
ADRAR PÔLE NUMERIQUE
> INFRASTRUCTURES SYSTÈMES & RÉSEAUX
> CYBERSÉCURITÉ INFRASTRUCTURES & APPLICATIONS
> DEVOPS / SCRIPTING & AUTOMATISATION
> DEVELOPPEMENT WEB & MOBILE
> TRANSFORMATION NUMERIQUE DES ENTREPRISES
www.adrar-numerique.com

Présentation de Sass

Sass est un langage dynamique de génération de fichier css.

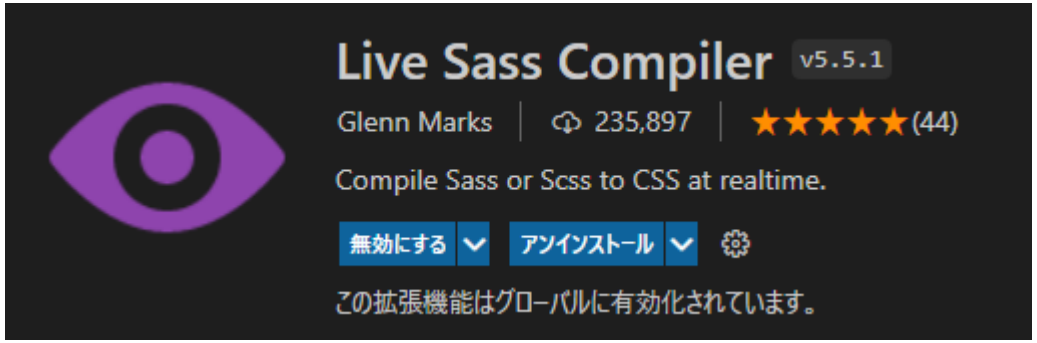
Pourquoi l'utiliser ?

Il permet de faire beaucoup plus de chose que le css comme des variables, Fonctions mathématiques, Fonctions, Boucles, ect.

Présentation de Live Sass Compiler

Comment le mettre en place ?

Il faut télécharger « Live Sass Compiler » , ce dernier va permettre dans notre cas de compiler les fichiers sass en css.



Pour ne pas se tromper de version, il faut vérifier que l'auteur est bien « Glenn Marks » .

Premier projet SASS

On va d'abord créer notre premier projet sass. Il devra contenir :

- Un fichier index.html
- Un fichier style.scss

À partir de ce moment-là, comme à votre habitude, vous devez lier votre fichier HTML et Css.

Vous l'avez peut-être remarqué, nous parlons ici de fichier « css ».

C'est à ce moment que notre compilateur intervient ! Une fois que vous avez édité votre fichier scss, vous devez lancer votre extension Live Sass Compiler qui s'occupera donc de générer un fichier css qui sera interprété par le navigateur.

Premier projet SASS

Avant de commencer, il faut établir le cœur de la syntaxe SASS, à savoir **le Nesting**.

Le principe du Nesting est d'imbriquer les sélecteurs CSS les uns dans les autres.
C'est-à-dire qu'un sélecteur, ainsi que ses règles CSS, se verra être déclaré entre les accolades d'un premier sélecteur.

Une fois compilé, ils formeront une combinaison de sélecteur.
Cela permet de gagner en clarté grâce à une hiérarchie visuelle similaire à celle du HTML.

En css :

```
div {
  background-color: royalblue;
}
div p {
  color: white;
}
```

En sass :

```
div {
    background-color: royalblue;
    p{
        color : white;
    }
}
```

Nous l'avons vu avec le précèdent exemple avec la « `div p` » mais un petit rappel s'impose.
Il est possible de cumuler les sélecteurs en css, c'est ce que l'on appelle les **combinateurs**.

Fort heureusement, Sass permet également de faire ça :

En css :

```css
div {
  background-color: royalblue;
}
div p {
  color: white;
}
div :hover {
  color: salmon;
}
```

En sass :

```sass
div {
    background-color: royalblue;
    p{
        color : white;

    }
    :hover{
        color: salmon;
    }
}
```

Comme nous l'avons dit dans l'introduction, il est possible de créer des variables.
Elles sont définies ainsi :

```
$fond: royalblue;

div {
    background-color: $fond;
    p{
        color : white;

    }
    :hover{
        color: salmon;
    }
}
```

Nous constatons que la variable se crée en deux étapes :

1. En premier, on commence par le symbole « $ » suivi par un nom de variable, ici, c'est le mot « fond ».

2. Nous avons la valeur « royalblue » qui est une terme par défaut au même titre que « blue », « red », etc…

Les variables SASS

Pour plus de clarté, on peut également créer un deuxième fichier SCSS qui permettra d'importer les variables dans votre fichier principal par **@import** :

Style.scss:

```
@import "./variables.scss";

div {
    background-color: $fond;
    p{
        color : white;

    }
    :hover{
        color: salmon;
    }
}
```

variables.scss:

```
$fond: royalblue;
```

Les Mixins

En SCSS, il est possible de stocker du code CSS, un peu comme le font les variables avec des valeurs. Elles se présentent comme ceci :

Style.scss:

```scss
$theme: rgb(217, 149, 243);

@mixin cardArticle {
    background-color: $theme;
    color: #333;
    border: 3px solid #333;
}
```

Ensuite, pour utiliser la mixin, on l'appel via un include :

```scss
@include cardArticle;
```

Il est également possible d'ajouter des paramètres à une mixin, à l'instar des fonctions en JS entre autre. Voici un exemple :

Déclaration de la Mixin:

```
@mixin bordure($couleur, $largeur: 1px) {
    border: $largeur solid $couleur;
    border-radius: 5px;
}
```

Appelle de la Mixin:

```
.button {
    @include bordure(blue);
}

.card {
    @include bordure(red, 2px);
}
```

Ici, notre variable a été déclarée dans les parenthèses de la mixin, ce qui nous permet de l'utiliser en tant que valeur. Attention, un paramètre n'aura pas la même portée qu'une variable déclarée en dehors de votre mixin !

Les Mixins

L'avantage des Mixins est qu'elles évitent de répéter du code lors de son écriture.

Toutefois, si elles sont appliquées à plusieurs sélecteurs, le code CSS compilé montrera des répétitions de code, chose qu'on souhaite éviter selon les bonnes pratiques du DRY (Don't Repeat Yourself).

Doc officielle SASS : https://sass-lang.com/documentation/

Les Extends

Les extensions ou « Extends » de SASS sont similaires aux mixins, mais au lieu de dupliquer un ensemble de propriété CSS, elles dupliquent des **sélecteurs**.

On déclare une extension de la façon suivante : ***%nomExtension {règlesCSS}.***

Ici nous définissons une Extension à utiliser sur tous nos titres courants.

```
%regularTitle {
    font-size: $size-second-title;
    font-family: $font-title;
    color: orange;
   }
```

Les Extends

Pour les utiliser, on appelle l'Extension voulu dans chaque sélecteur visé grâce à la syntaxe suivante :

```
@extend %nomExtension
```

Les sélecteurs visés seront alors regroupés lors de la compilation pour n'obtenir qu'un seul bloc de règle CSS.

```
.card__title{
    @extend %regularTitle;
   }
   .primaryNav__link {
    @extend %regularTitle;
   }
```

Ces appels de l'Extension `%regularTitle` sera compilé de la manière suivante :

```
.primaryNav__link, .card__title {
    font-size: 2rem;
    font-family: AbrilFatface;
    color: orange;
   }
```

Vous avez la possibilité d'organiser vos **@extend** et **mixins** de la même manière que vos variables, en les plaçant dans des fichiers distincts pour améliorer la lisibilité et la réutilisabilité de votre code Sass.

Vous comprenez donc que comme les Mixins, les Extends contribuent fortement à l'écriture d'un code lisible qui suit le principe du Don't Repeat Yourself déjà cité précédemment.