GIT – Distributed Version Control System

Local system will have a local copy and working copy. Server will have the remote repository

Trunk Based Development

Implementations : GIT hub, Atlassian Bit Bucket, Git Lab

<https://github.com>

[stk.rajee@gmail.com/Rajee@3483](mailto:stk.rajee@gmail.com/Rajee@3483)

How to merge a code from branch to Trunk(Master).?

Fork – Helps in Contribution

**How to add eclipse project to github**

1. Create GitHub account and sign in
2. Start a project = create a repository and copy the GIT repository url
3. Start eclipse
4. open GIT Repository Perspective and click on clone Git Repository
5. Create a project in Eclipse
6. Do a right click on Project -> Team -> Share ->Add it to git Repo
7. Commit and Push the project to Repository
8. Whenever you make change, you will have to commit and push the changes to Repository

* **GIT stash** : Sometime we do not want to commit our code but we do not want to lose the unfinished code. In this case, we use git stash command to record the current state of the working directory and index in a stash.
* **Stage in GIT** : step before commit. Files are ready for commit
* **Difference GIT clone and GIT Remote** : Git clone is used to create a new local repository by copying another repository from a url. GIT Remote is used in an existing Repository. It adds a new reference to existing remote repository for tracking further changes
* **GIT Version Control** : Manages the changes to the source code over time. Keeps track of all changes.
* **Bare repository in GIT :** git init – bare command is used to create a bare repository. It doesnot contain any working or checked out copy of source files.It stores the git revision history in the root folder of repository instead of in a .git subfolder. Mainly used for sharing purpose. You cant go directly edit in the bare repository as there is no working tree. You can just store the history
* **How do you put a local repository in a GIT hub server.? Refer the steps above “How to add eclipse project to GIT hub”**
* **GIT & SVN :**

Decentralized – GIT is decentralized meaning you have a local copy. You can always commit to it and then push it to remote. In SVN, it is centralized. You have to always connect to the repository for check in

Complex to learn – GIT is difficult to learn. SVN is easier to learn

Unable to handle large binary files – GIT is bot slow when handling large binary file whereas it is not the case with SVN

Internal Directory – GIT creates only one .git directory. SVN creates .svn directory in each folder

UI – GIT doesn’t have. SVN has good user interfaces

* **Advantages of GIT[Distributed Version Control system] over SVN(Centralized Version Control System) :**

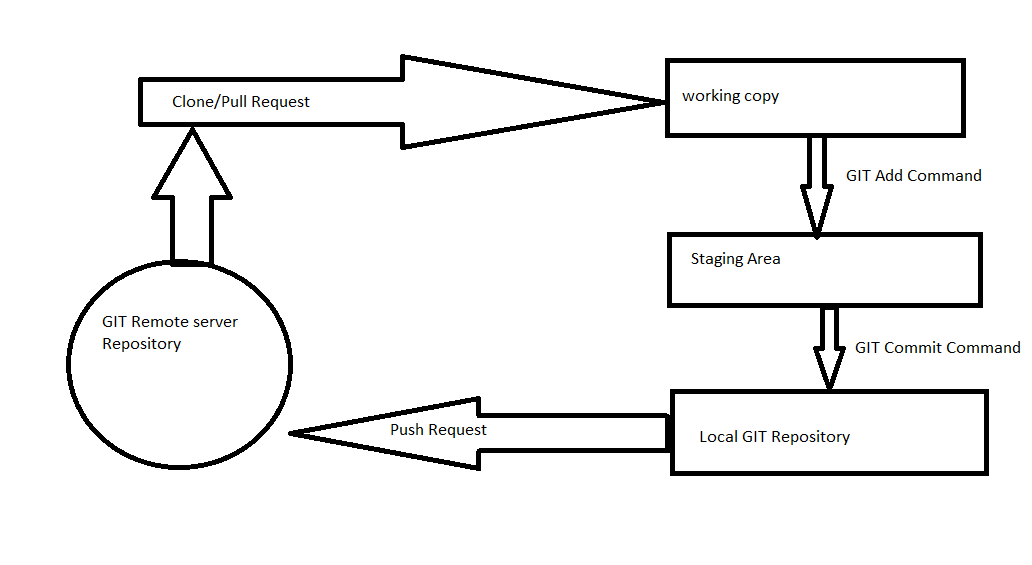
SVN needs internet connection to commit the files to central repo. In case of GIT, no need of internet connection to commit as all operations are done locally. Local repo is clone of central repo.

Central server goes down, developers cant continue with their check ins.

Branching & Merging is easy in GIT

GIT supports non linear development

* **GIT workflow:**



* **GIT Clone vs GIT Pull :** Clone to get the local copy. Pull to update the local copy. Clone is done only once. Pull is done frequently
* **Git -Checkout** – switch branches or restore working tree files
* **Branch vs Tag –** Branch is symbolic name for line of development. Tag is a symbolic name for a revision. Suppose you have a production release and code is released. Now you have tag this code for future reference.