# Git

## Git Commands

**git config --global user.name "Your Name"** - Set the name that will be attached to your commits and tags.

**git config --global user.email "you@example.com"** - Set the e-mail address that will be attached to your commits and tags.

**git config --global color.ui auto** - Enable some colorization of Git output.

**git init [project name]git clone [project url]**

**git status**

**git diff [file]**

**git diff --staged [file]**

**git checkout -- [file]**

**git add [file]**

**git commit [-m "message here"]**

**git rm [file] -** Remove file from working directory and add deletion to staging area.

**git branch [-a]** List all local branches in repository. With -a: show all branches (with remote).

**git branch [name]** Create new branch, referencing the current HEAD.

**git checkout [-b] [name]** Switch working directory to the specified branch. With -b: Git will create the specified branch if it does not exist.

**git merge [from name]** Join specified [from name] branch into your current branch (the one you are on currenlty).

**git branch -d [name]** Remove selected branch, if it is already merged into any other. -D instead of -d forces deletion

**git log [-n count]** List commit history of current branch. -n count limits list to last n commits.

**git log --oneline --graph --decorate** An overview with references labels and history graph. One commit per line.

**git log ref..** List commits that are present on current branch and not merged into ref. A ref can be e.g. a branch name or a tag name.

**git log ..ref** List commit, that are present on ref and not merged into current branch.

**git reflog** List operations (like checkouts, commits etc.) made on local repository

**git fetch [remote]** Fetch changes from the remote, but not update tracking branches.

**git fetch --prune [remote]** Remove remote refs, that were removed from the remote repository.

**git pull [remote]** Fetch changes from the remote and merge current branch with its upstream.

**git push [--tags] [remote]** Push local changes to the remote. Use --tags to push tags.

**git reset [file]** - Get file back from staging area to working directory

**git reset --soft**

**git reset --mixed**

**git reset --hard**

**git revert HEAD** - Undo the commit and create a new commit

**git branch –d [head]** - Delete the Current Branch

**git branch -d <branch>** - (Only works if it is merged with another branch)

**git branch -D <branch>** - (Forcing the delete)

**git reflog --no-abbrev** - contains all log information of the steps you have done

**git checkout** [sha]

**git checkout -b** [branchname] <sha>

**git branch --merged master** lists branches merged into master

**git branch --merged** lists branches merged into HEAD (i.e. tip of current branch)

**git branch --no-merged** lists branches that have not been merged

**git stash**

**git stash apply**

**git stash pop**

**git stash drop**

## CVCS vs DVCS ?

|  |  |
| --- | --- |
| CVCS | DVCS |
| One Central repository which is a server | Every user has a complete repository which is called as local repository |
| Every User who needs to access must be connected to the Network | We can go offline in DVCS but Network is needed to share repositories with others |
| SVN | Git, Mercurial, Bazaar |

## Which Branching Strategy used in your Project ?



## Which branch will be pushed to release?

release branch

## What is GIT?

GIT is a distributed version control system and source code management (SCM) system with an emphasis to handle small and large projects with speed and efficiency.

## What is a repository in GIT?

A repository contains a directory named .git, where git keeps all of its metadata for the repository. The content of the .git directory are private to git.

## What is Git Client?

“Git” is really just a collection of individual commands you execute in the shell (Appendix A). This interface is not appealing for everyone. Some may prefer to do Git operations via a client with a graphical interface

## What is GIT version control?

With the help of GIT version control, you can track the history of a collection of files and includes the functionality to revert the collection of files to another version. Each version captures a snapshot of the file system at a certain point of time. A collection of files and their complete history are stored in a repository.

## what is the use of SCM?

Source Code Management (SCM) and version control systems ensure all members of a team stay on top of changes to source code and related files. These tools are also crucial in coordinating parallel work on different features and the integration of the features for software releases.

## Why is git more secure?

Git is secure against modifying the content without being able to know that something went wrong. Each commit references to the parents hash. Each item in the commit has its hash as reference and the commit itself is also hashed. So changing just one item would change the complete hash of the commit itself.

## What are the advantages of using GIT ? / Why do you prefer Git ?

1. Data redundancy and replication
2. High availability
3. Only one.git directory per repository
4. Superior disk utilization and network performance
5. Collaboration friendly
6. Any sorts of projects can use GIT

## What is the difference between Git and Github?

Git is a revision control system, a tool to manage your source code history.

GitHub is a hosting service for Git repositories.

GitHub is a website where you can upload a copy of your Git repository. It is a Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

## How do you rate GIT in terms of speed?

Git is fast. Speed and performance has been a primary design goal of the Git from the start. With Git, nearly all operations are performed locally, giving it a huge speed advantage on centralized systems that constantly have to communicate with a server somewhere.

Git was built to work on the Linux kernel, meaning that it has had to effectively handle large repositories from day one. Git is written in C, reducing the overhead of runtimes associated with higher-level languages.

## Mention some of the best graphical GIT client for LINUX?

Some of the best GIT client for LINUX is

a) Git Cola

b) Git-g

c) Smart git

d) Giggle

e) Git GUI

f) qGit

## What is Subgit? Why to use Subgit?

‘Subgit’ is a tool for a smooth, stress-free SVN to Git migration. Subgit is a solution for a company -wide migration from SVN to Git that is:

1. It is much better than git-svn
2. No requirement to change the infrastructure that is already placed
3. Allows to use all git and all sub-version features
4. Provides genuine stress –free migration experience.

## Name a few Git repository hosting services

Pikacode

Visual Studio Online

GitHub

GitEnterprise

SourceForge.net

## What is the difference between GIT and SVN?

The difference between GIT and SVN is

1. Git is less preferred for handling extremely large files or frequently changing binary files while SVN can handle multiple projects stored in the same repository.
2. GIT does not support ‘commits’ across multiple branches or tags. Subversion allows the creation of folders at any location in the repository layout.
3. Gits are unchangeable, while Subversion allows committers to treat a tag as a branch and to create multiple revisions under a tag root.

## What is GitOps?

GitOps is a paradigm or a set of practices that empowers developers to perform tasks which typically fall under the purview of IT operations. GitOps requires us to describe and observe systems with declarative specifications that eventually form the basis of continuous everything.

## What language is used in GIT?

GIT is fast, and ‘C’ language makes this possible by reducing the overhead of runtimes associated with higher languages.

## Difference between perforce and git?

Git is DVCS

perforce is CVCS

## How git instaweb is used?

‘Git Instaweb’ automatically directs a web browser and runs webserver with an interface into your local repository.

## Why GIT better than Subversion?

GIT is an open source version control system; it will allow you to run ‘versions’ of a project, which show the changes that were made to the code overtime also it allows you keep the backtrack if necessary and undo those changes. Multiple developers can checkout, and upload changes and each change can then be attributed to a specific developer.

## What is the function of ‘git config’?

The ‘git config’ command is a convenient way to set configuration options for your Git installation. Behaviour of a repository, user info, preferences etc. can be defined through this command.

## what is git init vs git init –bare ?

**git init** will create a .git folder containg all the files related to git like ref, object

**git init --bare** will not create a .git folder it will simply create all the files inside .git folder in our dir which we ran git init --bare . This command it used for setting up a remote repository in our local server

## How can you create a repository in Git?

In Git, to create a repository, create a directory for the project if it does not exist, and then run command “git init”. By running this command .git directory will be created in the project directory, the directory does not need to be empty.

## What are bare and non-bare repositories?

developers clone and fetch from the bare repository and push updates to it... if you set up a repository into which developers push changes, it should be bare. In effect, this is a special case of the more general best practice that a published repository should be bare.

## What is ‘bare repository’ in GIT?

To co-ordinate with the distributed development and developers team, especially when you are working on a project from multiple computers ‘Bare Repository’ is used. A bare repository comprises of a version history of your code.

## How do you identify a git base directory?

Folder which containes .git folder

## What is HEAD?

When working with Git, only one branch can be checked out at a time - and this is what's called the "HEAD" branch. Often, this is also referred to as the "active" or "current" branch. Git makes note of this current branch in a file located inside the Git repository, in . git/HEAD .

## What is ‘head’ in git and how many heads can be created in a repository?

A ‘head’ is simply a reference to a commit object. In every repository, there is a default head referred as “Master”. A repository can contain any number of heads.

## What is the command you can use to write a commit message?

The command that is used to write a commit message is “**git commit –m**”. The –a on the command line instructs git to commit the new content of all tracked files that have been modified. You can use “git add<file>” before git commit –a if new files need to be committed for the first time.

## What does commit object contain?

a) A set of files, representing the state of a project at a given point of time

b) Reference to parent commit objects

c) An **SHA Simple Hashing Algorithm** name, a 40 character string that uniquely identifies the commit object.

## Explain what is commit message?

Commit message is a feature of git which appears when you commit a change. Git provides you a text editor where you can enter the modifications made in commits.

## How can you fix a broken commit message?

To fix any broken commit, you will use the command “**git commit --amend**”. By running this command, you can fix the broken commit message in the editor.

## What are hooks?

Git hooks are scripts that run automatically every time a particular event occurs in a Git repository. They let you customize Git's internal behavior and trigger customizable actions at key points in the development life cycle.

## What does ‘hooks’ consist of in git?

This directory consists of Shell scripts which are activated after running the corresponding Git commands. For example, git will try to execute the post-commit script after you run a commit.

## How to restrict few filetypes from getting committed?

Using **.gitignore global** or using **Hooks**

## What is the best way to stop developers from doing an accidental commits?

By creating a Hook using shell script we can stop developers from doing commits on DEV & Master

*#!/bin/sh*

*branch="$(git rev-parse --abbrev-ref HEAD)"*

*if [ "$branch" = "dev" ]; then*

*echo "Dev Branch commit is blocked"*

*exit 1*

*fi*

*if [ "$branch" = "master" ]; then*

*echo "Master Branch commit is blocked"*

*exit 1*

*fi*

## In Git how do you revert a commit that has already been pushed and made public?

**git revert <name of bad commit>**

There can be two answers to this question and make sure that you include both because any of the below options can be used depending on the situation:

Remove or fix the bad file in a new commit and push it to the remote repository. This is the most natural way to fix an error. Once you have made necessary changes to the file, commit it to the remote repository for that I will use

git commit -m “commit message” .

Create a new commit that undoes all changes that were made in the bad commit.to do this I will use a command

## Why is it advisable to create an additional commit rather than amending an existing commit?

There are couple of reason

a) The amend operation will destroy the state that was previously saved in a commit. If it’s just the commit message being changed then that’s not an issue. But if the contents are being amended then chances of eliminating something important remains more.

b) Abusing “git commit- amend” can cause a small commit to grow and acquire unrelated changes.

## What is the purpose of branching in GIT?

The purpose of branching in GIT is that you can create your own branch and jump between those branches. It will allow you to go to your previous work keeping your recent work intact.

## To delete a branch what is the command that is used?

**git branch –d [head]**

**git branch -d <branch>** (Only works if it is merged with another branch)

**git branch -D <branch>** (Forcing the delete)

Once your development branch is merged into the main branch, you don’t need development branch.

To delete a branch use, the command.

## How to recover a Deleted Branch ?

**git reflog --no-abbrev** - contains all log information of the steps you have done

**git checkout [sha]**

**git checkout -b [branchname] [sha]**

## How will you know in GIT if a branch has been already merged into master?

**git branch --merged master** lists branches merged into master

**git branch --merged** lists branches merged into HEAD (i.e. tip of current branch)

**git branch --no-merged** lists branches that have not been merged

## What is the function of ‘git checkout’ in git?

A ‘git checkout’ command is used to update directories or specific files in your working tree with those from another branch without merging it in the whole branch.

## What is the use of ‘git log’?

To find specific commits in your project history- by author, date, content or history ‘git log’ is used.

## What is git Is-tree?

‘git Is-tree’ represents a tree object including the mode and the name of each item and the SHA-1 value of the blob or the tree.

## What is the function of ‘GIT PUSH’ in GIT?

‘GIT PUSH’ updates remote refs along with associated objects.

## What is “Staging Area” or “Index” in GIT?

Before completing the commits, it can be formatted and reviewed in an intermediate area known as ‘Staging Area’ or ‘Index’.

## What is ‘git add’ is used for?

‘git add’ adds file changes in your existing directory to your index.

## What is the function of ‘git rm’?

git rm is used to remove a file from a Git repository.

## Git Remote vs Git Add vs Git Clone ?

git remote add <alias> <URL> -> adds remote URL

git add -> adds file to staging area

git clone -> downloads an existing Git repository to your local computer.

## What is the function of ‘git diff ’ in git?

‘git diff ’ shows the changes between commits, commit and working tree etc.

## What is ‘git status’ is used for?

As ‘Git Status’ shows you the difference between the working directory and the index, it is helpful in understanding a git more comprehensively.

## What is the difference between the ‘git diff ’and ‘git status’?

'git diff ' depicts the changes between commits, commit and working tree

'git status' shows you the difference between the working directory and the index

## Git checkout ?

git checkout command lets you navigate between the branches created by git branch

## What is merge ?

merges commit from one branch to another

## Git 3 way merge ?

With a three-way merge, it can compare the two files, but it can also compare each of them against the original copy (before either of you changed it). So it can see that you removed the first line, and that your friend added the last line. And it can use that information to produce the merged version.

## Git mergeconflict how to solve ?

we can use git mergetool to resolve the conflict and make changes to our file

git reset – hard or git reset -- soft or git reset --mixed

## What is Rebase ? What is the syntax for “Rebasing” in Git?

Rebasing is the process of moving or combining a sequence of commits to a new base commit.

## git cherrypick

git cherry-pick is a powerful command that enables arbitrary Git commits to be picked by reference and appended to the current working HEAD. Cherry picking is the act of picking a commit from a branch and applying it to another.

## How can you bring a new feature in the main branch?

To bring a new feature in the main branch, you can use a command “git merge” or “git pull command”.

## What is a ‘conflict’ in git?

A ‘conflict’ arises when the commit that has to be merged has some change in one place, and the current commit also has a change at the same place. Git will not be able to predict which change should take precedence.

## How can conflict in git resolved?

To resolve the conflict in git, edit the files to fix the conflicting changes and then add the resolved files by running “git add” after that to commit the repaired merge, run “git commit”. Git remembers that you are in the middle of a merger, so it sets the parents of the commit correctly

## What is another option for merging in git?

“Rebasing” is an alternative to merging in git.

## Git fetch vs Git Pull vs Git Clone ?

**git fetch** will only update your local repository, we need to use checkout or merge command to update it to our working copy

**git pull** is a combination of **git fetch + git merge,** it will update your both local repo and make changes in working directory

**git clone** is used for just downloading exactly what is currently working on the remote server repository and saving it in your machine's folder where that project is placed

I have even heard of people running **git fetch periodically in a cron job** in the background.

## What is the difference between git pull and git fetch?

Git pull command pulls new changes or commits from a particular branch from your central repository and updates your target branch in your local repository.

Git fetch is also used for the same purpose but it works in a slightly different way. When you perform a git fetch, it pulls all new commits from the desired branch and stores it in a new branch in your local repository. If you want to reflect these changes in your target branch, git fetch must be followed with a git merge. Your target branch will only be updated after merging the target branch and fetched branch. Just to make it easy for you, remember the equation below:

Git pull = git fetch + git merge

## What is the difference between ‘git remote’ and ‘git clone’?

‘git remote add’ just creates an entry in your git config that specifies a name for a particular URL. While, ‘git clone’ creates a new git repository by copying and existing one located at the URI.

## What is the function of git clone?

The git clone command creates a copy of an existing Git repository. To get the copy of a central repository, ‘cloning’ is the most common way used by programmers.

git clone <Repo URL>

## Git restore ?

**git-restore** is a tool to revert non-commited changes. Non-commited changes are:

a) changes in your working copy,

b) content in your index (a.k.a. staging area).

## git reset –hard vs –soft vs –mixed ?

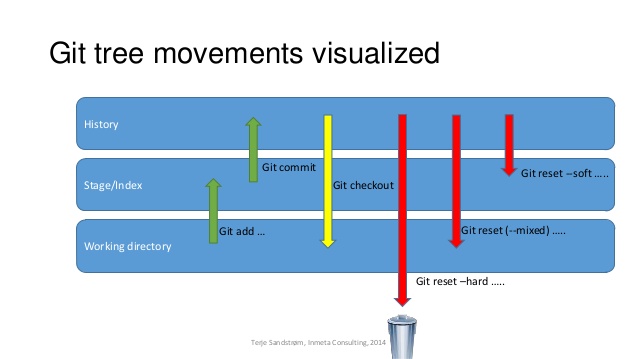
In other words, --soft is discarding last commit, --mix is discarding last commit and add, --hard is discarding last commit,add and any changes you made on the codes which is the same with git checkout HEAD

## Git revert vs git reset ?

difference between git reset and git revert is that git reset will reset the state of the branch to a previous state by dropping all the changes post the desired commit while git revert will reset to a previous state by creating new reverting commits.

**git revert HEAD** - Undo the commit and create a new commit

## What is git reset? How many types of reset is possible? What is the difference?



## how to get back to previous commit ?

git reset --hard

## What is the function of ‘git reset’?

The function of ‘Git Reset’ is to reset your index as well as the working directory to the state of your last commit.

## What is GIT stash?

GIT stash takes the current state of the working directory and index and puts in on the stack for later and gives you back a clean working directory. So in case if you are in the middle of something and need to jump over to the other job, and at the same time you don’t want to lose your current edits then you can use GIT stash.

## What is GIT stash drop?

When you are done with the stashed item or want to remove it from the list, run the git ‘stash drop’ command. It will remove the last added stash item by default, and it can also remove a specific item if you include as an argument.

## What is the function of ‘git stash apply’?

When you want to continue working where you have left your work, ‘git stash apply’ command is used to bring back the saved changes onto the working directory.

## Git Tag

Tags are ref's that point to specific points in Git history. Tagging is generally used to capture a point in history that is used for a marked version release (i.e. v1. 0.1). A tag is like a branch that doesn't change. Unlike branches, tags, after being created, have no further history of commits.