**开源软件基础课程报告**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 姓名 | 学号 | Github账号 | 分工 | 成绩 |
| 董苏 | 201592380 | 694040837 | 组长 |  |
| 陈夕希 | 201592209 |  | 成员 |  |
| 周晓峰 | 201592323 |  | 成员 |  |
| 王梓懿 | 201592138 |  | 成员 |  |
| 吕东芪 | 201592327 |  | 成员 |  |

|  |  |
| --- | --- |
| 报告题目 | 教务成绩爬取系统 |
| 项目网址 | https://github.com/694040837/dlut\_spider-master |
| 完成日期 | 2018年1月8日 |

大连理工大学软件学院

**目录**

[1 项目概述 3](#_Toc503012788)

[1.1项目背景 3](#_Toc503012789)

[2 需求分析 4](#_Toc503012790)

[2.1 可行性分析 4](#_Toc503012791)

[2.1.1技术可行性 4](#_Toc503012792)

[2.1.2 经济可行性 4](#_Toc503012793)

[2.1.3 操作可行性 5](#_Toc503012794)

[2.2功能需求 5](#_Toc503012795)

[2.2.1 实现抓包 5](#_Toc503012796)

[2.2.2 模拟登陆 5](#_Toc503012797)

[2.2.3 包的解析 5](#_Toc503012798)

[2.2.4 数据存储 6](#_Toc503012799)

[3 项目设计 6](#_Toc503012800)

[3.1 模拟登录 6](#_Toc503012801)

[3.2 数据解析设计 7](#_Toc503012803)

[3.3 数据存储设计 8](#_Toc503012808)

[4 编码实现 8](#_Toc503012809)

[4.1 模拟登录实现 8](#_Toc503012810)

[4.2 HTML数据解析实现 10](#_Toc503012811)

[4.3数据存储实现 11](#_Toc503012812)

[5 项目测试 12](#_Toc503012815)

# 1 项目概述

## 1.1项目背景

教务处是学校等教育机构的教学管理机构，主管各教师的教课任务、目标、进度及计划，学生的学习要求、目标、计划及考试等教与学各方面的事务，是学校的主要机构之一。教务系统管理平台充分利用互联网络B/S管理系统模式，以网络为平台，为各个学校教务系统的管理提供一个平台，帮助学校管理教务系统，用一个帐号解决学校教务教学管理，并且学校可以自由选择学校需要的教务管理系统，灵活地定制符合学校自己实际情况的教务系统。

教务成绩爬取系统主要使用python语言编写，目的是从教务处获取成绩并且把成绩转换成固定格式文件保存起来，我们学校教务处的地址是：http://zhjw.dlut.edu.cn/，往常每次我们获取成绩都需要先进入教务处，输入账号密码进行登录，然后点击成绩查询，最后输入相关信息获取成绩表格。在使用爬取系统之后，我们可以更便捷的获取教务处成绩信息。

* 1. 项目简介

本系统利用Python动态网络开发技术，借助cookie和requests库实现了教务系统模拟登陆功能，并通过input\_window编写了模拟登陆的GUI界面，然后使用BeautifulSoup来将得到的网页数据进行html解析，然后保存为csv文件，结合等脚本语言，以及配合多种网页开发工具，实现了整个爬取教务系统获取成绩的系统。

成绩爬取系统设计的目的旨在使成绩查询功能简洁、让数据结构灵活且精致，让学生能够轻松的查看自己的成绩，而不是通过繁琐的登陆过程后从教务网站中获取。

# 2 需求分析

## 2.1 可行性分析

## 2.1.1技术可行性

本设计的开发涉及多方面的技术，包括开发方法、开发平台、网络结构、系统布局和结构、输入输出技术、系统相关技术等。需要根据用户所提出的各种功能要求和限制条件，在技术的角度上研究本系统实现所涉及的技术以及这些技术的成熟度和现实性。

成熟技术是被多人采用并被反复证明行之有效的技术，因此采用成熟技术一般具有较高的成功率。另外，成熟技术经过长时间、大范围使用、补充和优化，其精细程度、优化程度、可操作性、经济性等方面要比新技术好。鉴于以上原因，本设计开发过程中，在可以满足系统开发需要、能够适应系统发展、保证开发成本的条件下，我们尽量采用成熟技术。

本设计所需要的技术总的来说是成熟和可行的，但是在开发队伍中如果没有人掌握这种技术，而且在项目组中又没有引进掌握这种技术的人员，那么这种技术对本系统的开发仍然是不可行的。凭借小组成员以前所学的软件开发方法和编程的知识，并且查阅相关资料和书籍可以为网站的开发提供足够的技术保障。

教务成绩爬取系统作为一款面向学生网络应用，需要注意的还有网络环境，如何降低网络环境的波动对系统性能产生的影响。比如面对期末这种高峰期时大量的用户登录，原有的资源就可能捉襟见肘，如何在低谷和高峰之间进行切换，在面对大量用户时不会导致明显的性能能下降，在面对少量用户时避免资源的浪费。

2.1.2 经济可行性

经济可行性是指可以使用的资源的可能性（资源包括人力资源、自然资源和资金条件）。包括两个方面的内容：一是某一备选方案占有和使用经济资源的可能性，进而实现政策目标的可能性；二是实施某一政策方案所需花费的成本和取得的收益。

根据成本-效益分析法来分析本设计的经济可行性。在成本方面有：开发软件的价格成本，安装和运行软件的成本以及学习成本。在效益方面有： 提高了登录效率，由于信息供给更为方便，成绩查询更便捷，由于进行了开发本系统的学习，组员能力得到提升。

本系统基于多种开源库开发，完全可以实现免费、开源，并且我们所开发的这套系统是基于提高用户使用效率，节省工作时间，简化操作管理的理念来设计的。并且本系统是个人独立设计开发的，并不需要投入大量经费，系统今后的运行和维护也相当简便，无需投入额外的资金，其成本不会出现超于预期的可能。并且在其过程中也可以提高个人的开发水平，因此在经济上是完全可行的。

## 2.1.3 操作可行性

在互联网+时代，信息技术早已深入到社会中的每个地方，它为人们带来来便利的工作方式，优越的工作环境，因此也对人们的工作效率和操作水平提出了更高要求。在这种发展趋势下，减少费时费力的人工操作，通过各种智能化软件来提高工作效率和工作精度就成为了一个很重要的方向。

一个成功的系统应该是可操作性强的，对于任何一个熟悉上网方式的人都应该能够无障碍的使用它。同时它的功能又应该是全面的，用户可以用符合日常习惯的方式来实现所要达到的目的。对于一些新颖的功能，应该通过给予用户一些奖励的方式使用户对其产生兴趣，并且试用它。

本系统的用户平台面向普通学生用户，其界面简洁，采用可视化界面，普通注册用户只需要轻点鼠标和键盘就可以使用系统，正是由于这些特性，本系统十分适于新手使用，无须学习复杂的教程便可以获得好的体验。因此本系统在操作上是可行的。

2.2功能需求

## 2.2.1 实现抓包

1）实现对教务系统包的抓取

## 2.2.2 模拟登陆

1）实现登录界面

2）实现数据的传输

3）实现数据正误判断

## 2.2.3 包的解析

1）实现对已抓取包的解析

2）实现对需要信息的提取

## 2.2.4 数据存储

1）分析已提取的数据

2）实现将分析后的数据转换成固定格式

3）将数据转为csv格式

4）导出并存储数据

# 3 项目设计

本成绩爬取系统主要分为模拟登陆、数据解析、数据转换及存储等模块，设计框架如图3.1

图3.1 系统框架图

## 3.1 模拟登陆

此模块预计设计大体流程为利用浏览器协议分析插件，分析页面请求和响应的时间、内容、以及浏览器用到的COOKIE等，可以非常直观的查看相应的信息。点击start是开始检测，点击stop暂停检测，点击clear清除内容。一般在使用之前，点击stop暂停，然后点击clear清屏，确保看到的是访问当前页面获得的数据。然后去教务成绩查询网站，查看登陆信息，分析get及post数据和cookie可知大体流程为首先，我们POST学号和密码--->然后返回cookie的值，然后发送cookie给服务器--->返回页面信息，然后根据这个原理来编写界面实现模拟登陆。



图3.2 教务登陆界面

## 3.2 数据解析设计

当拿到教务网站传来的数据包后我们需要对其进行解析来提取我们想要的数据，而html解析则主要利用python库BeautifulSoup来进行。

BeautifulSoup简介：Beautiful Soup是python的一个库，最主要的功能是从网页抓取数据，Beautiful Soup提供一些简单的、python式的函数用来处理导航、搜索、修改分析树等功能。它是一个工具箱，通过解析文档为用户提供需要抓取的数据，因为简单，所以不需要多少代码就可以写出一个完整的应用程序。Beautiful Soup自动将输入文档转换为Unicode编码，输出文档转换为utf-8编码。你不需要考虑编码方式，除非文档没有指定一个编码方式，这时，Beautiful Soup就不能自动识别编码方式了。然后，你仅仅需要说明一下原始编码方式就可以了。Beautiful Soup已成为和lxml、html6lib一样出色的python解释器，为用户灵活地提供不同的解析策略或强劲的速度。

Beautiful Soup将复杂HTML文档转换成一个复杂的树形结构,每个节点都是Python对象,所有对象可以归纳为4种：

1. Tag

HTML中的标签

1. NavigableString

标签内部文字转为的可遍历字符串

1. BeautifulSoup

BeautifulSoup 对象表示的是一个文档的全部内容.大部分时候,可以把它当作 Tag 对象，是一个特殊的 Tag

1. Comment

Comment 对象是一个特殊类型的 NavigableString 对象，其实输出的内容仍然不包括注释符号，但是如果不好好处理它，可能会对文本处理造成意想不到的麻烦

## 3.3 数据存储设计

进行完数据的获取和html解析之后，我们计划使用csv库，将解析后的数据转化成固定的格式而后保存下来。

# 4 编码实现

## 4.1 模拟登陆实现

登录界面input\_window.py:

**from** tkinter **import** \*  
**class** window(object):  
 **def** input\_information(self):  
 login\_data = {}  
 **def** cancel():  
 root.quit()  
  
 **def** confirm():  
 login\_data[**'number'**] = lab1.get()  
 login\_data[**'password'**] = lab2.get()  
 root.quit()  
  
 root = Tk()  
 root.title(**"模拟登陆"**)  
 frame = Frame(root)  
 frame.pack(padx=30, pady=16, ipadx=4)  
 \_number = StringVar()  
 lab1 = Entry(frame,textvariable = \_number)  
 lab1.grid(row=3, column=1, sticky=**'ew'**, columnspan=2)  
 \_password = StringVar()  
 lab2 = Entry(frame,textvariable = \_password)  
 lab2.grid(row=7, column=1, sticky=**'ew'**, columnspan=2)  
 button1 = Button(frame, text = **"确认"**,default=**'active'**,command = confirm)  
 button1.grid(row=11, column=2)  
 button2 = Button(frame,text = **"取消"**,command = cancel)  
 button2.grid(row=11, column=1, sticky=W)  
  
 root.update\_idletasks()  
 x = (root.winfo\_screenwidth() - root.winfo\_reqwidth()) / 2  
 y = (root.winfo\_screenheight() - root.winfo\_reqheight()) / 2  
 root.geometry(**"+%d+%d"** % (x, y))  
 root.mainloop()  
  
 **return** login\_data

模拟登陆实现loginDownload.py：

**import** urllib.parse  
**import** urllib.request  
**import** http.cookiejar  
**import** requests  
**import** random  
  
**class** Download(object):  
 **def** fuck\_login(self,number,password):  
 user\_agent = [ \  
 **"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/22.0.1207.1 Safari/537.1"**, \  
 **"Mozilla/5.0 (X11; CrOS i686 2268.111.0) AppleWebKit/536.11 (KHTML, like Gecko) Chrome/20.0.1132.57 Safari/536.11"**, \  
 **"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1092.0 Safari/536.6"**, \  
 **"Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1090.0 Safari/536.6"**, \  
 **"Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/19.77.34.5 Safari/537.1"**, \  
 **"Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.9 Safari/536.5"**, \  
 **"Mozilla/5.0 (Windows NT 6.0) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.36 Safari/536.5"**, \  
 **"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 5.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_8\_0) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3"**, \  
 **"Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.0 Safari/536.3"**, \  
 **"Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24"**, \  
 **"Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24"** ]  
 header = {**"User-Agent"**: random.choice(user\_agent),**'Referer'**: **'http://zhjw.dlut.edu.cn/menu/s\_main.jsp'**}  
 s = requests.session()  
 user\_data = {**"zjh"**: number,  
 **"mm"**: password}  
 afterURL = **"http://zhjw.dlut.edu.cn/gradeLnAllAction.do?type=ln&oper=fainfo&fa"** loginURL = **"http://zhjw.dlut.edu.cn/loginAction.do"** login = s.post(loginURL, data=user\_data, headers=header)  
 response = s.get(afterURL, cookies=login.cookies, headers=header)  
 *#print(response.raise\_for\_status())  
 #print(response.content)* content = response.content  
 requests.session().close()  
 **return** content

4.2 HTML数据解析实现html\_parser.py

**import** re  
**import** urllib.parse  
**import** urllib.request  
  
**from** bs4 **import** BeautifulSoup  
**class** HtmlParser(object):  
  
 **def** parse(self, html\_cont):  
 datas = []  
 soup = BeautifulSoup(html\_cont,**'html.parser'**,from\_encoding=**'GBK'**)  
 count = 0  
 data\_node = soup.find\_all(**'td'**, align=**"center"**)  
 *# print(data\_node)* res\_data = {}  
 **for** data **in** data\_node:  
 counter = count  
 **if** counter % 7 == 0 **and** count != 0:  
 res\_data = {}  
 datas.append(res\_data)  
 *# print(datas)* **if** counter % 7 == 0:  
 res\_data[**'number'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 1:  
 res\_data[**'smallNumber'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 2:  
 res\_data[**'name'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 3:  
 res\_data[**'EnglishName'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 4:  
 res\_data[**'credit'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 5:  
 res\_data[**'attributes'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **if** counter % 7 == 6:  
 res\_data[**'score'**] = **""**.join(data.get\_text().split())  
 count = count + 1  
 **return** datas

## 4.3数据存储实现csv\_output.py

**import** csv  
**class** CsvOutput(object):  
 **def** output\_csv(self, datas):  
 **if**(datas!=**None**):  
 fout = open(**'grade.csv'**, **'w'**, newline=**''**)  
 headers = [**'number'**, **'smallNumber'**, **'name'**, **'EnglishName'**,**'credit'**,**'attributes'**,**'score'**]  
 writer = csv.writer(fout)  
 writer.writerow(headers)  
 **for** i **in** range(len(datas)):  
 cont = []  
 cont.append(datas[i].get(**'number'**))  
 cont.append(datas[i].get(**'smallNumber'**))  
 cont.append(datas[i].get(**'name'**))  
 cont.append(datas[i].get(**'EnglishName'**))  
 cont.append(datas[i].get(**'credit'**))  
 cont.append(datas[i].get(**'attributes'**))  
 cont.append(datas[i].get(**'score'**))  
 print(cont)  
 writer.writerow(cont)  
 fout.close()

# 5 项目测试

首先用浏览器进行了抓包测试，查看了post的数据，出现了\_\_EVENTVALIDATION和\_\_VIEWSTATE这两个相对陌生的东西，百度了一下：

\_\_EVENTVALIDATION：

\_\_EVENTVALIDATION只是用来验证事件是否从合法的页面发送，只是一个数字签名，所以一般很短。

“id”属性为“\_\_EVENTVALIDATION”的隐藏字段是ASP.NET 2.0的新增的安全措施。该功能可以阻止由潜在的恶意用户从浏览器端发送的未经授权的请求.

为了确保每个回发和回调事件来自于所期望的用户界面元素，ASP.NET运行库将在事件中添加额外的验证层。服务器端通过检验表单提交请求的内容，将其与“id”属性为“\_\_EVENTVALIDATION”隐藏字段中的信息进行匹配。根据匹配结果来验证未在浏览器端添加额外的输入字段（有可能为用户在浏览器端恶意添加的字段），并且该值是在服务器已知的列表中选择的。ASP.NET运行库将在生成期间创建事件验证字段，而这是最不可能获取该信息的时刻。像视图状态一样，事件验证字段包含散列值以防止发生浏览器端篡改。

说明：“id”属性为“\_\_EVENTVALIDATION”隐藏字段一般在表单的最下方，如果表单在浏览器端尚未解析完毕时，用户提交数据有可能导致验证失败。

\_\_VIEWSTATE  
ViewState是ASP.NET中用来保存WEB控件回传时状态值一种机制。在WEB窗体(FORM)的设置为runat="server",这个窗体(FORM)会被附加一个隐藏的属性\_VIEWSTATE。\_VIEWSTATE中存放了所有控件在ViewState中的状态值。   
ViewState是类Control中的一个域，其他所有控件通过继承Control来获得了ViewState功能。它的类型是system.Web.UI.StateBag，一个名称/值的对象集合。   
当请求某个页面时，ASP.NET把所有控件的状态序列化成一个字符串，然后做为窗体的隐藏属性送到客户端。当客户端把页面回传时，ASP.NET分析回传的窗体属性，并赋给控件对应的值

了解后，我们知道这两个字段肯定是不可缺少的我们右键查看网页的源代码，在源代码中发现了两个type为hidden的输入框，它们的值正是我们所需要的。

随后对爬取系统进行了白盒测试语句覆盖级别的测试，并未发现太大问题，随后利用黑盒测试技术对系统功能进行了测试，能够实现预期功能。