Sass: Syntactically Awesome Stylesheet

Laura Farinetti
Dipartimento di Automatica e Informatica
Politecnico di Torino
laura.farinetti@polito.it

Sass

- Syntactically Awesome Stylesheet
- Sass is an extension of CSS that adds power and elegance to the basic language, adding nested rules, variables, mixins, selector inheritance, and more
- It is translated to well-formatted, standard CSS using the command line tool or a web-framework plugin
  - A valid CSS stylesheet is also a valid Sass stylesheet

Sass syntaxes

- Sass has two syntaxes
- The new main syntax is known as “SCSS” (for “Sassy CSS”), and is a superset of CSS3 syntax
  - Every valid CSS3 stylesheet is valid SCSS as well
  - SCSS files use the extension .scss
- The second, older syntax is known as the “indented” syntax (or just “Sass”)
  - Intended for people who prefer conciseness over similarity to CSS: Instead of brackets and semicolons, it uses the indentation of lines to specify blocks
  - Although no longer the primary syntax, the indented syntax will continue to be supported
  - Files in the indented syntax use the extension .sass
- Files can be automatically converted from one syntax to the other

Nesting

- Often in CSS there are several selectors that begin in the same way
- Sass avoids repetition by nesting selectors within one another

- CSS3

```
.center {
  text-align: center;
}

.center h1 {
  margin-bottom: 10px;
}
```

Saas

```
.center {
  text-align: center;
  h1 {
    margin-bottom: 10px;
  }
}
```

Nesting

- Another example

```
/* style.css */

#navbar {
  width: 80%;
  height: 23px; }
  #navbar ul {
    list-style-type: none; }
  #navbar li {
    float: left; }
    #navbar li a {
      font-weight: bold; }
```

```
/* style.scss */

#navbar {
  width: 80%;
  height: 23px;

  ul { list-style-type: none; }
  li {
    float: left;
    a { font-weight: bold; }
  }
}
```

http://saas.lang.com

Nesting

- Properties can be nested too

```
/* style.css */

.fakeshadow {
  border-style: solid;
  border-left-width: 4px;
  border-left-color: #888;
  border-right-width: 2px;
  border-right-color: #ccc; }
```

```
/* style.scss */

.fakeshadow {
  border: {
    style: solid;
    left: {
      width: 4px;
      color: #888;
    }
    right: {
      width: 2px;
      color: #ccc;
    }
  }
}
```

Nesting

- Pseudoclasses can be nested too (e.g. :hover)
- Sass special character &
  - in a selector & is replaced with the parent selector

```
/* style.css */

a {
  color: #ce4dd6; }
  a:hover {
    color: #ffb3ff; }
  a:visited {
    color: #c458cb; }
```

```
/* style.scss */

a {
  color: #ce4dd6;
  &:hover { color: #ffb3ff; }
  &:visited { color: #c458cb; }
}
```

Nesting: Sass vs. Scss

```
/* CSS */

table.hl {
  margin: 2em 0;
}
table.hl td.ln {
  text-align: right;
}

li {
  font-family: serif;
  font-weight: bold;
  font-size: 1.2em;
}
```

.scss .sass

```
table.hl {
  margin: 2em 0;
  td.ln {
    text-align: right;
  }
}

li {
  font: {
    family: serif;
    weight: bold;
    size: 1.2em;
  }
}
```

.scss .sass

```
table.hl
  margin: 2em 0
  td.ln
    text-align: right

li
  font:
    family: serif
    weight: bold
    size: 1.2em
```

Variables

- Sass allows to declare variables that can be used throughout the stylesheet
- Variables begin with $ and are declared like properties
  - They can have any value that's allowed for a CSS property, such as colors, numbers, or text

```
/* style.css */

#navbar {
  border-bottom-color: #ce4dd6;
  border-bottom-style: solid; }

a {
  color: #ce4dd6; }
  a:hover {
    border-bottom: solid 1px; }
```

```
/* style.scss */

$main-color: #ce4dd6;
$style: solid;

#navbar {
  border-bottom: {
    color: $main-color;
    style: $style;
  }
}

a {
  color: $main-color;
  &:hover { border-bottom: $style 1px; }
}
```

Operations and functions

- Sass supports basic math operations and many useful functions
  - http://sass-lang.com/docs/yardoc/Sass/Script/Functions.html
- The standard math operations (+, -, *, /, and %) are supported for numbers
- There are many useful functions for colors, e.g. to change lightness, hue, saturation, …

```
/* style.scss */

#navbar {
  $navbar-width: 800px;
  $items: 5;
  $navbar-color: #ce4dd6;

  width: $navbar-width;
  border-bottom: 2px solid $navbar-color;

  li {
    float: left;
    width: $navbar-width/$items - 10px;

    background-color:
      lighten($navbar-color, 20%);
    &:hover {
      background-color:
        lighten($navbar-color, 10%);
    }
  }
}
```

```
/* style.css */

#navbar {
  width: 800px;
  border-bottom: 2px solid #ce4dd6; }
  #navbar li {
    float: left;
    width: 150px;
    background-color: #e5a0e9; }
    #navbar li:hover {
      background-color: #d976e0; }
```

Interpolation

- Variables can be used for more than property values
- Using #{} they can be included into property names or selectors

```
/* style.css */

.rounded-top-left {
  border-top-radius: 10px;
  -moz-border-radius-top: 10px;
  -webkit-border-top-radius: 10px; }
```

```
/* style.scss */

$vert: top;
$horz: left;
$radius: 10px;

.rounded-#{$vert}-#{$horz} {
  border-#{$vert}-#{$horz}-radius: $radius;
  -moz-border-radius-#{$vert}#{$horz}: $radius;
  -webkit-border-#{$vert}-#{$horz}-radius: $radius;
}
```

Variables: Sass vs. Scss

```
/* CSS */

.content-navigation {
  border-color: #3bbfce;
  color: #2b9eab;
}

.border {
  padding: 8px;
  margin: 8px;
  border-color: #3bbfce;
}
```

.scss

```
$blue: #3bbfce;
$margin: 16px;

.content-navigation {
  border-color: $blue;
  color:
    darken($blue, 9%);
}

.border {
  padding: $margin / 2;
  margin: $margin / 2;
  border-color: $blue;
}
```

.sass

```
$blue: #3bbfce
$margin: 16px

.content-navigation
  border-color: $blue
  color: darken($blue, 9%)

.border
  padding: $margin / 2
  margin: $margin / 2
  border-color: $blue
```

Mixins

- Mixins allow to re-use whole chunks of CSS, properties or selectors
- Mixins are defined using the "@mixin" directive, which takes a block of styles that can then be included in another selector using the "@include" directive

```
/* style.scss */

@mixin rounded-top-left {
  $vert: top;
  $horz: left;
  $radius: 10px;

  border-#{$vert}-#{$horz}-radius: $radius;
  -moz-border-radius-#{$vert}#{$horz}: $radius;
  -webkit-border-#{$vert}-#{$horz}-radius: $radius;
}

#navbar li { @include rounded-top-left; }
#footer { @include rounded-top-left; }
```

```
/* style.css */

#navbar li {
  border-top-left-radius: 10px;
  -moz-border-radius-topleft: 10px;
  -webkit-border-top-left-radius: 10px; }

#footer {
  border-top-left-radius: 10px;
  -moz-border-radius-topleft: 10px;
  -webkit-border-top-left-radius: 10px; }
```

Arguments

- It is possible to pass arguments to the mixins
- Arguments are declared as a comma-separated list of variables inside parentheses
  - Each of those variables is assigned a value each time the mixin is used
  - Mixin arguments can also be given default values

```
/* style.css */

#navbar li {
  border-top-left-radius: 10px;
  -moz-border-radius-topleft: 10px;
  -webkit-border-top-left-radius: 10px; }

#footer {
  border-top-left-radius: 5px;
  -moz-border-radius-topleft: 5px;
  -webkit-border-top-left-radius: 5px; }

#sidebar {
  border-top-left-radius: 8px;
  -moz-border-radius-topleft: 8px;
  -webkit-border-top-left-radius: 8px; }
```

```
/* style.scss */

@mixin rounded($vert, $horz, $radius: 10px) {
  border-#{$vert}-#{$horz}-radius: $radius;
  -moz-border-radius-#{$vert}#{$horz}: $radius;
  -webkit-border-#{$vert}-#{$horz}-radius: $radius;
}

#navbar li { @include rounded(top, left); }
#footer { @include rounded(top, left, 5px); }
#sidebar { @include rounded(top, left, 8px); }
```

Mixins: Sass vs. Scss

```
/* CSS */

#data {
  float: left;
  margin-left: 10px;
}
#data th {
  text-align: center;
  font-weight: bold;
}
#data td, #data th {
  padding: 2px;
}
```

.scss

```
@mixin table-base {
  th {
    text-align: center;
    font-weight: bold;
  }
  td, th {padding: 2px}
}

@mixin left($dist) {
  float: left;
  margin-left: $dist;
}

#data {
  @include left(10px);
  @include table-base;
}
```

.sass

```
@mixin table-base
  th
    text-align: center
    font-weight: bold
  td, th
    padding: 2px

@mixin left($dist)
  float: left
  margin-left: $dist

#data
  @include left(10px)
  @include table-base
```

Selector inheritance

- Sass can tell one selector to inherit all the styles of another without duplicating the CSS properties

```
/* CSS */

.error, .badError {
  border: 1px #f00;
  background: #fdd;
}

.error.intrusion,
.badError.intrusion {
  font-size: 1.3em;
  font-weight: bold;
}

.badError {
  border-width: 3px;
}
```

.scss

```
.error {
  border: 1px #f00;
  background: #fdd;
}
.error.intrusion {
  font-size: 1.3em;
  font-weight: bold;
}

.badError {
  @extend .error;
  border-width: 3px;
}
```

.sass

```
.error
  border: 1px #f00
  background: #fdd

.error.intrusion
  font-size: 1.3em
  font-weight: bold

.badError
  @extend .error
  border-width: 3px
```

@import

```
/* style.scss */

@import "rounded";

#navbar li { @include rounded(top, left); }
#footer { @include rounded(top, left, 5px); }
#sidebar { @include rounded(top, left, 8px); }
```

```
/* style.css */

#navbar li {
  border-top-left-radius: 10px;
  -moz-border-radius-topleft: 10px;
  -webkit-border-top-left-radius: 10px; }

#footer {
  border-top-left-radius: 5px;
  -moz-border-radius-topleft: 5px;
  -webkit-border-top-left-radius: 5px; }

#sidebar {
  border-top-left-radius: 8px;
  -moz-border-radius-topleft: 8px;
  -webkit-border-top-left-radius: 8px; }
```

- Stylesheets can be big: the @import directive that allows to break styles up into multiple stylesheets

```
/* _rounded.scss */

@mixin rounded($vert, $horz, $radius: 10px) {
  border-#{$vert}-#{$horz}-radius: $radius;
  -moz-border-radius-#{$vert}#{$horz}: $radius;
  -webkit-border-#{$vert}-#{$horz}-radius: $radius;
}
```

References

- Sass {style with attitude}
  - http://sass-lang.com/
- CSS to Sass online converter
  - http://css2sass.heroku.com/

Licenza d'uso

- Queste diapositive sono distribuite con licenza Creative Commons "Attribuzione - Non commerciale - Condividi allo stesso modo 2.5 Italia (CC BY-NC-SA 2.5)"
- Sei libero:
  - di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera
  - di modificare quest'opera
- Alle seguenti condizioni:
  - **Attribuzione** — Devi attribuire la paternità dell'opera agli autori originali e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
  - **Non commerciale** — Non puoi usare quest'opera per fini commerciali.
  - **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.
- http://creativecommons.org/licenses/by-nc-sa/2.5/it/