

**UX LAB**

# **CSS FOR DESIGNERS**

**PART TWO**

Presented by: Jaret Stezelberger + Matthew Tobiasz

August 31<sup>th</sup>, 2015

# REVIEW FROM LAST WEEK

- HTML
- CSS
- Browser Inspector
- Online Courses

# UX-LAB HOMEWORK

---

Let's review!

# MUST LEARN MORE

---

SASS

Sass

(Syntactically Awesome StyleSheets)

# WHAT IS A PREPROCESSOR?

A preprocessor is a program that takes one type of data and converts it to another type of data.



# WHAT IS SASS?

Sass is an extension of CSS that adds power and elegance to the basic language.

- Variables
- Nested rules
- Mixins
- Inheritance
- Inline imports
- Operators

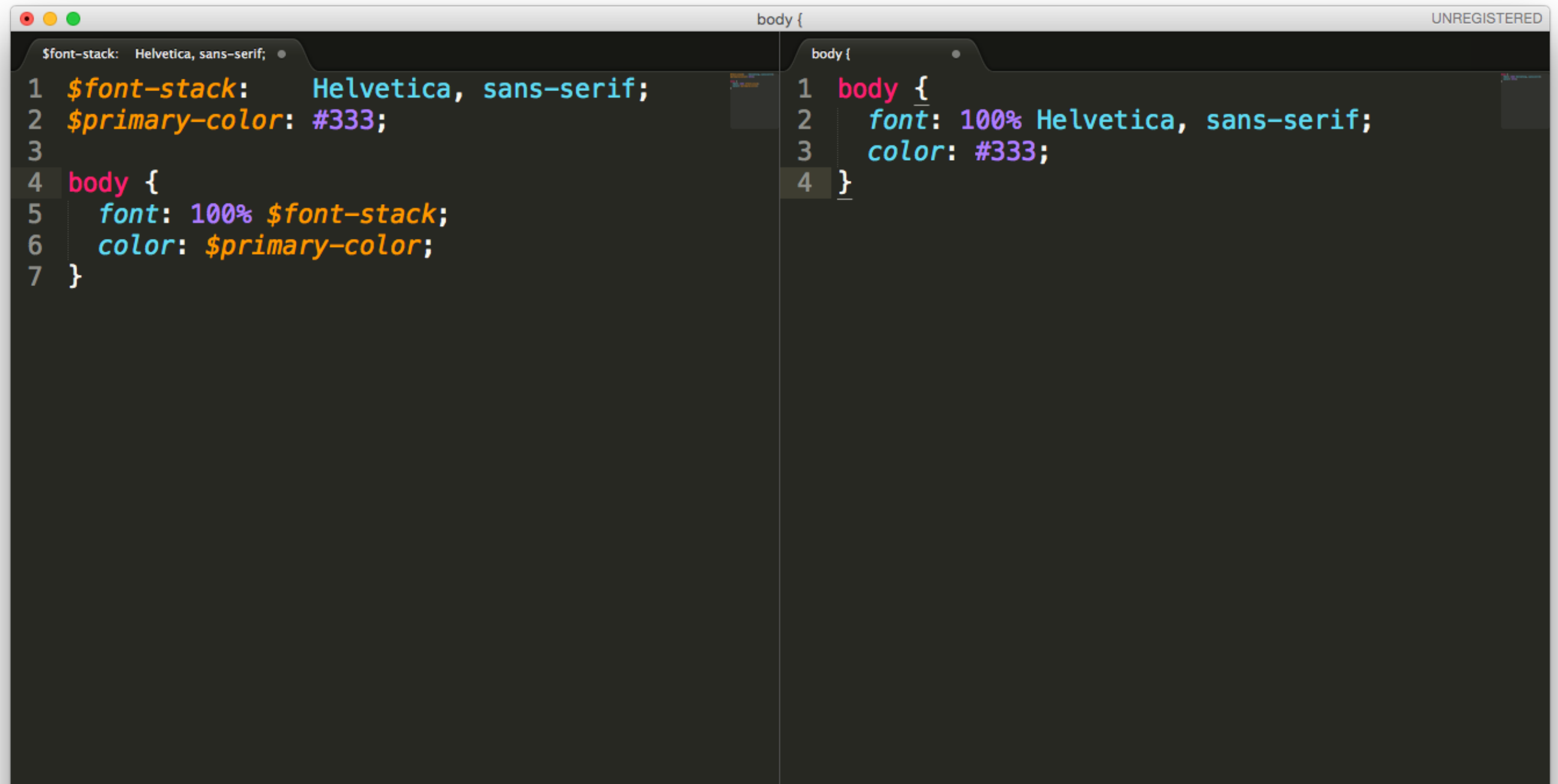
More on these here:

<http://sass-lang.com/guide>

# VARIABLES

Variables store information that you want to reuse throughout your stylesheet. Things like colors, font stacks, or any CSS value.

Below, the variables defined for the `$font-stack` and `$primary-color` are output as CSS with the variable values placed in the SCSS



The image shows a code editor with two panels. The left panel, titled '\$font-stack: Helvetica, sans-serif;', contains SCSS code. The right panel, titled 'body {', contains the resulting CSS code. The SCSS code defines two variables and uses them in a body selector. The CSS code shows the variables resolved to their actual values.

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

```
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```



# NESTED RULES

Nest your CSS selectors in a way that follows the same visual hierarchy of your HTML. Don't over nest your selectors. This is bad practice & hard to maintain.

- Notice the **ul**, **li**, and **a** selectors are nested inside the **nav** selector.
- This organizes your CSS and makes it more readable.



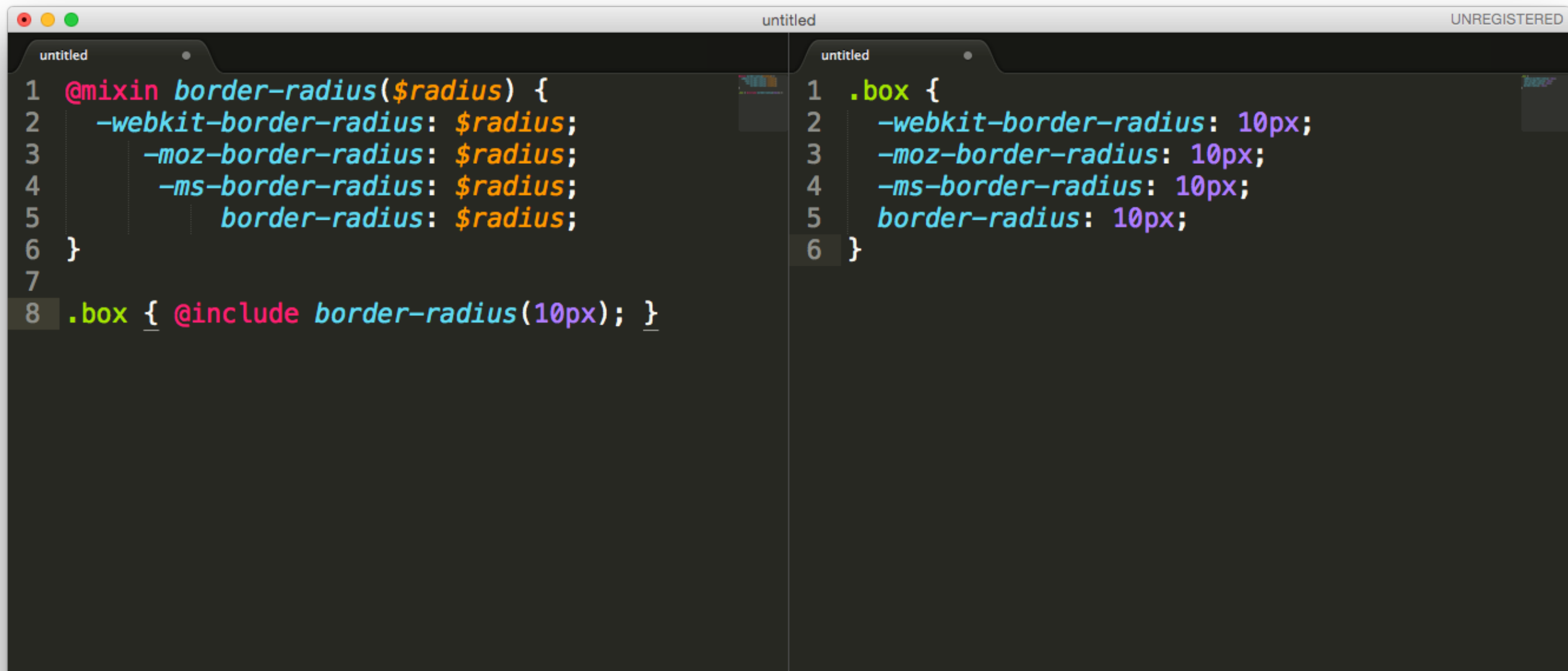
```
1 nav {  
2   ul {  
3     margin: 0;  
4     padding: 0;  
5     list-style: none;  
6   }  
7  
8   li { display: inline-block; }  
9  
10  a {  
11    display: block;  
12    padding: 6px 12px;  
13    text-decoration: none;  
14  }  
15 }
```

```
1 nav ul {  
2   margin: 0;  
3   padding: 0;  
4   list-style: none;  
5 }  
6  
7 nav li {  
8   display: inline-block;  
9 }  
10  
11 nav a {  
12   display: block;  
13   padding: 6px 12px;  
14   text-decoration: none;  
15 }
```

# MIXINS

Mixins allow you to define styles that can be re-used throughout the stylesheet .

- Create a mixin by using the `@mixin` directive and giving it a name.
- We've named our mixin: `border-radius`.
- Using the variable `$radius` , we pass in a radius of whatever we want.
- Use a mixin as a CSS declaration starting with `@include` followed by the mixin name

A screenshot of a code editor with two side-by-side windows, both titled 'untitled'. The editor has a dark background with syntax-highlighted CSS code. The left window shows the definition of a mixin named 'border-radius' using the '@mixin' directive, which takes a parameter '\$radius'. It lists four vendor-prefixed border-radius properties and the standard 'border-radius' property, all set to '\$radius'. The right window shows the usage of this mixin within a class selector '.box'. It lists the same four properties, but they are all set to '10px' instead of a variable. The '@include' directive is used to call the 'border-radius' mixin with '10px' as the argument.

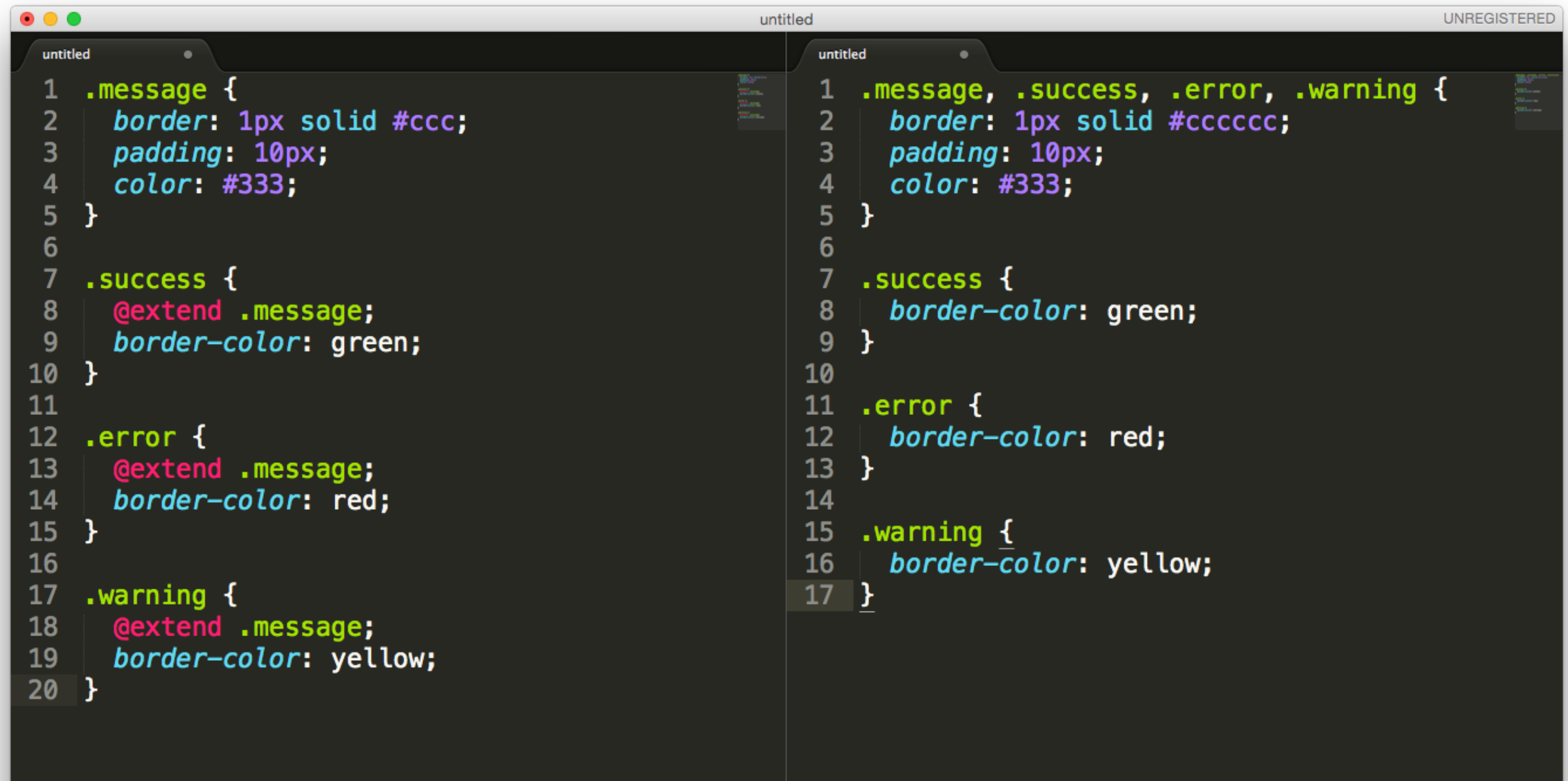
```
1 @mixin border-radius($radius) {  
2     -webkit-border-radius: $radius;  
3     -moz-border-radius: $radius;  
4     -ms-border-radius: $radius;  
5     border-radius: $radius;  
6 }  
7  
8 .box { @include border-radius(10px); }
```

```
1 .box {  
2     -webkit-border-radius: 10px;  
3     -moz-border-radius: 10px;  
4     -ms-border-radius: 10px;  
5     border-radius: 10px;  
6 }
```

# INHERITANCE

When one class should have all the styles of another class, as well as its own specific styles.

- Using `@extend` takes the CSS properties in `.message` and applies them to `.success`, `.error`, & `.warning`.



```
untitled UNREGISTERED

1 .message {
2   border: 1px solid #ccc;
3   padding: 10px;
4   color: #333;
5 }
6
7 .success {
8   @extend .message;
9   border-color: green;
10 }
11
12 .error {
13   @extend .message;
14   border-color: red;
15 }
16
17 .warning {
18   @extend .message;
19   border-color: yellow;
20 }

untitled

1 .message, .success, .error, .warning {
2   border: 1px solid #cccccc;
3   padding: 10px;
4   color: #333;
5 }
6
7 .success {
8   border-color: green;
9 }
10
11 .error {
12   border-color: red;
13 }
14
15 .warning {
16   border-color: yellow;
17 }
```

# OPERATORS

Doing math in your CSS is very helpful.

Sass has a handful of standard math operators:

- **+** addition
- **-** subtraction
- **\*** multiplication
- **/** division
- **%** modulus operator computes the remainder after dividing its first operand by its second



```
untitled UNREGISTERED
```

```
1 .container { width: 100%; }
2
3 article[role="main"] {
4   float: left;
5   width: 600px / 960px * 100%;
6 }
7
8 aside[role="complimentary"] {
9   float: right;
10  width: 300px / 960px * 100%;
11 }
```

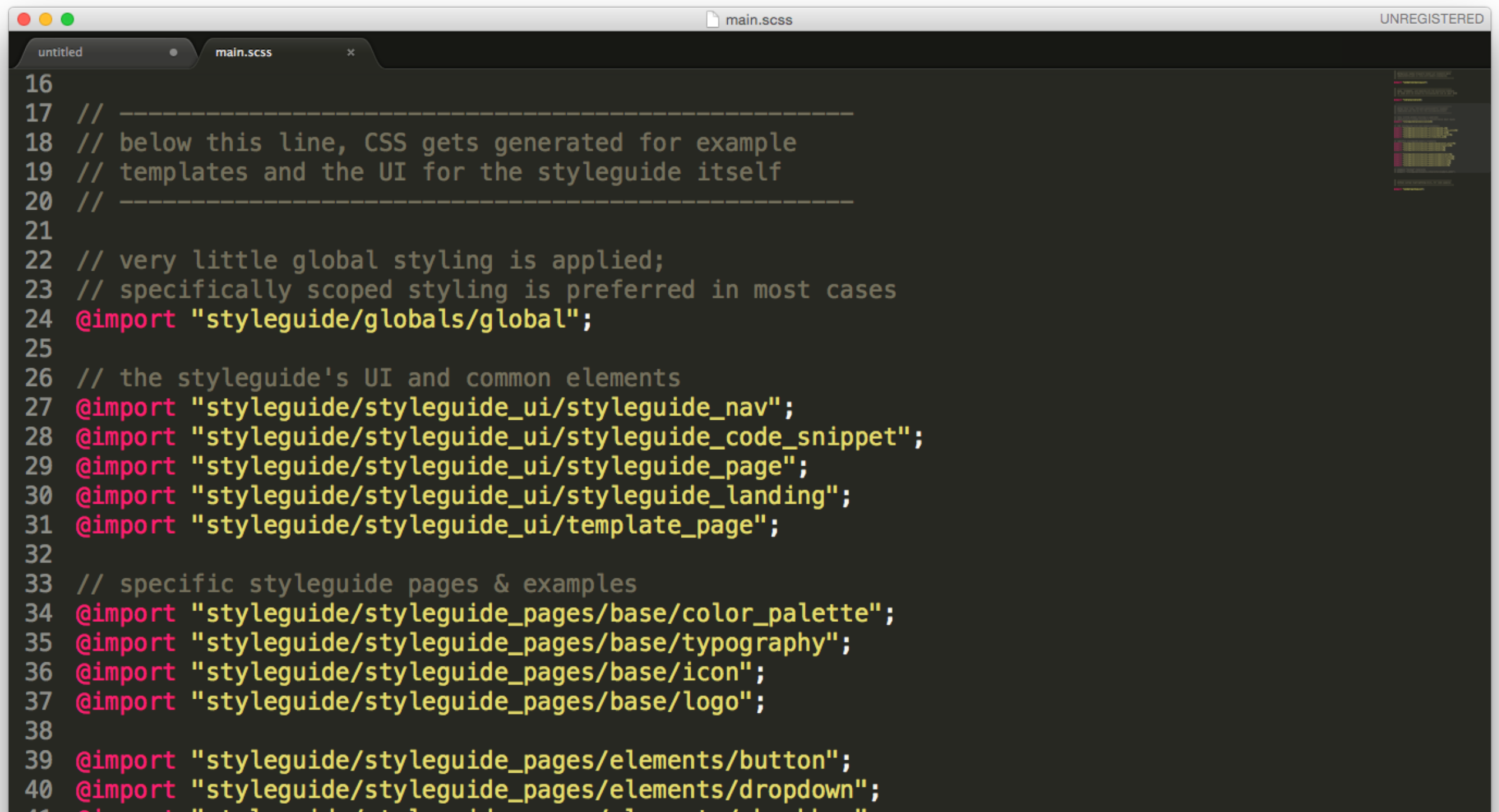
```
untitled
```

```
1 .container {
2   width: 100%;
3 }
4
5 article[role="main"] {
6   float: left;
7   width: 62.5%;
8 }
9
10 aside[role="complimentary"] {
11   float: right;
12   width: 31.25%;
13 }
```

# INLINE IMPORTS

**@import** option that lets you combine multiple CSS files into one

- Split your CSS into smaller maintainable files
- Generate a single CSS file to the web browser, creating fewer HTTP requests

A screenshot of a code editor window titled 'main.scss' with a tab labeled 'main.scss' and a close button. The editor shows SCSS code with line numbers 16 to 40. The code includes several @import statements for various styleguide components. The editor has a dark theme and a sidebar on the right showing a file tree.

```
16
17 // -----
18 // below this line, CSS gets generated for example
19 // templates and the UI for the styleguide itself
20 // -----
21
22 // very little global styling is applied;
23 // specifically scoped styling is preferred in most cases
24 @import "styleguide/globals/global";
25
26 // the styleguide's UI and common elements
27 @import "styleguide/styleguide_ui/styleguide_nav";
28 @import "styleguide/styleguide_ui/styleguide_code_snippet";
29 @import "styleguide/styleguide_ui/styleguide_page";
30 @import "styleguide/styleguide_ui/styleguide_landing";
31 @import "styleguide/styleguide_ui/template_page";
32
33 // specific styleguide pages & examples
34 @import "styleguide/styleguide_pages/base/color_palette";
35 @import "styleguide/styleguide_pages/base/typography";
36 @import "styleguide/styleguide_pages/base/icon";
37 @import "styleguide/styleguide_pages/base/logo";
38
39 @import "styleguide/styleguide_pages/elements/button";
40 @import "styleguide/styleguide_pages/elements/dropdown";
41 @import "styleguide/styleguide_pages/elements/form";
```

# SASS EXAMPLES

---

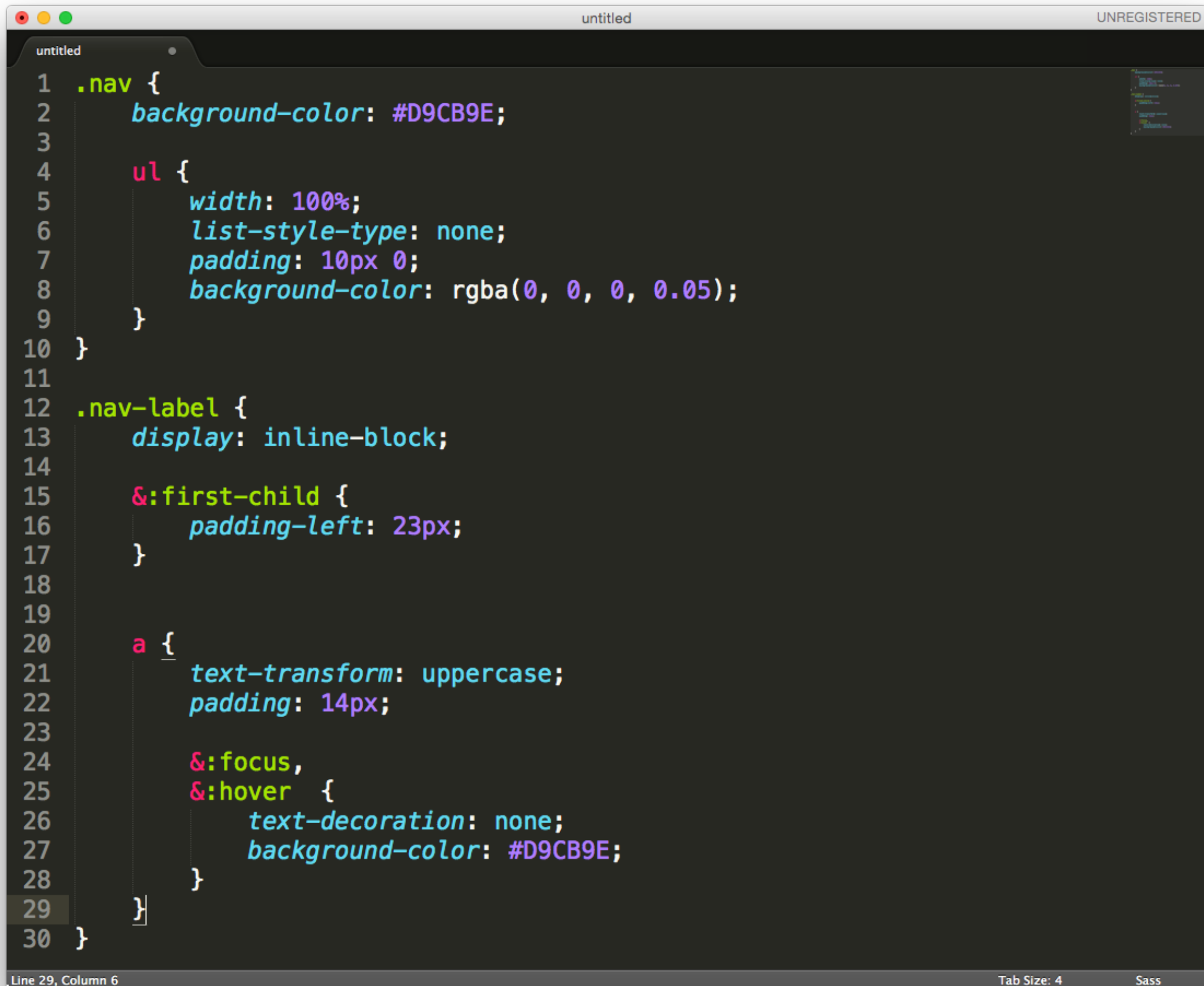
Nesting, Inheritance, Mixins

# NESTING EXAMPLE: BEFORE SASS

```
untitled UNREGISTERED
1  .nav {
2      text-align: left;
3      background-color: #D9CB9E;
4  }
5
6  .nav > ul {
7      width: 100%;
8      list-style-type: none;
9      margin: 0px;
10     padding: 10px 0px 10px 0px;
11     background-color: rgba(0, 0, 0, 0.05);
12 }
13 .nav-label {
14     display: inline-block;
15     text-transform: uppercase;
16 }
17 .nav-label > a {
18     padding: 10px 15px 10px 15px;
19 }
20 .nav-label:first-child {
21     padding-left: 23px;
22 }
23 .nav-label > a:focus, .nav-label > a:hover {
24     text-decoration: none;
25     background-color: #D9CB9E;
26 }
27
```



# NESTING EXAMPLE: AFTER SASS



The image shows a screenshot of a code editor window titled "untitled" with a status bar indicating "UNREGISTERED". The editor contains Sass code for a navigation menu. The code is as follows:

```
1  .nav {
2    background-color: #D9CB9E;
3
4    ul {
5      width: 100%;
6      list-style-type: none;
7      padding: 10px 0;
8      background-color: rgba(0, 0, 0, 0.05);
9    }
10 }
11
12 .nav-label {
13   display: inline-block;
14
15   &:first-child {
16     padding-left: 23px;
17   }
18
19   a {
20     text-transform: uppercase;
21     padding: 14px;
22
23     &:focus,
24     &:hover {
25       text-decoration: none;
26       background-color: #D9CB9E;
27     }
28   }
29 }
30 }
```

The status bar at the bottom indicates "Line 29, Column 6" and "Tab Size: 4". The word "Sass" is also visible in the bottom right corner.



# NESTING EXAMPLE: BEFORE & AFTER

```
1 .nav {
2   text-align: left;
3   background-color: #D9CB9E;
4 }
5
6 .nav > ul {
7   width: 100%;
8   list-style-type: none;
9   margin: 0px;
10  padding: 10px 0px 10px 0px;
11  background-color: rgba(0, 0, 0, 0.05);
12 }
13 .nav-label {
14   display: inline-block;
15   text-transform: uppercase;
16 }
17 .nav-label > a {
18   padding: 10px 15px 10px 15px;
19 }
20 .nav-label:first-child {
21   padding-left: 23px;
22 }
23 .nav-label > a:focus, .nav-label > a:hover
24   text-decoration: none;
25   background-color: #D9CB9E;
26 }
27
```

```
1 .nav {
2   background-color: #D9CB9E;
3
4   ul {
5     width: 100%;
6     list-style-type: none;
7     padding: 10px 0;
8     background-color: rgba(0, 0, 0, 0.05);
9   }
10 }
11
12 .nav-label {
13   display: inline-block;
14
15   &:first-child {
16     padding-left: 23px;
17   }
18
19   a {
20     text-decoration: underline;
21     text-transform: uppercase;
22     padding: 14px;
23
24     &:focus,
25     &:hover {
26       text-decoration: none;
27       background-color: #D9CB9E;
28     }
29   }
30 }
```

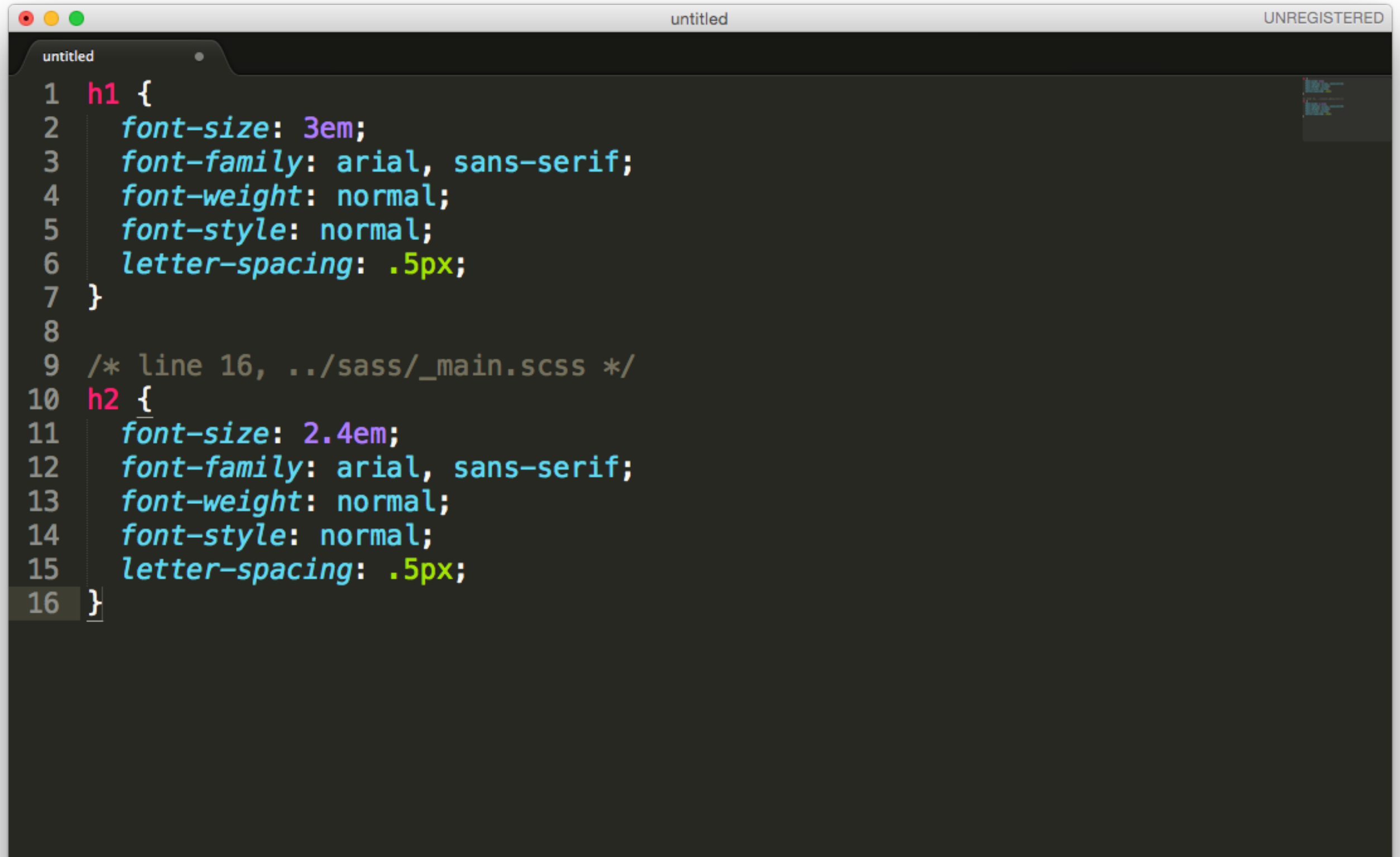
# MIXIN EXAMPLE

Mixins are included in the document with the `@include` directive.

A screenshot of a code editor window titled "untitled" with a status bar indicating "UNREGISTERED". The editor contains 16 lines of Sass code. Line 1: `$font-family: arial, sans-serif;`. Line 2: empty. Line 3: empty. Line 4: `@mixin heading($font-size) {`. Line 5: `font-size: $font-size;`. Line 6: `font-family: $font-family;`. Line 7: `font-weight: normal;`. Line 8: `font-style: normal;`. Line 9: `letter-spacing: .5px;`. Line 10: `}`. Line 11: empty. Line 12: `$h1-fontsize: 3em;`. Line 13: `$h2-fontsize: 2.4em;`. Line 14: empty. Line 15: `h1 { @include heading($h1-fontsize); }`. Line 16: `h2 { @include heading($h2-fontsize); }`. The code is color-coded: variables in orange, mixin directive in pink, and other keywords in blue and green. The line numbers 1 through 16 are visible on the left side of the editor. A small icon is visible in the top right corner of the editor area.

# MIXIN EXAMPLE

Compiled as normal CSS

A screenshot of a code editor window titled 'untitled' with a status bar indicating 'UNREGISTERED'. The editor contains SCSS code for two selectors, h1 and h2. The h1 selector is defined on lines 1-7, and the h2 selector is defined on lines 10-16. A comment on line 9 indicates that the h2 selector is a mixin from another file. The code is syntax-highlighted, with selectors in red, property names in blue, values in purple, and units in green. The h2 selector is underlined on line 10.

```
1  h1 {  
2    font-size: 3em;  
3    font-family: arial, sans-serif;  
4    font-weight: normal;  
5    font-style: normal;  
6    letter-spacing: .5px;  
7  }  
8  
9  /* line 16, ../sass/_main.scss */  
10 h2 {  
11   font-size: 2.4em;  
12   font-family: arial, sans-serif;  
13   font-weight: normal;  
14   font-style: normal;  
15   letter-spacing: .5px;  
16 }
```

# MIXIN EXAMPLE

Compiled as normal CSS

```
untitled
1  $font-family: arial, sans-serif;
2
3
4  @mixin heading($font-size) {
5      font-size: $font-size;
6      font-family: $font-family;
7      font-weight: normal;
8      font-style: normal;
9      letter-spacing: .5px;
10 }
11
12 $h1-fontsize: 3em;
13 $h2-fontsize: 2.4em;
14
15 h1 { @include heading($h1-fontsize); }
16 h2 { @include heading($h2-fontsize); }
```

```
untitled
1  h1 {
2      font-size: 3em;
3      font-family: arial, sans-serif;
4      font-weight: normal;
5      font-style: normal;
6      letter-spacing: .5px;
7  }
8
9  /* line 16, ../sass/_main.scss */
10 h2 {
11     font-size: 2.4em;
12     font-family: arial, sans-serif;
13     font-weight: normal;
14     font-style: normal;
15     letter-spacing: .5px;
16 }
```

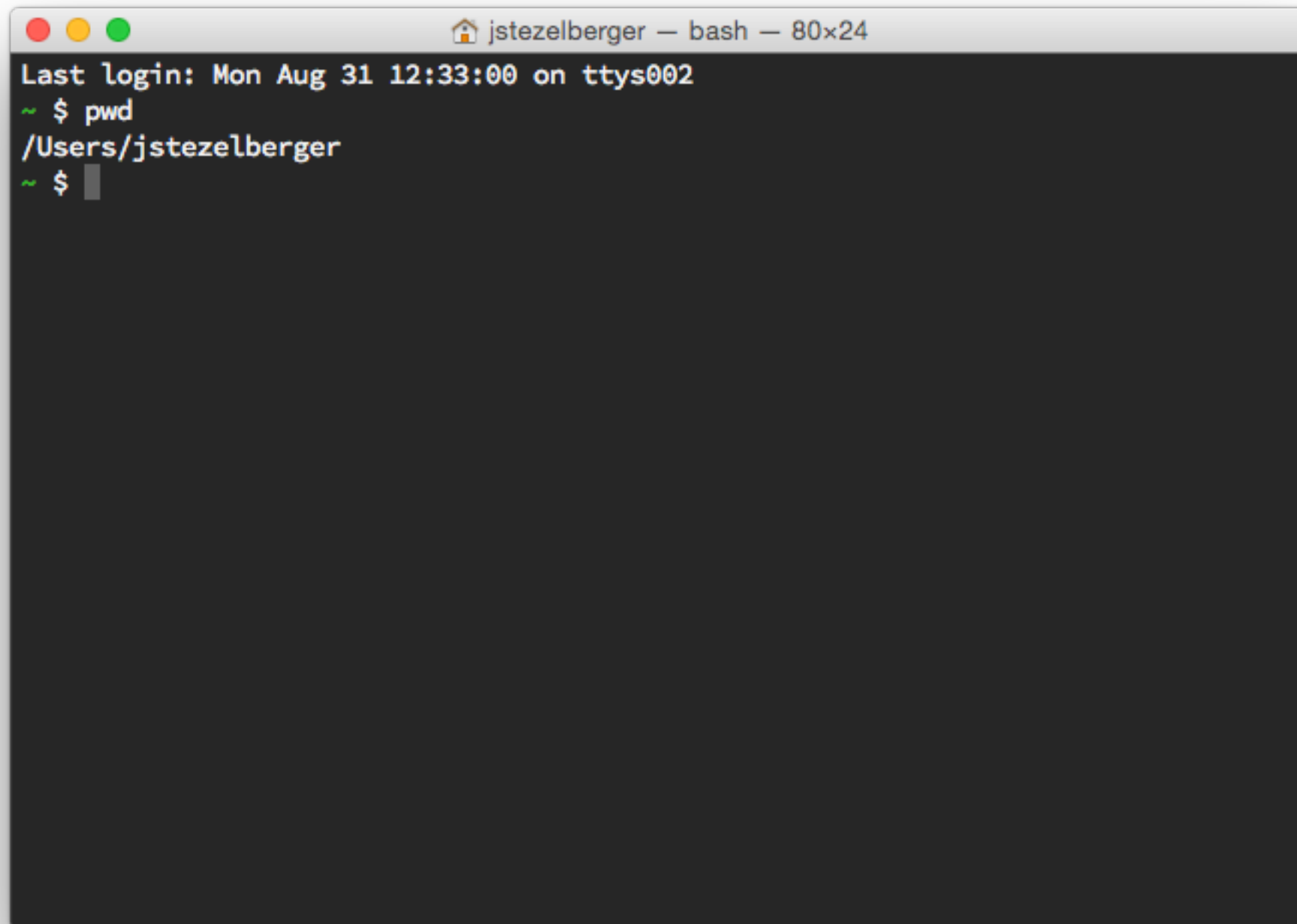
# GETTING STARTED

---

Using the Terminal, Installing Sass & the Compass Library

# TERMINAL

- <http://ss64.com/osx/>
- [https://github.com/0nn0/terminal-mac-cheatsheet/wiki/Terminal-Cheatsheet-for-Mac-\(-basics-\)](https://github.com/0nn0/terminal-mac-cheatsheet/wiki/Terminal-Cheatsheet-for-Mac-(-basics-))



```
jstezelberger — bash — 80x24
Last login: Mon Aug 31 12:33:00 on ttys002
~ $ pwd
/Users/jstezelberger
~ $
```

# INSTALLING SASS & COMPASS



<http://compass-style.org/>

# **HOMEWORK!**

---

Refactor your homework using Sass, Further customize the project.



# REFACTOR YOUR HOMEWORK USING SASS

- Create an example for each of the basics we learned earlier; Variables, Nested rules, Mixins, etc.
- Look for repetitive code and replace it with variables
- Pay attention to simplicity
- Create a Mixin for reusable elements

# FURTHER CUSTOMIZE THE PROJECT

- Change up the styling of header tags <h1>, <h2>, etc.
- Apply a different font and colour scheme to the site.
- UX Lab is missing a logo in the header, create a logo and add it to the site. (Hint: css background images).
- Rework the page design entirely, here's one idea on what it could look like: <http://www.initializr.com/try>

# **UX LAB**

## **PART THREE**

To Be Continued

**THANKS**

**FOR LISTENING**

# APPENDIX

# CSS NAMING CONVENTIONS

<http://thesassway.com/advanced/modular-css-naming-conventions>