**Git** is a popular **version control system**. It was created by Linus Torvalds in 2005.

It is used for: <https://www.w3schools.com/git/git_intro.asp>

* 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 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!

**What is GitHub:** Git is not the same as GitHub. GitHub is a tool that uses Git software.

It is the largest host of source code in the world and has been owned by Microsoft since 2018. It allows us to create, store, track and collaborate code on software projects. **It uses Git software** providing the **distributed version control** of Git, access control, bug tracking, software feature requests, task management and continuous integration.

1. Install Git: <https://git-scm.com/download/win>
2. Launch command prompt and check git version ( **git - -version**)
3. Create **New** repository in github.com or **git init** project in local host using cmd prompt.
4. Create a directory where you want to save this project folder.
5. Check the files in our current working directory using cmd terminals: **git status.**
6. **Clone** a project to local host one time using cmd terminals.
   * Create a new branch and work in our local repository.
     + Create new branch: **git branch** xyzbranchname.
     + Check how many branch there currently**: git branch**
     + Switch to new branch: **git checkout** name
7. Git Staging environment:
   * **Adding all changes command: git add .**
   * **Add only changes to specific file: git add filename**
8. Move from stage to commit to master repo. When we commit, we should always include a message. It is easy to see what has changed and when.
   * Command: **git commit -m “xyz message”**
9. To push local ref (branch) to remote ref by setting up upstream for the local branch:
   * **git push –set-upstream origin** xyzbranchname
   * If upstream is already in remote github then use only: **git push  
     (**The git push command is used to upload local repository content to a remote repository)
10. Create merge request MR)/ pull request from GitHub.
    * Merge pull request.
    * Confirm Merge.
    * Delete branch.
11. Always git pull from master branch frequently to get up-to-date copy from master branch. The Git master to branch rebase will ensure that the master and develop branches have all the commits, regardless of where they originated.
    * Git checkout master
    * Git pull
    * Git checkout banchname
    * Git rebase master  
      