**关于Scrapy的调研报告**

班级：07111505

组别：第8组

学号：1120151831

姓名：李继领

2017年9月26日

本报告从下面几个方面展开对Scrapy的调研的内容：1、Scrapy简介； 2、Scrapy的架构； 3、Scrapy的运行流程； 4、Scrapy的搭建与简单使用。

# 一、Scrapy简介

Scrapy是python开发的一个快速、高层次的屏幕抓取和web抓取框架，用于抓取web站点并从页面中提取结构化的数据。

Scrapy最初是为了页面抓取（更确切地说，网络抓取）而设计的，也可以应用于获取API所返回的数据或者通用的网络爬虫。Scrapy应用广泛，可以用于数据挖掘、监测和自动化测试等。

Scrapy拥有以下优点：

（1）快速、强大：用户书写爬取数据的规则，剩余任务交给Scrapy来完成；（2）易扩展：可扩展设计，添加新功能而不需要触及核心；

（3）便携： 用python书写（不支持python3），可以运行在Linux，Windows,Mac和BSD上

Scrapy是python还开发的基于twisted的爬虫框架，除Scrapy外，python还开发出其他类型的网络爬虫框架:

grab - 网络爬虫框架（基于pycurl/multicar）

pyspider – 一个强大的爬虫系统

cola – 一个分布式爬虫框架

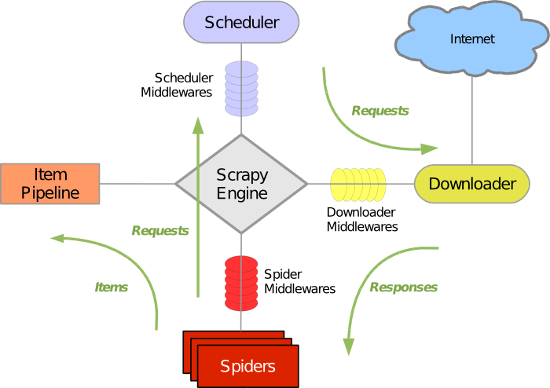
portia – 基于Scrapy的可视化爬虫

restkit – python的HTTP资源工具包

demiurge – 基于PyQuary的爬虫微框架

# 二、Scrapy的架构

Scrapy 使用了 Twisted异步网络库来处理网络通讯。整体架构大致如下图所示：



Scrapy主要包括了以下组件：

**（1）引擎(Scrapy)**  
用来处理整个系统的数据流处理, 触发事务(框架核心)

**（2）调度器(Scheduler)**  
用来接受引擎发过来的请求, 压入队列中, 并在引擎再次请求的时候返回. 可以想像成一个URL（抓取网页的网址或者说是链接）的优先队列, 由它来决定下一个要抓取的网址是什么, 同时去除重复的网址

**（3）下载器(Downloader)**  
用于下载网页内容, 并将网页内容返回给蜘蛛(Scrapy下载器是建立在twisted这个高效的异步模型上的)

**（4）爬虫(Spiders)**  
爬虫是主要干活的, 用于从特定的网页中提取自己需要的信息, 即所谓的实体(Item)。用户也可以从中提取出链接,让Scrapy继续抓取下一个页面

**（5）项目管道(Pipeline)**  
负责处理爬虫从网页中抽取的实体，主要的功能是持久化实体、验证实体的有效性、清除不需要的信息。当页面被爬虫解析后，将被发送到项目管道，并经过几个特定的次序处理数据。

**（6）下载器中间件(Downloader Middlewares)**  
位于Scrapy引擎和下载器之间的框架，主要是处理Scrapy引擎与下载器之间的请求及响应。

**（7）爬虫中间件(Spider Middlewares)**  
介于Scrapy引擎和爬虫之间的框架，主要工作是处理蜘蛛的响应输入和请求输出

**（8）调度中间件(Scheduler Middewares)**  
介于Scrapy引擎和调度之间的中间件，从Scrapy引擎发送到调度的请求和响应。

# 三、Scrapy运行流程

Scrapy运行流程大概如下：

引擎从调度器中取出一个链接(URL)用于接下来的抓取

引擎把URL封装成一个请求(Request)传给下载器

引擎把URL封装成一个请求(Request)传给下载器

爬虫解析Response

解析出的是链接（URL）,则把URL交给调度器等待抓取

解析出实体（Item）,则交给实体管道进行进一步的处理

下载器把资源下载下来，并封装成应答包(Response)

# 四、Scrapy的搭建与简单使用

## （1）安装

Scrapy框架运行平台及相关辅助工具

* [Python](http://www.python.org/) 2.7（Python最新版3.5，这里选择了2.7版本）
* Python Package: [pip](http://www.pip-installer.org/en/latest/installing.html) and [setuptools](https://pypi.python.org/pypi/setuptools). 现在 [pip](http://www.pip-installer.org/en/latest/installing.html) 依赖 [setuptools](https://pypi.python.org/pypi/setuptools) ，如果未安装，则会自动安装[setuptools](https://pypi.python.org/pypi/setuptools) 。
* [lxml](http://lxml.de/). 大多数Linux发行版自带了lxml。如果缺失，请查看http://lxml.de/installation.html
* [OpenSSL](https://pypi.python.org/pypi/pyOpenSSL). 除了Windows(请查看 [平台安装指南](http://scrapy-chs.readthedocs.io/zh_CN/latest/intro/install.html#intro-install-platform-notes))之外的系统都已经提供。

您可以使用pip来安装Scrapy(推荐使用pip来安装Python package).

 pip install Scrapy

Windows下安装流程：

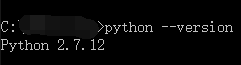
1、安装Python 2.7之后，您需要修改 PATH 环境变量，将Python的可执行程序及额外的脚本添加到系统路径中。将以下路径添加到 PATH 中:

 C:\Python27\;C:\Python27\Scripts\;

除此之外，还可以用cmd命令来设置Path：

c:\python27\python.exe c:\python27\tools\scripts\win\_add2path.py

安装配置完成之后，可以执行命令python --version查看安装的python版本。（如图所示）



2、从 <http://sourceforge.net/projects/pywin32/> 安装 *pywin32*

请确认下载符合您系统的版本(win32或者amd64)

从 <https://pip.pypa.io/en/latest/installing.html> 安装 [pip](http://www.pip-installer.org/en/latest/installing.html)

3、打开命令行窗口，确认 pip 被正确安装:

 pip --version

4、到目前为止Python 2.7 及 pip 已经可以正确运行了。接下来安装Scrapy:

 pip install Scrapy

至此windows下Scrapy安装已经结束。

## （2）Scrapy简单使用

1、在cmd中创建Scrapy项目工程。

 scrapy startproject tutorial

[复制代码](javascript:void(0);)

H:\python\scrapyDemo>scrapy startproject tutorial

New Scrapy project 'tutorial', using template directory 'f:\\python27\\lib\\site-packages\\scrapy\\templates\\project', created in:

H:\python\scrapyDemo\tutorial

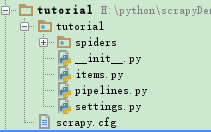
You can start your first spider with:

cd tutorial

scrapy genspider example example.com

[复制代码](javascript:void(0);)

2、文件目录结构如下：



解析scrapy框架结构：

* scrapy.cfg: 项目的配置文件。
* tutorial/: 该项目的python模块。之后您将在此加入代码。
* tutorial/items.py: 项目中的item文件。
* tutorial/pipelines.py: 项目中的pipelines文件。
* tutorial/settings.py: 项目的设置文件。
* tutorial/spiders/: 放置spider代码的目录。

3、编写简单的爬虫

1、在item.py中配置需采集页面的字段实例。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2

3 # Define here the models for your scraped items

4 #

5 # See documentation in:

6 # http://doc.scrapy.org/en/latest/topics/items.html

7

8 import scrapy

9 from scrapy.item import Item, Field

10

11 class TutorialItem(Item):

12 title = Field()

13 author = Field()

14 releasedate = Field()

[复制代码](javascript:void(0);)

2、在tutorial/spiders/spider.py中书写要采集的网站以及分别采集各字段。

[复制代码](javascript:void(0);)

1 # -\*-coding:utf-8-\*-

2 import sys

3 from scrapy.linkextractors.sgml import SgmlLinkExtractor

4 from scrapy.spiders import CrawlSpider, Rule

5 from tutorial.items import TutorialItem

6

7 reload(sys)

8 sys.setdefaultencoding("utf-8")

9

10

11 class ListSpider(CrawlSpider):

12 # 爬虫名称

13 name = "tutorial"

14 # 设置下载延时

15 download\_delay = 1

16 # 允许域名

17 allowed\_domains = ["news.cnblogs.com"]

18 # 开始URL

19 start\_urls = [

20 "https://news.cnblogs.com"

21 ]

22 # 爬取规则,不带callback表示向该类url递归爬取

23 rules = (

24 Rule(SgmlLinkExtractor(allow=(r'https://news.cnblogs.com/n/page/\d',))),

25 Rule(SgmlLinkExtractor(allow=(r'https://news.cnblogs.com/n/\d+',)), callback='parse\_content'),

26 )

27

28 # 解析内容函数

29 def parse\_content(self, response):

30 item = TutorialItem()

31

32 # 当前URL

33 title = response.selector.xpath('//div[@id="news\_title"]')[0].extract().decode('utf-8')

34 item['title'] = title

35

36 author = response.selector.xpath('//div[@id="news\_info"]/span/a/text()')[0].extract().decode('utf-8')

37 item['author'] = author

38

39 releasedate = response.selector.xpath('//div[@id="news\_info"]/span[@class="time"]/text()')[0].extract().decode(

40 'utf-8')

41 item['releasedate'] = releasedate

42

43 yield item

[复制代码](javascript:void(0);)

3、在tutorial/pipelines.py管道中保存数据。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2

3 # Define your item pipelines here

4 #

5 # Don't forget to add your pipeline to the ITEM\_PIPELINES setting

6 # See: http://doc.scrapy.org/en/latest/topics/item-pipeline.html

7 import json

8 import codecs

9

10

11 class TutorialPipeline(object):

12 def \_\_init\_\_(self):

13 self.file = codecs.open('data.json', mode='wb', encoding='utf-8')#数据存储到data.json

14

15 def process\_item(self, item, spider):

16 line = json.dumps(dict(item)) + "\n"

17 self.file.write(line.decode("unicode\_escape"))

18

19 return item

[复制代码](javascript:void(0);)

4、tutorial/settings.py中配置执行环境。

[复制代码](javascript:void(0);)

1 # -\*- coding: utf-8 -\*-

2

3 BOT\_NAME = 'tutorial'

4

5 SPIDER\_MODULES = ['tutorial.spiders']

6 NEWSPIDER\_MODULE = 'tutorial.spiders'

7

8 # 禁止cookies,防止被ban

9 COOKIES\_ENABLED = False

10 COOKIES\_ENABLES = False

11

12 # 设置Pipeline,此处实现数据写入文件

13 ITEM\_PIPELINES = {

14 'tutorial.pipelines.TutorialPipeline': 300

15 }

16

17 # 设置爬虫爬取的最大深度

18 DEPTH\_LIMIT = 100

[复制代码](javascript:void(0);)

5、新建main文件执行爬虫代码。

1 from scrapy import cmdline

2 cmdline.execute("scrapy crawl tutorial".split())

最终，执行main.py后在data.json文件中获取到采集结果的json数据。



**参考文献**

1. scrapy\_百度百科

2.伯乐在线——“Python爬虫的工具列表”

3.博客园——“Python之Scrapy爬虫框架安装及简单使用”

4.技术杂谈——“Python爬虫进阶一之爬虫框架概述”

5.https://scrapy.org/s