python 爬虫基础学习

1.简介

1.1 学习目标

- 理解网络爬虫基础知识，会使用python的一些标准库urllib/urllib2/requests实现简单的爬虫应用
- 掌握爬虫程序的结构和设计原则
- 掌握爬虫程序的调试工具和技巧

1.2 主要内容

- http协议介绍
- pyton标准库里对Http的实现及其用法
- 正则表达式，用来对爬下的内容进行初步分析，获取我们想要的数据
- 多线程用来提高爬虫的执行效率，分布式爬虫简介
- 实例：文本数据，图片数据，AJAX数据

2.Http协议

2.1Http报文展示

使用谷歌浏览器的开发者模式查看报文

- http请求报文

```
GET / HTTP/1.1
Host: blog.kamidox.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/55.0.2883.75 Safari/537.36
Accept：
text/html,application/xhtml+xml,application/xml;q=0.9
,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: zh-CN,zh;q=0
```

- **请求响应**

```
HTTP/1.1 200 OK
Server: GitHub.com
Content-Type: text/html; charset=utf-8
Last-Modified: Sun, 14 May 2017 13:29:24 GMT
Access-Control-Allow-Origin: *
Expires: Fri, 05 Jan 2018 13:42:14 GMT
Cache-Control: max-age=600
Content-Encoding: gzip
X-GitHub-Request-Id:
A2A4:1F36C:637EC62:696546D:5A4F7E5D
Content-Length: 4851
Accept-Ranges: bytes
Date: Fri, 05 Jan 2018 13:32:14 GMT
Via: 1.1 varnish
Age: 0
Connection: keep-alive
X-Served-By: cache-bur17526-BUR
X-Cache: MISS
X-Cache-Hits: 0
X-Timer: S1515159135.829690,VS0,VE32
```

```
Vary: Accept-Encoding
X-Fastly-Request-ID:
c25127f4e16f5839951280901b5b8e66d861de3f
```

2.1.1请求及构成

1. GET / HTTP/1.1
   方法 相对路径 协议
2. Host：主机
3. Accept：可接受的媒体类型
4. User-Agent：浏览器身份
5. Accept-Encoding：编码类型
6. Accept-Language：介绍的语言

2.1.2应答及其构成

1. 应答码

- 2xx成功
  - 200:ok
  - 206:Partial Content
- 3xx:重定向
  - 301 Moved Permanently
  - 303 See Other
  - 304 Not modified
  - 307 Internal Redirect
- 4xx：客户端错误
  - 404：not found
- 5xx：服务端错误
  - Internal Server Error
  - Not Implemented、

2. Server：应答服务器
3. Content-type：应答的数据类型

- text/*
- image/*
- audio/*
- video/*
- /

4. Last-Modified：上一次修改时间
5. Content-Encoding：应答编码类型
6. Content-Length:应答的内容长度

2.3.url简介

2.3.1 组成部分

- 协议
- 路径
- 参数

2.3.2 url的编码方法

除英文字母，数组和部分符号，其他全部使用百分号+十六进制码值进行编码。

- **情况1：网址路径中包含汉字**
  例如浏览器输入”http://zh.wikipedia.org/wiki/春节“，HTTP请求的头信息，会发现IE实际查询的网址
  是"http://zh.wikipedia.org/wiki/%E6%98%A5%E8%8A%82"。
  结论1：网址路径的编码，用的是utf-8编码。
- **情况2：查询字符串包含汉字**
  例："http://www.baidu.com/s?wd=春节"。
  结论2：查询字符串的编码，用的是操作系统的默认编码。
- **情况3：Get方法生成的URL包含汉字**
  <meta http-equiv="Content-Type"
  content="text/html;charset=xxxx">
  如果上面这一行最后的charset是UTF-8，则URL就以UTF-8编码；如果是GB2312，URL就以GB2312编码。

结论3:GET和POST方法的编码，用的是网页的编码。

- **情况4：Ajax调用的URL包含汉字**
  举例来说，有这样两行代码：

```
url = url + "?q=" +document.myform.elements[0].value;
// 假定用户在表单中提交的值是"春节"这两个字
http_request.open('GET', url, true);
```

结论4：在Ajax调用中，IE总是采用GB2312编码（操作系统的默认编码），而Firefox总是采用utf-8编码。

2.3.3javascript的编码函数

- escape()
  escape()不能直接用于URL编码，它的真正作用是返回一个字符的Unicode编码值。它的具体规则是，除了ASCII字母、数字、标点符号"@ * _ + - . /"以外，对其他所有字符进行编码。在\u0000到\u00ff之间的符号被转成%xx的形式，其余符号被转成%uxxxx的形式。对应的解码函数是unescape()。
  注意：，escape()不对"+"编码。但是我们知道，网页在提交表单的时候，如果有空格，则会被转化为+字符。服务器处理数据的时候，会把+号处理成空格。所以，使用的时候要小心。
- encodeURI()
  对整个URL进行编码，因此除了常见的符号以外，对其他一些在网址中有特殊含义的符号"; / ? : @ & = + $, #"，也不进行编码。编码后，它输出符号的utf-8形式，并且在每个字节前加上%。对应的解码函数是decodeURI()。
  注意：它不对单引号'编码。
- encodeURIComponent()
  它用于对URL的组成部分进行个别编码，而不用于对整个URL进行编码。因此，"; / ? : @ & = + $, #"，这些在encodeURI()中不被编码的符号，在encodeURIComponent()中统统会被编码。至于具体的编码方法，两者是一样。对应的解码函数是decodeURIComponent()。

2.4.Cookie介绍

Cookie是服务器在客户端保存的信息。
请求时，客户端需要把未超时的Cookie发送回给服务器端

```
Cookie: bid=”kmlFWje+MYs”; ll=”118201”
```

应答时，服务器会把新的cookies发给客户端,以便下次请求时带上这些cookies.

```
Set-Cookie: bid=”M/Q1d6PwmB4”; path=/;
domain=.douban.com; expires=Thu, 08-Dec-
2016 15:34:09 GMT
Set-Cookie: ll=”118201”; path=/; domain=.douban.com;
expires=Thu, 08-Dec-2016
15:34:09 GMT
```

2.4.1Cookie格式

客户端发送Cookie： Cookie: key1=value1; key2=value2; key3=value3
服务器端保存Cookie： Set-Cookie: key1=value1; path=/; domain=xx

2.4.2Cookie的属性

- Domain and Path：定义Cookie的作用域。当指定domains时，这个domain及其子域名都会包含这个Cookie
- Expires:定义Cookie的生命周期
- HttpOnly:禁止脚本访问

3.python里的网络组件

3.1 urllib的介绍

3.1.1urlib.urlopen(url[,data[,proxies]])

- url: 请求的url
- data：如果有，则变成POST方法，数据格式必须是application/x-www-form-urlencoded
- 返回类文件句柄
    - 类文件句柄的常用方法
        - read(size)
        - readline()
        - readlines()
        - close()
        - getcode()

3.1.2 HTTPMessage的方法

- urlib.urlopen().info():得到 httplib.HTTPMessage实例
- httplib.HTTPMessage
    - headers
    - gettype()
    - getheader() / getheaders()
    - items() / keys() / values()：解析后的头部信息
- dir(oject):获取对象的方法

3.1.3 urllib.urlretrieve（url[,filename[,reporthook[,data]]]）

- url：远程地址
- filenme：要保存的文件
- reporthook：下载状态报告
    - 参数1：当前传输的块数
    - 参数2：块大小
    - 参数3：数据总大小
    - 需要注意：content-length不是必须的
- data： POST 的application/x-www-form-urlencoded 格式数据
- 返回值： filename, HTTPMessage)

3.1.4 工具函数

- urllib.urlencode（params）
    - 把字典数据转换为URL编码(默认utf-8)
    - 用途
        - 对url参数进行编码
        - 对post上的去form数据进行编码
- urlparse.parse_qs（param）
    - 把URL编码转换为字典数据
- urlparse.urlparse（url）
    - 解析url。可以得到协议，域名，参数等
- 其他
    - quote
    - unquote
    - pathname2url
    - url2pathname

3.2urllib2的介绍

3.2.1urllib与urllib2的区别

urllib2提供了比urllib更丰富的功能

- urllib2.Request - 提供http header定制功能
- urllib2提供更强大的功能，例：cookie处理。鉴权，可定制化等。
- urllib2能不能完全代替urllib？ 不能。 urllib2没与urllib.urlencode对应的方法。

3.2.2urllib2.urlopen()

- 参数
    - url
    - data
    - timeout（比urllib多的参数）
- 错误处理 HTTPError, e（try ...catch处理）

3.2.3 urllib2.Request（）

设置url请求的headers

- 参数
    - url
    - data - optional
    - headers - 字典
- 使用Request添加或修改http头
    - Accept: application/json
    - Content-Type: application/json
    - User-Agent: Chorme

3.2.4 urllib2.build_opener

定制http的行为

- BaseHandler及其子类
    - HTTPHandler
    - HTTPSHandler
    - HTTPCookieProcessor
- build_opener
    - 参数化Handler列表
    - 返回OpenerDirector对象
- 默认会创建的Handler链
    - ProxyHandler（如果设置了代理）
    - UnknownHandler
    - HTTPHandler
    - HTTPDefaultErrorHandler
    - HTTPRedirectHandler
    - FTPHandle
    - FileHandler
    - HTTPErrorProcessor
    - HTTPSHandler (如果安装了ssl模块)
- 例：request_post_debug()
    - 打印http调试的信息
- 保存opener为默认

```
opener =
urllib2.build_opener(urllib2.HTTPHandler(debuglevel=1
),

urllib2.HTTPSHandler(debuglevel=1))
urllib2.install_opener(opener)
```

3.2.5 cookies处理

- cookielib.CookieJar
    - 提供解析并保存Cookie的接口
- HTTPCookieProcessor
    - 提供自动处理cookie的功能

3.2.6 爬虫实例：豆瓣热播电影

- 热播电影数据格式
  输入网址https://movie.douban.com/cinema/nowplaying/xiamen/，开发者模式查看
- HTMLParser简介 解析html的数据
    - feed：向解析器喂数据，可以分段提供
    - handle_starttag（seld,tag,attrs）：处理html的开始标签
        - tag:标签名称
        - attrs:属性列表
    - handle_data(self,data)：处理标签里的数据体
        - data：数据文本

3.3 requests介绍

reuqest的官方文档的地址：http://requests.readthedocs.io/en/latest/。

3.3.1 requests简介

- 和urllib/urllib2的区别：

    - requests不是标准库
    - 最好用的http库，pythonic风格
- 安装：pip install requests

3.3.2 请求

- requests.request
    - method：get/post/head/put/delete
    - url：请求地址
    - params：请求参数
    - data：字典数据
    - json：上传的json数据
    - headers：自定义htpp头
    - cookies：发送额外的cookies
    - verify：是否检验证书
- requests.get
    - url
    - data
    - json
    - 和request参数一样(除了method)
- requests.head
- requests.put
- requests.delete

3.3.3 应答

- requests.Response
    - status_code 状态码
    - headers 应答的http头
    - json 应答的json数据
    - text 应答的unicode编码的文本
    - content 应答的字节流数据
    - cookies 应答的cookies，自动处理

3.4 使用request库的爬虫实例

- 豆瓣热播电影（request重构）
    - url地址：https://movie.douban.com/cinema/nowplaying/wuxi/
    - 爬取的内容：名称，导演，评分，演员，封面
- 爬取豆瓣音乐新碟榜单曲
    - url地址：https://music.douban.com/
    - 爬取内容：歌曲名，歌手，评分，将单曲封面下载
- 登录豆瓣并修改签名
    - 登录流程分析
        - 向哪个url发送请求？
        - 发送哪些数据？
        - 有哪些特殊的头字段？
        - 验证码问题如何解决？
    - 登录使用的技术
        - 使用requests.session来处理cookies
        - 模拟浏览器的登录行为
    - 修改签名流程分析
        - 向哪个url发送请求？
        - 发送哪些数据？
        - 有哪些特殊的头字段？
        - 返回值长什么样？
- 登录知乎并修改个人简介

4.正则表达式

4.1 认识正则表达式

- python里的正则表达式re
    - pattern：匹配模式，遵循正则表达式语法
    - method：匹配方法，search/match/split/findall/finditer/sub/subn

4.1.1re模块介绍

- re.search:搜索字符串，找到匹配的第一个字符串
- re.match:长字符串开始匹配
- search vs match
    - search：搜索字符串，任意位置的匹配
    - match：只能从字符串的起始位置开始匹配
- split:使用正则表达式来分割字符串
- findall:根据正则表达式从左到右搜索匹配项，返回匹配的字符串列表
- finditer:根据正则表达式从左到右搜索匹配项，返回一个迭代器（MatchObject对象）
- sub:字符串替换
    - pattern:正则表达式
    - repl:替换项（字符串或函数）
    - string:待处理字符串
- subn与sub一样，返回值多了替换的字符串个数

4.1.2MatchObject

能匹配带正则表达式时返回re.MatchObject

- group():返回匹配的组
    - 索引0表示全部匹配的字符串
    - 索引1表示全部匹配的子组
    - 参数可以是一个也可以多个
    - 命名组
- groupdict():返回命名组的字典
- groups():返回匹配的子组，索引从1开始的所有子组
- start/end/span:返回匹配的位置

4.2正则表达式语法

- 通配符
- 特殊通配符
- RegexObject
    - re.complie()返回的结果
        - search

- match
- findall
- split
- finditer
- sub

4.3爬虫实例：唐诗三百首