WebForce3

Du 9 mai au 30 août 2018

# Informations diverses

## Formateurs

Formateur HTML, CSS/SASS, JavaScript, Angular/Ionic, Symfony :

Formateur : Mike

Tél : 06.68.00.51.26

Mail : [msylvestreets@yahoo.fr](mailto:msylvestreets@yahoo.fr)

## Communication

Il y a le site de WebForce3 <https://cours.wf3.fr/> mais Discord <https://discordapp.com/> est à privilégier.

# Le HTML

Un site web c’est du contenu affiché sur un navigateur.

Un site peut être local, et il n’est pas forcément nécessaire d’aller sur internet pour le voir. On peut afficher un fichier html en cliquant dessus du bouton droit et ouvrir avec un navigateur.

Le HTML est un langage de balisage. On y code la disposition des éléments sur une page. Ce que l’on voit dans le navigateur est l’interprétation du code ; pour voir le code lui-même, notamment html, taper Ctrl+U ou avec click droit > « Afficher code source. »

Une balise html est généralement ouvrante et fermante. En plus du nom de la balise, celle-ci peut avoir des attributs.

Exemples de balises ouvrantes et fermantes :

< !--p pour paragraphe -->

<p>Lorem ipsum</p>

< !--de h1 à h6 -->

<h1>Titre 1</h1>

On aura compris que ce qui est entre < !-- et --> est un commentaire html.

Exemples de balises uniquement ouvrantes :

< !--image, avec un attribut -->

<img src="images/portrait.jpg">

< !--retour à la ligne -->

<br>

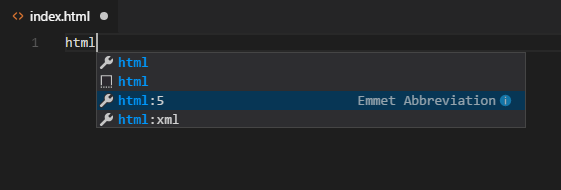
< --trait horizontal -->

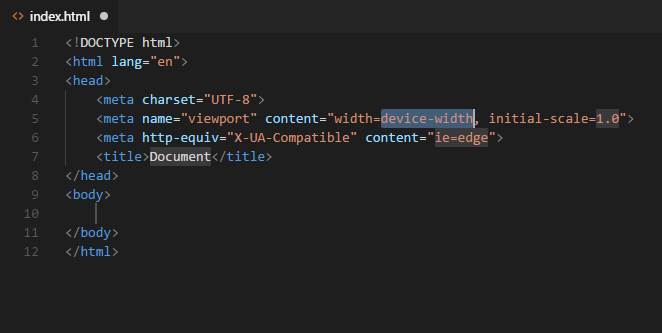
<hr>

A noter qu’avec le html5 on a laissé tomber les slashs chers aux puristes. On n’écrit plus <br/> ni <hr/>.

VSCode permet de générer du code avec des touches de raccourci :

On créé un fichier index.html dans l’explorateur de VCCode. Pour générer les éléments de base d’un fichier html, on commence à taper dans la première ligne « html », on sélectionne un élément de liste (html:5 au hasard) puis <tab> ou <enter> :





Grosso modo :

<!DOCTYPE html>

<html>

<head>

<!-- code -->

</head>

<body>

<!-- code -->

</body>

</html>

Las balises Doctype et html sont W3C obligatoires.

Charset pour l’encodage, utf-8 pour les langues européennes. Sans cela, on aurait des drôles de caractères accentués.

Ligne suivante, viewport, c’est pour le responsive.

La ligne suivante n’est pas indispensable.

La balise title : Titre de la page, qui s’affiche sur l’onglet du navigateur. Pas indispensable pour le fonctionnement de la page mais obligatoire en termes de référencement. Sans cette balise, pas de référencement possible.

Tout code qu’on écrit se situe entre <head> et </head> et entre <body> et </body>.

Balises en première approche :

p Sert à afficher un paragraphe. Bien noter qu’il prend toute la largeur de la ligne.

span Sert à isoler une portion de paragraphe, par exemple un mot.

b, i, u Respectivement pour mettre en gras, en italique, souligné. Réminiscences de la préhistoire d’internet, ils existent encore en raison de la rétro-comptabilité. A éviter évidemment. Le HTML est la disposition, la *beauté* c’est pour le css. Bien noter qu’ils n’ont pas de signification intrinsèque, ils sont stylés en css.

h1~h2 Titres. Noter que de h1 ne doit, suivant les règles de référencement annoncées par Google, y en avoir qu’un seul.

main Il ne doit y en avoir qu’un seul (SEO). Sans cette balise, la page fonctionne, mais sa présence est quand même obligatoire (wf3).

header En-tête de la page. Aujourd’hui on y a mis des menus. Il peut y en avoir plusieurs sur une page. Noter qu’il existe un pendant inférieur, « footer. »

<ol> Liste ordonnée, c’est-à-dire que les éléments de la liste sont numérotés.

<ul> Liste non ordonnée. A la place des nombres, il y a des puces ou autres choses selon ce qui a été défini dans le CSS.

<li> List Item, c-à-d « élément de liste. » C’est une balise qui doit se trouver dans une liste, ordonnée ou désordonnée.

<a> Lien.

A propos des listes, un exemple :

<ul>

<li>Toto</li>

<li>Titi</li>

</ul>

Pour réaliser des menus, on utilise volontiers les balises de listes, et en particulier la liste non ordonnée <ul>.

Toutes les règles html et css sont édictées par la W3C.

Attention : quand on affiche une page web dans notre ordinateur, le code html/css/js se trouve dans l’ordinateur. Ne rien y mettre de secret, tels que des mots de passe.

Les balises peuvent avoir des attributs. Il y a par exemple les attributs id, name ou encore class.

<header id="header" name="header" class="header">

< !-- code -->

</header>

« id » est un identifiant unique de la balise. Ça s’utilise plus volontiers en JS qu’en CSS. En CSS on préférera les balises « class ». L’attribut « name » n’est pas un identifiant unique, il peut y en avoir plusieurs. Utilisé surtout pour les formulaires ; peut aussi être utilisé pour les css.

Attributs de <a> :

<a href="www.google.fr" target="\_blank">Google</a>

L’attribut href est la page vers laquelle pointe le lien. L’attribut target="blank" indique que la page pointée par le lien doit s’afficher dans un nouvel onglet. href possède des sortes de sous-attributs :

href="tel:+33600000000"

href="mail:alex@google.fr"

Sur un site internet, indiquer « tel » ne changera rien mais sur un mobile, ça permettra d’afficher le clavier numérique.

## <i>, <b>, <u>

Ces balises sont désormais, et depuis longtemps, bannies. Sus à la décoration dans les balises html. Reste qu’on peut avoir envie de souligner ici, mettre en gras plus loin, et en italique plus loin encore. Pour cela on utilisera les balises <strong> et <em> pour emphasis. Bien que par défaut le premier soit stylé avec du gras et le second avec de l’italique, l’idée est que strong et em ont une signification *sémantique* et qu’il s’agit de styler cette signification.

## Les balises div et span

Les balises ont un style css par défaut, et ce qui caractérise div et span l’une par rapport à l’autre est que div a un style display:block, et span un style display:inline. Avant le html5, une page se structurait avec une multitude de balises div, qu’on appelait des boîtes.

Un peu vieillottes, ces balises ne représentent pas moins deux façons fondamentales de structurer les données. La première structure typiquement des paragraphes, des ensembles de paragraphes. Le second des mots à l’intérieur d’un paragraphe.

D’une part le display:bloc de div, correspondant au paragraphe de Word, qui s’il prend la hauteur du contenu, occupe toute la largeur disponible, c’est-à-dire, celle que lui laisse son conteneur. Le display:inline, équivalent de la police sur Word, n’a de largeur que celle de son contenu. C’est la raison pour laquelle un style text-align:center ne peut pas marcher sur un élément dont le display est inline. Il faut bricoler avec margin-right et margin-left en leur donnant des valeurs « auto. »

## La sémantique html5

Auparavant, donc, on utilisait beaucoup de div. html5 se veut plus subtil d’un point de vue sémantique afin de permettre aux robots de Google de mieux analyser la page. Pour cela, des balises ont été créées :

* section
* article
* nav
* header
* footer
* main
* aside

## Position, float

Le navigateur lit le code, de droite à gauche, du haut vers le bas. Il place un premier élément, puis un deuxième dont la position va forcément dépendre du premier, puis un troisième qui va encore dépendre des deux précédents. Et ainsi de suite. On appelle cela le flux.

Float permet de sortir un élément du flux et donc de tricher avec le placement normal d’un élément. C’est de la bidouille intégrale, inventée pour empêcher les intégrateurs de se servir des tables pourtant si pratiques, et depuis remplacée dans l’usage par des frameworks tels que Bootstrap ou des techniques telles que flexbox.

# CSS

## CSS avec SASS

Une alternative à SASS est LESS. L’un s’est imposé et l’autre est tombé en désuétude. Le premier c’est SASS, le second LESS.

Pour utiliser SASS dans VSCode, commencer par créer à la racine du projet un dossier « .vscode » dans lequel il y a un fichier settings.json avec le code suivant :

{

"liveSassCompile.settings.formats":[

{

"format": "expanded",

"extensionName": ".css",

"savePath": "/css"

},

{

"extensionName": ".min.css",

"format": "compressed",

"savePath": "/dist/css"

}

],

"liveSassCompile.settings.excludeList": [

"/node\_modules/",

".vscode/\*\*"

],

"liveSassCompile.settings.generateMap" : true,

"liveSassCompile.settings.autoprefix": [

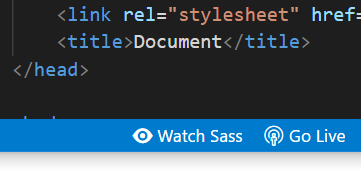
"> 1%",

"last 2 versions"

]

}

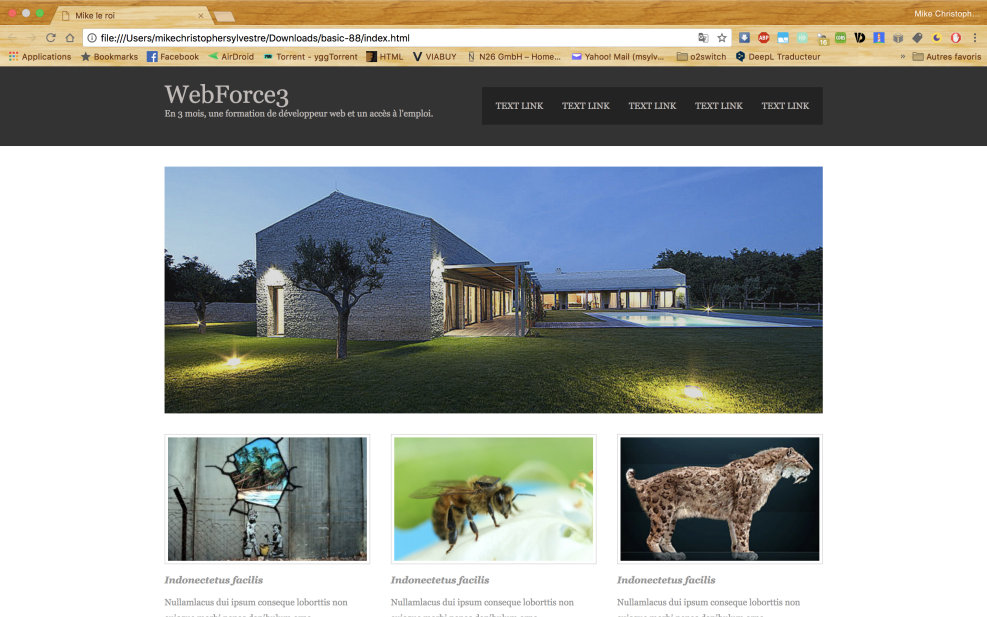
Ensuite, il faut que sass soit « à l’écoute », en cliquant sur Watch SASS :



A partir de là, on ne touche plus au code css, on n’écrit que du sass. A chaque enregistrement, sass recompile le css, ce qui veut dire que tout ce qui aurait été modifié directement dans le fichier css serait écrasé. Si une erreur empêche la compilation, le css est effacé. En production, évidemment, ce sont les fichiers css qu’il faut déployer.

## L’intégration de htmlcss 1 énoncé exercice html-css 1)

On a la maquette suivante :



Avec, pour les images les url :

<https://www.eleganceandgraceweddings.com/wp-content/uploads/2018/04/barre-de-seuil-porte-d-entree-leroy-merlin-image-et-photos-elegant-entrepreneur-en-construction-de-maison-neuve-montreal-laval-rive-sud-10-of-barre-de-seuil-porte-d-entree-leroy-merlin-image.jpg>

et

<https://enmodesneakers.fr/wp-content/uploads/2017/12/6-image-unwelcome-intervention-7797-290x180.jpg>

La première est la maison, la suivante va représenter les trois suivantes.

Génération de code html5 :

<!DOCTYPE html>

<html lang="fr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>HTML CSS 1</title>

</head>

<body></body>

</html>

On corrige juste la langue et le titre.

Table des matières

[Informations diverses 1](#_Toc518335150)

[Formateurs 1](#_Toc518335151)

[Communication 1](#_Toc518335152)

[Environnement de travail 1](#_Toc518335153)

[Visual Studio Code 1](#_Toc518335154)

[Quelques touches de raccourci pratiques dans Visual Studio Code 4](#_Toc518335155)

[Personnalisation de VSCode 4](#_Toc518335156)

[Chrome et son inspecteur 4](#_Toc518335157)

[PerfectPixel 5](#_Toc518335158)

[Le HTML 5](#_Toc518335159)

[<i>, <b>, <u> 8](#_Toc518335160)

[Les balises div et span 8](#_Toc518335161)

[La sémantique html5 9](#_Toc518335162)

[Position, float 9](#_Toc518335163)

[CSS 9](#_Toc518335164)

[CSS avec SASS 9](#_Toc518335165)

[L’intégration de htmlcss 1 énoncé exercice html-css 1) 10](#_Toc518335166)