



北京圣思园科技有限公司  
<http://www.shengsiyuan.com>

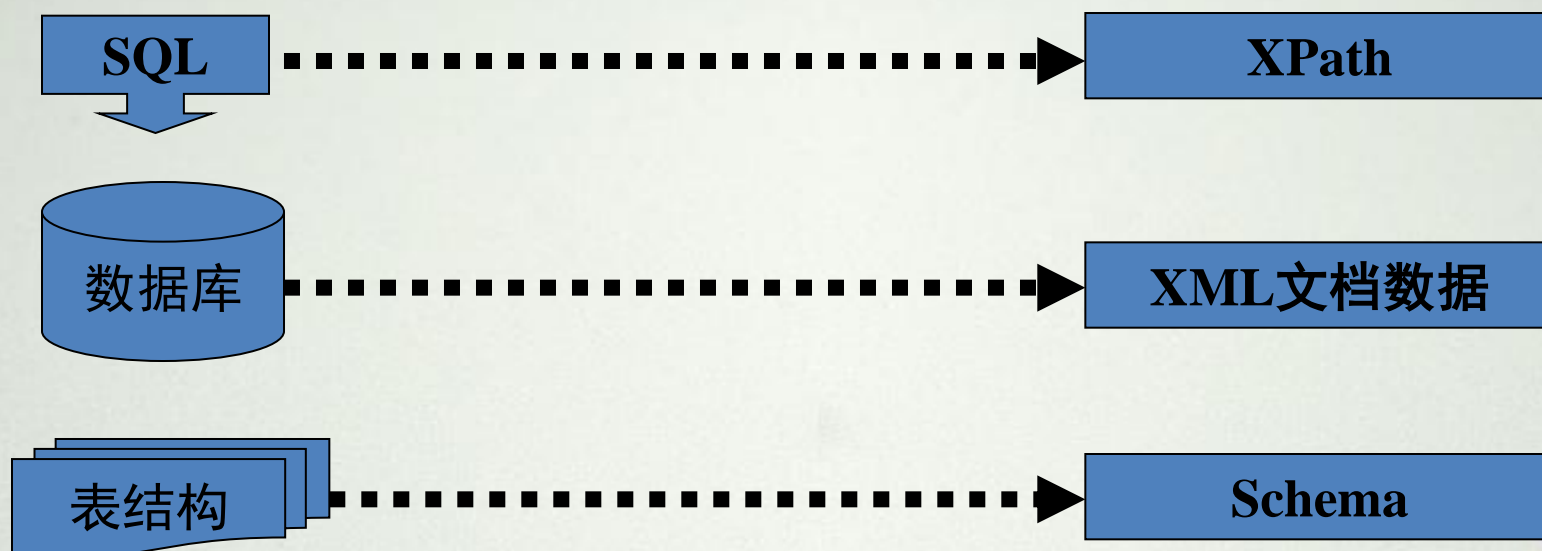
主讲人：张龙

# Schema (**Important!**)

- 理解**Schema**的数据类型
- 理解**Schema**的元素类型
- 理解验证与约束



# 什么是Schema



**XML Schema**是用一套预先规定的**XML**元素和属性创建的，这些元素和属性定义了**XML**文档的结构和内容模式。

**XML Schema**规定**XML**文档实例的结构和每个元素/属性的数据类型。

# 什么是Schema（续）

```
<书本>  
  <名称>书剑恩仇录</名称>  
  <作者>金庸</作者>  
</书本>
```

XML

```
<!ELEMENT 书本 (名称,作者)>  
<!ELEMENT 名称 (#PCDATA)>  
<!ELEMENT 作者 (#PCDATA)>
```

DTD

```
<element name="书本" type="书本类型" />  
<complexType name="书本类型" >  
  <element name="名称" type="string"/>  
  <element name="作者" type="string"/>  
</complexType>
```

Schema





# 为何要Schema

- **DTD 的局限性**

- **DTD**不遵守**XML**语法（写**XML**文档实例时候用一种语法，写**DTD**的时候用另外一种语法）
- **DTD**数据类型有限（与数据库数据类型不一致）
- **DTD**不可扩展
- **DTD**不支持命名空间（命名冲突）

- **Schema的新特性**

- **Schema**基于**XML**语法
- **Schema**可以用能处理**XML**文档的工具处理
- **Schema**大大扩充了数据类型，可以自定义数据类型
- **Schema**支持元素的继承—**Object-Oriented'**
- **Schema**支持属性组



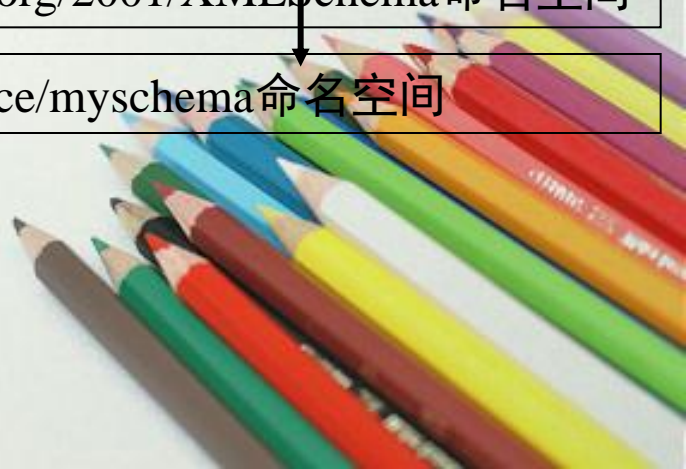
# Schema的文档结构

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://mynamespace/myschema"
>
    <!--放入实际内容 -->
</xs:schema>
```

所有Schema文档使用schema作为其根元素

用于构造schema的元素和数据类型来自http://www.w3.org/2001/XMLSchema命名空间

本schema定义的元素和数据类型属于http://mynamespace/myschema命名空间



# Schema的数据类型

- 简单类型
  - 内置的数据类型（**built-in data types**）
    - 基本的数据类型
    - 扩展的数据类型
  - 用户自定义数据类型（通过**simpleType**定义）
- 复杂类型（通过**complexType**定义）



# Schema的数据类型—基本数据类型

数据类型	描述
<b>string</b>	表示字符串
<b>boolean</b>	布尔型
<b>decimal</b>	代表特定精度的数字
<b>float</b>	表示单精度32位浮点数
<b>double</b>	表示双精度64位浮点数
<b>duration</b>	表示持续时间
<b>dateTime</b>	代表特定的时间
<b>time</b>	代表特定的时间，但是是每天重复的
<b>date</b>	代表日期
<b>hexBinary</b>	代表十六进制数
<b>anyURI</b>	代表一个URI，用来定位文件
<b>NOTATION</b>	代表 NOTATION类型





# Schema的数据类型—扩展的数据类型

数据类型	描述
<b>ID</b>	用于唯一标识元素
<b>IDREF</b>	参考ID类型的元素或属性
<b>ENTITY</b>	实体类型
<b>NMTOKEN</b>	NMTOKEN类型
<b>NMTOKENS</b>	NMTOKEN类型集
<b>long</b>	表示整型数，大小介于-9223372036854775808和9223372036854775807之间
<b>int</b>	表示整型数，大小介于-2147483648和 2147483647之间
<b>short</b>	表示整型数，大小介于-32768和32767之间
<b>byte</b>	表示整型数，大小介于-128和127之间

# Schema的数据类型—数据类型特性

特性	描述
<b>enumeration</b>	在指定的数据集中选择，限定用户的选值
<b>fractionDigits</b>	限定最大的小数位，用于控制精度
<b>length</b>	指定数据的长度
<b>maxExclusive</b>	指定数据的最大值（小于）
<b>maxInclusive</b>	指定数据的最大值（小于等于）
<b>maxLength</b>	指定长度的最大值
<b>minExclusive</b>	指定最小值（大于）
<b>minInclusive</b>	指定最小值（大于等于）
<b>minLength</b>	指定最小长度
<b>Pattern</b>	指定数据的显示规范



# Schema的元素类型

- **schema**
  - **element**
  - **attribute**
  - **group**
  - **attributeGroup**
  - **simpleType**
  - **simpleContent**
  - **complexType**
- choice**
  - list**
  - union**
  - unique**
  - sequence**
  - restriction**




# schema元素

- 作用：包含已经定义的**schema**
- 用法：<**xs:schema**>
- 属性：
  - **xmlns**
  - **targetNamespace**





# element元素

- 作用：声明一个元素
- 属性：
  - **name**
  - **type**
  - **ref**
  - **minOccurs**
  - **maxOccurs**
  - **substitutionGroup**
  - **fixed**
  - **default**
- 示例： 

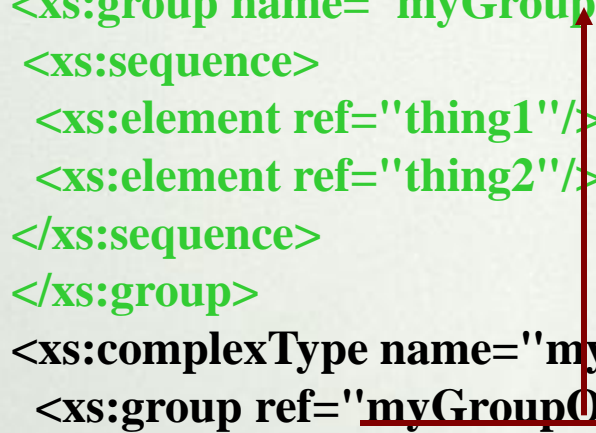
```
<xs:element name="cat" type="xs:string"/>
<xs:element name="dog" type="xs:string"/>
<xs:element name="pets">
  <xs:complexType>
    <xs:sequence minOccurs="0"
maxOccurs="unbounded">
      <xs:element ref="cat"/>
      <xs:element ref="dog"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```



# group元素

- 作用：把一组元素声明组合在一起，以便它们能够一起被复合类型应用
- 属性：**name/ref**
- 示例：

```
<xs:element name="thing1" type="xs:string"/>
<xs:element name="thing2" type="xs:string"/>
<xs:attribute name="myAttribute" type="xs:decimal"/>
<xs:group name="myGroupOfThings">
  <xs:sequence>
    <xs:element ref="thing1"/>
    <xs:element ref="thing2"/>
  </xs:sequence>
</xs:group>
<xs:complexType name="myComplexType">
  <xs:group ref="myGroupOfThings"/>
  <xs:attribute ref="myAttribute"/>
</xs:complexType>
```



# attribute元素

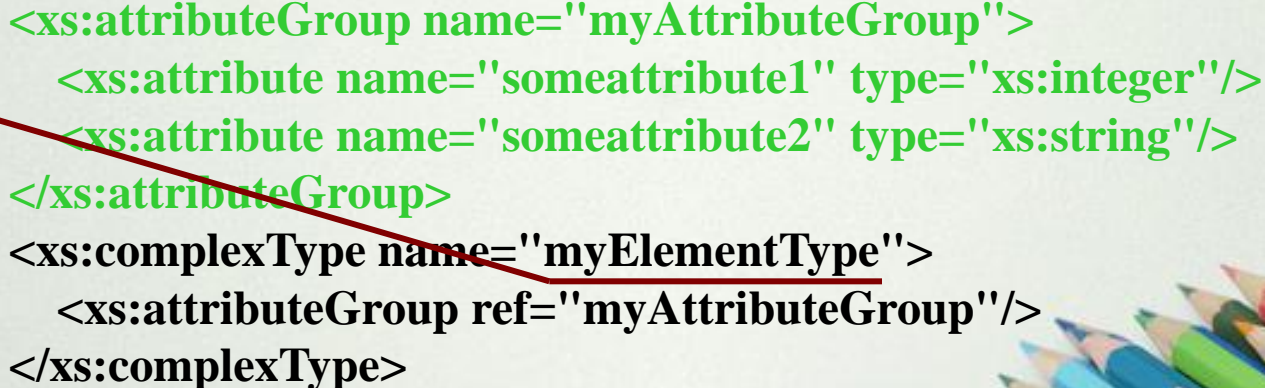
- 作用：声明一个属性
- 属性：**name/type/ref/use**
- 示例：

```
<xs:complexType name="myComplexType">  
  <xs:attribute name="mybaseattribute" type="xs:string" use="required"/>  
</xs:complexType>
```



# attributeGroup元素

- 作用：把一组属性声明组合在一起，以便可以被复合类型应用
- 属性：**name/ref**
- 示例：



```
<xs:attributeGroup name="myAttributeGroup">
  <xs:attribute name="someattribute1" type="xs:integer"/>
  <xs:attribute name="someattribute2" type="xs:string"/>
</xs:attributeGroup>
<xs:complexType name="myElementType">
  <xs:attributeGroup ref="myAttributeGroup"/>
</xs:complexType>
```

The diagram illustrates the use of the `attributeGroup` element. A red bracket on the left groups the first four lines of XML code, which define the `myAttributeGroup`. A red arrow points from this bracket to the `ref="myAttributeGroup"` attribute in the `myElementType` definition, showing how the group is referenced.



# simpleType元素

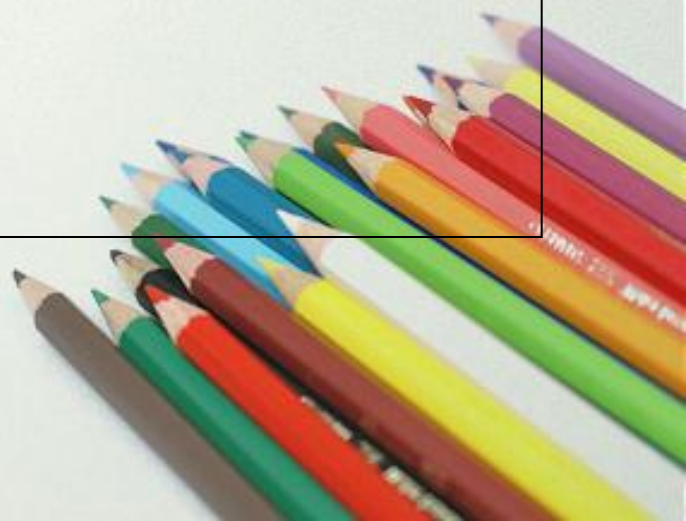
- 作用：定义一个简单类型，它决定了元素和属性值的约束和相关信息
- 属性：**name**
- 内容：应用已经存在的简单类型，三种方式：
  - **restrict**→限定一个范围
  - **list**→从列表中选择
  - **union**→包含一个值的结合



# simpleType元素（续）

1.子元素为：<xs:restriction> → 定义一个约束条件

```
<xs:simpleType  
  name="freezeboilrangeInteger">  
  <xs:restriction base="xs:integer">  
    <xs:minInclusive value="0"/>  
    <xs:maxInclusive value="100"/>  
  </xs:restriction>  
</xs:simpleType>
```



# simpleType元素（续）

## 2.子元素为：<xs:list>

从一个特定数据类型的集合中选择定义一个简单类型元素

```
<xs:simpleType name="listOfDates">  
  { <xs:list itemType="xs:date"/>  
</xs:simpleType>
```



# simpleType元素（续）

## 3.子元素为：<xs:union>

从一个特定简单数据类型的集合中选择定义一个简单类型元素

```
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  <xs:attribute name="allframesize">
    <xs:simpleType>
      <xs:union>
        <xs:simpleType>
          <xs:restriction base="roadbikesize"/>
        </xs:simpleType>
        <xs:simpleType>
          <xs:restriction base="mountainbikesize"/>
        </xs:simpleType>
      </xs:union>
    </xs:simpleType>
  </xs:attribute>
```





# simpleType元素 (续)

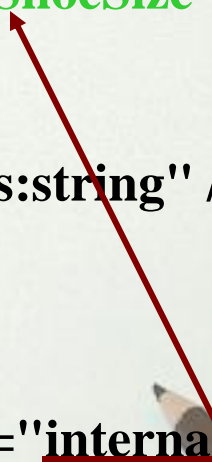

```
<xs:simpleType name="roadbikesize">
  <xs:restriction base="xs:positiveInteger">
    <xs:enumeration value="46"/>
    <xs:enumeration value="52"/>
    <xs:enumeration value="55"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="mountainbikesize">
  <xs:restriction base="xs:string">
    <xs:enumeration value="small"/>
    <xs:enumeration value="medium"/>
    <xs:enumeration value="large"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```



# complexType元素

- 作用：定义一个复合类型，它决定了一组元素和属性值的约束和相关信息
- 属性：**name**
- 示例：

```
<xs:complexType name="internationalShoeSize">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
      <xs:attribute name="sizing" type="xs:string" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:element name="myShoeSize" type="internationalShoeSize"/>
```



## complexType与simpleType区别（重要）

- **simpleType**类型的元素中不能包含元素或者属性。
- 当需要声明一个元素的子元素和/或属性时，用**complexType**；
- 当需要基于内置的基本数据类型定义一个新的数据类型时，用**simpleType**。



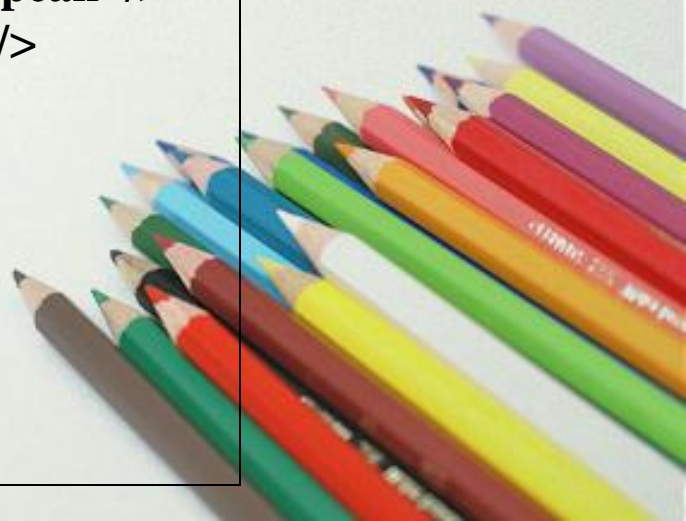
# simpleContent元素

- 作用：应用于complexType，对它的内容进行约束和扩展

- 示例：

```
<xs:element name="shoeSize">
  <xs:complexType>
    <xs:simpleContent>(元素下不包括子元素)
      <xs:extension base="xs:decimal">
        <xs:attribute name="sizing">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="US"/>
              <xs:enumeration value="European"/>
              <xs:enumeration value="UK"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

(extension表示  
这个元素类型)






# choice元素

- 作用：允许唯一的一个元素从一个组中被选择
- 属性：**minOccurs/maxOccurs**
- 示例：

```
<xs:complexType name="chadState">  
  <xs:choice minOccurs="1" maxOccurs="1">  
    <xs:element ref="selected"/>  
    <xs:element ref="unselected"/>  
    <xs:element ref="dimpled"/>  
    <xs:element ref="perforated"/>  
  </xs:choice>  
  <xs:attribute name="candidate" type="candidateType"/>  
</xs:complexType>
```



# sequence元素

- 作用：给一组元素一个特定的序列
- 示例：

```
<xs:element name="zooAnimals">
  <xs:complexType>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="elephant"/>
      <xs:element name="bear"/>
      <xs:element name="giraffe"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```



# 用schema的数据及元素类型声明XML文档的元素和属性

- 声明元素
  - **<xs:element>**
- 声明属性
  - **<xs:attribute>**



# 声明元素的示例

## 声明一个元素

```
<xs:element name="cat" type="xs:string"/>
<xs:element name="dog" type="xs:string"/>
<xs:element name="redDog" type="xs:string" substitutionGroup="dog" />
<xs:element name="brownDog" type="xs:string" substitutionGroup="dog" />
<xs:element name="pets">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="cat"/>
      <xs:element ref="dog"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```





# 声明元素的方法

1

使用内置的数据类型

```
<xs:element name="cat" type="xs:string"/>  
<xs:element name="dog" type="xs:string"/>  
<xs:element name="redDog" type="xs:string" substitutionGroup="dog" />  
<xs:element name="brownDog" type="xs:string" substitutionGroup="dog" />
```



# 声明元素的方法（续）

2

```
<xs:simpleType name="shapes">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="triangle"/>  
    <xs:enumeration value="rectangle"/>  
    <xs:enumeration value="square"/>  
  </xs:restriction>  
</xs:simpleType>  
<xs:element name="geometry" type="shapes"/>
```

使用用户定义的simpleType

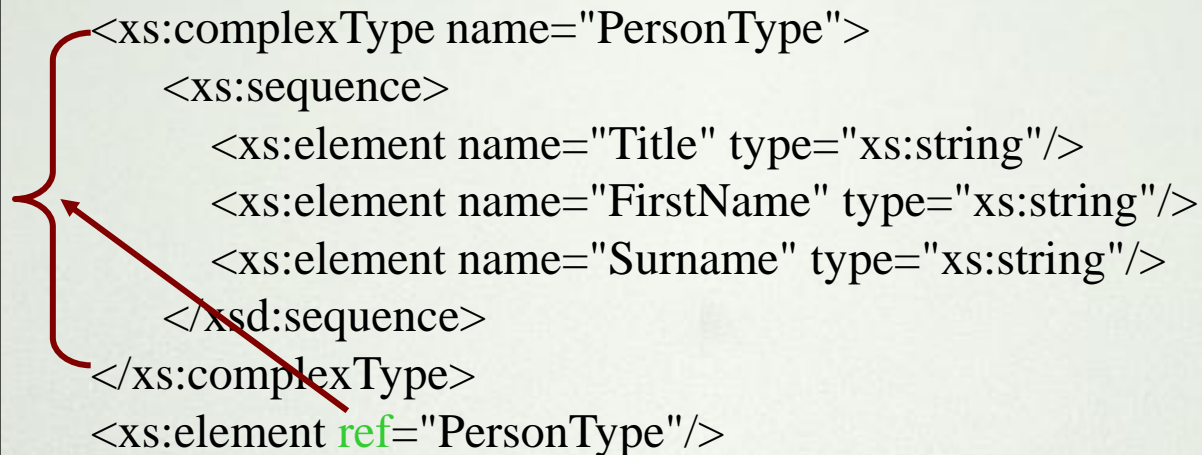
```
<xs:element name="geometry">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:enumeration value="triangle"/>  
      <xs:enumeration value="rectangle"/>  
      <xs:enumeration value="square"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

# 声明元素的方法（续）

3

引用已经定义的元素


```
<xs:complexType name="PersonType">
  <xs:sequence>
    <xs:element name="Title" type="xs:string"/>
    <xs:element name="FirstName" type="xs:string"/>
    <xs:element name="Surname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
<xs:element ref="PersonType"/>
```



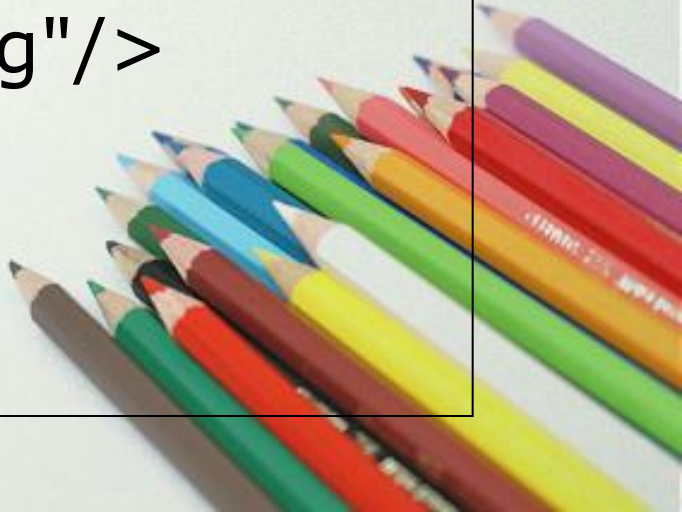
# 声明元素的方法（续）

4

使用complexType



```
<xs:element name="pets">  
  <xs:complexType>  
    <xs:choice minOccurs="0"  
      maxOccurs="unbounded">  
      <xs:element ref="cat"/>  
      <xs:element ref="dog"/>  
    </xs:choice>  
  </xs:complexType>  
</xs:element>
```





# 声明属性的方法

**<xs:attribute name="*name*"  
type="*type*" use="*XXX*" >**

required、  
optional、  
prohibited

简单类型  
复杂类型



# Schema 样例详解

```
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
  targetNamespace="http://tempuri.org/po.xsd"
  xmlns="http://tempuri.org/po.xsd" >
  <xs:element name="purchaseOrder" type="PurchaseOrderType"/>
  <xs:element name="comment" type="xs:string"/>
  <xs:complexType name="PurchaseOrderType">
    <xs:sequence>
      <xs:element name="shipTo" type="USAddress"/>
      <xs:element name="billTo" type="USAddress"/>
      <xs:element ref="comment" minOccurs="0"/>
      <xs:element name="items" type="Items"/>
    </xs:sequence>
    <xs:attribute name="orderDate" type="xs:date"/>
  </xs:complexType>
```

根元素

子元素

shipTo

billTo

coment

items

属性

orderDat

e

# Schema样例详解（续）

```
<xs:complexType name="USAddress">  
  <xs:sequence>  
    <xs:element name="name" type="xs:string"/>  
    <xs:element name="street" type="xs:string"/>  
    <xs:element name="city" type="xs:string"/>  
    <xs:element name="state" type="xs:string"/>  
    <xs:element name="zip" type="xs:decimal"/>  
  </xs:sequence>  
  <xs:attribute name="country"  
    type="xs:NMTOKEN"  
    fixed="US"/>  
</xs:complexType>
```



# Schema样例详解（续）

```
<xs:complexType name="Items">  
  <xs:sequence>  
    <xs:element name="item" minOccurs="0"  
      maxOccurs="unbounded">  
      <xs:complexType>  
        <xs:sequence>  
          <xs:element name="productName" type="xs:string"/>  
          <xs:element name="quantity">  
            <xs:simpleType>  
              <xs:restriction base="xs:positiveInteger">  
                <xs:maxExclusive value="100"/>  
              </xs:restriction>  
            </xs:simpleType>  
          </xs:element>  
        </xs:sequence>  
      </xs:complexType>  
    </xs:element>  
  </xs:sequence>  
</xs:complexType>
```





# Schema样例详解（续）

```
</xs:element>
  <xs:element name="USPrice" type="xs:decimal"/>
  <xs:element ref="comment" minOccurs="0"/>
  <xs:element name="shipDate" type="xs:date"
minOccurs="0"/>
</xs:sequence>
  <xs:attribute name="partNum" type="SKU" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<!-- Stock Keeping Unit, a code for identifying products -->
<xs:simpleType name="SKU">
  <xs:restriction base="xs:integer">
    <xs:minInclusive="2"/>
    <xs:maxInclusive="10"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```



Order.xsd

# 分析XML实例，书写Schema文件

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

```
<学生名册>
```

```
  <学生 学号="1">
```

```
    <姓名>张三</姓名>
```

```
    <性别>男</性别>
```

```
    <年龄>20</年龄>
```

```
  </学生>
```

```
  <学生 学号="2">
```

```
    <姓名>李四</姓名>
```

```
    <性别>女</性别>
```

```
    <年龄>19</年龄>
```

```
  </学生>
```

```
  <学生 学号="3">
```

```
    <姓名>王五</姓名>
```

```
    <性别>男</性别>
```

```
    <年龄>21</年龄>
```

```
  </学生>
```

```
</学生名册>
```



# 总结

- **Schema**是另一种文档类型定义，它遵循**XML**的语言规范。
- **Schema**是可扩展的，支持命名空间；
- **Schema**支持更多的数据类型与元素类型；
- **Schema**用**element**声明元素，用**attribute**声明元素的属性；
- **Schema**用**simpleType**定义简单类型，用**complexType**定义复杂类型。



北京圣思园科技有限公司

<http://www.shengsiyuan.com>

All Rights Reserved

