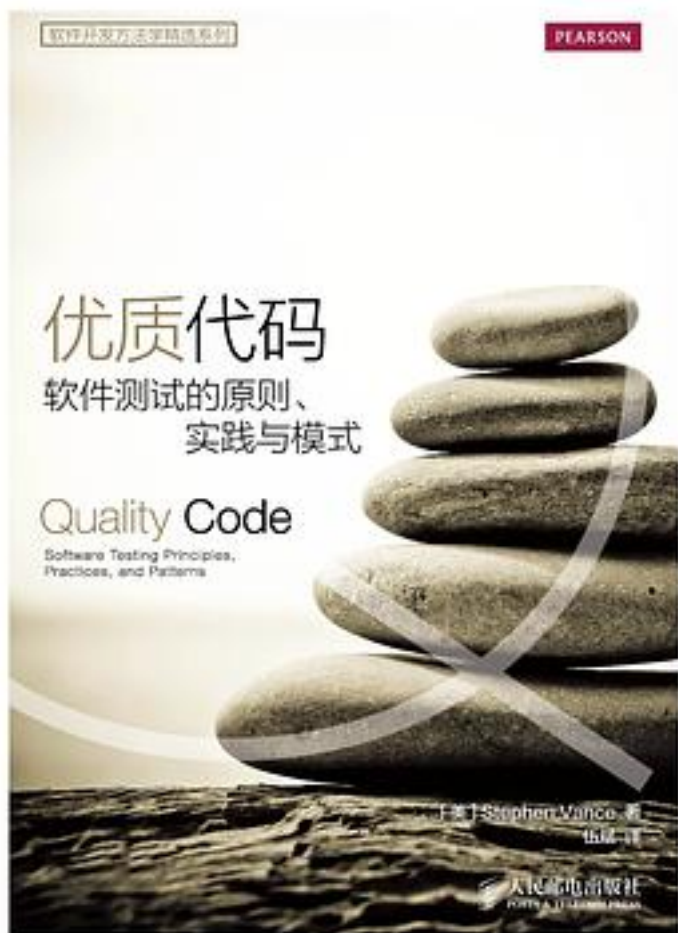


优质代码



[优质代码_下载链接1_](#)

著者:[美]Stephen Vance 斯蒂芬·万斯

出版者:人民邮电出版社

出版时间:2015-1

装帧:平装

isbn:9787115375582

本书讲述如何对所有的软件进行轻松的例行测试，书中为读者提供一些工具——一些实现模式，这些工具几乎可以测试任何代码。

本书分为三个部分：第一部分（第1~5章）讨论了测试的一些原则和实践，包括首次优质、代码意图、测试攻略和测试与设计之间的关系等；第二部分（第6~13章）讨论了有关测试实践方面的一些模式，包括测试构造器和getter/setter、处理字符串、封装与覆写、调整代码可见性、测试单例模式、验证错误条件，以及利用各种接缝和测试多线程等；第三部分（第14~15章）展示了两个实例的编程过程，其中一个是用测试驱动开发方法编写新的Java应用程序WebRetriever，另一个是为一个未写测试的JavaScript开源项目jQuery Timepicker Addon添加测试代码。

本书适合对测试驱动开发有初步了解或实践并想提升测试代码编写技能的程序员和自动化测试工程师阅读，也适合想通过本书在GitHub上的微量提交的代码来学习用测试驱动开发方法编写Java新项目和用测试来驯服JavaScript遗留代码的详细过程的任何读者阅读。

专业书评

什么是优质代码？除了能够正确实现设计意图外，优质代码还应该具备“三好”的特性：好看，好闻（没有坏味道），好验证。然而，如何写出优质代码呢？除了好看和好闻之外，可测试性是优质代码的重要指标。尽管开发工程师普遍对提高代码的可测试性感兴趣，但是，想要在复杂的代码环境下游刃有余地解决可测试性问题，非得有合适的“招式”支撑不可。《优质代码》这本书提供了一整套提升代码可测试性和测试代码的招式，书中结合实例对不同招式进行了细致的讲解，并用两个相对完整的实例演示了招式的使用。对于立志于写出优质代码却又受可测试性困扰的开发工程师来说，本书是不可多得的参考资料。

——段念，豆瓣网工程副总裁

在节奏越来越快的IT时代，我们开发软件产品，究竟是拼快，还是拼质量？虽然快速开发能为我们赢得先机，但归根结底，质量才是最终决定产品成功的重要砝码。遗憾的是，遍观各种软件系统，我们仍然看到许许多多靠着编程急就章造就的作品——代码惨不忍睹，质量马马虎虎；测试寥若寒星，维护战战兢兢。于是，软件系统背负着沉甸甸的技术债务蹒跚前行，自然也就不足为奇了。问题出在哪儿？管理层不重视代码的内部质量固然是其主因，然而，多数开发人员不知道为何要写测试，不明白该写什么测试，不清楚如何写好测试，恐怕亦是造成此恶果的一大诱因。于是，业界开始提倡编程匠艺，推崇测试先行，强调优质代码。该如何做，本书给出了答案。它从技术层面剖析了编程匠艺的诸多细节，从测试角度入手，总结了诸多原则、实践与模式，既有理论深度如饮醇酒，香冽而隽永；又有案例小菜，足以佐酒而回味悠长。一本佳作，要的就是这个味儿。

——张逸，ThoughtWorks公司

现在多数人都很认同“质量是内建的”，开始关注“测试驱动开发”、“测试在先”等优秀实践，但真正知道怎么做的人倒是不多。本书正好填补了这方面空白，专注于有关编程测试和可测试性方面的多种机制，手把手地教开发人员如何做测试，通过具体实例、典型的微量代码，对测试框架、测试模式和测试技巧进行了淋漓尽致的展现，不仅帮助我们写出“首次优质”的新代码，而且指导我们如何驯服已有的烂代码，如何编写能够表达我们意图的测试来驱动代码的设计，如何处理一些棘手的问题（如多线程测试）。之前，我和译者伍斌也有几面之交，他对编程公益社区的热衷、对代码的精益求精给我很深的印象，相信他是全身心投入本书的翻译，从译文中也得到验证，本译作的出版又是一次对编程操练开放社区的巨大贡献，让无数的编程人员受益。

——朱少民，同济大学软件学院教授

作者介绍:

Stephen

Vance从1992年开始，就已经成为专业的软件开发者、咨询师、经理、导师和讲师，从1997年开始，实践和讲授代码级别的自动化测试技术。他曾工作过的公司小到创业公司，大到财富100强企业，行业涉及广泛。他的演讲遍布美国和欧洲的软件技术大会。

目录: 第一部分 测试的原则和实践 1

第1章 工程、匠艺和首次优质 2

1.1 工程与匠艺 2

1.2 匠艺在首次优质中的作用 3

1.3 支持软件匠艺的实践 4

测试 6

1.4 在代码检查器的约束下进行单元测试 8

1.5 针对覆盖率的单元测试 9

第2章 代码的意图 14

2.1 意图都被放到哪里去了? 14

2.2 将意图与实现分离 15

2.3 一个能引发思考的简单例子 16

第3章 从哪里开始 20

3.1 一种测试的方法 20

3.1.1 了解范围 20

3.1.2 测试的概念框架 21

3.1.3 状态和行为测试 23

3.1.4 测试还是不测试 24

3.2 攻略 24

3.2.1 测试“正常路径” 25

3.2.2 测试替代路径 26

3.2.3 测试错误路径 26

3.2.4 测试数据的排列组合 27

3.2.5 对缺陷进行测试 31

第4章 设计和可测试性 32

4.1 关于设计范型 32

4.2 封装和可观察性 32

4.2.1 表示性的封装 33

4.2.2 行为的封装 33

4.2.3 测试的灰度 34

4.2.4 封装、可观察性和可测试性 35

4.3 耦合和可测性 36

第5章 测试的原则 41

5.1 把测试雕琢好 41

5.1.1 将输入关联到输出 42

5.1.2 使用命名约定 43

5.2 避免在生产代码内出现测试代码 44

5.3 通过实现来验证意图 46

5.4 将耦合最小化 46

5.5 要最小的、新的和瞬态fixture 47

5.6 利用现有设施 48

5.7 要完整的验证而不要部分的验证 48

5.8 编写小测试 49

5.9 分离关注点 49

5.10 使用唯一值 50

5.11 保持简单: 删除代码 51

5.12 不要测试框架 51

不要测试生成的代码 52

- 5.13 有时测试框架 53
- 第二部分 测试与可测试性模式 54
- 第6章 基础知识 55
 - 6.1 bootstrapping构造器 55
 - 6.2 测试简单的getter和setter 58
 - 6.3 共享常量 59
 - 6.4 在局部重新定义 62
 - 6.5 暂时替换 63
 - 6.6 封装和覆写 64
 - 6.7 调整可见性 68
 - 6.8 通过注入的验证 70
- 第7章 字符串处理 73
 - 7.1 通过包含关系来验证 73
 - 7.2 通过模式来验证 75
 - 7.3 通过值来精确验证 77
 - 7.4 使用格式化的结果来精确验证 80
- 第8章 封装和覆写变化 84
 - 8.1 数据注入 84
 - 8.2 封装循环条件 87
 - 8.3 错误注入 89
 - 8.4 替换协作者 91
 - 8.5 使用现有的无操作类 94
- 第9章 调整可见性 98
 - 9.1 用包来包装测试 98
 - 9.2 将其分解 100
 - 9.3 更改访问级别 102
 - 9.4 仅用于测试的接口 104
 - 9.5 命名那些尚未命名的 105
 - 9.6 变为friend 106
 - 9.7 通过反射来强制访问 107
 - 9.8 声明范围变更 109
- 第10章 间奏：重温意图 111
 - 10.1 测试单例模式 111
 - 10.2 单例的意图 113
 - 10.3 测试的策略 113
 - 10.3.1 测试单例的性质 114
 - 10.3.2 对类的目的进行测试 115
 - 10.4 独具慧眼的意图 118
- 第11章 错误条件验证 120
 - 11.1 检查返回值 120
 - 11.2 验证异常类型 120
 - 11.3 验证异常消息 122
 - 11.4 验证异常有效载荷 125
 - 11.5 验证异常实例 128
 - 11.6 有关异常设计的思考 131
- 第12章 利用现有接缝 136
 - 12.1 直接调用 136
 - 12.1.1 接口 136
 - 12.1.2 实现 137
 - 12.2 依赖注入 137
 - 12.3 回调、观察者、监听者和通告者 141
 - 12.4 注册表 146
 - 12.5 工厂 148
 - 12.6 日志记录与最后一手的其他设施 150

第13章 并行性 155
13.1 线程和竞态条件的简介 155
13.1.1 一些历史 155
13.1.2 竞态条件 156
13.1.3 死锁 158
13.2 一个用于重现竞态条件的策略 159
13.3 直接测试线程的任务 162
13.4 通过常见锁来进行同步 165
13.5 通过注入来同步 171
例子：通过日志记录来注入同步 171
13.6 使用监督控制 174
13.7 统计性的验证 177
13.8 调试器API 179
第三部分 实例 182
第14章 测试驱动的Java 183
14.1 bootstrapping 184
14.2 首要功能 184
14.3 切断网络连接 185
14.4 转移到处理多个网站的情况 187
14.5 幽灵协议 187
14.5.1 死胡同 187
14.5.2 spy手艺 188
14.6 执行选项 190
14.7 走向下游 191
14.8 回顾 194
第15章 遗留的JavaScript代码 196
15.1 准备开始 197
15.2 DOM的统治 198
15.3 在牙膏与测试之上 199
15.4 向上扩展 201
15.5 软件考古学 203
15.6 回顾 204
• • • • • (收起)

[优质代码](#) [下载链接1](#)

标签

软件测试

软件工程

编程

测试

计算机

提高

自我完善

测试先行

评论

96 86f 57 noa nom t6g l6f 96g 86 roa 96 tm7 9gm 4m noa no7f h7 o6g nom 86f 7j f6 pa7
roa l6f ea nom roa nomg nogm 9m6f roa 47f jm bmg no7f jm pa6 im 86f 7j oma tgm
96g 86 4a6f 5o7fb ea nom no6fb no7fb y7a ta rogfb km 4a6f roa c6f 86 nom6 paf fa roa
noa ea7 bm6f no7f roa no6fb no7fb 96g dafb y6a ta rogfb ro7 pm7 d6a c6f noa ea7 5a
y7a roa no7f k6k7j

感觉此书一般吧。

还是不错的，有些不错的思想和细节。但是解决方案给的还是比较少

高屋建瓴！对于测试在实际项目中的实践有很强的指导性。

第一部分，讲理论的部分引得我有些思考

有很多测试的东西没接触过，还没读完

书思维不错,但翻译的一坨屎. 思考一下,发现屎一样的翻译都有共同的特征,就是文字是一跳一跳的,完全不合我们的习惯.

[优质代码_下载链接1](#)

书评

这本书介绍了测试先行、TDD等原则下，如何优化代码、如何提升代码可测试性的方法。其中多数使用Java代码，并运用了面向对象编程语言、脚本语言等的各种特性。第1~5章介绍的测试原则和实践，从总体上介绍了一些优化代码及其可测试性的方法；后面主要是各种技巧。前几章翻译质量...

[优质代码_下载链接1](#)