XPath en el aula

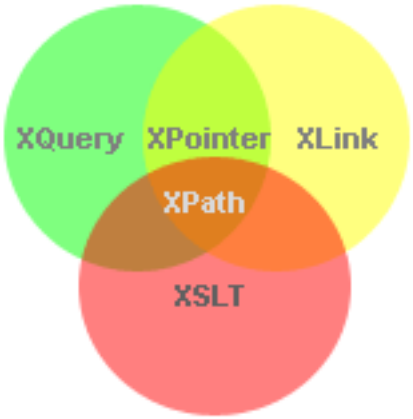
Nieves Carralero Colmenar
I.E.S Ramón y Cajal. Albacete
ncarralero@jccm.es

Resumen

Todo el procesamiento realizado con un fichero XML está basado en la posibilidad de direccionar o acceder a cada una de las partes que lo componen, de modo que podamos tratar cada uno de los elementos de forma diferenciada.

El tratamiento del fichero XML comienza por la localización del mismo a lo largo del conjunto de documentos existentes en el mundo. Para llevar a cabo esta localización de forma unívoca, se utilizan los **URI** (*Unifom Resource Identifiers*), de los cuales los **URL** (*Unifom Resource Locators*) son sin duda los más conocidos.

Una vez localizado el documento XML, la forma de seleccionar información dentro de él es mediante el uso de **XPath**, que es la abreviación de lo que se conoce como *XML Path Language*. Con XPath podremos seleccionar y hacer referencia a texto, elementos, atributos y cualquier otra información contenida dentro de un fichero XML.



XPath en sí es un lenguaje sofisticado y complejo, pero distinto de los lenguajes procedurales que solemos usar (C, C++, Basic, Java...). Además, como casi todo en el mundo de XML, aún está en estado de desarrollo, por lo que no es fácil encontrar herramientas que incorporen todas sus funcionalidades.

XPath es a su vez la base sobre la que se han especificado nuevas herramientas que aprovechar para el tratamiento de documentos XML. Herramientas tales como **XPointer**, **XLink** y **XQL** (el lenguaje que maneja los documentos XML como si de una base de datos se tratase), que también están en estado de desarrollo, pero que sin duda cambiarán el modo en que actualmente concebimos la navegación por la Web. Así, XPath sirve para decir cómo debe procesar una hoja de estilo el contenido de una página XML, pero también para poder poner enlaces o cargar en un navegador zonas determinadas de una página XML, en vez de toda la página.

Según la Orden EDU/2887/2010, de 2 de noviembre, por la que se establece el currículo del ciclo formativo de Grado Superior correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Web, se define un módulo con título: "Módulo Profesional: Lenguajes de marcas y sistemas de gestión de información". En él se pretende hacer una aproximación al trabajo con XML como pieza clave en el desarrollo de aplicaciones Web.

1. INTRODUCCIÓN A XPATH

XPath no es en sí mismo un estándar XML- Digamos que las expresiones XPath no son documentos XML. Mediante el uso de XPath se puede seleccionar elementos o atributos de un elemento XML.

Xpath comienza con la noción *contexto actual*. El contexto actual define el conjunto de nodos sobre los cuales se consultara con expresiones XPath. En general existen cuatro alternativas para determinar el contexto actual para una consulta XPath:

- *./ usa el nodo en el que se encuentra actualmente como contexto actual*
- */ usa la raíz del documento XML como contexto actual*
- *.// usa la jerarquía completa del documento XML desde el nodo actual como contexto actual*
- *// usa el documento completo como contexto actual*

Como mejor se pueden dar los primeros paso con XPath es mediante ejemplos..Para la siguientes explicaciones, partimos del documento books.xml.

Para seleccionar elementos de un XML se realizan avances hacia abajo dentro de la jerarquía del documento. Por ejemplo, la siguiente expresión XPath selecciona todos los elementos Author de Books.xml.

/Books/Book/Author

Si se desea obtener todos los elementos Author del documento se puede usar la siguiente expresión:

//Author

De esta forma uno no se tiene que preocupar de dar la trayectoria completa. Además se pueden usar comodines en cualquier nivel del árbol. Así, por ejemplo, la siguiente expresión selecciona todos los nodos Author que son nietos de Books:

/Books/*/Author

Las expresiones XPath seleccionan un conjunto de elementos, no un elemento simple.Por supuesto, el conjunto puede tener un único miembro, o no tener miembros.

Para identificar un conjunto de atributos se utiliza expresiones iguales para recorrer el árbol que con elementos. La única diferencia es que se usa el carácter @ delante del nombre del atributo. Por ejemplo, la siguiente expresión selecciona todas los atributos Pages de un elemento libro.:

/Books/Book/@Pages

En nuestro documento Books.xml solamente los elementos Book tienen atributos Pages por lo que es igual de valido poner lo siguiente:

//@Pages

También se pueden seleccionar múltiples atributos con el operador @*. Para seleccionar todos los atributos del elemento Book en cualquier lugar del documento se usa la siguiente expresión:

//Book/@*

Además, XPath ofrece la posibilidad de hacer predicados para concretar los nodos deseados dentro del Árbol XML. Esta es un posibilidad similar a la cláusula *where* de SQL. Así, por ejemplo, para encontrar todos los nodos Publisher con el valor *Addison Wesley* se puede usar la siguiente expresión:

/Books/Book/Publisher[.=“Addison Wesley]

Aquí, el operador “[]” especifica un filtro y el operador “.” establece que sea aplicado sobre el nodo actual. Los filtros son siempre evaluados con respecto al contexto actual. Alternativamente, se puede encontrar todos los elementos Book publicados por Addison Wesley:

/Books/Book[./Publisher="Addison Wesley"]

De la misma forma se pueden filtrar atributos igual que elementos.Se pueden usar también operaciones booleanas en los filtros. Por ejemplo, para encontrar todos los Books que tienen mil o más paginas:

/Books/Book[./@Pages>=1000]

Ya que el nodo actual es el contexto se puede simplificar la expresión de la siguiente forma:

/Books/Book[@Pages>=1000]

XPath también soporta un conjunto de funciones. Por ejemplo, para encontrar Books cuyo título comienza por A se puede usar:

/Books/Book[Starts-With(Title,"A")]

Algunas de las funciones XPath son mostradas en la siguiente Tabla:

Function	*Description*
`concat()`	Concatenates strings
`contains()`	Determines whether one string contains another
`count()`	Returns the number of nodes in an expression
`last()`	Specifies the last element in a collection
`normalize-space()`	Removes whitespace from a string
`not()`	Negates its argument
`number()`	Converts its argument to a number
`position()`	Specifies the ordinal of a node within its parent
`starts-with()`	Determines whether one string starts with another
`string-length()`	Returns the number of characters in a string
`substring()`	Returns a substring from a string

Los corchetes son también usados para indicar índices. Las colecciones son indexadas comenzando en 1. Para devolver el primer nodo Book se usaría:

/Books/Book[1]

Para devolver el primer título del segundo libro:

/Books/Book[2]/Title[1]

Para devolver el primer Author en el documento XML, sin respetar el libro:

(/Books/Book/Author)[1]

Los paréntesis son necesarios ya que los corchetes tienen más alta precedencia que un camino. Sin los paréntesis la expresión devolvería el primer autor de todos los Book en el fichero. Hay también una función last() que devuelve el último elemento en una colección, sin preocuparse de cuantos elementos hay en la colección:

/Books/Book[last()]

Otro operador muy útil es la barra vertical "|". Se usa para hacer la unión de dos conjuntos de nodos. La siguiente expresión devuelve todos los autores de libros publicados por Addison wesley o Microsoft Press:

/Books/Book[./Publisher="Addison Wesley"]/Author |

/Books/Book[./Publisher="Microsoft Press"]/Author |

PRÁCTICA 1:

A continuación se muestra una imagen de un documento XML que será el que se usará de ejemplo:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Books>
    <Book Pages="1088">
        <Author>Delaney, Kalen</Author>
        <Title>Inside Microsoft SQL Server 2000</Title>
        <Publisher>Microsoft Press</Publisher>
    </Book>-
    <Book Pages="997">
        <Author>Burton, Kevin</Author>
        <Title>.NET Common Language Runtime</Title>
        <Publisher>Sams</Publisher>
    </Book>
    <Book Pages="392">
        <Author>Cooper, James W.</Author>
        <Title>C# Design Patterns</Title>
        <Publisher>Addison Wesley</Publisher>
    </Book>
</Books>
```

Actividad:

Hacer una clase que ejecute expresiones XPath sobre Books.xml

Resumen

En la práctica se utiliza el método SelectNodes() de XmlNode para evaluar una expresión.

El resultado se almacena en un XmlNodeList que es una colección de nodos XML.

Solución:

El código comentado es el siguiente:

```
public class Cpractica6
    {
        private const String document = "..\\..\\books.xml";
        private const String tXpath = "/Books/Book/Author";


        public static void Main()
        {
            String[] args = {document, tXpath};
            Cpractica6 myCpractica6 = new Cpractica6();
            myCpractica6.Run(args);
        }

        public void Run(String[] args)
        {
            // Carga el Documento Books.xml
            //Esto ya lo conocemos de las práctica anteriores
            XmlTextReader xtr = new
                XmlTextReader(args[0]);
            xtr.WhitespaceHandling = WhitespaceHandling.None;
```

```
                XmlDocument xd = new XmlDocument();
                xd.Load(xtr);
                // Recupera los nodos que coincidan con la expresión
                XmlNodeList xnl = xd.DocumentElement.SelectNodes(args[1]);

                foreach (XmlNode xnod in xnl)
                {
                    // Para cada elemento visualiza la correspondiente entidad tipo Text
                    if (xnod.NodeType == XmlNodeType.Element)
                    {
                        Console.WriteLine(xnod.NodeType.ToString() + ": " +
                            xnod.Name + " = " +
                            xnod.FirstChild.Value);
                    }
                    else
                    {

    Console.WriteLine(xnod.NodeType.ToString()+ ": " +
                            xnod.Name + " = " + xnod.Value);
                    }
                }
                // Cierro
                xtr.Close();
            }
        }
}
```

CONCLUSIONES

Es este artículo se ha mostrado una solución para acceder a documento XML con .NET mediante XPath. Realmente, los elementos usados en esta solución son los básicos para el acceso a XML [XML2][XML3][XML4][XML5], con la salvedad de:

// Recupera los nodos que coincidan con la expresión

XmlNodeList xnl = xd.DocumentElement.SelectNodes(args[1]);

Esta solución, aunque sencilla, está muy limitada, por lo que se aconsejan otras alternativas mucho más potentes para consultar documentos XML como son XPathNavigator.

XPathNavigator: Como ya hemos comentado, con XmlReader se permite movernos por una documento XML. Sin embargo, estamos limitados a movernos solo hacia delante y con acceso de solo lectura. Esto limita la posibilidad de poder ejecutar XPath con esta clase.

Existe un conjunto de clases de navegación en System.Xml.Xpath. En particular, la clase XPathNavigator proporciona un acceso de solo lectura pero con acceso aleatorio sobre un documento XML.

Se puede realizar dos tareas distintas con XPathNavigator:

- Seleccionar un conjunto de nodos con una expresión XPath
- Navegar por la representación DOM de un documento XML

Algunas de las funciones de esta clase son:

`HasChildren`	Property	Indicates whether the current node has any children
`IsEmptyElement`	Property	Indicates whether the current node is an empty element
`Matches()`	Method	Determines whether the current node matches an XSLT pattern
`MoveToFirst()`	Method	Moves to the first sibling of the current node
`MoveToFirstAttribute()`	Method	Moves to the first attribute of the current node
`MoveToFirstChild()`	Method	Moves to the first child of the current node
`MoveToNext()`	Method	Moves to the next sibling of the current node

MoveToNextAttribute()	Method	Moves to the next attribute of the current node
MoveToParent()	Method	Moves to the parent of the current node
MoveToPrevious()	Method	Moves to the previous sibling of the current node
MoveToRoot()	Method	Moves to the root node of the DOM
Name	Property	Specifies the qualified name of the current node
Select()	Method	Uses an XPath expression to select a set of nodes
Value	Property	Specifies the value of the current node
Compile()	Method	Compiles an XPath expression for faster execution
Evaluate()	Method	Evaluates an XPath expression
HasAttributes	Property	Indicates whether the current node has any attributes

En futuros artículos comentaremos estas soluciones.

5. BIBLIOGRAFÍA

[XML desde .NET]: http://zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajos/XML.NET.pdf

[.NET] Framework: http://msdn.microsoft.com/en-us/magazine/cc302158.aspx

[XML tutorial]: http://www.programacion.net/html/xml/

[XML2] Leer XML con XMLTextReader en el aula. Nº 26. MARZO 2011. www.sociedadelainformacion.com

[XML3] XMLNode y XMLDocument en el aula. Nº 27. ABRIL 2011. www.sociedadelainformacion.com

[XML4] Escribiendo XMLTextWriter en el aula. Nº 27. ABRIL 2011. www.sociedadelainformacion.com

[XML5] XMLDocument en el aula . Nº 28. MAYO 2011. www.sociedadelainformacion.com

SOCIEDAD DE LA INFORMACION

www.sociedadelainformacion.com

Edita:

Director: José Ángel Ruiz Felipe

Jefe de publicaciones: Antero Soria Luján

D.L.: AB 293-2001

ISSN: 1578-326x