Leer XML con XMLReader en el aula

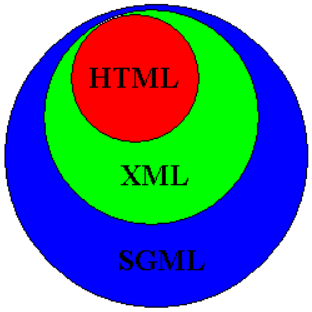
Nieves Carralero Colmenar.
IES Pedro Mercedes. Junta de Comunidades de Castilla-La Mancha. España.
ncarralero@edu.jccm.es

Resumen

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. El presente artículo se centra en ofrecer una buena práctica para el trabajo con XML y .NET usando el objeto XMLReader.

1. INTRODUCCIÓN

Desde hace varios años la Web habla lenguaje XML. XML permite una flexibilidad importante a la hora de desarrollar aplicaciones Web. Esa flexibilidad va desde la presentación de los datos al usuario hasta el almacén de la información que es consultada.



La versión 1.0 del lenguaje XML es una recomendación del W3C de Febrero de 1998. Está basado en el anterior estándar SGML (Standard Generalized Markup Language, ISO 8879), que data de 1986, pero que empezó a gestarse desde principios de los años 70, y a su vez basado en el GML creado por IBM en 1969. Esto significa que aunque XML pueda parecer moderno, sus conceptos están muy asentados y aceptados por la comunidad informática. Está además asociado a la recomendación del W3C DOM (Document Object Model), aprobado también en 1998. Éste no es más que un modelo de objetos (en forma de API) que permite acceder a las diferentes partes que pueden componer un documento XML o HTML.

SGML proporciona un modo consistente y preciso de aplicar etiquetas para describir las partes que componen un documento, permitiendo además el intercambio de documentos entre diferentes plataformas. Sin embargo, el problema que se atribuye a SGML es su excesiva dificultad; baste con pensar que la recomendación ocupa unas 400 páginas.

Así que, manteniendo su misma filosofía, de él se derivó XML como subconjunto simplificado, eliminando las partes más engorrosas y menos útiles. Como su padre, y este es un aspecto importante sobre el que se incidirá después, XML es un *metalenguaje*: es un lenguaje para definir lenguajes. Los elementos que lo componen pueden dar información sobre lo que contienen, no necesariamente sobre su estructura física o presentación, como ocurre en HTML.

Usando SGML, por otro lado, se definió precisamente el HTML, lenguaje muy conocido. La diferencia entre HTML y XML radica en los siguientes puntos: En una primera aproximación se puede decir que mediante XML también podríamos definir el HTML, con lo que podríamos considerar los conjuntos de la figura mostrada abajo. De hecho, HTML es simplemente un lenguaje, mientras que XML como se ha dicho es un metalenguaje, esto es, un lenguaje para definir lenguajes. Y esa es la diferencia fundamental, de la que derivan todas las demás, que iremos viendo a lo largo de este tema.

Desde su creación, XML ha despertado encontradas pasiones, y como para cualquier tema en Internet, hay gente que desde el principio se ha dejado iluminar por sus expectativas, mientras que otras muchas lo han denostado

o simplemente ignorado. Desde 1999 XML se ha convertido en una realidad empresarial palpable. Los programas que lo soportan han crecido del mismo modo exponencial, y a día de hoy no hay empresa de software que se precie que no anuncie la compatibilidad de sus productos más vendidos con este nuevo estándar: Microsoft (.Net), Oracle (Oracle 10i, Web Application Server) o Lotus (Notes) son tres claros ejemplos de ello. Aún más increíble es pensar que hay empresas que se han creado en torno a él, u otras que han movido su actividad hacia su ámbito (de SGML a XML, por ejemplo, como ArborText).

Una cuestión muy común cuando se habla de XML es si es o será el sustituto natural de HTML. La respuesta es no, básicamente XML no ha nacido sólo para su aplicación en Internet, sino que se propone como lenguaje de bajo nivel (a nivel de aplicación, no de programación) para intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo, y casi cualquier cosa que podamos pensar. Sin ir más lejos, algunos lenguajes de los que hablaremos, definidos en XML, recorren áreas como la química y la física, las matemáticas, el dibujo, tratamiento del habla, y otras muchas.

Acceso a XML con XMLReader.

En la siguiente práctica se muestra un ejemplo de lectura de documento XML usando el objeto XMLReader. La salida será los elementos y sus etiquetas mostrados en formato secuencial.

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>
```

Práctica 1:

Práctica 1: Realizar una lectura del documento XML anterior usando XmlTextReader. En la práctica no se distinguen los tipos de elementos ni se leen los atributos. Según la profundidad de los elementos, se hará un sangrado en la salida para mejorar su presentación.

La salida de la práctica 1 es todo el documento XML incluidas las declaraciones y los espacios en Blanco. Sin embargo no incluye los atributos.

Solución:

La Clase XmlReader permite un acceso de sólo lectura (read only) y sólo hacia adelante (forward only). Para hacer esto usa una forma similar a la de un cursor en una tabla de una base de datos. Un puntero señala un nodo de un documento y este cursor puede ser movido para recorre el documento. Esta clase incluye un método Read() que devuelve el siguiente nodo del documento.

La clase XmlReader es una clase abstracta, no se puede crear una instancia de esa clase en una aplicación. Generalmente se usa la clase XmlTextReader que es la encargada de implementar la clase XmlReader con cadenas de texto (Streams)

La funciones de la clase XmlReader son:

Member	*Type*	*Description*
`Depth`	Property	Specifies the depth of the current node in the XML document
`EOF`	Property	Represents a Boolean property that is `true` when the current node pointer is at the end of the XML file
`GetAttribute()`	Method	Gets the value of an attribute
`HasAttributes`	Property	Returns `true` when the current node contains attributes
`HasValue`	Property	Returns `true` when the current node can have a `Value` property
`IsEmptyElement`	Property	Returns `true` when the current node represents an empty XML element
`IsStartElement()`	Method	Determines whether the current node is a start tag
`MoveToElement()`	Method	Moves to the element containing the current attribute
`MoveToFirstAttribute()`	Method	Moves to the first attribute of the current element
`MoveToNextAttribute()`	Method	Moves to the next attribute
`Name`	Property	Specifies a qualified name of the current node
`NodeType`	Property	Specifies the type of the current node
`Read()`	Method	Reads the next node from the XML file
`Skip()`	Method	Skips the children of the current element
`Value`	Property	Specifies the value of the current node

El código comentado es el siguiente:

```
StringBuilder sbNode = new StringBuilder();
String st = "l";

// Creo un nuevo XmlTextReader con el fichero books.xml
XmlTextReader xtr = new XmlTextReader("..\\..\\books.xml");
```

```
// Recorro entero el fichero con Read()
while(xtr.Read())
      {
      sbNode.Length = 0;
      for(int intI=1; intI <= xtr.Depth; intI++) //xtr.Depth:
Profundidad del nodo actual
      {
      sbNode.Append(" ");
      }
      sbNode.Append(xtr.Name + " "); //nombre del elemento. Si no es
      elemento => ""
      st = xtr.NodeType.ToString();
      sbNode.Append(xtr.NodeType.ToString());

      if (xtr.HasValue)
      {
            sbNode.Append(": " + xtr.Value);
      }
//Escribo el nodo en la consola
Console.WriteLine(sbNode.ToString());

}
// Limpio
xtr.Close();
```

Aspectos a tener en cuenta en el aula.

Esta práctica debe entenderse como una primera aproximación al acceso a XML desde .NET. Realmente, aunque esta solución es muy rápida, está muy limitada ya que esta alternativa no permite eliminar, por ejemplo, espacios en blanco, ni distinguir entre los tipos de elementos ni considera los atributos XML.

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/

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