Python 爬虫基础

1. 课程介绍

1.1 课程介绍

- 什么是爬虫?
  - 一个自动从网络获取数据的程序
- 爬虫能干什么?
  - 新闻数据：今日头条
  - 机器学习：股票数据获取及分析
  - 网络搜索引擎的一个部件

培训对象

具有一定的 python 基础知识，想学习 python 在网络爬虫方面的基础知识的同学。如果没有 python 基础知识，有其他语言如 java 等经验的也同样适用本课程。

培训目标

- 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

知识准备

前提：了解基本的 Python 语言知识

推荐：
http://learnxinyminutes.com/docs/python/ (http://learnxinyminutes.com/docs/python/)

学习编程语言的方法

如何获得帮助：
1. 搜索引擎
2. 官方文档
3. 向人求助

授课宗旨：授人以渔
1. 方法比知识重要
2. 传道比授业重要

课程内容

- Http 协议介绍
- Python 标准库里对 Http 的实现及其用法
- 正则表达式，用来对爬下来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
- 课程总结：回顾与展望
- 课程实例：文本数据，图片数据，AJAX 数据

2. Http 协议介绍

2.1 认识 http

http 报文展示

Http 报文

使用 chrome 开发者工具查看 blog.kamidox.com 的 http 请求

http 请求及其构成

Http 请求报文介绍

1. 方法：GET/POST
2. URI: 相对路径
3. Host: 目标主机
4. Accept: 可接受的媒体类型
5. User-Agent: 浏览器身份
6. Accept-Encoding: 编码类型
7. Accept-Language：接受的语言

http 应答及其构成

Http 应答报文介绍

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
     4xx: 客户端错误
   - 400 Bad Request
   - 404 Not Found
     5xx: 服务端错误
   - 500 Internal Server Error
   - 501 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

更多请求和应答

1. 下载 css 的请求
   - If-Modified-Since
   - 304 Not Modified
2. 下载图片的请求
   - ContentType: image/*

URL简介

url 组成部分

- 组成部分：schema://path?query
  - 协议
  - 路径
  - 参数
- 举例
  - http://blog.kamidox.com (http://blog.kamidox.com)
  - https://www.baidu.com (https://www.baidu.com)
  - http://cn.bing.com/?t=1&b=2 (http://cn.bing.com/?t=1&b=2)

url 编码规则

- 编码规则
  - 除英文字母，数字和部分符号外，其他的全部使用百分号+十六进制码值进行编码
  - 例子：百度搜索中文

url 参数规则

- 参数规则
  - 参数以问号作为开始
  - 参数对是 key=value 样式
  - 参数对之间使用 & 号连接
- 例子
  - 百度搜索中文的URL分析

作业

- 扩展阅读
  - 阮一峰的网络日志 关于URL编码
    http://www.ruanyifeng.com/blog/2010/02/url_encoding.html
    (http://www.ruanyifeng.com/blog/2010/02/url_encoding.html)
- 使用搜索引擎学习 http 应答码

2.2 Cookie 介绍

Cookies 基础

Cookie 数据长什么样

清除浏览器历史数据
登录 www.douban.com (http://www.douban.com) 查看 cookies 数据

Cookie 的格式

客户端发送 Cookie 时：

Cookie: key1=value1; key2=value2; key3=value3

服务器端保存 Cookie 时：

Set-Cookie: key1=value1; path=/; domain=xx

Cookie 属性

Domain and Path: 定义 cookie 的作用域。当指定 domain 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: 禁用脚本访问

Cookie 的用途

1. 登录信息：判断用户是否已经登录
2. 购物车：保存用户购买的商品列表

Cookie 小结

服务器在客户端存储的信息。

请求时，客户端需要把未超时的 cookies 发送回给客户端。

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

从登录行为看 Cookie

以douban为例来模拟用户修改签名的操作

作业

通过搜索引擎学习一下 cookie 可能会引起什么样的安全问题。

3. Python 标准库里的网络组件

3.1 urllib 介绍

urllib 基础知识

对 http 协议的最简单的实现

urllib.urlopen

 # urllib.urlopen

- urllib.urlopen
    - url: scheme(http: / file:)
    - 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): size = -1 / None
    - readline()
    - readlines()
    - close()
    - getcode()

探求 HTTPMessage 的方法

 # 方法

- HTTPMessage 没有官方文档，如何找出其有用的方法?
- info(): 返回 httplib.HTTPMessage 实例
- httplib.HTTPMessage

    - headers
    - gettype()
    - getheader() / getheaders()
    - items() / keys() / values()

urllib.urlretrieve

urllib.urlretrieve

- urllib.urlretrieve
    - url: 远程地址
    - filename: 要保存到本地的文件
    - reporthook: 下载状态报告
    - data: POST 的 application/x-www-form-urlencoded 格式的数据
    - 返回 (filename, HTTPMessage)
- reporthook:
    - 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 不是必需的

工具函数

- urllib.urlencode
    - 把字典数据转换为 URL 编码
    - 用途
        - 对 url 参数进行编码
        - 对 post 上去的 form 数据进行编码
- urlparse.parse_qs
    - 把 URL 编码转换为字典数据
- 其它
    - quote
    - unquote
    - pathname2url
    - url2pathname
- 例子

    - 百度搜索中文为例分析URL构成

一个简单的爬虫的实例：从雅虎财经获取股票数据

雅虎财经股票数据接口介绍

- 股票数据
    - 深市数据链接：http://table.finance.yahoo.com/table.csv?s=000001.sz (http://table.finance.yahoo.com/table.csv?s=000001.sz)
    - 上市数据链接：http://table.finance.yahoo.com/table.csv?s=600000.ss (http://table.finance.yahoo.com/table.csv?s=600000.ss)
- 时间参数
    - a, b, c, d, e, f, s
    - 示例：取 2012年1月1日 至 2012年4月19日的数据 http://table.finance.yahoo.com/table.csv?a=0&b=1&c=2012&d=3&e=19&f=2012&s=600690.ss (http://table.finance.yahoo.com/table.csv?a=0&b=1&c=2012&d=3&e=19&f=2012&s=600690.ss)

作业

1. 安装 ipython
2. 使用 ipython notebook 熟悉课程介绍的 API
3. pip install jupyter
4. 从雅虎财经网站获取上证 600100到 600109十个股票最近一个月的日交易数据

3.2 urllib2 介绍

urllib 和 urllib2 的区别

urllib2 提供了比 urllib 更丰富的功能：

- urllib2.Request - 提供 http header 定制能力
- 提供更强大的功能，包括 cookie 处理，鉴权，可定制化等。

urllib2 能不能完全替代 urllib?

- urllib.urlencode

urllib2.urlopen

- urlopen()
  - url
  - data
  - timeout
- 错误处理 HTTPError, e
- 示例：urlopen

urllib2.Request

- urllib2.Request()
  - 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
- 示例：request

urllib2.build_opener

- BaseHandler 及其子类
  - HTTPHandler
  - HTTPSHandler
  - HTTPCookieProcessor
- build_opener
  - 参数 Handler 列表
  - 返回 OpenerDirector
- 默认会创建的 Handler 链
  - ProxyHandler (如果设置了代理)
  - UnknownHandler
  - HTTPHandler
  - HTTPDefaultErrorHandler
  - HTTPRedirectHandler
  - FTPHandler

    - FileHandler
    - HTTPErrorProcessor
    - HTTPSHandler （如果安装了 ssl 模块）

- 示例：request_post_debug
    - 打印 http 调试信息
    - POST 数据
- 保存 opener 为默认
    - urllib2.install_opener
    - 示例：install_debug_opener

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
- 演示访问 douban 的 cookie 传输过程
- 示例：handle_cookies
- 问题思考：用 cookie 来模拟登录?
    - 验证码问题?

爬虫实例：豆瓣热播电影

热播电影数据格式

使用 Chrome 的开发者工具查看豆瓣热播电影的数据格式

http://movie.douban.com/nowplaying/xiamen/ (http://movie.douban.com/nowplaying/xiamen/)

HTMLParser简介

- feed: 向解析器喂数据，可以分段提供
- handle_starttag: 处理 html 的开始标签
    - tag: 标签名称
    - attrs: 属性列表

- handle_data：处理标签里的数据体
  - data: 数据文本

爬取数据并解析

作业：从豆瓣电影抓取即将上映的电影

抓取的信息要求：

- 电影名称
- 电影时长
- 上映时间
- 导演
- 演员信息

3.3 requests 介绍

requests 基础知识

requests 简介

Http for humans

- 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

请求

- requests.request
  - method: get/post/head/put/delete
  - url
  - params: 请求的参数
  - data: 字典，字节流，或类文件句柄
  - json: 上传的 json 数据
  - headers: 自定义 http 头
  - cookies: 发送额外的 cookies
  - verify: 是否检验证书
- requests.get
  - url

    - 和 request 的参数一样
- requests.post
    - url
    - data
    - json
    - 和 request 的参数一样
- requests.head
- requests.put
- requests.delete

for REST API

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。

高级用法

- Session: 同一个会话内参数保持一致，且会重用 TCP 连接以提高性能。也会尽量保持连接也提高性能。
- SSL 证书认证：开启，关闭，自定义 CA 证书
- 上传普通文件和复杂结构的文件
- 代理访问

作业

阅读 requests 库的官方文档
1. 了解基本用法
2. 了解高级用法

http://requests.readthedocs.org/en/latest/ (http://requests.readthedocs.org/en/latest/)

爬虫实例：豆瓣热播电影

- 用 requests 重新实现豆瓣热播电影

- 增加功能：下载每个电影的海报图片

热播电影数据格式

使用 Chrome 的开发者工具查看豆瓣热播电影的数据格式

http://movie.douban.com/nowplaying/xiamen/
(http://movie.douban.com/nowplaying/xiamen/)

使用 requests 重构代码

增加图片下载功能

作业

豆瓣音乐 http://music.douban.com/ (http://music.douban.com/)
实现一个爬虫，获取 新碟榜 单曲

要求：
歌曲名称
歌手名字
豆瓣评分
解析单曲封面图片 url 并把图片下载下来

爬虫实例：登录豆瓣

功能

- 登录豆瓣
- 修改签名

登录流程分析

1. 向哪个 url 发送请求
2. 发送哪些数据?
3. 有哪些特殊的头字段?
4. 验证码问题如何解决?

登录使用的技术

1. 使用 requests.Session来处理cookies
2. 模拟浏览器的登录行为

修改签名流程分析

1. 向哪个 url 发送请求
2. 发送哪些数据?
3. 有哪些特殊的头字段?
4. 返回值长什么样?

作业：登录知乎

1. 登录知乎
2. 修改个人简介

4. 正则表达式

4.1 认识正则表达式

程序员分两种

- 懂正则表达式
- 不懂正则表达式

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

一个例子：提取价格

re 模块介绍

- re.search: 搜索字符串，找到匹配的第一个字符串
- re.match: 从字符串开始开始匹配
- search vs. match
  - search: 搜索字符串任意位置的匹配
  - 只从字符串的起始位置开始匹配
- split: 使用正则表达式来分隔字符串
- 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，返回一个迭代器迭代返回 MatchObject
- sub 字符串替换
  - pattern 正则表达式
  - repl 替换项，字符串或函数

    - string 待处理的字符串
- subn 与 sub 一样，返回值多了替换的字符串个数

MatchObject

能匹配到正则表达式时返回 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 开始表示匹配的子组
    - 参数可以一个也可以多个
    - 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

- 什么是正则表达式：一个字符串
- 最简单的正则表达式：'test'
- 特殊字符：7.2.1. Regular Expression Syntax

通配符

特殊通配符

RegexObject

re.compile() 返回的结果

- search
- match
- findall
- split
- finditer
- sub

可以大大地提高效率

作业：正则表达式练习

识别电话号码

- 11位数字：13774347721
- 带连字符：137-7434-7721
- 带中国国家码: +8613774347721
- 带中国国家码及连字符：+86137-7434-7721

有多种方案，想一想有没有更简洁的方案。

阅读正则表达式的官方标准文档

4.3 爬虫实例：唐诗三百首

唐诗三百首

http://www.gushiwen.org/gushi/tangshi.aspx (http://www.gushiwen.org/gushi/tangshi.aspx)

- 爬取诗词标题
- 爬取诗词作者
- 爬取诗词的网页地址

数据分析

数据抓取

匹配结果

作业：宋词精选

宋词精选

http://www.gushiwen.org/gushi/songci.aspx (http://www.gushiwen.org/gushi/songci.aspx)

- 词牌名
- 标题
- 作者

4.4 爬虫实例：唐诗三百首完整内容

数据分析

数据抓取

异常处理

作业：宋词精选内容

5. 多线程

5.1 认识线程

什么是线程

1. 并发执行的单元
2. 从一个实例理解线程
3. 线程带来的并发性

原子操作

1. 什么是原子操作
2. 为什么需要原子操作
3. 怎么样保证原子操作

一个并发的例子：
多线程执行
i = i + 1

操作分解：
1. CPU 读取 i 的值
2. CPU 执行 + 1 操作
3. CPU 把结果写入 i 变量所在内存

线程同步

管理关键资源的机制

例子：金库管理

- 采购需要进去拿钱去买东西
- 销售会把卖东西得来的钱放到金库里

- 进门需要带一把很贵的纯金打造的感应钥匙

1. 锁
    - aquire：上锁，获得金库钥匙
    - release：解锁，把钥匙放回
    - threading.Lock：钥匙带在进门的人身上，任何人要进来必须等里面的人出来才可以。
    - threading.RLock：钥匙放在部门经理那，同一个部门的人可以一起进来。
2. 信号量
    - threading.Semaphore：多配了几把钥匙，每个钥匙都带在进去的人身上
    - init_value: 有多少把钥匙
3. 条件
    - threading.Condition：金库里的钱花光了。采购拿到钥匙后，也拿不到钱。这个时候可以使用条件来实现。当销售部门卖完产品拿到钱后，会把钱放回金库，再通知采购去取钱。
    - wait, notify, notify_all
    - 调用 wait 前需要获得资源锁
4. 事件
    - 和条件类似。不同的是不能象条件那样只通知一个人。条件可以通过 notify 通知最早排队的人。对金库的例子，当销售把钱放进金库时，如果使用事件机制，那么所有排除的人都知道金库有钱了，他们都可以进去取。
    - 另外一个不同点是事件没有锁机制，只是单纯在等待事件的发生。而条件是有锁机制的。

锁

信号量

条件

事件

线程的简单例子

5.2 爬虫实例：从百度图片下载壁纸

数据分析

http://image.baidu.com/channel/wallpaper (http://image.baidu.com/channel/wallpaper)

爬虫实现

多线程爬虫

多线程爬虫的问题

Python 多线程的限制条件：Python 解析器的全局锁导致即使在多核CPU，一个时间片内也只能有一个Python程序在执行

- 多线程只适用于有IO等待的场景。如果是纯计算的场景，多线程无法优化性能
- 使用多进程 multiprocessing
- 使用分布式
- 关于并发有并发控制，关注 twisted/gevent 等基于事件的框架

6. 课程小结

课程回顾

- Http 协议
  - Http 请求和应答
  - Http 状态码
  - 常用的 http 头
  - Http 请求参数
  - Cookie 的格式，作用
- Python 对 Http 的实现
  - urllib
  - urllib2
  - requests
  - 实例：从雅虎财经获取股票数据
  - 实例：豆瓣热播电影
  - 实例：登录豆瓣并修改签名
- 正则表达式
  - 正则表达式的基础知识
  - 正则表达式的语法
  - 正则表达式的作用
  - 实例：唐诗三百首爬虫
- 多线程

    - 线程的概念
    - 线程同步机制
    - 消费者生产者模型
    - 实例：从百度图片下载壁纸

课程展望

- scrapy 爬虫框架
- beautifulsoup 解析器
- Selector/Xpath -> scrapy
- 并发
    - twisted
    - gevent
- 分布式爬虫
    - 任务队列：https://github.com/nvie/rq (https://github.com/nvie/rq)
    - 任务队列与存储结合：https://github.com/rolando/scrapy-redis (https://github.com/rolando/scrapy-redis)
    - 数据处理：https://github.com/grangier/python-goose (https://github.com/grangier/python-goose)

!!!最最重要的!!!
Talking is cheap, show me the code!

-> 老板

Talking is cheap, show me the result!

开始实现你的爬虫吧。遇到问题后再来学习新的机制和解决方案。这些优秀的开源库不是无端发明出来的。都是大量先行者遇到问题后，想出解决方案，最终沉淀下来的精化。

不知道用爬虫来做什么?
知乎搜索一下：何明科
https://www.zhihu.com/people/he-ming-ke (https://www.zhihu.com/people/he-ming-ke)
刷一遍他的高票回答，你就知道原来用爬虫可以做这么酷的事情，顺便还把钱赚了