



# e-portfolio SASS

CSS with  
superpowers

# Structure

- Introduction to CSS
- Problems of CSS
- What is SASS
- Features of SASS
- Live Demo

# Introduction to CSS

- Cascading Style Sheets
- Defines **how to display** specific HTML elements
- Basic Selectors:
  - **Element Selector**: elements based on the element name (e.g. body, div, p)
  - **Class Selector**: elements with the specific class name (e.g. .container, .heading, .panel)
  - **Id Selector**: one specific element based on the unique id (e.g. #main, #content, #footer)
- Selectors can be combined

# Problems of CSS

- CSS files can get very long
- Repeating code
- Nightmare to change value in single file
- Working on one file with a team is a issue
- Slight variations cause many lines of code
- No logic base styling





# What is SASS?

- Syntactically **A**wesome **S**tyle **S**heets
- Pre-processor for CSS
- write cleaner code and prevent repeating code
- make coding process simpler and more efficient
- Two formatting conventions (based on file extension)
  - .SASS (HAML syntax)
  - .SCSS (recommended)

# .SASS vs. .SCSS

## SASS

```
@mixin button-base()  
  @include typography(button)  
  @include ripple-surface  
  @include ripple-radius-bounded  
  
  display: inline-flex  
  position: relative  
  height: $button-height  
  border: none  
  vertical-align: middle  
  
  &:hover  
    cursor: pointer  
  
  &:disabled  
    color: $mdc-button-disabled-ink-color  
    cursor: default  
    pointer-events: none
```

## SCSS

```
@mixin button-base() {  
  @include typography(button);  
  @include ripple-surface;  
  @include ripple-radius-bounded;  
  
  display: inline-flex;  
  position: relative;  
  height: $button-height;  
  border: none;  
  vertical-align: middle;  
  
  &:hover { cursor: pointer; }  
  
  &:disabled {  
    color: $mdc-button-disabled-ink-color;  
    cursor: default;  
    pointer-events: none;  
  }  
}
```

# How to use it

- Compile one file: `sass styles.scss:styles.css`
- Compile directory: `sass scss:css`
- Watch file: `sass --watch styles.scss:styles.css`
- Watch directory: `sass --watch scss:css`
- Compress output: `sass scss:css --style compressed`

# Features

- Nesting
- Variables
- Interpolation
- Partials
- Mixins
- Inheritance
- Flow control
- Built-in functions
- ...



# Nesting

- Mimic your HTML hierarchy
- Reduces length and complexity of selectors

# Nesting

```
nav {  
  ul {  
    margin: 0;  
    padding: 0;  
    list-style: none;  
  }  
  
  li { display: inline-block; }  
  
  a {  
    display: block;  
    padding: 6px 12px;  
    text-decoration: none;  
  }  
}
```

```
nav ul {  
  margin: 0;  
  padding: 0;  
  list-style: none;  
}  
  
nav li {  
  display: inline-block;  
}  
  
nav a {  
  display: block;  
  padding: 6px 12px;  
  text-decoration: none;  
}
```

# Parent Selector

- Used in nested selectors
- refers to the outer selector
- re-use the outer selector in more complex ways

# Parent Selector

```
nav {  
  a {  
    background-color: red;  
    &:hover {  
      background-color: blue;  
    }  
  }  
}  
  
.text {  
  color: black;  
  
  &-red {  
    color: red;  
  }  
  &-blue {  
    color: blue;  
  }  
}
```

```
nav a {  
  background-color: red;  
}  
nav a:hover {  
  background-color: blue;  
}  
  
.text {  
  color: black;  
}  
.text-red {  
  color: red;  
}  
.text-blue {  
  color: blue;  
}
```



# Variables

- Can be declared anywhere
- Can be scoped
- reduce repetition, do complex math, configure libraries
- Types: Booleans, Numbers, Colors, Strings, Lists, Maps, Null

# Variables

```
$global-variable: global value;

.content {
  $local-variable: local value;
  global: $global-variable;
  local: $local-variable;
}

.sidebar {
  global: $global-variable;

  // This would fail, because $local-variable isn't in scope:
  // local: $local-variable;
}
```

```
.content {
  global: global value;
  local: local value;
}

.sidebar {
  global: global value;
}
```

# Interpolation

- Embeds the result of an expression into the CSS
- can be used almost anywhere
- Fun fact: You can use interpolation in comments

# Interpolation

```
$stop-or-bottom: top;  
$left-or-right: left;  
$logo: 'logo-file-name';  
$logo-width: 30px;  
  
.logo {  
  #{Stop-or-bottom}: 0;  
  #{Left-or-right}: 0;  
  background-image: url("/icons/#{$logo}.svg");  
  width: #{ $logo-width + 2*10px };  
}
```

```
.logo {  
  top: 0;  
  left: 0;  
  background-image: url("/icons/logo-file-name.svg");  
  width: 50px;  
}
```



# Partials

- SASS files that are only imported in other files
- Make code modular and easier to maintain
- Filename is prefixed by an underscore: **\_partial.scss**
- Will be embedded in the importing file
- Will not be printed into own file
- Variables and other SASS features get imported too

# Partials

```
// _reset.scss
```

```
html,  
body,  
ul,  
ol {  
  margin: 0;  
  padding: 0;  
}
```

```
// basefile.scss
```

```
@import 'reset';
```

```
body {  
  font: 100% Helvetica, sans-serif;  
  background-color: #efefef;  
}
```

```
html, body, ul, ol {  
  margin: 0;  
  padding: 0;  
}
```

```
body {  
  font: 100% Helvetica, sans-serif;  
  background-color: #efefef;  
}
```

# Mixins

- Reusable code
- Can take arguments

# Mixins

```
@mixin square($size, $radius: 0) {  
  width: $size;  
  height: $size;  
  border-radius: $radius;  
}  
  
.avatar {  
  @include square(100px);  
}  
  
.avatar-round {  
  @include square(100px, 10px);  
  border-color: black;  
}
```

```
.avatar {  
  width: 100px;  
  height: 100px;  
  border-radius: 0;  
}  
  
.avatar-round {  
  width: 100px;  
  height: 100px;  
  border-radius: 10px;  
  border-color: black;  
}
```



# Inheritance

- Inherit the styles of another selector
- Handles nested selectors
- updates the selector instead of copying the code (like a mixin would)
- Placeholder Selectors (%) can be extended but are not present in the CSS

# Inheritance

```
.foreground {  
  color: red;  
  
  &:hover {  
    color: blue;  
  }  
}  
  
%background {  
  background: green;  
}  
  
.c {  
  @extend .foreground;  
  @extend %background;  
  width: 100%;  
}
```

```
.foreground, .c {  
  color: red;  
}  
  
.foreground:hover, .c:hover {  
  color: blue;  
}  
  
.c {  
  background: green;  
}  
  
.c {  
  width: 100%;  
}
```

# Flow Control

- Makes SASS more like a programming language
- Control whether styles get emitted, or to emit them multiple times
- If / Else If / Else
- For Each
- For Range
- While

# Flow Control

```
Stop-or-bottom: top;
Sleft-or-right: left;

.logo {
  @if (not ($stop-or-bottom == 'top' or $stop-or-bottom == 'bottom')) {
    @error '#{Stop-or-bottom} not allowed in Stop-or-bottom. Can only be top or bottom';
  }
  @if (not ($sleft-or-right == 'left' or $sleft-or-right == 'right')) {
    @error '#{Stop-or-bottom} not allowed in Sleft-or-right. Can only be left or right';
  }
  #{Stop-or-bottom}: 0;
  #{Sleft-or-right}: 0;
}
```



# Built-in functions

- Can be used in attribute values, Interpolation, Variables, Mixins, ...
- Colors (lighten, darken, ...)
- Lists (append, index, ...)
- Maps (has-key, merge, ...)
- Math (floor, abs, ...)
- Meta (variable-exists, ...)
- Selectors (extend, replace, ...)
- String (slice, to-upper-case, ...)



**LIVE DEMO**

# Thank you for your attention

If you want to learn more about SASS, go to <https://sass-lang.com>