Session关联

讲师：潘sir

session会话管理

原理

- session是一种管理用户状态和信息的机制
- session一般保存在服务端，而且一般保存在内存里，而cookie一般保存在客户端本地
- 客户端和服务端通过一个sessionID来进行沟通，为了防止不同的客户端之间出现重复和冲突，sessionID一般使用长随机字符，一般32位或48字节
- python的session跟上述session不是一回事，python-session相当于一个浏览器，用来模拟会话

session自动关联

- 使用session登录之后就不用手动关联cookie，session会自动化进行关联

```
import requests, re

# 相当于无界面浏览器，自动保存cookie到内存
s = requests.session()

url = 'http://127.0.0.1:5001/login'

data = {
    "username": "dcs",
    "password": "123",
}

# get请求login页获取csrf_token
r = s.get(url)
# 获取csrf_token
csrf_token = re.findall(r'csrf_token.*?value="(.*?)">-', r.text)
data["csrf_token"] = csrf_token

# 第一次登录，session自带cookie
r2 = s.post(url, data=data)

# 访问登录后才能访问的页面，session自带cookie
url2 = 'http://127.0.0.1:5001/user/dcs'
r3 = s.get(url2)
print(r3.status_code)
# print(r3.text)
assert "资料编辑" in r3.text
```

```
200
```

- session免登录，手动设置cookie

```
import requests

# 相当于无界面浏览器，自动保存cookie到内存
s = requests.session()

# 手动加载抓包获取到的cookie
# 1. 实例化一个cookie对象
c = requests.cookies.RequestsCookieJar()
# 2. 手动设置设置cookie
c.set('cookie', 'Hm_lvt_9669a981f0cf960985189c066a88f491=1563445548;
_ga=GA1.1.333594538.1563445549;
Hm_lpvt_9669a981f0cf960985189c066a88f491=1564994444;
Hm_lvt_46e79e71af0709a5b9106bf20cecc493=1566988188;
Hm_lpvt_46e79e71af0709a5b9106bf20cecc493=1566993846;
session=.eJwlj0mKAzEMAP_icw6WLMtWPtNYGxMCM9CdnEL-ng5zr4KqV9lyj-
OnXB_7My5lu3m5FvLGCL0uS8lYQtijQhCu5Cp95eAmIK5tzq6WKtMGcgT1SkZT3XOQq0B072JfMpiq
UtNJXpWBGLCdEjkG4DL0WSGBGwpauRQ79twef_f4PXsEszvAiAGKdaiDTkirE2g4sYqss4389J5H7P
8TQOX9AdT4PrQ.EHolTA.XyoaRdRkA2_9jRQIR0fn_FFBCrQ')
# 3. 把cookie更新到session中
s.cookies.update(c)
print(s.cookies)

# 不需要登录，页可访问登录后才能访问的页面，session自带cookie
url2 = 'http://127.0.0.1:5001/user/dcs'
r3 = s.get(url2)
print(r3.status_code)
# print(r3.text)
assert "资料编辑" in r3.text
```

```
<RequestsCookieJar[<Cookie
cookie=Hm_lvt_9669a981f0cf960985189c066a88f491=1563445548;
_ga=GA1.1.333594538.1563445549;
Hm_lpvt_9669a981f0cf960985189c066a88f491=1564994444;
Hm_lvt_46e79e71af0709a5b9106bf20cecc493=1566988188;
Hm_lpvt_46e79e71af0709a5b9106bf20cecc493=1566993846;
session=.eJwlj0mKAzEMAP_icw6WLMtWPtNYGxMCM9CdnEL-ng5zr4KqV9lyj-
OnXB_7My5lu3m5FvLGCL0uS8lYQtijQhCu5Cp95eAmIK5tzq6WKtMGcgT1SkZT3XOQq0B072JfMpiq
UtNJXpWBGLCdEjkG4DL0WSGBGwpauRQ79twef_f4PXsEszvAiAGKdaiDTkirE2g4sYqss4389J5H7P
8TQOX9AdT4PrQ.EHolTA.XyoaRdRkA2_9jRQIR0fn_FFBCrQ for />]>
200
```

小结

目的：为了访问登录后的页面，进行一些校验操作，但可能遇到以下问题：

1. 有效期：cookie有效期不定，有的1天，有的1个月？考虑使用一个脚本定时去更新cookies；或每次执行前去获取一次cookie
2. 验证码：验证码就是为了防止你搞自动化的，如果在测试环境，就要弄一个万能验证码
3. 碰到某些请求分析困难的场景，可以考虑使用selenium
    1. Selenuim去打开浏览器
    2. 定位账号/密码，登录
    3. 获取cookies
    4. 把cookies传入session
    5. 后续使用session去请求

Token登录

简介

- 并不是所有的网站都使用cookie，某些网站使用token
- 对比登录前后的抓包数据，即可知道是否是cookie登录
- 一般app的登录，都是使用token校验

Post请求使用token

- 如何获取：
    1. 手动请求，抓包分析token在哪个位置，可能出现在：json返回内容；html返回内容；cookies里
    2. json转换字典，html使用正则匹配，cookies使用对象
- 请求时token存放的位置：请求头；url地址；请求体body；cookies，真实请求放哪里，代码就模拟放哪里

```
import requests

s = requests.session()
```

```
url = '登录地址'
body = {
    "username": "panwj",
    "password": "123",
}

r = s.post(url, data=body)
# 获取token，一般在返回内容中
token = r.json()["token"]

# 1. token放头部
h1 = {
    "token": token
}
# 更新头
s.headers.update(h1)
url1 = '登录之后的页面'
body1 = {
    'xxx': 'hhh',
}
r1 = s.post(url1, data=body1)

# 2. token在url后面params
url2 = '登录之后的页面地址'
params = {
    'xxx': 'hhh',
    'token': token
}
r2 = s.get(url2, params=params)

# 3. token在body中
url3 = '登录之后的页面地址'
body3 = {
    'xxx': 'hhh',
    'token': token
}
r3 = s.get(url3, data=body3)

# 4. token放在cookies里
url4 = '登录之后的页面地址'
c = requests.cookies.RequestsCookieJar()
c.set("token", token)
s.cookies.update(c)
body4 = {
    'xxx': 'hhh',
}
r4 = s.get(url4, data=body4)
```

- csrf_token也是类似的功能，只是名称不一样，例如还有：JSESSIONID
- 一般验证登录有三种方式，由开发设计，但要会自己抓包分析
    1. 仅cookie
    2. 仅token(csrf_token/JESSIONID)
    3. cookie + token

TSMS登录示例

登录分析

1. 打开charles抓包工具，
2. 打开：http://127.0.0.1:5001/login ，使用错误密码登录两次
3. 在raw中查看变更的参数，有可能出现在headers，也有可能出现在body，本例是body的csrf_token在变化，不变的参数直接复制粘贴，但变动的参数需要动态获取
4. 分析动态参数csrf_token从哪里来的，一般来说都来自上一个页面，先刷新一下上一个页面/login，检查返回内容中是否含相关参数 csrf_token，发现html中包含该参数

登录示例

- 手动获取

```
import requests, time, re

url = 'http://127.0.0.1:5001/login'

data = {
    "username": "dcs",
    "password": "123",
}
# get请求login页获取csrf_token
r = requests.get(url)
# 获取csrf_token
csrf_token = re.findall(r'csrf_token.*?value="(.*?)">-', r.text)
data["csrf_token"] = csrf_token

# 登录
r2 = requests.post(url, data=data, cookies=r.cookies)
# 获取重定向的cookie
cks = r2.history[0].cookies

# 访问登录后才能访问的页面
url2 = 'http://127.0.0.1:5001/user/dcs'
r3 = requests.get(url2, cookies=cks)
print(r3.status_code)
# print(r3.text)
assert "资料编辑" in r3.text
```

```
200
```

- session自动关联

```
import requests, re

# 相当于无界面浏览器，自动保存cookie到内存
s = requests.session()

url = 'http://127.0.0.1:5001/login'

data = {
    "username": "dcs",
    "password": "123",
}

# get请求login页获取csrf_token
r = s.get(url)
# 获取csrf_token
# 所有的字符都是直接按照字面的意思来使用
csrf_token = re.findall(r'csrf_token.*?value="(.*?)">-', r.text)
data["csrf_token"] = csrf_token

# 第一次登录，session自带cookie
r2 = s.post(url, data=data)

# 访问登录后才能访问的页面，session自带cookie
url2 = 'http://127.0.0.1:5001/user/dcs'
r3 = s.get(url2)
print(r3.status_code)
# print(r3.text)
assert "资料编辑" in r3.text
```

```
200
```

课后题

1. 使用cookie绕过登录，先用博客园，再用其他网站
2. 学会使用session关联登录之后的请求，如：登录之后发帖，评论，回复等操作

```
import requests

s = requests.session()

# 将抓包获取到到cookie放入
h = {
```

```
    "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.67 Safari/537.36",
    "cookie":
".Cnblogs.AspNetCore.Cookies=CfDJ8D8Q4oM3DPZMgpKI1MnYlrn8v5O9GT5Lh6BDeVBm8077A
Ji3KcmKvvHibIi_cj8OYZcDdBfqUgZpzmGo615ALrhYKXlhv99o8WiSBEGF9TGletu95BDNUHEt9QQ
ahrREuBDwVgNpCLasVa2uhjm0jRa8dPur-LWOfUB_-ehpvDjWK-
V9WIlTwOaHoT59m1YOi38Pg45wrMihoW0eOUApU_3BLGLZAaF6pMXgyFF6vlPJ_qnYXv2Z2xrPgE1z
pGOE3vEejLN4cJ0PAbvtEP4VMdpFpk1_ngq3GvEL9osQvKOTWnyTwOcvxFkT1aynLDIOQRQgr376l8
kqUW5qwpafqceMPYplHkR10k1HBgHW2AaRHOeo8348JcEz63hd6QRIjq0cSu80PQk2WqTobyOG2zE9
rvXaL0dqcEwoNN3wwxQnPu2Qgmf3JYEzX8hFxtXGJHLW6n2dkP8SAQYLQyuUhSY",
}
# 将headers更新到session中
s.headers.update(h)
# 登录关注页
url = 'https://home.cnblogs.com/followers/'
r = s.get(url, verify=False)
print(r.status_code)
print(r.text)
```