requests

requests是python的一个HTTP客户端库，跟urllib，urllib2类似，那为什么要用requests而不用urllib2呢？官方文档中是这样说明

的：

*python的标准库urllib2提供了大部分需要的HTTP功能，但是API太逆天了，一个简单的功能就需要一大堆代码。*

我也看了下requests的文档，确实很简单，适合我这种懒人。下面就是一些简单指南。

*强烈推荐！requests官方文档已有了中文版，请见http://cn.python-requests.org/zh\_CN/latest/*

1. **安装**

安装很简单，我是win系统，就在[这里](http://docs.python-requests.org/en/latest/user/install/#install)下载了安装包（网页中download the tarball处链接），然后$ python setup.py install就装好了。当然，有easy\_install或pip的朋友可以直接使用：easy\_install requests或者pip install requests来安装。至于linux用户，[这个页面](http://docs.python-requests.org/en/latest/user/install.html#install)还有其他安装方法。测试：在IDLE中输入import requests，如果没提示错误，那说明已经安装成功了！



1. **小试牛刀**

>>>**import**requests

* r = requests.get('http://www.zhidaow.com') # 发送请求
* r.status\_code # 返回码

200

>>> r.headers['content-type'] # 返回头部信息

'text/html; charset=utf8'

>>> r.encoding # 编码信息

'utf-8'

* r.text #内容部分（PS，由于编码问题，建议这里使用r.content）

u'<!DOCTYPE html>\n<html xmlns="http://www.w3.org/1999/xhtml"...'

...

是不是很简单？比urllib2和urllib简单直观的多？！那请接着看快速指南吧。

1. **快速指南**

**3.1 发送请求**

发送请求很简单的，首先要导入requests模块：

>>>**import**requests

接下来让我们获取一个网页，例如我个人博客的首页：

>>>r = requests.get('http://www.zhidaow.com')

接下来，我们就可以使用这个r的各种方法和函数了。另外，HTTP请求还有很多类型，比如

POST,PUT,DELETE,HEAD,OPTIONS。也都可以用同样的方式实现：

* r = requests.post("http://httpbin.org/post")
* r = requests.put("http://httpbin.org/put")
* r = requests.delete("http://httpbin.org/delete")
* r = requests.head("http://httpbin.org/get")
* r = requests.options("http://httpbin.org/get")

因为目前我还没用到这些，所以没有深入研究。

**3.2 在URLs中传递参数**

有时候我们需要在URL中传递参数，比如在采集百度搜索结果时，我们wd参数（搜索词）和rn参数（搜素结果数量），你可以手工组成URL，requests也提供了一种看起来很NB的方法：

* payload = {'wd': '张亚楠', 'rn': '100'}
* r = requests.get("http://www.baidu.com/s", params=payload)
* **print** r.urlu'http://www.baidu.com/s?rn=100&wd=%E5%BC%A0%E4%BA%9A%E6%A5%A0'

上面wd=的乱码就是“张亚楠”的转码形式。（好像参数按照首字母进行了排序。）



**3.3 获取响应内容**

可以通过r.text来获取网页的内容。

* r = requests.get('https://www.zhidaow.com')
* r.text

u'<!DOCTYPE html>\n<html xmlns="http://www.w3.org/1999/xhtml"...'

文档里说，requests会自动将内容转码。大多数unicode字体都会无缝转码。但我在cygwin下使用时老是出

现UnicodeEncodeError错误，郁闷。倒是在python的IDLE中完全正常。另外，还可以通过r.content来获取页面内容。

* r = requests.get('https://www.zhidaow.com')
* r.content

b'<!DOCTYPE html>\n<html xmlns="http://www.w3.org/1999/xhtml"...'

文档中说r.content是以字节的方式去显示，所以在IDLE中以b开头。但我在cygwin中用起来并没有，下载网页正好。所以就替代了urllib2的urllib2.urlopen(url).read()功能。（基本上是我用的最多的一个功能。）

**3.4 获取网页编码**

可以使用r.encoding来获取网页编码。

* r = requests.get('http://www.zhidaow.com')
* r.encoding

'utf-8'

当你发送请求时，requests会根据HTTP头部来猜测网页编码，当你使用r.text时，requests就会使用这个编码。当然你还可以修改requests的编码形式。

* r = requests.get('http://www.zhidaow.com')
* r.encoding

'utf-8'

>>>r.encoding = 'ISO-8859-1'

像上面的例子，对encoding修改后就直接会用修改后的编码去获取网页内容。

**3.5 json**

像urllib和urllib2，如果用到json，就要引入新模块，如json和simplejson，但在requests中已经有了内置的函数，r.json()。就拿查询IP的API来说：

>>>r = requests.get('http://ip.taobao.com/service/getIpInfo.php?ip=122.88.60.28')

>>>r.json()['data']['country']

'中国'

**3.6 网页状态码**

我们可以用r.status\_code来检查网页的状态码。

>>>r = requests.get('http://www.mengtiankong.com')

>>>r.status\_code

200

>>>r = requests.get('http://www.mengtiankong.com/123123/')

>>>r.status\_code

404

>>>r = requests.get('http://www.baidu.com/link?url=QeTRFOS7TuUQRppa0wlTJJr6FfIYI1DJprJukx4Qy0XnsDO\_s9baoO8u1wvjxgqN') >>>r.url

u'http://www.zhidaow.com/

>>>r.status\_code

200

前两个例子很正常，能正常打开的返回200，不能正常打开的返回404。但第三个就有点奇怪了，那个是百度搜索结果中的302跳转地址，但状态码显示是200，接下来我用了一招让他原形毕露：

>>>r.history

(<Response [302]>,)

这里能看出他是使用了302跳转。也许有人认为这样可以通过判断和正则来获取跳转的状态码了，其实还有个更简单的方法：

>>>r = requests.get('http://www.baidu.com/link?url=QeTRFOS7TuUQRppa0wlTJJr6FfIYI1DJprJukx4Qy0XnsDO\_s9baoO8u1wvjxgqN',

allow\_redirects = False)

>>>r.status\_code

302

只要加上一个参数allow\_redirects，禁止了跳转，就直接出现跳转的状态码了，好用吧？我也利用这个在最后一掌做了个简单的获取网页状态码的小应用，原理就是这个。

**3.7 响应头内容**

可以通过r.headers来获取响应头内容。

>>>r = requests.get('http://www.zhidaow.com')

* r.headers

{

'content-encoding': 'gzip',

'transfer-encoding': 'chunked',

'content-type': 'text/html; charset=utf-8';

...

}

可以看到是以字典的形式返回了全部内容，我们也可以访问部分内容。

* r.headers['Content-Type']



'text/html; charset=utf-8'

* r.headers.get('content-type') 'text/html; charset=utf-8'

**3.8 设置超时时间**

我们可以通过timeout属性设置超时时间，一旦超过这个时间还没获得响应内容，就会提示错误。

* requests.get('http://github.com', timeout=0.001) Traceback (most recent call last):

File "<stdin>", line 1, in <module>

requests.exceptions.Timeout: HTTPConnectionPool(host='github.com', port=80): Request timed out. (timeout=0.001)

**3.9 代理访问**

采集时为避免被封IP，经常会使用代理。requests也有相应的proxies属性。

**import**requests

proxies = {

"http": "http://10.10.1.10:3128",

"https": "http://10.10.1.10:1080",

}

requests.get("http://www.zhidaow.com", proxies=proxies)

如果代理需要账户和密码，则需这样：

proxies = {

"http": "http://user:pass@10.10.1.10:3128/",

}

**3.10 请求头内容**

请求头内容可以用r.request.headers来获取。

>>> r.request.headers

{'Accept-Encoding': 'identity, deflate, compress, gzip',

'Accept': '\*/\*', 'User-Agent': 'python-requests/1.2.3 CPython/2.7.3 Windows/XP'}

**3.11 自定义请求头部**

伪装请求头部是采集时经常用的，我们可以用这个方法来隐藏：

r = requests.get('http://www.zhidaow.com')

**print** r.request.headers['User-Agent']



#python-requests/1.2.3 CPython/2.7.3 Windows/XP

headers = {'User-Agent': 'alexkh'}

r = requests.get('http://www.zhidaow.com', headers = headers)

**print** r.request.headers['User-Agent']



#alexkh

**3.12 持久连接keep-alive**

*requests的keep-alive是基于urllib3，同一会话内的持久连接完全是自动的。同一会话内的所有请求都会自动使用恰当的*

*连接。*

也就是说，你无需任何设置，requests会自动实现keep-alive。

1. **简单应用**

**4.1 获取网页返回码**

**def**get\_status(url):

r = requests.get(url, allow\_redirects = False)



**return** r.status\_code



**print** get\_status('http://www.zhidaow.com')



#200

**print** get\_status('http://www.zhidaow.com/hi404/')



#404

**print** get\_status('http://mengtiankong.com')



#301

**print** get\_status('http://www.baidu.com/link?url=QeTRFOS7TuUQRppa0wlTJJr6FfIYI1DJprJukx4Qy0XnsDO\_s9baoO8u1wvjxgqN')



#302

**print** get\_status('http://www.huiya56.com/com8.intre.asp?46981.html')



#500

**后记**

1、官方文档requests的具体安装过程请看：http://docs.python-requests.org/en/latest/user/install.html#installrequests的官方指南文

档：http://docs.python-requests.org/en/latest/user/quickstart.htmlrequests的高级指南文档：http://docs.python-requests.org/en/latest/user/advanced.html#advanced2、本文内容部分翻译自官方文档，部分自己归纳。3、大多数用的IDLE格式，累死了，下次直接用编辑器格式，这样更符合我的习惯。4、还是那句话，有问题留言或email。5、图注：requests官方文档上的一只老鳖。