全程66互联网-接口自动化第一次

### 内容1： 接口自动化 灵魂三问：

**什么是接口自动化？**

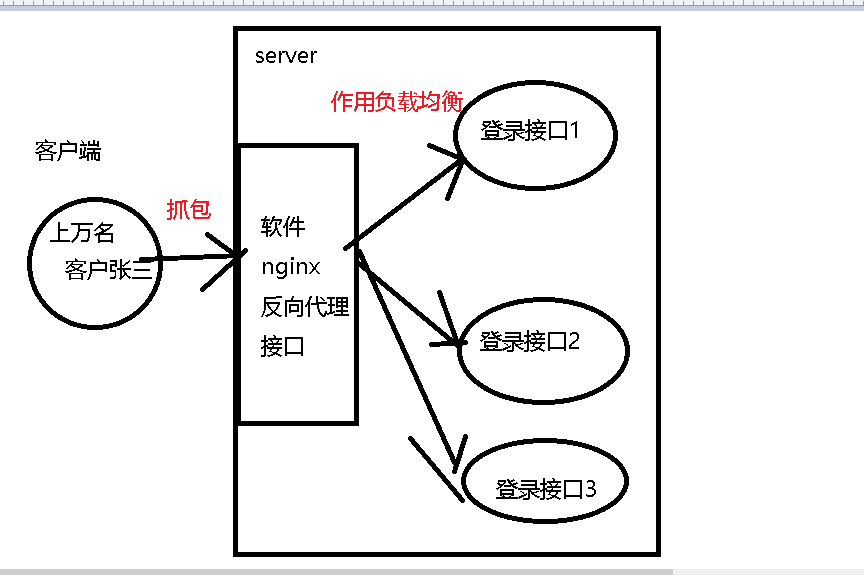
回顾：

1 接口（JAVA程序员） 才是一个 系统真正功能的体现 所有人页面都是一样的 只是接口数据不一致~

2 接口测试

2.1 第一步：拿到接口文档 -----如果没有接口文档怎么办？ 答：虽然可以抓包，但是不建议的

2.1.1 能抓到 抓字段含义 看不懂~  
 2.1.2 抓不到



接口文档 --------开发经理 架构师 -----》前后端分离开发标准~ swagger 在线接口文档 方便交流

2.2 接口测试

测试工具： 1postman 2jmeter xmeter 3 fiddler ----》大厂 互联网 100%自研工具 TW

填入接口文档 四要素 （url+method+headers +body）即可访问===》

优点： 简单

测试代码： Python （接口自动化 以代码方式 实现 接口测试）

缺点： 编写麻烦 +调试 要求高~

优点： 一次编写 可以到处运行~

**为什么接口自动化？**

需求： 互联网企业 微服务 大接口----》小接口 大量的接口 \*10\*10（巨大工作量~执行用例 ）

解决：

分工

1 测试人员维护 写用例 ---用例数据库 ----》接口用例 随着开发越来越多~

2 测试开发 写一套Python 读取用例库发出接口请求 +测试

其他优点：

1 提升执行用例效率~

2 代码扩展 2.1 代码出个漂亮报告 allure 2.2 整合 日志观察用例

3 分工

**学员角度：**

**1和测试卷**

**2加强Python编程适应企业方式**

**3 初级测试开发**

**如何掌握接口自动化（剧情）？**

**前四次 （库使用基础）**

**1 Requests ---》Python 发出接口访问请求的~~**

**2 Pytest ----》数据驱动 组织测试用例**

**3 logging -----》日志库**

**4 Pymysql ----》Python 读取 mysql中的数据的~（Excel---》mysql 测试用例 ）**

**Python 语法 加强 ----》企业编程方式： 1API表 2 工具类封装 （代码不是一行行 ，主要会复用）**

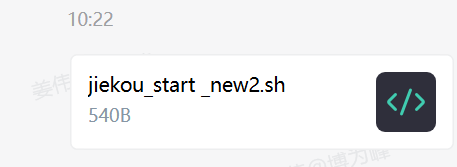
**后四次：**

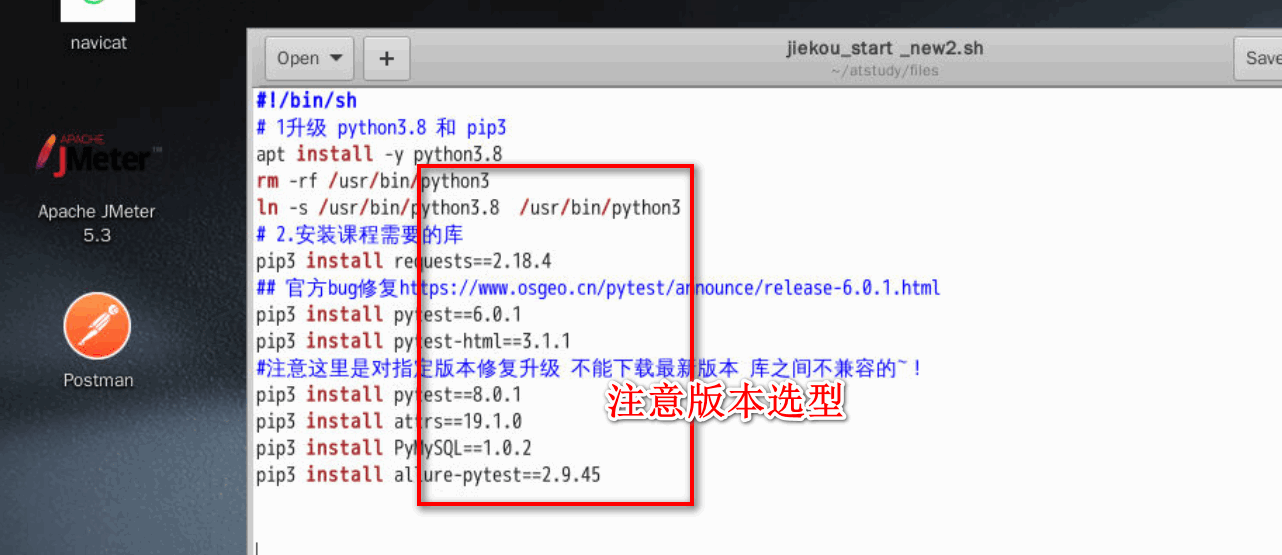
**综合运用~ 写一套自定义框架 加载用例执行 ~目标： 无人值守（git+jenkins） 产出报告（allure）**

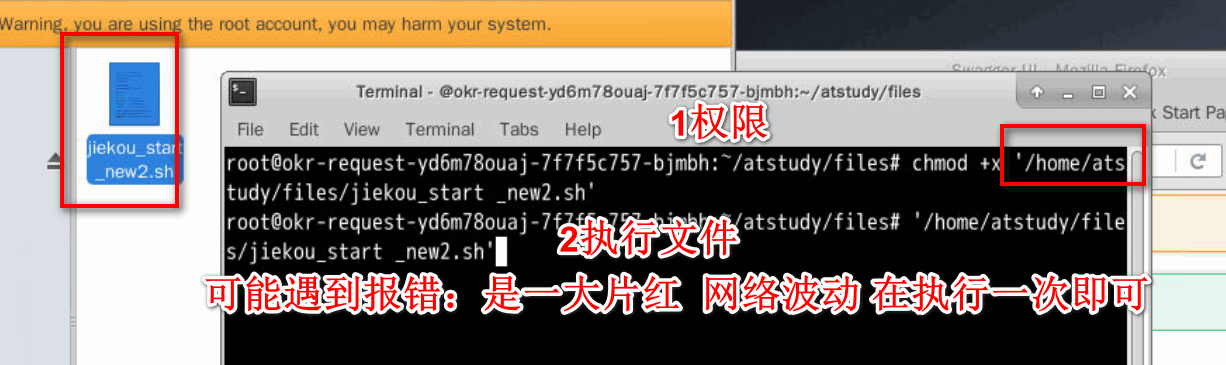
### 内容2： 环境：

1 **被测系统 ----》OKR项目 （和工具形成对比）+接口文档**

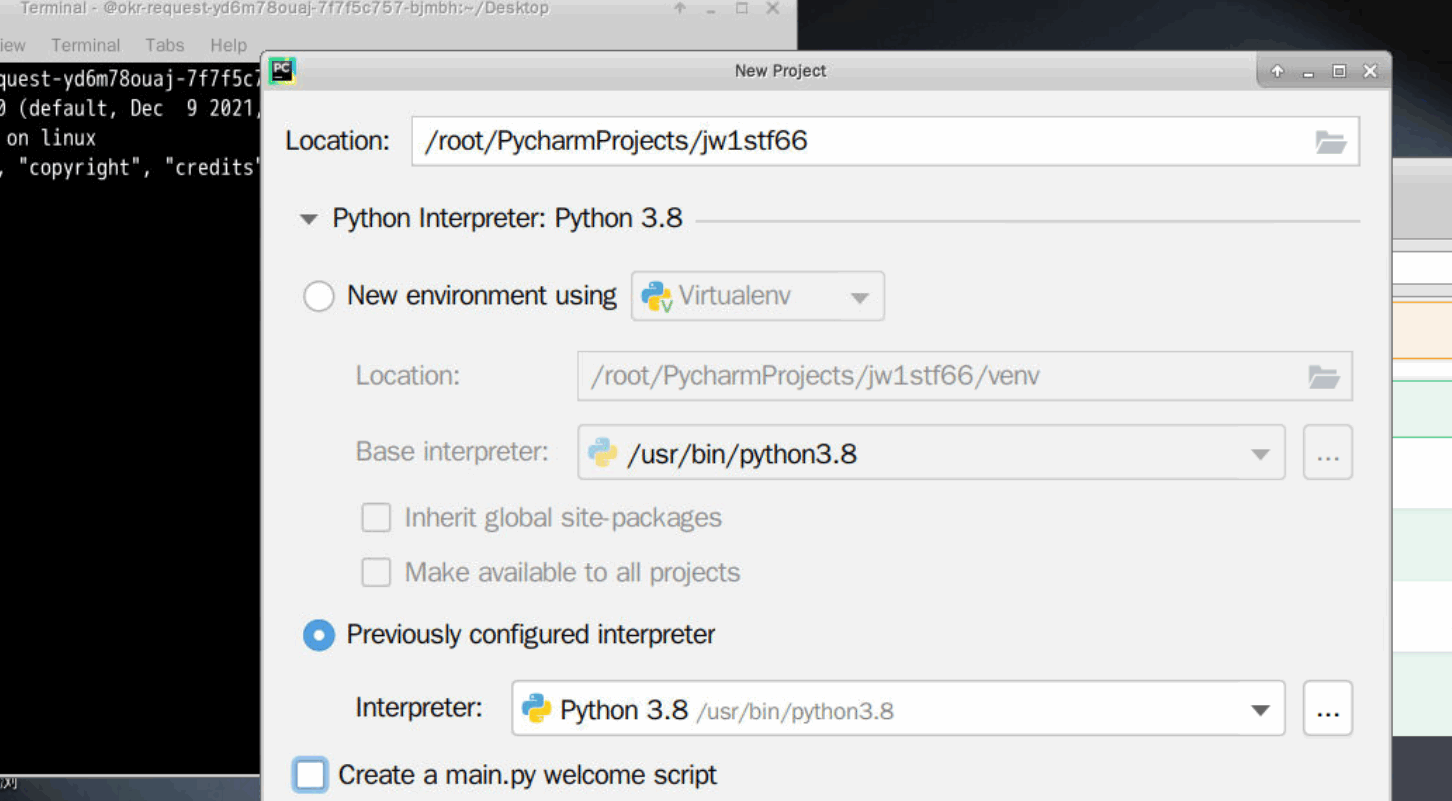
2 **开发环境 ----》python +pycharm +库 注意版本 pip===》shell脚本**





****

**验证:**

****

**验证代码**

|  |
| --- |
| **import requests**  **shiji=requests.get("http://www.baidu.com")**  **shiji.encoding="UTF-8" #中文处理**  **print(shiji.text) #响应体---来自于官网** |

### 内容3： requests库

**例子 ： 小白， requests 库 访问OKR-api上下游接口？**

**思路：**

**1**接口文档  **----》描述接口访问方式**

**2 requests ---》库 ----》官网权威教学~**

https://pypi.org/

https://requests.readthedocs.io/en/latest/api/ 优势： 权限 +卷 缺点:要求 英文

**需求： 事件紧，没有大量时间阅读 官网 +英文薄弱+遗忘？**

解决： 代码 常见 API （库提供的功能和方法） 整理表 ，编程 边参考API表 一边干活

|  |  |
| --- | --- |
| **库-对象名称** | 对象API功能方法 |
| **requests** | 发请求 方法：  1.1get （url=&拼接参数值,headers请求头）  1.2 post请求  1.2.1 发送请求体式字符串  post(url，headers请求头，data=请求体数据)  1.2.2 发送请求体json| 字典  post(url，headers请求头，json=请求体数据) |
|  | 响应结果：  encoding='utf-8'  2.1 状态码  status\_code  2.2 响应头  结果.headers  2.3 响应体  2.3.1 字符串响应体 text  2.3.2 json响应体 json() |

|  |
| --- |
| **#1 接口文档要素**  **import requests**  **# 运维服务器 + 接口地址**  **login\_url="http://localhost:6088/api/User"**  **login\_headers={'Content-Type':'application/json'}**  **login\_method='post'**  **login\_body={**  **"username": "admin" ,**  **"password": "123"}**  **# = 左边API表**  **# = 右边实际参数 来自于接口文档**  **shiji=requests.post(url=login\_url,headers=login\_headers,json=login\_body)**  **print(shiji.status\_code)**  **print(shiji.headers)**  **print(shiji.json())**  **# 下游代码---》一个值 环境变量～ 正则表达提取器 存 {{}} ${}引用**  **shouquanma=shiji.headers['Authorization']**  **print(shouquanma)**  **# 发出下游接口**  **# 运维服务器 + 接口地址**  **user\_url='http://localhost:6088/api/User?IsEnabled=true&Page=1'**  **url\_headers={'Authorization':shouquanma}**  **user\_method='get'**  **# = 左边API表**  **# = 右边实际参数 来自于接口文档**  **shiji=requests.get(url=user\_url,headers=url\_headers)**  **print(shiji.status\_code)**  **print(shiji.headers)**  **print(shiji.json())** |

小结：

1整理 带走 API表

2根据API表 +接口文档 能发出任意类型接口数据

预告 ：

1.降低代码风险

2 pytest 组织用例 （断言 assert）