Preprocesadores CSS

Un preprocesador CSS es una herramienta que te permite generar CSS a partir de la syntax única del preprocesador.

Estas herramientas disponen de utilidades que no están en el lenguaje CSS y, por tanto, no las interpretan los navegadores directamente.

Como los navegadores no interpretan el código utilizado hay que transformarlo a CSS.

VENTAJAS	DESVENTAJAS
Código más organizado	Deben aprender a usarse
Proyectos más fáciles de mantener	Puede haber malas prácticas
Reutilizacion de codigo	Peligro de sobreingenieria
Ahorro de tiempo	
Asegura compatibilidad entre navegadores	

Preprocesadores CSS

Los preprocesadores más utilizados son los siguientes:

- SASS
- LESS
- Stylus
- PostCSS

Preprocesador SASS

- SASS es multiplataforma y puede usarse, por tanto, tanto en PC Windows como en Mac o Linux.
- SASS se basa, al menos en la versión original, en Ruby, por lo que es necesario que este lenguaje de programación esté instalado en el sistema.
- También se puede instalar utilizando npm, aunque es un poco más lento: ***npm i -g sass***
- Los archivos SASS deben tener la extension .sass

SASS

Cuando trabajamos con un preprocesador, cada vez que se modifique el archivo .sass, debemos transformarlo a un archivo .css par que el navegador lo pueda interpretar:

> sass archivo.sass archivo.css

SASS

Sass nos ofrece el modo monitorización, el cual realiza la conversión automáticamente cuando detecta un cambio en los archivos indicados:

> sass --watch archivo.sass archivo.css

Para monitorear carpetas completas:

> sass --watch carpetaSASS:carpetaCSS

SASS Sintaxis

La sintaxis SASS, conocida como "sintaxis indentada" o simplemente "sintaxis sass" permite escribir los estilos CSS de manera más concisa.

En este caso, el anidamiento de selectores se indica con tabulaciones en vez de con llaves y las propiedades se separan con saltos de línea en vez de con puntos y coma.

```
body
 font: 100% Helvetica

 color: blue
```

SASS Variables I

- Piense en las variables como una forma de almacenar información que desea reutilizar en toda su hoja de estilo.
- Puedes almacenar cosas como colores, fuentes o cualquier valor CSS que creas que querrás reutilizar. Sass usa el símbolo $ para hacer que algo sea una variable.
- Cuando se procesa el Sass, toma las variables que definimos con $ y genera CSS normal con nuestros valores de variables colocados en el CSS.
- Esto puede ser extremadamente poderoso al trabajar con colores de marca y mantenerlos consistentes en todo el sitio.

SASS Variables II

```
$font-stack:    Helvetica, sans-serif
$primary-color: #333


body
  font: 100% $font-stack
  color: $primary-color
  background-color: aqua
```

SASS Nesting I

- En HTML se asume que el código se anida en una estructura jerárquica. CSS, en cambio, ignora esta función y obliga al usuario a declarar las propiedades una y otra vez.
- SASS devuelve la anidación a las hojas de estilo al permitir que las subcategorías hereden las propiedades de la categoría superior.
- Esto también garantiza la ligereza del código y la reducción de la carga de trabajo para escribir y esperar.
- Si se emplea el nesting en el código, se usa el símbolo & para reemplazar a una parte del selector por el superior.

SASS Nesting II

```
nav
 ul
   margin: 0
   padding: 0
   list-style: none

 li
   display: inline-block

 a
   display: block
   padding: 6px 12px
   text-decoration: none
```

```
nav ul {
 margin: 0;
 padding: 0;
 list-style: none;
}
nav li {
 display: inline-block;
}
nav a {
 display: block;
 padding: 6px 12px;
 text-decoration: none;
}
```

SASS Nesting II

```
a
  color: $blue
  &:visited
    color: $red
  &:hover
    color: $purple
```

```
a {
color: #0000FF;
}

a:visited {
color: #FF0000;
}

a:hover {
color: #551A8B;
}
```

SASS Import

- SASS tiene una directiva muy útil que permite incorporar otros archivos en la hoja de estilos
- La información del archivo importado se utiliza como si formara parte del código fuente.
- Esto mantiene la claridad en las hojas de estilo.
- La función también puede importar varios archivos en un solo paso:

```
@import "variables"
@import "partials/styles"
@import "partials/test-a", "partials/test-b"
```

SASS Import

- Puede crear archivos Sass parciales que contienen pequeños fragmentos de CSS que puede incluir en otros archivos Sass.
- Esta es una excelente manera de modularizar su CSS y ayudar a mantener las cosas más fáciles de mantener.
- Un parcial es un archivo Sass llamado con un guión bajo. Puede nombrarlo como _partial.scss.
- El guión bajo le permite a Sass saber que el archivo es solo un archivo parcial y que no debe generarse en un archivo CSS.

SASS Mixins I

- Otra directiva importante la constituyen las llamadas mixins, reglas fijas que pueden invocarse en la hoja de estilo cuando sea necesario sin tener que volver a insertar el código completo.
- Una mixin puede contener todo lo que está permitido en SASS.
- Un mixin le permite hacer grupos de declaraciones CSS que desea reutilizar en su sitio.

SASS Mixins II

```
@mixin big-blue-text
 font-family: Arial
 font-size: 25px
 font-weight: bold
 color:#0000ff
```

```
.clase
 @include big-blue-text
 background-color: aqua
```

SASS Extend

- El uso de @extend le permite compartir un conjunto de propiedades CSS de un selector a otro.
- El uso de @extend le permite compartir un conjunto de propiedades CSS de un selector a otro.

```
.button-scope
 margin: 5px
 border-radius: 2px
.home-button
 @extend .button-scope
 background-color: salmon
.back-button
 @extend .home-button
```

SASS Functions I

- SASS conoce numerosas funciones que facilitan el trabajo en la hoja de estilo.
- Se trata de workflows predefinidos que evitan tener que ejecutarlos manualmente.
- Las funciones pueden asignarse a diferentes categorías:
    - **Colores:** con estas funciones pueden ajustarse los valores de color, la saturación, la transparencia y muchas otras propiedades. Con mix(), p.ej., pueden mezclarse dos colores y crear uno nuevo.
    - **Listas:** en las listas (series de valores de propiedad CSS) las funciones sirven para leer el número de entradas o fusionar varias listas en una sola.

SASS Functions II

- Las funciones pueden asignarse a diferentes categorías:
  - **Cadenas:** las strings son cadenas fijas de caracteres, como las que se utilizan en los textos. Las funciones de esta categoría pueden colocar una cadena de caracteres entre comillas o convertir un texto completo a mayúsculas.
  - **Selectores:** con las funciones de esta categoría se manipulan selectores completos. Con selector-unify() puedes fusionar dos selectores en uno solo (ahorrando mucha escritura).

SASS Functions III

- Las funciones pueden asignarse a diferentes categorías:
  - **Números:** dentro del tema de los números, valores o unidades, encontrarás funciones que pueden, entre otras cosas, redondear hacia arriba y hacia abajo, buscar el número más grande de un conjunto o entregar un número aleatorio.
  - **Mapas:** los mapas en SASS son estructuras de datos compuestas de claves y propiedades. Con las funciones adecuadas, estos conjuntos de datos pueden manipularse fusionando dos mapas o borrando una clave de un mapa.

SASS Functions IV

- Las funciones pueden asignarse a diferentes categorías:
  - **Introspección:** estas funciones permiten revisar el contenido de la hoja de estilo completa para comprobar, entre otras cosas, si en el código se encuentra una característica determinada, un mixin o una función en concreto.
  - **Miscellaneous:** en el punto miscelánea SASS también incluye la útil función if() –que no debe confundirse con la directiva del mismo nombre. La diferencia se explica más abajo en el punto “Ramificación”.

SASS Functions V

- SASS también nos permite crear nuestras propias funciones.
- Las funciones le permiten definir operaciones complejas en valores SassScript que puede reutilizar en toda su hoja de estilo.
- Facilitan la abstracción de fórmulas y comportamientos comunes de una manera legible.

SASS Functions VI

```
@function mul($base, $exponent)
 $result: $base * $exponent
  @return $result



.sidebar
 float: left
 margin-left: mul(4, 3) * 1px
```