

目录

前言	1.1
PySpider简介	1.2
PySpider安装与基本用法	1.3
PySpider安装	1.3.1
安装和启动的常见问题	1.3.1.1
PySpider基本用法	1.3.2
项目操作举例	1.3.2.1
查找提取元素	1.3.2.2
PySpider的高级用法	1.4
self.crawl	1.4.1
config.json	1.4.2
data目录	1.4.3
phantomjs	1.4.4
PySpider经验与心得	1.5
PySpider的心得	1.5.1
删除项目	1.5.1.1
PySpider常见的坑	1.5.2
PySpider案例	1.6
汽车之家的品牌等数据	1.6.1
汽车之家的车型详细数据	1.6.2
过程	1.6.2.1
代码	1.6.2.2
结果	1.6.2.3
百度热榜	1.6.3
附录	1.7
参考资料	1.7.1

Python爬虫框架：PySpider

- 最新版本: v2.0
- 更新时间: 20210908

简介

PySpider是一个简单易用且强大的Python主流爬虫框架。此处总结PySpider的安装和基本的使用，以及安装和启动时常见问题，并且给出查找定位元素的PyQuery的基本用法举例，以及详细描述了项目中从新建到调试的每一步的细节如何操作，以及一些高级用法，比如self.craw、config.json、data目录、phantomjs，和一些心得，比如删除项目，以及常见的坑，并且给出一些实际的例子供参考，包括汽车之家的品牌车型车系数据、汽车之家的车型详细数据、百度首页热榜列表等。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- [crifan/python_spider_pyspider: Python爬虫框架：PySpider](#)

如何使用此Gitbook源码去生成发布为电子书

详见：[crifan/gitbook_template: demo how to use crifan gitbook template and demo](#)

在线浏览

- [Python爬虫框架：PySpider book.crifan.com](#)
- [Python爬虫框架：PySpider crifan.github.io](#)

离线下载阅读

- [Python爬虫框架：PySpider PDF](#)
- [Python爬虫框架：PySpider ePUB](#)
- [Python爬虫框架：PySpider Mobi](#)

版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分內容参考自网络，均已备注了出处。如有发现侵犯您版权，请通过邮箱联系我 `admin` 艾特 `crifan.com`，我会尽快删除。谢谢合作。

鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

更多其他电子书

本人 `crifan` 还写了其他 `100+` 本电子书教程，感兴趣可移步至：

[crifan/crifan_ebook_readme: Crifan的电子书的使用说明](#)

`crifan.com`, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-09-08 13:59:23

PySpider简介

PySpider 的基本信息：

- 是个Python的爬虫框架
- 最大特点：
 - 带图形界面WebUI的调试
 - 简单易用
 - 同时功能也很强大
- GitHub
 - <https://github.com/binux/pyspider>
- 文档：
 - 官方，英文：<http://docs.pyspider.org/>
 - 非官方，中文：<http://www.pyspider.cn/index.html>
- 作者
 - 网名： Binux
 - 别名：足又兆虫
 - 博客：[Binuxの杂货铺](#)
 - Github: binux (Roy Binux)

PySpider 对比 Scrapy

对于两个流行的Python的爬虫框架，PySpider和Scrapy，常常会被人拿来对比。

对此，之前简单总结如下：

- PySpider：简单易上手，带图形界面（基于浏览器页面）
 - 一图胜千言：在WebUI中调试爬虫代码

The screenshot shows the PySpider WebUI interface. On the left is a code editor with Python code for extracting car data from a specific URL. The code uses BeautifulSoup to parse the HTML and extract various details like car model, price, and dealer information. On the right is a terminal window showing the execution of the script and the resulting JSON output, which includes fields such as 'carModelName', 'carModelPrice', 'cityDealerPrice', and 'cityDealerPriceInt'. Below the terminal, there's a preview of the extracted data in a table format.

```
1.13     # self.log.info("message(%s, project_name, curSerieDict, url=eachModelDetailDict['url'])")
1.14     url=eachModelDetailDict['url']
1.15     print("%d %d %s" % (curIdx, curSerieDict))
1.16     self.crawl(eachModelDetailDict['url'],
1.17     callback=self.carModelSpecPage, fetch_type='js', save=curSerieDict)
1.18
1.19     # print('allSerieDictList', allSerieDictList)
1.20
1.21     #def carModelSpecPage(self, project, msg):
1.22     #    self.log.info("message(%s, project, msg)", msg)
1.23     #    print("error message(%s, msg)" , msg)
1.24     #    return msg
1.25
1.26     #catch_status_code_error
1.27     def carModelSpecPage(self, response):
1.28         print("carModelSpecPage(%s, response)", response)
1.29         # https://www.autohome.com.cn/spec/32708/#pvareaid=2042128
1.30         curSerieDict = response.save
1.31         print("curSerieDict", curSerieDict)
1.32
1.33         cityDealerPriceInt = 0
1.34         cityDealerPrice = response.doc('.cardetail-infor-price')
1.35         #cityDealerPrice attr(class="price")
1.36         print("cityDealerPrice %s" % cityDealerPrice)
1.37         if cityDealerPrice:
1.38             cityDealerPriceIntStr = cityDealerPriceInt.text()
1.39             cityDealerPriceFloatStr = float(cityDealerPriceIntStr)
1.40             cityDealerPriceInt = int(cityDealerPriceFloatStr)
1.41             print("cityDealerPriceInt %s" % cityDealerPriceInt)
1.42             print("cityDealerPriceInt", cityDealerPriceInt)
1.43
1.44             maprPriceInt = 0
1.45             # body > div > div > div.colmain > div.cardetail-fn > div.cardetail-infor-
1.46             # price.fn-clear > div.cardetail-infor > div.cardetail-infor-
1.47             # price
1.48             maprPriceEl = response.doc('.cardetail-infor-price')
1.49             if maprPriceEl:
1.50                 maprPriceInt = maprPriceEl.attr('data-price')
1.51                 print("maprPriceInt", maprPriceInt)
1.52                 if maprPriceEl:
1.53                     maprPriceEl = maprPriceEl.attr('data-price')
1.54                     print("maprPriceEl", maprPriceEl)
1.55
1.56
1.57
1.58
1.59
1.60
1.61
1.62
1.63
1.64
1.65
1.66
1.67
1.68
1.69
1.70
1.71
1.72
1.73
1.74
1.75
1.76
1.77
1.78
1.79
1.80
```

- Scrapy：可以高级定制化实现更加复杂的控制
 - 一图胜千言：Scrapy一般是在命令行界面中调试页面返回数据：

安装和启动的常见问题

详见：[【整理】pyspider vs scrapy](#)

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook 最后更新: 2020-07-30 14:00:04

PySpider安装与基本用法

概述：

- 安装

```
pip install pyspider
```

- 启动

```
pyspider
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新： 2021-04-14 22:39:25

PySpider安装

下面介绍，如何安装 PySpider。

Mac中安装PySpider

此处以Mac中为例，去介绍如何安装PySpider。

如果只是简单的直接安装的话，则可以去：

```
pip install pyspider
```

安装phantomjs

如果后续用到网页中要允许js的代码，则需要用到无头浏览器 phantomjs，就需要先去安装 phantomjs

```
brew cask install phantomjs
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-01-16 21:34:45

安装和启动的常见问题

此处整理在安装和启动 `pyspider` 期间，经常遇到的问题和其原因及解决办法。

启动报错：`async=True SyntaxError: invalid syntax`

- 运行 `pyspider` 报错：

```
File "/Users/limao/.pyenv/versions/3.8.0/Python.frame
      ^
SyntaxError: invalid syntax
```

- 原因：PySpider（很久没继续维护了）最新支持版本是 Python 3.6，其把 `async` 作为普通函数参数，是没问题的。
 - 但是 Python 3.7 之后把 `async` 改为了系统保留字，表示 异步，所以 `async` 不能再作为普通函数参数名
 - 所以当前环境是 Python 3.7+ 时，就会报语法错误了
- 解决办法：2种思路：
 - （自己）把代码（中这类的语法错误）都改掉
 - 把当前Python版本换成旧版本： Python 3.6
 - 此处选择换旧版本 Python 3.6
- 步骤：

Mac中换Python为3.6

此处以 `pyenv` 为例：

先去安装Python 3.6的某个版本：

```
pyenv install 3.6.8
```

再去设置使用 Python 3.6：

- 本地

```
pyenv local 3.6.8
```

- 或全局

```
pyenv global 3.6.8
```

然后重新安装：

```
pip install pyspider
```

之后即可正常运行

```
pyspider
```

详见：

[【已解决】Mac中给Python3安装PySpider](#)

推荐用虚拟环境

在Python的开发中，为了避免不同开发环境的互相影响，一般都会用虚拟环境工具，比如 `pipenv`、`virtualenv`。

注意：此处即使用Python虚拟环境，由于前面提到的问题，PySpider也需要用 Python 3.6 的

pipenv中安装PySpider

此处是 `pipenv`，为了指定使用 Python 3.6 则，可以在 `Pipfile` 中加上：

```
[requires]
python_version = "3.6"
```

贴出完整配置：

```
[[source]]
#url = "https://pypi.python.org/simple"
url = "https://pypi.tuna.tsinghua.edu.cn/simple"
verify_ssl = true
name = "pypi"

[packages]
pymysql = "*"

[dev-packages]

[requires]
python_version = "3.6"
```

创建并安装PySpider：

```
pipenv install pyspider
```

或：

先创建虚拟环境，再安装PySpider

```
pipenv install  
pipenv shell  
pip install pyspider
```

virtualenv中安装PySpider

关于virtualenv：

- 创建虚拟环境： virtualenv venv
- 激活激活环境并进入：
 - Mac/Linux： source venv/bin/activate
 - 或：
 - . venv/bin/activate
 - Win： venv\Scripts\activate.bat
- 退出虚拟环境： deactivate

在 virtualenv 中安装PySpider：

```
pip install pyspider
```

启动报错： pycurl报ImportError错 或 Curl 报ConfigurationError错

Mac 中，如果安装期间出错：

```
|__main__.ConfigurationError: Curl is configured to use SSL, but  
|we have not been able to determine which SSL backend it is using
```

或运行时报错：

```
ImportError: pycurl: libcurl link-time ssl backend (openssl)
```

- 原因： 此处导入pycurl时，发现libcurl运行时所依赖的ssl的底层是openssl，和当时编译时的版本不匹配
- 解决办法： 重新编译安装，使得版本一致
- 步骤：

```
pip uninstall -y pycurl
export PYCURL_SSL_LIBRARY=openssl
export LDFLAGS=-L/usr/local/opt/openssl/lib;export CPPFLAGS=-I/usr/local/opt/openssl/include
```

附上，进入虚拟环境后再操作的例子：

```
→ ChildQuPeiYinApp_downloadDemo pipenv shell
Launching subshell in virtual environment...
. /Users/crifan/.local/share/virtualenvs/ChildQuPeiYinApp_
→ ChildQuPeiYinApp_downloadDemo . /Users/crifan/.local/st
→ ChildQuPeiYinApp_downloadDemo pip uninstall pycurl
Skipping pycurl as it is not installed.
→ ChildQuPeiYinApp_downloadDemo export PYCURL_SSL_LIBRARY=
→ ChildQuPeiYinApp_downloadDemo export LDFLAGS=-L/usr/locat
Collecting pycurl
    Downloading https://files.pythonhosted.org/packages/e8/e
      100% |██████████| 215kB 198kB/s
Installing collected packages: pycurl
  Running setup.py install for pycurl ... done
Successfully installed pycurl-7.43.0.2
```

注意：上述的：

- `/usr/local/opt/openssl` 是你的openssl安装路径
 - 如果你的不是这个路径，要换成你的实际路径
 - 对应的
 - `/usr/local/opt/openssl/lib` 是 lib 库的路径
 - `/usr/local/opt/openssl/include` 是 include 头文件的路径

详见：

- [【记录】Mac中安装和运行pyspider](#)
- [【已解决】pipenv虚拟环境中用pip安装pyspider出错：`__main__.ConfigurationError: Curl is configured to use SSL, but we have not been able to determine which SSL backend it is using`](#)
- [【已解决】pyspider运行出错：`ImportError pycurl libcurl link-time ssl backend \(openssl\) is different from compile-time ssl backend \(none/other\)`](#)

启动报错：fatal error openssl/ssl.h file not found

如果上面步骤：

安装和启动的常见问题

```
export LDFLAGS=-L/usr/local/opt/openssl/lib;export  
CPPFLAGS=-I/usr/local/opt/openssl/include;pip install pycurl  
--compile --no-cache-dir
```

报错：

```
In file included from src/docstrings.c:4:  
src/pycurl.h:165:13: fatal error: 'openssl/ssl.h' file  
# include <openssl/ssl.h>  
1 error generated.  
error: command 'gcc' failed with exit status 1
```

- 直接原因：找不到 openssl/ssl.h
- 多种可能
 - 之前没安装过 openssl
 - 解决办法：Mac中去安装：

```
brew install openssl
```

- 然后再重试即可
- Mac中已安装过 openssl
 - 所以此处Mac中是有 openssl/ssl.h 的，只是传入的路径不对
 - 解决办法：找到已安装的 openssl 的实际路径，传入正确的路径。
 - 步骤：

找到已安装的openssl的实际安装路径

```
brew info openssl
```

可以看到有：

```
/usr/local/Cellar/openssl@1.1/1.1.1d (7,983 files, 17.9MB)
```

其中的：

```
/usr/local/Cellar/openssl@1.1/1.1.1d
```

就是我们要的，此处openssl的实际安装路径

通过

```
open /usr/local/Cellar/openssl@1.1/1.1.1d
```

确认是有对应的：

```
/usr/local/Cellar/openssl@1.1/1.1.1d/include/openssl/ssl.h
```

这个文件的。所以传入路径应该改为：

```
/usr/local/Cellar/openssl@1.1/1.1.1d/include
```

完整命令是：

```
export LDFLAGS=-L/usr/local/opt/openssl/lib;export CPPFLAGS=-I/usr/local/opt/openssl/include
```

详见：

【已解决】Mac中pip安装pycurl报错：fatal error openssl/ssl.h file not found

启动报错：Error Could not create web server listening on port 25555

- 现象：运行 `pyspider` 时能看到有错误
 - `Error: Could not create web server listening on port 25555`
- 原因：之前已启动过 `pyspider`，其内部会默认启动 `phantomjs`，而虽然之前虽然已关闭掉 `pyspider`，但是没有杀掉 `phantomjs` 的进程，导致端口 25555 被占用，而报错
- 解决办法：杀掉端口是 25555 的 `phantomjs` 进程即可
- 步骤：
 - 找 `phantomjs` 进程ID:
 - `ps aux | grep 25555`
 - 杀掉对应进程
 - `kill -9 xxx`
- 举例

```
* limao@xxx ~ /dev/crifan/python/demo_spider ps aux | g
limao          35620  0.0  0.0  4277272   820 s002  R+
limao          33983  0.0  0.4  6130968  34128 s002  S
limao@xxx ~ /dev/crifan/python/demo_spider kill -9 339
```

之后即可正常启动 `pyspider`，且能看到 `phantomjs` 可以正常启动了：

```
phantomjs fetcher running on port 25555
```

启动报错：Deprecated option domaincontroller use http_authenticator.domain_controller instead

启动报错：

```
File "/Users/limao/.pyenv/versions/3.6.8/lib/python3.6/s:
    raise ValueError("Invalid configuration:\n - " + "\n
ValueError: Invalid configuration:
 - Deprecated option 'domaincontroller': use 'http_authent
```

- 原因: wsgidav 版本兼容问题
- 解决办法: 换兼容的没问题的旧版本 2.4.1
- 步骤:

```
pip install wsgidav==2.4.1
```

启动报错: ImportError cannot import name DispatcherMiddleware

启动报错:

```
File "/Users/limao/.pyenv/versions/3.6.8/lib/python3.6/s:
    from werkzeug.wsgi import DispatcherMiddleware
ImportError: cannot import name 'DispatcherMiddleware'
```

- 原因: werkzeug 版本兼容问题
- 解决办法: 换兼容的没问题的旧版本 0.16.1
- 步骤:

```
pip install werkzeug==0.16.1
```

PySpider基本用法

使用PySpider的基本步骤

下面来介绍一下PySpider的使用的步骤和操作：

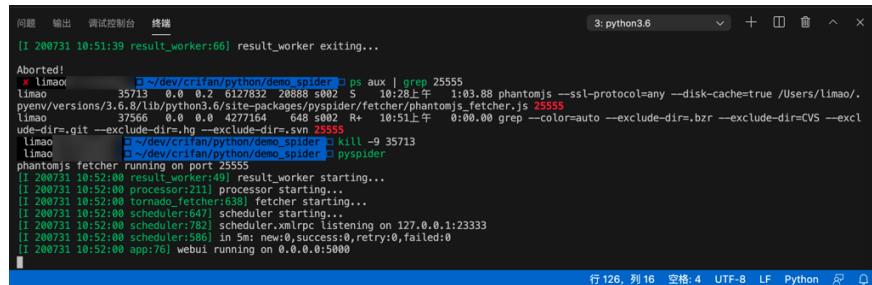
运行PySpider

在某个目录下的终端命令行中输入

```
pyspider
```

即可启动运行，输出举例：

```
□ pyspider
phantomjs fetcher running on port 25555
[I 200731 10:28:35 result_worker:49] result_worker starting...
[I 200731 10:28:35 processor:211] processor starting...
[I 200731 10:28:35 tornado_fetcher:638] fetcher starting...
[I 200731 10:28:35 scheduler:647] scheduler starting...
[I 200731 10:28:35 scheduler:782] scheduler.xmlrpc listening...
[I 200731 10:28:35 scheduler:586] in 5m: new:0,success:0,retry:0,failed:0
[I 200731 10:28:35 app:84] webui exiting...
```



A screenshot of a terminal window titled '3: python3.6'. The window shows the output of a PySpider run. It starts with '[I 200731 10:51:39 result_worker:66] result_worker exiting...'. Then it lists several processes being killed: 'kill -9 35713', 'kill -9 37566', 'kill -9 4277164', and 'kill -9 25555'. Following these, it shows the startup of various components: '[I 200731 10:52:00 result_worker:146] result_worker starting...', '[I 200731 10:52:00 processor:211] processor starting...', '[I 200731 10:52:00 tornado_fetcher:638] fetcher starting...', '[I 200731 10:52:00 scheduler:647] scheduler starting...', '[I 200731 10:52:00 scheduler:782] scheduler.xmlrpc listening on 127.0.0.1:23333', '[I 200731 10:52:00 scheduler:586] in 5m: new:0,success:0,retry:0,failed:0', and '[I 200731 10:52:00 app:76] webui running on 0.0.0.0:5000'.

注：

- 如果是用虚拟环境安装的PySpider，记得先进入虚拟环境后再运行PySpider
 - 比如用的 pipenv，则是

```
pipenv shell
pyspider
```

- pyspider 等价于 pyspider all

进入 WebUI

安装和启动的常见问题

然后去用浏览器打开：

<http://0.0.0.0:5000/>

即可进入爬虫的 管理界面 = WebUI

新建爬虫项目

点击 Create , 去新建一个爬虫项目

输入：

- 爬虫名称：
- 入口地址：自动生成的代码中，会作为起始要抓取的url
 - 也可以不填
 - 后续可以在代码中修改

然后再点击新建的爬虫项目，进入调试页面

新建出来的项目，默认状态是 TODO

点击新建出来的项目名，直接进入调试界面

然后右边是编写代码的区域

左边是调试的区域，用于执行代码，显示输出信息等用途

调试爬虫代码

编写代码，调试输出信息，保存代码

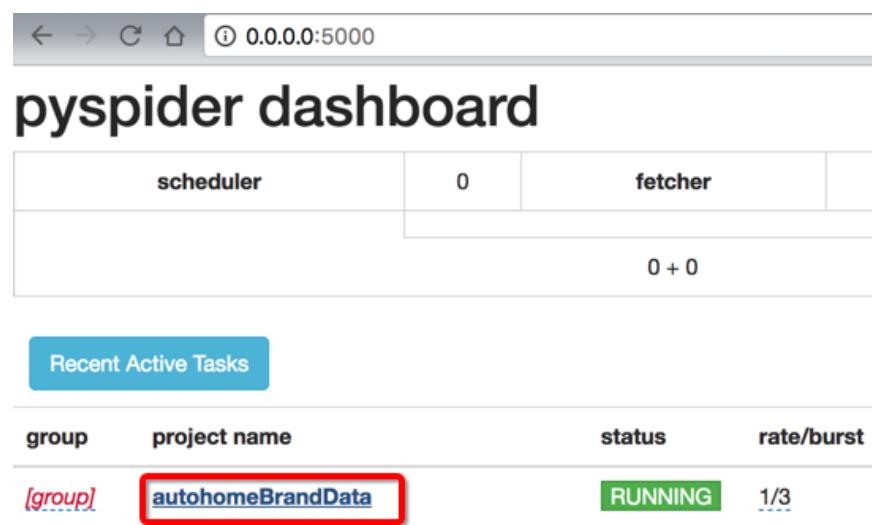
调试代码期间，对于想要返回上一级：

先说之前不熟悉的时候的操作：

之前调试运行时，不知道还有回到上一级，在想要返回上一级时，都直接是点击左上角的项目名字



返回项目列表：



然后重新进去，重新点击Run，直到跑到对应的层级，去继续调试。

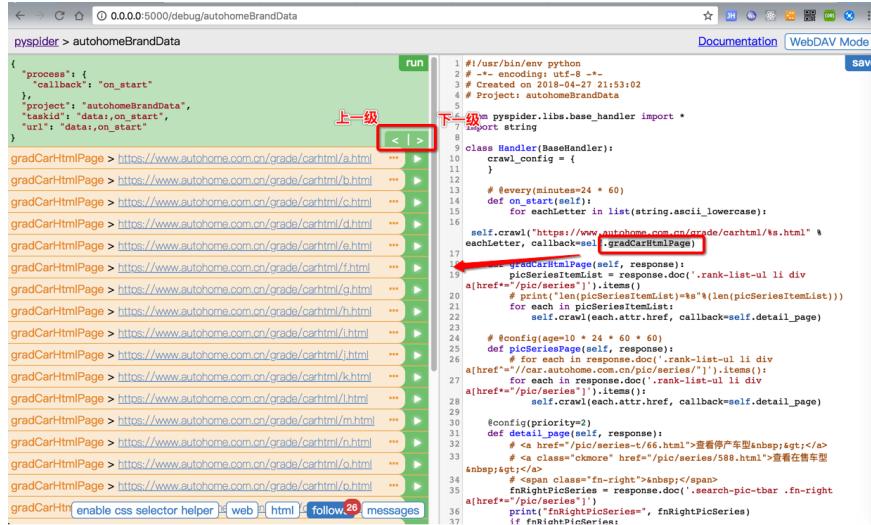
再说后来知道了PySpider内置支持这种逻辑操作：

PySpider对在调试期间所需要在上一个连接和下一个连接之间切换的操作，支持的很好：

点击 < | > 的 < 或 >，则可以 返回上一级 或 进入下一级

实际效果演示：

安装和启动的常见问题

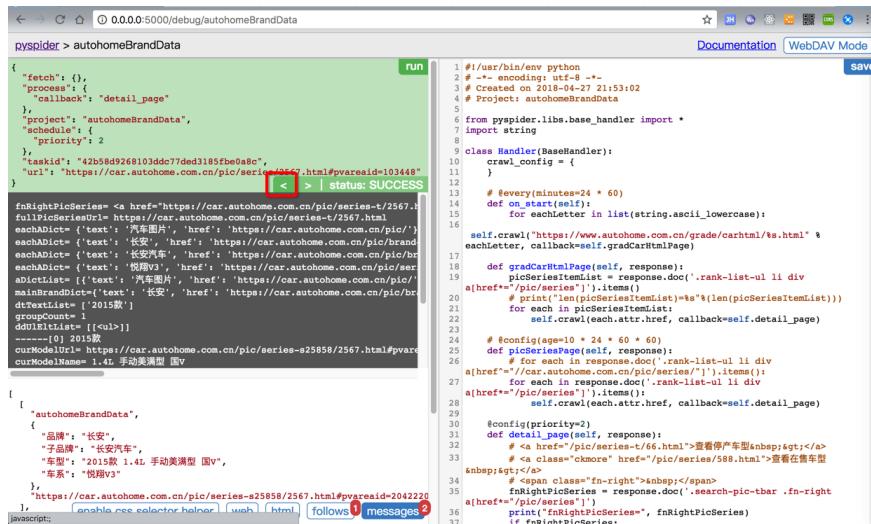


```
pySpider > autohomeBrandData
{
    "process": {
        "callback": "on_start"
    },
    "project": "autohomeBrandData",
    "taskid": "datai_on_start",
    "url": "datai_on_start"
}
}

gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/a.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/b.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/c.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/d.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/e.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/f.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/g.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/h.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/i.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/j.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/k.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/l.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/m.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/n.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/o.html ...
gradCarHtmlPage > https://www.autohome.com.cn/grade/carhtml/p.html ...
gradCarHn[enable css selector helper] web [html] follow 28 messages
```

```
run Documentation WebDAV Mode
1 #!/usr/bin/env python
2 # -*- encoding: utf-8 -*-
3 # Created on 2018-04-27 21:53:02
4 # Project: autohomeBrandData
5
6 from pySpider.libs.base_handler import *
7 import string
8
9 class Handler(BaseHandler):
10     crawl_config = {
11         'allow_domains': [
12             'www.autohome.com.cn'
13         ],
14         'interval': 24 * 60
15     }
16
17     def gradCarHtmlPage(self, response):
18         picSeriesItemList = response.doc('.rank-list-ul li div a[href^="/picseries/"]').items()
19         # print(len(picSeriesItemList) * "%(len(picSeriesItemList))")
20         for each in picSeriesItemList:
21             self.crawl("https://www.autohome.com.cn/grade/carhtml/%s.html" % eachLetter, callback=self.gradCarHtmlPage)
22
23     # @config(page=10 + 24 * 60 + 60)
24     def picSeriesPage(self, response):
25         picSeriesItemList = response.doc('.rank-list-ul li div a[href^="/picseries/"]').items()
26         # print(len(picSeriesItemList) * "%(len(picSeriesItemList))")
27         for each in response.doc('.rank-list-ul li div a[href^="/picseries/"]').items():
28             self.crawl(each.attr.href, callback=self.detail_page)
29
30     # @config(priority=2)
31     def detail_page(self, response):
32         # <a href="/picseries-166.html">查看停产车型<br></a>
33         # <a class="ckmore" href="/picseries/588.html">查看在售车型<br></a>
34         # <span class="fn-right"><br></span>
35         fnRightPicSeries = response.doc('.search-pic-bar .fn-right a[href^="/picseries"]')
36         print("fnRightPicSeries", fnRightPicSeries)
37         if fnRightPicSeries:
```

想要返回上一级的爬取函数的话，点击 左箭头



```
pySpider > autohomeBrandData
{
    "fetch": {},
    "process": {
        "callback": "detail_page"
    },
    "project": "autohomeBrandData",
    "schedule": {
        "priority": 2
    },
    "taskid": "42b58d9268103ddc77ded3185fbba8c",
    "url": "https://car.autohome.com.cn/pic/series-103448.html#pvareaid=103448"
}

fnRightPicSeries <- href="https://car.autohome.com.cn/pic/series-t/2567.j
fullList <- href="https://car.autohome.com.cn/pic/t/2567.j
eachBjctc <- href="https://car.autohome.com.cn/pic/t/2567.j
eachBjctc <- href="https://car.autohome.com.cn/pic/brand
eachBjctc <- href="https://car.autohome.com.cn/pic/ser
eachBjctc <- href="https://car.autohome.com.cn/pic/ser
abictList <- href="https://car.autohome.com.cn/pic/ser
mainbrandict <- href="https://car.autohome.com.cn/pic/ser
dtTextList <- href="https://car.autohome.com.cn/pic/ser
groupList <- href="https://car.autohome.com.cn/pic/ser
d0ULList <- href="https://car.autohome.com.cn/pic/ser
-----[0] 2015款
curModelUrl= https://car.autohome.com.cn/pic/series-a25858/2567.html#pv
curModelName= 1.4L 手动精英型 [IV]

[
    {
        "autohomeBrandData": [
            {
                "品牌": "长安",
                "子品牌": "长安汽车",
                "车型": "2015款 1.4L 手动精英型 国V",
                "车系": "悦翔V3"
            }
        ],
        "https://car.autohome.com.cn/pic/series-a25858/2567.html#pvareaid=204220
    },
    {
        "autohomeBrandData": [
            {
                "brand": "长安汽车"
            }
        ],
        "https://car.autohome.com.cn/pic/series-a25858/2567.html#pvareaid=204220
    }
]
}

[enable css selector helper] web [html] follows 1 messages
```

```
run Documentation WebDAV Mode
1 #!/usr/bin/env python
2 # -*- encoding: utf-8 -*-
3 # Created on 2018-04-27 21:53:02
4 # Project: autohomeBrandData
5
6 from pySpider.libs.base_handler import *
7 import string
8
9 class Handler(BaseHandler):
10     crawl_config = {
11         'allow_domains': [
12             'www.autohome.com.cn'
13         ],
14         'interval': 24 * 60
15     }
16
17     def gradCarHtmlPage(self, response):
18         picSeriesItemList = response.doc('.rank-list-ul li div a[href^="/picseries/"]').items()
19         # print(len(picSeriesItemList) * "%(len(picSeriesItemList))")
20         for each in picSeriesItemList:
21             self.crawl("https://www.autohome.com.cn/grade/carhtml/%s.html" % eachLetter, callback=self.gradCarHtmlPage)
22
23     # @config(page=10 + 24 * 60 + 60)
24     def picSeriesPage(self, response):
25         picSeriesItemList = response.doc('.rank-list-ul li div a[href^="/picseries/"]').items()
26         # print(len(picSeriesItemList) * "%(len(picSeriesItemList))")
27         for each in response.doc('.rank-list-ul li div a[href^="/picseries/"]').items():
28             self.crawl(each.attr.href, callback=self.detail_page)
29
30     # @config(priority=2)
31     def detail_page(self, response):
32         # <a href="/picseries-166.html">查看停产车型<br></a>
33         # <a class="ckmore" href="/picseries/588.html">查看在售车型<br></a>
34         # <span class="fn-right"><br></span>
35         fnRightPicSeries = response.doc('.search-pic-bar .fn-right a[href^="/picseries"]')
36         print("fnRightPicSeries", fnRightPicSeries)
37         if fnRightPicSeries:
```

然后再点击Run:

安装和启动的常见问题

```
{
    "fetch": {},
    "process": {
        "callback": "gradCarHtmlPage"
    },
    "project": "autohomeBrandData",
    "schedule": {},
    "taskid": "40aa2a3ad7c7a8973043d60a32df38f0",
    "url": "https://www.autohome.com.cn/grade/carhtml/c.html"
}

fnRightPicSeries= <a href="https://car.autohome.com.cn/pic/series-t/2567.html" target="blank">2015款</a>
fullPicSeriesUrl= https://car.autohome.com.cn/pic/series-t/2567.html
eachADict= {'text': '汽车图片', 'href': 'https://car.autohome.com.cn/pic/'}
eachADict= {'text': '长安', 'href': 'https://car.autohome.com.cn/pic/brand/'}
eachADict= {'text': '长安汽车', 'href': 'https://car.autohome.com.cn/pic/brand/car/'}
eachADict= {'text': '悦翔v3', 'href': 'https://car.autohome.com.cn/pic/series-s/25858.html'}
aDictList= [{"text": "汽车图片", "href": "https://car.autohome.com.cn/pic/brand/car/"}]
mainBrandDict= {'text': '长安', 'href': 'https://car.autohome.com.cn/pic/brand/car/'}
dtTextList= ['2015款']
groupCount= 1
ddUlElList= [<ul>]
-----[0] 2015款
curModelUrl= https://car.autohome.com.cn/pic/series-s/25858/2567.html#pvareaid=2042220
curmodelName= 1.4L 手动美满型 国V
```

[
[
{"autohomeBrandData":
{
"品牌": "长安",
"子品牌": "长安汽车",
"车型": "2015款 1.4L 手动美满型 国V",
"车系": "悦翔v3"
},
"https://car.autohome.com.cn/pic/series-s/25858/2567.html#pvareaid=2042220"
],
[
"autohomeBrandData":
]

[enable css selector helper](#) [web](#) [html](#) [follows 1](#) [messages 2](#)

然后就可以返回上一级了。

然后也才注意到，每行的follow的左边开始显示的是：callback函数名

此处的是 detail_page

```
self.crawl("https://www.autohome.com.cn/grade/carhtml/t/s.html" t
eachLetter, callback=self.gradCarHtmlPage)
17     def gradCarHtmlPage(self, response):
18         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='pic/series/']).items()
19             # print(len(picSeriesItemList)) *len(picSeriesItemList))
20             for each in picSeriesItemList:
21                 self.crawl(each.attr.href, callback=self.detail_page)
22
23     # configuration=10 + 24 + 60 * 60
24     def detail_page(self, response):
25         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='//car.autohome.com.cn/pic/series/']).items()
26             for each in response.doc('.rank-list-ul li div
a[href*='pic/series/']).items():
27                 self.crawl(each.attr.href, callback=self.detail_page)
28
29     @config(priority=2)
30     def detail_page(self, response):
31         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='//car.autohome.com.cn/pic/series/']).items()
32             # print(len(picSeriesItemList)) *len(picSeriesItemList))
33             for each in picSeriesItemList:
34                 self.crawl(each.attr.href, callback=self.detail_page)
35
36     # configuration=10 + 24 + 60 * 60
37     def detail_page(self, response):
38         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='//car.autohome.com.cn/pic/series/']).items()
39             for each in response.doc('.rank-list-ul li div
a[href*='pic/series/']).items():
40                 self.crawl(each.attr.href, callback=self.detail_page)
41
42     # configuration=10 + 24 + 60 * 60
43     def detail_page(self, response):
44         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='//car.autohome.com.cn/pic/series/']).items()
45             for each in response.doc('.rank-list-ul li div
a[href*='pic/series/']).items():
46                 self.crawl(each.attr.href, callback=self.detail_page)
47
48     # configuration=10 + 24 + 60 * 60
49     def detail_page(self, response):
50         fnRightPicSeries= response.doc('.rank-list-ul li div
a[href*='//car.autohome.com.cn/pic/series/']).items()
51             for each in response.doc('.rank-list-ul li div
a[href*='pic/series/']).items():
52                 self.crawl(each.attr.href, callback=self.detail_page)
```

[enable css selector helper](#) [web](#) [html](#) [follow 101](#) [messages](#)

而对应的上一级的结果中，也是上一级的callback：

安装和启动的常见问题

The screenshot shows the PySpider IDE interface. On the left, a list of crawled URLs for 'gradCarHtmlPage' is displayed, including links like <https://www.autohome.com.cn/grade/carhtml/a.html>, <https://www.autohome.com.cn/grade/carhtml/b.html>, etc. On the right, the source code for the task is shown:

```
1 #!/usr/bin/env python
2 # -*- encoding: utf-8 -*-
3 # Created on 2018-04-27 21:53:02
4 # Project: autohomeBrandData
5
6 from pyspider.libs.base_handler import *
7 import string
8
9 class Handler(BaseHandler):
10     crawl_config = {
11         ...
12     }
13
14     # every(minutes=24 * 60)
15     def on_start(self):
16         for eachLetter in list(string.ascii_lowercase):
17             self.crawl("https://www.autohome.com.cn/grade/carhtml/%s.html" % eachLetter, callback=self.gradCarHtmlPage)
18
19     def gradCarHtmlPage(self, response):
20         a[href="#/pic/series/"].items()
21         # print(len(picSeriesItemList)*len(picSeriesItemList))
22         for eachLetter in list(string.ascii_lowercase):
23             self.crawl(each.attr.href, callback=self.detail_page)
24
25         # config(page=10 * 24 * 60 * 60)
26         def picSeriesPage(self, response):
27             a[href="#/car/autohome.com.cn/pic/series/"].items()
28             for each in response.doc('.rank-list-ul li div a[href="#/pic/series/"]').items():
29                 self.crawl(each.attr.href, callback=self.detail_page)
30
31         # config(priority=2)
32         def detail_page(self, response):
33             <a href="#/pic/series-t/66.html">查看停产车型</a>
34             <a href="#/pic/series-t/588.html">查看在售车型</a>
35             # a class="chmore" href="#/pic/series/588.html"
36             a[href="#/pic/series/"]
37             if fnRightPicSeries:
38                 <span class="fn-right">&ampnbsp</span>
39             fnRightPicSeries = response.doc('.search-pic-bar .fn-right')
40             print("fnRightPicSeries=", fnRightPicSeries)
41             if fnRightPicSeries:
```

运行爬虫去爬取数据

调试完毕后，返回项目，status改为DEBUG或RUNNING，点击Run

想要暂停运行：status改为STOP

保存已爬取的数据

当爬取完毕数据，需要保存下来时，可以有多种保存方式：

- mysql数据库
- MongoDB数据库
- CSV或Excel文件

保存到csv或Excel文件

基本思路：确保自己代码中，最后return返回的字段是你要的字段

如何得到CSV文件：在任务运行期间或完毕后，去 Results -> 点击下载 CSV，即可得到你要的csv格式的数据文件。

结果：PySpider会自动在已有字段中加上额外的 url 字段

用VSCode编辑csv文件

- 如果想要去除多余的不需要的 url 字段，则可以通过文本编辑器，比如 VSCode 去列编辑模式，批量删除，或者查找和替换，都可以实现
- 最后会多余一列，标题是 ...，内容全是 {}，所以直接用编辑器比如VSCode去替换为空以清空，即可

详见：

[【已解决】PySpider如何把json结果数据保存到csv或excel文件中 – 在路上](#)

Excel去打开CSV文件结果乱码

csv文件编码默认为UTF8（是好事，通用的），但是如果用（不论是Mac还是Win中的）excel去打开，结果（估计对于中文系统，都是）会默认以GKB（或GB18030）打开，所以会乱码

解决办法：[【已解决】Mac或Win中用Excel打开UTF8编码的csv文件显示乱码](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-04-14 20:26:27

项目操作举例

此处就以一个简单的项目为例来说明，从头到尾是如何运行和操作的：

首次运行 pypider，会提示是否运行使用网络，点击 允许



输出：

```
→ ChildQuPeiYinApp_downloadDemo pypider
phantomjs fetcher running on port 25555
[I 180925 11:25:51 result_worker:49] result_worker starting...
[I 180925 11:25:51 processor:211] processor starting...
[I 180925 11:25:51 tornado_fetcher:638] fetcher starting...
[I 180925 11:25:52 scheduler:647] scheduler starting...
[I 180925 11:25:52 scheduler:782] scheduler.xmlrpc listening on 127.0.0.1:23333
[I 180925 11:25:52 scheduler:586] in 5m: new:0,success:0,retry:0,failed:0
[I 180925 11:25:52 app:76] webui running on 0.0.0.0:5000
```

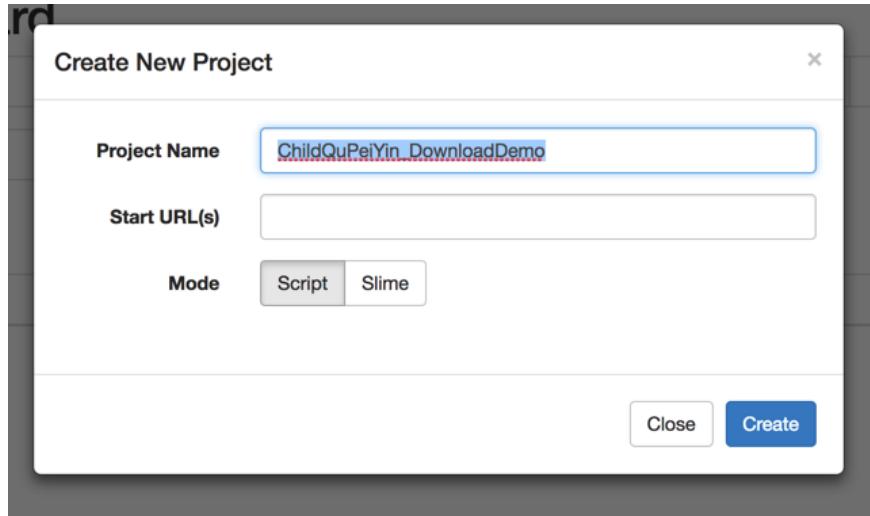
打开页面：

<http://0.0.0.0:5000>

The screenshot shows the pypider dashboard interface. At the top, there's a header bar with browser controls and a status message "不安全 | 0.0.0.0:5000". Below the header is the title "pypider dashboard". The main area has four cards showing metrics: "scheduler" (0), "fetcher" (0), "processor" (0), and "result_worker" (0). Below these cards is a section titled "Recent Active Tasks" with a table. The table has columns: group, project name, status, rate/burst, avg time, progress, and actions. There are no active tasks listed in the table.

去 [Create New Project](#) 新建项目

安装和启动的常见问题



进入项目调试界面：

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2018-09-25 11:27:00
# Project: ChildQuPeiYin_DownloadDemo
# Configuration file for the project.

from pyspider.libs.base_handler import *

class Handler(BaseHandler):
    crawl_config = {
        ...
    }

    @config(age=10 * 24 * 60 * 60)
    def on_start(self):
        self.crawl(_START_URL_, callback=self.index_page)

    def index_page(self, response):
        for each in response.doc('a[href^="http"]').items():
            self.crawl(each.attr.href, callback=self.detail_page)

    def detail_page(self, response):
        return {
            "url": response.url,
            "title": response.doc('title').text(),
        }

```

编写代码，或者已写好代码后去粘贴代码，再点击保存：

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2018-09-25 14:40:51
# Project: ChildQuPeiYinApp
# Configuration file for the project.

import copy
import os
import codecs
import json
from datetime import datetime, timedelta
from pyspider.libs.base_handler import *
# OutputFolder =
# "/Users/crifan/dev/dev_root/company/naturling/projects/crawler_projects/crawler_childqupeiyin_app/output"
OutputFolder =
"/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output"

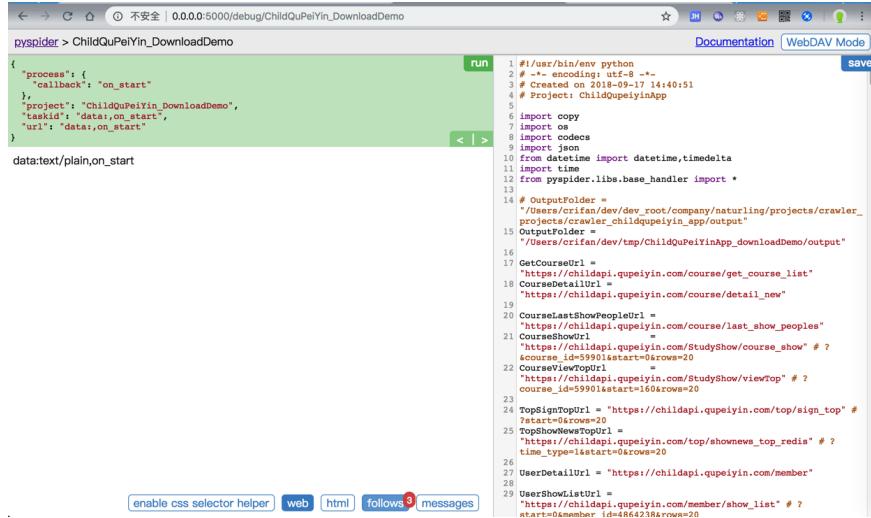
GetCourseUrl =
"https://childapi.qupeiyin.com/course/get_course_list"
CourseDetailUrl =
"https://childapi.qupeiyin.com/course/detail_new"
CourseLastShowPeopleUrl =
"https://childapi.qupeiyin.com/course/last_show_peoples"
CourseShowDetail =
"https://childapi.qupeiyin.com/studyshow/course_show" # ?
&course_id=59901&start=0&rows=20
CourseViewTop =
"https://childapi.qupeiyin.com/studyShow/viewTop" # ?
course_id=59901&start=160&rows=20
TopSignUrl =
"https://childapi.qupeiyin.com/top/sign_top" # ?
&start=0&rows=20
TopShowNewsTopUrl =
"https://childapi.qupeiyin.com/top/shownews_top_redis" # ?
time_type=last&start=0&rows=20
UserDetailUrl =
"https://childapi.qupeiyin.com/member/show_list" # ?
start=0&member_id=486423&rows=20

```

接着点击 Run ，开始运行。

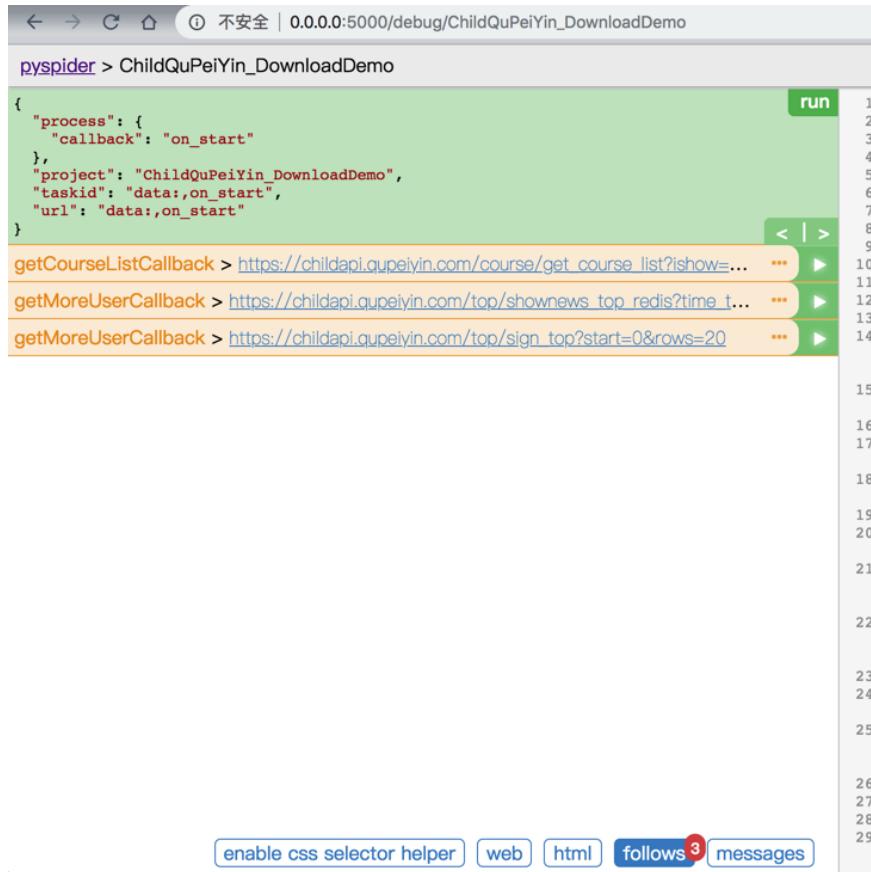
会出现 Follow ，点击 Follow

安装和启动的常见问题



```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2018-09-17 14:40:51
# Project: ChildQuPeiYinApp
#
# import copy
# import os
# import codecs
# import time
# from datetime import datetime, timedelta
# import time
# from pyppider.libs.base_handler import *
#
# OutputFolder =
# "/Users/crifan/dev/dev_root/company/naturling/projects/crawler_
# projects/crawler_childQuPeiYin_app/output"
# OutputFolder =
# "/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output"
#
# GetCourseUrl1 =
# "https://childapi.gupeiyin.com/course/get_course_list"
# CourseDetailUrl1 =
# "https://childapi.gupeiyin.com/course/detail_new"
# CourseLastShowPeopleUrl1 =
# "https://childapi.gupeiyin.com/course/last_show_peoples"
# CourseShowUrl1 =
# "https://childapi.gupeiyin.com/studyshow/course_show" ?
# start=0&rows=20
# CourseViewTopUrl1 =
# "https://childapi.gupeiyin.com/studyshow/viewtop" ?
# course_id=59901&start=160&rows=20
# TopSignTopUrl1 =
# "https://childapi.gupeiyin.com/top/sign_top" ?
# start=0&rows=20
# TopShowNewsTopUrl1 =
# "https://childapi.gupeiyin.com/top/shownews_top_redis" ?
# time_type=latest&start=0&rows=20
# UserDetailUrl1 =
# "https://childapi.gupeiyin.com/member"
# UserShowListUrl1 =
# "https://childapi.gupeiyin.com/member/show_list" ?
# start=0&member_id=486423&rows=20
```

其中Follow后的 3，指的是有产生了3条请求链接，可供后续继续访问



```
{ "process": { "callback": "on_start" }, "project": "ChildQuPeiYin_DownloadDemo", "taskid": "data:,on_start", "url": "data:,on_start" }
```

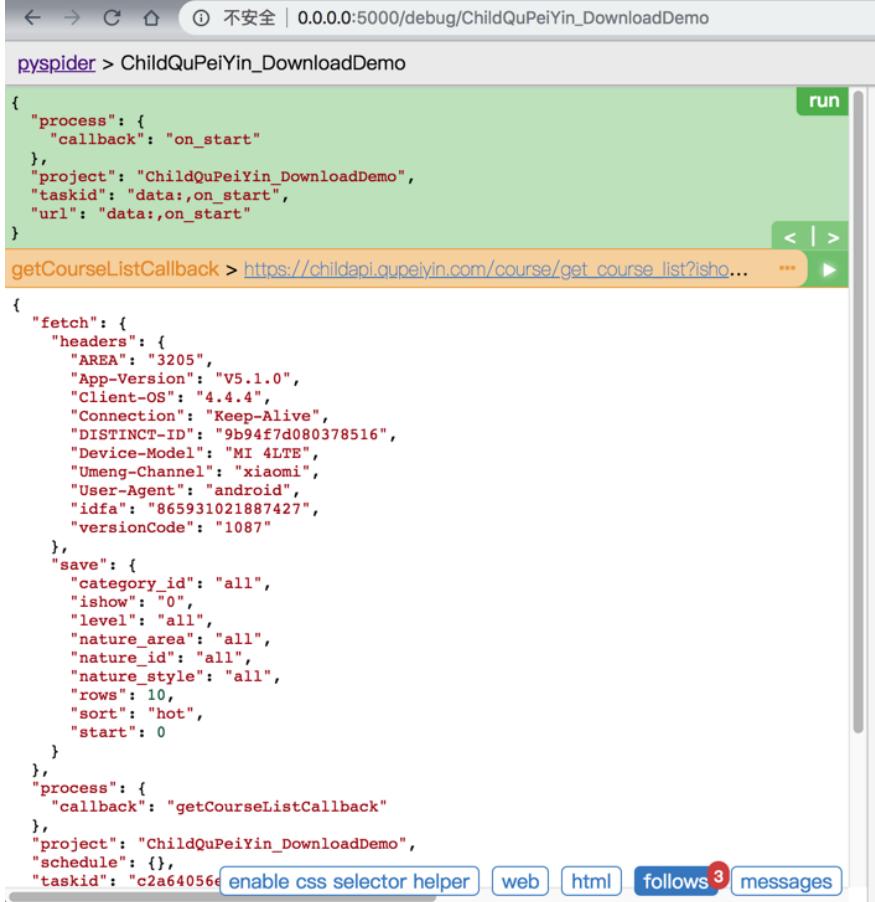
getCourseListCallback > https://childapi.gupeiyin.com/course/get_course_list?ishow=... ... ▶

getMoreUserCallback > https://childapi.gupeiyin.com/top/shownews_top_redis?time_t... ... ▶

getMoreUserCallback > https://childapi.gupeiyin.com/top/sign_top?start=0&rows=20 ... ▶

点击第一个的 三个点，去展开：

安装和启动的常见问题



The screenshot shows the PySpider IDE interface. At the top, there's a browser-like header with icons for back, forward, refresh, and a URL bar showing "0.0.0.0:5000/debug/ChildQuPeiYin_DownloadDemo". Below the header, the title "pyspider > ChildQuPeiYin_DownloadDemo" is displayed. The main area contains a code editor with the following Python code:

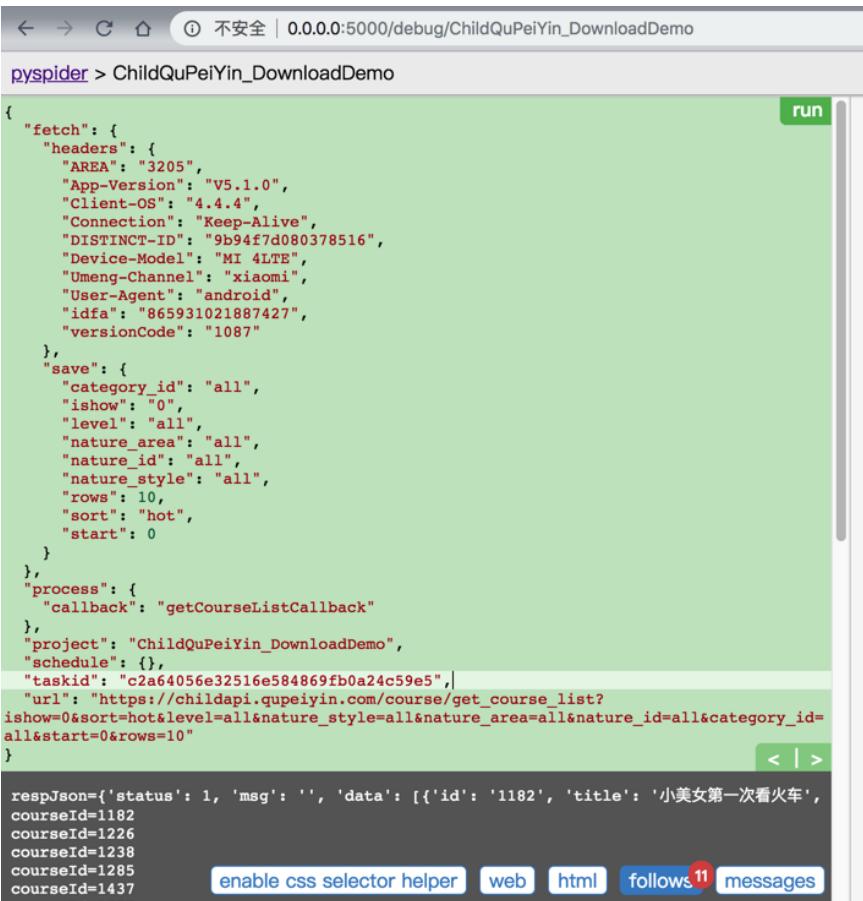
```
{  
    "process": {  
        "callback": "on_start"  
    },  
    "project": "ChildQuPeiYin_DownloadDemo",  
    "taskid": "data:on_start",  
    "url": "data:on_start"  
}  
  
{  
    "fetch": {  
        "headers": {  
            "AREA": "3205",  
            "App-Version": "V5.1.0",  
            "Client-OS": "4.4.4",  
            "Connection": "Keep-Alive",  
            "DISTINCT-ID": "9b94f7d080378516",  
            "Device-Model": "MI 4LTE",  
            "Umeng-Channel": "xiaomi",  
            "User-Agent": "android",  
            "idfa": "865931021887427",  
            "versionCode": "1087"  
        },  
        "save": {  
            "category_id": "all",  
            "ishow": "0",  
            "level": "all",  
            "nature_area": "all",  
            "nature_id": "all",  
            "nature_style": "all",  
            "rows": 10,  
            "sort": "hot",  
            "start": 0  
        }  
    },  
    "process": {  
        "callback": "getCourseListCallback"  
    },  
    "project": "ChildQuPeiYin_DownloadDemo",  
    "schedule": {},  
    "taskid": "c2a64056c enable css selector helper web html follows 3 messages"  
}
```

At the bottom right of the code editor, there are several buttons: "run", "< | >", "...", and a green arrow button. Below the code editor, there's a status bar with buttons for "enable css selector helper", "web", "html", "follows 3", and "messages".

可以看到当前请求的详细参数

点击 右箭头 = >

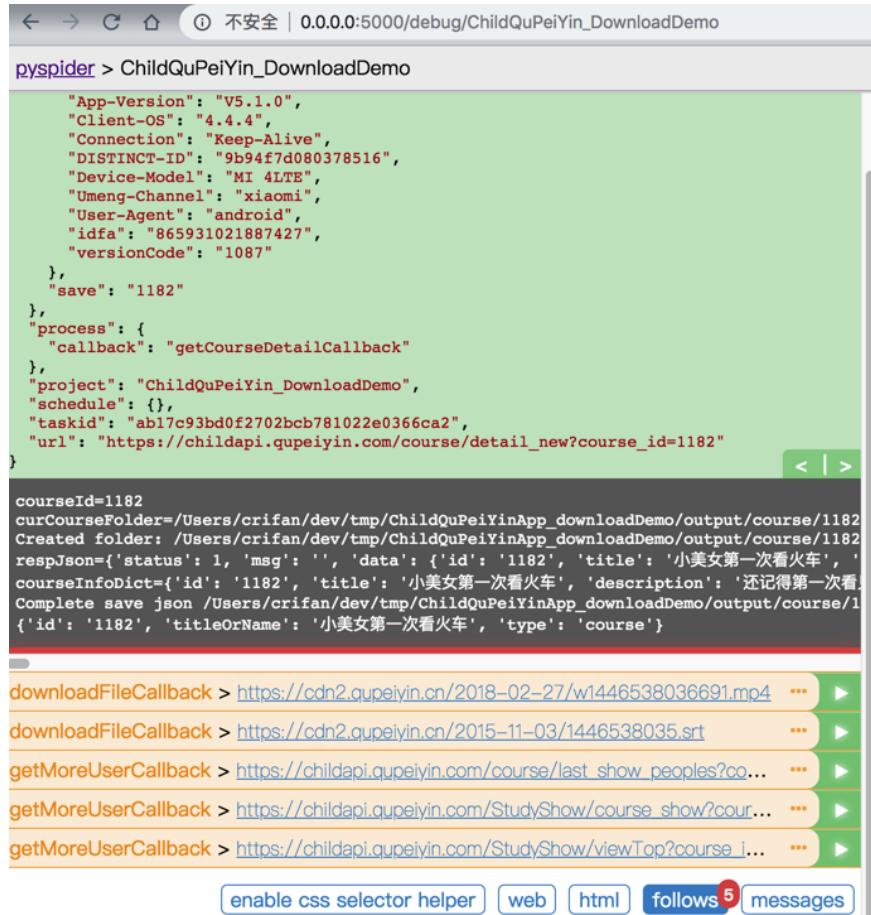
安装和启动的常见问题



```
{  
    "fetch": {  
        "headers": {  
            "AREA": "3205",  
            "App-Version": "V5.1.0",  
            "Client-OS": "4.4.4",  
            "Connection": "Keep-Alive",  
            "DISTINCT-ID": "9b94f7d080378516",  
            "Device-Model": "MI 4LTE",  
            "Umeng-Channel": "xiaomi",  
            "User-Agent": "android",  
            "idfa": "865931021887427",  
            "versionCode": "1087"  
        },  
        "save": {  
            "category_id": "all",  
            "ishow": "0",  
            "level": "all",  
            "nature_area": "all",  
            "nature_id": "all",  
            "nature_style": "all",  
            "rows": 10,  
            "sort": "hot",  
            "start": 0  
        },  
        "process": {  
            "callback": "getCourseListCallback"  
        },  
        "project": "ChildQuPeiYin_DownloadDemo",  
        "schedule": {},  
        "taskid": "c2a64056e32516e584869fb0a24c59e5",  
        "url": "https://childapi.gupeiyin.com/course/get_course_list?  
ishow=0&sort=hot&level=all&nature_style=all&nature_area=all&nature_id=all&category_id=  
all&start=0&rows=10"  
    }  
}  
  
respJson={'status': 1, 'msg': '', 'data': [{'id': '1182', 'title': '小美女第一次看火车',  
courseId=1182  
courseId=1226  
courseId=1238  
courseId=1285  
courseId=1437  
courseId=1527  
courseId=1607  
courseId=1608  
courseId=1645  
courseId=1692  
curPageParamDict={'category_id': 'all', 'ishow': '0', 'level': 'all', 'nature_area': '  
nextPageParamDict={'category_id': 'all', 'ishow': '0', 'level': 'all', 'nature_area': '  
nextPageParamDict={'category_id': 'all', 'ishow': '0', 'level': 'all', 'nature_area': '  
getCourseDetailCallback > https://childapi.gupeiyin.com/course/detail_new?course... ... ▶  
getCourseListCallback enable css selector helper ▶ web ▶ html ▶ follows (11) messages
```

可以看到输出

然后再点击后续的链接，去运行：



The screenshot shows a browser window with the URL `0.0.0.0:5000/debug/ChildQuPeiYin_DownloadDemo`. The page title is `pyspider > ChildQuPeiYin_DownloadDemo`. The content of the page is a JSON dump of a Python script's execution. The script performs the following steps:

- Sets headers including `App-Version`, `Client-OS`, `Connection`, `DISTINCT-ID`, `Device-Model`, `Ueng-Channel`, `User-Agent`, `idfa`, and `versionCode`.
- Creates a `process` object with a `callback` of `getCourseDetailCallback`.
- Defines a `project` of `ChildQuPeiYin_DownloadDemo`, a `schedule` object, a `taskId` of `ab17c93bd0f2702bcb781022e0366ca2`, and a `url` of `https://childapi.qupeiyin.com/course/detail_new?course_id=1182`.
- Logs the course ID `courseId=1182` and the creation of a folder at `/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182`.
- Shows the `respJson` object containing the course details.
- Logs the completion of saving the JSON file.
- Logs the course info dictionary.
- Logs the course details again.
- Lists several download links:
 - `downloadFileCallback > https://cdn2.qupeiyin.cn/2018-02-27/w1446538036691.mp4`
 - `downloadFileCallback > https://cdn2.qupeiyin.cn/2015-11-03/1446538035.srt`
 - `getMoreUserCallback > https://childapi.qupeiyin.com/course/last_show_peoples?co...`
 - `getMoreUserCallback > https://childapi.qupeiyin.com/StudyShow/course_show?cour...`
 - `getMoreUserCallback > https://childapi.qupeiyin.com/StudyShow/viewTop?course_i...`

At the bottom of the browser window, there are buttons for `enable css selector helper`, `web`, `html`, `follows 5 messages`, and `messages`.

安装和启动的常见问题

The screenshot shows a browser window with the URL `0.0.0.0:5000/debug/ChildQuPeiYin_DownloadDemo`. The page title is `pyspider > ChildQuPeiYin_DownloadDemo`. A green button labeled `run` is visible in the top right corner. The main content is a JSON object representing a download configuration:

```
{  
    "fetch": {  
        "headers": {  
            "AREA": "3205",  
            "App-Version": "v5.1.0",  
            "Client-OS": "4.4.4",  
            "Connection": "Keep-Alive",  
            "DISTINCT-ID": "9b94f7d080378516",  
            "Device-Model": "MI 4LTE",  
            "Umeneng-Channel": "xiaomi",  
            "User-Agent": "android",  
            "idfa": "865931021887427",  
            "versionCode": "1087"  
        },  
        "save": {  
            "filename": "course_1182_video.mp4",  
            "saveFolder":  
                "/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182"  
        }  
    },  
    "process": {  
        "callback": "downloadFileCallback"  
    },  
    "project": "ChildQuPeiYin_DownloadDemo",  
    "schedule": {},  
    "taskid": "63759fcb5bd030a752bd4b3975115652",  
    "url": "https://cdn2.qupeiyin.cn/2018-02-27/w1446538036691.mp4"  
}
```

At the bottom of the JSON object, there is a red highlighted section containing the following code:

```
fileInfo={'filename': 'course_1182_video.mp4', 'saveFolder': '/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182/course_1182_video.mp4'}  
fileFullPath=/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182/course_1182_video.mp4  
Complete save file /Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182/course_1182_video.mp4
```

Below the JSON object, there are several small buttons: `enable css selector helper`, `web`, `html`, `follows`, and `messages`.

点击 `<` 返回上一级:

安装和启动的常见问题

```
pyspider > ChildQuPeiYin_DownloadDemo
{
    "fetch": {
        "headers": {
            "AREA": "3205",
            "App-Version": "V5.1.0",
            "Client-OS": "4.4.4",
            "Connection": "Keep-Alive",
            "DISTINCT-ID": "9b94f7d080378516",
            "Device-Model": "MI 4LTE",
            "Umeng-Channel": "xiaomi",
            "User-Agent": "android",
            "idfa": "865931021887427",
            "versionCode": "1087"
        },
        "save": {
            "filename": "course_1182_video.mp4",
            "saveFolder": "/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182"
        }
    },
    "process": {
        "callback": "downloadFileCallback"
    },
    "project": "ChildQuPeiYin_DownloadDemo",
    "schedule": {},
    "taskId": "63759fcb5bd030a752bd4b3975115652",
    "url": "https://cdn2.qupeiyin.cn/2018-02-27/w1446538036691.mp4"
}
fileInfo={'filename': 'course_1182_video.mp4', 'saveFolder': '/Users/crifan/dev/tmp/Chil
fileFullPath=/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182/cou
Complete save file /Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/118
< >
run
javascrit;
```

enable css selector helper web html follows messages

再点击 Run

```
pyspider > ChildQuPeiYin_DownloadDemo
{
    "fetch": {
        "headers": {
            "AREA": "3205",
            "App-Version": "V5.1.0",
            "Client-OS": "4.4.4",
            "Connection": "Keep-Alive",
            "DISTINCT-ID": "9b94f7d080378516",
            "Device-Model": "MI 4LTE",
            "Umeng-Channel": "xiaomi",
            "User-Agent": "android",
            "idfa": "865931021887427",
            "versionCode": "1087"
        },
        "save": "1182"
    },
    "process": {
        "callback": "getCourseDetailCallback"
    },
    "project": "ChildQuPeiYin_DownloadDemo",
    "schedule": {},
    "taskId": "ab17c93bd0f2702bcb781022e0366ca2",
    "url": "https://childapi.qupeiyin.com/course/detail_new?course_id=1182"
}
fileInfo={'filename': 'course_1182_video.mp4', 'saveFolder': '/Users/crifan/dev/tmp/Chil
fileFullPath=/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182/cou
Complete save file /Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/118
< | >
run
enable css selector helper web html follows messages
```

回到上一级的输出了：

The screenshot shows the PySpider interface with the following details:

- Code Area:** The code block contains Python-like pseudocode for a spider named "ChildQuPeiYin_DownloadDemo". It includes sections for "fetch", "process", and "project". The "fetch" section defines headers and saves the response. The "process" section handles course detail callbacks. The "project" section specifies the project name, task ID, URL, and schedule.
- Output Area:** The output shows the execution results:

```
courseId=1182
curCourseFolder=/Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182
Created folder: /Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182
respJson={'status': 1, 'msg': '', 'data': {'id': '1182', 'title': '小美女第一次看火车', 'courseInfoDict': {'id': '1182', 'title': '小美女第一次看火车', 'description': '还记得第一次看
Complete save json /Users/crifan/dev/tmp/ChildQuPeiYinApp_downloadDemo/output/course/1182
{'id': '1182', 'titleOrName': '小美女第一次看火车', 'type': 'course'}}
```
- Log Area:** The log area shows several requests and their URLs:
 - downloadFileCallback > <https://cdn2.gupeiyin.cn/2018-02-27/w1446538036691.mp4>
 - downloadFileCallback > <https://cdn2.gupeiyin.cn/2015-11-03/1446538035.srt>
 - getMoreUserCallback > https://childapi.gupeiyin.com/course/last_show_peoples?co...
 - getMoreUserCallback [enable css selector helper] web html follows 5 messages

如此，即可，根据需要去，反复的：

- 点击某个请求的 Run ，进入下一级
- 点击 < 返回上一级

去调试，直到得到你需要的结果，即可完成。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook 最后更新： 2021-04-14 23:02:46

PySpider中查找提取元素

PySpider中内置的用于查找和定位html网页中元素的库是： PyQuery

PyQuery 算是一个 css选择器， 模拟 JS 领域的 jQuery ， 所以叫做 PyQuery 。

具体细节是， PySpider针对html的响应 response ， 默认提供了一个 doc 属性， 其内置了 PyQuery 解析后结果， 所以你可以用 response.doc("your_css_selector") 去选择你要的html中的内容了。

而具体的your_css_selector的写法，则就变成 PyQuery 的写法了。

举例： 提取汽车之家车型车系相关数据

下面通过想象的例子来解释， PyQuery 的常见用法：

比如想要提取：

```
<ul class="rank-list-ul" 0="">
  <li id="s3170">
    ...
    <div>
      ...
      <a id="atk_3170" href="//car.autohome.com.cn/pic/seri:
    ...
  </div>
</li>
...
</ul>
```

中的 href 的值

- 语言描述可以是： class 是 rank-list-ul 的 ul 元素， 下面的 li ， 下面的 div ， 下面的 a ， 且 href 值是包含 /pic/series 的
- 转换成PyQuery的语法是
 - .rank-list-ul li div a[href*="/pic/series"]
 - 也可以换成另外的写法：
 - ul[class='rank-list-ul'] li div
 - a[href*="/pic/series"]
 - ul[class='rank-list-ul'] a[href*="/pic/series"]

- 如果你确定此规则不会误匹配其他元素，也可以省略中间的节点的查找
- 对应PySpider的代码是：

- 获取匹配到的第一个元素

```
firstMatchADoc = response.doc('.rank-list-ul li d
```

- 获取到所有匹配到的元素

```
for eachADoc in response.doc('.rank-list-ul li di  
print("eachADoc=%s" % eachADoc)
```

以及对于：

```
<dl id="33" olr="6">
  <dt><a href="//car.autohome.com.cn/price/brand-33.html#pvareaid=103459" 
    src="//car2.autoimg.cn/cardfs/series/g26/M0B/AE/B3/
  <div><a href="//car.autohome.com.cn/price/brand-33.htm"
</dt>
<dd>

  <div class="h3-tit"><a href="//car.autohome.com.cn/pric...
  <ul class="rank-list-ul" 0>

    <li id="s3170">
      <h4><a href="//www.autohome.com.cn/3170/#levelsources"
        <div>指导价: <a class="red" href="//www.autohome.com...
        <div><a href="//car.autohome.com.cn/price/series-3170-
          href="//car.autohome.com.cn/pic/series/3170.htm...
          class="js-che168link" href="//www.che168.com/c...
          href="//club.autohome.com.cn/bbs/forum-c-3170-1...
          href="//k.autohome.com.cn/3170/#pvareaid=103459...
        </div>
      </li>

      <li id="s692">
        <h4><a href="//www.autohome.com.cn/692/#levelsources"
          <div>指导价: <a class="red" href="//www.autohome.com...
          <div><a href="//car.autohome.com.cn/price/series-692-
            href="//car.autohome.com.cn/pic/series/692.htm...
            class="js-che168link" href="//www.che168.com/c...
            href="//club.autohome.com.cn/bbs/forum-c-692-1...
            href="//k.autohome.com.cn/692/#pvareaid=103459...
          </div>
      </li>
    ...
    ...
    ...
  
```

对应的代码是：

想要从

```
<a href="//car.autohome.com.cn/price/brand-33.html#pvareaid=103459" 
  src="//car2.autoimg.cn/cardfs/series/g26/M0B/AE/B3/
```

获取brand的logo的 img 的代码：

```
brandDoc = response.doc('dl dt')
brandLogoDoc = brandDoc.find('a img')
brandLogoUrl = brandLogoDoc.attr["src"]
```

从：

```
<div><a href="//car.autohome.com.cn/price/brand-33.html#pv&
```

中获取brand的name的 a 的代码:

```
brandNameDoc = brandDoc.find('div a')
brandName = brandNameDoc.text()
```

从:

```
<div class="h3-tit"><a href="//car.autohome.com.cn/price/bi
```

获取merchant的所有的 a 的代码:

```
merchantDocGenerator = response.doc("dd div[class='h3-tit']")
merchantDocList = list(merchantDocGenerator)
merchantDocLen = len(merchantDocList)
```

注意: `.items()` 返回的是 `generator`, 想要得到 `list`, 需要用 `list(yourGenerator)` 去转换得到

从:

```
<ul class="rank-list-ul" >
  ...
</ul>
```

获取 `rank-list-ul` 的 `class` 的 `dd` 下面的 `ul` 的merchant的代码:

```
merchantRankDocGenerator = response.doc("dd ul[class='rank-list-ul']")
merchantRankDocList = list(merchantRankDocGenerator)
merchantRankDocListLen = len(merchantRankDocList)
```

以及获取每个元素:

- 属性值: 用 `attr`
 - 类型是: `dict`
- 字符串值: 用 `text()`
 - 类型是: `str`

举例:

```
for curIdx, merchantItem in enumerate(merchantDocList):
    merchantName = merchantItem.text()
    merchantItemAttr = merchantItem.attr
    merchantUrl = merchantItemAttr["href"]
```

PyQuery资料

`response.doc` 返回后的 PyQuery对象，之后可以继续用PyQuery去操作

此处列出PyQuery的一些典型的操作函数：

- `PyQuery.filter(selector)`
- `PyQuery.find(selector)`
- `PyQuery.items(selector=None)`
- `PyQuery.siblings(selector=None)`

另外，常见的一些属性来说：

- `PyQuery.text(value=<NoDefault>)`：当前节点的text文本值
- `PyQuery.html(value=<NoDefault>, **kwargs)`：当前节点的html值

详见：

- 官网文档
 - [pyquery – PyQuery complete API — pyquery 1.2.4 documentation](#)
 - [Traversing — pyquery 1.2.4 documentation](#)
 - [Attributes — pyquery 1.2.4 documentation](#)
 - [CSS — pyquery 1.2.4 documentation](#)
- 独立教程
 - [HTML解析库Python版jQuery：PyQuery](#)

PySpider的高级用法

下面介绍PySpider中，除了基本用法之外的，可以算作是高级，稍微更加复杂一些的用法。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2020-07-30 14:00:04

self.crawl详解

官方文档

此处要介绍的 PySpider 的获取网络请求的函数是: `self.crawl`, 功能很强大。

具体详细的解释, 可以参考:

- 官网的英文文档:
 - 比如: [crawl参数 params](#)
- 某热心网友整理的 中文文档:
 - [self.crawl - pyspider中文文档 - pyspider中文网](#)

给 GET 的请求添加 查询参数

给 `self.crawl` 中给 `params` 传递对应字典变量, PySpider 内部会自动把字典编码为url的查询参数 query string.

官方实例:

```
self.crawl('http://httpbin.org/get', callback=self.callback)
```

等价于:

```
self.crawl('http://httpbin.org/get?a=123&b=c', callback=self.callback)
```

自己之前用的例子有:

```
topSignTopParam = {
    "start": 0,
    "rows": 20
}
self.crawl(TopSignTopUrl,
           callback=self.getMoreUserCallback,
           params=topSignTopParam,
           save={
               "baseUrl": TopSignTopUrl,
               "isNeedCheckNextPage": True,
               "curPageParam": topSignTopParam
           }
)
```

给callback函数加上额外的参数

使用 self.crawl 的 save 参数即可，然后callback中用 response.save 获取传入的值

举例：

```
def getUserDetail(self, userId):
    self.crawl(UserDetailUrl,
               callback=self.userDetailCallback,
               params={"member_id": userId },
               save=userId
    )

def userDetailCallback(self, response):
    userId = response.save
    print("userId=%s" % userId)
```

当请求出错时也执行callback回调函数

需要给callback回调函数加上修饰符 @catch_status_code_error

举例：

```
def picSeriesPage(self, response):
    ...
    self.crawl(curSerieDict["url"], callback=self.carModelSpecPage)

@catch_status_code_error
def carModelSpecPage(self, response):
    curSerieDict = response.save
    print("curSerieDict=%s", curSerieDict)
    ...
    return curSerieDict
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2020-07-30 14:00:04

独立的配置文件: config.json

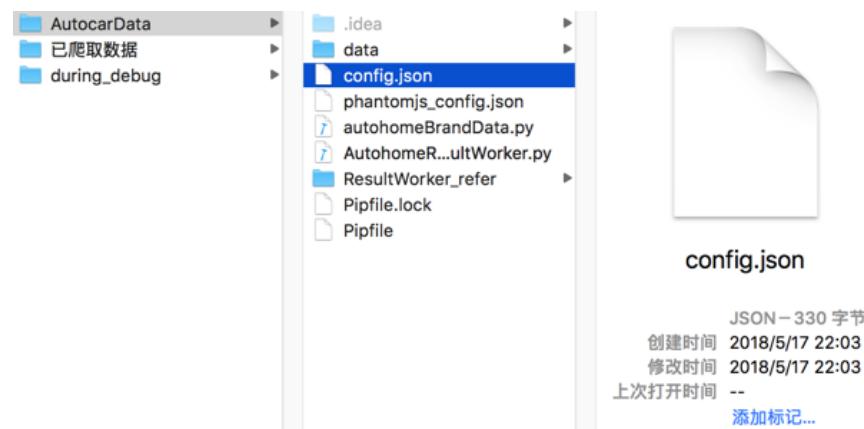
如果需要用到复杂一点的配置, 比如 `result worker`, 则是需要单独写配置文件。

PySpider的配置文件一般叫做 `config.json`

比如用如下内容:

```
{
    "taskdb":      "mysql+taskdb://root:crifan_mysql@127.0.0.1:3306/taskdb?charset=utf8",
    "projectdb":   "mysql+projectdb://root:crifan_mysql@127.0.0.1:3306/projectdb?charset=utf8",
    "resulldb":    "mysql+resulldb://root:crifan_mysql@127.0.0.1:3306/resulldb?charset=utf8",
    "result_worker": {
        "result_cls": "AutohomeResultWorker.AutohomeResultWorker"
    }
}
```

将 `config.json` 保存在 `pyspider` 命令运行所在的当前目录下:



然后去 `-c` 指定配置文件:

```
pyspider -c config.json
```

举例

一个配置更简单的 `config.json`

```
{  
    "webui": {  
        "port": 5000,  
        "need-auth": false  
    },  
    "scheduler": {  
        "delete_time": 30  
    }  
}
```

运行调用效果：

```
□ pyspider -c use_python_spider_framework/config.json  
[I 200731 14:48:06 result_worker:49] result_worker starting...  
phantomjs fetcher running on port 25555  
[I 200731 14:48:06 processor:211] processor starting...  
[I 200731 14:48:06 scheduler:647] scheduler starting...  
[I 200731 14:48:06 tornado_fetcher:638] fetcher starting...  
[I 200731 14:48:06 scheduler:782] scheduler.xmlrpc listening  
[I 200731 14:48:06 scheduler:126] project crawlBaiduHotList  
[I 200731 14:48:06 scheduler:965] select crawlBaiduHotList  
[I 200731 14:48:06 scheduler:586] in 5m: new:0,success:0,re  
[I 200731 14:48:06 tornado_fetcher:188] [200] crawlBaiduHot  
[D 200731 14:48:06 project_module:145] project: crawlBaidu  
[I 200731 14:48:06 processor:202] process crawlBaiduHotList  
[I 200731 14:48:06 scheduler:360] crawlBaiduHotList_PySpider  
[I 200731 14:48:06 app:76] webui running on 0.0.0.0:5000
```

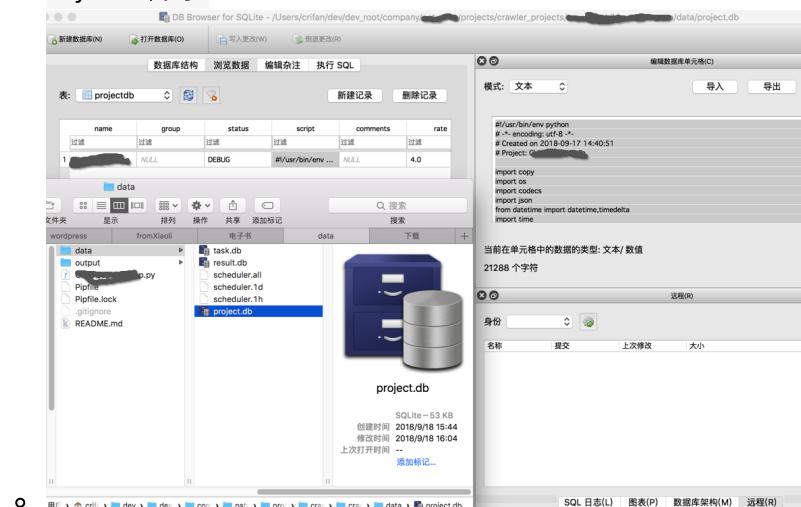
```
limao@: ~/dev/crifan/python/demo_spider] 3 pyspider -c use_python_spider_framework/config.json  
[I 200731 14:48:06 result_worker:49] result_worker starting...  
phantomjs fetcher running on port 25555  
[I 200731 14:48:06 processor:211] processor starting...  
[I 200731 14:48:06 scheduler:647] scheduler starting...  
[I 200731 14:48:06 tornado_fetcher:638] fetcher starting...  
[I 200731 14:48:06 scheduler:782] scheduler.xmlrpc listening on 127.0.0.1:23333  
[I 200731 14:48:06 scheduler:126] project crawlBaiduHotList_PySpider_0731_1436 updated, status:DEBUG, paused:False, 0 tasks  
[I 200731 14:48:06 scheduler:965] select crawlBaiduHotList_PySpider_0731_1436:_on_get_info data:,_on_get_info  
[I 200731 14:48:06 scheduler:586] in 5m: new:0,success:0,retry:0,failed:0  
[I 200731 14:48:06 tornado_fetcher:188] [200] crawlBaiduHotList_PySpider_0731_1436:_on_get_info data:,_on_get_info 0s  
[D 200731 14:48:06 project_module:145] project: crawlBaiduHotList_PySpider_0731_1436 updated.  
[I 200731 14:48:06 processor:202] process crawlBaiduHotList_PySpider_0731_1436:_on_get_info data:,_on_get_info --> [200] len:12 -> result:None  
folio: msg:0 err:None  
[I 200731 14:48:06 scheduler:360] crawlBaiduHotList_PySpider_0731_1436 on_get_info {'min_tick': 0, 'retry_delay': {}, 'crawl_config': {}}  
[I 200731 14:48:06 app:76] webui running on 0.0.0.0:5000
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-04-14 20:55:10

PySpider所在目录下的 data 目录

在你运行 `pyspider` 后，自动会在命令执行路径下生成 `data` 文件夹，其中包含几个（SQLite）文件：

- `project.db`：保存了用户的爬虫项目相关信息，包括项目的 Python 代码
 - 比如用（SQLite）工具去查看，可以看到详细数据
 - 比如 Mac 中的 DB Browser for SQLite 查看的效果：
 - Python 代码：



- 对应数据库结构字段：



- `result.db`：项目运行的结果数据
- `task.db`：项目相关的任务信息
 - 其中如果开始运行爬虫，还会出现相关的调度信息：
 - `scheduler.all`, `scheduler.1d`, `scheduler.1h`：保存了任务执行后所有，1天，1小时内相关的信息，和 WebUI 中的 `progress` 中的 `all`, `1d`, `1h` 对应：

status	rate/burst	avg time	progress	actions
TODO	4/10	5m	1h: 22020 1d: all: 1369505	<button>Run</button> <button>Active Tasks</button> <button>Results</button>

指定data目录

用data-path参数

- 方法1：配置 config.json 中的 data-path

```
{  
    "data-path": "/root/xxx/crawler/pyspider/data",  
    "webui": {  
        "port": 7700,  
        "username": "admin",  
        "password": "yourPassword",  
        "need-auth": true  
    },  
    "scheduler": {  
        "delete_time": 30  
    }  
}
```

- 方式2：命令行传递参数 --data-path

```
--data-path="your_data_folder_path"
```

crifan.com，使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved，
powered by Gitbook 最后更新：2021-04-14 20:51:12

Phantomjs

如何解决部分页面内部不显示，无法抓取的问题？

折腾 [【已解决】PySpider中页面部分内容不显示 – 在路上](#)，遇到个问题：

页面中的部分内容不显示，所以无法抓取。

经过研究发现，其实是：

这部分不显示的内容，是原网页中通过后续调用js去生成和获取的，所以可以通过：

给 `self.crawl` 添加

```
fetch_type='js'
```

会使得内部调用 `phantomjs`，模拟js，渲染生成页面内容。

从而，此处 `PySpider`，在这种需要显示js加载的页面内容时，可以利用 `phantomjs`。

用了 `phantomjs` 后又出错： `FETCH ERROR HTTP 599 Connection timed out after milliseconds`

后续继续运行，加了 `fetch_type='js'` 的代码，去爬取页面数据，结果遇到了：

[【未解决】pyspider运行出错：FETCH_ERROR HTTP 599 Connection timed out after milliseconds](#)

尝试了多种办法，都无法解决此问题。

所以目前的情况是：

如果加了 `phantomjs`，结果在大量爬取页面期间，又会导致出错 `FETCH_ERROR HTTP 599 Connection timed out after milliseconds`，而暂时找不到解决办法。

给 `Phantomjs` 添加额外参数

之前折腾过：

【未解决】pyspider中如何给phantomjs传递额外参数 – 在路上

基本上没有实现想要的效果。但是可供参考。

phantomjs中的proxy是什么意思

对于pyspider来说， phantomjs-proxy参数指的是：

你另外所运行的 phantomjs 的实例 = host:port

比如：

在一个终端中运行：

```
pyspider phantomjs --port 23450 --auto-restart true
```

然后去另外一个终端中运行pyspider：

```
pyspider -c config.json all
```

其中 config.json 包含了：

```
"phantomjs-proxy": "127.0.0.1:23450"
```

就可以使得此处的pyspider在启动时不另外启动phantomjs了。

而是去在需要用到phantomjs时，去连接本地电脑127.0.0.1的23450端口中的phantomjs去处理，去加载页面了。

而对于phantomjs本身来说：

proxy， 指的是代理， 比如翻墙的代理， 等等。

具体相关设置， 可以参考：

[Command Line Interface | PhantomJS](#)

中的：

- `-proxy=address:port` specifies the proxy server to use (e.g. `-proxy=192.168.1.42:8080`)
- `-proxy-type=[http|socks5|none]` specifies the type of the proxy server (default is http).
- `-proxy-auth` specifies the authentication information for the proxy, e.g. `-proxy-auth=username:password`)

详见： [【已解决】pyspider中phantomjs中的proxy是什么意思 – 在路上](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2020-07-30 14:00:04

PySpider经验与心得

折腾了一些PySpider项目，有些经验和心得，整理如下：

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2020-07-30 14:00:04

PySpider的心得

对于加载更多内容，除了想办法找js或api，
也可以换个其他的思路

问题：想要获取单个页面的更多的内容，一般页面都是向下滚动，加载更多。内部往往是js实现，调用额外的api获取更多数据，加载更多数据。

思路：所以一般往往会去研究和抓包，搞清楚调用的api。但是其实有思路多去看看网页中与之相关的其他内容，往往可以通过其他途径，比如另外有个单独的页面，可以获取我所需要的所有的车型车系的数据。就可以避免非要去研究和抓包api了。

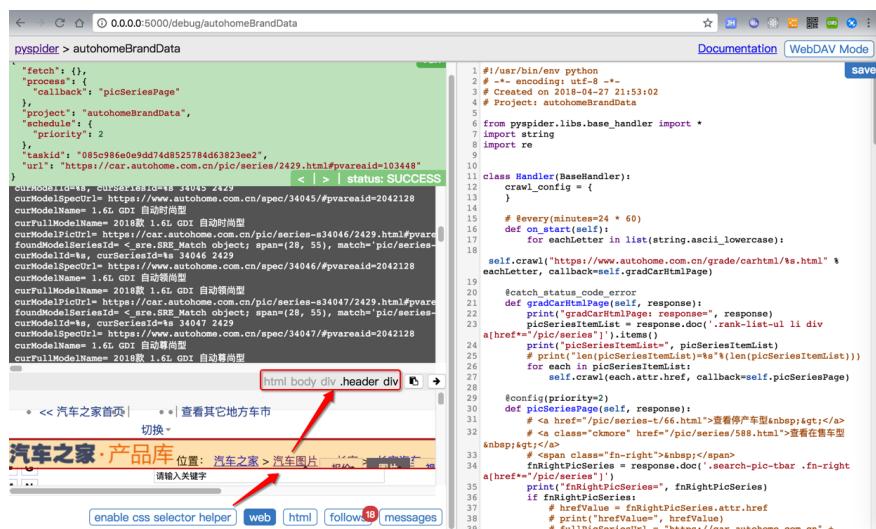
详见：【已解决】pyspider中如何加载汽车之家页面中的更多内容

调试界面中的 enable css selector helper

点击web后可以看到html页面内容

再点击 enable css selector helper 后

之后点击某个页面元素，则可以直接显示出对应的css的selector



不过话说我个人调试页面期间，很少用到。

都是直接去Chrome浏览器中调试页面，查看html源码，寻找合适的css selector。

发送 POST 请求且传递格式 为 application/x-www-form- urlencoded 的 form data 参数

代码：

```
@config(age=10 * 24 * 60 * 60)
def index_page(self, response):
    # <ul class="list-user list-user-1" id="list-user">
    for each in response.doc('ul[id^="list-user"] li'):
        self.crawl(each.attr.href, callback=self.detail_callback)

    maxPageNum = 10
    for curPageIdx in range(maxPageNum):
        curPageNum = curPageIdx + 1
        print("curPageNum=%s" % curPageNum)
        getShowsUrl = "http://xxx/index.php?m=home&action=listUser&page={}&order=1".format(curPageNum)
        headerDict = {
            "Content-Type": "application/x-www-form-urlencoded"
        }
        dataDict = {
            "counter": curPageNum,
            "order": 1,
            "match_type": 2,
            "match_name": "",
            "act_id": 3
        }
        self.crawl(
            getShowsUrl,
            method="POST",
            headers=headerDict,
            data=dataDict,
            cookies=response.cookies,
            callback=self.parseGetShowsCallback
        )

    def parseGetShowsCallback(self, response):
        print("parseGetShowsCallback: self=%s, response=%s" % (self, response))
        respJson = response.json
        print("respJson=%s" % (respJson))
```

实现了：

- 发送 POST
 - 传递header
 - "Content-Type": "application/x-www-form-urlencoded"
 - 传递 data

- 一个 dict , 包含对应的 key 和 value
- 顺带传递了 cookie
 - cookies=response.cookies
- 获得返回的 JSON
 - callback 中用 response.json

无法继续爬取时，注意是否是重复url导致的

当发现没有继续爬取后续数据时，记得想想是不是重复url导致的。

比如此处的：

```
POST /selfReadingBookQuery2
{ "offset": 0, "limit":10}
```

和：

```
POST /selfReadingBookQuery2
{ "offset": 10, "limit":10}
```

虽然 (json参数) 变化了，但是url没变

-> 导致不 (重复) 爬取

解决办法：让每次的url不同

实现方式：比如给url后面加上 #hash 值

举例说明

```
timestampStr = datetime.now().strftime("%Y%m%d_%H%M%S_%f")
curUrlWithHash = curUrl + "#" + timestampStr

self.crawl(curUrlWithHash,
    ...)
```

的：

```
/selfReadingBookQuery2#20190409_162018_413205
/selfReadingBookQuery2#20190409_162117_711811
```

即可实现，每次请求url都不同，就可以继续爬取了。

如果还是不行，或者说，为了更加保险，可以再去加上itag，比如：

```
# add hash value for url to force re-crawl when POST url needed
timestampStr = datetime.now().strftime("%Y%m%d_%H%M%S_%f")
curUrlWithHash = curUrl + "#" + timestampStr

fakeItagForceRecrawl = "%s_%s_%s" % (timestampStr, offset,
                                         self.crawl(curUrlWithHash,
                                         itag=fakeItagForceRecrawl, # To force re-crawl for next
                                         method="POST")
```

当连续多个请求都出现599超时连接后，且尝试retry也都全部失败后，会自动暂停

之前遇到过多次，类似这种：

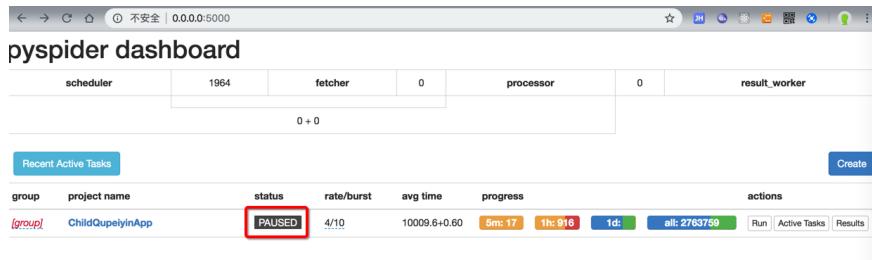
```
Connection timed out after 20000 milliseconds 20.00s
[I 180922 11:25:42 processor:202] process ChildQupeiyinApp:47chb3c0976-76f95608ff6062a7fd03 https://childapi.qupeiyin.com/show/detail?show_id=128683164 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('Connection timed out after 20000 milliseconds')
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:b052697808420792254c93271cf5b4 https://childapi.qupeiyin.com/show/detail?show_id=128683326 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20000 milliseconds')
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:c052e07d8842092d71c54 https://childapi.qupeiyin.com/show/detail?show_id=128683326
[I 180922 11:25:41 scheduler:958] task retry 0/3 ChildQupeiyinApp:3134_994_98_2 ChildQupeiyinApp:3134_994_98_2
[E 180922 11:25:41 tornado_fetcher:212] [599] ChildQupeiyinApp:371b6efcbca7470e7c5d857f30c0 https://childapi.qupeiyin.com/show/detail?show_id=128683458, HTTP 599;
Connection timed out after 20001 milliseconds 20.00s
[E 180922 11:25:41 processor:202] process ChildQupeiyinApp:c052e07d8842092d71c54 https://childapi.qupeiyin.com/show/detail?show_id=128684548 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20001 milliseconds')
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:371b6efcbca7470e7c5d857f30c0 https://childapi.qupeiyin.com/show/detail?show_id=128684548
[E 180922 11:25:42 tornado_fetcher:212] [599] ChildQupeiyinApp:e5941879f798721106eddd365dc5c69 https://childapi.qupeiyin.com/show/detail?show_id=128684578, HTTP 599;
Connection timed out after 20001 milliseconds 20.00s
[E 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:c3e065a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=129410670, HTTP 599;
Connection timed out after 20001 milliseconds 20.00s
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:c3e041879f8721106eddd365dc5c695 https://childapi.qupeiyin.com/show/detail?show_id=129684378 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20001 milliseconds')
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:c3e0d5a2199256898652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=129410670 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20001 milliseconds')
[I 180922 11:25:41 scheduler:959] task retry 0/3 ChildQupeiyinApp:e5041879f8721106eddd365dc5c69 https://childapi.qupeiyin.com/show/detail?show_id=128684378
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=129410670
[E 180922 11:25:42 tornado_fetcher:212] [599] ChildQupeiyinApp:800400e81a217bc05877ff41ee74a05 https://childapi.qupeiyin.com/show/detail?show_id=130095443
Connection timed out after 20000 milliseconds 20.00s
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:800400e81a217bc05877ff41ee74a05 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20000 milliseconds')
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:800400e81a217bc05877ff41ee74a05 https://childapi.qupeiyin.com/show/detail?show_id=130095443
[E 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:3783982a70d6c82b8c30d6106d933d7 https://childapi.qupeiyin.com/show/detail?show_id=130096232, HTTP 599;
Connection timed out after 20000 milliseconds 20.00s
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:3783982a70d6c82b8c30d6106d933d7 https://childapi.qupeiyin.com/show/detail?show_id=130096232 -> [599] len:0
-> result:None fol:0 msg:0 err:Exception('HTTP 599: Connection timed out after 20000 milliseconds')
[I 180922 11:25:43 scheduler:959] task retry 0/3 ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130095443
[I 180922 11:26:08 scheduler:126] project ChildQupeiyinApp updated, status:STOP, paused:True, 1667087 tasks
[C 180922 11:26:13 scheduler:663] scheduler exiting...
[E 180922 11:26:13 tornado_fetcher:671] fetcher exiting...
[I 180922 11:26:13 processor:229] processor exiting...
[I 180922 11:26:13 result_worker:66] result_worker exiting...
```

```
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:3783982a70d6c82b8c30d6106d933d7 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[E 180922 11:25:42 tornado_fetcher:212] [599] ChildQupeiyinApp:3783982a70d6c82b8c30d6106d933d7 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[I 180922 11:25:42 scheduler:959] task retry 0/3 ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[E 180922 11:25:42 tornado_fetcher:212] [599] ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[E 180922 11:25:42 processor:202] process ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[I 180922 11:25:43 scheduler:959] task retry 0/3 ChildQupeiyinApp:c3e0d5a4219925688652ce1e737321 https://childapi.qupeiyin.com/show/detail?show_id=130096232
[I 180922 11:26:08 scheduler:126] project ChildQupeiyinApp updated, status:STOP, paused:True, 1667087 tasks
[C 180922 11:26:13 scheduler:663] scheduler exiting...
[E 180922 11:26:13 tornado_fetcher:671] fetcher exiting...
[I 180922 11:26:13 processor:229] processor exiting...
[I 180922 11:26:13 result_worker:66] result_worker exiting...
```

上述 status:STOP, paused:True 就是表示暂停了。

对应着界面上 status 自动变成 PAUSED

安装和启动的常见问题



The screenshot shows the pyspider dashboard interface. At the top, there is a header bar with icons and the URL '0.0.0.0:5000'. Below it, a table provides an overview of the system's components: scheduler (1964), fetcher (0), processor (0), and result_worker (0 + 0). A 'Recent Active Tasks' section follows, featuring a table with columns: group, project name, status, rate/burst, avg time, progress, and actions. One row is highlighted with a red border around the 'status' column, which displays 'PAUSED'. The 'actions' column contains buttons for Run, Active Tasks, and Results. A 'Create' button is located at the top right of this section.

-> 估计是内部逻辑发现多次是 599 的错误，就自动暂停重试了。避免了后续无效的请求

-> 还是很智能的，因为此处实际上是网络断了，导致无法请求的。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook最后更新：2021-04-14 23:09:48

删除项目

如何清除之前的或正在运行的任务

对于一个写好的爬虫，且已经点击 Run 运行，或者运行了一段时间后，主动停止了。

接着想要去删除之前下载的数据，则：

[官网的解释](#)是：

设置 group 为 delete，以及 status 为 STOP 后，过了(默认) 24 小时 后，会自动删除该项目所有信息。

但是往往没法满足我们需求：

我不想要等待，只想现在就去：删除掉所有的信息，包括之前已经爬取的数据，之前的调度的任务等等数据。

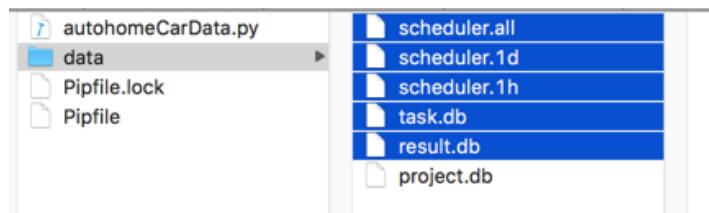
经过一番研究后，发现了解决方案：

- 先去停止项目
 - WebUI中设置 status 为 STOP
 - 终端中用 Control+C 强制停止 pypspider 的运行

```
[ 180428 10:34:47 scheduler[647] scheduler starting...
[ 180428 10:34:47 scheduler[106] project autohomeBrandData updated, status:DEBUG, paused:False, 0 tasks
[ 180428 10:34:47 scheduler[965] select autohomeBrandData:__on_get_info data:__on_get_info
[ 180428 10:34:47 scheduler[580] in Sm new:0,success:0,retry:0,failed:0
[ 180428 10:34:47 scheduler[782] scheduler.xmlrpc listening on 127.0.0.1:23333
[ 180428 10:34:47 tornado_Fetcher[188] [200] autohomeBrandData:__on_get_info data:__on_get_info 0s
[ 180428 10:34:47 app[78] webui running on 0.0.0.0:9300
[ 180428 10:34:47 app[78] project: autohomeBrandData updated
[ 180428 10:34:47 processor[202] process autohomeBrandData:__on_get_info data:__on_get_info -> [200] len:12 -> result:None fail:0 msg:0 err:None
[ 180428 10:34:47 scheduler[364] autohomeBrandData __on_get_info {"min_tick": 0, "retry_delay": 0, "crawl_config": {}}
<[ 180428 10:35:20 app[84] webui exiting...
[ 180428 10:35:20 processor[229] processor exiting...
[ 180428 10:35:20 scheduler[653] scheduler exiting...
[ 180428 10:35:20 result_worker[86] result_worker exiting...
[ 180428 10:35:20 tornado_Fetcher[671] fetcher exiting...
[ 180428 10:35:20 tornado_Fetcher[671] fetcher exiting...
Aborted!
+ AutoCarData
+ AutoCarData
```

- 再去删除文件： result.db 和 task.db
 - 如果还有任务相关的

scheduler.all , scheduler.1d , scheduler.1h ，则一并删除



不要轻易在没备份代码情况下删除project.db

注意不要删除，保存了项目（配置和）代码的： project.db ，否则代码就没了。 (我最开始就这么干过， 😂)

之后去重新运行pyspider，再去刷新WebUI界面：

<http://0.0.0.0:5000/>

即可看到干净的项目，没有了之前的任务和数据了。

指定多久之后删除项目，即指定项目删除等待时间

PySpider中的项目，想要删除：

默认逻辑是，status设置为STOP（如果有group，那么group的status也要设置为delete），再等24小时后，才会自动删除

但是往往我们不想要等待那么久

想要指定删除的时间，则有2种方式去设置参数。

举例说明，比如想要 30秒 后删除，则可以：

- 文件： config.json : 设置 scheduler 的 delete_time 参数

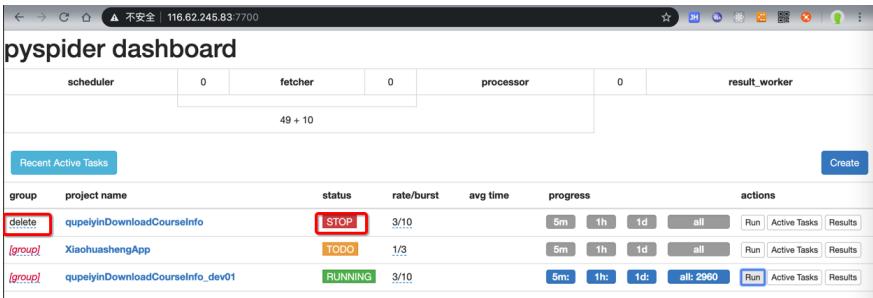
```
"scheduler": {  
    "delete_time": 30  
}
```

- 命令行传入： scheduler.DELETE_TIME

```
pyspider -c config.json scheduler --delete-time=30
```

然后 WebUI 中设置

- status 是 STOP
- group 设置为 delete



Recent Active Tasks						
group	project name	status	rate/burst	avg time	progress	actions
delete	qupeiyinDownloadCourseInfo	STOP	3/10	5m	1h	1d all Run Active Tasks Results
[group]	XiaohuashengApp	TODO	1/3	5m	1h	1d all Run Active Tasks Results
[group]	qupeiyinDownloadCourseInfo_dev01	RUNNING	3/10	5m	1h	1d all: 2960 Run Active Tasks Results

然后过了30秒后，去刷新，该项目就被删除了，看不到了：

安装和启动的常见问题

The screenshot shows the pyspider dashboard interface. At the top, there is a summary table with columns: scheduler (0), fetcher (0), processor (0), and result_worker (0). Below this, a message says "65 + 8". A "Recent Active Tasks" section follows, containing a table with columns: group, project name, status, rate/burst, avg time, progress, and actions. Two projects are listed:

group	project name	status	rate/burst	avg time	progress	actions		
[group]	XiaohuashengApp	TODO	1/3	5m	1h	1d	all	Run Active Tasks Results
[group]	qupeiyinDownloadCourseInfo_dev01	RUNNING	3/10	76.6+134.87	5m:	1h:	1d:	all: 9159 Run Active Tasks Results

A "Create" button is located at the top right of the "Recent Active Tasks" section.

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook最后更新: 2021-04-14 20:49:26

PySpider常见的坑

关于折腾PySpider期间，遇到很多或大或小的坑，常见和具体细节相关的坑，已记录到对应部分中了。

此处再继续整理出，其他的一些常见的坑。

HTTP 599 Operation timed out after milliseconds with out of bytes received

类似现象：

```
after 120001 milliseconds with 0 bytes receive
after 120000 milliseconds with 1723300 out of 2343850 bytes
```

解释：

-> 意思是：超时了（超过设置的最大超时时间了），但是只下载了总共数据的其中一部分

-> 重点是后半句，意思是下载到了数据的，只是直到超时都还没下载完全

-> 这种情况的最大可能原因就是：网速太慢

-» 所以

解决办法

根本办法：换个更快的网络

比如，我公司是 1MB/s 的网络，家里是 10MB/s 的网络，换到家里下载，就不会出现这个问题了

临时的规避的缓解的办法：增大延迟 `timeout` (+增大其他容错参数 `connect_timeout`, `retries`)

- 给单个 `self.crawl` 增大参数

```
self.crawl(urlToDownload,
            callback=self.downloadFileCallback,
            connect_timeout=100,
            timeout=600,
            retries=15,
            save=fileInfo)
```

- 或：增大全局参数

```
class Handler(BaseHandler):
    crawl_config = {
        "connect_timeout": 100,
        "timeout": 600,
        "retries": 15,
    }
```

参数含义解释详见官网：

- self.crawl - pyspider

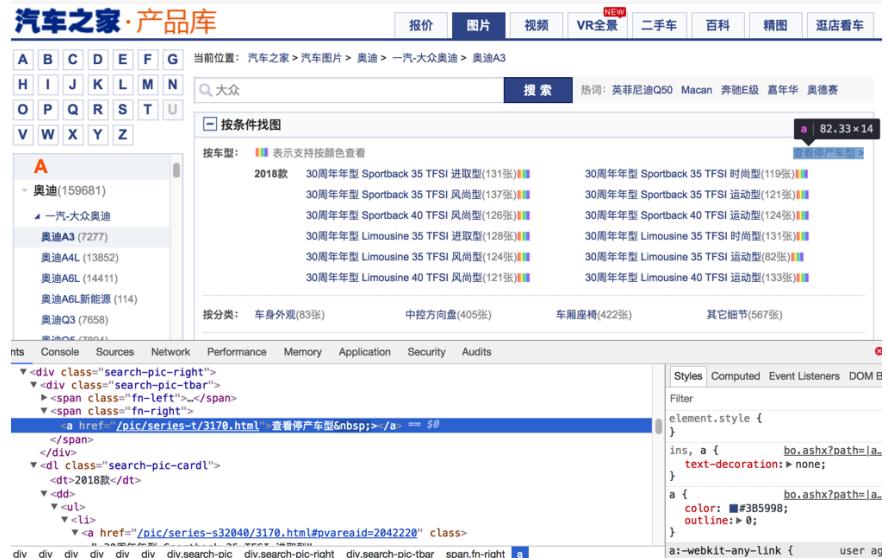
css的选择器不工作

背景：网页中的源码本来是：

```
<a href="//car.autohome.com.cn/pic/series/3170.html#pvarea:
```

或者类似的：

```
href="/pic/series-t/3170.html"
```



The screenshot shows a browser's developer tools with the element inspector open. The element being inspected is an anchor tag with the href attribute set to "/pic/series-t/3170.html". The CSS selector for this element is displayed as "a[href="/pic/series-t/3170.html"]". The right-hand panel of the developer tools shows the computed styles for this element, which include a background color of "#3B5998" and an outline of "0px". The page itself is a car listing page from AutoHome, showing various car models and their details.

然后去写css选择器：

```
a[href^="//car.autohome.com.cn/pic/series/"]
```

但是却无法匹配

原因： PySpider 内部的css选择器用的是 `PyQuery`， 其默认把`href`的路径， 加上了对应的host， 所以此时获取到的html实际上变成了：

```
<a href="https://car.autohome.com.cn/pic/series/3170.html#"
```

详见：

[response.doc](#)

`Reponse.doc()` 返回的就是一个`PyQuery`的对象 **Links have made as absolute by default**

猜测：估计是为了方便小白用户，所以默认加上了host，但是坑了其他人啊。

解决办法：此处被逼的css选择器写法只能改为：

```
a[href*="pic/series/"]
```

或类似的代码：

```
fnRightPicSeries = response.doc('.search-pic-tbar .fn-right').eq(0).text()
fullPicSeriesUrl = fnRightPicSeries.attr.href
```

已经得到的是，加了host/domain的绝对路径了：

```
fullPicSeriesUrl= https://car.autohome.com.cn/pic/series-t...
```

```

fetch: {},
process: {
  "callback": "detail_page"
},
"project": "autohomeCarData",
"schedule": {
  "priority": 2
},
"taskid": "730f08375d51b4feee428e6a73c6ff6",
"url": "https://car.autohome.com.cn/pic/series/3170.html#pvareaid=103448"
}

fnRightPicSeries = <a href="https://car.autohome.com.cn/pic/series-t/3170.html">
fullPicSeriesUrl = https://car.autohome.com.cn/pic/series-t/3170.html
eachDictc = {'text': '汽车图片', 'href': 'https://car.autohome.com.cn/pic/'}
eachDictc['text'] = '奥迪', 'href': 'https://car.autohome.com.cn/pic/brand-3'
eachDictc['text'] = '一汽-大众奥迪', 'href': 'https://car.autohome.com.cn/pic/brand-3'
eachDictc['text'] = '奥迪A3', 'href': 'https://car.autohome.com.cn/pic/series-3'
eachDictc['text'] = '奥迪A3', 'href': 'https://car.autohome.com.cn/pic/series-3'
mainBrandDictc = {'text': '汽车图片', 'href': 'https://car.autohome.com.cn/pic/'}, 
mainBrandDictc['text'] = '奥迪A3', 'href': 'https://car.autohome.com.cn/pic/series-3'
dtTextListc = ['2018款']
groupCountc = 1
ddUlListc = [[{'ul': 1}]] 
-----[0] 2018款
currModelNamec = '30周年年型 Sportback 35 TFSI 进取型'
currFullModelNamec = '2018款 30周年年型 Sportback 35 TFSI 进取型'

detail_page > https://car.autohome.com.cn/pic/series-t/3170.html

```

详见： [【已解决】pyspider中的css选择器不工作 – 在路上](#)

Error Could not create web server listening on port 25555

原因：对应的25555端口被占用了

根本原因：之前的PySpider没有正常的彻底的被关闭，所以残留了。

解决办法：彻底 kill 干掉之前的PySpider的进程即可。

举例：

普通Linux类系统，用：

- 找到占了25555端口的进程的id： `ps aux | grep 25555`
- 再去杀掉进程： `kill process_id -9`

即可。

如果是Mac中，则用 `lsof`

```

→ AutocarData lsof -i:25555
COMMAND      PID    USER      FD      TYPE             DEVICE SIZE,
phantomjs 46971 crifan    12u    IPv4  0xe4d24cdcaf5e481f
→ AutocarData kill 46971

```

PAUSED 后无法立刻继续运行

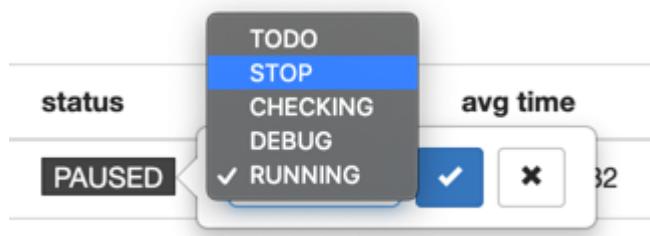
当PySpider在爬取期间发现太多的url都是 `retry` 重试，则会变成 `PAUSED`

安装和启动的常见问题

The screenshot shows the PySpider dashboard interface. At the top, there's a header with tabs like 'scheduler', 'fetcher', 'processor', and 'result_worker'. Below the header, a table lists a single project: 'DianpingChildrenEnglish' with a status of 'PAUSED'. The 'status' column has a dropdown menu open, showing options: 'STOP', 'CHECKING', 'DEBUG', and 'RUNNING'. A 'Create' button is located in the top right corner of the dashboard area.

猜测：其内部有比较智能的判断，推测是断网或者网络异常了，所以暂停下载

通过直接把 PAUSED 改为 STOP



或改为了 RUNNING

status	rate/burst
RUNNING	3/8

但是刷新一下页面，就还是显示PAUSED

-> 不能立刻开始继续下载

-> 往往要等很长时间之后才能继续下载

而如果自己想要立刻继续下载，经过研究，可以：

- 先改为STOP

status	rate/burst
STOP	3/8

◦

- 停止PySpider
 - Control+C停止

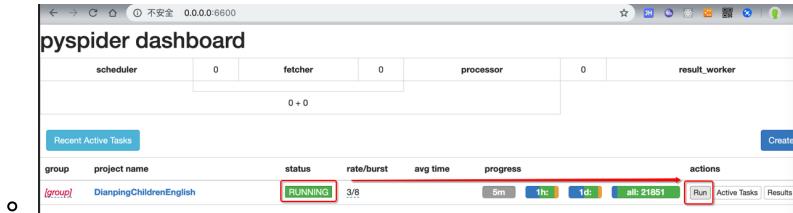
This screenshot shows a terminal window with several lines of log output. The logs indicate that various PySpider components are exiting, such as 'scheduler', 'fetcher', and 'result_worker'. The final line shows the command 'Aborted!' followed by 'crawler_dianping_com git:(master) x'.

```
[1 190428 14:38:47 scheduler:126] project DianpingChildrenEnglish updated, status:STOP, paused:True, 1685 tasks
[1 190428 14:38:57 scheduler:561] dm: new:1,success:1,retry:8,failed:0 DianpingChildrenEnglish:1,1,8,0
[C:\Users\zhang\PycharmProjects\PySpider\dianping\crawler_dianping_com\190428\14:39:01] process exiting...
[1 190428 14:39:01 scheduler:651] scheduler exiting...
[1 190428 14:39:01 tornado_fetcher:671] fetcher exiting...
[1 190428 14:39:01 result_worker:66] result_worker exiting...
Process Process->1:
[1 190428 14:39:02 app:84] webui exiting...
Aborted!
crawler_dianping_com git:(master) x
```

- 重新运行PySpider

```
Aborted!
-> crawler_dianping_com git:(master) ✘ pypspider -c config.json
[I 190428 14:39:16 result_worker:49] result_worker starting...
[I 190428 14:39:17 processor:211] processor starting...
[I 190428 14:39:17 tornado_fetcher:638] fetcher starting...
[I 190428 14:39:17 scheduler:647] scheduler starting...
[I 190428 14:39:17 scheduler:126] project DianpingChildrenEnglish updated, status:STOP, paused:False, 0 tasks
phantomjs fetcher running on port 25555
[I 190428 14:39:17 scheduler:782] scheduler.xmlrpc listening on 127.0.0.1:23333
[I 190428 14:39:17 app:76] webui running on 0.0.0.0:6600
```

- 再改为RUNNING, 点击Run



即可立刻继续运行了

```
输出 调试控制台 终端 1: Python
dianping.com/shop/102333675
[W 190428 14:40:00 tornado_fetcher:423] [403] DianpingChildrenEnglish:302852c5db867afe12ce6811e70ad011 http://www.dianping.com/shop/127627501 0.62s
[E 190428 14:40:00 processor:202] process DianpingChildrenEnglish:302852c5db867afe12ce6811e70ad011 http://www.dianping.com/shop/127627501 > [403] len:0 -> result:None fol:0 msg:0 err:HTTPError('HTTP 403: Forbidden.')
[I 190428 14:40:00 scheduler:959] task retry 0/3 DianpingChildrenEnglish:302852c5db867afe12ce6811e70ad011 http://www.dianping.com/shop/127627501
[I 190428 14:40:01 scheduler:965] select DianpingChildrenEnglish:d7d360fe90985b2cff11d71f8d106938 http://www.dianping.com/shop/110134436#143958_275181
[I 190428 14:40:01 scheduler:965] select DianpingChildrenEnglish:b08cae1b46485c6d17a8d9ad6417c56 http://www.dianping.com/shop/48454988
[I 190428 14:40:01 tornado_fetcher:419] [200] DianpingChildrenEnglish:c3a9d512eda492969a3027b5410994ef http://www.dianping.com/shop/102630687#143957_178132 1.55s
[I 190428 14:40:01 processor:202] process DianpingChildrenEnglish:c3a9d512eda492969a3027b5410994ef http://www.dianping.com/shop/102630687#143957_178132 -> [200] len:40962 -> result:{'shopUrl': fol:0 msg:0 err:None}
[I 190428 14:40:01 result_worker:33] result DianpingChildrenEnglish:c3a9d512eda492969a3027b5410994ef http://www.dianping.com/shop/102630687#143957_178132 -> {'shopUrl': 'http://www.dianping.com/shop/102630687#143957_178132'}
[I 190428 14:40:01 scheduler:966] task done DianpingChildrenEnglish:c3a9d512eda492969a3027b5410994ef http://www.dianping.com/shop/102630687#143957_178132
[I 190428 14:40:01 scheduler:965] select DianpingChildrenEnglish:f2bba1374458ce85daaa3bc5c192a384 http://www.dianping.com/shop/122752579#143959_452287
```

post时data传递dict有时候不行

比如

[crifan/PySpiderChinaProvinceCity](#)

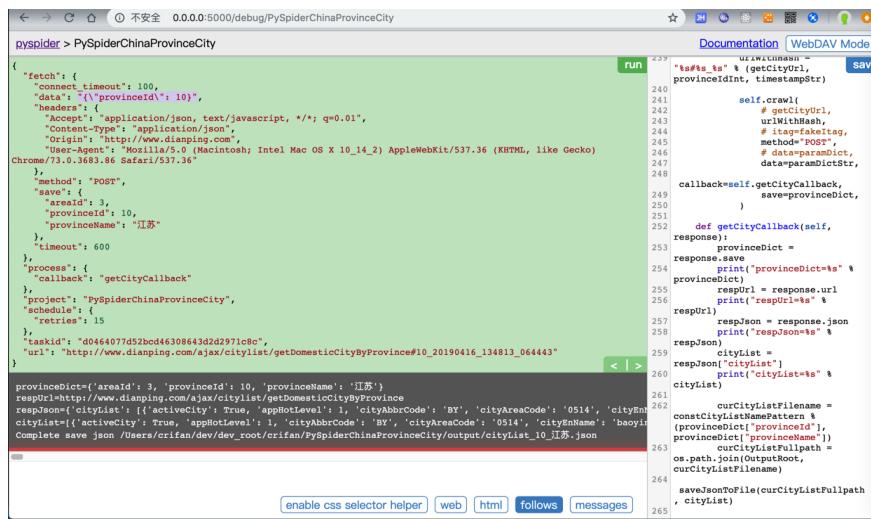
```

getCityUrl = "http://www.dianping.com/ajax/citylist/getDomesticCityList"
for eachProvince in provinceList:
    print("eachProvince=%s" % eachProvince)
    provinceIdInt = eachProvince["provinceId"]
    paramDict = {
        "provinceId": provinceIdInt,
    }
    paramDictStr = json.dumps(paramDict)

    ...
    self.crawl(
        # getCityUrl,
        urlWithHash,
        # itag=fakeItag,
        method="POST",
        # data=paramDict,
        data=paramDictStr,
        callback=self.getCityCallback,
        save=provinceDict,
    )
)

```

data 要传递 json 变量 paramDict 去 json.dumps 后的字符串 paramDictStr 才可以:



如果换成 dict :

```
getCityUrl = "http://www.dianping.com/ajax/citylist/getDomestic"
for eachProvince in provinceList:
    print("eachProvince=%s" % eachProvince)
    provinceIdInt = eachProvince["provinceId"]
    paramDict = {
        "provinceId": provinceIdInt,
    }
    paramDictStr = json.dumps(paramDict)

    ...
    self.crawl(
        # getCityUrl,
        urlWithHash,
        # itag=fakeItag,
        method="POST",
        data=paramDict,
        # data=paramDictStr,
        callback=self.getCityCallback,
        save=provinceDict,
    )
```

且已经指定了全局配置 `crawl_config` 中的 `headers` 的 `Content-Type` 是 `application/json`

-> 以为 PySpider 的 `self.crawl` 会自动把 `dict` 类型的 `data` 去 `json dump` 成字符串，结果却是：

只保留了 `key` 和 `value`：

```
"data": "provinceId=10",
```

导致报错：

```
requests.exceptions.HTTPError: HTTP 400: Bad Request
```

之前还遇到过一个类似的例子：

```
SelfReadingUrl = "http://www.xxxxxxxxxx.cn:83/Reading.svc/se

def on_start(self):
    jValueTemplateSelfReading = "{\"userId\":\"%s\", \""
    paramDictSelfReading = {
        "curUrl": SelfReadingUrl,
        "offset": 0,
        "limit": DefaultPageSize,
        "jValueTemplate": jValueTemplateSelfReading
    }
    self.getBookQuery2(paramDictSelfReading)

def getBookQuery2(self, curParamDict):
    ...
    jValueStr = jValueTemplate % (gUserId, offset, limit)
    jcJsonDict = {
        "J": jValueStr,
        "C": 0
    }
    jcJsonDictStr = json.dumps(jcJsonDict)
    .....
    self.crawl(curUrlWithHash,
               itag=fakeItagForceRecrawl, # To force re-crawl
               method="POST",
               # data=jcJsonDict,
               data= jcJsonDictStr,
               # callback=curCallback,
               callback=self.getBookQuery2Callback,
               headers=curHeaders,
               save=curParamDict
    )

```

其中也是：

给 `data` 参数用了 `json` 去 `dump` 后的字符串变量： `jcJsonDictStr`

而不是 `json` 的 `dict` 变量： `jcJsonDict`

才最终正确获取到数据的。

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook最后更新： 2021-09-08 11:13:04

PySpider案例

下面把一些之前写过的 PySpider 的爬虫分享出来，供参考。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-04-14 20:12:48

汽车之家的品牌等数据

需求

想要抓取

汽车之家

<https://www.autohome.com.cn/car/>

The screenshot shows the homepage of AutoHome (汽车之家) with the URL https://www.autohome.com.cn/car/. At the top, there is a navigation bar with links for '首页' (Home), '论坛' (Forum), '二手车' (Used Cars), '新车' (New Cars), '导购' (Buy Guide), '行情' (Market Price), '购车手册' (Purchase Manual), and '车百科' (Car Encyclopedia). Below the navigation bar, there is a search bar and a filter section for '级别' (Level) with categories: 微型车, 小型车, 紧凑型车, 中型车, 大中型车, SUV, MPV, 跑车, 皮卡, 豪华, 轻客. A dropdown menu titled '按品牌' (By Brand) shows a list of brands: 热门 (Hot), A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, V, W, X, Y, Z. Below this is a grid of brand logos: 大众 (VW), 丰田 (Toyota), 奔驰 (Mercedes-Benz), 本田 (Honda), 宝马 (BMW), 奥迪 (Audi), 吉利汽车 (Geely), 比亚迪 (BYD), 别克 (Buick), 长安 (Changan), 福特 (Ford), 日产 (Nissan), 现代 (Hyundai), 雷克萨斯 (Lexus). The main content area displays a list of Audi models under the heading '一汽-大众奥迪'. The models listed are: 奥迪A3 (指导价: 20.31-24.97万), 奥迪A4L (指导价: 30.58-39.68万), 奥迪A6L (指导价: 41.98-65.38万), 奥迪A6新能源 (指导价: 50.80-50.80万), 奥迪Q2L (指导价: 21.88-26.88万), 奥迪Q3 (指导价: 22.68-23.73万), 奥迪Q3 Sportback (指导价: 27.43-35.68万), 奥迪Q5L (指导价: 39.68-48.70万), 奥迪Q5L Sportback (指导价: 42.68-49.99万), 奥迪e-tron (指导价: 54.68-64.80万), 奥迪A4 (指导价: 暂无), 奥迪A6 (指导价: 暂无), 奥迪Q4 (指导价: 暂无), 奥迪Q5 (指导价: 暂无), 奥迪RS 4 (指导价: 81.98-84.98万), 奥迪RS 5 (指导价: 87.68-88.98万), 奥迪RS 6 (指导价: 147.88-177.98万), 奥迪RS 7 (指导价: 144.88-172.95万), 奥迪RS Q8 (指导价: 146.88-184.74万). Each model entry includes links for '报价' (Quotation), '图库' (Image Library), '二手车' (Used Car), '论坛' (Forum), and '口碑' (Review).

所有的品牌的车系和车型数据

核心过程

虽然前面汽车之家的主页是

<https://www.autohome.com.cn/car/>

但是对于：方便爬虫去找所有品牌的入口

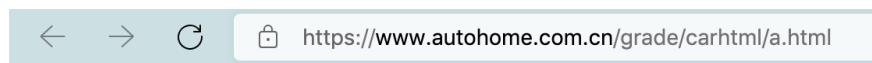
经过研究是：

```
self.crawl("https://www.autohome.com.cn/grade/carhtml/%s.h
```

其中： %s 支持从 a 到 z

比如对于 a 是：

<https://www.autohome.com.cn/grade/carhtml/a.html>



奥迪

[上汽奥迪](#)

- [奥迪A7L](#)

指导价: 暂无

报价 [图库](#) [二手车](#) [论坛](#) [口碑](#)

- [奥迪Concept Shanghai](#)

指导价: 暂无

报价 [图库](#) [二手车](#) [论坛](#) [口碑](#)

[一汽-大众奥迪](#)

- [奥迪A3](#)

指导价: [20.31-24.97万](#)

报价 [图库](#) [二手车](#) [论坛](#) [口碑](#)

- [奥迪A4L](#)

指导价: [30.58-39.68万](#)

报价 [图库](#) [二手车](#) [论坛](#) [口碑](#)

- [奥迪A6L](#)

指导价: [41.98-65.38万](#)

报价 [图库](#) [二手车](#) [论坛](#) [口碑](#)

- [奥迪A6L新能源](#)

指导价: [50.80-50.80万](#)

<https://car.autohome.com.cn/pic/series/692.html#pvareaid=103448>

其中，我们要去抓取的是某个品牌的某个系列，比如：[奥迪A4L](#)的 [图库](#)：

<https://car.autohome.com.cn/pic/series/692.html#pvareaid=103448>

然后再去 [picSeriesPage](#) 中：

```
fnRightPicSeries = response.doc('search-pic-tbar')
```

找到： [search-pic-tbar](#) 下面的 [href*="/pic/series"](#)：

安装和启动的常见问题



然后，单独再去找到车型详情页 spec 页面，比如

```
modelSpecUrlTemplate =  
"https://www.autohome.com.cn/spec/%s/#pvareaid=2042128"
```

对应的地址，比如：

<https://www.autohome.com.cn/spec/32708/#pvareaid=2042128>

对应页面：



其顶部，有很多相关的字段，比如： 品牌 (=车厂=厂商) 、 子品
牌 、 厂商指导价 等等

以及页面下面有，全系（不同配置的）车型的数据：

安装和启动的常见问题

The screenshot shows a web browser displaying a car specification page. At the top, there's a navigation bar with links like '车型首页', '参数配置', '车型图片', etc. Below the navigation, there are two rows of vehicle details:

大众迈腾	大众迈腾	日产轩逸	日产轩逸
10.98万 比新车省15.9万 5.80万公里 2015年	11.88万 比新车省13.4万 7.00万公里 2016年	11.80万 比新车省16万 1.40万公里 2021年	6.88万 比新车省7.8万 5.00万公里 2017年

Below this is a promotional banner for '¥100000000' worth of rebates. The main content area is titled '同车系其它车型' (Other models in the same series) and lists three more vehicle options:

1.4升 涡轮增压 131马力 国VI	1.4升 涡轮增压 160马力 国VI
2021款 230TSI DSG风尚版 前置前驱 / 7挡干式双离合	2021款 改款 280TSI DSG舒适版 前置前驱 / 7挡干式双离合
14.99万 起 获取底价	17.69万 起 获取底价
507条 团购 +对比	43条 团购 +对比

At the bottom of the page, there's a footer with links like '车型首页', '参数配置', '车型图片', etc.

然后用 `carModelSpecPage`，再去解析不同字段

最终即可获取到，所有从车品牌的所有的车系的所有车型的基本信息了。

详见：

- [【已解决】写Python爬虫爬取汽车之家品牌车系车型数据 – 在路上](#)

PySpider

核心文件：`autohomeBrandData.py`

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2018-04-27 21:53:02
# Project: autohomeBrandData

from pypspider.libs.base_handler import *
import string
import re

class Handler(BaseHandler):
    crawl_config = {
    }

    # @every(minutes=24 * 60)
    def on_start(self):
        for eachLetter in list(string.ascii_lowercase):
            self.crawl("https://www.autohome.com.cn/grade/")

    @catch_status_code_error
    def gradCarHtmlPage(self, response):
        print("gradCarHtmlPage: response=", response)
        picSeriesItemList = response.doc('.rank-list-ul li')
        print("picSeriesItemList=", picSeriesItemList)
        # print("len(picSeriesItemList)=%s"%(len(picSeriesItemList)))
        for each in picSeriesItemList:
            self.crawl(each.attr.href, callback=self.picSeriesPage)

    @config(priority=2)
    def picSeriesPage(self, response):
        # <a href="/pic/series-t/66.html">查看停产车型 </a>
        # <a class="ckmore" href="/pic/series/588.html">查看全部车型</a>
        # <span class="fn-right">&ampnbsp</span>
        fnRightPicSeries = response.doc('.search-pic-tbar .fn-right')
        print("fnRightPicSeries=", fnRightPicSeries)
        if fnRightPicSeries:
            # hrefValue = fnRightPicSeries.attr.href
            # print("hrefValue=", hrefValue)
            # fullPicSeriesUrl = "https://car.autohome.com.cn/pic/series/588.html"
            fullPicSeriesUrl = fnRightPicSeries.attr.href
            print("fullPicSeriesUrl=", fullPicSeriesUrl)
            self.crawl(fullPicSeriesUrl, callback=self.picSeriesDetail)

    # continue parse brand data
    aDictList = []
    # for eachA in response.doc('.breadnav a[href^="/"]'):
    for eachA in response.doc('.breadnav a[href*="/pic/"]'):
        eachADict = {
            "text": eachA.text(),
            "href": eachA.attr.href
        }
        aDictList.append(eachADict)
```

```
        }

        print("eachADict=", eachADict)
        aDictList.append(eachADict)

        print("aDictList=", aDictList)

        mainBrandDict = aDictList[-3]
        subBrandDict = aDictList[-2]
        brandSerieDict = aDictList[-1]
        print("mainBrandDict=%s, subBrandDict=%s, brandSer:

        dtTextList = []
        for eachDt in response.doc("dl.search-pic-card1 dt"):
            dtTextList.append(eachDt.text())

        print("dtTextList=", dtTextList)

        groupCount = len(dtTextList)
        print("groupCount=", groupCount)

        for eachDt in response.doc("dl.search-pic-card1 dt"):
            dtTextList.append(eachDt.text())

        ddUlEltList = []
        for eachDdUlElt in response.doc("dl.search-pic-card1 dd ul"):
            ddUlEltList.append(eachDdUlElt)

        print("ddUlEltList=", ddUlEltList)

        modelDetailDictList = []

        for curIdx in range(groupCount):
            curGroupTitle = dtTextList[curIdx]
            print("-----[%d] %s" % (curIdx, curGroupTitle))

            for eachLiAElt in ddUlEltList[curIdx].items("li"):
                # 1. model name
                # curModelName = eachLiAElt.text()
                curModelName = eachLiAElt.contents()[0]
                curModelName = curModelName.strip()
                print("curModelName=", curModelName)
                curFullmodelName = curGroupTitle + " " + curModelName
                print("curFullmodelName=", curFullmodelName)

                # 2. model id + carSeriesId + spec url
                curModelId = ""
                curSeriesId = ""
                curModelSpecUrl = ""
                modelSpecUrlTemplate = "https://www.autohome.com.cn/specs/
                curModelPicUrl = eachLiAElt.attr.href
                print("curModelPicUrl=", curModelPicUrl)
```

```

# https://car.autohome.com.cn/pic/series-s:
foundModelSeriesId = re.search("pic/series-",
                                curModelPic)
print("foundModelSeriesId=", foundModelSeriesId)
if foundModelSeriesId:
    curModelId = foundModelSeriesId.group(1)
    curSeriesId = foundModelSeriesId.group(2)
    print("curModelId=%s, curSeriesId=%s",
          curModelSpecUrl = (modelSpecUrlTemplate % curSeriesId))
    print("curModelSpecUrl=", curModelSpecUrl)

# 3. model status
modelStatus = "在售"
foundStopSale = eachLiAElt.find('i[class*="icon"]')
if foundStopSale:
    modelStatus = "停售"
else:
    foundWseason = eachLiAElt.find('i[class*="icon"]')
    if foundWseason:
        modelStatus = "未上市"

modelDetailDictList.append({
    "url": curModelSpecUrl,
    # "车系ID": curSeriesId,
    # "车型ID": curModelId,
    # "车型": curFullModelName,
    # "状态": modelStatus
    "brandSerieId": curSeriesId,
    "modelId": curModelId,
    "model": curFullModelName,
    "modelStatus": modelStatus
})

print("modelDetailDictList=", modelDetailDictList)

allSerieDictList = []
for curIdx, eachModelDetailDict in enumerate(modelDetailDictList):
    print("modelDetailDictList[%d] = %s" % (curIdx, eachModelDetailDict))
    print("modelDetailDictList[%d].keys() = %s" % (curIdx, eachModelDetailDict.keys()))
    print("modelDetailDictList[%d].values() = %s" % (curIdx, eachModelDetailDict.values()))
    print("modelDetailDictList[%d].items() = %s" % (curIdx, eachModelDetailDict.items()))

    # define in mysql
    # CREATE TABLE `tbl_autohome_car_info` (
    #     `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
    #     `cityDealerPrice` int(11) unsigned NOT NULL DEFAULT '0',
    #     `msrpPrice` int(11) unsigned NOT NULL DEFAULT '0',
    #     `mainBrand` char(20) NOT NULL DEFAULT '',
    #     `subBrand` varchar(20) NOT NULL DEFAULT '',
    #     `brandSerie` varchar(20) NOT NULL DEFAULT '',
    #     `brandSerieId` varchar(15) NOT NULL DEFAULT '',
    #     `model` varchar(50) NOT NULL DEFAULT '',
    #     `modelId` varchar(15) NOT NULL DEFAULT '',
    #     `modelStatus` char(5) NOT NULL DEFAULT ''

```

```
        `url` varchar(200) NOT NULL DEFAULT '' COMMENT '车型URL',
        PRIMARY KEY (`id`)
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
    ....
    curSerieDict = {
        "url": eachModelDetailDict["url"],
        # "品牌": mainBrandDict["text"],
        # "子品牌": subBrandDict["text"],
        # "车系": brandSerieDict["text"],
        # "车系ID": eachModelDetailDict["车系ID"],
        # "车型": eachModelDetailDict["车型"],
        # "车型ID": eachModelDetailDict["车型ID"],
        # "状态": eachModelDetailDict["状态"]
        "mainBrand": mainBrandDict["text"],
        "subBrand": subBrandDict["text"],
        "brandSerie": brandSerieDict["text"],
        "brandSerieId": eachModelDetailDict["brandSerieId"],
        "model": eachModelDetailDict["model"],
        "modelId": eachModelDetailDict["modelId"],
        "modelStatus": eachModelDetailDict["modelStatus"]
    }
    allSerieDictList.append(curSerieDict)
    # print("before send_message: [%d] curSerieDict=%s" % (curIdx, curSerieDict))
    # self.send_message(self.project_name, curSerieDict)
    print("[%d] curSerieDict=%s" % (curIdx, curSerieDict))
    self.crawl(eachModelDetailDict["url"],
               callback=self.carModelSpecPage,
               fetch_type='js',
               retries=5,
               connect_timeout=50,
               timeout=300,
               save=curSerieDict)

    # print("allSerieDictList=", allSerieDictList)
    # return allSerieDictList

    # def on_message(self, project, msg):
    #     print("on_message: msg=", msg)
    #     return msg

    @catch_status_code_error
    def carModelSpecPage(self, response):
        print("carModelSpecPage: response=", response)
        # https://www.autohome.com.cn/spec/32708/#pvareaid=10000000000000000000000000000000
        curSerieDict = response.save
        print("curSerieDict", curSerieDict)

        cityDealerPriceInt = 0
        cityDealerPriceElt = response.doc('.cardetail-info').find('span')
        print("cityDealerPriceElt=%s" % cityDealerPriceElt)
        if cityDealerPriceElt:
```

```
cityDealerPriceFloatStr = cityDealerPriceElt.text
print("cityDealerPriceFloatStr=", cityDealerPriceFloatStr)
cityDealerPriceFloat = float(cityDealerPriceFloatStr)
print("cityDealerPriceFloat=", cityDealerPriceFloat)
cityDealerPriceInt = int(cityDealerPriceFloat * 10000)
print("cityDealerPriceInt=", cityDealerPriceInt)

msrpPriceInt = 0
# body > div.content > div.row > div.column.grid-10
# 厂商指导价=厂商建议零售价格=MSRP=Manufacturer's suggested retail price
msrpPriceElt = response.doc('.cardetail-infor-pricetext')
print("msrpPriceElt=", msrpPriceElt)
if msrpPriceElt:
    msrpPriceStr = msrpPriceElt.attr("data-price")
    print("msrpPriceStr=", msrpPriceStr)
    foundMsrpPrice = re.search("(?P<msrpPrice>[\d\.]+)", msrpPriceStr)
    print("foundMsrpPrice=", foundMsrpPrice)
    if foundMsrpPrice:
        msrpPrice = foundMsrpPrice.group("msrpPrice")
        print("msrpPrice=", msrpPrice)
        msrpPriceFloat = float(msrpPrice)
        print("msrpPriceFloat=", msrpPriceFloat)
        msrpPriceInt = int(msrpPriceFloat * 10000)
        print("msrpPriceInt=", msrpPriceInt)

# curSerieDict["经销商参考价"] = cityDealerPriceInt
# curSerieDict["厂商指导价"] = msrpPriceInt
curSerieDict["cityDealerPrice"] = cityDealerPriceInt
curSerieDict["msrpPrice"] = msrpPriceInt

return curSerieDict
```

worker文件： AutohomeResultWorker.py

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Project: autohomeBrandData
# Function: implement custom result worker for autohome crawler
# Author: Crifan Li
# Date: 20180512
# Note:
#   If you want to modify to your mysql and table, you need to
#   (1) change change MySqlDb config to your mysql config
#   (2) change CurrentTableName to your table name
#   (3) change CreateTableSqlTemplate to your sql to create table
#   (4) before use this ResultWorker, run py file to execute
#   (5) if your table field contain more type, edit insert sql

import pymysql
import pymysql.cursors
from pyspider.result import ResultWorker

CurrentTableName = "tbl_autohome_car_info"
CreateTableSqlTemplate = """CREATE TABLE IF NOT EXISTS `%s` (
    `id` int(11) unsigned NOT NULL AUTO_INCREMENT COMMENT '自增ID',
    `cityDealerPrice` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '城市经销商价',
    `msrpPrice` int(11) unsigned NOT NULL DEFAULT '0' COMMENT '厂商指导价',
    `mainBrand` char(20) NOT NULL DEFAULT '' COMMENT '品牌',
    `subBrand` varchar(20) NOT NULL DEFAULT '' COMMENT '子品牌',
    `brandSerie` varchar(20) NOT NULL DEFAULT '' COMMENT '车系',
    `brandSerieId` varchar(15) NOT NULL DEFAULT '' COMMENT '车系ID',
    `model` varchar(50) NOT NULL DEFAULT '' COMMENT '车型',
    `modelId` varchar(15) NOT NULL DEFAULT '' COMMENT '车型ID',
    `modelStatus` char(5) NOT NULL DEFAULT '' COMMENT '车型状态',
    `url` varchar(200) NOT NULL DEFAULT '' COMMENT '车型url',
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;"""

class AutohomeResultWorker(ResultWorker):

    def __init__(self, resultdb, inqueue):
        """init mysql db"""
        print("AutohomeResultWorker init: resultdb=%s, inqueue=%s" % (resultdb, inqueue))
        ResultWorker.__init__(self, resultdb, inqueue)

        self.mysqlDb = MySqlDb()
        print("self.mysqlDb=%s" % self.mysqlDb)

    def on_result(self, task, result):
        """override pyspider on_result to save data into mysql db"""
        # assert task['taskid']
        # assert task['project']
        # assert task['url']
```

```

# assert result
print("AutohomeResultWorker on_result: task=%s, res=%s" % (task, result))
insertOk = self.mysqlDb.insert(result)
print("insertOk=%s" % insertOk)

class MySqlDb:
    config = {
        'host': '127.0.0.1',
        'port': 3306,
        'user': 'root',
        'password': 'crifan_mysql',
        'database': 'AutohomeResultdb',
        'charset': "utf8"
    }

defaultTableName = CurrentTableName
connection = None

def __init__(self):
    """init mysql"""
    # 1. connect db first
    if self.connection is None:
        isConnected = self.connect()
        print("Connect mysql return %s" % isConnected)

    # 2. create table for db
    createTableOk = self.createTable(self.defaultTableName)
    print("Create table %s return %s" %(self.defaultTableName, createTableOk))

def connect(self):
    try:
        self.connection = pymysql.connect(**self.config)
        print("connect mysql ok, self.connection=%s" % self.connection)
        return True
    except pymysql.Error as err:
        print("Connect mysql with config=%s, self.config=%s" % (self.config, self.connection))
        return False

def quoteIdentifier(self, identifier):
    """
        for mysql, it better to quote identifier xxx using ''
        in case, identifier:
            contain special char, such as space
            or same with system reserved words, like select
    """
    quotedIdentifier = "'%s'" % identifier
    # print("quotedIdentifier=%s" % quotedIdentifier)
    return quotedIdentifier

def executeSql(self, sqlStr, actionDescription=""):
    print("executeSql: sqlStr=%s, actionDescription=%s" % (sqlStr, actionDescription))

```

```

    if self.connection is None:
        print("Please connect mysql first before %s" %
        return False

    cursor = self.connection.cursor()
    print("cursor=%s", cursor)

    try:
        cursor.execute(sqlStr)
        self.connection.commit()
        print("++ Ok to execute sql %s for %s" % (sqlStr,
        return True
    except pymysql.Error as err:
        print("!!! %s when execute sql %s for %s" % (err,
        return False

    def createTable(self, newTablename):
        print("createTable: newTablename=%s", newTablename)

        createTableSql = CreateTableSqlTemplate % (newTablename)
        print("createTableSql=%s", createTableSql)

        return self.executeSql(sqlStr=createTableSql, action="CREATE")

    def dropTable(self, existedTablename):
        print("dropTable: existedTablename=%s", existedTablename)

        dropTableSql = "DROP TABLE IF EXISTS %s" % (existedTablename)
        print("dropTableSql=%s", dropTableSql)

        return self.executeSql(sqlStr=dropTableSql, action="DROP")

    # def insert(self, **valueDict):
    def insert(self, valueDict, tablename=defaultTableName):
        """
            inset dict value into mysql table
            makesure the value is dict, and its keys is the column name
        """
        print("insert: valueDict=%s, tablename=%s" % (valueDict,
        tablename))

        dictKeyList = valueDict.keys()
        dictValueList = valueDict.values()
        print("dictKeyList=%s", dictKeyList, "dictValueList=%s",
        dictValueList)

        keyListSql = ", ".join(self.quoteIdentifier(eachKey)
        print("keyListSql=%s", keyListSql)
        # valueListSql = ", ".join(eachValue for eachValue in
        valueListSql = """
        formattedDictValueList = []
        for eachValue in dictValueList:

```

```

        # print("eachValue=", eachValue)
        eachValueInSql = ""
        valueType = type(eachValue)
        # print("valueType=", valueType)
        if valueType is str:
            eachValueInSql = "%s" % eachValue
        elif valueType is int:
            eachValueInSql = "%d" % eachValue
        # TODO: add more type formatting if necessary
        print("eachValueInSql=", eachValueInSql)
        formattedDictValueList.append(eachValueInSql)

    valueListSql = ", ".join(eachValue for eachValue in
    print("valueListSql=", valueListSql)

    insertSql = """INSERT INTO %s (%s) VALUES (%s)""" %
    print("insertSql=", insertSql)
    # INSERT INTO tbl_car_info_test (`url`, `mainBrand`)

    return self.executeSql(sqlStr=insertSql, actionDescription="insert")

def delete(self, modelId, tablename=defaultTableName):
    """
        delete item from car model id for existing table
    """
    print("delete: modelId=%s, tablename=%s" % (modelId, tablename))

    deleteSql = """DELETE FROM %s WHERE modelId = %s"""
    print("deleteSql=", deleteSql)

    return self.executeSql(sqlStr=deleteSql, actionDescription="delete")

def testMysqlDb():
    """
    test mysql
    """

    testDropTable = True
    testCreateTable = True
    testInsertValue = True
    testDeleteValue = True

    # 1. test connect mysql
    mysqlObj = MysqlDb()
    print("mysqlObj=", mysqlObj)

    # testTablename = "autohome_car_info"
    # testTablename = "tbl_car_info_test"
    testTablename = CurrentTableName
    print("testTablename=", testTablename)

    if testDropTable:
        # 2. test drop table

```

```
dropTableOk = mysqlObj.dropTable(testTablename)
print("dropTable", testTablename, "return", dropTableOk)

if testCreateTable:
    # 3. test create table
    createTableOk = mysqlObj.createTable(testTablename)
    print("createTable", testTablename, "return", createTableOk)

if testInsertValue:
    # 4. test insert value dict
    valueDict = {
        "url": "https://www.autohome.com.cn/spec/5872/",
        "mainBrand": "宝马", #品牌
        "subBrand": "华晨宝马", #子品牌
        "brandSerie": "宝马3系", #车系
        "brandSerieId": "66", #车系ID
        "model": "2010款 320i 豪华型", #车型
        "modelId": "5872", #车型ID
        "modelStatus": "停售", #车型状态
        "cityDealerPrice": 325000, #经销商参考价
        "msrpPrice": 375000 # 厂商指导价
    }
    print("valueDict=", valueDict)
    insertOk = mysqlObj.insert(valueDict=valueDict, tableName=testTablename)
    print("insertOk=", insertOk)

if testDeleteValue:
    toDeleteModelId = "5872"
    deleteOk = mysqlObj.delete(modelId=toDeleteModelId, tableName=testTablename)
    print("deleteOk=", deleteOk)

def testAutohomeResultWorker():
    """just test for create mysql db is ok or not"""
    autohomeResultWorker = AutohomeResultWorker(None, None)
    print("autohomeResultWorker=%s" % autohomeResultWorker)

if __name__ == '__main__':
    testMysqlDb()
    # testAutohomeResultWorker()
```

配置文件: `config.json`

```
{
    "resultdb": "mysql+resultdb://root:crifan_mysql@127.0.0.1:3306/test?charset=utf8",
    "result_worker": {
        "result_cls": "AutohomeResultWorker.AutohomeResultWorker"
    },
    "phantomjs-proxy": "127.0.0.1:23450",
    "phantomjs": {
        "port": 23450,
        "auto-restart": true,
        "load-images": false,
        "debug": true
    }
}
```

相关内容

相关页面

汽车之家的入口页面：

The screenshot shows the Autohome homepage for car models. It features a navigation bar with vehicle categories like Mini Car, Small Car, etc., and a search bar. Below the navigation is a grid of Audi models with their names and prices. A developer tools window is overlaid on the page, showing the DOM structure and CSS styles for the Audi RS 4 model. The developer tools panel includes tabs for Elements, Sources, Network, and Styles.

对应的PySpider调试页面

安装和启动的常见问题

The screenshot shows the PySpider IDE interface. At the top, there's a browser-like header with '0.0.0.0:5000/debug/autohomeCarData'. Below it, the PySpider command-line interface shows the command 'pyspider > autohomeCarData'. The main area contains the Python code for the project:

```
fetch: {},  
process: {  
    "callback": "picSeriesPage"  
},  
project: "autohomeCarData",  
schedule: {}  
taskid: "4e9c45a1012210c21c8c7037ec891ace",  
url: "https://www.autohome.com.cn/car/"  
<< 汽车之家首页 | 城市: 北京 上海 深圳 广州 杭州 南京 查看所有城市>>  
当前位置: 首页 > 全部级别  
级别: 全部 小型车 微型车 紧凑型车 中型车 中大型车 大型车 SUV MPV  
enable css selector helper web html follow messages
```

The code defines a project named 'autohomeCarData' with a single task. It uses the 'picSeriesPage' callback. The schedule is empty. The task ID is '4e9c45a1012210c21c8c7037ec891ace' and the URL is 'https://www.autohome.com.cn/car/'. The interface preview shows a car search page with categories like 全部, 小型车, 微型车, 紧凑型车, 中型车, 中大型车, 大型车, SUV, 和 MPV.

调试PySpider

期间会涉及到PySpider的调试

比如：

用了debug模式去调试：

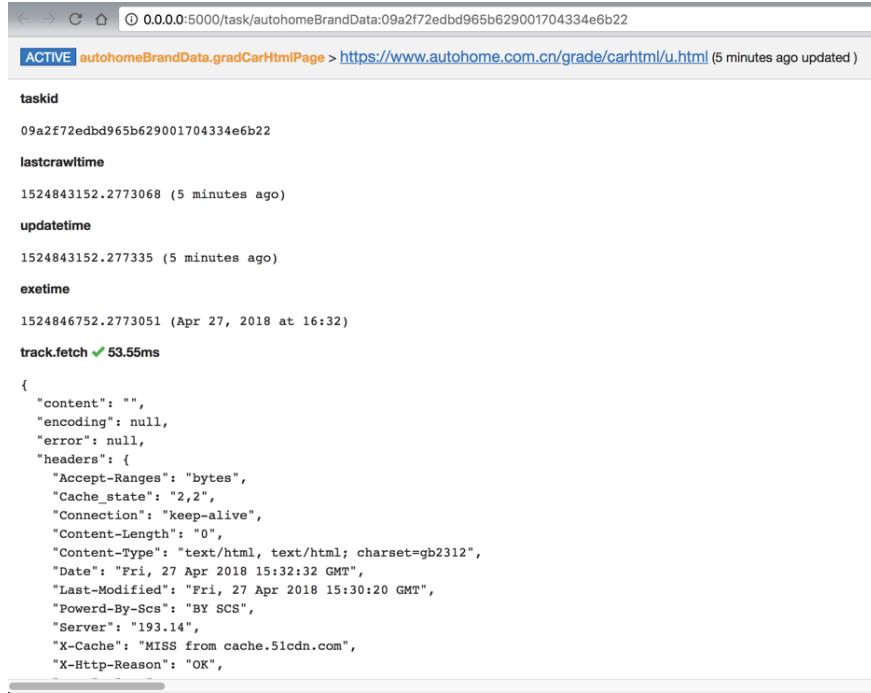
The screenshot shows the PySpider task list interface. A task named 'autohomeBrandData' is listed under a group. The status is 'DEBUG'. The task has run 1/3 times, with an average time of 232.9+29.45 seconds. The progress bar shows 5m: 172, 1h: 6630, 1d: 5212. A tooltip provides more detailed statistics: 1h of 6630 tasks: net: 0 failed: 0 success(50.2%): 3329 retry(0.0%): 3 failed(0.0%): 0.

遇到一个出错的：

The screenshot shows the PySpider task list interface. A task named 'autohomeBrandData' is listed under a group. The status is 'PROCESS_ERROR'. The task has run 3 minutes ago, with an average time of 53.6+0.53ms. The log shows multiple error entries, all labeled 'PROCESS_ERROR' with the message 'autohomeBrandData > https://www.autohome.com.cn/grade/carhtml/u.html'. The errors occurred at different times, mostly between 3 and 5 minutes ago.

点击进去可以看到（出错的子任务的）详情信息：

安装和启动的常见问题



A screenshot of a browser window showing the URL 0.0.0.0:5000/task/autohomeBrandData:09a2f72edbd965b629001704334e6b22. The page displays task details:

```
taskid  
09a2f72edbd965b629001704334e6b22  
  
lastcrawltime  
1524843152.2773068 (5 minutes ago)  
  
updatetime  
1524843152.277335 (5 minutes ago)  
  
exetime  
1524846752.2773051 (Apr 27, 2018 at 16:32)  
  
track.fetch ✓ 53.55ms  
  
{  
    "content": "",  
    "encoding": null,  
    "error": null,  
    "headers": {  
        "Accept-Ranges": "bytes",  
        "Cache-State": "2,2",  
        "Connection": "keep-alive",  
        "Content-Length": "",  
        "Content-Type": "text/html, text/html; charset=gb2312",  
        "Date": "Fri, 27 Apr 2018 15:32:32 GMT",  
        "Last-Modified": "Fri, 27 Apr 2018 15:30:20 GMT",  
        "Powered-By-Scs": "BY SCS",  
        "Server": "193.14",  
        "X-Cache": "MISS from cache.51cdn.com",  
        "X-Http-Reason": "OK",  
    }  
}
```

此处明显是：



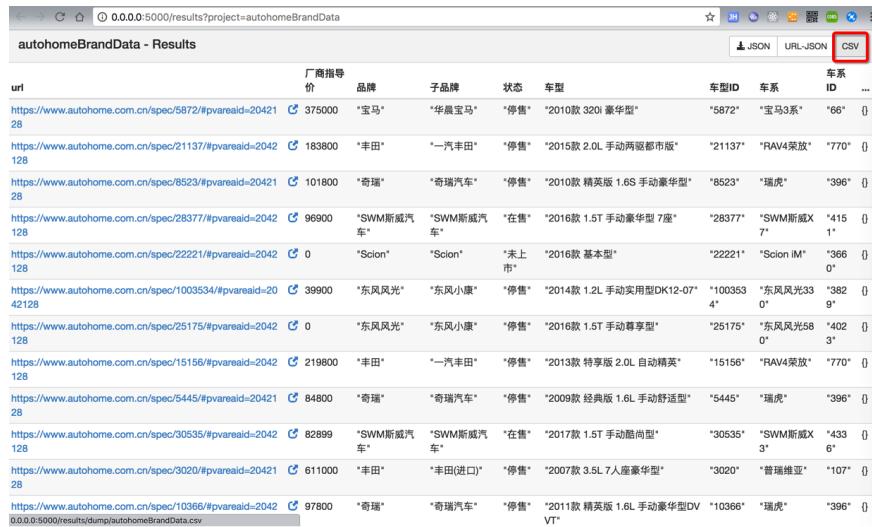
A screenshot of a browser window showing the URL 0.0.0.0:5000/task/autohomeBrandData:09a2f72edbd965b629001704334e6b22. The page displays task details and includes a message: "Document is empty".

```
Document is empty  
  
track.process ✘ 0.53ms  
  
Document is empty  
libs/base_handler.py", line 155, in _run_func  
    ret = function(*arguments[:len(args) - 1])  
File "<autohomeBrandData>", line 19, in gradCarHtmlPage  
File "/Users/crifan/.local/share/virtualenvs/AutocarData-x1-iqIq4/lib/python3.6/site-packages/pyspider/libs/response.py", line 144, in doc  
elements = self.etree  
File "/Users/crifan/.local/share/virtualenvs/AutocarData-x1-iqIq4/lib/python3.6/site-packages/pyspider/libs/response.py", line 160, in etree  
self._elements = lxml.html.fromstring(self._content)  
File "/Users/crifan/.local/share/virtualenvs/AutocarData-x1-iqIq4/lib/python3.6/site-packages/lxml/html/_init_.py", line 876, in fromstring  
doc = document_fromstring(html, parser=parser, base_url=base_url, **kw)  
File "/Users/crifan/.local/share/virtualenvs/AutocarData-x1-iqIq4/lib/python3.6/site-packages/lxml/html/_init_.py", line 765, in document  
    'document is empty')  
lxml.etree.ParserError: Document is empty  
  
{  
    "exception": "Document is empty",  
    "follows": 0,  
    "log": "libs/base_handler.py", line 155, in _run_func\n            ret = function(*arguments[:len(args) - 1])\nFile "<autohomeBrandData>\",  
    "ok": false,  
    "result": null,  
    "time": 0.0005340576171875  
}  
  
schedule
```

导出结果

PySpider运行完毕（或运行期间）可以去results页面去导出结果：

安装和启动的常见问题



The screenshot shows a browser window with the URL `0.0.0.0:5000/results?project=autohomeBrandData`. The page title is "autohomeBrandData - Results". At the top right, there are three buttons: "JSON", "URL-JSON", and "CSV", with "CSV" being highlighted by a red box. The main content is a table with the following columns: url, 厂商指导价 (Manufacturer's suggested retail price), 品牌 (Brand), 子品牌 (Sub-brand), 状态 (Status), 车型 (Vehicle model), 车型ID (Vehicle model ID), 车系 (Vehicle series), 车系ID (Vehicle series ID), and ... (ellipsis). The table lists various car models from different manufacturers like BMW, Toyota, Chevrolet, etc., with their respective details.

url	厂商指导价	品牌	子品牌	状态	车型	车型ID	车系	车系ID	...
https://www.autohome.com.cn/spec/5872/#pvareaid=20421	375000	"宝马"	"华晨宝马"	"停售"	"2010款 320i 豪华型"	"5872"	"宝马3系"	"66"	{}
28									
https://www.autohome.com.cn/spec/21137/#pvareaid=2042	183800	"丰田"	"一汽丰田"	"停售"	"2015款 2.0L 手动两驱都市版"	"21137"	"RAV4荣放"	"770"	{}
128									
28									
https://www.autohome.com.cn/spec/8523/#pvareaid=2042	101800	"奇瑞"	"奇瑞汽车"	"停售"	"2010款 精英版 1.6S 手动豪华型"	"8523"	"瑞虎"	"396"	{}
128									
https://www.autohome.com.cn/spec/28377/#pvareaid=2042	96900	"SWM斯威汽车"	"SWM斯威汽车"	"在售"	"2016款 1.5T 手动豪华型 7座"	"28377"	"SWM斯威X7"	"415"	{}
128									
https://www.autohome.com.cn/spec/22221/#pvareaid=2042	0	"Scion"	"Scion"	"未上市"	"2016款 基本型"	"22221"	"Scion iM"	"366"	{}
128									
https://www.autohome.com.cn/spec/1003534/#pvareaid=20	39900	"东风风光"	"东风小康"	"停售"	"2014款 1.2L 手动实用型DK12-07"	"100353"	"东风风光330"	"382"	{}
42128									
128									
https://www.autohome.com.cn/spec/25175/#pvareaid=2042	0	"东风风光"	"东风小康"	"停售"	"2016款 1.5T 手动尊享型"	"25175"	"东风风光580"	"402"	{}
128									
https://www.autohome.com.cn/spec/15156/#pvareaid=2042	219800	"丰田"	"一汽丰田"	"停售"	"2013款 特享版 2.0L 自动精英"	"15156"	"RAV4荣放"	"770"	{}
128									
https://www.autohome.com.cn/spec/5445/#pvareaid=20421	84800	"奇瑞"	"奇瑞汽车"	"停售"	"2009款 经典版 1.6L 手动舒适型"	"5445"	"瑞虎"	"396"	{}
28									
https://www.autohome.com.cn/spec/30535/#pvareaid=2042	82899	"SWM斯威汽车"	"SWM斯威汽车"	"在售"	"2017款 1.5T 手动酷尚型"	"30535"	"SWM斯威X3"	"433"	{}
128									
https://www.autohome.com.cn/spec/3020/#pvareaid=20421	611000	"丰田(进口)"	"丰田(进口)"	"停售"	"2007款 3.5L 7人座豪华型"	"3020"	"普瑞维亚"	"107"	{}
28									
https://www.autohome.com.cn/spec/10366/#pvareaid=2042	97800	"奇瑞"	"奇瑞汽车"	"停售"	"2011款 精英版 1.6L 手动豪华型DVVT"	"10366"	"瑞虎"	"396"	{}
28									
http://0.0.0.0:5000/results/dump/autohomeBrandData.csv									

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook最后更新: 2021-09-08 13:09:24

汽车之家的车型详细数据

需求

爬取

汽车之家

<https://www.autohome.com.cn/car/>

的汽车品牌的车型车型的详细数据

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新: 2021-09-08 13:55:04

过程

本身汽车之家入口是

<https://www.autohome.com.cn/car/>

但是经研究，更利于爬虫抓取的是：

从：

<https://www.autohome.com.cn/grade/carhtml/a.html>

到：

<https://www.autohome.com.cn/grade/carhtml/z.html>

其中可以找到车系的数据：

audi

奥迪

一汽-大众奥迪

• 奥迪A3

指导价: 19.32-23.46万
报价 图库 二手车 论坛 口碑

• 奥迪A4L

指导价: 30.58-39.68万
报价 图库 二手车 论坛 口碑

• 奥迪A6L

指导价: 40.98-65.38万
报价 图库 二手车 论坛 口碑

• 奥迪A6L新能源

指导价: 50.80-50.80万
报价 图库 二手车 论坛 口碑

• 奥迪Q2L

指导价: 21.88-26.85万
报价 图库 二手车 论坛 口碑

•

对应的值分别是：

- carBrandName : 奥迪
- carBrandLogoUrl :
https://car2.autoimg.cn/cardfs/series/g26/M0B/AE/B3/100x100_f40_autohomecar_wKgHEVs9u5WAV441AAKdxZGE4U148.png
- carMerchantName : 一汽-大众奥迪
- carMerchantUrl : <https://car.autohome.com.cn/price/brand-33-9.html#pvareaid=2042363>
- carSeriesName : 奥迪A3

安装和启动的常见问题

- carSeriesUrl :
https://www.autohome.com.cn/3170/#levelsource=000000000_0&pvareaid=101594
- carSeriesMsrp : 19.32-23.46万
- carSeriesMsrpUrl :
<https://www.autohome.com.cn/3170/price.html#pvareaid=101446>

然后进去车系主页=车系详情页

【奥迪A3】 奥迪奥迪A3报价奥迪A3图片_汽车之家

https://www.autohome.com.cn/3170/#levelsource=000000000_0&pvareaid=101594



The screenshot shows the Audi A3 car series page on AutoHome. At the top, there's a banner for Skoda with the text "备受宠爱60余载" and "明锐 9.99万-14.99万 SKODA". Below the banner, the page title is "一汽-大众奥迪 奥迪A3". The main content area features a large image of the Audi A3, with a color selection bar below it. To the right, there are sections for "精彩活动推荐" (Promotional Activities) and "促销信息推荐" (Promotional Information Recommendations). A comparison chart at the bottom lists Audi A3 and Volkswagen Golf, showing their respective scores and overall ratings.

可以抓取到：

- 汽车品牌Logo图片
- 汽车主品牌=主机厂=（汽车）厂商
- 车系
- 厂商指导价
- 汽车级别

以及页面下方的，全部的车系的数据：

安装和启动的常见问题

The screenshot shows the Audi A3 product page on the Autohome website. The main content area lists several car models with their details:

- 1.4升 涡轮增压 150马力 国VI (highlighted with a red box)
 - 口碑 图片 19.32万 [询底价]
 - 15.45万起 4条 [特惠]
 - [+对比]
- 2020款 改款 Sportback 35 TFSI 进取型 国VI
 - 口碑 图片 21.37万 [询底价]
 - 16.98万起 501条 [特惠]
 - [+对比]
- 2020款 Sportback 35 TFSI 时尚型 国VI
 - 口碑 图片 22.86万 [询底价]
 - 18.16万起 496条 [特惠]
 - [+对比]
- 2020款 Sportback 35 TFSI 运动型 国VI
 - 口碑 图片 22.86万 [询底价]
 - 18.16万起 506条 [特惠]
 - [+对比]
- 2020款 改款 Limousine 35 TFSI 进取

On the right side of the page, there are sections for "VR智能展厅" (Virtual Reality Intelligent Showroom) and "猜你喜欢" (Guess You Like). There is also a banner at the bottom indicating 6319位奥迪A3车友正在热聊.

可以抓取到：

- carModelGroupName: 1.5升 涡轮增压 169马力 国VI
- carModelYear: 2020款
- carModelEmissionStandards: 国VI
- carModelPower: 1.5T
- carModelGearBox: 7挡双离合

如果历史车型很多，分多个年份，则还要切换年份后再抓取：

购车询底价
热卖活动

狂撒10亿，多重好礼，低价购车
818百城车展 [查看详情 >](#)

狂撒1C
818百

奥迪A3 在售 停售款 ^

PK 热门车型综合对比 预约

全部 | 2020款 | 1.4T | 7挡双离合

1.4升 涡轮增压 150马力 | 车型信息 | 指导价 | 经销商参考价 | 成交价

2020款 改款 Sportback 型 国VI | 口碑 | 图片 | 19.32万 万 | 15.45万起 | 4条

前置前驱 / 7挡双离合 | 视频 | 配置

2020款 Sportback 35 TFSI 时尚型 国VI | 口碑 | 图片 | 21.37万 万 | 16.98万起 | 501条

前置前驱 / 7挡双离合 | 视频 | 配置

2020款 Sportback 35 TFSI 风尚型 国VI | 口碑 | 图片 | 22.86万 万 | 18.16万起 | 496条

前置前驱 / 7挡双离合 | 视频 | 配置

javascript:void(0);

之后，还要抓取每个车型的详细配置参数：

[【奥迪Q2L e-tron 2019款 Q2L e-tron 纯电智酷型参数配置表】价格单](#) [奥迪汽车之家](#)

<https://car.autohome.com.cn/config/spec/42875.html#pvareaid=3454541>

安装和启动的常见问题

https://car.autohome.com.cn/config/spec/42875.html#pvareaid=3454541

进入苏州车市 | 查看其它地方车市

汽车之家
看车·买车·用车

文章 | 评测 | 车家号 | 视频 | 直播 | 车型对比 | 图片 | 报价

NEW 新能源 微型车 小型车 紧凑型车 中型车 中大型车

当前位置：首页 > 小型SUV > 奥迪Q2L e-tron > 2019款 Q2L e-tron 纯电智酷型 > 参数配置

< 返回 奥迪Q2L e-tron频道 | 2019款 Q2L e-tron 纯电智酷型

车型首页 | 参数配置 | 车型图片 | 车型报价 | 口碑 | 车型详解 | 保养 | 二手车

筛选条件

总功率: 100 kW 续航里程: 265 公里

座位数: 5座

<input type="checkbox"/> 高亮显示差异参数	奥迪Q2L e-tron 2019款 Q2L e-tron 纯电智酷型	奥迪Q2L e-tron 2019款 Q2L e-tron 纯电智享型
<input type="checkbox"/> 隐藏相同参数		
<input type="checkbox"/> 隐藏暂无内容参数		
注: ●标配 ○选配 -无	+ 对比	+ 对比
厂商指导价	22.68万	23.73万
补贴后售价	22.68万	23.73万
经销商参考价	22.68万起	23.73万起

基本参数

车身	厂商	一汽-大众奥迪	一汽-大众奥迪
电动机	级别	小型SUV	小型SUV
变速箱	能源类型	纯电动	纯电动
底盘转向	上市时间	2019.11	2019.11
车轮制动	工信部纯电续航里程(km)	265	265
主/被动安全装备	快充时间(小时)	0.6	0.6
辅助/操控配置	慢充时间(小时)	17	17
外部/防盗配置	快充电量百分比	80	80
内部配置	最大功率(kW)	100	100
座椅配置	最大扭矩(N·m)	290	290

https://car.autohome.com.cn/config/spec/42875.html#pvareaid=3454541

		厂商指导价: 22.68万 奥迪Q2L e-tron 2019款 Q2L e-tron 纯电智酷型	厂商指导价: 23.73万 奥迪Q2L e-tron 2019款 Q2L e-tron 纯电智享型
基本参数		+ 对比 > < + 对比	
车身	级别	一汽-大众奥迪	一汽-大众奥迪
电动机	能源类型	小型SUV	小型SUV
变速箱	上市时间	纯电动	纯电动
底盘转向	工信部纯电续航里程(km)	2019.11	2019.11
车轮制动	快充时间(小时)	265	265
主/被动安全装备	慢充时间(小时)	0.6	0.6
辅助/操控配置	快充电量百分比	17	17
外部/防盗配置	最大功率(kW)	80	80
内部配置	最大扭矩(N·m)	100	100
座椅配置	电动机(Ps)	290	290
多媒体配置	长*宽*高(mm)	136	136
灯光配置	车身结构	4237*1785*1548	5门5座SUV
玻璃/后视镜	最高车速(km/h)	5门5座SUV	150
空调/冰箱	官方0-100km/h加速(s)	150	-
选装包	实测0-100km/h加速(s)	-	-
	实测100-0km/h制动(m)	-	-
	实测续航里程(km)	-	-
	实测快充时间(小时)	-	-
	实测慢充时间(小时)	-	-
	NEDC综合油耗(L/100km)	-	-
	整车质保	三年或10万公里	三年或10万公里
车身			

期间，有些内容的处理，还是很复杂的。

具体细节详见：

- 【未解决】用Python爬取汽车之家的车型车系详细数据
- 【已解决】Mac中恢复或重新搭建PySpider开发环境
- 【已解决】PySpider访问汽车之家报错：
`requests.exceptions.HTTPError HTTP 403 Forbidden`
- 【已解决】PySpider中获取PyQuery获取到节点的子元素
- 【已解决】PySpider中PyQuery如何匹配某个元素中多个属性值
- 【已解决】汽车之家车型车系数据：支持新版车系页面
- 【已解决】汽车之家车型车系数据：支持旧版车系页面
- 【已解决】汽车之家车型车系数据：抓取车型的详细参数配置
- 【未解决】汽车之家车型车系数据：想办法破解js和css加密的文字显示

- 【已解决】汽车之家车型车系数据：解决部分车型基本参数字段个数不一致问题
- 【已解决】汽车之家车型车系数据：优化去掉js加速抓取车型参数配置
- 【规避解决】汽车之家配置页面无法直接获取到文字数据
- 【已解决】汽车之家车型车系数据：carBrandId是空
- 【未解决】汽车之家车型车系数据：能源类型是空白的车型
- 【已解决】汽车之家车型车系数据：车身结构的值包含span标签
- 【未解决】汽车之家车型车系数据：有些参数字段的值包含span特殊标签
- 【已解决】汽车之家车型车系数据：抓取车型的详细参数配置
- 【已解决】车型车系数据缺失如红旗H5等部分车型数据

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-09-08 13:52:05

PySpider代码

核心文件： **autohome_20200902.py**

- 直接下载：[autohome_20200902.py](#)
- 贴出如下

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2020-09-02 21:22:43
# Project: autohome_20200902

import string
import re
import copy
import json

from lxml import etree

# from bs4 import BeautifulSoup

from pyspider.libs.base_handler import *

AutohomeHost = "https://www.autohome.com.cn"
CarSpecPrefix = "%s/spec" % AutohomeHost # "https://www.au

class Handler(BaseHandler):
    UserAgent_Mac_Chrome = "Mozilla/5.0 (Macintosh; Intel N
    crawl_config = {
        "headers": {
            "User-Agent": UserAgent_Mac_Chrome,
        }
    }

    def genSpecUrl(self, specId):
        # return "%s/%s" % (CarSpecPrefix, specId)
        return "%s/%s/" % (CarSpecPrefix, specId)

    def genConfigSpecUrl(self, specId):
        configSpecTemplate = "https://car.autohome.com.cn/c
        # https://car.autohome.com.cn/config/spec/43593.htm
        return configSpecTemplate % specId

    def to10KPrice(self, originPrice):
        tenKPrice = ""
        # 19.08 / '19.08' -> '19.08万'
        if isinstance(originPrice, str):
            tenKPrice = "%s万" % originPrice
        elif isinstance(originPrice, float):
            tenKPrice = "%.2f万" % originPrice
        elif isinstance(originPrice, int):
            tenKPrice = "%s.00万" % originPrice

        return tenKPrice

    def extractSpecId(self, specUrl):
```

```

carSpedId = ""
# https://www.autohome.com.cn/spec/41511/#pvareaid=
# https://www.autohome.com.cn/spec/2304/
foundSpecId = re.search("spec/(?P<specId>\d+)", spec)
print("foundSpecId=%s" % foundSpecId)
if foundSpecId:
    carSpedId = foundSpecId.group("specId")
    print("carSpedId=%s" % carSpedId)
return carSpedId

# @every(minutes=24 * 60)
def on_start(self):
    # autohomeEntryUrl = "https://www.autohome.com.cn/"
    # self.crawl(autohomeEntryUrl, callback=self.carBrand)
    for eachLetter in list(string.ascii_lowercase):
        letterUpper = eachLetter.upper()
        # # for debug
        # letterUpper = "H"
        print("letterUpper=%s" % letterUpper)
        self.crawl("https://www.autohome.com.cn/grade/",
                   save={"initials": letterUpper},
                   callback=self.gradCarHtmlPage)

@catch_status_code_error
def gradCarHtmlPage(self, response):
    print("gradCarHtmlPage: response=", response)

    # picSeriesItemList = response.doc('.rank-list-ul')
    # print("picSeriesItemList=", picSeriesItemList)
    # print("len(picSeriesItemList)=%s" % len(picSeriesItemList))
    # for each in picSeriesItemList:
    #     self.crawl(each.attr.href, callback=self.picSeries)

    saveDict = response.save
    print("saveDict=", saveDict)
    initials = saveDict["initials"]
    print("initials=", initials)
    respText = response.text
    # print("respText=", respText)

    """
    <dl id="33" olr="6">
        <dt><a href="//car.autohome.com.cn/price/brand-
            src="//car2.autoimg.cn/cardfs/series/g
                <div><a href="//car.autohome.com.cn/price/
        </dt>
    """
    # brandDoc = response.doc('dl dt')
    # print("brandDoc=%s" % brandDoc)
    # brandListDoc = response.doc('dl[id and olr] dt')
    # dlListDoc = response.doc('dl[id and olr]').items()

```

```

# dllListDoc = response.doc("dl[id*=''][orl*='']").items()
# dllListDoc = response.doc("dl[orl*='']").items()
# dllListDoc = response.doc("dl").items()
# dllListDoc = response.doc("dl:regex(id, \d+]").items()
# dllListDoc = response.doc("dl:regex(id, [0-9]+)").items()
# dllListDoc = response.doc("dl[id]").items()
dllListDoc = response.doc("dl[orl]").items()
print("type(dlListDoc)=%s" % type(dlListDoc))
dlList = list(dlListDoc)
print("len(dlList)=%s" % len(dlList))
print("dlList=%s" % dlList)
for curBrandIdx, eachDlDoc in enumerate(dlList):
    print("%s [%d] %s" % ('#' * 30, curBrandIdx, '#' * 30))

    dtDoc = eachDlDoc.find("dt")
    # print("dtDoc=%s" % dtDoc)
    # <a href="//car.autohome.com.cn/price/brand-33.html"
    brandLogoDoc = dtDoc.find('a img')
    # print("brandLogoDoc=%s" % brandLogoDoc)
    carBrandLogoUrl = brandLogoDoc.attrs["src"]
    print("carBrandLogoUrl=%s" % carBrandLogoUrl)
    # <div><a href="//car.autohome.com.cn/price/brand-33.html"
    brandADoc = dtDoc.find('div a')
    print("brandADoc=%s" % brandADoc)
    # <a href="https://car.autohome.com.cn/price/brand-33.html"
    carBrandName = brandADoc.text()
    print("carBrandName=%s" % carBrandName)
    carBrandUrl = brandADoc.attrs["href"]
    print("carBrandUrl=%s" % carBrandUrl)
    carBrandId = ""
    foundBrandId = re.search("brand-(?P<carBrandId>\d+)", carBrandName)
    # print("foundBrandId=%s" % foundBrandId)
    if foundBrandId:
        carBrandId = foundBrandId.group("carBrandId")
    print("carBrandId=%s" % carBrandId) # 63

    # <div class="h3-tit"><a href="//car.autohome.com.cn/price/brand-33.html"
    merchantDocGenerator = response.doc("dd div[class='list-item']")
    # ddDoc = eachDlDoc.find("dd")
    ddDoc = eachDlDoc.find("dd")
    # print("ddDoc=%s" % ddDoc)

    merchantDocGenerator = ddDoc.items("div[class='list-item']")
    merchantDocList = list(merchantDocGenerator)
    # print("merchantDocList=%s" % merchantDocList)
    merchantDocLen = len(merchantDocList)
    print("merchantDocLen=%s" % merchantDocLen)

    # <ul class="rank-list-ul" 0>
    merchantRankDocGenerator = response.doc("dd div[class='list-item']")
    # merchantRankDocGenerator = response.doc("dd div[class='list-item']")

```

```

merchantRankDocGenerator = ddDoc.items("ul[class='list-group'] li")
merchantRankDocList = list(merchantRankDocGenerator)
# print("merchantRankDocList=%s" % merchantRankDocList)
merchantRankDocListLen = len(merchantRankDocList)
print("merchantRankDocListLen=%s" % merchantRankDocListLen)

for curIdx, merchantItem in enumerate(merchantRankDocList):
    # for curIdx, merchantItem in enumerate(merchantRankDocList):
        # print("%s" % "="*80)
        print("%s [%d] %s" % ('='*30, curIdx, '='*30))
        # print("type(merchantItem)=%s" % type(merchantItem))
        # print("[%d] merchantItem=%s" % (curIdx, merchantItem))
        # print("[%d] merchantItem=%s" % (curIdx, merchantItem))
        carMerchantName = merchantItem.text()
        print("carMerchantName=%s" % carMerchantName)
        merchantItemAttr = merchantItem.attrs
        # print("merchantItemAttr=%s" % merchantItemAttr)
        carMerchantUrl = merchantItemAttr["href"]
        print("carMerchantUrl=%s" % carMerchantUrl)

        # curSubBrandDict = {
        #     "brandName": brandName,
        #     "carBrandLogoUrl": carBrandLogoUrl,
        #     "carMerchantName": carMerchantName,
        #     "carMerchantUrl": carMerchantUrl,
        # }
        # self.send_message(self.project_name, curSubBrandDict)

    merchantRankDoc = merchantRankDocList[curIdx]
    # print("merchantRankDoc=%s" % merchantRankDoc)
    # print("type(merchantRankDoc)=%s" % type(merchantRankDoc))

    # type(merchantRankDoc)=<class 'lxml.html.html_element.HTMLParserElement'>
    # merchantRankHtml = etree.tostring(merchantRankDoc)

    # type(merchantRankDoc)=<class 'pyquery.pyquery.PyQuery'>
    # merchantRankHtml = merchantRankDoc.html()

    # print("merchantRankHtml=%s" % merchantRankHtml)

    # <li id="s3170">
    # carSeriesDocGenerator = merchantRankDoc.select("li[id='s3170'] ul li")
    # carSeriesDocGenerator = merchantRankDoc.select("li[id='s3170'] ul li")
    carSeriesDocGenerator = merchantRankDoc.items("li[id='s3170'] ul li")
    # print("type(carSeriesDocGenerator)=%s" % type(carSeriesDocGenerator))
    carSeriesDocList = list(carSeriesDocGenerator)
    # print("type(carSeriesDocList)=%s" % type(carSeriesDocList))
    # print("carSeriesDocList=%s" % carSeriesDocList)
    carSeriesDocListLen = len(carSeriesDocList)
    # print("carSeriesDocListLen=%s" % carSeriesDocListLen)

```

```
        for curSeriesIdx, eachCarSeriesDoc in enumerate(carSeriesList):
            print("%s [%d] %s" % ('-'*30, curSeriesIdx, eachCarSeriesDoc))
            # print("[%d] eachCarSeriesDoc=%s" % (curSeriesIdx, eachCarSeriesDoc))
            # print("type(eachCarSeriesDoc)=%s" % type(eachCarSeriesDoc))
            # <h4><a href="http://www.autohome.com.cn/310/1/>
            carSeriesInfoDoc = eachCarSeriesDoc.find('div', class_='car-series-item')
            # print("type(carSeriesInfoDoc)=%s" % type(carSeriesInfoDoc))
            # print("carSeriesInfoDoc=%s" % carSeriesInfoDoc)
            carSeriesName = carSeriesInfoDoc.text()
            print("carSeriesName=%s" % carSeriesName)
            carSeriesUrl = carSeriesInfoDoc.attr('href')
            print("carSeriesUrl=%s" % carSeriesUrl)

            # <div>指导价: <a class="red" href="http://www.autohome.com.cn/310/1/1/>
            # 厂商指导价=厂商建议零售价格=MSRP=Manufacturers Suggested Retail Price
            carSeriesMsrpDoc = eachCarSeriesDoc.find('div', class_='car-series-item')
            carSeriesMsrpDoc = eachCarSeriesDoc.find('div', class_='car-series-item')
            # print("carSeriesMsrpDoc=%s" % carSeriesMsrpDoc)
            carSeriesMsrp = carSeriesMsrpDoc.text()
            print("carSeriesMsrp=%s" % carSeriesMsrp)
            carSeriesMsrpUrl = carSeriesMsrpDoc.attr('href')
            print("carSeriesMsrpUrl=%s" % carSeriesMsrpUrl)

            carSeriesDict = {
                "carBrandName": carBrandName,
                "carBrandId": carBrandId,
                "carBrandLogoUrl": carBrandLogoUrl,
                "carMerchantName": carMerchantName,
                "carMerchantUrl": carMerchantUrl,
                "carSeriesName": carSeriesName,
                "carSeriesUrl": carSeriesUrl,
                "carSeriesMsrp": carSeriesMsrp,
                "carSeriesMsrpUrl": carSeriesMsrpUrl
            }
            # self.send_message(self.project_name,
            self.crawl(carSeriesUrl,
                        callback=self.carSeriesDetailPage,
                        save=carSeriesDict,
            )

@catch_status_code_error
def carSeriesDetailPage(self, response):
    print("in carSeriesDetailPage")
    carSeriesDict = response.save
    print("carSeriesDict=%s" % carSeriesDict)

    carModelDict = copy.deepcopy(carSeriesDict)

    carSeriesUrl = response.url
    print("carSeriesUrl=%s" % carSeriesUrl)
```

```

carSeriesMainImgUrl = ""
# carSeriesId = ""
carSeriesLevelId = ""
carSeriesMsrp = ""
carSeriesMinPrice = ""
carSeriesMaxPrice = ""

# carSeriesUrl=https://www.autohome.com.cn/2123/#level
foundSeriesId = re.search("www\.autohome\.com\.cn/|", carSeriesUrl)
carSeriesId = foundSeriesId.group("seriesId")
# carSeriesId = int(carSeriesId)
print("carSeriesId=%s" % carSeriesId) # 2123
carModelDict["carSeriesId"] = carSeriesId

carSeriesHtml = response.text
print("type(carSeriesHtml)=%s" % type(carSeriesHtml))
# print("carSeriesHtml=%s" % carSeriesHtml)

foundLevelId = re.search("var\s+levelid\s+=" , carSeriesHtml)
print("foundLevelId=%s" % foundLevelId)
isNewLayoutHtml = bool(foundLevelId)
print("isNewLayoutHtml=%s" % isNewLayoutHtml)
foundShowCityId = re.search("var\s+showCityId\s+=" , carSeriesHtml)
print("foundShowCityId=%s" % foundShowCityId)
isOldLayoutHtml = bool(foundShowCityId)
print("isOldLayoutHtml=%s" % isOldLayoutHtml)

if isOldLayoutHtml:
    # Q开头
    # https://www.autohome.com.cn/grade/carhtml/q.htm
    # -
    # 东风悦达起亚-千里马
    # https://www.autohome.com.cn/142/#levelsources
    # 其他:
    #
    # 一汽丰田-花冠
    # https://www.autohome.com.cn/109/#levelsources
    #
    # 起亚-起亚 SUV
    # https://www.autohome.com.cn/4550/#levelsources

    .....
<div class="car_detail " id="tab1-2">
    <div class="models">
        <!--年代-->
        <div class="header">
            <div class="car_price">
                <span class="years">2005款</span>
                <span class="price">指导价 (停售)</span>
                <span class="price">二手车价格:</span>
            ...
        </div>
    </div>
</div>

```

```

<div class="car_detail current" id="tab1-1">
    <div class="models">
        <!--年代-->
        <div class="header">
            <div class="car_price">
                <span class="years">2006款</span>
                <span class="price">指导价 (停售)</span>
            <div style="display: flex; justify-content: space-between; margin-top: 10px;">
                <div><span>参数配置</span></div>
                <div><span>更多车型</span></div>
            </div>
        </div>
    </div>
</div>

carDetailDivGenerator = response.doc("div[class='car_detail current']")
print("carDetailDivGenerator=%s" % carDetailDivGenerator)
carDetailDivList = list(carDetailDivGenerator)
print("carDetailDivList=%s" % carDetailDivList)
for curDivIdx, eachCarDetailDoc in enumerate(carDetailDivList):
    print("%s [%d] %s" % ('#' * 30, curDivIdx, '#' * 30))

    if curDivIdx == 0:
        # use first car model as series: main :
        <div class="models_info">
            <dl class='models_pics'>
                <dt><a href='//car.autohome.com.cn/cars/%s'><img alt='%s' src='https://car0.autoimg.com/m/11/2006/06/2006061011284167568_180x180.jpg' height='180' /></a></dt>
            </dl>
        </div>

        # modelMainImgDocListGenerator = response.doc("div[class='main_img']")
        # modelMainImgDocList = list(modelMainImgDocListGenerator)
        # firstModelMainImgDoc = modelMainImgDocList[0]
        firstModelMainImgDoc = eachCarDetailDoc
        firstModelMainImgUrl = firstModelMainImgDoc['url']
        print("firstModelMainImgUrl=%s" % firstModelMainImgUrl)
        carSeriesMainImgUrl = firstModelMainImgUrl
        print("carSeriesMainImgUrl=%s" % carSeriesMainImgUrl)

        carModelDict["carSeriesMainImgUrl"] = carSeriesMainImgUrl

    # <div class="car_price">
    #     <span class="price">指导价 (停售)</span>
    carPriceStrongDocGenerator = eachCarDetailDoc
    print("carPriceStrongDocGenerator=%s" % carPriceStrongDocGenerator)
    if carPriceStrongDocGenerator:
        carPriceStrongDocList = list(carPriceStrongDocGenerator)
        print("carPriceStrongDocList=%s" % carPriceStrongDocList)
        carPriceStrongDoc = carPriceStrongDocList[0]
        print("carPriceStrongDoc=%s" % carPriceStrongDoc)
        carPriceMinMax = carPriceStrongDoc['text']
        print("carPriceMinMax=%s" % carPriceMinMax)
        if carPriceMinMax:
            foundMinMax = re.search("(?P<min>[\d.]+)(?P<max>[\d.]+)", carPriceMinMax)
            print("foundMinMax=%s" % foundMinMax)
            if foundMinMax:

```

```
        minPrice = foundMinMax.grou
        print("minPrice=%s" % minPi
        minPriceFloat = float(minPi
        print("minPriceFloat=%s" %
        maxPrice = foundMinMax.grou
        print("maxPrice=%s" % maxPi
        maxPriceFloat = float(maxPi
        print("maxPriceFloat=%s" %
        averageMsrpPrice = (minPrice
        print("averageMsrpPrice=%s"

        # carSeriesMsrp = "%.2f万"
        carSeriesMsrp = self.to10Kf
        print("carSeriesMsrp=%s" %
        # carSeriesMinPrice = "%.2f"
        carSeriesMinPrice = self.to10Kf
        print("carSeriesMinPrice=%s"
        # carSeriesMaxPrice = "%.2f"
        carSeriesMaxPrice = self.to10Kf
        print("carSeriesMaxPrice=%s"

        carModelDict["carSeriesMsrp"]
        carModelDict["carSeriesMinPrice"]
        carModelDict["carSeriesMaxPrice"]
        print("")

        self.processSingleCarDetailDiv(carModelDict)

    elif isNewLayoutHtml:
        # https://www.autohome.com.cn/3170/#levelsources

        """
        <div class="information-pic">
            <div class="pic-main">
                ...
                <picture>
                    ...
                    
            ...
        </div>
        """

        mainImgDoc = response.doc("div[class='information-pic']
        print("mainImgDoc=%s" % mainImgDoc)
        carSeriesMainImgUrl = mainImgDoc.attr["src"]
        print("carSeriesMainImgUrl=%s" % carSeriesMainImgUrl)
        carModelDict["carSeriesMainImgUrl"] = carSeriesMainImgUrl

        """
        <script type="text/javascript">
            ...
        </script>
        """
```

```

var seriesid = '2123';
var seriesname='哈弗H6';
var yearid = '0';
var brandid = '181';
var levelid = '17';
varlevelname='紧凑型SUV';
var fctid = '4';
var SeriesMinPrice='9.80';
var SeriesMaxPrice='14.10';
.....
infoKeyList = [
    "seriesid",
    # "seriesname", # has got
    # "yearid", # no need
    "brandid",
    "levelid",
    "levelname",
    # "fctid", # unknown meaning
    "SeriesMinPrice",
    "SeriesMaxPrice",
]
InfoDict = {}
for eachInfoKey in infoKeyList:
    curPattern = "var\s+%\s*\s*=\s*(?P<infoValue"
    print("curPattern=%s" % curPattern)
    foundInfo = re.search(curPattern, carSeries)
    print("foundInfo=%s" % foundInfo)
    # if foundInfo:
    #     infoValue = foundInfo.group("infoValue")
    #     print("infoValue=%s" % infoValue)
    InfoDict[eachInfoKey] = infoValue
    print("InfoDict=%s" % InfoDict)

# if "seriesid" in InfoDict:
carSeriesId = InfoDict["seriesid"] # 2123
carModelDict["carSeriesId"] = carSeriesId
# carModelDict["carSeriesName"] = InfoDict["seriesname"]
# if "brandid" in InfoDict:
carModelDict["carBrandId"] = InfoDict["brandid"]
# if "levelid" in InfoDict:
carSeriesLevelId = InfoDict["levelid"] # 17
carModelDict["carSeriesLevelId"] = carSeriesLevelId
# if "levelname" in InfoDict:
carModelDict["carSeriesLevelName"] = InfoDict["levelname"]
# if "SeriesMinPrice" in InfoDict:
carSeriesMinPrice = InfoDict["SeriesMinPrice"]
carModelDict["carSeriesMinPrice"] = self.to10K(carSeriesMinPrice)
# if "SeriesMaxPrice" in InfoDict:
carSeriesMaxPrice = InfoDict["SeriesMaxPrice"]
carModelDict["carSeriesMaxPrice"] = self.to10K(carSeriesMaxPrice)

```

```

    """


...
- 更多
- ...
- ...
- ...

...


haltADocGenerator = response.doc("li[class='more-dropdown']")
print("type(haltADocGenerator)=%s" % type(haltADocGenerator))
print("haltADocGenerator=%s" % haltADocGenerator)
haltADocList = list(haltADocGenerator)
print("haltADocList=%s" % haltADocList)
for curLiIdx, eachHatADoc in enumerate(haltADocList):
    print("%s [%d] %s" % ('#' * 30, curLiIdx, eachHatADoc))
    self.processSingleHaltA(carModelDict, eachHatADoc)

# """
# <div class="information-summary">
#     <dl class="information-price">
#         ...
#             <dd class="type">
#                 <span class="type__item">紧凑型车</span>
#             </dd>
#         ...
#     </dl>
# </div>
# carLevelDoc = response.doc("div[class='information-summary']")
# print("carLevelDoc=%s" % carLevelDoc)
# carSeriesLevelName = carLevelDoc.text()
# print("carSeriesLevelName=%s" % carSeriesLevelName)
# carModelDict["carSeriesLevelName"] = carSeriesLevelName

carSeriesContentDoc = response.doc("div[class='content__list-item']")
# print("carSeriesContentDoc=%s" % carSeriesContentDoc)
# carSpecWrapDoc = carSeriesContentDoc.find("div")
# carSpecWrapDoc = carSeriesContentDoc.find("div")
# carSpecWrapDocGenerator = carSeriesContentDoc.find("div")
# print("carSpecWrapDocGenerator=%s" % carSpecWrapDocGenerator)
carSpecWrapDocList = list(carSpecWrapDocGenerator)
print("carSpecWrapDocList=%s" % carSpecWrapDocList)
for curSpecWrapIdx, eachSpecWrapDoc in enumerate(carSpecWrapDocList):
    print("%s [%d] %s" % ('#' * 30, curSpecWrapIdx, eachSpecWrapDoc))
    self.processSingleSpecWrapDiv(carModelDict, eachSpecWrapDoc)

def processSingleCarDetailDiv(self, carModelDict, curCarModelGroupDict):
    print("in processSingleCarDetailDiv()")
    curCarModelGroupDict = copy.deepcopy(carModelDict)

```

```
# <span class="years">2006款</span>
modelYearDoc = curCarDetailDoc.find("span[class='ye
print("modelYearDoc=%s" % modelYearDoc)
carModelYear = modelYearDoc.text()
print("carModelYear=%s" % carModelYear)
curCarModelGroupDict["carModelYear"] = carModelYe
    """
    """
<div class="modelswrap">
    <!-- 信息 start -->
    <div class="models_info">
        <dl class='models_prop'>
            <dt>发动机: </dt>
            <dd><span>1.3L</span><span>1.6L</span></dd>
        </dl>
        <dl class='models_prop'>
            <dt>变速箱: </dt>
            <dd><span>手动</span><span>自动</span></dd>
            <dt>车身结构: </dt>
            <dd><span>三厢</span></dd>
        </dl>
    """
# modelsPropDdList = curCarDetailDoc.find("div[class='modelswrap']")
modelsPropDdGenerator = curCarDetailDoc.items("div
print("modelsPropDdGenerator=%s" % modelsPropDdGen
modelsPropDdList = list(modelsPropDdGenerator)
print("modelsPropDdList=%s" % modelsPropDdList)
engineValueDoc = modelsPropDdList[0]
print("engineValueDoc=%s" % engineValueDoc)
carModelEngine = engineValueDoc.text()
print("carModelEngine=%s" % carModelEngine)

gearBoxValueDoc = modelsPropDdList[1]
print("gearBoxValueDoc=%s" % gearBoxValueDoc)
carModelGearBox = gearBoxValueDoc.text()
print("carModelGearBox=%s" % carModelGearBox)

bodyStructureValueDoc = modelsPropDdList[2]
print("bodyStructureValueDoc=%s" % bodyStructureVa
carModelBodyStructure = bodyStructureValueDoc.text()
print("carModelBodyStructure=%s" % carModelBodyStru
    """
    """
curCarModelGroupDict["carModelGearBox"] = carModelG
curCarModelGroupDict["carModelDriveType"] = ""
curCarModelGroupDict["carModelBodyStructure"] = carM
    """
    """
# curCarModelGroupDict["carModelEnvStandard"] = ""
# carModelPower = carModelEngine
# print("carModelPower=%s" % carModelPower)
# curCarModelGroupDict["carModelPower"] = carModelP
curCarModelGroupDict["carModelEngine"] = carModelEng
```

```

carModelGroupName = "%s %s %s" % (carModelEngine,
print("carModelGroupName=%s" % carModelGroupName)
curCarModelGroupDict["carModelGroupName"] = carModelGroupName

"""
<table class='models_tab tableline' cellspacing='0'
<tr>
    <td class='name_d'>
        <div class='name'><a title='2006款 1.6L
    </td>
    <td class='price_d'>
        <div class='price01'>8.18万</div>
    </td>
"""
modelsTrDocGenerator = curCarDetailDoc.items("table")
print("modelsTrDocGenerator=%s" % modelsTrDocGenerator)
modelsTrDocList = list(modelsTrDocGenerator)
print("modelsTrDocList=%s" % modelsTrDocList)
for curTabIndex, eachModelTrDoc in enumerate(modelsTrDocList):
    print("%s [%d] %s" % ('='*30, curTabIndex, '='*30))
    self.processSingleModelsTr(curCarModelGroupDict)

def processSingleModelsTr(self, curCarModelGroupDict, curTrCarModeDict):
    curModelTrDoc = copy.deepcopy(curCarModelGroupDict)
    # print("curModelTrDoc=%s" % curModelTrDoc)
    nameADoc = curModelTrDoc.find("td[class='name_d']")
    print("nameADoc=%s" % nameADoc)
    carModelName = nameADoc.text()
    print("carModelName=%s" % carModelName)

    carModelSpecUrl = nameADoc.attr["href"]
    # bug -> wrong url:
    # https://www.autohome.com.cn/142/spec/2304/
    # need repace
    # https://www.autohome.com.cn/142/spec/2304/
    # to
    # https://www.autohome.com.cn/spec/2304/
    foundSpecId = re.search("spec/(?P<specId>\d+)", carModelSpecUrl)
    carModelSpecId = foundSpecId.group("specId")
    print("carModelSpecId=%s" % carModelSpecId) # 2304
    carModelSpecUrl = self.genSpecUrl(carModelSpecId)
    print("carModelSpecUrl=%s" % carModelSpecUrl)

    priceDivDoc = curModelTrDoc.find("td[class='price_d']")
    print("priceDivDoc=%s" % priceDivDoc)
    carModelMsrp = priceDivDoc.text()
    print("carModelMsrp=%s" % carModelMsrp)

    curTrCarModeDict["carModelName"] = carModelName
    curTrCarModeDict["carModelSpecUrl"] = carModelSpecUrl

```

```

        curTrCarModeDict["carModelMsrp"] = carModelMsrp

        self.processSingleResult(curTrCarModeDict)

    def processSingleHaltA(self, carModelDict, curHatADoc):
        curHaltCarDict = copy.deepcopy(carModelDict)
        print("curHatADoc=%s" % curHatADoc)
        yearName = curHatADoc.text()
        print("yearName=%s" % yearName)
        yearId = curHatADoc.attr["data-yearid"]
        print("yearId=%s" % yearId)

        # getHaltSpecUrl = "https://www.autohome.com.cn/ashx/car/Spec_Lis...
        carSeriesId = curHaltCarDict["carSeriesId"]
        carSeriesLevelId = curHaltCarDict["carSeriesLevelId"]
        if carSeriesId and carSeriesLevelId:
            getHaltSpecUrl = "https://www.autohome.com.cn/ashx/car/Spec_Lis...
                # https://www.autohome.com.cn/ashx/car/Spec_Lis...
                print("getHaltSpecUrl=%s" % getHaltSpecUrl)
                self.crawl(getHaltSpecUrl,
                           callback=self.haltCarSpecCallback,
                           save=curHaltCarDict,
                           )
        else:
            pass

    def processSingleSpecWrapDiv(self, curCarModelDict, curSpecWrapDoc):
        curSpecWrapCarDict = copy.deepcopy(curCarModelDict)
        # print("curSpecWrapDoc=%s" % curSpecWrapDoc)
        .....
        <!--即将上市 start-->
        <div class="spec-wrap active" id="specWrap-1">

            <dl class="halt-spec">
                <dt>
                    <div class="spec-name">
                        <span>参数配置未公布</span>
                    </div>
                <dt>
                    <div class="spec-name">
                        <span>1.5升 涡轮增压 169马力 国VI</span>
                    </div>
                .....
            # dlDoc = curSpecWrapDoc.find("dl[class='']")
            # dlDoc = curSpecWrapDoc.find("dl")
            dlListDocGenerator = curSpecWrapDoc.items("dl")
            print("dlListDocGenerator=%s" % dlListDocGenerator)
            dlDocList = list(dlListDocGenerator)
            print("dlDocList=%s" % dlDocList)
            for curDlIdx, eachDlDoc in enumerate(dlDocList):
                print("%s [%d] %s" % ('='*30, curDlIdx, '='*30))

```

```

        self.processSingleSpecDl(curSpecWrapCarDict, eachDlDoc)

    def processSingleSpecDl(self, curSpecWrapCarDict, curDlDoc):
        curDlCarDict = copy.deepcopy(curSpecWrapCarDict)
        # print("curDlDoc=%s" % curDlDoc)
        """
        <dt>
            <div class="spec-name">
                <span>1.5升 涡轮增压 169马力 国VI</span>
        """
        dtDoc = curDlDoc.find("dt")
        # print("dtDoc=%s" % dtDoc)
        groupSpecNameSpanDoc = dtDoc.find("div[class='spec-name']")
        print("groupSpecNameSpanDoc=%s" % groupSpecNameSpanDoc)
        carModelGroupName = ""
        if groupSpecNameSpanDoc:
            carModelGroupName = groupSpecNameSpanDoc.text()
            print("carModelGroupName=%s" % carModelGroupName)

        curDlCarDict["carModelGroupName"] = carModelGroupName

        # <dd data-sift1="2020款" data-sift2="国VI" data-sift3="涡轮增压">
        ddListDoc = curDlDoc.items("dd")
        print("ddListDoc=%s" % ddListDoc)
        for curDdIdx, eachDdDoc in enumerate(ddListDoc):
            print("%s [%d] %s" % ('-'*30, curDdIdx, '-'*30))
            self.processSingleSiftDd(curDlCarDict, eachDdDoc)

    def processSingleSiftDd(self, curDlCarDict, curDdDoc):
        print("in processSingleSiftDd")
        curDdCarDict = copy.deepcopy(curDlCarDict)

        curDdAttr = curDdDoc.attr
        """
        正常:
            <dd data-sift1="2020款" data-sift2="国VI" data-sift3="涡轮增压">
            ...
        特殊:
        无sift:
            <dd data-electricspecid="47050">
            电动的 sift位置不同:
                https://www.autohome.com.cn/5240/
                <dd data-electricspecid="42875" data-sift1="1">
            混动 sift位置也不同:
                https://www.autohome.com.cn/4460/
                <dd data-electricspecid="37077" data-sift1="1">
            ...
        # print("curDdAttr=%s" % curDdAttr)
        # carModelDataSift1 = curDdAttr["data-sift1"]
        # print("carModelDataSift1=%s" % carModelDataSift1)
        carModelYear = curDdAttr["data-sift1"]
        """

```

```

print("carModelYear=%s" % carModelYear)
carModelDataSift2 = curDdAttr["data-sift2"]
print("carModelDataSift2=%s" % carModelDataSift2)
carModelDataSift3 = curDdAttr["data-sift3"]
print("carModelDataSift3=%s" % carModelDataSift3)
carModelDataSift4 = curDdAttr["data-sift4"]
print("carModelDataSift4=%s" % carModelDataSift4)

curDdCarDict["carModelYear"] = carModelYear
# curDdCarDict["carModelEnvStandard"] = carModelEnvStandard
# curDdCarDict["carModelPower"] = carModelPower
# curDdCarDict["carModelGearBox"] = carModelGearBox
# curDdCarDict["carModelDataSift1"] = carModelDataSift1
curDdCarDict["carModelDataSift2"] = carModelDataSift2
curDdCarDict["carModelDataSift3"] = carModelDataSift3
curDdCarDict["carModelDataSift4"] = carModelDataSift4

.....
<div class="spec-name">
    <div class="name-param">
        <p data-gcjid="41511" id="spec_41511">
            <a href="/spec/41511/#pvareaid=3454492" class="athm-link">
                <span class="athm-badge athm-badge--green">已上架</span>
            <span class="athm-badge athm-badge--orange">已上架</span>
            <p><span class="type-default">前置前驱</span></p>
        </div>
    </div>
.....
specNameDoc = curDdDoc.find("div[class='spec-name']")
# print("specNameDoc=%s" % specNameDoc)
specADoc = specNameDoc.find("p a[class='name']")
# print("specADoc=%s" % specADoc)
carmodelName = specADoc.text()
print("carmodelName=%s" % carmodelName) # 2020款 1.5T
carModelSpecUrl = specADoc.attr["href"]
print("carModelSpecUrl=%s" % carModelSpecUrl) # http://www.athm.com.cn/spec/41511/
typeDefaultListDoc = specNameDoc.items("p span[class='type-default']")
print("typeDefaultListDoc=%s" % typeDefaultListDoc)
typeDefaultList = list(typeDefaultListDoc)
print("typeDefaultList=%s" % typeDefaultList)
carModelDriveType = ""
carModelGearBox = ""
if typeDefaultList:
    .....
    正常:
        <p>
            <span class="type-default">前置前驱</span>
            <span class="type-default">7挡双离合</span>
        </p>
    特殊:

```

```
https://www.autohome.com.cn/4605/
<p>
    <span class="type-default">电动</span>
    <span class="type-default">前置前驱</span>
    <span class="type-default">AMT (组合10挡)</span>
</p>

https://www.autohome.com.cn/4460/
<p>
    <span class="type-default">电动</span>
    <span class="type-default">前置四驱</span>
    <span class="type-default">8挡手自一体</span>
</p>

https://www.autohome.com.cn/5240/
<p>
    <span class="type-default">电动</span>
    <span class="type-default">前置前驱</span>
    <span class="type-default">电动车单速变速箱</span>
</p>
.....
# spanTypeDefault0 = typeDefaultList[0]
spanTypeDefault0 = typeDefaultList[-2]
print("spanTypeDefault0=%s" % spanTypeDefault0)
carModelDriveType = spanTypeDefault0.text()
print("carModelDriveType=%s" % carModelDriveType)
# 前置前驱
# 前置四驱
# spanTypeDefault1 = typeDefaultList[1]
spanTypeDefault1 = typeDefaultList[-1]
print("spanTypeDefault1=%s" % spanTypeDefault1)
carModelGearBox = spanTypeDefault1.text()
print("carModelGearBox=%s" % carModelGearBox)
# 7挡双离合
# AMT (组合10挡)
# 8挡手自一体
# 电动车单速变速箱

curDdCarDict["carModelName"] = carModelName
if not curDdCarDict["carModelYear"]:
    foundYearType = re.search("(?P<yearType>\d{4}款")
    if foundYearType:
        yearType = foundYearType.group("yearType")
        print("yearType=%s" % yearType)
        carModelYear = yearType
        print("extract year=%s from modelName=%s" % (yearType, carModelName))
        curDdCarDict["carModelYear"] = carModelYear

curDdCarDict["carModelSpecUrl"] = carModelSpecUrl
curDdCarDict["carModelDriveType"] = carModelDriveType
curDdCarDict["carModelGearBox"] = carModelGearBox
```

```
....  
<div class="spec-guidance">  
    <p class="guidance-price">  
        <span>10.40万</span>  
        <a href="//j.autohome.com.cn/pc/carcounter/">  
    </p>  
</div>  
  
<div class="spec-guidance">  
    <p class="guidance-price">  
        <span><span>暂无</span></span>  
    ....  
specGuidanceDoc = curDdDoc.find("div[class='spec-guidance']")  
# print("specGuidanceDoc=%s" % specGuidanceDoc)  
guidancePriceSpanDoc = specGuidanceDoc.find("p[class='guidance-price']")  
# print("guidancePriceSpanDoc=%s" % guidancePriceSpanDoc)  
carModelMsrp = guidancePriceSpanDoc.text()  
print("carModelMsrp=%s" % carModelMsrp)  
curDdCarDict["carModelMsrp"] = carModelMsrp  
  
self.processSingleResult(curDdCarDict)  
  
@catch_status_code_error  
def haltCarSpecCallback(self, response):  
    prevCarModelDict = response.save  
    carModelDict = copy.deepcopy(prevCarModelDict)  
    print("carModelDict=%s" % carModelDict)  
  
    respJson = response.json  
    print("respJson=%s" % respJson)  
  
    ....  
    [  
        {  
            "name": "1.5升 涡轮增压 169马力",  
            "speclist": [  
                {  
                    "specid": 36955,  
                    "specname": "2019款 红标 1.5GDIT 自动",  
                    "specstate": 40,  
                    "minprice": 102000,  
                    "maxprice": 102000,  
                    "fueltype": 1,  
                    "fueltypedetail": 1,  
                    "driveform": "前置前驱",  
                    "drivetype": "前驱",  
                    "gearbox": "7挡双离合",  
                    "evflag": "",  
                    "newcarflag": "",  
                    "subsidy": ""},  
....
```

```

        "paramisshow": 1,
        "videoid": 0,
        "link2sc": "http://www.che168.com/c",
        "price2sc": "7.58万",
        "price": "10.20万",
        "syear": 2019
    }, {
        "specid": 36956,
        "specname": "2019款 红标 1.5GDI 自动精英型",
        "specstate": 40,
        "minprice": 109000,
        "maxprice": 109000,
        "fueltype": 1,
        "fueltypedetail": 1,
        "driveform": "前置前驱",
        "drivetype": "前驱",
        "gearbox": "7挡双离合",
        "evflag": "",
        "newcarflag": "",
        "subsidy": "",
        "paramisshow": 1,
        "videoid": 0,
        "link2sc": "",
        "price2sc": "",
        "price": "10.90万",
        "syear": 2019
    },
    ...
    ...
}

if respJson:
    for eachModelGroupDict in respJson:
        modelName = eachModelGroupDict["name"]
        modelSpecList = eachModelGroupDict["speclist"]
        for eachModelDict in modelSpecList:
            curCarModelDict = copy.deepcopy(carModelDict)

            carModelYear = "%s款" % eachModelDict["year"]
            # carModelSpecUrl = "%s/%s" % (CarSpecUrl, carModelYear)
            carModelSpecUrl = self.genSpecUrl(eachModelDict)

            curCarModelDict["carModelGroupName"] = modelName
            curCarModelDict["carModelYear"] = carModelYear
            # curCarModelDict["carModelEnvStandard"] = ""
            # curCarModelDict["carModelPower"] = ""
            curCarModelDict["carModelDriveType"] = eachModelDict["driveform"]
            curCarModelDict["carModelGearBox"] = eachModelDict["gearbox"]
            curCarModelDict["carModelName"] = eachModelDict["specname"]
            curCarModelDict["carModelSpecUrl"] = carModelSpecUrl
            curCarModelDict["carModelMsrp"] = eachModelDict["price"]

            self.processSingleResult(curCarModelDict)

```

```

@catch_status_code_error
def processCarSpecConfig(self, curCarModelDict):
    print("in processCarSpecConfig")
    carModelDict = copy.deepcopy(curCarModelDict)
    print("carModelDict=%s" % carModelDict)
    carModelSpecUrl = carModelDict["carModelSpecUrl"]
    print("carModelSpecUrl=%s" % carModelSpecUrl)
    carModelSpecId = self.extractSpecId(carModelSpecUrl)
    print("carModelSpecId=%s" % carModelSpecId)
    carModelDict["carModelSpecId"] = carModelSpecId # +
    carConfigSpecUrl = self.genConfigSpecUrl(carModelSpecUrl)
    # https://car.automobile.com.cn/config/spec/43593.htm
    print("carConfigSpecUrl=%s" % carConfigSpecUrl)

    self.crawl(carConfigSpecUrl,
               # fetch_type="js",
               callback=self.carConfigSpecCallback,
               save=carModelDict,
               )
}

def getItemFirstValue(self, inputContent, itemIndex):
    print("in getItemFirstValue")
    # firstItemValue = self.extractTrFirstTdValue(inputContent)
    firstItemValue = self.extractDictListFirstValue(inputContent)
    return firstItemValue

def extractDictListFirstValue(self, paramItemDictList,
                             itemIndex):
    [
        ...
        {
            "id": 1149,
            "name": "能源类型",
            "pnid": "1_-1",
            "valueitems": [
                {"specid": 39893,
                 "value": "纯电动"},
                {"specid": 42875,
                 "value": "纯电动"}
            ]
        }
        ...
        {
            "id": 1292,
            "name": "<span class='hs_kw39_configpl'></span>",
            "pnid": "1_-1",
            "valueitems": [
                {"specid": 39893,
                 "value": "0.6"}
            ]
        }
    ]

```

```
        }, {
            "specid": 42875,
            "value": "0.6"
        }]
    },
    ...
    ;
    {
        "id": 1255,
        "name": "整车<span class='hs_kw36_configpl'></span>",
        "pnid": "1_-1",
        "valueitems": [
            {
                "specid": 39893,
                "value": "三<span class='hs_kw7_configpl'></span>"
            }, {
                "specid": 42875,
                "value": "三<span class='hs_kw7_configpl'></span>"
            }
        ]
    }
}
"""
paramItemDict = paramItemDictList[itemIndex]
print("paramItemDict=%s" % paramItemDict)
firstItemValue = self.extractValueItemsValue(paramItemDict)
print("firstItemValue=%s" % firstItemValue)
return firstItemValue

def extractValueItemsValue(self, curItemDict, valueIndex):
    """
    :param curItemDict: dict
    :param valueIndex: int
    :return: str
    """
    valueItemList = curItemDict["valueitems"]
    print("valueItemList=%s" % valueItemList)
    # firstItemDict = valueItemList[0]
    specificItemDict = valueItemList[valueIndex]
    print("specificItemDict=%s" % specificItemDict)
    # specificItemDict={'specid': 43593, 'value': '<span class='hs_kw7_configpl'></span>'}
```

```
specificItemValue = specificItemDict["value"]
# specificItemValue=<span class='hs_kw57_configxt'>
print("specificItemValue=%s" % specificItemValue)
return specificItemValue

# def extractTrFirstTdValue(self, rootDoc, trNumber, i):
def extractTrFirstTdValue(self, rootDoc, trNumber):
    """
    <tr data-pnid="1_-1" id="tr_2">
        <th>
            <div id="1149"><a href="https://car.autohome.com.cn/config/hs_kw57_configxt?pnid=1_-1&trNumber=2" class="hs_kw57_configxt">纯电动</a></div>
        </th>
        <td style="background:#F0F3F8;">
            <div>纯电动</div>
        </td>

    <tr data-pnid="1_-1" id="tr_3">
        <th>
            <div id="0">上市<span class="hs_kw40_configxt" style="background:#F0F3F8;"></span>
        </th>
        <td style="background:#F0F3F8;">
            <div>2019.11</div>
        </td>
        <td>
            <div>2019.11</div>
        </td>
        <td>
            <div></div>
        </td>
        <td>
            <div></div>
        </td>
        <td>
            <div></div>
        </td>
    </tr>

    <tr data-pnid="1_-1" id="tr_20" style="background:#F0F3F8;">
        <th>
            <div id="1255"><a href="https://car.autohome.com.cn/config/hs_kw36_configaJ?pnid=1_-1&trNumber=20" class="hs_kw36_configaJ"></a></div>
        </th>
        <td style="background:#F0F3F8;">
            <div>三<span class="hs_kw7_configaJ"></span>
        </td>
        <td>
            <div>三<span class="hs_kw7_configaJ"></span>
        </td>
        <td>
            <div></div>
        </td>
        <td>
            <div></div>
        </td>
    </tr>
```

```

        <div></div>
    </td>
</tr>
"""

trQuery = "tr[id='tr_%s']" % trNumber
# print("trQuery=%s" % trQuery)
trDoc = rootDoc.find(trQuery)
# print("trDoc=%s" % trDoc)
tdDocGenerator = trDoc.items("td")
# print("tdDocGenerator=%s" % tdDocGenerator)
tdDocList = list(tdDocGenerator)
# print("tdDocList=%s" % tdDocList)
firstTdDoc = tdDocList[0]
# print("firstTdDoc=%s" % firstTdDoc)
firstTdDivDoc = firstTdDoc.find("div")
print("firstTdDivDoc=%s" % firstTdDivDoc)
# if isRespDoc:
#     respItem = firstTdDivDoc
# else:
#     firstItemValue = firstTdDivDoc.text()
#     respItem = firstItemValue
# print("respItem=%s" % respItem)
# return respItem
respItemHtml = firstTdDivDoc.html()
print("respItemHtml=%s" % respItemHtml)
return respItemHtml

# def extractWholeWarranty(self, firstDivDoc):
def extractWholeWarranty(self, firstDivHtml):
    print("in extractWholeWarranty")
    carModelWholeWarranty = ""
    # <div>≡<span class="hs_kw7_configxv"></span>10<span
    # print("firstDivDoc=%s" % firstDivDoc)
    # carModelWholeWarranty = firstDivDoc.text() # ≡10
    # firstDivHtml = firstDivDoc.html()
    print("firstDivHtml=%s" % firstDivHtml)
    # ≡<span class="hs_kw7_configCC"></span>10<span cl
    # carWholeQualityQuarantee = re.sub("[^<>]+(?P<fir
    foundYearDistance = re.search("(?P<warrantyYear>[^
    if foundYearDistance:
        warrantyYear = foundYearDistance.group("warrant
        distanceNumber = foundYearDistance.group("disti
        distanceUnit = foundYearDistance.group("distanc
        carModelWholeWarranty = "%s年或%s万%s" % (warran
    else:
        # special:
        # https://car.autohome.com.cn/config/spec/4670(
        # <div>≡<span class="hs_kw58_configWh"></span>
        # ≡<span class="hs_kw58_configOf"></span>
        foundYearNotLimitDistance = re.search("(?P<war
        print("foundYearNotLimitDistance=%s" % foundYear

```

```

        if foundYearNotLimitDistance:
            warrantyYear = foundYearNotLimitDistance.gi
            print("warrantyYear=%s" % warrantyYear)
            carModelWholeWarranty = "%s年不限公里" % war
            print("carModelWholeWarranty=%s" % carModelWholeWai
            return carModelWholeWarranty

    def getWholeWarranty(self, inputContent, itemIndex):
        # firstDivDoc = self.getItemFirstValue(inputContent)
        # print("firstDivDoc=%s" % firstDivDoc)
        # carModelWholeWarranty = self.extractWholeWarranty(
        firstDivDocHtml = self.getItemFirstValue(inputContent)
        print("firstDivDocHtml=%s" % firstDivDocHtml)
        carModelWholeWarranty = self.extractWholeWarranty(
        return carModelWholeWarranty

    @catch_status_code_error
    def carConfigSpecCallback(self, response):
        print("in carConfigSpecCallback")
        curCarModelDict = response.save
        print("curCarModelDict=%s" % curCarModelDict)
        carModelDict = copy.deepcopy(curCarModelDict)

        configSpecHtml = response.text
        print("configSpecHtml=%s" % configSpecHtml)
        print("")

        # for debug
        return

    # # config json item index - spec table html item :
    # ItemIndexDiff = 2

        # isUseSpecTableHtml = True
        # isUseConfigJson = False
        # valueContent = None
        # energyTypeIdx = 2

        # # Method 1: after run js, extract item value fro
        # """
        # <table class="tbcs" id="tab_0" style="width: 932px;">
        #     <tbody>
        #         <tr>
        #             <th class="cstitle" show="1" pid="tab_0_1">
        #                 <h3><span>基本参数</span></h3>
        #             </th>
        #         </tr>
        #         <tr data-pnid="1_-1" id="tr_0">
        #             <td>
        #                 # tbodyDoc = response.doc("table[id='tab_0'] tbody")
        #                 # print("tbodyDoc=%s" % tbodyDoc)

```

```

# valueContent = tbodyDoc
# isUseSpecTableHtml = True
# isUseConfigJson = False
# energyTypeIdx = 2

# Method 2: not run js, extract item value from config json
# get value from config json
# var config = {"message": "ok", "returncode": "0", "paramTypeItems": [{"id": "P0001", "name": "P0001", "firstValue": "1", "valueItems": [{"id": "V0001", "name": "V0001", "value": "1"}, {"id": "V0002", "name": "V0002", "value": "2"}]}]
# foundConfigJson = re.search("var\s*config\s*=\s*(?P<config>.*)", config)
print("foundConfigJson=%s" % foundConfigJson)
if foundConfigJson:
    configJson = foundConfigJson.group("configJson")
    print("configJson=%s" % configJson)
    # configDict = json.loads(configJson, encoding='utf-8')
    configDict = json.loads(configJson)
    print("configDict=%s" % configDict)

    # if "result" in configDict:
    configResultDict = configDict["result"]
    print("configResultDict=%s" % configResultDict)
    # if "paramtypeitems" in configResultDict:
    paramTypeItemDictList = configResultDict["paramtypeitems"]
    print("paramTypeItemDictList=%s" % paramTypeItemDictList)
    # paramTypeItemNum = len(paramTypeItemDictList)
    # print("paramTypeItemNum=%s" % paramTypeItemNum)
    basicParamDict = paramTypeItemDictList[0]
    print("basicParamDict=%s" % basicParamDict)
    basicItemDictList = basicParamDict["paramitems"]
    print("basicItemDictList=%s" % basicItemDictList)
    # print("type(basicItemDictList)=%s" % type(basicItemDictList))
    # basicItemNum = len(basicItemDictList)
    # print("basicItemNum=%s" % basicItemNum)

    # valueContent = basicItemDictList
    # isUseSpecTableHtml = False
    # isUseConfigJson = True

    # process each basic parameter
    basicItemDictLen = len(basicItemDictList)
    print("basicItemDictLen=%s" % basicItemDictLen)
    for curIdx, eachItemDict in enumerate(basicItemDictList):
        print("[%d] eachItemDict=%s" % (curIdx, eachItemDict))
        curItemId = eachItemDict["id"]
        print("curItemId=%s" % curItemId)
        curItemName = eachItemDict["name"]
        print("curItemName=%s" % curItemName)
        curItemFirstValue = self.extractValueItems(eachItemDict)
        print("curItemFirstValue=%s" % curItemFirstValue)

        curIdNameKeyMapDict = None
        if curItemId != 0:
            curIdNameKeyMapDict = {}
            curIdNameKeyMapDict[curItemId] = curItemName

```

```

curIdNameKeyMapDict = self.findMapping()
else:
    # id = 0
    foundSpan = re.search("<span>", curItem)
    print("foundSpan=%s" % foundSpan)
    isSpecialName = bool(foundSpan)
    print("isSpecialName=%s" % isSpecialName)
    if isSpecialName:
        # id=0 and contain '<span>' special
        foundSuffixHour = re.search("</span>.*<span>(.*)</span>", curItem)
        print("foundSuffixHour=%s" % foundSuffixHour)
        isSpecialSuffixHour = bool(foundSuffixHour)
        print("isSpecialSuffixHour=%s" % isSpecialSuffixHour)
        if isSpecialSuffixHour:
            prevIsQuickCharge = self.isPrevIsQuickCharge
            print("prevIsQuickCharge=%s" % prevIsQuickCharge)
            if prevIsQuickCharge:
                # current is MUST 慢充时间(小时)
                curIdNameKeyMapDict = {
                    "id": 0,
                    # "name": "<span class='...>(.*)</span>",
                    "name": "慢充时间(小时)",
                    "namePattern": "</span>.*<span>(.*)</span>",
                    "key": "carModelSlowChargeTime"
                }

            if not curIdNameKeyMapDict:
                prevIsActualTestEnduranceMode = False
                print("prevIsActualTestEnduranceMode=%s" % prevIsActualTestEnduranceMode)
            if prevIsActualTestEnduranceMode:
                # current is MUST 实测快充时间(分钟)
                curIdNameKeyMapDict = {
                    "id": 0,
                    # "name": "<span class='...>(.*)</span>",
                    "name": "实测快充时间(分钟)",
                    "namePattern": "</span>.*<span>(.*)</span>",
                    "key": "carModelActualFastChargeTime"
                }

            if not curIdNameKeyMapDict:
                prevPrevIsActualTestEnduranceMode = False
                print("prevPrevIsActualTestEnduranceMode=%s" % prevPrevIsActualTestEnduranceMode)
            if prevPrevIsActualTestEnduranceMode:
                # current is MUST 实测慢充时间(分钟)
                curIdNameKeyMapDict = {
                    "id": 0,
                    # "name": "<span class='...>(.*)</span>",
                    "name": "实测慢充时间(分钟)",
                    "namePattern": "</span>.*<span>(.*)</span>",
                    "key": "carModelActualSlowChargeTime"
                }

```

```

        else:
            curIdNameKeyMapDict = self.findMapDict("id_name")
        else:
            curIdNameKeyMapDict = self.findMapDict("name_id")

        print("curIdNameKeyMapDict=%s" % curIdNameKeyMapDict)
        if curIdNameKeyMapDict:
            curItemKey = curIdNameKeyMapDict["key"]
            print("curItemKey=%s" % curItemKey)
            # processedItemValue = self.processSpecialKeyValue(itemValue)
            processedItemValue = self.processSpecialKeyValue(itemValue)
            print("processedItemValue=%s" % processedItemValue)
            carModelDict[curItemKey] = processedItemValue
            print("++ added %s=%s" % (curItemKey, processedItemValue))

        print("after extract all item value: carModelDict=%s" %
              carModelDict)

        self.saveSingleResult(carModelDict)

        # if isUseConfigJson:
        #     energyTypeIdx += ItemIndexDiff

        # if valueContent:
        #     self.processDiffEnergyTypeCar(carModelDict, valueContent)
        # else:
        #     self.saveSingleResult(carModelDict)

    # def processSpecialKeyValue(self, itemKey, itemValue,
    def processSpecialKeyValue(self, itemKey, itemValue, re):
        print("in processSpecialKeyValue")
        print("itemKey=%s, itemValue=%s" % (itemKey, itemValue))
        if itemKey == "carModelWholeWarranty":
            print("process special carModelWholeWarranty")
            # 整车质保
            # 三<span class='hs_kw5_configJS'></span>10<span class='hs_kw5_configFS'></span>
            itemValue = self.extractWholeWarranty(itemValue)
            print("itemValue=%s" % itemValue)
        elif itemKey == "carModelBodyStructure":
            print("process special carModelBodyStructure value")
            # (1) https://www.autohome.com.cn/spec/46292/#>
            # 5门7座<span class='hs_kw3_configFS'></span>
            # -> 5门7座MPV
            #
            # (2) https://www.autohome.com.cn/spec/1002900,
            # <span class='hs_kw21_configqk'></span>
            # -> 皮卡
            foundSpan = re.search("(?P<bodySpan><span>.+?</span>)皮卡", itemValue)
            print("foundSpan=%s" % foundSpan)
            if foundSpan:
                bodySpan = foundSpan.group("bodySpan")
                print("bodySpan=%s" % bodySpan)

```

```

# extract body structure
"""
<div class="filtrate-list filtrate-list-co
    <span class="title">车身结构: </span>
    <label class="lbTxt" for="PL2$!{1 - 1}>
        <input type="checkbox" class="sele
        MPV
    </label>
</div>
"""

# soup = BeautifulSoup(curHtml, "html.parser")
# # print("soup=%s" % soup)
# # bodyStructureSpanSoup = soup.find(text="车身结构:")
# # # print("bodyStructureSpanSoup=%s" % bodyStructureSpanSoup)
# # emptySoup = bodyStructureSpanSoup.next_sib
# # # print("emptySoup=%s" % emptySoup)
# # siblingLabelSoup = emptySoup.next_sib
# # # print("siblingLabelSoup=%s" % siblingLabelSoup)
# # parentDivSoup = bodyStructureSpanSoup.parent
# # print("parentDivSoup=%s" % parentDivSoup)
# # inputSoup = parentDivSoup.find("input", attrs={"name": "carStruct"})
# # print("carStructSoup=%s" % carStructSoup)
# carStructDoc = response.doc("input[name=ca
print("carStructDoc=%s" % carStructDoc)
bodyStructureValue = carStructDoc.attr["va
print("bodyStructureValue=%s" % bodyStructureValue)
itemValue = itemValue.replace(bodySpan, bodyStructureValue)
print("itemValue=%s" % itemValue)

return itemValue

def isPrevItemIsQuickCharge(self, curIdx, itemDictList):
    print("in isPrevItemIsQuickCharge")
    print("curIdx=%s" % curIdx)

    prevIsQuickCharge = False

    if curIdx > 0:
        prevIdx = curIdx - 1
        print("prevIdx=%s" % prevIdx)
        prevItemDict = itemDictList[prevIdx]
        print("prevItemDict=%s" % prevItemDict)
        prevItemId = prevItemDict["id"]
        print("prevItemId=%s" % prevItemId)
        prevItemName = prevItemDict["name"]
        print("prevItemName=%s" % prevItemName)
        """
        "id": 1292,
        # "name": "<span class='hs_kw39_configpl'>
        "name": "快充时间(小时)",

```

```

    """
    QuickChargeItemId = 1292
    if prevItemId == QuickChargeItemId:
        prevIsQuickCharge = True

    print("prevIsQuickCharge=%s" % prevIsQuickCharge)
    return prevIsQuickCharge

def checkIsActualTestEnduranceMileage(self, prevSomeNum, curIdx):
    print("in checkIsActualTestEnduranceMileage")
    print("prevSomeNum=%s, curIdx=%s" % (prevSomeNum, curIdx))

    isActualTestEnduranceMileage = False

    minAllowIdx = prevSomeNum - 1

    if curIdx > minAllowIdx:
        prevSomeIdx = curIdx - prevSomeNum
        print("prevSomeIdx=%s" % prevSomeIdx)
        prevSomeItemDict = itemDictList[prevSomeIdx]
        print("prevSomeItemDict=%s" % prevSomeItemDict)
        prevSomeItemId = prevSomeItemDict["id"]
        print("prevSomeItemId=%s" % prevSomeItemId)
        prevSomeItemName = prevSomeItemDict["name"]
        print("prevSomeItemName=%s" % prevSomeItemName)

        if prevSomeItemId == 0:
            """
                "id": 0,
                # "name": "<span class='hs_kw22_config'>续航里程</span>",
                "name": "实测续航里程(km)",
                "namePattern": "</span>续航里程\\(km\\)$",
                "key": "carModelActualTestEnduranceMileage"
            """

            foundActualTestEnduranceMileage = re.search(
                "foundActualTestEnduranceMileage=%s"
            )
            if foundActualTestEnduranceMileage:
                isActualTestEnduranceMileage = True

    print("isActualTestEnduranceMileage=%s" % isActualTestEnduranceMileage)
    return isActualTestEnduranceMileage

def isPrevItemIsActualTestEnduranceMileage(self, curIdx):
    print("in isPrevItemIsActualTestEnduranceMileage")
    print("curIdx=%s" % curIdx)
    return self.checkIsActualTestEnduranceMileage(1, curIdx)

def isPrevPrevItemIsActualTestEnduranceMileage(self, curIdx):
    print("in isPrevPrevItemIsActualTestEnduranceMileage")
    print("curIdx=%s" % curIdx)
    return self.checkIsActualTestEnduranceMileage(2, curIdx)

```

```
def findMappingDict(self, itemId=0, itemName=""):
    foundMapDict = None

    paramIdNameKeyMapDict = [
        # 汽油车 参数
        # https://car.autohome.com.cn/config/spec/4157
        # https://car.autohome.com.cn/config/spec/1006
        {
            "id": 1149,
            "name": "能源类型",
            "key": "carEnergyType",
        }, {
            "id": 1311,
            "name": "环保标准",
            "key": "carModelEnvStandard",
        }, {
            "id": 0,
            # "name": "上市<span class='hs_kw51_configv
            "name": "上市时间",
            "namePattern": "^上市",
            "key": "carModelReleaseTime",
        }, {
            "id": 1185,
            # "name": "<span class='hs_kw40_configvR'>
            "name": "最大功率(kW)",
            "key": "carModelMaxPower",
        }, {
            "id": 1186,
            # "name": "<span class='hs_kw40_configvR'>
            "name": "最大扭矩(N·m)",
            "key": "carModelMaxTorque",
        }, {
            "id": 1150,
            "name": "发动机",
            "key": "carModelEngine",
        }, {
            "id": 1245,
            "name": "变速箱",
            "key": "carModelGearBox",
        }, {
            "id": 1148,
            "name": "长*宽*高(mm)",
            "key": "carModelSize",
        }, {
            "id": 1147,
            "name": "车身结构",
            "key": "carModelBodyStructure",
        }, {
            "id": 1246,
            "name": "最高车速(km/h)",
            "key": "carModelMaxSpeed"
        }
    ]
    return foundMapDict
```

```

        "key": "carModelMaxSpeed",
    }, {
        "id": 1250,
        "name": "官方0-100km/h加速(s)",
        "key": "carModelOfficialSpeedupTime",
    }, {
        "id": 1252,
        # "name": "<span class='hs_kw26_configvR'>",
        "name": "实测0-100km/h加速(s)",
        "key": "carModelActualTestSpeedupTime",
    }, {
        "id": 1253,
        # "name": "<span class='hs_kw26_configvR'>",
        "name": "实测100-0km/h制动(m)",
        "key": "carModelActualTestBrakeDistance",
    }, {
        "id": 1251,
        # "name": "工信部<span class='hs_kw10_config'>",
        "name": "工信部综合油耗(L/100km)",
        "key": "carModelMiltCompositeFuelConsumption",
    }, {
        "id": 1254,
        # "name": "<span class='hs_kw26_configvR'>",
        "name": "实测油耗(L/100km)",
        "key": "carModelActualFuelConsumption",
    }, {
        "id": 1255,
        # "name": "整车<span class='hs_kw73_configvR'>",
        "name": "整车质保",
        "key": "carModelWholeWarranty",
    },
}

# 电动车 参数
# https://car.autohome.com.cn/config/spec/3989
# https://car.autohome.com.cn/config/spec/4287
{
    "id": 1291,
    "name": "工信部纯电续航里程(km)",
    "key": "carModelMiltEnduranceMileagePureElec"
}, {
    "id": 1292,
    # "name": "<span class='hs_kw39_configpl'>",
    "name": "快充时间(小时)",
    "key": "carModelQuickCharge",
}, {
    # "id": 0,
    # "# "name": "<span class='hs_kw10_configpl'>",
    # "name": "慢充时间(小时)",
    # "namePattern": "</span>\(小时\)$",
    # "key": "carModelSlowCharge",
}
},

```

```

        "id": 0,
        # https://car.autohome.com.cn/config/spec/1
        # {'id': 0, 'name': "<span class='hs_kw39_configpl'>",
        "name": "快充电量百分比",
        "namePattern": "</span>百分比$",
        "key": "carModelQuickChargePercent",
    }, {
        "id": 0,
        "name": "电动机(Ps)",
        "key": "carModelHorsePowerElectric",
    }, {
        "id": 0,
        # "name": "<span class='hs_kw22_configpl'>",
        "name": "实测续航里程(km)",
        "namePattern": "</span>续航里程\(\km\)$",
        "key": "carModelActualTestEnduranceMileage"
    # }, {
        # "id": 0,
        # "# "name": "<span class='hs_kw22_configpl'>",
        # "name": "实测快充时间(小时)",
        # "namePattern": "</span>\(小时\)$",
        # "key": "carModelActualTestQuickCharge",
    # }, {
        # "id": 0,
        # "# "name": "<span class='hs_kw22_configpl'>",
        # "name": "实测慢充时间(小时)",
        # "namePattern": "</span>\(小时\)$",
        # "key": "carModelActualTestSlowCharge",
    }
]

isItemZero = itemId == 0
print("isItemZero=%s" % isItemZero)

foundSpan = re.search("<span>", itemName)
print("foundSpan=%s" % foundSpan)
isSpecialName = bool(foundSpan)
print("isSpecialName=%s" % isSpecialName)
isNotSpecialName = not isSpecialName
print("isNotSpecialName=%s" % isNotSpecialName)

if not isItemZero:
    for eachMapDict in paramIdNameKeyMapDict:
        eachItemId = eachMapDict["id"]
        if eachItemId == itemId:
            foundMapDict = eachMapDict
            break

if not foundMapDict:
    if itemName and isNotSpecialName:
        for eachMapDict in paramIdNameKeyMapDict:

```

```

eachItemName = eachMapDict["name"]
if eachItemName == itemName:
    foundMapDict = eachMapDict
    break

if not foundMapDict:
    if (isItemZero and isSpecialName):
        for eachMapDict in paramIdNameKeyMapDict:
            if "namePattern" in eachMapDict:
                eachItemNamePattern = eachMapDict["namePattern"]
                print("eachItemNamePattern=%s" % eachItemNamePattern)
                foundMatchName = re.search(eachItemNamePattern, itemName)
                print("foundMatchName=%s" % foundMatchName)
                if foundMatchName:
                    foundMapDict = eachMapDict
                    break
print("foundMapDict=%s from id=%s, name=%s" % (foundMapDict, id, name))
return foundMapDict

def processDiffEnergyTypeCar(self, carModelDict, valueContent):
    carEnergyType = self.getItemFirstValue(valueContent)
    # 纯电动 / 汽油 / 插电式混合动力 / 油电混合
    carModelDict["carEnergyType"] = carEnergyType

    if carEnergyType == "汽油":
        # https://car.autohome.com.cn/config/spec/43597
        # https://car.autohome.com.cn/config/spec/41572

        # self.processGasolineCar(valueContent, carModelDict)

        # https://car.autohome.com.cn/config/spec/10064

        gasolineCarKeyIdxMapDict = {
            "carModelEnvStandard": 3,
            "carModelReleaseTime": 4,
            "carModelMaxPower": 5,
            "carModelMaxTorque": 6,
            "carModelEngine": 7,
            "carModelGearBox": 8,
            "carModelSize": 9,
            "carModelBodyStructure": 10,
            "carModelMaxSpeed": 11,
            "carModelOfficialSpeedupTime": 12,
            "carModelActualTestSpeedupTime": 13,
            "carModelActualTestBrakeDistance": 14,
            "carModelMlitCompositeFuelConsumption": 15,
            "carModelActualFuelConsumption": 16,
        }
        wholeWarrantyIdx = 17

    if isUseConfigJson:

```

```

        for eachKey in gasolineCarKeyIdxMapDict.keys():
            gasolineCarKeyIdxMapDict[eachKey] += 1
            wholeWarrantyIdx += ItemIndexDiff

        self.processSingleEnergyTypeCar(gasolineCarKeyIdxMapDict)

    elif carEnergyType == "纯电动":
        # https://car.autohome.com.cn/config/spec/4287

        # self.processPureElectricCar(valueContent, carModelDict)

        pureElectricCarKeyIdxMapDict = {
            "carModelReleaseTime": 3,
            "carModelMileEnduranceMileagePureElectric": 4,
            "carModelQuickCharge": 5,
            "carModelSlowCharge": 6,
            "carModelQuickChargePercent": 7,
            "carModelMaxPower": 8,
            "carModelMaxTorque": 9,
            "carModelHorsePowerElectric": 10,
            "carModelSize": 11,
            "carModelBodyStructure": 12,
            "carModelMaxSpeed": 13,
            "carModelOfficialSpeedupTime": 14,
            "carModelActualTestSpeedupTime": 15,
            "carModelActualTestBrakeDistance": 16,
            "carModelActualTestEnduranceMileage": 17,
            "carModelActualTestQuickCharge": 18,
            "carModelActualTestSlowCharge": 19,
        }
        wholeWarrantyIdx = 20

        if isUseConfigJson:
            for eachKey in pureElectricCarKeyIdxMapDict.keys():
                pureElectricCarKeyIdxMapDict[eachKey] -= 1
                wholeWarrantyIdx += ItemIndexDiff

        self.processSingleEnergyTypeCar(pureElectricCarKeyIdxMapDict)

    elif carEnergyType == "插电式混合动力":
        # https://car.autohome.com.cn/config/series/446

        # self.processPhevCar(valueContent, carModelDict)

        phevCarKeyIdxMapDict = {
            "carModelEnvStandard": 3,
            "carModelReleaseTime": 4,
            "carModelMileEnduranceMileagePureElectric": 5,
            "carModelQuickCharge": 6,
            "carModelSlowCharge": 7,
            "carModelQuickChargePercent": 8,
        }

```

```

        "carModelMaxPower": 9,
        "carModelMaxTorque": 10,
        "carModelEngine": 11,
        "carModelHorsePowerElectric": 12,
        "carModelGearBox": 13,
        "carModelSize": 14,
        "carModelBodyStructure": 15,
        "carModelMaxSpeed": 16,
        "carModelOfficialSpeedupTime": 17,
        "carModelActualTestSpeedupTime": 18,
        "carModelActualTestBrakeDistance": 19,
        "carModelActualTestEnduranceMileage": 20,
        "carModelActualTestQuickCharge": 21,
        "carModelActualTestSlowCharge": 22,
        "carModelMiltCompositeFuelConsumption": 23,
        "carModelActualFuelConsumption": 24,
    }
wholeWarrantyIdx = 25

if isUseConfigJson:
    for eachKey in phevCarKeyIdxMapDict.keys():
        phevCarKeyIdxMapDict[eachKey] += ItemIndexDiff
    wholeWarrantyIdx += ItemIndexDiff

self.processSingleEnergyTypeCar(phevCarKeyIdxMapDict)

elif carEnergyType == "油电混合":
    # https://car.automobile.com.cn/config/spec/3550

    # self.processHevCar(valueContent, carModelDict)

    hevCarKeyIdxMapDict = {
        "carModelEnvStandard": 3,
        "carModelReleaseTime": 4,
        "carModelMaxPower": 5,
        "carModelMaxTorque": 6,
        "carModelEngine": 7,
        "carModelHorsePowerElectric": 8,
        "carModelGearBox": 9,
        "carModelSize": 10,
        "carModelBodyStructure": 11,
        "carModelMaxSpeed": 12,
        "carModelOfficialSpeedupTime": 13,
        "carModelActualTestSpeedupTime": 14,
        "carModelActualTestBrakeDistance": 15,
        "carModelMiltCompositeFuelConsumption": 16,
        "carModelActualFuelConsumption": 17,
    }
wholeWarrantyIdx = 18

if isUseConfigJson:

```

```

        for eachKey in hevCarKeyIdxMapDict.keys():
            hevCarKeyIdxMapDict[eachKey] += ItemIndexDiff
            wholeWarrantyIdx += ItemIndexDiff

        self.processSingleEnergyTypeCar(hevCarKeyIdxMapDict)
    else:
        errMsg = "TODO: add support %s!" % carEnergyType
        raise Exception(errMsg)

    def processSingleEnergyTypeCar(self, keyIdxMapDict, valueContent):
        keyList = keyIdxMapDict.keys()
        keyListLen = len(keyList)
        print("keyListLen=%s" % keyListLen)
        for eachItemKey in keyList:
            print("eachItemKey=%s" % eachItemKey)
            eachItemIndex = keyIdxMapDict[eachItemKey]
            print("eachItemIndex=%s" % eachItemIndex)
            eachItemValue = self.getItemFirstValue(valueContent)
            print("eachItemValue=%s" % eachItemValue)
            carModelDict[eachItemKey] = eachItemValue

        # 整车质保
        carModelWholeWarranty = self.getWholeWarranty(valueContent)
        print("carModelWholeWarranty=%s" % carModelWholeWarranty)
        carModelDict["carModelWholeWarranty"] = carModelWholeWarranty

        self.saveSingleResult(carModelDict)

# def processGasolineCar(self, valueContent, carModelID):
#     # 汽油

#     # https://car.autohome.com.cn/config/spec/43593.html
#     # https://car.autohome.com.cn/config/spec/41572.html

#     # 环保标准
#     carModelEnvStandard = self.getItemFirstValue(valueContent)
#     carModelDict["carModelEnvStandard"] = carModelEnvStandard

#     # 上市时间
#     carModelReleaseTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelReleaseTime"] = carModelReleaseTime

#     # 最大功率(kW)
#     carModelMaxPower = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxPower"] = carModelMaxPower

#     # 最大扭矩(N·m)
#     carModelMaxTorque = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxTorque"] = carModelMaxTorque

#     # 发动机

```

```

# carModelEngine = self.getItemFirstValue(valueContent)
# carModelDict["carModelEngine"] = carModelEngine

# # 变速箱
# carModelGearBox = self.getItemFirstValue(valueContent)
# carModelDict["carModelGearBox"] = carModelGearBox

# # 长*宽*高(mm)
# carModelSize = self.getItemFirstValue(valueContent)
# carModelDict["carModelSize"] = carModelSize

# # 车身结构
# carModelBodyStructure = self.getItemFirstValue(valueContent)
# carModelDict["carModelBodyStructure"] = carModelBodyStructure

# # 最高车速(km/h)
# carModelMaxSpeed = self.getItemFirstValue(valueContent)
# carModelDict["carModelMaxSpeed"] = carModelMaxSpeed

# # 官方0-100km/h加速(s)
# carModelOfficialSpeedupTime = self.getItemFirstValue(valueContent)
# carModelDict["carModelOfficialSpeedupTime"] = carModelOfficialSpeedupTime

# # 实测0-100km/h加速(s)
# carModelActualTestSpeedupTime = self.getItemFirstValue(valueContent)
# carModelDict["carModelActualTestSpeedupTime"] = carModelActualTestSpeedupTime

# # 实测100-0km/h制动(m)
# carModelActualTestBrakeDistance = self.getItemFirstValue(valueContent)
# carModelDict["carModelActualTestBrakeDistance"] = carModelActualTestBrakeDistance

# # 工信部综合油耗(L/100km)
# carModelMiitCompositeFuelConsumption = self.getItemFirstValue(valueContent)
# carModelDict["carModelMiitCompositeFuelConsumption"] = carModelMiitCompositeFuelConsumption

# # 实测油耗(L/100km)
# carModelActualFuelConsumption = self.getItemFirstValue(valueContent)
# carModelDict["carModelActualFuelConsumption"] = carModelActualFuelConsumption

# self.saveSingleResult(carModelDict)

# def processPureElectricCar(self, valueContent, carModelDict):
#     # 纯电动

#     # https://car.autohome.com.cn/config/spec/42875.html
#     # 上市时间
#     carModelReleaseTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelReleaseTime"] = carModelReleaseTime

#     # 工信部纯电续航里程(km)

```

```

# carModelMileagePureElectric = self.getItemFirstValue(valueContent)
# carModelDict["carModelMileagePureElectric"] = carModelMileagePureElectric

# 快充时间(小时)
# carModelQuickCharge = self.getItemFirstValue(valueContent)
# carModelDict["carModelQuickCharge"] = carModelQuickCharge

# 慢充时间(小时)
# carModelSlowCharge = self.getItemFirstValue(valueContent)
# carModelDict["carModelSlowCharge"] = carModelSlowCharge

# 快充电量百分比
# carModelQuickChargePercent = self.getItemFirstValue(valueContent)
# carModelDict["carModelQuickChargePercent"] = carModelQuickChargePercent

# 最大功率(kW)
# carModelMaxPower = self.getItemFirstValue(valueContent)
# carModelDict["carModelMaxPower"] = carModelMaxPower

# 最大扭矩(N·m)
# carModelMaxTorque = self.getItemFirstValue(valueContent)
# carModelDict["carModelMaxTorque"] = carModelMaxTorque

# 电动机(Ps)
# carModelHorsePowerElectric = self.getItemFirstValue(valueContent)
# carModelDict["carModelHorsePowerElectric"] = carModelHorsePowerElectric

# 长*宽*高(mm)
# carModelSize = self.getItemFirstValue(valueContent)
# carModelDict["carModelSize"] = carModelSize

# 车身结构
# carModelBodyStructure = self.getItemFirstValue(valueContent)
# carModelDict["carModelBodyStructure"] = carModelBodyStructure

# 最高车速(km/h)
# carModelMaxSpeed = self.getItemFirstValue(valueContent)
# carModelDict["carModelMaxSpeed"] = carModelMaxSpeed

# 官方0-100km/h加速(s)
# carModelOfficialSpeedupTime = self.getItemFirstValue(valueContent)
# carModelDict["carModelOfficialSpeedupTime"] = carModelOfficialSpeedupTime

# 实测0-100km/h加速(s)
# carModelActualTestSpeedupTime = self.getItemFirstValue(valueContent)
# carModelDict["carModelActualTestSpeedupTime"] = carModelActualTestSpeedupTime

# 实测100-0km/h制动(m)
# carModelActualTestBrakeDistance = self.getItemFirstValue(valueContent)
# carModelDict["carModelActualTestBrakeDistance"] = carModelActualTestBrakeDistance

```

```

#      # 实测续航里程(km)
#      carModelActualTestEnduranceMileage = self.getItemFirstValue(valueContent)
#      carModelDict["carModelActualTestEnduranceMileage"] = carModelActualTestEnduranceMileage

#      # 实测快充时间(小时)
#      carModelActualTestQuickCharge = self.getItemFirstValue(valueContent)
#      carModelDict["carModelActualTestQuickCharge"] = carModelActualTestQuickCharge

#      # 实测慢充时间(小时)
#      carModelActualTestSlowCharge = self.getItemFirstValue(valueContent)
#      carModelDict["carModelActualTestSlowCharge"] = carModelActualTestSlowCharge

#      # 整车质保
#      carModelWholeWarranty = self.getWholeWarranty(valueContent)
#      carModelDict["carModelWholeWarranty"] = carModelWholeWarranty

#      self.saveSingleResult(carModelDict)

# def processPhevCar(self, valueContent, carModelDict):
#     # 插电式混合动力 = PHEV = Plug-in Hybrid Electric vehicle
#     # https://car.autohome.com.cn/config/series/4460
#
#     # 环保标准
#     carModelEnvStandard = self.getItemFirstValue(valueContent)
#     carModelDict["carModelEnvStandard"] = carModelEnvStandard

#     # 上市时间
#     carModelReleaseTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelReleaseTime"] = carModelReleaseTime

#     # 工信部纯电续航里程(km)
#     carModelMiitEnduranceMileagePureElectric = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMiitEnduranceMileagePureElectric"] = carModelMiitEnduranceMileagePureElectric

#     # 快充时间(小时)
#     carModelQuickCharge = self.getItemFirstValue(valueContent)
#     carModelDict["carModelQuickCharge"] = carModelQuickCharge

#     # 慢充时间(小时)
#     carModelSlowCharge = self.getItemFirstValue(valueContent)
#     carModelDict["carModelSlowCharge"] = carModelSlowCharge

#     # 快充电量百分比
#     carModelQuickChargePercent = self.getItemFirstValue(valueContent)
#     carModelDict["carModelQuickChargePercent"] = carModelQuickChargePercent

#     # 最大功率(kW)
#     carModelMaxPower = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxPower"] = carModelMaxPower

```

```

#       # 最大扭矩(N·m)
#       carModelMaxTorque = self.getItemFirstValue(valueContent)
#       carModelDict["carModelMaxTorque"] = carModelMaxTorque

#       # 发动机
#       carModelEngine = self.getItemFirstValue(valueContent)
#       carModelDict["carModelEngine"] = carModelEngine

#       # 电动机(Ps)
#       carModelHorsePowerElectric = self.getItemFirstValue(valueContent)
#       carModelDict["carModelHorsePowerElectric"] = carModelHorsePowerElectric

#       # 变速箱
#       carModelGearBox = self.getItemFirstValue(valueContent)
#       carModelDict["carModelGearBox"] = carModelGearBox

#       # 长*宽*高(mm)
#       carModelSize = self.getItemFirstValue(valueContent)
#       carModelDict["carModelSize"] = carModelSize

#       # 车身结构
#       carModelBodyStructure = self.getItemFirstValue(valueContent)
#       carModelDict["carModelBodyStructure"] = carModelBodyStructure

#       # 最高车速(km/h)
#       carModelMaxSpeed = self.getItemFirstValue(valueContent)
#       carModelDict["carModelMaxSpeed"] = carModelMaxSpeed

#       # 官方0-100km/h加速(s)
#       carModelOfficialSpeedupTime = self.getItemFirstValue(valueContent)
#       carModelDict["carModelOfficialSpeedupTime"] = carModelOfficialSpeedupTime

#       # 实测0-100km/h加速(s)
#       carModelActualTestSpeedupTime = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualTestSpeedupTime"] = carModelActualTestSpeedupTime

#       # 实测100-0km/h制动(m)
#       carModelActualTestBrakeDistance = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualTestBrakeDistance"] = carModelActualTestBrakeDistance

#       # 实测续航里程(km)
#       carModelActualTestEnduranceMileage = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualTestEnduranceMileage"] = carModelActualTestEnduranceMileage

#       # 实测快充时间(小时)
#       carModelActualTestQuickCharge = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualTestQuickCharge"] = carModelActualTestQuickCharge

#       # 实测慢充时间(小时)
#       carModelActualTestSlowCharge = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualTestSlowCharge"] = carModelActualTestSlowCharge

```

```

#       # 工信部综合油耗(L/100km)
#       carModelMiiCompositeFuelConsumption = self.getItemFirstValue(valueContent)
#       carModelDict["carModelMiiCompositeFuelConsumption"] = carModelMiiCompositeFuelConsumption

#       # 实测油耗(L/100km)
#       carModelActualFuelConsumption = self.getItemFirstValue(valueContent)
#       carModelDict["carModelActualFuelConsumption"] = carModelActualFuelConsumption

#       # 整车质保
#       carModelWholeWarranty = self.getWholeWarranty(valueContent)
#       carModelDict["carModelWholeWarranty"] = carModelWholeWarranty

#       self.saveSingleResult(carModelDict)

# def processHvCar(self, valueContent, carModelDict):
#     # 混合电动汽车=HEV=Hybrid Electric Vehicle

#     # https://car.autohome.com.cn/config/spec/35507.html

#     # 环保标准
#     carModelEnvStandard = self.getItemFirstValue(valueContent)
#     carModelDict["carModelEnvStandard"] = carModelEnvStandard

#     # 上市时间
#     carModelReleaseTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelReleaseTime"] = carModelReleaseTime

#     # 最大功率(kW)
#     carModelMaxPower = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxPower"] = carModelMaxPower

#     # 最大扭矩(N·m)
#     carModelMaxTorque = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxTorque"] = carModelMaxTorque

#     # 发动机
#     carModelEngine = self.getItemFirstValue(valueContent)
#     carModelDict["carModelEngine"] = carModelEngine

#     # 电动机(Ps)
#     carModelHorsePowerElectric = self.getItemFirstValue(valueContent)
#     carModelDict["carModelHorsePowerElectric"] = carModelHorsePowerElectric

#     # 变速箱
#     carModelGearBox = self.getItemFirstValue(valueContent)
#     carModelDict["carModelGearBox"] = carModelGearBox

#     # 长*宽*高(mm)
#     carModelSize = self.getItemFirstValue(valueContent)
#     carModelDict["carModelSize"] = carModelSize

```

```

#     # 车身结构
#     carModelBodyStructure = self.getItemFirstValue(valueContent)
#     carModelDict["carModelBodyStructure"] = carModelBodyStructure

#     # 最高车速(km/h)
#     carModelMaxSpeed = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMaxSpeed"] = carModelMaxSpeed

#     # 官方0-100km/h加速(s)
#     carModelOfficialSpeedupTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelOfficialSpeedupTime"] = carModelOfficialSpeedupTime

#     # 实测0-100km/h加速(s)
#     carModelActualTestSpeedupTime = self.getItemFirstValue(valueContent)
#     carModelDict["carModelActualTestSpeedupTime"] = carModelActualTestSpeedupTime

#     # 实测100-0km/h制动(m)
#     carModelActualTestBrakeDistance = self.getItemFirstValue(valueContent)
#     carModelDict["carModelActualTestBrakeDistance"] = carModelActualTestBrakeDistance

#     # 工信部综合油耗(L/100km)
#     carModelMiiitCompositeFuelConsumption = self.getItemFirstValue(valueContent)
#     carModelDict["carModelMiiitCompositeFuelConsumption"] = carModelMiiitCompositeFuelConsumption

#     # 实测油耗(L/100km)
#     carModelActualFuelConsumption = self.getItemFirstValue(valueContent)
#     carModelDict["carModelActualFuelConsumption"] = carModelActualFuelConsumption

#     # 整车质保
#     carModelWholeWarranty = self.getWholeWarranty(valueContent)
#     carModelDict["carModelWholeWarranty"] = carModelWholeWarranty

#     self.saveSingleResult(carModelDict)

def processSingleResult(self, curCarModelDict):
    print("in processSingleResult")
    self.processCarSpecConfig(curCarModelDict)

    # self.saveSingleResult(curCarModelDict)

def saveSingleResult(self, curCarModelDict):
    carModelDict = copy.deepcopy(curCarModelDict)
    # print("before filter: carModelDict=%s" % carModelDict)
    # process
    for curKey, curValue in carModelDict.items():
        print("curKey=%s, curValue=%s" % (curKey, curValue))
        if curValue is None:
            carModelDict[curKey] = ""
        elif curValue == "-":
            # '-' -> ''

```

```
carModelDict[curKey] = ""

elif "暂无" in curValue:
    # '暂无 暂无 暂无', '暂无' -> ''
    carModelDict[curKey] = ""

# print("after filter: carModelDict=%s" % carModelDict)
"""

插电式混合动力
https://www.autohome.com.cn/spec/37077/
{
    "carBrandId": "33",
    "carBrandLogoUrl": "https://car2.autohome.com.cn/pic/2019/11/20191115145521_15045.jpg",
    "carBrandName": "奥迪",
    "carEnergyType": "插电式混合动力",
    "carMerchantName": "奥迪(进口)",
    "carMerchantUrl": "https://car.autohome.com.cn/pic/2019/11/20191115145521_15045.jpg",
    "carModelActualFuelConsumption": "",
    "carModelActualTestBrakeDistance": "",
    "carModelActualTestEnduranceMileage": "",
    "carModelActualTestQuickCharge": "",
    "carModelActualTestSlowCharge": "",
    "carModelActualTestSpeedupTime": "",
    "carModelBodyStructure": "5门5座SUV",
    "carModelDataSift2": "国V",
    "carModelDataSift3": "2.0T",
    "carModelDataSift4": "8挡手自一体",
    "carModelDriveType": "前置四驱",
    "carModelEngine": "2.0T 252马力 L4",
    "carModelEnvStandard": "国V",
    "carModelGearBox": "8挡手自一体",
    "carModelGroupName": "2.0升 涡轮增压 252马力 前置四驱",
    "carModelHorsePowerElectric": "128",
    "carModelMaxPower": "270",
    "carModelMaxSpeed": "228",
    "carModelMaxTorque": "700",
    "carModelMiltCompositeFuelConsumption": "",
    "carModelMiltEnduranceMileagePureElectric": "",
    "carModelMsrp": "79.08万",
    "carModelName": "2019款 55 e-tron",
    "carModelOfficialSpeedupTime": "5.9",
    "carModelQuickCharge": "2.5",
    "carModelQuickChargePercent": "",
    "carModelReleaseTime": "2018.11",
    "carModelSize": "5071*1968*1716",
    "carModelSlowCharge": "10.8",
    "carModelSpecId": "37077",
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/37077/",
    "carModelWholeWarranty": "三年或10万公里",
    "carModelYear": "2019款",
    "carSeriesId": "4460",
    "carSeriesLevelId": "19",
    "carSeriesLevelName": "中大型SUV",
```

```
"carSeriesMainImgUrl": "https://car3.autohome.com.cn/spec/43593/carseriesmainimgurl.jpg",
"carSeriesMaxPrice": "79.08万",
"carSeriesMinPrice": "79.08万",
"carSeriesMsrp": "79.08-79.08万",
"carSeriesMsrpUrl": "https://www.autohome.com.cn/spec/43593/carseriesmsrpurl.html",
"carSeriesName": "奥迪Q7新能源",
"carSeriesUrl": "https://www.autohome.com.cn/spec/43593/carseriesurl.html"
}
```

汽油

```
https://www.autohome.com.cn/spec/43593/
{
    "carBrandId": "33",
    "carBrandLogoUrl": "https://car2.autohome.com.cn/spec/43593/carbrandlogourl.jpg",
    "carBrandName": "奥迪",
    "carEnergyType": "汽油",
    "carMerchantName": "一汽-大众奥迪",
    "carMerchantUrl": "https://car.autohome.com.cn/spec/43593/carmerchanturl.html",
    "carModelActualFuelConsumption": "",
    "carModelActualTestBrakeDistance": "",
    "carModelActualTestSpeedupTime": "",
    "carModelBodyStructure": "5门5座两厢车",
    "carModelDataSift2": "国VI",
    "carModelDataSift3": "1.4T",
    "carModelDataSift4": "7挡双离合",
    "carModelDriveType": "前置前驱",
    "carModelEngine": "1.4T 150马力 L4",
    "carModelEnvStandard": "国VI",
    "carModelGearBox": "7挡双离合",
    "carModelGroupName": "1.4升 涡轮增压 150马力",
    "carModelMaxPower": "110",
    "carModelMaxSpeed": "200",
    "carModelMaxTorque": "250",
    "carModelMiltCompositeFuelConsumption": "百公里综合油耗6.5L",
    "carModelMsrp": "19.32万",
    "carModelName": "2020款 改款 Sportback 版",
    "carModelOfficialSpeedupTime": "8.4",
    "carModelReleaseTime": "2020.04",
    "carModelSize": "4312*1785*1426",
    "carModelSpecId": "43593",
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/43593/carmodelspecurl.html",
    "carModelWholeWarranty": "三年或10万公里",
    "carModelYear": "2020款",
    "carSeriesId": "3170",
    "carSeriesLevelId": "3",
    "carSeriesLevelName": "紧凑型车",
    "carSeriesMainImgUrl": "https://car3.autohome.com.cn/spec/43593/carseriesmainimgurl.jpg",
    "carSeriesMaxPrice": "23.46万",
    "carSeriesMinPrice": "19.32万",
    "carSeriesMsrp": "19.32-23.46万",
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/spec/43593/carseriesmsrpurl.html"
}
```

```
        "carSeriesName": "奥迪A3",
        "carSeriesUrl": "https://www.autohome.com.cn/config/spec/16147/"
    }

https://car.autohome.com.cn/config/spec/16147.{
{
    "carBrandId": "91",
    "carBrandLogoUrl": "https://car3.autoimg.cn/v3/car/carbrand/logo/91.png",
    "carBrandName": "红旗",
    "carEnergyType": "汽油",
    "carMerchantName": "一汽红旗",
    "carMerchantUrl": "https://car.autohome.com.cn/config/spec/16147",
    "carModelActualFuelConsumption": "",
    "carModelActualTestBrakeDistance": "",
    "carModelActualTestSpeedupTime": "",
    "carModelBodyStructure": "4门5座三厢车",
    "carModelDriveType": "后驱",
    "carModelEngine": "2.0T 204马力 L4",
    "carModelEnvStandard": "国IV(国V)",
    "carModelGearBox": "6挡手自一体",
    "carModelGroupName": "2.0升 涡轮增压 204马力 前驱",
    "carModelMaxPower": "150",
    "carModelMaxSpeed": "",
    "carModelMaxTorque": "260",
    "carModelMlitCompositeFuelConsumption": "百公里综合油耗6.5升",
    "carModelMsrp": "37.98万",
    "carmodelName": "2013款 2.0T 尊贵型",
    "carModelOfficialSpeedupTime": "",
    "carModelReleaseTime": "2013.05",
    "carModelSize": "5095*1875*1485",
    "carModelSpecId": "16147",
    "carModelSpecUrl": "https://www.autohome.com.cn/config/spec/16147",
    "carModelWholeWarranty": "四年或10万公里",
    "carModelYear": "2013款",
    "carSeriesId": "2771",
    "carSeriesLevelId": "5",
    "carSeriesLevelName": "中大型车",
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/v3/car/carseries/mainimg/2771.png",
    "carSeriesMaxPrice": "31.78万",
    "carSeriesMinPrice": "25.28万",
    "carSeriesMsrp": "25.28-31.78万",
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/config/spec/16147",
    "carSeriesName": "红旗H7",
    "carSeriesUrl": "https://www.autohome.com.cn/config/spec/16147"
}

https://www.autohome.com.cn/spec/46144.{
{
    "carBrandId": "36",
    "carBrandLogoUrl": "https://car3.autoimg.cn/v3/car/carbrand/logo/36.png",
    "carBrandName": "奔驰",
    "carEnergyType": "汽油",
    "carMerchantName": "奔驰",
    "carMerchantUrl": "https://car.autohome.com.cn/config/spec/46144",
    "carModelActualFuelConsumption": "百公里综合油耗10.5升",
    "carModelActualTestBrakeDistance": "100米",
    "carModelActualTestSpeedupTime": "百公里加速时间8.5秒",
    "carModelBodyStructure": "5门5座旅行车",
    "carModelDriveType": "前置前驱",
    "carModelEngine": "2.0T 245马力 L4",
    "carModelEnvStandard": "国VI",
    "carModelGearBox": "7挡手自一体",
    "carModelGroupName": "2.0升 涡轮增压 245马力 前置前驱",
    "carModelMaxPower": "180",
    "carModelMaxSpeed": "200",
    "carModelMaxTorque": "370",
    "carModelMlitCompositeFuelConsumption": "百公里综合油耗6.5升",
    "carModelMsrp": "46.98万",
    "carmodelName": "2018款 2.0T 豪华运动版",
    "carModelOfficialSpeedupTime": "百公里加速时间8.5秒",
    "carModelReleaseTime": "2018.08",
    "carModelSize": "4688*1830*1439",
    "carModelSpecId": "46144",
    "carModelSpecUrl": "https://www.autohome.com.cn/config/spec/46144",
    "carModelWholeWarranty": "四年或10万公里",
    "carModelYear": "2018款",
    "carSeriesId": "2772",
    "carSeriesLevelId": "6",
    "carSeriesLevelName": "中大型车",
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/v3/car/carseries/mainimg/2772.png",
    "carSeriesMaxPrice": "46.98万",
    "carSeriesMinPrice": "46.98万",
    "carSeriesMsrp": "46.98万",
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/config/spec/46144",
    "carSeriesName": "奔驰C级",
    "carSeriesUrl": "https://www.autohome.com.cn/config/spec/46144"
}
```

```
"carEnergyType": "汽油",
"carMerchantName": "北京奔驰",
"carMerchantUrl": "https://car.autohome.com.cn",
"carModelActualFuelConsumption": "",
"carModelActualTestBrakeDistance": "39.01",
"carModelActualTestEnduranceMileage": "",
"carModelActualTestQuickCharge": "",
"carModelActualTestSlowCharge": "",
"carModelActualTestSpeedupTime": "9.01",
"carModelBodyStructure": "5门5座SUV",
"carModelDataSift2": "国VI",
"carModelDataSift3": "1.3T",
"carModelDataSift4": "7挡双离合",
"carModelDriveType": "前置前驱",
"carModelEngine": "1.3T 163马力 L4",
"carModelEnvStandard": "国VI",
"carModelGearBox": "7挡双离合",
"carModelGroupName": "1.3升 涡轮增压 163马力",
"carModelHorsePowerElectric": "",
"carModelMaxPower": "120",
"carModelMaxSpeed": "207",
"carModelMaxTorque": "250",
"carModelMiitCompositeFuelConsumption": "6.5L/100km",
"carModelMiitEnduranceMileagePureElectric": "250km",
"carModelMsrp": "30.38万",
"carModelName": "2020款 GLA 200",
"carModelOfficialSpeedupTime": "9",
"carModelQuickCharge": "",
"carModelQuickChargePercent": "",
"carModelReleaseTime": "2020.07",
"carModelSize": "4417*1834*1610",
"carModelSlowCharge": "",
"carModelSpecId": "46144",
"carModelSpecUrl": "https://www.autohome.com.cn/spec/42875/",
"carModelWholeWarranty": "三年不限公里",
"carModelYear": "2020款",
"carSeriesId": "3248",
"carSeriesLevelId": "17",
"carSeriesLevelName": "紧凑型SUV",
"carSeriesMainImgUrl": "https://car3.autohome.com.cn",
"carSeriesMaxPrice": "30.38万",
"carSeriesMinPrice": "30.38万",
"carSeriesMsrp": "30.38-30.38万",
"carSeriesMsrpUrl": "https://www.autohome.com.cn/spec/42875/",
"carSeriesName": "奔驰GLA",
"carSeriesUrl": "https://www.autohome.com.cn/spec/42875/"}}
```

纯电动

<https://www.autohome.com.cn/spec/42875/>

{

```
"carBrandId": "33",
"carBrandLogoUrl": "https://car2.autohome.com.cn/config/spec/35507.1",
"carBrandName": "奥迪",
"carEnergyType": "纯电动",
"carMerchantName": "一汽-大众奥迪",
"carMerchantUrl": "https://car.autohome.com.cn/config/spec/35507.1",
"carModelActualTestBrakeDistance": "",
"carModelActualTestEnduranceMileage": "",
"carModelActualTestQuickCharge": "",
"carModelActualTestSlowCharge": "",
"carModelActualTestSpeedupTime": "",
"carModelBodyStructure": "5门5座SUV",
"carModelDataSift2": "100KW",
"carModelDataSift3": "265公里",
"carModelDataSift4": "单速",
"carModelDriveType": "前置前驱",
"carModelGearBox": "电动车单速变速箱",
"carModelGroupName": "电动 136马力",
"carModelHorsePowerElectric": "136",
"carModelMaxPower": "100",
"carModelMaxSpeed": "150",
"carModelMaxTorque": "290",
"carModelMiitEnduranceMileagePureElectric": "265",
"carModelMsrp": "22.68万",
"carModelName": "2019款 Q2L e-tron 纯电动",
"carModelOfficialSpeedupTime": "",
"carModelQuickCharge": "0.6",
"carModelQuickChargePercent": "80",
"carModelReleaseTime": "2019.11",
"carModelSize": "4237*1785*1548",
"carModelSlowCharge": "17",
"carModelSpecId": "42875",
"carModelSpecUrl": "https://www.autohome.com.cn/config/spec/35507.1",
"carModelWholeWarranty": "三年或10万公里",
"carModelYear": "2019款",
"carSeriesId": "5240",
"carSeriesLevelId": "16",
"carSeriesLevelName": "小型SUV",
"carSeriesMainImgUrl": "https://car2.autohome.com.cn/config/spec/35507.1",
"carSeriesMaxPrice": "23.73万",
"carSeriesMinPrice": "22.68万",
"carSeriesMsrp": "22.68-23.73万",
"carSeriesMsrpUrl": "https://www.autohome.com.cn/config/spec/35507.1",
"carSeriesName": "奥迪Q2L e-tron",
"carSeriesUrl": "https://www.autohome.com.cn/config/spec/35507.1"
}
```

油电混合

```
https://car.autohome.com.cn/config/spec/35507.1
{
    "carBrandId": "52",
```

```
"carBrandLogoUrl": "https://car2.autohome.com.cn/Content/Images/Brands/lexus.png",
"carBrandName": "雷克萨斯",
"carEnergyType": "油电混合",
"carMerchantName": "雷克萨斯",
"carMerchantUrl": "https://car.autohome.com.cn/Content/Brands/lexus.html",
"carModelActualFuelConsumption": "",
"carModelActualTestBrakeDistance": "",
"carModelActualTestEnduranceMileage": "",
"carModelActualTestQuickCharge": "",
"carModelActualTestSlowCharge": "",
"carModelActualTestSpeedupTime": "",
"carModelBodyStructure": "5门5座两厢车",
"carModelDataSift2": "",
"carModelDataSift3": "",
"carModelDataSift4": "",
"carModelDriveType": "前驱",
"carModelEngine": "1.8L 99马力 L4",
"carModelEnvStandard": "国IV(国V)",
"carModelGearBox": "E-CVT无级变速",
"carModelGroupName": "1.8升 99马力",
"carModelHorsePowerElectric": "82",
"carModelMaxPower": "100",
"carModelMaxSpeed": "",
"carModelMaxTorque": "",
"carModelMiiCompositeFuelConsumption": null,
"carModelMiiEnduranceMileagePureElectric": null,
"carModelMsrp": "26.70万",
"carmodelName": "2018款 CT200h 多彩生活特装版",
"carModelOfficialSpeedupTime": "",
"carModelQuickCharge": "",
"carModelQuickChargePercent": "",
"carModelReleaseTime": "2018.08",
"carModelSize": "4360*1765*1455",
"carModelSlowCharge": "",
"carModelSpecId": "35507",
"carModelSpecUrl": "https://www.autohome.com.cn/spec/35507.html",
"carModelWholeWarranty": "六年或15万公里",
"carModelYear": "2018款",
"carSeriesId": "2063",
"carSeriesLevelId": "3",
"carSeriesLevelName": "紧凑型车",
"carSeriesMainImgUrl": "https://car2.autohome.com.cn/Content/Images/Brands/lexus.png",
"carSeriesMaxPrice": "28.20万",
"carSeriesMinPrice": "21.50万",
"carSeriesMsrp": "21.50-28.20万",
"carSeriesMsrpUrl": "https://www.autohome.com.cn/spec/35507.html",
"carSeriesName": "雷克萨斯CT",
"carSeriesUrl": "https://www.autohome.com.cn/spec/35507.html"
}
```

....

```
carAllKeyList = [
    "carBrandName",
    "carBrandId",
    "carBrandLogoUrl",
    "carMerchantName",
    "carMerchantUrl",
    "carSeriesId",
    "carSeriesName",
    "carSeriesUrl",
    "carSeriesMsrp",
    "carSeriesMsrpUrl",
    "carSeriesMainImgUrl",
    "carSeriesMinPrice",
    "carSeriesMaxPrice",
    "carSeriesLevelName",
    "carSeriesLevelId",
    "carModelName",
    "carModelSpecUrl",
    "carModelSpecId",
    "carModelMsrp",
    "carModelYear",
    "carModelGearBox",
    "carModelDriveType",
    "carModelBodyStructure",
    "carModelEngine",
    "carModelGroupName",
    "carModelDataSift2",
    "carModelDataSift3",
    "carModelDataSift4",
    "carEnergyType",
    "carModelEnvStandard",
    "carModelReleaseTime",
    "carModelMaxPower",
    "carModelMaxTorque",
    "carModelSize",
    "carModelMaxSpeed",
    "carModelOfficialSpeedupTime",
    "carModelActualTestSpeedupTime",
    "carModelActualTestBrakeDistance",
    "carModelMiitCompositeFuelConsumption",
    "carModelActualFuelConsumption",
    "carModelMiitEnduranceMileagePureElectric",
    "carModelQuickCharge",
    "carModelSlowCharge",
    "carModelQuickChargePercent",
    "carModelHorsePowerElectric",
    "carModelActualTestEnduranceMileage",
    "carModelActualTestQuickCharge",
    "carModelActualTestSlowCharge",
    "carModelWholeWarranty",
]
```

```
for eachCarKey in carAllKeyList:  
    if eachCarKey not in carModelDict:  
        print("found miss key: %s" % eachCarKey)  
        carModelDict[eachCarKey] = ""  
  
carModelSpecUrl = carModelDict["carModelSpecUrl"]  
self.send_message(self.project_name, carModelDict,  
  
def on_message(self, project, msg):  
    print("on_message: msg=%s" % msg)  
    return msg
```

crifan.com, 使用署名4.0国际(CC BY 4.0)协议发布 all right reserved,
powered by Gitbook最后更新: 2021-09-08 13:56:21

输出结果

全部结果： autohome_20200901_car44451.xlsx

- 在线下载：[autohome_20200901_car44451.xlsx](#)
- 截图效果

The screenshot displays two Excel spreadsheets. The top spreadsheet, titled 'autohome_20200901_car44451', has columns A through M and contains approximately 40 rows of data. The bottom spreadsheet, also titled 'autohome_20200901_car44451!\$A\$44452:\$M\$6429', has columns A through M and contains approximately 20 rows of data, showing more detailed information such as engine type and body structure.

单个结果举例

单个车型车系的详细数据

汽油

- <https://www.autohome.com.cn/spec/43593/>

```
{  
    "carBrandId": "33",  
    "carBrandLogoUrl": "https://car2.autoimg.cn/cardata/audi/logo_160x160.jpg",  
    "carBrandName": "奥迪",  
    "carEnergyType": "汽油",  
    "carMerchantName": "一汽-大众奥迪",  
    "carMerchantUrl": "https://car.autohome.com.cn/partnerDetail/10000000000000000000000000000000.html",  
    "carModelActualFuelConsumption": "",  
    "carModelActualTestBrakeDistance": "",  
    "carModelActualTestSpeedupTime": "",  
    "carModelBodyStructure": "5门5座两厢车",  
    "carModelDataSift2": "国VI",  
    "carModelDataSift3": "1.4T",  
    "carModelDataSift4": "7挡双离合",  
    "carModelDriveType": "前置前驱",  
    "carModelEngine": "1.4T 150马力 L4",  
    "carModelEnvStandard": "国VI",  
    "carModelGearBox": "7挡双离合",  
    "carModelGroupName": "1.4升 涡轮增压 150马力 国VI",  
    "carModelMaxPower": "110",  
    "carModelMaxSpeed": "200",  
    "carModelMaxTorque": "250",  
    "carModelMiltCompositeFuelConsumption": "5.8",  
    "carModelMsrp": "19.32万",  
    "carModelName": "2020款 改款 Sportback 35 TFSI 进取型",  
    "carModelOfficialSpeedupTime": "8.4",  
    "carModelReleaseTime": "2020.04",  
    "carModelSize": "4312*1785*1426",  
    "carModelSpecId": "43593",  
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/43593.html",  
    "carModelWholeWarranty": "三年或10万公里",  
    "carModelYear": "2020款",  
    "carSeriesId": "3170",  
    "carSeriesLevelId": "3",  
    "carSeriesLevelName": "紧凑型车",  
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/v3/webseries/10000000000000000000000000000000/3170/3170_main.jpg",  
    "carSeriesMaxPrice": "23.46万",  
    "carSeriesMinPrice": "19.32万",  
    "carSeriesMsrp": "19.32-23.46万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/spec/3170.html",  
    "carSeriesName": "奥迪A3",  
    "carSeriesUrl": "https://www.autohome.com.cn/3170.html"  
}
```

- <https://car.autohome.com.cn/config/spec/16147.html>

```
{  
    "carBrandId": "91",  
    "carBrandLogoUrl": "https://car3.autoimg.cn/cardata/  
    "carBrandName": "红旗",  
    "carEnergyType": "汽油",  
    "carMerchantName": "一汽红旗",  
    "carMerchantUrl": "https://car.autohome.com.cn/pr/  
    "carModelActualFuelConsumption": "",  
    "carModelActualTestBrakeDistance": "",  
    "carModelActualTestSpeedupTime": "",  
    "carModelBodyStructure": "4门5座三厢车",  
    "carModelDriveType": "后驱",  
    "carModelEngine": "2.0T 204马力 L4",  
    "carModelEnvStandard": "国IV(国V)",  
    "carModelGearBox": "6挡手自一体",  
    "carModelGroupName": "2.0升 涡轮增压 204马力",  
    "carModelMaxPower": "150",  
    "carModelMaxSpeed": "",  
    "carModelMaxTorque": "260",  
    "carModelMiltCompositeFuelConsumption": "9.8",  
    "carModelMsrp": "37.98万",  
    "carModelName": "2013款 2.0T 尊贵型",  
    "carModelOfficialSpeedupTime": "",  
    "carModelReleaseTime": "2013.05",  
    "carModelSize": "5095*1875*1485",  
    "carModelSpecId": "16147",  
    "carModelSpecUrl": "https://www.autohome.com.cn/s/  
    "carModelWholeWarranty": "四年或10万公里",  
    "carModelYear": "2013款",  
    "carSeriesId": "2771",  
    "carSeriesLevelId": "5",  
    "carSeriesLevelName": "中大型车",  
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/c/  
    "carSeriesMaxPrice": "31.78万",  
    "carSeriesMinPrice": "25.28万",  
    "carSeriesMsrp": "25.28-31.78万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/  
    "carSeriesName": "红旗H7",  
    "carSeriesUrl": "https://www.autohome.com.cn/2771  
}
```

- <https://www.autohome.com.cn/spec/46144/>

```
{  
    "carBrandId": "36",  
    "carBrandLogoUrl": "https://car3.autoimg.cn/cardata/  
    "carBrandName": "奔驰",  
    "carEnergyType": "汽油",  
    "carMerchantName": "北京奔驰",  
    "carMerchantUrl": "https://car.autohome.com.cn/pr/  
    "carModelActualFuelConsumption": "",  
    "carModelActualTestBrakeDistance": "39.01",  
    "carModelActualTestEnduranceMileage": "",  
    "carModelActualTestQuickCharge": "",  
    "carModelActualTestSlowCharge": "",  
    "carModelActualTestSpeedupTime": "9.01",  
    "carModelBodyStructure": "5门5座SUV",  
    "carModelDataSift2": "国VI",  
    "carModelDataSift3": "1.3T",  
    "carModelDataSift4": "7挡双离合",  
    "carModelDriveType": "前置前驱",  
    "carModelEngine": "1.3T 163马力 L4",  
    "carModelEnvStandard": "国VI",  
    "carModelGearBox": "7挡双离合",  
    "carModelGroupName": "1.3升 涡轮增压 163马力 国VI",  
    "carModelHorsePowerElectric": "",  
    "carModelMaxPower": "120",  
    "carModelMaxSpeed": "207",  
    "carModelMaxTorque": "250",  
    "carModelMiitCompositeFuelConsumption": "6.1",  
    "carModelMiitEnduranceMileagePureElectric": "",  
    "carModelMsrp": "30.38万",  
    "carmodelName": "2020款 GLA 200",  
    "carModelOfficialSpeedupTime": "9",  
    "carModelQuickCharge": "",  
    "carModelQuickChargePercent": "",  
    "carModelReleaseTime": "2020.07",  
    "carModelSize": "4417*1834*1610",  
    "carModelSlowCharge": "",  
    "carModelSpecId": "46144",  
    "carModelSpecUrl": "https://www.autohome.com.cn/s/  
    "carModelWholeWarranty": "三年不限公里",  
    "carModelYear": "2020款",  
    "carSeriesId": "3248",  
    "carSeriesLevelId": "17",  
    "carSeriesLevelName": "紧凑型SUV",  
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/c/  
    "carSeriesMaxPrice": "30.38万",  
    "carSeriesMinPrice": "30.38万",  
    "carSeriesMsrp": "30.38-30.38万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/  
    "carSeriesName": "奔驰GLA",  
}
```

```
"carSeriesUrl": "https://www.autohome.com.cn/324c
```

```
}
```

插电式混合动力

<https://www.autohome.com.cn/spec/37077/>

```
{  
    "carBrandId": "33",  
    "carBrandLogoUrl": "https://car2.autoimg.cn/cardfs/ser:  
    "carBrandName": "奥迪",  
    "carEnergyType": "插电式混合动力",  
    "carMerchantName": "奥迪(进口)",  
    "carMerchantUrl": "https://car.autohome.com.cn/price/bi  
    "carModelActualFuelConsumption": "",  
    "carModelActualTestBrakeDistance": "",  
    "carModelActualTestEnduranceMileage": "",  
    "carModelActualTestQuickCharge": "",  
    "carModelActualTestSlowCharge": "",  
    "carModelActualTestSpeedupTime": "",  
    "carModelBodyStructure": "5门5座SUV",  
    "carModelDataSift2": "国V",  
    "carModelDataSift3": "2.0T",  
    "carModelDataSift4": "8挡手自一体",  
    "carModelDriveType": "前置四驱",  
    "carModelEngine": "2.0T 252马力 L4",  
    "carModelEnvStandard": "国V",  
    "carModelGearBox": "8挡手自一体",  
    "carModelGroupName": "2.0升 涡轮增压 252马力 国V",  
    "carModelHorsePowerElectric": "128",  
    "carModelMaxPower": "270",  
    "carModelMaxSpeed": "228",  
    "carModelMaxTorque": "700",  
    "carModelMiltCompositeFuelConsumption": "2.4",  
    "carModelMiltEnduranceMileagePureElectric": "56",  
    "carModelMsrp": "79.08万",  
    "carModelName": "2019款 55 e-tron",  
    "carModelOfficialSpeedupTime": "5.9",  
    "carModelQuickCharge": "2.5",  
    "carModelQuickChargePercent": "",  
    "carModelReleaseTime": "2018.11",  
    "carModelSize": "5071*1968*1716",  
    "carModelSlowCharge": "10.8",  
    "carModelSpecId": "37077",  
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/37077",  
    "carModelWholeWarranty": "三年或10万公里",  
    "carModelYear": "2019款",  
    "carSeriesId": "4460",  
    "carSeriesLevelId": "19",  
    "carSeriesLevelName": "中大型SUV",  
    "carSeriesMainImgUrl": "https://car3.autoimg.cn/cardfs/ser:  
    "carSeriesMaxPrice": "79.08万",  
    "carSeriesMinPrice": "79.08万",  
    "carSeriesMsrp": "79.08-79.08万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/4460/",  
    "carSeriesName": "奥迪Q7新能源",  
}
```

```
"carSeriesUrl": "https://www.autohome.com.cn/4460/#level1",  
}
```

纯电动

<https://www.autohome.com.cn/spec/42875/>

```
{  
    "carBrandId": "33",  
    "carBrandLogoUrl": "https://car2.autoimg.cn/cardfs/ser:  
    "carBrandName": "奥迪",  
    "carEnergyType": "纯电动",  
    "carMerchantName": "一汽-大众奥迪",  
    "carMerchantUrl": "https://car.autohome.com.cn/price/bi  
    "carModelActualTestBrakeDistance": "",  
    "carModelActualTestEnduranceMileage": "",  
    "carModelActualTestQuickCharge": "",  
    "carModelActualTestSlowCharge": "",  
    "carModelActualTestSpeedupTime": "",  
    "carModelBodyStructure": "5门5座SUV",  
    "carModelDataSift2": "100Kw",  
    "carModelDataSift3": "265公里",  
    "carModelDataSift4": "单速",  
    "carModelDriveType": "前置前驱",  
    "carModelGearBox": "电动车单速变速箱",  
    "carModelGroupName": "电动 136马力",  
    "carModelHorsePowerElectric": "136",  
    "carModelMaxPower": "100",  
    "carModelMaxSpeed": "150",  
    "carModelMaxTorque": "290",  
    "carModelMiitEnduranceMileagePureElectric": "265",  
    "carModelMsrp": "22.68万",  
    "carModelName": "2019款 Q2L e-tron 纯电智酷型",  
    "carModelOfficialSpeedupTime": "",  
    "carModelQuickCharge": "0.6",  
    "carModelQuickChargePercent": "80",  
    "carModelReleaseTime": "2019.11",  
    "carModelSize": "4237*1785*1548",  
    "carModelSlowCharge": "17",  
    "carModelSpecId": "42875",  
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/42  
    "carModelWholeWarranty": "三年或10万公里",  
    "carModelYear": "2019款",  
    "carSeriesId": "5240",  
    "carSeriesLevelId": "16",  
    "carSeriesLevelName": "小型SUV",  
    "carSeriesMainImgUrl": "https://car2.autoimg.cn/cardfs/  
    "carSeriesMaxPrice": "23.73万",  
    "carSeriesMinPrice": "22.68万",  
    "carSeriesMsrp": "22.68-23.73万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/5240/  
    "carSeriesName": "奥迪Q2L e-tron",  
    "carSeriesUrl": "https://www.autohome.com.cn/5240/#lev  
}
```

油电混合

安装和启动的常见问题

<https://car.autohome.com.cn/config/spec/35507.html>

```
{  
    "carBrandId": "52",  
    "carBrandLogoUrl": "https://car2.autoimg.cn/cardfs/service/carbrand/logo/52/52_1536x1536_1x1_1600x1600.jpg",  
    "carBrandName": "雷克萨斯",  
    "carEnergyType": "油电混合",  
    "carMerchantName": "雷克萨斯",  
    "carMerchantUrl": "https://car.autohome.com.cn/price/bigdata/52",  
    "carModelActualFuelConsumption": "",  
    "carModelActualTestBrakeDistance": "",  
    "carModelActualTestEnduranceMileage": "",  
    "carModelActualTestQuickCharge": "",  
    "carModelActualTestSlowCharge": "",  
    "carModelActualTestSpeedupTime": "",  
    "carModelBodyStructure": "5门5座两厢车",  
    "carModelDataSift2": "",  
    "carModelDataSift3": "",  
    "carModelDataSift4": "",  
    "carModelDriveType": "前驱",  
    "carModelEngine": "1.8L 99马力 L4",  
    "carModelEnvStandard": "国IV(国V)",  
    "carModelGearBox": "E-CVT无级变速",  
    "carModelGroupName": "1.8升 99马力",  
    "carModelHorsePowerElectric": "82",  
    "carModelMaxPower": "100",  
    "carModelMaxSpeed": "",  
    "carModelMaxTorque": "",  
    "carModelMiltCompositeFuelConsumption": "4.6",  
    "carModelMiltEnduranceMileagePureElectric": "",  
    "carModelMsrp": "26.70万",  
    "carModelName": "2018款 CT200h 多彩生活特别限量版",  
    "carModelOfficialSpeedupTime": "",  
    "carModelQuickCharge": "",  
    "carModelQuickChargePercent": "",  
    "carModelReleaseTime": "2018.08",  
    "carModelSize": "4360*1765*1455",  
    "carModelSlowCharge": "",  
    "carModelSpecId": "35507",  
    "carModelSpecUrl": "https://www.autohome.com.cn/spec/35507.html",  
    "carModelWholeWarranty": "六年或15万公里",  
    "carModelYear": "2018款",  
    "carSeriesId": "2063",  
    "carSeriesLevelId": "3",  
    "carSeriesLevelName": "紧凑型车",  
    "carSeriesMainImgUrl": "https://car2.autoimg.cn/cardfs/service/carseries/mainimg/2063/2063_1536x1536_1x1_1600x1600.jpg",  
    "carSeriesMaxPrice": "28.20万",  
    "carSeriesMinPrice": "21.50万",  
    "carSeriesMsrp": "21.50-28.20万",  
    "carSeriesMsrpUrl": "https://www.autohome.com.cn/2063/",  
    "carSeriesName": "雷克萨斯CT",  
}
```

```
"carSeriesUrl": "https://www.autohome.com.cn/2063/#level1",
```

```
}
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新: 2021-09-08 13:57:43

百度热榜

此处以 爬取百度热榜的内容列表 为例，去演示如何使用 PySpider

代

码 **crawlBaiduHotList_PySpider_1501**

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
# Created on 2020-07-31 15:01:00
# Project: crawlBaiduHotList_PySpider_1501

from pyspider.libs.base_handler import *
from pyspider.database import connect_database

class Handler(BaseHandler):
    crawl_config = {
    }

    # @every(minutes=24 * 60)
    def on_start(self):
        UserAgent_Chrome_Mac = "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36"
        curHeaderDict = {
            "User-Agent": UserAgent_Chrome_Mac,
        }
        self.crawl('https://www.baidu.com/', callback=self.on_baiduHome)

    # @config(age=10 * 24 * 60 * 60)
    def baiduHome(self, response):
        # for eachItem in response.doc('span[class="title-item"]'):
        titleItemGenerator = response.doc('span[class="title-item"]')
        titleItemList = list(titleItemGenerator)
        print("titleItemList=%s" % titleItemList)
        # for eachItem in titleItemList:
        for curIdx, eachItem in enumerate(titleItemList):
            print("[%d] eachItem=%s" % (curIdx, eachItem))
            itemTitleStr = eachItem.text()
            print("itemTitleStr=%s" % itemTitleStr)
            curUrl = "%s#%d" % (response.url, curIdx)
            print("curUrl=%s" % curUrl)
            curResult = {
                # "url": response.url,
                # "url": curUrl,
                "百度热榜标题": itemTitleStr,
            }
            # return curResult
            # self.send_message(self.project_name, curResult)
            self.send_message(self.project_name, curResult)

    def on_message(self, project, msg):
```

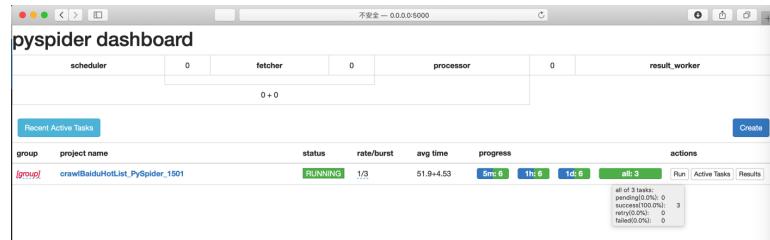
```
print("on_message: msg=", msg)
return msg
```

效果和结果

- 调试

```
pySpider > crawlBaiduHotList_PySpider_1501
pySpider > crawlBaiduHotList_PySpider_1501
run
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 # Created on 2020-5-7-3 15:01:00
4 # Project: crawlBaiduHotList_PySpider_1501
5 from pyspider.lib.base_handler import *
6 from pyspider.database import connect_database
7
8 class Handler(BaseHandler):
9     model_config = {
10         'model': None,
11         'model_name': None,
12         'model_desc': None,
13         'model_fields': None,
14     }
15     def on_start(self):
16         self.set_header('User-Agent', 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.89 Safari/537.36')
17         curlUrl = 'https://www.baidu.com'
18         self.create('User-Agent_Chrome_Mac', curlUrl, 'User-Agent', 'User-Agent_Chrome_Mac',
19                     'curlUrl', 'curlUrl', 'curlUrl')
20         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
21         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
22         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
23         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
24         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
25         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
26         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
27         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
28         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
29         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
30         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
31         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
32         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
33         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
34         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
35         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
36         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
37         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
38         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
39         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
40         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
41         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
42         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
43         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
44         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
45         # self.create('User-Agent_Chrome_Mac', 'User-Agent_Chrome_Mac', 'curlUrl', 'curlUrl', 'curlUrl')
46
47 def on_message(self, project, msg):
48     print('on_message: msg=', msg)
49     return msg
```

- 运行



- 点击 Results -> Results结果

crawlBaiduHotList_PySpider_1501 - Results	
url	百度热搜标题
https://www.baidu.com/#5	"2020ChinaJoy逛展指南"
https://www.baidu.com/#4	"特朗普改口称不推荐选举"
https://www.baidu.com/#3	"19岁贫困男高考后查出白血病"
https://www.baidu.com/#2	"赵正永一审被判死缓"
https://www.baidu.com/#1	"马云粉丝后援会闭站"
https://www.baidu.com/#0	"北斗正式开通"

- 点击 CSV -> 导出CSV

Results - crawlBaiduHotList_PySpider_1501 - pyspider	
url	百度热搜标题
https://www.baidu.com/#5	"2020ChinaJoy逛展指南"
https://www.baidu.com/#4	"特朗普改口称不推荐选举"
https://www.baidu.com/#3	"19岁贫困男高考后查出白血病"
https://www.baidu.com/#2	"赵正永一审被判死缓"
https://www.baidu.com/#1	"马云粉丝后援会闭站"
https://www.baidu.com/#0	"北斗正式开通"

- CSV预览

Preview 'BaiduHotTitleList_pySpider.csv' - item_spider	
url	百度热搜标题
https://www.baidu.com/#5	"2020ChinaJoy逛展指南"
https://www.baidu.com/#4	"特朗普改口称不推荐选举"
https://www.baidu.com/#3	"19岁贫困男高考后查出白血病"
https://www.baidu.com/#2	"赵正永一审被判死缓"
https://www.baidu.com/#1	"马云粉丝后援会闭站"
https://www.baidu.com/#0	"北斗正式开通"

更多细节

详见：

【已解决】用Python爬虫框架PySpider实现爬虫爬取百度热榜内容列表

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新：2021-04-14 20:37:47

附录

下面列出相关参考资料。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新: 2020-07-30 14:00:04

参考资料

- 【未解决】用Python爬取汽车之家的车型车系详细数据
- 【已解决】Mac中恢复或重新搭建PySpider开发环境
- 【已解决】PySpider访问汽车之家报错:
`requests.exceptions.HTTPError HTTP 403 Forbidden`
- 【已解决】PySpider中获取PyQuery获取到节点的子元素
- 【已解决】PySpider中PyQuery如何匹配某个元素中多个属性值
- 【已解决】汽车之家车型车系数据：支持新版车系页面
- 【已解决】汽车之家车型车系数据：支持旧版车系页面
- 【已解决】汽车之家车型车系数据：抓取车型的详细参数配置
- 【未解决】汽车之家车型车系数据：想办法破解js和css加密的文字显示
- 【已解决】汽车之家车型车系数据：解决部分车型基本参数字段个数不一致问题
- 【已解决】汽车之家车型车系数据：优化去掉js加速抓取车型参数配置
- 【规避解决】汽车之家配置页面无法直接获取到文字数据
- 【已解决】汽车之家车型车系数据：carBrandId是空
- 【未解决】汽车之家车型车系数据：能源类型是空白的车型
- 【已解决】汽车之家车型车系数据：车身结构的值包含span标签
- 【未解决】汽车之家车型车系数据：有些参数字段的值包含span特殊标签
- 【已解决】汽车之家车型车系数据：抓取车型的详细参数配置
- 【已解决】车型车系数据缺失如红旗H5等部分车型数据
-
- 【已解决】用Python爬虫框架PySpider实现爬虫爬取百度热榜内容列表
- 【已解决】PySpider中如何在单个页面返回多个结果保存到自带的Results页面中的列表中
- 【已解决】PySpider抓包百度热榜标题列表结果
- 【已解决】Mac中安装phantomjs
- 【已解决】Mac中启动PySpider
- 【已解决】Mac中pip安装pycurl报错：`fatal error openssl/ssl.h file not found`
- 【已解决】Mac中给Python3安装PySpider
- 【已解决】CentOS7中安装全局的PySpider并配置和运行
- 【已解决】CentOS中全局的PySpider中如何指定data目录位置
- 【未解决】PySpider运行批量下载时报错：`HTTP 599 Operation timed out after milliseconds with out of bytes received`
- 【已解决】PySpider无法继续爬取剩余绘本数据
- 【已解决】PySpider中如何更改默认5000端口

- 【已解决】 PySpider中使用IP代理proxy出错:
requests.exceptions.HTTPError HTTP 404 Not Found
-
- 【已解决】 写Python爬虫爬取汽车之家品牌车系车型数据 – 在路上
- 【无法解决】 PySpider的部署运行而非调试界面上RUN运行
- HTML解析库Python版jQuery: PyQuery
- pyspider是开源强大的python爬虫系统 - pyspider中文网
- 【已解决】 PySpider中保存数据到mysql
- 【已解决】 PySpider中如何清空之前运行的数据和正在运行的任务 – 在路上
- 【未解决】 pyspider中如何给phantomjs传递额外参数 – 在路上
- 【已解决】 PySpider中页面部分内容不显示 – 在路上
- 【未解决】 pyspider运行出错: FETCH_ERROR HTTP 599
Connection timed out after milliseconds
- 【记录】 Mac中安装和运行pyspider
- 【整理】 pyspider vs scrapy
- 【已解决】 pyspider中phantomjs中的proxy是什么意思 – 在路上
- 【已解决】 pyspider中运行result_worker出错:
ModuleNotFoundError No module named mysql
- 【已解决】 PySpider中传递参数给下一级且当下一级失败时也可以执行
- 【已解决】 pyspider中出错: TypeError __init__() got an unexpected keyword argument resultdb – 在路上
- 【已解决】 pyspider运行出错: ImportError pycurl libcurl link-time ssl backend (openssl) is different from compile-time ssl backend (none/other)
- 【已解决】 pyspider中pymysql中insert失败且except部分代码没有执行
- 【已解决】 pyspider运行出错: Error Could not create web server listening on port 25555 – 在路上
- 【已解决】 pyspider中的css选择器不工作 – 在路上
- 【已解决】 pyspider中如何写规则去提取网页内容 – 在路上
- 【已解决】 PySpider如何把json结果数据保存到csv或excel文件中 – 在路上
- 【已解决】 PySpider中如何单个页面返回多个json数据结果 – 在路上
- 【已解决】 Mac或Win中用Excel打开UTF8编码的csv文件显示乱码
- 【已解决】 pyspider中如何加载汽车之家页面中的更多内容 – 在路上
- 【已解决】 PySpider中如何发送POST请求且传递格式为application/x-www-form-urlencoded的form data参数 – 在路上
- 【已解决】 PySpider中如何强制让重复的url地址继续爬取 – 在路上
-
- 安装pyspider遇到的坑 (python3.6) _盛夏88688的博客-CSDN博客 _python 3.6 with报错use async with instead
- How to delete a spider and the data? · Issue #380 · binux/pyspider

- [Frequently Asked Questions - pyspider](#)
- [python爬虫 - crawl 连接网页超时, HTTP 599 - SegmentFault 思否](#)
-

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved,
powered by Gitbook最后更新: 2021-09-08 13:44:31