Todo

1.直接把源代码工程运行看看。 Python2的版本，py-3无法直接运行。

记录这套书/视频/文章的学习过程（知识点、error思考等），方便后续结合code总结回归

\*\自动化测试\6Python接口测试框架实战与自动化进阶\

# Django\_imooc工程

## 3-1 开发接口环境搭建

**创建项目方式一：pycharm**

**Create newproject , Django , application name =web**

settings.py # 系统文件，数据库 工程等

urls.py #告诉Django应创建哪些网页来响应浏览器请求

创建项目方式二：

pip install django  
import django  
print(django.get\_version()) #  
在Python根目录下的Scripts目录中，有“ django-admin.exe ”  
django-admin startproject mysite  
python manage.py runserver  
浏览器访问

方式1：python manage.py runserver 127.0.0.1:8000

方式2：pycharm中manage.py配置runserver 127.0.0.1:8000

再去浏览器访问

## 3-2 django之接口工作原理

Django的应用程序 叫app，可以有很多个。

方式1：pycharm专业版中新建时，在more-application name中创建。

方式2：**python manage.py startapp api**

app的名字=api

一、在settings中添加我们的应用程序（app），不然无法解析

INSTALLED\_APPS

'web'

**二、Django\_imooc/ Django\_imooc/urls.py**

地址映射login方法

from web.views import Login  
urlpatterns = [  
url(r'^admin/', admin.site.urls),  
url(r'^login',Login)  
]

三、login方法写在views中

**Django\_imooc/web/views.py**

from django.http.response import HttpResponse

# Create your views here.  
def Login(request):  
 return HttpResponse(‘this is first test’)

**整体原理：接收到地址，转发，返回**

## 3-3 django之post接口开发

**url 下加：路由中添加地址和对应调用的函数**

from web.views import Login  
  
urlpatterns = [  
 url(r'^admin/', admin.site.urls),  
 url(r'^login', Login)  
]

**views中加：添加对应的方法**

from django.http.response import HttpResponse

from django.shortcuts import render\_to\_response

# Create your views here.  
def Login(request):  
if request.method == "POST":  
 username =request.POST.get('username')  
return HttpResponse(username)  
else:  
return render\_to\_response('login.html')

**Django\_imooc/ templates下添加login.html**

添加表单form

<form action="/login/" method="POST">  
<h1>用户名：<input name="username"></h1>  
<h1>密码：<input name="password"></h1>  
<input type="submit" value="登录">  
</form>

login.html调用login方法（login方法在views.py中实现），method=post

添加TEMPLATES路径

'DIRS': [os.path.join(BASE\_DIR, 'templates')]

**Django\_imooc/ Django\_imooc/settings.py**

注释掉安全拦截

MIDDLEWARE = [  
# 'django.middleware.csrf.CsrfViewMiddleware',

访问<http://127.0.0.1:8000/login/>

# DjangoInterface工程

## 第4章 requests库的相关使用

## 4-2 requests简单使用-post

requests 是一个http library库

request.post的参数默认是字典

**见base\demo4\_2.py**

res.text的类型是 <class 'str'>

res.json()的类型是 <class ‘list’>

## 4-3 重构发送post请求

封装了 post请求（写成方法）

**demo.py**

## 4-4 重构get请求+格式化响应数据

把post换成了get ，

1、get时省略data参数，url后直接加？参数（一般不用）

2、get时加data参数，url后的？参数省略

def send\_get(url, data):  
 res = requests.get(url=url, data=data).json()  
 return json.dumps(res, indent=2, sort\_keys=True)

用request库，构造两个简单的接口

**demo.py**

## 4-5 使用类封装接口测试脚本

添加run\_main方法用于判断是post还是get，优化入参data=none，优化res=none，return

添加RunMain：类和if name =main，优化初始化：def init

**demo.py (4-5 使用类封装接口测试脚本).txt**

\*\000\00CodeHub0203\2PythonCode\DjangoInterface\base\

## 第5章

## 5-2 unittest和request重构封装

创建包 base 等

**test\_method.py中导入demo.py和加载执行**

**5-2 unittest和request重构封装.mp4 ，第9分钟运行时报错。**

**TypeError: \_\_init\_\_() takes at least 3 arguments**

**5-2 unittest和request重构封装.mp4**

**31s-40S ，发现demo.py init方法被删了(3分26秒也能发现)**

## 5-6 unittest之常见面试解答及知识回顾

**初中级面试问题**

**1、使用python开发测试框架**

语言：python

接口开发：requests包

case管理：unittest

测试报告：htmltestrunner.py，放lib下

测试数据：excel，（mysql数据库）

持续集成：jenkins

**2、如何管理case：**

Unittest.Testsuite

**3、case有依赖：**

globals()全局变量来使用，要用的中间值存在数据库里

**4、case写在哪里：**

写在excel，数据库

**5、简述case的执行：**

case字母的先后顺序执行

unittest.skip跳过

**do 练习**

## 第6章 mock服务入门到实战

Mock，是服务端还没开发完成。

模拟接口的返回，类似于fiddler的autoResponse

Mock是Python中一个用于支持单元测试的库，它的主要功能是使用mock对象替代掉指定的Python对象，以达到模拟对象的行为。

从Python 3.3开始，mock模块已经被合并到标准库中，被命名为**unittest.mock**

from unittest import mock

## 7-1 如何设计一个接口自动化测试框架

**需要思考：**

接口地址，

请求数据，

接口类型，get post

预期结果，

header，用于保存cookies

数据依赖，接口自动化的难点

## 7-6 封装获取excel字段的常量方法

把excel中的case的列值，封装到一个data\_config.py文件中。

方便做数据分离，使获取数据时调用。

## 7-7 封装获取excel的数据

封装GetData类，

初始化时，实例化opera\_excel对象

def \_\_init\_\_(self):  
 self.opera\_excel = OperationExcel()

调用对象的get\_cell\_value方法，入参为row和col

其中row是每个case的行号，col为需要获取的列（通过data\_config中的get\_\*方法获取）

## 7-9 主流程封装及错误解决调试

跳过，视频里当场解决了一些bug

## 7-10 返回数据格式处理以及调错.mp4

**！！！放弃这系列视频，前一章节结尾时的视频里的代码，和后一章刚开始的，明显有修改。**

鬼知道改了什么东西。

**后面的内容是主要听一听思想即可。理解思想。**

7-11 获取接口返回状态.mp4

7-12 通过预期结果判断case是否执行成功.mp4

7-13 将测试结果写入到excel中.mp4

## 7-14 数据依赖问题从设计思路开始.mp4

讲了思路，把上一个response的货物ID存在变量中，赋值给下一个接口调用的request的数据中。

7-15 数据依赖问题方法封装之通过case\_id获取case数据.mp4

7-16 数据依赖问题之根据规则提取响应数据.mp4

7-17 数据页面相关.mp4

7-18 数据依赖问题之依赖结构构建.mp4

7-19 数据依赖问题之流程实施.mp4

7-2 学习python操作excel获得内容.mp4

7-20 case运行结果统计.mp4

7-21 构建发送邮件服务.mp4

## 7-22 结果统计+报告通知.mp4

理解框架的思路后，自己写一遍，去要高薪。

## 8

8-1 持续集成环境搭建.mp4

8-2 持续集成之项目配置.mp4

## 9

9-1 获取cookie思路分析.mp4

9-2 模拟登录获取cookie请求订单接口.mp4

登录时调用的接口是：<https://www.imooc.com/passport/user/login>

9-3 重构封装携带cookie请求处理流程.mp4

## 10

10-1 连接数据库查询数据.mp4

10-2 获取数据库数据重构及转换数据.mp4

10-3 返回数据和数据库数据进行对比.mp4

10-4 格式化数据对结果进行回写.mp4

## 11

第11章1.2.3节.mp4