# CSS and JS

Please try and sit near the front!

Slides created in conjunction with: Ilchul Yoon, Nelson Padua Perez

# Agenda

- Exercise 1/Warm Up
- CSS Selectors Wrap Up
- Box Model
- Background Colors
- Fonts
- Intro to JS

# Exercise Questions?

Remember to Validate!!

# Warm Up

Draw out DOMExample.html’s DOM tree!

# Kinds of Selectors

- Descendant selector
  - Override the type, class and id selector styles
  - Typically with two elements where the second is a descendant

- Examples

```
li a {font-size: 2em}

#header h2 {font-weight: normal;}

#content h2 {font-weight: bold;}
```

# Kinds of Selectors

- Universal selector
  - Applies to all elements in context
  - Example: * {font-family: arial, Helvetica; }

- Pseudo-elements
  - Allows you to style an item that is not marked by elements
  - Two pseudo-elements :first-letter, and :first-line

# Child Selectors

- A child selector matches when an element is the child of some element. A child selector is made up of two or more selectors separated by ">".
- Example

```
body > p { line-height: 1.3; }
```

sets the style of all p elements that are children of body:

```
div ol > li p { color: tan;}
```

What does this do?

# Adjacent Sibling Selectors

- The selector matches if E1 and E2 share the same parent in the document tree and E1 immediately precedes E2, ignoring non-element nodes (such as text nodes and comments)
- Syntax: E1 + E2, where E2 is the subject of the selector
- Example

```
math + p { text-indent: 0 }

h1 + h2 { margin-top: -5mm }
```

# Attribute Selectors

- Match elements which have certain attributes defined in the source document
- Syntax:
  - [att] - For when the element sets the att attribute, value doesn’t matter
  - [att=val] - For when the attribute is set to val specifically
- Examples:

```
h1[title] { color: blue; }

span[class=example] { color: blue; }
```

# Other Examples

- a[title] { }
    -
- .myPrefStyle{ }
    -
- #mySchedule { }
    -
- div.myPrefStyle{ }
    -

# Other Examples

- a[title] { }
  - Anchor elements with a title attribute
- .myPrefStyle{ }
  - Any elements with the class myPrefStyle (same class name can appear in many elements)
- #mySchedule { }
  - An element with id mySchedule (only one element on the page)
- div.myPrefStyle{ }
  - A div with the class myPrefStyle

# Other Examples

- div#mySchedule{ }
    - 
- div table { }
    - 
- input[type = “submit”] { }
    -

# Other Examples

- div#mySchedule{ }
    - A div with the id mySchedule
- div table { }
    - A table with a div ancestor
- input[type = “submit”] { }
    - An input element with a type attribute that has the value submit

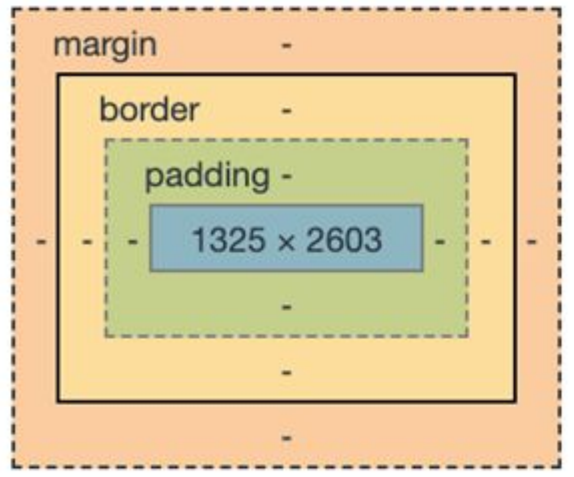
# Box Model

# CSS Box Model

- Refers to margin, border, padding and content HTML element components
- Allows us to define space between elements
- Cheat Sheet:
  - Content: What lies in the middle of the box
  - Padding: What immediately surrounds the content
  - Border: Surrounds the padding and represents the border of the box
  - Margin: Surrounds the border

# CSS Box Model

- Cheat Sheet:
  - Content: What lies in the middle of the box
  - Padding: What immediately surrounds the content
  - Border: Surrounds the padding and represents the border of the box
  - Margin: Surrounds the border



# Pro Tips for Box Model

- You may adjust the individual padding/margin properties:
  - padding-bottom, padding-left, padding-top, padding-right
  - margin-bottom, margin-left etc.

- Margin, border, padding and background are not inherited properties (children will reset)

# Want to save time? Use Shorthand Properties

Instead of listing out each property in seperate lines, you may use any of the following tags, with values separated by a space:

- background
- font
- padding
- [illegible]

Example: shorthandProperties.html/CSS

# Background Properties

- background-color
- background-image:  location (url) of image
- background-repeat: how image repeat
    - Possible values for repetition
        - no-repeat : one instance of the image
        - repeat : tile
        - repeat–y : repeats on the y-axis
        - repeat–x : repeats on the x-axis
- background-attachment :
    - indicates attachment of the image to the containing element.
    - Possible values are:
        - scroll : default value
        - fixed : image will stay stationary as the scrolling takes place

# Background Properties

Example: background.html/css && backgroundStretch.html/css

- Background-position
  - Possible values (some combinations are valid)
    - top, bottom, center, left, right
- Background-size
  - Possible values
    - auto auto - rertaims the original size
    - 000px 000px
    - x% y%
    - contain | cover | …
- Background images can be used in elements other than body (e.g., div)
  - There is a shorthand property for backgrounds
    - background: lightblue url("campusBldg.jpg") no-repeat fixed center;

# Generic Font Families

- **serif**: Times New Roman, Georgia, Times, …
- **sans-serif**: Verdana, Helvetica, Arial, …
- **monospace**: Consolas, …
- **cursive**: *Lucida Handwriting*, …
- **fantasy**: Comic Sans MS, …

Take a look at background.html for an example

# Choosing Fonts

- Specify a particular font
  - font-family: arial;
  - Works if the font is available on user’s machine
- Specify a generic family
  - font-family: serif;
  - Choices include: serif, sans-serif, monospace, cursive, fantasy
- specify a list of fonts
  - will be attempted in order
  - font-family: foobar, arial, sans-serif;

# Google Fonts

- Google supports a set of nice fonts anyone can link in HTML docs

- https://fonts.google.com/
- Choose fonts and put a link in HTML
- Use the fonts with “font-family” CSS property

# WTWAW (What To Walk Away With)

Make sure you know:

- Use various types of selectors (namely type, class and id)
- Change the background using CSS
- Use different Fonts
- Explain what the box model (and it’s elements are)

# JavaScript!

# JavaScript

- Finally some programming!

- JavaScript is a programming language that allows us to:
  - Create interactive web pages
  - Control a browser application
    - Open and create browser windows
    - Download and display contents
  - Interact with the user
  - Interact with HTML Forms

# JS and ECMAScript

- JavaScript implements ECMAScript

What is ECMAScript?

- A scripting language standard
- ActionScript and JScript are other implementations

# How is JavaScript different?

JavaScript implementation includes:

- ECMAScript
- DOM (Document Object Model)
- BOM (Browser Object Model)

# JavaScript Engine

Example: templateJS.html

- JavaScript engine process JavaScript code
  - Safari: JavaScriptCore
  - Chrome: V8
  - Firefox: Spidermonkey
  - Edge: Chakra
- To write JavaScript programs you need a web browser and a text editor
- A JavaScript program can appear:
  - In a file by itself typically named with the extension .js
  - In html files between a <script> and </script> tags.

# What is “use strict”?

- JavaScript's strict mode, introduced in ES5
- A way to opt in to a restricted variant of JavaScript, thereby implicitly opting-out of "sloppy mode".
- Several changes to normal JavaScript semantics:
    - Makes JavaScript silent errors throw errors
    - Prohibits some syntax likely to be defined in future versions of ECMAScript.
- Examples not allowed
    - Declaring function in blocks if (a < b) { function f() {} }
    - Setting a value to an undeclared variable

# Processing HTML with JS

- DOM – Document Object Model
  - Structured representation of the HTML page
  - Every HTML element is represented as a node
  - Browser uses HTML to build the DOM and can fix problems with the HTML so a valid DOM is generated
- Lifecycle
  - Set the user interface
    - Parse the HTML and build the DOM
    - Process (execute) JavaScript code
  - Enter a loop and wait for events to take place

# Processing HTML with JS

- When JavaScript is seen in a page, the DOM construction is halted and JavaScript code execution is started.

- JS can modify the DOM (e.g., creating, modifying nodes)
  - One reason why <script></script> elements appear at the bottom of a page (speed)

# Event Handling

- Relies on a single-threaded execution model
- An event queue keeps track of events that have taken place, but have not been processed (event-handler function for the event has not been called)
- All generated events (whether are user-generated or not) are placed in the event queue in the order they were detected by the browser
  - The browser mechanism that detects events and that adds them to the event queue is separate from the thread that is handling the events

# Browser’s Global Objects

- Browsers provide two global objects: window and document
- window object – represents the window in which a page resides
    - Provides access to other global objects (e.g., document)
    - Keeps track of user’s global variables
    - Allows JavaScript to access Browser’s APIs
- document object
    - Property of the window object that represents the DOM of the current page
    - Via this object you can access & modify the DOM

# Types of JavaScript Code

- Function Code
    - Code contained in a function
- Global Code
    - Code placed outside all functions
    - Automatically executed by JS engine
- As in Java, a stack is used to keep track of function calls. Each function call generates a function execution context (stack frame)
- There is one frame called the global execution context created when the JS program starts executing.

# WTWAW

After today make sure you know:

- What is JavaScript?
- How does JS fit in with the DOM?
- What is the JavaScript LifeCycle?
- How does JS handle events?