**Introduction:**

* **Git is a popular Version Control System (VCS)**
* **It was created by Linus Torvalds in 2005 and it is maintained by Junio Hamano**
* **Git is used for**

1. **Tracking code changes**
2. **Tracking who made the changes like history of files**
3. **Coding Collaborations**

A person who has requirement

VersionControlSoftware

client

SRS

File1 - - -> 1.0 version

File2 - - -> 1.1 version

File3 - - -> 1.2 version

**w**

Software Requirement Specification

**Keep track of changes made to the file by the developer as per the changes made by the client requirement would be difficult in developer machine.**

**To resolve this problem, we need to use “versioncontrolsystem”.**

Developer Machine

Module-1

=>files

Module-2

=>files

Module-3

=>files

Project

Dev3

Dev2

Dev1

JIRATOOL

What is VersionControlSystem (VCS) and types of VCS?

It is a system that records changes made to the file or set of files over the time, so that we can recall the specific version later.

i.e., for every source code change in a file a new version will be created

e.g.: JDK1.0V, JDK1.1v, JDK1. ]2, ….

Spring1.x, Spring 2.x, Spring3.x

**Types of version control system**

**There are 3 types of version control system**

* **Local version control system**
* **Centralized version control system**
* **Distributed version control system**

1. **Local Version Control System**

**It is used to maintain the file version and retrieve the file based on the specific version**

**Refer: LocalVersionControlSystem.png**

Draw Backs:

* It is easy to forget in which driver you are in and accidentally write the data to the wrong file or copy from other files.
* If the hard disk is corrupted there would a possible loss of secured data.
* By mistake we can delete few files also.

E-drive

File-2

c-drive

file-1

**File**

Developer Machine

F-drive

File-4

D-drive

Fie-3

1. Centralized Version Control System

**CENTRALIZED VERSION CONTROL SYSTEM**

SERVER

commit

update

PC # 3

PC # 2

PC # 1

Working copy

Working copy

Working copy

Repository

* Developers can collaborate the code in one repository and do the change.

e.g. of centralized version softwares: SVN, subversion, perforce…

* Centralized versions server will have single server that contains all the version files.
* For many years this has been the standard version control system
* More no of developers would connect to centralized version computer to checkout the files.

Note:

Checkout -> taking the code from repository to local machine.

Push -> sending the code from local machine to repository (CVS)

Advantage:

* Everyone know to certain degree what everyone else on the project is doing
* Administrator will have full control over which can do what and it is easier to manage.

Disadvantage:

* Single point of failure (SPF) would represent the centralized system.
* If the server goes down due to network traffic, during that hour nobody can collaborate at all or save changes to the server.
* If the hard disk of the centralized system gets corrupted and proper backup haven’t been taken then there is every possibility of loss of data.

Distributed Version Control System: (GitHub)

**Remote Repository**

push

Repository

**Workplace**

**(c, D, E)**

**Local** **Repository**

pull

UPDATE

COMMIT

PC # 3

PC # 2

PC # 1

Working Copy

Working Copy

Working Copy

Repository

Repository

Repository

E.g. Git, Mercurial, Darcs, Baazar, etc.

* Developers will not get the latest version but also the complete history of the files.
* Push will not only happen with latest snapshot of the files rather they will push the old files also.
* If the main server goes off, there is a local repository which would have maintained the copy of the repository where the entire code is available (history of versions).
* If the remote repository is down, then developer can do changes in the local repository and when the main repository is up the code can be pushed to remote repository from local repository.

Note:

In local version control system and centralized version control system getting up the complete history of changes is not possible.

It is possible to get only the latest version, but not the entire history.

Push will not happen w.r.t version rather push will happen only with the latest change.

Version history

File -> 1.0v

File -> 1.1v

File -> 1.2v

File -> 1.3v

Installation of Git software:

1. Open chrome
2. Type git download in the search box
3. Click on downloads git and as per your operating system specification click or select the operating system (windows)
4. Now Downloads for windows page will be open as per your operating system bit download. Download link is given below
5. Download a git software from the following link

<https://git-scm.com/download/win>, click on standalone don’t click on portable select bit 64 or 32b

There are 2 types of Git software

* Git Server

1. It is a repository
2. It is the largest host of source code in the world
3. It is used to store/manage the source code of the project
4. Some of the Git server tools are: Github, BitBucket, GitLab, …..

GitServer (Github, Gitlab)

TCS

Infosys

Payment

(Source Code)

Cards

(Source Code)

Mindtree

payment

cards

loans

Offers

Loans

(Source Code)

Offers

(Source Code)

Wipro

Citibank project

Q> How to connect to github?

url -> <http:///repo.citibank:9999>

url ->

username ->

password ->

git url will be same for all the developers but username and password will different for the developers.

Dev1: <http:///repo.citibank:9999/projects/cards>

Username: Venkatesh

Password: xxxxxxxxx

DEV2: <http:///repo.citibank:9999/projects/cards>

Username: venkyvenkatesh

Passwords: xxxxxxxxxxxx

The developer can’t access other developer projects. Like cards developer can’t access loans project code.

Note:

When we join a company team leader or manager will share the url username, password every developer will connect to gitserver and get source code from the git server and do the changes locally and then move the code from the local repository to the main repository with the versions.

Git server physical location where it is installed can’t be seen. It would be installed on the cloud platform like ASW, AZURE or an any datacenters.

Where should we provide URL, username, password?

To type these details, we need git client.

* Git Client

1. Download a git software from the following link

<https://git-scm.com/download/win>

1. It is a tool which is used to connect to our gitserver.

It we install git client(git s/q) we get the following tools for free.

1. git bash => linux commands are required
2. git gui => Graphical user interface where all the actions will be done through clicks.
3. git cmd => command line tools where developer should provide url, username and password

Note:

Gitclient is a .exe file which can be installed with just few clicks.

Working with gitbash:

TO know the git version, click on git bash and type

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git version

git version 2.45.2.windows.1

To want help then type this

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git help

Git Architecture:

Local repository

Stage Area

Working Area

Remote Server

JdbcUtility.java

Local system/Developer machine

(git client is installed)

**PUSH**

**Commit**

**Add**

Developer will write code and keep it in local drives

(C, D, E)

Github, Bitbucket





.git

Stage area

Local repository

(untracked)

Connection.java

tracked

**pull**

There are 3 regions

1. workplace => It is a place where developers maintain their source code
2. stage area => Once the code is in stage area, we commit it to the local repository with some standard message, from local repository we “push” to main repository by providing URL, username and password.

What is git and github?

Git -> It is a client tool where the user will enter URL, username, password of github repository.

Github -> it is a server where the host of repositories/projects is maintained.

Git commands: (case sensitive)

1. git version
2. git help
3. git config
4. git init
5. git clone
6. git add
7. git status
8. git rm
9. git restore
10. git commit
11. git log
12. git push
13. git pull
14. git branch
15. git checkout
16. git stash

1)git version

This command is used to check the version of git

Admin@DESKTOP-9BCSDMB MINGW64 ~

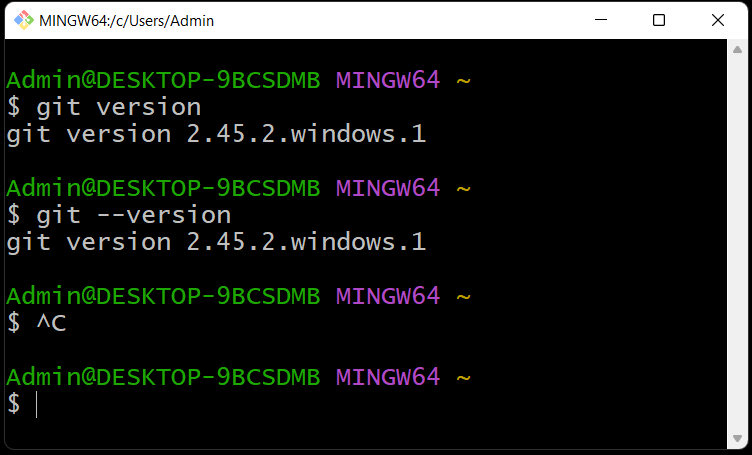
$ git version

git version 2.45.2.windows.1

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git –version

git version 2.45.2.windows.1



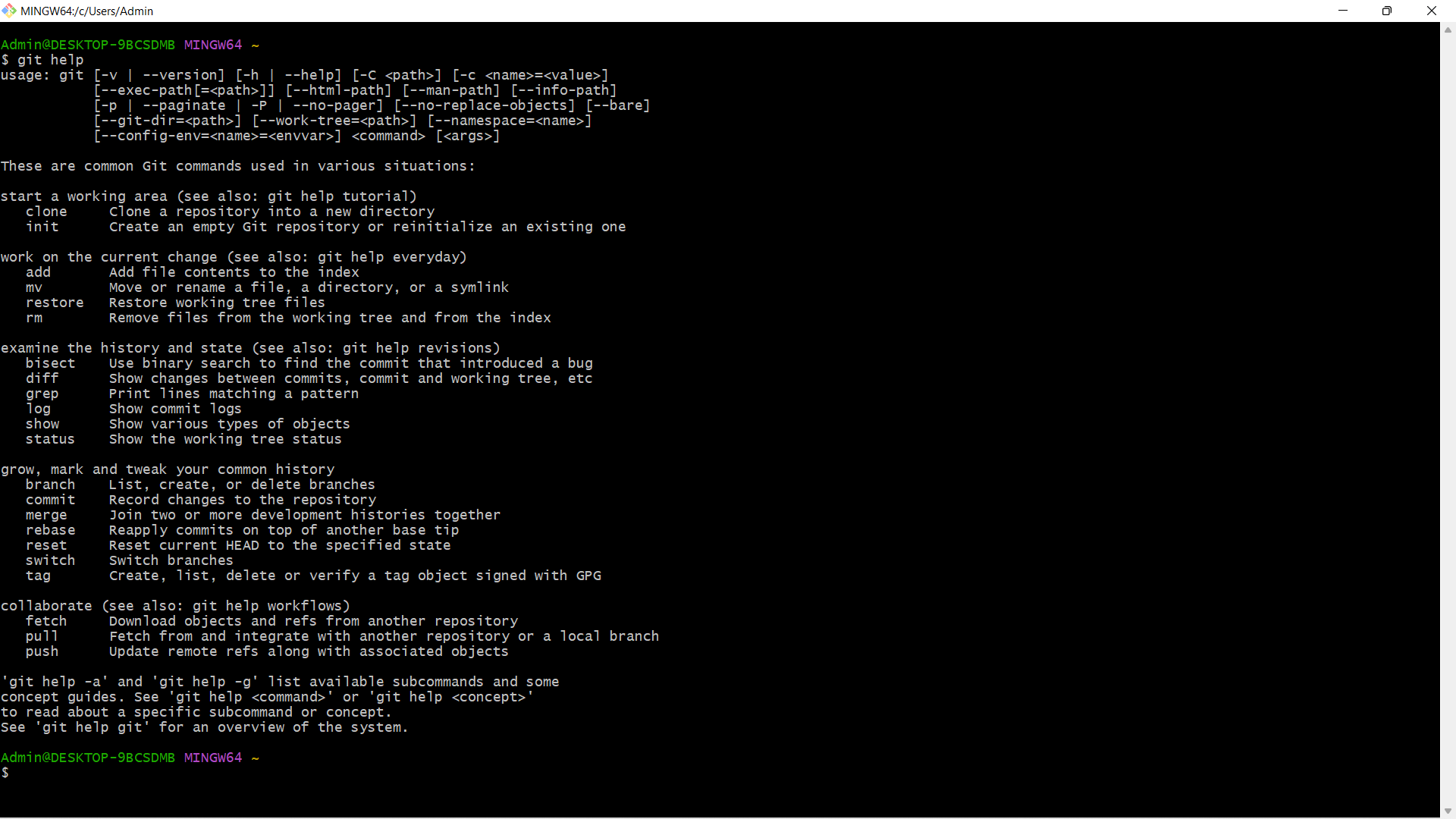
2)git help:

If we want to see the list of commands then we can use git help

Syntax: git help

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git help



Note: This command is useful to get the documentation of any command

e.g. git help <command-name>

How to create an git hub account?

1. Open chrome type github.com
2. Click on click on sign up
3. Enter you email – id
4. Click on ok
5. Now type password
6. Click on continue
7. Enter username
8. Click on continue
9. It will ask you to receive product updates and announcements via email? It will tell to types yes or no type no
10. Click on continue
11. It will show you a puzzle
12. Solve the puzzle given by github to verify your account
13. Then click on create account

3)git config:

It is used when the git software is used for the first time.

The command will set the developer identity like name, emailed,…

This configuration information will be used by git software for every push operation encountered.

>git config - -list // this command is used to provide the list of configuration if you don’t have credentials then create by the commands.

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git config –list

Output:

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

http.sslbackend=openssl

http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt

core.autocrlf=true

core.fscache=true

core.symlinks=false

pull.rebase=false

credential.helper=manager

credential.https://dev.azure.com.usehttppath=true

init.defaultbranch=master

//to set the username and email

>git config - -global user.name “Venkatesh”

>git config - -global user.email “vanvenkatesh988@gmail.com”

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git config --global user.name "venkatesh"

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git config --global user.email "vanvenkatesh7@gmail.com"

Output:

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ git config --list

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

http.sslbackend=openssl

http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt

core.autocrlf=true

core.fscache=true

core.symlinks=false

pull.rebase=false

credential.helper=manager

credential.https://dev.azure.com.usehttppath=true

init.defaultbranch=master

user.name=venkatesh

[user.email=vanvenkatesh7@gmail.com](mailto:user.email=vanvenkatesh7@gmail.com)

global => It indicates the user can work with git commands from different drives of computer.

Note: git config - -list - -show-origin //display the location of git configuration holded by git software.

Important operations associated with git

git init

* Normally a folder will be created in the developers works place and inside the folder the source code would be place.
* Normally this is the first command which we execute to setup the git for operations like clone, push, pull, …
* This command internally created one folder called .git
* .git is used by git software to identify the folder which should participate in pushing to “local” and “remote” repositories.
* Syntax: git init
* To change the directory, use the following command

Admin@DESKTOP-9BCSDMB MINGW64 ~

$ cd E:

Admin@DESKTOP-9BCSDMB MINGW64 /e

$ cd gitsession/

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession

$ cd workspace-1/

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1

$ pwd

/e/gitsession/workspace-1

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1

$ git init

Initialized empty Git repository in E:/gitsession/workspace-1/.git/

To change the directory use the command cd…

To know the present working directory use command pwd

To know the status of working directory use this command

Command: git status

* This command is used to check the status of the working directory
* Git status
* Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1 (master)
* JdbcUtiility.java

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1 (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

JdbcUtility.java

nothing added to commit but untracked files present (use "git add" to track)

Git status normally will give outputs in the following ways

1. Untracked files (red color)=> it means the files are present still in woking area and these files can’t be committed to “local repository” no to “remote repository”
2. Tracked files(green color) => it means the files are moved from working area to stage area so these files can be committed to “local repository” and to “remote repository”.
3. Modified files(red color): it means the files are present still in working area and these files can be staged or it can also be restored back to the normal phase.
4. To add the file to stage area type the following command git add <filename>

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1 (master)

$ git add JdbcUtility.java

1. Then check the status by this command git status

Admin@DESKTOP-9BCSDMB MINGW64 /e/gitsession/workspace-1 (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage new file: JdbcUtility.java

Eg: git rm –cached <filename>

If we want to push all the files from workspace to stage area, we use the following command

Syntax: git add .

git add –a

It is also possible to unstage the files from stagged area to workspace, using the following command

Syntax: git rm - -cached <file-name>

To restore the old file we use the following command

Syntax: git restore <file-name>

The files which are ready for commit be in stage area, to perform commit operation we use the following command

Syntax: git commit -m <some message>

Eg#1: git commit -m “first commit”// This file commit all the files present in stage area

Eg#2: git commit -m “second commit” filename //this will commit only that file into local repository.

Steps followed to create a remote repository and push it to remote repository:

1. Open github.com by providing the credentials
2. Create a new repository and enter some name (Establish-connection) and click on create repository.
3. To perform push operation we need to use the following command

git branch -M main

git remote add origin https://github.com/VENKATESH76668/Establish-connection.git

git push -u origin main

Dev-1

Configuratio-n setup

Init

Add - -a

Commit -m “message”

push

Copy of project

clone

Gitclient

(workspace-1)

Dev-2

Configuration setup

Pull

(latest change code will be pulled)

Copy of project

clone

push

Gitclient

(workspace-2)

url, username,password

Github project

Difference between pull and clone

Git pull -> it is used to fetch the latest changes made in remote repository to working directory.

Syntax: git pull

Git clone -> it is used to clone the repository to the working directory of the developer

Syntax: git clone <url>