01-02-03 – HTML et CSS

I – HTML : généralités

Le langage HTML est un langage _____ : il décrit ce qu'on veut afficher.

Il est basé sur l'encapsulation des données dans des _____ descriptives.

Il existe deux types de balises :

+ Les **balises normales** sont basées sur le fait d'avoir 3 parties : `<p>Bonjour à tous</p>`
  + `<p>` : une balise d'_____
  + `Bonjour à tous` : un contenu _____ (innerHTML en anglais)
  + `</p>` : une balise de fermeture

Exemples :
`<p> </p>` _____
`<h1> </h1>` _____
`<h6> </h6>` _____
`<ul> </ul>` _____
`<ol> </ol>` _____
`<li> </li>` _____

+ Les **balises orphélines** ne possèdent que la partie ouverture et n'ont ni contenu interne ni fermeture.

Exemples :
`<br>` _____
`<hr>` _____
`<img>` _____

+ Les **balises d'ouverture** peuvent fournir des informations en renseigant certains _____, de sortes de paramètres :

Balise `<img src="url_de_image" width="500" alt="texte_de_remplacement">`
Balise `<a href="url_de_destination" target="_blank">Texte du lien</a>`
Balise `<p style="background-color:yellow;">Des phrases</p>`

II – HTML : les 3 catégories d'affichage

On notera également qu'il a 3 catégories de balises en terme d'affichage :

+ les balises **block** : elles provoquent un passage à la ligne. On peut choisir `width` et `height`.
`<p></p>` `<h1></h1>` `<li></li>` `<div></div>` …

+ les balises **inline** : elles ne provoquent pas de passage à la ligne et on ne peut pas régler `width` et `height.`
`<a></a>` `<span></span>` `<strong></strong>` `<em></em>` `<mark></mark>` …

+ les balises **inline-block** : elles ne provoquent pas de passage à la ligne mais on peut régler `width` et `height.`
`<img>`

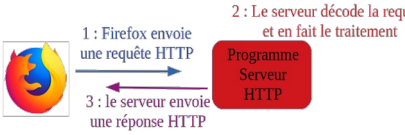
III – Principe du HTTP

0 - Le serveur écoute.

1 - Le client lance une **REQUETE HTTP** au serveur.

2 - Le serveur répond avec une **REPONSE HTTP** contenant éventuellement un **message HTML**.

3 - Le client reçoit et analyse cette réponse.



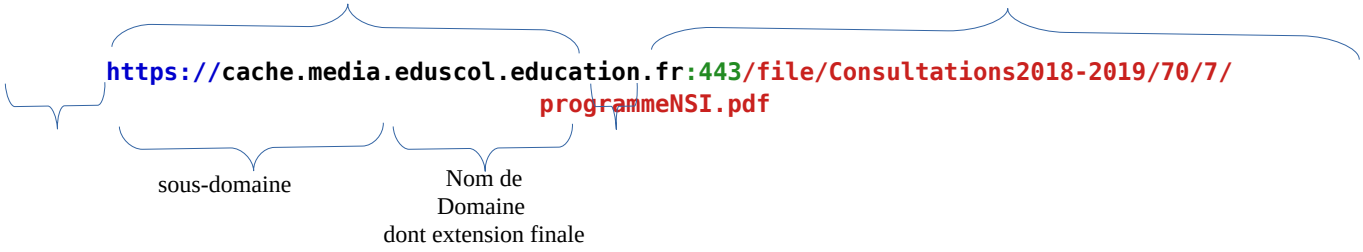
Si la réponse HTTP contient un message HTML, le client lancera peut-être d'autres **REQUETES HTTP** lorsqu'il rencontra une balise `<img>`, `<video>`, `<link>` pour un fichier CSS à part ou encore `<script>` lors de son analyse.

IV – URL : Uniform Ressource Locator

Trois façons de localiser une ressource :
→ en fournissant **l'URL complète :** localisation sans ambiguïté sur Internet.
→ en fournissant **une adresse absolue :** localisation sans ambiguïté depuis la racine du serveur actuel.
→ en fournissant **une adresse relative :** on localise la ressource par rapport à la localisation du fichier HTML actuel.

URL complète : composée du **protocole**, de l'**adresse du serveur**, du **PORT** éventuel et de **l'adresse absolue de la ressource** demandée sur ce serveur.

Si la partie adresse de la ressource commence par **un slash /,** on dit qu'il s'agit d'une **adresse ABSOLUE** par rapport à la racine du serveur.



Adresse absolue : On fournit uniquement la partie correspondant à l'adresse absolue de la ressource sur le serveur.
Cette adresse **commence par un slash** qui indique « depuis la racine du serveur actuel ».

```
/file/Consultations2018-2019/70/7/programmeNSI.pdf
```

Adresse relative : On fournit l'adresse de la ressource par rapport à la localisation actuel de la ressource HTML initiale.
Cette adresse **ne commence pas par un slash.**

```
70/7/programmeNSI.pdf
```

L'adresse relative précédente pourrait correspondre à cette configuration de répertoires :

```
/ (Racine du serveur)
    file
  |     Consultations2018-2019
  |  |  fichierweb.html
  |  |     70
  |  |   |     7
  |  |   |   |  programmeNSI.pdf
```

V – Web Sémantique : donner du sens aux mots et à leur position dans la page

HTML intègre des moyens d'aider les robots des moteurs de recherche à comprendre de quoi parlent vraiment les pages.

```
<!DOCTYPE html> → on signale le type de ce qui suit en 100 % ASCII
<html lang="fr"> → contient l'intégralité du document, englobe tout le HTML

    <head> → contient des informations qu'on veut transmettre au navigateur, sans affichage direct
    </head>

    <body> → contient toutes les balises qui doivent provoquer un affichage

        <header> → l'en-tête de la page, la partie « haute » : souvent la banderolle...
        </header>

        <nav> → contient le menu de navigation
        </nav>

        <main> → contient le corps de la page, ce dont la page parle vraiment
        </main>

        <footer> → contient le pied de page : contact, informations diverses...
        </footer>

    </body>
</html>
```

On peut également donner des indications sur l'importance de tel ou tel contenu en utilisant les balises inline suivantes :
`<em></em>` : emphase (assez important). Par défaut en italique. `<strong></strong>` : important. Par défaut en gras.
`<mark></mark>` :très important. Par défaut en surligné jaune fluo.

VI – CSS : Cascading Style Sheet

Le HTML précise le contenu de la page. Le CSS précise l'apparence que doit avoir ce contenu.
On peut inclure du style CSS de 3 façons :
→ directement dans la balise en utilisant l'attribut style. A éviter et à reserver au prototypage.
→ dans une balise **<style>** intégrée à la page HTML. A éviter car le style ne s'appliquera qu'à cette page.
→ dans un fichier CSS à part localisé à l'aide d'une balise **<link>** situé dans la balise **<head>** de la page HTML.

```
<!DOCTYPE html>
    <html lang="fr">
        <head>
            <meta charset="UTF-8" />                          → orpheline, d'où le / optionnel final
            <title>CSS à part (page 2)</title>
            <link rel="stylesheet" href="style_1.css" />     → orpheline, d'où le / optionnel final
        </head>
        <body>
             …
        </body>
    </html>
```

Le CSS est un langage différent de HTML.

On précise le type des balises sur lesquelles on veut agir et on place entre accolades les styles CSS qu'on veut leur imposer. Exemple :

```
header, footer                  → Toutes les balises header et footer
{
      background-color:#AAAA00;  → auront un fond coloré jaune (rouge et vert à fond FF)
      color:#5500AA;             → et une écriture de texte plutôt bleu-violet (un peu de rouge 55 et pas mal de bleu AA)
}
```

Pour vous renseigner sur des effets style CSS, une recherche « w3school + effet voulu ».

VII – CLASSE : Différenciation d'effets CSS sur des balises de même type

Si vous ne souhaitez pas que toutes les balises p aient le même style CSS, il faut les différencier en utilisant un attribut HTML nommé **class.** On pourra alors faire agir le CSS sur les balises ayant cette classe.

Code HTLM
```
<p>Paragrahe1.</p>
<p class="typeA">Paragraphe 2.</p>
<p class="typeA">Paragraphe 3.</p>
```

Code CSS
```
.typeA        → on notera la présence du point pour signaler CLASS typeA et pas balise typeA
{
   font-style: italic;
   background-color: grey; color : white;
}
```

VIII – Balises génériques

Pour agir sur un ensemble de balises, on peut les englober dans **des balises-conteneurs génériques** et agir sur le style CSS de la balise conteneur elle-même : les enfants héritent des propriétés CSS du parent-conteneur.

La balise générique est une balise block. La balise générique est une balise inline.

```
<div class="typeA">        → les balises div aura le style de la classe typeA.
      <ol>
            <li>Premier choix du QCM</li>          → les enfants contenus aussi !
            <li>Deuxième choix du QCM</li>
      </ol>
</div>
```