chapter21-XML技术

主要内容

1. JDOM简介
2. JDOM解析XML
3. DOM4J简介
4. DOM4J解析XML

JDOM

- JDOM是一种使用 XML 的独特 Java 工具包，用于快速开发 XML 应用程序。它的设计包含 Java 语言的语法乃至语义，JAXP（用于 XML 语法分析的 Java API）包含了三个软件包
  - ■org.w3c.dom ，W3C 推荐的用于 XML 标准规划文档对象模型的 Java 工具
  - ■org.xml.sax ，用于对 XML 进行语法分析的事件驱动的简单 API
  - ■javax.xml.parsers ，工厂化工具，允许应用程序开发人员获得并配置特殊的语法分析器工具. JDOM 能够替换 org.w3c.dom 软件包来有计划地操作 XML 文档

JDOM简介：

JDOM是一个开源项目，它基于树型结构，利用纯JAVA的技术对XML文档实现解析、生成、序列化以及多种操作。

JDOM 直接为JAVA编程服务。它利用更为强有力的JAVA语言的诸多特性（方法重载、集合概念等），把SAX和DOM的功能有效地结合起来，以弥补DOM及SAX在实际应用当中的不足之处。

即：JDOM=DOM的修改性+SAX的大文件可读性

JDOM

•在使用设计上尽可能地隐藏原来使用XML过程中的复杂性。利用JDOM处理XML文档将是一件轻松、简单的事。

• **JDOM** 主要用来弥补DOM及SAX在实际应用当中的不足之处。这些不足之处主要在于SAX没有文档修改、随机访问以及输出的功能，而对于DOM来说，JAVA程序员在使用时来用起来总觉得不太方便。

JDOM

- DOM的缺点主要是由于DOM是一个接口定义语言（IDL），它的任务是在不同语言实现中的一个最低的通用标准，并不是为JAVA特别设计的。
- 在 JDOM 中，XML 元素就是 Element 的实例，XML 属性就是 Attribute 的实例，XML 文档本身就是 Document 的实例

W3c的局限性

- 语言独立

  ■DOM 并不是用人们心目中的 Java 语言设计 的。虽然这种方法保留了在不同语言中非常相似的 API，它也使那些习惯 Java 语言的程序员感到更麻烦。

  - 例如：Java 语言内建了一种 String 类，而 DOM 则规范定义了自己的 Text 类。

- 严格的层次结构

  ◆DOM API 直接沿袭了 XML 规范。在 XML 中，每件东西都是一个结点，因此您能在 DOM中找到一个几乎每件东西都可以扩展的基于Node 的接口和返回 Node 的一系列方法。就多态性的观点来讲，它是优秀的，但鉴于如上解释，它在 Java 语言中的应用是困难而且不便的，其中从 Node 向叶类型作显式向下类型转换会导致代码的冗长和难以理解。

W3c的局限性

- 接口驱动

  ■公共 DOM API 仅由接口组成。w3c 对提供实现并不感兴趣，它只对定义接口（比较有意义）感兴趣。但它也意味着作为 Java 程序员使用 API 在创建 XML对象时增加了负担，因为 w3c 标准大量使用工厂化的类和类似的灵活的但不直接的模式。

  ■ 在某些应用中，XML 文档是仅由语法分析器建立的，而从不会由应用程序级代码建立，这是不相关的。但是，随着 XML 更广泛的使用，并不是所有问题都继续需要由语法分析器来驱动。应用程序的开发人员需要一个更方便的方法有计划地构造 XML 对象。

- 对于程序员，这些约束意味着庞大（在内存占用和接口大小方面）的和难掌握的API，学习和使用都很难

JDOM的优势

- JDOM 是作为一种轻量级 API 被制定的，最主要的是它是以 Java 为中心的。它在遵循 DOM 主要规则的基础上除去了上述缺点
- JDOM 是 Java 平台专用的只要有可能，API 都使用 Java 语言的内建String 支持，因此文本值也适用于 String 。它还可利用 Java 2 平台的类集，如 List 和Iterator ，给程序员提供了一个丰富的并且和 Java 语言类似的环境。
- 在 JDOM 中，XML 元素就是 Element 的实例，XML 属性就是 Attribute 的实例，XML文档本身就是 Document 的实例。由于在 XML 中所有这些都代表了不同的概念，因此它们总是作为自己的类型被引用，而不是作为一 个含糊的“结点”。
- 类驱动，因为 JDOM 对象就是像 Document 、Element 和 Attribute 这些类的直接实例，因此创建一个新 JDOM 对象就如在 Java 语言中使用 new 操作符一样容易。它还意味着不需要进行工厂化接口配置 -- JDOM 的使用是直截了当的。

JDOM的优势

- JDOM 对象就是像 Document、Element 和 Attribute 这些类的直接实例 ，因此创建一个新 JDOM 对象就如在Java 语言中使用 new 操作符一样容易。JDOM 的使用是直截了当的。
- JDOM 使用标准的 Java 编码模式。只要有可能，它使用 Java new 操作符而不使用复杂的工厂模式，使对象操作即便对于初学用户也很方便。

JDOM 包概览

- JDOM是由以下几个包组成的

1.org.jdom 包含了所有的xml文档要素的java类
2 org.jdom.adapters 包含了与dom适配的java类
3 org.jdom.filter 包含了xml文档的过滤器类
4 org.jdom.input 包含了读取xml文档的类
5 org.jdom.output 包含了写入xml文档的类
6 org.jdom.transform 包含了将jdom xml文档接口转换为其他xml文档接口
7 org.jdom.xpath 包含了对xml文档xpath操作的类

JDOM

- org.jdom这个包里的类是你解析xml文件后所要用到的所有数据类型
  - Attribute
  - CDATA
  - Coment
  - DocType
  - Document
  - Element
  - EntityRef
  - Namespace
  - ProscessingInstruction
  - Text

JDOM

输入类，一般用于文档的创建工作

- SAXBuilder
- DOMBuilder
- ResultSetBuilder

org.jdom.output输出类，用于文档转换输出

- XMLOutputter
- SAXOutputter
- DomOutputter
- JTreeOutputter

JDOM主要方法

Document类

Document的操作方法：

```
Element root = new Element("GREETING");

Document doc = new Document(root);

root.setText("Hello JDOM!");
```

或者简单的使用

```
Document doc = new Document(new
Element("GREETING").setText("Hello JDOM!t"));
```

JDOM与DOM的比较

这点和DOM不同。Dom则需要更为复杂的代码，如下：

```
DocumentBuilderFactory factory
=DocumentBuilderFactory.newInstance();

DocumentBuilder builder =factory.newDocumentBuilder();

Document doc = builder.newDocument();

Element root =doc.createElement("root");

Text text = doc.createText("This is the root");

root.appendChild(text);

doc.appendChild(root);
```

JDOM主要方法

- 可以使用SAXBuilder的build方法来解析一个流从而得到一个Document对象
  - Document build(java.io.File file)
  - Document build(org.xml.sax.InputSource in)
  - Document build(java.io.InputStream in)
  - Document build(java.io.InputStream in,java.lang.String systemId)
  - Document build(java.io.Reader characterStream)
  - Document build(java.io.ReadercharacterStream, java.lang.String systemId)
  - Document build(java.lang.String systemId)
  - Document build(java.net.URL url)

JDOM主要方法

DOM的Document和JDOM的Document之间的相互转换使用方法

- DOMBuilder builder = new DOMBuilder();
- org.jdom.Document jdomDocument = builder.build(domDocument);
- DOMOutputter converter = new DOMOutputter();// work with the JDOM document...
- org.w3c.dom.Document domDocument = converter.output(jdomDocument);
- // work with the DOM document...

XML文档的输出

XMLOutPutter类：

JDOM的输出非常灵活,支持很多种io格式以及风格的输出

```
Document doc = new Document(...);
XMLOutputter outp = new XMLOutputter();
outp.output(doc, fileOutputStream); // Raw output
outp.setTextTrim(true); // Compressed output
outp.output(doc, socket.getOutputStream());
outp.setIndent(" ");// Pretty output
outp.setNewlines(true);
outp.output(doc, System.out);
```

JDOM的主要操作类

No.	类名称	描述
1	Document	Document类定义了一个XML文件的各种操作，用户可以通过它所提供的方法来存取根元素以及存取处理命令文件层次的相关信息。
2	Element	Element类定义了一个XML元素的各种操作，用户可以通过它所提供的方法得到元素的文字内容、属性值以及子节点。
3	Attribute	Attribute类表示了XML文件元素中属性的各个操作。
4	XMLOutputter	XMLOutputter类会将一个JDOM结构树格式化为一个XML文件，并且以输出流的方式加以输出。
5	DOMBuilder	DOMBuilder类用来建立一个JDOM结构树。

JDOM的主要方法使用（一）

Ducument类

Document的操作方法：

```
//创建一个element对象
 Element root = new Element("GREETING");
//创建一个document对象来获取根元素
 Document doc = new Document(root);
//给根元素添加内容
 root.setText("Hello JDOM!");
 //或者简单的使用
 Document doc = new Document(new   Element("GREETING").setText("Hello JDOM! "));
```

JDOM的主要方法使用（二）

从文件、流、系统ID、URL得到Document对象：

```
//根据SAX程序创建一个Jdom对象
    SAXBuilder builder = new SAXBuilder();
//创建document对象
    Document doc = builder.build(url);
```

JDOM的主要方法使用（三）

XML文档输出

XMLOutPutter类：JDOM的输出非常灵活,支持很多种io格式以及风格的输出

```
XMLOutputter  out = new XMLOutputter();
output.setFormat(output.getFormat
  ().setEncoding("utf-8")) ;
  out.output(doc, fileOutputStream);
```

JDOM的主要方法使用（四）

Element

浏览Element树

```
Element root = doc.getRootElement();//获得根元素element
//（这里的List是java.util.List）
List allChildren = root.getChildren();
// 获得所有子元素的一个list
List namedChildren = root.getChildren("name");
// 获得指定名称子元素的list
Element child = root.getChild("name");
//获得指定名称的第一个子元素
```

JDOM的主要方法使用（六）

Attribute类

<table width="100%" border="0"> </table>

```
String width = table.getAttributeValue("width");
//获得attribute
int border = table.getAttribute("width").getIntValue();
table.setAttribute(“vspace”, “0”); //设置attribute
table.removeAttribute("vspace");    // 删除一个或全部attribute
table.getAttributes().clear();
```

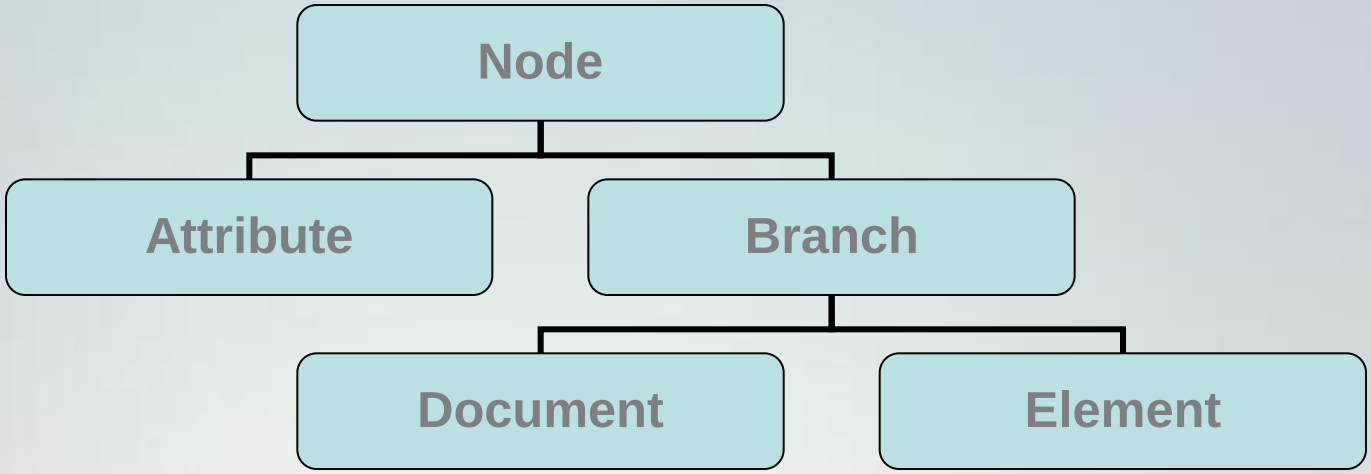
使用JDOM来生成以下xml文件并解析


```xml
<?xml version="1.0" encoding="UTF-8"?>
<students>
    <student id = "s01">
        <name>张三</name>
        <age>12</age>
    </student>
</students>
```

DOM4J简介

- DOM在java平台上的应用
- DOM4J是一套开源的类库。
- 与JDOM不同的是：DOM4J使用接口和抽象基类，它的API要复杂一些，但是灵活性更好

DOM4J的体系结构



Document：用于描述XML文档。

Element：用于描述XML文档中的元素。

Attribute：用于描述XML文档中的元素的属性。

DOM4J主要接口

No.	接口	描述
1	Attribute	Attribute定义了XML 的属性
2	Branch	Branch 为能够包含子节点的节点如XML 元素（Element）和文档（Docuemnt）定义了一个公共的行为
3	CDATA	CDATA 定义了XML CDATA 区域
4	CharacterData	CharacterData 是一个标识接口，标识基于字符的节点。如CDATA、Comment、Text
5	Comment	Comment 定义了XML的注释
6	Document	定义了XML 文档
7	Element	Element定义XML 元素
8	Text	Text 定义XML 文本节点

XML文档基本操作——Document

- 获取xml文档对象Document。

```
File xmlFile = new File( “test.xml” );

SAXReader reader = new SAXReader();

Document xmlDoc = reader.read(xmlFile);
```

XML文档基本操作——Element

- 获取根元素对象

```
Element root = xmlDoc.getRootElement();
```

- 获取根元素名称

```
String elementName = root.getName();
```

- 获取直接子元素对象

```
//通过元素名称获取元素对象
Element element = root.element("元素名称");
// 获取所有直接子元素对象集合
List elementList = root.elements();
```

- 获取元素对象中的文本

```
//获取element元素的内部文本
String text = element.getText();
//获取element子元素内部文本
String text = element.elementText("子元素名称" );
```

XML文档基本操作——Attribute

南京网博
专业专注 全心服务

- 获取元素属性对象

```
//通过属性名获取元素属性对象
    Attribute   attribute = element.attribute(“属性名称");
//通过索引获取属性对象
    Attribute  attribute = element.attribute(0);
//获取所有属性对象集合
    List  attributeList = root.attributes();
```

- 获取属性值

```
//通过属性对象
            String value = attribute.getValue();
//通过元素对象
    String  value = element.attributeValue(“属性名称”);
```

XML文档基本操作——创建

- 创建文档Document

```
Document document = DocumentHelper.createDocument();
```

- 创建元素

```
Element element = document.addElement(“元素名称");
```

- 创建属性

```
element.addAttribute(“属性名”,“属性值");
```

添加属性方法返回的是原元素对象。如同StringBuffer。

- 创建文本

```
element.addText(“文本");
```

XML文档基本操作——删除节点

南京网博
专业专注 全心服务

- remove(Element element)
- remove(Attribute attribute)
- remove(Text text)
- ……
- remove(Node node)

  Node接口的子类对象都可以移除

  注意：remove是移除直接子节点时使用

- parentElement.remove(childElement);

XML文档基本操作—建立xml文件

- 建立漂亮的xml文档，可以指定字符编码

```
OutputFormat format = OutputFormat.createPrettyPrint();
format.setEncoding( “GBK");
FileWriter fw = new FileWriter( “students2.xml” );
XMLWriter writer = new XMLWriter(fw);
writer.write(document);
writer.close();
```

小结：

- JDOM&DOM4J创建XML的步骤：
  1. 创建Document对象
  2. 为Document对象创建根元素
  3. 给根元素添加子元素，并设置子元素的内容
  4. 输出XML文档
     - 创建XML输出文档的对象
     - 设置文档的编码格式（可选）
     - 输出XML文档

JDOM&DOM4J读取XML的步骤：

南京网博
专业专注 全心服务

1.获取XML文档Document对象

2.获取根元素对象以及其子元素的文本内容

3.输出元素的文本内容