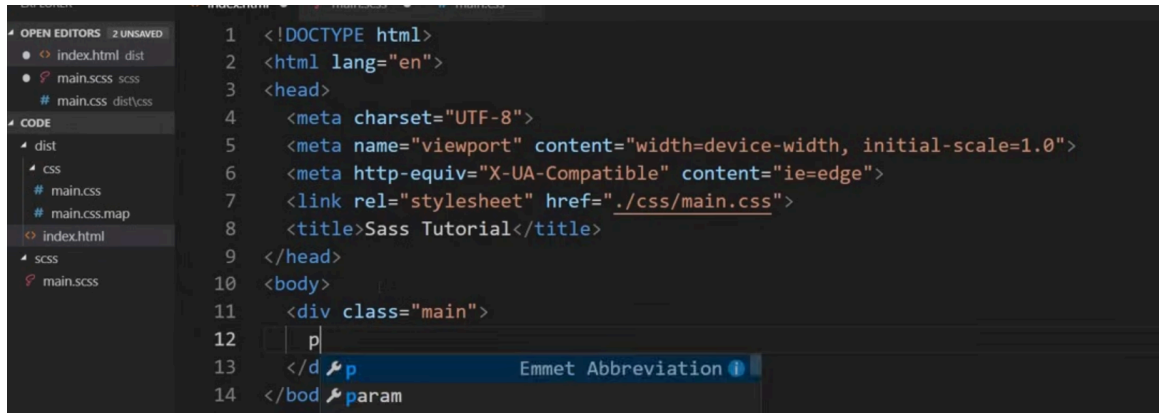


Variable and maps

folder structures and the html file

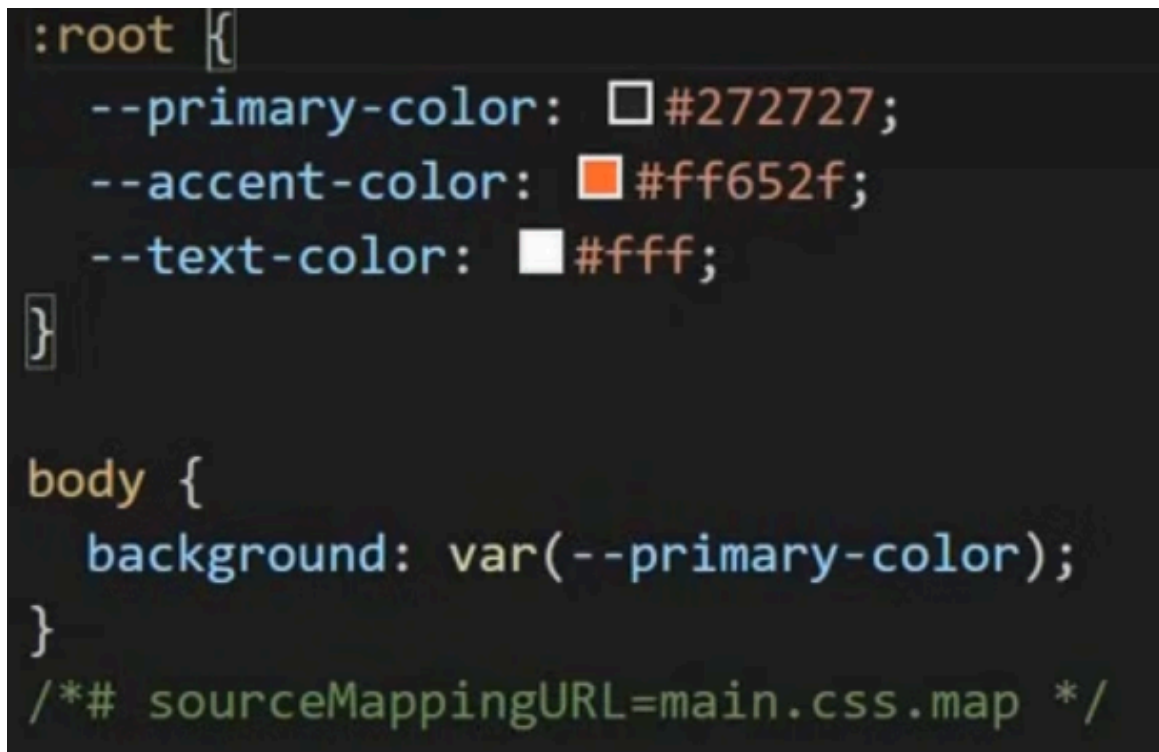
css will be automatically generated



The screenshot shows the VS Code editor with a project structure on the left sidebar. The project structure includes a 'dist' folder containing 'css' and 'main.css.map', and an 'index.html' file. The main editor displays the content of 'index.html', which is an HTML document with a head section containing meta tags for charset, viewport, and http-equiv, a link to the main.css file, and a title 'Sass Tutorial'. The body section contains a div with the class 'main' and a paragraph element. The paragraph element is currently selected, and an Emmet Abbreviation tooltip is visible, showing the text 'param'.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <link rel="stylesheet" href="/css/main.css">
8   <title>Sass Tutorial</title>
9 </head>
10 <body>
11   <div class="main">
12     <p>
13   </div>
14 </body>
```

variable in css



The screenshot shows a CSS file with a root selector and a body selector. The root selector contains three variables: --primary-color, --accent-color, and --text-color. The body selector uses the --primary-color variable for the background property. The file ends with a source map comment indicating the source mapping URL is 'main.css.map'.

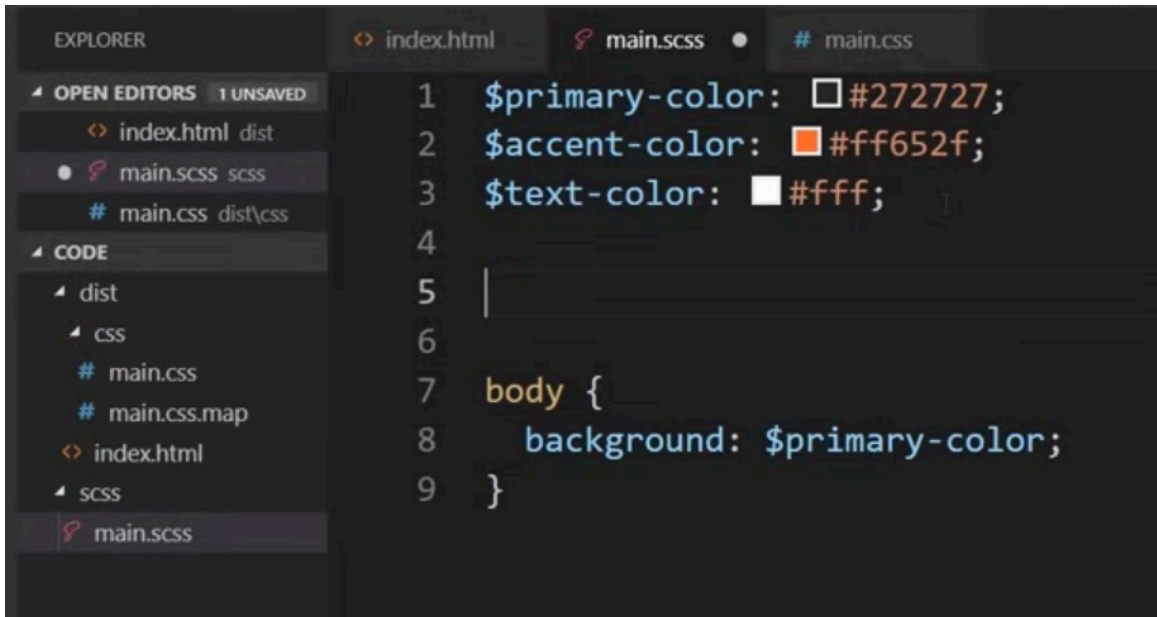
```
:root {
  --primary-color: #272727;
  --accent-color: #ff652f;
  --text-color: #fff;
}

body {
  background: var(--primary-color);
}

/*# sourceMappingURL=main.css.map */
```

Variable in scss

we use the dollar sign to create the variable in the CSS



Sass Variables

Variables are a way to store information that you can re-use later.

With Sass, you can store information in variables, like:

- strings
- numbers
- colors
- Booleans
- lists
- nulls

\$myFont: Helvetica, sans-serif;

\$myColor: red;

\$myFontSize: 18px;

```
$myWidth: 680px;
```

```
body {  
  font-family: $myFont;  
  font-size: $myFontSize;  
  color: $myColor;  
}
```

```
#container {  
  width: $myWidth;  
}
```

Using Sass !global

The default behavior for variable scope can be overridden by using the `!global` switch.

`!global` indicates that a variable is global, which means that it is accessible on all levels.

Look at the following example (same as above; but with `!global` added):

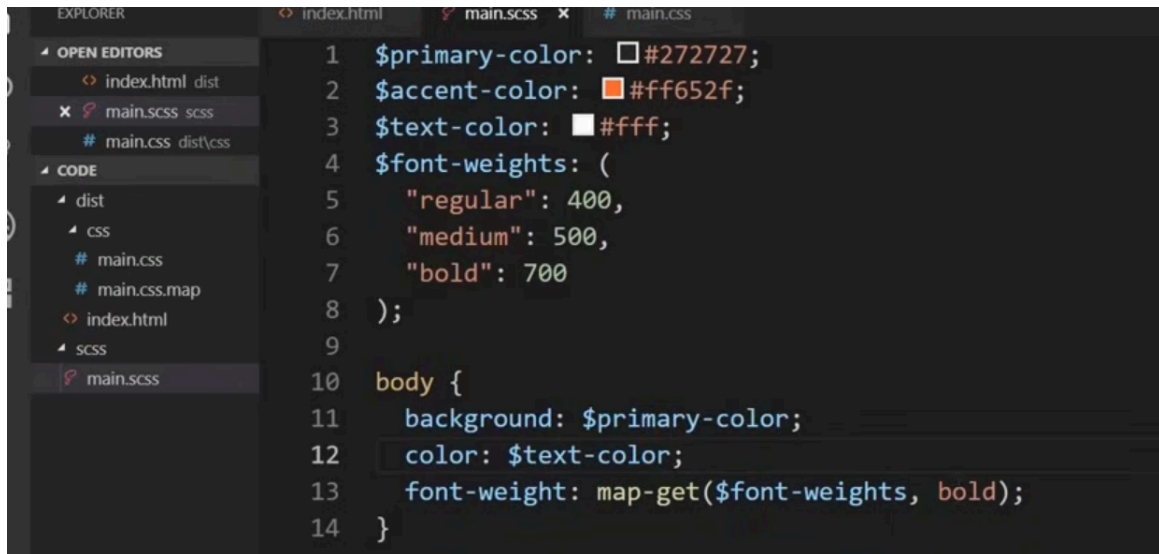
```
$myColor: red;
```

```
h1 {  
  $myColor: green !global;  
  color: $myColor;  
}
```

```
p {  
  color: $myColor;
```

```
}
```

Map in the scss



The screenshot shows a VS Code editor with the following SCSS code in `main.scss`:

```
1 $primary-color: #272727;
2 $accent-color: #ff652f;
3 $text-color: #fff;
4 $font-weights: (
5   "regular": 400,
6   "medium": 500,
7   "bold": 700
8 );
9
10 body {
11   background: $primary-color;
12   color: $text-color;
13   font-weight: map-get($font-weights, bold);
14 }
```

The Explorer sidebar on the left shows the project structure with files like `index.html`, `main.scss`, and `main.css`.

converted css file



The screenshot shows the converted CSS file with the following content:

```
body {
  background: #272727;
  color: #fff;
  font-weight: 700;
}
/*# sourceMappingURL=main.css.map */
```