The Pragmatic Programmers

TURING 图灵程序设计丛书

JavaScript 测试驱动开发

[美] Venkat Subramaniam 著

毛姝雯 译

Jolt大奖图书《高效程序员的45个习惯》作者再次出笔，详述如何告别遗留代码、掌握良好开发实践

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

JavaScript 测试驱动开发

Test-Driving JavaScript Applications

Rapid, Confident, Maintainable Code

[美] Venkat Subramaniam 著

毛姝雯 译

人民邮电出版社

北 京

图书在版编目（CIP）数据

JavaScript测试驱动开发 / （美）文卡特·苏布拉马尼亚姆（Venkat Subramaniam）著 ; 毛姝雯译. -- 北京: 人民邮电出版社, 2018.3
（图灵程序设计丛书）
ISBN 978-7-115-47715-6

Ⅰ. ①J… Ⅱ. ①文… ②毛… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2018)第002902号

内 容 提 要

JavaScript 已经成为使用最广泛的语言之一，它强大且高度灵活，但同时也颇具风险，所以应该用更出色的开发实践来支持。自动化测试和持续集成就是很好的方法，可以降低 JavaScript 带来的风险。本书介绍 JavaScript 自动化测试及其相关实践，主体内容包括两部分：第一部分涵盖自动化测试的基础，介绍如何为同步函数和异步函数编写测试，以及当代码包含复杂的依赖关系时如何实现自动化测试；第二部分通过一个测试驱动开发的实例，让读者能够运用在第一部分所学的内容，为客户端和服务器端编写自动化测试。本书在帮助读者学习和研究测试工具和技术的同时，还会介绍一些软件设计原则，有助于实现轻量级设计，并得到可维护的代码。

本书适合有一定经验的 JavaScript 开发人员，尤其是对 JavaScript 自动化测试感兴趣的读者。

◆ 著 [美] Venkat Subramaniam
译 毛姝雯
责任编辑 朱 巍
执行编辑 温 雪 杨 婷
责任印制 周昇亮
◆ 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
三河市潮河印业有限公司印刷
◆ 开本：800×1000 1/16
印张：19
字数：449千字 2018年3月第1版
印数：1－3 000册 2018年3月河北第1次印刷
著作权合同登记号 图字：01-2017-7986号

定价：79.00元

读者服务热线：(010)51095186转600 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

站在巨人的肩上

Standing on Shoulders of Giants

iTuring.cn

版权声明

对身为教授的PSK叔叔的深情回忆，点燃了我对编程的热情。

本书赞誉

这本书不仅逐步说明了如何测试JavaScript，还提出了一种全面而又简化的方法来促成良好的设计。Venkat用词生动幽默，使得这本书的内容既有教学意义，又不乏趣味性。

——Kimberly D. Barnes，GoSpotCheck高级软件工程师

Venkat以他一贯的独特风格把JavaScript测试去芜存菁。他采用了一种非常受欢迎的实用主义方式来探讨这个有时很棘手的平台的测试问题。与此同时，还介绍了框架、工具的使用，并且提供了很有价值的建议和见解。强烈推荐这本书！

——Neal Ford，Thoughtworks公司主管、软件架构师、Meme Wrangler

如果你已经很熟悉另一门语言的TDD实践，这本书则回答了与测试驱动JavaScript应用相关的所有问题；如果你是JavaScript程序员，但还没有接触TDD，那么Venkat将通过几个真实的示例让你沿着这条路充满自信地走下去。

——Naresha K.，Channel Bridge软件实验室首席技术专家

这本书展示了如何把TDD用于前端技术，这正是我一直梦寐以求的。它包含了实用的示例和大量有用的信息。我学到了良好的实践，不再编写遗留代码。

——Astefanoaie Nicolae Stelian，PRISA高级程序员

致　谢

我花了许多周末和夜晚远离家人来编写本书，它的成功出版离不开家人的大力支持。所以，我首先要感谢妻子Kavitha以及两个儿子Karthik和Krupakar。

其次，我要感谢技术审阅人，他们贡献了自己宝贵的时间、知识和才智，从而为本书增色不少。非常感谢Kim Barnes（@kimberlydbarnes）、Nick Capito、Neal Ford（@neal4d）、Rod Hilton（@rodhilton）、Brian Hogan（@bphogan）、Charles Johnson（@charbajo）、Pelle Lauritsen（@pellelauritsen）、Daivid Morgan（@daividm）、Kieran Murphy、Ted Neward（@tedneward）、Maricris Nonato（@maricris_sn）、Al Scherer（@al_scherer）、Jason Schindler（@Volti_Subito）和Nate Schutta（@ntschutta）。他们都为完善本书提供了帮助。本书中任何遗留的错误都归因于我。

我还要感谢本书的早期读者，感谢他们在本书的论坛[①]上提供了宝贵的反馈，并在勘误页面[②]上提交了书中的错误。感谢Robert Guico、Naresha K.、Dillon Kearns、Chip Pate、Astefanoaie Nicolae Stelian和Bruce Trask。特别感谢Tim Wright的火眼金睛，他仔细试验了每一个示例。

我有幸让Jackie Carter编辑了我的又一本书，每次跟她联系都让我想起我为什么请求她编辑本书。我从她身上学到了很多东西，关于写作，关于耐心，等等。谢谢Jackie，感谢你的帮助、指导、鼓励和交流。

本书的写作灵感来源于与我在各个会议中有过交流的开发者。感谢他们提出了非常有趣的问题，并且与我进行了深入的交流，这使得我的想法得以成形。另外，感谢会议的组织者为我提供了与优秀的开发者进行交流的平台。

非常感谢Pragmatic Bookshelf的优秀工作人员，感谢他们采纳了本书，和我一起将编写本书的想法变成现实。感谢Janet Furlow、Andy Hunt、Susannah Pfalzer、Dave Thomas以及为本书提供帮助的其他工作人员。

① https://pragprog.com/book/vsjavas#forums
② https://pragprog.com/titles/vsjavas/errata

前　言

我很早就热衷于编写代码。数十年过去了，编程的热情虽然丝毫未减，但是我对软件开发经济学也产生了兴趣。变更的成本以及我们应对变更的速度至关重要。软件行业的发展日新月异，并且正不断走向成熟。我们通过编程获得报酬，而如果编写的代码很糟糕，那么就会有人付钱请我们返工，对代码修修补补。这是个恶性循环。作为专业的程序员，我们必须大幅提高编写代码的标准。

JavaScript真的是一只“黑天鹅”[①]——谁能想到它会成为使用最为广泛的语言之一呢？它强大，具有高度的灵活性，但同时也颇具风险。我其实是非常喜欢JavaScript的，我喜欢它的强大，也喜欢它的灵活性。

仅仅因为有风险就回避其强大和灵活的解决方案，这是不可取的；相反，我们应该用更出色的开发实践来运用这些特性。使用自动化测试以及持续集成就是一种更出色的开发实践。我们可以依赖自动化测试和短反馈循环来降低JavaScript带来的风险。

近几年来，JavaScript的整个体系发展迅猛，同时也涌现了大量自动化测试工具。正是由于这些工具以及短反馈循环、持续集成等开发实践，软件工程才没有止于理论，如今使用JavaScript的每个程序员都可以应用这些准则。我撰写本书的目的就在于激发、鼓励以及指导你提高自己，从而迈向更高的标准。感谢你阅读这本书。

本书内容

本书主要介绍自动化测试及其相关实践，用以维持严格的开发进度。通过本书，你将学习如何使用一些工具和技术来自动化验证客户端（包括jQuery和Angular）和服务器端（Node.js和Express）的JavaScript应用。

你将在本书中学习如何有效地使用以下工具：

- Karma
- Mocha

---

① https://en.wikipedia.org/wiki/Black_swan_theory。在发现澳大利亚之前，欧洲人认为所有天鹅都是白色的，还常用“黑天鹅”指代不可能存在的事物。但欧洲人的这个信念却随着第一只黑天鹅的出现而崩塌了。因此，黑天鹅的存在代表不可预测的重大稀有事件，意料之外却又改变一切。人们总是对一些事物视而不见，并习惯以有限的生活经验和不堪一击的信念来解释这些意料之外的重大冲击。这就是“黑天鹅理论”。——译者注

- Chai
- Istanbul
- Sinon
- Protractor

在研究这些工具的同时，你也将学习一些软件设计原则，遵循这些原则有助于实现轻量级设计，得到可维护的代码。如果你正在使用其他的测试工具，比如Jasmine或Nodeunit，那么也可以很容易地将本书中的测试方法运用在这些工具上。

本书包括两大部分。第一部分涵盖了自动化测试的基础。在这一部分中，你将学习如何为同步函数和异步函数编写测试，以及当代码包含复杂的依赖关系时如何实现自动化测试。在第二部分中，通过一个测试驱动开发的实例，你将运用在第一部分所学的内容来为客户端和服务器端编写自动化测试。

本书包括以下章节。

- **第1章，自动化测试让你重获自由**
  这一章阐述为何自动化验证对于可持续开发如此重要。
- **第2章，测试驱动设计**
  这一章通过一个小案例来引导你为服务器端和客户端代码编写自动化测试。你将学习如何创建测试列表，如何增量开发，以及一次只为一项测试编写最少代码。
- **第3章，异步测试**
  有些异步函数执行回调，有些则返回`Promise`对象。这一章将介绍测试异步函数所面临的挑战，以及不同的测试方法。
- **第4章，巧妙处理依赖**
  无论是客户端还是服务器端，依赖都普遍存在。它们导致测试变得非常困难、脆弱、不确定以及缓慢。这一章将阐述如何尽可能地移除依赖，也将通过测试替身（test double）对一些复杂的依赖进行解耦和替换，从而让测试更为便利。
- **第5章，Node.js测试驱动开发**
  这一章将引导你测试驱动一个功能完整的服务器端应用。它展示了如何从高层次的策略设计出发，使用测试来改进设计。在这一章中，你将对代码覆盖率进行估算，从而了解自动化测试究竟验证了多少代码。
- **第6章，Express测试驱动开发**
  使用Express可以轻松编写Web应用。在这一章中，你将学习如何通过自动化测试轻松编写可维护的代码。首先，你将学习为数据库连接设计自动化测试，然后了解模块功能，最后学习路由方法。
- **第7章，与DOM和jQuery协作**
  这一章将为第6章中开发的客户端应用创建自动化测试，其中展示了如何为直接操纵DOM的代码以及依赖jQuery库的代码编写测试。

- **第8章，使用AngularJS**
  AngularJS是说明性的、响应式的和高性能的。它不仅简化了客户端代码的编写，而且为自动化测试的编写提供了一组强大的工具。在这一章中，通过为Express应用编写另一个版本的客户端程序，你将学习测试AngularJS 1.*x*应用所需要的技术。
- **第9章，Angular 2测试驱动开发**
  仅仅说AngularJS经历了一次重大革新还远远不够。Angular 2在很多方面都不同于AngularJS 1.*x*——组件取代了控制器，管道取代了过滤器，依赖注入更为明确，基于注解的通信——而且Angular 2是用TypeScript而非JavaScript编写的。在这一章中，你将使用Angular 2和JavaScript重写前一章中的客户端应用，完全从头开始，测试优先。
- **第10章，集成测试和端到端测试**
  进行端到端测试或者UI层的测试是很有必要的，但是这类测试必须控制在最低限度，应着重于在其他测试中没有被覆盖到的那些部分。在这一章中，你将学习哪些方面是需要关注的，哪些对于测试来说是至关重要的，以及哪些是需要避免的。这一章还将展示如何编写完全自动化的集成测试——从数据库层到模块功能，再到路由，最后到UI。
- **第11章，测试驱动你自己的应用**
  这一章对全书进行了总结，通过示例讨论了如何实现自动化测试，以及测试的层次、规模和好处。最后讨论了如何将这些运用到你自己的项目中。

本书读者

如果你使用JavaScript编程，那么本书正是为你准备的。程序员、具有实践经验的架构师、团队领导、技术项目经理，以及任何想要以可持续的速度编写可维护的JavaScript应用的读者都能从中获益。

本书假设读者熟悉JavaScript，所以不会讲授任何JavaScript语法知识。各章会使用不同的技术来编写示例程序，我们假设读者对这些技术已经有所了解。例如，第8章和第10章就假设读者已经了解AngularJS 1.*x*，但其他章并不依赖这些知识。

本书的每一页都有一些能让你直接运用到自己项目中的内容，包括单元测试、集成测试、代码覆盖，或者是使用这些技术的理由。架构师和技术领导能够使用本书来指导团队改进技术实践。技术项目经理能够从中知晓测试驱动开发JavaScript的原因，了解测试的层次和规模，并决定如何在项目中运用自动化测试，同时还可以使用本书来激励团队以快速反馈循环的方式编写应用。

如果已经非常熟悉这些技术了，那么你可以使用本书对其他开发者进行培训。

本书是为广泛使用JavaScript并且想要提高自己技术的那些开发者编写的。

网络资源

你可以从Pragmatic Bookshelf的官方网站上下载本书中所有示例的源代码[①]，还可以在论坛上

① http://www.pragprog.com/titles/vsjavas

提交勘误、发表评论或提出疑问。[①]

如果阅读的是本书的PDF版本，那么你可以点击代码清单上方的链接来查看或者下载该示例代码。

附录中的“网络资源”部分列出了书中引用到的所有资源，以便你参考。

你自己的工作空间

在阅读本书的过程中，你也许想要亲自实践一下书中的代码示例，这就需要安装各种工具，并为每个示例新建目录和文件。这项工作单调乏味，不过你可以使用本书预先创建的工作空间。

该工作空间主要包含一些package.json文件，这些文件描述了编写每个示例需要使用的依赖库和工具。此外，该工作空间还为每个示例创建了合理的目录结构和一些便于你编辑的文件。当你开始实践这些示例时，无须反复敲打命令来安装所有需要用到的工具。相反，一旦切换到某个示例项目的目录下，只需一条`npm install`命令就可以下载你的系统所需要的所有内容。安装完成后，就可以尽情为该示例编写代码了。

现在，花一分钟从Pragmatic Bookshelf Media链接[②]下载这个工作空间吧。如果你使用的是Windows系统，那么请将tdjsa.zip下载到C:\目录下；如果你使用的是其他系统，那么请将该文件下载到home目录。下载完成后，使用`unzip tdjsa.zip`命令来解压该文件。

解压完成后，你应该会看到一个名为tdjsa的目录，该目录下包含了一些子目录，每个子目录对应一章，并且其中包含了每一章的示例以供你进行练习。

注意，将代码从PDF阅读器中复制粘贴到代码编辑器或IDE中通常会存在问题。复制的结果取决于PDF阅读器以及代码编辑器或IDE。复制并粘贴后，你可能需要对代码进行格式化，否则可能无法运行。每个人的具体情况可能有所不同。你可以从前文中列出的网站上下载源代码，避免从PDF阅读器中复制粘贴代码。

电子书

扫描如下二维码，即可购买本书电子版。

① 也可以到本书中文版页面下载代码，查看和提交勘误：http://www.ituring.com.cn/book/1920。——编者注

② https://media.pragprog.com/titles/vsjavas/code/tdjsa.zip

目　录

第1章 自动化测试让你重获自由

当我们编写的应用成功上线后，每个人都会获益良多。产品故障的成本非常高，应该尽可能降低这种风险。如今，技术发达、信息发达、信息透明度高，一旦应用发生故障，全世界都会知道。有了自动化测试，我们就能够尽早发现故障，降低风险，从而开发出健壮的应用。

自动化测试对代码设计产生了深远影响。它促使我们编写模块化、高内聚、低耦合的代码，让代码易于修改，这有利于降低变更的成本。

也许你急于着手编写代码，但了解一下为什么要用到自动化测试以及可能面临的一些阻碍，可以让你为深入学习后续章节的技术做好准备。我们来快速探讨一下自动化测试的优势以及它带来的挑战，并研究一下如何运用短反馈循环。

1.1 变更的挑战

代码在其生命周期内会被多次修改。如果一位程序员告诉你他的代码从创建起就从未修改过，这就意味着他的项目后来被取消了。如果一个应用想要存续下去，就必须不断改进。我们会不断增强应用的现有功能，添加新功能，并且经常修复应用中的bug。每次变更都面临着一些挑战：

- 变更的成本应该合理
- 变更应该带来正面的影响

我们依次讨论一下这些挑战。

1.1.1 变更的成本

良好的设计应该是灵活的、易于扩展的、维护成本低的。但是如何能够分辨出这些特性呢？我们不能等着查看设计的结果来了解其质量——这样可能已经太晚了。

测试驱动的设计有助于解决这个问题。在这种设计方法中，我们首先创建一个初始的、主要的、策略性的设计。然后，通过一系列策略步骤，运用一些基本的设计原则（参见*Agile Software*