Git using guide:

**Author** : Ch8n

**Source**: NewBoston

# GIT Shell: vERSION CONTROL SOFTWARE

## Structure of file storing :

Working area >> Staging area >> Repository

**Working area:** file on which you are working

**Staging area:** updates are stored in staging area using git add operation

**Repository:** Commit add finally to repository

## Navigation:

|  |  |
| --- | --- |
| Code | desc |
| Pwd | Current path |
| cd ~ | root |
| cd .. | One directory up |
| Cd <directory> | Set directory |
| Clear | Clear screen |
| Ls | List all available directories in a folder |
| Ls –la | List all hidden + available directories in a folder |
| Git log | Log of all events committed by current user |
| Git log --author=”<username>” | Log of a particular user **\*\*note:Press “Q” to exit log** |
| Git diff | Display difference/modification in repository and working stage |
| Git diff –staged | Display difference/modification in repository and staging area |

## Config Commands:

|  |  |
| --- | --- |
| Code | desc |
| Git help <command> | List and description of all possible command |
| Git config –list | All configurable properties of current user |
| Git config –global  <property> <”parameter”> | Specific property alteration ex. Git config –global user.name “ch8n” |
| Git init | Make a directory master for a repository |
| Git add . or Git add <filename.extinsion>  Git commit –m “<message>” | Add changes to current path “.” Represent all files in current master folder.  Save changes with a message (take backup to which we can return to.) |
|  |  |

## Files manipulation:

|  |  |
| --- | --- |
| Code | desc |
| Git rm <filename.extension> | Removes file from staging state and working copy. **\*\*note: have to commit to save in repository** |
| Rename file: | **Procedure 1:**  **Step1**:Rename file in explorer  **Step2:** git add <filename.extension>  **Step3:** git rm <previous filename.extension>  **Step4:** git status to check whether git understand you renamed file  **Procedure 2:**  **Step1**:git mv <CurrentFileName.extension> <newFileName.extension>  **\*\*note: moving is similar to renaming** |
| Git mv <filename.extension> <folder>/<optionl:NewfileName.extension> | Moving file from one folder to other **\*\*note:optinally we can rename it** |
| Git checkout -- <filename.extension> | Takeout <filename.extension> from repository and put it in your working space. Restoring. |

## Advance:

|  |  |
| --- | --- |
| Code | desc |
| Git commit –am “<message>” | Method to directly save from working stage>>repository skipping staging area **\*\*note1: Mostly used for updates . \*\*note2:don’t use when rename/move \*\*note3:don’t use when one file is left to edit (as –a is for take all).** |
| Git reset HEAD <filename.extension> | Reset staged file to working file |
| Restore from any previous commit: | **Step1:**git log to get log which contain commit id  **Step2:**git checkout <fewStartingAlphabet of Commit Id> -- <filenameOfwhatyouWantTOrestore.extension>  **Step3:**git commit –m”<message>” |

# GITHUB: PUBLIC PUBLISHED REPOSITORY

To use git shell with Github we need to connect it using **REMOTE.**

|  |
| --- |
| Code: |
| git remote add <tag-Name-To-reffer-URL-of-github-Reportory-youCreated-onwebsite> |
| Example: |
| >>git remote add git\_conn https://github.com/ch8n/Git\_practice\_personal.git {press enter} |
| >>Git remote [to check if above code is working?] |

Send file online using gitshell is called **PUSH.** When you download a file from GitHub its called **Fetching.**

|  |
| --- |
| Code: |
| Git push –u <tag-Name-To-reffer-URL-of-github-Reportory-youCreated-onwebsite> master |
| Example: |
| Git push –u git\_conn master [\*\*note: -u is here acting as select all files] |

# Button on website :

|  |  |
| --- | --- |
| RAW | Copy code:   * Open a repository * Select a file * Select raw button then copy else you would be copying html spaces as well |
| BLAME | a **view mode** in which code appears with the user who edited it |
| HISTORY | shows all commits for one file. |
| Watch/unwatch | How many people are following your project and display on you homepage |
| Star/unstar | Similar to bookmarks[no notification] |
| fork | Branches of the master file |

# issue tab:

Makes a todo list which anyone can fetch and solve and push back for pull request.

In use we put all the details about what is to be done in repository , we add description using text or upload image etc.

We can put **label** to a issue as well.to classify type of issue etc. you can make custom label from label button next to filter in issue tab

We can assig a **Assignee** to the issue , the person responsible for handing the work.

# Wiki Tab:

Wiki for the repository .. you can add home page and other pages to detail out the stuff.

You can add navigation to those using default or custom **sidebars.**

# GITHUB Desktop app: github on local desktop

If want to upload/update our repository and want something to be ignored i.e not to be uploaded on github we make a **gitIgnore** file.

|  |
| --- |
| Step1: Create a txt file with name “.gitignore” in repository |
| Step2: put file type and name you want to ignore like .<folder> or <folder>/<filename.extension> |
| Example: .image\_folder css/index.css |
| To commit changes: changes tab>> fill-summry>>fill-desc>> press Sync. |

Always make update on copy called **branch.** If branch is worked well can be merged back to **Master.**

**To create branch:**

|  |
| --- |
| On github>>repositoryOnwhichYouareWorkig>>BranchOption>>entername in textfield>>create branch. |

**Pull request** is request submitted by branch to submit its changes and merge with master file.