Sass Tutorial

Sass is a **CSS pre-processor**.

Sass **reduces repetition of CSS** and therefore saves time.

## **What is Sass?**

* **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**

## **Why Use Sass?**

Stylesheets are getting

1. **Larger**
2. **more complex**
3. **Harder to maintain.**

This is where a CSS pre-processor can help.

Sass lets you use **features that do not exist in CSS**, like

1. **variables,**
2. **nested rules,**
3. **mixins,**
4. **imports,**
5. **inheritance,**
6. **built-in functions,**
7. **and other stuff.**

## **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.

## **Sass File Type**

Sass files has the ".scss" file extension.

## **Sass Comments**

Sass supports standard CSS comments /\* comment \*/, and in addition it supports inline comments // comment:

## **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**

Sass uses the **$ symbol**, followed by a name, to declare variables:

Sass Variable Syntax:

**$*variablename*: *value*;**

EXAMPLE:

$myFont: Helvetica, sans-serif;

$myColor: red;

$myFontSize: 18px;

$myWidth: 680px;

body {

font-family: $myFont;

font-size: $myFontSize;

color: $myColor;

}

## **Sass Variable Scope**

Sass variables are only **available at the level of nesting where they are defined**.

Look at the following example:

$myColor: red;  
  
h1 {  
  $myColor: green;  
  color: $myColor;  
}  
  
p {  
  color: $myColor;  
}

**OUTPUT:**

# Hello World

This is a paragraph.

## **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;

}

**OUTPUT:**

# Hello World

This is a paragraph.