Partie 1 - Web, Langage HTML, Dynamic HTML et Feuilles de styles

Olivier GLÜCK
Université LYON 1/UFR d'Informatique
Olivier.Gluck@ens-lyon.fr
http://www710.univ-lyon1.fr/~ogluck

Copyright

Remerciements

- Quelques transparents sont directement tirés des supports de cours de :
  - Fabrice Kordon, Isabelle Mounier, Christian Queinnec (PARIS 6)
  - Dominique Bouillet (INT)
  - Laurent Lefèvre (ENS LYON)
  - Olivier Aubert (LYON 1)
- Merci à eux !
- Des figures et exemples sont issus des livres cités en bibliographie

Organisation pratique et contenu du module

Le module PW : organisation (1)

- Un des modules obligatoires du M1IF
- A l'articulation de :
  - « Réseaux » de M1IF
  - « Modèle C/S, applications de l'Internet et administration réseau » de M2SIR
  - La programmation en langage C et shell Unix
- Débouchés
  - Surtout Webmaster
  - Un peu Administrateur système
- Polycopié de cours : AML ou en ligne sur http://www710.univ-lyon1.fr/~ogluck/

Le module PW : organisation (2)

- 30 h de présentiel étudiant :
  - 15 h de cours + 15 h de travaux pratiques
- TP en groupes tous les 15 jours (2 gr. une semaine, 2 gr. la semaine suivante) :
  - **Les groupes sont fixés une fois pour toutes**
  - Salles Réseaux : TP11 et TD4 (Linux/Windows 2000)
    - pas d'accès extérieur
    - possibilité de câblage
    - root sur les machines
- Contrôle continu (**40%**) :
  - Présence aux TPs
  - Comptes-rendus de TP + évaluation en direct
- Il est impératif de travailler en dehors des cours et TP ; un mini-projet dans chaque TP (3 sujets de TP sur 5 séances de 3h)

Le module PW : contenu

Le Web et sa programmation !

- Partie 1 (3 séances de 1h30)
  - Web, Langage HTML, Dynamic HTML et Feuilles de styles
- Partie 2 (2 séances)
  - Le langage JavaScript
- Partie 3 (3 séances)
  - Formulaires, protocole HTTP et programmation CGI
- Partie 4 (2 séances)
  - Le langage PHP

Bibliographie

- « *Webmaster in a nutshell* », S. Spainhour & R. Eckstein, 3ième édition, O'REILLY, ISBN 0-596-00357-9
- « *Création d'un site Web du débutant à l'expert* », Daniel Ichbiah, Eska, ISBN 2-7472-0227-5
- « *HTML et JavaScript* », P. Chaléat et Daniel Charnay, Eyrolles, ISBN 2-212-11157-6
- Internet…
  - http://allhtml.com/
  - http://www.lissaexplains.com/french/
  - http://www.philgate.com/
  - http://developpementweb.online.fr/index1.html
  - http://www.laltruiste.com/accueil.php?compteur=1&evolution=1
  - http://www.commentcamarche.net/ccmdoc/
  - http://www.w3.org/

Plan de la partie 1 (3 séances)

- Internet et le Web
- Premier outil du Web : le langage HTML
- HTML "avancé"
  - Les cadres
  - Les images sensibles
  - Référencer son site (les META données)
  - Les évolutions de HTML
- Introduction à la programmation Web
- Dynamic HTML
  - Pourquoi Dynamic HTML ?
  - Les feuilles de styles

Internet et le Web

Internet...

- Un réseau de réseaux
- Un ensemble de logiciels et de protocoles
- Basé sur l'architecture TCP/IP
- Fonctionne en mode Client/Serveur
- Offre un ensemble de **services** (e-mail, transfert de fichiers, connexion à distance, WWW, …)
- Une somme « d'inventions » qui s'accumulent
  - mécanismes réseau de base (TCP/IP)
  - gestion des noms et des adresses
  - des outils et des protocoles spécialisés
  - le langage HTML

Internet...

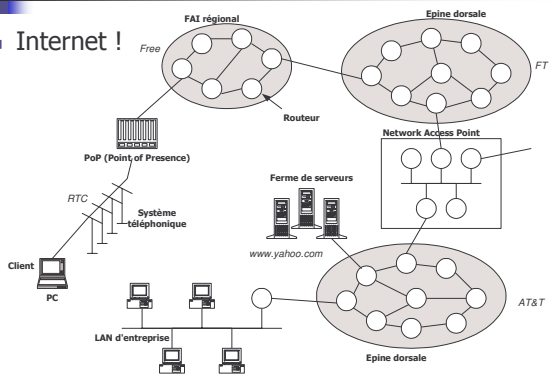
- Une construction à partir du « bas »
  - réseau local (laboratoire, département)
  - réseau local (campus, entreprise)
  - réseau régional
  - réseau national
  - réseau mondial
- 3 niveaux d'interconnexion
  - postes de travail (ordinateur, terminal...)
  - liaisons physiques (câble, fibre, RTC...)
  - routeurs (équipement spécialisé, ordinateur...)

Intranet et Extranet

- Intranet
  - utilisation des protocoles, outils et services de l'Internet dans un cadre privé (établissement, entreprise, état...)
  - intégration des services existants : accès aux BD, messageries, forums …
- Extranet : ouverture vers des partenaires (fournisseurs, clients...)

Réseaux de réseaux...

- Internet !

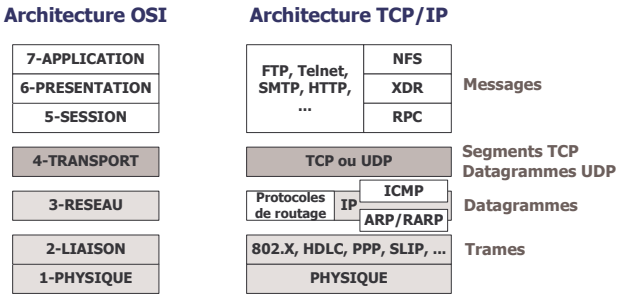


Les protocoles de l'Internet

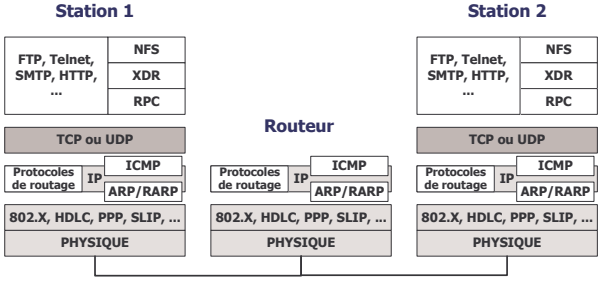
- Une version simplifiée du modèle OSI
  - Application FTP, WWW, telnet, SMTP, …
  - Transport TCP, UDP (entre 2 processus aux extrémités)
  - Réseau IP (routage)
  - Transmission entre 2 sites : pas de protocole spécifique

TCP	*Transport Control Protocol*
UDP	*User Datagram Protocol*
IP	*Internet Protocol*

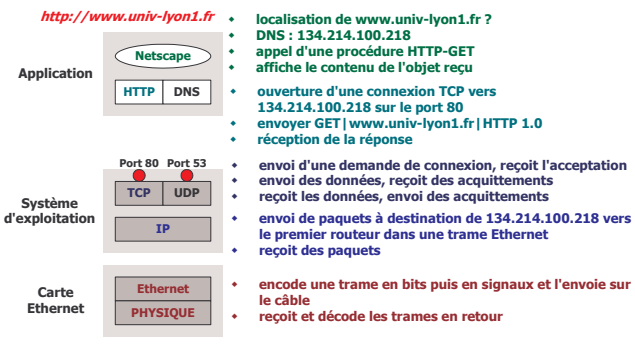
Architecture TCP/IP



Interconnexion dans TCP/IP



Exemple d'une requête HTTP



Bref historique d'Internet

- 1959-1968 : Programme ARPA
  - ministère américain de la défense : lancer un réseau capable de supporter les conséquences d'un conflit nucléaire
- 1969 : ARPANET, l'ancêtre d'Internet
  - les universités américaines s'équipent de gros ordinateurs et se connectent au réseau ARPANET
- 1970-1982 : Ouverture sur le monde
  - premières connexions avec la Norvège et Londres
- 1983 : Naissance d'Internet
  - protocole TCP/IP -> tous les réseaux s'interconnectent, les militaires quittent le navire

Bref historique d'Internet

- 1986 : Les autoroutes de l'Information
  - la *National Science Fondation* décide de déployer des super-ordinateurs pour augmenter le débit d'Internet
- 1987-1992 : Les années d'expansion
  - les fournisseurs d'accès apparaissent, les entreprises privées se connectent au réseau
- 1993-2003 : L'explosion d'Internet
  - ouverture au grand public
  - avènement du WEB et du courrier électronique
  - -> marché considérable

Bref historique d'Internet

1970 **Arpanet, premier e-mail**
premier Ethernet expérimental, TCP/IP

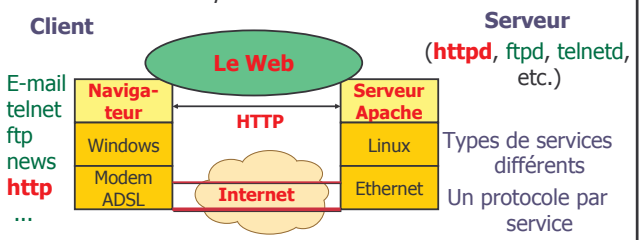
X.25
Transpac
1980 première carte Ethernet, Unix BSD sockets, NFS
1000 sites connectés, DNS
Token Ring
FDDI
10 000 sites connectés, apparition du WWW
1990 Arpanet -> Internet, X.25 -> Frame Relay, ATM
100 WWW, première radio sur Internet
Lycos, Java, IP sur ATM
Altavista
ligne Renater France/US à 155Mbit/s
2000

Les applications domestiques

- Internet, Internet, Internet...
  - recherche d'informations
  - communication entre personnes (e-mail, forums, messagerie instantanée, *chat*…)
  - divertissements interactifs
  - commerce électronique, vente aux enchères
  - gestion comptes en banques, opérations boursières
  - démarches administratives
  - *peer-to-peer* : (<> client/serveur) Napster
  - téléphonie, visiophonie, radio, vidéos à la carte…
  - enseignement à distance, travail à domicile

Les services d'Internet

- Un service = une **application** qui utilise un **protocole** et un numéro de **port**
- Fonctionnement en mode **Client/Serveur** au dessus de TCP/IP



World Wide Web

- Architecture pour accéder à des documents liés entre eux et situés sur des machines reliées par Internet
- Architecture basée sur 3 concepts :
  - la localisation --> **URL**
  - le protocole --> **HTTP**
  - le langage --> **HTML**
- Popularité due à :
  - interfaces graphiques conviviales
  - très grande quantité d'informations
  - grande diversité des informations

Le jargon du Web

- Une page Web :
  - contient des "objets"
  - désignée par une adresse (URL)
- La plupart des pages Web contiennent :
  - du code HTML de base
  - des objets référencés
- L'URL a au moins deux composantes :
  - le nom d'hôte contenant la page Web
  - le chemin d'accès sur l'hôte
- L'Agent Utilisateur pour le Web est le *browser* :
  - MS Internet Explorer
  - Netscape Communicator
  - ...
- Le serveur Web :
  - Apache (domaine public)
  - MS Internet Information Server

`www.someSchool.edu/someDept/pic.gif`

Origines du Web

- Naissance au CERN : besoin d'échanges de documents, rapports, croquis, photos... entre des grosses équipes internationales pour des expériences demandant de longs investissements de mise en œuvre
  - mars 89 : Tim Berners-Lee : réseau de documents
  - septembre 90 : 1er prototype (mode texte)
  - décembre 91 : démonstration publique à la conférence Hypertext'91 de San Antonio

Envol du Web

- Février 93 : 1ère interface graphique *Mosaic* (Marc Andreesen)
- 1994 : M. Andreesen crée *Netscape Comm. Corp.* (développements logiciels pour le web)
- 1994 : création du W3C (**WWW Consortium**) par le CERN et le MIT (Tim Berners-Lee président) (développements du Web, standards...)
- 1996 : apparition des feuilles de styles (CSS)

Fonctionnement du Web

- Le client (navigateur ou *browser*) dialogue avec un serveur Web selon le protocole HTTP
- Le serveur vérifie la demande, les autorisations et transmet l'information
- Le navigateur interprète le fichier reçu et l'affiche (le navigateur, un *plug-in* ou un *helper*)
- A ce schéma de base, peuvent s'ajouter :
  - des **contrôles** par compte individuel, par domaine, par adresse IP...
  - des **exécutions** de code côté serveur et/ou côté client

Adressage des documents

- Il faut nommer, localiser et accéder à une page :
--> 3 questions : Quoi ? Où ? Comment ?
- Solution :
  - URL - *Uniform Resource Locator* : Adresse universelle de ressource
  - en 3 parties : le protocole (comment), le nom DNS (où) et le nom du document (quoi)
- URL --> URI (*Universal Resource Identifier*)
  - un sur-ensemble des URLs
- URL classique (simplifiée) :

`http://www.monsite.fr/projet/doc.html`

Adressage des documents

- Différentes composantes d'une URL :

`proto://host_name:port/path/extra_path?arguments`

  - la racine "`/`" de `path` est définie par la configuration du serveur Web
    (**Attention :** à ne pas confondre avec la racine du système de fichiers sur le serveur)
  - `/path` peut contenir une étiquette (point d'ancrage)

`http://www.monsite.fr/projet/doc.html#label`

  - `extra_path` (après `.cgi` par ex.) et `arguments` permettent de passer des informations à des programmes s'exécutant sur le serveur

Adressage des documents

- URL relative :
  - un lien vers "`images/new.gif`" dans la page `http://www.monsite.fr/projet/doc.html`

    est - par défaut - un lien vers

    `http://www.monsite.fr/projet/images/new.gif`
  - le navigateur client reconstruit l'URL absolue pour faire la requête
  - la balise HTML <BASE href="url"> permet de positionner la racine absolue pour les URLs relatives du document contenant cette balise

Vision côté client

- Le Web est un ensemble de pages (documents) pouvant contenir des liens vers d'autres pages n'importe où dans le monde
- Consultation des pages via un navigateur
- L'utilisateur suit ces liens par simple click --> notion d'hypertexte (information répartie)
- Le navigateur (*browser*)
  - analyse l'URL demandée
  - demande au DNS l'adresse IP du site distant
  - établit une connexion TCP vers le numéro de port de l'URL (80 par défaut)
  - formule la requête au serveur

Vision côté client

- Le navigateur (*browser*)
  - va rechercher la page demandée
  - interprète les commandes de formatage et de mise en forme (police, gras, couleurs...)
  - va rechercher et affiche des images
  - animation (code JavaScript, gifs...)
  - affiche la page correctement formatée
- Paramétrage à plusieurs niveaux
  - valeurs par défaut du navigateur
  - valeurs fixées dans le document
  - préférences de l'utilisateur (navigateur)
  - exemples : couleur des liens (visités ou non), du texte, fond de la page, polices...

Vision côté serveur

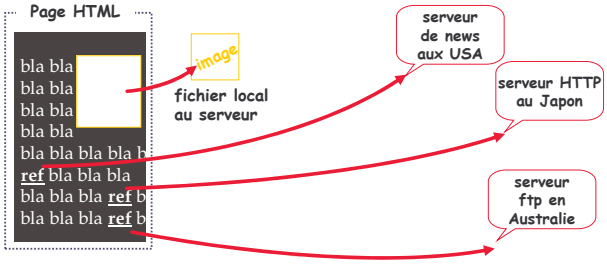
- Le serveur est en permanence à l'écoute des requêtes formulées par les clients (qui peuvent être très nombreux !)
- Il vérifie la validité de la requête...
  - adresse correcte (URL)
  - client autorisé à accéder au document
- ... et y répond : envoi du texte, des images, du code à exécuter sur le client, d'un message d'erreur, d'une demande d'authentification, ...
- Il peut exécuter un programme localement qui va générer une réponse HTML (pages dynamiques)

Le premier outil du Web : HTML

Principes de HTML

Structuration d'un texte à l'aide de balises

Ici, 4 liens hyper-textes



HTML

- *Hyper Text Markup Language*
  - un langage de description permettant de structurer et d'afficher différents objets sur un écran (textes, tableaux, images, vidéos, …)
  - une implémentation simple de SGML (*Standard Generalized Markup Language*)
  - un document HTML doit pouvoir s'afficher dans n'importe quel navigateur **MAIS**
    - les affichages ne sont pas identiques selon le contexte et le navigateur utilisé
- Pourquoi apprendre le HTML ?

HTML

- Définit la structure logique d'un document WEB
- Composé d'un ensemble de commandes de formatage
- Basé sur la notion d'**environnement** possédant un début et une fin
  - --> **délimiteurs** : balises ou marqueurs
- Pour mettre en œuvre du HTML, il suffit de
  - un éditeur de texte
  - un navigateur WEB

HTML

- Les balises sont définies entre <>
  - <BALISE>…</BALISE>
- La plupart des environnements peuvent être imbriqués selon des règles bien définies
  - <H1><B>Mon titre</B></H1>
- Il n'est pas permis de faire chevaucher des environnements
  - <H1><B>Mon titre</H1></B>
- Les minuscules/majuscules n'interviennent pas dans la définition des balises

HTML - les bases

- Les balises dénotent des "constructions" documentaires
  - styles de paragraphe (normal, énumérations, titres…)
  - tableaux
  - styles de caractères (gras, italique, souligné…)
  - références à des images
  - références hyper-texte
  - formulaires
  - etc…
- Fichiers par défaut : Welcome.html, index.html

Règles de "bonne conception"

- **Mise en page et styles** : maintenir une certaine uniformité sur l'ensemble du site
- **Clarté** : éviter la confusion, présenter de façon simple avec des menus, éviter les textes longs… ; l'information doit être trouvée dès les premières secondes
- **Hyperliens** : ne pas en abuser, faire en sorte qu'ils indiquent clairement où ils mènent
- **Identité** : le visiteur doit savoir facilement sur quel site il est (quelle que soit la page visitée) et savoir à qui s'adresser --> logos, @mail, date de dernière maj…
- **Réactivité** : ne pas abuser des images, exécutions de code… (temps de chargement/exécution)
- **Navigateurs** : faire attention à leurs particularités --> optimiser le site pour IE et Netscape…

Exemples

- Trop complexe
- Confus
- Pas de sens de lecture

Exemples

- Bonne hiérarchie
- On retrouve facilement la catégorie

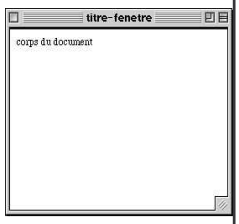
Exemples

- Montrer où on est en train de naviguer

Structure «minimale» d'un document

Indentation !

```
<HTML>
  <HEAD>
    <TITLE>titre-fenetre</TITLE>
  </HEAD>
  <BODY>
      corps du document
  </BODY>
</HTML>
```



HEAD : pour différencier du reste du texte, contient les titres

TITLE : affiché en haut de la fenêtre, utilisé dans les signets (*bookmarks*)

BODY : contient le document

Balises et Attributs

```
...
<BODY>
  <ADDRESS>
      Jean Dupont - Tel: 04 72 72 80 80
  </ADDRESS>
</BODY>
...
```

- Exemple de balise : <ADDRESS>
  - un style "adresse", généralement affiché en italique
- Les commentaires : **<!--** ne sera pas interprété **-->**
- Comme pour tout langage, RESPECTER L'IDENTATION

Balises et Attributs

- Attributs avec affectation d'une valeur ou non
  - <BALISE attribut1="10" attribut2="blue">
- Attributs "de cœur" (disponibles pour la plupart des balises)
  - class="name" : applique un style au contenu
  - id="name" : donne un nom unique à la balise
  - lang="langage" : spécifie la langue du contenu (ISO639)
  - des attributs liés aux événements : onclik="action", ondblclick, onkeydown, onkeypress, onkeyup, onmouseover, onmouseout, …

Balise <HR> et Balise

- Balise <HR> : trait horizontal
  - <HR size="2"> (en pixels)
  - <HR width="30%">
  - <HR width="100"> (en pixels)
  - <HR align="center/left/right">
  - <HR noshade> (pas de relief)
  - pas de fermeture
- Balise
 : saut de ligne
  - passage à la ligne
  - pas de fermeture

Balise <P> et Balises de titres

- Balise <P>
  - début d'un nouveau paragraphe
  - fermeture optionnelle en HTML mais obligatoire en XHTML (en XML, toute balise ouverte doit être fermée)
  - attribut align="center/left/right/justify" (justify pas toujours interprété)
- Balises <Hn>…</Hn>, n varie de 1 à 6
  - affectent la taille des caractères (taille décroissante)
  - texte en caractères gras et suivi d'un saut de ligne
  - attribut align="center/left/right"
  - <Hn> est considéré comme un paragraphe et ne doit pas être placé dans <P>…</P>

Balise <P>

```
<HTML><HEAD>
  <TITLE>titre-fenetre</TITLE>
</HEAD><BODY>
<P>Ceci est un paragraphe.</P>
<P>Ceci en est
un autre, le texte sur
plusieur lignes est
reformat&eacute;.</P>
</BODY></HTML>
```

Balises de titres

```
<HTML><HEAD>
  <TITLE>titre-fenetre</TITLE>
</HEAD><BODY>
<H1>Titre "H1"</H1>
<P>Texte sous le titre.</P>
<H2>Titre "H2"</H2>
<P>Texte sous le titre.</P>
<H3>Titre "H3"</H3>
<P>Texte sous le titre.</P>
</BODY></HTML>
```

Balise <PRE> - pré-formatage

- Texte pré-formaté
- Permet de conserver le formatage du texte tel qu'il a été saisi dans le fichier source HTML (respect de la mise en page précise : espacement, saut de ligne...)
- Utile par exemple pour faire un tableau "manuellement"

Les caractères spéciaux

- Pour être sûr qu'ils soient correctement interprétés par tous les navigateurs :
  - caractères accentués : **&<lettre><accent>;**

  accent : `acute` (aigu), `grave` (grave), `circ` (^), `uml` (¨)

  exemples : `&eacute;` (é), `&Icirc;` (Î), `&uuml;` (ü)
  - autres caractères :

  `&ntilde;` (ñ), `&ccedil;` (ç), `&szlig;` (ß), `&copy;` (©)

  `&aelig;` (æ), `&` (&), ` ` (espace), `&quot;` (")

  `>` (>), `<` (<)

Balises <UL> <OL> <LI> - listes

- Non numérotées

  <UL> : Unnumbered List

  <LI>…</LI> : List Item

  </UL>
  - attribut `type="square/circle/disc"` pour <UL> ou <LI> **(priorité au tag interne en HTML !)**
- Numérotées

  <OL>

  <LI>…</LI>

  </OL>
  - attribut `type="1/A/a/I/i"` pour <OL>
  - attribut `start="valeur"` pour <OL> (valeur de départ)
  - attribut `value="valeur"` pour <LI> (réinitialise le séquencement à la nouvelle valeur)

Balises <UL> et <LI>


```
...
<UL>
  <LI>&eacute;l&eacute;ment 1</LI>
  <LI>&eacute;l&eacute;ment 2</LI>
  <UL>
    <LI>&eacute;l&eacute;ment 3.1</LI>
    <LI>&eacute;l&eacute;ment 3.2</LI>
  </UL>
</UL>
...
```


Balises de formatage de texte


```
<HTML><HEAD>
  <TITLE>titre-fenetre</TITLE>
</HEAD><BODY>
<P>Avant une <B>liste</B> :</P>
<UL><LI><I>&eacute;l&eacute;ment 1</I></LI>
<LI>&eacute;l&eacute;ment 2</LI>
<UL><LI>&eacute;l&eacute;ment 3.1</LI>
<LI>&eacute;l&eacute;ment 3.2</LI>
</UL></UL>
</BODY></HTML>
```


Balises de formatage de texte

- <CENTER>…</CENTER> - centrer le contenu
- Taille, police et couleur de caractères

<FONT size="taille" face="police" color="blue">...</FONT>

  - taille comprise entre 1 et 7
  - par défaut : 3 (équivalent à 12 pt)
  - taille absolue
    - <FONT size="7">TRES GROS</FONT>
    - <FONT size="1">très petit</FONT>
  - taille relative
    - <FONT size="+1">plus gros</FONT>

Obsolète

Balises de formatage de texte

- <CENTER>…</CENTER> est remplacé par **<DIV>…</DIV>** qui permet en plus un alignement à droite ou à gauche
  - attribut align="left/center/right"
  - attribut nowrap : le contenu n'est pas renvoyé à la ligne automatiquement --> ascenseur
- <FONT>…</FONT> est remplacé
  - par les feuilles de styles (CSS)
  - ou par <SMALL>, <BIG>, …

Formatage de caractères

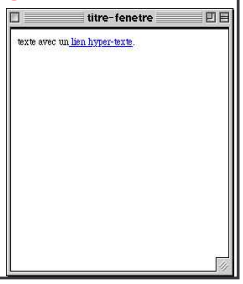
- Styles physiques
  - sont indépendants du navigateur utilisé et de sa configuration
    - <B>…</B> : gras
    - <I>…</I> : italique
    - <TT>…</TT> : machine à écrire (police largeur fixée)
    - <STRIKE>…</STRIKE> : texte barré
    - <U>…</U> : souligné
    - _… : indice
    - […] : exposant
    - <SMALL>…</SMALL> : petite police
    - <BIG>…</BIG> : grande police

Formatage de caractères

- Styles logiques
  - peuvent dépendre du navigateur utilisé et de sa configuration
    - <STRONG> : gras
    - <EM> : mettre un texte en valeur (italique)
    - <DFN> : définition
    - <CITE> : citation bibliographique
    - <CODE> : programme informatique
    - <BLOCKQUOTE> : tabulation+espaces paragraphes
    - …

Balises pour les liens hyper-texte

```
<HTML><HEAD>
  <TITLE>titre-fenetre</TITLE>
</HEAD><BODY>
<P>texte avec un <A href="cible.html">
lien hyper-texte</A>.</P>
</BODY></HTML>
```



Balise <A> : liens et étiquettes

- Définition d'un hyperlien <A href="nom">texte</A>
  - **href**="nom" : nom du document à lier
    - une URL standard : `http://www.univ-lyon1.fr`
    - une adresse mail : `mailto:toto@zozo.fr` (mais spam)
    - un chemin relatif : `Cours/index2.html`
    - un chemin absolu : `/` est la racine du serveur Web !
    - une étiquette : `#LABEL`
    - une combinaison URL, chemin relatif, étiquette
  - texte : texte utilisé pour décrire/représenter le lien dans le document
- Définition d'une étiquette
  - <A **name**="LABEL">…</A>
  - le texte entre <A> et </A> n'est pas une zone cliquable

Balise <BODY> - paramètre la page

- Fond d'écran
  <BODY **background**="nom_image">
  - répétition d'une image sur l'écran du navigateur
- Couleurs
  <BODY
  bgcolor="#000000" : couleur de fond
  text="#000000" : couleur du texte
  link="#000000" : couleur des liens non encore visités
  vlink="#000000" : couleur des liens déjà visités
  alink="#000000" : couleur des liens lors du clic
  - Noms des couleurs
    - en clair : aqua, black, gray, olive...
    - en notation hexadécimale RGB

Balise <IMG> - insertion d'images

```
<HTML><HEAD>
  <TITLE>titre-fenetre</TITLE>
</HEAD><BODY>
<P>Texte sur l'image.</P>
<P><IMG src="logo.jpg"></P>
<P>Texte sous l'image.</P>
</BODY></HTML>
```

Balise <IMG> - insertion d'images

- Insertion d'une image dans un document :

<IMG src="nom**"**
  - nom local ou URL distante
  - extensions : jpeg, gif, xbm ...

<IMG height="hauteur**" width="**largeur**"**
  - informe le navigateur de la hauteur et la largeur de l'image en pixels (évite au navigateur de les calculer)
  - peut aussi permettre de redimensionner l'image

<IMG align="bottom/middle/top/left/right**"**
  - alignement par rapport au texte (*left* et *bottom* par défaut)

Balise <IMG> - insertion d'images

<IMG border="nb_pixels**"**
  - affiche un cadre autour de l'image

<IMG vspace="nb_pixels**" hspace="**nb_pixels**"**
  - tailles des marges (espacement image/texte)

<IMG alt="ceci est une info-bulle**"**
  - définit un commentaire pour l'image
  - affiché à la place de l'image pendant le téléchargement
  - affiche une info-bulle quand la souris passe sur l'image (certains navigateurs seulement)
  - donne des infos aux indexeurs
- L'attribut **alt** est aussi utilisable dans d'autres balises

Accélérer le chargement des pages ?

- Les visiteurs ne patientent pas plus de 10 secondes pour charger une page..
- Données Multimédia réellement utiles ?
- Ré-utilisation des images par ex. (de index.html)
- **Réduire** la taille des images
- Utiliser les attributs *height* et *width*, pour permettre au navigateur de fixer le texte le plus tôt possible
- Découper les longues pages

Balise <EMBED>

- Permet
  - d'inclure des fichiers multimédias : sons, vidéos
  - de les jouer automatiquement
- Attributs

  <EMBED **src**="nom_fichier" **autostart**="true|false" **hidden**="true|false" width="145" height="60" **loop**="n|infinite">
  - autostart permet le lancement automatique du fichier
  - hidden affiche ou non la boite de dialogue associée au fichier (lecteur multimédia) avec les dimensions width et height
- Attention à la lenteur du chargement du fichier

Tableaux

- Complètement définis et normalisés dans HTML 4.0
- Permettent de positionner précisément les objets dans le navigateur
- Les cases (cellules) peuvent contenir des données multimédia (textes, images, liens...)
- Les tableaux sont très utilisés en particulier pour gérer la mise en page

Balises <TABLE> <TR> <TH> <TD>

- <TABLE>…</TABLE> définit un tableau dans un nouveau paragraphe
- Spécification d'une ligne

  <TR>…</TR>

  marquent le début et la fin d'une ligne du tableau
- Spécification des cellules
  - Cellules d'en-tête (centré, en gras)

    <TH>…</TH>
  - Cellules de données

    <TD>…</TD>

Balise <TABLE>

- Accélérer l'affichage des tableaux (HTML 4.0)
  - fixer le nombre de colonnes
    - ex : <TABLE **cols**="nb">
  - fixer la taille du tableau
    - par défaut, la taille est calculée automatiquement par le navigateur en fonction du contenu des cellules et de la taille de la fenêtre
    - largeur fixée

      <TABLE **width**="nb_pixels ou %"> : taille en pixels ou % par rapport à la fenêtre
    - hauteur fixée

      <TABLE **height** ="nb_pixels ou %">

Balise <TABLE>

- Epaisseur des bordures du tableau
  - encadrement du tableau : attribut **border**

    <TABLE border> : équivalent à border="1"
  - épaisseur des traits de bordure

    <TABLE border="nb_pixels">
- Espacements
  - distance en pixels entre les bords et le contenu des cellules

    <TABLE **cellpadding**="nb_pixels">
  - distance entre les cellules du tableau en pixels

    <TABLE **cellspacing**="nb_pixels">
- Positionnement du tableau dans la page
  - <TABLE **align**="left|center|right">

Balise <TABLE>

- Image d'arrière plan
  - <TABLE **background**="nom_image">
- Couleurs du fond et des bords
  - <TABLE **bgcolor**="couleur">
  - <TABLE **bordercolor**="couleur">
- Ombrage
  - <TABLE **bordercolordark**="couleur"> : définit la couleur la plus sombre de l'ombrage
  - <TABLE **bordercolorlight**="couleur"> : définit la couleur la plus claire de l'ombrage

Balises <TR>, <TH> et <TD>

- Attributs communs aux lignes et cellules
  - alignement horizontal du contenu dans une cellule ou une ligne

  <TR **align**="left|center|right">…</TR>

  <TD align="left|center|right">…</TD>
  - alignement horizontal du contenu dans une cellule ou une ligne

  <TR **valign**="top|middle|bottom">…</TR>

  <TD valign="top|middle|bottom">…</TD>
  - Couleurs d'arrière plan d'une cellule ou d'une ligne

  <TR **bgcolor**="couleur ou #RRVVBB">…</TR>

  <TD bgcolor="couleur ou #RRVVBB">…</TD>

Balises <TH> et <TD> - cellules

- Largeur et hauteur d'une cellule
  - La hauteur pour une cellule est appliquée à toutes les cellules de la ligne
  - La largeur d'une cellule est appliquée à toutes les cellules de la colonne.
    - il suffit de formater une seule cellule
    - les tailles sont prises en compte si non contraires à la taille du tableau, sinon le navigateur essaie d'harmoniser

  <TH **width**="nb_pixels|%" **height**="nb_pixels|%">
- Empêcher le passage automatique à la ligne dans la cellule

  <TD **nowrap**>

Fusion de cellules - Balise <CAPTION>

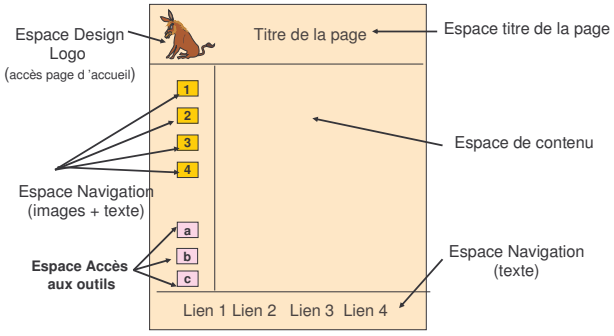
- Fusions de cellules
  - définit le nombre de colonnes sur lesquelles s'étend une cellule

  <TH **colspan**="nombre">…</TH> : nombre de cellules à fusionner sur une ligne
  - définit le nombre de lignes sur lesquelles s'étend une cellule

  <TD **rowspan**="nombre">…</TD> : nombre de cellules à fusionner sur une colonne
  - ces fusions doivent être cohérentes par rapport au lignes et colonnes du tableau
- Balise <CAPTION align="left|center|right"> : permet de donner un titre au tableau

Les cadres en HTML

Organiser sa page HTML



Les cadres ou "*frames*"

- Permettent de créer des cadres indépendants à l'intérieur d'une fenêtre du navigateur
  - développés par Netscape
  - existent depuis la version 3.0
- Permettent d'accélérer le chargement des pages
  - seule le cadre qui change est chargé
- Souvent utilisés pour avoir
  - un petite cadre pour un index fixe
  - un grand cadre pour les données à afficher
- Décriés par certains concepteurs
  - certains moteurs de recherche ont du mal à analyser les sites à base de cadres --> entrée sans *frame*
  - ascenseur par cadre pénible à utiliser
  - un cadre n'a pas d'URL --> pas de *bookmark* possible

Définition des cadres - <FRAMESET>

- Utilisation de la balise <**FRAMESET**>
  <FRAMESET>…</FRAMESET>
  - contient la description de chacun des cadres présents
  - **pas de balise <body> !!!**

```
<!-- fichier index.html -->
<html><head>
<title>Mise en place de frames</title>
</head>
<frameset cols="20%,80%">
  <frame src="menu.html" name="menu">
  <frame src="contenu.html" name="contenu">
</frameset>
</html>
```


Attributs de <FRAMESET>

- **ROWS** - Division en **zones horizontales**
  - ROWS="h1,h2, … ,hn"
  - hi peut être
    - hi=n : hauteur du cadre en pixels
    - hi=n% : hauteur du cadre en % de la taille de la zone mère
    - hi=* : le cadre prend toute la place restante disponible
  - ex : 2 frames horizontales de taille identique
    - <frameset rows="50%,50%">
- **COLS** - Division en **zones verticales**
  - COLS="l1,l2, … ,ln"
- **FRAMEBORDER** - Frontière avec effet 3D
  - valeurs yes (1) ou no (0) - 1 par défaut
- **BORDERCOLOR** - Couleur des bordures
- **BORDER** - largeur de la bordure entre cadres
  - BORDER="0" --> cadres invisibles

Portabilité des cadres - <NOFRAMES>

- Si les cadres ne sont pas supportés par le navigateur
  <NOFRAMES>
  texte d'erreur à afficher
  </NOFRAMES>
- A l'intérieur d'une balise <FRAMESET>, on ne peut mettre que les balises suivantes :
  - <FRAMESET>…</FRAMESET>
  - <FRAME>
  - <NOFRAMES>…</NOFRAMES>

Description d'un cadre - <FRAME>

- Utilisation de la balise <FRAME>
- Attributs **src** et **name** de <FRAME>
  - contenu du cadre
    - **src**="URL de la page à afficher dans le cadre"
  - nom du cadre
    - **name**="nom du cadre"
    - sert à référencer le cadre pour qu'il devienne la cible de n'importe quel autre lien hypertexte

Les attributs de <FRAME>

- **MARGINWIDTH**="n" (15 par défaut)
  - nombre de pixels entre les frontières gauche et droite de la zone et le document HTML à afficher dans la zone
- **MARGINHEIGHT**="n"
  - nombre de pixels entre les frontières haute et basse de la zone et le document HTML à afficher dans la zone
- **NORESIZE** - pas de valeur
  - le navigateur empêche le redimensionnement de la zone à l'aide de la souris

Les attributs de <FRAME>

- **SCROLLING**="yes|no|auto"
  - indique si la zone doit posséder une barre de défilement
  - yes - affiche une barre même si le document est plus petit que le cadre
  - no - n'affiche jamais la barre - des parties du document peuvent ne pas être atteinte
  - auto - la barre apparaît si nécessaire (valeur par défaut)

Utilisation des cadres - attribut **target**

- Attribut **target**="nom du cadre" de <A> et <FORM>
  - désigne le cadre cible du lien
  - nom réservé ou nom local au frameset

```
<!-- fichier index.html -->
<html><head>
<title>Mise en place de frames</title>
</head>
<frameset cols="20%,80%">
  <frame src="menu.html" name="menu">
  <frame src="contenu.html" name="contenu">
</frameset>
</html>
```

```
<!-- fichier menu.html -->
<html><head>
<title>Mise en place de frames</title>
</head><body>
Menu<br>
<a href="1.html" target="contenu">Cours1</a>
<br>
<a href="2.html" target="contenu">Cours2</a>
<br>
<a href="3.html" target="contenu">Cours3</a>
<br>
</body></html>
```


Utilisation des cadres - attribut target

- Il faut maintenant créer contenu.html, 1.html...

```
<!-- contenu.html -->
<html><head>
<title>Mise en place de frames (2)</title>
</head><body>
ici apparaitra le contenu<br>
</body>
</html>
```

```
<!-- 1.html -->
<html><head>
<title>Cours 1</title>
</head><body>
Internet...<br>
</body>
</html>
```


Les liens et les cadres - attribut **target**

- L'attribut **target**
  - target contient le nom d'un cadre existant
    - la cible est affichée dans le cadre nommé
  - target n'est pas renseigné
    - la cible est affichée dans le cadre courant
  - target contient le nom d'un cadre inexistant
    - la cible est affichée dans une nouvelle fenêtre
  - target contient une valeur réservée

_self : la cible est affichée dans le cadre où est défini le lien

_parent : la cible est affichée dans le cadre "père" (celui qui l'a créé)

_blank : la cible est affichée dans une autre fenêtre sans structure de cadre

_top : la cible affichée remplace la fenêtre courante --> supprime tous les cadres

- utilisez **_top** dès qu'un lien pointe sur une page extérieure

Imbrication de cadres

```
<!-- fichier index.html -->
<html><head>
<title>Imbrication de cadres</title>
</head>
<frameset rows="25%,75%" frameborder="yes">
        <frame src="haut.html" name="haut">
        <frameset cols="*,2*" border="10">
                <frame src="basgauche.html" name="basgauche">
                <frame src="basdroit.html" name="basdroit">
        </frameset>
</frameset>
</html>
```

Mettre frameborder="no"

Imbrication de documents

```
<!-- fichier index.html -->
<html><head>
<title>Imbrication de documents</title>
</head>
<frameset rows="25%,75%">
        <frame src="vide.html" name="haut">
        <frame src="fils.html" name="bas">
</frameset>
</html>
```

- **Le résultat est le même que précédemment mais l'imbrication de documents permet de nommer la zone "bas"**

```
<!-- fichier fils.html -->
<html><head>
</head>
<frameset cols="*,2*" border="10">
        <frame src="vide.html" name="basgauche">
        <frame src="vide.html" name="basdroit">
</frameset>
</html>
```

Imbrication de cadres

```
<!-- fichier index.html -->
<html><head>
<title>Imbrication de cadres</title>
</head>
<frameset rows="*,*,10%">
        <frame src="vide.html" name="haut" noresize>
        <frameset cols="3*,*">
                <frame src="vide.html" name="milieugauche">
                <frame src="vide.html" name="milieudroit">
        </frameset>
        <frameset cols="*,*,2*">
                <frame src="vide.html" name="basgauche">
                <frame src="vide.html" name="basmilieu" scrolling="yes">
                <frame src="vide.html" name="basdroit">
        </frameset>
</frameset>
</html>
```

Un dernier exemple

```
<!-- descend.html -->
<html><head>
<title>Imbrication</title>
</head>
<frameset rows="*,*">
  <frame src="plushaut.html">
  <frame src="plusbas.html">
</frameset>
</html>
```

```
<!-- adroite.html -->
<html><head>
<title>adroite</title>
</head>
<frameset cols="*,*">
  <frame src="plushaut.html">
  <frame src="plusbas.html">
</frameset>
</html>
```

```
<!-- plushaut.html -->
<html><head>
<title>plushaut</title>
</head><body>
<a href="plusbas.html" target="_parent">
Plus haut</a> (plusbas.html _parent)<br>
<a href="plusbas.html" target="_top">
Au sommet</a> (plusbas.html _top)<br>
<a href="adroite.html" target="_parent">
Tourner</a> (adroite.html _parent)<br>
</body></html>
```

```
<!-- plusbas.html -->
<html><head>
<title>plusbas</title>
</head><body>
<a href="descend.html">Plus bas</a>
</body></html>
```

La balise <IFRAME>

- Permet de définir des cadres "*inline*" (dans le corps du document à l'intérieur de <body>)

```
<!-- iframe.html -->
<html><head>
</head><body>
 avant iframe
 <iframe src="vide.html" width="400" height="500" frameborder="1">
 </iframe>
 apres iframe
</body></html>
```

```
<!-- vide.html -->
<html><head>
</head><body>
  je suis l'iframe
</body></html>
```

Cadres et JavaScript

- Comment changer dynamiquement les contenus d'un ou plusieurs cadres ?
  - méthode avec JavaScript (plus tard...)

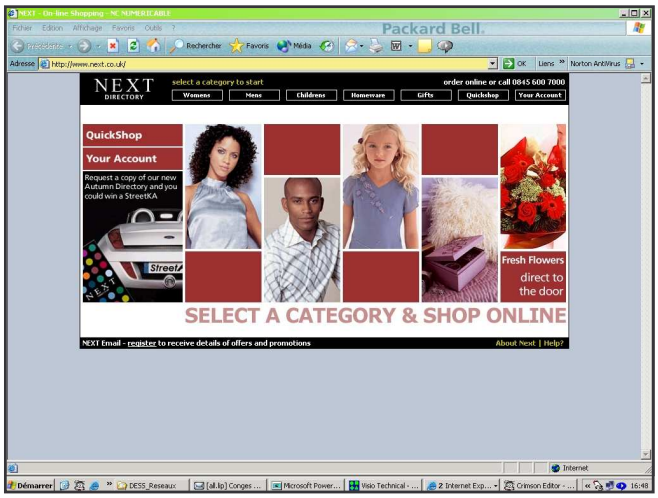
Les images sensibles

Images sensibles ou cliquables

- Images et liens

  <img src="venus_mini.gif> permet d'afficher une image "mini" et en cliquant dessus de voir l'image agrandie

  -> 1 image = 1 lien
- Principe des images cliquables
  - l'image est découpée en zones
  - on peut associer un lien différent à chaque zone de l'image
  - très souvent utilisé pour la page de garde des sites
  - facile à maintenir (découpage des zones cliquables dans un fichier séparé)



Le système de coordonnées

- Il faut commencer par définir les zones en pixels !
  - les zones peuvent être des rectangles, cercles, …

La méthode ancienne - principe

- /cgi-bin/imagemap et ISMAP
  - transmission via l'URL de la coordonnée du click au serveur
  - un programme sur le serveur récupère la position du click dans l'image et retourne la page associée à la coordonnée du click
  - méthode ancienne mais utilisable sur de nombreux navigateurs
  - non *standalone* (aller/retour entre le client et le serveur)
  - nécessite l'installation sur le serveur du programme spécifique…
  - à utiliser s'il y a trop de zones dans l'image

La méthode ancienne - exemple

- Créer un fichier /map/3zones.map sur le serveur

  default enter.html
  rect internet.html 5,5 21,14
  poly css.html 38,6 50,6 48,18 45,15 40,21 35,24
  circle http://www.univ-lyon1.fr 52,32 45,32
- Ajouter une entrée dans imagemap.conf (serveur)

  3z: /map/3zones.map
- Utilisation dans la page HTML cliente

  <img src="enter.jpg" ISMAP>

La méthode USEMAP

- Géré entièrement par le client : tout est inclus dans la page HTML qui comporte l'image cliquable
  - il est préférable de mettre la définition des zones dans un fichier séparé
- Le client détermine à partir des coordonnées du click de la souris le lien qu'il doit exécuter
- Pas d'interaction avec le serveur
  - minimisation des accès réseau
  - peut être testé dans un navigateur local sans faire appel à un serveur HTTP

Utilisation de USEMAP

- Définition d'une image sensible

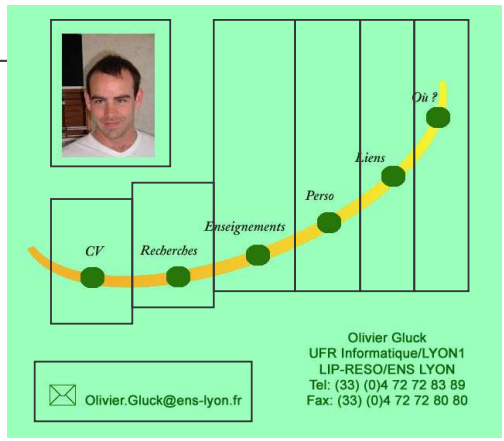
  <img src="enter.jpg" **usemap**="enter.html#3zones">
- Définition d'une carte (map)

  <**map** name="3zones"> … </**map**>
- Définition d'une zone dans la carte

  <**area shape**="fig" **href**="url" **coords**="x0,y0,x1,y1,...">
  - fig peut prendre les valeurs
    - rect=rectangle (angles supérieur gauche et inférieur bas)
    - poly=polygones (coords de chaque extrémité du vecteur)
    - circle=cercle (centre et rayon)
  - attributs **title**="infobulleIE" **alt**="infobulleNetscape"

Un exemple avec USEMAP

```
<!-- index.html -->
<html><head>
</head><body>
...
<map name="choix">
  <area shape="rect" coords="80,50,180,180" href="cv.html">
  <area shape="rect" coords="0,230,160,460" href="cv.html">
  <area shape="rect" coords="160,230,250,460" href="recherches.html">
  <area shape="rect" coords="250,0,370,380" href="enseignements.html">
  <area shape="rect" coords="370,0,445,380" href="perso.html">
  <area shape="rect" coords="445,0,520,380" href="liens.html">
  <area shape="rect" coords="520,0,640,380" href="where.html">
  <area shape="rect" coords="0,460,280,560"
                     href="mailto:Olivier.Gluck@ens-lyon.fr">
</map>
<img src="first0.JPG" border="no" usemap="#choix">
</body></html>
```

Images sensibles et JavaScript

- JavaScript permet de rendre dynamique l'affichage de l'image en fonction de la zone de la carte dans laquelle pointe la souris
- -> permet de faire des cartes cliquables animées
- Exemples d'événement JavaScript
  - onMouseOver
  - onMouseOut

Référencer son site

Référencer son site

- Un site n'a un intérêt que s'il est connu et facilement atteignable
  - -> enregistrer son site manuellement sur les principaux moteurs de recherche (google, yahoo, altavista, hotbot, voila…)
  - -> utiliser des produits commerciaux ou gratuit qui font le référencement automatiquement…
  - -> faire inscrire des liens vers son propre site dans des sites déjà référencés...
- Faut-il encore ensuite que le site soit correctement référencé…
  - -> il faut renseigner dans sa page les balises META et utiliser l'attribut ALT pour les images

La balise <META>

- Fournit des informations sur le document HTML qui sont utilisées par les robots d'indexation
- Introduite à partir de HTML 2.0
- Ces informations ne sont pas visibles à l'écran
- Placée entre <HEAD>…</HEAD> de la page d'accueil
- Utilisation classique avec attributs NAME et CONTENT

<META NAME="info" CONTENT="valeur de info">

  - des valeurs particulières de l'attribut NAME sont utilisées par les robots

La balise <META> - exemples

- <META NAME="description" CONTENT="...">
  - description du site sous la forme d'un texte (200 caractères max.)
  - ce texte sera affiché par les moteurs en guise de présentation du site
  - exemple : CONTENT="Page personnelle de …"
- <META NAME="keywords" CONTENT="...">
  - définit une liste de mots-clés qui permet d'identifier le contenu du site (500 caractères max)
  - exemple : CONTENT="enseignement, recherche, réseaux, outils internet, photographie"
- <META NAME="author" CONTENT="...">

La balise <META> - exemples

- <META NAME="robots" CONTENT="valeur">
  - donne des indications aux robots d'indexation sur
    - l'indexation de la page en cours
    - le suivi des liens présents sur la page
  - valeur=all|none|index|noindex,follow|nofollow (all par défaut)
  - exemple : CONTENT="index,follow"
- Autres valeurs pour l'attribut NAME

Identifier-URL : URL du site

Date-Creation-yyyymmdd : date de création du site

Date-Revision-yyyymmdd : date de dernière modification

Reply-to : adresse mail

Revisit-after : indique la ré-indexation du site par le robot après n jours

La balise <META>

- L'attribut LANG
  - spécifie la langue du document (également utilisé par les robots d'indexation) ou du contenu de la balise
  - valeur normalisée ISO
  - exemples :

<META NAME="description" LANG="fr"...>

<META NAME="keywords" LANG="en"...>

<META NAME="description" LANG="en-us"...>

<META NAME="keywords" LANG="de"...>

Remarques

- Les balises <META> doivent surtout être présentes dans la page d'accueil…
- La balise <TITLE> est également utilisée par les robots pour décrire le site
  - titre du site
  - limitée à 100 caractères
  - utilisée par les *bookmarks*

Autres utilisations de <META>

- L'attribut HTTP-EQUIV="info" CONTENT="valeur"
  - (info,valeur) circule dans l'en-tête HTTP du serveur vers le client quand la page est chargée - permet entre autre de paramètrer le navigateur
  - quelques exemples
    - indiquer le codage des caractères à utiliser pour l'affichage de la page par le navigateur

      <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=ISO-8859-1">
    - date limite d'expiration des pages dans le cache du client

      <META HTTP-EQUIV="expires" CONTENT="Wed, 30 Sept 2006 12:00:00 GMT">

Autres utilisations de <META>

- L'attribut HTTP-EQUIV="info" CONTENT="valeur"
  - quelques exemples
    - interdire la mise en cache sur le client

      <META HTTP-EQUIV="pragma" CONTENT="no-cache">
    - supprimer l'apparition dans une frame

      <META HTTP-EQUIV="Window-target" CONTENT="_top">
    - définir la langue du document

      <META HTTP-EQUIV="Content-Language" CONTENT="en-GB">
  - plein d'autres valeurs possibles (cf. cours HTTP)

voir http://vancouver-webpages.com/META/

Autres utilisations de <META>

- L'attribut HTTP-EQUIV et la valeur **refresh**
  - redirection automatique vers une autre URL

```
<META HTTP-EQUIV="refresh" CONTENT="10;
  URL=http://www.nouveau-site.com">
```

  - actualisation automatique de la page

```
<META HTTP-EQUIV="refresh" CONTENT="60">
```

  - utilisé par exemple pour les systèmes de caméras surveillance…

```
<!-- camera.html -->
<html><head>
<title>video-surveillance</title>
<meta http-equiv="refresh" content="5">
</head><body>
        <img src="camera1.gif"><br>
        <img src="camera2.gif">
</body></html>
```


Exemple de redirection automatique

```
<!-- redirect.html -->
<html><head>
 <title>Redirection automatique</title>
 <meta http-equiv="refresh" content="3;URL=camera.html">
</head><body>
 Veuillez noter l'@ de mon nouveau site<br>
 Vous allez être redirigé dans 3 secondes
</body></html>
```

Les évolutions de HTML

Les évolutions de HTML (1)

- 1992 : première définition de HTML (HTML 1.0)
- 1993 : ajout des formulaires et tableaux
- 1994 : HTML 2.0 normalisation des principaux éléments
- 1994 : HTML 3.0 (extension de HTML soumise en tant que draft)
- 1995 : balises non standard spécifiques à Netscape
- 1996 : guerre Netscape/IE
- 1996 : HTML 3.2 - standard basé sur les pratiques existantes

Les évolutions de HTML (2)

- 1997 : HTML 4.0 - apparition des feuilles de style, du *scripting* et du modèle Document Object Model (objets manipulables par les scripts)
- 1998 : XML : eXtensible Markup Language
  - langage de description de documents
  - permet de définir ses propres balises
  - forme allégée de SGML (description sémantique des documents qui sépare les problèmes de syntaxe de la structuration de l'information)
- 1999 : HTML 4.01 - modifications mineures
- 2000 : XHTML 1.0 - version XML de HTML 4.01
  - XHTML : eXtensible HTML (langage XML dédié aux documents hypertexte = HTML XML-compatible)

Les évolutions de HTML (3)

- Qui normalise ?
  - Initialement IETF (HTML 1.0 et 2.0)
  - Maintenant le W3C (World Wide Web Consortium)

    http://www.w3.org/
    - association d'utilité publique composée de différents groupes d'intérêt : Adobe, Microsoft, IBM, Sun…)
    - force de propositions et recommandations pour de nouvelles normes pour le WWW

http://validator.w3.org

Problème syntaxe/validation

- HTML 4.01 définit une syntaxe précise et formelle
- Tout document HTML devrait respecter cette syntaxe
- Elle peut être validée automatiquement
- En pratique, très peu de documents HTML sont valides
  - les auteurs ne respectent pas totalement la norme
  - la seule validation de leur document est leur propre test dans un navigateur particulier
  - le HTML généré par des éditeurs automatiques est souvent invalide

Problème syntaxe/validation

- Le résultat est tout de même "acceptable"
  - les navigateurs font de leur mieux pour tolérer les erreurs
  - les erreurs de syntaxe ne sont pas signalées (exemple : définir des cadres entre <body>…</body>
- Problèmes
  - différences de rendu visuel entre les navigateurs
  - difficile d'exploiter les documents de manière automatique (analyse syntaxique par *emacs* par ex…)
  - les navigateurs définissent leurs propres extensions non-standard
  - de plus en plus de navigateurs différents

XML - exemple

```
<!-- liste_cours.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<cours_olivier>
 <auteur>Olivier Glück</auteur>
  <cours>
        <titre>Internet, Web et HTML</titre>
        <niveau>Master M1</niveau>
  </cours>
 <cours>
        <titre>HTML avancé</titre>
        <niveau>Licence L3</niveau>
  </cours>
</cours_olivier>
```

XML

- XML est un langage de description de documents
  - il ne s'agit pas de décrire l'apparence mais la structure logique du document : format universel de documents structurés et de données (en vue d'échange sur le Web)
  - l'objectif est d'avoir un résultat qui ne dépende ni d'une plate-forme particulière, ni d'une application spécifique
- Il est probable que dans le futur tout programme pourra lire ou écrire du XML
- XML n'est ni le successeur de HTML, ni un langage de mise en page Web
- Dans le HTML, la structure du document et les informations de mise en page sont mélangées

XML

- Problème : la mise en page ne doit pas être la même pour
  - l'affichage dans une fenêtre de navigateur
  - une impression
  - l'affichage sur un PDA, une télé, un téléphone WAP…
- Problème inverse :
  - les documents issus de traitement de texte, tableur, base de données, … doivent être convertis en HTML avant d'être publiés sur le Web
- XML fournit un format universel auquel il faut associer des langages de mise en page selon l'utilisation souhaitée. Pour XML --> Web :
  - convertir les fichiers XML en fichiers XHTML (en PHP…)
  - ou utiliser *eXtended Stylesheet Langage* (XSL) qui permet de convertir du XML dans d'autres formats

XML - exemple

```
<!-- liste_cours_xsl.xml -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="cours.xsl"?>
<cours_olivier>…</cours_olivier>

<!-- cours.xsl -->
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html><head><title>Liste des cours</title></head><body>
  <ul>
        <xsl:for-each select="cours_olivier/cours">
        <xsl:sort select="niveau"/>
              <li><i><xsl:value-of select="titre"/></i>
              <b> (cours de <xsl:value-of select="niveau"/>)</b></li>
        </xsl:for-each>
  </ul></body></html>
</xsl:template>
</xsl:stylesheet>
```

XHTML

- XHTML est du HTML XML-compatible --> une transformation formelle pour respecter les règles du XML
  - insérer une déclaration du type de document W3C (DTD)
    <!DOCTYPE html PUBLIC …> il y en a 3 pour XHTML
  - les noms de balises et attributs en **minuscules**
  - chaque balise nécessite une balise de fin : <hr **/**>, <br **/**>, <meta name="…" content="..." **/**>, <frame src="..." name="..." **/**>, <img src="..." **/**>, …
  - toutes les valeurs d'attributs entre guillemets, chaque attribut doit avoir une valeur :
    <table border="1"> et non pas <table border>
    <hr noshade="noshade" /> et non pas <hr noshade>
  - attribut **name** remplacé par **id** (provisoirement **name** et **id** car les anciens navigateurs ne comprennent pas **id**)

Introduction à la programmation WEB - le Web interactif

Pourquoi la programmation Web ?

- Le HTML ne permet pas d'interactivité avec l'utilisateur -> les pages visualisées sont "statiques"
- Pages statiques
  - la page visualisée N fois sur le même navigateur donnera toujours le même résultat
- Pages dynamiques
  - la page visualisée dépend des manipulations de l'utilisateur - elle est générée dynamiquement selon certains paramètres définis par l'utilisateur
  - deux principaux types d'interactions
    - côté client
    - côté serveur

Programmation Web côté client

- Les exécutions associées ont lieu sur le poste client. Le navigateur doit supporter ces exécutions...
- Des scripts embarqués dans la page HTML qui sont transférés depuis le serveur vers le client
  - "*HTML-embedded scripting*"
  - exemples : javascript, vbscript, …
- Des applets
  - java, ActiveX, …
  - le code de *l'applet* est envoyé au client
  - java tourne sur toutes les plate-formes
  - le navigateur doit néanmoins disposer d'une console java...
- Des *plugins* propriétaires

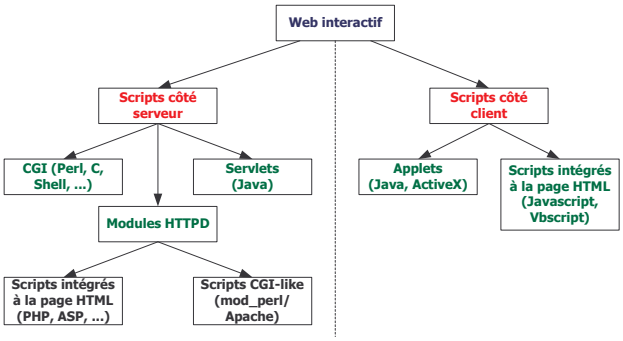
Programmation Web côté serveur

- Les exécutions associées ont lieu sur le serveur ! Le résultat de ces exécutions est une page HTML qui est renvoyée vers le navigateur client
- CGI - *Common Gateway Interface*
  - interface de communication qui définit le format d'échanges des données entre le client et le serveur HTTP
  - les paramètres utilisateur sont transmis par le serveur HTTP à un programme qui traite les requêtes et produit un résultat en HTML (page dynamique)
  - scripts en Perl ou Shell, programmes C, Ada, …
  - souvent utilisé pour les formulaires

Programmation Web côté serveur

- Interpréteurs intégrés au serveur HTTP
  - scripts embarqués dans la page HTML (*httpd modules*, *html-embedded scripting*)
    - exemples : PHP, ASP, …
  - scripts à la CGI (CGI-like)
    - exemples : *mod_perl/Apache*
- Les servlets
  - le pendant des Applets
  - en langage Java

Programmation Web - récapitulatif



Dynamic HTML

Dynamic HTML

- Idée : intégrer de la gestion dynamique d'événements grâce à JavaScript...
- Mis en place par Microsoft (IE4) et Netscape
- En cours de normalisation par W3C mais très lent
  - Netscape et IE intègrent sans cesse de nouvelles fonctionnalités non standard...
- Permet de
  - modifier dynamiquement le contenu, la structure, les styles des pages -> interactions avec l'utilisateur
  - rendre visibles ou invisibles certains éléments de la page (blocs ...)

Dynamic HTML

- DHTML est un mariage des concepts suivants
  - les feuilles de styles
  - les formulaires
    - interface de saisie
  - *javascript* et *vbscript*
    - modification (limitée) dynamique de la page sans faire appel au serveur
  - **DOM - *Dynamic Object Model***
    - association d'un objet à chaque élément d'un document HTML
    - les scripts embarqués peuvent alors agir sur ces objets
      - création de nouvelles fenêtres,
      - modifications des champs d'un formulaire,
      - modifications des styles appliqués à l'objet, ...

Dynamic HTML

- DHTML introduit aussi la notion de bloc
  - ils sont définis par la balise
    - liés aux feuilles de style
    - un bloc définit un sous-ensemble de la page contenant texte/images/formulaires ayant éventuellement des propriétés particulières (marges, bordures, couleur, visibilité, ...)
    - positionnement des blocs en CSS
    - déplacement des blocs à l'aide de Javascript --> création d'animations

Problèmes de compatibilité

- Dynamic HTML n'est pas normalisé
- IE ou Netscape versions 4 et supérieures
- L'interprétation du code peut être différente selon le navigateur
  - un même script qui fonctionne sur tous les navigateurs
  - un script différent pour chaque navigateur
  - on verra plus tard comment faire !

Les feuilles de style

Principe des feuilles de style

- Même idée que les styles des traitements de texte
  - distinction entre la structuration du texte et les propriétés typographiques que l'on applique au texte
  - les styles permettent d'appliquer certaines propriétés à des balises HTML
  - plusieurs documents peuvent partager les mêmes instructions typographiques
  - homogénéité de typographie et de présentation des pages auxquelles s'appliquent les styles
  - l'apparence des documents peut facilement être modifiés en utilisant les styles définis

Principe des feuilles de style

- Les styles
  - peuvent être définis directement dans la page HTML
  - mais sont généralement stockés dans un fichier séparé (c'est la feuille de style !)
  - permettent d'accélérer le chargement des pages (la feuille de style est chargée une fois pour toute !)
- Deux façons de décrire des feuilles de style
  - **CSS - *Cascading Style Sheets***
    - langage déclaratif type HTML
  - JASS
    - issu du langage *Javascript*
    - plus orienté vers la manipulation dynamique des propriétés décrivant un style

Principe des feuilles de style

- Mise en cascade des styles
  - possible d'imbriquer dans un même document plusieurs styles avec un niveau de priorité différent
  - les propriétés non définies pour la balise courante sont héritées des balises qui contiennent la balise courante
  - un style défini directement dans une page HTML sera plus prioritaire qu'un style défini dans un fichier externe
  - idée : on peut imaginer que plus tard, le lecteur (client) pourra redéfinir ses propres styles qui seront prioritaires sur les styles définis par le concepteur de la page

La balise <STYLE>

- Entre <head>…</head>
- L'attribut TYPE
  - text/css pour la norme CSS
  - text/javascript pour JASS (description JavaScript)
- Exemple (attention à la casse des mots-clés)

```
<html><head>
<style type="text/css">
  P {font-size: 16pt; color: blue;}
</style>
</head>
```

```
<html><head>
<style type="text/javascript">
  tags.P.fontSize=16;
  tags.P.color="blue";
</style>
</head>
```


Classes de style - attribut CLASS

- On peut définir différentes "classes"
  - une classe particulière pour un élément particulier (<h1>, <p>, <cite>, …)
    P.bleu {color: blue;}
    P.i {font-style: italic;}
    - s'applique aux paragraphes qui se revendiquent de la classe "bleu" ou "i"
  - une classe générale associée à un élément particulier
    P {font-size: 16pt; color: red;}
    - s'applique à tous les paragraphes
  - une classe particulière pour n'importe quelle balise
    .vert {color: green;}
    - s'applique à toute balise revendiquant la classe "vert"
  - **attention ! on ne peut pas cumuler des classes**

Un premier exemple

```
<!-- 1.html -->
<html><head>
        <style type="text/css">
                .vert {color: green;}
                P {font-size: 16pt; color: red;}
                P.bleu {color: blue;}
                P.italic {font-style: italic;}
                H1.titre1 {border: solid; border-width: 1;
                        text-align: center; color: cyan;}
        </style>
</head><body>
        ...
</body></html>
```


Un premier exemple

```
<!-- 1.html -->
<html><head>...
</head><body>
        <h1>h1 standard</h1>
        <h1 class="vert">h1 standard classe vert</h1>
        <h1 class="titre1">h1 classe titre1</h1>
        <p>redéfinition du standard P</p>
        <p class="bleu">P classe bleu</p>
        <p class="italic">P classe italic</p>
        Si on spécifie plusieurs classes compatibles entre elles...
        <p class="bleu" class="italic">P classes bleu et italic</p>
        <p class="italic" class="bleu">P classes italic et bleu</p>
        Si on spécifie plusieurs classes incompatibles entre elles...
        <p class="bleu" class="vert">P classes bleu et vert</p>
        <p class="vert" class="bleu">P classes vert et bleu</p>
        Si on utilise une classe non associée à l'élément...
        <h1 class="italic">h1 classe italic</h1>
</body></html>
```

on ne peut pas cumuler des classes !

Et sous Netscape ca marche ?

- Ouf, on est sauvé ! Mais cela ne valide pas le code pour autant...

Les sous-classes de style - attribut ID

- Permet de faire varier certains paramètres d'une classe
- Exemple : une classe définit un style général de paragraphe mais on souhaite pouvoir changer uniquement la définition de la couleur
  - première possibilité : faire une deuxième classe (la duplication de code est toujours une mauvaise chose)
  - mieux : définir un modificateur qui n'agit que sur les paramètres souhaités
- **#** est le caractère permettant la définition de sous-classes

Les sous-classes de style - attribut ID

- Exemples :

```
<style type="text/css">
  .standard {color: green; font-size: 10pt;}
  #cyan {color: cyan;}
  P {font-size: 16pt; color: yellow;}
  P.special {font-style: italic; color: red;}
  P#bleu {color: blue;}
</style>
```

- L'attribut ID
  - permet de réaliser une exception dans une classe ou être utilisé seul (alors équivalent à CLASS)

Un deuxième exemple

```
<!-- 2.html -->
<html><head>
    <style type="text/css">
        .standard {color: green; font-size: 10pt;}
        #cyan {color: cyan;}
        P {font-size: 16pt; color: yellow;}
        P.special {font-style: italic; color: red;}
        P#bleu {color: blue;}
    </style>
</head><body>
        <p>redéfinition du standard P</p>
        <p class="standard">P classe standard</p>
        <p class="special">P classe special</p>
        <p class="standard" id="cyan">P classe standard mais cyan</p>
        <p class="standard" id="bleu">P classe standard mais bleu</p>
        <p class="special" id="bleu">P classe special mais bleu</p>
        <h1 class="standard" id="cyan">h1 classe standard mais cyan</h1>
        ID devient une classe...
        <h1 id="cyan">h1 mais cyan</h1>
        Impossible...
        <h1 class="standard" id="bleu">h1 classe standard mais bleu</h1>
</body></html>
```

Les feuilles de style

- La définition des styles se fait dans un ou plusieurs fichiers séparés
- Permet d'appliquer les styles à un ensemble de documents
- Inclusion de commentaires comme en C /* */ avec CSS, idem avec JASS mais // en plus
- Aucune règle concernant le nommage du fichier et en particulier l'extension mais il est de bon ton de mettre .css ou .jass !
- Inclusion d'une feuille de style dans un document avec la balise <LINK>

La balise <LINK>

- Permettra de référencer d'autres fichiers externes dans le futur
- Entre <head>…</head>
- Trois attributs
  - **REL="stylesheet"** indique que le fichier inclus est une feuille de style (d'autres valeurs dans le futur…)
  - **TYPE="text/css"** ou "text/javascript" - norme de styles utilisée dans le fichier
  - **HREF="mystyles.css"** permet d'indiquer l'URL de la feuille de style
- Il est possible de lier plusieurs feuilles de style tout en continuant d'utiliser des définitions internes
- Définition ponctuelle de style

  <P **STYLE**="font-size=35; color=blue;">P en bleu 35</P>

Utilisation de feuilles de style

```
/* styles CSS pour balise P */
/* pstyle.css */
P {font-size: 16pt; color: yellow;}
P.special {font-style: italic; font-size: 30pt; color: red;}
P#bleu {color: blue;}

/* styles CSS généraux */
/* allstyle.css */
.standard {color: green; font-size: 10pt;}
#vert {color: green;}
.noirsurblanc {background-color: white; color: black;}
.blancsurnoir {background-color: black; color: white;}
#rouge {color: red;}
```


Utilisation de feuilles de style

```
<!-- 4.html -->
<html><head>
        <link rel="stylesheet" type="text/css" href="allstyle.css">
        <link rel="stylesheet" type="text/css" href="pstyle.css">
</head><body>
        <span class="noirsurblanc">
                HTML<br>
                <span class="blancsurnoir" ID="rouge">
                        et<br>
                </span>
                <span class="blancsurnoir">
                        STYLES<br>
                </span>
                c'est facile !<br>
        </span>
        <p class="special" id="vert">P special vert (2 feuilles)</p>
        <p style="color: red; background-color=blue;">
                mais je prefere utiliser ce style local<br>
                en surcharge du style général de P
        </p>
</body></html>
```

Compléments sur les styles

- Groupement de balises

  H1, H2, H3 {color: red;}

  tous les titres H1, H2, H3 seront en rouge
- Sélection contextuelle d'application d'un style

  H3 A {color: black;} : les liens définis dans <H3>…</H3> seront noirs

  H3 H3 A {color: red;} : les liens dans <H3><H3><A>…</A><H3></H3> seront rouges

- Héritage des styles

  <H1 class="titre1"><cite>HTML</cite></H1>

  des balises imbriquées dans des balises de niveau supérieur héritent du style défini dans ces balises

Compléments sur les styles

font-size: 14pt | +1pt Taille de la police
font-family: Arial | Times | … Type de police
font-weight: normal | bold | 100…900 Poids de la police
font-style: normal | italic | oblique Style de la police
font-variant: normal | small-caps Petites majuscules
text-transform: none | uppercase | lowercase Maj., Min.
text-decoration: none | underline | overline | blink | line-through
text-align: left | center | right | justify Alignement du texte
text-indent: 1pt Retrait de la première ligne
vertical-align: top | middle | bottom | … Alignement vertical
letter-spacing: 1pt Espacement des lettres
word-spacing: 3pt Espacement des mots

Compléments sur les styles

line-height: 1pt Hauteur de la ligne
color: rgb(0, 255, 0) Couleur de la police
background-image: url Image d'arrière-plan
background-color: transparent|rgb(0,255,0) Couleur d'arrière-plan
background-position: Position de l'arrière-plan
margin (ou -left, -right, -top, -bottom): 3px Marges
border: 1px Epaisseur de la bordure
border-style: solid|double|groove|ridge|dotted|dashed|...
boder-style-left (-top|-right|-left|bottom): solid|...
border-color: blue Couleur de la bordure
padding (-top|-right|-left|bottom): 1px Remplissage interne
display: block|inline|none à la ligne suivante|à la suite|caché

Compléments sur les styles

- Propriétés concernant les listes

list-style-type:
- pour les listes non numérotées : **square|circle|disc**
- pour les listes numérotées :
  - **decimal** 1, 2, 3, ...
  - **decimal-leading-zero** 01, 02, 03, ...
  - **lower-roman** i, ii, iii, ...
  - **upper-roman** I, II, III, ...
  - **lower-alpha** a, b, c, ...
  - **upper-alpha** A, B, C, ...

list-style-image: <images/puce.gif>;
- pour utiliser une image comme puce

list-style-position: outside|inside;
- pour positionner la puce dans le paragraphe ou non

Compléments sur les styles

- Appliquer des styles à certains états : les pseudo-classes et pseudo-éléments

```
a:link {color: red;}
a:visited {color: blue;}
a:active {color: green;}  // la souris clique sur le lien
a:hover {color: yellow;} // la souris passe sur le lien
p:first-line {color: red;}
p:first-letter {font-size: 200%;}
input:focus, textarea:focus {font-weight: bold;}
```

- Autre méthode pour définir une couleur

```
color: rgb(0, 255, 0);
color: rgb(25%, 50%, 25%);
```


Les balises <SPAN> et <DIV>

- Idée : attribuer des définitions de style à un ensemble d'objets (texte, image, formulaire...)
- La balise <SPAN> permet de définir un bloc "in-line" de texte (ou **conteneur**) auquel s'appliquera le style référencé par l'attribut CLASS et/ou ID
- La balise <DIV> permet aussi de définir un bloc ou conteneur mais
  - les blocs <DIV> se placent les uns en dessous des autres et ont une dimension
  - les blocs <SPAN> se placent les uns à côté des autres et ont la dimension qu'occupe leur contenu
- Attributs **class** pour CSS, **id** pour Javascript

Les balises <SPAN> et <DIV>

```
<!-- 3.html -->
<html><head>
        <style type="text/css">
        .noirsurblanc {background-color: white; color: black;}
        .blancsurnoir {background-color: black; color: white;}
        #rouge {color: red;}
        </style>
</head><body>
        <span class="noirsurblanc">
                HTML<br>
                <span class="blancsurnoir" ID="rouge">
                        et<br>
                </span>
                <span class="blancsurnoir">
                        STYLES<br>
                </span>
                c'est facile !<br>
        </span>
</body></html>
```

Positionner un bloc en CSS

- Un bloc se positionne par rapport à son conteneur (le conteneur initial est défini par <body>)
  - en précisant des marges

```
<!-- 6.html -->          /* height: automatic; width: 80%; */
<html><head>
        <style type="text/css">
        div.conteneur {
                background-color: blue;
                height: 100px; width: 100px;}
        div.bloc1 {
                background-color: yellow;
                height: 50px;width: 50px;
                margin-left: 20px;margin-top: 40px;}
        </style>
</head><body>
        <div class="conteneur" id="conteneur">
                <div class="bloc1" id="bloc1">bloc1</div>
        </div>
</body></html>
```

Positionner un bloc en CSS

- La propriété **float** permet de positionner un bloc à droite ou à gauche dans son conteneur, le reste du conteneur occupera l'espace restant

```
<!-- 7.html -->
<html><head>
        <style type="text/css">
        div.conteneur {
                background-color: yellow;
                height: 200px;width: 200px;}
        img.bloc1 {
                float: left;}
        </style>
</head><body>
        <div class="conteneur" id="conteneur">
                <img class="bloc1" id="bloc1" src="moi_120.JPG" />
                bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla
                bla bla bla bla bla bla bla bla bla bla
        </div></body></html>
```


Positionner un bloc en CSS

- En utilisant des positions absolues/relatives
  - **position: relative;** --> positionnement par rapport à l'élément précédent
  - **position: absolute;** --> positionnement par rapport au coin supérieur gauche du conteneur

  top: Npx; --> distance depuis le bord supérieur

  left: Npx; --> distance depuis le bord gauche
- Autres unités de mesures que `px` : `mm`, `cm`, `pt`, ...
- Autres propriétés possibles après `position` :
  - rendre le bloc visible ou invisible

  visibility: visible|hidden; (ne s'applique qu'aux blocs)
  - superposer des blocs

  z-index: 1|2|3...;

Positionner un bloc en CSS

```
<!-- 8.html -->
<html><head>
        <style type="text/css">
        div.conteneur {
                background-color: blue;
                height: 200px;width: 200px;
                position: absolute; top: 20px; left:40px;}
        div.bloc1 {
                background-color: yellow;
                height: 50px;width: 50px;
                position: absolute; top: 50px; left:100px;}
        </style>
</head><body>
        <div class="conteneur" id="conteneur">
                bla bla bla bla bla bla bla bla bla bla bla bla
                <div class="bloc1" id="bloc1">bloc 1</div>
        </div>
</body></html>
```


Positionner un bloc en CSS

```
<!-- 8bis.html -->
<html><head>
        <style type="text/css">
        div.conteneur {
                background-color: blue;
                height: 200px;width: 200px;
                position: absolute; top: 20px; left:40px;}
        div.bloc1 {
                background-color: yellow;
                height: 50px;width: 50px;
                position: relative; display: inline;}
        </style>
</head><body>
        <div class="conteneur" id="conteneur">
                bla bla bla bla bla bla bla bla bla bla bla bla
                <div class="bloc1" id="bloc1">bloc 1</div>
        </div>
</body></html>
```


Positionner un bloc en CSS

```
<!-- 8ter.html -->
<html><head>
        <style type="text/css">
        div.bloc1 {
                background-color: blue;
                height: 200px;width: 200px;
                position: absolute; top: 20px; left:40px; z-index: 2;}
        div.bloc2 {
                background-color: yellow;
                height: 50px;width: 50px;
                position: absolute; top: 20px; left:40px; z-index: 1;}
        </style>
</head><body>
        <div class="bloc1" id="bloc1">bloc 1</div>
        <div class="bloc2" id="bloc2">bloc 2</div>
</body></html>
```


XML et CSS

- On peut définir un style pour chaque balise XML

<!-- liste_cours_css.xml -->

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="cours.css"?>
<cours_olivier>...</cours_olivier>
```

<!-- cours.css -->

```
cours_olivier { };
cours { };
auteur {background-color: white;};
titre {display: block; font-family: Arial; font-style: italic;};
niveau { font-family: 'Courrier New'; font-weight: bold;};
```

Pour le TP : JavaScript et les images

Slide 88 Partie 2 (JavaScript)

```
<!-- ex8.html -->
<HTML><HEAD><SCRIPT LANGUAGE="JavaScript">
 var img = new Image(400,400);
 // permet d'accélérer le chargement de l'image
 // (pré-chargement de img1.gif dans le cache du navigateur)
 img.src = "img1.gif"
</SCRIPT></HEAD><BODY>
<IMG src="img0.gif" width="100" height="100"
    onMouseOver="this.src='img1.gif';this.width=400;this.height=400;"
    onMouseOut="this.src='img2.gif';this.width=200;this.height=200;">
</BODY></HTML>
```

La propriété *document* de window (6)

Slide 68 Partie 2 (JavaScript)

```
<!-- index.html -->
<HTML><HEAD>
  <SCRIPT LANGUAGE="JAVASCRIPT" SRC="function.js">
  </SCRIPT>
</HEAD><BODY>
  <div id="title1" style="position:absolute; top:50; left:20;">
    <h1>DHTML...</h1>
  </div>
  <div style="position:absolute; top:10; left:250;">
    <a href="javascript:changeText();">Modifier le texte</a><br>
    <a href="javascript:changeColor();">Modifier la couleur</a><br>
    <a href="javascript:changePosition();">Déplacer le bloc</a>
    <a href="javascript:AfficheBloc();">Affiche le bloc</a>
    <a href="javascript:MasqueBloc();">Masque le bloc</a>
  </div>
  <script language="JavaScript">
     var elm = document.getElementById('title1');
  </script>
</BODY></HTML>
```