**GIT COMMANDS**

* **Pwd - to see the current directory**
* **Touch filename - to add a new file**
* **Git init - to make a new repository**
* **Git clone - to clone a repository**
* **Git status – to see imp info about repo**
* **Git config -l - for checking user configuration**
* **Git log – to see the commit info**
* **Git log --oneline - to see commits info in a single line( press q to get out of it)**

**Up/down, spacebar, J, F (to scroll)**

**U, B (to scroll)**

* **Git mv name1 name2 > to change file name1 from name to name2**
* **Git log --stat - stats like all the modifications in files**
* **Git log -p - detailed stats patch that is what all was changed**
* **Git log -p --stat - both combined**

**That's right! git log -p -w will show the patch information, but will not highlight lines where *only* whitespace changes have occurred.**

* **Git log -p fififi5 - to open specific commit and all others will also open**
* **Git show - for specific commit**
* **Git show fififi5 - for only this commit**

**STAGING**

**git add <file1> <file2> … <fileN>**

**or Git add - for staging all files from the working directory.**

* **Git commit - to move staged files to the repository.**
* **Git commit -m "Initial commit" - for specifying message simultaneously**
* **Git commit -a -m hello - for skipping the staging area**
* **Git diff and git log -p do the same thing.**

**IGNORING FILES**

**Step 1 - touch .gitignore**

**.gitignore**

**ADD FILES TO .gitignore to ignore some files while staging.**

**TAGGING**

* **git tag -a - to add tag**
* **git tag - to view tags**
* **git tag -d - to delete tag**
* **git tag -a v1.0 a87984 - tagging past commit**

**BRANCHING**

* **Git branch - to see all the branches**
* **Git branch sidebar - to add a branch named sidebar**
* **Git checkout sidebar - shifts the head pointer to sidebar branch, now onwards the commits are added to this sidebar branch.**
* **Git branch -d sidebar - to delete the sidebar branch if the copies of the commits of the sidebar branch does not exist elsewhere.**
* **Git branch -D sidebar - to surely delete the branch named sidebar.**
* **Git branch newbranch SHA - to add a branch - newbranch to the SHA mentioned**
* **Git branch –b twothingsatthesametime - to add a new branch and move the head pointer to the new branch**
* **Git log –all --graph - to show how branches will work with some sort of a tree representation.**

**MERGING**

* **Git merge branchname - to merge a branch named branchname**
* **Git merge --abort - to abort the merging**

**UNDOING**

* **Git commit –amend - to edit the most recent commit**
* **git revert <SHA-of-commit-to-revert> - creates a new commit with reversed changes.**
* **Git reflog - to see history**
* **git reset <reference-to-commit>**

**It can be used to:**

* **move the HEAD and current branch pointer to the referenced commit**
* **erase commits with the --hard flag**
* **moves committed changes to the staging index with the --soft flag**
* **unstages committed changes --mixed flag**
* **Typically, ancestry references are used to indicate previous commits. The ancestry references are:**
* **^ – indicates the parent commit**
* **~ – indicates the first parent commit**
* **Git reset HEAD <file> - to remove a staged file.**

**GITHUB**

* **You’ll need to follow the given steps If you’re using sublime as your default editor.**

1. **open -a /Applications/Sublime\ Text.app to open in sublime**
2. [**https://jeffreyeverhart.com/2017/09/14/open-files-folder-sublime-text-terminal/**](https://jeffreyeverhart.com/2017/09/14/open-files-folder-sublime-text-terminal/) **link to use sublime with git.**

|  |  |
| --- | --- |
| **git clone URL** | [**Git clone is used to clone a remote repository into a local workspace**](https://git-scm.com/docs/git-clone) |
| **git push** | [**Git push is used to push commits from your local repo to a remote repo**](https://git-scm.com/docs/git-push) |
| **git pull** | [**Git pull is used to fetch the newest updates from a remote repository**](https://git-scm.com/docs/git-pull) |

* **Git remote -v - to see the links for fetch and push**
* **Git fetch - git fetch fetches remote updates but doesn't merge; git pull fetches remote updates and merges.**

| **Command** | **Explanation & Links** |
| --- | --- |
| **git remote** | [**Lists remote repos**](https://git-scm.com/docs/git-remote) |
| **git remote -v** | [**List remote repos verbosely**](https://git-scm.com/docs/git-remote#Documentation/git-remote.txt--v) |
| **git remote show <name>** | [**Describes a single remote repo**](https://git-scm.com/docs/git-remote#Documentation/git-remote.txt-emshowem) |
| **git remote update** | [**Fetches the most up-to-date objects**](https://git-scm.com/docs/git-remote#Documentation/git-remote.txt-emupdateem) |
| **git fetch** | [**Downloads specific objects**](https://git-scm.com/docs/git-fetch) |
| **git branch -r** | [**Lists remote branches**](https://git-scm.com/docs/git-branch#Documentation/git-branch.txt--r)**; can be combined with other branch arguments to manage remote branches** |

* **git remote update will fetch the contents of all remote branches and allow us to merge the contents ourselves.**
* **Nano filename – to open file in editor.**
* **Making changes in readme – open file in using nano command and then follow these steps to save the file------- ctrl+o + enter + ctrl+x**

**Git push orgin master**