XML Schema.

Introducción

Hasta ahora sabemos crear definiciones de tipo de documento para validar los documentos XML. Ahora veremos una nueva posibilidad que nos ofrece la familia de tecnologías XML para llevar a cabo dicha validación. Se trata de los esquemas XML o XML Schema. Estos tienen una gran ventaja sobre los DTD y es que se escriben en lenguaje XML y no necesitan de un lenguaje particular como ocurría con las DTD. Debemos dejar claro que con ambas técnicas se consiguen resultados similares, por lo que una vez explicados los esquemas XML, es decisión del programador emplear un tipo u otro.

Objetivos

En este punto, profundizaremos en los siguientes conocimientos:

1. Conocer los **esquemas XML** (*Schema XML*).
2. Construir esquemas que contienen las definiciones tipo para construir documentos **XML validos**.
3. Aprender a definir tipos simples y tipos complejos de datos para los esquemas de XML.
4. Entender los **esquemas W3C XML** como una alternativa a los **ficheros DTD** para la definición de sintaxis y estructura de los ficheros XML. Estos constituyen una herramienta más potente y flexible que los DTDs.

Una vez finalizada esta unidad didáctica, serás capaz de crear **Esquemas W3C XML** que puedan ayudarnos a crear documentos XML interesantes.

Sintaxis de los esquemas W3C XML

El concepto originario de esquema en XML fue el de un documento que describía una serie de restricciones y de convecciones en la estructura de una base de datos. Desde el punto de vista de XML, esa definición se ajusta en parte al propósito de dichos esquemas. Son documentos que describen el contenido que podemos usar en otros documentos. En ese sentido, su utilidad es más parecida a la de las DTDs, aunque restringen la estructura del documento de forma mucho más precisa. Los esquemas indican tipos de dato, número mínimo y máximo de ocurrencias y otras características más específicas. Tras varios intentos infructuosos a la hora de establecer un lenguaje estándar de desarrollo de esquemas, los esquemas W3C se han erigido como un estándar de creación de esquemas en XML.

Este lenguaje es muy extenso. La especificación del mismo está considerada más compleja que incluso la especificación 1.0 de XML. Para propósitos más modestos, existen otra serie de lenguajes más sencillos que pueden cumplir perfectamente su cometido.

Según la Especificación del W3C XML Schema (http://www.w3.org/XML/Schema), los esquemas expresan vocabularios compartidos que permiten a las máquinas extraer las reglas hechas por las personas. Los esquemas proveen un significado para definir la

estructura, contenido y semántica de los documentos XML. De algún modo, ofrecen nuevas posibilidades en el tratamiento de documentos. Podríamos señalar muchos puntos diferentes entre los esquemas y las DTD, pero nos centraremos en algunos puntos en los que hay ventajas al usar esquemas en lugar de DTDs.

- Usan la **sintaxis propia de XML**, al contrario de lo que sucede con las DTDs.
- Permiten **especificar de manera precisa los tipos de datos**, mientras que las DTDs no los especifican.
- Son **extensibles**, es decir, podemos crear nuevos elementos.
- Para procesar el documento, las herramientas y analizadores empleados para tratar los documentos XML deben ser capaces de procesar también las DTDs.
- Con los esquemas W3C **es posible expresar carga semántica** y esto es muy importante para el tratamiento de la información.

Por otro lado, algunas desventajas claras de las DTDs son las siguientes:

- **No permite el uso de namespaces**, circunstancia que hace que sea más difícil y estos son muy útiles ya que permiten definir elementos con igual nombre dentro del mismo contexto, siempre y cuando se anteponga un prefijo al nombre del elemento.
- Tiene una **tipología para los datos del documento extremadamente limitada**, pues no permite definir el que un elemento pueda ser de un tipo número, fecha, etc. sino que sólo presenta variaciones limitadas sobre cadenas.
- El **mecanismo de extensión es complejo y frágil ya que está basado en sustituciones sobre cadenas** y no hace explicitas las relaciones, es decir, que dos elementos que tienen definido el mismo modelo de contenido no presentan ninguna relación.

Al igual que sucede con las bases de datos, en los esquemas XML hay parte de la información contenida que no es explícita, sino que es inherente a la estructura que se ha creado con dicho esquema. El gran logro de XML es la habilidad para modelar los datos partiendo desde distintas fuentes de datos. El manejo de los esquemas XML implica de algún modo, aprender cómo se escriben y modelan los datos básicos.

Podemos considerar un esquema como un contenedor de componentes que incluyen elementos, atributos, etc.

Inicio XML Schema. Espacios de nombres

Los XML schemas son documentos XML. Así, todo esquema debe comenzar con una declaración XML: esta es la primera línea del documento.

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

En la segunda línea encontramos la declaración del elemento esquema:

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

Como todo documento XML, un schema debe tener un elemento raíz, este es el elemento schema

```
</xsd:schema>
```

Espacios de nombres dentro del XML schema.

Existe una relación estrecha entre espacios de nombres y XML schemas, en varios sentidos.
Al crear un XML schema hacemos uso de los elementos y atributos especificados en el estándar de XML Schema. Para que esto sea posible debemos incluir en el elemento raíz del esquema (el elemento “schema”) una referencia al espacio de nombres “http://www.w3.org/2001/XMLSchema”. Esto se hace incluyendo el atributo “xmlns” en el elemento “schema”:

“xmlns:” identifica el espacio de nombres al que pertenecen los componentes incluidos en el esquema, asignando opcionalmente un prefijo (este prefijo suele ser “xs” o “xsd”. Así, la sentencia

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

indica que los elementos y tipos de datos utilizados en el esquema provienen del espacio de nombres “http://www.w3.org/2001/XMLSchema”, al que se le asigna el prefijo “xs”. Además, podemos hacer que el esquema que estamos creando tenga asociado un espacio de nombres propio (target namespace). Para ello se puede utilizar el atributo **targetNamespace** del elemento “schema”; que crea un espacio de nombres al que pertenecen los elementos que se definen en el esquema. Por ejemplo:

```
targetNamespace:"http://www.prueba.es/esquema1"
```

También se puede especificar si los elementos y los atributos declarados en él deben estar certificados por un espacio de nombres, ya sea explícitamente mediante un prefijo o implícitamente de forma predeterminada, cuando se utilicen en un documento instancia XML. Para ello se pueden utilizar los atributos **elementFormDefault** y **attributeFormDefault** del elemento <xsd:schema>. Los posibles valores de estos atributos son:

- “**qualified**”: indica que, en los documentos instancia XML que referencien a este esquema, los elementos (en el caso de elementFormDefault)/atributos (en el caso de attributeFormDefault) del “target namespace” deben estar cualificados con un prefijo.
- “**unqualified**”: Este es el valor por defecto. Indica que los elementos/atributos no necesitan estar prefijados en el documento instancia.

En los documentos instancia XML

En los documentos instancia XML, la referencia a un esquema XML se hace mediante atributos en el elemento raíz del documento instancia XML. Estos atributos, especificados en el estándar, se encuentran definidos en el espacio de nombres “http://www.w3.org/2001/XMLSchema-instance”. Por lo tanto, para poder usarlos hay que hacer referencia a dicho espacio de nombres en el documento instancia XML, mediante el atributo “xmlns”. Normalmente a este espacio de nombres se le asigna el prefijo “xsi” (aunque se puede usar cualquier prefijo), con lo que la referencia al espacio de nombres quedaría así:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

Con esto podemos utilizar los atributos “**noNamespaceSchemaLocation**” y “**schemaLocation**”, que nos permiten asociar un documento instancia XML con un esquema:

- **noNamespaceSchemaLocation**: identifica un documento de schema que no tiene un espacio de nombres de destino (no incluye el atributo “targetNamespace”) y lo asocia al documento XML instancia.
- **schemaLocation**: Asocia un documento de esquema que tiene un espacio de nombres de destino (targetNamespace) con un documento de XML instancia. Este atributo tendrá dos valores, separados por un espacio en blanco. El primer valor coincide con el del “targetNamespace” especificado en el schema. El segundo es la ubicación donde se encuentra definido el XML schema.

Por ejemplo, supongamos que tenemos el archivo “apuntes.xsd” que contiene un XML schema. Si en dicho schema se definió

```
targetNamespace:”http://www.prueba.es/esquema1”
```

En un documento XML instancia que queremos asociar a este esquema tendremos:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation:”http://www.prueba.es/esquema1 apuntes.xsd”
```

Si por el contrario en el documento “apuntes.xsd” no se especifica Un “targetNamespace”, en el documento instancia XML tendremos:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=”apuntes.xsd”
```

Para establecer cuál es el prefijo de un determinado espacio de nombres dentro del documento instancia (XML) utilizaciones:

```
xmlns: prefijo= "espacio de nombres"
xmlns:apu="http://www.prueba.es/esquema1"
```

Ejemplo sencillo

Ejemplo de un documento XML al que asociamos un espacio de nombres.

```
<p:persona
     xmlns:p="http://www.prueba.es/persona"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation= "http://www.prueba.es/persona people.xsd">
    <p:nombre>John</p:nombre>
    <p:direccion>John</p:direccion>
</p:persona>
```

El documento XSD que contiene el espacio de nombres es “people.xsd”

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.prueba.es/persona"
elementFormDefault="qualified"
attributeFormDefault="qualified">
  <xs:element name="persona">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nombre" type="xs:string" minOccurs="0"
             maxOccurs="unbounded"/>
        <xs:element name="direccion" type="xs:string" minOccurs="0"
             maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

En resumen

El elemento “schema” incluye el atributo xmlns (XML NameSpace), que especifica el espacio de nombres para el esquema. La sintaxis genérica del atributo xmlns es la siguiente:

```
xmlns:prefijo ="URI_DE_UN_ESPACIO_DE_NOMBRE"
```

En el ejemplo, el formato xmlns:xsd indica que todos los elementos o atributos que lleven el prefijo “xsd:” pertenecen al espacio de nombres especificado en la URI (http://www.w3.org/2001/XMLSchema). Los prefijos se utilizan para distinguir entre diferentes espacios de nombres. Se puede utilizar cualquier prefijo, siempre que se especifique el espacio de nombres XML asociado. Si el esquema referencia un único espacio de nombres, por ejemplo, si sólo utiliza elementos y atributos definidos en la especificación XML Schema, no es obligatorio usar el prefijo. La declaración del elemento esquema en este caso podría quedar así:

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

En este caso no sería necesario usar el prefijo “xsd:” delante de todos los elementos y atributos del esquema. Los espacios de nombres se estudiarán en detalle más adelante. Dentro del elemento “schema” se encuentran todas las declaraciones de elementos y atributos que pueden incluirse en los ejemplares XML que utilicen esta definición de esquema, que se denominan documentos instancia.
Así, en el ejemplo siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Libro xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="libro.xsd" precio="20">
    <Título>Fundamentos de XML Schema</Título>
    <Autores>Allen Wyke</Autores>
    <Autores>Andrew Watt</Autores>
    <Editorial>Wiley</Editorial>
</Libro>
```

El elemento raíz en las instancias XML para este schema se llama “Libro” y tiene tres elementos hijo y un atributo.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Libro">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Título" type="xsd:string"/>
        <xsd:element name="Autores" type="xsd:string" maxOccurs="10"/>
        <xsd:element name="Editorial" type="xsd:string"/>
      </xsd:sequence>
      <xsd:attribute name="precio" type="xsd:double"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Los hijos son “Título”, “Editorial”, que deben aparecer ambos una vez, y “Autores” que puede aparecer de una a diez veces. El hecho de que estén agrupados en una secuencia (<xsd:sequence>)indica que los elementos deben aparecer en ese orden, es decir, primero el “Título”, luego los “Autores” y por último la “Editorial”. Los tres elementos son de tipo string (cadena de caracteres). El atributo de libro se llama “precio” y es de tipo “double”.

Vemos que la referencia al XML schema desde el ejemplar XML se hace desde dentro de la etiqueta de inicio del elemento raíz, donde se especifican dos cosas:

- `“xmlns:xsi = http://www.w3.org/2001/XMLSchema-instance”`: indica que queremos utilizar los elementos definidos en http://www.w3.org/2001/XMLSchema-instance.
- `“xsi:noNamespaceSchemaLocation="libro.xsd"`: indica que vamos a usar ese fichero (libro.xsd) que contiene el XSchema, pero sin asociar un espacio de nombres a esas definiciones. Sin esta sentencia, no tendremos esquema de validación.