Lexpin

¿Qué es SASS?

SASS (Syntactically Awesome Style Sheets) es un preprocesador de CSS que permite escribir estilos de una manera más eficiente y organizada, para luego compilarlos en un archivo CSS estándar el cual será usado en un sitio web.

Sintaxis:

SASS permite trabajar con diferentes extensiones para mejorar el desarrollo de hojas de estilo, algunas de las más comunes son:

- **.scss**: Es la extensión más utilizada y es la sintaxis más cercana a la sintaxis de CSS. Los archivos .scss contienen código SASS pero también pueden contener código CSS estándar, lo que los hace más fáciles de leer y escribir para los desarrolladores que ya están familiarizados con CSS.

```scss
body {
    background-color: beige;
}
```

- **.sass**: Es una sintaxis más corta y menos estricta que la sintaxis .scss. En lugar de utilizar llaves y puntos y comas, utiliza sangrías y saltos de línea para indicar la jerarquía y las separaciones.

```sass
body
    background-color: beige
```

¿Qué ofrece SASS?

SASS agrega características como variables, anidamiento, mixins, herencia, entre otros, lo que facilita la escritura y mantenimiento de hojas de estilo más grandes y complejas.

- **Variables**: SASS permite definir variables para valores que se utilizan repetidamente en una hoja de estilo, lo que puede hacer que el código sea más fácil de mantener y actualizar.

```sass
$variable: green

div
    background-color: $variable
```

- **Anidamiento**: SASS permite anidar selectores CSS, lo que puede reducir la necesidad de escribir selectores repetitivos y aumentar la legibilidad del código.

```
$variable: green
$variable2: arial
$variable3: white

section
    background-color: $variable
    div
        width: 100%
        border: 1px solid $variable3
        p
            font-family: $variable2
            color: $variable3
```

- **Mixins**: Los mixins son fragmentos de código que se pueden reutilizar en diferentes partes del archivo de estilo. Pueden tener o no parámetros. Los mixins se definen con la sintaxis "@mixin" seguida del nombre del mixin y sus parámetros (si los tiene). Luego, se utiliza la sintaxis "@include" seguida del nombre del mixin para insertar el código en cualquier parte del archivo.

```
@mixin my-mixin()
    color: white
    background-color: gray


.my-class
    @include my-mixin()
```

Mixin sin parámetros

```
@mixin my-mixin($color1,$color2)
    color: $color1
    background-color: $color2


.my-class
    @include my-mixin(white,gray)
```

Mixin con parámetros

- **BEM**: SASS permite utilizar el operador "&" para poder abreviar al hacer referencia al selector padre.

```
<section .class="caja">
    <div class="caja_hijo">
        <p class="caja_hijo_parrafo">Hola Mundo, Aprendiendo SASS!</p>
    </div>
</section>
```

HTML

```
$variable: green
$variable2: arial
$variable3: white

.caja
    background-color: $variable
    &_hijo
        width: 100%
        border: 1px solid $variable3
        &_parrafo
            font-family: $variable2
            color: $variable3
```

SASS

- **Each**: Es un loop(ciclo) se utiliza para iterar(recorrer) sobre una lista de elementos y aplicar un conjunto de estilos a cada uno de ellos.

```html
<div>
    <p class="parrafo-red">Parrafo red</p>
    <p class="parrafo-green">Parrafo green</p>
    <p class="parrafo-blue">Parrafo blue</p>
    <p class="parrafo-yellow">Parrafo yellow</p>
</div>
```

HTML

```sass
$colors: red, green, blue, yellow

@each $color in $colors
    .parrafo-#{$color}
        color: $color
```

SASS

- **Funciones**: SASS tiene una serie de funciones incorporadas que se pueden utilizar para realizar operaciones en los valores de las variables, también permite crear funciones personalizadas.

  Por ejemplo:

```sass
@function degradado($color-1, $color-2, $direction: to left)
    @return linear-gradient($direction, $color-1, $color-2)

.caja
    background: degradado(orange, gray,to left)
```

- **Extend**: permite reutilizar las reglas CSS existentes en una nueva regla CSS, sin tener que copiar y pegar código. Esta técnica se conoce como "herencia" y es similar a la herencia en la programación orientada a objetos.

```sass
.caja
    &_parrafo2
        background: skyblue
        color: white
        cursor: pointer
    &_parrafo2:hover
        color: orange
    &_parrafo2::first-letter
        color: red
    &_parrafo3
        @extend .caja_parrafo2
        border-radius: 10px
    &_parrafo4
        @extend .caja_parrafo3
        padding: 10px
```

- **Condicionales**
  - **If:** Se utiliza para comprobar si una expresión es verdadera o falsa y, a continuación, aplicar un estilo en consecuencia.

```sass
$color: red;

div
    @if $color == red
        background-color: $color
    @else
    background-color: blue
```

- **Bucles:** se pueden utilizar bucles for y while para repetir un conjunto de estilos varias veces.

  - **For:**

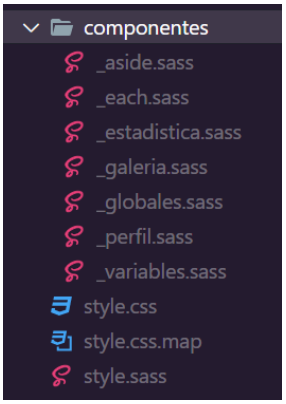
```sass
@for $counter from 1 through 5
    .elemento-#{$counter}
        width: 50px * $counter
        height: 50px * $counter
```

  - **Each:**

```sass
$colors: red, green, blue

@each $color in $colors
    .bg-#{$color}
        background-color: $color
```

- **Import**: se utiliza para incluir archivos SASS dentro de tu archivo principal, lo que permite dividir tu código en archivos más pequeños y modulares.




```sass
@import "componentes/_variables"
@import "componentes/_globales"
@import "componentes/_perfil"
@import "componentes/_estadistica"
@import "componentes/_each"
@import "componentes/_galeria"
@import "componentes/_aside"
```

¿Cuáles son las ventajas y desventajas de usar SASS?

Ventajas:

- **Mejora la eficiencia**: Permite escribir hojas de estilo de manera más rápida y eficiente gracias propias de los lenguajes de programación que CSS no incluye, y que reducen la cantidad de código repetitivo mediante la reutilización de código.
- **Mayor organización**: Facilita la organización de las hojas de estilo mediante el uso de anidamiento y herencia, lo que reduce la necesidad de escribir selectores repetitivos y aumenta la legibilidad del código.
- **Permite trabajar con diferentes extensiones**: SASS permite trabajar con diferentes extensiones como .scss y .sass, lo que da mayor flexibilidad a los desarrolladores y permite trabajar con la sintaxis que mejor se adapte a sus necesidades.
- **Mejora la mantenibilidad**: Al ser más organizado y eficiente, el código de SASS es más fácil de mantener y actualizar en el tiempo, lo que reduce el tiempo de desarrollo y aumenta la calidad del código.

Desventajas:

- **Curva de aprendizaje**: SASS tiene una curva de aprendizaje más pronunciada que CSS estándar, por lo que puede llevar algún tiempo aprender a utilizar **todas** sus características y funciones.
- **Requiere un compilador**: Para convertir el código SASS en CSS estándar, es necesario utilizar un compilador, lo que añade un paso adicional en el proceso de desarrollo.
- **Mayor complejidad**: A medida que se utilizan más características avanzadas de SASS, el código puede volverse más complejo y difícil de mantener para aquellos que no estén familiarizados con la sintaxis.