Anexo I Introducción a XML

1 XML

1.1 Un pequeño ejemplo

Antes de nada veamos un pequeño ejemplo de un documento XML. En primer lugar vamos a construir un archivo llamado filmoteca.css que contendrá la definición de estilos a aplicar en el documento. El archivo debe estar en la misma ubicación que va a estar el archivo xml, casablanca.xml.

Los ejemplos aparecen como adjuntos en la presente unidad.

Pero, como ocurre con HTML, no existe consenso universal sobre cómo interpretar archivos CSS, cada cliente Web lo hace a su manera. Abre el archivo con Firefox e Internet Explorer y comprueba la diferencia.

1.2 Características

XML es un lenguaje de marcas que se ha estandardizado y se ha convertido en uno de los formatos más populares para intercambiar información. Es un lenguaje de marcado que deriva del lenguaje SGML, pero mucho más sencillo.

XML ha pasado de se considerado una alternativa a HTML, a convertirse en el lenguaje con mayor impacto en el desarrollo de aplicaciones informáticas para Internet e Intranet.

Las principales características que ofrece XML son:

- **Conjunto de marcas abiertas y ampliables**: podemos definir nuevas marcas para codificar la estructura y contenido de distintos tipos de documentos.

- **Distinción entre la estructura y la presentación de los documentos:** las marcas de un documento XML no indican nada sobre cómo debe presentarse el documento. Para indicar cómo se debe presentar un documento en pantalla o en papel, será necesario crear una hoja se estilo aparte y asociarla al documento. Para crear hojas de estilo para documentos XML disponemos de dos alternativas; las hojas de estilo CSS, ya utilizadas con páginas HTML, y las XSL, diseñadas específicamente para XML.

- **Gestión de hipervínculos avanzada**: los hipervínculos XML pueden crear una relación entre más de dos documentos.

- **Modularidad**: decimos que un documento es modular cuando está formado por distintos archivos XML que se presentan como si se tratara de un único documento.

Existen muchos lenguajes estándares de marcado que derivan de XML, veamos algunos:

- **RSS** (Really Simple Syndication), para producir contenidos sindicables. El ejemplo más claro es los contenido que muestra noticias, a los que nos podemos subscribir.

- **Atom**, semejante a RSS.

- **ePUB**, formato de libro digital. En realidad es una archivo comprimido (ZIP) que contiene tres documentos XML, que son los que especifican la estructura y el contenido del documento.

- **SVG** (Scalable Vector Graphics) Permite definir imágenes vectoriales escalables para poder ser publicadas en una página web.

- **DITA** (Darwin Information Typing Architecture), que se utiliza para producir documentos técnicos.
- **MathML**, pensado para expresiones matemáticas.
- **ODF** u **OD** (Open Document for Office Applications), es un formato abierto que pretende ser un estándar como formato de intercambio entre documentos de oficina. Fue adoptado por el software Open Office.
- **OOXML** (Office Open XML) , formato rival del anterior y propiedad de Microsoft Office.
- **OSDF** (Open Software Description Format), basado en especificaciones de las empresas Marimba y Microsoft, sirve para describir la tecnología y los componentes con los que se ha desarrollado un determinado software, a fin de facilitar el proceso de instalación.
- **RDF** (Reosurce Description Format), sirve para desarrollar documentos que describan recursos. Se utiliza para describir modelos de metadatos. Se basa en los modelos conceptuales, como el modelo entidad / relación.
- **SMIL** (Syncronized Multimedia Integration Language) , utilizado para producir presentaciones de TV en la web.
- **SOAP** (Simple Object Access Protocol), protocolo estándar de comunicación entre objetos utilizados para comunicar servicios web.
- **WDSL** (Web Services Description Language), para definir servicios web.
- **VoiceXML**, se utiliza para representar diálogos vocales entre personas y computadoras.
- **XHTML**, versión de HTML que es compatible con las normas XML.

Existen muchos usos para XML

- **Contenido web**, XML fue defendido por Tim Bernes-Lee. La idea era sustituir a HTML como formato principal en los documentos de la web. Sin embargo el triunfo de HTML 5 ha limitado esta idea.
- **Intercambio de información entre aplicaciones**, el hecho de que XML almacene información mediante documentos de texto plano, facilita que se utilice como estándar, ya que no requiere software especial para leer su contenido. Numerosos servicios en Internet ofrecen resultados de consultas en este formato, el tiempo, resultados de elecciones, datos estadísticos, consultas a bases de datos, etc.
- **Documentación**, es un formato muy defendido para almacenar documentos. El ejemplo más destacado es ePUB.
- **Bases de datos**, todas las grandes bases de datos empresariales son capaces de utilizar XML para extraer datos o incluso como formato fundamental de algunos sistemas gestores de bases de datos. Existen lenguajes como XQuery o XPath, que permiten navegar entre los datos de un documento XML.
- **Formato de imagen vectorial**, el formato SVG es reconocido por todos los navegadores. Son imágenes que no pierden calidad al ampliar o reducirse.
- **Archivos de configuración**, para almacenar información de configuración o de archivos de tipo log es también un formato ideal.

1.3 Tecnologías relacionadas con XML

El ecosistema XML aporta un gran número de tecnologías y lenguajes (la mayoría de lenguajes están también basados en XML) para conseguir obtener más funciones de los documentos XML.

Las tecnologías más importantes son:

- **DTD (**Document Type Definition), definición de tipo de documento. Es un lenguaje que permite especificar reglas que han de cumplir los documentos XML a los que se asocien.
- **XML Schema**. La función que realiza esta tecnología es la misma que la anterior. La diferencia está en que los documentos XML Schema poseen una sintaxis 100% XML (DTD se basa en SGML), por lo que es un formato orientado a reemplazar al anterior.
- **Relax NG**. Otro formato de definición de validaciones para documentos XML. Es una alternativa a las dos anteriores. Tiene una sintaxis más sencilla.
- **Namespacing**, espacios de nombres. Es una norma que permite conseguir nombres de elementos que carecen de ambigüedad. Es decir nombres únicos para los distintos elementos que están dentro de los documentos XML.
- **XPath**. Lenguaje de consulta que permite seleccionar o acceder a partes de un documento XML.
- **CSS,** permiten dar formato a los documentos XML o HTML.
- **XSLT**. Sirve para lo mismo que CSS: dar formato a un documento XML. Tiene muchas más posibilidades que CSS. Tiene la capacidad de convertir un documento XMl en un documento HTML o incluso a tipos comerciales como PDF.
- **XQuery**. Permite consultar datos de los documentos XML, manejándolos como si fueran parte de una base de datos.
- **DOM (**Document Object Model) tecnología que permite acceder a la estructura jerárquica del documento. Normalmente para poder ser utilizada por un lenguaje de programación (como JavaScript) y poder dar dinamismo a su contenido.
- **SAX**. (*Simple API for XML)* permite el uso de herramientas para acceder a la estructura jerárquica del documento XML a través de otro lenguaje. Se usa mucho en la programación de aplicaciones en lenguaje Java.
- **XForms**. Formato de documentos pensados para ser usados como formularios de introducción de datos.
- **XLink**. Permite crear hipervínculos en un documento XML.
- **XPointer**. Semejante al anterior, especifica enlaces a elementos externos al documento XML.

1.4 Cuestiones básicas

Los documentos XML son documentos de texto cuyos metadatos se indican mediante etiquetas (marcas) que permiten clasificar el texto a través de su significado. Dichas etiquetas se deciden a voluntad. La idea es definir un documento cuya forma, significado y estructura se ajusten a nuestras necesidades.

Un ejemplo

```
<alumno>
      <nombre>Luis</nombre>
      <apellidos>Pérez</apellidos>
</alumno>
```

Veamos algunas características:

- Un elemento se define con un nombre entre los símbolos **<** y **>**. Eso forma una etiqueta de apertura.

- ➢ El elemento termina con la etiqueta de cierre, la cual se indica con el símbolo **/** antes del nombre de la etiqueta y todo ello nuevamente encerrado entre los símbolos **<** y **>**.
- ➢ El contenido entre la apertura y el cierre queda afectado por esas etiquetas. De hecho se tratará del contenido del elemento.
- ➢ XML distingue entre mayúsculas y minúsculas, siendo buena práctica escribir las etiquetas en minúsculas.
- ➢ Se pueden espaciar y tabular las etiquetas a voluntad. Es buena práctica que un elemento interno a otro aparezca en el código con una sangría mayor.
- ➢ Los comentarios en el código se inician con los símbolos **<!--** y terminan con -- **>** El texto dentro de esos símbolos no se tienen en cuenta como parte del documento XML y se usa para hacer anotaciones.
- ➢ Según la **W3C** (organismo de estandarización de XML), el texto en un documento XML debe de estar codificado en Unicode (normalmente UTF-8)

1.5 Estructura

Un documento XML puede presentar dos partes diferentes: el prólogo y el cuerpo.

- **Prólogo** es una parte opcional. Es la primera zona del documento y sirve para indicar qué tipo de documento es. Los apartados que puede contener son:
  - Declaración del documento, que permite presentar el tipo de documento, la versión de la norma a la que se adhiere, la codificación de carácter (US-ASCII, UTF-8, UTF-7, UCS-2, EUCJP, Big5, ISO-8859-1, ISO-8859-7, etc. y otras características.

    ```
    <?xml version="1.0" encoding="ISO-8859-1"?>
    ```

    La etiqueta anterior está compuesta de forma diferente al resto de las etiquetas que aparecen en el documento. A los símbolos "<" y ">" se les añade un símbolo de cierre de interrogación. Estas instrucciones se conocen como i*nstrucciones de proceso*.

    La codificación más habitual y la recomendada por el W3C es la UTF-8.
  - Instrucciones de procesamiento, puede incluir instrucciones para indicar un documento para validar el XML, darle formato, u otras funciones

    ```
    <?xml-stylesheet type="text/css" href="filmoteca.css"?>
    ```
- **Cuerpo**, los datos que un documento XML nos ofrece están en lo que se conoce como cuerpo. Es un árbol único de elementos marcados, con anidamiento estricto. En nuestro ejemplo, es todo lo que va entre <pelicula> y </pelicula>, este elemento se conoce como la raíz del elemento. Dentro de un elemento, sea raíz o no, puede haber: más elementos, atributos, texto normal, entidades, comentarios.

1.6 Elementos

Los elementos XML pueden tener contenido (más elementos, caracteres, o ambos a la vez), o bien ser elementos vacíos.

Un elemento con contenido es, por ejemplo:

```
<nombre>Fulano Mengánez</nombre>
```

Para diferenciar entre los diferentes elementos que componen un documento XML hemos de utilizar etiquetas. Las etiquetas están compuestas por un nombre y unos atributos, debe existir una de apertura y otra de cierre y deben de estar correctamente situadas.

Todas las etiquetas que aparecen en nuestros documentos XML deben seguir unas normas muy sencillas:

- Deben de estar delimitadas por los símbolos < y >.
- El nombre de la etiqueta debe comenzar por cualquier letra, un guión bajo (_) o dos puntos (:). A partir de ese momento, podemos utilizar también el guión (-)
- Las mayúsculas y las minúsculas son diferentes, algo que no es cierto en HTML, donde <BR> y <br> son la misma etiqueta. En un documento XML <cliente> y <Cliente> son etiquetas diferentes.
- Cada etiqueta de apertura debe tener una de cierre.

Las etiquetas también pueden estar vacías. Un elemento vacío, es el que no tiene contenido. Se pueden escribir de dos maneras diferentes:

```
<actor></actor>
```

o de forma abreviada:

```
<actor/>
```

1.7 Comentarios

Un documento XML puede contener anotaciones en forma de comentario. Los comentarios no son parte del contenido de información del documento, y pueden ser ignorados por los procesadores XML. Los comentarios se escriben como

<!-- ...*texto del comentario*... -->

El texto de un comentario no puede contener la secuencia --.

1.8 Atributos

Las etiquetas pueden aprovecharse para incluir otros datos, utilizando atributos. Ea HTML ya vimos los atributos. Por ejemplo, la etiqueta HTML para crear un enlace a otra página podría ser el siguiente:

```
<a href=http://www.yahoo.es>Enlace a yahoo</a>
```

En este caso, la etiqueta a se refiere a un enlace y href es un atributo.

En XML no es diferente a HTML, la etiqueta denota el nombre del elemento, y el atributo sus propiedades. La diferencia es que XML es más estricto que HTML.

```
<gato raza="Persa">Micifú</gato>
```

Al igual que en otras cadenas literales de XML, los atributos pueden estar marcados entre comillas verticales (') o dobles ("). Cuando se usa uno para delimitar el valor del atributo, el otro tipo se puede usar dentro. Un elemento puede contener varios atributos.

```
<verdura clase="zanahoria" longitud='15" y media'>
<cita texto="'Hola buenos dias', dijo él">
```

Los atributos se usan para diferenciar entre elementos del mismo tipo. Por ejemplo:

```
<gato raza="Persa">Micifú</gato>
<gato raza="Siames">Milu</gato>
```

1.9 Texto

El texto debe estar siempre entre una etiqueta de apertura y de cierre. Esto significa que todo texto es parte de un elemento XML. Se puede utilizar cualquier carácter Unicode, pero hay que tenerte cuidado utilizar caracteres que podrían dar lugar a confusión como < y >.

1.10 Anidamiento de elementos

El contenido de los elementos no está limitado a solo texto; los elementos pueden contener otros elementos, que a su vez pueden contener texto u otros elementos, y así sucesivamente.

Un documento XML es un árbol de elementos. No hay límite para la profundidad del árbol, además de que los elementos pueden repetirse.

Al elemento que está dentro de otro se le conoce como hijo. El elemento en el que éste está contenido se conoce como padre.

```
<nombre>
    <nombre-pila>Juan</nombre-pila>
    <apellido>Pérez</apellido>
</nombre>
```

El documento debe tener un solo elemento raíz. Dicho de otro modo, todos los elementos del documento deben ser hijos de un solo elemento.

1.11 Entidades predefinidas

En XML 1.0, se definen cinco entidades para representar caracteres especiales y que no se interpreten como marcado por el procesador XML. Es decir, que así podemos usar el carácter "<" sin que se interprete como el comienzo de una etiqueta XML, por ejemplo.

Las entidades son:

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

1.12 Secciones CDATA

Existe otra construcción en XML que permite especificar datos, utilizando cualquier carácter, especial o no, sin que se interprete como marcado XML. Las secciones CDATA se utilizan para indicarle al analizador que un determinado fragmento del documento XML no debe ser analizado. Existen caracteres que no pueden ser incluidos directamente como contenido, como son (<) (>) o (&), para los que tenemos que usar expresiones equivalentes.

En ocasiones, utilizar estos equivalentes no resulta práctico ni deseable. El caso más frecuente que se suele presentar es aquel en el que queremos incluir código fuente en algún lenguaje de programación, como por ejemplo JavaScript, o incluso utilizar HTML.

Un elemento CDATA debe comenzar con <![CDATA[y terminar con]]>

```
<ejemplo>
      <![CDATA[
            <html>
            <head><title>Rock & Roll</title></head>
            </html>
      ]]>
</ejemplo>
```

1.13 Elaboración de documentos bien formados

Se dice que un documento XML está bien formado cuando cumple las reglas sintácticas indicadas. Los procesadores XML pueden rechazar cualquier documento que no esté bien formado.

Los documentos que no están bien formados, se les considera incorrectos y así serán marcados por los parsers de XML.

Aunque un documento XML esté bien formado, puede ser no válido. Los documentos validos cumplen la sintaxis de un documento validador, que es lo que vamos a explicar en la Unidad 4.

Las reglas que debe cumplir un documento bien formado son:

Reglas generales

- Dentro del texto no se pueden utilizar los caracteres especiales <, >, &, ni las comillas doble ni simples. Para poder utilizarlos deben ir incluidos en secciones CDATA o utilizar entidades.
- Los espacios en blanco adicionales, tabuladores y saltos de línea no se mostrarán.

Reglas para los elementos

- Siempre se debe cerrar antes el último elemento abierto.
- Todos los elementos poseen etiquetas de apertura y cierre. Si la etiqueta no tiene contenido, se debe cerrar en la propia etiqueta.
- Todos los elementos XML deben contener un único elemento raíz que contendrá al resto de elementos.
- Los nombres de los elementos comienzan con letras y pueden ir seguidos de letras, números, guiones o de puntos. No pueden contener espacios en blanco.

Reglas para los atributos

- Los nombres de los atributos siguen las mismas reglas que los de los elementos.
- Los atributos solo se pueden colocar en etiquetas de apertura y nunca en las de cierre.
- Los valores de los atributos deben ir entrecomillados (simples o dobles)
- Todos los atributos deben tener valor asociado.
- Ninguna etiqueta tiene dos atributos con el mismo nombre.

1.14 Utilización de espacios de nombres en XML

El poder de XML proviene de su flexibilidad, del hecho de que podamos definir nuestras propias etiquetas para describir nuestros datos. Veamos un ejemplo:

```
<direccion>
      <nombre>
       <titulo>Sra.</titulo>
       <nombre-pila>Maria</nombre-pila>
       <apellidos>Martin López</apellidos>
      </nombre>
   <calle>Mayor 14</calle>
   <ciudad>Madrid</ciudad>
      <codigo-postal>34829</codigo-postal>
</direccion>
```

El documento incluye el elemento <titulo> para el título o tratamiento de cortesía, una elección perfectamente razonable como nombre de un elemento. Si creamos una librería online, podría elegir el crear un elemento <titulo> para almacenar el título de un libro. Todas son elecciones razonables, pero todas ellas crean elementos con el mismo nombre. ¿Cómo saber si, dado un elemento <titulo> se refiere a una persona o a un libro ? Esto se soluciona con los *namespace*s.

Para usar un namespace, definimos un prefijo *namespace* y lo mapeamos a una cadena particular.

Así es cómo se deberían definir prefijos namespace para nuestros dos elementos <titulo>:

```
<?xml version="1.0"?>
<resumen_usuario
  xmlns:direccion="http://www.xyz.com/direcciones/"
  xmlns:libros="http://www.zyx.com/libros/"
>
... <direccion:nombre><titulo>Mrs.</titulo> ... </direccion:nombre> ...
... <libros:titulo>El Señor de los Anillos</libros:titulo> ...
```

En este ejemplo, los dos prefijos del namespace son direccion y libros.

Hay que darse cuenta que la definición de un namespace para un elemento particular significa que todos los elementos hijos pertenecen al mismo namespace. El primer elemento <titulo> pertenece al namespace direccion debido a que su elemento padre <direccion:Nombre>, ya pertenecía a ese namespace.

El punto final: La cadena de una definición de namespace es solo una cadena. Si, esa cadena parece una URL, pero no lo es. Podríamos definir xmlns:direccion="pepe" y funcionaría exactamente igual, Lo único que importa acerca de la cadena del namespace es que sea única; por esto la mayor parte de las definiciones de namespace parecen URLs. Los parser XML no van a la dirección http://www.zyx.com/libros/ para buscar una DTD o un Esquema, simplemente usan esos textos como cadenas.

1.15 Herramientas software

Disponemos de las siguientes herramientas para comprobar que un documento este bien formado:

- Un cliente Web
- Un editor de XML
- Un validador online: https://validator.w3.org/