

**Evaluation 2017-2018**

Filière : Télécommunications

Classe : T2

Cours : SI1

Date : Juin 2018

Temps : 15h00 à 16h35. Pour ceux qui ont droit au temps supplémentaire : 15h00 à 17h04

Examinatrice : H. Chabbi Drissi

Nom et prénom : Zambrani Yaniel

Note :

Note max = 6

---

**Toutes les réponses sont à reporter sur l'énoncé**

---

Aucun document n'est autorisé mis à part les quick reference disponibles sur le site du cours dans l'état sans aucun ajout de votre part.

Toute réponse doit être pleinement justifiée.

Le tableau ci-dessous vous indique les points accordés à chaque partie. Le temps estimé y est donné à titre indicatif.

	Points max	Temps estimés	Points obtenus
Lecture énoncé	-	5min	-
Exercice 1	1	10min	0.8
Exercice 2	0.6 + 0.2	15min	0.65
Exercice 3	1	15min	1
Exercice 4	0.8	10min	0.8
Exercice 5	1.1	25min	0.9
Exercice 6	0.5	15min	0.5
<b>Total</b>	<b>5.2</b>	<b>95min</b>	<b>4.65</b>

**Convention :** Dans la suite, chaque V = 0.1pt

## Exercice 1 Connaissances théoriques

Pour chacune des questions suivantes :

- entourer les réponses correctes. V par bonne réponse
- ou compléter les phrases (là où il est indiqué : ..... ) VV par bonne réponse

1. Dans le TP concernant jdbc, vous avez complété une application java qui se connecte à un mysql. Que faut-il changer précisément dans ce code pour que l'application s'exécute correctement mais en accédant à votre compte oracle de l'école et non plus au mysql. *Il faut changer le driver ainsi que l'adresse du mysql pour la changer vers celle du compte oracle.* ✓✓

2. Il est impossible d'écrire une seule application qui se connecte en même temps sur deux SGBDs qui ne soient pas de la même famille (2 oracles ou 2 mysqls par exemple) : VRAI ou FAUX ? X

3. Un fichier xml valide est forcément bien formé : VRAI ou FAUX ? X

4. Soit le fichier madtd.dtd qui contient la DTD suivante :

```
1 <!-- DTD -->
2 <!ELEMENT root (a)
3 <!ELEMENT a (#PCDATA)>
```

(a) Peut-on utiliser cette DTD pour valider avec, le xml suivant : VRAI ou FAUX ? ✓

```
1 <?xml version="1.0"?>
2 <a/>
```

(b) Soit le xml suivant :

```
1 <?xml version="1.0"?>
2 <root>
3   <a/>
4 </root>
```

Que faire pour déclarer la DTD précédente comme DTD externe à ce fichier xml

*<!DOCTYPE root SYSTEM "madtd.dtd">*

5. On peut imposer via une DTD qu'un fichier xml ne puisse pas contenir de commentaires xml (donc pas de <!--...-->) : VRAI ou FAUX ? ✓

6. Un fichier xml contenant du texte mélangé avec des balises est dit de type -centric

*→ document - centric*

*[Scribbles]*

## Exercice 2 XSD et DTD

Soit la XSD : Conferences.xsd donnée en annexe :

1. Donner l'instance **minimale** de XML qui soit validée par cette XSD. VV

```
<?xml version="1.0"?>
<Conferences>
  <Titre>
    Auteur
    <A>
    </A>
  </Titre>
</Conferences>
```

V X

2. Donner une **instance de XML avec le nombre maximums d'apparition** d'éléments et de texte qui soit validée par cette XSD. Le texte sera systématiquement remplacé par trois petits points ... VV

```
<?xml version="1.0"?>
<Conferences>
  ...
  <Titre>
    Auteur
    <A> ... </A>
    Auteur
    <T> ... </T>
  </Titre>
  ...
  <Titre>
    Auteur
    <A> ... </A>
    <A> ... </A>
  </Titre>
</Conferences>
```

V

m even.

3. Proposer une DTD qui valide votre 2ième instance XML VV

<! Element Conferences (#PCData | title)\* >

<! Element title ( ~~A~~ A, (A|T)? ) >

<! Element A (#PCData) >

<! Element T (#PCData) >

VV

4. **Bonus** Proposer une instance XML minimale valide par rapport à votre DTD mais pas par rapport à la XSD. VV

< Conferences >

< title >

< A > < / A >

< / title >

< title >

< A > < / A >

< / title >

< title >

< A > < / A >

< title >

< / Conferences >

VV

Voilà même en étant plus expéditif: < Conferences >  
< / Conferences >

### Exercice 3 Modélisation

Compléter la DTD lacunaire donnée ci-dessous et qui doit répondre à la demande suivante. 1pt

On veut écrire des instances xml, pour le livret des projets de semestre d'une filière. Ce livret

— présente d'abord la liste des projets (non vide) puis la liste des étudiants (qui peut-être vide)

Attribut Livret : — on doit connaître la filière concerné par ce livret. Par défaut, cette information vaut télécom. On peut éventuellement connaître le semestre qui ne peut alors être que "SP" ou "SA".

Pour chaque projet on doit connaître :

élément — Son titre

— Une description qui est structurée en paragraphe. Une description contient au moins un paragraphe. Chaque paragraphe contient du texte dont des mots peuvent être mis entre les tags <Keyword>...</Keyword>.

attribut — un acronyme qui lui est unique

donnée — Éventuellement le nom et prénom du mandant ou éventuellement le nom de l'entreprise cliente mais pas les deux en même temps.

Pour chaque étudiant on doit connaître :

— son nom

— son prénom

— éventuellement le projet qui lui est affecté.

```

1  <!-- La liste complète de tous les éléments de type PCDATA dans cette DTD
   se trouvent ci-dessous-->
2  <!ELEMENT titre (#PCDATA)>
3  <!ELEMENT Keyword (#PCDATA)>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT nomEntreprise (#PCDATA)>
6  <!ELEMENT prenom (#PCDATA)>
7      <!-- A partir de là il faut compléter la DTD -->
8  <!-- Élément racine -->
9  <!ELEMENT Livret (Lprojets, Letudiants)>
10
11  <!-- Element Lprojets (projet+) > ✓
12  <!-- Element Letudiants (etudiant*) > ✓
13
14  <!-- Element projet (titre, description, ((prenom, nom)| nomEntreprise)? ) > ✓
15  <!ELEMENT description (p+)> ✓
16
17  <!ELEMENT p (#PCDATA, Keyword)*> ✓
18
19  <!-- ATT LIST projet acronyme ID # Required > ✓
20  <!-- ATT LIST livret filière CDATA "télécom" > ✓
21  <!-- ATT LIST livret semestre (SP|SA) # IMPLIED > ✓

```

```

22 <!-- Élément Lprojets -->
23 <!--ELEMENT Lprojets
24
25 <| ELEMENT étudiant (nom, prenom) > ✓
26
27 <| ATTLIST étudiant projet IDREF #IMPLIED> ✓
28
29
30
31
32
33
34
35
36
37
38
39
40 <!-- Élément Letudiants -->
41 <!--ELEMENT Letudiants
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

```

## Exercice 4 XPATH

On vous demande pour chaque XPATH suivant, de donner les numéros de lignes de tout ce qui va être retourné à chaque fois, avec la sémantique suivante :

2;6;9 indique la ligne 2 + la ligne 6 + la ligne 9

2-6;9 indique toutes les lignes entre la ligne 2 et la ligne 6 + la ligne 9

RIEN: indique qu'aucune ligne n'est retournée.

ERREUR: indique que cet XPATH est syntaxiquement faux

Ces XPATH sont à exécuter sur le fichier librairie.xml donné en annexe. 0.2pt par XPATH

1 /livre[@categorie="CUISINE"]

2

3 ~~XXXXXXXXXX~~

4

5 Rien → pas d'enfant direct livre

6

7

8 ✓✓

9

10

11

12

13

14

15 /librairie/livre[@categorie="CUISINE" and @lang="en"]

16

17 Rien (livre n'a pas d'attribut lang)

18

19

20

21

22 ✓✓

23

24

25

26

27

28

29 //livre/titre[@lang="en"]/../prix/..

30

31 3-8 ; 9-14 ; 15-20 ✓✓

32

33

34

35

36

37

38

39

40

41

42 /librairie/livre/@categorie[.="CUISINE"]/..

43

44

45

46

47

48

49

50

51

52

53

54

55

3-8, 31-36

✓

## Exercice 5 Exécuter une XSL

Soit la XSL : Transformation.xsl donnée en annexe :

1. A quel(s) noeud(s) précisément s'applique la règle 2. V

A tous les enfants de librairie, donc tous les livres.

X

2. A quel(s) noeud(s) précisément s'applique la règle 3. V

Aux livres qui n'ont pas de fils "prix" ✓



3. Donner le résultat de l'exécution de cet XSL sur elle-même. Si vous pensez que cela résultera en une erreur écrire : **Pas Possible** sinon donnez le résultat en **soignant la lisibilité** sinon la solution sera considérée comme fausse! **VV**

W

```

<xsl:stylesheet?
  Nombres magiques: (0,0)
</xsl:stylesheet?
  styleheet
  
```

même chose répétée 3 fois de plus

4. Donner le résultat de l'exécution de cet XSL sur le fichier `librairie.xml` donné en annexe. **Soigner la lisibilité** sinon la solution sera considérée comme fausse! **0.6pt**

V < librairie >

V Nombres magiques : ( 6 , 3 )

< livre prix = "12.00" titre = "Un bol... un plat" >

~~XXXXXXXXXXXXXXXXXXXX~~

Cuisine

< /livre >

V < livre prix = "30.00" titre = "Everyday Italian" >

Cuisine

< /livre >

V < livre prix = "27.50" titre = "The Philosopher's Stone" >

ENFANTS

< /livre >

V < livre prix = "39.95" titre = "Learning XML" >

WEB

< /livre >

< /librairie

5. On remplace la règle 1 ainsi :

```
1 <!-- Règle 1 -->
2 <xsl:template match="/" />
```

Donner le résultat de l'exécution de la nouvelle version du XSL sur le XML précédent. Si vous pensez que cela résultera en une erreur écrire **Pas Possible**. Si cela devait donner la même chose que le résultat obtenu avec l'exécution précédente écrire **Idem précédent résultat**. Si vous pensez que cela donne autre chose écrire **Nouvelle exécution** et donner le résultat en soignant la lisibilité. V

Idem précédent résultat

X

## Exercice 6 Ecrire une XSL

Ecrire une XSL qui à partir d'un fichier XML ayant la même structure que le fichier librairie.xml, donne en sortie un fichier XML qui lui est identique en structure mais qui ne contient que ses livres de la catégorie WEB. Compléter le code ci-dessous. On vous demande de commenter chaque règle et XPATH que vous écrirez dans cette XSL pour en indiquer l'utilité. 0.5pt

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
3
4   <xsl:template match="/" >          <!-- selection de la racine pour appliquer le template
5                                     par défaut -->
6     <xsl:apply-templates />
7   </xsl:template >
8
9   <xsl:template match="#librairie" > <!-- on écrit les balises de librairie quand on l'attend -->
10    <xsl:copy>
11      <xsl:apply-templates select="/* librairie " />
12      <!-- affichage de tous les enfants de librairie (=livre) avec l'attribut
13                                     catégorie à la valeur 'WEB' -->
14      <xsl:copy>
15        <xsl:apply-templates select="/* librairie " />
16      </xsl:copy>
17    </xsl:template>
```

géné en dessous en dessous.

```

16 <xsl:template match = "livre [@catégorie != "WEB"]" />
17
18   <!-- les livres non-concernés ne sont pas affichés -->
19
20 <xsl:template match = "livre [@catégorie = "WEB"] >
21   <!-- affichage des livres de la catégorie WEB -->
22
23   <xsl:copy-of select = "." />
24
25 </xsl:template>
26
27   <!-- on recopie le noeud
28   Courant et ses enfants
29   (donc les livres WEB
30   et ses enfants) -->
31 </xsl:stylesheet>

```