## 集算器SPL抓取网页数据

网站上的数据源是我们进行统计分析的重要信息源。当我们浏览网页，看到自己感兴趣数据内容时，希望能够快速抓取网页上的数据，这对于数据分析相关工作来说极其重要，也是必备的技能之一。但是网络数据抓取大多需要复杂的编程知识，操作也比较繁琐。这里介绍如何用集算器SPL快速抓取网页数据。

1. **基本流程图**
2. **抓取网页数据接口**

**3、定义规则  
 A、web\_info**

**B、init\_url**

**C、help\_url**

**D、target\_url**

**E、page\_url**

**4、抓取股票历史数据**

**5、用户自定义程序  
 A、数据提取程序接口**

**B．数据保存程序接口**

**C、数据提取程序样例**

**D、数据保存程序样例  
 E、自定义程序的使用**

**1、基本流程图**

网址页

解析网址

下载页

提取数据

开始网址

保存数据

下载网址

结束

否

否

是

队列

是否为空

Help Url

Parse Url

Target Url

Extract Data

Start Url

Save Data

Download Url

Finished

No

No

Yes

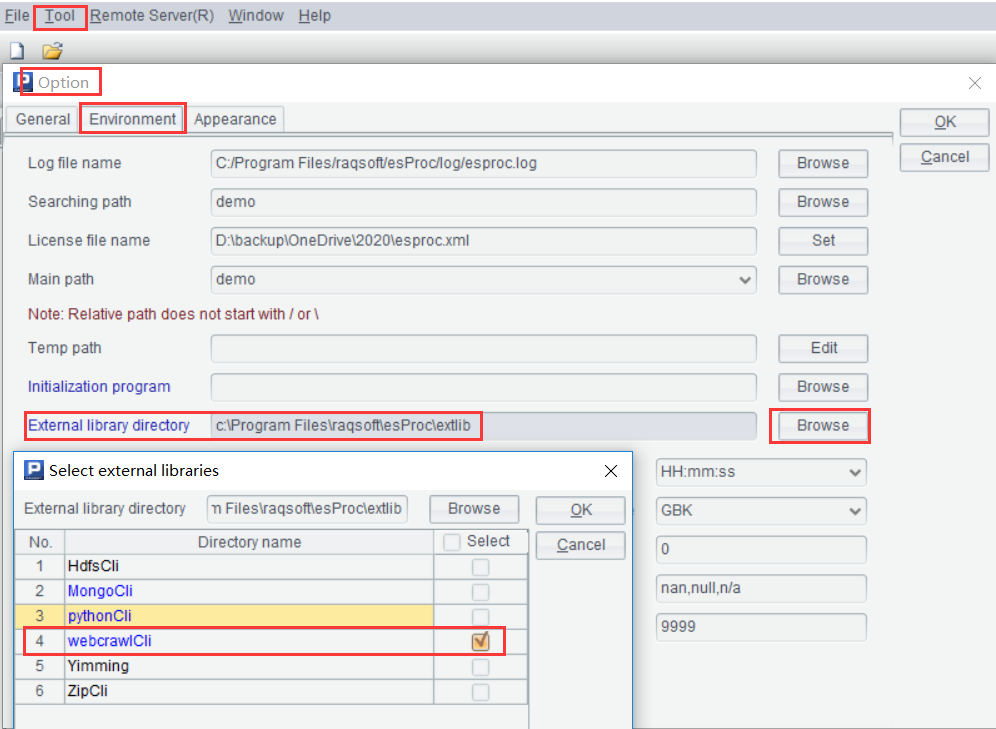
queue

is empty?

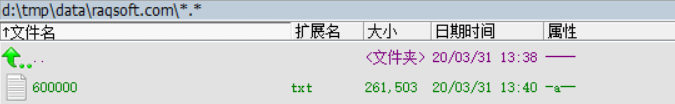
**T**raverse from the given starting URL, put the parsed and filtered URLs into the download URL queue, and divide it into help\_url and target\_url. The help\_url only collects the web URLs, and the target\_url can collect the web URLs, and can also extract the data, and save the extracted data. Crawl the web data until the traversed URL is empty, then the crawl is finished.

从给定的开始地址进行遍历，将解析过滤后的网址放入下载地址队列，分成网址页help\_url与下载页target\_url, 网址页只收集网址，下载页即能收集网址，也能提取数据，把提取到的数据保存起来。抓取网页数据直到遍历地址为空，则抓取工作结束。

**2、抓取网页数据接口**

web\_crawl(jsonstr) 是抓取网页数据接口，参数jsonstr是定义规则的字符串，抓取数据时，根据定义规则遍历URL、下载、提取、保存相关内容数据。  
 本接口依赖集算器外部库webcrawlCli。它缺省安装在集算器软件的esProc\extlib\webcrawlCli路径下，在集算器的外部库设置中勾选webcrawlCli项,重启集算器后，就可以使用web\_crawl接口。  
  
  
web\_crawl简单用法，如抓取指定股票数据，SPL脚本demo.dfx：

|  |  |
| --- | --- |
|  | **A** |
| 1 | " [{web\_info:{save\_path:'d:/tmp/data', save\_post:'false'}},  {init\_url:['http://www.aigaogao.com/tools/history.html?s=600000']},  {page\_url:{extractby: \"//div[@id='ctl16\_contentdiv']/\",class:'default'}}]" |
| 2 | =web\_crawl(A1) |
| 3 | =file("D:/tmp/data/raqsoft.com/600000.txt").import@cqt() |

获取股票代码600000的数据文件：  
   
**文件内容：**  


**3、定义规则**

根据基本流程图，将定义规则分成网站信息、初始网址、网址页、下载页、提取数据五部分。具体内容如下：

[

{**web\_info:**{domain:'www.banban.cn', save\_path:'d:/tmp/data/webmagic', thread\_size:2, cookie:{name:"jacker", laster:"2011"},

user\_agent:'Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:39.0) Gecko/20100101 Firefox/39.0'}},

{**init\_url:**['https://www.banban.cn/gupiao/list\_cybs.html', 'https://www.banban.cn/gupiao/list\_sh.html']},

{**help\_url:**['gupiao/list\_(sh|sz|cyb)\.html', '/shujv/zhangting/', '/agu/$']},

{**target\_url:**{reg\_url:'/agu/365\d'}},

{**target\_url:**{filter:'gupiao/list\_(sh|sz|cyb)\.html', reg\_url:'gupiao/[sz|sh]?(60000\d)/',new\_url:'http://www.aigaogao.com/tools/history.html?s=%s'}},

{**page\_url:**{filter:'history.html\?s=\d{6}', extractby: "//div[@id='ctl16\_contentdiv']/ "}},

{**page\_url:**{extractby: "//div[@id='content\_all']/ "}},

{**page\_url:**{filter:'/agu/365\d', extractby: "//div[@id='content']/ "}}

]  
**规则简要说明：**  
**web\_info：网站信息，**根据要下载的网站，设置域名、本地存储位置、用户代理信息、用户自定义程序等相关的信息。

**init\_url：初始网址**，URL遍历的入口网址。  
**help\_url：网址页，**定义网址页规则，收集网页内容中的URL，但不提取此页面数据内容。  
**target\_url：下载页，**定义下载页规则**，**收集网页内容中的URL，同时也提取此页面的内容。  
**page\_url：提取数据，**定义页面内容提取规则，在下载页target\_url中根据此规则提取内容。

**注意：**json书写结构细节，节点{}中的[]表示list列表，节点{}中的{}表示map键值结构，书定时要注意，否则书写不对易引起解析错误。

**定义规则说明**

A、web\_info

设置要下载的信息，内容包括：

domain：设置域名。

save\_path：文件存储路径。

user\_agent：指用户代理信息。 作用:使服务器能够识别客户使用的操作系统及版本、CPU 类型、浏览器及版本、浏览器渲染引擎、浏览器语言、浏览器插件等。

sleep\_time：抓取间隔。

cycle\_retry\_times：重试次数。

charset：设置编码。

use\_gzip：是否为gzip压缩。

time\_out：抓取超时设置。

cookie\_name：cookie信息，键值结构。

thread\_size：抓取时线程数。  
save\_post：是否要为存储的文件名称追加编码串，以防网名文件被覆盖，缺省值为true。如books/a.html, music/a.html,都是要下载的页面，保存时若此参数为true,则存储文件名分别为a\_xxxcgk.txt,a\_xabcdw.txt，文件不会被覆盖；若为false,保存文件名为a.txt,后存储的就会将已存在的同名文件覆盖。

class\_name：用户自定义的存储类。

class\_argv：传递给class\_name类的字符串参数。

B、init\_url

初始的URL。

为List列表结构，可设置多个URL.

C、help\_url

网址页主要是定义要收集的URL过滤规则, 符合规则的URL会被加入下载网址队列，但是不会提取其具体内容。过滤规则支持正则表达式，如：  
gupiao/list\_(sh|sz|cyb)\.html 表示URL中只有包括字符串gupiao/list\_sh.html、gupiao/list\_sz.html、gupiao/list\_cyb.html链接才能通过。

为List列表结构，可定义多个规则。

D、target\_url

下载页是要抓取内容数据的URL，需要从这个页面里提取内容。若此URL符合help\_url过滤规则，那么也会在本页面中收集URL。

约定定义规则格式:  
{**target\_url:**{**filter**: pageUrl, **reg\_url**:urlRegex, **new\_url**:newUrl}},  
表示在符合pageUrl条件的页面中，找出符合urlRegex条件的href链接，若newUrl定义了，则可与urlRegex过滤结果组合成新的URL。  
例如在页面中找到链接a\_100.html符合过滤条件reg\_url=a\_(\d+)\.html,则有newUrl=b\_%s.php,那么urlRegex过滤a\_100.html的结果为100,将与newUrl合并，新的下载页为b\_100.php。

其中filter表示定义过滤的URL规则；若无此定义，表示所有的target\_url都要用此规则。  
reg\_url表示要收集的URL规则，必写；无reg\_url的target\_url规则则无意义。  
new\_url表示定义新的页面，需要与reg\_url过滤结果结合成新的URL。

**举例说明：**  
**3.1** 定义规则：{**target\_url:**{filter:'gupiao/list\_(sh|sz|cyb)\.html', reg\_url:'gupiao/([sz|sh]?6000\d{2})/',new\_url:'http://www.raqsft.com/history.html?s=%s'}}  
在下载页gupiao/list\_sh.html中包含如下内容：  
<li><a href="/gupiao/600010/">包钢股份(600010)</a></li>

<li><a href="/gupiao/600039/">四川路桥(600039)</a></li>

<li><a href="/gupiao/600048/">保利地产(600048)</a></li>

A、gupiao/list\_sh.html符合filter条件  
B、href串符合reg\_url条件，将产生[600010, 600039, 600048]  
C、过滤结果与newUrl生成新的URL：  
http://www.raqsft.com/history.html?s=600010  
http://www.raqsft.com/history.html?s=600039  
http://www.raqsft.com/history.html?s=600048

new\_url中的%s为合并字符串的占位符。  
  
**3.2** 定义规则：{**target\_url:**{reg\_url:'/ gupiao/60001\d'}},

在下载页gupiao/list.html中包含如下内容：  
<li><a href="/gupiao/600010/">包钢股份(600010)</a></li>

<li><a href="/gupiao/600039/">四川路桥(600039)</a></li>

<li><a href="/gupiao/600048/">保利地产(600048)</a></li>

href中符合reg\_url条件的，则收集到的URL为：

http://www.xxx.com/gupiao/600010/  
其它两个href不符合过滤条件。  
设置filter是为了在过滤后的页面中去收集URL,当help\_url多时，过滤后缩小了范围，提高了效率。

target\_url规则可定义多条，以适应不同的条件。

E、page\_url

提取数据，主要作用于下载页面内容提取，它表示使用这个抽取规则，将提取到的结果保存。定义此规则参考xpath,jsonpath使用说明。它只提取主要内容，但对内容细节还需要className类来抽取。  
约定定义规则格式:  
{**page\_url:**{**filter**: pageUrl, **extractby**: contentReg, **class**: className }},  
其中filter表示符合过滤条件的url规则，若无此定义，表示所有的target\_url都要用此规则。  
extractby表示页面内容提取规则。若定义class，表示由className类执行内容提取；若className=  
”default”, 表示用当前缺省方式提取，也就是针对table表中的内容提取数据。若缺省提取不满足需求，用户可自定义类来实现，具体实现参考后面**用户自定义程序。**  
  
例如：extractby :"//div[@class=news-content]/text()"，从网页中提取此节点下的数据。

page\_url可针对不同的页面制定不同的规则。通过filter过滤后的页面中去提取数据，减少要处理的URL数量，当target\_url多时，能提高效率。

若无对应的extractby规则，则表示提取target\_url页面中所有的内容。  
若定义了多条page\_url规则 ，则首个符合规则的内容将被提取。  
假如A页面内容的符合规则R1,R2,R3, 提取内容时首先是R2，则不再根据R1、R3规则提取数据。  
说明：若没有定义target\_url规则，但当前页面有适合的page\_url规则，则此页面的内容也会被提取。

4、抓取股票历史数据

下面用抓取股票历史数据来说明，web\_crawl()接口是如何应用的。基本操作：先获取股票代码，然后通过股票代码查询历史数据，从下载页面中提取数据后保存。A、在https://www.banban.cn/gupiao/list\_xxx.html页面help\_url提取上证、深证、创业板的股票代码。  
B、将股票代码与http://www.aigaogao.com/tools/history.html?s=%s结合，生成需要下载网址target\_url.  
C、针对下载页target\_url中的内容提取。

D、显示提取后的内容。

SPL实现代码Stock.dfx：

|  |  |  |
| --- | --- | --- |
|  | **A** | **B** |
| **1** | "[{web\_info:{domain:\"www.banban.cn\", save\_path:\"d:/tmp/data/webcrawl\",  save\_post: \"false\"，thread\_size:2 }},  {init\_url:[\"https://www.banban.cn/gupiao/list\_cybs.html\", \"https://www.banban.cn/gupiao/list\_sh.html\"]},  {help\_url:[\"gupiao/list\_(sh|sz|cyb)\.html\"]},  {target\_url:{filter:\"gupiao/list\_(sh|sz|cyb)\.html\", reg\_url:'gupiao/[sz|sh]?( \d{6})/',new\_url:\"http://www.aigaogao.com/tools/history.html?s=%s\"}},  {page\_url:{filter:\"history.html\?s=\d{6}\", extractby:\"//div[@id='ctl16\_contentdiv']/\"}},  {page\_url:{extractby:\"//div[@id='content\_all']/\"}}] " |  |
| **2** | =web\_crawl(A1) |  |
| **3** | =file("D:/tmp/data/ webcrawl/www.banban.cn/600010.txt").import@cqt() |  |

加载其中的股票600010数据为：

**5、用户自定义程序**

对于内容提取，缺省提供了对html中的table内容进行抽取。 但是世界上没有千篇一律的网页一样，也没有一劳永逸的提取算法。在使用网页数据抓取过程中，你会碰到各种类型的网页，这个时候，你就要针对这些网页，来实现对应抽取方法。存储方式类似，缺省提供的是文件保存，若想其它方式如数据库存储，还需要用户自己开发程序。参考下面接口，可将自定义程序融入网页数据抓取流程中。

**A、数据提取程序接口**

下载页的内容组织形式多样，各具不同，为了适应更多的内容提取需求，用户可自定义提取数据程序。  
接口程序:  
package com.web;

**import** us.codecraft.webmagic.Page;

**public** **interface** StandPageItem {

**//数据提取处理。**

**void** parse(Page p);

}

需要实现com.web.StandPageItem接口parse(Page p)，数据提取在此实现。

B、数据保存程序**接口**

提取数据存储方式种类繁多，各具不同，为了适应更多的数据存储需求，用户可自定义数据存储程序。  
接口程序:  
**package com.web;**

**import us.codecraft.webmagic.ResultItems;**

**import us.codecraft.webmagic.Task;**

**import us.codecraft.webmagic.pipeline.Pipeline;**

**public** **interface** StandPipeline **extends** Pipeline {

**public** **void** setArgv(String argv);

**public** **void** process(ResultItems paramResultItems, Task paramTask);

}  
同样需要实现com.web.StandPipeline类中的setArgv(), process()。

setArgv()输入参数接口，process()处理存储数据接口。

C、数据提取程序样例

实现com.web.StandPage接口parse(Page p)，   
参考代码：

package com.web;

import java.util.List;

import us.codecraft.webmagic.Page;

import us.codecraft.webmagic.selector.Selectable;

public class StockHistoryData implements StandPageItem{

@Override

public void parse(Page page) {

StringBuilder buf = new StringBuilder();

List<Selectable> nodes = page.getHtml().xpath("table/tbody/").nodes();

for(Selectable node:nodes){

String day = node.xpath("//a/text()").get();

List<String> title = node.xpath("//a/text() | tr/td/text() ").all();

if (title.size()<5) continue;

String line = title.toString().replaceFirst(", , ", ", ");

buf.append(line+"\n");

}

page.putField("content", buf.toString());

}

}

将要保存的数据存放到page的字段"content"中，在保存处理时，将从字段"content"中获取。

**D、数据保存程序**样例

实现com.web.StandPageline类中的setArgv(),process()  
参考代码：  
package com.web;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** us.codecraft.webmagic.ResultItems;

**import** us.codecraft.webmagic.Task;

**import** org.apache.commons.codec.digest.DigestUtils;

import us.codecraft.webmagic.utils.FilePersistentBase;

**public** **class** StockPipeline extends FilePersistentBase implements StandPipeline {

**private** Logger logger = LoggerFactory.*getLogger*(getClass());

**private** String m\_argv;

**private** String m\_path;

**public** **static** String *PATH\_SEPERATOR* = "/";

**static** {

String property = System.*getProperties*().getProperty("file.separator");

**if** (property != **null**) {

*PATH\_SEPERATOR* = property;

}

}

**public** StockPipeline() {

m\_path = "/data/webcrawl";

}

//获取存储路径与存储文件名前缀

**public** **void** setArgv(String argv) {

m\_argv = argv;

**if** (m\_argv.indexOf("save\_path=")>=0){

String[] ss = m\_argv.split(", ");

m\_path = ss[0].replace("save\_path=", "");

m\_argv = ss[1];

}

}

**public** **void** process(ResultItems resultItems, Task task) {

String saveFile = **null**;

Object o = **null**;

String path = **this**.m\_path + *PATH\_SEPERATOR* + task.getUUID() + *PATH\_SEPERATOR*;

**try** {

**do**{

String url = resultItems.getRequest().getUrl();

o = resultItems.get("content");

**if** (o == **null**){

**break**;

}

**int** start = url.lastIndexOf("/");

**int** end = url.lastIndexOf("?");

**if** (end<0){

end=url.length();

}

String link = url.substring(start+1, end);

**if** (m\_argv!=**null** && !m\_argv.isEmpty()){

link = m\_argv+"\_"+link;

}

**if** (link.indexOf(".")>=0){

link = link.replace(".", "\_");

}

//加md5Hex为防止重名

String hex = DigestUtils.*md5Hex*(resultItems.getRequest().getUrl());

saveFile = path + link+"\_"+ hex +".json";

}**while**(**false**);

**if** (saveFile!=**null**){

PrintWriter printWriter = **new** PrintWriter(**new** FileWriter(getFile(saveFile )));

printWriter.write(o.toString());

printWriter.close();

}

} **catch** (IOException e) {

logger.warn("write file error", e);

}

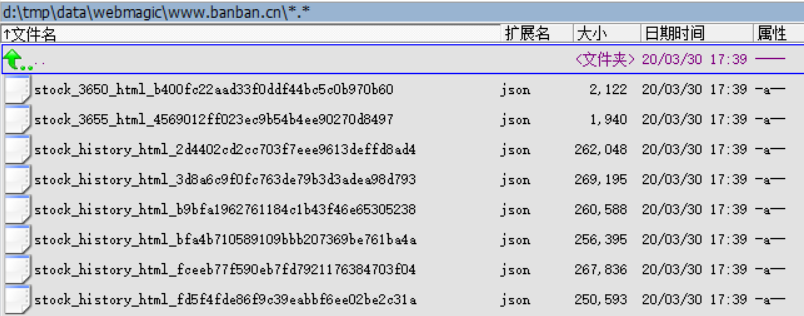
}

}

E、自定义程序的使用

将上述接口文件及java文件编译后打包成webStock.jar文件，放在esProc\extlib\webcrawlCli路径下重启集算器。数据存储程序，在web\_info中配置；数据提取程序，在page\_url中配置。下面是加载两个自定义类程序的dfx脚本。  
  
SPL代码脚本mytest.dfx:

|  |  |  |
| --- | --- | --- |
|  | **A** | **B** |
| **1** | "[{web\_info:{domain:\"www.banban.cn\", save\_path:\"d:/tmp/data/webmagic\", thread\_size:2,  user\_agent:\"Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:39.0) Gecko/20100101 Firefox/39.0\",  class\_name:'com.web.StockPipeline',class\_argv:'stock'}},  {init\_url:[\"https://www.banban.cn/gupiao/list\_cybs.html\", \"https://www.banban.cn/gupiao/list\_sh.html\"]},  {help\_url:[\"gupiao/list\_(sh|sz|cyb)\.html\", \"/shujv/zhangting/\"]},  {target\_url:{filter:\"gupiao/list\_(sh|sz|cyb)\.html\", reg\_url:'gupiao/[sz|sh]?( \d{6})/',new\_url:\"http://www.aigaogao.com/tools/history.html?s=%s\"}},  {page\_url:{filter:\"history.html\?s=\d{6}\", extractby:\"//div[@id='ctl16\_contentdiv']/\",  class:'com.web.StockHistoryData'}}]" |  |
| **2** | =web\_crawl(A1) |  |
|  |  |  |

**生成结果：**  
  
类似针对内容提取或数据存储，参考上面程序的实现，用户可自定义java程序。