**GIT**

**Introduction:**

Git is an example of a distributed version control system (DVCS) commonly used for open source and commercial software development. DVCSs allow full access to every file, branch, and iteration of a project, and allows every user access to a full and self-contained history of all changes. Unlike once popular centralized version control systems, DVCSs like Git don’t need a constant connection to a central repository. Developers can work anywhere and collaborate asynchronously from any time zone.

**How to install GIT ?**

**In Windows:**

Source : <http://git-scm.com/download/win>

We can download git files from this link and start our installation process.

**In Linux :**

Open terminal and install git by using the following command.

Command : $ sudo apt install git-all

It asks for password and as soon as we enter our password git installation process starts.

**GIT commands and its usage:**

* **git init** : This command initializes a empty git repository and starts tracking on the existing directory.
* **git clone** : Clone creates a local copy of a project that already exists remotely. This includes all the project’s files, history, and branches.
* **git add** : Git adds files to staging area. This is the first part of two step process.  Staging and committing separately gives developers complete control over the history of their project without changing how they code and work.
* **git commit** : This saves the snapshot to the project history and completes the change-tracking process. In short, a commit functions like taking a photo. Anything that’s been staged with git add will become a part of the snapshot with git commit
* **git status** : Status shows the status of changes as untracked, modified, or staged.
* **git branch** : shows the branches being worked on locally.
* **git merge** : merges lines of development together. This command is typically used to combine changes made on two distinct branches.
* **git pull** : updates the local line of development with updates from its remote counterpart. Developers use this command if a teammate has made commits to a branch on a remote, and they would like to reflect those changes in their local environment.
* **git push** :updates the remote repository with any commits made locally to a branch.

**Working with Git :**

Initialize a local repository using init command. This initializes an empty repository.

$ git init

Before committing the changes we need to stage them first. This adds files to our local repository.

$ git add .

Committing changes into git

$ git commit -m "Commit message"

Note : -m option is mandatory while committing a change. –m refers to commit message which is a primary meaning of communication to understand about the project.

Push changes to the local repository in Git ;

$ git push origin master

Master will be default branch which will be pointed to, but we can changes the branch using checkout command.

Eg : $ git checkout develop

**Branching :**

As git is a distributed version control system, we can work on a project on many lines. Branching means deviating from the main line of project and working on any other development branch, just to avoid making the main line messier. Initially the head will be pointing towards the master, but we move on to the other branches by using checkout.

**Checkout :**

To prepare **for** working on <branch>, **switch** to it by updating the index and the files in the working tree, and by pointing HEAD at the branch. Local modifications to the files in the working tree are kept, so that they can be committed to the <branch>.

Command :

$ git checkout -b <branch>