SAX解析常用API

1、

SAXParserFactory factory = SAXParserFactory.newInstance();

Parser.parse(“books.xml”, handler);

2、

//handler参数是Default**Handler**对象

DefaultHandler handler = new SAXParserHandler();

3、

//Default**Handler**最重要的几个回调方法

**startDocument**和**endDocument**分别在文档开始和结束时各调用一次

**startElement**和**endElement**每当遇到起始或终止**标签**时调用

**characters**在每当遇到**标签中间的**字符数据时调用（非换行符的Text）

4、

//Default**Handler**实现类

import java.util.ArrayList;

import org.xml.sax.Attributes;

import org.xml.sax.SAXException;

import org.xml.sax.helpers.DefaultHandler;

import com.imooc.entity.Book;

public class SAXParserHandler extends DefaultHandler {

String value = null;

Book book = null;

private ArrayList<Book> bookList = new ArrayList<Book>();

public ArrayList<Book> getBookList() {

return bookList;

}

int bookIndex = 0;

/\*\*

\* 用来标识解析开始

\*/

@Override

public void **startDocument()** throws SAXException {

// TODO Auto-generated method stub

super.startDocument();

System.out.println("SAX解析开始");

}

/\*\*

\* 用来标识解析结束

\*/

@Override

public void **endDocument()** throws SAXException {

// TODO Auto-generated method stub

super.endDocument();

System.out.println("SAX解析结束");

}

/\*\*

\* 解析xml元素

\*/

@Override

public void **startElement**(String uri, String localName, String qName,

Attributes attributes) throws SAXException {

//调用DefaultHandler类的startElement方法

super.startElement(uri, localName, qName, attributes);

if (qName.equals("book")) {

bookIndex++;

//创建一个book对象

book = new Book();

//开始解析book元素的属性

System.out.println("======================开始遍历某一本书的内容=================");

// //已知book元素下属性的名称，根据属性名称获取属性值

// String value = attributes.getValue("id");

// System.out.println("book的属性值是：" + value);

//不知道book元素下属性的名称以及个数，如何获取属性名以及属性值

int num = attributes.getLength();

for(int i = 0; i < num; i++){

System.out.print("book元素的第" + (i + 1) + "个属性名是："

+ attributes.getQName(i));

System.out.println("---属性值是：" + attributes.getValue(i));

if (attributes.getQName(i).equals("id")) {

book.setId(attributes.getValue(i));

}

}

}

else if (!qName.equals("name") && !qName.equals("bookstore")) {

System.out.print("节点名是：" + qName + "---");

}

}

@Override

public void **endElement**(String uri, String localName, String qName)

throws SAXException {

//调用DefaultHandler类的endElement方法

super.endElement(uri, localName, qName);

//判断是否针对一本书已经遍历结束

if (qName.equals("book")) {

bookList.add(book);

book = null;

System.out.println("======================结束遍历某一本书的内容=================");

}

else if (qName.equals("name")) {

book.setName(value);

}

else if (qName.equals("author")) {

book.setAuthor(value);

}

else if (qName.equals("year")) {

book.setYear(value);

}

else if (qName.equals("price")) {

book.setPrice(value);

}

else if (qName.equals("language")) {

book.setLanguage(value);

}

}

@Override

public void **characters**(char[] ch, int start, int length)

throws SAXException {

// TODO Auto-generated method stub

super.characters(ch, start, length);

value = new String(ch, start, length);

if (!value.trim().equals("")) {

System.out.println("节点值是：" + value);

}

}

}