Sass

(Syntastically Awesome StyleSheets)

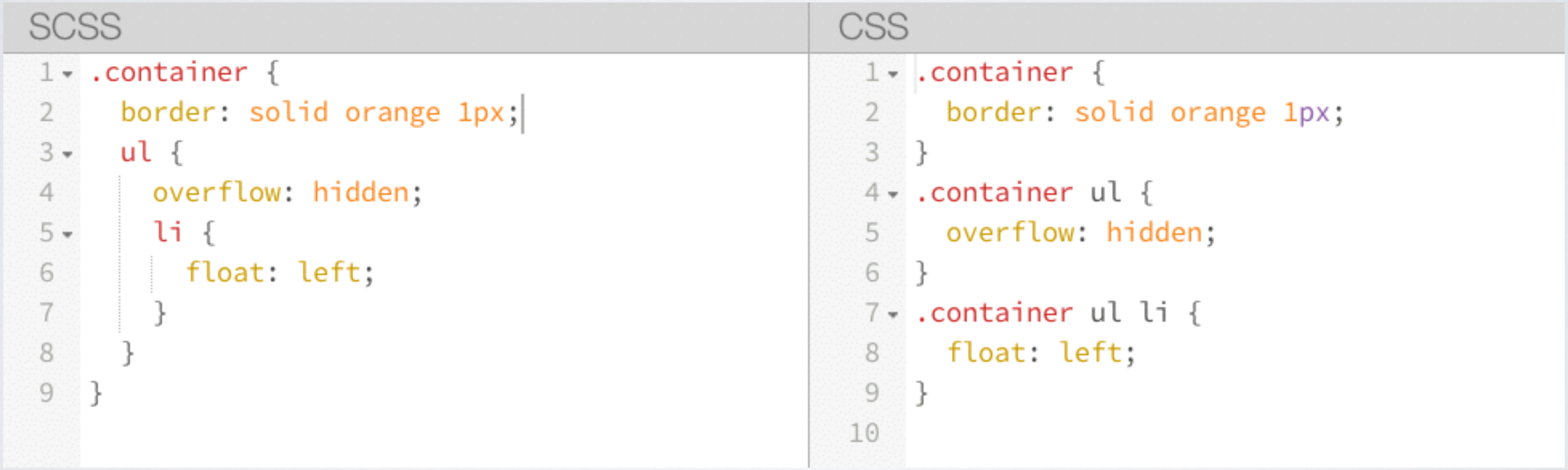
As interpreted by Eric Wood

WHAT IS IT?

- ~~~~~sassy~~~~~
- Pre-processor for CSS
- Legible! (if written correctly…)
- Modular and reusable
- Cool acronym

BASIC FEATURES

NESTING!

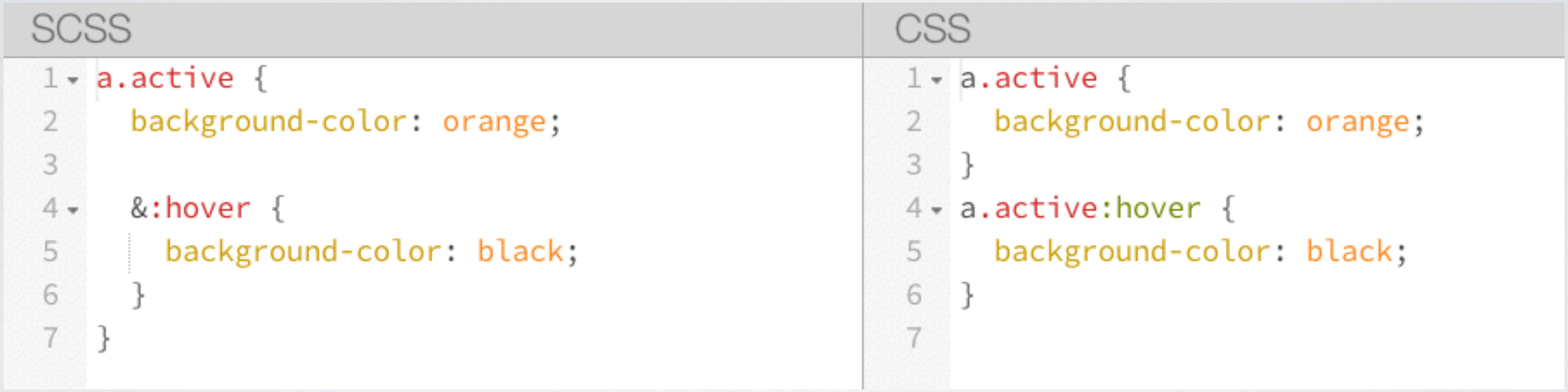


```scss
.container {
  border: solid orange 1px;
  ul {
    overflow: hidden;
    li {
      float: left;
    }
  }
}
```

```css
.container {
  border: solid orange 1px;
}
.container ul {
  overflow: hidden;
}
.container ul li {
  float: left;
}
```

AMPERSANDS!

Perfect for use with pseudo-selectors!



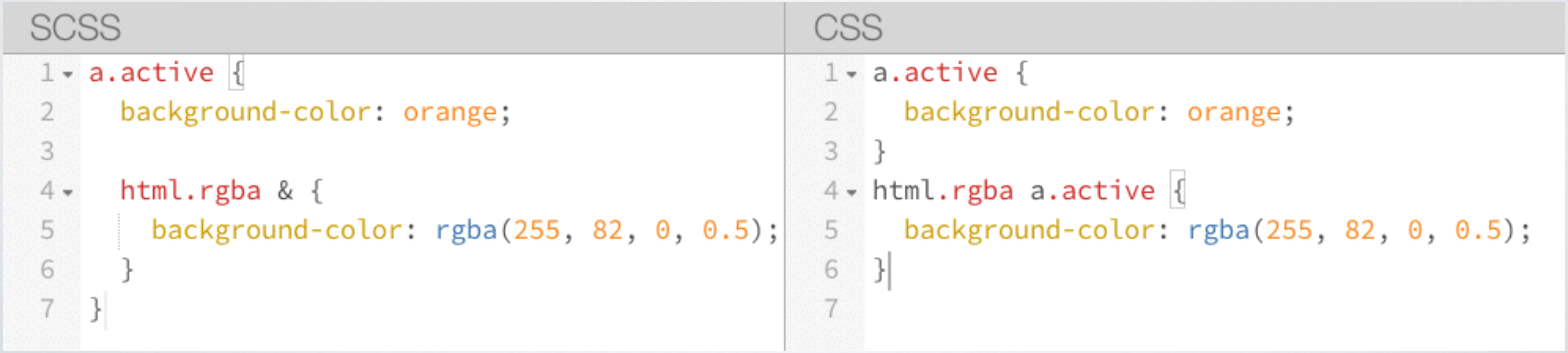
SCSS

```scss
a.active {
  background-color: orange;

  &:hover {
    background-color: black;
  }
}
```

CSS

```css
a.active {
  background-color: orange;
}
a.active:hover {
  background-color: black;
}
```

Really useful with Modernizr:



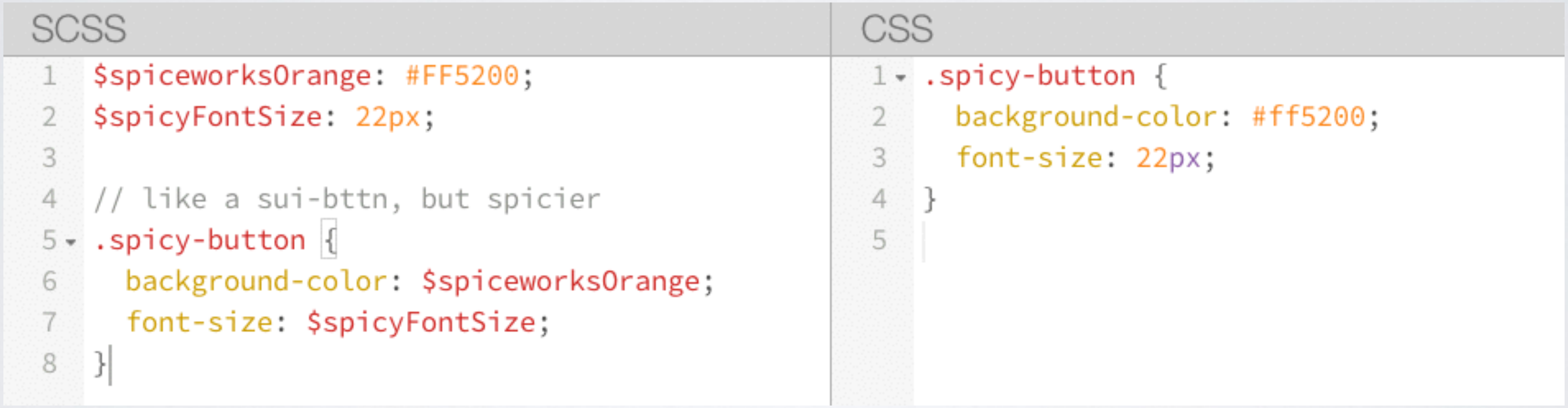
SCSS

```scss
a.active {
  background-color: orange;

  html.rgba & {
    background-color: rgba(255, 82, 0, 0.5);
  }
}
```

CSS

```css
a.active {
  background-color: orange;
}
html.rgba a.active {
  background-color: rgba(255, 82, 0, 0.5);
}
```

VARIABLES!



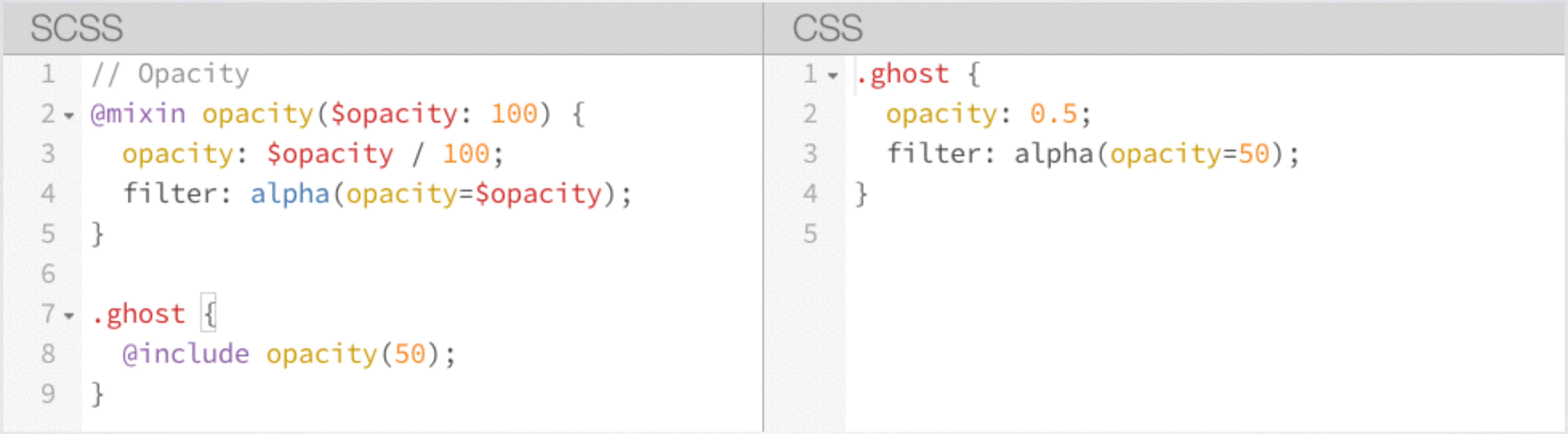
SCSS

```scss
$spiceworksOrange: #FF5200;
$spicyFontSize: 22px;

// like a sui-bttn, but spicier
.spicy-button {
  background-color: $spiceworksOrange;
  font-size: $spicyFontSize;
}
```

CSS

```css
.spicy-button {
  background-color: #ff5200;
  font-size: 22px;
}

```

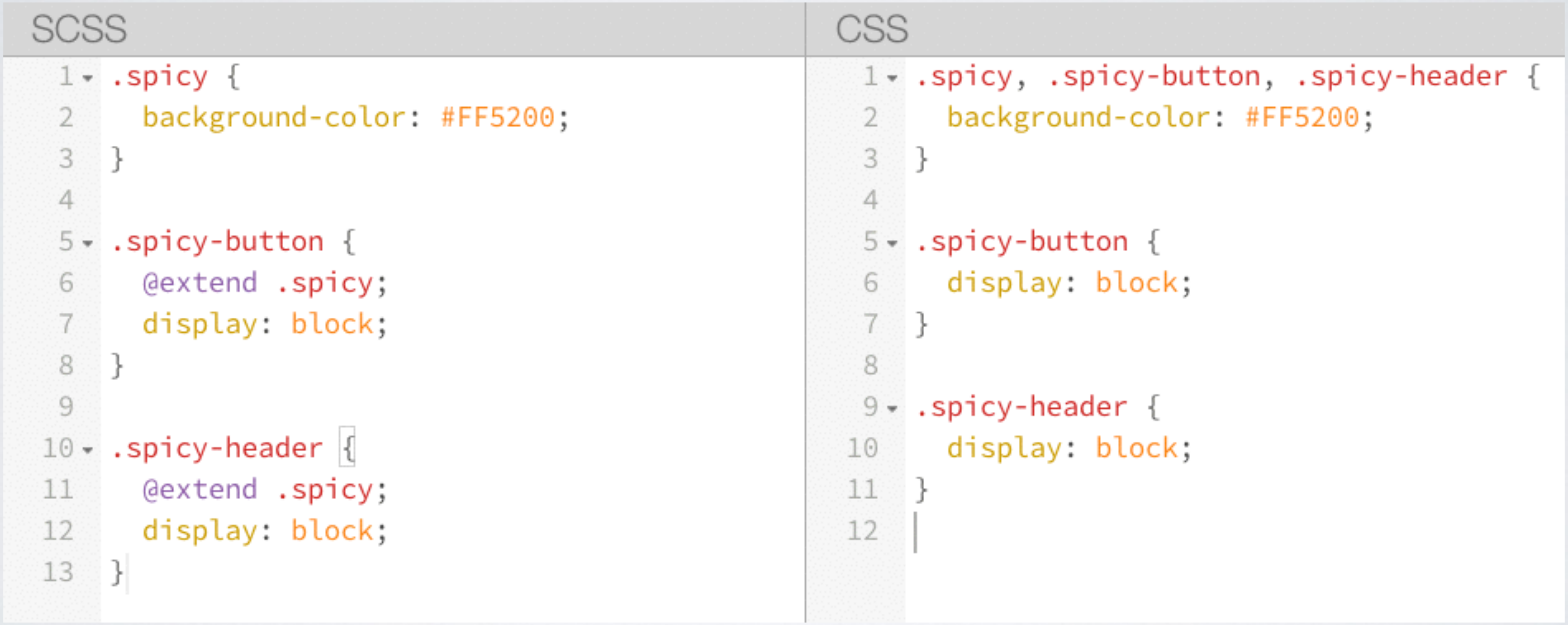
MIXINS!



```scss
// Opacity
@mixin opacity($opacity: 100) {
  opacity: $opacity / 100;
  filter: alpha(opacity=$opacity);
}

.ghost {
  @include opacity(50);
}
```

```css
.ghost {
  opacity: 0.5;
  filter: alpha(opacity=50);
}
```

EXTEND / INHERITANCE



SCSS

```scss
.spicy {
  background-color: #FF5200;
}

.spicy-button {
  @extend .spicy;
  display: block;
}

.spicy-header {
  @extend .spicy;
  display: block;
}
```

CSS

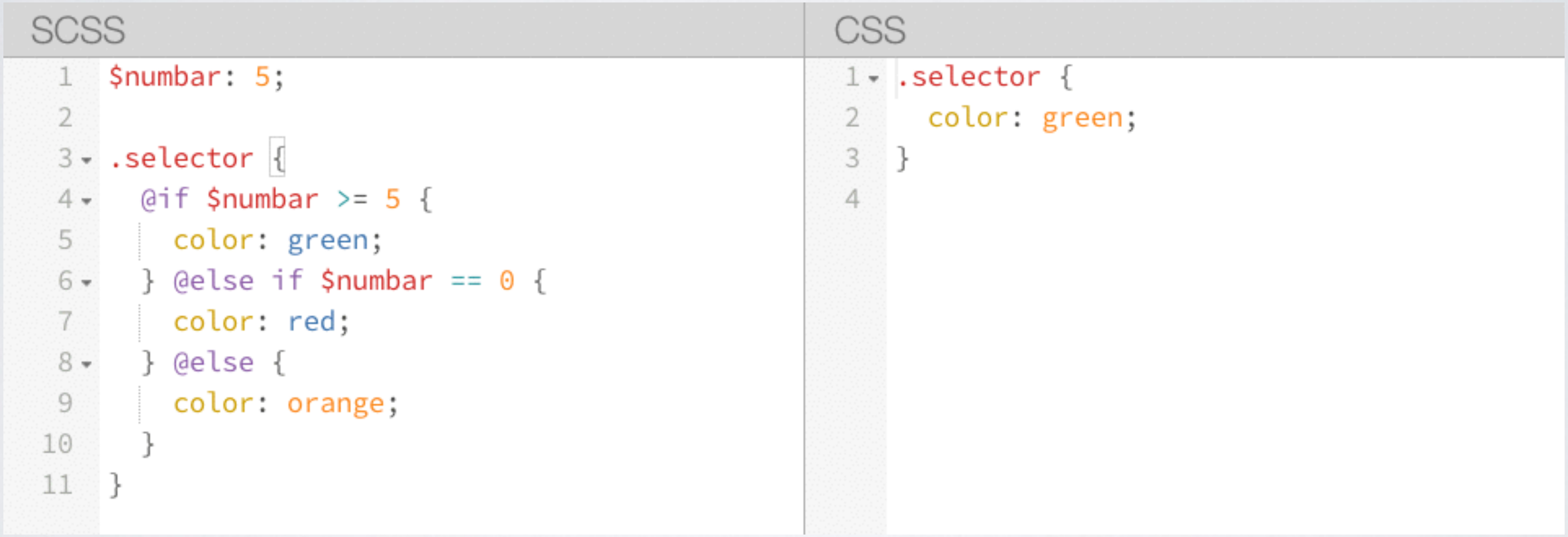
```css
.spicy, .spicy-button, .spicy-header {
  background-color: #FF5200;
}

.spicy-button {
  display: block;
}

.spicy-header {
  display: block;
}
```

MORE ADVANCED STUFF

MATH!

Vertically centering an element:
http://codepen.io/eric-wood/pen/zopAI

FLOW CONTROL!



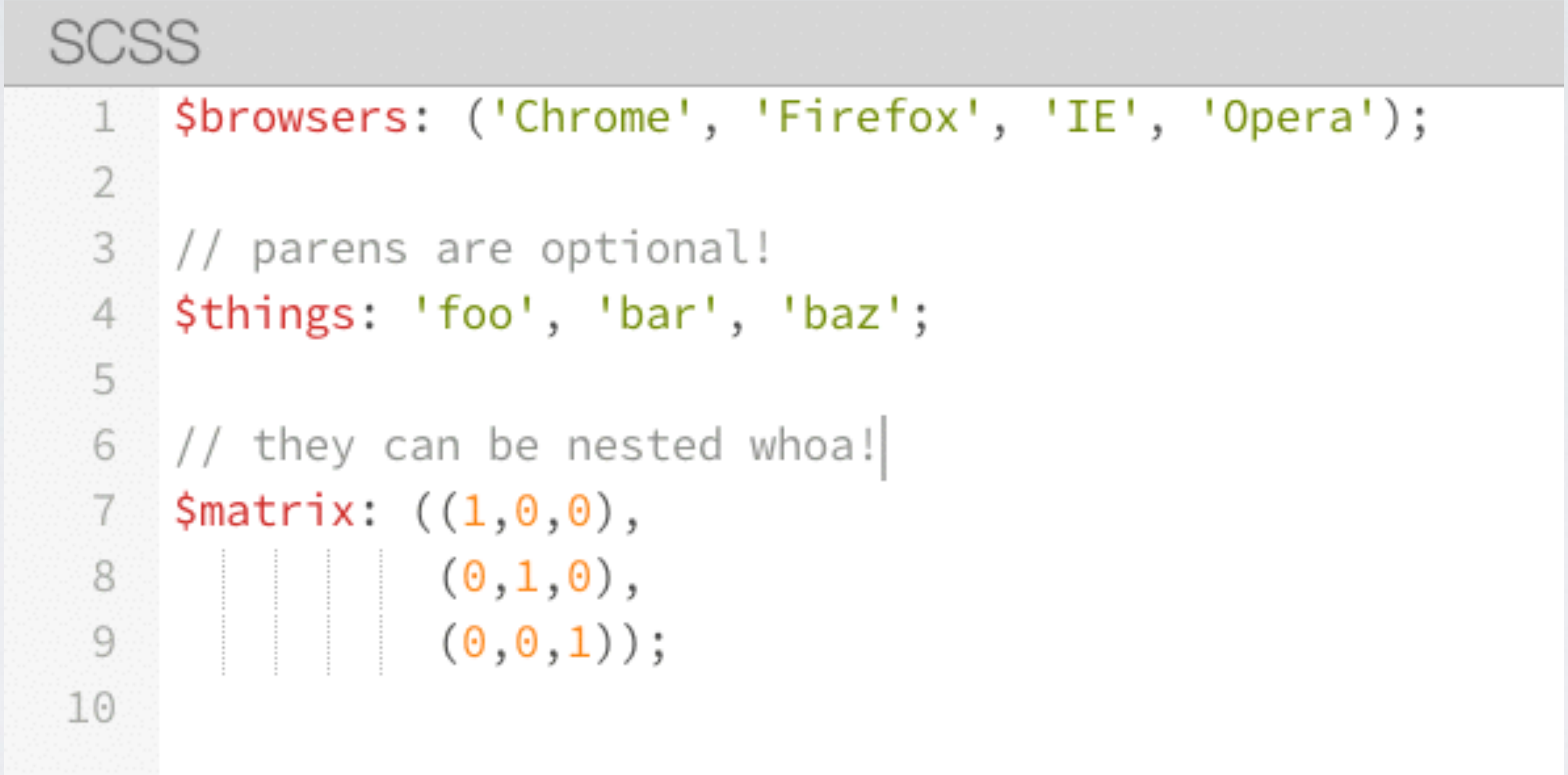
SCSS

```scss
$numbar: 5;

.selector {
  @if $numbar >= 5 {
    color: green;
  } @else if $numbar == 0 {
    color: red;
  } @else {
    color: orange;
  }
}
```

CSS

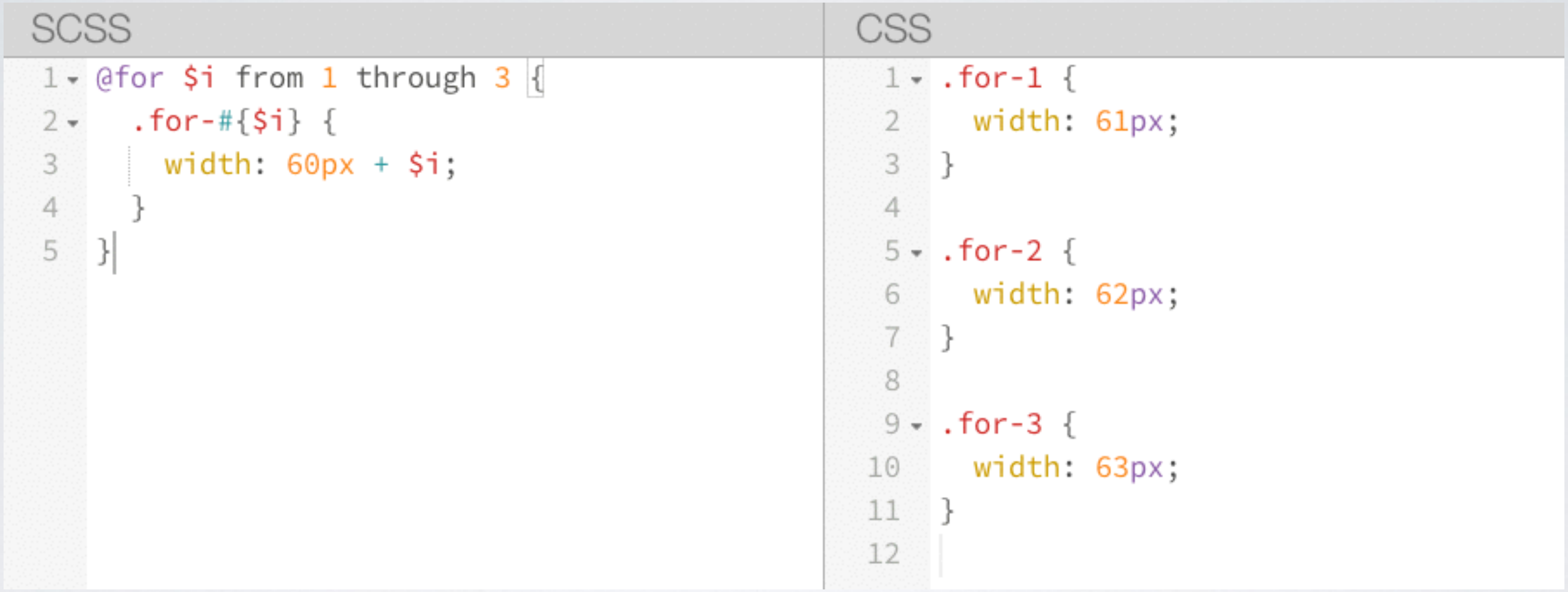
```css
.selector {
  color: green;
}

```

LISTS



```scss
$browsers: ('Chrome', 'Firefox', 'IE', 'Opera');

// parens are optional!
$things: 'foo', 'bar', 'baz';

// they can be nested whoa!
$matrix: ((1,0,0),
          (0,1,0),
          (0,0,1));

```

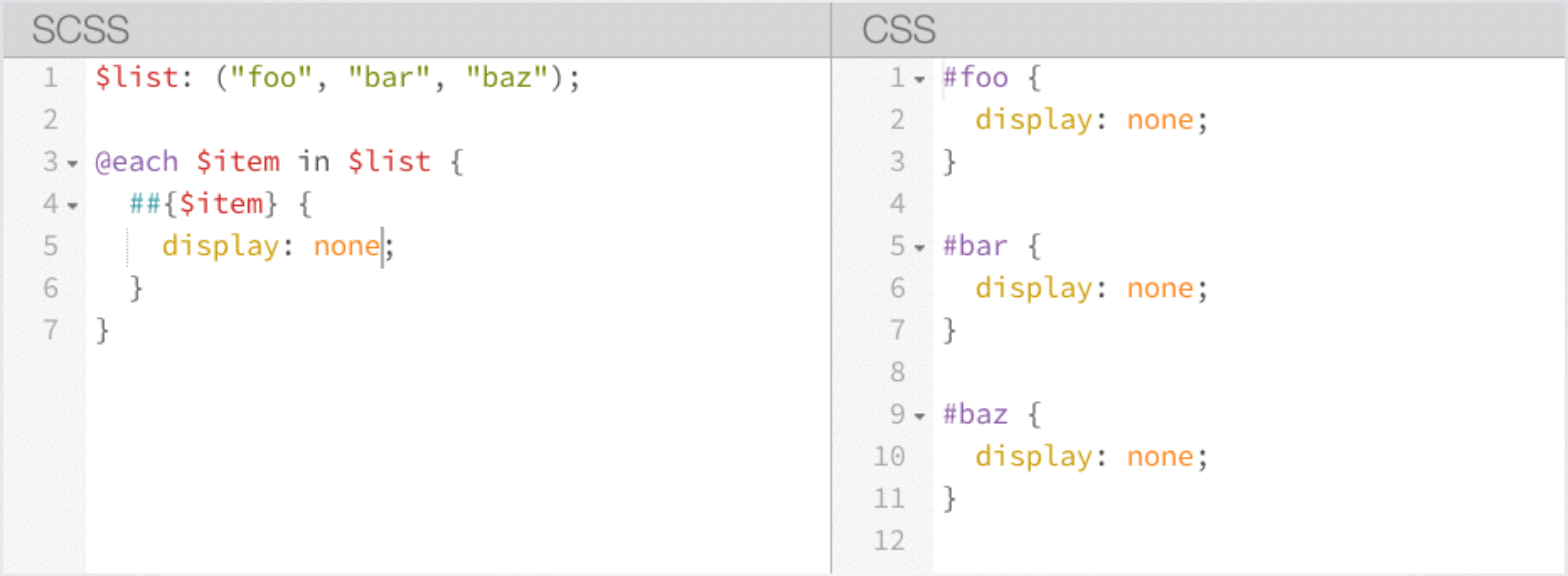
FOR LOOPS!



```scss
@for $i from 1 through 3 {
  .for-#{$i} {
    width: 60px + $i;
  }
}
```

```css
.for-1 {
  width: 61px;
}

.for-2 {
  width: 62px;
}

.for-3 {
  width: 63px;
}
```

EACH LOOPS!



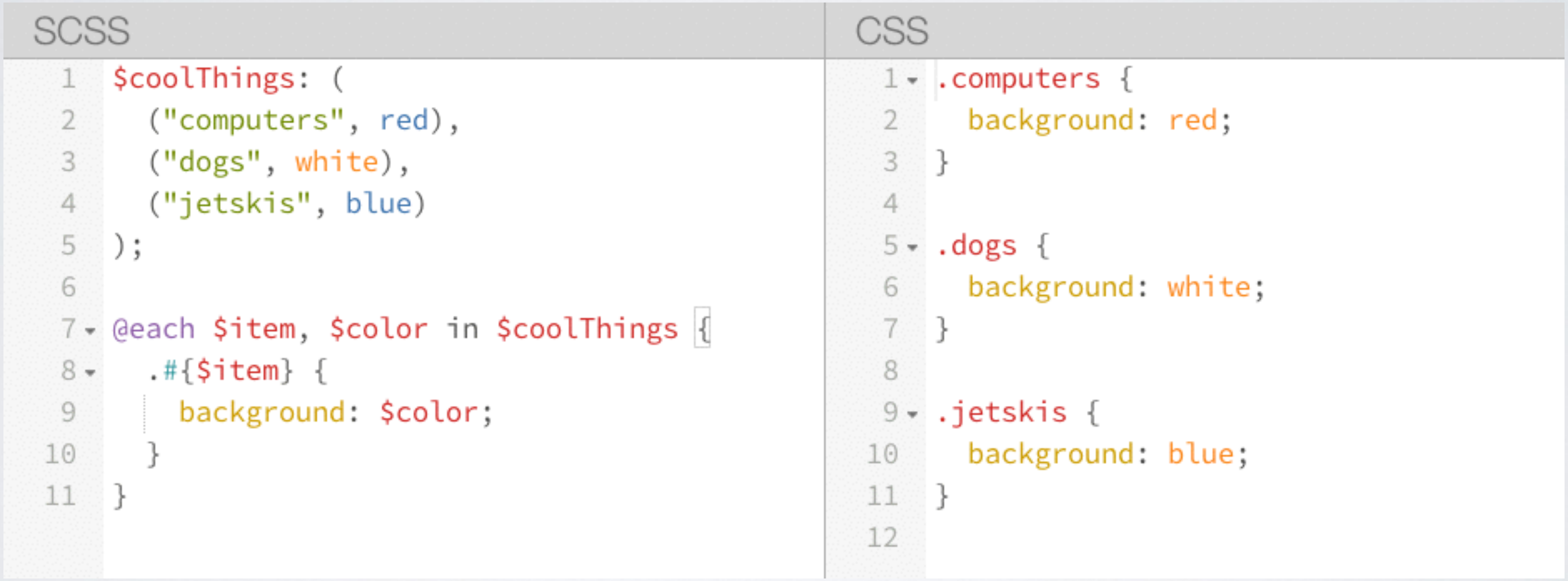
SCSS

```scss
$list: ("foo", "bar", "baz");

@each $item in $list {
  ##{$item} {
    display: none;
  }
}
```

CSS

```css
#foo {
  display: none;
}

#bar {
  display: none;
}

#baz {
  display: none;
}

```

MORE EACH LOOPS

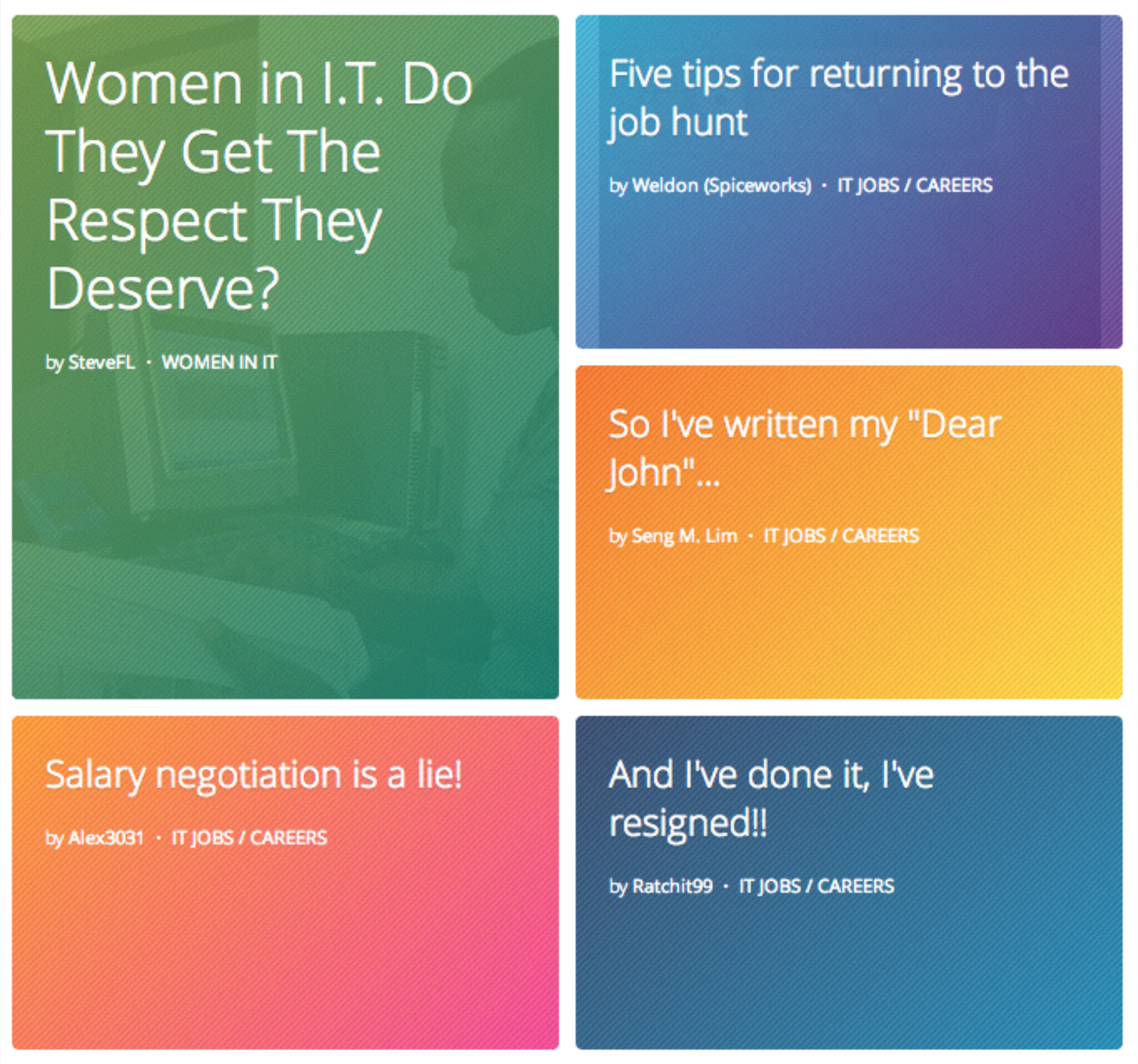


SCSS

```scss
$coolThings: (
  ("computers", red),
  ("dogs", white),
  ("jetskis", blue)
);

@each $item, $color in $coolThings {
  .#{$item} {
    background: $color;
  }
}
```

CSS

```css
.computers {
  background: red;
}

.dogs {
  background: white;
}

.jetskis {
  background: blue;
}
```



Women in I.T. Do They Get The Respect They Deserve?

by SteveFL · WOMEN IN IT

Five tips for returning to the job hunt

by Weldon (Spiceworks) · IT JOBS / CAREERS

So I've written my "Dear John"...

by Seng M. Lim · IT JOBS / CAREERS

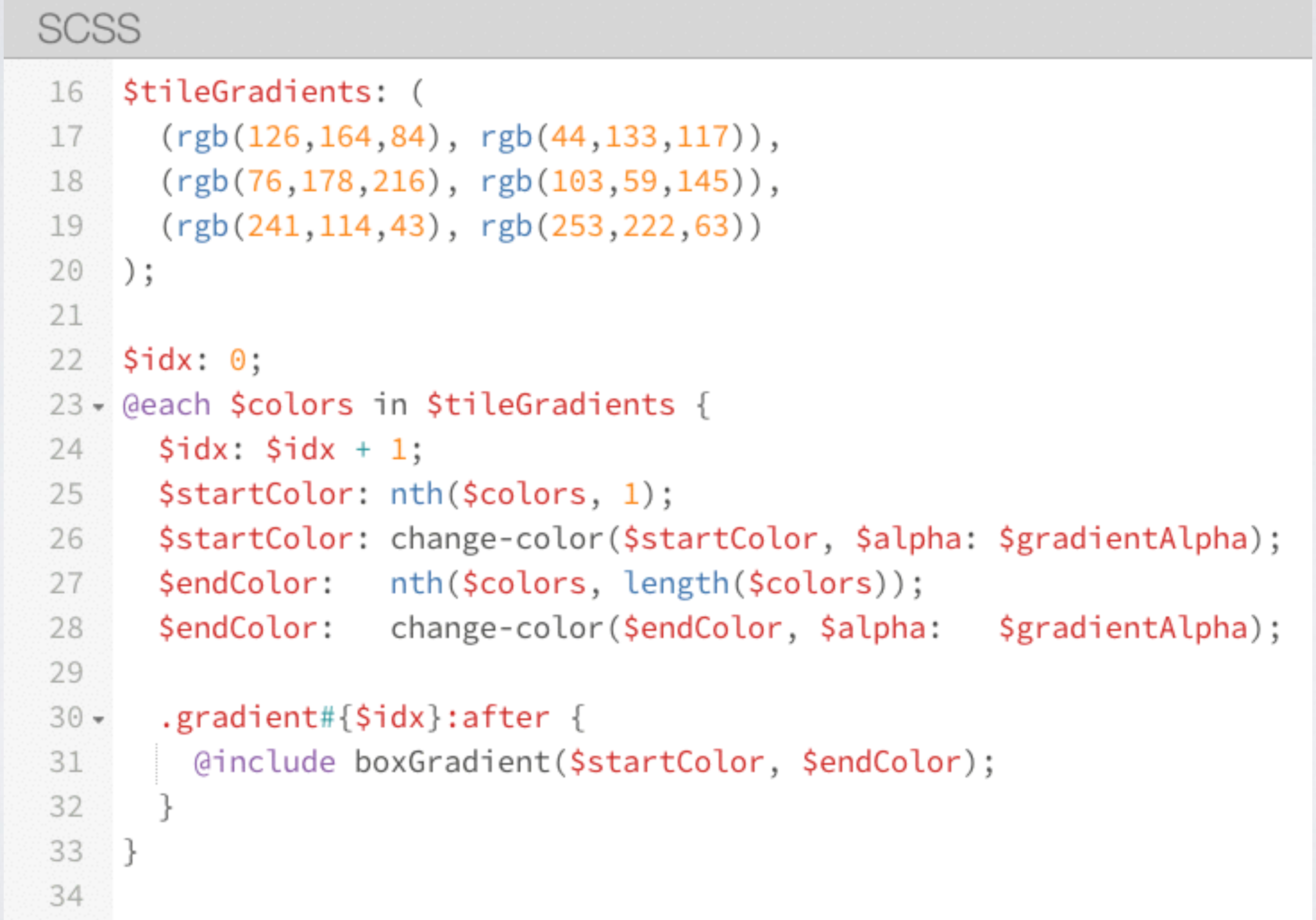
Salary negotiation is a lie!

by Alex3031 · IT JOBS / CAREERS

And I've done it, I've resigned!!

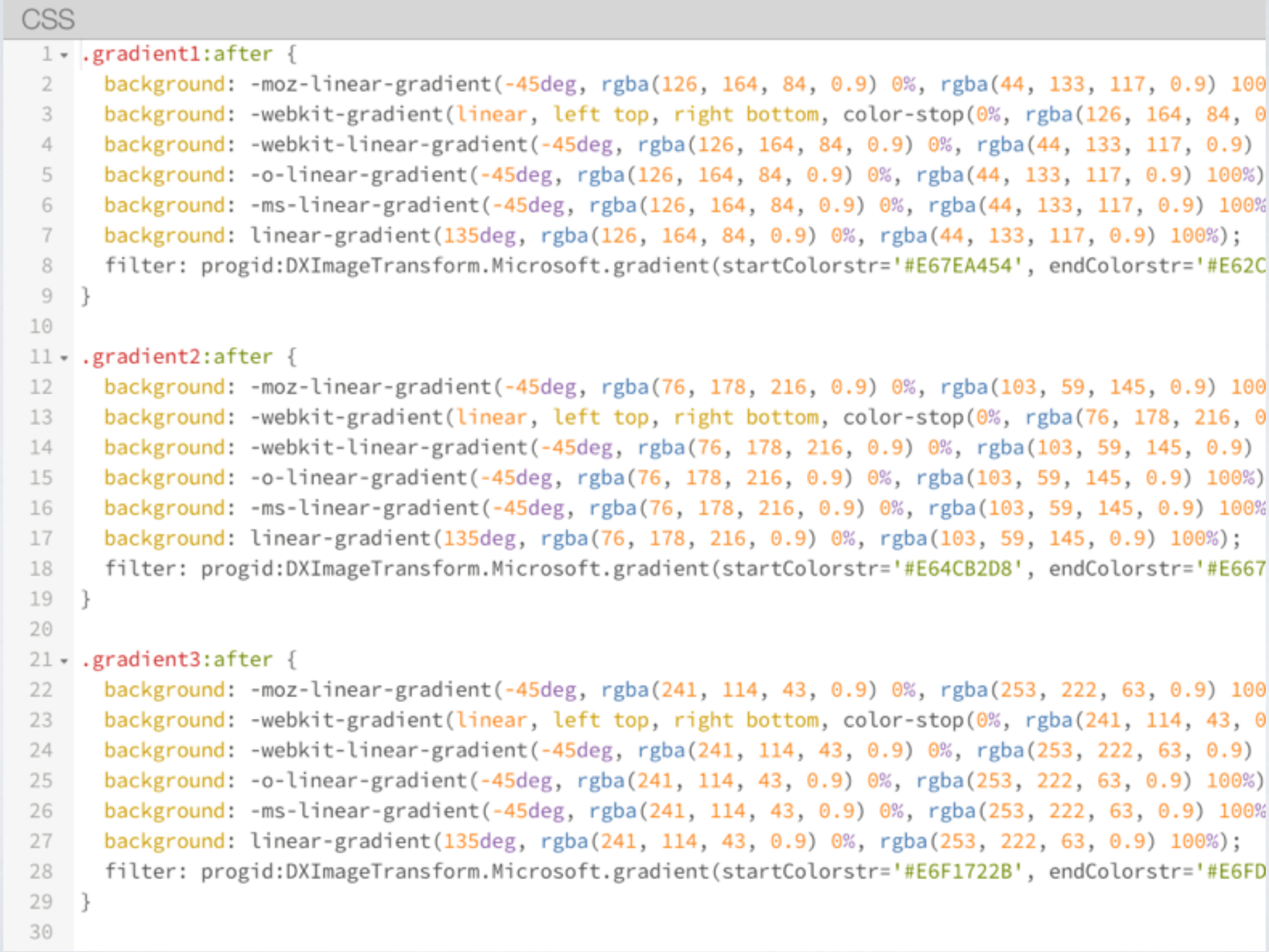
by Ratchit99 · IT JOBS / CAREERS

LISTS — LEVEL 2



```scss
$tileGradients: (
  (rgb(126,164,84), rgb(44,133,117)),
  (rgb(76,178,216), rgb(103,59,145)),
  (rgb(241,114,43), rgb(253,222,63))
);

$idx: 0;
@each $colors in $tileGradients {
  $idx: $idx + 1;
  $startColor: nth($colors, 1);
  $startColor: change-color($startColor, $alpha: $gradientAlpha);
  $endColor:   nth($colors, length($colors));
  $endColor:   change-color($endColor, $alpha:   $gradientAlpha);

  .gradient#{$idx}:after {
    @include boxGradient($startColor, $endColor);
  }
}
```

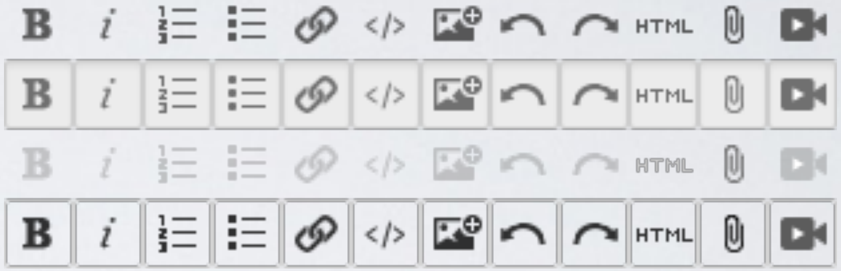
LISTS — LEVEL 2



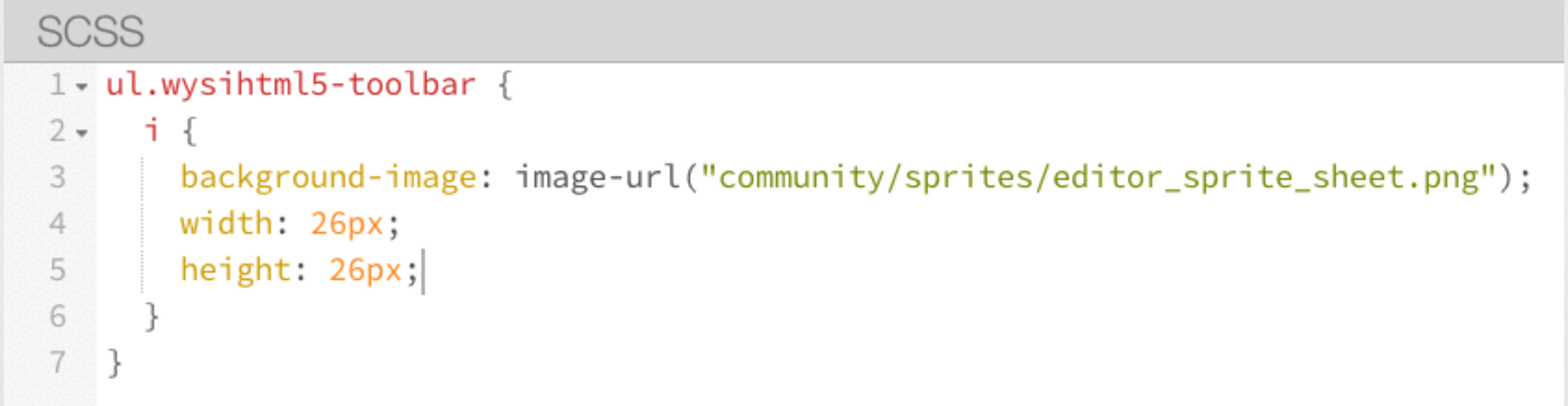
```css
.gradient1:after {
  background: -moz-linear-gradient(-45deg, rgba(126, 164, 84, 0.9) 0%, rgba(44, 133, 117, 0.9) 100
  background: -webkit-gradient(linear, left top, right bottom, color-stop(0%, rgba(126, 164, 84, 0
  background: -webkit-linear-gradient(-45deg, rgba(126, 164, 84, 0.9) 0%, rgba(44, 133, 117, 0.9)
  background: -o-linear-gradient(-45deg, rgba(126, 164, 84, 0.9) 0%, rgba(44, 133, 117, 0.9) 100%)
  background: -ms-linear-gradient(-45deg, rgba(126, 164, 84, 0.9) 0%, rgba(44, 133, 117, 0.9) 100%
  background: linear-gradient(135deg, rgba(126, 164, 84, 0.9) 0%, rgba(44, 133, 117, 0.9) 100%);
  filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#E67EA454', endColorstr='#E62C
}

.gradient2:after {
  background: -moz-linear-gradient(-45deg, rgba(76, 178, 216, 0.9) 0%, rgba(103, 59, 145, 0.9) 100
  background: -webkit-gradient(linear, left top, right bottom, color-stop(0%, rgba(76, 178, 216, 0
  background: -webkit-linear-gradient(-45deg, rgba(76, 178, 216, 0.9) 0%, rgba(103, 59, 145, 0.9)
  background: -o-linear-gradient(-45deg, rgba(76, 178, 216, 0.9) 0%, rgba(103, 59, 145, 0.9) 100%)
  background: -ms-linear-gradient(-45deg, rgba(76, 178, 216, 0.9) 0%, rgba(103, 59, 145, 0.9) 100%
  background: linear-gradient(135deg, rgba(76, 178, 216, 0.9) 0%, rgba(103, 59, 145, 0.9) 100%);
  filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#E64CB2D8', endColorstr='#E667
}

.gradient3:after {
  background: -moz-linear-gradient(-45deg, rgba(241, 114, 43, 0.9) 0%, rgba(253, 222, 63, 0.9) 100
  background: -webkit-gradient(linear, left top, right bottom, color-stop(0%, rgba(241, 114, 43, 0
  background: -webkit-linear-gradient(-45deg, rgba(241, 114, 43, 0.9) 0%, rgba(253, 222, 63, 0.9)
  background: -o-linear-gradient(-45deg, rgba(241, 114, 43, 0.9) 0%, rgba(253, 222, 63, 0.9) 100%)
  background: -ms-linear-gradient(-45deg, rgba(241, 114, 43, 0.9) 0%, rgba(253, 222, 63, 0.9) 100%
  background: linear-gradient(135deg, rgba(241, 114, 43, 0.9) 0%, rgba(253, 222, 63, 0.9) 100%);
  filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#E6F1722B', endColorstr='#E6FD
}
```

MAKING SPRITE SHEETS SUCK LESS

wysiwyg sprites →

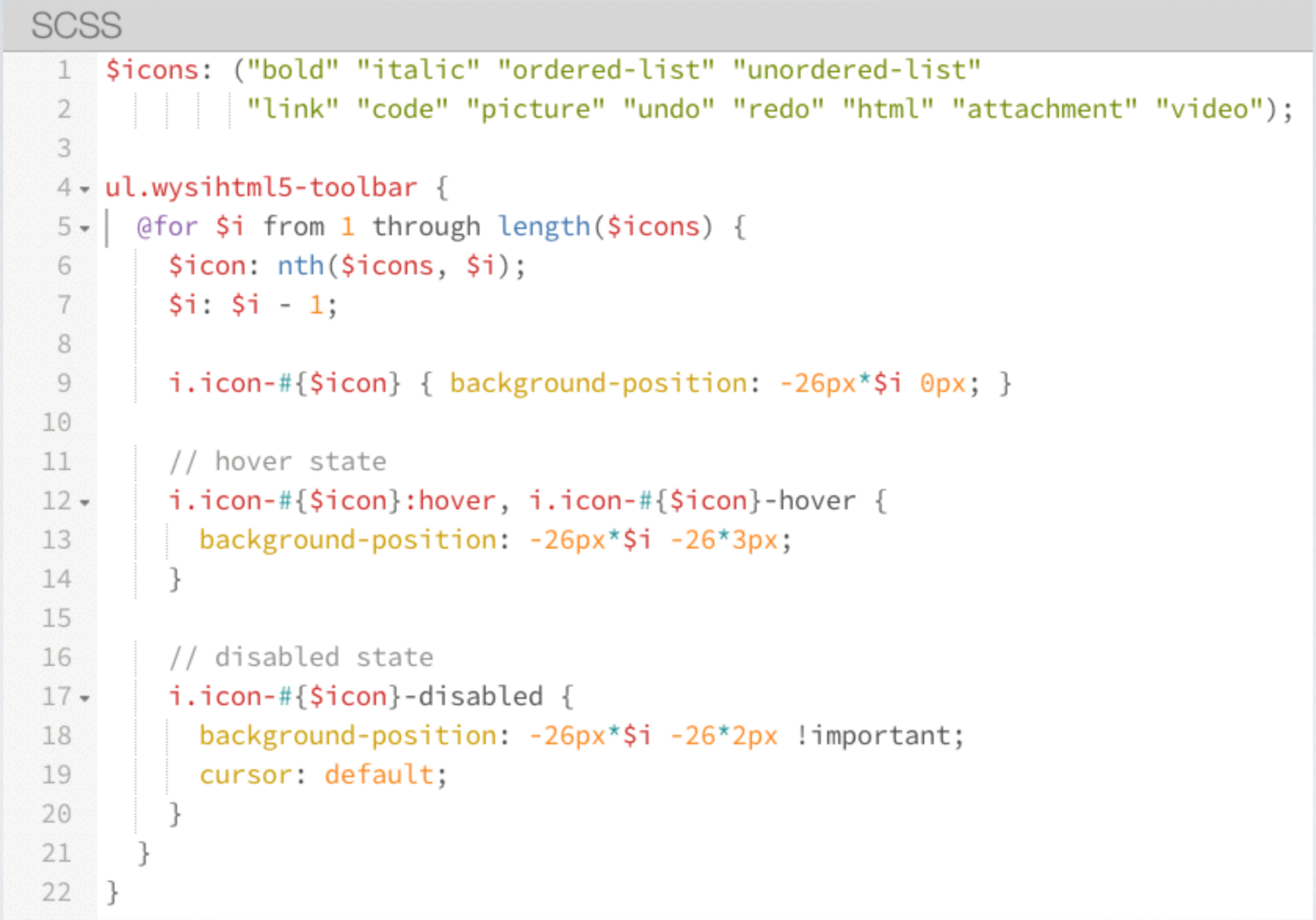


Give all icons the sprite image…
Then we can set the background-position to get the right icon!



SCSS

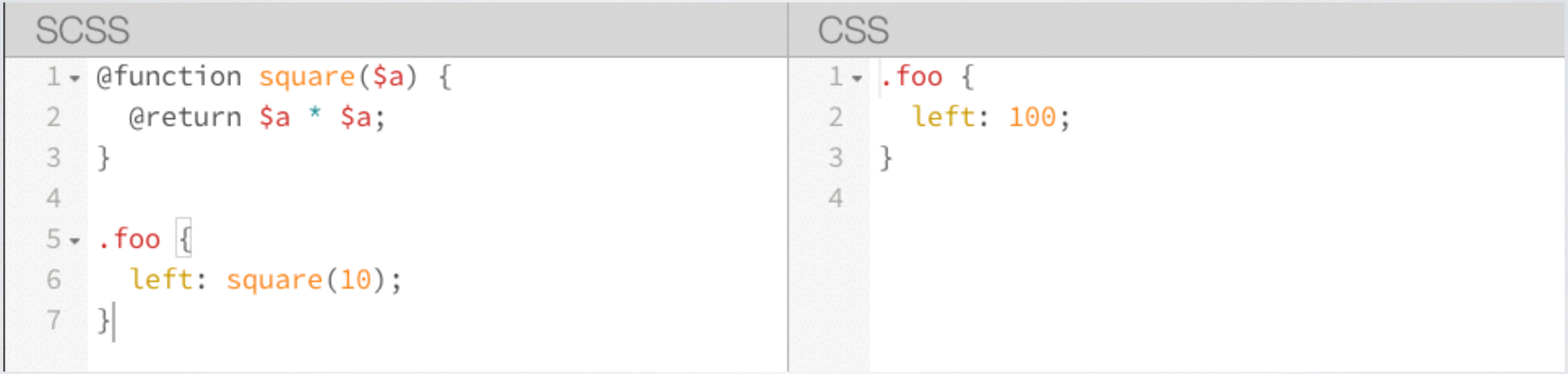
```scss
ul.wysihtml5-toolbar {
  i {
    background-image: image-url("community/sprites/editor_sprite_sheet.png");
    width: 26px;
    height: 26px;
  }
}
```



SCSS

```scss
$icons: ("bold" "italic" "ordered-list" "unordered-list"
         "link" "code" "picture" "undo" "redo" "html" "attachment" "video");

ul.wysihtml5-toolbar {
  @for $i from 1 through length($icons) {
    $icon: nth($icons, $i);
    $i: $i - 1;

    i.icon-#{$icon} { background-position: -26px*$i 0px; }

    // hover state
    i.icon-#{$icon}:hover, i.icon-#{$icon}-hover {
      background-position: -26px*$i -26*3px;
    }

    // disabled state
    i.icon-#{$icon}-disabled {
      background-position: -26px*$i -26*2px !important;
      cursor: default;
    }
  }
}
```

FUNCTIONS!



```scss
@function square($a) {
  @return $a * $a;
}

.foo {
  left: square(10);
}
```

```css
.foo {
  left: 100;
}

```

SASS-RAILS

WHAT IS IT?

- Adds SASS compilation to the asset pipeline
- Provides amazing helpers!
- You will never need to use ERB with your SASS
    - (doing so is henceforth illegal)
    - (if you think you need to use ERB with your SASS, come talk to me first!)
    - (I've only seen one file in the whole Community that had a legit excuse)

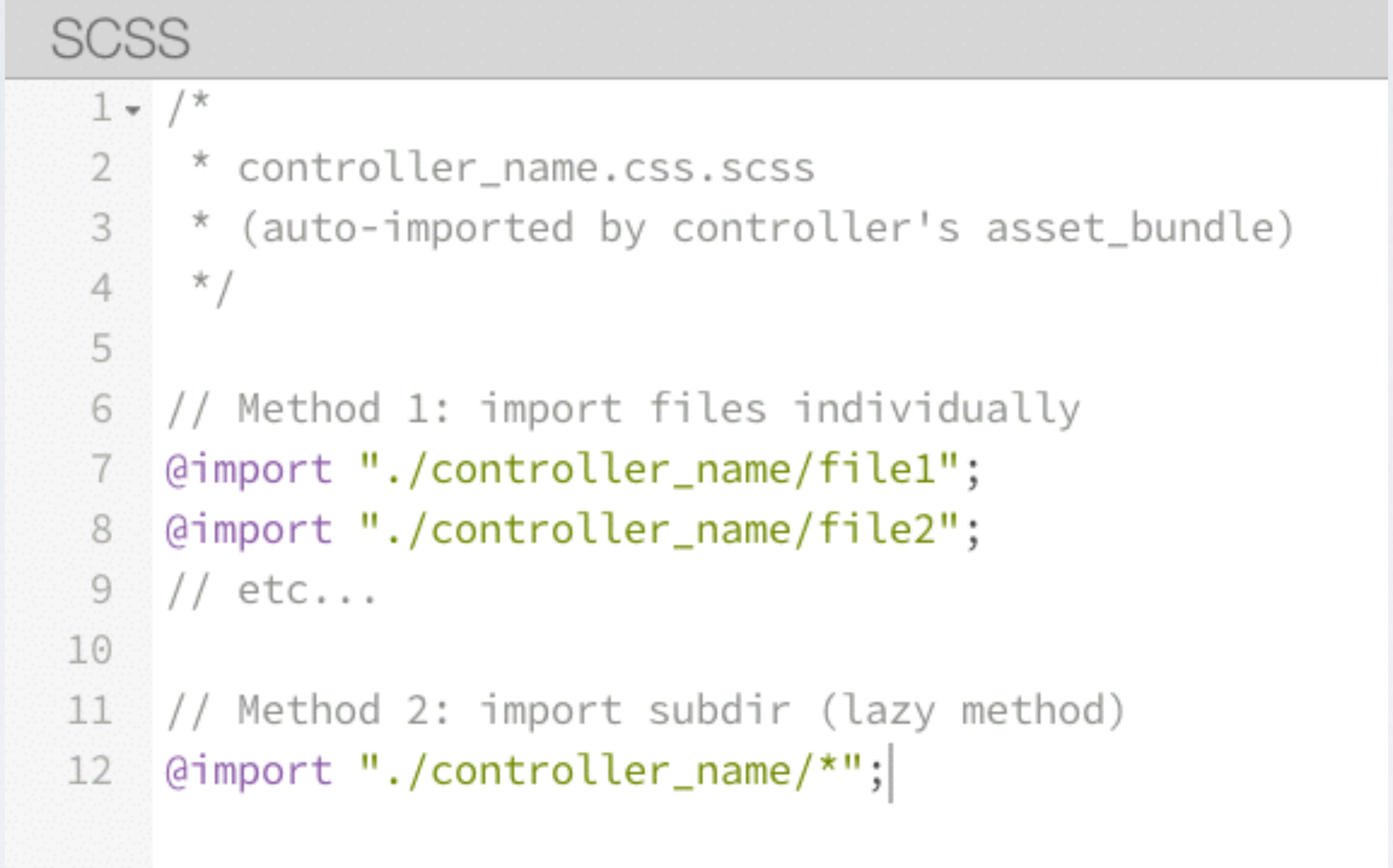
ASSET HELPERS

- asset-path(“puppies.png”, image)
  - “/assets/puppies.png”
- asset-url(“puppies.png”, image)
  - url(/assets/puppies.png)
- image-path(“puppies.png”)
  - “/assets/puppies.png”
- image-url(“puppies.png”)
  - url(/assets/puppies.png)

DON’T USE SPROCKETS INCLUDES WITH SASS!

- **Bad**:
    - `//= require “other_file.css”`
    - `//= require_tree “./directory”`
- **GOOD:**
    - `@import “other_file.css”;`
    - `@import “directory/*”;`

COMMUNITY BEST-PRACTICES



```scss
/*
 * controller_name.css.scss
 * (auto-imported by controller's asset_bundle)
 */

// Method 1: import files individually
@import "./controller_name/file1";
@import "./controller_name/file2";
// etc...

// Method 2: import subdir (lazy method)
@import "./controller_name/*";
```

GENERAL BEST-PRACTICES

DON'T NEST TOO DEEP!

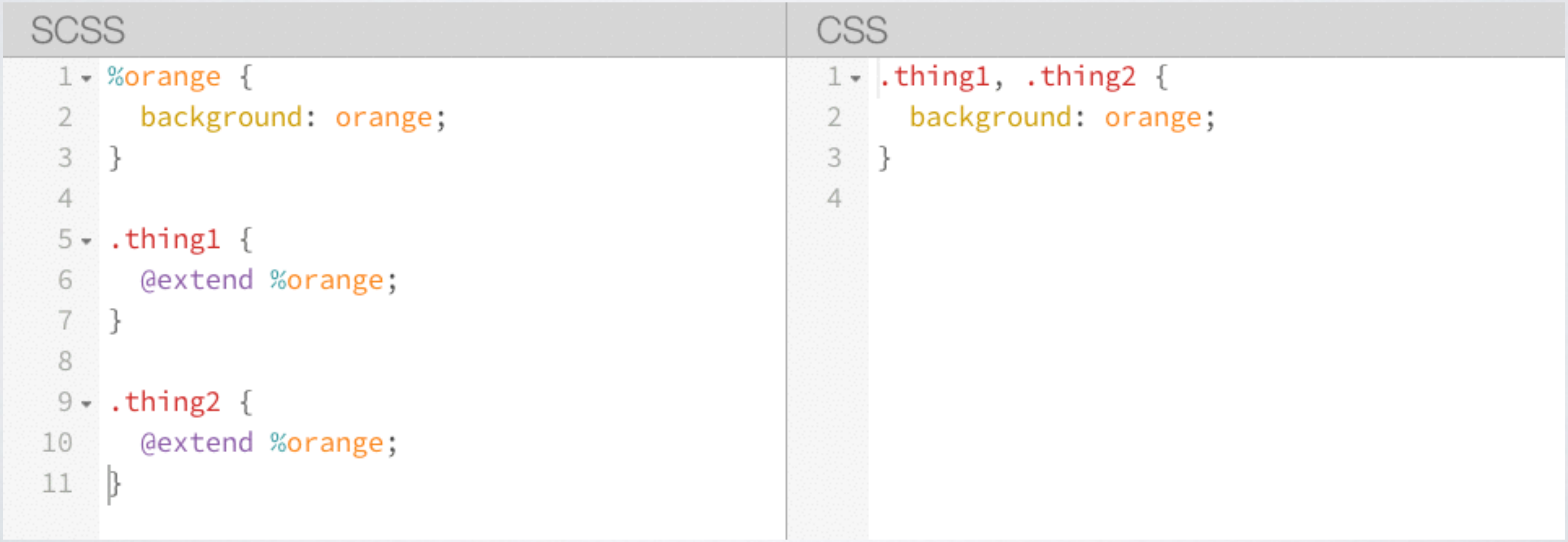
Yes, nesting is great, but avoid nesting it deeper than you would with code in any other language!

(everything in moderation, please)

EXTEND VS. MIXIN

Don’t need arguments for your mixin?

You should probably use @extend instead! (it’s smarter):



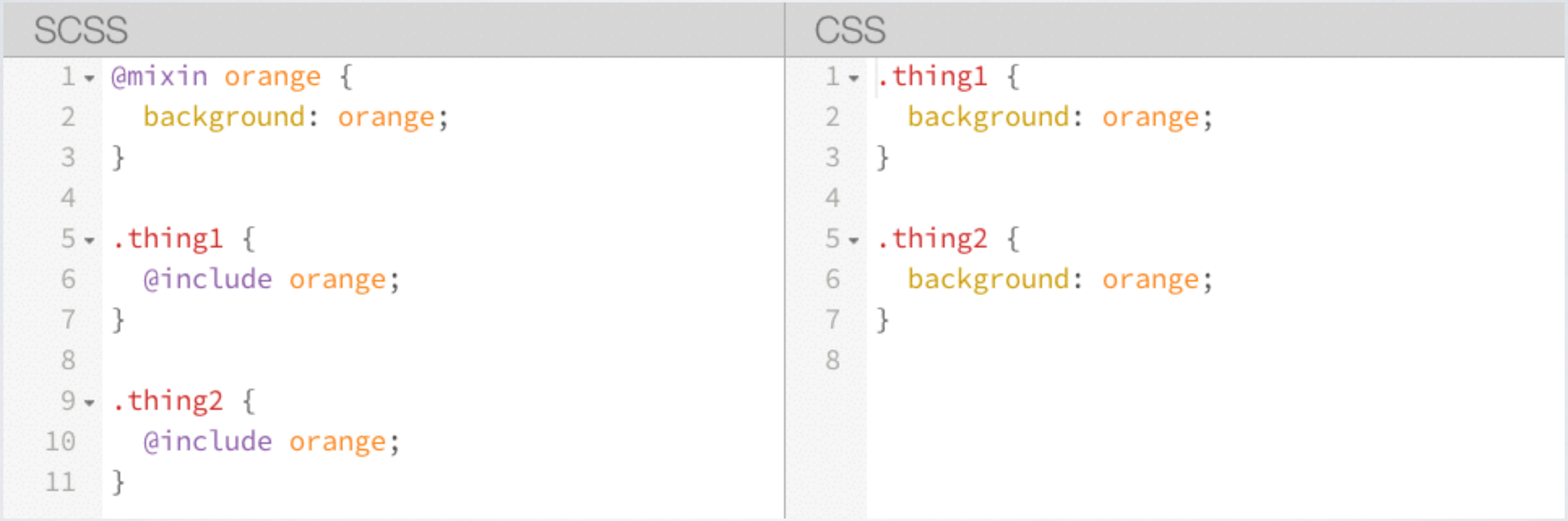
SCSS

```scss
%orange {
  background: orange;
}

.thing1 {
  @extend %orange;
}

.thing2 {
  @extend %orange;
}
```

CSS

```css
.thing1, .thing2 {
  background: orange;
}

```

@mixin just shoves code in wherever…
This example is fine, but it can get out of hand quickly!



SCSS

```scss
@mixin orange {
  background: orange;
}

.thing1 {
  @include orange;
}

.thing2 {
  @include orange;
}
```

CSS

```css
.thing1 {
  background: orange;
}

.thing2 {
  background: orange;
}
```

RESOURCES

- Official docs
- Sassmeister (used to create these code snippets!)
- All of the gorgeous Spiceworks Community codebase 👍

ASK ME STUFF