

Fiche XML n°6

Explorer un document XML avec XPATH

Éléments de correction pour les exercices

Soit la DTD «*edition.dtd*» et le document «*jailu.xml*»

```
<!ELEMENT edition (auteur+, livre+)>
<!ATTLIST edition nom CDATA #REQUIRED>
```

```
<!ELEMENT auteur EMPTY>
<!ATTLIST auteur
  nom CDATA #REQUIRED
  email CDATA #IMPLIED
  no ID #REQUIRED
  pays CDATA #IMPLIED>
```

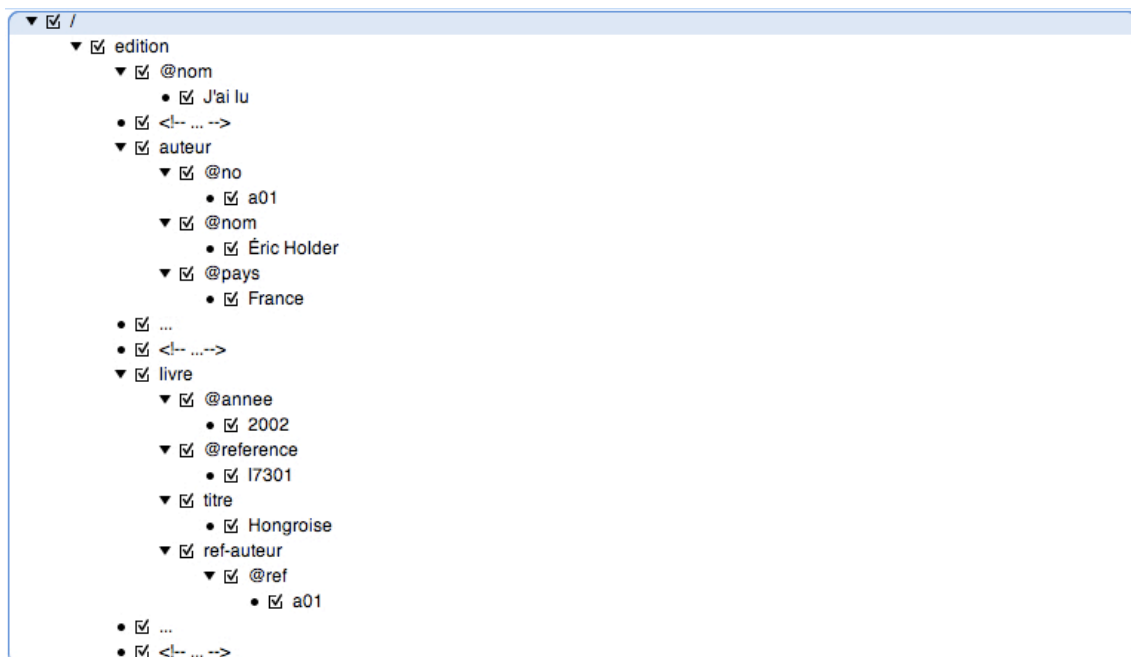
```
<!ELEMENT livre (titre, ref-auteur)>
<!ATTLIST livre
  reference ID #REQUIRED
  annee CDATA #REQUIRED>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT ref-auteur EMPTY>
<!ATTLIST ref-auteur ref IDREF #REQUIRED>
```

Exercice 1

Question 1. Construire l'arbre XML résolu pour le fichier ci-contre.

Solution :

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE edition SYSTEM "edition.dtd" [
  <!ENTITY fr "France">
  <!ENTITY usa "États Unis">
  <!ENTITY rs "Russie">
]>
<edition nom="J'ai lu">
  <!-- Les auteurs -->
  <auteur no="a01" nom="Éric Holder" pays="&fr;" />
  <auteur no="a02" nom="Winston Groom" pays="&usa;" />
  <auteur no="a03" nom="Stephen King" pays="&usa;" />
  <auteur no="a04" nom="Isaac Asimov" pays="&rs;" />
  <!-- Les ouvrages -->
  <livre annee="2002" reference="I7301">
    <titre>Hongroise</titre><ref-auteur ref="a01"/>
  </livre>
  <livre annee="1986" reference="I3816">
    <titre>Forest Gump</titre><ref-auteur ref="a02"/>
  </livre>
  <livre annee="1996" reference="I5157">
    <titre>La ligne verte</titre><ref-auteur ref="a03"/>
  </livre>
  <livre annee="1983" reference="I2266">
    <titre>Simetierre</titre><ref-auteur ref="a03"/>
  </livre>
  <livre annee="1964" reference="I542">
    <titre>Un défilé de robots</titre><ref-auteur ref="a04"/>
  </livre>
  <livre annee="1950" reference="I453">
    <titre>I, robot</titre><ref-auteur ref="a04"/>
  </livre>
  <!-- ... -->
</edition>
```



Question 2. Donner l'expression XPath 1.0 et XPath 2.0 référençant :

1. les auteurs (proposez toutes les variantes possibles);
2. les auteurs français ;
3. le deuxième auteur ;
4. les titres de livre qui contiennent la chaîne 'robot' ;
5. les ouvrages ayant été publiés en 1996 ;
6. les auteurs ayant publiés en 1996 ;
7. le titre des ouvrages d'Isaac Asimov.

Remarque : pour chacune de ces expressions, vous donnerez aussi le résultat retourné.

Solution.

Requête	XPath 1.0	XPath 2.0	Résultat
les auteurs (proposez toutes les variantes possibles)	/child::edition/child::auteur //edition/auteur //auteur	idem	<auteur no="a01" nom="Éric Holder" pays="France"/> <auteur no="a02" nom="Winston Groom" pays="États Unis"/> <auteur no="a03" nom="Stephen King" pays="États Unis"/> <auteur no="a04" nom="Isaac Asimov" pays="Russie"/>
les auteurs français	/edition/auteur[@pays='France']	/edition/auteur[@pays eq 'France']	<auteur no="a01" nom="Éric Holder" pays="France"/>
le deuxième auteur	/edition/auteur[2] ou /edition/auteur[position()=2]	/edition/auteur[position() eq 2]	<auteur no="a02" nom="Winston Groom" pays="États Unis"/>
les titres de livre qui contiennent la chaîne 'robot'	/edition/livre/titre[contains(., 'robot')] /edition/livre/titre[contains(text(), 'robot')] /edition/livre/titre[contains(., 'robot')]		<titre>Un défilé de robots</titre> <titre>I, robot</titre>
les ouvrages ayant été publiés en 1996	/edition/livre[@annee = '1996']	/edition/livre[@annee eq '1996']	<livre annee="1996" reference="I5157"> <titre>La ligne verte</titre> <ref-auteur ref="a03"/> </livre>
les auteurs ayant publiés en 1996	/edition/auteur[@no = /edition/livre [@annee = '1996']/ref-auteur/@ref] ou id(/edition/livre[@annee = '1996']/ ref-auteur/@ref)	/edition/auteur[@no = /edition/livre[@annee eq '1996']/ref-auteur/@ref] ou id(/edition/livre[@annee eq '1996']/ ref-auteur/@ref)	<auteur no="a03" nom="Stephen King" pays="États Unis"/>
le titre des ouvrages d'Isaac Asimov	/edition/livre[ref-auteur/@ref= edition/auteur[@nom='Isaac Asimov']/@no]/titre	/edition/livre[ref-auteur/@ref = /edition/auteur [@nom eq 'Isaac Asimov']/@no]/titre ou idref(/edition/auteur[@nom eq 'Isaac Asimov']/@no) /../../node()[name() eq 'livre']/titre	<titre>Un défilé de robots</titre> <titre>I, robot</titre>

Exercice 2.

«Calculer» le résultat des expressions XPath suivantes (indiquer si elles sont ou pas valables en XPath 1.0 et en XPath 2.0) sur le document ci-contre :

Requête	Résultat
«distinct-values(/edition/auteur/@pays)»	France États Unis Russie
«doc("jailu.xml")/edition/auteur»	<auteur no="a01" nom="Éric Holder" pays="France"/> <auteur no="a02" nom="Winston Groom" pays="États Unis"/> <auteur no="a03" nom="Stephen King" pays="États Unis"/> <auteur no="a04" nom="Isaac Asimov" pays="Russie"/>
«id(/edition/livre[every \$n in /edition/livre/@annee satisfies \$n le @annee] /ref-auteur/@ref)/@nom»	Éric Holder
«id(/edition/livre[number(@annee) ge max(/edition/livre/@annee)]/ref-auteur/@ref)/@nom»	Éric Holder NB : C'est en fait la même requête que la précédente, mais d'écriture plus simple !

Ce sont toutes des requêtes XPath 2.0 uniquement.

Exercice 3.

Question. Étant donné un paramètre "\$nom", donner l'expression XPath permettant de désigner tous les éléments frères du noeud courant de nom "\$nom".

Solution : ../node()[name()='`\$nom`']

Cette solution suppose que si le noeud courant possède aussi ce nom, il sera sélectionné. "../\$nom" serait tout aussi bien.

Exercice 4.

Un élément est marqué lorsqu'il possède un attribut de nom "marque" et dont la valeur est "oui". Autrement, il est considéré comme non marqué (pas d'attribut "marque" ou un attribut "marque" ayant la valeur "non").

Question. Donner l'expression XPath permettant de désigner les noeuds éléments marqués dont aucun des ancêtres n'est marqué.

Solution : Il suffit de tester la présence de la marque à "oui" et de tester, dans les ancêtres, l'absence de noeud marqué. Ceci donne : //*[@marque = "oui" and not(ancestor::*[@marque = "oui"])]

Exercice 5.

Soit la DTD "edt.dtd" décrivant une classe de documents représentant des emplois du temps d'étudiants dont "edt1213.xml" est un exemple. Ces deux documents sont présentés en fin d'exercice.

Question 1. Donner le résultat de la requête XPath suivante sur ce document. Expliquez ce qu'elle recherche.

```
/edt/liste_profs/prof[
  @idprof=/edt/liste_semaines/semaine/jour/creneau/@nprof
]/concat(prenom, ' ', nom)
```

Solution 1. Les enseignants qui interviennent effectivement.

Max Durant

Jucien Hadirent

Question 2. Parmi les requêtes suivantes, indiquez, en justifiant votre choix, lesquelles donnent le même résultat que la requête de la question précédente ? En cas de réponse négative, donnez les raisons.

- `/edt/liste_semaines/semaine/jour/creneau/@nprof/concat(prenom, ' ', nom)`

Solution : non, cette requête n'a pas de sens puisqu'elle cherche les sous-éléments "prenom" et "nom" de l'attribut "nprof" !

- `id(/edt/liste_semaines/semaine/jour/creneau/@nprof)/concat(prenom, ' ', nom)`

Solution : oui

- `/edt/liste_profs/prof[
 @idprof = ../../liste_semaines/semaine/jour/creneau/@nprof
]/concat(prenom, ' ', nom)`

Solution : oui

- `/edt/liste_profs/prof[
 @idprof = ../../../../liste_semaines/semaine/jour/creneau/@nprof
]/concat(prenom, ' ', nom)`

Solution : non, il y a trop de '..' ce qui fait que l'on remonte "au dessus de la racine" !

- `concat(id(/edt/liste_semaines/semaine/jour/creneau/@nprof)/prenom, ' ', id(/edt/liste_semaines/semaine/jour/creneau/@nprof)/nom)`

Solution : non, il est essayé de concaténer deux séquences (l'une de noms et l'autre de prénoms).

- `/edt/liste_profs/prof[
 some $c in /edt/liste_semaines/semaine/jour/creneau
 satisfies $c/@nprof eq @idprof
]/concat(prenom, ' ', nom)`

Solution : oui

- `/edt/liste_profs/prof[
 every $c in /edt/liste_semaines/semaine/jour/creneau
 satisfies $c/@nprof eq @idprof
]/concat(prenom, ' ', nom)`

Solution : non, elle retourne rien, car il faudrait un enseignant faisant tous les enseignements.

- `/edt/liste_profs/prof[id(@idprof)]/concat(prenom, ' ', nom)`

Solution : non, elle retourne tous les enseignants (elle teste l'existence de l'id de chaque enseignant).

- `/edt/liste_profs/prof[exists(idref(@idprof))]/concat(prenom, ' ', nom)`

Solution : oui

Question 3. Écrire les requêtes XPath (sans utiliser "//") permettant d'obtenir les informations suivantes :

- Les jours (date) où il y a un enseignement en semaine 12 ;

```
lu 19/03/2007
me 21/03/2007
je 22/03/2007
ve 23/03/2007
```

Solution : Cette requête n'est pas difficile. Il suffit de rechercher la bonne semaine. Ensuite, il faut énumérer les jours en concaténant le nom, le numéro, le mois et l'année (prise en attribut de la racine).

```
/edt/liste_semaines/semaine[@num_semaine eq 's12']/  
  jour[creneau]/concat(@nomjour, " ", @njour, "/", @mois, "/", @edt/@année)
```

«jour[creneau]» est important, car il pourrait y avoir des jours vides (à cause du «creneau*»).

- Les plages horaires (heure de début) qui sont occupées le mardi 27 mars ;

```
/edt[1]/types_creneaux[1]/plage[4]/hdeb[1] - 14:00:00
/edt[1]/types_creneaux[1]/plage[5]/hdeb[1] - 15:30:00
/edt[1]/types_creneaux[1]/plage[6]/hdeb[1] - 17:00:00
```

Solution : Deux solutions sont possibles : avec ou sans utiliser "id()". Dans les deux cas, il faut obtenir la séquence des plages du jour recherché. Ensuite, il faut trouver les plages correspondantes.

```
/edt/types_creneaux/plage[
@ordre=/edt/liste_semaines/semaine/jour[@nomjour eq 'ma'
and @njour eq '27'
and @mois eq '03']/creneau/@nplage
]/hdeb
```

ou

```
id(/edt/liste_semaines/semaine/jour[@nomjour eq 'ma'
and @njour eq '27'
and @mois eq '03']/creneau/@nplage)/hdeb
```

- La journée (sa date) la plus chargée de l'emploi du temps ;
28/03

Solution : Pour cette requête, deux approches sont possibles : utiliser "max()" ou sa définition (un élément d'un ensemble "E" pour lequel toute valeur de "E" est inférieure ou égale à cet élément).

```
/edt/liste_semaines/semaine/jour[
every $j in /edt/liste_semaines/semaine/jour
satisfies count(creneau) ge count($j/creneau) ]/concat(@njour,' ',@mois)
```

ou

```
/edt/liste_semaines/semaine/jour[
count(creneau) eq max(/edt/liste_semaines/semaine/jour/count(creneau))
]/concat(@njour,' ',@mois)
```

- Les jours (date) où tous les enseignements sont sur Projet XML ;
27/03

Solution : Là, il faut s'appuyer sur le quantificateur "every"

```
/edt/liste_semaines/semaine/jour[
every $c in creneau/matiere satisfies $c eq "J2EE"]/concat(@njour,' ',@mois)
```

- Le nombre moyen de créneaux par jour sur toutes les semaines ;
3.25

Solution :
avg(/edt/liste_semaines/semaine/jour/count(creneau))

- Même question si on compte seulement les jours d'enseignement ;
2.6

Solution :
sum(/edt/liste_semaines/semaine/jour/count(creneau)) div (count(/edt/liste_semaines/semaine)*5)

- Les professeurs (prénom, nom et téléphone) qui n'enseignent pas le mercredi de la semaine 12 ;
Jean-Pierre Fabriol : 0252125487

Solution : Il faut rechercher les enseignants de cette journée et s'assurer que l'enseignant étudié n'y est pas.

```
/edt/liste_profs/prof[
every $c in /edt/liste_semaines/semaine[@num_semaine eq "s12"]/jour[@nomjour eq "me"]/creneau
satisfies not($c/@enprof eq @idprof)
]/concat(prenom,' ',nom, ' : ', tel)
```

Attention, la solution suivante n'est pas bonne. En effet, il existe nécessairement une liste d'enseignants dont au moins un est différent de celui étudié !

```
/edt/liste_profs/prof[
  @idprof != /edt/liste_semaines/semaine[
    @num_semaine eq "s12"]/jour[@nomjour eq "me"]/creneau/@nprof]/concat(prenom, ' ', nom, ' : ', tel)
```

Par contre, la suivante est bonne.

```
/edt/liste_profs/prof[
  not(@idprof = /edt/liste_semaines/semaine[
    @num_semaine eq "s12"]/jour[@nomjour eq "me"]/creneau/@nprof)
]/concat(prenom, ' ', nom, ' : ', tel)
```

Pour chaque requête, les résultats présentés (sous la requête) sont ceux obtenus avec oXygen 8.1 et XPath 2.0. Les chemins ne font pas partie de la réponse, oXygen les places pour localiser la réponse. Seule l'information après le '/' est valable.



Cette création est mise à disposition selon le Contrat Paternité-Pas d'Utilisation Commerciale-Pas de Modification 2.0 France disponible en ligne <http://creativecommons.org/licenses/by-nc-nd/2.0/fr/> ou par courrier postal à Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.