
ProxyPool

发布 *2.1.0*

2021 年 04 月 08 日

Contents

1 安装	3
1.1 使用	4
2 Contents	7
2.1 用户指南	7
2.2 ChangeLog	11
Python 模块索引	15
索引	17


```
*****
***  _ _ _ _ _  *****  _ _ _ _ _  *****  _  *****
*** |  _ _  \_  *****  |  _ _  \  *****  |  |  *****
*** | |_/ / \_ _ _ _ _ _ _ _ _ _ | |_/ / _ _ _ * _ _ _ | |  *****
*** |  _ _/ |  _// _ \ \ \ / | | | |  _// _ \ / _ \ | |  *****
*** | |  | | | ( _ ) | > < \ | _ | | | | ( _ ) | ( _ ) | |  _ _  ****
*** \_ |  | _ | \ _ _ / / _ \ \ \ _ _ | \_ |  \ _ _ / \ _ _ / \ _ _ _ /  ****
****                               _ _ / /                               ****
***** / _ _ _ / *****
*****                               *****
*****                               *****
*****
```

Python 爬虫代理 IP 池

安装

- 下载代码

```
$ git clone git@github.com:jhao104/proxy_pool.git
```

- 安装依赖

```
$ pip install -r requirements.txt
```

- 更新配置

```
HOST = "0.0.0.0"
PORT = 5000

DB_CONN = 'redis://@127.0.0.1:8888'

PROXY_FETCHER = [
    "freeProxy01",
    "freeProxy02",
    # ....
]
```

- 启动项目

```
$ python proxyPool.py schedule
$ python proxyPool.py server
```

1.1 使用

- API

Api	Method	Description	Arg
/	GET	API 介绍	无
/get	GET	随机返回一个代理	无
/get_all	GET	返回所有代理	无
/get_status	GET	返回代理数量	无
/delete	GET	删除指定代理	proxy=host:ip

- 爬虫

```
import requests

def get_proxy():
    return requests.get("http://127.0.0.1:5010/get/").json()

def delete_proxy(proxy):
    requests.get("http://127.0.0.1:5010/delete/?proxy={}".format(proxy))

# your spider code

def getHtml():
    # ....
    retry_count = 5
    proxy = get_proxy().get("proxy")
    while retry_count > 0:
        try:
            html = requests.get('http://www.example.com', proxies={"http": "http://{}".format(proxy)})
            # 使用代理访问
            return html
        except Exception:
            retry_count -= 1
            # 删除代理池中代理
```

(下页继续)

(续上页)

```
        delete_proxy(proxy)
    return None
```


2.1 用户指南

2.1.1 如何运行

下载代码

本项目需要下载代码到本地运行, 通过 `git` 下载:

```
$ git clone git@github.com:jhao104/proxy_pool.git
```

或者下载特定的 `release` 版本:

```
https://github.com/jhao104/proxy\_pool/releases
```

安装依赖

到项目目录下使用 `pip` 安装依赖库:

```
$ pip install -r requirements.txt
```

更新配置

配置文件 `setting.py` 位于项目的主目录下:

```
# 配置 API 服务

HOST = "0.0.0.0"           # IP
PORT = 5000                # 监听端口

# 配置数据库

DB_CONN = 'redis://@127.0.0.1:8888/0'

# 配置 ProxyFetcher

PROXY_FETCHER = [
    "freeProxy01",         # 这里是启用的代理抓取方法, 所有 fetch 方法位于 fetcher/
    ↪ proxyFetcher.py
    "freeProxy02",
    # ....
]
```

更多配置请参考[配置参考](#)

启动项目

如果已配置好运行环境, 具备运行条件, 可以通过 `proxyPool.py` 启动. `proxyPool.py` 是项目的 CLI 入口. 完整程序包含两部份: `schedule` 调度程序和 `server` API 服务, 调度程序负责采集和验证代理, API 服务提供代理服务 HTTP 接口.

通过命令行程序分别启动调度程序和 API 服务:

```
# 启动调度程序
$ python proxyPool.py schedule

# 启动 webApi 服务
$ python proxyPool.py server
```

2.1.2 如何使用

爬虫代码要对接代理池目前有两种方式: 一是通过调用 API 接口使用, 二是直接读取数据库.

调用 API

启动 ProxyPool 的 server 后会提供如下几个 http 接口:

Api	Method	Description	Arg
/	GET	API 介绍	无
/get	GET	随机返回一个代理	无
/get_all	GET	返回所有代理	无
/get_status	GET	返回代理数量	无
/delete	GET	删除指定代理	proxy=host:ip

在代码中可以通过封装上面的 API 接口来使用代理, 例子:

```
import requests

def get_proxy():
    return requests.get("http://127.0.0.1:5010/get/").json()

def delete_proxy(proxy):
    requests.get("http://127.0.0.1:5010/delete/?proxy={}".format(proxy))

# your spider code

def getHtml():
    # ....
    retry_count = 5
    proxy = get_proxy().get("proxy")
    while retry_count > 0:
        try:
            # 使用代理访问
            html = requests.get('http://www.example.com', proxies={"http": "http://{}".format(proxy)})
            return html
        except Exception:
            retry_count -= 1
            # 删除代理池中代理
            delete_proxy(proxy)
    return None
```

本例中我们在本地 127.0.0.1 启动端口为 5010 的 server, 使用 /get 接口获取代理, /delete 删除代理.

读数据库

目前支持配置两种数据库: REDIS 、 SSDB.

- REDIS 储存结构为 hash, hash name 为配置项中的 TABLE_NAME
- SSDB 储存结构为 hash, hash name 为配置项中的 TABLE_NAME

可以在代码中自行读取.

2.1.3 配置参考

配置文件 `setting.py` 位于项目的主目录下, 配置主要分为四类: 服务配置、数据库配置、采集配置、校验配置.

服务配置

- HOST

API 服务监听的 IP, 本机访问设置为 127.0.0.1, 开启远程访问设置为: 0.0.0.0.

- PORT

API 服务监听的端口.

数据库配置

- DB_CONN

用户存放代理 IP 的数据库 URI, 配置格式为: `db_type://[[user]:[pwd]]@ip:port/[db]`.

目前支持的 db_type 有: `ssdb` 、 `redis`.

配置示例:

```
# SSDB IP: 127.0.0.1 Port: 8888
DB_CONN = 'ssdb://@127.0.0.1:8888'
# SSDB IP: 127.0.0.1 Port: 8899 Password: 123456
DB_CONN = 'ssdb://:123456@127.0.0.1:8888'

# Redis IP: 127.0.0.1 Port: 6379
DB_CONN = 'redis://@127.0.0.1:6379'
# Redis IP: 127.0.0.1 Port: 6379 Password: 123456
DB_CONN = 'redis://:123456@127.0.0.1:6379'
# Redis IP: 127.0.0.1 Port: 6379 Password: 123456 DB: 15
DB_CONN = 'redis://:123456@127.0.0.1:6379/15'
```

- TABLE_NAME

存放代理的数据载体名称, ssdb 和 redis 的存放结构为 hash.

采集配置

- PROXY_FETCHER

启用的代理采集方法名, 代理采集方法位于 `fetcher/proxyFetcher.py` 类中.

由于各个代理源的稳定性不容易掌握, 当某个代理采集方法失效时, 可以在该配置中注释掉其名称.

如果有增加某些代理采集方法, 也请在该配置中添加其方法名, 具体请参考 `/dev/extend_fetcher`.

调度程序每次执行采集任务时都会再次加载该配置, 保证每次运行的采集方法都是有效的.

校验配置

- VERIFY_RUL

用于检验代理是否可用的地址, 默认为 `http://www.baidu.com`, 可根据使用场景修改为其他地址.

- VERIFY_TIMEOUT

检验代理的超时时间, 默认为 10, 单位秒. 使用代理访问 VERIFY_RUL 耗时超过 VERIFY_TIMEOUT 时, 视为代理不可用.

- MAX_FAIL_COUNT

检验代理允许最大失败次数, 默认未 0, 即出错一次即删除.

2.2 ChangeLog

2.2.1 2.2.0 (2021-04-08)

1. 启动时检查数据库连通性;
2. 新增免费代理源 米扑代理;
3. 新增免费代理源 Pzzqz;
4. 新增免费代理源 神鸡代理;
5. 新增免费代理源 极速代理;
6. 新增免费代理源 小幻代理;

2.2.2 2.1.1 (2021-02-23)

1. Fix Bug [#493](#), 新增时区配置; (2020-08-12)
2. 修复 **66** 代理采集; (2020-11-04)
3. 修复 **全网代理**采集, 解决 HTML 端口加密问题; (2020-11-04)
4. 新增 **代理盒子**免费源; (2020-11-04)
5. 新增 POOL_SIZE_MIN 配置项, runProxyCheck 时, 剩余代理少于 POOL_SIZE_MIN 触发抓取; (2021-02-23)

2.2.3 2.1.0 (2020.07)

1. 新增免费代理源 **西拉代理** (2020-03-30)
2. Fix Bug [#356](#) [#401](#)
3. 优化 Docker 镜像体积; (2020-06-19)
4. 优化配置方式;
5. 优化代码结构;
6. 不再储存 raw_proxy, 抓取后直接验证入库;

2.2.4 2.0.1 (2019.10)

1. 新增免费代理源 **89 免费代理**;
2. 新增免费代理源 **齐云代理**

2.2.5 2.0.0 (2019.08)

1. WebApi 集成 Gunicorn 方式启动, Windows 平台暂不支持;
2. 优化 Proxy 调度程序;
3. 扩展 Proxy 属性;
4. 新增 cli 工具, 更加方便启动 proxyPool

2.2.6 1.14 (2019.07)

1. 修复 Queue 阻塞导致的 ProxyValidSchedule 假死 bug;
2. 修改代理源 **云代理**抓取;
3. 修改代理源 **码农代理**抓取;

4. 修改代理源 **代理 66** 抓取, 引入 PyExecJS 模块破解加速乐动态 Cookies 加密;

2.2.7 1.13 (2019.02)

1. 使用.py 文件替换.ini 作为配置文件;
2. 优化代理采集部分;

2.2.8 1.12 (2018.04)

1. 优化代理格式检查;
2. 增加代理源;
3. fix bug #122 #126

2.2.9 1.11 (2017.08)

1. 使用多线程验证 useful_pool;

2.2.10 1.10 (2016.11)

1. 第一版;
2. 支持 PY2/PY3;
3. 代理池基本功能;

u

`user`, 7

U

`user` (模块), 7