单元测试规范

- 测试代码必须放在`src/test/groovy`目录下
- 测试类的命名如下：
  - 被测试的接口+Spec
  - 被测试的类+Spec
- 测试类和被测试类必须在相同包名下
- 测试用例的命名为:测试方法名+`_`+用例简单描述,例如:
  - `insertAdminLog_调用方法会将数据存储到数据库中`
  - `insertAdminLog_调用方法会将数据存储到数据库中，如果Description长度大于500将截断`
- 测试粒度
  - 简单的数据源(数据库、MQ、缓存)操作不必测试
  - 操作数据源前后有业务逻辑操作必须测试
  - setter/getter方法不必测试
  - 构造方法中有业务逻辑操作必须测试
  - 被测试类中所有除上述以外的方法必须测试
  - 每个被测试方法至少一个测试用例(非特殊情况一个测试用例中不能覆盖多个方法)
- 每个被测试方法的代码覆盖率和路径覆盖率都必须达到100%
- 每个测试用例必须遵循测试的FIRST原则
  - `F(Fast)`：测试要能快速运行
  - `I(Isolate)`：测试用例要独立，不能相互依赖
  - `R(Repeatable)`：测试要可以重复运行且不能受到外界环境（网络、服务、中间件等）的影响
  - `S(Self-verifying)`：测试会自己检查产出，不准使用人肉验证(System.out或logger等)，必须使用assert来验证
  - `T(Timely)`：测试要及时做，与写代码紧密相连
- 每个测试用例必须遵守BCDE原则，以保证被测试模块的交付质量。
  - `B(Border)`:边界值测试，包括循环边界、特殊取值、特殊时间点、数据顺序等
  - `C(Correct)`:正确的输入，并得到预期的结果
  - `D(Design)`:与设计文档相结合，来编写单元测试
  - `E(Error)`:强制错误信息输入（如：非法数据、异常流程、非业务允许输入等），并得到预期的结果
- 每个测试用例必须遵循`Arrange-Act-Assert`(3A)模式
  - `Arrange`：（设置测试数据、变量、环境等）
  - `Act`：（调用要测试的函数、代码）
  - `Assert`：（验证输出是否是预期的结果）
- 测试用例名称中不能介绍清楚用例功能时候需要在用例中写详细的用例描述
- 测试用例中必须写维护人员。如下代码：

```
def "calculateData_测试计算数据过程中抛出异常的情况"(){
  reportInfo "当前用例测试抛出IO异常时的处理方案"
  reportInfo "当前用例测试发生超时的处理方案"
  reportInfo "维护者 -- 小明 小黑 小黄"
}
```

- 对于不可测的代码建议做必要的重构，使代码变得可测，避免为了达到测试要求而书写不规范的测试代码