**《Python》课程设计报告**

**（计算机学院）**

题目：

班级：

姓名：

指导教师：

2020-2021学年度暑假

微知库自动完成程序的开发

摘要：在校学习中，我们经常遇到要求在微知库中进行课程的学习的情况。而微知库中课程繁多，时间紧任务重，这时候我们就需要自动化软件来协助我们进行课程的学习。

关键词：Python，requests，软件开发。

# 需求分析

我校信息管理资源库平台，其中有多个课程需要“完成”。



（微知库平台首页）

其中一个课程需要完成的内容竟高达506项：



（需要刷506个）



（某课程的考核标准）

因此，急需一个自动化程序来辅助实现自动观看课程视频，自动提交讨论等功能。

## 功能分析

自动化程序的主要功能是：

1) 实现自动观看视频的功能。

2) 实现自动发起讨论（提问/投票等）的功能。

3) 多线程执行功能。

4) 刷课的开始，暂停，结束以及运行状况显示的功能。

## 功能图

根据以上需求分析，自动化程序功能如图1-1所示。

图1-1自动化程序的功能图

# 详细功能设计及代码实现

## 开发环境

本次开发使用Pycharm IDE + Python3.9 + requests[[1]](#footnote-1)模块+BeautifulSoup4[[2]](#footnote-2)模块进行。

## 项目准备

如图2-1，在某代码协作平台创建一个使用SVN[[3]](#footnote-3)版本控制的私有项目wzk\_auto.



图2-1 创建SVN项目

如图2-2，使用Pycharm连接到刚刚创建的项目。、

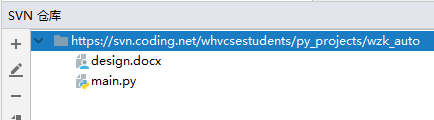


图2-2 连接到项目

## 登录

### 取登录链接

首先观察登录页的URL[[4]](#footnote-4)：

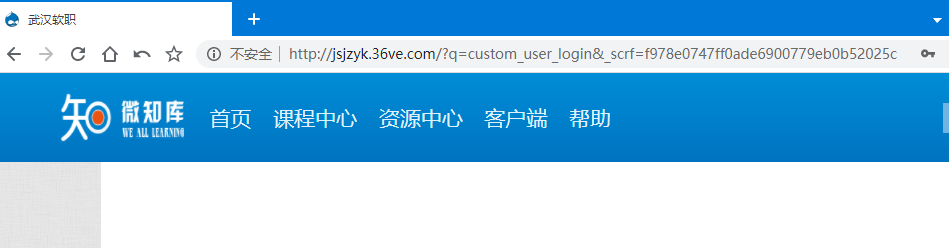


图2-3 微知库登录页URL

这里我们注意到登录页除了通常的登陆参数还带有一个 ”scrf” 参数。在MD5[[5]](#footnote-5)查询网站上查询到是六位大小写和数字构成的随机值，可能是登陆验证的验证码。

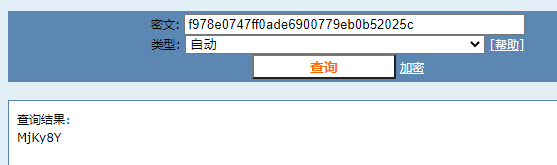


图2-4解密结果

删除 \_scrf 字段，发现无法打开登录页。

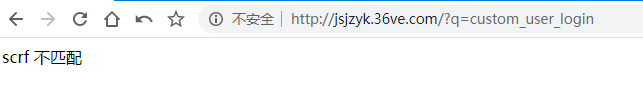


图2-5 scrf不匹配

打开“微知库”首页，在登录按钮所对应的链接上找到了 \_scrf 参数的值，所以我们只需要从这里取登录链接就好。

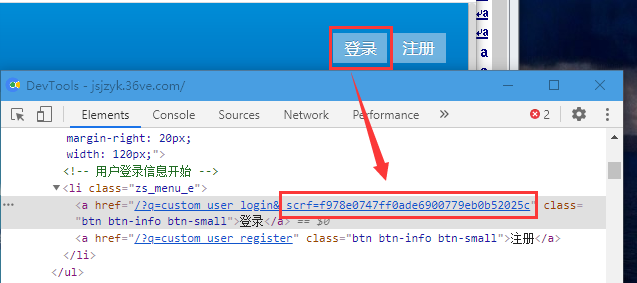


图2-6 带 \_scrf 参数的登陆链接

下为获取正确登陆链接的代码：



### 登陆行为

#### 登录验证

打开登录页面。



图2-7 微知库登陆页面

随意输入一些账号密码，我们注意到密码在移开输入框焦点后变成了一长串密文。



图2-8 加密的密码

删除密码框的password属性，查看明文：

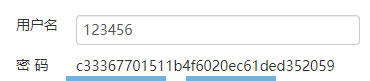


图2-9 加密后的明文

同样的，在MD5查询网站查到密文就是MD5加密后的密码。

尝试输入用户名密码进行登录。发现还有滑动验证码。



图2-10 滑动验证码

观察其附近的网页源代码，发现其引入了一个外部js文件。



图2-11 引入的js文件

打开js文件，发现了一个Github页面。

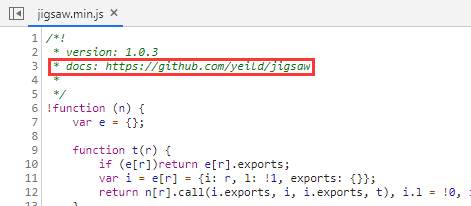


图2-12 Github页面

打开Github页面，我们发现了一句话和一段代码：

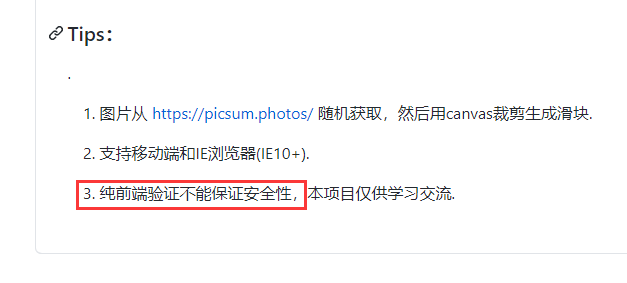


图2-13 一句话

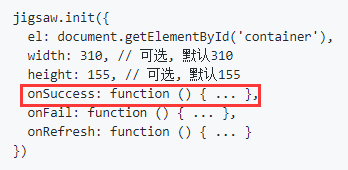


图2-14 一段代码

在html中对应位置查找，发现了登录参数：

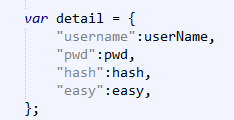


图2-15 登陆参数

"username":userName, #用户名 字符型

"pwd":pwd, #密码 MD5加密后的密码。



"hash":hash,#hash 从网页中获取的元素，32位MD5。具体到网页中，是一个输入框中的值。应该和前面的\_scrf参数一样是用于防止自动化程序。



图2-16 csrf\_token

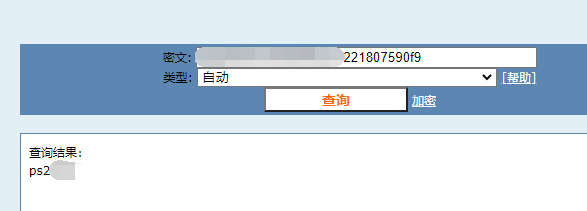


图2-17 解密结果



图2-18 相关代码

"easy":easy,# 1是否弱密码，测试过程中发现无论输入值为何，都提示弱密码。可能是32位MD5只包含小写字母和数字导致。但发送给服务器的值始终为0（False）。

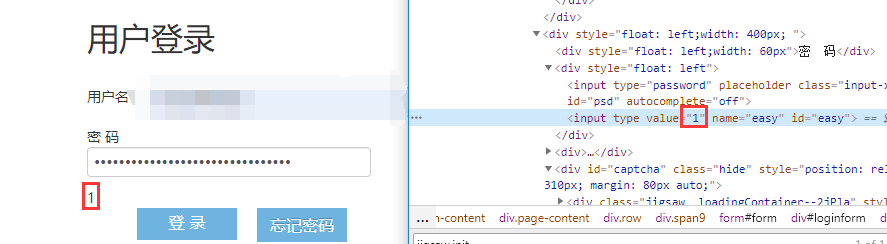


图2-16 弱密码

#### 构造登录请求

正常登陆一次，得到登录请求。

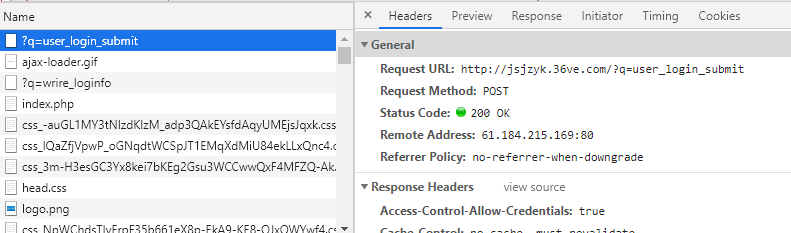


图2-17 登录请求

查看网页源代码中相关部分，发现登陆参数先被Base64[[6]](#footnote-6)编码后通过POST方法发送到微知库网站（/?q=user\_login\_submit）。



图2-18 网页源代码中相关部分

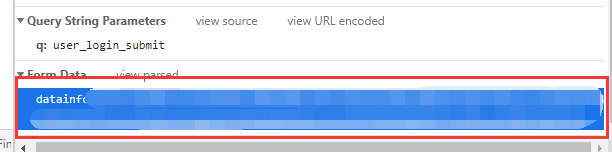


图2-19 实际发送的信息

解码后的结果也验证了这一点：

{"username":"\*\*\*\*\*\*\*\*\*\*\*","pwd":"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*","hash":"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*","easy":"0"}

那么我们对构造的请求参数进行Base64加密即可。



请注意构造请求的信息格式要与正常的信息格式相同！包括空格与标点符号！

另外POST提交的参数为：

{“datainfo”:[Base64编码后的登录请求]}

登陆成功后网页发出的回应是这样的：

﻿{"status":0,"data":"","is\_true\_name":0,"expiretime":1111111111,"userid":"22222","href":""}

### 登录部分的完整代码

*# 替换引号字符并移除空格***def** replace\_sym(data):  
 data = str(data)  
 data = data.replace(**"'"**,**'"'**)  
 data = data.replace(**" "**, **''**)  
 **return** data  
  
*# 登录函数***def** login(username, password, wzk\_url = **'http://jsjzyk.36ve.com'**):  
 header = {**"User-Agent"**: **"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36"**}  
 *# 取登录页验证参数 \_scrf* login\_part = BeautifulSoup(request.get(wzk\_url, headers=header).text,features=**"html.parser"**).find(name=**'a'**,attrs={**'class'**:**'btn btn-info btn-small'**}).get(**'href'**)  
 *# http://jsjzyk.36ve.com + /?q=custom\_user\_login&\_scrf=\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** login\_url = wzk\_url + login\_part  
 print(login\_url)  
 *# 取登录信息 csrf\_token* login\_csrf\_token = BeautifulSoup(request.get(login\_url, headers=header).text,features=**"html.parser"**).find(name=**'input'**,attrs={**'name'**:**'csrf\_token'**,**'id'**:**'csrf\_token'**}).get(**'value'**)  
 *# 加密明文密码* pwd = hashlib.md5(password.encode(encoding=**'UTF-8'**)).hexdigest()  
 *# 简单密码标记* easy = **"0"** *# 构造登录请求* data = {**"username"**: username,  
 **"pwd"**: pwd,  
 **"hash"**:login\_csrf\_token,  
 **"easy"**:easy}  
 *# 替换引号字符并移除空格* data = replace\_sym(data)  
 print(**'登录信息：'**,data)  
 *# 构造待提交的登录请求* data = {**"datainfo"**:base64.b64encode(str(data).encode(encoding=**'UTF-8'**)).decode(encoding=**'UTF-8'**)}  
 *# 提交请求的URL* login\_post\_url = wzk\_url + **'/?q=user\_login\_submit'** *# 提交请求* req = request.post(login\_post\_url, data=data, headers=header)  
  
 print(req.text) *# 这个用来测试是否登陆成功(查看登陆前与登录后的源代码有何差别)  
  
 # csrf验证未通过，请重新提交！* **if r'\u7528\u6237\u540d\u6216\u5bc6\u7801\u9519\u8bef\uff0c\u8bf7\u91cd\u65b0\u8f93\u5165\uff01' in** req.text:  
 print(**"用户名或密码错误，请重新输入！"**)  
 **elif r'\u7531\u4e8e\u60a8\u8fde\u7eed\u8f93\u5165\u5bc6\u7801\u9519\u8bef\uff0c\u8d26\u53f7\u5c06\u4f1a\u9501\u5b9a5\u5206\u949f\uff0c\u8bf75\u5206\u949f\u540e\u518d\u767b\u5f55\uff01' in** req.text:  
 print(**"由于您连续输入密码错误，账号将会锁定5分钟，请5分钟后再登录！"**)  
 **elif r"csrf\u9a8c\u8bc1\u672a\u901a\u8fc7\uff0c\u8bf7\u91cd\u65b0\u63d0\u4ea4\uff01" in** req.text:  
 print(**"csrf验证失败！"**)  
 **else**:  
 print(**"登录成功"**)

## 获取课程和链接

首先以登录用户的身份打开“微知库”“我学的课程”页面。



图2-20 微知库“我学的课程页面”

链接部分是包含了userid的，从登陆成功的回应可以得到userid。

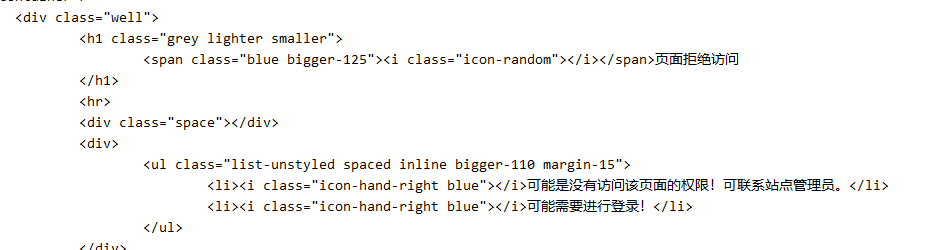


图2-21 页面拒绝访问

## 获取课程中的链接

打开课程页面，获取页面源代码：



图2-22 已刷完和未刷页面



图2-23 相关源代码

分析：radius bg-img 为已刷完的链接，radius bg-img-half为未刷完的链接，radius[空格]为未刷的链接。而所有链接都放在“中间内容”里。



图2-24 中间内容

相关代码如下：

course\_page = request.get(url, headers=header)  
中间内容 = BeautifulSoup(course\_page.text, features=**"html.parser"**).find(name=**'div'**, attrs={**'class'**: **'courselist'**})  
所有页面链接元素 = BeautifulSoup(str(中间内容), features=**"html.parser"**).find\_all(name=**'li'**, onclick=**True**)  
所有页面链接元素 = [链接元素.find(**'a'**).get(**'href'**) **for** 链接元素 **in** 所有页面链接元素 **if** 链接元素.find(**'radius bg-img-half'**)!=**'' or** 链接元素.find(**'radius '**)!=**''**]

1. Requests 是一个 Python 的 HTTP 客户端库。 [↑](#footnote-ref-1)
2. Beautiful Soup 是一个可以从HTML或XML文件中提取数据的Python库。 [↑](#footnote-ref-2)
3. SVN是subversion的缩写，是一个开放源代码的版本控制系统，通过采用分支管理系统的高效管理，简而言之就是用于多个人共同开发同一个项目，实现共享资源，实现最终集中式的管理。 [↑](#footnote-ref-3)
4. URL是统一资源定位符，是互联网上标准资源的地址。 [↑](#footnote-ref-4)
5. MD5信息摘要算法（英语：MD5 Message-Digest Algorithm），一种被广泛使用的密码散列函数，可以产生出一个128位（16字节）的散列值（hash value），用于确保信息传输完整一致 [↑](#footnote-ref-5)
6. Base64是网络上最常见的用于传输8Bit字节码的编码方式之一，Base64就是一种基于64个可打印字符来表示二进制数据的方法。Base64编码是从二进制到字符的过程，可用于在HTTP环境下传递较长的标识信息。 [↑](#footnote-ref-6)