SASS NAVIDAD

EDTALLER-139.

Desventajas de usar CSS plano:
- No es un lenguaje de programación.
- Para grandes proyectos:
  > Especificidad - Es la manera mediante la cual los navegadores deciden qué valores de una propiedad CSS son más relevantes para un elemento y, por lo tanto, serán aplicados.
  > Cascada - Es la ejecución en orden secuencial basada en la importancia, especificidad y orden.
  > Herencia - Basada en el uso de clases, por medio del cual una clase se deriva de otra de manera que extiende su funcionalidad.
- Modularización: Dividir código en fragmentos, archivos diferentes, para llevar una organización.

Sass es un superset de CSS, por tanto, es un lenguaje que amplia las capacidad de otro leguaje. Convierte CSS en un lenguaje de programación.
Sass es un preprocesador de CSS. Convierte o compila a CSS vía terminal o prepros.
Sass es 100% compatible con CSS.
Pasos de trabajo:
1. Renombrar: styles.css -> styles.scss.
2. Modularizar: Divide tu hoja de estilos en varios archivos.
3. Importar desde una archivo principal.
4. Compilar: Nombrar los archivos que no queramos compilar con un "_".

Sass for Web Designers.

· Sass hace que lo que construimos sea más eficiente, manejable y mantenible.
· CSS no tiene ninguna abstracción y es muy repetitivo.
· Sass, como Dan te enseñará en este libro, tiene todas las herramientas de abstracción que necesitamos. Los valores repetitivos se convierten en variables.
· Directivas importantes: @mixin @include.
· Sass no dificulta su trabajo, lo hace más fácil.

· Pero la realidad es que Sass (y otros preprocesadores CSS) puede ser un poderoso aliado, una herramienta que cualquier style-crafter puede insertarse fácilmente en su trabajo diario.
· Alguna vez necesitó cambiar, digamos, un color en su hoja de estilo, y encontró que tuviste que buscar y reemplazar el valor varias veces? No desea que CSS le permita hacer esto?
· Sass hace que la creación de hojas de estilo sea más rápida, fácil y flexible.
· Parte del problema es que CSS no fue diseñado originalmente para hacer las cosas que hacemos con él hoy. Claro, el progreso avanza a buen ritmo gracias a la rápida innovación del navegador y la implementación de CSS3.
· Principio DRY (Don't Repeat Yourself):

Todo conocimiento debe tener una representación única, inequívoca y autorizada dentro de un sistema.
La idea es que duplicar el código puede causar fallas y confusión a los desarrolladores.
Escriba patrones que se repiten comúnmente una vez y reutilice en toda la aplicación.
Querían que CSS fuera lo suficientemente potente como para diseñar páginas web y separar el contenido de la presentación, al mismo tiempo que fuera fácil de entender y usar.
· Sass es una extensión de CSS3.
· Si está acostumbrado a la concisión de los lenguajes de programación como Ruby o Python, la sintaxis de Sass le resultará familiar y es posible que se sienta más como en casa.
· Cuando se usa de manera adecuada e inteligente, Sass puede ser una gran ayuda para crear sitios web.

· Expanded:
$ sass --watch --style expanded screen.scss:screen.css
· Compact:
$ sass --watch --style compact screen.scss:screen.css
· Compressed:
$ sass --watch --style compressed screen.scss:screen.css
El estilo comprimido está destinado a la eficiencia, no a los humanos. El estilo comprimido se presta particularmente bien a aplicaciones web con mucho tráfico, en las que el rendimiento de cada archivo Es crucial.
· ¡NO EDITES TU SALIDA!
En este punto, es importante tener en cuenta que cuando utilices Sass, ya no editarás ningún archivo .css. Los archivos .scss son el lugar donde vivirá y respirará. La razón es que cualquier cambio que realice en el archivo .css eventualmente será anulado tan pronto como actualice el .scss y Sass compile la salida.

· Tenga cuidado al anidar, por supuesto. A veces no es necesario ser tan detallado con los selectores, y el anidamiento excesivo puede dificultar la lectura.
· /*! This is a multi-line comment that will
appear in the final .css file. Even in compressed style.
*/
· Las variables son la característica más utilizada de los preprocesadores CSS.
Pasemos a mi segunda característica favorita de Sass: mixins. Donde las variables te permiten definir y reutilizar valores en toda la hoja de estilo, los mixins te permiten definir y reutilizar bloques de estilos. CSS3 LOVES MIXINS.
· No es necesario escribir esas pilas de prefijo de proveedor una y otra vez. Escribe una vez, reutilízalas cuando quieras.

· LIBRERIA MIXIN:
¿No sería eficiente escribir todos estos elementos una vez y reutilizarlos para cualquier proyecto en el que esté usando Sass?

The Compass framework
http://compass-style.org
The Bourbon library
https://www.bourbon.io

· @extend versus @mixin

Cuando un mixin escribirá las mismas reglas en cada declaración desde la que se llama, @extend creará múltiples selectores separados por comas para estilos compartidos. El uso excesivo de un mixin puede resultar en un archivo CSS inflado en el que el contenido del mixin está presente en el CSS compilado cada vez que se llama en Sass. Si se encuentra usando una mezcla una y otra vez a lo largo de la hoja de estilo, tenga en cuenta cómo se compilará y considere si tiene sentido usar @extend o convertir esos estilos repetidos en una clase que se reutiliza en el marcado. No se exceda en @extend usted mismo Usar @extend es una forma poderosa de compartir estilos entre clases, pero tenga cuidado; cuando se usa demasiado, el CSS compilado comienza a ponerse un poco complicado. Extender la misma clase repetidamente a lo largo de la hoja de estilo puede resultar en una declaración de monstruo. Al usar Sass, es fácil olvidar cómo se verá finalmente la hoja de estilo compilada; asegúrese de estar al tanto de cómo Sass produce su trabajo.