* **cd your-project-directory: Changes the working directory to your project’s root folder.**
* **git init: Initializes a new Git repository in the current directory.**
* **git add .: Stages all files in the directory for the next commit.**
* **git commit -m "Initial commit": Creates a commit with the message “Initial commit”.**
* **git remote add origin remote-repository-url: Adds a remote repository URL where your code can be pushed.**
* **git push -u origin main: Pushes your code to the remote repository and sets the upstream branch.**

**Branching and Merging**

* **git branch**: Lists all local branches.
* **git branch <branch-name>**: Creates a new branch.
* **git checkout <branch-name>**: Switches to the specified branch.
* **git checkout -b <branch-name>**: Creates and switches to a new branch.
* **git merge <branch-name>**: Merges the specified branch into the current branch.
* **git branch -d <branch-name>**: Deletes the specified branch.

**Viewing and Managing Changes**

* **git status**: Shows the status of changes in the working directory.
* **git diff**: Displays differences between working directory and staging area.
* **git log**: Shows the commit history.
* **git log --oneline**: Displays the commit history in a simplified format.
* **git show <commit-hash>**: Shows detailed information about a specific commit.

**Undoing Changes**

* **git reset**: Resets the branch to a specific commit (affects index and working directory).
* **git reset --hard HEAD~1**: Resets to one commit before the latest and discards changes.
* **git revert <commit-hash>**: Creates a new commit that undoes changes from a specific commit.
* **git checkout -- <file>**: Discards changes in the working directory for a specific file.

**Managing Remote Repositories**

* **git remote -v**: Lists remote repositories and their URLs.
* **git fetch**: Retrieves updates from the remote repository without merging.
* **git pull**: Fetches and merges updates from the remote repository.
* **git push**: Uploads local commits to the remote repository.
* **git remote remove <remote-name>**: Removes a remote repository from your configuration.

**Tags**

* **git tag**: Lists all tags in the repository.
* **git tag <tag-name>**: Creates a new tag at the current commit.
* **git tag -d <tag-name>**: Deletes a local tag.
* **git push origin <tag-name>**: Pushes a specific tag to the remote repository.
* **git push origin --tags**: Pushes all tags to the remote repository.

A **random variable** is a fundamental concept in probability and statistics that represents a numerical outcome of a random phenomenon.

**Related Commands**

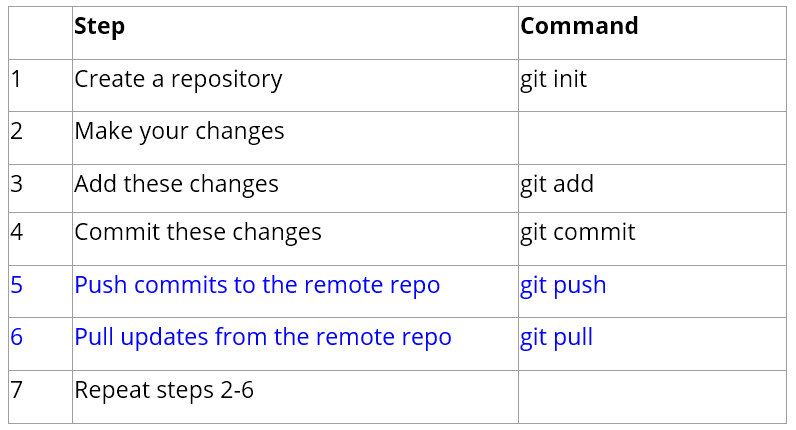
1. **cd <directory>**
   * Changes the current working directory to <directory>.
2. **ls**
   * Lists files and directories in the current directory.
3. **mkdir <directory>**
   * Creates a new directory named <directory>.
4. **touch <file>**
   * Creates a new, empty file named <file>.
5. **rm <file>**
   * Deletes the file named <file>.
6. **cp <source> <destination>**
   * Copies a file from <source> to <destination>.
7. **mv <source> <destination>**
   * Moves or renames a file from <source> to <destination>.
8. **cat <file>**
   * Displays the contents of <file> in the terminal.
9. **echo <text>**
   * Outputs <text> to the terminal or into a file.
10. **find <directory> -name <filename>**
    * Searches for files and directories with <filename> in <directory>.
11. **grep <pattern> <file>**
    * Searches for <pattern> in <file> and displays matching lines.
12. **chmod <permissions> <file>**
    * Changes the permissions of <file> to <permissions>.
13. **chown <user>:<group> <file>**
    * Changes the ownership of <file> to <user> and <group>.
14. **tar -czvf <archive.tar.gz> <directory>**
    * Creates a compressed archive of <directory> named <archive.tar.gz>.
15. **zip <archive.zip> <file1> <file2>**
    * Compresses <file1> and <file2> into a ZIP archive named <archive.zip>.

Removing and Moving Content

git rm command : Removes the file from the working directory

git mv command :A thin convenience command to move a file

To rename a file in Git, run the git mv command as:$ git mv file\_oldNamefile\_newName



**Common Options to git log**

|  |  |
| --- | --- |
| -p | Show the patch introduced with each commit |
| --stat | Show statistics for files modified in each commit |
| --shortstat | Display only the changed/insertions/deletions line from the –stat command |
| --name-only | Show the list of files modified after the commit information |
| --name-status | Show the list of files affected with added/modified/deleted information as well |
| --abbrev-commit | Show only the first few characters of the SHA-1 checksum instead of all 40 |
| --relative-date | Display the date in a relative format (for example, “2 weeks ago”) instead of using the full date format |
| --graph | Display an ASCII graph of the branch and merge history beside the log output |
| --pretty | Show commits in an alternate format. Option values include oneline, short, full, fuller and format (where you specify your own format) |
| --oneline | Shorthand for –pretty=oneline –abbrev-commit used together |

**Options to limit the output of git log**

| **Option** | **Description** |
| --- | --- |
| -<n> | Show only the last n commits |
| --since, --after | Limit the commits to those made after the specified date |
| --until, --before | Limit the commits to those made before the specified date |
| --author | Only show commits in which the author entry matches the specified string |
| --committer | Only show commits in which the committer entry matches the specified string |
| --grep | Only show commits with a commit message containing the string |
| -s | Only show commits adding or removing code matching the string |