XPATH
Prezentare

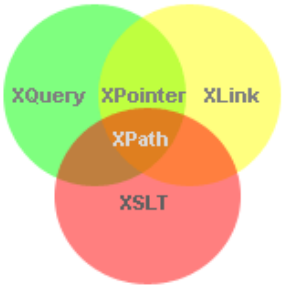
1. Despre XPath

XPath este un „limbaj” ce permite regasirea unor parti dintr-un fisier XML precum si „navigarea” prin fisiere XML. Documentele XML au o structura arborescenta, datorita relatiei „tata-fiu” a nodurilor din acel document, ceea ce permite limbajului XPath sa poata sa acceseze diferite elemente din cadrul documentelor XML. Acesta este si *scopul principal* al acestui limbaj. Datorita acestei posibilitati, limbajul permite selectarea nodurilor sau a unor seturi de noduri din documentul XML.

Pentru a permite accesarea diferitelor elemente ale documentului XML, XPath foloseste o sintaxa foarte asemanatoare cu cea utilizata pentru accesarea fisierelor intr-un sistem de fisiere UNIX (ex: „/grupa/studenti/student/nume” pentru fisierul XML de mai jos). XPath nu respecta structura documentelor XML (nu este un fisier XML).

Numele limbajului se trage de la faptul ca utilizeaza „cai” (PATHS) pentru identificarea elementelor XML.

XPath este un standard W3C de sine statator si o componenta de baza a XSLT, XLink, XQuery si XPointer, asa cum se poate observa din figura de mai jos:



Poza downloadata pe data de 18.10.2007 de la adresa
http://www.w3schools.com/xpath/default.asp

2. Sintaxa

In cadrul XPath exista sapte tipuri de noduri: element (nodurile documentului), atribut, text, namespace, instructiuni de procesare, comentariu, si radacina documentului (root). Datorita structurii arborescente a documentelor XML, intre nodurile acestuia se stabilesc *relatii de rudenie* (ex: parinte, copil, urmas, strabun, etc.).

Selectarea unui nod se face prin urmarea unei **cai** XPath. Aceste cai pot fi **absolute** si atunci incep cu „/” sau pot fi **relative** si atunci NU incep cu „/”. In primul caz, calea catre destinatie porneste din radacina documentului, iar in cel de-al doilea, calea porneste din nodul curent. O cale XPath contine unul sau mai multi pasi separati de „/”:

- pentru expresii absolute: /PAS1/PAS2/...
- pentru expresii relative: PAS1/PAS2/...

Pasii unei expresii XPath sunt evaluati in ordine, de la stanga la dreapta. Daca s-a ajuns la evaluarea PAS2, aceasta se realizeaza relativ la fiecare nod rezultat in urma evaluarii PAS1.

Pentru o mai usoara intelegere a acestor notiuni, vom folosi urmatorul fisier XML pentru exemplificare.

```
<?xml version=„1.0” encoding=„ISO-8859-1”?>
<grupa id=„324CB”>
    <studenti>
        <student id=„1212”>
            <nume>Popescu</nume>
            <prenume>Ion</prenume>
        </student>
        <student id=„1213”>
            <nume>Ionescu</nume>
            <prenume>Dan</prenume>
        </student>
    </studenti>
    <advisor id=„2323”>
        <nume>Georgescu</nume>
        <prenume>Alex</prenume>
    </advisor>
</grupa>
```

Un **pas** dintr-o cale XPath este un tuplu care are urmatoarele trei componente:

nume_axa + nod_test + lista_de_predicate (optional)

ex: „child::student[nume=Ionescu]”

Fiecare componenta a unui pas selecteaza anumite noduri, iar privite de la stanga la dreapta, fiecare adauga un set de restrictii suplimentar (ex: nume_axa=„child” se refera la toti copii nodului curent, nod_test=„student” se refera la copiii nodului curent care au numele „student”, iar lista_de_predicate=„nume=Ionescu” se refera la copiii nodului curent care au numele „student" si care au un nod „nume” cu valoarea Ionescu.

In continuare vor fi prezentate cele trei componente ale unui pas:

- **nume_axa** = specifica relatia intre nodurile care sunt selectate de catre acest pas si nodul curent (ex: „child” specifica faptul ca nodurile identificate in pasul curent sunt copii ai nodului curent; nod curent in acest caz e nodul identificat de pasul anterior). XPath se bazeaza pe navigarea de-a lungul acestor axe.

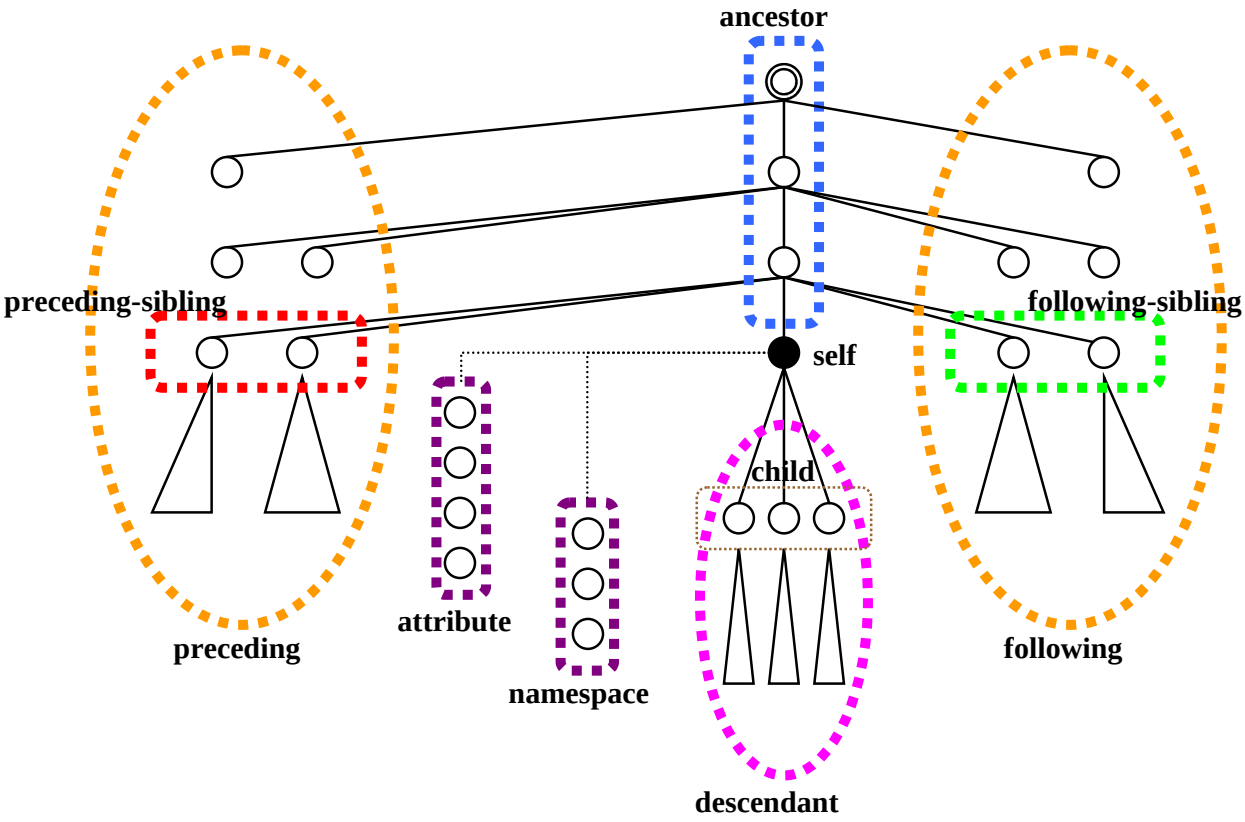
OBSERVATII:
  - Daca nu se specifica „nume_axa” atunci se utilizeaza axa implicita, aceasta fiind axa „child::”
  - Atributele sunt specificate fie folosind numele de axa „attribute::” fie prescurtarea „@”(ex: „@id”)
- **nod_test** = specifica fie numele nodului de identificat (ex: „student”) fie tipul acestuia (ex: „child::text()” identifica nodurile fii care sunt de tipul TEXT = acele noduri de tip „#PCDATA” specificat in DTD-ul unui document XML.
  OBSERVATIE: Intr-un document XML, un text este considerat si el un element. (ex: <e1></e1>Salut!<e2></e2>, aici „Salut!” este tot un element, de tip TEXT)
- **lista_de_predicate** = sunt functii folosite pentru a filtra dupa anumite conditii nodurile selectate in pasul curent

2.1 Axe XPath

In cadrul documentelor XML exista mai multe axe: ancestor, descendant, ancestor-or-self, descendant-or-self, preceding, following, preceding-sibling, following-sibling, parent, child, self, attribute, namespace. In tabelul de mai jos, se poate observa descrierea acestor axe, iar schema aflata mai jos intregeste imaginea, prezentand intr-o forma grafica cum sunt situate aceste axe in cadrul unui document XML:

Numele Axei	**Rezultatul evaluarii**
ancestor	Selecteaza toti strabunii nodului curent (parinti, bunici, etc.)
descendant	Selecteaza toti urmasii nodului curent (copii, nepoti, etc.)
ancestor-or-self	Selecteaza toti strabunii nodului curent, precum si nodul curent
descendant-or-self	Selecteaza toti urmasii nodului curent, precum si nodul curent
preceding	Selecteaza tot ce este in document inaintea tagului de inceput al nodului curent
following	Selecteaza tot ce urmeaza in document dupa tagul de inchidere al nodului curent
preceding-sibling	Selecteaza toate rudele nodului curent aflate inaintea acestuia
following-sibling	Selecteaza toate rudele nodului curent aflate dupa acesta
parent	Selecteaza parintele nodului curent
child	Selecteaza toti copii nodului curent
self	Selecteaza nodul curent
attribute	Selecteaza toate atributele nodului curent
namespace	Selecteaza toate nodurile cu namespace-urile nodului curent

Dupa cum se poate observa din schema de mai jos, unele dintre aceste axe descriu cate un singur nod (ex: parent, self), in timp ce altele descriu o multime de noduri (care uneori poate fi vida – de ex. ancestors pentru primul nod din fisierul XML).



Axele unui document XML
Poza este realizata de Arnaud Sahuguet si afost downladata pe data de 18.10.2007 de la adresa lambda.uta.edu/cse6331/spring02/**XML**2.ppt

Exemplu	**Rezultat**
child::student	Selecteaza toate nodurile student care sunt copii ai nodului curent
attribute::id	Selecteaza atributul id al nodului curent
child::*	Selecteaza toti copiii ai nodului curent
attribute::*	Selecteaza toate atributule ai nodului curent
child::text()	Selecteaza toate nodurile de tip text care sunt copii ai nodului curent
child::node()	Selecteaza toti copiii de tip nod ai nodului curent
descendant::student	Selecteaza toate nodurile student care sunt descendentii nodului curent
ancestor::nume	Selecteaza toate nodurile nume care sunt stramosi ai nodului curent

2.2 Noduri in XPath

Nod_test poate lua una din valorile de mai jos:

- text() returneaza un element de tip text, nu are taguri delimitatoare
- node() returneaza un element oarecare (chiar si de tip TEXT) – echivalent cu (* sau @* sau text())
- name() returneaza numele tagului curent

Cele mai utile expresii XPath pentru selectarea nodurilor sunt:

Expresie	Descriere
nodename	Selecteaza toate nodurile care sunt copii ai nodului curent
/	Selecteaza tot ce se afla in document (toate nodurile de dupa radacina)
//	Selecteaza toate nodurile din document care satisfac criteriile de selectie, indiferent unde se afla in document, dar dupa nodul curent
.	Selecteaza nodul curent
..	Selecteaza parintele nodului curent
@	Selecteaza atributele

Wildcards – sunt utilizate pentru a selecta noduri

- * returneaza orice element de tip nod
- @* returneaza orice atribut
- node() returneaza orice nod de orice tip

Exemple:

- nume – toti copiii cu numele „nume” (Popescu, Ionescu, Georgescu)
- ./nume – toti copii de tip nume ai nodului curent (echivalent cu nume)
- child::nume – toti copii de tip nume ai nodului curent (echivalent cu nume)
- student/nume – toti nepotii cu numele „nume” si care sunt copiii nodului student (Popescu, Ionescu)
- */nume toti nepotii cu numele „nume” (Popescu, Ionescu, Georgescu)
- . – nodul curent
- / – nodul radacina
- .. – parintele nodului curent
- .// – nodul curent si toti descendentii lui
- // – nodul radacina si toti descendentii lui
- text() – toti copiii de tip text ai nodului curent
- node() – toti copiii nodului curent (include nodurile de tip text si pe cele de tip atribut)
- //nume – toate nodurile de tip nume din document
- //text() – toate nodurile de tip text din document
- @id – atributul id al nodului curent
- /descendant::node()/child::nume – toti nepotii cu numele „nume”

OBSERVATIE: .//nume, nume, si //nume returneaza elemente diferite: prima expresie returneaza toti descendentii de tip nume ai nodului curent, incluzandu-l si pe el, a doua intoarce doar nodurile de tip nume care sunt copii ai nodului curent, iar ultima intoarce toate nodurile de tip nume gasite in document. (ex: sa presupunem ca nodul curent este studenti. Atunci .//nume returneaza Popescu, Ionescu; nume nu returneaza nimic deoarece nodul studenti nu are copii de tip nume, iar //nume returneaza Popescu, Ionescu, Georgescu)

2.3 Predicate Xpath

Predicatele sunt folosite pentru a identifica un anumit nod sau un nod care are o anumita valoare. Intotdeauna predicatele sunt scrise intre paranteze drepte ([]) si sunt conditii care elimina nodurile care nu le satisfac.

Exemple:

- [2] – al doilea copil al nodului curent
- nume[2] – al doilea copil de tip nume al nodului curent
- [last()] – ultimul copil al nodului curent
- student [nume=„Popescu”] – copii de tip student ai nodului curent care au unul sau mai multi copii de tip nume a caror valoare este „Popescu”
- studenti[.//nume = „Popescu”] – copilul de tip studenti ai nodului curent care au intre descendentii lor un element de tip nume care are valoarea „Popescu”
- student[@id < „1213”] – studentii care au valoarea atributului id < 1213
- /descendant::node()/child::nume[parent/attribute::id = “1212”] – toti nepotii cu numele „nume” si care au parintele ce are valoarea atributului id 1212

2.4 Operatori Xpath

Operator	Descriere	Exemplu	Valoarea Returnata
|	sau intre seturi de noduri	//book | //cd	Returneaza un set de noduri ce contine nodurile din book si cd
+	adunare	6 + 4	10
-	scadere	6 - 4	2
*	inmultire	6 * 4	24
div	impartire	8 div 4	2
mod	modulo (restul impartirii)	5 mod 2	1
=	egal	price=9.80	adevarat daca pretul e 9.80 fals daca pretul e 9.90
!=	diferit	price!=9.80	adevarat daca pretul e 9.90 fals daca pretul e 9.80
<	mai mic	price<9.80	adevarat daca pretul e 9.00 fals daca pretul e 9.80
<=	mai mic sau egal	price<=9.80	adevarat daca pretul e 9.00 fals daca pretul e 9.90
>	mai mare	price>9.80	adevarat daca pretul e 9.90 fals daca pretul e 9.80
>=	mai mare sau egal	price>=9.80	adevarat daca pretul e 9.90 fals daca pretul e 9.70
or	sau	price=9.80 or price=9.70	adevarat daca pretul e 9.80 fals daca pretul e 9.50
and	si	price>9.00 and price<9.90	adevarat daca pretul e 9.80 fals daca pretul e 8.50

OBSERVATII:

- XPath converteste fiecare operand la un numar inainte de a face evaluarea operatiei.
- Daca se face verificarea de egalitate intre o valoare si un set de noduri, se returneaza "true" daca exista cel putin un nod din set egal cu valoarea respectiva
- Daca se face verificarea de inegalitate intre o valoare si un set de noduri, se returneaza "true" daca exista cel putin un nod din set inegal cu valoarea respectiva

2.5 Functii Xpath

Functii pe seturi de noduri:

Nume	Descriere
nr=count(node_set)	nr. de noduri din set
node_set=id(value)	un set de noduri care au ca ID value
number=last()	pozitia numerica a ultimului nod din lista procesata
string=name(node)	numele nodului
local-name(node)	numele nodului, fara prefix (prefix:nume)
namespace-uri(node)	namespace-ul nodului
position()	pozitia nodului curent procesat din lista de noduri

Functii pe siruri:

Nume	Descriere
concat(s1,..)	un sir format din sirurile concatenate
contains(sir,subsir)	"true" daca sir contine subsir
normalize-space(sir)	elimina spatiile de la inceput si sfarsit
starts-with(sir,subsir)	"true" daca sir incepe cu subsir
string(value)	converteste la sir de caractere
string-length(sir)	lungimea sirului de caractere sir
substring(sir,start,len)	subsirul din sir ce incepe la poz start si are lungimea len
substring-after(sir,subsir)	subsirul ce se afla in sir dupa subsir
substring-before(sir,subsir)	subsirul din sir aflat inainte de subsir
translate(sir,str1,str2)	inlocuieste fiecare sir de caractere str1 din sir cu sirul de caractere str2

Functii cu numere:

Nume	Descriere
ceiling(nr)	intregul mai mic cel mai apropiat de nr (sau =)
floor(nr)	intregul mai mare cel mai apropiat de nr (sau =)
number(string)	converteste la o valoare numerica
round(nr)	cel mai apropiat intreg
sum(nodeset)	suma tuturor valorilor dintr-un nodeset

Functii pe booleene:

Nume	Descriere
boolean(value)	converteste la boolean
false()	returneaza false
true()	returneaza true
not(cond)	returneaza inversul lui cond
lang(language)	returneaza true daca language e identic cu valoarea nodului xsl:lang

Pe langa aceste functii mai exista functii de acces, de detectare si urmarire a erorilor, de context, functii ce lucreaza cu timpi si durate, precum si functii care lucreaza pe secvente. Mai multe exemple gasiti la adresa http://www.w3schools.com/xpath/xpath_functions.asp

3. Cazuri de utilizare

Acest limbaj este utilizat in orice aplicatie care doreste sa lucreze cu elementele unui fisier XML. In limbaje ca XSLT este folosit pentru a naviga prin document si pentru a identifica diferitele elemente din cadrul acestuia pentru a putea fi prelucrate, iar in limbaje ca XQuery este folosit pentru a identifica si a returna diferite componente ale documentului XML. De asemenea este folosit si de catre XLink si XPointer precum si de catre parsere ca XML DOM sau SAX.

4. Tooluri

Exista diferite instrumente care permit folosirea sintaxei XPath. Printre acestea amintim XMLSpy si StylusStudio dar si multe altele (inclusiv Java).

La adresa http://www.zrinity.com/developers/xml/xpath/ se regaseste un instrument care permite testarea online a corectitudinii expresiilor XPath relativ la documentele XML pe care aveti de aplicat aceste expresii. Pentru a putea testa diversele expresii XPath, se copiaza continutul documentului XML pe care se doreste sa se lucreze in fereastra pusa la dispozitie pe site (exista un camp in care se afla initial un exemplu de-al lor), iar dupa aceea se introduce expresia XPath in campul de deasupra celui in care ati introdus documentul XML (de asemenea, initial acolo este o expresie XPath). Dupa introducerea acesteia, se apasa butonul de rulare XPath care va calcula corectitudinea documentului XML si a expresiei XPath si va returna rezultatul. **Atentie!** Acest rezultat este returnat in doua feluri. In susul paginii aveti evidentiata portiunea din documentul XML la care se refera expresia XPath, iar dedesubt, aveti rezultatul acestei expresii, sub forma unor obiecte ce sunt repezentari ale elementelor din documentul XML ce sunt returnate.

In continuare va prezentam o implementare in Java care permite rularea expresiilor XPath.

```
public class XMLTest {

	public static void main(String argv[])
	{
		if (argv.length != 1)
		{
			System.err.println("numele fisierului");
			System.exit(1);
		}
		try{

		// cod de la http://xml.apache.org/xalan-j/xpath_apis.html
		//1. Instantiate an XPathFactory.
		  javax.xml.xpath.XPathFactory xpfactory =
				javax.xml.xpath.XPathFactory.newInstance();

		  // 2. Use the XPathFactory to create a new XPath object
		  javax.xml.xpath.XPath xpath = xpfactory.newXPath();

		  // 3. Compile an XPath string into an XPathExpression
		  javax.xml.xpath.XPathExpression expression = xpath.compile("EXPRESIE
XPATH DE TESTAT");

		  // 4. Evaluate the XPath expression on an input document
```

```
                String result = expression.evaluate(new org.xml.sax.InputSource(new
FileInputStream(new File(argv[0]))));
                System.out.println("result is "+result);
               }catch (Exception e){System.out.println(e);}
    }//main
}// class
```

Mai multe detalii referitoare la implementarea XPath in java se pot gasi aici.

Ca exercitiu, aveti de testat urmatoarele expresii atat folosind programelul scris in Java cat si folosind softul online de la adresa de mai sus si comparati rezultatele obtinute:

- grupa
- /grupa
- /grupa/studenti/student
- /grupa/studenti/student[2]
- /grupa//student[2]
- //nume
- //@id
- /grupa//studenti/student[position()<2]
- //student[@id= „1213”]

5. Concluzii

Xpath este un limbaj folosit pentru navigarea prin documentele XML si pentru identificarea si accesarea diferitelor elemente si atribute ale acestuia. Cu ajutorul acestui limbaj se pot regasi foarte usor informatii intr-un document XML. Acest limbaj este extreme de important, el stand la baza altor limbaje, cum ar fi XSLT, XQuery, XLink si XPointer. De asemenea, parserele de XML sunt construite tot pe baza acestui limbaj.

6. Bibliografie

- http://www.w3.org/TR/xpath
- http://www.zvon.org/xxl/XPathTutorial/General/examples.html
- http://www.w3schools.com/xpath/
- http://lambda.uta.edu/cse6331/spring02/**XML**2.ppt
- http://xml.apache.org/xalan-j/xpath_apis.html
- http://www.zrinity.com/developers/xml/xpath/
- http://java.sun.com/j2se/1.5.0/docs/api/javax/xml/xpath/XPathExpression.html#evaluate(org.xml.sax.InputSource,%20javax.xml.namespace.QName)
- http://xml.apache.org/xalan-j/xpath_apis.html#namespacecontext
- http://java.sun.com/j2se/1.5.0/docs/api/javax/xml/xpath/XPath.html#evaluate(java.lang.String,%20java.lang.Object,%20javax.xml.namespace.QName)