**GIT HUB - COMMON ACTIVITY**

Reference Link - https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

1. Install & Configure GITHUB

* *$ git config --global user.name "John Doe"*
* *$ git config --global user.email* [*johndoe@example.com*](mailto:johndoe@example.com)
* *$ git config --global core.editor "'C:/Program Files/Notepad++/notepad++.exe' -multiInst -nosession"*
* *$ git config --list*
* *$ git config user.name*
* *$ git help <verb>*
* *$ cd /c/user/my\_project*
* *$ git init*

1. Create local repository [Direct creation of folder in windows]
2. Add files to local repo [Direct move/paste for files/folders to local repo in windows]
3. Move files

* *$ git mv file\_from file\_to*

1. Revert changes in local repo

* Revert change made before staging
* *$ git checkout -- <file name>*

1. Remove files from local repo

* *$ git rm PROJECTS.md*
* *$ git rm \\*.txt [Removes all files ending with .txt]*

1. Commit local repo

* Check status
* *$ git status*
* *$ git status -s*
* Check difference
* *$ git diff*
* *$ git diff --staged*
* Stage Component
* *$ git add <name>*
* Commit Changes
* *$ git commit -m “comments”*
* *$ git commit [Provide comments in comment window & Press Esc and type :WQ or :WQ! To come out]*
* *$ git commit -a [Stage & Commit if any existing data modified]*

1. Commit History

* View all commit comments and history
* *$ git log*
* Changes introduced in each commit
* *$ git log -p*
* *$ git log --patch*
* *$ git log -p -2 [It will list earlier two changes introduced]*
* Abbreviated status for each commit
* *$ git log --stat*
* Some more format
* *$ git log --pretty=oneline*
* *$ git log --pretty=format:"%h - %an, %ar : %s"*
* *$ git log --pretty=format:"%h %s" –graph*
* *$ git log --since=2.weeks*

1. Undoing Things

* To change existing commit message
* *$ git commit –amend*
* To add new files to existing commit
* *$ git add <forgotten\_file>*
* *$ git commit --amend*
* To un-stage a staged file
* *$ git reset <staged\_file>*

1. Remote Repository

* Clone remote repo
* *$ git clone* [*https://github.com/libgit2/libgit2*](https://github.com/libgit2/libgit2)
* *$ git clone https://github.com/libgit2/libgit2 mylibgit*
* Display remote repository
* *$ git remote*
* *$ git remote -v*
* Add & Fetch data from repository
* *$ git remote add <short\_name> <URL name> [Add a remote repo & short name]*
* *S git fetch <short\_name> [This command fetch’s all info from remote and places in the name of short\_name/branch\_name]*
* *$ git diff <branch\_name> <short\_name/branch\_name> [This command shows the difference between local and remote branch]*
* *$ git merge <branch\_name> <short\_name/branch\_name> [This command merge’s local branch with the remote branch]*
* *$ git pull <short name> <branch name> [This command dose fetch and merge process]*
* Push the changes from local to remote
* *$ git push <short\_name> <branch\_name>*
* Inspection, Renaming & Removing Remote
* *$ git remote show <short\_name>*
* *$ git remote rename <short\_name> <new\_short\_name>*
* *$ git remote remove <short\_name>*

1. Tagging concept

* *$ git tag [Displays all tag’s available for each commit]*
* *$ git tag -l “v2.” [Displays all versions starting with tag “v2.”]*
* *$ git tag -a v1.0 -m “My version 1.0” [Setting a tag and tag description]*
* *$ git show v1.0 [To view changes in tag v1.0]*
* Tagging to older commits
* *$ git log –pretty=oneline [Display’s 40-byte hashing code and commit description for each commit]*
* *$ git tag -a v0.9 <First 7 bytes of respective hashing code>*
* Pushing older tag copy to remote
* *$ git push origin v1.0 [Pushes the copy of version 1.0 to remote]*
* *$ git push origin --tags [Pushes all commits which has tags to remote]*
* Checking out older tag copy in local
* *$ git checkout v1.0 [Checkout a copy of v1.0 in the local repo]*

1. Git Branching

* *$ git branch <branch\_name> [Create new branch]*
* $ *git log –online –decorate [To check the control/HEAD in which branch]*
* *$ git checkout <branch\_name> [To change the control/HEAD to specific branch]*
* *$ git checkout master [To change the control/HEAD to master branch]*
* *$ git log –online –decorate –graph –all [To check all branch, control & level, even using graphs]*
* *$ git checkout -b <branch\_name> [To create new branch and switch control in same time]*
* *$ git merge <branch\_name> [Two types of merge, Fast-forward & Recursive]*
* Branch Management
* *$ git branch [List out branches in repo]*
* *$ git branch -v*
* *$ git branch –merged*
* *$ git branch –no-merged*
* *$ git branch -d <branch\_name> [Delete the branch]*

1. Remote Branching

* *$ git ls-remote [Display full list of remote references]*
* *$ git remote show [Display full list of remote references]*
* *$ git push <short\_name> <branch\_name> [To push branch from local to remote]*
* *$ git push <short\_name> <local\_branch\_name>:<remote\_branch\_name> [To push the branch in different name from local to remote]*
* *$ git push <short\_name> --delete <branch\_name>*

1. Common Commands

* *$ clear [To clear the screen]*
* *$ cd <repo/file path>*
* *$ cd .. [Step down from current folder]*

Yet to learn core rebasing concepts:

https://git-scm.com/book/en/v2/Git-Branching-Rebasing