1 Ferramentas, validacións de Schemas e exemplo completo

1.1 Sumario

1.2 Ferramentas

Para poder facer schemas usaremos as mesmas ferramentas que usabamos en XML.

Ver ferramentas empregadas en XML : Utilización de ferramentas

1.3 Validación de schemas

Para validar un schema poderemos facelo dende:

http://www.w3.org/2001/03/webdata/xsv

1.4 Exemplo completo de creación dun arquivo .XSD partindo dun arquivo .XML

Nesta sección imos a crear un arquivo .xsd partindo do arquivo .xml empregando 3 formas diferentes de facelo.

Arquivo .xml orixinal "shiporder.xml":

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

<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

O documento XML consiste nun elemento raiz ?shiporder?, que contén un atributo requerido chamado ?orderid?. O elemento ?shiporder? contén tres tipos diferentes de fillos: ?orderperson?, ?shipto? e ?item?. O elemento ?item? aparece dúas veces, e contén un ?título? e un elemento opcional ?note?, unha ?cantidade? e un elemento ?prezo?.

A liña superior xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance? dille ó procesador XML que este documento deberá estar validado contra un schema. A liña ?xsi:noNamespaceSchemaLocation="shiporder.xsd"? especifica onde reside o schema (está no mesmo cartafol que ?shiporder.xml?).

1.4.1 Método 1: Forma sinxela de creación dun Schema XSD

Agora queremos crear un Schema XSD para poder validar o documento anterior XML.

Comenzaremos facendo un novo arquivo chamado "shiporder.xsd". Para crear o schema simplemente seguiremos a estructura no documento XML e definiremos cada un dos elementos que imos atopando. . Comezaremos coa declaración standard seguida do elemento **xs:schema**:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
...
</xs:schema>
```

No schema superior estamos empregando o namespace standard (xs), e a URI asociada con este namespace é a definición da linguaxe do Schema, que ten o valor standard http://www.w3.org/2001/XMLSchema.

A continuación, temos que definir o elemento "shiporder". Este elemento ten un atributo e contén outros elementos, por tanto considerámolo como tipo complexo. Os elementos fillos de "shiporder" están rodeados por un elemento xs:sequence que define a orde da secuencia dos sub elementos:

```
<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

A continuación temos que definir o elemento ?orderperson? como un tipo simple (xa que non contén ningún atributo ou outros elementos). O tipo (xs:string) está prefixado no espazo de nomes asociado có Schema XML que indica un schema predefinido de tipo de datos.

```
<xs:element name="orderperson" type="xs:string"/>
```

A continuación temos que definir dous elementos que son de tipo complexo: ?shipto? e ?item?. Comenzamos definindo o elemento ?shipto?:

```
<xs:element name="shipto">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="address" type="xs:string"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Cós schemas podemos definir o número posible de ocorrencias para un elemento empregando os atributos maxOccurs e minOccurs. maxOccurs especifica o número máximo de ocorrencias para un elemento e minOccurs especifica o número mínimo de ocorrencias para un elemento. O valor por defecto para maxOccurs e minOccurs é 1.

Agora podemos definir o elemento ?item?. Este elemento pode aparecer moitas veces dentro de ?shiporder?. Isto especificase axustando o atributo maxOccurs para o elemento ?item? ó valor ?unbounded?, que significa que pode aparecer tantas veces o elemento ?item? como desexe o autor. O elemento ?note? é opcional. Isto indícase axustando o atributo minOccurs a cero:

```
<xs:element name="item" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="note" type="xs:string" minOccurs="0"/>
      <xs:element name="quantity" type="xs:positiveInteger"/>
      <xs:element name="price" type="xs:decimal"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Agora podemos declarar o atributo do elemento ?shiporder?. Xa que é un atributo obrigatorio deberemos empregar use=?required?.

Nota: As declaracións de atributos sempre irán ó final:

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

Aquí temos o listado completo do schema chamado "shiporder.xsd":

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="orderperson" type="xs:string"/>
      <xs:element name="shipto">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="address" type="xs:string"/>
            <xs:element name="city" type="xs:string"/>
            <xs:element name="country" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="item" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="title" type="xs:string"/>
            <xs:element name="note" type="xs:string" minOccurs="0"/>
            <xs:element name="quantity" type="xs:positiveInteger"/>
            <xs:element name="price" type="xs:decimal"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="orderid" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```

1.4.2 Método 2: División do Schema na creación do XSD

O método anterior de deseño é moi simple, pero pode ser dificil de ler e manter cando son documentos complexos.

O seguinte método de deseño está baseado na definición de todos os elementos e atributos primeiro, e a continuación referencialos usando o atributo **ref**.

Aquí temos o novo arquivo de deseño ("shiporder.xsd"):

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">


<xs:element name="orderperson" type="xs:string"/>
<xs:element name="name" type="xs:string"/>
<xs:element name="address" type="xs:string"/>
<xs:element name="city" type="xs:string"/>
<xs:element name="country" type="xs:string"/>
<xs:element name="title" type="xs:string"/>
<xs:element name="note" type="xs:string"/>
<xs:element name="quantity" type="xs:positiveInteger"/>
<xs:element name="price" type="xs:decimal"/>


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


<xs:element name="shipto">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="name"/>
      <xs:element ref="address"/>
      <xs:element ref="city"/>
```

```
      <xs:element ref="country"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="item">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="note" minOccurs="0"/>
      <xs:element ref="quantity"/>
      <xs:element ref="price"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="orderperson"/>
      <xs:element ref="shipto"/>
      <xs:element ref="item" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="orderid" use="required"/>
  </xs:complexType>
</xs:element>

</xs:schema>
```

1.4.3 Método 3: Empregando nomes para os tipos de datos do Schema XSD

O terceiro método de deseño define clases ou tipos, que permiten a reutilización das definicións de elementos. Isto faise nomeando os elementos de tipos simples e complexos e a continuación referenciándoos a través do atributo type do elemento.

Aquí temos a terceira versión do arquivo de schema ("shiporder.xsd"):

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:simpleType name="stringtype">
  <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="inttype">
  <xs:restriction base="xs:positiveInteger"/>
</xs:simpleType>

<xs:simpleType name="dectype">
  <xs:restriction base="xs:decimal"/>
</xs:simpleType>

<xs:simpleType name="orderidtype">
  <xs:restriction base="xs:string">
    <xs:pattern value="[0-9]{6}"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="shiptotype">
  <xs:sequence>
    <xs:element name="name" type="stringtype"/>
    <xs:element name="address" type="stringtype"/>
    <xs:element name="city" type="stringtype"/>
    <xs:element name="country" type="stringtype"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="itemtype">
  <xs:sequence>
    <xs:element name="title" type="stringtype"/>
```

```
    <xs:element name="note" type="stringtype" minOccurs="0"/>
    <xs:element name="quantity" type="inttype"/>
    <xs:element name="price" type="dectype"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="shipordertype">
  <xs:sequence>
    <xs:element name="orderperson" type="stringtype"/>
    <xs:element name="shipto" type="shiptotype"/>
    <xs:element name="item" maxOccurs="unbounded" type="itemtype"/>
  </xs:sequence>
  <xs:attribute name="orderid" type="orderidtype" use="required"/>
</xs:complexType>

<xs:element name="shiporder" type="shipordertype"/>

</xs:schema>
```

A restrición elemento indica que o tipo de datos ven derivado do namespace do W3C XML Schema. Polo tanto, o seguinte fragmento quere dicir que o valor do elemento ou atributo debe ser unha cadea:

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

O elemento restriction é usado moitas veces para aplicar restricións ós elementos. Mira as seguintes liñas do schema anterior:

```
<xs:simpleType name="orderidtype">
  <xs:restriction base="xs:string">
    <xs:pattern value="[0-9]{6}"/>
  </xs:restriction>
</xs:simpleType>
```

Isto indica que o valor do elemento ou atributo debe ser unha cadea, debe ter exactamente 6 caracteres nunha fila e eses caracteres deben ser un número entre 0 e 9.

Codigo fonte do exemplo: www.w3schools.com