Date 07/05/20

Reference: https://guides.github.com/activities/hello-world/

Reference: blog.ajinda.com/version-control-software/git-and-github/

Git

Git is a source code management software/system. Other source code management softwares in the market are TFS (Team Foundation Server) provided by Microsoft & SVN (Apache Subversion) provided by Apache.

In other words, we can say, git is a version control system (VCS) for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for software development, but it can be used to keep track of changes in any files.

Source code management softwares have many features, some of them are as followings:

1. Track files & their changes
2. Codes comparison (Compare codes)
3. Source code versioning
4. Source code branching
5. Merging branches
6. Compare branches

etc.

Git software is today available for almost all platforms. We can install git on windows & run git commands through command line i.e. CMD. When we install git, this software also installs their own command line tool called 'Git Bash', we can

Date 07/05/2017

collaboration - कलेबरेशन - act of working jointly
Repository - रिपाजिटोरी - गोदाम
Revision - रिविशन - संशोधन, सुधार/दोहराई
Remote - रिमोट - a device that can be used to control machines from a distance.

use it too to run git commands.

GitHub: GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.
Or we may say that, GitHub is a web based git repository hosting service.
When we install git on our machines then we get all features of repository locally but our system(machine) is storage is not centralized, due to this reason multiple developers can not do collaborative development on your system. To solve this problem, we need a remote git repository like GitHub. In the use of git, repository is always present in every developer's system locally as well as on a centralized place like/for ex GitHub where we have to sync/push all changes of local repository to there so that all local changes can be update on centralized storage like/for ex on GitHub.

1 repository = 1 project with their all source-code files

i.e. 1 repository provides space exactly for 1 project/application

Date 14/05/2017

Git Commands

DELTA Pg No. Date / /

Installing Git extension on Windows:

1. Search for 'git extensions' on google
2. Find website https://gitextensions.github.io/ and then open this.
3. Click on link at first line of page i.e. Download latest version: Download Link
4. Click to download 'GitExtensions-*-setupComplete.m
5. Now install this extension & restart computer &
6. After installation open cmd and GitBash (desktop shortcut maybe)

Configure local system: To configure local system so that your system (machine) can be identified in git repository system when multiple developers work, we do the followings: Use Git command line tool 'Git Bash':

```
git config --global user.name "jitendra" ↵
git config --global user.email "goto.jiterdra@gmail.com" ↵
```

To get configured username & email, use following commands:

```
git config user.name ↵
git config user.email ↵
git config --list ↵   ⇒ This lists you all locally configured git key/value pairs including username and email.
```

Date
14/05/2019

DELTA Pg No.
Date / /

Create repository: Use Git Bash and go to the directory where you want to create/initialize repository/application/project or already created project.

git init magento2 ↵

hidden folder

You will see, there creates a directory (i.e. repository) with name 'magento2'. Directory magento2 contains another directory '.git' inside it having many files & folders. Actually .git contains git repository configuration files for the current repository magento2. magento2 directory may also contain a file with name '.gitignore'. This files describes whose files & folders would not be treated as part of your git repository versioning system.

Example: content of .gitignore may be as:

/nbproject/ (a folder in magento2)
/.gitignore (.gitignore file itself)

Download (clone) remote repository (from GitHub):

Get remote repository path first which you want to download. For me, I have created an account on github.com so all my repositories can be seen on url:

https://github.com/{username}/rakyadav

One of my remote git repositories url is

https://github.com/gitendrakyadav/hello-world

Date
14/05/2017

DELTA Pg No.
Date / /

In the Git terminology 'download a repository' is called 'clone a repository'.
Go to the directory where you want to download the remote repository, right-click in the directory & select 'Git Bash Here' to open Git command line tool:
git clone https://github.com/jitendrakyadav/hello-world ↵
We will see that a directory/repository 'hello-world' has been created/downloaded/cloned.

Change a file in git repository & track operations on it: Any file which we create in version-control, may be in one of following states:
1. Un-tracked → A file with some changes before 'git add command' remains in untracked state.
2. Staged → After using 'git add command', file enters into staged state. It means, now file is available in versioning preparation list and ready to move into tracked list.
3. Tracked → After using 'git commit -m "_"', file enters into tracked state. It means file & all its changes are now under version control system and git repository/git system is monitoring this file and any no. of changes making on this

Date
14/05/2017

DELTA Pg No.
Date / /

file can be seen/tracked in future.

LF will be replaced by CRLF in git - (warning) - What is that and is it important?

Answer: In Unix system the end of a line is represented with a line feed (LF). In windows a line is represented with a carriage return (CR) and a line feed (LF) thus (CRLF). When you get code from git that was uploaded from a unix system they will only have an LF.

If you want to turn this warning off, type this in the git command line:

git config --global core.autocrlf true

If you are on a Linux or Mac system that uses LF line endings, then:

git config --global core.autocrlf input

If you are a windows programmer doing a windows-only project, then:

git config --global core.autocrlf false

Basic commands:

ls or dir ⇒ This is a generic command used to list all the files and folders

status ⇒ This command will tell you what branch you are on, what files have changed, or lists

Date 15/05/2015

DELTA Pg No. Date / /

all the new or modified files to be committed and also hints on what to do next.

add ⇒ It moves file from untracked state to staged state.
Example: git add xyz.php ↵
git add . ↵ (It moves all files available into untracked state to staged state)

commit ⇒ It moves files from staged state to tracked state.
Ex: git commit -m "initial commit" ↵

checkout -- filename ⇒ It discards changes of file i.e. throws the file to last committed state
Ex: git checkout -- xyz.php ↵
git checkout . ↵ (It discards changes of all files)

log ⇒ It lists history commits.
When log history is long you need to press shift+z twice to come out of list

log --oneline ⇒ It also lists history commits but the output will be shorten for you.

diff ⇒ It shows differences between two copies of same file; 1st which is in versioning system/tracked list and 2nd whose modifications are not committed yet.

diff --cached ⇒ In previous command, 2nd copy was in untracked state, but here, we use it when 2nd copy is in staged state.

Date
15/05/2017

DELTA Pg No.
Date / /

Ex: git diff --cached xyz.php ↵

reset ⇒ It removes file(s) from staged state and sends to untracked state but preserves file(s) changes.

Ex: git reset HEAD xyz.php ↵

git reset HEAD . ↵

rm ⇒ When you want to remove a file / permanent delete from git repository system, this command helps you.

or

It removes file from the working tree and from the index.

Ex: git rm index.php ↵ (It will remove/delete the file index.php from your directory/repository)

git commit -m "delete/remove index.php" ↵

↑ and it will update your tracked list.

Now the process is complete.

Date 24/05/201

Configure local git repository to connect remote git repository

DELTA Pg No. Date / /

or

Adding an existing project to GitHub using the command line:

Putting your existing work on GitHub can let you share and collaborate in lots of great ways.

1. Create a new repository on GitHub.
   Goto GitHub.com → Click on [+▽] sign in header section → New repository
   To avoid errors, do not initialize the new repository with README, license, or gitignore files. You can add these files after your project has been pushed to GitHub.
2. Open Git Bash
3. Change the current working directory to your local project root directory. For Ex: D:\xampp\htdocs\2017\magento2 (i.e. inside magento2 directory)
4. Initialize the local directory as a Git repository:
   git init ↵
   This command will create a directory '.git' in your project root directory, having git repository configuration files for the current repository/project.
5. Add the files in your new local repository. This stages them for the first commit.
   git add . ↵
   Add the files in the local repository and stages

Date -24/05/2017

DELTA Pg No. Date / /

them for commit. To unstage a file, use 'git reset HEAD YOUR-FILE'.

6. Commit the files that you have staged in your local repository.
   git commit -m "first commit" ↵
   Commits the tracked changes and prepares them to be pushed to a remote repository. To remove this commit & modify the file, use 'git reset --soft HEAD~1' and commit and add the file again.
7. At the top of your newly created GitHub repository's (i.e. if newly created remote repository name is 'magento2' then https://github.com/jitendrakyadav/magento2) Quick Setup page, click to copy the remote repository URL.
8. In the command prompt, add the URL for the remote repository where your local repository will be pushed:
   git remote add origin REMOTE-REPOSITORY-URL ↵
   // Sets the new remote
   For Ex: git remote add origin https://github.com/jitendrakyadav/magento2.git
9. Push the changes in your local repository to GitHub.
   git push -u origin master ↵
   It pushes the changes from your local repository to the remote repository you specified as the origin.
   This command verifies your GitHub username & password.

Date 25/06/2017

Precisely - प्रिसाइस्ली - ठीक ठीक, निश्चित रूप से, शुद्ध रूप से

DELTA Pg No. Date / /

mv ⇒ It is used when you want to move or rename a file, directory, or a symlink.
Ex: git mv index.php index_modified.php
//i.e. ~~rec~~ rename a file index.php

push ⇒ It updates remote refs along with associated objects i.e. when we inside a branch & execute command "git push", it updates the same branch on remote server (github.com).

pull ⇒ Fetch from & integrate with another repository or a local branch. In it's default mode, "git pull" is shorthand for "git fetch" followed by "git merge FETCH_HEAD". More precisely, "git pull" runs "git fetch" with the given parameters and calls "git merge" to merge the retrieved branch head into the current branch. With --rebase, it runs "git rebase" instead of "git merge".

fetch ⇒ "git fetch" downloads objects and refs (branches and/or tags) i.e. commits from remote repository. But It does not show affects on local even "git log" can not show/track these remote commits on local after above command execution. To make visible these downloads, you will have to merge these downloaded branches into your local branch. For Ex: Through "git fetch" there downloads a remote commit/branch "remote/master", so go

Date 25/06/2017

DELTA Pg No. Date / /

to inside your local "master" branch & execute the command "git merge origin/master". Now your downloaded remote master branch has been merged into local master branch. Now you can see remote master branch commit effect into your local master branch & can view the remote master branch commit held on remote server with date/time in your "git log".

branch ⇒ List, create or delete branches.

Ex: git branch ↵ //will list all branches in current repository

git branch 123_form_create ↵ //will create a branch named "123_form_create" in current repository by duplicating the files/folder of branch in which you are currently & execution moves/enters into newly created branch i.e. 123_form_create

git branch -d test1 ↵ //To delete a local branch "test1"

//when you delete a branch you must be in another branch, not in the same branch

git push origin --delete test2 ↵ //delete a branch from remote repository

//Delete a branch from local & remote are 2 different things, they do not affect to each other

Note: The "-d" option is an alias for "--delete", which only deletes the branch if it has already been fully merged in its upstream branch. You could also use "-D", which is an alias for "--delete --force", which deletes the branch irrespective of its merged status.

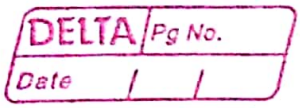
checkout ⇒ Switch branches from one to other or restore (throws the file to last commited state) working tree files.

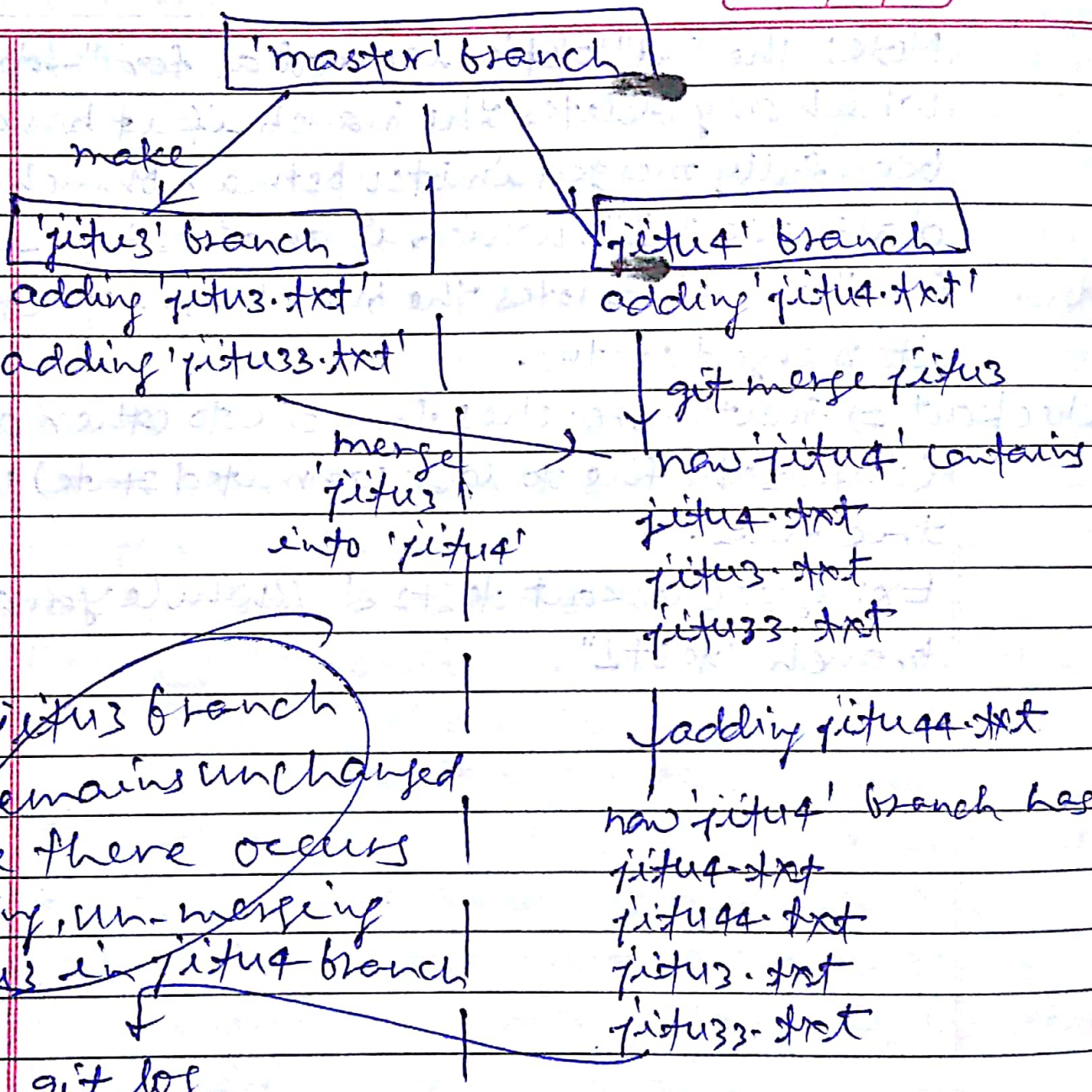
Ex: git checkout test2 ↵ //while you are in branch "test1"

Date 16/02/2018

revert ⇒ पूर्व स्थिति में लौटना

git revert





git log

commit 1 jitu4.txt added
commit 2 Merge branch jitu3 into jitu4
commit 3 update jitu4.txt

git revert <commit-id> // to revert to a particular commit while after merge commit will also be preserved.

In case of merge commit

DELTA Pg No. Date / /

git revert <commit-id> -m (1) or (2)

Note: In case of merge, there are more than 1 parent. So we need to mention to which parent it wants to return i.e. first or second

Ex: In case of merge commit:

commit 4041 600

Merge: 3402e5d 45a4a46

Author: ... (1) (2)

Date: ...

Merge branch 'jithu3' into jithu4

Note: hello-word repository → git-commands.txt