**为xml文档编写一个schema文档，并XML Spy校验**

**1.**

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="学生名册">

<xs:complexType>

<xs:sequence>

<xs:element ref="学生" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="学生">

<xs:complexType>

<xs:sequence>

<xs:element ref="姓名"/>

<xs:element ref="性别"/>

<xs:element ref="年龄"/>

</xs:sequence>

<xs:attribute name="学号" type="xs:int" use="required"/>

</xs:complexType>

</xs:element>

<xs:element name="姓名" type="xs:string"/>

<xs:element name="性别">

<xs:simpleType>

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

<xs:enumeration value="女"/>

<xs:enumeration value="男"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element name="年龄" type="xs:string"/>

</xs:schema>

**2.**

<?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>男</性别>

<年龄>21</年龄>

</学生>

<学生 学号="3">

<姓名>王二</姓名>

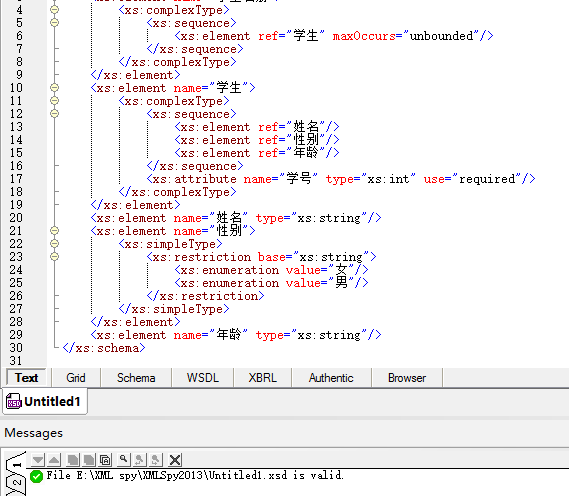
<性别>男</性别>

<年龄>22</年龄>

</学生>

</学生名册>

1. 运行结果截图



**二、小论文------对XML schema文档结构的分析**

# 对Xml Schema 文档结构的分析

**摘要**：随着XML课程的跟进，我们对XML的了解也是越来越深入，知道了XML 应用于 Web 开发的许多方面，常用于简化数据的存储和共享。在最近，了解到了DTD与XML schema,使用DTD虽然带来较大的方便，但是，DTD存在一些缺陷：一是它用不同于XML的语言编写，需要不同的分析器技术。这增加了工具开发商的负担，降低了软件瘦身的可能性，此外开发人员需要多学一门语言及其语法。而XML Schema是按标准XML规则编写的，更容易掌握。二是DTD不支持名称空间。随着大部分的数据处理日益以XML为中心，信息的相互联系变得日益普及与深入，名称空间作用也将凸现。三是DTD在支持继承和子类方面的局限性。由于面向对象技术的出现，对继承和子类的支持已成为软件技术领域的主流概念。最后，DTD没有数据类型的概念，无法对特定元素施加数据类型，对强制性结构化无计可施，例如，如何规定名为Date的数据必须包含有效值。

这些就要依靠XML Schema了。XML Schema不仅可以定义XML文档的结构而且还允许约束文档的内容，这不同于DTD。另外，一个 XML Schema自身就是一个XML文档，其基于标签的语法比DTD中的特殊字符要清楚多了。XML Schema正是针对这些DTD的缺点而设计的，它完全使用XML作为描述手段，具有很强的描述能力、扩展能力和处理维护能力。

所以，本文将通过简单的实例，介绍XML schema的文档结构，让大家对XML schema的文档结构有一个更加深刻的认识。

**关键词**：XML schema

# 目录

[对Xml Schema 文档结构的分析 3](#_Toc14582)

[1.XML Sschema 简介 5](#_Toc11284)

[2.Schema架构图 5](#_Toc25709)

[3.Schema 文档结构 5](#_Toc32559)

[4.Schema元素类型 6](#_Toc26890)

[5.Schema 实例分析 6](#_Toc9128)

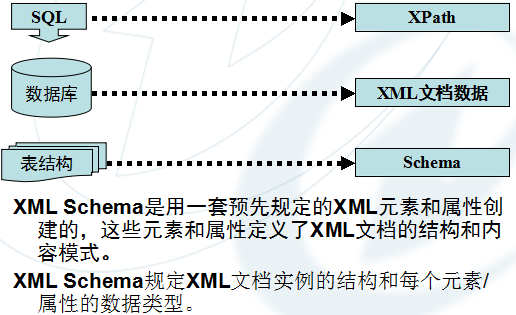
[6.参考文献 9](#_Toc2339)

**1.XML Schema简介**

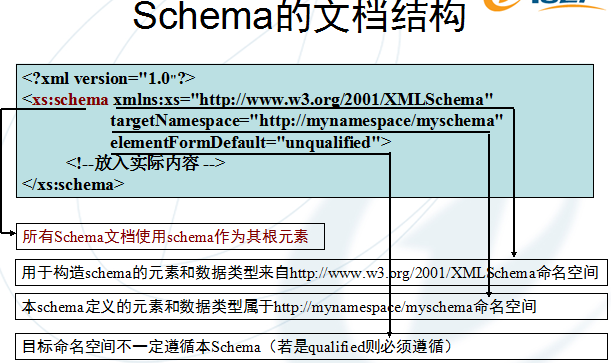
XML Schema 是基于 XML 的 DTD 替代者，XML Schema 可描述 XML 文档的结构，XML Schema 语言也可作为 XSD（XML Schema Definition）来引用。XML Schema 的作用是定义 XML 文档的合法构建模块，类似 DTD。

XML的功能如下：定义可出现在文档中的元素，定义可出现在文档中的属性，定义哪个元素是子元素，定义子元素的次序，定义子元素的数目，定义元素是否为空，或者是否可包含文本，定义元素和属性的数据类型，定义元素和属性的默认值以及固定值。

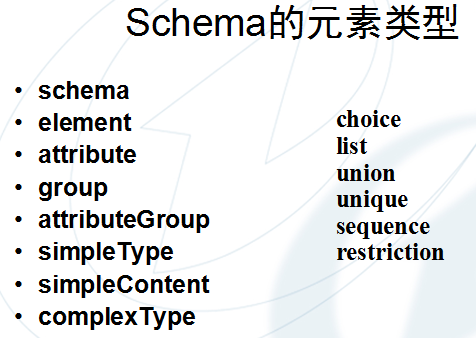
**2.Schema架构图**



**3.Schema 文档结构**



**4.Schema 元素类型**



1. **Schema实例分析**

**首先，先看一下这个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>

**分析**：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解析器要根据某个Schema来验证当前这个文件。

而这一行xsi:noNamespaceSchemaLocation="shiporder.xsd"，则告知XML解析器这个Schema当前所在的位置，在这里的 .xsd文件与 .xml文件在同一个文件夹里。

**接下来，为此XML文件创建Schema文档，同时分析Schema结构：**

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

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

（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，其标准值是 http://www.w3.org/2001/XMLSchema。）

**<xs:element name="shiporder">**

**<xs:complexType>**（因为shiporder元素包含属性以及其他的元素，所以是复杂类型）

**<xs:sequence>**（因为shiporder包含子元素，sequence约束了子元素的顺序）

**<xs:element name="orderperson" type="xs:string"/>**（prderperson不包含任何属性和子元素，为简单类型，是字符型)

**<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">**

(通过maxOccurs来约束item出现的最大次数）

**<xs:complexType>**

**<xs:sequence>**

**<xs:element name="title" type="xs:string"/>**

**<xs:element name="note" type="xs:string" minOccurs="0"/>**

（通过minOccurs来约束item出现的最小次数）

**<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"/>**

（定义属性，由于此属性为必选属性，所以use=”required”）

**</xs:complexType>**

**</xs:element>**

**</xs:schema>**

1. **参考文献：**

1.<http://www.cnblogs.com/xiaoxiaogogo/p/3601299.html>

2.<http://www.runoob.com/schema/schema-intro.html>

3.http://blog.csdn.net/wangjian\_an/article/details/51770454