##Users need to be part of github.users.

[github.users](https://lists.qualcomm.com/ListManager?id=github.users&query=github.users&type=list) to get access in github.qualcomm.com

##Git Basics

<https://git-scm.com/book/en/v2/Getting-Started-Git-Basics>

URL:

<http://blog.udacity.com/2015/06/a-beginners-git-github-tutorial.html>

##

How to push from branch to master in git need to explore.

##

git clone repository.  copy the repository to the work area.

git pull repository.  sync with the repository to pull out new updates to the current copy.

##

vl-sbireddy-ice{63}> mkdir GIT\_TEST

vl-sbireddy-ice{59}> cd GIT\_TEST

vl-sbireddy-ice{60}> pwd

/usr2/sbireddy/GIT\_TEST

vl-sbireddy-ice{61}> ls

vl-sbireddy-ice{62}> git init

Initialized empty Git repository in /usr2/sbireddy/GIT\_TEST/.git/

**ADD:**

* git add . which tells Git to track all files in the project that are not yet tracked. This is the quickest and simplest way to track any new files.
* git add FILENAME explicitly tracks a single file

Ex: Add the file. Then it becomes tracked one.

vl-sbireddy-ice{90}> git add test1.pl

##Apply the command to move from staged to unstaged file and untracked file.

vl-sbireddy-ice{90}> git rm --cached test1.pl

##To Discard changes in working directory

vl-sbireddy-ice{90}> git checkout test1.pl

vl-sbireddy-ice{90}> git checkout -- test1.pl

vl-sbireddy-ice{90}> git status

##

vl-sbireddy-ice{113}> git commit -m "Initial Project Version."

[master (root-commit) 4db672c] Initial Project Version.

Committer: Subba Reddy <sbireddy@vl-sbireddy-ice.qualcomm.com>

Your name and email address were configured automatically based

on your username and hostname. Please check that they are accurate.

You can suppress this message by setting them explicitly:

git config --global user.name "Your Name"

git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

git commit --amend --reset-author

2 files changed, 18 insertions(+)

create mode 100755 test1.pl

create mode 100755 test2.pl

vl-sbireddy-ice{114}> git config --global user.name "SUBBAREDDY BIREDDY"

vl-sbireddy-ice{115}> git config --global user.email sbireddy@qti.qualcomm.com

vl-sbireddy-ice{116}> git commit --amend --reset-author

[master 866df02] Initial Project Version.

2 files changed, 18 insertions(+)

create mode 100755 test1.pl

create mode 100755 test2.pl

vl-sbireddy-ice{117}>

**CLONE:**

copy the repository from one folder to another.

vl-sbireddy-ice{122}> git clone GIT\_TEST GIT\_TEST\_1

**REMOVE:**

git rm file.txt removes the file from the repo but also deletes it from the local file system.

vl-sbireddy-ice{153}> git rm -f test2.pl

To remove the file from the repo and **not** delete it from the local file system use:  
git rm --cached file.txt

**UNSTAGE**:

# (use "git reset HEAD <file>..." to unstage)

**DISCARD:**

# (use "git checkout -- <file>..." to discard changes in working directory)

vl-sbireddy-ice{147}> git checkout test1.pl

**HISTORY:**

vl-sbireddy-ice{178}> git log --follow test1.pl

vl-sbireddy-ice{182}> git log test1.pl

vl-sbireddy-ice{183}> git log --stat test1.pl

##The below command gives code level difference.

vl-sbireddy-ice{188}> git log -p test1.pl

##If you are trying to --follow a file deleted in a previous commit use.

vl-sbireddy-ice{192}> git log --follow -- test2.pl

**RETREIVE:**

> git checkout 866df021f4b1a28dc2baeff13959b694b1b22e13 test2.pl

> git commit test2.pl -m “Retreived the original version of the file.”

**PULL**

**PUSH**

## **pushing changes**

Your changes are now in the **HEAD** of your local working copy. To send those changes to your remote repository, execute   
git push origin master  
Change *master* to whatever branch you want to push your changes to.   
  
If you have not cloned an existing repository and want to connect your repository to a remote server, you need to add it with  
git remote add origin <server>  
Now you are able to push your changes to the selected remote server

##After creating the repository add the files to the repository to keep track of the changes.

$ git add \*.c

$ git add LICENSE

$ git commit -m 'initial project version'

##Regularly Used Git Commands.

Init – initial configuration.

Let’s create a directory inside which we will be working. Alternately, you could use Git to manage one of your existing projects, in which case you would not create the demo directory as below.

Clone

Add

## To unstage

# (use "git rm --cached <file>..." to unstage)

vl-sbireddy-ice{90}> git checkout test1.pl ##To discard the changes.

Commit

Diff

Pull

Log

To check the history of your project, you can run the following command.

git log

##

git config --global user.name 'Shaumik'

git config --global user.email 'sd@gmail.com'

git config --global color.ui 'auto'

### ## Check the Status of Your Repository

* git status

**Removing Files:**

Let’s say you have added files to Git that you do not want it to track. In such a situation, you tell Git to stop tracking them. Yet, running a simple git rm will not only remove it from Git, but will also remove it from your local file system as well! To tell Git to stop tracking a file, but still keep it on your local system, run the following command:

git rm --cached [file\_name]

##

To check the history of your project, you can run the following command.

git log

##

There’s another way to access a repository, which is [cloning](http://git-scm.com/docs/git-clone). Similar to checking out a repository in other systems, running git clone <repository URL> will pull in a complete copy of the remote repository to your local system. Now you can work away on it, making changes, staging them, committing them, and pushing the changes back.

## **Putting Your Code in the Cloud**

Once you have learned how to manage your code on your system, the next step is to put it in the cloud. Since Git doesn’t have a central server like Subversion, you need to add each source to collaborate with others. That is where the concept of remotes comes in. A remote refers to a remote version of your repository.

If you wish to put your code in the cloud, you could create a project on [GitHub](https://github.com/), [GitLab](https://www.gitlab.com/), or [BitBucket](https://bitbucket.org/) and push your existing code to the repository. In this case, the remote repository in the cloud would act as a remote to your repository. Conveniently, a remote to which you have write access is called the origin.

After you create a remote repository, you have the ability to add a remote origin and then push the code to the origin.

git remote add origin https://github.com/sdaityari/my\_git\_project.git

git push -u origin master

## **Cloning a Repository**

There’s another way to access a repository, which is [cloning](http://git-scm.com/docs/git-clone). Similar to checking out a repository in other systems, running git clone <repository URL> will pull in a complete copy of the remote repository to your local system. Now you can work away on it, making changes, staging them, committing them, and pushing the changes back.https://github.com/manisaikumar/git\_manisai.git

##