Sass Libraries, Advanced Functionality, Modular/OOSASS

Sass Libraries

- Since Sass gives you the ability to write functions (extensions), there are several libraries of commonly used Sass functionality
- Some of these are:
  - Bourbon bourbon.io
  - Scut davidtheclark.github.io/scut
  - Compass compass-style.org

Sass Libraries - Bourbon

- "A simple and lightweight mixin library for Sass"
- Makes it far easier to write cross-browser-friendly Sass

Sass Libraries - Bourbon

Installation

```
$ gem install bourbon
```

```
$ bourbon install #in the scss directory
```

```
 /*in your main SASS file*/
 @import 'bourbon/bourbon';
```

Sass Libraries - Bourbon

- Commonly used Bourbon mixins/functions:
  - background-image
  - linear-gradient
  - border-image
  - font-face
  - tint
  - shade
- We'll cover `linear-gradient` and `background-image` usage later in the class

Sass Libraries - Bourbon

```
/*mixes a color with white*/
background: tint(orange, 50%);

/*mixes a color with black*/
background: shade(red, 70%);
```

Exercise

Implementing Sass Libraries

- Implement Bourbon into a blank website (install the gem, run bourbon install in your directory, start using mixins)
- Try out at least 2 of the library's methods, perhaps:
    - tint
    - shade

Sass if

- Sass has a version of control flow just like Ruby, JavaScript, and other languages

```
h1{
  @if $fs == 1{ font-size: 10px }
  @else if $fs == 2{ font-size: 18px }
}
```

- It's not recommended to use them that much day-to-day, mostly when writing your own custom mixins

Sass `@for` loop

- A great way to make a utility class with an incremental measurement, like a width or font size

```
@for $i from 1 through 100 {
  .cell-#{$i} { width: 1% * $i; }
}
```

- The above Sass would give you `.cell-1`, `.cell-50`, `.cell-25` classes that would easily allow you to make percentage width cells in an element

Exercise

- Use a Sass `for` loop to make a utility class that allows you to easily apply 6 font sizes between 12px and 72px.
- Use the example from the last slide as a starting point

Sass @each loop

- Allows you to iterate over a Sass list, which is like an array

```
@each $person in zach, ray, sally {
  .icon-#{$person} {
    background-image: url('/images/#{$person}.png');
  }
}
```

Exercise

- Use a Sass each loop to create utility classes for the `list-style-type` CSS property
- These classes should allow you to apply the `disc`, `circle`, and `decimal` list-style-types just by using CSS classes
- Use your code from the last slide to get started

Placeholders

Placeholders are Sass selectors that don't get used unless they're called in an @extend directive

```
%thick-line {
  border: 10px solid black;
}
/*will not get rendered unless called in @extend */

.profile {
  @extend %thick-line;
}
```

Exercise

- Create a button class using a Sass placeholder (`%button`)
- Extend this class into two real classes that each add on to the styles of the original button (`.button-style-one`, `.button-style-two`)
- Let's go over an example together first!