Rappels de SNT sur le{{< a link="/docs/SNT_2nde/pages/page4/web/" caption="Web" >}} ## des tâches différentes Les langages web se partagent les tâches : Une bonne pratique dans le développement d'un site internet (côté front-end, ce qui est executé sur la machine du client) consiste à utiliser :

- HTML pour le **contenu** , et la **structure** (avec un contenu correctement balisé, sémantique, accessible)
- CSS pour la mise en forme et le design de la page, les **propriétés** relatives à chaque élément de Style (balise html).
- et JavaScript pour gérer les **interactions** (qui peuvent éventuellement amener à modifier le contenu via les méthodes du DOM[^1]).

Le langage HTML est un langage constitué de *balises*, comme par exemple : `<title>le titre de ma page</title>` qui permet d'afficher le titre de la page dans l'onglet du navigateur.

Les éléments mis dans le programme à l'aide de ces balises vont permettre d'ajouter et structurer le contenu de la page : des titres de différents niveaux, du texte, des images, des médias (sons, videos),mais surtout, des hyperliens :

- entre les pages du sites
- vers les pages de sites externes

> Le web est justement basé sur l'utilisation de ces hyperliens, qui permettent de naviguer de pages en pages, sur internet.

Part 7

Le HTML et la page Web

7.1 Les balises principale

Def : un element HTML, comme p, h1, a, ... est défini en HTML par des balises.

Les balises sont les instructions en langage HTML des éléments structurants de la page web.

Les balises peuvent comporter des **attributs**, auxquels on associe des **valeurs**.

Les balises se distinguent entre celles qui n'ont pas d'attribut obligatoire, et celles qui en ont. Un attribut va ajouter des fonctionnalités à l'élément HTML, et modifier son allure ou son fonctionnement.

7.1.1 Balises sans attribut

Les principales balises structurantes sont :

- h1, h2, h3, ... h6
- div

- p
- span
- ...

Elles s'utilisent de la manière suivante :

```
<h1>Texte</h1>
<p>Texte</p>
```

Il existe aussi des **balises qui doivent être combinées**, comme celles de listes :

- ul (parent) => li (enfants) pour une liste à puces
- ol (parent) => li (enfants) pour une liste ordonnée (numérotée)

```
<ul>
  <li>Premier</li>
  <li>Deuxieme</li>
</ul>

<ol>
  <li>Premier</li>
  <li>Deuxieme</li>
</ol>
```

Résultat :

- Premier
- Deuxieme

1. Premier
2. Deuxieme

7.1.2 balises avec attribut obligatoire *(à faire après les premiers exercices de la fiche de TD)*

Les attributs seront obligatoires pour certaines balises selon leur fonction. Certains attributs sont facultatifs et vont juste enrichir leur comportement.

lien **href** est un attribut obligatoire.

```
<a href="lien_vers_la_ressource.html">texte cliquable</a>
```

image Il s'agit d'une balise *orpheline*.

src est un attribut obligatoire.

```
<img src="lien_vers_la_ressource.jpg">
```

alt est un attribut facultatif, que l'on ajoute pour afficher un texte lorsque l'adresse de l'image est erronée, ou que celle-ci ne s'affiche pas.

```
<img src="lien_vers_la_ressource.jpg" alt="texte alternatif">
```

7.1.3 Remarque sur les adresses (href et src)

L'adresse placée pour l'attribut `href` ou bien `src` peut être *relative/absolue*, *locale/externe*.

Absolue Une adresse est *absolue* lorsque le chemin de celle-ci commence par `/`. Par exemple, pour un lien interne : `/docs/2nde/chimie/images/photo1.jpeg`

Et pour un lien externe : `http://nom_du_domaine.xyz/docs/2nde/chimie/images/photo1.jpeg`

Relative L'adresse est relative lorsqu'il n'y a pas `/`. Ce lien ne peut être qu'interne : `src = "chimie/images/photo1.jpeg"`

7.2 le squelette du document *(à faire après les exercices sur le DOM)*

Cette partie du script est le contenu minimum à mettre dans vos pages HTML :

```
<!DOCTYPE html>

<html>
    <head>

        <!-- encodage des caractères -->
        <meta charset="UTF-8">
        <!-- Titre -->
        <title>Titre de la page web</title>
        <!-- Lien vers la feuille de style -->
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>

    <body>
        <!-- corps de la page -->

    </body>
</html>
```

Le **doctype** indique au navigateur la version HTML utilisée par la page (ici HTML5).

L'élément racine `<html>` : C'est lui qui va recueillir les deux principaux éléments de la hiérarchie : `<head>` et `<body>`.

À ce niveau, le code HTML est alors divisé en deux parties.

On peut lui associer l'attribut langue, précisant la langue utilisée dans le document :

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

L'en-tête (élément `<head>`) donne l'encodage des caractères (ici UTF-8).

Préciser l'encodage des caractères est primordial pour exploiter la bonne page de code et ne pas se retrouver avec les caractères spéciaux ou accentués. Le choix de l'UTF-8 est désormais préconisé par le W3C pour tous les protocoles échangeant du texte sur internet (dont HTML).

On peut aussi y ajouter des éléments `<link>` et `<script>` :

- link : permet de mettre en relation la page avec d'autres documents externes : `<link rel="stylesheet" type="text/css" href="style.css">`
- `<script>` permet d'ajouter des scripts (JavaScript) qui vont s'éxécuter côté client dans le navigateur dès leur chargement.

Si vous avez un script qui est très gros mais indépendant, il est préférable de le placer tout à la fin, afin de ne pas retarder le navigateur dans sa construction de l'arbre du DOM et de l'affichage de la page.

7.3 Imbrication et filiation des balises

La premiere page de l'histoire du Web ne présentait pas de mise en forme particulière. Le code ne contenait pas de balises *fermantes*. Les pages n'étaient pas hierarchisées.

7.3.1 Mise en boîte

Avec la norme actuelle du HTML, les pages présentent un structure qui propose un parcours de lecture, un design. L'idée est de capter l'attention du lecteur le plus longtemps possible, de transmettre une information dans un certain ordre.

Et pour mettre en forme des éléments à différentes positions, il faut insérer les éléments HTML dans des "boîtes".

D'où l'usage de balises dite "container", qui vont permettre ces structures en lignes et colonnes, comme `div`, `section`, `article`, ...

La boite principale sera l'élément `body`, qui contiendra toutes les autres.

7.3.2 Arbre du DOM

L'imbrication des balises traduit le lien (la filiation) entre les éléments.

On représente la structure d'un document html à l'aide d'un arbre. On parle d'**arbre DOM** (Document Object Model) du document.

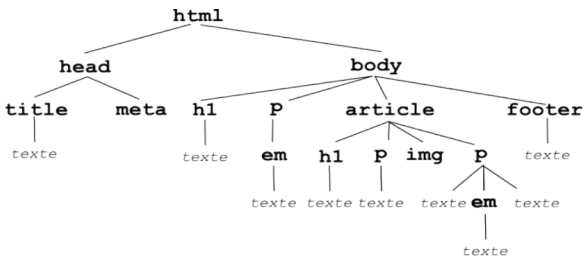

FIGURE 4 – Exemple d'arbre du DOM

Pour cet exemple : Le **nœud** article est le **père** des noeuds h1, p, img et p. Les nœuds h1, p, img et p sont les **fils** du nœud article.

Observons 2 autres exemples. ISSU DE LA **PAGE SNT : cours HTML**

Dans le premier, les éléments de la page se mettent l'un sous l'autre. Ici, les balises ne présentent pas de hierarchie entre elles. Elles sont toutes au même niveau.

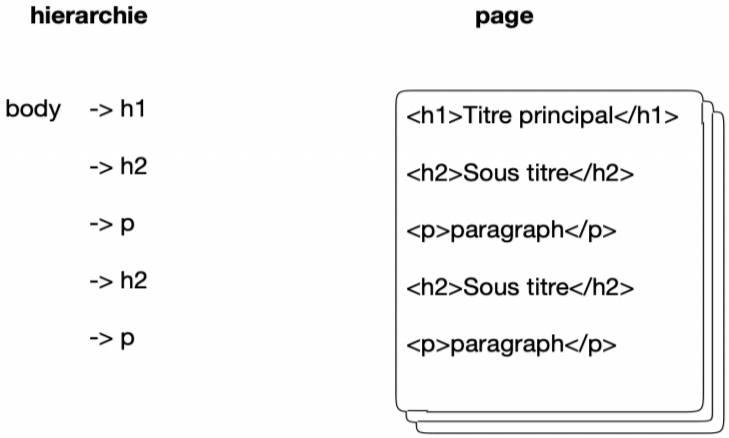

FIGURE 5 – balises sans hierarchie

Dans ce deuxieme exemple : l'élément `h1` est au même niveau de hierarchie que l'ensemble suivant, constitué de 2 colonnes. Tout cet ensemble sera mis dans un container `div`.

Les 2 colonnes contiennent pour la premiere, les éléments `h2` et `p`, et pour la deuxieme, une image.

Chacune de ces 2 colonnes devront être mises dans un élément parent, un autre container `div` par exemple. Cela va amener la descrition suivante :

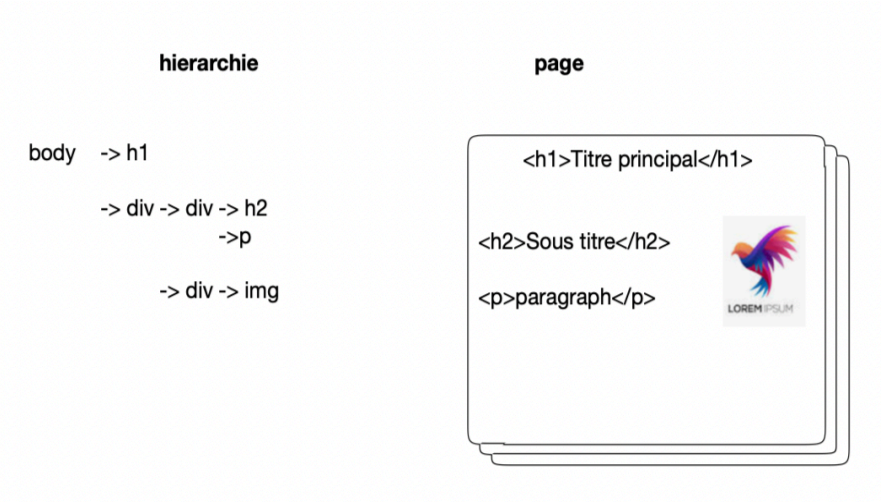

FIGURE 6 – balises avec hierarchie

Le schéma suivant montre la position de ces *boîtes* dans la fenêtre :

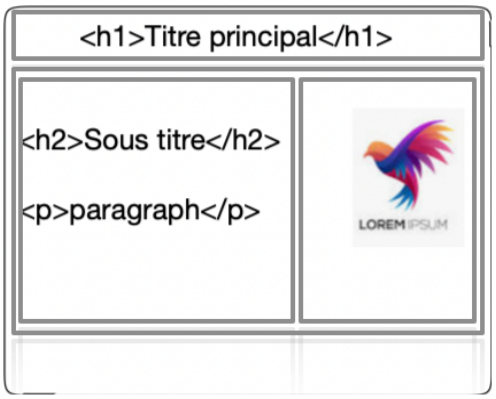

FIGURE 7 – elements dans la page du navigateur

L'image en forme d'arbre, montre la hierarchie (généalogie) entre ces éléments.

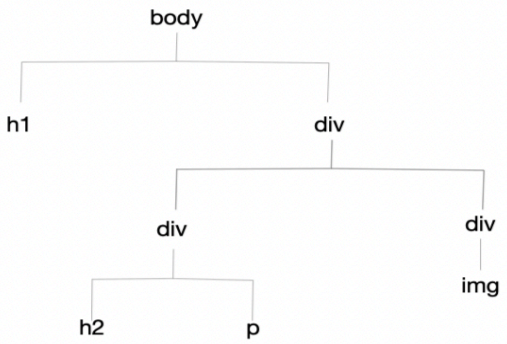

FIGURE 8 – balises sans hierarchie

7.3.3 Quand faut-il recourir à des balises parentes ?

Supposons que l'on ait besoin de structurer la page comme sur l'image suivante :

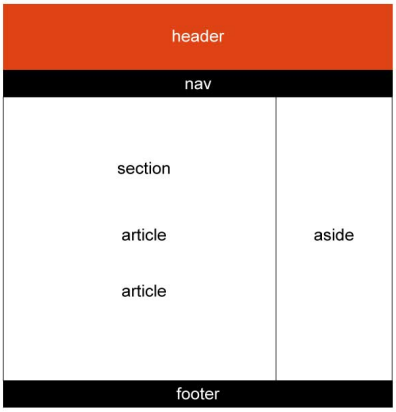

FIGURE 9 – page avec sections

Les éléments *containers* s'ils sont de type paragraphes, sections, header, nav, article, ... vont se placer naturellement l'un sous l'autre. Il n'y a pas besoin d'imbriquer les balises.

Par contre, s'il faut disposer côte à côte 2 éléments containers, pour faire 2 colonnes, il faudra les disposer à l'interieur d'un autre élément parent. Puis modifier leur disposition naturelle (paramètre display).

Pour cet exemple, au niveau de l'élément au fond blanc, il faudra :

- un container parent, par exemple un élément `div`
- (colonne gauche) : un container fils de `div` qui contiendra lui-même les éléments fils 'section , article , article'
- (colonne droite) : un élément fils de `div` qui sera l'élément `aside`

Part 8

Le CSS

Les sigles « CSS » sont l'abréviation de « Cascading StyleSheets » ou « feuilles de styles en cascade » en français.

8.1 Les 3 manières d'écrire des declarations en css

8.1.1 directement dans la balise

avec l'attribut optionnel `style`, on peut mettre les déclarations dans la balise elle-même. Pratique pour de petits tests. Cette modification ne concerne alors que l'éléments choisi.

C'est un peu comme la pratique de faire du formatage direct avec Writer, en utilisant la barre d'outils. [Montrer cette pratique avec le logiciel Writer]

8.1.2 Entre les balises `style`, dans la page css

Les règles sont alors appliquées à toute la page. Equivalent à la pratique dans un fichier Writer selon laquelle nous avons mis à jour les nouvelles propriétés pour touts les éléments d'un style défini.

8.1.3 Dans un fichier annexe

Ce fichier doit alors être pointé avec la balise `<link rel="stylesheet" type="text/css" href ="style.css">`

L'avantage de cette pratique, c'est de pouvoir réutiliser la feuille de style pour TOUTES les pages d'un même site. C'est l'équivalent de la pratique selon laquelle nous avons sauvegardé le fichier comme un modèle de feuille de style. Puis nous l'avons appliqué au prochain fichier ouvert.

8.2 Ecriture d'une déclaration en CSS

Une déclaration, c'est :

selecteur { règle ; }

Une règle, c'est : **propriété : valeur**

8.3 Les propriétés textuelles

Exemple de propriétés vues avec Writer	equivalent en css	valeurs possibles
police de caracteres	font-family	Helvetica, "Trebuchet MS", Verdana, sans-serif ;
taille	font-size	12px ; 1.0 em ; 80% ;
effets de caractere (font effect)	text-decoration	none ; underline ;
effets de caractere (font effect)	font-weight	normal ; bold ;
alignement	text-align	left, center, right
couleur de la police	color	color : #00a400 ; rgb(214, 122, 127) ;

Pour font-family : Les valeurs sont séparées par des virgules, indiquant chacune une police alternative. Le moteur choisira la première valeur pour laquelle la police correspondante est installée sur l'ordinateur

8.4 Les propriétés des boites

Exemple de propriétés vues avec Writer	equivalent en css	valeurs possibles
	display	block ; inline ; inline-block ; flex ; none ;
borders	border	solid ; 2px dotted ;
effet caractere couleur de fond	background-color	voir css texte color
	margin	2px
borders paddings	padding	10px 50px 30px 0 ;

The display CSS property sets whether an element is treated as a block or inline box.

- Les balises s'affichant en "block" (bloc) : elles prennent toute la largeur disponible et s'affichent avec un saut de ligne avant et après. On peut les dimensionner avec les propriétés width et height. Elles n'accèptent pas vertical-align. Ils peuvent avoir tout type de contenu, inline ou block.
- Les balises s'affichant en "inline" (dans la ligne) : elles prennent uniquement la largeur dont elles ont besoin, sans ajouter de saut de ligne. Elles acceptent vertical-align. Leur taille s'adapte au contenu. Celui-ci ne peut être de type block.
- cas particulier de `img` : Sa propriété display par défaut vaut inline, mais ses dimensions par défaut sont définies par les valeurs intrinsèques de l'image, à la façon de inline-block. Il est tout à fait possible d'utiliser les propriétés border/border-radius, padding/margin, width, et height sur une image.

Les types de valeur des propriétés width et height sont :

- soit des valeurs de type « longueur » qui vont être généralement exprimées en px ou en em ;
- soit des valeurs en pourcentage, auquel cas le pourcentage donné sera relatif à la dimension de l'élément parent.

8.5 Selection, compléments

selecteur	effet
td > img	Plusieurs sélecteurs séparés par un caractère supérieur : inclusion directe.Les balises img directement incluses dans une cellule de tableau sont concernées. Il ne doit pas exister de niveaux intermédiaires.
td + img	Plusieurs sélecteurs séparés par un signe + : vient après.Désigne les balises img venant après une balise p.
p ~ img	Plusieurs sélecteurs séparés par un signe ~ : une balise qui suit une autre (directement ou pas)
:hover	Le caractère : (deux points) introduit une pseudo-classe. La pseudo-classe désigne un élément lorsqu'il se trouve dans une certaine situation : ici lorsqu'il est survolé par la souris.Voir la liste complète des pseudo-class.
img[width]	Désigne les balises img comportant un attribut width quelle que soit la valeur de ce dernier. On teste donc seulement la présence de l'attribut.
p[lang = "fr"]	Ici on limite la portée aux balises p qui comportent l'attribut lang avec la valeur "fr".p lang="fr"

8.6 Codage de la couleur

page SNT images, TP données couleur EXIF voir sur http ://www.proftnj.com/RGB3.htm