# 自动化测试框架重构

1. **测试的功能性区分**
   1. CI测试：在PR时进行的针对编译是否存在错误进行的测试。但不能校验实际使用时的正确与否
   2. **自动化测试**：编辑器打包之后对于编译出来的应用进行用例测试。编译出来的程序启动到场景开始运行分别经过若干步骤：
      1. 应用启动到各个平台的运行，在**场景加载之前**进行
      2. **场景加载时**
      3. **场景加载之后**对于各个场景的切换
         1. 功能测试：
            1. 单元性测试（区分于测试API用的单元测试）
            2. 按钮测试
         2. 性能测试：可能项
2. **功能测试的具体需求划分**
   1. 单元性测试
      1. 静态场景测试
      2. 动画场景测试
      3. 粒子效果测试
      4. 物理引擎效果测试
   2. 交互测试
      1. 类slider非精准性点击测试
      2. button精准点击测试
      3. 场景移动测试
      4. 镜头切换测试
3. **针对已知需求的测试框架**
   1. 单元性测试
      1. 使用截图比对
         1. 当前方法：通过websocket机制发送截图命令和接受截图完成的命令，后切换场景。或运行当前场景至指定帧
            1. **优点**：能够保证截图是从外部截图，获取的是当前平台的人眼可见的图像效果
            2. **缺点**：时间太久，websocket通信机制需要时间，截图需要时间，随着测试例的增多会有效率降低的问题。
         2. 未来可能方法：转为快速截图的方式，截图由客户端完成并发送给服务端。最后进行统一比对，取决于截图由客户端执行还是服务端执行。当前方法就是服务端执行截图并执行比较。
            1. 优点：快
            2. 缺点：不能保证截图和肉眼效果相同。不能保证每个平台都有权限截图。影响性能
      2. 状态比对
         1. 未来可能方法：由于程序内状态只能由程序自己获取，所以需要在客户端进行状态断言。通过**引入断言库的方法进行状态真假比对并向服务端发送断言结果**
   2. 交互测试
      1. 计划内：按钮测试，内部赋予变量值的类slider测试
      2. 计划外：场景移动测试（如滑动屏幕观察场景，是较为随机的测试例）
4. **针对截图比对的改进优先级：**
   1. 测试例调整，针对不同场景需要有不同的操作队列，一个操作完成后，等待截图再进行下个操作，该功能需要在截图比对完成之后在同一个场景中切换到不同的状态，从而实现同一个场景的多角度比较
      1. 需要添加或重新塑造stateCode，以及消息收发机制
      2. 需要对测试用例的代码进行更新，暴露出可控的trigger，来提供给框架使用并控制
      3. **需要优先重构框架**
   2. 动画测试：针对持续模拟的**动画**模块支持传入自定义的 deltatime, 保证每次模拟效果完全一致
      1. 需要暴露trigger，提供给框架使用
      2. 需要明确预期效果，以及明确预期图片的更新机制
   3. 交互测试：支持模拟点击输入
      1. 需要保证模拟点击输入在各个平台的兼容性
      2. 需要保证模拟框架的稳定性
   4. 框架支持：编辑器构建支持传入自定义宏
      1. 这是为日后的开放框架的可能性提供支持
   5. 特效测试：物理，粒子测试时的随机数和定点数的固化。
      1. 需要明确采样方法和测试方法的区别，测试是否能够涵盖日常使用范围，而不是仅针对预期中用户的行为。（玩家进入酒吧点了一碗炒饭，酒吧炸了）
5. Next
   1. GFX层的测试，满足自动化测试能快速定位底层问题。