

Git权威指南



[Git权威指南_下载链接1](#)

著者:蒋鑫

出版者:机械工业出版社华章公司

出版时间:2011-6-27

装帧:平装

isbn:9787111349679

《Git权威指南》是Git领域的集大成之作，是一本关于Git的百科全书，在广度、深度和实战性上让同类作品望尘莫及。作者是国内顶尖的版本控制专家和咨询顾问之一，本书得到了Git官方维护者Junio C

Hamano和Teye创始人范凯（Robbin）先生等数位专家的高度认可和极力推荐，权威性毋庸置疑。

全书一共9篇，共41章和4个附录，内容几乎涵盖了Git的所有方面。第1篇介绍了版本控制工具的演变历史、Git的各种优点，以及它在3种主流操作系统中的安装与配置。第2篇和第3篇既是本书的基础，又是本书的核心，不仅介绍了Git的操作和使用，而且还讲解了Git的原理。第2篇详细讲解了个人用户如何使用Git，包括Git初始化、日常操作、暂存区、对象、重置、检出、恢复进度、历史变更、克隆、库管理等；第3篇详细讲解了Git协议和团队如何使用Git，包括Git支持的协议、冲突解决、里程碑、分支、远程版本库和补丁文件交互等。第4篇全面介绍了Git的协同模型，即它在实际工作中的使用模式，包括各种经典的Git协同模型、Topgit协同模型、子模组协同模型、子树合并、Android多版本库协同、Git与SVN协同模型等。第5篇介绍了Git服务器的架设，首先讲解了HTTP协议、Git协议、SSH协议的使用，然后讲解了Gitolite、Gitis、Gerrit等服务器的架设方法，最后还讲解了Git版本库的托管。第6篇介绍了版本库的迁移，包括如何从CVS、SVN、Hg等版本库迁移到Git，以及Git版本库整理等方面的内容。第7篇讲解了Git的其他应用，包括etckeeper、Gistore等的安装、配置和使用，以及补丁中的二进制文件和云存储等内容。第8篇介绍了Git的跨平台操作，以及它的钩子和模板、稀疏检出和浅克隆、嫁接和替换等重要特性。第9篇是附录，详细给出了Git的命令索引，以及CVS、SVN和Hg与Git的比较与命令对照，方便读者查阅。

作者简介:

蒋鑫，国内顶尖的版本控制专家和咨询顾问之一，对Subversion和Git等版本控制工具有十分深入的研究，参与了Git以及Gitis、Gitolite、Repo、Topgit、Gistore等与Git相关的开源软件的开发或创建，在大量实践中积累了丰富的经验。此外，他还是一位开源软件实践者，作为北京群英汇信息技术有限公司的创始人兼高级顾问，一直从事开源软件的定制以及面向研发团队的项目管理软件的推广和顾问咨询工作，致力于推动开源软件在中国的发展。

本书官网：<http://www.ossxp.com/doc/gotgit/>

作者博客：<http://blog.ossxp.com/>

作者微博：<http://weibo.com/gotgit/>

本书微群：<http://q.weibo.com/567527>

本书豆瓣小组：<http://www.douban.com/group/gotgit/>

目录: 前言

第1篇 初识Git

第1章 版本控制的前世和今生/ 2

1.1 黑暗的史前时代/ 2

1.2 CVS—开启版本控制大爆发/ 5

1.3 SVN—集中式版本控制集大成者/ 7

1.4 Git—Linux 的第二个伟大作品/ 9

第2章 爱上 Git 的理由/ 11

2.1 每日工作备份/ 11

2.2 异地协同工作/ 12

2.3 现场版本控制/ 13

2.4 避免引入辅助目录/ 15

2.5 重写提交说明/ 15

- 2.6 想吃后悔药/ 16
- 2.7 更好用的提交列表/ 17
- 2.8 更好的差异比较/ 18
- 2.9 工作进度保存/ 18
- 2.10 代理SVN提交实现移动式办公/ 19
- 2.11 无处不在的分页器/ 20
- 2.12 快/ 21
- 第3章 Git的安装和使用/ 22
 - 3.1 在Linux 下安装和使用 Git/ 22
 - 3.1.1 包管理器方式安装/ 22
 - 3.1.2 从源代码进行安装/ 23
 - 3.1.3 从Git版本库进行安装/ 23
 - 3.1.4 命令补齐/ 25
 - 3.1.5 中文支持/ 25
 - 3.2 在Mac OS X 下安装和使用 Git/ 26
 - 3.2.1 以二进制发布包的方式安装/ 26
 - 3.2.2 安装 Xcode/ 27
 - 3.2.3 使用 Homebrew 安装 Git/ 29
 - 3.2.4 从Git源码进行安装/ 29
 - 3.2.5 命令补齐/ 30
 - 3.2.6 其他辅助工具的安装/ 30
 - 3.2.7 中文支持/ 31
 - 3.3 在Windows 下安装和使用 Git (Cygwin篇) / 31
 - 3.3.1 安装 Cygwin/ 32
 - 3.3.2 安装 Git/ 36
 - 3.3.3 Cygwin 的配置和使用/ 37
 - 3.3.4 Cygwin 下 Git 的中文支持/ 40
 - 3.3.5 Cygwin 下 Git 访问 SSH 服务/ 41
 - 3.4 Windows 下安装和使用 Git (msysGit篇) / 45
 - 3.4.1 安装 msysGit/ 46
 - 3.4.2 msysGit 的配置和使用/ 48
 - 3.4.3 msysGit中shell环境的中文支持/ 49
 - 3.4.4 msysGit中Git的中文支持/ 50
 - 3.4.5 使用 SSH 协议/ 51
 - 3.4.6 TortoiseGit 的安装和使用/ 52
 - 3.4.7 TortoiseGit 的中文支持/ 55
- 第2篇 Git独奏
- 第4章 Git 初始化/ 58
 - 4.1 创建版本库及第一次提交/ 58
 - 4.2 思考：为什么工作区根目录下有一个 .git 目录/ 60
 - 4.3 思考：git config 命令的各参数有何区别/ 63
 - 4.4 思考：是谁完成的提交/ 65
 - 4.5 思考：随意设置提交者姓名，是否太不安全/ 67
 - 4.6 思考：命令别名是干什么的/ 68
 - 4.7 备份本章的工作成果/ 69
- 第5章 Git 暂存区/ 70
 - 5.1 修改不能直接提交吗/ 70
 - 5.2 理解 Git 暂存区 (stage) / 76
 - 5.3 Git Diff 魔法/ 78
 - 5.4 不要使用 git commit -a/ 81
 - 5.5 搁置问题，暂存状态/ 82
- 第6章 Git对象/ 83
 - 6.1 Git对象库探秘/ 83
 - 6.2 思考：SHA1 哈希值到底是什么，是如何生成的/ 88

6.3 思考：为什么不用顺序的数字来表示提交/ 90

第7章 Git 重置/ 93

7.1 分支游标master探秘/ 93

7.2 用 reflog 挽救错误的重置/ 95

7.3 深入了解git reset命令/ 96

第8章 Git 检出/ 99

8.1 HEAD 的重置即检出/ 99

8.2 挽救分离头指针/ 102

8.3 深入了解 git checkout 命令/ 103

第9章 恢复进度/ 105

9.1 继续暂存区未完成的实践/ 105

9.2 使用 git stash/ 108

9.3 探秘 git stash/ 109

第10章 Git 基本操作/ 114

10.1 先来合个影/ 114

10.2 删除文件/ 114

10.2.1 本地删除不是真的删除/ 115

10.2.2 执行 git rm 命令删除文件/ 116

10.2.3 命令git add -u快速标记删除/ 117

10.3 恢复删除的文件/ 118

10.4 移动文件/ 119

10.5 一个显示版本号的 Hello World/ 120

10.6 使用 git add -i 选择性添加/ 122

10.7 Hello World 引发的新问题/ 124

10.8 文件忽略/ 125

10.9 文件归档/ 129

第11章 历史穿梭/ 130

11.1 图形工具：gitk/ 130

11.2 图形工具：gitg/ 131

11.3 图形工具：qgit/ 135

11.4 命令行工具/ 140

11.4.1 版本表示法：git rev-parse/ 141

11.4.2 版本范围表示法：git rev-list/ 144

11.4.3 浏览日志：git log/ 146

11.4.4 差异比较：git diff/ 150

11.4.5 文件追溯：git blame/ 151

11.4.6 二分查找：git bisect/ 152

11.4.7 获取历史版本/ 156

第12章 改变历史/ 157

12.1 悔棋/ 157

12.2 多步悔棋/ 159

12.3 回到未来/ 161

12.3.1 时间旅行一/ 162

12.3.2 时间旅行二/ 167

12.3.3 时间旅行三/ 171

12.4 丢弃历史/ 174

12.5 反转提交/ 177

第13章 Git 克隆/ 179

13.1 鸡蛋不装在一个篮子里/ 179

13.2 对等工作区/ 180

13.3 克隆生成裸版本库/ 183

13.4 创建生成裸版本库/ 184

第14章 Git库管理/ 187

14.1 对象和引用哪里去了/ 187

- 14.2 暂存区操作引入的临时对象/ 189
- 14.3 重置操作引入的对象/ 191
- 14.4 Git管家: git-gc/ 193
- 14.5 Git管家的自动执行/ 196
- 第3篇 Git和声
- 第15章 Git协议与工作协同/ 200
 - 15.1 Git 支持的协议/ 200
 - 15.2 多用户协同的本地模拟/ 202
 - 15.3 强制非快进式推送/ 203
 - 15.4 合并后推送/ 207
 - 15.5 禁止非快进式推送/ 208
- 第16章 冲突解决/ 210
 - 16.1 拉回操作中的合并/ 210
 - 16.2 合并一: 自动合并/ 212
 - 16.2.1 修改不同的文件/ 212
 - 16.2.2 修改相同文件的不同区域/ 214
 - 16.2.3 同时更改文件名和文件内容/ 215
 - 16.3 合并二: 逻辑冲突/ 217
 - 16.4 合并三: 冲突解决/ 218
 - 16.4.1 手工编辑完成冲突解决/ 221
 - 16.4.2 图形工具完成冲突解决/ 221
 - 16.5 合并四: 树冲突/ 225
 - 16.5.1 手工操作解决树冲突/ 227
 - 16.5.2 交互式解决树冲突/ 228
 - 16.6 合并策略/ 230
 - 16.7 合并相关的设置/ 231
- 第17章 Git 里程碑/ 233
 - 17.1 显示里程碑/ 234
 - 17.2 创建里程碑/ 236
 - 17.2.1 轻量级里程碑/ 237
 - 17.2.2 带说明的里程碑/ 238
 - 17.2.3 带签名的里程碑/ 239
 - 17.3 删除里程碑/ 242
 - 17.4 不要随意更改里程碑/ 243
 - 17.5 共享里程碑/ 243
 - 17.6 删除远程版本库的里程碑/ 246
 - 17.7 里程碑命名规范/ 247
- 第18章 Git 分支/ 253
 - 18.1 代码管理之殇/ 253
 - 18.1.1 发布分支/ 253
 - 18.1.2 特性分支/ 256
 - 18.1.3 卖主分支/ 257
 - 18.2 分支命令概述/ 258
 - 18.3 “Hello World” 开发计划/ 259
 - 18.4 基于特性分支的开发/ 260
 - 18.4.1 创建分支 user1/getopt/ 261
 - 18.4.2 创建分支 user2/i18n/ 262
 - 18.4.3 开发者 user1 完成功能开发/ 263
 - 18.4.4 将 user1/getopt 分支合并到主线/ 264
 - 18.5 基于发布分支的开发/ 265
 - 18.5.1 创建发布分支/ 266
 - 18.5.2 开发者 user1 工作在发布分支/ 267
 - 18.5.3 开发者 user2 工作在发布分支/ 268
 - 18.5.4 开发者 user2 合并推送/ 270

- 18.5.5 发布分支的提交合并到主线/ 271
- 18.6 分支变基/ 275
 - 18.6.1 完成 user2/i18n 特性分支的开发/ 275
 - 18.6.2 分支 user2/i18n 变基/ 277
- 第19章 远程版本库/ 284
 - 19.1 远程分支/ 284
 - 19.2 分支追踪/ 287
 - 19.3 远程版本库/ 290
 - 19.4 PUSH 和 PULL 操作与远程版本库/ 292
 - 19.5 里程碑和远程版本库/ 294
 - 19.6 分支和里程碑的安全性/ 294
- 第20章 补丁文件交互/ 296
 - 20.1 创建补丁/ 296
 - 20.2 应用补丁/ 297
 - 20.3 StGit 和 Quilt/ 300
 - 20.3.1 StGit/ 300
 - 20.3.2 Quilt/ 304
- 第4篇 Git协同模型
- 第21章 经典Git协同模型/ 308
 - 21.1 集中式协同模型/ 308
 - 21.1.1 传统集中式协同模型/ 309
 - 21.1.2 Gerrit 特殊的集中式协同模型/ 310
 - 21.2 金字塔式协同模型/ 311
 - 21.2.1 贡献者开放只读版本库/ 312
 - 21.2.2 以补丁方式贡献代码/ 313
- 第22章 Topgit 协同模型/ 314
 - 22.1 作者版本控制系统的三个里程碑/ 314
 - 22.2 Topgit 原理/ 316
 - 22.3 Topgit 的安装/ 317
 - 22.4 Topgit 的使用/ 319
 - 22.5 用Topgit方式改造Topgit/ 330
 - 22.6 Topgit 使用中的注意事项/ 334
- 第23章 子模组协同模型/ 336
 - 23.1 创建子模组/ 336
 - 23.2 克隆带子模组的版本库/ 339
 - 23.3 在子模组中修改和子模组的更新/ 340
 - 23.4 隐性子模组/ 343
 - 23.5 子模组的管理问题/ 345
- 第24章 子树合并/ 347
 - 24.1 引入外部版本库/ 347
 - 24.2 子目录方式合并外部版本库/ 349
 - 24.3 利用子树合并跟踪上游改动/ 351
 - 24.4 子树拆分/ 353
 - 24.5 git-subtree 插件/ 353
- 第25章 Android 式多版本库协同/ 356
 - 25.1 关于 repo/ 357
 - 25.2 安装 repo/ 357
 - 25.3 repo和清单库的初始化/ 359
 - 25.4 清单库和清单文件/ 360
 - 25.5 同步项目/ 361
 - 25.6 建立 Android 代码库本地镜像/ 363
 - 25.7 repo 的命令集/ 365
 - 25.8 repo 命令的工作流/ 370
 - 25.9 好东西不能 Android 独享/ 371

- 25.9.1 repo+Gerrit 模式/ 371
- 25.9.2 repo 无审核模式/ 371
- 25.9.3 改进的 repo 无审核模式/ 372
- 第26章 Git 和 SVN 协同模型/ 378
- 26.1 使用 git-svn 的一般流程/ 380
- 26.2 git-svn 的奥秘/ 386
 - 26.2.1 Git 库配置文件的扩展及分支映射/ 387
 - 26.2.2 Git工作分支和 Subversion 如何对应/ 388
 - 26.2.3 其他辅助文件/ 390
- 26.3 多样的 git-svn 克隆模式/ 390
- 26.4 共享 git-svn 的克隆库/ 393
- 26.5 git-svn 的局限/ 394
- 第5篇 搭建Git服务器
- 第27章 使用 HTTP 协议/ 398
 - 27.1 哑传输协议/ 398
 - 27.2 智能 HTTP 协议/ 400
 - 27.3 Gitweb 服务器/ 401
 - 27.3.1 Gitweb的安装/ 402
 - 27.3.2 Gitweb的配置/ 403
 - 27.3.3 版本库的 Gitweb 相关设置/ 404
 - 27.3.4 即时Gitweb服务/ 405
- 第28章 使用 Git 协议/ 406
 - 28.1 Git 协议语法格式/ 406
 - 28.2 Git 服务软件/ 406
 - 28.3 以 inetd 方式配置运行/ 406
 - 28.4 以 runit 方式配置运行/ 407
- 第29章 使用 SSH 协议/ 409
 - 29.1 SSH 协议语法格式/ 409
 - 29.2 服务架设方式比较/ 409
 - 29.3 关于 SSH 公钥认证/ 411
 - 29.4 关于 SSH 主机别名/ 411
- 第30章 Gitolite 服务架设/ 413
 - 30.1 安装Gitolite/ 414
 - 30.1.1 服务器端创建专用账号/ 414
 - 30.1.2 Gitolite 的安装/升级/ 415
 - 30.1.3 关于 SSH 主机别名/ 417
 - 30.1.4 其他的安装方法/ 418
 - 30.2 管理 Gitolite/ 419
 - 30.2.1 管理员克隆 gitolite-admin 管理库/ 419
 - 30.2.2 增加新用户/ 420
 - 30.2.3 更改授权/ 422
 - 30.3 Gitolite 授权详解/ 423
 - 30.3.1 授权文件的基本语法/ 423
 - 30.3.2 定义用户组和版本库组/ 424
 - 30.3.3 版本库ACL/ 424
 - 30.3.4 Gitolite 授权机制/ 426
 - 30.4 版本库授权案例/ 427
 - 30.4.1 对整个版本库进行授权/ 427
 - 30.4.2 通配符版本库的授权/ 428
 - 30.4.3 用户自己的版本库空间/ 429
 - 30.4.4 对引用的授权：传统模式/ 430
 - 30.4.5 对引用的授权：扩展模式/ 430
 - 30.4.6 对引用的授权：禁用规则的使用/ 431
 - 30.4.7 用户分支/ 431

- 30.4.8 对路径的写授权/ 432
- 30.5 创建新版本库/ 432
 - 30.5.1 在配置文件中出现的版本库，即时生成/ 433
 - 30.5.2 通配符版本库，管理员通过推送创建/ 434
 - 30.5.3 直接在服务器端创建/ 435
- 30.6 对 Gitolite 的改进/ 435
- 30.7 Gitolite 功能拓展/ 436
 - 30.7.1 版本库镜像/ 436
 - 30.7.2 Gitweb 和 Git daemon 支持/ 438
 - 30.7.3 其他功能拓展和参考/ 439
- 第31章 Gitis 服务架设/ 441
 - 31.1 安装 Gitis/ 442
 - 31.1.1 Gitis 的安装/ 442
 - 31.1.2 服务器端创建专用账号/ 442
 - 31.1.3 Gitis 服务初始化/ 443
 - 31.2 管理 Gitis/ 443
 - 31.2.1 管理员克隆 gitolit-admin 管理库/ 443
 - 31.2.2 增加新用户/ 444
 - 31.2.3 更改授权/ 446
 - 31.3 Gitis 授权详解/ 447
 - 31.3.1 Gitis 默认设置/ 447
 - 31.3.2 管理版本库 gitosis-admin/ 447
 - 31.3.3 定义用户组和授权/ 448
 - 31.3.4 Gitweb 整合/ 449
 - 31.4 创建新版本库/ 449
 - 31.5 轻量级管理的 Git 服务/ 450
- 第32章 Gerrit 代码审核服务器/ 452
 - 32.1 Gerrit 的实现原理/ 452
 - 32.2 架设 Gerrit 的服务器/ 456
 - 32.3 Gerrit 的配置文件/ 461
 - 32.4 Gerrit 的数据库访问/ 462
 - 32.5 立即注册为 Gerrit 管理员/ 464
 - 32.6 管理员访问 SSH 的管理接口/ 467
 - 32.7 创建新项目/ 468
 - 32.8 从已有的 Git 库创建项目/ 472
 - 32.9 定义评审工作流/ 473
 - 32.10 Gerrit 评审工作流实战/ 477
 - 32.10.1 开发者在本地版本库中工作/ 477
 - 32.10.2 开发者向审核服务器提交/ 478
 - 32.10.3 审核评审任务/ 478
 - 32.10.4 评审任务没有通过测试/ 480
 - 32.10.5 重新提交新的补丁集/ 482
 - 32.10.6 新修订集通过评审/ 483
 - 32.10.7 从远程版本库更新/ 485
 - 32.11 更多 Gerrit 参考/ 486
- 第33章 Git 版本库托管/ 487
 - 33.1 Github/ 487
 - 33.2 Gitorious/ 489
- 第6篇 迁移到Git
- 第34章 CVS版本库到Git的迁移/ 492
 - 34.1 安装cvs2svn (含 cvs2git) / 492
 - 34.1.1 Linux下cvs2svn的安装/ 492
 - 34.1.2 Mac OS X 下 cvs2svn 的安装/ 493
 - 34.2 版本库转换的准备工作/ 494

- 34.2.1 版本库转换注意事项/ 494
- 34.2.2 文件名乱码问题/ 494
- 34.2.3 提交说明乱码问题/ 494
- 34.3 版本库转换/ 496
- 34.3.1 配置文件解说/ 496
- 34.3.2 运行cvs2git完成转换/ 500
- 34.4 迁移后的版本库检查/ 501
- 第35章 更多版本控制系统的迁移/ 502
- 35.1 SVN版本库到Git的迁移/ 502
- 35.2 Hg 版本库到Git的迁移/ 503
- 35.3 通用版本库迁移/ 505
- 35.4 Git 版本库整理/ 511
- 35.4.1 环境变量过滤器/ 513
- 35.4.2 树过滤器/ 513
- 35.4.3 暂存区过滤器/ 513
- 35.4.4 父节点过滤器/ 514
- 35.4.5 提交说明过滤器/ 514
- 35.4.6 提交过滤器/ 515
- 35.4.7 里程碑名字过滤器/ 516
- 35.4.8 子目录过滤器/ 516
- 第7篇 Git的其他应用
- 第36章 etckeeper/ 518
- 36.1 安装 etckeeper/ 518
- 36.2 配置 etckeeper/ 519
- 36.3 使用 etckeeper/ 519
- 第37章 Gistore/ 520
- 37.1 Gistore 的安装/ 520
- 37.1.1 软件依赖/ 520
- 37.1.2 从源码安装 Gistore/ 521
- 37.1.3 用 easy_install 安装/ 521
- 37.2 Gistore 的使用/ 522
- 37.2.1 创建并初始化备份库/ 522
- 37.2.2 Gistore 的配置文件/ 523
- 37.2.3 Gistore 的备份项管理/ 524
- 37.2.4 执行备份任务/ 525
- 37.2.5 查看备份日志/ 525
- 37.2.6 查看及恢复备份数据/ 527
- 37.2.7 备份回滚及设置/ 528
- 37.2.8 注册备份任务别名/ 529
- 37.2.9 自动备份: crontab/ 529
- 37.3 Gistore 双机备份/ 529
- 第38章 补丁中的二进制文件/ 531
- 38.1 Git 版本库中二进制文件变更的支持/ 531
- 38.2 对非 Git 版本库中二进制文件变更的支持/ 535
- 38.3 其他工具对 Git 扩展补丁文件的支持/ 536
- 第39章 云存储/ 538
- 39.1 现有云存储的问题/ 538
- 39.2 Git 式云存储畅想/ 539
- 第8篇 Git杂谈
- 第40章 跨平台操作 Git/ 542
- 40.1 字符集问题/ 542
- 40.2 文件名大小写问题/ 543
- 40.3 换行符问题/ 545
- 第41章 Git 的其他特性/ 549

41.1 属性/ 549
41.1.1 属性定义/ 549
41.1.2 属性文件及优先级/ 550
41.1.3 常用属性介绍/ 552
41.2 钩子和模板/ 557
41.2.1 Git 钩子/ 557
41.2.2 Git 模板/ 562
41.3 稀疏检出和浅克隆/ 563
41.3.1 稀疏检出/ 563
41.3.2 浅克隆/ 566
41.4 嫁接和替换/ 568
41.4.1 提交嫁接/ 568
41.4.2 提交替换/ 568
41.5 Git 评注/ 570
41.5.1 评注的奥秘/ 570
41.5.2 评注相关命令/ 573
41.5.3 评注相关配置/ 574
第9篇 附录
附录A Git 命令索引/ 576
A.1 常用的Git命令/ 576
A.2 对象库操作相关命令/ 578
A.3 引用操作相关命令/ 578
A.4 版本库管理相关命令/ 579
A.5 数据传输相关命令/ 579
A.6 邮件相关命令/ 580
A.7 协议相关命令/ 580
A.8 版本库转换和交互相关命令/ 581
A.9 合并相关的辅助命令/ 581
A.10 杂项/ 582
附录B Git 与 CVS 面对面/ 583
B.1 面对面访谈录/ 583
B.2 Git 和CVS 命令对照/ 585
附录C Git 与 SVN 面对面/ 587
C.1 面对面访谈录/ 587
C.2 Git 和SVN 命令对照/ 589
附录D Git 与 Hg 面对面/ 592
D.1 面对面访谈录/ 592
D.2 Git和Hg 命令对照/ 593
• • • • • (收起)

[Git权威指南_下载链接1](#)

标签

git

版本控制

Git

计算机

软件开发

软件工程

编程

Git官方推荐

评论

文笔是一个问题。

还算不错，但是需要对git有一定的了解，至少有一段时间的使用。有些地方讲的太细了，总得来说还算ok。

Git 的原理 作为进阶阅读和操作是估计是不错的选择。内容详实让人能知其然也知其所以然。Git 的魅力啊。不过作为 Git 上手读物 算是挑错书了如果未来需要进阶 再翻。

Android的代码管理方式？？哈哈。恩，Hg、Git这些分布式代码管理软件确实不适合把整个大项目的代码分解成小模块管理，在这一点还不如SVN灵活

: TP311.56/4181

好吧，封底推荐有我的份

有空入手一本详读。。

国内最好的 Git 教材。

大而全，从核心操作到周边应用什么都讲了。前半本仔细读过，后半本可以当手册用了。
。

manual book. git reflog show master

前三篇看了看

请自行跳过大量的废话以及无用的纯粹占用空间的文字，不过内容丰富，回到未来那个章节什么玩意啊！！！！

后面几章开始没那么有兴趣读了，当做将来的参考手册吧。

good book!

书本材质比较差，油墨很容易擦掉。作为git的中文参考书，这本书还是有可取之处的。
。

废话太多,

Git 入门!

内容相当充实，居然是本国人写的书，很难得

国内作者难得的好书，深入浅出的介绍了 Git 的原理和使用方法，从个人使用讲到群体协作，还介绍了 Github 的使用方法，非常赞

书写得很细致，但有点过分精致了。从本书其实可以发现，中文与windows的支持还是git的弱项，需要很多trick和tip才能支持。对于软件开发，还是linux+en是最好的环境。

[Git权威指南_下载链接1](#)

书评

如果你只想了解Git怎么用，不建议你买此书，pro git 是很好的关于git的介绍。如果你想成为Git高手，你的选择是正确的。对Git的各个方面介绍的很完善，让你明白git的各个命令后面发生了什么。

买过来看了以后，内容很不错，纸质很好，很斤的起翻，适合做为参考资料或床头书经常看看，讲的条理性比较强，适合有几年开发经验的，在开发方面和代码管理方面吃过堑，犯过错的开发人员。很多事情别人说千万次，不如自己吃亏一次，看书的时候，作者诙谐忠肯的语言，常让我会...

这本书基本上讲解了作为版本管理的基本操作。最关键的是每一个操作后面都告诉你这个操作的原理是什么。授之以渔才是书本最重要的东西。当然可能有些人不是很习惯，会觉得有点杂乱无章。我个人认为，这是方法不对。我推

荐先全文浏览一遍，然后再上机操作。加入自己的想法，根据...

这本书基本上没必要购买。

一本把简单内容复杂化的书不是好书。如果你需要了解git到前世今生，及各种细节请购买吧。

总体评价：1、语言组织糟糕，阅读成本太高。2、无效信息太多，信噪比太低。

结论：基本没必要看，浪费时间。我甚至要质疑那些以“如果你要成为git高手， ...

作者并没有循序渐进式的，从整体到细节处给你详细的讲解，总感觉有种在不断加内容，让整本书看起来更厚的感觉，有些地方需要细致的却是只有点到为止，平时的工作中，实战讲的又非常少，总感觉弄一些可能不会经常用的命令来给你讲解，权威指南意在指南，不是剖析，作者的定位的...

如果你想熟练掌握git，读pro

git绝对是够了，如果你吃饱了没事干还想多看点，选这本也还凑合。

世间的技术书籍，80%就是把技术文档抄一遍、再配上三言两语的讲解，有真知灼见的极少。这本书中有一些作者自己的东西，可以看出作者比较专业，但是作者的叙述不够生动，像是当顾问...

使用git，如果你只是想达到会用就行，那么这本书不适合你，网上随便搜git的各个命令，有的是。看这本书，你会明白这一切的背后究竟发生了什么，作者就像名侦探柯南一样，带领我们探索发现那一个个操作的幕后真相，作者不仅对.git目录做了深入研究，更是从git源码里挖掘，找寻依...

这本书很厚,有差不多六百也,内容很多,但是我感觉很枯燥.不适合git不怎么好的人看. 后悔,对于我这样的非技术牛人的人来说还是买版本控制之道比较靠谱. 那个比较适合我这样的想要系统一点的入门的人来看. 这本书则看的我头都大了. 有些后悔买了.

看了一点，感觉 真尼码细致 从原理上解释了为什么 git 很牛逼很快 同时还做了很多实验去证明 只不过对于想入门git的人来说，太细致了，学习曲线很高 不如<http://git-scm.com/documentation> 来的直接

一旦谈到“权威”，往往意味着严肃、原理、枯燥、刻板，不适合入门。这本书验证了这个经验。

本书并不适合IT入门者和初学者，前面几章讲了太多关于CVS和SVN的东西。确实，如果对某种东西比较熟悉的情况下，再学一个新的类似的东西时，确实会与之前学过的产生比较，这是不自觉的...

工作区、暂存区、版本库的作用、意义 SHA1码，commit tree blob master HEAD是什么 git reset 和 git checkout 干了什么 fast forward提交 是什么

主要看了第一部分和第二部分，概念解释比较详细的，值得一读。注意gitolite的安装已经修改了，但只在脚注中写明，第一次看的时候没有主要浪费了不少时间。

第一遍读这本书的时候简直就想抽他，看的云里雾里，当时一心只想取代SVN把Git用起来，要知道SVN配合乌龟是很容易用的，这个‘很容易’当然仅限于用起来，完全没有掌握整个workflow。
一圈Git速成下来虽然掌握了基本命令但始终不得要领，这时候再回过来看这本书，终于才明白作...

内容很详细，如果只是想学如何使用Git不推荐买这本书，如果想知道Git的工作原理，深入理解Git运行机制，那么强力推荐读这本书（你妹啊。。。竟然还嫌我的评论短。。。还短。。。还短。。。让你短。。。。。。。）

作为一个初学者通过这本书学习git。起点比较高。一上来就讲cat-file ls-tree等不常用的命令，使读者很迷惑。
不建议初学者通过这本书学习git。可以作为参考手册。
带深入了解git后再阅读这本书。

* 这本书能缩到一般，甚至1/3最好 *
比如gitolite配置，我按照书里面搞了一头包以后还是请G哥出山才搞定 *****
一下是垃圾 ***** * 好吧，豆瓣说我的评论太短了 * 好吧，豆瓣说我的评论太短了2
* 好吧，豆瓣说我的评论太短了3 * 好吧，豆瓣说我的评论太...

好书讲的很详细，不过初学者可能看不懂，建议初学者看pro git。

豆瓣说我的评论太短了，豆瓣说我的评论太短了，豆瓣说我的评论太短了，豆瓣说我的评论太短了，豆瓣说我的评论太短了，豆瓣说我的评论太短了

[Git权威指南_下载链接1](#)