Learn GIT Basics

Full Course at https://sdet.live/git

Git Notes - TheTestingAcademy (Pramod Sir)

Install Git into the System

1. Linux - Simply open up a new terminal and install Git via your distribution's package manager. For Ubuntu, the command is: **sudo apt-get install git**

2. Windows - we recommend git for windows as it offers both a GUI client and a BASH command line emulator.
3. OS X - The easiest way is to install Homebrew, and then just run **brew install git** from your terminal.

What is GIT?

- Git is a version control system that is used for tracking changes to files.
- It does this through a series of **snapshots of your project.** It works with those snapshots to version and manage your source code, and it does this in a simple way.

Why Use Git?

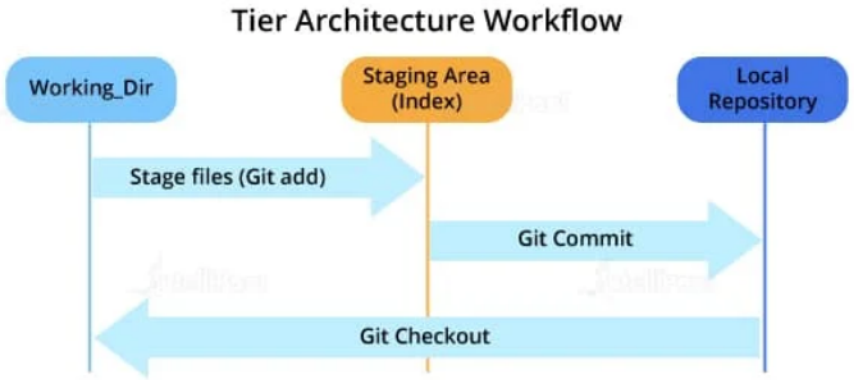
- Can work offline.
- Collaborating with others is easy!
- Branching is easy!
- Branching is fast!
- Merging is easy!
- Git is fast.
- Git is flexible.

GIT vs Github vs Gitlab

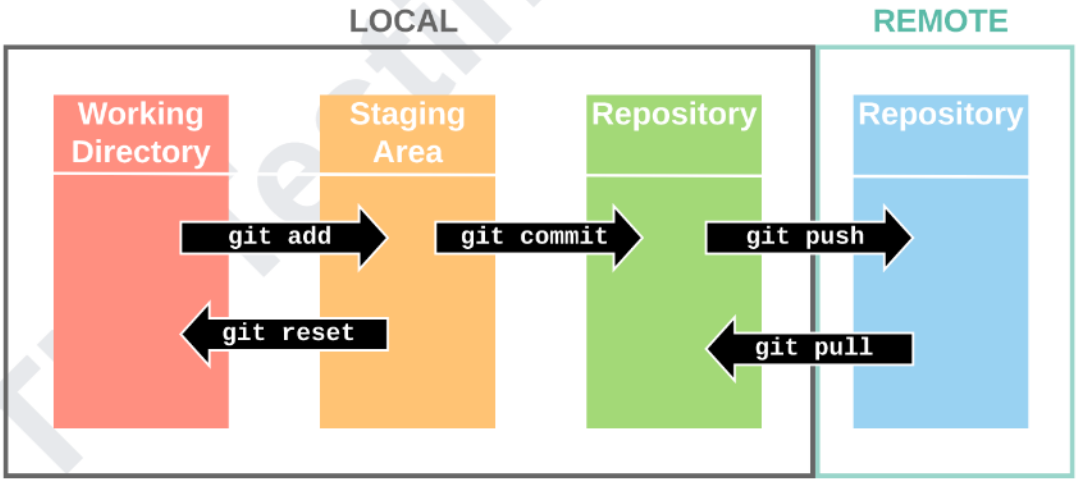
- Git is a version control system, while GitHub and GitLab are web-based Git repositories.
- GitHub is a company that provides Git hosting, and it offers both a cloud-based hosting service and on-premises enterprise versions.
- GitLab is similar to GitHub, but it is an open-source Git hosting platform. It provides a web-based interface for working with Git repositories, as well as a range of tools for collaboration, project management, and continuous integration.

Git is a version control system, GitHub is a Git hosting service, and GitLab is an open-source Git hosting platform.

Git Architecture



With Remote Repo



Configuring Git

```
$ git config --global user.name "My Name"
$ git config --global user.email myEmail@example.com
```

.git Directory

The .git directory contains all the configurations, logs, branches, HEAD, and more

Commands

Creating a new repository

```
git init
```

Checking the status

```
git status
```

Staging

- Git has the concept of a "staging area".
- You can think of this like a blank canvas, which holds the changes which you would like to commit.
- It starts out empty, but you can add files to it (or even single lines and parts of files) with the git add command, and finally commit everything (create a snapshot) with git commit

```
git add
git add hello.txt
git add . // Add everything
git status
```

Commiting
A commit represents the state of our repository at a given point in time.
It's like a snapshot, which we can go back to and see how thing were when we took it.

```
git commit -m "Initial commit."
```

Blame

Examine specific parts of the code's history and find out who was the last author to modify that line "`bash.
Show what revision and author last modified each line of a file
https://git-scm.com/docs/git-blame

Remote repositories

```
git remote add
```

Connecting to a remote repository

1. In order to upload something to a remote repo.
2. Create a Repo at Github.com and Add Remote to it.
3. The Git command to do this is git push and takes two parameters.
   the name of the remote repo (we called ours origin) and the branch to push to

```
git remote add origin https://github.com/PramodDutta/Restfulbooker.git
```

Add File and Push

```
git add .
git commit -m “blah blah”
git remote add origin https://github.com/PramodDutta/Restfulbooker.git
git push origin main
```

Cloning a repository

- Download locally and have a fully working copy of your project

```
git clone https://github.com/PramodDutta/Restfulbooker.git
```

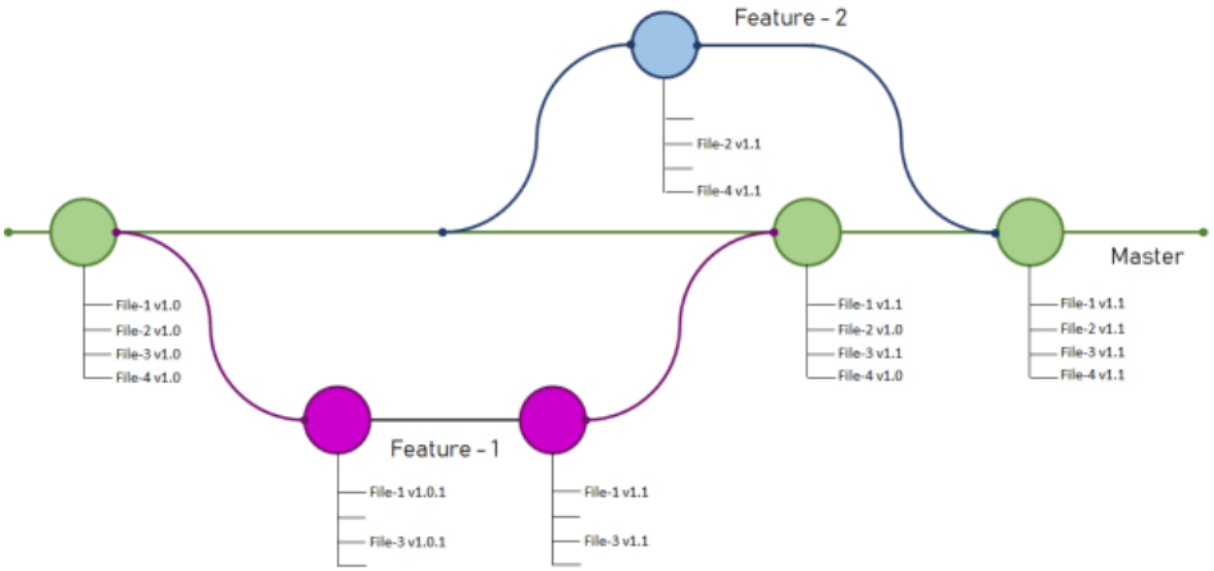
Getting changes from a server

If you make updates to your repository, people can download your changes with a single command

```
git pull origin master
```

Branches

- Copy of the original project, called a branch.
- Developers can work on their own branch, without the risk of their codebase changing due to someone else's work.
- An already working, stable version of the code won't be broken.
-



Creating new branches

```
git branch new_feature
```

Switching branches

```
git checkout new_feature
```

Merging branches

```
git add feature.txt
git commit -m "New feature added"
git checkout main
git merge
// Remove old branch
git branch -d amazing_new_feature
```

Merge and Rebase

https://www.geeksforgeeks.org/git-difference-between-merging-and-rebasing/

TEMPORARY COMMITS

```
git stash
//Save modified and staged changes
```

```
git stash list
//list stack-order of stashed file changes
git stash pop
//write working from top of stash stack
git stash drop
//discard the changes from top of stash stack
```

Little Advanced

1. Setting up .gitignore
    a. log files
    b. task runner builds
    c. the node_modules folder in node.js projects
    d. folders created by IDEs like Netbeans and IntelliJ
    e. personal developer notes

2. Check the diff between commits, use the

```
git log
```

3. Want to see in that commit, diff?

```
git show <commit Id>
```

4. Want to see diff from 1 to 2 ids

```
git diff 09bd8cc..ba25c0ff
```

5. Reverting a file to a previous version.

```
git checkout 09bd8cc1 hello.txt
```

6. Fixing a commit and Revert to Head
    a. newest commit can be accessed by the HEAD alias.

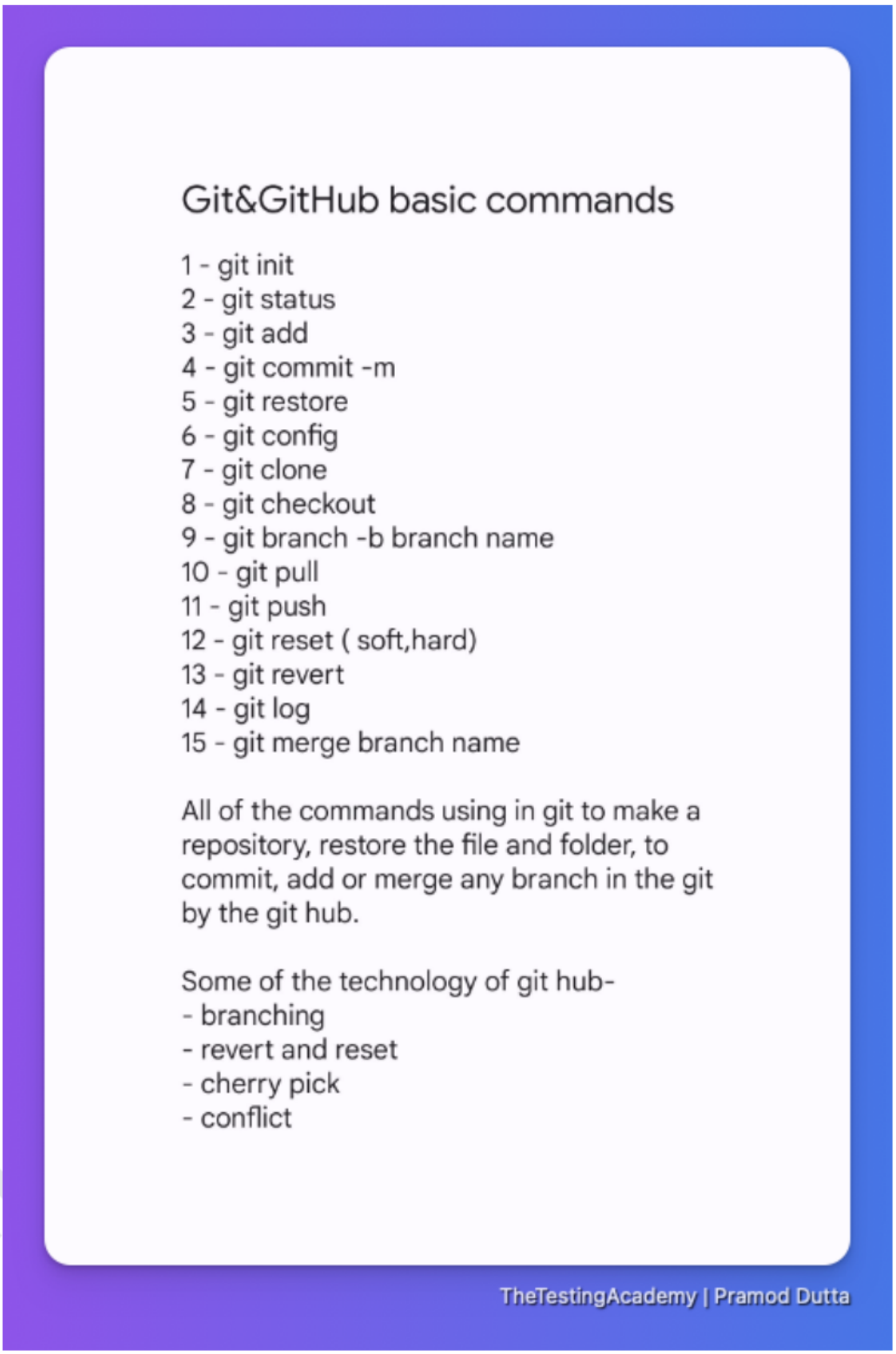
```
git revert HEAD
git revert b10cc123 // Or
```

7. Resolve conflict
    a. git merge tim_branch

```
<<<<<<< HEAD
// Use a for loop to console.log contents.
for(var i=0; i<arr.length; i++) {
    console.log(arr[i]);
}
=======
// Use forEach to console.log contents.
arr.forEach(function(item) {
```

```
    console.log(item);
});
>>>>>>> Tim's commit.

$ git add -A
$ git commit -m "Array printing conflict resolved."
```



Reference and Download PDF

- https://education.github.com/git-cheat-sheet-education.pdf

- https://tutorialzine.com/2016/06/learn-git-in-30-minutes
- https://www.geeksforgeeks.org/git-difference-between-merging-and-rebasing/
- https://learnxinyminutes.com/docs/git/

Git Interview QnA