



inicio

Temario

05

Frameworks CSS + Bootstrap

- ✓ Librerías y Frameworks
- ✓ Bootstrap
- ✓ Estructura de Bootstrap

06

SASS

- ✓ Procesadores CSS
- ✓ Sintaxis SASS
- ✓ Operadores, condicionales y bucles
- ✓ Maps

07

Animaciones

- ✓ Gradientes
- ✓ Transformaciones
- ✓ Transiciones
- ✓ Animaciones

Mapa de conceptos



Definición

Qué es SASS: “Syntactically Awesome Stylesheets”. Permite crear hojas de estilos estructuradas, limpias y fáciles de mantener.

SASS permite escribir hojas de estilo para generar ficheros CSS más optimizados, incorporando mayor contenido semántico.

Instalación

Preparando las herramientas y el entorno de trabajo

1

Instalar nodejs

2

En el directorio del proyecto de VSC.
Crear una nueva carpeta SCSS, y dentro un archivo styles.scss.

3

Desde el terminal
Ejecutar el comando `npm init` que creará el archivo package.json

Instalación

Preparando las herramientas y el entorno de trabajo

4

Instalar SASS. comando
`npm install -g sass`

5

Para compilar el scss,
ejecutar el comando
`sass --watch scss:css`

Para detener la
compilación presionar:
`Ctrl+C`

Instalación Alternativa

Si su PC no soporta Node, puede emplear la extensión de VSC para compilar SASS. Se llama **Live SASS Compiler**



Sintaxis SASS

Nesting

```
1  /* CSS */
2  ul {
3    list-style: none;
4  }
5  ul li {
6    padding: 15px;
7    display: inline-block;
8  }
9  ul li a {
10   text-decoration: none;
11   font-size: 16px;
12   color: #444;
13 }
```

```
1  // SCSS
2  ul {
3    list-style: none;
4    li {
5      padding: 15px;
6      display: inline-block;
7      a {
8        text-decoration: none;
9        font-size: 16px;
10       color: #444;
11      }
12    }
13 }
```

A diferencia de HTML que se emplea una **anidación estricta**, CSS no exige una estructura que permita controlar el caos .

Con la anidación de SASS, los estilos conservan una estructura semejante a HTML.

Sintaxis SASS

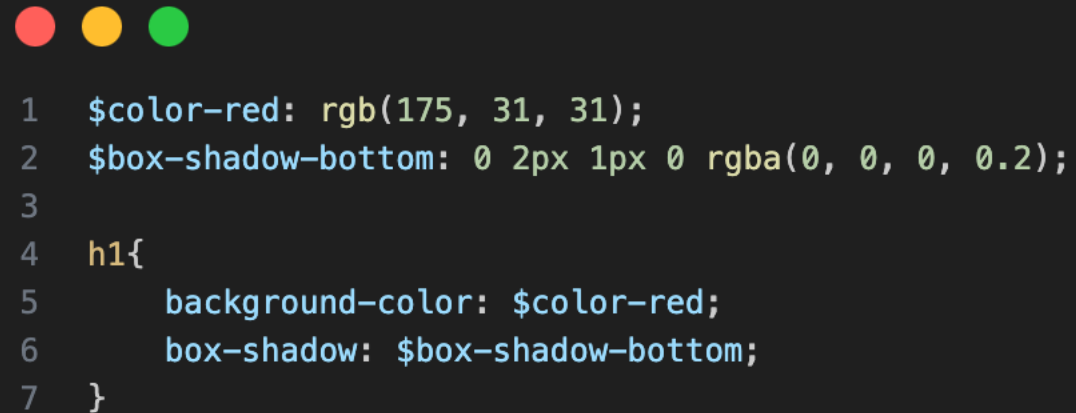
& + Nesting

```
1  h1{  
2      background-color: blue;  
3      &:hover{  
4          background-color: yellow;  
5      }  
6  }
```

El ampersand (**&**) se emplea para anidar permitiendo hacer referencia al selector externo. Permitiendo emplear el selector en situaciones más estructuradas como agregar una pseudoclase.

Sintaxis SASS

Variables



```
1  $color-red: rgb(175, 31, 31);  
2  $box-shadow-bottom: 0 2px 1px 0 rgba(0, 0, 0, 0.2);  
3  
4  h1{  
5      background-color: $color-red;  
6      box-shadow: $box-shadow-bottom;  
7  }
```


Sintaxis SASS

Operaciones

```
1 // SCSS
2 $ancho: 750px;
3
4 h1{
5     width: $ancho/2+50;
6 }
```

```
1 /* css */
2 h1 {
3     width: 425px;
4 }
```

Sintaxis SASS

@import

Se pueden crear estilos especializados en archivos independientes como si fueran librerías y luego pueden ser importados.

Nota: El nombre del archivo scss que se importa debe iniciar con “_”

_underline.scss

```
1 @import "colores";  
2 h1{  
3     color: $color-green;  
4 }
```

Importa el archivo _colores.scss

Sintaxis SASS

Placeholder Class (%) + Directiva @extend

Las clases Placeholder permiten generar conjuntos de estilos aplicables a los elementos, destacando que si en algún momento no se emplean la clase con los estilos se conservan en el scss pero no se incluye en el css, evitando incluir estilos innecesarios.

```
1 // placeholder class
2 %box {
3     padding: 10px;
4     font-size: 1.2em;
5 }
6 // directiva @extend
7 .caja {
8     @extend %caja;
9 }
```

Estructuras

Selector de elementos a partir de variables `#{$variable}`.




```
1 <!-- HTML -->
2 <div id="box-1">HOLA MUNDO</div>
```




```
1 SCSS
2 $i: 0;
3
4 #box-#{ $i+1 }{
5     background-color: red;
6 }
```


Estructuras


Condicionales << if >>. << if-else >>. << if-else-if >>



```
1 // SCSS
2 $i: 0;
3
4 @if $i==0{
5     #box-#{$i+1}{
6         background-color: red;
7     }
8 }
```



```
1 $nombre: juan;
2 $i: 0;
3
4 @if $nombre==maria{
5     #box-#{$i+1}{
6         background-color: red;
7     }
8 }@else{
9     #box-#{$i+2}{
10         background-color: red;
11     }
12 }
```



```
1 // SCSS
2 $nombre: juan;
3 $i: 0;
4
5 @if $nombre==maria{
6     #box-#{$i+1}{
7         background-color: red;
8     }
9 }@else if $nombre==juan{
10     #box-#{$i+2}{
11         background-color: red;
12     }
13 }
```

Estructuras

Ciclos << for + to || through >>



```
1 <!-- HTML -->
2 <div id="fila-1">uno</div>
3 <div id="fila-2">dos</div>
4 <div id="fila-3">tres</div>
5 <div id="fila-4">cuatro</div>
6 <div id="fila-5">cinto</div>
```



```
1 @for $i from 1 to 5{
2     #fila-#{ $i }{
3         @if $i%2==0{
4             background-color: red;
5         }@else{
6             background-color: blue;
7         }
8     }
9 }
```



```
1 #fila-1 {
2     background-color: blue;
3 }
4
5 #fila-2 {
6     background-color: red;
7 }
8
9 #fila-3 {
10     background-color: blue;
11 }
12
13 #fila-4 {
14     background-color: red;
15 }
```


Estructuras

Ciclos << **for** + **to** || **through** >> << **while** >>



```
1 <!-- HTML -->
2 <div id="fila-1">uno</div>
3 <div id="fila-2">dos</div>
4 <div id="fila-3">tres</div>
5 <div id="fila-4">cuatro</div>
6 <div id="fila-5">cinto</div>
```



```
1 @for $i from 1 through 5{
2     #fila-#{ $i }{
3         @if $i%2==0{
4             background-color: red;
5         }@else{
6             background-color: blue;
7         }
8     }
9 }
```



```
1 #fila-1 {
2     background-color: blue;
3 }
4 #fila-2 {
5     background-color: red;
6 }
7 #fila-3 {
8     background-color: blue;
9 }
10 #fila-4 {
11     background-color: red;
12 }
13 #fila-5 {
14     background-color: blue;
15 }
```

Estructuras

Ciclos << each >>



```
1  @each $nombre in juan, pedro, pablo{
2    .user-#{ $nombre }{
3      background-image: url("../img/#{ $nombre }.png");
4    }
5  }
```

Interpolación con una lista de elementos SassScript válida (sucesión de elementos separados por coma)



```
1  .user-juan {
2    background-image: url("../img/juan.png");
3  }
4  .user-pedro {
5    background-image: url("../img/pedro.png");
6  }
7  .user-pablo {
8    background-image: url("../img/pablo.png");
9  }
```


Estructuras



```
1 $usuarios: (  
2   juan:"./img/juan.jpg",  
3   pedro:"./img/pedro.jpg",  
4   pablo:"./img/pablo.jpg"  
5 );  
6 @each $usuario, $url in $usuarios{  
7   .user-#{ $usuario }{  
8     background-image: url("#{ $url }");  
9   }  
10 }
```

Es un tipo de dato similar al JSON, emplea un binomio **llave: valor**.

\$map: (llave1: valor1, llave2: valor2, ... , llaveN: valorN);

Empleando << Map >>



```
1 .user-juan {  
2   background-image: url("./img/juan.jpg");  
3 }  
4 .user-pedro {  
5   background-image: url("./img/pedro.jpg");  
6 }  
7 .user-pablo {  
8   background-image: url("./img/pablo.jpg");  
9 }
```

@mixin & @include