XML Schema 元素替换 XML Schema 字符串数据类型

XSD 实例

本节会为您演示如何编写一个 XML Schema。您还将学习到编写 schema 的不同方法。

XML 文档

让我们看看这个名为 "shiporder.xml" 的 XML 文档：

```
<?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>
```

上面的XML文档包括根元素 "shiporder"，其中包含必须名为 "orderid" 的属性。"shiporder" 元素包含三个不同的子元素："orderperson"、"shipto" 以及 "item"。"item" 元素出现了两次，它含有一个 "title"、一个可选 "note" 元素、一个 "quantity" 以及一个 "price" 元素。

上面这一行 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"，告知XML解析器根据某个 schema 来验证此文档。这一行：xsi:noNamespaceSchemaLocation="shiporder.xsd" 规定了 schema 的位置（在这里，它与 "shiporder.xml" 处于相同的文件夹）。

创建一个 XML Schema

现在，我们需要为上面这个 XML 文档创建一个 schema。

我们可以通过打开一个新的文件来开始，并把这个文件命名为 "shiporder.xsd"。要创建schema，我们仅仅需要简单地遵循 XML 文档中的结构，定义我们所发现的每个元素。首先我们开始定义一个标准的 XML 声明：

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

在上面的 schema 中，我们使用了标准的命名空间 (xs)，与此命名空间相关联的 URI 是 Schema 的语言定义（Schema language definition），其标准值是 http://www.w3.org/2001/XMLSchema。

接下来，我们需要定义 "shiporder" 元素。此元素拥有一个属性，其中包含其他的元素，因此我们将它认定为复合类型。"shiporder" 元素的子元素被 xs:sequence 元素包围，定义了子元素的次序：

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

然后我们需要把 "orderperson" 元素定义为简易类型（这是因为它不包含任何属性或者其他的元素）。类型 (xs:string) 的前缀是由命名空间的前缀规定的，此命名空间与指示预定义的 schema 数据类型的 XML schema 相关联：

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

接下来，我需要把两个元素定义为复合类型："shipto" 和 "item"。我们从定义 "shipto" 元素开始：

```
<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>
```

通过 schema，我们可使用 maxOccurs 和 minOccurs 属性来定义某个元素可能出现的次数。maxOccurs 定义某元素出现次数的最大值，而 minOccurs 则定义某元素出现次数的最小值。maxOccurs 和 minOccurs 的默认值都是 1！

现在，我们可以定义 "item" 元素了。这个元素可在 "shiporder" 元素内部出现多次。这是通过把 "item" 元素的 maxOccurs 属性的值设定为 "unbounded" 来实现的，这样 "item" 元素就可出现创作者所希望的任意多次。请注意，"note" 元素是可选元素。我们已经把此元素的 minOccurs 属性设定为 0 了：

```
<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>
```

现在，我们可以声明 "shiporder" 元素的属性了。由于这是一个必选属性，我们规定 use="required"。

注意：此属性的声明必须被置于最后：

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

这是这个名为 "shiporder.xsd" 的 schema 文件的文档清单：

```
<?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>
```

分割 Schema

前面的设计方法非常容易，但当文档很复杂时却难以阅读和维护。

接下来介绍的设计方法基于首先对所有元素和属性的定义，然后再使用 ref 属性来引用它们。

这是用新方法设计的 schema 文件("shiporder.xsd")：

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

<!-- definition of simple elements -->
<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"/>

<!-- definition of attributes -->
<xs:attribute name="orderid" type="xs:string"/>

<!-- definition of complex elements -->
<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>
```

使用指定的类型（Named Types）

第三种设计方法定义了类或者类型，这样使我们有能力重复使用元素的定义。具体的方式是：首先对简易元素和复合元素进行命名，然后通过元素的 type 属性来指向它们

这是利用第三种方法设计的 schema 文件 ("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>
```

restriction 元素显示出数据类型源自于 W3C XML Schema 命名空间的数据类型。因此，下面的片段也就意味着元素或属性的值必须是字符串类型的值：

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

restriction 元素常被用于向元素施加限制。请看下面这些来自以上 schema 的片段：

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

这段代码指示出，元素或属性的值必须为字符串，并且必须是连续的六个字符，同时这些字符必须是 0-9 的数字。

← XML Schema 元素替换 XML Schema 字符串数据类型 →