**June 11, 2021**

**GIT**

Git is a version control system.

Git helps you keep track of code changes.

Git is used to collaborate on code.

**GIT AND GITHUB**

Git and GitHub are different things.

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

**What Git Does**

* 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

### 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.

**Configure Git**

git configure user.name “user-name”  
git configure user.email “user-email”

These credentials can be used while login

**Adding files**

When you add any file to a git repository it does not add it though git is aware of the file.

Git keeps a track of two files

Tracked and Untracked which can be viewed by using the command

git status gives the status of git repository that lists out commits and list of

tracked and untracked files.

**Git Staging**

One of the core functions of Git is the concepts of the Staging Environment, and the Commit.

git add index.html

The above command adds the file to the repository and it is ready to commit

It appears in the tracked files of the repository.

You can also stage more than one file at a time.

Using --all or -A instead of individual filenames will stage all changes (new, modified, and deleted) files.

Create a file named style.css add and link the file to index.html

**Git Commit**

Adding commits keep track of our progress and changes as we work. Git considers each commit change point or "save point". It is a point in the project you can go back to if you find a bug, or want to make a change.

When we commit, we should ****always**** include a ****message****.

By adding clear messages to each commit, it is easy for yourself (and others) to see what has changed and when.

git commit -m “html and css file added”

Here the -m represents the message.So that it gives a clear understanding of commiting.

Git log gives the status of all the commits.

## Git Commit without Stage

Sometimes, when you make small changes, using the staging environment seems like a waste of time. It is possible to commit changes directly, skipping the staging environment. The -a option will automatically stage every changed, already tracked file.

git commit -a -m “updated index.html file”

* ?? - Untracked files
* A - Files added to stage
* M - Modified files
* D - Deleted files

**Git help**

If you are having trouble remembering commands or options for commands, you can use Git help.

There are a couple of different ways you can use the help command in command line:

* git command -help -  See all the available options for the specific command
* git help --all -  See all possible commands

**Git branch**

Let add some new features to our index.html page.

We are working in our local repository, and we do not want to disturb or possibly wreck the main project.

So we create a new branch:

git branch “hello”

it can be verified by using git branch command

git checkout “branch name” changes the current working branch

Now we have moved to a different branch Let us add some changes to the

Index.html file.

After adding it stage it and commit the changes.

The changes will be succesfully commited in the hello branch

Using -b option while checkout creates a new branch if it is not present

git branch checkout -b “sa”

Now change to master branch and check the changes .You will see the code that is commited in the master branch.

**Merging branch**