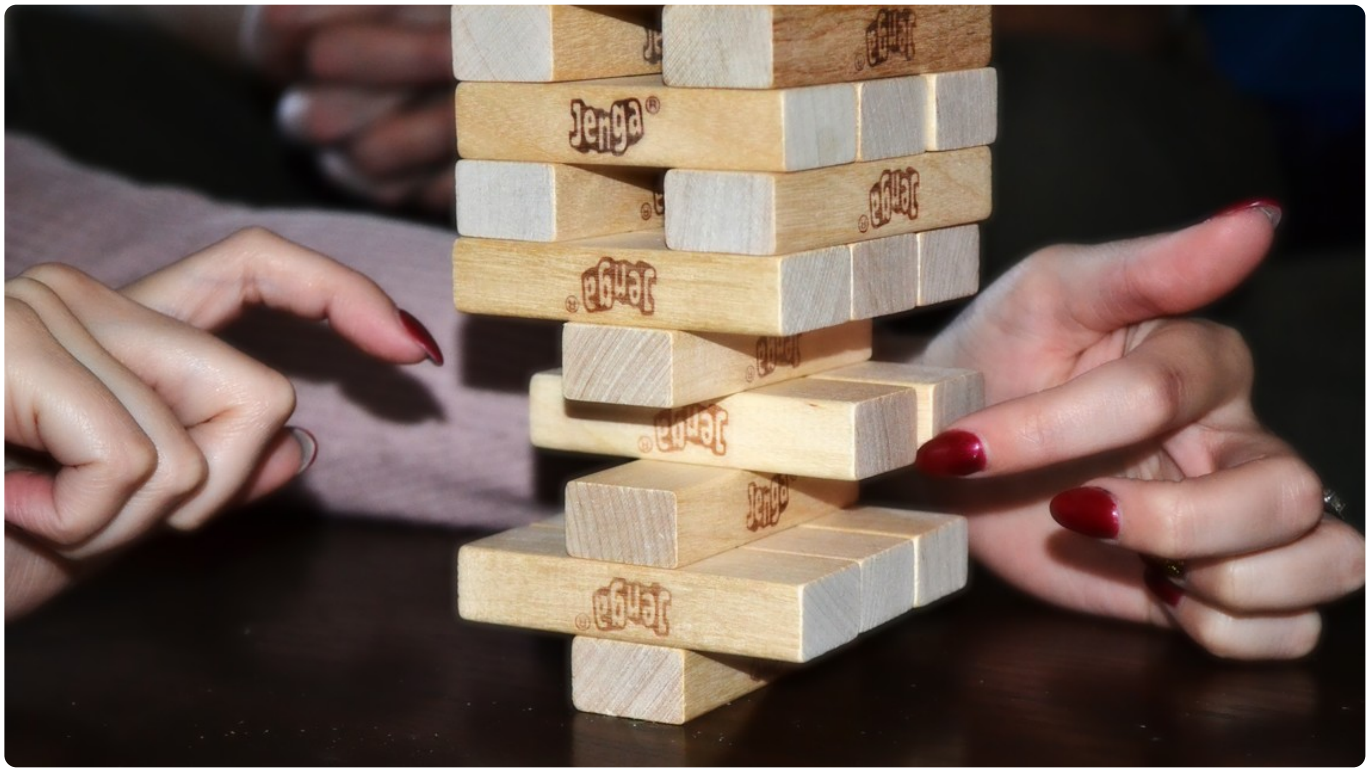
17 | 精益求精：聊聊提高GUI测试稳定性的关键技术

2018-08-06 茹炳晟



讲述：茹炳晟
时长 10:29 大小 4.81M

不知不觉，我已经介绍完了 GUI 测试相关的知识点，你可以先回顾一下这些知识点，是否还有不清楚的地方，也欢迎你给我留言进行讨论。同时，我希望这些知识点，已经帮你搭建了 GUI 自动化测试的知识体系。

那么，今天我将从实际工程应用的角度，和你一起聊聊 GUI 测试的稳定性问题。

如果你所在的公司已经规模化地开展了 GUI 测试，那我相信你们也一定遇到过测试稳定性的问题。**GUI 自动化测试稳定性，最典型的表现形式就是，同样的测试用例在同样的环境上，时而测试通过，时而测试失败。** 这也是影响 GUI 测试健康发展的一个重要障碍，严重降低了 GUI 测试的可信性。

所以，今天我分享的主题就是，如何提高 GUI 测试的稳定性。虽然从理论上来讲，GUI 测试有可能做到 100% 稳定，但在实际项目中，这是一个几乎无法达到的目标。根据我的经验，如果能够做到 95% 以上的稳定性，就已经非常不错了。

要提高 GUI 测试稳定性，首先你需要知道到底是什么原因引起的不稳定。你必须找出尽可能多的不稳定因素，然后找到每一类不稳定因素对应的解决方案。

为此，根据我的实践经验，以及所遇到的场景，我为你总结了五种造成 GUI 测试不稳定的因素：

1. 非预计的弹出对话框；
2. 页面控件属性的细微变化；
3. 被测系统的 A/B 测试；
4. 随机的页面延迟造成控件识别失败；
5. 测试数据问题。

并且，我提供了针对这五种不稳定因素的解决思路。

非预计的弹出对话框

非预计的弹出对话框，一般包含两种场景；

1. **GUI 自动化测试用例执行过程中，操作系统弹出的非预计对话框，** 有可能会干扰 GUI 测试的自动化执行。
   比如，GUI 测试运行到一半，操作系统突然弹出杀毒软件更新请求、病毒告警信息、系统更新请求等对话框。这种对话框的弹出往往是难以预计的，但是一旦发生就有可能造成 GUI 自动化测试的不稳定。
2. **被测软件本身也有可能在非预期的时间弹出预期的对话框，** GUI 自动化测试有可能会因此而失败。
   比如，被测软件是一个电子商务网站，你在网站上进行操作时，很可能会随机弹出“用户调查”对话框。虽然这种对话框是可知的，但是具体会在哪一步弹出却是不可预期的。而这，往往会造成 GUI 自动化测试的不稳定。

怎么解决这类问题呢？

先试想一下，如果你在手工测试时，遇到了这种情况，会如何处理？很简单啊，直接点击对话框上的“确认”或者“取消”按钮，关闭对话框，然后继续相关的业务测试操作。

对 GUI 自动化测试来说，也是同样的道理。具体做法是：

> 当自动化脚本发现控件无法正常定位，或者无法操作时，GUI 自动化框架自动进入“异常场景恢复模式”。
>
> 在“异常场景恢复模式”下，GUI 自动化框架依次检查各种可能出现的对话框，一旦确认了对话框的类型，立即执行预定义的操作（比如，单击“确定”按钮，关闭这个对话框），接着重试刚才失败的步骤。

需要注意的是：这种方式只能处理已知可能出现的对话框。而对于新类型的对话框，只能通过自动化的方式尝试点击上面的按钮进行处理。每当发现一种潜在会弹出的对话框，我们就把它的详细信息（包括对象定位信息等）更新到“异常场景恢复”库中，下次再遇到相同类型的对话框时，系统就可以自动关闭了。

页面控件属性的细微变化

如果页面控件的属性发生了变化，哪怕只是细微的变化，也会导致测试脚本的定位元素失效。

比如，“登录”按钮的 ID 从“Button_Login_001”变成了“Button_Login_888”，那么如果 GUI 自动化测试脚本还是按照原来的“Button_Login_001”来定位“登录”按钮，就会因为 ID 值的变化，定位不到它了，自动化测试用例自然就会失败。

如何解决这个问题呢？还是先试想一下，如果手动操作时遇到了这个问题会怎么处理，然后再把手动处理的方式用编程语言实现。

当“登录”按钮的 ID 从“Button_Login_001”变成了 “Button_Login_888”，你手动操作时可能一眼就发现了。那你是怎么做到一眼发现的呢？

细想一下，你会发现人的思维过程应该是这样的：

> 你发现页面上的按钮（Button）就那么几个，而且从 ID 中包含的关键字（Login）可以看出是“登录”按钮，再加上这个按钮的 ID

> 是“Button_Login_001”，“Button_Login_888”怎么看都是同一个对象，只是 ID 最后的数字发生了变化而已。

现在，我来提炼一下这个定位控件的思路：

1. 通过控件类型（Button）缩小了范围；
2. 通过属性值中的关键字（Login）进一步缩小范围；
3. 根据属性值变化前后的相似性，最终定位到该控件。

看到这里，你得到什么启发了吗？

采用“组合属性”定位控件会更精准，而且成功率会更高，如果能在此基础上加入“模糊匹配”技术，可以进一步提高控件的识别率。

“模糊匹配”是指，通过特定的相似度算法，控件属性发生细微变化时，这个控件依旧可以被准确定位。

目前，一些商用 GUI 自动化测试工具，比如 UFT，已经实现了模糊匹配。通常情况下，你只需要启用“模糊匹配”选项即可。如果某个对象的定位是通过模糊匹配完成的，那么，测试报告中将会显示该信息，明确告知此次对象识别是基于模糊匹配完成的，因为 GUI 自动化工具并不能保证每次模糊匹配都一定正确。

但是，开源的 GUI 自动化测试框架，目前还没有现成的框架直接支持模糊匹配，通常需要你进行二次开发，实现思路是：实现自己的对象识别控制层，也就是在原本的对象识别基础上额外封装一层，在这个额外封装的层中加上模糊匹配的实现逻辑。

通常，我不建议把模糊匹配逻辑以硬编码的方式写在代码里，而是引入规则引擎，将具体的规则通过配置文件的方式与代码逻辑解耦。

被测系统的 A/B 测试

A/B 测试，是互联网产品常用的一种测试方法。它为 Web 或 App 的界面或流程提供两个不同的版本，然后让用户随机访问其中一个版本，并收集两个版本的用户体验数据和业务数据，最后分析评估出最好的版本用于正式发布。

A/B 测试通常会发布到实际生产环境，所以就会造成生产环境中 GUI 自动化测试的不稳定。

这种问题的解决思路是，在测试脚本内部对不同的被测版本做分支处理，脚本需要能够区分 A 和 B 两个的不同版本，并做出相应的处理。

随机的页面延迟造成控件识别失败

随机的页面延迟，也是 GUI 测试防不胜防的。既然是随机的，也就是说我们没有办法去控制它，那有没有什么办法去减少它造成的影响呢？

一个屡试不爽的办法就是，加入重试（retry）机制。重试机制是指，当某一步 GUI 操作失败时，框架会自动发起重试，重试可以是步骤级别的，也可以是页面级别的，甚至是业务流程级别的。

对于开源 GUI 测试框架，重试机制往往不是自带的功能，需要自己二次开发来实现。

比如，eBay 的 GUI 自动化测试框架，分别实现了步骤级别、页面级别和业务流程级别的重试机制，默认情况下启用的是步骤级别的重试，页面级别和业务流程级别的重试可以通过测试发起时的命令行参数进行指定。

需要特别注意的是，对于那些会修改一次性使用数据的场景，切忌不要盲目启用页面级别和业务流程级别的重试。

测试数据问题

测试数据问题，也是造成 GUI 自动化测试不稳定的一个重要原因。

比如，测试用例所依赖的数据被其他用例修改了；再比如，测试过程中发生错误后自动进行了重试操作，但是数据状态已经在第一次执行中被修改了。

这样的场景还有很多，我会在后面的测试数据准备系列文章中详细展开，并分析由此引入的测试不稳定性问题的解决思路。

总结

根据我的实践经验，我归纳了五种造成 GUI 自动化测试不稳定的主要因素，并给出了对应的解决思路。

1. 对于非预计的弹出对话框引起的不稳定，可以引入“异常场景恢复模式”来解决。
2. 对于页面控件属性的细微变化造成的不稳定，可以使用“组合属性”定位控件，并且可以通过“模糊匹配技术”提高定位识别率。
3. 对于 A/B 测试带来的不稳定，需要在测试用例脚本中做分支处理，并且需要脚本做到正确识别出不同的分支。
4. 对于随机的页面延迟造成的不稳定，可以引入重试机制，重试可以是步骤级别的，也可以是页面级别的，甚至是业务流程级别的。
5. 对于测试数据引起的不稳定，我在这里没有详细展开，留到后续的测试数据准备系列文章中做专门介绍。

思考题

在工作中，你还遇到过哪些造成 GUI 测试不稳定的因素，你又是如何来解决的？

欢迎你给我留言。

上一篇　16 | 脑洞大开：GUI测试还能这么玩（Page Code Gen + Data Gen + Headless）？

下一篇　18 | 眼前一亮：带你玩转GUI自动化的测试报告

精选留言 (15)

写留言

sylan215
2018-08-06

29

1. 非预期弹框：
对于非预期的弹框也可以通过检查置顶窗口是否是预期软件窗口，从而确定是否被第三方弹框影响。
2. 页面控件变化：
如果是 selenium 的话，建议优先使用 xpath，这样就算 id、clases、name 等控件属性...
展开

作者回复: 非常高质量的留言，已经关注你的公众号了👍

Cynthia🌸...
2018-08-07

3

页面延迟那个，具体表现应该就是找不到元素吧，那么等一等是不是就可以找到了？
我记得selenium里面有等待函数，而且还可以用sleep。
之前做过的项目，最早没有加等待，就经常因为不稳定而报错，加上等待之后这种概率少多了
展开

张红占
2018-08-06

1

能否通过实例讲解 总结的有点high level
展开

作者回复: 这部分内容的确可以通过实例讲解，但是这样的篇幅会比较大，而且重点也不会太突出，所以选择了总结问题以及对应解决思路的讲解方法。

口水窝
2019-04-02

网络不稳定，无线与流量切换，浏览器版本升级等造成GUI测试不稳定。

展开

小老鼠
2018-10-25

对于非预计的弹出对话框引起的不稳定，可以引入“异常场景恢复模式”来解决。
问：如何作到随时捕获异常？比如用Java或python

对于页面控件属性的细微变化造成的不稳定，可以使用“组合属性”定位控件，并且可以通过“模糊匹配技术”提高定位识别率。...

展开

彼端密悦
2018-10-15

关于 webdriver 中的二维码登录测试，有什么可行的解决思路吗？

展开

KP
2018-09-19

针对控件的定位，模糊匹配是否会影响整个脚本的运行时间和效率？如何抓取显示时间短的控件并做即时响应感觉会更难。

人心向善
2018-08-24

稳定性测试真的让人头疼说心惊胆战一点不为过，举个例子：业务需求要求完成7*24小时的稳定性，跑到尾声的时候发现大量报错且准确率已经不足95%的比例，这种时候是最痛苦的，重新跑吧前面的5.6天就白进行了，不重跑吧，不能保证最终的测试结果，而且还有好多非自然因素，断网或断网环境莫名挂掉，这些都会让稳定性测试猝不及防，顺利的话周期会很短，不顺利的话就很难说要取决的因素太多了.....

展开

涅槃Ls
2018-08-08

打卡17

展开

作者回复: 感谢支持

阳光灿烂的...

2018-08-07

老师每天都期待讲一个高大上的接口测试框架，或者接口测试框架设计思路。

展开

文大头

2018-08-07

我遇到的更多的不稳定，大多是开发对页面做了修改，特别是页面框架的改动，导致元素定位失败。为此，定位元素时尽可能使用元素的相对位置，而不是绝对的xpath路径定位，xpath中也选取相对稳定的元素属性定位，另外xpath本身也支持模糊匹配，我很少需要单独写模糊匹配的代码。
针对延时问题，除了前面留言说的硬等到超时报错外，可以观察是否有其他元素在正常...

展开

hi！girl

2018-08-06

老师，步骤级别、页面级别和业务流程级别的重试机制可以给一个实例吗

展开

作者回复: 步骤和页面级别的retry是会在测试框架中实现的，往往是在try catch中实现重试，而用例级别的retry会在用例调用级别，也就是发起测试的ci流水线中实现。

hi！girl

2018-08-06

对于第四点，通常需要延迟的主要是涉及网络请求的页面，能否先给出一个合理的动态时间等待，后选择重试呢

图•美克尔
2018-08-06

异常场景恢复模式是将在整个操作过程外加try catch实现的吗?

展开

作者回复: 最简单的实现的确是通过try catch

Robert小七
2018-08-06

异常恢复场景是否包含了重试机智？如何解决定位失败后可能产生的无限重试?

作者回复: 如果启用了异常场景恢复模式，那么通常的流程是先有三次步骤级别重试，如果失败了才会启动异常场景恢复模式。一般不会出现无限重试的场景