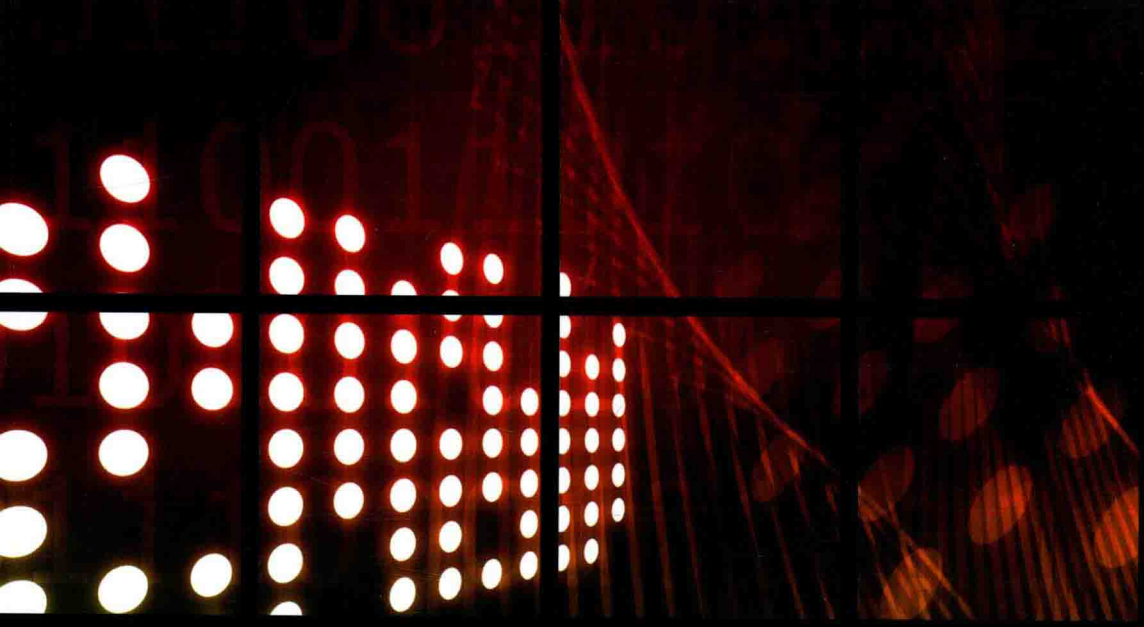

手机测试Robotium实战教程

Practical Mobile Phone Testing with Robotium

杨志伟 编著

全面讲解了用Robotium建立测试工程、测试项目搭建、自动化测试脚本编写、测试框架完善、Robotium自动化测试用例、测试代码批量运行、持续集成、Crash处理、跨应用解决方案、代码覆盖率、代码覆盖率展现、常见错误及解决方法等实战技术和技巧，帮助读者尽快学懂用Robotium进行移动测试。

中国工信出版集团 人民邮电出版社 POSTS & TELECOM PRESS

手机测试Robotium实战教程

Practical Mobile Phone Testing with Robotium

杨志伟 编著

人民邮电出版社
北京

图书在版编目（CIP）数据

手机测试Robotium实战教程 / 杨志伟编著. -- 北京：人民邮电出版社，2015.12
ISBN 978-7-115-40915-7

Ⅰ. ①手… Ⅱ. ①杨… Ⅲ. ①移动电话机－软件－测试－教材 Ⅳ. ①TN929.53

中国版本图书馆CIP数据核字(2015)第260774号

内 容 提 要

本书讲解了用 Robotium 进行移动测试的主要技术，并通过实例，让读者达到学以致用的目的，主要内容为：移动端自动化测试的工具选择、测试开发环境搭建、Robotium 入门、建立测试工程、运行第一个 Robotium 测试实例、被测 App 详细功能介绍、实战测试项目搭建、自动化测试脚本编写、测试框架完善、Robotium 自动化测试用例、测试代码批量运行、持续集成、Crash 处理、跨应用解决方案、代码覆盖率、代码覆盖率展现、常见错误及解决方法等实战技术和技巧，将帮助读者尽快学懂用 Robotium 进行移动测试的知识。

本书适合移动端功能测试人员、Web 端功能测试人员、自动化测试人员、测试开发人员、移动端开发人员阅读学习，也可以作为大专院校相关专业师生的学习用书和培训学校的教材。

◆ 编　　著　杨志伟
责任编辑　张　涛
责任印制　张佳莹　焦志炜

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：14.75
字数：316 千字　　2015 年 12 月第 1 版
印数：1－3 000 册　　2015 年 12 月北京第 1 次印刷

定价：49.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京崇工商广字第 0021 号

前　言

移动互联网近几年呈现出井喷式的发展，技术方面也跟着发生了快速的变化，作者于2011年开始接触Android自动化方面的测试，在这几年的工作、学习和实践中，碰到了不少的测试问题，在不断地遇到问题、解决问题、思考的实战过程中也积累了不少经验，虽然现在网络上有不少关于Android自动化测试方面的资料，但都比较零散，没有系统性地介绍，学习者看了以后还是一知半解，且市面上Android自动化测试方面的书也很少，基于此，就萌发了这样一个想法，把这几年工作实践中用的一些技术、实践心得、工具梳理汇总一下，然后通过书的形式和各位同行分享各种测试技术，这不但对自己是一种提升，也可以帮助读者一同提高软件测试水平。

起先我将整理的内容发布到了百度阅读上，网友看了给出的反馈很不错，很多网友都不习惯在线阅读，纷纷来信表达想看纸质图书，所以，这本书就应运而生了。

Robotium是一款开源的自动化测试框架，可以支持native和hybrid的自动化测试，它的API简单明了，使用起来简单方便，用它做软件测试速度也快。本书主要讲解了用Robotium进行移动测试的技术和技巧，包括移动端自动化测试开发环境搭建、建立测试工程、Robotium测试实例、自动化测试脚本编写、测试框架编写、测试代码批量运行、持续集成、代码覆盖率展现、常见错误及解决方法等知识，帮助读者尽快学会用Robotium进行移动测试的技术。

这是一本偏向实践性的书籍，在实践过程中多多少少会碰到一些问题，所以作者特意为方便读者答疑创建了一个QQ群，用来交流阅读本书过程中遇到的问题，QQ群号：323269785。

本书用到的源代码都可以从网站下载（http://qun.qzone.qq.com/group#!/323269785/share），本书每个章节所用到的工具或者代码都会放到这个网址对应章节的文件夹下，读者可自行选择下载。

由于本人水平有限，书中存有漏洞和错误在所难免，欢迎读者针对本书出现的不足之处提出您的宝贵意见或者建议，可以把您的反馈发送到我的GMAIL邮箱：if1else0@gmail.com。以便以后修订完善。编辑联系邮箱：zhangtao@ptpress.com.cn。

希望本书能给您在Android自动化测试方面带来一些帮助，谢谢大家的阅读。

编　者

目　录

Chapter

1

第1章

自动化测试简介

1.1 何为自动化测试

什么是自动化测试？在一些人眼中，觉得自动化测试是一种比较高大上的东西。但在我看来，自动化测试其实就是通过一定的编程手段，自动执行本来需要手动执行的一系列测试的活动。只要有一定的编程基础，人人都可以参与到自动化测试中来，享受到自动化测试带来的便捷性。现在有很多商业的、开源的自动化测试工具可以更好、更容易地帮助我们进行自动化测试，因此，自动化测试的门槛并没想象中高。

举个小例子来说明自动化测试是怎样让我们的生活变得美好的。测试组长 A 对测试组员 B 和 C 说："有个 Bug 不好重现，为了节约时间，你们两个人分别测一百次，看看会不会重现这个 Bug，这个测试优先级比较高，要尽快哦。" B 二话不说，埋头苦测，把喝水和上厕所的时间都给省了。C 测试之前想了一下，这些测试也不是太复杂，我调用×××工具的几个 API 就可以解决问题了，测试程序中再加个 for 循环不就完事了，二话不说，二十分钟后，就调试好这个测试程序。

半天过去了，B 好不容易抬起了头，终于把这个让人烦的第一百次给测试了，一想到还有好几个 user story 的测试用例要设计，不禁悲从中来。而此时的 C 呢，把需要设计的测试用例弄得差不多了，正悠闲地喝着咖啡呢。

上面的例子一点也不夸张，在现实的工作中很常见，可见掌握一些自动化技能，对工作效率的提高是有很大帮助的。

1.2 自动化测试和手动测试的对比

自动化测试与手动测试相比，具有以下方面的优势。

1. 执行速度快

自动化测试比手动测试速度要快很多，在用例数目多的情况下，这种优势会更加明显。

假设有一个测试用例，需要执行上百次，若手动测试执行，会很枯燥乏味，也会很疲惫，但自动化测试是用机器来执行测试，优势会很明显，它可以成千上万次、昼夜不分地重复执行都没任何问题，

还可以通过多增加硬件支持，缩短运行时间。例如，本来是一台手机运行 1000 个用例，多增加一台手机，就能缩短一半的运行时间，比多增加一个人来手动运行性价比高很多。

2. 可靠性高

手动测试免不了一些人为的失误，如某个测试人员因为生病了状态不好，执行 Case 的过程中就更容易发生一些差错，重复性的次数越多，就越容易发生错误。

自动化测试通过编程执行，确保每次执行的操作都是唯一的，非 0 就是 1，不像手动测试那样还受一些外在因素的制约。

3. 复用性高

举个 Android 平台的例子，我们都知道 Android 平台厂家多，碎片化严重，而兼容性测试又非常重要，这时，自动化测试只需要将脚本放到不同的设备或者 OS 上逐个运行，查看结果是否有问题即可，但手动测试必须用人力去验证应用的兼容性，压力会比较大。

4. 节省人力资源

如果自动化程度高，就可以更好地把测试人员解放出来，节省一定的人力成本。或者让这部分解放出来的测试人员有更多时间去做其他一些有利于保障产品质量的活动。

列举了这么多自动化测试的优点，但不能因此否认手动测试的价值，首先，并不是所有的用例场景都可以进行自动化测试，因为一些技术上的限制，做到 100%自动化测试是不现实的，或者说是性价比不高的。很多测试场景用手动测试会比较适合，如一些用户体验方面的用例和一些视觉感官上面的测试等，切勿只用自动化测试，尽量做到自动化了的测试会真的给实际工作带来了效率上的提升。

1.3　移动端自动化测试工具的选择

之所以需要选择，是因为可选项太多了，不同的平台，会有很多不同的自动化测试工具供选择，其中既有商业性的收费工具，也有开源免费的自动化测试工具。

这本书的主题 Robotium 就是其中一种开源的自动化测试框架，在正式展开前，先看看当下还有哪些在 Android 测试方面比较常用的测试框架可供选择。

1.3.1　Appium

Appium 也是一种自动化测试框架，可以用来测试 natvie、hybrid 和 mobile web APP，Appium 最吸引人的一点是，它既支持 Android 方面的自动化测试，又支持 iOS 方面的测试。

它还支持使用不同的编程语言编写测试代码，但这种便捷性是以牺牲一部分执行速度换来的，因为这需要额外的转换时间，转换成对应的可操作的底层的测试框架。Appium 在 Android 方面底层使用的测试框架是 uiautomator 和 instrumentation，iOS 则使用 Apple 提供的 UIAutomation。明白了这点，也就不奇怪为什么 Appium 可以支持跨进程地操作 iOS 的测试了。

Appium 经常被“吐槽”的还有一点，就是稳定性还有待提升，但它的这些优点也是很明显的，参与的人越来越多，贡献的力量也就越大，经过一段时间的发展，相信 Appium 会

更加稳定和流行。

读者可以访问 Appium 的官方网站 http://appium.io/，在这里可以获得关于它的一些详细介绍。

国内 TesterHome 社区也有不少 Appium 方面的资料和讨论。

1.3.2 uiautomator

uiautomator 是 Google 官方提供的一款自动化测试框架，其主要特点是支持跨进程的操作，这一点极大地方便了对应用外控件的操作。

当然了，uiautomator 在提供便捷性的同时，也存在着一些不足，如只支持 Android SDK Platform、API 16 及以上的。除此之外，它对 WebView 的支持也不好。

还有一点就是调试很不方便，要调试首先需要在启动的命令行配置好调试项后才能进行，启动后还需在 Eclipse 中新建 Remote Java Application 选项，相对来说比较麻烦。虽然也可以通过输出 Log 的形式来调试，但不如其他工具有插件支持调试方式便捷。

uiautomator 的编译运行也比较麻烦，测试代码一有修改，就需要重写、编译产生 jar 文件，然后还要将 jar 文件 Push 到手机指定的目录下才能使用，虽然可以通过批处理文件将这些步骤封装起来，但还是略显麻烦。

想了解更详细的信息，请到它的官方网站，那里可以查看 uiautomator 的 API 详细介绍、命令行运行 uiautomator 的参数及命令解释，以及一些入门的例子。

官方网站：http://developer.android.com/tools/testing/testing_ui.html。

1.3.3 Robotium

Robotium 也是一款开源的自动化测试框架，可以支持 native 和 hybrid 的自动化测试，API 使用起来简单方便，执行速度也快。

但 Robotium 有一个比较大的局限性就是不支持跨进程的操作，但也有相应的解决方案，后面章节会介绍到。

Robotium 在 ADT 插件的支持下，可以很简单地在 Eclipse 下调试运行，这一点比 uiautomator 便捷多了。

Robotium 官方网站的 wiki 标签下还提供了一些简单易懂、容易上手的示例项目，在这里也可以查看 Robotium 的 API 文档，Robotium 的方法命名很直观，一般通过方法名就可以知道这个方法所能实现的功能。

最后，Robotium 是开源的，托管在 GitHub https://github.com/RobotiumTech/Robotium。

它的代码量不大，可以很容易地进行二次开发，然后定制出自己需要的功能。

当然还有其他的自动化测试框架可供选择，如 monkey、monkeyrunner、Testdroid(商业)、Eggplant(商业)等，每种工具都有优缺点，关键在于根据产品的特性，选择一款适合的自动化测试工具，不要盲目跟风，合适才是最重要的，当然可以混合使用多种自动化测试工具来克服一些工具本身的局限性，以便得到更好的自动化测试效果。

Chapter

2

第2章

测试开发环境搭建

2.1　JDK 安装及其环境变量配置

因为要用到 Java 语言，所以 JDK 是首先需要安装的。可以通过下面的链接，选择对应平台的 JDK 版本进行下载。

http://www.oracle.com/technetwork/java/javase/downloads/index.htm。

本书所有的例子选择在 JDK 8 平台下进行。

运行下载的 JDK 安装文件，根据提示完成安装。在安装完毕后，设置环境变量。设置环境变量的目的是可以在任意路径下执行 javac/java 等工具。

（1）在计算机桌面选中计算机后单击鼠标右键，在弹出的菜单中选择“属性”项，打开“系统属性”对话框，选择“高级”选项卡，单击“环境变量”按钮，如图 2.1 所示，在“环境变量”对话框中单击“新建”按钮，添加 JAVA_HOME，变量值为 JDK 的安装目录，如 JDK 的安装目录为 D:\Program Files\Java\jdk1.8.0_11，则添加的环境变量如图 2.2 所示。

（2）将 JAVA_HOME 下的 bin 文件添加到 Path 中，选中 Path 系统环境变量，单击“编辑”按钮，如图 2.3 所示，在 Path 最后面添加%JAVA_HOME%\bin;，如图 2.4 所示。

（3）添加完环境变量后，可以打开 Windows 命令处理程序窗口，执行命令 java -version 验证环境变量是否添加成功，如果添加成功，则显示安装的 Java 版本号，如图 2.5 所示。

图 2.1

图 2.2

图 2.3