**IS 360  
CSS LAB 1  
EKEDAHL**

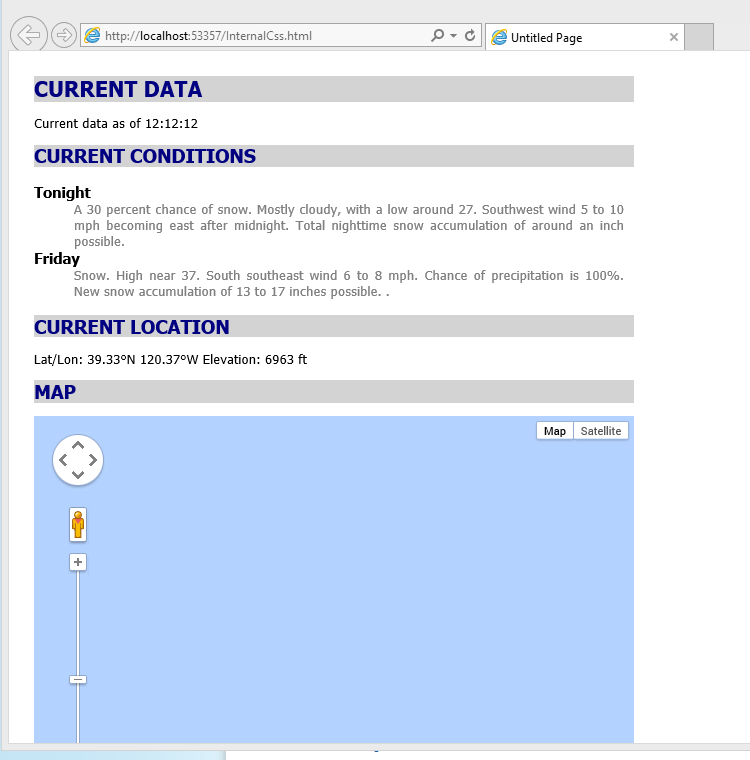
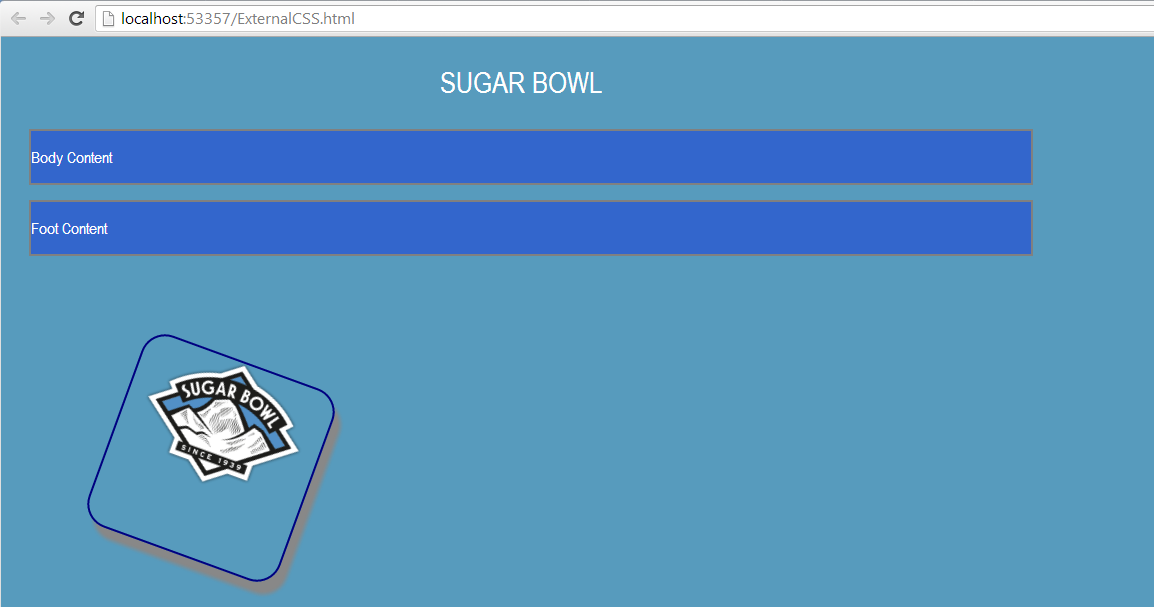
In this lab, you will work with Cascading Style Sheets. You will perform the following tasks:

* Work with internal and external style sheets.
* Format text.
* Have some fun by calling Google Maps. I’ll give you the JavaScript code. This is a prelude to what we will be doing next.

For all of the following problems, I suggest that you use the HTML 5 DTD.  
  
<!DOCTYPE html>

# Lab Abstract

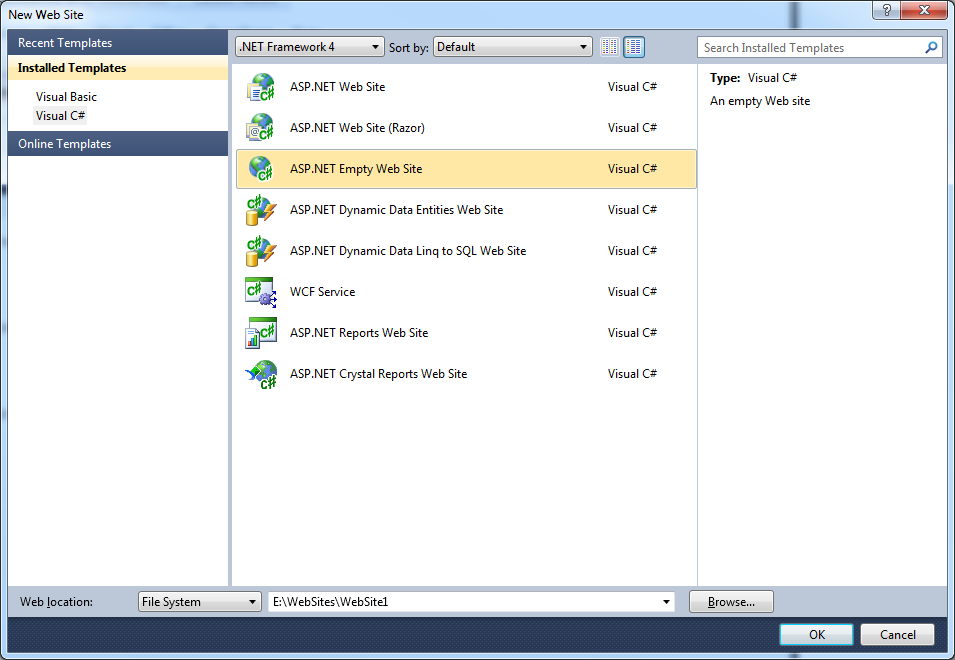
This lab has two sections.

* In the first section, you will use an internal style sheet to create a formatted Web page containing a sample weather report. At the end of this first section, you should end up with a Web page that will display formatted content using appropriate semantic tags, and display a Google map. Your page should resemble the following:  
    
  
* In the second section, you will create an external style sheet and reference it from another Web page. This second page is interesting as it demonstrates the effect of cascading styles. This page, when complete, should have the following appearance:  
    
  

# PART 1: INTERNAL STYLES

In this first exercise, you will work with internal styles and format text using CSS. The first step is to create a new HTML 5 page as you have done before.

## Hands-On Activity: CREATE AN INTERNAL STYLE

1. Start Visual Studio 2013 and create an empty Web Project. **File, New, Web site.**Select **ASP.NET Empty Web Site**.  
     
   
2. Create a blank HTML 5 page. (**Web site, Add New Item**). In the Add New item dialog box, select **HTML Page**. Name the page **InternalCss.htm**.
3. Create a <style> block inside of the <head> section of the HTML 5 page. See the example in the next step.  
     
   <!DOCTYPE HTML>  
   <html xmlns="http://www.w3.org/1999/xhtml" >  
   <head>  
    <title>Untitled Page</title>  
    **<style type="text/css">  
    </style>**</head>  
   <body>  
     
   </body>  
   </html>
4. Create the <body> section as follows. Note that I have used the semantic article, and section tags to structure the document. ***I’m not sure if the definition list is the most appropriate structure?***

<body>

<article>Current Data</article>

<p>Current data as of <time>12:12:12</time></p>

<section>Current Conditions</section>

<dl>

<dt>Tonight</dt>

<dd>

A 30 percent chance of snow. Mostly cloudy, with a low around 27.

Southwest wind 5 to 10 mph becoming east after midnight. Total   
 nighttime snow accumulation of around an inch possible.

</dd>

<dt>Friday</dt>

<dd>

Snow. High near 37. South southeast wind 6 to 8 mph. Chance of   
 precipitation is 100%. New snow accumulation of 13 to 17 inches   
 possible.

</dd>

</dl>

<section>Current Location</section>

<p id="Lat">Lat/Lon: 39.33&deg;N 120.37&deg;W Elevation: 6963 ft</p>

<section>Map</section>

<p></p>

<div id="map-canvas"></div>

</body>

Be careful that your tag syntax is correct. Otherwise, the CSS will not be applied correctly.

## Hands-On Activity: CREATE ELEMENT FORMATS

In this part of the lab, you will create the formatting for the <article> and <section> tags. By using element selectors, all articles and sections will be formatted the same way.

All of the following CSS styles must appear between the **<style>** tags that you previously created.

1. The outermost nested tag is the body. The following style will indent the entire document 25 pixels from the top and left margin of the browser window:

body

{

padding-left:25px;

padding-top:25px;

}

1. View the page in the browser. You should see all off the text indented from the top and left margins.
2. Next, create the formatting for the <article> and <section> styles. Create these immediately after the styles that you just created.

article

{

width:600px;

background-color:lightgray;

font-family: Tahoma;

font-size: 16pt;

font-weight: bold;

color: navy;

text-transform:uppercase;

}

section

{

width:600px;

background-color:lightgray;

font-family: Tahoma;

font-size: 14pt;

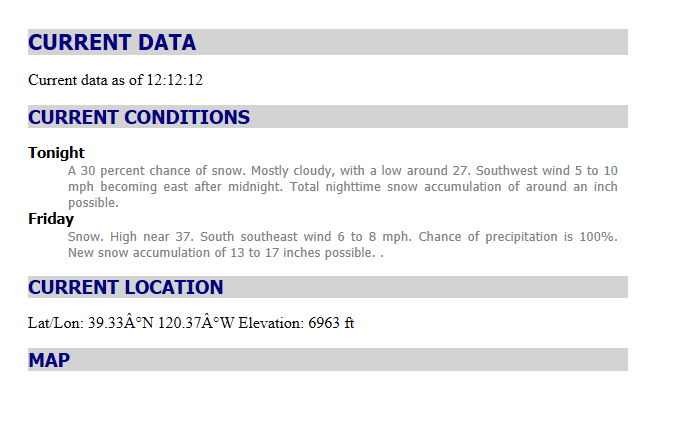
font-weight: bold;

color: navy;

text-transform:uppercase;

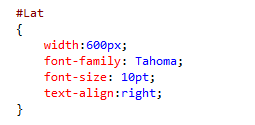
}

1. Next, create the formatting for the <dt> and <dd> tags. Use an **11pt bold, Tahoma** font for the <dt> tag. Use a **10 point Tahoma** font for the <dd> tag. Also, set the foreground color (text) to gray, the width to 550 pixels, and justify the text. It is up to you to create the styles properly.
2. Test the Web page. At this point, your page should resemble the following:



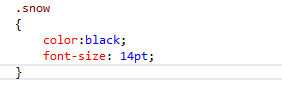
## Hands-on Exercise: CREATING A STYLE FOR AN ID.

There is a one-to-one relationship between an element, its ID, and corresponding CSS. That is, the value of an element’s ID attribute must be the same from one page to the next. In this example, you will modify the paragraph describing the current location to use an ID attribute.

1. Modify the paragraph tag in the Current Location section as follows such that it uses the ID attribute. Note that you are setting the ID attribute to “Lat”  
     
   
2. Add the following ID to the end of the <style> with which you have been working. Here, you are defining a style for the attribute named **Lat**. Remember that ID styles begin with a hash tag.  
   
3. View the page in a browser. You should see that the paragraph is right justified in the 600 pixel wide box.

## Hands-on Exercise: CREATING AND USING A CLASS

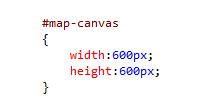
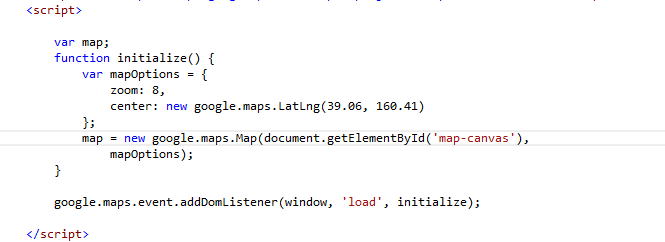
So far in this lab, you have styled elements and an ID attribute. Next you will work with classes. Working with CSS classes is a two-step process. You create the CSS class itself. CSS class names begin with a period (.). To use the class, add the class attribute to the respective tag:

1. At the end of the <style> section with which you have been working, create the following class:  
     
   
2. Modify the two paragraphs in the Current Conditions section so that the word snow appears between <span> tags. Here you are seeing the inline <span> tag used to mark a block of text. Other than that, the tag does nothing.   
     
   A 30 percent chance of <span class="snow">snow</span>.
3. The word snow should be formatted. Note that not all browser and browser versions fully support text shadowing.

## Hands-on Activity: A FIRST BIT OF JavaScript

The following is not so much related to CSS but does lead us in to the topic of JavaScript. In this exercise, you will use a <div> tag to create a placeholder for a Google map.

A <script> tag contains JavaScript just as a <style> tag contains CSS. Both appear inside of a document’s <head> tag. Note that JavaScript can appear elsewhere in a document but more about later.

1. Create the following <div> tag that will act as a placeholder for the map:  
     
   <div id="map-canvas"></div>
2. Create the following ID style in the <style> section with which you have been working.  
   
3. Create the first <script> block by enter the following code just after the end of the <style> block with which you have been working.  
     
   
4. Create a second <script> block with the following code:  
     
   

The JavaScript code you just wrote does the following:

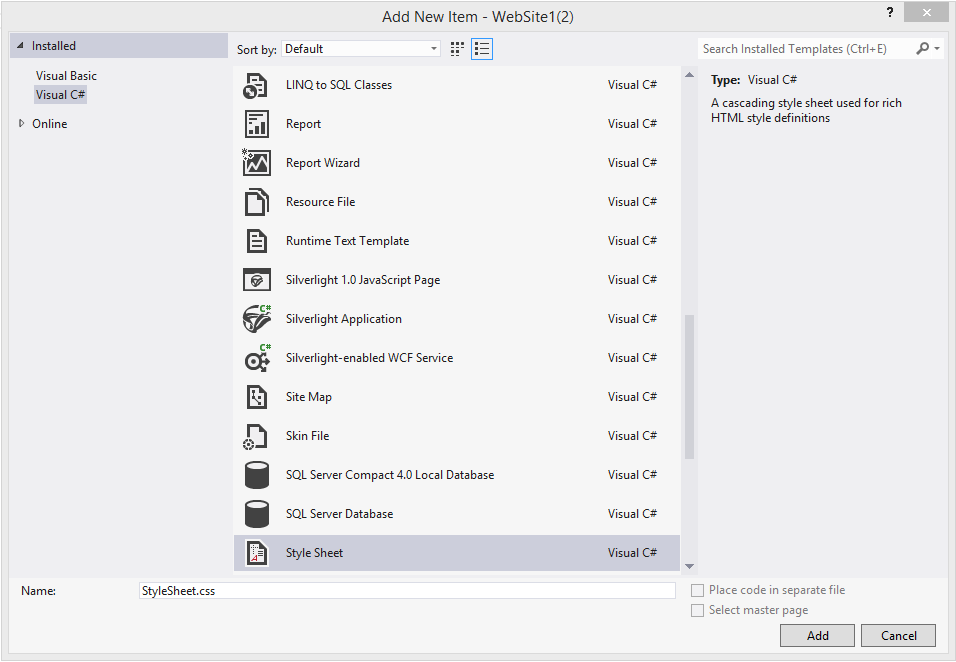
* The first script tells the system to read the JavaScript code stored at the specified URL into you your page, and execute that code as if it were in the page.
* In the second script, the var keyword declares variables and the function keyword creates a function. Conceptually, this is no different than variables and functions in VB.
* The following statement gets the <div> element by its ID value (document.getElementByID). Again, this is why the ID attribute is important. It allows you to reference an element with code.
* The following statement registers and event handler. google.maps.event.addDomListener(window, ‘load’, initialize). Conceptually, this is similar to a VB event. When the document window loads in the browser window (load event), the initialize function is called thereby initializing and rendering the Google map.

1. Run the program. You should see the map.

# PART 2: EXTERNAL STYLES

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

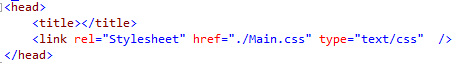
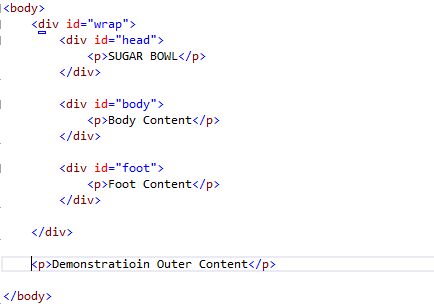
## Hands-on Activity: CREATING AN EXTERNAL STYLE SHEET.

1. Create a new empty style sheet. On the menu click **Web site**, **Add New Item**. From the dialog box, select StyleSheet.css.
2. Change the name to Main.css:  
     
   

In the above figure, you can see that Visual Studio created the .css file and added a template style for the <body> tag.

## 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 style sheet file.

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.  
     
   
3. Finally, create the following body text:  
     
   

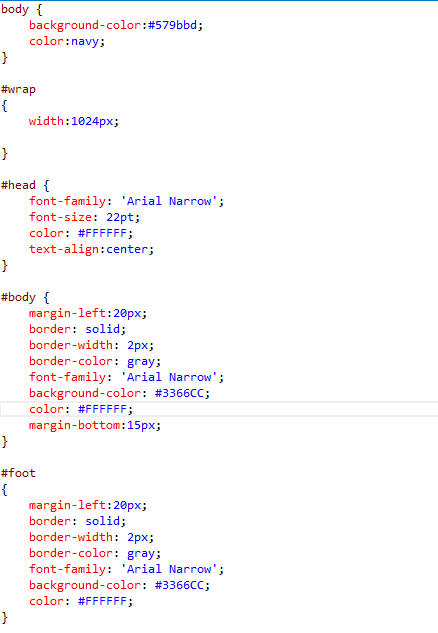
This text is very simple but does demonstrate some important points:

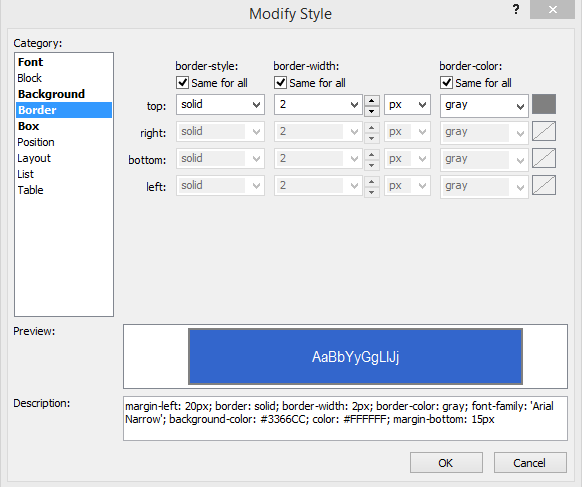
* <div> tags are nested and the content (paragraphs) are nested inside of the nested <div> tags. Thus, the formatting will be applied to the paragraphs, based on the formatting of the parent <div> tag.
* The final paragraph appears outside of the outer “wrap” <div> tag. It will use any formatting applied to the body, but not formatting applied to the “wrap”, “head”, or “foot”.
* In other words, styles cascade down to their child elements.

## Hands-on Activity: CREATING A PAGE LAYOUT STYLE

You have created the style sheet, the main page, and the link to the style sheet. Next. You will create the styles themselves. You can enter the rule in the editing window, just as you entered a style in the <style> section of the HTML 5 page.

1. Create the following styles in the style sheet named **Main.css**.

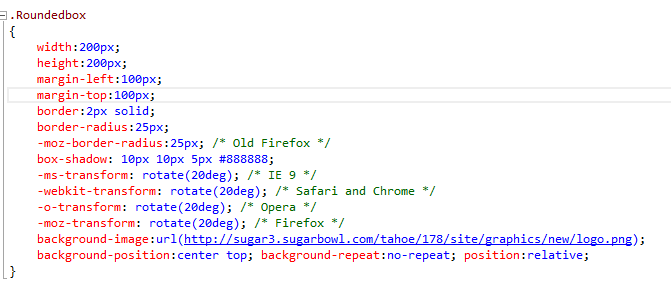


1. 
2. Close the dialog box.
3. Display the Web page.

## Hands-on Activity: ENHANCING THE BOXES

In this next exercise, you will continue to work with external style sheets and perform more detailed formatting using some of the new CSS3 features.

1. Using the same external style sheet and Web page, create a class named RoundedBox. Again, I’ll give you the style but make sure that you understand it.



1. On the correspond page, create the <div> tag to display the box. It should contain the logo in a box rotated 30 degrees  
     
   <div class="Roundedbox"></div>