GeeksforGeeks
A computer science portal for geeks

Custom Search

COURSES

Login

HIRE WITH US

CSS Preprocessor | SASS

SASS is the abbreviation of Syntactically Awesome Style Sheets. It is a CSS pre-processor with syntax advancements. The style sheets in the advanced syntax are processed by the program and compiled into regular CSS style sheets, which can be used in the website. It is a CSS extension that allows to use features like variables, nesting, imports, mixins, inheritance, etc, all in a CSS-compatible syntax.

Note: The link describes the installation process of SASS. Ref: https://sass-lang.com/install

There are two types of syntax available for SASS:

- SCSS(Sassy CSS): The files using this syntax use *.scss* extension.
- Indented syntax (referred to as just "sass"): older syntax, Files using this syntax use *.sass* extension.

Note: This example use *.scss* extension.

Working Steps:

- Write the SCSS code.
- Compile the SCSS code into CSS code using the command *sass input.scss output.css*. The first filename (input.scss) is the scss file that is to be compiled and the second file name (output.css) is the processed CSS file, to be included/attached in the html document.
- Include the compiled CSS file in the html file.

Now see how to make effective use of the important features of SCSS like variables, nesting, mixins and operators.

- The main HTML file is named *index.html*
- SCSS file is *styling.scss* and the CSS file is *style.css*
- Command to compile the SCSS file: *sass styling.scss style.css*

Example: File name *index.html*

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8"/>

        <title>SASS</title>

        <meta name="viewport" content=
            "width=device-width, initial-scale=1">

        <link rel="stylesheet" href="style.css">
    </head>

    <body>
        <div id="d1">Welcome to GeeksforGeeks.
            <ul>
                <li>Algo</li>
                <li>DS</li>
                <li>Languages</li>
                <li>Interviews</li>
                <li>CS subjects</li>
            </ul>
        </div>
    </body>
</html>
```
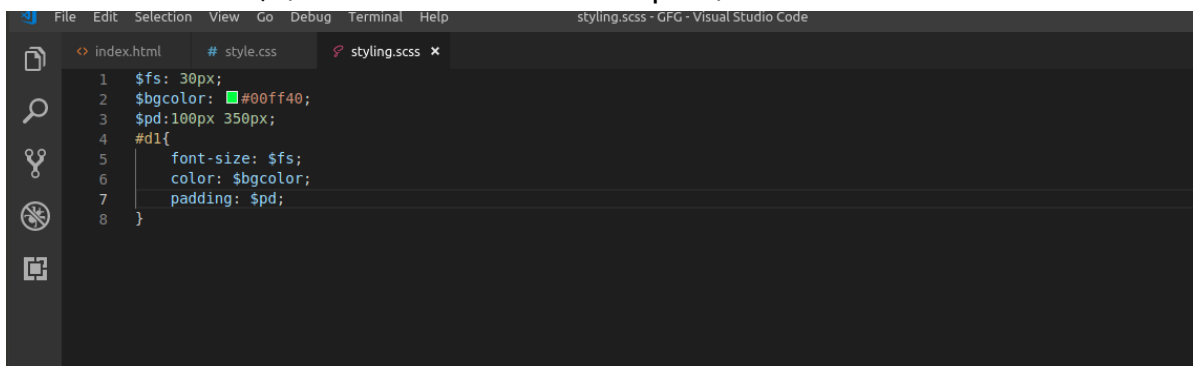
- **Variables:** Variables can be used to store CSS value that may be reused. To declare a variable in SASS, '$' character is used. Example: $v_name

```
File  Edit  Selection  View  Go  Debug  Terminal  Help          styling.scss - GFG - Visual Studio Code

  <> index.html        # style.css      $ styling.scss  ×
    1   $fs: 30px;
    2   $bgcolor: ▮#00ff40;
    3   $pd:100px 350px;
    4   #d1{
    5       font-size: $fs;
    6       color: $bgcolor;
    7       padding: $pd;
    8   }
```

After compiling the CSS code save it file by *style.css*

```
File  Edit  Selection  View  Go  Debug  Terminal  Help          style.css - GFG - Visual Studio Code

 <> index.html        # style.css   ✕     ⌁ styling.scss

  1    #d1 {
  2        font-size: 30px;
  3        color: ■#00ff40;
  4        padding: 100px 350px; }
  5
  6    /*# sourceMappingURL=style.css.map */
  7

  OUTPUT    PROBLEMS    DEBUG CONSOLE    TERMINAL

  nayonika@developer:~/Desktop/GFG/i1$ sass styling.scss style.css
  nayonika@developer:~/Desktop/GFG/i1$ █
```

- **Nesting:** SASS allows CSS rules to be nested within each other, which follows the same visual hierarchy of HTML. For Example: CSS property can be used to the <li> tag nested inside the div tag.

```
 <> index.html        ⌁ styling.scss  ✕     # style.css

  1    $fs: 30px;
  2    $bgcolor: ■#00ff40;
  3    $col2: ■#ff0066e1;
  4    $pd:100px 350px;
  5    #d1{
  6        font-size: $fs;
  7        color: $bgcolor;
  8        padding: $pd;
  9        li{
 10            color:$col2;
 11        }
 12    }
 13    |
```
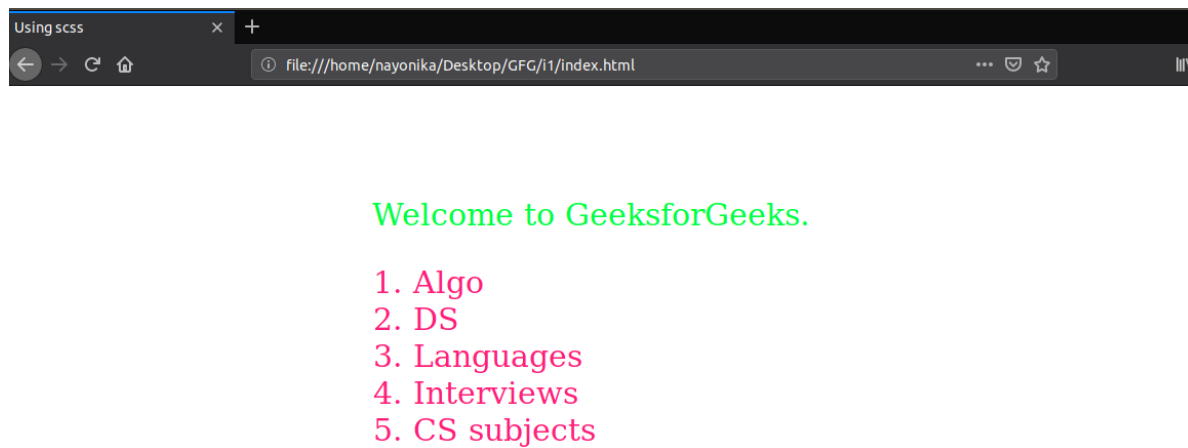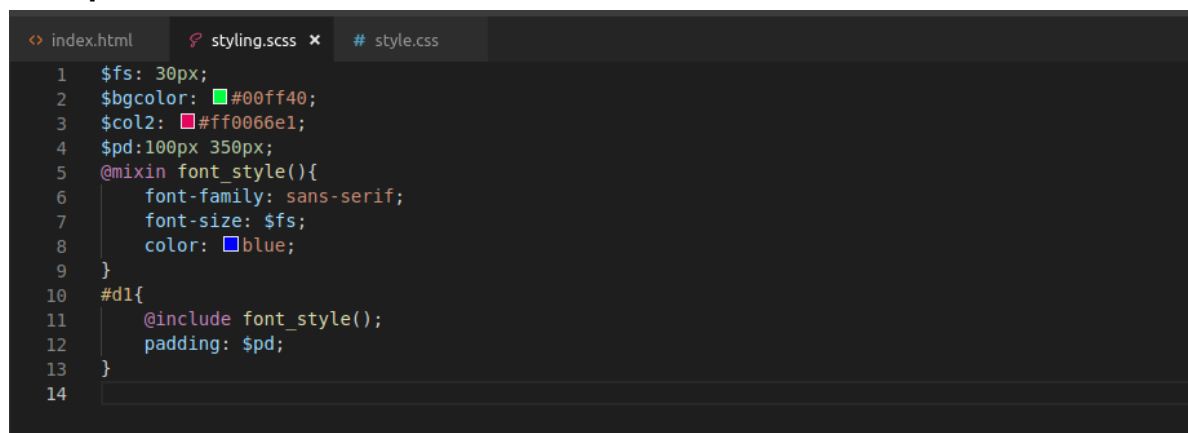
After compiling the CSS code save it file by *style.css*

```
 <> index.html        ⌁ styling.scss     # style.css   ✕

  1    #d1 {
  2      font-size: 30px;
  3      color: ■#00ff40;
  4      padding: 100px 350px; }
  5      #d1 li {
  6        color: ■#ff0066e1; }
  7
  8    /*# sourceMappingURL=style.css.map */
  9

  OUTPUT    PROBLEMS    DEBUG CONSOLE    TERMINAL

  nayonika@developer:~/Desktop/GFG/i1$ sass styling.scss style.css
  nayonika@developer:~/Desktop/GFG/i1$ █
```
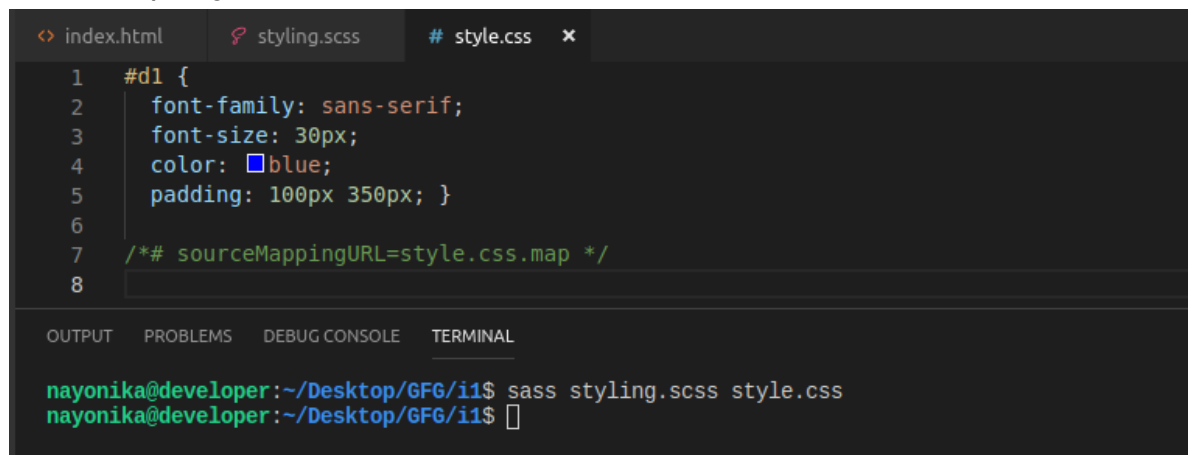
Output:

Welcome to GeeksforGeeks.

1. Algo
2. DS
3. Languages
4. Interviews
5. CS subjects

- **Mixins:** Mixins helps to make a set of CSS properties reusable i.e. it save one code and use it again and again. It can be included in other CSS rules by using *@include* directive.

Example:

```scss
$fs: 30px;
$bgcolor: #00ff40;
$col2: #ff0066e1;
$pd:100px 350px;
@mixin font_style(){
    font-family: sans-serif;
    font-size: $fs;
    color: blue;
}
#d1{
    @include font_style();
    padding: $pd;
}
```

After compiling the CSS code becomes:

```css
#d1 {
    font-family: sans-serif;
    font-size: 30px;
    color: blue;
    padding: 100px 350px; }

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

```
OUTPUT    PROBLEMS    DEBUG CONSOLE    TERMINAL

nayonika@developer:~/Desktop/GFG/i1$ sass styling.scss style.css
nayonika@developer:~/Desktop/GFG/i1$
```

The output of web page:

Using scss × +

file:///home/nayonika/Desktop/GFG/i1/index.html

Welcome to GeeksforGeeks.

1. Algo
2. DS
3. Languages
4. Interviews
5. CS subjects
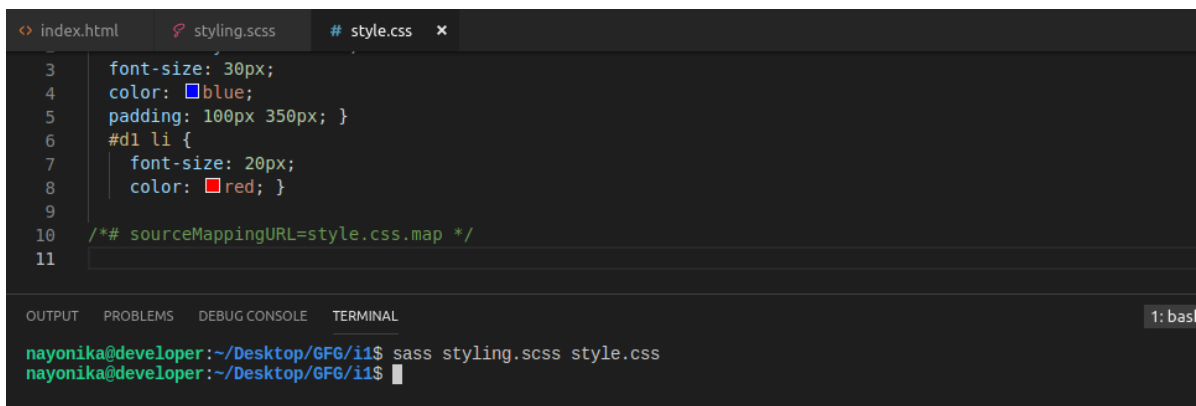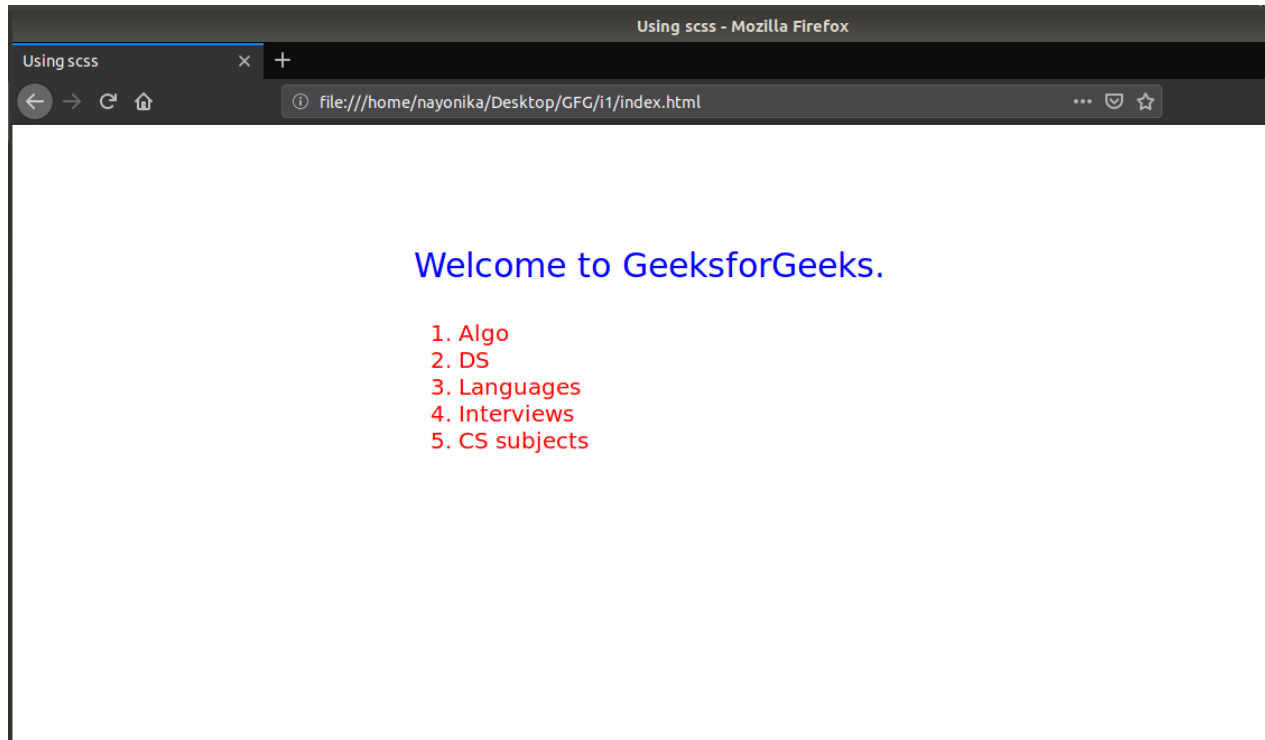
Example: Mixins can also take variables as arguments. The values are passed while including them in the CSS rules.

```scss
$col2: #ff0066e1;
$pd:100px 350px;
@mixin font_style(){
    font-family: sans-serif;
    font-size: $fs;
    color: blue;
}
@mixin list_style($size,$color) {
    font-size: $size;
    color: $color;
}
#d1{
    @include font_style();
    padding: $pd;
    li{
        @include list_style(20px, red);
    }
}
```

The compiled CSS code:

```css
    font-size: 30px;
    color: blue;
    padding: 100px 350px; }
    #d1 li {
        font-size: 20px;
        color: red; }

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

OUTPUT   PROBLEMS   DEBUG CONSOLE   TERMINAL   1: bash

```
nayonika@developer:~/Desktop/GFG/i1$ sass styling.scss style.css
nayonika@developer:~/Desktop/GFG/i1$
```

Final Output:

Recommended Posts:

What is the difference between ng-if and data-ng-if directives ?

D3.js | d3.timeSunday Function

D3.js | d3.utcSaturdays() Function

D3.js | d3.timeSaturdays() Function

HTML | DOM animationend Event

p5.js | createColorPicker() Function

D3.js | d3.utcSaturday Function

Web Audio API | AudioNode context property

D3.js | d3.timeSundays() Function

D3.js | d3.timeThursday Function

D3.js | d3.utcDays() Function

D3.js | d3.utcSundays() Function

HTML | DOM onafterprint Event

D3.js | d3.timeSaturday Function

NAYONIKA
Check out this Author's underline contributed articles.

If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.

Article Tags : CSS   Web Technologies   CSS-Misc

👍

6

☐ To-do ☐ Done   **3**

Based on **2** vote(s)

(Feedback/ Suggest Improvement) (Add Notes) (Improve Article)

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

About Us
Careers
Privacy Policy
Contact Us

LEARN

Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE

Courses
Company-wise
Topic-wise
How to begin?

CONTRIBUTE

Write an Article
Write Interview Experience
Internships
Videos