The 7–1 Pattern

In this lesson, we'll learn how to organize our projects using the 7-1 pattern.

WE'LL COVER THE FOLLOWING

The 7-1 pattern

The architecture known as the **7–1 pattern** (7 folders, 1 file), is a widely-adopted document structure that serves as a basis for large projects.

You have all your partials organised into 7 different folders, and a single file sits at the root level (usually named `main.scss`) to handle the imports — this is the file that compiles into CSS.

What follows is a sample 7–1 directory structure. I've included some examples of files that would sit inside of each folder:

```
sass/
|
|– abstracts/ (or utilities/)
|   |– _variables.scss    // Sass Variables
|   |– _functions.scss    // Sass Functions
|   |– _mixins.scss       // Sass Mixins
|   |- _mixins.scss       // Sass Mixins
|
|- base/
|   |- _reset.scss        // Reset/normalize
|   |- _typography.scss   // Typography rules
|
|- components/ (or modules/)
|   |- _buttons.scss      // Buttons
|   |- _carousel.scss     // Carousel
|   |- _slider.scss       // Slider
|
|- layout/
|   |- _navigation.scss   // Navigation
|   |- _grid.scss         // Grid system
|   |- _header.scss       // Header
|   |- _footer.scss       // Footer
|   |- _sidebar.scss      // Sidebar
|   |- _forms.scss        // Forms
|
|- pages/
|   |- _home.scss         // Home specific styles
|   |- _about.scss        // About specific styles
|   |- _contact.scss      // Contact specific styles
|
|- themes/
|   |- _theme.scss        // Default theme
|   |- _admin.scss        // Admin theme
|
|- vendors/
|   |- _bootstrap.scss    // Bootstrap
|   |- _jquery-ui.scss    // jQuery UI
|
`- main.scss              // Main Sass file
```

Let’s now look at the typical contents within each folder.

Abstracts (or utilities)

Contains SASS tools, helper files, variables, functions, mixins, and other config files. These files are just meant to be helpers which don’t output any CSS when compiled.

Base

Contains the boilerplate code for the project. Including standard styles such as resets and typographic rules, which are commonly used throughout your resets and typographic rules, which are commonly used throughout your project.

Components (or modules)

Contains all of your styles for buttons, carousels, sliders, and similar page components (kind of like widgets). Your project will typically contain a lot of component files — as the whole site/app should be mostly composed of small modules.

Layout

Contains all styles involved with the layout of your project. Such as styles for your header, footer, navigation and the grid system.

Pages

Any styles specific to individual pages will sit here. For example, it's not uncommon for the home page of your site to require page specific styles that no other page receives.

Themes

This is likely not used in many projects. It would hold files that create project specific themes. For example, if sections of your site contain alternate color schemes.

Vendors

Vendors contain all third party code from external libraries and frameworks — such as Normalize, Bootstrap, jQueryUI, etc. However, there is often a need to override vendor code. If this is required, it's good practice to create a new folder called vendors-extensions/ and then name any files after the vendors they overwrite. The file `vendors-extensions/_bootstrap.scss` would contain all your Bootstrap overrides — as editing the vendor files directly isn't generally a good idea.

Main.scss

This file should only contain your imports!

For example:

```
@import 'abstracts/variables';
@import 'abstracts/functions';
@import 'abstracts/mixins';

@import 'vendors/bootstrap';
@import 'vendors/jquery-ui';

@import 'base/reset';
@import 'base/typography';

@import 'layout/navigation';
@import 'layout/grid';
@import 'layout/header';
@import 'layout/footer';
@import 'layout/sidebar';
@import 'layout/forms';

@import 'components/buttons';
@import 'components/carousel';
@import 'components/slider';
@import 'components/images';

@import 'pages/home';
@import 'pages/about';
@import 'pages/contact';

@import 'themes/theme';
@import 'themes/admin';
```

Note: There's no need to include the `_` or `.scss` file extension when importing.

Get up and running with 7–1:

Instead of creating this structure manually, you can grab the official boilerplate from Github.

You can download or clone it with the following terminal command:

```
git clone https://github.com/HugoGiraudel/sass-boilerplate.git
```

And that's it! You've learned how you can go about structuring your SASS projects.

The thing to keep in mind is that there are no explicit rules. You should always structure your projects in a way that is meaningful to you (and your team!). The way that helps you quickly and easily find and isolate your styles is the way to go!

In the next section, we're going to finally learn how to setup & run the SASS build process.

Quick Quiz!

1 In the 7-1 pattern, which folder would ideally contain the files for our mixins?

COMPLETED 0%

1 of 2