

细说 Python 的三大网络请求库 urllib、urllib2、urllib3

作者: [YYJeffrey](#)

原文链接: <https://ld246.com/article/1574254515672>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

不知道有没有人跟我一样想过这个问题，Python怎么会有那么多网络请求的库urllib、urllib2...这我底要用哪一个，几个库又有什么区别，真是让人头大，下文我就针对urllib、urllib2、urllib3来做一详细的阐述。

简介

urllib、urllib2、urllib3均能通过网络访问互联网上的资源文件，它们通过使用统一资源定位符（UR）并结合re模块完成很多意想不到的操作。

1. urllib：Python2和Python3内置的网络请求库，Python3的urllib实际是Python2版本中urllib2和urllib3的合并
2. urllib2：它只存在于Python2版本的内置库中，功能与urllib基本类似，主要上urllib的增强
3. urllib3：Python2和Python3均可以使用，但这不是标准库，需要使用pip安装使用，urllib3提供线程安全池和文件post等

注意：在Python2中urllib和urllib2一般搭配使用的（不然Python2整两个内置库干嘛），urllib具有urllib2没有的功能，而urllib2具有urllib没有的功能。

urllib

urllib这个Python标准库基本上包含了基础的网络请求功能，以下urllib的演示均为Python3中的用法Python2中单独用urllib会比较难受，上面也提到了它最好配合urllib2一起使用。

urllib发起GET请求

urlopen()方法发起请求，read()方法获取网页数据

```
from urllib import request
```

```
res = request.urlopen("http://httpbin.org/get")
print(res.read().decode()) # read()方法读取的数据是bytes的二进制格式，需要解码
```

urllib发起POST请求

urlopen()默认是GET方式请求，当传入data参数时会发起POST请求，此时传递的参数必须是bytes式

```
from urllib import request~~~~
```

```
res = request.urlopen("http://httpbin.org/post", data=b'hello=world')
print(res.read().decode())
```

urllib为请求添加Headers

通过urllib发起的请求默认的头信息是"User-Agent": "Python-urllib/3.6"，一般网站会验证请求头的合法性，如果需要修改可以通过urllib.request中的Request对象

```
from urllib import request
```

```
url = "http://httpbin.org/get"
```

```
headers = {'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/537.36'}
```

```
req = request.Request(url=url, headers=headers) # 传递的Request对象
res = request.urlopen(req)
print(res.read().decode())
```

此时httpbin网站返回的内容如下：

```
{
  "args": {},
  "headers": {
    "Accept-Encoding": "identity",
    "Host": "httpbin.org",
    "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/537.36"
  },
  "origin": "210.33.11.241, 210.33.11.241",
  "url": "https://httpbin.org/get"
}
```

urllib制定proxy代理

在使用爬虫时经常会配置代理隐藏我们的IP地址

```
from urllib import request
```

```
url = 'http://httpbin.org/ip'
proxy = {'http': '117.95.200.71:9999', 'https': '183.154.54.188:9999'} # 可以使用西刺代理配置
```

```
# 创建代理处理器
proxies = request.ProxyHandler(proxy)
# 创建 opener 对象
opener = request.build_opener(proxies)
```

```
resp = opener.open(url)
print(resp.read().decode())
```

urllib下载数据至本地

```
from urllib import request
```

```
url = 'http://httpbin.org/image/jpeg'
request.urlretrieve(url, '1.jpg')
```

urllib2

urllib2的用法其实基本和urllib相似，且它只存在于Python2版本，它的使用时需要配合urllib一起使用，要说它和urllib的区别还要看下面这个表，下表主要说明了在Python2中的urllib、urllib2中的方法合到Python3后方法名的区别。

Python 2

Python 3

<code>urllib.urlretrieve()</code>	<code>urllib.request.urlretrieve()</code>
<code>urllib.urlcleanup()</code>	<code>urllib.request.urlcleanup()</code>
<code>urllib.quote()</code>	<code>urllib.parse.quote()</code>
<code>urllib.quote_plus()</code>	<code>urllib.parse.quote_plus()</code>
<code>urllib.unquote()</code>	<code>urllib.parse.unquote()</code>
<code>urllib.unquote_plus()</code>	<code>urllib.parse.unquote_plus()</code>
<code>)</code>	
<code>urllib.urlencode()</code>	<code>urllib.parse.urlencode()</code>
<code>urllib.pathname2url()</code>	<code>urllib.request.pathname2url()</code>
<code>url()</code>	
<code>urllib.url2pathname()</code>	<code>urllib.request.url2pathname()</code>
<code>me()</code>	
<code>urllib.getproxies()</code>	<code>urllib.request.getproxies()</code>
<code>urllib.URLopener</code>	<code>urllib.request.URLopener</code>
<code>urllib.FancyURLopener</code>	<code>urllib.request.FancyURLopener</code>
<code>opener</code>	
<code>urllib.ContentTooShortError</code>	<code>urllib.error.ContentTooShortError</code>
<code>tTooShortError</code>	
<code>urllib2.urlopen()</code>	<code>urllib.request.urlopen()</code>
<code>urllib2.install_opener()</code>	<code>urllib.request.install_opener()</code>
<code>ner()</code>	
<code>urllib2.build_opener()</code>	<code>urllib.request.build_opener()</code>
<code>er()</code>	
<code>urllib2.URLError</code>	<code>urllib.error.URLError</code>
<code>urllib2.HTTPError</code>	<code>urllib.error.HTTPError</code>
<code>urllib2.Request</code>	<code>urllib.request.Request</code>
<code>urllib2.OpenerDirector</code>	<code>urllib.request.OpenerDirector</code>
<code>ector</code>	
<code>urllib2.BaseHandler</code>	<code>urllib.request.BaseHandler</code>
<code>urllib2.HTTPDefaultErrorHandler</code>	<code>urllib.request.HTTPDefaultErrorHandler</code>
<code>HTTPDefaultErrorHandler</code>	
<code>urllib2.HTTPRedirectHandler</code>	<code>urllib.request.HTTPRedirectHandler</code>
<code>PRedirectHandler</code>	
<code>urllib2.HTTPCookieProcessor</code>	<code>urllib.request.HTTPCookieProcessor</code>
<code>PCookieProcessor</code>	
<code>urllib2.ProxyHandler</code>	<code>urllib.request.ProxyHandler</code>
<code>r</code>	
<code>urllib2.HTTPPasswordMgr</code>	<code>urllib.request.HTTPPasswordMgr</code>
<code>asswordMgr</code>	
<code>urllib2.HTTPPasswordMgrWithDefaultRealm</code>	
<code>lib.request.HTTPPasswordMgrWithDefaultRealm</code>	<code>urllib.request.HTTPPasswordMgrWithDefaultRealm</code>

urllib2.AbstractBasicAuthHandler	urllib.request
AbstractBasicAuthHandler	
urllib2.HTTPBasicAuthHandler	urllib.request.H
TPBasicAuthHandler	
urllib2.ProxyBasicAuthHandler	urllib.request.P
oxyBasicAuthHandler	
urllib2.AbstractDigestAuthHandler	urllib.reque
t.AbstractDigestAuthHandler	
urllib2.HTTPDigestAuthHandler	urllib.request.
TTPDigestAuthHandler	
urllib2.ProxyDigestAuthHandler	urllib.request.
roxyDigestAuthHandler	
urllib2.HTTPHandler	urllib.request.HTTPHandl
r	
urllib2.HTTPSHandler	urllib.request.HTTPSHa
dlr	
urllib2.FileHandler	urllib.request.FileHandler
urllib2.FTPHandler	urllib.request.FTPHandler
urllib2.CacheFTPHandler	urllib.request.CacheF
PHandler	
urllib2.UnknownHandler	urllib.request.Unkno
nHandler	

urllib2发起GET请求

这里展示了一个官方使用urllib2的GET请求

```
import urllib2

res = urllib2.urlopen("http://httpbin.org/get")
print res.read().decode()
```

urllib3

urllib3是一个HTTP客户端Python库，Python2和Python3在未安装的情况下均可以使用pip install urllib3即可安装，它提供了很多urllib库中没有的重要特性：

- 线程安全
- 连接池
- 客户端SSL/TLS验证
- 文件分部编码上传
- 协助处理重复请求和HTTP重定位
- 支持gzip和deflate压缩编码
- 支持HTTP和SOCKS代理

- 100%测试覆盖率

urllib3发起GET请求

urllib3主要的特点就是在请求之前需要创建一个连接池对象

```
import urllib3

http = urllib3.PoolManager() # 线程池生成请求
res = http.request('GET', 'http://httpbin.org/get')
print(res.data.decode())
```

urllib3发起POST请求

post请求只需要将request()方法的第一个参数改为POST，并设置fields参数即可

```
import urllib3

http = urllib3.PoolManager() # 线程池生成请求
res = http.request('POST', 'http://httpbin.org/post', fields={'hello': 'world'})
print(res.data.decode())
```

urllib3设置headers

设置headers的方法和设置参数一样方便，只需要加入headers参数即可

```
headers = {'X-Something': 'value'}
res = http.request('POST', 'http://httpbin.org/post', headers=headers, fields={'hello': 'world'})
```

requests

最后介绍一个杀手级神器——requests，你会发现它一样也很好用，requests使用了urllib3的库，并继承了所有urllib2的特性，最关键的是它不需要urllib3一样在编码时收到开启线程池，简单粗暴，不它同样也要使用pip install requests安装。

格外简单的演示：

```
import requests

res = requests.get('http://httpbin.org/get')
print(res.text)
```

在实际使用中推荐使用后两种，因为毕竟方便，而若时没有安装这些库的情况下，Python3的urllib是一个不错的选择。

本文参考：<https://blog.csdn.net/jiduochou963/article/details/87564467>

urllib3官方文档：<https://urllib3.readthedocs.io/en/latest/>

requests官方文档：<https://2.python-requests.org/en/master/>