**04|为什么要做自动化测试?什么样的项目适合做自动化测试?**

在上一篇文章中，我为你介绍了什么是单元测试，以及如何做好单元测试，今天我来跟你聊聊什么是自动化测试，为什么要做自动化测试，以及什么样的项目适合做自动化测试。

**什么是自动化测试？**

不管你是刚入行的小白，还是已经在做软件测试的工作，相信你一定听说过或者接触过自动化测试。那么，自动化测试到底是什么意思呢？

顾名思义，自动化测试是，把人对软件的测试行为转化为由机器执行测试行为的一种实践，对于最常见的GUI自动化测试来讲，就是由自动化测试工具模拟之前需要人工在软件界面上的各种操作，并且自动验证其结果是否符合预期。

你是不是有点小激动？这似乎开启了用机器代替重复手工劳动的自动化时代，你可以从简单重复劳动中解放出来了。但现实呢？

自动化测试的本质是先写一段代码，然后去测试另一段代码，所以实现自动化测试用例本身属于开发工作，需要投入大量的时间和精力，并且已经开发完成的用例还必须随着被测对象的改变而不断更新，你还需要为此付出维护测试用例的成本。

当你发现自动化测试用例的维护成本高于其节省的测试成本时，自动化测试就失去了价值与意义，你也就需要在是否使用自动化测试上权衡取舍了。

**为什么需要自动化测试？**

为了让你更好地理解自动化测试的价值，即为什么需要自动化测试，我先来跟你聊聊自动化测试通常有哪些优势：

1．自动化测试可以替代大量的手工机械重复性操作，测试工程师可以把更多的时间花在更全面的用例设计和新功能的测试上；

2．自动化测试可以大幅提升回归测试的效率，非常适合敏捷开发过程；

3．自动化测试可以更好地利用无人值守时间，去更频繁地执行测试，特别适合现在非工作时间执行测试，工作时间分析失败用例的工作模式；

4．自动化测试可以高效实现某些手工测试无法完成或者代价巨大的测试类型，比如关键业务7×24小时持续运行的系统稳定性测试和高并发场景的压力测试等；

5．自动化测试还可以保证每次测试执行的操作以及验证的一致性和可重复性，避免人为的遗漏或疏忽。

而为了避免对自动化测试的过度依赖，你还需要了解自动化测试有哪些劣势，这将帮你绕过实际工作中的“坑”。

1．自动化测试并不能取代手工测试，它只能替代手工测试中执行频率高、机械化的重复步骤。你千万不要奢望所有的测试都自动化，否则一定会得不偿失。

2．自动测试远比手动测试脆弱，无法应对被测系统的变化，业界一直有句玩笑话“开发手一抖，自动化测试忙一宿”，这也从侧面反映了自动化测试用例的维护成本一直居高不下的事实。

其根本原因在于自动化测试本身不具有任何“智能”，只是按部就班地执行事先定义好的测试步骤并验证测试结果。对于执行过程中出现的明显错误和意外事件，自动化测试没有任何处理能力。

3．自动化测试用例的开发工作量远大于单次的手工测试，所以只有当开发完成的测试用例的有效执行次数大于等于5次时，才能收回自动化测试的成本。

4．手工测试发现的缺陷数量通常比自动化测试要更多，并且自动化测试仅仅能发现回归测试范围的缺陷。

5．测试的效率很大程度上依赖自动化测试用例的设计以及实现质量，不稳定的自动化测试用例实现比没有自动化更糟糕。

6．实行自动化测试的初期，用例开发效率通常都很低，大量初期开发的用例通常会在整个自动化测试体系成熟，和测试工程师全面掌握测试工具后，需要重构。

7．业务测试专家和自动化测试专家通常是两批人，前者懂业务不懂自动化技术，后者懂自动化技术但不懂业务，只有二者紧密合作，才能高效开展自动化测试。

8．自动化测试开发人员必须具备一定的编程能力，这对传统的手工测试工程师会是一个挑战。

**什么样的项目适合自动化测试？**

看到这里，你心里可能在暗自嘀咕，“有没有搞错啊，自动化测试的劣势居然比优势还多”。那为什么还有那么多的企业级项目在实行自动化测试呢？那么，我接下来要讲的内容就是，到底什么样的项目适合自动化测试？

**第一**，需求稳定，不会频繁变更。

自动化测试最怕的就是需求不稳定，过高的需求变更频率会导致自动化测试用例的维护成本直线上升。刚刚开发完成并调试通过的用例可能因为界面变化，或者是业务流程变化，不得不重新开发调试。所以自动化测试更适用于需求相对稳定的软件

项目。

**第二**，研发和维护周期长，需要频繁执行回归测试

1．在我看来，软件产品比软件项目更适合做自动化测试。

首先，软件产品的生命周期一般都比较长，通常会有多个版本陆续发布，每次版本发布都会有大量的回归测试需求。

同时，软件产品预留给自动化测试开发的时间也比较充裕，可以和产品一起迭代。

其次，自动化测试用例的执行比高于1：5，即开发完成的用例至少可以被有效执行5次以上时，自动化测试的优势才可以被更好地体现。

2．对于软件项目的自动化测试，就要看项目的具体情况了。

如果短期的一次性项目，就算从技术上讲自动化测试的可行性很高，但从投入产出比(ROI)的角度看并不建议实施自动化，因为干辛万苦开发完成的自动化用例可能执行一两次，项目就结束了。我还遇到过更夸张的情况，自动化测试用例还没开发

完，项目都已经要上线了。

所以，对于这种短期的一次性项目，我觉得你应该选择手工探索式测试，以发现缺陷为第一要务。而对于一些中长期项目，我的建议是：对比较稳定的软件功能进行自动化测试，对变动较大或者需求暂时不明确的功能进行手工测试，最终目标是用20％的精力去覆盖80％的回归测试。

**第三**，需要在多种平台上重复运行相同测试的场景.

这样的场景其实有很多，比如：

* 对于GUI测试，同样的测试用例需要在多种不同的浏览器上执行；
* 对于移动端应用测试，同样的测试用例需要在多个不同的Android或者iOS版本上执行，或者是同样的测试需要在大量不同的移动终端上执行；
* 对于一些企业级软件，如果对于不同的客户有不同的定制版本，各个定制版本的主体功能绝大多数是一致的，可能只有个别功能有轻微差别，测试也是需要覆盖每个定制版本的所有测试；
* ……

这些都是自动化测试的最佳应用场景，因为单个测试用例都需要被反复执行多次，能够使自动化测试的投资回报率最大化。

**第四**，某些测试项目通过手工测试无法实现，或者手工成本太高。

对于所有的性能和压力测试，很难通过手工方式实现。

比如，某一个项目要求进行一万并发用户的基准性能测试(Benchmark test），难道你真的要找一万个用户按照你的口令来操作被测软件吗？又比如，对于7×24小时的稳定性测试，难道你也要找一批用户没日没夜地操作被测软件吗？

这个时候，你就必须借助自动化测试技术了，用机器来模拟大量用户反复操作被测软件的场景。当然对于此类测试是不可能通过GUI操作来模拟大量用户行为的，你必须基于协议的自动化测试技术，这个我会在后续的性能测试章节详细叙述。

**第五**，被测软件的开发较为规范，能够保证系统的可测试性。

从技术上讲，如果要实现稳定的自动化测试，被测软件的开发过程就必须规范。比如，GUI上的控件命名如果没有任何规则可寻，就会造成GUI自动化的控件识别与定位不稳定，从而影响自动化测试的效率。

另外，某些用例的自动化必须要求开发人员在产品中预留可测试性接口，否则后续的自动化会很难开展。

比如，有些用户登录操作，需要图片验证码，如果开发人员没有提供绕开图片验证码的路径，那么自动化测试就必须借助光学字符识别(OCR)技术来对图片验证码进行模式识别，而它的设计初衷是为了防止机器人操作，可想而知OCR的识别率会很低，就会直接影响用例的稳定性。

**第六**，测试人员已经具备一定的编程能力。

如果测试团队的成员没有任何开发编程的基础，那你想要推行自动化测试就会有比较大的阻力。这个阻力会来自于两个方面：

* 前期的学习成本通常会比较大，很难在短期内对实际项目产生实质性的帮助，此时如果管理层对自动化测试没有正确的预期，很可能会叫停自动化测试；
* 测试工程师通常会非常热衷于学习使用自动化测试技术，以至于他们的工作重点会发生错误的偏移，把大量的精力放在自动化测试技术的学习与实践上，而忽略了测试用例的设计，这将直接降低软件整体的质量。

**总结**

自动化测试是，把人工对软件的测试转化为由机器执行测试行为的一种实践，可以把测试工程师从机械重复的测试工作中解脱出来，将更多的精力放在新功能的测试和更全面的测试用例设计上。

然而自动化测试试一把“双刃剑”，虽然它可以从一定程度上解放测试工程师的劳动力，完成一些人工无法实现的测试，但并不适用于所有的测试场景，如果维护自动化测试的代价高过了节省的测试成本，那么在这样的项目中推进自动化测试就会得不偿失。

思考题

你在实际项目中接触过哪些自动化测试，自动化测试用例的执行比通常是多少？如果执行比过低，需要频繁更新测试用例，那你思考过你的项目是否真的适合自动化测试吗？或者说，这个项目的哪些部分更适合实施自动化测试？