# Version Control System

Version Control System (VCS) is a software that helps software developers to work together and maintain a complete history of their work.

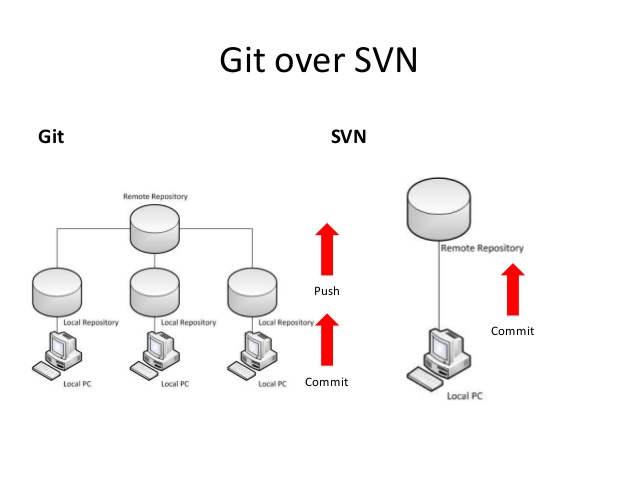
Listed below are the functions of a VCS −

* Allows developers to work simultaneously.
* Does not allow overwriting each other’s changes.
* Maintains a history of every version.

### Following are the types of VCS −

* Centralized version control system (CVCS). SVN
* Distributed/Decentralized version control system (DVCS). Github, gitlab, bitbucket etc.,

SVG VS GIT



### Advantages of Git

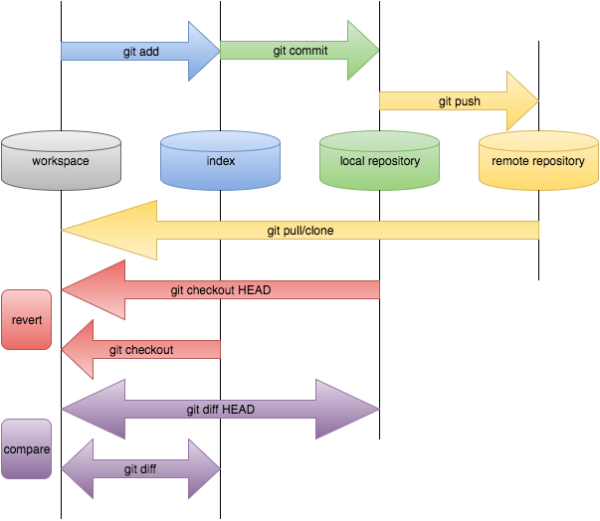
### Free and open source

### Fast and small

### Easier branching

### Implicit backup

# GIT ARCHITECHTURE:



Local Workspace: this is the space local workspace where it contain all the files, folders and can do all the changes and work on it. Once we do all the changes and sure of having the changes in centralized repository (github) we start with the command git add

Command: git add <file name1> <file name 2> (or)  
 git add .(adds all the files in current directory)  
git add –A (adds all the changes in the repository)

Staging/Index: Once the changes are done and sure with the changes we save the data in staging or index. We assign a commit id for reference here in staging.

Command: git commit –m “message for reference”

Local Repository: the files here in Local Repository are ready to push github.

Command: git push

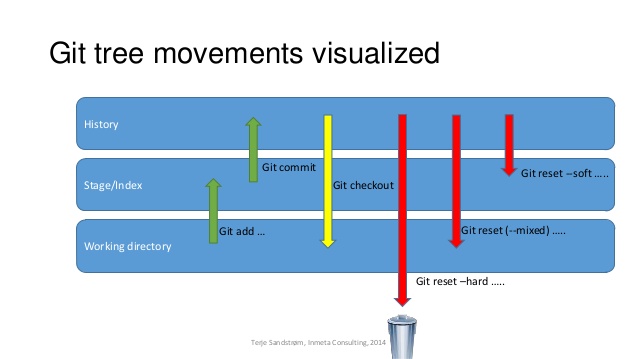
Remote Repository/ Centralised repository: this is the centralized repository(github) where all your code is stored and can be accessed to all the developers and others in the organization.

Git Revert vs Git Reset: These are the commands that are used to

Git Revert: Git revert is used to revert back a commit which is pushed into the github. This is used in a scenario where we need to undo the changes in the github repository.

Command: git revert <the commit id to be reverted>

Git reset: Git reset is also a command which is used to revert back the changes, the difference is git revert will leave the footprint (ie., it will leave a commit id for each and every commit ) where as git reset dosent give any commit id.



Git reset --soft: using the bellow command we can bring back the committed data in back to staging

Git reset --mixed: using the bellow command we can bring back the committed data back to local working dorectory

Git reset --hard: using the bellow command we can bring back the committed data and delete it.

Considering below example and need to bring back cid1

Command: git reset --soft/--hard/--mixed <cid.2>(cid.2 one before cid.1)

## Situation/Example: 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A(cid.1) | B(cid.2) | C(cid.3) | D(cid.4) | E(cid.5) | F(cid.6) | G(cid.7) | H(cid.8) |

A is the latest commit and H is the older one.

* 1. We need to go back to the C commit using revert and rebase.

Using Revert command:

* Doing each revert one at a time

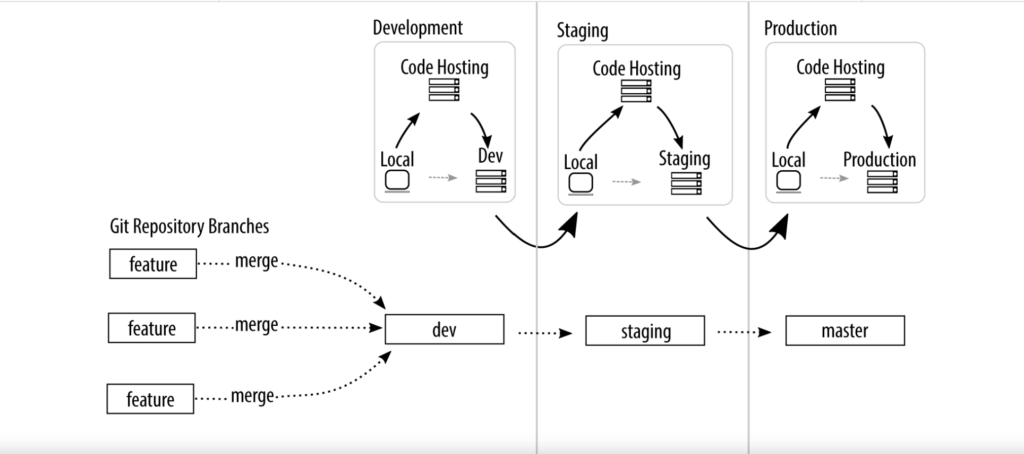
git revert cid.1   
git revert cid.2

* Doing multiple reverts at a time  
  git revert --no-edit <commit-ids>

using reset command: (not recommended)

* we can also go back multiple commits by usig the reset command

git reset <target\_commit\_id >

Git branching Stratergies:,  


Important commands:

* to create a branch 🡪 git checkout –b <branch-name>
* to go from one branch to other 🡪 git checkout <branch-name>
* to know the current branch 🡪 git branch
* to see the all branches 🡪 git branch --all
* Merge 🡪 UI
* Merge 🡪 CLI  
  git merge <branch-name> (from the target branch)

## Git Cherry pick:

To get a specified commit id of some other branch we can use git cherry-pick  
command: git cherry-pick <commit-id> (from the target branch)

Git Stash: when you want to record the current state of the working directory and the index, but want to go back to a clean working directory. The command saves your local modifications away and reverts the working directory to match the initial

Commands:

git stash save (to save the current changes)

Git stash list (to see the information of saved data)

Git stash pop (to bring back the workspace)

Git shash drop (to delete the workspace)

## Git tag:

Commands:

* git tag <tag-name> (create tag)
* git tag <tag-name> -m “message” (create tag with a message)
* git push --tags (push the tags)
* git push origin <tag-name> (push the tag to github)
* git tag –n (list all the tags)

## Version:

V1.2.3.152

V1 🡪 major release

2 🡪 minor release

3 🡪 hot fix/bug fix

152 🡪 internal reference

Roles and Responsibilities in Git:

* maintaining git repositories (as a maintainer)
* giving permissions to the developer and managing roles
* implementing branching strategies based on the requirement
* pushing your work into github (git add commit & push)

GIT Interview Question:

###########

\* What is GIT ?

\* What is difference between GIT &Github ?

\* Why we use GIT ?

\* What is SCM &VCS ?

\* What are the process of pushing the code to GithubRepository ?

\* Why do we commit ?

\* Difference between github and SVN (centralised and distributed repository)

\* What are the commands of GIT to push the code ?

\* How you can merge a git repository with another ?

\* What is branching in git ?

\* Different types of branching in GIT ?

\* What is merge conflict in git ?

\* How you can resolve merge conflict if you are merging same

project and in the same branch ?