62-31.3 XML

XML-02-P05-Creation_types_complexes

Cédric Benoit

Plan du cours

- Présentation et organisation
- Introduction à XML
- Schémas XML
- **Types de données**
  - Types prédéfinis
  - Création de types simples
  - **Création de types complexes**

Points abordés

- Introduction
- Simple Content Model
- Complex Content Model
- Mixed Content Model
- Empty Content Model

Introduction (1)

Il existe quatre "classes" de types complexes :

- SimpleContent
- ComplexContent
- MixedContent
- EmptyContent

Rappel

		Texte	Eléments	Attributs
SimpleType		OUI	NON	NON
ComplexType - simpleContent		OUI	NON	OUI
ComplexType - complexContent		NON	OUI	OUI
ComplexType - mixedContent		OUI	OUI	OUI
ComplexType - emptyContent		NON	NON	OUI

Introduction (2)

Le type complexe "SimpleContent" :

- Ne peut contenir que du texte
- Peut contenir des attributs

Exemple:

```
<food type="dessert">Ice cream</food>
```

Introduction (3)

Le type complexe "ComplexContent" :

- Ne peut contenir que des éléments, mais pas de texte
- Peut contenir des attributs

Exemple:

```
<employe Type="fixe">
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employe>
```

Introduction (4)

Le type complexe "MixedContent" :

- Peut contenir des éléments et du texte
- Peut contenir des attributs

Exemple:

```
<description>
It happened on <date lang="norwegian"> 03.03.99 </date> ....
</description>
```

Introduction (5)

- Le type complexe "EmptyContent" :
  - Ne peut pas contenir d'autres éléments et ni de texte
  - Peut contenir des attributs

Exemple:

```
<product RefId="F1345"/>
```

SimpleContent

Ce type complexe ne contient que des attributs et du texte.

Création en partant d'un type simple et en ajoutant une série d'attributs.

Mécanisme nommé : ***extension***.

SimpleContent - Type anonyme - Exemple

```
<xs:element name="Titre">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute ref="langue"/>
                <xs:attribute ref="note"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>
```

SimpleContent - Type nommé - Exemple

```
<xs:complexType name="TypeTitre">
    <xs:simpleContent>
        <xs:extension base="xs:string">
            <xs:attribute ref="langue"/>
            <xs:attribute ref="note"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
<xs:element name="Titre" type="TypeTitre"/>
```

SimpleContent – Restriction et Extension (1)

Déclarer la restriction en premier avec SimpleType.

Exemple :

```
<xs:simpleType name="TypeString255">
        <xs:restriction base="xs:string">
                <xs:maxLength value="255"/>
        </xs:restriction>
</xs:simpleType>
```

SimpleContent – Restriction et Extension (2)

Puis déclarer le type avec l'extension basé sur le type simple personnalisé. Exemple :

```
<xs:complexType name="TypeTitre255">
    <xs:simpleContent>
        <xs:extension base="TypeString255">
            <xs:attribute ref="langue"/>
            <xs:attribute ref="note"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

ComplexContent (1)

Création en définissant une liste (et un ordre) d'éléments et d'attributs.

ComplexContent ne peut contenir que des éléments et des attributs.

ComplexContent (2) - Exemple

```
<xs:element name="auteur">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="nom"/>
            <xs:element ref="prenom"/>
            <xs:element ref="datenaissance" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

ComplexContent (3)

La structure complexContent est construite avec les outils suivants :

- all
- choice
- sequence
- maxOccurs, minOccurs
- group, attributeGroup
- any, anyAttribute
- substitution

ComplexContent – All (1)

L'outil **all** indique que chaque élément de la liste :

- peut apparaître dans n'importe quel ordre
- ne doit apparaître qu'une fois au maximum
- minOccurs = 0 ou 1
- maxOccurs = 1

ComplexContent – All (2)

Exemple :

```
<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>
```

ComplexContent – Choice

L'outil **choice** indique qu'un des éléments de la liste peut être présent. Exemple :

```
<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>
```

ComplexContent – Sequence

L'outil **sequence** permet de fixer l'ordre des éléments.

Exemple :

```
<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>
```

ComplexContent – Occurrences (1)

Une occurrence indique la fréquence d'apparition des éléments. Il est possible d'indiquer l'occurrence :

- minimale → **minOccurs** (défaut : 1)
- maximale → **maxOccurs** (défaut : 1)

Pour l'occurrence maximale, la valeur infinie est par **unbouded**.

ComplexContent – Occurrences (2)

Exemple :

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

ComplexContent – Group (1)

L'outil **group** permet de définir un groupe d'éléments. Dans cet outil, il est possible d'utiliser les outils mentionné précédemment comme :

- all
- choice
- sequence

ComplexContent – Group (2)

Exemple :

```
<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>

...
```

ComplexContent – Group (3)

Exemple :

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

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

ComplexContent – AttributeGroup (1)

L'outil **attributeGroup** permet de définir un 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>
```

ComplexContent – Any (1)

L'outil **any** permet d'étendre un document xml avec des éléments qui ne sont pas spécifiés dans le schéma.

ComplexContent – Any (2)

```
<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>

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

ComplexContent – Any (3)

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

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

</persons>
```

ComplexContent – AnyAttribute (1)

L'outil **anyAttribute** permet d'étendre un document xml avec des attributs qui ne sont pas spécifiés dans le schéma.

ComplexContent – AnyAttribute (2)

```
<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>
```

ComplexContent – AnyAttribute (3)

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

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

ComplexContent – Substitution (1)

L'outil **substitutionGroup** permet de mettre en place un système de traduction d'éléments (substitution). Par exemple dans un système d'information multi-langue.

ComplexContent – Substitution (2)

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

<xs:complexType name="custinfo">
  <xs:sequence>
    <xs:element ref="name"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="customer" type="custinfo"/>
<xs:element name="kunde" substitutionGroup="customer"/>
```

ComplexContent – Substitution (3)

```
<customer>
  <name>John Smith</name>
</customer>
```

OK

```
 <kunde>
   <navn>John Smith</navn>
 </kunde>
```

OK

ComplexContent – Substitution (4)

Il est possible de bloquer la substitution

```
<xs:element name="name" type="xs:string" block="substitution"/>
<xs:element name="navn" substitutionGroup="name"/>

<xs:complexType name="custinfo">
 <xs:sequence>
  <xs:element ref="name"/>
 </xs:sequence>
</xs:complexType>

<xs:element name="customer" type="custinfo" block="substitution"/>
<xs:element name="kunde" substitutionGroup="customer"/>
```

ComplexContent – Substitution (5)

```
<customer>
  <name>John Smith</name>
</customer>
```

OK

```
 <kunde>
   <navn>John Smith</navn>
 </kunde>
```

Pas OK

MixedContent (1)

Un élément complexe (ComplexType) peut contenir du texte entre les éléments. Pour cela, il faut utiliser l'attribut **mixed** et lui attriber la valeur **true** (mixed="true").

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

MixedContent (2) – Type anonyme

Exemple avec un **complexType** anonyme :

```
<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>
```

MixedContent (3) – Type nommé

Exemple avec un **complexType** nommé :

```
<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"/>
```

EmptyContent (1)

Un élément complexe (ComplexType) peut être vide (**EmptyContent**). Dans ce cas, il ne contient que des attributs.

- Un attribut est par défaut optionnel → *optional*
- Si l'attribut doit être obligatoire → *required*
- Si l'attribut ne doit pas être saisi (utilisé avec une restriction) → *prohibitued*

Cet élément n'a pas de texte (même pas des espaces), pas d'éléments et peut être construit à partir de

- simpleContent
- complexType

EmptyContent (2)

Exemple à partir de simpleContent :

```
<xs:simpleType name="empty">
        <xs:restriction base="xs:string">
                <xs:enumeration value=""/>
        </xs:restriction>
</xs:simpleType>

<xs:complexType name="TypeCommandList">
        <xs:simpleContent>
                <xs:extension base="empty">
                        <xs:attribute name="idList" type="xs:IDREFS"/>
                </xs:extension>
        </xs:simpleContent>
</xs:complexType>
```

EmptyContent (3)

Exemple à partir de complexType

```
<xs:complexType name="TypeCommandList2">
    <xs:attribute name="idList" type="xs:IDREFS"/>
</xs:complexType>
...
<xs:element name="letter" type="lettertype"/>
```

Dans le document XML, on peut avoir les valeurs suivantes :

```
<commandList idList="T1 T2"/>
```

Merci pour votre attention !