Git

What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

What does Git do?

* Manage projects with **Repositories**
* **Clone** a project to work on a local copy
* Control and track changes with **Staging** and **Committing**
* **Branch** and **Merge** to allow for work on different parts and versions of a project
* **Pull** the latest version of the project to a local copy
* **Push** local updates to the main project

Working with Git

* Initialize Git on a folder, making it a **Repository**
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered **modified**
* You select the modified files you want to **Stage**
* The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

Why Git?

* Over 70% of developers use Git!
* Developers can work together from anywhere in the world.
* Developers can see the full history of the project.
* Developers can revert to earlier versions of a project

### What is GitHub?

* Git is not the same as GitHub.
* GitHub makes tools that use Git.
* GitHub is the largest host of source code in the world, and has been owned by Microsoft since 2018.
* In this tutorial, we will focus on using Git with GitHub.

## **Git Install**

You can download Git for free from the following website: [https://www.git-scm.com/](https://git-scm.com/)

## **Using Git with Command Line**

To start using Git, we are first going to open up our Command shell.

For Windows, you can use Git bash, which comes included in Git for Windows. For Mac and Linux you can use the built-in terminal.

The first thing we need to do, is to check if Git is properly installed:

Use commands to check:

git --version

git version 2.30.2.windows.1

If Git is installed, it should show something like git version X.Y

## **Configure Git**

Now let Git know who you are. This is important for version control systems, as each Git commit uses this information:

git config --global user.name "Dhanesh-Thampi"

git config --global user.email "dhanesh.thampi@beinex.com"

## **Creating Git Folder**

mkdir myproject

cd myproject

mkdir **make**s a **new directory**.

cd **changes** the **current working directory**.

## 

**Git commands:**

1. git init

Initializes a new Git repository in the current directory.

2. git clone<repository URL>

Copies a repository from remote server to your local machine.

3. git add <file>

Adds changes in a specific file to staging area, preparing them to be commit.

4. git add

Adds changes in the current directory to staging area.

5. git commit –m “commit message”

Records changes staged in the current branch with a descriptive message.

6. git push

Uploads local repository content to a remote repository.

7. git pull

Fetches and downloads content from a remote repository and updates the local repository.

8. git status

Displays the status of changes as untracked, modified or staged.

9. git log

Displays a history of commits.

10. git branch

Lists all existing branches and indicates the current active branch.

11. git checkout<branch>

Switches to specific branch.

12. git merge<branch>

Merges changes from the specific branch into the currently active branch.

13. git remote –v

Lists all remote operations associated with the local respository along with their URLs.

14.git remote add<name><URL>

Adds a new remote repository.

15. git remote rm<name>

Removes a remote repository.

16. git fetch

Fetches changes from a remote repository without merging them into the current branch.

17. git reset<file>

Unstages changes for a specific file.

18. git reset –hard

Discards all changes and resets the repository to the state of the last commit.

19. git diff

Shows the difference between working directory, staging area and last commit.

20. git tag

Lists, creates or deletes tags (labels) to mark specific points in history.