

CSS

Overview

Naveen Pete

CSS Basics

- Stands for Cascading Style Sheets
- HTML provides structure
- JavaScript provides behavior to your HTML document
- CSS provides the presentation to your HTML document
- Provides the tool to design and create great looking web apps
- CSS Goals
 - Create a consistent look across many web pages
 - Separate structure from presentation, so you can provide different style sheets for printing, browsing, or other scenarios

CSS Basics

- **Style**
 - Is a rule that describes how to format a specific part of an HTML document
- **Style Sheet**
 - Is a set of style rules
- **Selector**
 - A style can be applied to many elements based on a selector
 - Used to locate and select elements based on tag name, class name, element id, etc.
 - The idea is reuse styles across elements and pages

Defining Styles

- Defining and applying a style
 - A style is composed of two parts
 - **Selector** – locates the elements in the HTML document that will be styled
 - **Declaration block** – contains the formatting instructions or declarations
 - Multiple declarations are separated with a semicolon
 - A declaration takes the following form
 - **<css-property>: <value>**

```
selector {  
  property: value;  
  anotherProperty: value;  
}
```

Defining Styles

- Adding comments
 - Comments can be added within a style sheet by using `/*` and `*/`
 - `/*` characters start the comment
 - `*/` characters end the comment
 - Comments may span multiple lines

Defining Styles

- Creating an inline style
 - The global attribute called 'style' can be used to provide an inline style
 - Since an inline style is defined on the element to which you wish to add styling, selector is not needed; just declaration block is enough

```
<ul>  
  <li style="color: red;">Get vegetables</li>  
  <li style="color: red;">Buy milk</li>  
</ul>
```

- Avoid this technique because
 - It violates the goal of separation between structure and presentation
 - No reusability

Defining Styles

- Creating an embedded style
 - <style> element can be used to create an embedded style sheet within your HTML document
 - CSS selectors must be used to assign the style definitions to elements on the page
 - The <style> element is specified within <head> element
 - Any number of style rules can be defined within <style> element
- Avoid this technique because,
 - It does not provide file separation
 - It does not promote reuse across HTML documents

```
<html>
<head>
  <style>
    body {
      background-color: white;
      color: gray;
    }
  </style>
</head>
...
</html>
```

Defining Styles

- Creating an external style sheet
 - The ideal approach is to create an external style sheet file
 - This file should then be linked to pages using <link> element

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="./default.css" />
  </head>
  ...
</html>
```

```
/* default.css */
body {
  background-color: white;
  color: gray;
}
```

- Attributes
 - rel – specifies the relationship between HTML document and external CSS file
 - type – specifies the MIME type of external CSS file
 - href – specifies the relative location of external CSS file

Defining Styles

- Creating an external style sheet (Continued)
 - A style sheet file can have as many style rules as needed
 - More than one external style sheet files can be linked to an HTML document using <link> tag
- Using 'media' attribute
 - The <link> element has a media attribute that can specify the target device
 - You can create a CSS file for each device and link all CSS files into the HTML document
 - When the HTML document is rendered, the browser determines the media type and uses the appropriate CSS file

```
<link rel='stylesheet' type='text/css' href='../screen.css' media='screen' />  
<link rel='stylesheet' type='text/css' href='../print.css' media='print' />
```

Defining Styles

- Creating an external style sheet (Continued)
 - Imported style sheets from other style sheets
 - The @import rule enables you to import a CSS file into another style sheet file

```
@import url('../header.css');  
@import url('../menu.css');  
  
body {  
    background-color: white;  
    color: gray;  
}
```

- **Note:** @import rules must be at the top of the style sheet

CSS Colors

- Colors can be specified in several ways
 - Color names
 - Built-in colors, HTML supports 140 standard color names
 - <https://htmlcolorcodes.com/>
 - <http://colours.neilorangepeel.com/>
 - Hexadecimal
 - # + String of 6 Hexadecimal numbers (from 0 to F)
 - #800080 – for purple
 - #FFFFFF – for ivory
 - More choices, keeping the names of colors short
 - <https://www.webpagefx.com/web-design/color-picker/>
 - rgb()
 - 3 channels: Red, Green, and Blue. Each ranges from 0 to 255
 - rgb(0, 255, 0)
 - rgb(100, 0, 100)

CSS Colors

- `rgba()`
 - Similar to `rgb()`, but with an alpha (transparency) channel that ranges from 0.0 to 1.0
 - `rgba(11, 99, 150, 1)`
 - `rgba(11, 99, 150, 0.6)`
 - `rgba(11, 99, 150, 0.2)` */* more transparent */*
- `hsl()`
 - A color can be specified using hue, saturation, and lightness (HSL) in the form

```
hsl(hue, saturation, lightness)
```

 - Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue
 - Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color
 - Lightness is also a percentage, 0% is black, 50% is neither light or dark, 100% is white

Color & Background

- Use

- 'color' to set text color
- 'background-color' to set background color

```
p {  
  color: tomato;  
}  
  
h1 {  
  background-color: DodgerBlue;  
}
```

- 'background-image' specifies an image to use as the background of an element

```
body {  
  background-image: url('paper.gif');  
  background-repeat: no-repeat;  
  background-size: cover;  
}
```

Borders

- ‘border-color’ property is used to set the color of the four borders
- ‘border-style’ property specifies what kind of border to display
- ‘border-width’ property specifies the width of the four borders

```
p {  
    border-width: 5px;  
    border-style: solid;  
    border-color: red;  
}
```

- ‘border’ property is a shorthand property for the following individual border properties:
 - border-width
 - border-style (required)
 - border-color

```
/* Border - Shorthand Property  
p {  
    border: 5px solid red;  
}
```

Borders

- 'border-radius' property is used to add rounded borders to an element

```
p {  
    border: 2px solid red;  
    border-radius: 5px;  
}
```

CSS Selectors

- A selector can be one of the following:
 - Element selector
 - Id selector
 - Class selector
- **Element selector**
 - Is based on the name of the tag
 - Select all instances of a given element

```
li {  
    border: 2px solid red;  
}
```


CSS Selectors

- **Id selector**

- Is based on the id of the element
- Usually prefixed with the hash (#) symbol
- Selects an element with a given id
- Id must be unique across the HTML document, and hence this approach limits the reusability on a page

```
#special {  
    background-color: yellow;  
}
```

- **Class selector**

- Is a style with a class name of your choice
- Prefixed with the period (.) symbol
- Also called a named style
- Class name can be assigned to any element using the class attribute
- Promotes reuse

```
.completed {  
    text-decoration: line-through;  
}
```

Advanced Selectors

- Universal selector

- If you want to apply a style to every element, you can use asterisk (*) symbol
- Avoid using the universal selector because of the performance cost

```
* {  
  border: 1px solid lightgrey;  
}
```

- Descendant selector

- You might want to change the style of elements only if the elements are descendants of another element

```
li a {  
  color: red;  
}
```

- “li a” is called a ‘selector chain’

- A group of selectors that specify a path to the element that interest you
- Specifies an ancestor element, followed by a space, and then the descendant element
- The descendant element can be a child, grandchild or distant descendant

Advanced Selectors

- Child selector

- You might want to change the style of elements only if the elements are direct children of another element
- You can do this by specifying a parent element, followed by a greater-than symbol (>) and then specifying the child element

```
li > a {  
    color: red;  
}
```

- Adjacent selector

- Can be used to select an element if it is preceded by a specific element
- The plus sign (+) denotes an adjacent selector

```
h4 + ul {  
    border: 4px solid red;  
}
```

Advanced Selectors

- Attribute selector
 - Selects elements based on the existence of the specified attribute
 - For e.g., `a[title]` selects all hyperlinks whose title attribute is defined
- Attribute value selector
 - Selects all elements where the specified attribute has the specified value

```
a[href="http://www.google.com"] {  
  background-color: blue;  
}
```

- `nth-of-type(n)` selector

```
li:nth-of-type(odd) {  
  background-color: yellow;  
}
```

- Selects every `` element that is the second `` element of its parent

Specificity

- Inheritance
 - An element can get its style from a parent when no other styles are defined that are more specific
- Specificity
 - Is the means by which browsers decide which CSS property values are the most relevant to an element and, therefore, will be applied
 - Certain times, there could be multiple styles that would impact a single element
 - In such scenario the CSS has to decide which one wins
 - Whatever style is closest to the element, i.e., whichever style is more specific, it wins and is considered

Specificity

- The following list of selector types increases by specificity:
 - Type selectors and pseudo-elements
 - E.g. `h1`, `::before`
 - Class selectors, attributes selectors and pseudo-classes
 - E.g. `.header`, `input[type="radio"]`, `:hover`
 - ID selectors
 - E.g. `#example`
- **Note**
 - Inline styles added to an element (e.g., `style="font-weight:bold"`) always overwrite any styles in external stylesheets, and thus can be thought of as having the highest specificity

Fonts & Text

- ‘font-family’ property can be used to set the typeface of the elements that match the selector

```
p {  
  font-family: arial;  
}  
  
h1 {  
  font-family: "times new roman", serif;  
}
```

- ‘font-size’ property can be used to set the size of the font
 - Font sizes can be specified by using
 - Absolute units – useful when the output environment is known
 - Relative units – size is based on the parent element’s size

```
h1 { font-size: 12px; }  
h1 { font-size: 1in; }  
h1 { font-size: 1.2em; }  
h1 { font-size: 150%; }
```

Fonts & Text

- ‘font-weight’ property specifies the weight (or boldness) of the font

```
p { font-weight: normal; }  
p { font-weight: bold; }  
p { font-weight: 200; }  
p { font-weight: 600; }
```

- The number can be between 1 and 1000, inclusive
 - Higher numbers represent weights that are bolder than (or as bold as) lower numbers
-
- ‘line-height’ property sets the amount of space used for lines, such as in text

```
p { line-height: normal; }  
p { line-height: 2.5; }  
p { line-height: 3em; }
```


Fonts & Text

- ‘text-align’ property describes how inline content like text is aligned in its parent block element

```
p { text-align: left; }  
p { text-align: right; }  
p { text-align: center; }  
p { text-align: justify; }
```

- The number can be between 1 and 1000, inclusive
- Higher numbers represent weights that are bolder than (or as bold as) lower numbers
- ‘line-height’ property sets the amount of space used for lines, such as in text

```
p { line-height: normal; }  
p { line-height: 2.5; }  
p { line-height: 3em; }
```

Fonts & Text

- ‘text-decoration’ property specifies the appearance of decorative lines used on text
 - It is specified as one or more space-separated values representing the various longhand text-decoration properties
 - text-decoration-line
 - Sets the kind of decoration used, such as ‘underline’ or ‘line-through’
 - text-decoration-color
 - Sets the color of the decoration
 - text-decoration-style
 - Sets the style of the line used for the decoration, such as solid, wavy, or dashed

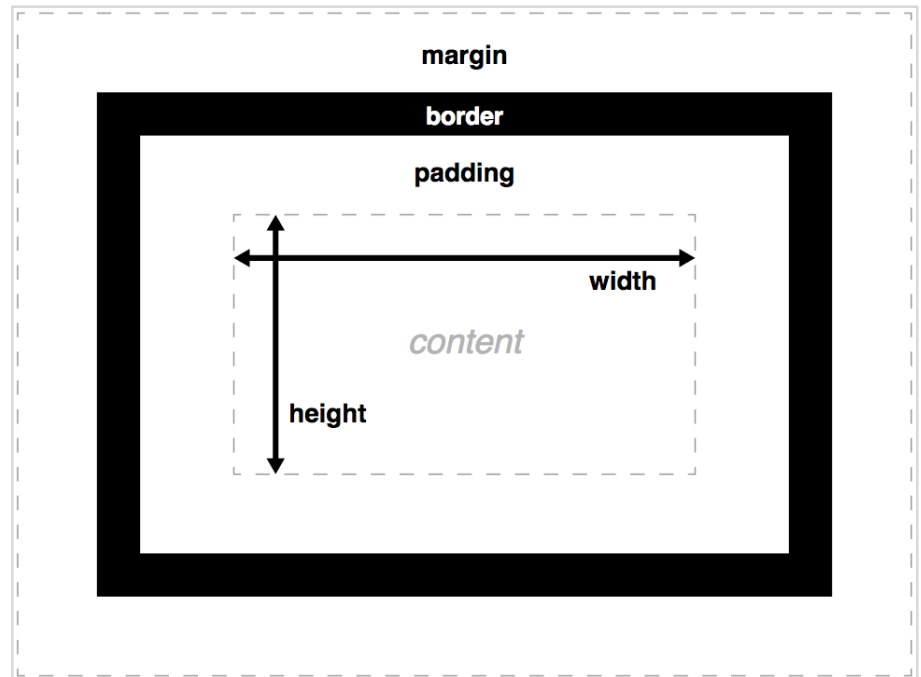
```
p { text-decoration: line-through; }  
p { text-decoration: none; }  
p { text-decoration: underline red; }  
p { text-decoration: wavy overline lime; }
```

Fonts & Text

- Using Google Fonts
 - Navigate to <https://fonts.google.com/>
 - Use the search bar in the top right corner to find the font of your choice
 - Click the red circle with the plus sign to the right of the font name
 - Now click the grey tab that appears in the lower right corner of the screen
 - Click the "customize" tab and select any of the options you need
 - Click the "embed" tab and copy the link beneath the "standard" option
 - Paste this link tag into the <head> of your web page, before the link to your custom CSS

The Box Model

- Is the foundation of layout on the Web
- Defines the spacing around boxes
- Each element is represented as a rectangular box, with the box's content, padding, border, and margin built up around one another like the layers of an onion
- Margin
 - Is the space outside the border, between the border and the next element
- Padding
 - Is the space inside the border, between the border and the content



The Box Model

- ‘width’ property specifies the width of an element
 - By default, the property defines the width of the content area

```
p { width: 300px; }  
p { width: 25em; }  
p { width: 75%; }  
p { width: auto; }
```

- ‘height’ property specifies the height of an element
- ‘padding’ property sets the padding area on all four sides of an element

```
p { padding: 10px; }  
p { padding: 5% 10%; }  
p { padding: 1em 2em 3em; }  
p { padding: 10px 50px 30px 0; }
```

The Box Model

- 'margin' property sets the margin area on all four sides of an element

```
p { margin: 10px; }  
p { padding: 5% 10%; }  
p { padding: 1em 2em 3em; }  
p { padding: 10px 50px 30px 0; }
```

Q & A

- Thank you!