CSS- 09- k- Nomenclature BEM

Le choix d'une nomenclature est un aspect important d'un projet. Il permet d'écrire un CSS de qualité et d'anticiper comment les autres membres de son équipe écriront leurs classes CSS. Ceci résulte en un code uniforme, facilement maintenable plutôt qu'une agglomération de différents styles de codes écrits par différents développeurs.

BEM est l'une des nomenclatures CSS les plus répandues. Elle permet d'éviter de nombreux effets secondaires *(side effects)* tout en améliorant la performance des feuilles de styles.

Lorsqu'un projet grossi, il n'est pas rare d'attribuer le même nom de classe à différentes composantes. Par exemple, imaginons que vous créez une composante de héros et que vous souhaitez afficher son titre en italique.

Vous écrivez donc le code suivant:

```
.title {
  font-style: italic;
}
```

Cependant, si vous ou l'un de vos collègues travaille éventuellement sur une composante d'article et qu'à l'intérieur de celle-ci se trouve aussi un titre, il est probable que son réflexe ou le votre soit de le cibler via la classe .title à nouveau, ce qui engendra des effets secondaires indésirables. Le titre de l'article héritera du style italique initialement attribué au titre du héros et tous les nouveaux styles appliqués au titre de l'article s'appliqueront aussi au titre du héros 😷.

Heureusement, BEM permet d'éviter cette confusion!

Origine du nom

Le nom BEM provient de l'abréviation de: Blocs, Éléments et Modificateurs qui sont les trois piliers de cette nomenclature.

Blocs

Les blocs sont des noms de classes représentant des composantes de base pouvant être facilement identifiable dans une page par leur simple nom.

Par exemple: .site-header, .hero, .article, etc. sont tous des composantes que nous devrions être à même de reconnaitre.

```
<div class="hero">
  ...
</div>
```

Éléments

Les éléments sont des sous-composantes avec des noms génériques ayant un lien étroit avec leur bloc.

Par exemple: title, list, item, etc. sont des noms de composantes génériques qui pourraient être présents dans chacun des blocs précédemment mentionnés. Styliser une de ces composantes à partir de son nom générique, par exemple .title, dans chacune de ces composantes entrainerait assurément des effets secondaires indésirables entre chacune d'entre elles.

Heureusement, avec la nomenclature BEM ces effets secondaires seraient évités, puisque la classe d'un élément est constituée du nom de son bloc suivi de deux barres de soulignement _ et du nom de l'élément.

Par exemple:

```
<div class="hero">
  <h2 class="hero__title">Titre</h2>
</div>
```

Modificateurs

Les modificateurs sont des drapeaux, ou en anglais: un *"flags"*, permettant de changer le comportement ou l'apparence d'un bloc ou d'un élément.

Par exemple: active, disabled, big, etc.

Avec BEM, un modificateur est séparé de son bloc ou de son élément à l'aide de deux tirets --.

Par exemple:

```
<div class="hero">
  <h2 class="hero__title hero__title--big">Titre</h2>
</div>
```

Remarquez qu'un modificateur ne remplace pas une classe de base, mais est ajouté en surplus.

Imbrication

Il faut faire attention avec l'imbrication de BEM. Personne ne veut travailler dans un code où des classes CSS ressemblent à:

```
.homepage__hero__wrapper__title { ... }
```

Il est donc important de bien savoir diviser ses blocs. Dans l'exemple précédent, il serait logique d'avoir un bloc de départ .homepage ainsi qu'un bloc .hero.

Le bloc .hero pourrait avoir différents niveaux d'éléments, mais il n'est pas nécessaire de nommer chacun d'entre eux.

Par exemple, il n'est pas nécessaire dans son nom de classe de spécifier que le titre se trouve dans le wrapper. Ainsi une division de la sorte permettrait d'obtenir un code plus lisible:

👌

```
.hero__wrapper { ... }
.hero__title { ... }
```

Get BEM

Site de référence consacré uniquement à la nomenclature BEM.

>

Nesting

Il peut être fastidieux de toujours répéter le même bloc au début de chaque nom de classe. D'où pourquoi la nomenclature BEM se marie si bien avec SASS. En tirant profit de l'imbrication (*nesting)* de SASS, il est possible de définir un bloc de ensuite ses éléments à l'intérieur de lui, sans jamais se répéter!

Par exemple:

```
.hero {
  &__wrapper { width: 100%; }
  &__title { font-style: italic; }
}
```

Ce qui générera le code CSS suivant:

```
.hero__wrapper { width: 100%; }
.hero__title { font-style: italic; }
```

Modificateurs

Les modificateurs sont faciles à écrire à l'aide de @extend.

Par exemple

```
.hero__title {
  font-style: italic;

  &--big {
    @extend .hero__title;
    font-size: 40px;
  }
}
```

Ce qui générera le code CSS suivant:

```
.hero__title, .hero__title--big {
  font-style: italic;
}
.hero__title--big {
  font-size: 40px;
}
```

Découpage

Il est fortement suggéré que chaque élément assez important pour être un bloc est son propre fichier SASS distinct.

Par exemple, le fichier *hero.scss* ne contiendrait qu'un seul bloc, soit .hero. Le fichier *site-header.scss* ne contiendrait que le bloc .site-header et ainsi de suite.

Alternatives

- OOCSS
- SMACSS
- SUITCSS
- etc.