

Parte 3 de 4: Sass y Compass

EXTENDS
MEDIA QUERIES CON SASS
SPRITES

Apuntes de: **Rosa María Medina Gómez**

Adaptados a Power Point por José Jesús Torregrosa García

Curso de Formación del Profesorado a distancia
Cefire Específico de FP de Cheste

Generalitat Valenciana Curso 2018 - 2019









Introducción

En esta unidad vamos a ver qué son y cómo usar un **extend**, **media query**, **sprite**, y las **funciones matemáticas** y **colores** (gradientes, filtros y transformaciones).

EXTENDS

En la unidad anterior, vimos que el CSS de la izquierda lo podemos transformar en el de la derecha:

```
8  .btn-a{
9      background:  #777;
10     border: 1px solid  #ccc;
11     font-size: 1em;
12     text-transform: uppercase;
13 }
14 .btn-b{
15     background:  #ff0;
16     border: 1px solid  #ccc;
17     font-size: 1em;
18     text-transform: uppercase;
19 }
```

```
2  .btn-a, .btn-b{
3      background:  #777;
4      border: 1px solid  #ccc;
5      font-size: 1em;
6      text-transform: uppercase;
7  }
```

EXTENDS

Mediante **Sass** podemos combinar estos selectores usando la directiva **extend**:

```
_buttons.scss ●  
1  .btn-a {  
2      background: #777;  
3      border: 1px solid #ccc;  
4      font-size: 1em;  
5      text-transform: uppercase;  
6  }  
7  .btn-b {  
8      @extend .btn-a;  
9      background: #ff0;  
10 }
```

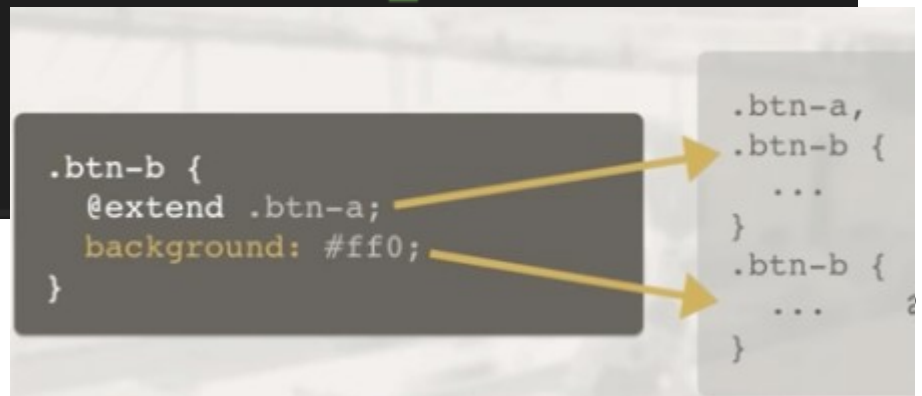
```
application.scss x  
1  @import "_buttons";
```

EXTENDS

Con esto conseguimos lo siguiente:

application.css x

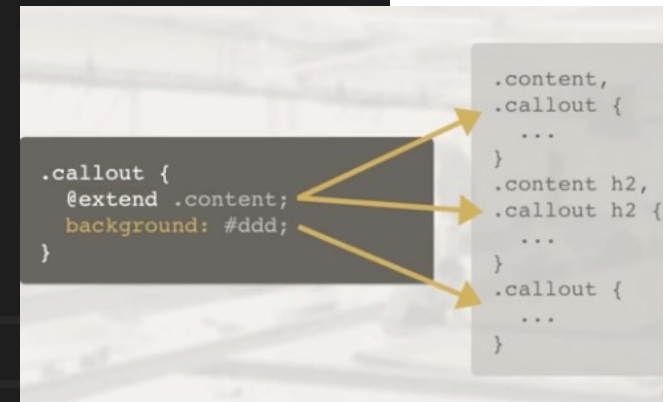
```
1  /* line 1, ../../sass/teoriaSesion3/_buttons.scss */
2  .btn-a, .btn-b {
3      background: #777;
4      border: 1px solid #ccc;
5      font-size: 1em;
6      text-transform: uppercase;
7  }
8
9  /* line 7, ../../sass/teoriaSesion3/_buttons.scss */
10 .btn-b {
11     background: #ff0;
12 }
```



EXTENDS

¿Qué es lo que ocurriría si usamos selectores **nested** y usamos **extend**?

```
application.scss
1  .content{
2      border: 1px solid #ccc;
3      padding: 20px;
4      h2{
5          font-size: 3em;
6          margin: 20px 0;
7      }
8  }
9  .callout{
10     @extend .content;    background: #ddd;
11 }
```



EXTENDS

En este ejemplo, estamos extendiendo `.content` en nuestra declaración, pero `.content` tiene un bloque para los `h2` dentro. Cuando compilamos el código anterior, no solo se añade lo que contiene `.content` a `.callout`, sino que también hereda el `h2`.

```
# application.css x
1  /* line 1, ../../sass/teoria
2  .content, .callout {
3    border: 1px solid #ccc;
4    padding: 20px;
5  }
6  /* line 4, ../../sass/teoria
7  .content h2, .callout h2 {
8    font-size: 3em;
9    margin: 20px 0;
10 }
11
12 /* line 9, ../../sass/teoria
13 .callout {
14   background: #ddd;
15 }
```

EXTENDS

extend es bastante útil ya que podemos combinar selectores en nuestra hoja de estilos. Pero tiene algunos problemas. Por ejemplo:

```
_buttons.scss x
1  .btn-a{
2      background: #777;
3      border: 1px solid #ccc;
4      font-size: 1em;
5      text-transform: uppercase;
6  }
7  .btn-b{
8      @extend .btn-a;
9      background: #ff0;
10 }
11 .sidebar .btn-a{
12     text-transform: lowercase;
13 }
```

```
application.scss x
1  @import "_buttons";
```


EXTENDS

Al heredar `.btn-b` las propiedades de `.btn-a`, las hereda tanto del selector `.btn-a` como también del selector `.sidebar .btn-b` (cuando `.btn-a` está dentro de `.sidebar`), por lo que también se creará una regla idéntica para cuando `.btn-b` esté dentro de `.sidebar`.

```
# application.css x
1  /* line 1, ../../sass/teoriaSesion3
2  .btn-a, .btn-b {
3      background: #777;
4      border: 1px solid #ccc;
5      font-size: 1em;
6      text-transform: uppercase;
7  }
8
9  /* line 7, ../../sass/teoriaSesion3
10 .btn-b {
11     background: #ff0;
12 }
13
14 /* line 11, ../../sass/teoriaSesion
15 .sidebar .btn-a, .sidebar .btn-b {
16     text-transform: lowercase;
17 }
```

EXTENDS

Por tanto, desde el momento en el que el .btn-b extiende de .btn-a, cualquier cambio que hagas desde .btn-a usando otros selectores afectará a .btn-b. Esto lo podemos solventar usando selectores "*placeholder*" como veremos a continuación.

EXTENDS Y PLACEHOLDER

Los selectores *placeholder* en Sass están precedidos por el símbolo del porcentaje (%), además, pueden ser extendidos, pero nunca serán un selector en sí mismos. Si volvemos al ejemplo anterior de los botones y usamos un *placeholder* de forma que las partes comunes las agrupamos y hacemos que .btn-a y .btn-b extiendan del *placeholder*.

EXTENDS Y PLACEHOLDER

_buttons.scss x

```
1 %btn{
2   background: #777;
3   border: 1px solid #ccc;
4   font-size: 1em;
5   text-transform: uppercase;
6 }
7 .btn-a{
8   @extend %btn;
9 }
10 .btn-b{
11   @extend %btn; background: #ff0;
12 }
13
14 .sidebar .btn-a{
15   text-transform: lowercase;
16 }
```



application.css x

```
1 /* line 1, ../../sass/teori
2 .btn-a, .btn-b {
3   background: #777;
4   border: 1px solid #ccc;
5   font-size: 1em;
6   text-transform: uppercase;
7 }
8
9 /* line 10, ../../sass/teori
10 .btn-b {
11   background: #ff0;
12 }
13
14 /* line 14, ../../sass/teori
15 .sidebar .btn-a {
16   text-transform: lowercase;
17 }
```

application.scss x

```
1 @import "_buttons";
```

EXTENDS Y PLACEHOLDER

Los selectores **placeholder** son útiles cuando queremos reusar estos estilos CSS :

```
application.scss x
1 @import "_buttons";
```

```
_buttons.scss x
1 %ir{
2   border: 0;
3   font: 0/0 a; // anulamos la fuente
4   text-shadow: none;
5   color: transparent;
6   background-color: transparent;
7 }
8 .logo{
9   @extend %ir;
10 }
11 .social{
12   @extend %ir;
13 }
```



```
# application.css x
1 /* line 1, ../../sass/teoriaSesi
2 .logo, .social {
3   border: 0;
4   font: 0/0 a;
5   text-shadow: none;
6   color: transparent;
7   background-color: transparent;
8 }
```

EXTENDS Y PLACEHOLDER

Por último, antes de acabar este punto, decir que debemos tener **cuidado** ya que las **versiones anteriores a IE9** tiene el límite de 4095 selectores CSS por archivo, por tanto, si usamos muchos extends o muchos imports puede que alcancemos el total de 4095 selectores que tiene como límite, si se llega a sobrepasar ese límite todo lo que se sobrepase será ignorado.

Math + Color

Operaciones aritméticas básicas:

Con Sass tenemos la posibilidad de realizar operaciones con **números** (+, -, *, /, %). Por defecto, cuando realizamos una operación matemática en Sass nos devuelve, si es decimal, 5 dígitos tras la coma redondeados hacia arriba




```
font: normal 2em/1.5 Helvetica, sans-serif;
```

Math + Color

Concatenación de strings:

La forma de concatenar string en Sass es mediante el operador **+**. Cuando concatenamos **string**, si el operando de la izquierda tiene comillas simples, el resultado es un **string** con comillas simples. Sin embargo, si no va entre comillas, el resultado será sin ellas (aunque el otro operando las tenga).



```
4 fuente1{
5   |   font: normal 2em/1.5 Helvetica, sans-serif;
6   | }
7 fuente2{
8   |   font: "Helvetica " + "Neue"; // "Helvetica Neue";
9   | }
10 fuente3{
11   |   font: 'sans-' + serif; // 'sans-serif'
12   | }
```


Math + Color

Cuando realizamos operaciones en Sass donde las unidades entre las operaciones que realizamos varían como, por ejemplo:

```
application.scss x
1  h2{
2    font-size: 10px + 4pt;
3  }
```

Aquí estamos sumando píxeles y puntos. Sass nos hace la conversión de forma que tras compilarlo obtenemos:

```
# application.css x
1  /* line 1, ../../sass/tec
2  h2 {
3    font-size: 15.33333px;
4  }
```

Math + Color

Si usamos por ejemplo unidades relativas como **em**, no es capaz de hacer esta conversión y nos devuelve un error diciéndonos que no son compatibles las unidades:

```
application.scss x
1  h1{
2  |   font-size: 10px + 4em;
3  }
```

```
# application.css x
1  /*
2  Error: Incompatible units: 'em' and 'px'.
3  on line 2 of /home/jose/Escritorio/
```

```
error sass/teoriaSesion3/application.scss (Line 2: Incompatible units: 'em' and 'px'.)
Compilation failed in 1 files.
```

Math + Color

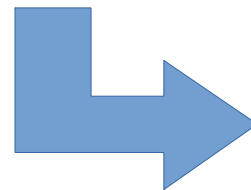
Funciones matemáticas

- **round(\$number)**: Redondea al número entero más cercano
- **ceil(\$number)**: Redondea hacia arriba
- **floor(\$number)**: Redondea hacia abajo
- **abs(\$number)**: Obtiene el valor absoluto
- **min(\$list)**: Obtiene el número más pequeño de una lista
- **max(\$list)**: Obtiene el número más grande de una lista
- **percentage(\$number)**: Convierte el número en un porcentaje

Math + Color

¿Cómo podemos utilizar estas funciones?

```
application.scss x
1 $context: 1000px;
2 h2{
3   | line-height: ceil(1.2);
4 }
5 .sidebar{ |
6   | width: percentage(350px/$context);
7 }
```



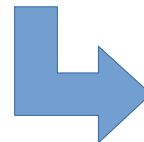
```
# application.css x
1  /* line 2, ../../s
2  h2 {
3    | line-height: 2;
4  }
5
6  /* line 5, ../../s
7  .sidebar {
8    | width: 35%;
9  }
```

Math + Color

Funciones matemáticas + Colores

Estas funciones matemáticas simplifican bastante los cambios de colores de los elementos de una web, por ejemplo aquí abajo vemos una suma en hexadecimal:

```
application.scss x
1  $color-base: #333;
2  .addition{
3      background: $color-base + #123;
4  }
```



```
# application.css x
1  /* line 2, ../../sass/tec
2  .addition {
3      background: #445566;
4  }
```

Math + Color

De la misma forma, podríamos haber restado, multiplicado o dividido colores:

```
application.scss x
1  $color-base: #333;
2  .subtraction{
3    background: $color-base - #123;
4  }
5
6  .multiplication{
7    background: $color-base *2;
8  }
9  .division{
10   background: $color-base /2;
11 }
```



```
# application.css x
1  /* line 2, ../../sass/tec
2  .subtraction {
3    background: #221100;
4  }
5
6  /* line 6, ../../sass/tec
7  .multiplication {
8    background: #666666;
9  }
10
11 /* line 9, ../../sass/tec
12 .division {
13   background: #191919;
14 }
```

Math + Color

Funciones de colores

Dentro de este rango de funciones podemos encontrar:

- **lighten**(\$color, \$amount): Genera un color más claro
- **darken**(\$color, \$amount): Genera un color más oscuro
- **saturate**(\$color, \$amount): Modifica la intensidad del color (añadiéndole)
- **desaturate**(\$color, \$amount): Modifica la intensidad del color (quitándole)
- **mix**(\$color1, \$color2, [\$weight]): Mezcla dos colores, el tercer parámetro es opcional y lo que hace es indicar el % del primer color que usamos
- **grayscale**(\$color): Convierte un color en escala de grises
- **invert**(\$color): Devuelve el inverso de un color
- **complement**(\$color): Devuelve el complementario.

Math + Color

application.scss x

```
1  $color-base: #333;
2  $color-escala: #87bf64;
3  $white: white;
4  $black: black;
5  $dark-text: #2e3135;
6  $offwhite: #f7f8f8;
7  $error: #e32908;
8
9  .mixed1 {
10 |   color: mix($white, $black);
11 | }
12  .mixed2 {
13 |   color: mix($black, $error);
14 | }
15  .mixed3{
16 |   color: mix($dark-text, $offwhite,
17 | }
18  .lighten{
19 |   color: lighten($color-base, 20%);
20 | }
```

```
20 | }
21  .darken{
22 |   color: darken($color-base, 20%);
23 | }
24  .saturate{
25 |   color: saturate($color-base, 20%);
26 | }
27  .desaturate{
28 |   color: desaturate($color-base, 20%);
29 | }
30  .grayscale{
31 |   color: grayscale($color-escala);
32 | }
33  .invert{
34 |   color: invert($color-escala);
35 | }
36  .complement{
37 |   color: complement($color-escala);
38 | }
```


Math + Color

```
application.scss x
38  }
39
40
41  $color-base: #333;
42  .subtraction{
43  |    background: $color-base - #123;
44  }
45
46  .multiplication{
47  |    background: $color-base *2;
48  }
49  .division{
50  |    background: $color-base /2;
51  }
52  $color-base: #333;
53  .addition{
54  |    background: $color-base + #123;
55  }
```

Responsive

El diseño **responsive** se ha convertido en una práctica muy común. Uno ejemplos típico es según las características del dispositivo utilizar un estilo u otro. A continuación veremos una **una media query**

```
application.scss x
1  .sidebar{
2      border: 1px solid #ccc;
3  }
4  @media(min-width: 700px){
5      // Para los dispositivos
6      // con al menos 700px de ancho
7      .sidebar{
8          float: right;
9          width: 30%;
10     }
11 }
```

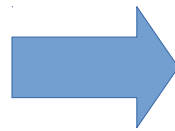


```
# application.css x
1  /* line 1, ../../sass/teorias
2  .sidebar {
3      border: 1px solid #ccc;
4  }
5
6  @media (min-width: 700px) {
7      /* line 7, ../../sass/teori
8      .sidebar {
9          float: right;
10         width: 30%;
11     }
12 }
```

Responsive

Podemos combinar fácilmente las *media queries* con *mixins* utilizando la *directiva @content*. Lo que hacemos con esta directiva es incluir un bloque de propiedades dentro un mixin.

```
application.scss x
1  @mixin respond-to{
2      @media(min-width: 700px){
3          @content
4      }
5  }
6  .sidebar{
7      border: 1px solid #ccc;
8      @include respond-to{
9          float: right;
10         width: 30%;
11     }
12 }
```



```
# application.css x
1  /* line 6, ../../sass/teoria
2  .sidebar {
3      border: 1px solid #ccc;
4  }
5  @media (min-width: 700px) {
6      /* line 6, ../../sass/teoria
7      .sidebar {
8          float: right;
9          width: 30%;
10     }
11 }
```

Responsive

Una de las prácticas más frecuentes en estos casos es enviar al mixin un argumento sobre el media, de forma que se realice esta comprobación, por ejemplo, aplicar un estilo sólo si estamos usando una tablet:

```
application.scss x
1  @mixin respond-to($media){
2      @if $media == tablet{
3          @media(min-width: 700px){
4              @content
5          }
6      }
7  }
8  .sidebar{
9      border: 1px solid #ccc;
10     @include respond-to(tablet){
11         float: right;
12         width: 30%;
13     }
14 }
```

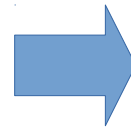


```
# application.css x
1  /* line 8, ../../sass/teoria
2  .sidebar {
3      border: 1px solid #ccc;
4  }
5  @media (min-width: 700px) {
6      /* line 8, ../../sass/teoria
7      .sidebar {
8          float: right;
9          width: 30%;
10     }
11 }
```

Responsive

En esta declaración podríamos añadir un **else o else-if**, y **eso no es flexible**. Abajo conseguimos que se incluya si se cumple con el ancho mínimo, de forma que podamos usar este mixin en cualquier lugar con un estilo para un min-width.:

```
application.scss x
1  @mixin respond-to($query){
2      @media(min-width: $query){
3          @content
4      }
5  }
6  .sidebar{
7      border: 1px solid #ccc;
8      @include respond-to(900px){
9          float: right;
10         width: 30%;
11     }
12 }
```



```
# application.css x
1  /* line 6, ../../sass/teorias
2  .sidebar {
3      border: 1px solid #ccc;
4  }
5  @media (min-width: 900px) {
6      /* line 6, ../../sass/teorias
7      .sidebar {
8          float: right;
9          width: 30%;
10     }
11 }
12
```

Responsive

Si queremos que por ejemplo este mixin no sólo se use para un mínimo, sino que pueda ser para un máximo también, la posibilidad que tenemos para modificarlo, sería:

```
application.scss x
1  @mixin respond-to($val, $query){
2      @media($val: $query){
3          @content
4      }
5  }
6  .sidebar{
7      border: 1px solid #ccc;
8      @include respond-to(max-width, 600px){
9          float: right;
10         width: 30%;
11     }
12 }
```

```
# application.css x
1  /* line 6, ../../sass/teori
2  .sidebar {
3      border: 1px solid #ccc;
4  }
5  @media (max-width: 600px) {
6      /* line 6, ../../sass/teo
7      .sidebar {
8          float: right;
9          width: 30%;
10     }
11 }
```



Sprites

En el siguiente punto vamos a ver cómo y cuándo usar *sprites*, además de la importancia que tienen.

Los *sprites* nos permiten combinar imágenes en un solo archivo, de forma que reducimos la cantidad de peticiones **http** que realizamos. Además, los *sprites* utilizan **background-position** para cambiar la imagen y sólo mostrar la porción que necesitamos. Hay numerosas páginas web que utilizan *sprites* como youtube.

Sprites

Si por ejemplo vamos a usar estos tres iconos donde, como podemos ver, cada uno tiene un determinado tamaño y todos están almacenados en el directorio de **images/icons**:



`images/icons/
cancel.png
166 x 166`



`images/icons/
next.png
113 x 185`



`images/icons/
warning.png
185 x 169`

Sprites

Para usar estos iconos, utilizamos la función **sprite-map** indicándole la ruta de todas las imágenes que representan nuestros iconos.

Esto nos generará un nuevo archivo de imagen con todos los iconos dentro.



Sprites

Además, vamos a utilizar el **elemento <i>** de HTML como contenedor de dichos iconos, poniéndole como imagen de fondo, el archivo que se genera automáticamente.

Para recorrer los iconos disponibles y generar una clase por cada nombre de icono debemos utilizar la directiva **@each** para recorrer la lista que a su vez nos proporciona la función **sprite_names**. En cada clase generada necesitamos las coordenadas de cada icono, devueltas por la función **sprite-position**, y las dimensiones, que nos proporciona **@include sprite-dimensions**.

Sprites

```
@import "compass/utilities/sprites";  
//compilar con "compass compile"  
$icons: sprite-map("../img/*.png");  
i{  
  background: $icons;  
  display: inline-block;  
}  
@each $i in sprite_names($icons) {  
  .icon-#{$i} {  
    background-position: sprite-position($icons, $i);  
    @include sprite-dimensions($icons, $i);  
  }  
}
```

Sprites

Aunque la directiva `@each` será algo que veamos en la siguiente unidad, quiero hacer un pequeño hincapié en explicaros cómo funciona:

```
$each $iterador in $lista{  
  // TO-DO something
```

<http://compass-style.org/help/tutorials/spriting/customizationoptions/>

(Nota: <http://stackoverflow.com/questions/15511874/file-toimport-not-found-or-unreadable-compass>)

```
}
```

Bibliografía y/o páginas de interés

- Sass: <http://sass-lang.com/>
- Ruby: <https://www.ruby-lang.org/en/>
- Sass Wikipedia: [https://en.wikipedia.org/wiki/Sass_\(stylesheet_language\)](https://en.wikipedia.org/wiki/Sass_(stylesheet_language))
- Less wikipedia: [https://en.wikipedia.org/wiki/Less_\(stylesheet_language\)](https://en.wikipedia.org/wiki/Less_(stylesheet_language))
- Stylus wikipedia:
[https://en.wikipedia.org/wiki/Stylus_\(stylesheet_language\)](https://en.wikipedia.org/wiki/Stylus_(stylesheet_language))
- compass: <http://compass.kkbox.com/>
- scout: <http://mhs.github.io/scout-app/>
- Koala: <http://koala-app.com/>