3.3 HTTP解析与Python实现

3.3.1 HTTP协议

3.3.2 Python-Requests解析HTTP方法

3.3.3 实例

3.3.3 实例

- 模拟登录：
  - 现在许多的网站都是需要登录验证后才能访问的，爬虫也是需要登录后，才能获取到后面的页面
  - https://www.cnblogs.com/qican/p/11277642.html
  - https://www.jb51.net/article/141305.htm
- Python模拟登录的多种方法(四种)
  - 方法一：直接使用已知的cookie访问
  - 方法二：模拟登录后再携带得到的cookie访问
  - 方法三：模拟登录后用session保持登录状态
  - 方法四：使用无头浏览器访问

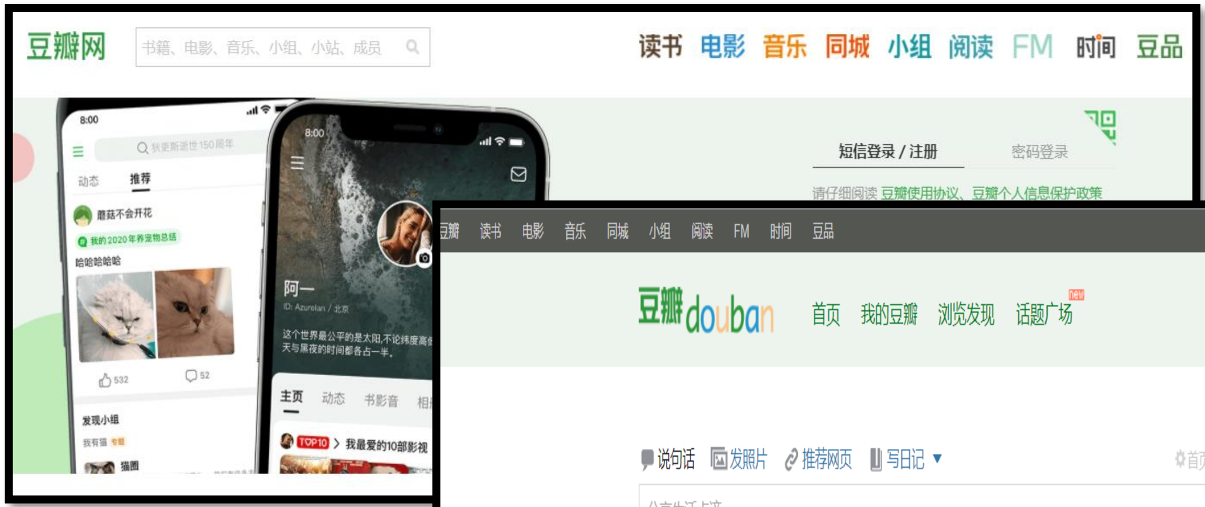
- 方法一.使用已知的cookie模拟登录
- 特点：
  - 简单，但需要先在浏览器登录
- 原理：
  - http是无状态的连接:服务器无法判断出请求是哪个客户端发起
  - 而“访问登录后才能看到的页面”这一行为，恰恰需要客户端向服务器证明：“我是刚才登录过的那个客户端”
  - 需要cookie来标识客户端的身份
- 使用已知的cookie模拟登录具体步骤：
  - 用浏览器登录，然后使用开发者工具查看cookie。
  - 在程序中携带该cookie向网站发送请求，让程序模拟成刚才登录的那个浏览器，得到只有登录后才能看到的页面

3.3.3 实例

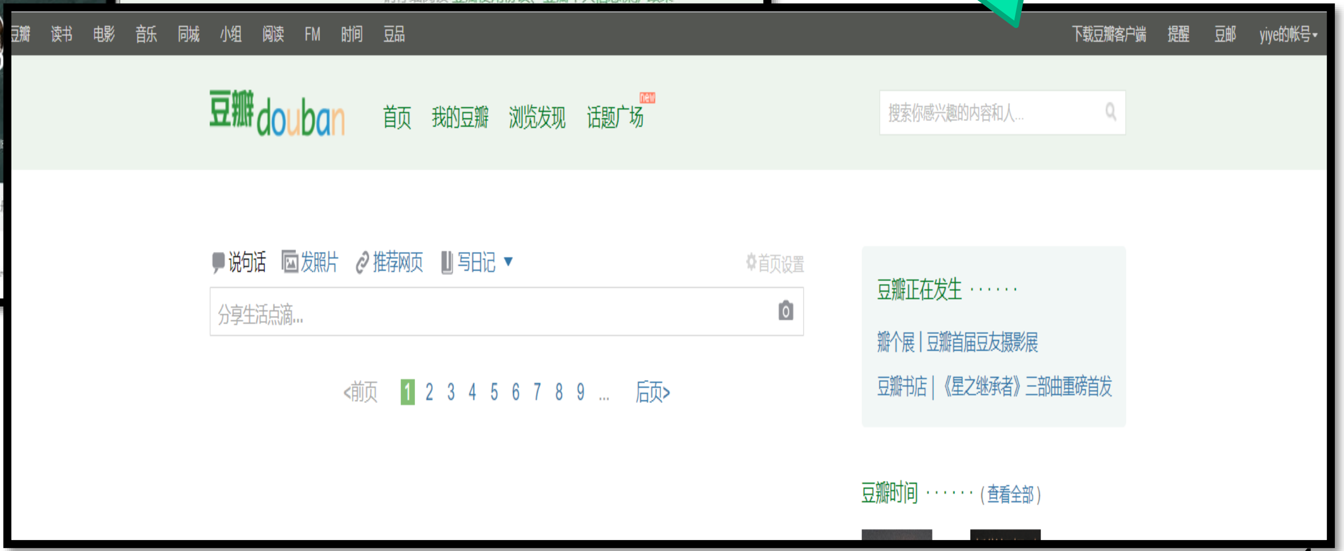
例1.使用已知的cookie进行模拟登录(豆瓣网https://www.douban.com/)

(2021年运行成功！）

1.采用自己的用户名、密码登录



豆瓣网
书籍、电影、音乐、小组、小站、成员
读书 电影 音乐 同城 小组 阅读 FM 时间 豆品
短信登录 / 注册
密码登录
请仔细阅读 豆瓣使用协议、豆瓣个人信息保护政策



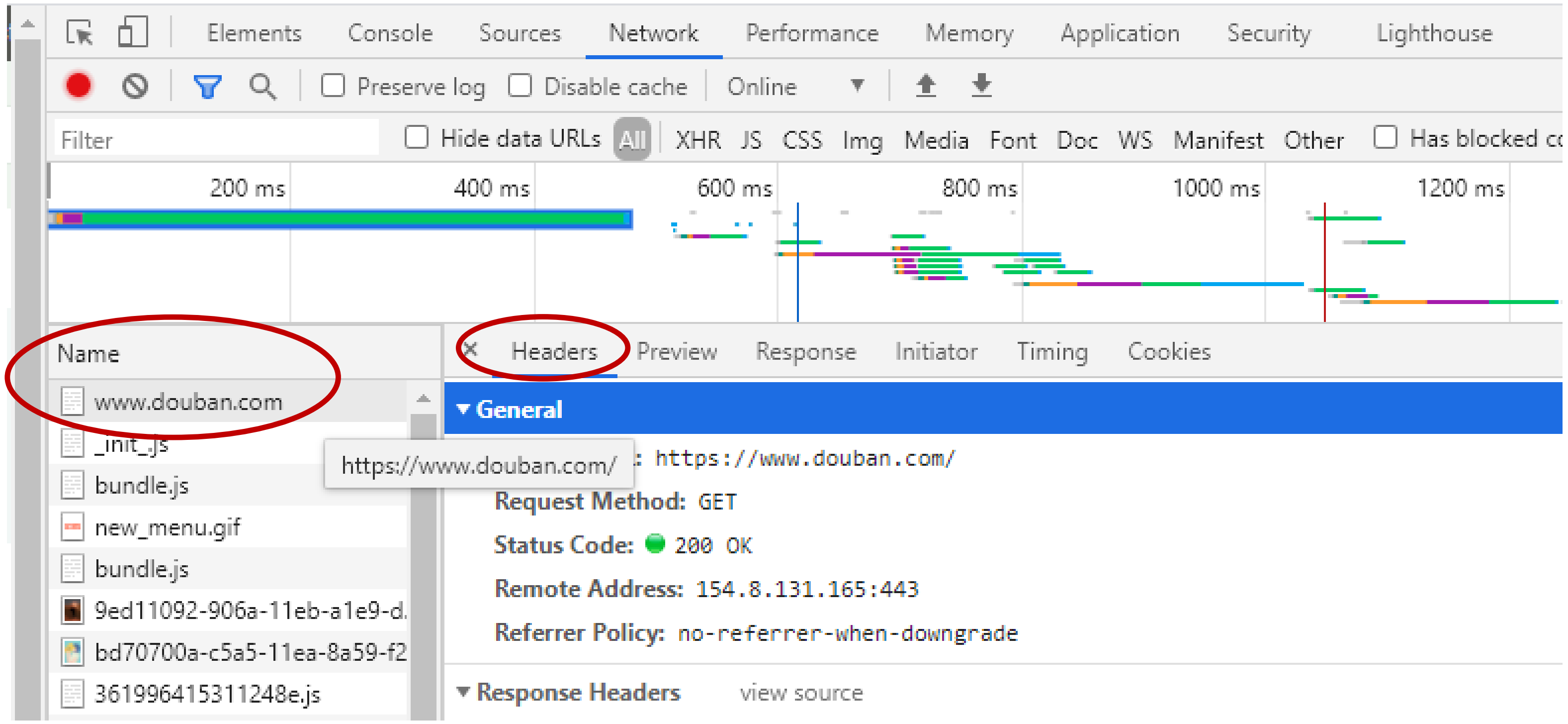
豆瓣 读书 电影 音乐 同城 小组 阅读 FM 时间 豆品
下载豆瓣客户端 提醒 豆邮 yiye的帐号
豆瓣douban
首页 我的豆瓣 浏览发现 话题广场
搜索你感兴趣的内容和人...
说句话 发照片 推荐网页 写日记
首页设置
分享生活点滴...
<前页 1 2 3 4 5 6 7 8 9 ... 后页>
豆瓣正在发生 ······
瓣个展 | 豆瓣首届豆友摄影展
豆瓣书店 | 《星之继承者》三部曲重磅首发
豆瓣时间 ······ (查看全部)

人工登录成功

3.3.3 实例

2、进入“开发者工具”，点击network+(ctrl+R)

找到Name中的一个地址(www.douban.com)，看Headers



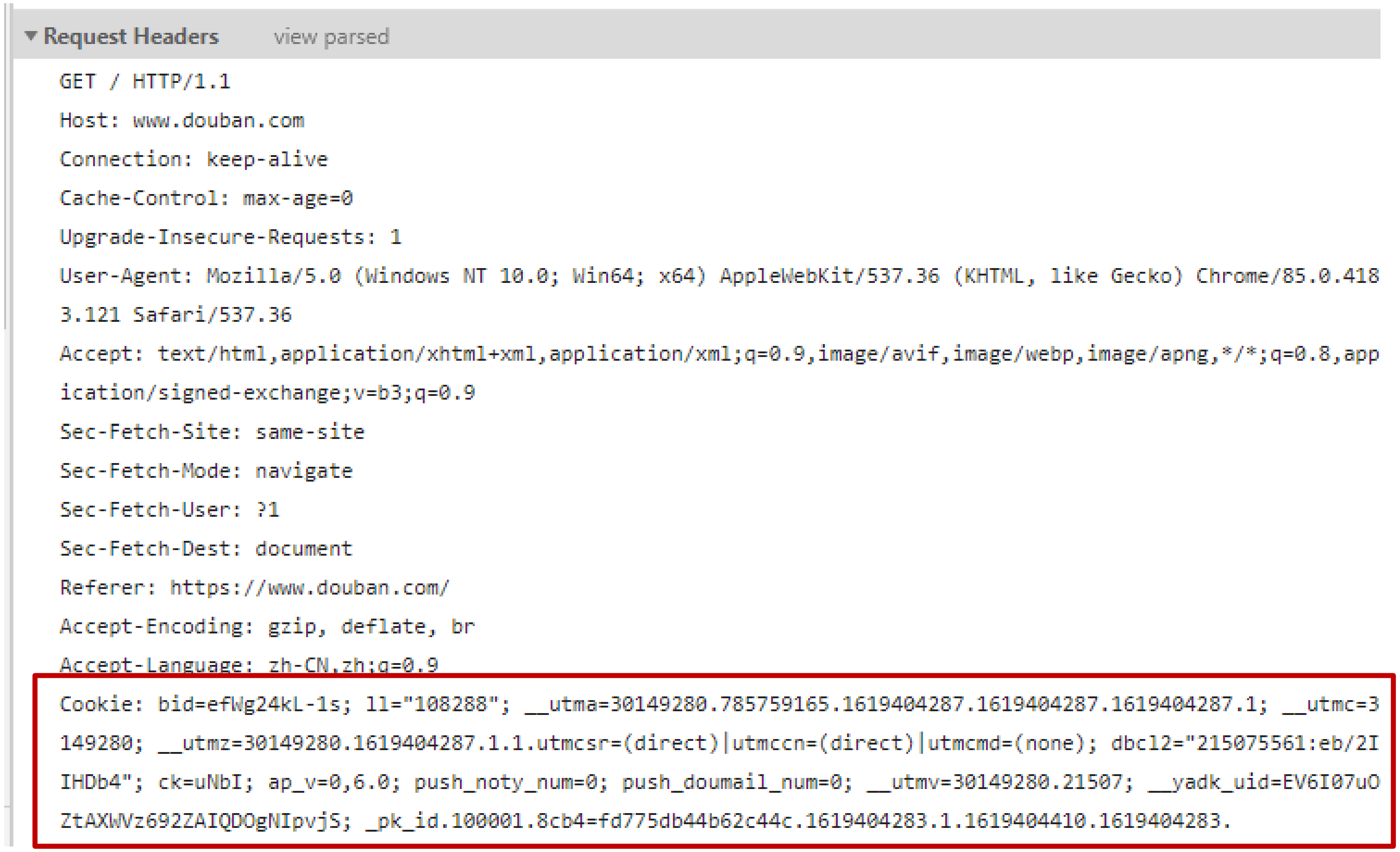
www.douban.com
init.js
bundle.js
new_menu.gif
bundle.js
9ed11092-906a-11eb-a1e9-d
bd70700a-c5a5-11ea-8a59-f2
361996415311248e.js

https://www.douban.com/

: https://www.douban.com/
Request Method: GET
Status Code: ● 200 OK
Remote Address: 154.8.131.165:443
Referrer Policy: no-referrer-when-downgrade

3.3.3 实例

3.找到Headers中的Request Header→view source→cookie

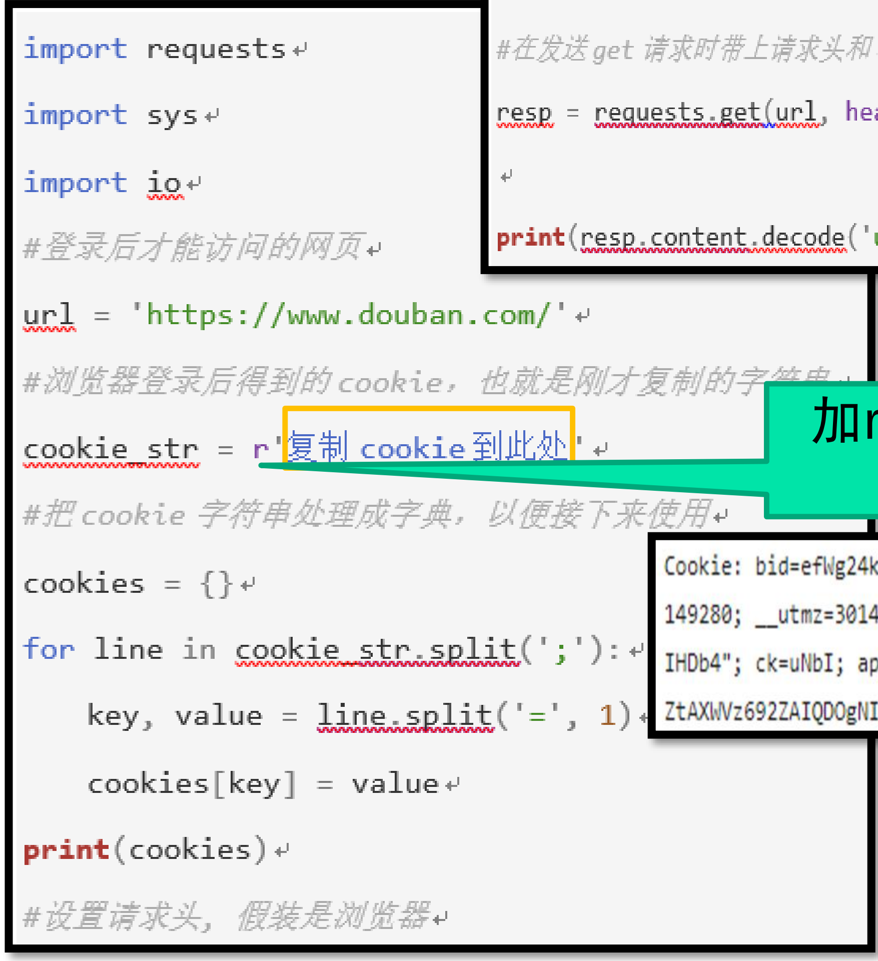


▾ Request Headers   view parsed

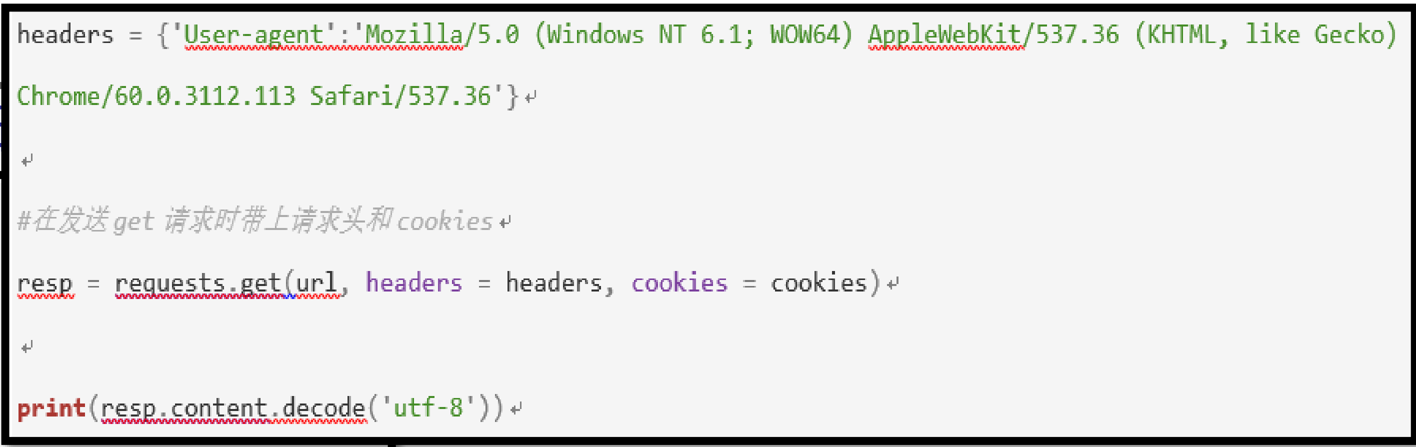
```
GET / HTTP/1.1
Host: www.douban.com
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.418
3.121 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app
ication/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://www.douban.com/
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Cookie: bid=efWg24kL-1s; ll="108288"; __utma=30149280.785759165.1619404287.1619404287.1619404287.1; __utmc=3
149280; __utmz=30149280.1619404287.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none); dbcl2="215075561:eb/2I
IHDb4"; ck=uNbI; ap_v=0,6.0; push_noty_num=0; push_doumail_num=0; __utmv=30149280.21507; __yadk_uid=EV6I07uO
ZtAXWVz692ZAIQDOgNIpvjS; _pk_id.100001.8cb4=fd775db44b62c44c.1619404283.1.1619404410.1619404283.
```

3.3.3 实例

4.书写代

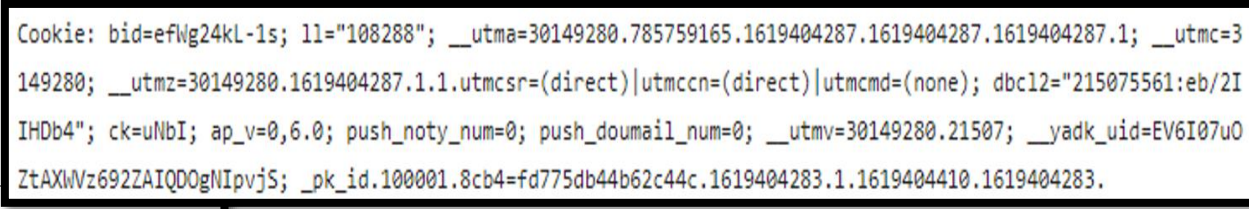


```
import requests
import sys
import io
#登录后才能访问的网页
url = 'https://www.douban.com/'
#浏览器登录后得到的cookie，也就是刚才复制的字符串
cookie_str = r'复制 cookie 到此处'
#把 cookie 字符串处理成字典，以便接下来使用
cookies = {}
for line in cookie_str.split(';'):
    key, value = line.split('=', 1)
    cookies[key] = value
print(cookies)
#设置请求头，假装是浏览器
```



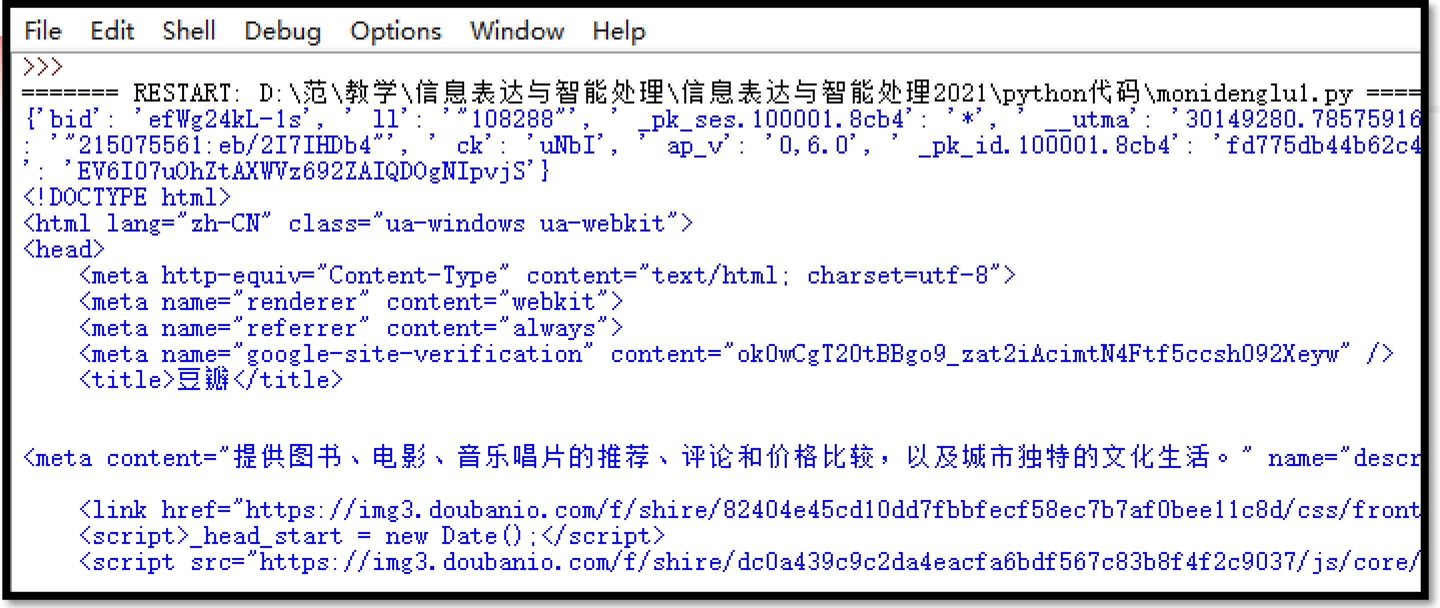
```
headers = {'User-agent':'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/60.0.3112.113 Safari/537.36'}

#在发送get 请求时带上请求头和cookies
resp = requests.get(url, headers = headers, cookies = cookies)

print(resp.content.decode('utf-8'))
```

加r，其后的引号中的字符串可以保留源异类符号

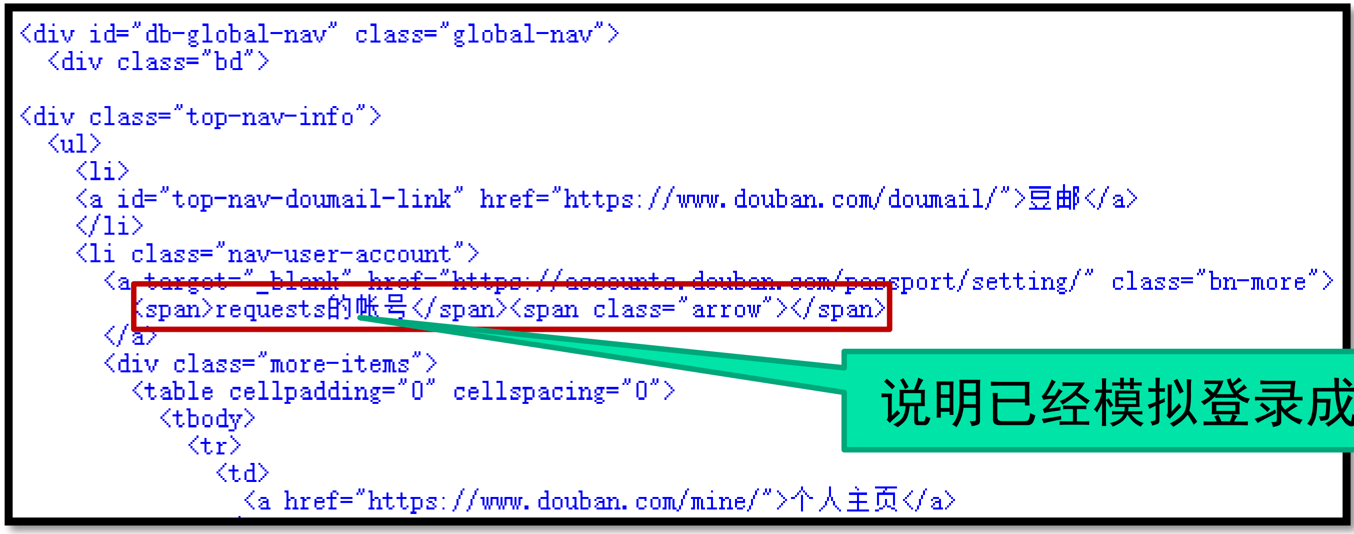


```
Cookie: bid=efWg24kL-1s; ll="108288"; __utma=30149280.785759165.1619404287.1619404287.1619404287.1; __utmc=3
149280; __utmz=30149280.1619404287.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none); dbcl2="215075561:eb/2I
IHDb4"; ck=uNbI; ap_v=0,6.0; push_noty_num=0; push_doumail_num=0; __utmv=30149280.21507; __yadk_uid=EV6I07uO
ZtAXWVz692ZAIQDOgNIpvjS; _pk_id.100001.8cb4=fd775db44b62c44c.1619404283.1.1619404410.1619404283.
```

运行结果



```
>>>
======= RESTART: D:\范\教学\信息表达与智能处理\信息表达与智能处理2021\python代码\monidenglu1.py ====
{'bid': 'efWg24kL-1s', 'll': '"108288"', '_pk_ses.100001.8cb4': '*', '__utma': '30149280.78575916
: '"215075561:eb/2I7IHDb4"', 'ck': 'uNbI', 'ap_v': '0,6.0', '_pk_id.100001.8cb4': 'fd775db44b62c4
': 'EV6I07uOhZtAXWVz692ZAIQDOgNIpvjS'}
<!DOCTYPE html>
<html lang="zh-CN" class="ua-windows ua-webkit">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <meta name="renderer" content="webkit">
    <meta name="referrer" content="always">
    <meta name="google-site-verification" content="ok0wCgT20tBBgo9_zat2iAcimtN4Ftf5ccsh092Xeyw" />
    <title>豆瓣</title>


<meta content="提供图书、电影、音乐唱片的推荐、评论和价格比较，以及城市独特的文化生活。" name="descr

    <link href="https://img3.doubanio.com/f/shire/82404e45cd10dd7fbbfecf58ec7b7af0bee11c8d/css/front
    <script>_head_start = new Date();</script>
    <script src="https://img3.doubanio.com/f/shire/dc0a439c9c2da4eacfa6bdf567c83b8f4f2c9037/js/core/
```



```
<div id="db-global-nav" class="global-nav">
  <div class="bd">

<div class="top-nav-info">
  <ul>
    <li>
    <a id="top-nav-doumail-link" href="https://www.douban.com/doumail/">豆邮</a>
    </li>
    <li class="nav-user-account">
      <a target="_blank" href="https://accounts.douban.com/passport/setting/" class="bn-more">
        <span>requests的帐号</span><span class="arrow"></span>
      </a>
      <div class="more-items">
        <table cellpadding="0" cellspacing="0">
          <tbody>
            <tr>
              <td>
                <a href="https://www.douban.com/mine/">个人主页</a>
```

说明已经模拟登录成功

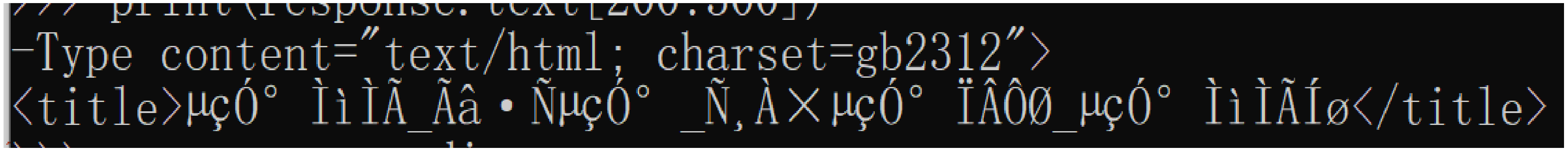
3.3.3 实例

- 例2.Python requests库中文乱码问题
  - Python requests库中文乱码问题 https://www.cnblogs.com/bw13/p/6549248.html
  - Python HTTP库requests中文页面乱码解决方案！ https://www.cnblogs.com/bitpeng/p/4748872.html
- Python中文乱码，是一个很大的坑
- Python requests库比Python的官方API接口好用多了
- 美中不足的是：这个库好像对中文的支持不是很友好，有些页面会出现乱码

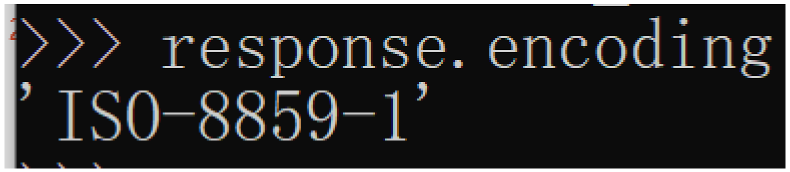
3.3.3 实例

- 使用电影天堂的网页，因为网页不太标准

```python
import requests


response = requests.get('http://www.dytt8.net/index.htm')
print(response.text[200:300])
```



```
>>> print(response.text[200:300])
-Type content="text/html; charset=gb2312">
<title>µçÓ° ÌìÌÃ_Ãâ•ÑµçÓ° _Ñ¸À×µçÓ° ÏÂÔØ_µçÓ° ÌìÌÃÍø</title>
```

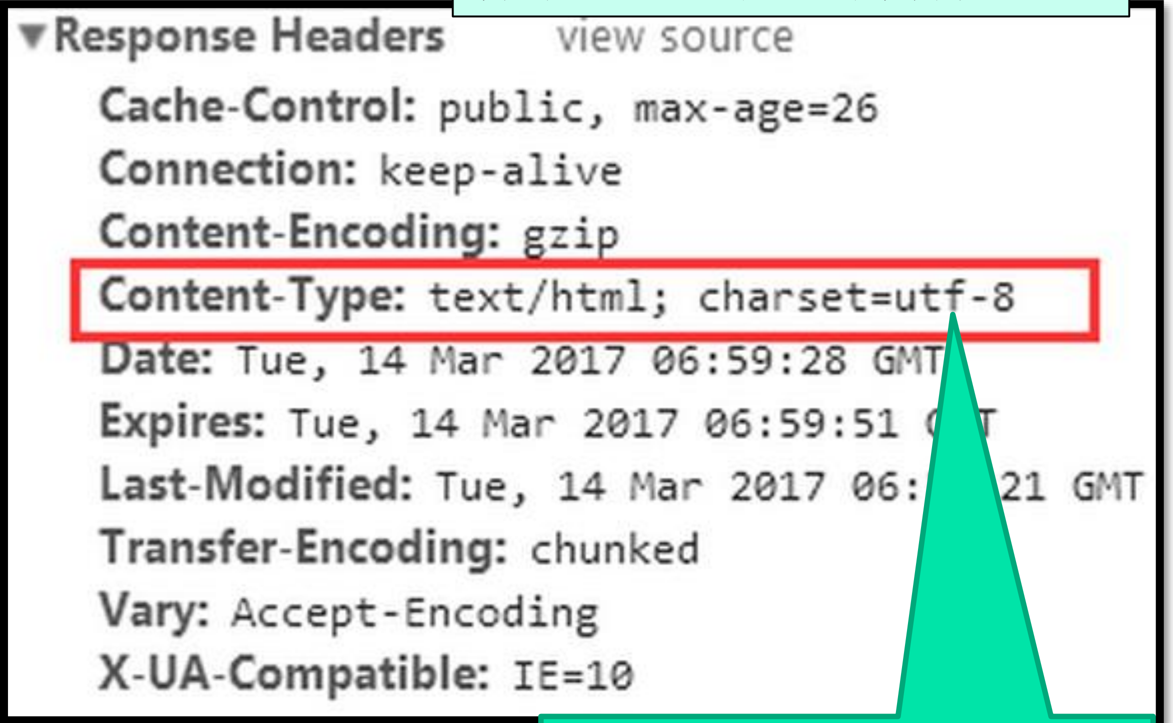


```
>>> response.encoding
'ISO-8859-1'
```

中文乱码原因：网站程序定义编码各异

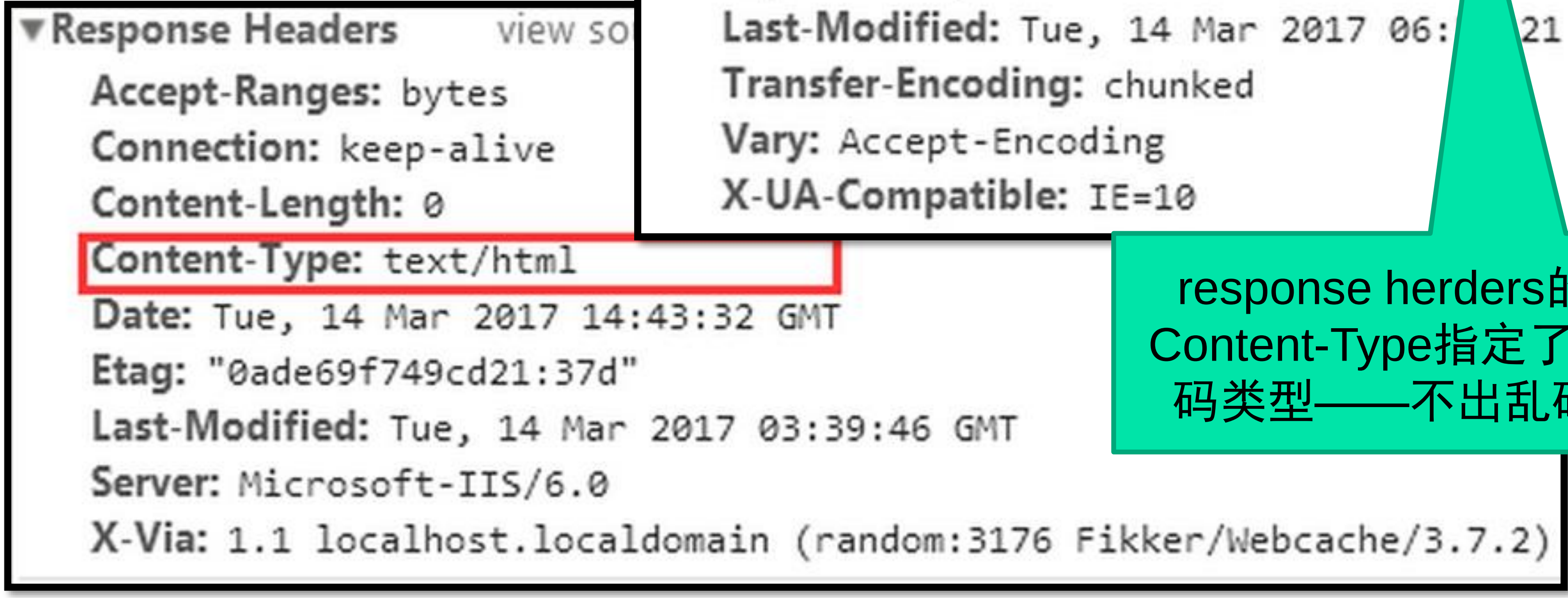
- 国内的站点一般是utf-8、gbk、gb2312

标准点的网页：博客园



```
▼Response Headers          view source
    Cache-Control: public, max-age=26
    Connection: keep-alive
    Content-Encoding: gzip
    Content-Type: text/html; charset=utf-8
    Date: Tue, 14 Mar 2017 06:59:28 GMT
    Expires: Tue, 14 Mar 2017 06:59:51 GMT
    Last-Modified: Tue, 14 Mar 2017 06:  21 GMT
    Transfer-Encoding: chunked
    Vary: Accept-Encoding
    X-UA-Compatible: IE=10
```

response header只指定了type，但是没有指定编码——可能出乱码



```
▼Response Headers          view so
    Accept-Ranges: bytes
    Connection: keep-alive
    Content-Length: 0
    Content-Type: text/html
    Date: Tue, 14 Mar 2017 14:43:32 GMT
    Etag: "0ade69f749cd21:37d"
    Last-Modified: Tue, 14 Mar 2017 03:39:46 GMT
    Server: Microsoft-IIS/6.0
    X-Via: 1.1 localhost.localdomain (random:3176 Fikker/Webcache/3.7.2)
```

response herders的Content-Type指定了编码类型——不出乱码

3.3.3 实例

- 中文乱码解决：
- 1.获取真实字符集：
  - 使用apparent_encoding
  - 从html的meta中抽取

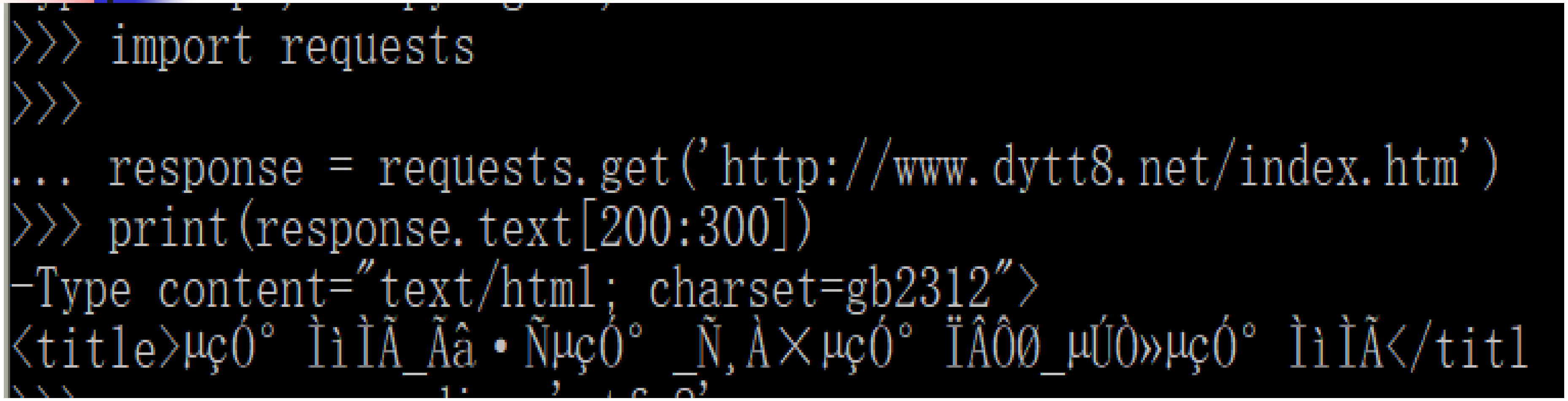
```
>>> response.apparent_encoding
'GB2312'
```

```
>>> requests.utils.get_encodings_from_content(response.text)
['gb2312']
```

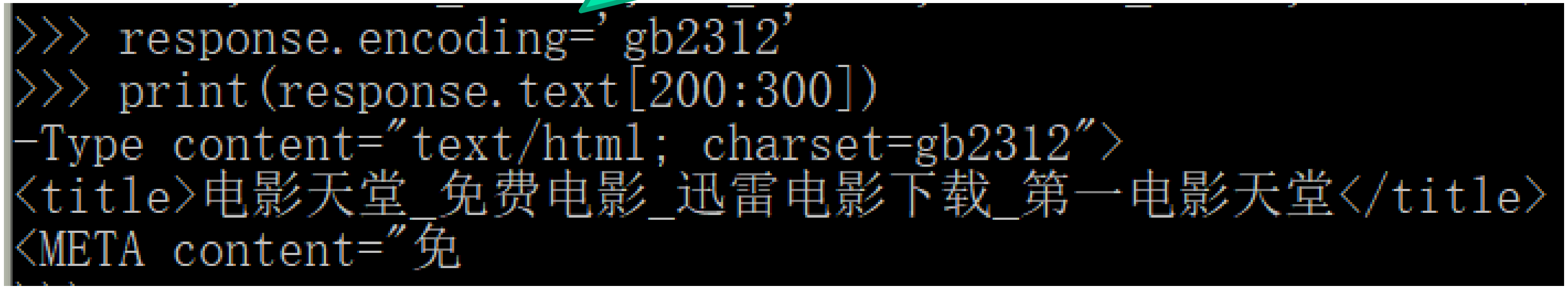
- 2.设置字符集：

```
# response.encoding = response.apparent_encoding
response.encoding = 'gb2312'
```

3.3.3 实例



```
>>> import requests
>>>
... response = requests.get('http://www.dytt8.net/index.htm')
>>> print(response.text[200:300])
-Type content="text/html; charset=gb2312">
<title>µçÓ° ÌìÌÃ_Ãâ·ÑµçÓ° _Ñ¸À×µçÓ° ÏÂÔØ_µÚÒ»µçÓ° ÌìÌÃ</titl
```



```
>>> response.encoding='gb2312'
>>> print(response.text[200:300])
-Type content="text/html; charset=gb2312">
<title>电影天堂_免费电影_迅雷电影下载_第一电影天堂</title>
<META content="免
```