FROM CSS TO SASS IN WORDPRESS

James Steinbach   WP Summit 2014

@jdsteinbach   #wpsummit

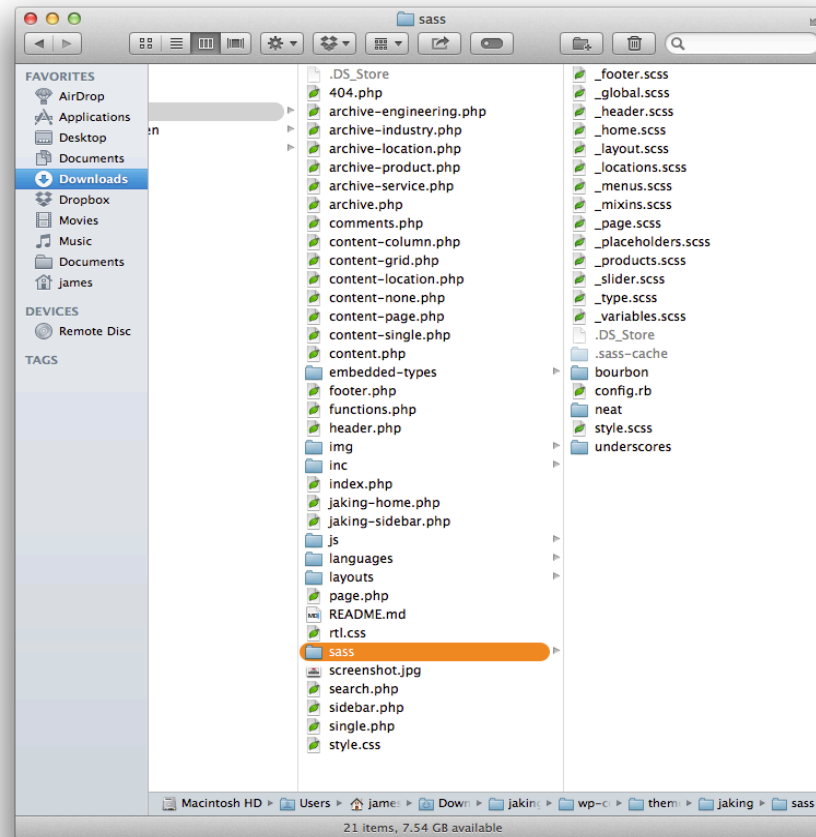# BENEFITS OF SASS

- Code organization (partials, nesting)

- Faster styling (mixins, functions)

- Scalable code (logic, variables)

NOT LIKE THIS:

3

MORE LIKE THIS:

CONVERT A STYLESHEET TO SASS

So where do we even start?

THE BARE MINIMUM

- CSS is valid Sass:
    - change style.css to style.scss
    - edit
    - compile back to style.css

SETTING UP PARTIALS

- Copy each "section" of style.css to a separate .scss file in your /sass/ folder.

- Name the file after what it does with an underscore prefix.

  – Example: styles relating to your navigation go to _navigation.scss

- Import that partial in your main style.scss file.

  – ```@import 'navigation';```

REFACTOR AS NECESSARY

```
.header .logo {
  //styles
}

.header .tagline {
  //styles
}

.header .menu {
  //styles
}
```

```
.header {

  .logo {
    //styles
  }

  .tagline {
    //styles
  }

  .menu {
    //styles
  }

}
```

REFACTOR AS NECESSARY

```
.menu a {
  -webkit-transition: color 0.2s;
  -moz-transition: color 0.2s;
  transition: color 0.2s;
}

.menu a {
  @include transition(color 0.2s);
}
```

COMPILE YOUR SASS FOR WP

Now that we broke it down, how do we put it back together?

WP REQUIREMENTS

- Theme root contains style.css with WP comments

- All Sass files in a subdirectory to keep theme root clean

- Ability to compress output for deployed code

- Avoid long, complex command line code

CLI COMPILE METHODS

- Sass CLI

  – Compile:
    ```
    sass style.scss style.css
    ```

  – Watch:
    ```
    sass --watch style.scss style.css
    ```

- Compass CLI

  – Watch:
    ```
    compass watch
    ```

GUI COMPILE METHODS

- Codekit: incident57.com/codekit

- Koala: koala-app.com

- Scout: mhs.github.io/scout-app

- Compass.app: compass.kkbox.com

COMPASS WATCH

- Easiest CLI syntax:
  `compass watch`

- Config.rb file for options:
  - File path (keep style.css in the right place)
  - Output
    `expanded`
    `compressed`

MY COMPASS CONFIG.RB

- Lives in /theme/sass/

  – style.css goes up a level to theme root
  `css_dir = '..'`

  – sass files live in same /sass/ directory as config.rb
  `sass_dir = ''`

  – development setting
  `output_style = :expanded`

  – deployment setting
  `output_style = :compressed`

WP STYLE.CSS COMMENTS

- style.scss in /theme/sass/

- Include '!' to preserve comments in :compressed output

```
/*!
Theme Name: Sassy Theme
Theme URI: http://jamessteinbach.com/
Author: James Steinbach
Author URI: http://jamessteinbach.com
Description: From CSS to Sass
*/

@import 'variables';
@import 'bourbon/bourbon';
//etc all your other imports…
```

ORGANIZE YOUR PARTIALS

Can I make all these new files easier to maintain?

CASCADES STILL MATTER

- Import your broadest styles first.

- Import your most specific styles last.

USE FOLDERS

- Group your .scss partials into folders:
  – Base (variables, mixins, reset, typography)
  – Layout (grid, header, footer)
  – Components (buttons, menus, forms, widgets)
  – Pages (home, landing page, portfolio)
  – 3rd Party (plugins, vendors, etc)

FULL DISCLOSURE

- My Sass folder structure is not usually this detailed.
  - /bourbon/
  - /neat/
  - /underscores/ (blank theme styles)
  - All my partials (from _variables.scss to _home.scss)

TRY THESE SASS TOOLS

Are there any other tricks for making this workflow better?

BOURBON

- Advanced Measurement/Color Functions
  - `golden-ratio(16px, 1);`
    returns 25.888px

  - `strip-units(12px);`
    returns 12

  - `tint(#330000, 50%);`
    returns #997f7f;

  - `rem(24px);`
    default $em-base: 16px, returns 1.5rem

BOURBON

- Prefixing Mixins
  - `@include animation();`
  - `@include background-image();`
  - `@include columns();`
  - `@include flexbox();`
  - `@include transform();`
  - `@include font-feature-settings();`

BOURBON NEAT

```scss
$grid-columns: 12;

$gutter: 1em;

.container {
  @include outer-container();
}

.grid-item {
  @include span-columns(3);
}
```

MY MIXINS: BREAKPOINTS

```scss
$breakpoints: (small: 600px, large: 920px);

@mixin breakpoint($name) {
  @if not index(map-keys($breakpoints),
$name) {
    @warn "Invalid breakpoint '#{$name}'.";
  } @else {
    @media (min-width: map-
get($breakpoints, $name)) {
      @content;
    }
  }
}
```

MY MIXINS: FONT SIZES

```
$font-sizes: (sm: 14, p: 16, bq: 20,
ssh: 24, sh: 30, h: 32, hero: 48);

@each $label, $size in $font-sizes {
  %#{$label} {
    font-size: $size * 1px;
    font-size: rem($size);
}
}

.page-title { @extend %h; }
```

MY MIXINS: ABS. POSITION

```
@mixin abs($t,$r,$b,$l) {
  position: absolute;
  top: $t;
  right: $r;
  bottom: $b;
  left: $l;
}
```

MY MIXINS: FONT-SMOOTHING

```scss
@mixin font-smoothing($val) {
  @if $val == 'antialiased' or $val ==
'subpixel-antialiased' {
    -webkit-font-smoothing: $val;
    -moz-osx-font-smoothing: $val;
  } @else {
    @warn 'Invalid value.'
  }
}
```

RESOURCES

Where do I get more information?

ADDITIONAL READING

- "Compass Compiling and WordPress Themes," Chris Coyier - css-tricks.com/compass-compiling-and-wordpress-themes/

- "How to Use Sass with WordPress," Andy Leverenz - elegantthemes.com/blog/tips-tricks/how-to-use-sass-with-wordpress-a-step-by-step-guide

- "Get Your Sass in Line," Micah Godbolt - godbolt.me/blog/get-your-sass-in-line.html

- "Architecture for a Sass Project," Hugo Giraudel - sitepoint.com/architecture-sass-project/

DOCUMENTATION

- Sass: sass-lang.com

- Compass: compass-style.org

- Bourbon: bourbon.io/docs

- Bourbon Neat: neat.bourbon.io/docs

- SassMeister App: sassmeister.com

JAMES STEINBACH

Senior Front End Developer
The Idea People in Charlotte, NC

@jdsteinbach | jamessteinbach.com/blog