GIT – Version Control System

To Save files history

**Download Git**

<https://git-scm.com/downloads>

**Check if installed**

git --version

git version 2.30.1 (Apple Git-130)

**Create account on GitHub**

Go to github.com and create your account. Keep the username and email that you used handy

**On your machine create a folder**

Create any folder on your machine

And go to that folder

**Create simple project**

**Initialize the Git repository**

git init

**Set config**

git config –global user.name <name>

git config –global user.email <email>

git config --global user.name trainings-weblinxsolutions

git config --global user.email info@weblinxsolutions.com

**To edit**

git config --global --edit

**Create a file**

Any file (Eg: first.txt and add any content to the file, a few lines etc)

**Check status**

git status

concept on branch (will tell later)

**Track a file -> Commit**

**Lets now add the file which it shows to add**

git add first.txt

**git status**

Notice the difference in status

**Staging**

**Timeline

Description automatically generated with low confidence**

**Now commit**

git commit -m “Initial Commit”

“-m” – is used to add a message

**To check commits**

git log

Every commit generates a unique hash code

Do Some minor changes

Add all files from staging area

**git add .**

Go back to a previous commit

**git checkout <commit\_id/branch>**

**---------------------**

Go to the previous state

**git checkout master**

**branches**

we always start at master

checkout is used to move around branches

create branch

**git branch dev**

**git branch**

to see all branches

**git checkout dev**

to move to dev branch

To create and checkout a branch

**git checkout -b feature/multiply**

create file and commit

Now if we go back to dev branch, we won’t see these changes

Now if we want the code created in **feature**/**multiply** to be in dev branch, use

**git checkout dev**

**git merge feature/ multiply**

See what git log shows now

Now merge to master

**git checkout master**

**git merge dev**

**git ignore**

If you want some files that git should not track at all

First create a .gitignore file

Then add your file to this file

**git diff**

If we want to see the difference in content of a file to that currently present use git diff.

Syntax : git diff <filename>

**Now lets see what GitHub is**

Create a new repo and follow steps

**git remote**

helps to create,view and delete connections to other repositories

git remote add origin https://github.com/salil-jt/first\_project.git

usually we give name as “origin”

and then use git push to push the code

git push -u origin master

tells what is the origin you have to push and which branch

first time push it will ask your username and password

Generate token using - <https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

Try pushing a few other files

To push other branches, go to that rbanch and push

git checkout dev

git push -u origin dev

-----------------------------------

In a same project multiple people can work, so we can add collaborators

Fork – will create a copy of that repo in your account

Lets create a new repo, and now clone it

git clone <URL>

make some changes and push

but all this is in your forked repo, not the actual one

So, now we will create a PR, click on contribute and create a pull request

Overall Flow

Diagram, schematic

Description automatically generated

To check which remote repo you are connected to

git remote -v

To pull changes from remote to your local on master

git pull