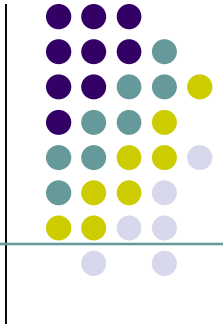
Université Gaston Berger de Saint-Louis
Année académique 2016-2017



XML

Le standard de stockage et d'échange de données

Références

1. XML Programming Success in a Day: Beginner's Guide to Fast, Easy, and Efficient Learning of XML Programming. Sam Key, 2015.

2. XML Programming: The Ultimate Guide to Fast, Easy, and Efficient Learning of XML Programming. Christopher Right, 2015

3. Beginning XML. Joe Fawcett and Danny Ayers, 2012.

4. XML, Cours et exercices. Modélisation, Schémas et DTD, design patterns, XSLT, DOM, Relax NG, XPath, SOAP, XQuery, XSL-FO, SVG, eXist. *Alexandre Brillant*, Édition : Eyrolles 2eédition, 2010

5. Schémas XML, Jean-Jacques Thomasson, Edition Eyrolles 2002

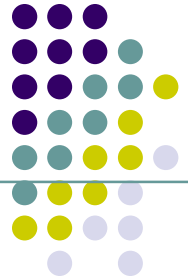
6. World Wide Web Consortium (W3C) : w3.org

7. World Wide Web School : w3schools.com/xml

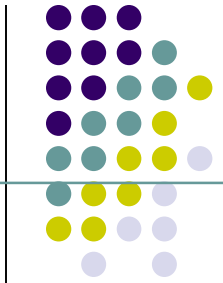
8. Xml.developpez.com

9. Youtube.com

Sommaire



- Introduction
- Structure d'un document XML
- Règles de syntaxe
- Les éléments
- Les attributs
- Les espaces de nom (namespaces)
- Visualisation d'un doc XML
- L'objet XMLHttpRequest
- Le parseur XML
- Le DOM XML
- Definition de type de doc XML
- Technologies XML
- XML dans le Server



Introduction

Qu'est-ce que XML?

- XML signifie eXtensible Markup Language
- XML est un langage de balisage comme le HTML
- XML a été conçu pour le transport et le stockage de données, et pas pour l'affichage
- Les balises XML ne sont pas prédéfinies. On les définit soi-même,
- XML est conçu pour être auto-descriptif
- XML est une recommandation du W3C
- Pré-requis: HTML et Javascript

Introduction

Exemple

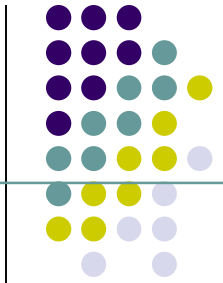
```
<?xml version="1.0" encoding="UTF-8"?>
<notebook>
    <note>
      <a>Caare</a>
      <de>Ngeer</de>
      <entete>Rappel</entete>
      <corps>N'oublies pas le cours de Xml!</corps>
    </note>
    <note>
      <a>saaliwu Géy</a>
      <de>Ngeer</de>
      <entete>Information</entete>
      <corps>Rencontre à la salle B17 à 10h30</corps>
    </note>
<notebook>
```

Introduction

Que fait XML?

- XML ne fait rien.
- Il décrit juste de l'information en utilisant des balises
- Il faut un morceau de logiciel pour envoyer, recevoir ou afficher des données XML.

Introduction



Qu'est-ce que XML simplifie?

- Il simplifie le partage des données:

De nombreux systèmes informatiques contiennent des données dans des formats incompatibles. L'échange de données entre les systèmes incompatibles (ou systèmes mis à niveau) est une tâche fastidieuse pour les développeurs web. De grandes quantités de données doivent être converties, et les données incompatibles sont souvent perdues.

- It simplifie le transport des données

XML stocke les données au format texte brut. Cela fournit, aux logiciels et matériels, un moyen indépendant de stockage, de transport et de partage des données.

Introduction

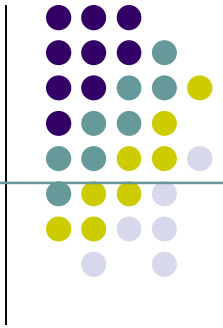
Qu'est-ce que XML simplifie?

- Il simplifie les changements de plate-forme

XML rend également plus facile l’extension ou la mise à niveau vers de nouveaux systèmes d'exploitation, de nouvelles applications, de nouveaux navigateurs, sans perdre de données.

- Il simplifie la disponibilité des données

Avec XML, les données peuvent être disponibles pour toutes sortes de "machines de lecture» comme les humains, les ordinateurs, les machines vocales, les fils rss, etc.



Introduction

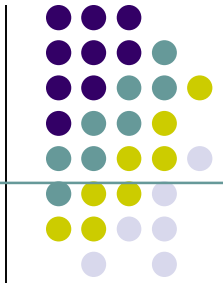
Pourquoi étudier XML?

- XML joue un rôle important dans de nombreux systèmes informatiques.

- C'est pourquoi, il est important pour tous les développeurs de logiciels d'avoir une bonne connaissance de XML.

- Apprendre XML nécessite une connaissance de base en :
  - HTML
  - CSS
  - JavaScript

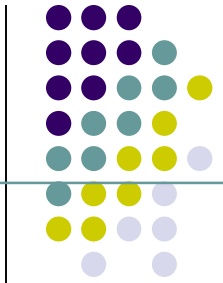
Introduction



La différence entre XML et HTML

- XML n'est pas un remplaçant de HTML.
- XML et HTML ont été conçus avec des objectifs différents:
  - XML est conçu pour transporter des données - en mettant l'accent sur ce que sont les données
  - HTML est conçu pour afficher des données - en mettant l'accent sur la façon dont les données sont vues
  - Les balises XML ne sont pas prédéfinies comme dans HTML.

Introduction



XML utilise-t-il des balises prédéfinies?

- Le langage XML n'a pas balises prédéfinies.
- Les mots-clés dans l'exemple ci-dessus (comme <a> et <de>) ne sont pas définis dans une norme XML. Ces balises sont «inventés» par l'auteur du document XML.
- HTML fonctionne avec des balises prédéfinies comme <p>, <h1> <table>, etc.
- Avec XML, l'auteur doit définir à la fois les balises et la structure du document.

Introduction

XML est extensible

- La plupart des applications XML fonctionneront comme prévu, même si de nouvelles données sont ajoutées (ou supprimées).

  Exemple : Considérons une application conçue pour afficher la version originale de note.xml (<a> <de> <entete> <donnees>).

  Imaginez une version plus récente de note.xml avec ajout des éléments <date> et <heure>, et l'élément <titre> supprimé..

- Avec la façon dont XML est construit, une ancienne version de l'application pourra encore fonctionner.

Introduction

XML est extensible

Exemple

```
<note>
  <a>Caare</a>
  <de>Ngeer</de>
  <entete>Reminder</entete>
  <corps>Don't forget me the
XML course!</corps>
</note>
```

```
<note>
  <date>2016-02-21</date>
  <heure>08:30</heure>
  <a>Caare</a>
  <de>Ngeer</de>
  <corps>Don't forget me the
XML course!</corps>
</note>
```

Introduction

XML est une recommandation du W3C depuis le 10 Février 1998.

Qu'est-ce qu'une «recommandation du W3C»?
W3C publie des documents qui définissent les technologies Web. Ces documents suivent un processus visant à promouvoir le consensus, l'équité, la responsabilité publique et la qualité.

A la fin de ce processus, le W3C publie des recommandations, qui sont considérés comme des standards Web.

Le **Consortium World Wide Web** (W3C) est une communauté internationale où les organisations membres, une équipe à temps plein, et le public travaillent ensemble au développement de standards Web. Dirigé par l'inventeur du Web Tim Berners-Lee et son PDG Jeffrey Jaffe, la mission du W3C est de mener le Web à son plein potentiel. (Site Web: www.w3.org)

Introduction

Utilisation de XML

XML est utilisé dans de nombreux aspects du développement web souvent pour séparer les données de la présentation.

Séparer les Données de la Présentation

- XML ne comporte pas d'informations sur la façon d'être affichée.
- Les mêmes données XML peuvent être utilisés dans de nombreux scénarios de présentation.
- De ce fait, avec XML, il y a une séparation complète entre les données et la présentation.

Introduction

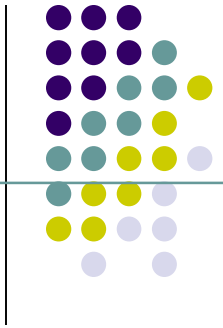
Utilisation de XML

Complément à HTML

Dans de nombreuses applications HTML, XML est utilisé pour stocker ou transporter des données, alors que HTML est utilisé pour formater et afficher les mêmes données.

Séparer les Données de HTML

- Lors de l'affichage des données au format HTML, vous ne devriez pas avoir à modifier le fichier HTML lorsque les données changent.
- Avec XML, les données peuvent être stockées dans des fichiers XML séparés.
- Avec quelques lignes de code JavaScript, vous pouvez lire un fichier XML et mettre à jour le contenu des données d'une page HTML.



Introduction

Utilisation de XML

Données de transaction

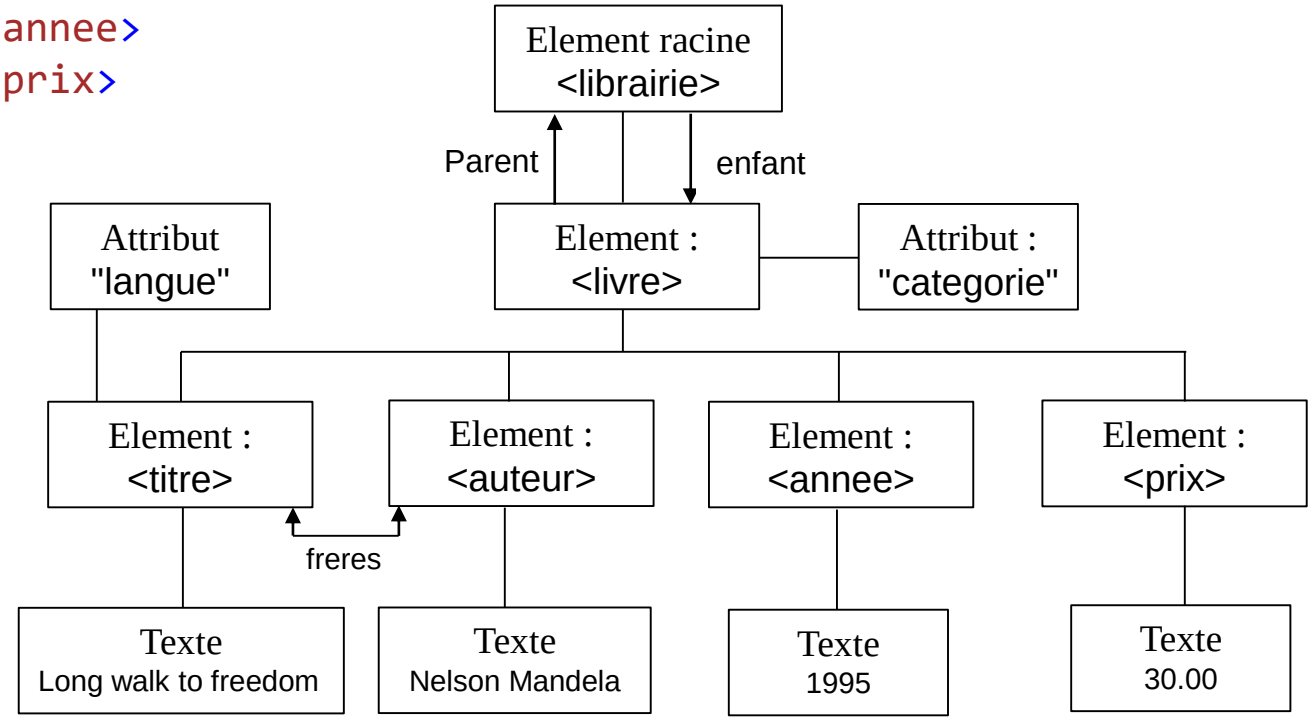
Des milliers de formats XML existe, dans de nombreuses industries différentes, pour décrire les transactions de données au jour le jour:

- Des actions et parts de marché
- Des transactions financières
- Des données médicales
- Des données mathématiques
- Des mesures scientifiques
- De nouvelles informations: Ex http://www.xmlnews.org
- Des services météorologiques

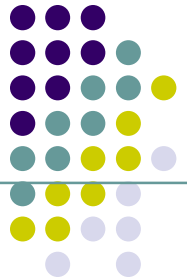
Structure d'un document XML

Les documents XML forment une structure arborescente qui commence à "la racine" et se ramifie en des «feuilles».

```
<?xml version="1.0" encoding="UTF-8"?>
<librairie>
  <livre categorie="Histoire">
    <titre langue="en">Long walk to freedom</titre>
    <auteur>Nelson Mandela</auteur>
    <annee>1995</annee>
    <prix>30.00</prix>
  </livre>
</librairie>
```



Structure d'un document XML



```
<?xml version="1.0"  encoding= "UTF-8" standalone= "no" ?>
<!DOCTYPE  exemple  SYSTEM   "exemple.dtd">
<librairie>
  <livre categorie="Réseau">
    <titre lang="fr">Introduction au réseau mobile</ titre >
    <auteur>Alassane Diop</auteur>
    <annee>2011</annee>
    <prix>30.00</prix>
  </livre >
  <!--Un deuxieme livre-->
  <livre categorie="Compilation">
    <titre lang="en">Théorie des langages et automates</ titre >
    <auteur>E. M. Nguer</auteur>
    <annee>2012</annee>
    <prix>29.99</prix>
    <image src="/images/la.jpg" />
  </livre >
</librairie>
```

Entête

Elément racine

Elément

Commentaire

Attribut

Elément vide

Contenu

Structure d'un document XML

Les documents XML sont formés sous forme d'**arbres d'éléments**. Un arbre XML commence par un **élément racine** et se ramifie de la racine aux **éléments enfants**. Tout élément peut avoir des sous-éléments (éléments enfants):

```
<racine>
  <enfant>
    <sousenfant>.....</sousenfant>
  </enfant>
</racine>
```

- Les termes parent, enfant, frère sont utilisés pour décrire les relations entre les éléments.
- Les parents ont des enfants. Les enfants ont des parents. Les frères sont des enfants ayant le même niveau.
- Un élément peut avoir un contenu texte (Nelson Mandela) et des attributs (categorie="History").

Structure d'un document XML

Syntaxe auto-descriptive

XML utilise beaucoup une syntaxe auto-descriptive.
Le prologue définit la version XML et le codage des caractères:

```
<?xml version="1.0" encoding="UTF-8"?>
```

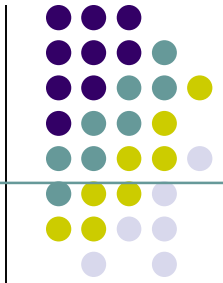
La ligne qui suit est l'**élément racine** du document:

```
<librairie>
```

La ligne suivante démarre un élément <livre>:

```
<livre categorie="Histoire">
```

Structure d'un document XML



Syntaxe auto-descriptive

Les éléments <livre> ont **4 sous elements**: <titre>,< auteur>, <annee>, <prix>.

```
<titre langue="en">Long walk to freedom</titre>
<auteur>Nelson Mandela</auteur>
<annee>2015</annee>
<prix>30.00</prix>
```

La ligne suivante termine l'élément : `</livre>`

Vous pouvez supposer, à partir de cet exemple, que le document XML contient des informations sur les livres dans une librairie.

Règles de syntaxe

Les règles de syntaxe de XML sont très simple et logique. Elles sont faciles à apprendre et à utiliser.

- **Un documents XML doit avoir un élément racine**

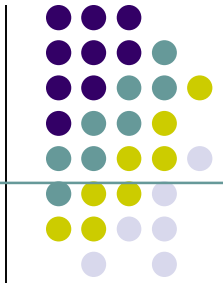
Cet élément **racine** est le **parent** de tous les autres éléments:

```
<racine>
  <enfant>
    <sousenfant>...</sousenfant>
  </enfant>
</racine>
```

Dans cet exemple, **<note>** est l'élément racine:

```
<?xml version="1.0" encoding="
UTF-8"?>
<note>
  <a>Caare</a>
  <de>Ngeer</de>
  <entete>Reminder</entete>
  <corps>Don't forget the XML
course!</corps>
</note>
```

Règles de syntaxe



Le prologue XML

La ligne `<?xml version="1.0" encoding="UTF-8"?>`
est appellée le **prologue** du document XML:

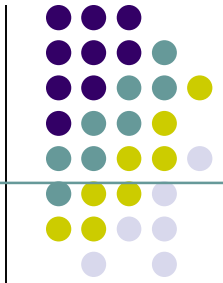
- Il est optionnel. S'il existe, il doit être en début du document.

- Un document XML peut contenir des caractères internationales, comme des caractères wolof ŋó, norvégiens øæå ou français êèé.

  - Pour éviter des erreurs, vous devez spécifier l'encodage utilisé, ou sauvegarder vos fichiers XML en `UTF-8`.

  - `UTF-8` est l'encodage par défaut des caractères pour les documents XML, HTML5, CSS, JavaScript, PHP et SQL.

Règles de syntaxe



Balise de fermeture XML

- En HTML, certains éléments peuvent bien fonctionner, même avec une balise de fermeture manquante:

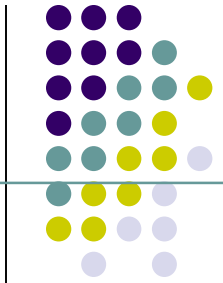
```
<p>Ceci est un paragraphe.
<br>
```

- En XML, il est illégal d'omettre la balise de fermeture. Tous les éléments doivent avoir une balise de fermeture:

```
<p>Ceci est un paragraphe.</p>
<br />
```

- Le prologue XML n'a pas de balise de fermeture. Ce n'est pas une erreur, il fait pas partie du document XML.

Règles de syntaxe



Sensiblité à la casse des balises XML

- Les balises XML sont sensibles à la casse. La balise <Lettre> est différente de la balise <lettre>.
- Les balises d'ouverture et de fermeture doivent être écrites avec la même casse:

```
<Message>Ceci est incorrect </message>

<message>Ceci est correct </message>
```

"Les balises d'ouverture et de fermeture" sont souvent appelées "Balises de début et de fin".

Utilisez ce que vous préférez. C'est exactement la même chose.

Règles de syntaxe

L'imbrication des éléments en XML

- En HTML, on peut avoir des éléments mal imbriqués :

```
<b><i>Ce texte est en gras et en italique</b></i>
```

- En XML, tous les éléments doivent être correctement imbriqués les uns dans les autres :

```
<b><i>Ce texte est en gras et en italique</i></b>
```

Dans l'exemple ci-dessus, "correctement imbriqué" signifie simplement que, puisque l'élément <i> est ouvert à l'intérieur de l'élément<b>, il doit être fermé à l'intérieur de l'élément <b>.

Règles de syntaxe

Citation des valeurs d'attribut

- Les éléments XML peuvent avoir des attributs sous forme de paires nom/valeur comme en HTML.
- En XML, les valeurs d'attributs doivent toujours être cités.

Incorrect

```
<note date=12/11/2007>
  <a>Jaxabi</a>
  <de>Jaañ</de>
</note>
```

L'erreur dans ce document est que la valeur de l'attribut date dans l'élément note n'est pas citée.

Correct

```
<note date="12/11/2007">
  <a>Jaxabi</a>
  <de>Jaañ</de>
</note>
```

Règles de syntaxe

Références d'entité

- Certains caractères ont une signification particulière en XML. Si vous placez un caractère comme "<" à l'intérieur d'un élément XML, il va générer une erreur parce que l'analyseur l'interprète comme le début d'un nouvel élément.

  Exemple : `<message>`salaire < 1000`</message>`

- Pour éviter cette erreur, on remplace le caractère "<" par une **référence d'entité** : `<message>salaire < 1000</message>`

Il y a 5 références d'entité prédéfinies en XML:

Seuls < et & sont strictement illégale en XML, mais il est une bonne habitude de remplacer > par &gt.

<	<	less than	Inferieur
>	>	greater than	superieur
&	&	ampersand	esperluette
'	'	apostrophe	apostrophe
"	"	quotation mark	Guillemet

Règles de syntaxe

Commentaires en XML

- La syntaxe des commentaires en XML est similaire à celle en HTML.

  <!--Ceci est un commentaire-->

  Deux tirets dans un commentaire n'est pas autorisé.

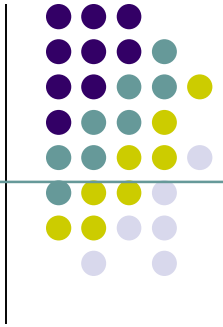
  Non autorisé: `<!-- Ceci est un -- commentaire -->`

  Étrange, mais admis: `<!--Ceci est un - - commentaire -->`

Espace blanc en XML

XML ne tronque pas plusieurs espaces blancs (HTML tronque plusieurs espaces blancs en un espace blanc simple):

XML:	Siyaar Bàmba
HTML:	Siyaar Bàmba



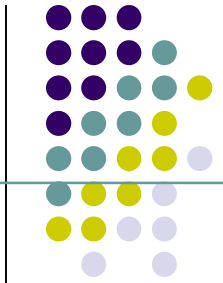
Règles de syntaxe

Caractère de fin de ligne en XML

- Les applications Windows represente une nouelle ligne par CR+LF: carriage return and line feed .
- Unix et Mac OSX utilise LF.
- Les anciens systemes Mac utilisent CR.
- XML représente une nouvelle ligne par LF.

Document XML bien formé

Un document XML conforme aux règles de syntaxe énoncées ci-dessus est dit "bien formé".



Les éléments

Un document XML contient des éléments XML.

Qu'est-ce qu'un élément XML?

Un élément XML est tout ce qui est compris entre une balise de début et sa balise de fin (y compris la balise).

```
<prix>29.99</prix>
```

Un élément XML peut contenir :

- du texte
- des attributs
- d'autres éléments
- ou un mélange de ce qui précède

Les éléments

```
<?xml version="1.0"  encoding= "UTF-8" standalone= "no" ?>
<!DOCTYPE  exemple  SYSTEM   "exemple.dtd">
<librairie>
  <livre categorie="Réseau">
    <titre lang="fr">Introduction au réseau mobile</ titre >
    <auteur>Alassane Diop</auteur>
    <annee>2011</annee>
    <prix>30.00</prix>
  </livre >
  <!--Un deuxieme livre-->
  <livre categorie="Compilation">
    <titre lang="en">Théorie des langages et automates</ titre >
    <auteur>E. M. Nguer</auteur>
    <annee>2012</annee>
    <prix>29.99</prix>
    <image src="/images/la.jpg" />
  </livre >
</librairie>
```

Elément racine

Elément

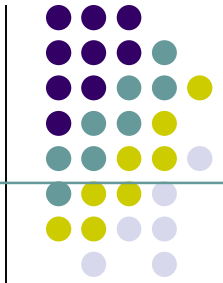
Commentaire

Attribut

Elément vide

Entête

Contenu



Les éléments

Dans l'exemple ci-dessus :

- <titre>, <auteur>, <annee> et <prix> ont du **contenu texte**, car ils contiennent du texte (comme 29.99).
- <librairie> et <livre> ont des contenus éléments, car ils contiennent des éléments.
- <livre> a un attribut (categorie = "Réseau")

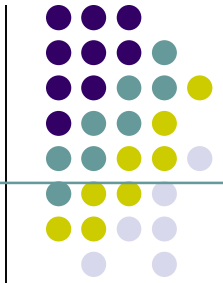
Éléments XML vides

Un élément sans contenu est dit vide. En XML, on peut indiquer un élément vide comme suit :

```
<element></element>
```

On peut également utiliser une balise dite auto-fermée: `<element />`
Les deux formes produisent des résultats identiques dans un logiciel XML (Lecteurs, Parseurs, Navigateurs). Un éléments vide peut avoir des attributs.

Les éléments

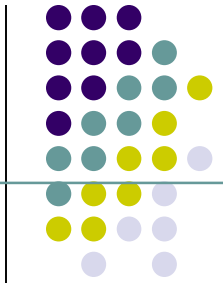


Règles de nommage en XML

Un élément XML doit suivre les règles de nommage suivantes:

- Il est sensible à la casse
- Il doit commencer par une lettre ou un trait de soulignement
- Il ne peut pas commencer par les lettres xml (ou XML, ou Xml, etc)
- Il peut contenir des lettres, des chiffres, des tirets de soulignement, et des signes de ponctuation.
- Il ne peut pas contenir d'espaces

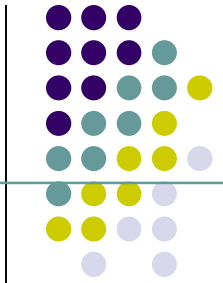
Tout nom peut être utilisé, pas de mots sont réservés (**sauf xml**).



Les éléments

Meilleures pratiques en nommage

- Créer des noms descriptifs comme ceci: <personne>, <prenoms>
- Créer des noms courts et simples comme ceci: <titre_livre> et pas comme: <le_titre_du_livre>.
- Eviter "-". Si vous nommez quelque chose par "titre-livre", un logiciel peut penser que vous voulez soustraire "livre" de "titre".
- Eviter ".". Si vous nommez quelque chose par "titre.livre", un logiciel peut penser que "livre" est une propriété de l'objet "titre".
- Eviter":". Les : sont réservés aux espaces de noms (plus tard).
- Les lettres non anglaises comme éòá sont parfaitement légales en XML, mais attention aux problèmes si votre logiciel ne les prend pas en charge.



Les éléments

Styles de nommage

- Il n'y a pas de styles de nommage définis pour les éléments XML. Mais voici les couramment utilisés:

Style	Example	Description
Lower case	<firstname>	Toutes les lettres minuscules
Upper case	<FIRSTNAME>	Toutes les lettres majuscules
Underscore	<first_name>	Le trait de soulignement sépare les mots
Pascal case	<FirstName>	Premiere letter de chaque mot en majuscule
Camel case	<firstName>	Premiere letter de chaque mot en majuscule excepté le premier le mot.

- Si vous choisissez un style de nommage, il est bon d'être cohérent! Les documents XML ont souvent une base de données correspondante. Une pratique courante consiste à utiliser les règles de nommage de la base de données pour les éléments XML.
- Le style Camel case est une règle de nommage commune en JavaScript.

Les éléments

Extensibilité des éléments

Les éléments XML peuvent être étendus pour transporter plus d'informations.

Exemple:

```
<note>
  <a>Seex Saar</a>
  <de>Jéen</de>
  <corps>Don't forget the travel!</corps>
</note>
```

Imaginons que nous avons créé une application qui extrait les éléments <a>, <de> et <corps> du document XML pour produire la sortie suivante:

MESSAGE
To: Seex Saar
From: Jéen
Don't forget the travel!

Les éléments

Extensibilité des éléments

- Imaginez que l'auteur du document XML y ajoute quelques informations supplémentaires :

```
<note>
  <date>2008-01-10</date>
  <a>Seex Saar</a>
  <de>Jéen</de>
  <entete>Reminder</entete>
  <corps>Don't forget the travel!</corps>
</note>
```

- Est-ce que l'application va planter ?
- Non. Elle doit toujours être en mesure de trouver les éléments <a>, <de> et <corps> dans le document XML et produire la même sortie.
- Ceci est l'une des beautés de XML. Il peut être étendu sans planter les applications.

Les attributs

Les éléments XML peuvent avoir des attributs, tout comme HTML.
Les attributs sont conçus pour contenir des données relatives à un élément spécifique.

L'attributs XML doit être citer

Les valeurs des attributs doivent toujours être cités soit par des apostrophes ('), soit par des guillemets (").

Pour le genre d'une personne, l'élément <person> peut être écrit ainsi :

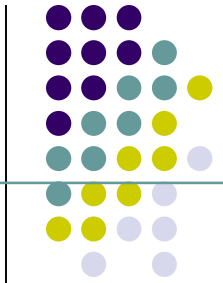
```
<personne genre="femelle">
```
ou comme suit:
```
<personne genre='femelle'>
```

Si la valeur de l'attribut elle-même contient des guillemets, vous pouvez utiliser des apostrophes, comme dans cet exemple:

```
<king name='Lat Joor "Ngóone Laatir" Jóob'>
```
Ou des entités :
```
<king name='Lat Joor &quot;Ngóone Latiir&quot; Jóob'>
```


E. M. NGUER – emnguer@ugb.edu.sn

Les attributs



Elements contre Attributs

Voyons les exemples ci-dessous:

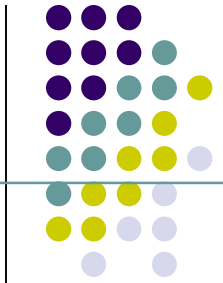
```
<person gender="female">
  <firstname>Aana</firstname>
  <lastname>Seen</lastname>
</person>

<person>
  <gender>female</gender>
  <firstname>Aana</firstname>
  <lastname>Seen</lastname>
</person>
```

Dans le premier exemple genre est un attribut. Dans le dernier, le genre est un élément. Ces deux exemples donnent les mêmes informations.

Il n'y a pas de règles lorsqu'il faut utiliser des attributs et lorsque qu'il faut utiliser des éléments en XML.

Les attributs



Une façon préférée

Les trois documents XML suivants contiennent exactement les mêmes informations:

Un attribut de date est utilisé dans le premier exemple:

```
<note date="2008-01-10">
  <to>Faal</to>
  <from>Jaañ</from>
</note>
```

Un élément <date> est utilisé ici :

```
<note>
  <date>2008-01-10</date>
  <to>Faal</to>
  <from>Jaañ</from>
</note>
```

Un élément <date> élargi est utilisé dans le troisième exemple:(C'est la favorite):

```
<note>
  <date>
    <year>2008</year>
    <month>01</month>
    <day>10</day>
  </date>
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Les attributs

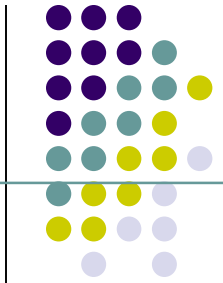
Doit on éviter les attributs XML?

Notons que :

- Un attribut ne peut contenir des valeurs multiples (les éléments oui)
- Un attribut ne peut contenir des structures d'arbres (les éléments oui)
- Un attribut n'est pas facilement extensible (pour les changements futurs)

Ne pas faire comme suit:

```
<note jour="10" mois="01" annee="2008"
a="Turé" de="Jeŋ" entete="Rappel"
corps="N'oublie pas notre rencontre!">
</note>
```



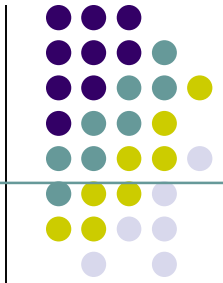
Les attributs

Les attributs pour les métadonnées

Parfois, les références d'ID sont attribués à des éléments. Ces ID peuvent être utilisés pour identifier les éléments XML de la même manière que l'attribut id en HTML. Cet exemple en est une illustration :

```
<messages>
  <note id="501">
    <to>Turé</to>
    <from>Jeŋ</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>
  <note id="502">
    <to>Jaaŋ</to>
    <from>Jeŋ</from>
    <heading>Re: Reminder</heading>
    <body>I will not</body>
  </note>
</messages>
```

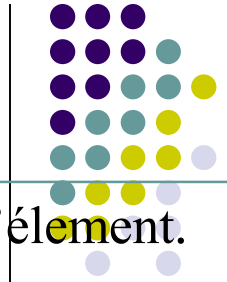
Les attributs



Les attributs pour les métadonnées

- Les attributs id ci-dessus servent à identifier des différentes notes. Ils ne font pas partie de la note elle-même.

- Ce qu'on est en train de dire ici est que les métadonnées (données sur les données) doivent être utilisées sous forme d'attributs, et les données elles même devraient être utilisées en tant qu'éléments.



Les espaces de noms

Les espaces de noms fournissent une méthode d'éviter les conflits de noms d'élément.

Conflits de nom

En XML, les noms des éléments sont définis par le développeur. Ce qui entraine souvent des conflits quand on mélange des documents XML provenant de d'applications XML différentes.

Considerons les deux tables XML suivantes :

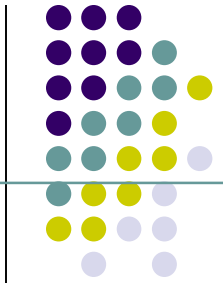
1- Informations sur un tableau HTML

```
<table>
  <tr>
    <td>Pommes</td>
    <td>Bananes</td>
  </tr>
</table>
```

2- Informations sur une table

```
<table>
  <name>table de café
        africain</name>
  <width>80</width>
  <length>120</length>
</table>
```

Ces deux fragments de code, mélangés ensemble, pourraient générer des conflits de noms. En effet, tous les deux contiennent un élément <table>, mais avec chacun un contenu et une signification differents. Un utilisateur ou une application XML ne pourra pas être en mesure de traiter ces différences.



Les espaces de noms

Résoudre les conflits de nom en utilisant un prefixe

Les conflits de nom en XML peuvent facilement être évités en utilisant un préfixe de nom.

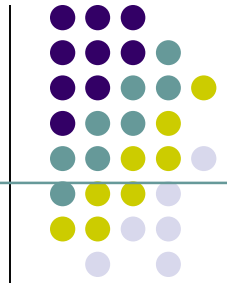
Le document suivant porte des données sur un tableau HTML, et des données sur une table de produits :

```
<h:table>
  <h:tr>
    <h:td>Pommes</h:td>
    <h:td>Bananes</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:nom>Table de Café Africain</f:nom>
  <f:largeur>80</f:largeur>
  <f:longueur>120</f:longueur>
</f:table>
```

Dans cet exemple, il n'y aura pas de conflit parce que les deux éléments <table> ont des noms différents.

Les espaces de noms



L'attribut xmlns

Lors de l'utilisation des préfixes, un espace de noms pour le préfixe doit être défini. L'espace de noms peut être défini par un attribut xmlns dans la balise de début avec la syntaxe suivante.

xmlns:*prefix*="*URI*".

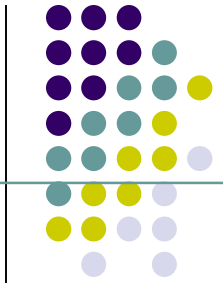
```
<racine>
<h:table xmlns:h="http://www.ugb.sn/TR/html4/">
  <h:tr>
    <h:td>Pommes</h:td>
    <h:td>Bananes</h:td>
  </h:tr>
</h:table>
<f:table xmlns:f="http://www.marchesor.com/fourniture">
  <f:nom>Table de Café Africain </f:nom>
  <f:largeur>80</f:largeur>
  <f:longueur>120</f:longueur>
</f:table>
</racine>
```

Les espaces de noms

L'attribut xmlns

Dans l'exemple ci-dessus:

- L'attribut xmlns dans le premier élément <table> fait du préfixe h: un espace de noms qualifié.
- L'attribut xmlns dans le second élément <table> fait du préfixe f: un espace de noms qualifié.
- Quand un espace de noms est défini pour un élément, tous les éléments enfant avec le même préfixe sont associés avec le même espace de noms.



Les espaces de noms

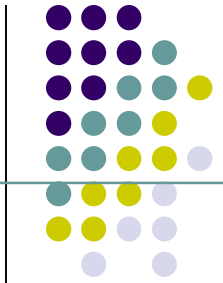
L'attribut xmlns

Les espaces de nom peuvent aussi être déclarés dans l'élément racine :

```
<racine
   xmlns:h="http://www.ugb.sn/TR/html4/"
   xmlns:f="http://www.marchesor.sn/fourniture">

<h:table>
  <h:tr>
    <h:td>Pommes</h:td>
    <h:td>Bananes</h:td>
  </h:tr>
</h:table>

<f:table>
  <f:nom>Table de Café Africain</f:nom>
  <f:largeur>80</f:largeur>
  <f:longueur>120</f:longueur>
</f:table>
</racine>
```



Les espaces de noms

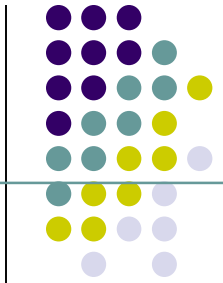
L'attribut xmlns

Remarque:

- L'URI de l'espace de noms n'est pas utilisé par l'analyseur pour rechercher des informations.
- La raison d'utiliser un URI est de donner à l'espace de noms un nom unique.
- Toutefois, les entreprises utilisent souvent l'espace de noms comme un pointeur vers une page Web contenant des informations d'espace de noms.

Uniform Resource Identifier (URI)

- Une URI (Uniform Resource Identifier) est une chaîne de caractères qui identifie une ressource Internet.
- L'URI le plus commun est l'URL (**Uniform Resource Locator**) qui identifie une adresse de domaine Internet. Un autre, pas si commun que URI, est l'URN (**Universal Resource Name**).



Les espaces de noms

Namespaces par défaut

Définir un espace de noms par défaut pour un élément nous évite d'utiliser des préfixes dans tous les éléments enfants.

La syntaxe est la suivante : xmlns = "*namespacsURI*"

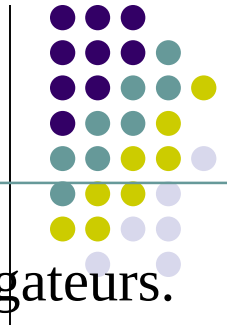
Informations sur un tableau HTML

```
<table xmlns="http://www.ugb.sn
/TR/html4/">
  <tr>
    <td>Apples</td>
    <td>Bananas</td>
  </tr>
</table>
```

Information about piece of furniture

```
<table xmlns="http://www.ugb.sn
/fourniture">
  <nom>Table de Café
Africain</nom>
  <largeur>80</largeur>
  <longueur>120</longueur>
</table>
```

Visualisation d'un document XML



Un fichier XML brute peut être consulté dans tous les principaux navigateurs. Mais ne vous attendez pas à l'afficher sous forme de pages HTML.

- La plupart des navigateurs afficheront un document XML avec les éléments en couleur.

- Souvent, un plus (+) ou un moins (-) à gauche des éléments peuvent être cliqués pour développer ou réduire la structure de l'élément.

- Pour afficher la source XML brute, on sélectionne "source de la page" ou "Afficher la source" dans le menu du navigateur.

Remarque: Dans Safari 5 (et versions antérieures), seul le texte de l'élément sera affiché. Pour afficher le XML brute, vous devez faire un clic-droit sur la page et sélectionnez "Afficher la source".

Visualisation d'un document XML

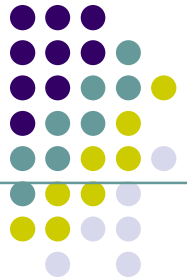
Afficher un fichier XML non valide

Si un fichier XML erroné est ouvert, certains navigateurs signalent une erreur, et d'autres l'affichent ou l'affichent de manière incorrecte.

Exemple: L'affichage de ce fichier XML dans Chrome produit l'erreur suivante.

```
<?xml version="1.0" encoding="UTF-8"?>
 - <note>
      <a>Aminata</a>
      <de>Ngeer</Fde>
      <entete>Rappel</entete>
      <corps>N'oublie pas le cours de XML !</corps>
   </note>
```

Visualisation d'un document XML



Pourquoi XML s'affichage ainsi ?

- Les documents XML ne portent pas d'informations sur la façon d'afficher les données.
- Comme les balises XML sont «inventés» par l'auteur du document XML, les navigateurs ne savent pas si une balise comme <table> décrit un tableau HTML ou une table à manger.
- Sans aucune information sur la façon d'afficher les données, le navigateur peut simplement afficher le document XML comme tel.

Formater du XML comme du HTML

- Un fichier XML peut être formaté en utilisant CSS, JavaScript ou XSLT.
- Le formatage XML avec CSS n'est pas recommandé.
- Utilisez JavaScript ou XSLT à la place.

Visualisation d'un document XML

Affichage de XML avec CSS

Voici un exemple sur comment formater un document XML avec CSS. Un fichier XML cd_catalog.xml et un fichier CSS cd_catalog.css sont utilisés.

cd_catalog.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="cd_catalog.css"?>
<CATALOG>
  <CD>
    <TITLE>Senegambia</TITLE>
    <ARTIST>Musaa Ngom</ARTIST>
    <COUNTRY>Gambia</COUNTRY>
    <COMPANY>Xippi</COMPANY>
    <PRICE>3000</PRICE>
    <YEAR>1985</YEAR>
  </CD>
  <CD>
    <TITLE>Solda</TITLE>
    <ARTIST>Umar Pen</ARTIST>
    <COUNTRY>Senegal</COUNTRY>
    <COMPANY>Jolloli</COMPANY>
    <PRICE>4000</PRICE>
    <YEAR>1983</YEAR>
  </CD>
  .
  .
  .
</CATALOG>
```

Browser result

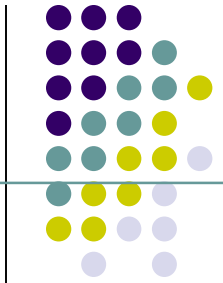
Senegambia
Musaa Ngom
Gambia
Xippi
3000
1985

Solda
Umar Pen
Senegal
Jolloli
4000
1983

cd_catalog.css

```
CATALOG {
    background-color: #ffffff;
    width: 100%;
}
CD {
    display: block;
    margin-bottom: 30pt;
    margin-left: 0;
}
TITLE {
    display: block;
    color: #ff0000;
    font-size: 20pt;
}
ARTIST {
    display: block;
    color: #0000ff;
    font-size: 20pt;
}
COUNTRY, PRICE, YEAR, COMPANY {
    display: block;
    color: #000000;
    margin-left: 20pt;
}
```

Visualisation d'un document XML



Affichage de XML avec XSLT

- Avec XSLT, on peut transformer un document XML en HTML.
- XSLT (eXtensible Stylesheet Language Transformations) est le langage de feuille de style recommandé pour XML.
- XSLT est beaucoup plus sophistiqué que CSS.
- Avec XSLT, on peut ajouter/supprimer des éléments et des attributs à partir du fichier de sortie.
- On peut également réorganiser et trier les éléments, effectuer des tests et prendre des décisions sur les éléments à masquer et à afficher, et faire beaucoup plus.
- XSLT utilise XPath pour trouver des informations dans un document XML.

Visualisation d'un document XML

Affichage de XML avec XSLT

L'exemple ci-dessous est un extrait d'un exemple d'utilisation de XSLT pour formater un document XML. Le fichiers XML breakfast_menu.xml et le fichier XSLT breakfast_menu.xsl sont utilisés.

- ***breakfast_menu.xml***

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="breakfast_menu.xsl" ?>
<breakfast_menu>
<food>
<name>Ceebu Jën</name>
<price>3000</price>
<description>Two of our famous Belgian Waffles with plenty of real
maple syrup</description>
<calories>650</calories>
</food>
...
</breakfast_menu>
```

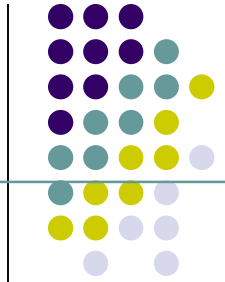
Visualisation d'un document XML

Affichage de XML avec XSLT

- ***breakfast_menu.xsl***

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" exclude-result-
    prefixes="xs" version="2.0">
    <xsl:template match="/">
      <html>
        <body>
          <xsl:for-each select="breakfast_menu/food">
             <div>
                <span><xsl:value-of select="name"/> - </span>
                <xsl:value-of select="price"/>
             </div>
          </xsl:for-each>
        </body>
      </html>
    </xsl:template>
</xsl:stylesheet>
```


On verra XSLT en details dans la suite.



Visualisation d'un document XML

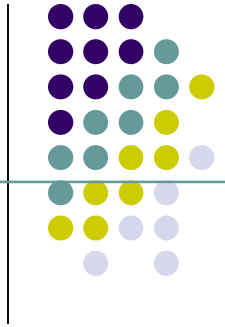
Affichage de XML avec JavaScript

L'objet XMLHttpRequest

- Tous les navigateurs modernes ont un objet XMLHttpRequest intégré pour accéder aux données à partir d'un serveur.
- L'objet XMLHttpRequest est utilisé pour échanger des données avec un serveur.

Il permet de :

- mettre à jour une page web sans la recharger
- demander des données au serveur après le chargement de la page
- Recevoir des données du serveur après le chargement de la page
- Envoyer des données au serveur en arrière plan



Visualisation d'un document XML

Affichage de XML avec JavaScript

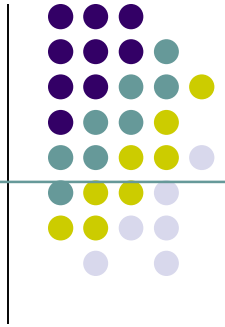
L'objet XMLHttpRequest

- **Envoi d'un objet XMLHttpRequest**

Une syntaxe JavaScript courante pour l'utilisation de l'objet XMLHttpRequest ressemble beaucoup à ceci:

Exemple

```
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      // Action typique à exécuter quand la document est prêt:
      document.getElementById("demo").innerHTML = xhttp.responseText;
    }
};
xhttp.open("GET", " filename ", true);
xhttp.send();
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

L'objet XMLHttpRequest

Explication :

- La première ligne dans l'exemple ci-dessus crée un objet **XMLHttpRequest**

```
var xhttp = new XMLHttpRequest();
```

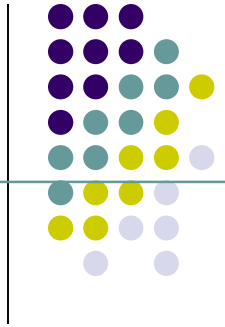
- La **propriété onreadystatechange** spécifie une fonction à exécuter chaque fois que l'état de l'objet XMLHttpRequest change:

```
xhttp.onreadystatechange = function()
```

- Lorsque la **propriété readyState** est 4 et la **propriété status** est 200, la réponse est prête: `if (this.readyState == 4 && this.status == 200)`
- La **propriété responseText** renvoie la réponse du serveur sous forme de chaîne de caractères :

La chaîne peut être utilisée pour mettre à jour une page Web par :

```
document.getElementById("demo").innerHTML = xhttp.responseText;
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

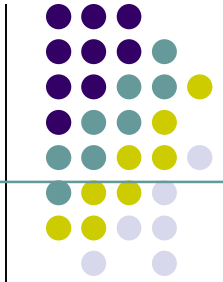
L'objet XMLHttpRequest

Pour d'anciennes versions d'IE (IE5 et IE6)

- Les anciennes versions d'Internet Explorer (IE5 et IE6) ne prennent pas en charge l'objet XMLHttpRequest.
- Pour gérer IE5 et IE6, on vérifie si le navigateur prend en charge l'objet XMLHttpRequest ou bien on crée un objet ActiveXObject:

Exemple

```
if (window.XMLHttpRequest) {
    // code pour navigateurs modernes
    xmlhttp = new XMLHttpRequest();
 } else {
    // code pour les anciens navigateurs de IE
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
}
```



Visualisation d’un document XML

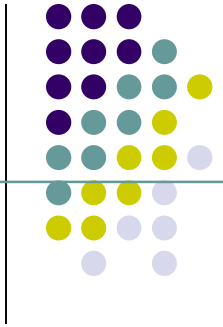
Affichage de XML avec JavaScript

Le parseur XML

- Tous les principaux navigateurs ont un analyseur XML intégré pour convertir du texte en un objet DOM XML, y accéder et le manipuler.

- Le DOM XML (Document Object Model) définit les propriétés et les méthodes d'accès et d'édition d’un document XML.

- Toutefois, avant d'accéder à un document XML, il doit être chargé dans un objet DOM XML.



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le parseur XML

Parser (Analyser) une chaîne de caractères.

L'exemple suivant analyse une chaîne de caractères dans un objet DOM XML et extrait les informations avec JavaScript:

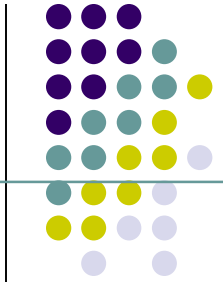
Exemple

```
<p id="demo"></p>
<script>
var text, parser, xmlDoc;

text = "<bookstore><book>" +
"<title>JavScript de base</title>" +
"<author>Moussa Lo</author>" +
"<year>2005</year>" +
"</book></bookstore>";
```

```
parser = new DOMParser();
xmlDoc =
parser.parseFromString(text,"text/xml");

document.getElementById("demo").innerHTML
 =
xmlDoc.getElementsByTagName("title")[0].child
Nodes[0].nodeValue;
</script>
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le parseur XML

Parser (Analyser) une chaîne de caractères.

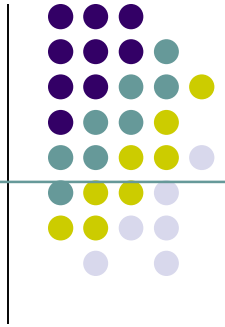
Explication :

- Une chaîne de caractères est définie:

```
text = "<bookstore><book>" +
"<title>JavScript de base</title>" +
"<author>Moussa Lo</author>" +
"<year>2005</year>" +
"</book></bookstore>";
```

- Un analyseur DOM XML est créé par :
  parser = new DOMParser();
- Le parseur crée un nouvel objet DOM XML à l'aide de la chaîne de caractères :

  xmlDoc = parser.parseFromString(text,"text/xml");



Visualisation d'un document XML

Affichage de XML avec JavaScript

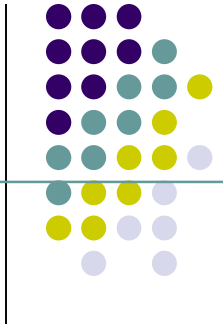
Le parseur XML

Anciennes versions d'Internet Explorer

- Les anciennes versions d'Internet Explorer (IE5, IE6, IE7, IE8) ne prennent pas en charge l'objet DOMParser.
- Pour les prendre en compte, on vérifie si le navigateur prend en charge l'objet DOMParser ou on crée un objet ActiveXObject:

Exemple

```
if (window.DOMParser) {  // code pour navigateurs modernes
        parser = new DOMParser();
        xmlDoc = parser.parseFromString(text,"text/xml");
}
else {  // code pour anciens navigateurs de IE
        xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
        xmlDoc.async = false;
        xmlDoc.loadXML(text);
}
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le parseur XML

Parseur intégré de l'objet XMLHttpRequest

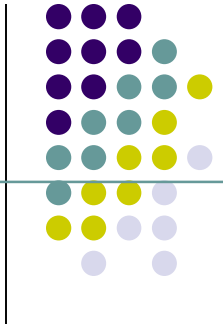
L' objet XMLHttpRequest possède un analyseur XML intégré.

- La **propriété responseText** renvoie la réponse sous forme de chaîne.
- La **propriété responseXML** renvoie la réponse en tant qu'objet DOM XML.

Si on souhaite utiliser la réponse en tant qu'objet DOM XML, on peut utiliser la propriété responseXML.

Exemple

```
xmlDoc = xmlhttp.responseXML;
txt = "";
x = xmlDoc.getElementsByTagName("ARTIST");
for (i = 0; i < x.length; i++) {
    txt += x[i].childNodes[0].nodeValue + "<br>";
}
document.getElementById("demo").innerHTML = txt;
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le DOM XML (Document Object Model)

Le DOM définit une norme pour accéder à des documents comme XML et HTML:

"Le modèle Objet de document (DOM) du W3C est une plate-forme et une interface indépendante du langage qui permet aux programmes et aux scripts d'accéder et de mettre à jour dynamiquement le contenu, la structure et le style d'un document."

Le DOM est divisé en 3 parties / niveaux :

- Core DOM - modèle standard pour tout document structuré
- XML DOM - modèle standard pour les documents XML
- HTML DOM - modèle standard pour les documents HTML

Le DOM définit les **objets** et **propriétés** de tous les éléments du document, ainsi que les méthodes (**interface**) pour y accéder.

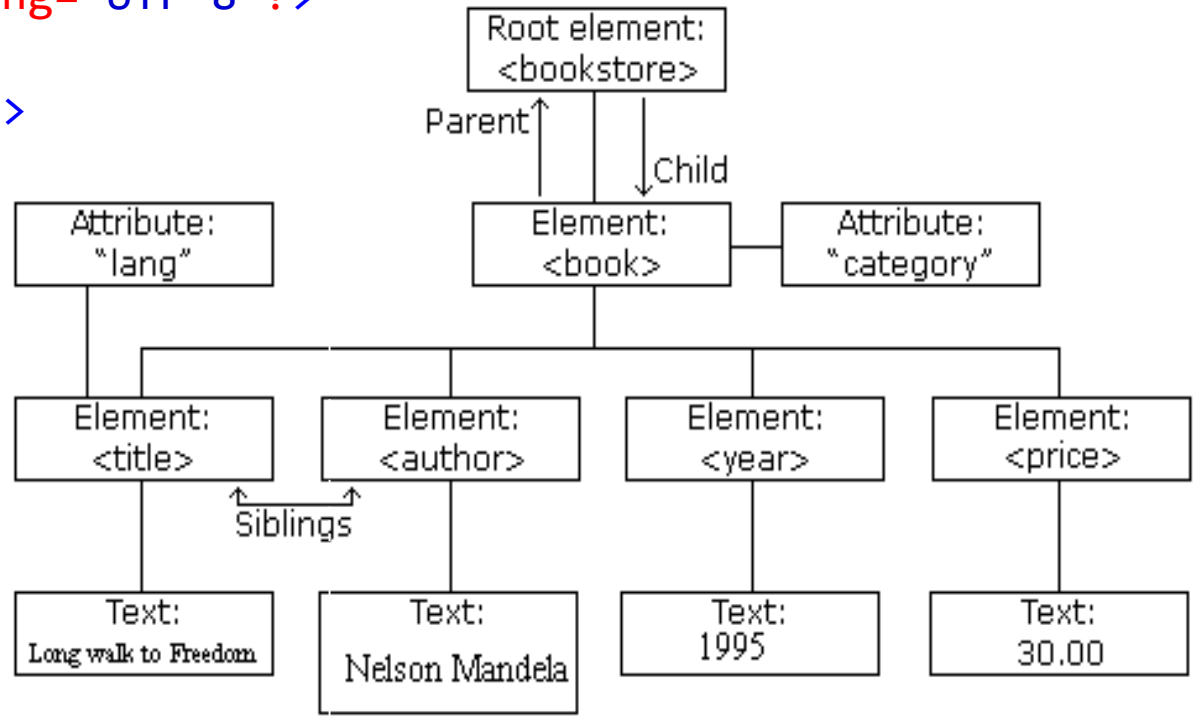
Visualisation d'un document XML

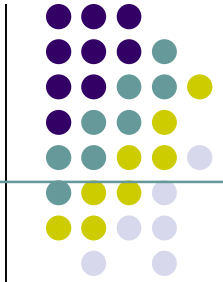
Affichage de XML avec JavaScript

Le DOM XML

- Le DOM XML présente un document XML comme arborescence.
- Le DOM HTML présente un document HTML comme arborescence.
- Comprendre le DOM est un devoir pour quiconque travaille avec HTML ou XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="History">
    <title lang="en">
      Long walk to freedom
    </title>
    <author>
      Nelson Mandela
    </author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  ...
</bookstore>
```





Visualisation d’un document XML

Affichage de XML avec JavaScript

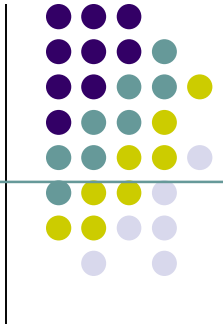
Le DOM XML

Un DOM définit une méthode standard pour accéder et manipuler des documents.

• Le DOM XML définit une méthode standard pour accéder et manipuler des documents XML.

• Le DOM XML considère un document XML comme une arborescence.

• Tous les éléments sont accessibles via l'arborescence DOM. Leur contenu (texte et les attributs) peut être modifié ou supprimé, et de nouveaux éléments peuvent être créés. Les éléments, leur texte et leurs attributs sont tous appelés nœuds.



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le DOM XML

Le DOM HTML

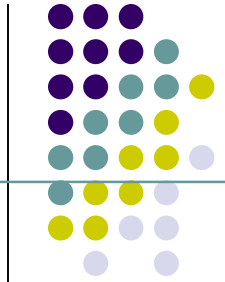
- Le DOM HTML définit une méthode standard pour accéder et manipuler des documents HTML.
- Tous les éléments HTML sont accessibles via le DOM HTML.
  Cet exemple modifie la valeur d'un élément HTML avec id = "demo":

Exemple :

Tous les éléments HTML peuvent être accessibles via le DOM HTML.

```
<h1 id="demo">Ceci est une entête</h1>

<button type="button"
onclick="document.getElementById('demo').innerHTML = 'Salut tout le monde !'">Click moi!
</button>
```



Visualisation d'un document XML

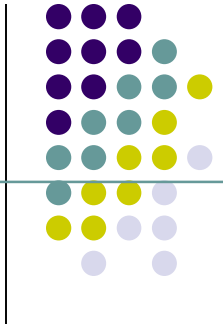
Affichage de XML avec JavaScript

Le DOM XML

Tous les éléments XML sont accessibles via le DOM XML.

Document XML

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="History">
    <title lang="en">Long walk to freedom </title>
    <author>
      Nelson Mandela
    </author>
    <year>2005</year>
     <price>30.00</price>
    </book>
    ...
</bookstore>
```



Visualisation d'un document XML

Affichage de XML avec JavaScript

Le DOM XML

txt = xmlDoc.getElementsByTagName("title")[0].childNodes[0].
nodeValue;

Le DOM XML est un standard pour savoir comment obtenir, modifier, ajouter et supprimer des éléments XML.

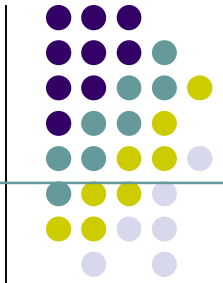
Cet exemple charge une chaîne de caractères dans un objet DOM XML et extrait les informations de celui-ci avec JavaScript:

Exemple :

```
<p id="demo"></p>
<script>
var text, parser, xmlDoc;
text = "<bookstore><book>" +
"<title>JavScript de base</title>" +
"<author>Moussa Lo</author>" +
"<year>2005</year>" +
"</book></bookstore>";
```

```
parser = new DOMParser();
xmlDoc =
parser.parseFromString(text,"text/xml");

document.getElementById("demo").innerHTML =
xmlDoc.getElementsByTagName("title")[0].chil
dNodes[0].nodeValue ;
</script>
```



Definition de type de document XML

Document XML bien formé (Rappel)

Un document XML ayant un syntaxe correct est dit "bien formé". C'est à dire:

- Un document XML doit avoir un élément root
- Un élément XML doit avoir une balise de fermeture
- Une balise XML est sensible à la casse
- Les éléments doivent être correctement imbriqués
- Un attribut XML doit être cité

XML ne tolère aucune erreur.

- Un navigateur HTML permet l'affichage de document HTML avec des erreurs (comme l'absence de balise de fermeture).
- Une erreur dans un document XML arrête l'application.
- La spécification XML du W3C indique que le programme doit cesser de traiter un document XML s'il constate une erreur. La raison est que le logiciel XML doit être petit, rapide et compatible.

Definition de type de document XML

Document XML valide

- Un document XML "**bien formé**" n'est pas le même qu'un document XML "**valide**".

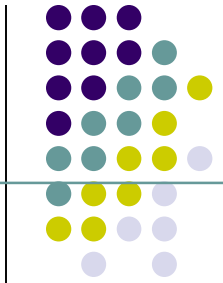
- Un document XML "**valide**" doit être bien formé. De plus, il doit être conforme à une définition de type de document.

- Il existe deux définitions différentes de type de document qui peuvent être utilisés avec XML:

  - **DTD -** Le type de définition de document d'origine

  - **XML Schema -** Une alternative au DTD basée sur XML

- Une définition de type de document définit les règles et les éléments et attributs légales d'un document XML.



Definition de type de document XML

Quand utiliser un DTD/Schema?

- Avec un DTD/Schema, des groupes indépendants de personnes peuvent convenir d'utiliser un DTD/Schema standard pour l'échange de données.
- Avec un DTD/Schema, on peut vérifier que les données reçues sont valides.
- On peut également utiliser un DTD/Schema pour vérifier ses propres données.

Quand ne pas utiliser un DTD/Schema?

- XML ne nécessite pas de DTD/Schema.
- Lorsqu’on fait des testes sur XML, ou lorsqu’on travaille avec de petits fichiers XML, la création d’un DTD/Schema peut être une perte de temps.
- Si on développe des applications, on attend que la spécification soit stable pour ajouter la définition du document. Sinon, le logiciel peut cesser de fonctionner en raison d'erreurs de validation.

Definition de type de document XML

Définir le type de document XML avec DTD

Le document XML ci-dessous est valide car il est bien formé et contient une définition de type de document à travers la déclaration DOCTYPE qui est une référence à un fichier DTD externe. Le contenu du fichier DTD est indiqué dans la suite.

- **Le fichier XML**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
<a>Jàmme</a>
<de>Saar</de>
<entete>Reminder</entete>
<corps>N'oublie pas l'exercice!</corps>
</note>
```

Definition de type de document XML

Définir le type de document XML avec DTD

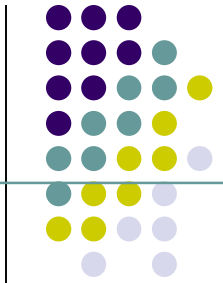
- **Le fichier Note.dtd**

```
<!DOCTYPE note [
   <!ELEMENT note (a,de,entete,corps)>
   <!ELEMENT a (#PCDATA)>
   <!ELEMENT de (#PCDATA)>
   <!ELEMENT entete (#PCDATA)>
   <!ELEMENT corps (#PCDATA)>
]>
```

- **Interprétation:**

- `!DOCTYPE note` définit que l'élément racine du document note
- `!ELEMENT note` définit que l'élément note doit contenir les éléments: a, de, entete et corps
- `!ELEMENT a` définit l'élément **a** de type "#PCDATA"
- `!ELEMENT de` définit l'élément **de** de type "#PCDATA"
- `!ELEMENT entete` définit l'élément entete de type "#PCDATA"
- `!ELEMENT corps` définit l'élément corps de type "#PCDATA"

#PCDATA signifie parse-able character data (données textuelles analysables).



Definition de type de document XML

Définir le type de document XML avec DTD

Déclaration d'entité: La déclaration DOCTYPE peut également être utilisé pour définir des caractères spéciaux et des chaînes de caractères utilisées dans le document:

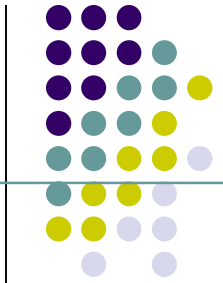
Exemple

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE note [
   <!ENTITY nbsp " ">
   <!ENTITY auteur "Auteur: Umar Jóob.">
   <!ENTITY copyright "Copyright: UGB.SN.">
]>

<note>
   <a>David</a>
   <de>Jéen</de>
   <entete>Rappel</entete>
   <corps>N'oublie pas l'AG!</corps>
   <pied>&auteur; &copyright;</pied>
</note>
```


Une entité a trois parties: une esperluette (&), un nom d'entité et un point-virgule (;).



Definition de type de document XML

Définir le type de document XML avec XML SCHEMA

- Un XML Schema (schéma XML) décrit la structure d'un document XML, tout comme un DTD. Il est une alternative au DTD basée sur XML.
- Un document XML avec une syntaxe correcte est dit "bien formé".
- Un document XML validé par rapport à un XML Schema est à la fois "bien formé" et "valide".

Exemple:

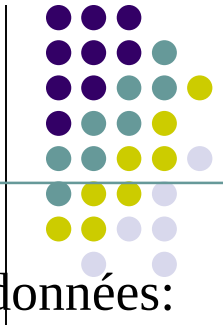
```
<xs:element name="note">
<xs:complexType>
  <xs:sequence>
    <xs:element name="a" type="xs:string"/>
    <xs:element name="de" type="xs:string"/>
    <xs:element name="entete" type="xs:string"/>
    <xs:element name="corps" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
```

Definition de type de document XML

Définir le type de document XML avec XML SCHEMA

Le schéma ci-dessus est interprété comme suit :

- `<xs:element` name="note"`>` definit l'élément appelé "note"
- `<xs:complexType>` définit que l'élément "note" est un type compléxe
- `<xs:sequence>` définit que le type compléxe est une sequence d'éléments
- `<xs:element` name="a" type="xs:string"`>` l'élément "a" est de type string (text)
- `<xs:element` name="de" type="xs:string"`>` l'élément "de" est de type string
- `<xs:element` name="entete" type="xs:string"`>` l'élément "entete" est de type string
- `<xs:element` name="corps" type="xs:string"`>` l'élément "corps" est de type string



Definition de type de document XML

XML Schema supporte les types de données

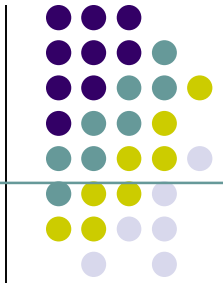
Une des plus grandes forces de XML Schema est qu’il supporte les types de données:

- Il est plus facile de décrire le contenu du document
- Il est facile de définir des restrictions sur les données
- Il est plus facile de valider l'exactitude des données
- Il est plus facile de convertir des données entre différents types de données

XML Schema utilise la syntaxe XML

Une autre grande force de XML Schema est qu'il est écrit en XML:

- On n'est pas obligé d'apprendre un nouveau langage
- On peut utiliser son éditeur XML pour modifier ses fichiers de schéma
- On peut utiliser son analyseur XML pour analyser ses fichiers de schéma
- On peut manipuler son schémas avec le DOM XML
- On peut transformer son schéma avec XSLT



Definition de type de document XML

DTD vs XML SCHEMA

XML Schema est plus puissant que DTD

- XML Schema est écrit en XML
- XML Schema est extensible à des additions
- XML Schema supporte les types de données
- XML Schemas supporte les namespaces

Pourquoi utiliser XML Schema ?

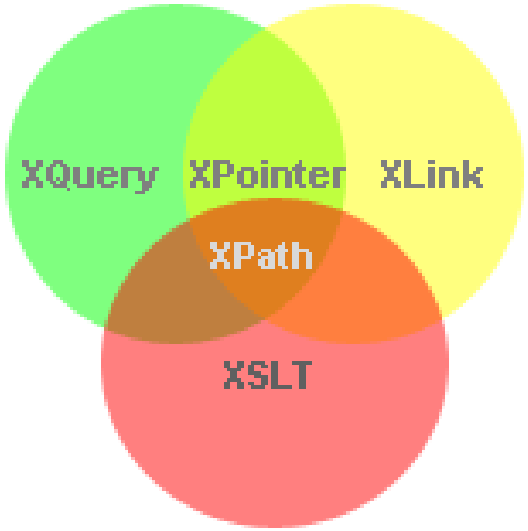
- Avec XML Schema, un fichier XML peut transporter une description de son propre format.
- Avec XML Schema, des groupes indépendants de personnes peuvent se mettre d'accord sur une norme d'échange de données.
- Avec XML Schema, on peut vérifier des données.

Technologies XML

Le langage XPATH

XPath est un langage permettant de trouver des informations dans un document XML.

C'est quoi XPath?



- XPath est un syntaxe pour définir les parties d'un document XML
- XPath utilise des expressions de chemin pour naviguer dans un document XML
- XPath contient une bibliothèque de fonctions standards
- XPath est un élément majeur dans XSLT
- XPath est également utilisé dans XQuery, XPointer et XLink
- XPath est une recommandation du W3C

Technologies XML

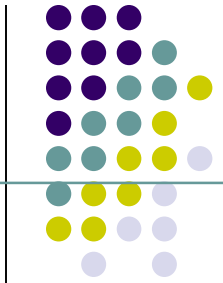
Le langage XPATH

Expressions de chemin XPath

- XPath utilise des expressions de chemin pour sélectionner des nœuds ou ensembles de nœuds dans un document XML. Ces expressions de chemin ressemblent beaucoup aux expressions utilisées lorsqu'on travaille avec un système de fichiers traditionnel de l'ordinateur.
- Aujourd'hui les expressions XPath peuvent également être utilisés en JavaScript, Java, XML Schema, PHP, Python, C et C ++, etc..

XPath est utilisé dans XSLT

- XPath est un élément majeur dans la norme XSLT. Sans une connaissance de XPath vous ne serez pas en mesure de créer des documents XSLT.



Technologies XML

Le langage XPATH

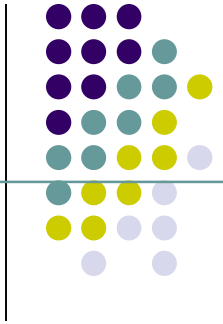
Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>

<librairie>

<livre categorie="Histoire">
  <titre langue="en">Long walk to freedom</titre>
  <auteur>Nelson Mandela</auteur>
  <annee>1993</annee>
  <prix>3000</prix>
</livre>
...
</librairie>
```

/librairie/livre[1] sélectionne le premier élément **livre** qui est l'enfant de l'élément **librairie**



Technologies XML

Le langage XPATH

Expression XPath	Résultat
/librairie/livre[1]	Sélectionne le premier élément livre enfant de l'élt librairie
/librairie/livre[last()]	Sélectionne le dernier élément livre enfant de l'élt librairie
/librairie/livre[last()-1]	Sélectionne l'avant dernier élément livre enfant de l'élément librairie
/librairie/livre[position()<3]	Sélectionne les deux premiers éléments livre qui sont enfants de l'élément librairie
//titre[@langue]	Sélectionne tous les éléments titre ayant un attribut lang
//titre[@langue='en']	Sélectionne tous les éléments titre ayant un attribut "langue" avec une valeur égale à "en"
/librairie/livre[prix>3500]	Sélectionne tous les éléments livre de l'élément librairie ayant un élément prix avec une valeur supérieure à 3500
/librairie/livre[prix>3500]/titre	Sélectionne tous les éléments titre de l'élt livre de l'élément librairie qui ont un élt de prix d'une valeur supérieure à 3500

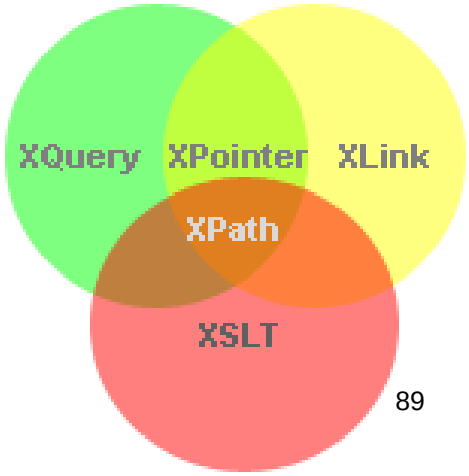
Technologies XML

XQuery

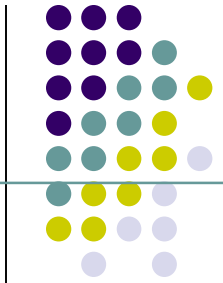
- XQuery est à XML ce que SQL est aux bases de données relationnelles.
- XQuery est conçu pour interroger des données XML - pas seulement des fichiers XML, mais tout ce qui peut apparaître comme XML, y compris les bases de données.
- XQuery est construit sur des expressions XPath
- XQuery est pris en charge par toutes les principales bases de données
- XQuery est une recommandation du W3C depuis le 23 Janvier 2007.

Exemple:

```
for $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title
```



Technologies XML



XQuery

- XQuery 1.0 et XPath 2.0 partagent le même modèle de données et prennent en charge les mêmes fonctions et les opérateurs. Si vous avez déjà étudié XPath vous aurez aucun problème pour comprendre XQuery.

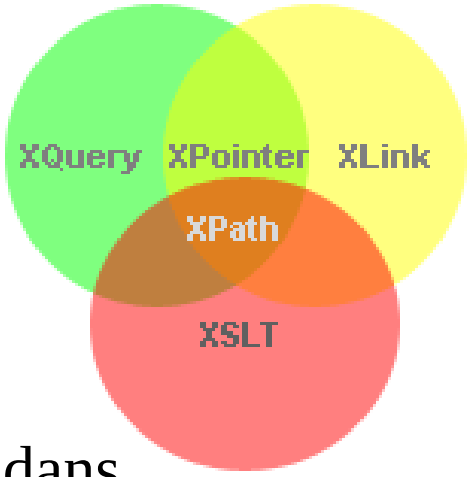
XQuery peut être utilisé pour:

- Extraire des informations à utiliser dans un service Web
- Générer des rapports de synthèse
- Transformer des données XML en XHTML
- Chercher des documents pertinentes sur le Web

Technologies XML

Xlink

- XLink est utilisé pour créer des hyperliens dans les documents XML
- Tout élément XML peut se comporter comme un lien
- Avec XLink, les liens peuvent être définis en dehors des fichiers liés
- XLink est une recommandation du W3C



Prise en charge de Xlink par les navigateurs

- Xlink n'est pas pris en charge par les navigateurs dans un document XML.
- Cependant, tous les principaux navigateurs supportent XLink dans SVG.

Technologies XML

Syntaxe XLink

En HTML, l'élément <a> définit un lien hypertexte. Cependant, cela ne fonctionne pas ainsi en XML.

Dans les documents XML, les noms des éléments sont définis par l'utilisateur – c'est pourquoi il est impossible pour les navigateurs de prédire les éléments liens qui seront appelés dans le document XML.

Exemple: Comment utiliser XLink pour créer des liens dans un document XML:

```
<?xml version="1.0" encoding="UTF-8"?>

<pagesweb xmlns:xlink="http://www.w3.org/1999/xlink">
  <pageweb xlink:type="simple" xlink:href="http://www.ugb.sn/sat">Visi
tez l'UFR SAT</pageweb>
  <pageweb xlink:type="simple" xlink:href="http://www.ugb.sn/seg">Visi
tez l'UFR SEG</pageweb>

</pagesweb>
```

Technologies XML

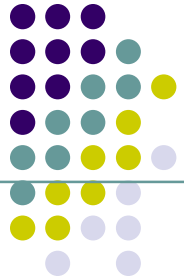
Syntaxe XLink

- Pour avoir accès aux fonctionnalités XLink, on doit déclarer l'espace de noms Xlink : "http://www.w3.org/1999/xlink".

- Les attributs xlink:type et xlink: href dans les éléments <pageweb> proviennent de l'espace de noms XLink.

- xlink:type = "simple" crée un lien simple comme celui de HTML (qui signifie "cliquez ici pour aller la-bàs").

- L'attribut xlink: href spécifie l'URL à lier.



Technologies XML

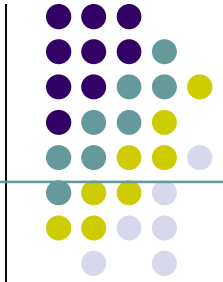
Exemple XLink

Le document XML suivant contient des caracteristiques de XLink :

```
<?xml version="1.0" encoding="UTF-8"?>
<librairie xmlns:xlink="http://www.w3.org/1999/xlink">

<livre titre="XML et XLink">
  <description
  xlink:type="simple"
  xlink:href="/images/hxml.gif"
  xlink:show="new">
  Un livre pour pratiquer XML .......
  </description>
</livre>

<livre  titre="JavaScript">
  <description
  xlink:type="simple"
  xlink:href="/images/js.gif"
  xlink:show="new">
  Notions de base de JavaScript.......
  </description>
</livre>
</librairie>
```



Technologies XML

Référence d'attributs XLink

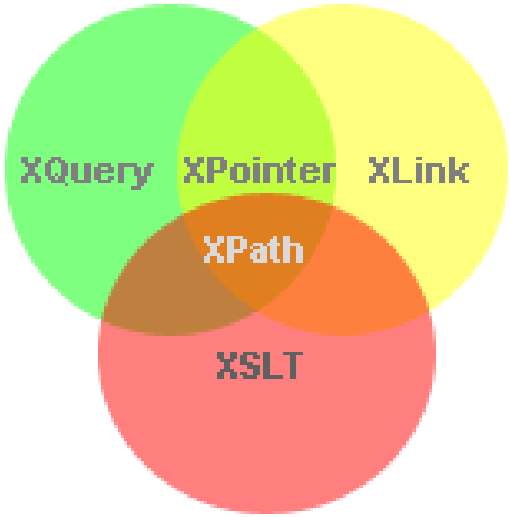
Attribute	Value	Description
xlink:actuate	onLoad onRequest other none	Définit quand la ressource liée est lue et affichée: onload - la ressource doit être chargée et affichée au chargement du document; onRequest - la ressource n'est pas lue ou affichée avant que le lien soit cliqué
xlink:href	*URL*	Spécifie l'URL de la resource à lier
xlink:show	embed new replace other none	Spécifie où ouvrir le lien. La valeur par défaut est "replace"
xlink:type	simple extended locator arc resource title none	Spécifie le type de lien

Voir exemple pratique: http://www.xml.com/pub/a/2000/09/xlink

Technologies XML

Xpointer

- XPointer permet aux liens de pointer vers des parties spécifiques d'un doc. XML
- XPointer utilise des expressions XPath pour naviguer dans le document XML
- XPointer est une recommandation du W3C



Prise en charge de Xpointer par les navigateurs

- Il n'y a pas de navigateur prenant en charge XPointer. Mais XPointer est utilisé dans d'autres langages XML

Technologies XML

Exemple XPointer

- Dans cet exemple, on utilise XPointer en conjonction avec XLink pour pointer vers une partie spécifique d'un autre document.
- Commençons par regarder le document XML cible :

```
<?xml version="1.0" encoding="UTF-8"?>
<racedemoutons>

<mouton breed="Laadum" id="Laadum">
  <image url="http://mouton.com/laadum.gif" />
  <histoire>L'ancêtre du Laadum était probablement de la Mauritanie</histoire>
  <temperament>Belle, grasse, mince et grand, le Laadum est un choix populaire
pour sa taille...</temperament>
</mouton>

<mouton breed="Bali-bali" id="Bali-bali">
  <image url="http://mouton.com/bali-bali.gif" />
  <histoire>L'ancêtre du Bali-Bali était probablement du Tchad..</histoire>
  <temperament>Le Bali-Bali ou Azawak ressemble plutôt à un taureau avec le
développement de l'arrière, de la cuisse, du bassin, des organes
internes...</temperament>
</mouton>
</racedemoutons>
```

Technologies XML

Exemple Xpointer

- Notez que le document XML ci-dessus utilise un attribut id sur chaque élément!
- Ainsi, au lieu de se lier à l'ensemble du document (comme avec XLink), XPointer permet d'accéder à des parties spécifiques du document.
- Pour créer un lien vers une partie spécifique d'une page, ajoutez le signe dièse (#) et une expression XPointer après l'URL dans l'attribut xlink: href, comme ceci:

```
xlink:href="http://mouton.com/racedemoutons.xml#xpointer(id('Laadum'))".
```

L'expression fait référence à l'élément dans le document cible, avec la valeur "Laadum" de l'id .

- XPointer permet également la méthode sténographique pour lier un élément à un id. On peut utiliser la valeur de l'identifiant directement, comme ceci:

```
xlink:href="http://mouton.com/racedemoutons.xml#Laadum"
```

Technologies XML

Exemple XPointer

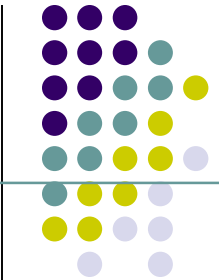
- Le document XML suivant contient des liens vers plus d'informations sur chaque mouton :

```
<?xml version="1.0" encoding="UTF-8"?>
<mesmoutons xmlns:xlink="http://www.w3.org/1999/xlink">

<monmouton>
  <description>
  Mbeng Tyson est mon mouton préférée. Il a gagné beaucoup de médialles.....
  </description>
  <fait xlink:type="simple" xlink:href="http://mouton.com/racedemoutons.xml#Laadum">
  Fait à propos de Mbeng Tyson
  </fait>
</monmouton>

<monmouton>
  <description>
  Ñay est la plus belle des moutons sur terre......
  </description>
  <fait xlink:type="simple" xlink:href="http://mouton.com/racedemoutons.xml#Bali-bali">
  Fait à propos de Ñay
  </fait>
</monmouton>

</mesmoutons>
```



XML dans le Server

- Les fichiers XML sont des fichiers textes tout comme les fichiers HTML.
- XML peut être facilement stocké et généré par un serveur web standard.
- Les fichiers XML peuvent être stockés sur un serveur Internet, exactement de la même façon que les fichiers HTML.

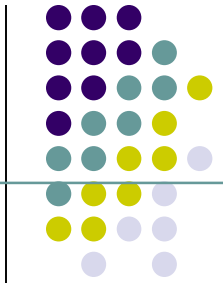
Générer du XML avec PHP

XML peut être généré sur un serveur sans aucun logiciel XML installé.
Le code PHP ci-dessous génère une réponse XML à partir du serveur:

```
<?php
header("Content-type: text/xml");
echo "<?xml version='1.0' encoding='UTF-8'?>";
echo "<note>";
echo "<de>Mame</de>";
echo "<a>Sangue</a>";
echo "<message>Rappelle toi de notre rencontre</message>";
echo "</note>";
?>
```


A noter que le type du contenu de l'en-tête de réponse doit être "text/xml".

FIN



FIN