

Sass

Instalar sass

```
npm install -g sass
```

La "-g" instala sass de forma global en el equipo.

```
npm sass -v
```

Comprobar versión.

```
sass --watch build/scss/main.scss:build/css/main.css
```

Compilar sass en el documento main.css (los ':' nos dice que están dentro de esas carpetas) y muestra los cambios realizados.

Variables

```
$negro : black;
```

Declarar la variable

```
$font-stack: Helvetica, sans-serif
```

Declarar la variable

```
a{
    color: $negro;
}
p{
    background-color : $negro;
}
```

Usar las variables en propiedades

Guardar variables en una lista de variables (Array)

```
/*
    🖱 Una lista de variables es un Array.
    🖱 Usaremos map-get() para usar las variables
*/

$colores: (
    "negro" : ■ black,
    "blanco" : ■ white,
);
$fuentes : (
    "normal" : Verdana,
    "titulo" : Helvetica
);

header{
    color : map-get( $colores , "negro" ) ;
    font-family : map-get( $fuentes , "normal" ) ;
}
h1{
    background : map-get( $colores , "blanco" ) ;
    font-family : map-get( $fuentes , "titulo" ) ;
}
```

Info obtenida de:

Documentación oficial de [Sass](#)

Canal de YouTube de: [Eduardo Fierro](#)

Mixin

Símil a las funciones en JavaScript, permite agrupar declaraciones de CSS.

Mixin sin Argumentos

Creación con @mixin

Usar @include

Si no hay argumento se pueden omitir los ()

Ej:

```
// Creamos el mixin con "()"
@mixin nombreMixin(){
    background: black;
}

header{
    @include nombreMixin(); // ➡ Usamos el mixin con ()
}

// Creamos el mixin sin "()"
@mixin otroMixin{
    background: black;
}

header{
    @include otroMixin; // ➡ Usamos el mixin sin ()
}
```

Info obtenida de:

Documentación oficial de [Sass](#)

Canal de YouTube de: [Eduardo Fierro](#)

Mixin con Argumentos

Solo 1 argumento

```
/* Mixin con Sólo 1 argumento */
@mixin nombre( $argumento1 ){
  color: $argumento1;
}

p{
  @include nombre( black );
}
```

Más de 1 argumento

```
/* Mixin con Más de 1 argumento */
@mixin nombre( $argumento1 , $argumento2 , $argumento3 ){
  color      : $argumento1;
  font-weight : $argumento2;
  font-size  : $argumento3;
}

p{
  @include nombre( black , bold , 2em );
}
```

Argumento con valor por defecto

<pre>@mixin nombre(\$argumento1 : red){ color: \$argumento1; } p{ @include nombre(); @include nombre(black); }</pre>	<pre>p { color: red; color: black; } /* Mixin con la</pre>
---	---

En este caso, el color por defecto es rojo, pero se puede sobrescribir el rojo por el negro. Importante para futuras mixin más avanzados

Info obtenida de:

Documentación oficial de [Sass](#)

Canal de YouTube de: [Eduardo Fierro](#)

Mixin con la regla @content

```
/*
  Mixin con la Regla @content
  ✎ Nos permite introducir múltiples propiedades CSS dentro de un mixin
  ✎ Muy usado para Responsive Design
*/
@mixin movil(){
  @media screen and (max-width: 480px) {
    @content;
  }
}

p {
  font-size: 4em;

  @include movil(){
    font-size: 2em;
  }
}
```

Resultado en CSS:

```
p {
  font-size: 4em;
}
@media screen and (max-width: 480px) {
  p {
    font-size: 2em;
  }
}
```

Mixin para Flex
Mixin para Grid
Mixin Responsive Design
Crear tu propio Mixin