WISE R Club

-------网络爬虫（Python系列）

钱晨

What is web crawler ?

Basic steps of web crawler:

1) Make request message to the server

2) To parse the HTML and get the information you want from it

3) Write the information to the file you want (csv ,xlsx, SQL and etc)

目录

contents

01

PART ONE

HTTP protocol

01/ Overview of http protocol

The HTTP is the protocol（协议） programs use to communicate over the World Wide Web.

How can a web server work?

(1). Set up connection—accept a client connection
(2). Receive request—read an HTTP request message from the network.
(3). Process request—interpret the request message and take action.
(4). Access resource—access the resource specified in the message.
(5). Construct response—create the HTTP response message with the right headers.
(6). Send response—send the response back to the client.
(7). Log transaction—place notes about the completed transaction in a log file.

01/ Overview of http protocol

Web Clients and Servers（web客户端和服务器）

Web content lives on web servers. Web servers speak the HTTP protocol, so they are often called HTTP servers. The clients send HTTP requests to servers, and servers return the requested data in HTTP responses.

When you browse to a page, such as “ http://channel.jd.com/1713-3259.html” your browser sends an HTTP request to the server “channel.jd.com”. The server tries to find the desired object (in this case, “1713-3259.html”) and, if successful, sends the object to the client in an HTTP response, along with the type of the object, the length of the object, and other information.

01/ Overview of http protocol

URIs（URLs）

Each web server resource has a name, so clients can point out what resources they are interested in. The server resource name is called a **uniform resource identifier**, or URI. URIs are like the postal addresses of the Internet, uniquely identifying and locating information resources around the world. URIs come in two types, called URLs and URNs.

The uniform resource locator (URL) is the most common form of resource identifier. **URLs describe the specific location of a resource on a particular server**. They tell you exactly how to fetch a resource from a precise, fixed location.

URNs are still experimental and not yet widely adopted.

01/ Overview of http protocol

Transactions

An HTTP transaction consists of a request command (sent from client to server), and a response result (sent from the server back to the client). This communication happens with formatted blocks of data called HTTP messages.

Methods

GET Send named resource from the server to the client.
PUT Store data from client into a named server resource.
DELETE Delete the named resource from a server.
POST Send client data into a server gateway application.
HEAD Send just the HTTP headers from the response for the named resource.

01/ Overview of http protocol

Status code

200~299 表示成功
300~399表示资源已经被移走
400~499表示出错
404 :Not found　　401:unauthorized　　403:forbidden
408：timeout
502:

Messages

HTTP messages sent from web clients to web servers are called request messages. Messages from servers to clients are called response messages.

Start line : it describes the method of browsing, URL, status and some other information.
Header: followed by the start line.
Request bodies carry data to the web server

01/ Overview of http protocol

Connections

Before an HTTP client can send a message to a server, it needs to establish a TCP/IP connection between the client and server using Internet protocol (IP) addresses and port numbers.

Architectural Components of the Web(web的结构组件)

Proxies

A proxy sits between a client and a server, receiving all of the client's HTTP requests and relaying the requests to the server (perhaps after modifying the requests).
Proxies can also filter requests and responses;

Agents

User agents (or just agents) are client programs that make HTTP requests on the user's behalf.

01/ URL

URIs（URLs）

Each web server resource has a name, so clients can point out what resources they are interested in. The server resource name is called a **uniform resource identifier**, or URI. URIs are like the postal addresses of the Internet, uniquely identifying and locating information resources around the world. URIs come in two types, called URLs and URNs.

The uniform resource locator (URL) is the most common form of resource identifier. **URLs describe the specific location of a resource on a particular server**. They tell you exactly how to fetch a resource from a precise, fixed location.

URNs are still experimental and not yet widely adopted.

01/ URL

Structure of URL

http://channel.jd.com/1713-3258.html

• The first part of the URL (http) is the URL scheme. The scheme tells a web client how to access the resource. In this case, the URL says to use the HTTP protocol.

• The second part of the URL (channel.jd.com) is the server location. This tells the web client where the resource is hosted.

• The third part of the URL (/1713-3258.html) is the resource path. The path tells what particular local resource on the server is being requested.

01/ URL

URL Syntax

Most URL schemes base their URL syntax on this nine-part general format:

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

The three most important parts of a URL are **the scheme, the host, and the path**

01/ URL

Scheme which protocol to use when accessing a server to get a resource.
Such as http, https, ftp, file and etc.

Host the hostname or dotted IP address of the server hosting the resource.

Port the port number on which the server hosting the resource is listening. Many schemes have default port numbers (the default port number for HTTP is 80, https is 443.).

The host and port components of the URL provide these two pieces of information about what machine is hosting the resource and where on that machine it can find the server that has access to the desired resource

01/ URL

Path 服务器上资源的本地名，用一个/与URL其他部分分割开来。他们之间用；分割

Params(参数) Used by some schemes to specify input parameters. Params are name/value pairs. A URL can contain multiple params fields, separated from themselves and the rest of the path by (;).They provide applications with any additional information that they need to access the resource

Query（查询） Used by some schemes to pass parameters to active applications (such as databases, bulletin boards, search engines, and other Internet gateways). It is separated from the rest of the URL by the “?” character.

01/ URL

Frag（片段） A name for a piece or part of the resource. The frag field is not passed to the server when referencing the object; it is used internally by the client. It is separated from the rest of the URL by the “#” character.

Some examples:

http://search.jd.com/Search?keyword=纯粹理性批判&enc=utf-8&suggest=1.def.0&wq=纯粹&pvid=thd53boi.9oosaa

https://list.tmall.com/search_product.htm?q=%B4%BF%B4%E2%C0%ED%D0%D4%C5%FA%C5%D0&type=p&spm=a220m.1000858.a2227oh.d100&xl=%B4%BF%B4%E2%C0%ED%D0%D4_1&from=.list.pc_1_suggest

01/ URL

https://list.tmall.com/search_product.htm?spm=a220m.1000858.1000720.1.Z15OgH&brand=84669&q=%B7%C9%D0%D0%BC%D0%BF%CB&sort=s&style=g&from=sn_1_brand-qp&active=1&tmhkmain=0#J_crumbs

URLs support a frag component to identify pieces within a resource. After your browser gets the entire resource from the server, it then uses the fragment to display the part of the resource in which you are interested.

01/ URL

Comparison of http and https:

http The Hypertext Transfer Protocol scheme conforms to the general URL format, except that there is no username or password. The port defaults to 80 if omitted.

https The https scheme is a twin to the http scheme. The only difference is that the https scheme 使用了Netscape的SSL。 SSL为其提供了端到端的加密机制。语法同http相同默认端口是443. 是以安全为目标的HTTP通道

01/ Message

The Parts of a Message

Each message contains either a request from a client or a response from a server. They consist of three parts: a start line describing the message, a block of headers containing attributes, and an optional body containing data.

Message syntax:

All http messages can be divided into two types: response and requests messages.
Both request and response messages have the same basic message structure.

01/ Message

Request message:

<method> <request-URL> <version>
<headers>
<entity-body>

Response message:

<version> <status> <reason-phrase>#reason phrase是status code文本解释
<headers>
<entity-body

01/ Message

Start lines:

Request line: method+ URL+ version
Response line: version+ status code+ reason phras

e.g:

HEAD /doc/6746680-6961226.html HTTP/1.1

HTTP/1.1 200 OK

01/ Message

some types of methods:

1.Get:

GET Get a document from the server. 通常用于请求服务器发送某个资源

GET/Search?callback=call13033&ad_ids=576%3A1&ad_type=4&area=1&enc=utf-8&keyword=%E9%85%B1%E6%B2%B9&xtest=new_search&page=1&_=1463474549877 HTTP/1.1

2. POST

Send data to the server for processing.

01/ Message

Status code

200~299 表示成功

300~399表示资源已经被移走

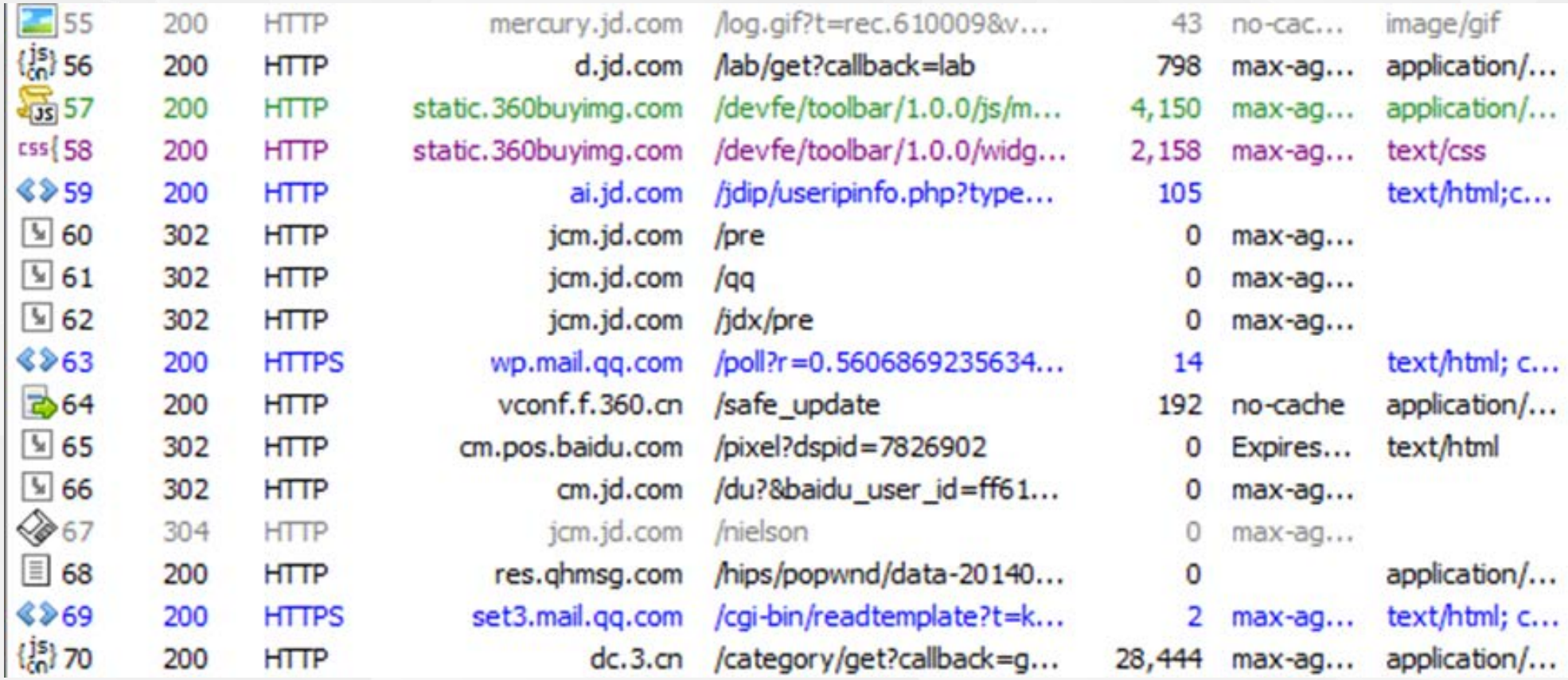
400~499表示出错

404 :Not found　　401:unauthorized　　403:forbidden

408：timeout

502:

01/ Message



55	200	HTTP	mercury.jd.com	/log.gif?t=rec.610009&v...	43	no-cac...	image/gif
56	200	HTTP	d.jd.com	/lab/get?callback=lab	798	max-ag...	application/...
57	200	HTTP	static.360buyimg.com	/devfe/toolbar/1.0.0/js/m...	4,150	max-ag...	application/...
58	200	HTTP	static.360buyimg.com	/devfe/toolbar/1.0.0/widg...	2,158	max-ag...	text/css
59	200	HTTP	ai.jd.com	/jdip/useripinfo.php?type...	105		text/html;c...
60	302	HTTP	jcm.jd.com	/pre	0	max-ag...	
61	302	HTTP	jcm.jd.com	/qq	0	max-ag...	
62	302	HTTP	jcm.jd.com	/jdx/pre	0	max-ag...	
63	200	HTTPS	wp.mail.qq.com	/poll?r=0.5606869235634...	14		text/html; c...
64	200	HTTP	vconf.f.360.cn	/safe_update	192	no-cache	application/...
65	302	HTTP	cm.pos.baidu.com	/pixel?dspid=7826902	0	Expires...	text/html
66	302	HTTP	cm.jd.com	/du?&baidu_user_id=ff61...	0	max-ag...	
67	304	HTTP	jcm.jd.com	/nielson	0	max-ag...	
68	200	HTTP	res.qhmsg.com	/hips/popwnd/data-20140...	0		application/...
69	200	HTTPS	set3.mail.qq.com	/cgi-bin/readtemplate?t=k...	2	max-ag...	text/html; c...
70	200	HTTP	dc.3.cn	/category/get?callback=g...	28,444	max-ag...	application/...

01/ Message

Headers:

1.request message

Connection: Allows clients and servers to specify options about the request/response connection.
Content-Type，内容类型，一般是指网页中存在的Content-Type，用于定义网络文件的类型和网页的编码，决定浏览器将以什么形式、什么编码读取这个文件
Update
Cache-control: 这个字段用于指定所有缓存机制在整个请求/响应链中必须服从的指令

Public	所有内容都将被缓存(客户端和代理服务器都可缓存)
Private	内容只缓存到私有缓存中(仅客户端可以缓存，代理服务器不可缓存)
public max-age=xxx (xxx is numeric)	缓存的内容将在 xxx 秒后失效，这个选项只在 HTTP 1.1 可用
No-store	不缓存

01/ Message

Pragma:和cache-control一样是一种随http message传输的一种指令但不专用于缓存

Host： 提供了主机名及端口号

Referer 提供给服务器客户端从那个页面链接过来的信息

Origin： Origin字段里只包含是谁发起的请求，并没有其他信息。跟Referer不一样的是，Origin字段并没有包含涉及到用户隐私的URL路径和请求内容，这个尤其重要。并且Origin字段只存在于POST请求，而Referer则存在于所有类型的请求。

User agent: 发送请求的应用程序名

01/ Message

Authorization：用于自身认证，是客户端传输给服务器的数据

Cookie ： Used by clients to pass a token（令牌） to the server—not a true security header, but it does have security implications

Cookie2： Used to note the version of cookies a requestor supports;

Response headers

Set-Cookie ： Not a true security header, but it has security implications; used to set a token on the client side that the server can use to identify the client。

01

PART ONE

HTML

01/ HTML

什么是 HTML?

HTML 是用来描述网页的一种语言。

HTML 指的是超文本标记语言 (Hyper Text Markup Language)

HTML 不是一种编程语言，而是一种标记语言 (markup language)

标记语言是一套标记标签 (markup tag)

HTML 使用标记标签来描述网页

01/ HTML

HTML 标签

HTML 标记标签通常被称为 HTML 标签 (HTML tag)。

HTML 标签是由尖括号包围的关键词，比如 <html>

HTML 标签通常是成对出现的，比如 和

标签对中的第一个标签是开始标签，第二个标签是结束标签

开始和结束标签也被称为开放标签和闭合标签

01/ HTML

class
classname
规定元素的类名（classname）

id
规定元素的唯一 id

style
style_definition
规定元素的行内样式（inline style）

title
text
规定元素的额外信息（可在工具提示中显示）

01/ HTML

HTML 属性

HTML 标签可以拥有属性。属性提供了有关 HTML 元素的更多的信息。

属性总是以名称/值对的形式出现，比如：name="value"。

属性总是在 HTML 元素的开始标签中规定。

01/ HTML

HTML 链接由 <a> 标签定义。链接的地址在 href 属性中指定：

This is a link

属性例子 1:

<h1> 定义标题的开始。

<h1 align="center"> 拥有关于对齐方式的附加信息。

TIY : 居中排列标题

属性例子 2:

<body> 定义 HTML 文档的主体。

<body bgcolor="yellow"> 拥有关于背景颜色的附加信息。

TIY : 背景颜色

属性例子 3:

<table> 定义 HTML 表格。<table border="1"> 拥有关于表格边框的附加信息。

01/ HTML

<html> 与 </html> 之间的文本描述网页

<body> 与 </body> 之间的文本是可见的页面内容

<h1> 与 </h1> 之间的文本被显示为标题

 与 之间的文本被显示为段落

HTML 链接是通过 <a> 标签进行定义的。

HTML 图像是通过 <img> 标签进行定义的。

01/ HTML

<body> 元素：

```
<body>
<p>This is my first paragraph.</p>
</body>
```

<body> 元素定义了 HTML 文档的主体。

这个元素拥有一个开始标签 <body>，以及一个结束标签 </body>。

元素内容是另一个 HTML 元素（p 元素）。

02

PART TWO

Python requests

02 Python requests

Basic usage of requests:

1.Make a request

```
r=requests.get(“URL”)
r=requests.post(“URL”)
r=equests.head(“URL”)
r=requests.put(“URL”)
```

2.Response Content

```
r.content
r.text
r.encoding
r.raise_for_status()#抛出异常
```

…….

02 Python requests

3.Passing Parameters In URLs

例如：对于这样的一个URL
http://search.jd.com/search?keyword=短袖T恤男&enc=utf-8&qrst=1&rt=1&stop=1&vt=2&sttr=1&offset=6&cid3=1349#J_crumbsBar

```
parameters= {'keyword': '短袖T恤', 'enc': 'utf-8',"cid3":"1349"}
r = requests.get("http://search.jd.com/search", params=parameters)
print (r.url)
```

输出结果
http://search.jd.com/search?cid3=1349&keyword=%E7%9F%AD%E8%A2%96T%E6%81%A4& enc=utf-8

02 Python requests

4.Json

r.json()
内置了json解码器
如果JSON解码失败， r.json 就会抛出一个异常。例如，相应内容是 401 (Unauthorized)，尝试访问 r.json 将会抛出 ValueError: No JSON object could be decoded 异常。

5. POST a multi-encoded file

传输的数据以字典形式进行。
如果传输的数据是json形式的，json.dumps()

02 Python requests

6.Headers:

字典形式

Proies:

字典形式

7.Timeout:

在经过以 timeout 参数设定的秒数时间之后停止等待响应:

01/ Python requests

Advanced usage of requests:

1.session object:

Session:在计算机中，尤其是在网络应用中，称为“会话控制”。Session 对象存储特定用户会话所需的属性和信息（比如说密码帐户名）。这样，当用户在应用程序的 Web 页之间跳转时，存储在 Session 对象中的变量将不会丢失，而是在整个用户会话中一直存在下去。当用户请求来自应用程序的 Web 页时，如果该用户还没有会话，则 Web 服务器将自动创建一个 Session 对象。当会话过期或被放弃后，服务器将终止该会话。Sess比如说，存储用户的首选项。

Session objective允许你通过requests库连续传输特定的参数或者cookie。

e.g:

```
import requests

s = requests.session()
login_data={"account":"qianchencpp","password":"qianchen"}
res=s.post("http://mail.163.com/",login_data)
```

01/ Python requests

2. SSL证书验证

Requests可以为HTTPS请求验证SSL证书，就像web浏览器一样。要想检查某个主机的SSL证书，你可以使用 verify 参数:

```
>>> requests.get('https://github.com', verify=True)
<Response [200]>
```

如果你将 verify 设置为False，Requests也能忽略对SSL证书的验证。

```
>>> requests.get('https://kennethreitz.com', verify=False)
<Response [200]>
```

默认情况下， verify 是设置为True的。
你也可以指定一个本地证书用作客户端证书，可以是单个文件（包含密钥和证书）或一个包含两个文件路径的元组:

```
>>> requests.get('https://kennethreitz.com', cert=('/path/server.crt', '/path/key'))
<Response [200]>
```

01/ Python requests

3. Body content workflow

By default, when you make a request, the body of the response is downloaded immediately. You can delay downloading until you access the attribute of the Response.content with the stream parameter.

e.g:

```
r=requests.get(URL, stream=True)
#then only the response headers are downloaded and the connection with the
#server remain open. This can allow ue to attain the content which we want.
```

01/ Python requests

4. streaming uploads

```
with open(“”,”编码” ) as f:
    requests.post(“URL”，data=f)
```

5.use generator:

```
def gen():
    yield
    yield
requests.post(“URL”,data=gen())
```

01 Python requests

6. streaming requests

iter_lines
Iterates over the response data, one line at a time. This avoids reading the content at once into memory for large responses.

```
#简单地设置 stream 为 True 便可以使用 iter_lines() 对相应进行迭代:
import json
import requests

r = requests.get('URL', stream=True)

for line in r.iter_lines():

    # filter out keep-alive new lines
    if line:
        print(json.loads(line))
```

PART TWO
Parsing the HTML

01/ Python BeautifulSoup

Methods of parsing HTML：

1）BeautifullSoup (focus)

BeautifulSoup is a Python library for pulling data out of HTML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree.

2）Regular Expressions(focus)

3) Xpath(xlml库)

01/ Python BeautifulSoup

Installing Beautiful Soup

-- Anaconda not necessary to install it
-- 在cmd 命令行中输入 pip install bs4

The parser for the BeautifulSoup:

“html.parser“ 内置库，速度适中，但解析能力差
“lxml“ 极强的解析能力和快速的速度，缺点是C拓展
“html5lib” 解析能力较好，但速度满

01/ Python BeautifulSoup

Making the soup:

```
from bs4 import BeautifulSoup


soup = BeautifulSoup(object, "html.parser")
soup = BeautifulSoup(object, "lxml")
soup = BeautifulSoup(object, "html5lib")
```

01/ Python BeautifulSoup

Kinds of objects:

Beautiful Soup transforms a complex HTML document into a tree of Python objects.

Three major kinds of objects to be concerned:

---Tag: 标签

---NavigableString：遍历文档树（可搜索的字符）

---BeautifulSoup：BeautifulSoup 对象

01/ Python BeautifulSoup

Tag: 标签

直接利用soup.(tag)就可以获取到相应的标签的信息，但所获取到的是第一个符合相应标准的标签的内容

The most important features of a tag are its name and attributes.

01/ Python BeautifulSoup

Search the tree ： Navigable String

两种使用较广的method： find()　find_all()

Differences:

find method will return a tag object and if you want to get the text use getText() method.

find_all will return a list object.

find_all(name, attrs,*kwargs)

The find_all() method looks through a tag’s descendants and retrieves all descendants that match your filters.

01/ Python BeautifulSoup

Arguments for navigable string:

Name:

Pass in a value for name and you'll tell Beautiful Soup to only consider tags with certain names.

可以是li, div, span, a 等标签

Attrs (CSS)

It will be better if the parameter of attrs can take the form of dict.

Such as:

attrs={'class':''}

01/ Python BeautifulSoup

The keyword arguments

Any argument that's not recognized will be turned into a filter on one of a tag's attributes.
如果传入的参数是非指定的（不是name,attrs等）那么默认将其认为是keyword参数对每一个标签的该属性进行过滤。传入的参数可以是string也可以是正则表达式。

If you pass in a value for an argument called id, Beautiful Soup will filter against each tag's 'id' attribute:
soup.find_all(id='link2') #
[Lacie]

If you pass in a value for href, Beautiful Soup will filter against each tag's 'href' attribute:
soup.find_all(href=re.compile("elsie"))
[Elsie]

01/ Python BeautifulSoup

find_parents() and find_parent().

These methods take the same parameters as find and find_all.
They work their way up the tree, looking at a tag's (or a string's) parents.

find_next_siblings and find_next_sibling

Signature: find_next_siblings(name, attrs, **kwargs)
Signature: (name, attrs, **kwargs)
These methods use .next_siblings to iterate over the rest of an element's siblings in the tree. The find_next_siblings() method returns all the siblings that match, and find_next_sibling() only returns the first one:

01/ Python BeautifulSoup

find_all_previous() and find_previous()

这2个方法通过 .previous_elements 属性对当前节点前面 [5] 的tag和字符串进行迭代,

find_all_previous() 方法返回所有符合条件的节点, find_previous() 方法返回第一个符合条件的节点.

find_all_next and find_next

find_all_next(name , attrs , recursive , text , **kwargs)

find_next(name , attrs , recursive , text , **kwargs)

这2个方法通过 .next_elements 属性对当前tag的之后的 [5] tag和字符串进行迭代, find_all_next() 方法返回所有符合条件的节点, find_next() 方法返回第一个符合条件的节点:

01/ Python BeautifulSoup

CSS选择器

Beautiful Soup支持大部分的CSS选择器 [6] ,在 Tag 或 BeautifulSoup 对象的 .select() 方法中传入字符串参数,即可使用CSS选择器的语法找到tag:

对于熟悉CSS选择器语法的人来说这是个非常方便的方法.Beautiful Soup也支持CSS选择器API,如果你仅仅需要CSS选择器的功能,那么直接使用 lxml 也可以,而且速度更快,支持更多的CSS选择器语法,但Beautiful Soup整合了CSS选择器的语法和自身方便使用API.

03

PART THREE

Examples

03

PART THREE

Something about Fiddler

01/ Fiddler

Take JD app as an example:

Request message:
POST /client.action?functionId=iosSearchHotWords HTTP/1.1
Host: portal.m.jd.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Connection: close
Cookie: pin=;wskey=;whwswswws=;
USER_FLAG_CHECK=6195b12e024456a028c3fe04e93e7982;
abtest=20160515105808347_62; mba_muid=1463279222635-
3de7a6943208e2fe62.2.1463451441442; mobilev=html5
User-Agent: 京东 43591 rv:5.0.1 (iPhone; iPhone OS 9.2.1; en_US)
Content-Length: 366
Accept-Encoding: gzip
Connection: close

01/ Fiddler

```
HTTP/1.1 200 OK
Server: JDWS/1.0.0
Date: Thu, 19 May 2016 02:07:21 GMT
Content-Type: text/plain;charset=utf-8
Transfer-Encoding: chunked
Connection: close
Vary: Accept-Encoding
Cache-Control: max-age=0
Expires: Thu, 19 May 2016 02:07:21 GMT
Content-Encoding: gzip

400a
□□      □l YW[i - W□ Puk :g w }  <□ 732 f      ɓu□ &
 □; N      □□□H □=Ht□5H  3 □= _   \{
 1b8h     o5s 5     □=        7/
```

01/ Fiddler

Something about simulation on zhihu:

```
PUT /captcha HTTP/1.1
Host: api.zhihu.com
Accept: */*
Authorization: oauth 5774b305d2ae4469a2c9258956ea49
Proxy-Connection: keep-alive
x-app-za:
OS=iOS&Release=9.2.1&Model=iPhone7,1&VersionName=3.13.0&VersionCode=448&
Width=1242&Height=2208
Accept-Language: en-us
Accept-Encoding: gzip, deflate
X-API-Version: 3.0.19
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: 19
User-Agent: osee2unifiedRelease/448 CFNetwork/758.2.8 Darwin/15.0.0
Connection: keep-alive
X-APP-Build: release
X-APP-VERSION: 3.13.0
```

01/ Fiddler

Cookie:
capsion_ticket="2|1:0|10:1463627199|14:capsion_ticket|44:ZWViMTY5MDg4NDhhNDRkZmE4YmY0OTE5Y2VjNWNmZjQ=|dd34d19f54cfadb32797edf4ed5bcc007d96f39a864425f07c9710beb78515fa";
_ga=GA1.2.543230649.1453941807;
cap_id="NGQwNjVmZDhiOGMzNGQ2MTk1YjY1YzFiODY5YmI2ZmI=|1463572075|79e0db2ce23006145c0c495d7ef6f0dc17a0857b";
l_cap_id="ZGJhOTllODkzMTU0NGY2ZjhiNmM2YmM3ZDNlZTA3MTA=|1463572075|5ba7865a58f3b656c9612e5915abfbdbcf8d7a39";
q_c1=298440e25bd042a0a8b8968516bf486d|1463572075000|1460779893000

01/ Fiddler

Response message:

HTTP/1.1 202 Accepted
Server: nnws/1.7.3.7
Date: Thu, 19 May 2016 03:10:17 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 3127
Connection: keep-alive
Content-Encoding: gzip
X-Za-Response-Id: 0b0da1902eed441db30b359f29fe1e49
Vary: Accept-Encoding

01/ Fiddler


0:11
11/19
Find... (press Ctrl+Enter to highlight all)
View in Notepad
Get SyntaxView
Transformer
Headers
TextView
ImageView
HexView
WebView
Auth
Caching
Cookies
Raw
JSON
XML
HTTP/1.1 202 Accepted
Server: nnws/1.7.3.7
Date: Thu, 19 May 2016 03:10:17 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 4616
Connection: keep-alive
X-Za-Response-Id: 0b0da1902eed441db30b359f29fe1e49
Vary: Accept-Encoding
{"img_base64":"R0lGODdh8AA8AIQAAP7+\/gMDAxYWFujo6EdHRycnJ9fX13d3d8fHx7e3t2ZmZoeHh1hYWDk

01/ Fiddler

Another way to capture captcha:

To see the unique part of the URL:
If you change the captcha on the web or on APP, the part which keeps the same can’t be the part which we need. The part keeps changing is exactly what we need. And then use the explorer of the fiddler to browse the URL, and put the data on the requests message,

THANKS