LESS & SASS

- **Why CSS is Painful**
  - The “Color Problem”
  - Duplication Issues
  - Cascading Avalanches
  - Lack of Calculations
  - The Problem with Imports

Cascading Style Sheets

- **Declarative Nature Means It’s Easy To Write**
  - “Editor Inheritance” becomes the norm
  - Duplication leads to errors

LESS

- **LESS is More**
  - Using LESS on the Client
  - Using LESS on the Server
  - Importing
  - Variables
  - Functions
  - Operations
  - Mixins
  - Nested Rules
  - Other Features

What is LESS?

- **Dynamic Style Sheet Language**
  - “Compiles” to CSS
  - Introduces programming features to CSS
  - Looks and Feels like CSS
    - I.e. all CSS is valid LESS

Using LESS on the Client

```
<head>
  <link rel="stylesheet/less"
        type="text/css"
        href="css/my.less">

  <script src="js/less.js"
          type="text/javascript"></script>
</head>
```

Node.js

```
$ npm install less

var less = require('less');

less.render(lessContents,
            function (e, css) {
              console.log(css);
            });
```

LESS Basics

- **LESS is meant to feel like CSS but better**
  - All CSS is valid…really
    - Renaming your .css to .less works
  - LESS adds to CSS

LESS Basics (2)

```
@baseFontSize: 14px;
/* Comments */
// Comments too
#main
{
  h1
  {
    font-size: @baseFontSize;
  }
}
```

Importing

- **@import works**
  - Embeds other .less files in a file
  - Allows for simple modularization
    - While maintaining merging of CSS
  - If Import is .css, it preserves the @import statement
  - If Import is .less, it merges it into file

Variables

```
@myColor: #ffeedd;

// They are Constants, this doesn’t work
@myColor: @myColor + 5%;

@a: Black;                        // Color
@b: 4px;                          // Units
@c: 1.0em;                        // Units
@d: Helvectica, sans serif; // Strings
@e: 1px #000 Solid 0 0;      // Complex Type
```

Operations

```
// Operations Just Work
font-size: 4px + 4;       // 8px
font-size: 20px * .8;     // 16px;
color: #FFF / 4;          // #404040;
width: (100% / 2) + 25%; // 75%
```

Color Functions

```
color: lighten(@color, 10%);
color: darken(@color, 10%);

color: saturate(@color, 10%);
color: desaturate(@color, 10%);

color: fadein(@color, 10%);
color: fadeout(@color, 10%);
color: fade(@color, 50%);

color: spin(@color, 10%);
color: mix(@color, #246);
```

More Functions

```
@hue: hue(@color);
@sat: saturation(@color);
@light: lightness(@color);
@alpha: alpha(@color);
@color: hsl(20%, 30%, 40%);

// Math
@rnd: round(3.14);
@top: ceil(3.14);
@bot: floor(3.14);
@per: percentage(.14);
```

Mixins

- **Repeatable sections**
  - Feel like functions
  - But insert more than one name/value pair
  - Can accept parameters, defaults and overloads

Mixins

```
.rounded-corners-all(@size) {
  border-radius: @size;
  -webkit-border-radius: @size;
  -moz-border-radius: @size;
}

#form
{
  .rounded-corners-all(5px);
}
```

Mixins

```
// Default Values
.rounded-corners-all(@size: 5px) {
  border-radius: @size;
  -webkit-border-radius: @size;
  -moz-border-radius: @size;
}

#form
{
  .rounded-corners-all;
}
```

Mixins

```
// Using overloads
.color(@color) {
  color: @color;
}

.color(@color, @factor) {
  color: lighten(@color, @factor);
}

#form
{
  .color(#888, 20%); // Uses 2nd overload
}
```

Mixins

```
// Using guards
.color(@color) when (alpha(@color) >= 50%) {
  color: Black;
}

.color(@color) when (alpha(@color) < 50%) {
  color: transparent;
}

#form
{
  .color(@mainColor); // Uses 1st overload
}
```

Mixins

```
// Using type guards
.width(@size) when (isnumber(@size)) {
  width: @size * 2;
}

.width(@size) when (ispercentage(@size)) {
  width: @size;
}

#form
{
  .width(50%); // Uses 2nd overload
}
```

Nested Rules

- **Allows you to structure rules in a logical way**
  - Hierarchies imply the cascading/specificity
  - LESS then deconstructs it into CSS for you

Nested Rules

```
// LESS Nested Rules
nav {
  font-size: 14px;
  font-weight: bold;
  float: right;
  ul {                   // Makes “nav ul {...}”
    list-style-type: none;
    li {                 // Makes “nav ul li {...}”
      float: left;
      margin: 2px;
    }
  }
}
```

Nested Rules

```
// Use Combinator (&) to mix with parent:
a {
  text-decoration: none;
  &:hover {
    text-decoration: underline;
  }
}

// Results in
a { text-decoration: none; }
a:hover { text-decoration: underline; }
```

Namespaces

```
// Namespacing for organizational grouping
#my-forms {
  .set-button {
    font-size: 14px;
    text-align: center;
  }
}

#submit-button {
  #my-forms > .set-button;
}
```

Scoping

```
// Variables/Mixins are Scoped
@size: 24px;

#form {

  @size: 18px;

  .button {
    font-size: @size; // 18px;
  }
}
```

String Interpolation

```
// Can use Ruby/PHP style string insertion
@root: "/images/";

#form {
  background: url("@{root}background.jpg");
  // Becomes url("/images/background.jpg")
}
```

Using JavaScript

```
// Embed with back-quotes to execute JS
@root: "/images";
@app-root: `"@{root}".toUpperCase()`;

#form {
  // Becomes url("/IMAGES/back.jpg");
  background: url("@{app-root}/back.jpg");
}
```

Agenda

- **SASS**
    - What is SASS
    - Using SASS on the Server
    - Variables
    - Rules
    - Importing
    - Extending
    - Mixins
    - Functions
    - Control Directives

What is SASS?

- **Dynamic Style Sheet Language**
  - Syntactically Awesome StyleSheets
  - “Compiles” to CSS
  - Introduces programming features to CSS

What is SASS? (2)

- **SASS has two syntaxes**
  - SASS and SCSS

```
/* SCSS */
$baseFontSize: 14px;

#main
{
  h1
  {
    font-size: $baseFontSize;
  }
}
```

SASS on the Server

- **Support for Server-Side**
  - Node.js

.

Node.js

```
$ npm install sass

var less = require('sass');

sass.render(sassContents,
            function (e, css) {
              console.log(css);
            });
```

Variables

```
$myColor: #ffeedd;

$a: Black;                      // Color
$b: 4px;                        // Units
$c: 1.0em;                      // Units
$d: Helvetica, sans-serif;      // Lists
$e: 1px #000 Solid 0 0;         // Also Lists
```

Operations

```
// Operations Just Work
font-size: 4px + 4;        // 8px
font-size: 20px * .8;      // 16px;
color: #FFF / 4;           // #404040;
width: (100% / 2) + 25%; // 75%
```

Color Functions

```
color: lighten($color, 10%);
color: darken($color, 10%);

color: saturate($color, 10%);
color: desaturate($color, 10%);

color: fade_in($color, .1);
color: fade_out($color, .1);

color: invert($color);
color: complement($color);
```

More Functions

```
$quoted: quote($sometext);
$unquoted: unquote($sometext);

$value: if(true, $color1, $color2);

$rnd: round(3.14);
$top: ceil(3.14);
$bot: floor(3.14);
$per: percentage(.14);
```

String Interpolation

```
// Can use Ruby/PHP style string insertion
$root: "/images/";

#form {
  background: url("#{$root}background.jpg");
  // Becomes url("/images/background.jpg")
}

// Also
$name: "my-class";
.#{$name} {
  color: Blue;
}
```

Rules

```scss
// SCSS
nav {
  font-size: 14px;
  font-weight: bold;
  float: right;
  ul {                // Makes "nav ul {...}"
    list-style-type: none;
    li {              // Makes "nav ul li {...}"
      float: left;
      margin: 2px;
    }
  }
}
```

Rules

```
// Use Parent Selector (&) to mix with parent:
a {
  text-decoration: none;
  &:hover {
    text-decoration: underline;
  }
}

// Results in
a { text-decoration: none; }
a:hover { text-decoration: underline; }
```

Rules

```
// Nested Properties too
.button {
  font: {
    family: Verdana, Helvetica, sans-serif;
    size: 14px;
  }
}

// Results in
.button {
  font-family: Verdana, Helvetica, sans-serif;
  font-size: 14px;
}
```

Directives

- **Operations on the CSS**
  - @import
  - @extend
  - @mixin
  - @function

@import

```
@import "foo.css";  // Emits CSS Import
                    // @import url(foo.css);

@import "foo.scss"; // Embeds in result
@import "foo";      // Also Embeds

// Nested Import too
#main {
 @import "colors";
}
```

@extend

```
// Inherits Styles from another
.button {
  color: Black;
}
.submit-button {
  @extend .button;
  border: 1px Black solid;
}
// Emits
.submit-button {border: 1px solid Black; }
.button, .submit-button {
  color: Black;
}
```

@extend

```
// multiple inheritance too
.submit-button {
  @extend a:hover;  // inherit from any rule
  @extend .button;  // multiple rules
  border: 1px Black solid;
}
```

@mixin

- **Repeatable sections**
  - Feel like functions
  - Used insert one or more than one name/value pair
  - Can accept parameters, defaults and overloads

@mixin

```
@mixin font-large {
  font: {
    size: 14px;
    family: san-serif;
    weight: bold;
  }
}

#form {
  @include font-large;
}
```

@mixin

```scss
// Parameters
@mixin rounded-corners-all($size) {
  border-radius: $size;
  -webkit-border-radius: $size;
  -moz-border-radius: $size;
}

#form {
  @include rounded-corners-all(5px);
}
```

@mixin

```
// Default Parameter Value
@mixin rounded-corners-all($size: 5px) {
  border-radius: $size;
  -webkit-border-radius: $size;
  -moz-border-radius: $size;
}

#form {
  @include rounded-corners-all;  // Optional
}
```

@function

```
// Value calculations
$app-width: 900px;
@function column-width($cols) {
  @return ($app-width / $cols) - ($cols * 5px);
}

.col2 {
  width: column-width(2);
}

.col3 {
  width: column-width(3);
}
```

Control Directives

- **For control flow:**
  - @if
  - @for
  - @each
  - @while

@if

```
h1 {
  @if $size > 14px {
    color: Blue;
  }
  @else if $size < 14px {
    color: Red;
  }
  @else {
   color: Green;
  }
}
```

@for

```
$page-width: 1000px;

@for $col from 1 through 4 {
  .col#{$col} {
    width: $page-width / $col;
  }
}
```

@each and @while

```
@each $item in first, second, third, fourth {
  .#{$item} {
    background-url: url(/images/#{$item}.jpg);
  }
}

$i: 1;
@while $i < 5 {
  h#{$i} {
    font-size: $i * 4px;
    $i: $i + 1;
  }
}
```