## RENDU-DEVOIR(PREPROCESSEUR)

**1) Définition :**

Un préprocesseur CSS est un outil qui permet de combler les lacunes des feuilles de styles afin d’améliorer et d’offrir des fonctionnalités supplémentaires et rendre le code plus flexible (mieux structuré).

**2) rôle et importance d’un préprocesseur :**

En effet, Leur rôle est de simplifier la vie et le travail des intégrateurs en leur fournissant de nouvelles fonctionnalités (quelque chose qui ne modifie pas le fonctionnement d'un langage, mais le rend plus simple à lire ou à écrire). Les préprocesseurs offrent de nouvelles possibilités en termes de conception CSS **: nested rules, variables, mixins, selector inheritance**, et ils accélèrent le développement côté client et facilitent les modifications ultérieures.

**3) les deux types de préprocesseurs**:

Il existe plusieurs préprocesseurs CSS, chacun possédant une syntaxe et un langage qui leur est propre. Bien qu'ils étendent la structure du CSS, ils restent indépendants des normes CSS existantes. Parmi les plus connus se trouvent [**Sass**](https://fr.wikipedia.org/wiki/Sass_(langage))et [**LESS**](https://fr.wikipedia.org/wiki/LESS_(langage))(source : <https://fr.wikipedia.org/wiki/Pr%C3%A9processeur_CSS>).

**4) Etudes comparative entre Sass et LESS**

Apres avoir effectuer quelques recherches sur les deux préprocesseurs, ci-dessous une synthèse listant les avantages et les inconvénients de chaque préprocesseur.

* **Sass**

**Avantage :**

* Une intégration plus rapide et plus souple
* Créé en premier, SASS possède un grand nombre de fonctions et est donc plus mature
* Il est supporté par le framework Foundation (<https://foundation.zurb.com/>)

**Inconvénient :**

* Nécessité d’installer Ruby pour pouvoir installer la gem Sass
* Compilation en css en ligne de commande
* **LESS**

**Avantage :**

* Utilisé par le framework Bootstrap de Twitter
* Compilation à l’aide d’application (SimpLESS, {Less})

**Inconvénients**

* Moins complet que SASS

Pour conclure, nous allons choisir le préprocesseur Sass (**Syntactically Awesome Style Sheet**) et utiliser Ruby et compass pour cette activité.

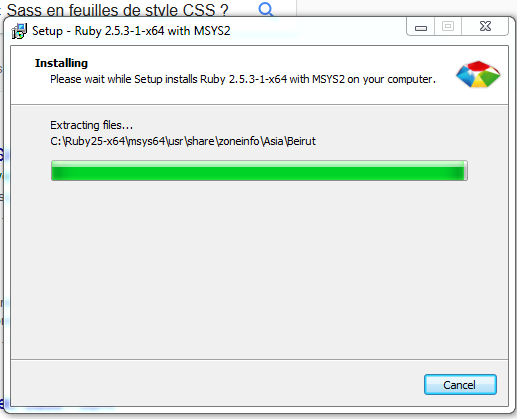
**5)**

**Installation**

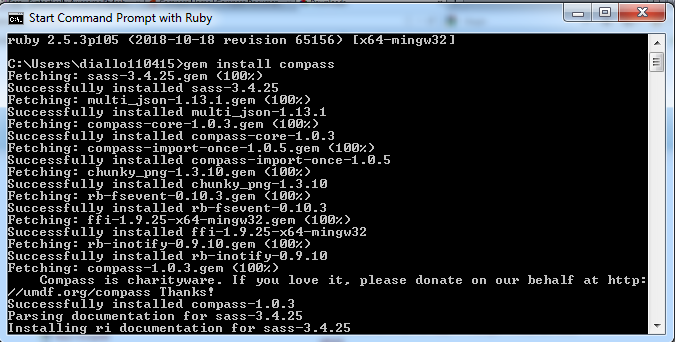
Il faut commencer par installer Ruby, puis le gem sass. Une fois ces deux opérations menées, il ne reste plus qu’à écrire le fichier SASS.

(SOURCE: <http://www.puce-et-media.com/sass-compass-breakpoint-et-susy/> )

Pour installer et configurer Ruby et compass, je me suis servie de la vidéo se trouve sur ce lien : <https://www.dailymotion.com/video/x160gxf>



Une fois Ruby installé, Nous pourrions l’utiliser tel quel, mais il existe une librairie de mixins qui s’appuie sur Sass et surtout en simplifie grandement l’interfaçage. Il s’agit de [compass](http://compass-style.org/). Du coup, avant d’aller plus loin, installons également compass. Cela se traduit encore par une ligne d’instructions à saisir dans la console ou le terminal : **gem install compass**



**6) Transformer les fichiers .scss en css** :

Pour transformer Sass en feuilles styles, nous avons plusieurs possibilités à savoir :

Une première méthode qui consiste à enregistrer le bout de code sous le format .scss, ensuite aller ligne de commande pour la transformation en css ci-dessous un exemple :

Créez un fichier test. Scss et renseignez-y les lignes :

.nav {

h2 {

margin-bottom: 8px;

}

li {

margin-bottom: 1px;

}

Dans un terminal, déplacez-vous à l'endroit où est enregistré le fichier style.scss et lancez la commande suivante : sass test.scss test.css

nav h2 {

margin-bottom: 8px; }

.nav li {

margin-bottom: 1px; }

.nav a {

NB : nous suppons déjà que la commande **compass watch chemin/access/au/dossier** qui demander à Sass de surveiller le dossier du projet afin de compiler le code à chaque fois que nécessaire

Une autre façon consiste à créer directement un fichier contenant **compass watch** au format .bat dans le répertoire d’arborescence où contiennent les fichiers sans passer en ligne de commande.

Une autre manière qui consiste aussi sans utiliser compass. Il s’agit de créer un fichier test.scss en mettant un bout de code dans le dossier ‘’sass’’ et d’aller en ligne de commande taper : sass test.scss test.css

Une fois validée, nous pouvons demander à sass de surveiller le dossier de travail en saisissant cette commande :

**Sass –watchd dossier sources/sass: dossier destination/css**

Conclusion : concrètement j’ai découvert un nouvel univers avec les préprocesseurs, la facilité d’intégration. Bien qu'ils étendent la structure du CSS, ils restent indépendants des normes CSS existantes. Le niveau d'abstraction et la flexibilité obtenus par les préprocesseurs CSS s'avérant plus bénéfiques que toute autre considération

**RESSOURCE:**

<https://sass-guidelin.es/fr/>

<https://www.alsacreations.com/article/lire/1717-les-preprocesseurs-css-c-est-sensass.html>

<http://thesassway.com/>

<https://sass-guidelin.es/fr/>

<https://la-cascade.io/tag/sass/>

<http://compass-style.org/help/documentation/command-line/>

<http://knacss.com/>

<https://rubyinstaller.org/downloads/>

<http://scout-app.io/classic/>

<http://compass-style.org/help/documentation/command-line/>

<http://epicadesign.fr/css-sass-syntaxes-de-base/>