

# Technologies Web

A.U: 2024/2025

Filières MGSI & IL: S2



# *PLAN*

- *Introduction*
- *Langage HTML5*
- *Langage CSS3*
- *Langage javascript*
- *Langage PHP5*
- *Conclusion*

# Objectifs du cours

- *Découvrir les balises HTML et leur utilité*
- *Apprendre à créer une page HTML simple et ses principaux éléments*
- *Editer une feuille de styles css*
- *Editer un site web avec JavaScript*
- *Apprendre les étapes de mise-en-ligne des interfaces web statiques*
- **S'initier à la programmation web dynamique à l'aide de PHP**

# *Technologies et logiciels*

- **HTML5**: Langage universel du Web.
- **CSS3**: Apparence des interfaces (couleur, typographie, structure...).
- **JavaScript**: Dynamisation des pages, micro-interactions & animations.
- **jQuery**: Bibliothèque JavaScript de référence.
- **Bootstrap**: Framework JavaScript pour site responsive.
- **AngularJS**: Framework JavaScript de Google.
- **NodeJS**: JavaScript coté serveur.
- **Ajax**: Echange de données en JavaScript.
- **PHP**: Gestion des fonctionnalités (espace privé, formulaire, API...).
- **MySQL** : Stockage & gestion des données.
- **Etc.**

# *PLAN*

- *Introduction*
- *Langage HTML5*
- *Langage CSS3*
- *Langage javascript*
- *Langage PHP5*
- *Conclusion*

# Introduction

- Concepts et Définitions
- Types de ressource
- Architecture du Web

# Concepts et Définitions

- **Internet** : C'est un réseau international d'ordinateurs qui communiquent entre eux grâce à des **protocoles** d'échange de données standards.
- **Protocoles** : Un protocole est un **ensemble de règles** qui définissent un langage afin **de faire communiquer plusieurs ordinateurs**.

**Exemples** : TCP/IP, HTTP, DNS, SMTP ...

- Le protocole **HTTP** (**H**yper**T**ext **T**ransfer **P**rotocol) permet un **transfert de fichiers hypertextes**(essentiellement au **format HTML**) localisés **entre un client et un serveur Web**. C'est le protocole le plus utilisé sur Internet .
- **Web**: C'est une **application** qui utilise le réseau **Internet**, et rend possible la recherche d'informations sur les divers **sites** grâce à l'utilisation des **navigateurs** et les adresses **URL**.

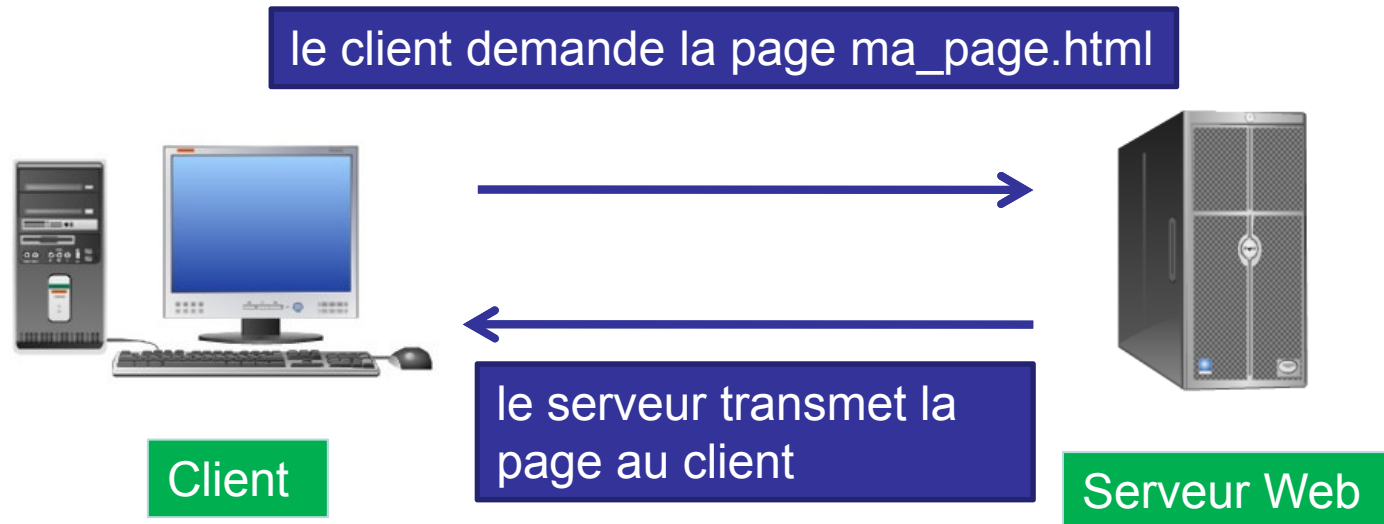
# Concepts et Définitions

- **Pages Web:** des **documents multimédia** composés de **texte, images, son, animations**, etc.
- **Site web:** :ensemble cohérent de pages web liées par des liens hypertextes et articulées autour d'une page d'accueil commune.
- **Serveur web:** **machine connectée à Internet** en permanence et chargée de **fournir les pages web demandées**.
- **URL ( Universal Resource Locator):**
  - ❖ **adresse qui désigne un document proposé par un serveur web.**
  - ❖ C'est un moyen de nommer un objet dans le monde **WWW**.
  - ❖ URL = adresse du serveur Web + adresse de la ressource sur le site
  - ❖ La syntaxe d'un URL est la suivante : **type** :\\**www**.**serveur**.**suffixe**
- **Navigateur:** : Un navigateur est un **logiciel** qui permet **d'accéder** aux **différentes ressources sur internet**. C'est le navigateur web qui fera la traduction entre ces langages informatiques et ce que vous verrez s'afficher à l'écran.



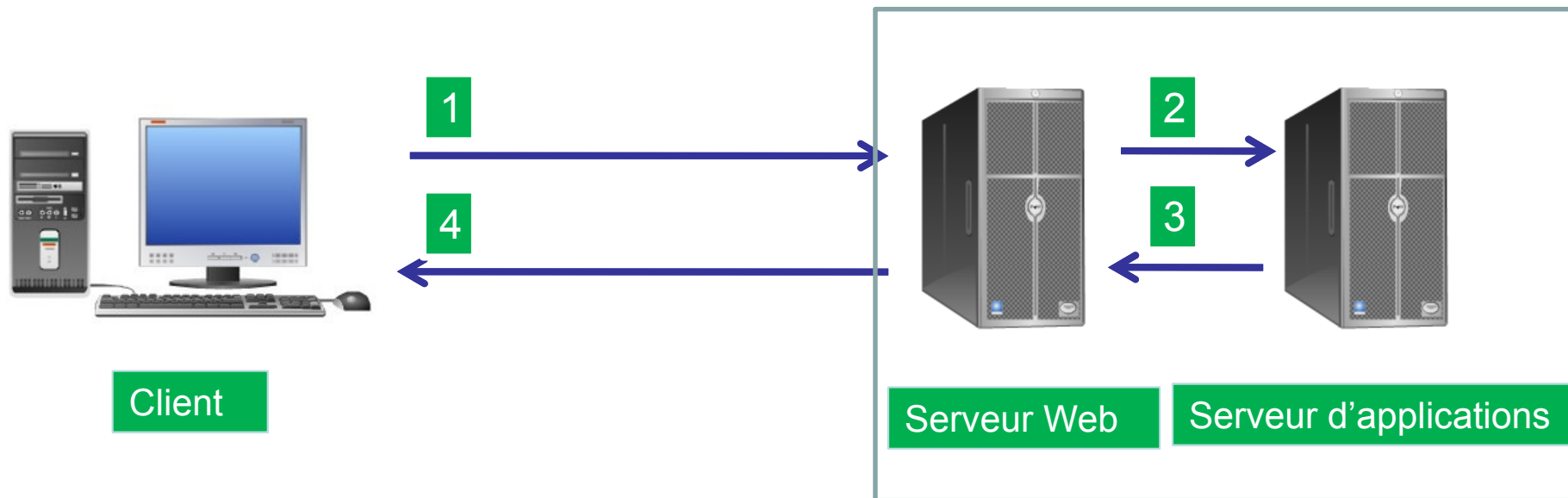
# Concepts et Définitions

- **Sites statiques:** ce sont des **sites** réalisés uniquement à l'aide des langages HTML et CSS.
  - ✓ Ils **fonctionnent** très bien mais leur **contenu ne peut pas être mis à jour automatiquement**
  - ✓ Il faut que le propriétaire du site (le **webmaster**) **modifie le code source pour y ajouter des nouveautés.**



# Concepts et Définitions

- **Sites dynamiques:** plus complexes, ils utilisent d'autres langages en plus de HTML et CSS, tels que *PHP* et *MySQL*. Le contenu de ces sites web est dit « **dynamique** » parce **qu'il peut changer sans l'intervention du webmaster**.



1 -le **Client** demande la page *ma\_page.php*

2-le **Serveur Web** sollicite le **Serveur d'applications** (plateforme PHP, .Net, etc.)

3-le **Serveur d'application** exécute la requête et transmet le résultat au **Serveur Web**

4 -le **Serveur Web** transmet la page au **Client**

# *Types de ressource*

## ❖ Ressources statiques

Fichier texte, XML, HTML, image, son, vidéo

## ❖ Ressources dynamiques côté client

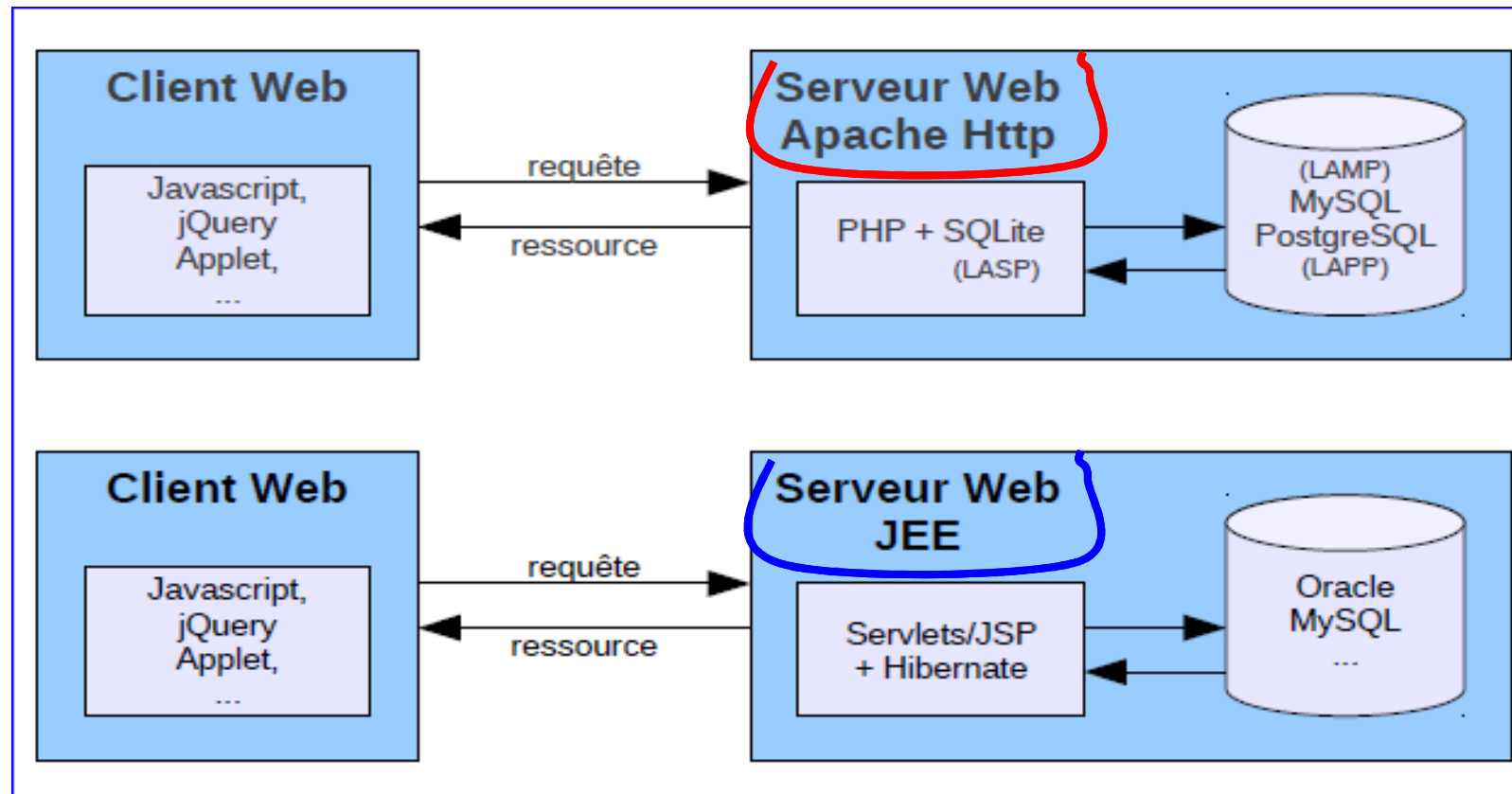
Applet (Java), Javascript/jQuery, ActiveX, ...

## ❖ Ressources dynamiques côté serveur

Servlets/JSP (Java server pages), scripts serveur  
(PHP), CGI (Common Gateway Interface) ...

# Architecture du Web

## ❖ Deux grandes familles d'architecture



# *PLAN*

- *Introduction*
- *Langage HTML5*
- *Langage CSS3*
- *Langage javascript*
- *Langage PHP5*
- *Conclusion*

# *Langage HTML5*

- ➡ Définition
- ➡ Images cliquables
- ➡ Structure d'une page HTML
- ➡ Tableaux
- ➡ Listes
- ➡ Formulaires
- ➡ Insertion des éléments multimédia
- ➡ Cadres
- ➡ Liens hypertexte
- ➡ Conclusion

# Définition

- ✓ Le HTML (Hypertext Markup Language) est **le langage à balises** qui est utilisé pour **construire des documents HTML (pages web)** pouvant être visualisés sur **internet par un navigateur**.
- ✓ Une **page Web** peut incorporer **du texte, des images, de l'animation et du son**.
- ✓ Règles de base :
  - Un fichier HTML "standard" doit impérativement **avoir un nom de fichier** (à ne pas confondre avec le titre de la page) qui se termine par l'extension .html.
  - La **page d'accueil** d'un site Web doit s'appeler **index.html**.
  - Les **commandes HTML (balises)** peuvent être **écrites en minuscules ou en majuscules**.

# Définition

## *Méthode de travail*

les étapes à suivre **pour tester les fichiers HTML** avant de les installer dans les répertoires du serveur :

- ❖ Appelez l'éditeur de texte(**bloc note** ou **Notepad**),
- ❖ **Ecrivez le code html,**
- ❖ **Enregistrez** sous un **nom** avec le suffixe **.html**
- ❖ Afficher votre document HTML Si vous n'êtes pas satisfait...
- ❖ Retournez dans l'éditeur de texte sans quitter le navigateur (Affichage → Source),
- ❖ Corrigez ou modifiez votre code html(fichier→ enregistrer),
- ❖ Utilisez la commande actualiser et juger de vos modifications.



# *Structure d'une page HTML*

## Le langage HTML permet la :

- ❖ **Mises en forme de documents** : polices, styles, tableaux, listes, cadres ...
- ❖ Créations **d'hyperliens**
- ❖ Insertions **d'images**
- ❖ Créations **images réactives**
- ❖ Insertions **d'images animées**
- ❖ Insertions **de composants multimédia** : **son, vidéo...**
- ❖ Définitions de **formulaire**s de saisie : CGI
- ❖ Insertions et activation d'applications interactives : **applets java, scripts**, etc.

# Structure d'une page HTML

## Éléments de base d'une page HTML

- **<HTML>** Définit le **contenu HTML** d'un document nécessite un marqueur de fin.
  - ❖ Règle : **<HTML>.....</HTML>**
- **<HEAD>** Définit **l'en-tête** d'un document HTML nécessite un marqueur de fin.
  - ❖ Règle : **<HEAD>.....</HEAD>**
- **<TITLE>** Définit **le titre** d'un document HTML nécessite un marqueur de fin.
  - ❖ Règle : **<TITLE>.....</TITLE>**
- **<BODY>** Définit le **corps** du document HTML nécessite un marqueur de fin.
  - ❖ Règle : **<BODY>.....</BODY>**

# Structure d'une page HTML

## Structure d'une page HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//FR"  
"http://www.w3.org/TR/html4/strict.dtd">
```

**<HTML>** *début du document HTML*

**<HEAD>** *début de l'en-tête du document*

**<TITLE>**

Titre de votre page

**</TITLE>** *titre du document*

**</HEAD>** *fin de l'en -tête du document*

**<BODY>** *début du corps du document*

Insérer vos textes, images, formulaires, etc. ici

**</BODY>** *fin du corps du document*

**</HTML>** *fin du document HTML*

# Structure d'une page HTML

## Déclaration du type de document

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//FR"  
"http://www.w3.org/TR/html4/strict.dtd">
```

Le "**DOCTYPE**", souvent appelé **DTD** (Déclaration de Type de Document) **permet de vérifier la validité du code des documents**, et s'il correspond bien à la **version de HTML déclarée**. Il existe **trois type de vérificateurs** :

1. le **strict**, qui n'accepte que les **commandes valides pour la version actuelle**:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//FR"  
"http://www.w3.org/TR/html4/strict.dtd">
```

2. le **transitionnel**, qui accepte les commandes de la **version actuelle** et celles de la **version précédente**:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//FR"  
"http://www.w3.org/TR/html4/loose.dtd">
```

3. le **frameset**, un vérificateur transitionnel **qui accepte les cadres dans vos pages**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//FR"  
"http://www.w3.org/TR/html4/frameset.dtd">
```

# Structure d'une page HTML

## En-tête d'une page HTML: <HEAD> </HEAD>

Le bloc d'entête **contient** toutes les **informations nécessaires aux navigateurs et aux moteurs de recherche**:

- ❖ le titre,
- ❖ les mots-clés de la page,
- ❖ la façon d'indexer cette page pour les moteurs de recherche,
- ❖ la langue,
- ❖ la date d'expiration,
- ❖ les copyrights,
- ❖ Etc...

# Structure d'une page HTML

## En-tête d'une page HTML: <HEAD> </HEAD>

### ➤ La balise <META>

Les balises **META** servent à placer des **métadonnées** (metadata) dans une page HTML. On placera ces informations dans l'élément **HEAD**, et **elles ne seront pas affichées sur la page**. La syntaxe de **META** est :

❖ <META name="valeur" content="valeur">

### Exemple

<META name="author" content="votre\_nom">

<META name="version" content="4.0">.

<META name="description" content="description de la page">

<META name=" keywords" content=" mot-clé1, mot-clé2,... ">

<META http-equiv=" content-type" content=" text/html;  
charset=UTF-8">

# Structure d'une page HTML

## En-tête d'une page HTML: <HEAD> </HEAD>

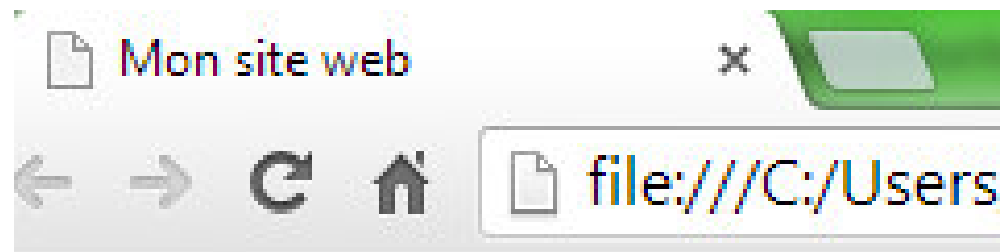
### ➤ La balise **<TITLE> </TITLE>**

Permet de définir le titre affiché dans la barre de titre de la fenêtre de votre navigateur.

❖ **<TITLE>** Apprentissage de HTML **</TITLE>**

**<title>Mon site web </title>**

Titre principal de la page



### **Déclarer d'autres éléments:**

Styles CSS **<Style> </Style>**.

Scripts **<Script ...> </Script>**

# Structure d'une page HTML

## Corps d'une page HTML: <BODY> </BODY>

- ❖ Texte, Tableaux, Listes , Cadres ...
- ❖ Des liens
- ❖ Des images
- ❖ Des images réactives
- ❖ Des images animées
- ❖ Des composants multimédia : son, vidéo...
- ❖ des formulaires de saisie : CGI
- ❖ La mise en forme de la page HTML.



# Structure d'une page HTML

## Corps d'une page HTML: <BODY> </BODY>

### ➤ **Balises de divisions**

- ❖ Les balises **<Hn>** et **</Hn>**: font référence aux différents **titres** et **sous-titres** d'un contenu. On peut **utiliser 6 niveaux de titre**

**<H1>** et **</H1>** (**grand**), jusqu'à **<H6>** et **</H6>** (**petit**)

- ❖ La balise **<BR>** (qui signifie break) permet **de revenir à la ligne**
- ❖ Les balises **<P>** et **</P>** définissent le **début** et la **fin** d'un **paragraphe**. Le paramètre **align** indique une position (**left, right, center**).
- ❖ La balise **<center>** et **</center>** permet de **centrer un texte**
- ❖ La balise **<HR>** : permet de **placer un trait séparateur**. On a plusieurs attributs possibles : "**width**" pour en préciser la **largeur** soit en pixels soit en pourcentage, "**color**" pour lui donner de la couleur.
- ❖ La balise **<pre>** et **</pre>** : cette balise permet **de respecter la mise en page précise d'un texte** ou d'une portion de texte, de la façon dont elle été définie dans le fichier HTML

# Structure d'une page HTML

## Exemple

**<HTML>**

**<HEAD><TITLE>** Deuxième exemple de page HTML **</TITLE> </HEAD>**

**<BODY>**

**<center>**

**<h1> TITRE </h1>**

**<h2> Sous titre </h2>**

**<h3> Chapitre </h3>**

**</center>**

**<HR>**

**<p>** Voici ma deuxième page Web . **</p>**

**<BR>**

Et là, une autre phrase après un retour à la ligne.

**<HR WIDTH = 50% size= 10 color=#000ff>**

**<pre >**

**Prix HT**

**TVA**

**PrixTTC**

**100**

**19.6**

**119.6 Euros**

**200**

**39.2**

**239.2 Euros**

**</pre>**

**Prix HT**

**TVA**

**Prix TTC**

**100**

**19.6**

**119.6 Euros**

**200**

**39.2**

**239.2 Euros**

**</BODY>**

**</HTML>**

# TITRE

## Sous titre

### Chapitre

Voici ma deuxième page Web.

Et là, une autre phrase après un retour à la ligne.

---

Prix HT	TVA	PrixTTC
100	19.6	119.6 Euros
200	39.2	239.2 Euros
300	58.8	358.8 Euros

### **Le même texte, non formaté**

Prix HT TVA Prix TTC 100 19.6 119.6 Euros 200 39.2  
239.2 Euros 300 58.8 358.8 Euros

# Structure d'une page HTML

## Corps d'une page HTML: <BODY> </BODY>

### ➤ *Balises de mise en forme*

- ❖ **<B>** et **</B>** Cette balise commence un texte en **caractères gras**.
- ❖ **<I>** et **</I>** Cette balise stylise un texte en **caractères italiques**.
- ❖ **<U>** et **</U>** Cette balise modifie un texte en caractères **soulignés**. (Elle n'est pas reconnue par tous les navigateurs).
- ❖ **<STRIKE>** ou **<S>** et **</S>** Cette balise va **barrer le texte sélectionné**.
- ❖ **<SUB>** et **</SUB>** Cette balise va placer **le texte sélectionné en indice**.
- ❖ **<SUP>** et **</SUP>** Cette balise va placer **le texte sélectionné en exposant**.

# Structure d'une page HTML

## Corps d'une page HTML: <BODY> </BODY>

### ➤ *Balises de mise en forme*

❖ **<FONT>** et **</FONT>** permet de **modifier l'apparence du texte**

- L'attribut **SIZE** Permet de modifier la **taille** du texte compris entre <font> et </font>
- L'attribut **COLOR** Permet de mettre le texte en **couleur**
- L'attribut **FACE** Permet de **choisir la police** dans laquelle le texte sera affiché

❖ **<BIG>** et **</BIG>** Cette balise va **agrandir la taille** de la police affichée.  
Synonyme de **<FONT SIZE=+1> </FONT>**.

❖ **<SMALL>** et **</SMALL>**

Cette balise va **réduire la taille** de la police. Synonyme de **<FONT SIZE=-1> </FONT>**

❖ **<!--** et **-->** Ces balises permettent **de commenter votre code**. Le texte compris entre ces balises ne sera pas pris en compte par le navigateur.

# Structure d'une page HTML

## Exemple :

<HTML>

<HEAD>

<TITLE> Texte en gras, italique, centre, taille et police </TITLE>

</HEAD>

<BODY>

<B> texte en gras </B> <BR>

<I> texte en italique </I> <BR>

<U> texte souligné </U> <BR>

<!-- On peut également imbriquer les balises -->

<B> <CENTER> texte centré en gras </CENTER> </B>

<B>

<FONT color="red" size=5 face="Arial,Times New Roman" > texte  
en rouge, gras, Arial ou Times New Roman de taille5 </FONT>

</B>

<BR>

</BODY>

</HTML>

# Résultat

**texte en gras**

*texte en italique*

texte souligné

**texte centré en gras**

**texte en rouge, gras, Arial ou Times New  
Roman de taille 5**

# HTML : les listes

## ➤ Listes ordonnées

❖ **<OL>** et **</OL>** Indique le début et la fin d'une **liste numérotée**

■ L'attribut "**type**" définit la numérotation avec des **chiffres** ou des **lettres** :

- **<OL TYPE=1>** donnera la numérotation par **défaut** de la liste.
- **<OL TYPE=a>** donnera une numérotation **alphabétique** en **minuscule** (a, b, c, ...)
- **<OL TYPE=A>** donnera une numérotation alphabétique en **majuscule** (A, B, C, ...)
- **<OL TYPE=i>** donnera une numérotation en lettre **latine** en **minuscule** (i, ii, iv, xci,...)
- **<OL TYPE=I>** donnera une numérotation en lettre **latine** en **majuscule** (I, II, IV, XCI,...)



# HTML : les listes

## ➤ Listes ordonnées

- L'attribut **START**=“nombre “ **Spécifié la valeur initiale** (1 par défaut)

❖ **<LI>** Désigne un **nouvel élément d'une liste**.

### Exemple 1

```
<!DOCTYPE html>
<html>
  <head>
    <title> page web </title>
  </head>
  <body>
    <h1> Les listes </h1>
    <!--Deux exemples de liste ordonnées-->
    <p> Liste n°1: </p>
    <ol>
      <li> les listes numérotées (élément OL)</li>
      <li> les listes non numérotées (élément UL) </li>
      <li> les listes descriptives (élément DL) </li>
    </ol>
    <p> Liste n°2: </p>
    <ol>
      <li> Introduction </li>
      <li> Partie I </li>
      <li> Partie II </li>
      <li> Conclusion </li>
    </ol>
  </body>
</html>
```

### Résultat

## Les listes

Liste n°1:

1. les listes numérotées (élément OL)
2. les listes non numérotées (élément UL)
3. les listes descriptives (élément DL)

Liste n°2:

1. Introduction
2. Partie I
3. Partie II
4. Conclusion

# HTML : les listes

## ➤ Listes ordonnées

### Exemple 2

```
<!DOCTYPE html>
<html>
  <head>
    <title> page web </title>
  </head>
  <body>
    <h1> Les listes </h1>

    <!--Deux exemples de liste ordonnées-->
    <p> Liste n°1: </p>
    <ol type="A">
      <li> les listes numérotées (élément OL)</li>
      <li> les listes non numérotées (élément UL) </li>
      <li> les listes descriptives (élément DL) </li>
    </ol>
    <p> Liste n°2: </p>
    <ol type="i">
      <li> Introduction </li>
      <li> Partie I </li>
      <li> Partie II </li>
      <li> Conclusion </li>
    </ol>
  </body>
</html>
```

### Résultat

## Les listes

Liste n°1:

- A. les listes numérotées (élément OL)
- B. les listes non numérotées (élément UL)
- C. les listes descriptives (élément DL)

Liste n°2:

- i. Introduction
- ii. Partie I
- iii. Partie II
- iv. Conclusion

# HTML : les listes

## ➤ Listes non ordonnées

❖ **<UL>** et **</UL>** Indique le début et la fin d'une **liste non ordonnée**.

■ L'attribut **"type"** définit le types des **puces** **cercle**, **carrée** et **disque plein**

■ **<UL TYPE=disc>** donnera une puce

■ **<UL TYPE=square>** donnera un carré

■ **<UL TYPE=circle>** donnera un cercle vide

❖ **<LI>** Désigne un nouvel élément d'une liste.

Exemple

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ma première page HTML</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Les listes</h1>

    <!--Un exemple de liste non-ordonnée-->
    <ul>
      <li>Pomme</li>
      <li>Vélo</li>
      <li>Guitare</li>
    </ul>
  </body>
</html>
```

Résultat

## Les listes

- Pomme
- Vélo
- Guitare

# HTML : les listes

## ➤ Listes descriptives

### ❖ **<DL>** et **</DL>**

Indique le début et la fin d'une liste **descriptive**.

### ❖ **<DT>**

L'élément **<DT>** précède donc chaque élément de la liste (spécifier un élément).

### ❖ **<DD>**

La balise **<DD>** correspond à la zone de définition de l'élément.

Exemple

Résultat Html

**<DL>**

**<DT>**html    **<DD>** facile

**<DT>**pascal **<DD>**moyen

**<DT>**c++    **<DD>**dur

**<DT>**cobol   **<DD>**très dur

**</DL>**

**Facile**

**Pascal**

**Moyen**

**C++**

**Dur**

**Cobol**

**Très dur**

# HTML : les listes

## ➤ Listes Imbriquées

Exemple

Résultat

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ma première page HTML</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Les listes</h1>

    <!--Listes imbriquées-->
    <ol>
      <li>Introduction</li>
      <li>Partie I
        <!--On imbrique une liste non-ordonnée dans une liste ordonnée-->
        <ul>
          <li>Définitions</li>
          <li>Auteurs</li>
          <li>Exemples</li>
        </ul>
      </li>
      <li>Partie II</li>
      <li>Conclusion</li>
    </ol>
  </body>
</html>
```

## Les listes

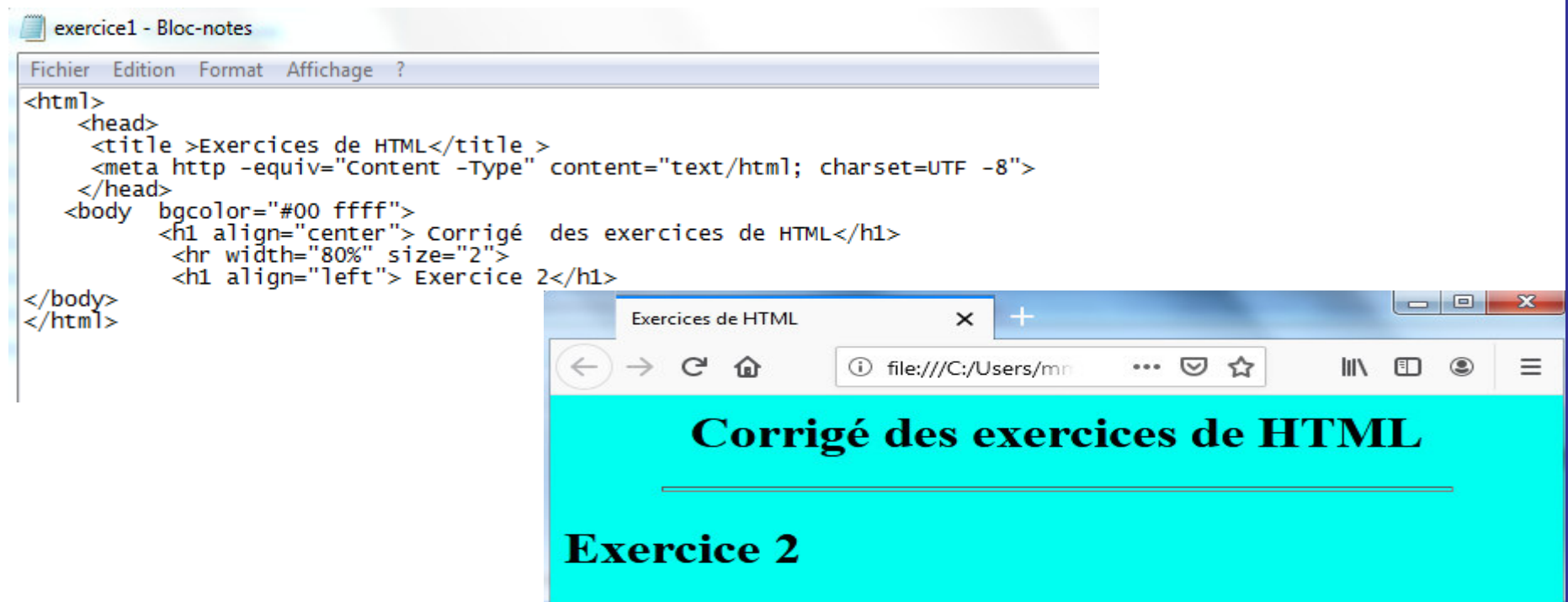
1. Introduction
2. Partie I
  - Définitions
  - Auteurs
  - Exemples
3. Partie II
4. Conclusion

# HTML : Exercices

## Exercice 1

Créer un fichier nommé exercice.html et on y met les balises : html, head, body. Le titre du document est : Exercices de HTML.

1. Changer la couleur du fond du document en #00ffff.
2. Placer le titre Corrigé des exercices de HTML centré en haut de la page.
3. Insérer une ligne horizontale de largeur 80% et d'épaisseur 2.
4. Placer le sous-titre Exercice 2 aligné à gauche.



# HTML : Exercices

## Exercice 2

Reproduisez cette liste:

- I. Chapitre 1
- II. Chapitre 2
  - i. Section 1
  - ii. Section 2
    - Graphe 1
    - Graphe 2
- III. Chapitre 3
  - Image 1
  - Image 2

```
exercice2 - Bloc-notes
Fichier Edition Format Affichage ?
<html>
  <head>
    <title>Exercices de HTML</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body bgcolor="#00ffff">
    <h2 align="left" style="border: 1px solid black;">Exercice 3</h2>
    <ol type="I">
      <li>Chapitre 1</li>
      <li>Chapitre 2
        <ol type="i">
          <li>Section 1</li>
          <li>Section 2
            <ul type="disc">
              <li>Graphe 1</li>
              <li>Graphe 2</li>
            </ul>
          </li>
        </ol>
      </li>
      <li>Chapitre 3
        <ul type="square">
          <li>Image 1</li>
          <li>Image 2</li>
        </ul>
      </li>
    </ol>
  </body>
</html>
```

# HTML : Insertion des images

❖ **La balise <IMG>** Cette balise permet **d'insérer une image** dans votre page.

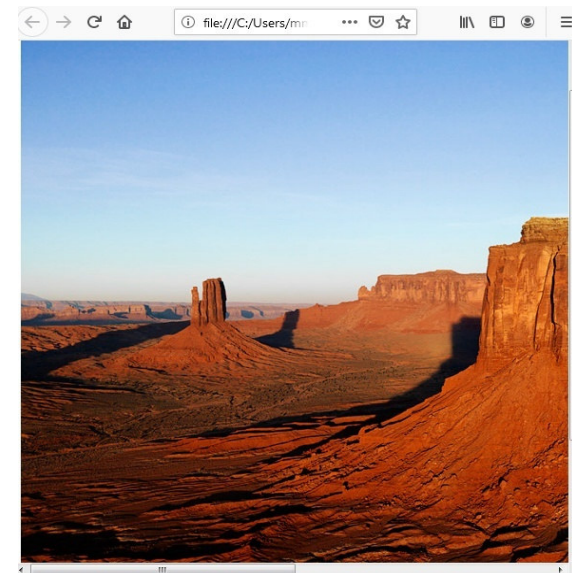
Cette balise **sera toujours complétée** par l'attribut **SRC** qui permet de **donner l'adresse du fichier graphique contenant l'image** :

**<IMG SRC="/repertoire/sous-repertoire/nom\_du\_fichier\_graphique">**

```
<!DOCTYPE html>
<html>
  <head>
    <title> insertion d'image en html </title>

  </head>
  <body>
    <p>
      <IMG src="Desert.jpg">
    </p>

  </body>
</html>
```





# HTML : Insertion des images

## ➤ Les attributs de la balise <IMG>

- **ALIGN=TOP** pour aligner le **haut de l'image** sur la ligne courante
- **ALIGN=MIDDLE** pour aligner le **milieu de l'image** sur la ligne courante
- **ALIGN=BOTTOM** pour aligner le **bas de l'image** sur la ligne courante
- **ALIGN= LEFT** et **RIGHT** font **flotter l'image** à **gauche** ou à **droite** du texte.
- **WIDTH=l** et **HEIGHT=H** où l et H représentent la **largeur** et la **hauteur** de l'image en pixels.

# HTML : Insertion des images

## ➤ Les attributs de la balise <IMG>

- **BORDER=n** où **n** représente l'épaisseur du trait **bordant** l'image.
- **HSPACE=n** où **n** représente le **nombre de pixels séparant horizontalement** l'image du texte ou d'une autre image qui lui serait accolé.
- **VSPACE=n** ; même chose que précédemment mais pour la **verticale**.
- L'attribut "**alt**"  
"**alt**" (comme alternative) permet de **préciser un texte qui sera affiche** si **l'image n'est pas trouvée** ou bien si vous laissez votre souris dessus.

# HTML : Insertion des images

## ➤ Exemples

```
<!DOCTYPE html>
<html>
  <head>
    <title> insertion d'image en html </title>
  </head>
  <body>
    <P>
      <!--explication si l'image n'apparaît pas-->
      <IMG src="Desert.jpg" alt="erreur">
    </P>

    <P>
      <!--image avec bordure-->
      <IMG src="Desert.jpg" border="10">
    </P>

    <P>
      <!--image avec largeur précise-->
      <IMG src="Desert.jpg" width="200">
    </P>

    <P>
      <!--image avec largeur et longueur précises-->
      <IMG src="Desert.jpg" width="200" height="200">
    </P>

    <P>
      <!--image avec largeur proportionnelle à celle de l'image-->
      <IMG src="Desert.jpg" border="1">
    </P>

  </body>
</html>
```

# HTML : Insertion d'un fichier audio

## ➤ **Formats de fichier audio**

Pour diffuser de la musique ou n'importe quel son, il existe de nombreux formats

### *Remarques*

- **Aucun navigateur ne gère tous ces formats à la fois.** Retenez surtout la compatibilité pour les **MP3** et **OGG**.
- MP3: Internet Explorer, Chrome, Safari.
- OGG: Chrome, Firefox, Opera.
- Vous pouvez utiliser un bon **convertisseurs AUDACITY** pour convertir un fichier audio d'un type à l'autre.
- La balise **<source>** va nous permettre de résoudre la problématique des **formats de fichiers différents**.

# HTML : Insertion d'un fichier audio

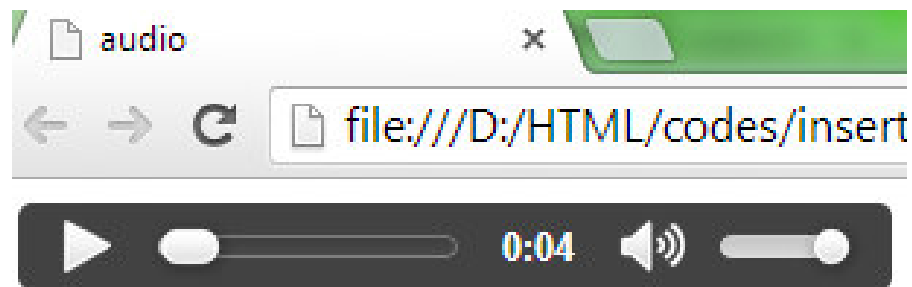
## ➤ Insertion d'un fichier audio

L'insertion d'un fichier audio se réalise très simplement par la balise :

```
<audio src="fichier_son"> </audio>
```

exemple

```
<body>  
  <audio src="music.mp3" controls>  
    Votre navigateur ne supporte pas la balise audio.  
  </audio>  
</body>
```



# HTML : Insertion d'un fichier audio

## ➤ Insertion d'un fichier audio

### ❑ Attributs de la balise `audio` :

- **src**: Définit l'adresse du fichier son.
- **controls**: Affiche les **contrôles du lecteur audio**. Il comportent les fonctions de lecture, d'arrêt, d'avancement et de volume.
- **autoplay**: Définit la **lecture automatique du fichier audio** dès que celui-ci sera disponible, soit au chargement de la page.
- **loop**: Spécifie que le **fichier son sera joué en boucle**. Ainsi, le morceau sonore est joué à nouveau lorsqu'il se termine.
- **preload**: Indique au navigateur qu'il doit **télécharger** le fichier audio au **chargement**. Cet attribut peut être précisé :
  - `preload="none"`: Pas de préchargement.
  - `preload="metadata"`: Préchargement des métadonnées (metadata) attachées au fichier.
  - `preload="auto"`: Préchargement automatique.

# HTML : Insertion d'un fichier audio

## ➤ Insertion d'un fichier audio

- La balise `<source>` va nous permettre de résoudre la problématique des **formats de fichiers différents**.
- La balise `<source>` est utilisée pour **spécifier plusieurs ressources audio**.
- À charge du navigateur de choisir le format qu'il prend en compte.

## Exemple

```
<audio controls>  
  <source src="music.ogg">  
  <source src="music.mp3">  
  <source src="music.acc">  
</audio>
```

# HTML : Insertion d'une vidéo

## ➤ Insertion de la vidéo

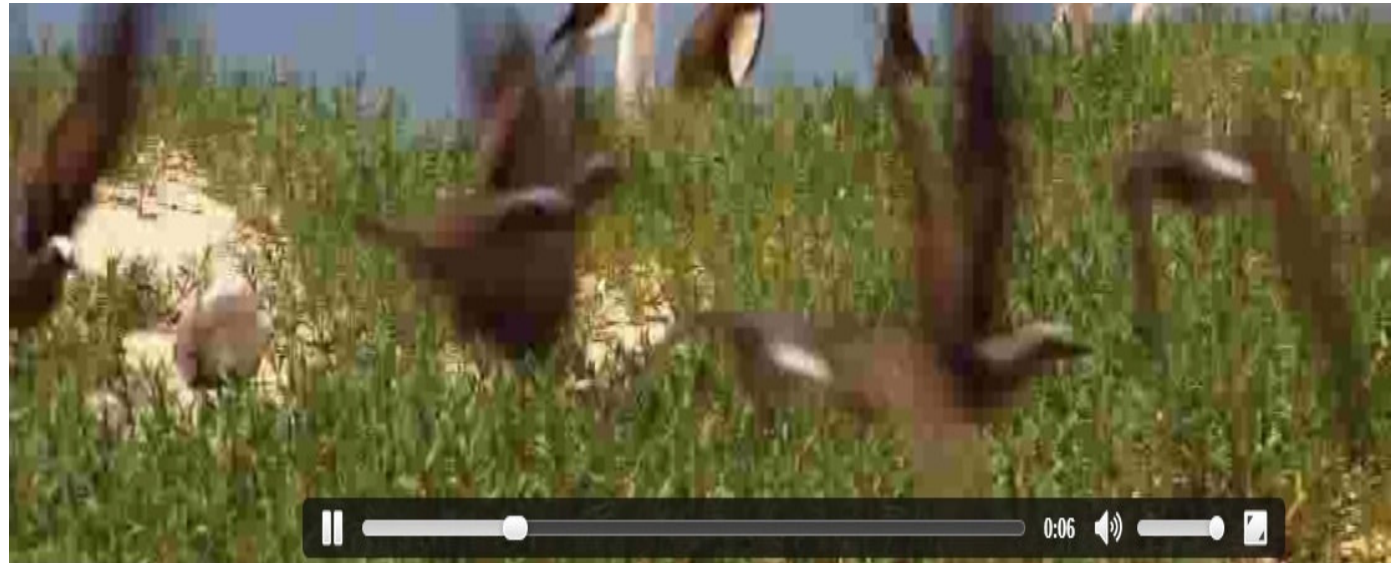
L'insertion est faite par la balise <video>:

```
<video src="fichier_video"> </video>
```

### Exemple

```
<video src="video.ogv" controls>
```

```
  Votre navigateur ne supporte pas la balise vidéo.  
</video>
```





# HTML : Insertion d'une vidéo

## ➤ *Insertion de la vidéo*

### Attributs de la balise `<video>`:

- **src**: Définit l'adresse du fichier vidéo.
- **width**: Détermine la **largeur** de la vidéo.
- **height**: Détermine la **hauteur** de la vidéo.
- **poster**: L'attribut **poster** permet de spécifier une image que le navigateur utilisera alors que la vidéo est en cours de téléchargement ou jusqu'à ce que l'utilisateur commence la lecture de la vidéo.

`<video src="video.ogv" poster="image.png">.`

- **Controls**: Affiche les **contrôles du lecteur** de vidéo. En l'absence de l'attribut `controls`, les contrôles du lecteur ne seront pas affichés par le navigateur.
- **autoplay**: Définit la **lecture automatique du fichier vidéo** dès que celui-ci sera disponible, soit au chargement de la page:

`<video src="video.ogv" autoplay>`

# HTML : Liens hypertextes

➤ Il existe 3 types de liens hypertextes

1. les liens **externes** (vers un autre site),
2. les liens **internes** à **votre site**
3. les liens **internes** à une **page**.

➤ Chaque type de lien requiert un certain **chemin d'accès**.

➤ **Chemin d'accès ?**

**Un chemin d'accès, c'est l'emplacement de la page de destination par rapport à la page où l'on se trouve.**

➤ Il y a deux types de chemins d'accès :

1. les chemins **absolus** (on **précise l'adresse complète**, par exemple : <http://www.soswindows.net>)
2. les chemins **relatifs** (on **précise le chemin parmi les répertoire de l'espace disque**, par exemple : [../annuaire/index.html](#)).

# HTML : Liens hypertextes

## ❖ Balise <A> et </A>

- Permet de créer un **lien hypertexte**. Son attribut indispensable est "**href**". En effet, il permet de **préciser l'adresse de la page de destination du lien** qui sera créé. Cette adresse peut être soit **absolue** soit **relative**.
- Si vous souhaitez faire un **lien interne à une page**, il faut créer une **ancree**. On fait le lien vers cet ancre comme ceci :

`<A href="#nom_ancre">Texte1</A>`

et on crée l'ancre comme ceci :

`<A name="nom_ancre">Texte2`

# HTML : Liens hypertextes

## <BODY>

Si la page vers laquelle est faite le lien **se trouve dans le même répertoire que la page en cours** : <BR>

<A href="page2.html">Lien vers la page 2</A> <BR><BR>

Si la page de destination est dans **un sous-répertoire de celui de la page courante** : <BR>

<A href="rep1/page2.html">Lien vers une page dans le sous-répertoire nommé "rep1"</A> <BR><BR>

<A href="http://www.votre-site.com">Lien vers un site externe</A>  
<BR><BR><BR>

<A href="#ligne20">Lien vers l'ancre 1</A> <!-- lien vers l'ancre nommé ancre1 -->

<BR><BR><BR><BR><BR><BR><BR><BR><BR><BR>

<A name="ligne20">Texte</A> <!-- définition de l' ancre1 -->

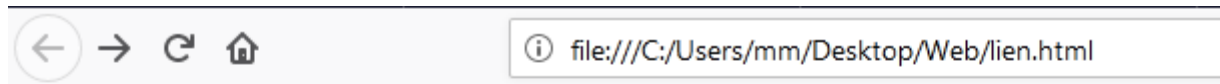
## </BODY>

# HTML : Liens hypertextes

## Exemples :

Créer un lien vers la page d'accueil de Wikipédia

```
<!DOCTYPE html>
<html>
  <head>
    <title>Cours HTML et CSS</title>
    <meta charset= 'utf-8'>
  </head>
  <body>
    <p>Pour créer des liens en HTML, on utilise l'élément a et son attribut href.</p>
    <p>Ex : lien vers <a href="https://www.wikipedia.org/">la home de Wikipedia</a></p>
  </body>
</html>
```



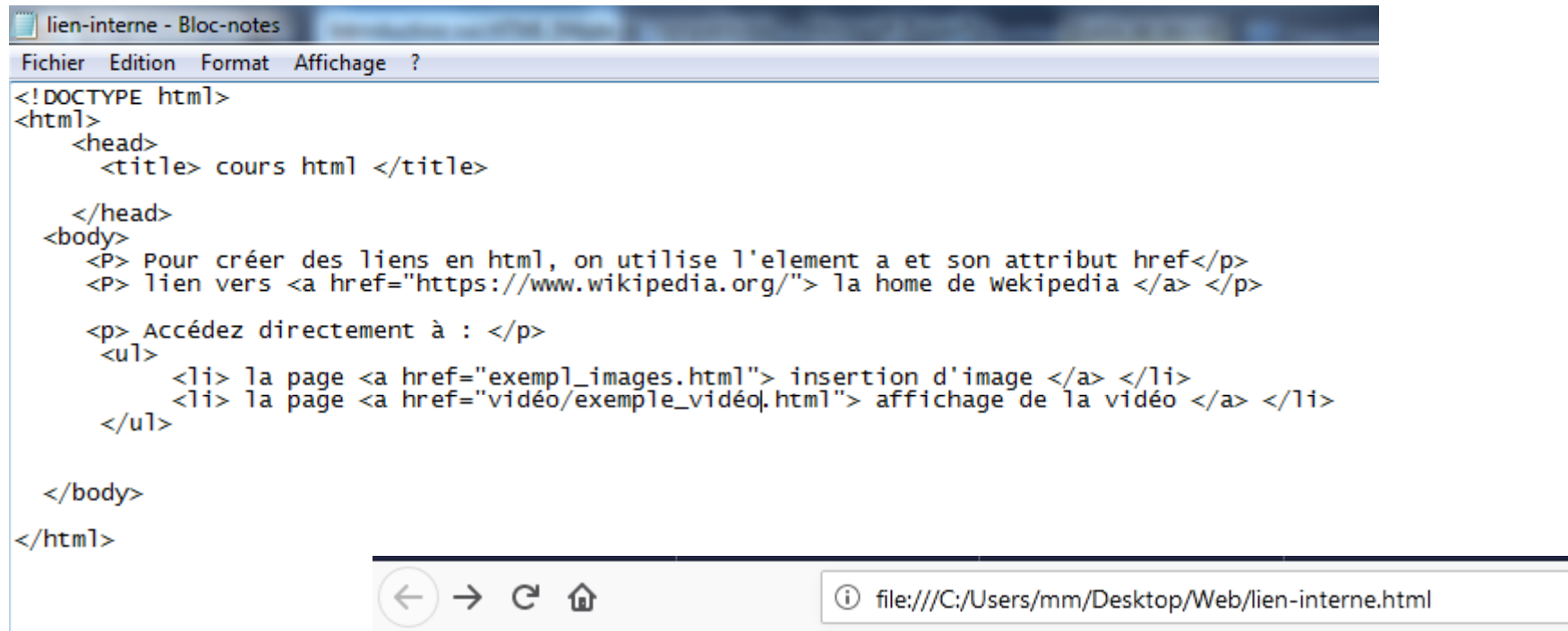
Pour créer des liens en html, on utilise l'element a et son attribut href

lien vers [la home de Wikipedia](https://www.wikipedia.org/)

# HTML : Liens hypertextes

## Exemples :

### Créer des liens internes en HTML



Pour créer des liens en html, on utilise l'element a et son attribut href

lien vers [la home de Wikipedia](#)

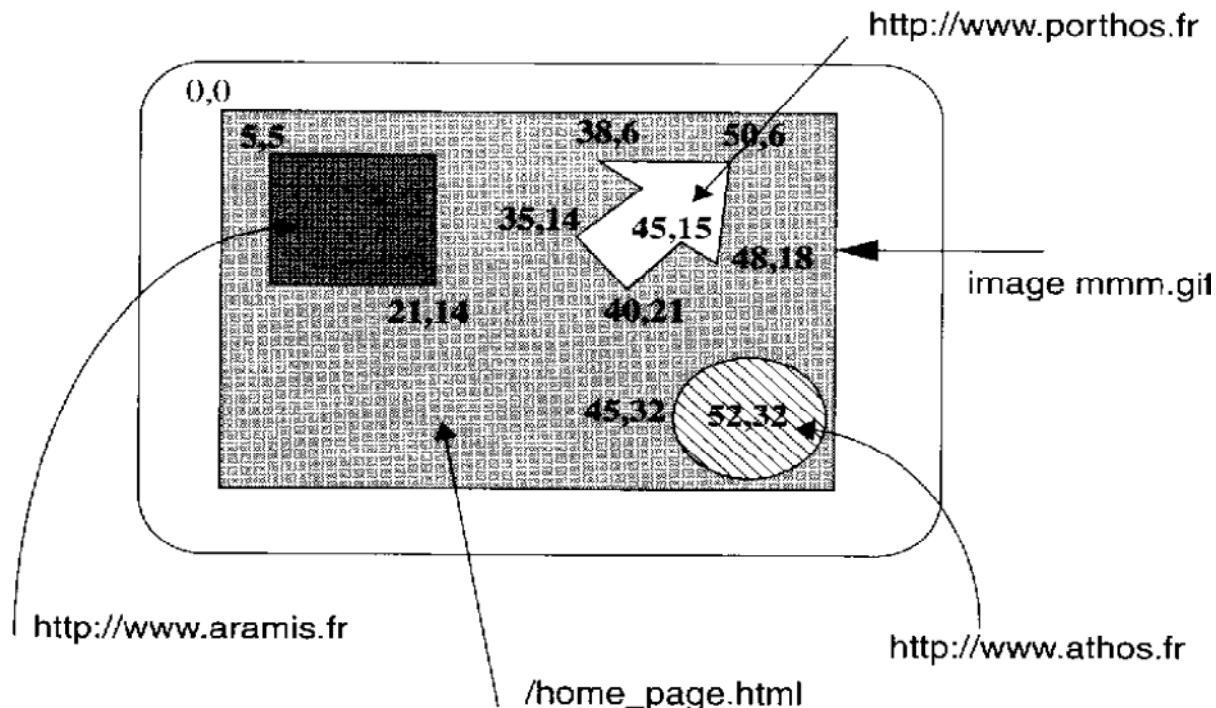
Accédez directement à :

- la page [insertion d'image](#)
- la page [affichage de la vidéo](#)

# HTML : images réactives(Cliquables)

## ➤ Images réactives(Cliquables)

- Une image réactive est **décomposée en différentes parties** permettant **d'accéder à des pages différentes**. Etant donné que l'utilisateur doit savoir où cliquer, les différentes parties doivent être séparées correctement.
- Pour réaliser une image cliquable, on commencera par **mettre l'image à la taille souhaitée**. En effet la seconde opération va consister à **faire un relevé des coordonnées zone par zone** des surfaces cliquables.



# HTML : images réactives(Cliquables)

## Création des images réactives

➤ Une **image réactive** est définie par les balises **<map>** et **</map>**. Pour faire référence à la définition de l'image réactive en utilisant l'attribut **name**.

### ➤ Syntaxe

```
  
<map name="nomdelamap">  
  <area [shape="forme"] coords="x,y,..." [href="URL"] [nohref]>  
</map>
```

➤ **USEMAP** Le paramètre USEMAP se met à l'intérieur d'une commande IMG et **indique** au client **que l'image en question est cliquable**. La valeur du paramètre USEMAP contient le nom d'une section MAP

- ❖ qui peut se trouver dans le même document  
(auquel cas on le déclare en commençant par #, e.g. "#nom\_carte")
- ❖ ou dans un autre document ("URL#nom\_de\_carte").



# *HTML : images réactives(Cliquables)*

## Création des images réactives

- **shape** : permet de définir la forme d'une zone

Vous pouvez utiliser les valeurs suivantes : **rect**, **poly**, **circle**. Lorsque deux régions se superposent, le navigateur utilise la première de la liste.

- **coords** : définit une **liste de coordonnées** séparées par une **virgule** de la région.

- **href** : définit l'**URL de la ressource internet** à laquelle la région est liée.

- **nohref** : indique que la région est une zone qui n'est liée à aucune ressource internet.

# HTML : images réactives(Cliquables)

## Création des images réactives

### ➤ Exemple

Source

```
  
<map name="mamap">  
  <area shape="rect" coords="0,0,50,50" href="news.htm">  
  <area shape="rect" coords="51,0,100,50" href="contact.htm">  
  <area shape="rect" coords="101,0,150,50" href="apropos.htm">  
</map>
```

Résultat



# HTML : Les Tableaux

## Création des tableaux

### ➤ **<TABLE>** et **</TABLE>**

La balise **<TABLE>** permet l'ouverture d'un tableau ; la fin de tableau est spécifiée par **</TABLE>**.

### ➤ **<TR>**

Cette balise **début** une **nouvelle ligne** dans le tableau. La balise de fin (**</TR>**) n'est pas obligatoire.

### ➤ **<TD>**

**crée une nouvelle cellule dans un tableau.** Cette cellule doit être contenue dans une ligne et donc une balise "**<TR>**" doit être déjà ouverte. la balise de fin (**</TD>**) est optionnelle.

# HTML : Les Tableaux

## Exemple

**<BODY>**

Voici notre premier tableau avec un bord  
de 1 :

**<BR><BR>**

**<TABLE border=1>**

**<TR>**

**<TD>**Cellule 1**</TD>**

**<TD>**Cellule 2**</TD>**

**</TR>**

**<TR>**

**<TD>**Cellule 3**</TD>**

**<TD>**Cellule 4**</TD>**

**</TR>**

**</TABLE>**

**</BODY>**

Voici notre premier tableau avec  
un border de 1 :

Cellule 1

Cellule 2

Cellule 3

Cellule 4

# HTML : Les Tableaux

- L'attribut **WIDTH=n** (pixels) ou n%  
Permet de **préciser la largeur d'un tableau** ou **d'une cellule** par rapport de la largeur de fenêtre. La largeur peut être donnée en pixel ou en pourcentage.
- L'attribut **HEIGHT=n**  
Cet attribut détermine la **hauteur d'une cellule <TD>** ou **du tableau entier dans la balise <TABLE>**.

## Les attributs de la balise <TR>

- |   |  |
|---|--|
| <ul style="list-style-type: none"><li>➤ L'attribut <b>VALIGN</b><br/><b>VALIGN=x</b> (alignement vertical) qui peut prendre les valeurs :<ul style="list-style-type: none"><li>❖ <b>TOP</b> place le texte en <b>haut</b> de la cellule,</li><li>❖ <b>BOTTOM</b> en <b>bas</b> de la cellule,</li><li>❖ <b>MIDDLE</b> au <b>centre</b> de la cellule.</li></ul></li></ul> | <ul style="list-style-type: none"><li>➤ L'attribut <b>ALIGN</b><br/><b>ALIGN=x</b> (alignement horizontal) qui peut prendre les valeurs :<ul style="list-style-type: none"><li>❖ <b>RIGHT</b> place le texte à <b>droite</b> de la cellule,</li><li>❖ <b>LEFT</b> à <b>gauche</b> de la cellule,</li><li>❖ <b>CENTER</b> <b>centré</b> le texte dans la cellule.</li></ul></li></ul> |
|---|--|

# HTML : Les Tableaux

## Les attributs de la balise <TD>

➤ Les attributs **COLSPAN** et **ROWSPAN**

❖ **COLSPAN=n** permet de faire **fusionner** n cellules d'une même **ligne**.

❖ **ROWSPAN=n** permet de faire **fusionner** n cellules d'un même **colonne**.

COLSPAN="3"		
Cellule 12c1	Cellule 12c2	Cellule 12c3
Cellule 13c1	Cellule 13c2	Cellule 13c3

ROWSPAN="3"	Cellule 11c2	Cellule 11c3
	Cellule 12c2	Cellule 12c3
	Cellule 13c2	Cellule 13c3

# HTML : Les Tableaux

## Exemple

Fichier Edition Format Affichage ?

```
<html>
  <head> <title> tableau en html </title> </head>
<body>
<table border=1>
  <tr>
    <td>Nom</td>
    <td>Prénom</td>
    <td>Age</td>
    <td>Mail</td>
  </tr>
  <tr>
    <td>xxx</td>
    <td>ali</td>
    <td>36</td>
    <td>xxx.ali@gmail.com</td>
  </tr>
  <tr>
    <td>yyy</td>
    <td>mostafa</td>
    <td>28</td>
    <td>yyy.mostafa@gmail.com</td>
  </tr>
</table>
</body>
</html>
```

Nom	Prénom	Age	Mail
xxx	ali	36	xxx.ali@gmail.com
yyy	mostafa	28	yyy.mostafa@gmail.com

# HTML : Les formulaires

## Création des formulaires

➤ <FORM> et </FORM> Elles délimitent un formulaire.

❖ L'attribut **ACTION** Désigne l'adresse du script(ou E-mail) qui va traiter les données acquises depuis le formulaire.

❖ L'attribut **METHOD**

**GET**: Avec cette méthode, les données du formulaire seront encodées dans une URL. Les données sont séparées de l'adresse de la page par le code ? et entre elles par le code &(limitée à 255 caractères).  
<http://www.xul.fr/page.html?couleur=bleu&forme=rectangle>

**POST**: c'est la méthode la plus utilisée pour les formulaires car elle permet d'envoyer un grand nombre d'informations. Les données du formulaire n'apparaissent pas dans l'URL.



# HTML : Les formulaires

## ➤ <INPUT> (pas de balise de fin !)

Cette balise permet de **placer un champ dans lequel les informations peuvent être recueillies.**

### ❖ L'attribut "**TYPE**"

Spécifie **le type de champ**. Il peut prendre plusieurs valeurs :

- **'text'** pour **entrer** du simple texte,

- **'hidden'** pour un **champ caché non visible** par le visiteur,

- **'password'** pour **entrer un mot de passe** dans lequel les caractères sont remplacés par des \*,

- **'submit'** pour créer un bouton pour **envoyer le formulaire**

- **'reset'** pour créer un bouton pour **effacer(vider) le formulaire** et le remet comme au chargement de la page.

# HTML : Les formulaires

## ❖ L'attribut "**value**"

Permet de donner une **valeur à un champ** avant que le visiteur ne commence à le remplir. Dans la cas d'un bouton (submit ou reset), cette valeur sera le texte affiché sur le bouton !

## ❖ L'attribut "**size**"

Donne **la largeur du champ** (20 par défaut).

## ❖ L'attribut "**maxlength**"

définit le **nombre maximal de caractères** pouvant être **tapés dans un champ**.

# HTML : Les formulaires

## Exemple 1

```
<FORM action="mailto:vous@votre-domaine.com" method="post" >  
  <INPUT type="hidden" name="intro" value="Voici les infos récoltées via le  
  formulaire:">  
  Entrez votre prénom : <INPUT type="text" name="prenom">  
  <BR>Entrez votre nom : <INPUT type="text" name="nom" size=25> <BR>  
  Entrez l'adresse de votre site : <INPUT type="text" name="url" size=40  
  value="http://">  
  <BR>  
  Entrez un mot de passe : <INPUT type="password" name="pass" size=10  
  maxlength=10>  
  <BR><INPUT type="submit" value="Envoyer" name="submit">  
  <INPUT type="reset" value="Effacer les données" name="reset">  
</FORM>
```

Entrez votre prénom :

Entrez votre nom :

Entrez l'adresse de votre site :

Entrez un mot de passe :

# HTML : Les formulaires

## ➤ <SELECT> et </SELECT>

Crée une **liste déroulante**.

❖ L'attribut "**name**"

Il permet **de donner un nom**

❖ L'attribut "**size**" (1 par défaut).

La valeur donnée à l'attribut size donne alors le **nombre de lignes visibles** dans la fenêtre.

## **<OPTION>**

**Ajoute un élément à la liste déroulante** créée avec un **SELECT**. Par défaut, c'est la première balise option qui sera sélectionné. Si vous voulez que ce soit une autre, il faut ajouter le mot suivant dans la balise en question : '**selected**'.

# HTML : Les formulaires

## Exemple 2

### <FORM>

Comment trouvez-vous ce tutorial jusqu'à présent

```
<SELECT name="liste">
```

```
<OPTION value="excellent">Excellent
```

```
<OPTION value="bon" selected>Bon
```

```
<OPTION value="moyen">Moyen
```

```
<OPTION value="faible">Faible
```

```
<OPTION value="nul">Pitoyable
```

```
</SELECT>
```

Comment trouver ce tutorial jusqu'à présent :

<BR><BR>Ou bien, on peut préciser le nombre de valeur à afficher :

```
<SELECT name="liste" size=3>
```

```
<OPTION value="excellent">Excellent
```

```
<OPTION value="bon" selected>Bon
```

```
<OPTION value="moyen">Moyen
```

```
<OPTION value="faible">Faible
```

```
<OPTION value="nul">Pitoyable
```

```
</SELECT>
```

```
</FORM>
```

Ou bien, on peut préviser le nombre de valeur à afficher :

Excellent
Bon
Moyen

# HTML : Les formulaires

## ➤ <TEXTAREA> et </TEXTAREA>

La balise <TEXTAREA> permet de **créer une fenêtre avec ascenseurs horizontaux et verticaux** dans laquelle on pourra saisir du texte.

La valeur donnée aux attributs **ROWS=n** (lignes) et **COLS=n** (colonnes) délimite la taille de cette fenêtre.

## <INPUT>

### ❖ L'attribut "TYPE"

On peut ajouter deux autres valeurs pour l'attribut type :

- **"checkbox"** : crée une **case à cocher**. **checked**: cochée ou non au chargement de la page.
- **"radio"** : crée un **bouton radio**. **checked** : sélectionné ou non au chargement de la page.

# HTML : Les formulaires

Exemple 3. créer une fenêtre avec ascenseurs horizontaux et verticaux

**<FORM>**

Entrez votre texte :

**<BR>**

**<TEXTAREA** name="texte" cols=40 rows=5**>**

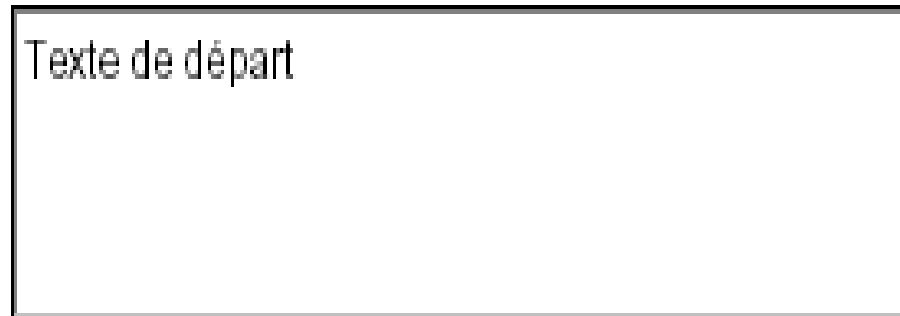
Texte de départ

**</TEXTAREA>**

**</FORM>**

Entrez votre texte :

Texte de départ

A screenshot of a web browser window displaying a form. The form has a title 'Entrez votre texte :'. Below the title is a text area with a black border. Inside the text area, the text 'Texte de départ' is visible at the top. The text area is large enough to show several lines of text, indicating it has scrollbars.

# HTML : Les formulaires

## Exemple 4 crée une case à cocher

### <FORM>

Quels sont vos hobbies :

<BR>

Le sport :

<INPUT type="checkbox" name="hobby1" value="sport" checked>

<BR>

Internet :

<INPUT type="checkbox" name="hobby2" value="internet" checked>

<BR>

Votre site :

<INPUT type="checkbox" name="hobby3" value="site">

<BR>

La lecture :

<INPUT type="checkbox" name="hobby4" value="lecture">

<BR>

La télévision :

<INPUT type="checkbox" name="hobby5" value="tv" checked>

### </FORM>

Quels sont vos hobbies :

Le sport : ☒

Internet : ☒

Votre site : ☐

La lecture : ☐

La télévision : ☒



# HTML : Les formulaires

## Exemple 5: crée un **bouton radio**

### <FORM>

Quel est votre niveau en HTML :

<BR>

<INPUT type="radio" name="niveau" value=3>Excellent

<BR>

<INPUT type="radio" name="niveau" value=2 checked>Moyen

<BR>

<INPUT type="radio" name="niveau" value=1>Débutant

### </FORM>

Quel est votre niveau en HTML :

☐ Excellent

☒ Moyen

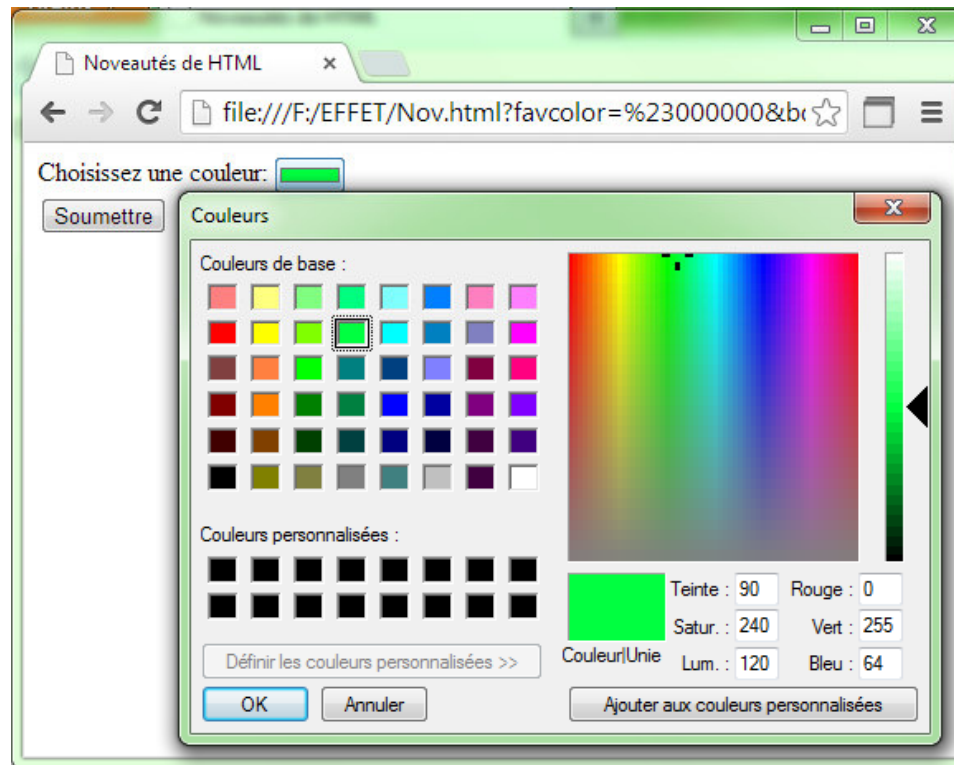
☐ Débutant

# HTML : Les nouveautés de HTML 5

## ❖ Type couleur

Choisir une couleur:

```
<input type="color" name="couleur">
```

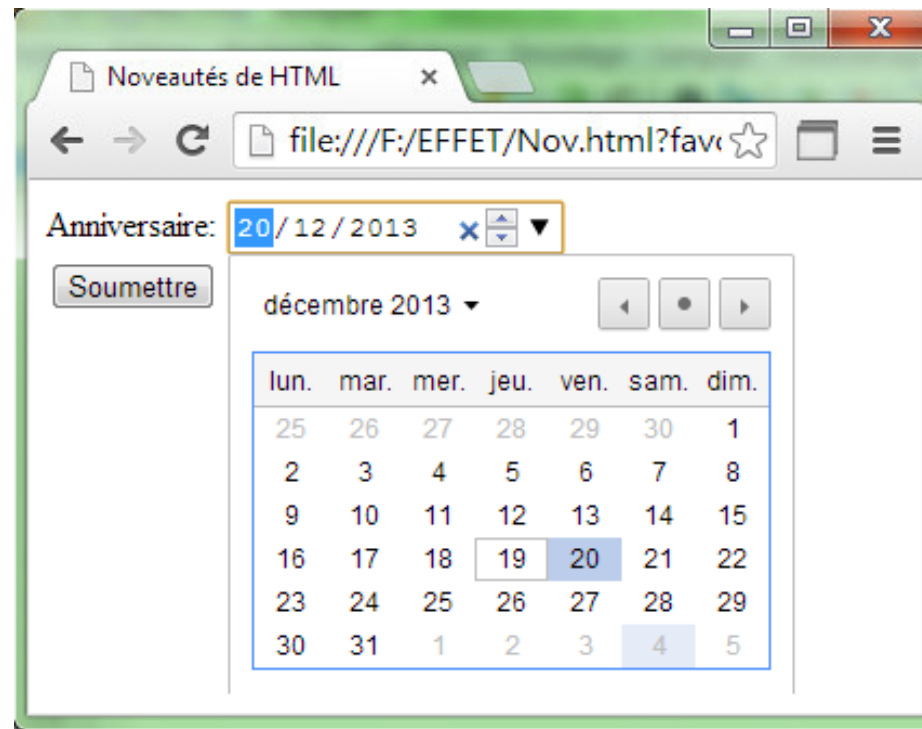


#00FFFF Aqua	#000000 Black	#0000FF Blue	#FF00FF Fuchsia	#808080 Gray	#008000 Green	#00FF00 Lime	#800000 Maroon
#000080 Navy	#808000 Olive	#800080 Purple	#FF0000 Red	#C0C0C0 Silver	#008080 Teal	#FFFFFF White	#FFFF00 Yellow

# HTML : Les nouveautés de HTML 5

## ❖ Type date

Anniversaire:



The screenshot shows a web browser window with the title "Nouveautés de HTML". The address bar displays "file:///F:/EFFET/Nov.html?fav". Below the address bar, there is a form with the label "Anniversaire:" followed by a date input field showing "20/12/2013". To the left of the input field is a "Soumettre" button. Below the input field, a calendar for "décembre 2013" is displayed. The calendar has a header with days of the week (lun., mar., mer., jeu., ven., sam., dim.) and a grid of dates. The date "20" is highlighted in blue.

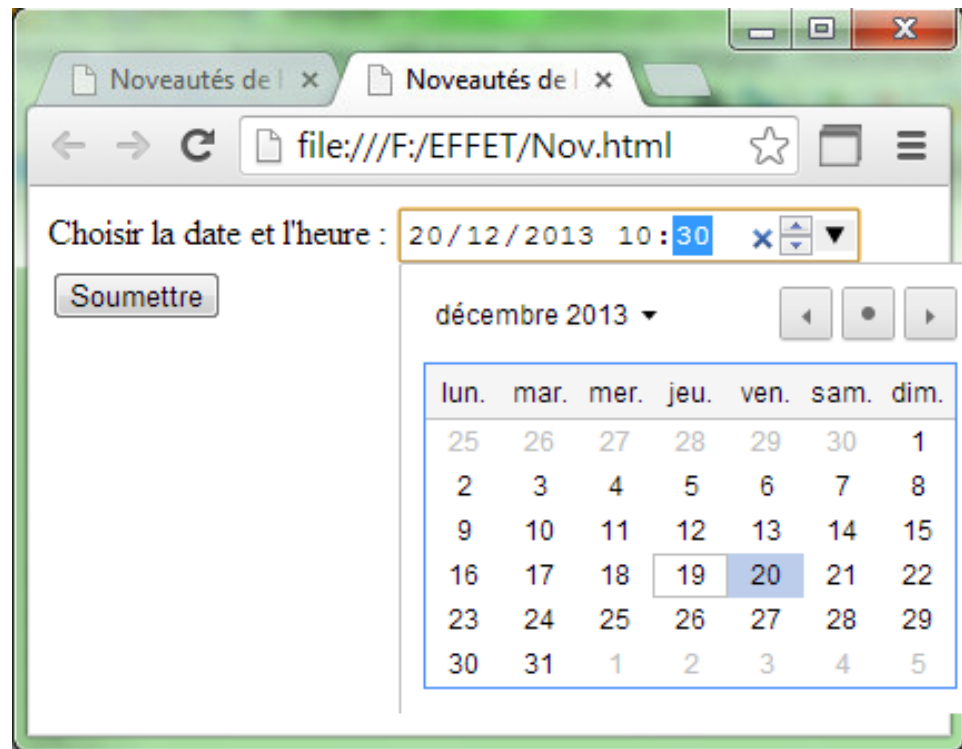
lun.	mar.	mer.	jeu.	ven.	sam.	dim.
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

# HTML : Les nouveautés de HTML 5

## ❖ Type date et heure

Choisir la date et l'heure:

`<input type="datetime-local" name="dateHeure">`

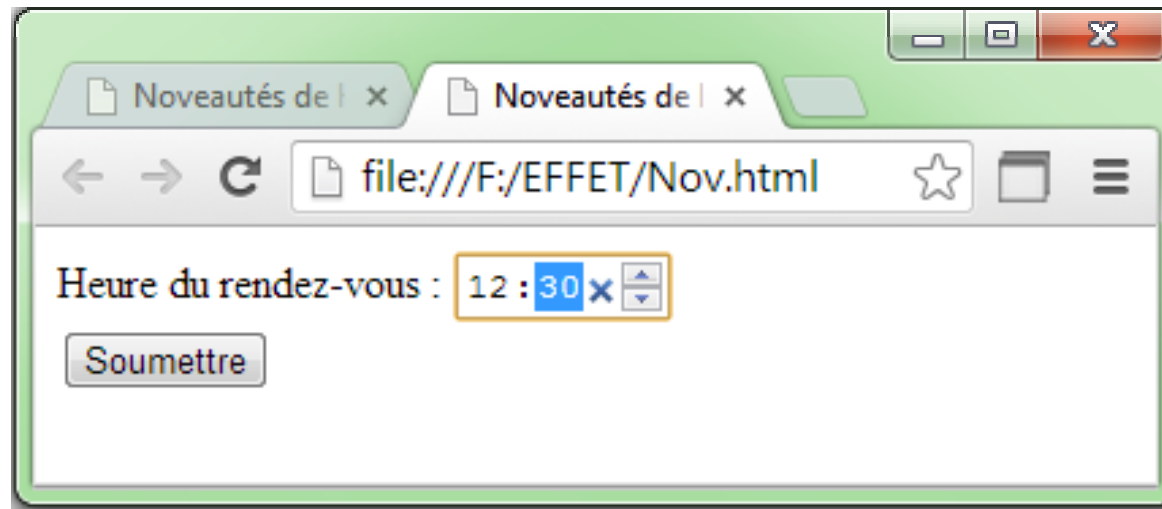


# HTML : Les nouveautés de HTML 5

## ❖ Type heure

Heure du rendez-vous :

```
<input type="time" name="rdv_time" />
```

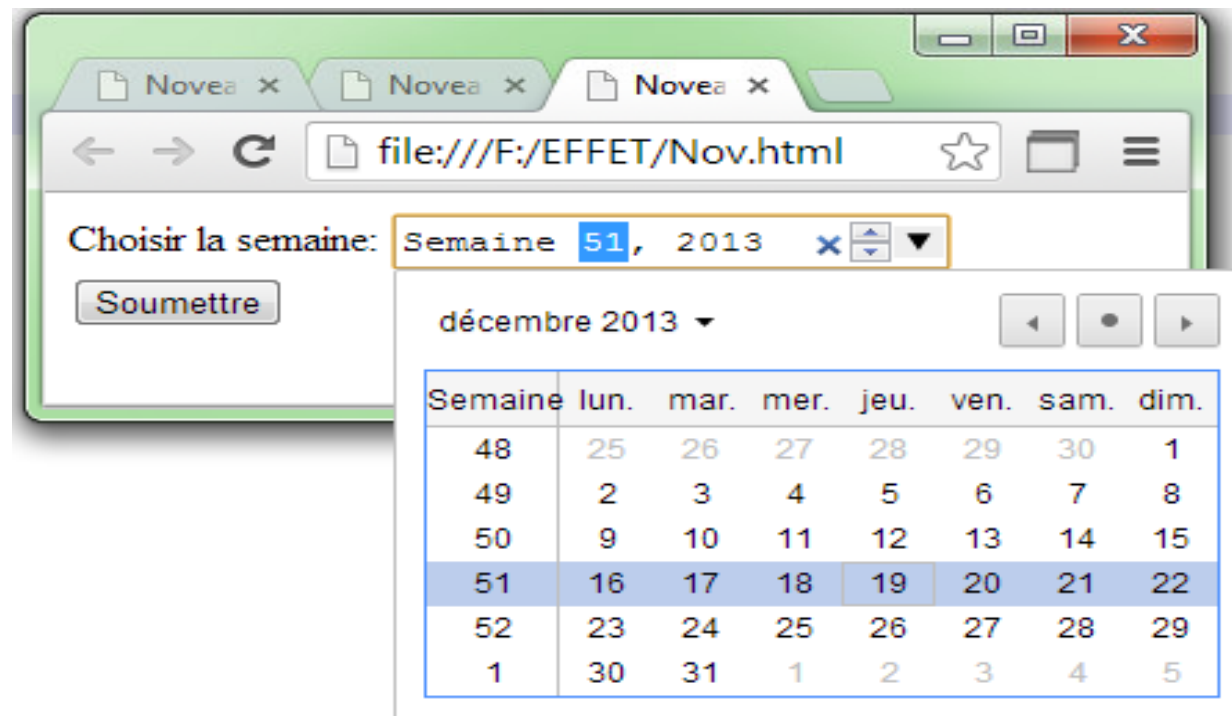


# HTML : Les nouveautés de HTML 5

## ❖ Type semaine

Choisir la semaine :

`<input type="week" name="semaine" >`

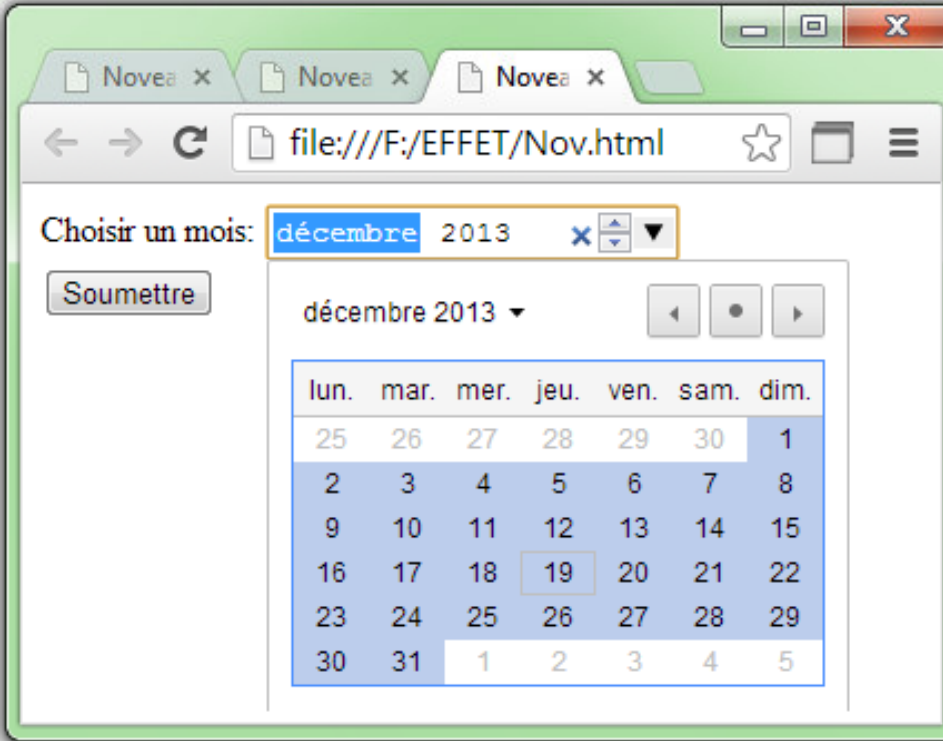


# HTML : Les nouveautés de HTML 5

## ❖ Type mois:

Choisir un mois:

`<input type="month" name="mois" />`



The screenshot shows a web browser window with three tabs labeled 'Novea'. The address bar displays 'file:///F:/EFFET/Nov.html'. The page content includes the text 'Choisir un mois:' followed by a date picker control. The date picker shows 'décembre 2013' with a dropdown arrow. Below this is a 'Soumettre' button. The date picker also displays a calendar grid for December 2013. The grid has columns for days of the week (lun., mar., mer., jeu., ven., sam., dim.) and rows for dates. The date '19' is highlighted with a blue border.

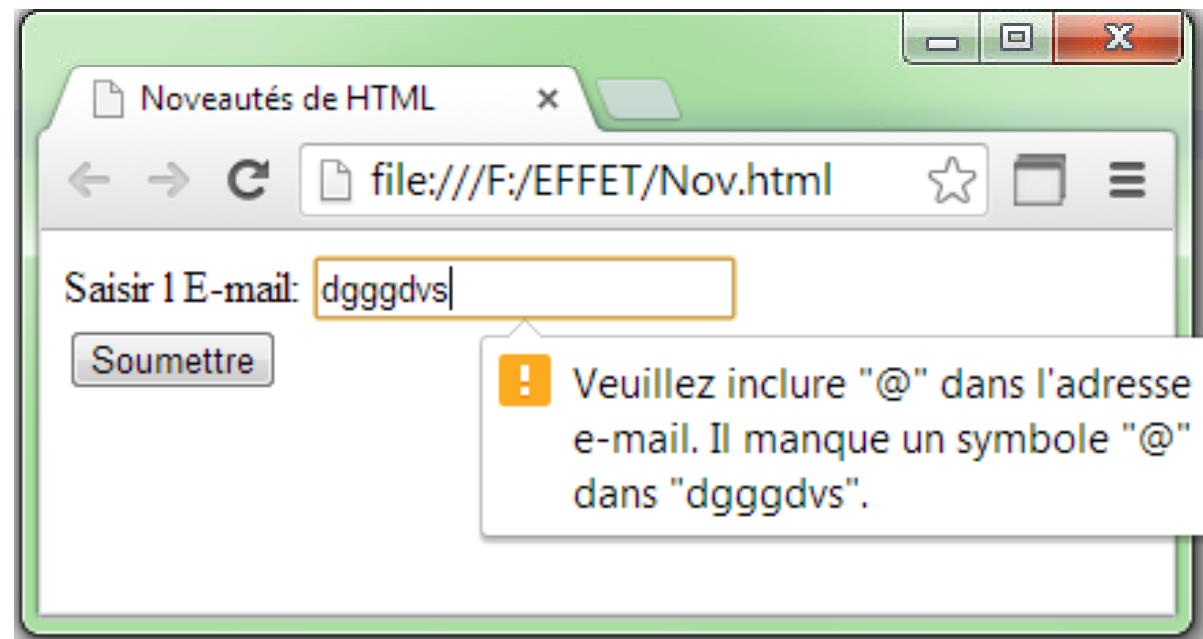
lun.	mar.	mer.	jeu.	ven.	sam.	dim.
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

# HTML : Les nouveautés de HTML 5

## ❖ Type E-mail

Saisir l'E-mail:

```
<input type="email" name="mail" >
```



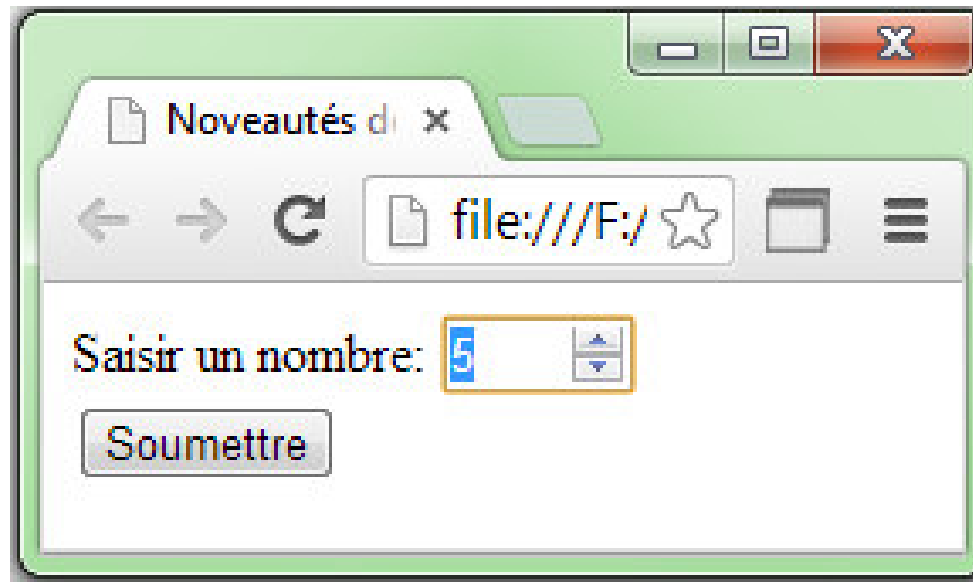


# HTML : Les nouveautés de HTML 5

## ❖ Type nombre

Saisir un nombre:

```
<input type="number" name="nombre" min="1" max="5">
```

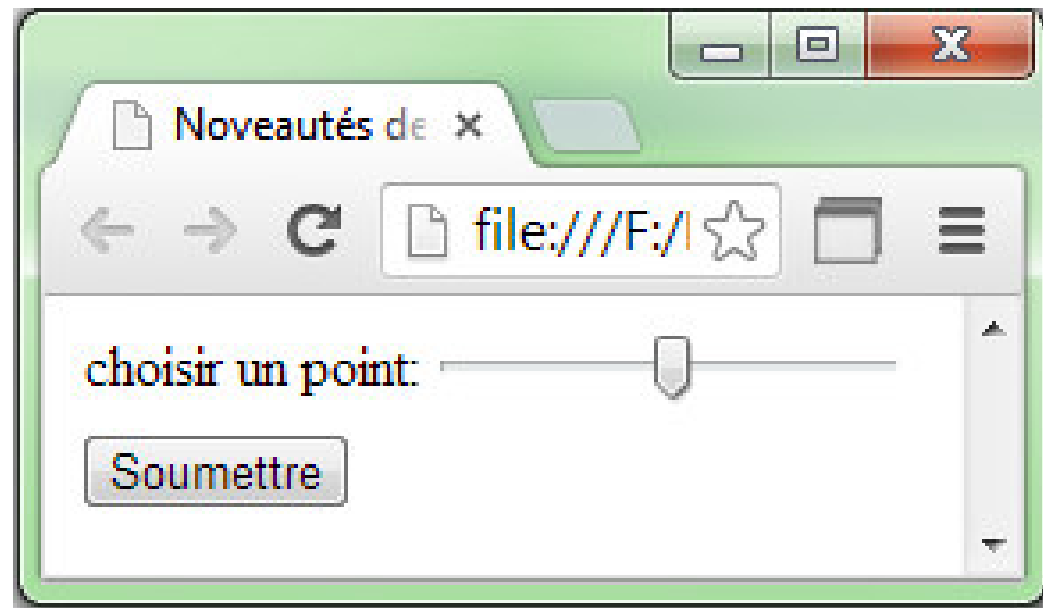


# HTML : Les nouveautés de HTML 5

## ❖ Type nombre

Saisir un nombre:

`<input type="range" name="n" min="1" max="10" >`

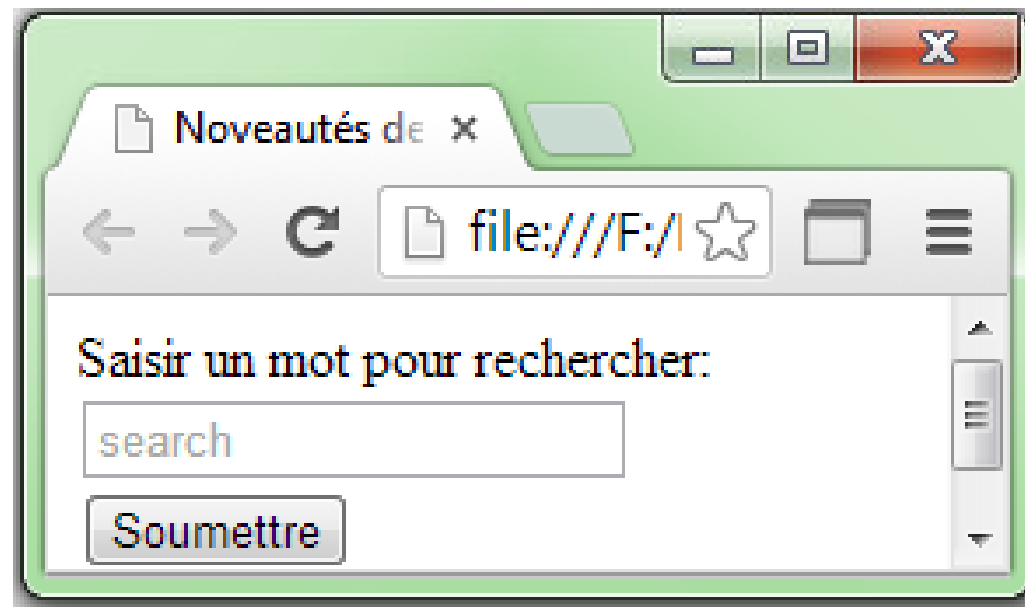


# HTML : Les nouveautés de HTML 5

## ❖ Type recherche

Saisir un mot de recherche:

```
<input id="mysearch2" type="search" placeholder="search">
```



# HTML : Les nouveautés de HTML 5

## ❖ Les nouvelles balises sémantiques

L'HTML5 introduit également un ensemble de nouvelles balises afin de donner plus de **sémantique** (de **sens**) à nos pages;

**<header>** : Qui indique que l'élément est **une en-tête**

**<footer>** : Qui indique que l'élément est **un pied-de-page**

**<nav>** : Qui indique un élément de **navigation tel qu'un menu**

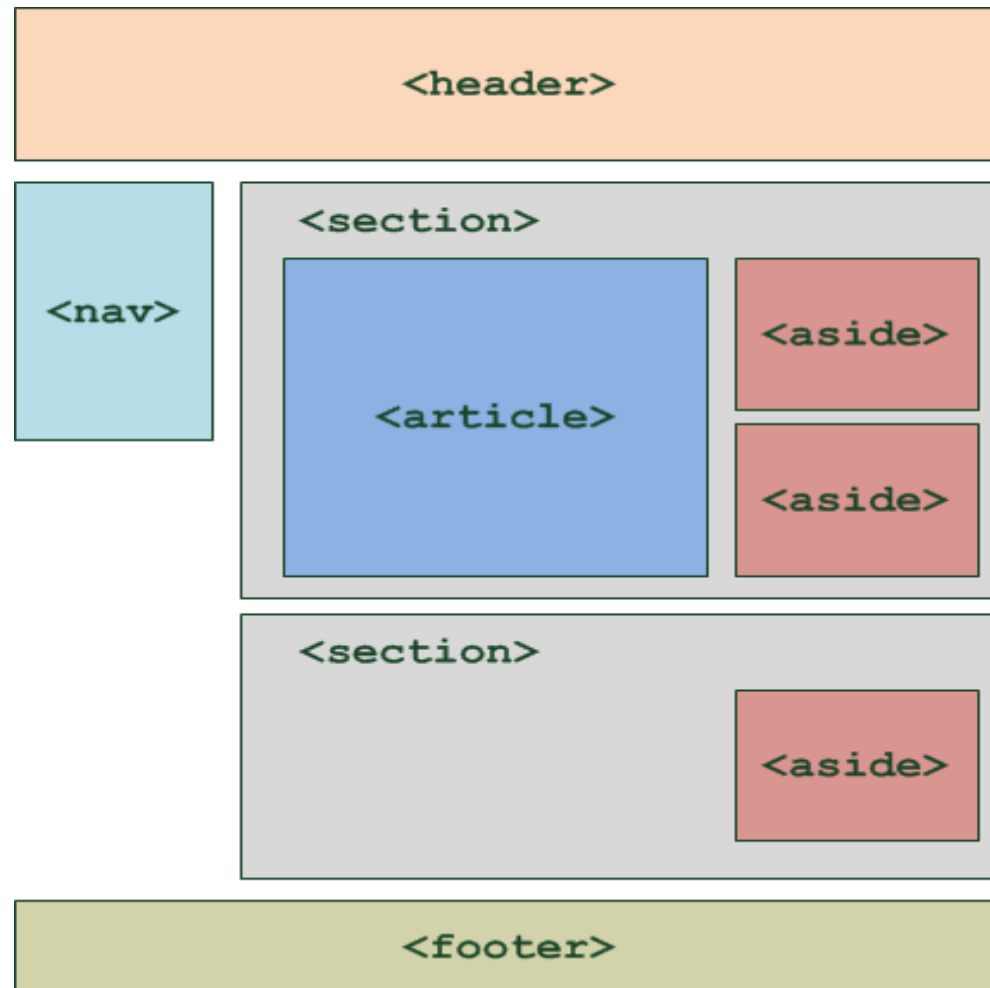
**<section>**: sert à **regrouper des contenus en fonction de leur thématique**. Elle englobe généralement une portion du contenu au centre de la page.

**<article>** : sert à **englober une portion** généralement autonome de la page.

**<aside>** : est conçue pour **contenir des informations complémentaires au document que l'on visualise**.

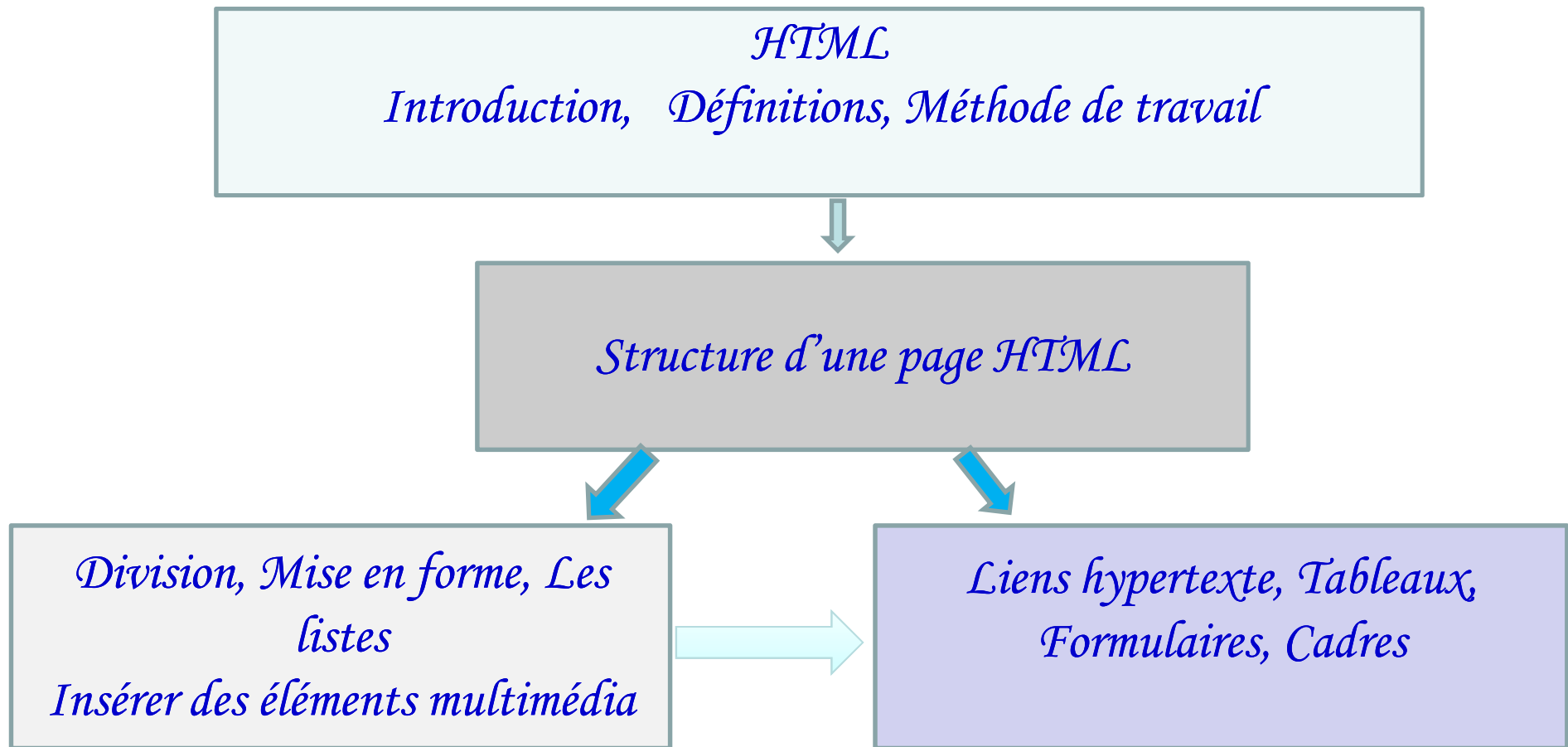
# HTML : Les nouveautés de HTML 5

## ❖ Les nouvelles balises sémantiques



# Conclusion

## A retenir pour ce chapitre:



# *PLAN*

- *Introduction*
- *Langage HTML5*
- *Langage CSS3 (Cascading Style Sheets)*
- *Langage javascript*
- *Langage PHP5*
- *Conclusion*

# *Langage CSS3 (Cascading Style Sheets)*

- ➡ Définitions
- ➡ Structure d'une règle
- ➡ Couleurs et arrière-plan
- ➡ Polices
- ➡ Mise en forme des listes via CSS
- ➡ Ombres
- ➡ Propriétés des boîtes(blocs)
- ➡ Liens
- ➡ Conclusion



# Définitions

- Une feuille de style est un **ensemble de règles graphiques** qui s'appliquent à une ou plusieurs **pages Web**.
- Il est **possible d'utiliser une même feuille de style pour plusieurs pages**.
- Le **CSS** est donc un langage de présentation, permettant de créer un vrai mise en forme:

## ❖ De la page:

- *Organiser* la page HTML en « blocs » : colonnes, bandeaux, couleur/image de fond, etc.

## ❖ Du texte:

- *Encadrer, souligner, taille/couleur du texte, espacement entre lettres, puces des listes, marges...*

## ❖ Des images:

- Définir les *marges* et les *tailles standard des images*

## ❖ Du site web

# Définitions

## 🌐 Comment appliquer CSS à une page HTML?

➤ Il existe **trois méthodes** pour **appliquer le style CSS** à un document HTML:

### ❖ Méthode 1:

Appliquer directement un **style CSS** à un document HTML avec l'attribut HTML « **style** ». Dans ce cas on parle d'un style local.

### ❖ Exemple:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
  </head>

  <body style="background-color: orange;">
    <h1>Un titre de niveau 1</h1>
    <p style="color: blue; font-size: 20px;">Un paragraphe</p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```



# Définitions

🌐 *Comment appliquer CSS à une page HTML?*

## ❖ Méthode 2:

Ajouter une section **<style>** au début du document html placé entre la balise **<head>** et **</head>**: s'applique **aux balises du document courant**

## ❖ Exemple:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Où écrire le CSS ?</title>
    <meta charset= "utf-8">
    <style>
      body{
        background-color: orange;
      }
      p{
        color: blue;
        font-size: 16px;
      }
    </style>
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>
    <p>Un paragraphe</p>
    <p>Un deuxième paragraphe</p>
  </body>
</html>
```

## Un titre de niveau 1

Un paragraphe

Un deuxième paragraphe

# Définitions

● *Comment appliquer CSS à une page HTML?*

## ❖ Méthode 3: C'est la meilleure technique

Elle consiste à appeler une **feuille de style externe** définie ailleurs. On fait l'appel par la balise <Link>

```
<link rel="stylesheet" type="text/css" href="styles/feuille1.css">
```

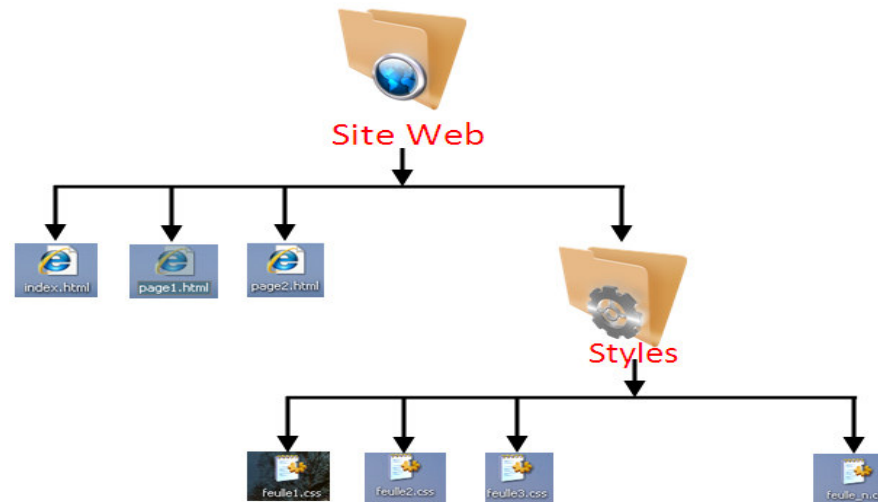
➤ **Une feuille de style externe** est un **fichier texte** ayant l'extension «**.css** ». Elle peut être **rangée** avec le reste **des fichiers du site Web** dans le même **répertoire** ou dans un **répertoire à part réservé** pour accueillir les **feuilles de style définies** (recommandé).

# Définitions

## Comment appliquer CSS à une page HTML?

### ❖ Exemple

- En Supposons maintenant que la feuille de style est rangée dans un répertoire nommé «**styles**». La feuille s'appelle «**feuille1.css**». La structure du site est la suivante:



- Maintenant on crée un lien depuis le document HTML (index.htm) vers la feuille de style (feuille1.css). Ce lien est créé en une ligne de code HTML:

```
< link rel="stylesheet" type="text/css" href="styles/feuille1.css">
```

# Définitions

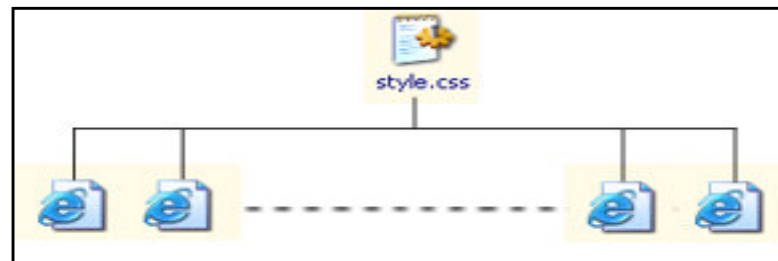
## ● Comment appliquer CSS à une page HTML?

### ❖ **Exemple:**

Maintenant la ligne de code doit s'inscrire dans la section d'entête du code HTML (entre <head> et </head>). Comme ceci:

```
<html>
<head>
  <link rel="stylesheet" type="text/css" href="style/style1.css">
</head>
<body>
</body>
</html>
```

Vous pouvez conclure qu'avec **une seule feuille de style** définie sur votre site Web vous pouvez **contrôler la totalité des pages** en y insérant un lien vers celle-ci.



# Structure d'une règle

## ❖ Syntaxe :



## ❖ Exemple:

**h1 { color: blue; }**

**Sélecteur(s)      Propriété(s)      Valeur(s)**

**h3, h4 {      font-weight:      bold;**  
**font-family :      arial; }**

# Structure d'une règle

➤ Les **sélecteurs** permettent de **définir la cible à laquelle on veut appliquer le style**.

1. Un **sélecteur simple** est constitué du nom de la balise sans les caractères de début < et de fin de balise />.

```
balise { propriété1: valeur1;  propriété2: valeur2; }
```

❖ Exemple :

```
p { color: blue;  background-color: red; }
```

2. Un **sélecteur multiple** porte sur plusieurs sélecteurs simples, séparés par une virgule.

```
balise1, balise2 { propriété1: valeur1;  propriété2: valeur2; }
```

❖ Exemple: plusieurs éléments

```
p, h1 {color: blue;  background-color: yellow; }
```



## Structure d'une règle

3. Un **sélecteur universel** \*: Pour **appliquer un style à tous les éléments**, nous utiliserons le sélecteur universel \* avant la définition d'une ou plusieurs propriétés.

```
* { propriété1: valeur1; propriété2: valeur2; }
```

❖ Exemple :

```
*{ background-color: yellow; }
```

indique que la couleur du fond de tous les éléments soit le jaune.

- Cela n'empêche pas de **modifier cette couleur de fond pour un élément particulier, en la redéfinissant uniquement pour celui-ci**, par exemple :

```
*{ background-color: yellow; }
```

```
P { background-color: gray; }
```

tous les éléments ont un fond **jaune**, sauf **<p>** qui a un fond **gris** redéfini spécialement.

# Structure d'une règle

4. **Les classes**: Pour créer une classe, le sélecteur est constitué du **nom choisi pour la classe** précédé d'un **point (.)**.

➤ Le sélecteur de **classe** est utilisé **pour spécifier un style pour un groupe d'éléments**.

```
■Nom_Classe {  
    propriété1: valeur1;  
    propriété2: valeur2;  
    .....  
    propriétéN:valeurN;  
}
```

❖ **Exemple :**

```
■corps {  
    font-family : verdana, arial;  
    font-style: italic ;  
}
```

❖ *l'utilisation de la classe corps dans html :*

```
<Body class=corps> ..... </body>
```

# Structure d'une règle

5. **Sélecteur d'identifiant id:** Pour les sélecteurs identifiants, la procédure est la même que pour les **classes**(on change le point par dièse(#)).

❖ Le sélecteur d'**ID** est utilisé pour spécifier un style pour un seul élément unique.

```
# Identifiant{  
    propriété1: valeur1;  
    propriété2: valeur2;  
    .....  
    propriétéN:valeurN;  
}
```

❖ **Exemple:**

```
# texte { font-family : arial;  
          font-style: italic ;  
          font-size:20px;  
          }
```

❖ *l'utilisation de l'identifiant texte dans html :*

**<p id=texte>** exemple de sélecteur identifiant appliquer sur un paragraphe**</p>**

# Structure d'une règle

1. Que se passe-t-il si vous associez à la page html le style CSS suivant ?

```
body {  
  font-family: Arial, verdana, sans-serif;  
  color: blue;  
}
```

En conservant la règle

```
body {  
  font-family: Arial, verdana, sans-serif;  
  color: blue;  
}
```

2. Ajoutez une et une seule nouvelle règle de style de manière à ce que le contenu des articles soit affiché en italique et en rouge, comme indiqué sur l'image ci-dessous.

La nature du document à créer, la disposition des paragraphes ne sera pas la même: du texte en colonne pour un journal, du texte aligné à gauche et à droite pour un courrier ou encore du texte centré pour une invitation. A chaque type de document doit correspondre une maquette appropriée.

## Section 1

### Article 1.1

Une fois le document terminé, il faut le relire pour vérifier que tout est clair et qu'il n'y a pas de faute d'orthographe. Certains logiciels possèdent un correcteur orthographique. Cette fonction très pratique signale les fautes de frappe ou les mots inconnus du dictionnaire du logiciel. La dernière étape consiste souvent à imprimer le document pour le publier et ainsi le communiquer à d'autres personnes.

### Article 1.2

Certaines polices de caractères sont fournies par le système d'exploitation, d'autres peuvent être ajoutées par les utilisateurs. Pour cette raison, la liste n'est pas la même d'un ordinateur à l'autre, même sur un logiciel identique.

Les paramètres de réglage d'imprimante sont importants si on ne veut pas utiliser de grandes quantités d'encre. Les cartouches surtout celles en couleurs coûtent cher. Par principe, il vaut mieux régler son imprimante en permanence sur une qualité d'impression "brouillon" et ne paramétrer une impression en qualité normale que pour les documents définitifs, après les avoir relus.

## Section 1

### Article 2.1

Pour faire de belle mise en page, le mieux est de bien observer la maquette des journaux, des magazines et des livres qui sont chez vous ou au lycée. Lorsque vous en trouvez qui vous plaisent, il faut essayer de les imiter. Il est aussi possible d'utiliser les modèles qui sont souvent fournis avec les logiciels de mise en page. .

Page html

# Structure d'une règle

1. Que se passe-t-il si vous associez à cette page le style CSS suivant ?

```
body {  
  font-family: Arial, verdana, sans-serif;  
  color: blue;  
}
```

En conservant la règle

```
body {  
  font-family: Arial, verdana, sans-serif;  
  color: blue;  
}
```

2. Ajoutez une et une seule nouvelle règle de style de manière à ce que le contenu des articles soit affiché en italique et en rouge, comme indiqué sur l'image suivante.

```
article {  
  color: red;  
}
```

La nature du document à créer, la disposition des paragraphes ne sera pas la même: du texte en colonne pour un journal, du texte aligné à gauche et à droite pour un courrier ou encore du texte centré pour une invitation. A chaque type de document doit correspondre une maquette appropriée.

## Section 1

### Article 1.1

Une fois le document terminé, il faut le relire pour vérifier que tout est clair et qu'il n'y a pas de faute d'orthographe. Certains logiciels possèdent un correcteur orthographique. Cette fonction très pratique signale les fautes de frappe ou les mots inconnus du dictionnaire du logiciel. La dernière étape consiste souvent à imprimer le document pour le publier et ainsi le communiquer à d'autres personnes.

### Article 1.2

Certaines polices de caractères sont fournies par le système d'exploitation, d'autre peuvent être ajoutées par les utilisateurs. Pour cette raison, la liste n'est pas la même d'un ordinateur à l'autre, même sur un logiciel identique.

Les paramètres de réglage d'imprimante sont importants si on ne veut pas utiliser de grandes quantités d'encre. Les cartouches surtout celles en couleurs coûtent cher. Par principe, il vaut mieux régler son imprimante en permanence sur une qualité d'impression "brouillon" et ne paramétrer une impression en qualité normale que pour les documents définitifs, après les avoir relus.

## Section 1

### Article 2.1

Pour faire de belle mise en page, le mieux est de bien observer la maquette des journaux, des magazines et des livres qui sont chez vous ou au libraire. Lorsque vous en trouvez un vous

# Couleurs et arrière-plan

## ➤ La couleur d'avant-plan: la propriété *'color'*

La propriété *'color'* décrit la couleur d'avant-plan d'un élément.

Supposons que nous voulons visualiser les grands titres en rouge. Nous savons jusqu'ici que les grands titres sont balisées en HTML avec l'élément **<h1>**. Le code suivant donne aux éléments **<h1>** une couleur rouge:

```
h1 {  
    color:#ff0000;  
}
```

Ou bien:

```
h1 {  
    color:red;  
}
```

## ➤ L'arrière-plan

La propriété *'background-color'* décrit la couleur d'arrière-plan des éléments.

L'élément **<body>** décrit la page toute entière. Pour changer la couleur d'arrière-plan de la page entière il faut appliquer la propriété *'background-color'* sur l'élément **<body>**. On aura alors:

```
body {  
    background-color:#ffff00  
}
```

Ce qui donne un arrière plan **jaune** pour la page qui utilise ce style.

# Couleurs et arrière-plan

## ➤ L'arrière-plan

- ❖ La propriété '**background-image**' sert à **insérer une image d'arrière-plan**.

Si on veut ajouter à l'exemple précédent, en plus d'arrière-plan jaune, une image d'arrière-plan nommé 'im.jpg' on n'a qu'à mettre le code suivant:

```
body {  
    background-color:#FFFF00;  
    background-image: url("im.jpg");  
}
```

**url** contient le chemin (relatif ou absolu) de l'image à mettre en arrière-plan.

- ❖ La propriété '**background-repeat**' permet de **répéter l'image d'arrière-plan** selon le choix.

**Par défaut, une image d'arrière-plan est répétée sur toute la page (si, bien sûr, les dimensions de l'image sont inférieures à celles de la page).**

# Couleurs et arrière-plan

## ➤ L'arrière-plan

➡ La propriété '*background-repeat*' peut prendre **quatre valeurs**:

*background-repeat: repeat-x* : Répétition **horizontale** (en haut) de l'image

*background-repeat: repeat-y* : Répétition **verticale** (à gauche) de l'image

*background-repeat: repeat* : Répétition **partout** de l'image

*background-repeat: no-repeat* : L'image **ne se répète pas**

L'exemple précédent redevient:

```
body {  
    background-color: #FFFF00;  
    background-image: url("im.jpg");  
    background-repeat: no-repeat;  
}
```



# Couleurs et arrière-plan

## ➤ L' arrière-plan

- ✦ La propriété '*background-attachment*' permet de **fixer** ou de laisser **défiler** l'image d'arrière-plan.

Par défaut, une image d'arrière-plan défile avec le contenu.

La propriété '*background-attachment*' possède 2 valeurs:

*background-attachment: scroll* : L'image d'arrière-plan peut **défiler**

*background-attachment: fixed* : L'image d'arrière-plan est **fixée**

L'exemple précédent redevient:

```
body {  
    background-color: #FFFF00;  
    background-image: url("im.jpg");  
    background-repeat: no-repeat;  
    background-attachment: fixed;  
}
```

# Couleurs et arrière-plan

## ➤ L'arrière-plan

- La propriété '**background-position**' définit l'emplacement de l'image d'arrière-plan par rapport à la page.

Par défaut, **une image d'arrière-plan est située en haut à gauche.**

- La propriété '**background-position**' possède comme valeur **les coordonnées (X,Y)** de l'emplacement de l'image d'arrière-plan.

### Exemple

**background-position: right top** : L'image est située en haut à droite.

**background-position: 4cm 2cm** : L'image est située à 4cm de la gauche et 2cm du haut de la page.

**background-position: 50% 50%** : L'image est centrée sur la page.

❖ les valeurs peuvent être exprimées en **pourcentage**, en **cm**, en **px** (pixels) ou encore en utilisant les mots-clés '**top**', '**bottom**', '**center**', '**left**' ou '**right**'.

# Couleurs et arrière-plan

## ➤ L'arrière-plan

- ❖ La propriété '**background**' est un raccourci pour toutes les propriétés énumérée précédemment. En effet, au lieu d'exprimer l'élément <body> de la façon:

```
body { background-color: #FFFF00;  
        background-image: url("im.jpg");  
        background-repeat: no-repeat;  
        background-attachment: fixed;  
        background-position: 50% 50%  
}
```

On peut raccourcir la syntaxe et on aura:

```
body {  
    background: #FFFF00 url("im.jpg") no-repeat fixed 50% 50%;  
}
```

# Polices

## ➤ Les propriétés de polices

❖ La propriété '**font-family**' permet de **lister un ensemble de polices** que le navigateur prendra en compte selon leur ordre d'énumération.

➡ Deux types de noms pour catégoriser les polices:

❖ **Le nom de famille**: appelé aussi police. Comme « **Arial** », « **Time New Roman** », « **Verdana** »...

❖ **La famille générique**: Elle regroupe les polices ayant des aspects uniformes. Par exemple on peut trouver la famille « **Sans sérif** » caractérisées par **l'absence d'empattements**, la famille « **Sérif** » qui **procèdent des empattements** et la famille « **Monospace** » caractérisées par **un espacement régulier** entre les caractères.

# Polices

## ➤ Les propriétés de polices

Exemple de la propriété '*font-family*' :

*h1 {font-family: verdana, arial, sans-serif;}*

*h2 {font-family: "Time New Roman", Georgia, serif;}*

*Notez que le nom de la police Time New Roman contient des espaces. Elle est alors mise entre des guillemets.*

- ❖ La propriété '*font-style*' permet de définir **le style de la police** qui peut être **normal** ou **italic**.

*h1 {  
font-family: verdana;  
font-style: italic;  
}*

tous les titres **<h1>** en **italic** et la police en **verdana**:

# Polices

## ➤ Les propriétés de polices

- ❖ La propriété '*font-variant*' permet de définir la **variante de la police** qui peut prendre la valeur **normal** ou **small-caps**.

Une police **small-caps** utilise les **lettres majuscule mais avec une taille réduite à la place des lettres en minuscules**.

L'exemple suivant illustre le mot « bonjour » en petites majuscules (small-caps) et en majuscule:

bonjour (en police normale)

BONJOUR (en police small-caps ou petites majuscules)

BONJOUR (en police majuscule)

*h1 { font-variant: small-caps; }*

Tous les titres <h1> seront mis en petites majuscules.

# Polices

## ➤ Les propriétés de polices

- ❖ La propriété '*font-weight*' définit le **degré de graisse** avec lequel présenter les polices. Il prends aussi deux valeurs: **normal** ou **bold**.

L'exemple suivant va **mettre toutes les polices placées dans les colonnes des tableaux décrits par <td> en gras.**

```
td { font-weight: bold; }
```

Cette ligne CSS permet carrément de **mettre tous les textes de la page en gras:**

```
body { font-weight: bold; }
```

- ❖ La propriété '*font-size*' permet de **définir la taille des polices**. La taille peut être exprimée par « **px** », « **pt** » ou encore par « **%** ».

```
h1 { font-size: 32px; }
```

```
h2 { font-size: 16pt; }
```

*px: pixel (unité de base pour mesurer les dimensions d'une image)*

*pt: point (il vaut à peu près 1/72 pouce ou encore 1/28 cm)*

# Polices

## ➤ Les propriétés de polices

- ❖ La propriété « **font** » permet de raccourcir les lignes CSS concernant la mise en forme des polices de la page. La syntaxe suivante:

```
h1 {  
    font-family: verdana;  
    font-size: 32pt;  
    font-weight: bold;  
    font-variant: small-caps;  
    font-style: italic;  
}
```

*Se réduit à:*

```
h1 {font:32pt bold small-caps italic verdana arial sans-serif;}
```



# Mise en forme des listes via CSS

➤ En HTML, on a appris les différents types de liste et les différentes balises pour les définir.

1- Listes non ordonnées

2- Listes ordonnées

3- Listes descriptives

## 1- Listes non ordonnées (à puces)

❖ Les listes à puces sont des listes non numérotées(à puces). En CSS il est possible d'avoir différentes **formes de puces** grâce à la propriété **list-style-type**. Les formes varient en fonction de la valeur de la propriété.

❖ list-style-type: **disc** /\* petit cercle plein \*/

❖ list-style-type: **circle** /\* petit cercle vide \*/

❖ list-style-type: **square** /\* petit carré plein \*/

# Mise en forme des listes via CSS

## Exemple

### Code CSS

```
.disc {  
list-style-type:disc  
}  
.circle  
{ list-style-type:circle;  
}  
.square {  
list-style-type:square;  
}
```

### Code HTML

```
<ul class="disc" >  
  <li>liste 1 avec disc</li>  
  <li>liste 2 avec disc</li>  
</ul>  
<ul class="circle" >  
  <li>liste 1 avec circle</li>  
  <li>liste 2 avec circle</li>  
</ul>  
<ul class="square" >  
  <li>liste 1 avec square</li>  
  <li>liste 2 avec square</li>  
</ul>
```

## 2- Listes ordonnées

Pour faire des listes numérotées et alphabétiques il suffit de remplacer `<ul>` par `<ol>` en HTML. Toujours avec la propriété **list-style-type** on peut mettre des types de chiffre ou de caractère de notre choix comme des :

# Mise en forme des listes via CSS

- ❖ **decimal** : affiche les puces sous forme de **numéros décimaux**
- ❖ **decimal-leading-zero** : Numéros de **deux décimales** (01,02, etc...)
- ❖ **lower-roman** : Chiffres **romains minuscules**
- ❖ **upper-roman** : Chiffres **romains majuscules**
- ❖ **lower-latin** : lettres minuscules (forme alphabétique)
- ❖ **upper-latin** : lettres majuscules (forme alphabétique)
- ❖ **lower-greek** : lettres grecques (forme alphabétique)

## Exemple

### Code CSS

```
.Rom_min {  
  list-style-type: lower-roman  
}  
.Rom_maj{  
  list-style-type: upper-roman;  
}
```

### Code HTML

```
<ol class="Rom_min">  
  <li>item1 avec chiffre romain en minuscules  
  <li>item 2 avec chiffre romain en minuscules  
</ol>  
<ol class="Rom_maj">  
  <li>item1 avec chiffre romain en majuscules  
  <li>item 2 avec chiffre romain en majuscule  
</ol>
```

# Mise en forme des listes via CSS

## 3- Listes avec images

A la place des **puces** et des **numéros**, il est aussi **possible de mettre des images**. Pour cela, utilisez la propriété **list-style-image** en mettant l'URL de l'accès local. Au cas où vous voulez désactiver l'affichage de l'image mettez none à la place de l'URL.

### Exemple

### Code CSS

### Code HTML

```
.listeimage1 {  
list-style-image:  
url("images/liste.gif");  
}
```

```
<ul class="listeimage1">  
  <li>liste 1 avec image en locale  
  <li>liste 2 avec image en locale</li>  
</ul>
```

Le chemin de l'image peut aussi contenir des URL comme dans cet exemple :

```
.listeimage2 {  
list-style-image :  
url("http://www.votredomain.co  
m/images/liste.gif" );  
}
```

```
<ul class="listeimage2">  
  <li>liste 1 avec image en ligne</li>  
  <li>liste 2 avec image en ligne</li>  
</ul>
```

# Mise en forme des listes via CSS

## 4- Position de la liste

***list-style-position*** est la propriété pour **spécifier l'emplacement de la liste par rapport à la bordure et au texte**. Il y a deux valeurs possibles :

- ❖ **outside**: qui est la **valeur par défaut**, ces caractères sont intégrés au texte de l'item.
- ❖ **inside**: les caractères de la numérotation sont placés dans **la marge de retrait** du texte de l'item.

### Exemple

### Code CSS

```
.bordure {border: medium double red;}  
.ins { font-size:36px;  
       list-style-type: circle ;  
       list-style-position: inside;}  
.out { font-size:36px;  
       list-style-type: circle;  
       list-style-position: outside;}
```

### Code HTML

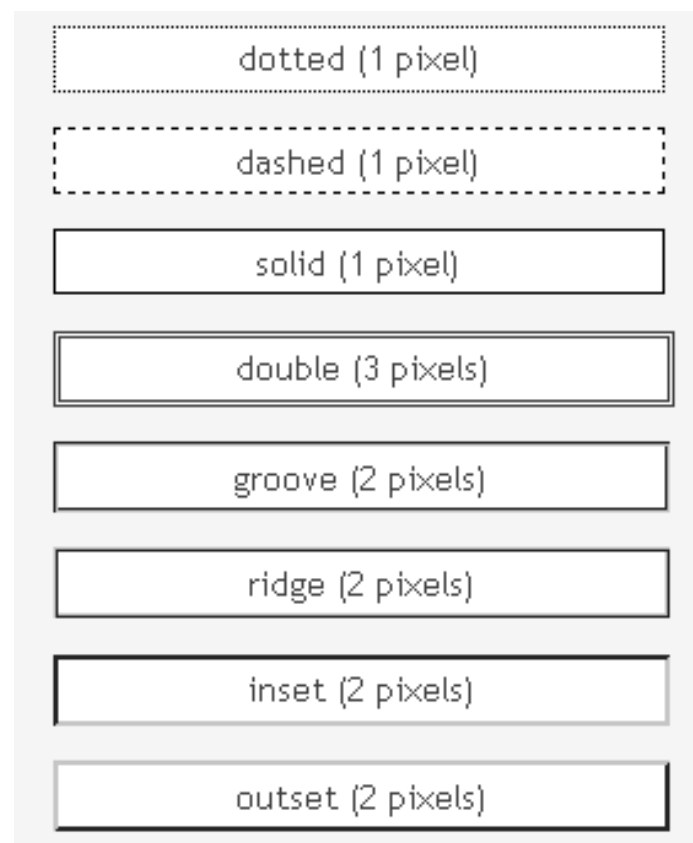
```
<div class="bordure">  
  <ul class="ins" >  
    <li>liste 1 inside  
    <li>liste 2 inside</ul>  
  <ul class="out" >  
    <li>liste 1 outside  
    <li>liste 2 outside  
  </ul> </div>
```

Observez dans l'exemple suivant que les listes avec ***list-style-position: inside***; sont un peu plus décalées de la bordure gauche du div que les listes avec ***list-style-position: outside*** ;

- liste 1 inside
- liste 2 inside
- liste 1 outside
- liste 2 outside

# Bordures

- ❖ ***border-width***: permet de définir la **largeur de la bordure**
- ❖ ***border-color***: permet de définir la **couleur de la bordure**
- ❖ ***border-style***: permet de définir le **type de la bordure**. Les différentes valeurs de cet attribut sont:
  - ➡ **none**: pas de bordure (par défaut)
  - ➡ **solid**: un trait simple.
  - ➡ **dotted**: pointillés.
  - ➡ **dashed**: tirets.
  - ➡ **double**: bordure double.
  - ➡ **groove**: en relief.
  - ➡ **ridge**: autre effet relief.
  - ➡ **inset**: effet 3D intérieur.
  - ➡ **outset**: effet 3D extérieur.
- ❖ ***border*** permet de regrouper les trois valeurs pour modifier l'apparence de la bordure :



# Bordures

## Exemple

### Code HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Cours HTML et CSS</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="cours.css">
  </head>
  <body>
    <h1>Titre de niveau 1</h1>
    <p class="p1">Solid</p>
    <p class="p2">Double</p>
    <p class="p3">Ridge</p>
    <p class="p4">Outset</p>
  </body>
</html>
```

### Cours.css

```
p{
  padding: 20px;
  margin: 20px 10px;
}

.p1{
  border-width: 5px;
  border-style: solid;
  border-color : #09C;
}

.p2{
  border: 15px double #09C;
}

.p3{
  background-color: #0C9;
  border: 5px ridge #09C;
}

.p4{
  background-color: #0C9;
  border: 5px outset #09C;
}
```

### Résultat

#### Un titre de niveau 1

Solid

Double

Ridge

Outset

# Ombres

➤ Les ombres font partie des nouveautés récentes proposées par CSS3. **deux types d'ombres :**

➤ Les ombres des **boîtes** ;

➤ Les ombres du **texte**.

❖ *les ombres des boîtes : box-shadow*

La propriété **box-shadow** s'applique à tout le bloc et prend quatre valeurs dans l'ordre suivant :

➤ Le décalage **horizontal** de l'ombre ;

➤ Le décalage **vertical** de l'ombre ;

➤ L'adoucissement du **dégradé** ;

➤ La **couleur** de l'ombre.

## Exemple

```
p { box-shadow: 6px 6px 0px black; }
```



# Ombres

## ❖ les ombres du texte : **text-shadow**

Avec **text-shadow**, vous pouvez **ajouter une ombre directement sur les lettres de votre texte** ! Les valeurs fonctionnent exactement de la même façon que **box-shadow** ::

- ➡ le décalage **horizontal** de l'ombre ;
- ➡ le décalage **vertical** de l'ombre ;
- ➡ l'adoucissement du **dégradé** ;
- ➡ la **couleur** de l'ombre.

### Exemple

```
p { text-shadow: 2px 2px 4px black; }
```

# DIV et SPAN

➤ Deux nouvelles balises:

## ❖ balise **div**

La balise `<div>` c'est une balise de type "**bloc**" n'est utilisée que **pour diviser la page**. Par exemple, sur un site, un bloc marqué par la balise `<div>` contient le **menu**, un autre contient **l'espace de login**, encore un autre contient un **cours**, etc...

## ❖ Exemple

`<div class=forme>`

`<p>`Je suis dans un **div** dont les bordures sont continues.`<p>`

`</div>`

`<p>`Je suis une phrase hors d'un **div**.`</p>`

`<div class=forme1>`

`<p>`Je suis une autre phrase dans un autre **div** avec des bordures en pointillées.`</p>`

`</div>`

**.forme** {

text-align: center;

border: solid 2px blue;

padding: 5px;

margin: 5px;

}

**.forme1** {

text-align: center;

border: dotted 2px blue;

padding: 5px;

margin: 5px;

}

Je suis dans un div dont les bordures sont continues.

Je suis une phrase hors d'un div.

Je suis une autre phrase dans un autre div avec des bordures en pointillées.

# DIV et SPAN

➤ Deux nouvelles balises:

## ❖ balise **span** (inline)

L'élément **span** permet de **découper une partie de ligne**. Balise de type ligne

**Important : un élément de type "inline" doit toujours être placé à l'intérieur d'un élément de type "bloc" (block). Il ne doit pas être directement encadré par la balise <div> mais par n'importe quelle autre balise de type "bloc".**

## ❖ Exemple

```
.form{  
    font-style: italic;  
    font-weight: bold;  
    color: red;  
}
```

*Je suis dans un div* dont les bordures sont continues.

Je suis une phrase hors d'un div.

Je suis une autre phrase dans un autre *div avec des bordures en pointillées.*

# DIV et SPAN

## ❖ Exemple

```
<!DOCTYPE html>
<html>
  <head>
    <title>Cours HTML et CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="cours.css">
  </head>

  <body>
    <h1>Un titre de niveau 1</h1>

    <div>
      <p>Un premier paragraphe</p>
      <p>Un autre paragraphe</p>

      <ul>
        <li>Un élément de liste</li>
        <li>Un <span>autre élément</span> de liste</li>
      </ul>
    </div>

    <p>Un <span>troisième</span> paragraphe</p>

    <div>
      <p>Un dernier paragraphe</p>
    </div>
  </body>
</html>
```

```
/*Couleur de fond du body (donc de la partie visible de la page) : bleu clair*/
body{
  background-color: lightBlue;
}

div{
  color: white; /*Textes des éléments dans les div blancs par héritage*/
  background-color: purple; /*Les div auront un fond violet*/
  width: 80%; /*Définit la largeur des div à 80% de leur parent (body ici)*/
  margin: 0 auto; /*Permet de centrer les div dans leur élément parent (body ici)*/
}

span{
  font-weight: bold; /*Les textes seront en gras*/
  background-color: yellow; /*Fond des span jaune*/
  color: black; /*Couleur du texte noire*/
}
```

## Un titre de niveau 1

Un premier paragraphe

Un autre paragraphe

- Un élément de liste
- Un autre élément de liste

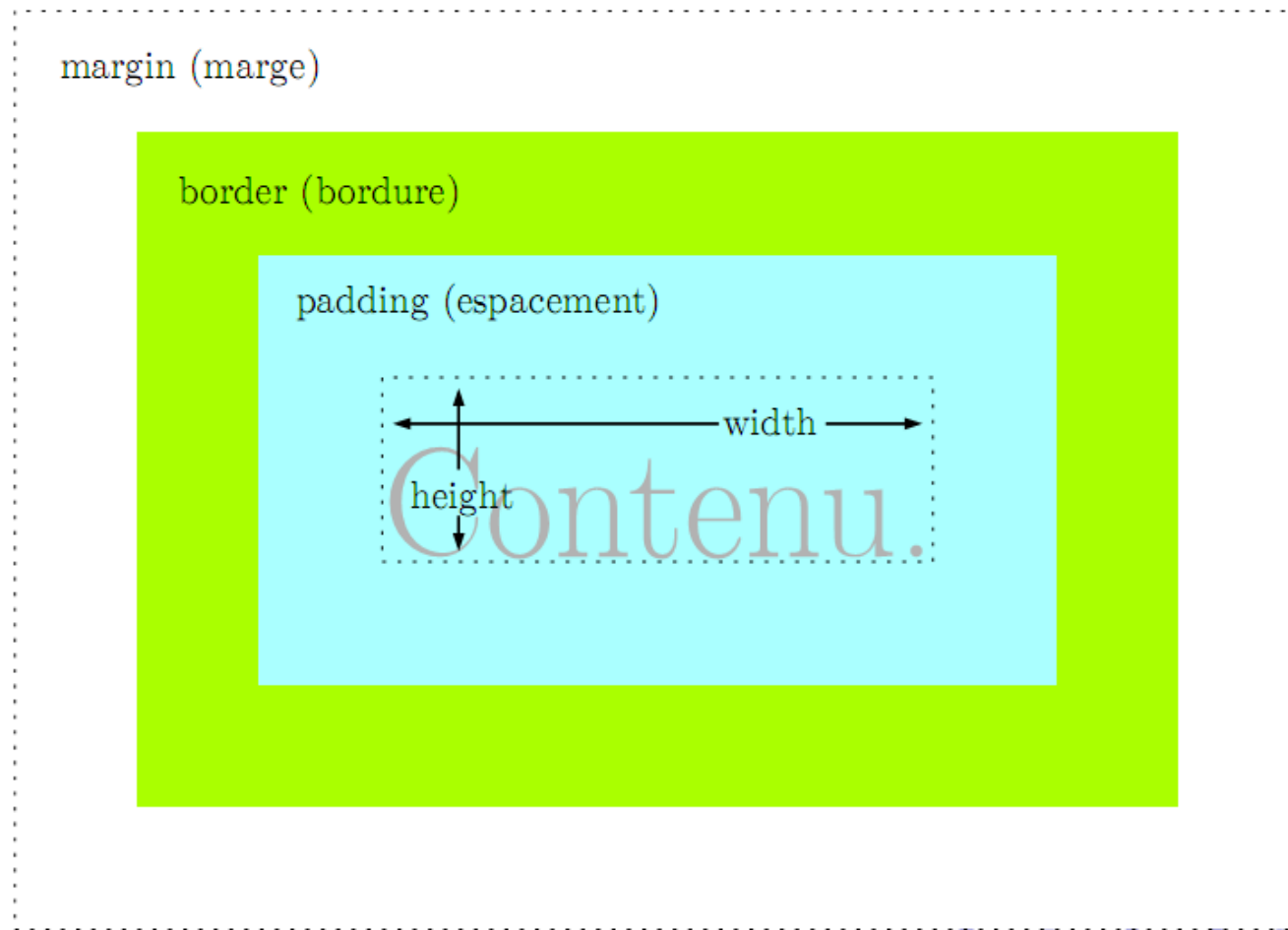
Un troisième paragraphe

Un dernier paragraphe

# Propriétés des boîtes(blocs)

## ❖ Boîte(Bloc)

Les boîtes CSS sont des **blocs de base** pour la construction des pages web



# Propriétés des boîtes(blocs)

## ➤ Largeur, Hauteur et Débordement

- **width** : Largeur du **contenu** de l'élément : **longueur** ou **auto**. Par défaut à auto, prend la taille maximale disponible.
- **height** : Hauteur du contenu de l'élément : **longueur** ou **auto**. Par défaut à auto, s'adapte à la taille du contenu.
- La propriété **overflow** contrôle le comportement :
  - ❖ **visible** : le contenu débordant est **affiché**.
  - ❖ **hidden** : le contenu débordant est caché et donc **illisible**..
  - ❖ **scroll** : des barres de **défilement horizontales et verticales apparaissent** sur les côtés droit et bas de la boîte de l'élément, ce qui permet d'accéder au contenu débordant.
  - ❖ **auto** : le navigateur fait apparaître les barres de *défilement* en cas de débordement uniquement.
  - ❖ Par défaut : **visible**. Internet Explorer 6 ne gère pas **overflow: visible**;

# Propriétés des boîtes(blocs)

## ➡ Marge et Espacement

➤ nous pouvons définir une **marge** autour de chaque élément.

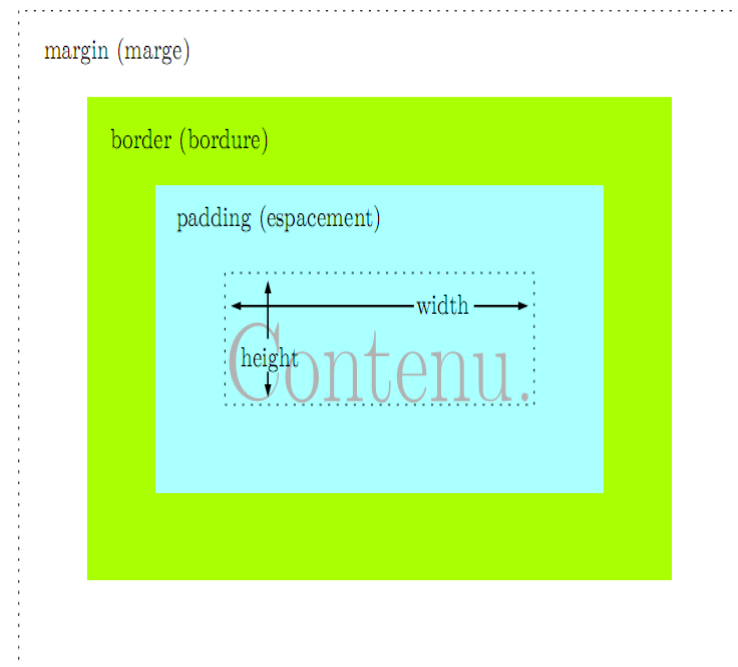
❖ **margin-top** : définit la **marge haute** ;

❖ **margin-right** : définit la **marge droite** ;

❖ **margin-bottom** : définit la **marge basse** ;

❖ **margin-left** : définit la **marge gauche**.

❖ **margin** permet de définir d'un seul coup ces quatre propriétés, dans l'ordre top right bottom left.



# Propriétés des boîtes(blocs)

## ➡ Marge et Espacement

➤ nous pouvons définir une **zone située** entre la **boîte de contenu** d'un élément et sa **bordure**:

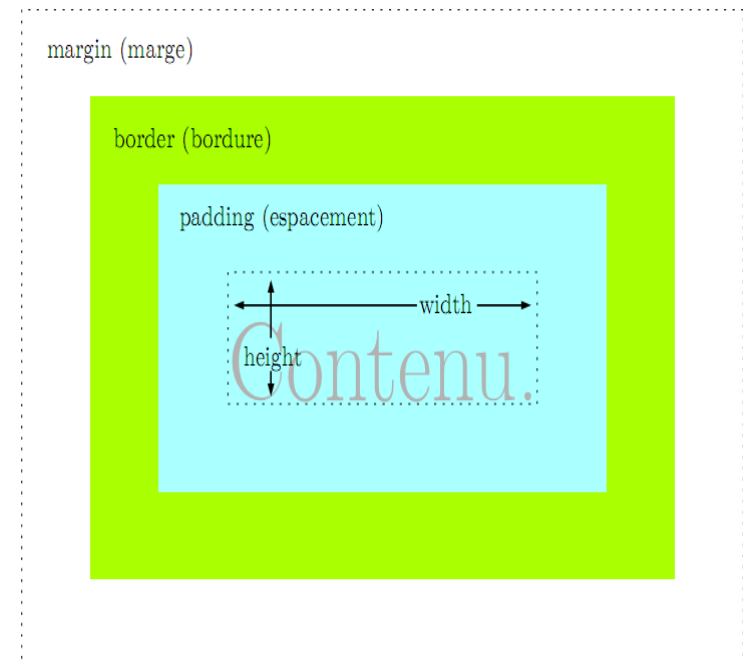
❖ **padding-top** : définit l'espacement **haut** ;

❖ **padding-right** : définit l'espacement **droit** ;

❖ **padding-bottom** : définit l'espacement **bas** ;

❖ **padding-left** : définit l'espacement **gauche**.

❖ **padding** permet de définir d'un seul coup ces quatre propriétés, dans l'ordre top right bottom left.

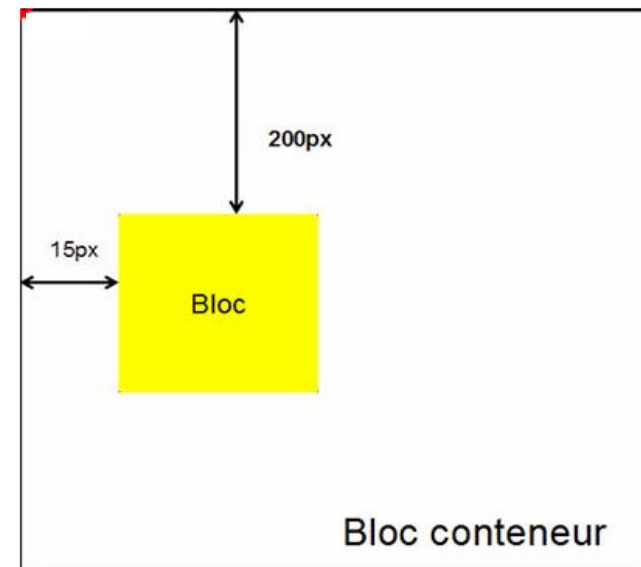




# Propriétés des boites(blocs)

## ► Marge et Espacement

Exemple 1: créer un bloc jaune de 100x100px à 15px à partir de la gauche et 200px à partie du haut du conteneur (voir la figure ci-dessous) :



Code.html

```
<HTML>
<head>
  <link rel="stylesheet" type="text/css" href="ex1.css">
</head>
<BODY>
<div class="conteneur">
<div class="bloc">
Bloc
</div>
</div>
</BODY>
</HTML>
```

Ex1.css

```
.conteneur {
padding-top: 200px;
padding-left: 15px;
}
.bloc {
width: 100px;
height: 100px;
background-color: yellow;
}
```

# Propriétés des boites(blocs)

## ➡ Marge et Espacement

Exemple 2: créer un bloc conteneur qui contiendra tous les blocs. On y place un bloc A, appelé sous classe normal. Puis on place un second bloc, le bloc B à la suite dans une autre balise div. (voir la figure ci-dessous) :

```
<HTML>
<head>
  <link rel="stylesheet" type="text/css" href="ex1.css" />
</head>
<BODY>
  <div class="conteneur">
    <div class="normalA">
      Bloc A
    </div>
    <div class="normalB">
      Bloc B
    </div>
  </div>
</BODY>
</HTML>
```

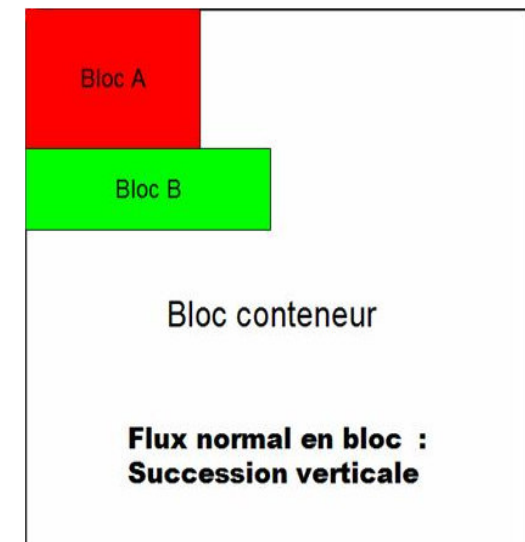
Code.html

Ex1.css

```
.conteneur {
  padding-top: 200px;
  padding-left: 15px;
}

.normalA {
  width:150px;
  height:150px;
  background-color:red;
  border:1px solid black;
}

.normalB {
  width:250px ;
  height:100px ;
  background-color:green;
  border:1px solid black;
}
```



# Propriétés des boites(blocs)

## ➤ Flottement et Positionnement

➤ la propriété **position** contrôle le positionnement

❖ **position: statique** ;(par défaut) les boites se placent **les unes après les autres**;

❖ **position: relative**;

◆ **left** : décale l'élément vers la **droite** (si sa valeur > 0) ou vers la **gauche** (si sa valeur < 0) ;

◆ **top** : décale l'élément vers le **bas** (si sa valeur > 0) ou vers le **haut** (si sa valeur < 0) ;

◆ **Right** : décale l'élément vers la **gauche** (si sa valeur est **positive**) ou vers la **droite** (si sa valeur < 0)

◆ **bottom** : décale l'élément vers le **haut** (si sa valeur est **positive**) ou vers le **bas** (si sa valeur > 0).

# Propriétés des boites(blocs)

## ➡ Flottement et Positionnement

➤ la propriété `position` contrôle le **positionnement**

❖ `position: absolute;`

Définir la **position de l'élément par rapport à son conteneur** à l'aide des propriétés **left, top, right, bottom**, qui définissent la position des bords de l'élément respectivement par rapport aux bords gauche, haut, droit et bas du conteneur.



# Propriétés des boites(blocs)

## ➡ Flottement et Positionnement

➤ la propriété **float** contrôle le **flottement**

**float** permet de **sortir un élément du flux normal de la page** et de le faire “flotter”

- ❖ L'élément flottant se met le plus à droite (**float: right**) ou à gauche (**float: left**) possible de son conteneur, donc l'élément suivant se positionne à côté. Les éléments flottants doivent **OBLIGATOIREMENT** avoir une **Largeur définie**.

**Attention les éléments flottants se placent les uns à côté des autres tant qu'ils ont la place, ensuite ils vont en dessous.**

# Propriétés des boites(blocs)

## ► Flottement et Positionnement : Exemple

```
<!DOCTYPE html>
<html>
  <head>
    <title>Cours HTML et CSS</title>
    <meta charset= "utf-8">
    <link rel="stylesheet" href="cours.css">
  </head>

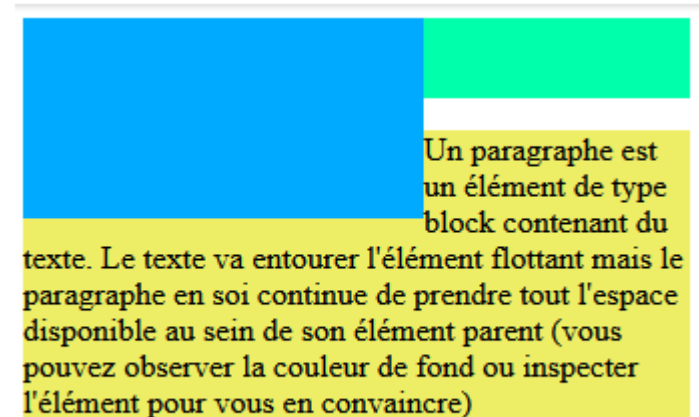
  <body>
    <div class="flottant"></div>
    <div></div>
    <p>Un paragraphe est un élément de type block contenant du texte. Le
    texte va entourer l'élément flottant mais le paragraphe en soi continue
    de prendre tout l'espace disponible au sein de son élément parent
    (vous pouvez observer la couleur de fond ou inspecter l'élément
    pour vous en convaincre)</p>
  </body>
</html>
```

Code html

Cours.css

```
.flottant{
  width: 200px;
  height: 100px;
  float: left;
  background-color: #0AF; /*Bleu*/
}
div{
  height: 40px;
  background-color: #0FA; /*Vert*/
}
p{
  background-color: #EE6; /*Jaune*/
}
```

résultat



# Propriétés des boîtes(blocs)

## ➡ Flottement et Positionnement

➤ sortir du flottement = **clear** : left, right ou both

❖ Puisque tous les éléments qui se positionnent les uns à côté des autres doivent avoir la propriété "**float**" y compris le dernier, il faut ensuite "**forcer**" le retour à la ligne.

❖ Il est nécessaire d'avoir un élément qui permet de revenir à la ligne.

❖ S'il y a eu du flottement à gauche, il faut : **clear:left**

❖ S'il y a eu du flottement à droite, il faut : **clear:right**

❖ S'il y a eu du flottement à gauche et à droite, il faut : **clear:both**



# Propriétés Display

➤ **display** permet de spécifier la manière dont un élément est affiché.

```
Balise{  
  display: valeur;  
}
```

❖ **valeur** peut prendre:

- ❖ **none**: l'élément **n'est pas affiché** et la **boîte** qui lui est **associée n'est pas créée**.
- ❖ **inline**: l'élément devient du **type en ligne** (comme `<span>` par exemple).
- ❖ **block**: l'élément devient **du type bloc** (comme `<h1>`, `<p>`, `<div>` ...).
- ❖ **list-item**: l'élément devient du **type liste** (équivalent de `<li>`).

# Liens

## Propriétés des liens

- ❖ Pour **contrôler les effets appliquées aux liens**, on utilise ce qu'on appelle des **pseudo-classes**.
- ❖ Les **pseudo-classes** permettent de tenir compte des conditions et événements différents pour la définition d'une propriété sur une balise HTML.
- ❖ Pour un lien, la **pseudo-classe** exprime l'état de celui-ci. Par exemple, le lien est-il **visité**, **actif** ou **survolé**.  
Le survol d'un lien fait se **changer en petite icône de main le pointeur de la souris** ;
- ❖ Pour chaque **état** on peut appliquer **des styles CSS déjà vu tel que les couleurs, les arrière-plans, les propriétés de texte**.
- ❖ En HTML les liens sont exprimées avec les balises **<a>**. En CSS on utilise « **a** » comme **sélecteur**.

# Liens

## Propriétés des liens

### Exemple

```
a {  
    color: blue;  
}
```

- Lien non visité:

Les liens sont alors mis en bleu. Cependant, ils peuvent avoir plusieurs états.

```
a:link {  
    color: bleu;  
    text-decoration: none;  
}
```

- Lien visité:

On ajoute au sélecteur « a » la pseudo-classe « :link » pour désigner un lien non visité.

```
a:visited {  
    color: orange;  
    background-color: green;  
}
```

On ajoute au sélecteur « a » la pseudo-classe « :visited » pour désigner un lien visité.

# Liens

## Propriétés des liens

### Exemple

#### - *Lien actif:*

```
a:active {  
    color: yellow;  
}
```

On ajoute au sélecteur « a » la pseudo-classe « **:active** » pour désigner un lien actif.

#### - *Lien survolé:*

L'un des effet les plus fréquents sur les pages Web consiste à changer les propriétés du lien au moment du survol (**quand on place le curseur de la souris en dessus**).

```
a:hover {  
    color: purple;  
    letter-spacing: 4px;  
    font-variant: small-caps;  
    text-decoration: underline;  
}
```

On ajoute au sélecteur « a » la pseudo-classe « **:hover** » pour désigner un lien survolé.

# Liens

## Propriétés des liens

### - *Lien focus:*

focus, qui permet de **modifier le style d'un lien** quand il **reçoit le focus** à l'aide de la **touche Tab** ou d'un **raccourci clavier**.

```
a:focus{  
    font-size:200%;  
    border: yellow 2px  
    double;  
}
```


On ajoute au sélecteur « a » la pseudo-classe « **:focus** » pour désigner un lien actif.

# TP d'application


## ➡ TP: réaliser la page suivante en utilisant HTML et CSS

/C:/Users/mm/Desktop/DWM-IA/te X


file:///C:/Users/mm/Desktop/DWM-IA/technologie web/Tp6/TP6.html



Accueil | Mon CV | Enseignement | Recherche | Contact |




La vision par ordinateur (aussi appelée vision artificielle ou vision numérique) est une branche de l'intelligence artificielle dont le but est de permettre à une machine de comprendre ce qu'elle « voit » lorsqu'on la connecte à une ou plusieurs caméras. Les applications vont de la vision industrielle (par exemple dans l'industrie de fabrication de bouteilles), à la recherche dans le domaine de l'intelligence artificielle et des ordinateurs ou robots capables de « comprendre » le monde qui les entoure.



Le calibrage d'une caméra consiste à déterminer les paramètres intrinsèque et extrinsèque en utilisant un objet connu appelé mire de calibrage. La mire peut être tridimensionnelle (calibrage 3D) ou plan (calibrage 2D). La contrainte de la mire n'est pas toujours présent dans les applications de vision par ordinateur, ce qui donne naissance des nouvelles méthodes appelées auto-calibrage qui permettent de calculer les paramètres intrinsèques et extrinsèques sans aucune connaissance préalable sur la scène. Les méthodes d'auto-calibrage utilisent des scènes 3D ou des scènes planes pour calculer les paramètres de la caméra de manière automatique.

A propos de moi:



Diplôme: Informatique

Domaine de recherche : Vision par Ordinateur

Université: Sidi Mohamed Ibnou abdellah

Laboratoire: LIAN

Publication: Auto-calibrage de cmaéra CCD

Copyright tous droits réservés  
Me Contacter

# Conclusion

**A retenir pour ce chapitre:**

## *Les feuilles de style*

*Introduction, Définitions, Comment appliquer CSS à une page HTML*



## *Structure d'une règle*



*Les couleurs et les arrière-plan,  
Les polices, Mise en forme des  
listes via CSS*



*Liens, Les ombres, Propriétés des  
boîtes,*