Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

2021 - 2022
WEBR3 / WEBR4 - XML

Introduction à XML

XPath

Ce document introduit la syntaxe XPath.

Le contenu de ce TD est largement inspiré de :

— *Essential XML Quick Reference : A Programmer's Reference to XML, XPath, XSLT, XML Schema, SOAP, and More* : https://www.pearson.com/store/p/essential-xml-quick-reference-a-programmer-s-reference-to-xml-xpath-xslt-xml-sc P100000683470/9780201740950

probablement un peu au delà d'un usage raisonnable (*fair use*).

D'autres ressources sont disponibles en ligne :

— *xpath cover page - W3C* : https://www.w3.org/TR/xpath/all/
— *XML Path Language (XPath)* : https://www.w3.org/TR/1999/REC-xpath-19991116/
— *Langage XML Path (XPath)* : http://xmlfr.org/w3c/TR/xpath

Premier tutoriel Structurez vos données avec XML (pour commencer en douceur, un peu léger) : https://tutoriel-xml.rolandl.fr/.

Version du document : 29 septembre 2021

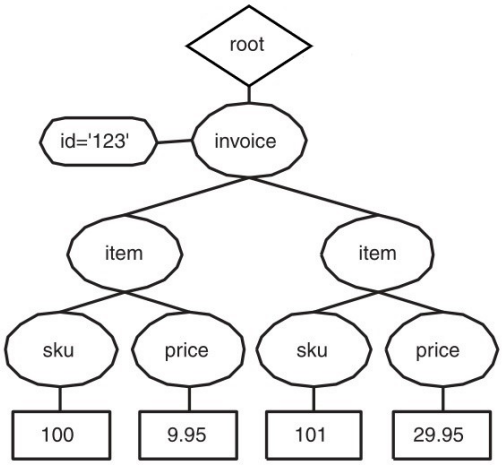

FIGURE 1 – Représentation sous la forme d'un arbre d'un fichier XML.

1 Présentation

XPath est une syntaxe utilisée pour adresser un *ensemble de nœuds* dans un document XML. Il faut entendre le terme *ensemble* au sens mathématique : si un élément y est présent, c'est de manière unique, sans doublon. Cet ensemble peut être vide.

Le standard XPath existe en trois version majeures. Nous nous limitons dans le cadre des laboratoire XML à sa version 1.0.

XPath est utilisé lors de la définition de schémas XML ou encore lors de la production de feuilles de transformation XSLT.

Lors de l'utilisation de la syntaxe XPath, tout document XML est représenté sous la forme d'un arbre. C'est dans cet arbre que la syntaxe XPath permet de se balader.

1.1 Fichier XML

Voici un fichier XML qui sert de support pour les exemples donnés au long de ce TD :

```xml
<?xml version="1.0" ?>
<invoice id='123'>
    <item>
        <sku>100</sku>
        <price>9.95</price>
    </item>
    <item>
        <sku>101</sku>
        <price>29.95</price>
    </item>
</invoice>
```

Le parcours de ce fichier à l'aide de la syntaxe XPath se fait dans l'arbre représenté à la FIG. 1. On remarque en particulier l'existence d'un *nœud racine du document*[1]

1. https://www.w3.org/TR/1999/REC-xpath-19991116/#root-node (consulté le 22 fé-

(*root*) distinct du *nœud* élément *racine du document* (*invoice* dans le cas présent).

Les nœuds qui sont identifiés par une expression XPath sont les éléments [2] constitutifs du document XML et de sa représentation sous la forme d'un arbre : nœud racine, éléments, attributs, élément textuel, commentaires, espace de nommage, instruction de traitement.

2 xmllint

L'utilitaire `xmllint` offre la possibilité d'une utilisation comme un *shell* au sein d'un document XML.

Exécuter :

```
$ xmllint --shell invoice.xml
```

pour ouvrir le fichier `invoice.xml` dans le *shell* de `xmllint`.

On peut alors récupérer des informations avec les commandes `ls` ou `pwd` ou encore se déplacer dans la structure en arbre du document avec la commande `cd`.

Pour évaluer une expression XPath dans le *shell* de `xmllint`, il faut utiliser la commande :

```
> xpath expression
```

où *`expression`* est une expression XPath (voir section 4).

Davantage d'exemples d'utilisation du *shell* d'`xmllint` sont donnés en section 6 (p. 9 et suivantes).

Remarque importante Comme il n'y a pas de prise en charge des espaces de nommage directement par XPath 1.0 (voir section 5), il n'est pas possible de produire des expressions XPath dans le *shell* d'`xmllint` pour des éléments plongés dans un espace de noms. Il faut dès lors les en sortir si on veut réaliser des tests avec cet outil.

3 Type de donnée

Il existe 4 types de données en XPath :

- ensemble de nœuds (*node set*) : il s'agit d'une collection de nœuds du documents sans duplication ;
- booléen (*boolean*) : ils prennent les valeurs `true` ou `false`
- nombre (*number*) : ce sont des flottants à la norme IEEE 754 ;
- chaîne de caractères (*string*) : c'est-à-dire une suite de caractères.

vrier 2021).

2. https://www.w3.org/TR/1999/REC-xpath-19991116/#data-model (consulté le 22 février 2021).

3.1 Conversion

Il existe des règles de conversion d'un type vers les autres. Pour en connaître tous les détails, consultez la documentation de :

- la fonction `string string(object?)` [3] qui convertit un objet de type quelconque en chaîne de caractères ;
- la fonction `boolean boolean(object)` [4] qui convertit un objet de type quelconque en booléen ;
- la fonction `number number(object?)` [5] qui convertit un objet de type quelconque en nombre.

Par exemple, la conversion d'un nœud vers une chaîne de caractères est obtenue en concaténant les nœuds textuels de ses descendants. Celle d'un ensemble de nœuds en *string* est obtenue en convertissant le premier nœud (dans l'ordre du document) de l'ensemble en chaîne de caractères.

4 Expression de localisation

Pour localiser un (ensemble de) nœud(s) dans un document, on fournit un *chemin* à travers une *expression de localisation* (*location path expression*).

4.1 Chemin absolu ou relatif

Un chemin peut être :

- *absolu* (*absolute*) : il commence à la racine du document, marquée par une oblique « `/` » ;
- relatif : il commence au nœud contextuel.

Un chemin constitué de plusieurs pas (voir la section 4.2) est relatif dès son second pas.

4.2 Suite de pas

Un chemin est constitué d'une suite de *pas* (*steps*).

Partant du nœud racine ou du nœud contextuel, on effectue le premier pas. Ceci produit un ensemble de nœuds. S'il existe un pas suivant dans l'expression de localisation, celui-ci est réalisé pour chaque nœud de l'ensemble obtenu au premier pas. Les nœuds de l'ensemble obtenu suite au premier pas deviennent nœud contextuel les uns à la suite des autres. On obtient alors un nouvel ensemble de nœuds, suite au second pas. S'il y a des nœuds en double dans un ensemble produit suite à un pas, ils sont enlevé de l'ensemble de sorte à n'y apparaître chacun qu'une seule fois. L'évaluation de l'expression de localisation continue ainsi jusqu'à son dernier pas.

Les pas sont séparés par une oblique « `/` ».

3. https://www.w3.org/TR/1999/REC-xpath-19991116/#function-string (consulté le 22 février 2021).
4. https://www.w3.org/TR/1999/REC-xpath-19991116/#function-boolean (consulté le 22 février 2021).
5. https://www.w3.org/TR/1999/REC-xpath-19991116/#function-number (consulté le 22 février 2021).

4.2.1 Exemples

Voici l'expression d'un chemin absolu constitué de 3 pas :

```
/pas1/pas2/pas3
```

Voici l'expression d'un chemin relatif constitué de 2 pas :

```
pas1/pas2
```

4.3 Pas

Un *pas*[6] (*location step*) est constitué de deux termes obligatoire et d'un ensemble de termes facultatifs.

Les parties obligatoire sont :

- l'*axe* selon lequel on se déplace dans l'arbre correspondant au document ;
- le *test* auquel les nœuds qui répondent positivement sont incorporés à l'ensemble de nœuds produits.

Les termes facultatifs sont les *prédicats*. Ils s'appliquent après le test pour filtrer davantage les nœuds de l'ensemble de nœuds adressés par l'expression de localisation.

4.3.1 Axe

Chaque pas se fait selon un *axe*[7] (*axe*). Il existe 12 axes. Ils sont renseignés à la TABLE 1.

Le document est parcouru de sa première vers sa dernière ligne, *sauf* pour les axes `parent`, `ancestor`, `ancestor-or-self`, `preceding` et `preceding-sibling` pour lesquels il est parcouru à rebours.

4.3.2 Test

À chaque pas est associé un *test*[8] (*node test*).

Les nœuds qui se trouvent sur l'axe spécifié pour un pas et pour lesquels le résultat du test de ce pas est positif sont ajoutés à l'ensemble de nœuds produits par ce pas.

Il existe deux types de tests :

- *test par nom* (*test by name*) : on fournit un nom, les nœuds de ce nom selon l'axe renseigné (voir section 4.3.1) sont ajoutés à l'ensemble de nœuds résultat. Avec l'étoile « `*` », tous les nœuds de cet axe sont ajoutés ;
- *test selon le type* (*test by type*) : les nœuds positifs à ce type de test sont ceux du type indiqué. On peut renseigner les types :
  - `text()` pour identifier les nœuds textuels ;
  - `comment()` pour identifier les nœuds commentaires ;

6. https://www.w3.org/TR/1999/REC-xpath-19991116/#section-Location-Steps (consulté le 22 février 2021).
7. https://www.w3.org/TR/1999/REC-xpath-19991116/#axes (consulté le 22 février 2021).
8. https://www.w3.org/TR/1999/REC-xpath-19991116/#node-tests (consulté le 22 février 2021).

Axe	Description
`self`	nœud contextuel
`child`	enfants (directs) du nœud contextuel
`parent`	parent du nœud contextuel
`descendant`	enfants, petits-enfants, etc. du nœud contextuel
`descendant-or-self`	nœud contextuel et ses enfants, petits-enfants, etc.
`ancestor`	parent, grand-parent, etc. du nœud contextuel
`ancestor-or-self`	nœud contextuel et ses parent, grand-parent, etc.
`following`	tous les nœuds qui suivent le nœud contextuel dans l'ordre du document à l'exception de ses attributs et descendants, donc les nœuds dans la balise ouvrante apparaît après la balise fermante du nœud courant
`following-sibling`	les nœuds frères du nœud contextuel à sa suite dans l'ordre du document
`preceding`	tous les nœuds qui précèdent le nœud contextuel dans l'ordre du document à l'exception de ses attributs et ascendants, donc les nœuds dans la balise fermante apparaît après la balise ouvrante du nœud courant
`preceding-sibling`	les nœuds frères du nœud contextuel le précédant dans l'ordre du document
`attribute`	attributs du nœud contextuel
`namespace`	nœuds espace de nommage du nœud contextuel

Table 1 – Axes.

— `processing-instruction(target?)` pour identifier les nœuds instruction de traitement ;

— `node()` pour identifier tous les nœuds dans l'axe indiqué, quel que soient leur type.

Les tests de nom *matchent* pour des nœuds éléments pour tous les axes sauf pour l'axe attribut où ils *matchent* pour des nœuds attributs et pour l'axe *namespace* pour des espace de noms.

4.3.3 Axe et test

L'indication de l'axe précède celle du test et est séparée de cette dernière par deux deux-points « `::` ».

L'expression générale minimale d'un pas est donc :

```
axe::test
```

L'axe `child` est l'axe par défaut. S'il est omis, c'est l'axe emprunté par l'expression XPath.

Exemples Avec le document XML de la section 1.1 :

— `/child::invoice` est un chemin absolu qui identifie le nœud racine `invoice` ;

— `/invoice` identifie le nœud racine `invoice` ;

— `/child::invoice/child::item` identifie les nœuds enfants `item` du nœud racine `invoice` ;

— `/invoice/item` identifie les nœuds enfants `item` du nœud racine `invoice` ;

— `/invoice/attribute::id` identifie l'attribut `id` du nœud racine `invoice` ;

— `/invoice/@id` identifie l'attribut `id` du nœud racine `invoice` (voir la section 4.3.6) ;

— `item/sku` est un chemin relatif qui identifie les nœuds enfants `sku` des nœuds enfants `item` du nœud contextuel ;

— `ancestor::item` identifie le nœud `item` ancêtre du nœud contextuel.

4.3.4 Prédicat

À la suite du test, on peut ajouter un ou plusieurs *prédicats*[9] (*predicate*). Ils servent à filtrer les nœuds de l'ensemble de nœuds obtenus après le test. Seuls les nœuds répondant positivement au prédicat sont versés dans l'ensemble de nœuds de l'expression. S'il y a plusieurs prédicats, ils sont évalués de gauche à droite.

Le prédicat est renseigné entre crochets « `[ ]` ».

L'expression du prédicat peut utiliser :

— les opérateurs de comparaison[10] : `=`, `!=`, `<`, `<=`, `>`, `>=`

9. https://www.w3.org/TR/1999/REC-xpath-19991116/#predicates (consulté le 22 février 2021).
10. https://www.w3.org/TR/1999/REC-xpath-19991116/#booleans (consulté le 22 février 2021).

— les opérateurs arithmétiques [11] : `+`, `-`, `*`, `div`, `mod`

— diverses fonctions [12] : `position()`, `last()`, etc.

— les fonctions de conversion d'un type vers un autre, en particulier d'un ensemble de nœuds vers un autre type (voir section 3.1).

Comparaison d'ensembles de nœuds Deux ensemble de nœuds sont :

— *égaux* s'il existe au moins un nœud qu'ils ont en commun ;

— *différents* s'il existe au moins un nœud qu'ils n'ont pas en commun.

Il est dès lors courant que deux ensembles de nœuds soient simultanément égaux et différents !

4.3.5 Axe, test et prédicats

L'expression générale d'un pas est donc :

```
axe::test[predicat_1][predicat_2]...[predicat_n]
```

Exemples Avec le document XML de la section 1.1 :

— `child::item[position()=2]` est une expression qui identifie le deuxième nœud `item` enfant du nœud contextuel ;

— `item[2]` est une expression qui identifie le deuxième nœud `item` enfant du nœud contextuel (voir la section 4.3.6) ;

— `/item[price][2]` identifie le 2e nœuds `item` doté d'un enfant nommé `price` et enfant du nœud contextuel ;

— `item[price][2]/sku/text()` identifie les nœuds textuels enfants des noeuds `sku` du 2e nœuds `item` doté d'un enfant nommé `price` et enfant du nœud contextuel ;

— `/invoice[@id>100]//price[.<15]` identifie l'ensemble des éléments `price` de valeur supérieure à 15 (conversion d'un nœud élément en nombre via une chaîne de caractère) dans la descendance de l'élément racine `invoice` de valeur d'attribut `id` supérieur à 100 ;

— `descendant::sku[last()]` identifie le dernier élément `sku` dans la descendance du nœud contextuel.

4.3.6 Notation raccourcie

Il existe des notations raccourcies [13] pour certains axes ou tests. Elles sont renseignées à la Table 2.

11. https://www.w3.org/TR/1999/REC-xpath-19991116/#numbers (consulté le 22 février 2021).
12. https://www.w3.org/TR/1999/REC-xpath-19991116/#corelib (consulté le 22 février 2021).
13. https://www.w3.org/TR/1999/REC-xpath-19991116/#path-abbrev (consulté le 22 février 2021).

Expression	Notation raccourcies
`child::`	(rien)
`attribute::`	`@`
`self::node()`	`.`
`parent::node()`	`..`
`descendant-or-self::node()/`	`//`
`[position()=number]/`	`[number]`

Table 2 – Notation raccourcie d'expressions.

4.4 Union de chemins

Une *expression de localisation* (*location path expression*) est en toute généralité, une *union* de chemins. L'ensemble de nœuds d'une telle union de chemins est l'union des ensembles de nœuds de chaque expression de localisation.

L'union est obtenue à l'aide de la barre verticale « `|` ».

Voici l'expression d'un chemin obtenu par l'union d'un chemin absolu constitué de 3 pas et d'un chemin relatif de 2 pas :

```
/pas1/pas2/pas3 | pas4/pas5
```

5 Espace de nommage

Il n'y a pas de prise en charge des espaces de nommage dans la norme XPath 1.0. Cette prise en charge incombe dès lors aux technologies qui utilisent XPath. On verra dans les TD prochains comment concilier espace de noms, XPath et schémas XML puis transformations XSLT.

6 Exercices

Les sources de données pour les exercices qui suivant sont les fichiers `esi.xml` ou `esi_minified.xml`.

Utilisez `xmllint` pour valider vos réponses !

Exercice 1

```
$ xmllint --shell esi.xml
/ > pwd
/
/ > ls
c--       29
---       17 coursesi
c--       62
/ > xpath /coursesi
Object is a Node Set :
Set contains 1 nodes:
1  ELEMENT coursesi
```

```
/ > xpath /coursesi/sections/section
Object is a Node Set :
Set contains 3 nodes:
1  ELEMENT section
    ATTRIBUTE id
      TEXT
        content=R
2  ELEMENT section
    ATTRIBUTE id
      TEXT
        content=I
3  ELEMENT section
    ATTRIBUTE id
     TEXT
       content=G
/ > exit
$
```

Produisez des expression de localisation *à partir du nœud racine du document* (reposant sur un chemin absolu donc) adressant les ensembles de nœuds suivants demandés :

1. Ensemble des éléments `<name>`.
2. Ensemble des éléments `<name>` enfants de `<profs>`.
3. Ensemble des textes des éléments `<name>` directement sous `<profs>`.
4. Textes des éléments `<prof>` .
5. Ensemble des textes des éléments sous `<prof>`.
6. Élément `<email>` de prof d'`id` « CUV ».
7. Valeur textuelle de l'élément `<email>` de prof d'`id` « CUV ».
8. Ensemble des éléments du document.
9. Ensemble des commentaires du document.
10. Nombre total de commentaires du document.
11. Commentaires enfants de l'élément racine du document.
12. Commentaires petits-enfants de l'élément racine du document.
13. Commentaires arrières petits-enfants de l'élément racine du document.
14. Ensemble des commentaires sous l'élément racine du document.
15. Dernier commentaire du document.
16. Premier commentaire sous l'élément racine.
17. Ensemble des `<aa>` données par « EGR ».
18. Ensemble des `<aa>` données par « ARO ».
19. Ensemble des `<name>` des `<aa>` données par « EGR ».
20. Ensemble des `<name>` des `<aa>` données par « ARO ».
21. Somme des heures de l'`<ue>` d'identifiant « ANAGIR3 ».

22. Ensemble des `<ue>` de la section gestion.
23. Ensembles des `<aa>` données par le 3e `<prof>`.
24. 2e `<aa>` données par l'antépénultième[14] `<prof>`.
25. Texte de l'objectif de la 2e `<aa>` données par le 8e `<prof>`.
26. Identifiants des enseignants de la 2e `<aa>` de la 1re `<ue>` si elle est données par le 8e `<prof>`.
27. Identifiants des enseignants de la 2e `<aa>` de la 1re `<ue>` si elle est données par le 4e `<prof>`.
28. Nombre de crédits de la 1re `<ue>` dont au moins une `<aa>` est donnée par le 5e `<prof>`.
29. Nombre de crédits de la 1re `<ue>` dont au moins 2 `<aa>` sont données par le 5e `<prof>`.

Exercice 2

```
$ xmllint --shell esi.xml
/ > cd coursesi/profs
profs > ls
ta-        5
-a-        5 prof
ta-        5
...
ta-        3
profs > ls prof[1]/name
tan       12 Absil Romain
profs > ls prof[last()]/email
t--       16 egeorges@he2b.be
profs > cd prof[11]
prof > ls name
tan       13 Rousseau Anne
prof > quit
$
```

Produisez des expression de localisation *à partir d'un nœud contextuel* ***`<prof>`*** (reposant sur un chemin relatif donc) adressant les ensembles de nœuds suivants demandés :

1. Acronyme du `<prof>` courant.
2. Élément `<name>` du `<prof>` courant.
3. Ensemble des éléments enfants du `<prof>` courant.
4. Ensembles des `<prof>` précédent le `<prof>` courant dans le document.
5. `<prof>` courant et ensembles des `<prof>` le suivant dans le document.

14. https://www.cnrtl.fr/definition/ant%C3%A9p%C3%A9nulti%C3%A8me (consulté le 22 février 2021).

Exercice 3

```
$ xmllint --shell esi.xml
/ > cd /coursesi/profs/prof[11]/email
email > ls
t--       17 arousseau@he2b.be
email >
```

Produisez des expression de localisation *à partir d'un nœud contextuel* `<email>` (reposant sur un chemin relatif donc) adressant les ensembles de nœuds suivants demandés :

1. Texte du nœud courant.
2. Élément `<name>` du `<prof>` d'`<email>` courant.
3. Acronyme du `<prof>` d'`<email>` courant.
4. Ensembles des `<email>` des `<prof>` précédant dans le document le `<prof>` d'`<email>` courant.
5. Valeur de l'`<email>` du 2e `<prof>` suivant dans le document le `<prof>` d'`<email>` courant.