* **Sass** stands for **S**yntactically **A**wesome **S**tyle**s**heet
* Sass is an **extension** to CSS
* Sass is a **CSS pre-processor**
* Sass is completely compatible with all versions of CSS
* Sass **reduces repetition** of CSS and therefore saves time
* Sass was designed by Hampton Catlin and developed by Natalie Weizenbaum in 2006
* Sass is free to download and use
* **Operating system** - Sass is platform independent (Cross platform)
* **Preprocessor Scripting language** - Sass is based on Ruby(originally) | dart.
* Sass files has the ".scss" file extension

# How Does Sass Work?

A browser does not understand Sass code. Therefore, you will need a Sass pre-processor to convert Sass code into standard CSS.

This process is called transpiling. So, you need to give a transpiler (some kind of program) some Sass code and then get some CSS code back.

**Tip:** Transpiling is a term for taking a source code written in one language and transform/translate it into another language.

# Comments

Sass supports standard CSS comments /\* comment \*/

In addition, it supports inline comments // comment

# Variable

$*variablename*: *value*;

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

$myColor: red;  
  
h1 {  
  $myColor: green !global;   
  color: $myColor; // green (without !global) & green (with !global)  
}  
  
p {  
  color: $myColor; // red (without !global) & green (with !global)  
}

# Partials

By default, Sass transpiles all the .scss files directly. However, when you want to import a file, you do not need the file to be transpiled directly.

Sass has a mechanism for this: If you start the filename with an underscore (\_partials), Sass will not transpile it. Files named this way are called partials in Sass.

# @import

The @import directive allows you to include the content of one file in another.

The CSS @import directive has a major drawback due to performance issues; it creates an extra HTTP request each time you call it.

However, the Sass @import directive includes the file in the CSS; so, no extra HTTP call is required at runtime!

# @mixin & @include

***function with or without parameters***

The @mixin directive lets you create CSS code that is to be reused throughout the website.

The @include directive is created to let you use (include) the mixin.

# @extend Directive

***Inheritance***

The @extend directive lets you share a set of CSS properties from one selector to another.

The @extend directive is useful if you have almost identically styled elements that only differ in some small details.