**Documentation of Different Git Commands**

* **Git Undoing Commands:**  There are many different commands to undo changes in the Git repository of your projects. Some of them are mentioned below.
  + **Checkout:** Git Checkout is simply used to move the Head Pointer to a specific commit or a specific branch. It let you navigate between different branches and update the files in the working directory. With Git checkout we can only make changes in the file without changing the commit history. The main difference between Git Checkout and Git Clone is that Git Checkout switch between different versions of code already available in the local history while Git Clone simply fetch the code from the remote repository.
  + **Revert:** Git Revert is used to rollback the changes in the previous commits. It is the safe way to undo the changes in the project. Git Revert doesn’t change or delete any commit instead it adds a new commit in the commit history where the file changes happen. It is the safest and the cleanest way of undoing changes.
  + **Reset:** Git reset is used to move back your repository to the previous commit. Git reset can delete files in the repository and can also delete the commits from the commit history. There are three different modes of git reset.
    - **Soft:** Git reset soft just only move head to the specific commit. It doesn’t delete or change any index or working directory.
    - **Mixed:** It is the default command of the git reset. It moves the head to the specific commit and also undo the changes in the directory and index. It also change the commit history of the git repository.
    - **Hard:** Git reset hard is a dangerous command because it can throw all your committed and uncommitted changes and overwrite the content in the working directory.
  + **Rebase:** Git rebase is used to integrate changes from one branch to another branch. It actually combines a commit or series of commit into a new commit. It allows to rewrite the commit history and allow to merge the commit history of two branches into one.
  + **Merge:** Git merge is used to merge the two branches into the one branch of the git repository of the project.