XML文件的基本操作

1. 解析XML的方法越来越多，但主流的方法就四种，即：DOM、SAX、JDOM和DOM4J。
2. // 1. 获取dom4j中读取xml文档的对象。

SAXReader reader = **new** SAXReader();

// 2. 通过读取对象reader中的read方法与要操作的xml文档相关联，从而获取xml文档的document对象。

File file = **new** File("stu1.xml");

Document document = reader.read(file);

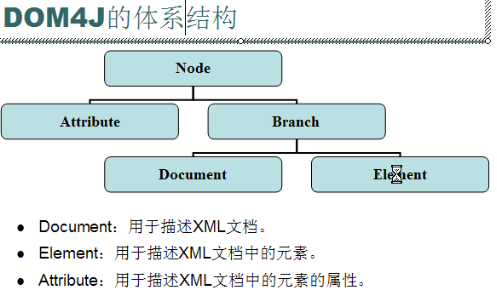
// 3. 通过xml文档的document对象获取根元素节点对象。

Element rootElement = document.getRootElement();

// 4. 通过根元素对象就可以对文档内容进行操作。

System.***out***.println(rootElement.getName());

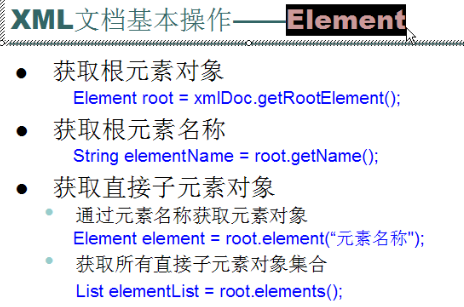
1. Element接口：
2. DOM4J的体系结构：

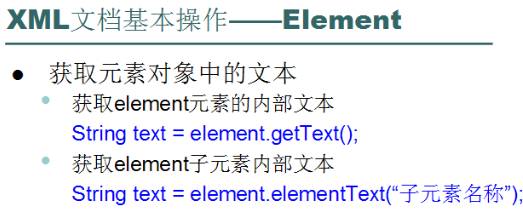


1. 把所有的对象都继承于Node节点，每个不同的对象都有一个节点类型，每个节点类型都用一个short值表示。

其中根节点rootElement为1，Attribute为2，Document为9.

1. XML文档基本操作：
2. 获取元素Element对象：

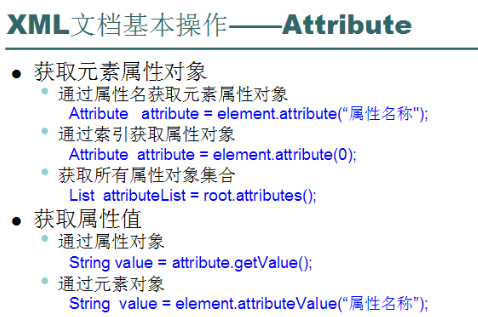




**获取元素对象中的文本方法：**

* 1. **首先获取元素对象，然后通过getText方法获取；**
  2. **直接利用父元素的elementText(“元素名称”)方法获取。**

1. **获取元素属性对象：（Attribute）**



**获取属性值**有两种方法：（类似于获取元素内容一样）

**（1）直接通过元素对象获取**

System.out.println(userElement.attributeValue("id"));

等同于

**（2） 现获取属性对象，然后再获取属性值。**

Attribute attri = userElement.attribute(0);

System.out.println( attri.getValue());

**获取属性个数**：

System.out.println("属性个数"+userElement.attributeCount());

1. **增加节点**：



**Element rootElement = document.getRootElement();**

**Element userElement = rootElement.addElement("user");**

**userElement.addAttribute("id","9").addElement("age").addText("88");**

**userElement.addElement("sex").addText("Male");**

**userElement.addElement("name").addText("zhaoliu");**

1. **方法的调用链**：

例如StringBuilder sb = new StringBuilder();

sb.append("zhangsan").append("lisi").append("wangwu");

因为**sb的append方法返回的仍然是此对象本身**；所以可以直接调用其他方法，实现了连续的调用，所以称为**方法的调用链**。

1. 输出内存中的**DOM树**的结构的方法：

**利用Node的asXML方法**。

1. 如果是Document，则输出的是**整个DOM的树结构**;
2. 如果是元素Element，则输出**此元素即子元素的树结构**;
3. 如果是Attribute属性，则输出**此属性的树结构**。
4. **删除节点**：**父节点删除直接子节点**。（remove方法）

**示例：**

Element rootElement = document.getRootElement();

Element userElement = (Element) rootElement.elements().get(4);

Attribute att = userElement.attribute(0);

userElement.remove(att);

rootElement.remove(userElement);

1. 练习题：**根据Id删除某个元素**。

**private** **static** **void** removeById(Document document, **int** id) {

Element rootElement = document.getRootElement();

List<Element> list = rootElement.elements();

Element userElement = **null**;

Iterator<Element> it = **null**;

**for** (it = list.iterator(); it.hasNext();) {

userElement = it.next();

**if** (id == Integer.*parseInt*(userElement.attribute(0).getValue())) {

System.***out***.println("找到了 Id = "+id+"的Element，并且删除了此Element。");

**break**;

}

userElement = **null**;

}

**if** (userElement != **null**) {

rootElement.remove(userElement);

}**else** {

System.***out***.println("没有找到相应的Id的Element。");

}

}