**Inspect and Undo changes**

**Git Log:**The git log command is a utility tool in Git used to review and read the history of a repository.  
It displays information such as commit hash, author metadata, commit date metadata, and commit title/message.  
The basic usage is: $ git log.

**Git Log Stat:**The --stat option in git log displays information about modified files, including the number of lines added or removed and a summary line of the total records changed.  
Usage: $ git log --stat.

**Git Log Patch:**  
The --patch or -p option in git log displays detailed information about modified files, including the location of added, removed, and updated lines.  
Usage: $ git log --patch or $ git log -p.  
Git Checkout:  
  
The git checkout command is used to switch between different versions of a target entity (files, commits, or branches).  
It operates on files, commits, and branches.  
Be cautious when using it, as there is no undo option.  
Operations include switching to a specific branch, creating and switching to a new branch, and checking out a remote branch.  
Operations on Git Checkout:  
  
**Switch to a Specific Branch:**  
syntax  
$ git checkout <branchname>  
Create and Switch Branch:  
syntax  
$ git checkout -b <branchname>  
Checkout Remote Branch:  
syntax  
$ git checkout <remotebranch>

**Git Diff:**Purpose:  
Inspect Changes:  
Shows the differences between the working directory, staging area, and the last commit.  
Usage:  
  
$ git diff  
Example:  
# Show differences between working directory and last commit  
$ git diff  
  
# Show differences between staging area and last commit  
$ git diff –staged

2. **Git Status:**Purpose:  
Inspect Changes:  
Provides information about the current state of the working directory and staging area.  
Usage:  
bash  
Copy code  
$ git status  
Example:  
bash  
Copy code  
# Check the status of the working directory and staging area  
$ git status

3**. Git Revert:**Purpose:  
Undo Changes:  
Creates a new commit that undoes the changes made in a previous commit.  
Usage:  
  
$ git revert <commit-hash>  
Example:  
# Revert the changes made in a specific commit  
$ git revert abc123

4. **Git Reset:**Purpose:  
Undo Changes:  
Resets the current branch to a specific commit, discarding changes made after that commit.  
Can be used in different modes: soft, mixed, and hard.  
Usage:  
$ git reset <commit-hash>  
Example:  
# Reset to a specific commit and keep changes in the working directory  
$ git reset --soft abc123

5. **Git Rm:**Purpose:  
Inspect Changes:  
Removes a file from both the working directory and the staging area.  
Usage:  
bash  
sntax  
$ git rm <filename>  
Example:  
bash  
syntax  
# Remove a file from the working directory and staging area  
$ git rm myfile.txt  
Additional Notes:  
Git Checkout (for inspection):  
You can use git checkout to inspect changes in a specific commit.  
bash  
syntax  
# Inspect changes in a specific commit  
$ git checkout <commit-hash>

**Collaborating**

**Git Fetch**  
Purpose:  
Downloads commits, objects, and refs from another repository.  
Fetches branches and tags from one or more repositories.  
Updates remote-tracking branches.  
Usage:  
$ git fetch <repository URL>  
$ git fetch <branch URL> <branch name>  
$ git fetch --all  
  
**Git Pull / Pull Request**  
Purpose:  
Git Pull: Fetches and merges changes from the remote server to the working directory.  
Pull Request: Notifies team members that a feature is completed and ready for review and merging.  
  
Usage:  
$ git pull <option> [<repository URL> <refspec>...]  
$ git pull <remote branch URL>  
$ git pull origin master  
  
**Git Push**Purpose:  
Uploads local repository content to a remote repository.  
Transfers commits from local to remote.  
Supports various options like --all, --prune, --mirror, --dry-run, --tags, and more.  
  
Usage:  
$ git push <option> [<Remote URL> <branch name> <refspec>...]  
$ git push origin master  
Additional Notes:

**Git Push Tags:**  
Pushes all local tags to the remote repository.  
Git Force Push:  
  
Used to push local repository to remote without dealing with conflicts.  
Can be risky, use with caution.  
Syntax:  
$ git push <remote> <branch> -f  
$ git push <remote> <branch> --force

**Safe Force Push Repository:**Use the --force-with-lease option to avoid unintentional overwrites.  
Syntax:  
$ git push <remote> <branch> --force-with-lease

**Delete a Remote Branch:**  
Remove a remote branch using git push.  
Syntax:  
$ git push origin --delete <branch name>  
Collaborating in Git involves fetching, pulling, and pushing changes, and understanding these operations is crucial for effective teamwork and version control. Use caution, especially with force pushes, to avoid data loss.