# Formalisation des Données
# Technologies XML
# Technologies XML
# TD

L3 MIAGE

Université Grenoble Alpes
Celine.Fouard@univ-grenoble-alpes.fr
Nicolas.Glade@univ-grenoble-alpes.fr

Ce cours a été rédigé par Céline Fouard et Nicolas Glade.
Il est très largement inspiré du cours d'Emmanuel Promayon, PhD donné à Polytech'Grenoble
Certains chapitres sont inspirés de *XML Cours et exercices*, Alexandre Brillant, éditions Eyrolles et *XSLT, Mastering XML Transformations*, Doug Tidwell, O'Reilly editions.
Les références des exercices sont donnés dans le texte.
Si vous souhaitez utiliser ce document, merci de contacter Celine.Fouard@univ-grenoble-alpes.fr ou Nicolas.Glade@univ-grenoble-alpes.fr

# Table des matières

*Chapitre II*

# XML

## 1 Points Clés du XML♣

*Question 1.1*
Citer les 8 points clés du XML

## 2 Calendrier♣

On considère le document **XML** suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<week>
      <date>20 oct</date>
      <tutorial>Algorithmique : statistiques</tutorial>
      <lecture>Boucles et conditions en Java</Lecture>
</week>

<holidays>
      <!-- Toussaint -->
      <date>27 oct</date>
</holidays>

<week>
      <date>3 nov</date>
      <tutorial>Algorithmique : tableaux</tutorial>
      <Lecture>Les tableaux N dimensions</lecture>
</week>
```

*Question 2.1*
Ce document est-il bien formé ?

*Question 2.2*
Quels sont les éléments qui possèdent des sous-éléments ?

*Question 2.3*
Représentez le document (éventuellement corrigé) sous forme d'arbre.

## 3 Bien formé or not ?

Dans les documents XML suivants, on considérera que le prologue a bien été déclarée de la manière suivante:

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

La question est la même pour tous les extraits suivants: le document ci-dessous est-il bien formé ?

1.
```xml
  <text>Ceci est un <doctype>document XML</doctype> </text>
```

2.
```xml
  Ceci est un document XML
```

3.
```xml
<item>Item 1</item>
<item>Item 2</item>
<item>Item 3</item>
```

4.
```xml
<list>
<item>Voiture</item>
<ITEM>Avion</ITEM>
<Item>Train</Item>
</list>
```

5.
```xml
<list>
<item>Voiture</itm>
<item>Avion</ITEM>
<item>Train</item>
</list>
```

6.
```xml
<book>
<chapter> <title> Introduction </title> </chapter>
<chapter>
<title> Récit </title>
<subChapter> <title> Partie 1 </title> </subChapter>
<subChapter> <title> Partie 2 </title> </subChapter>
</chapter>
<chapter> <title> Index </title> </chapter>
</book>
```

7.
```xml
<text> <bold><italic>XML</bold></italic> </text>
```

8.
```xml
<description>
Il y a des pommes <color>jaunes<color> et <color>rouges</color>.
</description>
```

9.
```xml
<listOfTags> <AAA></AAA> <BBB></BBB> <CCC/> <DDD/> </listOfTags>
```

10.
```xml
<permittedNames>
        <name/>
        <xsl:copy-of/>
        <A_long_element_name/>
        <A.name.separated.with.full.stops/>
        <a123323123-231-231/>
        <_12/>
</permittedNames>
```

11.
```
<elements>
        <A;name/>
        <last@name>
        <@#\$%^()%+?=/>
        <A*2/>
        <1ex/>
</elements>
```

```
<forbiddenNames>
        <xmlTag/>
        <XMLTag/>
        <XmLTag/>
        <xMlTag/>
        <xmLTag/>
</forbiddenNames>
```

13.
```
<elements-with-attributes>
        <el _ok = "oui" />
        <one attr = "une valeur"/>
        <several first="1" second = '2' third= "333"/>
        <apos_quote case1="Aujourd'hui" case2='Il lança : "Salut, tout le
            monde!" '/>
</elements-with-attributes>
```

```
<errors>
        <wrong_char a*b = "23432"/>
        <mismatched_separator value = "12'/>
        <wrong_separator_type value="aa"aa"/>
        <wrong_separator_type value='bb'bb'/>
        <wrong_start XML-ID = "xml234"/>
</errors>
```

15.
```
<example>
        <isLower> 23 < 46 </isLower>
        <ampersand> Dupond & fils </ampersand>
</example>
```

```
<example>
        <isLower> 23 &lt; 46 </isLower>
        <ampersand> Dupond &amp; fils </ampersand>
</example>
```

17.
```
<example>
        <right-bracket> A la fois > et &gt; sont autorisés</right-bracket>
        <double-quote> A la fois " et &quot; sont autorisés</double-quote>
        <apostrophe> A la fois ' et &apos; sont autorisés</apostrophe>
        Cela est utile dans : <el value=" &apos; &quot; &apos; "/>
</example>
```

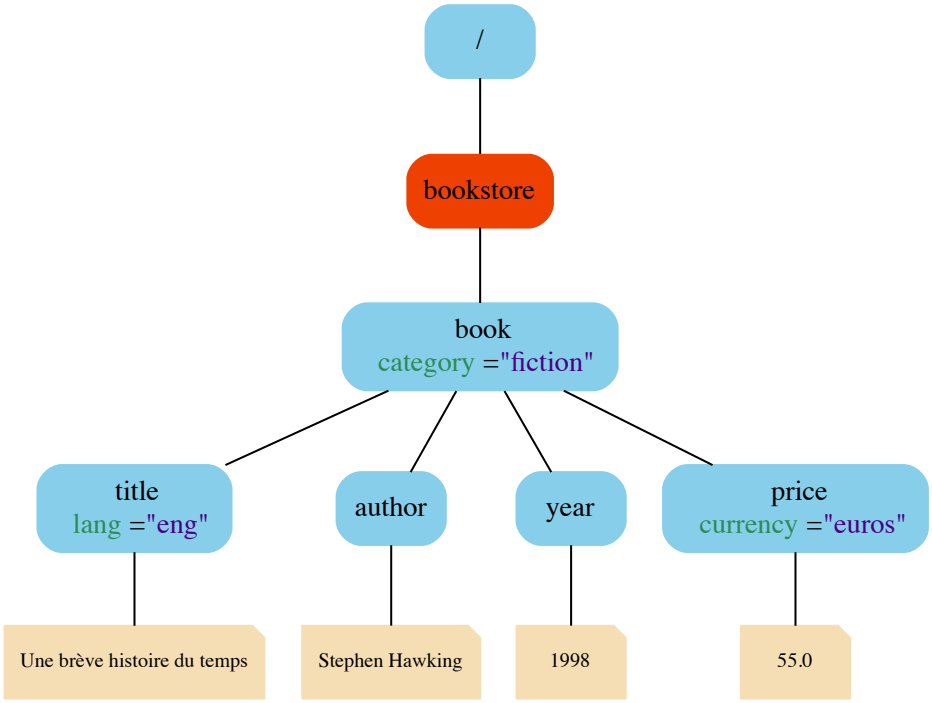
```
    <!-- doc A -->
    <example>
    <!-- <HEAD> -->
    <!-- Caractères <&< -->
    </example>
```

19.
```
<example> <!-- A -- B --> </example>
```

```
<example> <![CDATA[ <aaa>bb&cc<<<  ]]> </example>
```

## 4 Bookstore

On s'intéresse à l'arbre de la figure II.1 qui décrit une hiérarchie et des éléments:




20.

Figure II.1: Arbre représentant un magasin de livres.

*Question 4.1*
Transformez cet arbre en document XML.

*Question 4.2*
Créez une autre instance XML correspondant au même modèle (schéma).

## 5 Héritage

Soit le document suivant :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<aa>
    <bb>
        <cc>Introduction</cc>
    </bb>
    <bb>
        <cc>Cinquième promenade</cc>
        <dd>Première partie</dd>
        <dd/>
    </bb>
    <bb>
        <cc>Index</cc>
    </bb>
</aa>
```

*Question 5.1*
Ce document est-il bien formé ?

*Question 5.2*
Quel est l'élément qui possède le plus grand nombre de fils ?

*Question 5.3*
Représentez le document (éventuellement corrigé) sous forme d'arbre.

# 6 Cinématographie

On considère une filmothèque composée des éléments suivants:

- Premier film:
  Il s'appele Rocky, il est en Anglais, sa photo se trouve dans le fichier: rocky.jpg. Ce film est sorti durant l'année 1976, réalisé par John G. Avildsen. Les personnages pricipaux de ce film sont: Sylvester Stallone qui joue Rocky Balboa ; Talia Shire qui joue Adrian ; Burt Young qui joue Paulie ; Carl Weathers qui joue Apollo Creed ; Burgess Meredith qui joue Mickey.
- Deuxième film:
  Il s'appelle La Guerre des étoiles, il est en Francais, sa photo n'est pas disponible. Ce film est sorti durant l'année 1977, du réalisateur George Lucas. Les personnages pricipaux de ce film sont: Mark Hamill qui joue Luke Skywalker; Harrison Ford qui joue Han Solo; Carrie Fisher qui joue La princesse Leia.
- Troisième film:
  Il s'appelle Raiders of the Lost Ark, il est en Anglais, sa photo se trouve dans le fichier indiana.jpg. Ce film est sorti en 1981, par le réalisateur Steven Spielberg. Les personnages pricipaux de ce film sont: Harrison Ford qui joue Indiana Jones.

*Question 6.1*
Créez un document XML qui reprend toutes les informations ci-dessus.

*Question 6.2*
Dessinez l'arbre de données correspondant.

# 7 Organisation de la World Compagny

La société *World Company* possède un conseil d'administration et un comité exécutif.

*Question 7.1*
Répartissez vous en 4 équipes. Ecrivez un document xml bien formé qui structure l'information suivante:

- **équipes 1 et 3:** *Le conseil d'administration est composé de Jean-Charles Nariou, Président-Directeur Général, de Didier Larcier, représentant de la filliale Europe du groupe, de Henri Etang, administrateur indépendant, Madame Sylvia Jya, administratrice indépendant, de Marc de Larriachère.*
- **équipes 2 et 4:** *Le comité exécutif est composé de Jean-Charles Nariou, PDG, d'Yves Banbrait, représentant de la filliale Asie du groupe, de Yves Desmoulins, directeur des ressources humaines, d'Antonella Etang, directrice financière, d'Arnaud Straber et André Luczas. Sa secrétaire est Mlle Julie Epositos.*

*Question 7.2*
Représentez vos documents XML sous forme d'arbre.

*Chapitre III*

# XPath

## 1 Carnet d'adresses♣

On considère le document XML suivant:

```
<?xml version="1.0" encoding="UTF8"?>
<addressBook>
        <address>
                <firstName>John</firstName>
                <surname>Smith</surname>
                <email>smithj@world.org</email>
                <tel type="work">234-123-222</tel>
        </address>
        <address>
                <firstName>Alice</firstName>
                <surname>Brown</surname>
                <email>Alice.Brown@europe.com</email>
                <tel type="home">22-33-444</tel>
                <tel type="work">11-43-222</tel>
        </address>
        <address>
                <firstName>George</firstName>
                <surname>White</surname>
                <email>gw@rock.com</email>
        </address>
</addressBook>
```

*Question 1.1*
Donner une expression XPath qui permet de sélectionner la racine du document

*Question 1.2*
Donner 2 expressions absolues XPath qui permettent de sélectionner les nœud `adresse`.

*Question 1.3*
Donner 3 expressions absolues XPath qui permettent de sélectionner le nœud `<firstName>Alice</firstName>`

.

*Question 1.4*
Que sélectionne le path
`//email[text()='gw@rock.com']/../firstName/following-sibling::node()` ?

## 2 Des livres et XPath

On considère le document XML suivant:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<edition nom="J'ai lu">
    <!-- Les auteurs -->
    <auteur no="a01" nom="Stephen King" pays="usa"/>
    <auteur no="a02" nom="Isaac Asimov" pays="russie"/>
    <!-- Les ouvrages -->
    <livre annee="1996" reference="15157">
        <titre>La ligne verte</titre>
        <ref-auteur ref="a01"/>
    </livre>
    <livre annee="1983" reference="12266">
        <titre>Simetierre</titre>
        <ref-auteur ref="a01"/>
    </livre>
    <livre annee="1964" reference="1542">
        <titre>Un défilé de robots</titre>
        <ref-auteur ref="a02"/>
    </livre>
    <livre annee="1950" reference="1453">
        <titre>I, robot</titre>
        <ref-auteur ref="a02"/>
    </livre>
    <!-- ... -->
</edition>
```

*Question 2.1*
Dessiner l'arbre XML de ce document.

*Question 2.2*
Donner les expressions XPath qui référencent tous les auteurs.

*Question 2.3*
Donner les expressions XPath qui référencent le(s) auteur(s) américains.

*Question 2.4*
Donner les expressions XPath qui référencent le deuxième auteur.

*Question 2.5*
Donner les expressions XPath qui référencent les éléments titres des livres qui contiennent la chaîne `robot`.

*Question 2.6*
Donner les expressions XPath qui référencent le(s) titre(s) de(s) ouvrage(s) ayant été publiés en 1996.

*Question 2.7*
Donner les expressions XPath qui référencent le(s) auteurs ayant publiés en 1996.

*Question 2.8*
Donner les expressions XPath qui référencent les titres des ouvrages d'Isaac Asimov.

## 3 Des BDs et XPath

On considère le document XML suivant:

```xml
<?xml version="1.0" ?>
<bdtheque>
    <personnes>
        <personne id="AScotch">
            <nom>Arleston</nom>
            <prenom>Scotch</prenom>
        </personne>
        <personne id="DTarquin">
            <nom>Tarquin</nom>
            <prenom>Didier</prenom>
        </personne>
        <personne id="RSattouf">
            <nom>Sattouf</nom>
            <prenom>Riad</prenom>
        </personne>
        <personne id="Larcenet">
            <nom>Larcenet</nom>
            <prenom>Manu</prenom>
        </personne>
        <personne id="JYFerri">
            <nom>Ferri</nom>
            <prenom>Jean-Yves</prenom>
        </personne>
        <personne id="Petillon">
            <nom>Pétillon</nom>
        </personne>
    </personnes>

    <collection>
        <!-- Retour au collège de Riad Sattouf -->
        <bd>
            <titre>Retour au collège</titre>
            <auteur ref="RSattouf" />
            <illustrateur ref="RSattouf" />
            <sortie>2005</sortie>
            <resume>
                Avec Retour au collège, Riad Sattouf a eu une grande idée de
                BD-reportage : investir une classe de troisième dans un collège huppé
                de Paris. Se présentant comme un écrivain artiste-peintre (ça fait
                toujours mieux que dessinateur BD !) envoyé par le ministère de
                l'éducation nationale, Sattouf arrive à rejoindre pour quinze jours
                une classe de troisième au tempérament explosif. Avec ce séjour, il
                poursuit ses vieux démons, issus de la pire période de la jeunesse :
                la préadolescence. Les noms et physionomies des personnages ont été
                modifiés. Par contre, les situations et les propos rapportés sont
                absolument véridiques, peut-on lire dans les premières pages de
                l'ouvrage. Vu la façon dont Sattouf a croqué les différents profils de
                la classe, les élèves de troisième C, malgré les changements de prénom
                et de physionomie, n'auront sans doute aucun mal à se reconnaître.
            </resume>
        </bd>
        <!-- Lanfeust -->
        <serie>
```

```
54              <titreserie>Lanfeust de Troy</titreserie>
55              <bd numero="1" illustration="lanfeustdetroy1.png">
56                  <titre>L'ivoire du Magohamoth</titre>
57                  <auteur ref="AScotch" />
58                  <illustrateur ref="DTarquin" />
59                  <sortie>1994-10</sortie>
60                  <resume>
61                      Troy est un monde fantastique où toute personne possède
                            un pouvoir
62                      magique qui est plus ou moins utile... <perso>
                            Lanfeust</perso>,
63                      forgeron de son état découvre qu'il a non pas un seul
                            pouvoir mais
64                      tous, mais seulement lorsqu'il est en contact avec
                            un ivoire
65                      particulier. Il décide alors, sur le conseil du sage de
                            son village,
66                      de partir pour Eckmul afin de rencontrer le grand conseil
                            ...
67                  </resume>
68              </bd>
69              <bd numero="2">
70                  <titre>Thanos l'incongru</titre>
71                  <auteur ref="AScotch" />
72                  <illustrateur ref="DTarquin" />
73                  <sortie>1995-08</sortie>
74                  <resume>
75                      On retrouve notre fine équipe dans la capitale de Troy,
                            Eckmul, où
76                      elle apprend que <perso>Lanfeust</perso> n'est pas le seul
                            à posséder
77                      ce pouvoir absolu. Eh ! oui le méchant de
                            l'histoire
78                      (<perso>Thanos</perso>) a lui aussi cette capacité.
                            Mais ces deux
79                      personnages ne peuvent utiliser le pouvoir absolu que s'
                            ils sont en
80                      contact avec de l'ivoire du Magohamoth (monstre légendaire
                            et mythique
81                      source du pouvoir sur Troy). Et vous pensez bien que cet
                            ivoire ne se
82                      trouve pas dans toutes les bonnes épiceries. C'est
                            pourquoi il leur
83                      faudra récupérer l'épée d'un prince car son pommeau est
                            fait en ivoire
84                      du Magohamot
85                  </resume>
86              </bd>
87          </serie>
88          <!-- Le retour à la terre de Larcenet -->
89          <serie>
90              <titreserie>Le retour à la terre</titreserie>
91              <bd numero="1">
92                  <titre>La vraie vie</titre>
93                  <auteur ref="JYFerri" />
94                  <illustrateur ref="Larcenet" />
95                  <sortie>2002</sortie>
96                  <resume>
97                      <perso>Mariette</perso> et <perso>Manu</perso> en avaient
                            marre de la ville. Ils viennent de déménager
98                      à la campagne. L'air pur, les petits oiseaux, les champs
                            à perte de vue et tout ça...le bonheur,
99                      quoi ? Pas vraiment ! C'est qu'il n'est pas évident pour
```

```
                        deux citadins accoutumés  au bruit, à la
100                 promiscuité des transports en commun ou des embouteillages,
                        et à la proximité des commerces et lieux
101                 de délassement, de se désintoxiquer de tout cela.
102             </resume>
103         </bd>
104         <bd numero="2">
105             <titre>Les projets</titre>
106             <auteur ref="JYFerri" />
107             <illustrateur ref="Larcenet" />
108             <sortie>2003</sortie>
109             <resume>
110             </resume>
111         </bd>
112         <bd numero="3">
113             <titre>Le vaste monde</titre>
114             <auteur ref="JYFerri" />
115             <illustrateur ref="Larcenet" />
116             <sortie>2004</sortie>
117             <resume>
118             </resume>
119         </bd>
120     </serie>
121     <!-- L'enquête corse de Pétillon -->
122     <bd>
123         <titre>L'enquête corse</titre>
124         <auteur ref="Petillon" />
125         <illustrateur ref="Petillon" />
126         <sortie>2001</sortie>
127         <resume>
128             Une enquête  du détective  <perso>Jack Palmer</perso> en  Corse
                    ? Explosive,  forcément.  On
129             peut même dire que  Palmer est en plein boum. Trench-coat trop
                    grand, chapeau mou et regard
130             ahuri, il tente  désespérément de prendre langue avec  un
                   certain <perso>Ange Leoni</perso>.
131             Pas facile.  D'ailleurs, un des cafés de l'histoire  s'appelle
                   'café motus', et un autre
132             'café  omerta'.  Tout est  dit -  enfin,  façon de  parler.
                   Quand  on lit  Pétillon, il  est
133             difficile de respecter la loi du silence : on a plutôt tendance
                    à hurler de rire...
134         </resume>
135     </bd>
136   </collection>
137 </bdtheque>
```

*Question 3.1*
Dessiner l'arbre XML du sous arbre `personnes`.

*Question 3.2*
Donner les expressions XPath qui donnent le nombre d'artistes (auteur ou illustrateur).

*Question 3.3*
Donner une expression XPath qui référencent les titres des BDs dont l'auteur et l'illustrateur sont la même personne.

*Question 3.4*
Donner une expression XPath qui référencent le(s) titre(s) de(s) ouvrage(s) qui ont 2 `perso` dans leur résumé.

*Question 3.5*
Donner une expression XPath qui référence les noms des illustrateurs dont la BD a une illustration.

*Question 3.6*
Donner une expression XPath qui référencent les titres des séries qui ont exactement 2 bds.

*Question 3.7*
Donner une expression XPath qui retourne tous les ancètres communs à *"Didier Tarquin"* et *"Riad Sattouf"* (c'est-à-dire les noeuds de l'arbres qui ont à la fois *"Didier Tarquin"* et *"Riad Sattouf"* comme descendant).

*Question 3.8*
Donner une expression XPath qui compte le nombre d'éléments définis entre l'ouverture de l'élément `personne` correspondant à "Riad Sattouf" et l'ouverture de l'élément `personne` correspondant à "Petillon" dans l'ordre du document.

# 4 Des acteurs...

On considère le document XML suivant:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<liste-acteurs>
  <acteur id="clint">
    <prenom>Clint</prenom>
    <nom>Eastwood</nom>
    <naissance>31 Mai 1930</naissance>
    <nationalite>américaine</nationalite>
    <photo source="clint.jpg" alt="Clint Eastwood"/>
    <site url="http://www.clinteastwood.net/"/>
    <biographie>
      <p>
        Né d'un père comptable, le jeune Clinton mène avec ses parents
        une vie de nomade. Il passe son adolescence à Oakland et ne
        pense pas du tout à devenir acteur. Il fait des petits boulots
        sans grande conviction.  Puis il part à l'armée où il fait des
        rencontres décisives et obtient du travail chez Universal. Il
        fait sa première apparition en <annee>1955</annee> dans
        <film>La Revanche de la créature</film> puis enchaîne les
        petits rôles dans cinq films où personne ne le remarque
        véritablement.
      </p>
      <p>
        Avec le drame <film>Million dollar baby</film>, le cinéaste
        obtient une véritable consécration en remportant, douze ans
        après <film>Impitoyable</film>, l'Oscar du Meilleur film et du
        Meilleur réalisateur, ses comédiens <refacteur
        code="hilary">Hilary Swank</refacteur> et <refacteur
        code="morgan">Morgan Freeman</refacteur> repartant avec les
        statuettes de La Meilleure actrice et du Meilleur second rôle
        masculin.
      </p>
    </biographie>
  </acteur>
  <acteur id="hilary">
    <prenom>Hilary</prenom>
    <nom>Swank</nom>
    <naissance>30 Juillet 1974</naissance>
    <nationalite>américaine</nationalite>
    <photo source="hilaryswank.jpg" alt="Hilary Swank"/>
    <site url="http://www.hilaryswankfan.com/"/>
    <biographie>
```

```
      <p>
        Championne de natation, Hilary Swank s'oriente rapidement vers
        le métier d'actrice.  A l'âge de dix-huit ans, elle fait une
        courte apparition dans <film>Buffy, tueuse de vampires</film>
        en <annee>1992</annee>, une comédie fantastique de Fran Rubel
        Kuzui, et décroche en <annee>1994</annee> le rôle-titre de
        <film>Miss Karaté Kid</film> de Christopher Cain.
      </p>
      <p>
        C'est <film>Boys don't cry</film>, un drame de
        <realisateur>Kimberly Peirce</realisateur>, qui la révèle
        véritablement au grand public en <annee>2000</annee>.  Sa
        prestation du travesti Brandon Teena lui vaut l'Oscar et le
        Golden Globe de la Meilleure actrice, les prix
        d'interprétation des New York Film Critics, Los Angeles Film
        Critics et Chicago Film Critics, ainsi que le Broadcast Film
        Critics.
      </p>
      <p>
        En <annee>2005</annee>, elle rafle pour la deuxième fois (et
        en seulement deux nominations) l'Oscar de la Meilleure actrice
        grâce à sa performance de boxeuse surentraînée par <refacteur
        code="clint">Clint Eastwood</refacteur> dans <film>Million
        dollar baby</film>.
      </p>
    </biographie>
  </acteur>
  <acteur id="morgan">
    <prenom>Morgan</prenom>
    <nom>Freeman</nom>
    <naissance>1 Juin 1937</naissance>
    <nationalite>américaine</nationalite>
    <photo source="freeman.jpg" alt="Morgan Freeman"/>
    <biographie>
      <p>
        Morgan Freeman est diplômé du lycée de Greenwood, dans le
        Mississippi.  A dix-huit ans, il s'engage dans l'Air Force et,
        une fois ses obligations militaires accomplies, s'installe en
        Californie pour étudier la danse et l'art dramatique au Los
        Angeles City College. C'est à Broadway qu'il fait ses débuts
        de comédien en <annee>1967</annee>.
      </p>
      <p>
        A 68 ans, il obtient enfin la reconnaissance de la profession
        en remportant l'Oscar du Meilleur second rôle masculin pour sa
        prestation d'ancien boxeur borgne dans <film>Million dollar
        baby</film> (<annee>2005</annee>) de son fidèle ami <refacteur
        code="clint">Clint Eastwood</refacteur>.
      </p>
    </biographie>
  </acteur>
</liste-acteurs>
```

*Question 4.1*
Donner les expressions XPath déterminant:

1. les éléments `acteur`
2. le nombre d'acteurs décrits dans ce document
3. tous les noeuds `film`
4. la liste des films nommés dans la biographie de Hilary Swant

5. le premier noeud fils du premier noeud acteur
6. le premier noeud fils du deuxième noeud acteur
7. le dernier élément fils de chaque acteur
8. l'acteur dont l'identifiant est `clint`
9. l'acteur dont le prénom est `Clint`
10. les noms des acteurs dont la biographie contient exactement deux paragraphes.

# 5 Des films...

On considère à présent le document suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<films>
  <film lang="en">
    <photo href="rocky.jpg"/>
    <titre>Rocky</titre>
    <annee>1976</annee>
    <realisateur>John G. Avildsen</realisateur>
    <casting>
      <acteur id="rocky" personnage="Rocky Balboa">Sylvester Stallone</acteur>
      <acteur id="adrian" personnage="Adrian">Talia Shire</acteur>
      <acteur id="paulie" personnage="Paulie">Burt Young</acteur>
      <acteur id="creed" personnage="Apollo Creed">Carl Weathers</acteur>
      <acteur id="mickey" personnage="Mickey">Burgess Meredith</acteur>
    </casting>
    <synopsis>
      <perso ref="rocky"/> is a small-time boxer who lives in an
      apartment in Philadelphia, Pennsylvania, and his career has so
      far not gotten off the canvas. <perso ref="rocky"/> earns a
      living by collecting debts for a loan shark named Gazzo, but
      Gazzo doesn't think <perso ref="rocky"/> has the viciousness it
      takes to beat up deadbeats. <perso ref="rocky"/> still boxes
      every once in a while to keep his boxing skills sharp, and his
      ex-trainer, <perso ref="mickey"/>, believes he could've made it
      to the top if he was willing to work for it. <perso
      ref="rocky"/>, goes to a pet store that sells pet supplies, and
      this is where he meets a young woman named <perso
      ref="adrian"/>, who is extremely shy, with no ability to talk to
      men. <perso ref="rocky"/> befriends her. Adrain later surprised
      <perso ref="rocky"/> with a dog from the pet shop that <perso
      ref="rocky"/> had befriended.  <perso ref="adrian"/>'s brother
      Paulie, who works for a meat packing company, is thrilled that
      someone has become interested in <perso ref="adrian"/>, and
      <perso ref="adrian"/> spends Thanksgiving with <perso
      ref="rocky"/>. Later, they go to <perso ref="rocky"/>'s
      apartment, where <perso ref="adrian"/> explains that she has
      never been in a man's apartment before.  <perso ref="rocky"/>
      sets her mind at ease, and they become lovers.  Current world
      heavyweight boxing champion <perso ref="creed"/> comes up with
      the idea of giving an unknown a shot at the title.  <perso
      ref="creed"/> checks out the Philadelphia boxing scene, and
      chooses <perso ref="rocky"/>. Fight promoter Jergens gets things
      in gear, and <perso ref="rocky"/> starts training with <perso
      ref="mickey"/>. After a lot of training, <perso ref="rocky"/> is
      ready for the match, and he wants to prove that he can go the
      distance with <perso ref="creed"/>.
    </synopsis>
  </film>
  <film lang="fr">
```

```
    <titre>La Guerre des étoiles</titre>
    <annee>1977</annee>
    <realisateur>George Lucas</realisateur>
    <casting>
      <acteur id="lukemonfils" personnage="Luke Skywalker">Mark Hamill</acteur>
      <acteur personnage="Han Solo">Harrison Ford</acteur>
      <acteur id="leia" personnage="La princesse Leia">Carrie Fisher</acteur>
    </casting>
    <synopsis>
      Il y a bien longtemps, dans une galaxie très lointaine... La
      guerre civile fait rage entre l'Empire galactique et l'Alliance
      rebelle.  Capturée par les troupes de choc de l'Empereur menées
      par le sombre et impitoyable Dark Vador, la princesse <perso
      ref="leia"/> dissimule les plans de l'Etoile Noire, une station
      spatiale invulnérable, à son droide R2-D2 avec pour mission de
      les remettre au Jedi Obi-Wan Kenobi. Accompagné de son fidèle
      compagnon, le droide de protocole C-3PO, R2-D2 s'échoue sur la
      planète Tatooine et termine sa quête chez le jeune <perso
      ref="lukemonfils"/>.  Rêvant de devenir pilote mais confiné aux
      travaux de la ferme, ce dernier se lance à la recherche de ce
      mystérieux Obi-Wan Kenobi, devenu ermite au coeur des montagnes
      désertiques de Tatooine...
    </synopsis>
  </film>
  <film lang="en">
    <titre>Raiders of the Lost Ark</titre>
    <annee>1981</annee>
    <realisateur>Steven Spielberg</realisateur>
    <casting>
      <acteur id="indy" personnage="Indiana Jones">Harrison Ford</acteur>
    </casting>
    <synopsis>
      Renowned archeologist and expert in the occult, <perso
      ref="indy"/>, is hired by the U.S.  Government to find the Ark
      of the Covenant, which is believed to still hold the ten
      commandments. Unfortunately, agents of Hitler are also after the
      Ark. <perso ref="indy"/>, and his ex-flame Marion, escape from
      various close scrapes in a quest that takes them from Nepal to
      Cairo.
    </synopsis>
  </film>
  <film lang="fr">
    <titre>Wallace et Gromit le mystère du lapin-garou</titre>
    <annee>2003</annee>
    <realisateur>Nick Park, Steve Box</realisateur>
    <casting/>
    <synopsis>
      Une "fièvre végétarienne" intense règne dans la petite ville de
      Wallace et Gromit, et l'ingénieux duo a mis à profit cet
      engouement en inventant un produit anti-nuisibles humain et
      écolo, qui épargne la vie des lapins. L'astuce consiste
      simplement à capturer, à la main, un maximum de ces rongeurs et
      à les mettre en cage.  A quelques jours du Grand Concours Annuel
      de Légumes, les affaires de Wallace et Gromit n'ont jamais été
      aussi florissantes, et tout irait pour le mieux dans le meilleur
      des mondes, si un lapin-garou géant ne venait soudain s'attaquer
      aux sacro-saints potagers de la ville.  Pour faire face à ce
      péril inédit, l'organisatrice du concours, Lady Tottington, se
      tourne vers nos deux "spécialistes" et leur demande
      d'appréhender le monstre.
    </synopsis>
  </film>
</films>
```

*Question 5.1*
Donner les expressions XPath qui déterminent:

1. le nombre de films disponibles
2. les acteurs du deuxième film
3. les titres des films dont la fiche est en anglais
4. le nom du personnage dont l'identifiant est `lukemonfils`
5. le titre des films sortis en 1981
6. le nombre de références faites au personnage dont l'identifiant est `indy`
7. les films sans acteur
8. les films avec plusieurs réalisateurs
9. les titres des films dont le synopsis contient le mot `believe`

*Chapitre IV*

# *Modélisation de données*

## 1 Année Universitaire♣

On considère le document XML suivant:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<etudiant>
    <nom>Kuzbidon</nom>
    <prénom>Alex</prénom>
    <dateNaissance>1991-06-15</dateNaissance>
    <numéroTéléphone> 04 56 52 00 12</numéroTéléphone>
    <l3Validée>true</l3Validée>
    <ue nom="LW">
        <note>18.5</note>
        <validée>true</validée>
    </ue>
    <ue nom="BD">
        <note>15.0</note>
        <validée>true</validée>
    </ue>
    <ue nom="Anglais">
        <note>05.0</note>
        <validée>false</validée>
    </ue>
</etudiant>
```

*Question 1.1*
Créez un diagramme UML et un Schema XML correspondant, permettant au mieux de normaliser ces données.

## 2 Utilisation des cardinalités ♣

Soit le diagramme UML suivant:

*Question 2.1*
Créez un schéma permettant de représenter ces données.

*Question 2.2*
Créez un exemple de document XML associés.

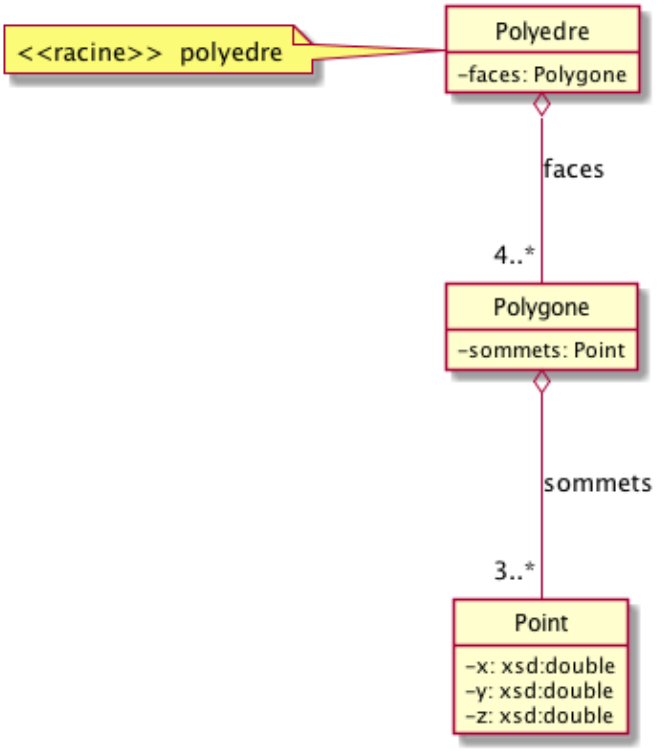



Figure IV.1: Représentation UML d'un polyèdre

## 3 Météorologie

On considère le diagramme UML suivant:

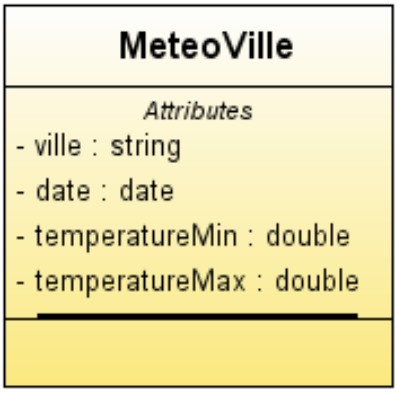



Figure IV.2: Diagramme UML de relevés météorologiques.

*Question 3.1*
Créez un schéma permettant de représenter ces données et deux exemples de documents XML associés.

On considère à présent ce diagramme UML un peu plus complexe

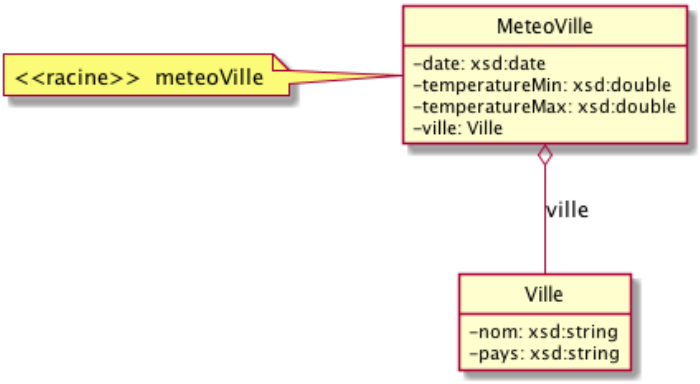



Figure IV.3: Diagramme UML de relevés météorologiques.

*Question 3.2*
Créez un schema permettant de représenter ces données et deux exemples de documents XML associés.

# 4 Championnat

On considère le code XML suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<championnat DIVISION="1" SAISON="2003-2004">
    <journée NUMERO="1" DATE="01/08/2003">
        <rencontre domicile="Auxerre" extérieur="Nice" scoreD="1" scoreE="2"
            rencontre/>
        <rencontre domicile="Guingamp" extérieur="Marseille" scoreD="0" scoreE=
            "1"/>
        <rencontre domicile="Lens" extérieur="LeMans" scoreD="0" scoreE="0"/></
            rencontre>
        <rencontre domicile="Lille" extérieur="Lyon" scoreD="1" scoreE="0"/>
        <rencontre domicile="Metz" extérieur="Ajaccio" scoreE="0" scoreE="1"/>
        <rencontre domicile="Monaco" extérieur="Bordeaux" scoreD="2" scoreE="0"
            />
        <rencontre domicile="Montpellier" extérieur="Rennes" scoreD="1" scoreE=
            "1">
        <rencontre domicile="ParisSG" extérieur="Bastia" scoreD="0" scoreE="0"/
            >
        <rencontre domicile="Sochaux" extérieur="Nantes" scoreD="2" scoreE="1"/
            >
        <rencontre domicile="Toulouse" extérieur="Strasbourg" SCOED="1" scoreE=
            "1"/>

    <journée NUMERO="2" DATE=08/08/2003>
        <rencontre domicile="Bastia" extérieur="Metz" scoreD="0" scoreE="2"/>
        <rencontre domicile="Bordeaux" extérieur="Montpellier" scoreD="0"
            scoreE="1"/>
        <rencontre domicile="LeMans" extérieur="Ajaccio" scoreD="0" scoreE="1"/
            >
        <rencontre domicile="Lille" extérieur="ParisSG" scoreD="1" scoreE="0"/>
        <rencontre domicile="Lyon" EXTERIEUR="Monaco" scoreD="3" scoreE="1"/>
        <rencontre domicile="Marseille" extérieur="Auxerre" scoreD="1" scoreE="
            0"/>
        <rencontre domicile="Nantes" extérieur="Lens" scoreD="2" scoreE="0"/>
        <rencontre domicile="Nice" extérieur="Sochaux" scoreD="1" scoreE="0"/>
        <rencontre domicile="Rennes" extérieur="Toulouse" scoreD="1" scoreE="0"
            />
        <rencontre domicile="Strasbourg" extérieur="Guingamp" scoreD="2" scoreE
            ="0"/>
    </journee>
<championnat NUMERO ="2"/>
```

*Question 4.1*
Ce document est-il bien formé ? Si ce n'est pas le cas, corrigez le.

*Question 4.2*
Transformez tous les attributs en sous-éléments.

*Question 4.3*
Représentez l'instance XML en arbre XML (3 rencontres de la première journée).

*Question 4.4*
Représentez le diagramme UML du schéma correspondant.

# 5 La classe !

Soit le diagramme UML suivant:

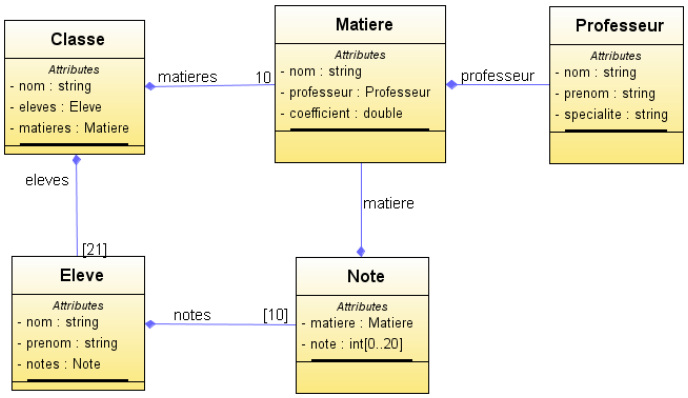



Figure IV.4: Diagramme UML pour des classes

*Question 5.1*
Créez un schema permettant de représenter ces données et un exemple de documents XML associés.

Construisez le schema petit à petit.

# 6 Les nuages...

Il existe diverses variétés de nuages. La plupart de ceux dont nous allons parler ne produit aucun "hydrométéore", sauf le cumulonimbus, qui est accompagné d'averses (parfois sous la forme de neige, de grésil ou de grêle).

L'altocumulus et le cirrocumulus partagent les mêmes "espèces" : lenticularis, stratiformis, castellanus et flocus. On retrouve ces deux espèces également chez le cirrus, ainsi que les espèces spissatus, uncinus et fibratus. Les espèces stratiformis, lenticularis et castellanus sont quant à elles partagées également avec les strato-cumulus.

Ces derniers peuvent se traîner au ras du sol et monter à 2000m, mais certains nuages ont une altitude minimale à peine plus élevée, puisqu'elle n'est que de 200m pour les cumulus, et de 300m pour les cumulonimbus. Il est vrai que ces derniers compensent en montant jusqu'à une altitude maximale de 18000m, soit plus haut encore que les cirrus, qui plafonnent à 12000m. L'altitude minimale de ces derniers coÔncide avec la fin de la présence possible des altocumulus, à 6000m. Et c'est autour de cette zone, entre 5000 et 7000m, que se trouvent les cirrocumulus. L'altitude minimale des altocumulus est de 2000m, soit quatre fois moins que l'altitude maximale des cumulus.

Ces pauvres cumulus ne sont pas favorisés en nom d'espèces, puisqu'ils se trouvent affligés de noms tels que fractus, mediocris, humilis et congestus... alors que les cumulonimbus ont des espèces aux noms plus... capillaires tels que calvus, capillatus. Les très gros cumulonimbus sont appelés mammatus.

*Question 6.1*
Réorganiser les informations *en vrac* du texte ci-dessus et écrire un document XML qui reprend toutes les informations du texte.

*Question 6.2*
Ecrire le XML Schema correspondant.

# 7 Des eaux minérales

Considérons les paragraphes suivant qui décrivent une étude sur des eaux minérales:

Une bouteille d'eau Cristaline de 150 cl contient par litre 71 mg d'ions positifs calcium, et 5,5 mg d'ions positifs magnésium. On y trouve également des ions négatifs comme des chlorures à 20 mg par

litre et des nitrates avec 1 mg par litre. Elle est recueillie à St-Cyr la Source, dans le département du Loiret. Son code barre est 3274080005003 et son pH est de 7,45. Comme la bouteille est sale, quelques autres matériaux comme du fer s'y trouvent en suspension.

Une deuxième bouteille d'eau Cristaline a été, elle, recueillie à la source d'Auèle dans les Alpes Maritimes. La concentration en ions calcium est de 98 mg/l, et en ions magnésium de 4 mg/l. Il y a 3,6 mg/l d'ions chlorure et 2 mg/l de nitrates, pour un pH de 7,4. Le code barre de cette bouteille de 50 cl est 3268840001008.

Une bouteille de même contenance est de marque Volvic, et a été puisée à... Volvic (Puy-de-DÙme), bien connu pour ses sources donnant un pH neutre de 7. Elle comprend 11,5 mg/l d'ions calcium, 8,0 mg/l d'ions magnésium, 13,5 mg/l d'ions chlorures et 6,3 mg/l d'ions nitrates. Elle contient également des particules de silice. Son code barre est 3057640117008.

Indications supplémentaires:

- un pH est un nombre réel compris entre 0 et 14
- un code barre comporte systématiquement 13 chiffres
- un ion peut être positif ou négatif

*Question 7.1*
Ecrire un document XML qui permet de représenter de manière organisée toutes les informations concernant la première bouteille.

*Question 7.2*
Ecrire un schéma XML qui pourrait être commun à chaque document XML représentant les informations d'une bouteille.

*Chapitre V*

# Vocabulaire et espace de noms

## 1 De l'XSD à l'UML à l'XML♣

On considère le document suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://iupgmp.free.fr/quick"
            xmlns="http://iupgmp.free.fr/quick"
            elementFormDefault="qualified">
  <xsd:element name="a" type="A"/>
  <xsd:complexType name="A">
    <xsd:sequence>
      <xsd:element name="b" type="xsd:string"/>
      <xsd:element name="c" type="C" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:simpleType name="C">
    <xsd:restriction base="xsd:int">
      <xsd:minInclusive value="0"/>
      <xsd:maxInclusive value="20"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

*Question 1.1*
Quel est le nom du vocabulaire défini par ce xml schéma ?

*Question 1.2*
Quel est le nom du vocabulaire par défaut ?

*Question 1.3*
Quel est le préfixe utilisé pour le vocabulaire W3C des schémas ?

*Question 1.4*
Nombre et nom(s) de(s) type(s) complexe(s) déclaré(s) dans ce schéma ?

*Question 1.5*
Nombre et nom(s) de(s) type(s) simple(s) déclarés dans ce schéma ?

*Question 1.6*
Quel est l'élément racine défini par ce schéma ? A-t-il des éléments fils ? Si oui, lesquels ?

*Question 1.7*
Dessinez le diagramme UML correspondant à ce schéma.

*Question 1.8*
Ecrire un document XML valide par rapport à ce schéma.

## 2 XML schéma et vocabulaire

On considère le document suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://polytech.ujf-grenoble.fr/medecin"
            xmlns="http://polytech.ujf-grenoble.fr/medecin"
            elementFormDefault="qualified">
  <xsd:element name="médecin" type="Médecin"/>
  <xsd:complexType name="Médecin">
    <xsd:sequence>
      <xsd:element name="nom" type="xsd:string"/>
      <xsd:element name="spécialité" type="Spécialité"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:simpleType name="Spécialité">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Ophtalmologie"/>
      <xsd:enumeration value="ORL"/>
      <xsd:enumeration value="Dermatologie"/>
      <xsd:enumeration value="Médecine du sport"/>
      <xsd:enumeration value="Informallergologie"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

*Question 2.1*
Quel est le nom du vocabulaire défini par ce xml schéma ?

*Question 2.2*
Quel est le nom du vocabulaire par défaut ?

*Question 2.3*
Quel est le préfixe utilisé pour le vocabulaire W3C des schémas ?

*Question 2.4*
Nombre et nom(s) de(s) type(s) complexe(s) déclaré(s) dans ce schéma ?

*Question 2.5*
Nombre et nom(s) de(s) type(s) simple(s) déclarés dans ce schéma ?

*Question 2.6*
Quel est l'élément racine défini par ce schéma ? A-t-il des éléments fils ? Si oui, lesquels ?

*Question 2.7*
Dessinez le diagramme UML correspondant à ce schéma.

*Question 2.8*
Ecrire un document XML valide par rapport à ce schéma.

# 3 Vocabulaire et espace de noms

On considère les 2 documents suivant:

```
<sx:element name="composition" type="Nutrition"/>
<sx:complexType name="Nutrition">
  <sx:sequence>
    <sx:element name="nom"/>
    <sx:element name="glucide" type="Pourcentage"/>
    <sx:element name="protide" type="Pourcentage"/>
    <sx:element name="lipide" type="Pourcentage"/>
  </sx:sequence>
</sx:complexType>
<sx:simpleType name="Pourcentage">
  <sx:restriction base="sx:int">
    <sx:minInclusive value="0"/>
    <sx:maxInclusive value="100"/>
  </sx:restriction>
</sx:simpleType>
```

```
<element name="composition" type="tns:Nutrition"/>
<complexType name="Nutrition">
  <sequence>
    <element name="nom"/>
    <element name="glucide" type="tns:Pourcentage"/>
    <element name="protide" type="tns:Pourcentage"/>
    <element name="lipide" type="tns:Pourcentage"/>
  </sequence>
</complexType>
<simpleType name="Pourcentage">
  <restriction base="int">
    <minInclusive value="0"/>
    <maxInclusive value="100"/>
  </restriction>
</simpleType>
```

*Question 3.1*
Ecrire les entêtes du document pour ces deux schémas en prenant bien soin d'identifier les vocabulaires et espaces de nom.

*Question 3.2*
Nombre et nom(s) de(s) type(s) complexe(s) déclaré(s) dans le schéma Nutrition v1.

*Question 3.3*
Nombre et nom(s) de(s) type(s) simple(s) déclaré(s) dans le schéma Nutrition v1.

*Question 3.4*
De quel type est l'élément "nom" dans le schéma Nutrition v1 ?

*Question 3.5*
Dans un document XML contraint au schéma Nutrition v1, les pourcentages de glucides, protides et lipides peuvent être donnés dans le désordre. Vrai ou faux ?

*Question 3.6*
Dans un document XML contraint au schéma Nutrition v2, on peut avoir un pourcentage de 99.5% . Vrai ou faux ?

*Question 3.7*
Quel est le nom de l'élément racine d'un document contraint au schéma Nutrition v1 ?

# 4 Plusieurs vocabulaires

On considère le document xml suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://polytech.ujf-grenoble.fr/adherent"
            xmlns="http://polytech.ujf-grenoble.fr/adherent"
            xmlns:idt="http://polytech.ujf-grenoble.fr/idty"
            xmlns:aut="http://polytech.ujf-grenoble.fr/authors"
            elementFormDefault="qualified">
  <xsd:import schemaLocation="idty.xsd" namespace="http://polytech.ujf-grenoble
      .fr/idty"/>
  <xsd:import schemaLocation="authors.xsd" namespace="http://polytech.ujf-
      grenoble.fr/authors"/>
  <xsd:element name="adherent" type="Adherent"/>
  <xsd:complexType name="Adherent">
    <xsd:sequence>
      <xsd:element name="identité" type="idt:Id"/>
      <xsd:element name="emprunts" type="Emprunts" minOccurs="0" maxOccurs="
          unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Emprunts">
    <xsd:sequence>
      <xsd:element name="titre" type="xsd:string"/>
      <xsd:element name="auteur" type="aut:InfoAuteur"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

Les informations sur un auteur sont

- son identité
- ses dates de naissance et de mort

Le type correspondant à l'identité d'un adhérent ou d'un auteur contient leur nom et leur prénom.

*Question 4.1*
Dans quel(s) fichier(s) doivent se trouver les types nécessaires pour rendre ce schéma valide ?

*Question 4.2*
Dessiner le diagramme UML correspondant à ces schéma.

*Question 4.3*
Créez les schémas xml correspondant en faisant particulièrement attention aux en-têtes.

*Question 4.4*
Ecrire un document xml modélisant un adhérent ayant emprunté un livre, et valide par rapport à ces schémas, en faisant bien attention aux en-têtes.

On reprend le schéma présenté au début en changeant l'entête:

```
<?xml version="1.0" encoding="UTF-8"?>
<tordu:schema targetNamespace="http://polytech.ujf-grenoble.fr/adherent"
              xmlns="http://polytech.ujf-grenoble.fr/idty"
              xmlns:adh="http://polytech.ujf-grenoble.fr/adherent"
              xmlns:aut="http://polytech.ujf-grenoble.fr/authors"
              xmlns:tordu="http://www.w3.org/2001/XMLSchema"
              elementFormDefault="qualified">
   ...
</tordu:schema>
```

*Question 4.5*
Cette entête pose-t-elle problème ?

*Question 4.6*
Comment modifier le reste du schéma précédent avec cette nouvelle entête ?

*Chapitre VI*

# XSLT: Transformations en XML

## 1 Hello XSLT♣

On considère le docuemnt `greeting.xml` suivant:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- greeting -->
<greeting xmlns="http://ujf-grenoble.fr/greeting">
    Hello World !
</greeting>
```

*Question 1.1*
Dessiner l'arbre XML de ce document.

On souhaite transformer ce document en un document xhtml `greegint.xhtml`.

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
    <head>
        <title>Greetings</title>
    </head>
    <body>
        <h1>
            Hello World !
        </h1>
    </body>
</html>
```

*Question 1.2*
Écrire une feuille de style XSLT qui produit la transformation.

*Question 1.3*
Pour chaque ligne de la feuille de transformation, dites s'il s'agit du mécanisme *Push*, *Pull* ou bien *Navigation*.

## 2 Centre de soins

On souhaite modéliser un centre de soins. Un centre de soins possède un identifiant unique, une liste de médecins, un ensemble de codes pour la tarification des interventions ainsi qu'une liste des fiches de soins.

Chaque médecin a un nom, un prénom, un identifiant et une spécialité. Par exemple, le docteur Harry Covert est ORL et a pour identifiant B81-1854. Pour la tarification, chaque code a un nom, un

libellé et un montant, par exemple, FP désignera un forfait pédiatrique avec un coût de 32 euros. Enfin, une fiche de soin est effectuée à une date précise, contient une référence au médecin traitant, le nom et le prénom d'un patient ainsi que chacun des actes qui ont été effectués sur le patient (représenté par le code de l'acte).

Une instance d'un tel document XML est par exemple:

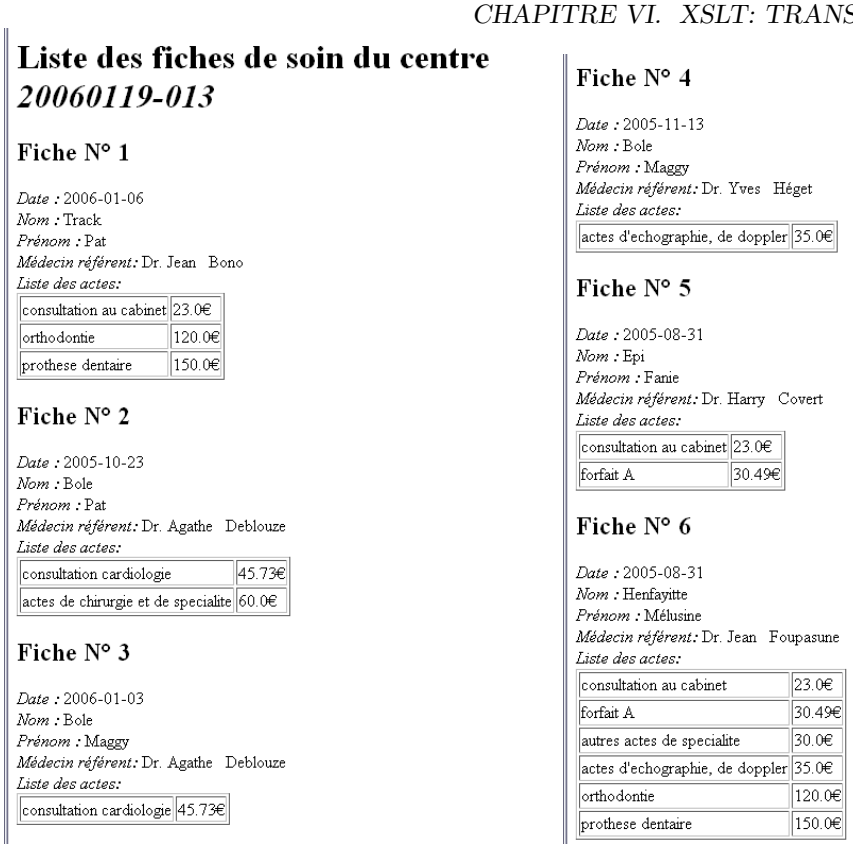
```
<?xml version="1.0" encoding="UTF-8"
    standalone="yes"?>
<centre id="20060119-013">

  <medecins>
      <medecin>
          <id>A12-4034</id>
        <nom>Bono</nom>
        <prénom>Jean</prénom>
        <spécialité>ORL</spécialité>
        </medecin>
        <medecin>
         <id>B45-1974</id>
         <nom>Deblouze</nom>
         <prénom>Agathe</prénom>
         <spécialité>Cardiologue</spé
             cialité>
      </medecin>

      <medecin>
         <id>A31-5146</id>
         <nom>Héget</nom>
         <prénom>Yves</prénom>
         <spécialité>Gynécologie</spé
             cialité>
      </medecin>

      <medecin>
         <id>B81-1854</id>
         <nom>Covert</nom>
         <prénom>Harry</prénom>
         <spécialité>ORL</spécialité>
      </medecin>

      <medecin>
         <id>C04-0139</id>
         <nom>Foupasune</nom>
         <prénom>Jean</prénom>
         <spécialité>ORL</spécialité>
      </medecin>
  </medecins>

  <codes>
    <codeCout>
        <nom>CS</nom>
        <libellé>consultation au
            cabinet</libellé>
        <cout>23.0</cout>
    </codeCout>
    <codeCout>
        <nom>CSC</nom>
        <libellé>consultation
            cardiologie</libellé>
        <cout>45.73</cout>
    </codeCout>
    <codeCout>
        <nom>FP</nom>
        <libellé>forfait pediatrique</
            libellé>
        <cout>32.0</cout>
    </codeCout>
    <codeCout>
        <nom>KC</nom>
        <libellé>actes de chirurgie et
            de specialite</libellé>
        <cout>60.0</cout>
    </codeCout>
    <codeCout>
        <nom>KE</nom>
        <libellé>actes d'echographie,
            de doppler</libellé>
        <cout>35.0</cout>
    </codeCout>
    <codeCout>
        <nom>K</nom>
        <libellé>autres actes de
            specialite</libellé>
        <cout>30.0</cout>
    </codeCout>
    <codeCout>
        <nom>KFA</nom>
        <libellé>forfait A</libellé>
        <cout>30.49</cout>
    </codeCout>
    <codeCout>
        <nom>KFB</nom>
        <libellé>forfait B</libellé>
        <cout>60.98</cout>
    </codeCout>
    <codeCout>
        <nom>ORT</nom>
        <libellé>orthodontie</libellé>
        <cout>120.0</cout>
    </codeCout>
    <codeCout>
        <nom>PRO</nom>
        <libellé>prothese dentaire</
            libellé>
        <cout>150.0</cout>
    </codeCout>
</codes>

<fiches>
    <ficheDeSoins>
      <date>2006-01-06</date>
      <medecin id="A12-4034"/>
      <patient>
          <nom>Track</nom>
          <prénom>Pat</prénom>
      </patient>
      <acte>
          <code>CS</code>
      </acte>
```

```
        <acte>
            <code>ORT</code>
        </acte>
        <acte>
            <code>PRO</code>
        </acte>
      </ficheDeSoins>

      <ficheDeSoins>
        <date>2005-10-23</date>
        <medecin id="B45-1974"/>
        <patient>
            <nom>Bole</nom>
            <prénom>Pat</prénom>
        </patient>
        <acte>
            <code>CSC</code>
         </acte>
        <acte>
            <code>KC</code>
        </acte>
      </ficheDeSoins>

      <ficheDeSoins>
        <date>2006-01-03</date>
        <medecin id="B45-1974"/>
        <patient>
            <nom>Bole</nom>
            <prénom>Maggy</prénom>
        </patient>
        <acte>
            <code>CSC</code>
        </acte>
      </ficheDeSoins>

      <ficheDeSoins>
        <date>2005-11-13</date>
        <medecin id="A31-5146"/>
        <patient>
            <nom>Bole</nom>
            <prénom>Maggy</prénom>
        </patient>
        <acte>
            <code>KE</code>
        </acte>
      </ficheDeSoins>

      <ficheDeSoins>
        <date>2005-08-31</date>
        <medecin id="B81-1854"/>
        <patient>
            <nom>Epi</nom>
            <prénom>Fanie</prénom>
        </patient>
        <acte>
            <code>CS</code>
        </acte>
        <acte>
            <code>KFA</code>
        </acte>
      </ficheDeSoins>

      <ficheDeSoins>
        <date>2005-08-31</date>
        <medecin id="C04-0139"/>
        <patient>
            <nom>Henfayitte</nom>
            <prénom>Mélusine</prénom>
        </patient>
        <acte>
            <code>CS</code>
        </acte>
        <acte>
            <code>KFA</code>
        </acte>
        <acte>
            <code>K</code>
        </acte>
        <acte>
            <code>KE</code>
        </acte>
        <acte>
            <code>ORT</code>
        </acte>
        <acte>
            <code>PRO</code>
        </acte>
      </ficheDeSoins>
  </fiches>

</centre>
```

*Question 2.1*
Écrire le diagramme UML correspondant au schéma pour valider ce document.

On souaite à présent représenter les informations du centre de soins sous la forme d'un document xhtml comme dans la figure ci-dessous:

*Question 2.2*
Écrire une feuille de transformation XSLT qui représente les infos du centre de soin comme la figure VI.1.



**Liste des fiches de soin du centre**
***20060119-013***

**Fiche N° 1**

*Date :* 2006-01-06
*Nom :* Track
*Prénom :* Pat
*Médecin référent:* Dr. Jean Bono
*Liste des actes:*

| | |
|---|---|
| consultation au cabinet | 23.0€ |
| orthodontie | 120.0€ |
| prothese dentaire | 150.0€ |

**Fiche N° 2**

*Date :* 2005-10-23
*Nom :* Bole
*Prénom :* Pat
*Médecin référent:* Dr. Agathe Deblouze
*Liste des actes:*

| | |
|---|---|
| consultation cardiologie | 45.73€ |
| actes de chirurgie et de specialite | 60.0€ |

**Fiche N° 3**

*Date :* 2006-01-03
*Nom :* Bole
*Prénom :* Maggy
*Médecin référent:* Dr. Agathe Deblouze
*Liste des actes:*

| | |
|---|---|
| consultation cardiologie | 45.73€ |

**Fiche N° 4**

*Date :* 2005-11-13
*Nom :* Bole
*Prénom :* Maggy
*Médecin référent:* Dr. Yves Héget
*Liste des actes:*

| | |
|---|---|
| actes d'echographie, de doppler | 35.0€ |

**Fiche N° 5**

*Date :* 2005-08-31
*Nom :* Epi
*Prénom :* Fanie
*Médecin référent:* Dr. Harry Covert
*Liste des actes:*

| | |
|---|---|
| consultation au cabinet | 23.0€ |
| forfait A | 30.49€ |

**Fiche N° 6**

*Date :* 2005-08-31
*Nom :* Henfayitte
*Prénom :* Mélusine
*Médecin référent:* Dr. Jean Foupasune
*Liste des actes:*

| | |
|---|---|
| consultation au cabinet | 23.0€ |
| forfait A | 30.49€ |
| autres actes de specialite | 30.0€ |
| actes d'echographie, de doppler | 35.0€ |
| orthodontie | 120.0€ |
| prothese dentaire | 150.0€ |

Figure VI.1: Représentation xhtml des informations d'un centre de soins.

## 3 Liste de Films... ...It is back

On considère le document XML sur la liste de films du TD précédent.

*Question 3.1*
Ecrire un fichier de transformation XSLT qui transforme le document xml en un document xhtml qui représente les données comme indiqué dans la figure VI.2

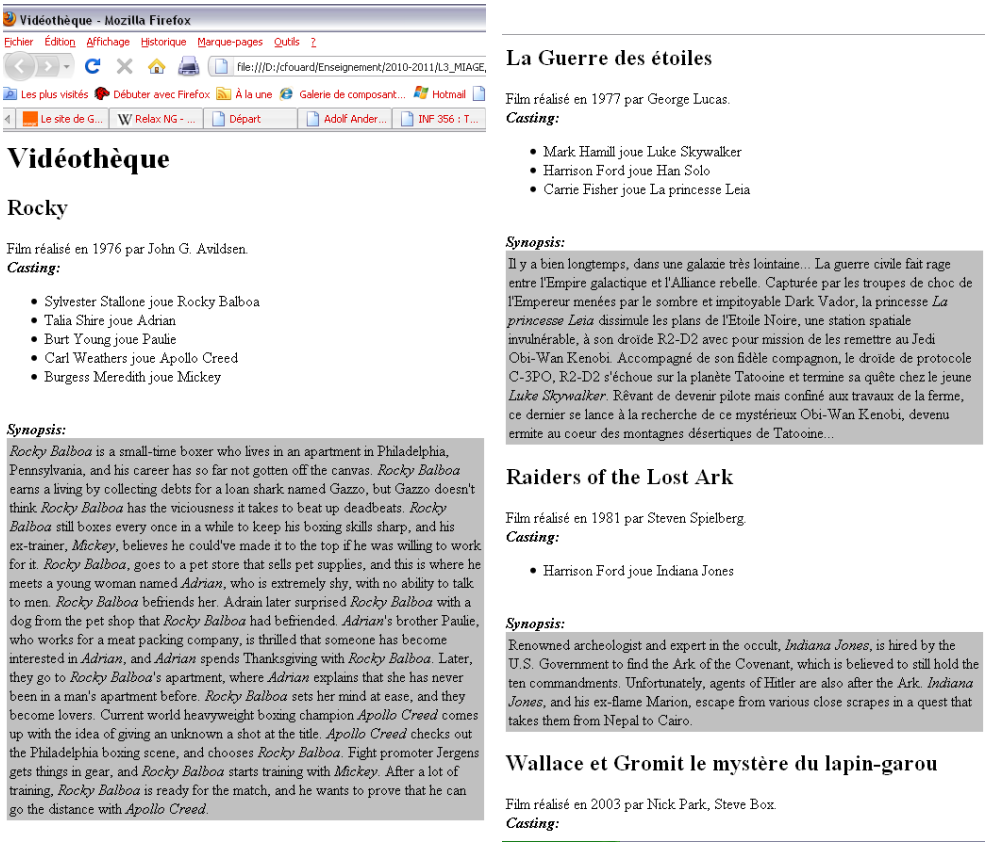



Figure VI.2: Représentation HTML d'une vidéothèque.

*Chapitre VIII*

# *Contraintes de cohérence : unicité et existence en XML Schema*

## 1 Les clefs du magasin♣

Le Schema XML suivant modélisant le magasin *HejDo* concurrent de la célèbre marque suédoise est défini comme suit : En plus d'une ville (`ville : xs:string`) et d'une année (`annee : xs:gYear`), attributs de sa racine (`hejdo : hd:Magasin`), le magasin contient:

- une liste de références de produits contenus dans tout le magasin, c'est à dire *stock et exposition compris* : la liste `magasin`.
- une liste de de références de produits mis en exposition dans ce même magasin : liste `expo`.

Un produit chez HejDo, par exemple le produit *Svalbard* qui désigne une table en bois, est désigné par un nom `nomProduit` et une référence `refProduit`. Un tel produit peut être composé de plusieurs pièces référencées par une lettre dans `refElementProduit`, par exemple le plateau de la table Svalbard qui est référencé A, et les pieds de cette même table référencés B ; il s'agit donc de produits différents. Leur quantité est connue grâce à l'attribut `quantite`.// Un exemple d'instance XML contrainte par ce SchemaXML est donnée ci-dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<hd:hejdo
    xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xmlns:hd='http://wwww.hejdo.fr/magasin'
    xsi:schemaLocation='http://wwww.hejdo.fr/magasin Magasin.xsd'
    ville = "Grenoble"
    annee = "2020">
    <hd:magasin>
        <hd:reference nomProduit="Svalbard" refProduit="1" refElementProduit="A
            " quantite="12"/>
        <hd:reference nomProduit="Svalbard" refProduit="1" refElementProduit="B
            " quantite="32"/>
        <hd:reference nomProduit="Mulnork" refProduit="2" refElementProduit="A"
             quantite="25"/>
        <hd:reference nomProduit="Traktill" refProduit="3" refElementProduit="A
            "  quantite="1"/>
        <hd:reference nomProduit="Traktill" refProduit="3" refElementProduit="A
            " quantite="1"/>
    </hd:magasin>
    <hd:expo>
        <hd:reference nomProduit="Svalbard" refProduit="1" refElementProduit="A
            " quantite="1"/>
        <hd:reference nomProduit="Svalbard" refProduit="1" refElementProduit="B
            " quantite="4"/>
```

```
18          <hd:reference nomProduit="Svalbard" refProduit="1" refElementProduit="B
                " quantite="1"/>
19          <hd:reference nomProduit="Mulnork" refProduit="2" refElementProduit="A"
                 quantite="2"/>
20          <hd:reference nomProduit="Traktill" refProduit="3" refElementProduit="A
                " quantite="0"/>
21          <hd:reference nomProduit="Traktill" refProduit="3" refElementProduit="A
                " quantite="1"/>
22          <hd:reference nomProduit="Arhneek" refProduit="4" refElementProduit="A"
                 quantite="0"/>
23      </hd:expo>
24  </hd:hejdo>
```

*Question 1.1*
Ecrivez le diagramme UML modélisant cette instance.

*Question 1.2*
Ecrivez le Schema XML modélisant cette instance.

*Question 1.3*
Ajoutez une clef d'unicité nommée `magRefUnique` : `xs:unique` spécifiant l'unicité des produits dans l'ensemble du magasin.

*Question 1.4*
Expliquez de quel élément cette clef d'unicité peut-elle être *élément fils*, les éléments sélectionnés par son sélecteur, la valeur de son ou ses champs.

*Question 1.5*
Testez cette clef d'unicité `magRefUnique` sur le document XML contraint fourni ci-dessus. Que se passe-t-il ?

*Question 1.6*
Ajoutez une clef d'unicité nommée `expoRefUnique` spécifiant l'unicité des produits exposés dans le magasin.

*Question 1.7*
Mêmes questions que précedemment.

*Question 1.8*
Les attributs d'un `Produit` sont tous requis dans le schéma de magasin HejDo. Si l'on enlève les spécifications `use="required"` pour les attributs `nomProduit`, `refProduit` et `refElementProduit`, que faut il faire pour garantir l'exitence de ces attributs à l'aide d'une clef d'unicité ?

*Question 1.9*
Corrigez l'instance XML de façon à ce que l'unicité des produits soit respectée.

On souhaite maintenant spécifier l'existence des références mises en exposition (liste `expo`) dans celles listées dans l'ensemble du magasin (liste `magasin`). Comme tout produit exposé est, de fait, présent dans le magasin, sa référence doit nécessairement se trouver dans la liste des références de la liste `magasin`.

*Question 1.10*
Ajoutez une clef d'existence nommée `refExist` de façon à ce que chaque produit référencé dans la liste `expo` existe dans la liste `magasin`.

*Question 1.11*
Expliquez de quel élément cette clef d'existence peut-elle être *élément fils*, les éléments sélectionnés par son sélecteur, la valeur de son ou ses champs. A quelle clef se réfère t-elle ? pourquoi ?

*Question 1.12*
Testez cette clef d'existence `refExist` sur le document XML contraint fourni ci-dessus. Que se passe-t-il ?

*Question 1.13*
Corrigez l'instance XML de façon à ce que l'existence des produits en exposition soit vérifiée dans la liste globale du magasin.

*Chapitre IX*

# Parseurs XML

## 1 Simple DOM♣

On considère le docuemnt `bouquins.xml` suivant:

```
<?xml version="1.0" encoding="utf-8"?>
<liste>
        <livre xml:lang="fr">
                <titre>XML</titre>
                <auteur>Harold</auteur>
                <auteur>Means</auteur>
        </livre>
        <livre>
                <titre>Quantum Computation</titre>
                <auteur>Nielsen</auteur>
        </livre>
</liste>
```

*Question 1.1*
Dessiner l'arbre XML de ce document.

Soit la méthode `main` de la classe `Simple`.

```
public class Simple {
    public static void main(String[] args) {
        DOMParser parser = new DOMParser() ;
        parser.parse("bouquins.xml") ;
        Document doc = parser.getDocument() ;

        doc.getNodeName();
        doc.getChildNodes().item(0);
        doc.getChildNodes().item(0).gtNodeName();
        doc.getChildNodes().item(0).getChildNodes().item(1);
        doc.getChildNodes().item(0).getChildNodes().item(0);
        doc.getChildNodes().item(0).getChildNodes().item(0).getNodeName();
        doc.getChildNodes().item(0).getChildNodes().
            item(1).getChildNodes().item(1).getChildNodes().
            item(0).getNodeName();
        doc.getChildNodes().item(0).getChildNodes().
            item(1).getChildNodes().item(1).getChildNodes().
            item(0).getNodeValue();
        doc.getElementsByTagName("titre").item(0).
            getChildNodes().item(0).getNodeValue();
        doc.getElementsByTagName("livre").item(0).hasAttributes();
```

```
22          doc.getElementsByTagName("livre").item(1).hasAttributes();
23      }
24  }
```

*Question 1.2*
Quelles sont les valeurs des expressions lignes 7 à 22 ?

# 2 DOM

## 2.a Ecriture d'expressions DOM

On considère le docuemnt `listeActeurs.xml` suivant:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<liste-acteurs>
  <acteur id="clint">
    <prenom>Clint</prenom>
    <nom>Eastwood</nom>
    <naissance>31 Mai 1930</naissance>
    <nationalite>américaine</nationalite>
    <photo source="clint.jpg" alt="Clint Eastwood"/>
    <site url="http://www.clinteastwood.net/"/>
    <biographie>
      <p>
        Né d'un père comptable, le jeune Clinton mène avec ses parents
        une vie de nomade. Il passe son adolescence à Oakland et ne
        pense pas du tout à devenir acteur. Il fait des petits boulots
        sans grande conviction.  Puis il part à l'armée où il fait des
        rencontres décisives et obtient du travail chez Universal. Il
        fait sa première apparition en <annee>1955</annee> dans
        <film>La Revanche de la créature</film> puis enchaîne les
        petits rôles dans cinq films où personne ne le remarque
        véritablement.
      </p>
      <p>
        Avec le drame <film>Million dollar baby</film>, le cinéaste
        obtient une véritable consécration en remportant, douze ans
        après <film>Impitoyable</film>, l'Oscar du Meilleur film et du
        Meilleur réalisateur, ses comédiens <refacteur
        code="hilary">Hilary Swank</refacteur> et <refacteur
        code="morgan">Morgan Freeman</refacteur> repartant avec les
        statuettes de La Meilleure actrice et du Meilleur second rôle
        masculin.
      </p>
    </biographie>
  </acteur>
  <acteur id="hilary">
    <prenom>Hilary</prenom>
    <nom>Swank</nom>
    <naissance>30 Juillet 1974</naissance>
    <nationalite>américaine</nationalite>
    <photo source="hilaryswank.jpg" alt="Hilary Swank"/>
    <site url="http://www.hilaryswankfan.com/"/>
    <biographie>
      <p>
        Championne de natation, Hilary Swank s'oriente rapidement vers
        le métier d'actrice.  A l'âge de dix-huit ans, elle fait une
        courte apparition dans <film>Buffy, tueuse de vampires</film>
        en <annee>1992</annee>, une comédie fantastique de Fran Rubel
        Kuzui, et décroche en <annee>1994</annee> le rôle-titre de
        <film>Miss Karaté Kid</film> de Christopher Cain.
      </p>
      <p>
        C'est <film>Boys don't cry</film>, un drame de
        <realisateur>Kimberly Peirce</realisateur>, qui la révèle
        véritablement au grand public en <annee>2000</annee>.  Sa
        prestation du travesti Brandon Teena lui vaut l'Oscar et le
        Golden Globe de la Meilleure actrice, les prix
        d'interprétation des New York Film Critics, Los Angeles Film
        Critics et Chicago Film Critics, ainsi que le Broadcast Film
        Critics.
```

```
      </p>
      <p>
        En <annee>2005</annee>, elle rafle pour la deuxième fois (et
        en seulement deux nominations) l'Oscar de la Meilleure actrice
        grâce à sa performance de boxeuse surentraînée par <refacteur
        code="clint">Clint Eastwood</refacteur> dans <film>Million
        dollar baby</film>.
      </p>
    </biographie>
  </acteur>
  <acteur id="morgan">
    <prenom>Morgan</prenom>
    <nom>Freeman</nom>
    <naissance>1 Juin 1937</naissance>
    <nationalite>américaine</nationalite>
    <photo source="freeman.jpg" alt="Morgan Freeman"/>
    <biographie>
      <p>
        Morgan Freeman est diplômé du lycée de Greenwood, dans le
        Mississippi.  A dix-huit ans, il s'engage dans l'Air Force et,
        une fois ses obligations militaires accomplies, s'installe en
        Californie pour étudier la danse et l'art dramatique au Los
        Angeles City College. C'est à Broadway qu'il fait ses débuts
        de comédien en <annee>1967</annee>.
      </p>
      <p>
        A 68 ans, il obtient enfin la reconnaissance de la profession
        en remportant l'Oscar du Meilleur second rôle masculin pour sa
        prestation d'ancien boxeur borgne dans <film>Million dollar
        baby</film> (<annee>2005</annee>) de son fidèle ami <refacteur
        code="clint">Clint Eastwood</refacteur>.
      </p>
    </biographie>
  </acteur>
</liste-acteurs>
```

Soient les expressions DOM suivantes (la variable `doc` contient l'arbre DOM du document):

```
doc.getChildNodes().item(0).getNodeName();
doc.getChildNodes().item(0).getNodeType();
doc.getChildNodes().item(0).getNodeValue();
```

*Question 2.1*
Quelle sont les valeurs obtenues ?

```
            System.out.println(doc.getChildNodes().item(0).getChildNodes().item
                (0));
            System.out.println(doc.getChildNodes().item(0).getChildNodes().item
                (0).getNodeName());
            System.out.println(doc.getChildNodes().item(0).getChildNodes().item
                (0).getNodeType());
            System.out.println(doc.getChildNodes().item(0).getChildNodes().item
                (0).getNodeValue());
```

*Question 2.2*
Même question avec les instructions précédentes.

```
doc.getChildNodes().item(0).getChildNodes().item(1).getNodeName();
doc.getChildNodes().item(0).getChildNodes().item(1).getNodeType();
doc.getChildNodes().item(0).getChildNodes().item(1).getNodeValue();
doc.getChildNodes().item(0).getChildNodes().item(1).hasAttributes();
```

```
doc.getChildNodes().item(0).getChildNodes().item(1).getAttributes().getLength()
    ;
doc.getChildNodes().item(0).getChildNodes().item(1).getAttributes().item(0);
```

*Question 2.3*
Même question avec le code précédent.

```
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(0)
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(0).
    getTextContent()
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(1)
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(1).
    getTextContent()
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(2)
doc.getChildNodes().item(0).getChildNodes().item(1).getChildNodes().item(2).
    getTextContent()
```

*Question 2.4*
Même question avec le code précédent.

```
((Element) doc.getElementsByTagName("acteur").item(1)).getElementsByTagName("p"
    ).item(1).getTextContent();
```

*Question 2.5*
Même question avec le code précédent.

## 2.b Ecriture d'expressions DOM

On considère à présent le document `listeFilms.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<films>
  <film lang="en">
    <photo href="rocky.jpg"/>
    <titre>Rocky</titre>
    <annee>1976</annee>
    <realisateur>John G. Avildsen</realisateur>
    <casting>
      <acteur id="rocky" personnage="Rocky Balboa">Sylvester Stallone</acteur>
      <acteur id="adrian" personnage="Adrian">Talia Shire</acteur>
      <acteur id="paulie" personnage="Paulie">Burt Young</acteur>
      <acteur id="creed" personnage="Apollo Creed">Carl Weathers</acteur>
      <acteur id="mickey" personnage="Mickey">Burgess Meredith</acteur>
    </casting>
    <synopsis>
      <perso ref="rocky"/> is a small-time boxer who lives in an
      apartment in Philadelphia, Pennsylvania, and his career has so
      far not gotten off the canvas. <perso ref="rocky"/> earns a
      living by collecting debts for a loan shark named Gazzo, but
      Gazzo doesn't think <perso ref="rocky"/> has the viciousness it
      takes to beat up deadbeats. <perso ref="rocky"/> still boxes
      every once in a while to keep his boxing skills sharp, and his
      ex-trainer, <perso ref="mickey"/>, believes he could've made it
      to the top if he was willing to work for it. <perso
      ref="rocky"/>, goes to a pet store that sells pet supplies, and
      this is where he meets a young woman named <perso
      ref="adrian"/>, who is extremely shy, with no ability to talk to
      men. <perso ref="rocky"/> befriends her. Adrain later surprised
      <perso ref="rocky"/> with a dog from the pet shop that <perso
```

```
      ref="rocky"/> had befriended.  <perso ref="adrian"/>'s brother
      Paulie, who works for a meat packing company, is thrilled that
      someone has become interested in <perso ref="adrian"/>, and
      <perso ref="adrian"/> spends Thanksgiving with <perso
      ref="rocky"/>. Later, they go to <perso ref="rocky"/>'s
      apartment, where <perso ref="adrian"/> explains that she has
      never been in a man's apartment before.  <perso ref="rocky"/>
      sets her mind at ease, and they become lovers.  Current world
      heavyweight boxing champion <perso ref="creed"/> comes up with
      the idea of giving an unknown a shot at the title.  <perso
      ref="creed"/> checks out the Philadelphia boxing scene, and
      chooses <perso ref="rocky"/>. Fight promoter Jergens gets things
      in gear, and <perso ref="rocky"/> starts training with <perso
      ref="mickey"/>. After a lot of training, <perso ref="rocky"/> is
      ready for the match, and he wants to prove that he can go the
      distance with <perso ref="creed"/>.
    </synopsis>
  </film>
  <film lang="fr">
    <titre>La Guerre des étoiles</titre>
    <annee>1977</annee>
    <realisateur>George Lucas</realisateur>
    <casting>
      <acteur id="lukemonfils" personnage="Luke Skywalker">Mark Hamill</acteur>
      <acteur personnage="Han Solo">Harrison Ford</acteur>
      <acteur id="leia" personnage="La princesse Leia">Carrie Fisher</acteur>
    </casting>
    <synopsis>
      Il y a bien longtemps, dans une galaxie très lointaine... La
      guerre civile fait rage entre l'Empire galactique et l'Alliance
      rebelle.  Capturée par les troupes de choc de l'Empereur menées
      par le sombre et impitoyable Dark Vador, la princesse <perso
      ref="leia"/> dissimule les plans de l'Etoile Noire, une station
      spatiale invulnérable, à son droide R2-D2 avec pour mission de
      les remettre au Jedi Obi-Wan Kenobi. Accompagné de son fidèle
      compagnon, le droide de protocole C-3PO, R2-D2 s'échoue sur la
      planète Tatooine et termine sa quête chez le jeune <perso
      ref="lukemonfils"/>.  Rêvant de devenir pilote mais confiné aux
      travaux de la ferme, ce dernier se lance à la recherche de ce
      mystérieux Obi-Wan Kenobi, devenu ermite au coeur des montagnes
      désertiques de Tatooine...
    </synopsis>
  </film>
  <film lang="en">
    <titre>Raiders of the Lost Ark</titre>
    <annee>1981</annee>
    <realisateur>Steven Spielberg</realisateur>
    <casting>
      <acteur id="indy" personnage="Indiana Jones">Harrison Ford</acteur>
    </casting>
    <synopsis>
      Renowned archeologist and expert in the occult, <perso
      ref="indy"/>, is hired by the U.S.  Government to find the Ark
      of the Covenant, which is believed to still hold the ten
      commandments. Unfortunately, agents of Hitler are also after the
      Ark. <perso ref="indy"/>, and his ex-flame Marion, escape from
      various close scrapes in a quest that takes them from Nepal to
      Cairo.
    </synopsis>
  </film>
  <film lang="fr">
    <titre>Wallace et Gromit le mystère du lapin-garou</titre>
    <annee>2003</annee>
```

```
    <realisateur>Nick Park, Steve Box</realisateur>
    <casting/>
    <synopsis>
      Une "fièvre végétarienne" intense règne dans la petite ville de
      Wallace et Gromit, et l'ingénieux duo a mis à profit cet
      engouement en inventant un produit anti-nuisibles humain et
      écolo, qui épargne la vie des lapins. L'astuce consiste
      simplement à capturer, à la main, un maximum de ces rongeurs et
      à les mettre en cage.  A quelques jours du Grand Concours Annuel
      de Légumes, les affaires de Wallace et Gromit n'ont jamais été
      aussi florissantes, et tout irait pour le mieux dans le meilleur
      des mondes, si un lapin-garou géant ne venait soudain s'attaquer
      aux sacro-saints potagers de la ville.  Pour faire face à ce
      péril inédit, l'organisatrice du concours, Lady Tottington, se
      tourne vers nos deux "spécialistes" et leur demande
      d'appréhender le monstre.
    </synopsis>
  </film>
</films>
```

En utilisant la variable `doc` de type `Document` pour l'arbre DOM, et `System.out.println` pour écrire, on souhaite écrire:

1. le nombre de films disponibles
2. le troisième acteur du deuxième film
3. les titres des films dont la fiche est en anglais
4. le nom du personnage dont l'identifiant est `lukemonfils`
5. le titre des films sortis en 1981
6. le nombre de références faites au personnage dont l'identifiant est `indy`
7. les films sans acteur

*Question 2.6*
Ecrire le code DOM Java correspondant.

Au lieu d'utiliser `getChildNodes().item(??)`, il est beaucoup plus simple d'utiliser `getElementsByTagName("nomDelElement")`, car on n'a pas besoin de compter les retours à la ligne entre chaque élément !

## 2.c DOM et XPath

Le programme Java ci-dessous permet de lire un document au format DOM et d'utiliser des expressions XPath pour accéder aux valeurs qu'il contient.

```
package test_dom_xpath;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import javax.xml.namespace.NamespaceContext;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.stream.events.Namespace;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpression;
```

```
14 import javax.xml.xpath.XPathExpressionException;
15 import javax.xml.xpath.XPathFactory;
16
17 import org.w3c.dom.Document;
18 import org.w3c.dom.NodeList;
19
20 public class XPathExample {
21
22     public static void main(String[] args) throws Exception {
23         // Parse le fichier XML dans un document DOM
24         String fileName = "src/test_dom_xpath/employes.xml";
25         Document document = getDocument(fileName);
26
27         // pour stocker les expression XPath
28         String xpathExpression = ".....................................";
29     }
30
31     /**
32      * Récupère la liste des contenus textes accessibles par l'expression xpath
            passée en paramètres
33      */
34     private static List<String> evaluateXPathString(Document document, String
        xpathExpression) throws Exception {
35         // Crée une fabrique d'objet XPath
36         XPathFactory xpathFactory = XPathFactory.newInstance();
37         // Crée un objet XPath
38         XPath xpath = xpathFactory.newXPath();
39         List<String> values = new ArrayList<>();
40         try {
41             // Crée une expression XPath
42             XPathExpression expr = xpath.compile(xpathExpression);
43             // Evalue l'expression XPath sur le document DOM
44             NodeList nodes = (NodeList) expr.evaluate(document, XPathConstants.
                   NODESET);
45 //             NodeList nodes = (NodeList) xpath.evaluate(xpathExpression,
    document, XPathConstants.NODESET);
46             for (int i = 0; i < nodes.getLength(); i++) {
47                 values.add(nodes.item(i).getNodeValue());
48             }
49         } catch (XPathExpressionException e) {
50             e.printStackTrace();
51         }
52         return values;
53     }
54
55
56     /**
57      * Crée un document DOM à partir d'un fichier XML
58      */
59     private static Document getDocument(String fileName) throws Exception {
60         DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
61         factory.setValidating(false);
62         factory.setNamespaceAware(true);
63         factory.setIgnoringComments(true);
64         factory.setIgnoringElementContentWhitespace(true);
65         DocumentBuilder builder = factory.newDocumentBuilder();
66         Document doc = builder.parse(fileName);
67         return doc;
68     }
69 }
```

Conséidérons le document XML ci-dessous.

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<employés>
    <employé id="1">
        <prénom>Ella</prénom>
        <nom>Mieuafair</nom>
        <service>
            <id>101</id>
            <nom>IT</nom>
        </service>
    </employé>
    <employé id="2">
        <prénom>Jean</prénom>
        <nom>Talu</nom>
        <service>
            <id>102</id>
            <nom>HR</nom>
        </service>
    </employé>
    <employé id="3">
        <prénom>Alex</prénom>
        <nom>Andrin</nom>
        <service>
            <id>103</id>
            <nom>FINANCE</nom>
        </service>
    </employé>
</employés>
```

Proposez des expressions XPath pour:

*Question 2.7*
...récupérer la liste des identifiants des employés.

*Question 2.8*
...récupérer la liste des identifiants des employés travaillant dans le service IT.

*Question 2.9*
...récupérer l'id de l'employé Alex.

*Question 2.10*
...récupérer les employés d'id $> 5$.

*Question 2.11*
...récupérer les employés dont l'id contient 1.

## 2.d Modification d'arbre DOM

Dans le document `listeFilms.xml`, donnez des expressions DOM pour faire ce qui est demandé. Vous utiliserez la variable `doc` de type `Document` pour l'arbre DOM.

*Question 2.12*
Ajouter un acteur au deuxième film: *Phil Brown* joue le personnage de *Uncle Owen*. On lui donne l'if *owen*.

*Question 2.13*
Supprimez *Hirrison Ford* de la liste des acteus de la guerre des étoiles et changez le titre du film: *Star Wars*.

*Question 2.14*
Rajoutez un film à la liste. On ajoutera le titre *Marie et Max*, le réalisateur *Adam Elliot* et la date *2009*.

# 3 SAX: Lecture d'un Mémo♣

On souhaite mettre en forme le document suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<memo xmlns="http://ujf-grenoble.fr/memo"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://ujf-grenoble.fr/memo memo.xsd">

    <from>John</from>
    <to>Bob</to>
    <content>
        <subject>Bonjour</subject>
        <text>Ca va ?</text>
    </content>
</memo>
```

*Question 3.1*
Ecrire le XMLSchema de ce document (on considérera que `from`, `to`, `title` et `text` sont de type `xsd:string`).

## 3.a Le texte du message

On se propose dans un premier temps d'extraire uniqument le texte du mémo (`Ca va ?`) sous forme d'une String.

Pour cela, on gardera une trace pour savoir si le parser a atteint l'élément `text` sous forme d'un booléen `inMemoText`. Lorsque l'on est bien dans le texte du mémo et que l'on atteint les caractères, on stocke le texte dans une chaine `memoText`.

On considère la classe `MemoGetText` et la classe `Test` suivantes:

```
package saxParser;

import org.xml.sax.Attributes;
import org.xml.sax.helpers.DefaultHandler;

public class MemoGetText extends DefaultHandler {
    private boolean inMemoText;
    private String  memoText;

    public void startDocument() {
        inMemoText = ___(1)____;
        memoText   = ___(2)____;
    }

    public void startElement(String namespaceURI, String localName, String
        qName, Attributes atts) {
        if (___(3)____) {
            ___(4)____
        }
    }

    public void endElement(String namespaceURI, String localName, String qName)
        {
        ___(5)____
    }

    public void characters(char[] ch, int start, int length) {
        String text = "";
        for (int i = start; i < start+length; i++) {
            text += ch[i];
```

```
        }
        if (___(6)____) {
            ___(7)____
        }
    }

    public String getMemoText() {
        return memoText;
    }

}
```

```

package saxParser;

import org.xml.sax.XMLReader;
import org.xml.sax.helpers.XMLReaderFactory;

public class TestMemo {

    public static void main(String[] args) {
        String filename = "memo.xml";
        try {
            XMLReader parser;
            parser = XMLReaderFactory.createXMLReader();
            MemoGetText memo = new MemoGetText();
            parser.setContentHandler(memo);
            parser.parse(filename);
            System.out.println("Le message du mémo est: " + memo.getMemoText())
                ;
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

*Question 3.2*
Compléter la classe `MemoGetText`.

## 3.b Mise en forme du mémo

On souhaite à présent recueillir toutes les informations du mémo et les mettre en forme (en français) sous la forme suivante:

```

---------------------- Memo ----------------------
De: John
A:Bob

[Bonjour]
Ca va ?

------------------- Fin de Memo --------------------
```

Pour cela, on va énumérer les états possibles de l'automate correspondant au *parser*, c'est-à-dire les différents éléments du document:

- `START`
- `MEMO`
- `FROM`

- `TO`
- `CONTENT`
- `SUBJECT`
- `TEXT`

A chaque état, il faut déterminer ce qui doit être fait lorsque l'on entre dans un nouvel élément, lorsque l'on en sort et lorsque l'on lit le texte.

*Question 3.3*
Dessiner le diagramme de séquence de l'application en fonction des états cités précédemment. Dans notre cas, il s'agit de l'ordre dans lequel les états (`START`, `MEMO`, `FROM`, `TO`, `CONTENT`, `SUBJECT`, `TEXT`) qui doivent être déclenchés dans les *callbacks* `startElement` et `endElement`.

*Question 3.4*
Dans le *callback* `beginElement()`, dans quel(s) état(s) l'application devra-t-elle écrire (dans un attribut de type chaîne de caractère):
a) `----------` `Memo` `----------` ?
b) `De:` ?
c) `A:` ?
d) `[` ?

*Question 3.5*
Dans le *callback* `ENDElement()`, dans quel(s) état(s) l'application devra-t-elle écrire (dans un attribut de type chaîne de caractère):
a) *'retour à la ligne'* ?
b) `]` ?
c) `----------` `Fin de Memo` `----------` ?

*Question 3.6*
Dans quels états doit-on écrire le texte qui est à l'intérieur des balises ?

On considère la classe `MemoMiseEnForme` et la classe `TestMemoMisEnForme` suivantes:

```
1
2  package saxParser;
3
4  import org.xml.sax.Attributes;
5  import org.xml.sax.helpers.DefaultHandler;
6
7  public class MemoMiseEnForme extends DefaultHandler {
8      // Enumération: on repère chaque état qui nous intéresse par
9      //  un entier (pour autoriser un branchement switch)
10
11     final int START = 0,
12             MEMO = 1,
13             FROM = 2,
14             TO = 3,
15             CONTENT = 4,
16             SUBJECT = 5,
17             TEXT = 6,
18             OTHER = 7;
19
20     private int state;
21     private String memoMisEnForme;
22
23     public void startDocument() {
24         state = ___(1)____;
```

```
25          memoMisEnForme = ___(2)____;
26      }
27
28      public void startElement(String namespaceURI, String localName, String
            qName, Attributes atts) {
29          switch (state) {
30              case START:
31                  // Si l'on est dans Start, on ne peut que avoir MEMO ou un
                        commentaire
32                  if (___(3)____) {
33                      state = ___(4)____;
34                      memoMisEnForme += ___(5)____;
35                  }
36                  break;
37              case MEMO:
38                  if (___(6)____) {
39                      state = ___(7)____;
40                      memoMisEnForme += ___(8)____;
41                  }
42
43                  if (___(9)____) {
44                      state = ___(10)____;
45                      memoMisEnForme += ___(11)____;
46                  }
47
48                  if (___(12)____) {
49                      state = ___(13)____;
50                      memoMisEnForme += ___(14)____;
51                  }
52
53                  break;
54
55              case CONTENT:
56                  if (___(15)____) {
57                      state = ___(16)____;
58                      memoMisEnForme += ___(17)____;
59                  }
60                  if (___(18)____) {
61                      state = ___(19)____;
62                  }
63
64              default:
65                  // Dans tous les autres cas, on ne fait rien
66                  break;
67          }
68      }
69
70      public void endElement(String namespaceURI, String localName, String qName)
            {
71          switch (state) {
72              // On n'a pas d'élément dans START...
73              case MEMO:
74                  memoMisEnForme += ___(20)____;
75                  state = ___(21)____;
76                  break;
77              case FROM:
78                  memoMisEnForme +=___(22)____;
79                  state = ___(23)____;
80                  break;
81              case TO:
82                  memoMisEnForme += ___(24)____;
83                  state = ___(25)____;
84                  break;
```

```
85              case CONTENT:
86                  memoMisEnForme += ___(26)____;
87                  state = ___(27)____;
88                  break;
89              case SUBJECT:
90                  memoMisEnForme += ___(28)____;
91                  state = ___(29)____;
92                  break;
93              case TEXT:
94                  memoMisEnForme += ___(30)____;
95                  state = ___(31)____;
96              default:
97                  // Dans tous les autres cas, on ne fait rien
98                  break;
99          }
100     }
101
102     public void characters(char[] ch, int start, int length) {
103         String text = "";
104         for (int i = start; i < start + length; i++) {
105             text += ch[i];
106         }
107         switch (state) {
108             case ___(32)____:
109             case ___(33)____:
110             case ___(34)____:
111             case ___(35)____:
112                 memoMisEnForme += ___(36)____;
113                 break;
114             default:
115                 // Sinon, on ne fait rien
116                 break;
117         }
118     }
119
120     public String getMemoText() {
121         return memoMisEnForme;
122     }
123 }
```

```
1
2  package saxParser;
3
4  import org.xml.sax.XMLReader;
5  import org.xml.sax.helpers.XMLReaderFactory;
6
7  public class TestMemoMisEnForme {
8
9      public static void main(String[] args) {
10         String filename = "memo.xml";
11         try {
12             XMLReader parser;
13             parser = XMLReaderFactory.createXMLReader();
14             MemoMiseEnForme memo = new MemoMiseEnForme();
15             parser.setContentHandler(memo);
16             parser.parse(filename);
17             System.out.println(memo.getMemoText());
18         } catch (Exception e) {
19             System.out.println(e);
20         }
21     }
22 }
```

*Question 3.7*
Compléter le code de la classe `MemoMiseEnForme`.

# 4 StAX: Lecture d'un Mémo

On souhaite mettre en forme le document suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<memo xmlns="http://ujf-grenoble.fr/memo"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://ujf-grenoble.fr/memo memo.xsd">

    <from>John</from>
    <to>Bob</to>
    <content>
        <subject>Bonjour</subject>
        <text>Ca va ?</text>
    </content>
</memo>
```

*Question 4.1*
Ecrire le XMLSchema de ce document (on considérera que `from`, `to`, `title` et `text` sont de type `xsd:string`).

## 4.a Le texte du message

On se propose dans un premier temps d'extraire uniqument le texte du mémo (`Ca va ?`) sous forme d'une String.

Pour cela, on gardera une trace pour savoir si le parser a atteint l'élément `text` sous forme d'un booléen `inMemoText`. Lorsque l'on est bien dans le texte du mémo et que l'on atteint les caractères, on stocke le texte dans une chaine `memoText`.

On considère la classe la classe `TestMemo` suivante:

```
package staxParser;

import java.io.FileInputStream;
import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamConstants;
import javax.xml.stream.XMLStreamReader;

public class TestMemo {

    public static void main(String[] args) {
        String filename = "memo.xml";

        boolean inMemoText = ___(1)____;
        String memoText = ___(2)____;

        try {
            // instanciation du StAX reader
            XMLInputFactory inputFactory = XMLInputFactory.newInstance();
            // ici le document en entrée est un fichier, on le lit sous forme
                de stream
            XMLStreamReader reader = inputFactory.createXMLStreamReader(new
                FileInputStream(filename), "UTF-8");

            // lecture du fichier tant qu'il y a encore des événèments (i.e des
                 choses à lire)
            while (reader.hasNext()) {
```

```
                // récupérer le prochain (pull)
                int event = reader.next();

                switch (event) {
                     case XMLStreamConstants.START_ELEMENT:
                         String elementName = ___(3)____;
                         if (___(4)____) {
                             inMemoText = ___(5)____;
                         }
                        break;
                     case XMLStreamConstants.END_ELEMENT:
                         inMemoText = ___(6)____;
                         break;
                     case XMLStreamConstants.CHARACTERS:
                         if (___(7)____) {
                             memoText = ___(8)____;
                         }
                         break;

                    default:
                        // Dans tous les autres cas, on ne fait rien
                        break;
                }
            }
        } catch (Exception e) {
            System.out.println(e);
        }

        System.out.println("Le message du mémo est: " + memoText);

    }
}
```

*Question 4.2*
Compléter la classe `MemoGetText`.

## 4.b Mise en forme du mémo

On souhaite à présent recueillir toutes les informations du mémo et les mettre en forme (en français) sous la forme suivante:

```

---------------------- Memo ----------------------
De: John
A:Bob

[Bonjour]
Ca va ?

------------------- Fin de Memo -------------------
```

Pour cela, on va énumérer les états possibles de l'automate correspondant au *parser*, c'est-à-dire les différents éléments du document:

- `MEMO`
- `FROM`
- `TO`
- `CONTENT`
- `SUBJECT`

- `TEXT`

A chaque état, il faut déterminer ce qui doit être fait lorsque l'on entre dans un nouvel élément, lorsque l'on en sort et lorsque l'on lit le texte.

*Question 4.3*
Dessiner le diagramme de séquence de l'application en fonction des états cités précédemment. Dans notre cas, il s'agit de l'ordre dans lequel les états (`MEMO`, `FROM`, `TO`, `CONTENT`, `SUBJECT`, `TEXT`) qui doivent être déclenchés dans les *callbacks* `startElement` et `endElement`.

*Question 4.4*
Dans le cas `BEGIN_ELEMENT`, dans quel(s) état(s) l'application devra-t-elle écrire (dans un attribut de type chaîne de caractère):
a) `---------- Memo ----------` ?
b) `De:` ?
c) `A:` ?
d) `[` ?

*Question 4.5*
Dans le cas `END_ELEMENT`, dans quel(s) état(s) l'application devra-t-elle écrire (dans un attribut de type chaîne de caractère):
a) *'retour à la ligne'* ?
b) `]` ?
c) `---------- Fin de Memo ----------` ?

*Question 4.6*
Dans quels états doit-on écrire le texte qui est à l'intérieur des balises ?

On considère la classe `TestMemoMisEnForme` suivante:

```
1
2  package staxParser;
3
4  import java.io.FileInputStream;
5  import javax.xml.stream.XMLInputFactory;
6  import javax.xml.stream.XMLStreamConstants;
7  import javax.xml.stream.XMLStreamReader;
8
9  public class TestMemoMisEnForme {
10
11     public static void main(String[] args) {
12         String filename = "memo.xml";
13
14         // Enumération: on repère chaque état qui nous intéresse par
15         //  un entier (pour autoriser un branchement switch)
16
17         final int MEMO = 1,
18                   FROM = 2,
19                   TO = 3,
20                   CONTENT = 4,
21                   SUBJECT = 5,
22                   TEXT = 6,
23                   OTHER = 7;
24
25         int state = START;
26         String memoMisEnForme = "";
27
28         try {
29             // instanciation du StAX reader
30             XMLInputFactory inputFactory = XMLInputFactory.newInstance();
```

```
31              // ici le document en entrée est un fichier, on le lit sous forme
                    de stream
32              XMLStreamReader reader = inputFactory.createXMLStreamReader(new
                    FileInputStream(filename), "UTF-8");
33
34              // lecture du fichier tant qu'il y a encore des événèments (i.e des
                     choses à lire)
35              while (reader.hasNext()) {
36                  // récupérer le prochain (pull)
37                  int event = reader.next();
38                  String elementName;
39                  switch (event) {
40                      case XMLStreamConstants.START_ELEMENT:
41                          elementName = ___(1)____;
42                          switch (state) {
43                              case START:
44                                  // Si l'on est dans Start, on ne peut que avoir
                                         MEMO ou un commentaire
45                                  if (___(2)____) {
46                                      state = ___(3)____;
47                                      memoMisEnForme += ___(4)____;
48                                  }
49                                  break;
50                              case MEMO:
51                                  if (elementName.equals(___(5)____)) {
52                                      state = ___(6)____;
53                                      memoMisEnForme += ___(7)____;
54                                  }
55
56                                  if (elementName.equals(___(8)____)) {
57                                      state = ___(9)____;
58                                      memoMisEnForme += ___(10)____;
59                                  }
60
61                                  if (elementName.equals(___(11)____)) {
62                                      state = ___(12)____;
63                                      memoMisEnForme += ___(13)____;
64                                  }
65
66                                  break;
67
68                              case CONTENT:
69                                  if (elementName.equals(___(14)____)) {
70                                      state = ___(15)____;
71                                      memoMisEnForme += ___(16)____;
72                                  }
73                                  if (elementName.equals(___(17)____)) {
74                                      state = ___(18)____;
75                                  }
76
77                              default:
78                                  // Dans tous les autres cas, on ne fait rien
79                                  break;
80                          }
81                          break;
82                      case XMLStreamConstants.END_ELEMENT:
83                          elementName = reader.getLocalName();
84                          switch (state) {
85                              case MEMO:
86                                  if (elementName.equals(___(19)____)) {
87                                      memoMisEnForme += ___(20)____;
88                                  }
89                                  state = ___(21)____;
```

```
90                                  break;
91                              case FROM:
92                                  memoMisEnForme += ___(22)____;
93                                  state = ___(23)____;
94                                  break;
95                              case TO:
96                                  memoMisEnForme += ___(24)____;
97                                  state = ___(25)____;
98                                  break;
99                              case CONTENT:
100                                 if (elementName.equals(___(26)____)) {
101                                     memoMisEnForme += ___(27)____;
102                                 }
103                                 state = ___(28)____;
104                                 break;
105                             case SUBJECT:
106                                 memoMisEnForme += ___(29)____;
107                                 state = ___(30)____;
108                                 break;
109                             case TEXT:
110                                 memoMisEnForme += ___(31)____;
111                                 state = ___(32)____;
112                             default:
113                                 // Dans tous les autres cas, on ne fait rien
114                                 break;
115                         }
116                         break;
117                     case XMLStreamConstants.CHARACTERS:
118                         switch (state) {
119                             case ___(33)____:
120                             case ___(34)____:
121                             case ___(35)____T:
122                             case ___(36)____:
123                                 String text = ___(37)____;
124                                 memoMisEnForme += ___(38)____;
125                                 break;
126                             default:
127                                 // Sinon, on ne fait rien
128                                 break;
129                         }
130                         break;
131
132                     default:
133                         // Dans tous les autres cas, on ne fait rien
134                         break;
135                 }
136             }
137         } catch (Exception e) {
138             System.out.println(e);
139         }
140
141         System.out.println(memoMisEnForme);
142
143     }
144 }
```

*Question 4.7*
Compléter le code de la classe `MemoMiseEnForme`.

# 5 SAX: Annuaire

On considère le carnet d'adresses sous forme XML suivant:

```
1
2  <?xml version="1.0" encoding="UTF-8"?>
3  <?xml version="1.0" encoding="UTF-8"?>
4  <annuaire xmlns="http://ujf-grenoble.fr/adresses"
5            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6            xsi:schemaLocation="http://ujf-grenoble.fr/adresses adresses.xsd">
7      <personne id="3229">
8          <nom>Covert</nom>
9          <prenom>Harry</prenom>
10         <adresse>Allée du Pote Agé, 38000 Grenoble</adresse>
11     </personne>
12     <personne id="3230">
13         <nom>Deblouze</nom>
14         <prenom>Agate</prenom>
15         <adresse>Rue des Tresses, 42100 Saint Etienne</adresse>
16     </personne>
17 </annuaire>
```

*Question 5.1*
Ecrire le XMLSchema de ce document.

On considère la classe `Personne` suivante:

```
1
2  package saxParser;
3
4  public class Personne {
5      private int id;
6      private String nom;
7      private String prenom;
8      private String adresse;
9
10     public Personne() {
11         id = 0;
12         nom = "Doe";
13         prenom = "Jhon";
14         adresse = "60 Rue de la Chimie, 38400 Saint-Martin-d'Hères";
15     }
16
17     public void setId(int id) {
18         this.id = id;
19     }
20
21     public void setNom(String nom) {
22         this.nom = nom;
23     }
24
25     public void setPrenom(String prenom) {
26         this.prenom = prenom;
27     }
28
29     public void setAdresse(String adresse) {
30         this.adresse = adresse;
31     }
32
33     public String toString() {
34         String res;
35         res  = "Contact No " + id + "\n";
36         res += "Nom: " + nom + "\n";
37         res += "Prénom: " + prenom + "\n";
38         res += "Adresse: " + adresse + "\n";
39
```

```
40          return res;
41      }
42
43
44  }
```

On souhaite lire le document XML et stocker les informations dans une liste de `Personnes`. Pour celan, on considère les états suivants:

- `START`
- `ANNUAIRE`
- `PERSONNE`
- `NOM`
- `PRENOM`
- `ADRESSE`
- `OTHER`

*Question 5.2*
Ecrire la classe `PersonneHandler` qui hérite de `DefaultHandler` et qui a comme attributs `state: int`, `personneCourante: Personne`, `annuaire: ArrayList<Personne>` et qui rempli l'attribut `annuaire` avec des `Personne` mises à jours avec les données XML.

# 6 StAX: Annuaire

On considère le carnet d'adresses sous forme XML suivant:

```
1
2   <?xml version="1.0" encoding="UTF-8"?>
3   <?xml version="1.0" encoding="UTF-8"?>
4   <annuaire xmlns="http://ujf-grenoble.fr/adresses"
5             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6             xsi:schemaLocation="http://ujf-grenoble.fr/adresses adresses.xsd">
7       <personne id="3229">
8           <nom>Covert</nom>
9           <prenom>Harry</prenom>
10          <adresse>Allée du Pote Agé, 38000 Grenoble</adresse>
11      </personne>
12      <personne id="3230">
13          <nom>Deblouze</nom>
14          <prenom>Agate</prenom>
15          <adresse>Rue des Tresses, 42100 Saint Etienne</adresse>
16      </personne>
17  </annuaire>
```

*Question 6.1*
Ecrire le XMLSchema de ce document.

On considère la classe `Personne suivante:`

```
1
2   package saxParser;
3
4   public class Personne {
5       private int id;
6       private String nom;
7       private String prenom;
8       private String adresse;
```

```
9
10      public Personne() {
11          id = 0;
12          nom = "Doe";
13          prenom = "Jhon";
14          adresse = "60 Rue de la Chimie, 38400 Saint-Martin-d'Hères";
15      }
16
17      public void setId(int id) {
18          this.id = id;
19      }
20
21      public void setNom(String nom) {
22          this.nom = nom;
23      }
24
25      public void setPrenom(String prenom) {
26          this.prenom = prenom;
27      }
28
29      public void setAdresse(String adresse) {
30          this.adresse = adresse;
31      }
32
33      public String toString() {
34          String res;
35          res  = "Contact No " + id + "\n";
36          res += "Nom: " + nom + "\n";
37          res += "Prénom: " + prenom + "\n";
38          res += "Adresse: " + adresse + "\n";
39
40          return res;
41      }
42
43
44  }
```

On souhaite lire le document XML et stocker les informations dans une liste de `Personnes`. Pour cela, on considère la classe `Annuaire` suivante:

```
1
2   package staxParser;
3
4   import java.io.FileInputStream;
5   import java.util.ArrayList;
6   import javax.xml.stream.XMLInputFactory;
7   import javax.xml.stream.XMLStreamConstants;
8   import javax.xml.stream.XMLStreamReader;
9
10  public class Annuaire {
11
12      final protected int START = 0,
13              ANNUAIRE = 1,
14              PERSONNE = 2,
15              NOM = 3,
16              PRENOM = 4,
17              ADRESSE = 5,
18              OTHER = 6;
19      private ArrayList<Personne> liste;
20
21      public Annuaire(String filename) {
22          liste = new ArrayList<Personne>();
23          parseXMLFile(filename);
24      }
```

```
25
26      private void parseXMLFile(String filename) {
27          int state = START;
28          Personne personneCourante = null;
29          String localName = "";
30
31          try {
32              // instanciation du StAX reader
33              XMLInputFactory inputFactory = XMLInputFactory.newInstance();
34              // ici le document en entrée est un fichier, on le lit sous forme
                    de stream
35              XMLStreamReader reader = inputFactory.createXMLStreamReader(new
                    FileInputStream(filename), "UTF-8");
36
37              // lecture du fichier tant qu'il y a encore des événèments (i.e des
                     choses à lire)
38              while (reader.hasNext()) {
39                  // récupérer le prochain (pull)
40                  int event = reader.next();
41
42                  switch (event) {
43                      // A compléter
44                  }
45              }
46          } catch (Exception e) {
47              System.out.println(e);
48          }
49
50      }
51
52      @Override
53      public String toString() {
54          String res = "------------- Annuaire -----------------\n";
55          for (Personne p : liste) {
56              res += p + "\n";
57          }
58          return res;
59      }
60  }
```

*Question 6.2*
Compléter la classe `Annuaire`.

*Chapitre X*

# Heritage et Sérialisation

## 1 Modéliser une matrice - héritage♣

Une matrice, en mathématiques, est un tableau contenant des coefficients. Un matrice est de taille $nxm$. Un exemple de matrice $3 \times 3$ est donné ci-dessous : $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

On cherche à modéliser une matrice sous la forme d'un schéma XML. On considère qu'une matrice (`Matrix`) est composée de plusieurs lignes (qu'on nommera `Row`. Le nombre de lignes n'est pas limité. Chaque ligne contient un nombre illimité de cases (nommées `Cell`) accueillant des coefficients de type `xs:double`. Chaque case (`Cell`) de la matrice est numérotée à l'aide d'un attribut (un entier) qu'on désignera par `cellNo`. De même, chaque ligne (`Row`) de la matrice est identifiée par un attribut (un entier aussi) qu'on notera `rowNo`.

Nous préfixerons tous les types de ce schéma par `nw`.

*Question 1.1*
Le type `Cell` est il un type simple ou un type complexe ? Même question pour le type `Row` et pour le type `Matrix`.

*Question 1.2*
Modélisez le type `Cell` comme héritant d'un type simple.

*Question 1.3*
Modélisez le type `Row`

*Question 1.4*
Modélisez le type `Matrix`

## 2 Modéliser une matrice - clefs uniques♣

On souhaite maintenant faire en sorte que les numéros de case et de ligne (`cellNo` et `rowNo`) soient uniques : pour chaque ligne, la valeur de l'attribut `cellNo` ne peut pas être répétée ; pour chaque matrice, la valeur de l'attribut `rowNo` ne peut pas être répétée.

*Question 2.1*
A quel niveau doit on contraindre le schéma pour forcer l'unicité des valeurs de l'attribut `cellNo`.

*Question 2.2*
Modifier le schéma de manière à ce que les valeurs de l'attribut `cellNo` soient uniques dans chaque ligne de la matrice.

*Question 2.3*
Le préfixe associé au namespace est il important dans les chemins xpath du selecteur et du champs cible ?

*Question 2.4*
Comment spécifier maintenant une clef unique pour l'attribut `rowNo` ? (à quel niveau doit-elle être déclarée ?)

## 3 Serialisation la matrice en Java♣

*Question 3.1*
Quel est le principe de la sérialisation ? Quelles sont les opérations nécessaires ?

*Question 3.2*
Proposer une ou des classe(s) Java correspondante(s) aux types XMLSchema que vous avez écrits.

*Question 3.3*
Est-ce une bonne idée d'ajouter des méthodes à ces classes qui permettent une sérialisation des données ? Pourquoi ?

*Question 3.4*
Proposer une façon de faire des opérations sur une ou des matrices. Proposez un exemple en Java.