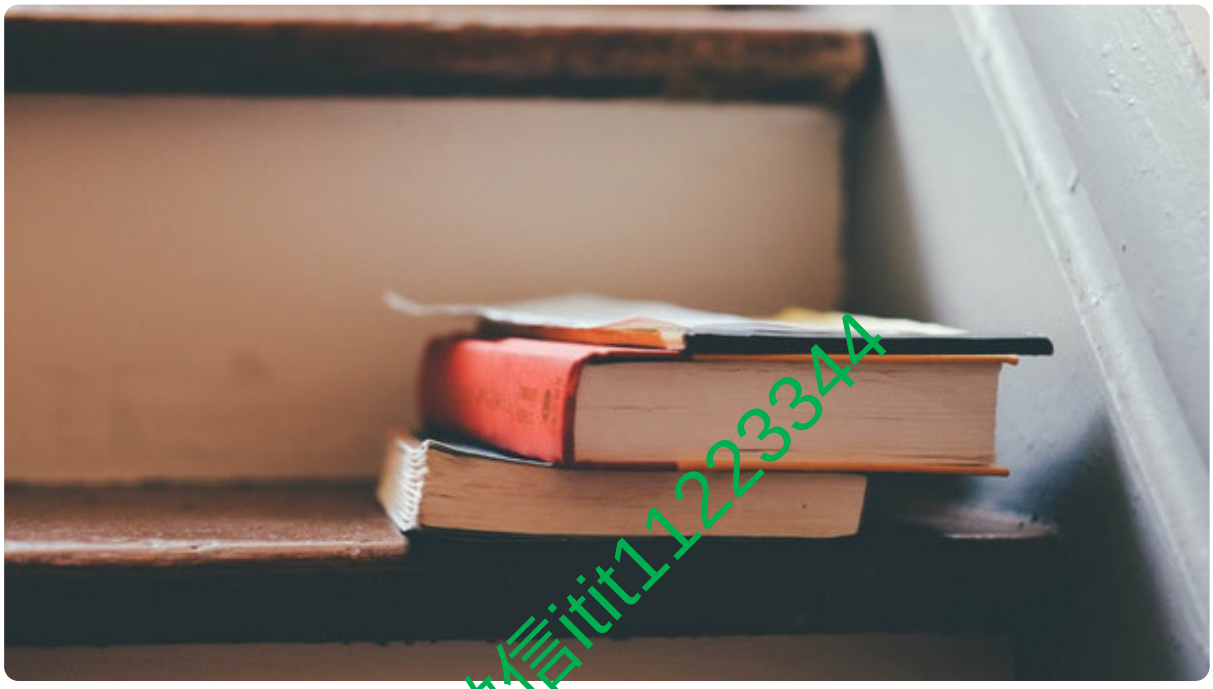
17 框架提升指南：从持续集成到测试数据收集

更新时间：2019-09-25 15:26:01



> 立志是事业的大门，工作是登堂入室的旅程。——巴斯德

估计大家也看出文弱如我居然还是半个武侠迷，其实远不止，年轻时候（PS：现在也不算老，必须不算！）我也是泡在天涯、起点上从武侠追到仙侠再追到玄幻的“书瘾”少年，甚至还曾经尝试去写了一部玄幻的网文，成了一个不知名的“小扑街”。

不管是在武侠还是在玄幻里，主角们除了得到了非常牛的武功秘籍和心法之外，还需要找到一个非常适合自己的外部环境来修炼，这样方能一日千里。我们的测试框架修炼之路也是如此；除了驱动架构思想和设计模式以外，还需要一个完美的外部环境。

在去构建这样一个外部环境之前，我们先回到最初的问题：我们做自动化的初心是什么？难道是为了秀出高大上的设计思路？

测试，更重要的是为了提升我们的测试效率。一个好的自动化测试框架，不仅仅要能让用户方便使用，还需要符合一些基本原则。

这些原则包括：可复用、易维护、定时处理、持续集成、可调试、测试结果自动通知等等。框架的概念是一系列的被事先定义好的标准和规范。在自动化测试中我们经常提到的对测试需求的解析、脚本设计、测试执行、测试报告、维护管理等等，通过框架将它们串联并封装起来，从而使框架的终端用户能够更方便地使用。

我们可能已经应用了或数据或关键字甚至是行为驱动的架构，也已经选择了分层的设计模式，这当然帮助我们的框架解决了很多问题：我们的自动化测试脚本可以通过公共用例层得以复用，而更多的维护工作也通过不同分层的设计模式将其维护的难度降到了最低，可以说在我们的代码层面上把能够做的做到了最好，所以，我习惯于把在代码内的分层和驱动统称为自动化测试框架的内部架构。

有内自然就有外，所以我们还设计了 “外部框架”。所谓的外部框架，其实就是以 Selenium 为核心，辅以外部第三方框架和工具，用以实现持续集成、自动部署、脚本执行、远程调用、报告优化、邮件发送等功能的框架结构。

持续集成：Jenkins

我们就以 JAVA 语言为例，抛砖引玉来聊一聊可以选用的外部框架。目前最流行的持续集成框架毫无疑问就是 Jenkins。当然，Jenkins 只是一个包含丰富插件的平台，持续集成则是一种软件开发实践，更重要的在于思想而不仅仅是工具的使用，持续集成让团队开发成员经常集成他们的代码，而对于测试人员来说，持续集成可以赋予我们两方面的含义：

1. 当我们自动化代码有变更时持续集成，验证自动化测试代码的有效性和结果的正确性；
2. 当研发的代码有更新部署时，我们的自动化测试代码自动执行，重复验证最新版本的测试环境是否符合预期。

持续集成也是在不断进化的，随着 Devops 的发展、随着 Docker 技术的不断应用，Jenkins X 也应运而生，它是个高度集成化的 CI（持续集成）/CD (持续交付) 平台，基于 Jenkins 和 Kubernetes 实现，目的在解决微服务体系架构下的云原生应用的持续交付的问题，简化整个云原生应用的开发、运行和部署过程。

自动化生命周期建设：MAVEN

Maven 是什么？了解的同学一定会说，它是一个非常全面和主流的项目管理的综合工具，简单

译、依赖管理、团队协作等等。对于我们测试来说，Maven 串联了我们自动化生命周期的各个部分：

- Maven+TestNG 来执行底层的 Selenium 脚本；
- Maven+ReportNG 生成可读性更强的测试报告；
- Maven+JavaMail 来将测试结果作为 HTML 邮件发送给相关人员。

到这里为止，我们构建了一套以 Sejenium 为核心，Jenkins + Maven + Selenium + ReportNG + JavaMail 的集持续集成、自动执行、定时任务、测试结果、邮件发送多位一体的自动化测试底层框架。那么我们还缺少什么呢?

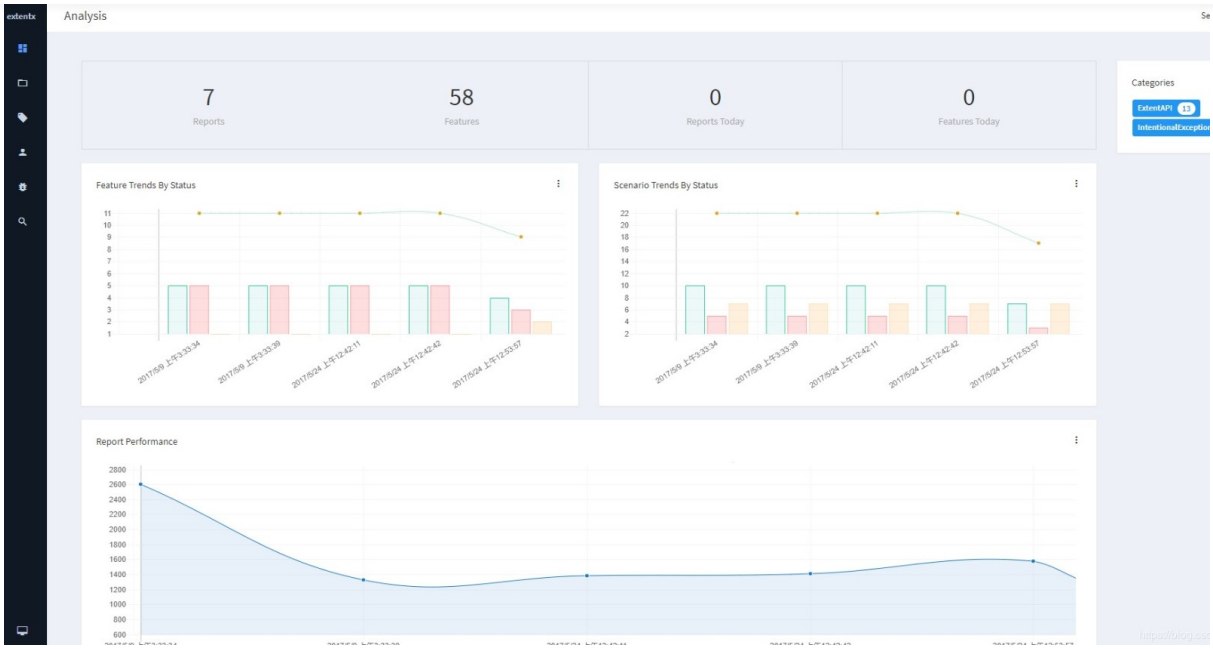
分布式测试：GRID

有时候我们会有大量的测试用例需要执行，又或者我们需要针对不同的浏览器、同一浏览器的多个版本进行兼容性测试，再或者我们希望能够尽量减少测试套件运行的时间，这时候我们就需要支付分布式测试，让我们的自动化测试脚本可以在多套环境中、多台测试机器中同步并行执行。

最早的时候，我们的并行模式可以完全通过 TCP 传输回放的模式来进行，但是 Selenium 提供了我们更好的解决方案 ——Selenium Grid。Grid 提供的所谓的分布式结构其实就是由一个主节点（hub）和若干个代理节点（node）组成。Hub 用来管理各子节点的信息和状态，并且通过精准请求专访的方式，然后把请求的命令分派给代理节点来执行。

测试大数据统计：EXTENTX

有时候会不会觉得 ReportNG 的测试结果仍然不够简明漂亮？有时候我们需要手动去记录每次自动化执行成功率等数据方便进行统计？针对于这样的思维，所以我们有了一套基于 Sails+MongoDB 的自动化测试结果统计工具，它可以非常便捷的集成到 TestNG 中，将测试结果通过监听的方式发送到我们的 MongoDB 中，并通过 Sails 前端清晰完美的呈现出来。



是不是看上去优美了很多，更便于我们的数据归集和统计。呃，作为 ExtentX 的忠实用户，我很伤心地通知大家：ExtentX 已经被废弃。目前有更新的报表服务产生 -- Klov，提供了对数据的详细分析，能够利用历史数据分析接口测试的执行情况，总体来说样式上的改变并不太大，所以同学们可以使用最新的 Klov Server。PS：请原谅一个“老者”仍然在小标题里用 ExtentX 来称呼它。

相信到了这里，一套比较完整的、功能非常强大的 UI 自动化测试框架就在我们的手中诞生了。还是之前曾经说过的，这样一套思路仅仅是我自己对于自动化测试的认识和摸索，在这里抛砖引玉带领大家一起看看更好的自动化世界，也许今天，也许明天，你或者他，可以带我们走入一个更优美、更易用的自动化测试框架。最后，大家可以一起思考一个问题，自动化测试框架中除了我们提到的持续集成、生命周期建设、分布式管理、大数据统计，还可以赋予什么样更有用的功能呢?

← 16 从脚本到框架： PAGE OBJECT模型与分层框架

18 平台设计深入讨论：技术的进步还是能力的倒退? →

精选留言 2

欢迎在这里发表留言，作者筛选后可公开显示

我之前使用grid，发现经常时不时启动远程浏览器后超时

0 回复 举报 2020-02-22

风落几番 回复 **一个小测试**

grid是一个可行的方案，也是现在相对比较稳定的。超时这个就只好在代码层面搞定它了~

回复 2020-02-26 10:46:38

一个小测试

老师，如果还要支持多线程并发执行用例，有什么好的方案，来确保执行时自动化用例之间不会互相影响呢

0 回复 2020-02-22