Do You Still Need Sass?

Florida Drupal Camp 2026

PRESENTED BY

Aubrey Sambor

Lead Front End Developer

A little about me

Aubrey Sambor

Lead Front End Developer

Loves CSS, knitting, beer, and coffee

Looking for work!

Mastodon
https://labyrinth.social/@starshaped

Website
https://aubreysambor.com

LinkedIn
https://www.linkedin.com/in/aubreysambor

Drupal.org
https://drupal.org/u/starshaped

What I'll cover

- An overview of native CSS features that can replace Sass functionality
  - AND new ones that aren't yet supported by all major browsers!
- Using PostCSS to implement up and coming CSS features that aren't supported yet in all browsers
- Ways to add native CSS to your Sass code to start using modern CSS!
- *Should* you still use Sass?

What can native CSS do today?

Custom properties

- Available in browsers since 2016, can be used in place of Sass variables `--my-nifty-variable, var(--my-nifty-variable)`
- Since they aren't compiled into CSS like Sass variables are, they can be changed on the fly using Javascript
- Can't be used as a media query variable, but we'll talk about this later!

Math in native CSS

Yes, CSS has the ability to do math!

- `calc()`: available since 2012
- `min()`, `max()`, `clamp()`: available in all browsers since 2020
  - `min()`: Takes the minimum value of a comma separated list of numbers
  - `max()`: Takes the maximum value of a comma separated list of numbers
  - `clamp()`: Clamps a value within a defined minimum and maximum range
- Trigonometric functions arrived in native CSS in 2023. `sin()`, `cos()`, `tan()`, and others.
- More math functions on MDN

Color functions

color-mix()

- Supported in all major browsers, `color-mix()` can be used in place of Sass's `mix()` function.
  - Percentage of the mix can be added to both values in `color-mix()`, while Sass `mix()` only takes one percentage value
- `color-mix(in srgb, pink 85%, white)`
- Can't use a CSS custom property in Sass color functions, as Sass variables compile at runtime while CSS custom properties compile when used

Let's take a look at how this works in native CSS vs Sass!

Relative color syntax

- Allows colors to be changed relative to another color
- Takes an origin color and splits it into 3 color channels
- `background-color: oklch(from purple l c h);`
  - You can manipulate each of these channels to create a new color!

Nesting

- Supported in all major browsers since 2024
- Works identical to Sass, except:
  - Sass and CSS interpret nested code in different ways
  - Concatenation is not possible like it is in Sass
  - You can't use a variable to hold onto the `&` selector to use later

Nesting in Sass and native CSS

```
.header, .footer {
  h2 {
    ...
  }
}

/* In Sass... */

.header h2,
.footer h2 {
  ...
}

/* In native CSS... */

:is(.header, .footer) h2 {
  ...
}
```

Concatenation in Sass

```
// This is not possible in native CSS!

.main-content {
  &__header {
    ...
  }

  &__footer {
    ...
  }
}
```

Saving the ampersand for later

```
// This is not possible in native CSS!

.main-content {
  $self: &;

  &--narrow {
    ...

    ${self}__header {
      ...
    }
  }
}

// Compiles to
.main-content {...}
.main-content--narrow {...}
.main-content--narrow .main-content__header {...}
```

CSS if() and @function, the new kids on the block!

if()

- Finally, the ability to add if statements to CSS!
- Only supported in Chromium browsers (Chrome, Edge, Opera, and others)
- Currently only supports 3 queries:
  - Media queries (`media()`)
  - Feature queries (`supports()`)
  - Style queries (`style()`)

if() example

```
.container {
  display: grid;
  grid-template-columns:
    if(
      media(width > 1440px): repeat(4, 1fr);
      media(width > 1024px): repeat(3, 1fr);
      media(width > 768px): repeat(2, 1fr);
      else: 1fr;
    );
}
```

@function

- Now you can add basic functions to CSS!
- Also only supported in Chromium browsers currently
- You can specify data types and return types
- No early return like in PHP and JavaScript functions

Pixels to REMs using @function

```
@function --rem(--px <number>) returns <length> {
  result: calc(var(--px) / 16)rem;
}

...

h2 {
  /* renders as 1.5rem */
  font-size: --rem(24);
}
```

What if I want Sass-like fuctionality on my site?

Enter PostCSS!

What is PostCSS?

- A NodeJS tool to transpile CSS using various plugins
- Some plugins replicate what is coming from the CSSWG (CSS Working Group), acting as a polyfill of sorts until all browsers support the feature
- Other plugins replicate Sass features or other features that may never be added to the CSS specification
- Others add nice to haves to CSS

PostCSS plugins with future CSS syntax

- postcss-custom-media: defines @custom-media following the Custom Media specification
- postcss-if-function: Transforms the CSS `if()` function into native CSS when the browser doesn't support it
  - Only works with `media()` and `supports()`, `style()` not yet supported
- postcss-contrast-color-function: follows the CSS Color Module Level 5 specification
- postcss-preset-env: collection of plugins all with future CSS syntax!

PostCSS plugins that replicate Sass functionality

- postcss-mixins: plugin that replicates mixin functionality in Sass
- postcss-map-get: replicates Sass's map-get functionality
- postcss-for: Adds `for` loops to code
- List of PostCSS plugins from the PostCSS website

PostCSS plugins with useful functionality

- cssnano: CSS minifier built on top of PostCSS
- postcss-pxtorem: converts px units to rem units
- postcss-import: transforms import rules by inlining styles, similar to Sass partials

PostCSS in action!

PostCSS downsides

- You might get used to plugins that don't follow CSS specifications, and once those specs get added to CSS, you'll have to update your code to follow the 'real' specification.
- Having too many plugins might affect performance
- Setup can be confusing depending on the technology you use
- You might not want any tooling at all on your site

Okay, this is great. What if I still want to use Sass, though?

You can use Sass AND native CSS together!

Using Sass and native CSS

- You can use custom properties with Sass variables
- You might want to use CSS custom properties for most of your code, but need to switch to Sass to use mixins and maps
- Sass has its own version of `calc()` which works with Sass variables
- As of dart-sass 1.40.0, you don't need to interpolate your Sass variables to use within `calc()`!
- You can use `calc()` in Sass to get around having to use `math.div` for the division operator

So, SHOULD you still use Sass?

It depends!

Reasons to still use Sass today

- Sass isn't going away anytime soon, so use it as much as you like!
- You don't feel like native CSS has quite caught up to Sass yet
- You make use of complex mixins, functions, and maps
- You still need to support legacy browser versions
- Updating your codebase to remove Sass would be too time consuming
- If you want to!

Wrapping up...

Wrapping up...

- **It's a great time to be a front end developer** with so many new additions to CSS:
  - Custom properties, math functions, color functions, nesting, if(), @function
- **PostCSS can enhance native CSS** by:
  - Adding CSS that might not be supported by all modern browsers
  - Adding functionality Sass supports today, such as mixins and functions
  - Adding useful functionality such as converting px to rem and CSS minification
- **You can use Sass with CSS** if you want to gradually switch your codebase from one to the other, but...
- You should still use Sass... **if you want to!**

Further reading

- Is it time to drop Sass?
- Do not drop Sass for CSS
- Going back to CSS-only after years of SCSS
- Sass Features in CSS
- How to Use PostCSS as a Configurable Alternative to Sass
- The gotchas of CSS Nesting
- Mixing Colors with CSS
- CSS Nesting, the :is() pseudo-class, and a guide to panicking about Sass
- CSS @function: Dynamic logic without Sass or JavaScript
- What You Need to Know about Modern CSS (2025 Edition)
- Collection of my examples on CodePen

A little about me

Aubrey Sambor

Lead Front End Developer

Loves CSS, knitting, beer, and coffee

Looking for work!

Mastodon
https://labyrinth.social/@starshaped

Website
https://aubreysambor.com

LinkedIn
https://www.linkedin.com/in/aubreysambor

Drupal.org
https://drupal.org/u/starshaped

Thank you!