Introduction to Git

Mojdeh Rastgoo

Git is free and open source distributed version control system designed to handle small and large projects. You can use git to backup your projects and files, manage your projects in small or large scale and easily communicate with abroad members. This tutorial is designed for beginners and provide basic instructions.

Useful site:

– `https://git-scm.com/`
– `https://git-scm.com/documentation`

Git projects are considered with three compartments, working directory (your local folder), staging area and repository. In the first compartment you can create, delete and edit your files. Staging area is used to keep track of the changes made to the working directory and finally repository is where you can stor your changes to the project as permanent versions of the project.

1 Git

1. Create a directory in your local computer "git-tutorial"
2. Create a text file called "git-tutorial.txt" with the following lines "Welcome to new students."
3. Initializing your git directory → `git init`
4. Checking the status of the git → `git status`
   The changes and un-commited materials are highlighted
5. We can select which files to track and which files to ignore by adding '.git-ignore' file. Create such a file with the following context: `*.pdf`
   `*.txt~`
6. Adding only git-tutorial.txt → `git add git-tutorial.txt`
   Adding all the files in the repo → `git add --all`
7. Commiting the statge → `git commit`
   Write the message of the commit, save and exit
   `git commit -m 'your message'`
   `git commit --allow-empty -m 'your message'`
   `git add --all + git commit` → `git commit -a`
8. Open "git-tutorial.txt" file and modify it by adding aditional lines

9. To see the changes between your current version and previous verison → `git diff`
10. Add your changes and commit again
11. Check the status → `git status`
12. To see the list of commits made in the repository → `git log`
    Each commit has unique 40 character ID, where the author, date, time of the commit as well as its message is highlighted.

13. To check where is the head of your repo → `git show HEAD`
14. Check the number of branches in the current repo → `git branch`
15. Create a new branch → `git branch ‘branch-name‘`
16. Move to other branches → `git checkout ‘branch-name‘`
    A useful tool for visualizing your branches within a repo → `gitg`
17. create a new text file in the current branch, add and commit the changes
18. Checkout to the master branch
19. Mege the created branch to the master branch → `git merge ‘branch-name‘`
20. To merge only specific commits → `git cherry-pick ‘commit-ID‘`
21. In master branch modify “git-tutorial.txt”, add the changes and commit
22. Checkout to created branch and modify “git-tutorial.txt” file, add and commit the changes and then go back to the master and merge the created branch to the master
    There will be conflict which needs to be solved manually by editing the file, such as:
    ```
    Welcome to new students.
    Git is a useful and powerful tool.
    <<<<<<< HEAD
    I create a third line here
    =======
    I create a new line here
    >>>>>>> new
    ```
    HEAD shows changes added in the current branch and new shows the new changes causing the conflict.
23. Solve the conflict, by chosing one version, commit the changes with the message “resolve the conflicts” and merge again.
    merge will state that the branch is already updated
24. Deleting a branch → `git branch -d ’branch-name’`

2 Git-hub

1. Create an account on the github
2. Create a repository online called “git-tutorial”
3. Add the remote address to your local repo → `git remote add origin ’url-address’`
4. Push your repository online → `git push origin master`
5. Fetch the updates from repository → `git fecth`
6. Rebase or merge your local with the online repo → `git rebase origin/master`
   `git fetch + git rebase origin/master` → `git pull origin/master`