XML Schema 学习总结

简单类型

1、简单元素：指只能包含文本内容，不能够包含子元素，也没有属性的元素。
格式：<xs:element name="xxx" type="yyy"/>
例子：

```
<xs:element name="name" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="gender" type="xs:boolean"/>
```

2、属性：所有的元素属性均被声明为简单类型。只有复杂类型的元素才可以拥有属性。
格式：<xs:attribute name="xxx" type="yyy"/>
例子：<xs:attribute name="lang" type="xs:string"/>

所有的属性默认都是可选的，我们可以通过使用 use 关键字明确的指出是可选或是必需：

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

我们可以通过使用 default 或 fixed 为简单类型（简单元素、属性）指定默认值或固定值，
如下：

```
<xs:element name="color" type="xs:string" default="red"/>
<xs:attribute name="lang" type="xs:string" fixed="CN"/>
```

对简单类型值的约束

约束	含义
enumeration	定义允许值的枚举
fractionDigits	指定最多允许的小数位数（必须大于或等于零）
length	精确指定允许的最大字符长度
maxExclusive	指定允许的最大数值，必须小于该值
maxInclusive	指定允许的最大数值，必须小于或等于该值
maxLength	指定允许的最大字符长度（必须大于等于零）
minExclusive	指定允许的最小数值，必须大于该值
minInclusive	指定允许的最小数值，必须大于或

	等于该值
minLength	指定允许的最小字符长度
pattern	指定允许值的模式，类似正则表达式
totalDigits	精确指定数字个数
whiteSpace	处理空白（保留：preserve；替换：replace；合并：collapse）

复杂类型

复杂类型指包含其他元素/属性的元素类型。

```
<message>
<to>rose</to>
<from>alex</from>
<body>Hi,My Girl!</body>
</message>
```

在上面的例子中，元素 message 就是一个复杂类型的元素，我们在 Schema 中这样描述：

```
<xs:element name="message">
<xs:complexType>
<xs:sequence>
<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
</xs:sequence>
</xs:complexType>
```

注意元素 to,from,body 包含在<xs:sequence></xs:sequence>中，表明这些元素必须按照定义的顺序出现在你的 XML 文件中。

当然，message 元素也可以包含一个 type 属性，指向我们定义的复杂类型，象这样：

```
<xs:element name="message" type="msg"/>
<xs:complexType name="msg">
<xs:sequence>
<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
</xs:sequence>
</xs:complexType>
```

复杂类型和简单类型之间最根本的区别就是：复杂类型的内容中可以包含其他元素，也可以带有属性（Attribute）,但简单类型既不能包含子元素，也不能带有任何属性。

Schema 综述

1、如何描述空元素，比如：<product prodid="1345" />？
因为是空元素，所以不包含子元素，同时由于包含属性，用 attribute 定义，象这样：

```
<xs:element name="product">
<xs:complexType>
<xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
</xs:element>
```

也可以这样：

```
<xs:element name="product" type="productType"/>
<xs:complexType name="productType">
<xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
```

2、如何描述只含有简单内容（文本/属性）的元素，比如：
<message date="2006-06-26">Hi,My Girl!</message>？
由于只包含简单内容，所以我们在元素内容定义的外面用 simpleContent 指出，当描述简单内容的时候，我们需要在简单内容里使用 extension 或者 restriction 来描述内容的数据类型。象这样：

```
<xs:element name="message">
<xs:complexType>
<xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute name="date" type="xs:date" />
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
```

其中 message 的属性 date 的数据类型为日期（xs:date）。

顺便提一下：XML Schema 中常用的数据类型有：
xs:string 、xs:decimal 、xs:integer 、xs:boolean 、xs:date 、xs:time 等。

3、如何定义混合类型，比如：

```
<message>
 This message comes from <from>Alex</from>
</message>
```

message 元素除了包含子元素 from 之外，还直接包含文本“This message comes from”。对于这种情况，我们需要在 complexType 中使用属性 mixed="true"指出。以下是 Schema:

```
<xs:element name="message">
<xs:complexType mixed="true">
<xs:element name="from" type="xs:string"/>
</xs:complexType>
</xs:element>
```

当然，如果包含更多的子元素，我们可以需要使用<xs:sequence>来限定那些子元素的顺序。

在 XML Schema 中，有 3 类共 7 种指示器（Indicator）：

一、定义元素如何出现：包括 all,sequence,choice 这三个。

1、all：默认值。不限制子元素的出现顺序，每个子元素必须出现且只能出现一次。例如：

```
<xs:element name="person">
<xs:complexType>
<xs:all>
<xs:element name="firstname" type="xs:string"/>
<xs:element name="lastname" type="xs:string"/>
</xs:all>
</xs:complexType>
</xs:element>
```

2、sequence：子元素在 XML 文件中按照 XML Schema 定义的顺序出现。

3、choice：两个或多个子元素中仅出现一个。例如：

```
<xs:element name="gender">
<xs:complexType>
<xs:choice>
<xs:element name="male" type="male"/>
<xs:element name="female" type="female"/>
</xs:choice>
</xs:complexType>
</xs:element>
```

二、次数限定类，包括 minOccurs 和 maxOccurs，前者指定最少出现次数，

后者指定最多出现次数。例如：

```
<xs:element name="person">
<xs:complexType>
<xs:sequence>
<xs:element name="full_name" type="xs:string"/>
<xs:element name="child_name" type="xs:string"
maxOccurs="10" minOccurs="0"/>
</xs:sequence>
</xs:complexType>
</xs:element>
```

如果元素出现的最大次数无限制，可以使用 maxOccurs="unbounded"。

三、组限定：包括 Group 和 attributeGroup，用来定义一组相关的元素。比如：

```
<xs:group name="persongroup">
<xs:sequence>
<xs:element name="firstname" type="xs:string"/>
<xs:element name="lastname" type="xs:string"/>
<xs:element name="birthday" type="xs:date"/>
</xs:sequence>
</xs:group>
```

```
<xs:attributeGroup name="personattrgroup">
<xs:attribute name="firstname" type="xs:string"/>
<xs:attribute name="lastname" type="xs:string"/>
<xs:attribute name="birthday" type="xs:date"/>
</xs:attributeGroup>
```

补充：<any>和<anyAttribute> ，在 XML Schema 中使用这两个元素可以放宽 Schema 对 XML 文件内容的限制。容许我们在 XML 文件中使用没有在 Schema 中定义的元素和属性。（很少使用）

元素属性 substitutionGroup 可以让元素 b 替换元素 a 在 XML 文件中出现。比如：

```
<xs:element name="cn_name" type="xs:string"/>
<xs:element name="en_name" substitutionGroup="cn_name"/>
```

这种情形类似 choice:

```
<xs:choice>
<xs:element name="cn_name" type="xs:string"/>
<xs:element name="en_name" type="xs:string"/>
</xs:choice>
```