

Advancing the Effective  
Use of Technology in Education

# NISD Technology Services



## Take Charge of Your Web Site with Cascading Style Sheets!



A workshop for the  
2007 TCEA Conference

M. Marlo Brown  
Web Development and Training  
Northside ISD, San Antonio, TX

[www.nisd.net/technology/training/brown](http://www.nisd.net/technology/training/brown)

## Introduction

Cascading Style Sheets (CSS) are powerful tools used to determine the appearance of Web pages. They were created to address an increasing number of non-standard HTML tags that were being used to achieve various visual effects, often with inconsistent results. Style sheets offer consistency and far more power and flexibility than “pure” HTML. Style sheets are also the standard method for controlling the appearance of text in Web pages.

CSS can be used with any Web editing program, such as Dreamweaver, Netscape Composer, GoLive, etc. Some programs, however, have limited capabilities, when it comes to writing CSS. For this reason, we’ll work with Microsoft Notepad in this lesson to write CSS directly. This allows us to use CSS with almost any Web page and almost any editing software.

CSS also allow you to update a single page, a few pages, or a whole website much faster and more easily than ever before. An external CSS file can be used to define a set of styles (also called rules) for multiple pages. Making a change to a rule applies the change to every page that uses that rule.

## Course Contents

This session will cover the following:

- Using Notepad to open and save a CSS file
- Modifying existing HTML tags, using inline CSS styles
- Creating embedded style sheets
- Creating external style sheets
- Using CSS styles to control fonts
- Using CSS to change the way that HTML tags behave
- Positioning page elements with CSS

## Table of Contents

Introduction .....	ii
Course Contents .....	ii
Table of Contents.....	iii
Cascading Style Sheets (CSS) – A Brief Introduction.....	1
Why Use CSS? .....	1
Why “Cascading?” .....	1
Microsoft Notepad.....	2
Why Use Notepad? .....	2
Other Word Processors.....	3
Practice Website .....	3
CSS Syntax .....	4
CSS Comments.....	5
Inline Styles .....	5
Let’s Get Started!.....	6
Embedded (Document-Level) Styles .....	6
External Styles.....	7
Creating an External Style Sheet .....	8
Linking or Importing an External Style Sheet .....	8
Fonts.....	9
Font Names.....	9
Which Font? .....	9
Font-size .....	11
Font Color .....	12
Classes .....	13
Text Classes.....	13
Creating Large and Small Text Style Classes .....	13
Borders, Margins and Padding.....	14
Create a Margin for Images.....	14
Text Boxes.....	15
Style Pseudo-Classes and Links.....	16
Link Appearance and Behavior .....	17
The Holy Grail: CSS Positioning .....	17
CSS Positioning in Three (Fairly) Easy Steps .....	18
Make a Positioning Stylesheet File.....	18
Background Color and Images .....	18
Page Heading (Banner Image) .....	19
Navigation Menu Styles .....	19
Main Content Styles.....	20
Activate the Positioning Styles.....	20
Modifying the Index Page .....	22
Further Reading .....	23
Notes: .....	24

## Cascading Style Sheets (CSS) – A Brief Introduction

### Why Use CSS?

Cascading Style Sheets allow you write a series of simple rules to control the appearance of nearly all aspects of a website, from simple one-page sites, to a large, complex site with hundreds or thousands of pages. Some of the advantages of CSS include:

- Less time spent in maintenance/updates
- Faster creation of new pages
- Smaller file sizes for pages – your pages will download faster for visitors
- More accessibility for persons with disabilities
- Better consistency in the “look and feel” of pages throughout the site
- Greater control over page appearance than was possible with HTML alone

In short, Cascading Style Sheets will save you time and give you more control over how your site looks. CSS will also make the site more accessible to visitors, especially those with disabilities or with slow Internet connections.

### Why “Cascading?”

Cascading refers to the way that Web browsers treat the three kinds of style sheets. There is a “cascade” of style sheets, in which the Web browser resolves any conflicts between style sheets by giving preference to some style sheets. The three types of CSS are:

**External style sheets**, which can be used by one Web page or an entire Web site. A single page can also link to more than one style sheet The browser reads the style sheet(s) and uses the style sheet instructions to format the page.

**Embedded style sheets** are part of the code of a single Web page. The browser gives preference to embedded style sheets, so any embedded instructions that conflict with external style sheets will override the external CSS.

**Inline styles** are commands within an HTML tag. These are given the highest priority by the browser, meaning that they override both external style sheets as well as embedded styles.

## Microsoft Notepad

Notepad is Microsoft's text editing program. It is like a very basic word processor, with very few features. This actually works to our advantage in HTML editing, as there are no distracting tools such as automatic text "correction" to interfere with writing HTML.

### Why Use Notepad?

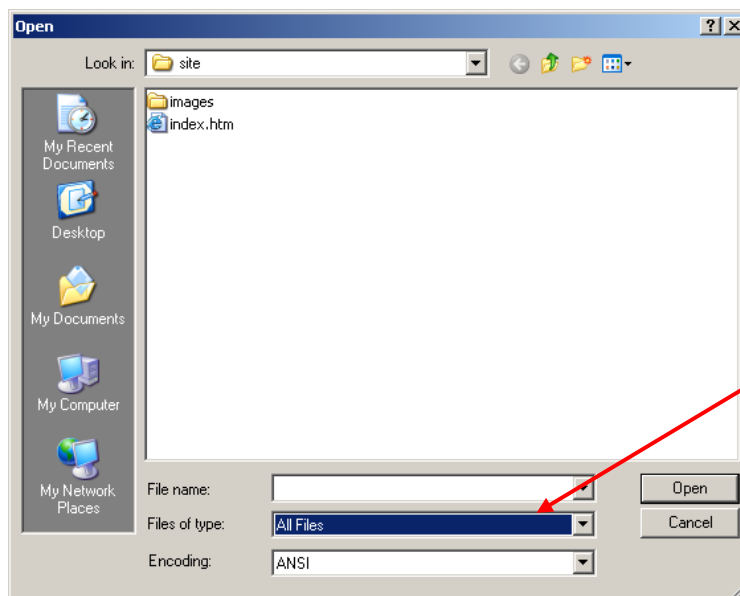
It helps you to learn CSS by typing the rules by hand, at least at first. This also lets you use CSS whenever you want, even if you are working with a Web editor that does not include CSS. Even programs that incorporate CSS may have limited functionality, so the ability to work at the code level will give you great flexibility.

Many Web-based programs – such as Moodle – let users see an "HTML" window as part of editing. With inline CSS styles, you control the appearance of almost any page.

### Opening a CSS file with Notepad

To open an existing CSS file in Notepad

1. Click **File**
2. Choose **Open**



3. Make sure to select **All Files** under **Files of type:**, or you will not be able to see anything but text files.
4. Browse to the file and click .

## Saving a CSS File with Notepad

1. Click **File**
2. Select **Save As**
3. Give the file a name ending with **.css**

Examples:

**styles.css    lopez.css    colors\_and\_fonts.css**

4. Be sure to select **All Files** under **Save as type:**.

## Other Word Processors

Word processors, such as Microsoft Word, Microsoft Wordpad, and other programs can be used to create CSS, as long as you save your CSS files as **text**. If you save in any other format, the Web browser will probably have difficulty reading your styles page(s). Remember that the filenames should end with the **.css** extension to ensure that browsers recognize your files as style sheet files.

## Practice Website

This set of exercises will use a practice website – the “Francis A. Hamer Middle School” site – to see the effects of CSS on existing Web pages. This simple site consists of four (4) Web pages and several images. The files will be provided by your instructor.



## CSS Syntax

The rules in Cascading Style Sheets follow a simple, but specific, syntax that isn't hard to learn. Let's take a look at a few simple rules:

```
h1 {color: blue; font-style: italic}
```

The above rules change the way that the `<h1>` (Heading 1) HTML tag works. In a page where these rule are in effect, the contents of all `<h1>` tags will be shown in blue and italicized.



Note the details of how the rules were written:

- The first part – the name of the rule – comes first. The formal term for this part is the **selector**.
  - In this example, it was the name of an HTML tag whose appearance will be changed.
- The second part – the actual rule – is enclosed in curved brackets. The formal term for this part is the **declaration**.
  - In each of the two rules shown, the **property** is followed by a colon and then the **value**.
  - A semicolon follows the first rule, to separate it from the second rule.

Multiple rules are often listed vertically, in embedded and external style sheets, to make them easier to read and edit. The first bracket is written right after the selector, with each rule on its own line, and the closing bracket below the rules. By indenting the rules, it's easy to see keep track of everything. Two selectors, and their accompanying rules, are shown in the example below:

```
h1 {  
    color: navy;  
    background-color: yellow;  
}  
h2 {  
    font-family: arial, helvetica, sans-serif;  
    color: green;  
}
```

## CSS Comments

Comments can be inserted into style sheets. Comments can help you to remember what a particular style is intended to do, or can be useful to someone else who may need to edit a style sheet long after it was created. Comments must begin with `/*` and end with `*/`. An example follows:

```
body {           /* Sets default font for page */
  font-family: arial, helvetica, sans-serif;
  font-size: 1.0em;
}
```



Inline styles are the most “powerful” of the three types, in that they will override any conflicting styles in external or embedded style sheets.

## Inline Styles

Inline styles are the simplest to write. They are entered within HTML tags, to change the appearance of that single tag’s contents. They can be used in Web applications – such as Moodle or CMS – where you can bring up an HTML editing window, but you cannot modify the `<head>` of the document to use embedded styles or to link to an external style sheet.

Inline styles are also useful for changing just a single element of a Web page, while leaving similar elements untouched. Because they override embedded and external styles, you can use an inline style to change a single element in a page, even though the inline style conflicts with a previously-defined embedded or external style.

Here’s an example of an embedded style. Notice how it differs from the previous examples:

```
<p align="left" style="font-family: tahoma;">
```

Note that we did not use the brackets that were present in the other styles. Instead, our syntax is much like that used in HTML attributes, such as the `align` attribute that precedes the style. In the second example, below, multiple inline styles are defined:

```
<blockquote style="background-color: #F8F8FF;
color: #000000; border: 1px solid; padding: 2px;
font-size: .9em;">
```

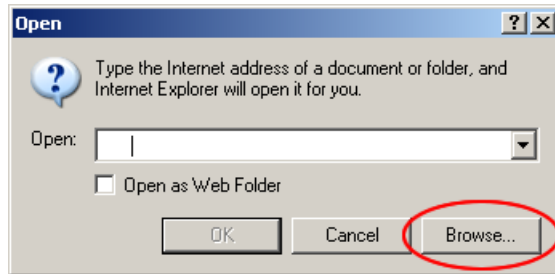
## Let's Get Started!

Let's try out some CSS on the sample site, starting with an inline style. Francis A. Hamer Middle School has some unfortunate school colors: salmon (#FF6666) and white (#FFFFFF). We'll use salmon in one of the Horizontal Rules (horizontal lines) of the Hamer "home" page.

1. If you have not already done so, copy the **CSS\_demo** folder (provided by the instructor) to the desktop of your computer.
2. Open Notepad.
3. Use Notepad to open the index.htm file of the Hamer Middle School website. This is the "home" page for Hamer site.
4. Look for the **<hr>** above the **Copyright** statement, near the bottom of the Web page.
5. Modify it as shown by the **bold, red** text below:

```
<hr style="color: #FF6666;">
<div align="center">
Copyright &copy; 2007, Northside ISD. All rights reserved.
</div>
```

6. **Save** index.htm.
7. Open Internet Explorer (or another Web browser).
8. Select **File > Open** and hit the **[Browse]** button in the **Open** window that appears.
9. Browse to the **index.htm** file of the Hamer website and click the **[Open]** button.
10. Click **[OK]**. You should see the first page of the Hamer site, with the first Horizontal Rule turned to a lovely salmon color.
11. Leave Notepad and the browser open. You'll need them both later.



Some browsers may not recognize certain styles, such as color changes in Horizontal Rules.

## Embedded (Document-Level) Styles

Embedded styles are the way to apply a rule or rules to just one page in your site. Embedded styles are given priority over external style sheets, so the embedded styles will take effect, even if they conflict with an external style.

On the other hand, writing embedded rules in multiple pages could become very time consuming. Even worse is that edits and updates will have to be performed on every page – and every page will have to be uploaded – if you choose to go with embedded styles rather than external styles.

Both embedded and external styles use standard CSS syntax, but embedded styles must be nested within **<style>** tags and placed in the **<head>** of the HTML document they're used in. We also use the HTML code for non-displaying comments (**<!--** and **-->**) at the top and bottom of the styles. This is for older browsers that may not recognize CSS. If the comment code were not used, your styles could be displayed as part of the Web page!

Let's imagine that you've decided to use the Verdana font in your index page, and the Arial font for the rest of the site. We'll use an external style to set the font for the whole site, but first, we'll use an embedded style to set the index page font and font size.

1. In Notepad, type the style (shown below in **red, bold** text) anywhere between the **<head>** **</head>** tags of the index.htm file:

```
<style type="text/css">  
<!--  
body {  
    font-family: verdana, helvetica, sans-serif;  
    font-size: 1.0em;  
}  
-->  
</style>
```

2. **Save** index.htm.
3. **Refresh** the page in your browser to see the change take place.

## External Styles

External style sheets are the most powerful type of CSS, because any number of Web pages can link back to a single external CSS file. By making changes in that single CSS file, you can change the look of every page that uses it!

If you need to make an exception to your external style sheet(s), simply use an embedded style to change a single Web page, or an inline style to change a single page element.

## Creating an External Style Sheet

1. Open a second Notepad window.
2. Click **File > Save As**.
3. Make sure you're saving as **All Files** (not as a Text Document)
4. Give the file your name, followed by the **.css** extension, such as:

marlo.css      vicky.css      fred.css      smith.css

5. **Save** the style sheet file.
6. Type the following in your new external style sheet:

```
body {  
    font-family: arial, helvetica, sans-serif;  
    font-size: 1.0em;  
}
```

7. **Save** the external style sheet.

If you take a look at the Web pages in the site, you'll find that nothing has changed. This is because we have to link or import the style sheet for it to take effect.

## Linking or Importing an External Style Sheet

We need to insert a command into the **<head>** part of each Web page, in order for the rules in the external style sheet to take effect. Style sheets can either be linked or imported. There is no major difference with recent browsers, but some older browsers may not understand the import command. You can take advantage of this by linking style sheets with simple CSS (for all browsers) and importing style sheets with more complex styles (so that only recent browsers will see them). Here is an example of both linking and importing in one Web page:

```
<link rel="stylesheet" href="marlo.css" type="text/css"  
media="screen">
```

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

In the example above, marlo.css contains font, color, margin and border rules, while center.css contains the rules for centering the content of the Web page in a 625-pixel-wide "box" for easier reading. In both examples, we set the media to "screen," in case we ever decide to create a set of "print" style for printing the Web page. This is beyond the scope of this class, but it's a good practice to follow.

1. Use Notepad to **Open** the **admin.htm** Web page.
2. Type the following anywhere between the **<head>** **</head>** tags of your document:

```
<link rel="stylesheet" href="marlo.CSS" type="text/css"
      media="screen">
```

3. Be sure that the filename of your CSS file in the above text (shown in a different font) is the same as the filename of the external style sheet you just created.
4. **Save** admin.htm and open it in your browser to see the changes.
5. Repeat the above steps with the following Web pages:

index.htm  
staff.htm  
aboutus.htm

## Fonts

The ability to set fonts is one of the great advantages of CSS. Fonts are generally listed in a series, separated by commas. They are arranged in order of preference, with our preferred font first, followed by a list of more “generic” fonts. This is so that the browser will look for the first font. If it can’t display it, it will skip to the next one, and so on. Typically, the last font in the series is either **serif** or **sans-serif**, as these are about the most generic fonts possible.

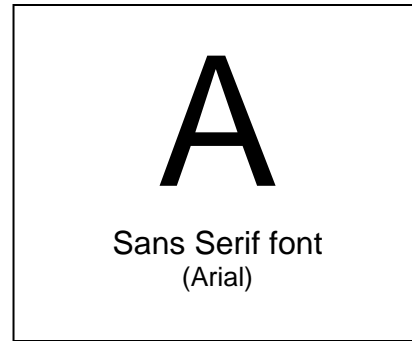
## Font Names

You must use the exact name of the font for it to be displayed. The easiest place to find the correct name is in the PC version of Microsoft Word. Just click the font drop-down and use the font name exactly as it appears in the list.

## Which Font?

Fonts can generally be divided in two large groups: serif fonts and sans-serif fonts. While good for printed documents, serif fonts, like Times New Roman, are not very good for reading on a computer screen. On the other hand, sans-serif fonts, like Arial, are generally easier to read on a computer screen.

Unusual fonts – such as Jokerman, Comic Sans MS, or Papyrus – can be used sparingly, in headings, for example, to lend interest to a page. If they are used for the main text, however, the clarity of your page will suffer. It is strongly suggested that you choose a professional-looking sans-serif font for your website. Likewise, serif fonts should be used sparingly.



A few suggested fonts:

Arial  
Century Gothic

Tahoma  
Verdana

Here are some examples of rules using fonts. Notice that **font names with more than one word must be enclosed in a set of quotes**. Font names are not case sensitive. Capitalize them if you wish.

```
p {  
    font-family: Arial, Helvetica, Sans-serif;  
    font-size: 1.0em;  
}  
h3 {  
    font-family: "Comic Sans MS", Arial, Sans-serif;  
}
```

We've already used CSS to set default fonts for the four pages in our site. Let's make a change to the style sheet:

1. Using Notepad, open your external style sheet (george.css, etc.).
2. Add the word **verdana** to the beginning of the font list. Your style sheet should now look like this:

```
body {  
    font-family: verdana, arial, helvetica, sans-serif;  
    font-size: 1.0em;  
}
```

3. Add the following additional style to your external style sheet:

```
h1 {  
    font-family: "arial black", arial, sans-serif;  
}
```

4. Save the style sheet and Refresh your Web browser to see the changes.



Ems are the recommended unit of measure for font styles.



## Font-size

The most common units of measure for font size on the Web are probably pixels or points. This is unfortunate, because some browsers (Microsoft Internet Explorer 6 in particular) can't resize the text in a Web page if points are pixels are used.

A better unit of measure is the **em**. It's supposed to be the width of an upper-case **M** in size. It's easier to think of it as a "percentage font." For example, a size of 1.0em is equivalent to the default size for a font. A size of .9em is equivalent to 90% of the default size, and 1.2em is like 120% of default.

### Important: Ems are a relative measure!

Ems will be displayed as a percentage of the font where they appear. For example, if you put some .8em text in the middle of a paragraph of .9em text, the size of the new text will be 80% of the size of the small text where you placed it.

Let's change the font-size of two of the HTML tags in our website. We're going to make the tag for bold text (**<strong>**) 5% larger than regular text. That doesn't sound like much, but it makes bold text really stand out in a paragraph. The really important tag is the **<tr>**. This is table row, and it will let us format the text in the 222 table cells in the Staff page in just one step!

1. Open **index.htm** in your browser.
2. Open your external style sheet, using Notepad
3. In your external style sheet, add the following text below the existing style:

```
strong {  
    font-size: 1.05em;  
}  
tr {  
    font-size: .8em;  
}
```

4. **Save** the CSS file and **Refresh** the index.htm and staff.htm pages in your browser to see the text in the Navigation menu and the staff table change in size.

This last exercise shows another advantage of CSS. Plain HTML only allows for seven sizes of text. With CSS and the ems unit of measure, there is no limit to the number of text sizes you can select.

## Font Color

To change the color of text before the advent of CSS-compliant browsers, you had to use the **<font>** tag, which was not officially part of HTML and caused plenty of headaches for some Web designers.

Now we use the color rule to change the color of almost any page element, including text! CSS color rules can be written using the names of colors or the six-character hexadecimal color codes. These 17 color names should work just about everywhere, but be careful of your spelling! Write **grey** instead of **gray**, and you'll probably find that your browser reads the color as **green!**



Hexadecimal color charts can be found on the Web. Try using a search engine with the search words:

hexadecimal  
color chart

aqua	green	orange	white
black	lime	purple	yellow
blue	maroon	red	
fuchsia	navy	silver	
gray	olive	teal	

We already changed the color of a single **<hr>** at the beginning of this lesson, so let's try something a little more dramatic:

1. Open your external style sheet with Notepad.
2. Add the text shown in bold below. To ensure continuity, your CSS file should match the text in this example when you're done:

```
body {  
    font-family: verdana, arial, helvetica, sans-serif;  
    font-size: 1.0em;  
}  
h1 {  
    font-family: "arial black", arial, sans-serif;  
    color: #003399;  
}  
strong {  
    font-size: 1.05em;  
}  
tr {  
    font-size: .8em;  
}  
hr {  
    color: #FF6666;  
}
```

3. **Save** the CSS file and browse through the website, **Refreshing** each page as you go.

## Classes

So far, we've only created rules that changed the way that HTML tags appeared. These styles took effect automatically, but they had limitations. One issue is that they made it difficult to affect just one word (or some other small part of a page) at a time, and we had no way to tell them to ignore some instances of a tag. It was all or nothing. Classes allow us to create custom styles. We give them names of our own choosing and use them only when we want to. Classes give us a great deal of freedom to create our own look for a document.

Classes generally have names that begin with a period. Class names can have letters or numbers, but no spaces or punctuation. The period at the beginning of the name is not used when we apply the style in a Web page.

## Text Classes

It would be useful to have a style that allowed us to make large text without having to use a heading. Headings always have a blank line above and below them, but a class would allow us to make any line or word or letter larger, no matter where it was in the document. Likewise, a small text class lets us choose exactly which text we want to decrease in size, regardless of what is near it.

## Creating Large and Small Text Style Classes

1. Open the external style sheet with Notepad.
2. Add the following two classes to your style sheet.

```
.largertext {  
    font-size: 1.2em;  
}  
.smalltext {  
    font-size: .7em;  
}
```

3. Save the CSS file.
4. Open any of your Web pages with Notepad.
5. Find the copyright statement near the bottom of the Web page, and use the following `<span>` tags to apply the style:

```
<span class="smalltext">  
Copyright &copy; 2007, Northside ISD. All rights reserved.  
</span>
```

6. **Save** the page and **Refresh** it in the browser to see the text change.
7. Repeat steps 4-6 with the other three Web pages.

## Borders, Margins and Padding

One of the frustrations of trying to put either a border or a margin around an object in a Web page with just HTML is that it's been necessary to place it in a table. The table added a lot of complexity to a Web page, and it seemed inelegant to be forced to use a table, just to put a little padding around an image, or a border around some text.

We'll deal with borders, margins, and padding. The important thing to remember is that margins are space outside a border; padding is space inside a border.

One really great feature of CSS is that all sides of an object are independent of each other. We can have four different margins, padding or borders on each of the four sides of an object.

## Create a Margin for Images

The photos on the Administration and About Us pages need to have a small amount of space between them and the surrounding text. We'll look at three ways of creating the rule for the margin. One method takes advantage of the clockwise order in which CSS addresses the four sides of an object: Top, Right, Bottom, Left. Eric Meyer suggests remembering this order – T-R-B-L – with the word “trouble.” The third method uses the `<img>` tag instead of a class.

### First Method

1. Using Notepad, add the following rule to your external style sheet:

```
.photomargin {  
    margin: 1px 3px 1px 3px;  
}
```

2. **Save** the CSS file.
3. Apply the photomargin class to the photo of the Texas Ranger badge in the aboutus.htm page. Here is the code:

```

```

4. **Save** the About Us page and **Refresh** it in your browser.
5. Repeat steps 3-4 for each of the three photos in the Administration page.

## Creating a Margin for Photos - Second Method

1. Using Notepad, add the following rule to your external style sheet:

```
.photomargin {  
    margin-top: 1px;  
    margin-right: 3px;  
    margin-bottom: 1px;  
    margin-left: 3px;  
}
```

2. Apply the photomargin class to the photo of the Texas Ranger badge in the aboutus.htm page. Here is the code:

```

```

3. **Save** the About Us page and **Refresh** it in your browser.
4. Repeat steps 3-4 for each of the three photos in the Administration page.

## Creating a Margin for Photos – Third Method

This method is the simplest and requires the least work. We simply use CSS, and the **<img>** tag, to make a margin for all images.

1. Open the external style sheet in Notepad.
2. Add the following rule:

```
img {  
    margin: 1px 3px 1px 3px;  
}
```

3. **Save** the CSS file and **Refresh** the Administration and About Us pages in your browser.

## Text Boxes

Now we'll bring together several of the things we've learned to create text boxes with CSS. These are useful for long quotes or other text that you would like to highlight and set apart. They work with the **<blockquote>** tag. Notice that we use different borders on the top/bottom, compared to the sides.

1. Open your external stylesheet with Notepad.
2. Add the following rules:

```
blockquote {  
    color: #000000;  
    background-color: ivory;  
    border-top: 1px solid #191970;  
    border-right: 0px;  
    border-bottom: 1px solid #191970;  
    border-left: 0px;  
    padding: 6px;  
    font-size: .9em;  
}
```

3. **Save** the CSS file.
4. Open **aboutus.htm** with Notepad.
5. Find the **<p>** and **</p>** of the second paragraph of Latin text, and replace them with **<blockquote>** and **</blockquote>**.
6. **Save** the page and **Refresh** it in your browser.

## Style Pseudo-Classes and Links

Pseudo-classes are used primarily to modify the appearance and behavior of links in a website. This is extremely powerful, but we have to be careful to avoid changing things so much that visitors don't recognize our links! People are accustomed to looking for blue, underlined text links. For this reason, we make three recommendations:

1. If you change the color of links, do not turn off the underlining.
2. If you turn off underlining, don't change the link colors.
3. If you must change color and turn off underlining, place the links in an obvious navigation menu.

Browsers recognize four conditions for links in a Web page:

- Not yet visited
- "Moused over" (while the cursor is on the link, but the visitor hasn't clicked)
- Being visited (when the visitor clicks, and while the new page loads)
- Visited

The CSS terms for these four states are `a:link`, `a:hover`, `a:active`, and `a:visited`. Let's modify our site so that links are always blue, but not underlined. The links will be underlined and bold when a visitor "mouses over" them and will not change colors after being "visited."

## Link Appearance and Behavior

1. Open your external style sheet, using Notepad.
2. Add the following styles:

```
a {
    text-decoration: none;
}
a:hover {
    font-weight: bold;
    text-decoration: underline;
}
a:visited {
    color: blue;
}
```

3. **Save** the CSS file and browse through your site, **Refreshing** pages to see the links. Try the links at the bottom of the pages.

## The Holy Grail: CSS Positioning

One of the most powerful features of CSS is that it can be used to determine the position of various page elements. This has multiple benefits: first, it allows us to get away from the use of tables for positioning page elements. Because we can use a single, central file to position the page elements in our site, we save a lot of time that would have been spent creating tables and tinkering with them.

Another very important benefit of CSS positioning is that allows us to disconnect the order of content in the HTML code from its position in the browser window. In other words, the main content of the page can be at the top of the HTML, with the navigation menus, banners, etc. at the bottom. Visitors with visual disabilities often use a program that reads the page content to them, in the order that it appears in the HTML code. With CSS positioning, the program can start reading with the main content, instead of having to wade through a banner description, navigation menu, etc.

We'll use some basic positioning techniques with the Hamer website. We'll move the navigation menu over to the side and restrict the size of the main content area for easier reading. We'll also put the banner image at the bottom of the HTML code to make the page more accessible to page reading software, while displaying the banner at the top of the browser window.

We will create the new styles in a separate positioning style sheet. This helps to avoid confusion with the other styles in use in the site.



### Note: Look for the section markers!

The three sections of each page -- Header, Navigation Menu, and Main Content -- are all marked clearly within the HTML code, using non-displaying comment lines. This should help you when we're moving the sections around in the Web pages.

## CSS Positioning in Three (Fairly) Easy Steps

Most CSS positioning projects consist of three parts:

1. Write the rules for positioning. Each selector begins with the # sign.

```
e.g., #NavigationMenu {  
    position: absolute;  
    top: 129px;  
    left: 3px;  
    width: 135px;  
}
```

2. **Import** or **Link** the CSS document in the **<head>** of the HTML page.

```
e.g., <style type="text/css" media="screen">  
    @import "position.css";  
</style>
```

3. Identify the sections in the HTML code with **<div>** and **</div>** tags.

```
e.g., <div id="NavigationMenu">  
    .  
    .  
    .  
</div>
```

## Make a Positioning Stylesheet File

1. Open Notepad.
2. Click **File > Save As** and save the file in the same folder as the Web pages. Give it the filename: **position.css**

## Background Color and Images

We'll begin by creating the rules for the background of the page. CSS allows a great deal of control over the behavior of background images. The following rules specify a white background for the site, identify a background image, and then tell the browser to repeat it down the left side of the screen (along the y-axis).

1. Open **position.css** with Notepad.
2. Add the following styles:

```
body {  
    background-color: white;  
    background-image: url(images/gray_bg.gif);  
    background-repeat: repeat-y;  
}
```

3. **Save position.css** and leave it open in Notepad.

### Page Heading (Banner Image)

We'll continue by positioning the 650x126 pixel banner. We'll be using **absolute positioning**, to specify exactly where we want the banner image to be placed. This is simpler than using **relative** or **static** CSS positioning, so it's a good choice for this first project.

The position of the header is referenced from the top-left corner of the screen. We will specify the width of the header, but not the height. That will be determined by the banner image (126 pixels), so we'll set the header height to **auto**.

Our styles for the header begin with a comment line, to help us to identify the page heading part of the style sheet. Comments are always a great idea when style sheets become long and complex.

1. Add the following styles to the position.css style sheet:

```
/*Header, sized for 650 pixel banner image*/  
#Header {  
    position: absolute;  
    top: 3px;      /*3px from top of screen*/  
    left: 3px;    /*3px from left edge of screen*/  
    width: 650px; /*Width of banner image*/  
    height: auto; /*Height determined by image*/  
}
```

2. **Save position.css** and leave it open.

### Navigation Menu Styles

Next we design and position the navigation bar, along the left side of the screen, on top of the **gray\_bg.gif** background image. Like the header, we'll be using **absolute positioning**, to specify the menu's exact position.

The position of the menu is referenced from the top-left corner of the screen. We will specify the width of the menu, but not the height. That will be determined by the contents of the page. Additionally, we'll set a line-height of 220%. This will automatically create "double-spacing" in our menu.

1. Add the following rules to the position.css style sheet:

```
/*Left side nav bar, positioned absolutely*/
/*# at beginning is for use with a div to position menu*/
#Navigation {
    position: absolute;
    top: 129px; /*Sets top below 126px banner, +3 px*/
    left: 3px; /*Menu is 3px from left edge of screen*/
    width: 135px; /*Width of navigation bar*/
    height: auto; /*Height determined by page contents*/
    padding: 3px 0px 0px 3px; /*Padding at top and left*/
    font-family: verdana, arial, helvetica, sans-serif;
    font-size: 16px;
    font-weight: bold;
    line-height: 220%; /*Sets spacing between menu lines*/
}
```

2. Save position.css and leave it open.

## Main Content Styles

Like the other two sections, the main content will be **absolutely positioned**. The tricky part is remembering how much width and height will be taken up by the header and navigation bar.

1. Add the following rules to the position.css style sheet:

```
/*Main contents of Web page*/
#Main {
    position: absolute;
    top: 129px; /*Sets top below 126px banner, +3 px*/
    left: 138px; /*135 nav menu + 3px*/
    width: 505px;
    height: auto;
    padding: 5px 0px 0px 5px; /*Padding at top and left*/
    /*Total width is 3+135+5+505=648 pixels*/
}
```

2. Save position.css and leave it open.

## Activate the Positioning Styles

1. Open aboutus.htm with Notepad.

2. Type the following anywhere between the `<head>` and `</head>` tags.

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

3. Find the following part of the Web page:

```
<!-- BEGIN HEADER -->
<div align="center">

</div>
<!-- END HEADER -->
```

4. Cut and paste the code shown in Step 3 to the bottom of the document, right above the `</body>` tag.
5. Modify the first `<div>` tag of the text you just moved to read:

```
<div id="Header">
```

6. Modify the first `<div>` tag of the navigation menu (located just below a comment stating **BEGIN NAVIGATION MENU**), to read:

```
<div id="Navigation">
```

7. Modify the navigation menu, removing the | characters and replacing each with a `<br>` tag. When you're done, the Navigation `<div>` should look like this:

```
<div id="Navigation">
<!-- BEGINNING OF NAVIGATION MENU -->
<br>
<a href="index.htm">Home</a>
<br>
<a href="admin.htm">Administration</a>
<br>
<a href="staff.htm">Staff</a>
<br>
<a href="aboutus.htm">About Us</a>
<br>
<a href="http://www.nisd.net">NISD</a>
<!-- END OF NAVIGATION MENU -->
</div>
```

8. Modify the beginning **<div>** statement that appears just below the comment stating BEGIN MAIN CONTENT. It should read:

```
<div id="Main">
```

9. **Save aboutus.htm** and **Refresh** it in your browser.
10. Repeat Steps 1-9 with the staff.htm and admin.htm pages.

## Modifying the Index Page

The index page was deliberately made a little harder to convert to CSS positioning, but it's still pretty easy. We have to delete all of the table information that is currently used to position the content.

1. Open **index.htm** in Notepad.
2. Move the Header to the bottom (just above **</body>**), as we did with the other pages.
3. Modify the bottom-of-page Navigation Menu section, just as was done with the other three pages. The section is marked with the comments:

```
<!-- BEGIN NAVIGATION MENU -->  
and  
<!-- END NAVIGATION MENU -->
```

4. In the Main Content section, we need to delete all code between:

```
<!-- BEGIN MAIN CONTENT -->  
<div align="left">  
and  
<!-- BEGIN ANNOUNCEMENTS -->
```

5. Delete the three lines between:

```
<!-- END ANNOUNCEMENTS -->  
and  
<!-- DO NOT DELETE BELOW THIS LINE -->
```

6. Modify the three **<div>** tags of the sections for Header, Main and Navigation to read, respectively:

```
<div id="Header">  
<div id="Main">  
<div id="Navigation">
```

7. Add the following line to the embedded style sheet in index.htm:

```
@import "position.css";
```

8. **Save** index.htm and **Refresh** it in your browser.

This lesson has only scratched the surface of what can be done with CSS. Although there are some excellent books available, there are also some superb Web pages on CSS. Take a look at the list below for a starting point.

## **Further Reading**

### **Cascading Style Sheets**

[http://en.wikipedia.org/wiki/Cascading\\_style\\_sheets](http://en.wikipedia.org/wiki/Cascading_style_sheets)

### **CSS at the Open Directory Project**

[http://dmoz.org/Computers/Data\\_Formats/Style\\_Sheets/CSS/](http://dmoz.org/Computers/Data_Formats/Style_Sheets/CSS/)

### **Cascading Style Sheets: Learning CSS**

<http://www.w3.org/Style/CSS/learning>

### **Accessibility Features of CSS - W3C NOTE 4 August 1999**

<http://www.w3.org/TR/CSS-access>

**Notes:**