Vinod Kumar

  LinkedIn : https://www.linkedin.com/in/vinoddkumardh/

Git & GitHub- Hand Note

# Git & GitHub-Key Concepts & Commands!

1. **Git**

Git is a tool that helps you track changes in your code. It allows multiple people to work on the same project without messing things up. You can also go back to previous versions if needed.

* It allows developer to collaborate and track the code history.

**In short: Git is a tool that helps us to track changes in your code. Git is a Version Control System. Which is:**

* **Most popular among the version control tools**
* **Free & Open Source.**
* **Fast & Scalable.**

**Setting Up Git on Your Local Machine 🚀**

To start using Git on your computer, you’ll need to install the following tools:

**1.Visual Studio Code**

**Visual Studio Code (VS Code)** as it provides a great environment for working with git  
Download VS Code: <https://code.visualstudio.com/>

**2️. install Git**

**For Windows Users:**

* Download and install **Git Bash** from <https://git-scm.com/downloads>.
* Git Bash provides a command-line interface like Linux/Mac for running Git commands.

**For Mac Users:**

* **No need to install Git separately!** Mac already comes with Git preinstalled.
* You can check if Git is available by running this command in **Terminal**:

git --version

If Git is missing install it using Homebrew (For Mac only)

brew install git

After installation, check if Git is working correctly or not by running “git –version” command.

**Next, Configure Git with your information:**

* Before we start using Git, set up your name and email which will helps track commits

git config --global user.name "Your Name"

git config --global user.email “your-email@example.com”

To check if updated or not, run:

git config --list (It will give username and email if setup successfully!)

With this, Git is successfully set up on our machine.

1. **GitHub**

GitHub is a website where you can store and share your code. It works with Git to keep your project safe, help you collaborate with others, and track changes easily.

* In short, it is a website that allows dev’s to store and manage code using Git.
* You can think of it like Google Drive but for Coding!

**Note: Git and GitHub are not the same!**

**Git –** Is like a notebook where you can write and track your work

**GitHub-**Is like Google Drive where you can store and share your notebook with others.

They work together; however, they are not the same!

**GitHub Repository or GitHub Repo:**

A GitHub repository (repo) is like a folder where your project files, code, and history can be stored on GitHub. It helps you organize and share your work with others.

**Types of Repositories:**

1. Public Repository – Anyone can see the code
2. Private Repository – Only you (or Invited People) Can see it.

We can create GitHub account and repository from the following link https://github.com

Your GitHub account may look like below, once it is customized with required details.

A screenshot of a computer

AI-generated content may be incorrect.

One can create repository on the GitHub profile clicking on the Repositories tab and give basic details like repo name description etc. Follow the below image.

Click on the Repositories then click on New as shown with arrows below

A screenshot of a computer

AI-generated content may be incorrect.

Enter the basic details

A screenshot of a computer

AI-generated content may be incorrect.

There is option “Initialize this repository with: If you want you can click the add a README file.

Add a README File: A **README** file is a text file (usually named README.md) that gives information about your project. It is the first thing that we see when we visit our GitHub repository.

**Key Command: “Commit”**

Once we create a repository which is a change can be committed with Commit Command.

A **commit** is like a **saving the changes** in our project. Every time that we make any changes need to commit them; Git stores a snapshot of your project at that moment.

**Syntax**

To commit changes, use:

**git commit -m “Our Commit Message”**

“-m”, stands for message and the message should briefly say about changes that we made or stage of the project.

1. **How to work with Git on GitHub Repositories (Basic Work Flow)**
2. **Cloning a Repository:**

**What It Does:** This command copies a repository (a project) from GitHub to your computer. You’ll get all the files and the project history.

Syntax; **git clone <repository-url>**

1. **Checking the Status:**

**What It Does:** displays the current state of the code

Syntax; **git status**

1. **Create a New Branch:**

Before making changes, it’s best practice to create a new branch to keep your work organized:

Syntax: **git checkout -b <new-branch-name>**

**Make Changes**: Edit the files in your local repository using your preferred code editor.

1. **Stage Changes:**

After making changes, you need to stage them before committing:

Syntax: **git add <File name> (**We can stage all changes by using “git add. “, command. This command stages all modified and new files.)

1. **Commit Changes:**

**O**nce your changes are staged, commit them with a descriptive message:

Syntax: **git commit -m "Your commit message describing the changes"**

1. **Pull Latest Changes:**

Before pushing your changes, it’s a good idea to pull the latest changes from the remote repository to avoid conflicts:

Syntax: **git pull origin <branch-name>**

Example: git pull origin main

This command fetches and merges changes from the specified branch.

1. **Push Changes:**

After ensuring your branch is up to date, you can push your commits to the remote repository.

Syntax: **git push origin <new-branch-name>**

Example: git push origin main. (This command uploads your local commits to the remote repository.)

1. **Merge Changes:**

This step is optional. (If you’re ready to add/integrate your changes into the main branch, you can create a pull request (PR) on platforms like GitHub. Alternatively, if you’re working locally, switch to the main branch and merge:

Syntax: **git checkout main**

Syntax: **git merge <new-branch-name>**

1. **Delete the Branch (Optional Step) :**

Once your changes are merged, you can delete your feature branch.

Syntax: **git branch -d <new-branch-name>**

**Summary of Commands of Basic Workflow:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Thank you .**

**Git Resources**

1. **Pro Git Book**[**https://git-scm.com/book/en/v2**](https://git-scm.com/book/en/v2)
2. **Git Official Documentation**[**https://git-scm.com/doc**](https://git-scm.com/doc)
3. **Git Tutorial by Atlassian  
   https://www.atlassian.com/git/tutorials**
4. **Learn Git Branching  
   https://learngitbranching.js.org/**

**GitHub Resources**

1. **GitHub Guides**[**https://guides.github.com/**](https://guides.github.com/)
2. **GitHub Learning Lab**[**https://lab.github.com/**](https://lab.github.com/)
3. **GitHub Docs**[**https://docs.github.com/en**](https://docs.github.com/en)
4. **Introduction to GitHub (Video Series)**[**https://www.youtube.com/playlist?list=PLg7yL1J2U2d\_8G9t7l4I0n6hxQpF-5bQO**](https://www.youtube.com/playlist?list=PLg7yL1J2U2d_8G9t7l4I0n6hxQpF-5bQO)

**Additional Resources**

1. **Codecademy: Learn Git Course  
   https://www.codecademy.com/learn/learn-git**
2. **Git Cheat Sheet**[**https://education.github.com/git-cheat-sheet-education.pdf**](https://education.github.com/git-cheat-sheet-education.pdf)

**Reference List**

1. Chacon, S., & Straub, B. (2014). *Pro Git*. Retrieved from <https://git-scm.com/book/en/v2>
2. Git SCM. (n.d.). *Git Official Documentation*. Retrieved from <https://git-scm.com/doc>
3. Atlassian. (n.d.). *Git Tutorial*. Retrieved from https://www.atlassian.com/git/tutorials
4. Learn Git Branching. (n.d.). *Learn Git Branching*. Retrieved from https://learngitbranching.js.org/
5. GitHub Guides. (n.d.). *GitHub Guides*. Retrieved from <https://guides.github.com/>
6. GitHub. (n.d.). *GitHub Learning Lab*. Retrieved from <https://lab.github.com/>
7. GitHub. (n.d.). *GitHub Documentation*. Retrieved from <https://docs.github.com/en>
8. GitHub. (n.d.). *Introduction to GitHub* [Video Series]. Retrieved from <https://www.youtube.com/playlist?list=PLg7yL1J2U2d_8G9t7l4I0n6hxQpF-5bQO>
9. Codecademy. (n.d.). *Learn Git Course*. Retrieved from https://www.codecademy.com/learn/learn-git
10. GitHub Education. (n.d.). *Git Cheat Sheet*. Retrieved from <https://education.github.com/git-cheat-sheet-education.pdf>