

Creating a CSS Layout

In recent years, designing websites with CSS has become a popular trend. The advantages to designing a website with CSS are that it allows developers to create stable layouts which work in a variety of browsers; it provides separation of structure (code) from presentation (formatting text) and developers can change the look and feel of their website if desired in a short amount of time.

Designing websites with CSS can also make websites load faster on slower Internet connections, since less structural code is read by the browser and external style sheets are cached in local memory on our visitor's computer. However, websites designed with CSS can be problematic for new comers, because the mythology behind creating, organizing and coding these websites is different than with tables. Once developers have grasped and understood the correct approach and method when designing and planning your website with CSS, developers realize their website has less code and greater flexibility for future changes.

In this article, we'll learn:

- How to create our layout in Fireworks
- Slice and optimize our graphics in Fireworks
- Create and design our website using the graphics from our layout
- Create our structural mark-up in HTML
- Create our CSS which will control our layout and presentation of our website
- Understand how CSS positioning works
- Understand how equalizing columns work in CSS

If you would like to follow this article's step-by-step development or even try your hand at creating a CSS-Layout, you will find the project file links helpful.

Finished Page

http://midwestwebdesign.net/tutorials/csslayout/index_finished.html

Project Zip

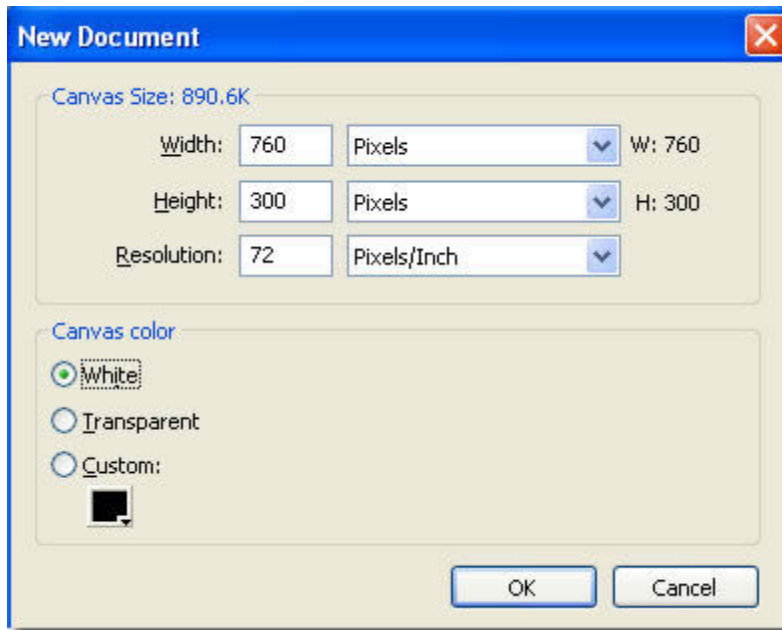
<http://midwestwebdesign.net/tutorials/csslayout/csslayout.zip>

Create our layout – Fireworks

A basic knowledge of Fireworks is needed to complete this section. If you're not comfortable with this editor, you may use Photoshop. If you're not comfortable with either, it's recommended you spend a few hours visiting either's help documentation. For our purposes, we'll use Fireworks. Open Fireworks by following these steps:

- From the Desktop, locate the start button
- Select **Programs>Macromedia>Macromedia Fireworks**

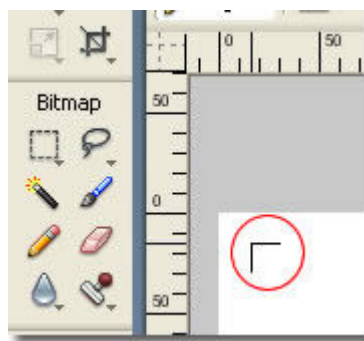
Inside Fireworks, from the main menu, select **File>New** and enter the values as shown:



Import header and footer image

We'll import the header and footer image provided from the project files download. In Fireworks, follow these steps:

- From the main menu, select **File>Import**
- Navigate to the directory which stores the header image
- Select **OK**
- Your mouse cursor turns to a carpenter's hand
- Left click near the top of the canvas



- Adjust the X-Y coordinates as follows:
 - X: 0
 - Y: 0

Follow the same steps to import our footer image, except adjust x-y coordinates as follows:

- X: 0

- Y: 200

Create the dashed line

Our dashed line will serve as a visual separator between our left and right “columns.” Follow these steps to create it:

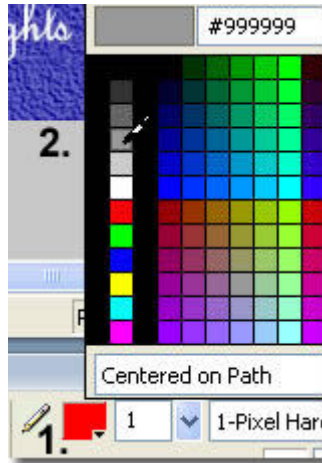
- From the main menu, select **View>Rulers**
- From the Vector tool bar, select the **line tool**



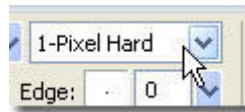
- Move your mouse cursor right below our header image to line up with 260 using the Ruler as a guide:



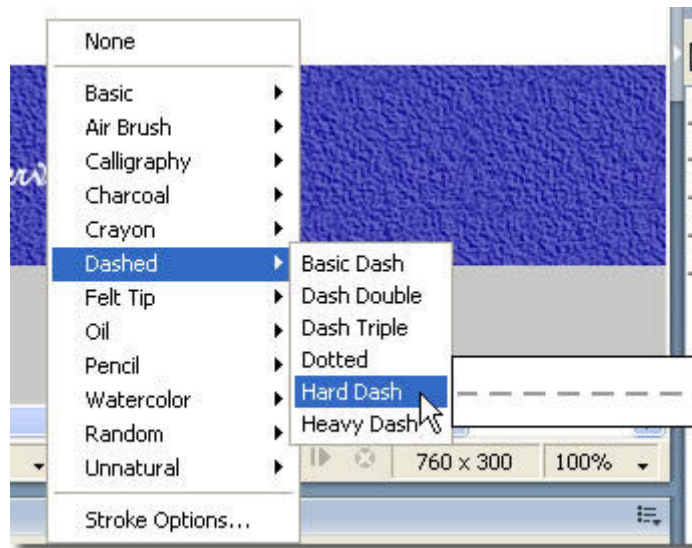
- Press and hold the shift key while dragging the line down and release
- In the properties panel, change the following settings:
 - W: 1
 - H: 100
 - X: 261
 - Y: 101
- With the object selected, change the stroke options color (1.) to #999999 (2.):



- Locate stroke category in the properties panel:



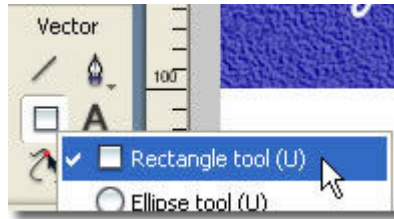
- Change the stroke category to hard dash:



Create the left column pattern

Our left column has a light gray background pattern to help visualize the difference between the two columns. Follow these steps to create it:

- From the Vector tool bar, select the rectangle tool:

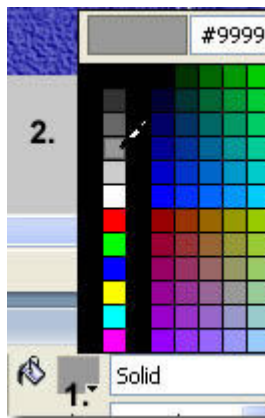


- Draw a rectangle in the vicinity of the left column
- In the properties panel, change the following settings:
 - W: 260
 - H: 100
 - X: 0
 - Y: 100
- Select light gray as a fill option (#CCCCCC)

Create the background

Our website has a subtle patterned image which repeats through our web page. Follow these steps to create it:

- From the Vector tools bar, select the rectangle tool
- Draw a rectangle in the vicinity of the right column area
- In the properties panel, change the following settings:
 - W: 38
 - H: 38
 - X: 379
 - Y: 135
- With the object selected, select fill option (1.) and enter color as dark gray (2.):



- Change texture to Mesh (1.) and set amount to 20% (2.):



Slicing our graphics

Each of our web graphics will be exported as JPEG images. With this in mind, follow these steps to set JPEG as the default export option:

- In the Optimize Panel, select **JPEG – Better Quality**:



Slice header and footer

We need to slice each of these graphics independently so that we can insert them in our web page when needed. Follow these steps to slice each one:

- Using the pointer tool, select the background object
- From the main menu, choose **Edit>Insert>Rectular Slice**
- Also, the graphic will have a green tint when sliced
 - In the properties panel, give our slice a name of **header**:



- Follow the same steps to slice our footer graphic, except give our slice a name of **footer**

Slice our page background

Follow these steps to slice our page background:

- Using the pointer tool, select the background object
- From the main menu, choose **Edit>Insert>Rectular Slice**
- In the properties panel, give our slice a name of **background**

Slice our columns

In order to create and simulate equalizing columns in CSS, we have to be creative with the use of images. In our layout, we'll create one slice which will span both our left and right columns. We'll then write a CSS rule which will repeat this graphic along the y-axis. When content is added to our web page, the graphic will repeat vertically which simulates equalizing columns. More on this concept will be discussed later when developing our web page. Follow these steps to slice our two columns:

- From the Tool bar, select the slice tool



- Drag the slice over our layout making sure it encompasses our left and right columns:



- In the properties panel, adjust the values as follows:
 - W: 760
 - H: 14
 - X: 0
 - Y: 112
- With the object selected, in the properties panel, give our slice a name of **wrapper**

Export and save our images

In Fireworks, we can export and save our images using the same method. Follow these steps:

- From the Vector tool bar, make sure slices are turned on to ensure each graphic needed is sliced:



- From the main menu, select **File>Export**
- In the Export drop down list, choose **Images only**
- Leave Include Selected slices only checked
- Navigate to your images directory Left click Export

Navigate to the directory where you have stored the project files download, your images should be present. At any time, you can modify colors, text and then export your images to your directory and they will be reflected in your website.

Save our layout

In order to make modifications to our layout in the future, we need to save it as a portable network graphic (PNG), which is Fireworks native editable format. Follow these steps:

- From the main menu, select **File>Save**
- Navigate to your directory where project files are stored
- In the File name text field, type **ryan_layout**
- Left click **Save**

Once completed, close Fireworks, we're done with it.

Analyzing our layout

From creating our Fireworks document, we arrive at the following structural requirements for our web page:

- Header
- Left column
- Right Column
- Footer

Our header will identify our website. Additionally, our left column will have an unordered list styled with CSS to simulate buttons and the right column will hold our content. Our footer will contain copyright information for our website.

Open our web page

Open index.html in your text editor and observe the following structure:

```
<div id="wrapper">
<div id="header">
</div>
<div id="leftcontent">
</div>
<div id="rightcontent">
</div>
<div id="footer">
</div>
</div>
```

Looking at our mark-up each DIV should be clear as to its purpose. However, our beginning DIV, wrapper, has a different purpose. Remember from our Fireworks layout we need a way to simulate equalizing columns. When we sliced our two columns, we used one slice named wrapper which encompassed both columns. When we write our CSS rule, we repeat this image along the y-axis and as content is added; our image will expand or contract vertically to simulate equalizing columns. This is how one image can span two distinct areas of a layout and simulate equalizing columns using CSS.

Finally, we have provided the basic structure for our web page so that we can:

- Focus on creating and writing our CSS
- How CSS is a method for separating structure from presentation

Open and attaching the style sheet

A default style sheet has been provided via the project files download link. We'll attach our style sheet via the import method. Between the opening and closing </head> tags, add the following:

```
</head>
<style type="text/css" media="screen">
<!--
@import "style.css";
-->
</style>
</head>
```

The rules we are about to write will only be seen by browsers version 5 or above. Now would be a good time to save your HTML file.

Styling our page background

Our web page will have a subtle repeating background image applied. In your style sheet, add the following:

```
body{
background-image:url(images/background.jpg);
background-color:#999999;
background-repeat:repeat;
font-family:Arial, Helvetica, sans-serif;
font-size:101%;
margin:20px 0;
padding:0;
}
```

Using our body selector, we set the following property-values:

- Background-image:
 - Sets a background image
- Background-color:
 - Set a page background that matches our image background
- Background-repeat:
 - Set to repeat
- Font-family:
 - Set to Arial
- Font-size:
 - Set to a proportional unit of measurement of 101%
- Margin:
 - Top and bottom set to a fixed width of 20 pixels
 - Left and right set to 0
- Padding:
 - Set to zero

We set a page background color which matches our repeating image for visitors who have a slow Internet connection. While the image is being retrieved, the visitors are presented with the default background color. We set padding to zero because browsers apply default margins and padding to HTML elements. Setting this globally ensures all elements will not have any value unless specifically written. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_body.html

Inserting our header image

We'll complete two orders of business here:

- Write a CSS rule to apply our header image
- Write a CSS rule to hide our heading one <h1> from browsers

The reason we insert a heading one tag is to:

- Avoid non-semantic standards of an empty DIV tag
- Format the tag for a printed version of a page if necessary

We'll write a rule to hide the heading one from our web version. First, let's write the CSS rule to show our image:

```
#header{
width:760px;
height:100px;
background-image:url(images/header.jpg);
}
```

We set the width and height property to the same value as our heading image. Additionally, we write the following CSS rule:

```
#header h1{
display:none;
}
```

We use a descendent selector to target all heading one tags in a DIV named header. By setting display to none, we remove the element from showing and the physical space it occupies. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_header.html

Writing our #wrapper

In order for our two columns to expand and contract vertically, we'll write a CSS rule for our wrapper:

```
#wrapper{
width:760px;
margin:20px auto;
background-image:url(images/wrapper.jpg);
background-repeat:repeat-y;
}
```

The following property-values are:

- Width:
 - Set to a fixed width of 760 pixels

- Margin:
 - Top and bottom set to a fixed width of 20 pixels
 - Left and right set to auto
- Background-image:
 - Set to our wrapper image
- Background-repeat:
 - Set to repeat down the y-axis

We set width on our wrapper to the same width of our wrapper image. In a fixed width layout, setting left and right margins to auto will center our layout in modern browsers. By setting our image to repeat down the y-axis, as content is added or removed from either left or right columns, our columns will expand or contract accordingly. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_wrapper.html

Do note there is a harmonic balance between the widths used in our Fireworks layout and the CSS rules we are writing. If the images are off by even one pixel, our layout will break. If you're not familiar with CSS to recalculate any math needed to change our layout, please leave the widths used in this article as is.

Create left and right columns

We know from our Fireworks design that our left column is 260 pixels wide and our right is 500 pixels wide. Furthermore, we know that we want to position these content areas as “columns.” When you position elements in CSS you take them out of the normal flow of the document. Therefore, we will position only these two content areas by using the float property. Add the following CSS rules to your style sheet:

```
#leftcontent{
  float:left;
  width:240px;
  padding:10px;
}
#rightcontent{
  float:left;
  width:480px;
  padding:10px;
}
```

If our left column is 260 pixels and our right column is 500 pixels in Fireworks, then why do the widths say otherwise?

Box Model Calculation

According to the W3C, a box's true width is its declared width plus borders and padding. In our example, our mathematical calculation can be expressed as follows:

```
240 (rendered width)=260 (declared width) = 10 pixel padding (left) +
10 pixel padding (right)
```

Padding and borders are counted twice. If your head is hurting at this slightly inept logic you probably not the only one. Think of it in this analogy: You purchase a package which is 60 pixels wide. When it arrives it has 10 pixels of foam padding on the left and right side of the box. Thus, the rendered width of the package is 40 pixels wide according to the W3C. In essence, when you declare a width on a box which is positioned you have two widths: a declared width and a rendered width. The latter is derived from any padding or borders which you assign to this element and is rendered to the browser. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_float.html

Our page is a bit sickly looking at the moment, but we'll fix that next.

Writing our #footer rule

Currently, the reason our footer doesn't extend to the bottom of our page is because it has no height. It has no way of knowing or understanding where the content ends for either left or right columns. In order to give our footer the height it needs to extend to the bottom of the page, we need to clear both floated DIVs to get the footer back into the flow of the document. Additionally, we also insert our footer graphic with the following CSS rule:

```
#footer{
clear:both;
background-image:url(images/footer.jpg);
width:760px;
height:100px;
}
```

The following property-values are:

- Clear:
 - Set to both will clear our floated DIVs (left and right columns) and put the footer back into the normal flow of the document
- Background-image:
 - Set to our footer image
- Width:
 - Set to a fixed width of 760 pixels which matches our image size
- Height:
 - Set to a fixed height of 100 pixels which matches our image size

Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_clear.html

Our page is really starting to take shape!

Formatting our copyright notice

Our background image in our footer makes the current text which is set to a color of black difficult to read. Let's add the following rule to make it a bit easier to read:

```
#footer p{
  font-size:.8em;
  color:#FFFFFF;
  padding:20px;
}
```

We set all paragraphs inside our footer rule to a color of white, and provide some space inside with padding. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_copyright.html

Style our contact link - footer

Our contact link in the footer is also difficult to read, let's add some link states to correct this:

```
#footer a:link{
  color:#FFFCC;
  text-decoration:underline;
}
#footer a:visited{
}
#footer a:hover, #footer a:active{
  color:#CCCCCC;
  text-decoration:none;
}
```

We provide the four states for our footer contact link. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_copyrightlink.html

Format heading two and three

Let's change the default appearance of our heading two and three tags which are present in our right column with the following rule:

```
h2{
  font-size:1.4em;
  color:#666666;
  padding:10px;
}
h3{
  font-size:1.2em;
  color:#666666;
  padding:10px;
}
```

We set an appropriate font size and color, along with padding to provide some horizontal and vertical off-set between our headings and the edge of our right column. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_formatheadings.html

Format paragraphs

Our paragraphs are sitting close to our right columns edge and they are too large and fairly difficult to read. Add the following rule to correct this:

```
p{
  font-size:.8em;
  line-height:1.7em;
  padding:10px;
}
```

Some people feel as though setting line-height on paragraphs of text makes for easier reading. Padding is set to provide a little breathing room inside our right column. Save and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_paragraphs.html

Formatting our menu

We have definitely saved the best part for last, styling our menu to simulate button behavior. We need to do the following:

- Remove default indentation
- Remove bullets
- Provide a width for our list items
- Make them simulate buttons
- Center them in our left column

Let's get to it!

Remove default indentation

List items by default have indentation applied to them. We write the following rule to turn this indentation off:

```
ul#navigation{
  margin:0;
  padding:0;
}
```

Padding set to zero satisfies Internet Explorer and Opera, while margin set to zero satisfies Mozilla. Before previewing your file, add the ID of navigation to your unordered list:

```
<ul id="navigation">
<li><a href="javascript:;">menu one</a>
<li><a href="javascript:;">menu one</a>
<li><a href="javascript:;">menu one</a>
<li><a href="javascript:;">menu one</a>
</ul>
```

Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_ulindentation.html

Remove bullets

List items by default have bullets. We don't want these if we simulate buttons, so we turn these off with the following rule:

```
ul#navigation li{
  list-style-type:none;
}
```

Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_ulbullets.html

Create our buttons

In order for our link text to simulate a button, we need to declare a global selector on our anchor tag inside our unordered list:

```
ul#navigation li a{
display:block;
width:180px;
margin:12px auto;
padding:4px;
text-align:center;
background-color:#CC9900;
color:#000000;
border:1px solid #000;
font-size:.8em;
text-decoration:none;
}
```

This rule is so massive it must do something important, and it most certainly does. The property-values are:

- Display:
 - Set to block will make the entire link text clickable or “hot”
- Width:
 - Sets each link button to a fixed width of 180 pixels
- Margin:

- Top and bottom set to a fixed width of 12 pixels to provide vertical off-set between each link button
- Left and right set to auto will center our link buttons in our left column
- Padding:
 - Set to a fixed width of 4 pixels to provide space inside our link button text

The remaining property-values are straight forward. Save your file and preview.

http://midwestwebdesign.net/tutorials/csslayout/index_ullinkbutton.html

Creating our link states

In order for CSS to simulate button behavior and rollover effects without the use of JavaScript we need to create the four states to our buttons as follows:

```
ul#navigation a:link{
  background-color:#CC9900;
  color:#000000;
}
ul#navigation a:visited{
  background-color:#990000;
  color:#ffffff;
}
ul#navigation a:hover, ul#navigation a:active{
  background-color:#CCCCCC;
  color:#000000;
}
```

We provide appropriate background color and a foreground color to suit our layout. Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_ullinkbuttonstates.html

Create a “you are here” marker

Most websites today provide a visual indicator to their visitor as to which page they’re currently viewing. Adding this to our web page is a cinch with the following addition:

```
ul#navigation li a.current{
  background-color:#990000;
  color:#ffffff;
}
```

We create a class named **current**, which is nested inside both a list item and an unordered tag. Before previewing, add the class to the first list item button:

```
<ul id="navigation">
<li><a href="javascript:;" class="current">menu one</a>
<li><a href="javascript:;">menu one</a>
<li><a href="javascript:;">menu one</a>
<li><a href="javascript:;">menu one</a>
</ul>
```

Save your file and preview the results.

http://midwestwebdesign.net/tutorials/csslayout/index_youarehere.html

Summary

In this article you learned how to create and design a web page using CSS. You also learned:

- How to create our layout in Fireworks
- How to optimize our individual graphics in Fireworks
- How to export our graphics to be used in our website
- Create a logical structure for our web page using DIVs
- Simulate equalizing columns
- Position columns using floats
- Box Model Calculation
- Make our footer extend to the bottom of our page
- Format various elements our page
- Create and simulate buttons using an unordered list
- Create a you are here marker to indicate to our visitor what page they're currently visiting

After this article, you have learned the fundamentals relating to creating a layout in Fireworks and then using CSS to create your web page. Take the knowledge acquired in this article to create and design any layout of your imagination!

If you have questions, please follow the link below.

<http://midwestwebdesign.net/tutorials/contact.aspx>