Module : Introduction à XML

Elaboré par :

Youssef Ben Hammadi

(ISET Djerba)

Public cible : Classes de 2ème année L2 - TI

Plan:

I. **Généralités sur XML**

II. **Les DTDs**

III. **Les schémas XML**

I. Généralités

1. Introduction:

Qu'est ce que XML?

- XML est l'abréviation d' eXtensible Markup Language. Il s'agit d'un langage de balisage structuré destiné pour la description, le stockage et le transfert de données.
- XML est indépendante de toute plateforme.
- XML est une recommandation W3C.

La différence entre XML et HTML

- XML décrit la structure de données alors que HTML permet l'affichage de données.
- Les balises HTML sont prédéfinies (limitées) alors que les balises XML sont extensibles selon le besoin de l'utilisateur.
- XML ne remplace pas HTML mais ils se complètent car dans la plus part des application web, XML est utilisé pour le stockage et transfert de données alors que HTML est utilisé pour l'affichage (XML+XSLT➔HTML).

I. Généralités

1. Introduction:

Exemple:

Comparer les deux exemples de codes ci-dessous.

Code XML (Atelier1_Exemple1.xml)	Code HTML (Atelier1_Exemple1.html)
<BIBLIOTHEQUE> <LIVRE> <TITRE>titre livre 1</TITRE> <AUTEUR>auteur 1</AUTEUR> <EDITEUR>editeur 1</EDITEUR> </LIVRE> <LIVRE> <TITRE>titre livre 2</TITRE> <AUTEUR>auteur 2</AUTEUR> <EDITEUR>editeur 2</EDITEUR> </LIVRE> <LIVRE> </BIBLIOTHEQUE>	 titre livre 1 auteur 1 <u>editeur 1</u> titre livre 1 auteur 1 <u>editeur 1</u>

I. Généralités

1. **Introduction:**

   Que remarquez vous?

- Le code XML permet de décrire la structure des données d'une bibliothèque.
- Le code HTML permet de spécifier l'affichage de données
- Le code XML comporte de nouvelles balises: <livre>, <auteur>, etc.
- Le code HTML ne comprend que des balises prédéfinies: <p>, <b>, etc.
- Le code XML décrit une structure arborescente.

Structure d'un document XML

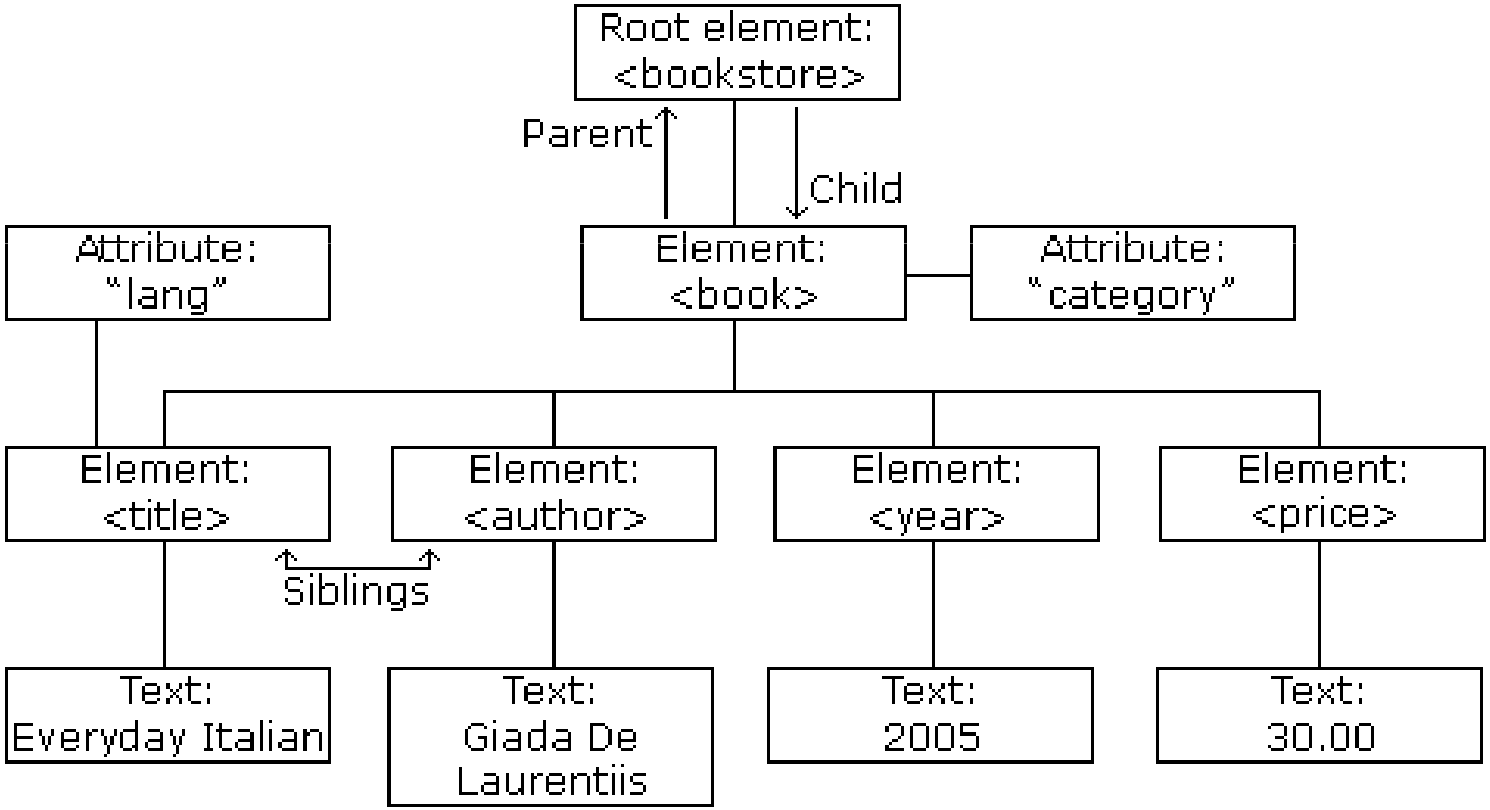
Schématiser les données de la bibliothèque sous forme d'une arbre et nommer ses différents éléments.

I. Généralités

1. Introduction:

Structure d'un document XML

Donner le code XML de cette structure.



I. Généralités

2. Les éléments et les attributs en XML

Les attributs:

```
<?xml version="1.0"?>
<BIBLIOTHEQUE>
    <LIVRE lang="arabe">
            <TITRE>titre livre 1</TITRE>
            <AUTEUR>auteur 1</AUTEUR>
            <EDITEUR>editeur 1</EDITEUR>
    </LIVRE>
    <LIVRE lang="arabe">
            <TITRE>titre livre 2</TITRE>
            <AUTEUR>auteur 2</AUTEUR>
            <EDITEUR>editeur 2</EDITEUR>
    </LIVRE>
</BIBLIOTHEQUE>
```

I. Généralités

2. Les éléments et les attributs en XML

En quoi se diffère ce code par apport au précédent?

- Un document XML est constitué par des éléments et des attributs.
- Chaque document comporte un élément racine (root element).
- L'élément racine comporte des sous-éléments (children elements)
- Un élément comprend du simple texte et/ou d'autres éléments
- Chaque élément peut avoir 0 ou plusieurs attributs

Comment choisir entre élément ou attribut?

- Ajouter l'information ISBN.
- Ajouter l'information DATE_EDITION.
- Argumenter votre choix (élément ou attribut).

I. Généralités

2. Les éléments et les attributs en XML

éléments vs attributs

```
<?xml version="1.0"?>
<BIBLIOTHEQUE>
    <LIVRE lang="arabe" ISBN="2-5678-6987-6">
            <DATE_EDITION>12-06-2010</DATE_EDITION>
            <TITRE>titre livre 1</TITRE>
            <AUTEUR>auteur 1</AUTEUR>
            <EDITEUR>editeur 1</EDITEUR>
    </LIVRE>
    <LIVRE lang='arabe' ISBN='6-9876-2354-9'>
            <DATE_EDITION>09-08-2010</DATE_EDITION>
            <TITRE>titre livre 2</TITRE>
            <AUTEUR>auteur 2</AUTEUR>
            <EDITEUR>editeur 2</EDITEUR>
    </LIVRE>
</BIBLIOTHEQUE>
```

I. Généralités

2. Les éléments et les attributs en XML

éléments vs attributs

- Les attribut ne peuvent pas contenir des informations multiples.
- Les attributs ne possèdent pas une structure arborescente.
- Les attributs sont non extensibles.
- Utiliser les éléments pour les données qui peuvent avoir des informations multiples ou une structure arborescente (informations relatives au données).
- Utiliser les attributs pour les métadonnées (informations non relatives au données).

I. Généralités

3. Les Règles d'écriture d'un document XML

Corriger le code ci-dessous pour qu'il soit correctement formé.

```
<?xml version="1.0"?>
<BIBLIOTHEQUE>
<LIVRE lang="arabe" ISBN="2-5678-6987-6'>
<TITRE>titre livre 1</TITRE>
<AUTEUR>auteur 1<AUTEUR>
<EDITEUR>editeur 1</EDITEUR>
</LIVRE>
<LIVRE lang='français' ISBN='6-9876-2354-9'>
<TITRE>titre livre 2</TTITRE>
<AUTEUR>auteur 2<EDITEUR>
</AUTEUR>editeur 2</EDITEUR>
</LIVRE>
</BIBLIOTHEQUE>
```

I. Généralités

3. Les Règles d'écriture d'un document XML

Pour avoir un document XML valide, appliques les règles suivantes:

- All XML Elements Must Have a Closing Tag
- XML Attributes Must be Quoted
- XML Tags are Case Sensitive
- XML Elements Must be Properly Nested
- XML Documents Must Have a Root Element
- Names can contain letters, numbers, and other characters
- Names cannot start with a number or punctuation character
- Names cannot start with the letters xml (or XML, or Xml, etc)
- Names cannot contain spaces
- Make names descriptive.

I. Généralités

3. Les Règles d'écriture d'un document XML

Les entités prédéfinies

Entity References	Character
&lt;	<
&gt;	>
&amp;	&
&quot;	"
&apos;	'

Les commentaires

<!-- This is a comment -->

I. Généralités

4. Exercices:

Exercice 1: Modéliser des articles avec bibliographie

L'objectif de l'exercice est de proposer un format XML permettant de stocker des articles quelconques. Un article est constitué d'un titre, d'un texte et d'une bibliographie. Le texte lui-même est une succession de paragraphes, chaque paragraphe pouvant contenir :

- des mots ou expressions importants et devant donc être différenciés du reste du paragraphe ;
- des références bibliographiques ;
- une entrée dans la bibliographie peut décrire soit un site web, soit un ouvrage ;
- un site web est décrit par un nom et une url ;
- pour un ouvrage, on trouve le titre, les auteurs, la date de parution et l'éditeur.

Questions :

1. Discuter des différentes possibilités de codage en XML.
2. Écrire une DTD et un document respectant cette DTD contenant au moins deux paragraphes et trois entrées bibliographiques (en utilisant les deux types d'entrées possibles).

I. Généralités

4. Exercices:

Exercice 2: Modéliser un site de brèves

Un site d'actualités veut présenter des nouvelles brèves, regroupées par thème. Quatre thèmes sont possibles : *actualités, sport, bourse* et *média*. Chaque brève correspond à un unique thème.

- Les brèves peuvent être rédigées en français ou anglais, chacune est datée et possède un titre.
- Il est également possible d'illustrer une brève par une photo et de fournir une ou plusieurs urls vers des sites détaillant l'information : chaque url sera agrémentée d'une courte phrase résumant le contenu de la page pointée.

Questions :

1. Discuter des différentes possibilités de codage en *XML*, en particulier pour la prise en compte de la langue et des thèmes.
2. Écrire une *DTD* et un document respectant cette *DTD* contenant au moins deux brèves.

II. Les DTD

1. Validation d'un document XML par un DTD

Définition:

Un DTD (Document Type Definition) défini la structure d'un document XML, ses éléments et ses attributs.

Utilités:

- With a DTD, each of your XML files can carry a description of its own format.
- With a DTD, independent groups of people can agree to use a standard DTD for interchanging data.
- Your application can use a standard DTD to verify that the data you receive from the outside world is valid.

II. Les DTD

1. **Validation d'un document XML par un DTD**

Déclaration d'un DTD

Nous voulons créer un DTD pour le fichier bibliotheque.xml.

- **Interne:**

Le DTD est déclaré à l'intérieur du fichier XML selon la syntaxe suivant:

<!DOCTYPE root-element [element-declarations]>

Exemple:

```
<?xml version="1.0"?>
<!DOCTYPE BIBLIOTHEQUE [
<!ELEMENT BIBLIOTHEQUE (LIVRE+)>
<!ELEMENT LIVRE (DATE_EDITION, TITRE, AUTEUR, EDITEUR)>
<!ELEMENT DATE_EDITION (#PCDATA)>
<!ELEMENT TITRE (#PCDATA)>
<!ELEMENT AUTEUR (#PCDATA)>
<!ELEMENT EDITEUR (#PCDATA)>
<!ATTLIST LIVRE
      lang CDATA #REQUIRED
      ISBN CDATA #REQUIRED
>
]>
<bibliotheque> <!-- contenu du fichier --> </bibliotheque>
```

II. Les DTD

1. Validation d'un document XML par un DTD

Déclaration d'un DTD

- Externe:

  Si le DTD est déclaré dans un fichier externe, il doit être lié au fichier XML selon la syntaxe suivant:

  <!DOCTYPE root-element SYSTEM "filename">

  Exemple:

```
 <?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT BIBLIOTHEQUE (LIVRE+)>
<!ELEMENT LIVRE (DATE_EDITION, TITRE, AUTEUR, EDITEUR)>
<!ELEMENT DATE_EDITION (#PCDATA)>
<!ELEMENT TITRE (#PCDATA)>
<!ELEMENT AUTEUR (#PCDATA)>
<!ELEMENT EDITEUR (#PCDATA)>
<!ATTLIST LIVRE
    lang CDATA #REQUIRED
    ISBN CDATA #REQUIRED
>
```

II. Les DTD

1. Validation d'un document XML par un DTD

Déclaration d'un DTD

- Externe:

  Si le DTD est déclaré dans un fichier externe, il doit être lié au fichier XML selon la syntaxe suivant:

  <!DOCTYPE root-element SYSTEM "filename">

  Exemple:

```
<?xml version="1.0"?>
<!DOCTYPE BIBLIOTHEQUE SYSTEM "C:\Users\youssef\
                                Desktop\Bibliothèque.dtd">

<BIBLIOTHEQUE>
 <!-- contenu du fichier -->
</BIBLIOTHEQUE>
```

II. Les DTD

1. **Validation d'un document XML par un DTD**

Eléments constitutifs d'un DTD

Un DTD regroupe dans sa déclarations:

- Des éléments
- Des attributs
- Des entités
- Des PCDATA (Parsed Character Data)
- Des CDATA (Character Data)

Les éléments:

- Déclaration:

<!ELEMENT element-name category>
or
<!ELEMENT element-name (element-content)>

II. Les DTD

1. **Validation d'un document XML par un DTD**

Les éléments:

- Empty elements

  <!ELEMENT element-name EMPTY>
- Elements with PCDATA

  <!ELEMENT element-name (#PCDATA)>
- Elements with Children (sequences)

  <!ELEMENT element-name (child1,child2,...)>
- Declaring Only One Occurrence of an Element

  <!ELEMENT element-name (child-name)>
- Declaring Minimum One Occurrence of an Element

  <!ELEMENT element-name (child-name+)>

II. Les DTD

1. **Validation d'un document XML par un DTD**

Les éléments:

- Declaring Zero or More Occurrences of an Element

  <!ELEMENT element-name (child-name*)>
- Declaring Zero or One Occurrences of an Element

  <!ELEMENT element-name (child-name?)>
- Declaring either/or Content

  <!ELEMENT note (child1, child2, (child3|child4))>

II. Les DTD

1. Validation d'un document XML par un DTD

Les attributs

- La déclaration d'un attribut se fait selon la syntaxe suivante:

  <!ATTLIST element-name attribute-name attribute-type default-value>

- Type des attributs

Type	Description
CDATA	The value is character data
(*en1*|*en2*|..)	The value must be one from an enumerated list
ID	The value is a unique id
IDREF	The value is the id of another element
IDREFS	The value is a list of other ids
NMTOKEN	The value is a valid XML name
NMTOKENS	The value is a list of valid XML names
ENTITY	The value is an entity
ENTITIES	The value is a list of entities
NOTATION	The value is a name of a notation
xml:	The value is a predefined xml value

II. Les DTD

1. **Validation d'un document XML par un DTD**

Les attributs

- Les valeurs par défaut

Value	Explanation
value	The default value of the attribute
#REQUIRED	The attribute is required
#IMPLIED	The attribute is not required
#FIXED *value*	The attribute value is fixed

- Value

DTD:
<!ELEMENT square EMPTY>
<!ATTLIST square width CDATA "0">

Valid XML:
<square width="100" />

II. Les DTD

1. **Validation d'un document XML par un DTD**

   Les attributs

- #REQUIRED

  syntaxe:

  <!ATTLIST element-name attribute-name attribute-type #REQUIRED>

  Exemple:

  DTD:
  <!ATTLIST person number CDATA #REQUIRED>

  Valid XML:
  <person number="5677"><name>Ahmed</name></person>

  Invalid XML:
  <person ><name>Ahmed</name></person>

II. Les DTD

1. **Validation d'un document XML par un DTD**

   Les attributs

- #IMPLIED

  Syntax:

  <!ATTLIST element-name attribute-name attribute-type #IMPLIED>

  Example:

  DTD:
  <!ATTLIST contact fax CDATA #IMPLIED>

  Valid XML:
  <contact fax="555-667788" />

  Valid XML:
  <contact />

II. Les DTD

1. **Validation d'un document XML par un DTD**

   Les attributs

- #FIXED

  Syntax:

  <!ATTLIST element-name attribute-name attribute-type #FIXED "value">

  Example:

  DTD:
  <!ATTLIST sender company CDATA #FIXED "Microsoft">

  Valid XML:
  <sender company="Microsoft" />

  Invalid XML:
  <sender company="W3Schools" />

II. Les DTD

1. **Validation d'un document XML par un DTD**

   Les attributs

- Enumerated Attribute Values

  Syntax:

  <!ATTLIST element-name attribute-name (en1|en2|..) default-value>

  Example:

  DTD:
  <!ATTLIST payment type (check|cash) "cash">

  XML example:
  <payment type="check" />
  or
  <payment type="cash" />

II. Les DTD

1. **Validation d'un document XML par un DTD**

Les entités:

- Les entités sont utilisées pur définir des raccourcis à des symboles spéciaux ou de simple standard texte.
- Déclaration:

Syntaxe:

<!ENTITY entity-name "entity-value">

Example:

DTD Example:
<!ENTITY writer "Donald Duck.">
<!ENTITY copyright "Copyright W3Schools.">

XML example:
<author>&writer;&copyright;</author>

II. Les DTD

1. **Exercice:**

   Enoncé:

   Créer un fichier XML pour stocker les données d'une bibliothèque, sachant que:

- Un auteur doit avoir au moins un livre
- Un auteur possède un nom, un prénom.
- Un livre est caractérisé par un titre, une langue, un éditeur, un ISBN et une catégorie.
- Les catégories sont: Informatique, Electrique et Mécanique.

Créer ensuite le DTD de ce fichier (interne et externe).

Le travail doit être argumenté.

II. Les DTD

1. **Exercice**

   Correction:

```
<?xml version="1.0"?>
<!DOCTYPE BIBLIOTHEQUE [

<!--DECLARATION DES ELEMENTS -->

<!ELEMENT BIBLIOTHEQUE (AUTEUR+)>
<!ELEMENT AUTEUR (NOM, PRENOM, LIVRE+)>
<!ELEMENT NOM (#PCDATA)>
<!ELEMENT PRENOM (#PCDATA)>
<!ELEMENT LIVRE (DATE_EDITION, TITRE, EDITEUR)>
<!ELEMENT DATE_EDITION (#PCDATA)>
<!ELEMENT EDITEUR (#PCDATA)>
<!ELEMENT TITRE (#PCDATA)>

<!--DECLARATION ATTRIBUTS -->

<!ATTLIST LIVRE lang CDATA #REQUIRED>
<!ATTLIST LIVRE ISBN CDATA #REQUIRED>
<!ATTLIST LIVRE category (Informatique | Mécanique | Electrique) "Informatique">

]>
```

II. Les DTD

1. Exercice

Correction (suite)

```
<BIBLIOTHEQUE>
    <AUTEUR>
        <NOM>Ahmed</NOM>
        <PRENOM>Mohamed</PRENOM>
        <LIVRE lang="arabe" ISBN="2-5678-6987-6" category="Informatique">
            <DATE_EDITION>12-06-2010</DATE_EDITION>
            <TITRE>titre livre 1</TITRE>
            <EDITEUR>editeur 1</EDITEUR>
        </LIVRE>
    </AUTEUR>
    <AUTEUR>
        <NOM>Youssef</NOM>
        <PRENOM>Ben Hammadi</PRENOM>
        <LIVRE lang="arabe" ISBN="6-9876-2354-9" category="Mécanique">
            <DATE_EDITION>09-08-2010</DATE_EDITION>
            <TITRE>titre livre 2</TITRE>
            <EDITEUR>editeur 2</EDITEUR>
        </LIVRE>
        <LIVRE lang="anglais" ISBN="6-9876-2354-9" category="Electrique">
            <DATE_EDITION>14-07-2008</DATE_EDITION>
            <TITRE>titre livre 3</TITRE>
            <EDITEUR>editeur 2</EDITEUR>
        </LIVRE>
    </AUTEUR>
</BIBLIOTHEQUE>
```

III. Les schémas XML:

1. Du'est ce que XML Schema?

Définition

- Un Schéma XML (XML schema Definition XSD) est un langage basé sur XML qui permet la définition de schémas (structure + type de données) des documents XML et facilite la communication entre applications.
- XML Schema est une alternative (successeur) pour les DTDs.

La différence entre les schémas XML et les DTDs

DTD	XML schema
N'est pas une syntaxe XML	Basé sur la syntaxe XML
Difficile à étendre	Facilement extensible
Données textuelles non typées	Supporte les types de données
Ne permet pas de spécifier exactement le nombre d'occurrences d'un élément	Permet de spécifier exactement le nombre d'occurrences d'un élément
Ne supporte pas les espaces de noms	Supporte les espaces de noms

III. Les schémas XML:

2. Exemple

Un simple document XML

```
<?xml version="1.0"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Un fichier DTD

```
<!ELEMENT note (to, from, heading, body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

III. Les schémas XML:

2. Exemple

Un schéma XML

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
<xs:element name="note">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="to" type="xs:string"/>
   <xs:element name="from" type="xs:string"/>
   <xs:element name="heading" type="xs:string"/>
   <xs:element name="body" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
</xs:schema>
```

III. Les schémas XML:

2. Exemple

Une référence à une DTD

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

Une référence à un schéma XML

```
<?xml version="1.0"?>
<note xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
    xsi:schemaLocation="note.xsd">
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

III. Les schémas XML:

3. Structure d'un XML Schema

L'élément <schema>

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
...
...
</xs:schema>
```

Le référencement d'un schéma dans un document XML

```
 <?xml version="1.0"?>
<note xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="note.xsd">

    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>

</note>
```

III. Les schémas XML:

3. Les éléments simples

Qu'est ce qu'un élément simple

Un élément simple est un élément XML qui peut contenir du texte seulement. Il ne peut pas contenir d'autres éléments ou d'attributs.

Définition d'un élément simple

<xs:element name="xxx" type="yyy"/>

Les types les plus communs sont:

xs: string

xs: decimal

xs: integer

xs: boolean

xs: date

xs: time

III. Les schémas XML:

3. Les éléments simples

Exemple

- Voici quelques éléments XML:

```
<lastname>Refsnes</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

- Et voici les définitions correspondantes (élément simple):

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

Valeur par défaut/fixe d'un élément simple

```
<xs:element name="color" type="xs:string" default="red"/>
<xs:element name="company" type="xs:string" fixed="Microsoft"/>
```

III. Les schémas XML:

3. Les attributs

Remarque

Tous les attributs sont déclarés comme des types simples. Si un élément a des attributs, il est considéré comme étant de type complexe. Mais l'attribut lui-même est toujours déclaré comme un type simple.

Comment définir un attribut?

La syntaxe pour définir un attribut est:

<xs:attribute name="xxx" type="yyy"/>

Exemple

Voici un élément XML avec un attribut:

<lastname lang="EN">Smith</lastname>

Et voici la définition de l'attribut correspondant:

<xs:attribute name="lang" type="xs:string"/>

Valeur par défaut/fixe d'un attribut

<xs:attribute name="genre" type="xs:string" default="female"/>

<xs:attribute name="lang" type="xs:string" fixed="EN"/>

III. Les schémas XML:

4. Les restrictions

Utilité

Les restrictions sont utilisées pour définir des valeurs acceptables pour les éléments ou les attributs XML. Les restrictions sur les éléments XML sont appelés facettes.

Restrictions sur les valeurs

```
<xs:element name="age">
 <xs:simpleType>
  <xs:restriction base="xs:integer">
   <xs:minInclusive value="0"/>
   <xs:maxInclusive value="120"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

III. Les schémas XML:

4. Les restrictions

Restrictions sur un ensemble de valeurs

```
<xs:element name="car">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:enumeration value="Audi"/>
   <xs:enumeration value="Golf"/>
   <xs:enumeration value="BMW"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

Restrictions sur une série de valeurs

```
<xs:element name="letter">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:pattern value="[a-z]"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

III. Les schémas XML:

4. Les restrictions

Expressions possibles

- <xs:pattern value="[A-Z][A-Z][A-Z]"/>
- <xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
- <xs:pattern value="[xyz]"/>
- <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
- <xs:pattern value="([a-z])*"/>
- <xs:pattern value="([a-z][A-Z])+"/>
- <xs:pattern value="m|f"/>

III. Les schémas XML:

4. Les restrictions

Restriction sur le longuer d'un élément

- ```
  <xs:element name="password">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:length value="8"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

- ```
  <xs:element name="password">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:minLength value="5"/>
        <xs:maxLength value="8"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  ```

III. Les schémas XML:

5. Les éléments complexes

Qu'est ce q'un élément complexe?

Un élément complexe est un élément XML qui contient d'autres éléments et / ou des attributs. Il existe quatre types d'éléments complexes:

- les éléments vides
- éléments qui contiennent uniquement des éléments autres
- éléments qui ne contiennent que du texte
- éléments qui contiennent les deux autres éléments et le texte

Note: Chacun de ces éléments peuvent contenir des attributs ainsi!

Exemples d'éléments complexes

- <product pid="1345"/>
- <employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>

III. Les schémas XML:

5. Les éléments complexes

Exemples d'éléments complexes

- <food type="dessert">Ice cream</food>
- <employee>
  This employee <firstname>John</firstname>
  <lastname>Smith</lastname> since <date>2002-04-23</date>. He is become a manager

  </employee>

Comment définir un élément complexe

soit l'élément complexe suivant:

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

III. Les schémas XML:

5. Les éléments complexes

Comment définir un élément complexe

cet élément peut être défini de deux façons:

- Sans définition de type:

```
<xs:element name="employee">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Si vous utilisez la méthode décrite ci-dessus, seul le terme «employé» élément peut utiliser le type complexe spécifié.

III. Les schémas XML:

5. Les éléments complexes

Comment définir un élément complexe

- Avec définition de type:

```
<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Si vous utilisez la méthode décrite ci-dessus, plusieurs éléments peuvent se référer au même type complexe:

```
<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>
```

III. Les schémas XML:

5. Les éléments complexes

Extension d'un élément complexe

Vous pouvez également baser un élément complexe sur un élément complexe existant et ajouter quelques éléments, comme ceci:

```
<xs:complexType name="personinfo">
 <xs:sequence>
  <xs:element name="firstname" type="xs:string"/>
  <xs:element name="lastname" type="xs:string"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
 <xs:complexContent>
  <xs:extension base="personinfo">
   <xs:sequence>
    <xs:element name="address" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="country" type="xs:string"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>


<xs:element name="employee" type="fullpersoninfo"/>
```

III. Les schémas XML:

6. Les éléments vides

Exemple:

- **Code XML**

```
<product prodid="1345" />
```

- **Définition 1:**

```
<xs:element name="product">
 <xs:complexType>
  <xs:attribute name="prodid" type="xs:positiveInteger"/>
 </xs:complexType>
</xs:element>
```

- **Définition 2:**

```
<xs:complexType name="prodtype">
<xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>

<xs:element name="product" type="prodtype"/>
```

III. Les schémas XML:

7. Les éléments(types) mixtes

Exemple:

- Un élément complexe de type mixte peut contenir des attributs, des éléments, et de texte.
- Code XML

```
<letter>
 Dear Mr.<name>John Smith</name>.
 Your order <orderid>1032</orderid>
 will be shipped on <shipdate>2001-07-13</shipdate>.
</letter>
```

III. Les schémas XML:

7. Les éléments(types) mixtes

Exemple:

- **Définition 1:**

```
<xs:element name="letter">
 <xs:complexType mixed="true">
  <xs:sequence>
   <xs:element name="name" type="xs:string"/>
   <xs:element name="orderid" type="xs:positiveInteger"/>
   <xs:element name="shipdate" type="xs:date"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

- **Définition 2:**

```
<xs:complexType name="lettertype" mixed="true">
 <xs:sequence>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="orderid" type="xs:positiveInteger"/>
  <xs:element name="shipdate" type="xs:date"/>
 </xs:sequence>
</xs:complexType>

<xs:element name="letter" type="lettertype"/>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs d'ordre:

- **All Indicator**

The <all> indicator specifies that the child elements can appear in any order, and that each child element must occur only once: <xs:element name="person">

```
<xs:complexType>
  <xs:all>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:all>
</xs:complexType>
</xs:element>
```

Note: When using the <all> indicator you can set the <minOccurs> indicator to 0 or 1 and the <maxOccurs> indicator can only be set to 1 (the <minOccurs> and <maxOccurs> are described later).

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs d'ordre:

- **Choice Indicator**

The <choice> indicator specifies that either one child element or another can occur:

```
<xs:element name="person">
  <xs:complexType>
    <xs:choice>
      <xs:element name="employee" type="employee"/>
      <xs:element name="member" type="member"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs d'ordre:

- **Sequence Indicator**

The <sequence> indicator specifies that the child elements must appear in a specific order:

```
<xs:element name="person">
  <xs:complexType>
   <xs:sequence>
     <xs:element name="firstname" type="xs:string"/>
     <xs:element name="lastname" type="xs:string"/>
   </xs:sequence>
  </xs:complexType>
</xs:element>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs d'occurrence:

Occurrence indicators are used to define how often an element can occur.

Note:

For all "Order" and "Group" indicators (any, all, choice, sequence, group name, and group reference) the default value for maxOccurs and minOccurs is 1.

```
<xs:element name="child_name" type="xs:string"
    maxOccurs="10" minOccurs="0"/>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs de group:

- **Element Groups**

```
<xs:group name="persongroup">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:element name="birthday" type="xs:date"/>
  </xs:sequence>
</xs:group>

<xs:element name="person" type="personinfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:group ref="persongroup"/>
    <xs:element name="country" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs de group:

- **Attribute Groups**

```
<xs:attributeGroup name="personattrgroup">
 <xs:attribute name="firstname" type="xs:string"/>
 <xs:attribute name="lastname" type="xs:string"/>
 <xs:attribute name="birthday" type="xs:date"/>
</xs:attributeGroup>

<xs:element name="person">
 <xs:complexType>
  <xs:attributeGroup ref="personattrgroup"/>
 </xs:complexType>
</xs:element>
```

III. Les schémas XML:

8. Les indicateurs:

Example:

- **Code XML:**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<persons xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="family.xsd">

<person>
  <full_name>Hege Refsnes</full_name>
  <child_name>Cecilie</child_name>
</person>
<person>
  <full_name>Tove Refsnes</full_name>
  <child_name>Hege</child_name>
  <child_name>Stale</child_name>
  <child_name>Jim</child_name>
  <child_name>Borge</child_name>
</person>
<person>
  <full_name>Stale Refsnes</full_name>
</person>
```



```
</persons>
```

III. Les schémas XML:

8. Les indicateurs:

Les indicateurs de group:

- **Schéma XML:**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

<xs:element name="persons">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="person" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string"
      minOccurs="0" maxOccurs="5"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>

</xs:schema>
```

III. Les schémas XML:

9. L'élément <any>:

Utilité:

L'élément <any> nous permet d'étendre le document XML avec des éléments non spécifiés par le schéma.

Scéma family.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
 <xs:element name="person">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:any minOccurs="0"/>
   </xs:sequence>
  </xs:complexType>
</xs:element>
```

III. Les schémas XML:

9. L'élément <any>:

Scéma children.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">

<xs:element name="children">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="childname" type="xs:string"
   maxOccurs="unbounded"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>

</xs:schema>
```

III. Les schémas XML:

9. L'élément <any>:

Le fichier family.xml

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

<persons xmlns="http://www.microsoft.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:SchemaLocation="family.xsd children.xsd">

<person>
  <firstname>Hege</firstname>
  <lastname>Refsnes</lastname>
  <children>
    <childname>Cecilie</childname>
  </children>
</person>

<person>
  <firstname>Stale</firstname>
  <lastname>Refsnes</lastname>
</person>

</persons>
```

III. Les schémas XML:

10. L'élément <anyAttribute>:

Utilité:

The <anyAttribute> element enables to extend the XML document with attributes not specified by the schema.

Scéma family.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">
 <xs:element name="person">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
   </xs:sequence>
   <xs:anyAttribute/>
  </xs:complexType>
</xs:element>
```

III. Les schémas XML:

10. L'élément <anyAttribute>:

Scéma attribute.sxd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified">

 <xs:attribute name="gender">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="male|female"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>

</xs:schema>
```

III. Les schémas XML:

10. L'élément <anyAttribute>:

Le fichier family.xml

The XML file below (called "Myfamily.xml"), uses components from two different schemas; "family.xsd" and "attribute.xsd":

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

<persons xmlns="http://www.microsoft.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:SchemaLocation="family.xsd attribute.xsd">

<person gender="female">
  <firstname>Hege</firstname>
  <lastname>Refsnes</lastname>
</person>

<person gender="male">
  <firstname>Stale</firstname>
  <lastname>Refsnes</lastname>
</person>

</persons>
```

III. Les schémas XML:

11. Exercice:

Enoncé:

Donner un schéma xml pour le fichier " shiporder.xml " suivant:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<shiporder orderid="889923"  >
 <orderperson>John Smith</orderperson>
 <shipto>
  <name>Ola Nordmann</name>
  <address>Langgt 23</address>
  <city>4000 Stavanger</city>
  <country>Norway</country>
 </shipto>
 <item>
  <title>Empire Burlesque</title>
  <note>Special Edition</note>
  <quantity>1</quantity>
  <price>10.90</price>
 </item>
 <item>
  <title>Hide your heart</title>
  <quantity>1</quantity>
  <price>9.90</price>
 </item>
</shiporder>
```

III. Les schémas XML:

11. Exercice:

Solution 1:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="shiporder">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="orderperson" type="xs:string"/>
   <xs:element name="shipto">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="address" type="xs:string"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
```

III. Les schémas XML:

11. Exercice:

Solution 1:

```
 <xs:element name="item" maxOccurs="unbounded">
      <xs:complexType>
       <xs:sequence>
         <xs:element name="title" type="xs:string"/>
         <xs:element name="note" type="xs:string" minOccurs="0"/>
         <xs:element name="quantity" type="xs:positiveInteger"/>
         <xs:element name="price" type="xs:decimal"/>
       </xs:sequence>
      </xs:complexType>
    </xs:element>
   </xs:sequence>
   <xs:attribute name="orderid" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```

III. Les schémas XML:

11. Exercice:

Solution 2:

The previous design method is very simple, but can be difficult to read and maintain when documents are complex.

The next design method is based on defining all elements and attributes first, and then referring to them using the ref attribute.

Here is the new design of the schema file ("shiporder.xsd"):

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- definition of simple elements -->
<xs:element name="orderperson" type="xs:string"/>
<xs:element name="name" type="xs:string"/>
<xs:element name="address" type="xs:string"/>
<xs:element name="city" type="xs:string"/>
<xs:element name="country" type="xs:string"/>
<xs:element name="title" type="xs:string"/>
<xs:element name="note" type="xs:string"/>
<xs:element name="quantity" type="xs:positiveInteger"/>
<xs:element name="price" type="xs:decimal"/>

<!-- definition of attributes -->
<xs:attribute name="orderid" type="xs:string"/>
```

III. Les schémas XML:

11. Exercice:

Solution 2:

```
<!-- definition of complex elements -->
<xs:element name="shipto">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="name"/>
   <xs:element ref="address"/>
   <xs:element ref="city"/>
   <xs:element ref="country"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>

<xs:element name="item">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="title"/>
   <xs:element ref="note" minOccurs="0"/>
   <xs:element ref="quantity"/>
   <xs:element ref="price"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

III. Les schémas XML:

11. Exercice:

Solution 2:

```
<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="orderperson"/>
      <xs:element ref="shipto"/>
      <xs:element ref="item" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="orderid" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```