

Sass

Tecnologías Web

Realizado por: Jesús Martín Zorrilla

Índice

1. ¿Qué es SASS?	2
2. Instalación(Windows)	4
3. Preparación antes de programar	6
4. ¿Qué nos ofrece SASS?	8
5. Conclusión	19

1. ¿Qué es SASS?

Es un preprocesador CSS. Estos son herramientas para los desarrolladores de sitios web, que permiten traducir un código de hojas de estilo no estándar, específico del preprocesador en cuestión, a un código CSS estándar, entendible por los navegadores.

Los preprocesadores básicamente nos ofrecen diversas utilidades que a día de hoy no se encuentran en el lenguaje CSS.

Podríamos sin duda elegir otras alternativas como Less o Stylus y estaría estupendo para nosotros y nuestro proyecto, ya que al final todos ofrecen más o menos las mismas utilidades. Pero sin embargo, Sass se ha convertido en el preprocesador más usado y el más demandado.

Existen dos tipos de sintaxis para escribir su código:

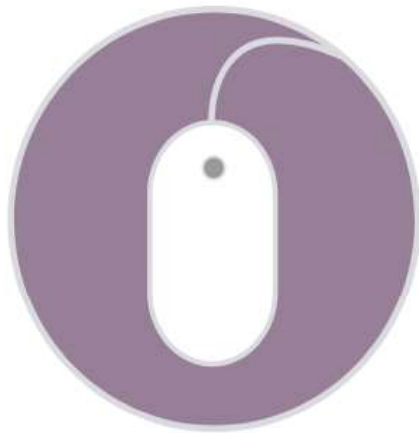
Sintaxis Sass: esta sintaxis es un poco diferente de la sintaxis de CSS estándar. No difiere mucho. Por ejemplo, te evita colocar puntos y coma al final de los valores de propiedades. Además, las llaves no se usan y en su lugar se realizan indentados.

Sintaxis SCSS: Es una sintaxis bastante similar a la sintaxis del propio CSS. De hecho, el código CSS es código SCSS válido. Podríamos decir que SCSS es código CSS con algunas cosas extra.

En la práctica, aunque podría ser más rápido escribir con sintaxis Sass, es menos recomendable, porque te aleja más del propio lenguaje CSS.

2. Instalación

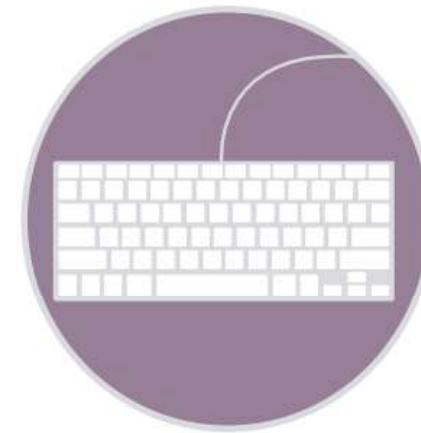
Applications



There are a good many applications that will get you up and running with Sass in a few minutes for Mac, Windows, and Linux. You can download most of the applications for free but a few of them are paid apps (and totally worth it).

- [CodeKit](#) (Paid) Mac
- [Ghostlab](#) (Paid) Mac Windows
- [Hammer](#) (Paid) Mac
- [LiveReload](#) (Paid, Open Source) Mac Windows
- [Prepros](#) (Paid) Mac Windows Linux
- [Scout-App](#) (Free, Open Source) Windows Linux Mac

Command Line



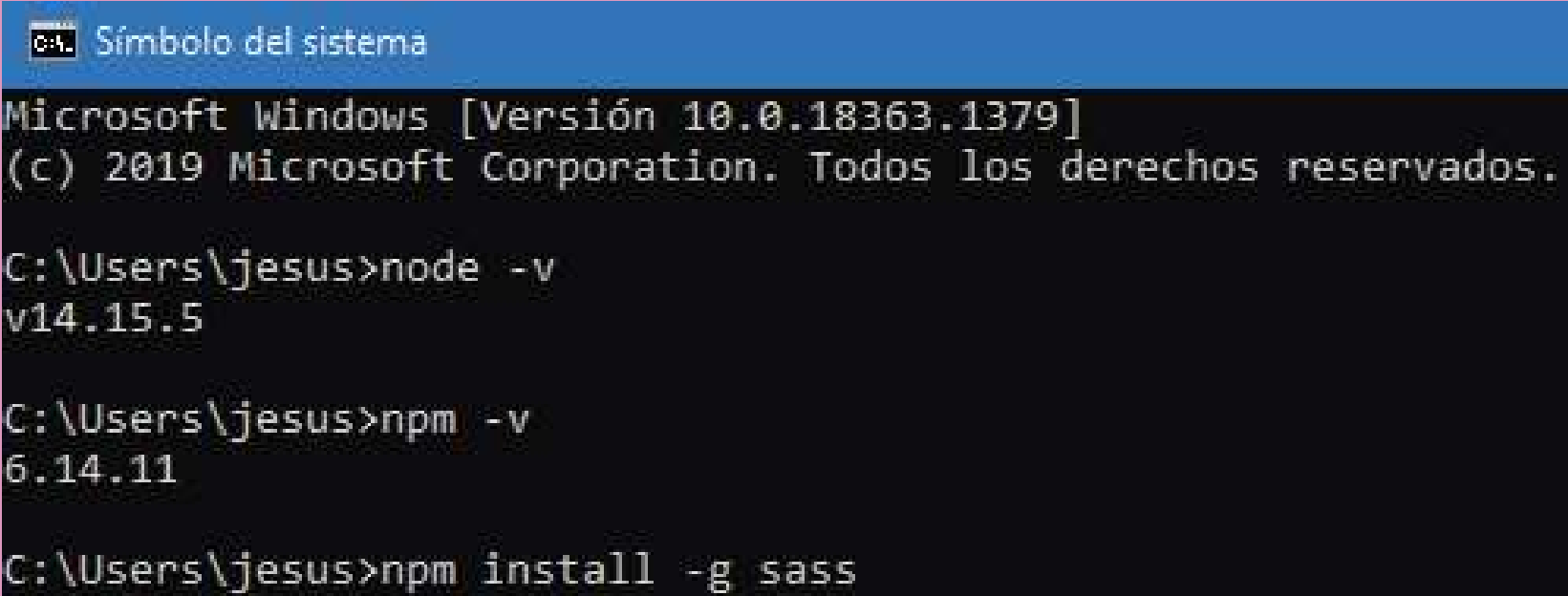
When you install Sass on the command line, you'll be able to run the `sass` executable to compile `.sass` and `.scss` files to `.css` files. For example:

```
sass source/stylesheets/index.scss build/stylesheets/index.css
```

First install Sass using one of the options below, then run `sass --version` to be sure it installed correctly. If it did, this will include `1.32.8`. You can also run `sass --help` for more information about the command-line interface.

Once it's all set up, **go and play**. If you're brand new to Sass we've set up some resources to help you learn pretty darn quick.

Descargamos nodejs desde su página oficial: <https://nodejs.org/es/>



```
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.18363.1379]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

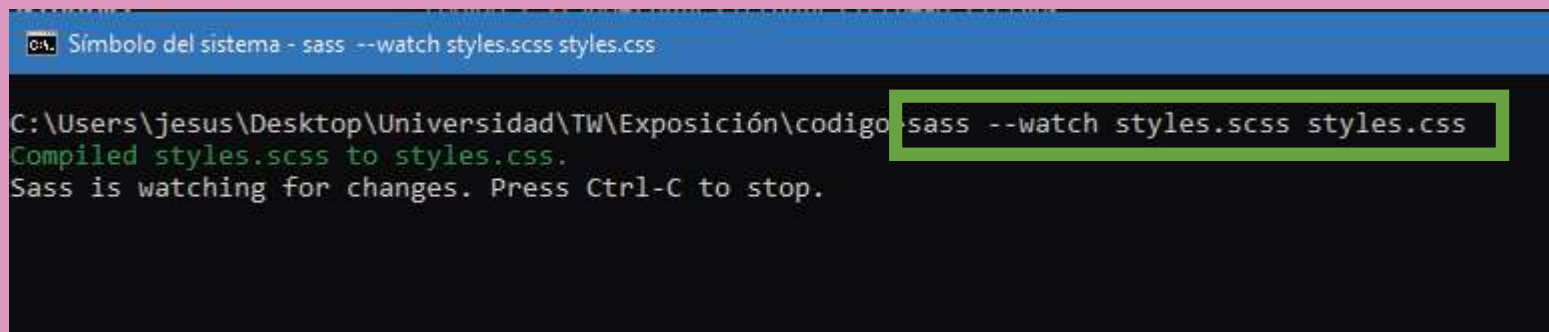
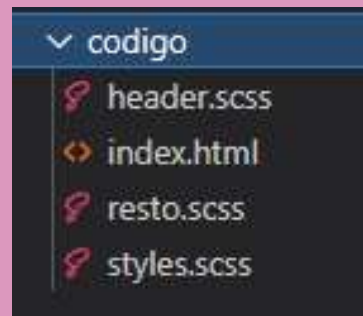
C:\Users\jesus>node -v
v14.15.5

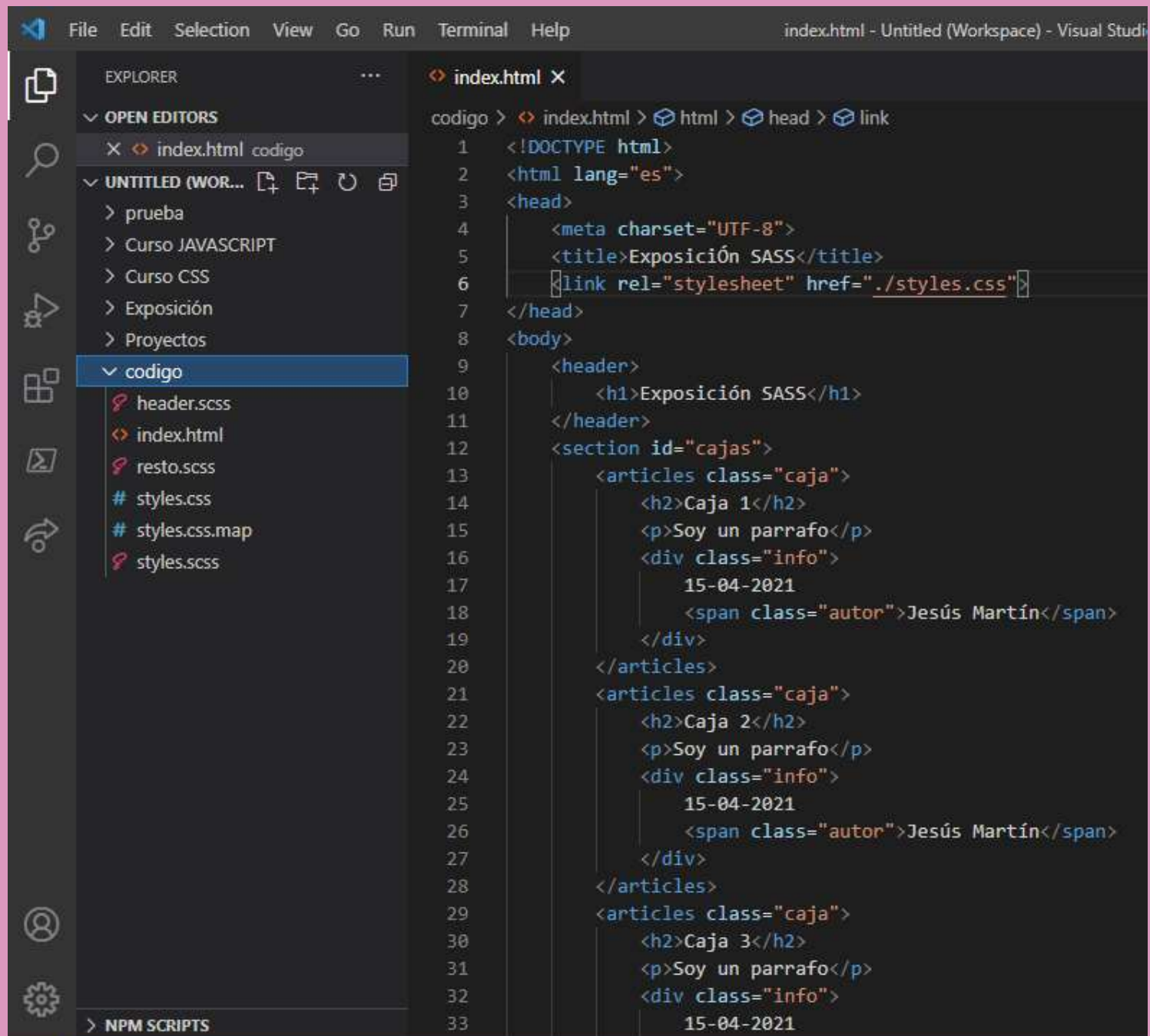
C:\Users\jesus>npm -v
6.14.11

C:\Users\jesus>npm install -g sass
```

Una vez hecho esto, ya tenemos instalado SASS.

3. Preparación antes de programar

A screenshot of a Windows command prompt window. The title bar reads 'Símbolo del sistema - sass --watch styles.scss styles.css'. The command prompt shows the following text: 'C:\Users\jesus\Desktop\Universidad\TW\Exposición\codigo>sass --watch styles.scss styles.css', 'Compiled styles.scss to styles.css.', and 'Sass is watching for changes. Press Ctrl-C to stop.' The command line is highlighted with a green box.



4. ¿Qué nos ofrece SASS?

1) Módulos

```
header.scss X
codigo > header.scss > header
1 header{
2   background-color: red;
3 }
```

```
resto.scss X
codigo > resto.scss > section#cajas
1 section#cajas{
2   background-color: green;
3 }
```

```
styles.scss X
codigo > styles.scss
1 //Modulos
2 @use "header";
3 @use "resto";
```



```
# styles.css X
codigo > # styles.css > header
1 header {
2   background-color: red;
3 }
4
5 section#cajas {
6   background-color: green;
7 }
8
9 /*# sourceMappingURL=styles.css.map */
10
```

2) Variables

```
header.scss X
codigo > header.scss > header
1 $color: rgb(255,100,58);
2 header{
3     text-align: center;
4     height: 100px;
5     line-height: 100px;
6     background-color: $color;
7 }
```



```
# styles.css X
codigo > # styles.css > ...
1 header {
2     text-align: center;
3     height: 100px;
4     line-height: 100px;
5     background-color: #ff643a;
6 }
7
8 section#cajas {
9     background-color: green;
10 }
11
12 /*# sourceMappingURL=styles.css.map */
13
```

3) Anidación

```
resto.scss X
codigo > resto.scss > #cajas > .caja
1  #cajas{
2      display: flex;
3      .caja{
4          font-size: 20px;
5          padding: 10px;
6          margin: 10px auto;
7          background-color: green;
8          .info{
9              font-size: 15px;
10         }
11     }
12 }
```



```
# styles.css X
codigo > # styles.css > #cajas .caja
1  header {
2      text-align: center;
3      height: 100px;
4      line-height: 100px;
5      background-color: #ff643a;
6  }
7
8  #cajas {
9      display: flex;
10 }
11 #cajas .caja {
12     font-size: 20px;
13     padding: 10px;
14     margin: 10px auto;
15     background-color: green;
16 }
17 #cajas .caja .info {
18     font-size: 15px;
19 }
20
21 /*# sourceMappingURL=styles.css.map */
22
```

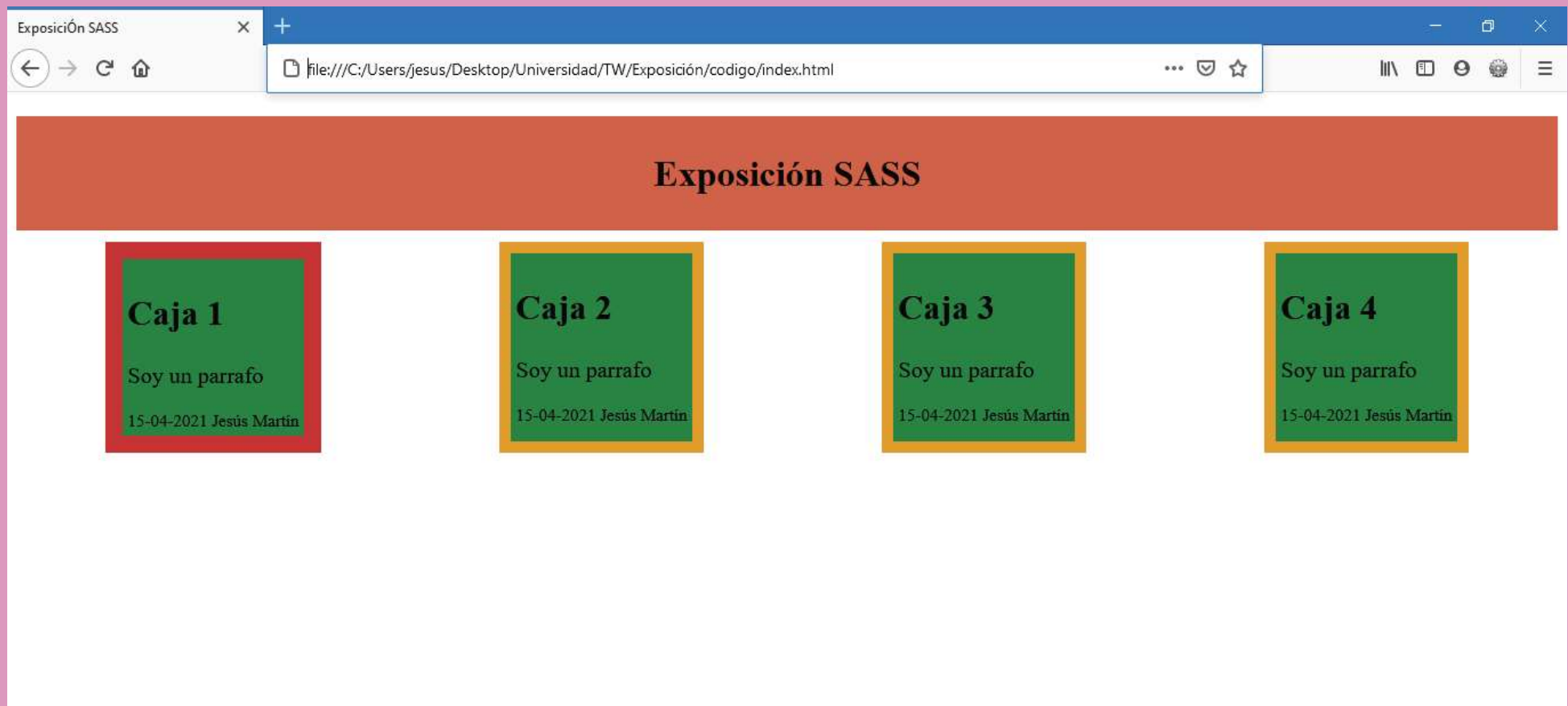
4) Mixins

```
resto.scss x
codigo > resto.scss > #cajas > .caja
1  @mixin crearBordes($tam, $colorBorde) {
2    border: $tam solid $colorBorde;
3  }
4  #cajas{
5    display: flex;
6    .caja{
7      font-size: 20px;
8      padding: 5px;
9      margin: 10px auto;
10     background-color: green;
11     @include crearBordes(10px, orange);
12     .info{
13       font-size: 15px;
14     }
15   }
16   .caja:first-child{
17     @include crearBordes(15px, red)
18   }
19 }
20
```



```
#cajas {
  display: flex;
}
#cajas .caja {
  font-size: 20px;
  padding: 5px;
  margin: 10px auto;
  background-color: green;
  border: 10px solid orange;
}
#cajas .caja .info {
  font-size: 15px;
}
#cajas .caja:first-child {
  border: 15px solid red;
}
```

Así se vería nuestra página por ahora:



5) Herencia

```
</section>
<div class="alert">
  Hola soy una alerta normal
</div>
<div class="alert-success">
  Hola soy alerta success
</div>
<div class="alert-danger">
  Hola soy alerta danger
</div>
</body>
</html>
```

```
resto.scss
codigo > resto.scss > .alert-danger
17   @include crearBordes(15px, red)
18   }
19 }
20 %alert-normal{
21   @include crearBordes(1px, gray);
22   background-color: #ccc;
23   color: gray;
24   padding: 50px;
25   width: 150px;
26   margin: 20px auto;
27   text-align: center;
28 }
29
30 .alert{
31   @extend %alert-normal;
32 }
33
34 .alert-success{
35   @extend %alert-normal;
36   color: green;
37   @include crearBordes(5px, green);
38 }
39
40 .alert-danger{
41   @extend %alert-normal;
42   color: red;
43   @include crearBordes(5px, red);
44 }
```



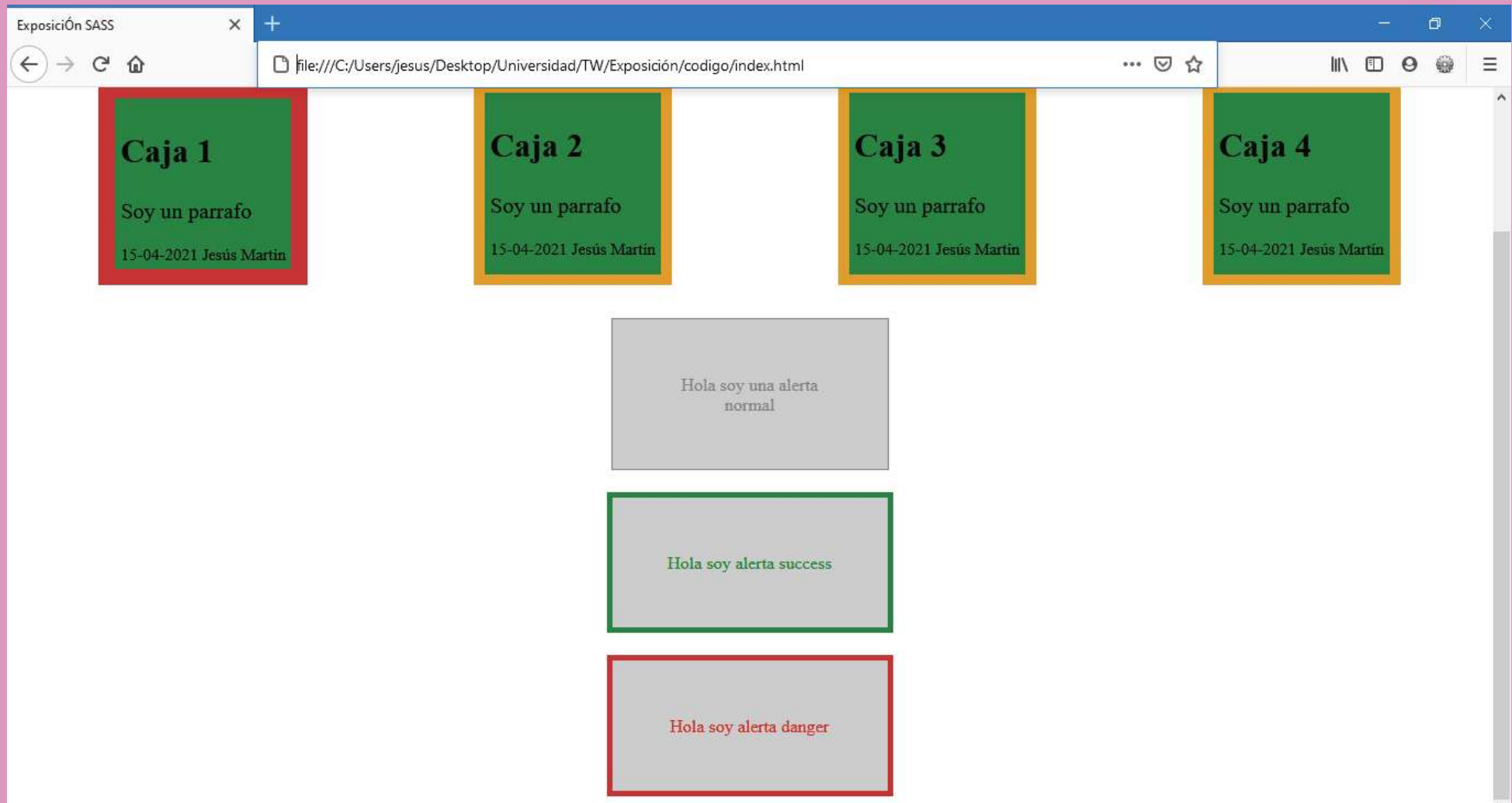
```
.alert-danger, .alert-success, .alert {
  border: 1px solid gray;
  background-color: #ccc;
  color: gray;
  padding: 50px;
  width: 150px;
  margin: 20px auto;
  text-align: center;
}

.alert-success {
  color: green;
  border: 5px solid green;
}

.alert-danger {
  color: red;
  border: 5px solid red;
}

/*# sourceMappingURL=styles.css.map */
```

Asi se vería nuestra página por ahora:



6) Condiciones

```
header.scss X
codigo > header.scss > header
1 $color: rgb(255,100,58);
2 header{
3   text-align: center;
4   height: 100px;
5   line-height: 100px;
6   background-color: $color;
7
8   @if $color == rgb(255,100,58){
9     color: white;
10  } @else if $color == red{
11    color: black;
12  } @else{
13    color: green;
14  }
15 }
```



```
# styles.css X
codigo > # styles.css > #cajas .caja
1 header {
2   text-align: center;
3   height: 100px;
4   line-height: 100px;
5   background-color: #ff643a;
6   color: white;
7 }
8
```


7) Bucles (Probaré el bucle 'for' y el 'while')

```
index.html X
codigo > index.html > html > body > div.listado
50     Hola soy alerta success
51 </div>
52 <div class="alert-danger">
53     Hola soy alerta danger
54 </div>
55 <div class="listado">
56     <h1>Listado de preprocesadores</h1>
57     <ul>
58         <li>
59             SASS
60         </li>
61         <li>
62             LESS
63         </li>
64         <li>
65             Stylus
66         </li>
67         <li>
68             PostCSS
69         </li>
70     </ul>
71 </div>
72
73 </body>
74 </html>
```

FOR

```
resto.scss x
codigo > resto.scss > ...
41   @extend %alert-normal;
42   color: red;
43   @include crearBordes(5px, red);
44 }
45
46 $contador: 1;
47
48 @for $contador from 1 through 4 {
49   .listado ul li:nth-child(#{$contador}){
50     color: orangered;
51     font-size: $contador * 10px ;
52   }
53 }
54
```

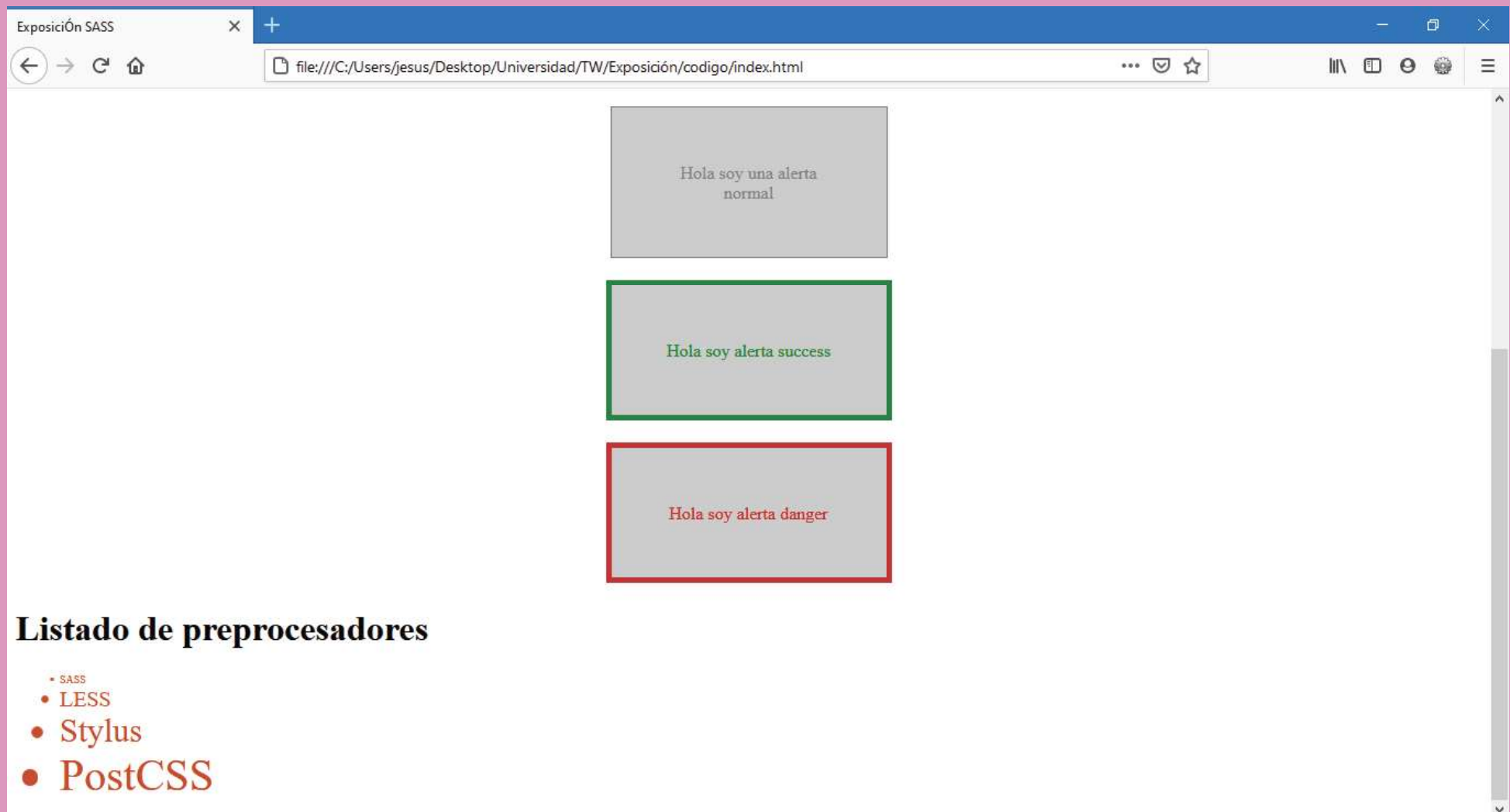
WHILE

```
resto.scss x
codigo > resto.scss > ...
41   @extend %alert-normal;
42   color: red;
43   @include crearBordes(5px, red);
44 }
45
46 $i: 1;
47 @while $i <= 4 {
48   .listado ul li:nth-child(#{$i}){
49     color: orangered;
50     font-size: $i * 10px ;
51   }
52   $i: $i + 1;
53 }
```



```
# styles.css x
codigo > # styles.css > #cajas.caja
40
41 .alert-danger {
42   color: red;
43   border: 5px solid red;
44 }
45
46 .listado ul li:nth-child(1) {
47   color: orangered;
48   font-size: 10px;
49 }
50
51 .listado ul li:nth-child(2) {
52   color: orangered;
53   font-size: 20px;
54 }
55
56 .listado ul li:nth-child(3) {
57   color: orangered;
58   font-size: 30px;
59 }
60
61 .listado ul li:nth-child(4) {
62   color: orangered;
63   font-size: 40px;
64 }
65
66 /*# sourceMappingURL=styles.css.map */
67
```

Así se vería nuestra página por ahora:



4. Conclusión

Usar un preprocesador de CSS como SASS reduce el tiempo para crear archivos CSS, permite tener una organización modular de los estilos de tu web, proporciona escrituras propias de lenguajes de programación como variables, funciones, bucles, etc.

Consejo: A nivel personal, esta ayuda que nos ofrece SASS se mejora exponencialmente si usas una metodología, por ejemplo: BEM.

