链滴

Java 爬虫 CSDN

作者：NiuGeH

原文链接：https://ld246.com/article/1566723774753

来源网站：链滴

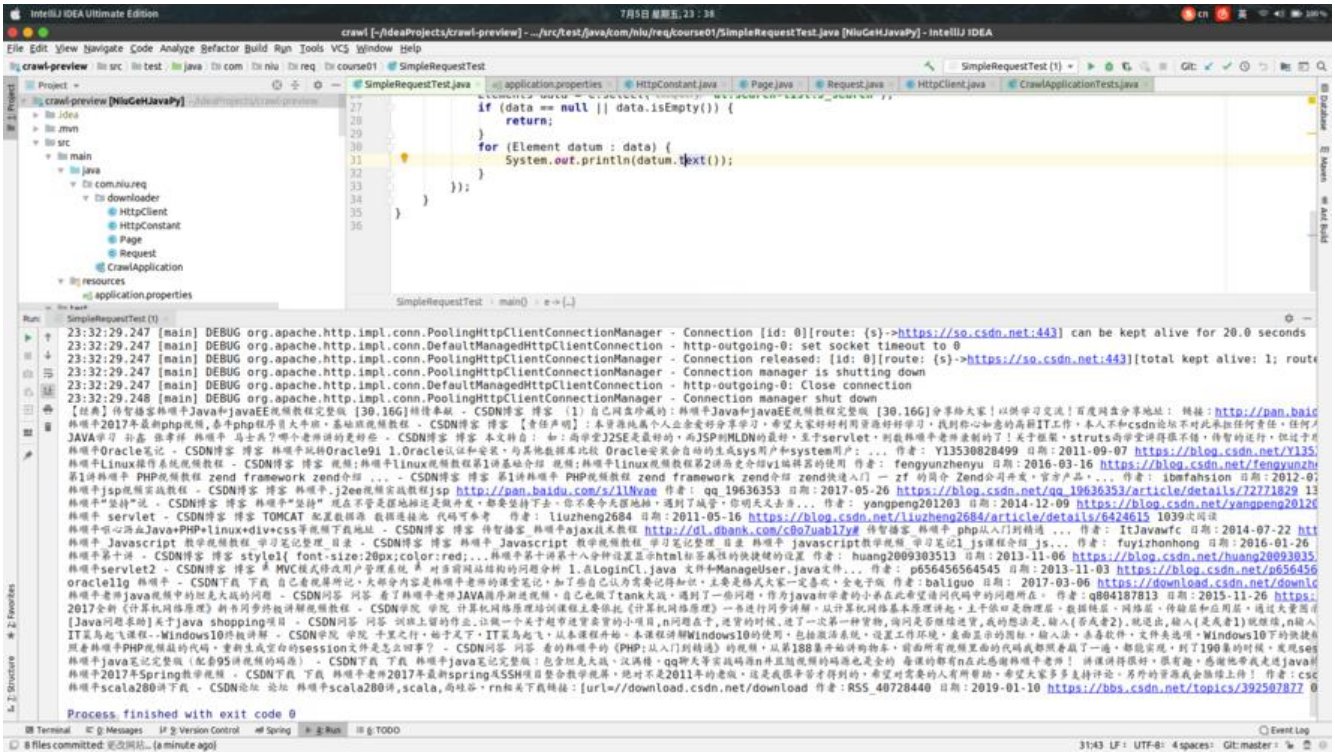
Java版的爬虫　爬取CSDN 搜索韩顺平的文章

这个是爬虫下的结果，结果为txt 可以使用poi 生成到excel 文件 具体我就不讲解了

万物皆可爬（Java 万能的）

本文项目使用Spring Boot 搭建 普通Maven项目亦可！！！

先看一下结果吧（不爬取广告栏）：

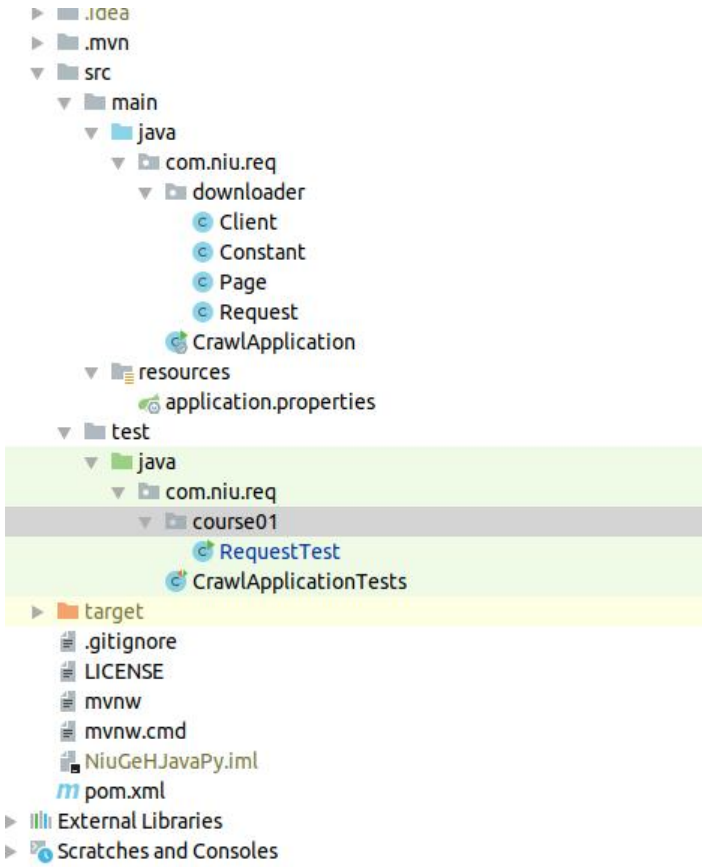


这个是CSDN官网的数据：

原文链接：Java 爬虫 CSDN

主要使用的是jsoup进行爬虫，英语比较好的话建议还是看下官方的文档 = =

这个是项目搭建结构



下载工具,包含各种类型的请求 Client.java

```
package com.niu.req.downloader;


import org.apache.http.NameValuePair;
import org.apache.http.client.config.RequestConfig;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.*;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.util.EntityUtils;
import org.springframework.util.StringUtils;

import java.io.UnsupportedEncodingException;
import java.util.*;

/**
 * @Description 下载工具，包含各种类型的请求
 */
public class Client {

    // 编码格式。发送编码格式统一用UTF-8
    private static final String ENCODING = "UTF-8";
```

```
// 设置连接超时时间，单位毫秒。
private static final int CONNECT_TIMEOUT = 60000;

// 请求获取数据的超时时间(即响应时间)，单位毫秒。
private static final int SOCKET_TIMEOUT = 60000;

public static Page doRequest(Request request) throws Exception {

    HttpRequestBase http = null;

    switch (request.getMethod()) {
        case POST:
            http = new HttpPost(request.getUrl());
            break;
        case GET:
        default:
            http = new HttpGet(request.getUrl());
            break;

    }

    RequestConfig requestConfig = RequestConfig.custom()
            .setConnectTimeout(CONNECT_TIMEOUT)
            .setSocketTimeout(SOCKET_TIMEOUT).build();

    http.setConfig(requestConfig);

    if (request.getHeaders() != null && !request.getHeaders().isEmpty()) {
        packageHeader(request.getHeaders(), http);
    }

    if (http instanceof HttpPost && request.getParams() != null) {
        packageParam(request.getParams(), (HttpPost) http);
    }


    Page page = Page.build(request);

    // 执行请求并获得响应结果
    try (final CloseableHttpClient httpClient = HttpClients.custom().build();
         final CloseableHttpResponse httpResponse = httpClient.execute(http)
    ) {
        if (httpResponse != null
                && !StringUtils.isEmpty(httpResponse.getStatusLine())
                && httpResponse.getEntity() != null
        ) {

            String content = EntityUtils.toString(httpResponse.getEntity(), ENCODING);
            page.setStatusCode(httpResponse.getStatusLine().getStatusCode());
            page.setRaw(content);

        }
    }
```

```
        return page;
    }


    /**
     * Description: 封装请求头
     *
     * @param params
     * @param httpMethod
     */
    private static void packageHeader(final Map<String, String> params, final HttpRequestBase
httpMethod) {
        if (params == null || params.isEmpty()) {
            return;
        }

        params.entrySet().stream().forEach(e -> {
            httpMethod.setHeader(e.getKey(), e.getValue());
        });
    }

    /**
     * Description: 封装请求参数
     *
     * @param params
     * @param httpMethod
     * @throws UnsupportedEncodingException
     */
    private static void packageParam(final Map<String, String> params, final HttpEntityEnclosi
gRequestBase httpMethod)
            throws UnsupportedEncodingException {

        if (params == null || params.isEmpty()) {
            return;
        }

        final List<NameValuePair> nvps = new ArrayList<NameValuePair>();
        params.entrySet().stream().forEach(e -> {
            nvps.add(new BasicNameValuePair(e.getKey(), e.getValue()));
        });

        httpMethod.setEntity(new UrlEncodedFormEntity(nvps, ENCODING));
    }


}
```

Constant.java Http 请求的一些常用参数

```
package com.niu.req.downloader;

import lombok.AllArgsConstructor;
```

```
import lombok.Getter;

/**
 * @Description http 请求的一些常用参数
 */
public class Constant {

    @Getter
    @AllArgsConstructor
    public enum Method {
        GET("GET"),POST("POST");
        String code;
    }

    @Getter
    @AllArgsConstructor
    public enum StatusCode {
        CODE_200(200),CODE_404(404),CODE_503(503),CODE_500(500);
        Integer code;

    }

    @Getter
    @AllArgsConstructor
    public enum Header {
        REFERER("Referer"),USER_AGENT("User-Agent");
        String code;
    }
}
```

Page.java 页面实体

```
package com.niu.req.downloader;

import lombok.Getter;
import lombok.Setter;

/**
 * @Description TODO
 */
@Getter
@Setter
public class Page {

    private Request request;

    private Integer statusCode = Constant.StatusCode.CODE_500.getCode();

    private String raw;

    private Page(){ }

    public static Page build(Request request){
```

```
        Page page = new Page();
        page.setRequest(request);

        return page;
    }
}
```

Request.java 封装请求实体类

```
package com.niu.req.downloader;

import lombok.Getter;
import lombok.Setter;

import java.util.Map;

/**
 * @Description TODO
 */
@Setter
@Getter
public class Request {

    public Request(String url){
        this.url = url;
    }

    private Constant.Method method = Constant.Method.GET;

    private String url;

    private Map<String,String> headers;

    private Map<String,String> params;

}
```

Request.java 测试类

```
package com.niu.req.course01;

import com.niu.req.downloader.Client;
import com.niu.req.downloader.Page;
import com.niu.req.downloader.Request;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

/**
 * @Description 主要爬CSDN 搜索韩顺平后页面的所有文章
 */
```

```
public class RequestTest {

    public static void main(String[] args) throws Exception {
        Request request = new Request("https://so.csdn.net/so/search/s.do?q=%E9%9F%A9%E
%A1%BA%E5%B9%B3&t=%20&u=");  // url 路径

        Page page = Client.doRequest(request);

        Document parse = Jsoup.parse(page.getRaw()); //得到Html文本

        Elements select = parse.select("div.search-list-con"); //获取要得到的节点

        select.forEach( e -> {
            Elements data = e.select("dl.search-list.J_search");  // 再找到节点下的节点
            if (data == null || data.isEmpty()) {
                return;
            }
            for (Element datum : data) {
                System.out.println(datum.text());// 得到text
            }
        });
    }
}
```

搭建结束 运行Request.java 即可