使用Dom4j解析XML - redarmychen的专栏 - CSDN博客

dom4j是一个Java的XML API，类似于jdom，用来读写XML文件的。dom4j是一个非常非常优秀的Java XML API，具有性能优异、功能强大和极端易用使用的特点，同时它也是一个开放源代码的软件，可以在SourceForge上找到它.

对主流的java XML API进行的性能、功能和易用性的评测，dom4j无论在那个方面都是非常出色的。如今你可以看到越来越多的Java软件都在使用dom4j来读写XML，例如hibernate，包括sun公司自己的JAXM也用了Dom4j。

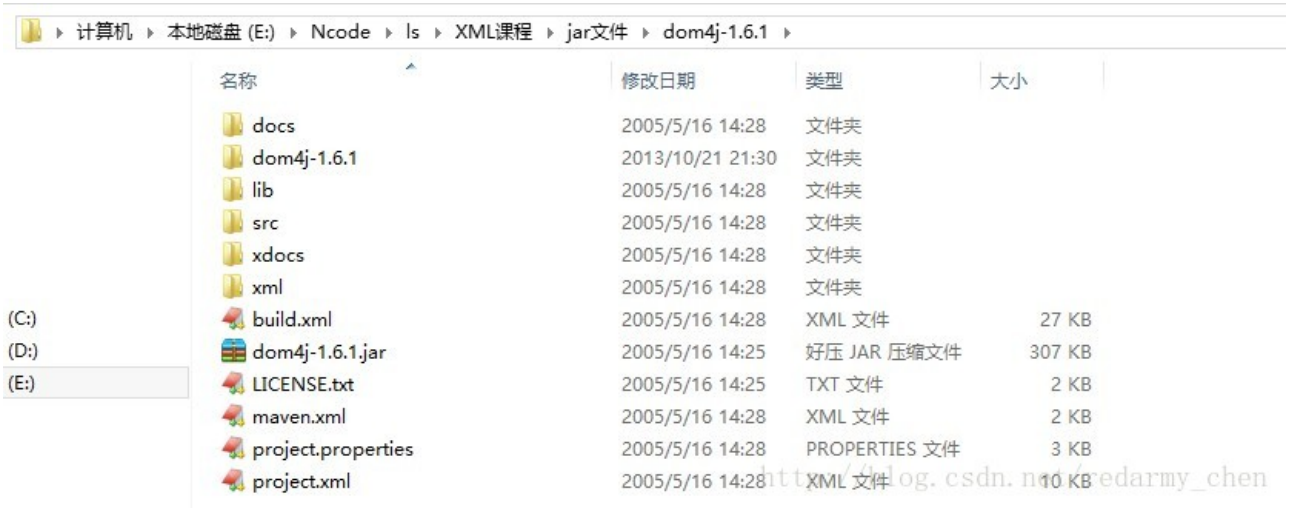
使用Dom4j开发，需下载dom4j相应的jar文件

1. 官网下载:http://www.dom4j.org/dom4j-1.6.1/

2. dom4j是sourceforge.NET上的一个开源项目,因此可以到http://sourceforge.net/projects/dom4j下载其最新版.

对于下载的zip文件进行解压后的效果如下:

打开dom4j-1.6.1的解压文件



在这里可以看到有docs帮助的文件夹，也有需要使用dom4j解析xml文件的dom4j-1.6.1.jar文件.我们只需要把dom4j-1.6.1.jar文件构建到我们开发的项目中就可以使用dom4j开发了.

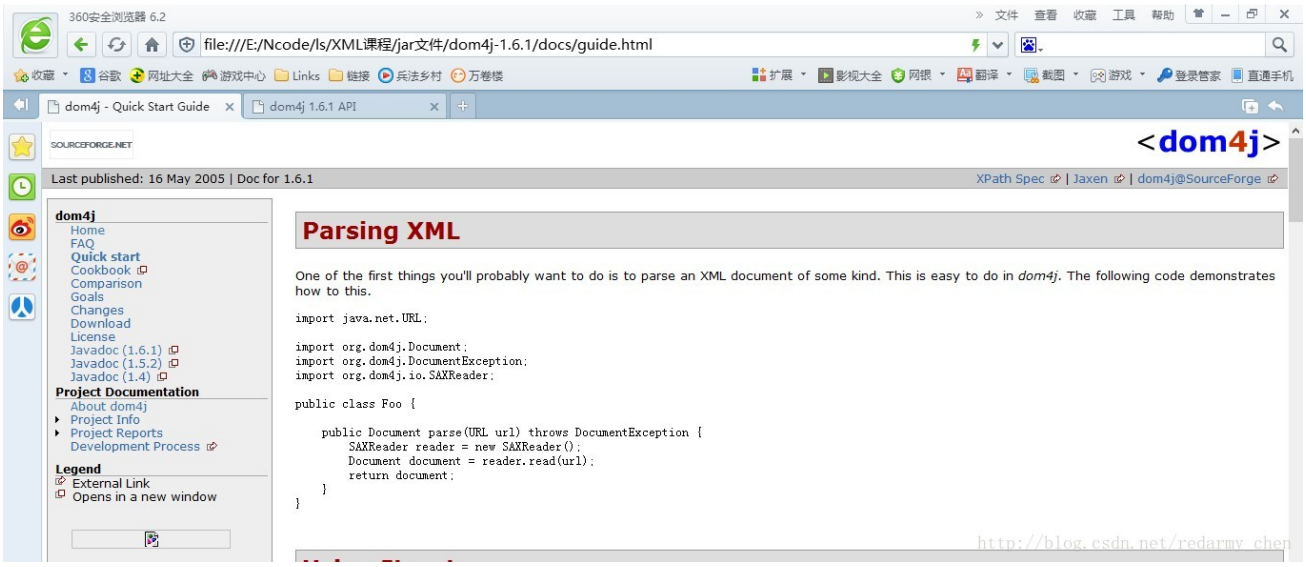
下面我以Myeclipse创建Java项目的构建方法为例说明.

首先创建一个demo项目，在demo项目中创建一个lib文件，把dom4j-1.6.1.jar文件拷贝到lib中，然后右键dom4j-1.6.1jar文件

点击Add to Build Path即可构建到项目中去了.

备注：如果进行的是web项目开发，我们只需要把它拷贝到web-inf/lib中去即可,会自动构建到web项目中.

在项目开发的过程中可以参考docs文件夹的（帮助文档）,找到index.html打开，点击Quick start可以通过帮助文档进行学习 dom4j进行xml的解析.



下面我对我认为api中重要的方法进行翻译说明如下：

一、DOM4j中，获得Document对象的方式有三种：

[java]view plaincopy

```
1. 1.读取XML文件,获得document对象
2.                  SAXReader reader = new SAXReader();
3.        Document    document = reader.read(new File("csdn.xml"));
4. 2.解析XML形式的文本,得到document对象.
5.                  String text = "";
6.                  Document document = DocumentHelper.parseText(text);
7. 3.主动创建document对象.
8.                 Document document = DocumentHelper.createDocument();              //创建根节点
9.                 Element root = document.addElement("csdn");
```

二、节点对象操作的方法

[java]view plaincopy

```
1. 1.获取文档的根节点.
2.       Element root = document.getRootElement();
3. 2.取得某个节点的子节点.
4.       Element element=node.element("四大名著");
5. 3.取得节点的文字
6.         String text=node.getText();
7. 4.取得某节点下所有名为"csdn"的子节点，并进行遍历.
8.        List nodes = rootElm.elements("csdn");
9. for (Iterator it = nodes.iterator(); it.hasNext();) {
10.       Element elm = (Element) it.next();
11. // do something
12.  }
13. 5.对某节点下的所有子节点进行遍历.
14. for(Iterator it=root.elementIterator();it.hasNext();){
15.         Element element = (Element) it.next();
16. // do something
17.  }
18. 6.在某节点下添加子节点
19.       Element elm = newElm.addElement("朝代");
20. 7.设置节点文字.  elm.setText("明朝");
21. 8.删除某节点.//childElement是待删除的节点,parentElement是其父节点
parentElement.remove(childElment);
22. 9.添加一个CDATA节点.Element contentElm = infoElm.addElement("content");contentElm.addCDATA("cdata
```

区域”);

三、节点对象的属性方法操作

[java]view plaincopy

```
1. 1.取得某节点下的某属性     Element root=document.getRootElement();        //属性名name
2.          Attribute attribute=root.attribute("id");
3. 2.取得属性的文字
4.     String text=attribute.getText();
5. 3.删除某属性 Attribute attribute=root.attribute("size"); root.remove(attribute);
6. 4.遍历某节点的所有属性
7.       Element root=document.getRootElement();
8. for(Iterator it=root.attributeIterator();it.hasNext();){
9.            Attribute attribute = (Attribute) it.next();
10.            String text=attribute.getText();
11.            System.out.println(text);
12.   }
13. 5.设置某节点的属性和文字.   newMemberElm.addAttribute("name", "sitinspring");
14. 6.设置属性的文字   Attribute attribute=root.attribute("name");   attribute.setText("csdn");
```

四、将文档写入XML文件

[java]view plaincopy

```
1. 1.文档中全为英文,不设置编码,直接写入的形式.
2.        XMLWriter writer = new XMLWriter(new  FileWriter("ot.xml"));
3.        writer.write(document);
4.        writer.close();
5. 2.文档中含有中文,设置编码格式写入的形式.
6.        OutputFormat format = OutputFormat.createPrettyPrint();// 创建文件输出的时候，自动缩进的格式
7.        format.setEncoding("UTF-8");//设置编码
8.        XMLWriter writer = new XMLWriter(newFileWriter("output.xml"),format);
9.        writer.write(document);
10.        writer.close();
```

五、字符串与XML的转换

[java]view plaincopy

```
1. 1.将字符串转化为XML
2.       String text = " Java班";
3.       Document document = DocumentHelper.parseText(text);
4. 2.将文档或节点的XML转化为字符串.
5.        SAXReader reader = new SAXReader();
6.        Document   document = reader.read(new File("csdn.xml"));
7.        Element root=document.getRootElement();
8.        String docXmlText=document.asXML();
9.        String rootXmlText=root.asXML();
10.        Element memberElm=root.element("csdn");
11.        String memberXmlText=memberElm.asXML();
```

六、案例(解析sida.xml文件并对其进行curd的操作)

1.sida.xml描述四大名著的操作，文件内容如下

[html]view plaincopy

```
1. xmlversion="1.0"encoding="UTF-8"?>
2. <四大名著>
3. <西游记 id="x001">
4. <作者>吴承恩1作者>
```

```
5. <作者>吴承恩2作者>
6. <朝代>明朝朝代>
7. 西游记>
8. <红楼梦 id="x002">
9. <作者>曹雪芹作者>
10. 红楼梦>
11. 四大名著>
```

2. 解析类测试操作

[java]view plaincopy

```
1. package dom4j;
2. import java.io.File;
3. import java.io.FileOutputStream;
4. import java.io.FileWriter;
5. import java.io.OutputStreamWriter;
6. import java.nio.charset.Charset;
7. import java.nio.charset.CharsetEncoder;
8. import java.util.Iterator;
9. import java.util.List;
10. import org.dom4j.Attribute;
11. import org.dom4j.Document;
12. import org.dom4j.Element;
13. import org.dom4j.io.OutputFormat;
14. import org.dom4j.io.SAXReader;
15. import org.dom4j.io.XMLWriter;
16. import org.junit.Test;
17. publicclass Demo01 {
18. @Test
19. publicvoid test() throws Exception {
20. // 创建saxReader对象
21.         SAXReader reader = new SAXReader();
22. // 通过read方法读取一个文件 转换成Document对象
23.         Document document = reader.read(new File("src/dom4j/sida.xml"));
24. //获取根节点元素对象
25.         Element node = document.getRootElement();
26. //遍历所有的元素节点
27.         listNodes(node);
28. // 获取四大名著元素节点中，子节点名称为红楼梦元素节点。
29.         Element element = node.element("红楼梦");
30. //获取element的id属性节点对象
31.         Attribute attr = element.attribute("id");
32. //删除属性
33.         element.remove(attr);
34. //添加新的属性
35.         element.addAttribute("name", "作者");
36. // 在红楼梦元素节点中添加朝代元素的节点
37.         Element newElement = element.addElement("朝代");
38.         newElement.setText("清朝");
39. //获取element中的作者元素节点对象
40.         Element author = element.element("作者");
41. //删除元素节点
42. boolean flag = element.remove(author);
43. //返回true代码删除成功，否则失败
44.         System.out.println(flag);
45. //添加CDATA区域
46.         element.addCDATA("红楼梦，是一部爱情小说.");
```

```
47. // 写入到一个新的文件中
48.         writer(document);
49.     }
50. /**
51.      * 把document对象写入新的文件
52.      *
53.      * @param document
54.      * @throws Exception
55.      */
56. publicvoid writer(Document document) throws Exception {
57. // 紧凑的格式
58. // OutputFormat format = OutputFormat.createCompactFormat();
59. // 排版缩进的格式
60.         OutputFormat format = OutputFormat.createPrettyPrint();
61. // 设置编码
62.         format.setEncoding("UTF-8");
63. // 创建XMLWriter对象,指定了写出文件及编码格式
64. // XMLWriter writer = new XMLWriter(new FileWriter(new
65. // File("src//a.xml")),format);
66.         XMLWriter writer = new XMLWriter(new OutputStreamWriter(
67. new FileOutputStream(new File("src//a.xml")), "UTF-8"), format);
68. // 写入
69.         writer.write(document);
70. // 立即写入
71.         writer.flush();
72. // 关闭操作
73.         writer.close();
74.     }
75. /**
76.      * 遍历当前节点元素下面的所有(元素的)子节点
77.      *
78.      * @param node
79.      */
80. publicvoid listNodes(Element node) {
81.         System.out.println("当前节点的名称：: " + node.getName());
82. // 获取当前节点的所有属性节点
83.         List list = node.attributes();
84. // 遍历属性节点
85. for (Attribute attr : list) {
86.             System.out.println(attr.getText() + "-----" + attr.getName()
87.                     + "---" + attr.getValue());
88.         }
89. if (!(node.getTextTrim().equals(""))) {
90.             System.out.println("文本内容：: : : " + node.getText());
91.         }
92. // 当前节点下面子节点迭代器
93.         Iterator
94. // 遍历
95. while (it.hasNext()) {
96. // 获取某个子节点对象
97.             Element e = it.next();
98. // 对子节点进行遍历
99.             listNodes(e);
100.         }
101.     }
102. /**
```

```
103.      * 介绍Element中的element方法和elements方法的使用
104.      *
105.      * @param node
106.      */
107. publicvoid elementMethod(Element node) {
108. // 获取node节点中，子节点的元素名称为西游记的元素节点。
109.         Element e = node.element("西游记");
110. // 获取西游记元素节点中，子节点为作者的元素节点(可以看到只能获取第一个作者元素节点)
111.         Element author = e.element("作者");
112.         System.out.println(e.getName() + "----" + author.getText());
113. // 获取西游记这个元素节点 中，所有子节点名称为作者元素的节点 。
114.         List
115. for (Element aut : authors) {
116.             System.out.println(aut.getText());
117.         }
118. // 获取西游记这个元素节点 所有元素的子节点。
119.         List
120. for (Element el : elements) {
121.             System.out.println(el.getText());
122.         }
123.     }
124. }
```

自己适当注释部分代码观察运行效果,反复练习，希望你对dom4j有进一步的了解.

七、字符串与XML互转换案例

[java]view plaincopy

```
1. package dom4j;
2. import java.io.File;
3. import java.io.FileOutputStream;
4. import java.io.OutputStreamWriter;
5. import org.dom4j.Document;
6. import org.dom4j.DocumentHelper;
7. import org.dom4j.Element;
8. import org.dom4j.io.OutputFormat;
9. import org.dom4j.io.SAXReader;
10. import org.dom4j.io.XMLWriter;
11. import org.junit.Test;
12. publicclass Demo02 {
13. @Test
14. publicvoid test() throws Exception {
15. // 创建saxreader对象
16.         SAXReader reader = new SAXReader();
17. // 读取一个文件，把这个文件转换成Document对象
18.         Document document = reader.read(new File("src//c.xml"));
19. // 获取根元素
20.         Element root = document.getRootElement();
21. // 把文档转换字符串
22.         String docXmlText = document.asXML();
23.         System.out.println(docXmlText);
24.         System.out.println("----------------------------");
25. // csdn元素标签根转换的内容
26.         String rootXmlText = root.asXML();
27.         System.out.println(rootXmlText);
```

```
28.         System.out.println("----------------------------");
29. // 获取java元素标签 内的内容
30.         Element e = root.element("java");
31.         System.out.println(e.asXML());
32.     }
33. /**
34.      * 创建一个document对象 往document对象中添加节点元素 转存为xml文件
35.      *
36.      * @throws Exception
37.      */
38. publicvoid test2() throws Exception {
39.         Document document = DocumentHelper.createDocument();// 创建根节点
40.         Element root = document.addElement("csdn");
41.         Element java = root.addElement("java");
42.         java.setText("java班");
43.         Element ios = root.addElement("ios");
44.         ios.setText("ios班");
45.         writer(document);
46.     }
47. /**
48.      * 把一个文本字符串转换Document对象
49.      *
50.      * @throws Exception
51.      */
52. publicvoid test1() throws Exception {
53.         String text = "Java班Net班";
54.         Document document = DocumentHelper.parseText(text);
55.         Element e = document.getRootElement();
56.         System.out.println(e.getName());
57.         writer(document);
58.     }
59. /**
60.      * 把document对象写入新的文件
61.      *
62.      * @param document
63.      * @throws Exception
64.      */
65. publicvoid writer(Document document) throws Exception {
66. // 紧凑的格式
67. // OutputFormat format = OutputFormat.createCompactFormat();
68. // 排版缩进的格式
69.         OutputFormat format = OutputFormat.createPrettyPrint();
70. // 设置编码
71.         format.setEncoding("UTF-8");
72. // 创建XMLWriter对象,指定了写出文件及编码格式
73. // XMLWriter writer = new XMLWriter(new FileWriter(new
74. // File("src//a.xml")),format);
75.         XMLWriter writer = new XMLWriter(new OutputStreamWriter(
76. new FileOutputStream(new File("src//c.xml")), "UTF-8"), format);
77. // 写入
78.         writer.write(document);
79. // 立即写入
80.         writer.flush();
81. // 关闭操作
82.         writer.close();
83.     }
```

```
84. }
```