3 - Lenguajes para el almacenamiento y transmisión de la información

OBJETIVOS:

* Conocer los tipos de lenguajes para el almacenamiento y transmisión de información.
* Aprender la sintaxis básica y los posibles elementos de XML.
* Diferenciar entre documentos bien formados y documentos válidos.
* Conocer qué son y para qué se usan los espacios de nombres en XML.

3.1 TIPOS DE LENGUAJES

Dentro de los lenguajes para el almacenamiento y transmisión de la información se pueden encontrar los tipos siguientes:

De marcas. ***XML*** *(eXtended Markup Language).* Es un metalenguaje extensible de etiquetas desarrollado por el W3C que permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto, XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

1.

```xml
<empleados>
  <empleado>
    <nombre>Jorge</nombre>
    <apellido>Mesa</apellido>
    <edad>28</edad>
  </empleado>
  <empleado>
    <nombre>Ana</nombre>
    <apellido>Sánchez</apellido>
    <edad>21</edad>
  </empleado>
  <empleado>
    <nombre>Pedro</nombre>
    <apellido>Lee</apellido>
    <edad>44</edad>
  </empleado>
</empleados>
```

Figura 3.1. Documento XML

XML es una simplificación y adaptación del SGML.

De listas. ***JSON*** *(JavaScript Object Notation).* Es un formato ligero para el intercambio de datos. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML. El mismo ejemplo anterior en JSON:

```json
{
  "empleados": [
      {"empleado": { "nombre":"Jorge" , "apellido":"Mesa", "edad":28 }},
      {"empleado": { "nombre":"Ana" , "apellido":"Sánchez", "edad":21}},
      {"empleado": { "nombre":"Pedro" , "apellido":"Lee","edad":44 }}
      ]
}
```

La ventaja de JSON frente a XML es que los datos ya están en formato ***JavaScript*** y son muy fáciles de procesar:

```javascript
listaEmp=JSON.parse(`
{
  "empleados": [
      {"empleado": { "nombre":"Jorge" , "apellido":"Mesa", "edad":28 }},
      {"empleado": { "nombre":"Ana" , "apellido":"Sánchez", "edad":21}},
      {"empleado": { "nombre":"Pedro" , "apellido":"Lee","edad":44 }}
      ]
}`);
```

ACTIVIDADES 3.1

* Vaya a la página *http:// www.json.org* y obtenga más información sobre JSON.
* Busque información en la web sobre JSON, JSON acrónimo de JavaScript Object Notation (Notación de Objetos de JavaScript) es un formato de texto sencillo para el intercambio de datos.
* Busque información en la web sobre AJAX.

3.2 DEFINICIÓN DE XML

XML se puede ver como un subconjunto de SGML mucho más simple, hasta el punto de que la especificación de XML es de una décima parte que SGML. Por otro lado, el lenguaje XML se puede definir como un metalenguaje, esto es, XML puede ser usado para definir otros lenguajes, al igual que el SGML. Por ejemplo, el lenguaje XHTML, ampliamente usado hoy en día en la *web*, se ha construido usando como base XML.

A continuación, se detallará como se construyen los documentos XML y las características principales de los mismos.

3.3 ESTRUCTURA Y SINTAXIS DE XML

Un documento XML está formado, en principio, por lo que se conoce como "texto plano", esto es, texto en el cual todos los caracteres se representan visualmente, sin existir caracteres no visibles exceptuando los de salto de línea, tabulador o espacio.

Como ya sabemos los documentos escritos usando XML contendrán marcas para separar la información que estructura el documento de la información que se quiere almacenar. Para construir dichas marcas, en XML se usan los caracteres "<" y ">" para delimitar el texto que se desea marcar, mientras que el carácter *"/"* sirve para indicar la etiqueta de finalización del marcado. Un posible ejemplo de documento XML sería el siguiente:

```
<nombre>Luis</nombre>

<persona nombre="Luis"/>
```

Esta construcción se denomina ***elemento*** y constituye la base principal de los documentos XML. Además de los elementos, un documento XML puede contener otros tipos de información. A continuación, se especificarán los más relevantes .

ETIQUETAS, ELEMENTOS Y ATRIBUTOS

Las etiquetas son el componente de XML que permite definir los elementos que conformarán un documento de la siguiente forma:

<etiqueta>Valor</etiqueta>, ej.: *<nombre>Luis</nombre>*

Como se puede observar los elementos se formarán usando una etiqueta de inicio, otra de fin, delimitadas mediante los caracteres "<" y ">", y que comparten el identificador textual pero añadiendo el carácter *"/"* a la etiqueta del final. En medio de las etiquetas de inicio y fin del elemento se representará el contenido que se desee almacenar en ese elemento. Este contenido puede a su vez englobar otros elementos. Por otro lado, los elementos pueden no contener ningún valor, pero en ese caso se deberá usar solamente una etiqueta de finalización que termina en un ***espacio opcional*** y el carácter "/", p.ej.: el elemento <br /> de XHTML.

Se considera que en el elemento XML engloba todo lo que se encuentra entre las correspondientes etiquetas de inicio y de fin y pueden contener tanto otros elementos como simplemente texto o una combinación de ambos.

A continuación, se muestra un ejemplo de XML para representar la ficha de un alumno que sería sintácticamente correcto:

```xml
<alumno>
  <nombre>Luisa</nombre>
  <apellido1>Giménez</apellido1>
  <apellido2>Martínez</apellido2>
  <teléfono>655332244</teléfono>
  <dirección>C/ Gran Vía, 33.</dirección>
  <sexo>Mujer</sexo>
</alumno>
```

XML elements must follow these naming rules:

- Element names are case-sensitive
- Element names must start with a letter or underscore or a colon
- Element names cannot start with the letters xml (or XML, or Xml, etc)
- Element names can contain letters, digits, hyphens, underscores, and periods
- Element names cannot contain spaces

Los siguientes ejemplos serían incorrectos:

<nombre>Luisa</finNombre> <!-- No coincide el nombre de la etiq. de cierre con la de apertura -->
<!apellido1>Giménez<apellido1> <!-- La etiqueta de cierre debe ir al final -->
<sexo> <!-- Una de dos o lleva apertura y cierre o si es un elemento vacío debe llevar una barra al final-->
<xMl_direccion></xMl_direccion> <!-- el nombre de la etiqueta NO puede comenzar con xml en cualquier combinación de mayúsculas o minúsculas -->
<!notas></!notas> <!-- el nombre de un elemento no puede comenzar con el carácter ! -->

Los elementos pueden asimismo contener atributos, los cuales se especificarán en la etiqueta de inicio del elemento. El objetivo de los atributos es poder proporcionar una información adicional sobre un elemento concreto. La sintaxis para representar los atributos consiste en especificar el nombre del atributo dentro de la etiqueta de inicio, a continuación un símbolo "=" y finalmente el valor del atributo delimitado por comillas dobles o por comillas simples:

```
<elem atrib1="val1" atrib2= 'val2' >Valor1</elem>
<elem atrib1="val3" atrib2="val4">Valor2</elem>
```

Siguiendo con el ejemplo presentado previamente, el sexo del alumno anteriormente se había representado usando un elemento mientras que en el siguiente ejemplo se utiliza un atributo para denotar dicha característica. Además se ha añadido el atributo "fechaNacimiento" para el elemento alumno y el atributo "tipo" para el elemento teléfono.

```
<alumno sexo="Mujer" fechaNacimiento="05/06/2002">
  <nombre>Luisa</nombre>
  <apellido1>Giménez</apellido1>
  <apellido2>Martínez</apellido2>
  <teléfono tipo="Móvil">655332244</teléfono>
  <dirección>C/ Gran Vía, 33.</dirección>
</alumno>
```

Se puede observar que generalmente se pueden usar tanto atributos como nuevos elementos para representar información. **Hay que tener en cuenta que los atributos no pueden contener información en forma de árbol**, es decir, no pueden contener otros elementos o atributos tal y como sucede con los elementos. De forma general, se puede establecer la recomendación de no usar atributos en exceso y dejarlos casi exclusivamente para representación de metadatos.

Siguiendo esta recomendación, en el ejemplo tanto el atributo "sexo" como el atributo "fechaNacimiento" se pueden convertir en elementos. Además en este ejemplo se ha añadido un identificador del alumno como atributo:

```
<alumno id="1004">
  <nombre>Luisa</nombre>
  <apellido1>Giménez</apellido1>
  <apellido2>Martínez</apellido2>
  <fechaNacimiento>
    <dia>05</dia>
    <mes>06</mes>
    <año>2002</año>
  </fechaNacimiento>
  <sexo>Mujer</sexo>
  <teléfono tipo="Móvil">655332244</teléfono>
  <dirección>C/ Gran Vía, 33.</dirección>
</alumno>
```

ACTIVIDADES 3.2

En el ejemplo siguiente:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<persona sexo="Mujer" id="Jennifer">
  <nombre>Jennifer López</nombre>
  <email>jlo@gmail.com</email>
  <relacion amigo-de="Gloria Estefan" />
</persona>
```

Indique los elementos y los atributos que haya.

CARACTERES ESPECIALES

En XML algunos símbolos son reservados del lenguaje por lo que para poder representarlos es necesario usar unos códigos, llamados entidades. Estos se definen usando el símbolo & seguido de una palabra clave y terminados por punto y coma. Estas construcciones son entidades predefinidas (las entidades se verán más adelante). A continuación, se detallan algunos de los más relevantes:

Entidad	Carácter
&quot;	"
&amp;	&
&apos;	'
&lt;	<
&gt;	>

INSTRUCCIONES DE PROCESAMIENTO

Más allá de los propios datos contenidos en los ficheros XML y las etiquetas de marcado en un fichero XML se pueden encontrar instrucciones especiales llamadas instrucciones de procesamiento. Éstas comienzan con <? y terminan con ?>. Una de las instrucciones de procesamiento más habituales es la que se usa para indicar que versión de XML se va a usar y cuál es la codificación de caracteres que se va a usar. Por ejemplo, si se usa XML 1.0 y ***utf-8*** la instrucción de procesamiento sería la siguiente:

```xml
<?xml version="1.0" encoding="utf-8"?>
```

COMENTARIOS Y SECCIONES CDATA

Dentro de un documento XML se puede añadir información que no pertenezca ni al marcado ni a la información del documento y que sirve para documentarlo en forma de comentarios internos. La sintaxis de un comentario consta de un texto delimitado por una marca inicial "<!--" y una marca final "-->"

```xml
<!-- Comentario valido en XML -->
```

Los comentarios son elementos especiales y no necesitan ninguna marca de cierre. Además hay que tener en cuenta que dentro de un comentario no se pueden usar dos guiones seguidos "--".

Además en XML se encuentran disponibles las secciones CDATA, que permiten marcar un texto para que éste no sea procesado por el *parser*, es decir, no serán analizadas sintácticamente. CDATA proviene de *"Character DATA"* (Datos de carácter) en contraposición a datos de marcado. La sintaxis de estas secciones se basa en la etiqueta de inicio "<![CDATA[" y la etiqueta de fin "]]>".

En el siguiente ejemplo las dos definiciones del elemento ***alumnos*** serían interpretadas de la misma forma:

```
<alumnos>
  <![CDATA[
  <alumno id="1004">
    <nombre>Luisa</nombre>
  </alumno>
  ]]>
</alumnos>

<alumnos>
  &lt;alumno id=&quot;1004&quot;&gt;
  &lt;nombre&gt;Luisa&lt;/nombre&gt;
  &lt;alumno&gt;
</alumnos>
```

ACTIVIDADES 3.3

Visite la web oficial de XML: *http://www.w3.org/XML/* .

Busque por Internet documentos XML que contengan secciones CDATA.

3.4 DOCUMENTOS XML BIEN FORMADOS

Una vez visto los elementos que pueden formar parte de un documento XML y sus características el siguiente paso será establecer cuando un documento es correcto. En este sentido, en XML se puede hablar de documentos "bien formados" y documentos "válidos".

Los documentos bien formados son aquellos que son sintácticamente correctos, es decir, que cumplen las reglas expuestas en los apartados previos. Sin embargo, los documentos válidos son aquellos que, además de estar bien formados, cumplen los requisitos de una definición de estructura.

Las definiciones de estructura se presentarán en el Capítulo 4.

Un documento está bien formado si cumple con lo siguiente:

- Debe tener la instrucción de procesamiento inicial <?xml version="1.0" encoding="UTF-8"?>
- El documento será de "texto plano" conteniendo caracteres válidos según la codificación indicada al principio (normalmente usaremos ***utf-8***)
- Debe de haber un único elemento raíz
- Todo elemento no vacío tiene que llevar etiqueta de cierre cuyo nombre debe coincidir en mayúsculas y minúsculas (case-sensitive) con la de apertura
- La etiqueta de un elemento vacío debe acabar en />. P.ej. , <br />, <hr/>
- Los nombres de los elementos deben ser válidos: deben empezar por una letra o por _ o **:**, seguidos de cualquier carácter alfanumérico o de _ - . : y que no empiecen por las letras xml en cualquier combinación de mayúsculas y minúsculas. No pueden contener espacios.
- Los valores de los atributos y elementos no pueden contener los caracteres: < > ' " **&** que deben ser sustituidos por sus correspondientes entidades **< > ' " &**

3.5 ESPACIOS DE NOMBRES

Los espacios de nombres se usan para distinguir etiquetas con el mismo nombre en documentos XML distintos de los cuales queremos hacer uso simultáneamente. Se usan de la siguiente manera:

<elemento xmlns:nombreEspacio="URI">

y para usarlos se pone el nombre del espacio delante del nombre de la etiqueta separado por dos puntos:

<nombreEsp:etiq>...</nombreEsp:etiq>Ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<clientes xmlns:banco="http://iespuertasdelcampo.es/clientesBanco.html"
          xmlns:tienda="http://iespuertasdelcampo.es/clientesTienda.html">
  <banco:cliente>
    <nif>12345678Z</nif>
    <cuenta>1234 5678 99 1234567890</cuenta>
    <nombre>Luisa</nombre>
    <ap1>Pérez</ap1>
    <ap2>Rodríguez</ap2>
    <saldo>7.895.323€</saldo>
  </banco:cliente>
  <tienda:cliente>
    <numCliente>2345</numCliente>
    <nombre>Fernando</nombre>
    <apellidos>López Llorente</apellidos>
    <gastos>895€</gastos>
    <puntosAcumulados>234</puntosAcumulados>
  </tienda:cliente>
</clientes>
```

El mismo ejemplo poniendo el atributo xmlns en cada elemento

```
<?xml version="1.0" encoding="UTF-8"?>
<clientes>
  <banco:cliente xmlns:banco="http://iespuertasdelcampo.es/clientesBanco.html" >
    <nif>12345678Z</nif>
    <cuenta>1234 5678 99 1234567890</cuenta>
    <nombre>Luisa</nombre>
    <ap1>Pérez</ap1>
    <ap2>Rodríguez</ap2>
    <saldo>7.895.323€</saldo>
  </banco:cliente>
  <tienda:cliente xmlns:tienda="http://iespuertasdelcampo.es/clientesTienda.html">
    <numCliente>2345</numCliente>
    <nombre>Fernando</nombre>
    <apellidos>López Llorente</apellidos>
    <gastos>895€</gastos>
    <puntosAcumulados>234</puntosAcumulados>
  </tienda:cliente>
</clientes>
```

El mismo ejemplo usando espacios de nombres por defecto:

```
<?xml version="1.0" encoding="UTF-8"?>
<clientes>
  <banco:cliente xmlns="http://iespuertasdelcampo.es/clientesBanco.html" >
    <nif>12345678Z</nif>
    <cuenta>1234 5678 99 1234567890</cuenta>
    <nombre>Luisa</nombre>
    <ap1>Pérez</ap1>
    <ap2>Rodríguez</ap2>
    <saldo>7.895.323€</saldo>
  </banco:cliente>
  <tienda:cliente xmlns="http://iespuertasdelcampo.es/clientesTienda.html">
    <numCliente>2345</numCliente>
    <nombre>Fernando</nombre>
    <apellidos>López Llorente</apellidos>
    <gastos>895€</gastos>
    <puntosAcumulados>234</puntosAcumulados>
```

```
  </tienda:cliente>
</clientes>
```

Uniform Resource Identifier (URI)
A Uniform Resource Identifier (URI) is a string of characters which identifies an Internet Resource.
The most common URI is the Uniform Resource Locator (URL) which identifies an Internet domain address.
Another, not so common type of URI is the Uniform Resource Name (URN).

RESUMEN DEL CAPÍTULO

En este capítulo se ha presentado una introducción a los lenguajes de marcas así como la motivación de su origen y la evolución que han seguido desde su nacimiento.

Por otro lado, se ha introducido la sintaxis básica de XML y se han definido los elementos básicos que pueden aparecer en un documento XML.

A continuación, se ha expuesto qué es un documento XML bien formado y cuándo es válido, dejando este último punto para su profundización en el Capítulo 4.

Por último, se han introducido el concepto de espacios de nombres y su utilidad dentro de un sistema que maneja múltiples documentos XML.

EJERCICIOS PROPUESTOS

1. Definir un documento XML que permita representar un libro. Deberá contener los campos típicos como "título", "autores", "editorial", "fecha de publicación" e "isbn"

2. A partir de la definición anterior escribir un documento XML que al menos contenga 10 entradas de libros.

3. Buscar un validador de XML *online* (por ejemplo el de W3C ***http:/validator.w3.org/***), introducir el documento generado en el ejercicio anterior y comprobar que el documento esté bien formado.

4. Crear un espacio de nombres ficticio e introducirlo en el XML del ejercicio 2 y comprobar que el documento XML sigue estando bien formado.

TEST DE CONOCIMIENTOS

¿Cuál de las siguientes líneas de XML es correcta?

a) <nombre>Luisa<nombre>.
b) <persona nombre="Luisa">.
c) <persona nombre="Luisa"/>.
d) <Nombre="Luisa"></nombre>.

Una sección CDATA sirve para:

a) Poner datos adicionales sobre el tipo de fichero XML.
b) Sirve para añadir metadatos adicionales en la cabecera XML.
c) Escribir un contenido que no va a ser interpretado como XML sino como solo texto.
d) No es un campo válido en XML.

¿Cuál de las siguientes afirmaciones es correcta?

a) Un fichero XML puede tener uno o más nodos raíz.
b) Un fichero XML puede tener un único nodo raíz.
c) Los elementos en XML no distinguen entre mayúsculas y minúsculas.
d) Los caracteres <, > y & se pueden usar sin problema en el contenido de los elementos de un fichero XML.

¿Cuál es la función principal de un espacio de nombres?

a) Poder distinguir elementos con el mismo nombre que provienen de distinta fuente.
b) Asegurar que un documento está bien formado.
c) Asegurar que un documento es válido.
d) No tiene una función específica.