

SASS & Compass

pour les Web Designers

Le préprocesseur *SASS* & le framework *Compass*

Première partie : Le préprocesseur *SASS*

Michel TISSIER
Formateur indépendant

2019

Le préprocesseur SASS & le framework Compass

Première partie

1. Introduction
 1. Qu'est-ce que SASS?
 2. Comment ça marche ?
 1. DRY (Don't Repeat Yourself)
2. Installation
 1. Installation Windows
 1. Installation de Ruby
 2. installation de SASS
 2. Installation Mac
 3. Installation Linux
 4. Installation sans ligne de commande
 1. Scout App
3. Préparation de l'environnement de travail
 1. Les fichiers de SASS
 1. Fichiers cachés
 2. Sourcemap
 2. Préparer son IDE
 1. plugin SASS
 3. Compilation à la volée
 4. Compilation sans ligne de commande
4. SASS
 1. Différence de syntaxe entre SCSS et SASS
 1. Comparaison des syntaxes SCSS / SASS originale
 2. Comparaison des syntaxes actuelles SCSS / SASS
 2. Imbrication
 3. Héritage
 4. import
 5. Variables et fonctions
 1. Variables
 2. Opérateurs
 3. Fonctions

Deuxième partie

5. SASS avancé
 1. Variable avancé
 2. Mixins
 3. Les conditions
 4. Les boucles
6. Compass
 1. Installation
 1. Windows
 2. Mac
 2. configuration
 3. Mixins Compass
 4. Images et sprites
7. Conclusion

1. Introduction

Sass: Syntactically Awesome Style Sheets,
Sass: Des feuilles de style syntaxiquement géniales

1.1 Qu'est ce que SASS ?

SASS est un préprocesseur CSS basé sur le langage Ruby. C'est à dire un langage dynamique de génération de feuille de style en cascade (CSS) (un langage de description compilé en CSS).

SassScript est le langage de script utilisé pour écrire le SASS.

Deux syntaxes existent :

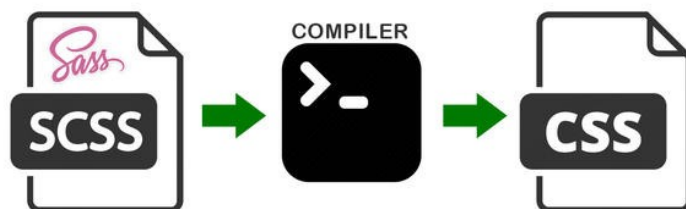
- l'originale nommée syntaxe indentée (**SASS**) proche de HAML,
- et la nouvelle version (Sass 3 ou Sassy CSS) nommé **SCSS** qui a un formalisme proche de CSS.

SASS : [https://fr.wikipedia.org/wiki/Sass_\(langage\)](https://fr.wikipedia.org/wiki/Sass_(langage)) - <https://sass-lang.com/>

HAML : <https://fr.wikipedia.org/wiki/Haml> - <http://haml.info/>

1.2 Comment ça marche ?

SASS est un préprocesseur qui se place avant la feuille de style et permet de composer cette dernière. En écrivant du SASS celui-ci réécrit une page qu'il compile en format CSS. SASS apporte l'avantage d'un langage simple, structuré, élégant qui permet de rédiger du code DRY plus rapide et efficace et plus simple à maintenir (en particulier pour les gros volumes).



1.2.1 DRY (Don't Repeat Yourself)

DRY est un principe de non redondance défini par Andy Hunt et Dave Thomas dans leur livre *The Pragmatic Programmer*, § 2.7 p.27 Tip 11 (ISBN:0-201-61622-X).

Every piece of knowledge must have a single, unambiguous, authoritative representation within a system.

Dans un système, toute connaissance doit avoir une représentation unique, non ambiguë et faisant autorité.

L'idée est que la duplication de code peut être source d'erreur et de confusion. Cela paraît logique que si des motifs se répètent on doit les écrire une fois et les réutiliser à travers l'application. Cela donne un code plus efficace et plus optimisé (charge, maintenance ...).

Or le CSS est tout sauf un langage DRY, il n'est fait que de répétition et de redondance.

SASS est un métalangage qui précède CSS et permet d'écrire dans un style plus clair et organisé. La syntaxe est plus simple et mieux structurée et permet de simplifier la maintenance.

2. Installation

SASS est un programme écrit en Ruby, il faut donc disposer de Ruby sur votre machine. Sur Linux et Mac OS X Ruby est souvent préinstallé, mais pas sur Windows.

2.1 Installation Windows

Il s'agit de Windows 10 dans cet exemple, mais c'est valable aussi pour Windows 8.

Les images de l'installation sont dans le dossier [images/install-sass-windows/](#)

2.1.1 Installation de Ruby

Téléchargez Ruby pour Windows : <https://rubyinstaller.org/> [ruby01.jpg]

Cliquez sur le bouton *download* : <https://rubyinstaller.org/downloads/> [ruby02.jpg]

On va télécharger la version conseillée. Dans notre cas c'est la *Ruby-Dev-Kit 2.5.5-1 (x64)* si votre Windows est en 64 bits, sinon choisissez la version x86 pour un Windows en 32 bits. [ruby03.jpg]

Ensuite lancer le fichier exécutable que vous venez de télécharger. [ruby04.jpg]

- Accepter la licence [ruby05.jpg]

- Cochez bien Add Ruby executables to your Path

ainsi que Associate .rb and .rbw files with this ruby installation [ruby06.jpg]

- Cochez si vous souhaitez avoir l'environnement de développement, sinon **non**. [ruby07.jpg]

Laissez s'installer (long avec l'environnement de dev).

- Fenêtre de confirmation d'installation. [ruby08.jpg]

- Si vous laissez coché vous verrez apparaître un Terminal pour vérifier l'installation [ruby09.jpg].

- Cliquez Finish

> Ouvrez le PowerShell [ruby10.jpg]

Vérifions si Ruby est installé en demandant sa version. [ruby11.jpg]

```
> gem -v
```

Autorisez Ruby en confirmant l'accès au pare-feu Windows [ruby12.jpg]

2.1.2 Installation de SASS

Toujours dans le terminal, on procède à l'installation de SASS

On ordonne à Ruby d'installer SASS [ruby13.jpg]

```
> gem install sass
```

Vérifions si SASS est installé en demandant sa version. [ruby14.jpg]

```
> sass -v
```

2.2 Installation Mac

Les images de l'installation sont dans le dossier [images/install-sass-mac/](#)

> Ouvrez un terminal : Applications/Utilitaires/Terminal

On vérifie si Ruby est installé en demandant sa version. [sass-mac-01.jpg]

```
> ruby -v
```

[sass-mac-02.jpg]

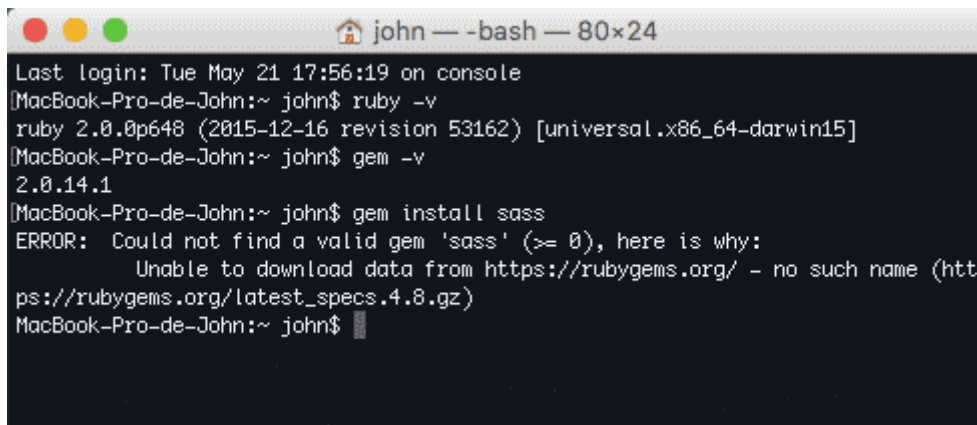
On installe SASS. [sass-mac-03.jpg]

```
> gem install sass
```

Si à ce moment là vous avez une erreur, c'est qu votre Mac est certainement trop vieux.

C'est fréquent avec les anciens OS X d'avant 2017, c'est à dire les versions précédant **Hight Sierra** (10.13) et **Mojave** (10.14) qui elles comportent *Homebrew* comme gestionnaire de paquet. En général le paquet concernant le SSL est obsolète et il faut le remplacer manuellement, une opération inaccessible à la plupart des gens, Merci Apple.

Dans ce cas je vous conseille d'installer Linux ou d'utiliser **Scout App** (chapitre 2.3).



```
john — -bash — 80x24
Last login: Tue May 21 17:56:19 on console
[MacBook-Pro-de-John:~ john$ ruby -v
ruby 2.0.0p648 (2015-12-16 revision 53162) [universal.x86_64-darwin15]
[MacBook-Pro-de-John:~ john$ gem -v
2.0.14.1
[MacBook-Pro-de-John:~ john$ gem install sass
ERROR: Could not find a valid gem 'sass' (>= 0), here is why:
    Unable to download data from https://rubygems.org/ - no such name (https://rubygems.org/latest_specs.4.8.gz)
[MacBook-Pro-de-John:~ john$
```

Si par contre ça marche, on reprend : on en était à l'installation de SASS

On installe SASS. [ill.]

```
> gem install sass
```

On vérifie la version de SASS. [ill.]

```
> sass -v
```

2.3 Installation Linux

Dans un terminal on teste pour savoir si Ruby est installé (deux méthodes équivalentes) :

```
> ruby -v
```

```
> gem -v
```

Si il n'est pas installé :

ce qui est surprenant sur Linux, mais j'ai rencontré le cas avec *LinuxMint*, on l'installe :

```
> sudo apt install ruby
```

Tapez votre mot de passe quand on vous le demande puis O (oui) pour autoriser l'installation.

Puis vérifiez pour avoir la version de Ruby.

```
> ruby -v
```

Si il est installé passez directement à l'installation de SASS :

Vérifier la version de SASS.

```
> sass -v
```

Si il n'y en a pas vous l'installe :

Installation de SASS

```
> sudo apt install ruby-sass ruby-listen
```

- *ruby-sass* est le paquet Sass de ruby,

- *ruby-listen* est le paquet qui permet d'espionner les fichiers pour les modifier.

Les maniaques comme moi referont un test de version

```
> sass -v
```

Voilà vous êtes prêt.

Installer Ruby et Sass sous Linux

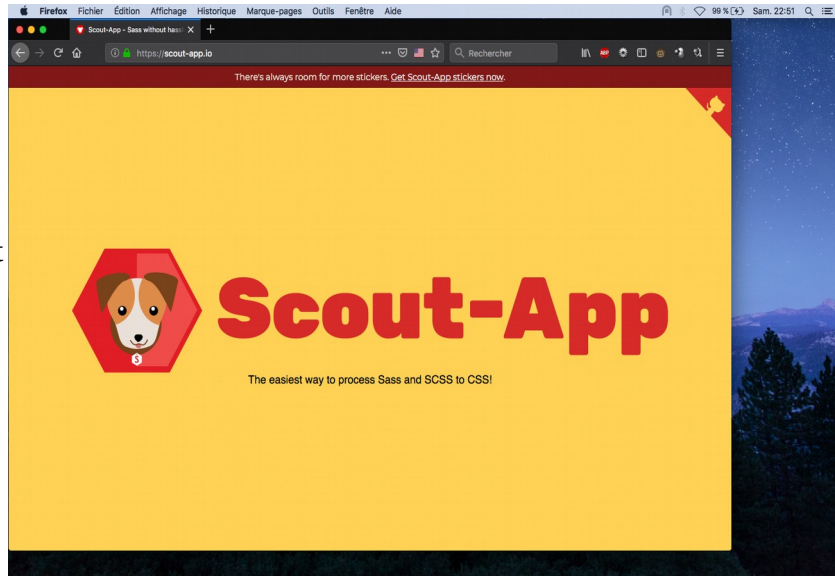
Ruby : <https://www.ruby-lang.org/fr/documentation/installation/#apt>

SASS : <https://sass-lang.com/install>

2.4 Installation sans ligne de commande

Le moyen le plus simple d'utiliser SASS est de passer par un logiciel tiers. Il y en a plusieurs : <https://sass-lang.com/install> mais mon choix s'est porté sur un logiciel gratuit qui fonctionne sur les 3 systèmes principaux (Linux, Mac, Windows), je veux parler de *Scout-App* : <https://scout-app.io/>

Les images de l'installation sont dans le dossier [images/install-sass-mac-scout/](#)



> Sur le site téléchargez la version dont vous avez besoin (ici OSX) en cliquant sur le lien (bouton). [\[mac-scout-02.png\]](#), [\[mac-scout-03.png\]](#), [\[mac-scout-04.png\]](#).

> Puis décompressez-le et glissez-le dans le dossier *Applications/* de votre Mac. [\[mac-scout-05.png\]](#), [\[mac-scout-06.png\]](#)

> Lancez l'application [\[mac-scout-07.png\]](#)

Si vous vous trouvez bloqué avec un avertissement dans le genre :

« **Impossible d'ouvrir X, car cette app provient d'un développeur non identifié** »

Apple ne sait plus quoi inventer. [\[mac-scout-08.png\]](#)

Pour ouvrir ce genre d'application (d'après le support Apple) :

Installer une application provenant d'un développeur non identifié

<https://support.apple.com/fr-fr/HT202491>

Ctrl (rester appuyé) et cliquez sur ouvrir dans le menu contextuel

Ensuite dans la fenêtre cliquez ouvrir

ça fini par s'ouvrir [\[mac-scout-09.png\]](#). 8-)

L'application se lance [\[mac-scout-10.png\]](#).

> Allez dans le menu (du haut) **File/Preferences** [\[mac-scout-11.png\]](#).

> Choisissez votre langue [\[mac-scout-12.png\]](#), [\[mac-scout-13.png\]](#).

Nous voici prêt pour nous lancer [\[mac-scout-14.png\]](#).

Pour créer un projet à l'aide de *Scout-app*, rendez-vous au chapitre **3.4 Compilation sans ligne de commande**. Ainsi que la vidéo sur le site de <https://scout-app.io/> .

3. Préparation de l'environnement de travail

3.1 Les fichiers de SASS

- A la racine de votre dossier où vous avez activé SASS vous trouverez un dossier caché **.nomDossier-cache** (pour nous **.sass-cache**), c'est là où SASS place son cache.

- Dans le dossier contenant vos CSS vous trouverez le nom de vos fichier suivi d'un **.map** (pour **styles.css** ça donne : **styles.css.map**), c'est le **Sourcemap**, ce qui permet à SASS de donner la ligne du fichier SCSS dans l'inspecteur de code de votre navigateur. Ne le jetez pas pendant la période de développement. Il est inutile par contre de le copier sur le serveur de production (quand le site est en ligne).

D'autres fichiers pourront apparaître surtout si vous installez le framework *Compass*.

3.2 Préparer son IDE

Petit florilège de plugin utiles pour les 4 éditeurs les plus courant (Nom, comment installer, plugins) :



Atom

Menu / File / Settings / install

Emmet
sass
File-Icons
linter
linter-sass-lint
Atom Beautify
Highlight Selected
Pigments
open-recent



Brackets

Menu / Fichiers / Gestionnaire d'extensions

Emmet
SASS/SCSS Hints
Brackets File Icons
HTML Skeleton (! + tab)



Sublime Text

Menu / Tools / Command Palette / install Package

Package Control
Emmet
SASS
File Icon
HTML5
Alignment
BracketHighlighter
Color Highlight
JSLint
jquery



Visual Studio Code

Menu / Files / Preferences / Extensions

CSS Peek
Auto Close Tag
vscode-icons
Instant Markdown
ESLint
Javascript (ES6) Code Snippets

Télécharger les éditeurs :

Atom : <https://atom.io/>

Brackets : <http://brackets.io/>

Sublime Text (payant) : <https://www.sublimetext.com/>

Visual Studio Code : <https://code.visualstudio.com/>

Téléchargez les navigateurs

Chromium : <https://chromium.woolyss.com/download/fr/>

Firefox Dev : <https://www.mozilla.org/fr/firefox/developer/>

Opera : <https://www.techspot.com/downloads/6150-opera-developer.html>

3.3 Compilation à la volée

Les images du tutoriel sont dans le dossier [images/sass-bases/](#)

Sur le bureau j'ai placé un dossier nommé **monsie/**.

Dans ce dossier j'ai créé un fichier *index.html* contenant un titre (h1) et surtout un lien vers la feuille de style (qui n'existe pas encore). [\[sass01.jpg\]](#)

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>mon site</title>
  <link rel="stylesheet" href="css/default.css">
</head>
<body>
  <h1>Mon site</h1>
</body>
</html>
```

[\[monsie/index.html\]](#)

J'ai aussi créé un dossier **css/** vide et un dossier **sass/** contenant un fichier **default.scss** vide.

- monsie/
 - sass/
 - default.scss
 - css/
 - index.html

L'idée est de travailler dans le fichier SCSS pour qu'il génère le fichier CSS.

> Dans le bash je me déplace dans mon dossier monsie/ [\[sass02.jpg\]](#)

```
> cd desktop\monsie
```

Évidemment vous adaptez le chemin à votre environnement (et utilisez la touche **tabulation**).

> Faites un petit LS si vous n'êtes pas sur de là où vous êtes. [\[sass03.jpg\]](#)

```
> ls
```

On active SASS dans le dossier de travail en lui précisant : [\[sass04.jpg\]](#)

- le dossier où se trouve les fichiers SASS (sass/)
- puis le dossier où il doit compiler le CSS (styles/).

Ici notre fichier default.scss est dans le dossier sass/ et il doit compiler dans le dossier styles/

```
> sass --watch sass:styles
```

Les noms de dossiers sont arbitraires, ça marche avec ce que vous définissez.

Notez que SASS vous signale qu'il surveille les changements : [\[sass05.jpg\]](#)

```
>>> Sass is watching for changes. Press Ctrl-C to stop.
      write styles/default.css
      write styles/default.css.map
```

Sass surveille les changements.
Pressez Ctrl + C pour arrêter
Il a créé deux fichiers :
le fichier CSS généré depuis sa version SCSS
et un fichier de travail (map).

Ne supprimer pas le fichier **map** car il est très utile, comme on verra plus bas.

> Vérifions dans le dossier styles/

Les deux fichiers s'y trouvent bien. Testons le à présent. [\[sass06.jpg\]](#)

> Dans le fichier default.scss (le fichier SASS)

- tapez le code suivant

- et enregistrez la modification dans votre éditeur. [sass07.jpg]

```
body {background-color: #00FF00;}
```

On note deux choses :

- Dans le bash SASS a automatiquement détecté la modification quand vous avez enregistré et il l'a pris en compte. Il a généré le fichier CSS.

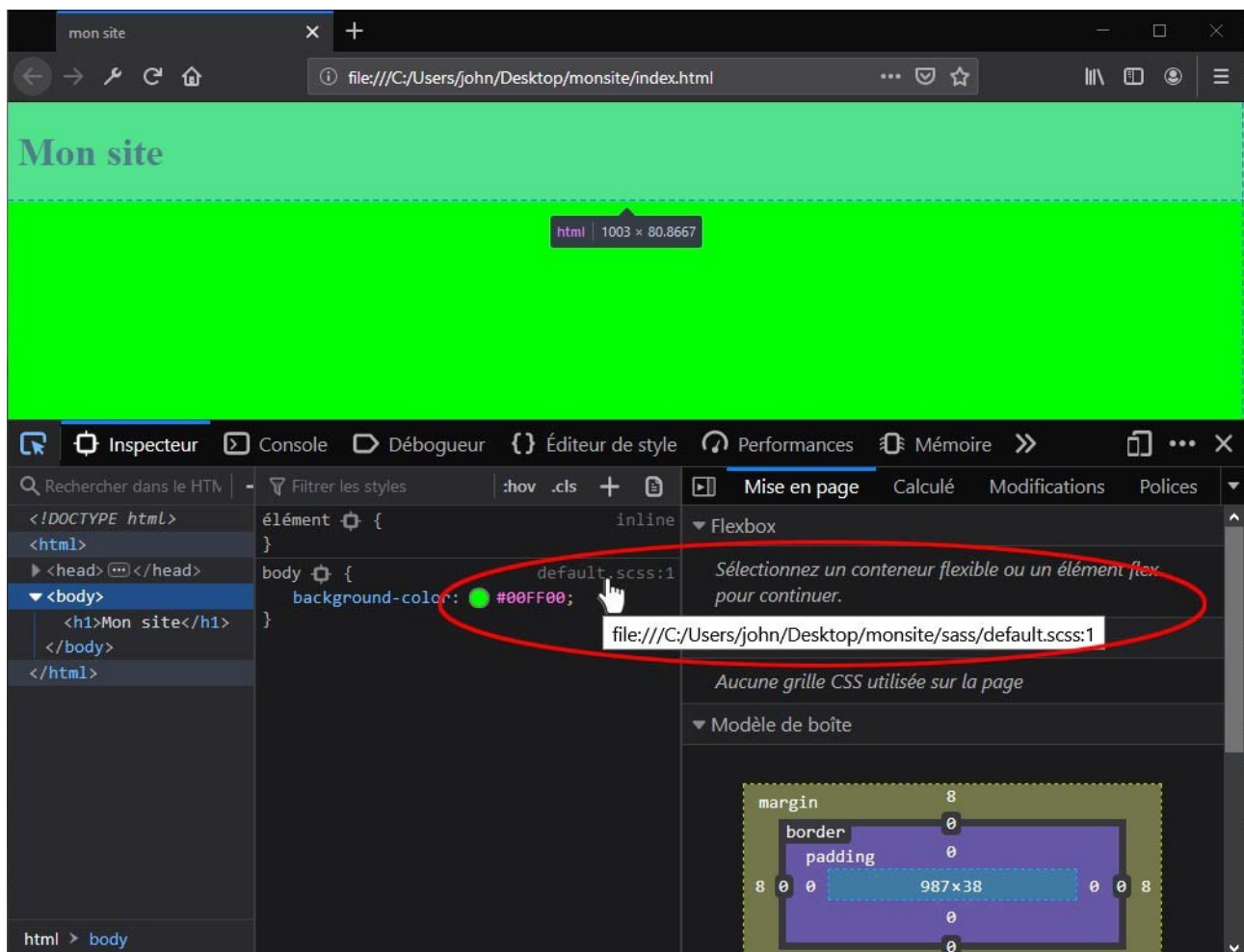
- Rafraîchissez (F5) votre navigateur et vous apercevrez la couleur de fond de la page bien verte.

Pour arrêter il suffit de faire Ctrl + C et une confirmation apparaît [sass08.jpg] Tapez o pour oui [sass09.jpg]

> Jetons un œil en comparant nos fichier **default.scss** et **default.css** [sass10.jpg]

Le CSS a bien été généré par le SCSS.

Revenons sur le fichier **map**, il permet d'afficher dans l'inspecteur du navigateur la ligne de code concerné par style css mais dans le fichier SCSS [sass11.jpg] plutôt pratique !



Si vous souhaitez découvrir d'autres options de commande de SASS tapez **sass --help** [sass12.jpg]

Maintenant que le *Workflow* est en place et qu'il marche, nous n'avons plus qu'à apprendre à nous servir de SASS v3 (SCSS).

3.4 Compilation sans ligne de commande

On peut très bien travailler avec SASS sans pratiquer la ligne de commande à l'aide d'un logiciel tiers tel que Scout-app. Je vous renvoie au chapitre [2.4 installation sans ligne de commande](#).

> Ouvrez votre application Scout-app [\[mac-scout-15.jpg\]](#) (dans le dossier [images/scout-app-practice/](#))
L'interface vous invite à glisser-déposer le dossier où se trouve votre site en développement.

> glisser-déposer votre dossier [\[mac-scout-16.jpg\]](#)

L'interface change laissant apparaître de nouvelles options [\[mac-scout-17.jpg\]](#)

- en haut à droite dans le menu, sur fond noir le nom de votre dossier et à présent du projet en cours (*monsie*)
- On retrouve le nom du projet dans la partie de droite (*monsie*) et en dessous le chemin vers ce dossier.
- Puis le – **Répertoire des feuilles de style** –
 - **Dossier d'entrée** correspond à notre dossier **sass**, c'est là qu'on va travailler.
 - **Dossier de sortie** correspond à notre dossier **css**, c'est là qu'ait généré le fichier css.

> Dans le champ Dossier d'entrée sélectionnez votre dossier où se trouve les fichiers SCSS (sass) [\[mac-scout-18.jpg\]](#)

> Dans le champ Dossier de sortie sélectionnez votre dossier où se trouve les fichiers CSS (css) [\[mac-scout-19.jpg\]](#)

On a bien, dans notre exemple :

Dossier d'entrée : Users/john/Desktop/monsie/sass

Dossier de sortie : Users/john/Desktop/monsie/css

> au dessous dans la section – **Environnement** –

- **Développement** si vous êtes en développement et que vous souhaitez avoir des CSS compilé lisible cochez Développement.
- **Production** si vous êtes en production (pour mettre en ligne) et que vous souhaitez avoir une feuille optimisée cochez Production, [\[mac-scout-20.jpg\]](#)

> Activer la compilation automatique en cliquant sur le bouton bleu dans le menu noir à côté du nom de votre projet [\[mac-scout-21.jpg\]](#) le bouton devient orange quand il est actif.

> Testons à présent si ça marche. Créez un fichier vide nommé default.scss dans le dossier SASS et ouvrez-le dans votre éditeur. [\[mac-scout-22.jpg\]](#)

> Ajoutez ce code :

```
// variable
$bg-color: #00FF00;

body {
    background-color: $bg-color;
}
```

> Quand vous allez enregistrer, un son retenti, et Scout génère une page default.css dans le dossier CSS [\[mac-scout-23.jpg\]](#) . Ouvrez la dans votre éditeur et vous constatez que le CSS a été créé à partir de votre page SCSS.

[\[monsie\]](#) Vérifions dans le navigateur en ouvrant notre page index.html. [\[mac-scout-24.jpg\]](#)

Pour arrêter la compilation automatique il suffit de cliquer sur le bouton orange dans la partie noire du menu tout en haut à gauche. [\[mac-scout-25.jpg\]](#) .

4. SASS

4.1 Différence de syntaxe entre SCSS et SASS

La syntaxe originale de SASS est celle de Ruby, elle a évolué vers une syntaxe plus proche de celle de CSS (Sassy CSS), car la préoccupation des concepteurs était de rendre SASS accessible au plus grand nombre.

4.1.1 Comparaison des syntaxes SCSS / SASS originale

SCSS (Sassy CSS)

```
// SCSS
@mixin cover {
  $color: red;
  @for $i from 1 through 5 {
    &.bg-cover#{ $i } { background-color:
adjust-hue($color, 15deg * $i) }
  }
}
.wrapper { @include cover }
```

SASS originale

```
// SASS
=cover
$color: red
@for $i from 1 through 5
  &.bg-cover#{ $i }
    background-color: adjust-hue($color, 15deg
* $i)
.wrapper
+cover
```

SCSS (Sassy CSS)

La syntaxe est similaire à CSS

- Utilise des accolades { }
- Utilise des points-virgules;
- Le signe de l'affectation est:
- Pour créer un mixin, il utilise la directive @mixin
- Pour utiliser mixin, il est précédé de la directive @include
- Les fichiers ont l'extension .scss.

SASS originale

La syntaxe est similaire à Ruby

- Pas d'accolades
- Pas d'indentation stricte
- Pas de points-virgules
- Le signe de l'affectation est = au lieu de:
- Pour créer un mixin, il utilise le signe =
- Pour utiliser mixin, il est précédé du signe +
- Les fichiers ont l'extension .sass.

La syntaxe est similaire à CSS (à tel point que chaque *CSS3 valide* valide également SCSS, mais la relation dans l'autre sens ne se produit évidemment pas).

4.1.2 Comparaison des syntaxes actuelle SCSS / SASS

SCSS (Sassy CSS)

La syntaxe SCSS utilise l'extension de fichier .scss. À quelques petites exceptions près, il s'agit d'un sur-ensemble de CSS, ce qui signifie que tous les CSS valides sont également des SCSS valides. En raison de sa similitude avec le CSS, il s'agit de la syntaxe la plus facile à utiliser et de la plus répandue.

```
@mixin button-base() {  
  @include typography(button);  
  @include ripple-surface;  
  @include ripple-radius-bounded;  
  
  display: inline-flex;  
  position: relative;  
  height: $button-height;  
  border: none;  
  vertical-align: middle;  
  
  &:hover { cursor: pointer; }  
  
  &:disabled {  
    color: $mdc-button-disabled-ink-color;  
    cursor: default;  
    pointer-events: none;  
  }  
}
```

SASS

La syntaxe indentée était la syntaxe originale de Sass, elle utilise donc l'extension de fichier .sass. Elle utilise l'indentation plutôt que les crochets pour indiquer l'imbrication des sélecteurs et les nouvelles lignes plutôt que les points-virgules pour séparer les propriétés.

```
@mixin button-base()  
  @include typography(button)  
  @include ripple-surface  
  @include ripple-radius-bounded  
  
  display: inline-flex  
  position: relative  
  height: $button-height  
  border: none  
  vertical-align: middle  
  
  &:hover  
    cursor: pointer  
  
  &:disabled  
    color: $mdc-button-disabled-ink-color  
    cursor: default  
    pointer-events: none
```

Sur le site de SASS : <https://sass-lang.com/documentation/syntax>



4.2 imbrication

Les bases de SASS sont disponibles dans la documentation officielle : <https://sass-lang.com/guide>

Les images du tutoriel sont à la suite dans le dossier [images/sass-bases/](#)

4.2.1 Règles d'imbrication (nesting)

On peut imbriquer les règles CSS les unes dans les autres pour éviter la répétition. 

SCSS (Sassy Sass)

```
body {
  background: #FFFF00;
}

p {
  margin: 1rem;
}

.table {
  width: 100%;

  th {
    background: #800080;
    font-weight: bold;
  }

  td {
    border: 1px solid #666;

    td:nth-child(odd) {
      background: #DADADA;
    }
  }
}
```

CSS généré

```
body {
  background: #FFFF00;
}

p {
  margin: 1rem;
}

.table {
  width: 100%;
}

.table th {
  background: #800080;
  font-weight: bold;
}

.table td {
  border: 1px solid #666;
}

.table td:nth-child(odd) {
  background: #DADADA;
}
```

  Pas plus de 2 niveaux d'imbrication pour la lisibilité.

nesting

/ˈnɛstɪŋ/

adjective

adjective: **nesting**; adjective: **nested**

1. (of a bird or other animal) building or occupying a nest.

"do not disturb nesting birds"

2. (of similar objects of graduated sizes) able to be placed or stored one inside the other

"solid wood nesting tables"

nidification

adjective

adjective: **nesting**; adjective: **nested**

1. (d'un oiseau ou d'un autre animal) construisant ou occupant un nid.

"ne pas déranger les oiseaux qui nichent"

2. (d'objets similaires de tailles graduées) pouvant être placés ou rangés l'un dans l'autre

"tables gigognes en bois massif"

Nesting en anglais signifie nidification, on perçoit le concept d'imbrication des branches pour faire le nid, ou bien la relation gigogne, l'un dans l'autre.

4.2.2 Imbrication de propriétés composites

Pour une déclaration de propriétés composites on part de la propriété de base, suivie par deux points superposés, puis on déclare les propriétés. On lieu de *background-color* par exemple on déclarera *color ...* [sass14.jpg]

SCSS

```
header[role="banner"] h1 {  
  padding: 15px 0;  
  font: {  
    size: 54px;  
    family: Georgia, Serif;  
    weight: bold;  
  }  
  line-height: 1;  
  text: {  
    transform: uppercase;  
    decoration: underline;  
    align: center;  
  }  
  background: {  
    color: #EA4C89;  
    size: 16px 16px;  
    image: url(image.png);  
    repeat: no-repeat;  
    position: top left;  
  }  
}
```

CSS

```
header[role="banner"] h1 {  
  padding: 15px 0;  
  font-size: 54px;  
  font-family: Georgia, Serif;  
  font-weight: bold;  
  line-height: 1;  
  text-transform: uppercase;  
  text-decoration: underline;  
  text-align: center;  
  background-color: #EA4C89;  
  background-size: 16px 16px;  
  background-image: url(image.png);  
  background-repeat: no-repeat;  
  background-position: top left;  
}
```

[nesting02.scss] [nesting02.css] On notera les deux points : suivant la propriété d'une déclaration composite.



4.2.3 Le sélecteur parent avec &

On peut faire référence au parent déclaré à l'aide de l'esperluette (&). Dans notre exemple le parent est la balise <a> et on fait référence à elle avec &. [\[sass15.jpg\]](#)

SCSS

```
a {
  text-decoration: none;
  color: #0000FF;
  background-color: none;

  &:hover {
    text-decoration: underline;
    color: #0000FF;
    background-color: #FFFF00;
  }

  .theme-green & {
    background-color: #00FF00;
  }

  &.alert {
    background-color: #FF0000;
  }
}
```

[\[nesting03.scss\]](#) [\[nesting03.css\]](#)

CSS

```
a {
  text-decoration: none;
  color: #0000FF;
  background-color: none;
}

a:hover {
  text-decoration: underline;
  color: #0000FF;
  background-color: #FFFF00;
}

.theme-green a {
  background-color: #00FF00;
}

a.alert {
  background-color: #FF0000;
}
```



&:sass

4.3 Héritage

Utile que si on fait du code générique, pour les pros donc ;-D

Cela permet à une déclaration CSS d'hériter des propriétés d'une autre déclaration tout en y ajoutant ses propres propriétés si nécessaire. On conçoit un bouton générique (.btn) dont l'un des héritiers est un bouton dont la propriété principale est la couleur rouge (.btn-danger).

- ici on déclare un bouton générique .btn [\[sass16.jpg\]](#)

SCSS

```
.btn {
  padding: 5px 12px;
  background: skyblue;
  color: white;

  &:hover {
    background: royalblue;
  }
}
.btn-danger {}
```

CSS

```
.btn, .btn-danger {
  padding: 5px 12px;
  background: skyblue;
  color: white;
}

.btn:hover, .btn-danger:hover {
  background: royalblue;
}
```

- puis un second identique au premier mais avec un fond rouge nommé .btn-danger. [\[sass17.jpg\]](#)

On utilise ici l'héritage (**extend**) pour récupérer toutes les propriétés de .btn sur .btn-danger.

Notez que le simple fait d'ajouter la règle **@extend** ajoute automatiquement .btn-danger au CSS

SCSS

```
.btn {
  padding: 5px 12px;
  background: skyblue;
  color: white;

  &:hover {
    background: royalblue;
  }
}

.btn-danger {
  @extend .btn;
}
```

CSS

```
.btn, .btn-danger {
  padding: 5px 12px;
  background: skyblue;
  color: white;
}

.btn:hover, .btn-danger:hover {
  background: royalblue;
}
```

On complète les propriétés [\[sass18.jpg\]](#)

SCSS

```
.btn {
  padding: 5px 12px;
  background: skyblue;
  color: white;

  &:hover {
    background: royalblue;
  }
}

.btn-danger {
  @extend .btn;
  background: red;

  &:hover {
    background: darkred;
  }
}
```

CSS

```
.btn, .btn-danger {
  padding: 5px 12px;
  background: skyblue;
  color: white;
}

.btn:hover, .btn-danger:hover {
  background: royalblue;
}

.btn-danger {
  background: red;
}

.btn-danger:hover {
  background: darkred;
}
```

[\[extend01.scss\]](#) [\[extend01.css\]](#)

4.4 import


La règle **@import** permet d'importer une page CSS externe. Ainsi on peut découper son CSS dans différents fichiers pour le rendre plus lisible et maintenable. 

SCSS

```
@import "reset.css";
```

CSS

```
@import url(reset.css);
```

Pour l'exemple j'ai pris le reset.css d'Eric Meyer : <https://meyerweb.com/eric/tools/css/reset/> que je vous recommande. Mais tout autre page aurait fait l'affaire.

On peut imaginer de séparer dans des pages différentes la méthode SMACSS en version SASS comme le montre Jonathan Path sur son compte Github :

SASS : <http://smacss.com/fr>

Github: <https://github.com/jonathanpath/SASS-SMACSS>

```
/*
 * SMACSS + SCSS starter v1.0
 * Inspired by SMACSS http://smacss.com
 * Author: @jonathanpath
 *
 * https://github.com/jonathanpath/SASS-SMACSS
 */

/* Base */
@import "base/mixins";
@import "base/normalize";
@import "base/site-settings";
@import "base/site-settings-load";
@import "base/helpers";

/* Layout */
@import "layout/header";
@import "layout/footer";
@import "layout/nav";
@import "layout/wrapper";
@import "layout/cols";

/* Modules */
@import "modules/btn";
@import "modules/slider";
@import "modules/icons";
@import "modules/animations";
@import "modules/tagcloud";
@import "modules/dropdown";

/* Non-Modular */
@import "non-modular/organisation";
```

<http://jonathanpath.com/sass+smacss/scss/style.scss>

Vous pouvez faire votre propre organisation selon votre bon vouloir.

4.5 Variables et fonctions

Dans la doc : <https://sass-lang.com/documentation/variables>

4.5.1 Variables

Une variable est une sorte de contenant pour une valeur. Vous pouvez y placer ce que vous souhaitez réutiliser. Sur une feuille CSS de 3000 lignes cela s'avérera très pratique si vous devez changer une couleur, vous ne le ferez que dans la variable au lieu de parcourir toute la feuille à la recherche des valeurs à changer. [sass20.jpg]

SCSS

```
// variables globales
$color-main : #999;
$color-light : #ccc;
$color-dark : #666;

$font-title : Georgia, Palatino, serif;
$font-main : Verdana, Geneva, sans-serif;
$font-code : "Courier New", Courier,
monospace;

body {
  padding: 0 8%;
  font-family: $font-main;
  font-size : 100%;
  background : $color-main;
}
h1, h2, h3, h4, h5, h6 {
  font-family: $font-title;
  color: $color-dark;
}
```

[variable01.scss] [variable01.css]

CSS

```
body {
  padding: 0 8%;
  font-family: Verdana, Geneva, sans-serif;
  font-size: 100%;
  background: #999;
}

h1, h2, h3, h4, h5, h6 {
  font-family: Georgia, Palatino, serif;
  color: #666;
}
```

On notera le nom générique des variables car le contenu ne doit pas influencer le nom du contenant. Préférez nommer vos variables en fonction de leur usage fonctionnel et non en rapport à des qualités graphiques :

- OUI : main, title, alert, light, dark, medium, large, thin ...
- NON : ~~blue, red, green, serif, non-serif, top, left, right~~ ...



4.5.2 Opérateurs

On peut utiliser des opérateurs arithmétiques dans SASS. [\[sass21.jpg\]](#)

SCSS

```
// variables globales
$padding : 20px;

.container {
  width: 100%;
}

article[role="main"] {
  float: left;
  width: 600px / 960px * 100%;
}

aside[role="complementary"] {
  float: right;
  width: 300px / 960px * 100%;
  padding: $padding + 10px;
}
```

[\[variable02.scss\]](#) [\[variable02.css\]](#)

CSS

```
.container {
  width: 100%;
}

article[role="main"] {
  float: left;
  width: 62.5%;
}

aside[role="complementary"] {
  float: right;
  width: 31.25%;
  padding: 30px;
}
```

Soyez attentif sur le type de vos données lors de vos opérations. Ainsi on ne multipliera pas des em avec des px, ou des lettre (string) avec des chiffres (int). Vérifiez toujours vos calculs.

Opérateur	Description	Exemple
+	addition	h2 {font-size : 5px + 2 ;} //7
-	soustraction	h2 {font-size : 5px - 3 ;} //2
*	multiplication	h2 {width: 3 * (5px + 5px);} //30px
/	division	h2 {width: 3px + (6px /2) *3;} //12px
%	modulo	@if(\$color-length % 2 != 0) {}
==	Égal à	A == B
!=	Différent de	A!= B
>	Supérieur à	A > B
>=	Supérieur ou égale	A >= B
<	Inférieur à	A < B
<=	Inférieur ou égale	h2 { @if(\$padding <= 20px) { } }
and	X et Y	@media (min-width: 768px) and (max-width: 950px) { }
or	X ou Y	A or B
not	X	not B

Modulo : [https://fr.wikipedia.org/wiki/Modulo_\(op%C3%A9ration\)](https://fr.wikipedia.org/wiki/Modulo_(op%C3%A9ration))

Un modulo en informatique c'est l'opération permettant de détecter un multiple (quotient) en testant le chiffre interrogé par une division et en vérifiant le reste après la virgule.

Si je teste 144 / 6 = 24. En modulo : 144 % 6 // il n'y a rien après la virgule, donc TRUE (1).

Si je teste 144 / 7 = 20,47. En modulo : 144 % 7 // il y a des chiffres après la virgule, donc FALSE (0).

On s'en sert pour aller à la ligne tous les X boîtes (box) dans un *portfolio* par exemple.

4.5.3 Fonctions

La liste des fonctions dans la doc : <https://sass-lang.com/documentation/functions>

Si vous cherchez *lighten* : <https://sass-lang.com/documentation/functions/color#lighten>

Une fonction est un sous-programme qui effectue une tâche ou un calcul et qui permet de renvoyer un résultat. Dans la plupart des cas, elle contient des paramètres. Des paramètres sont des informations qui seront récupéré par la fonction et qui serviront à l'exécution de la tâche ou du calcul. Ici nous allons voir les fonctions natives, déjà prêtes à l'emploi.

SCSS

```
// variables globales
$color-01 : rgb(0, 139, 139);
$color-02 : rgb(255, 20, 147);
$color-03 : #FF0000;

aside {
  background: darken($color-01, 30%);

  .btn {
    background: lighten($color-01, 20%);
  }

  .btn-cta {
    background: rgba($color-03, 0.5);
  }
}

.square1 {background: lighten($color-01, 16%);}
.square2 {background: $color-01;}
.square3 {background: darken($color-01, 16%);}

.square4 {background: lighten($color-02, 16%);}
.square5 {background: $color-02;}
.square6 {background: darken($color-02, 16%);}

// consulter la page fonction.html
```

[fonction01.scss] [fonction01.css] [fonction.html]

CSS

```
aside {
  background: black; }
aside .btn {
  background: #00f1f1; }
aside .btn-cta {
  background: rgba(255, 0, 0, 0.5); }

.square1 {
  background: #00dddd; }

.square2 {
  background: darkcyan; }

.square3 {
  background: #003939; }

.square4 {
  background: #ff66b9; }

.square5 {
  background: deeppink; }

.square6 {
  background: #c10069; }
```

Si vous êtes arrivé jusque là c'est que vous êtes courageux et vous avez déjà de quoi faire.

Une fois aguerrie avec les bases de l'algorithmie revenez souffrir avec la deuxième partie ;-D

Dans la deuxième partie :

Les fonctions avancées de SASS (Variables, Mixins, boucles, condition).

Le framework *Compass* et sa librairies de mixins.



Lancez vous dans l'atelier SASS ;-) (dans le dossier [atelier/](#))