自动化设计规范

- 讲师：Pansir

每个公司都有自己的统一规范，不通框架也有不通的管理规范，这里提供设计思路供参考

目录结构

目录结构规范：

1. 项目名称(根据服务划分)：project name，如：`test_tsms_project`
2. 子项目名称(根据子服务划分)：child project name，如：`test_tsms_api`，`test_tsms_mq`，`test_tsms_pile`，`test_tsms_kfk` 等
3. 接口分类目录(根据api或划分)：api name，function name，
   如：`test_tsms_send_api`，`test_tsms_temp_api`，`test_tsms_mq_produce`，`test_tsms_kfk_consume`
4. 接口/功能点细分(具体某个module即`*.py`文件)：case file name，
   如：`test_tsms_temp_api_create.py`，`test_tsms_temp_api_update.py`，`test_tsms_send_api_single.py`，`test_tsms_mq_consume_batch`(批量消费)
5. 类名命名 Test+文件名描述：例如：`TestTsmsSendApiUpdate`，`TestTsmsMqConsumeBathch`
6. 用例(函数/方法)名称：test_ + 功能 + 数字，例
   如：`test_send_api_single_01`，`test_temp_api_update_02`，`test_mq_consume_batch_10`

补充说明：目录分层，大致参照这个思路。如果模块细分，或版本细分，按分类逻辑加入即可，符合树形展开逻辑就行

命名规范

1. 模块名`*.py`：服务名+测试点，如：`test_tsms_send_api_signle.py`，表示：tsms模块，send_api 接口，signle单推功能
2. 测试用例类名`class xxx()`：和模块名一致：`class TestTsmsSendApiSignle()`，同上，但注意不使用下划线，大小写要明显
3. 用例名：`def xxx()`：test_ + 功能点(取至类名) + 编号，如：`test_send_api_signle_01`，表示第一条用例
4. 用例标记：`@pytest.mark.xxx`，每个用例都需要一个标记，标记规则为：
   - 可以借鉴bug级别：`fatal` 致命，`critical` 严重，`normal` 一般，`trivial` 轻微，`Enhancement`建议。但是一般只用前3种：critical/normal/travial
   - 可以自定义级别：`level1`，`level2`，`level3`，等等，具体要划分多少个级别，根据实际情况来定，但是一般分三级管理相对方便

参考优先级：

- 事物处理优先级：紧急/严重/重要/次要/微小

- bug级别：致命/严重/一般/轻微/建议

用例编写规范

1. 每个用例最好带上注释，说明测试点
2. 用例中的一些关键步骤和数据要有log日志
3. 尽量保证用例简洁，封装完善的前置与后置动作

```
```