

Git权威指南

作者：蒋鑫

Git权威指南

蒋鑫 著

ISBN: 978-7-111-34967-9

本书由“行行”整理，如果你不知道读什么书或者想获得更多免费电子书请加小编微信或QQ：491256034 小编也和结交一些喜欢读书的朋友 或者关注小编个人微信公众号id：d716-716 为了方便书友朋友找书和看书，小编自己做了一个电子书下载网站，网址：www.ireadweek.com QQ群：550338315

[目 录](#)

[前言](#)

[本书的组织](#)

[适用读者](#)

[排版约定](#)

[在线资源](#)

[致谢](#)

[第1篇 初识Git](#)

[第1章 版本控制的前世和今生](#)

[1.1 黑暗的史前时代](#)

[1.2 CVS——开启版本控制大爆发](#)

[1.3 SVN——集中式版本控制集大成者](#)

[1.4 Git——Linus的第二个伟大作品](#)

[第2章 爱上Git的理由](#)

[2.1 每日工作备份](#)

[2.2 异地协同工作](#)

[2.3 现场版本控制](#)

[2.4 避免引入辅助目录](#)

[2.5 重写提交说明](#)

[2.6 想吃后悔药](#)

[2.7 更好用的提交列表](#)

[2.8 更好的差异比较](#)

[2.9 工作进度保存](#)

[2.10 代理SVN提交实现移动式办公](#)

[2.11 无处不在的分页器](#)

[2.12 快](#)

[第3章 Git的安装和使用](#)

[3.1 在Linux下安装和使用Git](#)

[3.1.1 包管理器方式安装](#)

[3.1.2 从源代码进行安装](#)

[3.1.3 从Git版本库进行安装](#)

[3.1.4 命令补齐](#)

[3.1.5 中文支持](#)

[3.2 在Mac OS X下安装和使用Git](#)

[3.2.1 以二进制发布包的方式安装](#)

[3.2.2 安装Xcode](#)

[3.2.3 使用Homebrew安装Git](#)

[3.2.4 从Git源码进行安装](#)

[3.2.5 命令补齐](#)

[3.2.6 其他辅助工具的安装](#)

[3.2.7 中文支持](#)

[3.3 在Windows下安装和使用Git（Cygwin篇）](#)

[3.3.1 安装Cygwin](#)

[3.3.2 安装Git](#)

[3.3.3 Cygwin的配置和使用](#)

[3.3.4 Cygwin下Git的中文支持](#)

[3.3.5 Cygwin下Git访问SSH服务](#)

[3.4 Windows下安装和使用Git（msysGit篇）](#)

[3.4.1 安装msysGit](#)

[3.4.2 msysGit的配置和使用](#)

[3.4.3 msysGit中shell环境的中文支持](#)

[3.4.4 msysGit中Git的中文支持](#)

[3.4.5 使用SSH协议](#)

[3.4.6 TortoiseGit的安装和使用](#)

[3.4.7 TortoiseGit的中文支持](#)

[第2篇 Git独奏](#)

[第4章 Git初始化](#)

[4.1 创建版本库及第一次提交](#)

[4.2 思考：为什么工作区根目录下有一个.git目录](#)

[4.3 思考：git config命令的各参数有何区别](#)

[4.4 思考：是谁完成的提交](#)

[4.5 思考：随意设置提交者姓名，是否太不安全](#)

[4.6 思考：命令别名是干什么的](#)

[4.7 备份本章的工作成果](#)

[第5章 Git暂存区](#)

[5.1 修改不能直接提交吗](#)

[5.2 理解Git暂存区（stage）](#)

[5.3 Git Diff魔法](#)

[5.4 不要使用git commit-a](#)

[5.5 搁置问题，暂存状态](#)

[第6章 Git对象](#)

[6.1 Git对象库探秘](#)

[6.2 思考：SHA1哈希值到底是什么，是如何生成的](#)

[6.3 思考：为什么不用顺序的数字来表示提交](#)

[第7章 Git重置](#)

[7.1 分支游标master探秘](#)

[7.2 用reflog挽救错误的重置](#)

[7.3 深入了解git reset命令](#)

[第8章 Git检出](#)

[8.1 HEAD的重置即检出](#)

[8.2 挽救分离头指针](#)

[8.3 深入了解git checkout命令](#)

[第9章 恢复进度](#)

[9.1 继续暂存区未完成的实践](#)

[9.2 使用git stash](#)

[9.3 探秘git stash](#)

[第10章 Git基本操作](#)

[10.1 先来合个影](#)

[10.2 删除文件](#)

[10.2.1 本地删除不是真的删除](#)

[10.2.2 执行git rm命令删除文件](#)

[10.2.3 命令git add-u快速标记删除](#)

[10.3 恢复删除的文件](#)

[10.4 移动文件](#)

[10.5 一个显示版本的Hello World](#)

[10.6 使用git add-i选择性添加](#)

[10.7 Hello World引发的新问题](#)

[10.8 文件忽略](#)

[10.9 文件归档](#)

[第11章 历史穿梭](#)

[11.1 图形工具：gitk](#)

[11.2 图形工具：gitg](#)

[11.3 图形工具：qgit](#)

[11.4 命令行工具](#)

[11.4.1 版本表示法：git rev-parse](#)

[11.4.2 版本范围表示法：git rev-list](#)

[11.4.3 浏览日志：git log](#)

[11.4.4 差异比较：git diff](#)

[11.4.5 文件追溯：git blame](#)

[11.4.6 二分查找：git bisect](#)

[11.4.7 获取历史版本](#)

[第12章 改变历史](#)

[12.1 悔棋](#)

[12.2 多步悔棋](#)

[12.3 回到未来](#)

[12.3.1 时间旅行一](#)

[12.3.2 时间旅行二](#)

[12.3.3 时间旅行三](#)

[12.4 丢弃历史](#)

[12.5 反转提交](#)

[第13章 Git克隆](#)

[13.1 鸡蛋不装在一个篮子里](#)

[13.2 对等工作区](#)

[13.3 克隆生成裸版本库](#)

[13.4 创建生成裸版本库](#)

[第14章 Git库管理](#)

[14.1 对象和引用哪里去了](#)

[14.2 暂存区操作引入的临时对象](#)

[14.3 重置操作引入的对象](#)

[14.4 Git管家：git-gc](#)

[14.5 Git管家的自动执行](#)

[第3篇 Git和声](#)

[第15章 Git协议与工作协同](#)

[15.1 Git支持的协议](#)

[15.2 多用户协同的本地模拟](#)

[15.3 强制非快进式推送](#)

[15.4 合并后推送](#)

[15.5 禁止非快进式推送](#)

[第16章 冲突解决](#)

[16.1 拉回操作中的合并](#)

[16.2 合并一：自动合并](#)

[16.2.1 修改不同的文件](#)

[16.2.2 修改相同文件的不同区域](#)

[16.2.3 同时更改文件名和文件内容](#)

[16.3 合并二：逻辑冲突](#)

[16.4 合并三：冲突解决](#)

[16.4.1 手工编辑完成冲突解决](#)

[16.4.2 图形工具完成冲突解决](#)

[16.5 合并四：树冲突](#)

[16.5.1 手工操作解决树冲突](#)

[16.5.2 交互式解决树冲突](#)

[16.6 合并策略](#)

[16.7 合并相关的设置](#)

[第17章 Git里程碑](#)

[17.1 显示里程碑](#)

[17.2 创建里程碑](#)

[17.2.1 轻量级里程碑](#)

[17.2.2 带说明的里程碑](#)

[17.2.3 带签名的里程碑](#)

[17.3 删除里程碑](#)

[17.4 不要随意更改里程碑](#)

[17.5 共享里程碑](#)

[17.6 删除远程版本库的里程碑](#)

[17.7 里程碑命名规范](#)

[第18章 Git分支](#)

[18.1 代码管理之殇](#)

[18.1.1 发布分支](#)

[18.1.2 特性分支](#)

[18.1.3 卖主分支](#)

[18.2 分支命令概述](#)

[18.3 "Hello World"开发计划](#)

[18.4 基于特性分支的开发](#)

[18.4.1 创建分支user1/getopt](#)

[18.4.2 创建分支user2/i18n](#)

[18.4.3 开发者user1完成功能开发](#)

[18.4.4 将user1/getopt分支合并到主线](#)

[18.5 基于发布分支的开发](#)

[18.5.1 创建发布分支](#)

[18.5.2 开发者user1工作在发布分支](#)

[18.5.3 开发者user2工作在发布分支](#)

[18.5.4 开发者user2合并推送](#)

[18.5.5 发布分支的提交合并到主线](#)

[18.6 分支变基](#)

[18.6.1 完成user2/i18n特性分支的开发](#)

[18.6.2 分支user2/i18n变基](#)

[第19章 远程版本库](#)

[19.1 远程分支](#)

[19.2 分支追踪](#)

[19.3 远程版本库](#)

[19.4 PUSH和PULL操作与远程版本库](#)

[19.5 里程碑和远程版本库](#)

[19.6 分支和里程碑的安全性](#)

[第20章 补丁文件交互](#)

[20.1 创建补丁](#)

[20.2 应用补丁](#)

[20.3 StGit和Quilt](#)

[20.3.1 StGit](#)

[20.3.2 Quilt](#)

[第4篇 Git协同模型](#)

[第21章 经典Git协同模型](#)

[21.1 集中式协同模型](#)

[21.1.1 传统集中式协同模型](#)

[21.1.2 Gerrit特殊的集中式协同模型](#)

[21.2 金字塔式协同模型](#)

[21.2.1 贡献者开放只读版本库](#)

[21.2.2 以补丁方式贡献代码](#)

[第22章 Topgit协同模型](#)

[22.1 作者版本控制系统的三个里程碑](#)

[22.2 Topgit原理](#)

[22.3 Topgit的安装](#)

[22.4 Topgit的使用](#)

[22.5 用Topgit方式改造Topgit](#)

[22.6 Topgit使用中的注意事项](#)

[第23章 子模组协同模型](#)

[23.1 创建子模组](#)

[23.2 克隆带子模组的版本库](#)

[23.3 在子模组中修改和子模组的更新](#)

[23.4 隐性子模组](#)

[23.5 子模组的管理问题](#)

[第24章 子树合并](#)

[24.1 引入外部版本库](#)

[24.2 子目录方式合并外部版本库](#)

[24.3 利用子树合并跟踪上游改动](#)

[24.4 子树拆分](#)

[24.5 git-subtree插件](#)

[第25章 Android式多版本库协同](#)

[25.1 关于repo](#)

[25.2 安装repo](#)

[25.3 repo和清单库的初始化](#)

[25.4 清单库和清单文件](#)

[25.5 同步项目](#)

[25.6 建立Android代码库本地镜像](#)

[25.7 repo的命令集](#)

[25.8 repo命令的工作流](#)

[25.9 好东西不能Android独享](#)

[25.9.1 repo+Gerrit模式](#)

[25.9.2 repo无审核模式](#)

[25.9.3 改进的repo无审核模式](#)

[第26章 Git和SVN协同模型](#)

[26.1 使用git-svn的一般流程](#)

[26.2 git-svn的奥秘](#)

[26.2.1 Git库配置文件的扩展及分支映射](#)

[26.2.2 Git工作分支和Subversion如何对应](#)

[26.2.3 其他辅助文件](#)

[26.3 多样的git-svn克隆模式](#)

[26.4 共享git-svn的克隆库](#)

[26.5 git-svn的局限](#)

[第5篇 搭建Git服务器](#)

[第27章 使用HTTP协议](#)

[27.1 哑传输协议](#)

[27.2 智能HTTP协议](#)

[27.3 Gitweb服务器](#)

[27.3.1 Gitweb的安装](#)

[27.3.2 Gitweb的配置](#)

[27.3.3 版本库的Gitweb相关设置](#)

[27.3.4 即时Gitweb服务](#)

[第28章 使用Git协议](#)

[28.1 Git协议语法格式](#)

[28.2 Git服务软件](#)

[28.3 以inetd方式配置运行](#)

[28.4 以runit方式配置运行](#)

[第29章 使用SSH协议](#)

[29.1 SSH协议语法格式](#)

[29.2 服务架设方式比较](#)

[29.3 关于SSH公钥认证](#)

[29.4 关于SSH主机别名](#)

[第30章 Gitolite服务架设](#)

[30.1 安装Gitolite](#)

[30.1.1 服务器端创建专用账号](#)

[30.1.2 Gitolite的安装/升级](#)

[30.1.3 关于SSH主机别名](#)

[30.1.4 其他的安装方法](#)

[30.2 管理Gitolite](#)

[30.2.1 管理员克隆gitolite-admin管理库](#)

[30.2.2 增加新用户](#)

[30.2.3 更改授权](#)

[30.3 Gitolite授权详解](#)

[30.3.1 授权文件的基本语法](#)

[30.3.2 定义用户组和版本库组](#)

[30.3.3 版本库ACL](#)

[30.3.4 Gitolite授权机制](#)

[30.4 版本库授权案例](#)

[30.4.1 对整个版本库进行授权](#)

[30.4.2 通配符版本库的授权](#)

[30.4.3 用户自己的版本库空间](#)

[30.4.4 对引用的授权：传统模式](#)

[30.4.5 对引用的授权：扩展模式](#)

[30.4.6 对引用的授权：禁用规则的使用](#)

[30.4.7 用户分支](#)

[30.4.8 对路径的写授权](#)

[30.5 创建新版本库](#)

[30.5.1 在配置文件中出现的版本库，即时生成](#)

[30.5.2 通配符版本库，管理员通过推送创建](#)

[30.5.3 直接在服务器端创建](#)

[30.6 对Gitolite的改进](#)

[30.7 Gitolite功能拓展](#)

- [30.7.1 版本库镜像](#)
- [30.7.2 Gitweb和Git daemon支持](#)
- [30.7.3 其他功能拓展和参考](#)

[第31章 Gitosis服务架设](#)

- [31.1 安装Gitosis](#)
 - [31.1.1 Gitosis的安装](#)
 - [31.1.2 服务器端创建专用账号](#)
 - [31.1.3 Gitosis服务初始化](#)
- [31.2 管理Gitosis](#)
 - [31.2.1 管理员克隆gitolit-admin管理库](#)
 - [31.2.2 增加新用户](#)
 - [31.2.3 更改授权](#)
- [31.3 Gitosis授权详解](#)
 - [31.3.1 Gitosis默认设置](#)
 - [31.3.2 管理版本库gitosis-admin](#)
 - [31.3.3 定义用户组和授权](#)
 - [31.3.4 Gitweb整合](#)
- [31.4 创建新版本库](#)
- [31.5 轻量级管理的Git服务](#)

[第32章 Gerrit代码审核服务器](#)

- [32.1 Gerrit的实现原理](#)
- [32.2 架设Gerrit的服务器](#)
- [32.3 Gerrit的配置文件](#)
- [32.4 Gerrit的数据库访问](#)
- [32.5 立即注册为Gerrit管理员](#)
- [32.6 管理员访问SSH的管理接口](#)
- [32.7 创建新项目](#)
- [32.8 从已有的Git库创建项目](#)
- [32.9 定义评审 workflow](#)
- [32.10 Gerrit评审 workflow 实战](#)
 - [32.10.1 开发者在本地版本库中工作](#)
 - [32.10.2 开发者向审核服务器提交](#)
 - [32.10.3 审核评审任务](#)
 - [32.10.4 评审任务没有通过测试](#)
 - [32.10.5 重新提交新的补丁集](#)
 - [32.10.6 新修订集通过评审](#)
 - [32.10.7 从远程版本库更新](#)

欢迎访问：电子书学习和下载网站 (<https://www.shgis.cn>)

文档名称：《Git权威指南》蒋鑫 著.epub

请登录 <https://shgis.cn/post/274.html> 下载完整文档。

手机端请扫码查看：

