GitHub Documentation by Smit Joshi

Getting & Creating Projects

- ***git init*** : Initialize a local Git repository
- ***git clone <url> :*** Create a local copy of a remote repository

Basic Snapshotting

- ***git status :*** Check status
- ***git add [file-name.txt]*** : Add a file to the staging area
- ***git add -A :*** Add all new and changed files to the staging area
- ***git commit -m "[commit message]" :*** Commit changes
- ***git rm -r [file-name.txt]*** : Remove a file (or folder)

Branching & Merging

- ***git branch :*** List branches (the asterisk denotes the current branch)
- ***git branch -a :*** List all branches (local and remote)
- ***git branch [branch name] :*** Create a new branch
- ***git branch -d [branch name] :*** Delete a branch
- ***git push origin --delete [branch name] :*** Delete a remote branch
- ***git checkout -b [branch name] :*** Create a new branch and switch to it
- ***git checkout -b [branch name] origin/[branch name] :*** Clone a remote branch and switch to it
- ***git branch -m [old branch name] [new branch name] :*** Rename a local branch
- ***git checkout [branch name] :*** Switch to a branch
- ***git checkout - :*** Switch to the branch last checked out
- ***git checkout - - [file-name.txt] :*** Discard changes to a file
- ***git merge [branch name]*** : Merge a branch into the active branch
- ***git merge [source branch] [target branch] :*** Merge a branch into a target branch
- ***git stash*** : Stash changes in a dirty working directory
- ***git stash clear :*** Remove all stashed entries

Sharing & Updating Projects

- ***git push origin [branch name] :*** Push a branch to your remote repository
- ***git push -u origin [branch name]*** : Push changes to remote repository (and remember the branch)
- ***git push*** : Push changes to remote repository (remembered branch)
- ***git push origin --delete [branch name] :*** Delete a remote branch
- ***git pull :*** Update local repository to the newest commit
- ***git pull origin [branch name] :*** Pull changes from remote repository
- ***git remote add origin ssh://git@github.com/[username]/[repository-name].git :*** Add a remote repository
- ***git remote set-url origin ssh://git@github.com/[username]/[repository-name].git :*** Set a repository's origin branch to SSH

Inspection & Comparison

- ***git log :*** View changes
- ***git log –summary :*** View changes (detailed)
- ***git log –oneline*** : View changes (briefly)
- ***git diff [source branch] [target branch] :*** Preview changes before merging

Demo

To Install Git
```
krishnab@krishnab:~$ sudo apt install git
```

To Check where the git is installed
```
krishnab@krishnab:~$ which git
/usr/bin/git
```

Creating a playground to practice the git
```
krishnab@krishnab:~$ mkdir git-demo
krishnab@krishnab:~$ cd git-demo/
krishnab@krishnab:~/git-demo$
```

*Creating the simple files with **cat > [filename.extention]***
```
krishnab@krishnab:~/git-demo$ cat > hello.txt
This is the sample file.
^Z
[3]+  Stopped                 cat > hello.txt
krishnab@krishnab:~/git-demo$ ls
hello.txt
```

Git Configurations

Changing Git Default branch name
```
krishnab@krishnab:~/git-demo$ git config --global init.defaultBranch main
```

Configure the Default user
```
krishnab@krishnab:~/git-demo$ git config --global user.email "intern-
j05@addontechnologies.net"

krishnab@krishnab:~/git-demo$ git config --global user.name "Smit Joshi"
```

Starting Git

Initializing the Git
```
krishnab@krishnab:~/git-demo$ git init
Initialized empty Git repository in /home/krishnab/git-demo/.git/
```

Adding file to stagging area
```
krishnab@krishnab:~/git-demo$ git add hello.txt
```

Adding multiple files to stagging area
```
krishnab@krishnab:~/git-demo$ cat > hello2.txt > h3llo3.txt > hello4.txt
Sample Text
^Z
[8]+  Stopped                 cat > hello2.txt > h3llo3.txt > hello4.txt

krishnab@krishnab:~/git-demo$ ls
h3llo3.txt  hello2.txt  hello4.txt  hello.txt


krishnab@krishnab:~/git-demo$ git add -A    --this will add multiple files
```

Creating Commits

```
krishnab@krishnab:~/git-demo$ git commit -m "Commit Message"
[master (root-commit) 3bddf02] Commit Message
 4 files changed, 2 insertions(+)
 create mode 100644 h3llo3.txt
 create mode 100644 hello.txt
 create mode 100644 hello2.txt
 create mode 100644 hello4.txt
```

Analyzing the modifications in the files

```
krishnab@krishnab:~/git-demo$ git diff
diff --git a/hello.txt b/hello.txt
index 7e2c8c5..a687684 100644
--- a/hello.txt
+++ b/hello.txt
@@ -1 +1 @@
-This is the sample file.
+This line is changed
```

Checking the satus of the files

```
krishnab@krishnab:~/git-demo$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Creating new branches

```
krishnab@krishnab:~/git-demo$ git branch development
```

*Checking the avilable branches (default branch will be **master or main**)*

```
krishnab@krishnab:~/git-demo$ git branch
  development
* master
```

Switching to different branch

```
krishnab@krishnab:~/git-demo$ git checkout development
Switched to branch development
```

Creating and switching to the branch at the same time

```
krishnab@krishnab:~/git-demo$ git checkout -b production
Switched to a new branch production
```

Renaming the existing branch

```
krishnab@krishnab:~/git-demo$ git branch -m main
```

Merging the changes
There are two ways to merge
1. switch to the branch you want to merge the changes with then apply ***git merge [target branch]***
2. mention both the source and the target brach. ***git merge [source-branch] [target-branch]***

```
krishnab@krishnab:~/git-demo$ git merge development
Updating c0067ee..d2bae58
Fast-forward
 development.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 development.txt
```

Pulling / Pushing the changes to the gihub.

To push the changes to the github we first need to generate and add the ssy key to the github.

Go to the ~/.ssh directory

```
krishnab@krishnab:~/git-demo$ cd ~/.ssh
```

Replace the mail & file name, you can also set the passphrase if you want.

```
krishnab@krishnab:~/.ssh$ ssh-keygen -t ed25519 -C "your-mail@mail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/krishnab/.ssh/id_ed25519): github-ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in github-ssh
Your public key has been saved in github-ssh.pub
The key fingerprint is:
SHA256:eGQ0Z7ecL7rBhJgKLkVlGh+K60jh9TaUZVb6OrCY4U4 intern-
j05@addontechnologies.net
The key&apos;s randomart image is:
+--[ED25519 256]--+
|  . +  ++.o .    |
| . B .=..+ o o   |
|..+..o .o   +    |
|.oo o  *..   .   |
|.oo. =+ S.. . .  |
|+o..=.+..o . .   |
|o..E.. o  +      |
| .o     .  o     |
|   .      .      |
+----[SHA256]-----+
```

copy the key and add it to the github add-key
give name as yourname@device-name

```
krishnab@krishnab:~/.ssh$ cat github-ssh.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIBXd5YreJ+KeevivM4KciSo7RHgB+SThVGsH0+oAuZYR
intern-j05@addontechnologies.net
```

To add the remove repository

```
krishnab@krishnab:~/git-demo$ git remote add origin "git@github.com:smit-joshi-addon/git-demo.git"
```

to see the available remotes

```
krishnab@krishnab:~/git-demo$ git remote
origin
```

To remove remote repository

```
krishnab@krishnab:~/git-demo$ git remote remove origin
```

To Pull the changes from remote repository

```
krishnab@krishnab:~/git-demo$ git pull origin main
```

** In case if you see the following error*

fatal: refusing to merge unrelated histories

try this one insted with ***--allow-unrelated-histories***

```
krishnab@krishnab:~/git-demo$ git pull origin main --allow-unrelated-histories
```

Pushing the changes to the remote repository

```
krishnab@krishnab:~/git-demo$ git push origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (13/13), 1.08 KiB | 553.00 KiB/s, done.
Total 13 (delta 3), reused 0 (delta 0)
To github.com:smit-joshi-addon/git-demo.git
   629707b..f27bd42  main -> main
```

Just a Quick advide to Resolve the Conflicts

In the case of the conficts there will be changes in the single file in the multiple braches, if this happens then you can open the desired editor and then resolve the conflicts.

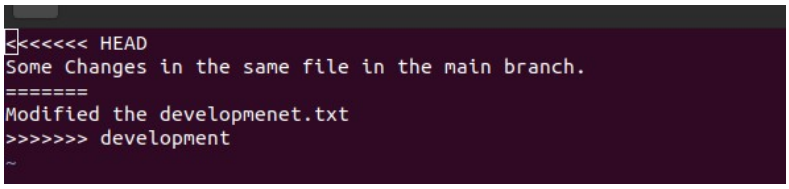
In this case i have three branches namely *“main”*, *“development”*, *“production”* from this three branches i have changes in the same file *development.txt* at both *“main”* and *“development”* branches.

Let’s try to merge these branches and see what hapens.

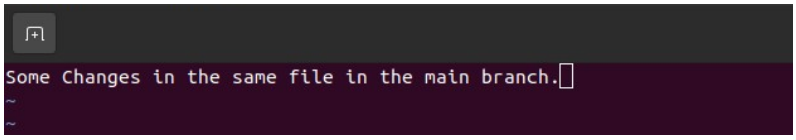
```
krishnab@krishnab:~/git-demo$ git merge main development
Auto-merging development.txt
CONFLICT (content): Merge conflict in development.txt
Automatic merge failed; fix conflicts and then commit the result.
```

As you can see there is a conflict here.
To resololve this open the conflicted file with some *IDE*, im using the *vim* here, if you don’t have *vim* installed you can use **sudo apt install vim** to install it.

```
krishnab@krishnab:~/git-demo$ vim development.txt
```



just remove the part you don’t want and then save the file.



I have removed the part that was not needed.
Now Just save the file and commit it.

Congratulations now you know the git basics.

If you can do the mentioned stuff without this documentation, you are good to go.