## What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

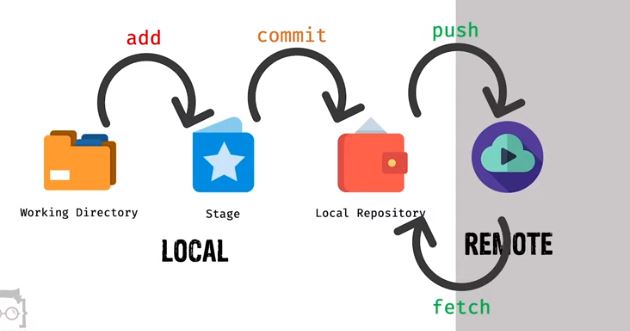
* Tracking code changes
* Tracking who made changes
* Coding collaboration

Git is a version control system.

Git helps you keep track of code changes.

Git is used to collaborate on code.

**Working procedure of git showing below figure**



* To do initialise working directory then types on the git terminal : first go to working directory and command : ***git init***
* To download repository from github : first go to your github repository and copy the repository link then type local machine git terminal :

***git clone (paste copy link)***

* To show any change or modification file or folder types: ***git status***

## **Stage files** are files that are ready to be committed to the repository you are working on.

* Full working directory send to staged : first go to root working directory then types : ***git add - -all or git add - -A***
* If only one directory is sent to staged: first go to this directory and type: ***git add .***
* Back from the stage to working directory type: ***git reset***
* Single file send to staged : ***git add file name or file location***

Ex: ***git add file1.txt or git add myfolder/file2.txt***

* All changes file send to staged: ***git add \****
* To remove file from directory : ***git rm filename***
* To remove forcefully : ***git rm -r filename***
* To remove recursively (for folder) : ***git rm -r filename***
* Not delete file from directory but delete notification send to stage :

Command : ***git rm - -cached filename***

* If you want to back from stage to working directory with deleted file:

Command : ***git reset - -hard***

## Git Commit

Since we have finished our work, we are ready to move from stage to commit for our repo.

Adding commits keep track of our progress and changes as we work. Git considers each commit change point or "save point". It is a point in the project you can go back to if you find a bug, or want to make a change.

When we commit, we should always include a message.

By adding clear messages to each commit, it is easy for yourself (and others) to see what has changed and when.

Command: ***git commit -m "First release of Hello World!"***

The commit command performs a commit, and the -m "*message*" adds a message.

Warning: Skipping the Staging Environment is not generally recommended.

Skipping the stage step can sometimes make you include unwanted changes.

## Git Commit without Stage

Sometimes, when you make small changes, using the staging environment seems like a waste of time. It is possible to commit changes directly, skipping the staging environment. The -a option will automatically stage every changed, already tracked file.

Ex: ***git commit -a -m "Updated index.html with a new line"***

* Back from commit to working directory : ***git reset HEAD~***

## Git Branches

In Git, a branch is a new/separate version of the main repository.

Branches allow you to work on different parts of a project without impacting the main branch.

When the work is complete, a branch can be merged with the main project.You can even switch between branches and work on different projects without them interfering with each other.Branching in Git is very lightweight and fast!.

* Create new branch : ***git branch branch name***
* Switch one branch to another : ***git checkout branch name***
* Merge branch : ***git merge branch name -m “message”***

**Note:** Using the -b option on checkout will create a new branch, and move to it, if it does not exist.

## Merge Conflict

A conflict arises when two separate branches have made edits to the same line in a file/same file, or when a file has been deleted in one branch but edited in the other. Conflicts will most likely happen when working in a team environment.

## **Git push command**

Git push sends the change of local repository to remote repository/github/global.

Command : ***git push***

: ***git push origin branch name***

**Push repository use https**

command : ***git remote add origin URL ( github repository url)***

: ***git remote -v***

: ***git push origin master***

**Push repository using SSH**

First connect with a remote repository.

Go to github > create new repository ( where you want to push) > then copy remote command

***git remote add origin git@github.com:MdsifatHossain/test.git***

* ***ssh-keygen -t ed25519 -C "***[***your\_email@example.com***](mailto:your_email@example.com)***"***
* If you are using a legacy system that doesn't support the Ed25519 algorithm, use:

***ssh-keygen -t rsa -b 4096 -C "your\_email@example.com"***

* start the ssh-agent in the background: ***eval $(ssh-agent -s)***
* Add your SSH private key to the ssh-agent. If you created your key with a different name, or if you are adding an existing key that has a different name, replace *id\_ed25519* in the command with the name of your private key file.

Command : ***ssh-add ~/.ssh/id\_ed25519***

***ssh-add ~/filename or path***

* ***git push -u origin master***

Then copy pub key and from file or command line : ***cat .pub file***

Go to github setting > ssh key generate > add new key > then paste copy key and save.

* Connect local machine file to github repository: copy github repository code link for ssh
* Then go to local machine git bash cli type:
* ***git remote add origin paste copy ssh link***

Ex : *git remote add origin* [*git@github.com*](mailto:git@github.com)*:MdsifatHossain/Git-with-GitHub.git*

Verify connection: ***git remote -v***

## **Git fetch command**

Git fetch download the change of remote repository to local repository.

Command: ***git fetch***

: ***git merge***

## **Git pull command**

Git pull command download full remote repository.

Git pull = fetch + merge