Short Notes On GIT Documented By Praveen

Introduction to git

- It is a open source
- Distribution version control system
- Installation
- Linux/unix
- The dependable packages
  1. Yum install open-devel
  2. Yum install curl-devel

1. Git installation

Download git

http://distfile.macports.org/git-core/git-x.x.x.tar.gz

cd git-x.x.x

make prefix=/u01 all

make prefix=/u01 install

2. Creating bare repository

go to directory where the repository should be created

mkdir git-repo

cd git –repo

git init –bare(by using this, the repository as a remote repository)

note: the avoe commands creates a set of files and meta data to store the user data.

3. Git-client machine

Client machine (developer machine)

→ssh access to git repository frem developer machine.

→windows client

- Install the git-bash client software

→unix client

- Install the git software which is installed in the server machine.

4. Gi-cloning repository

→ Cloning repository in the client machine whether that is win/unix same process.

- Create a directory(mkdir git-new)
- Git clone url
- Ls
- Cd git-new
- Ls –la (which list all the repo meta data)
- Git settings:
  →git config –list(list all the attributes of git configuration)
  →git config –global user.name Praveen

5. Git adding the files

- Process to add the files to remote repository:
  →create the file
  →stage the file
  →commit the file
  →push to remote repository
- Commands:
  →touch hello.txt
  →git status(it gives working tree status)
  →git add hello.txt(it stage the file)
  →git status
  →git commit -m hello.txt "this is my first commit"
  →git push origin master

6. Modified file

- Modified the file commit the changes:
  →vi hello.txt
  →git add hello.txt
  →git commit –m "comment"
  →git push

- From second machine:
  →git pull
  →cat hello.txt
  →git log

9. Merging the conflicts

- Git mergetool
- Enter
- Opennups merge window->select respective changes then save and quit.
- Git add
- Git commit
- Git branch (make sure your are in master branch)
- Git push →push the merged changes

10. Git commands:

- Git config –list(all env variable of git)
- Git config –global user.name Praveen(set the user name as praveen)
- Git config –global user.email Praveen@gmail.com(set the user email)
- Git config –global push.defaul simple
- Git clone
- Git add
- Git commit –m ""

- Git commit –a –m “”(staging and commit)
- Git push(send the commit to remote repo)
- Git status(status of working tree)
- Git branch(on which branch you are in)
- Git pull(get the changes from remote repo)
- Git mergetool(this will open your merge tool)

  → git config --global merge.tool p4merge(mergetool setup first command)

  → REMOTE\" \"$MERGED\"'mergetool.p4merge.cmd 'p4merge.exe \"$BASE\" \"$LOCAL\" \"$(mergetool setup second command)

- Git rm(delete the file in a working tree and to reflect the same in server remote repo you need to commit and push)
- Git log (history of your repo)
- Git log file name
- Git log –author=name
- Git shortlog
- Git log –online
- Git checkout (change set)
- Git diff (to check the difference between staging area and git)
- Git show (show changes of patcset)
- Git clean –n
- Git clean –f
- Git blame file1(information about each line modification done by author)
- Git fetch/pull (track remote branching into local working tree)
- Git pull(performs a git fetch and git merge)
  note: git fetch does not perform any operations on your local branches

11. Git-directory structure

- **Master →this is root directory and default one**
- **Branches → it contains the major releases**
- **Tags → it contains minor releases and always read only**

12. Git branching

- Default branch is master
- How do we know branch name?
  >git branch
- Create a new branch from master?
  >git branch kpk (branch will be created from master ->kpk)
  >git checkout kpk (to switch to kpk branch)
- Create branch and switch it immediately
  >git checkout –b kpk

- List all the branches including remote branch
  >git branch –a
- Display info of your branches
  >git branch –v
- Display remote branches
  >git branch -r
- Renaming a branch
  >git branch –m kpk Praveen

- Perform some changes on branch and push it with below command
  >git push origin praveen
- Delete branch in local repo
  >git branch –d Praveen
- Delete same in remote repo
  >git push origin :refs/heads/praveen
- Difference between brances
  >git diff kpk praveen

13. Parallel branching strategy

- Objectives:
  1. No data loss in production
  2. Branch is the dev code where check in and checkout
- Before deploying:

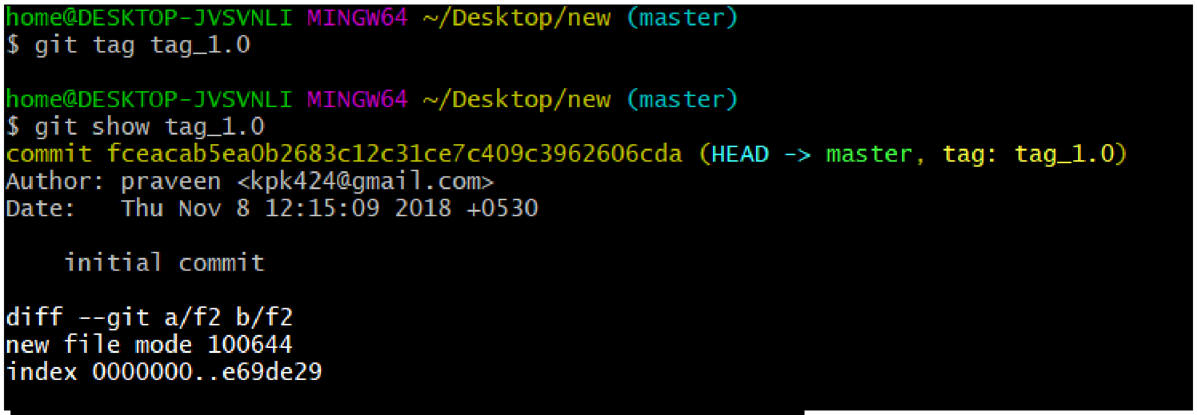
1. No data loss/overwriting code in production environment

- After deploying:

2. Make your master = prod code base(release branch)

14. Git tags

- In git tags are read only
- Snapshot of particular revision number having a name and meant for read only
- Create a lightweight tag
  >git tag tag_1.0
- Show git tag version
  >git show tag_1.0
- List all tags
  >git tag

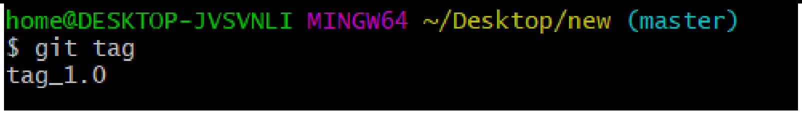


```
home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git tag tag_1.0

home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git show tag_1.0
commit fceacab5ea0b2683c12c31ce7c409c3962606cda (HEAD -> master, tag: tag_1.0)
Author: praveen <kpk424@gmail.com>
Date:   Thu Nov 8 12:15:09 2018 +0530

    initial commit

diff --git a/f2 b/f2
new file mode 100644
index 0000000..e69de29
```



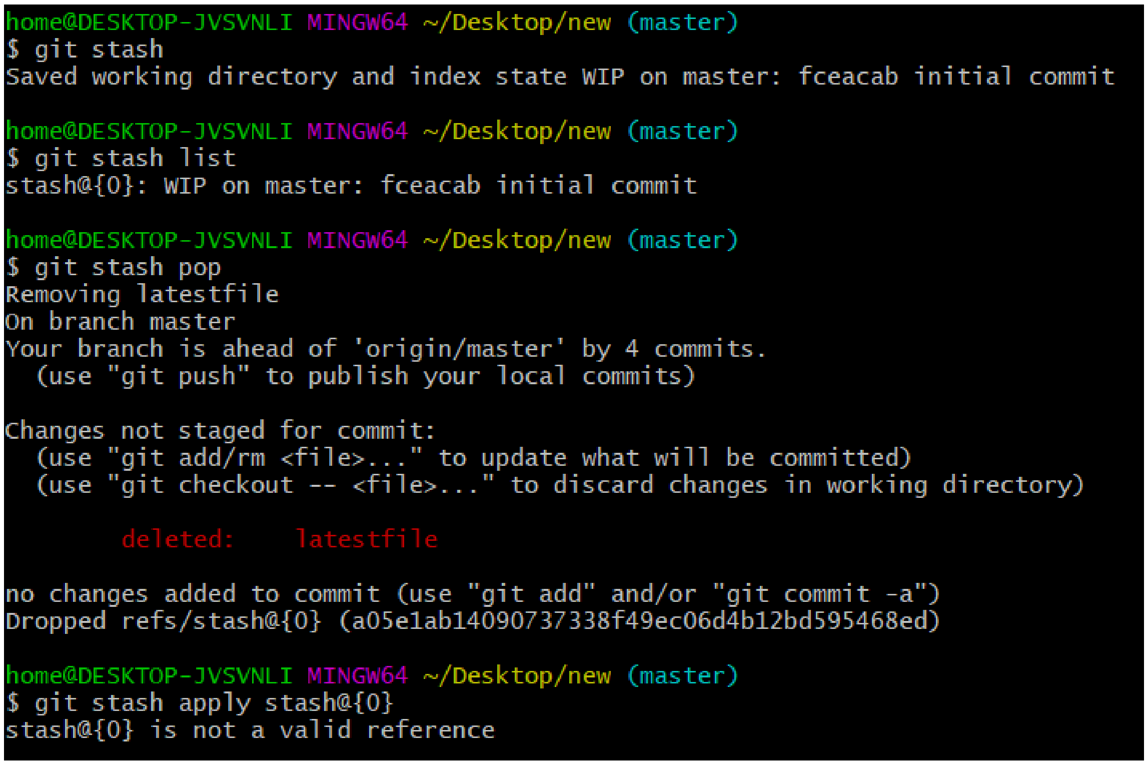
```
home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git tag
tag_1.0
```

- Create an annotated tag with some messages
  >git tag tag_1.1 –m “release 1.0 completed”
- Checkout tag
  >Git checkout tag_1.0
- Push the tag
  >git push origin tag_1.1
- Delete tag
  >git tag –d tag 1.0
- Delete same in remote repo
  >git push origin :refs/tags/tag_1.0

15. Git stashing

Stash stores the unstaged(git add) changes into a packages and stores in memory, when ever you want to apply the unchanged changes you can do this.

- Git stash
- List all stashes
  >git stash list
- Apply the stash when you wanted with below command
  >git stash pop
- Or you can use the id as well as shown below
  >git stash apply stash@{0}
- This may lead to conflicts, resolve conflicts manually
- Once the conflicts resolved commit the changes
- Delete the stash stash packet
  >git stash drop stash@{0}



```
home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git stash
Saved working directory and index state WIP on master: fceacab initial commit

home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git stash list
stash@{0}: WIP on master: fceacab initial commit

home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git stash pop
Removing latestfile
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    latestfile

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (a05e1ab14090737338f49ec06d4b12bd595468ed)

home@DESKTOP-JVSVNLI MINGW64 ~/Desktop/new (master)
$ git stash apply stash@{0}
stash@{0} is not a valid reference
```

16. Git remote repositories

Default will be origin but we can add multiple repositories as well

1) Git remote add <reponame> username@github.com/project/repo
2) Git remote show origin(which repository it has been pointed)
3) Git remote -v