Gettin' Sassy

SASS & SCSS

WHY SASS

Nested Syntax

Mixins

Maintainable

Variables

Installation

Windows - npm install -g sass or

Chocolatey Package manager - choco install sass

Mac Homebrew - brew install sass/sass/sass

Tools - http://koala-app.com , https://codekitapp.com, http://compass.kkbox.com

Nested Syntax

SASS allows you to use a nested syntax, which is code contained within another piece of code that performs a wider function. In SASS, nesting allows a cleaner way of targeting elements. In other words, you can nest your HTML elements by using Css Selectors.

Easier to read

Don’t have to rewrite selectors multiple times

Better organization and maintainable code.

Mixins

Mixins are like functions in other programming languages. They return a value or set of values and can take parameters including default values.

Example of Mixins

```
@mixin set-font( $family: 'Ubuntu' , $weight: 400 , $style: normal
) {

  font-family: $family , 'Arial', 'Helvetica', sans-serif;

  font-style: $style;

  font-weight: $weight;

}
```

```
h1.hover {
 @include set-font;
 color: $blue;
}
```

Variables

A variable allows you to store a value or a set of values, and to reuse these variables throughout your SASS files as many times you want and wherever you want. Easy, powerful, and useful.

You're building a site with a fancy red color everywhere; buttons, menus, icons, containers, etc. You're also using a couple of great fonts: Ubuntu and Nunito. Using traditional CSS, you would need to repeat the same code over and over, but with SASS you can store these values as variables:

Example of Variable

```
$red: #d91a56;

$ubuntu-font: 'Ubuntu', 'Arial', 'Helvetica', sans-serif;

$nunito-font: 'Nunito', 'Arial', 'Helvetica', sans-serif;
```

```
h1 {
  font: $ubuntu-font;
  color: $red;
}

a {
  font: $nunito-font;
  background-color: $red;
  padding: 6rem;
```