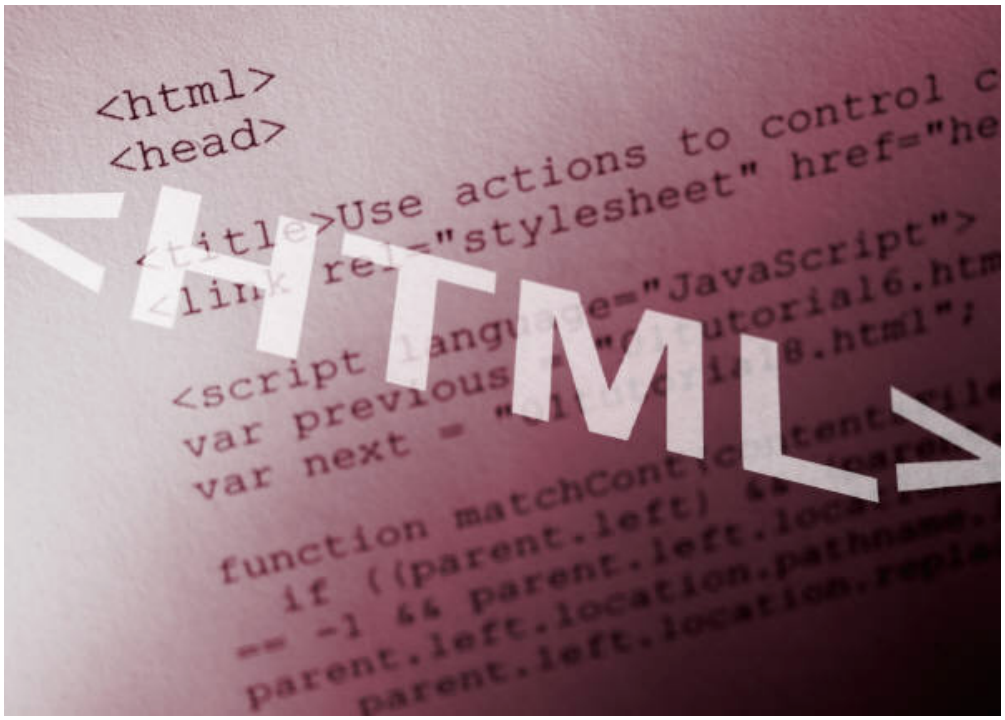
CONCEPTION DE DOCUMENTS WEB



- HTML & XHTML
- CSS
- JAVASCRIPT
- DOM
- AJAX
- JQUERY

PLAN DE L'EXPOSE 1

- HTML & XHTML
  - [Règles d'écriture]
  - Règles de conformités XHTML
  - Typologie des éléments
  - [Présentation de quelques éléments]
  - Structure des documents
  - Bonnes pratiques

HTML : HyperText Markup Language
Structuration de documents dédiés aux sites Web

XHTML : eXstensible HyperText Markup Language
Equivalent à HTML
Mais reformulation de HTML en **XML**
Ecriture plus rigoureuse

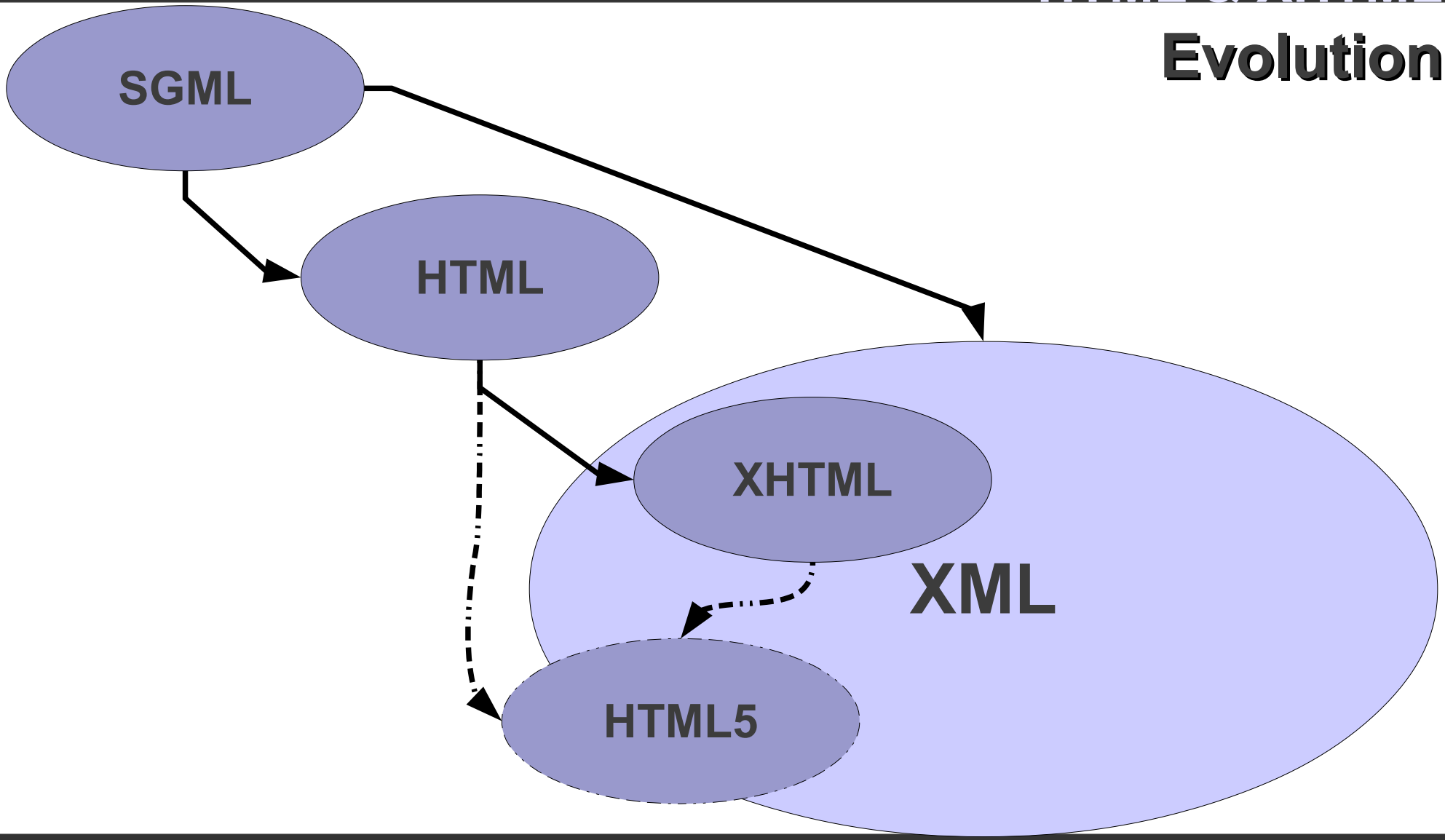
Fichiers au **format texte**

Extension **.html** ou htm

Sous l'autorité du **W3C**
World Wide Web Consortium
http://www.w3c.org

Langages non propriétaires

Evolution

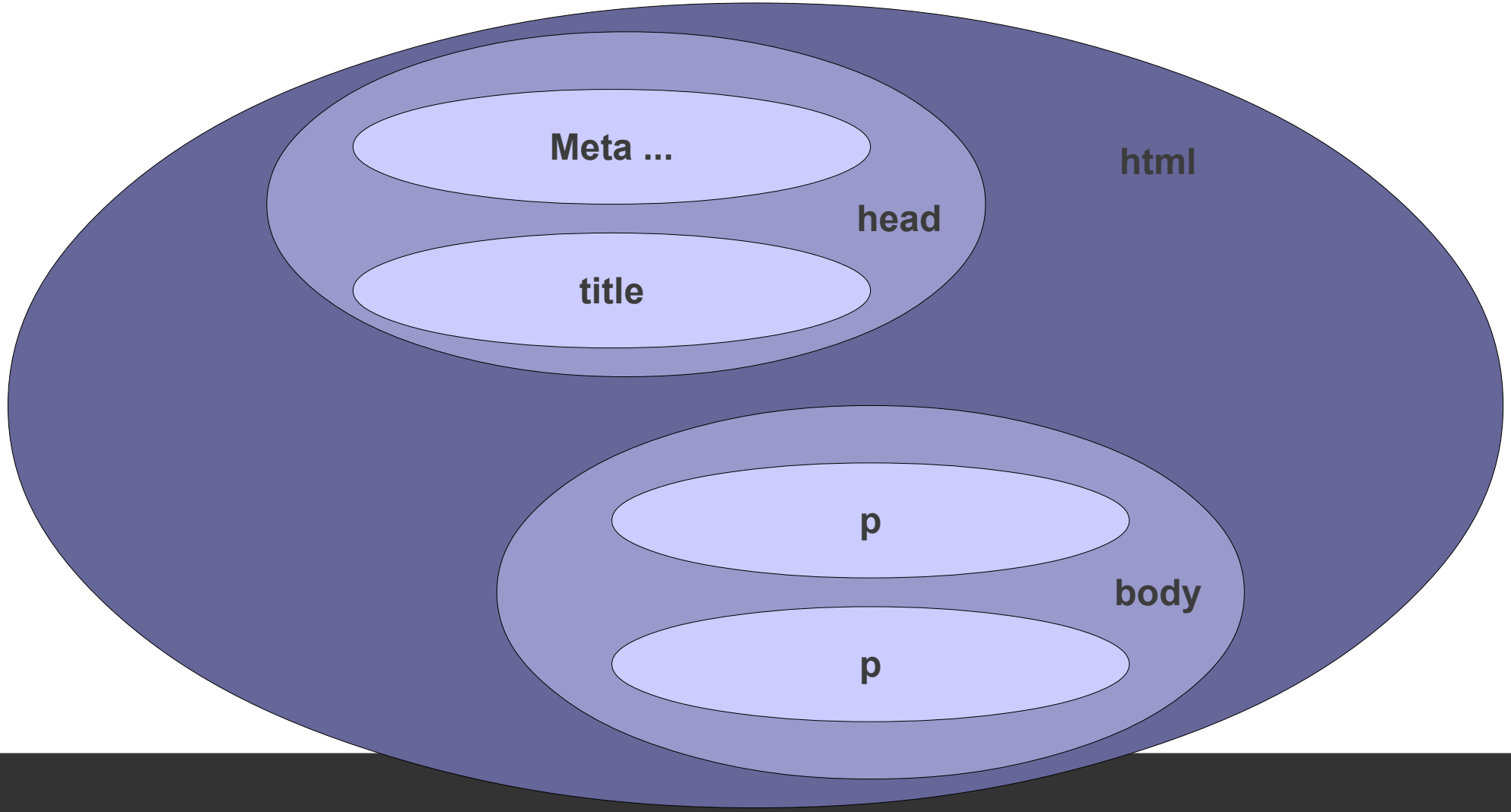


Langage de balisage
balises / marqueurs / tags

Paires balises ouvrantes et fermantes
** Un paragraphe **

Des balises uniques
Fermées en XHTML
**
** (retour à la ligne)
<hr /> (ligne de séparation horizontale)

L'imbrications des éléments



Voir l'exemple correspondant remis en cours – Hello World

Les attributs

paires : nom="valeur"

Dans les balises ouvrantes ou uniques

** Paragraphe unique **
** Paragraphe spécial **

Apostrophes doubles
(le plus courant)
ou simples

Les commentaires

< !-- Un commentaire -->

Non affiché par les navigateurs
Mais comme tout le reste du document
Lisible en affichant la source

Le chevauchement des balises est illégal

<h1>Texte sans chevauchement</h1>
est valide.

~~**<h1>Texte avec chevauchement</h1>**~~
~~**n'est pas valide.**~~

Toutes les balises doivent être fermées même les balises uniques.

** un paragraphe **
est valide.

****
est valide.

XHTML est sensible à la casse
les noms d'éléments et d'attributs
doivent être saisis en minuscules

****
est valide.

~~**<IMG SRC="UneImage.jpg" WIDTH="32" HEIGHT="32" ALT="Mon Image" />**~~
n'est pas valide.

Question : quid des valeurs d'attribut : ici "UneImage.jpg" ou "Mon Image", par exemple ?

Les valeurs d'attributs sont toujours spécifiées entre apostrophes

****
est valide.

~~****~~
n'est pas valide.

Tout attribut nécessite une valeur

<input type="checkbox" name="rouge" id=”elt1” value="ok" checked="checked" />
est valide.

~~**<input type="checkbox" name="rouge" id=”elt1” value="ok" checked />**~~
n'est pas valide.

L'attribut name doit être remplacé par l'attribut id

<form id="ceformulaire">...</form>
est valide.

~~<form name="autreformulaire">...</form>~~
n'est pas valide.

NB : les éléments constitutifs de formulaire, eux, conservent l'attribut name
Savez vous pourquoi ?
Dans la pratique, on fait souvent cohabiter les attributs name et id. Pourquoi ?

Les éléments script et style sont déclarés comme possédant un contenu de données textuelles analysées (PCDATA : Parsed Character DATA)

```
<script type="text/javascript">
<![CDATA[
var val = 100 + 50;
]]>
</script>
```

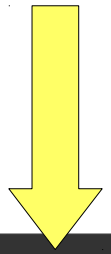
On ne met pas de tiret double à l'intérieur d'un commentaire

<!-- Un commentaire -->
<!--============-->
sont valides.

<!-- Un autre -- commentaire -->
<!---------------->
ne sont pas valides.

XHTML et HTML – Règles de conformités

Certains éléments du HTML sont déconseillés ou correspondent aux versions transitoires d'HTML ou XHTML



Retrouvez la liste des éléments HTML 4.01 sur http://www.la-grange.net/w3c/html4.01/index/elements.html
Et les attributs sur http://www.la-grange.net/w3c/html4.01/index/attributes.html

Interlude - Sondages

- Quelle version d'HTML / XHTML utilisez vous ?
  - **HTML 1.0**
  - **HTML 2.0**
  - **HTML 3.0**
  - **HTML 3.2**
  - **HTML 4.0**
  - **HTML 4.1**
  - **XHTML 1.0**
  - **XHTML 1.1**
  - **HTML 5**
  - **HTML5**
  - **Je ne sais pas**
  - **Je ne code pas**

Plutôt strict ?
Plutôt transitionnel ?
Je n'y prête pas attention.

Attention, il y a plusieurs pièges !

2 grands mode de rendu des éléments

Les rendus de type **block**
Les rendus de type **inline**

Pourquoi cette distinction ?

Cette typologie dicte le comportement
en terme de **positionnement** et d'**affichage**

Les éléments de type block

- Des blocs dans les documents – Exemples : paragraphes, listes...
  - Apparaissent les **uns en dessous des autres**
  - Ont des **dimensions et des marges externes ou internes fixées** par défaut, **à l'exception des blocs **
  - Sont **positionnables** (avec les feuilles de style CSS)

Les éléments de type bloc

- Peuvent **contenir d'autres blocs sauf les blocs de paragraphes** (<p>) et de **titres** (<h1>, <h2>, … <h6>) **qui ne peuvent contenir d'autres blocs**

- Peuvent contenir des éléments inline

Exemples d'éléments de la « famille » block :

- <h1></h1> <h2></h2> … <h6></h6>
-
- <table></table>
-
-
- <blockquote></blockquote>
- <dl></dl>
-
- etc.

Question : quelles sont les particularités de la balise div ?

Différents rendus de type block

- display : block
- display : list-item
- display : table
- display : table-row

Les éléments inline :

- Apparaissent **au fil du texte**, ils ne sont pas placés les uns au dessus des autres (ils restent à l'emplacement défini).
- N'ont **pas de marges internes ou externes** par défaut
- Ne sont **pas dédiés à un positionnement précis** (même si cela est possible avec les CSS)
- Servent à modifier, enrichir (...) des portions de textes, apporter du sens
- Doivent être dans un bloc (**obligatoirement**)

Les éléments inline :

- Ne peuvent **contenir que des éléments inline (→ pas de block)**
- **Un élément inline doit être contenu dans un élément de type block**

Exemples d'éléments de la « famille » inline :

- <a>
-
- <img>
- <q></q>
- <samp></samp>
-
-

Question : quelles sont les particularités de la balise span ?

Différents rendus de type inline

- display : inline
- display : inline-block
- table-cell

Propriété CSS **display**

- passer d'un type de rendu à un autre

- Avec les valeurs block et inline
  - **display: bloc**
    = se comporter comme un élément de type block
  - **display: inline**
    = se comporter comme un élément de type inline

Rendus spécifiques de formes tabulaires

- display : table
- display : inline-table
- display : table-row
- display : table-row-group
- display : table-header-group
- display : table-footer-group
- display : table-column
- display : table-column-group
- display : table-cell
- display : table-caption

Un document XHTML comprend plusieurs parties

- Un prologue XML
- Un DOCTYPE
- Une déclaration de l'espace de noms et de la langue
- Un en tête
- Un corps

HTML5 propose une simplification de certains éléments de structure

Le prologue XML (uniquement en XHTML)

Spécifier la version de XML
Préciser le type d'encodage de caractères

Par exemple :

<?xml version="1.0" encoding="ISO-8859-1" ?>

=

Version 1.0 de XML
Caractères encodés en ISO-8859-1

Le tag DOCTYPE précise la DTD

DTD
=
Document Type Definition
ou
Définition de Type de Document

Précise en quelque sorte la « **grammaire** » utilisée pour rédiger le document

Utile pour les navigateurs, les lecteurs de votre code ou les validateurs de code

HTML & XHTML – Structure des documents

XHTML 1.0 Strict

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

XHTML 1.0 Transitionnel

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

XHTML 1.0 Frameset

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

XHTML 1.1

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

La déclaration de l'espace de noms et de la langue

Dans la balise <html> du document.

Un espace de noms (namespace) est, en XML, un ensemble de types d'éléments et de noms d'attributs associés à une DTD.

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

Nous déclarons ici que l'espace de noms est celui proposé à l'URL : "http://www.w3.org/1999/xhtml"

la langue est xml:lang="fr"

NB : lang="fr" est supprimé en XHTML 1.1

L'en-tête

- Délimité par les balises **<head> et </head>**
- Contient des informations non affichées par les navigateurs courants
  - Le titre
  - Les méta données
  - Les références à d'autres ressources
  - Le type d'encodage des caractères

L'en-tête

- Délimité par les balises **<head> et </head>**
- Contient des informations non affichées par les navigateurs courants
  - Le titre
  - Les méta données
  - Les références à d'autres ressources
  - Le type d'encodage des caractères

Le titre

<title>Mon document</title>

NB : le titre n'est pas directement affiché dans le document, mais souvent par le navigateur, dans l'interface

Devinette : je suis un internaute des plus assidus et l'on me qualifie souvent d'aveugle. Je lis pourtant le titre des pages avec grande attention. Qui suis-je ?

Les méta données

- Certaines méta données - **les mots clés et la description** - sont utilisées pour décrire le contenu du document et faciliter sa "visibilité" dans les moteurs de recherche, tout comme le titre.
- Certaines méta données permettent de donner des **informations / instructions aux moteurs de recherche :** leur indiquer s'il faut suivre les liens lors de l'indexation d'un site, la fréquence souhaitée pour la relecture des informations.
- D'autres méta données permettent de préciser **le nom de l'auteur, la version…**

Les références à d'autres ressources

Faire référence à d'autres ressources
utilisées par le document :
feuilles de style CSS,
fichiers de scripts externes Javascript...

Exemple :

```
<link rel="stylesheet" type="text/css"
href="../style/main.css" />
```

On fait référence à une feuille de style

Le type d'encodage des caractères

<meta http-equiv="content-type" content="text/html;charset=ISO-8859-1" />

Le jeu de caractères utilisé par le document

Pour un document en français on dispose des encodages suivants :

- iso-8859-1 : encodage classique pour les langues de l'Europe occidentale (aussi appelé Latin-1)
- iso-8859-15 : même encodage comportant quelques caractères supplémentaires comme œ, €…
- utf-8 : encodage pour les caractères de la majorité des langues mondiales

A noter :

En utilisant l'encodage iso-8859-1 ou iso-8859-15 les caractères ASCII 7-BIT (codes 32 à 127) sont valides, avec 4 exceptions.

Ces **exceptions sont codées avec des entités** :

" (codé ")

& (codé &)

< (codé <)

> (codé >)

Remarque : ce sont des caractères utilisés par le balisage XHTML.

En utilisant l'encodage iso-8859-1 ou iso-8859-15 :

Les caractères, en dehors de la classification ASCII 7-BIT (donc les codes de 128 à 255), sont codés par **des entités ou des références numériques** :

Par exemple :

é (codé é é)

è (codé è ou è)

à (codé à ou à)

ô (codé ô ou ,)

etc.

En utilisant l'encodage UTF-8 :

- On peut coder n'importe quelle langue
- On peut insérer n'importe quel caractère sans utiliser les entités
  - Sauf :
    - " (codé ")
    - & (codé &)
    - < (codé <)
    - > (codé >)

En utilisant l'encodage iso-8859-1 ou iso-8859-15 :

Les caractères, en dehors de la classification ASCII 7-BIT (donc les codes de 128 à 255), sont codés par **des entités ou des références numériques** :

Par exemple :

é (codé é é)

è (codé è ou è)

à (codé à ou à)

ô (codé ô ou ,)

etc.

Remarque :

le blanc insécable (pas de césure)
s'écrit

** **

Le type de contenu

<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" /> ou

<meta http-equiv="content-type" content="application/xhtml+xml; charset=ISO-8859-1" />

- Le HTML est servi en tant que HTML (donc text/html)
- Selon le W3C, le XHTML 1.0 peut être servi en tant que HTML (donc text/html) ou en tant que XHTML (donc application/xhtml+xml)
- Le XHTML 1.1, lui doit être servi uniquement en tant que XHTML

Le corps d'un document peut contenir divers éléments :

- du texte (titres, paragraphes, listes, etc.)
- des images
- des hyperliens
- des tableaux
- des formulaires
- des cadres
- des objets externes (applets Java, Flash …)
- etc.

Délimité par les balises <body> et </body>

HTML5 propose de nouveaux type d'éléments

Connaissez vous l'attribut target ?

A quoi sert-il principalement ?

Savez vous s'il est conforme avec les versions strictes des langages HTML et XHTML ?

Existe t-il des alternatives ?

Comment fait-on ?

div et span = éléments pour structurer
des documents Web
(en association avec les CSS)

div = élément de type bloc

span = élément en ligne.

n'apportent aucune contrainte de présentation, ils sont « neutres » à cet égard.

Ils servent à « ajouter » de la structure.

Attention, ces éléments n'ont pas de sens particulier, ils sont neutres également sur le plan de la sémantique. Par conséquent, ils ne doivent pas remplacer systématiquement les autres éléments.

Les tableaux

- **Marqueurs <table> </table>**
- Chaque ligne est encadrée par **<tr> </tr>**
- Les cellules d'en-tête sont encadrées par **<th> </th>**
- Les cellules de valeur sont encadrées par **<td> </td>**
- Les marrqueurs **<thead></thead> <tfoot></tfoot> <tbody></tbody>** permettent de structurer les tableaux
- La balise **<caption></caption>** permet d'indiquer la légende du tableau.
- Il est souhaitable d'ajouter à la balise <table> l'attribut **summary** = indiquer un résumé du tableau (<table summary="ce que contient le tableau">).

HTML & XHTML – Présentation de quelques éléments

Exemple

```
<table>
<caption>L&eacute;gende</caption>
<thead>
<tr>
<th>Cellule d'en t&ecirc;te A</th>
<th>Cellule d'en t&ecirc;te B</th>
</tr>
</thead>
<tfoot>
<tr>
<td>Cellule de pied de tableau A</td>
<td>Cellule de pied de tableau B</td>
</tr>
</tfoot>
```

...suite

```
<tbody>
<tr>
<td>Valeur A ligne 1</td>
<td>Valeur B ligne 1</td>
</tr>
<tr>
<td>Valeur A ligne 2</td>
<td>Valeur B ligne 2</td>
</tr>
</tbody>
</table>
```

Légende du tableau

Cellule d'en tête A	Cellule d'en tête B
Valeur A ligne 1	Valeur B ligne 1
Valeur A ligne 2	Valeur B ligne 2
Cellule de pied de tableau A	Cellule de pied de tableau B

Colonnes étendues

```
<table>
<tr>
<th colspan="2">Cellule d'en t&ecirc;te
    &eacute;tendue en largeur</th>
</tr>
<tr>
<td>Valeur A ligne 1</td>
<td>Valeur B ligne 1</td>
</tr>
<tr>
<td>Valeur A ligne 2</td>
<td>Valeur B ligne 2</td>
</tr>
</table>
```

L'attribut colspan crée des cellules qui s'étendent sur plusieurs cellules d'un tableau, en ligne

Cellule d'en tête étendue en largeur	
Valeur A ligne 1	Valeur B ligne 1
Valeur A ligne 2	Valeur B ligne 2

Lignes étendues

```
<table>
<tr>
<th>Cellule d'en t&ecirc;te A</th>
<th>Cellule d'en t&ecirc;te B</th>
</tr>
<tr>
<td rowspan="2">Valeur A ligne 1 et 2
    (&eacute;tendu)</td>
<td>Valeur B ligne 1</td>
</tr>
<tr>
<td>Valeur B ligne 2</td>
</tr>
</table>
```

L'attribut rowspan crée des cellules qui s'étendent sur plusieurs lignes d'un tableau

Cellule d'en tête A	Cellule d'en tête B
Valeur A ligne 1 et 2 (étendu)	Valeur B ligne 1
	Valeur B ligne 2

Liens

- Lien vers une autre page - même site - même répertoire
  Lien
- Lien vers une autre page - même site - autre répertoire
  Lien
- Lien vers une autre page - nouvelle fenêtre du navigateur
  <a href="exemples/page1.html »
  target="_blank">Lien

  Attention, l'attribut target n'est pas valide pour les versions strictes de XHTML (et HTML). Il ne semble pas y avoir d'alternative proposée par le langage actuellement. Il est possible d'utiliser le JavaScript pour ouvrir une nouvelle fenêtre sur le click d'un lien, mais cette alternative présente d'autres inconvénients

Liens

- Lien vers un document
  Doc texte
- Liens vers un document d'un autre site
  Un site
  Un site
- Lien vers un fragment du document courant
  Lien
  Chaque marqueur de "fragment" est identifié par un signet de la forme Sommet
- Lien textuel vers un fragment d'un autre document
  Lien

Liens

- Lien sous forme d'image vers une autre page

```
<a href="index.html"><img border="0" src="../images/i.gif"
    width="16" height="18" /></a>
```

- Les images cliquables (ou réactives)

```
<map id="map1">
<area href="1.html" shape="rect" coords="0,0,100,100" />
<area href="2.html" shape="rect" coords="100,0,200,100" />
<area href="3.html" shape="rect" coords="0,100,100,200" />
<area href="4.html" shape="rect" coords="100,100,200,200" />
</map>
<img src="../images/map.gif" usemap="#map1" width="200"
    height="200" />
```

Chemins relatifs

- Descendre dans l'arborescence, vers un sous répertoire :
  nom_sous_repertoire/ ou ./nom_sous_repertoire/
  ./ représentant le positionnement courant
- Remonter dans l'arborescence :
  ../
- Remonter de plusieurs niveaux :
  ../../../
- Remonter puis de redescendre dans l'arborescence :
  ../../autre_repertoire/autre_sous_repertoire

Chemins relatifs ou absolus ?

1 document est positionné dans une **arborescence de répertoires et de fichiers**.

Le répertoire de plus haut niveau (contenant tous les autres documents ou fichiers), est appelé la « **racine** ».

On peut exprimer la destination d'un lien de manière **absolue (depuis la racine)** :

/rep/sousrep/index.html

Le problème est que si je déplace l'ensemble des documents, par exemple pour les inclure dans un répertoire placé différemment par rapport à la racine, les chemins ne seront plus valides.

Les images

Marqueur **<img>**

Ses principaux attributs sont :

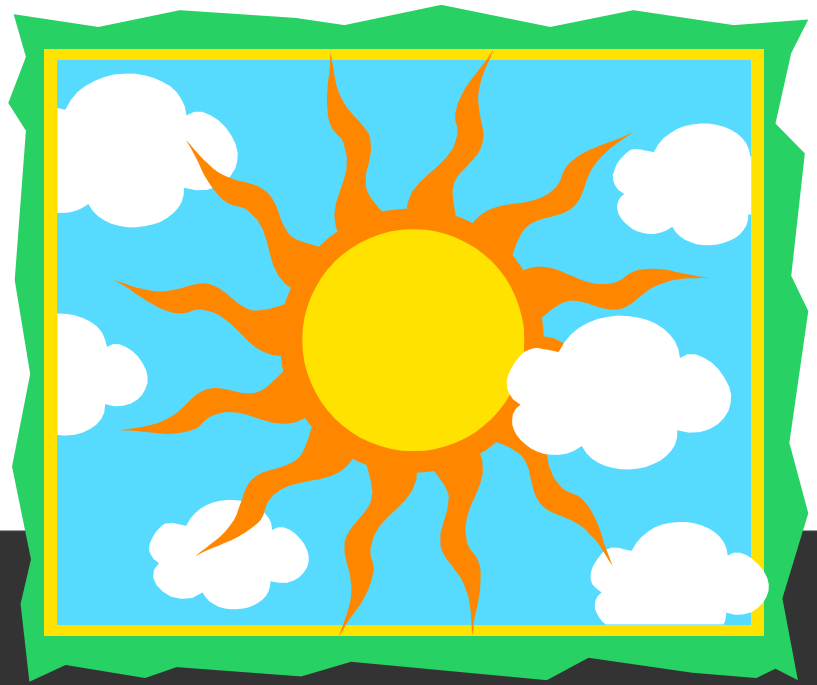
- src : emplacement du fichier source de l'image
- width : largeur
- height : hauteur
- alt : texte qui apparaît lorsque l'image ne s'affiche pas et comme info bulle de l'image

```
<img src="../../images/chose.gif" width="16"
     height="18" alt="exemple d'image" />
```

Principaux formats d'images

GIF (Graphic Interchange Format)

- Pour les graphismes aux tracés simples, sans dégradé de couleurs
- Limité à une palette de 256 couleurs (choisies parmi des millions).
- Peut être animé.
- Gère la transparence.



Principaux formats d'images

JPG OU JPEG (Joint Picture Expert Group)

- Adapté aux photos ou aux images présentant de nombreux dégradés
- Gère la compression (à perte)

Principaux formats d'images

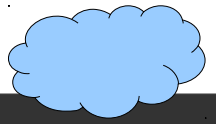
PNG (Portable Network Graphic)

- A l'origine : format alternatif au GIF
- Peu remplacer le GIF (éventuellement le JPG)
- Méthode de compression améliorée
- Attention, il n'est pas supporté par tous les navigateurs (versions très anciennes).
- La transparence peut poser quelques soucis avec certaines versions de navigateurs

Les formulaires

Marqueurs <form> et </form>

- Principaux attributs :
  - method : sous quelle forme seront envoyées les réponses (POST | GET)
  - action : l'adresse d'envoi (script ou adresse E-mail)
  - enctype (facultatif) : codage des données dans l'URL. Généralement, il n'est pas nécessaire de le préciser car la valeur habituelle est attribuée par défaut (application/x-www-form-urlencoded).



Pour des formulaires d'upload de fichiers (téléchargement de fichier depuis le PC du visiteur vers le serveur), par exemple, il faudra indiquer un enctype="multipart/form-data"

Exemples de balises form

```
<form id="monformulaire" method="post"
action="../rep/traitement.php">

<form method="post"
action="http://domaine/cgi-bin/script.pl">
```

Question : pourquoi, dans le premier exemple, utilise t-on l'attribut id et pas l'attribut name ? A quoi peut-il servir ?

Les formulaires

= "**conteneurs**" permettant de **regrouper des éléments** qui vont permettre à l'utilisateur de **choisir ou de saisir des données**, qui seront **envoyées à l'URL** indiqué dans l'attribut "**action**" selon la méthode indiquée.

Eléments "interactifs", de saisie (et d'affichage) de données ou de choix :

- Elément **input** : différents boutons et champs de saisie.
- Elément **textearea** : zone de saisie de texte.
- Elément **select** : liste à choix unique / multiples.

Les formulaires – Champs de type input

<input type="TypeDeChamp" value="ValeurParDefautOuVide" name="NomDuChamp" [ProprieteSupplementaire] />

L'**attribut type** => type d'élément :

- **texte** : saisie d'une ligne de texte
- **password** : champ de saisie de mot de passe
- **checkbox** : cases à cocher
- **radio** : boutons radios
- **file** : fichier qui sera envoyé avec le formulaire
- **hidden** : champ caché.
- **reset** : bouton de remise à zéro ("rétablir" à l'état d'origine)
- **submit** : bouton de soumission permettant l'envoi du formulaire
- **image** : image jouant le rôle d'un bouton de soumission

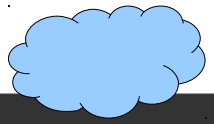
Pour une illustration et les règles de syntaxe, vous reporter au polycopié D'accompagnement du cours

Les formulaires – Envoi de données

Les données saisies ou sélectionnées dans le formulaire sont envoyées au script de traitement sous forme de

Paires nom / valeur

Nom (name) de l'élément de formulaire / valeur associée



Connaissez vous la différence entre les méthodes GET et POST ?

Les formulaires – Envoi de données

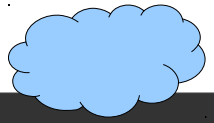
Selon la méthode, les modalités d'envoi au serveur sont différentes :

- GET envoie des éléments du formulaire comme un URL:

  http://site/rep/script.php?nom1=valeur1&nom2=valeur2…
  NB : la longueur de la chaîne URL est limitée

- POST envoie les données à la suite des en-têtes HTTP, dans le corps de la requête

  La quantité de données envoyées n'est plus limitée de la même manière



Quelle méthode privilégier ?

Les frameset (jeu de cadres) et frames (cadres)

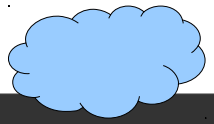
Afficher **plusieurs documents au sein d'une même fenêtre**, dans des cadres (frames).

Un fichier "parent" contient l'agencement des cadres

Il **ne possède pas de <body>** (corps de document), mais un <frameset> (jeu de cadres)

Exemple avec 2 cadres verticaux :

```
<frameset cols="10%,90%">
<frame src="f1.html" name="gauche" />
<frame src="f2.html" name="droite" />
</frameset>
```



Cette technique n'est généralement pas recommandée pour des question d'ergonomie, d'accessibilité, de maintenance…

Principaux attributs de la balise <frameset>

- **rows** : taille relative des cadres verticalement.
  % (entre 1 et 100) ou valeur en pixels. Il est possible d'utiliser le caractère * pour spécifier que l'on prend la taille restante <frameset cols="10%,*">
- **cols** : taille relative des cadres horizontalement.
  % (entre 1 et 100) ou valeur en pixels. Il est possible d'utiliser le caractère * pour spécifier que l'on prend la taille restante <frameset cols="10%,*">
- **frameborder** (yes ou no) : indique si les cadres ont une bordure ou non

NB : liste non exhaustives des attributs & de nombreux attibuts des frames ou framesets sont dépréciés

Principaux attributs de la balise <frame>

- **src** : URL du fichier source du cadre
- **name** : nom du cadre
- **noresize** : interdit à l'utilisateur de redimensionner les cadres
- **scrolling** (yes no auto) : permet ou non l'affichage d'une barre de défilement (auto signifie que la barre de défilement est affichée uniquement lorsque nécessaire)

NB : liste non exhaustives des attributs & de nombreux attibuts des frames ou framesets sont dépréciés

Imbrication de framesets

```
<frameset rows="20%, 80%">
        <frame src="f1.html" name="haut" noresize
          scrolling="no" />
        <frameset cols="20%, 80%">
              <frame src="f2.html" name="menu"
                noresize scrolling="no" />
              <frame src="f3.html" name="principal”
                scrolling="yes" />
        </frameset>
</frameset>
```

<iframe> = cadre incorporé

Inclure un cadre dans le corps d'un document.

On parle de cadre incorporé ou iframe (iframe = inline frame = cadre incorporé)

balise <iframe>

Privilégier XHTML à HTML ?

- Les documents XHTML sont conformes à XML, donc lisibles, ou éditables avec les outils XML.
- XHTML est conçu pour fonctionner à la fois avec les anciens navigateurs et les navigateurs récents. La compatibilité avec les futurs standards est bien plus probable.
- La rigidité de XHTML est plutôt une qualité qu'un défaut. L'écriture en XHTML « pousse à » adopter les normes actuelles de conception Web.

NB : l'arrivée de HTML5 ne remet elle pas en cause ces questions (au moins les 2 premières) ?

Valider ses documents ?

- Etre sûr de la syntaxe et de la grammaire utilisée et se conformer à la norme d'encodage
- Indirectement, assurer une meilleure interprétation du code par les navigateurs (ils tendent à être de plus en plus en phase avec les normes de codage)
- Indirectement, assurer une meilleure pérennité du code pour les navigateurs à venir
- Un document valide sera sans doute plus facile à faire évoluer vers les futures normes

NB : l'arrivée prochaine de HTML5 ne remet elle pas en cause ces questions (au moins les 3 dernières) ?

Valider ses documents ?

La validation des documents, même si elle est **nécessaire**, n'est **pas suffisante** : seule la validité de la syntaxe est vérifiée, en aucun cas la qualité structurelle ou sémantique du document.

Séparer contenu et structure vs présentation

Contenu et structure = HTML ou XHTML
Présentation = CSS

- Les 2 peuvent évoluer séparément
- On peut modifier le style de plusieurs documents de manière simultanée
- La lecture du code des documents est grandement facilitée
- Le même document peut être lu sur différents supports

Quelques remarques en guise de conclusion

Adopter un balisage structurel et sémantique

- Objectif de XHTML : décrire le contenu d'un document et le structurer
  - Adopter un balisage sémantique (un paragraphe est un paragraphe, un titre est un titre…)
  - Dépasser le seul aspect « présentation » dans les documents : interprétables autrement que dans un navigateur « standard » (moteurs de recherche, lecteurs braille…)
    = documents ouverts à d'autres type de supports
  - Clairement structurer les documents (haut de page, menu, pied de page...) :
    - Code généralement allégé, + lisible, + clair, maintenance + facile
    - Refontes graphiques / changements de mise en page largement facilités