HW1.2-作业

PB16080377，聂雷海

1. 爬虫概论
2. 目标站点：今日头条街拍美图（https://www.toutiao.com/search/?keyword=%E8%A1%97%E6%8B%8D）
3. 独立完成（参考书本）
4. Python爬虫
5. 目的：抓取 美图
6. 爬虫设计

用Python模拟Ajax请求，提取美图并下载。

1. 获取网页：

利用浏览器开发者工具，观察Headers和URL.根据经验，通过改变offset，来实现不同页面的访问。（offset区分不同url，较为常见）操作是将网站get请求需要的参数配置好，然后利用requests.get方法抓取url的信息。

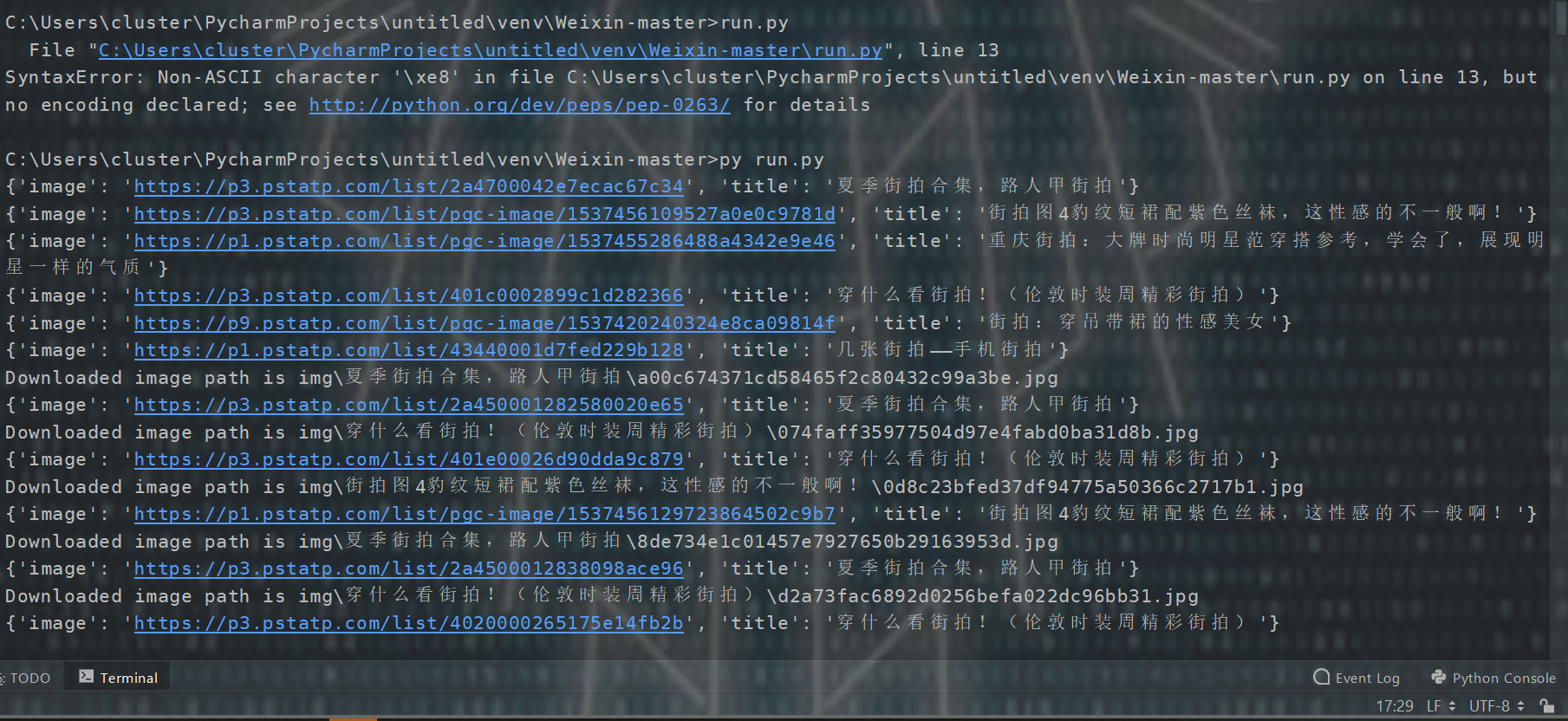
1. 获取图片：

将先前得到的链接信息转换为JSON格式。然后找到data,title内容，翻出Image\_list,此时用生成器将每一个图片链接和标题返回。

3）保存图片：

操作（2）返回有关图片的字典，为方便管理，利用item的title各自创建文件夹，然后请求链接(字典内img存储的url)，然后以二进制形式写入文件。

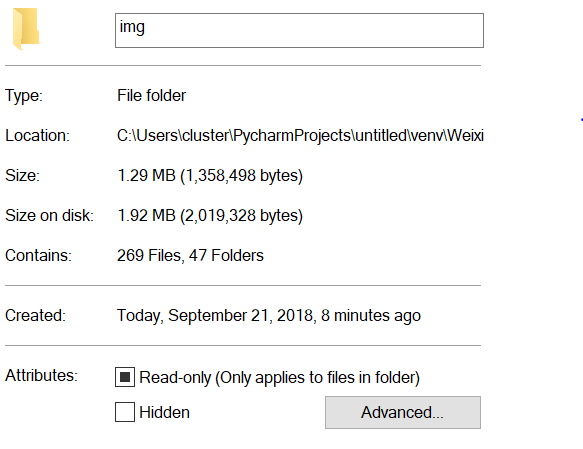
1. 爬取过程
2. 执行py run.py
3. 15s左右后停止。



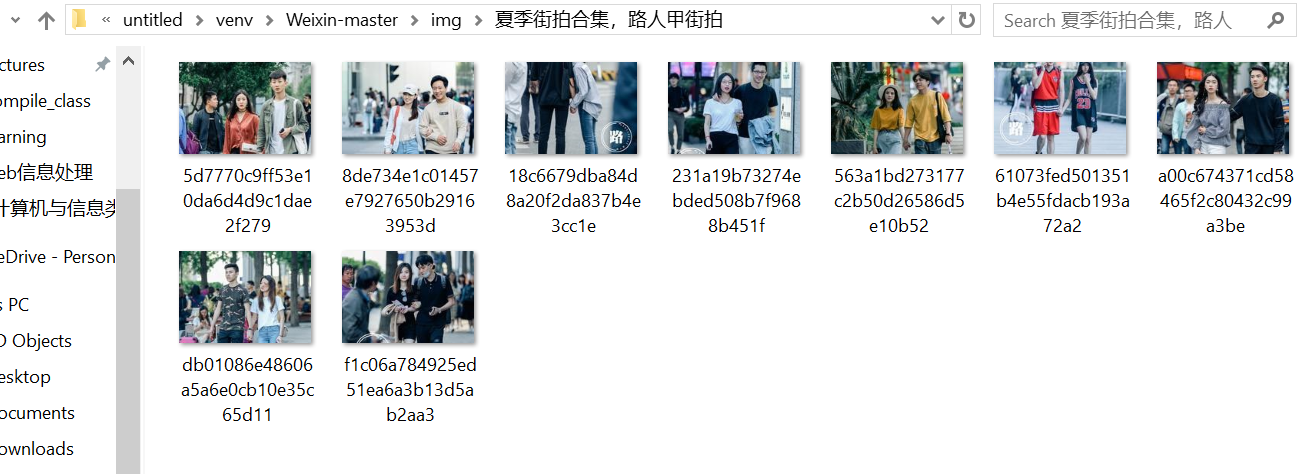
1. 爬取结果
2. 产生了47个文件夹。



2）一共2M**B**-内容



1. 结果预览：



1. 评价：

未考虑效率，没有使用分布式爬虫。用了简单粗暴的方式，顺序查找页面，然后下载.好在没有反爬虫机制,不用设置hearders.

附录——源码：

*import* requests

*from* urllib.parse *import* urlencode

*from* requests *import* codes

*import* os

*from* hashlib *import* md5

*from* multiprocessing.pool *import* Pool

def get\_page(*offset*):

params = {

'offset': offset,

'format': 'json',

'keyword': '街拍',

'autoload': 'true',

'count': '20',

'cur\_tab': '1',

'from': 'search\_tab'

}

base\_url = 'https://www.toutiao.com/search\_content/?'

url = base\_url + urlencode(params)

*try*:

resp = requests.get(url)

*if* codes.ok == resp.status\_code:

*return* resp.json()

*except* requests.ConnectionError:

*return* None

def get\_images(*json*):

*if* json.get('data'):

data = json.get('data')

*for* item in data:

*if* item.get('cell\_type') is not None:

*continue*

title = item.get('title')

images = item.get('image\_list')

*for* image in images:

*yield* {

'image': 'https:' + image.get('url'),

'title': title

}

def save\_image(*item*):

img\_path = 'img' + os.path.sep + item.get('title')

*if* not os.path.exists(img\_path):

os.makedirs(img\_path)

*try*:

resp = requests.get(item.get('image'))

*if* codes.ok == resp.status\_code:

file\_path = img\_path + os.path.sep + '{file\_name}.{file\_suffix}'.format(

*file\_name*=md5(resp.content).hexdigest(),

*file\_suffix*='jpg')

*if* not os.path.exists(file\_path):

*with* open(file\_path, 'wb') *as* f:

f.write(resp.content)

print('Downloaded image path is %s' % file\_path)

*else*:

print('Already Downloaded', file\_path)

*except* requests.ConnectionError:

print('Failed to Save Image，item %s' % item)

def main(*offset*):

json = get\_page(offset)

*for* item in get\_images(json):

print(item)

save\_image(item)

GROUP\_START = 0

GROUP\_END = 7

*if* \_\_name\_\_ == '\_\_main\_\_':

pool = Pool()

groups = ([x \* 20 *for* x in range(GROUP\_START, GROUP\_END + 1)])

pool.map(main, groups)

pool.close()

pool.join()