Définition des fichiers XML par les schémas

XML Schema Definition ou XSD

DTD et Schéma XML

- Un schéma comme une DTD permet de décrire la structure d'une classe de fichiers XML.
- Il est plus récent que la DTD et par conséquent plus fin en particulier sur les types de données.
- Contrairement à une DTD, un schéma est un fichier XML

Exemple de XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="personne">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nom" type="xs:string"/>
        <xs:element name="prenom" type="xs:string"/>
        <xs:element name="date_naiss" type="xs:date"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Exemple d'instance de XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<personne
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="personne.xsd">
  <nom>de Latour</nom>
  <prenom>Jean</prenom>
  <date_naiss>1967-08-03</date_naiss>
</personne>
```

Références

- XSD : recommandation W3C
  3 documents incluant DTD et schema des schemas
  - XML Schema Part 0 : Primer Second Edition (résumé)
    http://www.w3.org/TR/xmlschema-0
  - XML Schema Part 1 : Structures
    http://www.w3.org/TR/xmlschema-1
  - XML Schema Part 2 : Datatypes
    http://www.w3.org/TR/xmlschema-2
- http://www.w3schools.com (tutorial)

XSD et les types de données simples

- string
- boolean
- float
- double
- decimal
- timeDuration
- recurringDuration
- date
- uriReference
- ID
- IDREF
- ENTITY
- NOTATION
- Qname
- binary

XSD : Element simple

- Un élément simple ne peut contenir que du texte typé.
- Un élément simple ne peut pas contenir d'attribut.
- La syntaxe pour déclarer un élément simple est

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

- Exemple :

```
<xs:element name="nom" type="xs:string"/>
```

XSD : Element simple

- On peut donner une valeur par défaut. Cette clause est prise en compte quand aucune valeur n'est spécifiée. La syntaxe est :

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

- On peut fixer une valeur. Aucune autre valeur n'est autorisée.

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

XSD : Restriction de valeurs

- On peut définir des restrictions de valeurs dans un élément.
- La syntaxe est :

```
<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>
```

XSD : Restriction d'ensemble de valeurs

- On peut définir des restrictions d'ensemble de valeurs dans un élément.
- La syntaxe est :

```
<xs:element name="voiture">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Picasso"/>
      < xs:enumeration value="Ferrari"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XSD : Restriction d'ensemble de valeurs

- Une autre syntaxe est :

```
<xs:element name="voiture" type="TypeAuto" />

<xs:simpleType name="TypeAuto">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Picasso"/>
    < xs:enumeration value="Ferrari"/>
  </xs:restriction>
</xs:simpleType>
```

- Dans ce cas, le type ainsi défini peut être réutilisé pour d'autres éléments.

XSD : Restriction d'ensemble de valeurs à l'aide de pattern

- On peut définir des ensembles de valeurs à l'aide d'expressions régulières.
- Les symboles d'une expression régulière sont :
  - […] : un caractère parmi ceux entre crochets. On peut utiliser le symbole – pour définir une séquence de caractères possibles (a-z : toutes les minuscules de l'alphabet)
  - + : répétition de 1 à plusieurs
  - * : répétition de 0 à plusieurs
  - {n} : le nombre de caractères est fixé à n
  - val1|val2 : choix entre plusieurs valeurs prédéfinies.

XSD : Gestion des *whitespace*

- On peut préserver ou ne pas préserver les whitespace : blanc, tabulation, retour ligne, ….

```
<xs:element name="address">
  <xs:simpleType>
   <xs:restriction base="xs:string">
        <xs:whiteSpace value="xxx"/>        </xs:restriction>
  </xs:simpleType>
</xs:element>
```

- **xxx= preserve alors les *whitespace* sont conservés.**
  xxx= replace alors les *whitespace* sont remplacés par des espaces.
  xxx = collapse alors toute séquence de plusieurs *whitespace* est remplacé par un seul espace.

XSD : Restriction de longueur

- On peut imposer une longueur pour un élement simple avec les balises xs:length, xs:minLength et xs:maxLength.

```
<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>
```

XSD : Element complexe

- Un élément complexe est un élément qui contient d'autres éléments et/ou un ou plusieurs attributs.
- Un élément complexe est associé à un type complexe via la balise xs:complexType.

XSD : Element complexe

- La syntaxe pour déclarer un tel élément est :

```
<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>
```

XSD : Element complexe

- Pour réutiliser le type, on peut aussi écrire :

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

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

XSD : Extension de type

- On peut étendre un type complexe :

```
<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>
```

Indicateurs de Type Complexe

- Indicateurs d'ordre
  - all : tous / ordre indifférent
  - sequence : tous / dans l'ordre
    - équivalent DTD : (choix1, choix2, ...)
  - choice : un au choix
    - équivalent DTD : (choix1 | choix2 | ...)
- Indicateurs d'occurence
  - minOccurs (défaut 1)
  - maxOccurs (défaut 1)
  - valeur infinie : unbounded

Exemple d'indicateur

- (elt1,elt2*,(elt3|elt4+))*

```
<xs:sequence minOccurs="0" maxOccurs="unbounded">
  <xs:element name="elt1"/>
  <xs:element name="elt2" minOccurs="0"
                    maxOccurs="unbounded"/>
  <xs:choice>
    <xs:element name="elt3"/>
    <xs:element name="elt4" maxOccurs="unbounded"/>
  </xs:choice>
</xs:sequence>
```

Occurrences Standards

- ? : minOccurs="0" maxOccurs="1"
- + : minOccurs="1" maxOccurs="unbounded"
- * : minOccurs="0" maxOccurs="unbounded"
- {3} : minOccurs="3" maxOccurs="3"
- ...

Groupe d'éléments

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

XSD : Définition d'attributs

- Un attribut est une valeur unique d'un nœud déclaré à l'intérieur de la balise du noeud.

- La syntaxe pour déclarer un attribut est

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

- On peut donner une valeur par défaut

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

XSD : Définition d'attributs

- On peut fixer une valeur sans changement possible

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

- Un attribut peut être optionnel ou obligatoire. Par défaut, il est optionnel. Pour le rendre obligatoire

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

Exemple d'attribut

- XML :

```
<dvd titre="Le Retour du Roi"> ...
```

- XSD :

```
<xs:element name="dvd">
  <xs:complexType>
    <xs:attribute name="titre"   type="xs:string“
   use="required"/>
    ...
  </xs:complexType>
</xs:element>
```

Groupe d'attributs

```
<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>
```