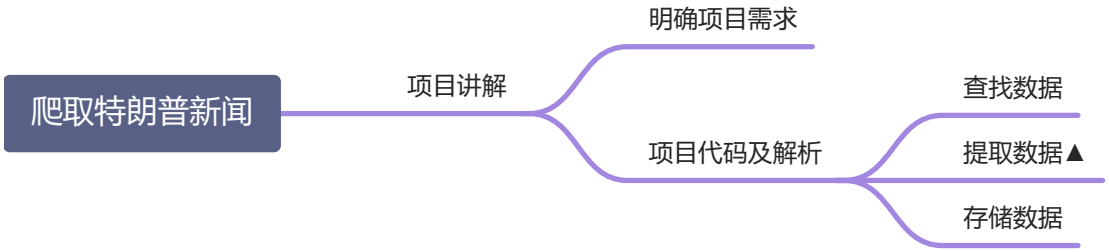
第7课 项目实操-爬取特朗普新闻

一、课程结构导图



注：▲为重点知识点。

二、项目讲解

2.1 明确项目需求

1. 爬取今日头条网站，前60条特朗普新闻中的热点新闻（标题，文章链接）。
2. 将爬取的热点新闻数据写入到csv文档中。

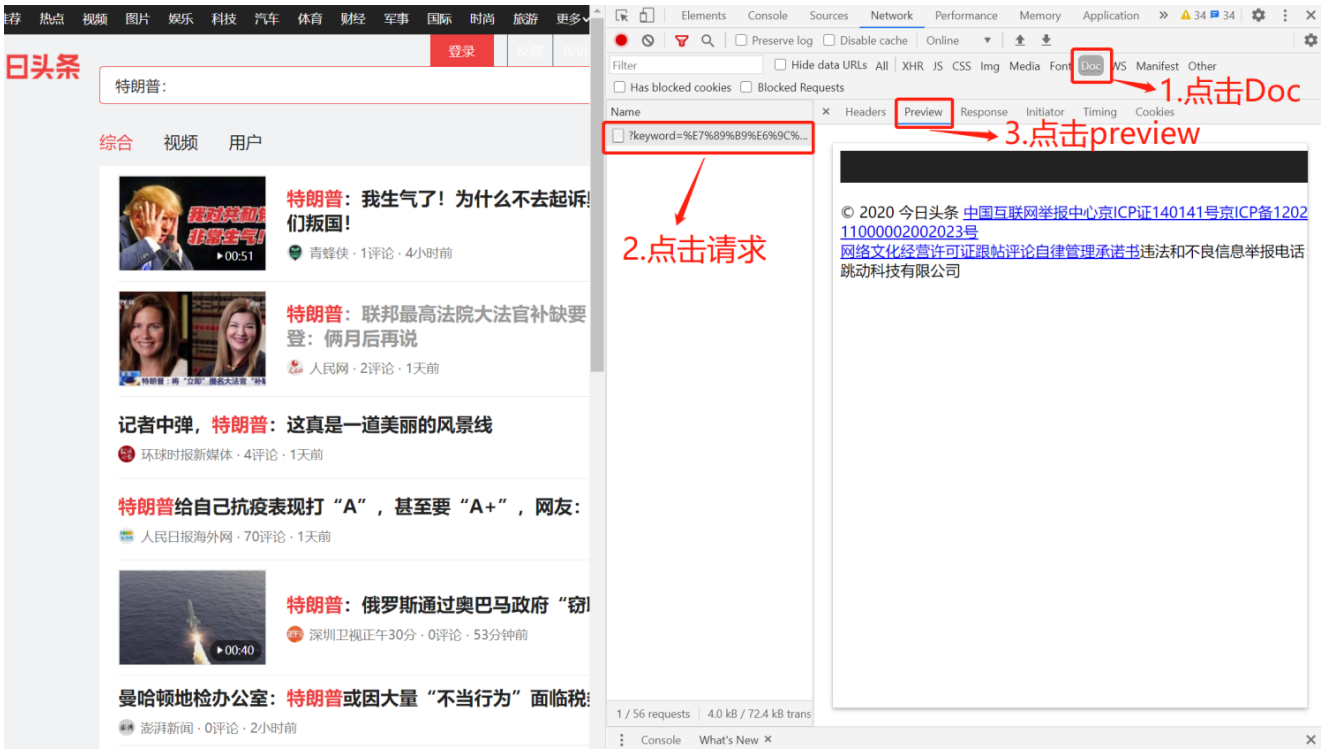
2.2 项目代码及解析

2.2.1 查找数据

目的1： 判断出新闻标题和文章链接是否在html中。

方法1： 查看第0个请求是否与网页显示内容一致（打开检查中的Network-点击Doc-点击请求-点击preview）。

图示：

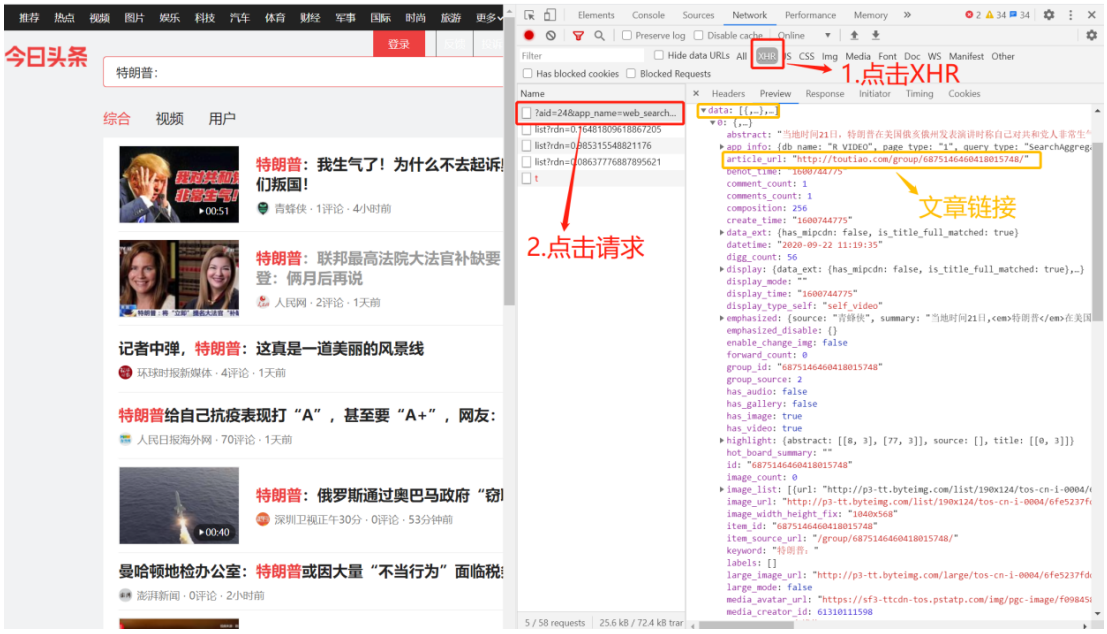


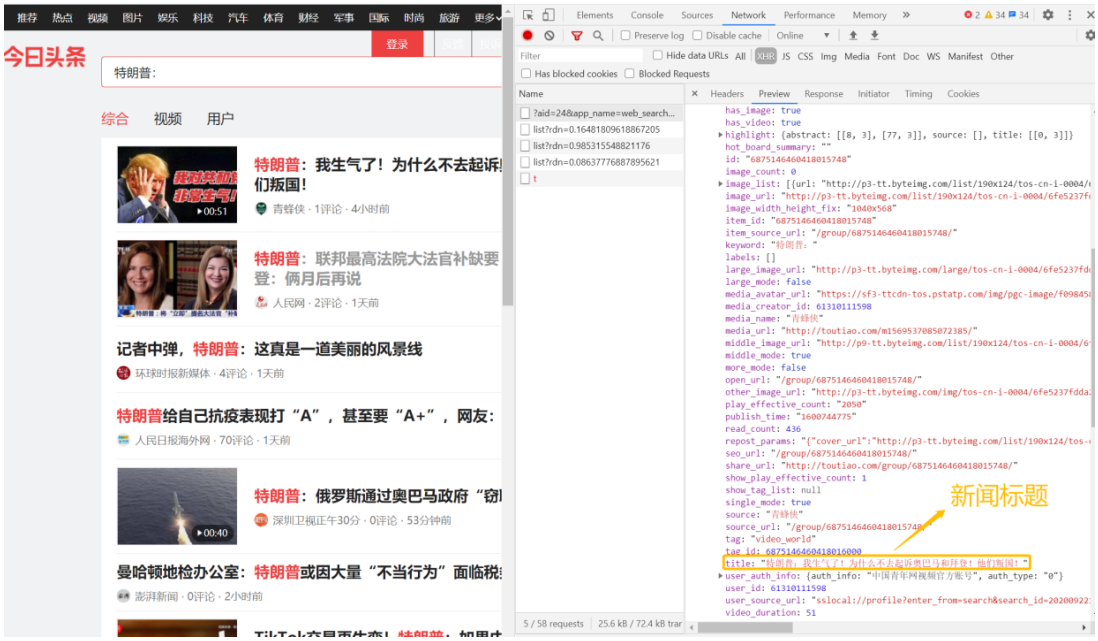
操作解析：

1. 通过观察，第0个请求中并没有包含新闻标题，文章url数据。也证明了数据不在html中，而是在xhr中。
2. 第0个请求一般会放在Network的Doc中，直接查找即可。

目的2：查找出xhr中所需爬取的新闻标题和文章链接。

图示：





代码解析：

1. 当我们打开xhr时，会发现有几个请求，我们只需逐一的查看每个请求中的preview，就能判断出第一个请求是包含了新闻标题和文章链接的相关数据。
2. 根据英文翻译，比较容易的得出，artice_url是文章链接，title是标题。所以我们已经找到了数据的位置，接下来我们根据json数据的层级关系对文章链接和新闻标题进行提取。

2.2.2 提取数据

目的1：根据数据层级关系，提取新闻标题与文章链接。

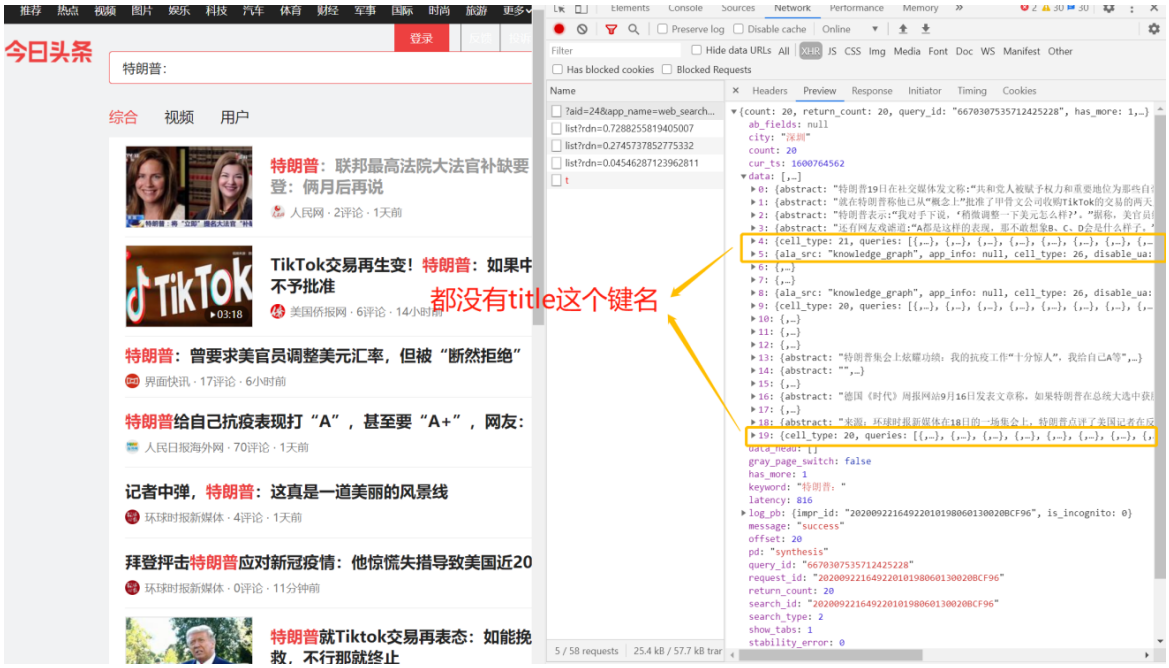
关键代码1：

```
import requests

url = "https://www.toutiao.com/api/search/content/"

headers={'user-agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like

# 封装params变量
params = {'aid': '24',
          'app_name': 'web_search',
          'offset': '0',
          'format': 'json',
          'keyword': '特朗普: ',
          'autoload': 'true',
          'count': '20',
          'en_qc': '1',
          'cur_tab': '1',
```

```
            'from': 'search_tab',
            'pd': 'synthesis',
            'timestamp': '1597805260196',
            '_signature': 'dtEYPAAgEBCwhqGxqNBu4nbQWSAACnwFpTASKTMh-7VzDcR4ykquvMXl2F.pPxwChz4GKYLO1cUznX
           }

res = requests.get(url,params=params,headers=headers)

# 定位数据
articles=res.json()
data=articles['data']

# 遍历data列表，提取出里面的新闻标题与链接
for i in data:
    list1=[i['title'],i["article_url"]]
    print(list1)

# 运行结果：
# KeyError: 'title'
```

代码解析：

1. 第8行代码到第21行代码，对查询参数进行一个封装，其中offset表示的是从那一篇新闻开始爬取，count表示的是爬取的新闻数量，keyword表示的是爬取的关键字。**所以代码params参数中'offset': '0'，'count': '20'，表示的是从第1篇新闻开始爬取，共爬取20篇。**
2. 该代码的报错提示是没有title这个键名的值，之所以出现该报错，原因是在data大字典下，并不是每一个元素都有title这个键名（如下图所示）。



3. 通过上图可以观察到，20篇新闻里，有3篇是没有title这个键名的，这些周边新闻并非是我们想要的，接下来，我们需要通过try....except语句来进行判断哪些是周边新闻，哪些是热点新闻。

目的2：添加try....except语句对爬取数据进行判断并筛选出热点新闻。

关键代码2：

```
import requests

url = "https://www.toutiao.com/api/search/content/"

headers={'user-agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like

# 封装params变量
params = {'aid': '24',
          'app_name': 'web_search',
          'offset': '20',
          'format': 'json',
          'keyword': '特朗普: ',
          'autoload': 'true',
          'count': '20',
          'en_qc': '1',
          'cur_tab': '1',
          'from': 'search_tab',
          'pd': 'synthesis',
          'timestamp': '1597805260196',
          '_signature': 'dtEYPAAgEBCwhqGxqNBu4nbQWSAACnwFpTASKTMh-7VzDcR4ykquvMXl2F.pPxwChz4GKYLO1cUznX
         }

res = requests.get(url,params=params,headers=headers)

# 定位数据
articles=res.json()
data=articles['data']

# 遍历data列表，提取出里面的新闻标题与链接
for i in data:
    try:
        list1=[i['title'],i["article_url"]]
        print(list1)
    except:
        print("此处无银三百两")
```

代码解析：

1. 代码中使用了try.....except来捕抓异常，当捕抓到KeyError时，会执行except语句，代码中的except语句也可以写为except KeyError。
2. 代码执行完后，已经成功爬取了网页前20篇新闻，并筛选出热点新闻中的标题和文章链接。由于offset能控制从那一篇开始爬取，接下来，我们需要通过改变offset参数的值，提取网页前60篇新闻中的热点新闻标题与文章链接。

目的3：爬取网页前60篇新闻中的热点新闻标题与文章链接。

关键代码3：

```
import requests

headers={'user-agent':'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like (

url='https://www.toutiao.com/api/search/content/'

offset=0

# 循环爬取60条数据
while True:
    params={'aid': '24', 'app_name': 'web_search', 'offset': offset, 'format': 'json', 'keyword': '特朗普
'cur_tab': '1', 'from': 'search_tab', 'pd': 'synthesis', 'timestamp': '1597805260196', '_signature': 'd
    res=requests.get(url,headers=headers,params=params)
    articles=res.json()
    data=articles['data']

    for i in data:
        try:
            list1=[i['title'],i["article_url"]]
            print(list1)
        except:
            pass

    # 每次循环，offset加20
    offset=offset+20
    if offset == 60:
        break
```

代码解析：

1. 该代码是通过控制起始爬取的新闻篇，也就是offset参数，来实现60篇新闻的爬取，共分为了三组来爬取，每组按20篇的数量，分别是第1篇-第20篇，第21篇-第40篇，第41篇-60篇。

2.2.3 存储数据

目的： 将爬取到的热点新闻中的标题和文章链接，存放到csv文档中。

关键代码：

```
# 导入模块
```

```
2  import csv
3
4  # 新建csv表格
5  csv_file=open('articles.csv','w',newline='',encoding='utf-8')
6
7  # 往csv表格写入内容
8  writer = csv.writer(csv_file)
9  list2=['标题','链接']
10 writer.writerow(list2)
11
12 # 关闭csv文件
13 csv_file.close()
```

代码解析：

1. 第5行代码，调用open()函数打开csv文件，传入参数：文件名“articles.csv”、写入模式“w”、newline=''。
2. 当运行完代码后，打开articles.csv，如出现乱码，则需要在第5行代码中，将encoding='utf-8'改为encoding='utf-8-sig'。
3. 第10行代码，writer.writerow()和writer.writerrows()的区别在于，前者只能每次写入一个列表，而后者可以同时写入多个列表，例如writer.writerrow(list2)，writer.writerrows(list1,list2)。