



Cours HTML

3WAcademy

Cours d'intégration Web

[7. Conventions et bonnes pratiques](#)

[7.1. Déclarations CSS](#)

[7.2. Ordre des déclarations](#)

[7.3. Les conventions de nommage](#)

[7.3.1. BEM](#)

[7.3.2. Title CSS](#)

[7.3.3. OOCSS - Object Oriented CSS](#)

[7.3.4. SMACSS - Scalable and Modular Architecture for CSS](#)

7. Conventions et bonnes pratiques

Un ensemble de règles et de principes ont été publiés par Nicolas Gallagher, auteur de `normalize.css`: <https://github.com/necolas/idiomatic-css>

Certains principes sont commun à tous les langages, à savoir:

- Écrire un code lisible et compréhensible par vous et par d'autres.
- Choisir une nomenclature et s'y tenir. Garder son code cohérent.
- Utiliser si possible une convention largement utilisées par d'autres.
- Choisir un mode d'indentation (tabulation ou espaces) et s'y tenir.
- Commenter le code avec si possible, des sections et des blocs de commentaires pour décrire votre code sur plusieurs lignes. Utilisez par exemple le style de Doxygen.

Exemple de commentaires

```
/* =====  
   Section comment block  
   ===== */  
  
/* Sub-section comment block  
   ===== */  
  
/**  
 * Short description using Doxygen-style comment format  
 *  
 * The long description is ideal for more detailed explanations and  
 * documentation.  
 * It can include example HTML, URLs, or any other information  
 * that is deemed necessary or useful.  
 *  
 * @tag This is a tag named 'tag'  
 *  
 * TODO: This is a todo statement that describes a task to be completed  
 *       at a later date. It wraps after 80 characters and following lines are  
 *       indented by 2 spaces.  
 */  
  
/* Basic comment */
```

7.1. Déclarations CSS

Quelques règles simples:

- Déclarez un sélecteur par ligne avec une virgule en fin de ligne si plusieurs sélecteurs.
- Préférez une déclaration par ligne à l'intérieur d'un bloc de déclarations sans oublier le point-virgule à la fin de chaque ligne.
- Indentez toutes les déclarations d'un même bloc avec un niveau d'indentation.
- laissez un espace après chaque virgule dans les listes de valeurs ou dans les paramètres de fonctions.
- Préférez les doubles guillemets
- Fermez chaque bloc de déclaration au même niveau que la première lettre du sélecteur de celle-ci.

```
.selector-1,
.selector-2,
.selector-3[type="text"] {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
    display: block;
    font-family: helvetica, arial, sans-serif;
    color: #333;
    background: #fff;
    background: linear-gradient(#fff, rgba(0, 0, 0, 0.8));
}

.selector-a,
.selector-b {
    padding: 10px;
}
```

7.2. Ordre des déclarations

Il existe plusieurs façons d'organiser vos déclarations au sein d'un bloc. Par exemple en suivant l'ordre alphabétique ou suivant votre propre convention. L'une des meilleurs pratiques est de décomposer les déclarations en 3 ou 4 parties:

Les déclarations concernant le positionnement

Les déclarations concernant le modèle de boîte et son display

Les déclarations concernant le texte

Les déclarations concernant l'habillage et les couleurs et le reste

```
.selector {
    /* Positioning */
    position: absolute;
    z-index: 10;
    top: 0;
```

```
right: 0;

/* Display & Box Model */
display: inline-block;
overflow: hidden;
box-sizing: border-box;
width: 100px;
height: 100px;
padding: 10px;
border: 10px solid #333;
margin: 10px;

/* Text */
font-family: sans-serif;
font-size: 16px;
text-align: right;

/* Other */
background: #000;
color: #fff;
}
```

7.3. Les conventions de nommage

7.3.1. BEM

<https://en.bem.info/method/naming-convention/>

BEM offre une ensemble de règles de nommage des classes CSS afin de mieux identifier les classes dans un document et de les regrouper en ensembles cohérents. ces règles définissent 3 types de classes:

- les blocs
- les éléments de ces blocs
- les modificateurs associés

Pour ces trois types, il est possible d'utiliser les minuscules, majuscules, chiffres ainsi que le tiret simple

Les éléments de blocs sont écrits à partir du bloc parent suivi par un double trait de soulignement (`_`) et d'un nom.

Les modificateurs indiquent un changement d'état du bloc ou de l'élément concerné. Ils sont écrits à partir du nom du bloc ou de l'élément concerné suivi par un double tiret (`--`) et d'un nom.

Exemple

```
<div class="block block--modifier">
  <p class="block__element block__element--modifier">...</p>
</div>
```

7.3.2. Title CSS

<http://www.sitepoint.com/title-css-simple-approach-css-class-naming/>

Il s'agit d'une alternative à BEM dans laquelle les blocs sont simplement écrits avec une majuscule. Les éléments des blocs et les modificateurs sont écrits en camelCase.

Exemple de code HTML:

```
<div class="Title isModified">
  <p class="descendant">...</p>
</div>
```

Et les déclarations CSS associées:

```
.Title {}
  .Title.isModified {}
  .Title .descendant {}
```

7.3.3. OOCSS - Object Oriented CSS

<https://www.smashingmagazine.com/2011/12/an-introduction-to-object-oriented-css-oocss/>

Il s'agit surtout de principes afin d'éviter toute redondance dans les déclarations CSS, comme en programmation Orientée Objets.

Les classes CSS doivent être pensées comme réutilisables le plus possible.

7.3.4. SMACSS - Scalable and Modular Architecture for CSS

<https://smacss.com/book/>

Il s'agit d'un ensemble de principes pour permettre de d'organiser les différentes déclarations en fonction de leur rôle, à savoir:

- Les déclarations de bases communes à l'ensemble du document, comme par exemple les déclaration de reset.css ou de normalyse.css.
- Les déclarations relatives au modèle (layout) principal de la page
- Les déclarations relatives aux modules (les blocs)
- Les déclarations relatives aux états (les modificateurs)
- Les déclarations relatives au thème (couleurs, fonts....)
- Les déclarations relatives aux éléments variables comme par exemple les media-queries