**Git commands:**

Install Git:

yum update -y

Yum install git -y

git --version

Create Git project

mkdir helloworld && cd helloworld

git init /#Initialize empty Git repository#/

echo “Hello World…”>hello.txt

Check status:

git status

/# it will show tracked and un-tracked files. Byt looking at the status you can add required files and commit those files and push to repo#/

Add files to staging area:

git add hello.txt

or

git add . or -A /# add all un-tracked files to staging area #/

git add -U

Commit changes:

git commit -m “initial commit”

Add github or gitlab url:

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

git push -u origin master

/# it will ask for userid and password.#

**note:** Check .git/config file it will show git repo url

What is the use of .gitignore file?

Vi .gitignore

Target/

\*.exe

Push it to repo.

git wont allows the files those are mentioned in the .gitignore file to add them to staging area.

Create branch and do changes and merge to master and push the code.

git branch /\*

/# it will show the available branches. \* indicates we are in that branch #/

git branch div /# to create branch div #/

git checkout div /# to switch to branch div #/

Or

Git checkout -b div /# create branch and checkout #/

**Note:** we can do what ever work we need and we can push it to repo. First time mention git push origin div because git dontknow where to push first time, later you can use git push once you checkout the branch.

Merge branch code to master:

git checkout master

git merge div

git push

How you avoid Merge conflict:

In master add a1 and a2 txt files, commit those files and push it to repo.

git checkout div

create a1.txt file add data and commit it.

Now merge the master code to div branch you will see conflict issue. Because working on same file in multiple branches and multiple commits caused this issue.

How to fix it?

vi a1.txt in div branch

as per developer edit the file and commit it.

Try git merge master it will say up to date.

Try merge in master branch using “git merge div” it will work.

Push the code

git log /# to show all the commit histories in repo.#/

git stash

/# To save the local changes in queue. And move head to last commit. #/

git stash list

/# it will show the list like stash@{0}, stash@{1}… #/

git stash apply --index stash@{0}

/# it will bring back the saved content to local only.#/

git stash pop --index stash@{0}

/# it will bring back the saved content to local and delete from stash list queue.#/

If you didn’t pass index it will pull resent stash(last in first out.)

**Git diff:**

git diff a1.txt

/# it will compare local file with indexed file#/

git diff <commit1> <commit2>

git diff --cached a1.txt

/# it will check added(staged) file with last commit #/

**Show file in particular commit:**

git show <commit\_id>:<filename>

**Tag:**

Just for marking important commits.

git tag /# to show available tags#/

git tag v1 /# to create lightweight tag noting but justing pointing to latest commit id #/

git show v1 /# just some info about commit and author…#/

git tag v2 <commit id> /# creating tag for specific commit. #/

git tag -a v3 -m "created annotated tag"

git push origin --tags /# to push all tags at once#/

git push origin tag v3 /# to push v3 tag#/

Git tag -d v2 /# delete tag in local system#/

Git push origin :refs/tags/v2 /# delete from central repo #/