



(X)HTML – CSS

Yannick Prié

UFR Informatique – Université Lyon 1

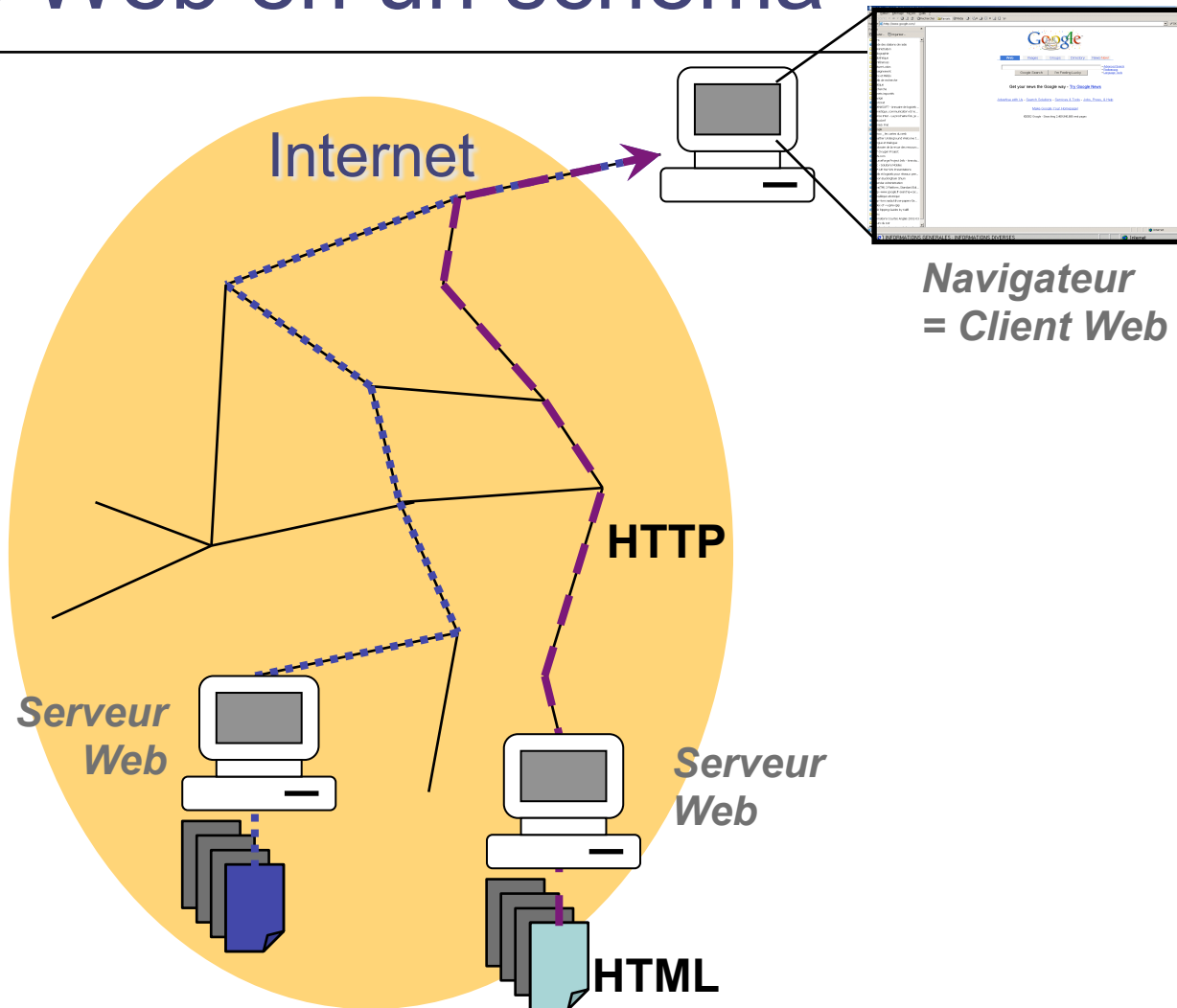
Master SIB M1 – 2008-2009



Objectifs du cours

- Présenter des documents XML dans un navigateur web
 - Historique rapide de HTML
 - XHTML strict
 - CSS

Le Web en un schéma





HTML

- DTD de SGML
- Ensemble d'éléments et d'attributs permettant de définir des documents hypertextes
 - structure, liens, images, tableaux, formulaires...
- Quatre versions
 - HTML 1.0 → HTML 4.01
- Problèmes principaux
 - Mélange structure physique et logique
 - Ex. : un élément pour mettre en gras
 - Syntaxe peu stricte
 - Ex : pas forcément obligatoire de fermer un élément
- Tout cela a une origine historique (guerre des navigateurs), et a paradoxalement contribué au développement massif du web (simplicité, tolérance)



CSS

- Séparation de la structure logique et de la présentation documents HTML
 - Structure logique = HTML = ensemble d'éléments de contenu
 - Présentation suivant une feuille de style (*style sheet*) qui traite les éléments de contenu en éléments de présentation
- Cascading Style Sheets
 - Feuilles de styles associées à HTML (à XML)
 - En cascade
 - on peut utiliser des feuilles de styles multiples
 - il y a un degré d'importance pour chaque feuille de style



XHTML

- XHTML 1.0 (1999)
 - reformulation XML de HTML4
 - définit trois DTD pour exprimer des documents HTML
 - XHTML-1.0-Strict
 - XHTML-1.0-Transitional
 - XHTML-1.0-Frameset
 - sémantique des balises
 - Définie dans HTML4
- XHTML 1.1
 - Modularisation
 - faciliter le mixage de fragments XML dans XHTML
- XHTML 2
 - Modulaire
 - Nouvelles fonctionnalités
 - En cours de discussion
- HTML le retour...
 - HTML 5 : W3C Working Draft (22 January 2008)



HTML → XHTML : à savoir

- De la rigueur...
 - Nom d'éléments en minuscules
 - `<P>` → `<p>`
 - Fermetures correctes : toute balise ouverte doit être fermée
 - `<p> ... </p>`
 - Éléments vides fermés :
 - `<hr />`
 - Un attribut associé à une valeur, entre guillemets
 - `` → ``
 - Attribut id pour identifier un élément (vs name avant)
 - `<h1 id="partie1" name="partie1"> ... </h1>`



Plan

- XHTML
 - Structure générale d'un document XHTML
 - XHTML Strict et Transitionnal
 - Éléments les plus utiles
- CSS
 - Généralités
 - Règles et sélecteurs
 - Propriétés utiles
 - Placement des styles
 - Héritage et cascade



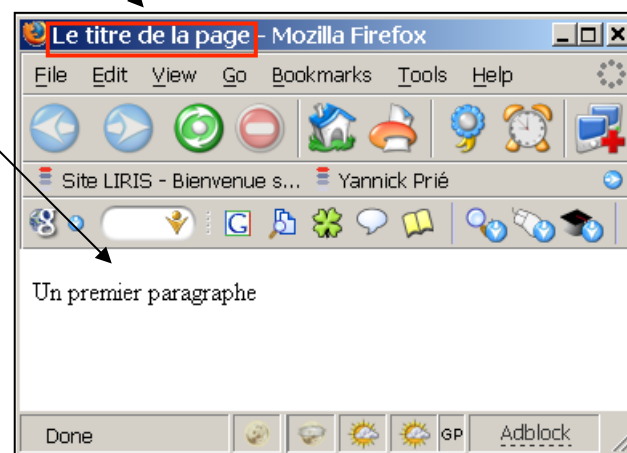
Structure générale d'un document XHTML

- Prologue
 - Déclaration XML
 - Déclaration de DTD
- En-tête
 - Élément **head**
- Corps
 - Élément **body**
- Commentaires n'importe où
 - `<!-- ... -->`

Premier exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Le titre de la page</title>
</head>
<body>
  <!-- un commentaire -->
  <p>Un premier
  paragraphe</p>
</body>
</html>
```

Remarque :
ici le navigateur utilise une feuille de style
par défaut pour afficher du XHTML strict
non lié à une feuille de style CSS.





XHTML Strict et Transitionnel

- Transitionnel

- Les éléments de présentation de HTML sont encore autorisés

- b, center, font, ...

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

- Strict

- Séparation stricte du document XHTML de sa présentation
→ utilisation de style CSS obligatoire

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

- Déclaration d'espace de nom dans la balise ouvrante html

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
```

- Pour des débutants : **XHTML strict + CSS**



En-tête : contenu de l'élément head

- Titre du document
 - `<title></title>`
- Autre informations non affichées à l'écran, utilisées par le navigateur, les moteurs, etc.
 - `<meta name="..." content="..." />`
 - `<meta http-equiv="Refresh" content="4" ; URL=http://www.google.com" />`
 - `<meta name="author" content="" />`
 - `<meta name="Keywords" content="motcle1, motcle2, motcle3"/>`
 - `<meta name="language" content="fr"/>`
 - `<base href="URL-de-base" />`
 - URL de base pour les URL relatives
- Styles
 - `<style />` → inclure une feuille de style CSS dans la page
 - `<link />` → lier le document à une ressource externe (typiquement, feuille de style)
- Scripts
 - `<script />` → ajouter un script à la page

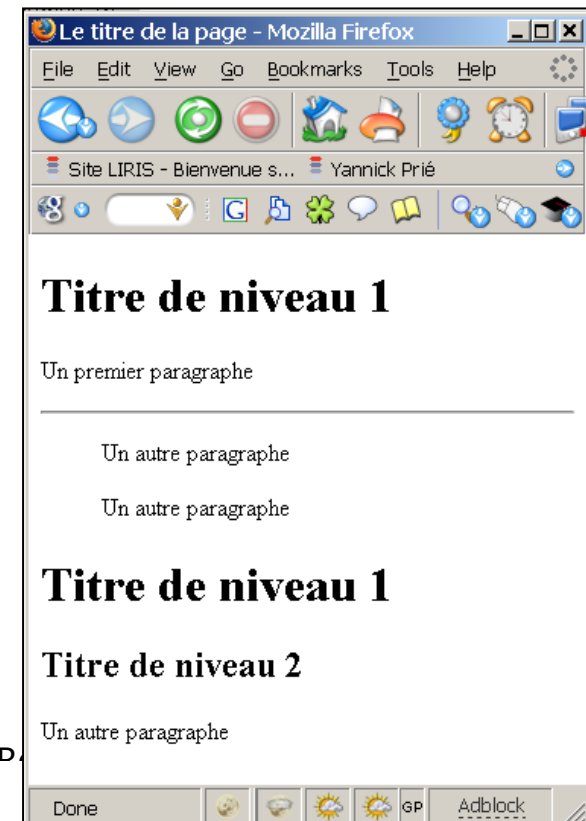


Corps

- Élément body
 - Toutes les informations visualisables
- Structuration
 - `<p> ... <p>` → paragraphe
 - `<h1> ... <h1>` → titre de niveau 1
 - ...
 - `<h6> ... <h6>` → titre de niveau 6
 - `<hr />` → ligne horizontale
 - `
` → saut de ligne

Corps : exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://
  www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Le titre de la page</title>
</head>
<body>
  <h1>Titre de niveau 1</h1>
  <p>Un premier paragraphe</p>
  <hr /> <!-- Une ligne horizontale -->
  <blockquote><p>Un autre paragraphe</p>
<p>Un autre paragraphe</p></blockquote>
  <h1>Titre de niveau 1</h1>
  <h2>Titre de niveau 2</h2>
  <p>Un autre paragraphe</p>
</body>
</html>
```





Mise en évidence

- Emphase
 - ` ... ` (emphasis)
- Emphase forte
 - ` ... `
- Exposants et indices
 - `^{...}` (superscript)
 - `_{...}` (subscripted)

Extraits, citations et références

- Citer quelque chose : `cite`
 - ... `<cite>Discours de la méthode</cite>` de `<cite>Descartes</cite>` ...
- Citation courte : `q`
 - Comme le disait `<cite>Ronsard</cite>` :
`<q>Mignonne allons voir si la rose...</q>`
- Citation longue : `blockquote`
 - `<h5><cite>Barbara</cite> de <cite>Jacques Prévert</cite></h5>`
`<blockquote cite="http://wwwuser.gwdg.de/">`
`<p>Rappelle-toi Barbara</p>`
...
`<p>Dont il ne reste rien.</p>`
`</blockquote>`



Autres balises de structuration

- Texte spécial
 - `<pre> ... </pre>` (preformatted text)
 - `<code> ... </code>`
 - pour mettre du texte représente du code informatique (par exemple du XML !)
- Indications sur le contenu textuel
 - Utilisation dans une feuille de style pour mettre en forme
 - `defn` (définition)
 - `acronym` (acronyme)
 - `abbr` (abréviation)
 - Utilisation par des logiciels pour connaître la sémantique du contenu textuel (utilisé ?)
 - Principe du web sémantique : généralisation, extériorisation des vocabulaires, *etc.*



Images

- Principe
 - Un élément image (img) est lié à un fichier image (attribut src)
 - Une image est une ressource externe, désignée par une URL, qui doit être chargée par le navigateur
- Exemple
 - ```

```
- Images cliquables
  - ```

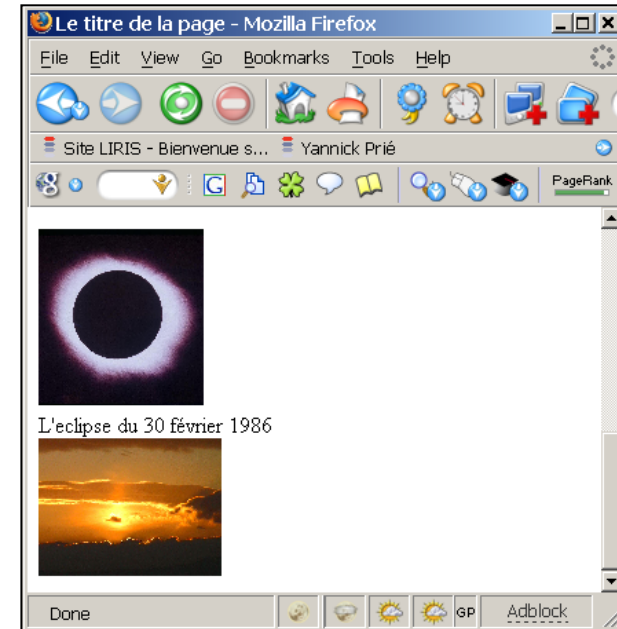
<map id="mymap">
  <area      href="section1.html" alt="Route 20"
    shape="rect" coords="0,0,49,49" />
  <area      href="section2.html" alt="Route 35"
    shape="rect" coords="0,49,49,99" />
</map>
```

Images : exemple

```
...  
  
<br />  
 <!-- erreur ! -->  
<br />  
  
...
```

○ Remarque importante

- il faut trois ressources pour afficher cette page
 - Fichier source HTML
 - Fichier `eclipse.jpg` (chemin local : dans le même dossier que le fichier source HTML)
 - Fichier `soleil.JPG` (disponible sur la machine `web.ccr.jussieu.fr`, en suivant le chemin `/cim2/SAXaussois/soleil.JPG`)
- trois requêtes HTTP en tout pour l'affichage



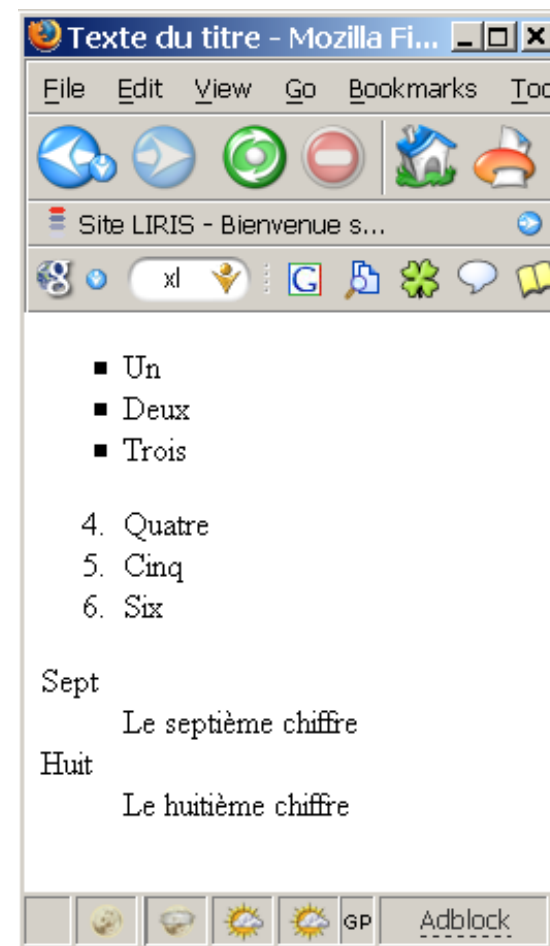


Listes

- Principe général
 - Un élément liste contient des élément items
- Listes classiques
 - Liste à puces
 - ` ... ` (unordered list)
 - Liste ordonnée
 - `<o1> ... </o1>` (ordered list)
 - Item de list
 - ` ... `
- Liste de définitions
 - Conteneur
 - `<d1> ... </d1>`
 - Terme de définition
 - `<dt> ... </dt>`
 - Description de définition (= définition elle-même)
 - `<dd> ... </dd>`

Listes : exemple

```
...  
<ul type="square">  
  <li>Un</li>  
  <li>Deux</li>  
  <li>Trois</li>  
</ul>  
<ol start="4">  
  <li>Quatre</li>  
  <li>Cinq</li>  
  <li>Six</li>  
</ol>  
<dl>  
  <dt>Sept</dt>  
  <dd>Le septième chiffre</dd>  
  <dt>Huit</dt>  
  <dd>Le huitième chiffre</dd>  
</dl>  
...
```





Liens hypertextes

○ Principe

- une ancre contenant le texte ou les éléments sur lesquels on peut cliquer
- le click redirige vers une URL
 - entraîne le chargement de la ressource désignée par l'URL

○ Syntaxe

- ``
ancre
``
- `href` : URL de destination du lien
- `title` : texte qui apparaîtra dans une info-bulle si on survole le lien

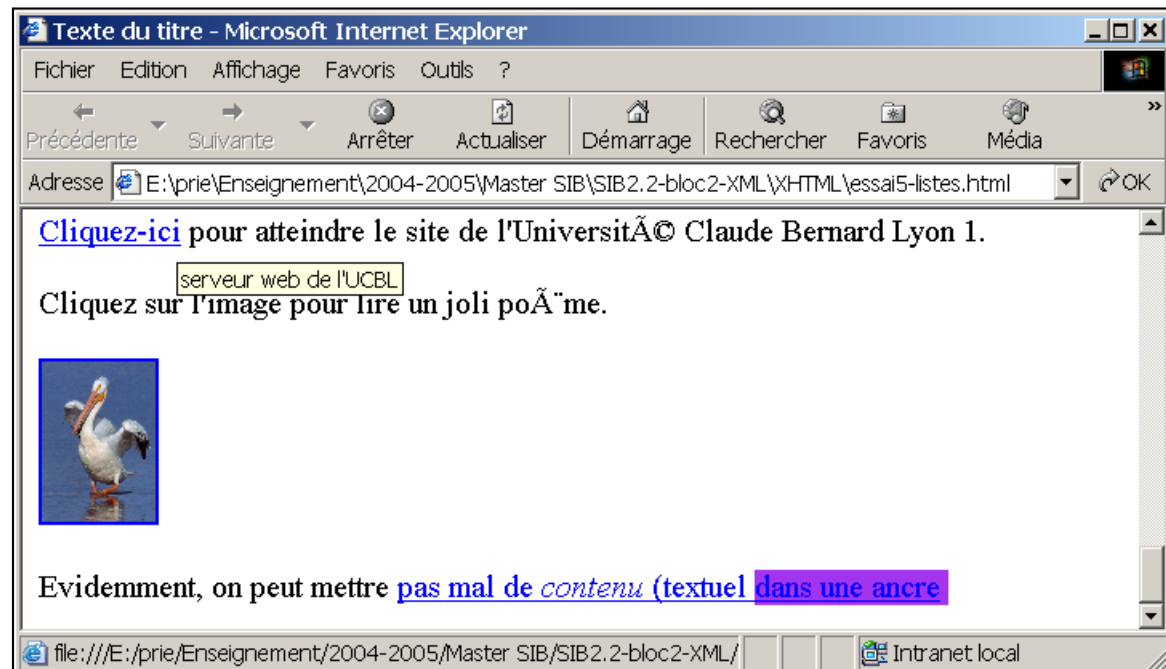
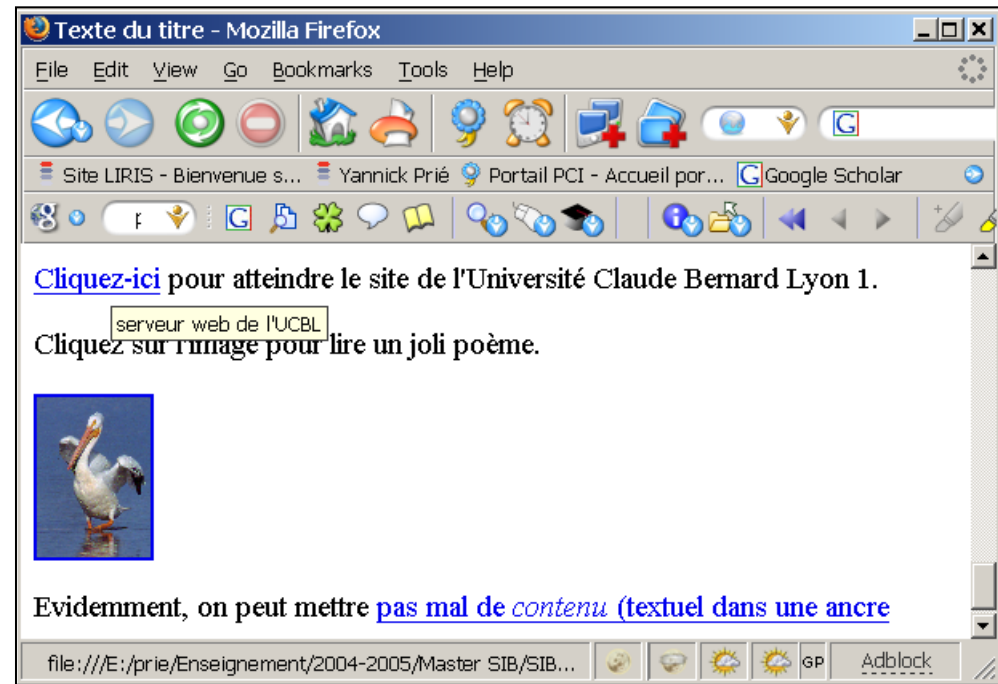


HTML : notion d'URL

- Uniform Ressource Locator
 - permet d'identifier une ressource sur le réseau
- Une ressource peut être
 - une page Web
 - une image (seule ou utilisée dans une page Web)
 - un programme
 - un fichier à télécharger...
- Une URL indique
 - un protocole (langage de communication entre deux programmes sur deux machines)
 - FTP (File Transfert Protocol),
 - HTTP (HyperText Transfert Protocol)...
 - l'adresse d'un serveur
 - un chemin dans l'arborescence des fichiers
- Forme générale : **protocole**://**adresse**/**chemin**
 - Exemples
 - `http://www.univ-lyon1.fr/`
 - `http://www710.univ-lyon1.fr/~yprie/Enseignement/SIB/SIB-UE3-bloc4/CM4.6-7.pdf`

Liens hypertextes : exemples

```
<p>Quelques exemples de liens</p>
<p><a href="URL" title="serveur
web de l'UCBL">Cliquez-ici</a>
pour atteindre le site de
l'Université Claude Bernard
Lyon 1.</p>
<!-- Remarque : on a ici un lien
absolu (URL complète) -->
<p>Cliquez sur l'image pour lire
un joli poème.</p>
<a href="poeme.html"></a>
<!-- Remarque : on a ici un lien
relatif -->
<p>Evidemment, on peut mettre <a
href="http://www.w3.org/TR/
xhtml1/">pas mal de
<em>contenu</em> (textuel
<span style="background-color:
#A234EE" éléments)</span> dans
une ancre</a> </p>
```





Tableaux

- Principe

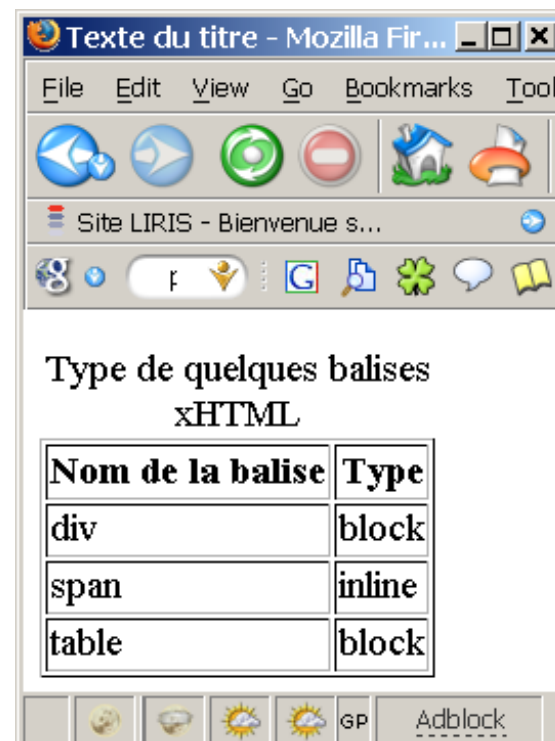
- Un tableau contient des lignes, lesquelles contiennent des cellules

- Éléments

- `<table> ... </table>` (élément général)
- `<tr> ... </tr>` (table row)
- `<td> ... </td>` (table cell)
- `<th> ... </th>` (table header)
- `<caption> ... </caption>` (table caption)
- ...

Tableaux : exemple

```
<table summary="Ce tableau présente des balises xHTML et leur type."
  border="1">
<caption>Type de quelques balises xHTML</caption>
<thead>
<tr>
  <th scope="col">Nom de la balise</th>
  <th scope="col">Type</th>
</tr>
</thead>
<tbody>
<tr>
  <td>div</td>
  <td>block</td>
</tr>
<tr>
  <td>span</td>
  <td>inline</td>
</tr>
<tr>
  <td>table</td>
  <td>block</td>
</tr>
</tbody>
</table>
```





Deux types d'éléments en HTML

- Éléments *block*
 - Définissent des blocs physiques dans le document
 - Par défaut, induisent un saut de ligne à la présentation
 - Exemple
 - `body`, `p`, `h1-h6`, `table`, `ul`, `li`, etc.
- Éléments *inline*
 - Dans le flux des caractères
 - « Semblables » à des caractères
 - Exemple
 - `em`, `img`, `strong`, `a`, etc.



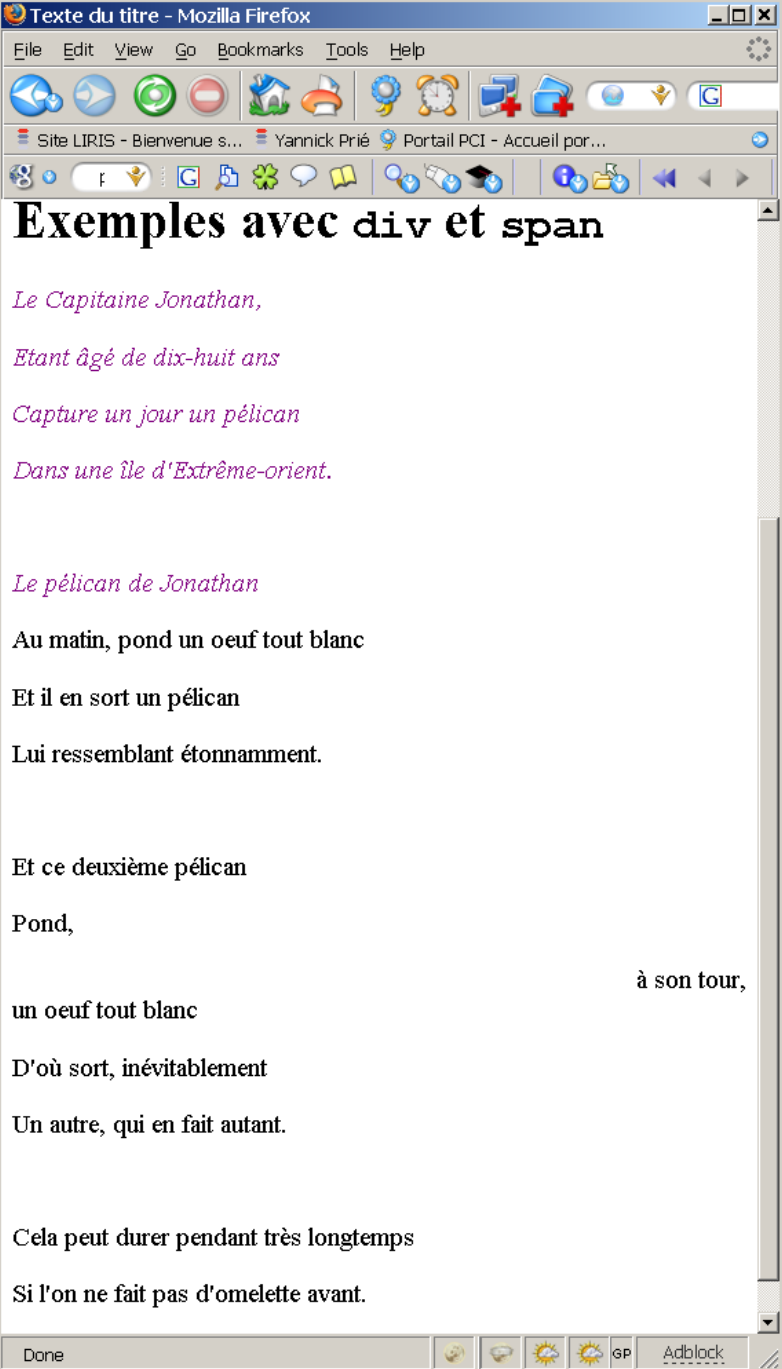
div/span pour spécifier des sous-parties de document sans leur donner un nom

- Deux éléments
 - destinés à enclore (contenir) d'autres éléments XHTML
 - pour les traiter globalement en leur affectant des styles
- **<div> ... </div>**
 - Contient d'autres éléments, élément block
 - Retour chariot à la fin
- ** ... **
 - Contient d'autres éléments, est dans le flux des éléments (inline)

div / span :

exemple (Desnos)

```
<h1>Exemples avec <code>div</code> et  
<code>span</code></h1>  
<span style="color: purple ; font-  
style: italic"><p>Le Capitaine  
Jonathan,</p>  
<p>Etant âgé de dix-huit ans</p>  
<p>Capture un jour un pélican</p>  
<p>Dans une île d'Extrême-orient.</p>  
<br />  
<p>Le pélican de Jonathan</p></span>  
<p>Au matin, pond un oeuf tout blanc</  
p>  
<p>Et il en sort un pélican</p>  
<p>Lui ressemblant étonnamment.</p>  
<br />  
<p>Et ce deuxième pélican</p>  
<p>Pond, <div style="text-align:  
right">à son tour, </div>un oeuf  
tout blanc</p>  
<p>D'où sort, inévitablement</p>  
<p>Un autre, qui en fait autant.</p>  
<br />  
<p>Cela peut durer <span style="text-  
align: right">pendant</span> très  
longtemps</p>  
<p>Si l'on ne fait pas d'omelette  
avant.</p>  
<br />
```



Texte du titre - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

Site LIRIS - Bienvenue s... Yannick Prié Portail PCI - Accueil por...

Exemples avec div et span

Le Capitaine Jonathan,

Etant âgé de dix-huit ans

Capture un jour un pélican

Dans une île d'Extrême-orient.

Le pélican de Jonathan

Au matin, pond un oeuf tout blanc

Et il en sort un pélican

Lui ressemblant étonnamment.

Et ce deuxième pélican

Pond,

un oeuf tout blanc

D'où sort, inévitablement

Un autre, qui en fait autant.

Cela peut durer pendant très longtemps

Si l'on ne fait pas d'omelette avant.

Done Adblock



Formulaires

- Objectif
 - Permettre à l'utilisateur d'envoyer des informations à un serveur
 - Saisir les informations
 - listes déroulantes, cases à cocher, zones de texte, etc.
 - bouton pour remettre à zéro
 - Envoyer les informations
 - méthode GET
 - les informations passent par l'URL
 - Ex. : `http://www.google.com/search?sourceid=mozclient&ie=utf-8&oe=utf-8&q=html+4.01`
 - Méthode POST
 - les informations sont échangées par le protocole HTTP
- Eléments
 - `<form> ... </form>` → définit un formulaire
 - `<input> ... </input>` → entrée de formulaire
 - `<textarea> ...</textarea>` → zone de texte
 - `<select> ... </select>` → liste déroulante
 - `<option> ... </option>` → option du select

Formulaires : exemple

```
<form method="post" action="http://serveur.com/
script.php">

<p><input type="text" name="Champ_saisie"
value="Texte" /></p>

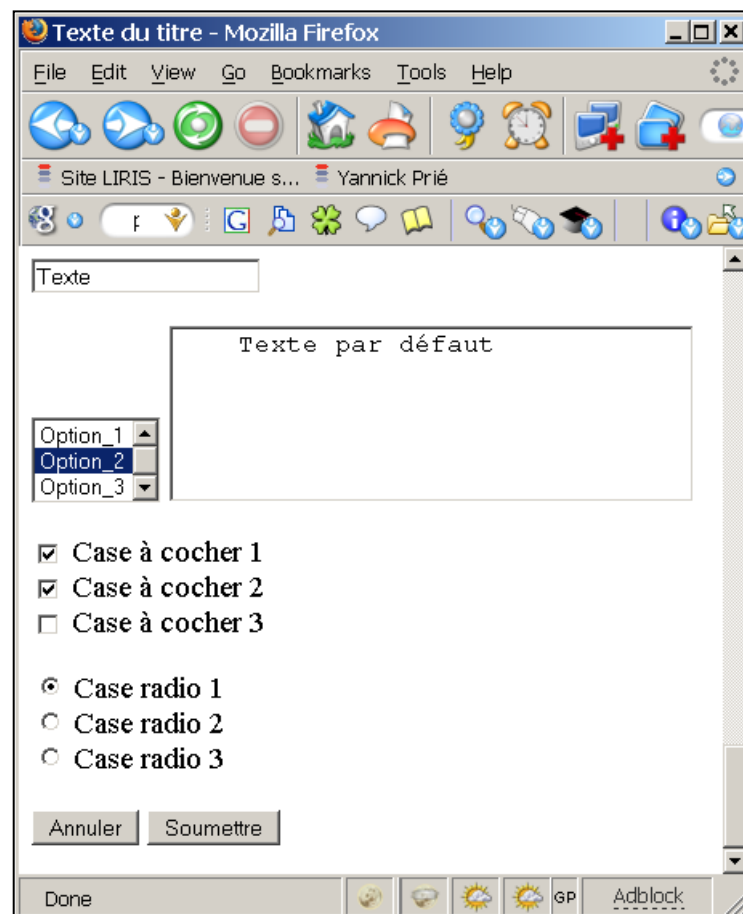
<p><select name="Liste_Choix" size="3">
<option value="Option_1">Option_1</option>
<option value="Option_2">Option_2</option>
<option value="Option_3">Option_3</option>
</select>

<textarea name="Zone_Texte" cols="30" rows="5">
Texte par défaut </textarea></p>

<p><input type="checkbox" name="case1" value="Case_1">
Case à cocher 1<br />
<input type="checkbox" name="case2" value="Case_2">
Case à cocher 2<br />
<input type="checkbox" name="case3" value="Case_3">
Case à cocher 3<br /></p>

<p><input type="radio" name="Case_Radio"
value="Case radio 1">Case radio 1<br />
<input type="radio" name="Case_Radio"
value="Case radio 2">Case radio 2<br />
<input type="radio" name="Case_Radio"
value="Case radio 3">Case radio 3<br /></p>

<input type="reset" name="Annulation" value="Annuler">
<input type="submit" name="Soumission"
value="Soumettre">
</form>
```





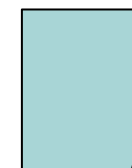
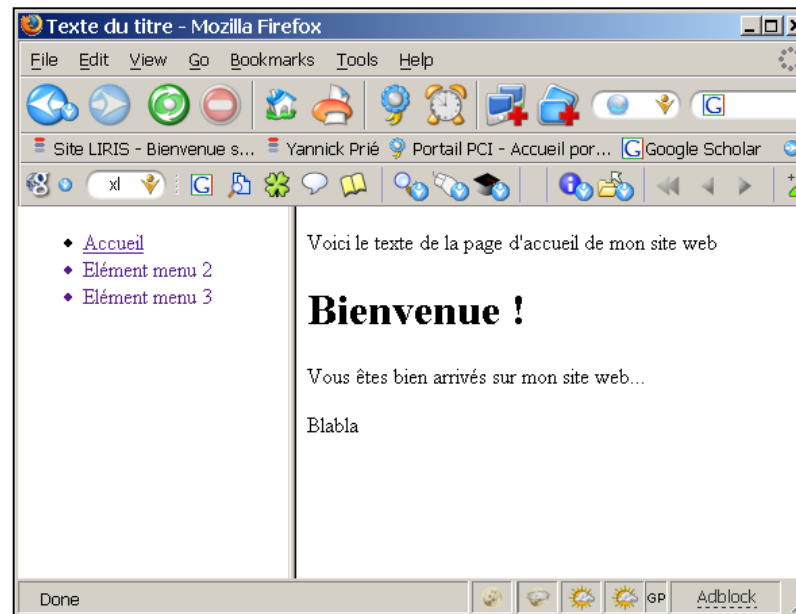
Frames

- Principe
 - diviser la fenêtre d'affichage en frames
 - afficher dans chaque frame une page HTML
 - Nombre de fichiers concernés
 - 1 pour la déclaration du frameset + 1 pour chaque frame
 - Ex : 3 frames → minimum 4 fichiers
- En XHTML
 - Le fichier principal obéit à la DTD frameset
 - Les autres sont en XHTML
- Remarque
 - Ne pas utiliser car pose beaucoup de problèmes
 - gestion
 - indexation

Frames : exemple

Le fichier principal fait appel à menu.html et accueil.html

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE HTML PUBLIC
  "-//W3C//DTD HTML 4.01 Frameset//EN"
  "http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>Texte du titre</title>
</head>
<frameset cols="200,*">
  <frame src="menu.html" name="cadre1" scrolling="no">
  <frame src="accueil.html" name="cadre2" scrolling="yes">
</noframes>
  <p>Ce projet utilise des cadres. Chez vous les cadres ne
  sont pas affichés.</p>
</noframes>
</frameset>
</html>
```



menu.html



accueil.html



Conclusion sur XHTML

- Possibilité de valider ses documents
 - <http://validator.w3.org/>
- Ce cours n'est pas une description complète des éléments et attributs de XHTML
 - Aller voir la référence
- Non abordés ici
 - Les scripts
 - Javascript
 - `<script> ... </script>`
 - Les objets pris en charge par des plugins
 - Applets JAVA, animations Flash, audio, vidéo, etc.
 - `<object> ... </object>`
 - Les sites web et leur gestion (conception, installation sur serveurs web...)

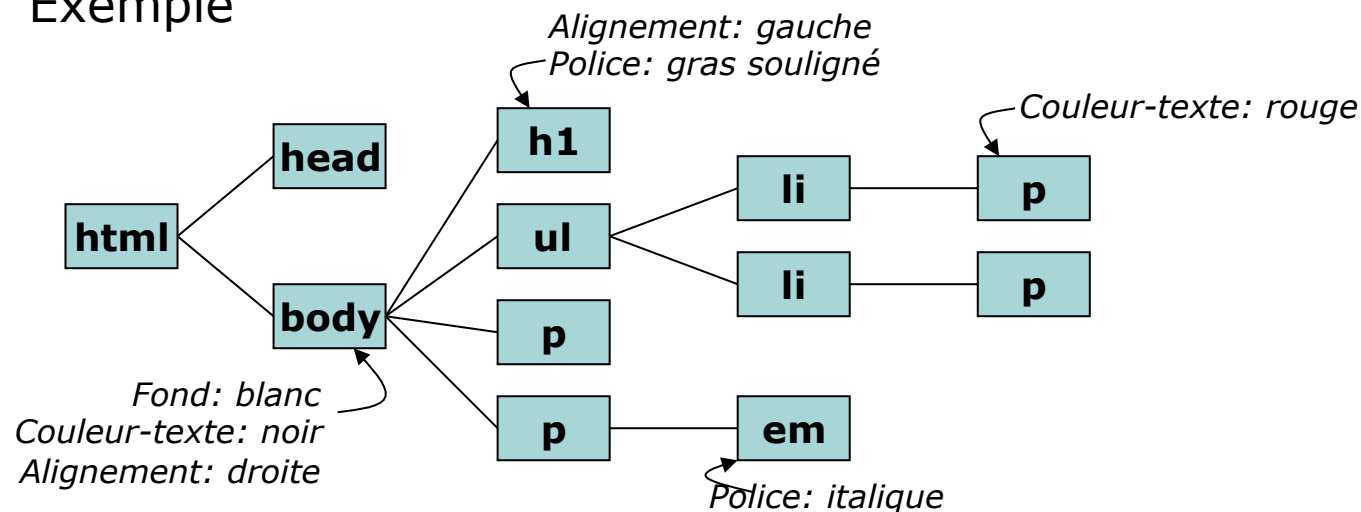


Plan

- XHTML
 - Structure générale d'un document XHTML
 - XHTML Strict et Transitionnal
 - Éléments les plus utiles
- CSS
 - Généralités
 - Règles et sélecteurs
 - Propriétés utiles
 - Placement des styles
 - Héritage et cascade

CSS : généralités

- Objectif
 - Décrire comment un document XHTML doit être affiché
 - Remplace les éléments d'affichage des anciennes versions de HTML (séparation réelle du contenu et de sa présentation)
 - L'affichage est pris en charge par le navigateur (normalement)
- Principe
 - Décoration de l'arbre des éléments XHTML
 - Associer un certain nombre d'attributs de style à un élément
- Exemple

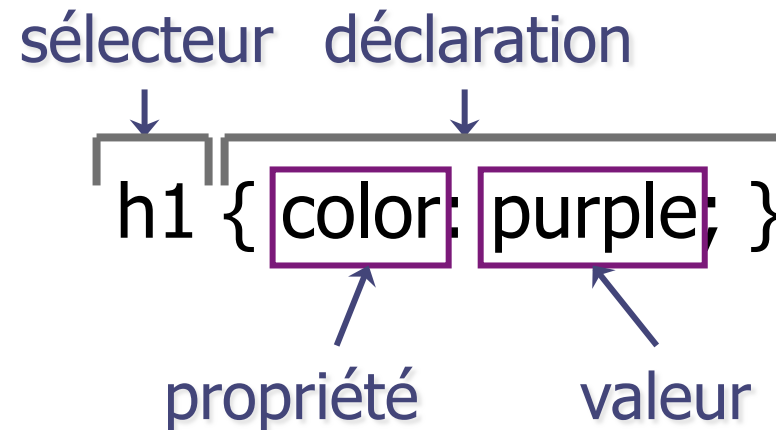




Feuille de style CSS

- Une feuille de style est composée d'un certain nombre de règles (*rules*)
- Une règle se compose
 - d'un sélecteur
 - d'une déclaration
- Une déclaration se compose d'un ensemble de propriétés/valeurs
- Remarque
 - L'ordre des règles est indifférent
 - Tous les styles ne peuvent pas s'appliquer à tous les éléments
 - Ex. : une image n'a pas de style de police

CSS : structure des règles



```
h1 { font-family: Arial, sans-serif; font-style: italic }
```

séparateur de propriétés/valeurs



Exemple de règle CSS

```
body {  
    background: #FFFFFF;  
    color: black; /* commentaire */  
    margin-left: 5%;  
    margin-right: 5%;  
    font-family: Tahoma, Optima,  
    Arial, sans-serif;  
}
```



Types de sélecteurs

- Simples et groupes
- Classes
- Pseudo-classes
- Pseudo-éléments
- Contextuels



Sélecteurs CSS : simple et groupe

- Simple
 - Lié à un type d'élément HTML
 - Utilisation de son nom
 - Exemple
 - `h1 { text-align: center; }`
- Groupe
 - Regroupement de règles qui s'appliquent à plusieurs éléments
 - Exemple
 - `h2, p { font-family: Optima, Arial, sans-serif; }`
- Impossibilité de considérer différemment des éléments de même type



Sélecteurs CSS : classes

- On peut assigner une classe à un élément HTML

```
<h1 class="header" >
```

- Celle-ci spécifie un sélecteur particulier dans le feuille de style

```
h1.header { text-align: center; }
```

- Une classe peut s'appliquer à de multiples éléments

```
.header { text-align: center; }
```

s'appliquera aussi à `<h2 class="header">`, *etc.*



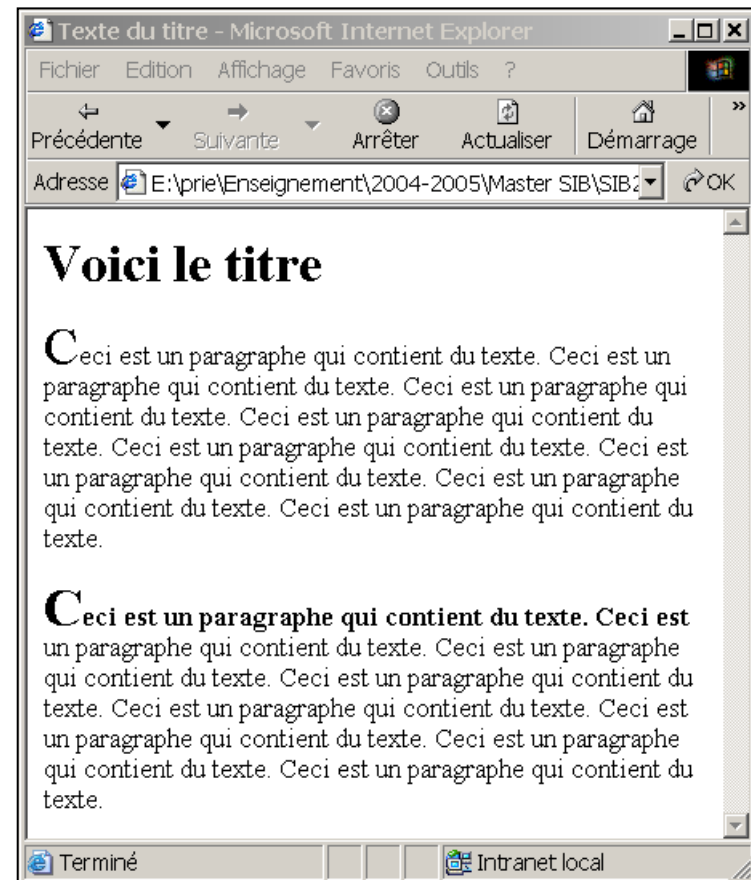
Sélecteurs CSS : pseudo-classes

- Sélecteurs qui sélectionnent des éléments en fonction de leur état à un moment donné
- Exemple et intérêt principal
 - `a:link` — lien non visité et inactif
 - `a:hover` — lien sur lequel passe le pointeur de la souris
 - `a:active` — lien sur lequel on clique
 - `a:visited` — lien déjà visité
- Exemple

```
a:link {color: blue;}  
a:visited {color: magenta;}  
a:hover {color: red; text-  
decoration:none; font-weight: bold;}  
a:active {color: red;}
```

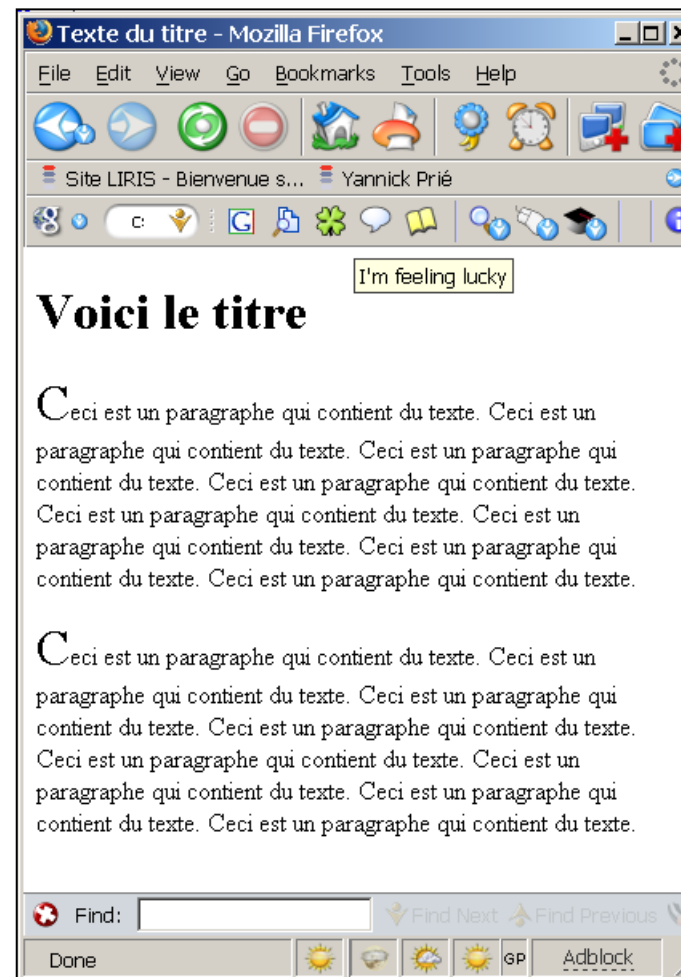
Sélecteurs CSS : pseudo-éléments

- **:first-letter**
 - première lettre dans un élément bloc (ex. p, h1, ...)
- **:first-line**
 - première ligne dans un élément bloc (ex. p, h1, ...)



Pseudo-éléments : exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD
  XHTML 1.0 Strict//EN" "http://
  www.w3.org/TR/xhtml1/DTD/xhtml1-
  strict.dtd">
<html>
<head>
<title>Texte du titre</title>
<style type="text/css">
  p:first-letter {font-size: 200%;}
  p.lignel:first-line {font: bold;}
</style>
</head>
<body>
<h1>Voici le titre</h1>
<p>Ceci est <!-- ... --> du texte. </p>
<p class="lignel">Ceci <!-- ... --> qui
  contient du texte. Ceci est un
  paragraphe qui contient du texte.
</p>
</body>
</html>
```



Sélecteurs CSS : contextuels

- Sélecteurs qui ne sélectionnent que des éléments dans un certain contexte

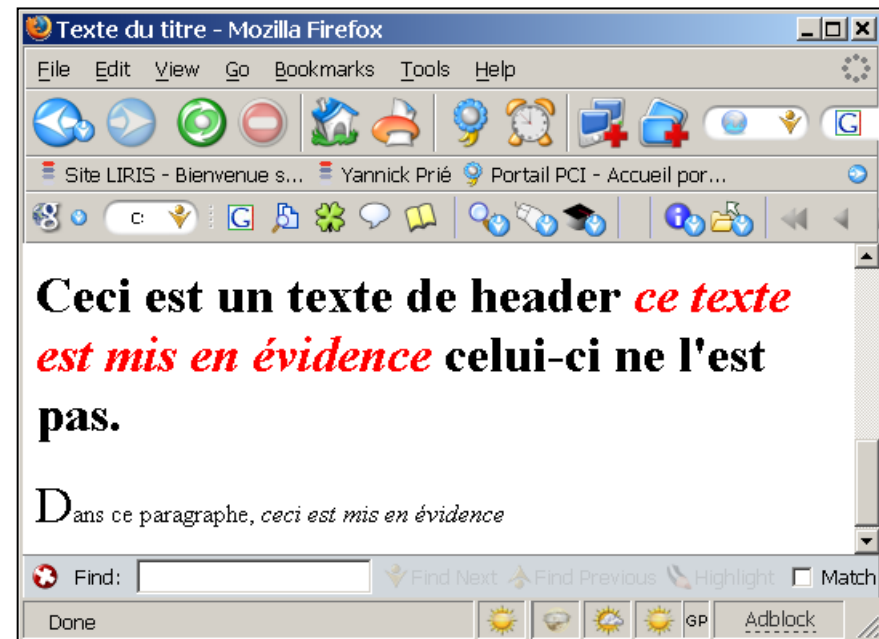
Style :

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

XHTML :

```
<h1>Ceci est un texte  
de header <em>ce texte  
est mis en évidence</  
em> celui-ci ne l'est  
pas.</h1>
```

```
<p>Dans ce paragraphe,  
<em>ceci est mis en  
évidence</em></p>
```





Quelques propriétés de textes

- font-size:
 - small | medium... | % | x pt
- font-family:
 - fontname1, fontname2 (*si la première n'est pas disponible*), familyname (*serif, sans-serif, etc.*)
- font-weight:
 - bold | lighter
- font-style:
 - italic, oblique
- text-align:
 - left | center | right | justify
- text-indent: (*retrait de première ligne*)
 - % | x cm



Les couleurs

- color:
 - red | blue...| hexcode
- background-color:
 - red | blue...| hexcode

■ Black = "#000000"

■ Silver = "#C0C0C0"

■ Gray = "#808080"

□ White = "#FFFFFF"

■ Maroon = "#800000"

■ Red = "#FF0000"

■ Purple = "#800080"

■ Fuchsia = "#FF00FF"

■ Green = "#008000"

■ Lime = "#00FF00"

■ Olive = "#808000"

■ Yellow = "#FFFF00"

■ Navy = "#000080"

■ Blue = "#0000FF"

■ Teal = "#008080"

■ Aqua = "#00FFFF"



Types d'éléments

- Éléments blocs
 - Prennent la forme d'un bloc dans la page
 - ensemble de lignes
 - ne peuvent être contenus que dans d'autres éléments blocs
 - Exemple
 - p, img, ul, table, h1, **div**, ...
 - Propriétés de bloc
- Éléments inline
 - S'inscrivent dans la continuité des éléments
 - ne forcent pas un changement de ligne
 - peuvent être inclus dans n'importe quel élément
 - Exemple
 - a, em, **span**, ...
- Éléments de listes
 - éléments HTML qui ont un marqueur (bullet, number) et un ordre



Rappels sur div et span

- **div**
 - élément contenant d'autres éléments, servant à définir un bloc
 - servira essentiellement à faire du positionnement de bloc
- **span**
 - élément contenant d'autres éléments, inline
 - servira essentiellement à regrouper des caractéristiques globales pour des éléments textuels



Propriétés de styles graphiques

- S'appliquent aux éléments blocs
- Attribut **position**
 - **static**
 - bloc placé en fonction de sa position dans l'ordre des balises, ne peut pas être modifiée dynamiquement
 - **absolute**
 - bloc placé précisément par rapport aux bords de la fenêtre d'affichage
 - la position peut être modifiée dynamiquement (déplacement)
 - **relative**
 - bloc positionné par rapport à sa position normale, dans le flux (ex. décalage de 20 pts à droite)



Propriétés de styles graphiques (2)

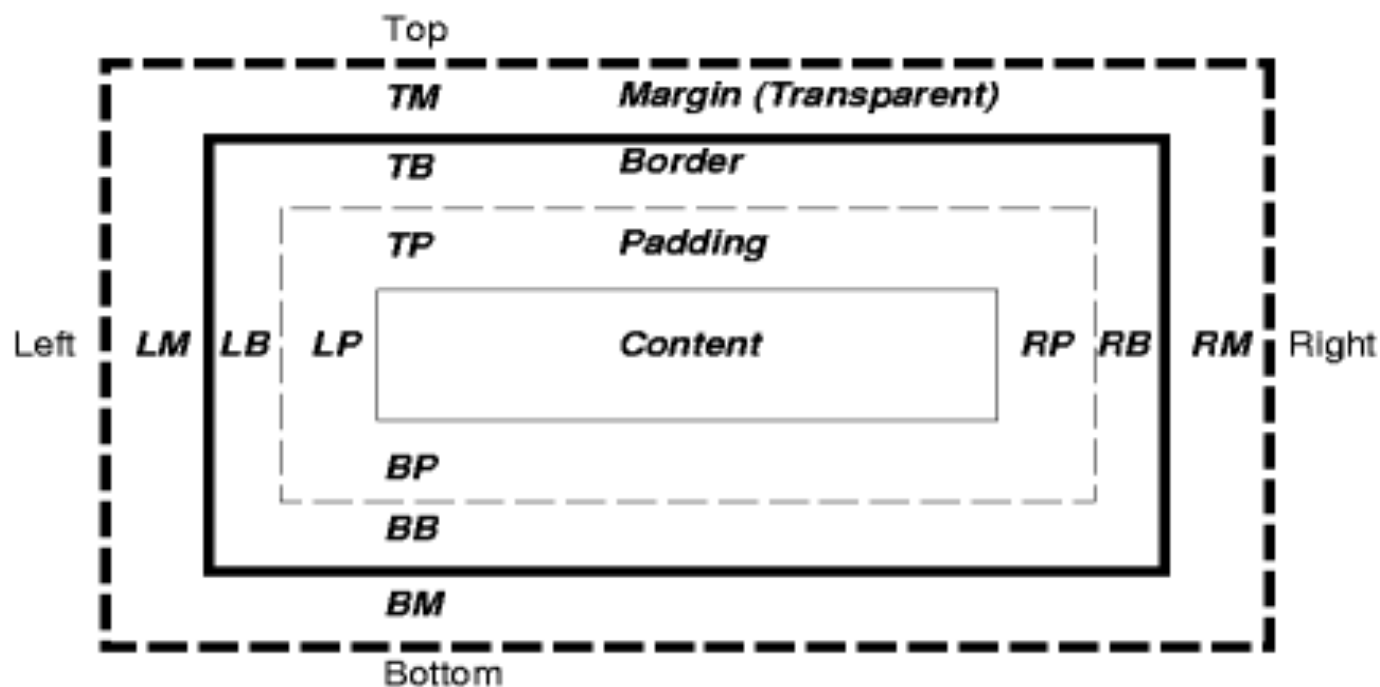
- Attribut **clip**
 - définit une zone de rognage du bloc
 - par exemple pour rogner une image
- Attribut **visibility** et **display**
 - indiquent si (visible|hidden) et comment un élément est affiché
- Attribut **z-index**
 - en cas de superposition de blocs d'affichage, indique l'ordre dans lesquels le navigateur doit les afficher (cf. logiciel de dessin)
- Remarques
 - les distances s'expriment en pixels (**px**), points (**pt**), unités métriques (**cm**, **mm**), ou pourcentages de la taille de la fenêtre (%)



Propriétés de styles graphiques (3)

- Arrière-plan
 - background-color
 - couleur de fond
 - background-image
 - image de fond
 - background-repeat
 - types de répétition de l'image
 - background-attachement
 - arrière-plan fixe quand on utilise l'ascenseur
 - background-position
 - position par rapport au coin supérieur gauche
 - background
 - attribut de résumé
- Exemple
 - `p { background: image.gif fixed repeat-y }`

Marge, bordures, padding



- Margin edge
- Border edge
- - - - - Padding edge
- Content edge

CSS définit des attributs pour définir précisément ces zones



Styles intégrés

- Déclarés comme attribut `style` d'un élément
- Exemple
 - `<h1 style="font-family: Arial; font-style: italic;">Un texte qui se retrouvera en arial italique</h1>`



Feuilles de styles incorporées

- La feuille de style est déclarée dans l'en-tête (**head**) du document XHTML
- Elle s'appliquera aux éléments dans **body**
- Exemple

```
<html>
  <head>
    <title>Texte du titre</title>
    <style type="text/css">
      .important {color: red;}
      h1 {font-family: Arial; font-style: italic; }
      em { color: green }
    </style>
  </head>
  <body>
    <!-- éléments sur lesquels s'appliqueront les styles -->
  </body>
</html>
```




Feuilles de styles liées

- Les règles se trouvent dans un fichier extérieur
- On indique au navigateur où se trouve cette ressource de style, en utilisant un élément `link` dans l'en-tête
 - `<link rel="stylesheet" type="text/css" href="fichier-de-style.css" >`
- Intérêt
 - Plusieurs documents XHTML peuvent faire appel à la même feuille de style



Héritage de styles

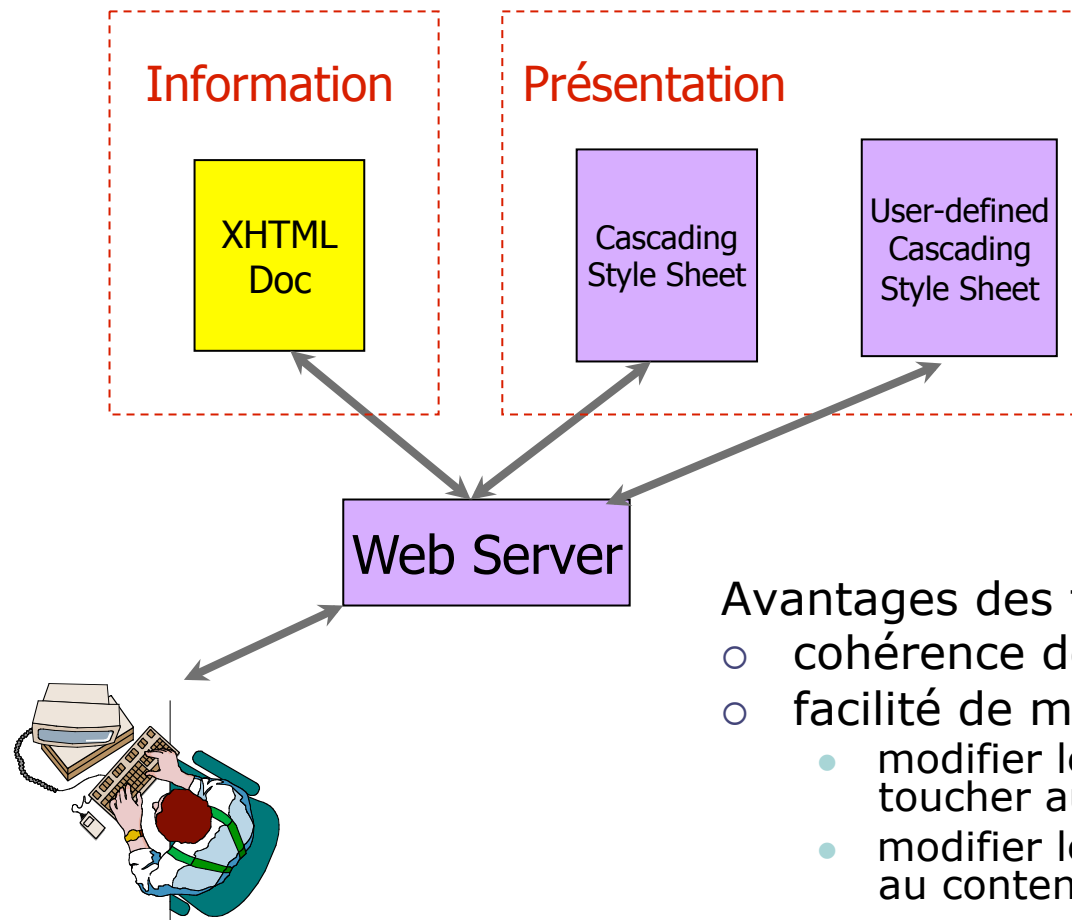
- Par défaut
 - Les styles d'un élément sont hérités par ses éléments descendants
 - Exemple :
 - `<p style="color: red ;">Du texte mis en évidence, pas mis en évidence.</p>`
 - Du texte *mis en évidence*, pas mis en évidence.
- A condition que l'héritage ait un sens
 - *i.e.* que les caractéristiques soient applicables à l'élément enfant
 - un positionnement de bloc n'a pas d'intérêt pour un élément em qui y est contenu
 - si un style est défini spécialement pour un élément (ex. `em { color: blue; }`), l'héritage ne se fait pas
 - Du texte *mis en évidence*, pas mis en évidence.



Cascading style sheets : cascade

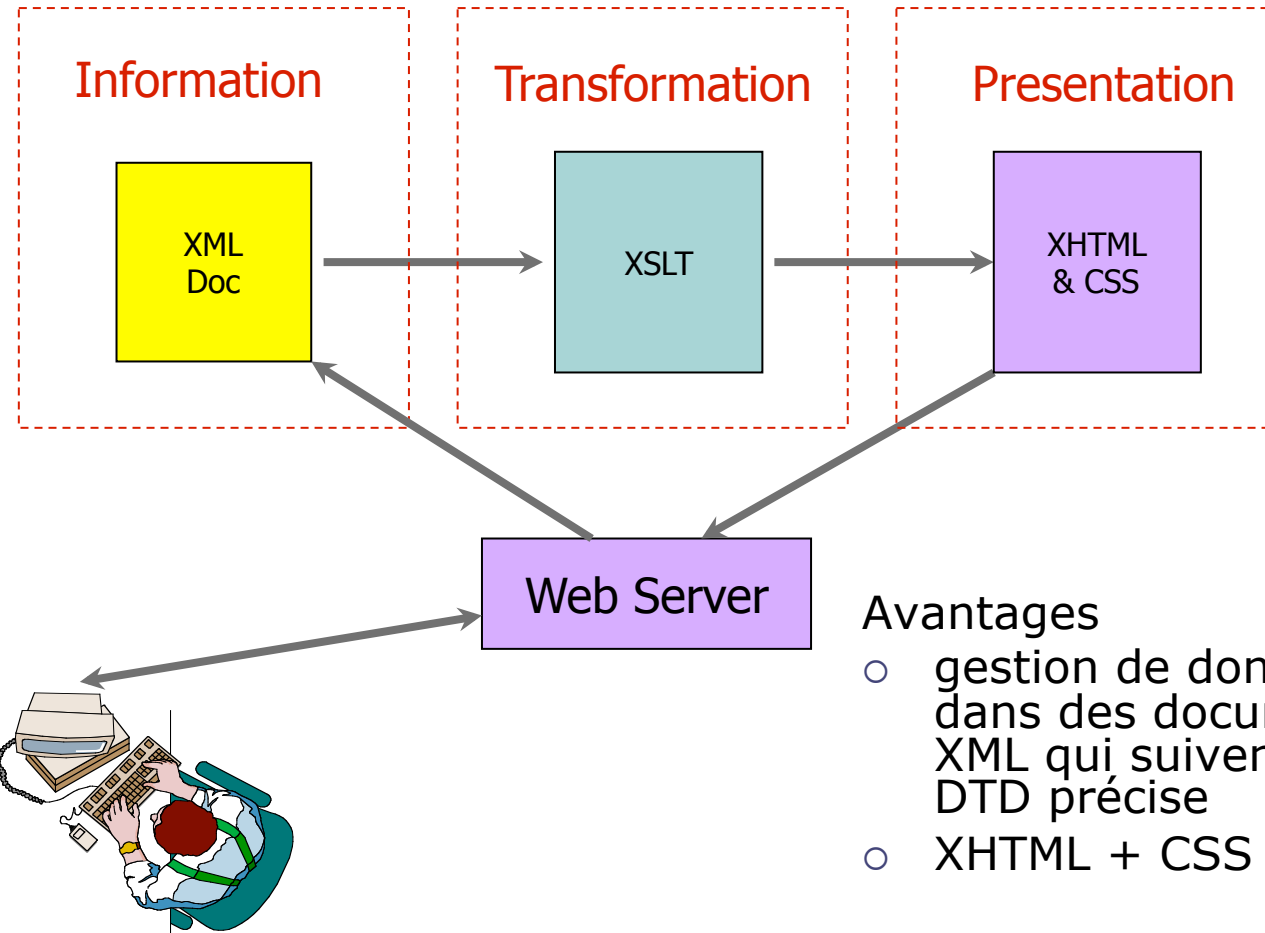
- On peut avoir concurrence entre plusieurs styles définis dans de multiples endroits
 - styles par défaut (1- navigateur)
 - fichiers CSS externes (2- spécifications globales au site)
 - élément `head` du document XHTML (3- spécification globales au doc.)
 - attributs `style` des éléments (4- spécification locales)
 - style utilisateur (5- spécification de l'utilisateur)
- Notion de cascade ou ordre de priorité des styles
 - trouver toutes les déclarations qui s'appliquent à un élément
 - les classer par spécificité
 - système de poids qui s'ajoutent
 - les classer par ordre d'apparence
 - plus un déclaration apparaît tard, plus elle a de poids
- Exemple
 - style (4) > style (4 hérité) > style (3) > style (2) > style (1)

Présentation avec des CSS



- Avantages des feuilles de style
- cohérence des présentations
 - facilité de maintenance
 - modifier le contenu sans toucher au style
 - modifier le style sans toucher au contenu

Présentation avec XSLT



Avantages

- gestion de données dans des documents XML qui suivent une DTD précise
- XHTML + CSS



Conclusion sur CSS

- Actuellement
 - CSS2.1 (working draft)
 - CSS3 en cours de développement
- Les navigateurs gèrent diversement les feuilles de style !
 - rester simple
 - tester avec plusieurs navigateurs
- Aller voir les références



Conclusion

- Présentation rapide de XHTML/CSS
- Après, il faut pratiquer
 - XHTML strict + CSS
- Très nombreuses ressources sur le web
 - éditeurs, assistants
 - tutoriaux
 - références



Remerciements

- Professional Web Authoring With XHTML and CSS – Roy Tennant
- Cours Lionel Médini