**接口自动化框架使用说明书**

**Version： v0.0.1**

**Author: 张涛**

**Date: 2022年2月8日**

**目录**

[一、 编程语言 3](#_Toc18264)

[二、 测试框架 3](#_Toc13395)

[三、 请求管理 4](#_Toc15089)

[四、 数据驱动 4](#_Toc2262)

[五、 日志管理 5](#_Toc23192)

[六、 测试报告 6](#_Toc20725)

[七、 配置管理 7](#_Toc4386)

[八、 工具管理 8](#_Toc29759)

[1.邮件管理 8](#_Toc30640)

[2.钉钉消息 8](#_Toc24559)

[3.压缩文件 8](#_Toc9384)

[九、 持久化集成 9](#_Toc7821)

# 编程语言

本框架采用python语言。

# 测试框架

基于使用pytest测试框架进行用例执行。

Unittest与[Pytest](https://so.csdn.net/so/search?q=Pytest&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/liudinglong1989/article/details/105986621/_blank)主要从用例编写规则、用例的前置和后置、参数化、断言、用例执行、失败重运行和报告这几个方面比较unittest和pytest的区别:

|  | **unittest** | **pytest** |
| --- | --- | --- |
| 用例编写规则 | 1)测试文件必须先import unittest  2)测试类必须继承unittest.TestCase  3)测试方法必须以“test\_”开头  4)测试类必须要有unittest.main()方法 | 1)测试文件名必须以“test\_”开头或者"\_test"结尾（如：test\_ab.py）  2)测试方法必须以“test\_”开头  3)测试类命名以"Test"开头 |
| 用例分类执行 | 默认执行全部用例，也可以通过加载testsuit，执行部分用例 | 可以通过@pytest.mark来标记类和方法，pytest.main加入参数("-m")可以只运行标记的类和方法 |
| 用例前置和后置 | 提供了setUp/tearDown，只能针对所有用例 | pytest中的fixture显然更加灵活。可以任意自定义方法函数，只要加上@pytest.fixture()这个装饰器，那么被装饰的方法就可以被使用 |
| 参数化 | 需依赖ddt库 | 使用@pytest.mark.parametrize装饰器 |
| 断言 | 很多断言格式(assertEqual、assertIn、assertTrue、assertFalse) | 只有assert一个表达式，用起来比较方便 |
| 报告 | 使用HTMLTestRunnerNew库 | 有pytest-HTML、allure插件 |
| 失败重跑 | 无此功能 | pytest支持用例执行失败重跑，pytest-rerunfailures插件 |

总结：

总体来说，unittest用例格式复杂，兼容性无，插件少，二次开发方便。pytest更加方便快捷，用例格式简单，可以执行unittest风格的测试用例，无须修改unittest用例的任何代码，有较好的兼容性。pytest插件丰富，比如flask插件，可用于用例出错重跑，还有xdist插件，可用于设备并行执行，效率更高。

# 请求管理

二次封装requests模块对http协议发送GET、POST、PUT、DELETE等请求类型。

requests库7个主要方法

| **方法** | **说明** |
| --- | --- |
| requsts.requst() | 构造一个请求，最基本的方法，是下面方法的支撑 |
| requsts.get() | 获取网页，对应HTTP中的GET方法 |
| requsts.post() | 向网页提交信息，对应HTTP中的POST方法 |
| requsts.head() | 获取html网页的头信息，对应HTTP中的HEAD方法 |
| requsts.put() | 向html提交put方法，对应HTTP中的PUT方法 |
| requsts.patch() | 向html网页提交局部请求修改的的请求，对应HTTP中的PATCH方法 |
| requsts.delete() | 向html提交删除请求，对应HTTP中的DELETE方法 |

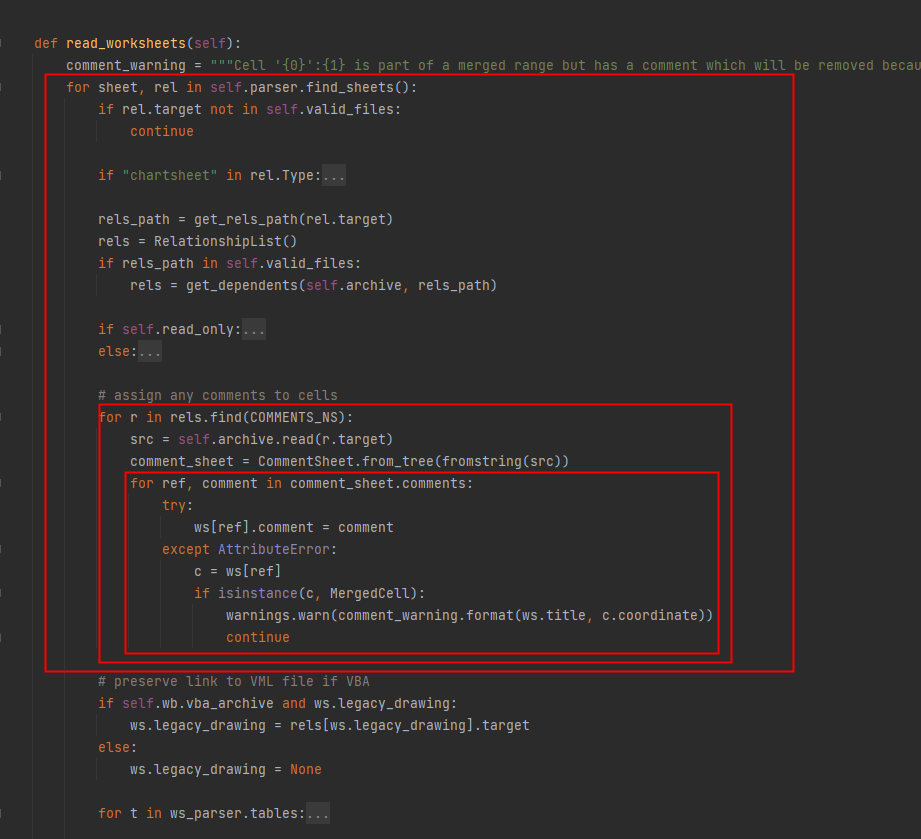
# 数据驱动

本框架采用excel数据驱动来管理测试用例。二次封装xlrd、xlwt对excel更简单的进行遍历读取、写入、字体及背景格式化。

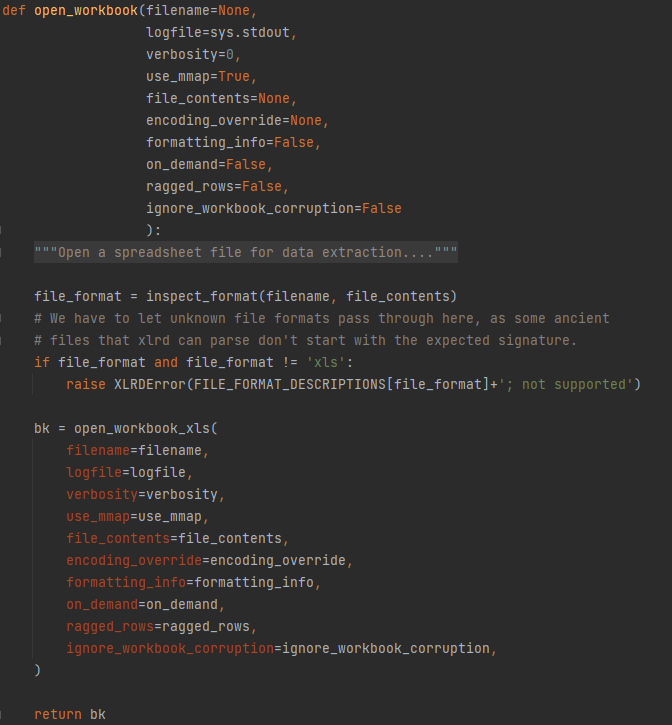
python读取excel几种方式的对比

| **模块** | **说明** |
| --- | --- |
| pandas | 如果你的excel表格需要做缺失值，合并单元格，筛选以及对某个值进行替换，但是对表格格式没有要求，首选pandas，文件的打开速度和openpyxl差不多。 |
| openpyxl | 它的功能是最全面的，可以直接再原文件上修改保存，保留原文件的格式，但是它读取文件的速度较慢，这个可以看看openpyxl打开excel源码，嵌套多层循环。总体来说，如果你需要直接在原文件上修改数据，可以选择openpyxl，如果要做筛选，可以结合pandas一块使用。 |
| xlrd/xlutils/xlwt | xlrd读取数据是最快的，如果你只是读取数据的话，首选xlrd。修改文件的时候需要用到xlwt，但是它不能直接使用，需要通过xlutils复制一份，然后再用xlwt保存。 |
| xlwings | 暂时没有实践 |

penpyxl打开excel源码，嵌套多层循环，如图所示：



xlrd打开xecel源码，如图所示:



# 日志管理

本框架进行二次封装logging模块进行日志管理，logging模块定义的函数和类为应用程序和库的开发实现了一个灵活的事件日志系统。logging模块是Python的一个标准库模块，由标准库模块提供日志记录API的关键好处是所有Python模块都可以使用这个日志记录功能。所以，应用日志可以将你自己的日志信息与来自第三方模块的信息整合起来。

logging模块默认定义了以下几个日志等级，它允许开发人员自定义其他日志级别，但是这是不被推荐的，尤其是在开发供别人使用的库时，因为这会导致日志级别的混乱。

| **日志（level）** | **描述** |
| --- | --- |
| DEBUG | 最详细的日志信息，典型应用场景是 问题诊断 |
| INFO | 信息详细程度仅次于DEBUG，通常只记录关键节点信息，用于确认一切都是按照我们预期的那样进行工作 |
| WARNING | 当某些不期望的事情发生时记录的信息（如，磁盘可用空间较低），但是此时应用程序还是正常运行的 |
| ERROR | 由于一个更严重的问题导致某些功能不能正常运行时记录的信息 |
| CRITICAL | 当发生严重错误，导致应用程序不能继续运行时记录的信息 |

**说明：**

上面列表中的日志等级是从上到下依次升高的，即：DEBUG < INFO < WARNING < ERROR < CRITICAL，而日志的信息量是依次减少的；

当为某个应用程序指定一个日志级别后，应用程序会记录所有日志级别大于或等于指定日志级别的日志信息，而不是仅仅记录指定级别的日志信息，nginx、php等应用程序以及这里要提高的python的logging模块都是这样的。同样，logging模块也可以指定日志记录器的日志级别，只有级别大于或等于该指定日志级别的日志记录才会被输出，小于该等级的日志记录将会被丢弃。

# 测试报告

本框架测试报告采用allure生成测试报告，allure源码采java编写，所以运行需要本次安装jdk(1.8版本之后含1.8).pytest中插件中有allure-pytest该插件可以生成测试可视化测试报告。

**allure用例描述:**

| **使用方法** | **参数值** | **参数说明** |
| --- | --- | --- |
| @allure.epic() | epic描述 | 敏捷里面的概念，定义史诗，往下是feature |
| @allure.feature() | 模块名称 | 功能点的描述，往下是story |
| @allure.story() | 用户故事 | 用户故事，往下是title |
| @allure.title(用例的标题) | 用例的标题 | 重命名html报告名称 |
| @allure.testcase() | 测试用例的链接地址 | 对应功能测试用例系统里面的case |
| @allure.issue() | 缺陷 | 对应缺陷管理系统里面的链接 |
| @allure.description() | 用例描述 | 测试用例的描述 |
| @allure.step() | 操作步骤 | 测试用例的步骤 |
| @allure.severity() | 用例等级 | blocker，critical，normal，minor，trivial |
| @allure.link() | 链接 | 定义一个链接，在测试报告展现 |
| @allure.attachment() | 附件 | 报告添加附件 |

报告如图所示;

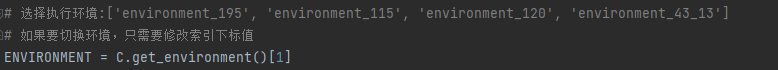


# 配置管理

本框架资源配置采用yaml文件作为配置文件，二次封装PyYAML，读取配置文件，将环境环境以对象的形式存于列表，如果需要某个环境，只需要编列列表，取到environment对应的字段即可拿到参数。

测试人员只需要关注执行哪个环境即可。

如图所示：



# 工具管理

## 1.邮件管理

二次封装email模块，自定义主题内容，并发送测试报告的压缩文件给多个人。

## 2.钉钉消息

基于dingtalkchatbot封装钉钉发送消息，并读取测试结果，发送测试结果给测试人员。如图所示;



## 3.压缩文件

二次封装zipfile进行测试报告压缩。

全部功能采用插拔式，可以自行配置选择是否开启功能。如图所示：



# 持久化集成

Jenkins配置方案;在jenkins上创建新的项目

第一步：填写项目名称

第二步：配置源码管理，选择git,添加项目Gitlib上仓库地址

第三步: 构建输入执行命令：python main.py

第四步：点击应用，然后保存

以后每次构建都将拉取最新代码并执行。然后生成allure报告