PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

LESSON 2: Preprocessors & Build Tools

HTML 300

OVERVIEW

1. Assignment Review
2. CSS Review
3. npm Overview
4. Gulp Overview
5. Setting up Sass Preprocessing
6. Sass Overview: variables, nesting, mixins, partials
7. Sass Activity
8. Assignment 2

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

DISCUSSION RECAP

DISCUSSION RECAP

- API "Speed Dating"
  - Form two lines facing each other
  - Talk for two minutes to the person in front of you.
  - Discuss some your course project proposal.
  - What API did you choose? Why?
  - Do you foresee any potential issues? (key? account? authentication?)

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

CSS REVIEW

CSS Review

- Selectors
- Specificity
- Box Model/Box Sizing
- Review of floats vs Flexbox/CSS Grid

SELECTORS

- **Simple selectors**: Match one or more elements based on element type, class, or id.

```
// Select by element
p {
    font-size: 1rem;
    color: #000;
    margin: 1rem 0;
}
// Select by class
.my-class {
    display: block;
    padding: 1rem;
    max-width: 300px;
    width: 100%;
}
// Select by ID
#my-id {
    border: 1px solid #000;
    padding: 2rem;
    margin: 1rem auto;
}
```

SELECTORS

- **Attribute selectors**: Match one or more elements based on their attributes/attribute values.

```
// Select by attribute presence
[data-box] {
    width: 50%;
    margin: 1rem auto;
}
// Select by attribute value
[data-box=“big”] {
    width: 100%;
}
// Select by attribute if it contains value
[data-box~=“col-”] {
    float: left;
}
```

SELECTORS

- **Pseudo-selectors**: Pseudo-classes target *states* and pseudo-elements target certain *parts* of existing elements.

Pseudo-classes

:active
:any-link
:blank
:checked
:current
:default
:defined
:dir()
:disabled
:drop
:empty
:enabled
:first
:first-child
:first-of-type
:fullscreen
:future
:focus
:focus-visible
:focus-within
:has()
:host
:host()
:host-context()
:hover
:indeterminate
:in-range
:invalid
:is()
:lang()
:last-child
:last-of-type
:left
:link
:local-link
:not()
:nth-child()
:nth-col()
:nth-last-child()
:nth-last-col()
:nth-last-of-type()
:nth-of-type()
:only-child
:only-of-type
:optional
:out-of-range
:past
:placeholder-shown
:read-only
:read-write
:required
:right
:root
:scope
:target
:target-within
:user-invalid
:valid
:visited
:where()

Pseudo-elements

::after (:after)
::backdrop
::before (:before)
::cue (:cue)
::first-letter (:first-letter)
::first-line (:first-line)
::grammar-error
::marker
::placeholder
::selection
::slotted()
::spelling-error

SELECTORS

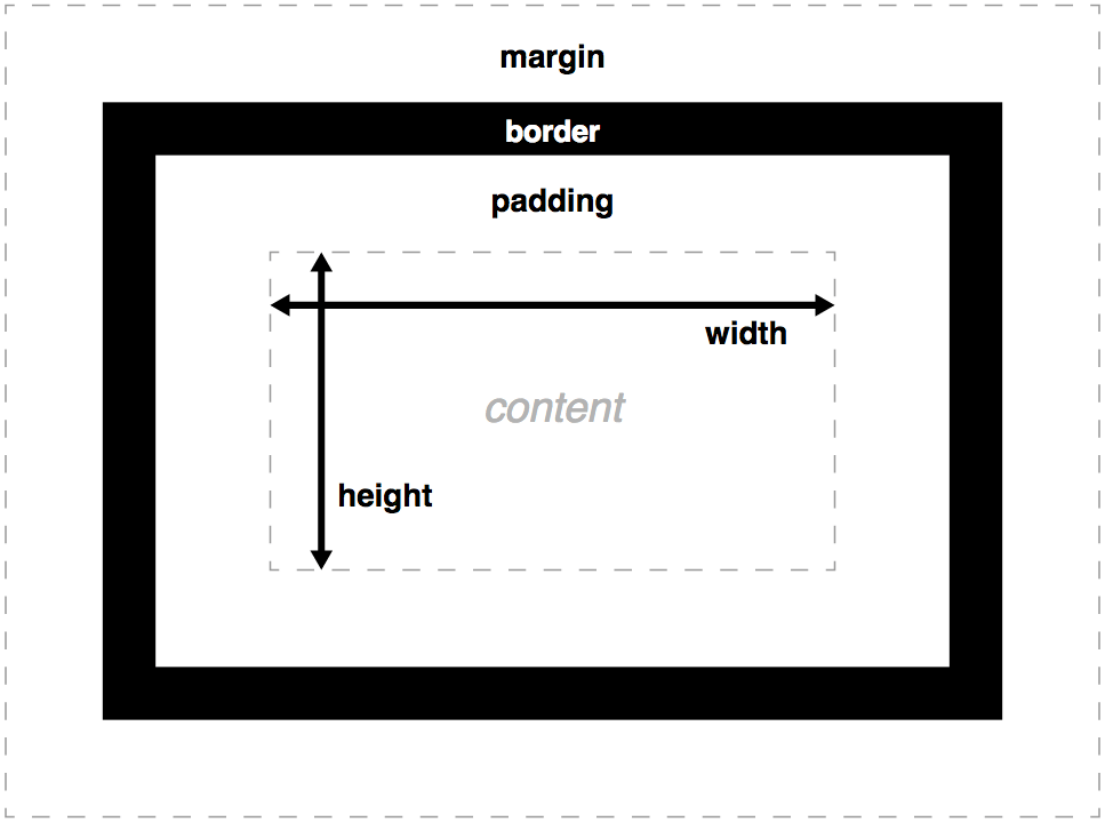
```
// Pseudo elements
p:first-letter {
    font-size: 2rem;
    font-weight: 700;
}

.btn:before {
   content: "\f002";
}

// Pseudo classes
.box:last-child {
    border: none;
}

.box:nth-of-type(3n) {
    background: #000;
}
```

SPECIFICITY

- **Specificity** is the means by which browsers decide which CSS property values are the most relevant to an element and, therefore, will be applied. It is based on the matching rules which are composed of different sorts of CSS selectors.
- Inline styles added to an always overwrite any styles in external stylesheets, and thus can be thought of as having the highest specificity.
- When an important rule is used on a style declaration, this declaration overrides any other declarations.
  Using !important, however, is **bad practice** and should be avoided if possible.

BOX MODEL

- CSS Box Model: 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



Box Sizing

- **Change the box model**: The total width of a box is the sum of its width, padding-right, padding-left, border-right, and border-left properties. It's possible to tweak the box model with the property box-sizing.
- Use box-sizing: border-box; to update the box model to include padding and border as part of the total width of the element

```
html {
    box-sizing: border-box;
}

*,
*:after,
*:before {
    box-sizing: inherit;
}
```

Layout (Floats)

- Floats originally intended for floating image inside text blocks
- Now are used to create responsive layouts for cross-browser experiences when needing to support older browsers (IE < 10)
- With flexbox and CSS grid, floats are only used for layouts if browser requirements deem it necessary
- Read more at Mozilla's info on floats:
  https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Floats

Layout (Flexbox/CSS Grid)

- Flexbox: a modern CSS layout specification to allow content to ‘flex’ by growing/shrinking/wrapping responsively
- Learn more flexbox here: https://learnflexbox.org/
- CSS Grid: a modern CSS layout specification to allow for content to be laid out in grid ‘tracks’
- Learn more CSS grid here: https://cssgrid.io/

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

npm Overview

npm

- Node Package Manager: npm is a package management tool for node and is used across modern web development for both build processes and managed libraries
- Visit https://nodejs.org/en/ and install node if you don't already have it
- npm uses helper config files called package.json, these files allow you to setup a series of packages you would like to install, and then anyone can install them all at once with a simple command of `npm install`

npm – Example package.json

```
{
  "name": "local-dev",
  "version": "1.0.0",
  "description": "Gulp workflow for local static HTML/CSS/JS",
  "main": "gulpfile.js",
  "author": "Bryce Benson",
  "license": "ISC",
  "devDependencies": {
    "babel-core": "^6.26.3",
    "babel-preset-env": "^1.6.1",
    "babel-preset-es2015-ie": "^6.7.0",
    "browser-sync": "^2.24.3",
    "gulp": "^3.9.1",
    "gulp-autoprefixer": "^4.1.0",
    "gulp-babel": "^7.0.1",
    "gulp-concat": "^2.6.1",
    "gulp-minify-css": "^1.2.4",
    "gulp-plumber": "^1.1.0",
    "gulp-rename": "^1.2.2",
    "gulp-sass": "^3.1.0",
    "gulp-uglify": "^3.0.0",
    "gulp-watch": "^5.0.0"
  }
}
```

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

Gulp Overview

Gulp

- A task manager for node.
  - You can establish pipelines for gulp to work within your project and handle your files or processing
- Uses a config file called gulpfile.js to spell out what exactly you want gulp to do for you
- Has many plugins available to handle different tasks
  - We'll look at gulp-sass to do our sass preprocessing

Gulp – Example Tasks

```
gulp.task("sass", function() {
  gulp
    .src("css/*.scss")
    .pipe(plumber())
    .pipe(sass())
    .pipe(autoprefix())
    .pipe(minifycss({ compatibility: "ie8" }))
    .pipe(gulp.dest("css/"))
    .pipe(browserSync.stream());
});

gulp.task("watch", function() {
  gulp.watch("css/*.scss", ["sass"]);
});

gulp.task("browser-sync", function() {
  browserSync.init({
    notify: false,
    server: {
      baseDir: "./"
    }
  });
});
```

Gulp

- We will be using gulp within our activity and assignments
- Our gulpfile will outline tasks to:
  - Process all of our SCSS files, running autoprefixer (automatically adds all needed browser prefixes) on them, minifying them, and outputting a final CSS file
  - Start a local node server to run browser sync which allows us to auto-reload our page to see changes instantly
  - Start a watch task to tell gulp to watch our HTML/SCSS files for changes and then process them again

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

SASS PREPROCESSOR

Preprocessors

- A **preprocessor** essentially takes an input, does something with it, and outputs compiled code, usually into a different language or syntax (SCSS -> CSS, CoffeeScript -> JavaScript)
- The Sass preprocessor does exactly this – starts with .scss (or .sass) files, processes them, and spits out compiled CSS ready to use.
- We'll be looking at the **scss** syntax (.scss is the standard sass syntax, .sass is also available with different syntax requirements)

Sass Features

- Some of the most prominent features that sass extends for CSS include:
  - Variables
  - Nesting
  - Partials
  - Functions & Mixins
  - Color functions

Sass Features

- Variables
- Uses $ to denote variable name followed by : and the value
- Able to reuse common or themed values
- Common uses: colors, font families, standard spacing, box/text shadow values

```
// Brand colors
$macaw: #00467f;
$dove: #f5f5f5;
$emu: #d8d8d8;

h1 {
  color: $macaw;
}
```

Sass Features

- Nesting
- Nested selectors help denote relations between elements
- Using & will append the parent selector to the nested one
- Careful not to go too many levels deep (2-3)

```
.go-back {
  font-family: $korolev;
  color: $teal;
  font-weight: 700;
  position: relative;
  text-decoration: none;
  text-transform: uppercase;
  padding-left: 1rem;
  &:before {
    content: "\f104";
    font-family: $fa;
    position: absolute;
    left: 0;
  }
}
```

Sass Features

- Partials
- Partials are smaller scss files that all compile into one large CSS file, usually denoted from a main.scss file
- Within the main.scss file, use the @import syntax to add in relative partial files (ignoring _).
  Name partials starting with _
  - e.g) "_variables.scss"
- Remember, ordering your imports will matter in the cascade

```
@import 'variables';
@import 'typography';
@import 'responsive';
@import 'globals';
@import 'content';
```

Sass Features

- Functions & Mixins
- Functions in are sass work similar to JS functions
- Mixins can accept parameters and work with variables
- Define mixins with @mixin title(parameters)
- Call mixins with @include title(parameters)

```
@function calculateRem($size) {
  $remSize: $size / 16px;
  @return $remSize * 1rem;
}
@mixin font-size($size) {
  font-size: $size;
  font-size: calculateRem($size);
}
```

Sass Features

- Color Functions
- Sass has built in functions for adjusting color
- Commonly invoked with a keyword(color, amount) syntax
- Take a look at example here: https://codepen.io/brycebenson/pen/OrGRWd

```
$yellow: #ffb617;

a {
    color: $yellow;
    &:hover {
        color: darken($yellow, 15%);
    }
}
```

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

Sass Activity

Let's Try It Out

- Make sure you have node.js installed
- Clone the activity repo from the UW Front-End GitHub
- In your terminal, run `npm install`
- Make sure your packages installed successfully, flag if you need assistance
- Now run `gulp` in your terminal
- Convert the included CSS to SCSS, make sure to use variables and nesting where appropriate

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

Assignment 2

MOCKUP

Steve Smith

Project Manager

Company: Front End Dev Co

Experience: 3 years

School: UW

Major: Marketing

Email: steve@fedc.com

steve.linkedinprofile.com

BUILDING WITH SASS/GULP

- Clone the assignment repo from the UW Front-End GitHub
- Follow the README.md instructions
- You'll be building the provided component with sass
- The 'ref' folder contains the source XD file as well as a jpg and SVG for reference
- Check canvas for the assignment rubric
- Make sure to use a media query to stack the image/title on top of the copy in mobile

QUESTIONS

Any Questions?

As always feel free to contact me though Canvas if you have any questions. I do have a full-time job, so I might not get back to you immediately.

If you do not hear back from me in 24 hours, please ping me again.