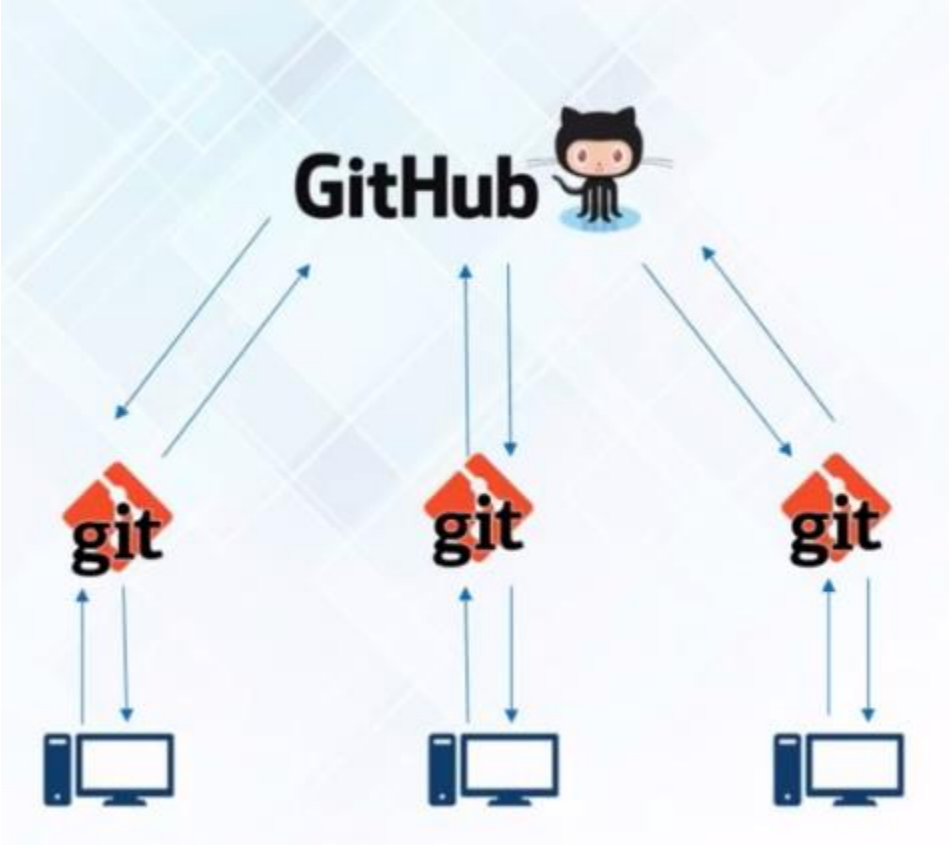
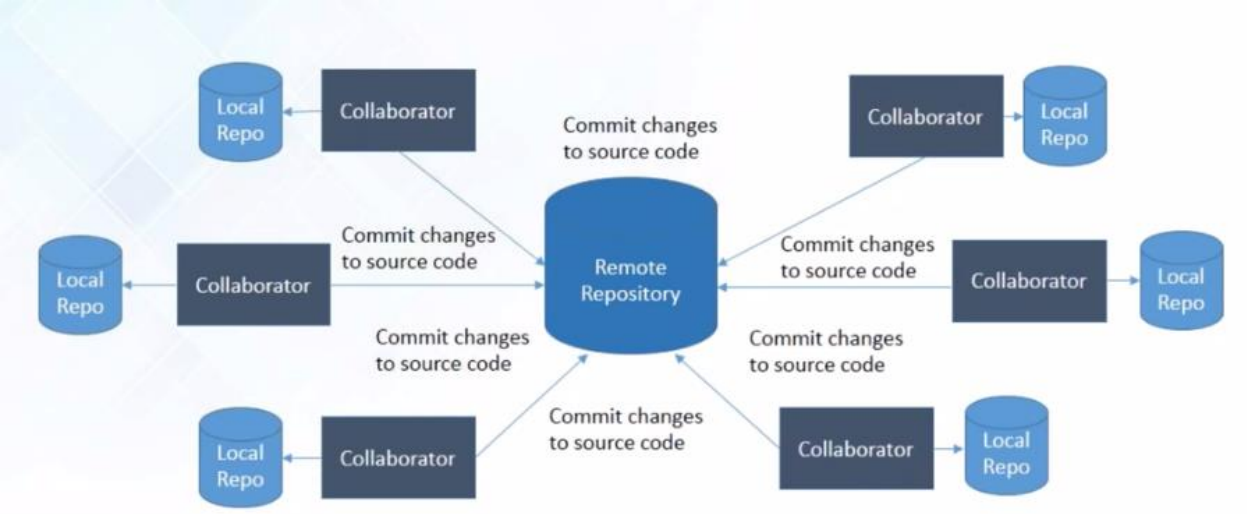
GITHUB and GIT fundamentals



GitHub is a central repository and Git is a version control tool installed on separate workstations to provide developers the freedom to create local repositories. Once the code is checked in on local repositories, it can be pushed to central repository. Also through Git we can pull the code from central repository. It is based on distributed version control system.



Git is compatible with SVN repository i.e. if we don't need to migrate the complete code base from SVN server to GIT server, simply install Git tool on each workstation and using Git commands we can communicate to SVN server and perform actions (pull and push) of code.

Git should be installed on Ubuntu / windows and we can use commands to perform GIT operations.

GIT- Commands:

Install Git on your Unix distribution:

****

sudo apt-get update # this will update apt (UNIX package manager)

sudo apt-get install git (this will install git)

To know git version:

****

git --version

To turn a directory into a local repository

****

git init

The files which needs to be commit in the repository should be moved to staging area and to do so below is the command

****

git add test.py

to add all the files in the staging area

****

git add .

to commit the files in local repository which are sent to staging

****

git commit -m "commit message"

To know the status of your file and current state i.e.

>> If files are in staging but not committed

>> The files which are committed

>> What is the current branch on which you are working

****

git status

to config the user and email with the git

****

git config --global user.name "sbxtreme"

git config --global user.email "sbxtreme13@gmail.com"

To list all the branches of your local repository

****

git branch

To create a new branch

****

git branch sec_branch

To move to a branch

****

git checkout sec_branch

To merge secondary branch to master branch

****

git checkout master

git merge sec_branch

Once all the changes are merged in master branch we can delete the secondary branch

To delete secondary branch

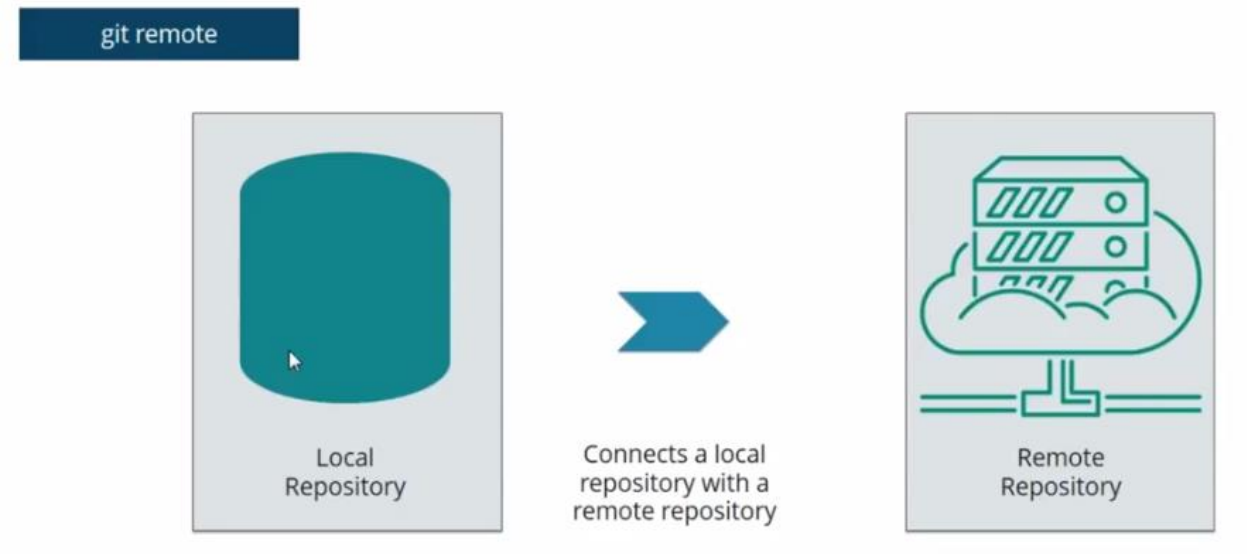
****

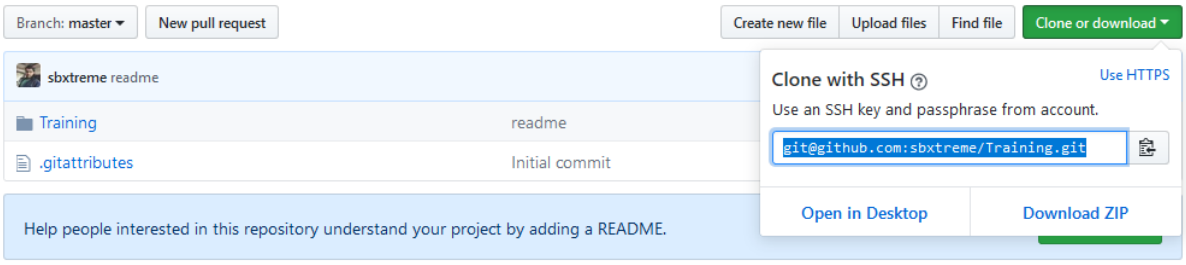
git branch -d sec_branch

it is safe as it checks for any unmerge files and deletes. In case of any unmerge file it will throw error.

git branch -D sec_branch

but this is not safe as it deletes the secondary branch without any unmerged file.





To connect to GitHub remote repository:

****

git remote add origin <SSH key to passpharse account>

You'll get SSH key from your github account

git remote add origin git@github.com:sbxtreme/Training.git

To remove the remote

****

git remote -v

git remote rm <name of remote> # get this name by running the above command

To clone a remote repository to our local system:

****

The command downloads the files and folders present in a repository from remote server (GITHUB) to our local system.

git clone <SSH key to passpharse account> # You'll get SSH key from your github account

git clone git@github.com:sbxtreme/Training.git

In case of any permission issue pls refer the below article to generate SSH key from your system and adding it to your GitHub profile.

https://stackoverflow.com/questions/2643502/git-permission-denied-publickey

To pull the changes from GITHUB onto your local repository:

****

git pull origin master

To push the changes from your local repository to Remote repository:

****

Before pushing the changes we need to add in staging area and commit the changes.

add testfile.txt # adding in staging area

git commit -m "Changes from Ubantu by sbxtreme" #commit the changes

git push origin master

push the committed changes from local repository to Remote repository

To know the complete commit history of a repository:

****

git log

git log --before="31-DEC-2017" # want to know the commits before 31st dec 2017

git log --author="sbxtreme" # want to know the commits done by author sbxtreme

git log --after="31-JAN-2018" # want to know the commits done after 31st jan 2018

git log --oneline

this gives the commit hash(number) related to all the commits happened on repository

To go back to previous commit or revert the changes:

****

git revert <commit hash>

git revert 9d5b4cb