**HELP**

git help -> list of commands

git help command -> help with command

**CONFIG**

git config --global user.name‘username’ -> set username

git config --global user.email ‘email’-> set email

git config --global color.ui ‘true|false’ -> set pretty colors

git config --list -> show set configuration

**INIT/CLONE**

git init -> start local repository

git clone url\_to\_remote\_repository -> clone remote repository

git clone url\_to\_remote\_repository foldername -> clone remote repository to folder with given name

**STATUS/DIFF**

git status -> show what’s changed since last commit

git diff -> show unstaged differences since last commit

git diff --staged -> show staged differences since last commit

**ADD**

git add filename/directory -> add files/directory to staging area

git add . -> add all new, modified and deleted files to staging area

git add -A -> add all new, modified and deleted files to staging area

git add --all ->add all new, modified and deleted files to staging area

git add -u -> add modified files to staging area

\* -> placeholder for any numbers of any character

**COMMIT**

git commit -m ‘message’ -> commit changes in staging area

git commit -a -m ‘message’ ->add all changes from tracked files and commit

git commit --amend -m ‘message’ -> add to the last commit with new commit message

git commit -a -> in case of conflict returns to merge commit message creation view

**LOG**

git log -> show log

git log --oneline -> show log with commits in one line

git log --graph -> show log with graph of different merged branches

git log -pretty=fromat: placeholders -> show formatted log with selected placeholders

git log --oneline -p -> show differences between commits

git log --oneline --stat -> show number of insertions and deletions between commits for each file

**BRANCH**

git branch -> show list of local branches

git branch -r -> show list of remote branches

git branch branch\_name -> create new local branch with given name

git branch -d branch\_name -> delete branch (branch will be not deleted if it’s not merged with other branch)

git branch -D branch\_name -> delete branch even if it’s not merged with other branch

**CHECKOUT**

git checkout --filename -> discard file changes since last commit

git checkout branch\_name -> switch on other branch

git checkout -b branch\_name ->create new branch and switch on it

git checkout tag\_name -> switch on tagged commit

**MERGE/REBASE**

git merge branch\_name -> merge given branch to current branch

git rebase branch\_name -> rebase given branch to current branch

git rebase -> rebase current branch (usually master) with branch fetched (usually origin/master) by git fetch command

git rebase --continue -> continue rebasing after solving spotted conflict (first you should add changes to staging area)

git rebase --skip -> skip patching in case of spotted conflict

git rebase --abort -> abort rebasing in case of conflict

**REMOTE**

git remote add remote\_name your\_github\_url ->assign local repository to remote repository with remote\_name (remote\_name is usually origin)

git remote -v -> show list of remote repositories

git remote rm remote\_name -> remove remote repository (remote\_name is usually origin)

git remote show remote\_name -> show our remote branches (remote\_name is usually origin)

git remote prun remote\_name -> clean up deleted remote branches (remote\_name is usually origin)

**PUSH**

git push -u repository\_name branch\_name -> push to repository (repository\_name is usually origin) (branch\_name is usually master)

git push repository\_name branch\_name -> push branch to repository (repository\_name is usually origin)

git push repository\_name :branch\_name -> delete remote branch (repository\_name is usually origin), doesn’t delete local branch

git push --tags -> push local tags to remote repository

**PULL/FETCH**

git fetch -> fetches changes from remote repository

git pull -> fetches changes from remote repository and merges them with local repository

**RESET**

git reset HEAD filename -> unstage file with given filename

git reset --soft HEAD^ -> reset last commit to staging

git reset --hard HEAD^ -> discard last commit and all changes

**TAG**

git tag -> show list of tags

git tag -a tag\_name -m ‘tag description’

**ARCHIVE**

git archive --format=format –output path\_to\_file SHA -> creates archive with given format (usually zip) in given path\_to\_file from SHA (usually master)

**VI COMMANDS**

j -> down

k -> up

h -> left

l -> right

ESC -> leave mode

i -> insert mode

:wq -> save and quit

:q! -> cancel and quit

**LOG PLACEHOLDERS**

%ad -> author date

%an -> author name

%h -> SHA hash

%s -> subject

%d -> ref names