jsoup Cookbook(中文版)

jsoup 简介

Java 程序在解析 HTML 文档时，相信大家都接触过 htmlparser 这个开源项目，我曾经在 IBM DW 上发表过两篇关于 htmlparser 的文章，分别是：从 HTML 中攫取你所需的信息和 扩展 HTMLParser 对自定义标签的处理能力。但

现在我已经不再使用 htmlparser 了，原因是 htmlparser 很少更新，但最重要的是有了 jsoup 。

jsoup 是一款 Java 的 HTML 解析器，可直接解析某个 URL 地址、HTML 文本内容。它提供了一套非常省力的 API，可通过 DOM，CSS 以及类似于 jQuery 的操作方法来取出和操作数据。

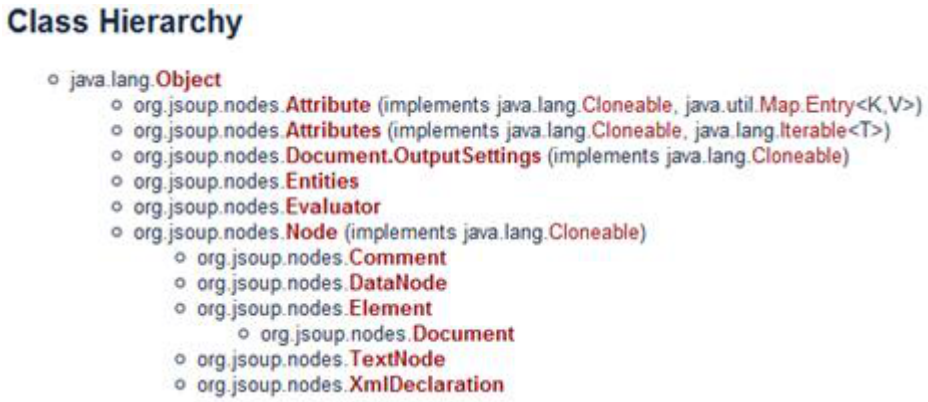
jsoup 的主要功能如下：

1. 从一个 URL，文件或字符串中解析 HTML；
2. 使用 DOM 或 CSS 选择器来查找、取出数据；
3. 可操作 HTML 元素、属性、文本；

jsoup 是基于 MIT 协议发布的，可放心使用于商业项目。

jsoup 的主要类层次结构如图 1 所示：

图 1. jsoup 的类层次结构



接下来我们专门针对几种常见的应用场景举例说明 jsoup 是如何优雅的进行 HTML 文档处理的。

回页首

文档输入

jsoup 可以从包括字符串、URL 地址以及本地文件来加载 HTML 文档，并生成 Document 对象实例。

下面是相关代码：

清单 1

```
// 直接从字符串中输入 HTML 文档
String html = "<html><head><title> 开源中国社区 </title></head>"
 + "<body><p> 这里是 jsoup 项目的相关文章 </p></body></html>";
Document doc = Jsoup.parse(html);

// 从 URL 直接加载 HTML 文档
Document doc = Jsoup.connect("http://www.oschina.net/").get();
String title = doc.title();

Document doc = Jsoup.connect("http://www.oschina.net/")
 .data("query", "Java")    // 请求参数
 .userAgent("I ’ m jsoup") // 设置 User-Agent
 .cookie("auth", "token") // 设置 cookie
 .timeout(3000)                 // 设置连接超时时间
```

```
 .post();                        // 使用 POST 方法访问 URL

// 从文件中加载 HTML 文档
File input = new File("D:/test.html");
Document doc = Jsoup.parse(input,"UTF-8","http://www.oschina.net/");
```

请大家注意最后一种 HTML 文档输入方式中的 parse 的第三个参数，为什么需要在这里指定一个网址呢（虽然可以不指定，如第一种方法）？因为 HTML 文档中会有很多例如链接、图片以及所引用的外部脚本、css 文件等，而第三个名为 baseURL 的参数的意思就是当 HTML 文档使用相对路径方式引用外部文件时，jsoup 会自动为这些 URL 加上一个前缀，也就是这个 baseURL。

例如 开源软件 会被转换成 开源软件 。

回页首

解析并提取 HTML 元素

这部分涉及一个 HTML 解析器最基本的功能，但 jsoup 使用一种有别于其他开源项目的方式——选择器，我们将在最后一部分详细介绍 jsoup 选择器，本节中你将看到 jsoup 是如何用最简单的代码实现。

不过 jsoup 也提供了传统的 DOM 方式的元素解析，看看下面的代码：

清单 2.

```
File input = new File("D:/test.html");
Document doc = Jsoup.parse(input, "UTF-8", "http://www.oschina.net/");

Element content = doc.getElementById("content");
Elements links = content.getElementsByTag("a");
for (Element link : links) {
  String linkHref = link.attr("href");
  String linkText = link.text();
}
```

你可能会觉得 jsoup 的方法似曾相识，没错，像 getElementById 和 getElementsByTag 方法跟 JavaScript 的方法名称是一样的，功能也完全一致。你可以根据节点名称或者是 HTML 元素的 id 来获取对应的元素或者元素列表。

与 htmlparser 项目不同的是，jsoup 并没有为 HTML 元素定义一个对应的类，一般一个 HTML 元素的组成部分包括：节点名、属性和文本，jsoup 提供简单的方法供你自己检索这些数据，这也是 jsoup 保持瘦身的原因。

而在元素检索方面，jsoup 的选择器简直无所不能，

清单 3.

```
File input = new File("D:\test.html");
Document doc = Jsoup.parse(input,"UTF-8","http://www.oschina.net/");
```

```
Elements links = doc.select("a[href]"); // 具有 href 属性的链接
Elements pngs = doc.select("img[src$=.png]");// 所有引用 png 图片的元素



Element masthead = doc.select("div.masthead").first();
// 找出定义了 class=masthead 的元素

Elements resultLinks = doc.select("h3.r > a"); // direct a after h3
```

这是 jsoup 真正让我折服的地方，jsoup 使用跟 jQuery 一模一样的选择器对元素进行检索，以上的检索方法如果换成是其他的 HTML 解释器，至少都需要很多行代码，而 jsoup 只需要一行代码即可完成。

jsoup 的选择器还支持表达式功能，我们将在最后一节介绍这个超强的选择器。

回页首

修改数据

在解析文档的同时，我们可能会需要对文档中的某些元素进行修改，例如我们可以为文档中的所有图片增加可点击链接、修改链接地址或者是修改文本等。

下面是一些简单的例子：

清单 4.

```
doc.select("div.comments a").attr("rel", "nofollow");
// 为所有链接增加 rel=nofollow 属性
doc.select("div.comments a").addClass("mylinkclass");
// 为所有链接增加 class=mylinkclass 属性
doc.select("img").removeAttr("onclick"); // 删除所有图片的 onclick 属性
doc.select("input[type=text]").val(""); // 清空所有文本输入框中的文本
```

道理很简单，你只需要利用 jsoup 的选择器找出元素，然后就可以通过以上的方法来进行修改，除了无法修改标签名外（可以删除后再插入新的元素），包括元素的属性和文本都可以修改。

修改完直接调用 Element(s) 的 html() 方法就可以获取修改完的 HTML 文档。

回页首

HTML 文档清理

jsoup 在提供强大的 API 同时，人性化方面也做得非常好。在做网站的时候，经常会提供用户评论的功能。有些用户比较淘气，会搞一些脚本到评论内容中，而这些脚本可能会破坏整个页面的行为，更严重的是获取一些机要信息，例如 XSS 跨站点攻击之类的。

jsoup 对这方面的支持非常强大，使用非常简单。看看下面这段代码：

清单 5.

```

```

```
String unsafe = "<p><a href='http://www.oschina.net/' onclick='stealCookies()'>
开源中国社区 </a></p>";
String safe = Jsoup.clean(unsafe, Whitelist.basic());
// 输出 :
// <p><a href="http://www.oschina.net/" rel="nofollow"> 开源中国社区 </a></p>
```

jsoup 使用一个 Whitelist 类用来对 HTML 文档进行过滤，该类提供几个常用方法：

表 1. 常用方法：

方法名	简介
none()	只允许包含文本信息
basic()	允许的标签包括：a, b, blockquote, br, cite, code, dd, dl, dt, em, i, li, ol, p, pre, q, small, strike, strong, sub, sup, u, ul, 以及合适的属性
simpleText()	只允许 b, em, i, strong, u 这些标签
basicWithImages()	在 basic() 的基础上增加了图片
relaxed()	这个过滤器允许的标签最多，包括：a, b, blockquote, br, caption, cite, code, col, colgroup, dd, dl, dt, em, h1, h2, h3, h4, h5, h6, i, img, li, ol, p, small, strike, strong, sub, sup, table, tbody, td, tfoot, th, thead, tr, u, ul

如果这五个过滤器都无法满足你的要求呢，例如你允许用户插入 flash 动画，没关系，Whitelist 提供扩展功能，例如 whitelist.addTags("embed","object","param","span","div"); 也可调用 addAttributes 为某些元素增加属性。

回页首

jsoup 的过人之处——选择器

前面我们已经简单的介绍了 jsoup 是如何使用选择器来对元素进行检索的。本节我们把重点放在选择器本身强大的语法上。下表是 jsoup 选择器的所有语法详细列表。

表 2. 基本用法：

tagname	使用标签名来定位，例如 a
ns|tag	使用命名空间的标签定位，例如 fb:name 来查找 <fb:name> 元素
#id	使用元素 id 定位，例如 #logo
.class	使用元素的 class 属性定位，例如 .head
[attribute]	使用元素的属性进行定位，例如 [href] 表示检索具有 href 属性的所有元素
[^attr]	使用元素的属性名前缀进行定位，例如 [^data-] 用来查找 HTML5 的 dataset 属性

[attr=value]	使用属性值进行定位，例如 [width=500] 定位所有 width 属性值为 500 的元素
[attr^=value], [attr$=value], [attr*=value]	这三个语法分别代表，属性以 value 开头、结尾以及包含
[attr~=regex]	使用正则表达式进行属性值的过滤，例如 img[src~=(?i)\.(png|jpe?g)]
*	定位所有元素

以上是最基本的选择器语法，这些语法也可以组合起来使用，下面是 jsoup 支持的组合用法：

表 3：组合用法：

el#id	定位 id 值某个元素，例如 a#logo -> <a id=logo href= … >
el.class	定位 class 为指定值的元素，例如 div.head -> xxxx
el[attr]	定位所有定义了某属性的元素，例如 a[href]
以上三个任意组合	例如 a[href]#logo 、a[name].outerlink
ancestor child	这五种都是元素之间组合关系的选择器语法，其中包括父子关系、合并关系和层次关系。
parent > child	
siblingA + siblingB	
siblingA ~ siblingX	
el, el, el	

除了一些基本的语法以及这些语法进行组合外，jsoup 还支持使用表达式进行元素过滤选择。下面是 jsoup 支持的所有表达式一览表：

表 4. 表达式：

:lt(n)	例如 td:lt(3) 表示 小于三列
:gt(n)	div p:gt(2) 表示 div 中包含 2 个以上的 p
:eq(n)	form input:eq(1) 表示只包含一个 input 的表单
:has(seletor)	div:has(p) 表示包含了 p 元素的 div
:not(selector)	div:not(.logo) 表示不包含 class=logo 元素的所有 div 列表
:contains(text)	包含某文本的元素，不区分大小写，例如 p:contains(oschina)

:containsOwn(text)	文本信息完全等于指定条件的过滤
:matches(regex)	使用正则表达式进行文本过滤：div:matches((?i)login)
:matchesOwn(regex)	使用正则表达式找到自身的文本

回页首

总结

jsoup 的基本功能到这里就介绍完毕，但由于 jsoup 良好的可扩展性 API 设计，你可以通过选择器的定义来开发出非常强大的 HTML 解析功能。再加上 jsoup 项目本身的开发也非常活跃，因此如果你正在使用 Java ，需要对 HTML 进行处理，不妨试试。

参考资料

学习

- jsoup 官方网站：http://jsoup.org
- 开源中国社区上 jsoup 的一些代码片段：http://www.oschina.net/code/list_by_project?id=12689
- Htmlparser 项目介绍：http://www.oschina.net/p/htmlparser
- jQuery 项目介绍：http://www.oschina.net/p/jquery
- 更多开源项目的介绍来自开源中国社区：http://www.oschina.net
- developerWorks Java 技术专区： 这里有数百篇关于 Java 编程每个方面的文章。
- 查看 HTML5 专题，了解更多和 HTML5 相关的知识和动向。

讨论

- 加入 developerWorks 中文社区。

HttpClient 简介

HTTP 协议可能是现在 Internet 上使用得最多、最重要的协议了，越来越多的 Java 应用程序需要直接通过 HTTP 协议来访问网络资源。虽然在 JDK 的 java.net 包中已经提供了访问 HTTP 协议的基本功能，但是对于大部分应用程序来说，JDK 库本身提供的功能还不够丰富和灵活。HttpClient 是 Apache Jakarta Common 下的子项目，用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包，并且它支持 HTTP 协议最新的版本和建议。HttpClient

已经应用在很多的项目中，比如 Apache Jakarta 上很著名的另外两个开源项目 Cactus 和 HTMLUnit 都使用了 HttpClient，更多使用 HttpClient 的应用可以参见 http://wiki.apache.org/jakarta-httpclient/HttpClientPowered。HttpClient 项目非常活跃，使用的人还是非常多的。目前 HttpClient 版本是在 2005.10.11 发布的 3.0 RC4 。

回页首

HttpClient 功能介绍

以下列出的是 HttpClient 提供的主要的功能，要知道更多详细的功能可以参见 HttpClient 的主页。

- 实现了所有 HTTP 的方法（GET,POST,PUT,HEAD 等）
- 支持自动转向
- 支持 HTTPS 协议
- 支持代理服务器等

下面将逐一介绍怎样使用这些功能。首先，我们必须安装好 HttpClient。

- HttpClient 可以在 http://jakarta.apache.org/commons/httpclient/downloads.html 下载
- HttpClient 用到了 Apache Jakarta common 下的子项目 logging，你可以从这个地址 http://jakarta.apache.org/site/downloads/downloads_commons-logging.cgi 下载到 common logging，从下载后的压缩包中取出 commons-logging.jar 加到 CLASSPATH 中
- HttpClient 用到了 Apache Jakarta common 下的子项目 codec，你可以从这个地址 http://jakarta.apache.org/site/downloads/downloads_commons-codec.cgi 下载到最新的 common codec，从下载后的压缩包中取出 commons-codec-1.x.jar 加到 CLASSPATH 中

回页首

HttpClient 基本功能的使用

GET 方法

使用 HttpClient 需要以下 6 个步骤：

1. 创建 HttpClient 的实例
2. 创建某种连接方法的实例，在这里是 GetMethod。在 GetMethod 的构造函数中传入待连接的地址
3. 调用第一步中创建好的实例的 execute 方法来执行第二步中创建好的 method 实例
4. 读 response
5. 释放连接。无论执行方法是否成功，都必须释放连接
6. 对得到后的内容进行处理

根据以上步骤，我们来编写用 GET 方法来取得某网页内容的代码。

- 大部分情况下 HttpClient 默认的构造函数已经足够使用。

```
HttpClient httpClient = new HttpClient();
```

-
- 创建 GET 方法的实例。在 GET 方法的构造函数中传入待连接的地址即可。用 GetMethod 将会自动处理转发过程，如果想要把自动处理转发过程去掉的话，可以调用方法 setFollowRedirects(false)。

```
GetMethod getMethod = new GetMethod("http://www.ibm.com/");
```

-
- 调用实例 httpClient 的 executeMethod 方法来执行 getMethod。由于是执行在网络上的程序，在运行 executeMethod 方法的时候，需要处理两个异常，分别是 HttpException 和 IOException。引起第一种异常的原因主要可能是在构造 getMethod 的时候传入的协议不对，比如不小心将"http"写成"htp"，或者服务器端返回的内容不

正常等，并且该异常发生是不可恢复的；第二种异常一般是由于网络原因引起的异常，对于这种异常（IOException），HttpClient 会根据你指定的恢复策略自动试着重新执行 executeMethod 方法。HttpClient 的恢复策略可以自定义（通过实现接口 HttpMethodRetryHandler 来实现）。通过 httpClient 的方法 setParameter 设置你实现的恢复策略，本文中使用的是系统提供的默认恢复策略，该策略在碰到第二类异常的时候将自动重试3次。executeMethod 返回值是一个整数，表示了执行该方法后服务器返回的状态码，该状态码能表示出该方法执行是否成功、需要认证或者页面发生了跳转（默认状态下 GetMethod 的实例是自动处理跳转的）等。

```
//设置成了默认的恢复策略，在发生异常时候将自动重试3次，在这里你也可以设置成自定义的恢复策略
getMethod.getParams().setParameter(HttpMethodParams.RETRY_HANDLER,
        new DefaultHttpMethodRetryHandler());
//执行 getMethod
int statusCode = client.executeMethod(getMethod);
if (statusCode != HttpStatus.SC_OK) {
  System.err.println("Method failed: " + getMethod.getStatusLine());
}
```

- 在返回的状态码正确后，即可取得内容。取得目标地址的内容有三种方法：第一种，getResponseBody，该方法返回的是目标的二进制的 byte 流；第二种，getResponseBodyAsString，这个方法返回的是 String 类型，值得注意的是该方法返回的 String 的编码是根据系统默认的编码方式，所以返回的 String 值可能编码类型有误，在本文的"字符编码"部分中将对此做详细介绍；第三种，getResponseBodyAsStream，这个方法对于目标地址中有大量数据需要传输是最佳的。在这里我们使用了最简单的 getResponseBody 方法。

```
byte[] responseBody = method.getResponseBody();
```

- 释放连接。无论执行方法是否成功，都必须释放连接。

```
method.releaseConnection();
```

- 处理内容。在这一步中根据你的需要处理内容，在例子中只是简单的将内容打印到控制台。

```
System.out.println(new String(responseBody));
```

下面是程序的完整代码，这些代码也可在附件中的 test.GetSample 中找到。

```
package test;
import java.io.IOException;
import org.apache.commons.httpclient.*;
import org.apache.commons.httpclient.methods.GetMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
public class GetSample{
  public static void main(String[] args) {
  //构造 HttpClient 的实例
```

```
  HttpClient httpClient = new HttpClient();
  //创建 GET 方法的实例
  GetMethod getMethod = new GetMethod("http://www.ibm.com");
  //使用系统提供的默认的恢复策略
  getMethod.getParams().setParameter(HttpMethodParams.RETRY_HANDLER,
     new DefaultHttpMethodRetryHandler());
  try {
   //执行 getMethod
   int statusCode = httpClient.executeMethod(getMethod);
   if (statusCode != HttpStatus.SC_OK) {
    System.err.println("Method failed: "
      + getMethod.getStatusLine());
   }
   //读取内容
   byte[] responseBody = getMethod.getResponseBody();
   //处理内容
   System.out.println(new String(responseBody));
  } catch (HttpException e) {
   //发生致命的异常，可能是协议不对或者返回的内容有问题
   System.out.println("Please check your provided http address!");
   e.printStackTrace();
  } catch (IOException e) {
   //发生网络异常
   e.printStackTrace();
  } finally {
   //释放连接
   getMethod.releaseConnection();
  }
 }
}
```

POST 方法

根据 RFC2616，对 POST 的解释如下：POST 方法用来向目的服务器发出请求，要求它接受被附在请求后的实体，并把它当作请求队列（Request-Line）中请求 URI 所指定资源的附加新子项。POST 被设计成用统一的方法实现下列功能：

- 对现有资源的注释（Annotation of existing resources）
- 向电子公告栏、新闻组，邮件列表或类似讨论组发送消息
- 提交数据块，如将表单的结果提交给数据处理过程
- 通过附加操作来扩展数据库

调用 HttpClient 中的 PostMethod 与 GetMethod 类似，除了设置 PostMethod 的实例与 GetMethod 有些不同之外，剩下的步骤都差不多。在下面的例子中，省去了与 GetMethod 相同的步骤，只说明与上面不同的地方，并以登录清华大学 BBS 为例子进行说明。

- 构造 PostMethod 之前的步骤都相同，与 GetMethod 一样，构造 PostMethod 也需要一个 URI 参数，在本例

中，登录的地址是 http://www.newsmth.net/bbslogin2.php。在创建了 PostMethod 的实例之后，需要给 method 实例填充表单的值，在 BBS 的登录表单中需要有两个域，第一个是用户名（域名叫 id），第二个是密码（域名叫 passwd）。表单中的域用类 NameValuePair 来表示，该类的构造函数第一个参数是域名，第二参数是该域的值；将表单所有的值设置到 PostMethod 中用方法 setRequestBody。另外由于 BBS 登录成功后会转向另外一个页面，但是 HttpClient 对于要求接受后继服务的请求，比如 POST 和 PUT，不支持自动转发，因此需要自己对页面转向做处理。具体的页面转向处理请参见下面的"自动转向"部分。代码如下：

```
String url = "http://www.newsmth.net/bbslogin2.php";
PostMethod postMethod = new PostMethod(url);
// 填入各个表单域的值
NameValuePair[] data = { new NameValuePair("id", "youUserName"),
new NameValuePair("passwd", "yourPwd") };
// 将表单的值放入 postMethod 中
postMethod.setRequestBody(data);
// 执行 postMethod
int statusCode = httpClient.executeMethod(postMethod);
// HttpClient 对于要求接受后继服务的请求，象 POST 和 PUT 等不能自动处理转发
// 301或者302
if (statusCode == HttpStatus.SC_MOVED_PERMANENTLY ||
statusCode == HttpStatus.SC_MOVED_TEMPORARILY) {
    // 从头中取出转向的地址
    Header locationHeader = postMethod.getResponseHeader("location");
    String location = null;
    if (locationHeader != null) {
      location = locationHeader.getValue();
      System.out.println("The page was redirected to:" + location);
    } else {
      System.err.println("Location field value is null.");
    }
    return;
}
```

HttpURLConnection 和 HttpClient+Jsoup 处理标签抓取页面和模拟登录

【案例一】

```
package com.app.html;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
```

```
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;

public class Html {
	private static final String loginURL = "http://login.goodjobs.cn/index.php/action/UserLogin";
	private static final String forwardURL = "http://user.goodjobs.cn/dispatcher.php/module/Personal/?skip_fill=1";

	/**
	 * 获取登录页面请求
	 * @param loginUrl 登录 URL
	 * @param params 登录用户名/密码参数
	 * @throws Exception
	 */
	public static String	createHtml(String...params)throws Exception{
		URL url = new URL(loginURL);
		HttpURLConnection conn = (HttpURLConnection) url.openConnection();
		conn.setDoOutput(true);
		loginHtml(conn, params);
		return forwardHtml(conn,url);
	}
	/**
	 * 登录页面
	 * @param conn
	 * @param params 登录用户名/密码参数
	 * @throws Exception
	 */
	private static void loginHtml(HttpURLConnection conn, String... params)
			throws Exception {
		OutputStreamWriter out = new OutputStreamWriter(conn.getOutputStream(), "GBK");
		StringBuffer buff=new StringBuffer();
		buff.append("memberName="+URLEncoder.encode(params[0], "UTF-8"));//页面用户名
		buff.append("&password="+URLEncoder.encode(params[1],"UTF-8"));//页面密码
		out.write(buff.toString());//填充参数
		out.flush();
		out.close();
	}
```

```
    /**
     * 转向到定向的页面
     * @param conn 连接对象
     * @param url 重新定向请求 URL
     * @param toUrl 定向到页面请求 URL
     * @throws Exception
     */
    public static String forwardHtml(HttpURLConnection conn,URL url)throws Exception{
        //重新打开一个连接
        String cookieVal = conn.getHeaderField("Set-Cookie");
        url = new URL(forwardURL);
        conn = (HttpURLConnection) url.openConnection();
        conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
        conn.setRequestProperty("User-Agent","Mozilla/4.0 (compatible; MSIE 6.0; Windows
NT 5.1; SV1; Foxy/1; .NET CLR 2.0.50727;MEGAUPLOAD 1.0)");
        conn.setFollowRedirects(false);//置此类是否应该自动执行 HTTP 重定向
        // 取得 cookie,相当于记录了身份,供下次访问时使用
        if (cookieVal != null) {
            //发送 cookie 信息上去,以表明自己的身份,否则会被认为没有权限
            conn.setRequestProperty("Cookie", cookieVal);
        }
        conn.connect();
        InputStream in = conn.getInputStream();
        BufferedReader buffReader = new BufferedReader(    new
InputStreamReader(in,"GBK"));
        String line = null;
        String content = "";
        while ((line = buffReader.readLine()) != null) {
            content +="\n" +line;
        }
        //IOUtils.write(result, new FileOutputStream("d:/index.html"),"GBK");
        write(content, "d:/forward.html");
        buffReader.close();
        return content;
    }

    /**
     *
     * @param content
     * @param htmlPath
     * @return
     */
    public static boolean write(String content, String htmlPath) {
            boolean flag = true;
```

```
                try {
                        Writer out = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(htmlPath), "GBK"));
                    out.write("\n" + content);
                    out.close();
                } catch (FileNotFoundException ex) {
                    ex.printStackTrace();
                    return false;
                } catch (UnsupportedEncodingException ex) {
                    ex.printStackTrace();
                    return false;
                } catch (IOException ex) {
                    ex.printStackTrace();
                    return false;
                }
                return flag;
            }


    public static void main(String[] args)throws Exception{
        String [] params={"admin","admin12"};
        System.out.println(createHtml(params));
    }
}
```

【案例二】

```
package com.app.html;
import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.apache.commons.httpclient.Cookie;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.cookie.CookiePolicy;
import org.apache.commons.httpclient.cookie.CookieSpec;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
```

```
public class HttpClientHtml {
    private static final String SITE = "login.goodjobs.cn";
    private static final int PORT = 80;
    private static final String loginAction = "/index.php/action/UserLogin";
    private static final String forwardURL = "http://user.goodjobs.cn/dispatcher.php/module/Personal/?skip_fill=1";

    /**
     * 模拟等录
     * @param LOGON_SITE
     * @param LOGON_PORT
     * @param login_Action
     * @param params
     * @throws Exception
     */
    private static HttpClient loginHtml(String LOGON_SITE, int LOGON_PORT,String login_Action,String ...params) throws Exception {
        HttpClient client = new HttpClient();
        client.getHostConfiguration().setHost(LOGON_SITE, LOGON_PORT);
        // 模拟登录页面
        PostMethod post = new PostMethod(login_Action);
        NameValuePair userName = new NameValuePair("memberName",params[0] );
        NameValuePair password = new NameValuePair("password",params[1] );
        post.setRequestBody(new NameValuePair[] { userName, password });
        client.executeMethod(post);
        post.releaseConnection();
        // 查看 cookie 信息
        CookieSpec cookiespec = CookiePolicy.getDefaultSpec();
        Cookie[] cookies = cookiespec.match(LOGON_SITE, LOGON_PORT, "/", false,
                client.getState().getCookies());
        if (cookies != null)
            if (cookies.length == 0) {
                System.out.println("Cookies is not Exists ");
            } else {
                for (int i = 0; i < cookies.length; i++) {
                    System.out.println(cookies[i].toString());
                }
            }
        return client;
    }
    /**
     * 模拟等录 后获取所需要的页面
     * @param client
     * @param newUrl
```

```
     * @throws Exception
     */
    private static void createHtml(HttpClient client, String newUrl)
                throws    Exception {
        PostMethod post = new PostMethod(newUrl);
        client.executeMethod(post);
        post.getParams().setParameter(HttpMethodParams.HTTP_CONTENT_CHARSET,
"GBK");
        String content= post.getResponseBodyAsString();
        SimpleDateFormat format=new SimpleDateFormat("yyyy-MM-dd");
        //IOUtils.write(content,          new          FileOutputStream("d:/"+format.format(new
Date())+".html"),"GBK");
        write(content,"d:/"+format.format(new Date())+".html");
        post.releaseConnection();
    }



    public static void main(String[] args) throws Exception {
        String [] params={"admin","admin123"};
        HttpClient client = loginHtml(SITE, PORT, loginAction,params);
        // 访问所需的页面
        createHtml(client, forwardURL);

        //System.out.println(UUID.randomUUID());
    }



    /**
     *
     * @param content
     * @param htmlPath
     * @return
     */
     public static boolean write(String content, String htmlPath) {
            boolean flag = true;
            try {
                        Writer   out   =   new   BufferedWriter(new   OutputStreamWriter(new
FileOutputStream(htmlPath), "GBK"));
                out.write("\n" + content);
                out.close();
            } catch (FileNotFoundException ex) {
                ex.printStackTrace();
                return false;
```

```
                } catch (UnsupportedEncodingException ex) {
                    ex.printStackTrace();
                    return false;
                } catch (IOException ex) {
                    ex.printStackTrace();
                    return false;
                }
                return flag;
            }
}
```

【案例三】

```
package com.app.html;
import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.apache.commons.httpclient.Cookie;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.cookie.CookiePolicy;
import org.apache.commons.httpclient.cookie.CookieSpec;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;

public class HttpClientHtml {
    private static final String SITE = "login.goodjobs.cn";
    private static final int PORT = 80;
    private static final String loginAction = "/index.php/action/UserLogin";
    private static final String forwardURL =
"http://user.goodjobs.cn/dispatcher.php/module/Personal/?skip_fill=1";

    /**
     * 模拟等录
     * @param LOGON_SITE
     * @param LOGON_PORT
     * @param login_Action
     * @param params
```

```
     * @throws Exception
     */
    private static HttpClient loginHtml(String LOGON_SITE, int LOGON_PORT,String login_Action,String ...params) throws Exception {
        HttpClient client = new HttpClient();
        client.getHostConfiguration().setHost(LOGON_SITE, LOGON_PORT);
        // 模拟登录页面
        PostMethod post = new PostMethod(login_Action);
        NameValuePair userName = new NameValuePair("memberName",params[0] );
        NameValuePair password = new NameValuePair("password",params[1] );
        post.setRequestBody(new NameValuePair[] { userName, password });
        client.executeMethod(post);
        post.releaseConnection();
        // 查看 cookie 信息
        CookieSpec cookiespec = CookiePolicy.getDefaultSpec();
        Cookie[] cookies = cookiespec.match(LOGON_SITE, LOGON_PORT, "/", false,
                client.getState().getCookies());
        if (cookies != null)
            if (cookies.length == 0) {
                    System.out.println("Cookies is not Exists ");
            } else {
                for (int i = 0; i < cookies.length; i++) {
                     System.out.println(cookies[i].toString());
                }
            }
        return client;
    }
    /**
     * 模拟等录 后获取所需要的页面
     * @param client
     * @param newUrl
     * @throws Exception
     */
    private static void createHtml(HttpClient client, String newUrl)
            throws  Exception {
        PostMethod post = new PostMethod(newUrl);
        client.executeMethod(post);
        post.getParams().setParameter(HttpMethodParams.HTTP_CONTENT_CHARSET,
"GBK");
        String content= post.getResponseBodyAsString();
        SimpleDateFormat format=new SimpleDateFormat("yyyy-MM-dd");
        //IOUtils.write(content,          new          FileOutputStream("d:/"+format.format(new
Date())+".html"),"GBK");
        write(content,"d:/"+format.format(new Date())+".html");
```

```
            post.releaseConnection();
        }



        public static void main(String[] args) throws Exception {
            String [] params={"admin","admin123"};
            HttpClient client = loginHtml(SITE, PORT, loginAction,params);
            // 访问所需的页面
            createHtml(client, forwardURL);

            //System.out.println(UUID.randomUUID());
        }



        /**
         *
         * @param content
         * @param htmlPath
         * @return
         */
         public static boolean write(String content, String htmlPath) {
                boolean flag = true;
                try {
                                Writer out = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(htmlPath), "GBK"));
                    out.write("\n" + content);
                    out.close();
                } catch (FileNotFoundException ex) {
                    ex.printStackTrace();
                    return false;
                } catch (UnsupportedEncodingException ex) {
                    ex.printStackTrace();
                    return false;
                } catch (IOException ex) {
                    ex.printStackTrace();
                    return false;
                }
                return flag;
            }
}
```

【案例四】

```
package com.app.html;
import java.io.BufferedReader;
```

```
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.apache.commons.httpclient.Cookie;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.cookie.CookiePolicy;
import org.apache.commons.httpclient.cookie.CookieSpec;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import com.app.comom.FileUtil;

public class HttpClientHtml {
	private static final String SITE = "login.goodjobs.cn";
	private static final int PORT = 80;
	private static final String loginAction = "/index.php/action/UserLogin";
	private static final String forwardURL =
"http://user.goodjobs.cn/dispatcher.php/module/Personal/?skip_fill=1";
	private static final String toUrl = "d:\\test\\";
	private static final String css = "http://user.goodjobs.cn/personal.css";
	private static final String Img = "http://user.goodjobs.cn/images";
	private static final String _JS = "http://user.goodjobs.cn/scripts/fValidate/fValidate.one.js";
	/**
	 * 模拟等录
	 * @param LOGON_SITE
	 * @param LOGON_PORT
	 * @param login_Action
	 * @param params
	 * @throws Exception
	 */
	private static HttpClient loginHtml(String LOGON_SITE, int LOGON_PORT,String
login_Action,String ...params) throws Exception {
```

```
        HttpClient client = new HttpClient();
        client.getHostConfiguration().setHost(LOGON_SITE, LOGON_PORT);
        // 模拟登录页面
        PostMethod post = new PostMethod(login_Action);
        NameValuePair userName = new NameValuePair("memberName",params[0] );
        NameValuePair password = new NameValuePair("password",params[1] );
        post.setRequestBody(new NameValuePair[] { userName, password });
        client.executeMethod(post);
        post.releaseConnection();
        // 查看 cookie 信息
        CookieSpec cookiespec = CookiePolicy.getDefaultSpec();
        Cookie[] cookies = cookiespec.match(LOGON_SITE, LOGON_PORT, "/", false,
                client.getState().getCookies());
        if (cookies != null)
            if (cookies.length == 0) {
                    System.out.println("Cookies is not Exists ");
            } else {
                for (int i = 0; i < cookies.length; i++) {
                    System.out.println(cookies[i].toString());
                }
            }
        return client;
    }
    /**
     * 模拟等录 后获取所需要的页面
     * @param client
     * @param newUrl
     * @throws Exception
     */
    private static String    createHtml(HttpClient client, String newUrl) throws    Exception {
        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
        String filePath = toUrl    + format.format(new Date() )+ "_" + 1 + ".html";
        PostMethod post = new PostMethod(newUrl);
        client.executeMethod(post);
        //设置编码
        post.getParams().setParameter(HttpMethodParams.HTTP_CONTENT_CHARSET,
"GBK");
        String content= post.getResponseBodyAsString();
        FileUtil.write(content, filePath);
        System.out.println("\n 写入文件成功!");
        post.releaseConnection();
        return filePath;
    }
    /**
```

```
     * 解析 html 代码
     * @param filePath
     * @param random
     * @return
     */
    private static String JsoupFile(String filePath, int random) {

        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
        File infile = new File(filePath);
        String url = toUrl + format.format(new Date()) + "_new_" + random+ ".html";

        try {
            File outFile = new File(url);
            Document doc = Jsoup.parse(infile, "GBK");
            String html="<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01 Transitional//EN'>";
            StringBuffer sb = new StringBuffer();
            sb.append(html).append("\n");
            sb.append("<html>").append("\n");
            sb.append("<head>").append("\n");
            sb.append("<title>欢迎使用新安人才网个人专区</title>").append("\n");
            Elements meta = doc.getElementsByTag("meta");
            sb.append(meta.toString()).append("\n");

            ////////////////////////////body//////////////////////////
            Elements body = doc.getElementsByTag("body");

            ////////////////////////////link//////////////////////////
            Elements links = doc.select("link");//对 link 标签有 href 的路径都作处理

            for (Element link : links) {
                String hrefAttr = link.attr("href");
                if (hrefAttr.contains("/personal.css")) {
                    hrefAttr = hrefAttr.replace("/personal.css",css);
                    Element hrefVal=link.attr("href", hrefAttr);//修改 href 的属性值
                    sb.append(hrefVal.toString()).append("\n");
                }
            }
            ////////////////////////////script//////////////////////////
            Elements scripts = doc.select("script");//对 script 标签
            for (Element js : scripts) {
                String jsrc = js.attr("src");
                if (jsrc.contains("/fValidate.one.js")) {
                    String oldJS="/scripts/fValidate/fValidate.one.js";//之前的 css
```

```
                    jsrc = jsrc.replace(oldJS,_JS);
                    Element val=js.attr("src", jsrc);//修改 href 的属性值
                    sb.append(val.toString()).append("\n").append("</head>");
                }
            }

            ////////////////////////////script/////////////////////////
            Elements tags = body.select("*");//对所有标签有 src 的路径都作处理
            for (Element tag : tags) {
                String src = tag.attr("src");
                if (src.contains("/images")) {
                    src = src.replace("/images",Img);
                    tag.attr("src", src);//修改 src 的属性值
                }
            }

            sb.append(body.toString());
            sb.append("</html>");

            BufferedReader in = new BufferedReader(new FileReader(infile));
            Writer out = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(outFile), "gbk"));
            String content = sb.toString();
            out.write(content);
            in.close();

            System.out.println("页面已经爬完");
            out.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return url;
    }

    public static void main(String[] args) throws Exception {
        String [] params={"admin","admin123"};
        HttpClient client = loginHtml(SITE, PORT, loginAction,params);
        // 访问所需的页面
        String path=createHtml(client, forwardURL);
       System.out.println( JsoupFile(path,1));
    }

}
```

【案例五】

```
package com.app.html;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.apache.commons.httpclient.Cookie;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.cookie.CookiePolicy;
import org.apache.commons.httpclient.cookie.CookieSpec;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.params.HttpMethodParams;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import com.app.comom.FileUtil;

public class HttpClientHtml {
	private static final String SITE = "login.goodjobs.cn";
	private static final int PORT = 80;
	private static final String loginAction = "/index.php/action/UserLogin";
	private static final String forwardURL =
"http://user.goodjobs.cn/dispatcher.php/module/Personal/?skip_fill=1";
	private static final String toUrl = "d:\\test\\";
	private static final String hostCss    = "d:\\test\\style.txt";
	private static final String Img = "http://user.goodjobs.cn/images";
	private static final String _JS = "http://user.goodjobs.cn/scripts/fValidate/fValidate.one.js";
	/**
	 *  模拟等录
	 * @param LOGON_SITE
	 * @param LOGON_PORT
	 * @param login_Action
	 * @param params
	 * @throws Exception
	 */
```

```
    private static HttpClient loginHtml(String LOGON_SITE, int LOGON_PORT,String
login_Action,String ...params) throws Exception {
        HttpClient client = new HttpClient();
        client.getHostConfiguration().setHost(LOGON_SITE, LOGON_PORT);
        // 模拟登录页面
        PostMethod post = new PostMethod(login_Action);
        NameValuePair userName = new NameValuePair("memberName",params[0] );
        NameValuePair password = new NameValuePair("password",params[1] );
        post.setRequestBody(new NameValuePair[] { userName, password });
        client.executeMethod(post);
        post.releaseConnection();
        // 查看 cookie 信息
        CookieSpec cookiespec = CookiePolicy.getDefaultSpec();
        Cookie[] cookies = cookiespec.match(LOGON_SITE, LOGON_PORT, "/", false,
                client.getState().getCookies());
        if (cookies != null)
            if (cookies.length == 0) {
                    System.out.println("Cookies is not Exists ");
            } else {
                for (int i = 0; i < cookies.length; i++) {
                     System.out.println(cookies[i].toString());
                }
            }
        return client;
    }
    /**
     * 模拟等录 后获取所需要的页面
     * @param client
     * @param newUrl
     * @throws Exception
     */
    private static String    createHtml(HttpClient client, String newUrl) throws    Exception {
        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
        String filePath = toUrl    + format.format(new Date() )+ "_" + 1 + ".html";
        PostMethod post = new PostMethod(newUrl);
        client.executeMethod(post);
        //设置编码
        post.getParams().setParameter(HttpMethodParams.HTTP_CONTENT_CHARSET,
"GBK");
        String content= post.getResponseBodyAsString();
        FileUtil.write(content, filePath);
        System.out.println("\n 写入文件成功!");
        post.releaseConnection();
        return filePath;
```

```
    }
    /**
     * 解析 html 代码
     * @param filePath
     * @param random
     * @return
     */
    private static String JsoupFile(String filePath, int random) {

        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
        File infile = new File(filePath);
        String url = toUrl + format.format(new Date()) + "_new_" + random+ ".html";

        try {
            File outFile = new File(url);
            Document doc = Jsoup.parse(infile, "GBK");
            String html="<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01 Transitional//EN'>";
            StringBuffer sb = new StringBuffer();
            sb.append(html).append("\n");
            sb.append("<html>").append("\n");
            sb.append("<head>").append("\n");
            sb.append("<title>欢迎使用新安人才网个人专区</title>").append("\n");
            Elements meta = doc.getElementsByTag("meta");
            sb.append(meta.toString()).append("\n");
             ////////////////////////////本地 css/////////////////////////////

            File cssFile = new File(hostCss);
            BufferedReader in = new BufferedReader(new FileReader(cssFile));
                    Writer out = new BufferedWriter(new OutputStreamWriter( new FileOutputStream(outFile), "gbk"));
            String content=in.readLine();
            while(content!=null){
                //System.out.println(content);
                sb.append(content+"\n");
                content=in.readLine();
            }

            in.close();
            ////////////////////////////处理 body 标签/////////////////////////
            Elements body = doc.getElementsByTag("body");


            ////////////////////////////处理 script 标签/////////////////////////
```

```
                Elements scripts = doc.select("script");//对 script 标签
                for (Element js : scripts) {
                    String jsrc = js.attr("src");
                    if (jsrc.contains("/fValidate.one.js")) {
                        String oldJS="/scripts/fValidate/fValidate.one.js";//之前的 css
                        jsrc = jsrc.replace(oldJS,_JS);
                        Element val=js.attr("src", jsrc);//修改 href 的属性值
                        sb.append(val.toString()).append("\n").append("</head>");
                    }
                }

                ////////////////////////////处理所有 src 的属性值//////////////////////////
                Elements tags = body.select("*");//对所有标签有 src 的路径都作处理
                for (Element tag : tags) {
                    String src = tag.attr("src");
                    if (src.contains("/images")) {
                        src = src.replace("/images",Img);
                        tag.attr("src", src);//修改 src 的属性值
                    }
                }

                sb.append(body.toString());
                sb.append("</html>");

                out.write(sb.toString());
                in.close();

                System.out.println("页面已经爬完");
                out.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
            return url;
        }

        public static void main(String[] args) throws Exception {
            String [] params={"admin","admin123"};
            HttpClient client = loginHtml(SITE, PORT, loginAction,params);
            // 页面生成
            String path=createHtml(client, forwardURL);
            System.out.println( JsoupFile(path,1));
        }

}
```