

Day 9 Task: Deep Dive in Git & GitHub for DevOps Engineers.

1. What is Git and why is it important?

Git is a mature, actively maintained free and open-source project originally developed in by Linus Torvalds, the famous creator of the Linux operating system kernel. Git is a version control system that allows you to track changes to files and coordinate work on those files among multiple people with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning-fast performance. It means that you can keep a record of who made changes to what part of a file, and you can revert back to earlier versions of the file if needed. Git also makes it easy to collaborate with others, as you can share changes and merge the changes made by different people into a single version of a file. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows

2. What is difference Between Main Branch and Master Branch??

Git is often used to make branches, where each programmer takes a latest cut of the codebase, makes their changes, then merges it back to the top copy branch.

A branch is essentially is a unique set of code changes with a unique name. Each repository can have one or more branches.

Every branch needs a unique name. The one given to the 'top copy' is obviously special to your team, because it represents the most complete, most recent (hopefully working) copy of your codebase.

It used to be called master, in the sense of a 'Master Copy' of a recording. It is quite common in technology for 'primary controllers' or 'top copies' to be called a 'master'. The main branch the one where all changes eventually get merged back into, and is called master.

But that name can be viewed as having come from slavery terminology. The master told their slaves what to do - or else. To avoid upset, the term 'main' can be used, without loss of meaning or any kind of emotional baggage. It is increasingly common to see a 'main' branch used to name the top copy.

3. Can you explain the difference between Git and GitHub?

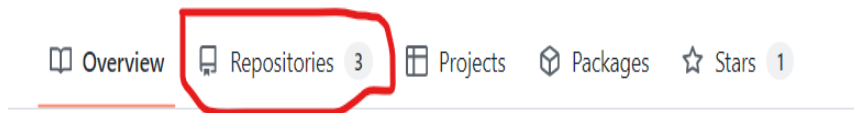
Git is a version control system that allows you to track changes to files and coordinate work on those files among multiple people with speed and efficiency.

GitHub is a product that allows you to host your Git projects on a remote server somewhere (or in other words, in the cloud)

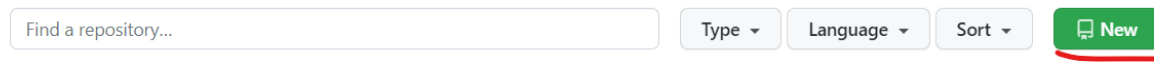
4. How do you create a new repository on GitHub?

- Create a github account (refer this link to create [GitHub](#))

- There is an option in main bar called Repositories, click there



- Click to New option




- Now, entire repository name and choice public or private as per your convent and click on create repository.

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * Repository name *

 Basanagoudapatil02 ▾ /

Great repository names are short and memorable. Need inspiration? How about **bookish-train**?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

 You are creating a public repository in your personal account.

Create repository

- Congratulations, your new repository created successfully.

5. What is difference between local & remote repository? How to connect local to remote?

Local repository : Local repositories reside on the computers of team members
Eg: linuxOs, windownsOs, macOs etc

Remote repository : Remote repositories are hosted on a server that is accessible for all team members - most likely on the internet or on a local network. Eg: Github, Gitit etc

how to connect local to remote, we will see in task-2 please refer below.

Tasks

Task-1: Set your user name and email address, which will be associated with your commits.

Command :

- `git init`
- `git config --global user.name <name>`
- `git config --global user.email <mail id>`
- `git config --list` (to check details)

```
ubuntu@ip-172-31-57-94:~/git_demo$ git config --global user.name "Omkar"
ubuntu@ip-172-31-57-94:~/git_demo$ git config --global user.email "Omkar.gmail.com"
ubuntu@ip-172-31-57-94:~/git_demo$ git config --list
user.name=Omkar
user.email=Omkar.gmail.com
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
ubuntu@ip-172-31-57-94:~/git_demo$
```

Task-2:

- Create a repository named "Devops" on GitHub

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner * Basanagoudapatil02 / Repository name * DevOp ✓

Great repository names are short and DevOp is available. inspiration? How about congenial-lamp?

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

ⓘ You are creating a public repository in your personal account.

Create repository

- Create a new file in Devops/Git/Day-02.txt & add some content to it
 - Create a file called Day-02.txt
 - Add it to git using command **git add filename**

- Make it commit using command **git commit -i m filename**

```
ubuntu@ip-172-31-90-202:~/demo$ nano Day-02.txt
ubuntu@ip-172-31-90-202:~/demo$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Day-02.txt

nothing added to commit but untracked files present (use "git add" to track)
ubuntu@ip-172-31-90-202:~/demo$ git add Day-02.txt
ubuntu@ip-172-31-90-202:~/demo$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Day-02.txt

ubuntu@ip-172-31-90-202:~/demo$ git commit -m Day-02.txt
[master 00525db] Day-02.txt
 1 file changed, 2 insertions(+)
 create mode 100644 Day-02.txt
ubuntu@ip-172-31-90-202:~/demo$
```

- Push your local commits to the repository on GitHub
 - To push use command **git push origin master**
 - Then entre your username and password

```
ubuntu@ip-172-31-57-94:~/git_demo$ git push origin master
Username for 'https://github.com': Basanagoudapatil02
Password for 'https://basanagoudapatil02@github.com:': 
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (10/10), 828 bytes | 828.00 KiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0
```