blue tile with a diagonal line texture.

- Press the *shift key* while dragging out a **60 x 60 pixel** square. Make any necessary adjustments to the dimensions in the *W and H boxes* in Property inspector
- Click in the Fill Color box in the Property inspector to open the swatches pop-up
- Type the Hex number **#9AA3AF** into the swatches text box and press **Enter**
- With the square selected, click the Fill Texture category in the Property inspector to open the texture menu
- Change the texture from *Hatch 1* to **Line-Diag 1**
- Change the *Amount of texture* to **8%**
- Move the background tile to a convenient spot, such as X: **287**, Y: **47**

We magnified the tile in this figure so you could see the diagonal stripe. The inset at the bottom left of the figure shows the texture at normal size.

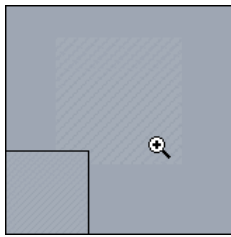


FIGURE 90

SLICING, OPTIMIZING & EXPORTING

To put it simply, slicing is something you do to prevent everything on the canvas from being exported as one, huge image. Another practical reason for slicing is that it enables you to export different objects on the canvas in different file formats (JPEG, GIF) and compress them by varying amounts.

Fireworks uses two kinds of visual indicators for slices—a green slice *overlay* and red slice *guides*.

The green slice overlay indicates the size and shape of individual slices. The red slice guides surround each slice overlay and extend across the canvas to other slices, indicating the actual cutting lines.

-To begin slicing, make slices visible by clicking on the *Show Slices and Hotspots* button, located in the Web section of the Tools panel 

Know the location of the Slicing tool (hint – it's in the Web section of the Tools panel) 

SLICING THE BUTTON BACKGROUND

-Right-click the button background image.

-Choose **Insert Slice** from the right-click contextual menu.

-In the Property inspector, name¹ the slice **navbg**

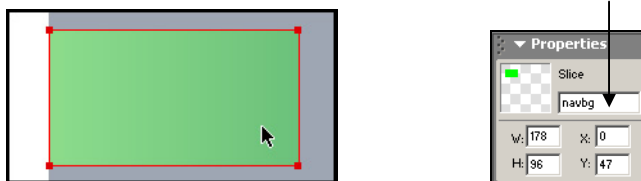


FIGURE 91- NAME THAT SLICE

Naming your slices will simplify choosing the correct image while you are working in Dreamweaver².

¹ It's a good idea to name slices as you make them. Upon making or inserting a slice, it has focus (is selected) and you can type the name directly into the Property inspector Slice box or into the Layers panel as described on the next page.

² You can also use the Layers panel to name slices.

SLICING THE LOGO

- Click on the **Slice** tool to select it. 
- Click on the canvas just above the top left corner of the logo.
- Drag diagonally, to just past the bottom right corner of the logo.
- Set the slice to a *width* of **248** pixels and a *height* of **62** pixels, and set its coordinates to **X:0** and **Y:187**
- Name the slice **logo**



FIGURE 92

SLICING THE SHIM

- Right click on the Shim image
- Choose **Insert Slice**
- Give the slice the name **shim**

SLICING THE MAIN TABLE CELL BACKGROUND IMAGE

- Right click on the gray and white checked table cell background image (select only the object with the Hatch fill and *not* the larger white rectangle it is sitting in)
- Choose **Insert Slice** from the right click contextual menu

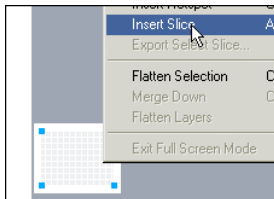


FIGURE 93

-Name the slice **tbg**

The table cell background image will look nicer with a border at its bottom edge. The simplest way to add a border is to extend the slice to include some of the white rectangle beneath it. So let's add three pixels of height to our slice.

-With the slice still selected, use the Property inspector to change its height attribute to **49 pixels**.

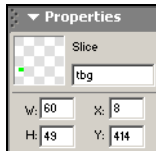


FIGURE 94

SLICING THE TABLE CORNERS AND BORDERS

Guides are not just to aid you in laying out shapes and images. They are great help when you need to draw complex slices. To set up guides around the inner and outer edges of the squares:

-Use the Zoom tool to magnify the image. 

-Make sure that *Snap to Guides* is activated (**View→Guides→Snap to Guides**)

-Click on the rulers and drag guides to the outside edges of the outer square and to the outside edges of the inner square.

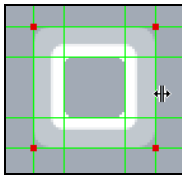


FIGURE 95- GUIDES PLACED AT THE OUTSIDE EDGES OF BOTH THE OUTERMOST AND INNERMOST SQUARES

-Click on the Slice tool to select it

-Slowly draw slices¹ over the corners and borders, allowing the magnetic guides to pull the blue slice preview lines into position.

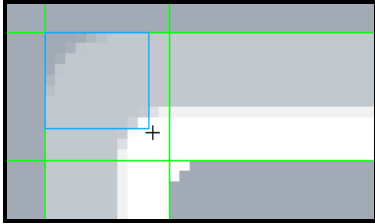


FIGURE 96- THIS SLICE SHOULD BE 12PX BY 12PX

Each of the four corner slices should end up measuring *12 pixels by 12 pixels*. The vertical sides should each be *12 pixels by 24 pixels*. The horizontal top and bottom slices should both be *24 pixels by 12 pixels*.

-Name the slices as you go, starting at the top left and moving clockwise: **tlcorner**, **tedge**, **trcorner**, **redge**, **brcorner**, **bedge**, **blcorner**, and **ledge**.

Our semi-cryptic names mean:

- tlcornet = Top Left corner
- tedge = Top edge
- trcorner = Top Right corner
- redge = Right edge
- brcorner = Bottom Right corner
- bedge = Bottom edge
- blcorner = Bottom Left corner
- ledge = Left edge

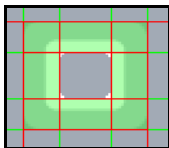


FIGURE 97- 8 SLICES

¹ Slices can be moved or resized. If you need to adjust the size or the position of a slice, we suggest zooming to a very high magnification: 400% or 800%, for example. Slices can be deleted without affecting the underlying image objects—if you'd just like to start all over from the beginning ☺.

SLICING THE STRIPED BACKGROUND TILE

- Set your View back to 100%
- Right click on the diagonally striped page background tile and choose **Insert Slice** from the context menu
- Name this slice **bgrid**

OPTIMIZING THE FOUNDATIONS IMAGES

- Open the *Optimize* panel (Press F6)

The two main things to consider when optimizing images are image quality and file size. Rarely does a single setting work for all the images. A slice that contains a photo or gradient, for example, works best exported in JPEG format¹. Export slices containing broad areas of flat color and text as GIFs.

To optimize JPEGs, increase or decrease the values in the *Quality* and *Smoothing* boxes. To optimize GIFs, increase or decrease the number of colors in the image, or change the value in the *Dither Amount* box.

In this project, we'll use the *GIF Adaptive 256* setting. There are subtle color differences from one slice to the other and Fireworks may snap the palettes of different slices to slightly different colors if you choose one of the *Web Snap* options.

Tip: While in Preview mode, you'll find the estimated file size of a selected slice displayed next to the tabs at the top of the document window.

¹ JPEGs export with a **.jpg** file extension, a holdover from the DOS era when only three-character extensions were allowed.

OPTIMIZING THE BUTTON BACKGROUND

This slice (*navbg*) represents a gradient so it should be exported as a **JPEG**.

- Switch to the **Select** tool and click on the green slice overlay on top of the image
- Click on the *Preview* tab at the top of the document window
- In the *Optimize* panel, choose *Settings*: **JPEG – Better Quality**
- Set the *Matte* color to match the canvas color (#A2AAB5)

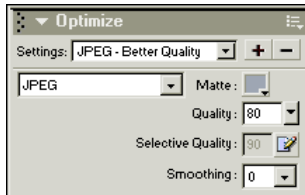


FIGURE 98

Setting the Matte color isn't necessary in this case, but, because it helps to lock the background colors into most JPEGs, it's a good habit to get into.

OPTIMIZING THE LOGO

Although this slice contains a gradient, we will use the GIF adaptive setting at 256 colors. Gradients used across a solid surface will band when converted to GIF format. When the gradient runs across a noncontiguous surface like a string of text, Fireworks assigns solid, graduated colors to the characters along the string, reducing, or even eliminating, the occurrence of banding. Exporting the logo in GIF format also enables you to assign a transparent background to the image.

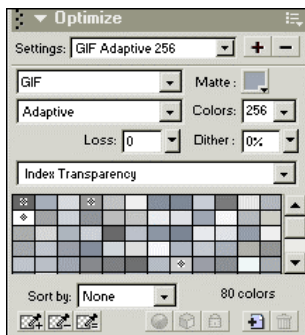


FIGURE 99

-In the Preview window, click on the logo image and choose *Settings: GIF Adaptive 256*

-Set *dithering* to **0%**

Note: The dithering option tiles the pixels in an 8-bit graphic format in a manner that softens the transition from one color to the other, creating an illusion of greater depth. From a distance, the colors will blend and create the appearance of the missing color. However, dithering can *increase* file size.

-Choose *Type of Transparency*: **Index Transparency**

We've left the number of colors set at 256, though since the image itself contains only 80 colors; you could probably lessen the number of colors to 64 with little quality loss. Leaving the color number setting greater than the number of colors contained in the image does not increase the file size because, during optimization, Fireworks cannot add colors that are not already there.

Setting index transparency will allow the image to seamlessly blend into the web page background.

OPTIMIZING THE SHIM

-Click on the **shim** image to select it

-In the *Optimize* panel, choose the **GIF Adaptive 256** setting

-Click on the *Select Transparent Color* button located on the button bar at the bottom of the Optimize panel

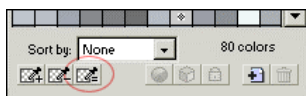


FIGURE 100

-Click with the *eyedropper* on the **shim** image to make it transparent

OPTIMIZING THE MAIN TABLE CELL BACKGROUND IMAGE

- Click on the **Table Cell Background** Image to select it
- In the *Optimize* panel, choose the **GIF Adaptive 256** setting
- Leave the *Transparency* box set to the default **No Transparency**

The remaining images do not require any transparent pixels.

OPTIMIZING THE TABLE CORNERS AND BORDERS

For each of the eight Table Corner and Border images—

- Click on the image to select it
- In the *Optimize* panel, choose the **GIF Adaptive 256** setting
- Leave the *Transparency* box set to the default **No Transparency**

OPTIMIZING THE STRIPED BACKGROUND TILE

- Click on the **Striped Background Tile** Image to select it
- In the *Optimize* panel, choose the **GIF Adaptive 256** setting
- Leave the *Transparency* box set to the default **No Transparency**

EXPORTING THE FOUNDATIONS IMAGES

When you are satisfied that your optimization settings will produce the quality and file sizes you want, it's time to export.

-Choose **File/Export**

The Export dialog box will open.

The Fireworks Export window is a bit confusing in certain circumstances. The **File name** box should really be hidden when exporting multiple slices. Perhaps Macromedia assumed that everyone would want to use Fireworks HTML ☹. When multiple slices are being exported, the **File name** is meaningless. Fireworks exports separate GIF or JPEG files based upon the names and optimization settings for your individual slices.

-Browse to your **Foundations site** and double-click the **assets** folder, which will appear in the *Save in* box at the top of the window

You'll notice it's populated with folders containing the bonus images we've provided. You don't want to overwrite them, so you'll create a new folder for them.

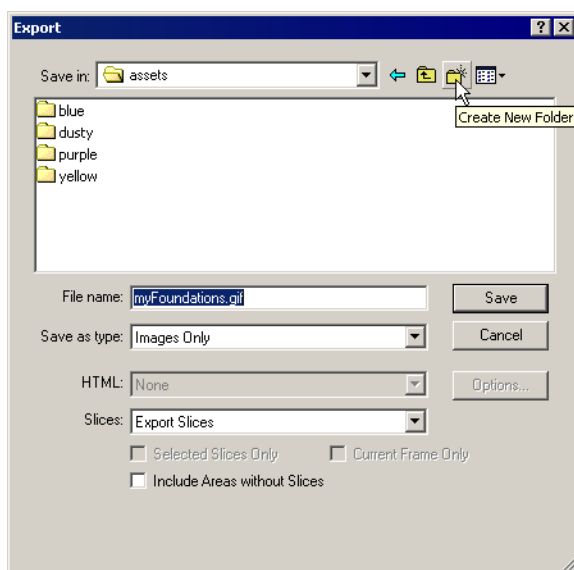


FIGURE 101

-Click the *Create New Folder* Icon.

A new folder will appear below the one called *yellow*. The folder name will be ready to edit.

-Type the new folder name **mfimages** (short for: *my foundations images*)

-Click on the *mfimages folder icon* to **highlight** it

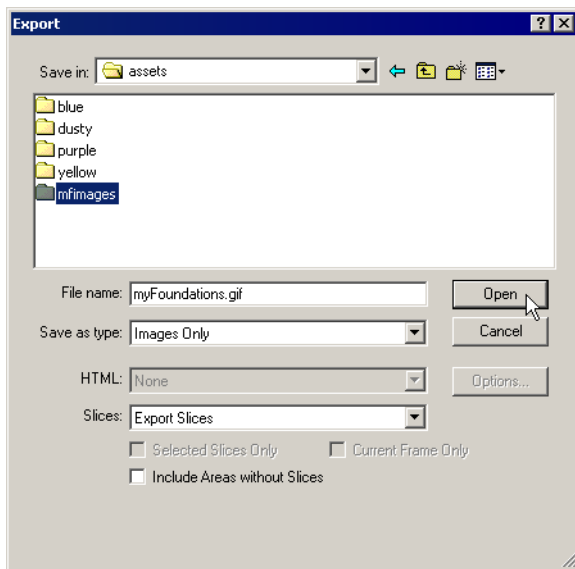


FIGURE 102

-Click the **Open** button

The *Save in* box at the top of the window will change from **assets** to **mfimages**.

You're now ready to *Save in: mfimages*. Enter the following settings in the window:

-*File name: myFoundations.gif*¹

-*Save as type: Images Only* (choosing the *Images Only* option grays out the HTML option box)

-*Slices: Export Slices* (Fireworks prefers precise instructions. If you don't tell it to Export Slices, everything in the document window exports as one, huge bitmap)

-*Include Areas without Slices: Unchecked* (The *Selected Slices Only* box is grayed out unless your document's slices were visible at the time you opened the Export window. If the *Selected Slices Only* box is available, do make sure it is unchecked.)

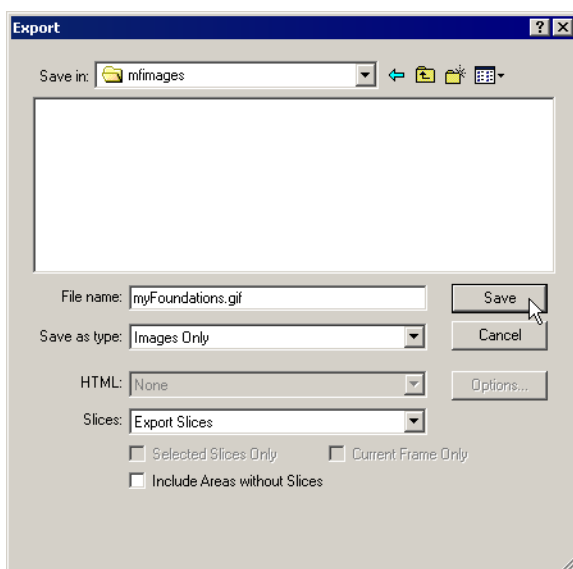


FIGURE 103

-Click the **Save** button.

The export window will close.

¹ Since we are exporting multiple slices, this box is meaningless. Fireworks will probably have entered a name based on your document's file name, which is fine. No changes are necessary.

If you'd like to see what Fireworks saved in your new folder, choose **File**→**Export** again. You'll see that Fireworks has exported each of your slices as a separate file: twelve **GIFs** and one **JPEG**. You should also see a new folder called **_notes**, which Fireworks automatically created to help it communicate with Dreamweaver.

Important: Should you forget to name a slice, Fireworks will give it a name (ranging from something as simple as "slice" to a more cryptic "r2d23cpo"). If that happens, name the slice and re-export to your mfimages folder.

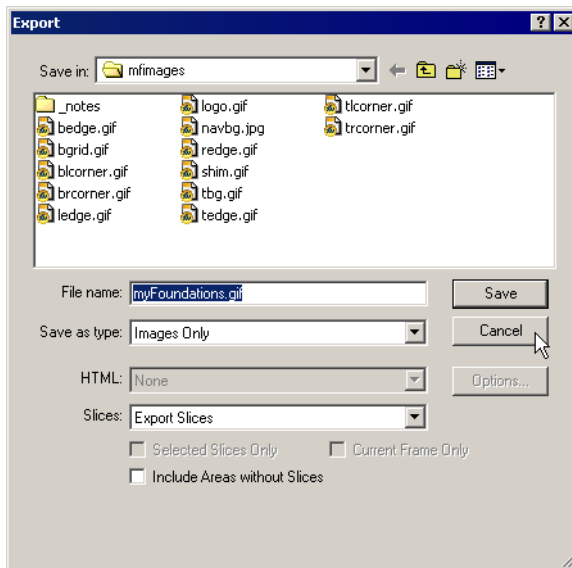


FIGURE 104

-Click **Cancel**

We're done!

“Variatio delectat” — Found scribbled on a wall in Pompeii.

Translated: “Variety is the spice of life.”

IN CLOSING

For the fun of it, we’ve included some color variations of the Foundations images. You’ll find Dusty, Blue, Yellow, and Purple PNG files in the Fireworks folder of your Foundations directory.



As always, have fun!

Let’s go build a Web page.

-Close Fireworks and open Dreamweaver.

Chapter 8—Building the Foundations Page

**“SHE WELL KNEW THE GREAT ARCHITECTURAL SECRET OF
DECORATING HER CONSTRUCTIONS, AND NEVER DESCENDED
TO CONSTRUCT A DECORATION.” —ANTHONY TROLLOPE**

Now that we have our assets covered¹, we’re ready to start building a page. The techniques we use may seem to be at odds with conventional Dreamweaver wisdoms. This is wholly intentional and quite necessary if we’re to meet our goal of producing the cleanest cross-browser code we possibly can. Dreamweaver is the well-equipped kitchen—but we are the chefs.

-Launch Dreamweaver and open **foundations/index.htm** in Design View.

INSERTING A TABLE

Foundations uses a single (multi-celled) fluid² table for structure. To keep our HTML markup as efficient as possible, we do not use Dreamweaver’s Table Layout View³.

-**Choose Insert→ Table** (Ctrl + Alt + T). The Insert Table dialog will open

-Fill out the property boxes just as we have in the Figure below

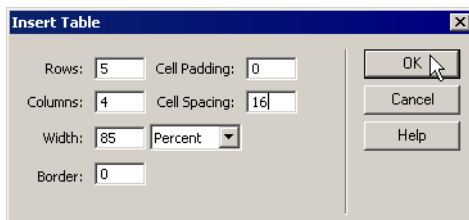


FIGURE 105

-Click **OK**

¹ No pun intended—*honest*.

² Fluid tables are tables whose width is set to a percentage of the browser window. As the window is resized, the table shrinks or expands to fill its allocated percent of the available window width.

³ If you understand how your table is built and its underlying code, it will be far easier to manage and edit that table as time goes by, and your code will always be as efficient as possible. Dreamweaver Layout Tables produce table markup that can get out of control very quickly. Dreamweaver Layout Tables are best avoided if at all possible.

The Accessibility dialog will open.

THE ACCESSIBILITY DIALOG

Since our table is for layout purposes only, we need neither a Caption¹ nor a Header. We should, however, summarize the table's purpose for the benefit of assistive readers. We do this by entering *Layout Table* in the Summary box.

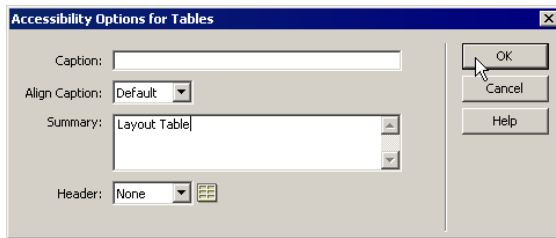


FIGURE 106

-Click **OK**

The table appears on the page aligned to the left edge of our window.

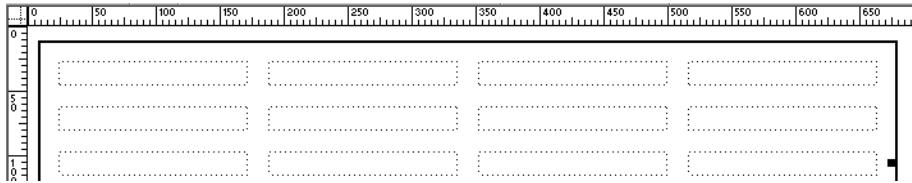


FIGURE 107

However, we want our table to be centered.

¹ Captions and Headers should not be used in tables meant for design purposes only. Doing so will confuse assistive technology used by the disabled.

CENTERING THE TABLE

-Click the <table> tag selector at the bottom of your window

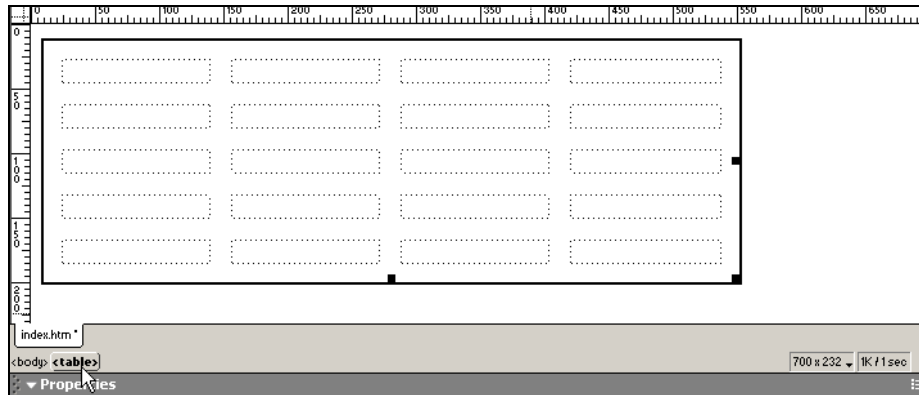


FIGURE 108

-Open the Align list on your Properties inspector and choose Center

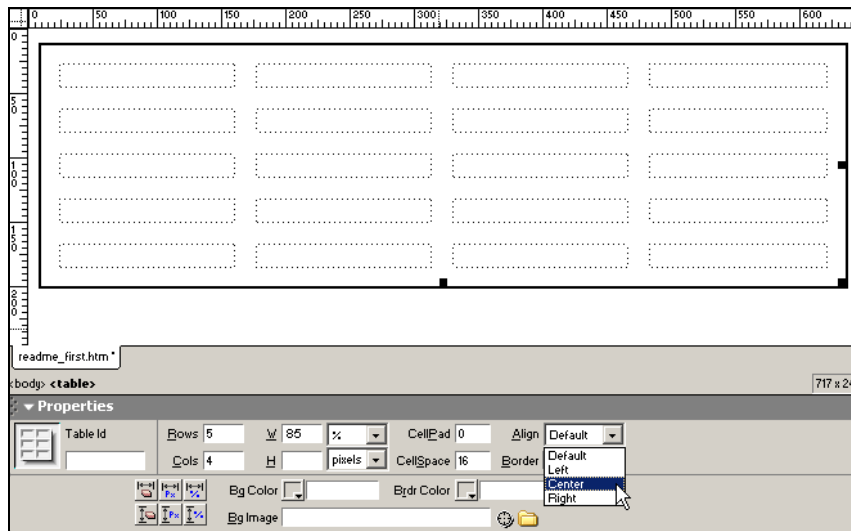


FIGURE 109

The table will snap to the center of your window.

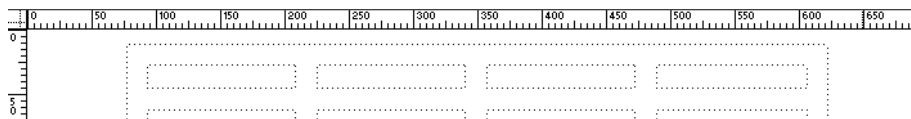


FIGURE 110

CHECKING THE SOURCE CODE

-Switch to Code View

Let's see what our table markup looks like. I've added comments to guide you on your journey:

```
<table width="85%" border="0" align="center" cellpadding="0" cellspacing="16" summary="Layout Table">
```

The opening `<table>` tag begins the table and sets its HTML properties ¹.

```
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

The first `<tr></tr>` pair denotes a row². Inside this row, there are 4 `<td></td>` pairs, denoting 4 columns within the row.

```
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

This is the second row. It also has 4 columns.

```
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

This is the third row. It also has 4 columns.

```
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

This is the fourth row. It also has 4 columns.

```
<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
```

This is the fifth row. It also has 4 columns.

```
</table>
```

This denotes the end of the table.

¹ If we were making a page for version 5 and higher browsers only, we could dispense with all of the HTML table properties (except Summary) and use CSS exclusively. We can probably remove the width and align properties—but we'll leave them in (at least for now).

² Each `<tr></tr>` pair denotes a row (tr stands for **table row**). Inside each row, there are four `<td></td>` pairs (td stands for **table data**), denoting four columns within the row. So our table has five rows—each with four columns.

The ** ** inside each `<td>` is a non-breaking space¹. Dreamweaver places one inside each empty table cell to keep the cell from disappearing in Navigator 4.

As you add content to a cell, the space is removed automatically. For our design, these spaces serve no useful purpose. But if one of our empty cells happened to carry a background color that was vital to our design, then that cell would need to contain either a transparent GIF or a non-breaking space for Navigator 4 to render it. However, this is not the case with our page.

When we finish building our table, we'll sweep through the code and remove unnecessary spaces.

It's a little tricky to do because of a quirk in Dreamweaver, but we'll get it done.

CODE FORMATTING

Why is my code flush left, while yours is indented? Two reasons:

I like the way it looks... and it saves bandwidth. Blank space in source code does take up memory. One byte per space. One thousand blank spaces in a page's source code will add 1K to its size. However, indents can make the source code more readable for some people. If you'd like to remove the indents from your source code, it couldn't be easier...

¹ If you wanted to make a non-breaking space () in Dreamweaver, you would press Shift + Ctrl + Spacebar.

-Choose **Edit**→ **Preferences**. The Preferences window will open

-Select the *Code Format* category

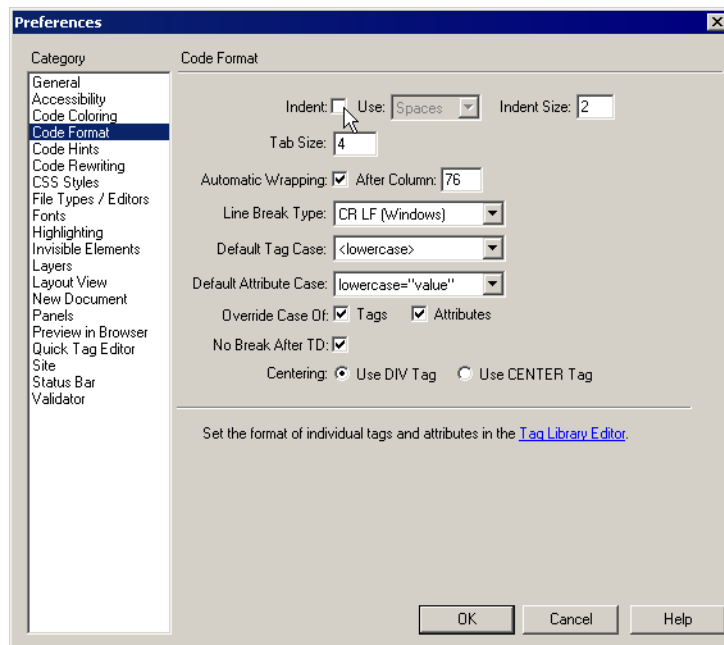


FIGURE 111

-Clear the mark from inside the *Indent* check box and click **OK**

-Now choose **Commands**→ **Apply Source Formatting**:

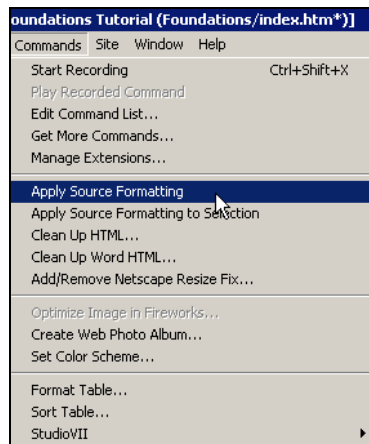


FIGURE 112

Look at your source code again.

It's flush left.

INSERTING THE LOGO

Back to business now! We need to insert our logo.

-Switch to **Design View** and place your cursor in the second column of the top row

-Choose **Insert→Image** (Ctrl + Alt + I)

The Select Image Source window will open.

-Make sure *Select Name From* is set to **File System**

-Browse to the *assets/mfimages* folder of your site and select **logo.gif**

-Make sure *Relative To:* **Document** is selected

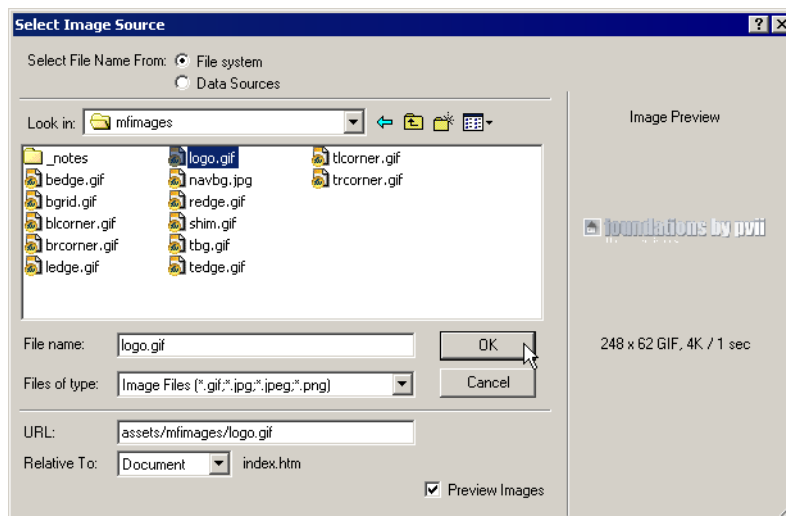


FIGURE 113

-Click **OK**

The Accessibility Dialog will open.

Alt text should always match any words contained in an image¹. Our logo image contains the words: “Foundations by PVII The Future is Now”. That says it all!

-Enter the text *Foundations by PVII - The Future is Now* into the Alternate Text box. No Long Description is needed (no need to delete the text: **http://**)

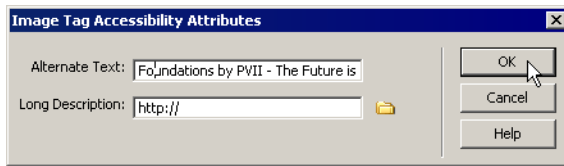


FIGURE 114

-Click **OK**

The image appears where your cursor was (in the second column, first row).

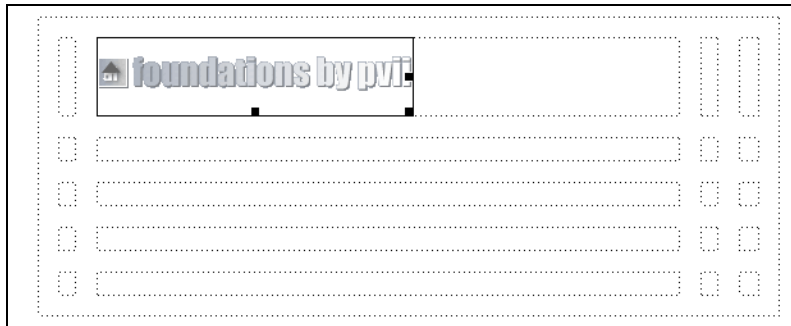


FIGURE 115

Notice that the table cells have collapsed. This is the reason why we set cell spacing to 16px in the Insert Table dialog. Without it, your table cells would have collapsed completely. Once we’ve completed our table markup, we’ll set the table’s cell spacing to zero and everything will fall into place.

Also—notice that the image looks somewhat ragged. It doesn’t blend very well with our page’s default white background. Don’t worry—we’ll fix that in a moment by creating a CSS rule. But first, we need to do a bit more planning.

Let’s mark up our table so that we know what each cell is meant to do. Let’s make a blueprint of sorts...

¹ If there are no words in the image, then the Alt text should describe (in as few words as possible) the meaning conveyed. Alt text is not *tooltips*. It just happens that some browsers work that way—but it’s not a specification and therefore cannot be depended upon to work in all browsers.

BLUEPRINTING OUR TABLE AND ASSIGNING IDs

As we discovered in Chapter 6, planning is very important. Before we begin styling our table, we’re going to mark it up so that we know where everything needs to go.

-Insert your cursor in **row-1/column-1**

-Type the word: **blank**

-Mark the remaining cells precisely as you see in the Figure below, using **Table 1** as a guide

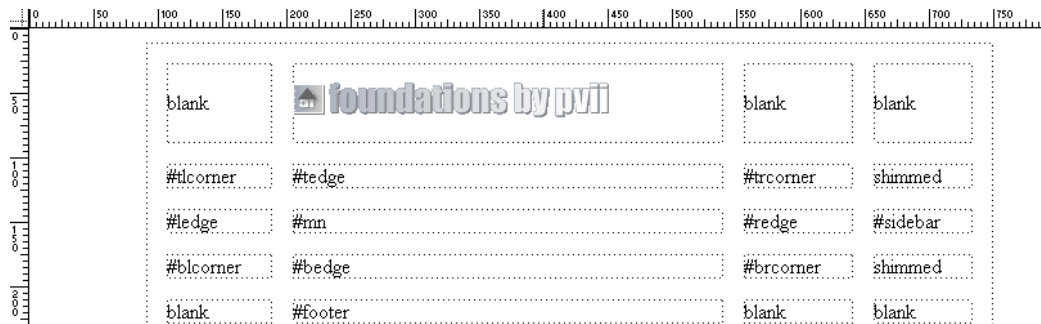


FIGURE 116- OUR “BLUEPRINT” (YOU CAN USE YOUR TAB KEY TO NAVIGATE THE CELLS)

	Column 1	Column 2	Column 3	Column 4
Row 1	blank	-----	blank	blank
Row 2	#tlcorner	#tedge	#trcorner	shimmed
Row 3	#ledge	#mn	#redge	#sidebar
Row 4	#blcorner	#bedge	#brcorner	shimmed
Row 5	blank	#footer	blank	blank

TABLE 1- GUIDE FOR ENTERING MARKER TEXT IN TABLE CELLS

We’re going to give IDs to all the table cells in which we’ve placed a leading octothorpe¹ (#). We can then make special CSS selectors (called ID Selectors) that apply to just one element according to that element’s ID². An ID selector is a powerful CSS tool that is perfectly suited to styling elements that are one-of-a-kind. Now let’s discuss the role each ID Selector will perform...

¹ In the early 1960s Bell Labs developed a new phone system. This system had certain features which were activated by using the # sign. A Bell Labs supervisor giving a training seminar felt the need to come up with a word to describe the # symbol. His thought process was as follows: There are eight points on the symbol so “OCTO” should be part of the name. There should be a few more letters or another syllable to make a noun, so what should that be? The supervisor happened to be active in a group that was trying to get JIM THORPE’s Olympic medals returned from Sweden. Adding THORPE would be unique, and he thought no one would ever suspect the word “OCTOTHORPE” was made up.

² While ID Selectors are the choice for one-of-a-kind elements, there are other types of CSS selectors that can be used for multiple elements that can exist anywhere in your site: *Custom Class Selectors*. Custom Class Selectors are always preceded with a period (.) as opposed to an octothorpe. CSS also allows the redefining of an HTML tag. These are called *Element Selectors* (or sometimes “tag” selectors). Element selectors have no preceding symbol. An example of an element selector is: `body {margin: 24px}`.

REVIEWING THE CSS ID SELECTORS

Eight of our selectors will serve to assign background images to the table cells that comprise the main content area's borders:

1. #tlcorner [top left corner]
2. #tedge [top edge]
3. #trcorner [top right corner]
4. #redge [right edge]
5. #brcorner [bottom right corner]
6. #bedge [bottom edge]
7. #blcorner [bottom left corner]
8. #ledge [left edge]

The remaining three ID Selectors will style the content areas of our page:

1. #mn [main page content]
2. #sidebar [right sidebar]
3. #footer [the footer beneath the main content]

What comes next can be approached two ways. We can assign the IDs to their associated `<td>` tags and create the CSS rules; or we can create the CSS rules and then assign the IDs. If we were coding our page manually¹, it really wouldn't matter which way we did it. Since Dreamweaver gives us an easy, visual way to assign IDs (providing they already exist), let's make our CSS rules first. Besides, it will eliminate potential spelling errors.

CREATING A STYLE SHEET

CSS (cascading style sheets) is the language of web *design*. You're going to become CSS authors—programmers in every sense of the word. CSS provides enormous efficiencies in terms of how we manage not just single pages, but entire sites. It allows us to separate the structure of our page's information from its style.

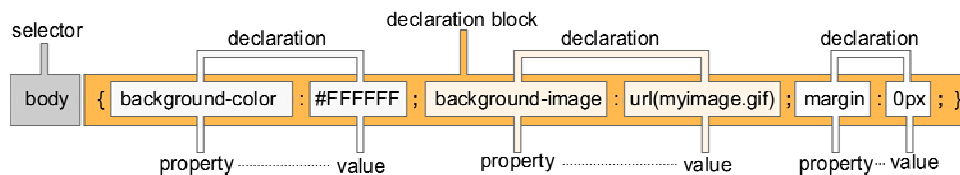
CSS is written as a series of *Rules* (which are extremely flexible). A style sheet can have just one rule that affects a single element on your web page, or multiple rules that affect many parts of the page, but they are always built the same way—the property part of the declaration must always be followed by a colon (:), and the value (or sometimes a set of them) is followed by a semicolon (;). Although CSS specifications don't insist on a semicolon after the last value in a declaration, it's good practice to include one. You'll come to appreciate the importance of this habit when manually editing style rules, since forgetting to add one at the end of the previous property-value pair could render your entire rule invalid. Dreamweaver MX now automatically terminates every value with a semicolon (although previous versions of Dreamweaver do not).

¹ We will make regular excursions into the code. Dreamweaver stands alone among its competitors in its rich code editing features. Learn to use them. You won't be sorry.

A CSS RULE DEFINED

```
body {background-color: #FFFFFF;background-image: url(myimage.gif);margin: 0px;}
```

In this rule, *body* is the *Selector* (the element we are styling). Everything inside the opening and closing squiggly brackets¹ is the *Declaration Block*. In this example, there are three declarations inside the block, each consisting of a property and a value²:



Rules can be written on a single line or on multiple lines. Dreamweaver writes multiple-lined rules, which tend to be easier to read—

```
body {
    background-color: #FFFFFF;
    background-image: url(myimage.gif);
    margin: 0px;
}
```

CSS EFFICIENCIES

Here is what the `<body>` tag would look like if we'd used the Dreamweaver Page Properties window to configure it:

```
<body bgcolor="#FFFFFF" background="myimage.gif" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
```

With our CSS rule employed, we reduce the `<body>` tag to its bare minimum:

```
<body>
```

We've saved ourselves a few characters. However, the real benefit comes when our CSS file is linked to many pages. The `<body>` rule is contained in just a single CSS file that is linked to all of our pages. Browsers need only download it once. Imagine the savings in a 50, a 100, or a 1000-page web site. More important—if you ever wanted to change the background or adjust margins, the changes are made in a single file: the CSS file. You only need to upload the edited CSS file to your server and all the pages it is linked to will automatically and instantaneously reflect the changes.

Now rather than bog you down with esoteric theory, we're going to dig in and learn by doing. In the course of doing, you will learn how CSS works... and why it's good. We'll bog you down with esoteric and possibly even existential stuff later on. Don't jump ahead though. We have our eyes on you.

¹ The closing squiggly bracket is placed on a line by itself at the end of the rule. This is for readability and is not required, although Dreamweaver does write rules in this manner by default.

² A property can actually have more than one value declared. An example would be this rule, which uses shorthand values for the margin property: `.foobar {margin: 6px 12px 18px 24px;}`. CSS shorthand is explained later in this chapter.

SETTING CSS PREFERENCES

Before we begin writing CSS with Dreamweaver, we're going to make a small change to the default preferences to make Dreamweaver write our CSS rules more efficiently.

-Choose **Edit**→ **Preferences**

-Select the *CSS Styles* Category

-In the *When Creating CSS Styles/Use Shorthand For* section, check the **Margin and Padding** and the **Border and Border Width** boxes

-In the *When Editing CSS Styles* section, select **According to Settings Above**¹

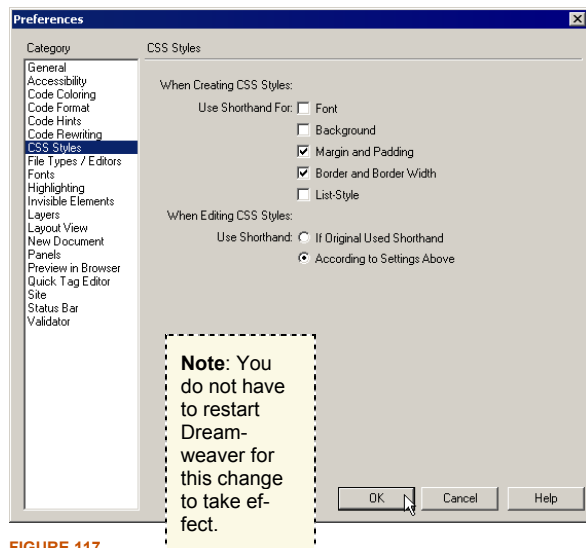


FIGURE 117

-Click **OK**

Dreamweaver will now use CSS shorthand for margin, padding, and border properties—which is supported by version 4 and higher browsers. We don't want to use font, background, or list shorthand because the syntax is a bit more complicated and browser support inconsistent. Now let's make a style sheet.

CSS SHORTHAND

CSS shorthand permits combining values. Properties that have possible top, bottom, left, and right values can be shortened to a single line and the direction can be omitted. This saves many keystrokes (and bandwidth). Instead of having four separate properties: padding-top; padding-bottom; padding-left; and padding-right—we can shorten up to a single property. To keep track of which value relates to which side, CSS has a default order: Top; Right; Bottom; Left. Values are always declared clockwise from the top. If you commit this order to memory, you will always avoid **TRouBLE**. When a single value is declared (as in padding: 24px;) the value relates to all four sides. When two values are declared (as in padding: 24px 12px;), the first value relates to both the Top and the Bottom, while the second value relates to both the Right and the Left. If three values are declared (as in padding: 24px 24px 12px;), it means that the first value relates only to the Top; the second value relates to both the Right and the Left; and the third value relates to the Bottom. If the Right and Left values are different or if all four values are different, then all four must be explicitly declared.

¹ This is a useful setting as it will cause Dreamweaver to re-write styles you copy and paste from other programs or editors whenever you use Dreamweaver to edit the rules. Dreamweaver will not “convert” a style until you actually edit it.

CREATING A STYLE SHEET AND LINKING IT

Dreamweaver makes it easy to create a style sheet and link it to your page simultaneously. Follow along please...

THE #TLCORNER ID SELECTOR

-Open the CSS panel (Shift + F11). If you've set up your panels as suggested in Chapter 2, you'll find a CSS tab on the Design Panel Group

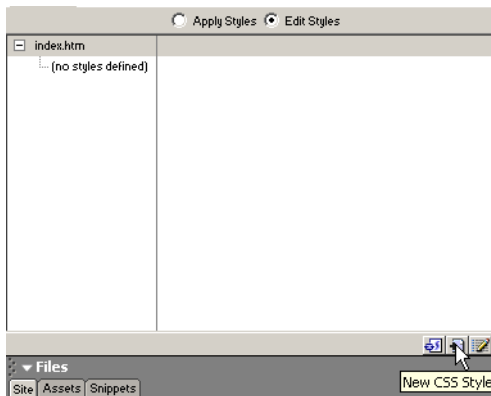


FIGURE 121

-Make sure the Edit Styles radio button is selected

-Click the New CSS Style icon 

The *New CSS Style* window will open.

-For *Type*, select the *Use CSS Selector* radio button

-Type the word **#tlcorner** in the *Selector* box

-For *Define In*, select (New Style Sheet File)

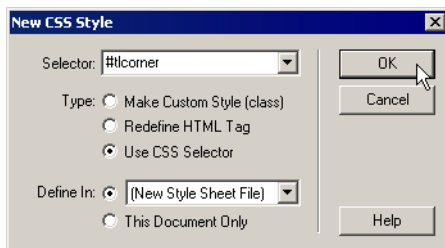


FIGURE 122

-Click **OK**

Note: If you have set Dreamweaver to use an external Style Sheet Editor like TopStyle or StyleMaster, you should temporarily disable the feature and follow along. Of course, feel free to use your own editor if you so desire.

The *Save Style Sheet As* window will open.

-For *Select File Name From*, choose the **File System** radio button

-In the *Save In* box, make sure it reads: **Foundations**. If not, click the *browse* icon to use your computer's file system to find the proper folder

-In the *File name* box, type in: **mfv5.css**

-*Save as type* should be set to **Style Sheet Files (*.css)**

-The *URL* is set by Dreamweaver, but do make sure *Relative To* is set to: **Document**

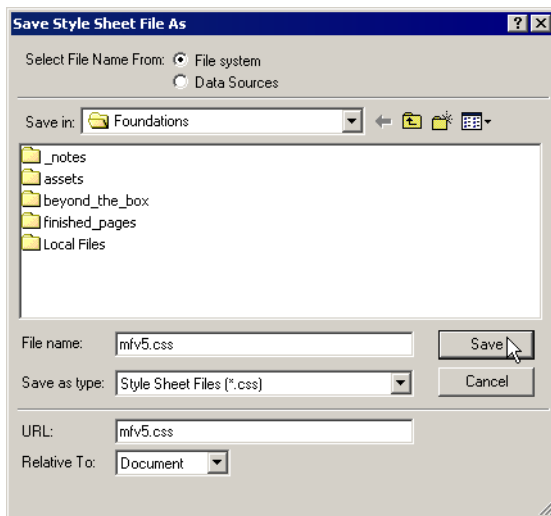


FIGURE 123

-Click **Save**

The CSS Style Definition window will open. Its title bar will reflect the style you are editing. In this case: **#tlcorner in mfv5.css**

The window will open to the Type Category.

-Switch to the Background Category

-In the Background Color box, type **#FFFFFF** (or click the color square and choose white from the color palette)

-Click the **Browse** button to the right of the Background Image box. The Select Image Source window will open

-Make sure **File System** is selected in the *Select File Name from* section

-Browse to your *assets/mfimages* folder and choose **tlcorner.gif**

-Click **OK** to close the Select Image Source window

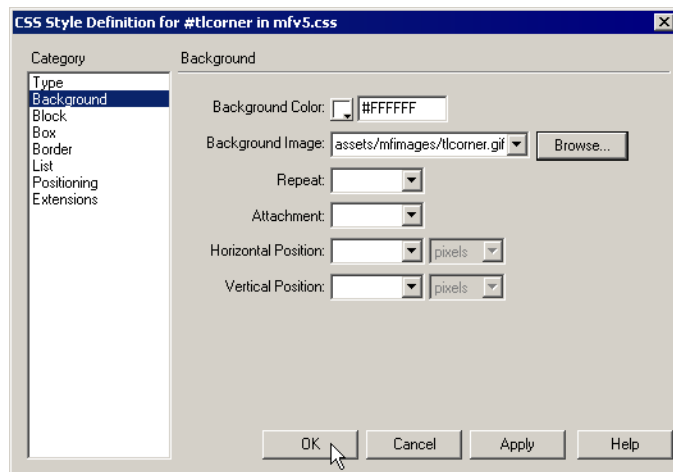


FIGURE 124

-Click **OK** to close the CSS Style Definition window

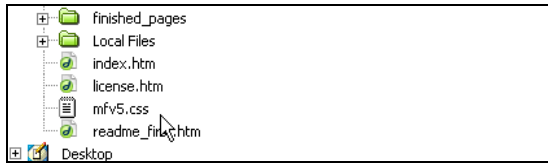
In one series of steps—we created a CSS file, we linked it to our page, and we created our first CSS style rule (an ID Selector).

-Switch to Code View and you'll see that Dreamweaver linked the new mfv5.css file to our page:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html lang="en">
<head>
<title>My Foundations Tutorial</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link href="mf5.css" rel="stylesheet" type="text/css">
</head>
<body>
```

-Switch back to Design View

A peek at the Site panel will reveal a new file in the Foundations root: **mfv5.css**



If for some reason you don't see mfv5.css in the Site panel, press F5 to refresh.

FIGURE 125

Let's see what's inside mfv5.css...

-Double-click the **mfv5.css** file icon in the Site panel

The CSS file will open in code View (which is the only view available for CSS files).

One style will be listed—the one we just made.

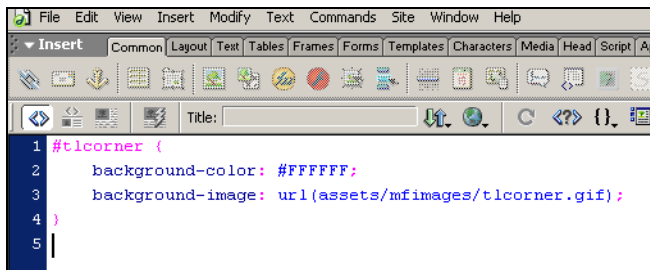


FIGURE 126

Mission control: We have a real CSS file here... and we can even edit it manually without leaving Dreamweaver! ¹

If we look at the rule, we can see a relationship between the selector *#tlcorner* and the background image's file name *tlcorner.gif*. Except for an octothorpe (#), they are the same. Planned well, we did.

We can use Dreamweaver's dialog-driven CSS Editor to make the remaining seven styles or we can stay inside mfv5.css and make the styles by hand. Let's do it by hand.

But first let's sync up and make sure our code editing options are the same.

Make sure you are in mfv5.css...

¹ That Dreamweaver MX allows us to open CSS files this way is an extraordinary new feature. As you learn more about CSS, you'll begin to feel more comfortable doing your "programming" manually... or at least tweaking your CSS manually. This gives you a built-in text editor so you never have to leave the Dreamweaver environment.

-Choose **View**→ **Code View Options**

Make sure that Word Wrap, Line Numbers, Highlight Invalid HTML, Syntax Coloring, and Auto Indent each have a leading check mark. If any are unchecked, click its line to activate it.

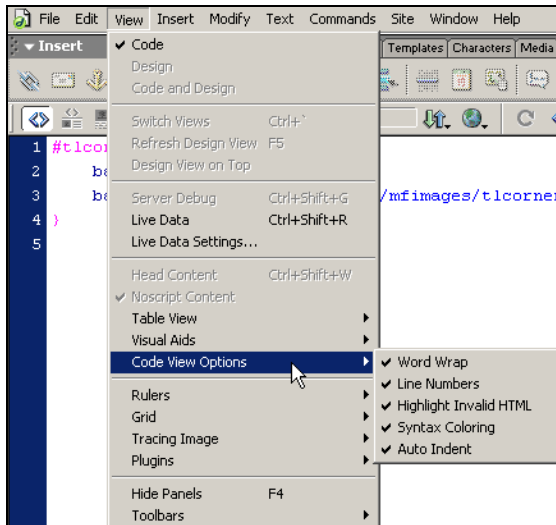


FIGURE 127

COPY AND PASTE: THE CSS CODE WARRIOR'S ALLY

-Drag-Select the entire `#tlcorner` rule from the leading octothorpe (#) to the ending squiggly bracket. The text will reverse over a dark background to indicate your selection

```
#tlcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tlcorner.gif);
}
```

-Press **Ctrl + C** to copy the code

-Place your cursor on the empty line just below the ending squiggly bracket

```
#tlcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tlcorner.gif);
}
```

-Press **Ctrl + V** to paste a copy of the rule

-If there is not a blank (empty) line below the rule you just pasted, make one by placing your cursor just after the closing squiggly bracket and pressing **Enter** once

-Repeat this process to paste **six more copies**, so that you end up with eight (8) identical `#tlcorner` rules in the file

The CSS code should now look like this:

```

1 #tlcorner {
2     background-color: #FFFFFF;
3     background-image: url(assets/mfimages/tlcorner.gif);
4 }
5 #tlcorner {
6     background-color: #FFFFFF;
7     background-image: url(assets/mfimages/tlcorner.gif);
8 }
9 #tlcorner {
10    background-color: #FFFFFF;
11    background-image: url(assets/mfimages/tlcorner.gif);
12 }
13 #tlcorner {
14    background-color: #FFFFFF;
15    background-image: url(assets/mfimages/tlcorner.gif);
16 }
17 #tlcorner {
18    background-color: #FFFFFF;
19    background-image: url(assets/mfimages/tlcorner.gif);
20 }
21 #tlcorner {
22    background-color: #FFFFFF;
23    background-image: url(assets/mfimages/tlcorner.gif);
24 }
25 #tlcorner {
26    background-color: #FFFFFF;
27    background-image: url(assets/mfimages/tlcorner.gif);
28 }
29 #tlcorner {
30    background-color: #FFFFFF;
31    background-image: url(assets/mfimages/tlcorner.gif);
32 }

```

FIGURE 128-#TLCORNERS, ATTACK OF THE CLONES

Starting with the second instance of **#tlcorner**, edit the selector and image file names to match the remaining seven IDs as marked in our table blueprint, going clockwise around the border from the top left corner.

-Edit the *second* rule to read:

```

#tedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tedge.gif);
}

```

-Edit the *third* rule to read:

```

#trcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/trcorner.gif);
}

```

-Edit the *fourth* rule to read:

```

#redge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/redge.gif);
}

```

-Edit the *fifth* rule to read:

```
#brcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/brcorner.gif);
}
```

-Edit the *sixth* rule to read:

```
#bedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/bedge.gif);
}
```

-Edit the *seventh* rule to read:

```
#blcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/blcorner.gif);
}
```

-Edit the *eighth* rule to read:

```
#ledge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/ledge.gif);
}
```

Your CSS code should now look like this:

```
1 #tlcorner {
2     background-color: #FFFFFF;
3     background-image: url(assets/mfimages/tlcorner.gif);
4 }
5 #tedge {
6     background-color: #FFFFFF;
7     background-image: url(assets/mfimages/tedge.gif).gif);
8 }
9 #trcorner {
10    background-color: #FFFFFF;
11    background-image: url(assets/mfimages/trcorner.gif).gif);
12 }
13 #redge {
14    background-color: #FFFFFF;
15    background-image: url(assets/mfimages/redge.gif).gif);
16 }
17 #brcorner {
18    background-color: #FFFFFF;
19    background-image: url(assets/mfimages/brcorner.gif).gif);
20 }
21 #bedge {
22    background-color: #FFFFFF;
23    background-image: url(assets/mfimages/bedge.gif).gif);
24 }
25 #blcorner {
26    background-color: #FFFFFF;
27    background-image: url(assets/mfimages/blcorner.gif).gif);
28 }
29 #ledge {
30    background-color: #FFFFFF;
31    background-image: url(assets/mfimages/ledge.gif).gif);
32 }
33 |
```

FIGURE 129- THE EDITED CSS FILE

-Save and close mf5.css before proceeding

BUILDING UP THE TABLE

Let's do a bit of work on the table cells that do not require IDs.

-Switch back to **index.htm**

-Delete the text *blank* in the first, third, and fourth cells of the first and fifth rows

blank		blank	blank
#tlcorner	#tedge	#trcorner	shimmed
#ledge	#mn	#redge	#sidebar
#blcorner	#bedge	#brcorner	shimmed
blank	#footer	blank	blank

FIGURE 130

The table should now look like this:

			
#tlcorner	#tedge	#trcorner	shimmed
#ledge	#mn	#redge	#sidebar
#blcorner	#bedge	#brcorner	shimmed
	#footer		

FIGURE 131

INSERT SHIM IMAGES AND IDs WHERE NEEDED

The cells labeled #tlcorner, #tedge, #trcorner, #redge, #brcorner, #bedge, #blcorner, and #ledge will employ background images driven by the CSS Selectors applied to them. We need to place **shim** images inside the eight table cells that comprise our borders, *plus* the two cells marked **shimmed**, to force those cells to collapse¹ to the size of our background images. The shim image² we created earlier in Fireworks is just what we need. Let's use it...

¹ Table cells that do not contain images are rendered with a default height and width determined by the browser. When you place an image in a cell, the cell collapses inward around the image, even if the image is 1 pixel by 1 pixel—but only if all the cells in a row have images that are the same height. Forget just one cell and your design will come apart at the seams... quite literally. This is the kind of thing that if you think about it more than a moment, your brain will collapse too. Oh, the same thing goes for the width of vertical <td> columns. Don't forget the shims.

² Because it's transparent, our shim can be resized to any dimension. We planned ahead, however, and made our shim the precise size we need for this design: 12x12 pixels.

- Insert your cursor inside the first cell, second row.
- Right-Click the `<td>` tag marker on the Tag Selector bar at the bottom left edge of your editing window
- Choose **Set ID**→ **tlcorner**¹
- Backspace over the placeholder text `#tlcorner` to remove it

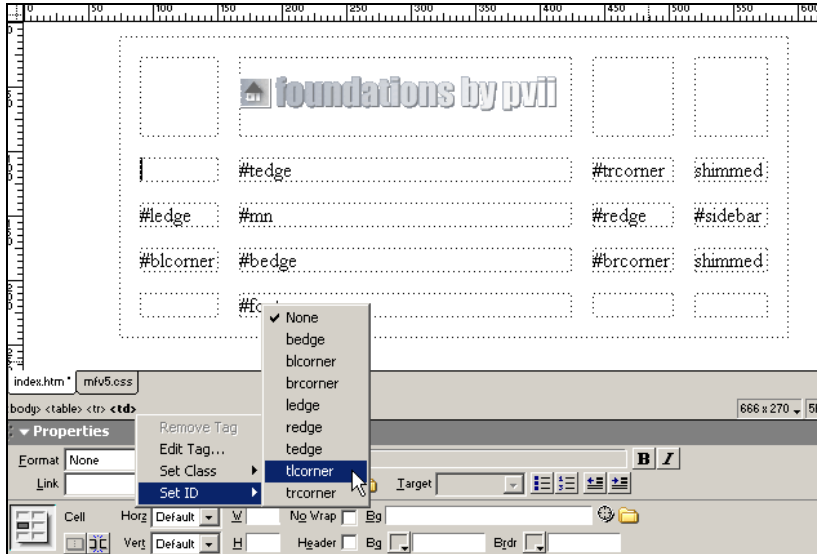


FIGURE 132

The background image will appear, but because we still have 16 pixels of cell spacing in our table, it will repeat to fill up the entire cell. That will be fixed in a few moments.

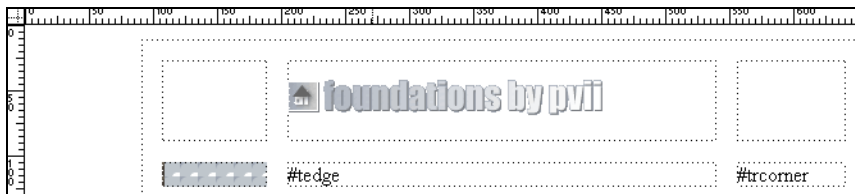


FIGURE 133- ONCE THE ID IS APPLIED, THE BACKGROUND IMAGE APPEARS, AND APPEARS, AND APPEARS...

- With your cursor still in the `#tlcorner` cell, Choose **Insert**→ **Image**
- The *Select Image Source* window will open...

¹ Dreamweaver has taken all the ID Selectors from the linked `mfv5.css` file and used them to populate the ID list. Note that when Dreamweaver presents the list of IDs, it omits the leading octothorpe, which is only used in the actual CSS rule. What looks like `#tlcorner` in the CSS file, gets entered on the tag as `id="tlcorner"`.

-Browse to your assets/mfimages folder and select **shim.gif**

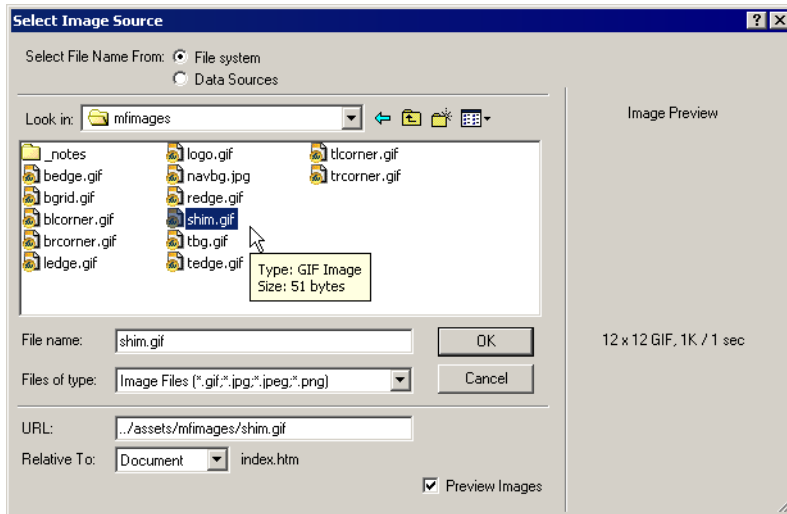


FIGURE 134

-Click **OK**

The *Image Tag Accessibility Attributes* dialog will open.

-Since our shim image articulates no information, we do not need Alternate Text. Since Alternate Text is a required attribute, the correct thing to do in this situation is to enter an empty string (the alt property followed by an equal sign and two empty quotes: `alt=""`)

There is a bug in the Accessibility dialog¹. So just leave the Alternate Text box empty. We'll use the Properties inspector to enter an empty string.

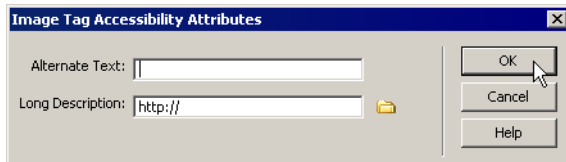


FIGURE 135

The text **http://** in the Long Description box is a placeholder to remind you that a URL is required. Dreamweaver will not insert it into your code unless it detects the box has been edited. You can leave this placeholder text in the box without worry of it winding up on your image tag.

-Click **OK**

-As soon as you click OK to close the Accessibility dialog, the shim image will be inserted and highlighted. Leave the shim highlighted

¹ If you attempt to enter two empty quotes in the Accessibility dialog, the rendered `<img>` tag will be corrupted with two extra quotes at the end.

-Drop down the *Alt* list on your Properties inspector and select **<empty>**

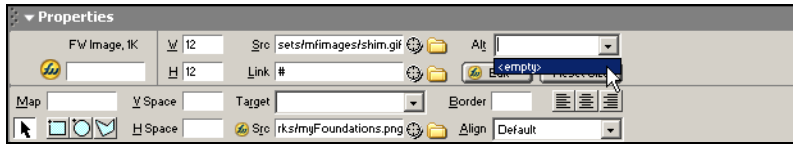


FIGURE 136

-View your code¹ to make sure the empty *Alt* string was correctly entered:

```
<td id="tlcorner"></td>
```

Notice that Dreamweaver has inserted **id="tlcorner"** inside the opening **<td>** tag.

Now you know how to set IDs and insert shims. Let's do the remaining cells:

R2-C2

- Place your cursor inside the *second cell, second row*
- Set ID to **tedge**
- Insert **shim.gif**
- Use the Property inspector to set an **empty Alt Text** string

R2-C3

- Place your cursor inside the *third cell, second row*
- Set ID to **trcorner**
- Insert **shim.gif**
- Use the Property inspector to set an **empty Alt Text** string

R2-C4

- Place your cursor inside the *fourth cell, second row*
- Insert **shim.gif** (no ID is required)
- Use the Property inspector to set an **empty Alt Text** string

¹ This would be a good time to use Split Code/Design View.

R3-C1

- Place your cursor inside the *first cell, third row*
- Set ID to **ledge**
- Insert **shim.gif**
- Use the Property inspector to set an **empty** *Alt Text* string

R3-C3

- Place your cursor inside the *third cell, third row*
- Set ID to **redge**
- Insert **shim.gif**
- Use the Property inspector to set an **empty** *Alt Text* string

R4-C1

- Place your cursor inside the *first cell, fourth row*
- Set ID to **blcorner**
- Insert **shim.gif**
- Use the Property inspector to set an **empty** *Alt Text* string

R4-C2

- Place your cursor inside the *second cell, fourth row*
- Set ID to **bedge**
- Insert **shim.gif**
- Use the Property inspector to set an **empty** *Alt Text* string

R4-C3

- Place your cursor inside the *third cell, fourth row*
- Set ID to **brcorner**
- Insert **shim.gif**
- Use the Property inspector to set an **empty** *Alt Text* string

R4-C4

- Place your cursor inside the *fourth cell, fourth row*
- Insert **shim.gif** (ID is not required for this cell)
- Use the Property inspector to set an **empty Alt Text** string

After completing the above tasks, your table should look like this:

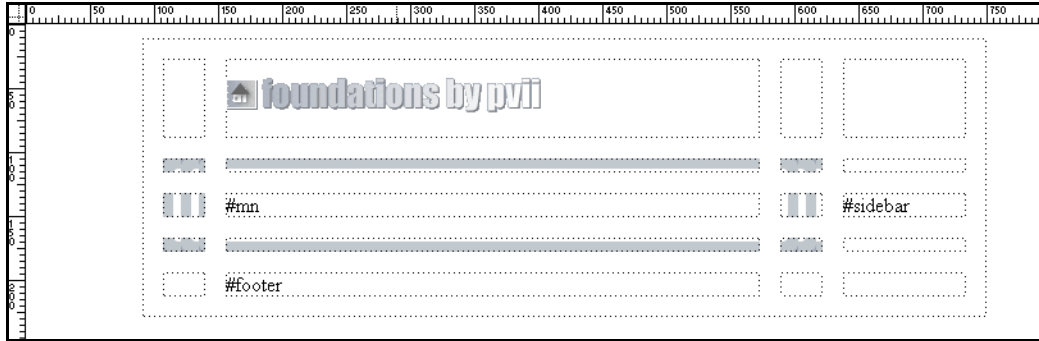


FIGURE 137

If your table doesn't look like the above Figure, make sure you've saved `mfv5.css` and that you've not made spelling errors when editing the rules. Leave the `#mn`, `#sidebar`, and `#footer` markers in place. We'll be working on those soon.

This is a good time for a code check...

CODE CHECK

At this point, our table code should look like this:

```
<table width="85%" border="0" align="center" cellpadding="0" cellspacing="16"
summary="Layout Table">
<tr>
<td>&nbsp;</td>
<td></td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
<tr>
<td id="tlcorner"></td>
<td id="tedge"></td>
<td id="trcorner"></td>
<td></td>
</tr>
<tr>
<td id="ledge"></td>
<td>#mn</td>
<td id="redge"></td>
<td>#sidebar</td>
</tr>
<tr>
<td id="blcorner"></td>
<td id="bedge"></td>
<td id="brcorner"></td>
<td></td>
</tr>
<tr>
<td>&nbsp;</td>
<td>#footer</td>
<td>&nbsp;</td>
<td>&nbsp;</td>
</tr>
</table>
```

We have six non-breaking spaces left. Since they serve no purpose (but to take up space), let's get rid of them. Instead of manually deleting each one, let's check out Dreamweaver's Find and Replace feature.

FIND AND REPLACE

With index.htm open in Code View...

-Select and highlight the first instance of * *; you find

-Press **Ctrl + F**

The Find and Replace window should open with * *; already inside the Search box.

-Set *Find In* to: **Current Document**.

-Leave *Search For* set to: **Source Code**

-Leave the *Replace With* box blank

-In the *Options* section, make sure *Ignore Whitespace Differences* is **checked** and that *Match Case* and *Use Regular Expressions* are **unchecked**

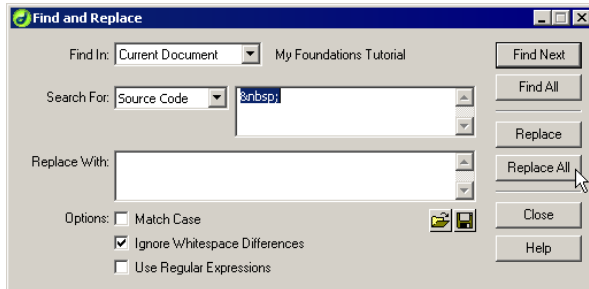


FIGURE 138

-Click **Replace All**. A dialog box will pop up informing you of the results (It should have found six instances of)

-Click **OK**. Here is how your table code should now look:

```
<table width="85%" border="0" align="center" cellpadding="0" cellspacing="16" summary="Layout Table">
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td id="tlcorner"></td>
<td id="tedge"></td>
<td id="trcorner"></td>
<td></td>
</tr>
<tr>
<td id="ledge"></td>
<td>#mn</td>
<td id="redge"></td>
<td>#sidebar</td>
</tr>
<tr>
<td id="blcorner"></td>
<td id="bedge"></td>
<td id="brcorner"></td>
<td></td>
</tr>
<tr>
<td></td>
<td>#footer</td>
<td></td>
<td></td>
</tr>
</table>
```

IF NON-BREAKING SPACES ARE BREAKING YOUR HEART

The original version of Dreamweaver MX can get downright playful when it discovers that we're trying to remove non-breaking spaces from table cells¹. We take them out and Dreamweaver keeps putting them back in. If this happens to you, you can try the following solution:

Note: The Figures below are examples only. They have no relation to the actual code on your Foundations pages.

-Switch to **Split Code View** (*Show Code and Design*)

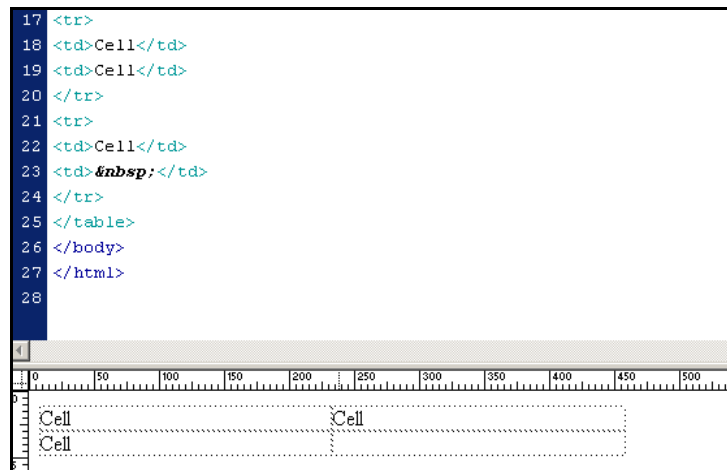


FIGURE 139

-Backspace over the ** ** to remove it

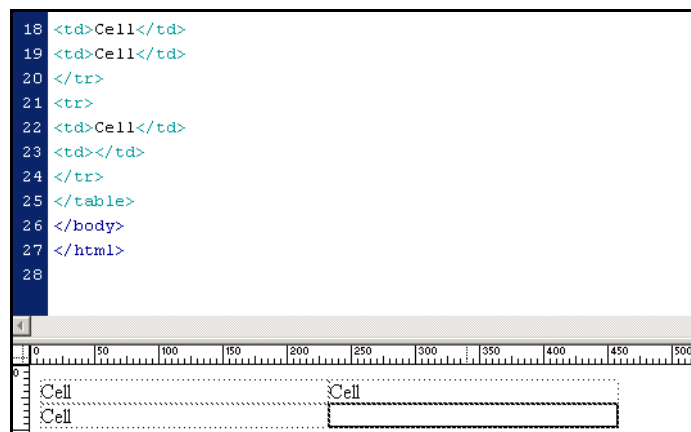


FIGURE 140

-Press **Ctrl + S** to save the page

¹ If you choose to allow Dreamweaver to keep the spaces, your page will still work fine. However, we feel that any unnecessary code is wasteful and have chosen to remove ours.

-With your cursor between the `<td>|</td>` tags, press your **spacebar** one time to create a *real* space

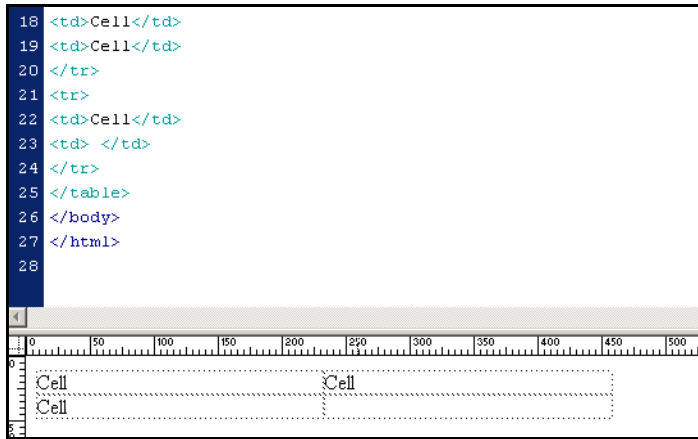


FIGURE 141

-Press **Ctrl + S** again

-Backspace to remove the space you just created

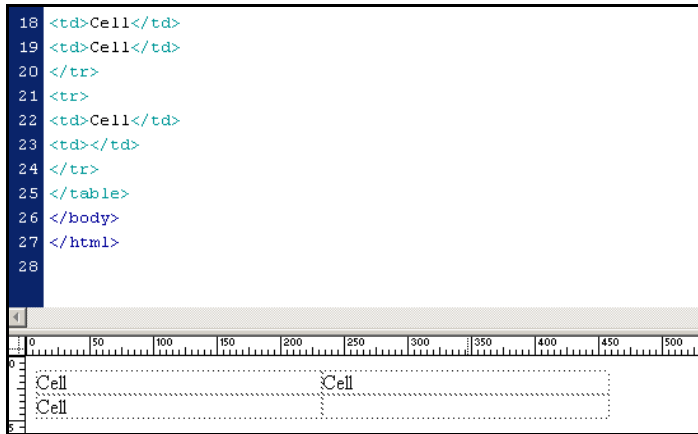


FIGURE 142

-Press **Ctrl + S** again

The pesky ** ** should be forever exorcized.

STYLING THE BODY ELEMENT

As we demonstrated earlier, using the *Page Properties* window to set body properties is not very efficient and the code it generates is incompatible with modern standards. We'll use CSS to style the `<body>` element of our page.

-Switch to Design View

-Open the CSS panel (Shift + F11)

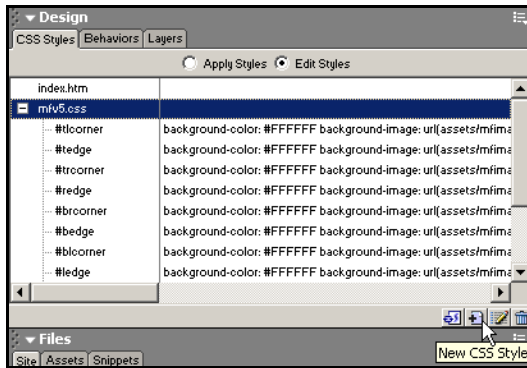


FIGURE 143

-Select (highlight) mf5v5.css

-Click the New CSS Style icon

The *New CSS Style* window will open

-For *Type*, select the *Redefine HTML Tag* radio button.

-Type the word **body** in the *Tag* box

-For *Define In*, select **mf5v5.css**

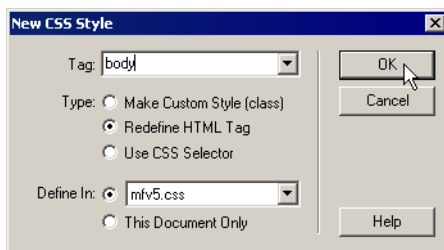


FIGURE 144

-Click **OK**

The CSS Style Definition window will open. Its title bar will reflect the rule you are editing. In this case: **body in mfv5.css**

The window will open to the *Type* Category.

-In the Font box list, choose the family: *Verdana, Arial, Helvetica, sans-serif*. This is one of several Font families included with Dreamweaver MX¹

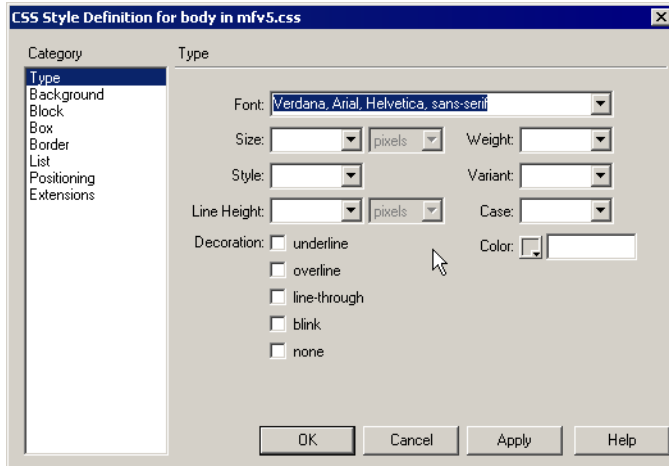


FIGURE 145

You can create your own font families, if you so desire... but the ones included with Dreamweaver MX were chosen for optimal compatibility across different operating systems and browsers.

A browser encountering the font-family: **Verdana, Arial, Helvetica, sans-serif**—in a CSS rule—will use the first one that has an exact match on the user's computer. Most modern PCs and Macs have Verdana. All PCs and some Macs have Arial. All Macs and some PCs have Helvetica. Since all computers have a default sans-serif font, it will be used if no matches are found.

Now select the *Background* Category

We'll set a Background Color and a Background Image for our page. However, let's do it in reverse order from what the window would seem to indicate.

¹ Some former UltraDev4 users who've upgraded to Dreamweaver MX have reported an anomalous Font-Family issue. When written to a style sheet, only font names that contain spaces should have containing quotation marks. For example, the fonts Arial or sans-serif should not have quotes around them. However, since there is a space in this font—"Trebuchet MS"—quotes are required. Be vigilant.

- Click the **Browse** button to the right of the *Background Image* box and browse to the image named **bgrid.gif** in your **mfimages** folder
- Click **OK** in the *Select Image Source* window and you'll return to the *Style Definition* window

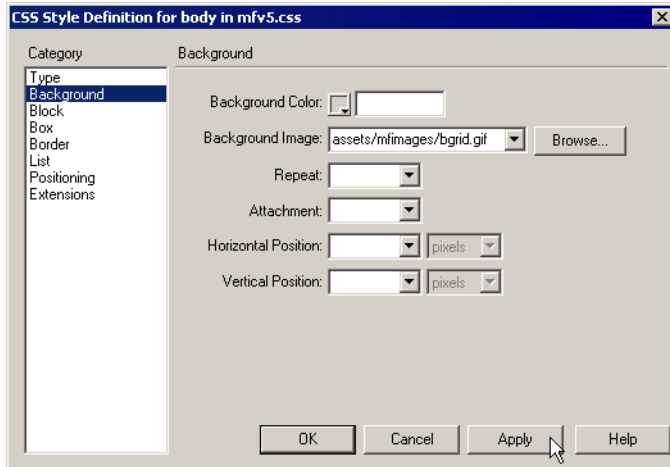


FIGURE 146

- Click the **Apply** button (the Style Definition window will remain open)

Your page will show the background image and the logo now blends in seamlessly.

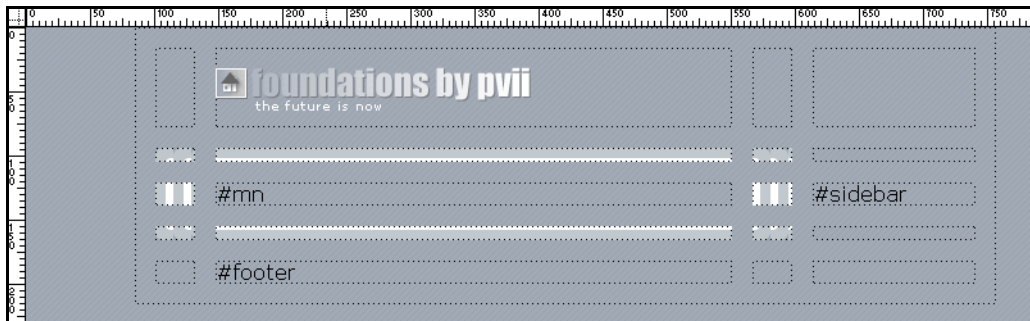


FIGURE 147

Leave the Style Definition window open...

-Click the color square between the Background color box's label and the box itself

The color palette will open.

-Click the little right-pointing flyout button on the top right edge of the palette

-The *Snap to Web Safe* feature is on (checked) by default. Select it to uncheck it. If it's already unchecked, then do nothing and press **Escape** to close the flyout menu

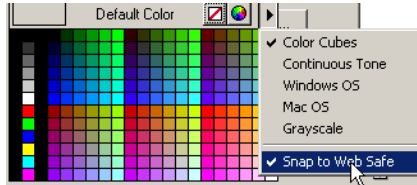


FIGURE 148

-Move your cursor towards the page. It will turn into an *eyedropper*  as you move off the palette area

-Move the eyedropper onto the blue page background. You'll see the color numbers on the palette's header changing as your cursor moves between the diagonal line pattern of the background

-Click on the page to “pick up” color **#A2AAB5**

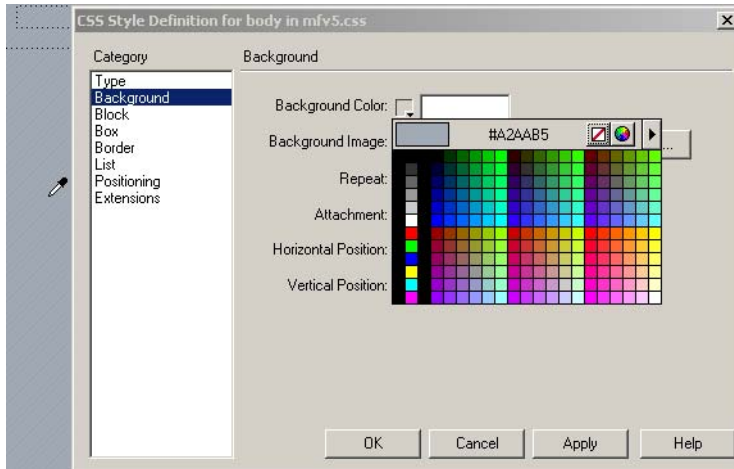


FIGURE 149

The color palette will close and **#A2AAB5** will be inserted into the Background Color box.

We set background color similar in shade to the background image. This color will appear instantly when the page is loading and will make your page look good while the background image is being loaded from the server.

-Now select the Box Category...

We're going to set page margins.

-Make sure the *Margin- Same for All* box is checked

-Enter *Top:0* and leave **pixels** set as the measure

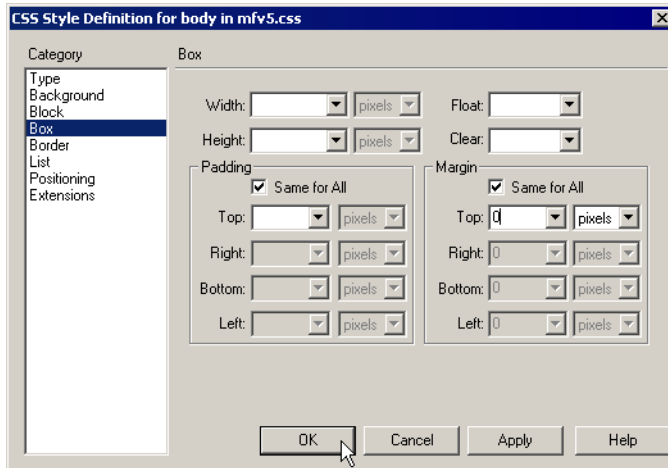


FIGURE 150

-Click **OK**

The space between the top ruler and the table top has closed to zero.

Open mfv5.css and you'll see our new body rule:

```
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
}
```

Close mfv5.css and switch back to index.htm

COLLAPSING THE TABLE

We can now safely remove the cell spacing from our table.

- Insert your cursor anywhere inside the table
- Click the **<table>** tag on Dreamweaver's tag selector bar
- In the Properties inspector/**CellSpace** box, replace the number **16** with **0**—then with your cursor still in the CellSpace box, press **Enter** to refresh the table

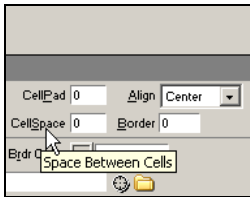


FIGURE 151

We're coming together here—



FIGURE 152

We're looking good except for the outer borders. Tables are funny creatures. They like to confuse us. We should forgive them because we are asking them to do something they were not designed to do. You may be thinking, "Let's just put shims in all the cells in the first and third columns." It won't work. However, two solutions *will* work:

1. We can simply add enough words to the main cell to cause it to make at least two lines. Am I serious? Yes. That will work. Moreover, I believe it's the cleanest solution.
- 2a. We can remove all width settings from the `<table>` tag. This will collapse the table to the narrowest width required to display your content.
- 2b. We can set the width of the `#mn` table cell to 100%. We can do that in one of two ways: we can set an HTML width property right on the `<td>` tag.

```
<td width="100%">
```

Setting the HTML property will fix things and Dreamweaver MX will actually render it perfectly. As odd as it seems, setting one cell of an 85% wide table to 100% does work. It really does.

- 2c. Or we can incorporate a width property set to 100% in the `#mn` ID Selector we'll soon be creating in our style sheet. This is a better solution because we are putting control of this issue in our style sheet—where it's most efficient. Dreamweaver will not, however, render this very well. It will make the main cell appear to be the width of the entire window, making it very hard to edit the page.

If you ever need to create a Foundations page with a main cell that won't have at least two lines of text, then I would suggest using either solution #2b or #2c. However, use it only on the pages where it is needed.

Speaking of our main cell, let's create our `#mn` selector...

CREATING THE #MN ID SELECTOR

-Click the New CSS Style icon at the bottom of your CSS panel. The New CSS Style window will open

-For *Type*, select **Use CSS Selector**

-Enter **#mn** in the *Selector* box

-*Define In*: **mfv5.css**

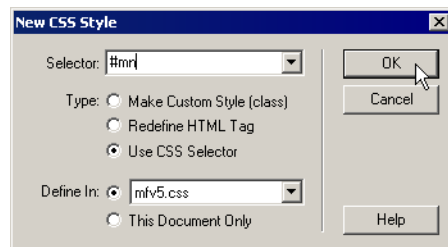


FIGURE 153

-Click **OK**

The CSS Style Definition window will open to the Type Category:

-For *Size*, enter **80** and select **%** as the measure

-For *Line Height*, enter **1.5** and select **multiple**¹ for the measure

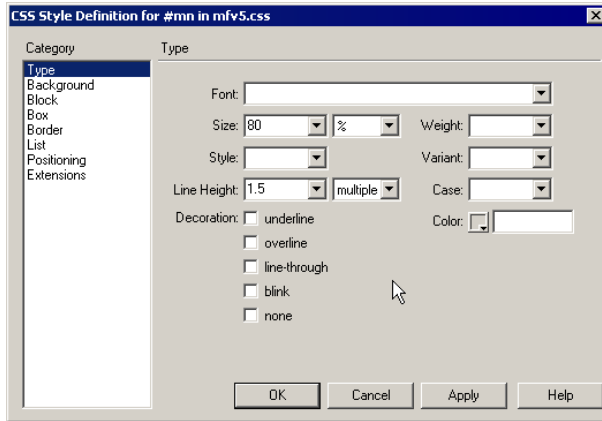


FIGURE 154

-Switch to the Background Category

-Set *Background Color* to: #FFFFFF

-Set *Background Image* to: **assets/mfimages/tbg.gif**

-For *Repeat*, select **repeat-x**²

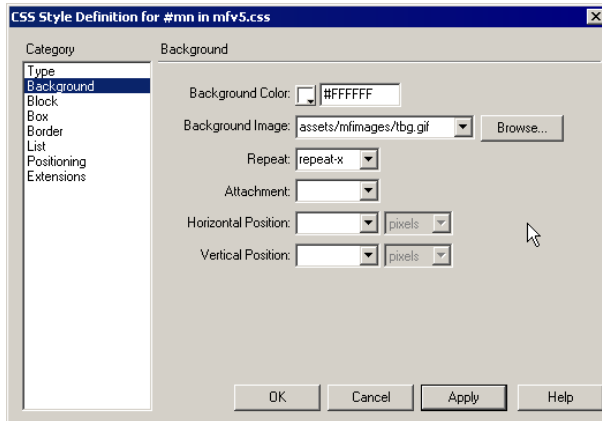


FIGURE 155

-Leave the window open and switch to the *Box* Category

¹ Setting line-height to a multiple is usually the most accurate because it causes modern browsers to use a more straightforward scaling algorithm, which avoids complex inheritance issues associated with *ems* or *percentages*. The only downside to a multiple is that IE3 will treat the multiple number as an explicit number. That is, a line-height set to 1.5 will be interpreted as 1.5 **pixels**. This won't be an issue for us, however, because we are only going to be using the line-height property in our version 5 style sheet.

² Setting a background image to Repeat-X means the image will tile from left to right (along the x-axis)—but not from top to bottom.

-Leave the *Padding- Same for All* box checked and enter **Top: 24** and **pixels** as the unit of measure

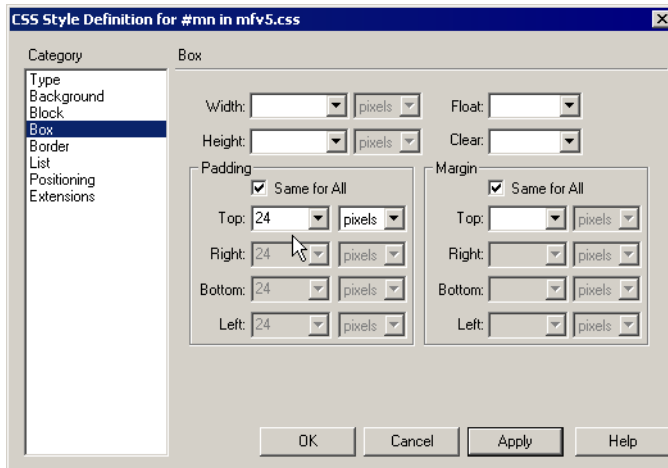


FIGURE 156

-Click **OK**

Here is our `#mn` rule:

```
#mn {
    font-size: 80%;
    line-height: 1.5;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
    background-color: #FFFFFF;
    padding: 24px;
}
```

The text in the main cell is not yet styled because we have not yet set an ID on the `<td>` tag. We'll do that in a moment after we add some text to the main cell.

Place your cursor in the cell, just to the right of the marker text `#mn` and press **Enter** once to create a new line¹. Dreamweaver automatically inserts opening and closing paragraph `<p></p>` tags around the lines.

-With your cursor on the new line, open the Snippets panel

-Click the **plus (+)** sign next to *PVII*

-Select the **Filler Text** snippet



FIGURE 157

-Click the **Insert** button

The text is inserted and... by crikey, the side borders have collapsed and now look downright perfect.

Dreamweaver does not refresh the page until you click somewhere outside the table. Be very, very careful to aim your pointer so as not to nudge the table borders.

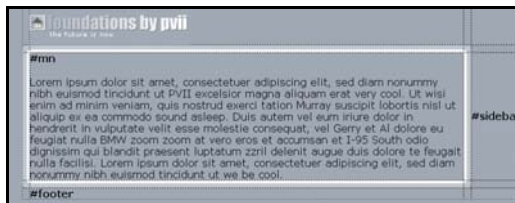


FIGURE 158

¹ When you type text into an element such as a `<td>`, in Dreamweaver, no tags are written around the text. However, as soon as you press enter, Dreamweaver will wrap each line of text inside paragraph `<p>` tags. This is a good feature most of the time.

NOW LET'S ID THE CELL—

-Insert your cursor anywhere inside the main cell

-On the *tag selector bar*, Right-click the `<td>` tag

-Choose **Set ID**→ **mn**

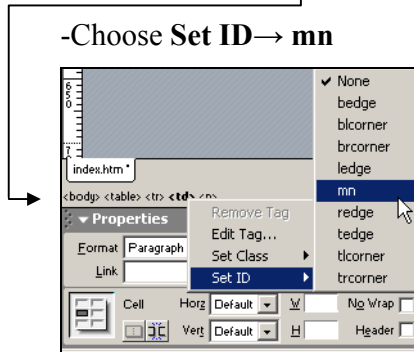


FIGURE 159

We have some style:

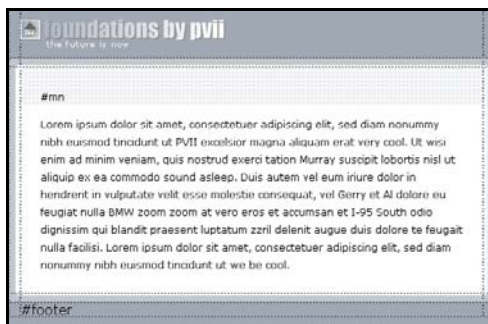


FIGURE 160

ADDING A LEVEL 1 HEADING

Our main content should have a level-1 Heading `<h1>` at the top, and probably should have more than one paragraph.

-Backspace over the marker text `#mn` and type: **Main Content**. Leave your cursor where it is

-On the Properties inspector, select **Heading 1** from the *Format* list

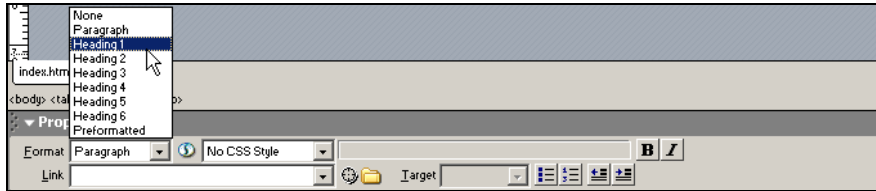


FIGURE 161

Main Content will become larger.

ADDING A PARAGRAPH

Let's add a second paragraph to the main cell.

-Place your cursor at the end of the filler-text and press **Enter** to create a new line (and a new paragraph)

-Type some text or insert the *PVII Filler* snippet again (you can also copy and paste the first paragraph)



FIGURE 162- WE NOW HAVE A SECOND PARAGRAPH

THE VERTICAL ALIGNMENT COMPROMISE

The sidebar marker is halfway down our page. When you use tables for layout, you have to take the good with the bad (and sometimes with the ugly). We can use CSS to fix this by setting vertical alignment to top, but Internet Explorer 4 (Mac) will ignore it. Since IE4 may well be as widespread in the Mac community as Navigator 4 is in the PC community, we really should fix this. Fortunately, the transitional Doctype we chose allows us to use a few presentational HTML properties for backwards compatibility.

-Insert your cursor inside the sidebar cell.

-On your Properties inspector, open the list next to the **Vert** label and choose **Top** (This sets the vertical alignment of the cell's contents to the top of the cell)

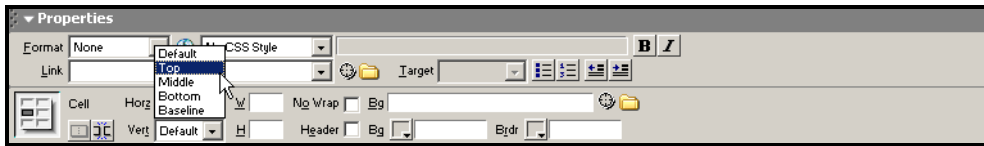


FIGURE 163

-Now repeat to set the main cell's vertical alignment to Top, as well

Why? Let's say you have a page where the content in the sidebar results in the sidebar being taller than the main cell. Guess where the main cell's content will start? *In the middle of the cell.*

Let's look at the code for the main sidebar cell:

```
<td valign="top" id="mn">
```

The HTML property `valign="top"` has been added.

STYLING THE H1

Now that we've fixed our vertical alignment issue in the HTML, we can move on to creating an `<h1>` style.

-Click the New CSS Style icon on your CSS panel

-For *Type*, select **Redefine HTML Tag**

-Enter **h1** in the *Tag* box or select it from the tag list

-Define In: **mfv5.css**

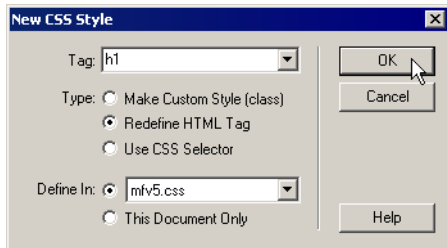


FIGURE 164

-Click **OK**

In the CSS Definition window, *Type Category*:

-For *Size*, Enter **180** and select **%** as the measure

-For *Line Height*, enter **1** and select **multiple** as the measure

-For *Color*, enter **#A2AAB5** (the same color as our page background)

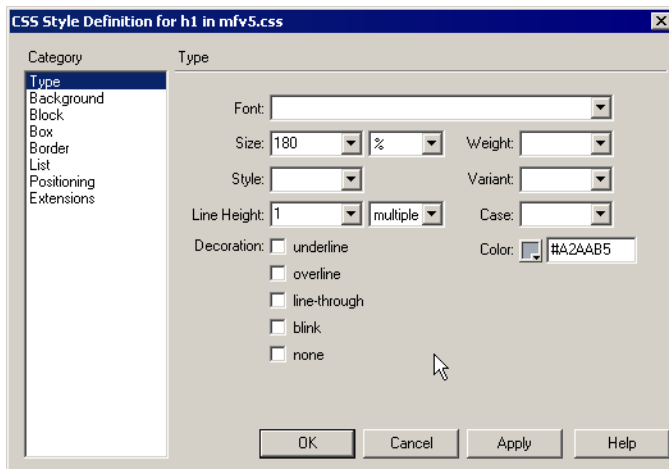


FIGURE 165

-Switch to the *Box* Category.

-Leave the *Margin- Same for All* box checked and enter *Top: 0* and **pixels** as the measure

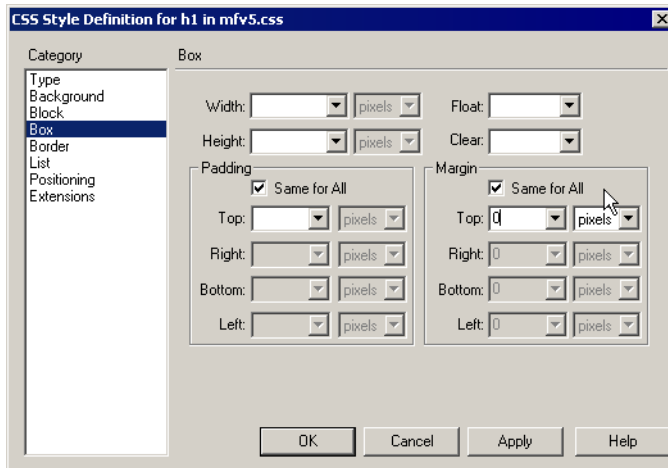


FIGURE 166

-Click **OK**

We set our heading's font size to **180%**, a bit more than double the size of our main text. We set line-height to **1** because if our heading ever wraps to two (or more) lines we want to keep the lines tightly spaced. Here is the rule:

```
h1 {
  font-size: 180%;
  line-height: 1;
  color: #A2AAB5;
  margin: 0px;
}
```

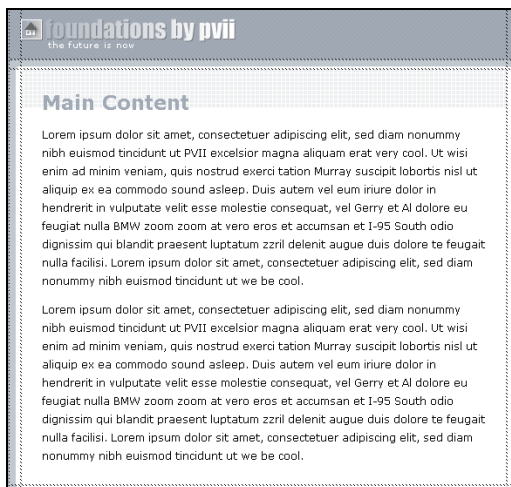


FIGURE 167

Dreamweaver doesn't quite get the positioning correct here. The bottom of the heading should be nearly even with the bottom of our cell background image. We'll have to live with that. However, positioning will be correct when viewed in a browser.

TIME FOR A BROWSER CHECK

Nervous? I feel fine ☺. But then again, I know that we'll have solutions for any problems that crop up. The Verdict:



FIGURE 168- FROM LEFT TO RIGHT: MSIE6 IS FINE. OPERA 6 IS FINE. NAVIGATOR 4 HAS A FEW FIXABLE PROBLEMS

The first thing that comes to mind is hear no evil, speak no evil, and *fugheaboutit*. It's not so bad. Navigator 4 just doesn't do CSS padding on table cells. We can't apply padding to the whole table because our borders would literally come apart at the seams. We don't want to nest a table if we don't have to. So what to do? Well, we do need to create a paragraph style for the main cell. There's too much space between them right now. We could use the left and right margin values of the `<p>` selector to give Navigator 4 breathing room on the sides. Then we'll need to give the `<h1>` selector a top margin. However, these edits will compromise our version 5 style sheet¹.

The solution is that we keep moving forward focusing only on version 5 and higher browsers. In fact, we won't be previewing again in Navigator 4 until we're finished with the version 5 style sheet... at which time, we'll create a duplicate style sheet for version 4 browsers, and do some editing until it works. *And it will work* ☺

¹ Bear in mind that there will come a day (perhaps within a year of this book's publish date) when version 4 browsers will be about as common as 5-legged goats. By limiting compromises in your HTML and in your version 5 style sheet, you are, in effect, extending the time that your page can exist before having to undergo major structural changes.

NETSCAPE 6, NETSCAPE 7, MOZILLA



FIGURE 169

Mozilla (Netscape 6 and 7) chose to interpret CSS specifications very rigidly. Images are treated as inline items—making the browser wrap a line.

Testing indicates that Netscape 7 for OSX does not have this problem. However, we do have to fix the issue for Netscape 6, Mozilla, and Netscape 7 (PC).

To fix this, we'll use a CSS *Child Selector* to target images whenever their direct parent is a `<td>` tag. This is the most efficient workaround, as it requires a single global rule: `td >img {display: block;}`

The rule tells compliant browsers to treat images that are first generation children¹ of `<td>` elements as block level elements (elements that have line breaks before and after them). This will eliminate the extra space at the bottom.

¹ Genealogical metaphors are, in fact, the language of CSS. A first generation child of a `<td>` would be any tag within the `<td>` whose direct parent is that `<td>`. In `<td></td>`, the image is a first child and the `<td>` is its parent. In `<td><p>Hello World</p></td>`, the `<img>` tag is actually a grandchild of the `<td>`.

CREATING A REMEDIAL CSS RULE

Let's create our image fix for Mozilla browsers.

-Click the *New CSS Style* icon on your CSS panel to open the New CSS Style window

-In the *Type* section, select **Use CSS Selector**

-In the *Selector* box, type¹: **td > img**

-Define In: **mfv5.css**

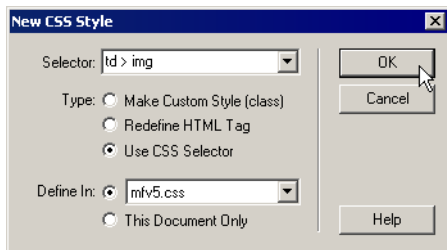


FIGURE 170

-Click **OK** and the CSS Style Definition window will open

-Switch to the Block category

-Open the Display box list and choose **block**

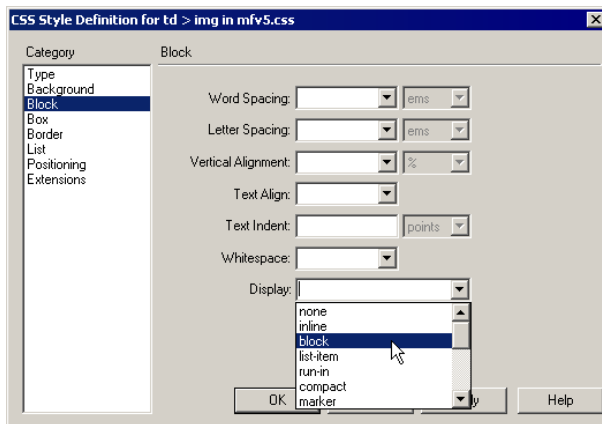


FIGURE 171

-Click **OK**

-Open mfv5.css and you'll see our new style at the bottom:

```
td > img {
    display: block;
}
```

¹ There are no requirements for white space around the *greater than* sign (>). The selector can be written with a space on either side of the symbol, on one side only, or with no space at all.

-Switch to **index.htm** and preview in Netscape 6 or higher (or Mozilla)

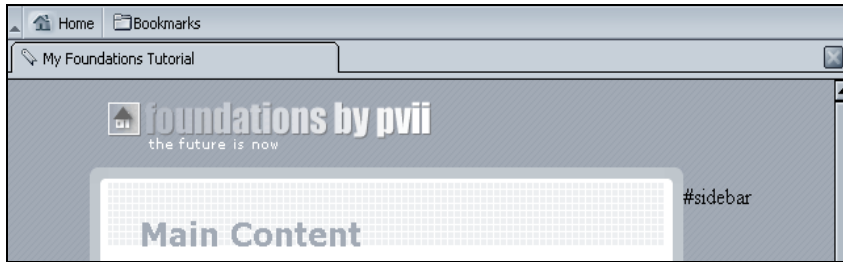


FIGURE 172

Now that's *much* better.

STYLING THE <P>

We have a pleasing amount of white space around the inside perimeter of our main cell. Now we need to tackle the issue of how much space is between each paragraph. We'll use margins. Since we only want to affect paragraphs in the main cell, we'll write a very specific selector called a *descendant selector*¹. The syntax is **#mn p**. This syntax tells the browser to affect only paragraphs that are inside of the element that has been assigned the id **mn**. This technique allows us to accomplish our goal while leaving ourselves free to have our way with paragraphs elsewhere on the page.

If you're up to it, let's write the rule manually. It's good practice.

-Open or switch to **mfv5.css**

-Scroll down to the **#mn** rule and place your cursor right after the closing squiggly bracket

-Press Enter to create a new line

-Type **#mn p** for your selector, press the space bar once and type in your opening squiggly bracket

```
#mn p {
```

-Press Enter to create a new line, then press Tab once to indent the line

-Type **margin: 12px 0px 0px;**

Pop quiz: do you remember how three values are interpreted for the 4 margins? If not, go back to [here](#)

-Press Enter to create a new line, press Backspace once to remove the tab space, and type in your closing squiggly bracket

-Press Enter again to create a new line for the next rule added

¹ Making a descendant selector based on an ID is very specific and very powerful, but these types of selectors can also descend from classes or even from other elements. For example: **.myclass strong {color: red;}** colors red *only* strong text within an element assigned the **.myclass** class. **ul ul {color: blue;}** colors blue an unordered list nested inside another unordered list.

This is how the finished rule looks:

```
#mn p {
    margin: 12px 0px 0px;
}
```

We used shorthand syntax. As we learned before, when we have three values listed, the first indicates **Top**, the second is both **Right and Left**, and the third is **Bottom**.

SAVE AND SWITCH

-Please choose **File**→ **Save** to save your changes to the CSS file—then switch back to the Foundations index.htm page

You can leave mvf5.css open in this instance, as we won't be doing any editing to the index page.

These margin values seem to work very nicely. Let's have a look:



FIGURE 173- LEFT TO RIGHT: MSIE 6, MOZILLA 1.1, OPERA 6

All three browsers look good, although Opera has a bit more space at the top—above the logo. Seems that zero <body> margins are not all that Opera needs. This browser also has a default <body> padding that needs to be turned off.

-Switch back to **mfv5.css** and scroll down to the body selector.

-Make a new line between the margin property and the closing squiggly bracket, press **Tab** once and type: **padding: 0px;**

```
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
```

Note: Setting padding to 0 on the body selector only affects Opera browsers.

Let's have a before and after look in Opera:

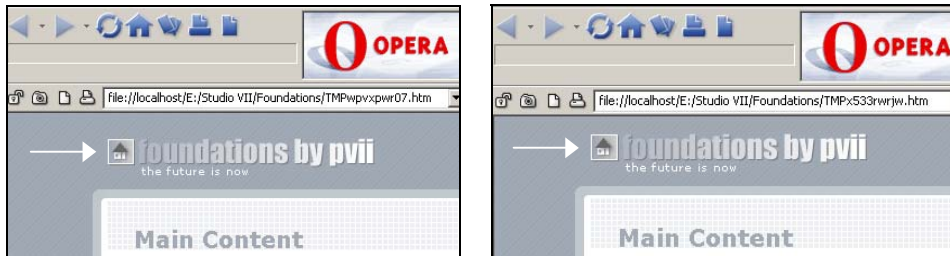


FIGURE 174- LEFT TO RIGHT: OPERA 6 BEFORE AND OPERA 6 AFTER SETTING BODY PADDING TO ZERO

Our CSS edit works.

-**Save and Close** mfV5.css

STYLING THE FOOTER ID SELECTOR

Footers are the perfect spot for copyright notices and contact information. Footer information is generally set to a smaller type size than the main content. Let's make our footer selector.

-Click the New CSS Style icon in your CSS panel

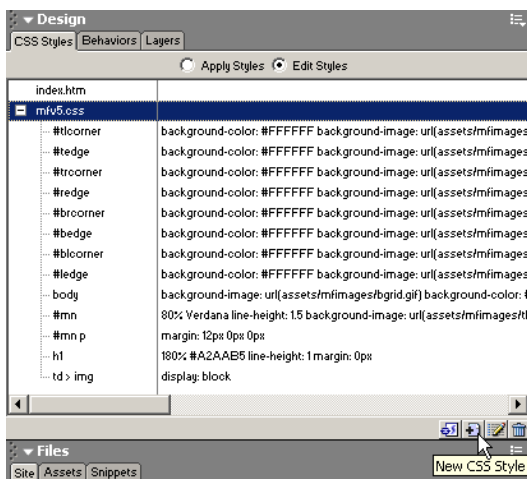


FIGURE 175

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#footer**

-Define In: **mfv5.css**

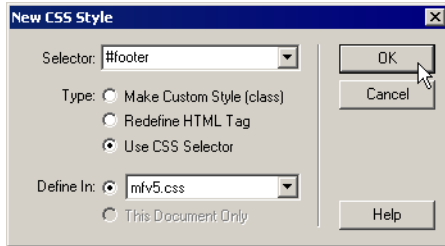


FIGURE 176

-Click **OK**

The CSS Style Definition window will open to the *Type* Category.

-Set *Size* to **70** and select **%** as the measure

-Set *Line Height* to **1** and **multiple** as the measure

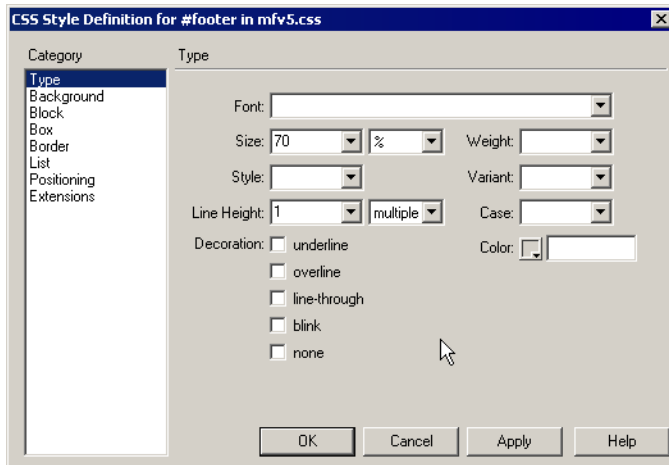


FIGURE 177

-Switch to the **Box** Category...

-**Uncheck** *Same for All* in the Padding section

-Enter **Top: 20 pixels** | **Right: 0 pixels** | **Bottom: 24 pixels** | **Left: 24 pixels**

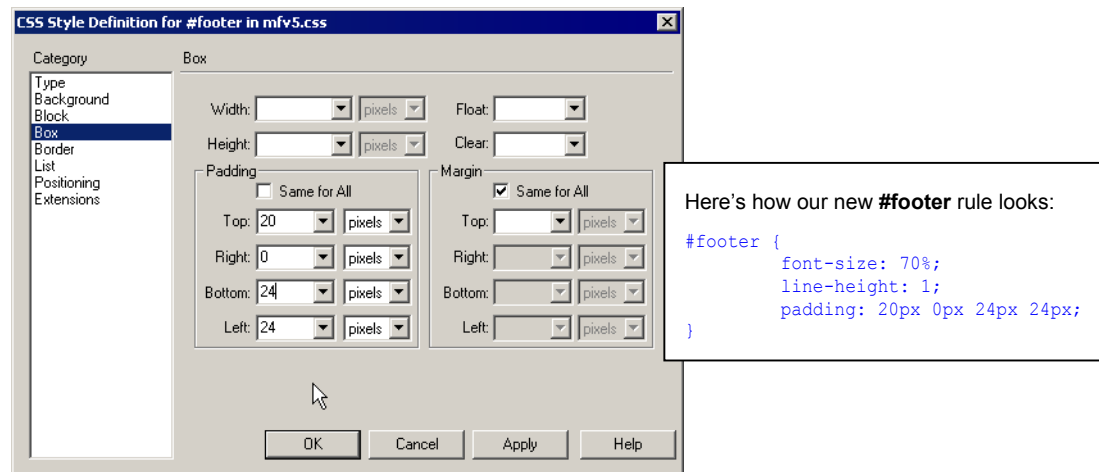


FIGURE 178

-Click **OK**

ID THE FOOTER <TD>

- Insert your cursor inside the footer cell
- Right-click the <td> tag on the Tag Selector bar
- Choose **Set ID**→ **footer**

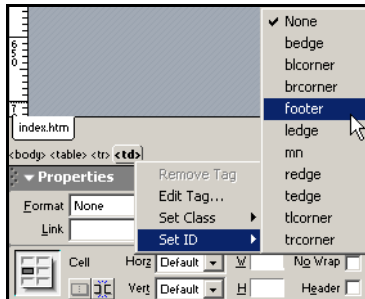


FIGURE 179

- Replace the marker text *#footer* with a copyright notice: **(c) Copyright 2002 Foundations Corp. All Rights Reserved.**¹

- Use the Properties inspector to change the format of the line to: **Paragraph**

Text size is set 10% smaller than the main body text, line-height is set to 1, and padding creates vertical white space, as well as horizontal alignment to the main content.

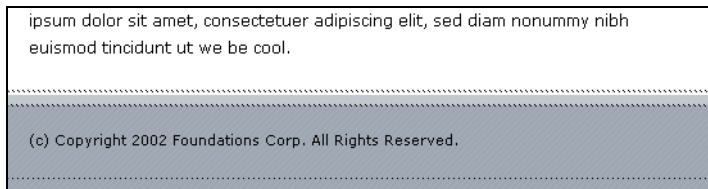


FIGURE 180

¹ I prefer using (c) rather than © because it's usually easier to read at small type sizes. If you want to use a real copyright symbol, choose Insert→Special Characters and select it from the list. Dreamweaver will insert **©** in your code to designate the copyright symbol.

ADD AN ADDRESS

Did you know that HTML has an `<address>` tag? It's the perfect tag to use when listing your *address*. Let's use it.

Note: More and more Search Engines are indexing `<address>` tags. If someone searches for *Foundations in Ohio*, our page would probably get lots of hits!

-Place your cursor at the end of the copyright notice (after the period) and switch to Code View

-Find the `#footer <td>`, which looks like this:

```
<td id="footer"><p>(c) Copyright 2002 Foundations Corp. All Rights Reserved.</p></td>
```

-Edit the `<td>` code to make it look precisely like this¹:

```
<td id="footer"><p>(c) Copyright 2002 Foundations Corp. All Rights Reserved.</p>
<address>
Foundations Corp.<br>
2222 Mockingbird Lane<br>
Ravenswood, OH U.S.A. 44444<br>
1-800-555-1212
</address>
<p>General E-Mail</p></td>
```

Note: We use the `<br>` tag to insert soft line breaks between lines. If you ever need to use soft returns while editing in Design View, press Shift + Enter.

-Switch back to Design View

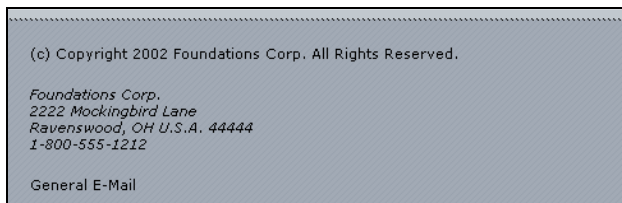


FIGURE 181

Now let's make an `<address>` selector.

¹ Dreamweaver is not able to correctly use the `<address>` tag in its visual editing mode. Please don't even try. Dreamweaver assumes that your `<address>` tag will contain a single line of text and no paragraphs either within the tag or after it. Pressing your Enter key in Design View after a line that is wrapped inside `<address></address>` tags, for example, will cause Dreamweaver to write multiple `<address>` tags for each line, which is not what we want it to do (and a clear case of code bloat).

CREATING AN <ADDRESS> SELECTOR

Most browsers display <address> text in italics. Let's change that to display normal (straight) text in white.

-Click the New CSS Style icon in your CSS panel

The New Style window will open.

-For *Type*, choose **Redefine HTML Tag**

-In the *Tag* box, enter **address** or select it from the tag list

-Define In: **mfv5.css**

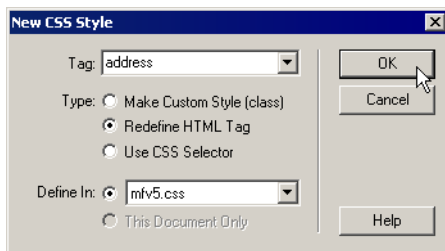


FIGURE 182

-Click **OK**

The CSS Style Definition window will open.

-For *Style*, select **normal**

-For *Color*, enter **#FFFFFF**

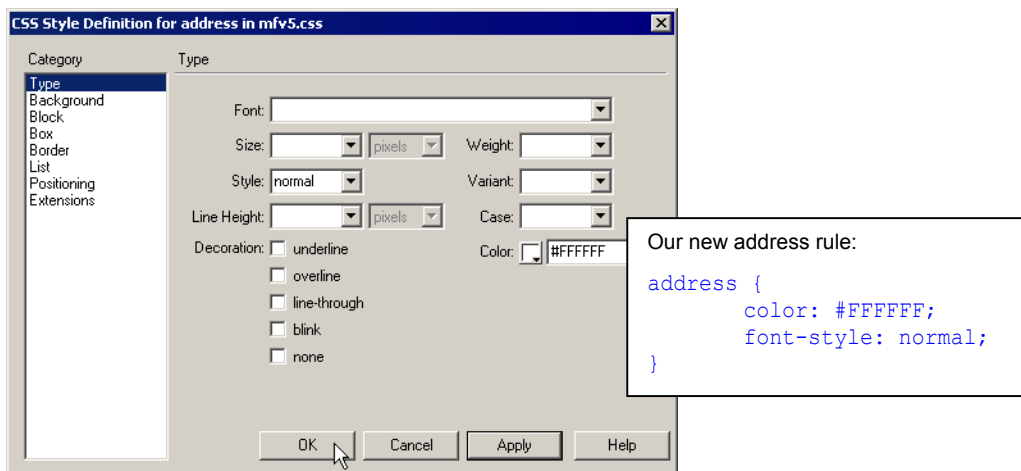


FIGURE 183

-Click **OK**

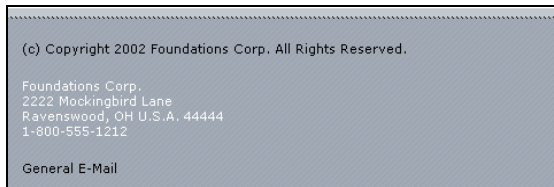


FIGURE 184- THE ADDRESS TEXT AFTER THE ADDRESS SELECTOR IS MADE.

Since we added the text: *General E-Mail*, let's go ahead and turn it into a temporary link. It'll allow us to explore a neat new feature in Dreamweaver MX.

-Drag to select the entire text string: *General E-Mail*

-Press **Ctrl + L** (Modify→ Make Link)

The Select File window will open.

-Type an **octothorpe** (#)¹ into the *File name* box. Upon entering the octothorpe, the URL box will change to display #

-Leave *Files of type* set to: **All Files (*.*)**

-Make sure *Relative To* is set to: **Document**

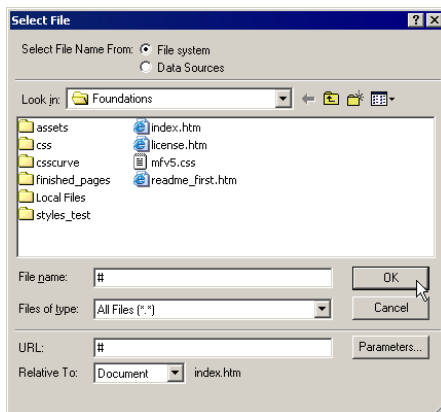


FIGURE 185

-Click **OK**

The text will appear as a default link (we'll be styling links a bit later on).

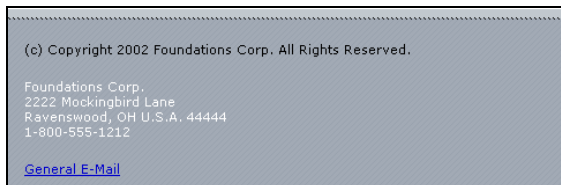


FIGURE 186- GENERAL E-MAIL IS NOW A HYPERLINK

¹ Using an octothorpe for a link creates a link that behaves visually like an ordinary hyperlink but does not load a new page. In many browsers, using the # link has the effect of scrolling a page back to the top. Use this method only for temporary links you know will be replaced with real links.

THE SIDEBAR ID SELECTOR

The sidebar cell will contain our navigation links. It can also contain news and any other type of information we deem appropriate for it. Let's create the sidebar's ID Selector now.

-Click the New CSS Style icon in your CSS panel

The New Style window will open.

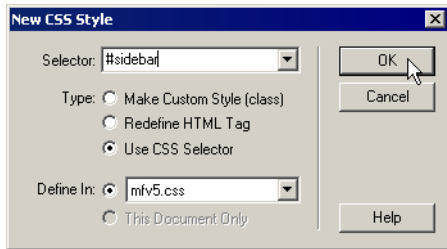


FIGURE 187

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#sidebar**

-Define In: **mfv5.css**

-Click **OK**

The CSS Style Definition window will open.

-In the *Size* box, enter **75** and set **%** as the measure

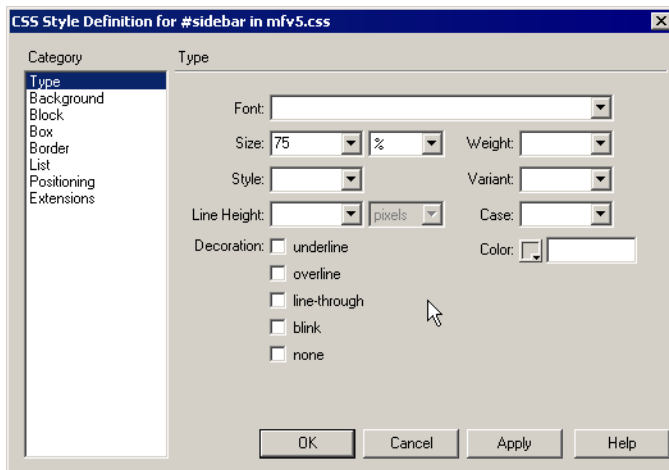
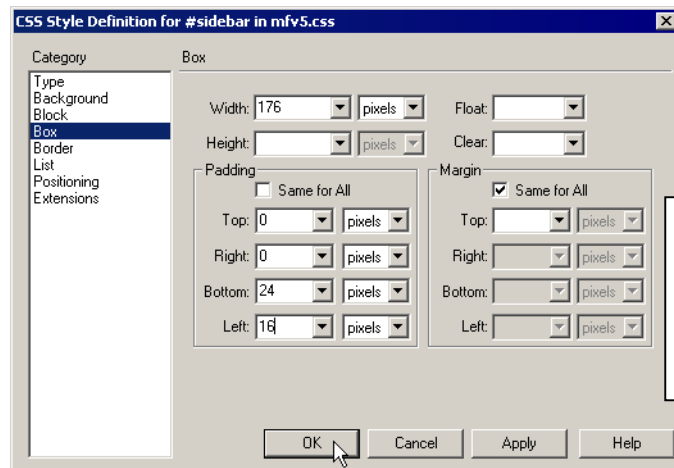


FIGURE 188

-Switch to the *Box* Category

-In the Width box, enter **176** and **pixels** for the measure

-Uncheck the *Padding/Same for All* box and enter: Top: **0 pixels** | Right: **0 pixels**
Bottom: **24 pixels** | Left: **16 pixels**



Our new #sidebar rule:

```
#sidebar {
    padding: 0px 0px 24px 16px;
    font-size: 75%;
    width: 176px;
}
```

FIGURE 189

-Click **OK**.

ID THE SIDEBAR CELL

-Place your cursor inside the sidebar cell

-Right-Click the <td> tag marker on the Tag Selector bar

-Choose **Set ID**→ **sidebar**

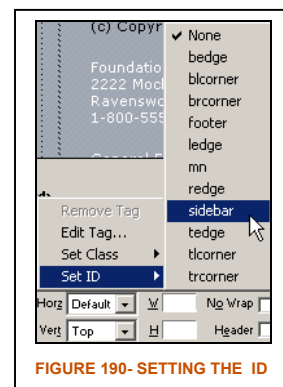


FIGURE 190- SETTING THE ID

The sidebar will expand to the declared width, the marker text will shift right as the cell padding is applied, and the text will reduce in size.

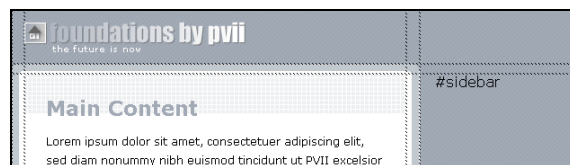


FIGURE 191

ADDING HYPERLINKS TO THE SIDEBAR

Let's add the hyperlinks for our main and sub-sectional navigation. They will appear as default links, but we will be styling them a bit later on.

-Backspace over the marker text *#sidebar* to remove it

-Choose **Insert**→ **Hyperlink**

-In the *Text* box, enter: **Section 1**

-In the *Link* box, enter: #

-In the *Tab Index* box, enter: 1

Tab Index: This property enables us to reset the default tab order of links on our page so that our main navigation links come first. That way, users navigating with the tab key will encounter our main navigation links before any other links.

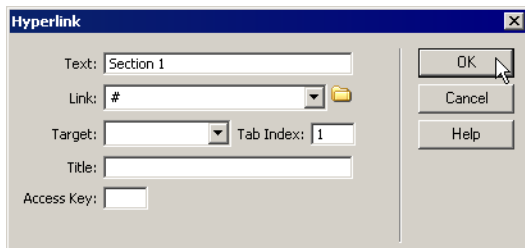


FIGURE 192

-Click **OK**

Section 1 is inserted as a hyperlink and is highlighted by Dreamweaver.

-Click your cursor just to the right of the **Section 1** link (so that the text is no longer highlighted) and press **Enter** to create a new line/paragraph

-Choose **Insert**→ **Hyperlink**

-Set the *Text* to: **Sublink 1**

-Set the *Link* to: #

-Set the *Tab Index* to: 2

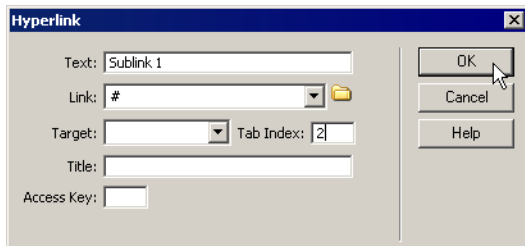


FIGURE 193

-Click **OK**

-**Repeat** the above process to create the following five additional links:

Sublink 2 | Link:# | Tab Index: 3

Sublink 3 | Link:# | Tab Index: 4

Section 2 | Link:# | Tab Index: 5

Section 3 | Link:# | Tab Index: 6

Section 4 | Link:# | Tab Index: 7

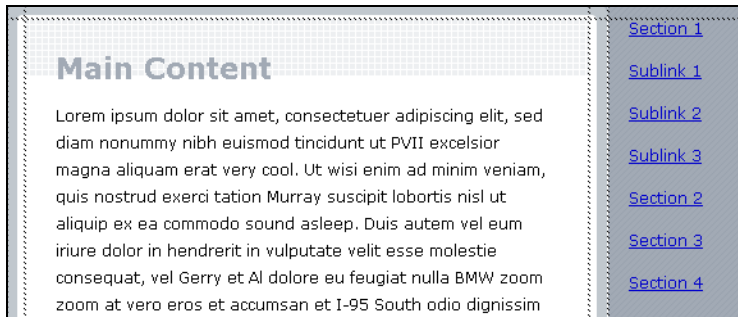


FIGURE 194: SEVEN HYPERLINKS ADDED TO THE SIDEBAR

Note: Although you can use the Properties inspector to create hyperlinks on existing text, the Insert→ Hyperlink method gives you the added ability to set Tab Index, as well as other properties that are not otherwise available. To access tag attributes at this level, you can also right-click the tag in Code View or you can use the Tag inspector (F9). To use the Tag Inspector, open its panel, and then select the tag using the Tag Selector bar.

Once the tag is selected, you will see all possible attributes and properties listed in the Tag Inspector panel.

CODE CHECK

Let's pause for a moment and inspect our table code.

-Switch to Code View—it should look like this:

```
<table width="90%" border="0" align="center" cellpadding="0" cellspacing="0" summary="Layout Table">
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>
<td id="tlcorner"></td>
<td id="tedge"></td>
<td id="trcorner"></td>
<td></td>
</tr>
<tr>
<td id="ledge"></td>
<td valign="top" id="mn"><h1>Main Content</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut
PVII excelsior magna aliquam erat very cool. Ut wisi enim ad minim veniam, quis nostrud exerci tation
Murray suscipit lobortis nisl ut aliquip ex ea commodo sound asleep. Duis autem vel eum iriure dolor in
hendrerit in vulputate velit esse molestie consequat, vel Gerry et Al dolore eu feugiat nulla BMW zoom
zoom at vero eros et accumsan et I-95 South odio dignissim qui blandit praesent luptatum zzril delenit
augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetur adipiscing elit,
sed diam nonummy nibh euismod tincidunt ut we be cool.</p>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut
PVII excelsior magna aliquam erat very cool. Ut wisi enim ad minim veniam, quis nostrud exerci tation
Murray suscipit lobortis nisl ut aliquip ex ea commodo sound asleep. Duis autem vel eum iriure dolor in
hendrerit in vulputate velit esse molestie consequat, vel Gerry et Al dolore eu feugiat nulla BMW zoom
zoom at vero eros et accumsan et I-95 South odio dignissim qui blandit praesent luptatum zzril delenit
augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetur adipiscing elit,
sed diam nonummy nibh euismod tincidunt ut we be cool.</p></td>
<td id="redge"></td>
<td valign="top" id="sidebar"><p><a href="#" tabindex="1">Section 1</a></p>
<p><a href="#" tabindex="2">Sublink 1</a></p>
<p><a href="#" tabindex="3">Sublink 2</a></p>
<p><a href="#" tabindex="4">Sublink 3</a></p>
<p><a href="#" tabindex="5">Section 2</a></p>
<p><a href="#" tabindex="6">Section 3</a></p>
<p><a href="#" tabindex="7">Section 4</a></p>
</td>
</tr>
<tr>
<td id="blcorner"></td>
<td id="bedge"></td>
<td id="brcorner"></td>
<td></td>
</tr>
<tr>
<td></td>
<td id="footer"><p>(c) Copyright 2002 Foundations Corp. All Rights Reserved.</p>
<address>
Foundations Corp.<br>
2222 Mockingbird Lane<br>
Ravenswood, OH U.S.A. 44444<br>
1-800-555-1212
</address>
<p><a href="#">General E-Mail</a></p></td>
<td></td>
<td></td>
</tr>
</table>
```

MAKING HYPERLINK STYLES

CSS is wonderful, but we have to be careful because it does enable us to style hyperlinks to the extent that they no longer look like links. This can confuse users. While stealth is a good quality for military aircraft, it's probably not the best quality for hyperlinks. Let's see if we can make links that are both stylish and **intuitive**.

There are four potential link styles or *states*:

1. **Normal [a:link]**- A link that has not yet been visited, or that is no longer in the browser's history.
2. **Visited [a:visited]**- A link that has been visited.
3. **Hover [a:hover]**- A link that the user has her mouse pointer over. Hover is a CSS2¹ property. In the context of the `<a>` tag, it's supported by version 5 and higher browsers (and also MSIE4). Navigator 4 simply ignores the property. Mozilla browsers also use the hover property when a link has been given focus by tabbing to it.
4. **Active [a:active]**- A link that has been clicked. The property is in effect while the mouse is being clicked and expires once the mouse button is released. For normal hyperlinks, active is usually not needed. It's not supported in CSS by Navigator 4 and newer browsers have default behaviors that draw focus boxes around links when they are tabbed to. MSIE (PC) also uses the active property for links that have focus. An example of a link in focus is one you have tabbed to using your keyboard's tab key. Mozilla browsers, on the other hand, treat this property correctly and a link is active only when a user is in the act of clicking it.

Foundations has three content sections: **#mn**, **#footer**, and **#sidebar**. We'll start with **#mn** and create specific link styles for each section

#MN HYPERLINKS

The **#mn** cell has a white background, so any dark-colored link will be legible and obvious. Since hyperlinks in the **mn** cell will tend to be within our text flow, we should underline them to make sure there is no doubt as to purpose. For the visited state, we'll set a dark gray color. For hover, we'll set the text to black—a subtle bit of visual feedback. Because of the way MSIE treats the active state, we'll simply make that a duplicate of the hover so that someone using his tab key for navigation will get the same visual clue provided by the hover state.

¹ CSS2 enables the hover property to be used on many elements, not just links. The rule `td.redcell:hover {background-color: red;}` will cause a table cell assigned the class **redcell** to turn red when the mouse is hovered over it. This level of support is currently available only in Mozilla browsers, including Netscape 6x. MSIE and Opera only support the hover class for `<a>` tags.

MAKE A TEST LINK

-Switch to Design View

-Place your cursor at the end of the first paragraph in the #mn cell, press your **spacebar** once

-Choose **Insert**→ **Hyperlink**

-In the *Text* box, type: **Read more...**

-In the *Link* box, type: #

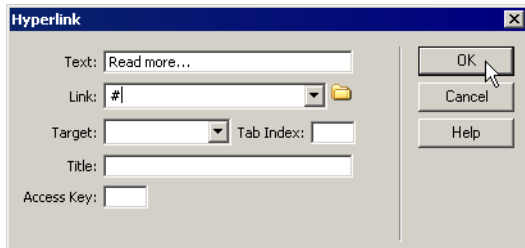


FIGURE 195

-Click **OK**

The link will be inserted and its style *default link blue*.

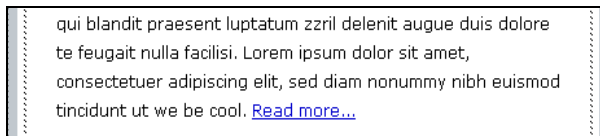


FIGURE 196

CREATING THE #MN LINK STYLES

THE #MN A:LINK CLASS

-Click the **New CSS Style** icon on your CSS panel to open the New Style window

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#mn a:link**

-*Define In:* **mfv5.css**

-Click **OK**

The CSS Style Definition window will open.

-For *Color*, enter **#0000CC**

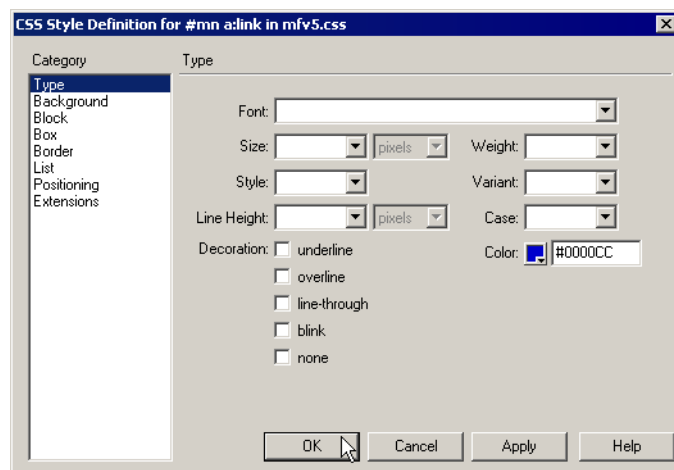


FIGURE 197

-Click **OK**

Our chosen color (#0000CC) is deep blue, a shade or so darker than the bright blue that most browsers set for default links.

#MN A:VISITED

-Repeat the above process to create the **#mn a:visited** style:

#mn a:visited—set *Color* to: **#666666** (dark gray)

#MN A:HOVER AND #MN A:ACTIVE

Since the hover and active classes are to be identical, we can combine¹ them into a single rule. Let's use Dreamweaver's editor to do this:

-Click the **New CSS Style** icon on your CSS panel to open the New Style window

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#mn a: hover, #mn a: active**

-*Define In*: **mfv5.css**

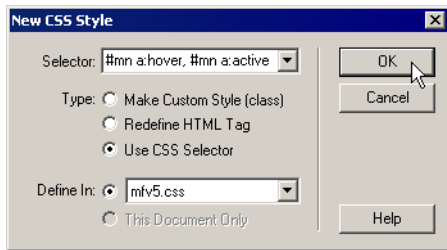


FIGURE 198

-Click **OK** (the CSS Style Definition window will open)

¹ Combining selectors is a wonderfully efficient way to make your CSS efficient and economical. Combined selectors are separated by commas. There is no effective limit on the number of selectors that can be combined.

-For *Color*, enter #000000

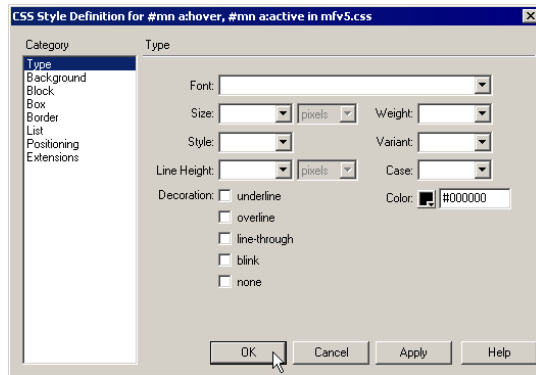


FIGURE 199

-Click **OK**

If you open mf5.css, you'll see our new selectors:

```
#mn a:link {  
    color: #0000CC;  
}  
#mn a:visited {  
    color: #666666;  
}  
#mn a:hover, #mn a:active {  
    color: #000000;  
}
```

THE FOOTER LINK STYLES

The `#footer` cell has no assigned background, so the blue textured page background shows through. Since it's likely that footer links would be constrained to a single line, we have a bit of creative latitude. We can change the link's background color on hover without being concerned that the linked text will break to two lines.

#FOOTER A:LINK

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#footer a:link**

-Define In: **mfv5.css**

-Click **OK**

The CSS Style Definition window will open.

-For *Color*, enter **#FFFFFF** (white)

-Click **OK**

#FOOTER A:VISITED

-Repeat the above process to create the visited link style

-In the *Selector* box, enter **#footer a: visited**

-Set its *Color* to: **#EEEEEE** (light gray)

#FOOTER A:HOVER, #FOOTER A:ACTIVE

Combine the `a:hover` and `a:active` classes into a single combined selector.

-In the *Selector* box, enter **#footer a:hover, #footer a:active**

Enter the following properties in the *Type* Category:

-Set *Color*: **#A2AAB5**

-Set *Decoration*: **none**

Switch to the *Background* Category.

-Set *Background Color*: **#FFFFFF**

PREVIEW

Since we already turned the text *General E-Mail* into a temporary link, preview in your default browser now to see the effect.

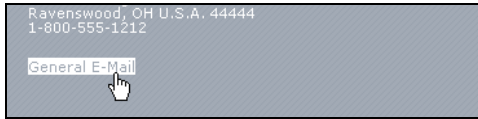


FIGURE 200- THE HOVER EFFECT IN MSIE6

-Open `mfv5.css` to see how the `#footer` link styles look:

```
#footer a:link {  
    color: #FFFFFF;  
}  
#footer a:visited {  
    color: #EEEEEE;  
}  
#footer a:hover, #footer a:active {  
    color: #A2AAB5;  
    text-decoration: none;  
    background-color: #FFFFFF;  
}
```

-Close `mfv5.css` and switch back to `index.htm`

THE SIDEBAR NAVIGATION STYLES

If you've been in the Web design game longer than twelve minutes, you've heard of JavaScript rollover buttons. CSS makes it possible to build rollovers that work the same way *without a single line of script*.

We'll assign CSS classes to the paragraph `<p>` tags wrapped around each link to power the navigation styles. Here is how the `#sidebar <td>` code looks like now:

```
<td valign="top" id="sidebar">
<p><a href="#" tabindex="1">Section 1</a></p>
<p><a href="#" tabindex="2">Sublink 1</a></p>
<p><a href="#" tabindex="3">Sublink 2</a></p>
<p><a href="#" tabindex="4">Sublink 3</a></p>
<p><a href="#" tabindex="5">Section 2</a></p>
<p><a href="#" tabindex="6">Section 3</a></p>
<p><a href="#" tabindex="7">Section 4</a></p></td>
```

We'll go through the entire process (step-by-step) in a moment and our efforts will result in code that will look like this:

```
<td valign="top" id="sidebar">
<p class="navbut"><a href="#" tabindex="1">Section 1</a></p>
<p class="navsub"><a href="#" tabindex="2">Sublink 1</a></p>
<p class="navsub"><a href="#" tabindex="3">Sublink 2</a></p>
<p class="navsub"><a href="#" tabindex="4">Sublink 3</a></p>
<p class="navbut"><a href="#" tabindex="5">Section 2</a></p>
<p class="navbut"><a href="#" tabindex="6">Section 3</a></p>
<p class="navbut"><a href="#" tabindex="7">Section 4</a></p></td>
```

If you're not yet fully sold on the benefits of using CSS—you *soon will be*.

The first step will be to create the **p.navbut**¹ custom class selector. Then we'll create the **p.navsub** class, the **navbut link styles**, and the **navsub link styles**. We'll also create a special ID selector because a small issue will arise ☺.

¹ A custom class prefaced with a tag name and a period (p.navbut) is called a Class Selector. The class **p.navbut**, therefore, deploys in such a way that any paragraph to which the navbut class is assigned will be selected. This type of selector has the added benefit of allowing us to style other paragraphs that are in the sidebar with more general rules—without affecting the paragraphs that house our navigation links. Class Selectors are more specific than Descendant Selectors (`#sidebar p`) or even Element Selectors (`p`) and thus take precedence in the event of conflicting values.

A NAVIGATIONAL FRAME OF REFERENCE

Let's imagine what our navigation links will look like once styled. The main links (navbut's) will have CSS borders and background colors that will make them look like buttons. To make them look even more like buttons, we'll use a background image in the navbut's *a:link* and *a:visited* classes. What good does using a background image do? Doesn't this defeat CSS efficiencies? Hmm... if we used Fireworks to make a navigation bar consisting of four main rollover buttons we'd need precisely eight images. We're going to accomplish the same feat with a single image for all four buttons. I'd say that's pretty darn efficient. Here's how it works:

The navbut's *a:link* and *visited* classes assign the image: **navbg.jpg** to the background of the link. The image (Figure below) is large enough to compensate should the user increase his browser's font size. It will always fill up the link's background—even if the link spans two lines. The image is under 2K, which is good; and it serves all four links, which is wonderfully efficient. Moreover, since it's done with CSS—JavaScript is not required.



FIGURE 201: NAVBG.JPG

When our link is hovered over, the background changes from an image to a nice, complementary solid color.

OUR FIRST TASK IS TO MAKE THE P.NAVBUTS CLASS SELECTOR.

This class will serve to position our navigation elements and set the text size.

-Click the **New CSS Style** icon in your CSS panel.

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **p.navbut**s

-Define In: **mfv5.css**

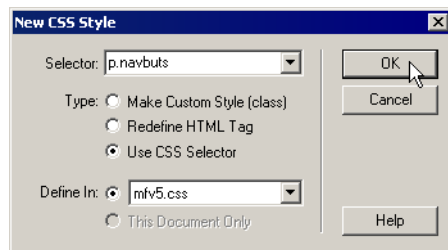


FIGURE 202

-Click **OK**

The CSS Style Definition window will open.

-For *Size*¹, enter **130** and **%** for the measure

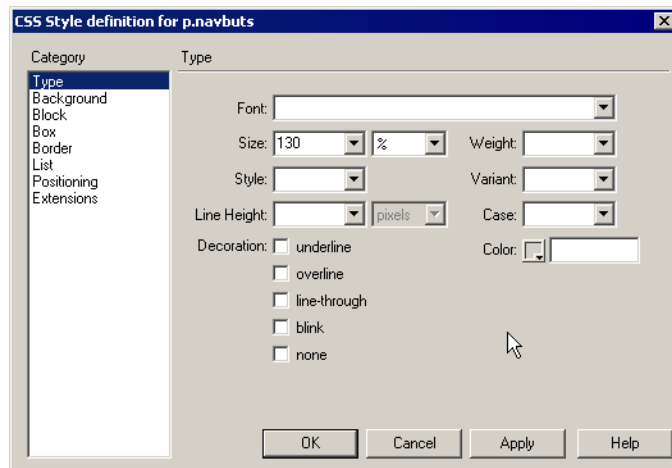


FIGURE 203

¹ 130%—can that be right? Since #sidebar declares that all text in this cell is 75%, then we are really talking about 130% of 75%, which is 98% of the browser's default. Font size is an inherited property so it's always a good idea to consult your style sheet whenever you have the urge to declare a font size to make sure you consider any inheritance issues that may be involved.

-Switch to the *Box* Category

-Uncheck *Margin/Same for All* and enter the following values in the Margin section:

Top: **0 pixels** | Right: **0 pixels** | Bottom: **4 pixels** | Left: **0 pixels**

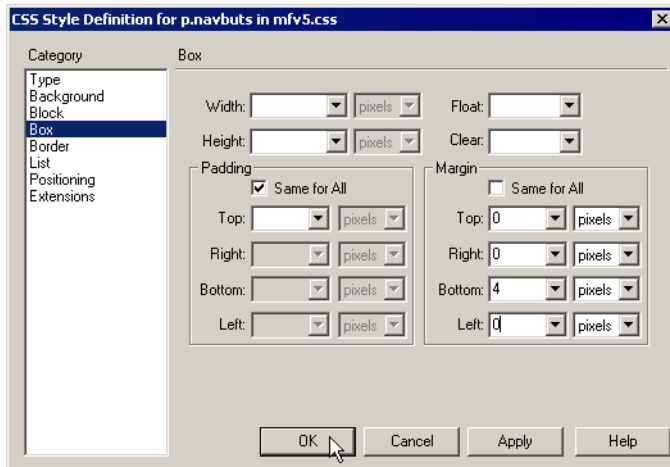


FIGURE 204

-Click **OK**

-Open `mf5.css` and you'll find a new rule:

```
p.navbutts {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}
```

-Leave `mf5.css` open

LET'S CREATE THE P.NAVSUBS SELECTOR MANUALLY IN MFV5.CSS:

-Use copy/paste to create a duplicate of the **p.navbut**s rule

```
p.navbut {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}  
p.navbut {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}
```

-Rename the copy's selector to: **p.navsub**s

-Change font-size to: 110%

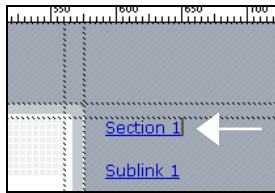
-Change the margin values to: 4, 0, and 2

```
p.navbut {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}  
p.navsub {  
    font-size: 110%;  
    margin: 4px 0px 2px;  
}
```

-**Save and Close** mfv5.css and switch back to index.htm

ASSIGN CLASSES TO THE MAIN AND SUB-SECTION LINKS

-Place your cursor just to the right of the text: **Section 1**.



Make sure you place your cursor to the right of the text. If you place your cursor inside the text block, you will have the `<a>` tag selected, instead of the `<p>` tag—which is the one we want to style.

FIGURE 205

-Open the CSS list on your Properties inspector and choose **navbut**s

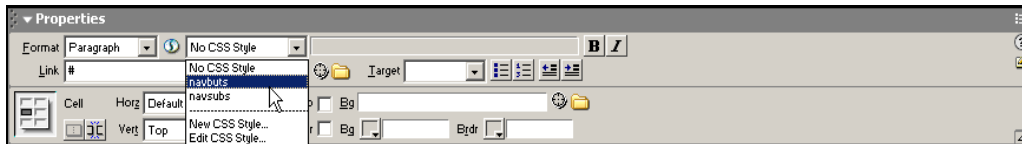


FIGURE 206

-Place your cursor just to the right of the text: **Sublink 1**

-Open the CSS list on your Properties inspector and choose **navsubs**

-Assign the **navsubs** class to the next two paragraphs, which contain the text: Sublink 2 and Sublink 3

-Assign the **navbut**s class to the remaining three paragraphs in the sidebar, which contain the text: Section 2, Section 3, and Section 4

When you've finished assigning the classes, the sidebar should look like this:



FIGURE 207

THE NBUTSPACER SELECTOR

Wouldn't it be nice if we could have a bit of space between the bottom of Sublink 3 and the top of Section 2? Should we press our Enter key to make a blank paragraph between them? No—we can use an ID Selector.

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **#nbutspacer**¹

-Define In: **mfv5.css**

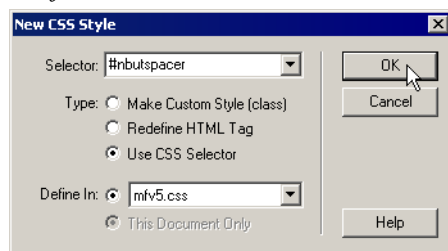


FIGURE 208

-Click **OK**

-When the CSS Style Definition window opens, switch to the *Box* Category

-Uncheck the Margin/Same for All box and enter *Bottom: 16 pixels*

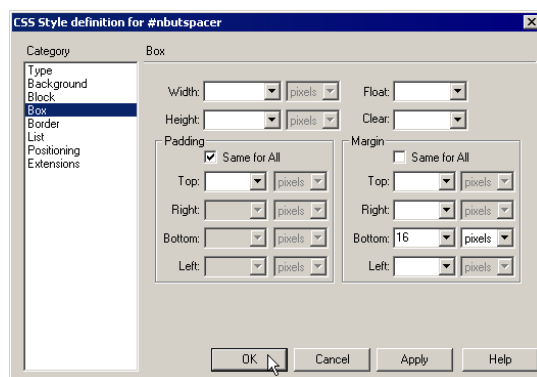


FIGURE 209

-Click **OK**

¹ The ID *nbutspacer* was chosen for its melodic sound after exhaustive research conducted by one of America's top ID consultancies. The Cleveland Conservatory is rumored to be working on the production of a new Symphony to be called the **Nbutspacer Suite**.

USING THE #NBUTSPACER ID

-Insert your cursor on the line that contains the text: **Sublink 3**

-Right-click the `<p.navsubs>` tag on the Tag Selector bar

-Choose **Set ID** → **nbutspacer**

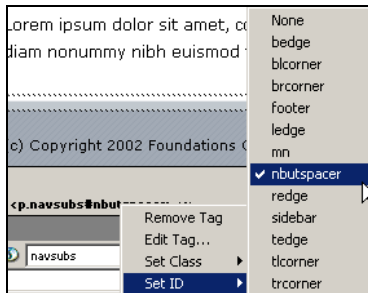


FIGURE 210

Here is the `nbutspacer` rule:

```
#nbutspacer {
    margin-bottom: 16px;
}
```

And here is how the paragraph code now looks:

```
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="4">Sublink 3</a></p>
```

The properties from the `navsubs` class and the `nbutspacer` ID selector are combined and we have some breathing space.

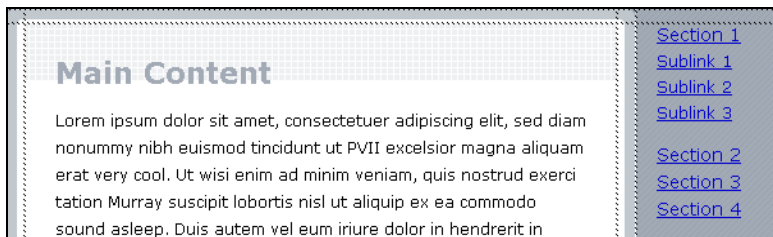


FIGURE 211

I know what you're thinking. *A submenu should be indented from its parent.* As you'll see in just a moment, we have a very creative way to accomplish that. We're ready to start lighting things up.

MAKING BUTTONS FROM TEXT PART 1: THE MAIN SECTION LINKS

Buttons from text; gold from straw. CSS does alchemy. Let's spin some buttons.

THE P.NAVBUTS A CLASS

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **p.navbutsa**

-Define In: **mf5v5.css**

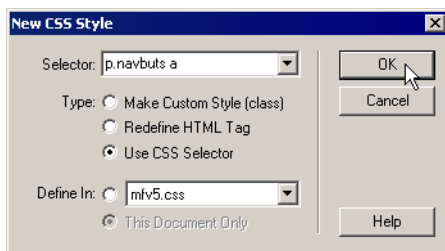


FIGURE 212

-Click **OK**

The Style Definition window will open.

-In the *Type Category*, check *Decoration: none*

-Set *Color: #000000*

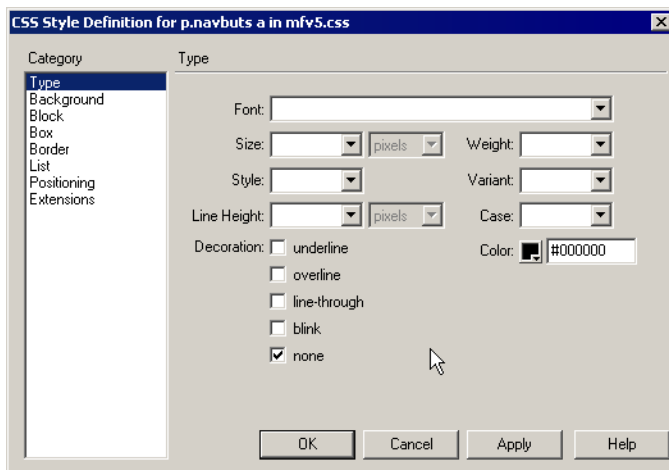


FIGURE 213

-Switch to the Background Category

-Set *Background Color*: #A2AAB5

-Set *Background Image*: assets/mfimages/navbg.jpg

-Set *Repeat*: no-repeat

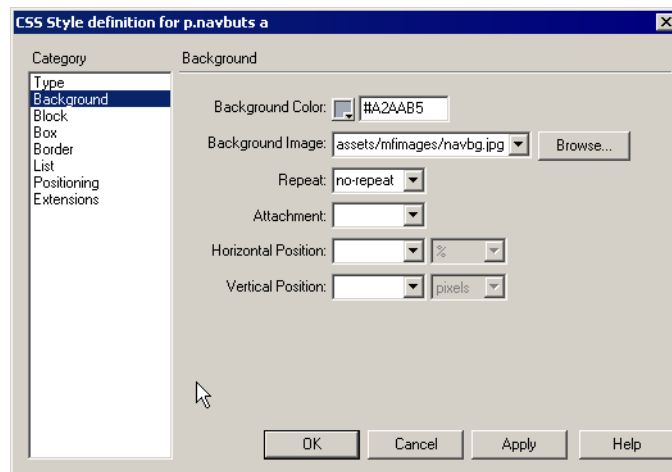


FIGURE 214

-Switch to the Block Category

-Set *Display*: **block**¹

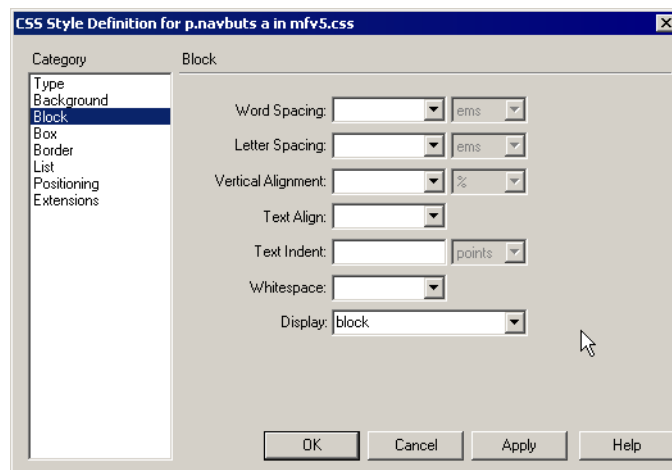


FIGURE 215

-Switch to the Box Category

¹ Setting the <a> tag to block makes it assume the width of its parent element (in this case, the entire paragraph it's in) and makes the entire *box* (as opposed to just the text) *clickable*.

-Set *Width*¹: **162** and **pixels** as the measure

-Uncheck *Padding/Same for All* and enter—

Top: 3 pixels | Right: 0 pixels | Bottom: 3 pixels | Left: 12 pixels

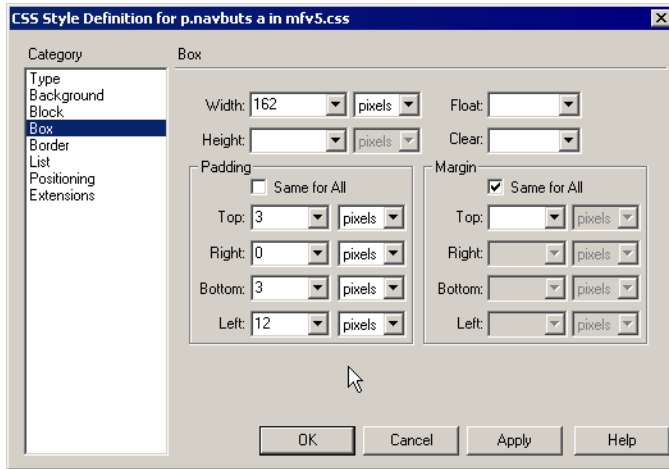


FIGURE 216

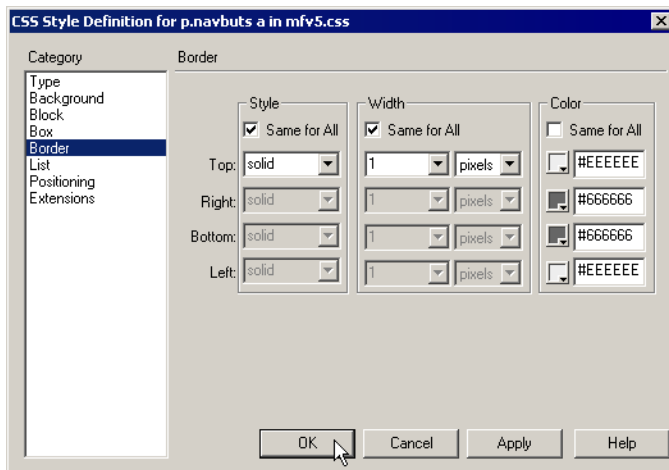
-Switch to the Border Category

-Leave *Style/Same for All* checked and set *Top: solid*

-Leave *Width/Same for All* checked. Enter **1** and **pixel** for the measure

-Uncheck *Color/Same for All* and enter—

Top: #EEEEEE, Right: #666666, Bottom: #666666, Left: #EEEEEE



Setting the top and left borders a light color and the right and bottom borders a dark color serves to simulate a raised 3-D effect. It makes our button look like a button!

FIGURE 217

-Click **OK**

¹ Width is set for the benefit of MSIE-PC to make the entire box *clickable*. But we need to be precise about how much width to set. We use a formula based upon the width of the parent element (#sidebar), which is 176px. Then we add up the left and right padding plus the left and right borders and subtract the total from the parent element's width: We have 12px padding on the left, a 1px border on the left, and a 1px border on the right. So our formula is: $176 - (12 + 1 + 1) = 162$.

Dreamweaver makes a valiant effort to display our CSS but falls a bit short. To fully appreciate what we’ve just accomplished, you’ll need to preview in your default browser (version 5 and higher).

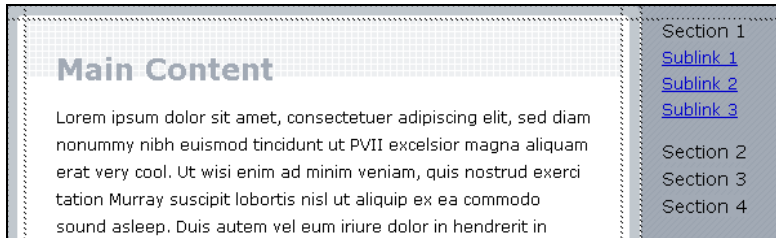


FIGURE 218

Here is the **p.navbut**s a rule:

```
p.navbut a {
    color: #000000;
    text-decoration: none;
    background-color: #A2AAB5;
    background-image: url(assets/mfimages/navbg.jpg);
    background-repeat: no-repeat;
    display: block;
    padding: 3px 0px 3px 12px;
    width: 162px;
    border-top: 1px solid #EEEEEE;
    border-right: 1px solid #666666;
    border-bottom: 1px solid #666666;
    border-left: 1px solid #EEEEEE;
}
```

The figure below shows our buttons in MSIE6, Mozilla 1.1, and Opera 6.

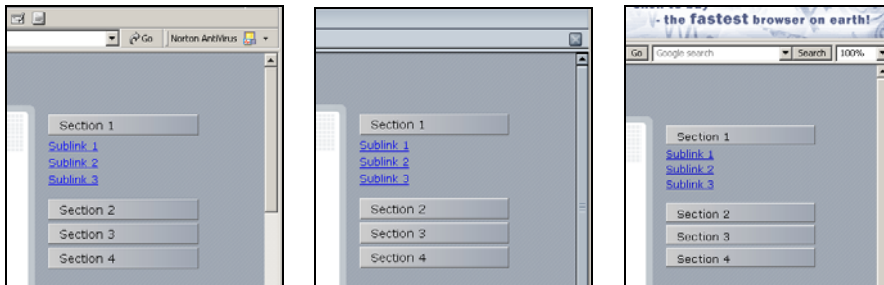


FIGURE 219—FROM LEFT TO RIGHT: MSIE 6 | MOZILLA 1.1 | OPERA 6

Don’t preview in Navigator 4¹. It won’t look good. We’ll fix that up later—in Chapter 9

Let’s do a hover style now.

¹ You did preview in Navigator 4, didn’t you? *I knew you would.*

THE P.NAVBUTS A:HOVER, P.NAVBUTS A:ACTIVE CLASS

We'll make a combined selector because we want the hover and active classes to be identical. As we mentioned earlier, MSIE uses the active state when a user tabs into the link.

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **p.navbut a:hover, p.navbut a:active**

-Define In: **mfv5.css**

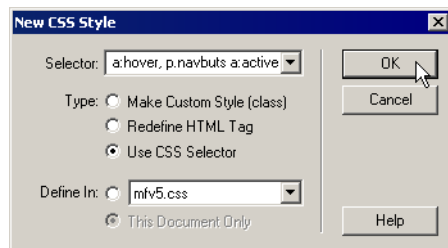


FIGURE 220

-Click **OK** to open the Style Definition window

-In the Type Category, set *Color*: **#FFFFFF**

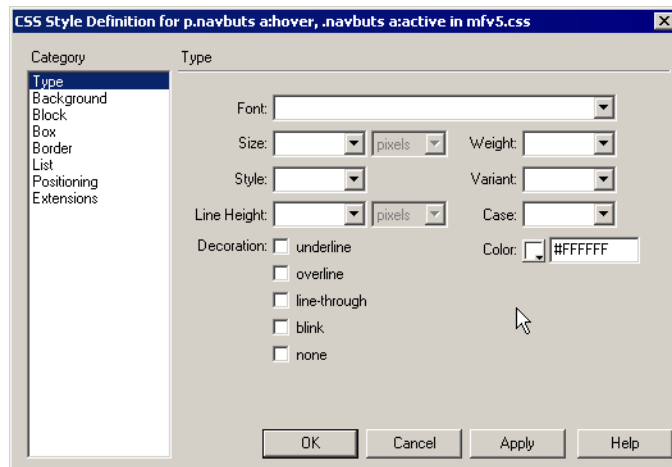


FIGURE 221

-Switch to the Background Category

-Set *Background Color*: #A2AAB5

-Set *Background Image*: none

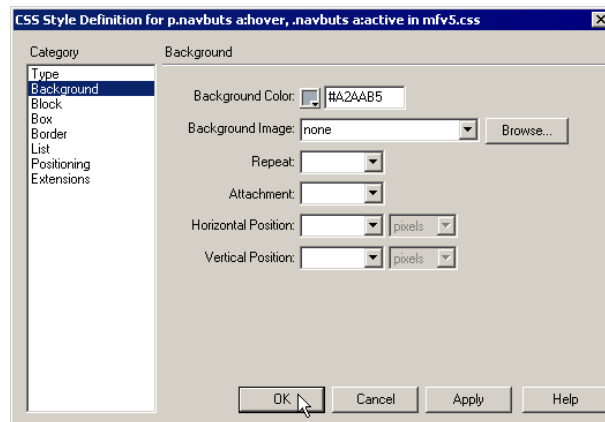


FIGURE 222

-Click **OK**

Here is the `p.navbut a:hover, p.navbut a:active` rule:

```
p.navbut a:hover, p.navbut a:active {
    color: #FFFFFF;
    background-color: #A2AAB5;
    background-image: none;
}
```

The `a:hover / a:active` rule inherits properties from the `p.navbut a` rule. That is, they are *carried over*. We want the borders to be the same, so we do nothing and the borders are inherited. We do not, however, want a background image for the hover/active classes—so we explicitly declare the background image to be **none**. The *background color* from `p.navbut a` should be inherited, and it is in MSIE and Mozilla browsers—but for some reason, Opera is not playing fair so we had to declare it again.

-Preview in your default browser



FIGURE 223

Hovering over a *button* turns text white and turns off the background image (we see just the background color). The borders remain the same—they are inherited from the `p.navbut a` rule.

We have lights and action.

MAKING BUTTONS FROM TEXT PART 2: THE SUB-SECTION LINKS

Are you having fun yet?

THE P.NAVSUBS A CLASS

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **p.navsubs a**

-Define In: **mfv5.css**

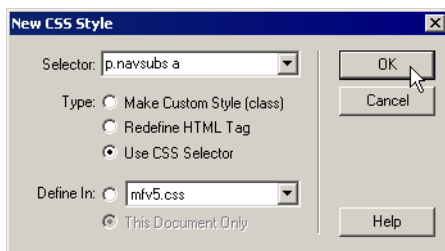


FIGURE 224

-Click **OK** to open the Style Definition window

-In the *Type Category*, check *Decoration: none*

-Set *Color: #000000*

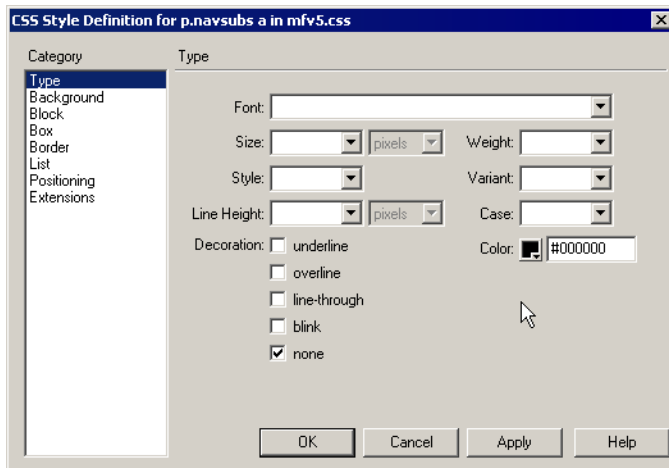


FIGURE 225

-Switch to the Block Category

-Set *Display*: **block**

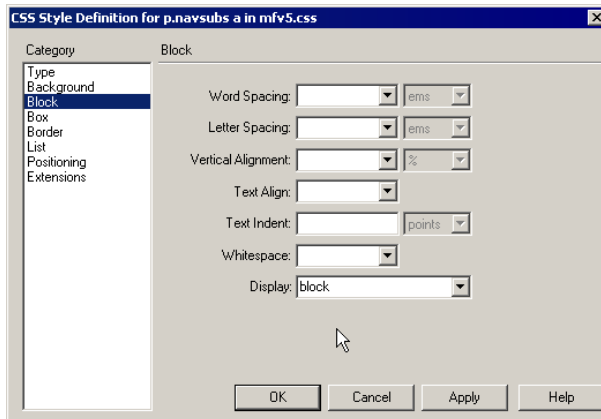


FIGURE 226

-Switch to the Box Category

-Set *Width*¹: **162** and **pixels** as the measure

-Uncheck *Padding/Same for All* and enter—

Top: **0** pixels | *Right*: **0** pixels | *Bottom*: **0** pixels | *Left*: **14** pixels²

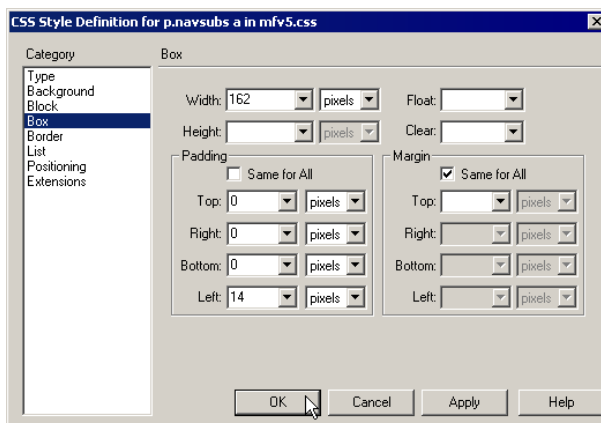


FIGURE 227

-Click **OK**

Here is the p.navsubs a rule:

```
p.navsubs a {
    color: #000000;
    text-decoration: none;
    display: block;
    padding: 0px 0px 0px 14px;
    width: 162px;
}
```

¹ We'll use the same formula we used for the p.navbut a class. We have 14px padding on the left and no borders So our formula is: 176 – 14 = 162.

² 14 pixels of left padding aligns the left edges of our sublinks and main links. How can that be if our main links have left padding set to 12 pixels? It's because the main links have a 1 pixel border on both the left and the right, while the sublinks have no border.

THE P.NAVSUBS A:HOVER, P.NAVSUBS A:ACTIVE CLASS

We'll make a combined selector again.

-Click the **New CSS Style** icon in your CSS panel

The New Style window will open.

-For *Type*, choose **Use CSS Selector**

-In the *Selector* box, enter **p.navsubs a: hover, p.navsubs a: active**

-Define In: **mfv5.css**

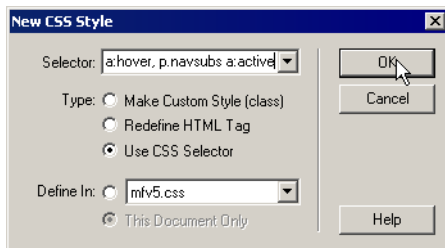


FIGURE 228

-Click **OK** to open the Style Definition window

-Set *Color*: **#A2AAB5**

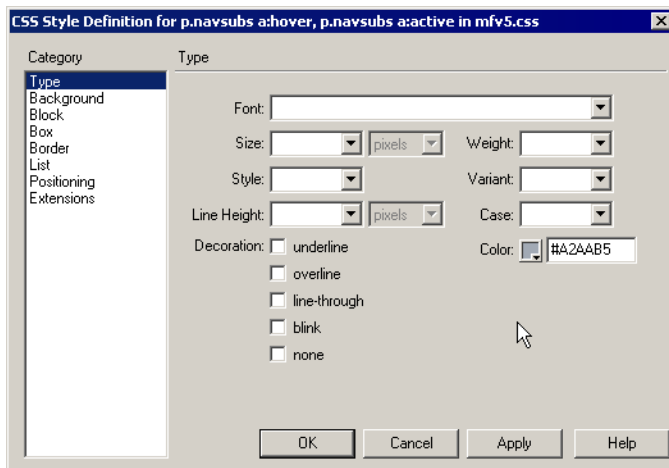
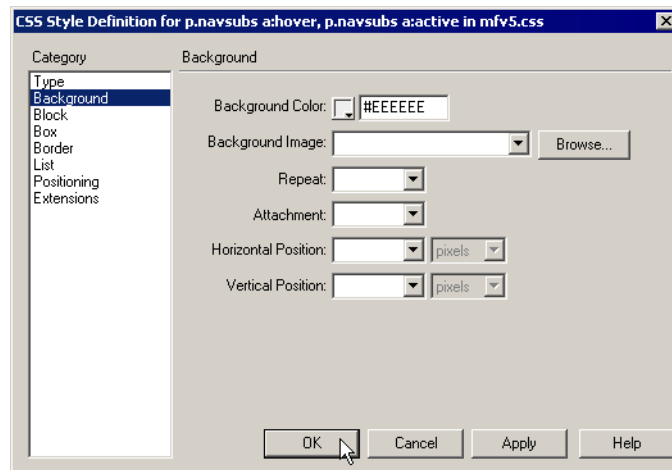


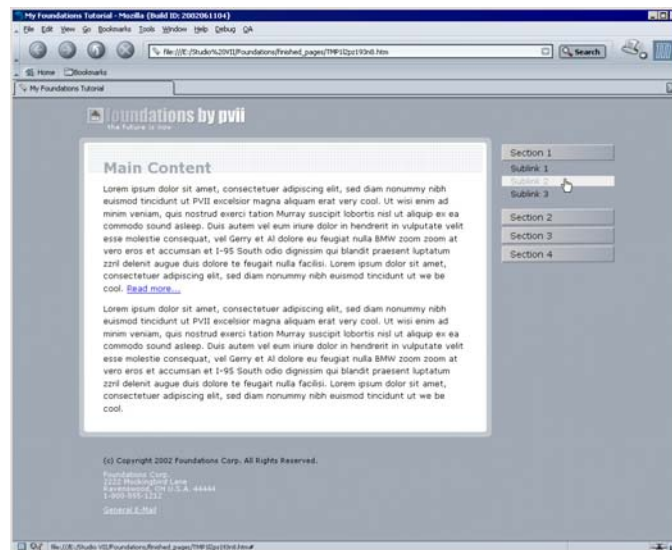
FIGURE 229

-Switch to the Background Category

-Set Background Color: #EEEEEE**FIGURE 230****-Click OK.**

Here's our p.navsubs a:hover, p.navsubs a:active rule:

```
p.navsubs a:hover, p.navsubs a:active {
    color: #A2AAB5;
    background-color: #EEEEEE;
}
```

-Preview in your default browser**FIGURE 231**

That's it for our links.

What we did—

By setting the text color to the same color as the page background, our hover class simulates the effect of punching through its light gray background.

Chapter 9—Making a Style Sheet for Version 4 Browsers

MAN: “DOCTOR, IT HURTS WHEN I DO *THIS*.”

DOCTOR: “THEN DON’T DO THAT.” —HENNY YOUNGMAN

Henny Youngman’s¹ doctor would have had the perfect prescription for the **cross-browser blues**: if it doesn’t work in *browser x*, then just don’t do it! If a particular CSS declaration causes version 4 browsers to burp, we can take the good doctor’s advice—or we can get creative. Here’s the situation:

We’ve just spent considerable time writing a very nice style sheet that works wonderfully in version 5 and higher browsers. But how does it work in Navigator 4? Take a moment now to preview our page in *that* browser (don’t be frightened—help is on the way).

TAKE TWO STYLE SHEETS AND CALL ME IN THE MORNING

If Navigator 4 (and MSIE4) did not exist, our work would be done. If we knew that every visitor to our page would be using a modern browser, we could stop right now and call it a wrap. Alas, this is probably not the case. What do we do? We know that version 4 browsers² have trouble with some of our styles—but they also have trouble reading a style sheet at all if it’s *imported* instead of *linked*. We can capitalize on this inability to *effectively* hide a style sheet from version 4 browsers. Here’s the plan:

We’ll make a copy of `mfv5.css` and name it **`mfv4.css`**. Then we’ll remove (or modify) all of the declarations that Navigator 4 dislikes. We’ll change our linked style sheet to `mfv4.css` and we’ll reattach `mfv5.css` by importing it. A version 4 browser will read the linked style sheet (which contains only style properties it supports)—but will not see the imported one (which contains advanced properties that would cause it problems). Version 5 browsers will see and read both style sheets.

Before we begin the process of branching into two style sheets, let’s look at a visual representation of the Navigator 4 issues that we need to address as an aid in planning our version 4 style sheet strategy.

¹ Henny Youngman (1906-1998) was known as the *King of the one-liners*. My favorite Youngman joke: A man goes to a psychiatrist. The doctor says, “You’re crazy.” The man says, “I want a second opinion!” The doctor says, “OK, you’re ugly too!”

² MSIE4 has measurably less trouble with our styles, so it’s safe to assume that fixing Navigator 4 will be our focus.

PLANNING THE VERSION 4 STYLE SHEET

We need to identify the issues that are causing distress in Navigator 4. If we can get a handle on them, and cure them, we'll be home free. Let's have a look.



FIGURE 232- THE VIEW FROM NAVIGATOR 4

Our solution is to make a separate style sheet that contains fixes for all of the noted Navigator 4 issues. We'll link this style sheet to our page and reattach our original style sheet by *importing it*. Navigator will read the linked (and fixed) style sheet, but will ignore the imported one. Version 5 browsers will read both the linked *and* the imported sheets.

Version 4 browsers support *linked* style sheets, but don't support the *@import* method and/or syntax¹—whereas version 5 browsers support both the *link* *and* *@import* methods.

¹ Netscape 4 browsers do not understand the *import* method at all. MSIE4, on the other hand, understands *import*—but will not find the style sheet if the URL is both wrapped in quotes **and** inside parentheses.

It works like this—

When a browser encounters two (or more) style sheets, it reads them according to where they come in a document's source code—from the top down. As it reads the style sheets, a browser may find conflicts. If it encounters two identically named selectors with conflicting properties—the one encountered *last* will take precedence. Call it a *tie breaker*. Consider this scenario:

A document contains both a linked and an imported style sheet, in the following order:

```
<link href="v4.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
@import url("v5.css");
-->
</style>
```

Inside v4.css is the following rule:

```
.foobar {color: red; font-weight: bold;}
```

Inside v5.css is a rule with the same selector name:

```
.foobar {color: blue;}
```

In the document, we have the following paragraph:

```
<p class="foobar">This is my paragraph.</p>
```

Here's what happens—

A version 4 browser sees only the rule from the linked style sheet. As far as it's concerned, there is no conflict and it renders the paragraph text **red** and **bold**:

This is my paragraph. [red and bold]

A version 5 browser reads both the linked and the imported style sheet—and renders the text **blue** and **bold**.

This is my paragraph. [blue and bold]

The version 5 browser, having read both style sheets, encounters a conflict: two identically named rules (foobar) are declaring two different colors for the same element. Since an element can only be one color at a time, a tie breaker is needed. *Blue* wins because it comes from the imported style sheet, which comes after the linked style sheet (in the document's source code). The text remains bold because there are no conflicting instructions.

To a version 5 browser, the rules from both style sheets are combined into one like this:

```
.foobar {
    color: red;           [from the first style sheet]
    font-weight: bold;    [from the first style sheet]
    color: blue;          [from the second style sheet]
}
```

In a way, browsers need to behave like good soldiers in a parade—with the combined style sheet being the sergeant barking out orders. Sergeant says: “Color red! Font-weight bold! Color blue!” The good soldier better be bold and blue after all is said and done.

MAKING A COPY OF MFV5.CSS

To set up our dual-style sheet solution¹ we first need to create a style sheet for version 4 browsers. We’ll do that by making a copy of `mfv5.css`—then we’ll rename the copy.

-Open **mfv5.css**

-Choose **File**→ **Save As**

-Enter *File name*: **mfv4.css**

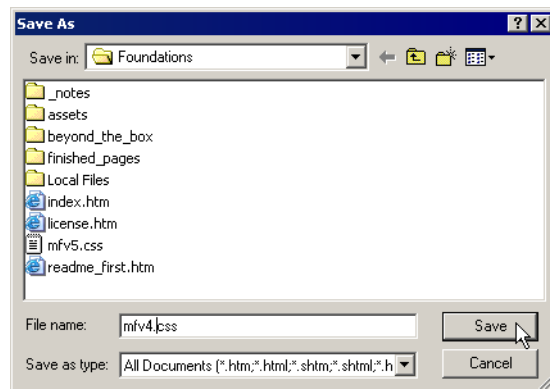


FIGURE 233

-Click **Save**

-Close `mfv4.css`

¹ Another popular way to serve dual style sheets is to use JavaScript to detect the browser version and determine the appropriate css file. In fact, PVII has a Dreamweaver Command called **StyleLoader** that does just that. However, learning to use the `@import` method lays a solid foundation because it requires you to learn about how styles cascade and are inherited.

RESET THE LINKED STYLE SHEET

-Switch to index.htm in Code View

-Place your cursor inside the style sheet link¹:

```
<link href="mfv5.css" rel="stylesheet" type="text/css">
```

Of course, you can simply change mfv5.css to mfv4.css, but I wanted to point out a nifty feature you may not have been aware of. If you look at your Properties inspector, you'll notice it has changed and now reflects the information in the <link> tag. You can click the yellow file folder icon to browse to mfv4.css or you can type it directly in the *Href* box.

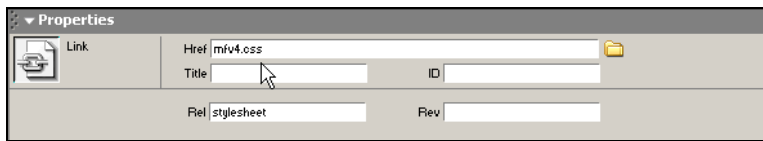


FIGURE 234

There is another feature I may as well point out, too. With your cursor inside the <link> tag, you can press **Ctrl + F5** to edit the tag. You can change the linked style sheet by typing directly in the *HREF* box, or you can click the Browse button to locate mfv4.css—clicking OK when you're done.

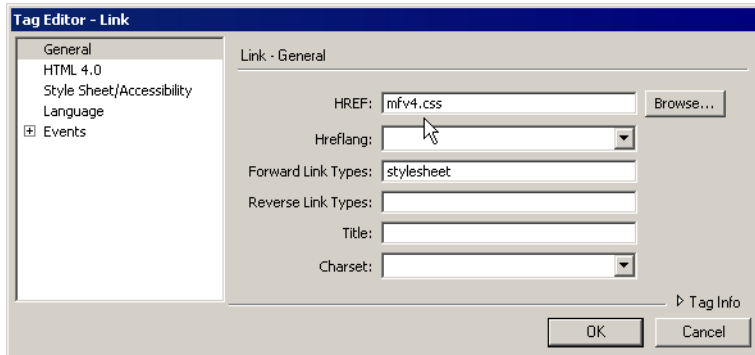


FIGURE 235

-Using one of the methods we just explored, you should now change your style sheet link so that it reads like this:

```
<link href="mfv4.css" rel="stylesheet" type="text/css">
```

There will be no apparent changes in Design View (or in any browser) because mfv4.css is an identical copy of mfv5.css—we'll be editing it in just a moment.

¹ The Style Sheet link is towards the top of your source code—just above the closing </head> tag.

IMPORTING THE VERSION 5 STYLE SHEET

Now that we've changed our linked style sheet to the new version 4 flavor, we need to reattach the version 5 style sheet via the `@import` rule. You can remain in Code View.

-Click the *Attach Style Sheet* icon along the bottom of the CSS Styles panel

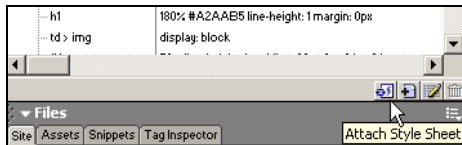


FIGURE 236

The Link External Style Sheet¹ window will open.

-Enter *File/URL*: **mfv5.css** (or Browse to it)

-Select *Add As*: **Import**

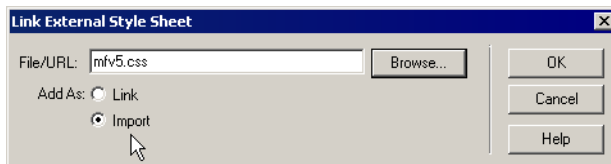


FIGURE 237

-Click **OK**

The `@import` rule should appear just below the linked style sheet:

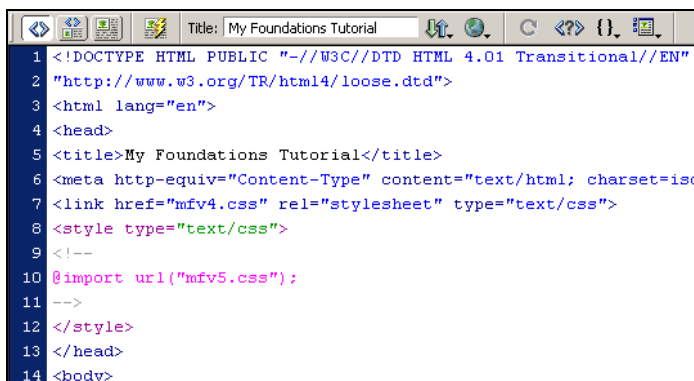


FIGURE 238

-Switch to Design View

-**Save** the page

¹ Since it's possible to either Link or Import a style sheet from this dialog, the window probably should have been titled: **Attach** External Style Sheet.

EDITING THE VERSION 4 STYLE SHEET

Working directly inside the **mfv4** style sheet—we'll start by setting our page's default font in a way that Navigator 4 will understand.

Note: Dreamweaver's Design View works very much like a version 5 (or higher) browser. What you see is the combined rendering of all style sheets attached to the page. To see the effects of edits to the version 4 style sheet, you'll need to preview the page in a version 4 browser.

SETTING FONT-FAMILY FOR VERSION 4 BROWSERS

Setting a font-family on the body selector—`font-family: Verdana, Arial, Helvetica, sans-serif;`—causes all text in our document to be styled because all tags inside the `<body>` tag will inherit the property unless explicitly instructed not to. However, Navigator 4 has problems with this. We can create a *combined element selector* to take care of Navigator in a single rule.

-Open **mfv4.css**

-Make a new line at the bottom of the file and **create** this rule:

```
td, p {
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

Navigator 4 will understand this rule just fine. All text whose parent element is a table cell `<td>` or paragraph `<p>`¹ will be rendered according to the declared font-family.

This rule is only necessary in the version 4 style sheet.

¹ Should you ever add a new element to your page and you notice that Navigator 4 is not rendering your font, you can append the combined selector to include other tags. For instance, if you add an unordered (bulleted) list, you would edit the selector to read: **td, p, ul**.

EDITING THE SIDEBAR LINK STYLES

Let's start by working on the **p.navbutts a** and **p.navsubs a** rules.

-Locate **p.navbutts a**, which currently looks like this:

```
p.navbutts a {  
    color: #000000;  
    text-decoration: none;  
    background-color: #A2AAB5;  
    background-image: url(assets/mfimages/navbg.jpg);  
    background-repeat: no-repeat;  
    display: block;  
    padding: 3px 0px 3px 12px;  
    width: 162px;  
    border-top: 1px solid #EEEEEE;  
    border-right: 1px solid #666666;  
    border-bottom: 1px solid #666666;  
    border-left: 1px solid #EEEEEE;  
}
```

Color #000000 (black) is fine. *Text-decoration* should be set to **underline** because Navigator 4 cannot possibly make our links look like buttons—so we better make sure users know they are links¹. Navigator 4 does not support *Background color* or *image* on the <a> tag. *Display block* is not healthy at all—Navigator 4 is thoroughly confused by this. *Padding* is not supported on <a> tags. In the absence of a background color, *Width* is no longer necessary. Navigator 4 does not support the *border* property on <a> tags.

-Edit **p.navbutts a** to remove all of the properties noted above so that it looks like this:

```
p.navbutts a {  
    color: #000000;  
    text-decoration: underline;  
}
```

¹ We're all creatures of habit (and convention). Underlined text and buttons almost *beg* to be clicked. If your links do not look like buttons, then underlining is probably a very smart thing to do.

Now let's edit **p.navsubs a**, which currently looks like this:

```
p.navsubs a {
    color: #000000;
    text-decoration: none;
    display: block;
    padding: 0px 0px 0px 14px;
    width: 162px;
}
```

Just as we did for *p.navbuts a*, we need to remove *display*, *padding*, and *width* properties. And we need to change decoration to **underline**. We'll also make color **white** (#FFFFFF) so the text stands out more.

-Edit **p.navsubs a** so that it looks like this:

```
p.navsubs a {
    color: #FFFFFF;
    text-decoration: underline;
}
```

Let's take a look at the **hover** and **active styles**. Here they are:

```
p.navbuts a:hover, p.navbuts a:active {
    color: #FFFFFF;
    background-color: #A2AAB5;
    background-image: none;
}
p.navsubs a:hover, p.navsubs a:active {
    color: #A2AAB5;
    background-color: #EEEEEE;
}
```

As luck would have it, Navigator 4 completely ignores *a:hover* and *a:active*—but MSIE4 supports them. So we can leave them as they are. No changes are required.

Let's examine the **p.navbutts** rule.

p.navbutts currently looks like this:

```
p.navbutts {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}
```

Early Navigator 4 releases do not do well when font-size is declared as a percentage¹. Let's make font-size 16 pixels so the main links will stand out. Bottom margin (4px) is fine, but we need to add some left margin to create room on the left—let's say *16px*. Using shorthand for the margin values, the second value (0px) applies to both right *and* left—and a fourth value would be the **left** value (remember [TRouBLe](#)).

-Edit **p.navbutts** so that it looks like this:

```
p.navbutts {  
    font-size: 16px;  
    margin: 0px 0px 4px 16px;  
}
```

¹ If one were very careful with inheritance issues, percentages could be used for Navigator 4. In fact, we do so on the PVII Web site. However, our site's audience is exclusively comprised of Web designers and developers, the vast majority of whom would only use Navigator 4 just to see how well our site works ☺. The problems with Navigator 4 and font-sizing involve early Windows releases, such as Navigator 4.0-4.04, and just about every Mac version.

Now let's edit **p.navsubs**, which currently looks like this:

```
p.navsubs {
    font-size: 110%;
    margin: 4px 0px 2px;
}
```

We'll make font-size 12px (4px smaller than p.navbuts) and set left margin to 28px (which will indent the sub links 12 pixels in from the main links).

-Edit **p.navsubs** so that it looks like this:

```
p.navsubs {
    font-size: 12px;
    margin: 4px 0px 2px 28px;
}
```

-Save and close mf4.css, then switch to index.htm.

-Preview in Navigator 4

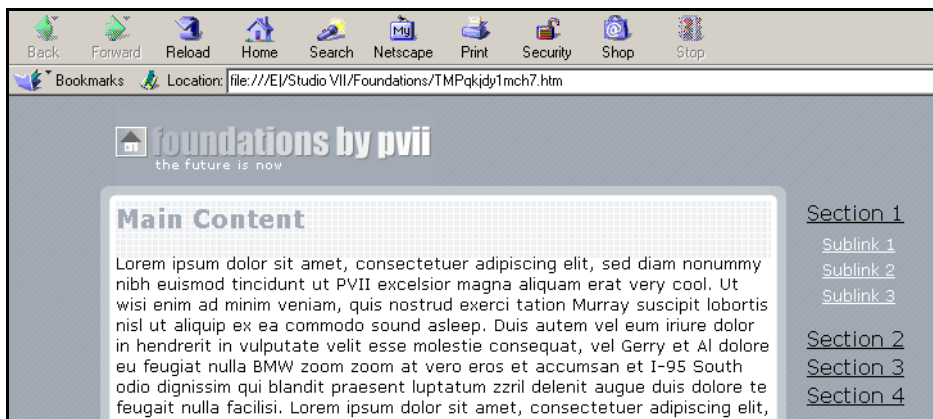


FIGURE 239- PREVIEW IN NAVIGATOR 4

We're shaping up nicely.

Notice that **#nbutspacer** (margin-bottom: 16px) is working perfectly to provide space between the *Sublink 3* and *Section 2* paragraphs.

Now let's move on to the **#sidebar** selector.

-Open **mfv4.css** and locate the **#sidebar** selector

#sidebar currently looks like this:

```
#sidebar {
    font-size: 75%;
    width: 176px;
    padding: 0px 0px 24px 16px;
}
```

The padding is not supported and can be removed. Font-size should be set to an absolute pixel measure.

-Edit **#sidebar** so that it looks like this:

```
#sidebar {
    font-size: 12px;
    width: 176px;
}
```

EDITING THE <h1> SELECTOR

The **h1** selector currently looks like this:

```
h1 {
    font-size: 180%;
    line-height: 1;
    color: #A2AAB5;
    margin: 0px;
}
```

We know that it needs to be lowered and that it needs some space on the left. We can take care of that by editing the *margin* property. *Font-size* should be changed to an absolute (pixel) measure. *Color* is fine. *Line-height* is very problematic in Navigator 4, so we'll just play safe and remove the property altogether.

-Edit the **h1** selector so that it looks like this:

```
h1 {
    font-size: 24px;
    color: #A2AAB5;
    margin: 24px 0px 0px 24px;
}
```

The margin values for both top and left are set to **24px**, which happens to equal the padding value in the *#mn* padding property.

-**Save and Close** mfv4.css

-Preview in Navigator 4



FIGURE 240

The heading is now properly positioned.

EDITING THE #MN AND #MN P SELECTORS

-Open mfv4.css

The **#mn** selector currently looks like this:

```
#mn {
    font-size: 80%;
    line-height: 1.5;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
    padding: 24px;
}
```

Font-size should be changed to a pixel measure—14px is a good size for body text in Navigator 4. *Line-height* must go. The *background image*, *repeat*, and *color* properties as declared, are nicely supported by Navigator 4 and can remain unchanged. *Padding* is not supported and should be removed.

-Edit the **#mn** selector so that it looks like this:

```
#mn {
    font-size: 14px;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
}
```

Now let's do the **#mn p** selector, which currently looks like this:

```
#mn p {
    margin: 12px 0px 0px;
}
```

The *margin* property will serve to create main content offsets in Navigator 4. We'll set the values (clockwise from top to left) to **6px 24px 6px 24px**. Since the top and bottom are equal *and* the left and right are equal, we need only list two values.

-Edit the **#mn p** selector so that it looks like this:

```
#mn p {
    margin: 6px 24px;
}
```

-**Save and Close** mfv4.css

-Switch to **index.htm** and Preview in Navigator 4

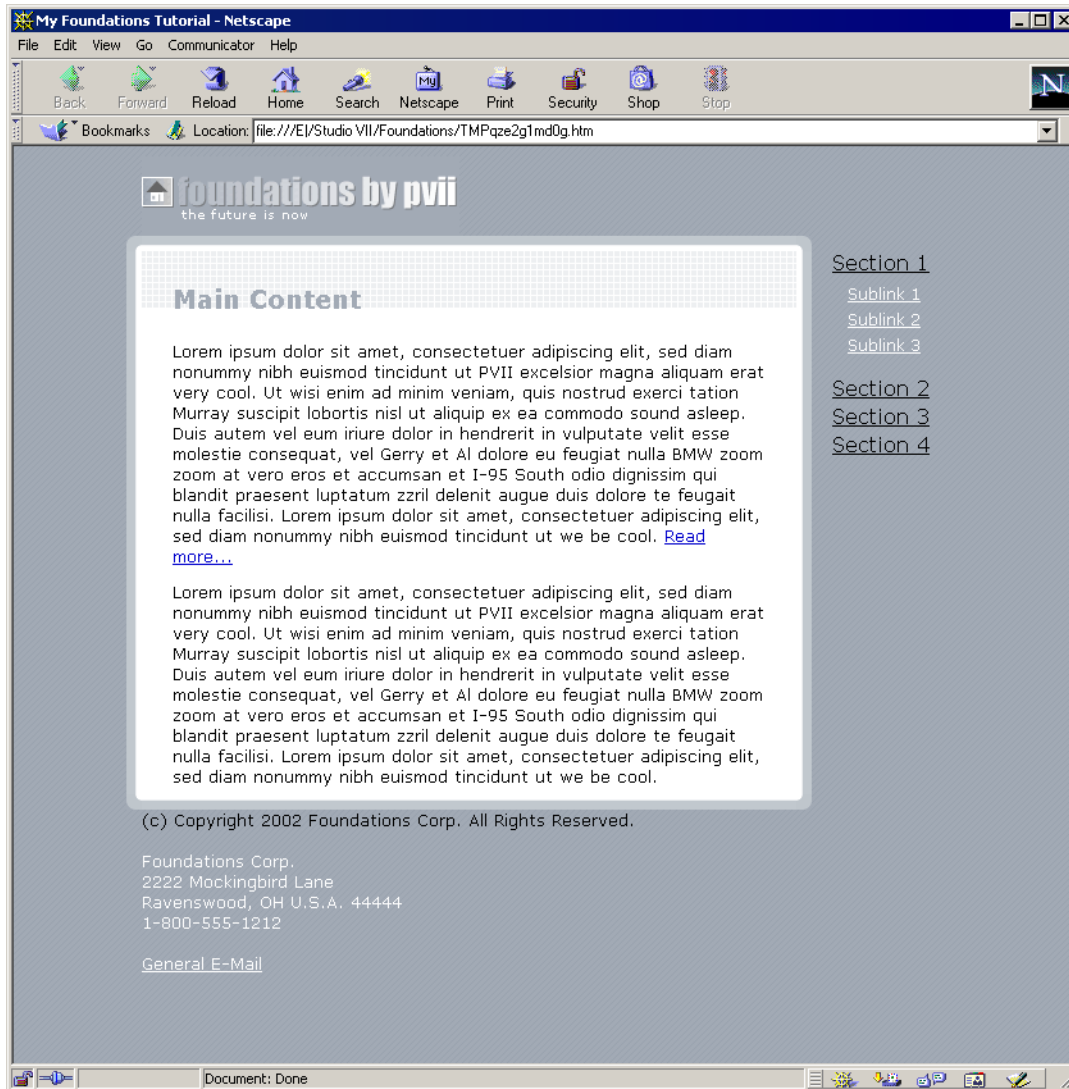


FIGURE 241

Bingo! Now all we need to do is fix the *#footer* and *address* selectors.

EDITING THE #FOOTER SELECTOR

The **#footer** selector currently looks like this:

```
#footer {
    font-size: 70%;
    line-height: 1;
    padding: 20px 0px 24px 24px;
}
```

Font-size measure should be changed to pixels. 12px is about as small as you want to go in Navigator 4. *Line-height* must go. *Padding* is not supported and will be removed.

-Edit the **#footer** selector so that it looks like this:

```
#footer {
    font-size: 12px;
}
```

Now let's create a new **#footer p** selector.

-Make a new line under the **#footer** selector and enter this rule:

```
#footer p {
    margin: 20px 0px 20px 24px;
}
```

Top and bottom margins are 20px—creating space between the bottom of the main content and the top of the copyright notice—and between the bottom of the copyright notice and the top of the address. Left margin is 24px—serving the same role as **#footer's** padding does. Right margin is set to zero.

Before we go any further, we need to create a **#footer p** selector in the **mfv5** style sheet to set a zero margin. We need to do this because version 5 browsers read both style sheets and will therefore see both the padding (from **#footer**) and the margin (from **#footer p**). The padding and margin values will be added together—resulting in too much offset.

-Open **mfv5.css**

-Create a new **#footer p** rule:

```
#footer p {
    margin: 0px;
}
```

-Save and close **mfv5.css** and switch back to **mfv4.css**

We'll now make our last edit—to the **address** selector.

EDITING THE ADDRESS SELECTOR

The **address** selector currently looks like this:

```
address {  
    font-style: normal;  
    color: #FFFFFF;  
}
```

Color #FFFFFF (white) is fine, as is *font-style*. We simply need to add a *margin* property so that the address section lines up on the left with the copyright, the main content, and the heading.

-Edit the **address** selector to add a margin property:

```
address {  
    font-style: normal;  
    color: #FFFFFF;  
    margin: 0px 24px;  
}
```

Now we'll need to edit the *address* selector in **mfv5.css**. If we don't edit the mfv5.css address selector margin, then the right and left margin values from the mfv4 rule will be added to the padding on the *#footer* selector—offsetting the address twice as much from the left and right as it needs to be. Since we zeroed out the *#footer p* margins in the mfv5 sheet earlier—we also need to set top and bottom margin values to create offset above and below the address block.

-**Save and Close** mfv4.css and open **mfv5.css**

-Find the address selector and add the margin property, so that it looks like this:

```
address {  
    font-style: normal;  
    color: #FFFFFF;  
    margin: 20px 0px;  
}
```

-**Save and Close** mfv5.css and switch to index.htm

PREVIEWING THE FINISHED STYLE SHEET IN NAVIGATOR 4

Let's preview index.htm in Navigator 4 and see if our edits work.

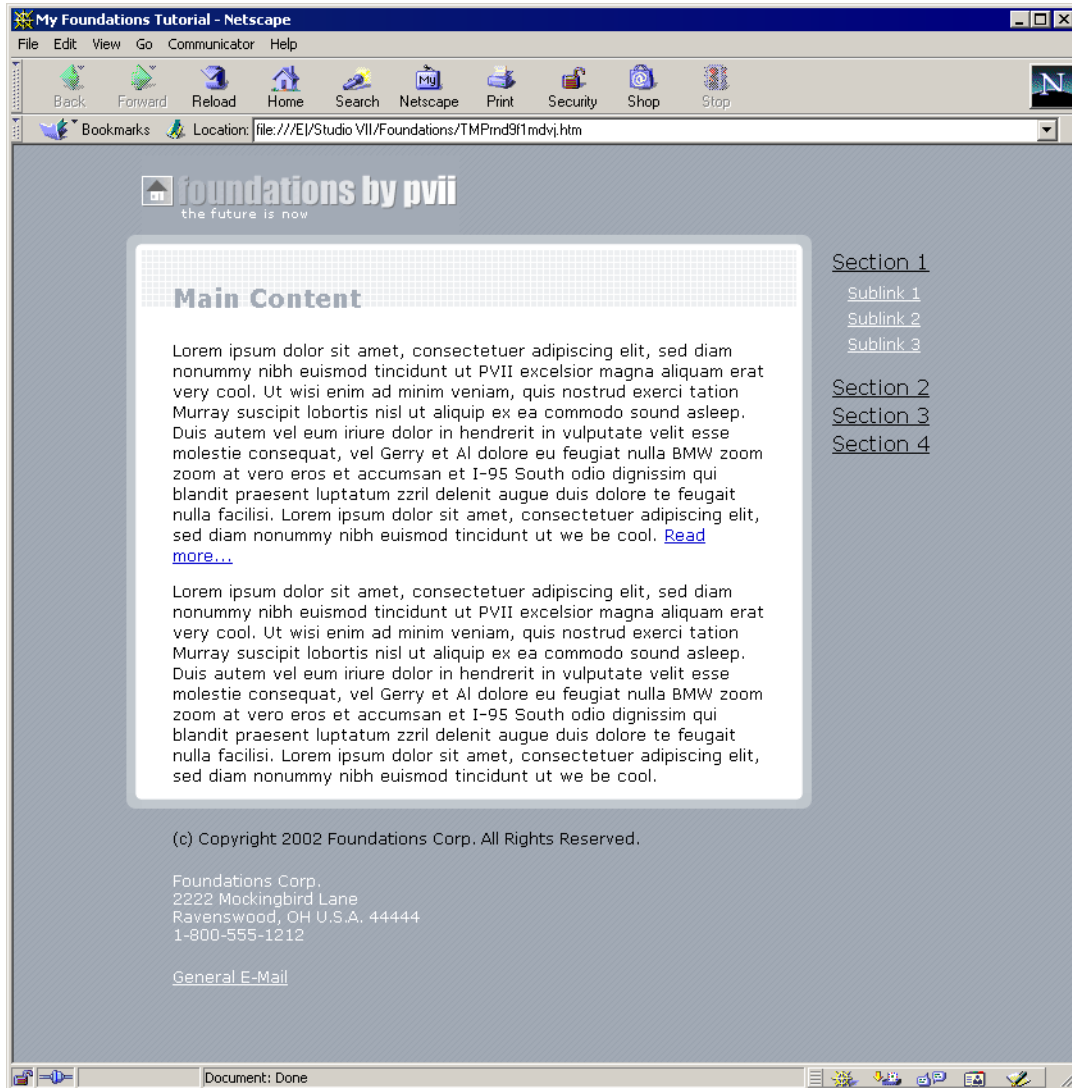


FIGURE 242

I think we've done a good job.

Notice that the Logo is now positioned close to the top of the window. By removing the padding properties from #mn, #sidebar, and #footer, Navigator snapped to its senses (in a vertical manner of speaking). The browser had been terribly confused and was actually adding up several padding values and applying them to the body element.

CODE CHECK

Your revised `mfv4.css` file should look like this:

```
#tlcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tlcorner.gif);
}
#tedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tedge.gif);
}
#trcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/trcorner.gif);
}
#redge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/redge.gif);
}
#brcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/brcorner.gif);
}
#bedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/bedge.gif);
}
#blcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/blcorner.gif);
}
#ledge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/ledge.gif);
}
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
#mn {
    font-size: 14px;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
}
h1 {
    font-size: 20px;
    color: #A2AAB5;
    margin: 24px 0px 0px 24px;
}
td > img {
    display: block;
}
#mn p {
    margin: 6px 24px;
}
#footer {
    font-size: 12px;
}
#footer p {
    margin: 20px 0px 20px 24px;
}

address {
    font-style: normal;
    color: #FFFFFF;
    margin: 0px 24px;
}
#sidebar {
    font-size: 12px;
    width: 176px;
}
#mn a:link {
    color: #0000CC;
}
#mn a:visited {
    color: #666666;
}
#mn a:hover, #mn a:active {
    color: #000000;
}
#footer a:link {
    color: #FFFFFF;
}
#footer a:visited {
    color: #EEEEEE;
}
#footer a:hover, #footer a:active {
    color: #A2AAB5;
    text-decoration: none;
    background-color: #FFFFFF;
}
p.navbut {
    font-size: 16px;
    margin: 0px 0px 4px 16px;
}
p.navsubs {
    font-size: 12px;
    margin: 4px 0px 2px 28px;
}
#nbutspacer {
    margin-bottom: 16px;
}
p.navbut a {
    color: #000000;
    text-decoration: underline;
}
p.navbut a:hover, p.navbut a:active {
    color: #FFFFFF;
    background-color: #A2AAB5;
    background-image: none;
}
p.navsubs a {
    color: #FFFFFF;
    text-decoration: underline;
}
p.navsubs a:hover, p.navsubs a:active {
    color: #A2AAB5;
    background-color: #EEEEEE;
}
td, p {
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

Your revised mfV5.css file should look like this:

```
#tlcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tlcorner.gif);
}
#tedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tedge.gif);
}
#trcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/trcorner.gif);
}
#redge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/redge.gif);
}
#brcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/brcorner.gif);
}
#bedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/bedge.gif);
}
#blcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/blcorner.gif);
}
#ledge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/ledge.gif);
}
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
#mn {
    font-size: 80%;
    line-height: 1.5;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
    padding: 24px;
}
h1 {
    font-size: 180%;
    line-height: 1;
    color: #A2AAB5;
    margin: 0px;
}
td > img {
    display: block;
}
#mn p {
    margin: 12px 0px 0px;
}
#footer {
    font-size: 70%;
    line-height: 1;
    padding: 20px 0px 24px 24px;
}

address {
    font-style: normal;
    color: #FFFFFF;
    margin: 20px 0px;
}
#sidebar {
    font-size: 75%;
    width: 176px;
    padding: 0px 0px 24px 16px;
}
#mn a:link {
    color: #0000CC;
}
#mn a:visited {
    color: #666666;
}
#mn a:hover, #mn a:active {
    color: #000000;
}
#footer a:link {
    color: #FFFFFF;
}
#footer a:visited {
    color: #EEEEEE;
}
#footer a:hover, #footer a:active {
    color: #A2AAB5;
    text-decoration: none;
    background-color: #FFFFFF;
}
p.navbuts {
    font-size: 130%;
    margin: 0px 0px 4px;
}
p.navsubs {
    font-size: 110%;
    margin: 4px 0px 2px;
}
#nbutspacer {
    margin-bottom: 16px;
}
p.navbuts a {
    color: #000000;
    text-decoration: none;
    background-color: #A2AAB5;
    background-image: url(assets/mfimages/navbg.jpg);
    background-repeat: no-repeat;
    display: block;
    padding: 3px 0px 3px 12px;
    width: 162px;
    border-top: 1px solid #EEEEEE;
    border-right: 1px solid #666666;
    border-bottom: 1px solid #666666;
    border-left: 1px solid #EEEEEE;
}
p.navbuts a:hover, p.navbuts a:active {
    color: #FFFFFF;
    background-color: #A2AAB5;
    background-image: none;
}
p.navsubs a {
    color: #000000;
    text-decoration: none;
    display: block;
    padding: 0px 0px 0px 14px;
    width: 162px;
}
p.navsubs a:hover, p.navsubs a:active {
    color: #A2AAB5;
    background-color: #EEEEEE;
}
#footer p {
    margin: 0px;
}
```

THE DREAMWEAVER MX @IMPORT ISSUE

At the time of this writing, the original shipping version of Dreamweaver MX contains a bug¹ that can compromise our version 4 style sheet strategy if we're not careful. While it is likely that Macromedia will issue a maintenance release at some future point, we need to address the issue *now*.

This bug involves only the original shipping version of Dreamweaver MX (6.0). Subsequent updates by Macromedia have fixed the bug. You should check the [Dreamweaver Support Center](#) section of Macromedia's Web site to make sure you have the latest updates.

Dreamweaver MX writes the @import rule like this:

```
<style type="text/css">
<!--
@import url("mfv5.css");
-->
</style>
```

Note the quotation marks that surround the file name inside the parentheses.

If you **move** a page (using the Site panel) or use **File→Save As** to create a copy² two folders deep in the site, Dreamweaver MX will rewrite the @import rule and remove the quotations:

```
<style type="text/css">
<!--
@import url(..../mfv5.css);
-->
</style>
```

This little change will cause the imported style sheet to become visible to MSIE4 (Navigator 4 will still not be able to find it).

The solution is to manually edit the @import rule whenever you move a file, rename a file, or make a copy using File→Save As. You can also use *Find and Replace* on a folder-by-folder basis. For example:

Find: **@import url(..../mfv5.css)** | Replace with: **@import url("..../mfv5.css")**

This bug affects only MSIE4. Navigator 4 is not affected.

¹ Bugs are inevitable—especially in a program as large and complex as Dreamweaver MX.

² Moving the style sheet itself (using Dreamweaver's site window) will also trigger the bug.

ABOUT DUPLICATE RULES AND REDUNDANT PROPERTIES

If you think about how our style sheets work, it's apparent that we have an opportunity to save considerable bandwidth. Consider the following rule:

```
#tlcorner {  
    background-color: #FFFFFF;  
    background-image: url(assets/mfimages/tlcorner.gif);  
}
```

This rule is present in both style sheets: *mf4* and *mf5*. Since version 5 browsers can read both style sheets, it only needs to be in one—the one that both version 5 and version 4 browsers can read, which is **mf4**. Knowing this, we can sweep through our style sheets—removing redundant properties and duplicate rules (with some exceptions).

When we're done, we'll have two fully optimized style sheets that do their jobs as efficiently as possible.

THE NAVIGATOR 4 URL ISSUE

Navigator 4 has an image path problem we need to address before we go any further. Here's the problem in a nutshell—

URLs in style sheets are supposed to be relative to the CSS document. However, Navigator 4 resolves URLs in relation to the HTML document that the CSS file is linked to. For instance, the property **background-image: url(assets/mfimages/tlcorner.gif)** resides in a CSS file called *mf4v.css*, which is linked to a page called *index.htm*.

A browser is supposed to follow the path from *mf4v.css* file to the image. Navigator 4, however, attempts to locate the image by starting from *index.htm*—which means it will only be successful if *mf4v.css* and *index.htm* are in the same folder. Right now, our style sheets and our index page are in the same folder (the site root) and that's why everything seems to be working fine.

Let's perform a little test...

-Open **index.htm**

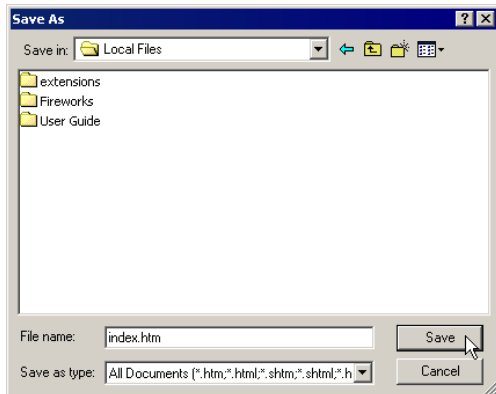


FIGURE 243

-Choose **File**→**Save As**

-Browse to the Foundations/Local Files folder

-Leave File name as: **index.htm**

Click **Save**

You just created a copy of index.htm in the Local Files folder.

-Preview the new page in a version 5 browser:

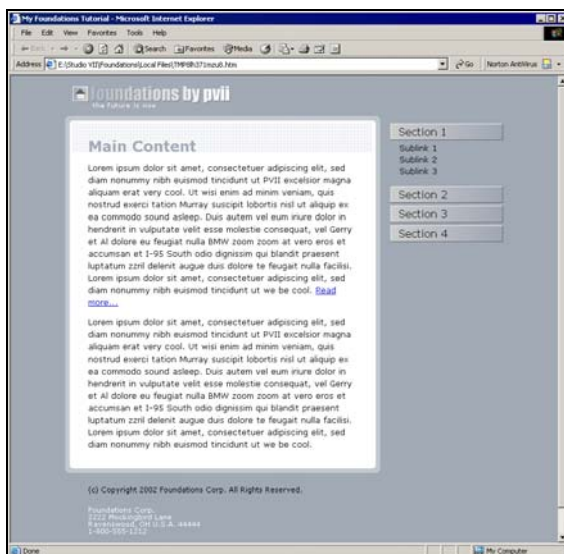


FIGURE 244

Looks fine. All images are present and accounted for.

-Now Preview in Navigator 4:

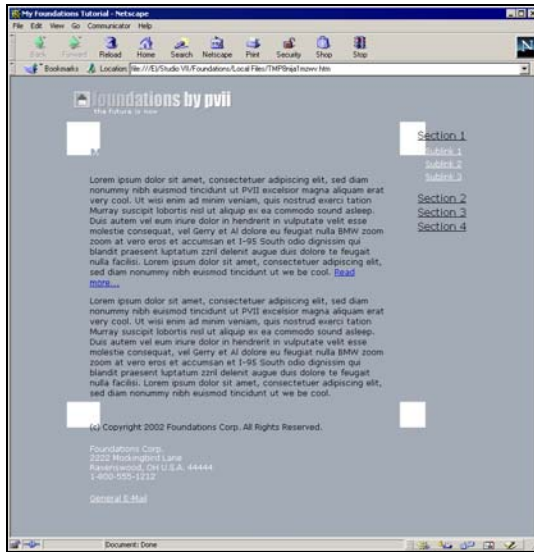


FIGURE 245

Not so good. We are missing all of the background images declared in our style sheet.

Let's explore some solutions—

SOLVING THE NAVIGATOR 4 CSS URL ISSUE

To get around this issue, you can change the image paths in the CSS file from *document-relative* to either *site root-relative* or *absolute*¹.

Don't panic—this is not a major problem. We have some *relatively* painless solutions:

Solution 1

You can make your CSS image URLs **site-root relative** if you have a local development server running *and* you can place your style sheets in the server's root web folder. Preview using the local server and Navigator 4 will be on best behavior.

If you don't know what I'm talking about, then you probably are not running a local development server and probably don't need to. The next solution may be right for you.

Solution 2

Make your CSS image paths **absolute** if you have a broadband² *always-on* connection to the internet (cable or DSL, e.g.). You can preview pages normally from within Dreamweaver (F12) and the images will be retrieved and rendered—providing you've uploaded them to your remote server/host.

One more problem—

Solving the Navigator URL problem presents us with another problem. If you choose to use absolute URLs, Dreamweaver will not be able to display the images in Design View. If you have a broadband connection, you will be able to see the images when you preview—but only if the CSS files on your server are current.

To remedy this problem, we can use Find and Replace in the mf4.css file to toggle the URL from absolute to relative—and back. When you are editing files locally, you set the URL to relative. Whenever you make changes to your style sheet, you toggle back to absolute before uploading the file to your server. This presents us with an opportunity to explore and use a very cool Dreamweaver feature called Saved Queries.

¹ A *site root-relative* path uses a leading forward slash to represent the root folder. The path `/assets/mfimages/foobar.jpg` is valid from any page in a site—no matter where that page is—so long as the page is being viewed using a running Web server. An *absolute* path is a full hyperlink starting at the root of your domain (or IP address). The path `http://www.youraddress.com/assets/images/foobar.jpg` is also valid from any location.

² This technique will work even if you have a dialup connection, as long as you are connected when you preview.

SAVED QUERIES

Here's how it works:

CONVERTING FROM RELATIVE TO ABSOLUTE URLs

-Open **mfv4.css**

-Choose **Edit→Find and Replace** (Ctrl + F)

-When the Find and Replace window opens, use the following settings:

Find In: **Current Document**

*Search for*¹: **assets/mfimages/**

Replace With: **http://www.mysite.com/assets/mfimages/**

Note: You need to replace www.mysite.com with your actual domain name or IP address.

Options: **Ignore Whitespace Differences** should be the only checked box

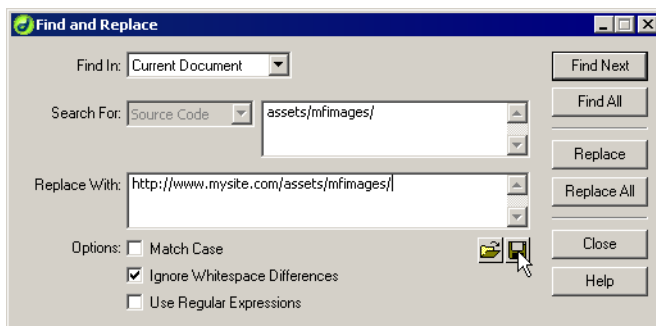


FIGURE 246- CLICKING THE SAVE QUERY ICON

-Click the **Save Query** icon 

The Save Query window will open.

¹ Source Code is the only choice for CSS files—so the *Search For* type list will be grayed out.

-Make sure *Save In* is showing your Foundations root folder

-In the File name box, type: **cssPath_absolute**

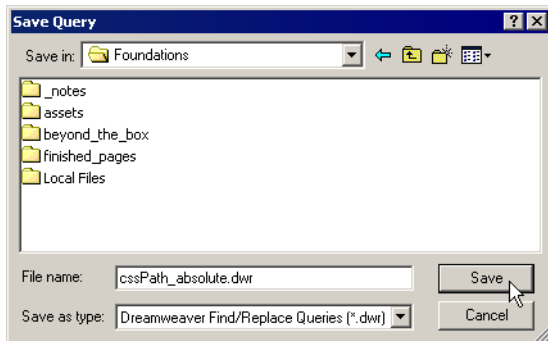


FIGURE 247

-Click **Save**

The Save query window will close.

-Click the **Replace All** button in the Find and Replace window

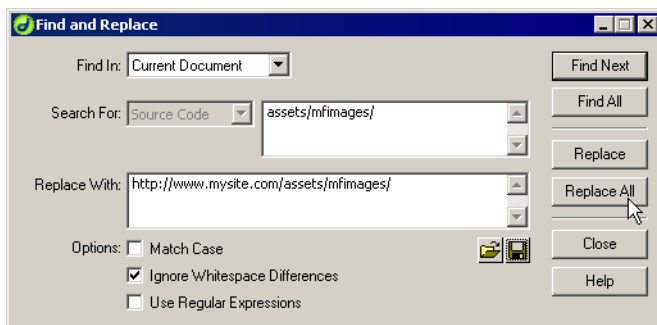


FIGURE 248

A response window will open informing you how many matches were found. There should be ten matches.

-Click **OK** to close the Response window

-Click **Close** to close the Find and Replace window

All the URLs will change to *absolute*.

-Save **mfv4.css**—but keep the file open

CONVERTING FROM ABSOLUTE BACK TO RELATIVE

-Choose **Edit→Find and Replace** (Ctrl + F)

-When the Find and Replace window opens, use the following settings:

Find In: **Current Document**

Search for: **http://www.mysite.com/assets/mfimages/**

Replace With: **assets/mfimages/**

Options: **Ignore Whitespace Differences** should be the only checked box

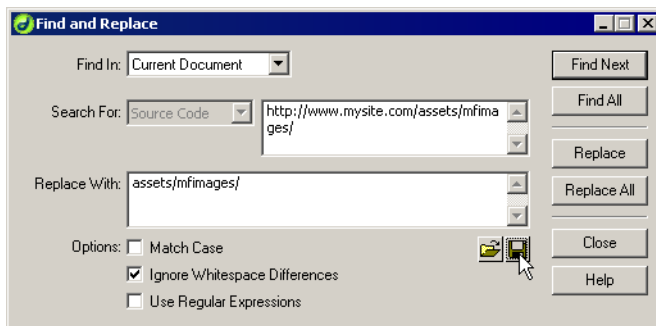


FIGURE 249

-Click the **Save Query** icon 

The Save Query window will open.

-In the File name box, type: **cssPath_relative**

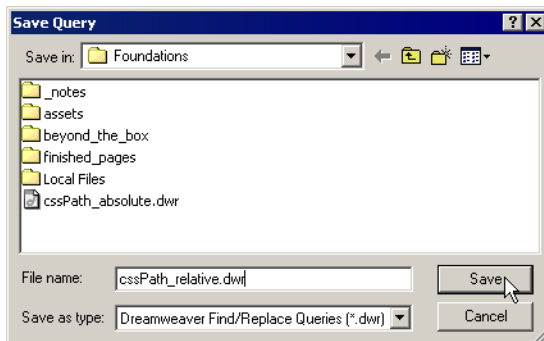


FIGURE 250

-Click **Save** to save the Query and close the *Save Query* window

-Click the **Replace All** button in the Find and Replace window to change all of your URLs back to relative

A response window will open informing you how many matches were found.

-Click **OK** to close the *Response* window

-Click **Close** to close the *Find and Replace* window

-**Save** and close mfv4.css

USING THE SAVED QUERY

To use (or *recall*) one of the saved queries:

-Open **mfv4.css**

-Choose **Edit→Find and Replace** (Ctrl + F) to open the Find and Replace window

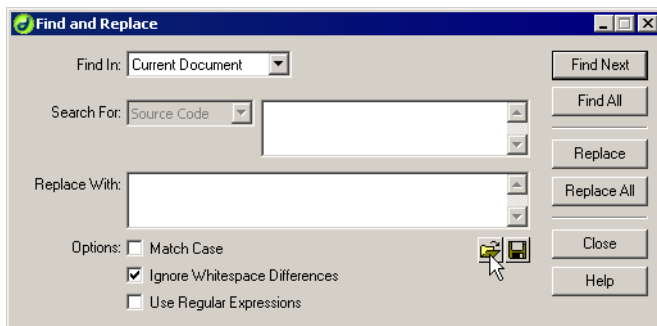


FIGURE 251

-Click the **Load Query** icon 

The Load Query window will open

-Select the **cssPath_absolute.dwr** query

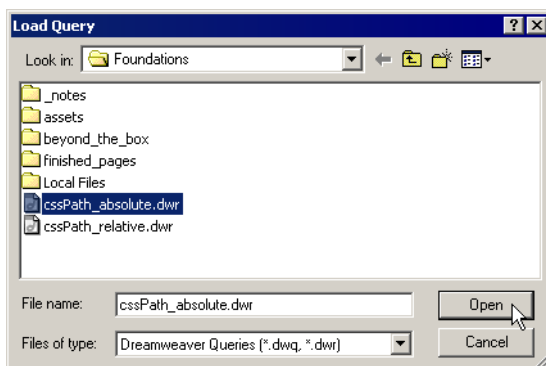


FIGURE 252

-Click **Open**

The Load Query window will close and the Find and Replace window will now show your query parameters.

-Make sure *Find In: Current Document* is selected

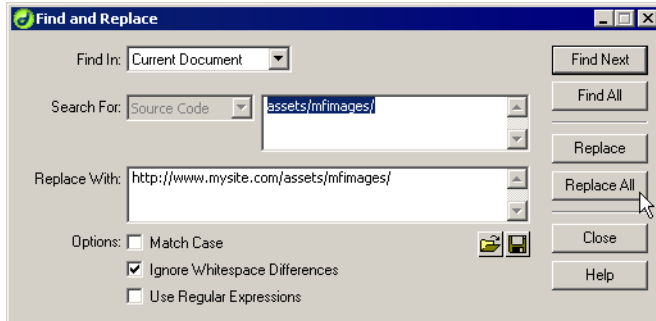


FIGURE 253

-Click **Replace All** to run the query

A response window will open

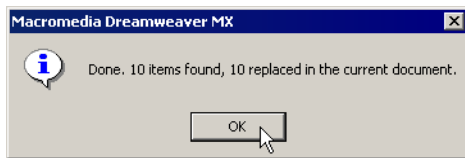


FIGURE 254

-Click **OK** to close the response window

-Click **Close** to close the Find and Replace window

SET THE PATHS BACK TO RELATIVE BEFORE CONTINUING TO EDIT

Since we have some more editing to do, run the `cssPath_relative.dwr` query again to set all URLs in `mfv4.css` back to relative—so we can see our background images in Dreamweaver's Design View.

-Run the `cssPath_relative.dwr` query

-**Save and Close** `mfv4.css`

Remember: You don't need to set the URLs to absolute until you are ready to upload your style sheet to the internet.

OPTIMIZING OUR TWO STYLE SHEETS

This section will walk you through the process of optimizing our two style sheets by eliminating duplicates and redundancies.

A HYPOTHETICAL EXAMPLE

If the rule `h1{color: red;}` appears in both style sheets, it's redundant. That's because version 4 browsers read it from the linked style sheet and version 5 browsers read it from *both* style sheets. Therefore, it only needs to be in the linked style sheet. Or consider this:

The rule `h1{color: red;}` appears in the linked sheet and `h1{color: red; padding: 6px;}` appears in the imported style sheet. We can leave the first rule alone, but we can change the second one to remove the duplicate color property. The rule in the imported style sheet can thus be shortened to: `h1{padding: 6px;}`.

We'll go through both CSS files—rule-by-rule.

- Open both **mfv4.css** and **mfv5.css**

- Make a backup copy of each (just in case)

After using our *Saved Find and Replace Query* to change the mfv4.css URLs to absolute, the following eight rules in mfv4.css render their mfv5.css counterparts redundant.

- Delete the **mfv5** versions of all eight of these rules:

#tlcorner | #tedge | #trcorner | #redge | #brcorner | #bedge | #blcorner | #ledge

THE OPTIMIZATION PROCESS

Let's start with the body rule.

THE BODY RULE

The current body rule from mfv4:

```
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
```

The current body rule from mfv5:

```
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
```

The mfv5 version is redundant.

-Delete the mfv5 rule

THE #MN RULE

The current mfv4 version:

```
#mn {
    font-size: 14px;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
}
```

The current mfv5 version:

```
#mn {
    font-size: 80%;
    line-height: 1.5;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
    padding: 24px;
}
```

-Edit the mfv5 rule to include just its unique properties:

```
#mn {  
    font-size: 80%;  
    line-height: 1.5;  
    padding: 24px;  
}
```

Moving on—

THE H1 RULE

The **h1** rule should remain unchanged in mfv4.

-Delete its **color** *property* from the mfv5 style sheet (color: #A2AAB5;)

TD > IMG

td > img is needed only in mfv5.

-Delete the entire rule from mfv4

#MN P

Requires no edits and should be left as they are—in both style sheets.

#FOOTER AND #FOOTER P

No changes. Leave as they are in both style sheets.

ADDRESS

Leave as it is in mfv4.

-Remove the **color** *property* from the mfv5 style sheet

#SIDEBAR

Leave as it is in mfv4

-Remove the **width** *property* from mfv5.

**#MN A:LINK | #MN A:VISITED | #MN A:HOVER, #MN A:ACTIVE | #FOOTER A:LINK
#FOOTER A:VISITED | #FOOTER A:HOVER, #FOOTER A:ACTIVE**

Leave all rules as they are in mfv4.

-Delete each of them from mfv5

P.NAVBUTS, P.NAVSUBS

Leave unchanged in both style sheets

#NBUTSPACER

Leave the mfv4 rule as it is.

-Delete the redundant mfv5 rule

P.NAVBUTS A

Leave unchanged in mfv4

-Remove the **color** *property* from mfv5

P.NAVBUTS A:HOVER, P.NAVBUTS A:ACTIVE

Leave unchanged in mfv4.

-Delete entire rule from mfv5

P.NAVSUBS A

Leave unchanged in both style sheets.

P.NAVSUBS A:HOVER, P.NAVSUBS A:ACTIVE

Leave unchanged in mfv4.

-Delete entire rule from mfv5.

TD, P

This rule is specific to mfv4 and requires no editing. It should be left unchanged.

-**Save** both CSS files now.

CSS CODE CHECK—MFV4

```
#tlcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tlcorner.gif);
}
#tedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tedge.gif);
}
#trcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/trcorner.gif);
}
#redge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/redge.gif);
}
#brcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/brcorner.gif);
}
#bedge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/bedge.gif);
}
#blcorner {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/blcorner.gif);
}
#ledge {
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/ledge.gif);
}
body {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    background-image: url(assets/mfimages/bgrid.gif);
    background-color: #A2AAB5;
    margin: 0px;
    padding: 0px;
}
#mn {
    font-size: 14px;
    background-color: #FFFFFF;
    background-image: url(assets/mfimages/tbg.gif);
    background-repeat: repeat-x;
}
h1 {
    font-size: 20px;
    color: #A2AAB5;
    margin: 24px 0px 0px 24px;
}
#mn p {
    margin: 6px 24px;
}
#footer {
    font-size: 12px;
}
#footer p {
    margin: 20px 0px 20px 24px;
}
address {
    font-style: normal;
    color: #FFFFFF;
    margin: 0px 24px;
}
```

```
#sidebar {
    width: 176px;
}
#mn a:link {
    color: #0000CC;
}
#mn a:visited {
    color: #666666;
}
#mn a:hover, #mn a:active {
    color: #000000;
}
#footer a:link {
    color: #FFFFFF;
}
#footer a:visited {
    color: #EEEEEE;
}
#footer a:hover, #footer a:active {
    color: #A2AAB5;
    text-decoration: none;
    background-color: #FFFFFF;
}
p.navbut {
    font-size: 16px;
    margin: 0px 0px 4px 16px;
}
p.navsubs {
    font-size: 12px;
    margin: 4px 0px 2px 28px;
}
#nbutspace {
    margin-bottom: 16px;
}
p.navbut a {
    color: #000000;
    text-decoration: underline;
}
p.navbut a:hover, p.navbut a:active {
    color: #FFFFFF;
    background-color: #A2AAB5;
    background-image: none;
}
p.navsubs a {
    color: #FFFFFF;
    text-decoration: underline;
}
p.navsubs a:hover, p.navsubs a:active {
    color: #A2AAB5;
    background-color: #EEEEEE;
}
td, p {
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

CSS CODE CHECK—MFV5

```
#mn {
    font-size: 80%;
    line-height: 1.5;
    padding: 24px;
}
h1 {
    font-size: 180%;
    line-height: 1;
    margin: 0px;
}
td > img {
    display: block;
}
#mn p {
    margin: 12px 0px 0px;
}
#footer {
    font-size: 70%;
    line-height: 1;
    padding: 20px 0px 24px 24px;
}
address {
    font-style: normal;
    margin: 20px 0px;
}
#sidebar {
    font-size: 75%;
    padding: 0px 0px 24px 16px;
}
p.navbut {
    font-size: 130%;
    margin: 0px 0px 4px;
}
p.navsubs {
    font-size: 110%;
    margin: 0px 0px 2px;
}

p.navbut a {
    text-decoration: none;
    background-color: #A2AAB5;
    background-image: url(assets/mfimages/navbg.jpg);
    background-repeat: no-repeat;
    display: block;
    padding: 3px 0px 3px 12px;
    width: 162px;
    border-top: 1px solid #EEEEEE;
    border-right: 1px solid #666666;
    border-bottom: 1px solid #666666;
    border-left: 1px solid #EEEEEE;
}
p.navsubs a {
    color: #000000;
    text-decoration: none;
    display: block;
    padding: 0px 0px 0px 14px;
    width: 162px;
}
#footer p {
    margin: 0px;
}
```

We’ve trimmed quite a bit. Preview in a version 4 and a version 5 browser to make sure things still work. If there are problems, use the Code Checks above as a guide in locating your problem.

We have also changed the very nature of how our two style sheets relate to each other. Instead of being a special style sheet just for version 4 browsers, **mfv4.css** is now our base style sheet!

Before you edit a working style sheet always make a backup copy. Work smart and be organized.

-Close both Style Sheets after Checking the Code

Chapter 10—Extra Ingredients

“MOUNTAIN MEASUREMENTS

1 SMIDGEN = A TEENSY LITTLE BIT

3 SMIDGENS = A PINCH

4 PINCHES = A LITTLE BIT

4 LITTLE BITS = A MIDDLELIN' AMOUNT

3 MIDDLELIN' AMOUNTS = 1 RIGHT SMART

AND IT TAKES 5 RIGHT SMARTS TO MAKE A WHOLE HEAP.”—JED CLAMPETT

We've built a very nice page. More important, we built a structurally sound and very efficient page—but we didn't build a Web site. If you are very new to Dreamweaver (or to Web Design in general), you may have a few questions floating around your head.

The techniques described in this Chapter are optional and will not effect processes or techniques discussed in subsequent Chapters. *Feel free to try all of them, some of them, or none at all.*

HOW DO I TURN TEMPORARY LINKS INTO REAL LINKS—AND MAKE MORE PAGES?

To make a new page, you can open the index.htm page and choose **File→Save As** to make an exact copy anywhere in your site. Once you've made more pages, you can edit the temporary links by inserting your cursor within the linked text and then using the Properties inspector **Link** box to link to a real page.

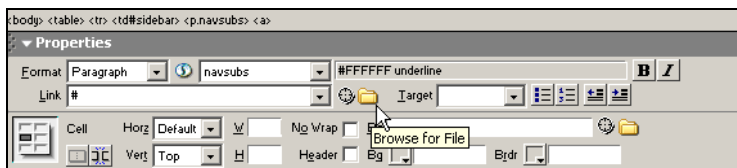


FIGURE 255

CAN I ADD IMAGES INSIDE THE MAIN CONTENT TABLE CELL?

Sure you can. If you want the image on its own line, simply create a new paragraph and insert the image. If you want the image centered on its line, create a new rule in the v4 style sheet:

```
p.centered {text-align: center;}
```

-Apply the **.centered** class to the paragraph that contains your image

If you want your text to wrap around the image, you have two choices:

1. USING CSS¹:

-Create two new rules in the v4 style sheet:

```
img.wrapleft {float: left;}
img.wrapright {float: right;}
```

Insert your image at the beginning of a paragraph and apply either the **wrapleft** or **wrapright** classes to the `<img>` tag.

2. USING HTML MARKUP:

-Select your image

-On the Properties inspector, set *Align*: **Left**

¹ Controlling the image placement with CSS affords more control on a global basis. However, chances are you would not need that level of control, so using the HTML property might be a better choice—and will work more reliably in older browsers.

CAN I ADD A TABLE INSIDE THE MAIN CONTENT CELL SO I CAN LIST MY COMPANY’S WIDGETS?

Adding a table is simple. Making it look good requires a few CSS rules to augment transitional HTML markup. The goal is to make it look as good as possible in a cross-browser way. Let’s create the styles first—

-Open **mfv4.css** and create these new rules:

```
table.nested td {
    background-image: none;
}
table.nested th {
    background-image: none;
    background-color: #EEEEEE;
    font-weight: bold;
    text-align: left;
}
```

If we don’t set background-image to **none**, Navigator 4 will use the **#mn** background-image. It is an inheritance bug. We need to turn background-image off in all cells that are children of **#mn**.

```
table.nested caption {
    text-align: left;
    font-weight: bold;
}
```

-**Save** and **close** *mfv4.css*, then **open** **mfv5.css**

-Create these rules in *mfv5.css*:

```
table.nested {
    border: 1px solid #CCCCCC;
    width: 100%;
}
table.nested caption {
    padding-top: 12px;
}
```

-**Save** and **Close** *mfv5.css*

Now let's insert the table—

-Insert your cursor at the end of the first paragraph inside the main content cell and choose **Insert→Table**

-Fill out the property boxes just as we have in the figure below

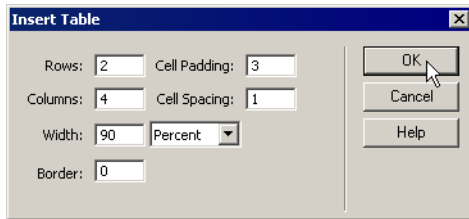


FIGURE 257

-Click **OK**

You may be wondering why we have set the width to 100% in the mv5 **table.nested** rule. It's because the main cell is styled with 24px of padding on all sides, so 100% ensures that it fills the entire available space with the same padding as any other content. Navigator 4 does not support the CSS padding on the main cell, so a 100% table would be flush up against the main cell's left edge and would not align with the rest of the cell's non-table content. Version 5 browsers will use the CSS rule and Navigator 4 will use the HTML property we just set in the Insert Table dialog.

The Accessibility dialog will open

-Fill out the property boxes just like we have in the figure below

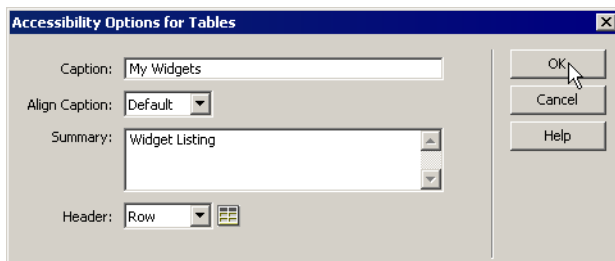


FIGURE 258

-Click **OK**

The table will appear.

-Use the Properties inspector to set *Align: Center*

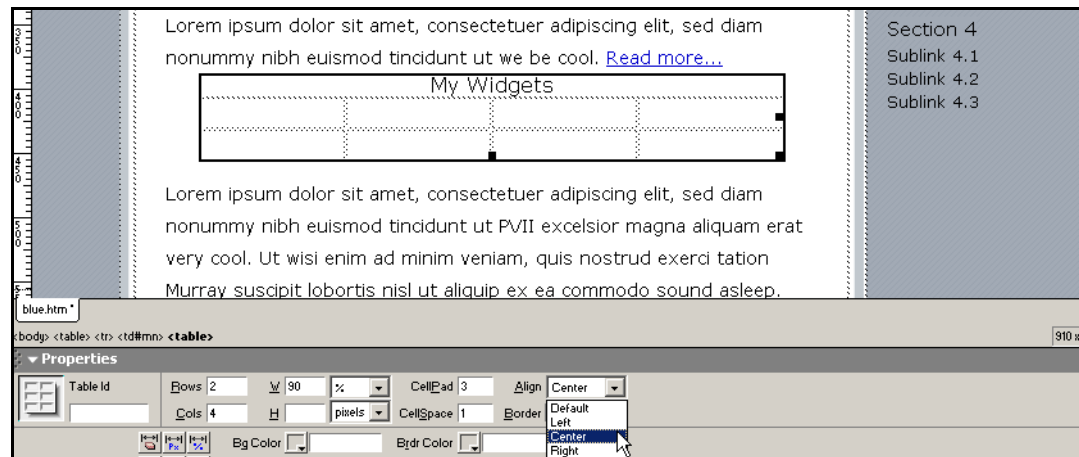


FIGURE 259

Enter some text into the cells:

-In the Header row, type the word *Heading* in each cell

-In the Data row, type the word *Item* in each cell

-Insert your cursor in the new table, then right-click the `<table>` tag marker on the *Tag Selector bar*.

-Choose **Set Class**→**nested**

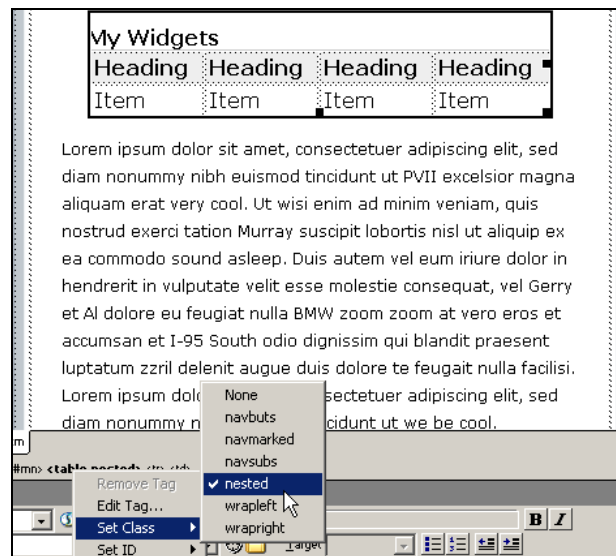


FIGURE 260

The styled table in Dreamweaver Design View:

My Widgets			
Heading	Heading	Heading	Heading
Item	Item	Item	Item

FIGURE 261

You may want to align the headers left. Even though we have set **text-align: left** in the **table.nested th** rule, Navigator 4 will only align the text if we use the HTML align property on the tag itself—and there's a bug in **Opera** that results in some rather bizarre behavior if we don't use the CSS property. To set the HTML align property:

Drag-select all four cells in the header row (or select the `<tr>` tag):

My Widgets			
Heading	Heading	Heading	Heading
Item	Item	Item	Item

FIGURE 262

-Set *Horz: Left*

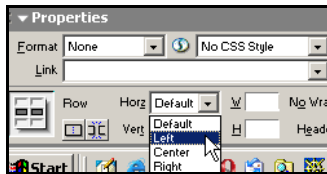


FIGURE 263

Here is the `<table>` markup:

```
<table width="90%" border="0" align="center" cellpadding="3" cellspacing="1" class="nested" summary="Widget Listing">
<caption>My Widgets</caption>
<tr align="left">
<th scope="col">Heading</th>
<th scope="col">Heading</th>
<th scope="col">Heading</th>
<th scope="col">Heading</th>
</tr>
<tr>
<td>Item</td>
<td>Item</td>
<td>Item</td>
<td>Item</td>
</tr>
</table>
```

-Preview the page in a version 5 browser:



FIGURE 264

-Preview the page in Navigator 4:



FIGURE 265

Navigator 4 loses track of text size in the paragraphs below the table. We need a more specific selector to work around this bug. Currently, text is set to 14px on the #mn selector. To fix this problem, let's declare font-size on the #mn p selector.

-Open mf4.css and locate the #mn p selector, which currently looks like this:

```
#mn p {
    margin: 6px 24px;
}
```

Edit the rule to add font-size:

```
#mn p {
    font-size: 14px;
    margin: 6px 24px;
}
```

-Save and Close mf4.css

Chapter 11—Adding Interactivity to the Menu

“A LITTLE SPLASH OF COLOR CAN REFRESH THE MIND’S EYE”. —ANONYMOUS

DRESSING UP THE MAIN BUTTONS

Let’s add some color to our link *buttons*. We’ll be working in *Fireworks* for a little while.

EDITING THE NAVBG IMAGE

-Open **myFoundations.png** in Fireworks

Make sure View Slcies is turned off.

-Before you proceed to the next step, click the shim image to pick up its fill and stroke settings

Note: When you draw an object in Fireworks, the tool you use will have the same stroke, fill, and pattern as the last selected object.

-Use the Rectangle tool to draw an 10-pixel wide by 96-pixel high rectangle



FIGURE 266

-Set the rectangle’s fill to a light yellow (I used #FAEBB6)

-Position the rectangle within the **navbg** rectangle—precisely at the top left edge (use the info panel or the Property inspector to make sure you are at the proper **X** and **Y** coordinates)

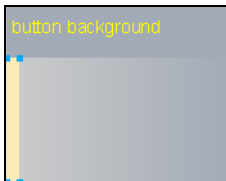


FIGURE 267

MAKING A SECOND NAVBG IMAGE FOR THE HOVER CLASS

We're going to copy the entire **navbg** image (both rectangles), then clone it—so we can make a second background image that can be applied to the **navbutts a:hover**, **navbutts a:active** selector.

-Drag a cursor box around both rectangles, selecting them both

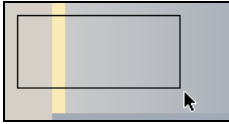


FIGURE 268



FIGURE 269

-Choose **Edit**→**Clone**

-Leave the clone group selected and use the Property inspector to set its coordinates to **X:0 and Y:495** (just about at the bottom left corner of your canvas)

-Click on the canvas to deselect the group, then select *just the inner yellow rectangle*

-Change the fill color to **pale green** (we used **#CBE4CB**)

-Select the larger rectangle and use the Property inspector to set its *Texture* to **Hatch 1**.

-Set the *Amount of Texture* to **13%**



FIGURE 270

SLICING IT

Let's do this the easy way. We'll copy the slice that's over the original **navbg** image. It's already got the optimization settings we need.

-Click the *Show Slices and Hotspots* button on your toolbar 

-Select the slice named **navbg**

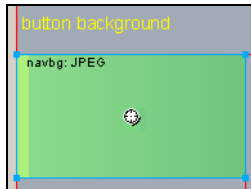


FIGURE 271

-Press **Ctrl+C** to copy it—then **Ctrl+V** to paste a copy

-Drag the copy down and drop it on top of your new image (**X:0** and **Y:495**)

-Use the Property inspector to name this new slice: **navbg_on**

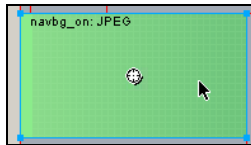


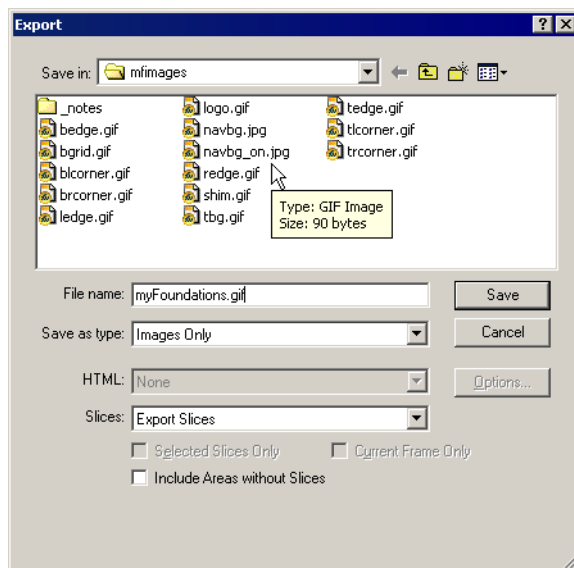
FIGURE 272

-Click on the canvas to deselect the slice and click the *Hide Slices and Hotspots*  button to turn slices off

EXPORTING TO FOUNDATIONS

-Choose **File**→**Export** (you should know the process by now—and Fireworks should also¹) and export to the **assets/mfimages** folder of your Foundations site

If you repeat the Export process, the Export window will show the new *navbg_on.jpg* image.



-**Save** the file and close Fireworks

¹ Fireworks saves the settings each time you Export. If the last time you used the Export function was the last time you exported your **myFoundations** images, then everything will be set as it was. If not, please refer to Chapter 7.

AND NOW FOR A LITTLE CSS WIZARDRY

We're going to do a bit of work inside our style sheets. We'll create a special version of the *p.navbutts a:hover*, *p.navbutts a:active* rule for our **mfv5** style sheet.

- Open **mfv4.css**
- Scroll down to the **p.navbutts a:hover**, **p.navbutts a:active** selector
- Select the entire rule and press **Ctrl+C** to copy it

Make sure you select **p.navbutts a:hover**, **p.navbutts a:active** and not **p.navsubts a:hover**, **p.navsubts a:active**

- Close **mfv4.css**
- Open **mfv5.css**
- Make a new line underneath the **p.navbutts a** rule and press **Ctrl+V** to paste the *p.navbutts a:hover*, *p.navbutts a:active* rule
- Save and Close **mfv5.css**

EDITING THE P.NAVBUTTS A:HOVER, P.NAVBUTTS A:ACTIVE RULE

- Open **index.htm**
- In the **mfv5.css** rules list in your CSS panel, find and double-click the **p.navbutts a:hover**, **p.navbutts a:active** rule

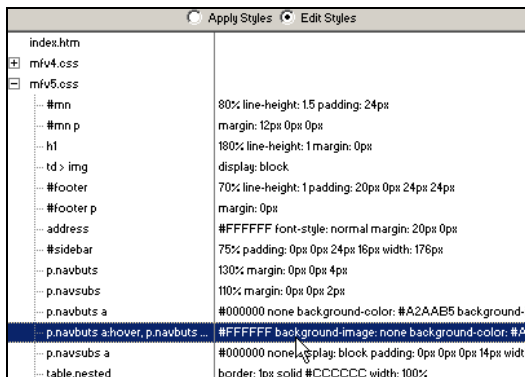


FIGURE 273

The *CSS Style Definition* window will open—

-In the *Type Category*, enter *Color: #858F9E* (we picked the page background color then darkened it several shades)

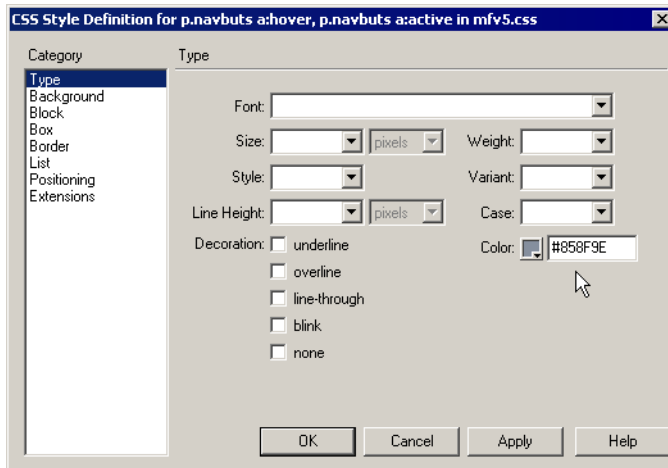


FIGURE 274

-Switch to the **Background Category**

Enter the following Values:

-*Background Color: #CCCCCC*

-*Background Image: assets/mfimages/navbg_on.jpg*

-*Repeat: no-repeat*

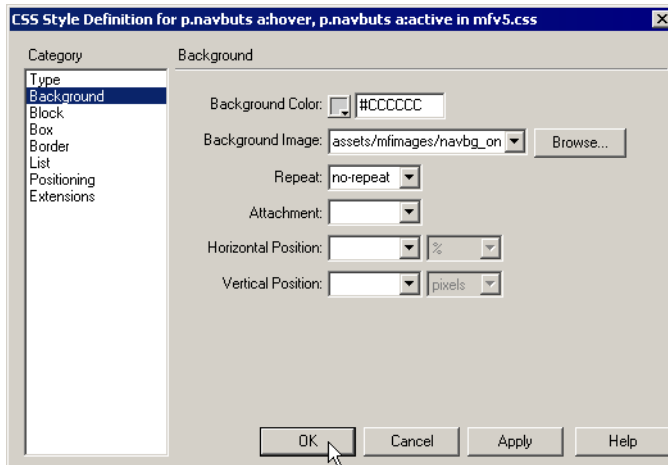


FIGURE 275

-Click **OK**

PREVIEW IN A VERSION 5 (OR HIGHER) BROWSER

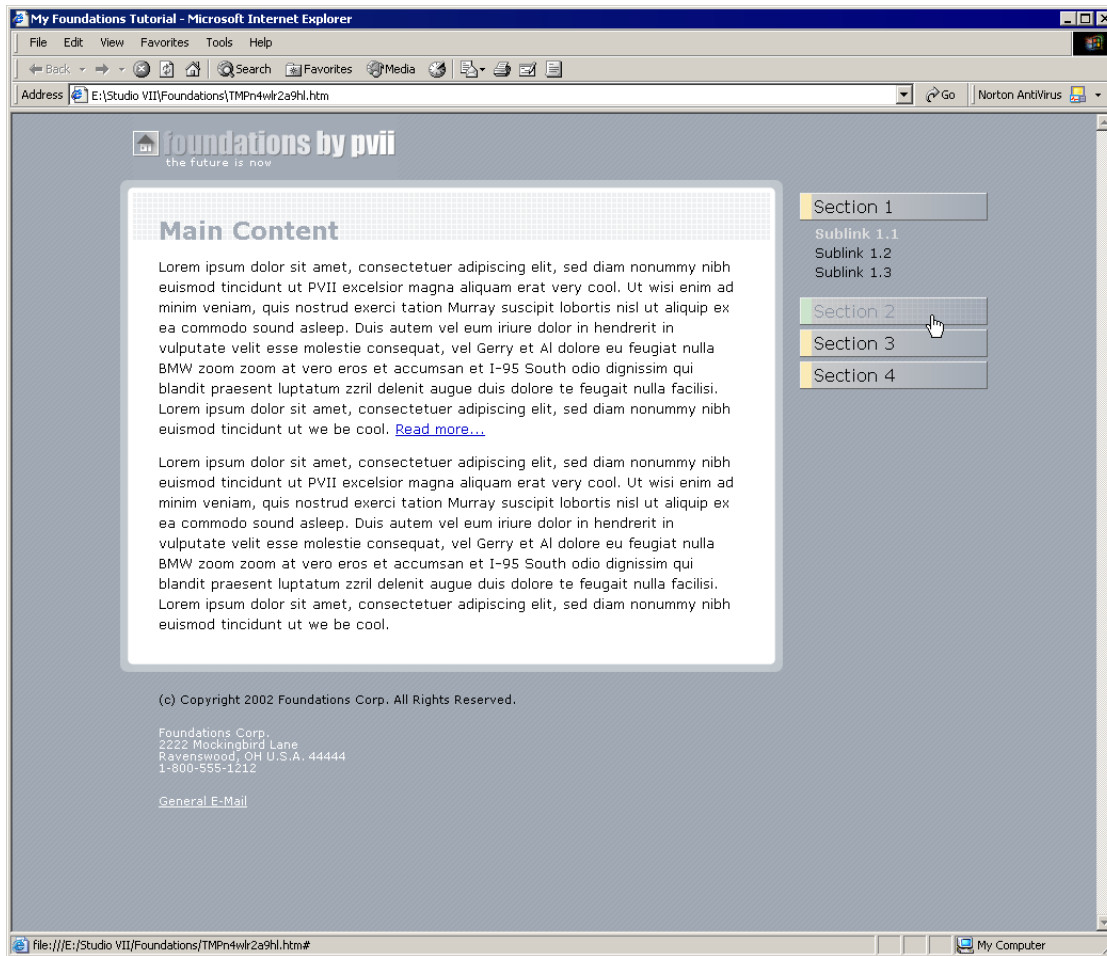


FIGURE 276

I really like that! In case you were wondering, the view from Navigator 7 is just as good.

Opera was pushed a bit beyond its capabilities. Its programmers left out a few key pieces—one of which is the ability to change background images via the hover class.

So now what do you want to do? I know—let's step outside the box.

Chapter 12—Stepping Smartly Outside the Box

“YOU’LL DECLARE IT’S SIMPLY TOPPING
TO BE THERE AND HEAR THEM SWAPPING
SMART TIDBITS
PUTTIN’ ON THE RITZ” —IRVING BERLIN

Our menu looks very *hierarchical*. I don’t know about you, but I feel compelled to make it do something *more*. I want more interactivity. We could use the Dreamweaver MX Pop-up Menu Builder¹ to make our sublinks fly out of our main buttons. Let’s not—I’ve got a much better idea. We’ll make a collapsible menu. When you click one of the main buttons, its sublinks will appear below and everything else will move down. Yes. That’s what we’ll do. And that begs the question: *how?*

Way back in Chapter 1, you installed the PVII *SwapClass* extension. This Dreamweaver MX add-on allows us to use JavaScript to manipulate CSS class assignments on various elements within a page. In a way, it’s similar to what one could accomplish with images using Macromedia’s Swap Image behavior—but in a much more powerful way. In a moment, you’ll see why JavaScript and CSS are more related than you might think.

HOW THE MENU WILL WORK

When our page opens in a browser, the menu section that relates to the current page should be completely open. That is, the user should see the section heading and each of the sublinks below it. This menu section should remain in its open state and never change. It’s purpose is to make sure that users will never lose track of where they are. Let’s refer to this menu as the **locked menu**.

The other menu section headings should serve as toggle switches. When a user clicks a heading, its sub-menu should open—and all other open sub-menus (except for the *locked* one) should be closed. A second click reverses the process and closes the sub-menu. Each subsequent click reverses what happened on the prior click.

¹ The Dreamweaver/Fireworks MX Pop-up Menu builder takes a much different approach than we dare. In order to make menus look *exactly* the same in version 4 and version 5 browsers, it builds the entire menu system inside of the JavaScript. This affords a great deal of control, but at a price. If a browser is lacking JavaScript support, or if the user has opted to disable script, or if a disabled person using assistive technology happens upon the page—the menu will not be rendered and there will be no way to navigate the site unless the author had the foresight to include a text menu for backup.

BROWSER LIMITATIONS AND ACCESSIBILITY

When a non-supporting browser is encountered, our goal is to have the menu appear in a fully opened state. That is, all menus are visible and accessible (and if a text-only or JavaScript-disabled browser is encountered, we'd like the menu to be accessible, as well). We'd also like the menu to be operable using the tab key—in browsers that support tabbing.

SPECIFIC BROWSERS

The menu's dynamic aspects (the *toggle*) will be fully functional in MSIE5 and higher (PC and Mac), and Netscape Navigator 6 and higher (PC and Mac)—which represents the vast majority of browsers in use today.

Navigator 4 and MSIE4 (as well as older browsers) will render the menu in its fully opened state at all times.

Opera has all the necessary parts to render our version 5 style sheet with the best of them. However, it's missing a few vital pieces when it comes to supporting current DOM¹ standards and so it will have to render our menu in the fully opened state. This presents a small hurdle, which we'll discuss in just a moment.

THE MENU MECHANICS

The SwapClass menu works by manipulating the CSS classes assigned to specific elements, page-wide elements, or both. So our style sheets are critical components. We make the styles, then we tell SwapClass how and when to manipulate them. Because of this approach, SwapClass is a very lean script. It takes up all of 650 bytes (less than 1K). With respect to our menu, SwapClass will work by taking three basic instructions² from us:

1. When a section header is clicked, find all open menus that you know about and close them.
2. Open the menu under the header that was just clicked.
3. Remember the menu you just opened and if the header is clicked again, close the menu.

SwapClass accomplishes all of this by changing (*swapping*) CSS rules and remembering the changes so it can reverse them when called upon to do so. SwapClass and CSS are a formidable team.

¹ DOM stands for Document Object Model. Browsers use the DOM to traverse the properties of all page elements (objects) in a hierarchical fashion. JavaScript works by accessing an object's properties through the browser's DOM. Modern browsers adhere to a standardized DOM. Older browsers generally use a proprietary DOM. While Opera 6 is a modern browser, its support for DOM standards is lacking in its ability to allow the manipulation of certain object properties.

² Remember these instructions. We'll be discussing them again when we apply the SwapClass behavior.

THE STYLES SWAPCLASS WILL USE

With respect to **Foundations**, SwapClass is focussed on the job of opening and closing menus. You may be inclined to think in terms of *visibility*. Dreamweaver's Show-Hide Layer behavior works by using an object's visibility property. While that's logically sound, CSS has a property that is perfectly suited for our task: the *display* property. Here's the difference between the **visibility** and **display** properties:

In terms of being seen or not seen, the *visibility* property has two values we can use:

Visible = Now you see it. **Hidden** = Now you don't.

When an object is *hidden*, it doesn't give up its space. Imagine an invisible force field surrounding an invisible object. Nothing can get in. This is not an issue with Layers¹ because they float above the page. Our menus don't float, they are grounded. If we were to make a menu disappear using its *visibility* property, there would be an empty space between where the menu *was* and where the next menu *is*. The resulting gap would look quite silly.

The *display* property has two values we can try:

Block = Now you see it. **None** = Now you don't.

When an object has a display property of *none*, it not only disappears—the space it occupied becomes available. If we could use CSS *geographically* and set the United States of America's display property to *none*, not only would the U.S.A. disappear—but Mexico would slide up and meet Canada. Eh? That's right.

So SwapClass will use styles that employ the *display* property because when a sub-menu is closed, we want the menus below it to slide up and fill the vacancy.

And now that you have a clear understanding of the CSS required, let's make the styles.

¹ *Layer* is an ancient term coined by Netscape in 1997 to describe its proprietary <layer> tag—which is not supported by current Web standards. The correct terminology for what many Dreamweaver users consider a *Layer* is **Absolutely Positioned Div**.

CREATING THE CSS STYLES FOR OUR SWAPCLASS MENU

We need one style that does nothing but set the *display* property to **none** and another that reverses the process—setting display to **block**. Armed with these styles, we can set all of our menus to *none* (except for the *locked* one), then SwapClass can allow users to selectively open any menu they would like to see.

-Open **mfv5.css**

-Create a new line below the last rule and **write** these two new rules:

```
.showit {
    display: block;
}
.hideit {
    display: none;
}
```

-**Save** mfv5.css (but leave it open as we'll be adding a few more rules)

We have a conundrum¹. Opera has done wonderfully so far. It reads our imported style sheet and does the static version of our menu just fine. But as mentioned earlier, this otherwise fabulous browser is DOM-deficient when it comes to manipulating an object's style properties. So, because Opera can read **mfv5.css**, it will open the page and present users with closed menus that cannot be opened. We need an intervention. We have two choices.

We can load our page with all the menus open, then use SwapClass (applied to the <body> onLoad) to close the menus once the page is fully loaded. Since Opera cannot swap classes, the menus will all remain open. Cool! However, browsers that *can* swap classes will have to wait until the page is fully loaded before it can execute the SwapClass script that closes the menus. You know how Web pages load? The content comes in gradually—the speed dependent on various factors. Imagine the menus coming in, coming in, coming in—there they are! Where'd they go? Very confusing, indeed.

We can employ a hack. We can write one or more styles in such a way that only certain browsers can understand them. We researched and considered various solutions, then found just what we needed at John Albin Wilkin's place:

<http://www.albin.net/CSS/OwenHack.html>

John named the hack after his son Owen because it uses a first-child selector ☺.

¹ A *conundrum* is a funny sounding word for which you may substitute any of the following words or phrases: *sticky wicket*, *a pickle*, *a challenge*, *a daunting task*, *a British sports car*, *I'm out of beer*, *how do I tell the boss his fly is undone?*

THE HACKS

Here's Hack Number 1:

```
/* Opera Workaround1 */
html>body .hideit {
    display: block;
}
```

The above selector is called a *child selector*. It's a CSS2 selector and Opera supports it. MSIE (PC) does not support child selectors. Yippee! We can use a child selector to set all the menus to show and we won't affect MSIE. Almost. MSIE5 (Mac) and Mozilla browsers (PC and MAC) also support child selectors. So, we'll take care of Opera—but we'll ruin things for MSIE Mac, Mozilla, Netscape 6, and Netscape 7 users. We can still use our child selector if only we can find another type of selector that neither Opera nor MSIE (PC) understands—but *that Mozilla does*.

Here's Hack Number 2:

```
/* Opera Workaround 2 */
head:first-child+body .hideit {
    display: none;
}
```

The above selector is a *first-child* and *adjacent-sibling* selector. It's a CSS2 selector. MSIE5 (Mac) and all Mozilla-based browsers understand it. Opera does not². We've solved the conundrum, gotten out of our pickle, unstuck the wicket, risen above the challenge, and we've found a buyer for the Spitfire—because when we combine these two hacks, it fixes all of our problems.



Oh—I almost forgot!

-Write the above two hacks (rules) in your **mfv5** style sheet file (comments and all)

-Save and Close mfv5.css

Now breathe out.

While we could spend a few dozen pages discussing CSS hacks and workarounds, it's entirely beyond the scope of this book. Suffice it to say that we've accomplished our goal of being able to load the page with the menu in its optimum state for all browsers. If you'd like to explore a complete list of selectors and the browsers that support them, here's where to go: <http://www.webreview.com/style/css2/charts/selectors.shtml>

¹ CSS can include code comments. A CSS comment always begins with a forward slash and asterisk, and ends with an asterisk and a forward slash. */*This is a comment*/*

² MSIE (PC) does not support first-child selectors either. But that's academic because we've already addressed that browser with the first hack.

ADDING MORE MENU SUBLINKS

If we're going to make a dynamic, interactive menu, we'll need to create sublinks for all of our section headers.

- Open **index.htm** if it isn't already open
- In Design View, edit the first block of sublinks to append a more descriptive number to the end of each (instead of Sublink 1, let's make it Sublink 1.1, etc.). Make the links look like this:

Sublink 1.1

Sublink 1.2

Sublink 1.3

Now we'll add sublinks under the other main section links

- Place your cursor on the end of the line that contains the link: **Section 2**
- Press **Enter** once to create a new paragraph
- Use the *Tag Selector bar* to reassign the class of the new paragraph to **navsubs**
- Type the following three lines, pressing Enter at the end of the first two:

Sublink 2.1

Sublink 2.2

Sublink 2.3

- Use the Tag Selector bar to assign the paragraph containing Sublink 2.3 the ID: **nbut-spacer**
- Assign each new sublink a *temporary hyperlink* using the **octothorpe (#)** character
- Repeat** to add a set of three sublinks below each of the remaining main section links

Remember to assign the **nbutspacer ID** to the last sublink paragraph in each group.

- Switch to **Code View**

The #sidebar <td> code should now look like this:

```
<td valign="top" id="sidebar"><p class="navbut"><a href="#" tabindex="1">Section
1</a></p>
<p class="navsubs"><a href="#" id="navmarked" tabindex="2">Sublink 1.1</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink 1.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="4">Sublink 1.3</a></p>
<p class="navbut"><a href="#" tabindex="5">Section 2</a></p>
<p class="navsubs"><a href="#">Sublink 2.1</a></p>
<p class="navsubs"><a href="#">Sublink 2.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#">Sublink 2.3</a></p>
<p class="navbut"><a href="#" tabindex="6">Section 3</a></p>
<p class="navsubs"><a href="#">Sublink 3.1</a></p>
<p class="navsubs"><a href="#">Sublink 3.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#">Sublink 3.3</a></p>
<p class="navbut"><a href="#" tabindex="7">Section 4</a></p>
<p class="navsubs"><a href="#">Sublink 4.1</a></p>
<p class="navsubs"><a href="#">Sublink 4.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#">Sublink 4.3</a></p></td>
```

The next order of business is to get **tabindex** properties on each link and to rearrange the existing tab indices. Tabindex 1 through tabindex 5 are correct.

-Add *tabindex* properties to **sublinks 2.1-2.3**

```
<p class="navsubs"><a href="#" tabindex="6">Sublink 2.1</a></p>
<p class="navsubs"><a href="#" tabindex="7">Sublink 2.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="8">Sublink 2.3</a></p>
```

-Change the Section 3 tabindex from **tabindex="6"** to **tabindex="9"**

```
<p class="navbut"><a href="#" tabindex="9">Section 3</a></p>
```

-Add *tabindex* properties to **sublinks 3.1-3.3**

```
<p class="navsubs"><a href="#" tabindex="10">Sublink 3.1</a></p>
<p class="navsubs"><a href="#" tabindex="11">Sublink 3.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="12">Sublink 3.3</a></p>
```

-Change the Section 4 tabindex from **tabindex="7"** to **tabindex="13"**

```
<p class="navbut"><a href="#" tabindex="13">Section 4</a></p>
```

-Add *tabindex* properties to **sublinks 4.1-4.3**

```
<p class="navsubs"><a href="#" tabindex="14">Sublink 4.1</a></p>
<p class="navsubs"><a href="#" tabindex="15">Sublink 4.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="16">Sublink 4.3</a></p></td>
```

Now we need to separate each block of sublinks into discrete page divisions.

THE GREAT DIVIDE

The final preparatory step is to separate each set of sublinks into *logical page divisions* (using the `<div>` tag) and then to assign both an **ID** (sub1-sub4) *and* a **Class** (hideit) to each `<div>`, which will enable the SwapClass script to reference the elements and styles needed for our menu to work.

We'd hate to miss the opportunity to show you another new Dreamweaver MX feature—especially if it's one that involves **Code View**! So we'll use a new technique to wrap each group of sublinks into it's own division `<div>`.

-You should still be in **Code View** (if not, please go there)

-Select the first group of sublinks, including their entire paragraph `<p>` tags. We'll help you along by coloring our text to simulate how the code will look once you select it:

```
<p class="navsubs"><a href="#" tabindex="2">Sublink 1.1</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink 1.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="4">Sublink 1.3</a></p>
```

-Right-click anywhere inside the selected code

-Choose **Insert Tag**

The *Tag Chooser* window will open

-Select the *HTML Tags* category on the left, and scroll down the window on the right until you find *div*

-Select **div**

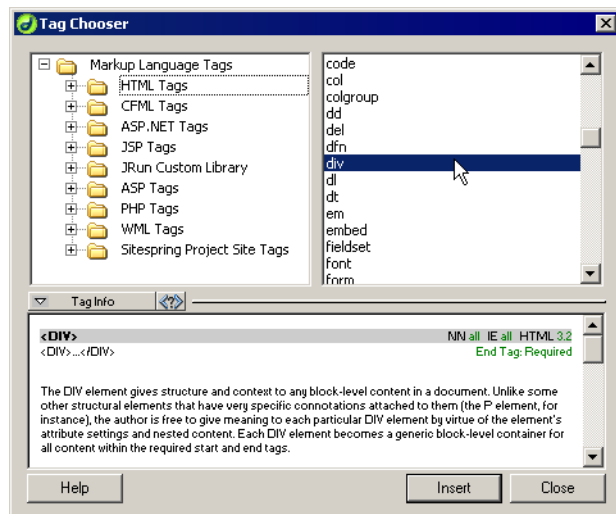


FIGURE 277

-Click **Insert**

The *Tag Editor* window will open

-Select *Style Sheet/Accessibility* and set the following attributes:

-Class: **Leave Blank**

We're not assigning a class to this <div> because we want this particular menu to stay open at all times. Setting the class (even to **showit**) will cause the menu to be affected by SwapClass and it will close the first time you click one of the other menu headers. If you later want to *activate* this menu, simply apply the **hideit** class to the <div> tag.

-ID: **sub1**

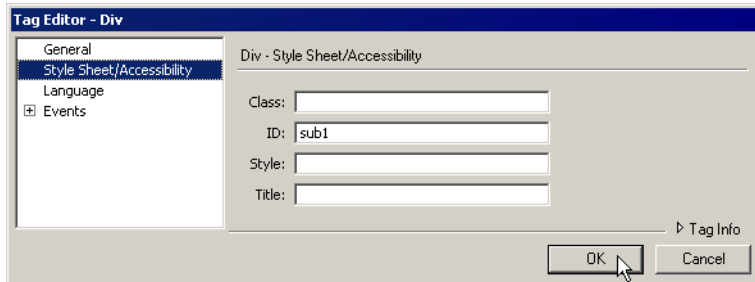


FIGURE 278

-Click **OK**

The *Tag Editor* window will close, but the *Tag Chooser* window will remain open.

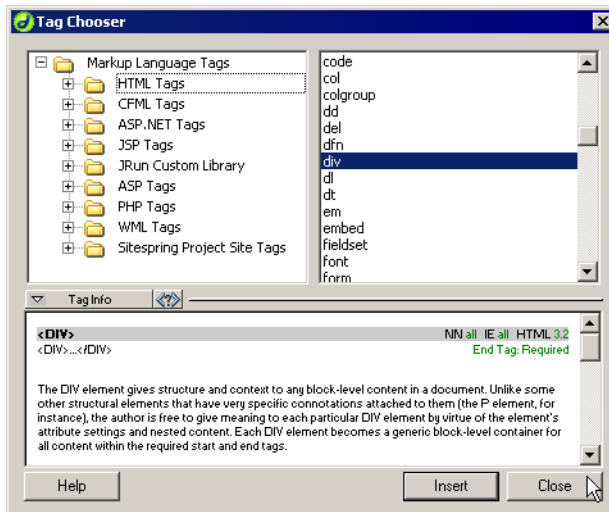


FIGURE 279

Important: DO NOT CLICK THE INSERT BUTTON THIS TIME¹—

-Click **Close**

Dreamweaver will insert opening and closing <div> tags, class, and ID:

```
<div id="sub1"><p class="navsubs"><a href="#" tabindex="2">Sublink 1.1</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink 1.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="4">Sublink 1.3</a></p></div>
```

¹ If you click Insert again, Dreamweaver will insert another pair of <div> tags and open up the Tag Editor window again. Whether this is a bug or a quirk—just don't forget to click Close after your session in the Tag Editor window.

NOW LET'S CREATE THE SECOND GROUP OF SUBLINKS

-Highlight the sublink paragraphs

```
<p class="navsubs"><a href="#" tabindex="6">Sublink 2.1</a></p>
<p class="navsubs"><a href="#" tabindex="7">Sublink 2.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="8">Sublink 2.3</a></p>
```

-Right Click inside the selected code

-Choose **Insert Tag**

The *Tag Chooser* window will open

-Select **div**

-Click **Insert**

The Tag Editor window will open

-Select *Style Sheet/Accessibility* and set the following attributes:

-Class: **hideit**

-ID: **sub2**

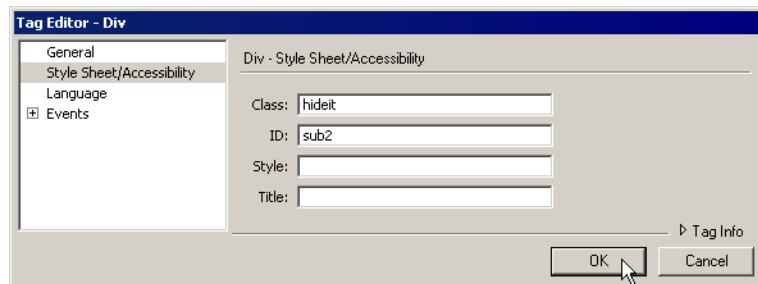


FIGURE 280

-Click **OK**

-Close the *Tag Chooser* window

Dreamweaver will insert opening and closing `<div>` tags, class, and ID:

```
<div class="hideit" id="sub2"><p class="navsubs"><a href="#" tabindex="6">Sublink 2.1</a></p>
<p class="navsubs"><a href="#" tabindex="7">Sublink 2.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="8">Sublink 2.3</a></p></div>
```

LET'S DO THE THIRD GROUP OF SUBLINKS

-Highlight the sublink paragraphs

```
<p class="navsubs"><a href="#" tabindex="10">Sublink 3.1</a></p>
<p class="navsubs"><a href="#" tabindex="11">Sublink 3.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="12">Sublink 3.3</a></p>
```

-Right Click inside the selected code

-Choose **Insert Tag**

The *Tag Chooser* window will open

-Select **div**

-Click **Insert**

The Tag Editor window will open

-Select *Style Sheet/Accessibility* and set the following attributes:

-Class: **hideit**

-ID: **sub3**

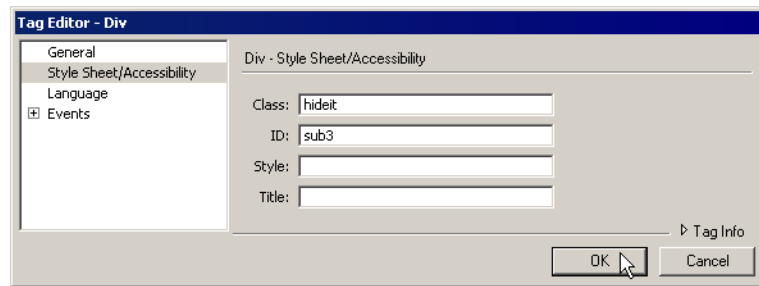


FIGURE 281

-Click **OK**

-Close the *Tag Chooser* window

Dreamweaver will insert opening and closing `<div>` tags, class, and ID:

```
<div class="hideit" id="sub3"><p class="navsubs"><a href="#" tabindex="10">Sublink 3.1</a></p>
<p class="navsubs"><a href="#" tabindex="11">Sublink 3.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="12">Sublink 3.3</a></p></div>
```

LET'S DO THE FOURTH GROUP OF SUBLINKS

-Highlight the sublink paragraphs

```
<p class="navsubs"><a href="#" tabindex="14">Sublink 4.1</a></p>
<p class="navsubs"><a href="#" tabindex="15">Sublink 4.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="16">Sublink 4.3</a></p> </td>
```

Note: Be sure not to select the ending </td> tag. Go just to the closing </p> tag.

-Right Click inside the selected code

-Choose **Insert Tag**

The *Tag Chooser* window will open

-Select **div**

-Click **Insert**

The Tag Editor window will open

-Select *Style Sheet/Accessibility* and set the following attributes:

-Class: **hideit**

-ID: **sub4**

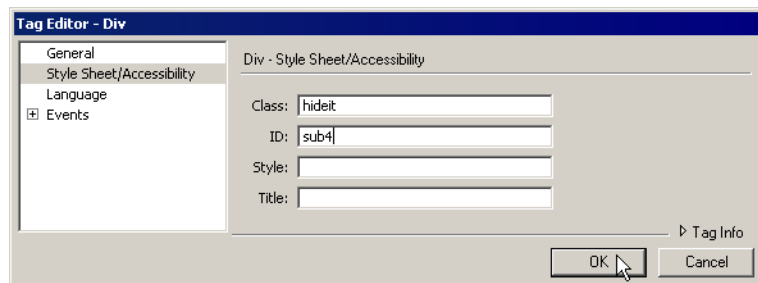


FIGURE 282

-Click **OK**

-Close the *Tag Chooser* window

Dreamweaver will insert opening and closing <div> tags, class, and ID:

```
<div class="hideit" id="sub4"><p class="navsubs"><a href="#" tabindex="14">Sublink 4.1</a></p>
<p class="navsubs"><a href="#" tabindex="15">Sublink 4.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="16">Sublink 4.3</a></p></div>
```

CODE CHECK

The Sidebar `<td>` code should now look like this:

```
<td valign="top" id="sidebar"><p class="navbut"><a href="#" tabindex="1">Section 1</a></p>
<div id="sub1">
<p class="navsubs"><a href="#" id="navmarked" tabindex="2">Sublink 1.1</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink 1.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="4">Sublink 1.3</a></p>
</div>
<p class="navbut"><a href="#" tabindex="5">Section 2</a></p>
<div id="sub2" class="hideit">
<p class="navsubs"><a href="#" tabindex="6">Sublink 2.1</a></p>
<p class="navsubs"><a href="#" tabindex="7">Sublink 2.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="8">Sublink 2.3</a></p>
</div>
<p class="navbut"><a href="#" tabindex="9">Section 3</a></p>
<div id="sub3" class="hideit">
<p class="navsubs"><a href="#" tabindex="10">Sublink 3.1</a></p>
<p class="navsubs"><a href="#" tabindex="11">Sublink 3.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="12">Sublink 3.3</a></p>
</div>
<p class="navbut"><a href="#" tabindex="13">Section 4</a></p>
<div id="sub4" class="hideit">
<p class="navsubs"><a href="#" tabindex="14">Sublink 4.1</a></p>
<p class="navsubs"><a href="#" tabindex="15">Sublink 4.2</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="16">Sublink 4.3</a></p></div></td>
```

Depending on how your Code Editing preferences are set, your `<div>` tags may not have line breaks. The `<div>` tags may appear on the same lines as the `<p>` tags, like this:

```
<div id="sub1"><p class="navsubs"><a href="#" id="navmarked" tabindex="2">Sublink 1.1</a></p>
```

EDITING THE TAG LIBRARY

Either way is correct. I prefer more separation and find it makes the code easier to read. Note that the last closing `</div>` tag is on the same line, between the closing `</p>` and closing `</td>`. That should not be changed or some browsers could render an extra space at the bottom of the table cell.

If you'd like your code to look like mine, you can do it manually, or you can select:

-Edit→Tag Libraries...

When the Tag Library Editor opens, select the **div** tag and set the following attributes:

- Line Breaks*: **Before, Inside, After**
- Contents*: **Formatted But Not Indented**
- Case*: Leave as **Default**

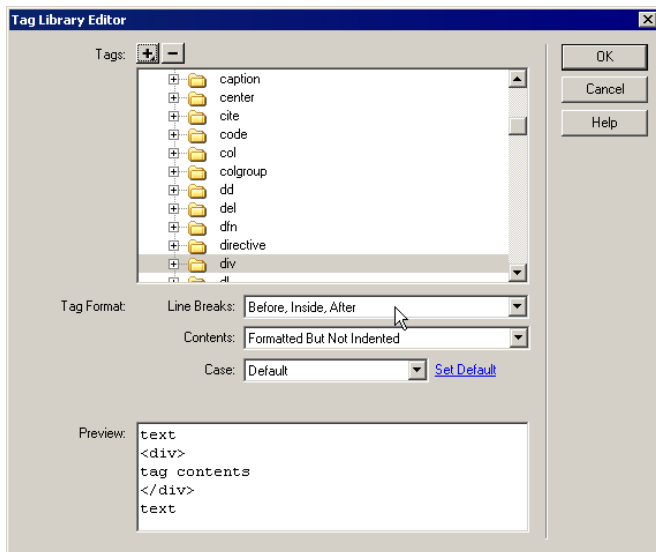


FIGURE 283

- Click **OK**

To affect code that is already in your document, you'll need to select **Commands**→**Apply Source Formatting**.

APPLYING THE SWAPCLASS JAVASCRIPT BEHAVIOR

If you preview the page now (in MSIE or Mozilla) the sublinks will be hidden (except for the first sublink group).

To make the menu come alive, we'll add the SwapClass JavaScript behavior.

APPLY SWAPCLASS TO THE MAIN SECTION LINKS

- Switch to **Design View**
- Place your cursor inside the main link: **Section 2**¹
- Open the **Behaviors** panel (shift+F3)
- Click the **plus (+)** sign and Choose **Studio VII**→**CSS**→**SwapClass by PVII**

The SwapClass window will open—

¹ Remember, we don't want the first menu to be involved—so we start with the second menu.

-From the *Swap From* list, choose **showit**

-From the *Swap To* list, choose **hideit**

If you're wondering why we are going From: **showit** and To: **hideit**, remember that our first instruction is to find all menus that are open and to close them. Refer to the [Menu Mechanics](#) section earlier in this Chapter.

-In the *Tags To Use* box, select **DIV**

-Click the **Select Tag** button (the word *Selected* will appear in parentheses after **DIV**)

-From the *Set This Element* list, choose **sub2**

-The *to the* _____ *Class* box will contain **showit**, but will be *grayed out*

Note: We're instructing SwapClass to look for the **sub2** element then change its class to **showit**. The **showit** class is grayed out and cannot be edited because it will always be the same class as the one selected in the *Swap From* box.

-Put a **check mark** in the *Toggle Option* box (because we want SwapClass to reverse its behavior each time the user clicks)

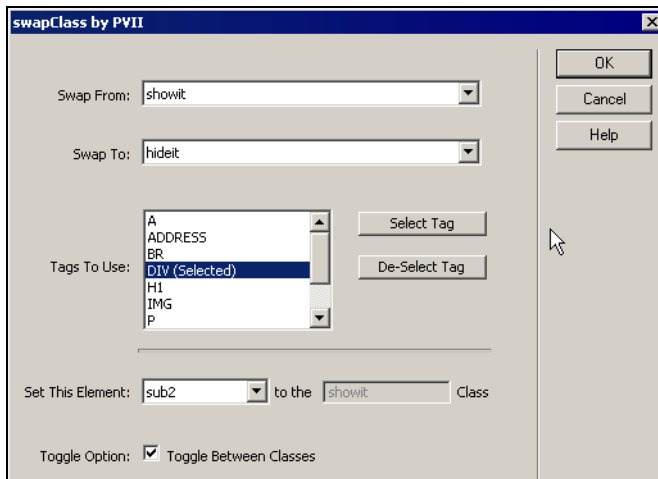


FIGURE 284

-Click **OK**

The Behaviors panel will show the following event:

onClick—swapClass by PVII

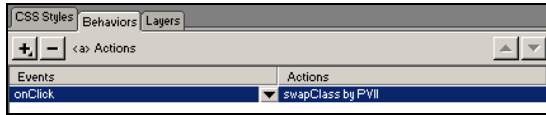


FIGURE 285

Note: Should you happen to see a different event, such as *onMouseOver*, click the arrow between the *Events* and *Actions* sections to open the *Events menu*. Look for the **Show Events For** option. Select it, then choose at least **4.0 and Later Browsers**. Now you'll be able to change the event to **onClick**.

Now let's do the remaining two main Section links

-Place your cursor inside the main link: **Section 3** and open the SwapClass window again

The settings will be identical to those used for the **Section 2** link, except that *Set This Element* will be set to: **sub3** instead of *sub2*.

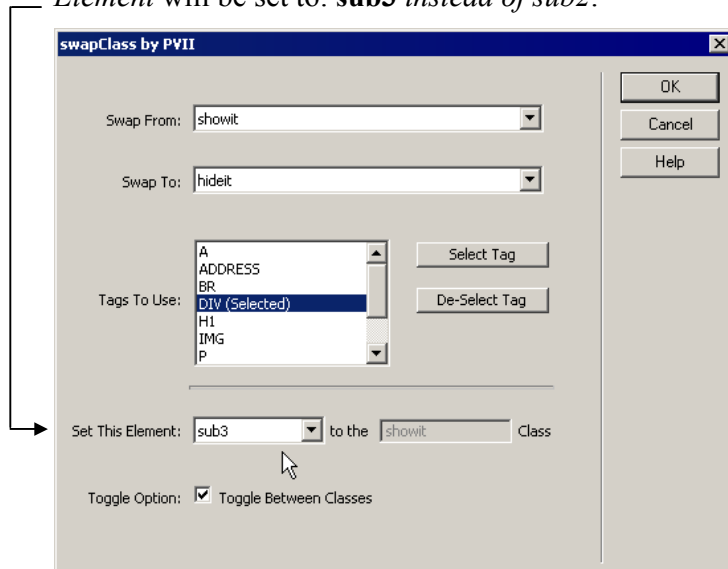


FIGURE 286

-Click **OK**

Be sure the event in the Behaviors panel is set to onClick.

-Place your cursor inside the main link: **Section 4** and open the SwapClass window

The settings will be identical to those used for the **Section 3** link, except that *Set This Element* will be set to: **sub4** instead of *sub3*.

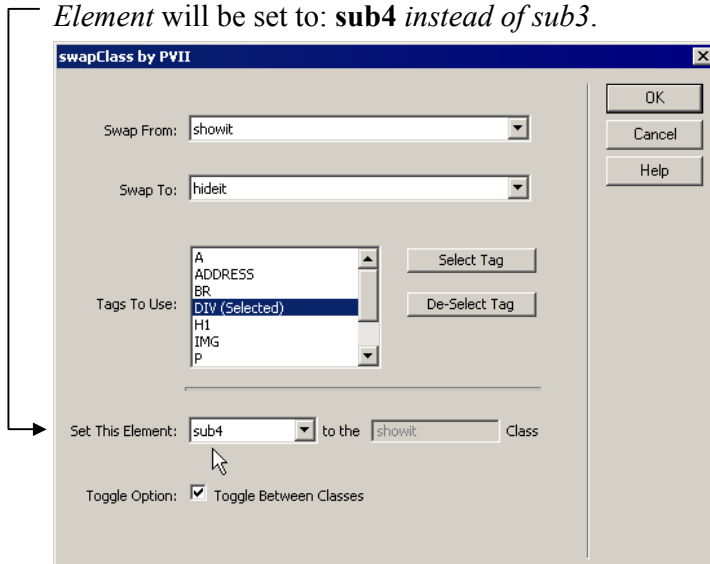


FIGURE 287

-Click **OK**

Be sure the event in the Behaviors panel is set to onClick.

Preview in MSIE 5 (and higher) or Navigator 6 (and higher)

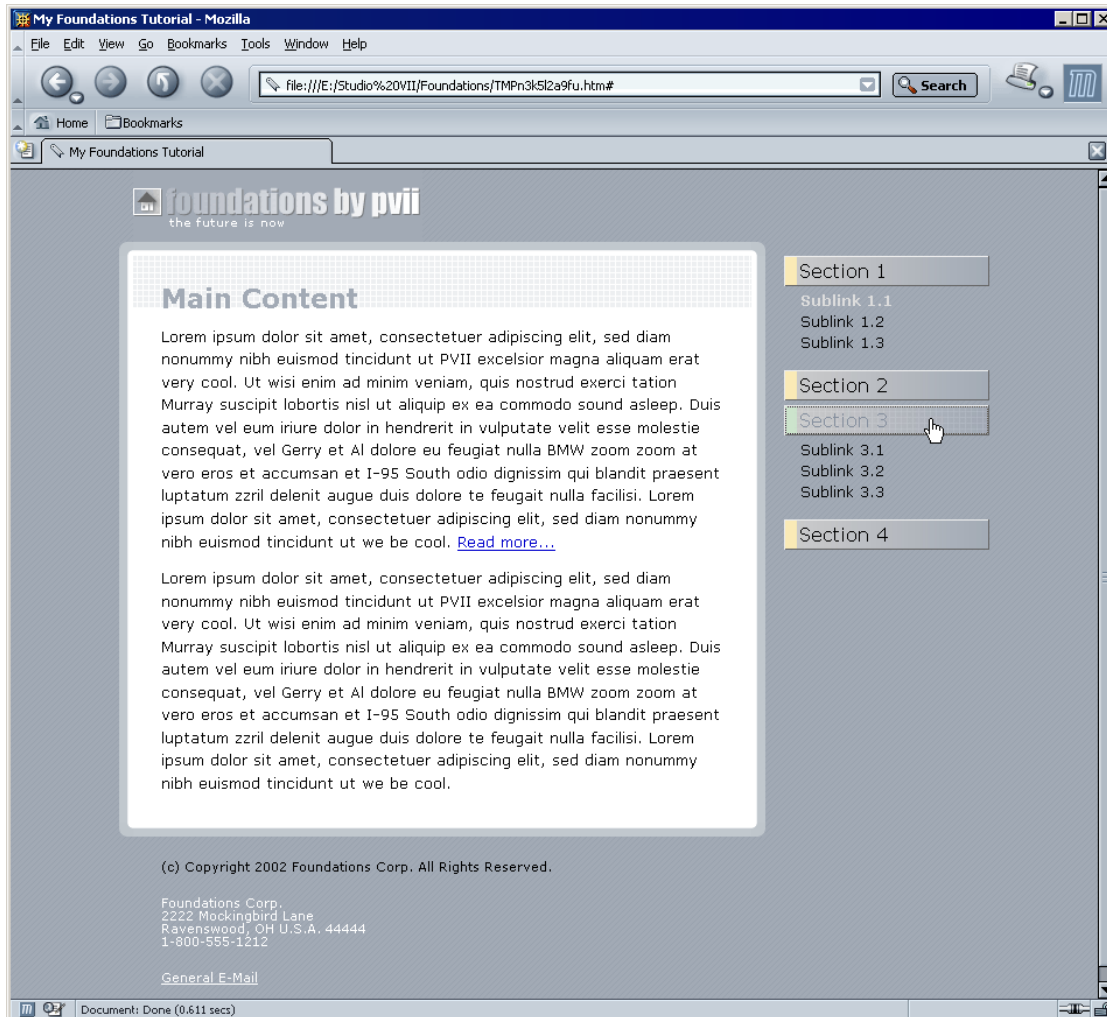


FIGURE 288

Clicking **Section 2**, **Section 3**, or **Section 4** will open its associated *sublink group* and close all the others—*except for the Section 1 group*.

Since we chose the **Toggle** option in the SwapClass window, clicking any of the *Section links* a second time will close its *sublink group*.

MENU MODIFICATIONS

Currently, our first menu section is **locked** to indicate that we are on a page that relates to that section. Since most Web surfers would tend to relate a hover effect to mean that something will happen if he or she clicks, it might be misleading to have a hover effect on our locked menu section header. Instead, the locked menu section header should always look like the header on a menu that has been opened—a kind of *you are here* indicator. We should define a rule to combine the three¹ states we're using on the other section header links: *a*, *hover*, and *active*. It should use properties common to the **p.navbutts a** and the **p.navbutts a:hover**, **p.navbutts a:active** rules—but the *background* and *color* properties should be the ones from the existing hover/active rule. This rule only needs to exist in the version 5 style sheet.

-Open **mfv5.css**

We'll name the new rule:

p.navbutts-dn a, p.navbutts-dn a:hover, p.navbutts-dn a:active

Here are the existing rules, from which we'll produce the new one:

```
p.navbutts a {
    text-decoration: none;
    background-color: #A2AAB5;
    background-image: url(assets/mfimages/navbg.jpg);
    background-repeat: no-repeat;
    display: block;
    padding: 3px 0px 3px 12px;
    width: 162px;
    border-top: 1px solid #EEEEEE;
    border-right: 1px solid #666666;
    border-bottom: 1px solid #666666;
    border-left: 1px solid #EEEEEE;
}
p.navbutts a:hover, p.navbutts a:active {
    color: #858F9E;
    background-color: #CCCCCC;
    background-image: url(assets/mfimages/navbg_on.jpg);
    background-repeat: no-repeat;
}
```

¹ One thing we haven't discussed, is the omission of a **Visited** state. We could do one, but our navigation bar is emulating the behavior of an image-based navbar and it might be a bit much to include a separate rule for visited links. You can add one if you like.

NOW LET’S GO THROUGH EACH OF THE LINES.

```
p.navbutts a {
    ✓text-decoration: none;
    ✗background-color: #A2AAB5;
    ✗background-image: url(assets/mfimages/navbg.jpg);
    ✗background-repeat: no-repeat;
    ✓display: block;
    ✓padding: 3px 0px 3px 12px;
    ✓width: 162px;
    ✓border-top: 1px solid #EEEEEE;
    ✓border-right: 1px solid #666666;
    ✓border-bottom: 1px solid #666666;
    ✓border-left: 1px solid #EEEEEE;
}
p.navbutts a:hover, p.navbutts a:active {
    ✓color: #858F9E;
    ✓background-color: #CCCCCC;
    ✓background-image: url(assets/mfimages/navbg_on.jpg);
    ✓background-repeat: no-repeat;
}
```

Since we want the locked section header to look like the other headers when they are being hovered over, we’ll use all of the background properties from the existing *hover/active* rule—and discard those from the **p.navbutts a** rule. We’ll choose the text color from the *hover/active* rule because it’s matched to that rule’s background properties.

HERE’S THE NEW RULE:

```
p.navbutts-dn a, p.navbutts-dn a:hover, p.navbutts-dn a:active {
    text-decoration: none;
    color: #858F9E;
    background-color: #CCCCCC;
    background-image: url(assets/mfimages/navbg_on.jpg);
    background-repeat: no-repeat;
    display: block;
    padding: 3px 0px 3px 12px;
    width: 162px;
    border-top: 1px solid #EEEEEE;
    border-right: 1px solid #666666;
    border-bottom: 1px solid #666666;
    border-left: 1px solid #EEEEEE;
}
```

-Add this new rule to the **mfv5** style sheet

The above rules affect paragraphs styled with the **p.navbutts-dn** selector, which does not yet exist. Let’s make it now...

This selector will set font-size and margins for the paragraph. Rather than creating a separate new rule, we can edit **p.navbutts** to make it a combined selector:

The current p.navbutts rule:

```
p.navbutts {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}
```

-Edit the rule to include the *p.navbutts-dn* selector:

```
p.navbutts, p.navbutts-dn {  
    font-size: 130%;  
    margin: 0px 0px 4px;  
}
```

-Apply the p.navbutts-dn class to the paragraph tag of the **Section 1** header

-Save and Close mf5.css

-Preview in a version 5 or higher browser:

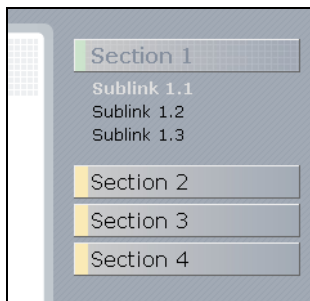


FIGURE 289

The **Section 1** header is now *locked*. You can point your mouse at it all day long and nothing will happen. We can call it a *down* state, if you like.

What happens when you create a page that is associated with **Section 2**? You'd probably want the Section 2 *sublink group* to be **locked** open. And you'd probably also want to assign the **p.navbutts-dn** class to the paragraph in which the Section 2 header is wrapped. To make this change, you would:

- Set **class="hideit"** on the *section 1 sublink group* **<div>** tag
- Remove the *hideit* class from the *section 2 sublink group* **<div>** tag
- Change the Section 1 **<p>** tag to the **navbutts** class
- Remove the **navmarked** ID on the Sublink 1.1 **<p>** tag
- Change the Section 2 **<p>** tag to the **navbutts-dn** class
- Apply the **navmarked** ID to one of the *Section 2 sublink* **<p>** tags (optional)
- Use the Behavior panel to remove the SwapClass event from the Section 2 **<a>** tag
- Apply a **SwapClass** behavior to the **Section 1** link to control its *sublink group*

That was easy!

Now let's move on to a completely different topic...

Chapter 13-Externalizing the SwapClass JavaScript

If you plan to deploy this type of menu in a site that encompasses more than a few pages, you may want to consider moving the JavaScript to an external file and linking that file to each page of your site.

Even though our SwapClass script is small, externalizing it provides management efficiencies much like an external CSS file. For example:

If every page of your site contains the SwapClass script, and you need to edit the script, then you'd need to edit and re-publish each page.

MOVE THE SCRIPT

The script is in the <head> region of your source code—probably just below the imported style sheet. It looks like this:

```
<script language="JavaScript" type="text/JavaScript">
<!--
function P7_swapClass(){ //v1.4 by PVII
var i,x,tB,j=0,tA=new Array(),arg=P7_swapClass.arguments;
if(document.getElementsByTagName){for(i=4;i<arg.length;i++){tB=document.getElementsByTagName(arg[i]);
for(x=0;x<tB.length;x++){tA[j]=tB[x];j++;}}for(i=0;i<tA.length;i++){
if(tA[i].className){if(tA[i].id==arg[1]){if(arg[0]==1){
tA[i].className=(tA[i].className==arg[3])?arg[2]:arg[3];}else{tA[i].className=arg[2];}
}else if(arg[0]==1 && arg[1]=='none'){if(tA[i].className==arg[2] || tA[i].className==arg[3]){
tA[i].className=(tA[i].className==arg[3])?arg[2]:arg[3];}
}else if(tA[i].className==arg[2]){tA[i].className=arg[3];}}}}
}
//-->
</script>
```

-Select the entire script, from the opening tag:

```
<script language="JavaScript" type="text/JavaScript">
```

to the closing tag: </script>

-Press **Ctrl+X** to remove the code and place a copy in your clipboard

-Close and Save **index.htm**

CREATE THE JAVASCRIPT FILE

-In the **Site panel**, select the site's *root folder*

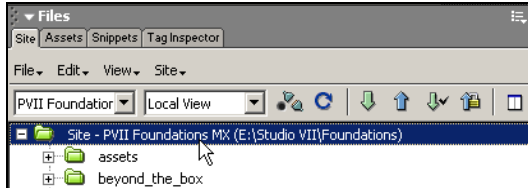


FIGURE 290

-From the Site panel's *File* menu, choose **File→New File**

-A new entry: **untitled.htm** will appear and its name will be *ready for you to edit*

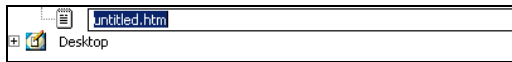


FIGURE 291

-Overwrite *untitled.htm* with **myscript.js**

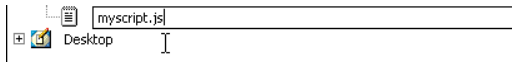


FIGURE 292

-Open **myscripts.js** (Code View is the only available view)

-**Delete** all the *HTML code* in the file so that Code View is completely blank

-Press **Ctrl+V** to paste in the JavaScript

-Remove the opening and closing script tags, and the comments, so that your **myscripts.js** file contains only the following:

```
function P7_swapClass(){ //v1.4 by PVII
var i,x,tB,j=0,tA=new Array(),arg=P7_swapClass.arguments;
if(document.getElementsByTagName){for(i=4;i<arg.length;i++){tB=document.getElementsByTagName(arg[i]);
for(x=0;x<tB.length;x++){tA[j]=tB[x];j++;}}for(i=0;i<tA.length;i++){
if(tA[i].className){if(tA[i].id==arg[1]){if(arg[0]==1){
tA[i].className=(tA[i].className==arg[3])?arg[2]:arg[3];}else{tA[i].className=arg[2];}
}else if(arg[0]==1 && arg[1]=='none'){if(tA[i].className==arg[2] || tA[i].className==arg[3]){
tA[i].className=(tA[i].className==arg[3])?arg[2]:arg[3];}
}else if(tA[i].className==arg[2]){tA[i].className=arg[3];}}}}
}
```

-**Save and Close** **myscripts.js**

LINK THE JAVASCRIPT FILE TO YOUR PAGE

- Open **index.htm** and switch to **Code View**
- Create a new line just above the closing `</head>` tag
- Right-click on the new line and choose **Insert Tag** to open the *Tag Chooser* window
- Select **HTML Tags**, then highlight **script** in the box on the right

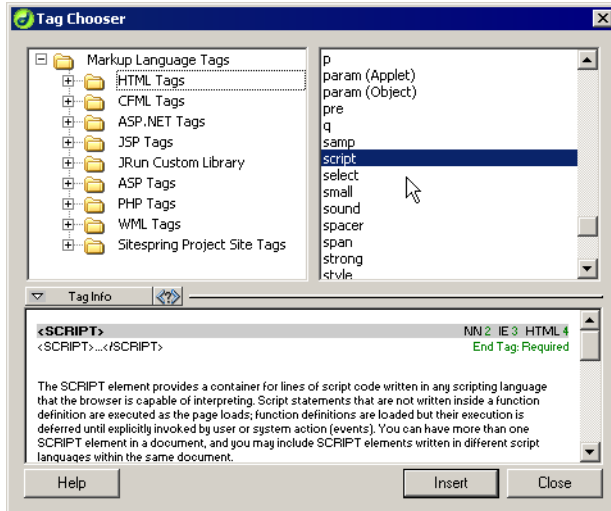


FIGURE 293

- Click **Insert** to open the *Tag Editor* window
- Leave the *Language* box empty
- Select *Type*: **text/javascript**
- For *Source*, type in: **myscripts.js** (or use the Browse button)

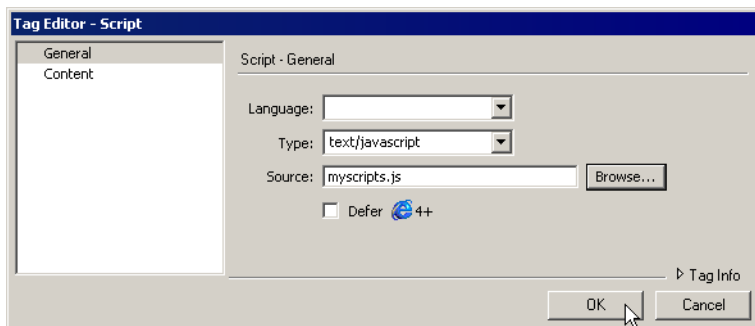
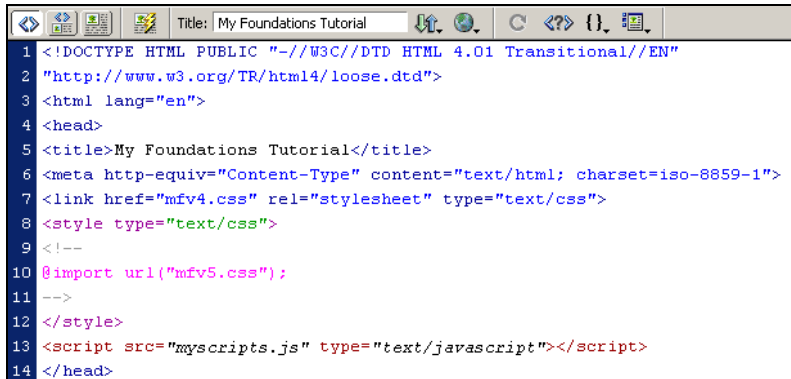


FIGURE 294

- Click **OK**
- The *Tag Chooser* window will still be open.
- Click **Close** (do not click *Insert* again)

Your link will be inserted where you right-clicked earlier:



```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
2 "http://www.w3.org/TR/html4/loose.dtd">
3 <html lang="en">
4 <head>
5 <title>My Foundations Tutorial</title>
6 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
7 <link href="mfv4.css" rel="stylesheet" type="text/css">
8 <style type="text/css">
9 <!--
10 @import url("mfv5.css");
11 -->
12 </style>
13 <script src="myscripts.js" type="text/javascript"></script>
14 </head>
```

FIGURE 295

-Preview the page. Everything should function as before.

-Save Index.htm

Note: Should you ever need to add new scripts or edit the existing SwapClass script, then make your additions or edits to this one file. You can also link additional JavaScript files to your page, if that is more suitable to your situation.

Chapter 14—Alternate Image Sets

**“DOING A LITTLE WORK AROUND THE HOUSE,
I PUT FAKE BRICK WALLPAPER OVER A REAL BRICK WALL—
JUST SO I'D BE THE ONLY ONE WHO KNEW.
WHEN PEOPLE COME OVER I'M GONNA SAY,
GO AHEAD, TOUCH IT...IT FEELS REAL.”—STEPHEN WRIGHT**

OVERVIEW

Changing image sets is as easy as changing your shoes and *lots* easier than changing the wallpaper. We’ve supplied three alternate sets for you: Yellow, Dusty, and Purple. Each set is derived from a Fireworks MX document—just like the one you created in Chapter 7.

Our page currently looks like this:

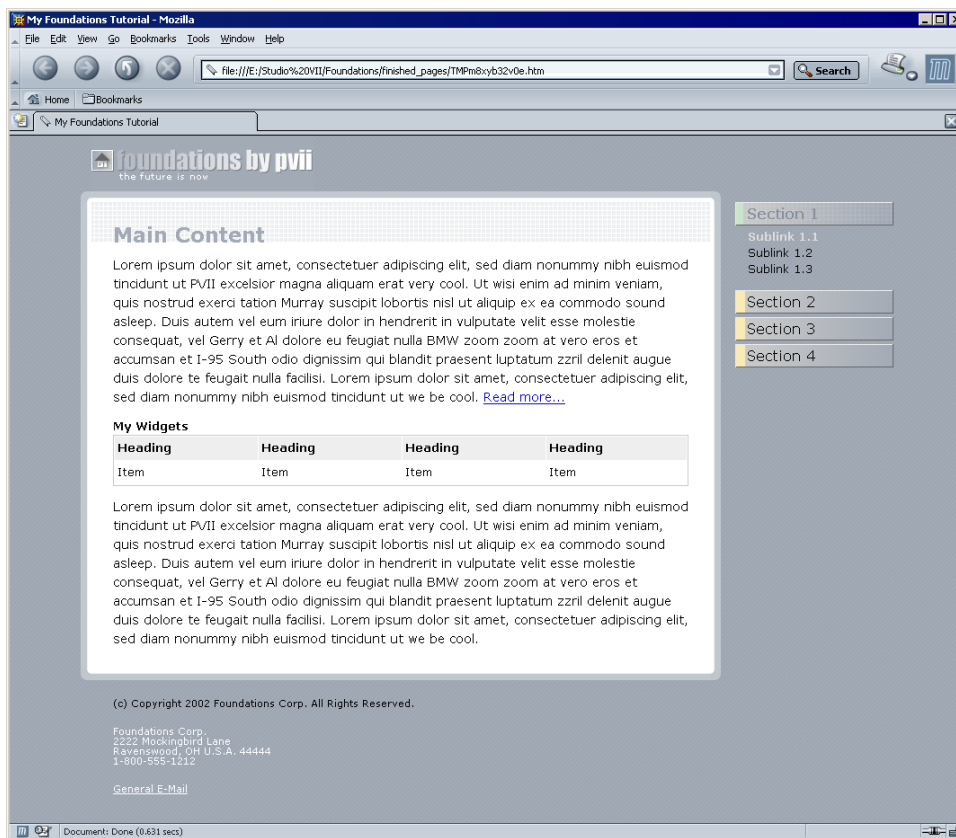


FIGURE 296- THE VIEW FROM MOZILLA 1.1

In a few moments, we’re going to witness a dramatic change.

CHANGING THE IMAGES

-Close all open Dreamweaver documents

-Launch Fireworks

-Choose **File**→**Open**

-Browse to the *Local Files\Fireworks* folder of your Foundations site and choose the file **yellow.png**

-Click **Open**

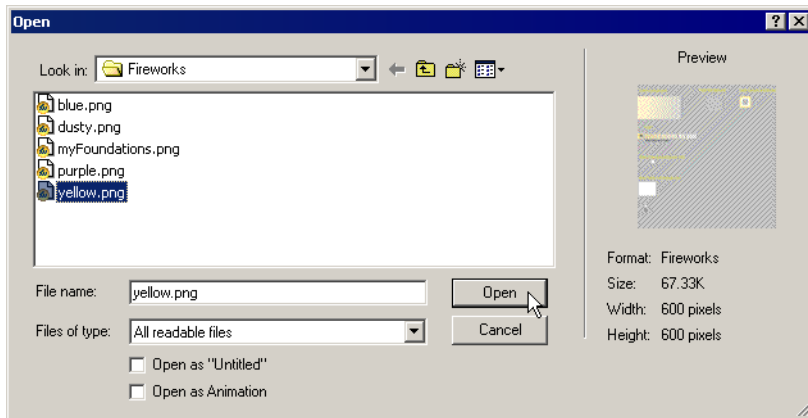
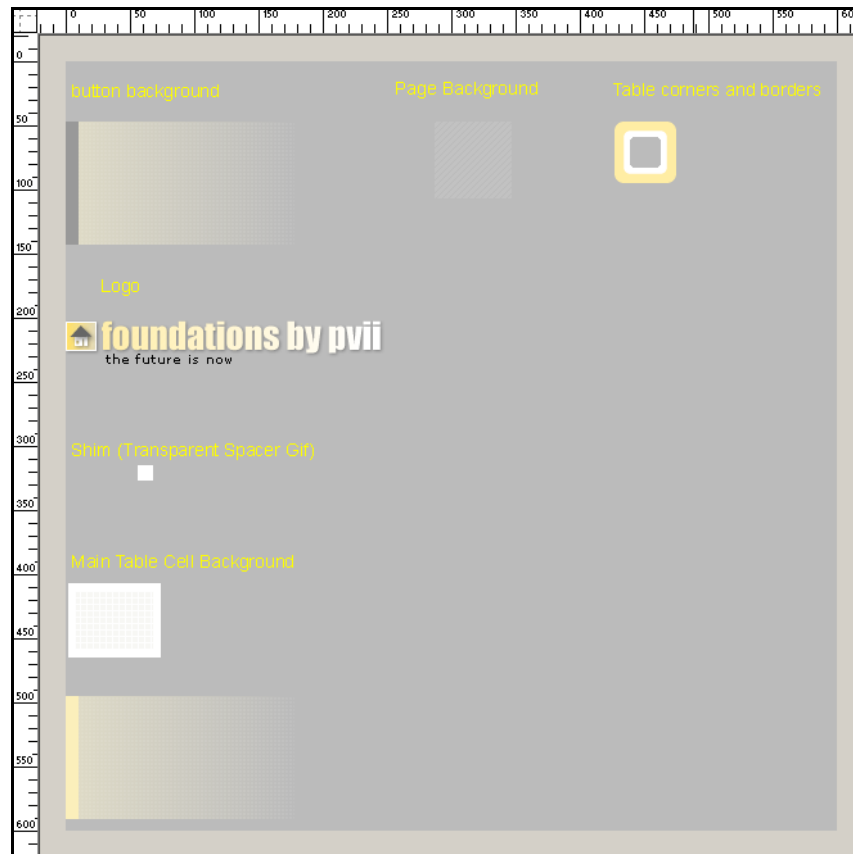


FIGURE 297

Look familiar? We made a copy of the Fireworks document—then applied different colors and textures to the elements. The slices are precisely the same size and placement. Slice names are identical. Optimization is the same.

YELLOW.PNG**FIGURE 298**

Ready to change your shoes?

All you have to do is export the images to your Foundations site. The existing images in the *mfimages* folder will be overwritten. Don't worry—it'll be just as easy to change things back.

-Choose **File→Export**

-Browse to the **assets\mfimages** folder in your Foundations site

-Make sure *Save as Type: Images Only* is selected

-Make sure *Slices: Export Slices* is selected and that *Include Areas without Slices* is **unchecked**

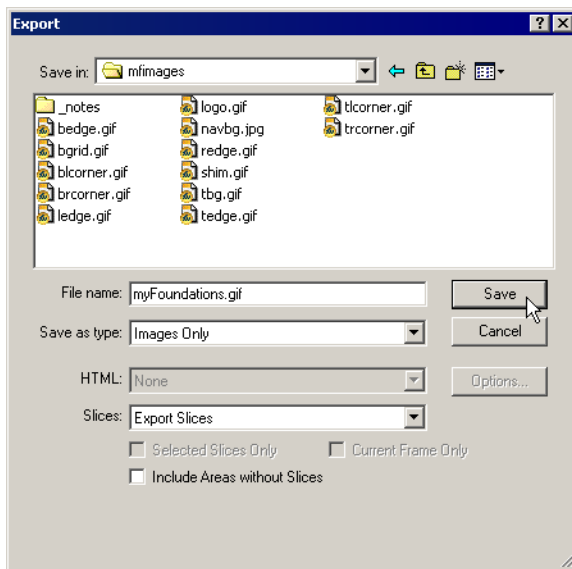


FIGURE 299

-Click **Save**

You should see an Overwrite warning

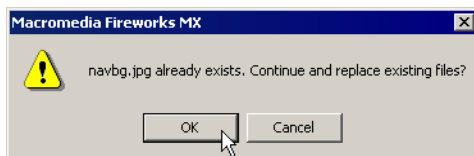


FIGURE 300

-Click **OK**

Now let us see what we have wrought.

-**Save**¹ and close the Fireworks document and shut down Fireworks

¹ Why does Fireworks prompt for a Save after exporting an image—even if you have not edited the document in any way? It's saving your Export settings.

-Switch back to Dreamweaver

-Open index.htm



FIGURE 301

Believe it or not, you've just completed the hard part. All that's left is a few edits to the style sheets—to coordinate colors.

Note: If you do not see the background images, please make sure you have run the [Saved Query](#) that sets the URLs in mfv4.css to relative paths.

MATCHING STYLE SHEET SETS FOR EACH COLOR SET

Before we change the style sheets, we need a plan. We know how easy it is to switch image sets back and forth. If we had a similarly easy way to switch style sheets, we'd be home free.

BACKING UP THE STYLE SHEETS

Before we begin, we need to back up our default style sheets (mfv4 and mfv5). Once we do, we can proceed to edit the originals to make their colors match the yellow image set.

-Open **mfv4.css**

-Choose **File→Save As**

-Enter *File name*: **BLUE_mfv4.css**

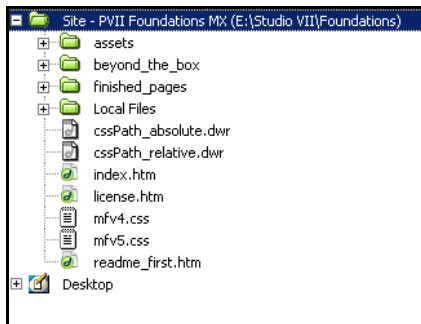


FIGURE 302

-Click **Save**

-Close **BLUE_mfv4.css**

-Open **mfv5.css**

-Choose **File→Save As**

-Enter *File name*: **BLUE_mfv5.css**

-Click **Save**

-Close **BLUE_mfv5.css**

If you look at your **Site** panel, you'll see the two copies listed.

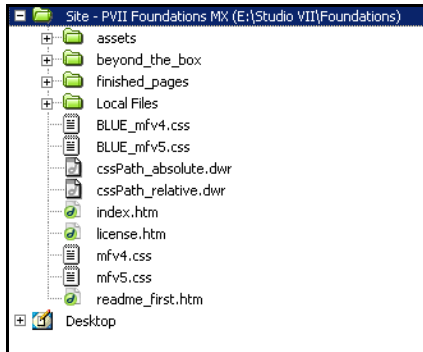


FIGURE 303

The two new CSS files you created are your backups for your original style sheets—which are associated with *blue* colors. Now we'll also need to make two additional copies—which will be backups for the yellow color scheme.

- Open **mfv4.css**
- Choose **File**→**Save As**
- Enter *File name*: **YELLOW_mfv4.css**
- Click **Save**
- Close **YELLOW_mfv4.css**
- Open **mfv5.css**
- Choose **File**→**Save As**
- Enter *File name*: **YELLOW_mfv5.css**
- Click **Save**
- Close **YELLOW_mfv5.css**

If you look at your **Site** panel, you'll see the two new copies listed.

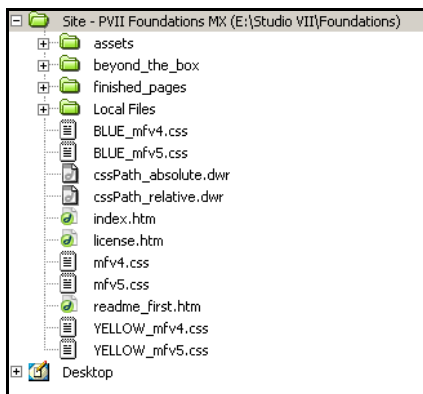


FIGURE 304

Now we have four new CSS files: two **YELLOWs** and two **BLUEs**. These are backups. The BLUEs need no editing. We'll edit the two YELLOWs to coordinate colors with our yellow images. Then we'll save the YELLOWs as our final (yellow) backups. To complete the makeover, we'll use the YELLOWs to overwrite our two default style sheets. To change back to the default theme, all we would need to do is export the original **myFoundations** images and overwrite the default style sheets with the two BLUEs backups.

SETTING DESIGN TIME STYLE SHEETS

We need to be able to edit the **Yellow** style sheets from within `index.htm` so that we can coordinate colors *visually*. *Design Time Style Sheets* (a new MX feature) is just the ticket. We can tell Dreamweaver to use the *Yellow* style sheets in Design View, while hiding the default style sheets. We can do this without having to change our page's source code.

-Open `index.htm`

-Open the **CSS panel** and click the *panel Options Menu expander*

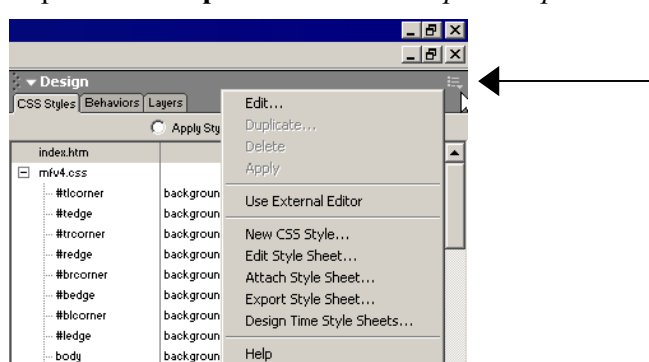


FIGURE 305

-Choose **Design Time Style Sheets**

The Design Time Style Sheet window will open

- Click the **Plus** [+] sign in the *Show Only at Design Time* section
- In the *Select File* window, choose **YELLOW_mfv4.css**
- Repeat the previous step to choose **YELLOW_mfv5.css**
- Click the **Plus** [+] sign in the *Hide at Design Time* section
- In the *Select File* window, choose **mfv4.css**
- Repeat the previous step to choose **mfv5.css**

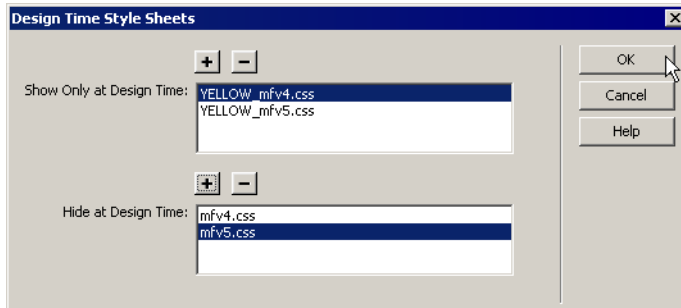


FIGURE 306

- Click **OK**

Your CSS panel¹ will show both YELLOW sheets. The default style sheets will be at the bottom of the panel, with the word “*Hidden*” after them.

- Click the **minus** (-) signs to the left of *yellow_mfv4.css “design”* and *yellow_mfv5.css “design”* to collapse the rules lists, which will make it easier to navigate the two style sheets and their rules.

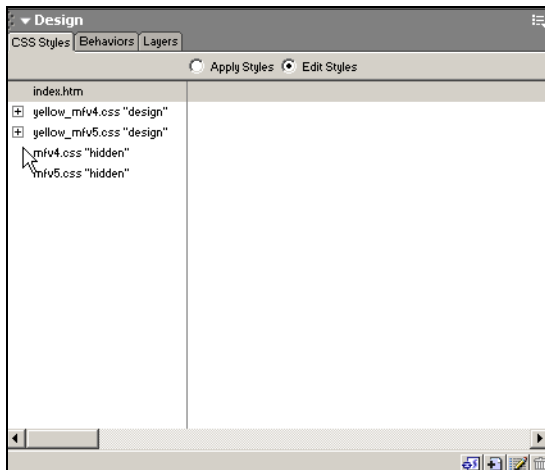


FIGURE 307

¹ Make sure you have the **Edit Styles** option selected and not **Apply Styles**.

EDITING THE YELLOW STYLE SHEETS

Now we're ready to edit the **Yellow** style sheets.

-Click the **plus (+)** sign next to *yellow_mfv4.css* “*design*” to expose the list of its rules

THE BODY RULE

-Double-click the **body** rule

The *CSS Style Definition* window will open

-Switch to the *Background* Category

-Click the *Background Color* color square

-Use the *eyedropper* to pick up the page background. As you move the eyedropper across the page, the color should change to #C2C2C2, then to #BBBBBB in between the diagonal lines. Choose **#C2C2C2**

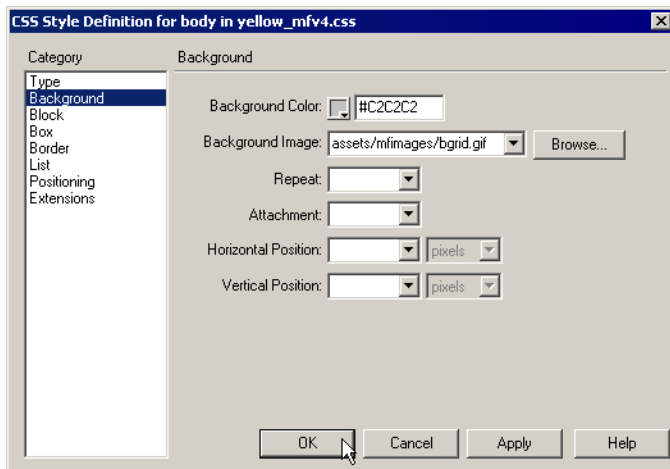


FIGURE 308

-Click **OK**

THE H1 RULE

- In the CSS panel, double-click the **h1** rule
- In the *Type* Category, change *Color* to: **#BBBBBB**
- Click **OK**

P.NAVBUTS A

- In the CSS panel, click the **minus** (-) sign to close the *yellow_mfv4* rules list
- Click the **plus** (+) sign next to **yellow_mfv5.css** to open its list of rules
- Double-click the **p.navbutts a** rule
- Switch to the *Background* Category
- Click the *Background Color* **color square**
- Use the *eyedropper* to pick up the yellow from the *mn table cell border* (it should be **#FFEDA4**)
- Click **OK**

P.NAVBUTS-DN A, P.NAVBUTS-DN A:HOVER, P.NAVBUTS-DN A:ACTIVE

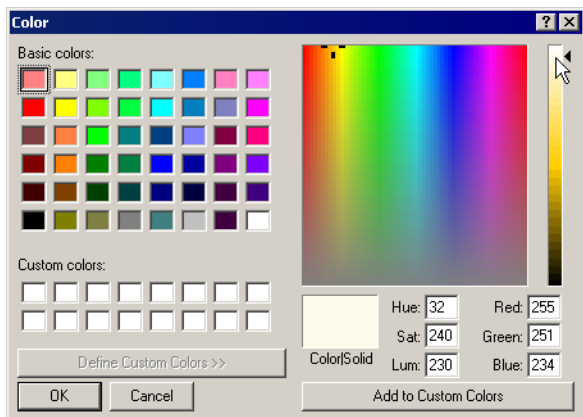
- In the CSS panel, click the **minus** (-) sign to close the *yellow_mfv4* rules list
- Click the **plus** (+) sign next to **yellow_mfv5.css** to open its list of rules
- Double-click the **p.navbutts-dn a, p.navbutts-dn a:hover, p.navbutts-dn a:active** rule
- Switch to the *Background* Category
- Set Background Color to **#CCCCCC**
- Click **OK**

P.NAVBUTS A:HOVER, P.NAVBUTS A:ACTIVE

- In the CSS panel, **close** the *yellow_mfv5* rules list and **open** the *yellow_mfv4* rules list
- Double-click the **p.navbutts a: hover, p.navbutts a: active** rule
- Switch to the *Background* Category
- For Background Color, set **#BBBBBB**
- Click **OK**

P.NAVSUBS A:HOVER, P.NAVSUBS A:ACTIVE

- Double-click the **p.navsubs a: hover, p.navsubs a: active** rule
- In the Type Category, set *Color:* **#BBBBBB**
- Switch to the *Background* Category
- Click the *Background Color color square*
- Use the *eyedropper* to pick up the yellow from the *mn table cell border* (it should be **#FFEDA4**)
- Click the *Background Color color square* again
- Click the **color wheel icon**  along the top of the swatch pop-up
- The *Color* window will open
- Move the shade slider arrow up to fade the yellow to a creamy white hue. Stop when you're happy 😊

**FIGURE 309**

- Click **OK** to close the color window
- Make a note of the number Dreamweaver has inserted in the *Color* box (mine was **#FFFBEA**)
- Click **OK** to close the CSS Style Definition window

#FOOTER A:HOVER, #FOOTER A:ACTIVE

-In the CSS panel, double-click the **#footer a:hover, #footer a:active** rule

-In the Type Category, set *Color*: **#BBBBBB**

-Switch to the *Background* Category

-Set the Background Color to: **#FFFBEA** (the number you noted above)

-Click **OK**

THAT'S IT FOR OUR EDITING

Your page should now look like this:

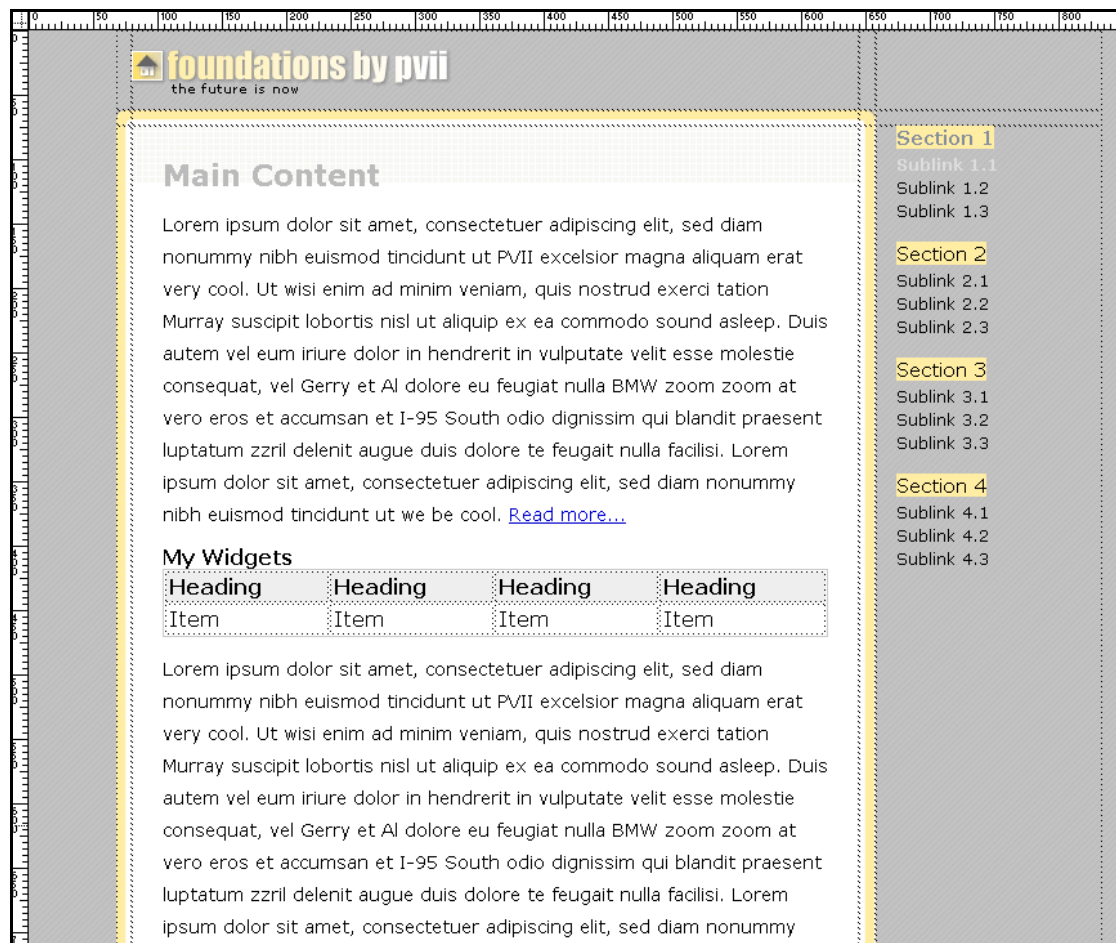


FIGURE 310

TURNING OFF DESIGN TIME STYLE SHEETS

Now you can turn off Design Time Style Sheets. They won't be needed any longer.

-Open the *Design Time Style Sheets* window (use the Option Menu expander in the CSS panel)

-Use the **minus** [-] sign to clear both sections

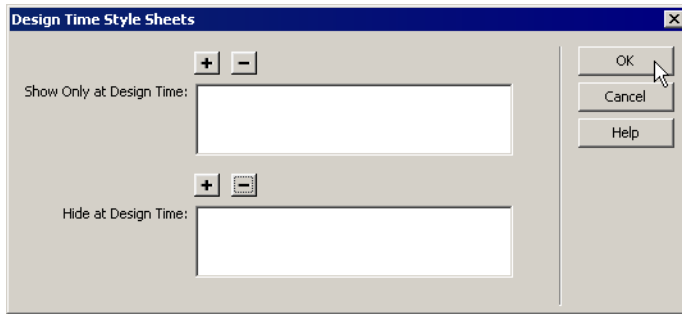


FIGURE 311

-Click **OK**

-**Close** index.htm

RESETTING THE DEFAULT STYLE SHEETS TO THE YELLOW COLOR THEME

- Open **YELLOW_mfv4.css**
- Choose **File**→**Save As**
- Select **mfv4.css** (it will appear in the *File name* box)

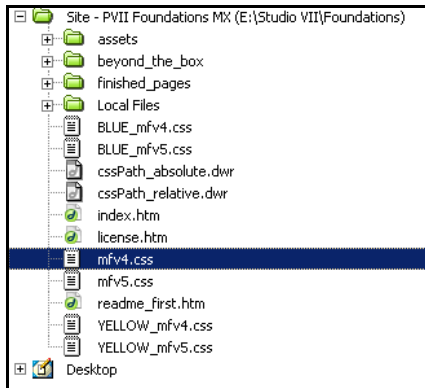


FIGURE 312

- Click **Save**
- You will be prompted for the overwrite.

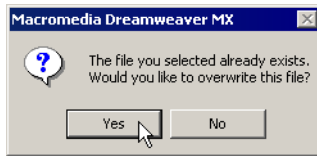


FIGURE 313

- Click **Yes**
- Close the file
- Open **YELLOW_mfv5.css**
- Choose **File**→**Save As**
- Select **mfv5.css** (it will appear in the *File name* box)
- Click **Save**
- Close the file

We're done. Now we can preview the page and see how we did—

-Open index.htm

Preview in a version 5 (or higher) browser:

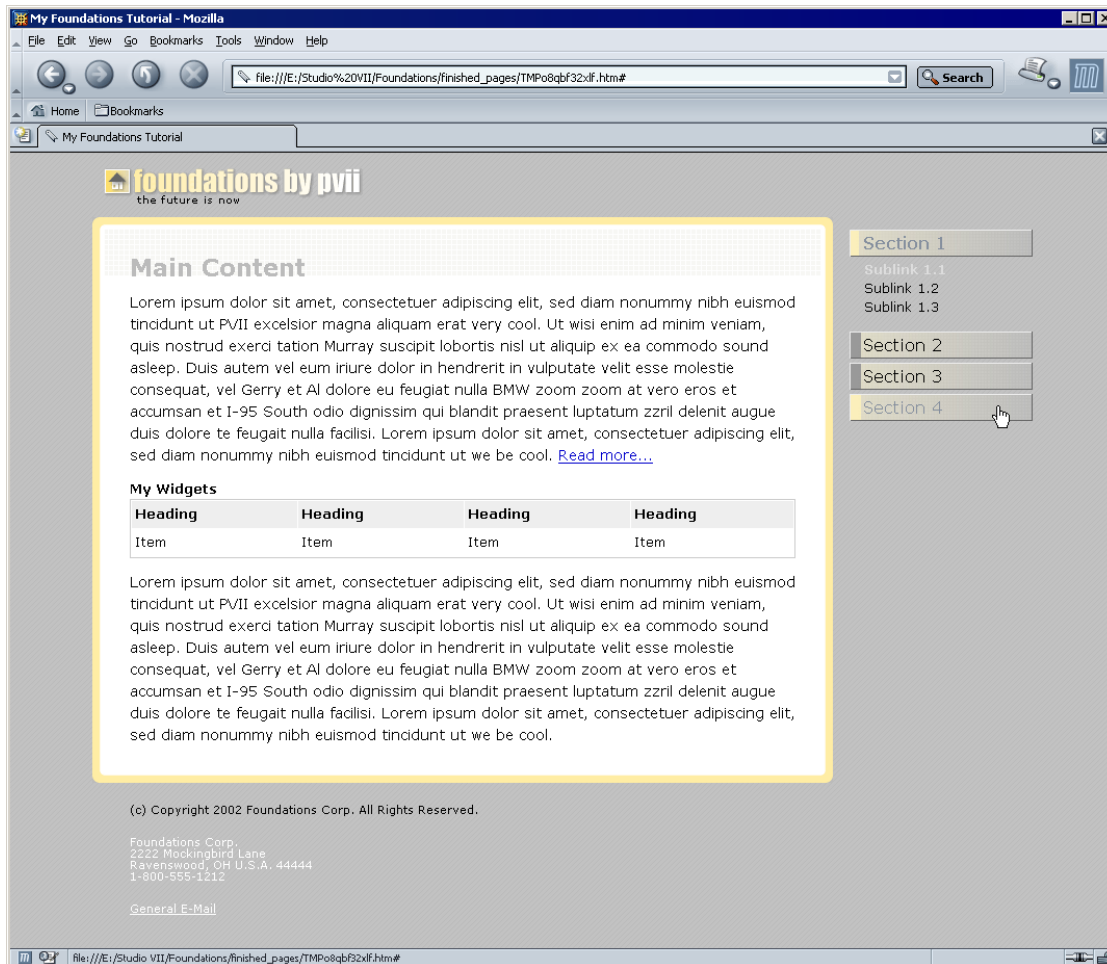


FIGURE 314

Cool!

Now Preview in Navigator 4:



FIGURE 315

Excellent.

Let's recap how this *multi-theme* plan works—

MANAGING MULTIPLE COLOR SCHEMES

We have four Fireworks image sets: Blue, Dusty, Purple, and Yellow. We have four corresponding sets of CSS files—each set containing a v4 and v5 style sheet optimized for the color scheme.

When we want to change color schemes, we:

- ⇒ Open its associated Fireworks document.
- ⇒ Export slices to the **mfimages** folder—overwriting our current images.
- ⇒ Overwrite our default style sheets with the style sheets associated with the color scheme we are changing to.

POP QUIZ:

Do you know how to change your color scheme back to the original blue one you made?

Just in case you don't, here's how:

-Close any open CSS file

-Open **BLUE_mfv4.css**

-Choose **File**→**Save As**

-Name the file: **mfv4.css**

-You will get an overwrite prompt—say **Yes** (you do want to overwrite the existing file)

-Close *mfv4.css*

-Open *BLUE_mfv5.css*

-Choose **File**→**Save As**

-Name the file: **mfv5.css** (say **Yes** to the overwrite prompt)

-Close *mfv5.css*

-Open *Fireworks* and load **myFoundations.png**

-**Export** the *myFoundations* images to the **mfimages** folder of your Foundations site, overwriting the current images

All will be blue again. Now for a little lobotomy—

Chapter 15-The Table Lobotomy

“JUST TAKE A LITTLE OFF THE TOP, PLEASE”.

—ICHABOD CRANE



LOOK MOM—NO TABLE

Page designs based on HTML tables often require more markup than *table-less* designs. However, as we saw in Chapter 6, if your plan calls for a certain style (like rounded corners) then tables are a practical and very viable cross-browser solution. If we didn’t require that our design have round corners our options would expand and we could eliminate the layout table. We could then plan and build a new page using some very different techniques. **This exercise is purely optional** and its pace is a bit faster than previous lessons. You’ll learn how to build a page from the inside out—separating its *structure* from its *presentation*. In fact, we’re going to do most of our work in *Code View*.

WHAT ARE THE BENEFITS OF TABLE-LESS DESIGNS?

CSS layout gives you maximum control. Edits to a single CSS file can reshape *every* page in your site—even if your site has thousands of pages. A document’s structural aspects are kept separate from its presentation because all presentational properties and values are contained in separate, external files—*CSS files*. We’ll discuss this topic often as we progress through the exercise.

PLANNING THE STRUCTURE

We’ll start with a blank page and divide it into six discrete blocks: Logo, Main Content, Footer, Navbar, and News—using `<div>` tags. As we build the main structure, we’ll ID each `<div>`, then enter placeholder content. Upon completing the structure, we’ll develop style sheets to shape and style our content. We’ll use absolute positioning to handle the layout.

While it is possible to use the CSS Float property to create multiple columns, there are just too many inconsistencies in the way different browsers interpret the property—and version 4 browsers, in particular, may act in a very unpredictable way. Using absolute positioning is a safe, effective way to achieve precise layouts in all version 4 and higher browsers—and it gives us the most control over our source code placement. To float an element, the element’s source code must be in a certain location. When an element is absolutely positioned, its html markup can be placed anywhere. For instance, we can position a sidebar on the right side of the screen and a main area to its left—and it won’t matter if the sidebar’s source code comes before or after the main area’s source code. This is not the case when floating. To float (and vertically align) a sidebar to the right of a main content area, the sidebar source code would have to come before the main area source code.

CREATE THE LOBOTOMY PAGE

- Create a new html document and name it **lobotomy.htm**
- Switch to **Code View**
- Use your new doctype **snippet** to add a full **html 4.01 transitional doctype**
- Edit the **<title>** to read: *Look Mom, No Tables* and add a **language** attribute to the opening **<html>** tag

The base page code should look like this:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html lang="en">
<head>
<title>Look Mom, No Tables</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>

</body>
</html>
```

CREATING THE BASE STRUCTURE

- Insert your cursor on the blank line **between the opening <body> and closing </body> tags**, then enter the following code (including the blank lines).

```
<body>
<div id="divlogo">
</div>
<div id="divmain">

<div id="divfooter">

</div>
</div>
<div id="divnavbar">

<div id="divnews">

</div>
</div>
</body>
```


THE BASE STRUCTURE DEFINED

Let's make sense of what our base structural markup means. As we discuss each element, we'll add in the appropriate content.

THE LOGO DIV

The first <div> is simple enough. It will house our logo and nothing else.

```
<div id="divlogo">
```

```
</div>
```

-Place your cursor on the blank line between the opening and closing DIV tags

-Choose **Insert→Tag**

-In the *Tag Chooser* window, select the **img** tag

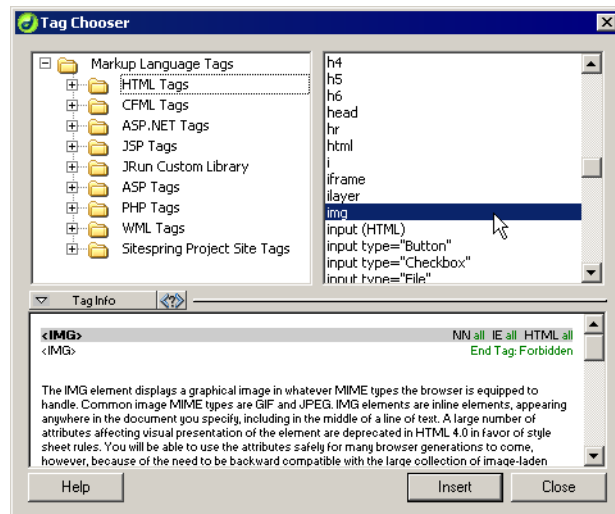


FIGURE 316

-Click **Insert**

The Tag Editor window will open—

- Set *Source* to: **beyond_the_box/assets/orange/logo.gif**
- Set *Alternate Text* to: **Foundations by PVII- The Future is Now**
- The *Width* and *Height* should be entered for you (if not, click the little calculator icon)
- Clear the *Border* field of the zero that Dreamweaver entered¹

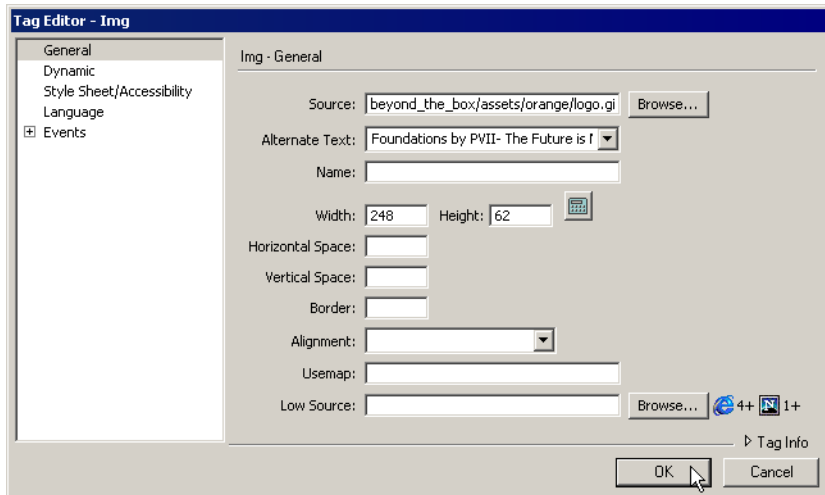


FIGURE 317

- Click **OK**
- Close** the Tag Chooser window

The logo DIV should now look like this:

```
<div id="divlogo">

</div>
```

Let's move on to **divmain...**

¹ If you later make the logo a hyperlink, Dreamweaver will insert a zero border value. If the image is not a hyperlink the border property is not necessary.

THE MAIN DIV

This DIV is a bit different. We will be nesting **divfooter** inside **divmain** to allow us more freedom to style each DIV uniquely. Since *divmain* will be absolutely positioned, and we want the footer content to appear directly below the main content, making *divfooter* a **child**¹ of *divmain* is the most logical solution. That's why we have two consecutive opening `<div>` tags followed by two consecutive closing `</div>` tags.

```
<div id="divmain">
```

```
<div id="divfooter">
```

```
</div>
```

```
</div>
```

The first ending `</div>` tag closes **divfooter**, while the second ending `</div>` tag closes its parent: **divmain**.

-Place your cursor on the blank line between the opening `<divmain>` and opening `<divfooter>` tags.

-Enter the following heading:

```
<h1>This is the Main Div</h1>
```

With **Auto Tag Completion**² on, once you enter `<h1>` Dreamweaver will insert the closing `</h1>` tag and place your cursor between the tags ready for you to enter your text.

-Make a new line (place your cursor just after the closing `</h1>` tag and press **Enter** once)

-Enter a paragraph `<p>` tag

```
<p>|</p>
```

-Type a few sentences. *Do it yourself* to get a feel for working in **Code View** (if it's too much to bear, you can *cheat* and use the filler text Snippet)

-When you've finished writing your paragraph, move your cursor just to the right of the closing `</p>` tag and press **Enter** once to create a new line

-Enter a second paragraph `<p>` tag and write a second paragraph

¹ If, instead of nesting, we stacked `<divfooter>` directly below `<divmain>`, then `<divfooter>` would become a child of the `<body>` and would appear at the top of the page—regardless of where its markup is. But `<divmain>` is going to be absolutely positioned. Can't we also make `<divfooter>` absolutely positioned and place it just below `<divmain>`? Unfortunately, there is no way to predict how tall `<divmain>` will be. Because it contains text, its height will change depending on how wide the browser window is or how large (or small) the text size is.

² If *Auto Tag Completion* is not working for you, it may be turned off. To enable it, choose **Edit→Preferences**, then choose the *Code Hints* Category. *Auto Tag Completion* is controlled by a checkbox at the top of the window.

Here's how our code should look (substituting your text, of course):

```
<div id="divmain">
<h1>This is the Main Div</h1>
<p>A re-engineering of your current world view will reaffirm your online presence
enabling a more effervescent solution. Upscaling voluminous networking exchange
solutions will result in your achieving an excellent systemic electronic data
interchange synchronization, thereby exploiting technical environments for mission
critical broad-based capacity-constrained systems. This will fundamentally cause a
morphing into a well designed and actionable information infrastructure whose
semantic content is downright null. To more fully clarify the current concept, a few
aggregate issues will require addressing to facilitate a distributed communication
venue. In integrating non-aligned structures into existing legacy systems, an even
more effervescent (bubbly) gateway blueprint in a backward compatible package of
tangible and immeasurable strategic value will result in right-sizing the conceptual
frameworks, but only when thinking outside the box. This being said, the ownership
issues inherent in dominant thematic implementations cannot be understated vis-a vis
David Bowie, for instance.</p>
<p>A re-engineering of your current world view will reaffirm your online presence
enabling a more effervescent solution. Upscaling voluminous networking exchange
solutions will result in your achieving an excellent systemic electronic data
interchange synchronization, thereby exploiting technical environments for mission
critical broad-based capacity-constrained systems. This will fundamentally cause a
morphing into a well designed and actionable information infrastructure whose
semantic content is downright null. To more fully clarify the current concept, a few
aggregate issues will require addressing to facilitate a distributed communication
venue. In integrating non-aligned structures into existing legacy systems, an even
more effervescent (bubbly) gateway blueprint in a backward compatible package of
tangible and immeasurable strategic value will result in right-sizing the conceptual
frameworks, but only when thinking outside the box. This being said, the ownership
issues inherent in dominant thematic implementations cannot be understated vis-a vis
David Bowie, for instance.</p>
<div id="divfooter">

</div>
</div>
```

Now let's enter the footer content...

<divfooter> currently looks like this:

```
<div id="divfooter">  
  
</div>  
</div>
```

-Place your cursor on the blank line between the opening and closing <div> tags and enter the following code:

```
<address>  
<strong>Foundations Corp.</strong><br>  
2222 Mockingbird Lane | Ravenswood, OH U.S.A. 44444<br>  
1-800-555-1212 | <a href="#">General E-Mail</a><br>  
Copyright (c) 2002 Foundations LLC. All Rights Reserved.  
</address>
```

The tags serve two purposes:

- 1.) Most browsers render the text between them bold
- 2.) Assistive readers *speak* the text in a louder or deeper voice.

We use
 tags so that each part of our address content appears on a separate line with no apparent margins above or below.

When you enter the opening <address> tag, Dreamweaver will complete it by writing the closing </address> tag right next to it—with your cursor positioned between them. To make your code look like ours, press **Enter** twice (after the opening **address** tag) to create a blank line between the opening and closing <address> tags:

```
<address>  
Content goes here  
</address>
```

Should you feel the urge to slip back into **Design View** to build your page, please resist. If you build the document (and add its initial content) using Dreamweaver's powerful code editing capabilities, you will have an intimate understanding of just how your document is structured. Use Design View to edit content *after* your page is structured and tested.

Note: As we build up the structure and add our initial content, you'll notice we are assigning IDs and Classes as we go. Upon completing the structure, we'll do a code check and assemble all of the IDs and Classes into a list—which will be used to identify the selectors that will comprise our style sheet.

-Switch to **Design View** to see what our evolving document structure looks like

THE EVOLVING STRUCTURE

In Design View you can see the raw structure of our new page. Our document embodies pure *Structured Information*. We can now style it any way we please, without changing its meaning, structure, or its HTML markup. The changes we make can impact an entire Web site—thousands of pages—*without having to edit the document itself*. This is the essence of CSS power.

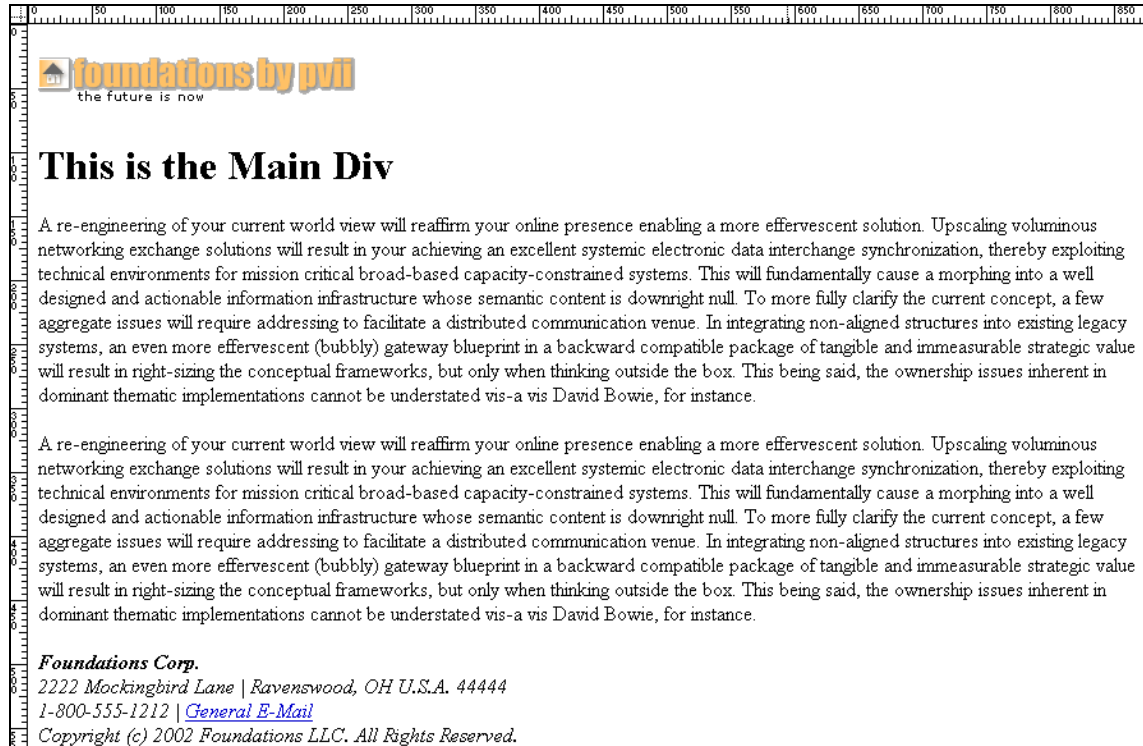


FIGURE 318

Let's add the navbar now.

-Switch back to **Code View**

THE NAVBAR DIV

<divnavbar> currently looks like this:

```
<div id="divnavbar">

<div id="divnews">

</div>
</div>
```

We've nested <divnews> inside <divnavbar> (just as we did with *divmain* and *divfooter*) because we want **news** content to appear *directly below the navigation links* and we want it to be styled uniquely.

-Place your cursor on the blank line between the opening <divnavbar> and opening <divnews> tags and enter the following code:

```
<div id="divnavbar">
<p id="skipnav"><a href="#skip" title="Skip Navigation Links (Alt+1)" accesskey="1">Skip
Links</a></p>
<p class="navbutts"><a href="#" tabindex="1">Main Section One</a></p>
<p class="navsubs"><a href="#" tabindex="2">Sublink One</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink Two</a></p>
<p class="navsubs"><a href="#" tabindex="4">Sublink Three</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="5">Sublink Four</a></p>
<p class="navbutts"><a href="#" tabindex="6">Main Section Two</a></p>
<p class="navbutts"><a href="#" tabindex="7">Main Section Three</a></p>
<p class="navbutts"><a href="#" tabindex="8">Main Section Four</a></p>
<div id="divnews">

</div>
</div>
```

What in blazes is **skipnav** all about? Let's put our accessibility hat on for a moment. Imagine you are blind and you are using assistive software that *reads* the page to you aloud. When you first reach the site, the content is read to you based on the order of the source code. The logo image alternate text is read, then the main content, then the footer, then each link in the navbar, and finally—the news. That's cool for the first time. But as you load a new page, you may find it annoying to have to listen to all of the links again. The skip link allows you to optionally follow a link that will cause the page to scroll to a *named anchor*: **skip**, which is conveniently located where the news content begins (allowing users to bypass the navigation links). Let's make the *named anchor* now, before we add the news content.

-On the blank line just below the opening <divnews> tag, enter this code:

```
<div id="divnews">
<a name="skip"></a>
</div>
</div>
```

When we create our style sheet, we'll make a **#skipnav** selector that hides the entire paragraph from visual browsers.

Now let's add our news content...

Our news DIV currently looks like this:

```
<div id="divnews">
<a name="skip"></a>
</div>
</div>
```

-Place your cursor just to the right of the closing `</a>` tag and press **Enter** once to create a new line

```
<div id="divnews">
<a name="skip"></a>

</div>
</div>
```

-Enter the following code, beginning on the new line you just created (*substituting your own text*):

```
<div id="divnews">
<a name="skip"></a>
<h2>And So it Goes...</h2>
<p>To more fully clarify the current exchange, a few aggregate issues will require
addressing to facilitate our distributed communication venue.</p>
<p>In developing skewed structures into existing legacy systems, a holistic gateway
blueprint is conceptually required irregardless of one's compulsion to think outside
the box, as conventional wisdom has it.</p>
</div>
</div>
```

That's it for our structure and initial content.

Before we move on to the tasks of styling and layout, let's use Dreamweaver's built-in Accessibility Checker to see if our page would meet **Section 508** and rules and WAI guidelines.

WAI is the W3C *Web Accessibility Initiative*. Go here to read about it:

<http://www.w3.org/WAI/gettingstarted>

Section 508 is a U.S. government-sponsored initiative, whose relevancy goes beyond America. Go here to learn more about it:

<http://www.section508.gov>

Usablenet is an independent company that makes accessibility tools for Web developers. In fact, they made the Accessibility Checker that comes with Dreamweaver MX:

<http://www.usablenet.com>

ACCESSIBILITY CHECK

-Save the page (Ctrl+S)

-Choose **File**→**Check Page**→**Check Accessibility**

The *Results panel* will open.

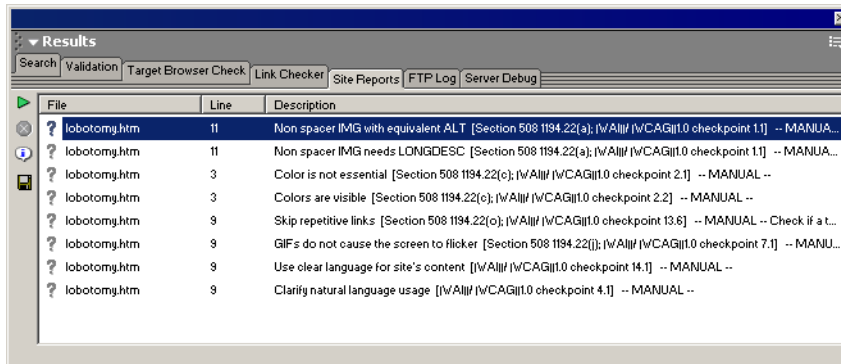


FIGURE 319- QUESTION MARKS ARE GOOD!

The page passes, but we need to review several issues in the context of the document because programmed checkers cannot read our mind or understand the meaning of the written word. **According to Dreamweaver's Accessibility checker**, our page passes muster and meets **WAI** and **Section 508** requirements because we can *reason away* all the issues raised by the *question marks*:

1. Logo.gif does have concise **Alternate Text**
2. Logo.gif is not so complex as to require **LONGDESC**
3. The information conveyed in our page **does not rely on Color**
4. At this point **color contrast** between foreground and background is sufficient to differentiate them
5. We have an anchor to **skip repetitive links**
6. There are no animated GIFs, and no **screen flicker** issues
7. Our **language** will be **clear** and **concise** once you enter *real* content ☺
8. We are using a *single natural language*: English

Note: Clicking the More Info icon ⓘ on the left edge of the Results panel will open the Reference panel and present you with additional information on the selected accessibility issue.

-Close the *Results panel*

You can run the Accessibility Checker at any time. Just remember that all Accessibility reports produce *Question Marks*, which merely indicate issues you need to reason, rationalize, or defend. Should you see an **X** instead of a **?**, that would indicate an issue that the checker knows is wrong and that *must* be fixed.

CODE CHECK

Here is what our lobotomy.htm <body> markup should look like at this point:

```
<body>
<div id="divlogo">

</div>
<div id="divmain">
<h1>This is the Main Div</h1>
<p>A re-engineering of your current world view will reaffirm your online presence enabling a
more effervescent solution. Upscaling voluminous networking exchange solutions will result in
your achieving an excellent systemic electronic data interchange synchronization, thereby
exploiting technical environments for mission critical broad-based capacity-constrained systems.
This will fundamentally cause a morphing into a well designed and actionable information
infrastructure whose semantic content is downright null. To more fully clarify the current
concept, a few aggregate issues will require addressing to facilitate a distributed
communication venue. In integrating non-aligned structures into existing legacy systems, an even
more effervescent (bubbly) gateway blueprint in a backward compatible package of tangible and
immeasurable strategic value will result in right-sizing the conceptual frameworks, but only
when thinking outside the box. This being said, the ownership issues inherent in dominant
thematic implementations cannot be understated vis-a-vis David Bowie, for instance.</p>
<p>A re-engineering of your current world view will reaffirm your online presence enabling a
more effervescent solution. Upscaling voluminous networking exchange solutions will result in
your achieving an excellent systemic electronic data interchange synchronization, thereby
exploiting technical environments for mission critical broad-based capacity-constrained systems.
This will fundamentally cause a morphing into a well designed and actionable information
infrastructure whose semantic content is downright null. To more fully clarify the current
concept, a few aggregate issues will require addressing to facilitate a distributed
communication venue. In integrating non-aligned structures into existing legacy systems, an even
more effervescent (bubbly) gateway blueprint in a backward compatible package of tangible and
immeasurable strategic value will result in right-sizing the conceptual frameworks, but only
when thinking outside the box. This being said, the ownership issues inherent in dominant
thematic implementations cannot be understated vis-a-vis David Bowie, for instance.</p>
<div id="divfooter">
<address>
<strong>Foundations Corp.</strong><br>
2222 Mockingbird Lane | Ravenswood, OH U.S.A. 44444<br>
1-800-555-1212 | <a href="#">General E-Mail</a><br>
Copyright (c) 2002 Foundations LLC. All Rights Reserved
</address>
</div>
</div>
<div id="divnavbar">
<p id="skipnav"><a href="#skip" title="Skip Navigation Links (Alt+1)" accesskey="1">Skip
Links</a></p>
<p class="navbutts"><a href="#" tabindex="1">Main Section One</a></p>
<p class="navsubs"><a href="#" tabindex="2">Sublink One</a></p>
<p class="navsubs"><a href="#" tabindex="3">Sublink Two</a></p>
<p class="navsubs"><a href="#" tabindex="4">Sublink Three</a></p>
<p class="navsubs" id="nbutspacer"><a href="#" tabindex="5">Sublink Four</a></p>
<p class="navbutts"><a href="#" tabindex="6">Main Section Two</a></p>
<p class="navbutts"><a href="#" tabindex="7">Main Section Three</a></p>
<p class="navbutts"><a href="#" tabindex="8">Main Section Four</a></p>
<div id="divnews">
<a name="skip"></a>
<h2>And So it Goes...</h2>
<p>To more fully clarify the current exchange, a few aggregate issues will require addressing to
facilitate our distributed communication venue.</p>
<p>In developing skewed structures into existing legacy systems, a holistic gateway blueprint is
conceptually required irregardless of one's compulsion to think outside the box, as conventional
wisdom has it.</p>
</div>
</div>
</body>
```

The markup we've shown in red, bold type indicates **CSS Selectors** that we'll need to create: **body**, **#divlogo**, **#divmain**, **h1**, **#divfooter**, **address**, **#divnavbar**, **#skipnav**, **navbutts**, **navsubs**, **#nbutspacer**, **#divnews**, and **h2**.

WRITING A STYLE SHEET FOR VERSION 4 BROWSERS

The first order of business is to create a new style sheet and link it to our page. For this exercise, we'll be creating the version 4 style sheet first.

CREATE THE VERSION 4 STYLE SHEET AND THE BODY SELECTOR

- Open **lobotomy.htm** if it's not already open
- Click the **New CSS Style** icon on the bottom of the CSS panel

The New CSS Style dialog will open. Set the following options:

- Type*: **Redefine HTML Tag**
- Tag*: **body**
- Define In*: **(New Style Sheet File)**

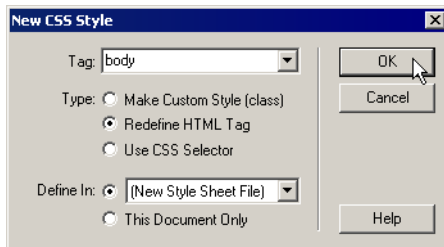


FIGURE 320

- Click **OK**

The Save Style Sheet File As dialog will open.

- Enter *File Name*: **mfv4_lobo.css**

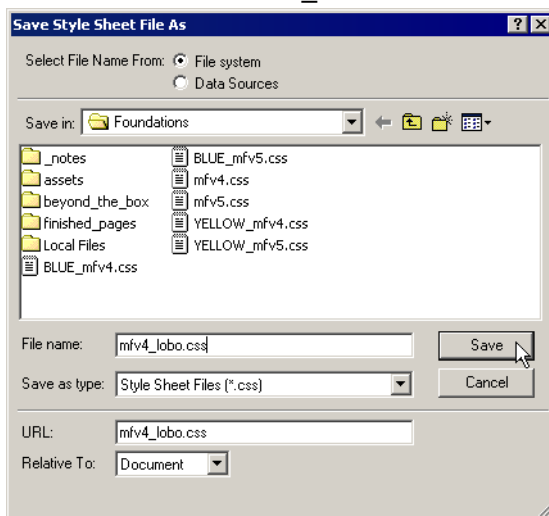


FIGURE 321

- Click **Save**

The CSS Style Definition window will open

-Switch to the *Background Category* and set the following properties:

-*Background Color: #BBBBBB*

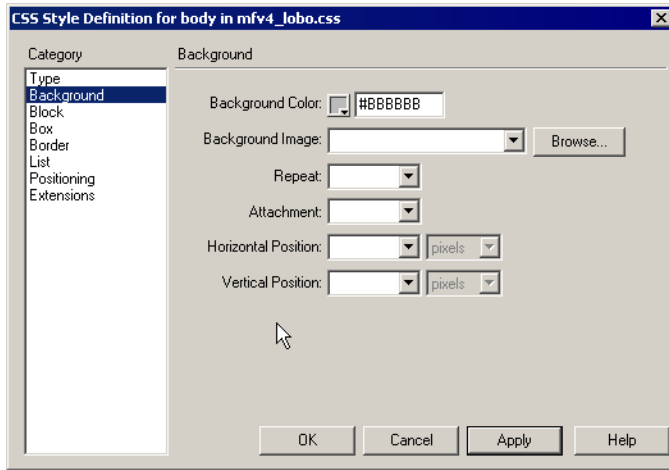


FIGURE 322

-Switch to the *Box Category*

-Make sure both *Same for All* boxes are checked, then enter *Top: 0* in both the **Padding** and **Margin** sections

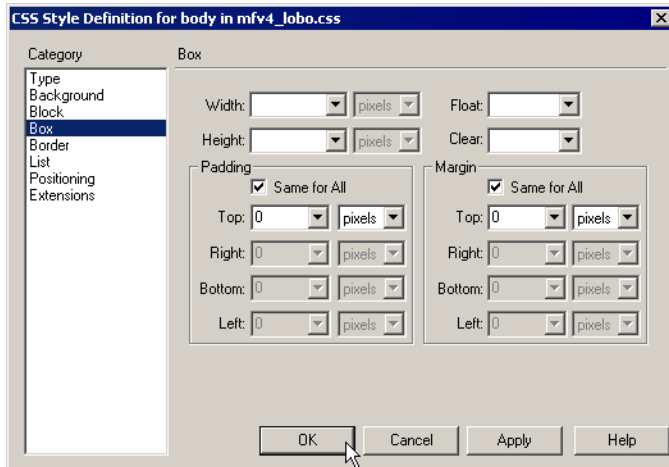


FIGURE 323

-Click **OK**

From this point on, we'll develop our style sheet in manual mode.

IDENTIFY AND CREATE THE REMAINING SELECTORS

Just before we created the version 4 style sheet, we swept through our source code and assembled a list of required CSS selectors. Let's go through them, discuss them, write them, and modify the list if more selectors are needed.

-Open **mfv4_lobo.css**

If you don't see mfv4_lobo.css in the Site panel, refresh the panel (F5)

BODY

We already made the body selector when we created our style sheet. It looks like this:

```
body {  
    background-color: #BBBBBB;  
    margin: 0px;  
    padding: 0px;  
}
```

SETTING OUR PRIMARY FONT USING A COMBINED SELECTOR

Experience tells us that declaring a *font-family* on the body selector will not work well in Navigator 4, so let's create a combined selector to address the specific tags that comprise our page's structure.

-Make a new line below the body selector and write this new rule:

```
div, p, address {  
    font-family: Verdana, Arial, Helvetica, sans-serif;  
}
```

#DIVNAVBAR

Now we get to play *building blocks* ☺. We'll untether the navigation bar from the page flow by making it absolutely positioned. Then we'll position it 24 pixels in from the left window edge, 80 pixels down from the top (leaving room for our logo), and we'll make it 190 pixels wide.

-Make this new rule:

```
#divnavbar {  
    position: absolute;  
    left: 24px;  
    top: 80px;  
    width: 190px;  
}
```

#DIVMAIN

Let's write this rule so that our main content is positioned 24 pixels to the right of the navigation bar. We can resolve this mathematically by adding *#divnavbar*'s left and width properties together then adding 24 to that: $(24 + 190) + 24 = 238$. We'll use the same top position as we did for *#divnavbar*. We'll set width to 500 pixels because relative width DIVs can behave erratically in version 4 browsers. If we add the left position (238) to the width (500) we arrive at an overall page width of 738. You can adjust these values as you see fit. We'll also make the main content area's background color white, and we'll assign a 6 pixel orange border.

-Write this rule:

```
#divmain {
    position: absolute;
    left: 238px;
    top: 80px;
    width: 500px;
    background-color: #FFFFFF;
    border: 6px solid #FFC166;
}
```

#DIVLOGO

Let's write this rule so that our logo is positioned 16 pixels from the top window edge and 238 pixels in from the left edge (aligning vertically with the main content).

-Write this rule:

```
#divlogo {
    position: absolute;
    left: 238px;
    top: 16px;
    width: 248px;
}
```

-**Save and Close** *mfv4_lobo.css*

Let's preview **lobotomy.htm** in Navigator 4 to see what we've built—and to identify any problems or issues.

How are you doing?

I know that writing a style sheet manually for the first time can be tedious—but if you resist the urge to use Dreamweaver's visual CSS Editor this one time, your investment will pay back huge dividends. The more you practice, the more familiar you will become with CSS structure and syntax.

-Switch to **lobotomy.htm** in *Design View*

-**Preview** in *Navigator 4*

Well, I'll be dipped in Belgian chocolate¹—we have taken our page divisions and arranged them like building blocks!

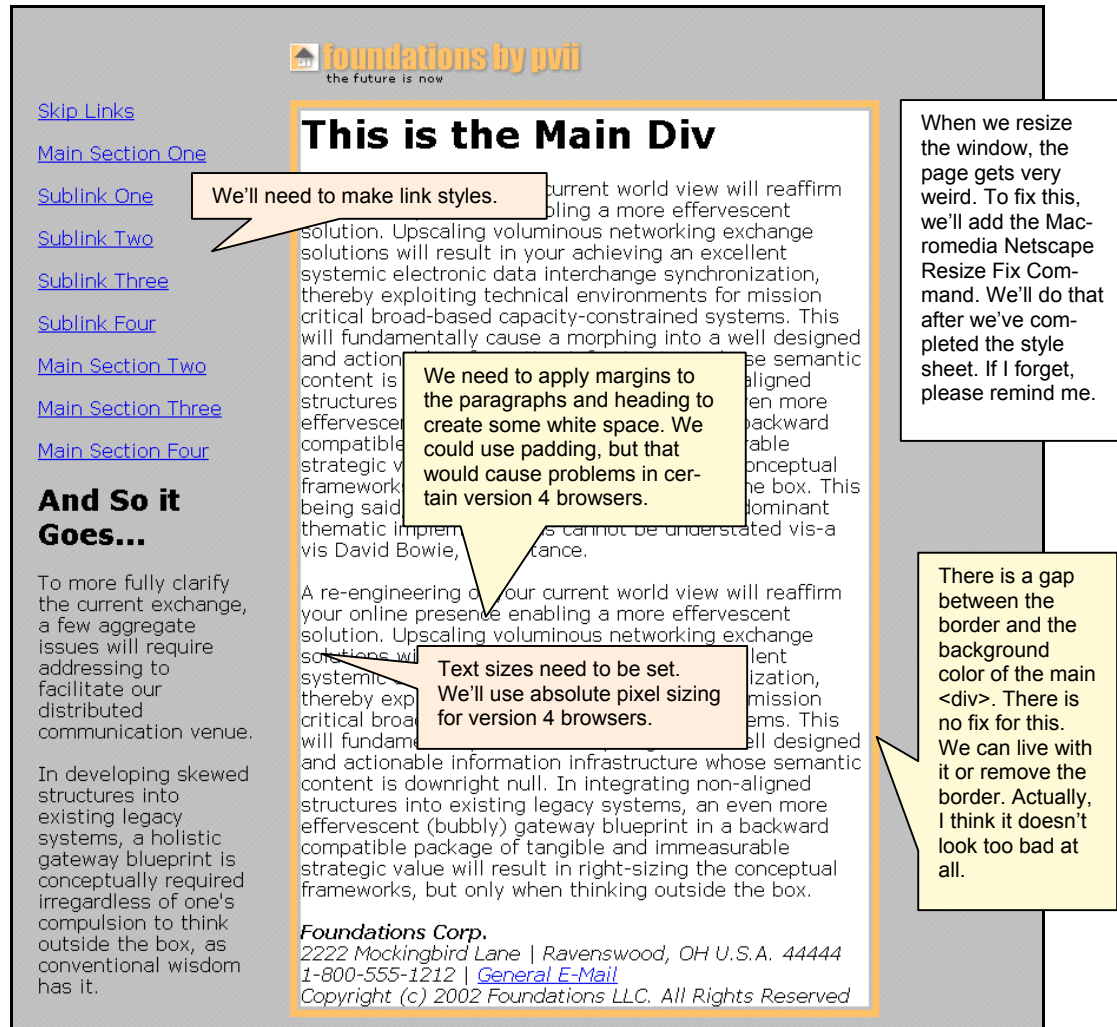


FIGURE 324

Warning: Dreamweaver has read our style sheet and now considers #divnavbar, #divmain, and #divlogo to be *Layers*. Dreamweaver doesn't know its dealing with seasoned coders so it will allow you to drag those *Layers* around and even to resize them. This would not be healthy for our style sheet because Dreamweaver will try to help us by rewriting our CSS rules—even if the CSS file is closed. Dreamweaver may also write inline style properties on your <div> tag. For these reasons, **please do not move or resize these DIVs in Dreamweaver's Design View window.**

-Open **mfv4_lobo.css**

¹ While I'd love to be dipped in Belgian chocolate, that is beyond the scope of this lesson.

#DIVMAIN P

Let's give all the paragraphs inside #divmain margin values to establish some white space inside the box. We'll also set font size.

-Write this rule:

```
#divmain p {  
    margin: 12px 30px;  
    font-size: 14px;  
}
```

#DIVMAIN H1

The heading inside #divmain will need some margin. We'll use shorthand to declare 30 pixels top, 30 pixels left and right, and 0 pixels bottom. We'll also set font size and color (to the same shade of orange we used for the #divmain border).

-Write this rule:

```
#divmain h1 {  
    margin: 30px 30px 0px;  
    color: #FFC166;  
    font-size: 20px;  
}
```

#DIVFOOTER

We'll set font size to 12px and color to medium gray.

-Write this rule:

```
#divfooter {  
    color: #999999;  
    font-size: 12px;  
}
```

ADDRESS

We'll set font style to normal (most browsers render <address> text in italics) and we'll set margins.

-Write this rule:

```
address {  
    font-style: normal;  
    margin: 12px 30px 36px;  
}
```


Now is a good time to preview the page to see how our rules are being rendered.

-Save and Close **mfv4_lobo.css**

-Switch to **lobotomy.htm** and **preview** in *Navigator 4*

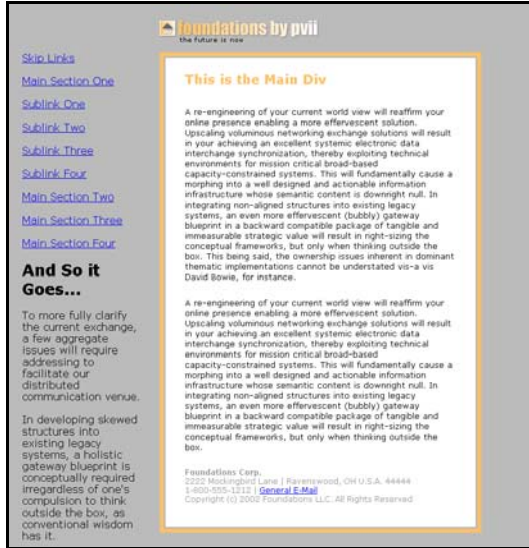


FIGURE 325

Let's style #divmain's <a> tag. The General E-Mail link looks a bit too blue.

-Open **mfv4_lobo.css**

#DIVMAIN A

Let's color our links orange and make them bold.

-Write this rule:

```
#divmain a {
    color: #FFC166;
    font-weight: bold;
}
```

#DIVMAIN A:HOVER, #DIVMAIN A:ACTIVE

-Write this rule:

```
#divmain a:hover, #divmain a:active {
    color: #000000;
    text-decoration: none;
}
```

Remember—the hover and active states are only recognized by MSIE4. Navigator 4 will ignore them.

#SKIPNAV

The skip link needs to be hidden. Let's create a simple rule using the display property.

-Write this rule:

```
#skipnav {  
    display: none;  
}
```

Note: Dreamweaver MX does not support the display property in Design View, so don't be concerned that the skip link is still on your page. It will be quite invisible in all version 4 and higher browsers.

P.NAVBUTTS

This rule will establish the margins between each of the main section links. We'll also set font size to 16px.

-Write this rule:

```
p.navbutts {  
    font-size: 16px;  
    margin: 0px 0px 3px;  
}
```

.NAVBUTTS A

Let's style the <a> tag.

-Write this rule:

```
.navbutts a {  
    text-decoration: underline;  
    color: #000000;  
}
```

.NAVBUTTS A:HOVER, .NAVBUTTS A:ACTIVE

-Write this rule:

```
.navbutts a:hover, .navbutts a:active {  
    background-color: #FFC166;  
    color: #333333;  
    text-decoration: none;  
}
```

P.NAVSUBS

This rule will establish the margins between each of the sub-links and will indent the sub-links in from the main section headings. We'll also set font size to 12px.

-Write this rule:

```
p.navsubs {
    font-size: 12px;
    margin: 0px 0px 2px 16px;
}
```

NAVSUBS A

Let's style the <a> tag.

-Write this rule:

```
.navsubs a {
    color: #FFFFFF;
}
```

.NAVSUBS A:HOVER, .NAVSUBS A:ACTIVE

-Write this rule:

```
.navsubs a:hover, .navsubs a:active {
    background-color: #FEDDAB;
    color: #BBBBBB;
    text-decoration: none;
}
```

#NBUTSPACER

We can't forget #nbutspacer. It creates space between the last sub-link and the main section heading directly below it.

-Write this rule:

```
#nbutspacer {
    margin-bottom: 16px;
}
```

Now is a good time to check in with Navigator 4.

-Save and Close **mfv4_lobo.css**

-Switch to **lobotomy.htm**

-Preview in Navigator 4



FIGURE 326

We're almost done. Let's tackle #divnews.

#DIVNEWS

We're going to style the paragraphs inside #divnews. We'll set font size, color, and margins.

-Write the rule:

```
#divnews p {
    color: #000000;
    font-size: 12px;
    margin: 12px 0px 0px;
}
```

#DIVNEWS H2

We're setting font size, color, and margins.

-Write the rule:

```
#divnews h2 {
    color: #666666;
    font-size: 16px;
    margin: 16px 0px 0px;
}
```

The Version 4 Style Sheet is complete.

-Save and Close `mfv4_lobo.css`

-Switch to `lobotomy.htm` and *preview in Navigator 4*

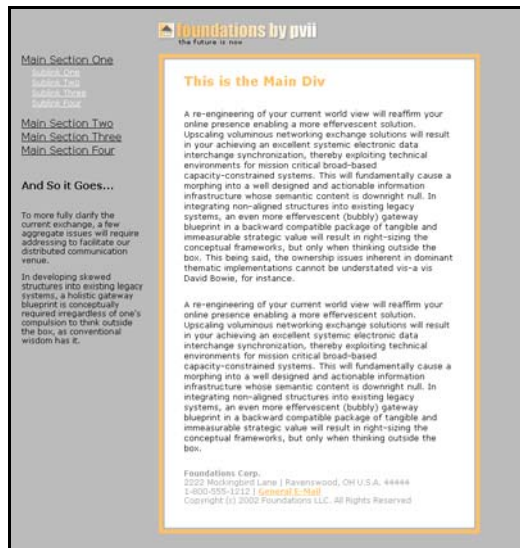


FIGURE 327

Looking pretty good, I'd say.

Let's do a code check...

CODE CHECK—MFV4_LOBO.CSS

Here's what your CSS file should look like:

```
body {
    background-color: #BBBBBB;
    margin: 0px;
    padding: 0px;
}
div, p, address {
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
#divnavbar {
    position: absolute;
    left: 24px;
    top: 80px;
    width: 190px;
}
#divmain {
    position: absolute;
    left: 238px;
    top: 80px;
    width: 500px;
    background-color: #FFFFFF;
    border: 6px solid #FFC166;
}
#divlogo {
    position: absolute;
    left: 238px;
    top: 16px;
    width: 248px;
}
#divmain p {
    margin: 12px 30px;
    font-size: 14px;
}
#divmain h1 {
    margin: 20px 30px 0px;
    color: #FFC166;
    font-size: 20px;
}
#divfooter {
    color: #999999;
    font-size: 12px;
}
address {
    font-style: normal;
    margin: 12px 30px 36px;
}
#divmain a {
    color: #FFC166;
    font-weight: bold;
}
#divmain a:hover, #divmain a:active {
    color: #000000;
    text-decoration: none;
}
#skipnav {
    display: none;
}
p.navbutts {
    font-size: 16px;
    margin: 0px 0px 3px;
}
.navbutts a {
    text-decoration: underline;
    color: #000000;
}
.navbutts a:hover, .navbutts a:active {
    background-color: #FFC166;
    color: #333333;
    text-decoration: none;
}
p.navsubs {
    font-size: 12px;
    margin: 0px 0px 2px 16px;
}
.navsubs a {
    color: #FFFFFF;
}
.navsubs a:hover, .navsubs a:active {
    background-color: #FEDDAB;
    color: #BBBBBB;
    text-decoration: none;
}
#nbutspacer {
    margin-bottom: 16px;
}
#divnews p {
    color: #000000;
    font-size: 12px;
    margin: 12px 0px 0px;
}
#divnews h2 {
    color: #000000;
    font-size: 16px;
    margin: 16px 0px 0px;
}
```

You forgot to remind me about the Netscape Resize Fix. It's a good thing *I remembered*.

APPLY THE NETSCAPE RESIZE FIX

Navigator 4's engine is so old that it sometimes forgets things. When you mix CSS, positioning, and then ask Navigator to resize its window and put everything back where it belongs—dementia sets in. As far as inanimate objects go, it's the closest thing to a seizure you'll ever see. Dreamweaver comes equipped with life support for just this situation.

-Choose **Commands**→**Add/Remove Netscape Resize Fix**

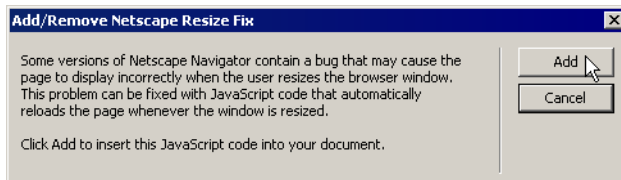


FIGURE 328

-Click **Add**

Dreamweaver will write a JavaScript in the <head> of the page. It looks like this:

```
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if ((appName=="Netscape") && (parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight; onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH) location.reload();
}
MM_reloadPage(true);
//-->
</script>
```

As we learned¹ in [Chapter 14](#), it's usually a good idea to remove JavaScripts from your page and place them into an external file, which can be linked to lots of pages. You can do it now, do it later, or you can leave the script in your page. The decision is yours to make. I trust that you will make the correct and efficient choice ☺.

Now it's time to see just how cool CSS can be when we work with a document whose presentation is cleanly separated from its structure...

¹ Actually, **you** learned. I already knew it. But I had to learn it myself prior to teaching you, so the pronoun *we* seems quite plausible, though perhaps marginally so.

CSS UNLEASHED—THE LOBOTOMY STYLE SHEET FOR VERSION 5 BROWSERS

We'll create the version 5 style sheet by making a copy of `mfv4_lobo.css`, which we'll then import into our document.

MAKING A COPY OF `MFV4_LOBO.CSS`

We'll make a copy of `mfv4_lobo.css`—then we'll rename the copy.

-Open `mfv4_lobo.css`

-Choose **File**→ **Save As**

-Enter *File name*: `mfv5_lobo.css`

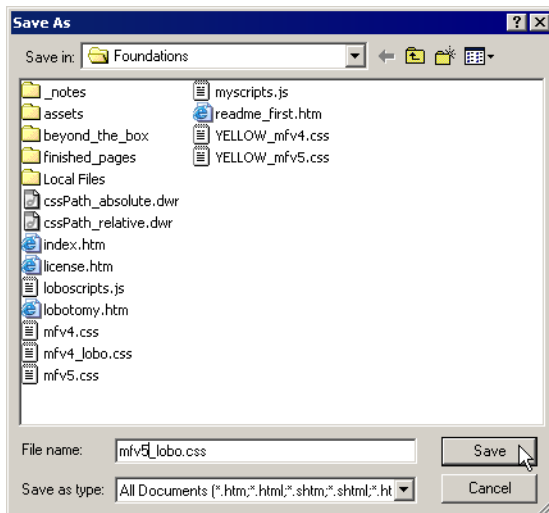


FIGURE 329

-Click **Save**

-Close `mfv5_lobo.css`

IMPORTING THE VERSION 5 STYLE SHEET

Now we need to import the version 5 style sheet into the **lobotomy.htm** document.

-In **lobotomy.htm**, click the *Attach Style Sheet* icon along the bottom of the CSS Styles panel

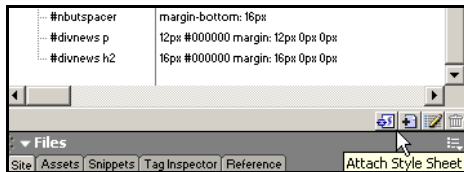


FIGURE 330

The Link External Style Sheet window will open.

-Enter *File/URL*: **mfv5_lobo.css** (or Browse to it)

-Select *Add As*: **Import**

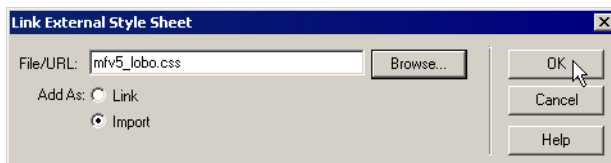


FIGURE 331

-Click **OK**

-**Save** lobotomy.htm

-**Open** *mfv5_lobo.css*

EDITING AND OPTIMIZING THE VERSION 5 STYLE SHEET

We'll review and optimize each rule. In the process, we'll add some interactivity and change the layout.

THE PLAN

Our objective is to make our version 5 layout fluid and interactive. While we're at it, we'll also move the navigation bar to the right side of the window.

On the surface, **position: absolute** may seem a lot more restrictive than it actually is. CSS allows us to ask a (modern) browser to absolutely position an element *in context to the available window size*. We can assign, for instance, a left value of 8%. That means that no matter how wide a window is, the distance between the element's left edge and the browser window's left edge will always be 8% of the window's available width. For example, if the browser window is 1600 pixels wide, the element will be 128 pixels in from the left edge. If the window is 800 pixels wide, the gap shrinks to 64 pixels. Since we can also set an element's width to a percentage value, we have a very powerful way to lay out a page.

LET'S DO THE MATH

If our goal is a fluid design with main content on the left and navigation on the right, here's what we do:

#divlogo and **#divmain** are positioned left: **8%**

#divlogo is assigned a pixel width because all it contains is an image.

#divmain is assigned width: **60%**

Because **#divmain** is 60% wide and 8% from the left edge, its right edge can be calculated to be 68% from the left window edge. However, **#divmain** has a 6-pixel border. Borders (and padding) are added to an element's width¹, so the position is actually 68%+12px= *I don't know*. This is not a big problem. We can allow for 12 extra pixels without fear of breaking our layout.

#divnavbar is assigned a left position of **72%**. That gives us 4% to take care of white space *and* the extra 12 pixels². Isn't math wonderful!

¹ MSIE5x (PC only) does it the other way around. That is, borders and padding are included *within an element's declared width*. Because we are dealing with a mere 12 pixels, the effect is minimal. MSIE5x-PC becomes a problem if you want to have two or more elements abutting border-to-border.

² In order for 4% of the window width to be 12 pixels, the window would have to be 300 pixels wide. So, our navigation <div> will not encroach on the main <div> at window widths greater than or equal to 300px.

EDITING THE VERSION 5 STYLE SHEET

Let's go down the list in order, rule by rule—editing and optimizing as we go. As we did in Chapter 9, we'll delete duplicate or redundant properties to make our style sheet as efficient as possible.

BODY & DIV, P, ADDRESS

The **body** and **div**, **p**, **address** rules require no edits. You can remove both of these redundant rules from the **mfv5_lobo** style sheet.

#DIVNAVBAR

Here is the existing rule:

```
#divnavbar {  
    position: absolute;  
    left: 24px;  
    top: 80px;  
    width: 190px;  
}
```

-Edit this rule to change its *left* and *width* values. The *position* and *top* properties are redundant¹, and can be removed

```
#divnavbar {  
    left: 72%;  
    width: 20%;  
}
```

#DIVMAIN

Here is the existing rule:

```
#divmain {  
    position: absolute;  
    left: 238px;  
    top: 80px;  
    width: 500px;  
    background-color: #FFFFFF;  
    border: 6px solid #FFC166;  
}
```

-Edit #divmain to change its *left* and *width* values. *Position*, *top*, *background-color*, and *border* properties are redundant.

```
#divmain {  
    left: 8%;  
    width: 60%;  
}
```

¹ All browsers read the linked (mfv4_lobo) style sheet, so unless we need to have a different value in the imported (mfv5_lobo) style sheet, we can remove duplicate properties from **mfv5_lobo**.

#DIVLOGO

Here is the existing rule:

```
#divlogo {
    position: absolute;
    left: 238px;
    top: 16px;
    width: 248px;
}
```

-Edit #divlogo to change its *left* value. *Position*, *top*, and *width* properties are redundant.

```
#divlogo {
    left: 8%;
}
```

-Save mfv5_lobo.css

PREVIEW AND BEHOLD A CSS MIRACLE

-Open **lobotomy.htm** and *preview* in a version 5 or higher browser

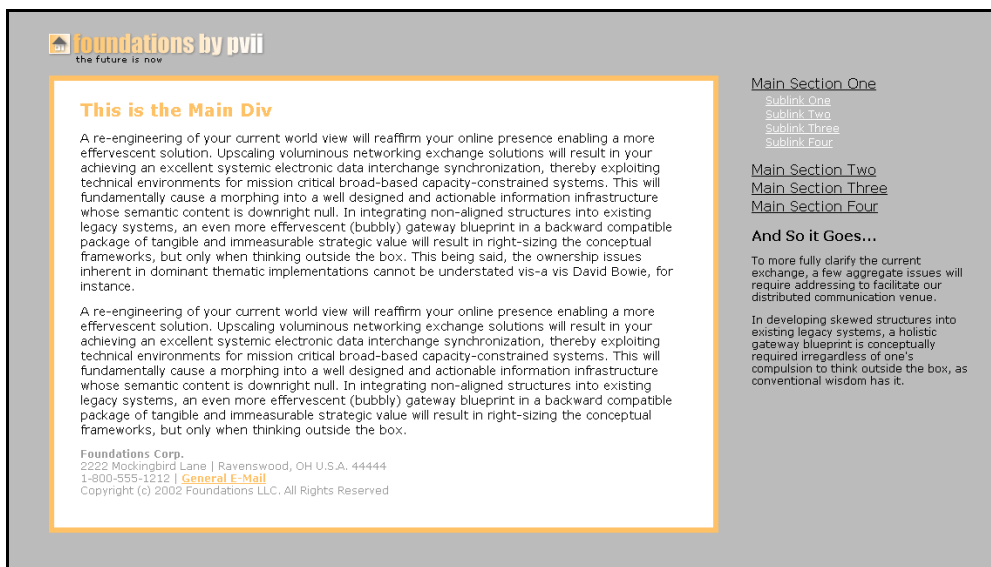


FIGURE 332- THE PREVIEWED PAGE IN A VERSION 6 BROWSER

To fully appreciate what's happening here you have to consider the fact that we have not made a single change to our HTML document—and if you preview the page in Navigator 4, it will look just like it did before.

BACK TO THE RULES

#DIVMAIN P

Here is the current rule:

```
#divmain p {  
    margin: 12px 30px;  
    font-size: 14px;  
}
```

-Edit #divmain p to change its font-size value and add line-height. The *margin* property is redundant.

```
#divmain p {  
    font-size: 90%;  
    line-height: 1.5  
}
```

#DIVMAIN H1

Here is the current rule:

```
#divmain h1 {  
    margin: 20px 30px 0px;  
    color: #FFC166;  
    font-size: 20px;  
}
```

-Edit #divmain h1 to change its font-size value and margins, and add a bottom border and bottom padding. The combination of margin and padding values creates an aesthetically pleasing amount of space above the heading, between the heading and its border, and between the border and the content below. The *color* property is redundant.

```
#divmain h1 {  
    margin: 24px 30px 12px;  
    font-size: 130%;  
    border-bottom: 1px dashed #FEC164;  
    padding-bottom: 16px;  
}
```

#DIVFOOTER

Here is the current rule:

```
#divfooter {
    color: #999999;
    font-size: 12px;
}
```

-Edit #divfooter to change color and font-size—and add background-color, padding, and border properties. Padding is used to balance offsets with respect to the footer content, background color, the content above, and the space below.

```
#divfooter {
    color: #FFFFFF;
    font-size: 80%;
    background-color: #FEC164;
    padding: 12px 30px;
    border-top: 1px solid #CFCFCF;
}
```

ADDRESS

Here is the current rule:

```
address {
    font-style: normal;
    margin: 12px 30px 36px;
}
```

-Edit address to set its margin to zero, which would otherwise conflict with the padding assigned to #divfooter. The *font-style* property is redundant.

```
address {
    margin: 0px;
}
```

#DIVMAIN A

Here is the current rule:

```
#divmain a {
    color: #FFC166;
    font-weight: bold;
}
```

-Edit #divmain a to change color—and add background-color, text-decoration, and padding properties. Padding is left/right only to make the background-color effect more balanced. The *font-weight* property is redundant.

```
#divmain a {
    color: #FFFFFF;
    background-color: #FEC164;
    text-decoration: none;
    padding: 0px 3px;
}
```

#DIVMAIN A:HOVER, #DIVMAIN A:ACTIVE

Here is the current rule:

```
#divmain a:hover, #divmain a:active {  
    color: #000000;  
    text-decoration: none;  
}
```

-Edit the rule to change its text-decoration value. The *color* property is redundant.

```
#divmain a:hover, #divmain a:active {  
    text-decoration: underline;  
}
```

#SKIPNAV

#skipnav is redundant and can be removed.

P.NAVBUTTS

Here is the current rule:

```
p.navbutts {  
    font-size: 16px;  
    margin: 0px 0px 3px;  
}
```

-Edit p.navbutts to change its font-size value. The margin property is redundant.

```
p.navbutts {  
    font-size: 90%;  
}
```

.NAVBUTTONS A

Here is the current rule:

```
.navbutts a {
    text-decoration: underline;
    color: #000000;
}
```

-Edit **.navbutts a** to change its text-decoration and color values—and to add a whole bunch more ☺. This is a *button-izing* process and uses the same techniques taught in Chapter 8. The main difference is that since all our DIVs are positioned with percentages, the width of our <a> element needs to be a percentage, too. The width property is a workaround for an MSIE-PC issue and allows that browser to make the entire <a> block *clickable*. We chose 99% to compensate for the 1px border around our links.

```
.navbutts a {
    text-decoration: none;
    color: #333333;
    display: block;
    background-color: #BBBBBB;
    width: 99%;
    padding: 2px 6px 2px 12px;
    border-top: 1px solid #D0D1CD;
    border-right: 1px solid #585B53;
    border-bottom: 1px solid #585B53;
    border-left: 1px solid #D0D1CD;
    background-image: url(beyond_the_box/assets/orange/navbg.gif);
}
```

.NAVBUTTONS A:HOVER, .NAVBUTTONS A:ACTIVE

Here is the current rule:

```
.navbutts a:hover, .navbutts a:active {
    background-color: #FFC166;
    color: #333333;
    text-decoration: none;
}
```

-Edit this rule to change color, and to turn off the background image. *Background-color* and *text-decoration* are redundant.

```
.navbutts a:hover, .navbutts a:active {
    color: #FFFFFF;
    background-image: none;
}
```


P.NAVSUBS

Here is the current rule:

```
p.navsubs {  
    font-size: 12px;  
    margin: 0px 0px 2px 16px;  
}
```

-Edit this rule to change font-size and set the left margin value to zero (we'll indent the sublinks via padding on the <a> selector):

```
p.navsubs {  
    font-size: 80%;  
    margin: 0px 0px 2px;  
}
```

.NAVSUBS A

Here is the current rule:

```
.navsubs a {  
    color: #FFFFFF;  
}
```

-Edit .navsubs a to change the color value and add the properties necessary to *button-ize* our sublinks.

```
.navsubs a {  
    color: #FAF7ED;  
    background-color: #BBBBBB;  
    text-decoration: none;  
    display: block;  
    padding: 1px 0px 1px 18px;  
    width: 100%;  
}
```

.NAVSUBS A:HOVER, .NAVSUBS A:ACTIVE

This rule is redundant and can be removed.

#NBUTSPACER

#nbutspacer is redundant and can be removed.

#DIVNEWS

Here is the current rule:

```
#divnews p {  
    color: #000000;  
    font-size: 12px;  
    margin: 12px 0px 0px;  
}
```

-Edit the rule to change font-size and add line-height. *Color* and *margin* properties are redundant.

```
#divnews p {  
    font-size: 80%;  
    line-height: 1.2;  
}
```

#DIVNEWS H2

Here is the current rule:

```
#divnews h2 {  
    color: #666666;  
    font-size: 16px;  
    margin: 16px 0px 0px;  
}
```

Edit the rule to change font-size. *Color* and *margin* properties are redundant.

```
#divnews h2 {  
    font-size: 100%;  
}
```

The version 5 style sheet is complete.

-Save and Close *mfv5_lobo.css*

CODE CHECK—MFV5_LOBO.CSS

Here's what your CSS file should look like after editing:

```
#divnavbar {
    left: 72%;
    width: 20%;
}
#divmain {
    left: 8%;
    width: 60%;
}
#divlogo {
    left: 8%;
}
#divmain p {
    font-size: 90%;
    line-height: 1.5;
}
#divmain h1 {
    margin: 24px 30px 12px;
    font-size: 130%;
    border-bottom: 1px dashed #FEC164;
    padding-bottom: 16px;
}
#divfooter {
    color: #FFFFFF;
    font-size: 80%;
    background-color: #FEC164;
    padding: 12px 30px 12px;
    border-top: 1px solid #CFCFCF;
}
address {
    margin: 0px;
}
#divmain a {
    color: #FFFFFF;
    background-color: #FEC164;
    text-decoration: none;
    padding: 0px 3px;
}
#divmain a:hover, #divmain a:active {
    text-decoration: underline;
}
p.navbutts {
    font-size: 90%;
}
.navbutts a {
    text-decoration: none;
    color: #333333;
    display: block;
    background-color: #BBBBBB;
    width: 99%;
    padding: 2px 6px 2px 12px;
    border-top: 1px solid #D0D1CD;
    border-right: 1px solid #585B53;
    border-bottom: 1px solid #585B53;
    border-left: 1px solid #D0D1CD;
    background-image: url(beyond_the_box/assets/orange/navbg.gif);
}

.navbutts a:hover, .navbutts a:active {
    color: #FFFFFF;
    background-image: none;
}
p.navsubs {
    font-size: 80%;
    margin: 0px 0px 2px;
}
.navsubs a {
    color: #FAF7ED;
    background-color: #BBBBBB;
    text-decoration: none;
    display: block;
    padding: 1px 0px 1px 18px;
    width: 100%;
}
#divnews p {
    font-size: 80%;
    line-height: 1.2;
}
#divnews h2 {
    font-size: 100%;
}
```

THE FINISHED PAGE

Let's do a comparison preview: Navigator 4 vs. Navigator 7.

-Open **lobotomy.htm**

-**Preview** in Navigator 4, then preview in Navigator 7 (or any version 5+ browser)



FIGURE 333- NAVIGATOR 4

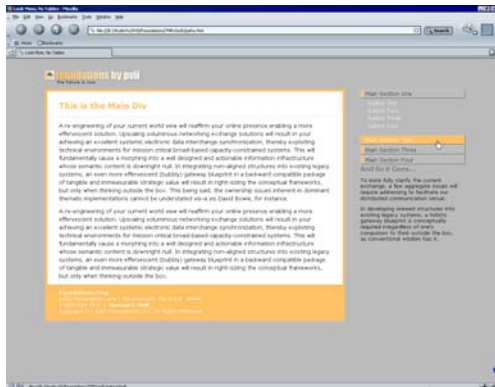


FIGURE 334- NAVIGATOR 7

The same information is presented in two different ways—optimized for the rendering and CSS capabilities of both old and new browsers. All of our styling was accomplished outside of the HTML structure, which remains pure, efficient, and accessible to all who may wish to view it.

DOES IT PRINT?

Not too well, actually. Let's look at print preview screens in both MSIE6 and Navigator 7:

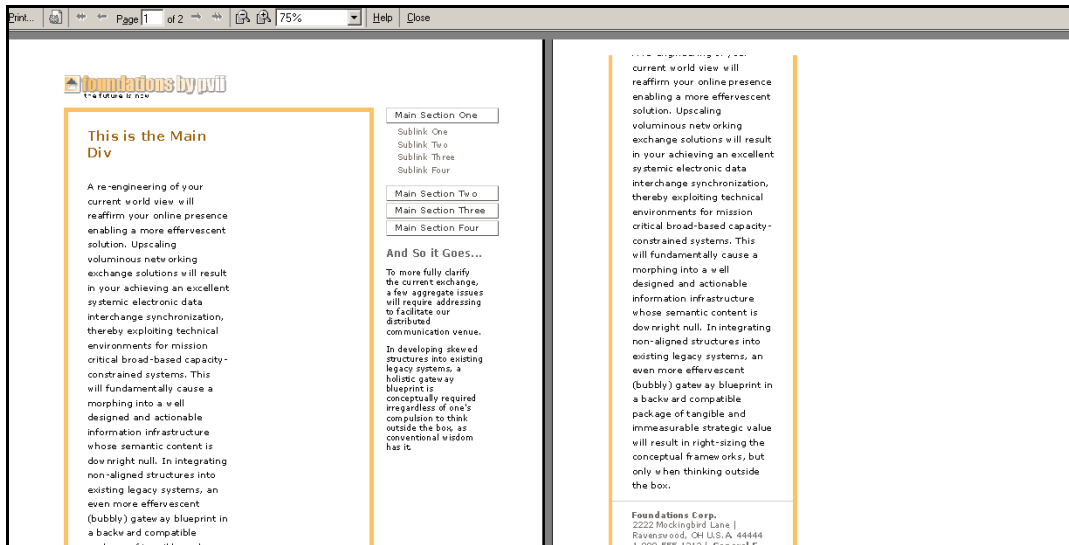


FIGURE 335- MSIE6 PRINT PREVIEW

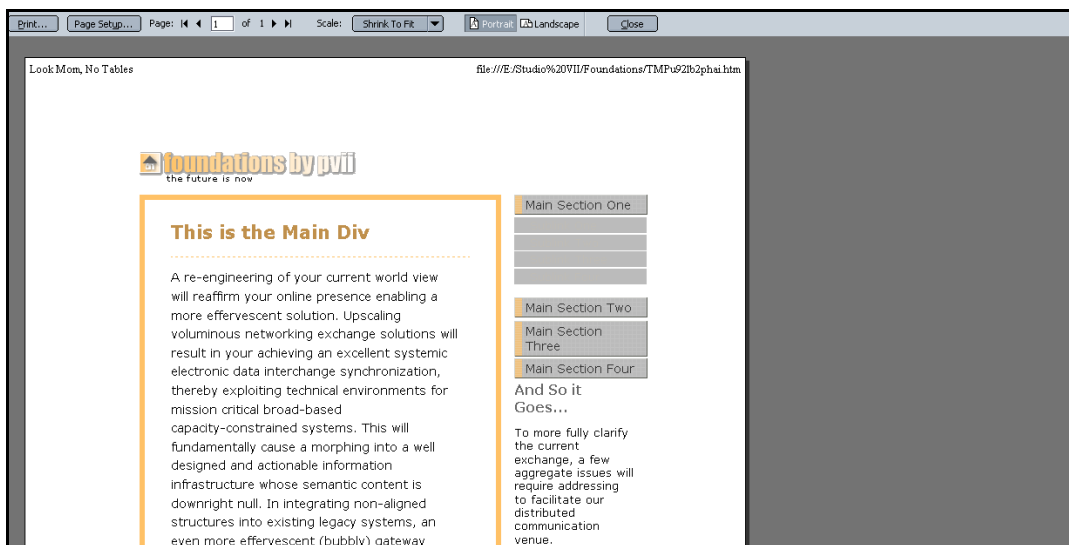


FIGURE 336- NAVIGATOR 7 PRINT PREVIEW

We have a solution...

A PRINT MEDIA STYLE SHEET TO THE RESCUE

CSS allows us to make style sheets for specific media, including *print*. There are other supported media types, such as *aural* (spoken), but they are beyond the scope of this book.

The *print media* attribute works only in version 5 and higher browsers. The rules in a print style sheet only affect printed pages. Version 4 browsers ignore the media type attribute with no ill effect.

CREATING THE PRINT STYLE SHEET

Let's make a copy of **mfv4_lobo.css** and save it with a new name (it's easier to create a new CSS file this way).

-Open **mfv4_lobo.css**

-Choose **File**→**Save As**

-Name the new file **lobo_print.css**

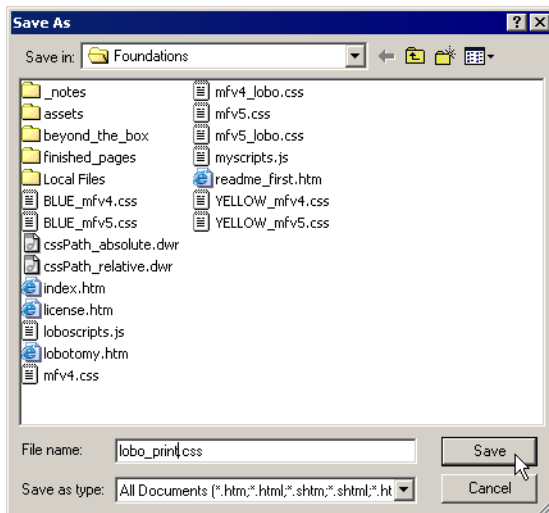


FIGURE 337

-Click **Save**

-Delete every rule in **lobo_print.css** and enter these new rules:

Let's redefine the body element to set its background to a more print-friendly white and text to 12 points. The **!important** notation means that the font-size value outweighs any other matching value declared for this element—in any style sheet or inline declaration that we, the page authors, create. It can only be superceded if the *user* sets his or her own style sheets (typically via the browser's User Preference facility) and uses a similar *!important* notation.

```
body {  
    background-color: #FFFFFF;  
    font-size: 12pt !important;  
}
```

The following combined selector uses **!important** notations to override more specific selectors in our other style sheets to ensure that all text is **black**, that there are no **margins** or **padding**, that the content stretches the *entire page width*, that *background* is always **white**, that *absolute positioning is turned off*, and that no **border** is rendered.

```
div, address, h1 {  
    color: #000000 !important;  
    margin: 0 !important;  
    padding: 0 !important;  
    width: 100% !important;  
    background-color: #FFFFFF !important;  
    position: static !important;  
    border: 0px none !important;  
}
```

The following selector nails down a specific margin for all paragraphs, level 1 headings, and level 2 headings that takes precedence over the margin settings in all other style sheets.

```
p, h1, h2 {  
    margin: 12pt 0pt !important;  
}
```

Navigator 7 seems to want to collapse the address margin unless we tell it otherwise and only if we note the value as **!important**. This rule assures that there is space below the address element.

```
address {  
    margin: 0 0 36pt 0 !important;  
}
```

The following rule prevents the logo and all of the navigation links from printing. There are no matching values in our other style sheets, so we don't need to make it **!important**.

```
#divlogo, p.navbutts, p.navsubs {  
    display: none;  
}
```

The following rule sets all hyperlinks to black and ensures that all background colors are disabled.

```
a:link, a:visited {
    color: #000000 !important;
    background-color: transparent !important;
}
```

Here is the complete lobo_print.css style sheet:

```
body {
    background-color: #FFFFFF;
    font-size: 12pt !important;
}
div, address, h1 {
    color: #000000 !important;
    margin: 0 !important;
    padding: 0 !important;
    width: 100% !important;
    background-color: #FFFFFF !important;
    position: static !important;
    border: 0px none !important;
}
p, h1, h2 {
    margin: 12pt 0pt !important;
}
address {
    margin: 0 0 36pt 0 !important;
}
#divlogo, p.navbutts, p.navsubs {
    display: none;
}
a:link, a:visited {
    color: #000000 !important;
    background-color: transparent !important;
}
```

Of course, for this style sheet to do anything, it needs to be linked to our page.

LINKING THE PRINT MEDIA STYLE SHEET

-Save and Close **lobo_print.css**

-Open or Switch to **lobotomy.htm** in Code View

-Find the imported style sheet and make a new blank line below its closing **</style>** tag

```
<style type="text/css">
<!--
@import url("mfv5_lobo.css");
-->
</style>
```

```
</head>
```

-Place your cursor on the blank line and type the link tag:

```
<link rel="stylesheet" href="lobo_print.css" type="text/css" media="print">
```

As you begin to type, Dreamweaver's Code Hints will help you along the way

Upon completing the tag, our code block should look like this:

```
<style type="text/css">
<!--
@import url("mfv5_lobo.css");
-->
</style>
<link rel="stylesheet" href="lobo_print.css" type="text/css" media="print">
</head>
```

If you switch to Design View you'll be shocked to discover that Dreamweaver is using the print media style sheet. We need to fix that by using Design Time Style Sheets.

-In Design View, **right-click** on a blank part of the page

-Choose **CSS Styles**→**Design Time Style Sheets...**

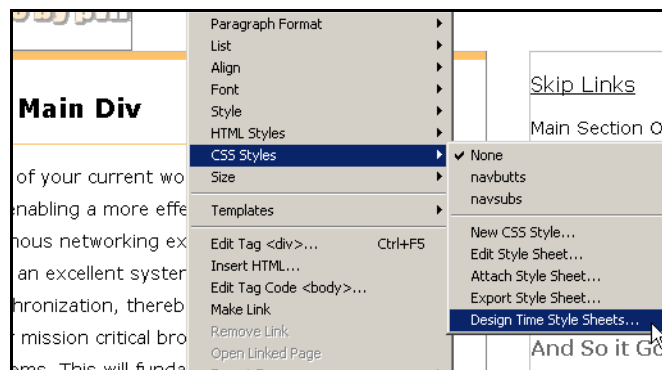


FIGURE 338

The Design Time Style Sheet window will open.

-Click the **plus (+)** sign in the *Hide at Design Time* section and choose **lobo_print.css** in the *Select File* dialog

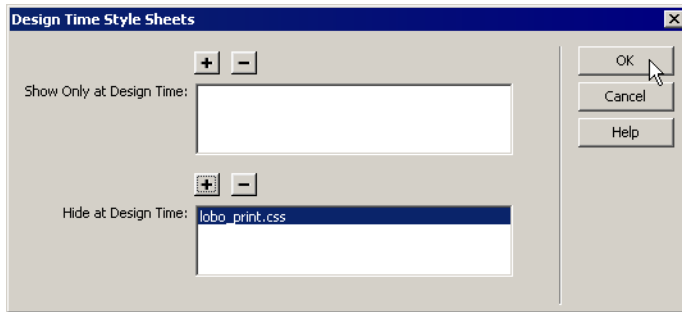


FIGURE 339

-Click **OK**

Everything will pop back into place.

Unfortunately, you'll have to do this for every new page that's linked to the print style sheet you create.

TESTING THE PRINT STYLE SHEET

Now let's check Print Preview in MSIE6 and Navigator 7 again to see if our print style sheet works:

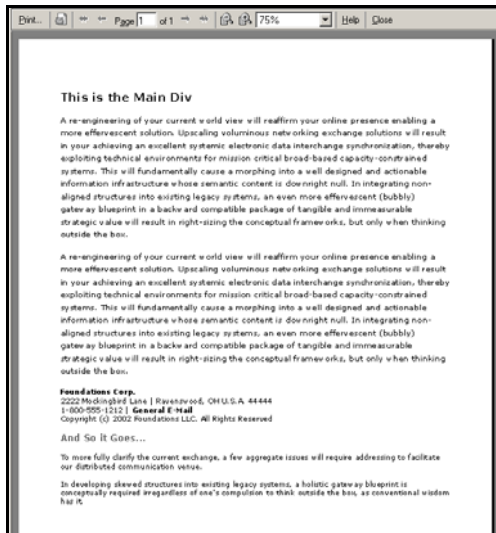


FIGURE 340- MSIE6 PRINT PREVIEW



FIGURE 341- NAVIGATOR 7 PRINT PREVIEW

That's a wrap, folks. I hope you enjoyed our table lobotomy.

Chapter 16—Afterthoughts

“AFTER ALL IS SAID AND DONE, WE DID QUITE A LOT.”—ANONYMOUS

We hope you enjoyed reading this book as much as we enjoyed writing it—which begs the question: *why did we write it?*

WE WROTE FOUNDATIONS BECAUSE—

We wrote Foundations because we knew no one else would. We didn’t want to write another cookbook of disconnected techniques nor did we want to write another reference book. We didn’t want to teach you how to use Table Layout View, Pop-up Menus, Fireworks Tables, or Timelines. Those are not foundational techniques. They don’t show how a Web page is structured—*how it really works*. Our goal was to teach you what a Web page is, how to build one, how to make it efficient, how to make it accessible, and how to use Dreamweaver to help accomplish it. We wanted to demonstrate how Web documents are made and styled—from the ground up.

We’d like to share some of the thoughts that went through our heads as we wrote. Some are obvious, others esoteric, some can only be understood in retrospect. Try to expand on them, ponder them, and draw your own conclusions.

THE WEB IS NOT A MAGAZINE.

DREAMWEAVER IS NEITHER A DESKTOP PUBLISHING TOOL NOR A WORD PROCESSING APPLICATION.

CODING TO INDUSTRY STANDARD SPECIFICATIONS TAKES OUT THE GUESSWORK.

CODING ACCESSIBLE PAGES NOT ONLY MAKES PAGES MORE ACCESSIBLE, IT ALSO MAKES PAGES MORE LOGICAL.

IT’S EASIER TO TAKE APART AND UNDERSTAND A COMPLEX MACHINE IF YOU ARE THE ONE WHO BUILT IT.

NEVER STAND STILL—SOMEONE *OR SOMETHING* MAY BE GAINING ON YOU.

ALWAYS BUILD ON A STRONG FOUNDATION

And with that, we bid you—

Adios, sala kahle, farvel, au revoir, auf wiedersehen, sayonara, addio, do svidanja—

Later, dudes.