TASK

SCSS

Visit our website

Introduction

WELCOME TO THE SCSS TASK!

Now that you have been introduced to HTML and CSS, we are going to take a look at how to make your styling more efficient, maintainable and more easily changeable through the use of variables, nesting, partials, modules, mixins, inheritance, and operators.

WHAT IS SASS AND SCSS?

In its own words, Sass is "CSS with superpowers" (Sass, 2020). It is a way to add styling to your web pages that allows you to increase the complexity, but still make it easy to maintain. SASS and SCSS are basically different syntaxes for writing the same code. Because it is almost exactly the same as CSS syntax, we will be learning SCSS.

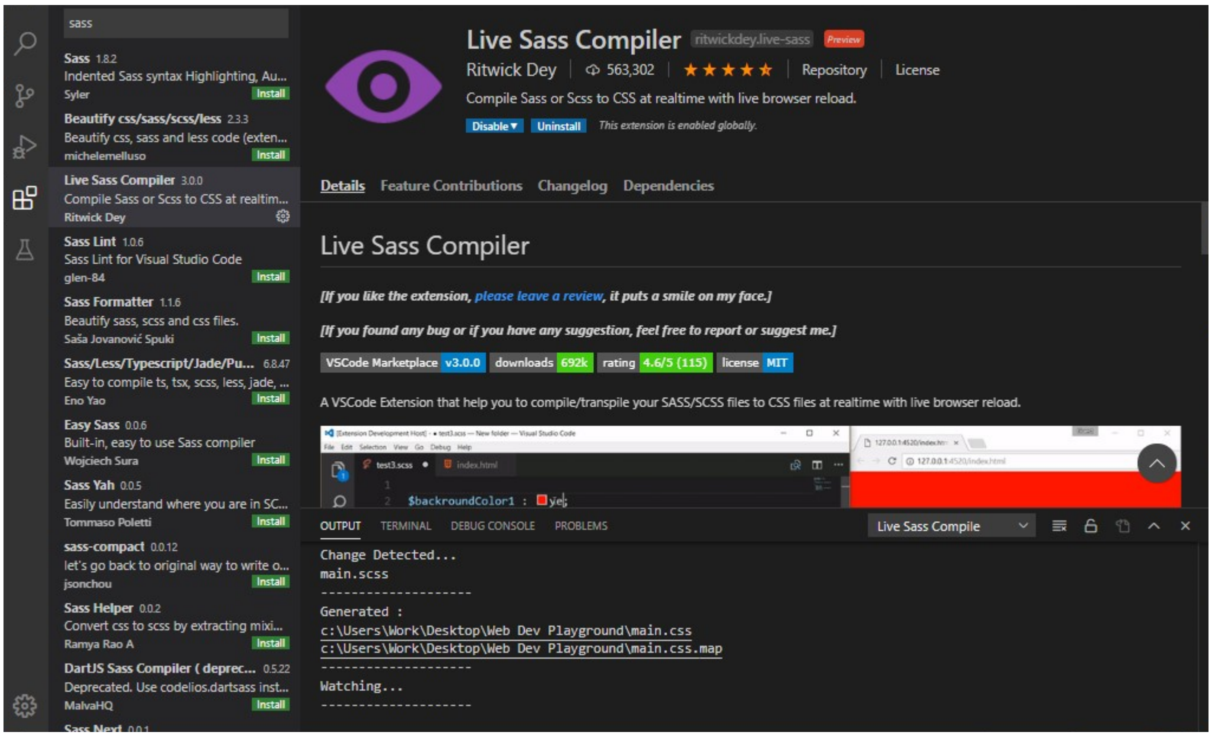
SASS/SCSS use added features to the CSS to add functionality. In this task, we will look at the following:

- Variables
- Nesting
- Partials
- Modules
- Mixins
- Inheritance
- Operators

INSTALLING SCSS

The file extension for SCSS files is **.scss**. However, a browser can only read **.css** files. So, to work with SCSS, you will need to download a compiler. In Visual Studio Code, go to the Extensions Marketplace and search for the "Live Sass Compiler" and install it. This will automatically compile your SCSS file into a **.css** file that you will then import into your HTML file. If you were to create a folder called **scss**, if you have a **.scss** file within this folder, clicking "watch sass" would generate a compiled CSS file within this **scss** folder.

Alternatively, if you are using Webstorm, you can follow the instructions **here** and **here**.


sass

Sass 1.8.2
Indented Sass syntax Highlighting, Au...
Syler — Install

Beautify css/sass/scss/less 2.3.3
Beautify css, sass and less code (exten...
michelemelluso — Install

Live Sass Compiler 3.0.0
Compile Sass or Scss to CSS at realtim...
Ritwick Dey

Sass Lint 1.0.6
Sass Lint for Visual Studio Code
glen-84 — Install

Sass Formatter 1.1.6
Beautify sass, scss and css files.
Saša Jovanović Spuki — Install

Sass/Less/Typescript/Jade/Pu... 6.8.47
Easy to compile ts, tsx, scss, less, jade, ...
Eno Yao — Install

Easy Sass 0.0.6
Built-in, easy to use Sass compiler
Wojciech Sura — Install

Sass Yah 0.0.5
Easily understand where you are in SC...
Tommaso Poletti — Install

sass-compact 0.0.12
let's go back to original way to write o...
jsonchou — Install

Sass Helper 0.0.2
Convert css to scss by extracting mixi...
Ramya Rao A — Install

DartJS Sass Compiler (deprec... 0.5.22
Deprecated. Use codelios.dartsass inst...
MalvaHQ — Install

Live Sass Compiler ritwickdey.live-sass Preview
Ritwick Dey | 563,302 | ★★★★★ | Repository | License
Compile Sass or Scss to CSS at realtime with live browser reload.
Disable ▾ Uninstall *This extension is enabled globally.*

Details Feature Contributions Changelog Dependencies

Live Sass Compiler

[If you like the extension, please leave a review, it puts a smile on my face.]

[If you found any bug or if you have any suggestion, feel free to report or suggest me.]

VSCode Marketplace v3.0.0 | downloads 692k | rating 4.6/5 (115) | license MIT

A VSCode Extension that help you to compile/transpile your SASS/SCSS files to CSS files at realtime with live browser reload.

```
$backroundColor1 : ye;
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS — Live Sass Compile

```
Change Detected...
main.scss
--------------------
Generated :
c:\Users\Work\Desktop\Web Dev Playground\main.css
c:\Users\Work\Desktop\Web Dev Playground\main.css.map
--------------------
Watching...
--------------------
```

HOW TO USE SCSS

To link your SCSS file to your HTML code, you are going to use the usual method by using a `<link>` tag within the `<head>` tags. However, when putting in the file name, you will type **[filename].css** even though your file will have a **.scss** extension. Have a look at the example below:

```html
<!doctype html>

<html>
    <head>
        <title>The World's Best Webpage</title>
        <link rel="stylesheet" href = "main.css" type = "text/css">
    </head>
</html>
```

In the example above, the SCSS code has been written in a file called **main.scss** and the compiler that we downloaded has created a file called **main.css** that can be read by the browser. Therefore, we link the CSS file in our HTML file.

We will now look at the features available in SCSS.

VARIABLES

By now you are comfortable with the concept of variables: the container that can hold a particular value. We can create variables in SCSS to represent any aspect of styling. The example below uses variables to define the colour scheme of the webpage:

```
$main-colour:rgb(146, 99, 29);
$secondary-colour: rgb(230, 102, 52);
$accent-colour: rgb(139, 32, 1);
$highlight-colour: rgb(235, 233, 228);
$extra-colour: rgb(230, 203, 51);
```

Note how variables are defined with the dollar sign (`$`). Once we have created the variables, we can use them in our code as normal:

```
h1 {
    color: $main-colour;
    font-family: 'Franklin Gothic Medium';
    font-size: 32pt;
    text-align: center;
}

h2 {
    color: $secondary-colour;
    font-family: 'Courier New', Courier, monospace;
    font-size: 20pt;
    text-align: center;
}
```

As you can see, we are using the main colour (**`$main-colour`**) for heading 1 and the secondary colour (**`$secondary-colour`**) for heading 2. This is especially useful when you want to change the colour scheme of your web page at a later stage. Now, instead of needing to change every instance you mentioned a colour, you can simply change the values of the variables.

Next, we will look at nesting.

NESTING

You will be familiar with nesting from your task on loops, and the concept in SCSS is very similar. We can use nesting to specify a style element that we only want if this element is within another particular element. Let's look at an example:

```
ul {
    color: $highlight-colour;
    background-color: $accent-colour;
    font-size: 16pt;
    font-style: italic;

    ol {
        color: $secondary-colour;
        background-color: $highlight-colour;
        font-weight: bold;
    }
}
```

In the code above, we have an ordered list that is nested in an unordered list. The ordered list will take on all the elements of the unordered list *as well as* the elements of the ordered list itself. However, ordered lists outside of an unordered list will not take those specific elements.

MODULES AND PARTIALS

Modules and partials are extra SCSS files that we can create that can be imported by our main SCSS files. That way, we can access the functions, variables and other elements from these files in our main SCSS file.

Modules and partials follow the same filename rule: the filenames need to start with an underscore (_), e.g. **_contact-page.scss**. This tells the compiler that it is not the main SCSS file and so does not need a .css equivalent. It also allows you to import the file into the main SCSS file. Have a look at the example below:

```
// _image-manipulation.scss
.center {
    display: block;
    margin-left: auto;
    margin-right: auto;
}
```

This is a file called **_image-manipulation.scss** that contains a function to center an image. We can import this file into our main file with the line:

```
@import "image-manipulation";
```

Now we can call this function in our HTML file because it is linked to our main SCSS file:

```
<img class = "center" src =
"https://live.staticflickr.com/3574/3338852116_e06a7a111f_b.jpg" alt="A
Street Called Awesome"> <br>
```

INHERITANCE

Inheritance in programming is the idea of extending the functionality of an element to another element. For example, we could create a new function called `.image-manip` that could be used to make our image placement and styling more consistent:

```
// _image-manipulation.scss

.center {
    display: block;
    margin-left: auto;
    margin-right: auto;
}

.image-manip {
    @extend .center;
    border: 5px solid yellow;
}
```

In the code above, we have extended the `center` function inside the `image-manip` function. That means that if we use the `image-manip` function, the center function will also be called.

```
<img class = "image-manip" src =
"https://live.staticflickr.com/3574/3338852116_e06a7a111f_b.jpg" alt="A
Street Called Awesome"> <br>
```

Now, when we add the **`image-manip`** function to our image tag, it will be centred and have a 5-pixel yellow border.

MIXINS

A mixin is similar to a function in JavaScript: it allows you to make and reuse groups of declarations in your code. Like functions in JavaScript, mixins can also take arguments. For example, say you wanted the left and right margins to be equal with any element, but the sizes of the margins will differ depending on what the element is. We can write a mixin:

```
@mixin margins($pixels) {
    margin-left: $pixels;
    margin-right: $pixels;
}
```

Here, we can see the mixin is called margins and it takes one argument, which is what the size of the margins will be. Because the left and right margins hold the same value, the sizes will be equal. We can then use the mixin by using **`@include`**:

```
p {
    color: $accent-colour;
    font-family: Helvetica;
    @include margins(468px);
}

a {
    @include margins(568px);
}
```

As you can see, the margins mixin has been used for paragraphs and links, but the sizes of the margins will be different because different values have been passed into it.

OPERATORS

Just like the operators you learned in JavaScript (+, -, *, /), SCSS allows you to use these same operators in your styling. The only difference is that in SCSS the percentage sign (%) is used for a percentage — it is not modulus. You can use these operators to give the value of image or font sizes and even colours! Have a look at the code below:

```
$paragraph-size: 14pt;

h1 {
    color: $main-colour;
    font-family: 'Franklin Gothic Medium';
    font-size: $paragraph-size + 18pt;
    text-align: center;
}

h2 {
    color: $secondary-colour;
    font-family: 'Courier New', Courier, monospace;
    font-size: $paragraph-size + 6pt;
    text-align: center;
}

p {
    color: $accent-colour;
    font-family: Helvetica;
    font-size: $paragraph-size;
}
```

Here we are describing the font size of h1 and h2 headings in relation to the size of the font used for the paragraphs, as specified in the variable `$paragraph-size`. That means that changing the value of the variable will alter the size of all three text types.

Compulsory Task

Follow these steps:

- Find a published web page that you particularly like (not something you've already made). It could be from Netflix, takealot, ucook, or any other website you like.
- Create an HTML file called **copy-cat.html**.
- Using HTML and SCSS, try to recreate the webpage as best you can.
- Add the URL of the webpage you are recreating as a link at the bottom of your webpage.
- You may want to do some research on more CSS and SCSS elements for greater functionality.