WEB222 - Web Programming Principles

Week 8: Introduction to CSS

Agenda

- Introduction to CSS
  - Syntax / Structure
  - Selectors
  - Web Colors, Units
- CSS Properties / Values
  - Text, Fonts
  - Text Effects
  - Lists, background, Links
- Updating CSS using the DOM

Introduction to CSS

- CSS stands for Cascading Style Sheet
- CSS is a stylesheet language used to describe the presentation of a document written in HTML (or XML).
- HTML: used to define structure and content of an HTML document/web page
- CSS: used to affect and modify the appearance and formatting of a document

Introduction to CSS

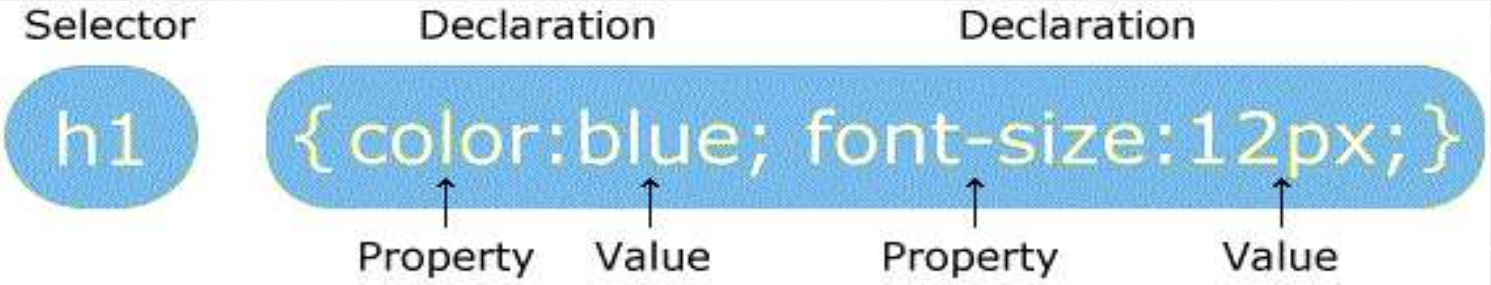
- CSS are really text files, or text in an HTML file , that allow the use of specific styles, attributes, and positioning of HTML objects.
- CSS Separates the style and layout from the content. And CSS could be removed from HTML doc, and stored in a separate CSS file.
- External Style Sheets in CSS files can save a lot of work. It controls the style and layout of multiple web pages all at once.
- All browsers support CSS today.

Advantages

- Define the look of your pages in one place, and apply it throughout the whole site.
- Easily change the look of your pages even after they're created. Change style once.
- Pages will be loaded faster, since they aren't filled with tags that define the look. The style definitions are kept in a single CSS document that is only loaded once when a visitor enters your site.

CSS Syntax / Structure

- The CSS syntax is made up of three parts
  - a selector
  - a property
  - a value
- The syntax of a CSS Rule / Entry:



CSS Syntax / Structure

- The selector properties and values are always enclosed between **{** and **}**
- The property is separated from its value or values by a colon **:**
- The values, if more than one are separated by a comma **,**
- A CSS declaration always ends with a semicolon **;**

CSS Syntax / Structure

- The selector is typically the HTML element(s) that you want to style.
- Each declaration consists of a property and a value.
- The property is the style attribute you want to change.
- Each property has a value.

CSS Example

```
p { color: red; text-align: center; text-decoration: underline; }
```

- To make the CSS more readable, you can put one declaration on each line, like this:

```
p {
    color: red;
    text-align: center;
    text-decoration: underline;
}
```

- These will affect all paragraphs (<p> elements) in web page that the css rules are applied.

Where to place your CSS

- CSS can be implemented/added in an html document in three different ways:
  - Inline
  - Internal Embedded
  - External
- In addition, each browser has it Browser default CSS settings

Implementing CSS in HTML

1. Browser default

   Rules are set by the browser for the various tags

2. Inline

   CSS rule is applied on a element using style attribute, e.g.

```
<p><img src="logo.png" alt="Seneca College"
          style="width:195px;height:43px"></p>
```

3. Internal Embedded

   CSS rules are included in the head part of an html document and can be used anywhere in the html document, e.g.

```
<style>
    p { background-color: yellow; }
</style>
```

- css_internal_inline.html

Implementing CSS in HTML

4. External

CSS rules are in a separate CSS file referenced from any html document using the html <link...> tag

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

or the @import CSS feature

```
<style type='text/css' media='screen'>
  @import url(http://www...../company.css);
</style>
```

- css_external.html

CSS – Multiple Style Sheets

- The priority order for CSS sources
  - If a CSS property of an element has different values in different CSS rules, the priority order will be:
    (from highest to lowest)
    Inline, Internal Embedded, External, Browser default

e.g., external CSS:

```
h3 {
   color:red;
   text-align:left;
   font-size:8pt;
}
```

e.g., Internal CSS:

```
h3 {
   text-align:right;
   font-size:20pt;
}
```

Final results for h3:

```
color:red;
text-align:right;
font-size:20pt;
```

CSS Cross-browser Consistency

- CSS reset and normalization
  - CSS reset: reset.css (resets the browser defaults)
  - CSS Normalization: normalize.css (attempts to provide better cross-browser consistency of the default styling)

- Demos

  consistency-default.html

  consistency-reset.html

  consistency-normalize.html

Basic CSS Selectors

- Selectors:
  - HTML tag Selectors
  - Class Selectors
  - Id Selectors
  - Contextual Selectors
  - Grouping Selectors
  - …

Tag Selectors

- HTML tag/type Selectors
  - Any html tag is a possible CSS selector. The selector is simply the tag that is linked to a particular style, e.g.

```
p { text-indent: 20px; }
```

  - the selector in above example is the p tag

Class Selectors

- A class selector matches elements based on the values of their class attribute.
  - e.g.

    ```
    .type1 { text-decoration:underline; }
    ```

  - The class selector ".type1" matches both of the following elements:

    ```
    <h5 class="type1  type2" >.....</h5>
    <p class="type1" >.....</p>
    ```

- The class attribute can have same value for different elements within an HTML document.

Class Selectors

- A class selector can also be used with the tag it is associated with – e.g. the p tag:

```
/* all paragraphs with class value "note" matches */
p.note { font-weight: bold; }
```

- In this case, only the first <p> element of the followings matches:

```
<p class="note" >.....</p>
<h3 class="note" >.....</h3>
<p>.....<span class="note" >.....</span>.....</p>
```

id Selectors

- The id selector matches an element based on the value of its id attribute.
- An id selector is assigned by using the indicator "#" to precede an id value, e.g.

```
#xyz656 { background-color: skyblue;}
```

or:

```
div#xyz656 { text-indent: 1em; }  /* unusual */
```

HTML:

```
<div id="xyz656" ></div>
```

- The value of id attribute must be unique in an HTML document.

Contextual Selectors

- Contextual selectors are used to indicate the context of a selector. Also called descendant combinator.
- The context of a selector is determined by what its parent element is. In other words, what the element is nested within or what precedes it in the document.
  - For example, if you want unordered lists that are nested under ordered lists to have a font size of 16px, you can use

```
ol ul { font-size: 16px; }
```

- This can be read as "for any unordered list that is nested within an ordered list" - change the font size to 16px.

Grouping Selectors

- To reduce the size of the style sheets, one can group selectors in comma-separated list:

```
h1, h2, h3 { font-family: Serif; color: blue;}
```

- Example:
  - css_selectors.html

CSS how to -

- The HTML element defines general container for flow content and does not inherently represent anything.
- It is a block-level element and can contain nearly any other tag. It cannot be inside <p> tags.

- You can use the tags to group elements for the purposes of styling (using the class or id attributes), e.g. when you want to center or position a content block on the page.
- It's always a good idea to close tags as soon as you create them. Then place the contents within the element.
- If you nest tags, make sure that you know where your content is going (in other words, which div it should be part of) – correct indentation will help with this.

CSS how to -

- Similar to the element in function with one difference - <*span*> is an inline element, i.e. it does not begin a new line.
- can only contain or other inline elements.
- The *span* element can be a selector for the purposes of styling using *style*, *class* and *id* attributes.
- The primary difference between the and tags is:
  - tags include paragraphs, because it is defining a logical division in the document.
  - tags are usually included in paragraphs and simply tells the browser to apply the style rules to whatever is within the tags.

- css-group-tags.html

Units used in CSS

1em = 12pt = 16px = 100%

- *Ems* - em
  - The "em" is a scalable unit that is used in web document media. An em is equal to the current font-size, for instance,
    - if the font-size of the document is 12pt, 1em is equal to 12pt.
  - *Ems* are scalable in nature, so
    - 2em would equal 24pt,
    - .5em would equal 6pt, etc.
  - *Ems* are becoming increasingly popular in web documents due to scalability and their mobile-device-friendly nature.
  - em stands for "M", the letter M being the widest character in a font.

Units used in CSS

1em = 12pt = 16px = 100%

- *Points* - pt
  - *Points* are traditionally used in print media (anything that is to be printed on paper, etc.).
  - One *point* is equal to 1/72 of an inch.
  - *Points* are much like pixels, in that they are fixed-size units and cannot scale in size.

Units used in CSS

1em = 12pt = 16px = 100%

- Pixels - **px**
  - Pixels are fixed-size units that are used in screen media (i.e. to be read on the computer screen).
  - One pixel is equal to one dot on the computer screen (the smallest division of your screen's resolution).
  - Many web designers use pixel units in web documents in order to produce a pixel-perfect representation of their site as it is rendered in the browser.
  - One problem with the pixel unit is that
    - it does not scale upward for visually-impaired readers
    - or downward to fit mobile devices.

Units used in CSS

1em = 12pt = 16px = 100%

- *Percent* - **%**
  - The *percent* unit is much like the "em" unit, save for a few fundamental differences.
    - First and foremost, the current font-size is equal to 100% (i.e. 12pt = 100%).
    - While using the percent unit, your text remains fully scalable for mobile devices and for accessibility.

- font-units.html

Units used in CSS

- The other ways of measurement
  - xx-small, x-small, small, medium, large, x-large, xx-large,
  - smaller, larger
  - thin, medium, thick
- Viewport-percentage lengths: vw and vh units
  - The viewport-percentage lengths are relative to the size of the initial containing block.

    vw unit - equal to 1% of the width of the initial containing block.

    vh unit - equal to 1% of the height of the initial containing block.
  - e.g.

    ```
    body { width: 80vw; }
    main { width: 90%; min-height: 100vh; }
    ```

Web colors

- Primary colors are sets of colors that can be combined to make a range of colors.
  - Normally, red, green and blue are used as primary colors - the RGB (Red-Green-Blue) color model.
- CSS colors are specified in 3 formats:
  - Hexadecimal Value Notation
    - Hex triplet: written as 3 double digit numbers, starting with a # sign.
    - e.g. h1 { background-color: #800080; }
  - RGB Value Notation
    - the combination of Red, Green, and Blue color values (RGB).
    - e.g. P { color: rgb(128,0,128);
      background-color: rgba(0, 191, 0, 0.5); } /*color with opacity*/
  - Named colors

Color Examples

Color (Named)	Color HEX	Color RGB
Black	#000000	rgb(0,0,0)
Red	#FF0000	rgb(255,0,0)
Green	#00FF00	rgb(0,255,0)
Blue	#0000FF	rgb(0,0,255)
Yellow	#FFFF00	rgb(255,255,0)
Aqua	#00FFFF	rgb(0,255,255)
Fuchsia	#FF00FF	rgb(255,0,255)
Gray	#808080	rgb(128, 128, 128)
Silver	#C0C0C0	rgb(192,192,192)
Gold	#FFD700	rgb(255,215,0)
White	#FFFFFF	rgb(255,255,255)

CSS Properties and Values

- A property is assigned to a selector in order to manipulate its style.
  - Examples of properties include color, margin, font and many more.

    p { **text-indent**: 1em; }
- A property can have one or more value. Values must be spelled exactly as described in the CSS rules.

  #cnt2 span { text-decoration: underline overline;}

  p { font-family: **"Times New Roman", Serif**; }
  - Times New Roman and serif in the above are two value examples for the font-family.
  - Multiple words for any value must be in quotations
- There are large number of properties and their values. We cannot cover all of them in the courses.

CSS text Properties:

- Examples

```
h2 {
  color: red;
  text-align: center; /* or left, right*/
  text-decoration: underline; /* or overline, line-through */
}

p.first { color: #0000ff; text-indent:100px; }

#num2 {
  color: rgb(255,128,0);
  text-transform: capitalize;  /* or uppercase, lowercase */
}
```

  - text.html

CSS font Properties:

- Examples

```
h3 { font-family: "Times New Roman", Times, serif; }
h4 { font-family: Arial, Helvetica, sans-serif; }
h5 span {
   font-size: 18pt; /* or 1.5em, 24px, 150% */
   font-style: italic; /* or oblique, normal */
   font-weight: 500; /* 100, ..., 900, normal(400) , lighter, bolder, bold(700) */
   font-variant: normal; /* small-caps; */
}
p span { font: italic small-caps bolder 1.5em "Lucida","Arial"; }
```

  - font.html

The font-family Property

- A font family is a list of one or more fonts that will be applied by a web browser to some text.
- The font family can use a specific named font, but the actual appearance will depend on the browser and the fonts installed on the system.
  - e.g., a default installation of I.E. always displays serif and Times as Times New Roman, and *sans-serif* and *Helvetica* as Arial.
- A font-family consists of a set of related fonts, grouped as font families

The font-family Property

- The web browser will only be able to apply a font if it is available on the system on which it operates, which is not always the case.
- So, list of font families in preferential order is used when rendering text.
- The font family list is separated by commas.
- To avoid unexpected results, the last font family on the font family list should be one of the five generic families which are by default always available on the system.

The 5 Generic Font Families

Family	Example (browser dependent)
serif	The quick brown fox jumps over the lazy dog. 0123456789
sans-serif	The quick brown fox jumps over the lazy dog. 0123456789
cursive	The quick brown fox jumps over the lazy dog. 0123456789
fantasy	The quick brown fox jumps over the lazy dog. 0123456789
monospace	The quick brown fox jumps over the lazy dog. 0123456789

- These font families are by default always available for all browsers

CSS Web Safe Font Combinations

Serif Fonts

font-family	Example text
Georgia, serif	**This is a heading** This is a paragraph
"Palatino Linotype", "Book Antiqua", Palatino, serif	**This is a heading** This is a paragraph
"Times New Roman", Times, serif	**This is a heading** This is a paragraph

Monospace Fonts

font-family	Example text
"Courier New", Courier, monospace	**This is a heading** This is a paragraph
"Lucida Console", Monaco, monospace	**This is a heading** This is a paragraph

"Web Fonts" - @font-face

- It is now possible to use fonts which are not necessarily installed on the client's system.
- This can be accomplished with the new @font-face CSS Rule, ie:

```
@font-face { font-family: 'FontName';
        src: url(url-to-file.woff2) format('woff2'),
             url(url-to-file.ttf) format('truetype');
             /* other font types (legacy browser support) */
}
```

- You can then use the font ('FontName'), anywhere in your CSS, ie:

```
.coolFont { font-family: 'FontName'; }
```

- For a detailed guide on the @font-face rule: https://css-tricks.com/snippets/css/using-font-face/

"Web Fonts" - Using a CDN

- The easiest, most convenient way to add a web font to your pages is to use a CDN ("Content Delivery Network")
- A popular free CDN is "Google Fonts" - https://fonts.google.com/
- For example, if we wish to use the popular "Lobster" font in our pages, we can use "Google Fonts" to supply the necessary CSS.
- From the Lobster font page, we can get the following code:

```
<link href="https://fonts.googleapis.com/css?family=Lobster" rel="stylesheet">
```

  - Which we can include in our HTML documents before our own CSS to give us access to the "Lobster" font-face
- NOTE: We can also include the provided link in our CSS files using the @import directive, ie:

```
@import url(https://fonts.googleapis.com/css?family=Lobster);
```

- font-2.html

The font-size Property

- Font size for different elements
  - h1 { font-size:250%; } – size relative to regular size (scales well)
  - p { font-size: 20pt; } – actual size in points,
  - div { font-size:20px; } – actual size in pixels,
  - a { font-size: smaller; } – smaller than regular size, default medium,
  - h1 { font-size: 1.5em; } – size relative to regular size (scales well)

font-size: Property values

Value	Description
xx-small	Sets the font-size to an xx-small size
x-small	Sets the font-size to an extra small size
small	Sets the font-size to a small size
medium	Sets the font-size to a medium size. This is default
large	Sets the font-size to a large size
x-large	Sets the font-size to an extra large size
xx-large	Sets the font-size to an xx-large size
smaller	Sets the font-size to a smaller size than the parent element
larger	Sets the font-size to a larger size than the parent element
length	Sets the font-size to a fixed size in px, cm, etc.
%	Sets the font-size to a percent of the parent element's font size
inherit	Specifies that the font size should be inherited from the parent element

CSS3 Text Effect

- Text shadow:
  - Specify horizontal shadow, the vertical shadow, the blur distance, and the color of the shadow.

```
h1
  {
    text-shadow: 5px 5px 5px red;
  }
```

- text_css3.html

CSS3 Text Effect

- CSS3 word wrapping
- If a word is too long to fit within an area, it expands outside
- In CSS3, the word-wrap property allows to force the text to wrap
- Even if it means splitting it in the middle of a word.
- E.g.

```
p.wrap { word-wrap: break-word; }
```

- text_css3.html

CSS Lists

- The list-style-type CSS property specifies appearance of a list item element.
- Examples:

```
{ list-style-type:circle; }
{ list-style-type:square; }
{ list-style-type:upper-roman; }
{ list-style-type:lower-alpha; }
```

- Default value: disc

- css_list.html

Property list-style-type Values

Value	Description	e.g.
none	No item marker is shown	
disc	A filled circle (default value)	
circle	A hollow circle	
square	A filled square	
decimal	Han decimal numbers	1, 2,
decimal-leading-zero	Decimal numbers	01, 02, 03, … 98
lower-roman	Lowercase roman numerals	i, ii, iii, iv, v…
upper-roman	Uppercase roman numerals	II, III, IV, V…
lower-greek	Lowercase classical Greek	α, β, γ…
lower-alpha, lower-latin	Lowercase ASCII letters	a, b, c, … z
upper-alpha, upper-latin	Uppercase ASCII letters	A, B, C, … Z

***More list*-style-type values| MDN**

background - Properties

- background-color: provides background color if background image is not specified or available
- background-position:
  - Values: left top (default), right bottom, center center
- background-repeat:
  - Values: repeat (default), repeat-x, repeat-y, no-repeat
- Background-image:
  - background-image: url(image.jpg);

    where image.jpg may be a relative or absolute path
- Shorthand property:

Body { background: url("seneca_logo.gif") no-repeat grey right top;}

- bg.html, bg.css

CSS3 Backgrounds

- Property "background-size": specifies the size of the background image.
- In CSS3 it is possible to specify the size of the background image, which allows us to re-use background images in different contexts.
- Resize a background image:

```
body {
    background-image : url(NiagaraFalls.jpg);
    background-size: cover; /* or 800px 600px; 100% 100%; */
    background-repeat: no-repeat;
}
```

- bg_new.html

CSS3 Backgrounds

- Property "background-origin" property specifies the positioning area of the background images.
- The background image can be placed within the content-box, padding-box, or border-box area.

- bg_new_origin.html

Styling Links

- CSS pseudo-class: is a keyword added to selectors to specify a special state of the element to be selected.
  - Syntax: Selector:pseudo-class { property: values; }
- Anchor Pseudo-classes: links are styled differently depending on what state they are in:
  - a:link - a normal, unvisited link
  - a:visited - a link the user has visited
  - a:hover - a link when the cursor hovers over it
  - a:active - a link the moment it is clicked
- Note: When setting the style for several link states,
  - ✓ a:hover MUST come after a:link and a:visited
  - ✓ a:active MUST come after a:hover
- css_link.html css_link-as-button.html

Updating CSS Using the DOM

- Now that we're familiar using CSS, why don't we make it more dynamic, using the DOM?
- Ways to update an element's CSS using DOM:
  - Use the DOM style object of an element
  - Modify the value of an element's style attribute
  - Create CSS rules using class selector, then set/add the class value to an element.

Using the DOM style object

- The style object of each element can be used to set CSS properties for the element.
- Syntax:

  element.style.*property* ="*new style*";

- e.g.

```
// highlight all paragraphs in the document
var paraElements = document.querySelectorAll("p");

for (var i = 0; i < paraElements.length; i++) {
   paraElements[i].style.backgroundColor = "yellow";
}
```

- Note:

  The style object's backgroundColor property corresponds to CSS rules' background-color property

HTML DOM Style Object

- Style Object Properties

Property	Description
backgroundColor	Sets or returns the background-color of an element
border	Sets or returns borderWidth, borderStyle, and borderColor in one declaration
borderColor	Sets or returns the color of an element's border (can have up to four values)
color	Sets or returns the color of the text
display	Sets or returns an element's display type
fontSize	Sets or returns the font size of the text
fontWeight	Sets or returns the boldness of the font
margin	Sets or returns the margins of an element (can have up to four values)
padding	Sets or returns the padding of an element (can have up to four values)
textDecoration	Sets or returns the decoration of a text
minHeight	Sets or returns the minimum height of an element
minWidth	Sets or returns the minimum width of an element
visibility	Sets or returns whether an element should be visible

Using setAttribute() method

- Syntax:

  element.setAttribute("style", new styles);

- e.g.

```
var elem = document.querySelector("#xy123");
// set multiple style properties in one statement
elem.setAttribute("style", "width:80px;background:blue;");
```

- Advanced:

  Some jQuery Functions And Their JavaScript Equivalents

Manipulating class attribute values

- With predefined CSS rules using class selector(s), we can change CSS by updating class attribute values.
- This can be done using

```
element.classList.add();  // to add classes to an element
element.classList.remove();  // to remove classes from an element
```

- For example:

```
document.querySelector("#someID").classList.add("boldify");
```

  - would add the property class="boldify" to an element with id="someID".
  - If the specified class already exist, the class will not be added.

- css-dom.html

Resourceful Links

- CSS Reference

  http://reference.sitepoint.com/css
- CSS Selectors

  http://reference.sitepoint.com/css/selectorref
- CSS properties

  http://reference.sitepoint.com/css/propertyref

Thank You!