网络爬虫作业报告

林涛 3130000064

# 作业要求

跟踪特定网页，下载该网页中所有链接的指定内容，去除广告等无关内容，组合成单一文件。主要作广度搜索，深度暂为1。

要求：

1、(基本)不使用第三方工具，如：HttpClient、HtmlParser等，自己作String处理。

2、(提高)可用第三方工具。

3、(可选)自己做一个第三方工具，再爬。

Java参考：<http://www.cjsdn.net/Doc/JDK50/index-files/index-1.html>

例：从 http://ds.eywedu.com/jinyong/tlbb/ 下载所有章节，组成完整《天龙八部》.txt。

例：下载某一网页中的所有.jpg图片，或.mp3、flash等。

# 自己作String处理

## 网页特征分析

想要抓取的小说是《书剑恩仇录》和《雪山飞狐》，选取的网页是梦远书城的金庸作品集（<http://www.my285.com/wuxia/jinyong/sjecl/> 和<http://www.my285.com/wuxia/jinyong/xsfh/>）。分成目录页和内容页分别抓取。

### 目录页特征

目录中的链接格式如下所示：

这是《雪山飞狐》的——

<tr>

<td bgcolor="#FFFFFF" align="center"><a href="05.htm">第05页</a></td>

<td bgcolor="#FFFFFF" align="center"><a href="06.htm">第06页</a></td>

<td bgcolor="#FFFFFF" align="center"><a href="07.htm">第07页</a></td>

<td bgcolor="#FFFFFF" align="center"><a href="08.htm">第08页</a></td>

</tr>

这是《书剑恩仇录》的——

<tr>

<td><a href="014.htm">第三回 避祸英雄悲失路 寻仇好汉误交兵</a> <a href="015.htm">(2)</a> <a href="016.htm">(3)</a> <a href="017.htm">(4)</a> <a href="018.htm">(5)</a> <a href="019.htm">(6)</a> <a href="020.htm">(7)</a> <a href="021.htm">(8)</a></td>

</tr>

<tr>

<td><a href="022.htm">第四回 置酒弄丸招薄怒 还书贻剑种深情</a> <a href="023.htm">(2)</a> <a href="024.htm">(3)</a> <a href="025.htm">(4)</a> <a href="026.htm">(5)</a> <a href="027.htm">(6)</a> <a href="028.htm">(7)</a> <a href="029.htm">(8)</a></td>

</tr>

每个链接都是以“<a href=”\*\*.htm”>”的形式出现的，其中\*\*是一个数字，网页中有一些链接不指向我们需要的内容页，但也有着上面的形式，可以通过判断\*\*是否为数字排除掉。

每一个标签对应章节的标题可以用起始的汉字提取。因为经过观察可以发现每一个标题都开始于“第”和“后”，分别指“第某回（页）”或“后记”。

### 内容页特征

内容页更为简单，以下是删节的部分内容页：

<td colspan="2">

<br>

　　清乾隆十八年六月，陕西扶风延绥镇总兵衙门内院，一个十四岁的女孩儿跳跳蹦蹦的走向教书先生书房。上午老师讲完了《资治通鉴》上“赤壁之战”的一段书，随口讲了些诸葛亮、周瑜的故事。午后本来没功课，那女孩儿却兴犹未尽，要老师再讲三国故事。这日炎阳盛暑，四下里静悄悄地，更没一丝凉风。那女孩儿来到书房之外，怕老师午睡未醒，进去不便，于是轻手轻脚绕到窗外，拔下头上金钗，在窗纸上刺了个小孔，凑眼过去张望。只见老师盘膝坐在椅上，脸露微笑，右手向空中微微一扬，轻轻吧的一声，好似甚么东西在板壁上一碰。她向声音来处望去，只见对面板壁上伏着几十只苍蝇，一动不动，她十分奇怪，凝神注视，却见每只苍蝇背上都插着一根细如头发的金针。这针极细，隔了这样远原是难以辨认，只因时交未刻，日光微斜，射进窗户，金针在阳光下生出了反光。<br>

<br>

　　陆菲青道：“三更半夜之际，竟劳动三位过访，真是想不到。却不知有何见教？”……陆菲青的招术则似慢实快。一瞬之间两人已拆了十多招。以罗信的武功，怎能与他拆到十招以上？只因陆菲青近年来深自收敛，知道罗信这些人只是贪图功名利禄，天下滔滔，实是杀不胜杀，是以出手之际，颇加容让。<br>

<br>

</td>

全部的正文内容都被包括在<td colspan=”2”>和</td>之间。每段文字的结尾有个<br>，每段文字之间有个<br>。

## 抓取方法

使用Java自带的net包中的网络功能，创建URL对象，用BufferedReader读取。输出到本地文件夹，用PrintWriter连接FileOutputStream实现。

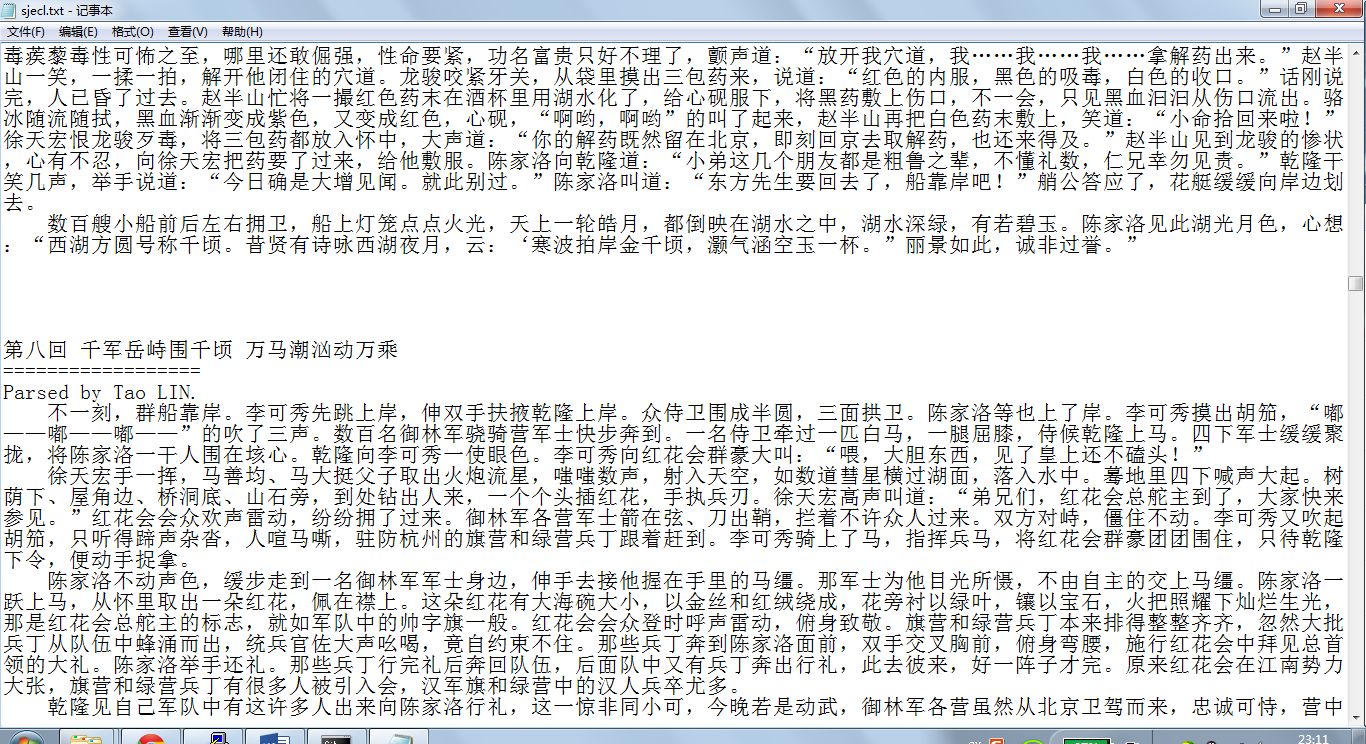
所有代码在Spider类中，除了main函数外，有parseIndex和parsePage分别处理目录页和内容页。

具体的策略与上面所述相同，需要灵活使用Java的String类的函数，如indexOf，substring，charAt，contains等。

输出时在每章开头加上章节名称和我自己的标识符。在处理每个内容页标签是在console输出提示以便于观察。

## 成果

通过改变main函数中的bookname，分别输出 “sjecl.txt”（《书剑恩仇录》）和 “xsfh.txt”（《雪山飞狐》）。部分结果如下图所示：



图表 1 《书剑恩仇录》爬虫结果

经肉眼观察，结果完全纯净，没有引入不需要的内容。

## Source Code

**package** spider;

**import** **java.net.\***;

**import** **java.io.\***;

**public** **class** **Spider** {

**public** **static** void main(String argv[]) **throws** Exception {

String bookname = "sjecl"; *// 书剑恩仇录*

*// String bookname = "xsfh"; // 雪山飞狐*

PrintWriter outputStream = **new** PrintWriter(**new** FileOutputStream(bookname + ".txt"));

parseIndex(**new** URL("http://www.my285.com/wuxia/jinyong/"+bookname+"/index.htm"), outputStream);

outputStream.close();

System.out.println("Finished.");

}

*/\**

*\* Parse the index page of sourceURL and the output is to outputStream*

*\*/*

**public** **static** void parseIndex(URL sourceURL, PrintWriter outputStream) **throws** Exception {

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(sourceURL.openStream()));

String buf;

**while**(!(**null**==(buf=in.readLine()))){

int cursor = 0;

*// It is a start if there is a hypertext reference.*

cursor = buf.indexOf("<a href=",cursor);

*// Get the title of the chapter,*

*// the titles begin with "第"("第X回") or "后"("后记")*

int startTitle = buf.indexOf("第");

**if**(startTitle == -1)

startTitle = buf.indexOf("后");

**if**(startTitle == -1)

**continue**;

int endTitle = buf.indexOf("</a>");

**if**(endTitle <= startTitle)

**continue**;

*// The title of the chapter*

String title = buf.substring(startTitle, endTitle);

*// Message on console*

System.out.println("Start " + title);

*// My special format for the beginning of each chapter*

outputStream.println("\r\n\r\n\r\n");

outputStream.println(title);

outputStream.println("==================");

outputStream.println("Parsed by Tao LIN.");

*// Find out each hypertext reference, and parse it.*

**while**(cursor != -1){

int left = buf.indexOf('"',cursor) + 1;

int right = buf.indexOf('"',left);

String href = buf.substring(left, right);

*// Check the href is what we need*

*// (it should begin with a digit)*

**if**(Character.isDigit(href.charAt(0)) == **true**){

*// Get the new URLs and parse each other using parsePage()*

URL subURL = **new** URL(sourceURL, href);

parsePage(subURL, outputStream);

}

cursor = right;

cursor = buf.indexOf("<a href=",cursor);

}

}

in.close();

}

*/\**

*\* Get the context in each page*

*\*/*

**public** **static** void parsePage(URL pageURL, PrintWriter outputStream) **throws** Exception {

*// System.out.println(pageURL.toString());*

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(pageURL.openStream()));

String buf;

boolean start = **false**;

**while**(**null** != (buf=in.readLine())){

*// The passage in contained between <td colspan="2"> and </td>*

*// Each paragraph is surrounded by <br>*

**if**(start){

**if**(buf.contains("</td>"))

**break**;

int indexEnd = buf.indexOf("<br>");

**if**(indexEnd > 0){

String para = buf.substring(0,indexEnd);

outputStream.println(para);

}

}

**else** **if**(buf.contains("<td colspan=\"2\">")){

start = **true**;

}

}

in.close();

}

**public** **static** void dispAll (URL sourceURL) **throws** Exception {

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(sourceURL.openStream()));

String buf;

**while**(!(**null**==(buf=in.readLine()))){

System.out.println(buf);

}

}

}

# 使用HttpClient

## 使用经过

HttpClient提供了访问Web和很多函数。按照新版的API，可以通过HttpClients.createDefault()创建一个CloseableHttpClient。然后让这个Client去execute一个HttpGet命令，返回一个CloseableHttpResponse，取得其entity。这个entity只要getContent就是熟悉的InputStream了。

运行能抓取整个网页。

后面本来该写个爬虫的，但其余方法与前面没有太大区别，而下面使用HtmlParser也不需要用到这个，所以只作为一个独立的实验放在这里了。

## Source Code

**package** spider;

**import** **java.io.BufferedReader**;

**import** **java.io.IOException**;

**import** **java.io.InputStream**;

**import** **java.io.InputStreamReader**;

**import** **org.apache.http.HttpEntity**;

**import** **org.apache.http.client.ClientProtocolException**;

**import** **org.apache.http.client.methods.CloseableHttpResponse**;

**import** **org.apache.http.client.methods.HttpGet**;

**import** **org.apache.http.impl.client.CloseableHttpClient**;

**import** **org.apache.http.impl.client.HttpClients**;

**public** **class** **SpiderClient**{

**public** **static** void main(String[] args) **throws** ClientProtocolException, IOException{

CloseableHttpClient httpclient = HttpClients.createDefault();

HttpGet httpget = **new** HttpGet("http://www.my285.com/wuxia/jinyong/xsfh");

CloseableHttpResponse response = httpclient.execute(httpget);

**try** {

HttpEntity entity = response.getEntity();

InputStream inputstream = entity.getContent();

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(inputstream));

String buf;

**while**(!(**null**==(buf=in.readLine()))){

System.out.println(buf);

}

httpget.abort();

} **finally** {

response.close();

}

}

}

# 使用HtmlParser

## 使用经过

给Parser传入url，调用其parse函数，获得一个nodelist。用Nodelist的elements函数可以得到子节点的迭代器，用迭代器遍历，可以用来迭代直到找到需要的元素。

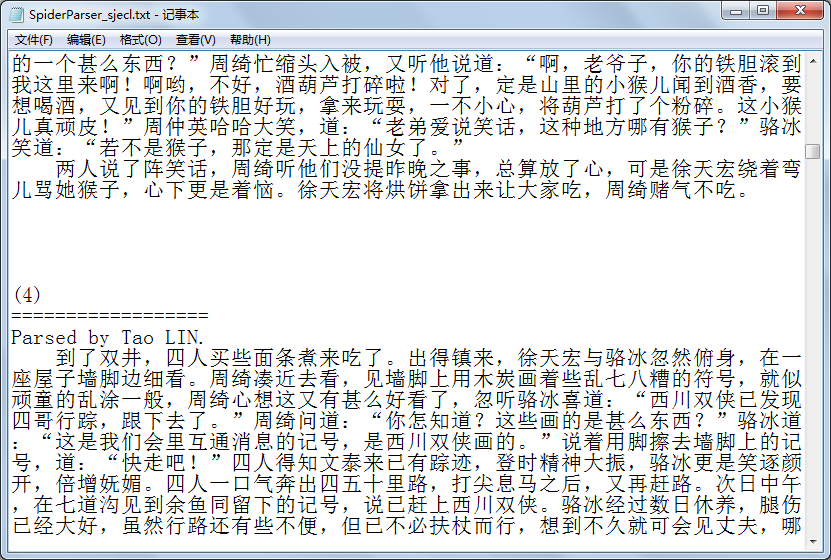
为了配合Parser的特性，我找了目录页的另一个特性——所有需要的链接都是放在一个表格中的。鉴于目录页中并不是只有一个表格，可以用表格的行数来判断是否是我们需要的表格，当行数超过15行时，认为是我们需要的含链接的表格。

将表格节点转化为TableTag对象，用其自带的getColumns和getRows得到每一格的内容，得到LinkTag对象。用toPlainTextString函数得到标题，用getLink得到链接。

对于内容页的抓取还是采用原来的方式。

## 成果

通过改变main函数中的bookname，分别输出 “SpiderParser\_sjecl.txt”（《书剑恩仇录》）和 “SpiderParser\_xsfh.txt”（《雪山飞狐》）。部分结果如下图所示：



图表 2 《书剑恩仇录》爬虫结果（SpiderParser版）

## Source Code

**package** spider;

**import** **java.io.BufferedReader**;

**import** **java.io.FileNotFoundException**;

**import** **java.io.FileOutputStream**;

**import** **java.io.InputStreamReader**;

**import** **java.io.PrintWriter**;

**import** **java.net.MalformedURLException**;

**import** **java.net.URL**;

**import** **org.htmlparser.Node**;

**import** **org.htmlparser.Parser**;

**import** **org.htmlparser.tags.LinkTag**;

**import** **org.htmlparser.tags.TableColumn**;

**import** **org.htmlparser.tags.TableRow**;

**import** **org.htmlparser.tags.TableTag**;

**import** **org.htmlparser.util.NodeList**;

**import** **org.htmlparser.util.ParserException**;

**import** **org.htmlparser.util.SimpleNodeIterator**;

**public** **class** **SpiderParser** {

**static** PrintWriter outputStream;

*/\**

*\* Parse a book.*

*\*/*

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

*// String bookname = "sjecl"; // ?????*

String bookname = "xsfh"; *// ????*

**try** {

outputStream = **new** PrintWriter(**new** FileOutputStream("SpiderParser\_" + bookname + ".txt"));

} **catch** (FileNotFoundException e1) {

e1.printStackTrace();

}

String url = "http://www.my285.com/wuxia/jinyong/"+bookname+"/index.htm";

*// Using htmlParser*

Parser parser = **null**;

**try** {

parser = **new** Parser(url);

} **catch** (ParserException e) {

e.printStackTrace();

}

NodeList nodelist = **null**;

**try** {

nodelist = parser.parse(**null**);

} **catch** (ParserException e) {

e.printStackTrace();

}

*// Nodelist of html*

nodelist = nodelist.toNodeArray()[0].getChildren();

findTable(nodelist);

outputStream.close();

}

*/\**

*\* Find out all tables in the node list.*

*\*/*

**public** **static** void findTable(NodeList nodelist){

**if**(**null** == nodelist){

**return**;

}

SimpleNodeIterator iterator = nodelist.elements();

**while**(iterator.hasMoreNodes()){

Node node = iterator.nextNode();

**if**(node **instanceof** TableTag){

parseTable(node);

}

*// Do it recursively to its children,*

*// even it is a table.*

NodeList childNodeList = node.getChildren();

findTable(childNodeList);

}

}

*/\**

*\* Select the table we need and parse it.*

*\*/*

**public** **static** void parseTable(Node node){

TableTag table = (TableTag) node;

*// The table we need is larger than 15.*

**if**(table.getChildren().size() > 15){

TableRow[] rows = table.getRows();

**for**(int i=0;i<rows.length;i++){

TableRow row = rows[i];

TableColumn[] columns = row.getColumns();

**for**(int j=0;j<columns.length;j++){

TableColumn column = columns[j];

Node[] links = column.getChildrenAsNodeArray();

**for**(int k=0;k<links.length;k++){

**if**(links[k] **instanceof** LinkTag){

LinkTag link = (LinkTag) links[k];

*// The title is the context of the link.*

String title = link.toPlainTextString();

*// Message on console*

System.out.println("Start " + title);

*// My special format for the beginning of each chapter*

outputStream.println("\r\n\r\n\r\n");

outputStream.println(title);

outputStream.println("==================");

outputStream.println("Parsed by Tao LIN.");

**try** {

parsePage(**new** URL(link.getLink()));

} **catch** (MalformedURLException e) {

e.printStackTrace();

} **catch** (Exception e) {

e.printStackTrace();

}

}

}

}

}

}

}

*/\**

*\* Parse each sub-page in the traditional way,*

*\* as it is much easier.*

*\*/*

**public** **static** void parsePage(URL pageURL) **throws** Exception{

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(pageURL.openStream()));

String buf;

boolean start = **false**;

**while**(**null** != (buf=in.readLine())){

*// The passage in contained between <td colspan="2"> and </td>*

*// Each paragraph is surrounded by <br>*

**if**(start){

**if**(buf.contains("</td>"))

**break**;

int indexEnd = buf.indexOf("<br>");

**if**(indexEnd > 0){

String para = buf.substring(0,indexEnd);

outputStream.println(para);

}

}

**else** **if**(buf.contains("<td colspan=\"2\">")){

start = **true**;

}

}

in.close();

}

}

# 参考资料

JavaTM Platform Standard Edition 6 API 规范<http://www.cjsdn.net/Doc/JDK60/>

HttpClient使用详解<http://blog.csdn.net/wangpeng047/article/details/19624529>

使用 HttpClient 和 HtmlParser 实现简易爬虫<https://www.ibm.com/developerworks/cn/opensource/os-cn-crawler/>

httpclient的一些学习心得<http://wallimn.iteye.com/blog/540566>

HttpClient Tutorial <http://hc.apache.org/httpcomponents-client-ga/tutorial/pdf/httpclient-tutorial.pdf>