

Formatting Pages with Cascading Style Sheets

*Web Publishing with HTML and CSS:
Lesson 9 and Appendix C*

Cascading Style Sheets

- The first implementation was known as CSS1
- Next generation is CSS2
 - Allows specifying of font colors, background colors and graphics, margins, spacing, type style and more
 - Many CSS2 tags are supported by the latest browser versions

CSS Support in Browsers

- Most modern browsers support CSS1 and some support CSS2.
- You can check the support various browsers have for CSS1 and CSS2 styles at:
http://www.w3schools.com/css/css_reference.asp

Validating CSS

- The W3C provides a CSS validator at:
<http://jigsaw.w3.org/css-validator/>
- You can use this tool to make sure your CSS syntax is correct
- You could validate your .css file always, but especially if your styles are not performing as you wish

Learning about CSS

- You could spend an entire class studying CSS
- The best use of CSS is through the use of an external style sheet
- You can use any text editor to create your external .css file
- Later in the semester we will also use the ASP.Net design environment to help us generate and link an external style sheet

About Cascading

- This refers to the ability to combine style information from more than one source
- External style sheets can be linked to more than one page
- Embedded styles can be applied to a single page
- Inline styles apply rules to specific page elements

Cascading Order of Styles

- The styles of external style sheets are applied first
- The styles of embedded styles sheet are applied second and override the external styles
- The inline styles override both external and embedded style sheets
- HTML formatting tags override any CSS styles

Inline Styles – Covered in HTML Lesson 5

- Allows the attachment of style rules to an individual element instead of across an entire page or site
- **** allows applying of style over any amount of text
- Use the **style** attribute to set the inline style rules

`<p> font will be small and bolded </p>`

Embedded Styles - Page level styles

- These are styles that are used within a given page
- Use the **<style>** and **</style>** tag in the head of the page
- Needs the type attribute **type="text/css"**

Example of Embedded CSS

```
<head>
  <style type="text/css">
    body
    {
        background-color: #330066;
    }
    .bigger
    {
        font-size: 16pt;
    }
  </style>
</head>
```

Using Embedded Styles

- Simply use the redefined HTML style or class style within the page
- These styles are only seen within the page where they are created
 - Ex.


```
<p class="bigger">Bigger text</p>
```

External Style Sheets

- These keep style rules as a separate file
- Allows use of many styles with a single file
- Modifications to style sheet file will change all the associated pages
- Uses the **.css** file extension

Attaching an External Sheet

- Like an HTML page, a style sheet is just an ASCII text file
- Usually this file is placed in your site folder
- Attach the style sheet by using the `<link>` tag in your page header

`<link>` tag

- Example:

```
<link rel="stylesheet"
href="mystyles.css" type="text/css" />
```
- This creates a persistent style that is applied to your page regardless of the user's local selections
- The name of the stylesheet linked in this case is: **mystyles.css**

Selectors in CSS

- Any existing HTML tag can serve as a CSS selector
- Be sure this is what you want to do...
 - The rules set in the selector will be applied to all instances of that tag on the page where the CSS is linked
 - Creating a `<p>` selector will apply the rules to all the `<p>` tags on your page.

Example: h1 selector

- Ex. `h1{`

```
color: #996633;
padding-top: 10px;
padding-bottom: 5px;
background-color: #FFFDCC;
}
```
- Uses the HTML tag you are redefining and `{ }`
- Each attribute is separated by a `;`
- These rules will be applied to all `<h1>` tags on this page

Contextual Selectors

- These are used to apply styles to elements, but only when they are nested.
- Ex. `div ol {`

```
color: blue;
}
```

Would only apply the color blue to `` elements nested inside `<div>` elements

Classes and IDs

- Sometimes you want to create your own user-defined style for use anywhere on the page.
- The **class** attribute is used for assigning this user-defined style to groups.
- The **id** attribute is used for assigning a user-defined style to specific elements.

Creating a User-Defined Class

- Begins the class style with a `.` and then the name of the new class
 - Ex. `.bigger { font-family: Arial; font-size: larger; }`
- This new class allows you to group several styles and apply them to text anywhere on your page.
 - `<p class="bigger"> Text inside p </p>`

Use IDs for Unique Styles

- When you want a specific style for one element on a page, assign it to an ID.
- To create a rule to work with an ID, use the `#` character when you define the style
 - Ex. `# footer { font-size: small; }`
- To identify the footer on your page, use the id attribute:
 - `<div id="footer" > Copyright 2006 </div>`

Units of Measure in CSS

- **em** – Relative: height of the element's font
- **ex** – Relative: height of x character in the element's font
- **px** – Relative: pixels
- **in** - Absolute: inches
- **cm** – Absolute: centimeters
- **mm** - Absolute: millimeters
- **pt** – Absolute: points
- **pc** – Absolute: picas

Common Style Sheet Properties

- Margin and padding
- Backgrounds, colors and images
- Border appearance
- Font Appearance and Style
- Text Alignment

Box Properties

- CSS box properties work on box-shaped regions of a page and can be used to:
 - Position elements
 - Control white-space around elements
 - Apply effects to borders
- Once you master the use of box properties, you no longer need to use tables to layout a page.

Controlling the Size of a Box

- Two properties that can be used to control the size of a box element: **width** and **height**.
- Unlike tables that, unless otherwise specified, are only as large as the widest bit of content, many other block-level elements are as wide as the browser window by default.



Borders

- The border property around elements can be seen as a box.
- There are three values associated with the border property, any of which can be eliminated.
 - border: width style color;



Border Values

- **width** can use any unit of measurement to specify the border width
- **width** can also use the thin, medium, thick options
- **style** can use the dotted, dashed, solid, double, groove, ridge, inset and outset options
- **color** specifies the color of the border and can be a hexadecimal or name color value



Using Individual Border Properties

- Additional border properties allow you to set styles for the border's sides individually...
 - border-top
 - border-right
 - border-bottom
 - border-left



Supplying Border Values

- If you provide only one value for a border property
 - the property will be applied to all four sides of the box.
- If you supply only two values
 - the first will be applied to the top and bottom
 - The second will be applied to the sides
- Supply three values
 - First goes to the top
 - Second to the sides
 - Third to the bottom



Margins in CSS

- Any element can have a margin.
- A margin is the space between an element and the adjacent elements to its top, left, bottom and right.
- The margin properties are:
 - margin
 - margin-top
 - margin-left
 - margin-bottom
 - margin-right



Use of Margins

- Margins can be used to achieve similar effects to the **positioning properties** and **padding**
- Margins can be specified as either a percentage, a length or using the keyword auto.

Margin Percentages

- Percentage margin values set the affected margin to that percentage of the **width of the parent element**
 - Ex. a margin-right: 20% sets the width of the right margin to 20% of the width of the element which contains the element.

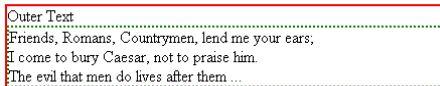
Margin Lengths

- Length margin values can be absolute or relative to the containing element.
 - Use of absolute values such as **px** will ensure the actual length of the margin will always be the same.
 - Use of relative values, like **em** will ensure the size of the margin will always be proportional to the calculated size of the content of the element.

```
/*-Styles in mystyles.css file */
.outer { border: 2px solid red;}
.inner { border: 2px dotted green;
margin: 0px;
padding: 0px; }

<head>
<link rel="stylesheet" href="mystyles.css" type="text/css" />
</head>
<body>
<div class="outer">
  Outer Text
  <div class="inner">
    Friends, Romans, Countrymen, lend me your ears; <br/>
    I come to bury Caesar, not to praise him. <br />
    The evil that men do lives after them ...
  </div>
</div>
```

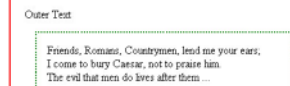
Nested <div>
tags with no
margins and
padding



```
/* Styles in mystyles.css file */
.outer { border: 2px solid red;
padding: 20 px; }
.inner { border: 2px dotted green;
margin: 15px;
padding: 15px; }

<head>
<link rel="stylesheet" href="mystyles.css"
type="text.css" />
</head>
<body>
<div class="outer">
  Outer Text
  <div class="inner">
    Friends, Romans, Countrymen, lend me your ears; <br/>
    I come to bury Caesar, not to praise him. <br />
    The evil that men do
    lives after them ...
  </div>
</div>
```

Nested <div>
tags with
padding and
inner margin



Float

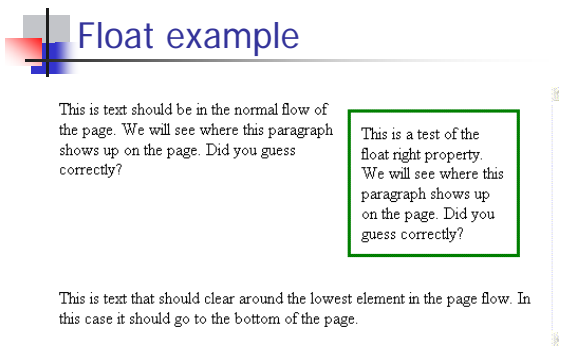
- Normally elements flow down the page from left to right and top to bottom
- If you want to alter this normal flow, you can use absolute positioning or the float property.
- Float is used to indicate that an element should be placed as far as possible to the left or right on the page and other content should wrap around it

```
/* Styles in mystyles.css file */
.right { border: 3px solid green;
padding: 10px;
margin: 10px;
float: right;
width: 30%; }
.bottom { clear: both; }
```

Float Right
Example

```
<head>
<link rel="stylesheet" href="mystyles.css" type="text.css" />
</head>
<body>
<p class="right"> This is a test of the float right property. We will
see where this paragraph shows up on the page. Did you guess
correctly?
</p>
<p> This is text should be in the normal flow of the page. We will see
where this paragraph shows up on the page. Did you guess correctly?
</p>
<p class="bottom"> This is text that should clear around the lowest
element in the page flow. In this case it should go to the bottom of
the page.
</p>
```

Float example



This is text should be in the normal flow of the page. We will see where this paragraph shows up on the page. Did you guess correctly?

This is a test of the float right property. We will see where this paragraph shows up on the page. Did you guess correctly?

This is text that should clear around the lowest element in the page flow. In this case it should go to the bottom of the page.

CSS Positioning

- There are 4 position schemes you can use to try to control how elements are laid out on a page:
 - static** – The default, normal flow
 - relative** – Positions relative to element that precedes it
 - absolute** – Allows positioning in any location on the page.
 - fixed** – not well supported, similar to absolute

Position Properties of Schemes

- There are 4 position properties you can use for any position scheme:
 - top**
 - left**
 - bottom**
 - right**
- These values are specified as the distance from the named side of the enclosing block.

Relative Positioning Example

```
/* Styles in mystyles.css file */
h2.moveleft {
  position: relative;
  left: -20px; }
h2.moveright {
  position: relative;
  left: 20px; }
```

This is a heading in normal position

his heading is moved left to its normal position

This heading is moved right to its normal position

```
<head>
<link rel="stylesheet" href="mystyles.css" type="text/css" />
</head>
<body>
<h2>This is a heading in normal position</h2>
<h2 class="moveleft">This heading is moved left to its normal position</h2>
<h2 class="moveright">This heading is moved right to its normal position</h2>
</body>
```

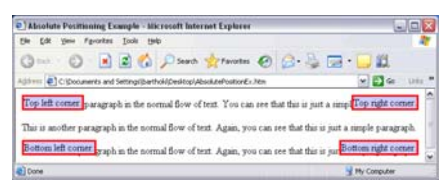
```
/* Styles in the mystyles.css file */
#topleft { position: absolute; top: 0px; left: 0px; }
#topright { position: absolute; top: 0px; right: 0px; }
#bottomleft { position: absolute; bottom: 0px; left: 0px; }
#bottomright { position: absolute; bottom: 0px; right: 0px; }

.box { border: 3px solid red; background-color: #ccccff; padding: 0px; margin: 10px; }
```

Absolute Positioning Example

```
<head>
<link rel="stylesheet" href="mystyles.css" type="text/css" />
</head>
<body>
<div class="box" id="topleft"> Top left corner. </div>
<div class="box" id="topright"> Top right corner. </div>
<div class="box" id="bottomleft"> Bottom left corner. </div>
<div class="box" id="bottomright"> Bottom right corner. </div>
<p> This is a the first paragraph in the normal flow of text. You can see that this is just a simple paragraph. </p>
<p> This is another paragraph in the normal flow of text. Again, you can see that this is just a simple paragraph. </p>
<p> This is another paragraph in the normal flow of text. Again, you can see that this is just a simple paragraph. </p>
```

Absolute Positioning Example



The <body> Tag

- The margin, padding and border of a page can be adjusted by adding styles to the <body> tag
- The default page font family and background color can also be set in the <body> tag
- The text-align for the entire page can also be set in the <body>

Links and CSS

- Links exist in multiple states: unvisited, visited, active, and hover.
- Using CSS you can change the color and appearance of a link in these states.
 - a: **link** { color: blue; }
 - a: **active** { color: red; }
 - a: **visited** { color: green; }
 - a: **hover** { color: yellow; }

Creating Layouts in CSS

- Using an external CSS file, all formatting for the page layout can be placed in the .css file
- The web page would just contain the content of the page, no layout formatting
- This allows the designer to separate format from content

```
/* Styles placed in the mystyles.css file */
body { font-family: Georgia; margin: 0px; background-color: #ff9900; }
#header { font: bold 48px Trebuchet MS;
padding-left: 30px;
border-bottom: 3px solid black;
background-color: #CC0000;
margin-bottom: 0px; }
#content { float: right;
padding: 1px 20px 1px 10px;
width: 70%;
margin: 0px;
border: none;
background-color: #ffffff; }
#nav { float: left;
width: 20%;
margin-top: 0px;
font-weight: bold;
padding: 10px;
border: none;
font-family: Trebuchet MS; }
#nav a { text-decoration: none; color: #000066; }
#nav a: hover { color: green; text-decoration: underline; }

h2 { margin-top: 10px; }
```

CSS File for a Two Column Page

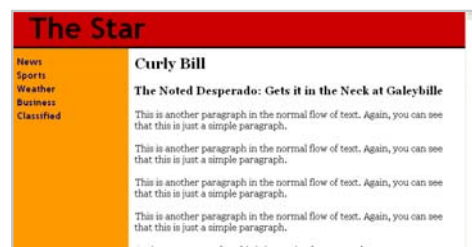
```
<body>
<div id="header"> The Star</div>
<div id="content">
<h2>Curly Bill</h2>
<h3>The Noted Desperado: Gets it in the Neck at Galeyville</h3>
<p> This is another paragraph in the normal flow of text. Again, you can see that this is just a simple paragraph. </p>
...
<p>Again, you can see that this is just a simple paragraph. </p>
</div>

<div id="nav">
<a href="#" ">News</a> <br />
<a href="#" ">Sports</a> <br />
<a href="#" ">Weather</a> <br />
<a href="#" ">Business</a> <br />
<a href="#" ">Classified</a> <br />
</div>

</body>
</html>
```

HTML File for a Two Column Page

Two Column Page Example





Sources for CSS Info

- Tutorials on the W3C Web site on CSS-
<http://www.w3schools.com/css/>
- Insight from the Web Design Group-
<http://www.htmlhelp.com/reference/css>
- W3C's CSS Home Page-
<http://www.w3.org/Style>



3 Approaches to Style Sheets

- External Style Sheets
- Embedded Style Sheets
- Inline Styles
- All these styles are supported in most web development environments like Dreamweaver and ASP.Net
- We will focus on External Style Sheets from now on in this course