TD2 : espaces des noms et schéma

Ex1.Espace des noms

Objectif: Cette section a pour objectif de vous familiariser avec la portée des espaces des noms dans un document XML (à commenter uniquement).

Q : Pour chaque portion de code, associer chaque élément et attribut à son espace des noms:

- ```
  <lower:aaa xmlns:lower = "http://zvon.org/lowercase" >
    <lower:bbb xmlns:lower = "http://zvon.org/lowercase" >
     <lower:ccc xmlns:lower = "http://zvon.org/lowercase" />
    </lower:bbb>
    <upper:BBB xmlns:upper = "http://zvon.org/uppercase" >
     <upper:CCC xmlns:upper = "http://zvon.org/uppercase" />
    </upper:BBB>
    <xnumber:x111 xmlns:xnumber = "http://zvon.org/xnumber" >
     <xnumber:x222 xmlns:xnumber = "http://zvon.org/xnumber" />
    </xnumber:x111>
  </lower:aaa>
  ```
- ```
  <lower:aaa xmlns:lower = "http://zvon.org/lowercase" >
    <lower:bbb >
     <lower:ccc />
    </lower:bbb>
    <upper:BBB xmlns:upper = "http://zvon.org/uppercase" >
     <upper:CCC />
    </upper:BBB>
    <xnumber:x111 xmlns:xnumber = "http://zvon.org/xnumber" >
     <xnumber:x222 />
    </xnumber:x111>
  </lower:aaa>
  ```
- ```
  <lower:aaa xmlns:lower = "http://zvon.org/lowercase" xmlns:upper = "http://zvon.org/uppercase" xmlns:xnumber = "http://zvon.org/xnumber" >
    <lower:bbb >
     <lower:ccc />
    </lower:bbb>
    <upper:BBB >
     <upper:CCC />
    </upper:BBB>
    <xnumber:x111 >
     <xnumber:x222 />
    </xnumber:x111>
  </lower:aaa>
  ```
- ```
  <lower:aaa xmlns:lower = "http://zvon.org/lowercase" xmlns:upper = "http://zvon.org/lowercase" xmlns:xnumber = "http://zvon.org/lowercase" >
    <lower:bbb >
     <lower:ccc />
    </lower:bbb>
    <upper:BBB >
     <upper:CCC />
    </upper:BBB>
  ```

```
 <xnumber:x111 >
  <xnumber:x222 />
 </xnumber:x111>
</lower:aaa>
```

- ```
<aaa >
 <lower:bbb xmlns:lower = "http://zvon.org/lowercase" >
  <lower:ccc />
 </lower:bbb>
 <lower:BBB xmlns:lower = "http://zvon.org/uppercase" >
  <lower:CCC />
 </lower:BBB>
 <lower:x111 xmlns:lower = "http://zvon.org/xnumber" >
  <lower:x222 />
 </lower:x111>
</aaa>
```

- ```
<aaa >
 <bbb xmlns = "http://zvon.org/lowercase" >
  <ccc />
 </bbb>
 <BBB xmlns = "http://zvon.org/uppercase" >
  <CCC />
 </BBB>
 <x111 xmlns = "http://zvon.org/xnumber" >
  <x222 />
 </x111>
</aaa>
```

- ```
<aaa xmlns:upper = "http://zvon.org/uppercase" xmlns:xnumber = "http://zvon.org/xnumber"
>
 <bbb xmlns = "http://zvon.org/lowercase" >
  <ccc />
  <upper:WWW />
  <xnumber:x666 />
 </bbb>
 <BBB xmlns = "http://zvon.org/uppercase" >
  <upper:WWW />
  <xnumber:x666 />
  <CCC />
 </BBB>
 <x111 xmlns = "http://zvon.org/xnumber" >
  <x222 />
  <upper:WWW />
  <xnumber:x666 />
 </x111>
</aaa>
```

- ```
<aaa xmlns = "http://zvon.org/lowercase" >
 <bbb >
  <ccc xmlns = "" >
   <ddd />
  </ccc>
 </bbb>
</aaa>
```

-

EX2:Schéma-Déclarations simples

Objectif: Mise en jambe

Q : Qu'est ce? (à commenter uniquement).

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" >
  <xsd:element name="root" type="xsd:integer"/>
</xsd:schema>
```

Q : Qu'est ce? (idém).

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" >
  <xsd:element name="root">
   <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
     <xsd:maxExclusive value="25"/>
    </xsd:restriction>
   </xsd:simpleType>
  </xsd:element>
</xsd:schema>
```

Q : Qu'est ce? (idém).

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" >
  <xsd:element name="root">
   <xsd:simpleType>
    <xsd:union>
     <xsd:simpleType>
      <xsd:restriction base="xsd:integer">
       <xsd:minInclusive value="0"/>
       <xsd:maxInclusive value="100"/>
      </xsd:restriction>
     </xsd:simpleType>
     <xsd:simpleType>
      <xsd:restriction base="xsd:integer">
       <xsd:minInclusive value="300"/>
       <xsd:maxInclusive value="400"/>
      </xsd:restriction>
     </xsd:simpleType>
    </xsd:union>
   </xsd:simpleType>
  </xsd:element>
</xsd:schema>
```

Q : Écrivez un XML Schéma qui donne un modèle de document XML ne contenant qu'un élément "AAA". Cet élément ne contient que du texte.

Q : Nous voulons un élément racine "AAA" ne contenant qu'un élément "BBB" et qu'un élément "CCC" (ces deux éléments sont de type string, par exemple). L'ordre d'apparition n'est pas important. On utilisera le pattern (patron?) "all".

Q : Nous voulons un élément racine "AAA" ne contenant qu'un élément "BBB" suivis d'un élément

"CCC"(ces deux éléments sont de type string, par exemple). L'ordre d'apparition est important. Utilisez le pattern (patron?) "sequence"; ses attributs ne sont pas nécessaires, leur valeurs par défaut étant 1.

Q : Ici, l'élément "AAA" contient un nombre quelconque d'éléments "BBB" et "CCC" (peut être zéro). On utilisera le pattern "sequence" avec les bonnes valeurs d'attributs. Le but ici est de voir la fonction des attributs "minOccurs" du pattern "sequence" et ceux des éléments fils (inclus dans la séquence). Le document suivant est valide:

```
<AAA xsi:noNamespaceSchemaLocation="correct_0.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
  <BBB>111</BBB>
  <CCC>YYY</CCC>
  <BBB>222</BBB>
  <BBB>333</BBB>
  <CCC>ZZZ</CCC>
</AAA>
```

Q : Désormais, l'élément "AAA" contient soit "BBB", soit "CCC". On utilisera le pattern "choice".

Q : Nous voulons un élément racine "root" qui ne contient que du texte et un seul attribut ("xx")

Q : Écrivez un XML Schéma qui donne un modèle de document XML ne contenant qu'un élément "AAA". Cet élément racine ne contient que du texte...qui est l'élément par défaut...

Q : Nous voulons maintenant que cet élément racine "AAA" puisse contenir à la fois du texte et un élément "BBB". Pour cela, il faut fixer l'attribut "mixed" de "complexType" à "vrai".

EX3: Schéma-Déclarations de types

Objectif: Le but est d'écrire un schéma W3C pour les documents recettes de cuisine, comme par exemple le fichier egg.xml

Q : Voici le document egg.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<recette
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="cook.xsd">
  <nom>Oeufs à la coque</nom>
  <portion>4</portion>
  <preparation value="1" unit="min"/>
  <cuisson value="3" unit="min"/>
  <ingredients>
   <ing><nom id="ing1">oeufs</nom><nbre>4</nbre></ing>
  </ingredients>
  <guide>
   <block>Plongez délicatement les <renvoi ref="ing1">oeufs</renvoi>
   dans de l'eau portée à ébullition. </block><block>Faîtes reprendre l'ébullition,
   puis baissez le feu et laissez cuire à petits bouillons pendant
   <cuisson value="3" unit="min"/> minutes. </block><block>Égouttez les
   <renvoi ref="ing1">oeufs</renvoi>, posez-les dans des coquetiers et coupez
   hortizontalement la partie supérieure pour les déguster.</block>
```

```
 </guide>
</recette>
```

- Écrivez le modèle de l'élément "recette". Il s'agit d'un élément de type complexe constitué d'une séquence d'éléments "nom", "portion", "préparation", "cuisson", "repos", "ingrédients" et "guide". Remarque: il peut ne pas y avoir de temps de repos ou de temps de cuisson.
- Écrivez les modèles des éléments "nom" et "portion". Ensuite, écrivez les modèles des éléments "préparation", "cuisson" et "repos" en utilisant le type "uniteTempsType" (Déclaré plus loin).
- Écrivez le modèle de l'élément "ingrédients": il s'agit d'une séquence non limitée d'éléments "ing" (pour ingrédient). Les éléments "ing" sont de type "iType". De même, on demande d'écrire le modèle de l'élément "guide".
- Déclarez le type "blockType". Cet élément complexe peut contenir une série non ordonnée d'éléments "renvoi" (de type "renvoiType"), "repos" (de type "uniteTempsType"), "strong" (de type "string") et "cuisson" (de type "uniteTempsType").
- Déclarez le type "renvoiType". Il s'agit d'un type dérivé du type de base "string" auquel on associe un attribut "ref", requis, de type "string". Remarquez que "renvoiType" est un type complexe mais qu'il ne contient pas d'éléments fils. Par conséquent, son contenu doit être inscrit dans un élément "xsd:simpleContent"
- Déclarez le type "uniteTempsType". On lui associe deux attributs ("value" et "unit"). Le type "uniteTempsType" est il simple ou complexe? L'attribut "value" est de type "int" (type de base) mais le type de l'attribut "unit" est une restriction du type de base de "string" (les valeurs de cet attributs sont limitées à "minute" et "heure").
- Déclarez le type "iType". Il s'agit d'un type complexe puisqu'un élément de ce type contient des éléments ("nom" et, au choix "nbre" et "poids"). L'élément "nom", fils d'un élément de type "iType", ne doit pas être confondu avec l'élément "nom" de "recette". C'est pourquoi on décide ici de préciser son type à l'intérieur même de la déclaration de type de "iType". Remarque: "nom" est de type complexe car on lui associe un attribut. Cependant, il ne contient pas d'éléments fils... on se rapportera à la déclaration de "blockType" pour gérer ce phénomène.
- Enfin, déclarez le type "poidsType". Il s'agit d'un type complexe dérivant du type de base simple "string" (Il est donc simpleContent). On lui associe un attribut "unit" de type (simple, bien évidement) dérivant de "string" et ne pouvant prendre que les valeurs "gramme" ou "kilogramme".

Q : Quelle est la forme du fichier Schéma W3C final?
