

GIT COMMANDS

There are many different ways to use Git. Git supports many command-line tools and graphical user interfaces. The Git command line is the only place where you can run all the Git commands.

The following set of commands will help you understand how to use Git via the command line.

Basic Git Commands:-

Here is a list of most essential Git commands that are used daily.

1. Git Config command
2. Git init command
3. Git clone command
4. Git add command
5. Git commit command
6. Git status command
7. Git push Command
8. Git pull command
9. Git Branch Command
10. Git Merge Command

1) Git config command:

This command configures the user. The Git config command is the first and necessary command used on the Git command line. This command sets the author name and email address to be used with your commits. Git config is also used in other scenarios.

Syntax:

1. `$ git config --global user.name "ImDwivedil"`
2. `$ git config --global user.email "Himanshudubey481@gmail.com"`

2) Git Init command:

This command is used to create a local repository.

Syntax:

1. `$ git init Demo`

3) Git clone command:

This command is used to make a copy of a repository from an existing URL. If I want a local copy of my repository from GitHub, this command allows creating a local copy of that repository on your local directory from the repository URL.

Syntax:

1. `$ git clone URL`

4) Git add command:

This command is used to add one or more files to staging (Index) area.

Syntax:

To add one file

1. `$ git add Filename`

To add more than one file

1. `$ git add*`

5) Git commit command:

Commit command is used in two scenarios. They are as follows.

Git commit -m

This command changes the head. It records or snapshots the file permanently in the version history with a message.

Syntax:

1. `$ git commit -m "Commit Message"`

Git commit -a

This command commits any files added in the repository with git add and also commits any files you've changed since then.

Syntax:

1. `$ git commit -a`

6) Git status command:

The status command is used to display the state of the working directory and the staging area. It allows you to see which changes have been staged, which haven't, and which files aren't being tracked by Git. It does not show you any information about the committed project history. For this, you need to use the git log. It also lists the files that you've changed and those you still need to add or commit.

Syntax:

1. `$ git status`

7) Git push Command:

It is used to upload local repository content to a remote repository. Pushing is an act of transfer commits from your local repository to a remote repo. It's the complement to git fetch, but whereas fetching imports commits to local branches on comparatively pushing exports commits to remote branches. Remote branches are configured by using the git remote command. Pushing is capable of overwriting changes, and caution should be taken when pushing.

Git push command can be used as follows.

Git push origin master

This command sends the changes made on the master branch, to your remote repository.

Syntax:

1. `$ git push [variable name] master`

Git push -all

This command pushes all the branches to the server repository.

Syntax:

1. `$ git push --all`

8) Git pull command:

Pull command is used to receive data from GitHub. It fetches and merges changes on the remote server to your working directory.

Syntax:

1. `$ git pull URL`

9) Git Branch Command:

This command lists all the branches available in the repository.

Syntax:

1. `$ git branch`

10) Git Merge Command:

This command is used to merge the specified branch's history into the current branch.

Syntax:

1. `$ git merge BranchName`