

SECOND EDITION

Version Control with Git



Jon Loeliger and Matthew McCullough

O'REILLY®

Beijing • Cambridge • Farnham • Koln • Sebastopol • Tokyo

Table of Contents

Preface	xi
1. Introduction	1
Background	1
The Birth of Git	2
Precedents	4
Timeline	6
What's in a Name?	7
2. Installing Git	9
Using Linux Binary Distributions	9
Debian/Ubuntu	9
Other Binary Distributions	10
Obtaining a Source Release	11
Building and Installing	11
Installing Git on Windows	13
Installing the Cygwin Git Package	14
Installing Standalone Git (msysGit)	14
3. Getting Started	19
The Git Command Line	19
Quick Introduction to Using Git	21
Creating an Initial Repository	21
Adding a File to Your Repository	22
Configuring the Commit Author	24
Making Another Commit	24
Viewing Your Commits	25
Viewing Commit Differences	26
Removing and Renaming Files in Your Repository	26
Making a Copy of Your Repository	27
Configuration Files	28

Configuring an Alias	30
Inquiry	30
4. Basic Git Concepts	31
Basic Concepts	31
Repositories	31
Git Object Types	32
Index	33
Content-Addressable Names	33
Git Tracks Content	34
Pathname Versus Content	35
Pack Files	36
Object Store Pictures	36
Git Concepts at Work	39
Inside the .git Directory	39
Objects, Hashes, and Blobs	40
Files and Trees	41
A Note on Git's Use of SHA1	42
Tree Hierarchies	43
Commits	44
Tags	46
5. File Management and the Index	47
It's All About the Index	48
File Classifications in Git	48
Using git add	50
Some Notes on Using git commit	52
Using git commit --all «	52
Writing Commit Log Messages	54
Using git rm	54
Using git mv	56
A Note on Tracking Renames	57
The .gitignore File	58
A Detailed View of Git's Object Model and Files	60
6. Commits	65
Atomic Changesets	66
Identifying Commits	67
Absolute Commit Names	67
refs and symrefs	68
Relative Commit Names	69
Commit History	72
Viewing Old Commits	72

Commit Graphs	74
Commit Ranges	78
Finding Commits	83
Using git bisect	83
Using git blame	87
Using Pickaxe	88
7. Branches	89
Reasons for Using Branches	89
Branch Names	90
Dos and Don'ts in Branch Names	91
Using Branches	91
Creating Branches	93
Listing Branch Names	94
Viewing Branches	94
Checking out Branches	97
A Basic Example of Checking out a Branch	97
Checking out When You Have Uncommitted Changes	98
Merging Changes into a Different Branch	99
Creating and Checking out a New Branch	101
Detached HEAD Branches	102
Deleting Branches	103
8. Diffs	107
Forms of the git diff Command	108
Simple git diff Example	112
git diff and Commit Ranges	115
git diff with Path Limiting	117
Comparing How Subversion and Git Derive diffs	119
9. Merges	121
Merge Examples	121
Preparing for a Merge	122
Merging Two Branches	122
A Merge with a Conflict	124
Working with Merge Conflicts	128
Locating Conflicted Files	129
Inspecting Conflicts	129
How Git Keeps Track of Conflicts	134
Finishing Up a Conflict Resolution	135
Aborting or Restarting a Merge	137
Merge Strategies	137
Degenerate Merges	140

Normal Merges	142
Specialty Merges	143
Applying Merge Strategies	144
Merge Drivers	145
How Git Thinks About Merges	146
Merges and Git's Object Model	146
Squash Merges	147
Why Not Just Merge Each Change One by One?	148
10. Altering Commits	151
Caution About Altering History	152
Using <code>git reset</code>	154
Using <code>git cherry-pick</code>	161
Using <code>git revert</code>	163
<code>reset</code> , <code>revert</code> , and <code>checkout</code>	164
Changing the Top Commit	165
Rebasing Commits	167
Using <code>git rebase -i</code>	170
rebase Versus merge	f 174
11. The Stash and the Reflog	181
The Stash	181
The Reflog	§ 189
12. Remote Repositories	195
Repository Concepts	196
Bare and Development Repositories	196
Repository Clones	> 197
Remotes	198
Tracking Branches	199
Referencing Other Repositories	/ 200
Referring to Remote Repositories	200
The <code>refspec</code>	202
Example Using Remote Repositories	204
Creating an Authoritative Repository	v 205
Make Your Own Origin Remote	206
Developing in Your Repository	208
Pushing Your Changes	209
Adding a New Developer	210
Getting Repository Updates	212
Remote Repository Development Cycle in Pictures	217
Cloning a Repository	217
Alternate Histories	218

Non-Fast-Forward Pushes	219
Fetching the Alternate History	221
Merging Histories	222
Merge Conflicts	223
Pushing a Merged History	223
Remote Configuration	223
Using git remote	224
Using git config	225
Using Manual Editing	226
Working with Tracking Branches	227
Creating Tracking Branches	227
Ahead and Behind	230
Adding and Deleting Remote Branches	231
Bare Repositories and git push	232
13. Repository Management	235
A Word About Servers	235
Publishing Repositories	236
Repositories with Controlled Access	236
Repositories with Anonymous Read Access	238
Repositories with Anonymous Write Access	242
Publishing Your Repository to GitHub	242
Repository Publishing Advice	243
Repository Structure	244
The Shared Repository Structure	244
Distributed Repository Structure	244
Repository Structure Examples	246
Living with Distributed Development	248
Changing Public History	248
Separate Commit and Publish Steps	249
No One True History	249
Knowing Your Place	250
Upstream and Downstream Flows	251
The Maintainer and Developer Roles	251
Maintainer-Developer Interaction	252
Role Duality	253
Working with Multiple Repositories	254
Your Own Workspace	254
Where to Start Your Repository	255
Converting to a Different Upstream Repository	256
Using Multiple Upstream Repositories	257
Forking Projects	259

14. Patches	263
Why Use Patches?	264
Generating Patches	265
Patches and Topological Sorts	272
Mailing Patches	273
Applying Patches	276
Bad Patches	283
Patching Versus Merging	283
15. Hooks	285
Installing Hooks	287
Example Hooks	287
Creating Your First Hook	288
Available Hooks	290
Commit-Related Hooks	290
Patch-Related Hooks	291
Push-Related Hooks	292
Other Local Repository Hooks	4 294
16. Combining Projects	295
The Old Solution: Partial Checkouts	296
The Obvious Solution: Import the Code into Your Project	297
Importing Subprojects by Copying	299
Importing Subprojects with <code>git pull -s subtree</code>	299
Submitting Your Changes Upstream	303
The Automated Solution: Checking out Subprojects Using Custom Scripts	304
The Native Solution: <code>gitlinks</code> and <code>git submodule</code>	305
<code>Gitlinks</code>	306
The <code>git submodule</code> Command	308
17. Submodule Best Practices	313
Submodule Commands	314
Why Submodules?	315
Submodules Preparation	316
Why Read Only?	316
Why Not Read Only?	317
Examining the Hashes of Submodule Commits	317
Credential Reuse	318
Use Cases	318
Multilevel Nesting of Repos	319
Submodules on the Horizon	320

18. Using Git with Subversion Repositories	321
Example: A Shallow Clone of a Single Branch	321
Making Your Changes in Git	324
Fetching Before Committing	325
Committing Through <code>git svn rebase</code>	326
Pushing, Pulling, Branching, and Merging with <code>git svn</code>	327
Keeping Your Commit IDs Straight	328
Cloning All the Branches	329
Sharing Your Repository	331
Merging Back into Subversion	332
Miscellaneous Notes on Working with Subversion	334
<code>svn:ignore</code> Versus <code>.gitignore</code>	334
Reconstructing the <code>git-svn</code> Cache	334
19. Advanced Manipulations	337
Using <code>git filter-branch</code>	337
Examples Using <code>git filter-branch</code>	339
<code>filter-branch</code> Pitfalls	344
How I Learned to Love <code>git rev-list</code>	345
Date-Based Checkout	345
Retrieve Old Version of a File	348
Interactive Hunk Staging	350
Recovering a Lost Commit	360
The <code>git fsck</code> Command	361
Reconnecting a Lost Commit	365
20. Tips, Tricks, and Techniques	367
Interactive Rebase with a Dirty Working Directory	367
Remove Left-Over Editor Files	368
Garbage Collection	368
Split a Repository	370
Tips for Recovering Commits	371
Subversion Conversion Tips	372
General Advice	372
Remove a Trunk After <code>svn</code> Import	373
Removing SVN Commit IDs	373
Manipulating Branches from Two Repositories	374
Recovering from an Upstream Rebase	374
Make Your Own Git Command	376
Quick Overview of Changes	376
Cleaning Up	377
Using <code>git-grep</code> to Search a Repository	378
Updating and Deleting refs	380

Following Files that Moved	381
Keep, But Don't Track, This File	382
Have You Been Here Before?	382
21. Git and GitHub	385
Repo for Public Code	385
Creating a GitHub Repository	388
Social Coding on Open Source	390
Watchers	391
News Feed	392
Forks	392
Creating Pull Requests	394
Managing Pull Requests	396
Notifications	398
Finding Users, Projects, and Code	401
Wikis	402
GitHub Pages (Git for Websites)	403
In-Page Code Editor	405
Subversion Bridge	^ 407
Tags Automatically Becoming Archives	408
Organizations	409
REST API	410
Social Coding on Closed Source	\ 411
Eventual Open Sourcing	411
Coding Models	412
GitHub Enterprise	414
GitHub in Sum	416
Index	417