Secteur Tertiaire Informatique
Filière étude - développement

Développer une application n-tiers

XML –

Accueil

Apprentissage

Période en entreprise

Évaluation

Code barre

SOMMAIRE

Ce livret ne se veut pas exhaustif : pour connaître tout sur la technologie visée, vous devrez vous référer aux sites et livres cités en référence.


LES BASES

I INTRODUCTION

I.1 QU'EST-CE QUE XML?

XML est un métalangage de description constitué de balises tout comme HTML.
Par contre, il dissocie le contenu d'une page de la façon de la présenter et permet la communication entre systèmes.

Tout comme son cousin HTML, XML est directement issu de SGML (Standard Generalized Markup Language = langage normalisé de balisage généralisé). Il est issu du monde de la GED (gestion électronique de documents). Ce n'est pas un langage "propriétaire".

XML est l'acronyme de e**X**tensible **M**arkup **L**anguage, cela signifie que XML n'est pas un langage sémantiquement figé comme peut l'être HTML mais au contraire un langage ouvert. L'auteur d'un document XML peut créer ses propres balises, par exemple la balise <SITE> peut être définie pour désigner un site touristique.

Cela s'écrirait de la façon suivante :

```
< SITE >Louvre</ SITE >
```

I.2 ORIGINE ET BUTS

Le W3C est un groupement d'entreprises reconnues dans le domaine informatique, qui a comme principal objectif de développer et promouvoir des technologies (protocoles de communication, langages de programmation, méthodes d'analyse) propres à l'Internet.

Cet organisme a un mécanisme spécifique pour adopter une norme. Lorsque le besoin d'une nouvelle technologie se fait ressentir, une note de proposition est passée au W3C qui va l'étudier, débattre de sa pertinence. Si elle est justifiée, une équipe est constituée, chargée de normaliser, développer puis promouvoir la nouvelle norme. Le premier stade d'une proposition est « requirement ». C'est ici que le besoin est exprimé, que les objectifs de la norme sont clairement notifiés. C'est en quelque sorte le cahier des charges de la norme.
Si elle est validée, elle passe à l'étape de « working draft », c'est à dire en cours de développement. L'équipe constituée par le W3C commence à concevoir, développer le projet dont elle a la charge. Il y aura autant de « working draft » que de versions présentées au W3C.
Lorsque le projet est au stade terminal, il passe en phase de « candidate recommandation ». Il restera à ce stade tant que des entreprises n'auront pas implémenté la norme, c'est à dire développé des outils permettant d'utiliser correctement et pleinement la norme.
Le dernier stade de l'évolution d'une norme du W3C est appelé « recommandation ». Cette étape signifie que la technologie est arrivée à maturité et qu'elle dispose d'outils fiables pour son utilisation.

XML a été développé par le XML Working Group (originellement connu comme le SGML Editorial Review Board) formé sous les auspices du **World Wide Web Consortium (**W3C**)** en 1996. Il était présidé par Jon Bosak de Sun Microsystems avec la participation active du XML Special Interest Group (anciennement SGML Working Group) également organisé par le W3C.

Les objectifs de conception de XML sont les suivants :

- XML est utilisable sans difficulté sur Internet;
- XML soutient une grande variété d'applications;
- XML est compatible avec SGML;
- Il est facile d'écrire des programmes traitant les documents XML;
- Le nombre d'options dans XML doit être réduit au minimum, idéalement à aucune;
- Les documents XML devraient être lisibles par l'homme et raisonnablement clairs;
- La conception de XML devrait être préparée rapidement;
- La conception de XML est formelle et concise;
- Il est facile de créer des documents XML;
- La concision dans le balisage de XML est de peu d'importance.

Remarque : Ceci est un extrait des spécifications officielles, bien entendu cela reste très théorique.

I.3 EST-CE LA FIN DE HTML ?

Bien évidement un langage quel qu'il soit ne peut s'imposer du jour au lendemain, cependant à terme, XML est appelé à succéder dignement à HTML.

Néanmoins, lorsque l'on se rend compte qu'il aura fallu environ 2 ans pour que la norme HTML 4.0 soit correctement implémentée dans les navigateurs du marché, on peut se dire que HTML a encore du temps devant lui. D'autant plus que beaucoup d'entreprises ou de particuliers utilisent encore des navigateurs comme Microsoft Internet Explorer ou Netscape Navigator dans leur ancienne version.

Cependant, les plus optimistes d'entre nous diront que XML va révolutionner l'internet, quant aux plus pessimistes, ils diront que XML est une simple évolution pour le WEB.

En étant réaliste, on ne peut envisager à l'heure actuelle d'architecturer tout un système d'information ou bien de construire un site internet entièrement en XML. D'une part, on ne dispose pas encore de tous les outils nécessaires à la conception de documents, d'autre part toutes les normes qui s'articulent autour de XML et notamment les spécifications concernant les feuilles de style ne sont pas encore achevées par le W3C.

Et pour finir, il n'existe à l'heure actuelle que Microsoft Internet Explorer 5.0 qui soit capable de comprendre le XML. A noter cependant que Mozilla - nom de code du prochain Netscape Navigator 5.0 - en version Bêta actuellement supportera aussi ce merveilleux langage.

Toutefois, on peut vraisemblablement imaginer que d'ici d'une à deux années XML sera réellement utilisé dans de bonnes conditions, ainsi XML pourra pleinement montrer toute sa puissance.

La norme HTML a donc encore de beaux jours devant elle...

Le langage XML est un langage à balises.

Les balises :	C'est un mot clef choisi par le concepteur du document qui permet de définir un élément. –Exemple : **<site>** Un couple de chevrons délimite une balise. A une balise ouvrante <MA_BALISE> correspond une fermante </MA_BALISE>.
Un nom de balise:	Il peut commencer par une lettre de l'alphabet ou le signe souligné _, La suite du nom est constituée de lettres, de chiffres, des signes soulignement, point ou tirets, Les espaces ne sont pas autorisés, Les noms XML distinguent les minuscules des majuscules.
Les balises vides :	 </BR>, sans éléments entre la balise ouvrante et la balise fermante peur s'écrire . Elles peuvent apparaître n'importe où.
Les éléments :	C'est un objet XML défini entre une balise de début et une balise de fin. –<site> – Contenu de l'élément –</site> Un élément peut contenir aussi d'autres éléments Un document XML a une structure d'arbre Les éléments ne doivent pas se chevaucher.
Les attributs :	Un élément peut être qualifié par un ou plusieurs attributs. Ces attributs ont la forme clef="valeur".
Les méta caractères:	❑ &amp devient à l'affichage & ❑ &lt devient à l'affichage < ❑ &gt devient à l'affichage > ❑ &quot devient à l'affichage " ❑ &apost devient à l'affichage ' ❑ Ceci évite les confusions avec les balises et les références d'entités. ❑ Ces méta-caractères sont aussi appelés références d'entité. ❑ Seules ces 5 références sont autorisées. Pour toute autre référence d'entité, il faut la définir dans un document DTD (vu dans les chapitres suivants).

Dans une application informatique de base, nous avons quatre concepts:

- La structure des données,
- La présentation
- La navigation
- Les traitements

Actuellement dans le Web, ces quatre aspects sont mélangés.

XML permet d'isoler chacun des aspects dans des instances différentes.

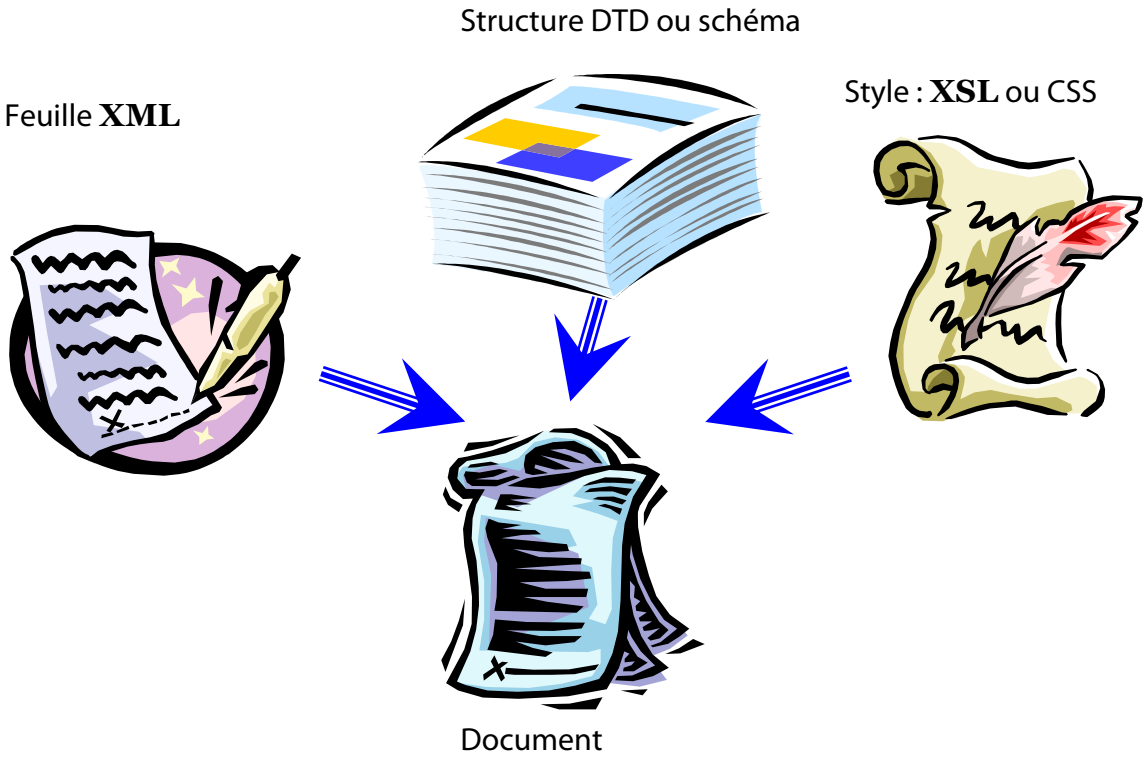
I.4 DEFINITION D'UN DOCUMENT

Afin de représenter un document en XML, il est nécessaire de posséder :

- une feuille XML qui contient les données (texte, image, vidéo, ...)
- une feuille de style XSL (ou CSS) pour la présentation physique des données
- éventuellement une DTD qui signifie *Document Type Définition* ou un schéma

La DTD est utilisée pour décrire la structure et le type des balises utilisées dans la feuille XML, ce qui bien évidemment donne une structure à notre document. A noter que son emploi est facultatif.

Dès lors, on dira qu'un document XML est "valide" *(valid)* s'il possède et respecte une DTD, et s'il n'en possède pas on utilisera le terme "bien formé" *(well formed)*. Un document dit bien formé doit se conformer aux règles de base de XML. En corollaire, un document valide (accompagné de sa DTD) est obligatoirement bien formé.



I.5 STRUCTURE D'UNE FEUILLE XML

Un document XML est composé de deux parties :
Le prologue, lui-même composés de plusieurs parties
Une **déclaration** XML, qui permet de définir :

- la version de XML utilisée,
- le codage des caractères
- la manière dont sont stockées les informations de balisage

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

Une **déclaration de type de document** (DTD) qui permet de définir la structure logique du document et sa validité.

```
<!DOCTYPE lace  SYSTEM "produit.dtd">
```

Une transformation définie par une feuille de style (CCS2 ou XSL) qui permet de définir la présentation.

```
<?xml-stylesheet type="text/css" href="produit.css"?>
```

L'instance qui correspond au balisage du document proprement-dit.
Le prologue équivaut à l'entête d'HTML. L'instance équivaut au corps d'HTML.
Dans l'exemple précédant, le document sera validé par une feuille externe appelée "produit.dtd", la présentation sera définie par une feuille de style appelée "produit.css" Ces deux lignes sont optionnelles.

I.6 LE DOCUMENT BIEN FORME

XML est "casse sensitive".
On peut utiliser des balises accentuées à condition d'utiliser ISO-8859-1.
Un document bien formé doit répondre aux règles suivantes:

- Un document XML ne contient qu'un élément nommé "élément racine".
- Toute balise ouverte doit être fermée ex : .....
- Les valeurs d'attributs doivent être entre en guillemets " "
- Les éléments contenus et contenant doivent être imbriqués.

Règle n° 1
Un document XML ne doit posséder qu'un seul élément racine qui contient tous les autres.
Un document XML est un arbre d'éléments imbriqués.

Document mal formé	Document bien formé
<?xml version="1.0" ?> <site> Carnac </site> <lieu> Bretagne </lieu>	<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu> Bretagne </lieu> </site>

Règle n° 2
Tous les éléments doivent être fermés

- A chaque balise ouvrant doit correspondre une balise fermante.
- A défaut, si un élément n'a pas de contenu, il est possible d'agréger la balise fermante à la balise ouvrant en terminant celle-ci par un "slash".

Document mal formé	Document bien formé
<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur="Bretagne"> </site>	<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur=" Bretagne " /> </site>

Règle n° 3

Les éléments contenus et contenant doivent être imbriqués.

- Tous les éléments fils doivent être contenus dans leur père.
- Si un document XML est un arbre, un élément est une branche.

Document mal formé	Document bien formé
<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur="Bretagne" /> <Curiosités> <appellation id="Menhir"> </Curiosités> </appellation> </site>	<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur="Bretagne" /> <Curiosités> <appellation id=" Menhir "> </appellation> </Curiosités> </site>

Règle n° 4

La valeur des attributs s'écrit entre guillemets.

- Tous les éléments fils doivent être contenus dans leur père.

Document mal formé	Document bien formé
<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur=Bretagne /> <Curiosités> </Curiosités> <Curiosités > </Curiosités> </site>	<?xml version="1.0" ?> <site> <nom> Carnac </nom> <lieu valeur="Bretagne" /> <Curiosités> </Curiosités> <Curiosités id="110"> </Curiosités> </site>

I.7 REALISONS NOTRE PREMIER DOCUMENT XML

La structure de base est très similaire à celle de HTML. Les feuilles XML peuvent être très simples et vous pouvez créer vos propres balises ou bien utiliser celles prédéfinies dans une DTD.

```
<?xml version="1.0" standalone="yes"?>
    <SALUTATIONS>
        Bonjour tout le monde !
    </ SALUTATIONS >
```

Vous pouvez saisir votre document XML à l'aide d'un éditeur classique type bloc notes de Windows.

Dans ce source, nous découvrons :

- des instructions de traitement
  <?XML version="1.0" standalone="yes"?>
- des balises ouvrantes et fermantes
  <SALUTATIONS> et </SALUTATIONS>
- du contenu
  Bonjour tout le monde !

 Prenez soin de sauvegarder votre document en format document texte et d'indiquer son nom entre guillemets.
Puis ouvrez votre document à l'aide de votre navigateur préféré.

Sens donné à une balise

Trois significations sont possibles :

- Structure : organisation du document représentée par un arbre d'éléments,
- Sémantique : liens avec le monde extérieur (exemple : fichiers),
- Style : comment est affiché ou traité le contenu de la page.

Structure d'un document

Une feuille XML est composée de la façon suivante :

- Un prologue : il contient diverses déclarations facultatives mais recommandées
- Du contenu du document avec les balises associées
- Des commentaires éventuels

•

Le prologue :
La première chose à indiquer est le type de document que l'on crée et l'existence ou non d'une DTD associée à ce document. Cela s'écrit sous la forme :

```
<?xml version='1.0' encoding='ISO-8859-1"standalone='yes'?>
```

Nous venons de déclarer un document du type XML dans sa version 1.0 qui utilise un encodage de type ISO-8859-1 (que nous verrons un peu plus loin) et qui ne possède pas de DTD.

 il faut bien faire attention à la casse, cette déclaration doit être écrite en minuscule sinon cela provoquera une erreur lors de l'interprétation du document

.

Si notre document possédait une DTD, nous aurions écrit :

```
<?xml version='1.0' encoding='UTF-8' standalone='no'?>
<!DOCTYPE exemple SYSTEM './exemple.DTD' >
```

Remarque: On rencontre aussi

```
<?xml version='1.0'?>
```

les autres attributs prennent alors les valeurs par défaut.

- encoding = 'UTF-8'
- . standalone='yes'

Les différents types de codage XML utilise les jeux de caractères de la norme ISO 10646 ou Unicode, pour plus d'informations vous trouverez la référence Unicode à l'adresse suivante :
http://www.unicode.org/
Les plus fréquemment utilisés par chez nous sont :

- •UTF-16 : codage des caractères sur 16 bits
- •UTF-8 : codage des caractères sur 8 bits
- •ISO-8859-1 : permet d'utiliser des noms de balises accentuées.

II POUR ALLER PLUS LOIN

II.1 LES ESPACES DE NOMS (NAMESPACES) XML

Depuis l'introduction des espaces de nom, la validation de documents XML utilisant des espaces de nom est devenue plus compliquée car les DTD ne les supportent pas. XML Schema comble cette lacune en ayant une intégration complète des espaces de nom. Dans cette partie, nous allons nous intéresser à la façon dont est supporté les espaces de nom ainsi qu'à leur utilisation dans les schémas XML et les documents instances

Qu'est-ce qu'un espace de nom ?

Un schéma peut-être vu comme une collection (ou vocabulaire) de définitions de types et de déclarations d'éléments dont le nom appartient à un espace de nom particulier appelé espace de nom cible (target namespace)

Le nombre de schémas XML étant potentiellement illimité, il peut apparaître des conflits si une balise est utilisée dans des schémas différents Si vous l'utilisez la balise 'sequence' comme une balise spécifique au schéma que vous créez, le processeur XML qui tentera de valider n'a aucun moyen de déterminer si l'élément sequence fait référence au connecteur de choix de XML Schema ou à vos propres définitions.

Une bonne métaphore consiste à comparer les espaces de nom aux packages du langage Java qui réduisent la portée de l'unicité du nom d'une classe à un sous-ensemble. Par exemple l'interface org.w3c.dom.Document est identifiable sans aucune ambiguïté et ne peut être confondue avec l'interface javax.swing.text.Document. Ceci parce que les deux interfaces n'appartiennent pas au même package. C'est exactement la même chose pour les espaces de nom de XML Schema

Il est en effet interdit d'avoir deux éléments de même nom mais ayant deux modèles de contenu différents dans le même espace de nom. Les espaces de nom permettent donc de fournir un contexte à un vocabulaire, ce qui facilite la création de schémas et la validation de documents.

L'espace de nom cible nous permet de différencier les définitions et déclarations de différents vocabulaires. La valeur de l'attribut targetNamespace doit être l'espace de nom associé au schéma.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema     xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
                        xmlns:sql="urn:schemas-microsoft-com:mapping-schema"
                        xmlns=monSite/Ressources/Schemas
                        targetNamespace=" monSite/Ressources/Schemas >

        <!-déclarations et définitions -->
</xsd:schema>
```

Le préfixe xsd:a été utilisé dans certains exemples précédents pour qualifier tous les éléments composant le langage XML Schema.

En fait, nous avons associé, grâce à la valeur de l'attribut xmlns, l'espace de nom de XML Schema avec le préfixe xsd. C'est ce qu'on appelle une qualification explicite. Pour être valide, tout élément appartenant au vocabulaire de XML Schema doit être précédé du préfixe xsd: (xsd:element par exemple). Signalons que nous aurions très bien pu choisir n'importe quel autre préfixe. Néanmoins, il est d'usage d'associer le préfixe xsd: à l'espace de nom de XML Schema.

Il existe également un procédé de qualification dit implicite qui utilise l'espace de nom par défaut. Tout document XML peut avoir un espace de nom par défaut ; c'est à dire que les éléments non préfixés du document appartiennent à l'espace de nom par défaut.

Espaces de noms réservés

Le préfixe xml est bien entendu réservé et ne doit pas être déclaré.

L'interopérabilité de XML suppose qu'il soit possible que des documents contiennent des balises réutilisables constituant des vocabulaires, des grammaires de différents secteurs utilisateurs.
Il est donc courant de mettre en œuvre plusieurs dtd (provenant de plusieurs domaines de spécialité) pour construire un document XML.
Les conflits de noms peuvent alors survenir si deux éléments portent des noms identiques dans deux dtd différentes (par exemple adresse peut représenter une adresse d'expédition ou une adresse mail).
Les domaines de noms (XML namespaces) sont une recommandation du W3C (rapidement adoptée après XML 1.0) pour résoudre le problème de conflits de noms dans un document XML. Le principe consiste à préfixer chaque nom d'élément par un nom « unique » qui identifie le domaine auquel il fait référence.

Les avantages de cette structuration

Les espaces de noms permettent de lever les ambiguïtés sur des éléments de dtd différentes qui porteraient le même nom (nom de produit, nom de personne, ..)
Les espaces de noms favorisent la modularité des documents XML et les sources multiples de ces derniers
Ils permettent, indirectement, de rendre plus lisible un document XML contenant des informations de sources diverses
Les namespaces ont un statut de recommandation du 14 Janvier 1999.
Un espace de noms est déclaré à l'aide de l'attribut xmlns:

- Soit en déclarant l'espace de nom dans 'élément

<MonElement xmlns="UriDTDaImporter">

Exemple d'utilisation sans préfixe

```
<site  xmlns="http//www.afpa.fr/exemples">
```

- Soit en associant un préfixe pour une utilisation plus fin

<MonElement xmlns:Préfixe="UriDTDaImporter">

Exemple d'utilisation avec préfixe

```
<site xmlns:afpa ="http!/www.afpa.fr/exemples">
<afpa:lieu>  ici  </afpa:lieu>
```

Pour assurer l'unicité des domaines, on a recours aux URI {Uniforn, Ressource Identifier), moyen standard pour accéder à des ressources internet.

II.2 VERIFICATION QU'UN DOCUMENT EST BIEN FORME EN JAVA

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

```
<!--
   Document   : MonDoc.xml
-->

<racine>
   <ele>Test </ele>
</root>
```

```
/*
 * TestSiBienForme.java
 */
import java.io.*;
import org.xml.sax.*;
import org.xml.sax.helpers.DefaultHandler;
import javax.xml.parsers.SAXParserFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.parsers.SAXParser;
/**
 */
public class TestSiBienForme
{
   public static void main(String[] args)
   {
      String strNomDoc = "C:/Documents and Settings/Propriétaire/Mes
documents/Professionnel/Xml/JavaXml/MonDoc.xml";
      DefaultHandler handler = new DefaultHandler();
      SAXParserFactory factory = SAXParserFactory.newInstance();
      try
      {
         SAXParser saxParser = factory.newSAXParser();
         saxParser.parse(new File(strNomDoc),handler);
         System.out.println("Le document " + strNomDoc + " est bien formé");
      }
      catch (Throwable e)
      {
         System.out.println("Le document " + strNomDoc + " est mal formé");
         e.printStackTrace();
      }
   }
}
```

Le résultat est :
Le document C:/Documents and Settings/Propriétaire/Mes documents/Professionnel/Xml/JavaXml/MonDoc.xml est mal formé
org.xml.sax.SAXParseException: "</racine>" prévu pour terminer l'élément qui commence à la ligne 7.
    at org.apache.crimson.parser.Parser2.fatal(Parser2.java:3182)
    at org.apache.crimson.parser.Parser2.fatal(Parser2.java:3176)
    at org.apache.crimson.parser.Parser2.maybeElement(Parser2.java:1513)
    at org.apache.crimson.parser.Parser2.parseInternal(Parser2.java:500)
    at org.apache.crimson.parser.Parser2.parse(Parser2.java:305)
    at org.apache.crimson.parser.XMLReaderImpl.parse(XMLReaderImpl.java:442)
    at javax.xml.parsers.SAXParser.parse(SAXParser.java:345)
    at javax.xml.parsers.SAXParser.parse(SAXParser.java:281)
    at TestSiBienForme.main(TestSiBienForme.java:22)

II.3 XML INCLUSIONS

La spécification XML Inclusions propose un mécanisme d'inclusion de ressources distantes dans un document XML.

La mise en oeuvre de cette proposition s'effectue par l'intermédiaire de l'insertion d'un élément <xi:include> à la position désirée pour l'inclusion de la ressource distante.

<xi:include href="URI" parse="type" encoding="encodage"/>

L'élément <xi:include> possède un espace de noms standardisé par le World Wide Web Consortium (W3C) : http://www.w3.org/2001/XInclude

Les inclusions peuvent se réaliser sur tout type de ressources XML, à l'image des noeuds d'élèments, d'attributs, de textes, d'instructions de traitement, de commentaire, mais aussi grâce aux XFointers, d'intervalles de noeuds ou de texte.

Les attributs

- href URI spécifie l'adresse URI de la ressource distante.
- parse xml ou text indique le type de format souhaité.
- encoding CDATA indique 'encodage de la ressource distante.

L'adresse URI de l'attribut href peut être absolue ou relative et peut contenir des fragments de localisation, soit des expressions XPointer.

L'attribut parse accepte deux valeurs,

- xml, indiquant que les ressources distantes doivent être analysées comme des données XML, puis sont fusionnées dans le document.
- text, signifiant que les ressources distantes doivent être incluses dans le document comme le contenu d'un noeud textuel.

Enfin, l'attribut d'encodage encoding accepte n'importe quel jeu de caractères standardisé, afin de traduire les données textuelles dans le format adéquat. Cet attribut n'a aucun effet lorsque parse possède la valeur xml.

L'utilisation des éléments et attributs d'inclusion, nécessite une déclaration préalable dans la Définition de Type de Document (DTD).

```
<!ELEMENT xi:include EMPTY>
<!ATTLIST xi:include
        xmlns:xi CDATA #FIXED "http//wwww3orgf200l/Xlnclude"
href    CDATA #REQUIRED
parse   (xml|text) "xml"
encoding CDATA #IMPLIED>
```

```
<!-- Document des éléments à inclure page.xml -->
<?xml version="1.0" encoding="ISO-8559-1" ?>
<!DOCTYPE page SYSTEM “defpage.dtd’>
 <page>
        <partie id="entete"> Librairie du Grand Colbert          </partie>
        <partie id="baspag">Copyright 2001 Groupe Library Ent SA 100 000 Francs</partie>
</page>
```

```
<!-- Document accueillant les inclusions : biblio.xml-->
<?xml version="1.0" encoding="ISO-8859-1"?>
< bibliothèque[
<!ELEMENT bibliothèque ANY>
<!ELEMENT xi:includle EMPTY>
<!ATTLIST xi:include
                xmlns:xi CDATA #FIXED "http://www.w3.org/2001/XInclude"
                href CDATA #REQUIRED
parse (xml|text) "xml"
encoding CDATA #IMPLIED>
<!ELEMENT < livre (titre,numero)>
<!ELEMENT <collection (titre.numero)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT numero (#PCDATA)>
]>
<bibliothèque>
        <xi:include  parse="xml” href="page.xml#xpointer(id('entete'))" />
        <livre>
                <titre>-La bible XML</titre>
                <numero>XM L0023</numero>
        </livre>
       <collection>
<titre>XML, par la pratique</titre>
<numero>XM Li 023</numero>
 </collection>
<piedpage>
<xi:include
parse="text"
encoding="ISO-8859-I"
href="page.xml#xpointer(string-range(/*[1]/*[2],
'Copyright 2001 Groupe Library Ent.'))"   />
</pied page>
 </bibliothèque>
```

```
<!-- Document résultant  -->
 <?xml version="1.0" encoding="lSO-8859-i" ?>
 <bibliothèque>
<partie id=’entete’>
Librairie du Grand Colbert
</partie>
<livre>
<titre>La bible XML</titre>
<numero>XM L0023</numero>
</livre>
<collection>
<titre>XML, par la pratique</titre>
<numero>XM Li 023</numero>
</collection>
<piedpage>
Copyright 2001 Groupe Library Ent.
</piedpage>
</bibliothèque>
```

Etablissement référent
DI Neuilly

Équipe de conception
Groupe études DI 2005

Remerciements :
Groupe études DI 2005

Date de mise à jour jj/mm/aa
afpa © Date de dépôt légal mois année

gagnez en compétences

afpa / Direction de l'Ingénierie13 place du Générale de Gaulle / 93108 Montreuil Cedex
association nationale pour la formation professionnelle des adultes
Ministère des Affaires sociales du Travail et de la Solidarité