IS 360 CSS LAB 1

In this lab, you will work with Cascading Style Sheets (CSS). You will perform the following tasks:

* Understand where and how you declare CSS styles affects which rules are applied as the browser renders the document. To do this, you will work with inline, embedded, and external styles
* Use simple and complex selectors (combinators) to select elements
* Format text using CSS
* Work with colors through named and hex colors and call the rgb function.
* Understand the basic of the box model (padding, margins, borders, and content).
* Format tables and lists
* Using the Chrome debugging tools to view the order in which various styles are applied and overridden

# Lab Abstract

This lab progresses from creating a very simple style rule to create complex rules and styles designed to format tables and create layouts.

* Create a first style rule to see how style rules inherit and override style rules defined by the user agent.
* Create embedded styles and watch what happens via the Chrome Debugging tools.
* Create a couple of inline styles to see how an inline style will override an embedded style.
* Create an external stylesheet.
* Use styles to format tables lists
* Throughout this lab, you will continually explore the styles that apply to typography, borders, and margins.

AS YOU COMPLETE THIS AND THE SUBSEQUENT CSS LABS, I STRONGLY STUGGEST THAT YOU EXPLORE. TRY DIFFERENT COLORS AND MARGINS. RESIZE ELEMENTS. THE MORE THAT YOU FIDDLE, THE MORE THAT YOU WILL LEARN.

# Creating a First Style

In this part of the lab, you will create a first HTML document without any style information.

**To complete various steps in this lab, you should be using Google Chrome. Other browsers provide similar debugging tools but they operate differently.**

## Hands-On Activity: Creating an embedded <style> block

1. Create a html file.
2. Create a **<style>** block inside of the **<head>** section of the HTML 5 page. See the example in the next step. Note the type is set to “text/css”.  
     
   <!DOCTYPE HTML>  
   <html xmlns="http://www.w3.org/1999/xhtml" >  
   <head>  
    <title>Untitled Page</title>  
    **<style type="text/css">  
    </style>**</head>  
   <body>

</body>  
</html>

1. Create the following content within the <body> element. Again, I suggest that you copy this content verbatim to your Web page.

<body>

<article>

<h1>Creating a First Style</h1>

<p>This is a paragraph inside of an <span class="SpecialWord">article</span>.  
 </p>

<section>

<h2>Typography</h2>

<p>This is a paragraph inside of a   
<span class="SpecialWord">section</span>.</p>

</section>

<section>

<h2 style="text-decoration: line-through;">Borders</h2>

<p>This is a paragraph inside of a   
<span class="SpecialWord">section</span>.</p>

</section>

</article>

</body>

At this point, you have created a little content and the embedded <style> block. You have yet to create any styles though.

**In this next set of steps, you will use the Chrome Debugging tools to look at the styles that are defined by the user agent. It’s my strong opinion that being able to use these tools help you to understand how CSS styles are applied and overridden. Google has done a wonderful job here.**

## Hands-On Activity: Interrogating default styles using Chrome and it’s CSS Investigator

1. View the HTML document in Chrome as you have been doing.
2. Next, activate the Chrome debugger by pressing **Shift Control-J**.   
     
   Because Google saves some history information, the exact screens that appear next might differ from my steps. Thus, you might need to explore a bit. You should see a window that resembles the following:  
     
     
     
   Chrome displaying rendered output in a left pane and the debugging tools in the right pane. 

   A menu appears across the top of the debugging tools. The console tab is selected.  
     
   In the above window, the browser output appears on the left and the Debugging windows appear on the right. Your screen might be aligned top to bottom. These windows are draggable so you can resize them to suit you.  
     
   As you can see, the debugging tools pane has a menu with which you can look at what your page is doing. In this lab, you will look at the document structure and CSS formatting. In subsequent labs, you will use these same tools to debug JavaScript code. Your HTML will differ from the one shown in the figure.
3. Click the **Elements** button. If you are asked to select a file, follow the instructions and select the file named **InternalCss.htm**. Your screen should resemble the following: Your html will differ from the one shown in the figure. I have simplified the code for the figures but your results should be very similar.  
     
   Chrome browser appearing with three vertical panes:

   The first contains the browser output,
   The second contains the html code
   The thrid contains the Styles being applied.  
     
   Again, your windows might be oriented differently but one window displays the rendered output. Another the HTML. The rightmost window has different tabs designed to look at different document parts. The **Styles** tab should be selected.  
     
   The HTML text appears document area (center window). The arrows allow you to expand and collapse the document elements. The <body> element is selected in the figure.  
     
   The Styles tab shows you the styles currently being applied. The margin is 8 pixels, which you can see from both the style and the figure. Pay attention to where the style is coming from. The string “user agent stylesheet” is telling you that this is a default style supplied by the user agent (browser). It’s not a style that you created.
4. Now, expand the <article> section and select the <h1> tag. Your screen should look like the following:  
     
   Chrome browser appearing with three vertical panes:

   The first contains the browser output,
   The second contains the html code
   The thrid contains the Styles being applied.

   The <h1> element is selected:  
     
     
   Things have become a bit more complicated but understanding these style rules is very important.

* First, you can see that the user agent is defining a default style for the <h1> tag. It sets the margins and font-weight.
* Some of the styles are crossed out meaning that this style rule has been overridden by another style rule.
* The style rule any(article,aside,nav,section) h1 is more specific than the h1 rule. It adds the font-size specification and overrides the margin attributes. This rule is also defined by the user agent.  
    
  Take a close look at the selector. It says “select any h1 tag that is a descendent of any article, aside, nav, or section tag”.

# Working with Simple Internal Styles

In this next exercise, you will create a first few internal styles and see the effect of those styles using the same Chrome tools that you just used.

*As you make changes to the HTML code, you can usually reload the page directly from the Chrome browser.* ***Right-click in the browser window and select Reload Page****. This is a quick way to reload any changes without having to restart a Chrome window or the debugging tools.*

## Hands-On Activity: CREATE AN INTERNAL STYLE

1. Create the following style in the embedded style block that you created in the previous steps:

<style type="text/css">

**body {**

**width: 800px;**

**padding-left: 25px;**

**padding-top: 25px;**

**font-family: Tahoma;**

**text-align: justify;**

**background-color: lightgray;**

**margin: auto;**

**}**

</style>

The above style sets the width of the document body to 800 pixels, the content width will never change. The content will be centered within the visible region of the browser (margin: auto) The background color of the page is set along with the font. Padding is defined (between content and border) of 25 pixels along the top and left margins.

1. View the document in the browser.
2. Select the <body> element in the HTML window. Its styles should be set similar to the following:  
     
   HTML area shows the <body> element selected.

   The Styles pane shows the body style applied.

   The default margin is overriden.  
     
     
   The styles for the <body> element appear. As you can see, the margin attribute has been overridden from the user agent. It was override by the margin set in the body.
3. Now, select the <p> element for the first article and heading:  
     
   HTML area shows the <body> element selected.

   The Styles pane shows the body style applied.

   The default margin is overriden  
     
   The paragraph <p> element inherits the styles defined in the body. The figure shows the calculated margin, border, and padding.
4. I suggest that you continue to select other objects in the document and see how their styles are applied overridden, and inherited.

# Getting into Some Formatting

The purpose of the previous section of this lab was to show you how, and the order in which, the CSS rules are applied as your document is rendered. In this part of the lab, you will look more into the styles themselves. We will continue to illustrate the CSS process via the Chrome debugging tools though so that you can better understand how the cascading rules work.

The following code sets some style attributes for all of the <h1> and <h2> elements on the page.

h1 {

font-size: 22pt;

font-weight: bold;

text-decoration: overline underline;

text-transform: uppercase;

letter-spacing: 0.3em;

color: maroon;

}

h2 {

font-size: 18pt;

font-weight: bold;

text-decoration: overline underline;

text-transform: uppercase;

letter-spacing: 0.2em;

color: maroon;

}

* The above styles set the font-size, font-weight and color.
* By setting the letter-spacing to 0.2em, we are increasing the default letter spacing by 20%. 0.3em is 30%
* The text-transform setting makes all of the characters upper-case.
* The text-decoration causes a line to be drawn above and below the tags.

## Hands-On Activity: CREATING TYPOGRAPHIC STYLES

1. Create the styles shown in the above code in the same <style> block that you have been using. View the document in Chrome. You should see the text appearing maroon with a line drawn above and below the tags.
2. Again, take a look at the Chrome debugging windows. Select one of the <h2> tags. You should see that our font is overriding the font size from the user agent. The font family is being inherited from the <body> tag.

## Hands-On Activity: CREATING SOME BORDERS

In this part of the lab, you will create a top and left border along the page to give it a little emphasis. The following styles should be added to the preexisting style that you created for the <body> tag:

margin-top:15px;

border-left: solid;

border-left-color: darkred;

border-left-width: 2px;

border-top: solid;

border-top-color: darkred;

border-top-width: 2px;

1. Create the styles shown in the above code in the same <style> block that you have been using. Be sure to create the styles inside the style for the <body> element.
2. View the document in the Browser. You should see the top and left border lines appearing maroon. Again, take a look at the Chrome debugging windows.

# Class Selectors

Recall that you create class selectors by defining style name beginning with a period. The following defines a class named .SpecialWord. Class selectors are case sensitive.

.SpecialWord {

font-weight: bold;

font-size: 1.05em;

}

You reference that class selector from an element as follows:

<span class="SpecialWord">article</span>

## Hands-On Activity: CREATING A CLASS

1. Add the above class to the existing <style> block.
2. Again, view the document in Chrome, and make sure that the debugging tools are activated.  
     
   You should see the “special words” formatted accordingly because the classes are already referenced in the document.

# Inline Styles

Inline styles override embedded styles. To see this, take a look at the following style:

<h2 style="text-decoration:line-through">Borders</h2>

You defined a text-decoration for the <h1> and <h2> tags as follows:

text-decoration: overline underline;

The inline style will override the embedded style because it is more specific. So the decoration will be line-through.

## Hands-On Activity: CREATING AN INLINE STYLE

1. In the main document, modify the second <h2> tag to include the above inline style:
2. View the document again in Chrome. You should see the text adorned with a line-through instead of top and bottom rules.
3. Now, take a close look at the debugging window for the Borders section:  
     
   Inherited element styles.

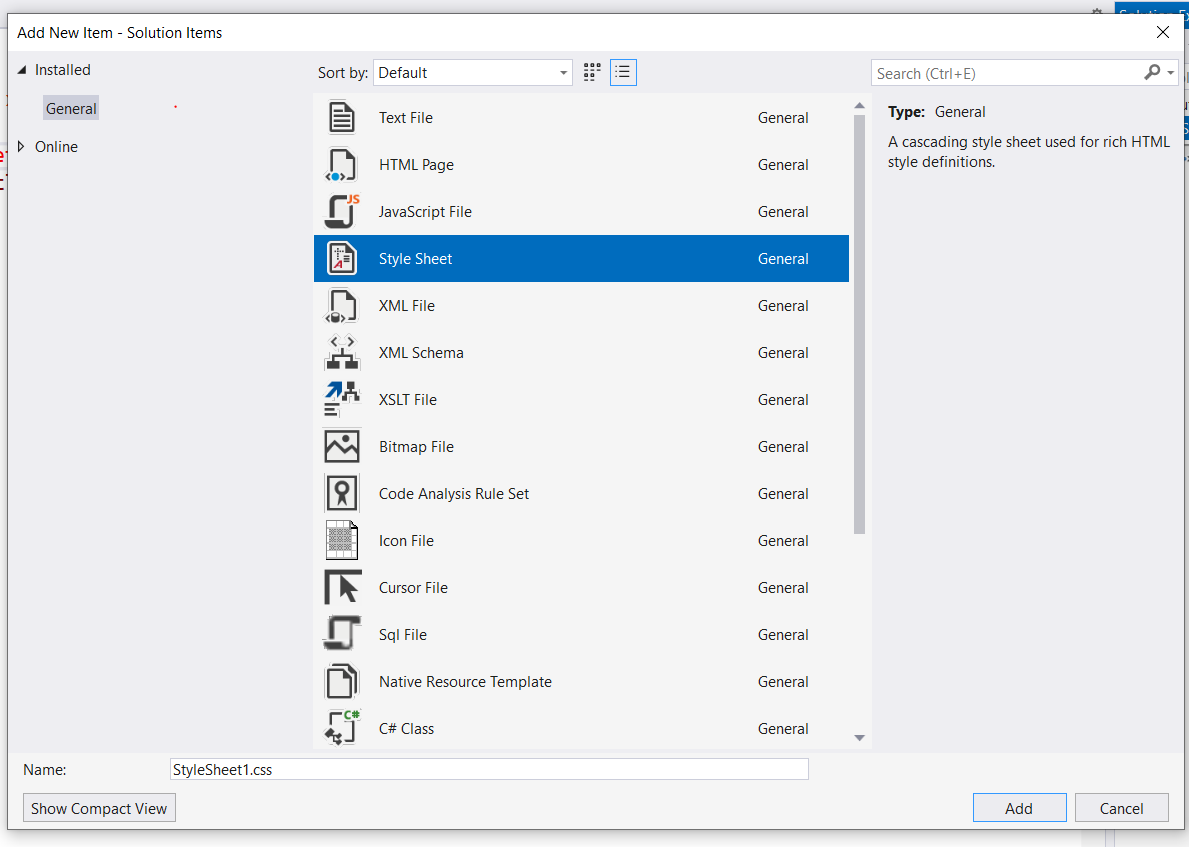
   elemeent.style overrides h2 (embedded), override h2 (user agent).   
     
   The element.style overrides the h2 style. The body styles are inherited.

# External Stylesheets

External Stylesheets are the most common way we implement styles. As you will see in later labs, we use frameworks of external Stylesheets to do much of the tedious layout work for us.

## Hands-On Activity: CREATING AN EXTERNAL STYLESHEET

In this part of the lab, you will create an external Stylesheet. And you will create a page that references the styles on that Stylesheet.

1. Create a new empty style sheet. In Visual Studio, right-click on your project folder in Solution Explorer. Select "Add," then "Add New Item." From the dialog box, choose StyleSheet.css
2. Change the name to **Main.css**:  
     
   
3. Click **OK** to create the style. It should be added to the Solution Explorer.

## Hands-on Activity: LINKING TO AN EXTERNAL STYLE SHEET

An external style sheet is referenced through a <link> tag. It has three attributes named rel, href, and type. Only the value of the href attribute changes and contains the URL of the Stylesheet file. In the following example, a relative reference is used. Thus, the stylesheet should appear in the same folder as the HTML file referencing it.

1. Add a new HTML5 page to the project. Name the page **ExternalCss.htm**.
2. Create a link in the <head> section of the HTML 5 page as follows so as to reference the external style sheet.

<head>

<meta charset="utf-8" />

<title>External Stylesheet Lab</title>

<**link rel="stylesheet" href="./Main.css" type="text/css" /**

</head>

1. Again, to keep the content simple, add the following content verbatim to the page:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title>External Stylesheet Lab</title>

<link rel="stylesheet" href="./Main.css" type="text/css" />

</head>

<body>

<nav id="NavWrap" class="BottomBorder">

<h1>Menu</h1>

<ul>

<li><a href="https://google.com">Google</a></li>

<li><a href="https://google.com">Sample Page</a></li>

<li><a href="https://google.com">Sample</a></li>

</ul>

</nav>

<div id="BodyWrap" class="BottomBorder">

<p>Sample Content</p>

</div>

<footer id="FooterWrap" class="BottomBorder">

<address>

Adebola Adesina<br />

University of Nevada, Reno<br />

</address>

</footer>

</body>

</html>

The above HTML is divided into three sections named NavWrap, BodyWrap, and FooterWrap. These are the semantic sections of the document. They each have ID attributes defines so they will appear only once in any given document. The NavWrap section contains a list of anchor tags. The BodyWrap section is designed to contain the main content, and the FooterWrap section

## Hands on Activity: CREATING EXTERNAL STYLES

You create styles in an external Stylesheet using the same syntax that you would use to create a style in an Internal Stylesheet.

1. Create the following styles in the file named **main.css**.

body {

height: 100%;

background-color: #376ba1;

color: white;

width: 1024px;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

padding-left:10px;

border-top: solid;

border-top-width: 3px;

border-top-color: white;

border-left: solid;

border-left-width: 3px;

border-left-color: white;

margin-top: 10px;

}

li {

list-style:none;

text-indent:0px;

display:inline;

}

a {

color:white;

}

p {

margin-top:0px;

padding-top:20px;

margin-bottom:0px;

padding-bottom:10px;

}

.BottomBorder {

border-bottom: solid;

border-bottom-width: 1px;

border-bottom-color: white;

}

#NavWrap {

font-size: 18pt;

background-color: #002266;

color:white;

padding-left:5px;

}

#BodyWrap {

background-color:#002b80;

color:white;

padding-left:5px;

}

#FooterWrap {

text-align: right;

background-color: #ccddff;

color: navy;

padding-right: 10px;

}

1. Test the page to make sure that styles are being applied as you would expect. Take a close look at these styles and try to understand how they are working together to create a well-formatted page.
2. Explore changing some of the padding and other sizes. Try changing colors too.

# Formatting Tables

CSS is well-designed for format tables. When styling tables, there are styles that you apply to the table itself. Other styles are generally applied to the table rows and finally, some styles apply to individual cells.

In this next sequence of steps, you will apply various styles to the same table so as to end up with a format that has alternating shading and rows that will be highlighted as the user hovers over them.

## Hands-On Activity: CREATING A TABLE

In this sequence of steps, you will create the table that you will format in subsequent steps.

1. In the same Web site with which you have been working, create a new Web Page named **MyTable.html** and add verbatim, the following content to the page:

<body>

<table>

<thead>

<tr>

<td>Name</td>

<td>E-Mail</td>

<td>Phone</td>

</tr>

</thead>

<tbody>

<tr>

<td>Joe Beets</td>

<td>joe@beets.unr.edu</td>

<td>775-555-1212</td>

</tr>

<tr>

<td>Mary Deems</td>

<td>mary@deems.unr.edu</td>

<td>775-555-1313</td>

</tr>

<tr>

<td>Zaphod Beeblebrox</td>

<td>zaphod@beeblebrox.unr.edu</td>

<td>775-555-1314</td>

</tr>

<tr>

<td>Mary Deems</td>

<td>mary@deems.unr.edu</td>

<td>775-555-1313</td>

</tr>

</tbody>

</table>

</body>

1. Create an embedded style in the page as you have been doing.

## Hands-On Activity: FORMATTING THE BODY AND TABLE

In this next sequence of steps, you will format the body and table elements as follows:

body {

width: 1024px;

margin: auto;

}

table {

border: solid;

border-width: 1px;

margin-bottom: 10px;

width: 80%;

text-align: right;

font-family: Arial;

}

* The body is rendered as a 1024px wide box centered in the browser window.
* The table has a solid border drawn around it. This style does not cause the border to be drawn around the cells themselves.
* The cell contents will be aligned along the right side of the cell. Fonts and border values are also set here at the table level.

1. Create the above styles in the **MyTable.html** page.
2. View the table in Chrome. Again, take a look at the table element in the Chrome Debugging window.  
     
   At this point. The table does not look very good. It’s centered in the page but there is no formatting applied to the rows and cells.

## Hands-On Activity: FORMATTING THE TABLE HEADER

The selectors that you use to select table rows and cells will vary based on how the table elements are built – Whether or not <thead> and <tbody> tags are used, for example.

The following CSS:

thead tr {

background-color: lightgray;

height: 40px;

}

is designed to format elements having the following structure:

<thead>

<tr>

<td>Name</td>

<td>E-Mail</td>

<td>Phone</td>

</tr>

</thead>

That is, the selector gets the <tr> elements enclosed in a <thead> tag and sets the background color and height of the row. Thus, the thead tr selector is a descendent selector. Descendants need not be immediate descendants to math the rule.

1. Add the above style to your <style> block and view the page in Chrome again. The title row should be formatted with a taller row and a lightgray background.

## Hands-On Activity: FORMATTING BORDERS (1)

Borders are typically applied to cells. In these following steps, you will add the following styles to format the cells.

td {

border: solid;

border-width: 1px;

border-color: navy;

padding: 10px;

}

The above style puts a 1px solid navy border around the cells. By setting the padding, you are adding whitespace between the content and the cell walls (inside of the border)

1. Add the above style to your <style> block and view the page in Chrome again.   
     
   The cells should have borders around them. The cells are surrounded with a 1px navy solid border. However, since the border surrounds the cells, they appear to as double borders in the grid. You will get rid of these double borders in the next steps.

## Hands-On Activity: FORMATTING BORDERS (2)

In this next sequence of steps, you will get rid of the double borders around the cells. To do so you set the following style for the <table>.

border-collapse: collapse;

1. Add the above style to the table style you have already created. That is, add it to the existing style for the <table> element.  
     
   When you view the page again, the double borders should no longer appear in the cells.

## Hands-On Activity: FORMATTING ALTERNATING ROWS

The idea of alternating rows applies to any sibling element that **repeats**. Different formatting for alternating rows is commonly applied to table rows and rows in a list although you can use them for other structures. The following pseudo-classes are commonly used with alternating rows:

* tr:nth-child(even) selects even siblings (table rows)
* tr:nth-child(odd) selects odd siblings (table rows)
* tr:nth-child(*n*) where n is a literal value, selects the nth sibling(table row)
* tr:first-child selects the first child (table row)
* tr:last-child selects the last child (table row)

So the following style will display the even rows with an aliceblue background color:

tr:nth-child(even) {

background-color: aliceblue;

}

1. Add the above style to the table style you have already created. When you view the page again, the alternating rows should appear shaded.

## Hands-On Activity: FORMATTING HOVER

Formatting hover is just a matter of applying the hover pseudo-class to the table row as follows:

tr:hover {

background-color: yellow

}

1. Add the above style to the table style you have already created. When you view the page again, and move the cursor from row to row, the current row will appear highlighted.

## Hands-On Activity: FORMATTING LISTS

You can apply the same idea of formatting alternating rows and hover to lists. In fact, they work exactly the same way as table rows:

1. Add the following list to end of the MyTable.html file.

<ul>

<li>This is the first list item.</li>

<li>This is the second list item.</li>

<li>This is the third list item.</li>

<li>This is the fourth list item.</li>

<li>This is the fifth list item.</li>

</ul>

1. Now add the following styles.

li:nth-child(odd) {

color: #777;

}

li:nth-child(even) {

color: blue;

}