

HTML / CSS

Hyper Text Markup Language

Cascading Style Sheets

Université D'Evry Val d'Essonne

Hicham EL KADIRI

Au sommaire

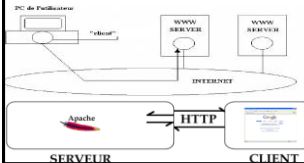
1. Architecture client-serveur
2. Les bases de HTML et XHTML
3. Les méta-informations en HTML
4. Les listes
5. Les images
6. Les tableaux
7. Les cadres
8. Formulaires HTML
9. Feuilles de style
10. HTML 5

Architecture client-serveur

L'interface **WWW** (World Wide Web) est basée sur l'architecture client-serveur. Ce type d'architecture fait intervenir deux programmes qui communiquent entre eux grâce au protocole **TCP/IP**.

- le premier, le « **client** » est un programme qui tourne sur le PC de l'utilisateur (navigateur internet Explorer, Mozilla,...) . Il est utilisé pour entrer en communication avec le réseau et son rôle est d'envoyer une demande d'information ou une requête.
- Le second, le « **serveur** » est chargé sur la machine sur laquelle sont stockées les informations et aura pour mission de renvoyer les informations demandées par le « client ».

Architecture client-serveur



Architecture client-serveur

Clients et serveurs dialoguent de la façon suivante :

- 1 l'utilisateur à travers l'interface graphique de son navigateur, le logiciel client, exprime symboliquement une requête à l'adresse du serveur (par un clic de souris sur un hyperlien ou sur un bouton submit d'un formulaire ...)
- 2 le client traduit la demande conformément au protocole invoqué dans la requête (le plus souvent HTTP); puis attend la réponse du serveur.
- 3 le serveur traite les données si nécessaire et retourne le résultat, le plus souvent le document demandé avec les fichiers annexés (images, sons ...)
- 4 le navigateur client met en forme cette réponse et la présente de façon convenable, compte-tenu des ressources de la station.

Architecture client-serveur

Le « **client** » par l'intermédiaire du réseau Internet contacte le serveur en lui donnant tous les paramètres nécessaires pour qu'il puisse localiser l'information (répertoire, nom de fichier...). Cette information est contenue dans l'**URL** « **U**niform **R**esource **L**ocator »

Le « **serveur** » renvoie une copie de la page à afficher accompagnée de toutes les instructions de formatage nécessaires, ainsi que les adresses de tous les liens associés à cette page. Lorsqu'un « **client** » comme le **browser Mozilla** reçoit une page **WWW**, il affiche la page et attend que l'utilisateur sélectionne un des objets (un mot par exemple) pour lesquels un lien a été établi sur cette page.

Les deux machines ne restent en communication que le temps du transfert d'information ce qui permet de réduire les problèmes d'encombrement du réseau.

Uniform Resource Locator

Adressage des documents

- Pour nommer et localiser une page - 3 questions :
 - Son nom ?
 - Sa position ?
 - Comment y accéder ?
- Solution : URL - (Uniform Resource Locator) adresse universelle de ressource contenant 3 parties :
 - Protocole (comment ?)
 - Nom DNS (où ?)
 - Nom du document (Comment y accéder ?)

URL TYPE : **http://nom_site/doc.htm**

Uniform Resource Locator

Interprétation par le navigateur

Le navigateur établit une connexion TCP pour chaque accès :

- En demandant au DNS l'adresse IP correspondant au nom du site.
- Avec un numéro de port défini selon le protocole (par défaut 80 pour HTTP, 443 pour HTTPS,...)
- En transmettant dans la partie données le texte de la requête (Ex : GET doc.html)
- URL : compléments

On peut préciser :

- Un numéro de port (site : nn)
- Une section d'un document (nom#section)
- Des arguments (nom? arg.)
- La page d'accueil d'un utilisateur (~utilisateur)

Exemple :

<http://www.site.fr:81/menu.html>
<http://www.site.fr/~paul/>
<http://www.site.fr/~paul/accueil.html#conclusion>

URL Uniform Resource Locator

La syntaxe d'un URL :

protocole://serveur:port/chemin_d'accès/fichier#etiquette?parametres

Exemple :

<https://dauphine.psl.eu/formations/masters/informatique>

HTTPS (HyperText Transfer Protocol Secure) est le protocole le plus utilisé sur Internet.

Le but du protocole HTTP est de permettre un transfert de fichiers (essentiellement au format HTML) localisés grâce à une chaîne de caractères appelée **URL** entre un navigateur (le client) et un serveur.

Client / Serveur

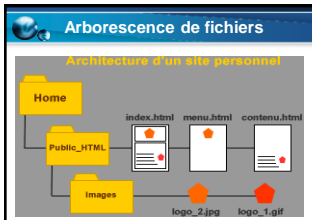
Une application est dite "Client/Serveur" lorsque son exécution s'effectue sur plusieurs machines.

Cela permet de :

- répartir la charge
- faciliter l'installation et la configuration des logiciels
- partager des ressources.

Une application Client/Serveur comprend généralement un :

- client en relation avec l'utilisateur
- serveur qui permet l'accès aux données
- middleware ou intergiciel qui permet de mettre en relation le client et le serveur en s'appuyant sur un protocole de communication.




Interprétation client

Vision côté Client

- Le Web est un ensemble de pages (documents) pouvant contenir des liens sur d'autres pages n'importe où dans le monde.
- Consultation des pages via un navigateur.
- L'utilisateur suit ces liens par simple clic.
- Notion d'hypertexte (information répartie).

Interprétation client

Navigateurs: 

-Le navigateur (browser) :

- Va rechercher la page demandée.
- Interprète les commandes de formatage.
- Mise en forme (police, gras, couleurs...).
- Recherche et affiche des images (gifs, jpg...).
- Affiche les animations (code Javascript, gifs animés...).
- Affiche la page correctement formatée.

Les bases de HTML

- Le **HTML** (« **HyperText Markup Language** ») est un langage dit de « marquage » (de « structuration » ou de « balisage ») dont le rôle est de formaliser l'écriture d'un document avec des balises de formatage. Les balises permettent d'indiquer la façon dont doit être présenté le document et les liens qu'il établit avec d'autres documents.
- Le langage HTML permet notamment la lecture de documents sur Internet à partir de machines différentes, grâce au protocole HTTP permettant d'accéder via le réseau à des documents repérés par une adresse unique, appelée URL.


Les bases de HTML

- On appelle **World Wide Web** (noté **WWW**) ou tout simplement **Web** la "toile virtuelle" formée par les différents documents (appelés « **pages web** ») liés entre-eux par des hyperliens.
- Les pages web sont généralement **organisées** autour d'une page d'accueil, jouant un point central dans la navigation à l'aide des **liens hypertextes**. Cet ensemble cohérent de pages web liées par des liens hypertextes et articulées autour d'une page d'accueil commune est appelée **site web**.
- Le Web est ainsi une énorme archive vivante composée d'une myriade de sites web proposant des pages web pouvant contenir du texte mis en forme, des images, des sons, des vidéos, etc.

HTML - Notion d'HyperTexte

■ **Hyperlien = Références entre documents**

- navigation d'un document à l'autre



Notion d'Hypermédia

- **Hypertexte enrichi du Multimédia**
 - Images, Sons, Vidéo, Animations

Notion de HTML

- Il est important de comprendre que le langage HTML est un standard, c'est-à-dire qu'il s'agit de recommandations publiées par un consortium international, le [World Wide Web Consortium \(W3C\)](http://www.w3.org/).
- il existe toujours une marge d'interprétation de la part des navigateurs, ce qui explique qu'une même page web puisse s'afficher différemment d'un navigateur Internet à l'autre.

XHTML: EXtensible Hyper Text Markup Language

- Le XHTML est une normalisation du HTML, qui le rapproche du XML
- Le XHTML est un HTML plus strict au niveau de ses règles :
 - Toute balise ouvrante doit comporter une balise fermante ou un trait oblique signalant que cette balise est ouvrante et fermante.
 - Les conteneurs ou balises peuvent s'imbriquer, mais pas s'entrelacer.
 - Les noms des balises et des propriétés doivent être en minuscules.
 - Les propriétés ou attributs doivent avoir une valeur.
 - Les valeurs des propriétés sont encadrées par des guillemets.
 - Le standard utilisé pour le code de la page doit être précisé

XHTML / HTML4 / HTML5

Le code suivant est correct en HTML 4, mais erroné en XHTML:

```
<P> Ceci est un paragraphe en HTML 4
<P> Ceci est un deuxième paragraphe en HTML 4
Ceci est une ligne blanche en HTML 4 <BR>
Ceci est une deuxième ligne en HTML 4
```

Le paragraphe commençant avec la balise <P> doit être en minuscule et clore par </p> pour être correct en XHTML :

```
<p> Ceci est un paragraphe en XHTML </p>
<p> Ceci est un deuxième paragraphe en XHTML </p>
Ceci est une ligne en XHTML <br />
Ceci est une deuxième ligne en XHTML
```

- Le trait oblique de fin
 indique à l'analyseur du navigateur de ne pas chercher une balise fermante. L'espace présent avant la barre oblique assure la compatibilité avec les anciennes versions des navigateurs.

Les balises solitaires du HTML4 en XHTML	
Balise HTML 4	Balise XHTML ou HTML5
<code></code>	<code></code>
<code><OPTION></code>	<code><option ... > ... </option></code>
<code><HR></code>	<code><hr ... /></code>
<code>
</code>	<code>
</code>
<code></code>	<code> ... </code>
<code><INPUT></code>	<code><input ... /></code>
<code><DT></code>	<code><dt> ... </dt></code>
<code><DD></code>	<code><dd> ... </dd></code>

Les propriétés sans valeur en HTML4	
Propriété HTML4	Propriété XHTML avec valeur
checked	checked="checked"
selected	selected="selected"
multiple	multiple="multiple"
compact	compact="compact"
disabled	disabled="disabled"
readonly	readonly="readonly"
ismap	ismap="ismap"
defer	defer="defer"
noresize	noresize="noresize"

Comment utiliser les balises **HTML**

- Une balise est un élément de texte (un nom) encadrée par le caractère inférieur (<) et le caractère supérieur (>), par exemple « `<H1>` ».
- Les balises HTML ne sont pas sensibles à la casse.
- Les balises HTML fonctionnent par paire afin d'agir sur les éléments qu'elles encadrent. La première est appelée « *balise d'ouverture* » (parfois *balise ouvrante*) et la seconde « *balise de fermeture* » (ou *fermante*). La balise fermante est précédée du caractère / :

` Ce texte est en gras `
 les balises `` et `` permettent de mettre en gras le texte qu'elles encadrent.

- Les balises HTML peuvent parfois être uniques : la balise `
` représente par exemple un retour à la ligne.
- Afin d'être le plus proche possible du standard **XHTML** (beaucoup plus stricte que le standard **HTML**), il est conseillé d'utiliser la notation suivante : `
`

Imbrication des balises

- Les balises HTML ont la particularité de pouvoir être imbriquées de manière hiérarchique afin de permettre le cumul de leur propriétés. En contrepartie le chevauchement de balises n'est pas toléré par le standard HTML. Voici un exemple de texte formaté avec des balises imbriquées :
`<i> Ce texte est en gras , et italique </i>`
 L'exemple ci-dessus donne le résultat suivant :
Ce texte est en gras , et italique
- En contrepartie l'exemple ci-dessous n'est pas correct :
`<i> Ce texte est en gras </i>, et italique `

Notion d'attribut

- Un attribut est un élément, présent au sein de la balise ouvrante, permettant de définir des propriétés supplémentaires. Les attributs se présentent la plupart du temps comme une paire clé=valeur, mais certains attributs ne sont parfois définis que par la clé.
- Voici un exemple d'attribut pour la balise `<p>` (balise définissant un paragraphe), permettant de spécifier que le texte doit être aligné sur la droite :
- `<p align="right"> Exemple de paragraphe </p>`

Chaque balise peut comporter un ou plusieurs attributs, chacun pouvant avoir (aucune) une ou plusieurs valeurs.

Espaces, saut de ligne

- Le langage HTML ne tient pas compte des espaces, des tabulations et des sauts de ligne. Cela permet notamment d'indenter le code HTML pour plus de lisibilité, sans modifier l'apparence de la page HTML dans le navigateur. Il existe une exception pour le code contenu dans des balises, dont l'objectif est justement de conserver le formatage du texte (espaces, sauts de lignes, etc.).

Préformaté: `<pre>...</pre>`

permet d'inclure un texte tel quel dans un document html sans devoir le convertir en html en conservant les espaces, les retours à la ligne et les tabulations.

Espaces, saut de ligne

- Le langage HTML possède par contre des éléments permettant expressément de définir chacun de ces éléments de mise en forme :
- Espace insécable** : il s'agit d'une espace ne pouvant être brisée par une fin de ligne. Sa représentation en HTML est ` `;
- Saut de ligne** : il s'agit d'un saut de ligne explicite. Sa représentation en HTML4 est `
` (`
` pour être conforme au XHTML ou HTML5).

Notion de document HTML

- Une page [HTML](#) est un simple fichier contenant du texte formaté avec des balises HTML. Par convention l'extension donnée au fichier est `.html`, mais une page web peut porter n'importe quelle extension.
- `.asp` pour une page générée dynamiquement en [ASP](#) (Active Server Pages) ;
- `.jsp` pour une page générée dynamiquement en [JSP](#) (Java Server Pages) ;
- `.php` pour une page générée dynamiquement en [PHP](#) ;
- `.pl` pour une page générée dynamiquement en [Perl](#) (Practical Extraction and Report Language) ;
- etc.

Déclaration du type de document

- Il est conseillé d'indiquer dans la page HTML, le type de document, c'est-à-dire une référence à la norme HTML utilisée, afin de spécifier le standard utilisé pour le codage de la page. Cette déclaration se fait par une ligne du type :

<!DOCTYPE html>

<HTML>

<HEAD>...</HEAD>

<BODY>Contenu de la page

</BODY>

</HTML>

La déclaration du document indique la DTD (*Document Type Definition*) utilisée, c'est-à-dire la référence des caractéristiques du langage utilisé.

- Pour HTML 5 : **<!DOCTYPE html>**
<html lang="fr-FR">

Exemple de Document HTML

Document HTML

Entête

Titre : « Document HTML »

Corps (couleur de fond = blanc)

Titre1 : « Titre »

Image : externe dans le fichier logo.gif

Ancre : « Sommaire » vers le fichier sommaire.htm

Titre : « Chapitre 1 »

Texte : « Texte Libre »

Exemple de Document HTML4

<HTML>

<HEAD>

<TITLE>Document HTML**</TITLE>**

<HEAD>

<BODY BACKGROUND="fond.jpg">

<H1 align=center>

****Titre

******</H1>**

<HR>

****Sommaire****

****Chapitre 1**
**

****Chapitre 2**
**

<H2 ID="CHAP1">Chapitre 1**</H2>**

Texte libre (Paragraphe 1.1) **
**

Texte libre (Paragraphe 1.2) **
**

<H2 ID="CHAP2">Chapitre 2**</H2>**

Texte libre (Paragraphe 2.1) **
**

<BODY>

</HTML>



Le document HTML minimum

Voici vos premières balises ou tags :

<html>	début d'un document
<head>	début de la zone d'en-tête
<title> </title>	définit le titre de la page
<meta>	métas informations
</head>	fin de la zone d'en-tête
<body>	début du corps
</body>	fin du corps
</html>	fin du document

Le document HTML vide

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="keywords" content="mot-clé, mot-clé">
  <title>Document HTML vide</title>
  <link rel="stylesheet" href="/style/styles.css">
</head>
<body>
</body>
</html>
```

Introduction aux méta tags

- Les métadonnées sont des informations situées au sein d'un document afin de le décrire. Les métadonnées sont ainsi utilisées par les moteurs de recherche lors du référencement de la page web. Grâce à ces balises non affichées, il est ainsi possible de renseigner des informations relatives à la page où au site afin de mieux en décrire le contenu, en particulier des informations sur le ou les auteur(s) du document, sa limite de validité, la langue utilisée, etc.

On appelle ainsi «**méta tags**» (en français «**méta-balises**») des balises spéciales situées dans l'en-tête du document HTML (c'est-à-dire la balise **HEAD**, située avant la balise **BODY**), afin notamment de fournir des informations permettant aux moteurs de recherche d'indexer la page web.

On distingue deux types de méta tags :

- Les métags **NAME**, permettant de décrire la page HTML :

```
<meta name="Nom du tag" content="Attribut">
```

- Les métags **HTTP-EQUIV**, permettant d'envoyer des informations supplémentaires au navigateur via le protocole HTTP :

```
<meta http-equiv="Nom du tag" content="Attribut">
```

Il est possible de renseigner plusieurs métags les uns après les autres dans l'en-tête de la page.

Les méta-informations

- Le tag META sera particulièrement utile pour faire reconnaître votre page par les moteurs de recherche du genre Google,

Définissez une description de votre page Web :

```
<meta name="description" content="description de votre page">
```

Définir l'auteur d'une page :

```
<meta name="author" content="Thomas LUC">
```

Définissez des mots-clés pour les moteurs de recherche :

```
<meta name="keywords" content="mot-clé, mot-clé, mot-clé, ...">
```

Cette balise indique au moteur de recherche que le contenu de CONTENT est une série de mots-clés qui définira plus finement votre page (Max 500 caractères dans content).

Actualiser le document toutes les 30 secondes :

```
<meta http-equiv="refresh" content="30">
```

Les méta-informations

Enchaînement de pages

```
<META HTTP-EQUIV="refresh" CONTENT=" 'x'; URL='adresse' ">
```

- Ce tag qui fait plutôt partie des trucs et astuces de HTML, appellera automatiquement une autre page (située à l'URL indiquée) après un délai d'attente de x secondes. Utilisé par exemple dans un frame, le rafraîchissement de la page permettra d'afficher à intervalle régulier différentes informations (publicitaires ou autres).

```
<META HTTP-EQUIV="Refresh" CONTENT=" '5';URL='accueil2.htm' ">
```

Ceci peut être intéressant pour l'ouverture automatique d'une séquence de quelques pages.

Définition de la fenêtre d'affichage

La fenêtre d'affichage est la zone visible par l'utilisateur d'une page Web. Cela varie selon l'appareil - il sera plus petit sur un téléphone portable que sur un écran d'ordinateur.

Vous devez inclure `<meta>` élément suivant dans toutes vos pages Web :

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

Cela donne au navigateur des instructions sur la façon de contrôler les dimensions et la mise à l'échelle de la page.

La `width=device-width` partie définit la largeur de la page pour suivre la largeur de l'écran de l'appareil (qui varie selon l'appareil).

`initial-scale=1.0` partie définit le niveau de zoom initial lorsque la page est chargée pour la première fois par le navigateur.

Voici un exemple de page Web sans la balise Meta Viewport et de la même page Web avec la balise Meta Viewport :

Affichage portable sans meta viewport – avec viewport



meta name="viewport"

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  
  <body>
    <h1>Chania</h1>
    <p>Pour comprendre cet exemple, vous devez ouvrir cette page sur un téléphone ou une tablette, ou redimensionner la fenêtre du navigateur.</p>
    
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut tamen atque eros.
    </p>
    <p>
      Ut nisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
    </p>
    <p>
      Ut autem vel eum irure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et justo odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugiat nulla facilisis.
    </p>
    <p>
      Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum.
    </p>
    <p>
      Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum.
    </p>
  </body>
</html>
```

Balise de présentation du texte

Taille des caractères	<code>Texte </code> (ex: y=2) ou y est un nombre qui va de 1 à 7 suivant la taille désirée	Texte
Combinaison: gras italique	<code><I> Texte </I></code>	Texte
Paragraphe	<code><p>... </p></code> en XHTML ou HTML5 <code><P></code> en HTML 4	passage à la ligne plus une ligne blanche
Break	<code>
</code> en XHTML ou HTML5 <code>
</code> en HTML4	passage à la ligne simple
Indentation	<code><dd>... </dd></code> ou <code><DD></code> en HTML4	retrait ou tabulation
Lignes horizontale	<code><HR WIDTH=60% SIZE=5 ></code>	longueur et épaisseur de la ligne
Centrer un texte	<code><center>...Titre centré...</center></code>	...Titre centré...

Exemple

```
<HTML>
<HEAD><TITLE></TITLE></HEAD>
<BODY>texte simple<BR>
<B>texte en gras</B><BR>
<STRONG>texte en gras</STRONG><BR>
<i>texte en italique</i><BR>
<EM>texte en italique</EM><BR>
<B></B><B>texte en gras et en italique</B></B><BR>
<FONT SIZE=5>texte</FONT>
<FONT COLOR="#0000FF">en bleu</FONT>
<!-- ceci est un commentaire-->
</BODY>
</HTML>
```

Visual Editor

Text simple
Texte en gras
Texte en gras
Texte en italique
Texte en italique
Texte en gras et en italique
Texte bleu
Texte

Caractères spéciaux

Le standard **HTML** demande de respecter le codage des caractères **ASCII**, c'est-à-dire que les caractères accentués ne sont pas autorisés. Il faut pour cela utiliser un caractère **&** et terminée par un point-virgule (;)

Code HTML	Caractère
"	"
&	&
<	<
>	>
 	(espace non séable)
&apost;	'
ç	ç
è	è
ê	é
È	È
Ê	Ê
ï	ï
Æ	Æ

Autres caractères utiles

Code HTML	Caractère
€	€
©	©
®	®
°	°
º	ª
«	«
»	»
µ	µ
¶	¶
¼	¼
½	½
¾	¾

Exemple
a < b if c > d équivaut à : a < b if c > d

Titres (Headlines)

```
<H1>Chapitre 1</H1>
<H2>Section 1.1</H2>
<H3>Sous Section 1.1.1</H3>
```

Exemple

```
<H1>Table of Contents</H1>
<H2><A href="#section1">Introduction</A></H2>
<A href="#section2">Some background</A><BR>
<A href="#section2.1">On a more personal note</A>
...the rest of the table of contents...
...the document body...
<H2 id="section1">Introduction</H2>
...section 1...
<H2 id="section2">Some background</H2>
...section 2...
<H3 id="section2.1">On a more personal note</H3>
...section 2.1...
```

Document HTML - Notepad++

Table of Contents

Introduction

Some background

On a more personal note

Section 1

Section 2

Section 2.1

Les Listes	
Code html	Affichage
Les listes de type "bullet": <pre> Portugal Grèce </pre>	Destination : <ul style="list-style-type: none"> • Portugal • Grèce
Les listes numérotées: <pre> Portugal Grèce </pre>	Destination : <ol style="list-style-type: none"> 1. Portugal 2. Grèce
Les listes "dictionnaire": <pre><ul style="list-style-type: none;"> IUT <small><td> Institut Universitaire de Technologie</td></small> BU <small><td> Bibliothèque Universitaire</td></small> </pre>	IUT Institut Universitaire de Technologie BU Bibliothèque Universitaire

Les Listes en HTML5

■ Liste sans ordre , Liste Ordonnée

```
<ul>
<li>Niveau 1</li>
<ol>
<li>Niveau 2</li>
<li>
<ol style="list-style-type: none;">
<li><small>10</small> Niveau 3</li>
<li><small>20</small> Niveau 3</li>
<li><small>21</small> Niveau 3</li>


```

Les listes	
<ul style="list-style-type: none"> la balise effectue le retour à la ligne; il est donc superflu de poser
 en fin de ligne. LI introduit toujours une nouvelle ligne dans une liste ordonnée ou non. Si on observe un retrait dans le texte qui suit une liste, c'est sans doute dû à l'oubli du marqueur de fin. Le paramètre TYPE de la balise permet de choisir la puce affichée pour chaque entrée de liste : (s'il est absent, le type disc est utilisé). 	

Les listes numérotées	
Code html	Affichage
<pre><ol type="1"> </pre>	1. 2.
<pre><ol type="A"> </pre>	A. B.
<pre><ol type="a"> </pre>	a. b.
<pre><ol type="I"> </pre>	I. II.

Les listes bulles	
Code html	Affichage
<pre><ul type="disc"> </pre>	<ul style="list-style-type: none">
<pre><ul type="circle"> </pre>	<ul style="list-style-type: none">
<pre><ul type="square"> </pre>	<ul style="list-style-type: none">

Les couleurs		
Voici les codes de quelques couleurs basiques :		
color	Hexadécimal	name
Bleu	#0000FF	blue
Blanc	#FFFFFF	white
Rouge	#FF0000	red
Gris	#C0C0C0	gray
Vert	#00FF00	green
Violet	#800080	purple
Jaune	#FFFF00	yellow
Noir	#000000	Black
Certains des codes de couleur précédents peuvent aussi être appelés par leur nom : https://htmlcolorcodes.com/fr/		

Les couleurs	
<ul style="list-style-type: none"> Paramètres de BODY <ul style="list-style-type: none"> <BODY BGCOLOR="couleur"> (= BackGround Color) Fixe la couleur du fond du document. <BODY TEXT="couleur"> Fixe la couleur du texte du document. En l'absence de TEXT, le texte est noir. <BODY LINK="couleur"> (= Link color) Fixe la couleur des liens du document. Par défaut, les liens sont bleus. <BODY VLINK="couleur"> (= Visited link color) Fixe la couleur des liens une fois qu'ils ont été activés (par défaut, ils sont violets). <BODY ALINK="couleur"> (= Activate link color) Fixe la couleur des liens actifs, c'est-à-dire la couleur des liens au moment où on clique dessus (par défaut, ils sont rouges). 	

Changer de couleur dans le texte	
<pre> texte </pre> <p>Le paramètre BGCOLOR fixe la couleur du texte dans tout le document.</p> <p>Pour changer la couleur du texte dans une partie de la page, on l'entoure de et .</p> <p>Rappel : dans la même balise, le paramètre SIZE peut modifier la taille des caractères.</p> <p>Exercice : (Je peux changer la couleur à mon gré dans une page)</p>	
	

Inclusion d'images

- Élément **IMG**
 -
 -
 - image incluse dans le document
 - Attribut optionnel
 - positionnement par rapport au texte
ALIGN = TOP, MIDDLE, BOTTOM, LEFT, RIGHT

Il permet de déterminer le type d'alignement de l'image avec le texte qui l'accompagne.
La valeur par défaut est ALIGN = BOTTOM, c'est-à-dire le texte aligné avec le bas de l'image

Forcer la taille de l'image
width=180 height=60
évite les formatages intempestifs au chargement du document

- Remarque
 - Type d'image généralement supporté
GIF, JPEG, GIF Animé, BMP, PNG

Image cliquable : éléments MAP et AREA

- Une image cliquable coté client est spécifié à l'aide de l'élément : **MAP**
- Elle est associée à un autre élément (ici **IMG**) via l'attribut **usemap** de celui-ci
- Pour voir comment mettre en œuvre une image cliquable, nous allons appliquer sur image de carte de France.

on considère son nom : carte.png

Image cliquable : éléments MAP et AREA

- Créer comme suit un élément **IMG** inclus dans un attribut **P**

```
<p> <IMG usemap="#Map" src="/image/carte.png" alt="région" /> </p>
```

L'ajout de l'attribut **usemap** dans la balise **IMG**, signale que l'image spécifiée par l'URI valeur de l'attribut **src** va être une image cliquable coté client, définie par un élément **MAP**

La valeur de l'attribut **usemap** de l'élément **IMG**, doit correspondre avec celle de l'attribut **name** de l'élément **MAP** associé.

Image cliquable : éléments MAP et AREA


- Ajouter ensuite dans l'élément **P**, après l'élément **IMG**, un élément **MAP** associé pour le moment vide :

```
<MAP name="Map" > </MAP>
```

La valeur de l'attribut **name** de l'élément **MAP** correspond bien à celle de l'attribut **usemap** de l'élément **IMG**.


L'élément HTML **<map>** est utilisé avec des éléments **<area>** afin de définir une image cliquable divisée en régions et peut lui associer un lien hypertexte.

- Le second type d'élément capital pour une image cliquable est l'élément **AREA**, cet élément possède 2 attributs :
 - shape** : qui spécifie la forme d'une région
 - coords** : qui spécifie la position et la forme de la région à l'écran

 Image cliquable : éléments MAP et AREA

- L'attribut **shape** accepte 4 valeurs possibles :
 - default** : spécifie la région entière
 - rect** : définit une région rectangulaire
 - circle** : définit une région circulaire
 - poly** : définit une région polygonale

Le nombre et l'ordre des valeurs de l'attribut **coords** dépendent de la valeur de l'attribut **shape**, comme le montre le tableau suivant. Il s'agit d'une liste de valeurs longueur, séparées par une virgule.

 Image cliquable : éléments MAP et AREA

Valeur de l'attribut shape	Formes	Coordonnées (coords) nécessaires
rect	Rectangle	En haut à gauche (coordonnées x et y), en bas à droite (coordonnées x et y)
circle	Cercle	Centre (coordonnées x et y), rayon
poly	Polygone	Chaque sommet (coordonnées x et y). Le dernier couple de coordonnées devrait être identique au premier pour fermer le polygone

 Création manuelle d'une image cliquable avec l'outil de dessin Microsoft **Paint**, **shape="rect"**

Pour un rectangle :

Placez votre curseur à l'emplacement approximatif du coin supérieur d'un rectangle, à l'emplacement de la région St-Pierre et Miquelon.

Les coordonnées de la position actuelle du curseur (ici encadrées d'un ovale)

Notez ces coordonnées



 `<area shape="rect" coords="171,334,184,358" href="http://www.spm.fr"></area>`

puis placer sur le coin inférieur droit du rectangle pour la même région et notez à nouveau les coordonnées.

Vous disposez des 4 paramètres nécessaires pour la définition d'une forme de type rectangle, que vous n'avez plus qu'à reporter dans la balise AREA comme ci-dessus



`<area coords="131,61,5" shape="circle" href="www.iledefrance.fr"></area>`

Pour le cercle :

Placez dans région Île de France au centre du cercle, et notez ses coordonnées. Déplacez horizontalement ou verticalement pour définir le rayon souhaité et notez la différence entre coordonnées : c'est le rayon, vous disposez des trois paramètres nécessaires pour la définition d'une forme de type cercle

`<area coords="131,61,5" shape="circle" href="www.iledefrance.fr"></area>`

Rayon calculé verticalement
 $= 61 - 56 = 5 \text{ px}$

Voir résultat ci-dessus
 Après un clic sur la zone de la région parisienne, le client se redirige vers la page Web : www.iledefrance.fr

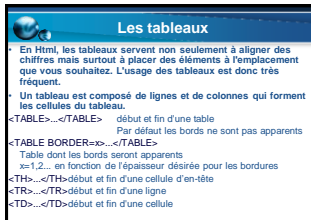
`<area coords="212,51,229,50,223,71,222,90,212,87,212,51" shape="poly" href="www.lalsace.fr"></area>`

Pour le polygone :

Placez sur un des angles de la région Alsace et notez ces coordonnées. Déplacez le curseur successivement sur les autres sommets du polygone souhaité et notez leurs coordonnées

`<area coords="212,51,229,50,223,71,222,90,212,87,212,51" shape="poly" href="www.lalsace.fr"></area>`

Notez les coordonnées suivantes



Les attributs associés

<TABLE BORDER= CELLPADDING= CELSPACING= WIDTH= HEIGHT= BGCOLOR= > </TABLE>

BORDER=

- * BORDER="0" le tableau n'a pas de contour
- * BORDER="n" le tableau a un contour d'épaisseur "n"

CELLPADDING=

- * Définit l'espace entre l'objet et le contour d'une cellule

CELLSPACING=

- * Définit l'épaisseur du trait entre les cellules

WIDTH=

- * Fixe la largeur du tableau

HEIGHT=

- * Fixe la hauteur du tableau

BGCOLOR=

- * BGCOLOR="#RRGGBB" Définit la couleur de fond de tout le tableau. RR, GG et BB sont les valeurs hexadécimales du Rouge(RR), Vert(GG) et Bleu(BB).

Les attributs de <TABLE>

Définit chaque élément de la ligne titre, les cellules d'en-tête qui présente par défaut le texte en gras et avec un alignement centre.

<TH COLSPAN= ROWSPAN= ALIGN= VALIGN= WIDTH= BGCOLOR= ></TH>

- * Décrit chaque élément du tableau

<TD COLSPAN= ROWSPAN= ALIGN= VALIGN= WIDTH= BGCOLOR= ></TD>

COLSPAN=

- * COLSPAN="n" (par défaut n=1) la cellule occupe n colonnes.

ROWSPAN=

- * ROWSPAN="n" (par défaut n=1) la cellule occupe n lignes.

ALIGN=

- * ALIGN= alignement horizontal du contenu de la cellule.
- * ALIGN="LEFT" (valeur par défaut) alignement à gauche de la cellule.
- * ALIGN="RIGHT" alignement à droite de la cellule.
- * ALIGN="CENTER" centrage dans la cellule.

VALIGN=

- * VALIGN= alignement vertical du contenu de la cellule.
- * VALIGN="BOTTOM" (valeur par défaut) alignement au bas de la cellule.
- * VALIGN="TOP" alignement au sommet de la cellule.
- * VALIGN="CENTER" centrage dans la cellule.

WIDTH= largeur de la cellule en pourcentage ou en pixel.

Les attributs de <TABLE>

- * Décrit le titre du tableau
<CAPTION ALIGN= >
</CAPTION>

ALIGN=

- * ALIGN="TOP" (par défaut) la légende du tableau est en haut
- * ALIGN="BOTTOM" la légende du tableau est en bas

Exemple 1

```
<!DOCTYPE html>
<html><head><head>
<body>
<table border="20" cellpadding="0"
  cellspacing="10">
<tr>
<th>Intitulé 1</th>
<th colspan="2">Intitulé 2</th>
</tr>
<tr>
<td>Item 1</td>
<td rowspan="2">Item 2</td>
<td>Item 3</td>
</tr>
<tr>
<td>Item 4</td>
<td>Item 6</td>
</tr>
</table></body> </html>
```

Exemple 2

```

<!DOCTYPE html>
<html><head></head><body>
<center><table border="1">
<caption>Comparatif modèle économique</caption>
<tr><th colspan="2">Modèle</th>
<th colspan="2">Vitesse</th>
<th colspan="2">Consommation</th>
</tr>
<tr>
<th>Marque</th>
<th>Type</th>
<th>Numéro de série</th>
<th>en km/heure</th>
<th>en litre/100 km</th>
</tr>
<tr>
<td>Peugeot 106</td>
<td>106</td>
<td>234.34</td>
<td>132</td>
<td>5.7</td>
</tr>
<tr>
<td>Citroën AX</td>
<td>AX</td>
<td>6789</td>
<td>126</td>
<td>5.5</td>
</tr>
</table>
</body>
</html>

```

Comparatif modèle économique

Modèle		Vitesse		Consommation	
Marque	Type	Numéro de série	en km/heure	en litre/100 km	
Peugeot	106	234.34	132	5.7	
Citroën	AX	6789	126	5.5	

Les Frames HTML

- Grâce à la technologie des frames (en français "cadres") il est désormais possible d'afficher plusieurs pages HTML dans différentes zones (ou cadres).
- Les actions dans un cadre peuvent agir sur un autre cadre
- Les frames ne font pas partie de la spécification du HTML 3.x, il faut donc déclarer sa page comme étant écrite en HTML 4.0 !
- Afin de créer un site contenant des cadres, il suffit de créer un fichier contenant l'agencement des cadres : ce fichier HTML a pour particularité d'avoir un conteneur <FRAMESET> à la place du jeu de balises <BODY>. C'est cette balise qui définit les cadres par leur dimension en pixels ou en pourcentage (%).

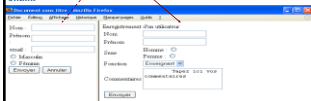
<FRAMESET>	Début de zone avec des fenêtres
</FRAMESET>	Fin de zone avec des fenêtres
<FRAMESET ROWS="...">	Fenêtres horizontales
<FRAMESET COLS="...">	Fenêtres verticales

2 cadres verticaux

```

<html> <head><title> Page de l'application </title> </head>
<FRAMESET COLS="30%,70%">
  <FRAME SRC="form1.htm" NAME="gauche">
  <FRAME SRC="form2.htm" NAME="droite">
</FRAMESET>
</html>

```



2 cadres horizontaux, on remplace COLS par ROWS

```

<html> <head></head>
<FRAMESET ROWS="30%,70%">
  <FRAME SRC="form1.htm"
  NAME="haut">
  <FRAME SRC="form2.htm"
  NAME="bas">
</FRAMESET>
</html>

```



Les attributs de la balise <FRAMESET>

- FRAMESET**
 - COLS**
le frameset est composé de colonnes
donne la largeur de chaque colonne
 - ROWS**
le frameset est composé de lignes
donne la hauteur de chaque ligne
 - Frameborder="yes|no"** Indique si le cadre a une bordure ou non
 - Border="n"** Indique la taille de la bordure
 - Bordercolor** Indique la couleur de la bordure
 - Framespacing="n"** Indique l'espace entre les cadres

La balise <FRAME> permet de définir un ou plusieurs cadres au sein de la balise <FRAMESET>

Les attributs de la balise <FRAME>

- FRAME**
 - Frameborder="yes|no"** Indique si le cadre a une bordure ou non
 - Border="n"** Indique la taille de la bordure
 - SCROLLING="yes|no|auto"**
Permet d'afficher ou non la barre de défilement, "auto" laisse le navigateur décider de son utilité
 - NORESIZE**
Empêche l'utilisateur de redimensionner les cadres
 - SRC**
document à afficher dans le cadre
 - NAME**
nomme un cadre
 - TARGET**
 - spécifie la fenêtre de destination avec un lien hypertexte
 - _BLANK**
Affiche la cible dans une nouvelle fenêtre
 - _PARENT**
Affiche la cible dans le cadre de niveau hiérarchiquement supérieur
 - _TOP**
Affiche la cible dans la fenêtre entière du navigateur
 - _SELF**
Affiche la cible dans le même cadre que le lien

Les cibles dans les Frames HTML

indexframe.htm

```
<FRAMESET COLS="100,100">
<FRAME SRC="som1.htm" NAME="fr1">
<FRAME SRC="intro.htm" NAME="fr2">
</FRAMESET>
```

som1.htm

```
<HTML><HEAD><BASE TARGET="fr2"></HEAD>
<BODY>
<A HREF="intro.htm">
  Accueil</A><BR>
<A HREF="produit.htm" TARGET="fr2">
  Produits</A><BR>
<A HREF="sommaire.htm" TARGET="_top">
  Support Technique</A><BR>
<A HREF="http://www.w3c.org/TR" TARGET="_self">
  Documentation W3C</A><BR>
<A HREF="som2.htm" TARGET="fr1">
  Suite Sommaire</A><BR>
</BODY></HTML>
```



Navigateurs non compatibles frame

On utilisera les balises <NOFRAME> et </NOFRAME> permettant de spécifier un texte HTML à afficher en cas de navigateur ne permettant pas d'afficher les frames. Le texte entre les balises <NOFRAME> et </NOFRAME> doit donc contenir les balises <BODY> ... </BODY>.

Exemple :

```
<html> <head></head>
<FRAMESET COLS="20%,80%">
<FRAME SRC="frame1.htm" NAME="gauche">
<FRAME SRC="frame2.htm" NAME="droite">
</FRAMESET>
```

```
<NOFRAME>
<BODY> Cette page HTML nécessite un navigateur supportant les
frames, veuillez nous en excuser.
</BODY>
</NOFRAME> </html>
```

Les Formulaires HTML

- Les formulaires interactifs permettent un dialogue avec les utilisateurs d'une manière interactive.
- Le lecteur saisit des informations en remplissant des champs ou en cliquant sur des cases à cocher, puis appuie sur un bouton de soumission (**submit**) pour l'envoyer soit à un **URL**, c'est-à-dire de façon générale à une adresse e-mail ou à un script de page web dynamique tel que PHP, ASP, JSP.

Les Formulaires HTML

La balise **FORM** constitue en quelque sorte un conteneur permettant de regrouper des éléments qui vont permettre à l'utilisateur de choisir ou de saisir des données, ensemble de données qui seront envoyées à l'URL indiqué dans l'attribut **ACTION** de la balise **FORM** par la méthode indiquée par l'attribut **METHOD**

Il est possible d'insérer n'importe quel élément HTML de base dans une balise **FORM** (textes, boutons, tableaux, liens,...) mais il est surtout intéressant d'insérer des éléments interactifs. Ces éléments interactifs sont :

- La balise **INPUT**: un ensemble de boutons et de champs de saisie
- La balise **TEXTAREA**: une zone de saisie
- La balise **SELECT**: une liste à choix multiples

Les Formulaires

`<form method="POST" action="URL d'expédition" >`
... les formulaires proprement dit ...

`</form>`

- action** = l'URL définit l'emplacement où doivent être envoyées les données collectées par le formulaire
- method** = la méthode via laquelle les données sont envoyées grâce à l'attribut **METHOD**:
 - method="GET"**: les données sont envoyées par l'intermédiaire de l'URL et sont ajoutées à la fin de celui-ci, l'ensemble de ces paires champ/valeur étant séparées entre elles par des caractères **&**
`http://serveur/chemin/prog.html?champ1=val1&champ2=val2`
 - method="POST"**: génère une requête HTTP spéciale qui envoie les données au serveur, les données sont envoyées comme corps du message

Les éléments des Formulaires HTML

Ligne de texte :

```
<form>
  <input type="text"
    name="nom" size="50">
</form>
```


Zone de saisie :

```
<form>
  <textarea name="nom"
    rows="4" cols="40">
    valeur par défaut
  </textarea>
</form>
```

Valeur par défaut

Les éléments des Formulaires HTML

Liste déroulante :



```

<form>
  <select name="nom" size="1">
    <option>lundi</option>
    <option>mardi</option>
    <option>mercredi</option>
    <option>jeudi</option>
    <option>vendredi</option>
  </select>
</form>

```

<option selected> mercredi : On peut présélectionner l'élément affiché dans la boîte d'entrée (par défaut, le premier élément de la liste sera retenu)

<select disabled> : désactive la liste, elle apparaît alors en grisé et on peut afficher le menu déroulant.

<select multiple="multiple" size="5"> : l'utilisateur pourra effectuer plusieurs choix, mais il doit maintenir la touche **[Ctrl]** du clavier enfoncée, et cliquer sur les différents éléments avec la souris.

Structuration de la liste de sélection

Les balises `<optgroup>` et l'attribut `label` permettent de regrouper plusieurs options de type sous un titre (label). La balise `<optgroup>` crée un décalage des options ainsi regroupées. Le titre du groupe n'est pas sélectionnable.



```

<form>
  <select>
    <optgroup label="Europe">
      <option>France</option>
      <option>Belgique</option>
      <option>Suisse</option>
    </optgroup>
    <optgroup label="Autres pays">
      <option>Etats-Unis</option>
      <option>Canada</option>
      <option>Inde</option>
    </optgroup>
  </select>
</form>

```

Les éléments des Formulaires HTML

Case à cocher à choix unique :

☐ tarif de jour ☐ tarif de nuit ☐ tarif de week-end

```

<form>
  <input type="radio" name="tarif" value="jour"> tarif de jour
  <input type="radio" name="tarif" value="nuit"> tarif de nuit
  <input type="radio" name="tarif" value="week-end"> tarif de week-end
</form>

```

L'attribut **checked** (optionnel) permet de présélectionner un bouton radio :

```

<input type="radio" name="tarif" value="jour" checked="checked">
tarif de jour

```

Les éléments des Formulaires HTML

Case à cocher à choix multiple (label permet de cocher la case en cliquant sur le libellé de chaque case identifiée par son ID) :

```

<form>
  <input type="checkbox" name="choix1" id="choix1" value="1"
  checked="checked" >
  <label for="choix1"> glace vanille </label>
  <input type="checkbox" name="choix2" id="choix2" value="2">
  <label for="choix2"> chantilly </label>
  <input type="checkbox" name="choix3" id="choix3" value="3">
  <label for="choix3"> chocolat chaud </label>
  <input type="checkbox" name="choix4" id="choix4" value="4">
  <label for="choix4"> biscuit </label>
  <input checked="" type="checkbox" value="1" />
  <input type="checkbox" value="2" />
  <input type="checkbox" value="3" />
  <input type="checkbox" value="4" />
  </form>

```

☒ glace vanille ☐ chantilly ☐ chocolat chaud ☐ biscuit

<fieldset>

L'élément HTML **<fieldset>** est utilisé afin de regrouper plusieurs contrôles interactifs ainsi que des étiquettes (**<label>**) dans un formulaire HTML.

```

form action="3">
<fieldset style="width:60%" >
<legend> Informations personnelles </legend>
<div>
<label for="name">Nom : </label>
<input type="text" id="name" value="Chris">
</div>
<br />
<div>
<label for="pwd">Mot de passe : </label>
<input type="password" id="pwd" value="Wookie" required>
</div>
</fieldset>
</form>

```

Informations personnelles

Nom :

Mot de passe :

Les éléments des Formulaires HTML

Bouton de commande :

- Avec l'introduction des langages de scripts (Javascript et VBscript) l'usage du bouton de commande présente un contrôle avant l'envoi du formulaire.
- Un petit exemple:

```

<form>
<input type="button" name="bouton1" value="bouton de test"
onclick="alert('test réussi !');">
</form>

```

Submit et Reset :

```

<form>
<input type="submit" name="bouton2" value="Envoyer">
<input type="reset" name="bouton3" value="Annuler">
</form>

```

Le bouton Submit envoie toutes les informations contenues dans le formulaire à l'URL désignée dans les attributs ACTION et METHOD de l'élément <FORM>

Le bouton Reset permet d'annuler les modifications apportées aux contrôles d'un formulaire et de restaurer les valeurs par défaut.

Exemple 1 en HTML4

```

<HTML><HEAD><META><BODY> <P>
<H4 align="center">ENQUETE sur la SATISFACTION des UTILISATEURS</H4>
<FORM ACTION="exemple4.html" METHOD="GET">
<PRE>
<INPUT TYPE="RADIO" NAME="SECTEUR1" VALUE="UNIV" CHECKED> Université
<INPUT TYPE="RADIO" NAME="SECTEUR1" VALUE="ONGS" > C.N.A.S.
<INPUT TYPE="RADIO" NAME="SECTEUR1" VALUE="NGS" > I.N.S.E.R.M.
<INPUT TYPE="RADIO" NAME="SECTEUR1" VALUE="PUB" > Autre PUBLIC
<INPUT TYPE="RADIO" NAME="SECTEUR1" VALUE="PRIV" > Privé
</PRE>
<P>
Vos Nom et prénom : <INPUT TYPE="TEXT" NAME="NOM1" SIZE="30">
<P>
Vos adresse électronique : <INPUT TYPE="TEXT" NAME="ADR1" SIZE="30">
<P>
Je suis intéressé<INPUT TYPE="CHECKBOX" NAME="BEAU" VALUE="beau">,
jeune<INPUT TYPE="CHECKBOX" NAME="JEUNE" VALUE="jeune">,
riche<INPUT TYPE="CHECKBOX" NAME="RICHE" VALUE="riche">,
en bonne santé<INPUT TYPE="CHECKBOX" NAME="SANTÉ" VALUE="en bonne santé">,
sans opinion <SELECT NAME="OPINION">
<OPTION> Très satisfait </OPTION>
<OPTION SELECTED> Satisfait </OPTION>
<OPTION> Indifférent </OPTION>
<OPTION> C'est nul ! </OPTION>
<SELECT>
<P>
Vos commentaires <TEXTAREA NAME="COM1" ROWS="3" COLS="40"></TEXTAREA>
<P>
<INPUT TYPE="SUBMIT" VALUE="Envoyer"> <INPUT TYPE="RESET" VALUE="Annuler">
<PRE>
</FORM>
</BODY></HTML>

```

Résultat 1 en HTML4

ENQUETE sur la SATISFACTION des UTILISATEURS

☒ UNIVERSITE
☐ C.N.A.S.
☐ I.N.S.E.R.M.
☐ Autre PUBLIC
☐ Privé

Vos Nom et prénom :

Votre adresse électronique :

Je suis intéressé ☐ ,
 jeune ☐ ,
 riche ☐ ,
 en bonne santé ☐

Votre opinion :

Vos commentaires :



Les feuilles de style CSS

Concept :

- Dans un document d'une certaine importance, il arrive fréquemment que l'on attribue à certains éléments des caractéristiques de **mise en forme identiques**. Par exemple, les noms de chapitres seront mis en police Arial, en gras et en couleur bleu.
- On peut imaginer que l'on puisse donner à cette définition de mise en forme un nom soit **"titre"** et qu'à chaque nouveau chapitre, plutôt que d'écrire chaque fois le nom du titre et puis de devoir le mettre en Arial, gras, bleu, l'on puisse dire à l'ordinateur, nom du chapitre mais dans la mise en forme que j'ai défini sous le nom de **"titre"**. Cette définition de mise en forme particulière, on va l'appeler feuille de style.



Cascading Style Sheets (CSS)

- On parle aussi de feuilles de style en cascade [Cascading Style Sheets ou CSS] car en cas de styles identiques, un ordre de priorité sera déterminé par le browser .
- Précisons pour terminer que les feuilles de style ne sont pas une composante directe du langage Html mais un développement à part dans la publication de pages Web.
- Exemple de HTML sans feuille de style :

```
<H1><B><FONT COLOR="blue"> Titre 1 </FONT></B></H1> : Titre 1
<H2><B><FONT COLOR="green"> Titre 1.1 </FONT></B></H2> : Titre 1.1
<H3><B><FONT COLOR="red"> Titre 1.1.1 </FONT></B></H3> : Titre 1.1.1
```



Cascading Style Sheets (CSS)

Utilité et avantages

- Séparation du contenu et de la mise en forme.
- Cohésion de la présentation tout au long du site avec les feuilles de style externes.
- Modifier l'aspect d'une page ou d'un site sans en modifier le contenu HTML et cela en quelques lignes plutôt que de devoir changer un grand nombre de balises.
- Une façon d'écriture concise et nette par rapport au Html qui devient vite fouillis.
- Réduire le temps de chargement des pages.
- Palier certaines insuffisances du langage Html (contrôle des polices, contrôle de la distance entre les lignes, contrôle des marges et des indentations (sans devoir utiliser de tableaux ou de balise <DD>...) et ainsi augmenter la créativité des développeurs Web.
- Permettre le positionnement au pixel près du texte et/ou des images.



Cascading Style Sheets (CSS)

Définition d'un style :

balise { propriété de style: valeur; propriété de style: valeur; }

Exemple :

H3 { font-family: Arial; font-style: italic; }

Donc ici, la balise H3 sera en Arial et en italique.

Et dans votre document, toutes les balises <H3> auront comme style *Arial et italique*.

- Les feuilles de style portent sur des balises principalement et quelques autres éléments comme par exemple **A:link** pour un lien non-visité et **A:visited** pour un lien visité. Comme balises souvent utilisées avec des feuilles de style, on peut citer les en-têtes **Hn, P, BODY...**
- Les propriétés de style sont entourées par des " { " et pas des " [" ou des parenthèses.
- Le couple "propriété de style/valeur" est séparé par un double-point (:).
- Chaque couple "propriété de style/valeur" est séparé par un point-virgule (;).
- Il n'y a pas de limite pour le nombre de couples "propriétés de style/valeur".

Cascading Style Sheets (CSS)

- Pour la lisibilité toujours, vous pouvez écrire vos styles sur plusieurs lignes → Cascade :
H3 { font-family: Arial;
font-style: italic;
font-color: green; }
- On peut attribuer plusieurs valeurs à une même propriété. Dans ce cas, on séparera les différentes valeurs par des virgules.
H3 { font: Arial, italic, green; }
- On peut attribuer un même style à plusieurs balises (séparées par des virgules).
H1, H2, H3 { font-family: Arial; font-style: italic; }

Cascading Style Sheets (CSS)

3 façons pour incorporer les styles dans le document Html :

1- A l'intérieur des balises <HEAD> </HEAD> : Permet de séparer les éléments de mise en forme du contenu.

```
<HTML>
<HEAD>
  <STYLE type="text/css">
    <!-- La ou les feuilles de style -->
  </STYLE>
</HEAD>
<BODY> .....</BODY>
```

- La balise <STYLE> avertit le navigateur que l'on va utiliser des feuilles de style.
- L'attribut type="text/css" informe que ce qui suit est du texte et qu'il s'agit de cascading style sheets (css).
- La balise Html de commentaires <!-- --> empêche que les browsers qui ne connaissent pas les feuilles de style, tentent d'interpréter ces instructions. Les informations à l'intérieur des tags de commentaires seront ignorées par ces browsers.
- Pour vos propres commentaires à propos des feuilles de style, on utilisera la convention désormais classique (C, C++, JavaScript...) de /* commentaires */.

Cascading Style Sheets (CSS)

2- A l'intérieur des balises <BODY> </BODY>

- Cette façon de faire nous paraît illogique et peu conforme à l'esprit des feuilles de style qui est de définir un style déterminé valable pour la globalité du document. Mais elle existe pour quelques utilisations spécifiques...

```
<HTML>
<BODY>
  <H1 style="font-family: Arial; font-style: italic;"> blabla </H1>
  <H1> blabla </H1>
</BODY>
</HTML>
```

Signalons :

- que le style Arial, italique n'affectera que cette seule balise H1.
- que la syntaxe est légèrement différente de la précédente.
- que l'écriture :
<STYLE type="text/css"> H1{font-family:Arial; font-style: italic; }</STYLE>
 fonctionne aussi.

Cascading Style Sheets (CSS)

3- A l'extérieur de la page HTML : Styles externes

- On crée d'abord, dans le répertoire du site, un fichier avec l'extension .css, soit **styles.css** qui contiendra toutes les feuilles de style.

- Ensuite, on crée une page normale soit **page1.htm** (bien entendu dans le même répertoire que le fichier styles.css, sinon il faudra indiquer le chemin où se trouve la feuille de style).

```
<HTML>
<HEAD>
  <LINK rel="stylesheet" type="text/css" href="CSS/styles.css">
</HEAD>
```

- La balise <LINK> avertit le browser qu'il faudra réaliser un lien.
- L'attribut rel=stylesheet précise qu'il y trouvera une feuille de style externe.
- L'attribut type="text/css" précise que l'information est du texte et du genre cascading style sheets.
- L'attribut classique de lien href=" " donne le chemin d'accès et le nom du fichier à lier.

CSS : Les classes et les ID

- Notion de classes
- Mais on désire parfois affecter des styles différents à une même balise. Pas de problèmes, les feuilles de style vous proposent la solution des classes (**class**).
- La définition d'un style était :
`balise { propriété de style: valeur; }`
- Elle devient :
`balise.nom_de_classe { propriété de style: valeur; }`
- Remarquez le point entre balise et nom_de_classe
- Ou, comme la mention de la balise est facultative :
`.nom_de_classe { propriété de style: valeur; }`
- Attention! L'emploi du point (.) devant le nom de classe est indispensable.
- Pour appeler l'effet de style dans le document, on ajoute le nom de la classe à la balise.
`<balise class="nom_de_classe"> ... </balise>`

CSS : Les classes et les ID

- Un exemple :
- Je souhaite mettre ce qui est important dans le texte en **gras** et en **bleu**. Je crée la classe « .essentiel » :
`.essentiel { font-weight: bold; font-color: #000080; }`
- Et dans le document Html, il suffit d'appeler la classe essentiel quand cela se révèle nécessaire :

```
<P class="essentiel"> ... blabla ... </P>
<H1 class="essentiel"> Titre 1 </H1>
<TABLE> <TR> <TD class="essentiel"> cellule </TD> <TD>...
```

CSS : Les classes et les ID

- Les ID fonctionnent exactement comme les classes.
- La syntaxe est :
`#nom_de_ID { propriété de style: valeur; }`
- Et pour l'appeler :
`<balise id="nom_de_ID"> </balise>`
- Notons qu'on ne pourra effectuer qu'un seul appel à #nom_de_ID par page. Ainsi,
`Pour #essentiel{ ... }`
`<P id="essentiel"> est correct.`
Mais si on rencontre dans la même page
`<H1 id="essentiel"> ce n'est plus correct !`

Les Sélecteurs (i)

- **Sélecteur de classe**
`h1.nouveau { color: #FF0000; }`
`<h1 class="nouveau">Tout Nouveau !</h1>`
`.nouveau { color: #FF0000; }` // Sélection de tous les éléments de même classe
- **Sélecteur ID**
`#intro { letter-spacing: 0.3em; }`
`<h1 ID= "intro">Introduction</h1>`
`h1#intro { letter-spacing: 0.3em; }`

Les Sélecteurs (ii)

- **Sélecteur contextuel**
 - On peut attribuer un même style à plusieurs balises (séparées par des virgules).

```
H1 EM { color: red; }
    • Affecte uniquement les éléments EM dans un H1
UL LI { font-size: small; }
UL UL LI { font-size: x-small; }
```

- **Mixage de sélecteurs**

```
.element H1 { color: red; }
#ident H1 { background: blue; }
DIV.sidenote H1 { font-size: large; }
H1 B, H2 B, H1 EM, H2 EM { color: red; }
```

CSS : et <DIV>

- **Utilité**

Pour "déconnecter" certains morceaux de paragraphe ou plusieurs paragraphes d'écriture avec des feuilles de style. Ce sont respectivement les balises SPAN et DIV qui créaient ainsi des petits blocs particuliers dans le document sans devoir repasser par les éléments structurels du Html classique.
- **DIV** définit un bloc (block level): boîte rectangulaire, retour à la ligne à la fin.
- **Span** définit un morceau de texte à l'intérieur d'un bloc (inline): boîte qui colle au texte.

CSS : SPAN

La balise ... permet d'appliquer des styles à des éléments de texte d'un paragraphe ou si vous préférez à un morceau d'un élément HTML.

Exemple :

```
<html>
<head>
  <style type="text/css">
    .element {font-size: x-large; color: blue;}
  </style>
</head>
<body>
  <p>un monde de <span class="element"> géants </span> </p>
</body>
</html>
```

Résultat :

Un monde de **géants**

CSS : DIV

La balise <DIV> ... </DIV> est un conteneur permettant de regrouper une suite de balise HTML ou plusieurs paragraphes ou si vous préférez, de délimiter une zone comportant plusieurs éléments devant bénéficier du même style.

```
<html>
<head>
  <style type="text/css">
    h1.cs { color : blue; text-align : center; }
    .zone { font-size: x-small; }
  </style>
  <link rel="stylesheet" type="text/css" href="style.css">
  /* le fichier style.css contient : p.cs { color : green; border : solid red; } */
</head>
<body>
  <div class="cs">import de h1 la balise &lt;div>&gt;
    <div class="zone">
      <p class="cs">commentaire :</p>
      <p>n'oubliez pas l'attribut class!</p>
    </div>
  </div>
</body>
</html>
```

IMPORT DE H1 La balise <DIV>

commentaire :

n'oubliez pas l'attribut class!



CSS : Les positions des éléments

- La position absolue (`position: absolute`) se détermine par rapport au coin supérieur gauche de la fenêtre du browser. Les coordonnées de ce point sont `top = 0` et `left = 0`. Les coordonnées d'un point s'expriment en pixels, de haut en bas pour `top` et de gauche à droite pour `left`.
- La position relative (`position: relative`) se détermine par rapport au largeur et hauteur du conteneur parent (`width` et `height`), cela revient à mesurer les coordonnées à partir de l'intérieur des marges internes du conteneur parent, ce dernier pourra être un autre élément du code Html comme un paragraphe `<P>`.



CSS : position du texte

- Plaçons en position absolue le texte "Page Html" à 100 pixels du haut de la fenêtre (`top`) et à 25 pixels de la gauche (`left`).

```
<HTML>
<HEAD>
  <STYLE type="text/css">
    .pub {position: absolute; top: 100px; left: 25px;
    color: red; font-size: x-large; font-weight: bold;}
  </STYLE>
</HEAD>
<BODY>
  <DIV class= "pub" > Page Html </DIV>
</BODY>
</HTML>
```



CSS : position d'une image

- Plaçons l'image « image.bmp » en position absolue à 100 pixels de haut de la fenêtre vers le bas (`top`) et à 25 pixels vers la gauche (`left`). Les dimensions de l'image sont fixées à `width=242` pixels et `height=84` pixels:

```
<HTML>
<BODY>
  <span style="position: absolute;
top: 100px; left: 25px; width: 242px; height: 84px;">
    <IMG src="image.bmp">
  </span>
</BODY>
</HTML>
```



CSS: Superposer du texte sur une image

```
<HTML> <BODY>
  <span style="position: absolute;
top: 100px; left: 25px; width: 242px; height: 84px;">
    <IMG src="image.bmp">
  <span style="position: absolute;
top: 110px; left: 30px; color: red; font-size: x-small;
font-weight: bold;">
    Je m'appelle Toma
  </span>
</BODY> </HTML>
```



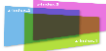


La propriété z-index

La propriété z-index permet de préciser l'empilement de certains éléments d'une page, c'est-à-dire sur l'axe vertical. Elle permet par exemple d'indiquer que pour deux éléments A et B partiellement ou totalement superposés, A sera placé au dessus de B ou inversement.

1. Seuls les éléments positionnés peuvent avoir un z-index. Un élément positionné est un élément dont la propriété CSS position a pour valeur relative, absolue ou fixed. Par défaut, les éléments d'une page ne sont pas positionnés (ils sont en position:static).

2. Les valeurs les plus élevées sont au premier plan, et les plus faibles sont au second plan. Un z-index de 2 sera placé au dessus d'un z-index de 1, et un z-index de -1 sera placé au dessus d'un z-index de -2.



Exemple z-index

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<style>
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
}
</style>
</head>
<body>
<h1>Ceci est un Titre</h1>

<p>Comme l'image a un z-index de -1, elle sera placée derrière le texte.</p>
</body>
</html>
```



Ceci est un Titre

Comme l'image a un z-index de -1, elle sera placée derrière le texte



Exemple z-index

```
<html><head>
<style>
div (height: 100px; width: 100px)
div#test1 (position: absolute; top:30; left:30; z-index: 4; border:solid red;
background:red;)
div#test2 (position: absolute; top:50; left:50; z-index: 2; border:solid green;
background:green;)
div#test3 (z-index: 10; border:dotted black; background:yellow;)
div#test4 (position: relative; z-index: 8; border:dashed blue; background:blue;)
</style>
</head><body>
<div id="test1">.....1...</div>
<div id="test2">.....2...</div>
<div id="test3">.....3...</div>
<div id="test4">.....4...</div> </body></html>
```



Exemple z-index

- Le résultat attendu est le suivant:
- div#test1** sera au premier plan;
- en dessous on aura **div#test2**;
- en dessous encore on aura **div#test3** (qui ne se place pas au-dessus malgré un z-index de 10 car il n'est pas positionné);
- enfin, **div#test4** sera le plus haut de la pile, mais ne recouvre pas les autres blocs ici car il est repoussé vers le bas par **div#test3** qui n'est pas positionné.
- Au final, si tous les blocs ont une couleur de fond opaque, on ne pourra apercevoir que **div#test1** en haut et **div#test4** plus bas. Les autres blocs sont recouverts par **div#test1**.

La propriété z-index animée

```

<!DOCTYPE html>
<html><head><style>
  div {position: absolute;}
</style></head>
<body>
  <div id="container">
    <div style="background-color: lightblue;
      border: 1px solid #333333;
      width: 100px;
      height: 100px;
      opacity: 50%;
    ">
    <div id="myBox">
      <div style="background-color: coral;
        opacity: 100%;
        z-index: 1;
        -webkit-animation: mymove 10s infinite linear; /* Chrome, Safari, Opera */
        animation: mymove 10s infinite linear;
      ">
    </div>
  </div>
</body>
</html>

```

La propriété z-index animée

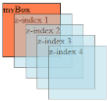
`<body style="position: absolute;">`
`<p>Note: les animations CSS ne fonctionnent pas dans Internet Explorer 9 et les versions antérieures. </p>`

`<p> Modifiez progressivement la propriété z-index de "myBox" de 1 à 5 et revenez à 1: </p>`

```

<div id="container">
  <div id="myBox">myBox</div>
  <div style="top:20px;left:20px;z-index:1;">z-index 1</div>
  <div style="top:40px;left:40px;z-index:2;">z-index 2</div>
  <div style="top:60px;left:60px;z-index:3;">z-index 3</div>
  <div style="top:80px;left:80px;z-index:4;">z-index 4</div>
</div>
</body>
</html>

```



La règle @keyframes

- La règle **@keyframes** permet d'indiquer quelles propriétés doivent être animées et comment elles doivent l'être.
- La règle **@keyframes** permet de préciser différentes valeurs auxquelles les propriétés animées doivent parvenir à certains stades de l'animation.
- La règle **@keyframes** permet de définir différents stades ou étapes pour l'animation et c'est ce qui permet d'avoir un contrôle total sur la progression de l'animation.
- Il faut donner un nom à une règle **@keyframes** afin de la réutiliser dans l'animation et lui indiquer les propriétés à animer et comment les animer.

La propriété animation-name

La propriété **animation-name** permet de définir une liste d'animations qui doivent s'exécuter. Elle prend en charge un ou plusieurs noms qui devront correspondre aux noms des règles **@keyframes** qui définissent les propriétés à animer et les différentes valeurs qu'elle doit avoir pendant l'animation.

Si on fournit plusieurs noms à la propriété **animation-name**, alors il faudra définir à minima la durée de l'animation pour chaque animation avec la propriété **animation-duration** si on veut des animations fonctionnelles. Dans le cas où l'on fournit moins de valeurs à **animation-duration** qu'à **animation-name**, alors les animations supplémentaires vont réutiliser les valeurs de **animation-duration** dans l'ordre.

Exemple: `animation: mymove 10s infinite linear;`
`animation-duration: 10s;`
`animation-timing-function: linear;`
`animation-delay: 0s;`
`animation-iteration-count: infinite;`
`animation-direction: normal;`
`animation-fill-mode: none;`
`animation-play-state: running;`
`animation-name: mymove;`



La propriété animation-timing-function

La propriété animation-timing-function permet la façon dont doit progresser l'animation entre les différentes valeurs de keyframes : la progression de l'animation peut être linéaire, s'accélérer de plus en plus, etc.

En fonction des valeurs suivantes à animation-timing-function :

ease : valeur par défaut. Entre deux valeurs de keyframes, l'animation va commencer relativement lentement, puis accélérer au milieu et se terminer lentement ;

linear : Entre deux valeurs de keyframes, l'animation aura une vitesse constante du début à la fin ;

ease-in : Entre deux valeurs de keyframes, l'animation va commencer lentement puis accélérer jusqu'à atteindre la prochaine valeur de keyframe ;

ease-out : Entre deux valeurs de keyframes, l'animation va commencer rapidement et décélérer progressivement jusqu'à atteindre la prochaine valeur de keyframe ;

ease-in-out : Entre deux valeurs de keyframes, l'animation commence lentement, accélère au milieu et finit lentement ;

cubic-bezier(x1, y1, x2, y2) : permet de définir une courbe de Bézier spécifique pour créer une animation à la vitesse totalement contrôlée.



La propriété animation-direction

La propriété animation-direction permet de spécifier le sens dans lequel une animation doit être jouée, c'est-à-dire si elle doit être jouée en partant du début ou de la fin pour une ou plusieurs de ses itérations ou répétitions.

Les valeurs de animation-direction :

normal : valeur par défaut. L'animation est jouée dans le sens dans lequel elle a été déclarée (du from vers le to) ;

reverse : l'animation est jouée dans le sens inverse pour toutes ses itérations ;

alternate : l'animation va être jouée une première fois dans le sens normal, puis dans le sens contraire, puis à nouveau dans le sens normal et etc. ;

alternate-reverse : l'animation va être jouée une première fois dans le sens inverse, puis dans le sens normal, puis à nouveau dans le sens inverse et etc.



La propriété animation-play-state

La propriété animation-play-state permet de définir si une animation doit être jouée ou être en pause. Elle prend soit la valeur running (l'animation s'exécute normalement), soit paused (l'animation est mise en pause).

Cette propriété va pouvoir être utile pour mettre une animation en pause à un certain point de l'animation ou selon une certaine action de l'utilisateur.

Par exemple, on peut proposer aux utilisateurs de mettre en pause une animation lorsqu'ils passent le curseur de leur souris sur l'élément pour lequel une animation est jouée ou lorsqu'ils cliquent (en gardant le clic enfoncé) sur l'élément en utilisant les pseudo classes :hover et :active.



La propriété animation

La propriété animation-iteration-count permet de définir combien de fois une animation va être jouée. Par défaut, une animation ne sera jouée qu'une fois.

Pour modifier ce comportement par défaut, on va pouvoir passer soit un nombre à animation-iteration-count qui va correspondre au nombre de fois que l'on souhaite jouer l'animation, soit le mot clef infinite qui signifie que l'animation va se répéter à l'infini.

La propriété animation-duration permet de définir le temps que doit durer une animation. On doit préciser une durée en secondes.

La propriété animation-delay permet de définir quand une animation doit commencer c'est-à-dire s'il doit y avoir un délai avant le lancement de l'animation.

Elle prend une valeur en secondes à animation-delay qui correspond au délai qu'il doit s'écouler avant le lancement de l'animation.

HTML 5

- Tout comme [HTML](#) ou [XHTML](#), les documents HTML5 nécessitent un [Doctype](#) indiquant la méthode de rendu standard au navigateur. Toutefois, pour les documents [XML](#) cette déclaration est facultative, le navigateur l'interprétant en mode standard par défaut.
- Il est à noter que la déclaration du Doctype n'est pas sensible à la casse et qu'il n'y a plus de référence à une [DTD](#).
- Exemple d'utilisation avec HTML5

```
<!DOCTYPE html>
```

HTML 5

- Le processus de détection de l'encodage a également été modifié :
- Utilisation d'une balise meta spécifique

<meta charset="UTF-8"> par exemple, la syntaxe utilisée dans les versions précédentes étant toujours compatible

- Pour les documents XHTML5, l'auteur doit spécifier l'encodage dans le prologue XML :

<?xml version="1.0" encoding="UTF-8"?>

Exemple HTML 5

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="utf-8" />
  <title>titre du site</title>

  <!-- meta -->
  <meta name="description" content="" />
  <meta name="keywords" content="" />
  <meta name="author" content="" />

  <!-- mon icon -->
  <link rel="shortcut icon" href="favicon.ico" />
  <!-- ma template.css -->
  <link href="css/template.css" type="text/css" rel="stylesheet"
media="screen" charset="utf-8" />
</head>
```

Exemple HTML 5 suite

```
<body>
  <header class="ma-class-en-css">
    <!-- menu du haut -->
    <nav class="ma-class-en-css">
      <ul>
        <li><a href="mon url" title="titre du lien">nom de mon lien</a></li>
        <li><a href="mon url" title="titre du lien">nom de mon lien</a></li>
        <li><a href="mon url" title="titre du lien">nom de mon lien</a></li>
      </ul>
    </nav>
    <h1>contenu de ma page</h1>
    <div id="content">
      <!-- article n°1 -->
      <article class="ma-class-en-css">
        <h2>titre de mon article</h2>
        <p>texte de mon article</p>
        <section class="ma-class-en-css">
          <h3>sous-titre mon article</h3>
          <p>texte de mon sous-titre</p>
          <figure class="ma-class-en-css">
            <a href="mon url">img src="url de mon image" alt="nom de mon image"</a>
            <figcaption>
              <h4>caption</h4>
            </figcaption>
          </figure>
        </section>
      </article>
```

Exemple HTML 5 suite

```

<!-- article n°2 -->
<article class="ma-class-en-cas">
  <h1>titre de mon article</h1>
  <!-- liste de mon article -->
  <section class="ma-class-en-cas">
    <h2>sous-titre mon article</h2>
    <!-- liste de mon sous-titre -->
    <figure class="ma-class-en-cas">
      <img href="mon url">img src="url de mon image" alt="nom de mon image"></img>
    </figure>
  </section>
</article>
<!-- menu du bas -->
<div class="ma-class-en-cas">
  <nav class="ma-class-en-cas">
    <a href="mon url">titre du lien</a>nom de mon lien</a>
    <a href="mon url">titre du lien</a>nom de mon lien</a>
    <a href="mon url">titre du lien</a>nom de mon lien</a>
  </nav>
</div>
</body>
</html>

```

Nouvelles balises HTML5

- section (Identifier un bloc de contenu)
- aside (Insérer un contenu sans rapport avec la page)
- header (Indiquer l'en-tête de la section concernée)
- footer (Indiquer le pied de page de la section concernée)
- nav (Indiquer une section avec beaucoup de liens internes au site)
- audio (Insérer un contenu audio)
- video (Insérer un contenu vidéo)
- progress (Indiquer un niveau d'avancement)

Nouvelles attributs HTML 5

Il y a également de nouveaux types pour la balise input :

- datetime
- datetime-local
- date
- month
- week
- time
- number
- range
- email
- url
- search
- color

Les balises frame, frameset et noframes ont été supprimées elles aussi, elles étaient déjà dépréciées car elles créaient des problèmes d'accessibilité et d'utilisation pour l'utilisateur final.

Pour la balise button :

- autofocus
- form
- formaction
- formenctype
- formmethod
- formnovalidate
- formtarget

Pour la balise form :

- novalidate

Pour la balise li :

- value (retiré)

Pour la balise menu :

- label
- type


Pour la balise meta :

- charset

Pour la balise ol :

- reversed

HTML 5

HTML 5

- En conclusion...
- HTML5 propose de nouveaux éléments très pratiques qui ont pour objectif d'harmoniser les médias et de structurer la mise en page par des éléments plus "sémantiques". Il permettra également de faciliter sensiblement l'accessibilité au contenu et l'interopérabilité étant donné que les formats propriétaires tels que Flash ou SilverLight pourraient être concurrencés par des éléments tels que <audio> ou <video>. De plus, l'analyse des pages par des robots ou par des synthétiseurs vocaux sera facilitée par les éléments <header>, <nav>...etc.