Git is the most popular, widely used, and modern version control system. It has almost become a defacto standard in the IT Industry in the category of version control systems. Unlike most version control systems, it is a distributed system by nature. In Git, the local copy of source code on your machine is a complete repository and is fully-functional, which makes working offline and working remotely very easy for the developers. After making changes in local repository, the developer commits his/her work locally, and later sync this local copy with the copy on the git server. Bit Bucket, Github, and Amazon's Code Commit are few of the examples of Git Server Services available in the cloud. Informa has chosen Github.com as their base. Using Git may be intimidating in the beginning but believe me once you get used to it, it all comes natural.

The first thing I want you to do is, go to Github.com and register yourself. Secondly, download and install Git from <https://git-scm.com/download/win> or this <https://git-scm.com/download/mac> depending on your operating system. Git installation will also install a tool called GitBash which comes along with the same download. Gitbash will let you manage your source code repositories with Github or any other Git Server Services as I mentioned above. No matter what management tool you use, the driving engine underneath is always Git. My personal favorite is Cmder ([http://www.cmder.net](http://www.cmder.net/)/) but we will use Gitbash for this blogpost. One more thing I want to mention here is, there's hardly any difference in the process of managing your repos whether you have personal account or business account.

Let’s get to the action:

Step 1 -  Login on github.com with your account and create a new repo.

Step 2 - Please check on the README file. This will create README.MD file used for comments/info about the project. Of course, we can add this file later but it takes 3 to 4 steps to do that. In case you have business account, please check private so it can be shared only by your company accounts and not the public. Enter your Repo name and click on create repository. ​

Step 3 - Go back to Windows Explorer and choose the root folder where you want to clone all your projects and start Gitbash, as shown in this example C:\TABISH\APPS\CLONES

Step 4 - Switch back to your Web Browser and copy your repo URI as shown below.

​Step 5 - Switch back to your Gitbash window and paste this Repo after typing clone command as shown below:

​Step 6 - After cloning, please change your directory to Test by using "cd test" as shown below. Git also create a hidden folder under your repo folder and call it “.git”. This is git’s playground and not be touched.

Step 7 - Start your IDE and make changes to any files or creat a new file to upload back to Github.

Step 8 - Switch back to GitBash window and add all the files using "git add . " command, this puts all the changes/new files along with new/changed folders and subfolders in staging area as shown below:

Step 9 - use git status to check and confirm. This step is optional.

Step 10 - Commit these changes to create a snapshot with some hint message as shown below:

Step 11 - Push these changes up to Github Repo in the cloud, This copy would be treated as master copy and all the developers when adding new features in the repo can branch off from the master branch.

Step 12 - Switch back to Web browser and refresh the screen.   
Voila! You have your changes uploaded.

This post is just an intro to Git. Git has so much more to offer like Branching, Forking and much much more.

**Branching** is the practice where the deveopers create copies of repos to work in parallel versions, which retains the original repo intact and they work on the branch making changes to it. Each copy made by the developer is considered a branch and the original repo from which the branch is taken is referred to as the master.  This is typically done when the developer wants to create new or update the existing features in the master. The intention is to merge back these branches into master branch. The owner of the master branch could accept or reject the new/updated features branch from any developer, so the feature can be reworked when rejected and reaccepted and merged when done into  master branch.

**Forking**is when the developer takes the copy of the existing repo and develop and maintain it as an entirely new repo. To call itself a fork, the newer repo must have its own name and it is managed separately from its original predecessor. Following is the list of commonly used Git Commands: [Cheatsheet](https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf)​.

Hope this helps you to get started. All the best!

For more information on this topic, you can reach me at tabish.sayed@informa.com.   
- Tabish Sayed