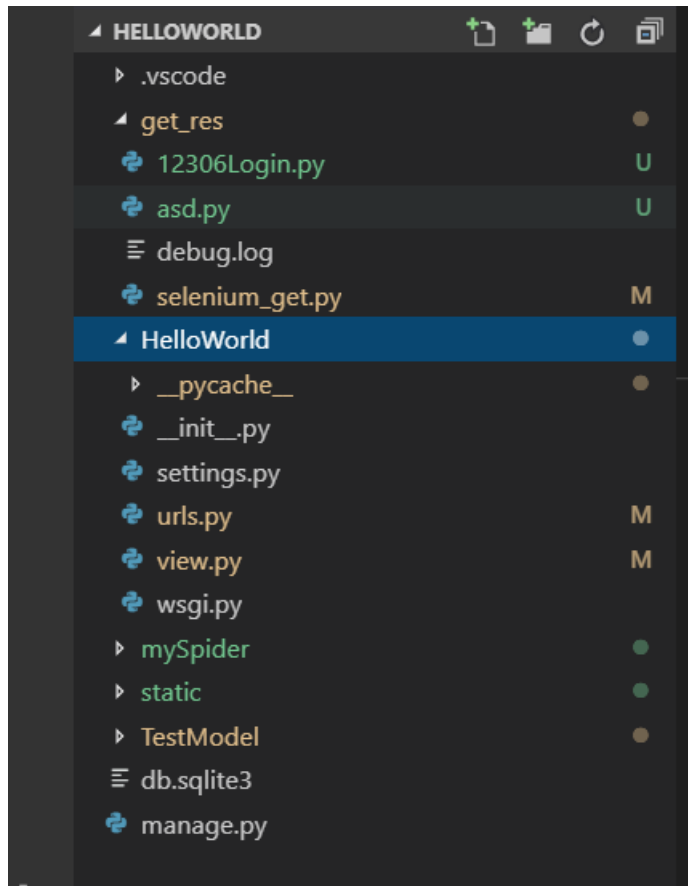


目录

- 当前项目的目录结构.....2
- 模型设计 models.py.....2
- 在 settings 中设置连接数据库 .....4
- Urls 及视图设计 .....5
  - Helloword/urls.py.....5
  - Helloword/view.py .....6
  - Testmodel/urls.py.....6
  - TestModel/views.py .....7
  - 在 settings 中设置路径 .....8
- 前端页面设计.....8
  - 网站拥有页面如下.....8
  - 首页.....9
  - Base.html .....10
  - 爬取豆瓣电影.....13
  - 展示页面 2.....14
  - 动态爬取京东代码为.....17
  - 12306 的自动登录（通过验证码） .....20

基于 Bootstrap 的 Django 网站  
将爬虫爬到的数据显示在网站

当前项目的目录结构



模型设计 models.py

```
# models.py
from django.db import models
from django.contrib.auth.models import User

class Test(models.Model):
    name = models.CharField('分类', max_length=128)

    def __str__(self):
        return self.name

    class Meta:
```

```

        verbose_name='博客分类'
        verbose_name_plural=verbose_name

class Tag(models.Model):
    name=models.CharField('标签',max_length=128)

    def __str__(self):
        return self.name

    class Meta:
        verbose_name='博客标签'
        verbose_name_plural=verbose_name

class Entry(models.Model):
    title = models.CharField('文章标题',max_length=128)
    author = models.ForeignKey(User,verbose_name='作者',on_delete=models.CASCADE)
    img =
models.ImageField(upload_to='blog_img',null=True,blank=True,verbose_name='博客配图')
    body = models.TextField('正文',)
    abstract = models.TextField('摘要',max_length=256,null=True,blank=True)
    visiting = models.PositiveIntegerField('访问量',default=0)
    category = models.ManyToManyField('Test',verbose_name='博客分类')
    tags = models.ManyToManyField('Tag',verbose_name='标签')
    created_time = models.DateTimeField('创建时间',auto_now_add=True)
    modified_time = models.DateTimeField('修改时间',auto_now=True)

    def __str__(self):
        return self.title

    class Meta:
        ordering = ['-created_time']
        verbose_name = '博客正文'
        verbose_name_plural = verbose_name

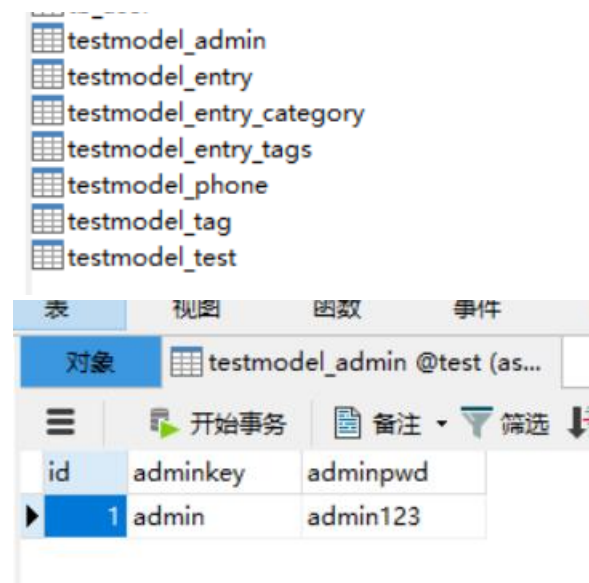
class phone(models.Model):
    img_url=models.CharField(max_length=128)
    link=models.CharField(max_length=128)
    price=models.CharField(max_length=128)
    name=models.CharField(max_length=128)
    comment=models.CharField(max_length=128)

```

在 **settings** 中设置连接数据库

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.mysql',  
        'NAME': 'test',  
        'USER': 'root',  
        'PASSWORD': '123456',  
        'HOST': 'localhost',  
        'PORT': '3306',  
    }  
}
```

数据库设计



The screenshot shows a database management interface. On the left, a list of tables is displayed: testmodel\_admin, testmodel\_entry, testmodel\_entry\_category, testmodel\_entry\_tags, testmodel\_phone, testmodel\_tag, and testmodel\_test. The 'testmodel\_admin' table is selected, and its details are shown in the main area. The table has three columns: id, adminkey, and adminpwd. A single record is visible with id 1, adminkey 'admin', and adminpwd 'admin123'.

表	视图	函数	事件
对象	testmodel_admin @test (as...		
id	adminkey	adminpwd	
1	admin	admin123	

对象 testmodel_admin @test (as...	
id	name
1	肖申克的救赎
2	霸王别姬
3	这个杀手不太冷
4	阿甘正传
5	美丽人生
6	泰坦尼克号
7	千与千寻
8	辛德勒的名单
9	盗梦空间
10	忠犬八公的故事
11	机器人总动员
12	三傻大闹宝莱坞
13	海上钢琴师
14	放牛班的春天
15	楚门的世界

对象 testmodel_phone @test (as...					
id	link	price	name	comment	img_url
1	https://it	5899.00	Apple iPhc	92万+条评价	https://img1
2	https://it	3298.00	【KPL官方	8.6万+条评价	https://img1
3	https://it	3988.00	华为 HUA	3.1万+条评价	https://img1
4	https://it	1299.00	荣耀8X 千	143万+条评价	https://img1
5	https://it	1299.00	荣耀10青	47万+条评价	https://img1
6	https://it	799.00	vivo U1 水	13万+条评价	https://img1
7	https://it	2799.00	荣耀V20 胡	23万+条评价	https://img1
8	https://it	2999.00	OPPO Ren	10+条评价	https://img1

## Urls 及视图设计

### Helloword/urls.py

```
class IndexView(View):
    def get(self, request):
        return render(request, 'TestModel/index.html')

class LoginView(View):
    def get(self, request):
        return render(request, 'TestModel/login.html')
```

```

def post(self,request):
    adminlist=models.admin.objects.filter()
    print(request.POST)
    username=request.POST.get("username",None)
    pwd=request.POST.get("pwd",None)
    print(username,pwd)

    for admin in adminlist:
        if username==admin.adminkey and pwd==admin.adminpwd:
            request.session['username']="管理员"
            request.session.set_expiry(600)
            return redirect("/TestModel/")
        else:

            error_msg="用户名或者密码错误"
            return render(request,"login.html",{"error":error_msg})

urlpatterns = [
    url(r'^admin/',admin.site.urls),
    url(r'^TestModel/',include('TestModel.urls'),name='TestModel'),
    url(r'^$',LoginView.as_view(),name='login')
]

```

Helloword/view.py

```

from django.shortcuts import render
from TestModel import models

def login(request):
    adminlist=models.admin.opject.all()
    return
render(request,'TestModel/login.html',{'adminlist':adminlist})

```

Testmodel/urls.py

```

from django.conf.urls import url
from . import views
app_name = 'TestModel'

```

```
urlpatterns = [
    url(r'^$', views.index,name='blog_index'),
    url(r'^(?P<blog_id>[0-9]+)', views.detail,name='blog_detail'),
    url(r'^show/',views.lists,name='movie_show'),
    url(r'^selemion_show/',views.phone_msg,name="phone_show")
]
```

## TestModel/views.py

```
from django.shortcuts import render
from . import models

def index(request):
    return render(request,'TestModel/index.html',locals())

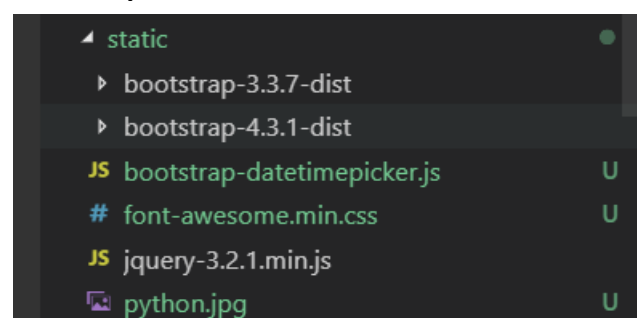
def detail(request,blog_id):

    return render(request,'TestModel/detail.html',locals())

def lists(request):
    movie_list=models.Test.objects.all()
    return
render(request,'TestModel/show.html',{'movie_list':movie_list})

def phone_msg(request):
    phone_list=models.phone.objects.all()
    return
render(request,'TestModel/selemion_show.html',{'phone_list':phone_list}
)
```

Bootstrap 和 font-awesome 放在了 static 目录下

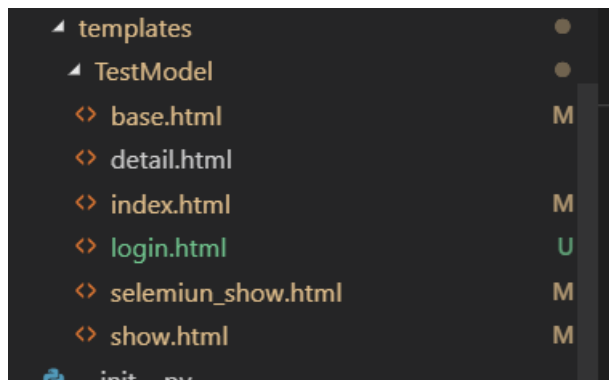


在 settings 中设置路径

```
STATICFILES_DIRS = [  
    os.path.join(BASE_DIR, 'static'),  
]
```

前端页面设计

网站拥有页面如下



其中 **base.html** 是公共页

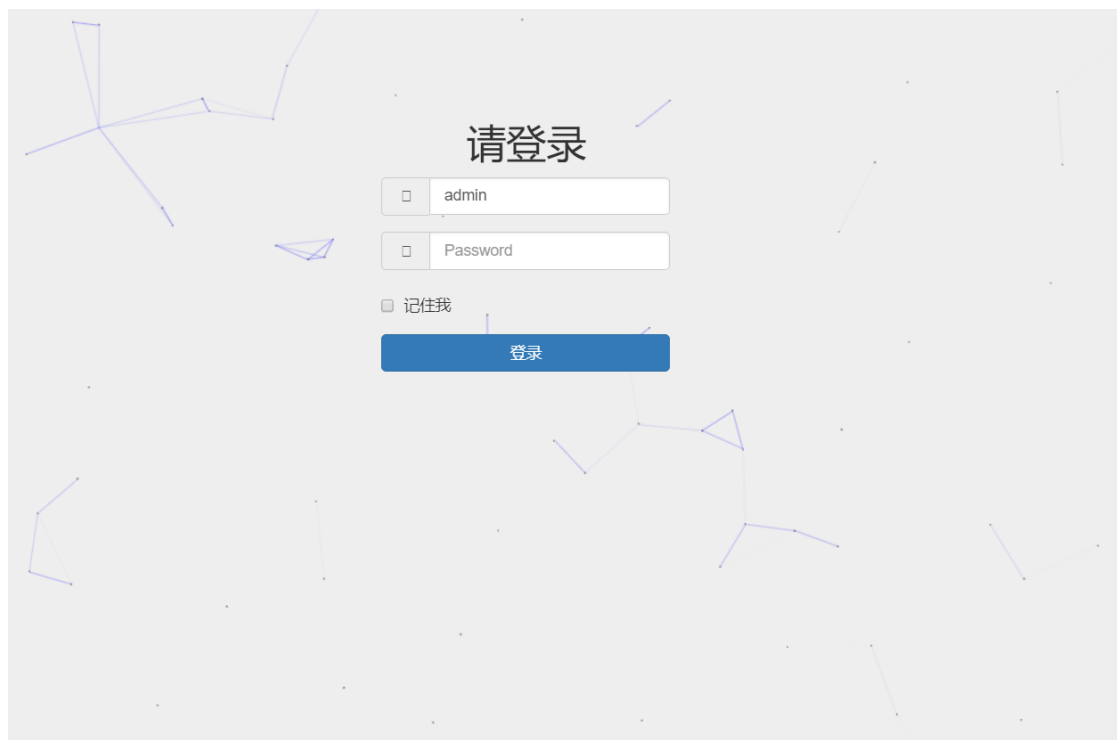
**index.html** 是首页

**login.html** 是登录页面

**Show.html** 是展示爬到的豆瓣电影的数据

**Selemion\_show.html** 展示的是动态爬取的京东手机商品的数据  
登录页面





首页



代码

Index.html

```
{% extends 'TestModel/base.html' %}

{% block title %}博客首页{% endblock %}

{% block content %}
    {% for line in show %}
        <div class="container">
```



```

<body style="background-color:lightgreen">

<nav class="navbar navbar-fixed-top">
  <div class="container-fluid">
    <!-- Brand and toggle get grouped for better mobile display -->
    <div class="navbar-header">
      <button type="button" class="navbar-toggle collapsed"
data-toggle="collapse" data-target="#my-nav" aria-expanded="false">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="/TestModel/">闲人_orz</a>
    </div>

    <!-- Collect the nav links, forms, and other content for toggling -->
    <div class="collapse navbar-collapse" id="my-nav">
      <ul class="nav navbar-nav">
        <li class="active"><a href="/blog/">博客</a></li>
        <li><a href="{%url 'TestModel:movie_show' %}">豆瓣top 电影</a></li>
        <li><a href="{%url 'TestModel:phone_show' %}">京东手机信息
</a></li>
      </ul>

      <form class="navbar-form navbar-left">
        <div class="form-group">
          <input type="text" class="form-control" placeholder="Search">
        </div>
        <button type="submit" class="btn btn-default">搜索</button>
      </form>

      <ul class="nav navbar-nav navbar-right">
        <li class="navbar-brand">欢迎你 {{request.session.username}}</li>
      </ul>

    </div><!-- /.navbar-collapse -->
  </div><!-- /.container-fluid -->
</nav>

{% block content %}{% endblock %}

<footer>
  <div class="footer" role="navigation">

```

```

    <div class="container">
      <div class="navbar-text">
        <ul class="footer-text">
          <li><a href="#">主页</a></li>
          <li><a href="#">联系我们</a></li>
          <li><a href="#">关于博主</a></li>
          <li><a href="#">文档支持</a></li>
          <li><a href="/blog/">博客首页</a></li>
        </ul>
        <p>Copyright © 2019 闲人_orz </p>
      </div>
    </div>
  </div>
</div>
</footer>

<script src="{% static 'jquery-3.2.1.min.js' %}"></script>
<script src="{% static
'bootstrap-3.3.7-dist/js/bootstrap.min.js' %}"></script>
<script type="text/javascript" color="0,0,255" opacity='0.7' zIndex="-2"
count="99"
src="//cdn.bootcss.com/canvas-nest.js/1.0.1/canvas-nest.min.js"></scrip
t>
{% block script %}{% endblock %}

</body>
</html>

```

其中头部和尾部是基于 **bootstrap** 创建的样式  
背景使用了 **canvas-nest** 的粒子效果

爬取豆瓣电影

爬取豆瓣top50电影

排名	电影名
1	肖申克的救赎
2	霸王别姬
3	这个杀手不太冷
4	阿甘正传
5	美丽人生
6	泰坦尼克号
7	千与千寻
8	辛德勒的名单
9	盗梦空间
10	忠犬八公的故事
11	机器人总动员
12	三傻大闹宝莱坞
13	海上钢琴师
14	放牛班的春天
15	楚门的世界
16	大话西游之大圣娶亲

Show.html

```
{% load staticfiles %}
<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="utf-8">
    <title>豆瓣 top25 电影</title>
    <link href="{% static 'bootstrap-3.3.7-dist/css/bootstrap.min.css' %}"
rel="stylesheet">
</head>
<body>
<h1>爬取豆瓣 top50 电影</h1>
<table class="table table-bordered">
    <thead>
    <tr>
```

```

        <th>排名</th>
        <th>电影名</th>
    </tr>
</thead>
<tbody>
{% for line in movie_list %}
    <tr>
        <td>{{line.id}}</td>
        <td>{{line.name}}</td>
    </tr>
{% endfor %}
</tbody>
</table>

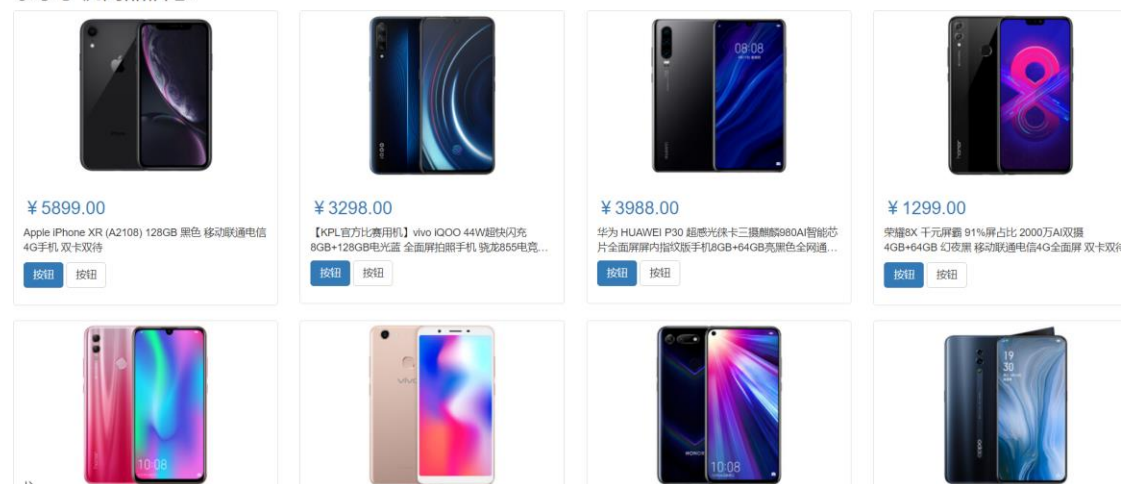
<script src="{% static 'jquery-3.2.1.min.js' %}"></script>
<script src="{% static
'bootstrap-3.3.7-dist/js/bootstrap.min.js' %}"></script>
</body>
</html>

```

其中用了 bootstrap 中的表格的网格样式对数据进行排版

## 展示页面 2

京东手机商品信息



Selemiun.html

```

{% load staticfiles %}
<!DOCTYPE html>
<html lang="zh-CN">

```

```

<head>
  <meta charset="utf-8">
  <title>动态爬取京东手机商品</title>
  <link href="{% static 'bootstrap-3.3.7-dist/css/bootstrap.min.css' %}"
rel="stylesheet">
  <link href="{% static 'TestModel/css/index.css' %}" rel="stylesheet">
</head>
<body>
<h2>京东手机商品信息</h2>
<div class="row">
  {% for line in phone_list %}
    <div class="col-sm-6 col-md-3">
      <div class="thumbnail">
        <a target="_blank" title="asdfa"
href="http://www.baidu.com">
          </a>
          <div class="caption">
            <h3
style="overflow:hidden;white-space:nowrap;text-overflow:ellipsis;"><a
href="{{line.link}}">¥{{line.price}}</a></h3>
            <p>{{line.name}}</p>
            <p>
              <a href="#" class="btn btn-primary" role="button">
                按钮
              </a>
              <a href="#" class="btn btn-default" role="button">
                按钮
              </a>
            </p>
          </div>
        </div>
      </div>
    </div>
  {% endfor %}
</div>
<script src="{% static 'jquery-3.2.1.min.js' %}"></script>
<script src="{% static
'bootstrap-3.3.7-dist/js/bootstrap.min.js' %}"></script>
</body>
</html>

```

用 bootstrap 缩略图的样式将爬到的数据展示出来

其中数据数据是由爬虫所爬得  
豆瓣电影爬虫代码为

```
import requests
import lxml
from lxml import etree
import csv
import pymysql

class Spider:
    def __init__(self, version):
        self.version = version
        self.result = []

    def get_page(self, start_num):
        url = 'https://movie.douban.com/top250?start=%s&filter=' % \
start_num
        res = requests.get(url)

        tree = etree.HTML(res.text)
        top205 = tree.xpath('//span[@class="title"][1]/text()')
        print(top205)
        return top205

    def go(self):
        print("Start..")
        for i in range(0, 1):
            top250 = self.get_page(i * 25)
            self.result += top250

        return self.result

if __name__ == "__main__":
    my_spider = Spider('1.0')
    res = my_spider.go()
    with open('D:/PySy.csv', 'a+', encoding='UTF-8') as csvfile:
        w = csv.writer(csvfile)
        w.writerow(res)

def getImage():
    my_spider = Spider('1.0')
    res = my_spider.go()
    db = pymysql.connect("localhost", "root", "123456", "mysql")
    cursor = db.cursor()
```



```
sql="INSERT INTO MovieTop(movie) VALUES (%s)"
for a in res:
    cursor.execute(sql,(a))
    db.commit()
db.close()
```

## 动态爬取京东代码为

```
from selenium import webdriver
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By
import selenium.common.exceptions
import json
import csv
import time
import pymysql

class JdSpider():
    def open_file(self):
        self.fm = input('请输入文件保存格式 (txt、json、csv): ')
        while self.fm!='txt' and self.fm!='json' and self.fm!='csv':
            self.fm = input('输入错误, 请重新输入文件保存格式 (txt、json、csv): ')

    def open_browser(self):
        self.browser = webdriver.Chrome()
        self.browser.implicitly_wait(10)
        self.wait = WebDriverWait(self.browser,10)

    def init_variable(self):
        self.data = zip()
        self.isLast = False

    def parse_page(self):
```

```

        try:
            skus =
self.wait.until(EC.presence_of_all_elements_located((By.XPATH, '//li[@cl
ass="gl-item"]'))))
            skus = [item.get_attribute('data-sku') for item in skus]
            links = ['https://item.jd.com/{sku}.html'.format(sku=item) for
item in skus]
            prices =
self.wait.until(EC.presence_of_all_elements_located((By.XPATH, '//div[@c
lass="gl-i-wrap"]/div[3]/strong/i'))))
            prices = [item.text for item in prices]
            names =
self.wait.until(EC.presence_of_all_elements_located((By.XPATH, '//div[@c
lass="gl-i-wrap"]/div[4]/a/em'))))
            names = [item.text for item in names]
            comments =
self.wait.until(EC.presence_of_all_elements_located((By.XPATH, '//div[@c
lass="gl-i-wrap"]/div[5]/strong'))))
            comments = [item.text for item in comments]
            img_urls =
self.wait.until(EC.presence_of_all_elements_located((By.XPATH, '//div[@c
lass="gl-i-wrap"]/div[1]/a/img'))))
            img_urls = [item.get_attribute('src') for item in img_urls]
            self.data = zip(links,prices,names,comments,img_urls)
        except selenium.common.exceptions.TimeoutException:
            print('parse_page: TimeoutException1')
            self.parse_page()
        except selenium.common.exceptions.StaleElementReferenceException:
            print('parse_page: StaleElementReferenceException')
            self.browser.refresh()

    def turn_page(self):
        try:

self.wait.until(EC.element_to_be_clickable((By.XPATH, '//a[@class="pn-ne
xt"]'))).click()
            time.sleep(1)

self.browser.execute_script("window.scrollTo(0,document.body.scrollHeig
ht)")
            time.sleep(2)
        except selenium.common.exceptions.NoSuchElementException:
            self.isLast = True
        except selenium.common.exceptions.TimeoutException:

```

```

        print('turn_page: TimeoutException2')
        self.turn_page()
    except selenium.common.exceptions.StaleElementReferenceException:
        print('turn_page: StaleElementReferenceException')
        self.browser.refresh()

def write_to_file(self):
    if self.fm == 'txt':
        for item in self.data:

self.fd.write('-----\n')
        self.fd.write('link: ' + str(item[0]) + '\n')
        self.fd.write('price: ' + str(item[1]) + '\n')
        self.fd.write('name: ' + str(item[2]) + '\n')
        self.fd.write('comment: ' + str(item[3]) + '\n')
    if self.fm == 'json':
        temp = ('link','price','name','comment')
        for item in self.data:

json.dump(dict(zip(temp,item)),self.fd,ensure_ascii=False)
    if self.fm == 'csv':
        writer = csv.writer(self.fd)
        for item in self.data:
            writer.writerow(item)
def write_to_mysql(self):
    db=pymysql.connect("localhost","root","123456","test")
    cursor=db.cursor()
    sql="INSERT INTO testmodel_phone(link,price,name,comment,img_url)
VALUES (%s,%s,%s,%s,%s)"
    for item in self.data:
        for item in self.data:

cursor.execute(sql,(item[0],item[1],item[2],item[3],item[4]))
        db.commit()
        db.close()

def close_file(self):
    self.fd.close()

def close_browser(self):
    self.browser.quit()

def crawl(self):
    #self.open_file()

```

```

        self.open_browser()
        self.init_variable()
        db=pymysql.connect("localhost","root","123456","test")
        cursor=db.cursor()
        sql="truncate table testmodel_phone"
        cursor.execute(sql)
        db.commit()
        db.close()
        print('开始爬取')

self.browser.get('https://search.jd.com/Search?keyword=%E6%89%8B%E6%9C%
BA&enc=utf-8')
        time.sleep(1)

self.browser.execute_script("window.scrollTo(0,document.body.scrollHeight)")
        time.sleep(2)
        count = 0
        while count!=2:
            count += 1
            print('正在爬取第 ' + str(count) + ' 页.....')
            self.parse_page()
            self.write_to_mysql()
            self.turn_page()
            #self.close_file()
            self.close_browser()
            print('结束爬取')

if __name__ == '__main__':
    spider = JdSpider()
    spider.crawl()

```

## 12306 的自动登录（通过验证码）

```

class Demo():
    def __init__(self):
        self.coordinate=[[ -105,-20],[ -35,-20],[40,-20],[110,-20],[ -105,
50],[ -35,50],[40,50],[110,50]]
    def login(self):
        login_url="https://kyfw.12306.cn/otn/resources/login.html"
        driver = webdriver.Chrome()

```

```

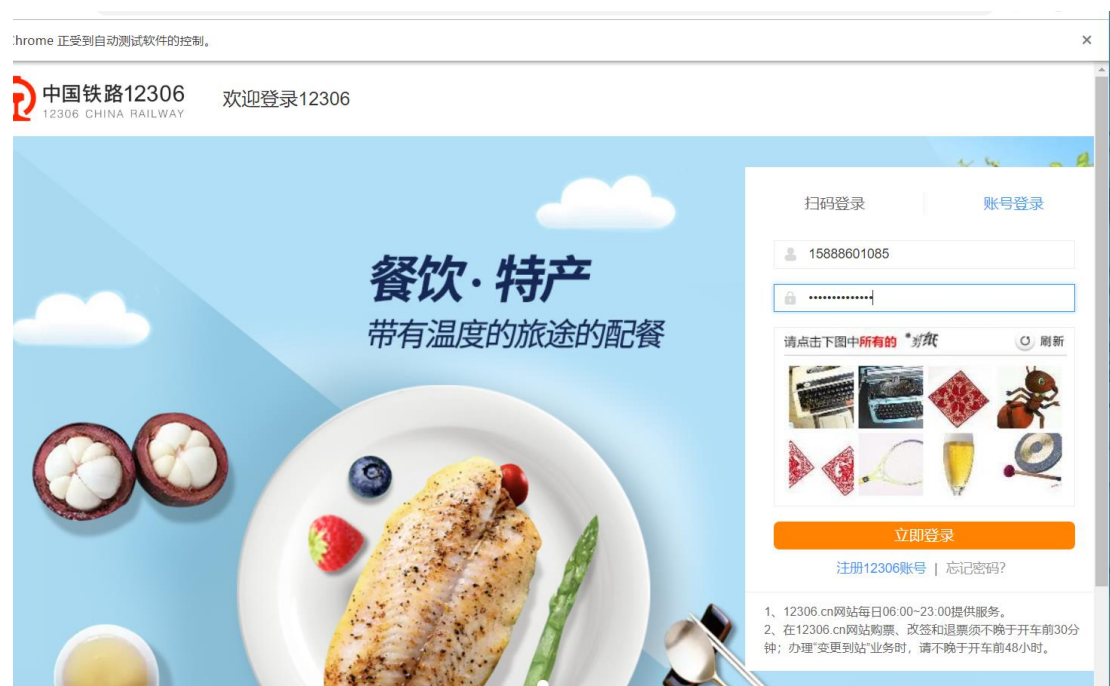
driver.set_window_size(1200, 900)
driver.get(login_url)
time.sleep(1)
account=driver.find_element_by_class_name("login-hd-account")
account.click()
userName=driver.find_element_by_id("J-userName")
userName.send_keys("15888601085")
password=driver.find_element_by_id("J-password")
password.send_keys("ljh13127984971")
self.driver=driver
def getVerifyImage(self):
    try:
        img_element =WebDriverWait(self.driver, 100).until(
            EC.presence_of_element_located((By.ID, "J-loginImg"))
        )
    except Exception :
        print(u"网络开小差,请稍后尝试")
    base64_str=img_element.get_attribute("src").split(",")[-1]
    imgdata=base64.b64decode(base64_str)
    with open('d:\\verify.jpg','wb') as file:
        file.write(imgdata)
    self.img_element=img_element
def getVerifyResult(self):
    driver1 = webdriver.Chrome()
    driver1.get('http://littlebigluo.qicp.net:47720/')
    upload = driver1.find_elements_by_tag_name('input')[0]
    time.sleep(3)
    upload.send_keys('d:\\verify.jpg') # send_keys
    submit = driver1.find_elements_by_tag_name('input')[1].click()
    response=driver1.find_element_by_xpath("/html/body/p[1]/font/fo
nt/b").text
    result=[]
    for i in response.split(" "):
        result.append(int(i)-1)
    self.result=result
    driver1.close
    print(result)
def moveAndClick(self):
    try:
        Action=ActionChains(self.driver)
        for i in self.result:
            Action.move_to_element(self.img_element).move_by_offset
(self.coordinate[i][0],self.coordinate[i][1]).click()
            Action.perform()

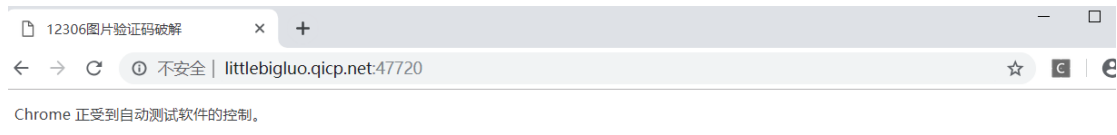
```

```
except Exception as e:
    print(e)
def submit(self):
    self.driver.find_element_by_id("J-login").click()
def __call__(self):
    self.login()
    time.sleep(3)
    self.getVerifyImage()
    time.sleep(1)
    self.getVerifyResult()
    time.sleep(1)
    self.moveAndClick()
    time.sleep(1)
    self.submit()

Demo()()
```

## 演示





## 请上传一张12306验证码图片

未选择任何文件

使用方法:

- 1-打开12306网站登录界面: [点击这里打开12306](#)
- 2-点击12306页面顶部**登录**按钮, 然后点击**账号登陆**, 鼠标右键点击**页面中间验证码图片**
- 3-选择**图片另存为**保存验证码图片, 并重命名以.jpg结尾
- 4-然后点击本页面**选择文件**按钮选择刚刚保存的图片
- 5-然后点击本页面**上传**按钮查看结果

上传非标准12306图片验证码文件, 本系统会拒绝连接

本破解基于深度学习算法实现:[点击这里查看详情](#)

有意见或建议? ? 欢迎交流:3490699170@qq.com



## 请上传一张12306验证码图片

未选择任何文件

请点击下图中**所有的** **\*剪纸**



经过仔细揣摩-图片貌似选: **3 5**

第一排图片从左到右编号依次为:1 2 3 4

第二排图片从左到右编号依次为:5 6 7 8

耗时:596毫秒!觉得俺bigluo给力如何??

如果不确定结果是否正确, 不妨登陆一下12306试试! !

有意见或建议? ? 欢迎交流:3490699170@qq.com或者[点击这里](#)

