ENRIQUE CANALS

Software Developer @ The Hybrid Group
Currently working with Sphero, creating cool things for connecting robots to the internet
Worked on several interesting web projects with Pebble
Current maintainer of the TextMate SCSS Bundle for syntax highlighting with support for Atom and Sublime

@ecanals | www.enriquecanals.com
WHAT WE'RE GOING TO COVER IN THIS SESSION

PART 1

- CSS basics and shortcomings
- What are CSS pre-processors
- How pre-processors work and don't work

PART 2

- Which one do I use? (Sass, Less, Stylus)
- Workflow Integration
- Using pre-processors in WordPress theme development
PART 1
CSS BASICS AND SHORTCOMINGS
BASIC CSS SYNTAX

This is the CSS you're used to.

```css
html, body {
  width: 100%;
  height: 100%;
  overflow: hidden;
}

body {
  background-color: #fff;
  color: #000;
  line-height: 1;
  position: relative;
}
```

Simple enough.
NOW SOMETHING A LITTLE MORE COMPLEX:

You can see how this gets ugly fast.
SHORTCOMINGS WITH CSS

- Not DRY
- Hard to read
- Harder to maintain
- Seems lacking in basic features
We need more than what the CSS language provides.
This is where CSS pre-processors come into play.
DEVELOPMENT WORKFLOW WITHOUT A PRE-PROCESSOR

- Requires re-writing a lot of code, as CSS variables are still not supported in all browsers.
- Working with multiple CSS files means you're dependent on using `@import` to import rules from other stylesheets
  - Bad for performance
  - Requires making additional HTTP requests
- If you're NOT writing CSS in multiple files...
You get it.
CSS on its own just doesn't provide the features web developers need to create large scale web applications.
CSS pre-processors provide what we need in order to write maintainable and extensible styles.
WHAT ARE CSS PRE PROCESSORS?
THE MOST POPULAR PRE-PROCESSORS

They all do the same thing in slightly different ways.
SO WHAT EXACTLY DO THEY DO?

Simply put, CSS pre-processors are a build tool used to concatenate stylesheets.

That's basically it.

But wait... there's more.

A lot more.
FEATURES

- Nesting
- Variables
- Mixins
- Operations and Functions
- Imports
- Partialso
A NOTE ABOUT THESE FEATURES

• Sass, Less and Stylus support all of these features.
• The differences between them are not significant enough to make an argument for one over the other; it's a matter of preference.
• None of these features add to or enhance CSS in anyway, they simply improve the development workflow for writing CSS.
NESTING

SASS (SCSS)

.hero {
  .hero-button {
    &:hover {
      
    }
  }
}

CSS OUTPUT

.hero {}

.hero .hero-button {}

.hero .hero-button:hover {}
**NESTING**

Protip: don't nest too much. Max. 4 levels.
VARIABLES

Variables allow you to store values and reuse them as needed in your code.

```
$cool_orange: #E2991F;
$main_color: $cool_orange;

h4 {
    color: $main_color;
}
```
VARIABLES

Variables can be used for all types of values.

$headingFont: 'League Gothic', Impact, sans-serif;
$headingColor: #eee;
$headingLineHeight: 1.2;
$headingLetterSpacing: normal;
$headingTextTransform: uppercase;
$headingTextShadow: none;
$headingFontWeight: normal;
$heading1TextShadow: $headingTextShadow;

These are variables used for this presentation.
Mixins allow you to define rules that output CSS.

```css
@mixin radial-gradient( $outer, $inner, $type: circle ) {
  background: $outer;
  background: -moz-radial-gradient( center, $type cover, $inner 0%, $outer 100% );
  background: -webkit-gradient( radial, center center, 0px, center center, 100%, color-stop(0%,$inner), color-stop(100%,$outer) );
  background: -webkit-radial-gradient( center, $type cover, $inner 0%, $outer 100% );
  background: -o-radial-gradient( center, $type cover, $inner 0%, $outer 100% );
  background: -ms-radial-gradient( center, $type cover, $inner 0%, $outer 100% );
  background: radial-gradient( center, $type cover, $inner 0%, $outer 100% );
}

.radial-background {
  @include radial-gradient( #add9e4, #f7fbfc );
}
```
OPERATIONS AND FUNCTIONS

Color Functions

$sick\_orange$: saturate($cool\_orange$, 50%);
$sat\_orange$: saturate($cool\_orange$, 50%);

$light\_orange$: lighten($cool\_orange$, 10%);

$not\_orange$: invert($cool\_orange$);
$not\_orange$: invert($cool\_orange$);

$dark\_orange$: darken($cool\_orange$, 25%);
$dark\_orange$: darken($cool\_orange$, 25%);

Math Operations

width: 360px * 3 / 6px;
width: 360px * 3 / 6px;

/* = 180px */
/* = 180px */

.small-image {
width: $imgWidth / 2;
}
.width: $imgWidth / 2;
IMPORTS

Importing multiple files now comes at no cost

@import "../template/mixins";
@import "../template/modules";

@import url(https://fonts.googleapis.com/css?family=Quicksand:400,700,400italic);
@import url(https://fonts.googleapis.com/css?family=Open+Sans:400italic);
Partials allow you to split your code into smaller, more manageable files that can then be used inside of other Sass files.

Files prefixed with an underscore (i.e. `_partial.scss`) are regarded as partials and are not generated into a CSS file.

```plaintext
css/
  base.scss
scss/
  base/
    _colors.scss
    _mixins.scss
    _reset.scss
layout/
  _grid
```

Partials are used with the `@import` directive.

```plaintext
@import 'reset'
@import 'mixins'
@import 'colors'
```
"I DON'T WANT TO ADD ANOTHER DEPENDENCY TO MY APPLICATION..."

Agreed. CSS pre-processors are meant to be used as build tools that run in your development environment. There's no need to add anything to your web server in order for you to use it.
"DO I NEED TO LEARN RUBY OR NODE IN ORDER TO USE A PRE-PROCESSOR?"

No. But you will need to have them installed on your workstation in order to use them.

Sass is written in Ruby. Less and Stylus are written in Javascript. Installing both is easy on Windows, Linux and OS X.
"A IS BETTER THAN B BECAUSE OF ..."

Sass, Less and Stylus all provide the same features. Each has its own unique way of accomplishing the same task. It all comes down to personal preference, so try all of them and make your own decision on what's best.
PART 2
WHICH ONE SHOULD I USE?

Sass, Less, or Stylus
HOW TO MAKE A DECISION?

Everyone has their own way of doing things.
If you're into Ruby, Sass might be your thing.
If you're into Javascript, Less or Stylus might be better too.
See what works best for you.
Both Sass and Haml were created by Hampton Catlin (@hcatlin) and Natalie Weizenbaum (@nex3)
TJ Holowaychuk (@tjholowaychuk) is the creator of Stylus, as well as the extremely popular Jade templating engine and Express, the Node web framework.
RELATED PROJECTS

Alexis Sellier (@cloudhead) is the original author of Less.

Bootstrap is built with Less; but has Sass support as well.
GETTING STARTED WITH SASS

- Has 2 different syntaxes
  1. `.sass` | CoffeeScript-esqe
  2. `.scss` | Just like CSS
- Foundation and Bootstrap Support
- Insanely efficient when used with compass
WORKFLOW INTEGRATION & FEATURES
INSTALLATION AND SETUP

INSTALL ONLY SASS

> gem install sass
> sass --watch /path/to/sass/file.scss

That's it.
INSTALLATION AND SETUP

INSTALL SASS + COMPASS

> gem install compass
> compass create new-project
> compass watch

Sass is included as a dependency in the compass gem
ADDITIONAL DEVELOPMENT TOOLS

Compass.app, Less.app, CodeKit, LiveReload, Crunchapp

Compass.app, SimpLess, Crunchapp

Compass.app, Crunchapp, SimpleLess, WinLess, LiveReload
WORKSTATION SETUP

Choose an editor / IDE that you like and that fits your needs.

Most popular editors have Sass support. Here are a few:
CONVENIENT FEATURES

Nested media queries

```
$break-small: 320px;
$break-large: 1200px;

.profile-pic {
    float: left;
    width: 250px;
    @media screen and (max-width: $break-small) {
        width: 100px;
        float: none;
    }
    @media screen and (min-width: $break-large) {
        float: right;
    }
}
```
CONVENIENT FEATURES

Creating a color palette

```
@base: #663333;
@lighter1: lighten(spin(@base, 5), 10%);
@lighter2: lighten(spin(@base, 10), 20%);
@darker1: darken(spin(@base, -5), 10%);
@darker2: darken(spin(@base, -10), 20%);

.one  {color: @base;}
.two  {color: @lighter1;}
.three {color: @lighter2;}
.four {color: @darker1;}
.five {color: @darker2;}
```

SASS (SCSS)  CSS OUTPUT
DEBUGGING

SASS

> sass --watch path/to/your/sass --debug-info

Enable debug info in your logs.
Chrome DevTools requires CSS source maps support.

Firefox
UNIT CONVERSION

PIXELS TO EMS

\[
\text{font-size: } (18\text{px} / $\text{context}) \times 1\text{em} // 1.125\text{em}
\]

PIXELS TO %

\[
\text{$page-width:1024px;}
\]

\[
\text{.container { width: percentage(200px / $page-width); }
\text{margin: percentage(15px / $page-width); }
\]
Read about all the features Sass provides in the docs.

Sass Reference Docs
USING SASS IN A WORDPRESS APPLICATION
Generated CSS must abide by WordPress' naming and style conventions.

There must be a `style.css` file with the appropriate comments at the top of the file.

```html
/*
Theme Name: MyTheme
Theme URI: http://example.com
Description: My cool new site.
Author: Enrique Canals
Author URI: enriquecanals.
Version: 1.0
*/
```
USING SASS IN A WORDPRESS APPLICATION

There's also the WP-SCSS plugin, which provides compiling and configuration options for writing Sass for your WordPress themes.

Source is also available on GitHub: https://github.com/ConnectThink/WP-SCSS
WP-SCSS Settings

WP-SCSS requires both directories be specified. Please update your settings.

Version 1.1.6
By: Connect Think
Help & Issues: Github

Configure Paths
Add the paths to your directories below. Paths should start with the root of your theme. example: */library/scss/*

Scss Location
CSS Location

Compiling Options
Choose how you would like SCSS to be compiled and how you would like the plugin to handle errors

Compiling Mode
Errors Display

Enqueuing Options
WP-SCSS can enqueue your css stylesheets in the header automatically.

Enqueue Stylesheets

Save Changes
THE END

Thank you
SPECIAL THANKS

TO ALL OF THE CREATORS OF THE TOOLS MENTIONED HERE:

- Natalie Weizenbaum & Hampton Catlin (Sass)
- TJ Holowaychuk (Stylus)
- Alexis Sellier (Less)
- Chris Eppstein (Compass)
- Mario Ricalde (TextMate SCSS Bundle)
- Hakim El Hattab for Reveal.js (how this presentation was made)
- Many more not mentioned...
LINKS

1. Reveal.js
2. The Sass Way
3. Compass
4. Sass
5. Sass Reference
6. Less
7. Stylus
8. TextMate SCSS Syntax Highlighting Bundle
9. enriquecanals.com
10. enriquecanals@gmail.com