CSS Pre-Processors

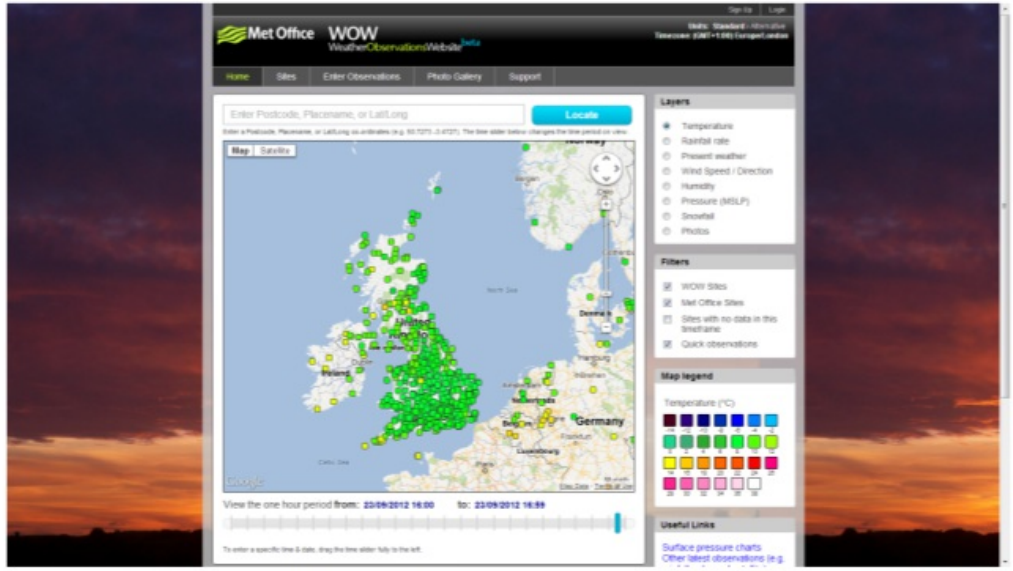
21st Century CSS

Steve Workman

@steveworkman

Who I am

- Mobile & Front-end Development Lead at PA Consulting Group
- Organiser of London Web Standards since 2010



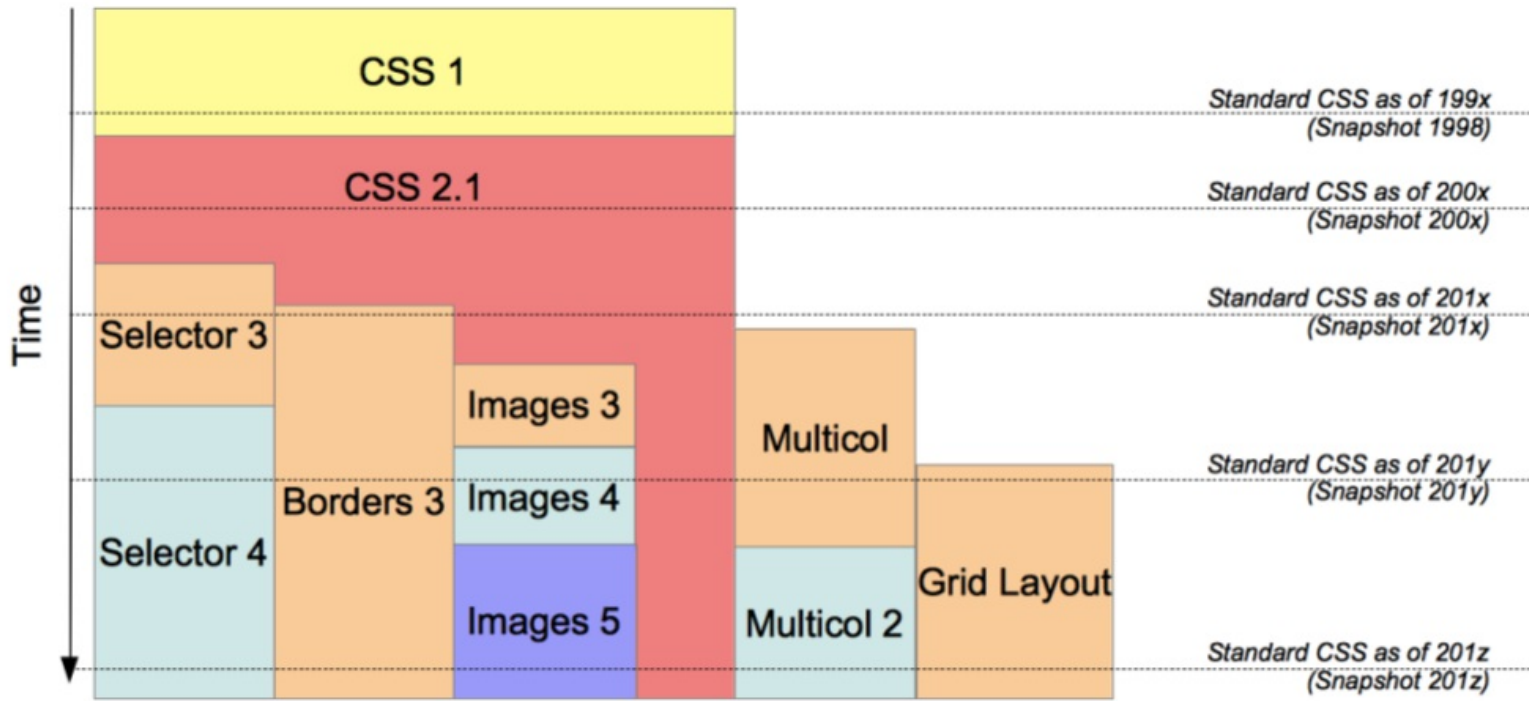
Say hello to the CSS WG

W3C CSS Working Group in 2010 - Photo by Daniel Glazman

A new kind of dependency hell

> *As a consequence of this design, feature modules couldn't progress unless the core modules progressed, and the core modules couldn't be complete until all the feature modules were complete.* ***Nothing could move forward until everything moved forward****. It was a new kind of dependency hell.*
> *- Elika J. Etemad AKA Fantasai, W3C*

Taking a Modular Approach



CSS Modules and snapshots - MDN

There's lots of work going on

Completed	Current	Upcoming	Notes
CSS Snapshot 2010	NOTE	–	Latest stable CSS
CSS Snapshot 2007	NOTE	–	
CSS Color Level 3	REC	REC	See Errata
CSS Namespaces	REC	REC	
Selectors Level 3	REC	REC	
CSS Level 2 Revision 1	REC	REC	See Errata
CSS Level 1	REC	–	Unmaintained, see Snapshot
Media Queries	REC	REC	

Stable	Current	Upcoming	Notes
CSS Style Attributes	CR	PR	

Testing	Current	Upcoming	Notes
CSS Backgrounds and Borders Level 3	CR	PR	
CSS Image Values and Replaced Content Level 3	CR	PR	
CSS Marquee	CR	PR	
CSS Multi-column Layout	CR	CR	
CSS Speech	CR	PR	
CSS Values and Units Level 3	CR	PR	
CSS Mobile Profile 2.0	CR	PR	Status unknown
CSS TV Profile 1.0	CR	?	Status unknown

Refining	Current	Upcoming	Notes
CSS Animations	WD	WD	
CSS Flexible Box Layout	CR	PR	
CSS Text Level 3	WD	WD	
CSS Fragmentation Level 3	WD	WD	
CSS Transforms	WD	WD	
CSS Transitions	WD	WD	
CSS Print Profile	LC	?	Status unknown

Revising	Current	Upcoming	Notes
CSS Conditional Rules Level 3	WD	WD	
CSS Fonts Level 3	WD	LC	
CSS Paged Media Level 3	LC	LC	Inactive
CSS Basic User Interface Level 3	CR	LC	
CSS Writing Modes Level 3	WD	WD	
CSSOM View	WD	WD	

Exploring	Current	Upcoming	Notes
CSS Cascading and Inheritance Level 3	WD	WD	Inactive
CSS Device Adaptation	WD	WD	
CSS Exclusions and Shapes	WD	WD	
CSS Generated Content for Paged Media	WD	WD	
CSS Grid Layout	WD	WD	
CSS Grid Template Layout	WD	WD	
CSS Line Grid	–	WD	
CSS Lists Level 3	WD	WD	
CSS Positioned Layout Level 3	WD	WD	
CSS Presentation Levels	WD	WD	Inactive
CSS Regions	WD	WD	
CSS Tables Level 3	–	WD	Inactive
Selectors Level 4	WD	WD	
CSS Object Model	WD	WD	
Compositing and Blending	WD	WD	
Filter Effects	–	WD	
CSS Box Alignment Module Level 3	WD	WD	
CSS Text Decoration Module Level 3	–	WD	
CSS Intrinsic & Extrinsic Sizing Module Level 3	WD	WD	
CSS Counter Styles Level 3	WD	WD	

Rewriting	Current	Upcoming	Notes
CSS Basic Box Model Level 3	WD	WD	Dangerously outdated; see CSS2.1.
CSS Generated Content Level 3	WD	WD	Severely outdated
CSS Line Layout Level 3	WD	WD	Severely outdated
CSS Ruby	WD	WD	Outdated and majorly underdefined
CSS Syntax Level 3	WD	WD	Severely outdated; see CSS2.1.

Abandoned	Current	Upcoming	Notes
Behavioral Extensions to CSS	WD	–	
CSS Hyperlink Presentation	WD	–	
CSS Grid Positioning	WD	–	

Calculations (CSS Values)

Example

```
1 section {
2   float: left;
3   margin: 1em; border: solid 1px;
4   // one col of 3 col layout minus 1em padding each side and 1px bord
5   width: calc(100%/3 - 2*1em - 2*1px);
6 }
```

CSS Variables Cascading Variables

> ***The proposed spec is NOT about variables*** *and I seriously wonder if we should not change the title of the document. You may call the feature it introduces "variables" but at the deeper level, that's not about variables.*

Daniel Glazman, CSSWG Co-Chair, August 2012
[CSS Cascading Variables is] a family of custom author-defined properties known collectively as **custom properties**, which allow an author to assign arbitrary values to a property with an author-chosen name, and **variables**, which allow an author to then use those values in other properties elsewhere in the document.

First requested in *1998*, first W3C proposal: *2008*, first public draft *April 2012*

A decade of BBC News

A decade of BBC News

2001

Lines of CSS: **1493**
Total File size: **31.85KB**
(1 file)

A decade of BBC News

2001

Lines of CSS: **1493**
Total File size: **31.85KB**
(1 file)

2007

Lines of CSS: **2270** *Up 52%*
Total File size: **61.93KB**
(12 files)

A decade of BBC News

2001

Lines of CSS: **1493**
Total File size: **31.85KB** (1 file)

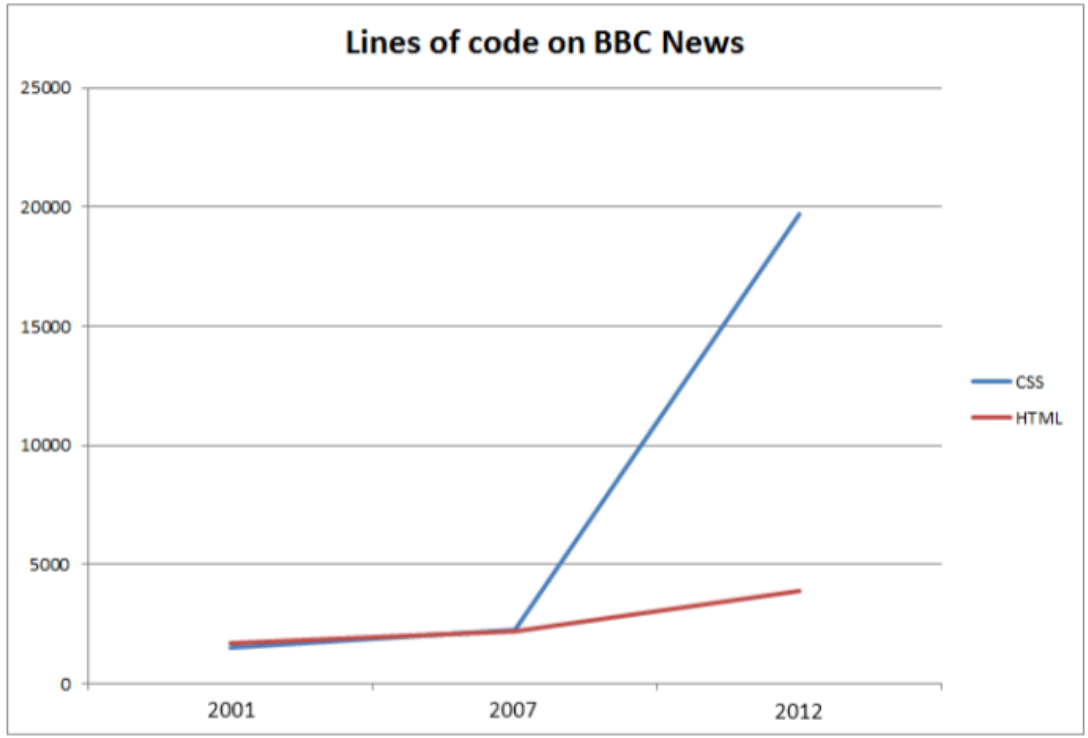
2007

Lines of CSS: **2270** *Up 52%*
Total File size: **61.93KB** (12 files)

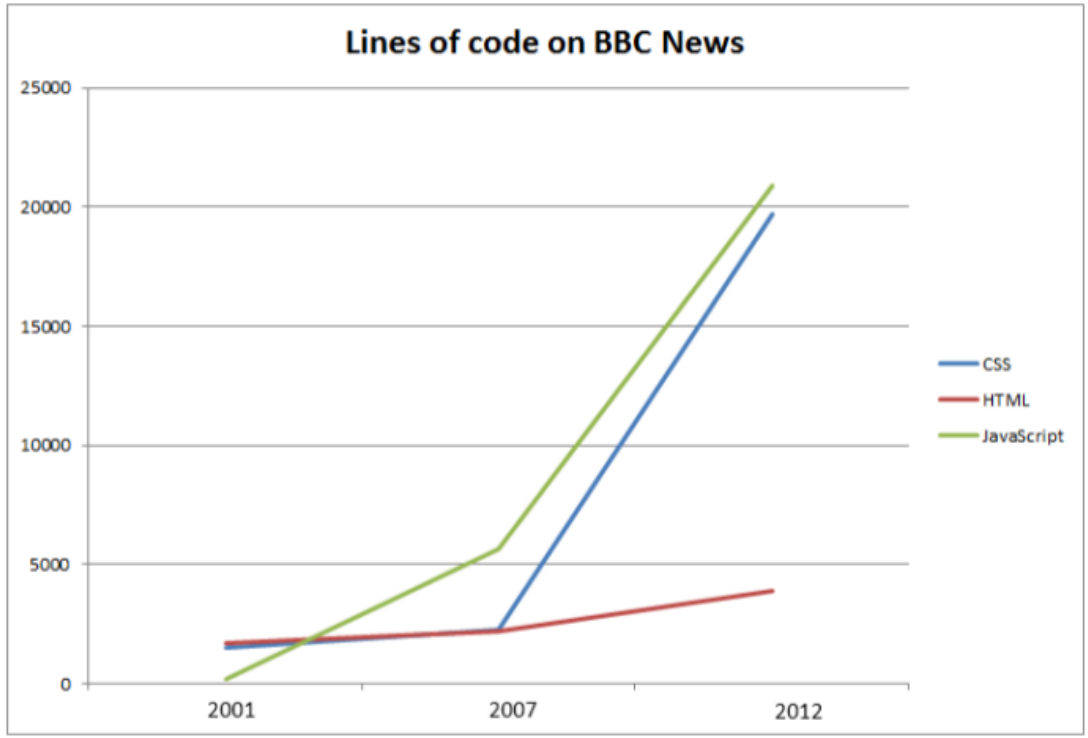
2012

Lines of CSS: **19712** *Up 860%!*
Total File size: **336.30KB** (12 files)

How much code?



How much code?



Why do we need these things?

Dilbert.com

Enter CSS Pre-processors

Both feature

- Nested rules
- Variables
- Mixins
- Logic & Loops
- Math functions
- Extensions
- Works with regular CSS

Nesting Rules

CSS

```
1  section {
2    margin: 10px;
3  }
4  section nav {
5    height: 25px;
6  }
7  section nav a, section nav a:vi
8    color: #0982C1;
9  }
10 section nav a:hover, section na
11   text-decoration: underline;
12 }
```

SASS

```
1  section {
2    margin:10px;
3    nav {
4      height:25px;
5      a, a:visited {
6        color: #0982C1;
7        &amp;:hover, &amp;:focus
8          text-decoration:underli
9        }
10     }
11   }
12 }
```

Nesting is really useful for @media

Standard CSS

```
.some-class {
   /* Default styling */
}

/* Hundreds of lines of CSS */

@media (max-width: 979px) {
  .some-class {
    /* Responsive styles */
  }
}
@media (min-width: 980px) {
  .some-class {
    /* Desktop responsive style
  }
}
```

SASS

```
.some-class {
  /* Default styling */
  @media (max-width: 979px) {
    /* Responsive styles */
  }
  @media (min-width: 980px) {
    /* Desktop responsive style
  }
}
```

Variables

Variables

- Define a property/value/string anywhere

Variables

- Define a property/value/string anywhere
- Use it again

Variables

- Define a property/value/string anywhere
- Use it again
- ...

Variables

- Define a property/value/string anywhere
- Use it again
- ...
- Profit!

Variables

- Define a property/value/string anywhere
- Use it again
- ...
- Profit!
- Less uses @, Sass uses $

$ or @

Neither implementation matches the current W3C spec, so why does it matter?

It doesn't matter, just be aware

$ - Selectors Level 4 Parent Selector

```
/* Puts a border around the li
ul $li p {
  border: 1px solid #ccc;
}

```

@ - Lots of usage

```
@media print {
  #sidebar {
    display:none;
    }
  }
@font-face {
  font-family: 'Museo500Regular
    url('../fonts/Museo500-Regu
    /* And lots more */
}
@keyframes identifier {
  0% { top: 0; left: 0; }
  50% { top: 50px; left: 50%; }
  100% { top: 100px; left: 100%
}
```

Mixins

CSS

```
1  header nav {
2    background-color: #333;
3    background-image: -moz-linear-gradient(left, #555, #333); // FF 3.6
4    background-image: -ms-linear-gradient(left, #555, #333); // IE10
5    background-image: -webkit-gradient(linear, 0 0, 100% 0, from(#555),
6    background-image: -webkit-linear-gradient(left, #555, #333); // Saf
7    background-image: -o-linear-gradient(left, #555, #333); // Opera 11
8    background-image: linear-gradient(left, #555, #333); // Le standard
9    background-repeat: repeat-x;
10   filter: e(%("progid:DXImageTransform.Microsoft.gradient(startColors
11 }
```

Mixins

CSS

```
1  header nav {
2    background-color: #333;
3    background-image: -moz-linear-gradient(left, #555, #333); // FF 3.6
4    background-image: -ms-linear-gradient(left, #555, #333); // IE10
5    background-image: -webkit-gradient(linear, 0 0, 100% 0, from(#555),
6    background-image: -webkit-linear-gradient(left, #555, #333); // Saf
7    background-image: -o-linear-gradient(left, #555, #333); // Opera 11
8    background-image: linear-gradient(left, #555, #333); // Le standard
9    background-repeat: repeat-x;
10   filter: e(%("progid:DXImageTransform.Microsoft.gradient(startColors
11 }
```

Less

```
1 header nav {
2   .horizontal(#555, #333);
3 }
```

Mixins contd…

Mixins.less

```
1  .horizontal(@startColor: #555, @endColor: #333) {
2      background-color: @endColor;
3      background-image: -moz-linear-gradient(left, @startColor, @endCol
4      background-image: -ms-linear-gradient(left, @startColor, @endColo
5      background-image: -webkit-gradient(linear, 0 0, 100% 0, from(@sta
6      background-image: -webkit-linear-gradient(left, @startColor, @end
7      background-image: -o-linear-gradient(left, @startColor, @endColor
8      background-image: linear-gradient(left, @startColor, @endColor);
9      background-repeat: repeat-x;
10     filter: e(%("progid:DXImageTransform.Microsoft.gradient(startColo
11   }
12
```

Clever @importing is the key

Bootstrap.less

```
1  // CSS Reset
2  @import "reset.less";
3
4  // Core variables and mixins
5  @import "variables.less"; // Modify this for custom colors, font-size
6  @import "mixins.less";
7
8  // Grid system and page structure
9  @import "scaffolding.less";
10 @import "grid.less";
11 @import "layouts.less";
12 @import "media-grid.less";
13
14 // Base CSS
15 @import "type.less";
16 @import "code.less";
17 @import "forms.less";
18 @import "tables.less";
19
20 // Components: common
21 @import "sprites.less";
22 @import "dropdowns.less";
23 @import "wells.less";
24 @import "component-animations.less";
25 @import "close.less";
26
27 ...
```

Bootstrap's responsive.less

```
1  // REPEAT VARIABLES + MIXINS
2  // -------------------------
3  // Required since we compile the responsive stuff separately
4  @import "variables.less"; // Modify this for custom colors, font-size
5  @import "mixins.less";
6
7  // RESPONSIVE CLASSES
8  @import "responsive-utilities.less";
9
10 // MEDIA QUERIES
11
12 // UP TO LANDSCAPE PHONE
13 @import "responsive-480px-max.less";
14
15 // Phones to portrait tablets and narrow desktops
16 @import "responsive-767px-max.less";
17
18 // Tablets to regular desktops
19 @import "responsive-768px-979px.less";
20
21 // Landscape tablets to large desktops
22 @import "responsive-980px-1199px.less";
23
24 // Large desktops
25 @import "responsive-1200px-min.less";
```

Modular CSS = DRY Code

Re-using CSS has always between projects has always been a challenge, tips and tricks from one project get lost and re-written for another. CSS pre-processors can change that

Modular CSS = DRY Code

Re-using CSS has always between projects has always been a challenge, tips and tricks from one project get lost and re-written for another. CSS pre-processors can change that

- Want a responsive grid and a bunch of gradients? grid.less and mixins.less from Twitter Bootstrap and you're done. For Sass? Try Mooxins

Modular CSS = DRY Code

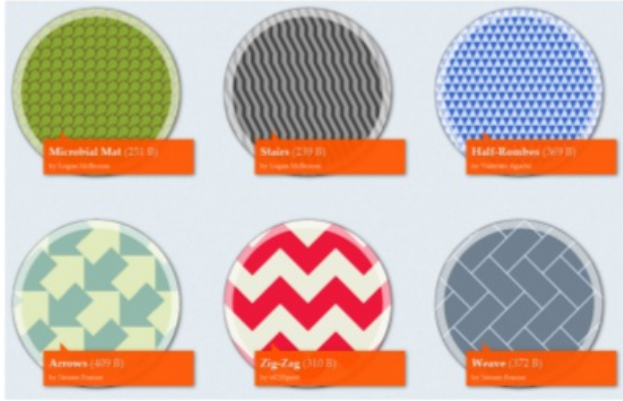
Re-using CSS has always between projects has always been a challenge, tips and tricks from one project get lost and re-written for another. CSS pre-processors can change that

- Want a responsive grid and a bunch of gradients? grid.less and mixins.less from Twitter Bootstrap and you're done. For Sass? Try Mooxins
- Build yourself a pattern library like ClearLess by ClearLeft

Modular CSS = DRY Code

Re-using CSS has always between projects has always been a challenge, tips and tricks from one project get lost and re-written for another. CSS pre-processors can change that

- Want a responsive grid and a bunch of gradients? grid.less and mixins.less from Twitter Bootstrap and you're done. For Sass? Try Mooxins
- Build yourself a pattern library like ClearLess by ClearLeft
- How about all the gradients from Lea Verou's CSS3 pattern gallery? Download css-patterns-less



Going deeper

Extensible classes

Less

```
.module-a {
  /* A bunch of stuff */
}
.module-b {
  /* Copies everything from .mo
   .module-a();
   border: 1px solid red;
}
```

```
.module-a {
  /* A bunch of stuff */
}
.module-b {
  /* A bunch of stuff */
   border: 1px solid red;
}
```

Sass

```
.module-a {
   /* A bunch of stuff */
}
.module-b {
   /* extends selectors, not co
   @extend .module-a;
   border: 1px solid red;
}
```

```
.module-a, .module-b {
   /* A bunch of stuff */
}
.module-b {
   border: 1px solid red;
}
```

Variable scoping

Less

```
@color: black;
.scoped {
  @color: white;
  color: @color;
}
.unscoped {
  // Black, as it should be
  color: @color;
}
```

Variable scoping

Less

```
1 @color: black;
2 .scoped {
3   @color: white;
4   color: @color;
5 }
6 .unscoped {
7   // Black, as it should be
8   color: @color;
9 }
```

Sass

```
1 $color: black;
2 .scoped {
3   $color: white;
4   color: $color;
5 }
6 .unscoped {
7    // White, as it was overwrit
8   color: $color;
9 }
```

Now for the funkier stuff

Loops & Logic

Sass

```
1  /* Sample Sass "if" statement *
2  @if lightness($color) &gt; 30%
3    background-color: #000;
4  } @else {
5    background-color: #fff;
6  }
7
8  /* Sample Sass "for" loop */
9  @for $i from 1px to 10px {
10   .border-#{$i} {
11     border: $i solid blue;
12   }
13 }
```

Loops & Logic

Sass

```
/* Sample Sass "if" statement *
@if lightness($color) &gt; 30%
  background-color: #000;
} @else {
  background-color: #fff;
}

/* Sample Sass "for" loop */
@for $i from 1px to 10px {
  .border-#{$i} {
    border: $i solid blue;
  }
}
```

```
/* Sample Sass "for" loop */
.border-1px {
border: 1px solid blue; }

.border-2px {
border: 2px solid blue; }

.border-3px {
border: 3px solid blue; }

.border-4px {
border: 4px solid blue; }

.border-5px {
border: 5px solid blue; }

.border-6px {
border: 6px solid blue; }

.border-7px {
border: 7px solid blue; }

.border-8px {
border: 8px solid blue; }

.border-9px {
```

Generating Rules

Less

```
// Do this function while @index &gt; 0
.indentX (@index) when (@index &gt; 0) {
  // Generate .indent-@index selector
  (~".indent-@{index}") { .indent(@index); }
  .indentX(@index - 1);
}
// Do this function when @index == 0
.indentX (0) {}

// Mixin to fill the indent with
.indent (@columns) {
  @colsminusone: @columns - 1;
  text-indent: @columns * 20px;
  background-color: darken(#f5f5f5, @colsminusone*6%);
}
// Run the function
.indentX (4);
```

CSS

```
.indent-4 {
  text-indent: 80px;
  background-color: #c7c7c7;
}
.indent-3 {
```

And then Less gets left behind

@content

A mixin include can now accept a **block of content**. The style block will be passed to the mixin and can be placed at the point @content is used

Sass

```
@mixin iphone {
  @media (max-width: 480px) {
    @content;
  }
}

@include iphone {
  body { color: red }
}
```

CSS

```
@media (max-width: 480px) {
  body { color: red }
}

```

Directive Interpolation

You can now output values of variables into standard CSS directives such a @font-face and @media

This allows for complicated mixins in a much simpler syntax

```
$media: screen;
$feature: -webkit-min-device-pixel-ratio;
$value: 1.5;

@media #{$media} and ($feature: $value) {
  ...
}

```

Even more features in Sass 3.2

Even more features in Sass 3.2

- Mixins can be defined in a nested context i.e. within @media queries

Even more features in Sass 3.2

- Mixins can be defined in a nested context i.e. within @media queries
- CSS Selectors Level 4 are now supported

Even more features in Sass 3.2

- Mixins can be defined in a nested context i.e. within @media queries
- CSS Selectors Level 4 are now supported
- min() and max() math functions

Even more features in Sass 3.2

- Mixins can be defined in a nested context i.e. within @media queries
- CSS Selectors Level 4 are now supported
- min() and max() math functions
- Placeholder selectors: an extension to @extend and mixins

Compass

Compass

- All of those basic Less mixins rolled into one

Compass

- All of those basic Less mixins rolled into one
- Sprites, vendor prefixes, grids, gradients, it's all there

Compass

- All of those basic Less mixins rolled into one
- Sprites, vendor prefixes, grids, gradients, it's all there
- Also things not possible with less: extra math functions like log(), sqrt(), pow(), plus tint() and shade()

Wait! It's not over

Wait! It's not over

Less 1.3.1 came out 3 days ago

A big (little) update

A big (little) update

- Much improved rule engine: Bootstrap compiled went from 6221 lines to 5270. **~18% improvement**

A big (little) update

- Much improved rule engine: Bootstrap compiled went from 6221 lines to 5270. **~18% improvement**
- New colour functions: tint, shade, multiply, screen, overlay, hardlight, difference, exclusion, average, negation, softlight, red, green, blue, contrast

A big (little) update

- Much improved rule engine: Bootstrap compiled went from 6221 lines to 5270. **~18% improvement**
- New colour functions: tint, shade, multiply, screen, overlay, hardlight, difference, exclusion, average, negation, softlight, red, green, blue, contrast
- Directive Interpolation, just like Sass 3.2

A big (little) update

- Much improved rule engine: Bootstrap compiled went from 6221 lines to 5270. **~18% improvement**
- New colour functions: tint, shade, multiply, screen, overlay, hardlight, difference, exclusion, average, negation, softlight, red, green, blue, contrast
- Directive Interpolation, just like Sass 3.2
- Supports new value units dpi, vmin, vm, dppx and dpcm

A big (little) update

- Much improved rule engine: Bootstrap compiled went from 6221 lines to 5270. **~18% improvement**
- New colour functions: tint, shade, multiply, screen, overlay, hardlight, difference, exclusion, average, negation, softlight, red, green, blue, contrast
- Directive Interpolation, just like Sass 3.2
- Supports new value units dpi, vmin, vm, dppx and dpcm
- Still doesn't resolve the extensible class issue

Is one better than the other?

Is one better than the other?

Yes, but don't think like that

Is one better than the other?

Sass

Less

CSS

When should I use it?

When should I use it?

All the time

Get Started

Command line

Less

```
npm install -g less
lessc style.less
style.css
```

Sass

```
gem install sass
gem install compass
sass --watch
style.scss:style.css
```

Get Started

Command line

Less

```
npm install -g less
lessc style.less
style.css
```

Sass

```
gem install sass
gem install compass
sass --watch
style.scss:style.css
```

.NET

dotless

Direct port of Less to .NET

```
PM> Install-Package
dotless
```

Bundle Transformer: Sass and SCSS

Translator adaptor for the basic Ruby library

```
PM> Install-Package
BundleTransformer.SassAnd
Scss
```

Both allow for .less/.scss files to be read at runtime and bundled & minified for production

Get Started contd...

PHP

Less

lessphp - a compiler for LESS written in PHP

```
require "lessc.inc.php";
$less = new lessc;
echo $less-
>compile(".block {
padding: 3 + 4px }");
```

Sass

PHPSass - a compiler for Sass written in PHP, a fork of the older PHamlP

```
require_once('SassParser.
php');
$parser = new
SassParser($settings);
$result = $parser-
>toCss($input);
```

Get Started contd...

Wordpress

Theme designers can write their CSS in Less or Sass and make use of in-theme bootstrappers so no plugins are required

Less

Use WP-LESS

```
// wp-content/themes/your-theme/functions.php
require dirname(__FILE__) . '/vendor/wp-less/bootstrap-
for-theme.php';
$less = WPLessPlugin::getInstance();
$less->dispatch();
```

Sass

Use WP-SASS

```
require_once( 'wp-sass/wp-sass.php' );
// enqueue a .scss style sheet
wp_enqueue_style( 'style',
get_stylesheet_directory_uri() . '/style.scss' );
```

Get Started Contd...

Javascript

Less is written in JavaScript, so you can run it client-side

```
<link rel="stylesheet/less" href="main.less"
type="text/css">
<script src="less.js" type="text/javascript"></script>
```

Get Started Contd...

Javascript

Less is written in JavaScript, so you can run it client-side

```
<link rel="stylesheet/less" href="main.less"
type="text/css">
<script src="less.js" type="text/javascript"></script>
```

> *So while it's a little more setup to get started, we (the sass core team) think that server side compilation is the best long term approach. Similarly, the less developers prefer server side compilation for production stylesheets.*
> *- Chris Eppstein on StackOverflow*

Get Started - Tools

Grunt

Yeoman

Code Kit

Find out more

Lesscss.org

sass-lang.com

Stylus
learnboost.github.com/stylus - A third way by @tjholowaychuk

smacss.com - A great e-book on modular CSS by Jonathan Snook

Thank you

Questions?

PA are hiring:

www.paconsulting.com/careers/