# xml

xml是干什么的

1.传输数据

！2.配置文件 Properties

学些什么内容

1.xml格式

2.解析xml文件

## 1.xml是什么

xml:Extensible Markup Language 可扩展标记语言

由w3c组织提供，2000年开发出1.0版本

xml是一种存储格式，本质上是字符串

## 2.xml是如何保存数据的

场景：保存中国的省份和城市信息

在XML中, 允许用户自己定义标签, 标签分为开始标签和结束标签, 开始标签和结束标签之间又可以嵌套其他的标签.

正是利用标签来保存数据, 利用标签之间的嵌套关系来保存数据之间的层级关系.

**3.xml的应用场景**

(1)传输数据

由于xml本质上是一段字符串, 具有跨平台的特性, 因此可以在不同系统之间进行数据交换。

比如天气数据，xml接口：<http://flash.weather.com.cn/wmaps/xml/china.xml>

(2)用作配置文件

由于xml可以保存有结构的数据, 因此xml常被用来用作应用程序的配置文件。

**4.xml文件、xml校验（了解）**

我们可以使用浏览器对XML进行校验。

**5.xml语法**

**5.1．文档声明**

可以用来声明xml文档的基本属性信息, 解析器会根据文档声明来决定如何解析这个xml文档.

注意:

一个xml文档有且仅有一个文档声明.(规范要求)

文档声明只能放在xml文档的第一行, 并且前面不能有任何内容

如果一个xml文档没有文档声明, 该文档是一个格式不良好的xml文档.

<?xml version=”1.0” encoding=”utf-8”?>

其中version用来指定xml文档所遵循的xml版本, 目前就是1.0， encoding属性用来通知解析器使用什么编码来解析当前xml文档. 注意, encoding属性指定的编码和xml文档保存时的编码必须得一致!!!

**5.2．元素**

一个标签就是一个元素, 如: <中国></中国>

标签体：标签分为开始标签和结束标签, 开始标签和结束标签之间的文本。

如：<a>xxxxx</a>

自闭标签：如果一个元素不包含标签体也不包含其他元素，那么可以将开始标签和结束标签合并

如：<a></a> ---> <a/>

根标签：一个xml文档有且仅有一个根标签

标签可以包含子标签，但一定要合理的嵌套, 不能出现交叉嵌套.

比如:

<a>welcome to <b>www.tarena.com.cn</a></b>

<a>welcome to www.tarena.com.cn</b></a>

<a>welcome to www.tarena.com.cn<b/></a>

<a>welcome to www.tarena<b/>.com.cn</a>

XML元素的命名规范:

区分大小写, 如: <a>和<A>是两个不同的标签

不能以数字或标点符号开头, 如: <123a>

不能以xml（或Xml、或XML）开头

不能包含空格

不能包含冒号

**5.3．属性**

一个标签可以有多个属性，多个属性之间用空格隔开，属性值一定要用双引号（"）或单引号（'）引起来，例如：

<中国 pyname=”zhongguo” gdp=”80万亿”>

属性和元素具有相同的命名规范(参考上面)

**5.4.注释 (了解)**

格式: <!-- 一段注释 -->

注意:

注释不能放在文档声明的前面

注释不能交叉嵌套, 如:

<!-- <!-- 注释内容 注释内容--> 注释内容 --> 这是不合理的注释

**5.5.转义字符 (了解)**

用其他符号来代替特殊符号

< &lt;

> &gt;

“ &quot;

‘ &apos;

& &amp;

**6.xml约束（了解）**

在xml技术里，可以编写一个文档来约束一个xml文档的写法，这称之为XML约束。

分为两种：DTD和Schema W3C组织提供

**7.xml解析**

xml有两种解析方式：Dom解析和SAX解析

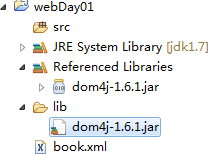
解析开发包：jaxp（sun） jdom dom4j

dom4j是简单易用，功能强大的开发包

案例：

第一步：创建book.xml

第二步：导包



第三步：案例开发

快速接收返回值：alt+shift+l

快速抽取方法：alt+shift+m

读取第一本书的书名

|  |
| --- |
| //创建解析器  SAXReader reader = **new** SAXReader();  //将xml文档解析成Document对象  Document dom = reader.read("book.xml");  //获取根元素  Element root = dom.getRootElement();  //获取第一本书  Element bookEle = root.element("书");  //获取第一本书的书名元素  Element bookName = bookEle.element("书名");  //获取书名  String text = bookName.getText();  System.***out***.println("第一本书的书名为："+text); |

查询第二本书的售价，并输出到控制台

|  |
| --- |
| **package** cn.tedu;  **import** java.util.List;  **import** org.dom4j.Document;  **import** org.dom4j.DocumentException;  **import** org.dom4j.Element;  **import** org.dom4j.io.SAXReader;  **public** **class** Demo2 {  **public** **static** **void** main(String[] args) **throws** DocumentException {  // 2.查询第二本书的售价，并输出到控制台  *findPrice*();    }  **private** **static** **void** findPrice() **throws** DocumentException {  //创建解析器  SAXReader reader = **new** SAXReader();  //获取Document对象  Document dom = reader.read("book.xml");  //获取根元素  Element root = dom.getRootElement();  //获取根元素下所有元素  List<Element> list = root.elements();  //获取第二本书  Element bookEle2 = list.get(1);  //获取售价节点  Element price = bookEle2.element("售价");  //获取售价  String text = price.getTextTrim();    System.***out***.println(text);  }  } |

3.给第一本书添加一个特价节点（2种方式）

创建工具类：XMLUtils

|  |
| --- |
| **package** cn.tedu.utils;  **import** org.dom4j.Document;  **import** org.dom4j.DocumentException;  **import** org.dom4j.io.SAXReader;  **public** **class** XMLUtils {    /\*\*  \* 解析xml，获取document对象  \*/  **public** **static** Document getDom(String xmlPath){  **try** {  //创建解析器  SAXReader reader = **new** SAXReader();  //获取Document对象  Document dom = reader.read(xmlPath);  **return** dom;  } **catch** (DocumentException e) {  e.printStackTrace();  **throw** **new** RuntimeException();  }  }    }  案例代码  //获取dom  Document dom = XMLUtils.*getDom*("book.xml");  //获取根并获取第一本书  Element bookEle = dom.getRootElement().element("书");    //方法一：  /\*Element price2 = bookEle.addElement("特价");  price2.setText("28.8元");\*/  //方法二:创建一个游离的element，再将游离的element添加到父元素  Element price3 = DocumentHelper.*createElement*("特价");  price3.setText("26.6元");  bookEle.add(price3);    //将document对象写到xml中  XMLWriter writer = **new** XMLWriter(  **new** FileOutputStream(**new** File("book.xml")),  OutputFormat.*createPrettyPrint*());    writer.write(dom);  writer.close();    System.***out***.println("执行完成"); |

4.给第二本书在作者节点前插入一个特价节点

|  |
| --- |
| 在工具类加入方法：  **public** **static** **void** write2xml(Document dom,String xmlPath){  **try** {  //将document对象写到xml中  XMLWriter writer = **new** XMLWriter(  **new** FileOutputStream(**new** File(xmlPath)),  OutputFormat.*createPrettyPrint*());  writer.write(dom);  writer.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  代码为：  //获取dom  Document dom = XMLUtils.*getDom*("book.xml");  //获取第二本书  Element bookEle2 = (Element) dom.getRootElement().elements().get(1);  //创建游离元素  Element price2 = DocumentHelper.*createElement*("特价");  price2.setText("8.8元");  //获取第二本书的list  List list = bookEle2.elements();  list.add(1, price2);    //写入xml  XMLUtils.*write2xml*(dom, "book.xml"); |