Vue3

Ibrahim ALAME

28 janvier 2024

1 Objectifs du chapitre

Ce chapitre va être une introduction approfondie au HTML . Nous allons voir toutes les bases du HTML dans ce chapitre : l'en-tête, les métadonnées les balises, les hyperliens, la structure des documents HTML etc.

Vous apprendrez entre autre à utiliser Emmet avec Visual Studio Code. Vous pouvez passer rapidement si vous connaissez déjà bien le langage mais un rafraîchissement ne peut pas vous faire de mal !

2 Structure d'une page HTML

2.1 Création de notre première page HTML

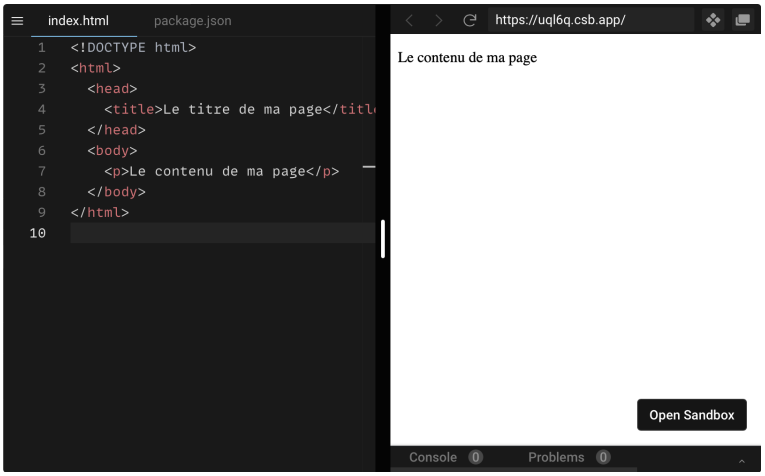
Créez un dossier pour notre cours où vous mettrez tous les projets que nous ferons ensemble : par exemple dyma-htmlcss.

1. Dans ce dossier créez un fichier index.html.
2. Ouvrez ensuite le dossier avec l'éditeur VS Code : soit en cliquant du droit sur le dossier, soit en ouvrant VS code et en allant dans File ¿ Open Folder.

Dans cette leçon, nous allons étudier la structure d'un document HTML.
Une page HTML minimale ressemble à cela :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Le titre de ma page</title>
  </head>
  <body>
    <p>Le contenu de ma page</p>
  </body>
</html>
```

Ce qui donne cela dans un navigateur :



Nous allons utiliser à chaque fois un éditeur en ligne pour que vous puissiez exécuter directement le code. Mais nous vous invitons à le reproduire aussi sur votre ordinateur pour bien apprendre les notations en même temps. Nous allons d'abord voir ce que sont les balises en HTML puis nous étudierons la structure du document.

2.2 Les éléments HTML

Prenons un élément HTML par exemple :

```
<title>Le titre de ma page</title>
```

- La première partie, `<title>` , est une balise ouvrante. Il s'agit du nom de l'élément HTML ici title encadrés par des chevrons. Elle permet de déclarer le début de l'élément HTML ainsi que son type.
- La deuxième partie est le contenu, ici Le titre de ma page, qui est simplement du texte dans notre cas.
- La troisième partie, `</title>`, est une balise fermante. Elle permet de déclarer la fin de l'élément HTML.

2.3 L'imbrication

Vous pouvez voir que les éléments HTML que nous avons déclarés s'imbriquent les uns dans les autres. Par exemple, `<title>`, est imbriqué dans `<head>`, lui-même imbriqué dans `<html>`. Cette imbrication permet de créer la structure du document en déclarant quels éléments HTML sont inclus dans tels élément HTML.

2.4 La structure d'une page HTML

Nous allons maintenant étudier ligne par ligne la structure du document.

2.4.1 Le type de document

La première ligne d'un fichierHTML est la suivante :

```
<!DOCTYPE html>
```

Il signifie au navigateur qu'il s'agit d'un document HTML à la version au moins 5. Comme nous l'avons vu, il n'y a plus aujourd'hui de version HTML, il existe uniquement un HTML Living Standard qui évolue constamment. Cette ligne, et l'extension .html du fichier, permettent également aux éditeurs de pouvoir parser le document pour mettre les couleurs et activer l'autocomplétion HTML.

2.4.2 L'élément racine html

La deuxième ligne est une balise ouvrante `<html>`. L'élément `<html>` contient tout le code de la page et est appelé élément racine.

2.4.3 L'en-tête head

L'élément HTML head est l'en-tête du document. Il ne peut y en avoir qu'un par page. Il contient les éléments que vous voulez inclure dans la page HTML sans qu'ils soient affichés aux utilisateurs. Il peut contenir de nombreux éléments que nous verrons au fur et à mesure. Il contient notamment le titre de la page.

2.4.4 Le titre de la page title

L'élément title permet de déclarer le titre de la page dans l'en-tête head. Il s'affiche dans l'onglet du navigateur. Il est également utilisé comme description de la page pour l'ajout en favori dans le navigateur.

2.4.5 le corps body

L'élément body contient tout le contenu à afficher aux utilisateurs de votre page. Il contiendra le texte, les éventuelles images, vidéos, musiques etc.

2.5 Utilisation des extensions Prettier et Emmet

Nous allons commencer à utiliser deux extensions dans VS Code.

2.5.1 Emmet

Cette extension est incluse dans VS code par défaut. Elle permet d'utiliser des raccourcis extrêmement utiles pour la génération de code HTML et CSS.

Nous allons l'utiliser tout le temps dans la formation et vous expliquerons à chaque fois comment gagner du temps en vous donnant les syntaxes appropriées.

2.5.2 Prettier

Prettier permet de mettre en forme le code automatiquement lorsque vous le sauvegardez. C'est une extension très utile et nous vous invitons à l'installer dans VS code ;

1. Allez dans Extensions puis recherchez Prettier. Ensuite cliquez sur Install. Prettier a plus de 14 millions de téléchargement sur VS code ce qui en fait l'une des extensions les plus utilisées.
2. Dans VS Code allez ensuite dans File puis Preferences puis Settings. Recherchez format on save et cochez la case.

3. Toujours dans VS Code, et toujours dans Settings. Recherchez cette fois Default Formatter et dans la liste déroulante sélectionnez Prettier - Code Formatter esbenp.prettier-vscode.

Prettier fonctionnera maintenant à chaque fois que vous sauvegardez !

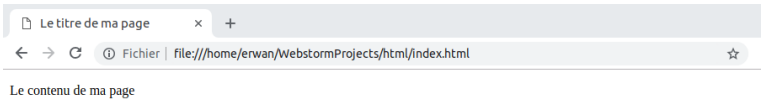
2.6 Affichage d'une page HTML

2.6.1 Ouvrir un document HTML local dans le navigateur

Vous pouvez afficher un document HTML à l'aide de n'importe quel navigateur.

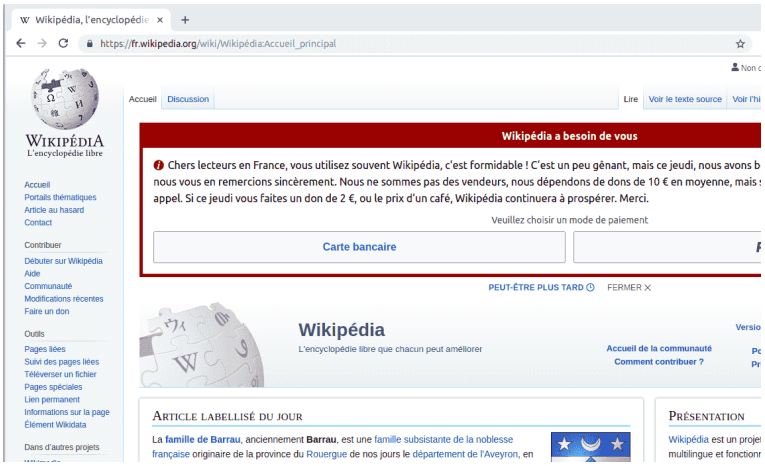
- Bien qu'il existe plus de 100 navigateurs différents, les plus performants sont actuellement probablement Chrome, Firefox et Brave.
- Dans la formation nous utiliserons Chrome, mais Firefox et Brave sont également très biens.
- Pour ouvrir un fichier .html avec un navigateur, cliquez droit et faites ouvrir avec Chrome par exemple. Ou ouvrir avec une autre application et sélectionnez un navigateur.

Vous aurez ensuite :



2.6.2 Ouvrir un document HTML depuis le Web

Lorsque vous ouvrez une page HTML en utilisant le Web et Internet, votre navigateur va envoyer une requête HTTP à l'adresse demandée, par exemple https ://fr.wikipedia.org/. Les serveurs Web de Wikipedia vont alors traiter la requête et retourner une réponse HTTP que votre navigateur interprétera pour afficher la page. Par exemple :



2.6.3 Ouvrir un document HTML local en utilisant un serveur local

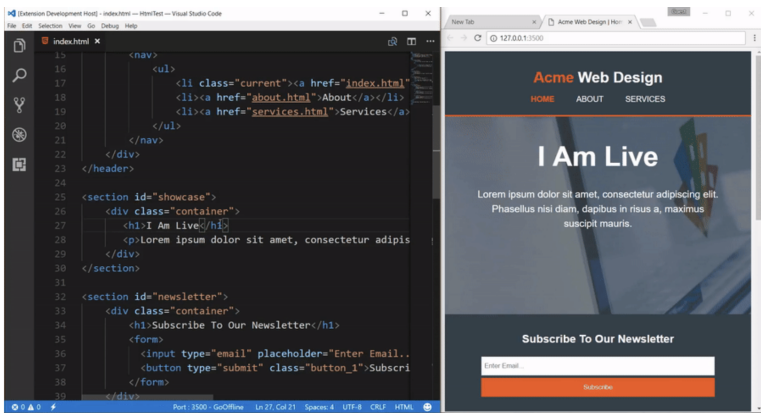
Il existe un inconvénient à ouvrir un fichier HTML directement dans votre navigateur : les changements ne sont pas pris en compte. Autrement dit, lorsque vous effectuez des modifications dans VS code du fichier HTML et que vous sauvegardez (avec Ctrl + s), la page affichée dans le navigateur ne se met pas automatiquement à jour. Il faut nécessairement rafraichir la page dans le navigateur pour voir les changements.

Il existe heureusement une solution : utiliser un serveur Web local qui va suivre les changements effectués dans votre fichier et mettre automatiquement la page à jour. Pour cela, allez dans VS code, puis dans Extensions puis recherchez Liver server et installez la.

Cette extension permet de lancer un serveur sur 127.0.0.1 qui est une adresse IP spéciale faisant référence à votre ordinateur. La requête n'ira ainsi pas sur Internet et vous n'avez pas besoin de connaître l'adresse IP réelle de votre ordinateur. Cette adresse IP a un alias, c'est-à-dire un nom de domaine qui y équivaut, localhost. Tapez localhost ou 127.0.0.1 est donc équivalent.

Live server permet de lancer un serveur de développement local qui a permet automatiquement de mettre à jour la page en cas de changement dans le code (c'est ce qu'on appelle le live reload). Pour lancer une page HTML avec l'extension vous pouvez soit cliquer droit dans le fichier dans l'éditeur et faire Open with Live Server. Vous pouvez également cliquer sur Go Live en bas à droite. Vous pouvez enfin utiliser le raccourci : Alt +L puis Alt + O.

Voici un exemple :



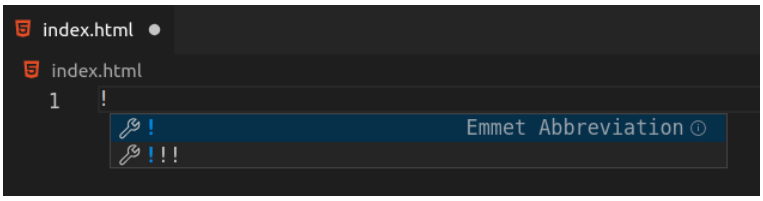
3 Les attributs HTML et les métadonnées

3.1 Génération d'un document HTML avec Emmet

1. Nous allons commencer par générer un document HTML avec Emmet.
2. Supprimez ce que vous aviez dans index.html.
3. Entrez ensuite :

```
!
```

Vous aurez une autocomplétion :



Vous pouvez alors presser la touche entrée et obtiendrez le document suivant :

```
<!DOCTYPE html>
<html lang="en">
  <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>Document</title>
  </head>
  <body></body>
</html>
```

- Si vous n'avez pas de suggestion qui apparait, vous pouvez presser les touches Ctrl + Espace.
- Si ce raccourci ne se déclenche pas sur un Tab allez dans Fichier ¿ Préférences ¿ Paramètres et recherchez Emmet. Cochez la case Emmet : Trigger Expansion On Tab.

3.2 Les attributs

Les éléments HTML peuvent avoir des attributs. Un attribut est par exemple lang :

```
<html lang="fr">
```

Les attributs contiennent des informations supplémentaires sur l'élément HTML sans qu'elles n'apparaissent directement sur la page. Pour créer un attribut il y a trois règles :

1. Il faut mettre un espace entre le nom de l'élément HTML dans la balise ouvrante, et le nom de l'attribut, ou entre l'attribut précédent si il y a plusieurs attributs.
2. Il faut donner un nom à l'attribut et ajouter le signe égal.

3. Il faut donner une valeur à l'attribut. Il faut la placer à l'intérieur de guillemets, qui peuvent être simples ou doubles dès lors que la valeur contient un des caractères suivants " ' ` = < >. Cependant, vous trouverez quasiment systématiquement des guillemets autour de la valeur et nous vous invitons donc à toujours en mettre.

Parfois, il est possible d'utiliser une notation raccourcie avec les attributs booléens :

```
<input type="text" disabled>
```

Ces attributs ne peuvent avoir qu'une seule valeur qui est la plupart du temps le nom de l'attribut.

3.2.1 L'attribut lang

Dans le document généré par Emmet, nous utilisons donc l'attribut lang sur l'élément html. Cet attribut permet de définir la langue principale du document. Par défaut, il est défini à en pour english mais mettez fr pour une page en français. La langue est utilisée pour l'indexation par les moteurs de recherche.

3.3 Les métadonnées

Les métadonnées sont des données apportant des informations sur d'autres données. Le langage HTML permet d'utiliser des métadonnées dans un document en utilisant la balise meta.

3.3.1 Le charset

La métadonnée permet de définir l'encodage des caractères du document. C'est le jeu de caractères à utiliser pour convertir les valeurs numériques en caractères. Ici, nous utilisons UTF-8 qui permet d'encoder n'importe quel caractère (latin, japonais, chinois, caractères spéciaux). C'est l'encodage le plus utilisé sur le Web, et plus de 95

3.3.2 Le viewport

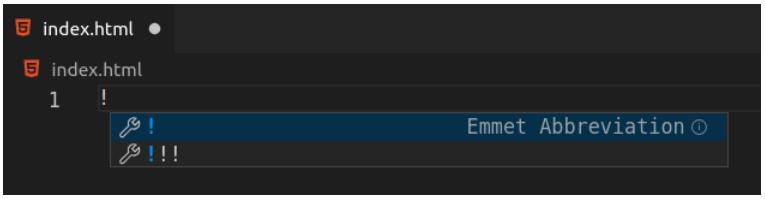
Le viewport est la surface de la fenêtre du navigateur. Nous allons étudier ce que veut dire cette balise meta :

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

Elle est utile pour l'affichage sur les mobiles et les tablettes. Pour bien comprendre il faut déjà voir les deux surfaces existantes pour les appareils mobiles et tablettes.

- La première est la surface physique qui est le nombre de pixels sur l'écran, qui est également appelée la définition de l'écran.
- La deuxième est la surface utilisable qui est le nombre de pixels virtuels que le terminal pense pouvoir afficher.

Ces surfaces ne sont pas égales sur les mobiles pour des raisons d'affichage. La taille du viewport d'un mobile n'est ni égale à sa surface physique, ni égale à sa surface virtuelle, et ce pour pouvoir afficher la plupart des pages Web. Elle est supérieure afin de pouvoir appliquer un dézoom de la page pour l'afficher en entier. Sans la balise viewport, le navigateur mobile réagit comme sur un ordinateur et met la page à l'échelle de l'écran, ce qui rend impossible la lecture du contenu. Sans la balise nous aurions l'affichage de gauche, et avec la balise l'affichage de droite :



La balise viewport permet donc d'indiquer au navigateur mobile comment ajuster les dimensions et l'échelle de la page à la largeur de l'appareil.

- `width="device-width"` permet de définir la largeur de la fenêtre du viewport à celle de l'appareil.
- `initial-scale="1.0` permet de définir le niveau de zoom initial.

Autrement dit, ces deux paramètres permettent de forcer l'appareil mobile à ne pas changer le zoom ou prétendre que la largeur disponible est plus importante que réellement pour tenter d'afficher l'intégralité de la page. Il nous permettra plus tard dans la formation d'adapter notre page et nos styles en fonction de la largeur réelle disponible et donc de créer un site responsive comme sur l'image à droite.

3.4 La meta X-UA-Compatible

Ce tag permet d'empêcher le mode compatibilité sur Internet Explorer et de forcer le navigateur à utiliser le dernier moteur de rendu disponible (dernière version du moteur Edge). Ce tag n'est pas utile si vous ne souhaitez pas supporter Internet Explorer qui représente moins de 1% du marché des navigateurs.

3.4.1 La meta description

La meta description sert à l'indexation de cette page par les moteurs de recherche et les annuaires. Elle doit contenir une description du contenu de la page en une ou deux phrases. Depuis 2017, Google affiche entre 260 et 300 caractères. Il faut donc adapter la description en conséquence. Elle s'utilise de cette manière :

```
<meta name="description" content="La description de la page." />
```