Technologies du Web

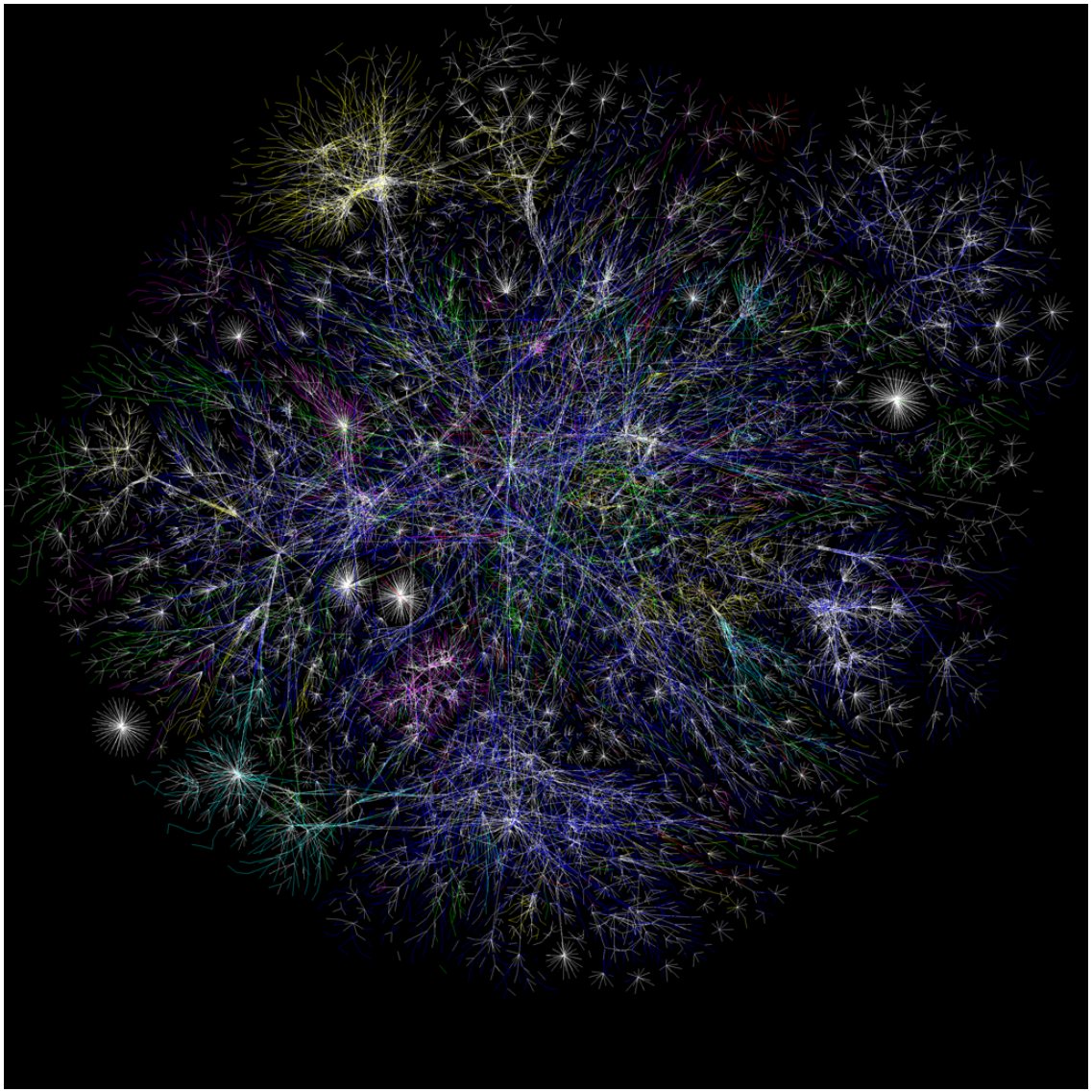
Introduction au World Wide Web et à HTML

Pierre Senellart (pierre.senellart@telecom-paristech.fr)

Mastère spécialisé *Management et nouvelles technologies*,
28 septembre 2009

Plan du cours

Un réseau de réseaux : ordinateurs interconnectés



http://www.opte.org/

La pile de protocoles Internet

Une pile de protocoles de communication, l'un au-dessus de l'autre

Application	HTTP, FTP, SMTP, DNS	
Transport	TCP, UDP, ICMP	(sessions, fiabilité...)
Réseau	IP (v4, v6)	(routage, adressage)
Lien	Ethernet, 802.11 (ARP)	(adressage de machines locales)
Physique	Ethernet, 802.11 (physique)	

IP (Internet Protocol)

- Adressage de machines et routage sur Internet
- Deux versions du protocole IP utilisées sur Internet : IPv4 (très répandu) et IPv6 (support encore un peu expérimental)
- IPv4 : Adresses de 4 octets affectées à chaque ordinateur, p. ex., 137.194.2.24. Des gammes de telles adresses sont données aux diverses institutions (fournisseurs d'accès à Internet, universités, etc.), pour les affecter comme elles le souhaitent.
- Problème : seulement 2^{32} adresses possibles (en fait, un grand nombre d'entre elles ne peut pas être données à de nouveaux hôtes, pour diverses raisons). Cela signifie que beaucoup de machines connectées à Internet n'ont pas d'adresse IPv4 et une forme de traduction d'adresse (network address translation, NAT) est nécessaire.
- IPv6 : Adresses de 16 octets ; espace d'adressage beaucoup plus grand ! Les adresses ont la forme suivante : 2001:660:330f:2::18 (soit 2001:0660:0330f:0002:0000:0000:0000:0018). Chacun des appareils connecté à l'Internet peut avoir sa propre adresse IP !

TCP (Transmission Control Protocol)

- L'un des deux protocoles de transport utilisés sur IP, avec UDP (User Datagram Protocol)
- Contrairement à UDP, fournit une transmission fiable des données (acknowledgments)
- Les données sont divisées en petits datagrammes qui sont envoyés sur le réseau, et éventuellement réordonnés à destination
- Comme UDP, chaque transmission TCP indique un numéro de port source et destination (entre 0 and 65535) pour la distinguer du reste du trafic
- Un client sélectionne usuellement un numéro de port aléatoire pour établir une connexion à un numéro de port fixe du serveur
- Le numéro de port du serveur identifie de manière conventionnelle un protocole d'application au-dessus de : 22 pour SSH, 25 pour SMTP, 80 pour HTTP, 110 pour POP3...

DNS (Domain Name System)

- Les adresses IPv4 sont difficiles à mémoriser et un service donné (p. ex., un site Web) peut changer d'adresses IP (p. ex., nouveau fournisseur d'accès à Internet)
- Problème accentué pour les adresses IPv6 !
- DNS : un protocole basé sur UDP/IP pour associer des noms lisibles par les humains (p. ex., `www.google.com`, `weather.yahoo.com`) à des adresses IP
- Noms de domaines hiérarchiques : `com` est un domaine de niveau principal (top-level domain, TLD), `yahoo.com` un sous-domaine de celui-ci, etc.
- Résolution hiérarchique de nom de domaine : serveurs racines avec des IP fixes savent qui est responsable des TLDs, les serveurs en charge d'un domaine savent qui est responsable d'un sous-domaine, etc.
- Rien de magique dans `www.google.com` : juste un sous-domaine de `google.com`.

Généralités

- Qu'est-ce que le Web (ou World Wide Web, Toile, WWW, W3) ?
  - Système hypertexte public : système contenant des documents liés entre eux par des hyperliens permettant de passer automatiquement d'un document à l'autre.
- Différence entre le Web et Internet ?
  - Internet : réseau mondial d'ordinateurs permettant aux utilisateurs de communiquer (courrier électronique), de publier des informations (Web), de transférer des données (FTP), de travailler à distance (telnet et ssh)...
  - Web : un aspect d'Internet.

Architecture client-serveur

Le client (navigateur : Internet Explorer, Firefox, Safari...)

- demande au serveur des informations
- affiche des pages pour l'utilisateur

Le serveur (Apache, Microsoft IIS...)

- reçoit en permanence les requêtes des clients
- renvoie les documents correspondants

HTTP

Protocole Ensemble normalisé de règles décrivant la manière de transmettre des informations, par exemple sur un réseau comme Internet entre un client et un serveur.

HTTP HyperText Transfer Protocol, le plus utilisé des protocoles de communication sur le World Wide Web. Permet à un client Web d'indiquer quelle page il veut obtenir, et au serveur Web de lui répondre en lui donnant cette page.

URL

- URL : *Uniform Resource Locator*
- Identifie l'endroit où se trouve une ressource sur le Web.
- Dans le cas du Web, ressource = document ou fragment
- `http://lea-linux.org:80/reseau/secu/firewall.html#intro`
  protocole machine port répertoire fichier fragment
- Principaux protocoles utilisés dans les URL :

  http, https deux protocoles du Web, sans et avec chiffrement et authentification ;

  ftp protocole de transfert de fichier, parfois utilisé sur le Web pour le téléchargement de gros fichiers ;

  mailto pseudo-protocole dénotant une adresse e-mail.
- Port par défaut : 80 pour HTTP, 443 pour HTTPS

Clients Web

- Navigateurs graphiques (cf. transparent suivant)
- Navigateurs textuels : w3m, lynx, links (libres, Windows, Mac OS, Linux, Unix) ; leur usage n'est guère plus répandu, mais ils simulent assez bien ce que « voit » un robot ou un navigateur auditif
- Autres navigateurs : navigateurs auditifs, etc.
- Mais aussi : robots des moteurs de recherche, logiciels de traduction automatique. . .

Clients Web

- Navigateurs graphiques (cf. transparent suivant)
- Navigateurs textuels : w3m, lynx, links (libres, Windows, Mac OS, Linux, Unix) ; leur usage n'est guère plus répandu, mais ils simulent assez bien ce que « voit » un robot ou un navigateur auditif
- Autres navigateurs : navigateurs auditifs, etc.
- Mais aussi : robots des moteurs de recherche, logiciels de traduction automatique. . .

Une très grande variété de clients ! En théorie, tout site Web doit être testé avec la plupart de ceux-ci (et au moins les plus répandus), dans leurs différentes versions.

Navigateurs graphiques

Navigateur	Moteur	Part	Distribution
Internet Explorer	Trident	65%	distribué avec Windows
Firefox	Gecko	25%	libre, Windows, Mac OS, Linux, Unix
Safari	WebKit	5%	distribué avec Mac OS
Google Chrome	WebKit	2%	libre, Windows
Opera	Presto	2%	gratuit, Windows, Mac OS, Linux, mobiles
Netscape	Gecko	$<1\%$	gratuit, Windows, Linux, Mac OS
Konqueror	KHTML	$<1\%$	libre, Linux, Mac OS, Unix
Camino	Gecko	$<1\%$	libre, Linux, Mac OS, Unix

Parts de marché : diverses sources, chiffres précis difficiles à obtenir. Firefox est beaucoup plus répandu en Europe ($\sim$ 30-35%) qu'aux États-Unis. Internet Explorer dominant, mais en baisse assez nette depuis quelques années.

Moteurs de rendus :

Trident est (traditionnellement) le moteur supportant le moins bien les standards du Web. En évolution.

WebKit et KHTML ont beaucoup en commun, puisque WebKit est issu d'un fork de KHTML.

Actualité des navigateurs graphiques

- Sortie récente de Firefox 3.5 (juin 2009) et d'Internet Explorer 8 (mars 2009).
- Internet Explorer 6, 7 et 8 utilisés à peu près à égalité (IE6 très courant en entreprise). Firefox 3.0 est encore devant Firefox 3.5, mais les versions antérieures sont négligeables.
- Google Chrome a a peine un an, et a déjà une part de marché non négligeable. Des versions Linux et Mac OS sont annoncées en développement.

Vocabulaire

serveur Web : soit le logiciel qui répond aux requêtes d'un client, soit la machine sur lequel fonctionne ce serveur.

site Web : ensemble de pages Web et d'éventuelles autres ressources, liées dans une structure cohérente, publiées par un propriétaire (une entreprise, une administration, une association, un particulier, etc.) et hébergées sur un ou plusieurs serveurs Web.

hébergeur Web : entreprise de services informatiques hébergeant (mettant en ligne) sur ses serveurs Web les ressources constituant les sites Web de ses clients.

Plan du cours

HyperText Markup Language

- normalisé par le W3C (World Wide Web Consortium) regroupant industriels (Microsoft, Google, Apple...) et académiques (INRIA, MIT...)
- format ouvert : lecture possible dans des conditions correctes sans contrainte matérielle ou logicielle
- un fichier texte avec des balises
- description de la structure et du contenu d'un document, accent sur l'accessibilité
- on ne décrit pas la présentation (ce sera le rôle de CSS)
- on ne décrit pas de comportement dynamique (ce sera le rôle de JavaScript et des langages côté serveur)

- HTML est un langage qui alterne texte et balises (`<blabla>` ou `</blabla>`)
  - Les balises permettent de structurer chaque partie du document et servent par exemple au navigateur pour réaliser la mise en page du document.
- Les fichiers HTML
  - sont structurés en deux parties principales : l'en-tête (`<head> ... </head>`) et le corps (`<body> ... </body>`)
- En HTML, les blancs (espace, tabulations, retours à la ligne) sont en général équivalents et servent juste à délimiter mots, balises... Leur nombre n'a pas d'importance.

Balises

- Leur syntaxe est (balises ouvrante et fermante)

```
<balise attributs>contenu</balise>
```

ou (balise sans contenu)

```
<balise attributs>
```

balise : mot clé désignant un élément particulier

contenu : peut contenir du texte ou d'autres balises

attributs : représente les différents paramètres associés à l'élément, sous la forme d'une liste de `nom="valeur"` ou `nom='valeur'`, séparés par des espaces (les guillemets ne sont pas toujours indispensables, mais elles le deviennent dès que `valeur` contient des caractères exotiques)

Balises

- Les noms des éléments et des attributs sont souvent écrits en minuscule, mais `<head>` et `<HeAd>` sont équivalents.
- Les balises sont ouvertes et refermées dans l'ordre (`<b><i></i></b>` et non `<b><i></b></i>`).
- Des règles strictes déterminent quelles balises peuvent être mises à l'intérieur de quelles balises.
- Sous certaines conditions, une balise peut être implicitement refermée, mais ces conditions sont assez complexes à décrire.
- `<!--zut.-->` dénote un commentaire, qui ne sera ni affiché ni interprété par le client Web (utile pour documenter une partie d'une page Web).

Balises : exemples

Exemples

- `<title>coucou</title>` pour attribuer le titre *coucou* au document
- `<em>cuicui</em>` pour mettre en *emphase* le texte *cuicui* (cela sera rendu, le plus souvent, par une mise en italique).
- `<strong>cuicui</strong>` pour indiquer que le texte **cuicui** est important (cela sera rendu, le plus souvent, par une mise en gras).

Contre-exemple

```
<strong><em>bouh</strong></em>
```

Structure d'un document

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html lang="fr">
  <head>
    <!-- En-tête du document -->
  </head>
  <body>
    <!-- Corps du document -->
  </body>
</html>
```

- La déclaration `<!DOCTYPE ...>` précise la version d'HTML utilisée.
- La langue du document est précisée avec l'attribut `lang` de la balise principale `<html>`.

En-tête

- L'en-tête du document est délimitée par les balises `<head> ... </head>`.
- L'en-tête contient des méta-informations concernant le document telles que son titre, son encodage, les fichiers annexes, etc. Les deux informations les plus importantes sont :
  - Le jeu de caractères de la page, à mettre tout au début de l'en-tête

```
<meta http-equiv="Content-Type"
      content="text/html; charset=utf-8">
```

  - Le titre de la page (la seule information obligatoire à préciser) ; celui-ci sera par exemple affiché dans la barre de titre du navigateur, il n'apparaît pas dans la page elle-même.

```
<title>Mon super site</title>
```

Jeux de caractères

Unicode : répertoire de caractères, assignant à chaque caractère, de quelque langue que ce soit, un nombre entier.

Exemples

A	$\longrightarrow$	65	ε	$\longrightarrow$	949
é	$\longrightarrow$	233	$\aleph$	$\longrightarrow$	1488

Jeu de caractères : moyen de représenter concrètement, par une suite de 0 ou de 1, un caractère Unicode.

Exemples (é)

iso-8859-1	11101001		Seulement pour certains caractères
utf-8	11000011	10101001	
utf-16	11101001	00000000	

utf-8 présente l'avantage de pouvoir représenter tous les caractères d'Unicode, de manière compatible avec l'ancien encodage ASCII.

- Les balises `<body> ... </body>` délimitent le corps du document.
- Le corps est structuré en sections, paragraphes, listes, etc.
- Il existe 6 balises permettant de représenter les titres de sections, par importance décroissante :
  - `<h1>Titre de la page</h1>`
  - `<h2>Titre de section principale</h2>`
  - `<h3>Titre de sous-section</h3>`
  - `<h3>Titre de sous-sous-section</h3>`
  - ...
- Les balises `<p> ... </p>` permet de délimiter un paragraphe. Tous les paragraphes de texte doivent être balisés ainsi.
- La balise `<hr>` (horizontal rule) indique une séparation majeure dans le document (rendue par exemple graphiquement par une ligne horizontale).
- Directement à l'intérieur de `<body> ... </body>` n'apparaissent que des balises de bloc : `<p>`, `<h1>`, `<form>`, `<hr>`, `<ul>`, `<table>` ainsi que la balise `<div>` qui dénote un bloc sans sémantique particulière.

Listes

- XTHML possède plusieurs balises permettant de présenter le texte sous forme de listes.
- On en distingue trois types :
  - les listes non numérotées,
  1. les listes numérotées,

  les listes de définitions (ou lexiques)
- Ces listes peuvent être emboîtées les unes à l'intérieur des autres.

- Les listes classiques :
  - Les listes non numérotées délimitées par les balises `<ul> ... </ul>` (unordered list).
  - Les listes numérotées délimitées par les balises `<ol> ... </ol>` (ordered list).
  - Tous les éléments d'une liste numérotée ou non sont délimités par les balises `<li> ... </li>` (list item)
- Les lexiques sont délimités par les balises `<dl> ... </dl>` (definition list) et leurs entrées par les balises `<dt> ... </dt>` (term) et `<dd> ... </dd>` (definition).

Exemples

```
<ol> <li>un</li> <li>deux</li> </ol>

<dl> <dt>lapin</dt> <dd>rongeur à oreilles</dd> </dl>
```

Tableaux

- Les tableaux sont délimités par les balises `<table>...</table>`.
- Les balises `<tr>...</tr>` (table row) délimitent les lignes.
- Les balises `<td>...</td>` (table data) délimitent les cellules.
- Attention ! On déclare les lignes à l'intérieur du tableau, les cellules à l'intérieur des lignes.

Exemple

```
<table>
  <tr> <td> l1, c1 </td> <td> l1, c2 </td> </tr>
  <tr> <td> l2, c1 </td> <td> l2, c2 </td> </tr>
</table>
```

Ajouter de la structure à un tableau en :

- donnant une légende au tableau avec les balises `<caption>... </caption>` juste après la balise ouvrante `<table>`.
- remplaçant les `<td> ... </td>` qui contiennent des en-têtes (de ligne, de colonne) par des `<th> ... </th>` (table header).

Images

- Pour insérer une image dans un document HTML, on utilise la balise `<img>`.
  - L'attribut `src` permet de préciser où se trouve l'image.
  - L'attribut `alt` permet de remplacer l'image par un texte quand elle n'est pas disponible. Il est obligatoire de l'utiliser, pour que tout agent (malvoyants, navigateur texte, incidents techniques, robots) ne pouvant voir votre image puisse avoir un texte alternatif.

```
<img src="http://www.lri.fr/images/lri.png" alt="LRI">
<img src="../images/montblanc.png" alt="Mont Blanc">
```

- Les formats d'images utilisables pour le Web sont :
  - Le JPEG (.jpg), un format adapté aux photos.
  - Le GIF (.gif) et le PNG (.png), des formats adaptés aux autres types d'image ; le PNG est à préférer dans tous les cas (transparence, profondeur de couleurs...) sauf besoin d'images animées (à utiliser avec parcimonie !).

Liens

- Ce qui différencie une page Web (page HyperTexte) d'un banal document : ce sont les liens !
- Ils sont introduits par la balise `<a> ... </a>` (cette balise est une balise en ligne, il faut la placer à l'intérieur d'un bloc).
- En cliquant sur un lien, on peut se déplacer vers :
  - un autre serveur ou un fichier du même serveur
  - une autre partie du même document

- Pour faire un lien, on utilise l'attribut `href` de la balise `<a>` dont le contenu formera le lien :

```
<a href="http://www.cnrs.fr/">
  <img src="images/cnrs.gif" alt="CNRS">
</a>

<a href="bio/indexbioinfo.html">Bioinformatique</a>
```

- Les ancres servent à atteindre un endroit précis dans le document.
  - On commence par définir les ancres, soit sur une balise existant déjà grâce à l'attribut `id`, soit avec un `<a id="...">` :

```
<h3 id="tutorials">Tutorials</h3>
<a id="tutorials">
```

  - Ensuite, on fait le lien avec cette ancre.

```
<a href="#tutorials">tutorials</a>
<a href="http://www.w3.org/#tutorials">tutorials</a>
```

  - On voit parfois l'ancienne syntaxe `<a name="...">`.

Et aussi...

- De nombreuses autres balises en ligne : `<abbr>`, `<cite>`, `<var>`...
- Quelques autres balises blocs : `<blockquote>`, `<address`...
- Représentation des caractères spéciaux (ex., « é ») :
  - directement dans le codage du document (utf-8 de préférence, pour représenter tous les caractères possibles), à privilégier ;
  - par leur code en décimal (é) ou en hexadécimal (é) ;
  - par des entités caractères nommées (é)

Plan du cours

Formulaires

- Permettent d'interagir avec l'utilisateur en lui proposant d'entrer des informations
- En HTML : uniquement l'interface de formulaire
- L'essentiel du travail sera fait par le script qui traitera la soumission du formulaire

Balise <form>

- Un formulaire HTML est placé à l'intérieur d'une balise `<form>`.
- Celle-ci prend les attributs suivants :

action	URL du script auquel sera soumis le formulaire.
method	Méthode HTTP, valant soit `"get"` soit `"post"`.
enctype	Encodage HTTP. Peut valoir `"application/x-www-form-urlencoded"` (valeur par défaut) ou `"multipart/form-data"`.

Example (Formulaire élémentaire)

```
<form action="action.php" method="get">
  <div><input type="submit"></div>
</form>
```

Ensembles de champ

En HTML, il est interdit de mettre des champs de formulaire directement à l'intérieur d'un `<form>`. Il faut d'abord les regrouper :

- Dans des paragraphes `<p>` si les champs de formulaires sont à l'intérieur de paragraphes de textes (rare).
- Dans des ensembles de champ `<fieldset>` pour regrouper des champs de formulaire de sémantique proche. On pourra alors donner une légende à l'ensemble de champs avec la balise `<legend>`.
- Dans des divisions `<div>` sans contenu sémantique sinon.

Exemple (Ensemble de champ)

```
<fieldset>
  <legend>Mensurations</legend>
  <input type="text" name="taille">
  <input type="text" name="poids">
</fieldset>
```

Étiquettes

- La plupart des champs sont naturellement accompagnés d'une étiquette (`<label>`).
- On peut la placer où on veut, en général juste à gauche ou à droite du champ.
- Son attribut `for` référence l'attribut `id` du champ correspondant.
- Dans les navigateurs graphiques, un clic sur l'étiquette d'un champ permet en général de sélectionner le champ.

Example (Étiquette)

```
<label for="taille">Taille :</label>
<input type="text" name="taille" id="taille">
```

Champs de saisie

- La balise `<input>` a une utilisation très vaste dans les formulaires. Elle représente un champ de saisie.
- L'attribut `type` détermine le type (texte, mot de passe, liste, etc.) du champ.
- L'attribut `name` (nom du paramètre de la requête HTTP) est obligatoire (sauf pour les types `"reset"` et `"submit"`) ; il permet de préciser au serveur à quelle saisie on fait référence.

Exemple (Zone de texte pour écrire un commentaire)

```
<input type="text" name="Commentaire">
```

Saisie d'une ligne de texte

- `type ="text"` est utilisé pour la saisie d'un texte dont la taille est inférieure à une ligne.
- L'attribut `value` permet de préciser la valeur par défaut (facultatif).
- La taille maximale de la chaîne de caractères à saisir peut être spécifiée à l'aide de l'attribut `maxlength` (facultatif).

Exemple

```
<input type="text" name="prenom" value="Jordy" maxlength="50">
```

Saisie d'un mot de passe

- `type ="password"` est utilisé pour la saisie d'un texte dont les caractères sont remplacés par des astérisques : c'est généralement utilisé pour la saisie des mots de passe. Le mot de passe est quand même transmis en clair au serveur !
- L'attribut `value` permet de préciser la valeur par défaut (facultatif).
- La taille maximale de la chaîne de caractères à saisir peut être spécifiée à l'aide de l'attribut `maxlength` (facultatif).

Exemple

```
<input type="password" name="pwd" value="12345678">
```

Choix multiples parmi une liste

- `type ="checkbox"` permet de choisir plusieurs éléments parmi une liste de possibilités.
- Cela se matérialise sous forme de cases à cocher.
- La valeur retournée est obligatoirement précisée à l'aide de l'attribut `value`.
- L'attribut `checked ="checked"` permet de cocher la case par défaut.
- Certains langages côté serveurs imposent que les noms de champs de formulaire à choix multiples se terminent par [].

Exemple

```
<input type="checkbox" name="pub[]" value="site"
  checked="checked" id="pub-site">
<label for="pub-site">Recevoir des offres de notre site</label>

<input type="checkbox" name="pub[]" checked="checked"
  value="externe" id="pub-externe">
<label for="pub-externe">Recevoir des offres externes</label>
```

Choix unique parmi une liste

- `type ="radio"` permet de choisir un seul élément parmi une liste de possibilités.
- Cela se matérialise sous forme de boutons radio.
- La valeur retournée est obligatoirement précisée à l'aide de l'attribut `value`.
- L'attribut `checked ="checked"` permet de préciser la valeur par défaut.

Exemple

```
Recevoir de la pub:
<input type="radio" name="pub" value="oui" id="pub-oui"
  checked="checked">
<label for="pub-oui">oui</label>

<input type="radio" name="pub" value="non" id="pub-non">
<label for="pub-non">non</label>
```

Fichiers joints

- `type ="file"` permet de joindre au formulaire un fichier.
- À cause de la taille de la requête due au téléchargement (upload) du fichier, il faut impérativement utiliser la méthode POST et l'encodage `multipart/form-data`.

Exemple

```
<label for="fichier">Fichier:</label>
<input type="file" name="fichier" id="fichier">
```

Champs cachés

- `type ="hidden"` permet de cacher des champs au client mais leur contenu est envoyé avec le formulaire.
- Ceci permet de préciser des informations, en utilisant l'attribut `value`, concernant l'interaction client/serveur.
- C'est à utiliser avec précaution car cela peut être à l'origine de problèmes de sécurité assez graves : ne pas oublier que le client peut éditer la page à la main pour changer la valeur de ces champs !

Exemple

```
<input type="hidden" name="monnaie_utilisee" value="EUR">
<input type="hidden" name="customerCB" value="c2415-345-8563">
```

Ré-initialisation d'un formulaire

- `type ="reset"` permet de réinitialiser le formulaire en affectant aux différents champs leur valeur par défaut.
- L'attribut `value` permet de changer le texte du bouton correspondant.

Exemple

```
<input type="reset" value="Tout effacer">
```

Soumettre le formulaire

- `type ="submit"` permet de soumettre le formulaire.
- Le client envoie le contenu du formulaire à l'adresse précisée par l'attribut `action` de la balise `<form>`.
- L'attribut `value` permet de changer le texte du bouton correspondant.

Exemple

```
<input type="submit" value="Envoyer">
```

Saisie de plusieurs lignes de texte

- Pour les saisies multiligne, on utilise la balise `<textarea>`.
- Le texte délimité par cette balise permet d'initialiser la valeur par défaut du champ.
- La balise fermante est obligatoire même si le champ est vide.
- Les attributs `rows` et `cols` (obligatoires) permettent de spécifier la taille en lignes et colonnes de la fenêtre de saisie.

Exemple

```
<textarea name="bio" cols="40" rows="5">
  Fille de Josiane Balasko,
  Marilou Berry fait ses premiers pas au cinéma à 8 ans...
</textarea>
```

Menus de sélection

- La balise `<select>` permet d'ajouter une liste de sélection :
  - L'attribut facultatif `size` permet de préciser le nombre de choix apparaissant sur la page Web. Par défaut, ce nombre est initialisé à 1.
  - L'attribut `multiple ="multiple"` permet d'autoriser des réponses multiples. Dans ce cas, pour PHP, on donnera toujours un nom se terminant par `[]`.
- Les choix du menu sont indiqués à l'aide de la balise `<option>` :
  - L'attribut `selected ="selected"` permet de spécifier le(s) choix par défaut.
  - L'attribut `value` permet de spécifier la valeur associée au choix.

Exemple

```
<select name="age">
  <option value="20">Moins de 20 ans</option>
  <option value="35" selected="selected">21 à 35 ans</option>
  <option value="50">36 à 50 ans</option>
  <option value="51">Plus de 51 ans</option>
</select>
```

Plan du cours

Les différentes version d'HTML

- HTML 4.01 (1999) strict (vu plus haut) et transitionnel

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

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

- XHTML 1.0 (2000) strict et transitionnel

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- XHTML 1.1 (en cours) : principalement un échec, inutilisable et inutilisé sur le Web actuel
- XHTML 2.0 et HTML 5 : deux normes concurrentes en cours d'élaboration

Soupe de balises

- On trouve encore beaucoup de documents HTML antérieurs à HTML 4.01
- En pratique : beaucoup de pages Web ne respectent aucun de ces standards (avec ou sans déclaration de type de document) $\Longrightarrow$ navigateurs ne respectent pas ces standards $\Longrightarrow$ soupe de balises !
- Pour de nouveaux développements : absolument indispensable de respecter les standards du Web (compatibilité avec les clients Web d'aujourd'hui et de demain), même si ce n'est pas suffisamment compris par les concepteurs de sites Web

HTML contre XHTML

- XHTML : une application d'XML
- Les balises sans contenus `<hr>`, s'écrivent `<hr />` en XHTML.
- Certains éléments peuvent ne pas être refermées en HTML (`<ol><li>un <li>deux </ol>`), mais la fermeture est obligatoire en XHTML.
- Les valeurs des attributs peuvent ne pas être entre guillemets (`<img src=toto.png alt=toto>`) en HTML, guillemets obligatoires en XHTML.
- Les noms des éléments et des attributs sont insensibles à la casse en HTML (`<HTMl laNg=fr>`), contrairement à XHTML ou tout doit être en minuscule.
- Attributs `xmlns` et `xml:lang` sur la balise `<html>` en XHTML.
- Et quelques autres petites subtilités...

Avantages respectifs de HTML et XHTML

Avantages de HTML 4.01

- Meilleur support par les navigateurs (Internet Explorer 6/7 comprennent mal XHTML).
- Moins de contraintes...
- Beaucoup plus utilisé sur le Web.

Avantages de XHTML 1.0

- Plus de contraintes... (donc plus simple !).
- Syntaxe claire, sans ambiguïté.
- Familiarité avec XML, utile dans d'autres contextes.
- Facilité d'utilisation dans des contextes XML (p.ex. XSLT).

Règles de compatibilité à respecter pour du XHTML

- Théoriquement, `<p>` et `<p></p>` sont synonymes en XML, donc en XHTML. En pratique, on utilisera la notation `<balise>` uniquement pour les balises n'ayant jamais de contenu (p.ex. `<br>`, `<hr>` ...).
- Théoriquement, `<br>` et `<br/>` sont synonymes. En pratique, on utilisera toujours `<br>`.
- Théoriquement, un document XHTML peut commencer par une ligne `<?xml version="1.0"encoding="utf-8"?>`. En pratique, on l'omettra.
- cf. `http://www.w3.org/TR/xhtml1/#guidelines`
- On peut utiliser `http://qa-dev.w3.org/~bjoern/appendix-c/validator/` pour vérifier le respect de ces quelques règles.

Strict contre Transitionnel

- À la fois en HTML 4.01 et en XHTML 1.0, on a le choix entre une déclaration de type de document Strict ou Transitional
- Différence : moins d'éléments autorisés en Strict
- Des éléments de présentation (`<u>` pour soulignement, `<center>`, `<font>`) en Transitional, héritage des anciennes versions d'HTML
- À éviter de nos jours : séparation fond/forme, présentation en CSS
- Pour de nouveaux projets : Strict
- Pour des modifications d'anciens sites avec beaucoup de balises présentationnelles : Transitional

XHTML 2.0 contre HTML 5

- XHTML 2.0 : initiative du W3C, incompatible avec HTML 4.01/XHTML 1.0, changements majeurs
- HTML 5 : initiative des développeurs de navigateurs, compatibilité avec HTML 4.01/XHTML 1.0, changements incrémentaux mais nombreux
- XHTML 2.0 abandonné en juillet 2009
- Des fonctionnalités de HTML 5 font leur apparition dans les navigatuers récents (excepté Internet Explorer, qui a cependant annoncé en septembre 2009 son intérêt)
- HTML 5 laisse le choix entre les conventions HTML 4.01 ou XHTML
- Nouvelles fonctionnalités : dessin 2D (`<canvas>`), multimédia (`<audio>` , `<video>`), meilleurs éléments structurants (`<section>` , `footer`), etc.

Plan du cours

Outils

- N'importe quel éditeur de texte (par exemple le bloc-notes de Windows, emacs, vim...). Privilégier les éditeurs avec coloration syntaxique. Bon choix : Scite (libre, léger, simple d'utilisation, multi-plateforme, coloration syntaxique pour HTML/CSS/PHP/JavaScript/...)
- N'importe quel navigateur, et plusieurs navigateurs avec un moteur de rendu différent
- Utiliser la fonction « Afficher la source » des navigateurs Web
- Firefox présente de nombreux avantages pour le développement/analyse de sites Web
- En particulier, extension Firebug de Firefox, excellent outil
- Pour vérifier qu'une page se conforme bien aux normes de HTML, utiliser le validateur du W3C : `http://validator.w3.org/`

Références

- Spécification de HTML 4.01
  `http://www.w3.org/TR/REC-html40/`
- Spécification de XHTML 1.0
  `http://www.w3.org/TR/xhtml1/`
- Brouillon de XHTML 2
  `http://www.w3.org/TR/xhtml2/`
- Brouillon de HTML 5
  `http://www.w3.org/html/wg/html5/`
- *HTML et XHTML : La Référence*, O'Reilly

Plan du cours

Pour chacune des pages Web données dans la liste disponible sur la page du cours :

- Regarder la source. Constatations ? Voit-on apparaître d'autres langages que HTML ?
- Quelle est la variante de HTML utilisée ?
- La page est-elle conforme aux standards du Web ? Utiliser le validateur.