# Scrapy爬虫框架使用

Scrapy，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用途广泛，可以用于数据挖掘、监测和[自动化测试](https://baike.baidu.com/item/%E8%87%AA%E5%8A%A8%E5%8C%96%E6%B5%8B%E8%AF%95/9815698)。

## 安装scrapy

打开windows下的cmd命令窗口。

pip3 install scrapy

如果安装提示：

error: Microsoft Visual C++ 14.0 is required.

**则要先安装twisted。**

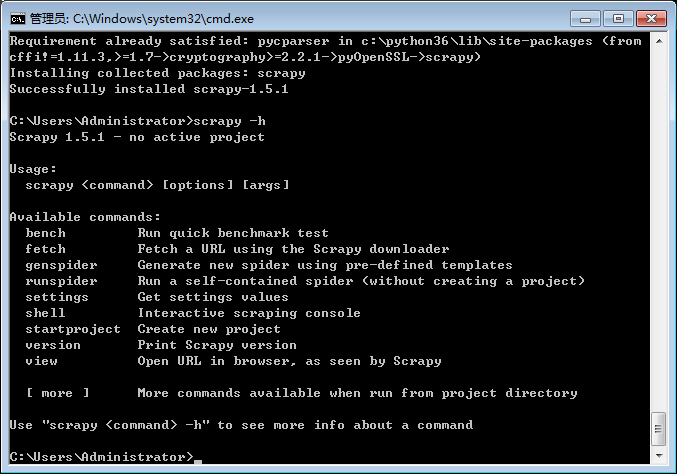
去<http://www.lfd.uci.edu/~gohlke/pythonlibs/#twisted>下载相应的whl文件。（如我的Twisted‑18.7.0‑cp36‑cp36m‑win\_amd64.whl），cp后面是python版本，amd64代表64位，运行命令：

pip3 install C:\Users\Administrator\Downloads\Twisted-18.7.0-cp36-cp36m-win\_amd64.whl

然后再安装scrapy：

pip3 install scrapy

安装完成后，用scrapy –h命令检查是否真正安装成功，如图所示：

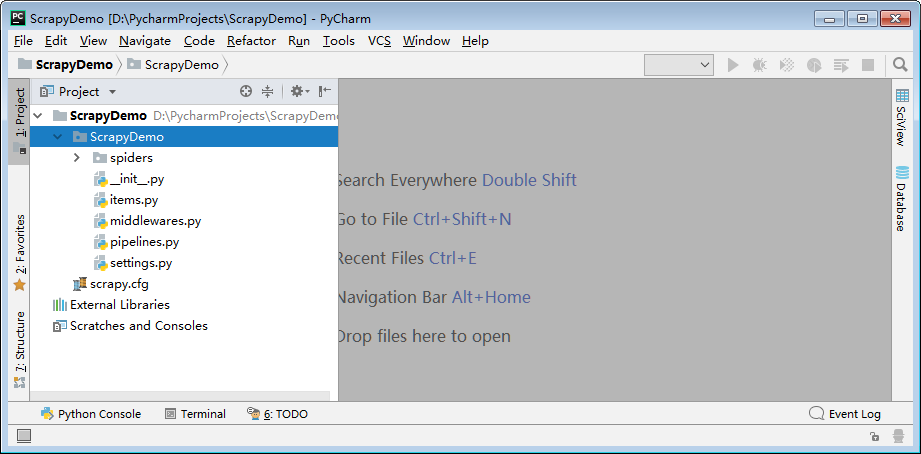


## 新建scrapy工程

（1）、执行命令：

scrapy startproject ScrapyDemo

（2）、在pycharm里打开工程



scrapy.cfg: 项目配置文件

ScrapyDemo/: 项目python模块, 呆会代码将从这里导入

ScrapyDemo/items.py: 项目items文件

ScrapyDemo/pipelines.py: 项目管道文件

ScrapyDemo/settings.py: 项目配置文件

ScrapyDemo/spiders: 放置spider的目录

## 编写scrapy爬虫

（1）、定义item

编辑items.py文件，定义我们要爬取的字段信息。

**import** scrapy  
  
**class** TopicItem(scrapy.Item):  
 title = scrapy.Field()  
 link = scrapy.Field()  
 outline = scrapy.Field()

（2）、在spiders下新建一个爬虫文件tiebar.py

import scrapy  
from scrapy.http import Request  
from ScrapyDemo.items import TopicItem  
  
class TiebarSpider(scrapy.Spider):  
 name = "tiebar" # 这个name是你必须给它一个唯一的名字 后面我们执行文件时的名字  
  
 # 这个列表中的url可以有多个，它会依次都执行，我们这里简单爬取一个  
 start\_urls = ["http://tieba.baidu.com/f?ie=utf-8&kw=python3&red\_tag=o0551587185"]  
  
 next\_url = "http://tieba.baidu.com/f?ie=utf-8&kw=python3"  
  
 def parse(self, response): # 默认函数parse  
 sites = response.xpath('//div[@class="t\_con cleafix"]') # ('匹配你所需信息的路径')  
 # xpath是scrapy里面的一种匹配方式，类似于正则表达式，还有其他几种匹配方式  
 # 这里我们首先获得的是我们需要的信息的那一大块sites。  
  
 print("！！！！！返回信息是：")  
 for i in sites:  
 title = i.xpath('.//div[@class="threadlist\_title pull\_left j\_th\_tit "]/a/text()')  
 if title:  
 title = title.extract()  
 else:  
 continue  
 link = i.xpath('.//div[@class="threadlist\_title pull\_left j\_th\_tit "]/a/@href')  
 if link:  
 link = link.extract()  
 outline = i.xpath('.//div[@class="threadlist\_abs threadlist\_abs\_onlyline "]/text()')  
 if outline:  
 outline = outline.extract()  
 #if title:  
 print (title[0])  
 print (link[0])  
 print (outline[0])  
 item = TopicItem()  
 item['title'] = title[0]  
 item['link'] = link[0]  
 item['outline'] = outline[0]  
 yield item  
  
 nextlink = response.xpath('//div[@class="pagination-default clearfix"]')  
 if nextlink:  
 nextlink = nextlink[0]  
 lstPageUrl = nextlink.xpath('./a/@href').extract()  
 for item in lstPageUrl:  
 yield Request("http:" + item, callback=self.parse)

（3）、保存爬取到的数据

编辑pipelines.py文件。

**import** os  
  
**class** ScrapydemoPipeline(object):  
 **def** process\_item(self, item, spider):  
 base\_dir = os.getcwd()  
 fiename = base\_dir + **'/tiebar\_python3.txt'  
  
 with** open(fiename, **'a'**) **as** f:  
 f.write(str(item[**'title'**]) + **'\n'**)  
 f.write(str(item[**'link'**]) + **'\n'**)  
 f.write(str(item[**'outline'**]) + **'\n\n'**)  
  
 **return** item

这里是保存为文本文件。#需要在setting.py里设置

ITEM\_PIPELINES = {  
 'ScrapyDemo.pipelines.ScrapydemoPipeline': 300,  
}

ScrapydemoPipeline就是我们pipelines.py文件里的类名。

## 运行scrapy爬虫

在pycharm的Terminal窗口里输入：

scrapy crawl tiebar

tiebar就是爬虫的名字，即tiebar.py文件里定义的name。