XMLTextReader en el aula

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

Resumen

En este artículo se pretende hacer una aproximación al trabajo con XML como pieza clave en el desarrollo de aplicaciones Web. El artículo se centra en ofrecer una actividad para el trabajo con XML y .NET usando el objeto XMLTextReader aprovechando toda su funcionalidad. 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, XML es una parte importante dentro del "Módulo Profesional: Lenguajes de marcas y sistemas de gestión de información

1. INTRODUCCIÓN

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.

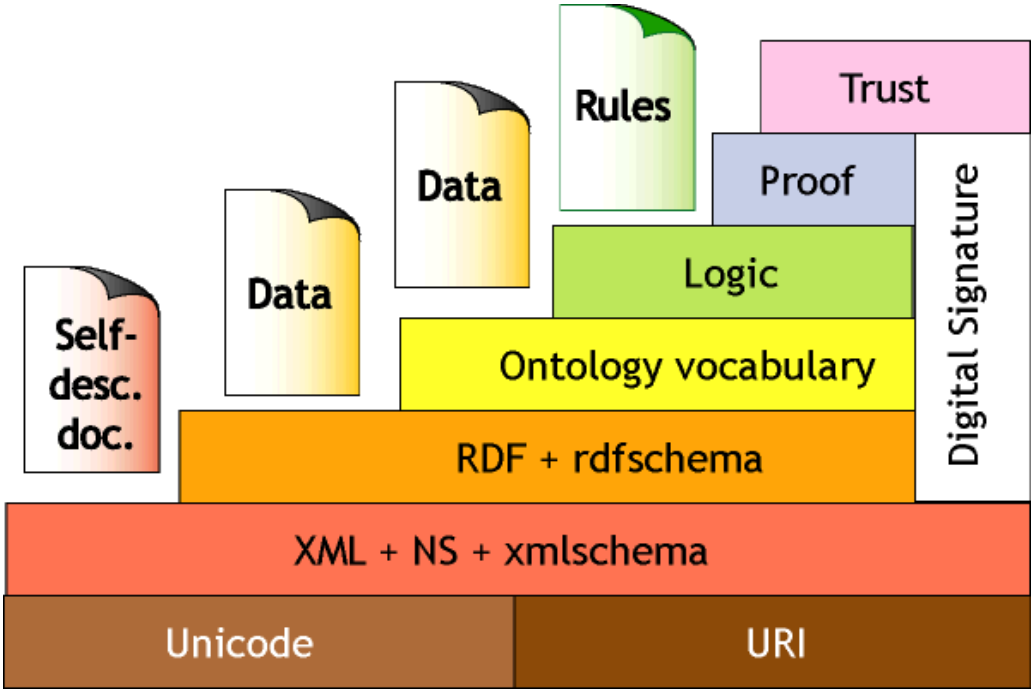
Usando SGML, por otro lado, se definió precisamente el HTML, lenguaje muy conocido. La diferencia entre HTML y XML radica en los siguientes puntos: En una primera aproximación se puede decir que mediante XML también podríamos definir el HTML, con lo que podríamos considerar los conjuntos de la figura 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.

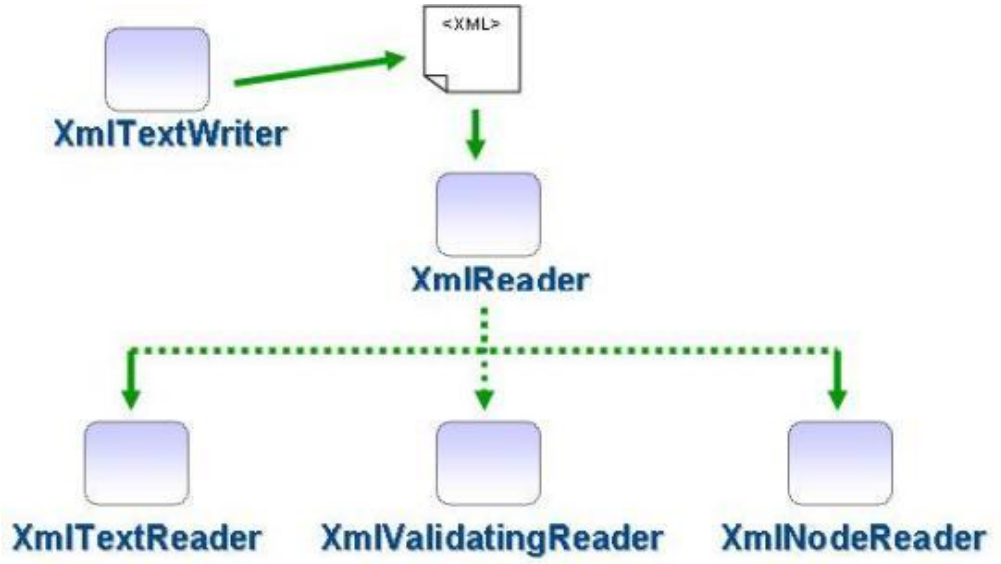
Una imagen que muestra todo lo que deriva de XML es la siguiente:



Actividad: Acceso a XML con XMLTextReader.

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

La relación de XMLTextReader con otras clases de .NET se muestra a continuación:



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:

Realizar una lectura del documento XML anterior usando XmlTextReader. La lectura se hace ordenada, detectando diferentes tipos de elementos y detectando también los atributos del documento.

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

Solución:

La Clase XmlTextReader 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 recorrer 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)

Sin embargo, con sólo XMLTextReader no se puede detectar los tipos de nodos ni los atributos. Por eso en necesario usar la clase XMLNodeType.

Hay que tener en cuenta que:

- Un ítem en un árbol XML representa un nodo.
- Un nodo puede representar espacios en blanco (whitespace), elementos, etc.
- *Los atributos no son considerados como nodos. Le obtienen como una colección asignada a un elemento*.
- DOM distingue esos tipos de nodos asignando un tipo a cada uno de ellos.

- En .NET esos tipos de nodos son listados en una enumeración llamada **XmlNodeType**.
- Los tipos son mostrados en la siguiente tabla.

Las funciones de la clase XmlTextReader son:

Member	*Represents*
Attribute	An XML attribute
CDATA	An XML CDATA section
Comment	An XML comment
Document	The outermost element of the XML document (that is, the root of the tree representation of the XML)
DocumentFragment	The outermost element of an XML document's subsection
DocumentType	A Document Type Description (DTD) reference
Element	An XML element
EndElement	The closing tag of an XML element
EndEntity	The end of an included entity
Entity	An XML entity declaration

EntityReference	A reference to an entity
None	An XmlReader object that has not been initialized
Notation	An XML notation
ProcessingInstruction	An XML processing instruction
SignificantWhitespace	Whitespace that must be preserved to re-create the original XML document
Text	The text content of an attribute, element, or other node
Whitespace	Space between actual XML markup items
XmlDeclaration	The XML declaration

El código comentado es el siguiente:

```
StringBuilder sbNode = new StringBuilder();

// Crea un nuevo XmlTextReader con el fichero
XmlTextReader xtr = new    XmlTextReader(args);
// Recorre el documento XML
while(xtr.Read())
{
      //Saca el tipo de nodo (Elemento o texto)
      if((xtr.NodeType == XmlNodeType.Element) ||
      (xtr.NodeType == XmlNodeType.Text) )
      {
            sbNode.Length = 0;
            for(int intI=1; intI <= xtr.Depth ; intI++) //Lo
visualiza con un sangrado dependiente de la profundidad
            {
                  sbNode.Append(" ");
            }
            sbNode.Append(xtr.Name + " ");
            sbNode.Append(xtr.NodeType.ToString());
            if (xtr.HasValue)
```

```
            {
                  sbNode.Append(": " + xtr.Value);
            }
            Console.WriteLine(sbNode.ToString());
            // Ahora, incorpora los atributos si los hay
            if (xtr.HasAttributes) //devuelve TRUE Si el elemento
contiene atributos
            {
                  while(xtr.MoveToNextAttribute()) //Mientras hay
atributos
                  {
                        sbNode.Length=0;
                        for(int intI=1; intI <= xtr.Depth;intI++)
                        {
                              sbNode.Append(" ");
                        }
                        sbNode.Append(xtr.Name + " ");
                        sbNode.Append(xtr.NodeType.ToString());
                        if (xtr.HasValue)
                        {
                              sbNode.Append(": " +
                                    xtr.Value);
                        }
                        Console.WriteLine(sbNode.ToString());
                  }
            }
      }
}
//Limpiar
xtr.Close();
```

Aspectos a tener en cuenta en el aula.

Esta práctica debe entenderse como una aproximación al acceso a XML desde .NET. Realmente, aunque esta solución es muy rápida, está muy

limitada ya que sólo permite leer XML. Además, la Clase XmlTextReader trata los nodos como parte de una cadena texto.

Como mejora, .NET incluye otra clase llamada XmlNode que es usada para representar un nodo individual de una representación DOM.

Si se instancia una objeto XmlNode para representar una porción determinada de un documento XML, se puede alterar las propiedades de los objetos y escribir los cambios en el fichero original.

A través de la interfaz de XmlNode se puede recuperar o insertar información en las entidades representadas en un objeto XmlNode. O también, se pueden usar métodos para navegar por el DOM. A modo de introducción, algunas de las funciones básicas de XMLNode son:

Member	*Type*	*Description*
`AppendChild()`	Method	Adds a new child node to the end of this node's list of children
`Attributes`	Property	Returns the attributes of the node as an `XmlAttributeCollection` object
`ChildNodes`	Property	Returns all child nodes of this node
`CloneNode()`	Method	Creates a duplicate of this node
`FirstChild`	Property	Returns the first child node of this node
`HasChildNodes`	Property	Returns `true` if this node has any children
`InnerText`	Property	Specifies the value of the node and all its children
`InnerXml`	Property	Specifies the markup representing only the children of this node
`InsertAfter()`	Method	Inserts a new node after this node

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