CSS Variables

This lesson introduces CSS variables and how we'll use them in our themes.

WE'LL COVER THE FOLLOWING ∧

- Inheritance
- Dynamically creating variables

Note: If you need a quick refresher about CSS variables, check this shot.

Inheritance

CSS variables (also called custom properties) have one special feature that makes them perfect for our theming needs. CSS variables are *inherited*.

Consider the following example:

HTML   CSS   Output

```html
1  <html>                          html
2  <head>
3  </head>
4  <body>
5     <div class="outer">
6        <div class="foo">
7
8        <div class="inner
9           <div class="foo
10       </div>
11    </div>
12 </body>
13 </html>
```

```css
1  .outer,                         css
2  .inner {
3     margin: 5px;
4     border: 1px solid □#333;
5     border-radius: 3px;
6  }
7
8  .outer {
9     --some-color: ■red;
10 }
11
12 .inner {
13    --some-color: ■blue;
14 }
15
16 .foo {
17    color: var(--some-color);
18 }
```

```
                                   output
Foo

Foo
```

When `var(--some-color)` is being evaluated, the hierarchical HTML tree is traversed until a definition of `--some-color` is found. This is great, as it means we can scope CSS variables to a certain section of our app, and also override them in an inner component if we need to.

This is all that we need for now when it comes to CSS variables. We'll be using them more when we define our themes later. What we call a **theme prop** is actually a CSS variable on the theme definition, and we'll be using those terms interchangeably.

Dynamically creating variables

Lastly, let's talk a bit about how we'll create those variables.

Most of our props will be computed on the go according to theme values. In SCSS, creating a CSS variable that we compute from some other SCSS value is super easy:

```scss
// Assume we get this from the theme
$color: red;

body {
    --color: #{$color};
}
```

With the power of SCSS, we'll be able to do all kinds of computations, and then create CSS variables with the results for easy consumption. Once you've created a CSS variable, the next step is to learn how to apply them conditionally. Move to the next lesson to learn this.