Python 爬虫---爬取教学管理课程表

1. 获取教务处登录信息

1)、获取验证码的url

按照下图步骤，查看网页的构造，我猜基本验证码的刷新都是利用js更新。



此时，我们就获取到了验证码的 url: /yzm?d=1578017120418

该路径是相对路径，所以需要加上此时网页的url。

完整路径为：<http://jxgl.wyu.edu.cn/yzm?d=1578017120418>。

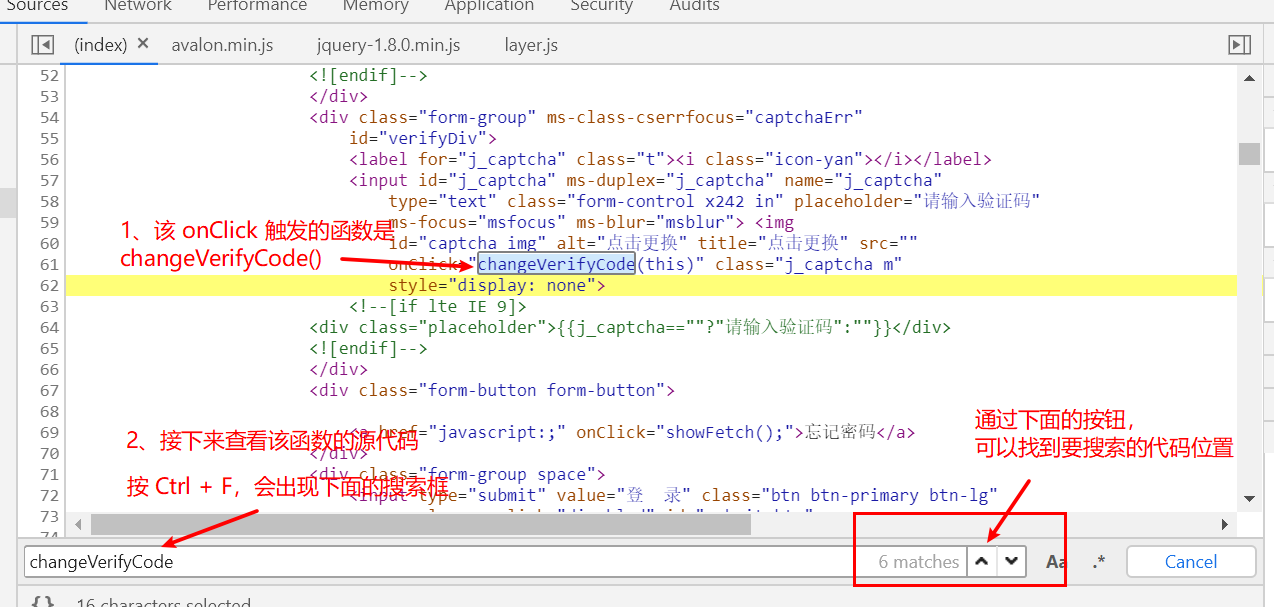
此时可能会有疑问。/yzm?d=1578017120418，该 url 携带的参数d后面的值是什么？

所以接下来需要查看该网页的源代码，找到验证码更新的触发事件。

1. 查找源代码



b) 查找源代码





可以看到上图changeVerifyCode() 源代码，原来d参数携带的值是当前的值。不过通过尝试，我发现携带任何数值都可以获取到验证码。

通过上述操作我们已经知道了该网站是如何获取到验证码的，所以接下来就是自己构造 url 获取验证码

下述代码d参数携带的值是当前的时间，利用 time 库中的 time()，需要引入该库

**def** getVerifiedCode():  
 *# 验证码地址* link = **'http://jxgl.wyu.edu.cn/yzm?d={}'**.format(time.time()) *#获取当前时间* headers = {**'User-Agent'**: **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36'** }  
 *# 获取验证码对象* verifiedNum = requests.get(link, headers = headers)  
 *#print('verified Cookie:{}'.format(verifiedNum.cookies.get\_dict().get('JSESSIONID'))) #获取 cookies  
 # 存储验证码，转成 jpg, 无法智能识别，只能靠人眼识别* file = open(**'verifiedCode.jpg'**, **'wb'**)  
 file.write(verifiedNum.content)  
 **return** verifiedNum

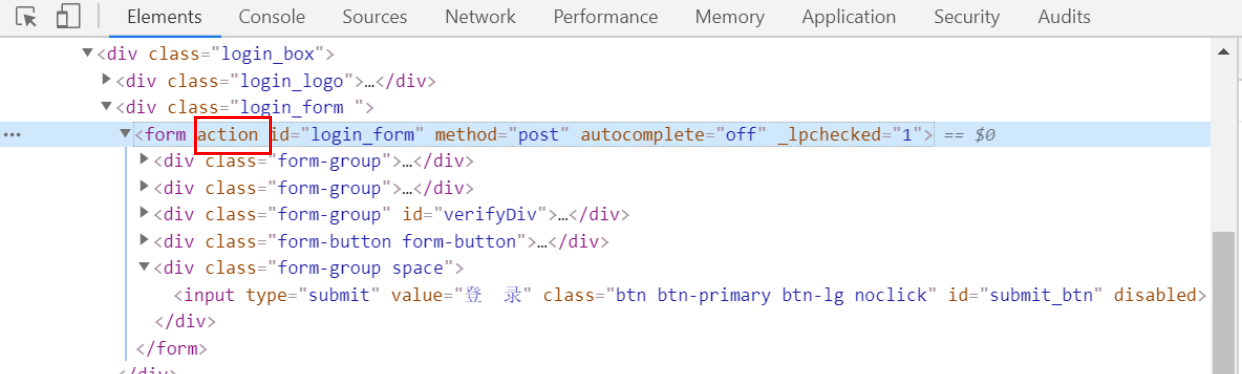
好了，通过上述步骤和代码我们就获取了验证码。

接下来我们就要查看教务处网站是如何登录的， 我了解到的基本的网站登录都是采用 post 请求，request header携带参数和 cookies，所以下一步就是要获取参数信息。

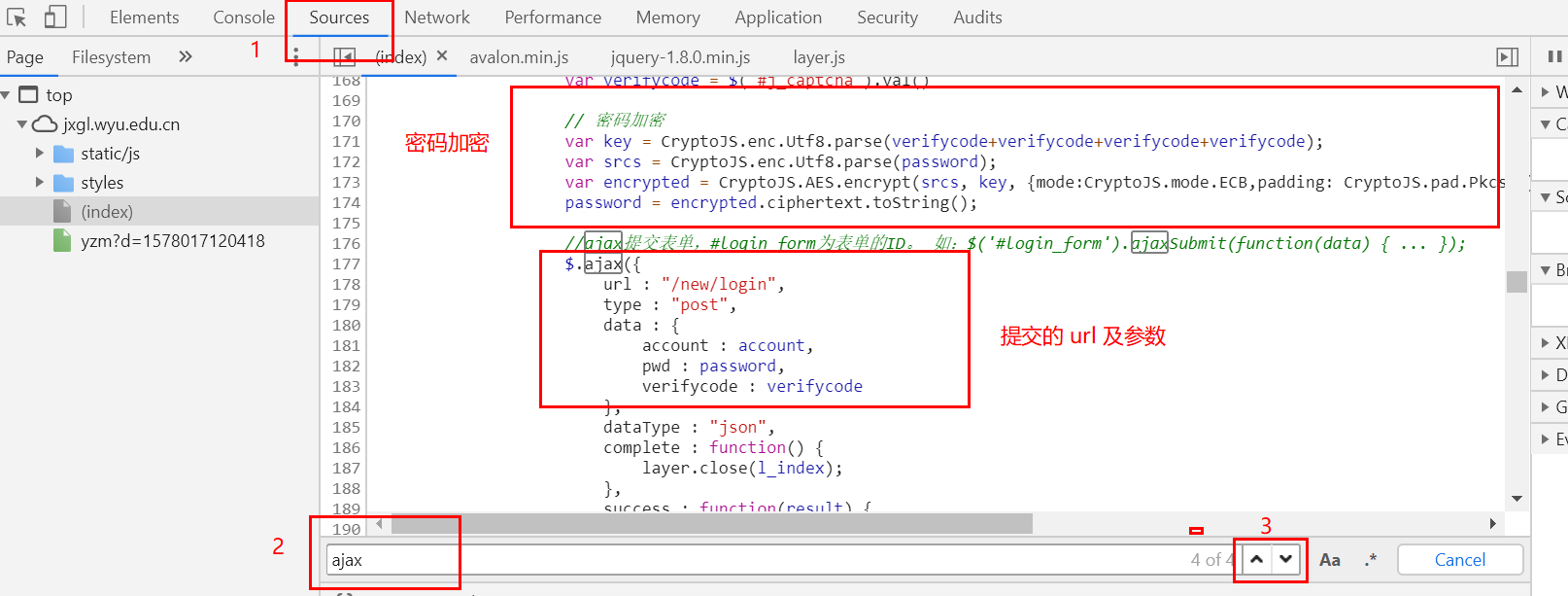
2）获取教务处提交账号密码信息。

最开始我是采取抓包分析的，发现密码是进过加密后（基本所有网站的密码都是加密传送），所以还是需要查看源代码，通过查看源代码也能获取到post提交的信息，所以抓包有点多此一举了，下面直接演示查看源代码。

1. 查看源代码



查看网页的构造，发现该登录框 form 表单(网站基本都是 form 表单提交)，然后惊讶发现，action 参数值没指定提交的url，最开始猜想会不是是采用 js 点击更换值，后面想想这样有点多次一举。那就可能是采用 ajax 提交。所以接下来查看 js 源代码。



果然采用的是 ajax 提交，而且采用的是AES 加密，最开始不知道 cryptoJS 是利用哪个库，怎么加密的。然后经过百度搜索，查找到了相关的知识。同时也查找到了 python 中相同加密的库 pycryptodome。

通过上述图，可以了解到提交的目的url：/new/login，相对地址

完整的 url ：<http://jxgl.wyu.edu.cn/new/login>， POST 请求

携带的参数：’account’, ‘pwd’, ‘verifycode’

1. 加密密码

通过分析上述源代码，发现密码是 AES 加密，采用的库

cryptoJS，python 中同样的库为：pycryptodome。

如何使用pycryptodemo库和原理可以先参考下：<https://www.jianshu.com/p/865f3e570780>

*# 如果text不足16位的倍数就用空格补足为16位  
# 不同于JS，pycryptodome库中加密方法不做任何padding，因此需要区分明文是否为中文的情况***def** add\_to\_16\_cn(*text*):  
 pad = 16 - len(*text*.encode(**'utf-8'**)) % 16  
 text = *text* + pad \* chr(pad)  
 **return** *text*.encode(**'utf-8'**)  
**def** encry(*key*, *pwd*):  
 *# 教务处的 js 加密，对应 python 也要用相应的加密  
 #var key = CryptoJS.enc.Utf8.parse(verifyCode+ verifyCode+ verifyCode+ verifyCode);  
 #var srcs = CryptoJS.enc.Utf8.parse('pwd');  
 #var encrypted = CryptoJS.AES.encrypt(srcs, key, {mode:CryptoJS.mode.ECB,padding: CryptoJS.pad.Pkcs7});  
  
 ## python AES 加密* mode = AES.MODE\_ECB  
 cryptos = AES.new(*key*, mode)  
 pw = add\_to\_16\_cn(*pw*)  
 cipher\_text = cryptos.encrypt(*pw*)  
 *#print('加密后的：{}'.format(b2a\_hex(cipher\_text))) #将密文转成16进制* **return** b2a\_hex(cipher\_text)

教务处加密的位密钥 key是输入4个验证码相加的字符串，所以相应的我们python代码中也要使用相同的密钥，即上述encry(key, pwd) 中的key 传入的值应该为：

*# 加密的密钥*key = (verifiedCode\*4).encode(**'utf-8'**)

VerifiedCode 是我们输入的验证码

上述操作就获取到了登录的信息了，接下来就要模拟登录。

1. Cookies模拟登录
2. 获取 cookies。

在第一部分中，我们已经获取到了验证码的信息，同时该信息中还包含着cookies，该cookies 是服务器用来验证提交的验证码的信息。

*# 获取验证码*verifiedCode = getVerifiedCode()  
*#获取 cookies*cookies = requests.utils.dict\_from\_cookiejar(verifiedCode.cookies)

getVerifiedCode 是第一部分的获取验证码的函数。

1. POST 登录
2. *# 登录***def** login():  
    link =**'http://jxgl.wyu.edu.cn/new/login'** headers = {**'User-Agent'**: **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36'** }

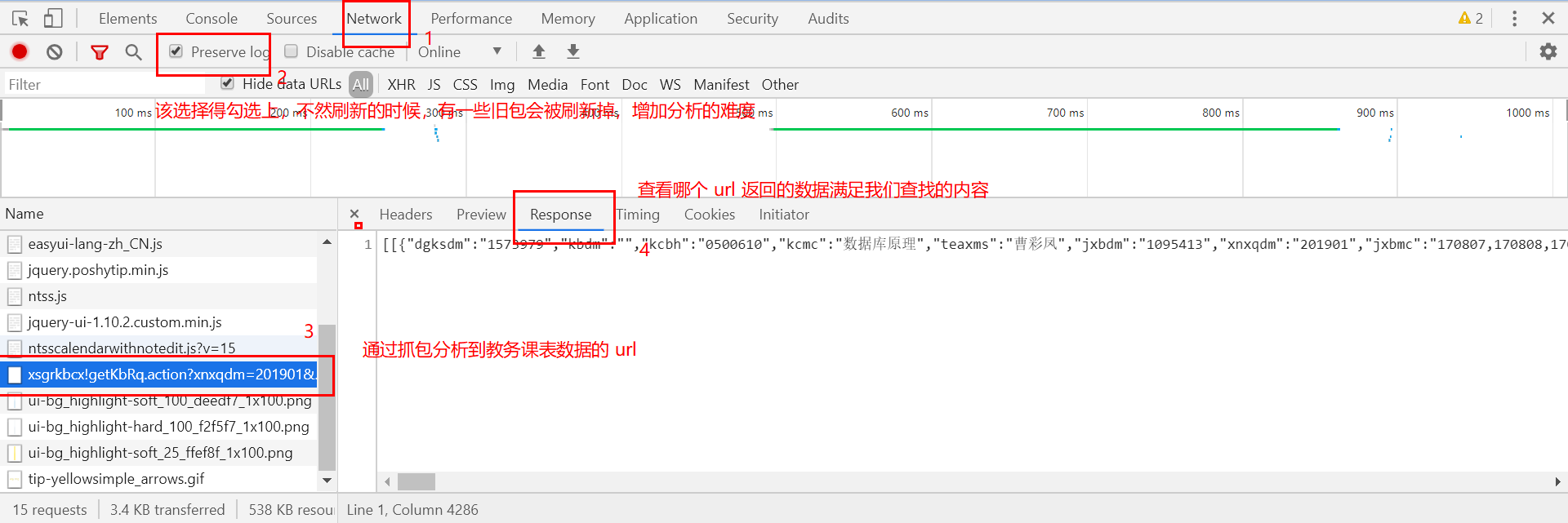
*# 请求携带的参数*

paras = {  
 **'account'**:**''**,  
 **'pwd'**:**''** }  
 *#输入用户名* print(**"账号:"**)  
 account = input()  
 *#输入密码* print(**"密码:"**)  
 pwd = pwd\_input()  
 paras[**'account'**] = account  
 paras[**'pwd'**] = pwd  
 print()  
 *# 获取验证码* verifiedCode = getVerifiedCode()  
 *#获取 cookies* cookies = requests.utils.dict\_from\_cookiejar(verifiedCode.cookies)  
 *# 输入验证码* verifiedCode = input(**'请输入验证码: '**)  
 *#print('从验证码响应中获取的 Cookies:{}'.format(cookies))* paras[**'verifycode'**] = verifiedCode  
 *# 加密的密钥* key = (verifiedCode\*4).encode(**'utf-8'**)  
 *#print('key: %s',key)* pw = paras[**'pwd'**]  
 *# 密码 AEC 加密* secret = encry(key, pw)  
 paras[**'pwd'**] = secret  
 *# 登录* login = requests.post(link, headers = headers, cookies = cookies, data = paras)  
 print(login.text)  
 print(login.status\_code)  
 print(**'登录后的 header:{0}'**.format(login.headers))  
 print(**'登录后的cookies:{}'**.format(login.cookies))  
 *# 返回一个元组* **return** login.status\_code, cookies

登录后，后面的请求都是以携带登录成功的 cookies 来请求数据

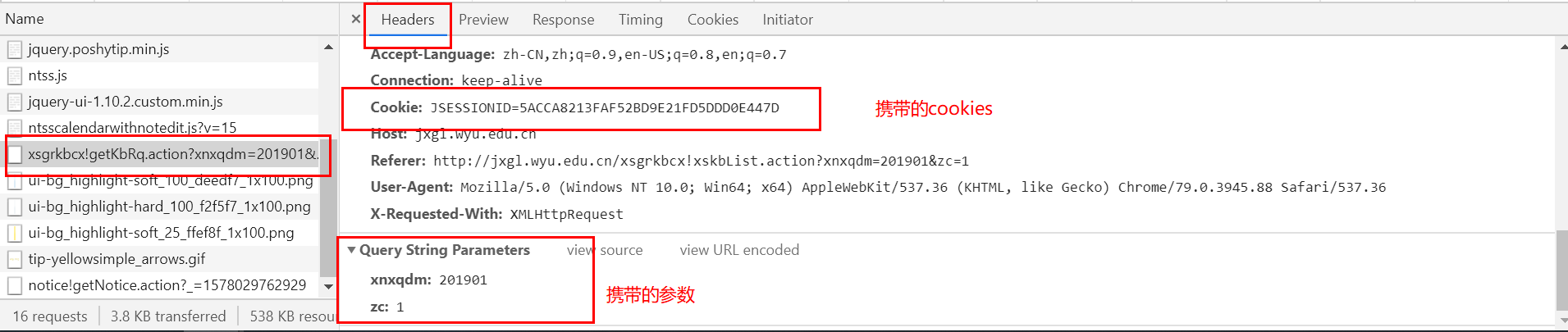
1. 获取课表信息

这部分需要先登录到教务管理系统，通过开发者的抓包分析出数据来源的url



此时我们就获取到课表数据的来源 url:xsgrkbcx!getKbRq.action?xnxqdm=201901&zc=01

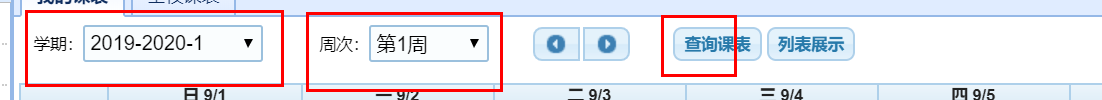
接下来我们就需要去分析下该请求携带的参数的值，



经过发送请求，抓包分析，发现请求携带的参数

xnxqdm: 对应请求查询学期的值

zc: 对应请求查询周次的值



至此，我们就已经获取了如何请求课表信息的方法。

所以需要来自己构建请求

#获取19个请求，从指定学期的第一周到19周

link2\_list = [**'http://jxgl.wyu.edu.cn/xsgrkbcx!getKbRq.action?xnxqdm={0}&zc={1}'**.format(xnxqdm, number) **for** number **in** range(1, 20)] *##获取课表，返回json，携带参数*

获取整个课表的信息的完整代码

**def** getPage(*link*, *cookies*, *params*):  
 headers = {**'User-Agent'**: **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36'** }page = requests.post(*link*, headers = headers, cookies = *cookies*, data = *params*)  
 print(**"获取到页面信息: "**,page.text, **'**\n**'**)  
 **return** page

**def** getAllClassSchedule(*cookies*):  
 xnxqdm = input(**"请输入获取指定学期的课表(例：201801, 2018第一学年):"**)  
 link2\_list = [**'http://jxgl.wyu.edu.cn/xsgrkbcx!getKbRq.action?xnxqdm={0}&zc={1}'**.format(xnxqdm, number) **for** number **in** range(1, 20)] *##获取课表，返回json，携带参数* pageAll = [getPage(link, *cookies*, params=**None**) **for** link **in** link2\_list]schedules = []  
 **for** page **in** pageAll:

#转换 json 数据scheduleInfo = json.loads(page.text)  
 schedules.append(scheduleInfo[0]) *#返回全部json对象* **return** schedules

1. 存储课表至 excel 文件中

使用到 xlwt 库，具体操作参见下代码。

1. 参考资料

书籍：《从零开始学 Python 网络爬虫》罗攀、蒋仟编著，机械工业出版社，ISBN 987-7-111-57999-1

博客：

Request cookies 登录：<https://blog.csdn.net/will4906/article/details/77124615>

JS和python 实现AES算法；<https://www.jianshu.com/p/865f3e570780>

Python操作excel 表：<https://zhuanlan.zhihu.com/p/38492442>

s