How to Install Git

1. Go to <https://www.git-scm.com/downloads>
2. Under Downloads, select the operating system (ex: Windows). Download the exe file & install it
3. On the options, select the option that install Git Bash.
4. On adding path to environment, select “Use Git and Optional Unix tools from the Windows Command Prompt.”
5. Leave the rest of the options as default.
6. Click Install
7. Once installation is complete, you can verify if Git is installed by typing “git --version”

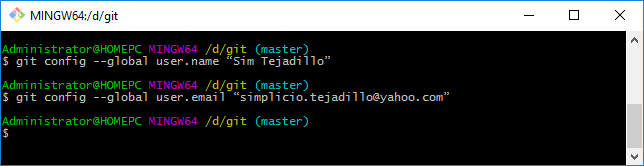
Setup Config Values

This is very important for some other developers since they need know who’s checking in and out a code and making changes.

1. To configure Git account for github, type the following command:

git config --global user.name “Sim Tejadillo” 🡪 This will setup your Git user name.

git config --global user.email “simplicio.tejadillo@yahoo.com” 🡪 This will setup your email

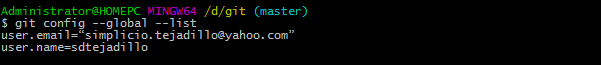


1. Confirm that you have set your username correctly with the following command.

git config user.name and git config user.email or git config --global --list

**Tip**: You can reset reset your git config -global settings you can use either of the following:

Edit the file via vi ~/.gitconfig or git config --global --unset-all user.name



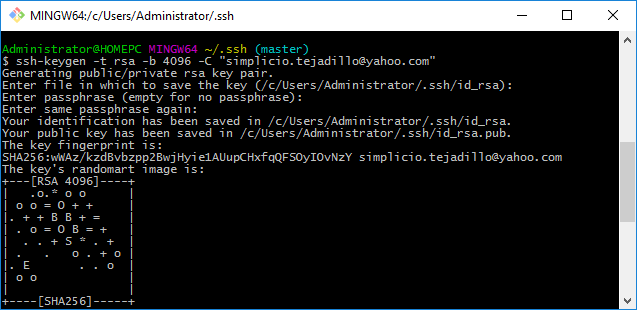
Initialized a Repository from Existing Code

1. Type the command: git init 🡪 This will tell Git to start tracking the directory
2. To remove the .git tracking directory, you need to use command: rm -rf .git

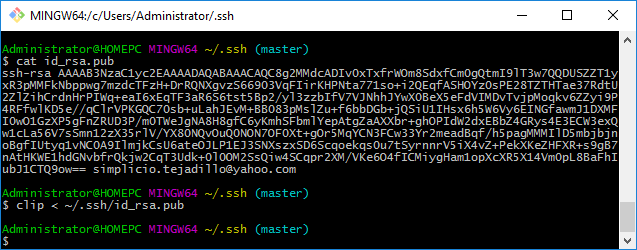
Create Public and Private Key

Setup private and public key pair that we can upload to github so that we can authenticate to github without entering our username and password.

1. Open Git Bash
2. Type “ssh-keygen -t rsa -b 4096 -C "your\_git\_email@example.com"
3. You can hit “Enter” for everything but you can also enter your own passphrase.

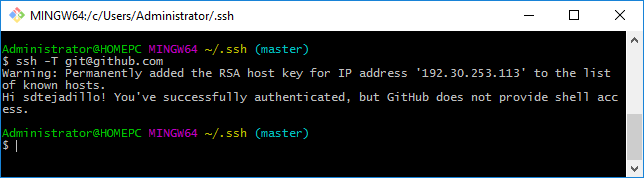


1. Go to the directory where .ssh is located (ex: C:/Users/Administrator/.ssh)
2. Using Git Bash, navigate to ~./ssh folder
3. Open the public key file using the command “cat id\_rsa.pub” or copy it to your computer clipboard by typing this command "clip < ~/.ssh/id\_rsa.pub”
4. Login to <http://github.com/> 🡪 Go to “Settings” 🡪 Under Personal Settings, select “SSH and GPG Keys”
5. Under SSH Keys, click “New SSH Key”.
6. Add SSH Keys Title and paste the key into the “Key” text box.
7. Click Add SSH Key button



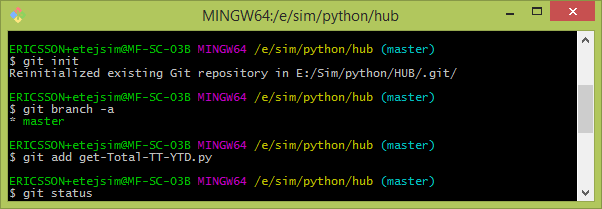
Testing SSH Public and Private Key Authentication

1. Open Git Bash
2. Type “ssh -T [git@github.com](mailto:git@github.com)”



Initializing Git Repository (Local)

1. On Git bash, type “git init”
2. Type “git remote add origin https://github.com/sdtejadillo/demo.git”

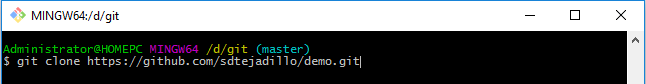


Cloning Remote Repository (Github)

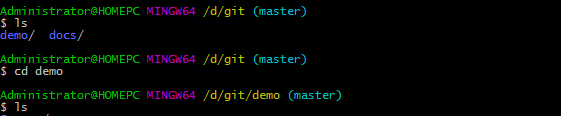
**Important**: We need to configure our Git account first for github repository.

1. Login to <https://www.github.com/> using your username and password
2. Select your repository to open it.
3. Copy the link to your repository ex: <https://github.com/sdtejadillo/demo.git>
4. Open Git Bash, type: git clone <url> <local directory> or git clone ../remote\_repo.git .

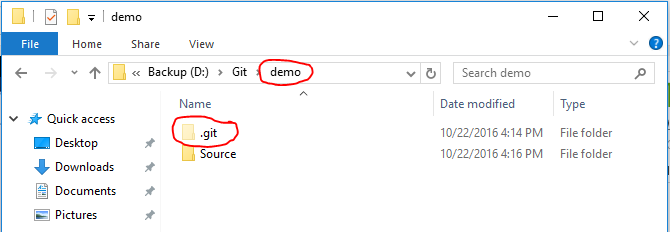
ex: “git clone <https://github.com/sdtejadillo/demo.git>”



1. Once closing is complete, we can check if the local repo in Gitbash by:

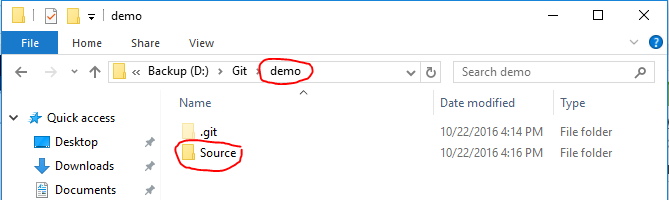


1. Or you can open file explorer and navigate to the local repo directory.

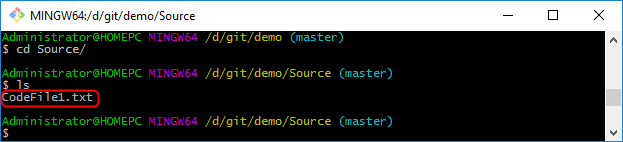


Upload Folder and Files into Remote Repository in Github

1. Create a sample directory under your local Git directory that is monitored by Git (ex: Source)
2. Open the new folder (Source) created and create a text file underneath (ex: HelloWorld.txt)
3. Type some text inside ex: HelloWorld.txt
4. Save your ex: HelloWorld.txt file
5. Close the file



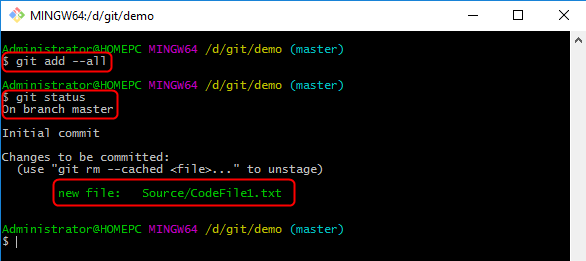
Verify new file to be uploaded in Git bash



1. To add files and folders into your remote repository, you need to type the following command:

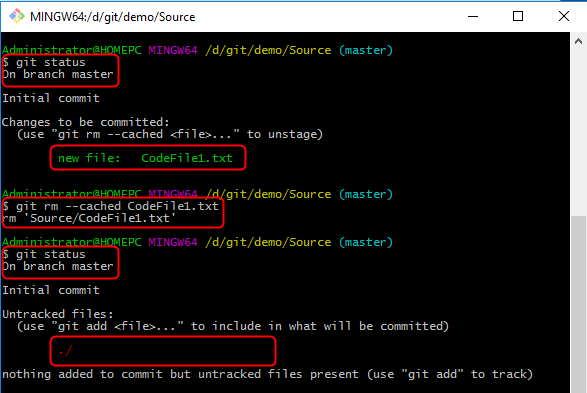
git add filename.txt 🡪 This will add a single file into Git staging area

git add -A 🡪 This will add all files & subfolders folders into staging area. Equivalent to git add . or git add –all



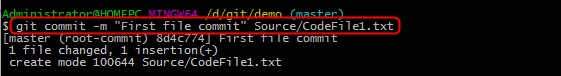
**Important**: To unstage files and folders from staging area you can use the command:

git rm --cached filename



1. To commit a file/folder change in the staging area you need to use the command:

git commit -m “Commit message very important” filename



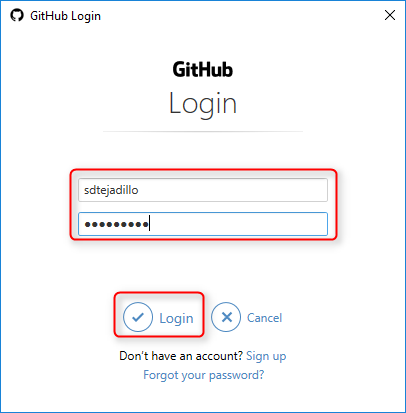
1. To push the file/folder change in the remote repository (ex: Github account) you need to use the following command: git push -u origin master

**Note**: -u is for your username and password flag

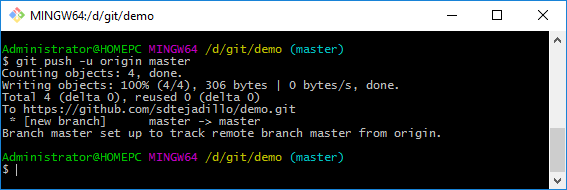
origin is from your local directory (local repo)

master is for your remote master directory (github)

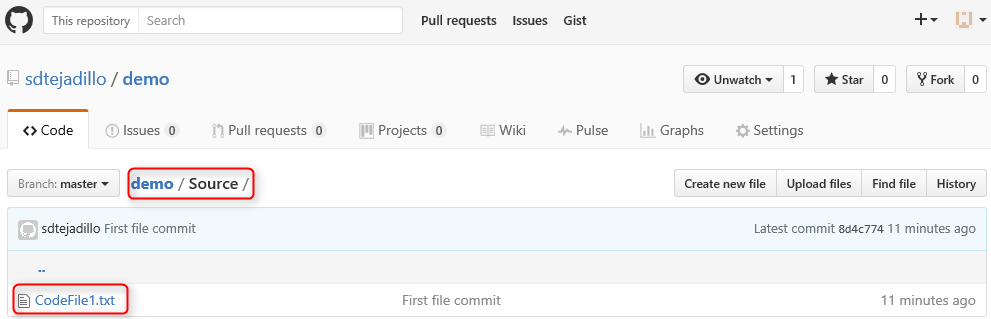
1. It may asked you to enter your remote repository user name and password.



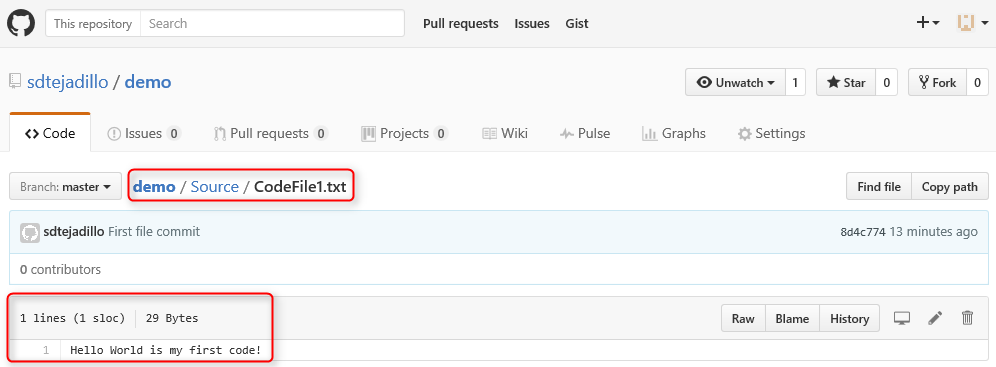
1. Once Git push is complete, you will see the following:



1. Open your Github account ex: <https://github.com/sdtejadillo/demo> and check if the new files or file changes are reflected.



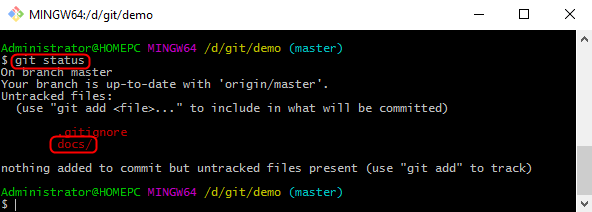
1. Open the file to see the content.



Ignore Folder and Files in the local repository

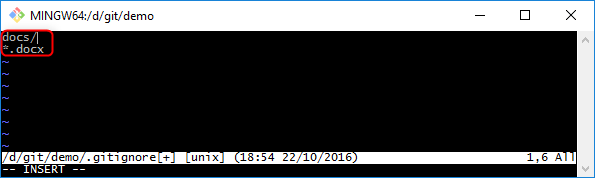
1. Create a .gitignore file from inside your machine local repository that you’re tracking with GIt.

touch .gitignore 🡪 To create a new file .gitignore. This is just a simple text file that tells Git on what to ignore.

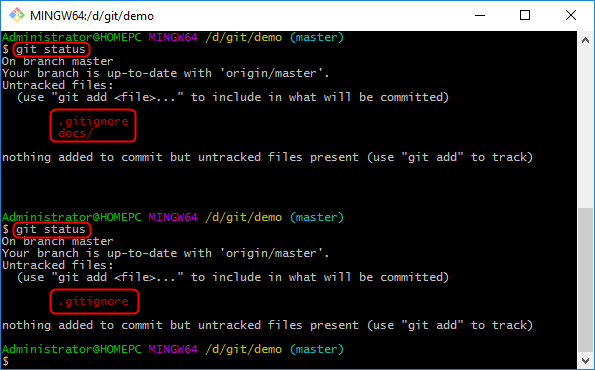


1. To add files & folders into the .gitignore file, just use this command:

vi .gitignore 🡪 The add the list of files and folders that you want to ignore.



1. On Git bash, type “git status” again and you will noticed that files and folders listed on .gitignore file is not anymore listed on staging area.



Add Files to Staging Area

1. To add new files or modified files into the staging area you need to use any of the following:

git add -A 🡪 This will add all files that has changed into the staging area

git add filename 🡪 This will add individual file that has changed into the staging area

git status 🡪 This will show you the status in the staging area

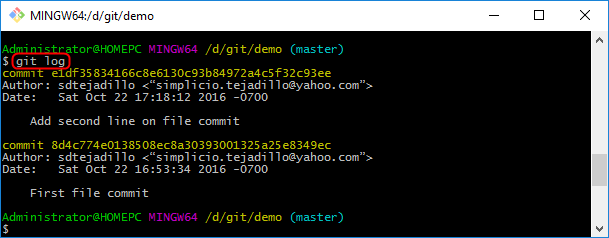
git reset filename 🡪 This will reset a file that was committed and submitted into the staging area

git reset 🡪 This will reset all files that was committed and submitted into the staging area

View Commit that you made

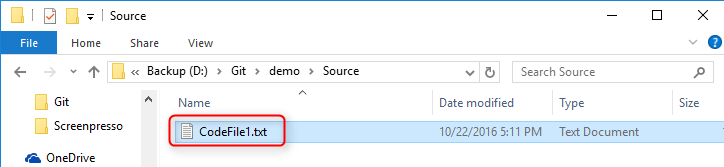
1. To view the commit that you have made use the below command:

git log 🡪 This will show you the hash number of your commit

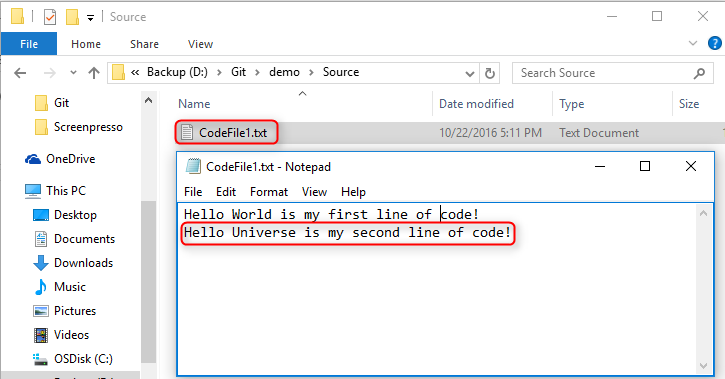


Code File Changes, Commit & Push into Remote Repository in Github

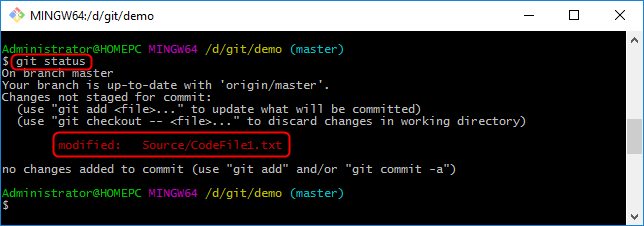
1. Open your code file using Visual Studio or PyCharm or NetBeans or others and make the change.



1. Edit the file and add a second line to it.

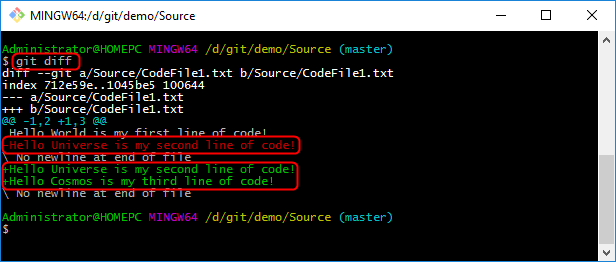


1. Save the file
2. Open Git Bash
3. Type the command: git status



1. Navigate to the folder where the modified file is located then type the command:

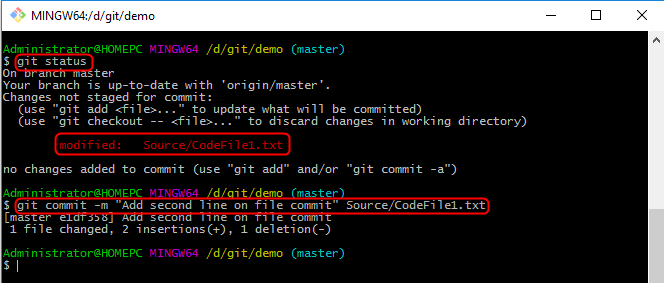
git diff 🡪 Show will show me that changes that you made to the code.



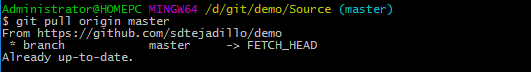
**Note**: The “+” is the line added on the code where “-” is the line remove from the code.

1. To add all changed files into staging area then type: git add –A
2. Type git status again to view the files that are ready to be commited.
3. To commit a file/folder change in the staging area use the command:

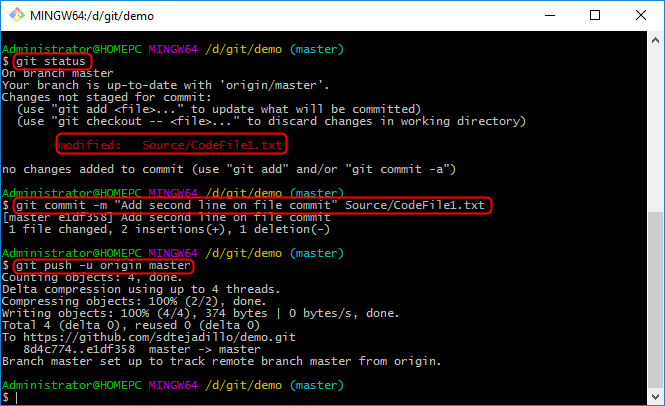
git commit -m “Another commit message very important” filename



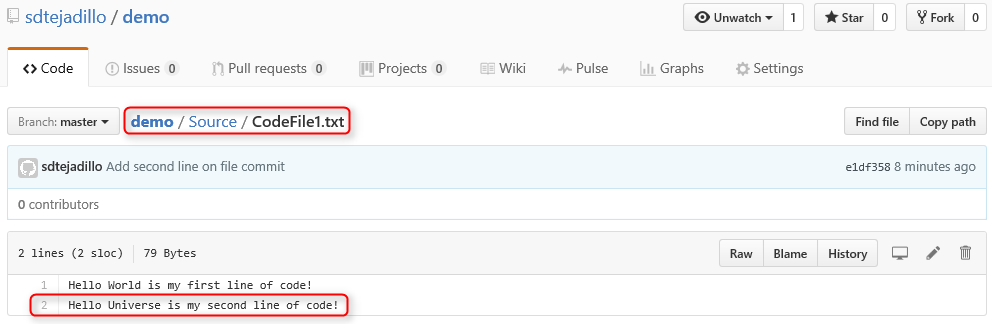
1. Type, git pull origin master 🡪 This will get or pull the latest changes of the code from the remote repository since the last time we made a pull from that it. We need to remember that on a project that lots of developer are pushing code then it’s always good to sync first before submitting a push or merged into the central repository.



1. Type, git push origin master 🡪 This will push your changes from staging area into the master branch of the remote repository. Once git push is complete, you will see the following messages:



1. Check the file in your github repository and see the changes.



Viewing Info. On Remote Repository

1. To view information on the remote repository, you need to use the following commands.

git remote -v 🡪 This will list the information of the repository

git branch -a 🡪

Central Version Control System (SVN)

A central version control system is located in one place where developers can checkout code from this repository and make their changes and check everything back in. This can be problematic if the central VSC server or central repository is down or you don’t have network connection to the central repository server. You will only be able to see the files that you checkout from that repository and no other information from that central repository. If that central repository file system is corrupted then the only way to recover is by backup.

Distributed Version Control System (GIT)

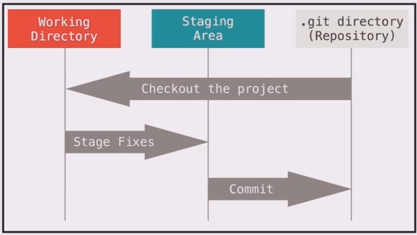
A distributed version control system is also located in a main single repository but every developer has a clone copy of that repository into their local machine where they make and commit their changes. The local and remote repository content is always the same based on the last time where you sync them together. If you don’t have access to that master repository then you still have access on all the changes made to your local repository. With this, all developers has a backup of the main repository on their machine.

Need Help on Git?

git help <verb> 🡪 Ex: git help config

git <verb> --help 🡪 Ex: git config --help or git add –help

Git Process Flow



Working Directory – This is your local clone copy of your repository where you make the changes

Staging Area – This is where we organized what changes we want to commit into our repository

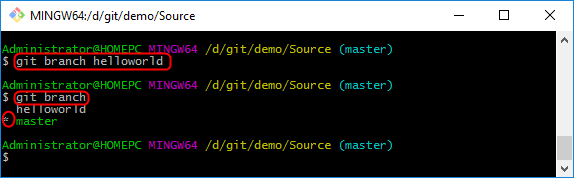
.Git Directory (Repository) – This is the main master repository where we push our changes

Create a Branch for a Desired Code Feature

1. To create a branch of your desired code feature and then start working on that branch.

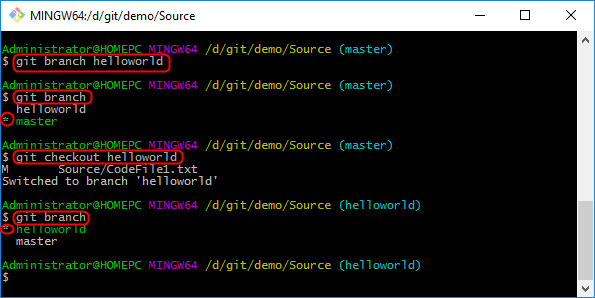
git branch <name-of-the-branch> 🡪 ex: git branch helloworld

1. To view the list of branches, type this command:

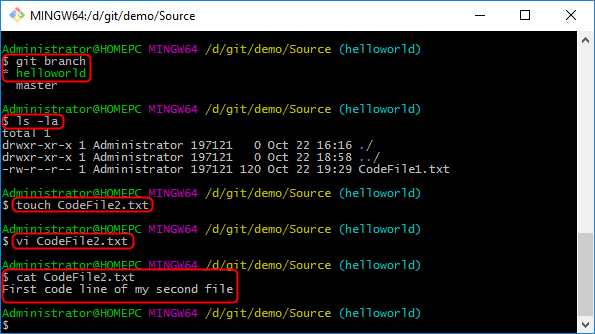
git branch 🡪 Note: \* before the branch name means this is the current branch you’re currently working

1. To start working on the “helloworld” branch then you need to check it out by using the command:

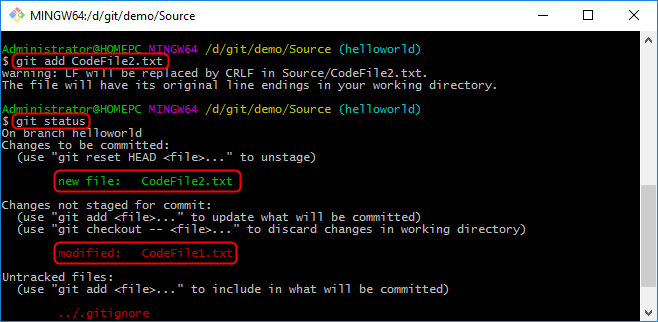
git checkout <name-of-the-branch>



1. You can now start working on the new “helloworld” branch and start adding or editing files from it.

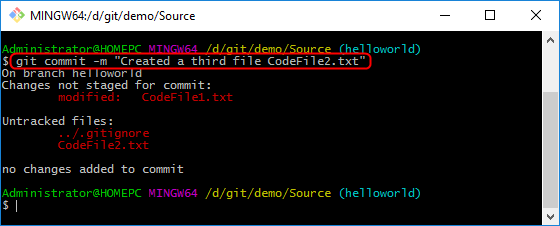


1. Type, git add <filename> or git add -A ex: git add CodeFiles2.txt 🡪 To add changed to staging area



1. To commit the changes, type: git commit -m “Created a third file CodeFile2.txt”

Note: Now, this have successfully commit your changes into your local “HelloWorld” branch but won’t affect your local master branch or remote repository in Github.

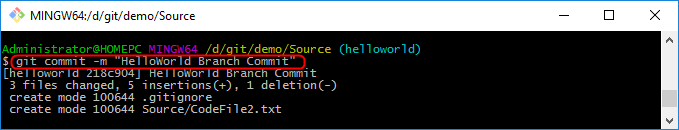


Push Branch into Remote Repository after Commit

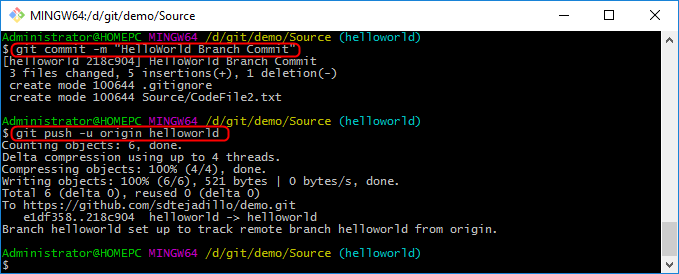
1. Type, git branch to see the current branch you’re currently working on.



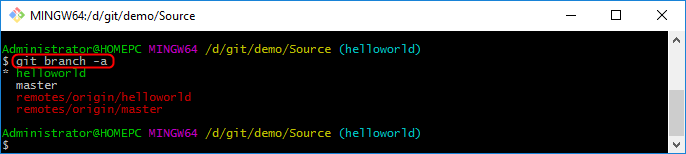
1. Type, git commit -m “Your changes comment”



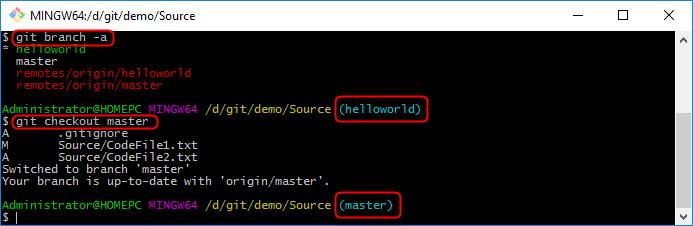
1. Type, git push -u origin helloworld to upload/checkin your changes into your local branch “HelloWorld” to the remote repository in Github.



1. Type, git branch -a to view all of our branches

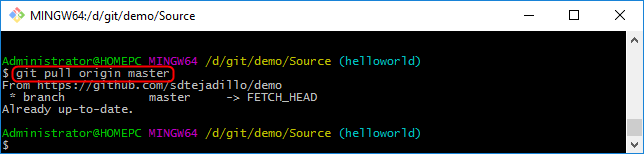


**Note:** The branch that are mark as red are your local branch that you need to push into your local repository. Most companies use branch to run unit test and others.

1. Once all branches unit test results are ok, then we are now ready to merge our branch with the master. To do that we need to use the following command. 

**Note**: After we checkout our local branch you will see that we now switch to the master branch

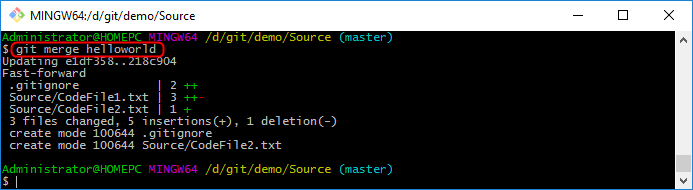
1. Now, we need to pull the changes down by typing git pull origin master to update your branch copy from the master before pushing any changes.



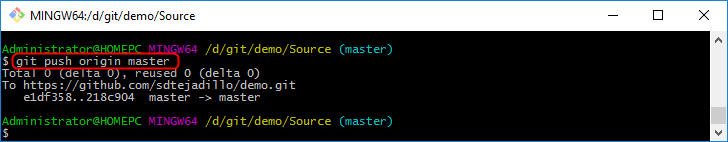
1. Type, git branch --merged to tell us the list of branches that we’ve merged so far. Note, the “helloworld” branch didn’t appear here yet since we didn’t merged it yet.



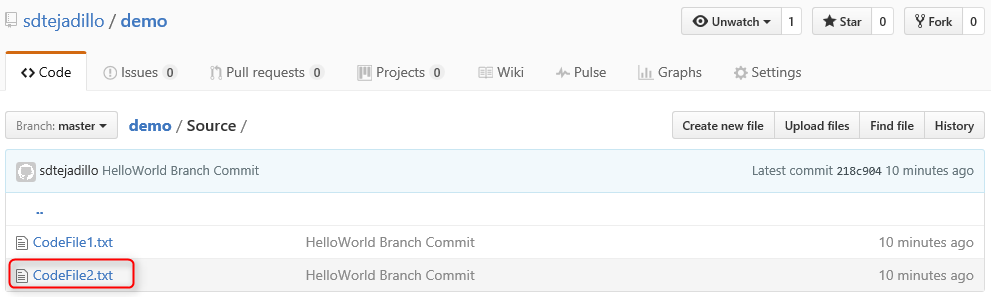
1. Now, in order to merge that new branch to master you need to use the command git merged <name-of-the-branch> ex: git merged helloworld



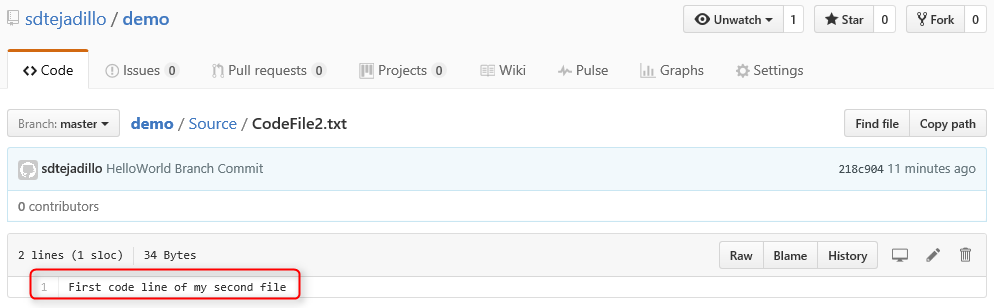
1. Type, git push origin master to push the changes into the remote master branch which is the remote repository in github.



1. Verify the changes that has merged by logging to your github repository and check your repository folder.



1. Verify the changes made to a file by opening the file.



1. Type, git branch --merged to view the branches that has successfully merged all the changes to the master.

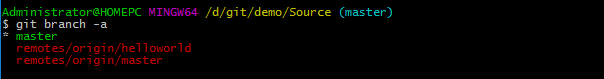


1. We can now delete that branch “helloworld” locally by typing this command

git branch -d helloworld



1. Type, git branch -a and see that “helloworld” branch is now deleted from your local repository. But you still see the remotes/origin/helloworld in the remote repository.

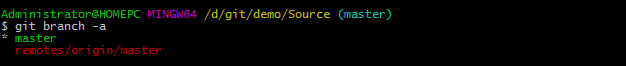


1. To delete a remote branch in the repository, use this command.

git push origin --delete helloworld 🡪 Origin refers to our local repository & helloworld is the branch.



1. Type, git branch -a and see that “helloworld” in the remote repository (previously marked in red) is now deleted and gone.



How to Switch Branch

1. Open Git Bash, type git checkout <local-branch-name> ex: git checkout helloworld
2. Type, git branch to see the current active branch

