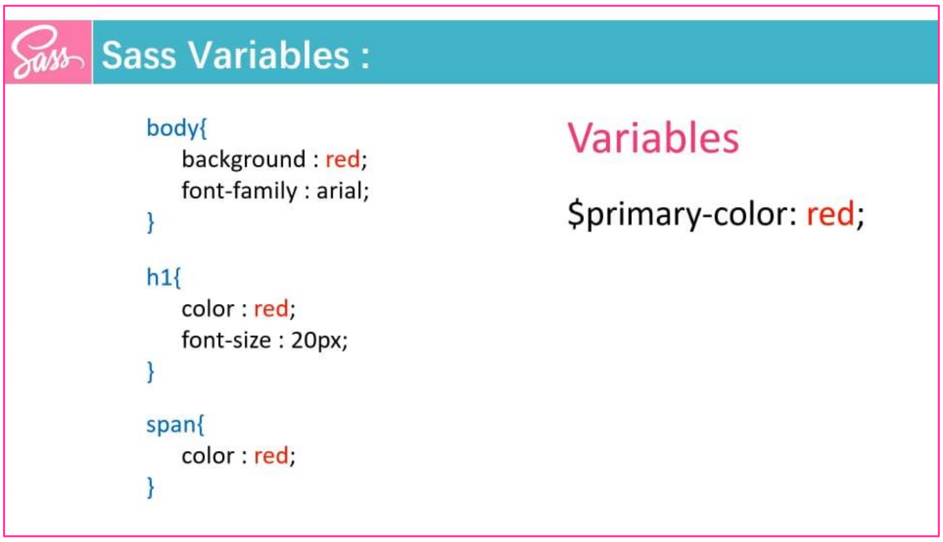
SASS (Syntactically Awesome Style Sheet)

It is the extension of CSS. It has a lot of benefits. It makes the code easier to understand. And increase the Code reusability.

1. Variable

In this Styling language of web, we declare the variables.



Sass Variables :

```
body{
    background : red;
    font-family : arial;
}

h1{
    color : red;
    font-size : 20px;
}

span{
    color : red;
}
```

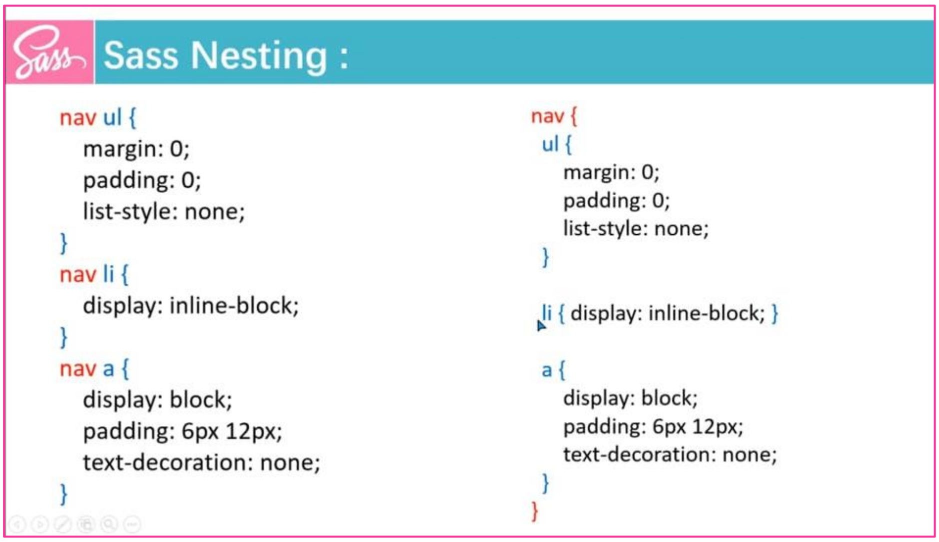
Variables

```
$primary-color: red;
```

Replace the Red in the Body,h1 and span selector with variable name like this:

```
body{
background:$primary-color
}
```

2. Nesting



Sass Nesting :

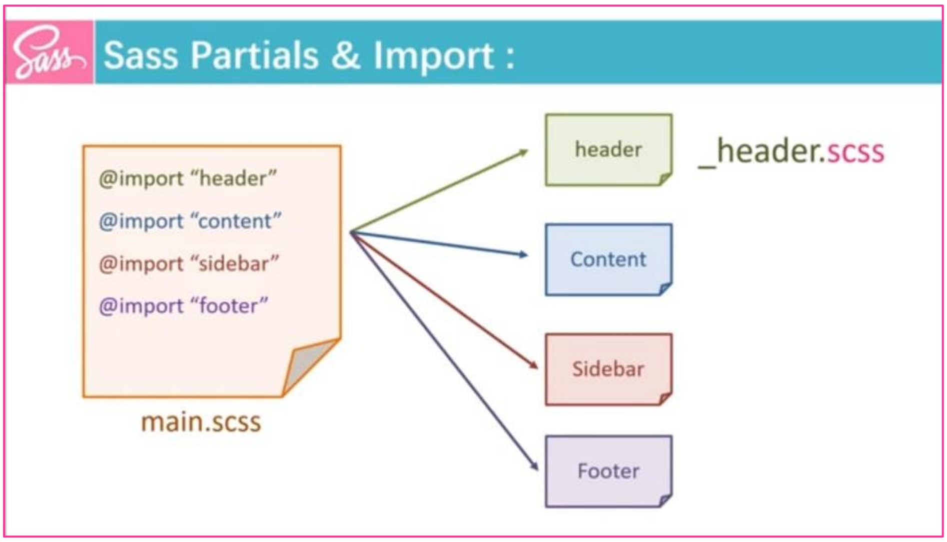
```
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}
nav li {
  display: inline-block;
}
nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none;
}
```

```
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

Use **the parent selector** Just one time and then nesting it.

Note is Prepared by Muhammad Zubair

3. Partials / Import



Sass Partials & Import :

```
@import "header"
@import "content"
@import "sidebar"
@import "footer"
```

main.scss

header

_header.scss

Content

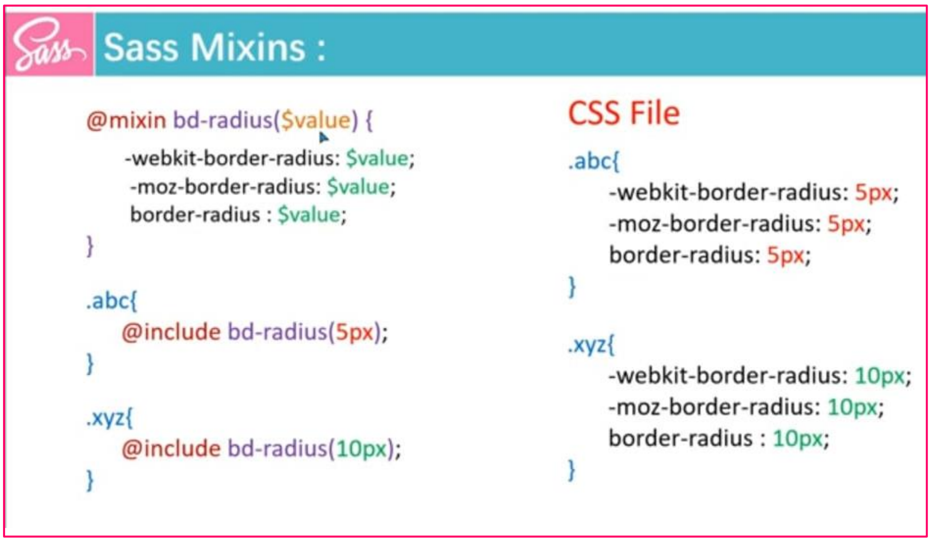
Sidebar

Footer

Divide the main file into section and store these section into multiple files and start the file name with Underscore. and import into main with the sequence.

@import"filename"

4. Mixins



Sass Mixins :

```
@mixin bd-radius($value) {
     -webkit-border-radius: $value;
     -moz-border-radius: $value;
     border-radius : $value;
}

.abc{
     @include bd-radius(5px);
}

.xyz{
     @include bd-radius(10px);
}
```

CSS File

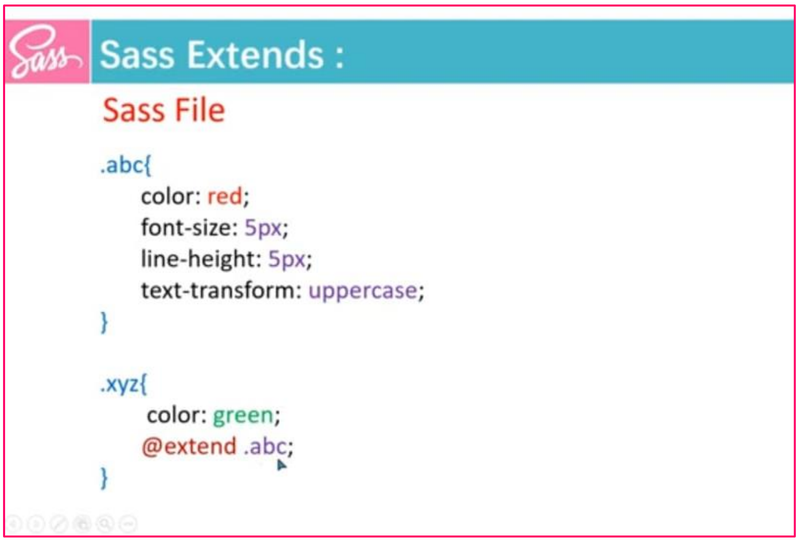
```
.abc{
     -webkit-border-radius: 5px;
     -moz-border-radius: 5px;
     border-radius: 5px;
}

.xyz{
     -webkit-border-radius: 10px;
     -moz-border-radius: 10px;
     border-radius : 10px;
}
```

Mixins is working like a function. We separate over repeatable code. Mixins can be parameterized or non parameterized.

If the maxins is parameterized so parameter must be provided otherwise the error occur.

Note is Prepared
by Muhammad Zubair

5. Extends



Sass Extends :

Sass File

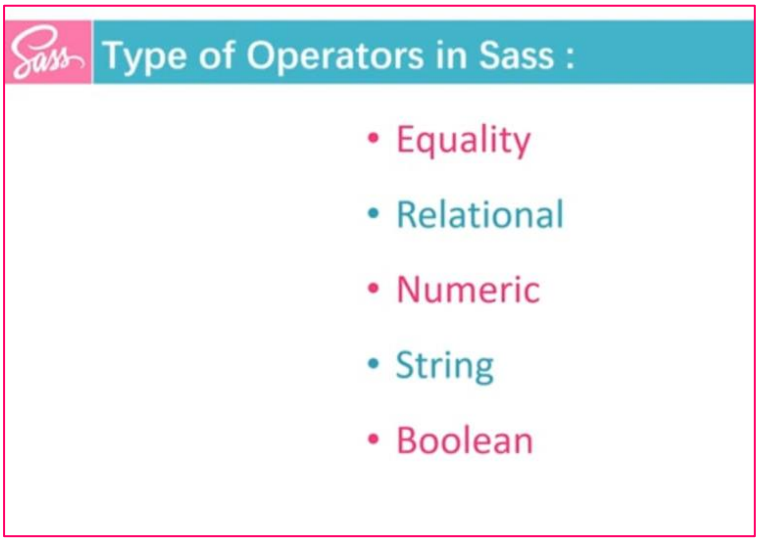
```
.abc{
      color: red;
      font-size: 5px;
      line-height: 5px;
      text-transform: uppercase;
}

.xyz{
       color: green;
      @extend .abc;
}
```

Extends is working like a Mixins.

If you don't show the code of extended class or id in CSS we use placeholder. Just began the extended class name with % like.

%abc{
}

6. Operators

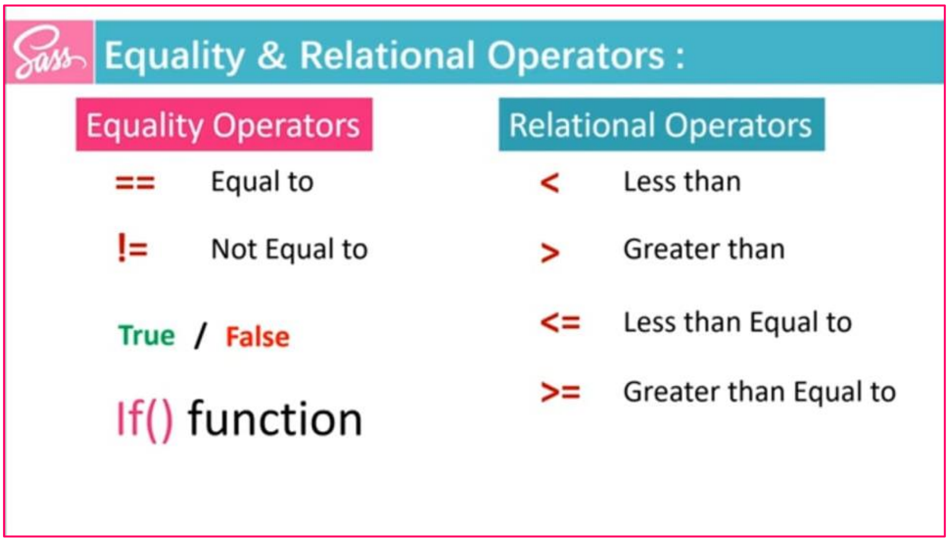


Type of Operators in Sass :

- Equality
- Relational
- Numeric
- String
- Boolean

These types of operators are used in SASS.

Note is Prepared
by Muhammad Zubair



Equality & Relational Operators :

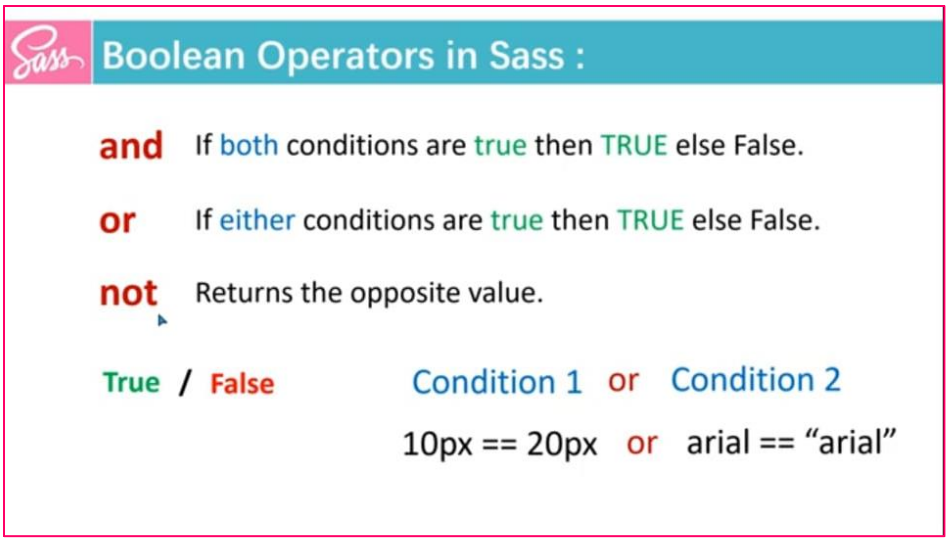
Equality Operators	
==	Equal to
!=	Not Equal to

True / False

If() function

Relational Operators	
<	Less than
>	Greater than
<=	Less than Equal to
>=	Greater than Equal to

These types of operators work same as other programming language
These operator are basically used in **if()** function to make the conditions.



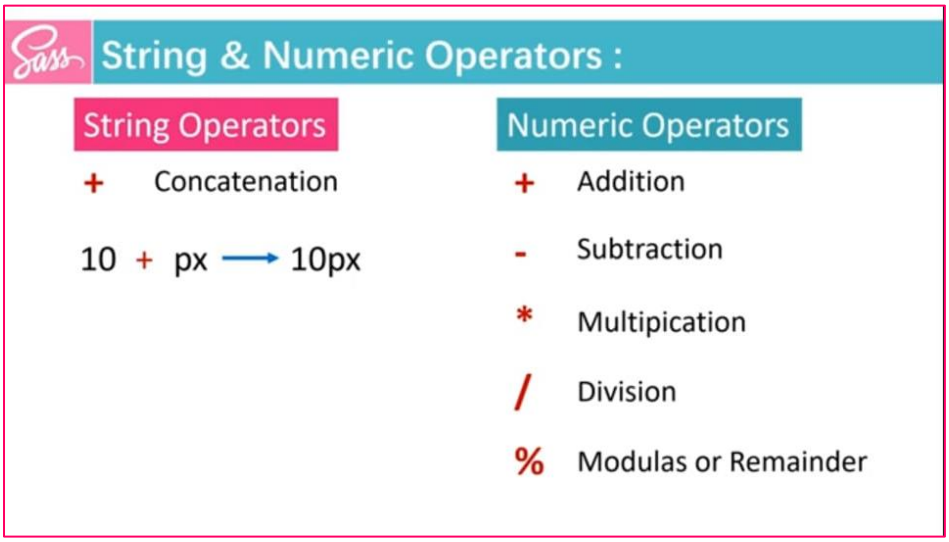
Boolean Operators in Sass :

and	If both conditions are true then TRUE else False.
or	If either conditions are true then TRUE else False.
not	Returns the opposite value.

True / False

Condition 1 or Condition 2

10px == 20px or arial == “arial”



String & Numeric Operators :

String Operators	
+	Concatenation

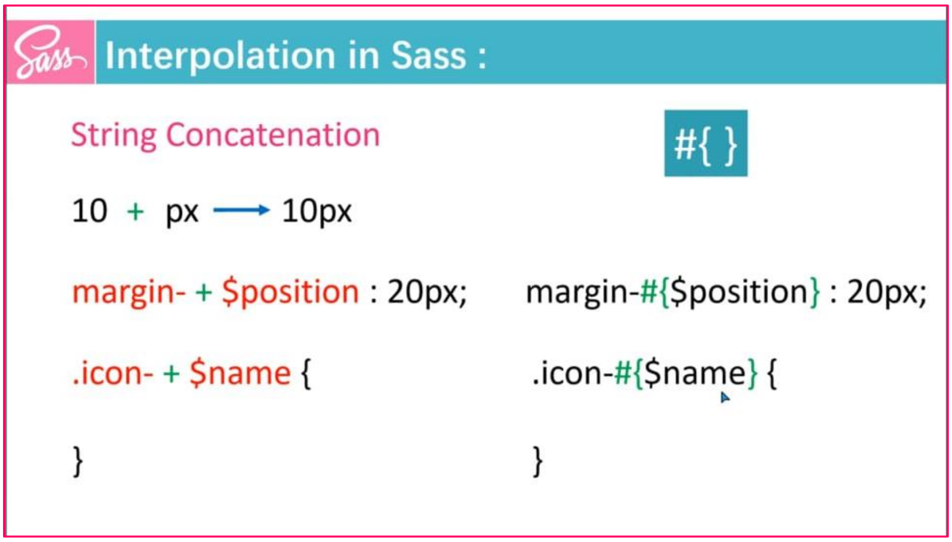
10 + px → 10px

Numeric Operators	
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulas or Remainder

String operator is used for the concatenation of value. Through string operator we just concatenate the right side value not the properties.

Note is Prepared
by Muhammad Zubair

7. Interpolation



Interpolation in Sass :

String Concatenation

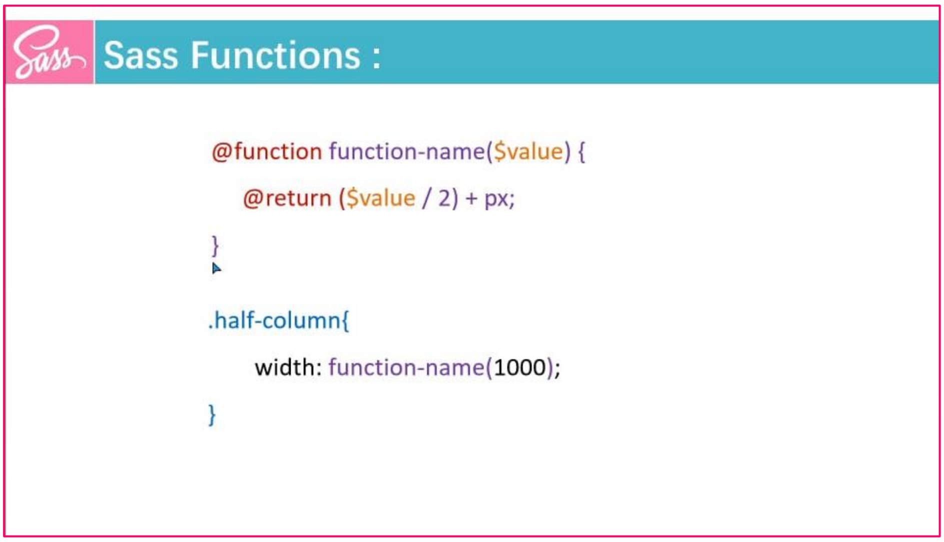
#{ }

10 + px → 10px

```
margin- + $position : 20px;      margin-#{$position} : 20px;

.icon- + $name {                 .icon-#{$name} {

}                                }
```

Interpolation is used for the concatenation of properties.

8. Functions



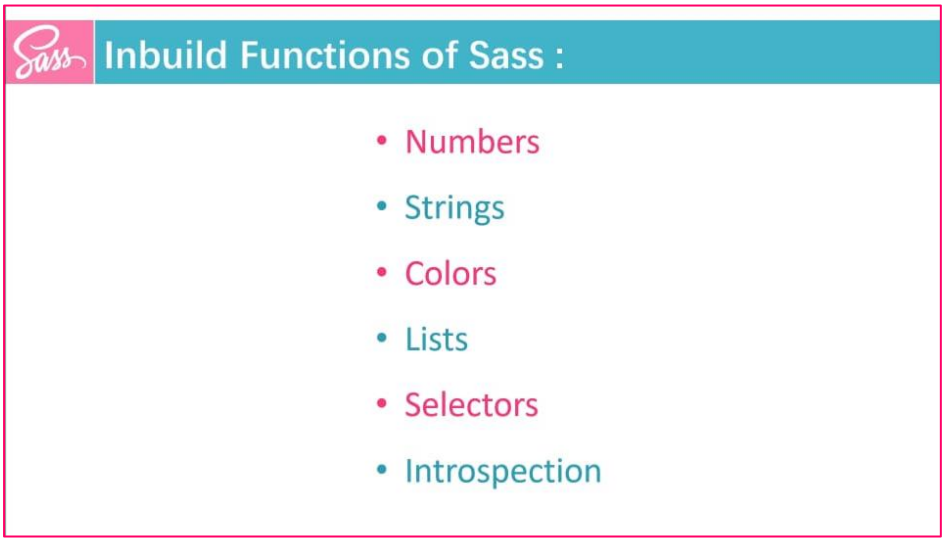
Sass Functions :

```
@function function-name($value) {
    @return ($value / 2) + px;
}

.half-column{
     width: function-name(1000);
}
```

The main difference between mixin and function is mixin does not return anything but the function return the value.

Note is Prepared
by Muhammad Zubair

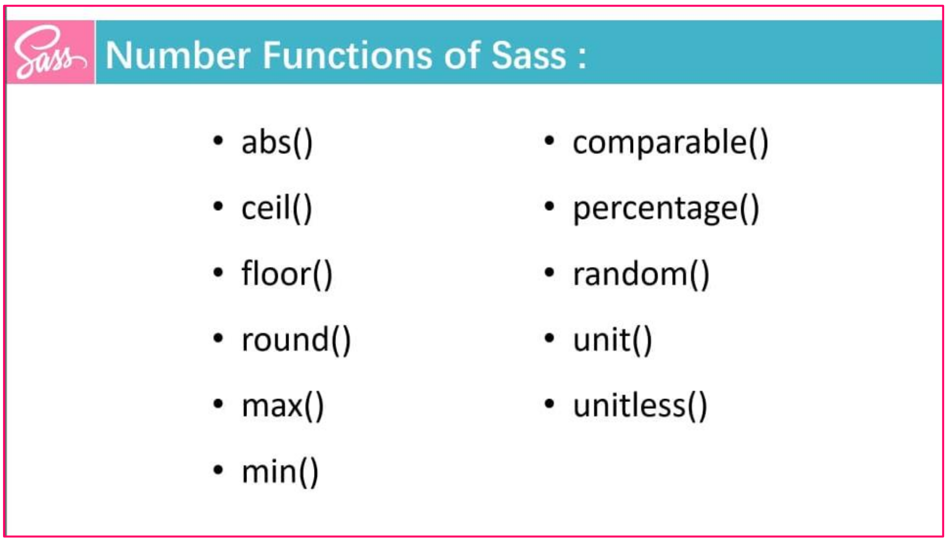
9. InBuild Functions



Inbuild Functions of Sass :

- Numbers
- Strings
- Colors
- Lists
- Selectors
- Introspection

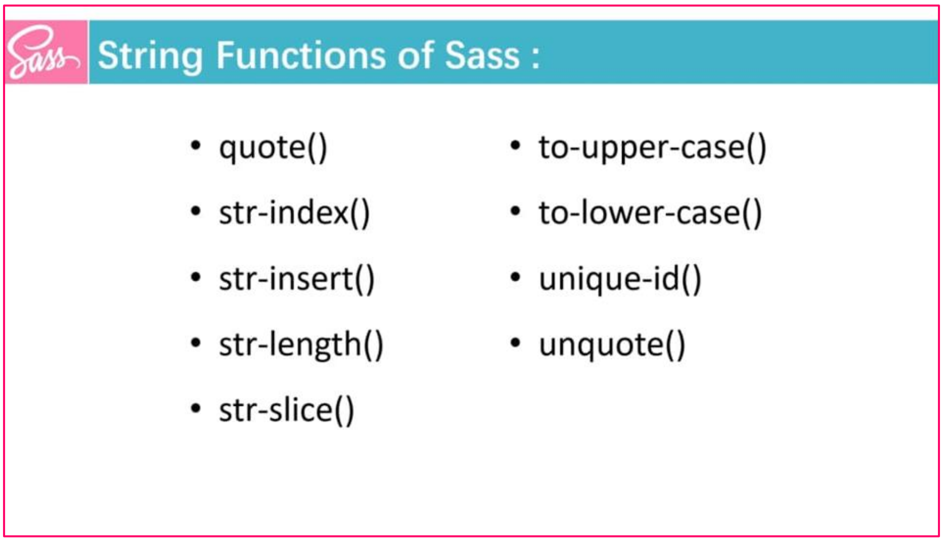
These are some built-in functions in SASS mostly functions are working like JavaScript functions.



Number Functions of Sass :

- abs()
- ceil()
- floor()
- round()
- max()
- min()
- comparable()
- percentage()
- random()
- unit()
- unitless()

Understand the functionality by its name.
Just abs() is changed it return the absolute value.

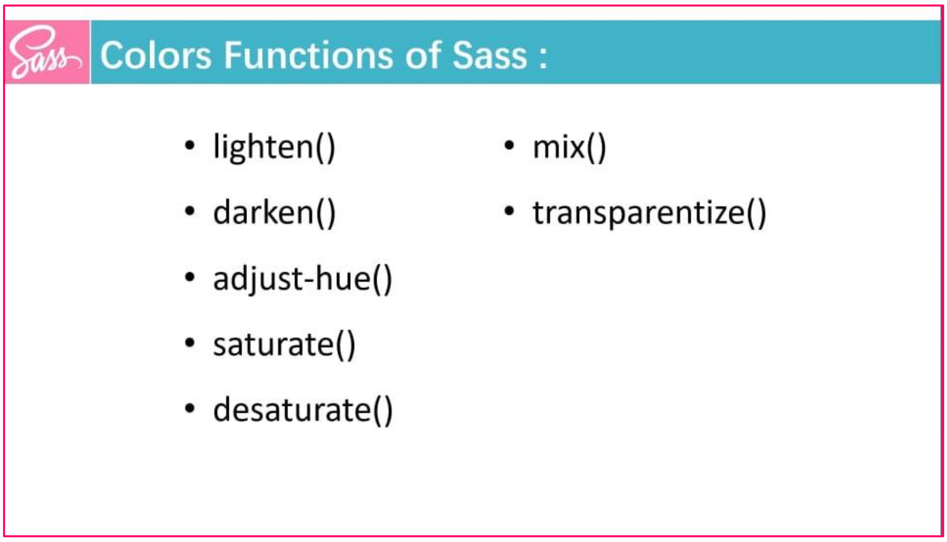


String Functions of Sass :

- quote()
- str-index()
- str-insert()
- str-length()
- str-slice()
- to-upper-case()
- to-lower-case()
- unique-id()
- unquote()

These string functions are also understandable by its name.

**Note is Prepared
by Muhammad Zubair**



Colors Functions of Sass :

- lighten()
- darken()
- adjust-hue()
- saturate()
- desaturate()
- mix()
- transparentize()

These are commonly used color built-in functions. These are also understandable by its name :
Saturate increased the brightness and desaturate decreased it.

Note is Prepared by Muhammad Zubair