INTRODUCCIÓN A SASS

Se puede instalar SASS a través de una aplicación o mediante una línea de comandos, y en nuestro caso usaremos la línea de comandos de VStudioCode.

En Windows instalamos Ruby desde su web, y en el terminal picamos:

Instalación mediante VisualCode

1.- Cierra cualquier proyecto abierto. Archivo>Cerrar Carpeta.

WIN
2.- Abrimos el terminal (CTRL+E) y clicamos:

```
gem install sass
```

MAC

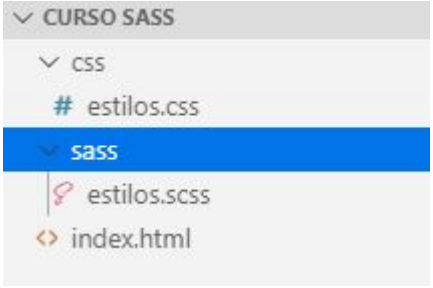
```
npm install -g sass
```

Una vez instalado, comprobamos la versión:

```
sass -v
```

Crear proyecto SASS

Partimos de una carpeta de proyecto, que añadiremos en VStudioCode (Curso SASS), con una estructura inicial de este tipo:



¿Qué aporta SASS?

Sass es un paso intermedio entre los archivos CSS y la programación, por lo que SASS es un tipo de archivo compilado a partir del cual se crea el archivo CSS que el navegador interpreta.

SASS > Compilador > CSS

Enlazar SASS con CSS

Necesitamos enlazar SASS con el archivo CSS. Para eso lo haremos desde terminal, y asegurándonos que estamos en la carpeta del proyecto:

```
sass --watch sass:css
```

Ahora, con el terminal siempre abierto y activo, todo lo que editemos en el archivo de estilos .scss, se añadirá compilado en el correspondiente estilo.css

SCSS

```
h1{
    color: green;
}
```

CSS generado

```
h1 {
  color: green; }

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

Al enlazar el contenido de las carpetas, si ahora creamos otro archivo scss, automáticamente creará el estilo correspondiente:

Probar creando footer.scss

Escribir en SASS (index_2.html)

En este ejemplo quieroque añadan lo siguiente:

footer: color de fondo #444
h4 dentro de footer: color red
link dentro de footer: color blue

Solución 1:

```
footer{
    background: #444;
}
footer h4{
    color:red;
}
footer h4 a{
    color: blue;
}
```

En SASS se trabaja con estructura de árbol, por lo que es más sencillo crear esta estructura:

```
footer{
    background: #444;
    h4{
        color:red;
        a{
            color: blue;
        }
    }
}
```

Y el estilo.css que genera es este:

```
footer {
  background: #444; }
  footer h4 {
    color: red; }
    footer h4 a {
      color: blue; }
```

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

VARIABLES (index_3.html)

A partir de las web colorhunt, vamos a buscar un patrón de 4 colores que nos guste:

color1: #43ab92
color2: #f75f00
color3: #c93838
color4: #512c62

Y editamos los estilos de section y sus diferentes h1,h2,p

```
section{
    background: #43ab92;
    h1{
        background: #f75f00;
    }
    h2{
        background: #c93838;
    }
    p{
        background: #512c62;
    }
}
```

Imaginamos el caso en el que uno de los colores se repite en varios estilos de nuestra web, para eso es vital trabajar con variables.

Las variables se designan en la parte superior y se hace con la siguiente estructura:

```
$colorPrincipal: #43ab92;

section{
    background: $colorPrincipal;
```

Añade el mismo color al párrafo del footer

ESTRUCTURA DE ARCHIVOS (index_4.html; estilos_4.css)

No hace falta que tengamos todos nuestros datos de estilo en un único archivo scss. En este ejemplo veremos cómo separar las variables y luego incorporarlas al proyecto.

Para crear archivos SASS como parte de una sección es importante utilizar:

_nombreArchivo.scss

Nosotros crearemos _variables.scss en la carpeta SASS
En este archivo ponemos todas las variables de mi Site:

```
$colorPrincipal: #43ab92;
$colorDos: #f75f00;
$colorTres: #c93838;
$colorCuatro: #512c62;
```

Y en el archivo de estilos, incluímos el vínculoa variables:

@import 'variables'; (sin _ y sin extensión)

Haz lo mismo para cada una de las secciones header, section y footer

MIXINS (index_5.html)

Es el punto diferencial de SASS ya que nos permite evitar repetición de código innecesario. En el caso que tratamos, vemos que hay una parte de código repetido en diferentes clases:

```
  background-color: #444;
    width: 200px;
    height: 200px;
    margin-top: 5px;
    padding: 10px;
```

Pues se trata de poner ese fragmento de código dentro de un mixin:

@mixin nombre-mixin{
}

Donde queramos referenciarlo pondremos:

@include nombre-mixin;

```
.cuadrado-1{
    @include cuadrado;
}
```

Parámetros de Mixin

Los parámetros es nos permiten configurar el mixin. En el ejemplo anterior de cuadro, podríamos hacer lo siguiente:

```
@mixin cuadrado ($fondo) {
    background-color:$fondo;
```

La llamada al mixin sería:

```
    @include cuadrado(#fff);
```

Si queremos utilizar varios parámetros, se deben separar por coma:

```
@mixin cuadrado ($fondo, $ancho)
```

Realiza un mixin que permita configurar color de fondo, ancho, alto

Valores por defecto

Si en alguna llamada de mixin queremos modificar sólo uno de los parámetros, si no especificamos los otros en la llamada, el compilador nos lanzará un error. Para evitar eso, los parámetros pueden tener valores por defecto para que en caso de no llamarlo, se rellenen con un valor inicial.

```
@mixin cuadrado ($fondo, $ancho:200px)
```

Si no especificamos el ancho, cogerá el valor de 200px que hemos especificado.