Git_commands

Top Git Commands Frequently I used.

Git is a free and open – source distributed version control system for tracking changes in any sets of files, usually used for coordinating work from small to very large project with speed and effectively. With it, we can know who did what, when, and why. Nowadays, Git has become a must-have tool for any developer, programmer, graphical designer and many other mores. Knowing Git commands is essential for developers to use Git to it's full Potential. Enlisting hundreds of Git commands, but only few significant commands are used regularly.

In this article, I will explain the commands that I usually used the most for the next level development.

- Git init
- Git status
- Git add
- Git branch
- Git checkout
- Git commit
- Git push
- Git pull
- Git merge
- Git diff
- Git reset
- Git log
- Git clone

#1 Git Init:

This git command will let you create a new repository. It can be used to convert an existing, un-versioned project to a Git repository or initialize a new, empty repository. This is usually the first command you'll run in a new project. But you can't see the **Init** file in your project folder, it's remained hidden.

Example:

```
$ git init
```

#2 Git Status:

This git command will display the state of the working directory and the staging area. It lets you see either your file is either your file is in staged or not, which file aren't being

Git_commands

tracked but Git. Shortly Git provides an overview of the current status of your repository.

Example:

```
$ git status
```

#3 Git add:

This git command Add your file or project in staging area. It tells Git that you want to include updates to a particular file in the next commit. But you have to add your file or project before commit your new or modified file.

Example:

```
$ git add <file-name>
```

This command adds your specific file only.

Or

```
$ git add .
```

This command adds your all file in git.

#4 Git branch:

This Git commands let's us add a branch to an existing **master** branch**,** viewing all existing branches, and delete a branch. There's already a **master(main)** branch in your project.

Example:

```
$ git branch <branch-name>
```

Create a new branch.

Or

```
$ git branch a
```

List all the remote and locate branches.

Or

```
$ git branch or git branch --list
```

View all branches and see which branch you're currently working on:

Or

```
$ git branch -d <branch-name>
```

Delete a branch.

Or

```
$ git branch -m <branch-name><new-branch-name>
```

Rename a branch.

#5 Git checkout:

This git command allows us to switch to an existing branch to next branch.

Example:

```
$ git checkout <branch-name>
```

This command adds your specific file only.

Or

```
$ git checkout -b <new-branch-name>
```

This command adds your all file in git.

Git_commands

#6 Git commit:

Every time you commit your code, you have to include a brief description of the change made. Before pushing to repository and after add the file to staging are this command is usually used. This commit message helps others understand the changes that have been done.

Example:

```
$ git commit -a
```

Comment any files added with git

Or

```
$ git commit -m "<Msg>"
```

Comment file with message.

Or

```
$ git commit -am "<Msg>"
```

Replace the two previous commands

Or

```
$ git commit –amend -m "<Msg>"
```

Delete a branch.

#7 Git Push:

This git command push the local file change to the remote repository. It also creates a branch automatically names as a **Master branch** if any branch doesn't exist**.**

Example:

```
$ git push or $ git push <remote> <branch-name>
```

Push with the branch name.

Or

```
$ git push origin master.
```

Push the file to master branch.

#8 Git Pull:

This git command achieves the last uploaded changes from the remote server to the local repository to get the latest updates of the remote repository.

Example:

```
$ git pull or $ git pull origin master
```

#9 Git merge:

This git command merges your created or added branches with the master branches. It lets you take the independent lines of development created by git branch and integrate them into a single branch.

Git_commands

Example:

```
$ git merge <branch-name>
```

Merge your branch

#10 Git diff:

This git command Show changes between commits, commit and working tree, etc. Mainly I used to see the either I forget to upload any changes to repository.

Example:

or

```
$ git diff --staged
```

#11 Git reset:

Git reset is a powerful command that is used to undo local changes to the state of a Git repo. It will undo the last change you have done on git repo.

Example:

#11 Git log:

Git log command shows a list of all the commits made to a repository.

Example:

```
$ git log
```

or

```
$ git log --oneline
```

- **Shows log in one-line**

#11 Git clone:

Git clone is primarily used to point to an existing repo and make a **clone** or copy of that repo at in a new directory, at another location. Means, in simple this command let you download your project in your local machine.

Example:

```
$ git clone https://github.com/.......... .git
```