## What is Git?

Git is a widely used modern version control system for tracking changes in computer files.

The term version control system suggests a system that records all the changes made to a file or set of data, so a specific version can be considered whenever needed. This feature makes the process of collaboration so feasible with all team members, making it considerably more comfortable to work over a big project.

Git makes it possible for several people involved in the project to work together and track each other's progress over time. In software development, the tool helps in Source Code Management. Git favors not only programmers but also non-technical users by keeping track of their project files.

While [working on Git,](https://www.simplilearn.com/tutorials/git-tutorial/git-tutorial-for-beginner) we actively use two repositories.

* Local repository: The local repository is present on our computer and consists of all the files and folders. This Repository is used to make changes locally, review history, and commit when offline.
* Remote repository: The remote repository refers to the server repository that may be present anywhere. This repository is used by all the team members to exchange the changes made.

Both repositories have their own set of commands. There are separate Git Commands that work on different types of repositories.

Diagram

Description automatically generated

Diagram

Description automatically generated

Branching

Branch1 Master

Branch2

Branch 3

Merging

**IsGitdifferentfromGitHub?**

**A.** [**Git**](https://en.wikipedia.org/wiki/Git)is a revision control system, a tool to manage your source code history. [**GitHub**](https://en.wikipedia.org/wiki/GitHub) is a hosting service for Git repositories. So they are not the same thing: Git is the tool, GitHub is the service for projects that use Git.

A picture containing text, electronics, screenshot, software

Description automatically generated

## A picture containing timeline Description automatically generated



* As a Lead create new project
* Create local git repo,

Open eclipse , enter repo in search, click git repository. Here you can click on create new repository link or

Right click on project- Team- share project- select checkbox create repo in parent folder- create repository-Finish

* add to index,

Right click on project- Team-Add to index. This will add files to staging area

* commit,

Rgit click- Team- commit – enter a commit – click on commit button on bottom right. This will save it in local repository

* team share-remote-push, login and push.

Before pushing , you need to have github account , create a new repository with same name, copy the url,

Right click on repository- remote push or right click project-team-remote-push

Enter url copied, provide user and password of your github repository, press next, select master branch, click Add spec, click on push. You can check branches in gitrepo. Check for new files in git remote repository.

When it is Unable to Push due to access issues- Create access token and enter the token in place of password while trying to push.

Token generation

Go to your git profile – settings-developer settings- access tokens- create Persona access token- Fine grained token- Allow access to all repositories and select read and write options in all list boxes. Copy the generated token. Now try to push the code again. In login popup instead of password enter this secure token and proceed.

Example:

gittoketwo

github\_pat\_11ANKAZXA0BqO0dMktVm51\_vC4e54d5JGY0t0kAMLeJDJXuG7VxMgc9l6iOBHLAhDkC3GRFJUBCkSp1VjS

* Lead shares link to team members
* Lead needs to add a team member in collaborator

Settings- collaborator- specify team member name- click on Add collaborator- copy link and share it to team member.

* Now login as team member and clone repo

Click on url sent by team lead. Accept invitation.

Go to eclipse-find repository- click onclone repository link- provide url, enter team member github user credentials- master branch- click next- give local path- click finish.

* Import the project to eclipse.

From the git repository, we need to import project

Right click package explorer- click import- select project from git, select existing project repo, finish

* Do some changes- add test2.java
* Now create new branch

Right click on branches – switch to new branch- give name ‘Branchx’

* Git repository- under branches- switch to New branch- give branch name
* Add to index
* Commit with comment to new branch
* Push the branch to remote repository(right click on branch and push branch, select merge option, enter team member credentials)
* Request PR for review code.( login to github and and check the project)- select branch- click on new pull request. Add reviewer. And click on “create pull request” button.
* Lead will review and approve or send comments.( login as lead , click on pull requests and approve or even merge directly by clicking merge pull request)
* When not approved, Add comment or modification by user again and push to remote
* Now login as lead and ask him to merge to master by clicking on comment or merge it by himself
* Now the team member clicks on merge- create a merge commit. Confirm merge
* Check if code is added or not

Push the code github using Intellij idea

* Click on VCS- git
* Authorize it
* Click on git- commit
* Git push

**Git commands used to push code**

git init – will create a local empty repository

Establish connection now

git remote add origin <https://github.com/onlinetraining>

Below 4 commands we have to execute every time to push code

git status

git add -A

First time we need to execute below commands one time

git config –global user.name “ramkrishnaraj”

git config –global user.email “ramkrishnaraj@gmail.com”

git commit -m “write some comment”( it adds all files to local repository)

git push -u origin master( sends all the files from git(local repo) to github(remote repo)

Enter username and password and click ok.

To get latest code

git pull origin master

<https://www.youtube.com/watch?v=cFYNHWIAx3Q&t=535s>

how to do github pull request

<https://www.youtube.com/watch?v=dcCR86PtS-8>

## Git Commands: Working With Local Repositories

### **git init**

* The command git init is used to create an empty Git repository.
* After the git init command is used, a .git folder is created in the directory with some subdirectories. Once the repository is initialized, the process of creating other files begins.

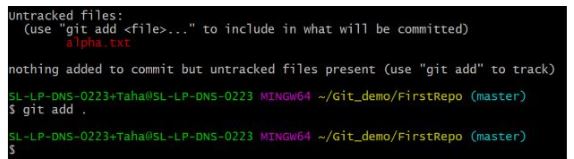
|  |
| --- |
| git init |



### **git add**

* Add command is used after checking the status of the files, to add those files to the staging area.
* Before running the commit command, "git add" is used to add any new or modified files.

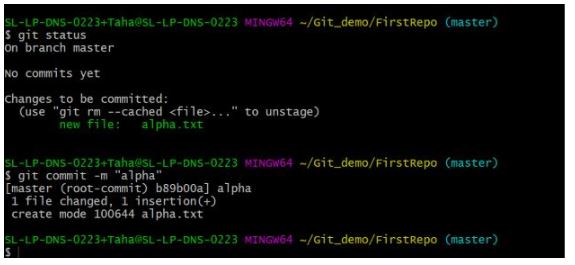
|  |
| --- |
| git add . |



### **git commit**

* The commit command makes sure that the changes are saved to the local repository.
* The command "git commit –m <message>" allows you to describe everyone and help them understand what has happened.

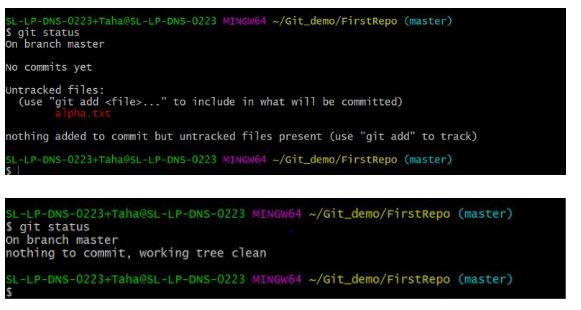
|  |
| --- |
| git commit -m “commit message” |



### **git status**

* The git status command tells the current state of the repository.
* The command provides the current working branch. If the files are in the staging area, but not committed, it will be shown by the git status. Also, if there are no changes, it will show the message no changes to commit, working directory clean.

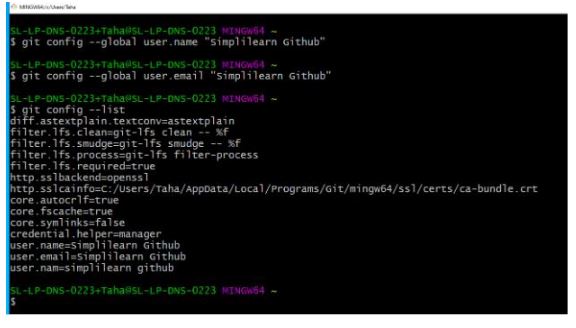
|  |
| --- |
| git status |



### **git config**

* The git config command is used initially to configure the user.name and user.email. This specifies what email id and username will be used from a local repository.
* When git config is used with --global flag, it writes the settings to all repositories on the computer.

|  |
| --- |
| git config --global user.name “any user name”  git config --global user.email <email id> |



### **git branch**

* The git branch command is used to determine what branch the local repository is on.
* The command enables adding and deleting a branch.

|  |
| --- |
| # Create a new branch   git branch <branch\_name> |
| # List all remote or local branches   git branch -a |
| # Delete a branch   git branch -d <branch\_name> |

### **git checkout**

* The git checkout command is used to switch branches, whenever the work is to be started on a different branch.
* The command works on three separate entities: files, commits, and branches.

|  |
| --- |
| # Checkout an existing branch   git checkout <branch\_name> |
| # Checkout and create a new branch with that name   git checkout -b <new\_branch> |

### **git merge**

* The [git merge](https://www.simplilearn.com/tutorials/git-tutorial/merge-conflicts-in-git) command is used to integrate the branches together. The command combines the changes from one branch to another branch.
* It is used to merge the changes in the staging branch to the stable branch.

|  |
| --- |
| git merge <branch\_name> |

popular and basic git commands used by developers.

## Git Commands: Working With Remote Repositories

### **git remote**

* The git remote command is used to create, view, and delete connections to other repositories.
* The connections here are not like direct links into other repositories, but as bookmarks that serve as convenient names to be used as a reference.

|  |
| --- |
| git remote add origin <address> |



### **git clone**

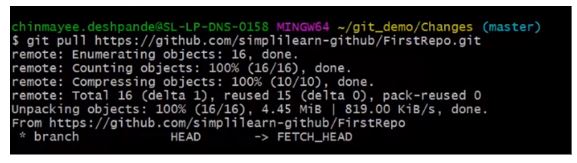
* The git clone command is used to create a local working copy of an existing remote repository.
* The command downloads the remote repository to the computer. It is equivalent to the Git init command when working with a remote repository.

|  |
| --- |
| git clone <remote\_URL> |

### **git pull**

* The [git pull command](https://www.simplilearn.com/tutorials/git-tutorial/git-pull-request) is used to fetch and merge changes from the remote repository to the local repository.
* The command "git pull origin master" copies all the files from the master branch of the remote repository to the local repository.

|  |
| --- |
| git pull <branch\_name> <remote URL> |



### **git push**

* The command [git push](https://www.simplilearn.com/tutorials/git-tutorial/git-push-command) is used to transfer the commits or pushing the content from the local repository to the remote repository.
* The command is used after a local repository has been modified, and the modifications are to be shared with the remote team members.

|  |
| --- |
| git push -u origin master |



## Some Advanced Git Commands

### **git stash**

* The git stash command takes your modified tracked files and saves it on a pile of incomplete changes that you can reapply at any time. To go back to work, you can use the stash pop.
* The git stash command will help a developer switch branches to work on something else without committing to incomplete work.

|  |
| --- |
| # Store current work with untracked files   git stash -u |
| # Bring stashed work back to the working directory   git stash pop |

### **git log**

* The git log command shows the order of the commit history for a repository.
* The command helps in understanding the state of the current branch by showing the commits that lead to this state.

|  |
| --- |
| git log |



Git will have lead account and developer account

Need to have github account.

Create repository in github and get url from that

ghp\_dIAI3hzILymkwGMBds1IXDRwtvEutE0MrbcF

github\_pat\_11ANKAZXA0gXAUv7rnFcPB\_1qBod22lLxgs67lm3BPgHGDCgpXvyLNrZ1QP7MyZLgOW4HRPYWC6MKpLTPR

ghp\_nCoslDV0MoLYhv9Kz5fJ5nxv8CuRo449dP1K

How to work with GitHubDesktop

Continuous Integration with Maven, Jenkins, Git & GitHub part 1

There are 4 different phases to execute

1.Run tests using pom.xml(pom.xml executes testing.xml internally).

Outside eclipse we cannot run testing.xml. so we use pom.xml to execute without eclipse

2.Through Maven Command line interface - This can be used to run tests without eclipse

3.Run.bat

4.Using Jenkins

Two plugins to be added in pom.xml

Maven compiler plugin – for compiling the code

Maven surefire plugin – for running testNG.xml file

Right click maven- update project

Run using pom.xml

Window preferences- java- instead of jre, Add jdk and Select jdk checkbox, if you see error “no compiler” while running using pom.xml

2.Run test cases using command prompt(without eclipse)

Install complete maven software on windows OS

<https://maven.apache.org/download.cgi>

After downloading configure environment variables,

M2\_HOME and MAVEN\_HOME path(bin folder path of maven) to be set for Jenkins to run your tests

%M2\_HOME%\bin and %MAVEN\_HOME% to be set in path for the corresponding

A screenshot of a computer

Description automatically generated with medium confidence

Run Mvn clean install in command prompt in your project directory

3. run test cases using run.bat

Go to project directory- create file- change it to run.bat

Edit-

cd location of project

Mvn clean install

And save it. Just double click on batch file to execute

4. Jenkins installation and configuration

Continuous integration process

QA, DEVOPS, DEV Teams exist in project

Devops will take code from repo and create a build(using maven)

Similarly Devops will take automation code from QA and execute against build(using jenkins) and validate it.

Then message will be sent to all QA members

Team members will get the new build now and continue their process

Devops team is responsible for build automation.

Devops does this process every day.

This is called continuous integration

In Jenkins test executed with headless browser.

In real time Devops will provide you url to QA to access Jenkins

No need to install in real time .

Jenkins.io/download

Longterm support- windows to download

<https://jenkins.io/download/thank-you-downloading-windows-installer-stable>

it will download zip file. Extract it to see msi file

double click on it , click next, click install giving location,click finish.

<http://localhost:8080> is the url to open Jenkins in your local.

need to provide password to unlock jenkins.

Java -jar Jenkins.war –httpPort=9090.

(<http://localhost:9090>, ef2253bdfc8f4877885e29b3479a1874)

Username – admin

Password - ef2253bdfc8f4877885e29b3479a1874

Go to Jenkins installation folder- secrets- initial admin password- copy it and use it to unlock

Create first admin user-

Give admin, admin, admin, email id , click on save and continue

(In real time you can see only your project not all the projects.)

It provides instance url

This is the url to be used to launch jenkins

Now we need to configurations inside Jenkins

Restart your Jenkins. Go to services and restart.

Now login.

Install maven plugins in Jenkins

Manage Jenkins-Manage plugins-maven integration, maven invoker etc select all options related to maven- click on Install without restart

Restart and login to Jenkins

IN two different ways we can run, Freestyle for run.bat

Maven project to run pom.xml

* 1. Create new item and give name

Select freestyle project and continue

Build- execute windows batch command- provide location of project- apply and save.

It is in grey as it is not yet executed.

Click on it and build now

Go to project dashboard- click on build now.

* 1. Using maven project(this is used in general)
  + **Run the test from local pom.xml**

Here we use pom.xml

Dashboard-Manage Jenkins- global tool configuration- Click on JDK

Enter name “ JDK” and enter java\_home path. Example: C:\Program Files\Java\jdk-17.0.4.1

Apply and save

Create new item, select maven project, go to build section- give complete path of pom.xml and enter ‘clean install’ in goals

Apply and save

Now click on Build now

For practice , we are executing Jenkins in local system.

In real time Jenkins is in devops server. Jenkins job will get the code from github repository and execute it.

* + **Run the tests from github**

Login to Jenkins

Create new item maven project

Click on project-configure-source code section

Select git in source code and enter githuburl

Change master to main

Under build- pom.xml and clean install

Manage Jenkins- global tool configuration- enter the path of git bin folder, add bin path in environment variables first – apply and save

A screenshot of a computer

Description automatically generated with medium confidence

* + **Auto build a job in jenkins if there is any change in code on Github repository**

1. Open Jenkins dashboard. Click on manage jenkins

2. Click on Configure system and under github configuration click advanced tab.

3. Check**'**Specify another hook url**'** for GitHub configuration.

4. Now you will get a url in the textbox. Copy this url as it is required in the next steps.

5. Now open your github repository. Go to settings -> webhooks -> add webhooks.

6. Now paste the url from step 4 in the payload url section. Next click on just push the event Now you should be able to see the added webhook in the list of webhooks.

7. Now go to jenkins dashboard. Go to your project configuration. In the build triggers section select github hook trigger for git scm polling. Save the changes.

Elite advantage-

12lakh plus amount deposited

24 for accident

Per annum -1 lakh

13th year – 25lakh 50 thousand tax free

80c-

80d

1010d

ghp\_ftVvFoDRQaHM34hZTAQKpH19hZRkqe1EHBrU

3 additional benefits

7506005025-rakesh

Women special benefit- 150 % to nominee after 12 years

Company bonus- 24286 per annum- 29lakh

Gst 18%-

4.5

2.25

29k gst

You can convert into pension

1161845 – sum assured- life insurance

121200- premium value

12th year

Or 20th year

9.5 % money back

25k – per month

600441

29.72 lakh

Why SSH key?

When working with a GitHub repository, you'll often need to identify yourself to GitHub using your username and password. An SSH key is an alternate way to identify yourself that doesn't require you to enter you username and password every time.

SSH keys come in pairs, a public key that gets shared with services like GitHub, and a private key that is stored only on your computer. If the keys match, you're granted access.

The cryptography behind SSH keys ensures that no one can reverse engineer your private key from the public one.

https://jdblischak.github.io/2014-09-18-chicago/novice/git/05-sshkeys.html

ghp\_ftVvFoDRQaHM34hZTAQKpH19hZRkqe1EHBrU

github\_pat\_11ANKAZXA0BqO0dMktVm51\_vC4e54d5JGY0t0kAMLeJDJXuG7VxMgc9l6iOBHLAhDkC3GRFJUBCkSp1VjS

Continuous integration using git github jenkins

<https://www.youtube.com/watch?v=F8Nfjwneeb4>

<https://www.youtube.com/watch?v=NJPkHtvyAIQ&t=1s>

how to work with git github in eclipse

<https://www.youtube.com/watch?v=HCeBd5GKNO8>

devops tutorial edureka git & Jenkins

<https://www.youtube.com/watch?v=2rd-T_ST9Eg>

<https://www.simplilearn.com/tutorials/git-tutorial/git-commands>

02271190900

supercardservice@rblbank.com