

DOMTest.java

**package** com.imooc.dmtest.test;

**import** java.io.IOException;

**import** javax.xml.parsers.DocumentBuilder;

**import** javax.xml.parsers.DocumentBuilderFactory;

**import** javax.xml.parsers.ParserConfigurationException;

**import** org.w3c.dom.Document;

**import** org.w3c.dom.Element;

**import** org.w3c.dom.NamedNodeMap;

**import** org.w3c.dom.Node;

**import** org.w3c.dom.NodeList;

**import** org.xml.sax.SAXException;

**public** **class** DOMTest {

**public** **static** **void** main(String[] args) {

//创建一个DOcumentBuilderFactory的对象

DocumentBuilderFactory dbf= DocumentBuilderFactory.*newInstance*();

//创建一个DocumentBuilder的对象

**try** {

//创建DocumentBuilder对象

DocumentBuilder db= dbf.newDocumentBuilder();

//通过DocumentBuilder对象的parser方法加载books.xml文件到当前项目下

Document document = db.parse("books.xml");

//获取所有book节点的集合

NodeList booklist = document.getElementsByTagName("book");

//通过nodelist的getLength（）方法可以获取bookList的长度

System.***out***.println("一共有" + booklist.getLength() + "本书");

//遍历每一个book节点

**for**(**int** i=0;i < booklist.getLength();i++){

System.***out***.println("===下面开始遍历第"+(i+1)+"本书的内容===");

//通过item(i)方法获取一个book节点，nodelist的索引值从0开始

Node book = booklist.item(i);

//获取book节点的所有属性集合

NamedNodeMap attrs = book.getAttributes();

System.***out***.println("第"+(i+1)+"本书共有"+attrs.getLength()+"个属性");

//遍历book的属性

**for**(**int** j=0;j<attrs.getLength();j++){

//通过item(index)方法获取book节点的某一个属性

Node attr = attrs.item(j);

//获取属性名

System.***out***.print("属性名；"+attr.getNodeName());

//获取属性值

System.***out***.println("--属性值："+attr.getNodeValue());

}

/\* //前提：已经知道book节点有且只能有一个id属性

//将book节点进行强制类型转换，转换成Element类型

Element book =(Element) booklist.item(i);

//通过getAttribute("id")方法获取属性值

String attrValue = book.getAttribute("id");

System.out.println("id属性的属性值为"+attrValue);\*/

//解析book节点的子节点

NodeList childNodes = book.getChildNodes();

//遍历childNodes获取每个节点的节点名和节点值

System.***out***.println("第"+(i+1)+"本书共有"+childNodes.getLength()+"个子节点");

**for**(**int** k=0;k<childNodes.getLength();k++){

//区分text类型的node以及element类型的node

**if**(childNodes.item(k).getNodeType()==Node.***ELEMENT\_NODE***){

//获取了element类型节点的节点名

System.***out***.print("第"+(k+1)+"个节点的节点名："+childNodes.item(k).getNodeName());

//获取了element类型节点的节点值

System.***out***.println("--节点值是："+childNodes.item(k).getFirstChild().getNodeValue());

}

}

System.***out***.println("===结束遍历第"+(i+1)+"本书的内容===");

}

} **catch** (ParserConfigurationException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}**catch** (SAXException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

以下是运行效果逐步截图

















