requests模块

Requests: 让 HTTP 服务人类 虽然Python的标准库中 urllib 模块已经包含了平常我们使用的大多数功能，但是它的 API 使用起来让人感觉不太好，而 Requests 自称 "HTTP for Humans"，说明使用更简洁方便。

> Requests 唯一的一个非转基因的 Python HTTP 库，人类可以安全享用：）

Requests 继承了urllib的所有特性。Requests支持HTTP连接保持和连接池，支持使用cookie保持会话，支持文件上传，支持自动确定响应内容的编码，支持国际化的 URL 和 POST 数据自动编码。

Requests的文档非常完备，中文文档也相当不错。Requests能完全满足当前网络的需求，支持Python 2.6--3.5，而且能在PyPy下完美运行。

开源地址：https://github.com/kennethreitz/requests

中文文档 API： http://docs.pythonrequests.org/zh_CN/latest/index.html

安装方式

利用pip安装

```
$ pip install requests -i https://pypi.douban.com/simple
```

基本GET请求（headers参数 和 parmas参数）

1. 最基本的GET请求可以直接用get方法

```
response = requests.get("http://www.baidu.com/")

# 也可以这么写
# response = requests.request("get", "http://www.baidu.com/")
```

2. 添加 headers 和 查询参数

如果想添加 headers，可以传入`headers`参数来增加请求头中的headers信息。如果要将参数放在url中传递，可以利用`params`参数。

```
import requests

kw = {'wd':'长城'}

headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.99 Safari/537.36"
}

# params 接收一个字典或者字符串的查询参数，字典类型自动转换为url编码，不需要urlencode()
response = requests.get("http://www.baidu.com/s?", params = kw, headers = headers)

# 查看响应内容，response.text 返回的是str格式的数据
print(response.text)

# 查看响应内容，response.content返回的字节流数据
print(respones.content)

# 查看完整url地址
print(response.url)

# 查看响应头部字符编码
print(response.encoding)
```

```
23
24 # 查看响应码
25 print(response.status_code)
```

运行结果

```
1 ......
2
3 ......
4
5 'http://www.baidu.com/s?wd=%E9%95%BF%E5%9F%8E'
6
7 'utf-8'
8
9 200
```

- 使用response.text 时，Requests 会基于 HTTP 响应的文本编码自动解码响应内容，大多数 Unicode 字符集都能被无缝地解码。
- 使用response.content 时，返回的是服务器响应数据的原始二进制字节流，可以用来保存图片等二进制文件。

为什么要使用User-Agent?

```
1  In [12]: resp1 = requests.get("https://www.baidu.com")
2
3  In [13]: resp1.status_code
4  Out[13]: 200
5
6  In [14]: len(resp1.text)
7  Out[14]: 2443
8
9  In [15]: headers
10 Out[15]: {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0;
   Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/54.0.2840.99 Safari/537.36'}
11
```

```
In [16]: resp2 = requests.get("https://www.baidu.com", headers=headers)

In [17]: resp2.status_code
Out[17]: 200

In [18]: len(resp2.text)
Out[18]: 156612
```

3. POST请求

github登录

```
import requests

login_url = "https://github.com/session"

headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.170 Safari/537.36",
    "Cookie": "_ga=GA1.2.1855430798.1461857641; _octo=GH1.1.783519559.1525492869; tz=Asia%2FShanghai; logged_in=no; _gat=1; _gh_sess=TTZZNnVPOTRwTlI2YW1UOWFqTUZwa1JHVENFOVpjYWxoekJLbS9GTGFQQkR4ODZVSTIrV1NTWUFQS0dETUhURmg4YVR3bGxjV1hJT2dMZ2NIOEptZlREUUxrOXV0eG1EcG5kUi9adUZQamFpWmxmMHVhY04vckJVWkxXbkNVa3ZHNCtjekF2WEVnMGVzaElqMnBpVUIvVGVSNzJmdjNQMFFxYWpONE1HMks2eDhpVzZ0Wk9ZQUZLMVJOOTRsYXJWUjV6VUNmRFhyaHlYczUzdUozSWR5M210akh3dkcvaXRhY2ZmanRNRC9IbElUMm5OSmkzVDhtbEwvSEdGWFMvd0xySWIxcFRrbGZ0RDQwQit5eGUvaGZKdWp5U1dYSnZ0VzRRb1FqUThZZXlreTBNU1RicUhheGJGQjFRcDlnN1N2b2RXRXRsT21lRFB5Q3RFSHQ3V1FpR05QSlo2TVBic2o3R0hDaUZOSmhST0l2ZXdabHVzVEdBcGdMRWpiS2lXME5jOWV2WTFJckFGUXI2WHpjeTZ5ZXpXMUJvSXlpdmRpZWNONUhFejFUMD0tLVhyaTZJdlZBMUdNWnBNU0QrWDFFQ1E9PQ%3D%3D--ab7943f35d872df42bb86b8086b5cec50d3ef0ce"
}
```

```
9
10 post_data = {
11     "commit": "Sign in",
12     "utf8": "✓",
13     "authenticity_token":
"UFKuL5RE8DTUXAc0cddcawtX3gWADuVQInNPqIacBfESUsUZwZ8jNQ24sYp
QVHS6vqFXlci9FqeTV9aZ+wqa+Q==",
14     "login": "czwspider",
15     "password": "qwer1234"
16 }
17
18 response = requests.post(login_url, post_data,
headers=headers)
19
20 print(response.status_code)
21
22 with open("github_login.html", "wb") as f:
23     f.write(response.content)
```

小栗子

通过requests获取新浪首页

```
1 import  requests
2 response = requests.get("http://www.sina.com")
3 print(response.request.headers)
4 print(response.content.decode())
```

结果

```
{'User-Agent': 'python-requests/2.12.4', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-alive'}
<!DOCTYPE html>
<!-- [ published at 2017-06-09 15:15:23 ] -->
<html>
<head>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>新浪首页</title>
    <meta name="keywords" content="新浪,新浪网,SINA,sina,sina.com.cn,新浪首页,门户,资讯" />
  ...
```

```
import  requests
response = requests.get("http://www.sina.com")
print(response.request.headers)
print(response.text)
```

结果

```
{'User-Agent': 'python-requests/2.12.4', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-alive'}
<!DOCTYPE html>
<!-- [ published at 2017-06-09 15:18:10 ] -->
<html>
<head>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>æ–°æµªé¦–é¡µ</title>
    <meta name="keywords" content="æ–°æµª,æ–°æµªç½‘,SINA,sina,sina.com.cn,æ–°æµªé¦–é¡µ,é—¨æˆ·,èµ„è®¯" />
    <meta name="description" content="æ–°æµªç½‘ä¸ºå…¨ç•ƒç”¨æˆ·24å°•æ—¶æ••ä¾›å…¨é•¢å•Šæ—¶çš„ä¸•æ–‡èµ„è®¯ï¼Œå†…å®¹è¦†ç›–å›½å†…å¤–ç ªå•‘æ–°é—»äº‹ä»¶ã€•ä½“å••›èµ›äº‹ã€•å¨±ä¹•æ—¶å°šã€•äº§ä¸šèµ„è®¯ã€•å®žç”¨ä¿¡æ•¯ç•‰ï¼Œè®¾æœ‰æ–°é—»ã€•ä½“è‚²ã€•å¨±ä¹•ã€•è´¢ç»•ã€•ç§‘æŠ€ã€•æˆ¿äº§ã€•æ±½è½¦ç•‰30å¤šä¸ªå†…å®¹é¢‘é•“ï¼Œå•Œæ—¶å¼€è®¾å•šå®¢ã€•è§†é¢‘ã€•è®ºå••›ç•‰è‡ªç”±äº’åŠ¨äº¤æµ•ç©ºé—´ã€‚" />
    <link rel="mask-icon" sizes="any" href="//www.sina.com.cn/favicon.svg" color="red">
```

产生问题的原因分析

1. requests默认自带的Accept-Encoding导致或者新浪默认发送的就是压缩之后的网页
2. 但是为什么content.decode()没有问题，因为requests，自带解压压缩网页的功能
3. 当收到一个响应时，Requests 会猜测响应的编码方式，由于在你调用 response.text 方法时对响应进行解码。Requests 首先在 HTTP 头部检测是否存在指定的编码方式，如果不存在，则会使用 chardet.detect 来尝试猜测编码方式（存在误差）

4. 更推荐使用response.content.deocde()

Cookies 和 Session

Cookies

如果一个响应中包含了cookie，那么我们可以利用 cookies参数拿到：

```
1
2  import requests
3
4  response = requests.get("http://www.baidu.com/")
5
6  # 返回CookieJar对象:
7  cookiejar = response.cookies
8
9  # 将CookieJar转为字典:
10 cookiedict = requests.utils.dict_from_cookiejar(cookiejar)
11
12 print (cookiejar)
13
14 print (cookiedict)
```

运行结果：

```
1 <RequestsCookieJar[<Cookie BDORZ=27315 for .baidu.com/>]>
2
3 {'BDORZ': '27315'}
```

实现github登录并跳转

```
1
2  #!/usr/bin/python
3  # -*- coding: utf-8 -*-
4
```

```
# 模拟 github 登录

# 0. 分析爬虫
'''
登录地址
确定请求地址 https://github.com/session
确定请求方式 POST
确定请求的内容
commit: Sign in
utf8: ✓
authenticity_token: FKPt8/jlSD6VqqKbJqQUylCZaArCLMEhyIYWtA12LSzK47nyaPOs8IoIZ04o5AGJiQIc04jX9b0lsWuETzc8+g==
login: czwspider
password: qwer1234
当请求成功 获取 页面的 Cookie 值并且，保存下来以后发送的请求都携带这个Cookie值
'''

import requests
import requests.utils

# 1。定义请求参数
login_url = "https://github.com/session"

headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.170 Safari/537.36",
```

```
    "Cookie": "_ga=GA1.2.1855430798.1461857641; _octo=GH1.1.783519559.1525492869; tz=Asia%2FShanghai; logged_in=no; _gat=1; _gh_sess=TTZZNnVPOTRwTlI2YW1UOWFqTUZwa1JHVENFOVpjYWxoekJLbS9GTGFQQkR4ODZVSTIrV1NTWUFQS0dETUhURmg4YVR3bGxjV1hJT2dMZ2NIOEptZlREUUxrOXV0eG1EcG5kUi9adUZQamFpWmxmMHVhY04vckJVWkxXbkNVa3ZHNCtjekF2WEVnMGVzaElqMnBpVUIvVGVSNzJmdjNQMFFxYWpONE1HMks2eDhpVzZ0Wk9ZQUZLMVJOOTRsYXJWUjV6VUNmRFhyaHlYczUzdUozSWR5M210akh3dkcvaXRhY2ZmanRNRC9IbElUMm5OSmkzVDhtbEwvSEdGWFMvd0xySWIxcFRrbGZ0RDQwQit5eGUvaGZKdWp5U1dYSnZ0VzRRb1FqUThZZXlreTBNU1RicUheGJGQjFRcDlnN1N2b2RXRXRsT21lRFB5Q3RFSHQ3V1FpR05QSlo2TVBic2o3R0hDaUZOSmhST0l2ZXdabHVzVEdBcGdMRWpiS2lXME5jOWV2WTFJckFGUXI2WHpjeTZ5ZXpXMUJvSXlpdmRpZWNONUhFejFUMD0tLVhyaTZJdlZBMUdNWnBNU0QrWDFFQ1E9PQ%3D%3D--ab7943f35d872df42bb86b8086b5cec50d3ef0ce"
}
post_data = {
    "commit": "Sign in",
    "utf8": "✓",
    "authenticity_token": "UFKuL5RE8DTUXAc0cddcawtX3gWADuVQInNPqIacBfESUsUZwZ8jNQ24sYpQVHS6vqFXlci9FqeTV9aZ+wqa+Q==",
    "login": "czwspider",
    "password": "qwer1234"
}


# 获取session对象通过session对象进行请求，session对象的作用就是自动记录Cookies值，代码中不需要关心
session = requests.session()
# 2. 发送请求获取响应
response = session.post(login_url, post_data, headers=headers)


# 3. 从响应中获取Cookies值并且在以后的通讯中都使用这个Cookies值
```

```python
# requests.utils.dict_from_cookiejar 用于把Cookie对象转换成词典对象
#
print(requests.utils.dict_from_cookiejar(response.cookies))

# 4. 爬取
setting_headers = {
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.170 Safari/537.36",
}
setting_url = "https://github.com/settings/profile"
settting_resp = session.get(setting_url, headers=setting_headers)

# print(response.content.decode('utf-8'))
print(settting_resp.status_code)

with open('github_login_03.html', 'wb') as f:
    f.write(settting_resp.content)


```

练一练

- 使用requests爬取任意输入的百度贴吧的网页，并保存到本地