

**软件工程综合课程设计**

--详细设计

**题 目** 中标统计分析系统

**院 系**  计算机科学与技术学院

**专 业**  软件工程

**学生姓名** 吴未名,王祥,曹立群,张桔,柳鹏磊

**学 号**  161330119

**指导教师** 张德平

**二0 一五 年 十二 月 二十五 日**

目录

[1. 引言 3](#_Toc438761619)

[1.1. 编写目的 3](#_Toc438761620)

[1.2. 背景 3](#_Toc438761621)

[1.3. 参考资料 3](#_Toc438761622)

[2. 架构详细设计 4](#_Toc438761623)

[2.1. 网络爬虫管理 4](#_Toc438761624)

[2.2. 爬虫数据抓取 5](#_Toc438761625)

[2.3. 爬虫数据存储与查询 5](#_Toc438761626)

[3. 各功能详细设计 6](#_Toc438761627)

[3.1. 中标数据爬虫 6](#_Toc438761628)

[3.2. 正则表达式处理 8](#_Toc438761629)

[3.3. 爬虫多线程的设计与实现 13](#_Toc438761630)

[3.4. 数据库连接与操作 13](#_Toc438761631)

[3.5. 基本页面元素输出 17](#_Toc438761632)

# 引言

## 编写目的

此详细设计说明书是为了说明整个系统的体系架构，以及需求用例的各个功能点在架构中的体现，为系统的详细设计人员进行详细设计时的输入参考文档。

## 背景

软件系统的名称：中标统计分析系统

开发工具：Python、PHP、Scrapy

开发者：吴未名、王祥、曹立群、张桔、柳鹏磊

## 参考资料

1. 刘世涛等，简析搜索引擎中网络爬虫的搜索策略，阜阳师范学院学报 (自然科学版)
2. 吴小竹等，基于Python的多线程SPIDER的设计与实现
3. 胡宏涛等，基于网络的信息获取技术浅析
4. 李学勇等，网络蜘蛛搜索策略比较研究
5. 高克宁等，支持Web信息分类的高性能蜘蛛程序
6. 吴强等，Python线程，Reactor平台程序设计和实战
7. 荣传湘等，搜索引擎中数据获取的设计与实现
8. Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, etal. Focused Craw ling using Context Graph s
9. 叶允明等，分布式Web Crawler 的研究: 结构、算法和策略

# 架构详细设计

## 网络爬虫管理

Rule配置文件

Rule配置文件列表

临

界

区

互联网

线程1搜索元URL如www.nkkj.gov.cn

线程2搜索元URL如www.bjgp.gov.cn

线程N

图 1多线程网络爬虫概要设计图模型

在这个模型中，Rule是整个系统的核心。结合之前的概要设计文档可知，Rule包含了爬虫初始化所需的全部信息，包括开始地址、各部分的XPath、允许域名等。

而多个Rule又构成了“Rule配置文件列表”，其中存储了多个Rule，可以在初始化时生成多个爬虫。

利用多线程技术，每个爬虫各司其职，针对不同的网站进行爬取操作，而用户只需要修改Rule即可生成或者修改爬虫。

## 爬虫数据抓取



图 2网络爬虫工作流程图

爬虫数据抓取的过程是一个有限条件下的循环问题。

首先，从配置文件中读取源URL，访问这个URL，并在打开的页面信息中利用Rule提供的Xpath等规则匹配对应的部分。接着，将例如“下一页”等的URl再次添加进爬取的列表，实现自动翻页的效果。然后，针对每一次深入抓取的数据，利用正则表达式等方式进行匹配，提取出中标人、中标时间、中标金额等有用信息，并根据情况写入数据库。

## 爬虫数据存储与查询

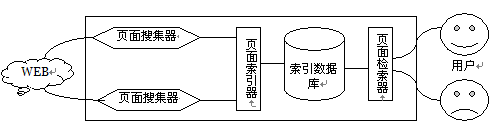


图 3爬虫搜索存储组成示意图

在成功进行数据抓取后，爬虫程序将对应的数据写入数据库，并在预先计划的字段上添加索引信息，便于日后的查询和修改。用户通过检索器与数据库进行通信。

# 各功能详细设计

## 中标数据爬虫



图 4 中标数据爬虫流程图

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | parse\_page | | |
| 功能概要 | 解析状态信息 | | |
| 记述形式 | def parse\_page(self, response) | | |
| 参数 | self, response | | |
| 类型 | 变量名 | I/O | 说明 |
| Object | response | I | 指向一个http返回数据 |

**def** parse\_page(self, response):  
 p = PageItem()  
 p[**'currentPage'**] = response.url  
 p[**'name'**] = self.name  
 p[**'id'**] = self.id  
 **return** p

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | parse\_item | | |
| 功能概要 | 解析中标内容、标题 | | |
| 记述形式 | def parse\_item(self, response): | | |
| 参数 | self, response | | |
| 类型 | 变量名 | I/O | 说明 |
| Object | response | I | 指向一个http返回数据 |

**def** parse\_item(self, response):  
 i = Article()  
 i[**'title'**] = response.xpath(self.rule.titlerule).extract()[0]  
 i[**'content'**] = response.xpath(self.rule.contentrule).extract()[0]  
 i[**'spider'**] = self.name  
 i[**'currentURL'**] = response.url  
 i[**'id'**] = self.id  
 **return** i

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | \_\_init\_\_ | | |
| 功能概要 | 爬虫开始函数 | | |
| 记述形式 | def \_\_init\_\_(self, rule): | | |
| 参数 | self, response | | |
| 类型 | 变量名 | I/O | 说明 |
| Object | Rule | I | 一个匹配规则对象 |

**def** \_\_init\_\_(self, rule):  
 self.rule = rule  
 self.name = rule.name  
 self.id = rule.id  
 self.allowed\_domains = rule.alloweddomain.split(**","**)  
 *# print(rule.starturl)* self.start\_urls = [rule.starturl]  
 rule\_list = []  
 *#添加`下一页`的规则* **if** rule.nextpagerule:  
 rule\_list.append(Rule(LinkExtractor(  
 restrict\_xpaths = rule.nextpagerule),  
 callback=**'parse\_page'**,  
 follow=True))  
 *#添加抽取文章链接的规则* rule\_list.append(Rule(LinkExtractor(  
 restrict\_xpaths = [rule.articlelinkrule]),  
 callback=**'parse\_item'**))  
 self.rules = tuple(rule\_list)  
 super(BeijingSpider, self).\_\_init\_\_()

## 正则表达式处理

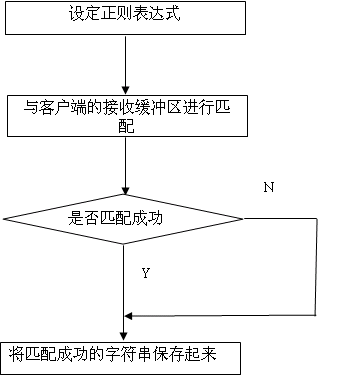


图 5正则表达式使用流程图

Label = **"<[^>]+>"**Title = **"(招标.\*)\*中标.\*|成交.\*"**.decode(**'utf8'**)  
Date1 = **"(中|定)标[^：:]\*\\s\*(：|:)\\s\*((\\d+\\s\*年\\s\*\\d+\\s\*月\\s\*\\d+\\s\*日)|(\\d+(-|/)\\s\*\\d{1,2}\\s\*(-|/)\\s\*\\d{1,2}))"**.decode(**'utf8'**)  
Date2 = **"(\\d+\\s\*年\\s\*\\d+\\s\*月\\s\*\\d+\\s\*日)|(\\d+(-|/)\\s\*\\d{1,2}\\s\*(-|/)\\s\*\\d{1,2})"**.decode(**'utf8'**)  
Orga = **"(成交|中标)[^0-9a-zA-Z]+(人|名称|单位)\\s\*(：|:)\\s\*[^\\s]+"**.decode(**'utf8'**)  
Money = **"(((人民币)|(RMB)|$|￥)\\s\*\\d+\\s\*((,|，)\\s\*\\d+\\s\*)\*(\\.\\d+)\*万\*元\*)|(\\d+\\s\*((,|，)\\s\*\\d+\\s\*)\*(\\.\\s\*\\d+\\s\*)\*万\*元)"**.decode(**'utf8'**)

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterLabel | | |
| 功能概要 | 匹配label信息 | | |
| 记述形式 | **def** filterLabel(self, content\_text): | | |
| 参数 | self, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 一个匹配规则对象 |

*# 去除标签***def** filterLabel(self, content\_text):  
 **return** re.sub(self.Label, **" "**, content\_text)

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterBlank | | |
| 功能概要 | 匹配空白信息 | | |
| 记述形式 | **def** filterBlank (self, content\_text): | | |
| 参数 | self, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 一个匹配规则对象 |

**def** filterBlank(self, content\_text):   
 **return ""**.join(content\_text.split())

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterTitle | | |
| 功能概要 | 匹配标题信息 | | |
| 记述形式 | **def** filterTitle (self, content\_text): | | |
| 参数 | self, title | | |
| 类型 | 变量名 | I/O | 说明 |
| String | title | I | 爬取的标题 |

*# 返回项目名称* **def** filterTitle(self, title):  
*# if "候选人".decode('utf8') in title:  
# return ""* cont = self.getSubString(self.Title, title)  
   
 **if** cont:  
 **if ")" in** cont:  
 cont = re.sub(**"\\)"**, **"\\)"**, cont)  
 **if "(" in** cont:  
 cont = re.sub(**"\\("**, **"\\("**, cont)  
 **return** re.sub(cont, **""**, title)  
 **return** cont

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterDate | | |
| 功能概要 | 匹配日期信息 | | |
| 记述形式 | **def** filterDate (self, content\_text): | | |
| 参数 | self, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 一个匹配规则对象 |

*# 返回项目中标日期***def** filterDate(self, content\_text):  
 str = self.getSubString(self.Date1, content\_text)  
 **if** str:  
 date = self.getAllSubString(**"\\d+"**.decode(**'utf8'**), str)  
 date = map(int , date)  
 **return** date  
 **else**:  
 str = self.getSubString(self.Date2, content\_text)  
 **if not** str:  
 **return** str  
 date = self.getAllSubString(**"\\d+"**.decode(**'utf8'**), str)  
 date = map(int, date)  
 **return** date

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterOrg | | |
| 功能概要 | 匹配中标人信息 | | |
| 记述形式 | **def** filterOrg (self, content\_text): | | |
| 参数 | self, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 一个匹配规则对象 |

*# 返回项目中标单位***def** filterOrg(self, content\_text):  
 str = self.getSubString(self.Orga, content\_text)  
 **if not** str:  
 **return** str  
 **else**:  
 **return** re.sub(**".\*(:|：)"**.decode(**'utf8'**), **""**, str)

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterMoney | | |
| 功能概要 | 匹配中标金额信息 | | |
| 记述形式 | **def** filterMoney (self, content\_text): | | |
| 参数 | self, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 一个匹配规则对象 |

**def** filterMoney(self, content\_text):  
 s = self.getSubString(self.Money, content\_text)  
 wan = False  
 **if not** s:  
 **return** s  
*# print s* s = self.getSubString(**"\\d+((,|，)\\d+)\*(\\.\\d+)\*万\*"**.decode(**'utf8'**), s)  
 **if not** s:  
 **return** s  
 **if ","**.decode(**'utf8'**) **in** s **or "，"**.decode(**'utf8'**) **in** s:  
 s = re.sub(**","**, **""**, s)  
 s = re.sub(**"，"**.decode(**'utf8'**), **""**, s)  
 **if "万"**.decode(**'utf8'**) **in** s:  
 wan = True  
 s = re.sub(**"万"**.decode(**'utf8'**), **""**, s)  
 s = string.atof(s)  
 **if** wan:  
 s \*= 10000  
 **return** s

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | getSubString | | |
| 功能概要 | 获取正则表达式匹配的字串 | | |
| 记述形式 | **def** getSubString (self, content\_text): | | |
| 参数 | self, regex, content | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content | I | 爬取的中标信息内容 |

*# 根据正规式返回第一个子串***def** getSubString(self, regex, content):  
 s = re.search(regex, content)  
 **if** s:  
 **return** s.group()  
 **else** :  
 **return** s

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | getAllSubString | | |
| 功能概要 | 获取正则表达式匹配的全部子串 | | |
| 记述形式 | **def** getAllSubString (self, content\_text): | | |
| 参数 | self, regex, content | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content | I | 爬取的中标信息内容 |

*# 根据正规式返回所有满足条件的子串***def** getAllSubString(self, regex, content):  
 pattern = re.compile(regex)  
 arr = pattern.findall(content)  
 **return** arr

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | filterResult | | |
| 功能概要 | 匹配label信息 | | |
| 记述形式 | **def** filterResult(self, title, content\_text): | | |
| 参数 | self, title, content\_text | | |
| 类型 | 变量名 | I/O | 说明 |
| String | content\_text | I | 爬取的中标信息内容 |
| String | title | I | 爬取的中标信息标题 |

*# 获取结果:左边参数为标题，右边参数为具体内容,返回4个字段[项目名称,项目中标单位,中标金额,[年,月,日]]***def** filterResult(self, title, content\_text):  
 data = [0, 0, 0, 0]  
 content\_text = self.filterBlank(content\_text)  
 content\_text = self.filterLabel(content\_text)  
 Tag\_Money = False  
 Tag\_Date = False  
 Tag\_Org = False  
 Tag\_Name = False  
 data[0] = self.filterTitle(title)  
 data[1] = self.filterOrg(content\_text)  
 data[2] = self.filterMoney(content\_text)  
 data[3] = self.filterDate(content\_text)  
 **if** data[0]:  
 Tag\_Name = True  
 **if** data[1]:  
 Tag\_Org = True  
 **else** :  
 **return** []  
 **if** data[2]:  
 Tag\_Money = True  
 **else** :  
 **return** []  
 **if** data[3]:  
 Tag\_Date = True  
 **else** :  
 **return** []  
 **if** Tag\_Money **and not** Tag\_Name:  
 data[0] = title  
 **if** Tag\_Money **and** Tag\_Date **and** Tag\_Org **and** Tag\_Name:  
 **return** data  
 **return** []

## 爬虫多线程的设计与实现

为了使网络爬虫的效率大幅提高，需要设计多线程函数，具体实现如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名 | Rule | | |
| 功能概要 | 开始多线程爬取操作 | | |
| 记述形式 | **def** Rule (self,Rules): | | |
| 参数 | self,Rules | | |
| 类型 | 变量名 | I/O | 说明 |
| List | Rules | I | 爬取规则的集合 |

settings.set(**"ITEM\_PIPELINES"** , {  
 **'pipelines.SeprojectPipeline'**: 300  
})  
  
  
db = DBSession()  
rules = db.query(Rule).filter(Rule.enable == 1)  
  
**for** rule **in** rules:  
 crawler = Crawler(settings)  
 spider = BeijingSpider(rule)  
 RUNNING\_CRAWLERS.append(spider)  
 crawler.signals.connect(spider\_closing, signal=signals.spider\_closed)  
 crawler.configure()  
 crawler.crawl(spider)  
 crawler.start()  
  
reactor.run()

说明：

每一个线程调用一个爬虫程序，以实现不同源URL同时进行搜索。

调用crawler.crawl(spider)函数创建线程：

调用reactor.run()函数随机获取生成线程的ID号以区别不同的线程。

## 数据库连接与操作

使用PHP连接Mysql数据库是B/S架构中最基础的部分，主要的步骤是：



|  |  |
| --- | --- |
| 函数名 | \_\_construct |
| 功能概要 | 初始化数据库 |
| 记述形式 | **def** filterLabel(self, content\_text): |
| 参数 | Null |

**<?php***/\*\*  
 \* Created by PhpStorm.  
 \* User: Wu  
 \* Date: 11/29/15  
 \* Time: 2:26 PM  
 \*/***class** DataBaseEngine{  
 **private $db**;  
 **function** \_\_construct(){  
 $this->**db** = **new** PDO(**'mysql:host=localhost;dbname=SE;charset=utf8'**, **'root'**, **'zfAQpBluc52s'**);  
 $this->**db**->setAttribute(PDO::***ATTR\_EMULATE\_PREPARES***, **false**); }  
 **function** query($sql)  
 {  
 $stmt = $this->**db**->query($sql);  
 **return** $stmt;  
 }  
 **function** getRawResultData($start, $span){  
 **return** $this->query(**"SELECT \* FROM RawData"**);  
 }  
 **function** getScrawlSettings(){  
 **return** $this->query(**"SELECT \* FROM rules"**);  
 }  
 **function** getCrawlerStatus(){  
 **return** $this->query(**"SELECT \* FROM status"**);  
 }  
 **function** modifyCrawlSettings($id,$name,$startURL,$domain,$link,$next,$title,$content,$enable){  
 $sql=**"UPDATE `rules` SET `name`=?,`alloweddomain`=?,`starturl`=?,`nextpagerule`=?,`articlelinkrule`=?,`titlerule`=?,`contentrule`=?,`enable`=? WHERE `id`=?"**;  
 $stmt = $this->**db**->prepare($sql);  
 $exeres = $stmt->execute(**array**($name, $domain, $startURL, $next, $link, $title, $content, $enable, $id));  
 }  
 **function** search($startdate,$enddate,$startmoney,$endmoney){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE date >= '**$startdate**' and date <= '**$enddate**' and money >=** $startmoney **and money <=** $endmoney**"**;  
 **return** $this->query($sql);  
 }  
 **function** searchTitle($title,$startdate,$enddate,$startmoney,$endmoney){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE date >= '**$startdate**' and date <= '**$enddate**' and money >=** $startmoney **and money <=** $endmoney **and title like '%**$title**%'"**;  
 **return** $this->query($sql);  
 }  
 **function** searchContent($content,$startdate,$enddate,$startmoney,$endmoney){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE date >= '**$startdate**' and date <= '**$enddate**' and money >=** $startmoney **and money <=** $endmoney **and content like '%**$content**%'"**;  
 **return** $this->query($sql);  
 }  
 **function** searchOrg($org,$startdate,$enddate,$startmoney,$endmoney){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE date >= '**$startdate**' and date <= '**$enddate**' and money >=** $startmoney **and money <=** $endmoney **and org like '%**$org**%'"**;  
 **return** $this->query($sql);  
 }  
 **function** addRule($name,$startURL,$domain,$link,$next,$title,$content,$enable){  
 $sql=**"INSERT INTO `rules` (`name`,`alloweddomain`,`starturl`,`nextpagerule`,`articlelinkrule`,`titlerule`,`contentrule`,`enable`) VALUES (?,?,?,?,?,?,?,?)"**;  
 $stmt = $this->**db**->prepare($sql);  
 $exeres = $stmt->execute(**array**($name, $domain, $startURL, $next, $link, $title, $content, $enable));  
 }  
 **function** deleteRule($id){  
 $sql = **"DELETE FROM `rules` WHERE id=?"**;  
 $stmt = $this->**db**->prepare($sql);  
 $exeres = $stmt->execute(**array**($id));  
 }  
 **function** selectUser($userName){  
 $sql = **"SELECT** *\** **FROM `user` WHERE username = '**$userName**'"**;  
 **return** $this->query($sql);  
 }  
 **function** addUser($userName,$password){  
 $sql = **"INSERT INTO `user` (`username`,`password`) VALUES (?,?)"**;  
 $stmt = $this->**db**->prepare($sql);  
 $exeres = $stmt->execute(**array**($userName,$password));  
 }  
 **function** selectByOrg($org){  
 $sql = **"SELECT DISTINCT(org) FROM `RawData` WHERE org LIKE '%**$org**%'"**;  
 **return** $this->query($sql);  
 }  
 **function** selectDataByOrg($org){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE org = '**$org**' ORDER BY date ASC"**;  
 **return** $this->query($sql);  
 }  
 **function** selectContentById($id){  
 $sql = **"SELECT** *\** **FROM `RawData` WHERE id=**$id**"**;  
 **return** $this->query($sql);  
 }  
 **function** selectStatus1(){  
 $sql = **"SELECT** *\** **FROM `status` ORDER BY timestamp DESC"**;  
 **return** $this->query($sql);  
 }  
 **function** modifyUserInfoById($userID,$password){  
 $sql = **"UPDATE `user` SET password=? WHERE id=?"**;  
 $stmt = $this->**db**->prepare($sql);  
 $stmt->execute(**array**($password,$userID));  
 }  
 **function** selectSumData($startdate,$enddate){  
 $sql=**"SELECT org,***SUM***(money) as summoney FROM `RawData` WHERE date >= '**$startdate**' and date <= '**$enddate**' group BY org ORDER BY summoney DESC"**;  
 **return** $this->query($sql);  
 }  
 **function** selectAllUser(){  
 $sql = **"SELECT** *\** **FROM `user`"**;  
 **return** $this->query($sql);  
 }  
 **function** modifyUser($id,$name,$type){  
 $sql = **"UPDATE `user` SET `username`=?,`type`=? WHERE `id`=?"**;  
 $stmt = $this->**db**->prepare($sql);  
 $exeres = $stmt->execute(**array**($name,$type,$id));  
 **return** $stmt;  
 }  
}

## 基本页面元素输出

|  |  |
| --- | --- |
| 函数名 | printHead |
| 功能概要 | 输出html的头信息 |
| 记述形式 | **function** printHead() |
| 参数 | Null |

**function** printHead()  
 {  
 **echo <<<HTML  
 <meta charset="UTF-8">  
 <title>数据查询</title>  
 <!-- Latest compiled and minified CSS -->  
 <link rel="stylesheet" href="./css/bootstrap.min.css">  
  
 <!-- Optional theme -->  
 <link rel="stylesheet" href="./css/bootstrap-theme.min.css">  
  
 <!-- Latest compiled and minified JavaScript -->  
 <script src="./js/jquery-2.1.4.min.js"></script>  
 <script src="./js/bootstrap.min.js"></script>  
 <link rel="stylesheet" type="text/css" href="css/datatables.min.css"/>  
 <link href="css/dashboard.css" rel="stylesheet">  
 <script type="text/javascript" src="js/datatables.min.js"></script>  
HTML**;  
 }

|  |  |
| --- | --- |
| 函数名 | printNavBar |
| 功能概要 | 输出导航栏 |
| 记述形式 | **function** printNavBar() |
| 参数 | Null |

**function** printNavBar()  
 {  
 **echo <<<HTML  
 <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">  
 <div class="navbar-header">  
  
 <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1">  
 <span class="sr-only">Toggle navigation</span><span class="icon-bar"></span><span class="icon-bar"></span><span class="icon-bar"></span>  
 </button> <a class="navbar-brand" href="index.php">软件工程课程设计</a>  
 </div>  
  
 <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">  
 <ul class="nav navbar-nav">  
 </ul>  
HTML**;  
 **if** (**isset**($\_SESSION[**"id"**])) {  
 **echo <<<HTML  
 <ul class="nav navbar-nav navbar-right">  
 <li class="dropdown">  
 <a href="#" class="dropdown-toggle" data-toggle="dropdown">用户中心<strong class="caret"></strong></a>  
 <ul class="dropdown-menu">  
 <li>  
 <a href="UserInfo.php">个人信息</a>  
 </li>  
 <li>  
 <a href="./Home.php">功能界面</a>  
 </li>  
 <li class="divider">  
 </li>  
 <li>  
 <a href="./Logout.php">退出登录</a>  
 </li>  
 </ul>  
 </li>  
 <li>  
 <a href="#"></a>  
 </li>  
 </ul>  
HTML**;  
 } **else** {  
 **echo <<<HTML  
 <ul class="nav navbar-nav navbar-right">  
 <li>  
 <a data-toggle="modal" data-target="#loginModal">登录</a>  
 </li>  
 <li>  
 <a data-toggle="modal" data-target="#signUpModal">注册</a>  
 </li>  
 <li>  
 <a href="#"></a>  
 </li>  
 </ul>  
HTML**;  
 }  
 **echo <<<HTML  
 </div>  
 </nav>  
HTML**;  
 }

|  |  |
| --- | --- |
| 函数名 | printRuleTable |
| 功能概要 | 输出规则表格 |
| 记述形式 | **function** printRuleTable() |
| 参数 | Null |

**function** printRuleTable()  
 {  
 $stmt = $this->**db**->getScrawlSettings();  
 **echo <<<HTML  
 <div class="table-responsive">  
 <table class="table table-striped">  
 <thead>  
 <tr>  
 <th>id</th>  
 <th>名称</th>  
 <th>开始地址</th>  
 <th>是否可用</th>  
 <th>操作</th>  
 </tr>  
 </thead>  
 <tbody>  
HTML**;  
 **while** ($row = $stmt->fetch(PDO::***FETCH\_ASSOC***)) {  
 $id = $row[**'id'**];  
 $name = $row[**'name'**];  
 $startURL = $row[**'starturl'**];  
 $enable = $row[**'enable'**];  
 $targetModal = **'#detailModal'** . $id;  
 **echo <<<HTML  
 <tr>  
 <td>**$id**</td>  
 <td>**$name**</td>  
 <td>**$startURL**</td>  
 <td>**$enable**</td>  
 <td>  
 <button type="button" class="btn btn-primary" data-toggle="modal" data-target=**$targetModal**>查看详情  
 <br>  
 </button>  
 <a class="btn btn-danger" href="./CrawlSettings.php?type=delete&id=**$id**" role="button">删除</a>  
 <!--<button type="button" class="btn btn-danger">删除</button>-->  
 </td>  
 </tr>  
HTML**;  
 }  
 **echo <<<HTML  
 </tbody>  
 </table>  
 </div>  
HTML**;  
 }