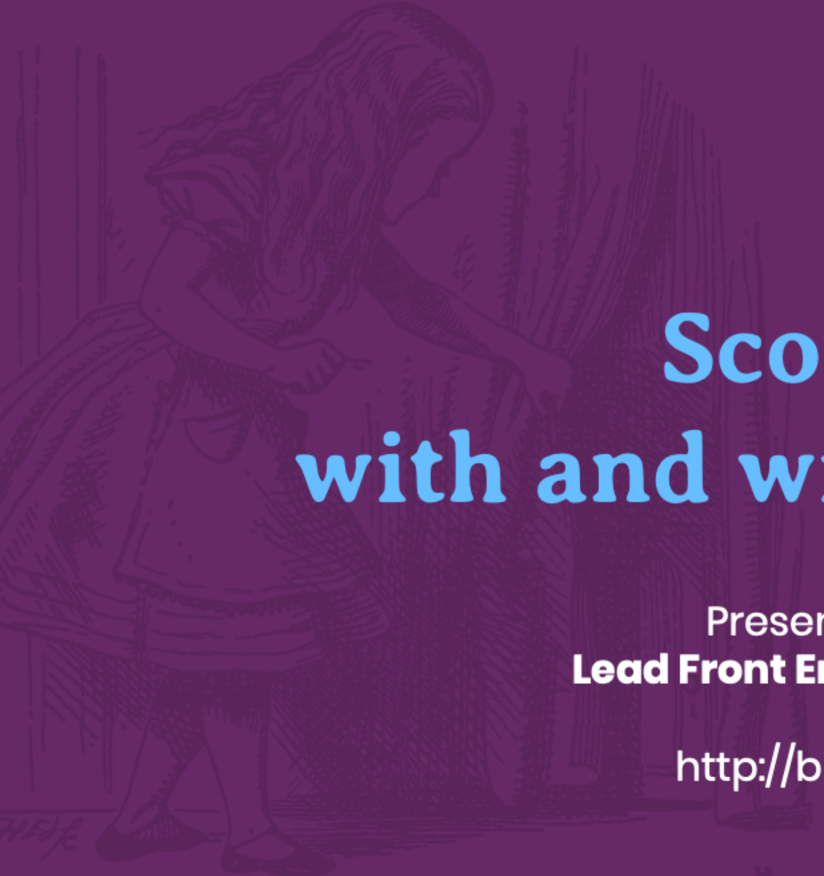
March 19, 2020



Scope in CSS with and without JavaScript

Presented by: **Brian Perry**
Lead Front End Developer - Bounteous

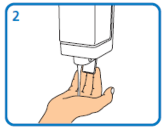
http://bit.ly/scope-midcamp

WE COOL?

Hand-washing technique with soap and water

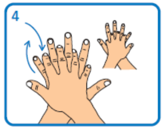
1

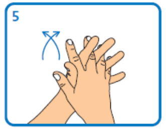
Yeah, I'm gonna take my horse to the old town road

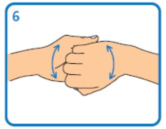
2

I'm gonna ride 'til I can't no more

3

I'm gonna take my horse to the old town road

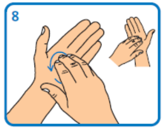
4

I'm gonna ride 'til I can't no more

5

(Kio, Kio)

6

I got the horses in the back

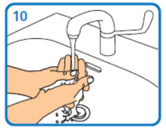
7

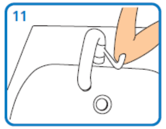
Horse tack is attached

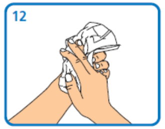
8

Hat is matte black

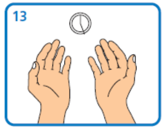
9

Got the boots that's black to match

10

Ridin' on a horse, ha

11

You can whip your Porsche

12

I been in the valley

13

You ain't been up off that porch, now

Create your own
https://washyourlyrics.com

Old Town Road
Lil Nas X

BRIAN PERRY

- Lead Front End Dev at Bounteous
- Rocking the Chicago ‘burbs
- Lover of all things components…

  …and Nintendo

d.o: brianperry

twitter: bricomedy

github: backlineint

nintendo: wabrian

brianperryinteractive.com

bounteous

MOUNTAIN STATES HEALTH ALLIANCE

fitbit

vsp Vision care for life

zipcar

Allstate You're in good hands.

EyeMed VISION CARE

Centura Health

FEDERAL-MOGUL MOTORPARTS

P&G

CHS Community Health Systems

AON Hewitt

Wilson

spring

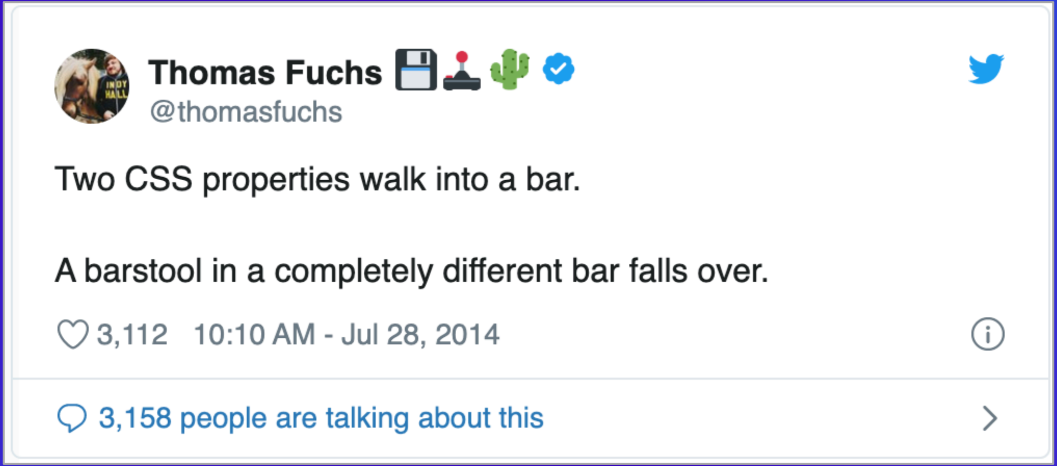
invaluable

CSS has **global scope.**

CSS has **global scope.**

That’s great when you want rules to apply globally…

… and not so great when you don’t.



Thomas Fuchs 💾🕹️🌵
@thomasfuchs

Two CSS properties walk into a bar.

A barstool in a completely different bar falls over.

3,112 10:10 AM - Jul 28, 2014

3,158 people are talking about this

Many **CSS in JavaScript** solutions provide **component scope.**

CSS: global scope

CSS-in-JS: component scope

People disagree about these approaches…
because… internet.

I think we should **take advantage of both**.

Thank You!

Brian Perry
Lead Front End Developer

Email: brian.perry@bounteous.com

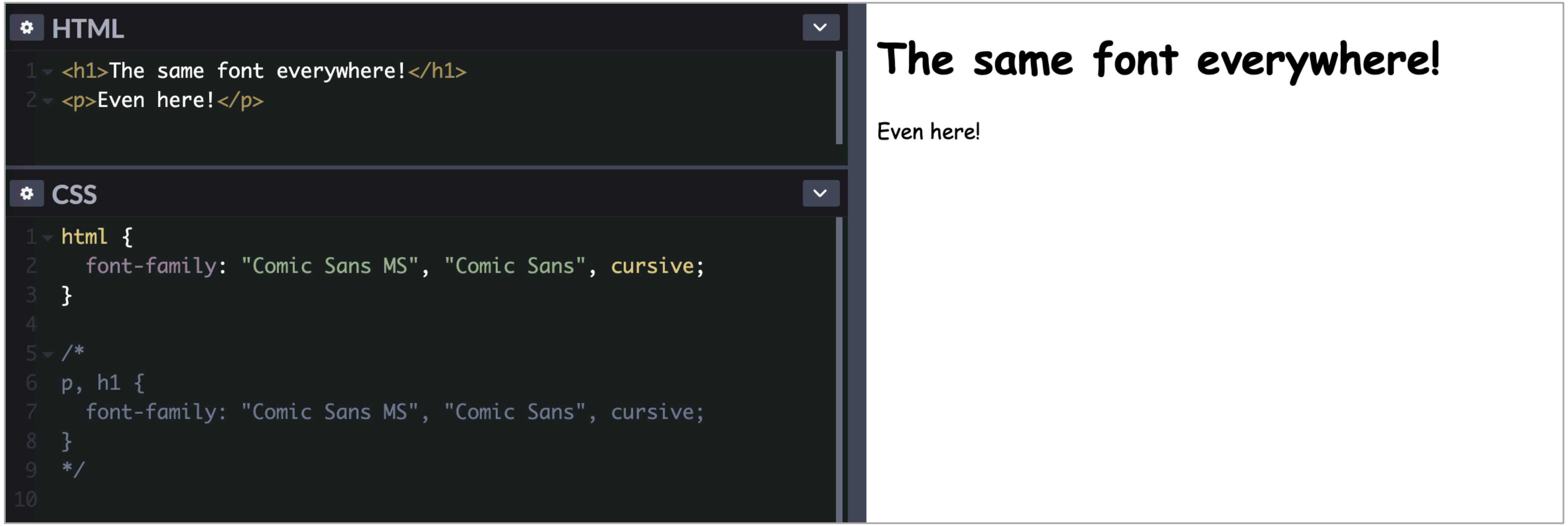
March 19, 2020

Scope in CSS with and without JavaScript

Presented by: **Brian Perry**
Lead Front End Developer - Bounteous

http://bit.ly/scope-midcamp

CSS has **global scope.**



```html
<h1>The same font everywhere!</h1>
<p>Even here!</p>
```

```css
html {
  font-family: "Comic Sans MS", "Comic Sans", cursive;
}

/*
p, h1 {
  font-family: "Comic Sans MS", "Comic Sans", cursive;
}
*/
```

The same font everywhere!

Even here!

Codepen: https://codepen.io/brianperry/pen/rNNQpyK

THE CASCADE (THE C IN CSS)

Stylesheets can have three different origins

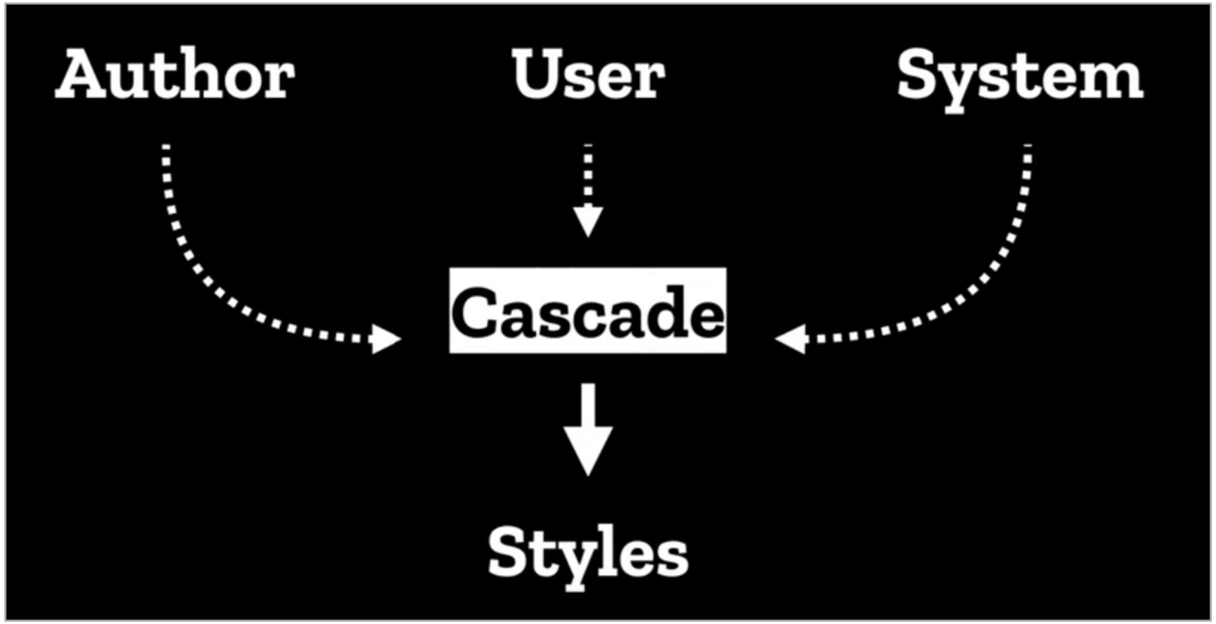
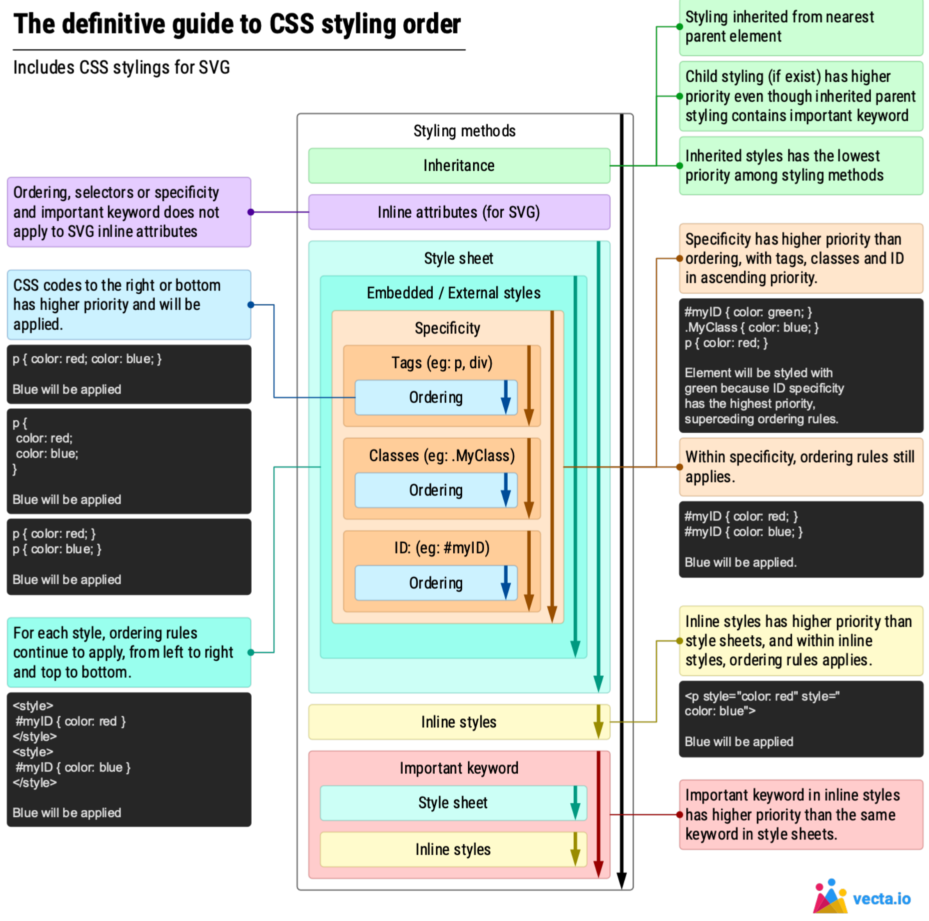


Image: https://www.miriamsuzanne.com/2019/10/03/css-is-weird/



The definitive guide to CSS styling order

Includes CSS stylings for SVG

Ordering, selectors or specificity and important keyword does not apply to SVG inline attributes

CSS codes to the right or bottom has higher priority and will be applied.

```
p { color: red; color: blue; }

Blue will be applied
```

```
p {
 color: red;
 color: blue;
}

Blue will be applied
```

```
p { color: red; }
p { color: blue; }

Blue will be applied
```

For each style, ordering rules continue to apply, from left to right and top to bottom.

```
<style>
 #myID { color: red }
</style>
<style>
 #myID { color: blue }
</style>

Blue will be applied
```

Styling inherited from nearest parent element

Child styling (if exist) has higher priority even though inherited parent styling contains important keyword

Inherited styles has the lowest priority among styling methods

Specificity has higher priority than ordering, with tags, classes and ID in ascending priority.

```
#myID { color: green; }
.MyClass { color: blue; }
p { color: red; }

Element will be styled with
green because ID specificity
has the highest priority,
superceding ordering rules.
```

Within specificity, ordering rules still applies.

```
#myID { color: red; }
#myID { color: blue; }

Blue will be applied.
```

Inline styles has higher priority than style sheets, and within inline styles, ordering rules applies.

```
<p style="color: red" style="
color: blue">

Blue will be applied
```

Important keyword in inline styles has higher priority than the same keyword in style sheets.

The way all rules are combined is very easy to understand and remember.

Image: https://css-tricks.com/the-c-in-css-the-cascade/#article-header-id-11

bounteous

CASCADING ORDER (ACCORDING TO W3C SPEC)

- Find all declarations that apply to the element and property for the target media type.
- **Sort according to importance and origin** (ascending order)
  - **User agent** declarations
  - **User normal** declarations
  - **Author normal** declarations
  - **Author !important** declarations
  - **User !important** declarations
- Sort rules with same origin and importance by **specificity**
- If all else fails, sort by **order specified**

Source: https://www.w3.org/TR/CSS2/cascade.html#cascade

bounteous

SPECIFICITY

element selector
Specificity: 0,0,1

class selector
attribute selector
pseudo-class
Specificity: 0,1,0 (10)

id selector
Specificity: 1,0,0

style attribute
Specificity: 1,0,0,0

(Note: The universal selector and inherited selectors have a specificity of 0, 0, 0, 0)

Source: https://stuffandnonsense.co.uk/archives/css_specificity_wars.html

Stuff & Nonsense

a

1 x element selector

Sith power: 0,0,1

p a

2 x element selectors

Sith power: 0,0,2

.foo

1 x class selector *

Sith power: 0,1,0

a.foo

1 x element selector
1 x class selector

Sith power: 0,1,1

p a.foo

2 x element selectors
1 x class selector

Sith power: 0,1,2

.foo .bar

2 x class selectors

Sith power: 0,2,0

p.foo a.bar

2 x element selectors
2 x class selectors

Sith power: 0,2,2

#foo

1 x id selector

Sith power: 1,0,0

a#foo

1 x element selector
1 x id selector

Sith power: 1,0,1

.foo a#bar

1 x element selector
1 x class selector
1 x id selector

Sith power: 1,1,1

.foo .foo #foo

2 x class selectors
1 x id selector

Sith power: 1,2,0

style

1 x style attribute

Sith power: 1,0,0,0

*** Same specificity**
class selector =
attribute attribute =
pseudo-classes

!important

bounteous

CSS-in-JS can provide **component scope** ...and other advantages.

WHY CSS-IN-JS?

- **Component scope** – styles can't 'leak out'
- **Co-locate styles with components**
  - Positive developer experience
  - An advantage for distributed and composable components
- **Dead code elimination** – bundler knows if css is being used
- **Worry free naming** – don't have to worry about conflicts or enforcing a naming convention
- **Respond to props** rather than juggling class names
- And yes, probably fear / lack of knowledge in some cases

AND WHY NOT?

- You **still need to know how CSS works** (even the hard stuff)
- Can come with a **bundle size / performance cost**
- **Potential barrier of entry** for CSS experts who aren’t JS experts
- **Additional layers of abstraction**
  - Some approaches use less css more object-like syntax
  - Potential syntax highlighting issues
- Risk of **inconsistency / one offs**

So which approach is **right?**

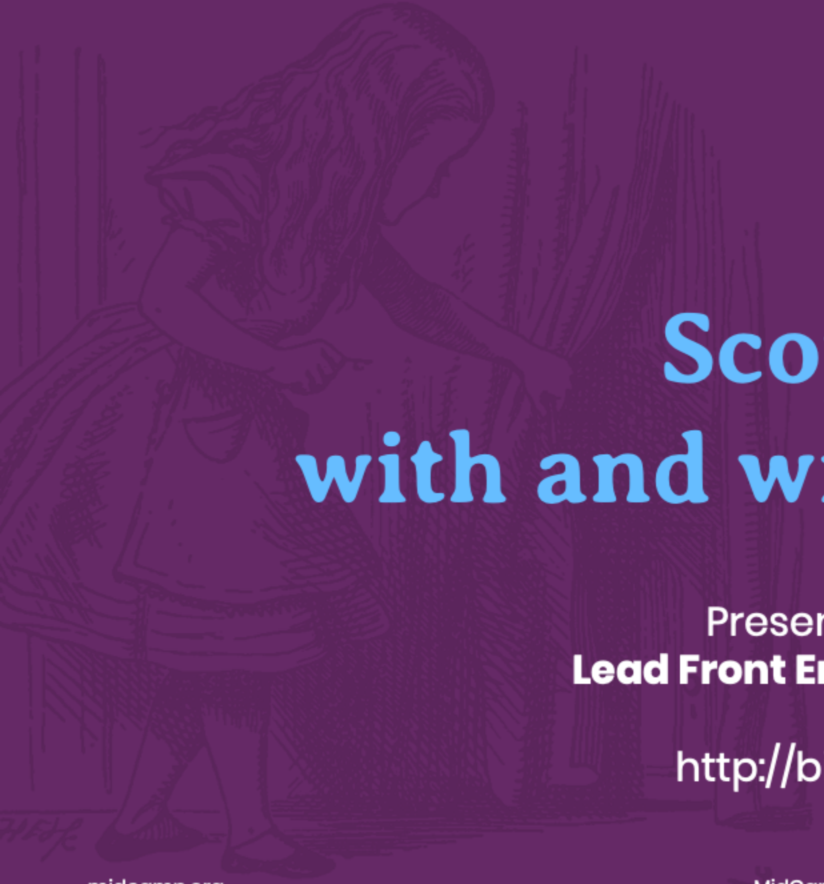
Why don't we have both?

Thank You!

Brian Perry
Lead Front End Developer

Email: brian.perry@bounteous.com

March 19, 2020



Scope in CSS with and without JavaScript

Presented by: **Brian Perry**
Lead Front End Developer - Bounteous

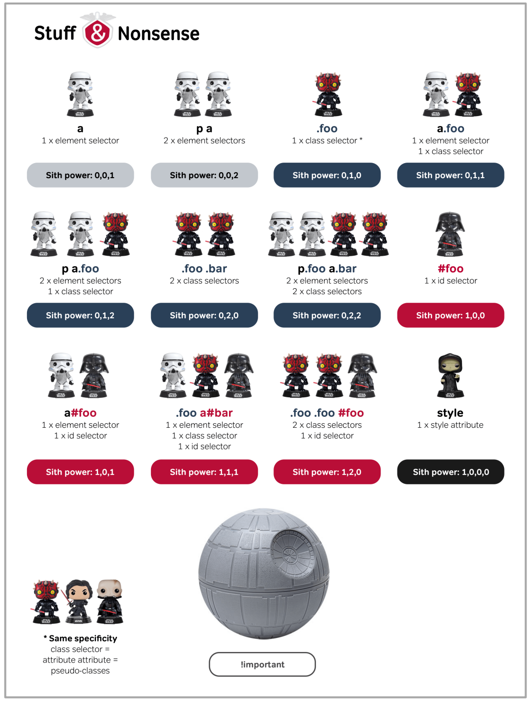
http://bit.ly/scope-midcamp

CSS has **global scope.**

HEY, REMEMBER CSS?

Cascade

Specificity



Stuff & Nonsense

a 1 x element selector Sith power: 0,0,1	**p a** 2 x element selectors Sith power: 0,0,2	**.foo** 1 x class selector * Sith power: 0,1,0	**a.foo** 1 x element selector 1 x class selector Sith power: 0,1,1
p a.foo 2 x element selectors 1 x class selector Sith power: 0,1,2	**.foo .bar** 2 x class selectors Sith power: 0,2,0	**p.foo a.bar** 2 x element selectors 2 x class selectors Sith power: 0,2,2	**#foo** 1 x id selector Sith power: 1,0,0
a#foo 1 x element selector 1 x id selector Sith power: 1,0,1	**.foo a#bar** 1 x element selector 1 x class selector 1 x id selector Sith power: 1,1,1	**.foo .foo #foo** 2 x class selectors 1 x id selector Sith power: 1,2,0	**style** 1 x style attribute Sith power: 1,0,0,0

*** Same specificity**
class selector =
attribute attribute =
pseudo-classes

!important

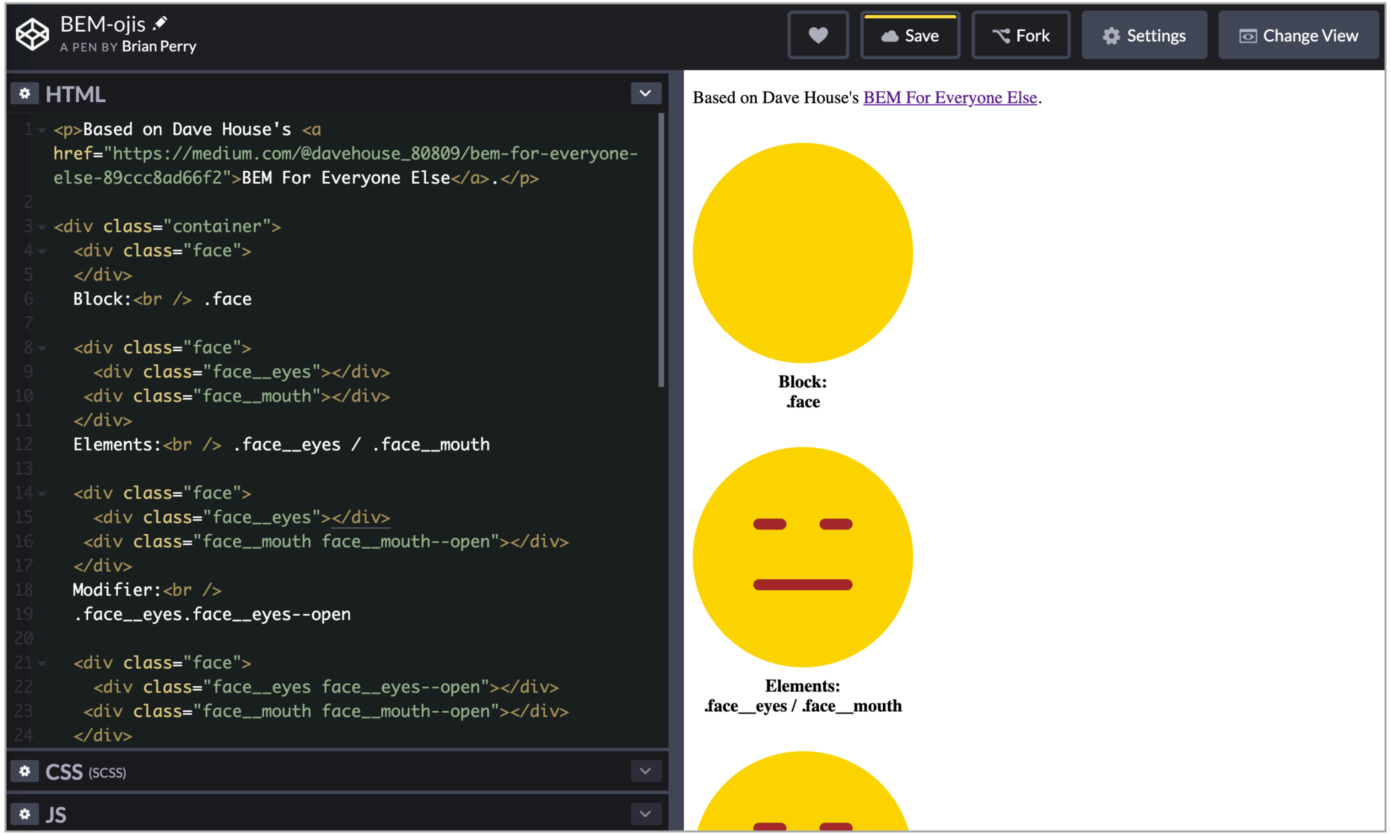
Inheritance

Some of this is tricky. **We should get our act together...**

CSS METHODOLOGIES

- Popular Methodologies
  - Object-Oriented CSS (OOCSS)
  - Block, Element, Modifier (BEM)
  - Scalable and Modular Architecture for CSS (SMACSS)
- File organization
- Naming conventions
- Can create something that ‘feels’ like component scope.

BEM



```html
<p>Based on Dave House's <a
href="https://medium.com/@davehouse_80809/bem-for-everyone-
else-89ccc8ad66f2">BEM For Everyone Else</a>.</p>

<div class="container">
  <div class="face">
  </div>
  Block:<br /> .face

  <div class="face">
    <div class="face__eyes"></div>
   <div class="face__mouth"></div>
  </div>
  Elements:<br /> .face__eyes / .face__mouth

  <div class="face">
    <div class="face__eyes"></div>
   <div class="face__mouth face__mouth--open"></div>
  </div>
  Modifier:<br />
  .face__eyes.face__eyes--open

  <div class="face">
    <div class="face__eyes face__eyes--open"></div>
   <div class="face__mouth face__mouth--open"></div>
  </div>
```

Based on Dave House's BEM For Everyone Else.

Block:
.face

Elements:
.face__eyes / .face__mouth

Concept: https://medium.com/@davehouse_80809/bem-for-everyone-else-89ccc8ad66f2 Codepen: https://codepen.io/brianperry/pen/BaaMYMJ

bounteous

BEM IS GREAT!

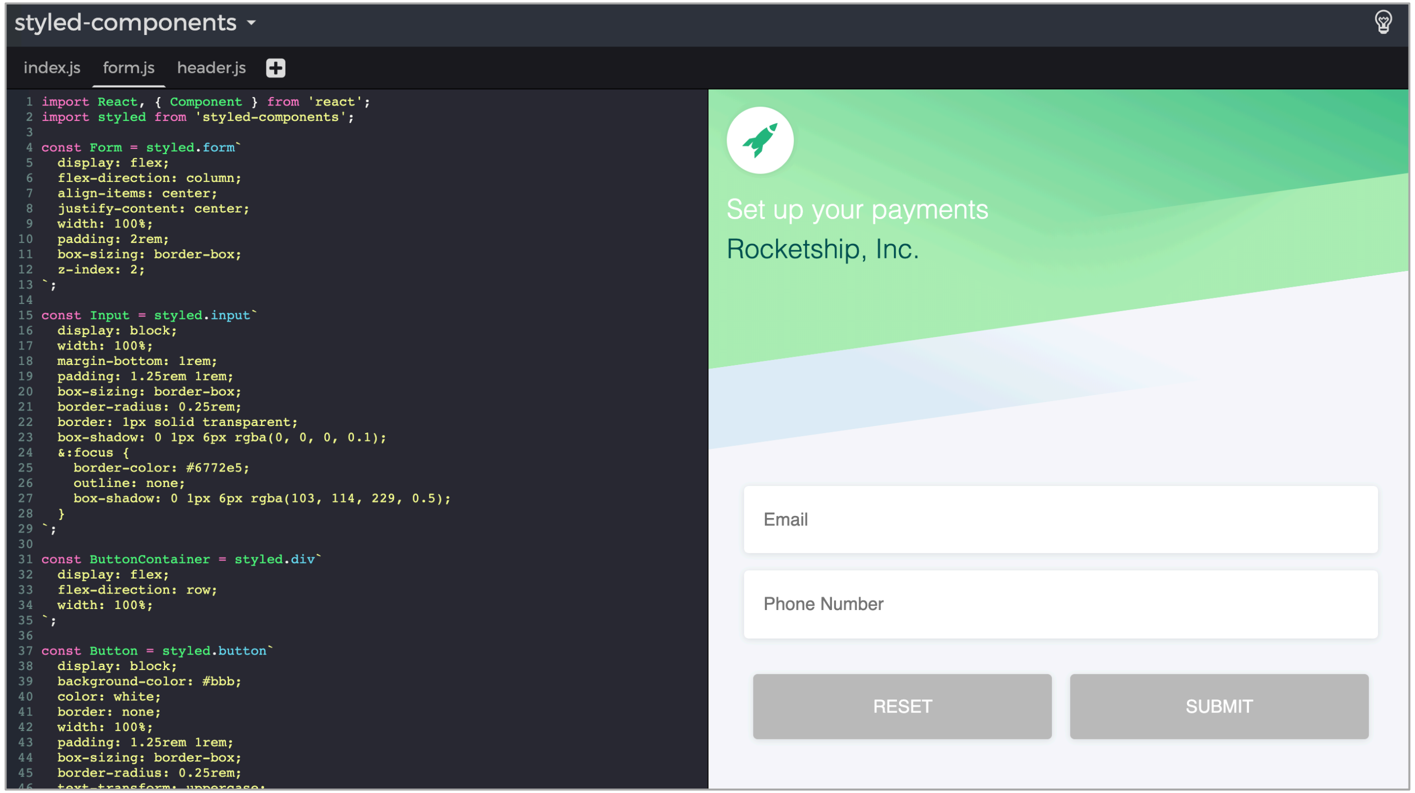
- Understandable structure that unifies markup and styles
- Specific and component focused classes

But...

- In practice often like component scope, but not truly scoped to component
- Not enforced at the tooling level
  - (sass-lint can help a bit here)
- Requires discipline across dev team
- Naming is still hard

CSS-in-JS can provide **component scope.**

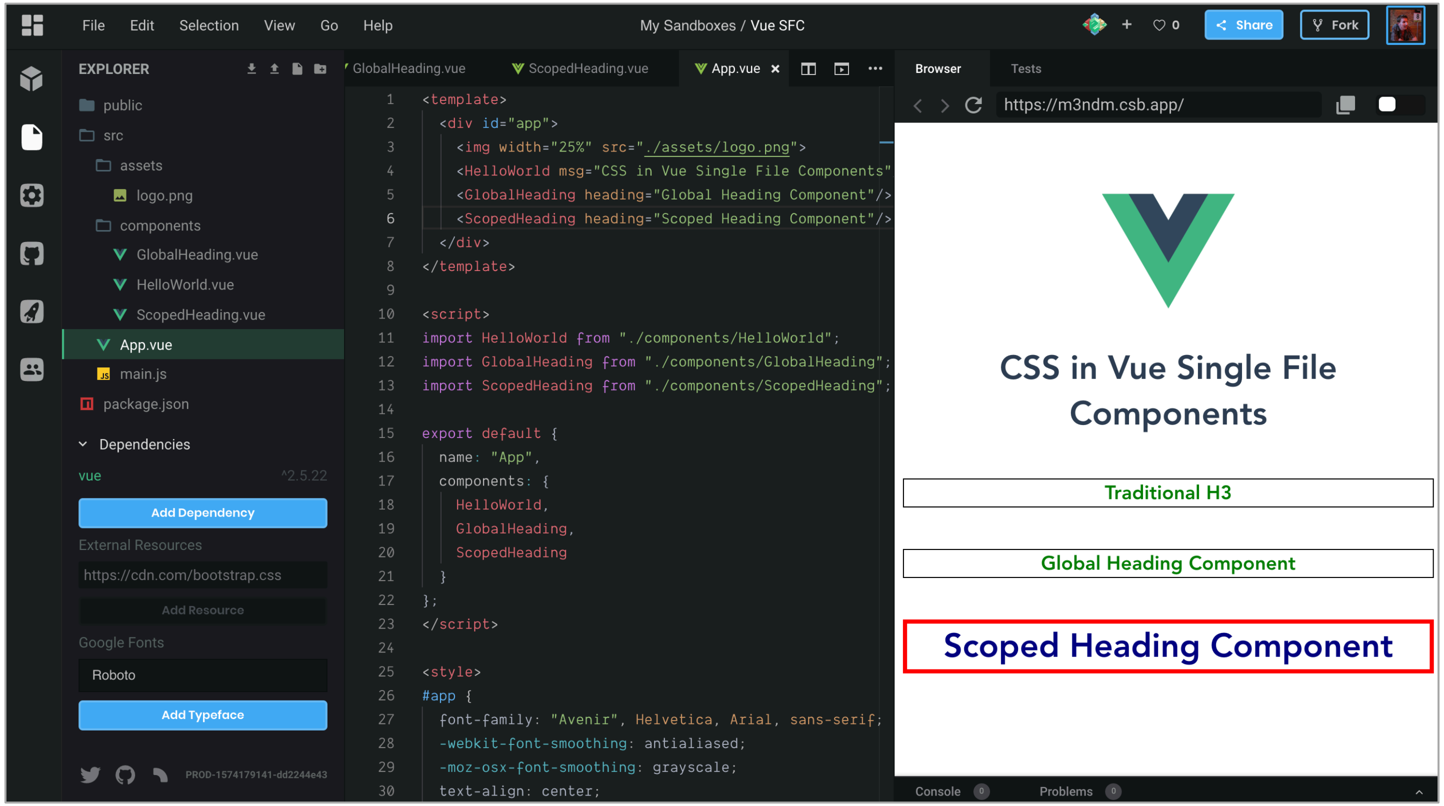
COMPONENT SCOPE - STYLED COMPONENTS



```
import React, { Component } from 'react';
import styled from 'styled-components';

const Form = styled.form`
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  width: 100%;
  padding: 2rem;
  box-sizing: border-box;
  z-index: 2;
`;

const Input = styled.input`
  display: block;
  width: 100%;
  margin-bottom: 1rem;
  padding: 1.25rem 1rem;
  box-sizing: border-box;
  border-radius: 0.25rem;
  border: 1px solid transparent;
  box-shadow: 0 1px 6px rgba(0, 0, 0, 0.1);
  &:focus {
    border-color: #6772e5;
    outline: none;
    box-shadow: 0 1px 6px rgba(103, 114, 229, 0.5);
  }
`;

const ButtonContainer = styled.div`
  display: flex;
  flex-direction: row;
  width: 100%;
`;

const Button = styled.button`
  display: block;
  background-color: #bbb;
  color: white;
  border: none;
  width: 100%;
  padding: 1.25rem 1rem;
  box-sizing: border-box;
  border-radius: 0.25rem;
  text-transform: uppercase;
```

Source: https://www.cssinjsplayground.com/

VUE SINGLE FILE COMPONENTS



```
<template>
  <div id="app">
    <img width="25%" src="./assets/logo.png">
    <HelloWorld msg="CSS in Vue Single File Components"
    <GlobalHeading heading="Global Heading Component"/>
    <ScopedHeading heading="Scoped Heading Component"/>
  </div>
</template>

<script>
import HelloWorld from "./components/HelloWorld";
import GlobalHeading from "./components/GlobalHeading";
import ScopedHeading from "./components/ScopedHeading";

export default {
  name: "App",
  components: {
    HelloWorld,
    GlobalHeading,
    ScopedHeading
  }
};
</script>

<style>
#app {
  font-family: "Avenir", Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
```

CSS in Vue Single File Components

Traditional H3

Global Heading Component

Scoped Heading Component

Source: https://codesandbox.io/s/vue-sfc-css-scope-m3ndm

You probably know what slide is coming next...

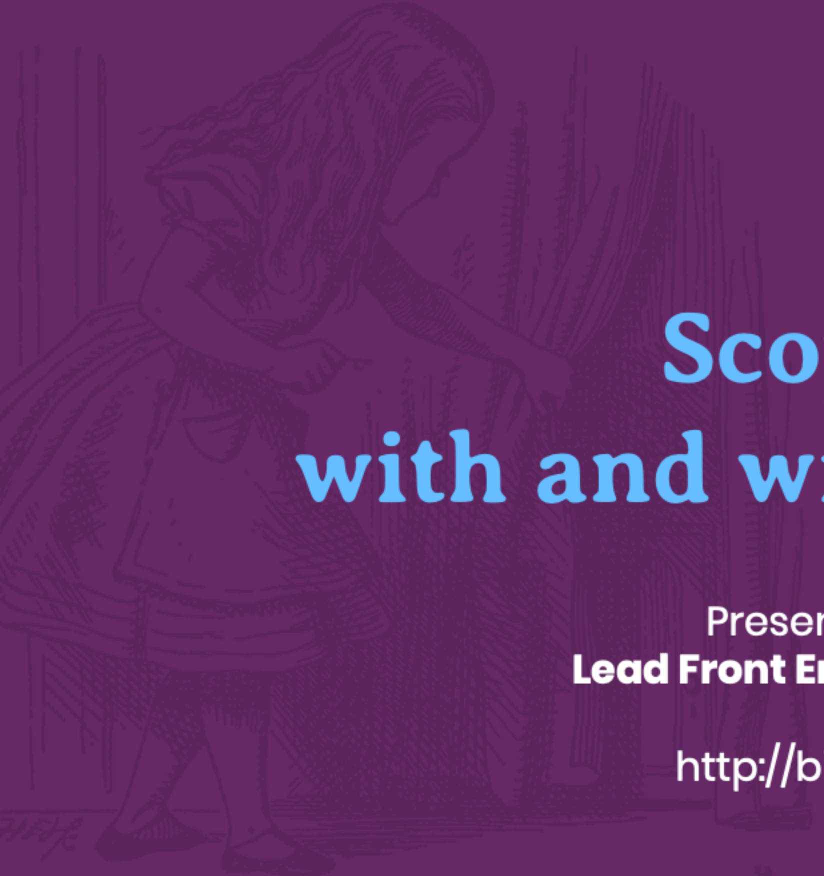
Why don't we have both?

Thank You!

Brian Perry
Lead Front End Developer

Email: brian.perry@bounteous.com

March 19, 2020



Scope in CSS with and without JavaScript

Presented by: **Brian Perry**
Lead Front End Developer - Bounteous

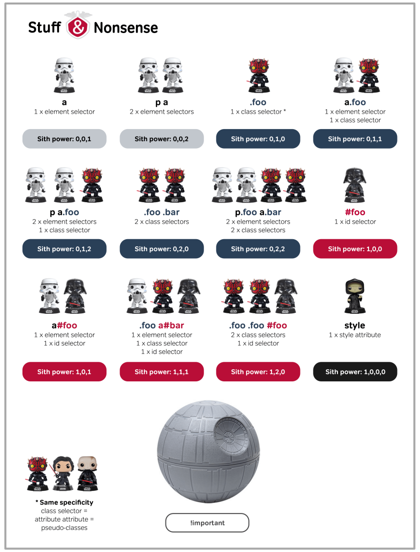
http://bit.ly/scope-midcamp

CSS has **global scope.**

HEY, REMEMBER CSS AND BEM?

Cascade

Specificity



Inheritance

BEM

Modifier:
.face__eyes.face__eyes--open
.face__mouth.face__mouth--open

But can ‘regular’ CSS have **component scope?**

But can ‘regular’ CSS have **component scope?**

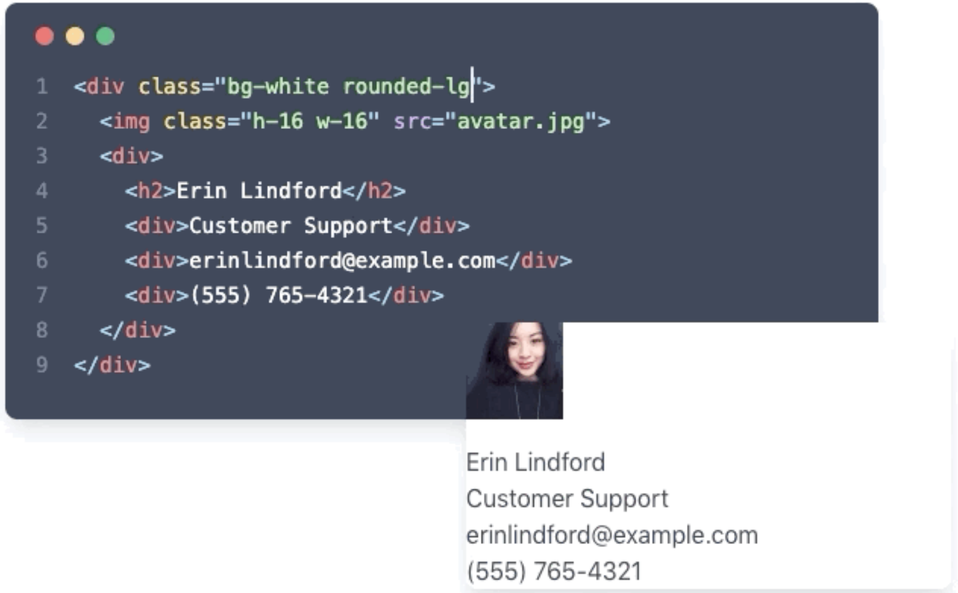
Not really, but…

ATOMIC (FUNCTIONAL) CSS IS A THING

- Small single purpose classes named based on visual function
- Immutable CSS
  - Classes don't change
  - Visual control shifts from CSS to markup
- Approaches:
  - Static – pre-existing set of utility classes
  - Programmatic – CSS generated based on markup

TAILWIND

A utility-first framework for rapidly building custom designs



```html
<div class="bg-white rounded-lg">
  <img class="h-16 w-16" src="avatar.jpg">
  <div>
    <h2>Erin Lindford</h2>
    <div>Customer Support</div>
    <div>erinlindford@example.com</div>
    <div>(555) 765-4321</div>
  </div>
</div>
```

Erin Lindford
Customer Support
erinlindford@example.com
(555) 765-4321

I'LL BE HONEST…

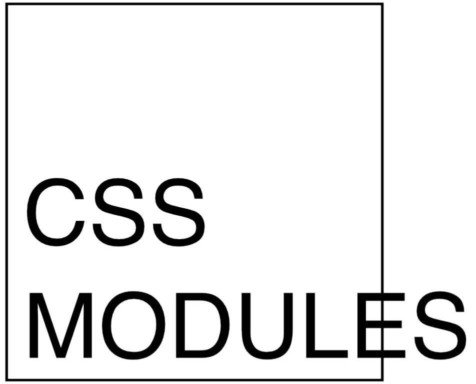
Some people love it, but to me it feels more like not using CSS…

CSS-in-JS can provide **component scope.**

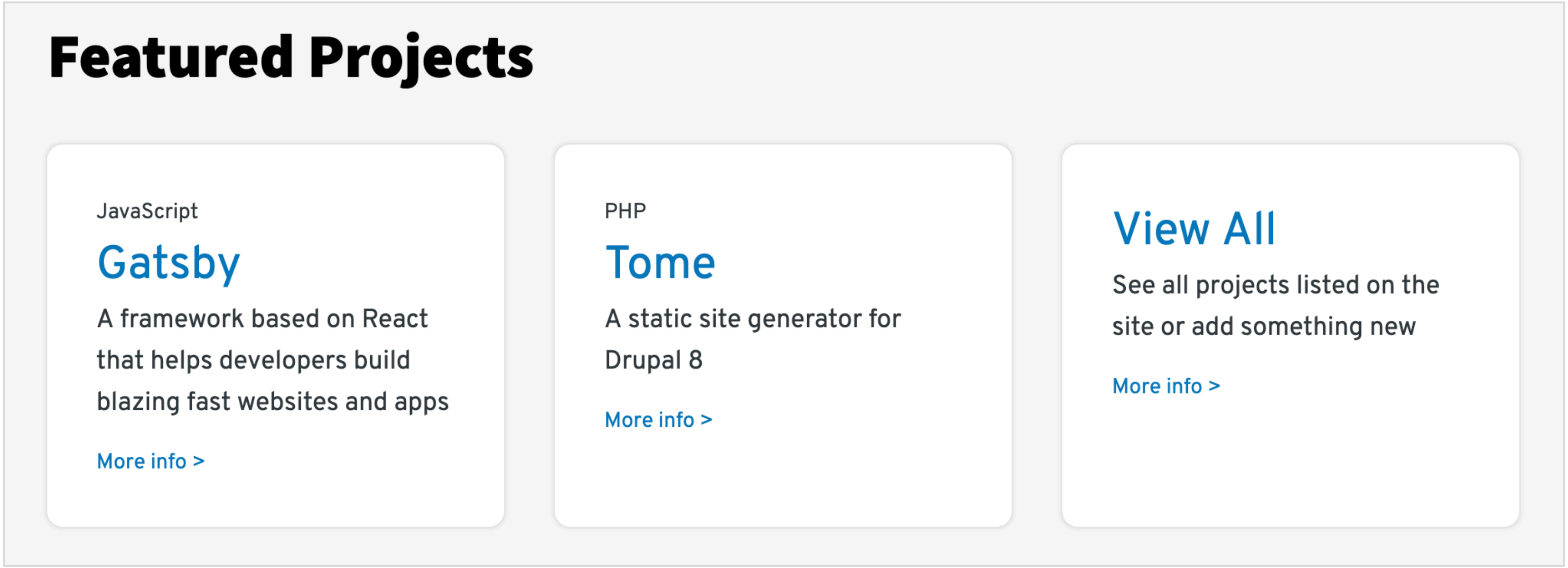
But could CSS-in-JS be **more like ‘regular’ CSS?**

CSS MODULES

- Not a package
- Handled by a build process (typically Webpack)
- Scopes styles locally by default
- Can play nice with Sass
- Ships with Create React App and Gatsby
- Has a super snarky logo



SCOPED CSS – PROJECT CARD

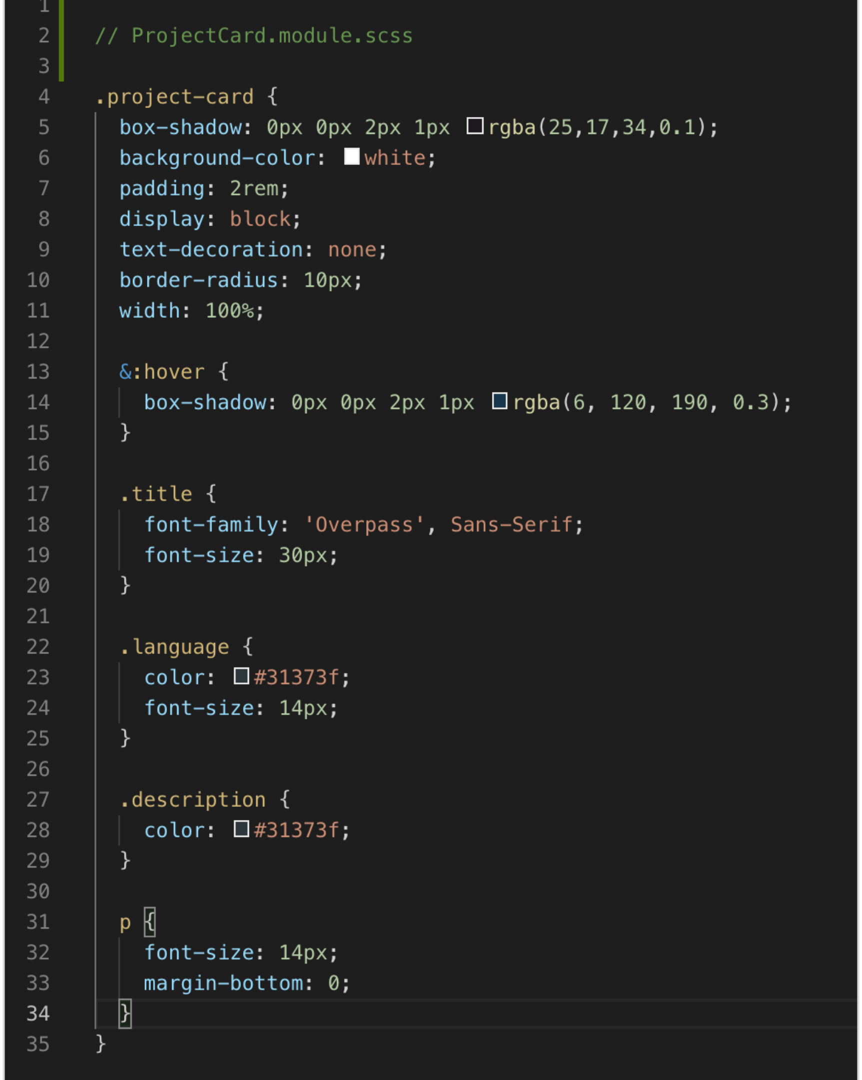


Featured Projects

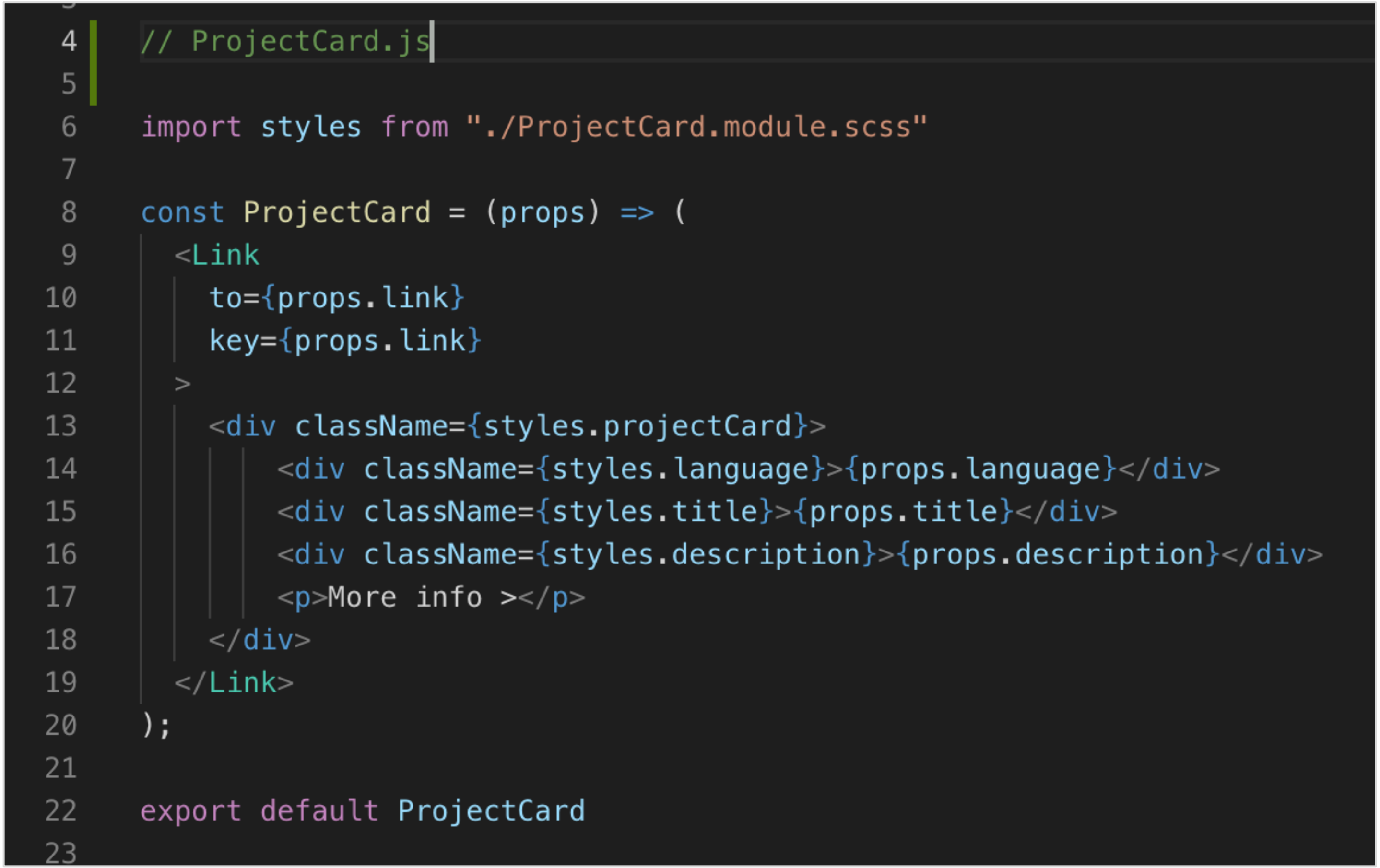
JavaScript
Gatsby
A framework based on React that helps developers build blazing fast websites and apps
More info >

PHP
Tome
A static site generator for Drupal 8
More info >

View All
See all projects listed on the site or add something new
More info >

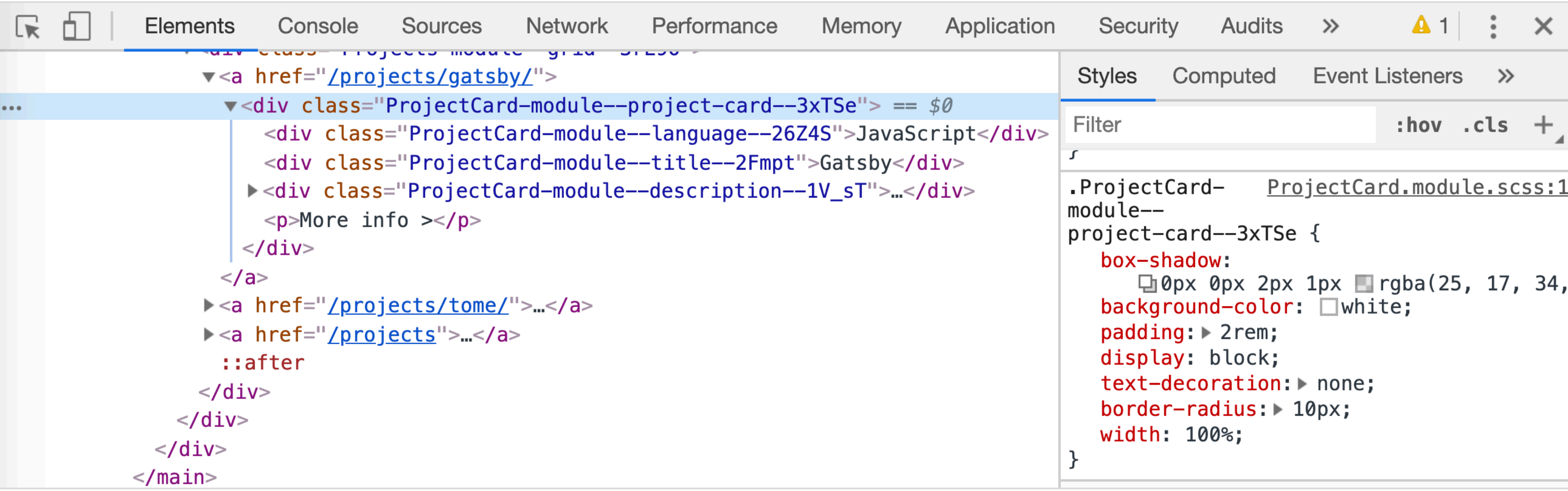


```scss

// ProjectCard.module.scss

.project-card {
  box-shadow: 0px 0px 2px 1px rgba(25,17,34,0.1);
  background-color: white;
  padding: 2rem;
  display: block;
  text-decoration: none;
  border-radius: 10px;
  width: 100%;

  &:hover {
    box-shadow: 0px 0px 2px 1px rgba(6, 120, 190, 0.3);
  }

  .title {
    font-family: 'Overpass', Sans-Serif;
    font-size: 30px;
  }

  .language {
    color: #31373f;
    font-size: 14px;
  }

  .description {
    color: #31373f;
  }

  p {
    font-size: 14px;
    margin-bottom: 0;
  }
}
```



```
// ProjectCard.js

import styles from "./ProjectCard.module.scss"

const ProjectCard = (props) => (
  <Link
    to={props.link}
    key={props.link}
  >
    <div className={styles.projectCard}>
        <div className={styles.language}>{props.language}</div>
        <div className={styles.title}>{props.title}</div>
        <div className={styles.description}>{props.description}</div>
        <p>More info ></p>
    </div>
  </Link>
);

export default ProjectCard
```

SCOPED CLASS NAMES

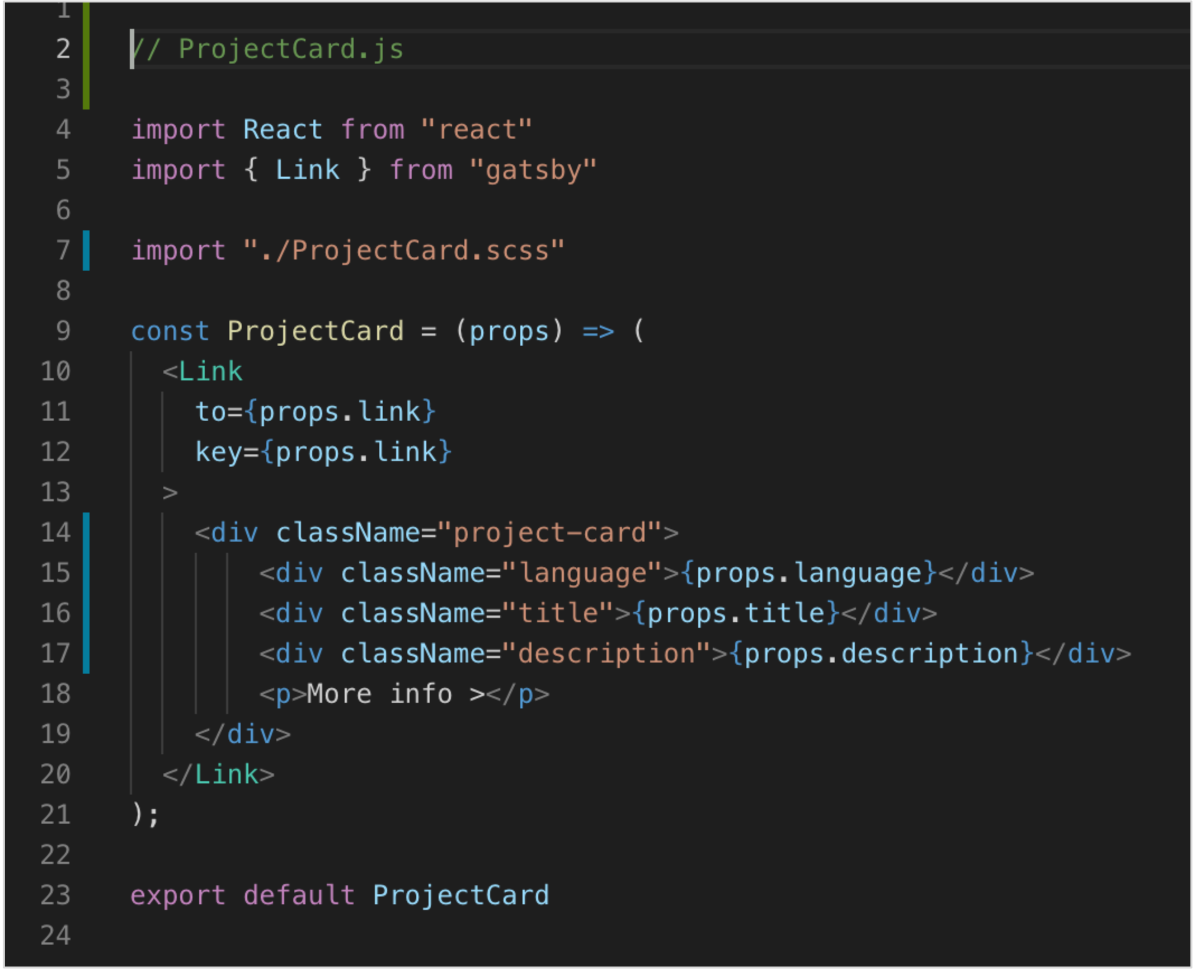


```
      ▼<a href="/projects/gatsby/">
...    ▼<div class="ProjectCard-module--project-card--3xTSe"> == $0
          <div class="ProjectCard-module--language--26Z4S">JavaScript</div>
          <div class="ProjectCard-module--title--2Fmpt">Gatsby</div>
         ▶<div class="ProjectCard-module--description--1V_sT">…</div>
          <p>More info ></p>
        </div>
      </a>
     ▶<a href="/projects/tome/">…</a>
     ▶<a href="/projects">…</a>
      ::after
    </div>
   </div>
  </div>
 </main>
```

```
.ProjectCard-        ProjectCard.module.scss:1
module--
project-card--3xTSe {
   box-shadow:
      0px 0px 2px 1px rgba(25, 17, 34,
   background-color: white;
   padding: ▶ 2rem;
   display: block;
   text-decoration: ▶ none;
   border-radius: ▶ 10px;
   width: 100%;
}
```

Why don't we have both?

GLOBAL CSS



```
// ProjectCard.js

import React from "react"
import { Link } from "gatsby"

import "./ProjectCard.scss"

const ProjectCard = (props) => (
  <Link
    to={props.link}
    key={props.link}
  >
    <div className="project-card">
        <div className="language">{props.language}</div>
        <div className="title">{props.title}</div>
        <div className="description">{props.description}</div>
        <p>More info ></p>
    </div>
  </Link>
);

export default ProjectCard
```

bounteous

STYLES SHARED BY MULTIPLE TEMPLATING ENGINES

Drupal

- Twig
- Unique styles and components

Combined

- Global styles that apply to both.
- Few (if any) shared components

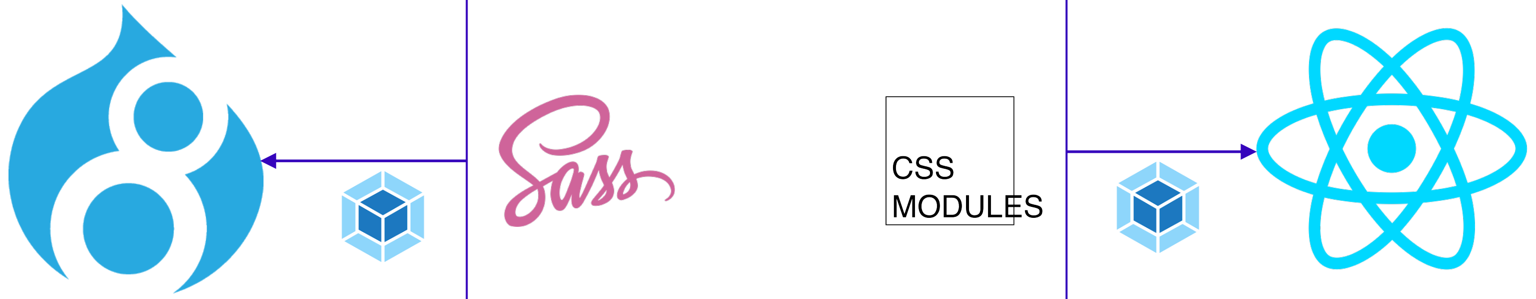
React

- JSX
- Unique styles and components

STYLES SHARED BY MULTIPLE TEMPLATING ENGINES

Drupal

React



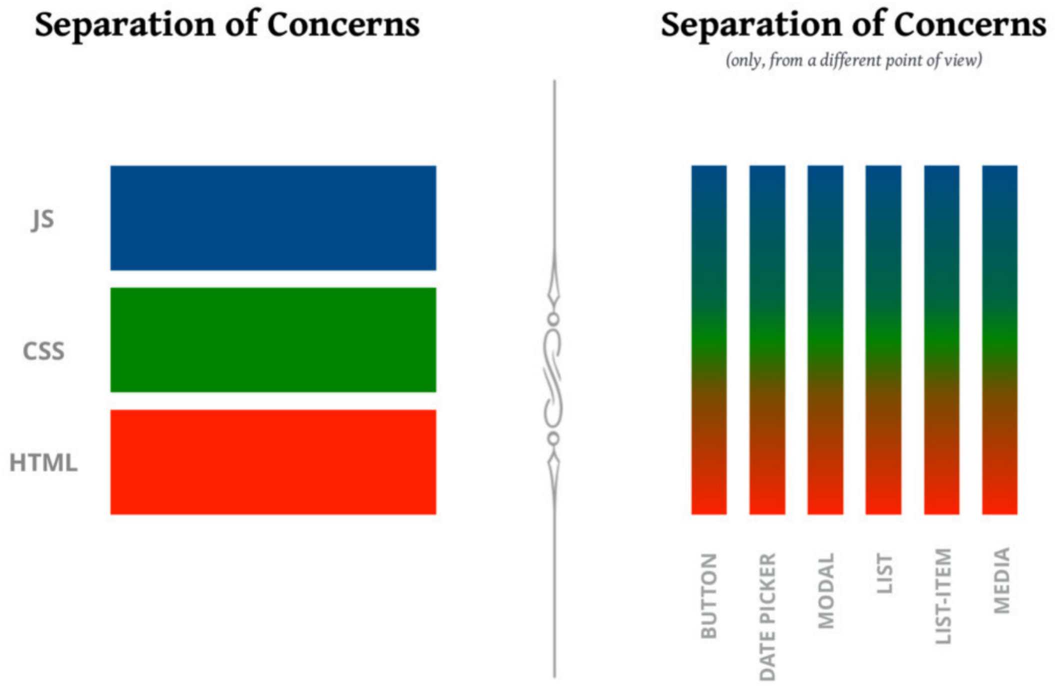
- Global CSS file - combines design system partials and Drupal specific partials

- Projects consume partials as needed

- Imported in JS
- Mix of global and component scope
- Also project specific CSS

We **can** have both!

SEPARATION OF CONCERNS IS IN THE EYE OF THE BEHOLDER



Source: https://speakerdeck.com/didoo/let-there-be-peace-on-css?slide=62

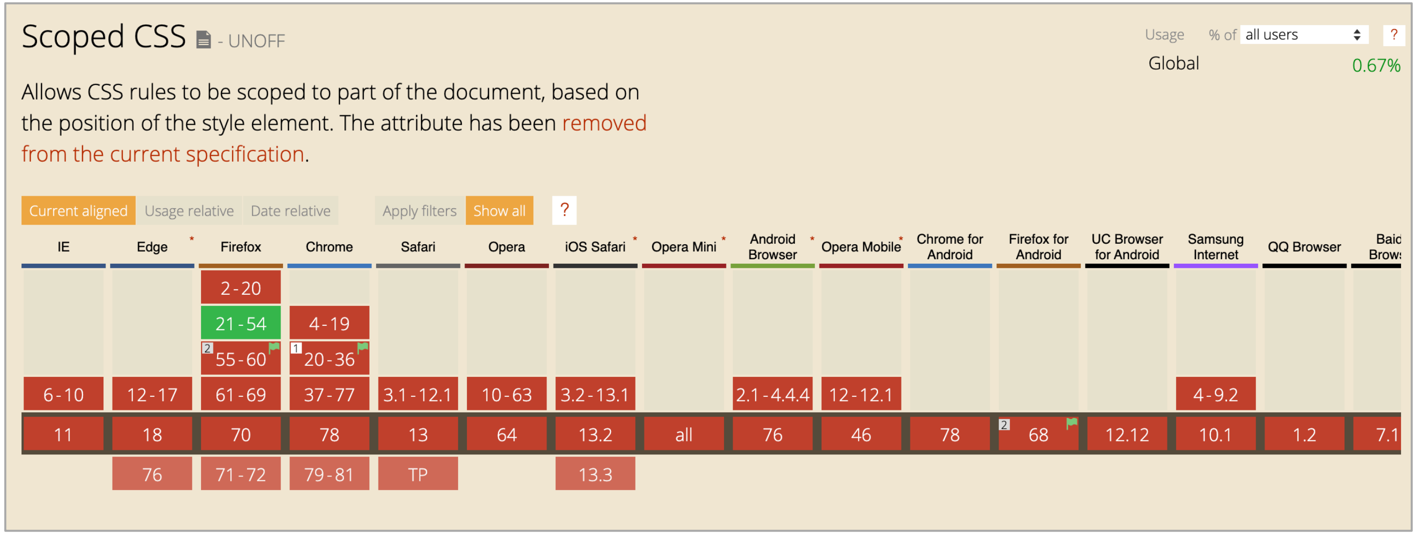
But could we have component scope with **less JS?**

POPULAR EXTENSIONS BECOME OFFICIAL

- CSS Preprocessors influence CSS
  - CSS Custom Properties
  - CSS Nesting
- Popular Drupal contrib projects make their way into core (I said Drupal, aren’t you proud?)
  - JSON:API
  - Panels Ecosystem -> Layout Builder
- Babel lets you develop with cutting edge JS today
  - Transpile today, browser native tomorrow.

COULD WE SEE THE SAME FOR CSS SCOPE?

- HTML Scoped Attribute… kind of didn't happen.
  - <style scoped>



Scoped CSS - UNOFF

Usage % of all users ?

Global 0.67%

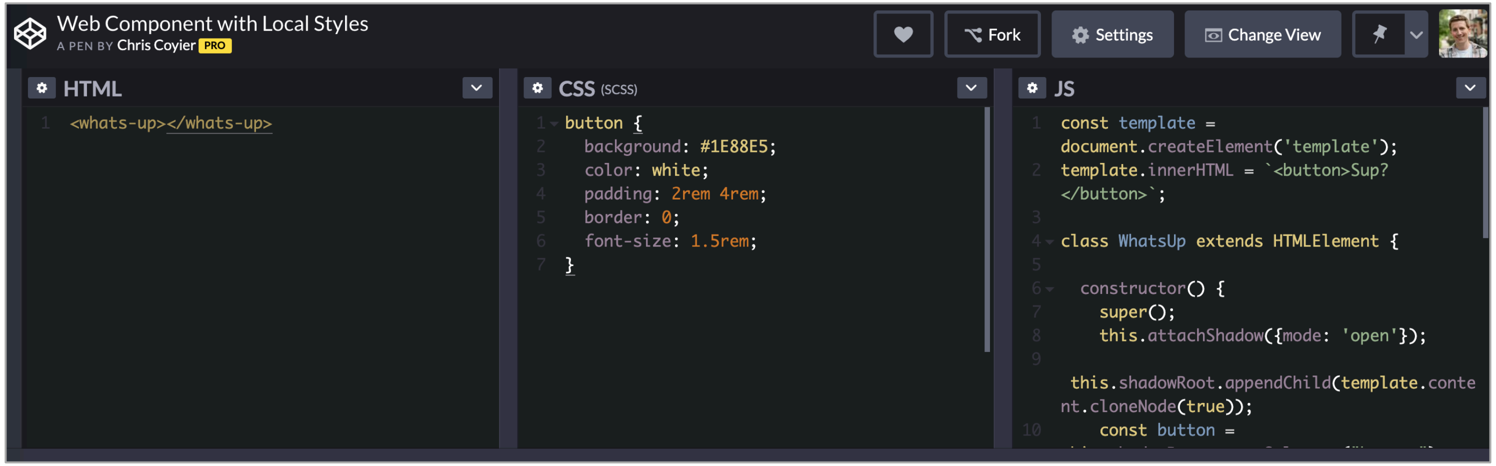
Allows CSS rules to be scoped to part of the document, based on the position of the style element. The attribute has been removed from the current specification.

Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baid Brows
		2-20													
		21-54	4-19												
		[2] 55-60	[1] 20-36												
6-10	12-17	61-69	37-77	3.1-12.1	10-63	3.2-13.1		2.1-4.4.4	12-12.1				4-9.2		
11	18	70	78	13	64	13.2	all	76	46	78	[2] 68	12.12	10.1	1.2	7.1
	76	71-72	79-81	TP		13.3									

COULD WE SEE THE SAME FOR CSS SCOPE?

- Web Components / Shadow DOM as a solution?
  - Shadow DOM styles scoped locally by default.
  - Approaches also exist to use or selectively override global styles.
  - Still requires writing a decent amount of JS.



HTML

```html
<whats-up></whats-up>
```

CSS (SCSS)

```scss
button {
  background: #1E88E5;
  color: white;
  padding: 2rem 4rem;
  border: 0;
  font-size: 1.5rem;
}
```

JS

```js
const template =
document.createElement('template');
template.innerHTML = `<button>Sup?
</button>`;

class WhatsUp extends HTMLElement {

  constructor() {
    super();
    this.attachShadow({mode: 'open'});

 this.shadowRoot.appendChild(template.conte
nt.cloneNode(true));
    const button =
```

Impossible to see, the future is.

CONTRIBUTION DAY

Saturday 10am to 4pm

You don't have to know code to give back!

New Contributor training 10am to Noon
with **AmyJune Hineline** of Kanopi Studios

Thank You!

Brian Perry
Lead Front End Developer

Email: brian.perry@bounteous.com