python爬虫如何POST request payload形式的请求?

1、背景

爬取数据、登陆网站的时候、发现在POST数据时，使用的数据格式是request payload，有别于之前常见的 POST数据格式（Form data）。而使用Form data数据的提交方式时，无法提交成功。



1.1payload(json格式的字符串) 其他形式的POST请求，是放到 Request payload 中（现在是为了方便阅读，使用了Json这样的数据格式），请求的Content-Type设置为application/json;charset=UTF-8或者不指定。

请看下面payload的请求

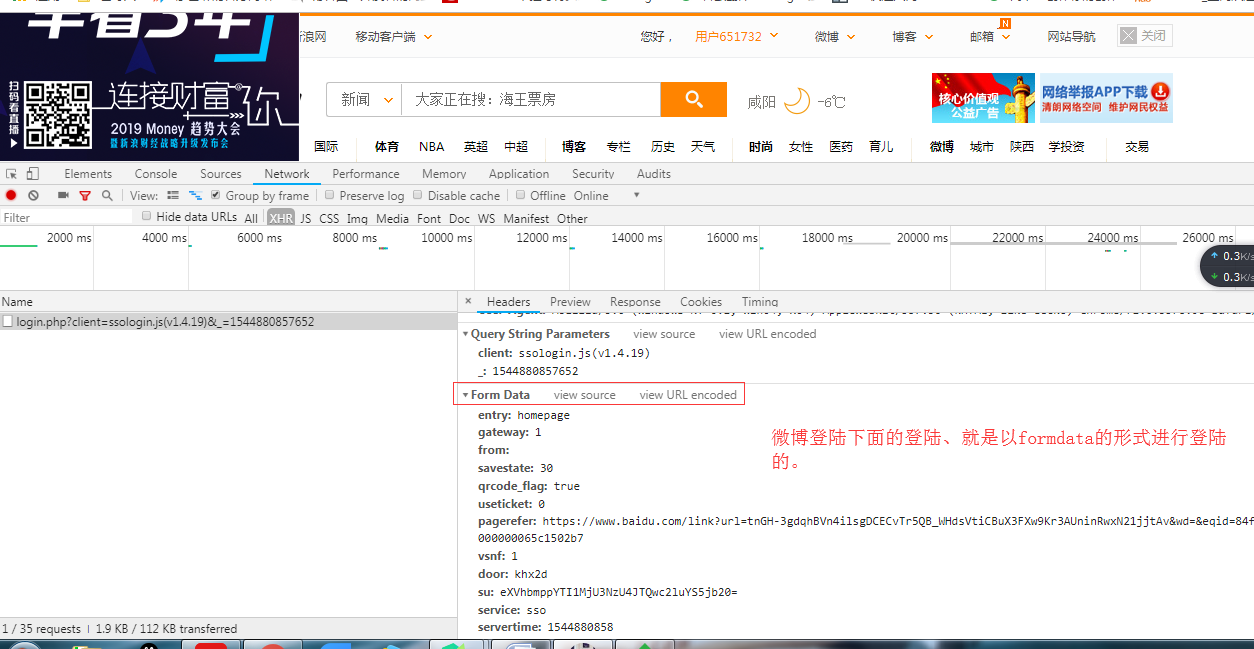
r = requests.post(postUrl, data=**json.dumps**(payloadData))，

重点是json.dumps json的作用是将 python数据类型、进行json数据类型编码

而json在 python中是str类型、 （也可以理解为将字典格式数据、转变为json格式数据）

2、 form（表单）登陆请求：如果一个请求的Content-Type设置为application/x-www-form-urlencoded，那么这个Post请求会被认为是Http Post表单请求，那么请求主体将以一个标准的键值对和&的querystring形式出现。这种方式是HTML表单的默认设置，所以在过去这种方式更加常见。

html=requests.post(url,data=params)



|  |
| --- |
| post\_data = { |
|  | 'entry': 'weibo', | |
|  | 'gateway': '1', | |
|  | 'from': '', | |
|  | 'savestate': '7', | |
|  | 'qrcode\_flag': 'false', | |
|  | 'useticket': '1', | |
|  | # 'pagerefer': 'https://www.baidu.com/', | |
|  | 'vsnf': '1', | |
|  | 'su': self.username, # 通过bs64加密后的得到的 | |
|  | 'service': 'miniblog', | |
|  | 'servertime': json\_data['servertime'], | |
|  | 'nonce': json\_data['nonce'], | |
|  | 'pwencode': 'rsa2', | |
|  | 'rsakv': json\_data['rsakv'], | |
|  | 'sp': pass\_word, # 通过rsa加密过的pass\_word | |
|  | 'sr': '1920\*1080', | |
|  | 'encoding': 'UTF-8', | |
|  | 'prelt': 28, | |
|  | 'url': 'https://weibo.com/ajaxlogin.php?framelogin=1&callback=parent.sinaSSOController.feedBackUrlCallBack', | |
|  | 'returntype': 'META' | |
|  | } | |
| login\_url = 'https://login.sina.com.cn/sso/login.php?client=ssologin.js(v1.4.19)' | |
| response\_data = self.session.post(login\_url, data=**post\_data**, verify=False) | | . | |

3、扫码登陆微信登陆

主要思想根据fiddler分析页面加载二维码前后、产生的参数变化、

比如微信二维码下面的uuid这个参数、就是扫码登陆微信的关键

import requests  
import re  
from PIL import Image  
import urllib3  
from bs4 import BeautifulSoup  
urllib3.disable\_warnings() #不看警告  
  
'''  
扫描以后返回的地址  
https://wx2.qq.com/cgi-bin/mmwebwx-bin/webwxnewloginpage?ticket=A7gGL22E-PUzviXmur53Azno@qrticket\_0&uuid=gc\_tSTRpaA==&lang=zh\_CN&scan=1520997537  
https://wx2.qq.com/cgi-bin/mmwebwx-bin/webwxnewloginpage?ticket=A7gGL22E-PUzviXmur53Azno@qrticket\_0&uuid=gc\_tSTRpaA==&lang=zh\_CN&scan=1520997537  
'''  
class WxBott(object):  
 def \_\_init\_\_(self):  
 self.session = requests.session() #登陆问题用session  
 self.session.verify = False #统一取消验证  
 def get\_uuid(self):  
 url = "https://login.wx.qq.com/jslogin?appid=wx782c26e4c19acffb&redirect\_uri=https%3A%2F%2Fwx.qq.com%2Fcgi-bin%2Fmmwebwx-bin%2Fwebwxnewloginpage&fun=new&lang=zh\_CN&\_=1520997435513"  
 #这个URL是由二维码的 headers 有一个参数不知道哪里来， 然后复制它， 在url里面找， 找到之后把有这个参数的url复制出来就是这个  
 result = self.session.get(url).text  
  
 #获取到以后 在进行正则匹配， 匹配其中的uuid  
 self.uuid = re.findall(r'uuid = "(.\*?)"',result)[0]  
  
 #每次出来的uuid 都是不相同的  
 print(self.uuid)  
  
 def get\_qcode(self):  
 '''  
 获取登陆二维码  
 '''  
  
 url = 'https://login.weixin.qq.com/qrcode/{}'.format(self.uuid)  
 with open('qqqqqcode.jpg','wb') as f:  
 f.write(self.session.get(url).content)  
 image = Image.open('qqqqqcode.jpg') #读取二进制图片  
 image.show()  
  
 def visit\_login(self):  
 url = 'https://login.wx.qq.com/cgi-bin/mmwebwx-bin/login?loginicon=true&uuid={}&tip=0&r=-591889347&\_=1521010308313'.format(self.uuid) #这个网址的由来是 复制了返回正真微信url的网址  
 while True:  
 result = self.session.get(url).text  
 print(result)  
 if '200' in result: #如果包含200就跳出  
 self.redirect\_url = re.findall(r'redirect\_uri="(.\*?)";',result)[0] #这里的redirect\_uri="(.\*?)"; 很精髓 只是把需要的东西这样写(.\*?) 别的不能动 后面加上[0]  
 break  
 print("登陆成功") #打开跳转的网址  
 print(self.redirect\_url)  
  
 def visit\_parse(self):  
 result = self.session.get(self.redirect\_url,allow\_redirects = False).text # allow\_redirects = False 防止跳转  
 soup = BeautifulSoup(result,'lxml')  
 self.skey = soup.find('skey').text  
 self.wxsid = soup.find('wxsid').text  
 self.wxuin = soup.find('wxuin').text  
 self.pass\_ticket = soup.find('pass\_ticket').text  
 self.isgrayscale = soup.find('isgrayscale').text  
  
 def visit\_init(self):  
 url = 'https://wx2.qq.com/cgi-bin/mmwebwx-bin/webwxinit?r=-1033474349'  
 data = {"BaseRequest":{"Uin":self.wxuin,"Sid":self.wxsid,"Skey":self.skey,"DeviceID":"e041800958560706"}}  
 result = self.session.post(url,data)  
if \_\_name\_\_ == '\_\_main\_\_':  
 wx = WxBott()  
 wx.get\_uuid()  
 wx.get\_qcode()  
 wx.visit\_login()  
 wx.visit\_parse()  
 wx.visit\_init()  
 print(wx.skey)  
 #print(wx.redirect\_url)

4、如果找不到登陆需要的变化参数、可以用selenium模拟浏览器登陆、获取cooking

拿到cooking以后、加到自己的请求代码里面、实现半模拟、半模拟的形式去访问。

明白坐标和等待便可以、去登陆自己想要登陆的网站。

import time  
from selenium import webdriver  
from selenium.webdriver.support.wait import WebDriverWait #事件循环， 不断的去检测  
from selenium.webdriver.support import expected\_conditions as EC #提供给我们的一些事件（包括事件有没有加载出来）  
from selenium.webdriver.common.by import By #做表定位  
  
#打开一个浏览器  
dirver = webdriver.Chrome()  
dirver.get('https://www.weibo.com')  
dirver.maximize\_window() #最大化  
  
#显示等待  
wait = WebDriverWait(dirver,20,0.5) #三个参数 1、dirver 对象 ，2、超时的时间20秒， 3、0.5秒刷新  
  
try:  
 #等待两个数据框的加载， 一般用id 或者xpath  
 wait.until(EC.presence\_of\_all\_elements\_located((By.ID,'loginname'))) #根据id寻找  
 wait.until(EC.presence\_of\_all\_elements\_located((By.XPATH,'//\*[@id="pl\_login\_form"]/div/div[3]/div[2]/div/input'))) #根据xpath  
  
 #加载完数据框以后， 然后输入  
 username = dirver.find\_element\_by\_id('loginname') #定位账号数据框  
 username.clear() #清除以前的账号  
 username.send\_keys('18082539819') #输入账号  
 time.sleep(5)  
  
  
 passwork = dirver.find\_element\_by\_xpath('//\*[@id="pl\_login\_form"]/div/div[3]/div[2]/div/input')  
 passwork.clear()  
 passwork.send\_keys('25257758')  
 time.sleep(5)  
 dirver.find\_element\_by\_xpath('//\*[@id="pl\_login\_form"]/div/div[3]/div[6]/a').click()  
except Exception as e:  
 print(e)  
finally:  
 time.sleep(10)  
 dirver.close()