# 1. XML概要

XML文件是纯文本文件，用来描述数据，常用做项目的配置文件。XML是可扩展标记语言（Extensible Markup Language），XML文件使用xml作为文件扩展名。下面介绍XML中的元素。

## 1.1 XML文档声明

所有XML文件内容的第一行是XML的文档声明，一般文档声明格式如下：

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

其中version="1.0"表示XML的版本号，目前一般就使用1.0的版本；encoding="utf-8"表示该文件以UTF-8方式编码，这是XML最常用的编码。同时要注意保存文件时，应确保保存文件的编码和encoding中设置的编码一致，一般IDE会自动帮我们保存为相一致的编码。

## 1.2 标签和属性

XML通过标签和属性存储数据。XML中的标签和属性是自己定义的。比如现在用student.xml来保存学生的数据，内容如下：

|  |
| --- |
| <?xml version="1.0" encoding="utf-8" ?>  <students>  <student age="12">张三</student>  <student age="13">李四</student>  </students> |

这个xml中就存储了两个学生，保存了每个学生的姓名和年龄。students和student就是标签名，标签用尖括号括起来，每个标签都有开始标签和结束标签。结束标签就是在标签名前面加斜杠。

标签里面可以直接有文本内容，也可以再嵌套标签，这样就形成了有层次的数据结构。如果一个标签没有内容，则是空标签，比如：<student></student>。对于空标签，可简写成：<student />。

XML文件是区分大小写的，并且标签名不能以数字开头。需要注意的是，一个XML文档中只能有一个根标签。在例子中，根标签是students，不能再写一个根标签。

在上述例子中，写在标签尖括号中的age就是一个属性。标签中可定义多个属性，而属性值必须放在引号里面。单引号和双引号都可以，但是不能单双引号混用。

## 1.3 注释、转义字符和CDATA块

XML中可以使用注释，语法是：<!-- 注释内容 -->。

XML中，像大于号>、小于号<这样的字符已经被标签使用了，如果把这些字符作为数据存储的话，就需要用到转义字符，否则会产生歧义。

|  |  |
| --- | --- |
| 需转义的字符 | 对应的转义字符 |
| < （小于号） | &lt; |
| > （大于号） | &gt; |
| " （引号） | &quot; |
| & | &amp; |
| 空格 | &nbsp; |

若一段数据中，有大量的特殊字符要转义，那么可以使用CDATA块，将内容包含在CDATA块中。对于CDATA块中的内容，XML只会将它解析成普通的文本。CDATA块的语法是：

<![CDATA[内容]]，比如：

|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"** *?>* <**students**>  <**student age="12"**>张三</**student**>  <**student age="13"**>李四</**student**>  <![CDATA[<table>我只是文本</table>]]> </**students**> |

## 1.4 处理指令

处理指令是用来指示XML解析引擎如何去解析XML文档内容的。处理指令以“<?>”开头，以“?>”结尾，XML声明语句就是最常见的处理指令。

除此之外，XML还能还有样式，也是通过处理指令完成的，这里就不讲了，因为基本不用。

# 2. 解析XML

XML中存储了数据，那么在程序中要想使用这些数据，必须用程序对XML进行解析来获得数据。用程序读取XML文件内容并获得数据就叫做XML解析。

Java中，常用的XML解析方式是SAX解析和DOM解析。SAX我们使用官方提供的API，DOM我们使用DOM4J第三方库。

## 2.1 SAX解析

SAX解析工具有关的包在org.xml.sax和javax.xml.parsers中。解析的步骤如下：

（1）首先通过SAXParserFactory工厂类创建一个SAXParser解析器对象。

（2）调用解析器的parse方法进行解析。方法原型是：

public void parse(String uri, DefaultHandler dh)。

参数1：表示XML文件的地址，可以是网络资源，也可以是本地资源地址。此函数的其他重载中可以传递File类型参数和InputStream类型参数。

参数2：要一个DefaultHandler的子类对象。SAX是基于事件处理XML的，DefaultHandler中定义了SAX的所有事件方法，比如开始读到文件的方法、读到一个标签的方法、读到一个文本的方法等。我们只需要在子类中实现具体的事件处理方法即可。

例子：简单使用下SAX。

（1）主类

|  |
| --- |
| **package** com.zhang.test;  **import** javax.xml.parsers.SAXParser; **import** javax.xml.parsers.SAXParserFactory;  **public class** Demo {  **public static void** main(String[] args) **throws** Exception {  *// SAXParserFactory是单例，通过静态方法拿到工厂对象再创建SAXParser解析器* SAXParser parser = SAXParserFactory.*newInstance*().newSAXParser();  *// 读取示例的student.xml文件。创建MyXMLHandler对象。MyXMLHandler类下面创建。* parser.parse(**"src/student.xml"**, **new** MyXMLHandler());  } } |

（2）MyXMLHandler类继承DefaultHandler类：

|  |
| --- |
| **package** com.zhang.test;  **import** org.xml.sax.Attributes; **import** org.xml.sax.SAXException; **import** org.xml.sax.helpers.DefaultHandler;  **public class** MyXMLHandler **extends** DefaultHandler {  *// 接收文档开始的通知* **public void** startDocument() **throws** SAXException {  System.***out***.println(**"事件————开始读取文档"**);  }   *// 接收元素（即标签）开始的通知* **public void** startElement(String uri, String localName, String qName, Attributes attributes) **throws** SAXException {  System.***out***.println(**"事件————读到标签："** + qName);  }   *//接收元素中字符数据的通知* **public void** characters(**char**[] ch, **int** start, **int** length) **throws** SAXException {  System.***out***.println(**"事件————读到文本："** + **new** String(ch, start, length));  }   *// 接收元素（标签）结束的通知* **public void** endElement(String uri, String localName, String qName) **throws** SAXException {  System.***out***.println(**"事件————读完标签："** + qName);  }   *// 接收文档结束的通知* **public void** endDocument() **throws** SAXException {  System.***out***.println(**"事件————读完文档"**);  } } |

以上重写的就是主要事件。其中，startDocument()和endDocument()是文档开始和结束的事件，一般情况下不需要有具体处理，可不重写。除非需要在读取文档前需要初始化资源等操作。

重要的数据处理集中在以下三个方法中：

（1）public void startElement(String uri, String localName, String qName, Attributes attributes)：每读取到一个开始标签就会调用此方法。uri是名称空间，localName是本地名称。这两个学完名称空间之后就会知道。现在只需要用到qName，这就是该标签的名称。参数attributes中就存放了该标签的所有的属性和值。该节点通常表示读到了某条数据。

（2）public void endElement(String uri, String localName, String qName)：每读到一个结束标签就会调用此方法。参数的含义和方法1中相同。

（3）public void characters(char[] ch, int start, int length)：每读到文本数据时调用此方法。ch参数表示字符数组，start和length表示开始位置和长度。因此用String data = new String(ch, start, length)；来得到文本数据。

上述代码的运行结果是：

|  |
| --- |
| 事件————开始读取文档  事件————读到标签：students  事件————读到文本：    事件————读到标签：student  事件————读到文本：张三  事件————读完标签：student  事件————读到文本：    事件————读到标签：student  事件————读到文本：李四  事件————读完标签：student  事件————读到文本：    事件————读到文本：<table>我只是文本</table>  事件————读到文本：  事件————读完标签：students  事件————读完文档 |

解释：为什么有空白呢？因为...如果写成紧缩的格式就没有了，比如这样：...