**1. XML 概念和体系**

**1.1XML 概念**

XML 指可扩展标记语言（EXtensible Markup Language）

XML 没有预定义标签，需要自行定义标签

<标签></标签>

**1.2XML 的特点**

XML 数据以纯文本格式存储

实现不同应用程序之间的数据通信

实现不同平台间的数据通信

实现不同平台间的数据共享

使用 XML 将不同的程序、不同的平台之间联系起来

|  |
| --- |
| 图片1 |

**1.3XML 的作用**

数据存储和数据传输

可以用来做配置文件

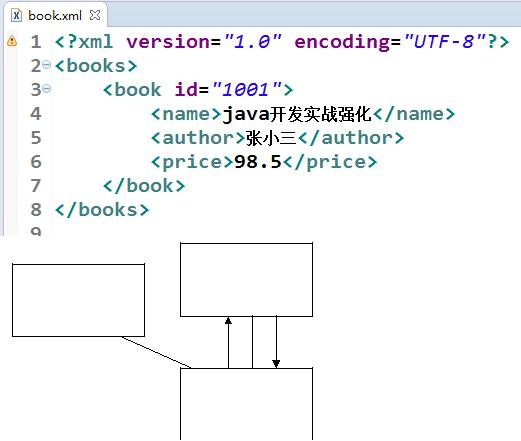
javaee开发中用xml来作配置文件,SSM,DUBOO分布式框架等等

**1.4XML 和 HTML 之间的差异**

**HTML里面的标签是已经设计好的,我们需要学习使用**

**XML是自己来定义标签,以<标签></标签>格式定义即可**

**1.5 下面是XML的结构样子:一个标准的 XML 文档**



根元素

属性 <books>

“id”

父 子

元素

<book>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 元素 | | |  |  |  | 元素 | | |  | 元素 | |  |
|  | <name> | | |  |  |  | <author> | | |  | <price> | |  |
|  |  |  |  | 同级兄弟 | | | |  |  |  |  |  |  |
|  |  |  | |  |  |  |  |  |  |  |  |  |  |
|  | 文本 | | |  |  |  | 文本 | | |  | 文本 | |  |
|  | Java 开发… | | |  |  |  | 张小三 | | |  | 98.5 | |  |
|  |  |  |  |  |  |  |  |  |

|  |
| --- |
| 我们可以看出xml是树形结构  由成对的标签构成元素,元素中间加入文本,元素之间有父子元素结构 |

**1.6 由XML 文件扩展出来的体系(了解)**

1.XHTML可扩展超文本标记语言

2. RSS（广泛用于网上新闻频道） ： Really Simple Syndication 简易信息聚合，使用 RSS 订阅能更快地获取信息，网站提供 RSS 输出，有利于让用户获取网站内容的最新更新

**2.XML 基本语法**

案列: 写出第一个xml文档:

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* <**student**>  <**name id="101"**>张三</**name**>  <**age**>20</**age**>  <**school**>长沙市一中</**school**> </**student**> *<!-- <?xml version='1.0' encoding='UTF-8'?> 文档的申明,必须写在xml文档中第一行 ,对当前xml文件进行约定:  <?xml>文档申明的标签 version:指定解析当前xml文件的xml解析器的版本为1.0 encoding='UTF-8',当前文档的编码类型  -->* |
|  |

### **encoding详解**

在解析这个xml的时候，使用什么编码去解析。 ---解码。

存储这些文字对应的二进制 。 那么这些文字对应的二进制到底是多少呢？ 根据文件使用的编码来得到。

xml编码要和保存文件时指定的编码一致,否则中文会乱码

**2.1XML 的基本语法**

1. 有且只有一个根元素
2. XML 文档声明必须放在文档的第一行
3. 所有标签必须成对出现
4. XML 的标签严格区分大小写
5. XML 元素可以嵌套其他元素
6. XML 中的属性值必须加引号
7. XML 中可以应用适当的注释<!-- -->

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* <**students**>  <**student**>  <**name id="101"**>张三</**name**>  <**age**>20</**age**>  <**school**>长沙市一中</**school**>  </**student**>  <**student**>  <**name id="102"**>李四</**name**>  <**age**>20</**age**>  <**school**>长沙市三中</**school**>  </**student**> </**students**> |

**2.2XML 元素**

**XML 元素指的是开始标签到结束标签的部分**

**一个元素中可以包含**

其他元素、普通文本、属性、以上的混合

**2.3XML 命名规则**

1. 名称可以包含字母、数字及其他字符
2. 名称不能以数字或者标点符号开始
3. 名称不能以字母 xml 开始(xml是关键字)
4. 名称不能包含空格做到见名知意

## **XML 约束(了解)**

如下的文档， 属性的ID值是一样的。 这在生活中是不可能出现的。 并且第二个学生的姓名有好几个。 一般也很少。那么怎么规定ID的值唯一， 或者是元素只能出现一次，不能出现多次？ 甚至是规定里面只能出现具体的元素名字。

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* <**student**>  <**name id="101"**>张三</**name**>  <**name id="102"**>李四</**name**>  <**age**>20</**age**>  <**age**>22</**age**>  <**school**>长沙市一中</**school**>  <**loc**>长沙岳麓区</**loc**> </**student**> |
| 以上student中含有多个name和age标签,对这个数据描述造成模棱两可的问题,解析数据获取不知道取哪个值,解决?  要使用xml的约束 |

## **DTD**

语法自成一派， 早期就出现的。 可读性比较差。

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* **<!DOCTYPE student** [  <!ELEMENT student (name,school,age,loc)> <!--指定约束student标签下只能包含子标签:name,age,school,loc四个标签,数量是一个-->  <!ELEMENT name (#PCDATA)> <!--对name标签的描述,数据类型是:#PCDATA文本类型-->  <!ELEMENT age (#PCDATA)>  <!ELEMENT school (#PCDATA)>  <!ELEMENT loc (#PCDATA)>  ]***>*** <**student**>  <**name**>李四</**name**>  <**school**>长沙市一中</**school**>  <**age**>22</**age**>  <**loc**>长沙岳麓区</**loc**> </**student**> |

引入外部约束

PCDATA(Parsed Character DATA)的意思是被解析的字符数据

### 元素的个数

＋　 一个或多个

\* 零个或多个

? 零个或一个

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* **<!DOCTYPE student SYSTEM "stu.dtd"*>*** *<!--引入外部的约束dtd文件-->* <**student**>  <**name**>李四</**name**>  <**age**>22</**age**>  <**school**>长沙市一中</**school**>  <**school**>雅礼中学</**school**>  <**loc**>长沙岳麓区</**loc**> </**student**> |
| **stu.dtd文件中的内容:**  **<!ELEMENT student** (**name**?,**age**,**school**+,**loc**\*)> *<!--指定约束student标签下只能包含子标签:name,age,school,loc四个标签,而且出现的顺序也要保持一致,数量是一个-->* **<!ELEMENT name** (#PCDATA)> *<!--对name标签的描述,数据类型是:#PCDATA文本类型-->* **<!ELEMENT age** (#PCDATA)> **<!ELEMENT school** (#PCDATA)> **<!ELEMENT loc** (#PCDATA)> |

**3.2 Schema 技术**

Schema 是 DTD 的代替者，名称为 XML Schema，用于描述

XML 文档结构，比 DTD 更加强大，最主要的特征之一就是

XML Schema 支持数据类型

Schema 是用 XML 验证 XML 遵循 XML 的语法

Schema 可以用能处理 XML 文档的工具处理

在一个xml文档中要引入多个约束文件时得使用Schema

**3.3Schema 的文档结构**

文件的文档解释：

|  |
| --- |
| *<?***xml version='1.0' encoding='UTF-8'***?>* <**student xmlns="http://www.w3school.com.cn" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.w3school.com.cn abc.xsd"**> *<!--*引入外部Schema约束-->   <**name**>李四</**name**>  <**age**>22</**age**>  <**school**>长沙市一中</**school**>  <**school**>长沙市一中</**school**>  <**loc**>长沙岳麓区</**loc**> </**student**> |
| *<?***xml version="1.0"***?>* <**xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"  targetNamespace="http://www.w3school.com.cn"  xmlns="http://www.w3school.com.cn"  elementFormDefault="qualified"**>   <**xs:element name="student"**>  <**xs:complexType**> *<!--complexType:表示当前元素是复合元素(含有子元素的称为复合元素)-->* <**xs:sequence**> *<!--表示子元素按序列来约束-->* <**xs:element name="name" type="xs:string"**/>  <**xs:element name="age" type="xs:string"**/>  <**xs:element name="school" type="xs:string"**/> *<!--如果含有多个相同的子元素,就写多个-->* <**xs:element name="school" type="xs:string"**/>  <**xs:element name="loc" type="xs:string"**/>  </**xs:sequence**>  </**xs:complexType**>  </**xs:element**>  </**xs:schema**> *<!-- xmlns: (相当于java中import)表示引入Schema的命名空间,用来验证Schema文件结构本身是否合法,xs表示xmlns的别名 targetNamespace:目标命名空间,下面的标签都是来自于目标命名空间的 xmlns="http://www.w3school.com.cn":引入目标命名空间 elementFormDefault="qualified" :引入xsd的全局元素的时候需要添加前缀xs unqualified: 引入xsd的全局元素的时候不需要添加前缀xs -->* |

所有的 Schema 文档使用 schema 作为其根元素

<http://www.w3.org/2001/XMLSchema>:用于验证当前 Schema文档的命名空间(用于验证 Schema 本身)同时它还规定了来自命名空间 <http://www.w3.org/2001/XMLSchema>的元素和数据类型

xmlns 相当于 java 中的 import

## **Schema**

其实就是一个xml ， 使用xml的语法规则， xml解析器解析起来比较方便 ， 是为了替代DTD 。

但是Schema 约束文本内容比DTD的内容还要多。 所以目前也没有真正意义上的替代DTD

DTD的约束,类似于java的继承(单一继承)

Schema约束,类似于java的实现接口(多实现)

名称空间的作用

一个xml如果想指定它的约束规则， 假设使用的是DTD ，那么这个xml只能指定一个DTD ， 不能指定多个DTD 。 但是如果一个xml的约束是定义在schema里面，并且是多个schema，那么是可以的。简单的说： 一个xml 可以引用多个schema约束。 但是只能引用一个DTD约束。

名称空间的作用就是在 写元素的时候，可以指定该元素使用的是哪一套约束规则。

重点:

|  |
| --- |
| 会写xml文档及引入Schema约束  学习SSM时经常要手动引入Schema约束 |