Git
Revision Control

Raymond E. Marcil
<marcilr@gmail.com>

Revision 42 (November 12, 2015)

Abstract

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.[1]

[1] Git - https://git-scm.com/

Contents

List of Figures

List of Tables

List of Definitions and Abbreviations

- **Branch** - [FIXME: Need data]
- **Git** - Quoting Linus: "I'm an egotistical bastard, and I name all my projects after myself. First 'Linux', now 'Git"'.
  ('git' is British slang for "pig headed, think they are always correct, argumentative").[2]
- **Tag** - [FIXME: Need data]

[2] Git FAQ
https://git.wiki.kernel.org/index.php/GitFaq#Why_the_.27Git.27_name.3F

Introduction

Git is a distributed revision control system with an emphasis on speed,[3] data integrity,[4] and support for distributed, non-linear workflows.[5] Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since become one of the most widely adopted version control systems for software development.[6]

As with most other distributed revision control systems, and unlike most clientserver systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server.[7] Like the Linux kernel, Git is free software distributed under the terms of the GNU General Public License version 2.[8]

Command Reference

[FIXME: Need data here.]

[3] Torvalds, Linus (2005-04-07). "Re: Kernel SCM saga..." linux-kernel (Mailing list). "So I'm writing some scripts to try to track things a whole lot faster."

[4] Torvalds, Linus (2007-06-10). "Re: fatal: serious inflate inconsistency". git (Mailing list). A brief description of Git's data integrity design goals.

[5] Linus Torvalds (2007-05-03). Google tech talk: Linus Torvalds on git. Event occurs at 02:30. Retrieved 2007-05-16.

[6] "Eclipse Community Survey 2014 results — Ian Skerrett". Ianskerrett.wordpress.com. 2014-06-23. Retrieved 2014-06-23.

[7] Chacon, Scott (24 December 2014). Pro Git (2nd ed.). New York, NY: Apress. pp. 2930. ISBN 978-1484200773.

[8] Git (software), From Wikipedia, the free encyclopedia, https://en.wikipedia.org/wiki/Git_(software)

Branching & Tagging

In short: Best practice is branch out, merge often and keep always in sync.

There are pretty clear conventions about keeping your code in a separate branches from master branch:

1. You are about to make an implementation of major or disruptive change
2. You are about to make some changes that might not be used
3. You want to experiment on something that you are not sure it will work
4. When you are told to branch out, others might have something they need to do in master

Rule of thumb is after branching out, you should keep in sync with the master branch. Because eventually you need to merge it back to master. In order to avoid a huge complicated mess of conflicts when merging back, you should commit often, merge often.[9]

[9] Git branching and tagging best practices
http://programmers.stackexchange.com/questions/165725/git-branching-and-tagging-best-practices

Project Host

Bitbucket vs. GitHub

Repo

"Repo is a repository management tool that we built on top of Git. Repo unifies the many Git repositories when necessary, does the uploads to a revision control system, and automates parts of the development workflow. Repo is not meant to replace Git, only to make it easier to work with Git in the context of Android. The `repo` command is an executable Python script that you can put anywhere in your path. In working with source files, you will use Repo for across-network operations. For example, with a single Repo command you can download files from multiple repositories into your local working directory."[10]

[FIXME: The above `repo` quote has been heavily modified. Need to rewrite with original verbage.]

[10] Developing – http://source.android.com/source/developing.html

Appendix

A successful Git branching model
by Vincent Driessen on Tuesday, January 05, 2010
Fine branching diagram here.
http://nvie.com/posts/a-successful-git-branching-model/

Bitbucket vs. GitHub: Which project host has the most?
The right choice boils down to a number of factors – you might even consider using both
http://www.infoworld.com/article/2611771/application-development/application-development-bitbucket-v

Developing
Has Repo and Gerrit info
http://source.android.com/source/developing.html

Fetching a remote
> `git clone`
> `git fetch`
> `git merge`
> `git pull`
https://help.github.com/articles/fetching-a-remote/

Git
https://git-scm.com/

Git (software)
From Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/Git_(software)

Git About
https://git-scm.com/about

Git branching and tagging best practices
Excellent details and semantics.
http://programmers.stackexchange.com/questions/165725/git-branching-and-tagging-best-practices

Git FAQ
https://git.wiki.kernel.org/index.php/GitFaq

Git repositories on gerrit
https://gerrit.googlesource.com/

GitHub
Project host
https://github.com/

Pro Git (the git book)
Available as pdf, epub, mobi, and html.
http://git-scm.com/book/en/v2

Re: repo + private repositories in github
Details on manifest for google repo use.
https://groups.google.com/forum/embed/#!topic/repo-discuss/kCXO-NdFvj4

Repo Command Reference
http://source.android.com/source/using-repo.html

repo Manifest Format
https://gerrit.googlesource.com/git-repo/+/master/docs/manifest-format.txt

Set Up Git
>Creating a repository
>Forking a repository
>Being social
https://help.github.com/articles/set-up-git/