requests_html请求库的使用

安装：pip3 install requests-html -i 换源地址

学习目标：

掌握`requests_html`的使用

如果把requests比喻成人工刨地的工具，那么requests_html就是开拖拉机刨地

> **官方文档**：https://requests-html.kennethreitz.org/
>
> **更多用法**：https://www.jianshu.com/p/dd234efeac3a
>
> **官方文档**：https://pypi.org/project/requests-html/
>
> 上述站点访问不了，试试翻墙之后访问或者去找类型的文档笔记，都有很多现成的

1. requests_html简介

Requests是模拟HTTP的测试库，但是Requests只负责网络请求，不会对响应结果进行解析。而该库的作者后来基于现有的框架进行二次封装，又发布了一个更好用的**Requests-html**库用于解析HTML。（该库最早发现于2018年7月（谷歌提供），**只支持python3.6及以上版本**）

request-html的牛逼之处:它集成了request库,beautifulsoup库,pyquery库,浏览器内核--ua

Requests-html**具有以下特性**

- 完全支持 JavaScript
- CSS、xPath 选择器
- 模拟用户代理
- 自动跟踪重定向
- 连接池和 cookie 持久化

2. 请求数据

```
# 1.导包
from requests_html import HTMLSession
# 2.实例化(注意，要加上括号)
session = HTMLSession()
# 3.准备start_url
url = 'https://www.baidu.com'
# 4.发送请求(get)
response_get = session.get(url)
# post请求
response_post = session.post(url,
data=data)
# request(请求选择化)
response =
session.request(method='get'/'post',
url=url)
```

3. response的属性与requests的response属性用法一致

```
response.url: 当前路径

response.text:  文本

response.encoding = 'gbk': 编码

response.content:  二进制的响应内容

response.json ==>json.loads(r.text)

response.status_code: 返回状态码

response.headers:  返回响应头

response.cookies:  返回cookies

response.history:  返回响应历史
```

若，我们需要获取某个网页的源码，使用html方法即可

```
# 1.导包
from requests_html import HTMLSession
# 2.实例化(注意，要加上括号)
session = HTMLSession()
# 3.准备start_url
url = 'https://www.baidu.com'
# 4.发送请求(get)
response_get = session.get(url)
# 5.获取html源码
html_str = response_get.html.html
print(html_str)
```

4. response.html的对象属性

```
response.html.absolute_links: 绝对链接
response.links:相对链接
response.base_url: 基本路径
response.html.html: 网页源码
response.html.text: 网页文本
response.html.encoding = 'gbk'
response.html.raw_html:页面的二进制流
```

除了以上的操作外，response的操作属性还可以和requests的请求响应的response属性通用

5. response与Xpath语法

```python
# 1.导包
from requests_html import HTMLSession
# 2.实例化(注意，要加上括号)
session = HTMLSession()
# 3.准备start_url
url = 'https://www.baidu.com'
# 4.发送请求(get)
response_get = session.get(url)
# 5.获取html源码
html_str = response_get.html.html
print(html_str)
# 6.使用xpath语法
data = response_get.html.xpath('xpath语法')
print(data)
```

演示

```
import requests_html
from requests_html import HTMLSession

# 获取请求对象
session = HTMLSession()

#发送get请求
sina =
session.get('https://news.sina.com.cn/')

# 获取响应文本信息
print(sina.text)

# 获取链接（links与abolute_links）

# 得到所有的链接，返回的是一个set集合
print(sina.html.links)

# 若获取的链接中有相对路径，我们还可以通过
absolute_links获取所有绝对链接
print(sina.html.absolute_links)

# request-html支持CSS选择器和XPATH两种语法来选
取HTML元素。首先先来看看CSS选择器语法，它需要使用
HTML的 find 函数来查找元素。
'''
    CSS选择器 and XPATH
```

```
        1.通过css选择器选取一个Element对象
        2.获取一个Element对象内的文本内容
        3.获取一个Element对象的所有attributes
        4.渲染出一个Element对象的HTML内容
        5.获取Element对象内的特定子Element对象，返回列表
        6.在获取的页面中通过search查找文本
        7.支持XPath
        8.获取到只包含某些文本的Element对象
'''
#  获取id为content-left的div标签，并且返回一个对象
content = sina.html.find('div#content-left', first=True)

# 获取Element对象内的指定的所有子Element对象，返回列表
a_s = content.find('a')
print(a_s)

#  获取content内所有文本
print(content.text)

#  获取content内所有属性
print(content.attrs)

#获取单个属性
href = content.attrs['href']
```