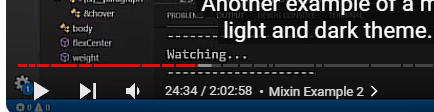
8 Switching Between Themes

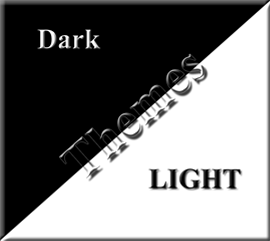
<https://www.youtube.com/watch?v=_a5j7KoflTs>



Contents

[Creating a theme @mixin 1](#_Toc163576997)

[Using a Boolean value for the theme mixin 1](#_Toc163576998)



Turn on [Watch Sass](2%20Creating%20the%20Project.docx)

In your main.scss file



# Creating a theme @mixin

We are going to be using a @mixin to accomplish this.

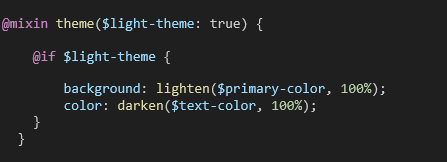
Add another @mixin right above your body rule, and below your other @mixin.

# Using a Boolean value for the theme mixin

**Write Boolean @mixin**

Write the definition above body rule in main. SCSS file. The two functions of lighten and darken are provided by SASS. This means that you can say their name, and they just work.

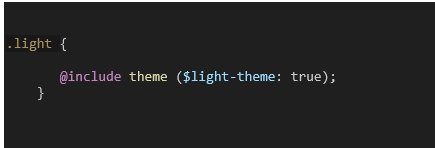
They take two arguments. The property, you want to either lighten or darken and then a percentage of HSL color format. This parameter will range from 0% to 100%. It is working with the third letter in HSL, which represents lightness. The first two letters represent Hue, and Saturation respectfully. But we are not using Hue and Saturation here.



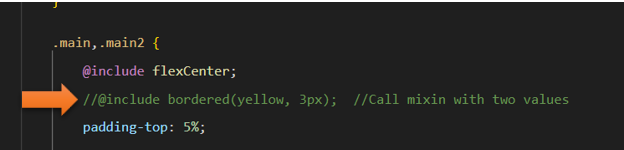
**Writing the light class to go into Boolean**

Under your main rule, write your light class and your dark class. look at how we went about writing the light theme, we used the @include, and then we passed in the variable of light-theme: true. How cool is that.

Oh, we had 3 curly braces closing out that rule above it, which was a nested rule for .main.



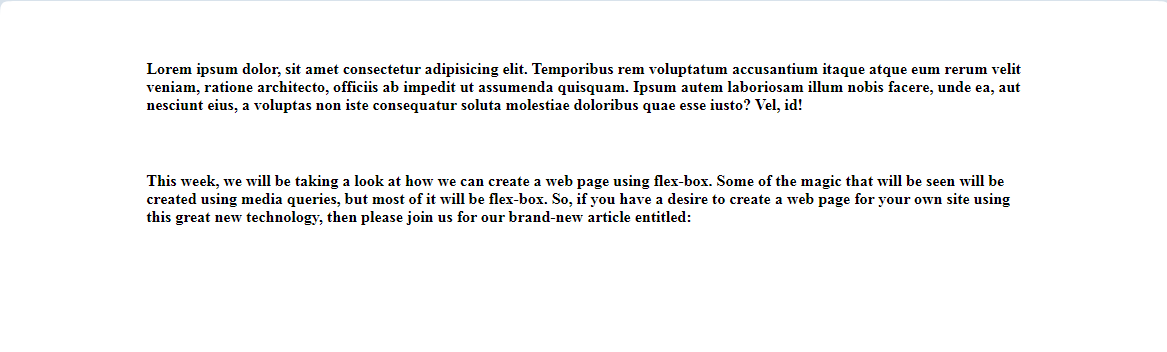
Remove your border from your .main rules. It is quite ugly.



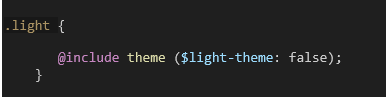
Go to your index.html to add the class to the body so we can use it.



And now your theme is light with dark text.



Now if you change that Boolean value to be false you will revert back to your dark theme. You make that change in the @include for the .light class



Go to your main.html to test it

