Git-Github Cheat-sheet

Commands	Description
	Initialization
git	If installed correctly, shows a list of available commands
git init	Create an empty Git repository or reinitialize an existing one
git clone	Clone a repository into a new directory
git tag	Creates a snapshot of a project at a certain time. When our project/branch is ready to put in production, we first create a tag with version number so that it is always available for the future reference. For help see: git tag -h
setx HOME "<Path>"	Changing Home Directory
	Work on current changes
add	Add file contents to the index
mv	Move or rename a file, a directory, or a symlink
reset	Reset current HEAD to the specified state
git rm <filename> **----------** **git commit -m "<commitMessage>"**	Remove files from the working tree and from the index. Remember to use git **push origin <branch_name>** to push the changes to remote.

nicks9188-@github

Git-Github Cheat-sheet

git rm --cached <filename> ---------- **git commit -m "<commitMessage>"**	If want to remove the file only from the Git repository and not remove it from the filesystem. Remember to use git **push origin <branch_name>** to push the changes to remote.
git stash	git stash -h
git stash pop	git stash pop -h
git commit -am "<message>"	= git commit -a -m = git add . + git commit git add . = git -a
	Examine the history and state
bisect	Use binary search to find the commit that introduced a bug
grep	Print lines matching a pattern
log	Show commit logs
show	Show various types of objects
status	Show the working tree status
git <command> -h	Git help
git log	Shows us a list of the commits in our project.
	Reverse & Redo
git log	This will show the last commit. Copy the commit code.

nicks9188-@github

Git-Github Cheat-sheet

git revert <hash/commit code>	Now use that code with git revert <copied code> to revert that commit coming back to previous one. *Creates new commit to undo the changes.*
:q!	After reverting an editor window will open. This is to close that window. If working on remote, remember to use **git push** to after this to reflect the changes on remote.
or	
git reset <commit-code>	
git reset --soft <commit-code>	Removes commit from history but keeps changes.
git reset --hard <commit-code>	Removes commit from history and removes changes.
	Use **reset** when working locally or alone Use **revert** when working with team. When we are working with a team, team is dealing with same code. So if we are making any reverse, we should use revert instead of reset.
	Working with remote repository
git remote -v	Shows the version/status of remote repository.
git remote add origin <github repository url>	This will add the local repo to the github repository. In case credentials for github is not configured, in first commit it will ask for github account credentials.
git push	To send the local file changes to remote repository. Just using git push without any suffix command will tell us about the branches that we have pushed.
git fetch / **git fetch origin master**	Retrieve new work done by other people. Fetching from a repository grabs all the new remote-tracking branches and tags without merging those changes into our own branches.
git push --set-upstream origin <branchName>	When created a branch locally, but didn't added that to remote. This command will add the branch and its changes to remote.

nicks9188-@github

Git-Github Cheat-sheet

	Can also be added by simple push. If not, use this command.
git pull <remoteName> <branchName> --------- **git pull origin <branchName>**	It is a convenient shortcut for completing both git fetch and git merge in the same command. Because pull performs a merge on the retrieved changes, we should ensure that our local work is committed before running the pull command. If we run into a merge conflict we cannot resolve, or if we decide to quit the merge, we can use git merge --abort to take the branch back to where it was in before you pulled.

Adding credentials permanently
 	
(if have to enter credentials again n again)
	You can store your credentials using the following command
	$ git config credential.helper store
	$ git push http://example.com/repo.git
	Username: <type your username>
	Password: <type your password>
	Also I suggest you to read
	$ git help credentials
	Or
	
	$ git config --global credential.helper wincred

	Branching
git branch	will list all the branches available in our current project locally. The current branch colour will be green.
git branch -a	will list all local as well as remote branches.
git branch <branch_name>	Will create a new branch locally. NOTE: initial commit is necessary in order to create a branch. Without any commit even master branch doesn't exist.

Git-Github Cheat-sheet

	Working on Branches
echo mater branch > newfileMaster.txt	Will Create a file named newfileMaster.txt on master branch. [Note: echo <filecontent> > <filename.extension>] Will create a file but first check at which branch we are.
git checkout <branchName>	To change branch.
git merge <branchName>	To merge two branches. Note: • Remember to be on master branch to merge changes to master. • Remember to push the changes to if working on remote. Merging the branch just merges the files to the master, leaving the initiated branch as it is.
git diff <sourceBranch ><targetBranch>	Shows us the difference between two branches.
git rebase	Takes the commit from original branch and applies them into current branch before the current changes. git rebase -h
	Cleaning-Deleting
git clean	Removes untracked files. git clean -h
git branch -d <branchname>	Will delete the branch from local.
git push origin --delete <branchName>	Will delete the branch from remote.
If unable to delete remote branch, try these	git branch -r -d origin/<branchName> Or

nicks9188-@github

Git-Github Cheat-sheet

	git remote prune origin Or git fetch origin --prune