# GIT AND GITHUB

## ⋙ Basics

### *Q) What is Git?*

> ⚠ Note:
> Git is a version control system that allows you to track changes to your files and collaborate with others. It is used to manage the history of your code and to merge changes from different branches.

### > Git and Github are different

#### *Git*

1. Git is a version control system that is used to track changes to your files.
2. Github is a web-based hosting service for Git repositories.

#### *Github*

1. It is a free and open-source software that is available for Windows, macOS, and Linux. Remember,GIT is a software and can be installed on your computer.
2. Github is an online platform that allows you to store and share your code with others. It is a popular platform for developers to collaborate on projects and to share code.

Alternatives: Gitlab, bitbucket, Onedev, Codeberg, Gitea , Google cloud source repositories

### > Why do we need version control systems?

- Version control systems are used to manage the history of your code. They allow you to track changes to your files and to collaborate with others.Consider version control as a checkpoint in game. You can move to any time in the game and you can always go back to the previous checkpoint.
- Before Git became mainstream, version control systems were used by developers to manage their code. They were called SCCS (Source Code Control System).

> **.gitkeep file**
>
> It is a file which is kept in a folder which is empty and we want to track it (git doesn't track empty folders)
> For example : here both images and logs are empty folder but as we want to keep track of images we add .gitkeep file into it

## > Basic Git commands

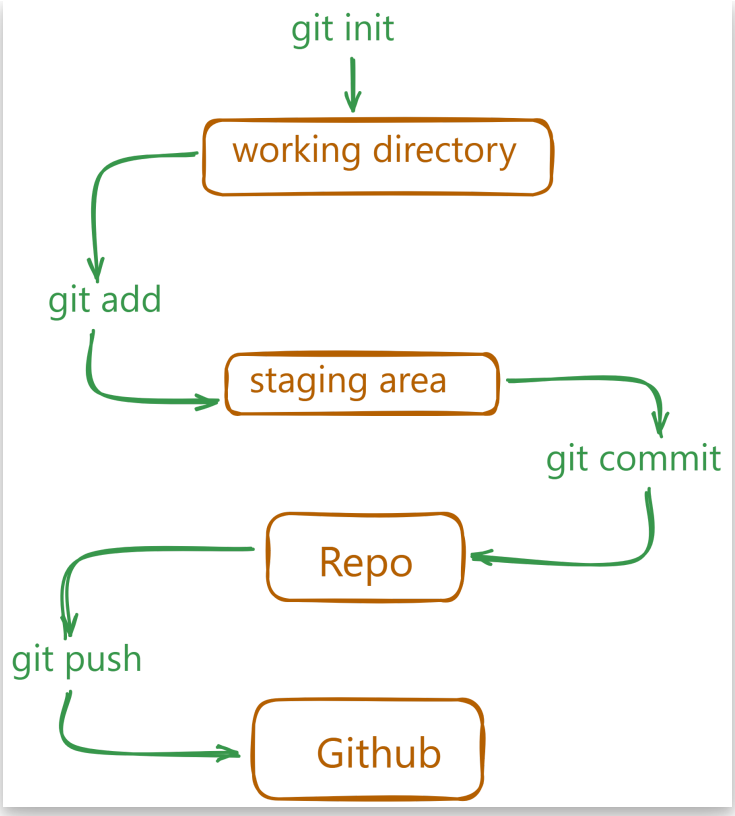
| Command | Description | Example |
|---|---|---|
| `git init` | Initializes a new Git repository | `git init` |
| `git clone <url>` | Clones a repository from a remote source | `git clone https://github.com/user/repo.git` |
| `git status` | Displays the status of the working directory | `git status` |
| `git add <file>` or `git add .` | Adds a file to the staging area | `git add filename.txt` |
| `git commit -m "message"` | Commits the staged changes with a message | `git commit -m "Initial commit"` |
| `git push` | Pushes the committed changes to a remote repo | `git push origin main` |
| `git pull` | Fetches and merges changes from the remote repo | `git pull origin main` |
| `git branch` | Lists all branches in the repository | `git branch` |
| `git checkout <branch>` | Switches to a different branch | `git checkout feature-branch` |
| `git merge <branch>` | Merges a branch into the current branch | `git merge feature-branch` |
| `git log` | Displays the commit history | `git log` |
| `git remote -v` | Shows the remote repositories | `git remote -v` |
| `git fetch` | Downloads objects and refs from another repository | `git fetch origin` |
| `git rebase <branch>` | Reapplies commits on top of another base tip | `git rebase main` |
| `git diff` | Shows the changes between commits, commit and working tree, etc. | `git diff` |
| `git stash` | Stashes changes in a dirty working directory | `git stash` |
| `git tag <tagname>` | Creates a tag for a specific commit | `git tag v1.0.0` |
| `git reset --hard <commit>` | Resets the index and working tree to a specific commit | `git reset --hard abc1234` |

## > Commit

- commit is a way to save your changes to your repository.
- It is a way to record your changes and make them permanent.


## > Complete git flow

A complete git flow, along with pushing the code to github looks like this:




## > Stage

Stage is a way to tell git to track a particular file or folder. You can use the following command to stage a file:

## > gitignore

- Gitignore is a file that tells git which files and folders to ignore.
- It is a way to prevent git from tracking certain files or folders.
- You can create a gitignore file and add list of files and folders to ignore by using the following command:

```
node_modules
.env
.vscode
```

## > Git Snapshots

A git snapshot is a point in time in the history of your code. It represents a specific version of your code, including all the files and folders that were present at that time. Each snapshot is identified by a unique hash code, which is a string of characters that represents the contents of the snapshot.

## > 3 Musketeers of git

The three musketeers of git are:

1. Commit Object
2. Tree Object
3. Blob Object

### ✷ Commit Object

Each commit in the project is stored in .git folder in the form of a commit object. A commit object contains the following information:

- Tree Object

- Parent Commit Object
- Author
- Committer
- Commit Message

### ✷ Tree Object

Tree Object is a container for all the files and folders in the project. It contains the following information:

- File Mode
- File Name
- File Hash
- Parent Tree Object

  ```
  Everything is stored as key-value pairs in the tree object. The key is the file name and the value is the file hash.
  ```

### ✷ Blob Object

Blob Object is present in the tree object and contains the actual file content. This is the place where the file content is stored.