Créer son site Web avec

WordPress

c’est super-simple, c’est super-rapide, c’est gratuit ~~et ça sent bon le poney~~

Bruxelles Formation | CEPEGRA | Olivier Céréssia | 2017  
  
webdesigner.cepegra-labs.be

# Quoi ?! Je ne code pas encore comme un ninja ?

Vous avez découvert pendant quelques semaines le code et nous vous avons volontairement laissés prendre vos marques avec ce nouvel univers.

Maintenant, il est temps qu’on vous dise la vérité : on ne code pas vraiment comme ça, dans la vraie-vie !

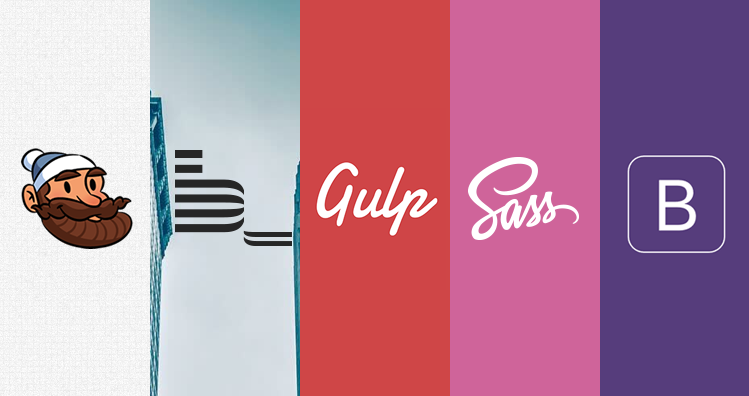
Mais rassurez-vous, vous avez toutes les bases et elles étaient nécessaires pour pouvoir vous rendre plus propre.

Dans ce long chapitre, nous allons découvrir des langages, des techniques, des programmes qui vont faire de vous des « pros ». Vous allez coder comme des « grands » quand vous aurez suivi l’entièreté de ce module et ça, c’est vraiment bien.

Au programme, plusieurs copains :

* Modular CSS
* BEM
* SASS
* Gulp
* Bootstrap

Nous allons tout d’abord apprendre le dérivé d’un langage que vous connaissez bien à travers SASS et nous allons voir tout ce que ce pré-processeur peut nous apporter. Ensuite nous changerons notre approche des feuilles de style et nous les diviserons en plein de petits fichiers pour plus facilement pouvoir les mettre à jour à l’avenir. Avec BEM nous allons essayer d’harmoniser et d’homogénéiser la façon d’approcher nos noms de classes et la façon dont on identifie certains éléments. Ensuite nous mettrons en place un outil qui va nous permettre d’accroître grandement notre productivité (Gulp) et nous finirons par voir Bootstrap pour vous montrer qu’on n’est pas toujours obligés de connaître CSS pour pouvoir faire des mises en page complexes.



# SASS : CSS with superpowers

*Boxent dans la même catégorie : LESS, Stylus*

Écrire en CSS, c’est chwette (répétez cette phrase 10 fois de suite devant votre miroir et j’apparaîtrai dans votre dos) mais il faut bien avouer que pour certaines choses c’est vachement redondant et on perd du temps.

Des exemples concrets ?

* Quand on indique 47 fois dans sa feuille de style le code hexa #ecd6ff (et qu’on a dû aller vérifier environ 46 fois « Tiens, c’est quoi déjà le code hexa de mon site ? ») (applicable pour les typos aussi par exemple)
* Quand on fait 90% du temps la même transition CSS et qu’on la ré-écrit à chaque fois en entier en oubliant à chaque fois la combinaison fatale du cubic-bezier qu’il faut écrire pour avoir l’effet-élastique-qui-déchire-sa-maman.

# Modular and easily maintainable CSS

Jusqu’ici, nous écrivions notre CSS en « block », tout ensemble. On mélangeait tout sans gêne et il faut bien reconnaître que, nos feuilles de styles se rallongeant, cela devient parfois compliqué de retrouver où on a bien pu mettre ce bout de code de la navigation de notre footer…

La bonne nouvelle c’est que des gens se sont déjà posé des questions avant nous. Ils se sont demandé comment faire pour que tout soit plus facile à transmettre, à lire, à mettre à jour, à prendre en main … Et plusieurs conventions de structure et de nommage ont vu le jour.

Nous allons donc tout séparer.