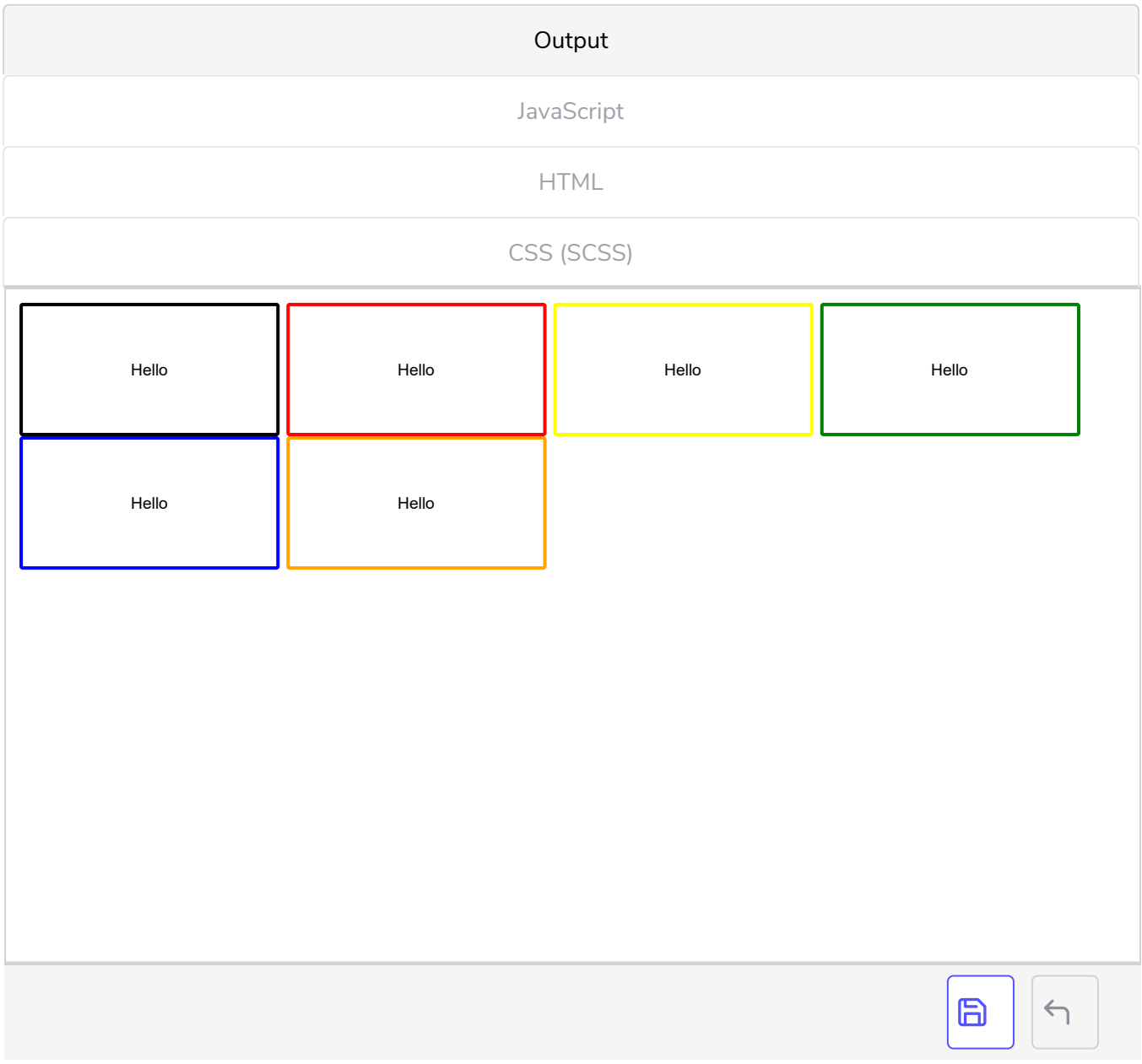
Creating Component Variations using CSS Variables

Consider the case where you need to build two or more different buttons. Same common base styles, just a bit of difference between the buttons.

Below is an example, and do not forget to hover over the buttons too.



In this case, the properties that differ are the `background-color` and `border-color` of the variants.

So, how would you do this?

Here's the typical solution.

Create a base class, say `.btn` and add the variant classes. Here's an example markup:

```
<button class="btn">Hello</button>
<button class="btn red">Hello</button>
```

`.btn` would contain the base styles on the button. For example:

```
.btn {
   padding: 2rem 4rem;
   border: 2px solid black;
   background: transparent;
   font-size: 0.6em;
   border-radius: 2px;
 }

 /*on hover */
 .btn:hover {
   cursor: pointer;
   background: black;
   color: white;
 }
```

So, where does the variant come in?

Here:

```
/* variations */

.btn.red {
  border-color: red
 }
.btn.red:hover {
  background: red
}
```

You see how we are duplicating code here and there? This is good, but we could make it better with CSS variables.

What's the first step?

Substitute the varying colors with CSS variables, and don't forget to add default values for the variables!

```
.btn {
    padding: 2rem 4rem;
    border: 2px solid var(--color, black);
    background: transparent;
    font-size: 0.6em;
    border-radius: 2px;
  }

  /*on hover*/
  .btn:hover {
   cursor: pointer;
    background: var(--color, black);
    color: white;
  }
```

When you do this: `background: var(--color, black)` you're saying, set the background to the value of the variable `--color` . However, if the variable doesn't exist, use the **default value** of `black`

Here's the good part.

With the variants, you just supply the new value of the CSS variable like so:

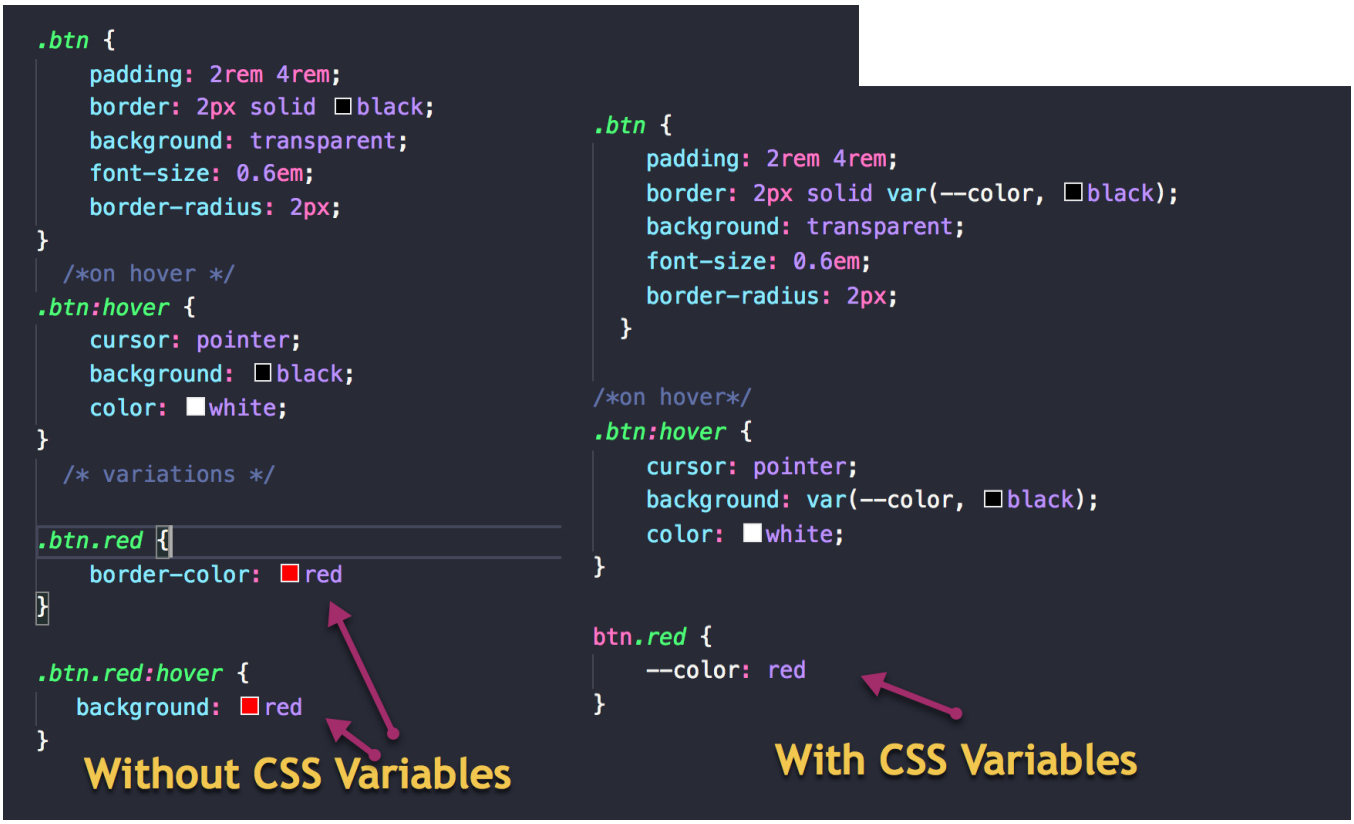
```
btn.red {
    --color: red
 }
```

That's all. Now when the `.red` class is used, the browser notes the different `--`

`color` variable value, and immediately updates the appearance of the button.

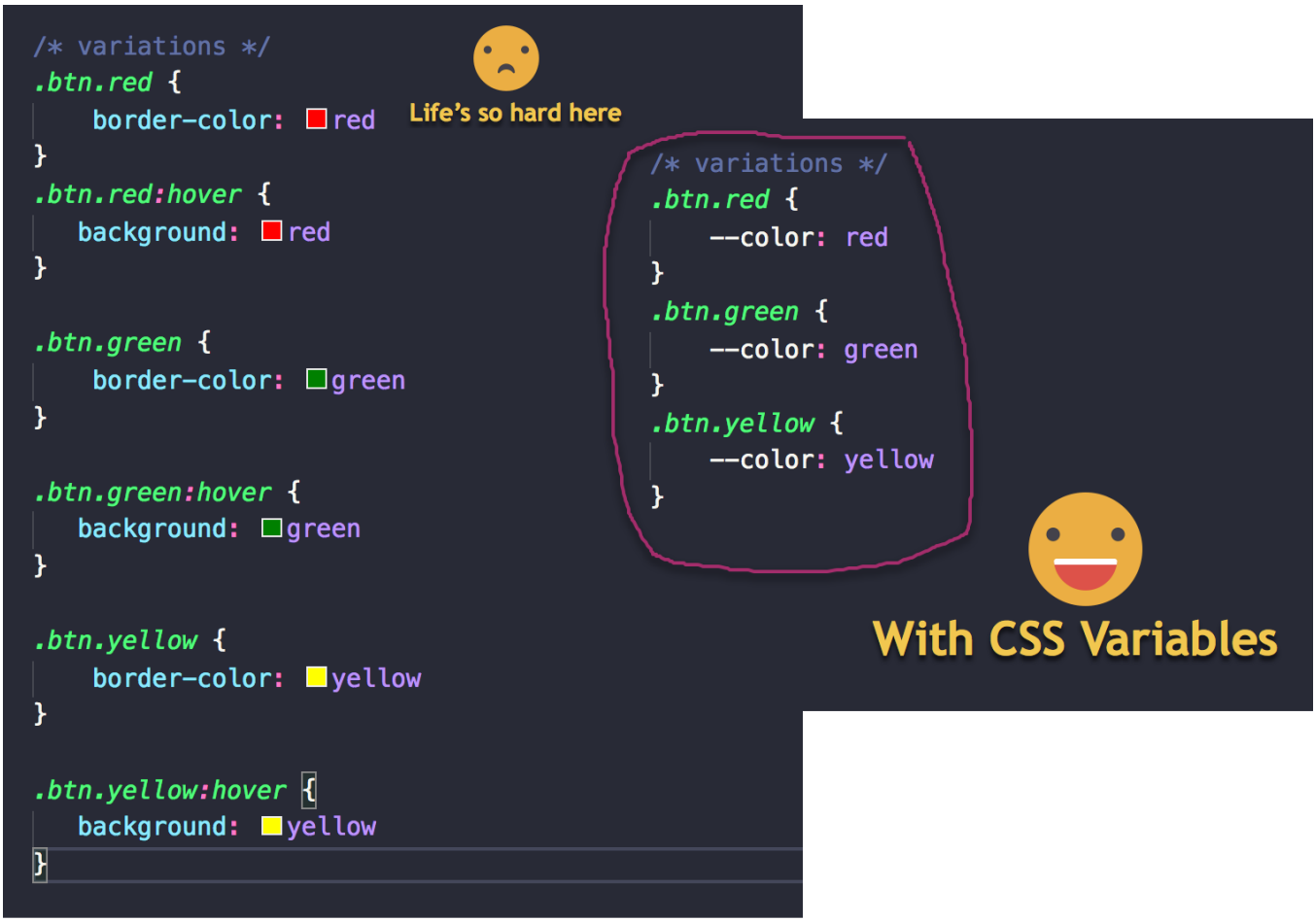
This is really good if you spend a lot of time building reusable components.

Here's a side by side difference



without css variables VS with css variables

Oh, and if you had more variants, you just saved yourself a lot of extra typing as seen below:



a lot more typing to do without CSS variables

As you can see, there's a lot of ease you get with crafting components with CSS variables.

Easy and fun, right?

See you in the next lesson.