

## <<软件开发与创新>>

### 图书基本信息

书名：<<软件开发与创新>>

13位ISBN编号：9787115342942

10位ISBN编号：7115342946

出版时间：人民邮电出版社

作者：ThoughtWorks公司

译者：ThoughtWorks中国公司

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<软件开发与创新>>

### 内容概要

在这本文集中，你会找到各种来自一线的建议，诸如持续集成、测试，以及改善软件交付流程，等等。

本书会介绍人们怎样在面向对象语言中使用函数式编程技术，现代Java Web应用，以及处理JavaScript中的各种问题。

书中还会概览当下最有趣的程序设计语言，以及信息可视化的发展情况。

这些都是久经考验的洞见，因为它们都是来自ThoughtWorks专家的实践经验。

本书通过各种实用及时的话题，展现了ThoughtWorks宽泛的才智和对编程技术的独特见解。

## <<软件开发与创新>>

### 作者简介

#### Farooq Ali

作为一位多专业技能的通才以及T型思考者，Farooq乐于从诸多不同的角度思考问题，帮助团队提出创新性的解决方案。

作为ThoughtWorks的首席咨询师，他曾经担当过许多不同的角色：开发人员、业务分析师、项目经理、用户体验设计师。

Farooq对于可视化思考充满激情，并将其运用于产品构思、代码美学、数据分析等诸多领域。

Farooq现任ThoughtWorks美洲社会影响力计划的负责人，为技术、创新与社会影响寻求更好的解决方案。

#### Ola Bini

Ola Bini来自瑞典，是一名程序设计语言极客，在ThoughtWorks芝加哥分部工作。

他是JRuby的核心开发者之一，从2006年开始参与JRuby的开发。

曾几何时，Ola厌倦了所有的现有语言，于是他创建了自己的语言Ioke。

后来，他又厌倦了，于是就有了Seph。

他曾撰写了《JRuby实战》（Practical JRuby on Rails）一书，还是Using JRuby的合著者。

他在无数的会议上发言，为很多开源项目贡献过代码。

他还是JSR292的专家组成员之一。

他热衷实现程序设计语言、正则表达式引擎以及寻找更好的YAML解析器的实现方式。

#### Brian Blignaut

Brian曾在ThoughtWorks担任了3年多的首席咨询师。

在此期间，他参与交付了很多知名公司的定制项目，包括大型的面向终端客户的网站和实时流计算平台。

他曾做过很多JavaScript测试方面的演讲，现在在伦敦做独立咨询师。

#### James Bull

James是一个有QA背景的敏捷软件开发者。

他在ThoughtWorks做了很多测试自动化方面的工作。

他坚信，整个团队都能从中受益的测试套件才是好的测试套件。

James没在摆弄电脑的时候，肯定是在摆弄爱车。

#### Neal Ford

Neal Ford在ThoughtWorks担任软件架构师、Meme Wrangler。

他还设计和开发过很多应用程序，给很多杂志写过文章，做过很多视频或者DVD演讲，还曾出版、编辑、参与过8本涉及多种题材的书。

他专注于设计和构建大型的企业应用程序。

他还是一名在国际上颇受欢迎的演讲者，曾在300多个开发者会议上做过2000多次演讲。

他的网站是<http://nealford.com>，也可以通过邮件[nford@thoughtworks.com](mailto:nford@thoughtworks.com)联系他，他欢迎各种反馈意见。

#### Martin Fowler

Martin自称作家、演讲者……其实他就是一个在软件开发这个话题上喋喋不休的博学者。

Martin从20世纪80年代中期就开始在软件行业工作了，也是在那时他接触到了面向对象的“新概念”。

## <<软件开发与创新>>

他在90年代的大多数时间是一名咨询师和培训师，帮助人们开发面向对象系统，专注于企业应用程序。  
他于2000年加入ThoughtWorks。

他最大的兴趣是，找出一种设计软件的方式，尽可能提高开发团队的生产率。  
在寻找的过程中，他努力找出好的软件设计模式，以及支持这种模式的开发流程。  
Martin成了敏捷方法的狂热追随者，还关注演进式软件设计。

Luca Grulla

Luca曾在ThoughtWorks工作过4年。

作为首席咨询师，他帮助客户采用敏捷和精益的方法交付高质量的软件，目前在伦敦的Forward做高级程序员。

他现在的工作是，试用新的语言和新的技术，每天向产品环境提交几个新特性。

他还是全球IT社区的活跃成员，经常在国际性的会议上发言，还是Italian Agile Day、EuroClosures等欧洲会议的委员会成员。

Alistair Jones

Alistair Jones既是开发人员与技术负责人，又是架构师及教练。

他组建出来的团队具有良好的决策能力，也能够产出优良的软件。

他喜欢向人们展示，与老式的交付方式相比，敏捷方法更需要（且更能形成）严明的纪律。

Aman King

Aman King是一个应用程序开发人员。

作为分布式团队的一员，他构建过很多复杂的商业应用。

TDD是他的阳光和空气，他做起重构来就好像和代码有仇一样。

Patrick Kua

Patrick Kua在ThoughtWorks是一名活跃的多面手，他不喜欢被人贴上标签。

Patrick多数时间都在领导技术团队，培训敏捷和精益方法。

有时，他也会拯救团队于水火之中。

Patrick热衷研究学习之道和持续改进，也乐于帮助他人，激起他们在这些领域的兴趣。

Marc McNeill

Marc关注把多种不同风格的团队团结起来，精诚一致地打造非凡的用户体验。

Marc拥有人因工程专业的博士学位，在ThoughtWorks工作的7年间，他把设计思维和精益创业介绍给了世界各地的客户团队。

他做事迅速而且注重成效，通过不断地试错帮助团队把想法转化成成功的产品。

他是《当用户体验设计遇上敏捷》（Agile Experience Design）一书的作者之一（另一合作者是Lindsay Ratcliffe）。

他的Twitter账号是@dancingmango。

Julio Maia

Julio Maia在ThoughtWorks做技术咨询师已经5年了。

他在集成、自动化、运营、测试基础和应用开发等领域帮助客户解决问题，构建软件解决方案。

Mark Needham

## <<软件开发与创新>>

Mark Needham是ThoughtWorks的一名软件开发人员。

在ThoughtWorks工作的6年中，他使用敏捷方法帮客户解决问题，在项目中用过C#、Java、Ruby，还有Scala。

Sam Newman

Sam Newman是ThoughtWorks的技术咨询师，在ThoughtWorks工作8年有余。

他在很多公司工作过，始终致力于利用技术扩展IT影响力。

Rebecca Parsons

Rebecca Parsons是ThoughtWorks的CTO，她已经想不起自己接触技术有多长时间了。

她对各种技术，尤其是程序设计语言富有热情。

她在莱斯大学取得了计算机科学的博士学位，其间，她专攻语言语义学和编译器。

她还做过进化计算和计算生物学方面的工作。

Cosmin Stejerean

Cosmin Stejerean是有8年多经验的专业软件开发者。

他现在在Simple做运维工程师，生活在得克萨斯州达拉斯市。

他之前是ThoughtWorks的首席咨询师和培训师。

## &lt;&lt;软件开发与创新&gt;&gt;

## 书籍目录

第1章 引言	1
第一部分 语言	
第2章 最有趣的语言	4
2.1 为什么语言很重要	5
2.2 一些有趣的语言	5
2.2.1 Clojure	5
2.2.2 CoffeeScript	10
2.2.3 Erlang	14
2.2.4 Factor	18
2.2.5 Fantom	21
2.2.6 Haskell	26
2.2.7 Io	30
2.3 总结	33
第3章 面向对象程序设计：对象优于类	34
3.1 对象优于类	35
3.2 类关注与对象关注	36
3.2.1 角色的角色	36
3.2.2 职责分离	42
3.2.3 测试的角度	45
3.2.4 代码库里的线索	46
3.3 “对象关注”的语言	47
3.3.1 Ruby	47
3.3.2 JavaScript	53
3.3.3 Groovy	56
3.3.4 Scala	58
3.4 要点回顾	58
3.5 总结	59
第4章 使用面向对象语言进行函数式编程	60
4.1 集合	60
4.1.1 转换思维	60
4.1.2 拥抱集合	63
4.1.3 勿忘封装	64
4.1.4 惰性求值	65
4.2 “一等公民”和高阶函数	67
4.3 状态最小化	69
4.4 其他理念	70
4.5 总结	73
第二部分 测试	
第5章 极限性能测试	76
5.1 问题描述	76
5.1.1 分离性能测试的传统方式	76
5.1.2 极限编程和敏捷软件开发	77
5.1.3 分离性能测试的不足	78
5.2 另辟蹊径	78
5.2.1 独立的多功能团队	79

## &lt;&lt;软件开发与创新&gt;&gt;

5.2.2	描述需求	80	
5.2.3	设定计划与排定优先级	81	
5.2.4	实现性能故事	82	
5.2.5	展示与反馈	83	
5.3	极限性能测试实践	83	
5.3.1	性能负责人	83	
5.3.2	自动化部署	84	
5.3.3	自动化分析	85	
5.3.4	结果仓库	85	
5.3.5	结果可视化	86	
5.3.6	自动化测试流程	86	
5.3.7	健全性测试	87	
5.3.8	持续性能测试	88	
5.3.9	规范的性能提升	88	
5.4	这对我们有何帮助	89	
5.4.1	更好的性能	89	
5.4.2	更低的复杂度	89	
5.4.3	更高的团队效率	90	
5.4.4	更合理的优先级排定	90	
5.4.5	开启持续交付	90	
5.5	总结	90	
第6章	测试驱动JavaScript	91	
6.1	JavaScript的复兴	91	
6.2	当前JavaScript的处理方式与问题	92	
6.3	分离关注点	92	
6.4	测试方式	100	
6.4.1	倾向于交互测试, 而非集成测试	100	
6.4.2	在具体用例中使用HTML夹具编写集成测试	100	
6.4.3	使用验收测试验证所有组件的集成	101	
6.5	持续集成	101	
6.6	工具	101	
6.6.1	单元测试	102	
6.6.2	语法检查	102	
6.6.3	mock框架	102	
6.7	总结	102	
第7章	构建更好的验收测试	103	
7.1	快速测试	103	
7.1.1	基于用户行程的测试	103	
7.1.2	并行执行测试集	104	
7.1.3	考虑使用多种测试驱动器	105	
7.1.4	将测试分开运行	107	
7.1.5	等待页面元素显示时要小心	107	
7.2	有弹性的测试	107	
7.2.1	单独选择页面元素	108	
7.2.2	等待页面元素显示时要小心(再次强调)	109	
7.2.3	在测试中设置测试依赖的数据	110	
7.2.4	测试集成点	110	

## &lt;&lt;软件开发与创新&gt;&gt;

7.3	易于维护的测试	111	
7.3.1	使用页面模型	111	
7.3.2	结构一致的测试集	112	
7.3.3	测试代码产品代码一视同仁	113	113
7.3.4	切勿受限于工具	113	
7.4	付诸实践	114	
7.4.1	一地团队	114	
7.4.2	维护测试, 人人有责	115	115
7.4.3	故事启动	115	
7.4.4	结对测试开发	115	
7.4.5	故事展示	116	
7.5	总结	116	
第三部分 软件开发问题			
第8章 现代Java Web应用 118			
8.1	过去的状况	118	
8.1.1	有状态的服务器	119	
8.1.2	依赖容器	119	
8.1.3	违反HTTP规范	120	
8.2	无状态服务器	120	
8.2.1	集群	120	
8.2.2	cookie救世	121	
8.2.3	区分用户特定的数据	121	
8.2.4	安全cookie	122	
8.3	容器是可选的	123	
8.3.1	容器外测试	123	
8.3.2	我们真的需要容器吗	125	
8.4	按新鲜程度分区	125	
8.4.1	缓存: 可扩展网站的秘密武器	125	125
8.4.2	选择缓存的内容	126	
8.4.3	按新鲜程度分区简介	126	
8.4.4	反向代理和内容发布网络简介	128	128
8.5	POST重定向到GET	129	
8.6	总结	130	
第9章 驾驭集成难题 131			
9.1	持续集成方法	132	
9.1.1	稳定基准	132	
9.1.2	集成stub	133	
9.1.3	构建流水线	134	
9.1.4	监控器	134	
9.2	定义集成契约	135	
9.3	度量和可见性	135	
9.4	总结	136	
第10章 实践中的特性开关 137			
10.1	简单特性开关	138	
10.2	可维护的特性开关	138	138
10.2.1	依赖注入	139	
10.2.2	注解	140	

## &lt;&lt;软件开发与创新&gt;&gt;

10.3	分离静态资源	141
10.4	阻止意外泄露	142
10.5	运行时开关	142
10.6	不兼容依赖	143
10.7	特性开关的测试	143
10.8	删除完成特性的开关	144
10.9	总结	144
第11章	交付创新	145
11.1	价值流向	146
11.2	新方法	147
11.2.1	协作文化	147
11.2.2	敏捷产品调研与发现	149
11.2.3	快速启动	153
11.2.4	持续设计, 持续交付	155
11.3	总结	156
第四部分	数据可视化	
第12章	一图胜千言	158
12.1	闻闻咖啡	158
12.2	可视化设计原则	159
12.3	可视化设计流程	160
12.3.1	定义领域任务	160
12.3.2	任务抽象	161
12.3.3	数据抽象	161
12.3.4	可视化编码	163
12.3.5	评估与完善	167
12.4	可视化设计模式	168
12.4.1	探索随时间变化的数据	168
12.4.2	探索相关性	170
12.4.3	探索层次与“局部到整体”关系	170
12.4.4	探索连结和网络	172
12.5	工具和框架	173
12.5.1	可视化程序库	173
12.5.2	图型化工具	174
12.6	总结	174
参考文献		176
索引		178

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>