# -\*- coding: utf-8 -\*-

"""

Spyder Editor

This is a temporary script file.

"""

爬虫步骤：

1，构造URL：右击界面，点击“检查”，再讲页签切换到NetWork下，点击我们的请求，可以在右侧方框内看到我们的Request URL。URL = 请求的协议（http/https）+网站的域名（www.baidu.com）+资源的路径+参数。

2，通过http（客户端和服务器端请求和应答的网络协议）发起requests->response

2.1 get-headers

import requests

url = 'https://www.zhihu.com.'

headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36'}

response = requests.get(url, headers=headers)

response.encoding = 'utf-8'

print(response.text)

2.2 post

import requests

url = 'https://fanyi.baidu.com'

data = {'from': 'zh',

'to': 'en',

'query': '人生苦短，我用python'

}

response = requests.post(url, data=data)

print(response)

3,response返回为html:正则表达式或lxml+xpath（XML Path Language，它是一种用来确定XML文档中某部分位置的语言）提取数据

3.1 xpath

from lxml import etree

xml="""<bookstore>

<book>

<title lang="en">Harry Potter</title>

<author>J K. Rowling</author>

<year>2005</year>

<price>29.95</price>

</book>

<book>

<title lang="chs">Python爬虫</title>

<author>Joe</author>

<year>2018</year>

<price>39.95</price>

</book>

</bookstore>

"""

#找根节点

root=etree.fromstring(xml)

print(root)

#book子节点

elements=root.xpath('book')

print(elements)

print(elements[0])

print(elements[1])

print(elements[0].getchildren()[0].text)

#查找属性

info(elements[0])

#不关心节点结构的方法来获取数据

attributesElements=root.xpath("//@lang")

print(attributesElements)

elements=root.xpath("/bookstore/book[price>30]/title")

for it in elements:

print(it.text)

3.1.1 通过etree.HTML()获取到html的Element对象

from lxml import etree

url = 'http://http://fanyi.youdao.com/'

response = requests.get(url, headers=headers)

html\_str = response.content.decode()

print(type(html\_str))

html = etree.HTML(html\_str)

print(html)

3.1.2 通过html.xpath(xpath路径)提取到我们想要的数据

import requests

from lxml import etree

url1 = 'https://book.douban.com/tag/%E6%97%A5%E6%9C%AC%E6%96%87%E5%AD%A6?start=0&type=T/'

headers = {'user-agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36'}

response = requests.get(url1, headers=headers)

html\_str = response.content.decode()

html = etree.HTML(html\_str)

#创建xpath列表

html\_name = html.xpath("//ul[@class='subject-list']/li")

for li in html\_name:

item = {}

#获取到每一个li下的书籍名字，并去除多余的\n

item['name'] = li.xpath('./div[2]/h2/a/text()')[0].replace("\\n", " ").strip()

print(item)

3.2 response返回为json(JavaScript Object Notation，是轻量级的文本数据交换格式):json提取数据

import json

3.2.1 json.dumps()用于将dict类型的数据转成str

test1\_dict = {'NAME': 'sw', 'phone': '10086'}

print(test1\_dict)

print('json.dumps转换前的类型是：', type(test1\_dict))

test1\_str = json.dumps(test1\_dict)

print(test1\_str)

print('json.dumps转换后的类型是：', type(test1\_str))

例子：当我们在把字典类型的数据写入文本时，是写入不成功的。这个时候就需要将字典类型转换成字符串，再写入到文本之中，json.dumps()的作用就是实现这一功能。

import requests

import json

url = 'http://fanyi.youdao.com/translate?smartresult=dict&smartresult=rule'

headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36'}

data = {'i': '人生苦短，我用python',

'from': 'AUTO',

'to': 'AUTO',

'smartresult': 'dict',

'client': 'fanyideskweb',

'salt': '15598805838805',

'sign': '601e9410133b355529e58d23a6c60578',

'ts': '1559880583880',

'bv': '565657d9b2f836d2c4c3a1fd81d7b3c3;',

'doctype': 'json',

'version': '2.1',

'keyfrom': 'fanyi.web',

'action': 'FY\_BY\_CLICKBUTTION'

}

response = requests.post(url, data=data, headers=headers)

html\_str = response.content.decode()

#将字符串转换成字典

dict\_json = json.loads(html\_str)

#将字典转换成字符串并写入fnayi.txt文件中

fanyi\_str = json.dumps(dict\_json, ensure\_ascii=False, indent=2)

with open('fanyi.txt', 'w', encoding='utf-8') as f:

f.write(fanyi\_str)

3.2.2 json.loads()用于将str类型的数据转成dict

test2\_dict = json.loads(test1\_str)

print(test2\_dict)

print('json.loads转换后的类型是：', type(test2\_dict))

例子：

import requests

import json

url = 'http://fanyi.youdao.com/translate?smartresult=dict&smartresult=rule'

headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36'}

data = {'i': '人生苦短，我用python',

'from': 'AUTO',

'to': 'AUTO',

'smartresult': 'dict',

'client': 'fanyideskweb',

'salt': '15598805838805',

'sign': '601e9410133b355529e58d23a6c60578',

'ts': '1559880583880',

'bv': '565657d9b2f836d2c4c3a1fd81d7b3c3;',

'doctype': 'json',

'version': '2.1',

'keyfrom': 'fanyi.web',

'action': 'FY\_BY\_CLICKBUTTION'

}

response = requests.post(url, data=data, headers=headers)

html\_str = response.content.decode()

#将字符串转换成字典

dict\_json = json.loads(html\_str)

#打印转换之后的数据以及数据类型

print(dict\_json)

print(type(dict\_json))

#获取翻译结果

ret = dict\_json['translateResult'][0][0]['src']

print('翻译结果是：',ret)

3.2.3 json.dump(),跟文件结合一起使用:将对象（列表）转换为字符串后，将转换后的数据写入到文件中

test1\_dict = {'NAME': 'sw', 'phone': '10086'}

print(test1\_dict)

print('test1\_dict的类型是：', type(test1\_dict))

with open('text1.txt', 'w') as f:

json.dump(test1\_dict, f)

3.2.4 json.load()用于从json文件中读取数据

with open('text1.txt', 'r') as f:

test1\_str = json.load(f)

print(test1\_str)

3.3 正则表达式获取数据（get请求）

#1)对URL发起HTTP请求http request,得到相应的http response响应， 我们需要的数据就在response的响应体里

#首先打开网页http://maoyan.com/board/4?offset=0，点击“榜单”，鼠标右键点击页面，点击“检查（N）“，Network->Headers->User-Agent

import requests

def get\_one\_page(URL):#http://maoyan.com/board/4?offset=0

ua\_headers={"User-Agent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.97 Safari/537.36"}

response=requests.get(URL,headers=ua\_headers)

if response.status\_code==200:#网页上有200部电影

return response.text

return None

print(get\_one\_page("http://maoyan.com/board/4?offset=0"))

#2)调用正则表达式

import re

#使用online正则表达式软件：source里面输入上述html文件，

#在regex中输入：<p class="name">.\*?title="([\s\S]\*?)"[\s\S]\*?<p class="star">([\s\S]\*?)</p>[\s\S]\*?<p class="releasetime">([\s\S]\*?)</p>

def parse\_one\_page(html):

pattern=re.compile('<p class="name">.\*?title="([\s\S]\*?)"[\s\S]\*?<p class="star">([\s\S]\*?)</p>[\s\S]\*?<p class="releasetime">([\s\S]\*?)</p>')

items=re.findall(pattern,html)

for it in items:

yield{

"title":it[0].strip(),

"actor":it[1].strip(),

"time":it[2].strip()

}

#3)存到本地的文件系统中或数据库中

import json

with open('cateye.txt','a',encoding="utf-8") as f:

f.write(json.dumps(item, ensure\_ascii=False)+'\n')

##动态爬虫百度

import urllib

url= "https://baike.baidu.com/s?"

keyword=input("请输入您需要搜索的信息：")

wd={"wd":keyword}

wd=urllib.parse.urlencode(wd)

fullUrl=url+wd

print(fullUrl)

#1对百度发起get请求

ua\_headers={"User-Agent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.97 Safari/537.36"}

reg=urllib.request.Request(fullUrl, headers=ua\_headers)

response=urllib.request.urlopen(reg)

with open("baiduSearch.html","wb") as f:

f.write(response.read())

#2使用正则表达式匹配百度对推荐

import re

pattern=re.compile(""" <a href="([s\S]\*?)" """)

items=re.findall(pattern, response.read().decode('utf-8'))

for it in items:

print(it[0],it[1])

with open("save\_baiduSearch.txt","a") as f:

f.write(json.dumps(item, ensure\_ascii=False)+'\n')

4,保存数据:

#with open('save.txt', 'a', encoding='utf-8') as f:

#f.write()

# f.close()

import requests

import json

from lxml import etree

url1 = 'https://book.douban.com/tag/%E6%97%A5%E6%9C%AC%E6%96%87%E5%AD%A6?start=0&type=T/'

headers = {'user-agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36'}

response = requests.get(url1, headers=headers)

html\_str = response.content.decode()

html = etree.HTML(html\_str)

#创建xpath列表

html\_name = html.xpath("//ul[@class='subject-list']/li")

for li in html\_name:

item = {}

#获取到每一个li下的书籍名字，并去除多余的\n

item['name'] = li.xpath('./div[2]/h2/a/text()')[0].replace("\\n", " ").strip()

with open('save\_bookname.txt', 'a', encoding='utf-8') as f:

f.write(json.dumps(item, ensure\_ascii=False)+'\n')

##simplest scraping

#1，We gonna show how to open a very simple webpage, and read all the content in it.

from urllib.request import urlopen

# 2，if has Chinese, apply decode()

html = urlopen("https://morvanzhou.github.io/static/scraping/basic-structure.html").read().decode('utf-8')

print(html)

#3，正则表达式：Then we select some text according to the tag arround text using regular expression

import re

res = re.findall(r"<title>(.+?)</title>", html)

print("\nPage title is: ", res[0])

#3.1，Another example to select paragraph content out of html.

res = re.findall(r"<p>(.\*?)</p>", html, flags=re.DOTALL) # re.DOTALL if multi line

print("\nPage paragraph is: ", res[0])

#3.2，And select links using regex

res = re.findall(r'href="(.\*?)"', html)

print("\nAll links: ", res)

#4，beautiful soup

from bs4 import BeautifulSoup

from urllib.request import urlopen

#4.1， if has Chinese, apply decode()

html = urlopen("https://morvanzhou.github.io/static/scraping/basic-structure.html").read().decode('utf-8')

print(html)

#4.2Parsing this html using a method called lxml, create a soup object. You can simply "h1" or "p" to call the heading 1 and paragraph tag from soup.

soup = BeautifulSoup(html, features='lxml')

print(soup.h1)

print('\n', soup.p)

#4.3 Or using some helpful functions to find tags. Access the attribute of found tags using a key just like what you do in a python dictionary.

all\_href = soup.find\_all('a')

all\_href = [l['href'] for l in all\_href]

print('\n', all\_href)

#4.4CSS 在装饰每一个网页部件的时候, 都会给它一个名字. 而且一个类型的部件, 名字都可以一样

#4.4.1First we still need to open a page, then we can apply beautifulsoup on this page's html.

from bs4 import BeautifulSoup

from urllib.request import urlopen

# if has Chinese, apply decode()

html = urlopen("https://morvanzhou.github.io/static/scraping/list.html").read().decode('utf-8')

print(html)

#4.4.2 Parsing this html using a method called lxml, create a soup object. Find all "li" tag which has a class=month.

soup = BeautifulSoup(html, features='lxml')

# use class to narrow search

month = soup.find\_all('li', {"class": "month"})

for m in month:

print(m.get\_text())

#Or using some helpful functions to find tags. Access the attribute of found tags using a key just like what you do in a python dictionary

jan = soup.find('ul', {"class": 'jan'})

d\_jan = jan.find\_all('li') # use jan as a parent

for d in d\_jan:

print(d.get\_text())

#5.find by regual expression

#5.1First we import re for regex. Then, open a page, then we can apply beautifulsoup on this page's html.

from bs4 import BeautifulSoup

from urllib.request import urlopen

import re

# if has Chinese, apply decode()

html = urlopen("https://morvanzhou.github.io/static/scraping/table.html").read().decode('utf-8')

print(html)

#5.2Parsing this html using a method called lxml, create a soup object. Find all "img" tag which has a src in a given pattern.

soup = BeautifulSoup(html, features='lxml')

img\_links = soup.find\_all("img", {"src": re.compile('.\*?\.jpg')})

for link in img\_links:

print(link['src'])

#5.3Or using some helpful functions to find tags. Access the attribute of found tags using a key just like what you do in a python dictionary.

course\_links = soup.find\_all('a', {'href': re.compile('https://morvan.\*')})

for link in course\_links:

print(link['href'])

#6,other example

doc=['<html><head><title>Page title</title></head>',

'<body><p id="firstpara" align="center">This is paragraph <b>one</b>.',

'<p id="secondpara" align="blah">This is paragraph <b>two</b>.',

'</html>']

soup=BeautifulSoup(''.join(doc),"html.parser")

print(soup)

print(soup.prettify())

html=soup.contents[0]

print(html.name)

body=soup.contents[0].contents[1]

print(body.name)

items=soup.findAll('p',id="firstpara")

for it in items:

print(it.text)

#7,scrape baidu news

#7.1Here we build a scraper to crawl Baidu Baike from this page onwards. We store a historical webpage that we have already visited to keep tracking it.

from bs4 import BeautifulSoup

from urllib.request import urlopen

import re

import random

base\_url = "https://baike.baidu.com"

his = ["/item/%E7%BD%91%E7%BB%9C%E7%88%AC%E8%99%AB/5162711"]

#7.2Select the last sub url in "his", print the title and url.

url = base\_url + his[-1]

html = urlopen(url).read().decode('utf-8')

soup = BeautifulSoup(html, features='lxml')

print(soup.find('h1').get\_text(), ' url: ', his[-1])

#7.3Find all sub\_urls for baidu baike (item page), randomly select a sub\_urls and store it in "his". If no valid sub link is found, than pop last url in "his".

# find valid urls

sub\_urls = soup.find\_all("a", {"target": "\_blank", "href": re.compile("/item/(%.{2})+$")})

if len(sub\_urls) != 0:

his.append(random.sample(sub\_urls, 1)[0]['href'])

else: # no valid sub link found

his.pop()

print(his)

#7.4Put everthing together. Random running for 20 iterations. See what we end up with.

his = ["/item/%E7%BD%91%E7%BB%9C%E7%88%AC%E8%99%AB/5162711"]

for i in range(20):

url = base\_url + his[-1]

html = urlopen(url).read().decode('utf-8')

soup = BeautifulSoup(html, features='lxml')

print(i, soup.find('h1').get\_text(), ' url: ', his[-1])

# find valid urls

sub\_urls = soup.find\_all("a", {"target": "\_blank", "href": re.compile("/item/(%.{2})+$")})

if len(sub\_urls) != 0:

his.append(random.sample(sub\_urls, 1)[0]['href'])

else: # no valid sub link found

his.pop()

##requests method

from urllib import request

import requests

import webbrowser

param = {"wd": "莫烦Python"}

r = requests.get('http://www.baidu.com/s', params=param)

print(r.url)

webbrowser.open(r.url)

#Download something:Using requests.get to download at once.

import os

os.makedirs('./img/', exist\_ok=True)

IMAGE\_URL = "https://morvanzhou.github.io/static/img/description/learning\_step\_flowchart.png"

import requests

r = requests.get(IMAGE\_URL)

with open('./img/image2.png', 'wb') as f:

f.write(r.content)

response=requests.get("http://www.sina.com.cn")

response.encoding="utf-8"

print (response.text)

with open("sina.html",'wb') as f:

f.write(response.text.encode('utf-8'))

req=request.Request('http://www.sina.com.cn')

response=request.urlopen(req)

print(response.read().decode('utf-8'))

###

from urllib import request

import requests

def info(object,spacing=20,collapse=1):

methodList=[method for method in dir(object)

if callable(getattr(object,method))]

processFunc=collapse and (lambda s:" ".join(s.split())) or (lambda s:s)

print('\n'.join(["%s %s"%(method.ljust(spacing),processFunc(str(getattr(object,method).\_\_doc\_\_)))

for method in methodList]))

info(requests)

import builtwith

print(builtwith.parse("http://www.sina.com.cn"))

import whois

print(whois.whois("https://www.baidu.com"))

import requests

response=requests.get("http://www.sina,com.cn")

response.encoding='utf-8'

print(response.status\_code)

##multiprocessing Distributed scraping:

1,set up

import multiprocessing as mp

import time

from urllib.request import urlopen, urljoin

from bs4 import BeautifulSoup

import re

base\_url = 'https://morvanzhou.github.io/'

if base\_url != 'https://morvanzhou.github.io/':

restricted\_crawl = True

else:

restricted\_crawl = False

2,Create a crawl function to open a url in parallel.

def crawl(url):

response = urlopen(url)

time.sleep(0.1) # slightly delay for downloading

return response.read().decode()

3,Create a parse function to find all results we need in parallel

def parse(html):

soup = BeautifulSoup(html, 'lxml')

urls = soup.find\_all('a', {"href": re.compile('^/.+?/$')})

title = soup.find('h1').get\_text().strip()

page\_urls = set([urljoin(base\_url, url['href']) for url in urls])

url = soup.find('meta', {'property': "og:url"})['content']

return title, page\_urls, url

4,Create a process pool and scrape parallelly.

unseen = set([base\_url,])

seen = set()

pool = mp.Pool(4)

count, t1 = 1, time.time()

while len(unseen) != 0: # still get some url to visit

if restricted\_crawl and len(seen) > 20:

break

print('\nDistributed Crawling...')

crawl\_jobs = [pool.apply\_async(crawl, args=(url,)) for url in unseen]

htmls = [j.get() for j in crawl\_jobs] # request connection

print('\nDistributed Parsing...')

parse\_jobs = [pool.apply\_async(parse, args=(html,)) for html in htmls]

results = [j.get() for j in parse\_jobs] # parse html

print('\nAnalysing...')

seen.update(unseen) # seen the crawled

unseen.clear() # nothing unseen

for title, page\_urls, url in results:

print(count, title, url)

count += 1

unseen.update(page\_urls - seen) # get new url to crawl

print('Total time: %.1f s' % (time.time()-t1, )) # 27.4 s !!!

&smartresult=ugc&sessionFrom=null

###post请求交互模式

from urllib import request,parse

youdaoUrl="http://fanyi.youdao.com/translate\_o?smartresult=dict&smartresult=rule"

headers={"User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36",

"X-Requested-With":"XMLHttpRequest",

"Accept": "application/json, text/javascript, \*/\*; q=0.01",

"Content-Type": "application/x-www-form-urlencoded; charset=UTF-8"}

isOut=False

while True:

if isOut==True:

break

key=input("请输入您要翻译的英文，输入closeme则退出:")

if key=="closeme":

isOut=True

continue

formdata={"i":key,

"from":"AUTO",

"to":"AUTO",

"smartresult":"dict",

"client":"fanyideskweb",

"salt":"15745550684518",

"sign":"1da6f1fa46ae49241446e532a89dfb7a",

"ts":"1574555068451",

"bv":"9ed1df12e1ecacf0048d2b29195a0070",

"doctype":"json",

"version":"2.1",

"keyfrom":"fanyi.web",

"action":"FY\_BY\_REALTlME",

"typoResult":"false"

}

data=bytes(parse.urlencode(formdata),encoding="utf-8")

req=request.Request(youdaoUrl,data,headers,method="POST")

response=request.urlopen(req)

info=response.read().decode("utf-8")

print(info)

#应用

抓取历年长沙市政府工作报告

1 分析网页

1.1 打开目标链接[http://video.changsha.gov.cn/xxgk/szfxxgkml/gzbg/index.html]

1.2 右键点击chrome页面，查看网页源代码,各子条目的信息都写在标签“h4”中，里面有着链接地址和名称:

<li>

<h4> <a href="./201901/t20190116\_3156774.html" target="\_blank">【长沙市政府门户网站】2019年政府工作报告</a> <span class="date">2019-01-16</span></h4>

</li>

1.3 点开索引页面中的一个链接，如“2019年政府工作报告”，页面中政府工作报告的正文就是我们需要提取的内容

http://www.changsha.gov.cn/xxgk/szfxxgkml/gzbg/201901/t20190116\_3156774.html

同样的，查看其网页源代码，可以看下以下内容：

<p style="list-style-type: none; box-sizing: border-box; margin-top: 1px; color: rgb(85,85,85); padding-bottom: 0px; padding-top: 0px; padding-left: 0px; margin-left: 1px; padding-right: 0px; margin-right: 1px">

<font style="font-size: 12pt; line-height: 150%">

现在，我代表市人民政府向大会报告工作，请予审议，并请各位政协委员和其他列席人员提出意见。

</font></p>

从图中看到，政府工作报告的正文内容都写在了标签“p”中。

2 代码实现

2.1 Step1: 用python登录索引页，读取其中的数据

# 载入所需程序包

import urllib

import urllib.request

mainsite = "http://video.changsha.gov.cn/xxgk/szfxxgkml/gzbg/index.html"

file = urllib.request.urlopen(mainsite)

mainpagedata = file.read().decode("utf-8", "ignore")

# 在python中查看读取的内容

print(mainpagedata)

2.2 Step2: 读取标签”h4“中的内容，并进行处理

from bs4 import BeautifulSoup

# 用beautifulsoup包对网站数据进行解析

mainsoup = BeautifulSoup(mainpagedata, "html.parser")

# 有用信息都只保存在'h4'标签中，直接提取'h4'中的元素，保存在列表maintags中

maintags = mainsoup.find\_all('h4')

# 查看maintags中的是怎样的数据

print(maintags)

#从后台可以看到，maintags是一个列表，其中的元素是每一个h4标签的内容。我们需要的是其中的链接地址和链接名称，因此要对读取到的数据进行清洗。

# 生成两个空列表，备用

list\_link = []

list\_name = []

# 将h4中的信息解析出来，分别为链接和标签，存放在以上两个空列表中

# 循环，遍历maintags中的对象

for t in maintags:

# 提取标签中的文本信息

name = t.get\_text().strip()

word = "政府工作报告"

# 条件判断，文本中出现“政府工作报告”的条目才被保存

if word in name:

name = "长沙" + re.sub(r"\D","", name) # 提取数字

name = name[:6] # 字符截取

link = t.find().attrs['href'] # 提取'href'中信息

# 字符处理，形成有效的链接

flink = "http://video.changsha.gov.cn/xxgk/szfxxgkml/gzbg"

link = flink+link[1:]

# 将提取出来的信息存在两个列表中

list\_link.append(link)

list\_name.append(name)

dic\_linkname = dict(zip(list\_link,list\_name)) # 将列表合并为字典

print(dic\_linkname) # 打印查看

#从打印出来的信息可以看到，构建的字典类型的对象dic\_linkname已经符合我们进行下一步操作的需要了。其结构为一个个的链接地址—链接名称的键值对：

2.3 Step3: 分别读取各链接中的数据，处理后保存

for l, n in dic\_linkname.items(): # 遍历dic\_linkname中的对象

link = l

name = n

# 提取网页信息

file = urllib.request.urlopen(link)

pagedata = file.read().decode("utf-8", "ignore")

# 利用BeautifulSoup解析网页信息

soup = BeautifulSoup(pagedata, "html.parser")

# 提取所有标签p中的内容，并将其保存并在content中

tags = soup.find\_all('p')

content = []

for t in tags:

text = t.get\_text().strip()

content.append(text)

content = "".join(content)

# 指定保存路径，将content中的内容写入到路径下的文档中，并链接对应的链接名称命名

path = '/Users/hym0509/Desktop/out/'

storePath = path + name +".txt"

fhandle = open(storePath, "w", encoding='utf-8')

fhandle.write(content)

fhandle.close()

#运行以上代码后，打开文件夹 /Users/hym0509/Desktop/out/后，

#我们就能看到程序提取的各年度长沙政府工作报告已经保存在了相应的文本文件里。