TEMA 4 - XML

Eduardo martín-Sonseca

Contenido

[1. Introducción 2](#_Toc125960468)

[2. Tipos de XML 2](#_Toc125960469)

# Introducción

XML es un lenguaje de marcas, y sirve para transmitir o almacenar información, **JSON** se encarga de transmitir o almacenar información, se utiliza mucho en el envió de formularios, deriva de JavaScript, XML deriva de HTML, la sintaxis es similar a la de HTML, un **parser** sirve para analizar la información de un XML.

# Tipos de XML

* **XHML**, es una adaptación a HTML
* **RDF** Sirve para describir recursos
* **RSS (Es el que más se utiliza)** Ejemplo de tipos de XML

# Usos de XML

Es el intercambio de información

* El **DTD** sirve para definir la estructura de datos
* **XML Schema** es mas actual que el DTD
* **XPATH** es un lenguaje que permite sacar información de un XML
* **XQUERY** es un lenguaje que permite sacar información de un XML de forma más compleja

<persona> es una etiqueta

<nombre>Jorge</nombre>

<apellido>Sánchez</apellido>

</persona>

Codificación XML guardar los datos en código ascii

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="stylesheet.xsl"?> Es la sintaxis a formar dentro del XML

Estructura de un documento XML

* **Elemento raíz**, siempre tiene que haber un elemento raíz
* **Atributos,**
* **Texto normal** Información general
* **Entidad** es una constante

<?xml version="1.0" encoding="UTF-8" ?>

<alumno sexo="varon" fechaNacimiento="5/6/1990">

<nombre>Pablo</nombre>

<apellido>Pérez</apellido>

<telefono tipo="movil">622443322</telefono>

<direccion>Ronda de Burgos

<direccion>Ronda de Burgos 83</direccion>

</alumno>

<?xml version="1.0" encoding="UTF-8" ?>

<alumno sexo="varon" fechaNacimiento="5/6/1990">

<nombre>Pablo</nombre>

<apellido>Pérez</apellido>

<telefono tipo="movil">622443322</telefono>

<?xml version=”1.0” encoding="UTF-8"?>

<documento>

<título>Prueba </título>

<ejemplo>

<![CDATA[

En HTML la negrita se escribe: <strong>

]]>

</ejemplo>

</documento>

Las secciones CDATA sirve para comentar

<!ENTITY dam "Desarrollo de Aplicaciones Multiplataforma">

Una entidad es un nombre <ciclo>&dam;</ciclo>

&gt; Símbolo >

&lt; Símbolo <

&amp; Símbolo &

&quot; Símbolo “

&apos; Símbolo '

<nombre sexo="Hombre">Antonio</nombre>

<nombre sexo='Mujer'>Sara</nombre>

<nombre sexo='Mujer">Eva</nombre>

Eso está mal escrito, por que cierra con comillas dobles

Si se coloca un elemento en un atributo, tiene que tener valor

Un documento valido el xml tiene que estar bien interpretado por el parser

# Atributos

<xs:attribute

name="nombre del atributo"

type="tipo global de datos"

ref="declaración del elemento global"

form="cualificación" <!--qualified o unqualified -->

id="identificador"

default="valor por defecto"

fixed="valor fijo"

use="uso" <!-- prohibited, optional o required -->

>

- Tipos de datos complejos

Pueden ser globales o locales,

* los **globales** se pueden utilizar en todo el archivo
* Los **locales** se utilizan dentro del elemento en si, aunque estos ultimos puede tener atributos globales, estos tipos de datos complejos se identifican con la etiqueta complexType.Estos tienen distintos tipos de contenido:

- Contenido simple

<xs:element name="nombre" type="xs:string"/>

- Contenido simple (de texto y atributo)

<xs:element name="documento">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute name="idioma" type="xs:string"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

- Contenido vacio

<xs:element name="casado">

<xs:complexType>

<xs:attribute name="valor" type="xs:string" />

</xs:complexType>

</xs:element>

- Contenido compuesto

Hay distintos tipos

o Secuencias

Dentro de un elemento es habitual indicar su contenido como una secuencia de elementos. Esto se hace con la etiqueta sequence, dentro de la cual se añade una etiqueta element para cada elemento que forma la secuencia.

Ejemplo:

<xs:simpleType name="emailT">

<xs:restriction base="xs:string">

<xs:pattern value="[A-Za-z]{3,}@.{3,}"></xs:pattern>

</xs:restriction>

</xs:simpleType>

<xs:element name="email">

<xs:complexType >

<xs:sequence>

<xs:element name="remite" type="emailT"/>

xs:element name="para" type="emailT" minOccurs="1"

maxOccurs="unbounded"/>

<xs:element name="CC" type="emailT" minOccurs="0"

maxOccurs="unbounded"/>

<xs:element name="CCO" type="emailT" minOccurs="0"

maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

En el ejemplo, el elemento email está compuesto de cuatro elementos. El remite (que tiene obligatoriamente que aparecer una vez), el para que aparecerá al menos una vez y que puede aparecer tantas veces como se desee y los apartados opcionales CC y CCO que pueden aparecer repetidos.

La etiqueta sequence posee los atributos minOccurs y maxOccurs para indicar que el bloque de elementos se puede repetir.

o Elecciones

Sirven para permitir elegir uno de entre varios elementos. Para ello se utiliza la etiqueta choice.

Ejemplo:

<xs:element name="identificación">

<xs:complexType>

<xs:choice>

<xs:element name="firma" type="xs:NCName"/>

<xs:element name="código" type="xs:NCName" />

</xs:choice>

</xs:complexType>

</xs:element>

En el ejemplo, el elemento identificación, consta de dos posibles elementos firma ó código, de los que sólo se podrá incluir uno.

La etiqueta choice también posee los atributos minOccurs y maxOccurs.

**Etiqueta all**

Es similar a choice y sequence, pero tiene como diferencia principal que los elementos que contiene pueden aparecer o no y además en el orden que quieran.

Ejemplo:

<xs:element name="identificación">

<xs:complexType>

<xs:all>

<xs:element name="firma" type="xs:NCName"/>

<xs:element name="código" type="xs:NCName" />

</xs:all>

</xs:complexType>

</xs:element>

En este caso la firma y el código pueden aparecer o no, aparecer los dos e incluso el orden será indiferente. Es una etiqueta muy potente que ahorra mucho trabajo (etiqueta all)

Esta etiqueta también tiene los atributos minOccurs y maxOccurs, pero sólo se puede indicar como valores cero o uno.

o Mezcla de elementos

A veces los contenidos de un documento XML son extremadamente complejos, por lo que se permite mezclar etiquetas choice, con etiquetas sequence y etiquetas all. Estas posibilidades permiten crear cualquier tipo de esquema por complejo que resulte.

Ejemplo:

<xs:element name="correo">

<xs:complexType>

<xs:sequence>

<xs:element name="remite" type="xs:string"/>

<xs:element name="para" type="xs:string"

minOccurs="1" maxOccurs="unbounded"/>

<xs:choice>

<xs:element name="cc" type="xs:string"

minOccurs="1" maxOccurs="unbounded"/>

<xs:element name="cco" type="xs:string"

minOccurs="1" maxOccurs="unbounded"/>

</xs:choice>

</xs:sequence>

</xs:complexType>

</xs:element>

En el ejemplo, el elemento correo consta de tres elementos: remite, para y un tercero que puede ser cc o cco.

o Añadir atributos

Los atributos del elemento se definen al final del apartado complexType (justo antes de cerrarle). Ejemplo:

<xs:element name="persona">

<xs:complexType>

<xs:sequence>

<xs:element name="nombre" type="xs:string"/>

<xs:element name="apellidos" type="xs:string"/>

</xs:sequence>

<xs:attribute name="sexo">

<xs:simpleType>

<xs:restriction base="xs:string">

<xs:enumeration value="Hombre"/>

<xs:enumeration value="Mujer"/>

</xs:restriction>

</xs:simpleType>

</xs:attribute>

<xs:attribute name="fechaNacimiento" type="xs:date" use="required"/>

</xs:complexType>

</xs:element>

Las personas contienen dos elementos en secuencia (nombre y apellidos) y dos atributos: no opcional (sexo) que sólo pueden tomar los valores Hombre o Mujer y uno obligatorio para la fecha de nacimiento

- Tipo de datos complejos mixtos

<xs:element name="documento">

<xs:complexType mixed="true">

<xs:choice>

<xs:element name="negrita" type="xs:string"

minOccurs="0" maxOccurs="unbounded"/>

</xs:choice>

<xs:attribute name="lenguaje" type="xs:language"/>

</xs:complexType>

</xs:element>

Aqui tenemos elementos que contienen otros elementos y ademas texto

- Definicion de varios atributos:

La idea es la misma que con los grupos de elementos: **facilitar la escritura del código del esquema**. Se trata de definir (lo lógico es de forma global) una serie de atributos que utilizan diferentes elementos y así mejorar el mantenimiento del esquema XML.

**Los grupos de atributos se declaran con la etiqueta attributeGroup**. Al definir el grupo se utiliza el nombre y al usarle en un elemento se hace referencia a dicho nombre mediante el atributo ref.

Ejemplo:

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:attributeGroup name="infoDoc">

<xs:attribute name="lenguaje" type="xs:language" use="required"/>

<xs:attribute name="tamaño">

<xs:simpleType>

<xs:restriction base="xs:positiveInteger">

<xs:minExclusive value="20" />

<xs:maxExclusive value="200" />

</xs:restriction>

</xs:simpleType>

</xs:attribute>

<xs:attribute name="título" type="xs:string" use="required"/>

</xs:attributeGroup>

<xs:element name="documento">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attributeGroup ref="infoDoc"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

<xs:element name="documentoRef">

<xs:complexType>

<xs:attributeGroup ref="infoDoc"/>

<xs:attribute name="referencia" type="xs:positiveInteger"

use="required" />

</xs:complexType>

</xs:element>

<xs:element name="documentos">

<xs:complexType>

<xs:choice minOccurs="1" maxOccurs="unbounded">

<xs:element ref="documento" />

<xs:element ref="documentoRef" />

</xs:choice>

</xs:complexType>

</xs:element>

</xs:schema>

En el ejemplo se define un grupo de atributos globales llamado **infoDoc que está formado por tres atributos (lenguaje, tamaño y título)**. Después dos elementos (documento y documentoRef) utilizan el grupo de atributos como si fuera un solo atributo.

De modo que un XML basado en este esquema y válido podría ser:

<?xml version="1.0" encoding="UTF-8"?>

<documentos xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"

xs:noNamespaceSchemaLocation="correo.xsd" >

<documento lenguaje="es" tamaño="123" título="Informe 32"/>

<documento lenguaje="es" tamaño="34" título="Alta 34" />

<documentoRef lenguaje="es" referencia="3234" título="Informe 56"/>

</documentos>