FETCH代码简要说明

[**FETCH代码简要说明** 1](#_Toc431215400)

[1 代码执行方式 1](#_Toc431215401)

[1.1 执行抓取任务 1](#_Toc431215402)

[1.2 查看抓取状态 1](#_Toc431215403)

[1.3 任务文件格式定义 1](#_Toc431215404)

[1.4 增加账号 1](#_Toc431215405)

[2 代码目录结构 2](#_Toc431215406)

[2.1 根目录——对应于代码之中的root\_path 2](#_Toc431215407)

[2.2 fetchclass目录——存放抓取功能类的目录 2](#_Toc431215408)

[3 主要代码说明 3](#_Toc431215409)

[3.1 入口文件main.py 3](#_Toc431215410)

[3.2 共有函数文件fetclass/common.py 3](#_Toc431215411)

[3.3 数据库操作文件fetclass/dbctrl.py 4](#_Toc431215412)

[3.4 基类文件fetclass/BaseFetch.py 5](#_Toc431215413)

[3.5 cjol抓取文件fetchclass/libcjol.py 6](#_Toc431215414)

[3.6 51job抓取文件fetchclass/libjob51.py 8](#_Toc431215415)

[4 数据库表格的结构 8](#_Toc431215416)

[4.1 总数记录表- total\_count 8](#_Toc431215417)

[4.2 Cjol分段抓取记录表-cjol\_seg\_count 8](#_Toc431215418)

[4.3 51job分段抓取记录表 9](#_Toc431215419)

[5 代码的二次开发流程 9](#_Toc431215420)

# 代码执行方式

## 执行抓取任务

通用命令模式为：python main.py moudle\_name cookie\_path task\_path

Moudle\_name:模块名称

Cookie\_path:cookie路径

Task\_path:任务路径

例如采用/data/cjol.txt作为cookie,执行/data/task001.txt作为任务路径，那么就可以执行：python main.py cjol /data/cjol.txt /data/task001.txt

## 查看抓取状态

采用Python viewstats.py 不带任何参数即可输出当前的数据源抓取情况

采用Python viewstats.py mail携带参数mail,即可输出当前的数据源抓取情况,并且发送邮件（邮件的发送操作依赖于配置文件）

## 任务文件格式定义

Cjol、51job目前的采集方式仍然是基于号段进行抓取，任务文件内容格式为：“开始值,结束值;执行进度值”，注意其中的标点均为英文。

例如，从5000-10000，当前执行到6000，那么该项任务的文件内容应该为：5000,10000;6000。

## 增加账号

增加账号进行抓取，进行抓取：

* + - 1. 首先需要在chrome插件之中进行配置账号快捷键，修改init.php文件，使得在浏览器登录之后可以在服务器特定位置创建cookie文件；
      2. 然后，只需增加cookie文件，而后创建一个对应任务文件执行即可。

# 代码目录结构

## 根目录——对应于代码之中的root\_path

conf：用于存放配置文件目录

cookie:用于存放cookie文件，下有各个数据源的分级目录，例如cookie/cjol

databse:数据库文件存放目录，存放sqllite数据库文件

db:用于存放简历的目录文件

error:用于存放出错信息目录，下有各个数据源的分级目录，例如error/51job

fetchclass:用于存放抓取功能类文件和通用函数文件

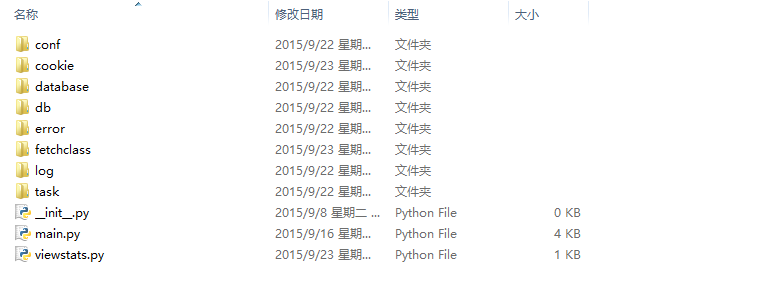
log：用于存放日志

task：用于存放任务文件，下有各个数据源的分级目录

\_\_init\_\_.py:模块化文件

main.py：程序入口文件

viewstats.py：抓取状态查看文件



## fetchclass目录——存放抓取功能类的目录



\_\_init\_\_.py:Python模块化文件

BaseFetch.py：抓取基类文件，所有的抓取功能类必须继承自改类

common.py：通用函数文件，主要包含通用的文件操作函数、url请求函数、邮件报警函数等

dbctrl.py：数据库控制文件，包含数据库操作的相关函数，例如数据的总数统计和分段统计信息记录

lib51.py：51job的抓取功能类，用于操作51job的简历抓取

libcjol.py：cjol的抓取功能类，用于操作cjol的简历抓取

log.py：日志初始化文件，用于初始化日志参数

# 主要代码说明

## 入口文件main.py

* + 1. 文件说明

该文件主要是程序运行入口，输入相对应参数之后，识别参数然后启动相应的工作流水线来执行任务。

该文件主要调用fetchclass目录下的各种抓取功能类，构建对象然后进行调用其中的runwork()函数来执行操作

## 共有函数文件fetclass/common.py

* + 1. 文件说明

该文件包含通用的、公共函数，主要有文件操作函数、邮件发送函数、url请求函数

* + 1. 主要函数说明
       1. URL请求相关函数

1. url\_post(url,postdict={},headers={},timeout=10)

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| 用于发送post请求，可以携带post参数并指定请求头信息 | **参数名** | **描述** |
| url | 所要请求的url链接 |  |
| postdict | 所需发送参数字典 |
| headers | 请求的header |
| timeout | 请求超时时间 |

1. url\_get(url,getdict={},headers={},timeout=10)

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 用于发送get请求，可以携带get参数并指定请求头信息 | url | 所要请求的url链接 |  |
| gettdict | 所需发送参数字典 |
| headers | 请求的header |
|  | Timeout | 请求超时时间 |

* + - 1. 文件操作相关函数

1. move\_file(src,dst)

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 安全方式移动文件，自动处理路径、重名问题 | src | 所需移动的源文件路径 |  |
| dst | 目的路径，必须指定为文件路径 |

1. clear\_dir(dir\_path='')

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 清空目录 | dir\_path | 所需清空的目录 |  |

1. write\_file(fpath,file\_str='')

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 安全方式写入文件，自动处理目录以及重名问题 | fpath | 所需写入的文件路径 |  |
| file\_str | 所需写入的文件内容 |

* + - 1. 邮件发送相关函数

1. sendEmail(moudle\_name='', title='', message='',msg\_type=0):

'''功能描述：通用发信模块，发送简要的email信息'''

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 通用发信模块，发送简要的email信息 | moudle\_name | 调用发信模块的名称 |  |
| title | 邮件主题 |
| message | 邮件内容 |
| msg\_type | 所发送邮件的类型，0：未知，1：报告 |

## 数据库操作文件fetclass/dbctrl.py

* + 1. 文件说明

该文件主要用于操作数据库，执行抓取记录的入库工作，对于数据库进行增删改查。

* + 1. 主要函数说明
       1. data\_total\_add(moudle\_name='')

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 记录每种数据源的抓取总数，每执行一次加1 | moudle\_name | 所对应抓取数据源的名称 |  |

* + - 1. data\_seg\_record(moudle\_name='',seg\_num = 0,addlist=[])

|  |  |  |  |
| --- | --- | --- | --- |
| **功能描述** | **参数** | | **备注** |
| **参数名** | **描述** |
| 记录每种数据源的分段抓取结果，目前的主要分段是按照ID进行分段，用于记录抓取数目、命中率百分比等 | moudle\_name | 所对应抓取数据源的名称 |  |
| seg\_num | 成功抓取的数目 |
| addlist | 其他可选参数，不同数据源内部内容不同，目前51、cjol均是采用依次为段开始/段结束/命中率百分比统一格式 |

## 基类文件fetclass/BaseFetch.py

* + 1. 文件说明

该文件主要用于定义抓取基类相关函数以及变量，所有的抓取功能类必须继承此类。

* + 1. 类说明

|  |  |  |  |
| --- | --- | --- | --- |
| 类名称 | | BaseFetch | |
| 属性 | | | |
| 属性名 | 属性说明 | | 备注 |
| root\_path | 项目根目录 | |  |
| db\_root | 简历存放根目录 | |  |
| db\_dir | 简历存放目录 | |  |
| cookie\_root | Cookie根目录 | |  |
| cookie\_dir | Cookie存放目录 | |  |
| task\_root | 任务根目录 | |  |
| task\_dir | 数据源的任务目录 | |  |
| task\_inuse\_dir | 正在执行任务的目录 | |  |
| task\_fin\_dir | 已完成任务的目录 | |  |
| error\_root | 出错根目录 | |  |
| error\_dir | 数据源出错目录 | |  |
| debug | 是否处于调试模式，暂时未使用 | |  |
| host | 主机host | |  |
| domain | 域名 | |  |
| module\_name | 数据源抓取模块名 | |  |
| ctmname | 企业名 | |  |
| username | 用户名 | |  |
| password | 密码 | |  |
| refer | 请求头部之中的refer信息 | |  |
| headers | 请求头部信息 | |  |
| cookie | Cookie字符串 | |  |
| cookie\_fpath | Cookie路径 | |  |
| cookie\_modtime | Cookie修改时间 | |  |
| login\_type | 登录类型，0:未知，1：用户名密码 2：登录类型 | |  |
| islogin | 是否已经登录 | |  |
| login\_wait | 登录尝试等待时间，默认300s | |  |
| login\_at | 登录时间 | |  |
| logout\_at | 登出时间 | |  |
| need\_login\_tags | 需要登录页面的标记列表 | |  |
| login\_success\_tag | 登录成功标记列表 | |  |
| resume\_tags | 简历识别标记列表，每个元素可唯一识别出简历 | |  |
| request\_try\_times | 单个请求的尝试的最大次数 | |  |
| task\_type | 所执行任务的类型 | |  |
| taskfpath | 所执行任务的存放路径 | |  |
| maxsleeptime | 随机休眠的最大秒数不同的数据源需要配置不同参数，尽量模拟人工操作避免被封号 | |  |
| last\_error\_at | 记录最近的出错时间以 | |  |
| recent\_error\_count | 时间内的出错次数 | |  |
| 函数 | | | |
| 函数名 | | 功能描述 | 备注 |
| init\_path(self) | | 必要的路径初始化 |  |
| load\_cookie(self,fpath) | | 从文件载入cookie字符串，并添加到header变量 |  |
| login(self) | | 用于处理登录过程，cookie失效报警以及自动登陆成功提醒等 |  |
| login\_status\_chk(self) | | 检查当前登录状态是否有效 |  |
| isResume\_chk(self) | | 检查返回内容是否为合格简历 |  |
| run\_work(self) | | 执行任务主工作入口函数 |  |
| url\_get(self,url,getDict={}) | | 提交GET请求 |  |
| url\_post(self,url,postDict={}) | | 提交POST请求 |  |
| resume\_exist\_chk(self,resume\_id='') | | 根据ID识别对应简历是否已经存在 |  |
| save\_resume(self,resume\_id='',html='') | | 将简历内容保存至特定文件，拼足8位ID，依次取得0-3 3-6 作为两级目录 |  |
| send\_mails(self,title='',msg\_txt='',msgtype=0) | | 发送email |  |
| save\_error\_file(self,msg) | | 将错误信息保存这至特定文件 |  |
| rand\_sleep(self) | | 随机休眠，用于模拟手动操作，避免触及抓取限制 |  |

## cjol抓取文件fetchclass/libcjol.py

* + 1. 文件说明

该文件主要实现cjol的抓取，继承自基类，重写其中的函数，函数和属性信息参考基类部分即可。

* + 1. 代码调用流程



* + 1. cjol返回内容

|  |  |  |
| --- | --- | --- |
| **返回内容标记** | **说明** | **备注** |
| <input id="LoginName" name="UserID" type="text" value="" placeholder="请输入用户名" />  <span id="valUserName" style="color:Red;visibility:hidden;">请输入用户名</span> | 需要登录 | 识别标注码为0:，此时需要跳入登录函数进行登录 |
| 包含“基本信息”、“简历编号” | 识别为成功简历 | 识别标注码为1 |
| 该简历已设置为不可查看 | 简历不可查看 | 识别标注码为2 |
| 该求职上传了附件简历,查看联系方式后可下载 | 附件简历 | 识别标注码为3 |
| 其他信息一律采用文件的形式进行记录 |  | 识别标识码为-1 |

* + 1. cjol返回内容识别标示码

|  |  |  |
| --- | --- | --- |
| **码** | **含义** | **备注** |
| 0 | 需要登录 |  |
| 1 | 正常简历 |  |
| 2 | 简历设置保密不能查看 |  |
| 3 | 附件简历，需要首先查看联系方式 |  |
| -1 | 内容无法识别，记录为错误文件 |  |
| -2 | 请求失败，内容为空 |  |

## 51job抓取文件fetchclass/libjob51.py

* + 1. 文件说明

该文件主要实现51JOB的抓取，继承自基类，重写其中的函数，函数和属性信息参考基类部分即可。

* + 1. 代码调用流程

参考cjol部分即可。

* + 1. cjol返回内容

|  |  |  |
| --- | --- | --- |
| **返回内容标记** | **说明** | **备注** |
| <td colspan="2" class="loginbar"> | 需要登录 | 识别标注码为0:，此时需要跳入登录函数进行登录 |
| '<div id="divResume"><style>','简历编号' | 识别为成功简历 | 识别标注码为1 |
| 此人简历保密 | 简历不可查看 | 识别标注码为2 |
| 该简历为恶意投递，已被屏蔽 | 系统屏蔽简历 | 识别标注码为3 |
| 您的操作过于频繁，请注意劳逸结合 | 操作频繁，尽量避免该提示的出现；调整随机休眠时间，尽量避免即可 | 识别标注码为4 |
| 其他信息一律采用文件的形式进行记录 |  | 识别标识码为-1 |

* + 1. cjol返回内容识别标示码

|  |  |  |
| --- | --- | --- |
| **码** | **含义** | **备注** |
| 0 | 需要登录 |  |
| 1 | 正常简历 |  |
| 2 | 简历设置保密不能查看 |  |
| 3 | 系统屏蔽此人简历 |  |
| 4 | 操作频繁，此时需要进行长时间休眠 |  |
| -1 | 内容无法识别，记录为错误文件 |  |
| -2 | 请求失败，内容为空 |  |

# 数据库表格的结构

目前数据库采用sqllite数据库，数据表格比较简单；主要有两类表格，一类表格只有一张表，用于记录抓取数据源的总数信息；另外一类表格是用来记录分数据源抓取的分段抓取结果，可以有多张表。

## 总数记录表- total\_count

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **类型** | **说明** | **备注** |
| id | 整形自增长 | 记录编号 |  |
| moudle | Varchar(200) | 数据源名称 |  |
| total | Int(10) | 总数目 |  |
| update\_at | timestamp | 记录更新时间 |  |

## Cjol分段抓取记录表-cjol\_seg\_count

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **类型** | **说明** | **备注** |
| id | 整形自增长 | 记录编号 |  |
| num | Int | 成功抓取的个数 |  |
| update\_at | timestamp | 记录更新时间 |  |
| begin\_num | Int | Id开始值 |  |
| end\_num | Int | Id结束值 |  |
| get\_rate | int | 命中率百分比 |  |

## 51job分段抓取记录表

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **类型** | **说明** | **备注** |
| id | 整形自增长 | 记录编号 |  |
| num | Int | 成功抓取的个数 |  |
| update\_at | timestamp | 记录更新时间 |  |
| begin\_num | Int | Id开始值 |  |
| end\_num | Int | Id结束值 |  |
| get\_rate | int | 命中率百分比 |  |

# 代码的二次开发流程

此处对于如何实现简历爬取细节不做赘述，仅作代码书写的指引；或者根据个人兴趣，自行发挥也可直接忽略此处描述。

1. 新建抓取模块文件，创建新的抓取功能类继承基类。
2. 修改继承的变量数值，并且根据具体需要增加子类变量并作初始化
3. 确定抓取任务执行流程，重写run\_work（）、load\_task（）、isResume\_chk()、login()等函数
4. 根据需要进行其他函数的定义和书写，并且按需调用即可。
5. 如需记录数据库信息，
   * 首先做好数据库表格的定义，并且做好文档记录；
   * 然后，在database/fetch.db文件之中新建表格；
   * 接着，则打开fetchclass/dbctrl.py文件，修改原有的data\_total\_add()、data\_seg\_record()函数或者适当增加函数，以备程序调用；
   * 最后，修改showresult.py之中的列表参数和函数，完成实时展示数据库记录的统计功能。
6. 完成以上流程之后，为了方便执行，只需要简单修改以下两个文件即可。
   * 需要在main.py入口文件之中增加新的抓取模块的执行方式以及参数配置
   * 需要在viewstats.py文件之中，增加数据源的抓取结果展示的入口参数