Sass Functions

Sudhakar Sharma

•

Mar 11

Sass Functions  
@if  
@else  
@for  
  
@if        : It is used in decision making.  
          It executes a set of statements when given condition is true  
  
@else        : It is used as a clause for "@if"  
          It is an alternative.  
         It executes a set of statements when given condition is false  
  
 Syntax: Simple Decision  
  
        @if  condition    
         {  
         }  
        @else  
        {  
        }  
  
  Syntax: Multiple Decisions  
         
        @if  condition  
        {  
        }  
        @else if condition  
        {  
        }  
        @else if condition  
        {  
        }  
        @else  
        {  
        }  
  
Ex:  
effects.scss  
  
  
$themeColor: green;  
$borderSize: 12px;  
  
@if $themeColor==red {  
    .container {  
        box-shadow: 2px 2px 3px rgb(255, 199, 199);  
        padding: 20px;  
        h2 {  
            color:rgb(148, 9, 9);  
        }  
        p {  
            color:rgb(252, 89, 89);  
        }  
    }  
} @else if $themeColor==green {  
    .container {  
        box-shadow: 2px 2px 3px rgb(88, 245, 15);  
        padding: 20px;  
        h2 {  
            color:rgb(3, 179, 70);  
        }  
        p {  
            color:rgb(8, 255, 82);  
        }  
    }  
}  
  
@if $borderSize>=15 {  
    .container {  
        border: $borderSize solid black;  
        h2 {  
            border:$borderSize - 10 solid black;  
        }  
    }  
} @else {  
    .container {  
        border: 2px solid black;  
        h2 {  
            border:none;  
        }  
    }  
}  
  
                       @for  
- It configures a loop or iterator  
- loop executes set of statements repeatedly until the condition is satisfied.  
- With a set of logic you can perform batch operations.  
  
Syntax:  
  @for  variable  from  startPoint  through  endPoint  
   {  
  
    }  
  
  @for  $i  from 1  through 10  
   {  
  
   }  
  
- Dynamic expressions are defined by using  "#{ }"  
  
        string#{variable}  
  
        #{ }         data binding expression [sass]  
        ${ }        javascript binding  
        { }        react binding  
        {{ }}        angular binding  
  
  
Ex: effects.scss  
  
  
@for $i from 1 through 5 {  
    .box-#{$i} {  
        margin-left: $i \* 100px;  
    }  
    .para-#{$i} {  
        font-size: 20px \* $i;  
    }  
}  
  
     home.html  
  
<!DOCTYPE html>  
<html>  
    <head>  
        <title>Keyframes</title>  
        <link rel="stylesheet" href="../sass/effects.css">  
        <style>  
            div {  
                border:1px solid black;  
                width: 100px;  
            }  
        </style>  
    </head>  
    <body>  
       <div class="box-1">  
           <img src="../public/images/jacket.jpg" width="100" height="100">  
       </div>  
       <div class="box-2">  
        <img src="../public/images/backpack.jpg" width="100" height="100">  
    </div>  
    <div class="box-3">  
        <img src="../public/images/shoe.jpg" width="100" height="100">  
    </div>  
    <div class="box-4">  
        <img src="../public/images/neckband.PNG" width="100" height="100">  
    </div>  
    <div class="box-5">  
        <img src="../public/images/shoe1.jpg" width="100" height="100">  
    </div>  
     <p class="para-1">Para-1</p>  
     <p class="para-2">Para-2</p>  
     <p class="para-3">Para-3</p>  
    </body>  
</html>  
  
@function  
@mixin  
  
@function          : It is used for refactor.                      
                  It is a technique of encapsulating a set of  
                  statements into an entity.  
                  You can re-use the logic, without re-writing.                           Function returns a value.  
  
 @mixin                : It is similar to function but will not return value.  
  
  
 Syntax:  
        @mixin  name(params) {  
  
         }  
  
        @include name(values);  
  
  
Ex:  
effects.scss  
  
@mixin ImageEffect($start, $end) {  
    @if $start>=3 {  
        .box-1 {  
            float: left;  
        }  
        .box-2 {  
            float: right;  
        }  
    }  
     
        @for $i from $start through $end {  
            .box-#{$i} {  
                margin-left: $i \* 100px;  
            }  
        }  
  
}  
@include ImageEffect(3,5);  
  
  
                     @function  
- It can perform the functionality  
- It can store the result  
  
  
@function  name (params)  
{  
   @return  value;  
}  
  
Ex: effects.scss  
  
@function SetBorder($size) {  
    @return  $size solid red;  
}  
  
div {  
    border:SetBorder(2px);  
}  
  
                         Bundling