O'REILLY®

TURING 图灵程序设计丛书

Git团队协作

掌握Git精髓，解决版本控制、工作流问题，实现高效开发

Git for Teams

[加] Emma Jane Hogbin Westby 著

童仲毅 译

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

Git团队协作

Git for Teams

[加] Emma Jane Hogbin Westby 著
童仲毅 译

安徽师范大学图书馆
藏书

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc.授权人民邮电出版社出版

人民邮电出版社
北 京

图书在版编目（CIP）数据

Git团队协作 / （加）艾玛·简·霍格宾·韦斯特比著 ; 童仲毅译. -- 北京 : 人民邮电出版社, 2017.6
（图灵程序设计丛书）
ISBN 978-7-115-45467-6

Ⅰ. ①G… Ⅱ. ①艾… ②童… Ⅲ. ①软件工具－程序设计 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字(2017)第084097号

内 容 提 要

本书是一本软件团队协作指南，采用以人为本的方式讲解版本控制，强调如何利用 Git 促进团队协作。第一部分介绍如何创建一个优秀的团队、如何构建工作流等。第二部分从实践的角度学习 Git 命令。第三部分介绍如何在 GitHub、Bitbucket 和 GitLab 平台上托管项目。

本书适合软件开发人员和项目管理人员阅读。

◆ 著 [加] Emma Jane Hogbin Westby
译 童仲毅
责任编辑 岳新欣
执行编辑 赵瑞琳
责任印制 彭志环

◆ 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
北京市昌平百善印刷厂印刷

◆ 开本：800×1000 1/16
印张：16.75
字数：396千字 2017年6月第1版
印数：1－3 500册 2017年6月北京第1次印刷
著作权合同登记号 图字：01-2017-0780号

定价：69.00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

版权声明

© 2015 by Emma Jane Hogbin Westby.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Posts & Telecom Press, 2017. Authorized translation of the English edition, 2015 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版，2015。

简体中文版由人民邮电出版社出版，2017。英文原版的翻译得到 O'Reilly Media, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有，未得书面许可，本书的任何部分和全部不得以任何形式重制。

O'Reilly Media, Inc.介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 *Make* 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过图书出版、在线服务或者面授课程，每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——*Wired*

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——*Business 2.0*

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——*CRN*

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——*Irish Times*

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野，并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去，Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——*Linux Journal*

试读结束，需要全本PDF请购买 www.ertongbook.com

Johannes Schindelin序

在 Git 面世之前，Linux 内核开发多年来一直使用专有版本控制系统 BitKeeper，并取得了巨大的成功。但存在一个问题：一些 Linux 开发者反对使用专有性质的版本控制系统，随即开始了漫长的口水战。正是因为这场冲突，授权给 Linux 开发者免费使用的 BitKeeper 许可证被吊销，于是 Git 应运而生。Linus Torvalds 花了两周时间，放下了 Linux 的工作，想要寻找一个替代 BitKeeper 的方案。由于没能找到一个满意的替代工具，他干脆自己写了第一个非常原始的版本，也就是我们现在所说的 Git：一个用 Unix 风格编写的 shell 脚本拼接起来的小程序。具有讽刺意味的是，Git 的分布式是使用 rsync 实现的，而这个工具的作者恰好就是那个将 BitKeeper 推下历史舞台的 Linux 开发者。

就我个人而言，我最初痴迷于 Git 简洁的数据结构，随后参与了 Git 的移植工作，后来又做了越来越多的改进，包括“交互式变基”（抱歉，这个名字有些晦涩）的发明，以及最后对 Git 的 Windows 移植版本的维护。10 年来，不论是作为跨学科项目的专职程序员，还是高度分布式开源项目的负责人，从事生命科学研究的我几乎每天都要用到 Git。

在巴黎举行的庆祝 Git 十周年的 Git Merge 大会上，我第一次见到了 Emma，她分享了一场名为“教人学懂 Git”（https://youtu.be/xYhHi8yK-Is）的激情演讲。这个演讲给我留下了很深的印象，它展现了 Emma 广泛的技能以及在教学和项目管理中的丰富经验。

这本书视角独特，它强调了 Git 如何能够促进团队协作，让我收获颇丰。这本书讲的是那么简单明了，而多年来，我一直沉溺于技术细节，教授 Git 时总是从头开始面面俱到，这可能是最令人受挫的教学方式之一了。这本书重点介绍工作流和角色之间的沟通，引导读者理解实际项目中遇到的真实需求。了解这些知识后，你会学到有趣的部分：如何使用 Git 来支持你的需求。

正如 Emma 的演讲一样，她的写作风格也令人愉悦，使这本书兼具教育性和趣味性。这本书给我的日常工作带来了有价值的见解。不论你在日常工作中担任什么角色，本书都不仅仅是一本手册。无论是探索团队协作的不同方式，还是探索现代版本控制系统帮助推进项目的方法，就让这本书来激发你释放 Git 的全部潜能，为你的工作提供支持吧。

Johannes Schindelin 博士
Windows 端 Git 维护者
2015 年 8 月于德国科隆市

Mark Atwood序

版本控制的重要性怎么强调都不为过。

我认为它的重要性不亚于黑板和书本的发明，因为它将众人的力量聚集在一起，从而创造了更大的价值。

在我的职业生涯中，我看到在软件开发中，版本控制系统从最初遭到抵触到现在遍地开花，也看到基础技术带来的大跃进，其中每一次飞跃都提升了我们创造的工作价值，加快了创造的速度。我们正在以更快的速度，和更多的人一起，完成更多的工作。

Git 带来的最近一次飞跃，对我们的工作流几乎没有作出任何限制。因此，我们需要探索和分享适合自己及团队的工作流，而不是使用以往为机器设计的难用的工作流。这本书介绍了其中一些工作流。我相信你在未来会遇到更多的工作流。

教育的重要性和难度也不言而喻。所谓的教育不只是简单地死记硬背或反复训练，而是更深入的教化：如何以特定方式思考，理解为何要如此思考，以及如何告诉别人你的想法。

正确地使用版本控制系统就是一种思考方式：用精益求精的软件开发所要求的深度和严谨来建模、记忆、交流。如果没有那样的理解，Git 不过是使用一些死记硬背、充满未知危险的“神奇咒语”。有了那样的理解，Git 会变得几乎无法察觉，留给你复杂的命令背后的模式，也就是软件的魔力根源。

这本书会帮助你提升自己对 Git 的理解，并学以致用。

Mark Atwood
惠普公司开源主管
2015 年 8 月于华盛顿州西雅图市

献给 Joe Shindelar。谢谢你！

前言

在将近二十年里，我在个人和团队开发中或多或少都采用了分布式的工作方式。我的第一份有报酬的 Web 开发工作是在 20 世纪 90 年代中期。那时，我维护文件版本的方式只是通过修改文件名来标记一个新的版本。我的工作区堆满了这些文件，它们的扩展名很不寻常，像 v4.old-er.bak 这样的名称到处都是。记录我的工作可不是件容易的事情。在一个对我来说很有挑战的项目中，我不得不用上了之前修改论文时的方法：把要修改的 Perl 脚本打印出来，将这些纸装到活页夹里，然后在脚本上用不同颜色的笔做记号，最后把改动转录回文本编辑器。（要是有照片可以分享就好了。）我通过翻阅活页夹来找到之前的脚本，从而记录版本。我完全不知道该如何搭建一个真正的版本控制系统（version control system，VCS），而仅仅执着于让正确的内容不会因为重构失败而丢失。

当开始和其他开发者一起工作时，不管是做开源项目还是客户的项目，我都不是第一个加入的开发者。在我加入时总是已经有了某个版本控制系统，一般来说是并发版本系统（concurrent versions system，CVS）。它不是最易用的系统，但和我写满了改动的活页夹比起来，它的可扩展性对于分布式工作团队来说显然强多了。很快我就开始对提交信息重视起来，并珍视这种能够审阅其他同事的工作的便利性。它促使我观察其他人提交到仓库中的代码。我可不想让他们认为我在偷懒！

与此同时，我在几个不同的社区大学里教授 Web 开发。2004 年在汉博学院由 Bernie Monette 设计的一个为期一年的项目中，我第一次有机会教授版本控制。整个班级被分成了几组。在第一个学期中，学生草拟了一个网站开发计划。在第二个学期中，团队被打乱了，新的团队被要求构建上一个团队所描述的网站。在第三个也是最后一个学期中，分组被再次打乱，最终的任务是对建好的网站进行 bug 修复和质量保证。每个团队都被强制要求使用版本管理来记录他们的工作。这些先前没有编程经验的学生并没有对使用版本控制感到兴奋，反而觉得这妨碍了他们的工作。但事情也变得更简单了，因为他们从来都没有意外地覆盖其他同学的工作。这件事在很大程度上教育了我，使我知道如果一个工具看上去并非必需，应该如何激励人们去使用它。

在那门课后的十年里，我学到了很多版本控制的教学方法，以及在成人教育中获得的最佳实践。这本书正是我学到的精华，讲述通过版本控制进行有效协作的方法。在整本书中，我都鼓励你因地制宜。不会有“Git 警察”突然出现，告诉你“你做错了”。也就是说，我

会尽我所能地告诉你“Git 风格”的工作方式，当你想要开始在团队中实践，或是获得自我提升时，给你一些指导。使用“通行”的工作方式有助于你和其他之前使用过类似技术的人找到共同语言。

这本书不是写给所有人的，而是写给那些热爱提前计划，然后遵循明确线路的人。我希望本书至少有助于弥补当前 Git 资料中的空缺。与其说这是一本软件指南，不如说它是一本团队协作指南。如果你的团队觉得书中内容有困惑之处，我希望你能够发送电子邮件至 emma@gitforteams.com 来告诉我；如果你觉得它有用的话，我希望你能让全世界都知道。

致谢

几年前，在布拉格一个墓地边上的小酒馆里，我向 Carl Wiedemann 请教了很多关于 Git 的问题。谢谢你，Carl。你的热情激励我化挫折为动力，避免别人重复我在学习 Git 时所走过的弯路。

和 Joe Shindelar 一起工作的时光是我宝贵的财富，那是我在十年的自由职业之后的第一份工作。Joe，你对卓越的追求提高了我在工作上的自我要求。我很感激你的耐心和领导。这本书起源于我们关于领导力和团队结构的谈话，以及我们为 Drupalize.Me 团队创建的 Git 文档。谢谢你。

O'Reilly 找到了杰出的 Christophe Portneuve 作为我的技术审稿人之一。Christophe，谢谢你在我编写前几章时的耐心。你的反馈是非常宝贵的。我很感谢我们在 Git Merge 大会上的讨论，这场对话帮助我厘清了书中用到的概念，这让我有一个崇高的目标，那就是转变人们学习 Git 的方式。我希望你参与的这本书会让你引以为荣。

Bernie Monette、Martin Poole 和 Drew McLelland：你们为我提供了一个平台，让我通过你们的项目来完善我对版本控制的理解。

Lorna Jane Mitchell，感谢你不倦的鼓励。谢谢你和我分享你自己的 Git 工作。这激励着我要对自己提出更高的要求。

推动我完成本书的“燃料”来自 200 Degrees Coffee，一家诺丁汉郡的烘焙坊。我首选的饮料是这家烘焙坊或者司法博物馆画廊前的 Divine Coffee 制作的馥芮白。谢谢 200 Degrees Coffee 和 Divine Coffee 为我提供了休憩的去处，让我想待多久就可以待多久。

致 O'Reilly 大家庭：你们对我所有的请求（以及错过的截稿期限）都处理得非常棒。谢谢 Rachel、Heather、Robert、Colleen、Brian、Josh、Rebecca、Kim，还有数不清的幕后英雄，是你们让这本书的出版成为了现实。

致 Git 核心社区：谢谢你们邀请我参加 2015 年的 Git Merge 大会。你们接纳了我在台上关于探索 Git 教学新方法的激烈演说。你们真心听取了我的建议，然后改进了 Git 的使用体验。我期待在这个优秀的社区里参加更多活动，这里有你们一直以来默默的奉献。

我也要感谢我的审稿人：Diane Tani、Novella Chiechi、Amy Brown、Blake Winton、Stuart Langridge、Stewart Russell、Dave Hammond、John Wynstra、Chris Tankersley、Mike Anello、Piotr Sipika、Nancy Deschenes、Robert Day、Dave Hammond、Sébastien Simard、

Tobias Hiep、Nick Gard、Christopher Maneu、Johannes Schindelin、Edward Thomson、mattj. sorenson、Douwe Maan、Sytse Sijbrandij、Rob Allen、Steven Pears、Laura Lemay。你们的反馈非常宝贵。

致我的伴侣 James Westby：感谢你耐心地等待我完成“最后一件事”。若是少了你的支持和鼓励，这本书也就不会面世了。

引言

本书采用以人为本的方式讲解版本控制。我不想从 Git 的历史讲起，而是在一开始先概览各式各样的团队协作方法。接下来，我们会绕回到 Git 命令，确保你在敲下命令键时总是知道为什么。通过使用特定的工作流，有时你可以节省自己未来的时间（并减少困惑）。这些说明会给你一个宏观的理解，告诉你当下的工作是如何影响到未来的工作的，也希望你能清楚为什么有些人如此执着于他们的 Git 方法论。

第一部分主要面向管理者、技术团队的负责人、首席技术官、项目经理，以及需要制定团队工作流的技术型项目经理。

*优秀的技术源自优秀的团队。*在第 1 章中，你会了解为什么要创建一个优秀团队。学完这一章，你将能分清团队中的所有角色，组织富有成效的会议，通过关键词识别出和团队脱节的成员，并使用策略来培养团队成员间的认同和信任。

*尽早为项目设定预期。*在第 2 章中，你会学到用于允许和拒绝访问 Git 仓库的不同权限策略。是否应该允许团队成员跳过评审直接将工作存至仓库？这是不是更像一个信任和被信任的问题？这两种方法各有其优点，这一章将详细介绍它们。

*带着清晰的目标工作。*在 Git 中，你将会使用分支来分离不同的工作。第 3 章向你展示如何使用分支来隔离团队中运用的不同构想。当然，你还需要知道如何将分散的工作拼成一个完整的软件。这一章介绍了一些常见的分支策略，其中包括 GitFlow。

*记录有助于日后工作的文档。*第 4 章是第一部分中所有概念的汇总。你会学到如何创建自己的文档，并浏览一个简单的软件产品的创建和部署的全过程。

第二部分主要面向开发者。在这一部分中，你将会学到 Git 命令究竟是如何工作的（终于讲到这儿了）。如果你很着急，希望立刻开始编写代码，只需要从第二部分开始看起，看完之后再回到第一部分。

*用实战技能武装自己。*第 5 章介绍分布式版本控制的所有基础概念。在这一章中你会学到如何创建仓库，以及通过提交、分支和标签在本地记录你对文件的更改。

*学会从错误中恢复。*第 6 章讲解如何浏览历史版本，包括如何修复提交、从时间线上移除

试读结束，需要全本PDF请购买 www.ertongbook.com

提交以及变基。

*和团队成员协同工作。*现在你已经对浏览自己的仓库的历史游刃有余，是时候和别人一起协作了。第 7 章将告诉你如何跟踪远端更改，将代码上传至一个共享仓库，并将其他成员的更新同步到你的本地仓库。

*通过同行评审，分享出色完成工作时的荣耀和责任。*在第 8 章中，你将会学习在团队中实践代码评审的流程。我们还会讲到在常见的评审方法中使用的命令，以及为自己的团队定制时的一些建议。

*探索项目历史，寻求问题解决之道。*在第 9 章中，你将学会用一些高级的 Git 方法来跟踪 bug。不过，不要害怕！这些命令不会比之前的命令更难学。

第三部分是最后一部分，介绍一些市面上流行的代码托管系统。这部分内容既适合管理者也适合开发者。

*通过开放的协作促进社区的成长。*第 10 章讲解在 GitHub 上建立和维护一个开源项目的方法。

*想要编写优良的代码，团队必须拥有自己的仓库。*在第 11 章中，你将学习如何在私有仓库中协作。这一章尤其适用于那些希望建立私有仓库，但没有充足资金购买 GitHub 上私有仓库的团队。

*良好的约束可以营造更好的氛围。*在第 12 章中，你将学习如何托管你自己的 GitLab 实例，并在上面运行项目。这对于防火墙内无法接触到公共互联网的开发者来说尤为有用。

本书不是面向所有人的。对于喜欢自己折腾和探索的读者来说，阅读本书未免会感到沮丧。反之，本书适合有些畏惧未知事物的读者。

补充资料和更清晰的流程图可以在本书的官方网站（http://gitforteams.com/）上找到。

排版约定

本书使用了下列排版约定。

- 楷体
  表示新术语或重点强调的内容。
- 等宽字体（`constant width`）
  表示程序片段，以及正文中出现的变量、函数名、数据库、数据类型、环境变量、语句和关键字等。
- 加粗等宽字体（**`constant width bold`**）
  表示应该由用户输入的命令或其他文本。
- 等宽斜体（*`constant width italic`*）
  表示应该由用户输入的值或根据上下文确定的值替换的文本。

该图标表示提示或建议。

该图标表示一般注记。

该图标表示警告或警示。

代码用例

本书的补充资料（示例代码、练习等）可以从网站 http://gitforteams.com 下载。

本书是为了帮你做好工作。一般来说，你可以在程序和文档中使用本书的代码。除非你使用了很大一部分代码，否则无需联系我们获得许可。例如，使用本书的几段代码写一个程序是不需要许可的。销售或分发 O'Reilly 书中示例的光盘（CD-ROM）是需要许可的。通过引用本书和示例代码来回答问题是不需要许可的。把本书中大量的示例代码并入到你的产品文档中是需要许可的。

我们赞赏但不强制要求注明信息来源。信息来源通常包括书名、作者和国际标准书号（ISBN）。例如："*Git for Teams* by Emma Jane Hogbin Westby (O'Reilly). Copyright 2015 Emma Jane Hogbin Westby, 978-1-491-91118-1."

如果你觉得对示例代码的使用超出了正当引用或这里给出的许可范围，请随时发送电子邮件到 permissions@oreilly.com，与联系我们。

Safari® Books Online

Safari Books Online（http://safaribooksonline.com/）是应运而生的数字图书馆。它同时以图书和视频的形式出版世界顶级技术和商务作家的专业作品（https://www.safaribooksonline.com/explore/）。

技术专家、软件开发人员、Web 设计师、商务人士和创意专家等，在开展调研、解决问题、学习和认证培训时，都将 Safari Books Online 视作获取资料的首选渠道。

Safari Books Online 为企业（https://www.safaribooksonline.com/enterprise/）、政府机构（https://www.safaribooksonline.com/government/）、教育机构（https://www.safaribooksonline.com/academic-public-library/）和个人读者提供了一系列的产品组合和价格体系（https://www.safaribooksonline.com/pricing/）。

会员可在一个支持完全搜索的数据库中访问数以千计的图书、培训视频和尚未发行的书稿。发行这些内容的是 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及其他数百家发行商（https://www.safaribooksonline.com/our-library/）。要想了解 Safari Books Online 的更多信息，请访问我们的网站（http://www.safaribooksonline.com/）。

联系我们

请把对本书的评价和问题发给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
奥莱利技术咨询（北京）有限公司

O'Reilly 的每一本书都有专属网页，你可以在那儿找到本书的相关信息，包括勘误表[1]、示例代码以及其他信息。本书的网站地址是：http://shop.oreilly.com/product/0636920034520.do

对于本书的评论和技术性问题，请发送电子邮件到：bookquestions@oreilly.com

要了解更多 O'Reilly 图书、培训课程、会议和新闻的信息，请访问以下网站：
http://www.oreilly.com

我们在 Facebook 的地址如下：http://facebook.com/oreilly

请关注我们的 Twitter 动态：http://twitter.com/oreillymedia

我们的 YouTube 视频地址如下：http://www.youtube.com/oreillymedia

电子书

扫描如下二维码，即可购买本书电子版。

注 1：本书中文版的勘误，可到 http://www.ituring.com.cn/book/1779 查看和提交。——编者注

目录

Johannes Schindelin 序 ····· xi
Mark Atwood 序 ····· xii
前言 ····· xiii
引言 ····· xvii

第一部分　制定工作流

第 1 章　团队作战 ····· 2
1.1　团队成员 ····· 2
1.2　思维策略 ····· 4
1.3　团队会议 ····· 6
1.3.1　项目启动 ····· 7
1.3.2　追踪进展 ····· 7
1.3.3　培养同理心 ····· 9
1.3.4　回顾 ····· 9
1.4　Git 中的团队协作 ····· 10
1.5　小结 ····· 11
第 2 章　命令与控制 ····· 12
2.1　项目治理 ····· 12
2.1.1　版权和贡献者协议 ····· 13
2.1.2　分发许可 ····· 14
2.1.3　领导力模型 ····· 15
2.1.4　行为守则 ····· 15

2.2 访问模型……16
2.2.1 适合分散贡献者仓库的模型……18
2.2.2 适合并列贡献者仓库的模型……20
2.2.3 共同维护的模型……22
2.2.4 自定义访问模型……24
2.3 小结……25

第 3 章 分支策略……26

3.1 理解分支……26
3.2 挑选约定……27
3.3 几种约定……28
3.3.1 主线分支开发……28
3.3.2 功能分支部署……30
3.3.3 状态分支……32
3.3.4 计划部署……35
3.4 更新分支……40
3.5 小结……43

第 4 章 工作流……45

4.1 初识工作流……45
4.1.1 记录工作过程……46
4.1.2 记录编码的决定……46
4.2 工单进展……47
4.3 基本工作流……49
4.3.1 使用同行评审的可信开发者……50
4.3.2 需要质量保证团队的不可信开发者……51
4.4 根据计划发布软件……52
4.4.1 发布稳定版本……52
4.4.2 正在进行的开发……53
4.4.3 发布后的补丁……53
4.5 非软件项目中的协作……54
4.6 小结……55

第二部分 在工作流中使用命令

第 5 章 单人团队……58

5.1 基于 issue 的版本控制……59
5.2 创建本地仓库……60
5.2.1 克隆已有的项目……62

5.2.2 将已有的项目迁移至 Git ····· 63
5.2.3 初始化空项目 ····· 65
5.2.4 查看历史记录 ····· 65
5.3 使用分支工作 ····· 66
5.3.1 列出分支 ····· 66
5.3.2 更新远程分支列表 ····· 67
5.3.3 使用不同的分支 ····· 67
5.3.4 创建新的分支 ····· 68
5.4 在仓库中添加更改 ····· 70
5.4.1 在仓库中添加部分文件修改 ····· 72
5.4.2 提交部分更改 ····· 73
5.4.3 从暂存区移除文件 ····· 74
5.4.4 编写扩展提交消息 ····· 74
5.4.5 忽略文件 ····· 75
5.5 使用标签 ····· 76
5.6 连接远程仓库 ····· 77
5.6.1 创建新的项目 ····· 78
5.6.2 添加第二个远程连接 ····· 78
5.6.3 推送你的更改 ····· 79
5.6.4 分支维护 ····· 80
5.7 命令指南 ····· 81
5.8 小结 ····· 82

第 6 章 回滚、还原、重置和变基 ····· 83

6.1 最佳实践 ····· 83
6.1.1 描述问题 ····· 84
6.1.2 使用分支进行试验性的工作 ····· 85
6.2 分步变基 ····· 88
6.2.1 开始变基 ····· 88
6.2.2 文件删除造成的变基中冲突 ····· 89
6.2.3 单个文件合并冲突造成的变基中冲突 ····· 92
6.3 定位丢失的工作概述 ····· 94
6.4 还原文件 ····· 97
6.5 使用提交 ····· 98
6.5.1 修补提交 ····· 99
6.5.2 使用 reset 合并提交 ····· 99
6.5.3 使用交互式变基修改提交 ····· 101
6.5.4 撤销分支合并 ····· 106
6.6 撤销共享历史记录 ····· 108
6.6.1 还原之前的提交 ····· 108

试读结束，需要全本PDF请购买 www.ertongbook.com