准备知识

1. 动态网页的** 爬取思路 **与静态网页** 基本一致 **

步骤	任务	说明	方法
1	**生成网址**	分析** 网址规律 **，批量生成网址	for循环，format函数
2	请求+获取网页数据	模拟人工打开网页，并将网页存储为数据对象	requests包
3	**解析数据**	分析数据规律，整理出所需字段	pyquery包，** json包 ** (还有lxml和beautifulsoup4)
4	存储数据	使用csv包将数据存储到csv文件中	csv包
5	批量爬取	对所有网址循环步骤2-4	for循环

2. 动态与静态的区别

- （1）网址规律:
  - 翻页时，动态网页的** 网址不变化 **
  - 无法在网页源代码直接看到内容

- （2）解析数据:
  - 不是网页源代码的字符串html，而是** 字典列表json **

生成网址

分析网址规律

1. ** 任务对象 **:

   - https://item.jd.com/100002781562.html#comment (https://item.jd.com/100002781562.html#comment)

2. ** 基本情况 **:

   - 不论怎么翻页，** 网址都没有变化 **
   - 每页10个评论，最多翻到100页

3. ** 原因分析及应对 **:

   - 真正的数据存在别的网址中，当我们点击对应的页数时，它才被“调用”，然后显示在我们的浏览器界面中
   - ** 用开发者工具中的network，观察翻页时文件的变化，找到存储“评论”的文件，再查看headers中的request url，这就是真正的数据网址。 **（确认评论数据存储在request url中）

4. ** 观察规律 **:

- ** 原网址 **:
  - https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678 (https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678
  - https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678 (https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678
  - https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678 (https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678
  - ……
  - https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678 (https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678
  - https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678 (https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=169138519678

- ** 总结规律 **:
  - 第1页，page为0
  - 第2页，page为1
  - 第3页，page为2
  - ……
  - 第99页，page为98
  - 第100页，page为99

  - 第p页，page为(p-1)

批量生成网址

In [ ]:

```
# 每个商品有自己的唯一id，例如100002781562、1540112、10080196292423等
product_id = '100002781562'

# format函数设置product_id和page
template = 'https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&

# 循环生成网址，并存入网址列表中
url_list = []
for p in range(1,101):
    url = template.format(product_id = product_id, page=(p-1))    # 分别填入product_id和
    url_list.append(url)
```

In [ ]:

```
# 查看url_list情况（获得10页的网址）
url_list
```

In [ ]:

```
# 函数：生成网址  generate_url_list(product_id, max_page)
# 参数说明：product_id为商品id，max_page为最大页数
# 返回值：url_list为网址列表
def generate_url_list(product_id, max_page):
    url_list = []
    template = 'https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComme
    for p in range(1,max_page+1):        # range取1到100，即p循环1到100页
        url = template.format(product_id = product_id, page=(p-1))
        url_list.append(url)
    return url_list
```

In [ ]:

```
# 调用函数  generate_url_list(product_id, max_page)
url_list_tmp1 = generate_url_list(product_id=100002781562, max_page=5)
url_list_tmp2 = generate_url_list(product_id=1540112, max_page=10)

print('商品id为100002781562的5条数据网址：',url_list_tmp1,'\n')
print('商品id为1540112的10条数据网址：',url_list_tmp2)
```

请求+获取网页数据

- 以第1页为例，来请求并获取网页数据
- 例子：https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=1691385196787&lc

(https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=1691385196787&lc

In [ ]:

```
import requests   # 导入requests包

url = 'https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&clien
resp = requests.get(url)   # 用get向服务器请求获取数据
resp # 查看状态码，2开头代表访问成功，4开头代表不成功
```

In [ ]:

```
resp.text   # 查看返回内容
```

In [ ]:

```
raw_comments = resp.text  # 将内容放入raw_comments中，类型为str
type(raw_comments)
```

In [ ]:

```
# 函数：获得json  get_json(url)
# 参数说明：url为单个网址
# 返回值：raw_comments为原始的评论数据
def get_json(url):
    resp = requests.get(url)
    raw_comments = resp.text
    return raw_comments
```

In [ ]:

```
# 调用函数  get_json(url) 以第8页为例
url_tmp = 'https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&c
raw_comments_tmp = get_json(url_tmp)
raw_comments_tmp
```

解析数据

观察raw_comments数据结构

- https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=1691385196787&lc

(https://api.m.jd.com/?appid=item-v3&functionId=pc_club_productPageComments&client=pc&clientVersion=1.0.0&t=1691385196787&lc

1. comments里有多个字典，每一个字典对应一个用户评论
2. key：content内容、creationTime时间、plusAvailable会员身份、nickname用户名、score评分、

In [ ]:

```
import json

json.loads(raw_comments)    # 如果raw_comments是字典列表，那么能正常loads；如果【不是规范
```

In [ ]:

```
comments = json.loads(raw_comments)['comments']    # 取comments键的值，并存入comments对象
comments
```

In [ ]:

```
# comments为字典列表，每一个字典是一条用户评论
comments[0].keys() # 查看字典的key
```

- 我们只取其中的content内容、creationTime时间、plusAvailable会员身份、nickname用户名、score评分、usefulVoteCount点赞数、replyCount回复数、productColor颜色、productSize尺码等

In [ ]:

```
data_list = []                                  # data_list为空列表，用于存储评论信息

for comment in comments:
    data = {}                                   # data为空字典

    data['content']=comment.get('content')    # 取content键的值，放入data中的content键值
    data['creationTime']=comment.get('creationTime')
    data['nickname']=comment.get('nickname')
    data['plusAvailable']=comment.get('plusAvailable')
    data['score']=comment.get('score')
    data['usefulVoteCount']=comment.get('usefulVoteCount')
    data['replyCount']=comment.get('replyCount')
    data['productColor']=comment.get('productColor')
    data['productSize']=comment.get('productSize')

    data_list.append(data)

print(data_list)
print(len(data_list))
```

In [ ]:

```
# 函数：解析数据  extract_comments(raw_comments)
# 参数说明：raw_comments为初始的评论列表
# 返回值：data_list为整理后的评论列表
def extract_comments(raw_comments):
    data_list = []

    comments = json.loads(raw_comments)['comments']

    for comment in comments:    # 将一条条评论写入data_list中
        data = {}
        data['content']=comment.get('content')
        data['creationTime']=comment.get('creationTime')
        data['nickname']=comment.get('nickname')
        data['plusAvailable']=comment.get('plusAvailable')
        data['score']=comment.get('score')
        data['usefulVoteCount']=comment.get('usefulVoteCount')
        data['replyCount']=comment.get('replyCount')
        data['productColor']=comment.get('productColor')
        data['productSize']=comment.get('productSize')
        data_list.append(data)

    return data_list
```

In [ ]:

```
# 调用函数   extract_comments(raw_comments)
comments = extract_comments(raw_comments)
print(data_list)
print('评论数量为：', len(data_list))
```

存储数据

In [ ]:

```
import csv    # 导入csv包

# 打开文件
file = open('output/umbrella_100002781562.csv', 'a+', encoding='utf-8', newline='') # 文
fieldnames = ['content', 'creationTime', 'nickname', 'plusAvailable', 'score', 'usefulV
writer = csv.DictWriter(file, fieldnames=fieldnames)    # 要求以字典的形式写入数据，fie
writer.writeheader()  # 将fieldnames设置的标题key写入首行

# 循环写入字典列表：因为有多条评论，需要一行行写入
for data in data_list:
    writer.writerow(data)  # 写入一行评论数据

file.close() # 关闭文件
```

批量爬取

下面对所有网址url_list循环步骤2-4

步骤	任务	函数	输入参数	返回值
1	生成网址	generate_url_list(product_id, max_page)	product_id为商品id，max_page为最大页数	url_list网址列表
2	**请求+获取网页数据**	get_json(url)	单个网址	raw_comments原始的评论数据
3	**解析数据**	extract_comments(raw_comments)	（单个网址）raw_comments原始的评论数据	data_list评论的字典列表
4	**存储数据**	--	--	--
5	批量爬取	--	--	--

与静态网页思路一致，直接写主函数

写主函数前，还有个** time知识点 **

- 如果我们访问速度过快，也可能会被服务器反爬，如果我们** 间隔一段时间 **再访问，则能降低被反爬的概率

In [ ]:

```
import time
import random

#random()随机返回[0,1)范围内的实数
print(random.random()*3)
print(random.random()*3)

# 休息对应的时间
time.sleep(random.random()*3)
```

In [ ]:

```python
# 函数：爬虫主函数  main(product_id, max_page, filename)
# 参数说明：product_id为商品id，max_page为最大页数，filename为文件名称
# 仅执行命令，不返回任何值
def main(product_id, max_page, filename):
    print('开始采集商品id为{product_id}的商品评论！'.format(product_id = product_id))

    # 生成所有网址url_list
    url_list = generate_url_list(product_id, max_page)

    # 打开文件
    file = open(filename, 'a+', encoding='utf-8', newline='')
    fieldnames = ['content', 'creationTime', 'nickname', 'plusAvailable', 'score','use
    writer = csv.DictWriter(file, fieldnames=fieldnames)
    writer.writeheader()

    # 对所有网址url_list循环步骤2-4
    for url in url_list:
        print('正在采集：{url}'.format(url=url))
        raw_comments = get_json(url)                          # 【步骤2：请求+获取网页数据
        time.sleep(random.random()*3)                      # 间隔不定长时间
        data_list = extract_comments(raw_comments) # 【步骤3：解析数据】
        for data in data_list:                  # 【步骤4：存储数据】
            writer.writerow(data)

    file.close()

    print('采集完毕！')
```

In [ ]:

```python
main(product_id = 100002781562, max_page=5, filename='output/umbrella_5.csv')
```

In [ ]:

```python
main(product_id = 1540112, max_page=10, filename='output/florida_10.csv')
```

- 我们读取下刚才生成的文件

In [ ]:

```python
import pandas as pd
```

In [ ]:

```
pd_reader = pd.read_csv('output/florida_10.csv')
pd_reader
```

In [ ]:

```
pd_reader.shape    # 查看行数，列数
```

In [ ]:

```
pd_reader.head(10)    # 查看前几行信息，默认为5
```

In [ ]:

```
pd_reader.tail(8)    # 查看后几行信息，默认为5
```

END