Techniques avancées de gestion des données

Bases de données avancées et XML

Samuel Hiard

Année académique 2022-2023

Contact, ressources

- Email :
  - samuel.hiard@hepl.be

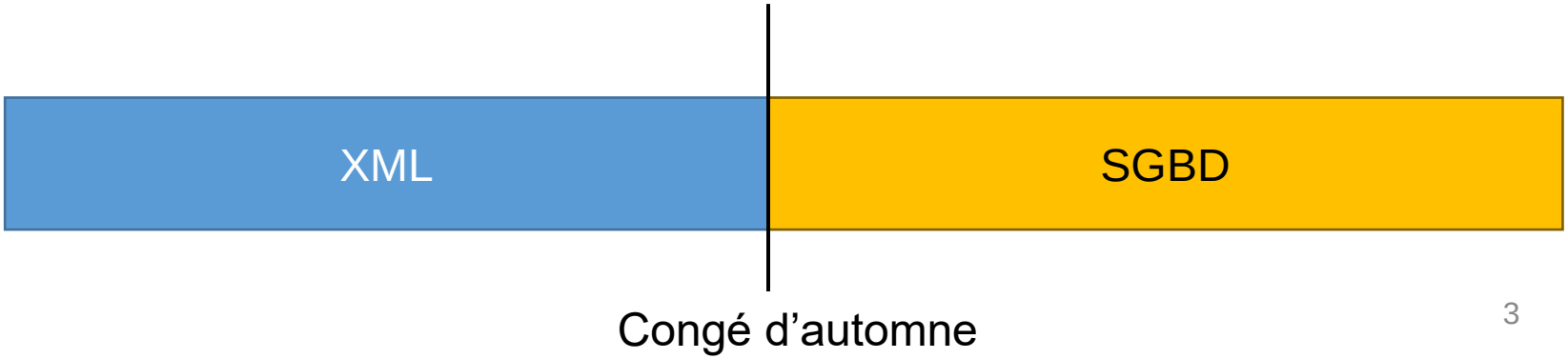
- Page Web du cours
  - Tout sur Teams, Equipe « SGBD 3ème Gestion 2022-2023 »

- Ma biblio :
  - Cours des années précédentes par Ludovic Kuty et Laurence Herbiet
  - Mon cours d'introduction aux bases de données (ULiege)
    - https://services.montefiore.uliege.be//verif/cours/bd/exercices.html
  - Ch. 18 et Annexe du cours Java III de Claude Vilvens
  - Cours XML d'Olivier Carton (Université Paris Diderot)
    https://www.irif.fr/~carton/Enseignement/XML/
  - W3Schools
    - https://www.w3schools.com/xml/

Organisation du cours

- UE Techniques avancées de gestion des données
  - **Administration de BD et XML (60h)** — **Samuel Hiard**
  - Gestion de projets (45h) — Souad Serrhini
  - Développement de projet (15h) — Cédric Thiernesse
  - Introduction aux mainframes (15h) — (?)

- Dans ma partie (60h)
  - Théorie (30h)
  - Pratique (30h)



Organisation du cours (2)

Semaine	Contenu
20/09	Plan de cours, Intro à XML
27/09	*Pas cours (Fête de la FWB)*
04/10	DTD et Namespaces
11/10	XML Schema Definition
18/10	SAX et DOM
25/10	XPath et XSLT
01/11	*Congé d'automne*
08/11	*Pas cours (Cédric Thiernesse)*
16/11	Répertoires, tables externes et blobs
23/11	BDD Distribuées et blockchain
30/11	REST, ORDS, APEX
07/12	Data warehouse et Jobs
14/12	BD Exotiques (Orienté-objet, NoSQL, MongoDB)
21/12	Questions/réponses

*sujet à modifications éventuelles

Organisation du cours (3)

Evaluation

- Pondération
  - Th : 50%, Pr : 50%
- Théorie
  - QCM pendant la période d'examens
- Pratique
  - 2 laboratoires : Un sur XML, l'autre sur SGBD
  - Labo XML : 50%
  - Labo SGBD : 50%
  - Plus d'infos en fin de leçon

Encore un mot sur l'évaluation

- Cours (AA) intégré dans une UE.
- On réussit l'UE = on réussit toutes les AA
- Moyenne **géométrique** pondérée
- Note UE = $(BDD)^{0,45}$*$(GestionProj)^{0,33}$*$(DevProj)^{0,11}$*$(Mainframes)^{0,11}$
- Exemple : 12/20, 10/20, 6/20, 7/20 → 9,86/20 → 10/20
- Autre Exemple : 16/20, 14/20, 5/20, 1/20 → 9,93/20 → 10/20
- Attention au 0/20 !

Autres détails pratiques

- Il est possible d'arriver en retard (bien que peu recommandé)
  - Entrez simplement
  - Asseyez-vous
  - Perturbez le cours le moins possible (si pendant exposé magistral)
- On peut aller aux toilettes/partir plus tôt
  - Juste, demandez-moi avant, que je sache ce qu'il se passe
- La présence aux laboratoires n'est pas obligatoire
  - Mais prévenez-moi

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

XML : Définition

- e**X**tensible **M**arkup **L**anguage

- Extensible ?
- Markup Language ?

Markup Language ?

- Langage balisé (constitué de tags) : `<nom_du_tag>`

- Quelques exemples de langages balisés :

MathML

```
<mrow> <mrow> <mrow> <mo>(</mo>
<mrow> <mrow> <mn>2</mn>
<mo>&#8290;</mo> <mi>k</mi>
</mrow> <mo>-</mo> <mn>1</mn>
</mrow> <mo>)</mo>
```

HTML

```
<HTML> <HEAD> <TITLE> Hello
world!</TITLE></HEAD>
<BODY><DIV id=class><p>Alright
then, hello world !
</p></div></BODY></HTML>
```

SVG

```
<path d="M136.128 28.837C129.728
29.637 104.999 5.605 119.328
37.637C136.128 75.193 60.394
84.037 C69.728 104.037 35.328
87.237 35.328 87.237C0.928
199.329z"/>
```

XML

```
<note>
   <de>Moi-même</de>
   <a>Mon voisin</a>
   <sujet>Coucou</sujet>
</note>
```

Extensible?

- Le vocabulaire (l'ensemble des balises autorisées) n'est pas figé.
- Les noms des balises XML sont libres. Il appartient aux auteurs de documents de fixer les balises utilisées. Il est seulement nécessaire que les auteurs s'entendent sur le vocabulaire, c'est-à-dire la liste des balises utilisées, lorsque des documents sont échangés.
- Cette liberté dans les noms de balises permet de définir des vocabulaires particuliers adaptés aux différentes applications. Il existe ainsi des vocabulaires pour décrire des dessins vectoriels, des échanges commerciaux ou des programmes de télévision.
- Ces vocabulaires particuliers sont appelés dialectes XML. Il en existe des centaines voire des milliers pour couvrir tous les champs d'application de XML.

Source : Cours XML d'Olivier Carton

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

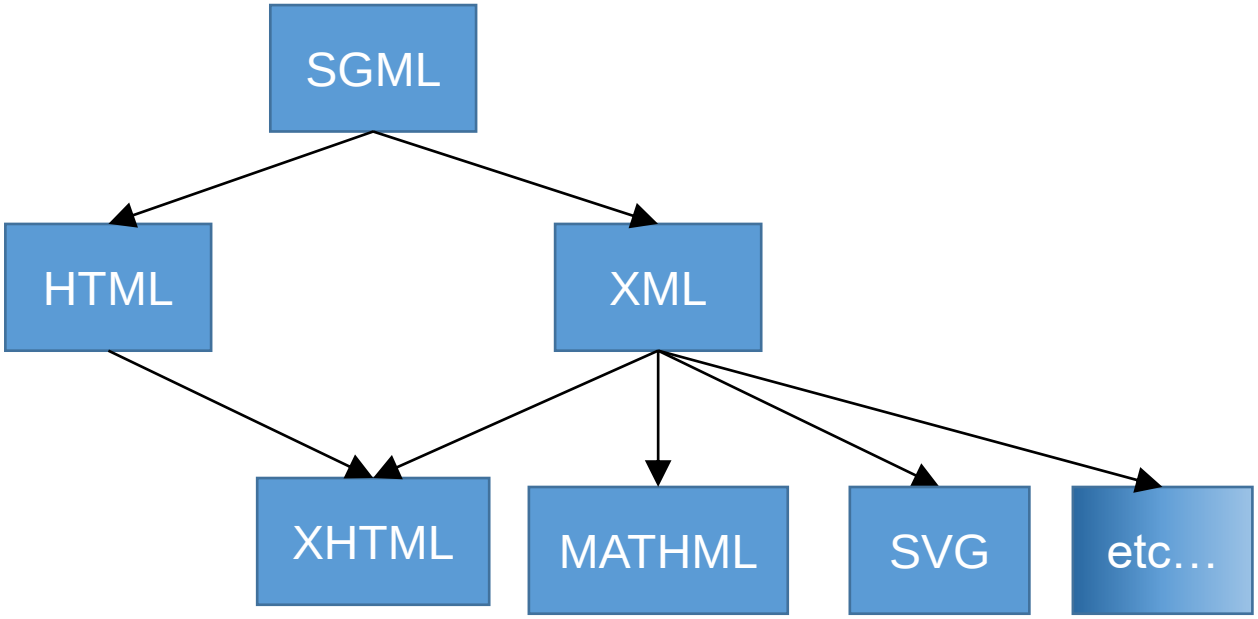
Différence entre HTML et XML

- HTML est un tel langage d'annotation qui prévoit un ensemble prédéfini d'annotations possibles.
  - Les programmes de lecture (browsers) peuvent donc les interpréter.
  - Ces programmes ont tendance à être laxistes sur la structure (remplacement implicite d'une fin de tag oubliée)

- SGML (Standard Generalized Markup Language) et XML qui en est une révision simplifiée permet d'utiliser un ensemble définissable d'annotations.
- SGML est assez ancien (standardisé en 1986), XML est plus récent (1998) et a en particulier été conçu pour être utilisé dans le cadre du Web (après HTML qui date de 1992).
- SGML et XML ne « font » rien. Ils doivent être interprétés par un programme (ou langage) spécifique.

- En 2000 apparait XHTML. On peut voir XHTML comme XML utilisé avec un ensemble particulier d'annotations
  - Version « stricte » d'HTML.

L’arbre (partiel) des langages ML

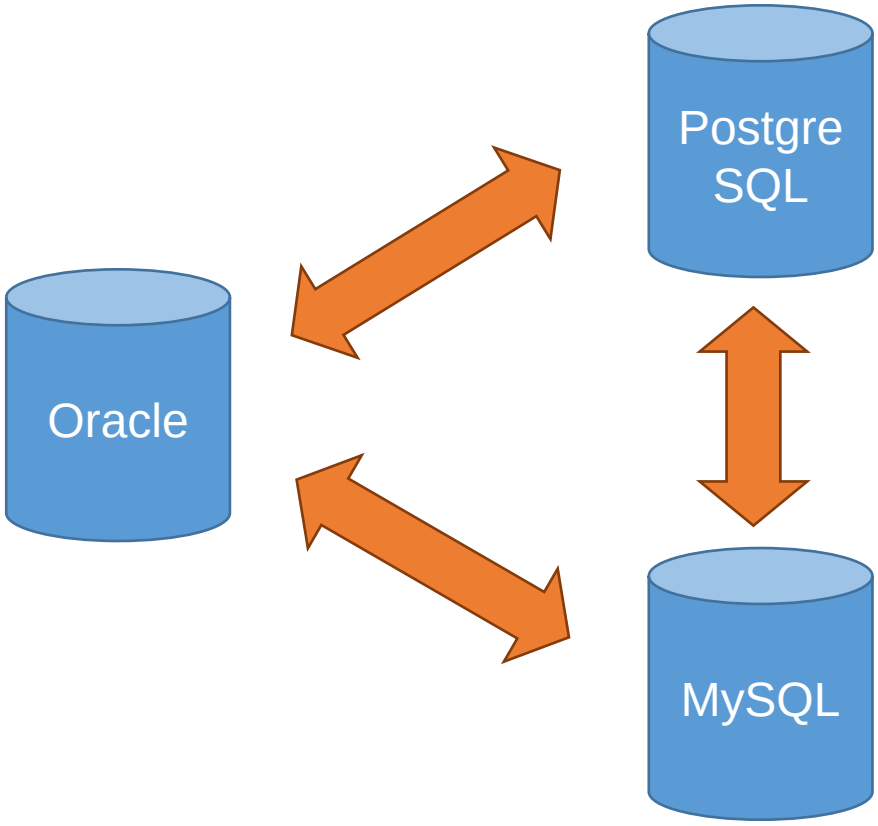


Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

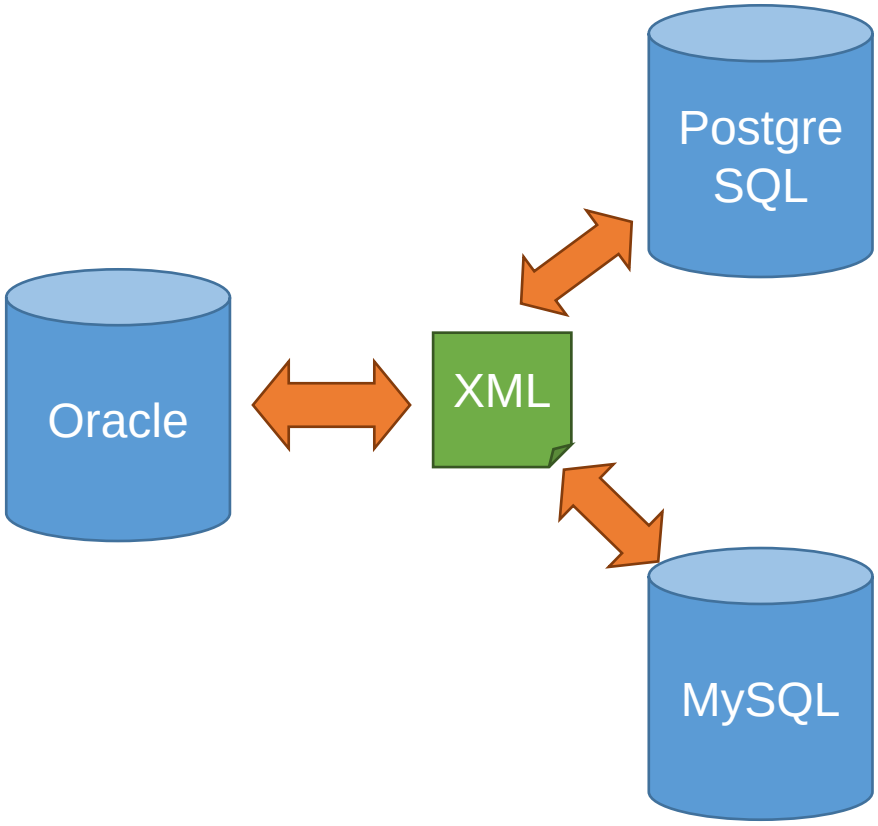
Utilité

- Exportation/Echange des données
- Sans XML : transformation entre chaque type de BD



Utilité (2)

- Avec XML : Format standardisé
- Nouvelle BD : 1 transformation (vs N transformations sans XML)



Utilité (3)

- Dans une base de données, le schéma est fixé et conservé séparément des données. En XML, le schéma (précisé par les annotations) est incorporé dans les données.

- L'intégration du schéma dans les données permet beaucoup de flexibilité. Le schéma et le contenu d'une base de donnée peuvent donc facilement être représentés en XML, et ce indépendamment du type de base de données utilisé.

- XML est donc un format très utile pour exporter de l'information d'une base de données et l'incorporer dans une autre.

- C'est aussi une format utile pour extraire des informations d'une base de données et les fournir à une application, par exemple un traitement de texte.

- Très utile également pour la configuration (ex : Maven)

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

Structure

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- Time-stamp: "bibliography.xml   3 Mar 2008 16:24:04" -->
<!DOCTYPE bibliography SYSTEM "bibliography.dtd" >
<bibliography>
    <book key="Michard01" lang="fr">
        <title>XML langage et applications</title>
        <author>Alain Michard</author>
        <year>2001</year>
        <publisher>Eyrolles</publisher>
        <isbn>2-212-09206-7</isbn>
        <url>http://www.editions-eyrolles/livres/michard/</url>
    </book>
    <book key="Zeldman03" lang="en">
        <title>Designing with web standards</title>
        <author>Jeffrey Zeldman</author>
        <year>2003</year>
        <publisher>New Riders</publisher>
        <isbn>0-7357-1201-8</isbn>
    </book>
  ...
</bibliography>
```

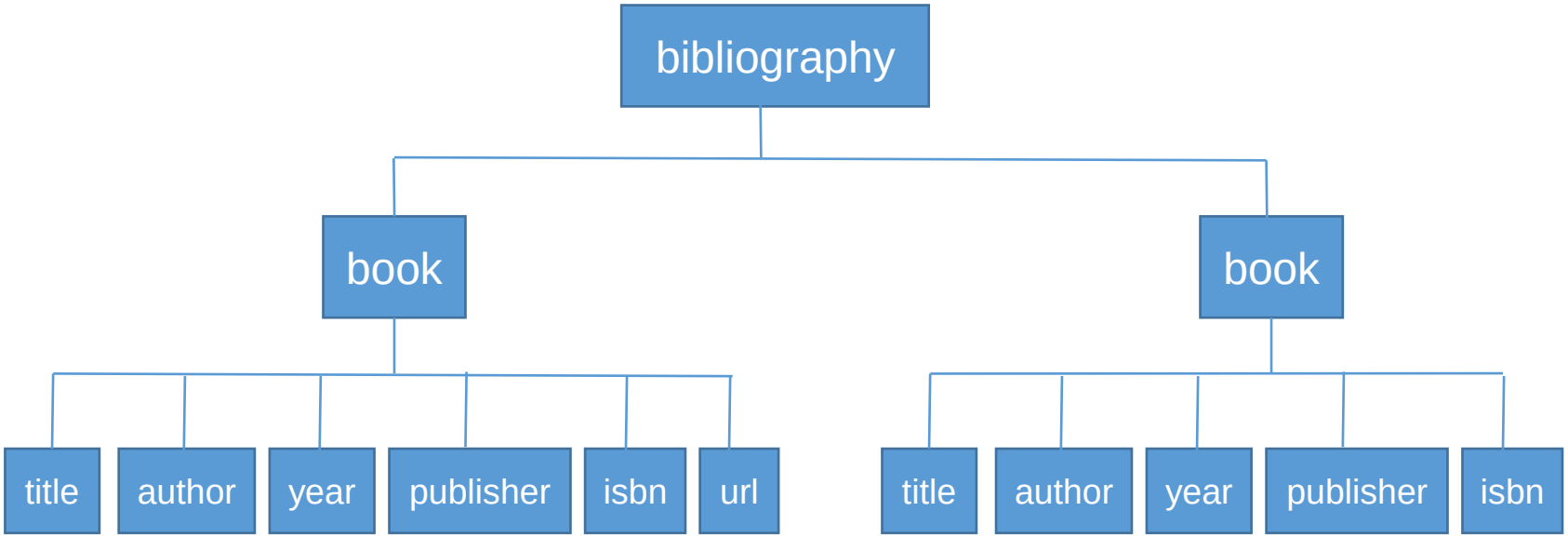
Légende

Entête
Commentaire
Déclaration de DTD
(plus tard)
Balise
Données

Source : Cours XML d'Olivier Carton

Structure (2)

Un fichier XML peut donc être vu comme un arbre



Structure (3)

Navigation dans un arbre

- On utilize le vocabulaire généralogique pour nommer les choses.
- *title* est un **enfant** de *book*, qui est son **parent**
- *title* et *authors* sont **frères**
- *bibliography* est l'**ancêtre** de chaque noeud élément
- Chaque noeud est le **descendant** de *bibliography*

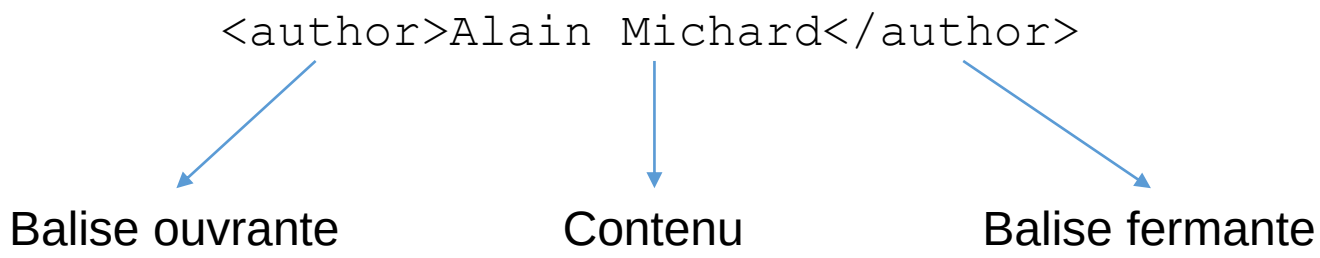
En-tête XML

- Optionnelle, mais fortement recommandée
- La première ligne du document. Rien avant, même pas un espace !
- Donne de l'information via 3 attributs : version, encoding et standalone
  - Version : `1.0` ou `1.1`
    - Version 1.1 très complexe pour pas grand chose, utilisez la version 1.0
  - Encoding
    - Encodage de caractères utilisé pour écrire le document
    - Par défaut: `UTF-8` (un sur-ensemble d'ASCII)
    - Si non spécifié, le parser pourra essayer de le déterminer en utilisant les premiers bytes
  - Standalone : `yes` ou `no`
    - `no` par défaut
    - Indique si le document nécessite des éléments définis de manière extérieure
    - Très complexe → Ne pas le préciser, sauf si vous voulez vérifier les détails

Elements

Un élément est constitué de

- Une balise ouvrante, dont le nom est un nom XML comme ”book” ou ”title” dans notre exemple
- Un contenu = éléments et/ou données de type caractère. Eventuellement vide
- Une balise fermante



Attributs

- Fait le lien entre un nom et une valeur
- Les attributs d'un élément se trouvent dans la balise ouvrante
- L'ordre n'a pas d'importance

- Syntaxe
  - `name="value"` ou `name='value'`

- Exemples
  - `<a href="./mapage.html" target="blank">`
  - `<book key="Michard01" lang="fr">`

Elements (2)

Elément racine

- Un élément contient tous les autres éléments
- On l'appelle élément "document" ou élément "root"
- Dans notre exemple, c'est "bibliography"

Elément vide

- Un élément peut être vide
- Il est écrit `<nom></nom>` sans un seul caractère (même pas un espace) entre les balises
- Version raccourcie, moins sujette à erreur: `<nom/>`
- Notez qu'il peut contenir des attributs

Domaine de valeurs

- Un nom de balise
  - Est composé de caractères
  - Est sensible à la casse
  - Caractères autorisés : Alphanumériques [a-z;A-Z;0-9], le tiret ‘-’, le point ‘.’, les deux-points ‘:’ et l’underscore ‘_’.
  - Un nom XML ne peut pas commencer par ‘-’, par ‘.’, par un chiffre, par « xml », et les deux-points ne peuvent apparaitre qu’une seule fois (pour les espaces de nom, voir plus tard)

- Le contenu XML
  - Est composé de caractères unicode (de 0 à 0x10FFFF)
  - Pas de réelle restrictions, sauf dans quelques cas précis

Contenu XML problématique

- Imaginons un contenu XML représentant une équation
- Ex :
  - `<equation>y = ax + b</equation>`

- Nous voulons maintenant représenter l'inéquation suivante
  - $y < 5x - 7$
- Et là, c'est le drame
  - `<equation>y < 5x - 7 </equation>`
  - Considéré comme une balise:
  - `<equation>y < 5x - 7 </equation>`

Erreur : un nom XML ne contient pas d'espaces

Solution : les entités

- Les entités sont des unités de stockage.
- Possèdent un nom et un contenu, appelé « replacement text ».
- On accède au contenu via une référence ‘&’

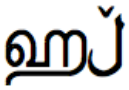
- Syntaxe
  - `&nom_de_l_entite;`
- Entités prédéfinies

nom	contenu
&lt;	<
&gt;	>
&amp;	&
&quot;	"
&apos;	‘

Entités et code point

- Autre solution : Utiliser une entité et une valeur unicode (code point) pour représenter n'importe quel caractère

- Syntaxe
  - `&#decimal;` ou `&#hexadecimal;`

- Exemple
  - `<text>&#x0BB9;&#x06B5;</text>`
  - Affiche 

Gestion des espaces

- Le traitement XML des caractères d'espacement est simple dans un premier temps.
- Chaque caractère d'espacement est équivalent à un espace et plusieurs espaces consécutifs sont encore équivalents à un seul espace.
- C'est le comportement habituel des langages de programmation classiques.
- Dans un second temps, le traitement des caractères d'espacement par une application est commandé par l'attribut `xml:space`.
- Les caractères d'espacement sont l'espace ' ' U+20, la tabulation U+09 ('\t' en notation du langage C), le saut de ligne U+0A ('\n' en C) et le retour chariot U+0D ('\r' en C).

Source : Cours XML d'Olivier Carton

Gestion des espaces (2)

- Les fins de lignes sont normalisées par l'analyseur lexical (*parser* en anglais). Ceci signifie que les différentes combinaisons de fin de ligne sont remplacées par un seul caractère U+0A avant d'être transmises à l'application. Cette transformation garantit une indépendance vis à vis des différents systèmes d'exploitation.
- Les combinaisons remplacées par cette normalisation sont les suivantes.
  - la suite des deux caractères U+0D U+0A
  - la suite des deux caractères U+0D U+85
  - le caractère U+85 appelé Next Line
  - le caractère U+2028 appelé Line Separator
  - le caractère U+0D non suivi de U+0A ou U+85
- Les deux caractères U+85 et U+2028 ne peuvent être correctement décodés qu'après la déclaration de l'encodage des caractères par l'entête. Leur usage dans l'entête est donc déconseillé.

Source : Cours XML d'Olivier Carton

Sections CDATA

- Insertion de texte brut qui ne sera pas parsé
- Plus facile que d'utiliser beaucoup d'entités
  - Par exemple, si la donnée à représenter est du code balisé…

- On pourra donc écrire

```
<![CDATA[
<p>Un extrait de code <i>HTML</i> dans un document
<b>XML</b></p> ]]>
```

- Au lieu de

```
&lt;p&gt;Un extrait de code &lt;i&gt;HTML&lt;/i&gt;
dans un document &lt;b&gt;XML&lt;/b&gt;&lt;/p&gt;
```

- Note : La suite de caractères «]]> » ne peut pas être utilisée dans une section CDATA, on utilisera alors >

Vocabulaire Français/Anglais

Français	English
Entité	Entity
Appel d’entité	Entity reference
Appel de caractère	Character reference
élément	Element
Balise	Tag
Balise ouvrante	Start-tag
Balise fermante	End-tag
Enfant	Child
Parent	Parent
Frère	Sibling
Ancêtre	Ancestor
Descendant	Descendant

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

Validation

- Un document XML vous laisse faire « ce que vous voulez ».
- Pas de contrainte sur la structure
- Comment vérifier si le document est bien écrit?

- Règles implicites de XML
  - Une balise ouvrante doit toujours correspondre avec une balise fermante
  - Règles de syntaxe
  - → Bien rédigé (well-formed)

- Schéma de validation
  - Un peu comme un schéma relationnel
  - Précise les règles de structure (tel élément est composé de tels autres éléments, attributs, etc.)
  - DTD ou XSD
  - → Valide (valid)

Validation (2)

- Comment vérifier la validité d'un fichier XML avec un DTD ou un XSD?
- Avec un *parser*

- Deux approches : SAX et DOM

- SAX (Simple API for XML)
  - Parcourt le fichier XML et génère des évènements (début de balise, fin de balise, etc)
  - Moins gourmand en ressources
  - Lecture séquentielle

- DOM (Document Object Model)
  - Génération d'un arbre en mémoire et parcourt de ce dernier
  - Plus gourmand en ressources
  - Parcourt de l'arbre possible

Recherche

- Possibilité d'extraire de l'information d'un fichier XML

- XPath
  - Utilise des expressions pour rechercher des nœuds et des ensembles de nœuds dans un document XML
  - Par exemple, `/bibliography/book[position()<3]` sélectionne les deux premiers éléments livres qui sont enfant de l'élément bibliography

- XQuery
  - C'est **LE** langage pour rechercher des données dans un document XML
  - XQuery est à XML ce que SQL est aux bases de données
  - Exemple :

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

Source : W3Schools

Transformation

- XML ne fait rien, il contient seulement de l'information
- Pour un affichage professionnel, il faut un outil de transformation

- CSS (Cascaded Style Sheet)
  - Feuille de style décrivant quoi faire avec un élément
  - Fonctionne aussi avec HTML

- XSLT (XML Stylesheet Language Transformation)
  - Transformations à l'aide du langage XSL
  - Permet de générer ce qu'on veut, y compris des pages HTML ou des documents XML
  - Se base sur XPath

Agenda

- Introduction à XML
  - Définition
  - Historique
  - Utilité
  - Syntaxe et structure
- Avant-gout de la suite
  - Validation
  - Recherche
  - Transformation
- Présentation du projet XML

Contexte

- La société **I**xe**M**e**Db**, dont le siège central se situe en dehors de l'union européenne pour éviter le plus longtemps possible les procès pour plagiat, a pour objectif de concurrencer d'autres plate-formes de présentation de films, comme IMDb et TMDb.

- Leur agent spécial a réussi à voler une partie des données du site de TMDb (grâce à l'API fournie par cette dernière…), mais pour que ce soit utilisable, il leur faut désormais structurer ces données (ils optent judicieusement pour le format XML) effectuer les transformations nécessaires pour pouvoir les présenter sur leur site web (grâce à XSLT).

Films

- La signalétique d'un film comporte toutes les informations relatives à ce film. Par exemple, son titre, sa durée, ses genres, le public concerné, les acteurs principaux, une affiche, ….
- Chaque film est une ligne (un enregistrement) stocké dans un fichier texte appelé movies.txt. On retrouve 15 champs par film
- Le format d'un enregistrement de la table externe est le suivant :

  _id(U+2023)title(U+2023)original_title(U+2023)release_date(U+2023)status(U+2023)vote_average(U+2023)vote_count(U+2023)runtime(U+2023)certification(U+2023)poster_path(U+2023)budget(U+2023)tagline(U+2023)genres:id(U+2024)name(U+2023)directors:id(U+2024)name(U+2023)actors:id(U+2024)name(U+2024)character

- Ou encore:

  _id▸title▸original_title▸release_date▸status▸vote_average▸vote_count▸runtime▸certification▸poster_path▸budget▸tagline▸genres:id.name▸directors:id.name▸actors:id.name.character.

Réalisations

- Conversion vers un document XML
  - Structure libre (mais au moins un attribut)
  - Langage : (PL/)SQL, Java, C, C++
- Document structure
  - Au minimum DTD. Eventuellement XSD
- Validation
  - Au minimum avec SAX, et calcul statistique. Eventuellement avec DOM et comparaison des performances (temps d'exécution et utilisation mémoire)
- Site Web
  - XSLT
  - Au minimum fonctionnel. Eventuellement mettre l'accent sur l'ergonomie, avec utilisation de technologies supplémentaires

Evaluation

- Compte pour 50% de la note de labo
  - Donc 25% de la note finale (**de ma partie**)

- Pondération
  - Conversion vers le document XML 5 points
  - Réalisation du DTD et/ou du XSD 4 points
  - Parsing et calcul statistique 8 points
  - Site web avec XSLT 8 points

- Valorisation
  - **Au minimum :** La base. Ce que j'attends vraiment au minimum du minimum. Arrêtez-vous là si vous visez la réussite simple (ce que vous aurez, mais seulement si c'est bien réalisé).
  - **Pour les pros :** Une réalisation un peu plus pro ou demandant un peu plus d'implications et de travail. Ne sera évalué que si l'item « au minimum » est réalisé. Pour ceux qui visent la distinction.
  - **Pour les experts :** Pour ceux que le sujet passionne et qui veulent vraiment se spécialiser dans cette discipline. Ou pour ceux qui visent l'excellence et la plus grande distinction.

Consignes supplémentaires

- A réaliser par groupe de deux
- A présenter au plus tard le 9 novembre 2022
  - Possibilité de présenter plus tôt

- Dans tous les cas : dossier complet (code source) à envoyer, par mail à samuel.hiard@hepl.be, le dimanche précédent la séance où vous comptez présenter, à 23h59 au plus tard.

- Amusez-vous bien ☺