SASS

Syntactically Awesome Style Sheets

SoftUni Team

Technical Trainers

Software University

Software University

https://softuni.org

Table of Contents

1. What is SASS?
2. SASS Features
   - Nesting & Referencing Parent Selector
   - Interpolation
   - Placeholder Selectors
   - Variables
   - Maps
   - Comments
   - Loops
   - Mixins and Content Blocks

Have a Question?

Software University

sli.do

#front-end

SASS

Syntactically Awesome Style Sheets

What is SASS?

- Most mature, stable, and powerful professional grade **CSS extension language**
  - CSS Compatible
  - Feature Rich
  - Mature
  - Industry Approved
  - Large Community

SASS Features

Variables, nested rules, mixins, functions, and more

SASS Features

- Sass is a stylesheet language that's **compiled to CSS**
- It allows you to use:
  - Variables
  - Nested rules
  - Mixins
  - Functions

Nesting

- One style rules **inside** another
  - Sass will automatically combine the outer rule's selector with the inner rule's

```
nav ul {
   margin: 0;
   padding: 0;
   list-style: none;
}
nav li {
   display: inline-block;
}
```

CSS Styles

```
nav a {
   display: block;
   padding: 6px 12px;
   text-decoration: none;
}
```

Nesting

```
nav {
    ul {
        margin: 0;
        padding: 0;
        list-style: none;
    }

    li {
        display: inline-block;
    }

    a {
        display: block;
        padding: 6px 12px;
        text-decoration: none;
    }
}
```

SCSS

SASS - Referencing Parent Selector - '&'

- A special selector invented by **SASS** that's used in nested selectors to refer to the **outer selector**

```
a {
    color: orange;

    &:hover {
        text-decoration: underline;
    }

    &:active {
        color: red;
    }
}
```

SASS - Placeholder Selectors - '%FOO'

- Special kind of selector known as a "**placeholder**"
- It starts with a **%** and it's **not included** in the CSS output
- Any complex selector that even contains a placeholder selector **isn't included in the CSS**

```
%pseudo {
    display: block;
    content: '';
}

.media:before {
    @extend %pseudo;
}
```

SASS - Placeholder Selectors - '%FOO'

```scss
%toolbelt {
    box-sizing: border-box;
    border-top: 1px rgba(#000, 0.12) solid;
    padding: 16px 0;
    width: 100%;

    &:hover {
        border: 2px rgba(#000, 0.5) solid;
    }
}
.action-buttons {
    @extend %toolbelt;
    color: #4285f4;
}
.reset-buttons {
    @extend %toolbelt;
    color: #cddc39;
}
```

```css
.action-buttons, .reset-buttons {
    box-sizing: border-box;
    border-top: 1px rgba(0, 0, 0, 0.12) solid;
    padding: 16px 0;
    width: 100%;
}
.action-buttons:hover,
.reset-buttons:hover {
    border: 2px rgba(0, 0, 0, 0.5) solid;
}
.action-buttons {
    color: #4285f4;
}
.reset-buttons {
    color: #cddc39;
}
```

SASS Variables

- Assign a value to a name that begins with **$**
- Refer to that **name** instead of the value itself

```
$colorBodyBackground: #EEE;

body {
    background-color: $colorBodyBackground;
}
```

- Only available within the **level** of nested selectors where they're defined

SASS Maps

- Maps in Sass hold pairs of **keys** and **values**
- Easy to look up a value by its corresponding key
- They're written (`<expression>: <expression>`, `<expression>: <expression>`)
  - The keys must be **unique**
  - The values may be **duplicated**
  - Must be written with **parentheses** around them
  - A map with no pairs is written **()**

SASS Maps

```scss
$icons: ("eye": "\f112", "start": "\f12e", "stop": "\f12f");

@each $name, $glyph in $icons {
  .icon-#{$name}:before {
    display: inline-block;
    font-family: "Icon Font";
    content: $glyph;
  }
}
```

SASS Comments

- SASS supports two types of comments
  - **Multiline comments** - preserved in CSS output /* */
  - **Single line comments** - not preserved in CSS output //

```
/* This will be compiled in the CSS file */

// This will not be compiled
```

SASS For Loop

- Counts **up** or **down** from one number to and evaluates a block for each number in between

```
@for $i from 1 through 3 {
    .item-#{$i} {
      width: 2em * $i;
    }
}
```

SASS For Loop

SCSS

```scss
$base-color: #036;

@for $i from 1 through 3 {
  ul:nth-child(3n + #{$i}) {
    background-color: lighten($base-color, $i * 5%);
  }
}
```

CSS

```css
ul:nth-child(3n + 1) {
  background-color: #004080;
}

ul:nth-child(3n + 2) {
  background-color: #004d99;
}

ul:nth-child(3n + 3) {
  background-color: #0059b3;
}
```

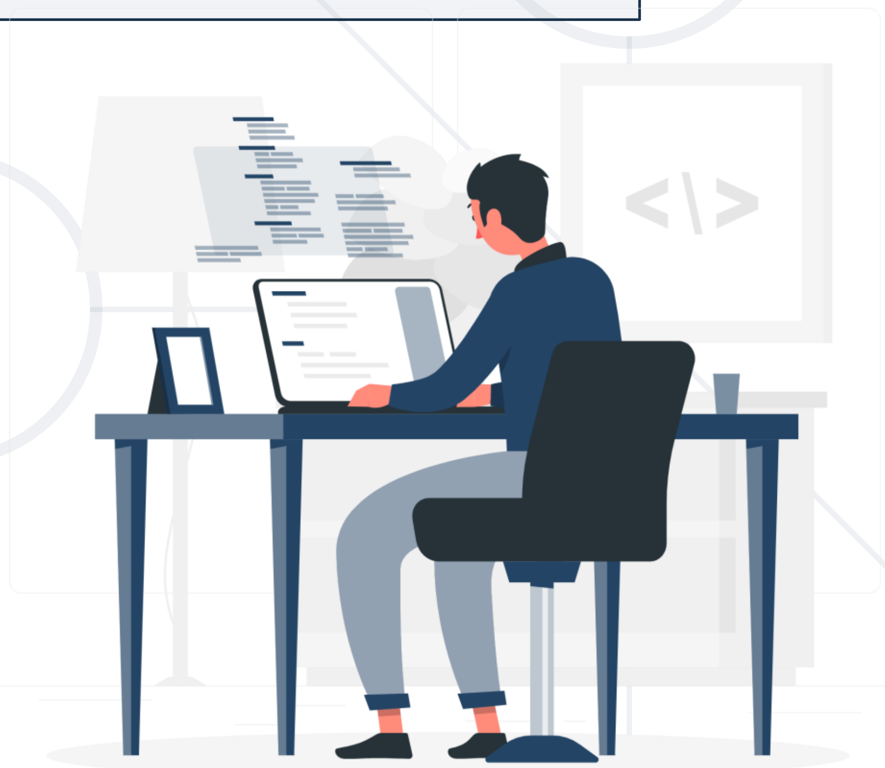
SASS Each Loop

- Emit styles or evaluate code for **each element** of a list or each pair in a map
  - Great for **repetitive styles** that only have a few variations between them

```
@each $animal in puma, sea-slug, egret, salamander {
    .#{$animal}-icon {
        background-image: url('/images/#{$animal}.png');
    }
}
```

SASS Each Loop

```
@each $header, $size in (h1: 2em, h2: 1.5em, h3: 1.2em) {
    #{$header} {
        font-size: $size;
    }
}
```



SASS Mixins

- Allow you to define styles that can be **re-used** throughout your stylesheet
  - It's easy to avoid using non-semantic classes and to **distribute collections of styles** in libraries

```
@mixin fa {
    font-family: 'FontAwesome';
    display: inline-block;
}

a.button:before {
    @include fa;
}
```

SASS - Interpolation - '#{ }'

- Just wrap an expression in #{}:

SCSS

```scss
@mixin corner-icon($name, $top-or-bottom, $left-or-right) {
    .icon-#{$name} {
      background-image: url("/icons/#{$name}.svg");
      position: absolute;
      #{$top-or-bottom}: 0;
      #{$left-or-right}: 0;
    }
}

@include corner-icon("mail", top, left);
```

CSS

```css
.icon-mail {
    background-image: url('/icons/mail.svg');
    position: absolute;
    top: 0;
    left: 0;
}
```

SASS Content Blocks

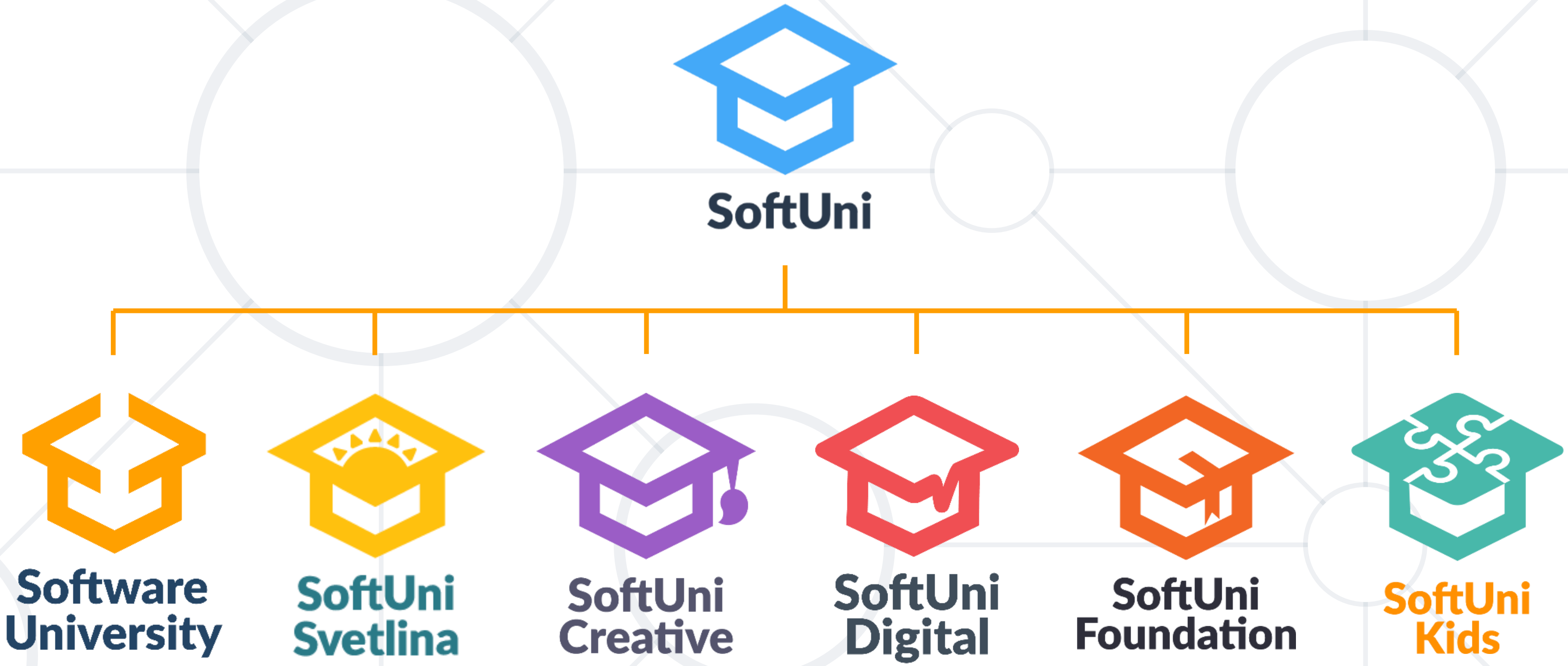
- It is possible to **pass** a block of styles to the mixin for placement within the styles **included** by the mixin
  - The styles will appear at the location of any **@content** directives found within the mixin

```
@mixin mq() {
    @media only screen and (min-width: 1044px) {
        @content;
    }
}
.site-header {
    @include mq {
        .site-title { paddin-top: 10rem; }
    }
}
```

Summary

- Nesting & Referencing Parent Selector
- Interpolation
- Placeholder Selectors
- Variables
- Maps
- Comments
- Loops
- Mixins and Content Blocks

Questions?


SoftUni
Software University
SoftUni Svetlina
SoftUni Creative
SoftUni Digital
SoftUni Foundation
SoftUni Kids

SoftUni Diamond Partners

NETPEAK

Postbank
Решения за твоето утре

SoftwareGroup
doing it right

XS SOFTWARE

SmartIT

Coca-Cola HBC
Bulgaria

INDEAVR
Serving the high achievers

tek experts

SBTech
we know sports

zühlke
empowering ideas

INFRAGISTICS®

SUPERHOSTING.BG

telenor

License

Trainings @ Software University (SoftUni)

- Software University - High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
  - Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg