Acceso a Datos

UT7 – BASES DE DATOS XML.

LENGUAJE XPATH

1. XPath

XPath (XML Path Language) es un lenguaje que permite construir expresiones que recorren y procesan un documento XML. XPath forma junto con XSLT y XSL-FO, una familia de lenguajes llamadas XSL, diseñados para acceder, transformar y dar formato de salida a los documentos XML.

Un documento XML tiene estructura de árbol, un elemento raíz que tiene hijos, que a su vez tienen otros hijos, etc. A su vez, cada elemento puede tener atributos y/o contenido textual.

Para recorrer un documento XML y extraer la información contenida en sus elementos se usará XPath.

Las especificaciones de XPath, desarrolladas por el W3C, se pueden consultar en http://www.w3.org/TR/xpath-full-text-3/

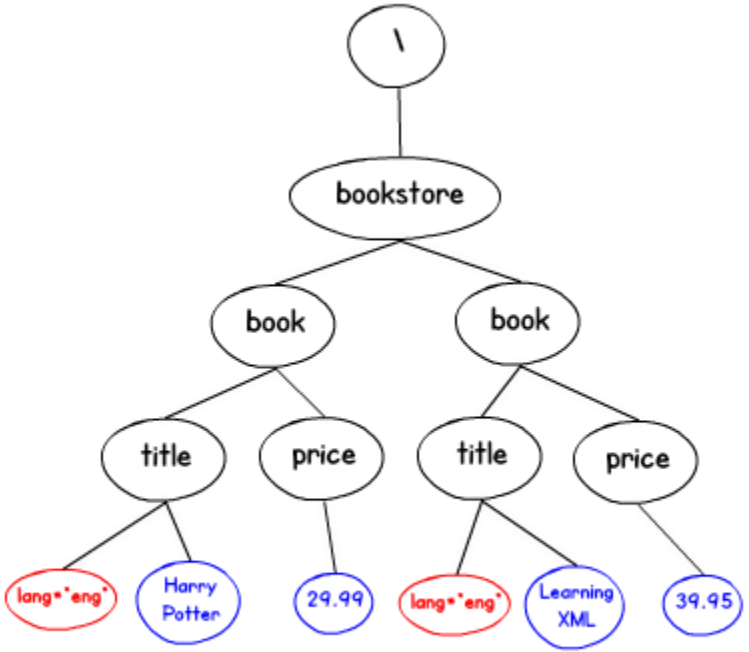
1. XPath

Las direcciones XPath al igual que los direccionamientos en un sistema de ficheros pueden ser absolutas (empiezan por /) o relativas (con respecto a nodo determinado, llamado nodo de contexto).

XPath trata un documento XML como un árbol de nodos (semejante a DOM).

1. XPath

Ejemplo:



1. XPath

Los tipos de nodos en XPath son:

- **Raíz del documento** (uno por documento): contiene todo el documento y se representa por el símbolo /. No tiene nodo padre, y tiene un nodo hijo, el nodo documento. Contiene también los comentarios e instrucciones de procesamiento, que no forman parte del documento en sí.
- **Elemento**: todo elemento de un documento XML es un nodo de XPath.
- **Atributo**: todo atributo de un documento XML es un nodo de XPath.
- **Texto**: todo contenido textual de un elemento es un nodo de XPath.
- **Comentario.**
- **Instrucciones de procesamiento.**
- **Espacios de nombres.**

1. XPath

Los tipos de datos básicos son: string, number, boolean y node-set (conjunto de nodos). Las expresiones XPath más comunes son:

Expresión XPath	Coincidencia
`elemento`	Elemento de nombre `elemento`
`/elemento`	Elemento de nombre `elemento` ubicado en la raíz del documento
`e1/e2`	Elemento `e2` hijo directo del elemento `e1`
`e1//e2`	Elemento `e2` descendiente (hijo, nieto, bisnieto…) del elemento `e1`
`//elemento`	Elemento de nombre `elemento` ubicado en cualquier nivel por debajo de la raíz del documento
`@atributo`	Atributo de nombre `atributo`
`*`	Cualquier elemento
`@*`	Cualquier atributo
`.`	Nodo actual
`..`	Nodo padre
`espNom:*`	Todos los elementos del espacio de nombres de prefijo `espNom`
`@espNom:*`	Todos los atributos del espacio de nombres de prefijo `espNom`

1. XPath

Vamos a utilizar el siguiente XML para los ejemplos: ciclosformativos.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<fp>
<ciclos>
  <ciclo siglas="SMR">
    <nombre>Sistemas microinformáticos y redes</nombre>
    <familiaProfesional>Informática y comunicaciones</familiaProfesional>
    <duracion unidad="horas">2000</duracion>
    <grado>Medio</grado>
  </ciclo>
  <ciclo siglas="DAM">
    <nombre>Desarrollo de aplicaciones multiplataforma</nombre>
    <familiaProfesional>Informática y comunicaciones</familiaProfesional>
    <duracion unidad="horas">2000</duracion>
    <grado>Superior</grado>
  </ciclo>
  <ciclo siglas="DAW">
    <nombre>Desarrollo de aplicaciones web</nombre>
    <familiaProfesional>Informática y comunicaciones</familiaProfesional>
    <duracion unidad="horas">2000</duracion>
    <grado>Superior</grado>
  </ciclo>
  <ciclo siglas="ASIR">
    <nombre>Administrador de sistemas informaticos en red</nombre>
    <familiaProfesional>Informática y comunicaciones</familiaProfesional>
    <duracion unidad="horas">2000</duracion>
    <grado>Superior</grado>
  </ciclo>
</ciclos>
```

```xml
<modulos>
  <modulo codigo="0373">
    <nombre>Lenguajes de marcas y sistemas de gestión de
    información</nombre>
    <duracion unidad="horas">133</duracion>
    <curso>1</curso>
    <ciclos>
      <ciclo siglas="ASIR"/>
      <ciclo siglas="DAM"/>
      <ciclo siglas="DAW"/>
    </ciclos>
  </modulo>
  <modulo codigo="0222">
    <nombre>Sistemas operativos monopuesto</nombre>
    <duracion unidad="horas">140</duracion>
    <curso>1</curso>
    <ciclos>
    <ciclo siglas="SMR"/>
    </ciclos>
  </modulo>
</modulos>
</fp>
```

2. Acceso a elementos/atributos vs acceso a su contenido textual

Cuando se escribe una ruta en XPath, por ejemplo a un cierto elemento /ciclo/nombre, lo que devuelve es el conjunto de nodos llamados nombre, descendientes de ciclo, descendientes del nodo raíz. Si no existiera la ubicación, se devolvería el conjuntovacío.

Para indicar al analizador que lo que se desea es el contenido textual, se utilizará la función **text()**

Ejemplo: //ciclo/nombre/text()

2. Acceso a elementos/atributos vs acceso a su contenido textual

Para acceder a un atributo, por ejemplo DNI de persona, se escribirá /persona/@dni. Si se quiere acceder a su valor, se usará la función **data()** (Esta función no funciona en todos los gestores).

Ejemplo: /ciclo/@siglas/data()

En XPath 3.0 se usa la misma función string()para ambos casos, según sigue:

/ciclo/nombre/string()
/ciclo/@siglas/string()

2. Acceso a elementos/atributos vs acceso a su contenido textual

Cuando se escribe una ruta en XPath, por ejemplo a un cierto elemento /ciclo/nombre, lo que devuelve es el conjunto de nodos llamados nombre, descendientes de ciclo, descendientes del nodo raíz. Si no existiera la ubicación, se devolvería el conjuntovacío.

Para indicar al analizador que lo que se desea es el contenido textual, se utilizará la función **text()**

Ejemplo: //ciclo/nombre/text()

3. Acceso a elementos con rutas simples

Son rutas equivalentes a las que indicaríamos en un direccionamiento absoluto en un sistema Linux, es decir, desde la raíz hasta el elemento del cual queremos extraer la Información.

Todos los elementos hijos del nodo raíz	`/*`
Nombre de ciclos formativos	`/fp/ciclos/ciclo/nombre`
Duración de los módulos profesionales	`/fp/modulos/modulo/duracion`
Elementos que se llamen nombre ubicados en cualquier lugar del documento	`//nombre`
Siglas de los ciclos formativos	`/fp/ciclos/ciclo/@siglas`
Todos los elementos de los módulos	`/fp/modulos/modulo/*`
Todos los atributos de los módulos	`/fp/modulos/modulo/@*`

4. Elementos del lenguaje.

a) Operadores booleanos

Operador	Codificación alternativa
and	
or	
not	
=	
!=	

>	&gt;
<	&lt;
>=	>=
<=	<=

b) Operadores aritméticos

Operador	Significado
+	Suma
-	Resta
*	Multiplicación
div	División
mod	Resto (módulo)

c) Otros operadores

Operador	Significado
|	Operación de conjunto, devuelve la unión de dos conjuntos de nodos

4. Elementos del lenguaje.

Ejemplo: Descendientes de /fp de nombre ciclo y modulo que contengan un descendiente directo nombre cuyo valor se muestra:

```
/fp//(ciclo | modulo)/nombre
```

4. Elementos del lenguaje.

A partir de aquí hay que recurrir a los elementos de XPath, comunes a cualquier lenguaje de programación, que serán necesarios para generar expresiones más complejas.
En particular, los operadores y las funciones.

3.1.1. Funciones numéricas

Función	Devuelve	Ejemplo
round()	Redondeo	round(3.14) = 3
abs()	Valor absoluto	abs(-7) = 7
floor()	Truncamiento	floor(7.3) = 7
ceiling()	Redondeo	ceiling(7.3)=8

4. Elementos del lenguaje.

3.1.2. Funciones de cadena

Función	Devuelve	Ejemplo
substring()	Subcadena	substring('Vivas', 1,4)="Viva"
starts-with()	Cadena comienza por	starts-with('XML', 'X')=true
contains()	Cadena contiene	contains('XML', 'XM')=true
concat	Devuelve la sucesión de los argumentos.	concat("ab","c")="abc"
normalize-space()	Espacios normalizados	normalize-space (' The end ') = 'The end'
translate()	Cambia caracteres en una cadena	Translate ('The end','The', 'An')= 'And end'
string-length()	Longitud de una cadena	String-lenght('Vivas') = 5
upper-case()	Cadena a mayúsculas	upper-case('xml') = 'XML'
lower-case()	Cadena a minúsculas	lower-case('XML') = 'xml'

4. Elementos del lenguaje.

Ejemplo: Nombre de los ciclos cuyas siglas empiecen por D:

/fp/ciclos/ciclo[starts-with(@siglas,'D')]/nombre

4. Elementos del lenguaje.

3.1.3. Funciones que devuelven elementos por su posición

Función	Devuelve
position () = n	Elemento que se encuentra en la n-ésima posición
elemento [n]	Elemento que se encuentra en la n-ésima posición
last()	Elemento último de un conjunto
last()-i	El último elemento – i (ej. Si i=1 → el penúltimo)

4. Elementos del lenguaje.

Ejemplo:

- Todos los datos del segundo ciclo:

  /fp/ciclos/ciclo[position()=2] ó también /fp/ciclos/ciclo[2]

- Todos los datos del último módulo:

  /fp/modulos/modulo[last()]

4. Elementos del lenguaje.

3.1.4. Funciones que devuelven nodos

Función	Devuelve
name()	Nombre del nodo actual
root()	El elemento raíz
node()	Nodos descendientes del actual
comment()	Comentarios
processing-instruction()	Instrucciones de procesamiento

Ejemplo:

a) Devuelve el nombre del nodo actual (en este caso ciclo):

/fp/ciclos/ciclo/name()

4. Elementos del lenguaje.

Ejemplo:

b) Todos los comentarios existentes en el documento:

```
//comment()
```

c) Todas las instrucciones de procesamiento existentes en el documento:

```
//processing-instruction()
```

d) Todos los nodos del árbol cuyo nombre tenga una longitud de cuatro caracteres:

```
//*[string-length(name())=4]
```

4. Elementos del lenguaje.

3.1.5. Funciones de agregado

Función	Devuelve
count()	Conteo de elementos
avg()	Media de valores
max()	Valor máximo
min()	Valor mínimo
sum()	Suma de valores

Ejemplo: Nombre de aquellos módulos que se imparten en dos ciclos:

/fp/modulos/modulo/ciclos[count(ciclo)>=2]/../nombre

5. Filtros de valores literales

Se trata de una extensión de las expresiones anteriores mediante la cual se pueden seleccionar elementos en función del valor de sus atributos o del propio elemento. En este caso los valores serán literales.

Ejemplos:

a) Datos de módulo de código 0373:

/fp/modulos/modulo[@codigo='0373’] (ó si se quiere tratar el literal como un número: /fp/modulos/modulo[@codigo=number(‘0373’)])

5. Filtros de valores literales

Ejemplos:

b) Duración del ciclo de siglas DAM:

```
/fp/ciclos/ciclo[@siglas='DAM']/duracion/string()
```

c) Duración de los módulosimpartidos en el ciclo de siglas DAM:

```
/fp/modulos/modulo/ciclos/ciclo[@siglas='DAM']/../../duración
```

d) Nombre de los ciclos cuyassiglas se encuentren, en orden alfabético entre ASIR y SMR:

```
/fp/ciclos/ciclo[@siglas>'ASIR' and @siglas<'SMR']/nombre
```

6. Consultas XPath anidadas

Las consultas XPath anidadas consisten en incluir una consulta XPath que devuelva un cierto valor dentro de la condición de otra consulta XPath.

Ejemplos:

a) Nombre de los ciclos en los que se cursen módulos de 135 horas o más de duración.

(En los elementos módulo se seleccionan aquellos que tengan una duración de 135 horas o más y se obtienen las siglas de los ciclos en los que se cursan)

/fp/modulos/modulo[duracion>=135]/ciclos/ciclo /@siglas/string()

(Se muestran los nombres de los elementos ciclo cuyassiglas sean iguales a las extraídas en el paso primero)

/fp/ciclos/ciclo[@siglas/string()=/fp/modulos/modulo[duracion>=135]/ciclos/ciclo/@siglas/string()]/nombre

6. Consultas XPath anidadas

Ejemplos:

b) Nombres de módulos que tengan una duración mayor que la de Sistemas Operativos monopuesto. Se usa la función number() para obtener el valor numérico de la duración, de lo contrario haría la comparación como cadenas (donde se cumple que “100”< “20”):

```
/fp/modulos/modulo [number (duracion) >=/fp/modulos/modulo[nombre="Sistemas
                    operativos monopuesto"] /number (duracion) ]/nombre
```

7. Acceso a elementos mediante Ejes

En XPath los ejes son expresiones que permiten acceder a trozos del árbol XML, apoyándose en las relaciones de parentesco entre nodos.

Función	Uso
self::*	Devuelve el propio nodo de contexto. Equivalente a "."
child::	Devuelve los nodos "hijo" del nodo de contexto.
parent::*	Devuelve el nodo padre del nodo contexto. Equivale a ".."
ancestor::*	Devuelve los "antepasados" (padre, abuelo, hasta el nodo raíz) del nodo contexto
ancestor-or-self::*	Devuelve los nodos "antepasados" del nodo de contexto además del propio nodo de contexto
descendant::*	Devuelve los nodos "descendientes" (hijo, nieto…) del nodo contexto
descendant-or-self::*	Devuelve los nodos "descendientes" (hijo, nieto…) del nodo contexto además del propio nodo de contexto. Equivalente a //
following::*	Devuelve los nodos que aparezcan después del nodo de contexto en el documento, excluyendo a los nodos descendientes, los atributos y los nodos de espacio de nombre.
preceding::*	Devuelve los nodos que aparezcan antes del nodo de contexto en el documento, excluyendo a los nodos ascendientes, los atributos y los nodos de espacio de nombre.
preceding-sibling::*	Devuelve los "hermanos mayores" del nodo contexto
following-sibling::*	Devuelve los "hermanos menores" del nodo de contexto
attribute::*	Atributos del nodo contexto. Equivale a @
namespace::*	Espacio de nombres del nodo de contexto

7. Acceso a elementos mediante Ejes

Ejemplos:

- Elementos hermanos menores del primer módulo:
  /fp/modulos/modulo[1]/following-sibling::*

- Descendientes del elemento raíz:
  /fp/descendant::nombre

https://zvon.org/xxl/XPathTutorial/Output_spa/example12.html

Dudas y preguntas