# You might not need SCSS

Ben Buchanan

weblog.200ok.com.au

Presented at SydCSS, 2019.02.07

## SCSS is awesome!

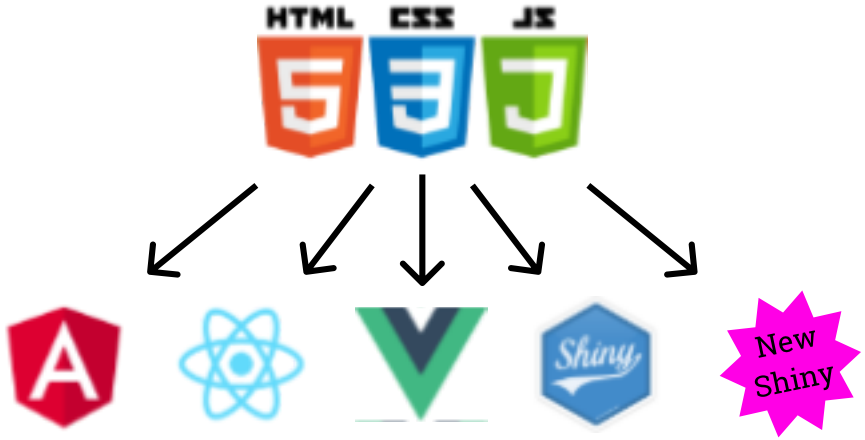
- Appeared in 2006
- Popular
- Works everywhere*

* that you care about

## SCSS is the jQuery of CSS

`youmightnotneedjquery.com` appeared in 2014

Are we even asking the question about SCSS?


HTML
CSS
JS
Shiny
New
Shiny

## Qbit private API

- HTML
- CSS
- ES6

## Qbit Public API

- Templates (Angular, React)
- SCSS variables
- JSON design tokens

## Qbit private API

- HTML
- CSS
- ES6

## Qbit Public API

- Templates (Angular, React)
- ~~SCSS variables~~
- CSS custom props
- JSON design tokens

# The place SCSS has in our stack

## SCSS is awesome at compile time

- Variables
- Calculations
- Mixins
- Style APIs via !default
- Bundling
- Minification
- Prefix management

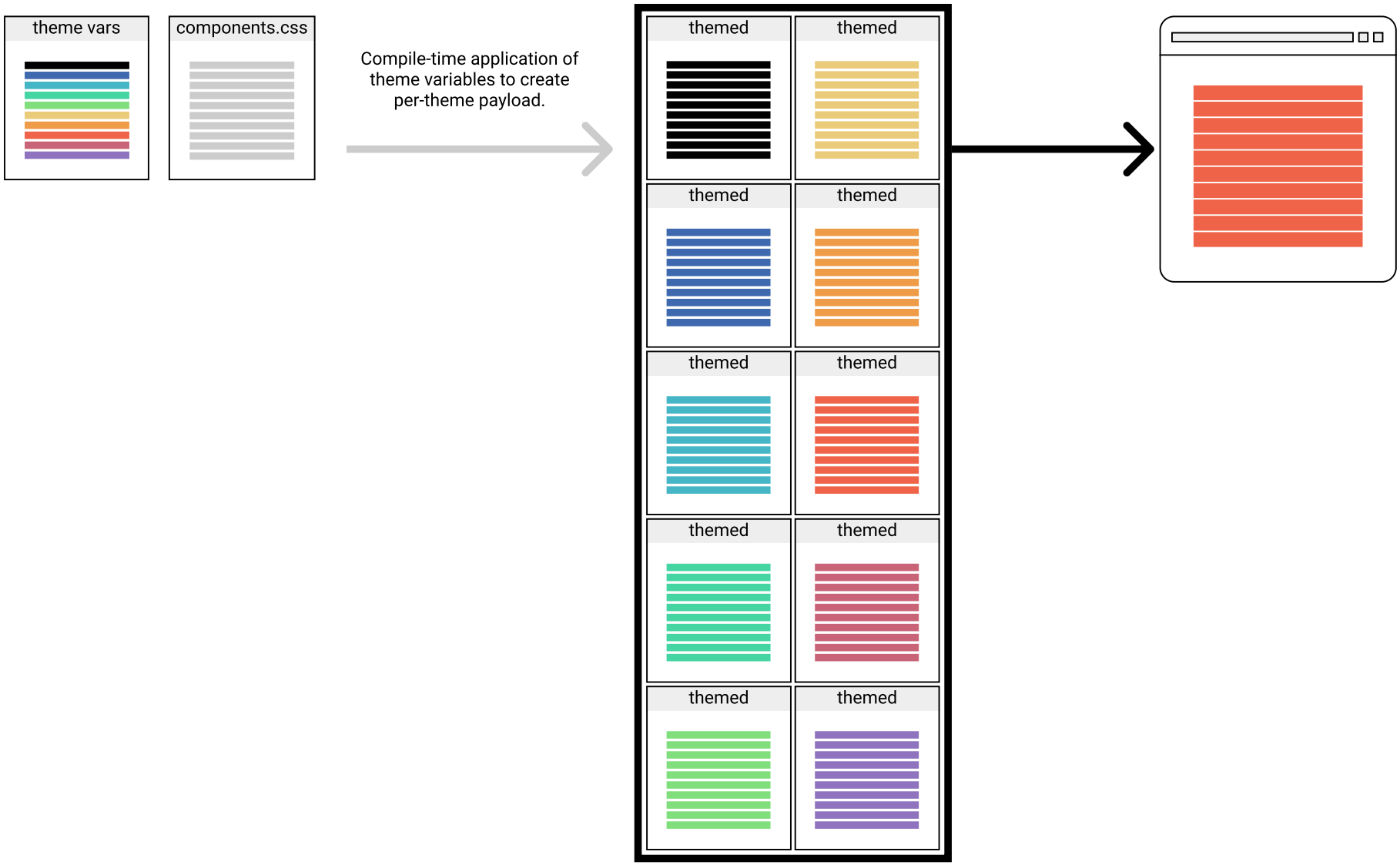
# Detailed list of SCSS runtime features

-
-
-

## SCSS as style API

- Provides error handling :)
- Precompiled modifications only :-/
- Imposes dependencies & stack choice on consumer :(

# SCSS for themes


theme vars
components.css
Compile-time application of theme variables to create per-theme payload.
themed
themed
themed
themed
themed
themed
themed
themed
themed
themed

## I wanted a better way

- Lighter payload
- Native portability
- Runtime features

# CSS Custom Properties & Calc()

## Custom properties

- they're *really* not variables
- affected by **inheritance and the cascade**

# Custom prop basics

```
:root {
  --colour-brand: #f00;
}
```

```
button {
  background: var(--colour-brand);
}
```

## Get/Set with JS

```
// DOM elements
const $el = document.getElementById('target');
const $root = document.documentElement;
// Get
window.getComputedStyle($el).getPropertyValue("--color");
// Set
$el.style.setProperty('--color', 'black');
```

# Themes / Colour Schemes

# Theme by resource

`theme-red.css`

```
:root {
  --colour-brand: #f00;
}
```

`theme-blue.css`

```
:root {
  --colour-brand: #00f;
}
```

`base.css`

```
button {
  background:
    var(--colour-brand);
}
```

# Theme by selector

```
.theme1 {
  --colour-brand: #f00;
}
.theme2 {
  --colour-brand: #00f;
}
```

```
<body class="theme1">
  <button>Red</button>
</body>
```

```
<button class="theme1">
  Red
</button>
<button class="theme2">
  Blue
</button>
```

## SCSS component vars

```
/* core var */
$colour-brand: #f00 !default;

/* component var */
$colour-button: $colour-brand !default;
```

```
button {
  background: $colour-button;
}
```

# CSS component props

```
/* set a default */
:root {
  --colour-button: var(--colour-brand);
}
```

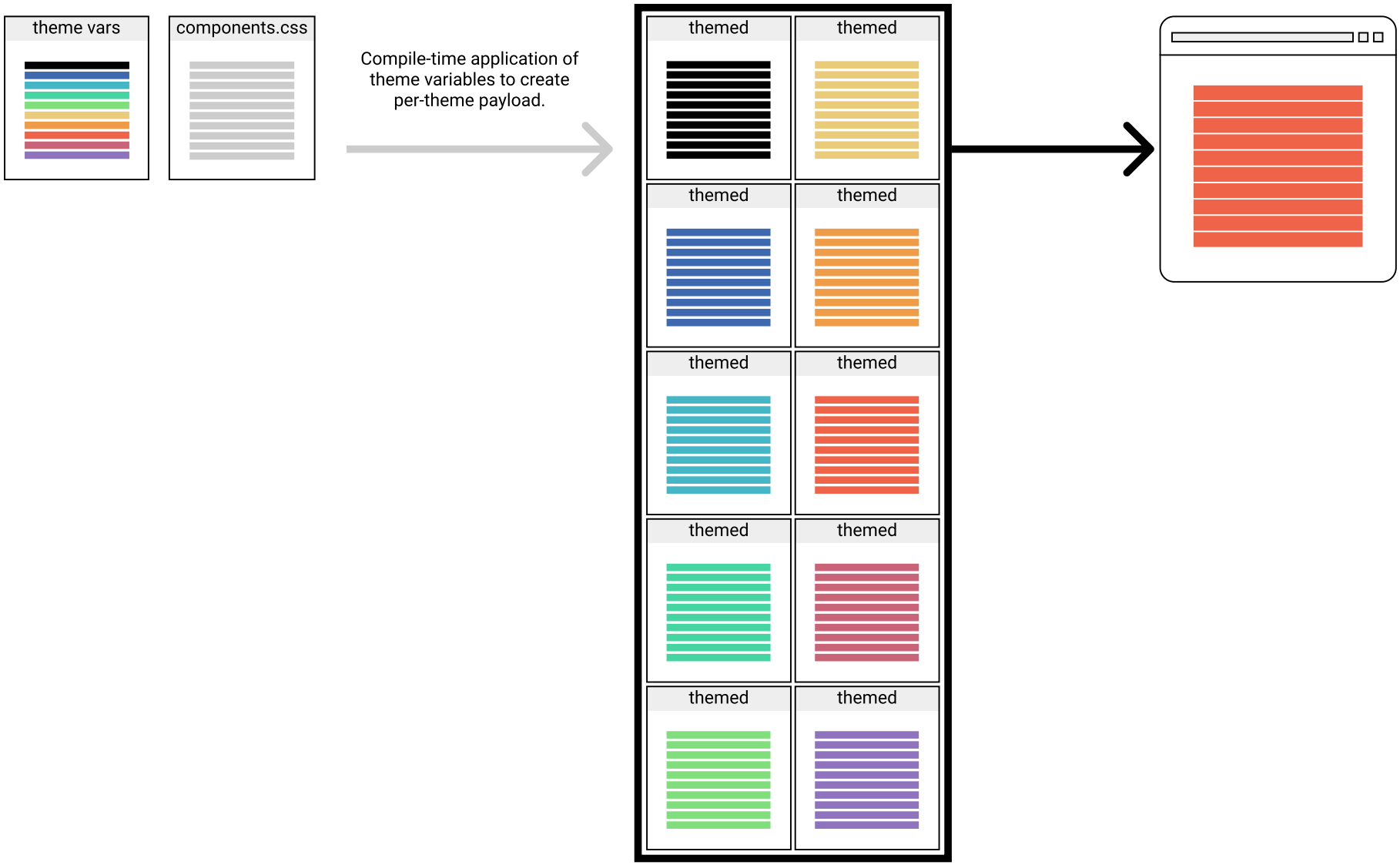
```
/* component var needs a boost. */
.theme1,
.theme2 {
  --colour-button: var(--colour-brand);
}
```

## But you don't need it!

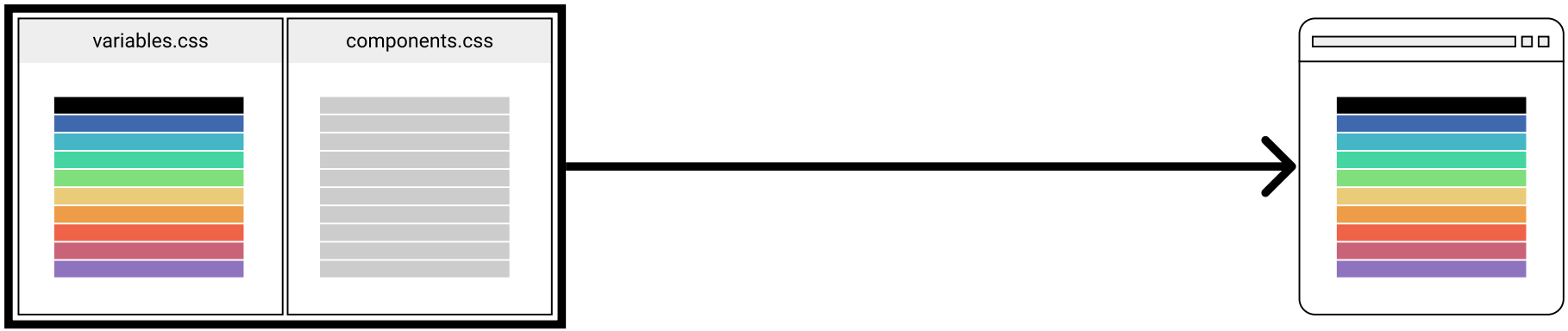
```
.my-custom-theme {
  --colour-brand: #0f0;
}
```

```
.my-custom-theme button {
  --colour-brand: #00f;
}
```

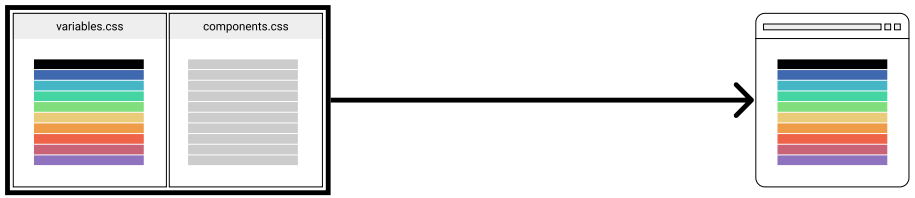
# Remember SCSS payloads?


theme vars
components.css
Compile-time application of theme variables to create per-theme payload.
themed
themed
themed
themed
themed
themed
themed
themed
themed
themed

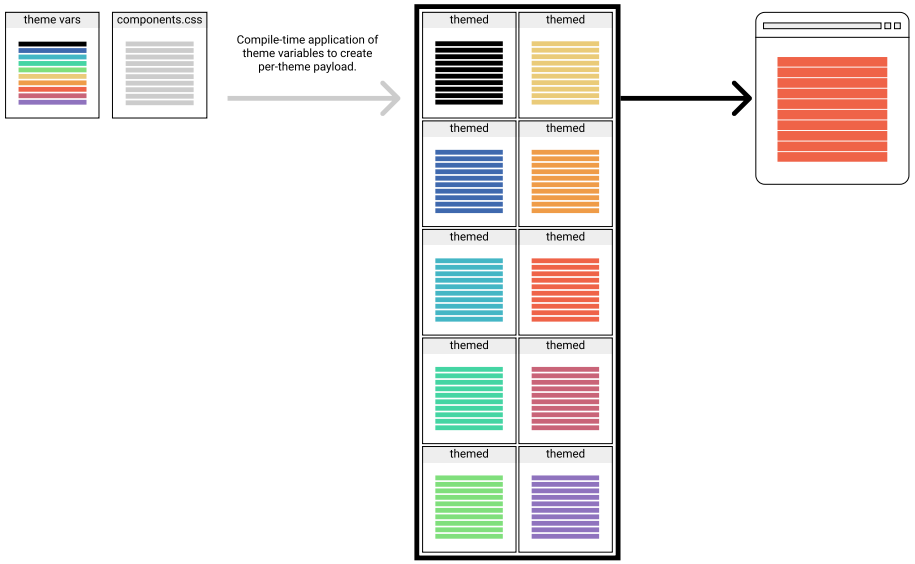
# CSS variable payload



# CSS variable payload




# SCSS variable payloads



## calc()

```
:root {
  --header-height: calc(var(--grid-unit) * 12);
}
```

# calc() voodoo

```
header {
  height: 50px;
}
main {
  height: calc(100% - 50px);
}
```

Mixed units.

## Tip: negative values

```
calc(valueYouNeedToBeNegative * -1)
```

## Gotcha: dang browsers

Some calculations fall foul of browser variation.
SCSS is at least normalised calculation.

# 90% of the effort is in the last 10%

## Think of ~~the children~~IE11

- Calc: supported but buggy
- Custom Props: no support

# PostCSS to the rescue!

```
postcss([
  postCSSCustomProperties({
    importFrom: './dist/css-variables.css'
  }),
  postCSSCalc()
]))
```

```
.selector {
  margin-top: 40px;
  margin-top: calc(var(--space-l) * 1.25);
}
```

## A choice

1. Put up with IE11 bloat in your CSS
2. Serve IE11 its own CSS

# Heresy alert!

```
var ua = navigator.userAgent;
var uamatch = ua.match(/(msie|trident(?=\/))\/?\s*([\d\.]+)/i) || [];
if(/trident/i.test(uamatch[1]) && parseInt(/\brv[ :]+(\d+)/g.exec(ua)[1], 10) === 11) {
  document.querySelector('head').innerHTML += '<link rel="stylesheet" href="ie11.css" type
}
```

Available at https://bitbucket.org/snippets/200ok

## What I really missed... was errors

```
Error: Undefined variable: "$button-border-widht".
  on line 31 of path/to/button.scss
  from line 8 of path/to/qbit-all.scss
>> -width: #{$button-border-widht};
   -------------------------------^
```

## What I really missed... was errors

```
.selector {
  color: var(--colour-that-does-not-exist);
}
```

This was really hard to debug.

## Stylelint to the rescue!

`gulpStylelint`

```
"plugins": [
  "stylelint-scss",
  "stylelint-value-no-unknown-custom-properties"
]
```

## Stylelint to the rescue!

`gulpStylelint`

```
path/to/component.css
 85:18  ✖  Unexpected custom property
 "--colour-hoover" inside declaration "color".
```

# Do we still need SCSS?

## CSS

- Variables
- Calculations
- Runtime power

## Many tools

- Bundling
- Concatenation
- Postprocessing
- Linting/errors

## SCSS

- Mixins
- Loops

## Demand native specs!

- Mixins
- Loops
- CSS `require`

## You might still use SCSS...

...for massive projects

...because it's still right for you

## You might not need SCSS...

...in your API

...for small projects

# Thank you.