**scrapyd学习**

# 简介

scrapyd是运行scrapy爬虫的服务程序，它支持以http命令方式发布、删除、启动、停止爬虫程序。而且scrapyd可以同时管理多个爬虫，每个爬虫还可以有多个版本。

特点：

1、可以避免爬虫源码被看到。

2、有版本控制。

3、可以远程启动、停止、删除

使用版本：

scrapyd：1.2.0

scrapy：1.5.0

# 安装

## pip

pip install scrapyd

可以进行安装

## github

在 <https://github.com/scrapy/scrapyd> 中下载最新版本的 zip

下载后解压，然后在cmd命令窗口中切换到解压目录，

使用 python setup.py install 手动安装

有个必备的组件库：

scrapy 通过pip安装 1.5.0

enum-compat ，通过pip 安装 0.0.2

scrapyd-client 通过pip 安装 1.1.0

w3lib 通过pip 安装 1.19.0

# 文档

官方文档：<http://scrapyd.readthedocs.io/en/stable/>

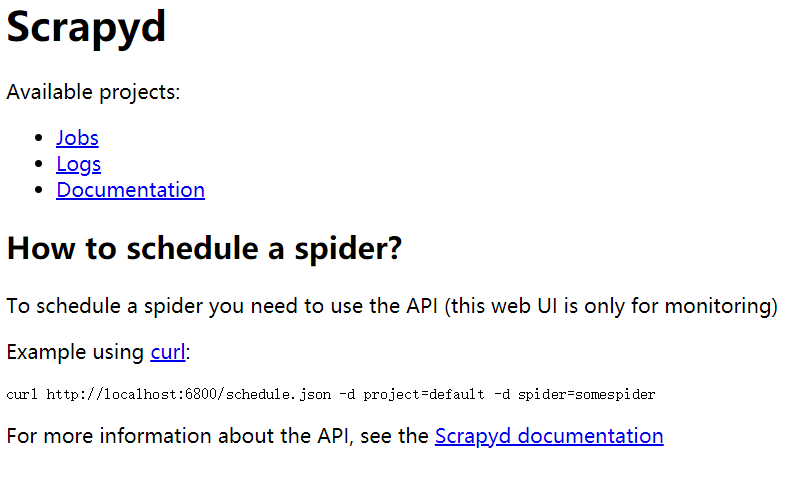
# 使用

## 启动scrapyd

在 cmd 窗口中， 运行 scrapyd

|  |
| --- |
| 2018-04-12T23:21:58+0800 [-] Loading F:\python3\_env\pycharm\_env\lib\site-packages\scrapyd-1.2.0-py3.6.egg\scrapyd\txapp.py...  2018-04-12T23:22:00+0800 [-] Scrapyd web console available at http://127.0.0.1:6800/  2018-04-12T23:22:00+0800 [-] Loaded.  2018-04-12T23:22:00+0800 [twisted.application.app.AppLogger#info] twistd 17.9.0 (F:\python3\_env\pycharm\_env\Scripts\python.exe 3.6.3) starting up.  2018-04-12T23:22:00+0800 [twisted.application.app.AppLogger#info] reactor class: twisted.internet.selectreactor.SelectReactor.  2018-04-12T23:22:00+0800 [-] Site starting on 6800  2018-04-12T23:22:00+0800 [twisted.web.server.Site#info] Starting factory <twisted.web.server.Site object at 0x0000021EBAA505C0>  2018-04-12T23:22:00+0800 [Launcher] Scrapyd 1.2.0 started: max\_proc=32, runner='scrapyd.runner' |

会显示如上信息，scrapyd默认绑定 6800 ，可以通过 <http://127.0.0.1:6800> 访问，访问后显示页面如下：



我们稍后来介绍此界面的操作

## 配置scrapyd.conf

官方说明 配置文档 位置：

* /etc/scrapyd/scrapyd.conf (Unix)
* c:\scrapyd\scrapyd.conf (Windows)
* /etc/scrapyd/conf.d/\* (in alphabetical order, Unix)
* scrapyd.conf
* ~/.scrapyd.conf (users home directory)

实际路径：

setup.py安装：python安装目录\Lib\site-packages\scrapyd-1.2.0-py3.6.egg\scrapyd

或

pycharm安装：python安装目录\Lib\site-packages\scrapyd

default\_scrapyd.conf：

如果没有指定 eggs 等的目录，默认存储在 C:\Users\你的windows账号\Recent

|  |
| --- |
| [scrapyd]  # 项目的 eggs 存储位置  eggs\_dir **=** eggs  # Scrapy日志的存储目录。如果要禁用存储日志，请将此选项设置为空，如下  # logs\_dir =  logs\_dir **=** logs  # Scrapyitem将被存储的目录，默认情况下禁用此选项，如果设置了 值，会覆盖 scrapy的 FEED\_URI 配置项  items\_dir **=**  # 每个蜘蛛保持完成的工作数量。默认为5  jobs\_to\_keep **=** 5  # 项目数据库存储的目录  dbs\_dir **=** dbs  # 并发scrapy进程的最大数量，默认为0，没有设置或者设置为0时，将使用系统中可用的cpus数乘以max\_proc\_per\_cpu配置的值  max\_proc **=** 0  # 每个CPU启动的进程数，默认4  max\_proc\_per\_cpu **=** 4  # 保留在启动器中的完成进程的数量。默认为100  finished\_to\_keep **=** 100  # 用于轮询队列的时间间隔，以秒为单位。默认为5.0  poll\_interval **=** 5.0  # webservices监听地址  bind\_address **=** 127.0**.**0.1  # 默认 http 监听端口  http\_port **=** 6800  # 是否调试模式  debug **=** off  # 将用于启动子流程的模块，可以使用自己的模块自定义从Scrapyd启动的Scrapy进程  runner **=** scrapyd**.**runner  application **=** scrapyd**.**app**.**application  launcher **=** scrapyd**.**launcher**.**Launcher  webroot **=** scrapyd**.**website**.**Root  [services]  schedule**.**json **=** scrapyd**.**webservice**.**Schedule  cancel**.**json **=** scrapyd**.**webservice**.**Cancel  addversion**.**json **=** scrapyd**.**webservice**.**AddVersion  listprojects**.**json **=** scrapyd**.**webservice**.**ListProjects  listversions**.**json **=** scrapyd**.**webservice**.**ListVersions  listspiders**.**json **=** scrapyd**.**webservice**.**ListSpiders  delproject**.**json **=** scrapyd**.**webservice**.**DeleteProject  delversion**.**json **=** scrapyd**.**webservice**.**DeleteVersion  listjobs**.**json **=** scrapyd**.**webservice**.**ListJobs  daemonstatus**.**json **=** scrapyd**.**webservice**.**DaemonStatus |

## 发布项目

### 拷贝 scrapyd-deploy

拷贝 python安装目录\Scripts 下的 scrapyd-deploy 到scrapy 项目的根目录下

### 修改 scrapy.cfg

* 1. 去掉url前的注释符号，这里url就是你的scrapyd服务器的网址

修改为：

**url** = **http://localhost:6800/addversion.json**

* 1. [deploy] 修改为 [deploy:127]，这个 target：127是爬虫服务器的名称 ，

这个 [deploy] 可以配置多个

### 查看scrapd服务配置

在 cmd 命令窗口中 ，切换到 scrapy 项目根目录

python scrapyd-deploy -l

显示：

127 <http://127.0.0.1:6800/>

### 发布爬虫

python scrapyd-deploy <target> -p <project> --version <version>

target：之前scrapy.cfg配置的 [deploy:127] 中的 127

project：项目名称，一般使用和scrapy项目一个名字

version：版本号，默认是当前时间戳

命令：

python scrapyd-deploy 127 -p first\_scrapyd

Packing version 1523550661

Deploying to project "first\_scrapyd" in http://localhost:6800/addversion.json

Server response (200):

{"node\_name": "Terry", "status": "ok", "project": "first\_scrapyd", "version": "1523550661", "spiders": 2}

### 控制API

所有的API都是通过http协议发送的请求，目前总共10个api

规则是：<http://ip:port/api_command.json>，有GET和POST两种请求

#### daemonstatus.json

检查服务的状态

GET请求：

|  |
| --- |
| curl http:**//**localhost:6800**/**daemonstatus**.**json |

结果示例：

|  |
| --- |
| { "status": "ok", "running": "0", "pending": "0", "finished": "0", "node\_name": "node-name" } |

#### addversion.json

增加项目到服务器，如果项目已经存在，则增加一个新的版本

POST请求：

* project (string, required) – 项目名
* version (string, required) – 项目版本，不填写则是当前时间戳
* egg (file, required) – 当前项目的egg文件

|  |
| --- |
| curl http://localhost:6800/addversion.json -F project=myproject -F version=r23 -F egg=@myproject.egg |

结果示例：

|  |
| --- |
| {"status": "ok", "spiders": 3} |

#### schedule.json

启动一个爬虫项目

POST请求：

* project (string, required) – 项目名
* spider (string, required) – 爬虫名，spider类中指定的name
* setting (string, optional) – 自定义爬虫settings
* jobid (string, optional) – jobid，之前启动过的spider，会有一个id
* \_version (string, optional) – 版本号，之前部署的时候的version，只能使用int数据类型，没指定，默认启动最新版本
* 其他额外的参数都会放入到spider的参数中

|  |
| --- |
| curl http://localhost:6800/schedule.json -d project=myproject -d spider=somespider |

结果示例：

|  |
| --- |
| {"status": "ok", "jobid": "6487ec79947edab326d6db28a2d86511e8247444"} |

#### cancel.json

取消一个 spdier 的运行

如果 spider 是运行状态，则停止其运行

如果 spider 是 挂起状态，则删除spider

POST请求：

* project (string, required) – 项目名
* job (string, required) -jobid

|  |
| --- |
| curl http://localhost:6800/cancel.json -d project=myproject -d job=6487ec79947edab326d6db28a2d86511e8247444 |

结果示例：

|  |
| --- |
| {"status": "ok", "prevstate": "running"} |

#### listprojects.json

获取当前已上传的项目的列表

GET请求：

|  |
| --- |
| curl http://localhost:6800/listprojects.json |

结果示例：

|  |
| --- |
| {"status": "ok", "projects": ["myproject", "otherproject"]} |

#### listversions.json

获取指定项目的可用版本

GET请求：

* project (string, required) – 项目名

|  |
| --- |
| $ curl http://localhost:6800/listversions.json?project=myproject |

结果示例：

|  |
| --- |
| {"status": "ok", "versions": ["r99", "r156"]} |

#### listspiders.json

获取指定版本的项目中的爬虫列表，如果没有指定版本，则是最新版本

GET请求：

* project (string, required) – 项目名
* \_version (string, optional) – 版本号

|  |
| --- |
| $ curl http://localhost:6800/listspiders.json?project=myproject |

结果示例：

|  |
| --- |
| {"status": "ok", "spiders": ["spider1", "spider2", "spider3"]} |

#### listjobs.json

获取指定项目中 所有挂起、运行和运行结束的job

GET请求

* project (string, option) - restrict results to project name

|  |
| --- |
| curl http://localhost:6800/listjobs.json?project=myproject | python -m json.tool |

结果示例：

|  |
| --- |
| {  "status": "ok",  "pending": [  {  "project": "myproject", "spider": "spider1",  "id": "78391cc0fcaf11e1b0090800272a6d06"  }  ],  "running": [  {  "id": "422e608f9f28cef127b3d5ef93fe9399",  "project": "myproject", "spider": "spider2",  "start\_time": "2012-09-12 10:14:03.594664"  }  ],  "finished": [  {  "id": "2f16646cfcaf11e1b0090800272a6d06",  "project": "myproject", "spider": "spider3",  "start\_time": "2012-09-12 10:14:03.594664",  "end\_time": "2012-09-12 10:24:03.594664"  }  ]  } |

#### delversion.json

删除指定项目的指定版本

POST请求

* project (string, required) - the project name
* version (string, required) - the project version

|  |
| --- |
| curl http://localhost:6800/delversion.json -d project=myproject -d version=r99 |

结果示例：

|  |
| --- |
| {"status": "ok"} |

#### delproject.json

删除指定项目，并且包括所有的版本

POST请求

* project (string, required) - the project name

|  |
| --- |
| curl http://localhost:6800/delproject.json -d project=myproject |

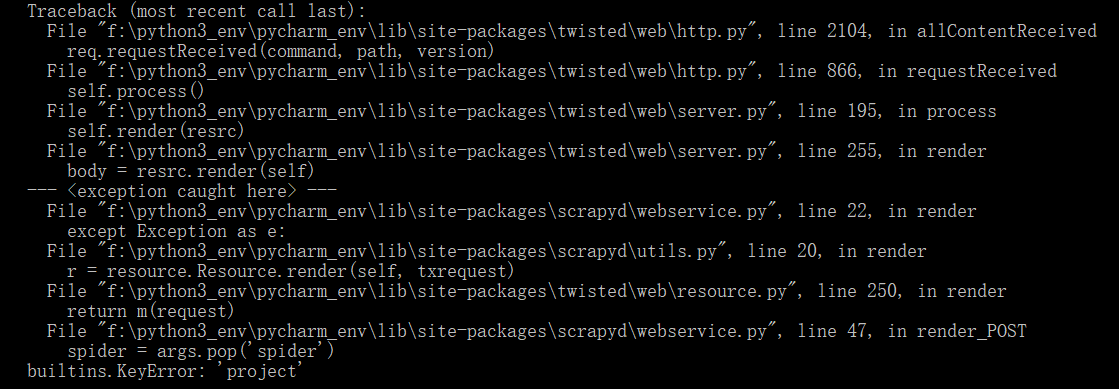
结果示例：

|  |
| --- |
| {"status": "ok"} |

# BUG处理

1. builtins.KeyError: 'project'

错误信息如下：



解决：

进行post提交时，需要将参数提交放入到 params 或 data 中，而不是json

如：

requests.post(url, params=params)

或

requests.post(url, data=params)

1. TypeError: \_\_init\_\_() missing 1 required positional argument: 'self'

修改 spider ，增加 ：

**def** \_\_init\_\_(self, \*\*kwargs):  
 super(DingdianSpider, self).\_\_init\_\_(self, \*\*kwargs)

1. redis.exceptions.ConnectionError: Error 10061 connecting to localhost:6379

有类似这样的错误，是由于项目中有连接其他服务，譬如这里是redis数据库，需要先启动 对应的服务