Starting Git

# Chapter 1 Introduction

1. Git is a distributed version control system (DVCS) and source code management software.

Note: Version Control System (VCS) is a software that helps software developers to work together and maintain a complete history of their work.

Listed below are the functions of a VCS:

* Allows developers to work simultaneously.
* Does not allow overwriting each other’s changes.
* Maintains a history of every version.

1. Git was initially designed and developed by Linus Torvalds.

# Chapter 2 First Time Installation

1. With help of NPM as npm install git-download --save
2. After that, you have to set global variables such as user Name and Email with following commands.

git config --global user.name “Your User Name”

git config --global user.email “Your Email in git”

1. To see your git configuration, use git config --list.
2. To get details of any command, use git “command-name” --help. i.e. git config –help.

# Chapter 3 Creating Repository For your existing project

1. Go to project directory, use command git init. It will create a .git folder in your project.



1. Here Working Area can be understood as local repository and Staging Area is place, where you have to decide which file you should commit to Git Repository.
2. To get status for your Working directory to git directory, use git status. It will list all the files that have not committed yet, or changed since from your last commit.
3. To add whole project for commit, use git add -A.
4. To add an individual file, use git add filename.
5. For removing files from staging Area, use command git reset filename.
6. For removing all, use command git reset.
7. For any file that you don’t want in your git repository, create file named ‘.gitignore’ text file. Here you have to list those files by relative path.

# Chapter 4 Your first Commit

1. First, create a remote repository to <https://github.com>. Signup and create repository for your project.
2. After creating repository, you will get a URL. This URL is used for commit.
3. For adding your remote repository for your project, use command git remote add origin ‘Your Repository URL’. If any remote origin already exists, then remove it by git remote remove origin.
4. For each commit to git, you have to leave a message regarding what changes are made to these files. These messages help developer to keep track of versions.
5. For your commit, use command git commit -m “your message”. Now if you check status, you will find a message as ‘Nothing to commit your working directory is clean’.
6. To get details about last commit, use command git log. It will give you a brief about your last commits.
7. For pushing the committed data, use command git push -u origin master. Here origin is your repository name and master is your branch.

# Chapter 5 Cloning a remote repository

1. Cloning is a process of getting a copy of your repository to local machine.
2. For cloning, use command git clone yourRepositoryURL.

# Chapter 6 Update repository

1. Use command, git diff. This command will show all changes you made. You can also use command, git status. It will list all files that you have updated since last commit.
2. Now add these files for commit by git add -A, and then push them.
3. It may happen that other developers are making changes for some feature on which your code depend.
4. So, to get the latest copy of project, use command git pull origin master.

# Chapter 7 Origin and Branch in Git

1. **Git** has the concept of "remotes", which are simply URLs to other copies of your repository. When you clone another repository, **git** automatically creates a remote named "**origin**" and points to it. **origin** is the default alias to the URL of your remote repository. An **alias** stores the URL of repositories
2. Basically, a branch in git represent a particular feature. By default, git uses master branch.
3. To list all branches, use command git branch. The one followed by \*, represents the current working branch.
4. To create a new branch, use command git branch branchName.
5. To select another branch, use command git checkout branchName.
6. Then, after updating, you can push into that branch followed by add and commit command.
7. Now to merge that branch, 3 steps are followed as
8. For locally, git branch --merged.
9. For remote, git merge branchName.
10. git push origin master.
11. Now delete that branch locally as, git branch -d branchName.
12. Now delete from your remote repository, use command

git push origin –delete branchName.

# Chapter 8 Final Conclusion

1. For logout, You may have to check windows credential manager and delete the GitHub entry under control panel > user accounts > credential manager > Windows credentials > Generic credentials to delete your git credential from local.

# Que1. [**Can a project have multiple origins?**](https://stackoverflow.com/questions/11690709/can-a-project-have-multiple-origins)

Ans: You can have as many remotes as you want, but you can only have one remote named "origin".

git remote add github https://github.com/Company\_Name/repository\_name.git

# push master to GitHub

$ git push github master

# Push my-branch to github and set it to track github/my-branch

$ git push -u github my-branch

# Make some existing branch track github instead of origin

$ git branch --set-upstream other-branch github/other-branch