



XML: eXtensible Markup Languaje

Tecnologías XML

Procesamiento y generación de XML

Dr. Juan Manuel Cueva Lovelle
Departamento de Informática
Universidad de Oviedo
cueva@uniovi.es

Software y estándares para la Web

Esquema

**Grado en
Ingeniería
Informática
del Software**

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

- **Procesamiento de XML**
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Procesamiento XML (I): ¿Qué es un parser XML?

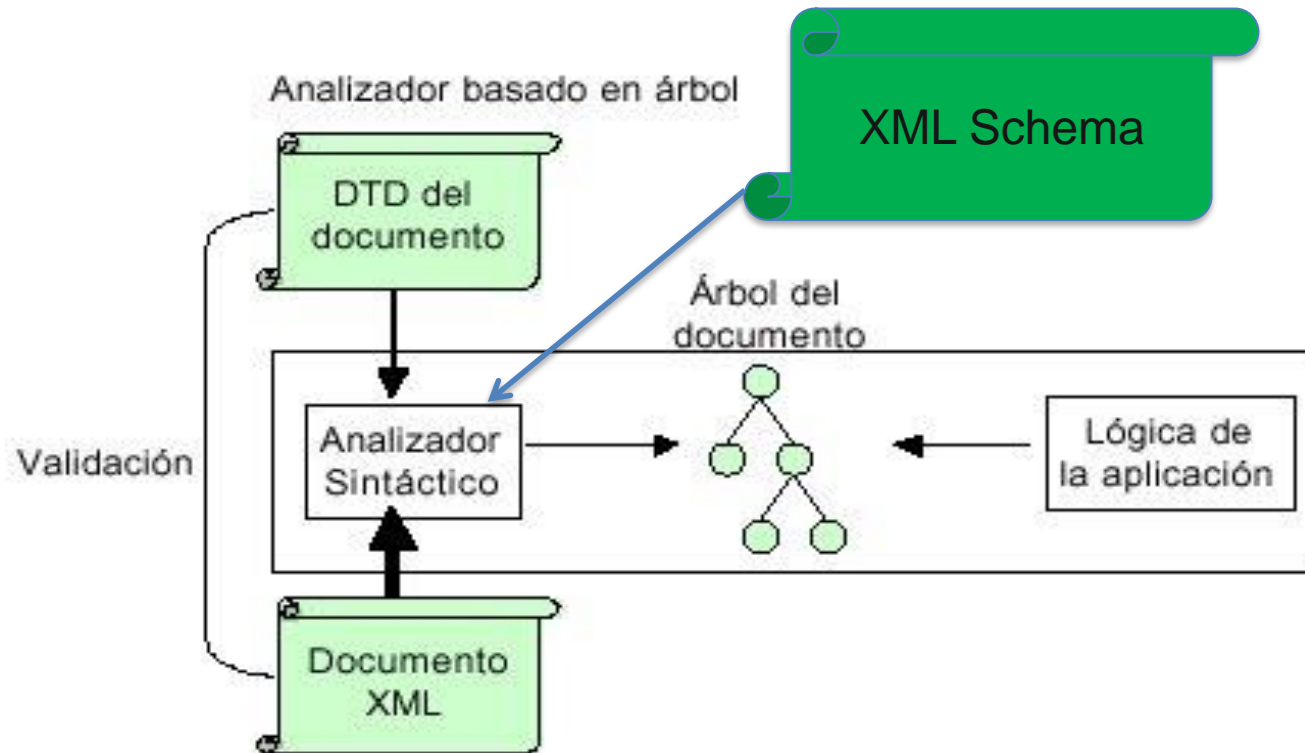
Grado en
Ingeniería
Informática
del Software

- Un **parser** o **procesador** o **analizador sintáctico** procesa el documento XML y verifica que es XML bien formado (y/o válido)
- Es la herramienta principal de cualquier aplicación que use XML.
- Podemos incorporarlos a nuestras aplicaciones, de manera que estas puedan manipular, generar y trabajar con documentos XML

Software y estándares para la Web

Procesamiento XML (II): Uso de *parsers* XML

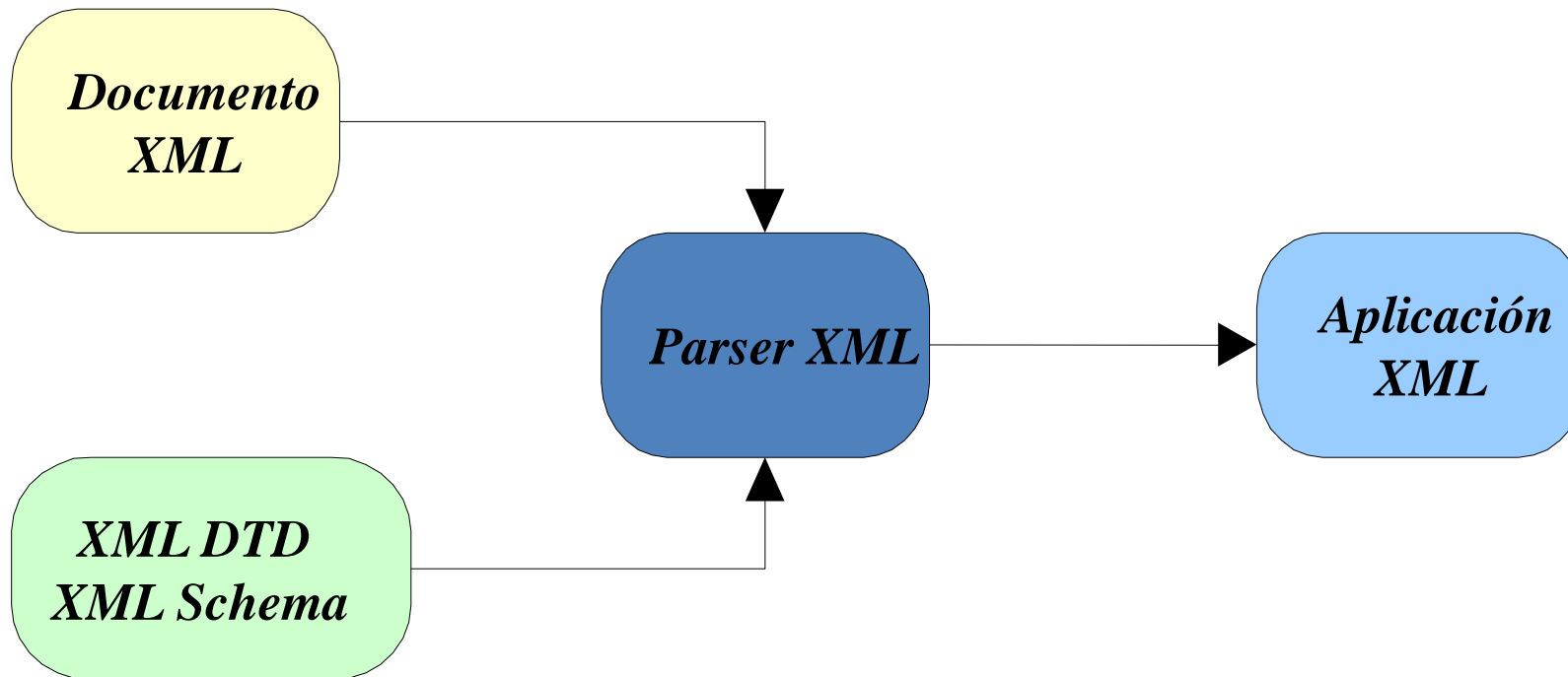
Grado en
Ingeniería
Informática
del Software



Software y estándares para la Web

Procesamiento XML (III): Parsing

Grado en
Ingeniería
Informática
del Software



- **Sin validación**: chequea que el documento esté **bien formado** de acuerdo a las reglas de **sintaxis** de XML
- **Con validación**: además de comprobar que el documento está bien formado, comprueba que éste sea válido utilizando un **DTD** o un **XML Schema**

En el caso de utilizar un DTD, es preferible utilizar un parser con validación

- Los navegadores incluyen sus propios *parsers*.
 - Microsoft Internet Explorer 5 (o superior), que utilizó el parser de MS (incluido en la librería MSXML.DLL)
 - Mozilla, que internamente utiliza el *parser* EXPAT (escrito en C)
- **DOM** y **SAX** son parsers XML que comprueban que el documento esté bien formado y válido
- También hay herramientas que a partir de un archivo XML generan el código necesario para procesarlo.

Software y estándares para la Web

Procesamiento XML (VI): SAX y DOM

Grado en
Ingeniería
Informática
del Software

- Las APIs de SAX y DOM están estandarizadas y existen un gran número de implementaciones para distintos lenguajes (C++, **Java**, C#,etc.)
 - Ej.: Apache Software Foundation proporciona **Crimson** (SAX y DOM sólo para Java), **Xerces** (SAX y DOM) y **Xalan** (XSL)
 - En el caso de Java, familia de paquete **org.xml.sax** y **org.w3c.dom** (básicamente contienen interfaces y clases abstractas)
 - Lo que no está estandarizado es cómo crear instancias de los parsers

Software y estándares para la Web

Procesamiento XML (VII): **JAXP** (Java API for XML Processing)

Grado en
Ingeniería
Informática
del Software

- Forma parte de J2SE a partir de la versión 1.4
- Define un API para trabajar con parsers SAX, DOM y transformaciones XSL
 - Proporciona factorías para crear instancias de parsers y transformadores XSL de manera portable
- Oracle proporciona una implementación de JAXP para versiones anteriores a J2SE 1.4
 - Incluye las APIs **org.xml.sax**, **org.w3c.dom** y **javax.xml**.
 - Incluye Crimson y Xalan como implementaciones por defecto
 - Se pueden usar otras implementaciones vía configuración

Software y estándares para la Web

Procesamiento XML (VIII): Parsers para otros lenguajes

Grado en
Ingeniería
Informática
del Software

- Existen “parsers” para la mayoría de los lenguajes
 - **Clasificación: XML parsers**
 - http://en.wikipedia.org/wiki/Category:XML_parsers
 - **XML Parsing for Java**
 - http://docs.oracle.com/cd/B28359_01/appdev.111/b28394/adx_j_parser.htm#ADXDK3000
 - **What XML parser should I use in C++?**
 - <http://stackoverflow.com/questions/9387610/what-xml-parser-should-i-use-in-c>
 - **How do I parse XML in python?**
 - <http://stackoverflow.com/questions/1912434/how-do-i-parse-xml-in-python>

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- **Procesamiento con C#**
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Procesamiento con C#

**Grado en
Ingeniería
Informática
del Software**

- Existen varias alternativas para procesar documentos XML con C#
 - XmlReader / XmlWriter
 - XmlDocument
 - LINQ (XDocument, XElement,...)

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- **C#: XmlReader**
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

- **XmlReader** proporciona métodos para el acceso rápido a datos
- No posee almacenamiento en caché
- Solamente permite recorrer los nodos hacia delante

Software y estándares para la Web

C#: XmlReader (II): Procesando

Grado en
Ingeniería
Informática
del Software


➡

```
<uno>  
  <dos>  
    <tres>Valor 1</tres>  
  </dos>  
  <dos>  
    <tres>Valor 2</tres>  
  </dos>  
</uno>
```


Software y estándares para la Web

C#: XmlReader (III): Procesando

Grado en
Ingeniería
Informática
del Software



```
<uno>  
  <dos>  
    <tres>Valor 1</tres>  
  </dos>  
  <dos>  
    <tres>Valor 2</tres>  
  </dos>  
</uno>
```

Software y estándares para la Web

C#: XmlReader (IV): Procesando

Grado en
Ingeniería
Informática
del Software

➔

```
<uno>  
  <dos>  
    <tres>Valor 1</tres>  
  </dos>  
  <dos>  
    <tres>Valor 2</tres>  
  </dos>  
</uno>
```

- Tipo de nodo
- Nombre del nodo
- Valor del nodo
- Atributos del nodo

Software y estándares para la Web

C#: XmlReader (V): Implementación

Grado en
Ingeniería
Informática
del Software

1. Construir el XmlReader

```
static void Main(string[] args)
{
    if (args.Length < 1)
        throw (new ArgumentNullException());

    String nombreArchivoXML = args[0];

    // XmlReader
    XmlReader lector = XmlReader.Create(nombreArchivoXML);
```



Software y estándares para la Web

C#: XmlReader (VI): Implementación

Grado en
Ingeniería
Informática
del Software

2. Recorrer los elementos

```
while (lector.Read())  
{  
    Console.WriteLine(lector.Name);  
    Console.WriteLine(lector.Value);  
    Console.WriteLine(lector.NodeType);  
}  
Console.ReadKey();
```

← Salto al siguiente Nodo

Software y estándares para la Web

C#: XmlReader (VII): Implementación

Grado en
Ingeniería
Informática
del Software

3. Recorrer los atributos

```
while (lector.Read())
{
    Console.WriteLine(lector.Name);
    Console.WriteLine(lector.Value);
    Console.WriteLine(lector.NodeType);

    { while (lector.MoveToNextAttribute())
      { Console.WriteLine(" {0}={1}", lector.Name, lector.Value); }
    }
}
Console.ReadKey();
```

Software y estándares para la Web

C#: XmlReader (VIII): Implementación

Grado en
Ingeniería
Informática
del Software

4. Procesar en función del **tipo de nodo XML**

XmlNodeType	Ejemplo	Name	Value	Atrib
Element	<libro> <libro isbn="ISBN-9439234832844">	X		X
EndElement	</libro>	X		
Text	<titulo>El Aleph</titulo>		X	
XmlDeclaration	<?xml version="1.0" ?>	X	X	
Processing Instruction	<?xml-stylesheet type="text/xsl"?>	X	X	
Comment	<!-- Comentario -->		X	

Software y estándares para la Web

C#: XmlReader (IX): Implementación

Grado en
Ingeniería
Informática
del Software

4. Procesar en función del tipo de Nodo XML

```
while (lector.Read())
{
    switch (lector.NodeType)
    {
        case XmlNodeType.Element:
            Console.WriteLine("Element.Name: {0}", lector.Name);
            while (lector.MoveToNextAttribute())
                Console.WriteLine("Attribute.Name {0} Value: {1}", lector.Name, lector.Value);
            break;
        case XmlNodeType.EndElement:
            Console.WriteLine("EndElement.Name: {0}", lector.Name);
            break;
        case XmlNodeType.Text:
            Console.WriteLine("Text.Value: {0}", lector.Value);
            break;
        case XmlNodeType.XmlDeclaration:
            Console.WriteLine("XmlDeclaration.Name: {0} Value: {1}", lector.Name, lector.Value);
            break;
        case XmlNodeType.ProcessingInstruction:
            Console.WriteLine("ProcessingInstruction.Name: {0} Value: {1}", lector.Name, lector.Value);
            break;
        case XmlNodeType.Comment:
            Console.WriteLine("Comment.Value: {0}", lector.Value);
            break;
    };
}
Console.ReadKey();
```

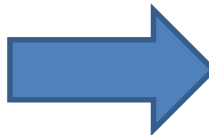
Software y estándares para la Web

C#: XmlReader (X): Implementación. Ejemplo

Grado en
Ingeniería
Informática

- Ejemplo: **TraductorXML**
- Es una aplicación C# que utilizando **XMLReader** es capaz de procesar cualquier archivo XML
- Ejemplos de archivos
 - pizzas.xml
 - libros.xml
 - prueba-1.xml
 - prueba-2.xml

libros.xml



```
Símbolo del sistema - traductorXML libros.xml

C:\Users\Juan Manuel Cueva\Documents\Visual Studio 2013\Projects\TraductorXML\TraductorXML\bin\Debug>traductorXML libros.xml
XmlDeclaration.Name: xml
XmlDeclaration.Value: version="1.0" encoding="ISO-8859-1"
Element.Name: libros
Atributos de <libros>
  xsi:NoNameSchemaLocation=libro.xsd
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
Element.Name: libro
Atributos de <libro>
  isbn=9788420633114
Element.Name: titulo
Text.Value: El Aleph
EndElement.Name: titulo
Element.Name: autor
Text.Value: Jorge Luis Borges
EndElement.Name: autor
Element.Name: any
Text.Value: 1946
EndElement.Name: any
Element.Name: precio
Atributos de <precio>
  moneda=Euro
Text.Value: 7.80
EndElement.Name: precio
Element.Name: editorial
```



```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <libros xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:NoNameSchemaLocation="libro.xsd">
  - <libro isbn="9788420633114">
    <titulo>El Aleph</titulo>
    <autor>Jorge Luis Borges</autor>
    <any>1946</any>
    <precio moneda="Euro">7.80</precio>
    <editorial>Alianza Editorial</editorial>
    <clasificacion>Literatura</clasificacion>
    <idioma>Español</idioma>
  </libro>
  - <libro isbn="9780470036662">
    <titulo>Domain-Specific Modeling. Enabling full code generation</titulo>
    <autor>Steven Kelly</autor>
    <autor>Juha-Pekka Tolvanen</autor>
    <any>2008</any>
    <precio moneda="Dolar USA">74.03</precio>
    <editorial>Wiley</editorial>
    <clasificacion>Informática</clasificacion>
    <idioma>Inglés</idioma>
  </libro>
</libros>
```

Se pueden abrir los archivos XML con un navegador para comprobar que están “bien formados”

Software y estándares para la Web

C#: XmlReader (XII): Implementación. Texto generado (I)

Grado en
Ingeniería
Informática
del Software

```
XmlDeclaration.Name: xml
XmlDeclaration.Value: version="1.0" encoding="ISO-8859-1"
Element.Name: libros
Atributos de <libros>
  xsi:NoNameSchemaLocation=libro.xsd
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
Element.Name: libro
Atributos de <libro>
  isbn=9788420633114
Element.Name: título
Text.Value: El Aleph
EndElement.Name: título
Element.Name: autor
Text.Value: Jorge Luis Borges
EndElement.Name: autor
Element.Name: any
Text.Value: 1946
EndElement.Name: any
Element.Name: precio
Atributos de <precio>
  moneda=Euro
Text.Value: 7.80
EndElement.Name: precio
Element.Name: editorial
Text.Value: Alianza Editorial
EndElement.Name: editorial
Element.Name: clasificacion
Text.Value: Literatura
EndElement.Name: clasificacion
Element.Name: idioma
Text.Value: Español
EndElement.Name: idioma
EndElement.Name: libro
```

Software y estándares para la Web

C#: XmlReader (XIII): Implementación. Texto generado (II)

Grado en
Ingeniería
Informática
del Software

Element.Name: **libro**
Atributos de <libro>
isbn=9780470036662
Element.Name: **titulo**
Text.Value: Domain-Specific Modeling. Enabling full code generation
EndElement.Name: titulo
Element.Name: **autor**
Text.Value: Steven Kelly
EndElement.Name: autor
Element.Name: **autor**
Text.Value: Juha-Pekka Tolvanen
EndElement.Name: autor
Element.Name: **any**
Text.Value: 2008
EndElement.Name: any
Element.Name: **precio**
Atributos de <precio>
moneda=Dolar USA
Text.Value: 74.03
EndElement.Name: precio
Element.Name: **editorial**
Text.Value: Wiley
EndElement.Name: editorial
Element.Name: **clasificacion**
Text.Value: Inform tica
EndElement.Name: clasificacion
Element.Name: **idioma**
Text.Value: Ingl,s
EndElement.Name: idioma
EndElement.Name: libro
EndElement.Name: libros

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO; // Para manejo de archivos
using System.Xml; //Para procesar XML

namespace TraductorXML
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                if (args.Length < 1)
                    throw (new ArgumentNullException());

                String nombreArchivoXML = args[0];

                XmlReader xml = XmlReader.Create(nombreArchivoXML);
```

```
while (xml.Read())
{
    switch (xml.NodeType)
    {
        case XmlNodeType.Element:
            Console.WriteLine("Element.Name: {0}", xml.Name);
            //Obtener los atributos si los tiene
            if (xml.HasAttributes)
            {
                Console.WriteLine("Atributos de <" + xml.Name + ">");
                while (xml.MoveToNextAttribute())
                {
                    Console.WriteLine(" {0}={1}", xml.Name, xml.Value);
                }
            }
            break;
        case XmlNodeType.EndElement:
            Console.WriteLine("EndElement.Name: {0}", xml.Name);
            break;
        case XmlNodeType.Text:
            Console.WriteLine("Text.Value: {0}", xml.Value);
            break;
        case XmlNodeType.XmlDeclaration:
            Console.WriteLine("XmlDeclaration.Name: {0}", xml.Name);
            Console.WriteLine("XmlDeclaration.Value: {0}", xml.Value);
            break;
        case XmlNodeType.ProcessingInstruction:
            Console.WriteLine("ProcessingInstruction.Name: {0}", xml.Name);
            Console.WriteLine("ProcessingInstruction.Value: {0}", xml.Value);
            break;
        case XmlNodeType.Comment:
            Console.WriteLine("Comment.Value: {0}", xml.Value);
            break;
    }
} //fin del switch
} //fin del while
```

```
Console.ReadKey();

} //fin del try
catch (FileNotFoundException)
{
    Console.WriteLine("Error: Archivo {0} no encontrado", args[0]);
}
catch (ArgumentNullException)
{
    Console.WriteLine("Formato correcto de uso:");
    Console.WriteLine("\n\t TraductorXML <archivo>");
}
catch (Exception e)
{
    Console.WriteLine("Error no documentado: "+e);
}
finally
{
    Console.WriteLine("\nGracias por usar TraductorXML");
    Console.WriteLine("Versión 1.0, 29-Noviembre-2011");
    Console.WriteLine("Autor: Juan Manuel Cueva Lovelle");
    Console.WriteLine("http://www.di.uniovi.es/~cueva");
    Console.ReadLine();
}
```

```
    } //fin del Main
} //fin de Program
} // fin namespace
```

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- **C#: XmlDocument**
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

C#: XmlDocument (I)

Grado en
Ingeniería
Informática
del Software

- **XmlDocument** representa un documento XML
- Permite cargar archivos y secuencias XmlReader
- Ofrece diversos métodos para la manipulación del documento
 - Load()
 - GetElementsByTagName()
 - AppendChild()
 - CreateNode()
 - RemoveChild()
 - Save()
 - ...
- A partir del XmlDocument se pueden consultar y modificar los nodos, elementos y atributos del documento XML.

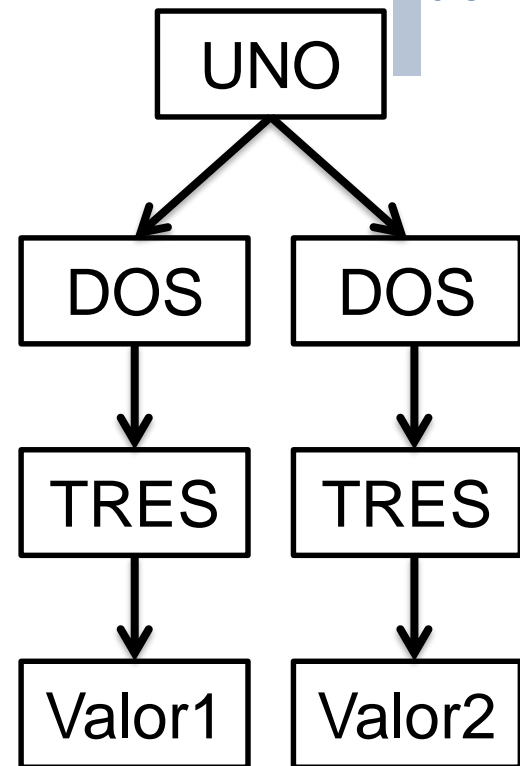
Software y estándares para la Web

C#: XmlDocument (II): Árbol DOM

Grado en
Ingeniería
Informática
del Software

Árbol DOM

```
<uno>  
  <dos>  
    <tres>Valor 1</tres>  
  </dos>  
  <dos>  
    <tres>Valor 2</tres>  
  </dos>  
</uno>
```



```
<uno>  
  <dos>  
    <tres>Valor 1</tres>  
  </dos>  
  <dos>  
    <tres>Valor 2</tres>  
  </dos>  
</uno>
```

doc.GetElementsByTagName("tres");



XmlNodeList
-XmlNode
-XmlNode

<uno>

<dos>

<tres>Valor 1</tres>

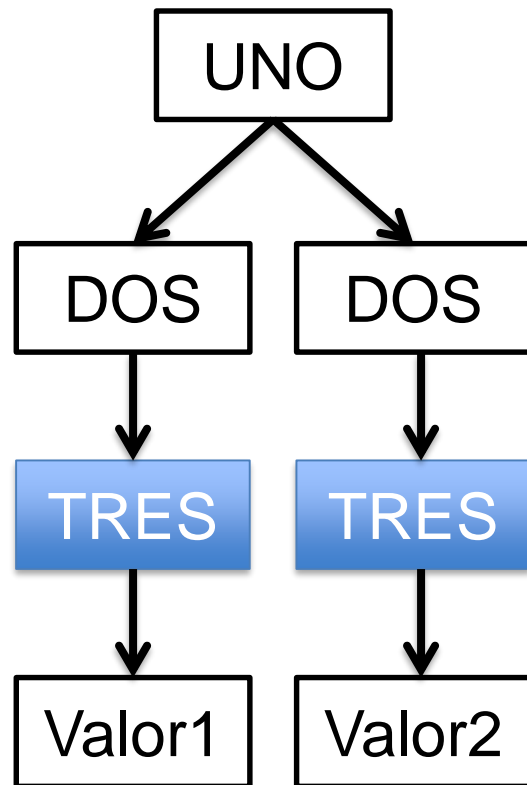
</dos>

<dos>

<tres>Valor 2</tres>

</dos>

</uno>



XmlNodeList[0]

XmlNodeList[1]



XmlNode

¿Cuál es el nombre de los nodos?

¿Cuál es el nombre de XmlNodeList[0].Name ?

Respuesta: **tres**

¿Cuál es el valor de los nodos?

¿Cuál es el valor de XmlNodeList[0].Value ?

Respuesta: **Valor 1**

XmlNodeList[0] FirstChild.Value contiene **Valor 1**

XmlNodeList[1] .FirstChild.Value contiene **Valor 2**

```
<uno>
```

```
<dos>
```

```
<tres>Valor 1</tres>
```

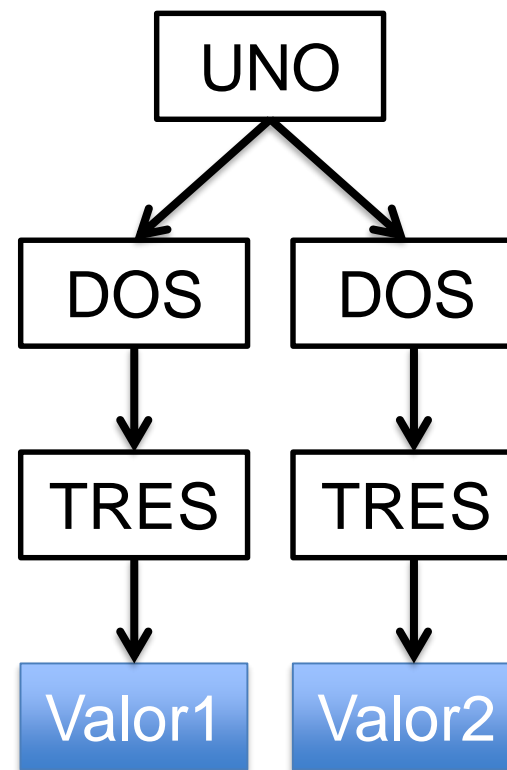
```
</dos>
```

```
<dos>
```

```
<tres>Valor 2</tres>
```

```
</dos>
```

```
</uno>
```



Software y estándares para la Web

C#: XmlDocument (V): Implementación

Grado en
Ingeniería

- Carga de archivo

```
static void Main(string[] args)
{
    String pathLocalArchivo = "libros.xml";

    if (args.Length > 1) {
        pathLocalArchivo = args[0];
    }

    // XmlDocument
    XmlDocument documento = new XmlDocument();
    documento.Load(pathLocalArchivo);
```



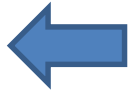
Software y estándares para la Web

C#: XmlDocument (VI): Implementación

Grado en
Ingeniería
Informática
del Software

- Elementos por nombre de etiqueta

```
XmlNodeList listaLibros = documento.GetElementsByTagName("libros");  
XmlNode libros = listaLibros[0];
```

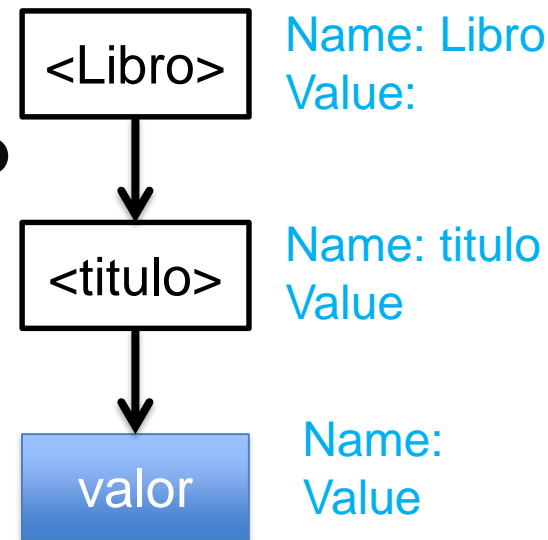


Software y estándares para la Web

C#: XmlDocument (VII): Implementación

Grado en
Ingeniería
Informática
del Software

- Nombre y valor de un elemento



```
XmlNode libros = listaLibros[0];
```

```
foreach (XmlElement libro in libros)  
{
```

```
    Console.WriteLine("Nodo {0}", libro.Name );
```

```
    foreach (XmlElement nodo in libro){
```

```
        Console.WriteLine("Nodo {0}={1}", nodo.Name, nodo.FirstChild.Value);
```

```
    }
```

```
}
```

Software y estándares para la Web

C#: XmlDocument (VIII): Implementación

Grado en
Ingeniería
Informática
del Software

- **Atributos de un elemento**

```
XmlNode libros = listaLibros[0];  
  
foreach (XmlElement libro in libros)  
{  
    Console.WriteLine("\t Atributo ISBN: {0}", libro.GetAttribute("isbn"));  
    foreach (XmlElement nodo in libro){  
        if (nodo.Name == "precio")  
        {  
            Console.WriteLine("\tAtributo Moneda: {0}",nodo.GetAttribute("moneda"));  
        }  
    }  
}
```



Software y estándares para la Web

C#: XmlDocument (IX): Ejercicio

Grado en
Ingeniería
Informática
del Software

- Desarrollar una aplicación C# utilizando **XMLDocument** capaz de procesar todos los nodos y atributos de cualquier archivo XML

```
6 <titulo>El Aleph</titulo>
7 <autor>Jorge Luis Borges</autor>
8 <any>1946</any>
9 <precio moneda="Euro">7.80</precio>
10 <editorial>Alianza Editorial</editorial>
11 <clasificacion>Literatura</clasificacion>
12 <idioma>Español</idioma>
13 </libro>
14
15 <libro isbn="9780470036662">
16 <titulo>Domain-Specific Modeling. Enabling full code
17 <autor>Steven Kelly</autor>
```



```
C:\Users\Jordan\Master Asignaturas\Arquitecturas Software y Procesamiento de Lenguajes\CLASE\...
Nodo libro
  Atributo ISBN: 9780420633114
Nodo titulo=El Aleph
Nodo autor=Jorge Luis Borges
Nodo any=1946
Nodo precio=7.80
  Atributo Moneda: Euro
Nodo editorial=Alianza Editorial
Nodo clasificacion=Literatura
Nodo idioma=Español
Nodo libro
  Atributo ISBN: 9780470036662
Nodo titulo=Domain-Specific Modeling. Enabling full code generation
Nodo autor=Steven Kelly
Nodo autor=Juha-Pekka Tolvanen
Nodo any=2008
Nodo precio=74.03
  Atributo Moneda: Dolar USA
Nodo editorial=Wiley
Nodo clasificacion=Informática
Nodo idioma=Inglés
```

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- **C#: LINQ**
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

- **LINQ (Language-Integrated Query)** conjunto de características presentado en Visual Studio 2008
- **Entre otras muchas cosas LINQ** agrega capacidades de consulta eficaces a la sintaxis de los lenguajes C# y Visual Basic.
- Incluye patrones estándar y de fácil aprendizaje para consultar y actualizar datos.
- Su tecnología se puede extender para utilizar almacenes de datos XML, “LINQ to XML”

- **LINQ to XML** interfaz de programación XML en memoria. Habilitada para LINQ trabajar con XML desde .Net.
- Permite escribir consultas en el documento XML en memoria para recuperar colecciones de elementos y atributos.
- Permite modificar y crear documentos XML en memoria (luego puede ser guardado)
- Es diferente de DOM:
 - Proporciona un nuevo modelo de objetos más ligero y fácil
 - Aprovecha la potencia de los lenguajes .Net

- Cargar un documento XML utilizando XElement

```
XElement documento = XElement.Load(nombreArchivoXML);
```

- Consultar todos los elementos del documento

```
var todosLosElementos = from c in documento.Descendants() select c;  
  
foreach (var elemento in todosLosElementos)  
{  
    Console.WriteLine("Elemento: {0} = {1}", elemento.Name, elemento.Value);  
}
```

- Consultar los elementos que coinciden con un nombre

```
var todosLosTitulos = from c in documento.Descendants("titulo") select c;  
  
foreach (var titulo in todosLosTitulos)  
{  
    Console.WriteLine("Titulo: {0} = {1}", titulo.Name, titulo.Value);  
}
```

- Consultar un atributo de los elementos que coinciden con un nombre

```
var atributoISBN = from c in documento.Descendants("libro")
                   select c.Attribute("isbn").Value;

foreach (var ISBN in atributoISBN)
{
    Console.WriteLine("ISBN: {0}", ISBN);
}
```

- Consultar un elemento , filtrando los elementos con coinciden con un nombre “libro” y tienen un determinado valor en un atributo “isbn”

```
public static void valoresDeElementoConCondicion(XElement documento)
{
    var editorialPorTitulo = from c in documento.Descendants("libro")
                             where c.Attribute("isbn").Value == "9788420633114"
                             select c.Element("editorial");

    foreach (var editorial in editorialPorTitulo)
    {
        Console.WriteLine("Editorial: {0}", editorial.Value);
    }
}
```


- Consultar valores de los elementos que coinciden con un nombre y tienen un elemento hijo con cierto valor.

```
var libroPorTitulo = from c in documento.Descendants("libro")
                      where c.Element("titulo").Value == "El Aleph"
                      select new
                      {
                          Titulo = c.Element("titulo").Value,
                          Editorial = c.Element("editorial").Value,
                          ISBN = c.Attribute("isbn").Value
                      };

foreach (var libro in libroPorTitulo)
{
    Console.WriteLine("Editorial: {0}", libro.Editorial);
    Console.WriteLine("ISBN: {0}", libro.ISBN);
    Console.WriteLine("Titulo: {0}", libro.Titulo);
}
```

- Realizar las siguientes consultas LINQ sobre el archivo libros.xml
 - Seleccionar todos los autores del documento
 - Seleccionar los libros con clasificación literatura
 - Seleccionar los libros con clasificación literatura e idioma Español
 - Seleccionar el valor del título de los libros con idioma Inglés
 - Seleccionar el valor del título y la clasificación de todos los libros

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- **Generación de XML**
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Generación de XML (I)

Grado en
Ingeniería
Informática
del Software

- El resultado de muchas aplicaciones es un archivo XML o un vocabulario XML
- Este archivo puede ser utilizado como entrada de otra aplicación
- Es otra forma de transformar XML

Software y estándares para la Web

Generación de XML (II): Ejemplo **Nikon.LOG** a **.klm**

Grado en
Ingeniería
Informática
del Software

- Algunas cámaras de fotos Nikon con los datos de su sensor GPS generan un archivo **.LOG** con las ubicaciones por donde ha estado la cámara desde que se activó el sensor GPS
- El formato de los datos está en el estándar **NMEA** (*National Marine Electronics Association*) en base a WGS84 (Sistema Geodésico Mundial 1984)
- El archivo **N1612080.LOG** es de la forma:

```
@NikonD5300/ver1.01/wgs84
$GPGGA,114503.00,4316.5479,N,00559.6923,W,1,,,3.1,M,,,,*0B
$GPRMC,114503.00,A,4316.5479,N,00559.6923,W,0.6,,081216,,,A*62
$GPGGA,114522.00,4316.5411,N,00559.6909,W,1,,,2.1,M,,,,*0F
$GPRMC,114522.00,A,4316.5411,N,00559.6909,W,0.1,,081216,,,A*60
$GPGGA,114527.00,4316.5300,N,00559.6971,W,1,,,26.1,M,,,,*34
...
```

Software y estándares para la Web

Generación de XML (III): Archivo N1612080.txt

Grado en
Ingeniería
Informática
del Software

```
Procesamiento del archivo = N1612080.LOG
Cabecera = @NikonD5300/ver1.01/wgs84
[1]$GPGGA,114503.00,4316.5479,N,00559.6923,W,1,,,3.1,M,,,,*0B
Prefijo = $GPGGA
Hora UTC = 11:45:03.00 UTC
Latitud = 43° 16.5479' N
Longitud = 005° 59.6923' W
FixGPS = 1
Numero Satélites =
Precisión de dilución horizontal (HDOP) =
Altitud sobre el nivel del mar = 3.1 M
Altura del Geoide sobre elipsoide WGS84 =
Tiempo desde la última actualización DGPS =
Identificador de la estación de referencia DGPS =
Checksum = *0B
[2]$GPRMC,114503.00,A,4316.5479,N,00559.6923,W,0.6,,081216,,,A*62
Prefijo = $GPRMC
Hora UTC = 11:45:03.00 UTC
Validez = A
Latitud = 43° 16.5479' N
Longitud = 005° 59.6923' W
Velocidad = 0.6 nudos = 1.1112 Km/h
Rumbo =
Fecha UT = 08/12/16 UT
Declinación magnética =
Checksum = A*62
```

. . .

Software y estándares para la Web

Generación de XML (IV): Archivo N1612080.klm

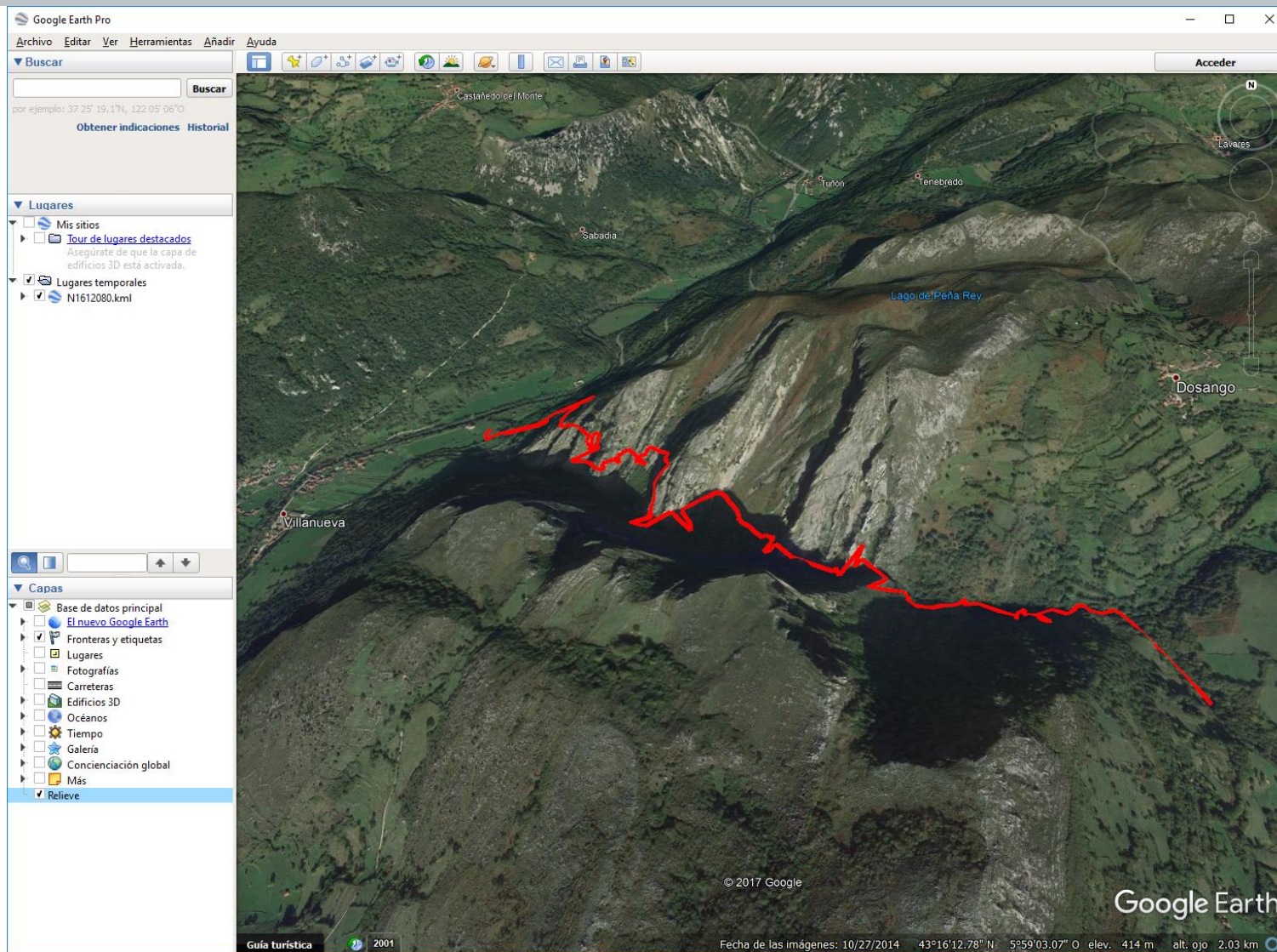
Grado en
Ingeniería
Informática
del Software

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
<name>N1612080.LOG</name>
<LineString>
<extrude>1</extrude>
<tessellate>1</tessellate>
<coordinates>
-5.994871666666667,43.275798333333334,0.0
-5.994871666666667,43.275798333333334,0.0
-5.994848333333334,43.275685,0.0
-5.994848333333334,43.275685,0.0
-5.994951666666667,43.2755,0.0
-5.994951666666667,43.2755,0.0
. . .
-5.973421666666667,43.264635,0.0
</coordinates>
<altitudeMode>relativeToGround</altitudeMode>
</LineString>
<Style id='lineaRoja'>
<LineStyle>
<color>#ff0000ff</color>
<width>5</width>
</LineStyle>
</Style>
</Placemark>
</Document>
</kml>
```


Software y estándares para la Web

Generación de XML (V): Archivo N1612080.klm en Google Earth

Grado en
Ingeniería
Informática
del Software



Software y estándares para la Web

Generación de XML (VI): Programa Nikon-NMEA.py

Grado en
Ingeniería
Informática
del Software

```
C:\WINDOWS\system32\cmd.exe

Nikon-NMEA.py
  Decodifica el archivo de Nikon en formato NMEA y genera un archivo con los datos en formato texto
  Juan Manuel Cueva Lovelle
  Universidad de Oviedo
  Versión 1.0 20-Noviembre-2016
Introduzca el nombre del archivo Nikon      = N1612080.LOG
Introduzca el nombre del archivo generado = N1612080.txt
Presione una tecla para continuar . . .
```

Software y estándares para la Web

Generación de XML (VII): Programa Nikon-NMEA.py

```
# Nikon-NMEA.py
# -*- coding: utf-8 -*-

# Procesado de archivos de GPS de la cámara Nikon
# Genera un archivo con los datos en formato texto
# Juan Manuel Cueva Lovelle
# Universidad de Oviedo
# Versión 1.0 20-Noviembre-2016

def decodificaNMEA(cadena):
    """Nikon-NMEA.py
    Decodifica el archivo de Nikon en formato NMEA y genera un archivo con los datos en formato texto
    Juan Manuel Cueva Lovelle
    Universidad de Oviedo
    Versión 1.0 20-Noviembre-2016"""
    tokens = cadena.split(',')

    prefijo = tokens[0]

    if prefijo=="$GPGGA":
        horaUTC = tokens[1]
        latitud = tokens[2]
        hemisferio = tokens[3]
        longitud = tokens[4]
        esteOeste = tokens[5]
        fixGPS = tokens[6]
        nSatelites = tokens[7]
        precisionHDOP = tokens[8]
        altura = tokens[9]
        metros = tokens[10]
        alturaGeoideSobreElipsoideWGS84 = tokens[11]
        tiempoUpdateDGPS = tokens[12]
        idDGPS = tokens[13]
        checksum = tokens[14]
```

Software y estándares para la Web

Generación de XML (VIII): Programa Nikon-NMEA.py (continuación)

```
resultado = "Prefijo
resultado += "\nHora UTC
resultado += "\nLatitud
resultado += "\nLongitud
resultado += "\nFixGPS
resultado += "\nNumero Satélites
resultado += "\nPrecisión de dilución horizontal (HDOP)
resultado += "\nAltitud sobre el nivel del mar
resultado += "\nAltura del Geoide sobre elipsoide WGS84
resultado += "\nTiempo desde la última actualización DGPS
resultado += "\nIdentificador de la estación de referencia DGPS
resultado += "\nChecksum
return resultado
elif prefijo=="$GPRMC":
    horaUTC = tokens[1]
    valided = tokens[2]
    latitud = tokens[3]
    hemisferio = tokens[4]
    longitud = tokens[5]
    esteOeste = tokens[6]
    velocidad = tokens[7]
    rumbo = tokens[8]
    fechaUT = tokens[9]
    declinacionMagnetica = tokens[10]
    EW = tokens[11]
    checksum = tokens[12]

    resultado = "Prefijo
    resultado += "\nHora UTC
    resultado += "\nValided
    resultado += "\nLatitud
    resultado += "\nLongitud
    resultado += "\nVelocidad
    resultado += "\nRumbo
    resultado += "\nFecha UT
    resultado += "\nDeclinación magnética
    resultado += "\nChecksum
    return resultado
else:
    return "Prefijo NMEA no disponible para ser procesado"

= " + prefijo
= " + horaUTC[0:2] + ":" + horaUTC[2:4] + ":" + horaUTC[4:] + " UTC"
= " + latitud[0:2] + u"\u00B0" + latitud[2:] + "' " + hemisferio
= " + longitud[0:3] + u"\u00B0" + longitud[3:] + "' " + esteOeste
= " + fixGPS
= " + nSatelites
= " + precisionHDOP
= " + altura + " " + metros
= " + alturaGeoideSobreElipsoideWGS84
= " + tiempoUpdateDGPS
= " + idDGPS
= " + checksum

= " + prefijo
= " + horaUTC[0:2] + ":" + horaUTC[2:4] + ":" + horaUTC[4:] + " UTC"
= " + valided
= " + latitud[0:2] + u"\u00B0" + latitud[2:] + "' " + hemisferio
= " + longitud[0:3] + u"\u00B0" + longitud[3:] + "' " + esteOeste
= " + velocidad + " nudos" + " = " + str(float(velocidad) * 1.852) + " Km/h"
= " + rumbo
= " + fechaUT[0:2] + "/" + fechaUT[2:4] + "/" + fechaUT[4:] + " UT"
= " + declinacionMagnetica + " " + EW
= " + checksum
```

Software y estándares para la Web

Generación de XML (IX): Programa Nikon-NMEA.py (continuación)

Grado en
Ingeniería
Informática
del Software

```
def main():
    print(decodificaNMEA.__doc__)
    nombreArchivo = input("Introduzca el nombre del archivo Nikon    = ")
    nombreSalida  = input("Introduzca el nombre del archivo generado = ")

    nLinea=0

    try:
        archivo = open(nombreArchivo,'r')
    except IOError:
        print ('No se encuentra el archivo ', nombreArchivo)
        exit()

    try:
        salida = open(nombreSalida,'w')
    except IOError:
        print ('No se puede crear el archivo ', nombreSalida)
        exit()

    salida.write("Procesamiento del archivo = " + archivo.name)

    cabecera=archivo.readline()
    salida.write("\nCabecera = " + cabecera)

    #datos de GPS en formato NMEA
    while True:
        linea = archivo.readline()
        nLinea=nLinea+1
        if not linea: break
        salida.write "[" + str(nLinea) + "]" + linea
        salida.write(decodificaNMEA(linea))

    archivo.close()
    salida.close()

if __name__ == "__main__":
    main()
```

Software y estándares para la Web

Generación de XML (X): Programa Nikon-NMEA-KML.py

Grado en
Ingeniería
Informática
del Software

```
C:\WINDOWS\system32\cmd.exe

Procesado de archivos de GPS de la cámara Nikon y generación de un archivo KML (Keyhole Markup Language)
KML es un formato de archivo que se utiliza para mostrar datos geográficos en un navegador terrestre.
Se utiliza por Google Earth, Google Maps y Google Maps para móviles.
KML utiliza una estructura basada en etiquetas con atributos y elementos anidados y está basado en el estándar XML

Versión 1.0 20/Noviembre/2016
Juan Manuel Cueva Lovelle. Universidad de Oviedo

Introduzca el nombre del archivo Nikon      = N1612080.LOG
Introduzca el nombre del archivo generado (*.kml) = N1612080
Presione una tecla para continuar . . .
```

Software y estándares para la Web

Generación de XML (XI): Programa Nikon-NMEA-KML.py

```
# Nikon-NMEA-KML.py
# -*- coding: utf-8 -*-
# Procesado de archivos de GPS de la cámara Nikon y generación de un archivo KML (Keyhole Markup Language)
# KML es un formato de archivo que se utiliza para mostrar datos geográficos en un navegador terrestre.
# Se utiliza por Google Earth, Google Maps y Google Maps para móviles.
# KML utiliza una estructura basada en etiquetas con atributos y elementos anidados y está basado en el estándar XML
# Versión 1.0 20/Noviembre/2016
# Juan Manuel Cueva Lovelle. Universidad de Oviedo

def decodificaNMEAonlat(cadena):
    """Decodifica la cadena generada por un GPS en formato NMEA y devuelve un string con la longitud, latitud y altura"""
    tokens = cadena.split(',')
    prefijo = tokens[0]
    if prefijo=="$GPGGA":
        horaUTC = tokens[1]
        latitud = tokens[2]
        hemisferio = tokens[3]
        longitud = tokens[4]
        esteOeste = tokens[5]
        fixGPS = tokens[6]
        nSatelites = tokens[7]
        precisionHDOP = tokens[8]
        altura = tokens[9]
        metros = tokens[10]
        alturaGeoideSobreElipsoideWGS84 = tokens[11]
        tiempoUpdateDGPS = tokens[12]
        idDGPS = tokens[13]
        checksum = tokens[14]
        # pone altura 0.0 sobre el terreno
        return latlonGrados(latitud,hemisferio,longitud,esteOeste,0.0)
    elif prefijo=="$GPRMC":
        horaUTC = tokens[1]
        valided = tokens[2]
        latitud = tokens[3]
        hemisferio = tokens[4]
        longitud = tokens[5]
        esteOeste = tokens[6]
        velocidad = tokens[7]
        rumbo = tokens[8]
        fechaUT = tokens[9]
        declinacionMagnetica = tokens[10]
        EW = tokens[11]
        checksum = tokens[12]
        #pone altura 0.0 sobre el terreno
        return latlonGrados(latitud,hemisferio,longitud,esteOeste,0.0)
    else:
        return "Prefijo NMEA no disponible para ser procesado"
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Generación de XML (XII): Programa Nikon-NMEA-KML.py

Grado en
Ingeniería
Informática
del Software

```
def latlonGrados(latitud,hemisferio,longitud,esteOeste, altura):
    """Convierte dos cadenas string de NMEA a una cadena con las coordenadas en grados con signo"""
    latitudGrados = float(latitud[0:2]) + float(latitud[2:])/60
    if hemisferio == 'S': latitudGrados = - latitudGrados

    longitudGrados = float(longitud[0:3]) + float(longitud[3:])/60
    if esteOeste == 'W': longitudGrados = -longitudGrados

    resultado = str(longitudGrados) + "," + str(latitudGrados) + "," + str(altura)+'\n'
    return resultado

def prologoKML(archivo, nombre):
    """ Escribe en el archivo de salida el prólogo del archivo KML"""

    archivo.write('<?xml version="1.0" encoding="UTF-8"?>\n')
    archivo.write('<kml xmlns="http://www.opengis.net/kml/2.2">\n')
    archivo.write("<Document>\n")
    archivo.write("<Placemark>\n")
    archivo.write("<name>"+nombre+"</name>\n")
    archivo.write("<LineString>\n")
    #la etiqueta <extrude> extiende la línea hasta el suelo
    archivo.write("<extrude>1</extrude>\n")
    # La etiqueta <tessellate> descompone la línea en porciones pequeñas
    archivo.write("<tessellate>1</tessellate>\n")
    archivo.write("<coordinates>\n")

def epilogoKML(archivo):
    """ Escribe en el archivo de salida el epílogo del archivo KML"""

    archivo.write("</coordinates>\n")
    archivo.write("<altitudeMode>relativeToGround</altitudeMode>\n")
    archivo.write("</LineString>\n")
    archivo.write("<Style id='lineaRoja'>\n")
    archivo.write("<LineStyle>\n")
    archivo.write("<color>#ff0000ff</color>\n")
    archivo.write("<width>5</width>\n")
    archivo.write("</LineStyle>\n")
    archivo.write("</Style>\n")
    archivo.write("</Placemark>\n")
    archivo.write("</Document>\n")
    archivo.write("</kml>\n")
```

Software y estándares para la Web

Generación de XML (XIII): Programa Nikon-NMEA-KML.py

```
def main():
    """Procesado de archivos de GPS de la cámara Nikon y generación de un archivo KML (Keyhole Markup Language)
    KML es un formato de archivo que se utiliza para mostrar datos geográficos en un navegador terrestre.
    Se utiliza por Google Earth, Google Maps y Google Maps para móviles.
    KML utiliza una estructura basada en etiquetas con atributos y elementos anidados y está basado en el estándar XML
    Versión 1.0 20/Noviembre/2016
    Juan Manuel Cueva Lovelle. Universidad de Oviedo"""
    print(main.__doc__)
    nombreArchivo = input("Introduzca el nombre del archivo Nikon    = ")
    try:
        archivo = open(nombreArchivo, 'r')
    except IOError:
        print ('No se encuentra el archivo ', nombreArchivo)
        exit()
    nombreSalida = input("Introduzca el nombre del archivo generado (*.kml) = ")
    try:
        salida = open(nombreSalida + ".kml", 'w')
    except IOError:
        print ('No se puede crear el archivo ', nombreSalida + ".kml")
        exit()

    # Procesamiento y generación del archivo kml
    nLinea=0
    # Lectura de la cabecera
    cabecera=archivo.readline()
    # Escribe la cabecera del archivo de salida
    prologoKML(salida, nombreArchivo)
    # Lectura de datos de GPS en formato NMEA
    while True:
        linea = archivo.readline()
        if not linea: break
        salida.write(decodificaNMEAonlat(linea))
    archivo.close()
    epilogoKML(salida)
    salida.close()

if __name__ == "__main__":
    main()
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Esquema

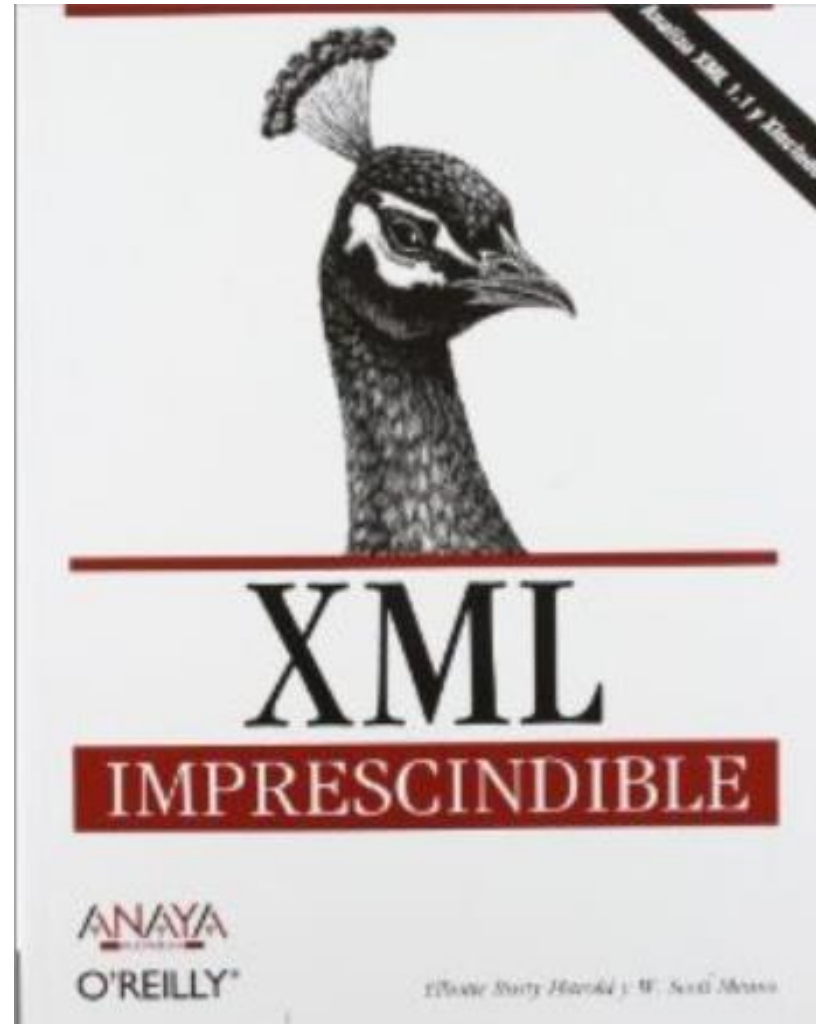
Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- **Bibliografía**
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Bibliografía

- “**XML imprescindible**”
- ANAYA/O'Reilly
(2005)
- E. Rusty Harold y W.
Scott Means



Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

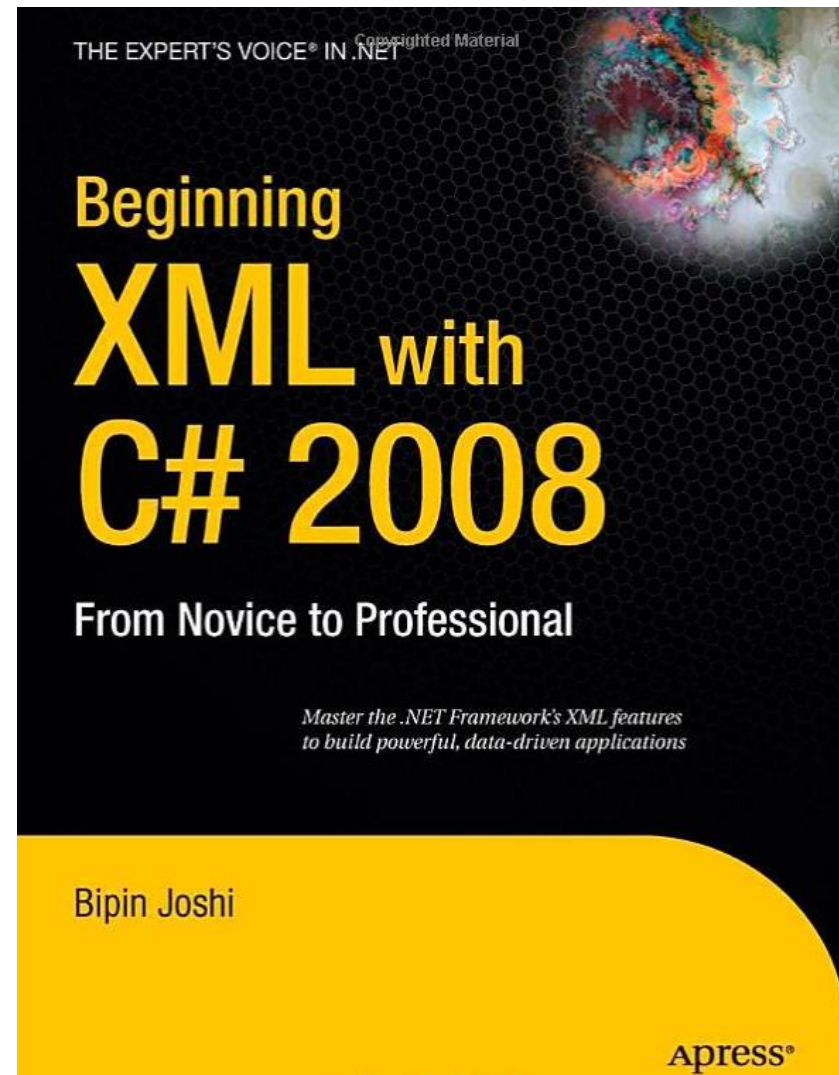
Bibliografía (II)

- Libro recomendado de consulta:
 - “**Beginning XML**”
 - John Wiley & Sons (2012)
 - Joe Fawcett, Liam R.E. Quin, and Danny Ayers



Grado en
Ingeniería
Informática
del Software

- Libro recomendado de consulta:
 - “**Beginning XML with C# 2008**”
 - Apress (2008)
 - Bipin Joshi



Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- **Referencias**
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Referencias (I)

Grado en
Ingeniería
Informática
del Software

- **How to: Parse XML with XmlReader**
 - <http://msdn.microsoft.com/en-us/library/cc189056%28VS.95%29.aspx>
- **Parsing XML in C#: A Quick Working Sample**
 - <http://www.doublecloud.org/2013/08/parsing-xml-in-c-a-quick-working-sample/>
- **Category:XML parsers**
 - http://en.wikipedia.org/wiki/Category:XML_parsers
- **XML Parsing for Java**
 - http://docs.oracle.com/cd/B28359_01/appdev.111/b28394/adx_j_parser.htm#ADXDK3000
- **What XML parser should I use in C++?**
 - <http://stackoverflow.com/questions/9387610/what-xml-parser-should-i-use-in-c>

- **How do I parse XML in python?**
 - <http://stackoverflow.com/questions/1912434/how-do-i-parse-xml-in-python>
- **Estándar W3C**
 - <http://www.w3.org/TR/xml11/>
 - **Especificación del estándar XML en W3C**
 - **Última versión 29-Septiembre-2006**
- **Tutoriales on-line de XML**
 - <http://www.w3schools.com/xml>

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- **Ejercicios resueltos**
- Ejercicios propuestos

Software y estándares para la Web

Ejercicio resuelto: Generador de árboles XML en SVG

Grado en
Ingeniería
Informática
del Software

- Escribir un programa que lea un archivo XML y genere un archivo SVG con la representación del árbol del archivo XML
- En la representación aparecerán todos los elementos y sus atributos
- SVG (Scalable Vector Graphics)
- SVG es un lenguaje definido en XML
- Los archivos **.svg** pueden visualizarse en los navegadores
- SVG es un formato estándar definido por el W3C
 - <http://www.w3.org/Graphics/SVG/>
- Solución en el archivo: **xml2svg.cs**

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Procesamiento de XML
- Procesamiento con C#
- C#: XmlReader
- C#: XmlDocument
- C#: LINQ
- Generación de XML
- Bibliografía
- Referencias
- Ejercicios resueltos
- **Ejercicios propuestos**

Software y estándares para la Web

Ejercicios propuestos (I): Aplicación cuya entrada es un archivo XML

**Grado en
Ingeniería
Informática
del Software**

- Escribir un programa que procese un archivo XML en un lenguaje de programación
- La entrada del programa es un archivo XML
- Se deja libre el diseño de la aplicación
- Se valorará la originalidad y la utilidad de la aplicación desarrollada
- El ejercicio se presentará con
 - Código fuente y ejecutable
 - Archivo leeme.txt indicando la versión y el compilador utilizado, así como una breve descripción de lo que hace el programa
 - Archivos de entrada utilizados en las pruebas

Software y estándares para la Web

Ejercicios propuestos (II): Aplicación cuya salida es un archivo XML

**Grado en
Ingeniería
Informática
del Software**

- Escribir un programa que genere un archivo XML en cualquier lenguaje de programación
- El diseño del programa y la definición del contenido del archivo de salida se deja libre a la creatividad del estudiante
- Se valorará la originalidad y la utilidad de la aplicación desarrollada
- El ejercicio se presentará con
 - Código fuente y ejecutable
 - Archivo leeme.txt indicando la versión y el compilador utilizado, así como una breve descripción de lo que hace el programa
 - Archivos de entrada utilizados en las pruebas
 - Archivos de salida generados
- Se permite la fusión del ejercicio (I) y (II) si la aplicación desarrollada tiene un archivo de entrada XML y genera otro XML