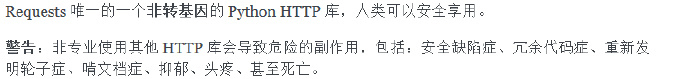
窗体顶端

***Requests模块***

   这个库的标准文档有个极其幽默的地方就是它的中文翻译，我就截取个开头部分，如下图：

​

   是不是很搞笑，在正文中还有许多，管中窥豹，可见一斑。通过我的使用，感觉Requests库的确是给那些初学者，入门小白，非专业人士使用的，不会产生打人，砸键盘，脱发等一系列反人类行为，很好的使社会安全等级有又提升了一个档次，让人类社会向路不拾遗，夜不闭户又迈进了一步。（手动笑脸）

   咱们先从安装Resquests是库谈起，安装只需在Windows的命令框中输入pip install requests,然后自动完成安装。这个库的许多方法我没有怎么用到，我只把一些入门及常用的方法介绍一遍，其他的请转移到www.python-requests.org

   这个阵地进行查看。

   Requests库使得人们可以非常方便的模拟浏览器的行为，去发送请求，并获得响应。

   首先通过import在程序中导入requests模块：

importrequests

   接下来为get请求的实列：

# 只有一个url参数的get请求

response =requests.get('https://docs.python.org')

# 并设置超时时间timeout

response =requests.get('https://docs.python.org', timeout = 1)

# 查看发送请求的url地址

print('请求的url地址' + response.url)

# 查看当前返回状态编码，传输文件编码形式

print('返回状态编码' + response.encoding)

# 查看当前返回内容，text 返回字符串，content 返回字节流

print('text返回内容  ' +response.text)

print('content返回内容 ' + str(response.content))

# 获取服务器返回的原始数据，html格式文件

# data =requests.get('http://www,baidu.com', stream = True)

#print(data.raw.read())

# 发送有多个参数的get请求

Params ={'key1': 'value1', 'key2': 'value2'}

baiDU\_dictParams\_response= requests.get('http://www.baidu.com', params=Params)

# 发送list格式参数的get请求

print("发送list格式参数的get请求:  ")

ListParams= {'key1': 'value1', 'key2': ['value2', 'value3']}

listParams\_response= requests.get('https://docs.python.org', params=ListParams)

# 查看发送请求的url地址

print('带list参数的get请求地址为： ' +listParams\_response.url)

post请求方式：

# 请求参数通过附加data方式传输

# 方式一，字典格式

Post\_Response= requests.post('https://blog.csdn.net',

                           data = {'key':'value'})

print('普通参数请求返回状态码：  ' +str(Post\_Response.status\_code))

# 方式二  json格式

jsonParams= {'key': 'value'}

postJsonResponse= requests.post('https://blog.csdn.net', data=json.dumps(jsonParams))

print('json参数请求返回状态码为：    ' +str(postJsonResponse.status\_code))

# 方式三 发送文件(该文件同级目录下需要有test.csv文件 )rb 只读模式 wb 若没有自动创建

files ={'file': open('D:\fff.csv', 'rb')}

fileResponse= requests.post('http://pythontab.com/postTest', files=files)

# 自定义headers 并发送，这是你自己浏览器的标识，在检查源码中可以获得

headers ={'user-agent': ' Mozilla/5.0(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134 '}

custom\_headers\_response= requests.get('https://blog.csdn.net', headers=headers)

print('自定义header发送请求状态码为：' + str(custom\_headers\_response.status\_code))

# 带cookie参数的请求：

cookies\_response= requests.get('https://blog.csdn.net')

# 自定义Cookies 并发送

cookies ={'user-cookies': 'myCookies'}

custom\_cookies\_response= requests.get('http://pythontab.com/testLogin', cookies=cookies)

print('自定义Cookies发送请求状态码为：' + str(custom\_cookies\_response.status\_code))

# 通过requests获取session

session =requests.Session()

# 举例：登录名　密码　 key为登陆表单中对应的input的name值

login\_data= {'email': 'email@example.com', 'password': 'password'}

# 发送数据

session.post("http://pythontab.com/testLogin",login\_data)

# 获取发送的session

session\_response= session.get('http://pythontab.com/notification/')

print('session请求返回的状态码为：' + str(session\_response.status\_code))

# 将页面以html的形式下载下来

# utf-8可以使中文字显示出来

baiDuHtmlContent= requests.get("https://www.zhihu.com")

withopen("百度.html", "wb")as html:

    html.write(baiDuHtmlContent.content.decode(“utf-8”))

html.close()

   这些代码没有什么特别的地方，只要按照requests模块中方法的要求，找准输入参数和输出内容，就能灵活运用这些内容。

   这里说一下代理服务器的相关内容。

   其实如果你想去用Google搜索一些内容，上youtobe看看电影，与Facebook的外国友人聊聊天，这些都需要一个叫做VPN的东西，就是将你的请求挂在在一个代理服务器的上面，然后代理服务器去向你指定的网站发出请求，获得一个缓存，并把这个缓存的数据返回给你。相当于一个互联网与你之间的一个中介，就像你租房子要通过58同城这样的中介网站一样。