

Exercise
Introduction To Version Control
Name: Mahesh Inder

1. Git Setup <https://confluence.atlassian.com/bitbucket/set-up-git-744723531.htm>

A. Installing git:

- sudo apt-get update
- sudo apt-get install git

```
ttn@TTN: ~  
→ ~ sudo apt-get update  
[sudo] password for ttn:  
Hit:1 http://in.archive.ubuntu.com/ubuntu bionic InRelease  
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]  
Get:3 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]  
Hit:4 http://packages.microsoft.com/repos/vscode stable InRelease  
Get:5 http://in.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]  
Get:6 http://in.archive.ubuntu.com/ubuntu bionic-updates/main i386 Packages [446 kB]  
Get:7 http://security.ubuntu.com/ubuntu bionic-security/main amd64 DEP-11 Metadata [204  
B]  
Get:8 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [517 kB]  
Get:9 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 DEP-11 Metadata  
[20.8 kB]  
Get:10 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 DEP-11 Metadata [2  
78 kB]  
Get:11 http://security.ubuntu.com/ubuntu bionic-security/universe DEP-11 48x48 Icons [1  
2.2 kB]  
Get:12 http://security.ubuntu.com/ubuntu bionic-security/universe DEP-11 64x64 Icons [4  
5.2 kB]  
Get:13 http://in.archive.ubuntu.com/ubuntu bionic-updates/main DEP-11 48x48 Icons [66.7  
kB]  
Get:14 http://in.archive.ubuntu.com/ubuntu bionic-updates/main DEP-11 64x64 Icons [123  
kB]  
Get:15 http://in.archive.ubuntu.com/ubuntu bionic-updates/universe i386 Packages [716 k  
B]  
Get:16 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 DEP-11 Metada  
ta [2,464 B]  
Get:17 http://in.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [725  
kB]  
Get:18 http://in.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 DEP-11 Metadat  
a [202 kB]  
Get:19 http://in.archive.ubuntu.com/ubuntu bionic-updates/universe DEP-11 48x48 Icons [1  
195 kB]  
Get:20 http://in.archive.ubuntu.com/ubuntu bionic-updates/universe DEP-11 64x64 Icons [3  
343 kB]  
Get:21 http://in.archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 DEP-11 Metad  
ata [2,464 B]  
Get:22 http://in.archive.ubuntu.com/ubuntu bionic-backports/universe amd64 DEP-11 Metad  
ata [7,344 B]  
Fetched 3,955 kB in 3s (1,314 kB/s)  
Reading package lists... Done  
→ ~ sudo apt-get install git  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
git is already the newest version (1:2.17.1-1ubuntu0.4).  
0 upgraded, 0 newly installed, 0 to remove and 18 not upgraded.  
→ ~
```

```
ttn@TTN: ~  
→ ~ git --version  
git version 2.17.1  
→ ~
```

2. Initialize a Git Repository.

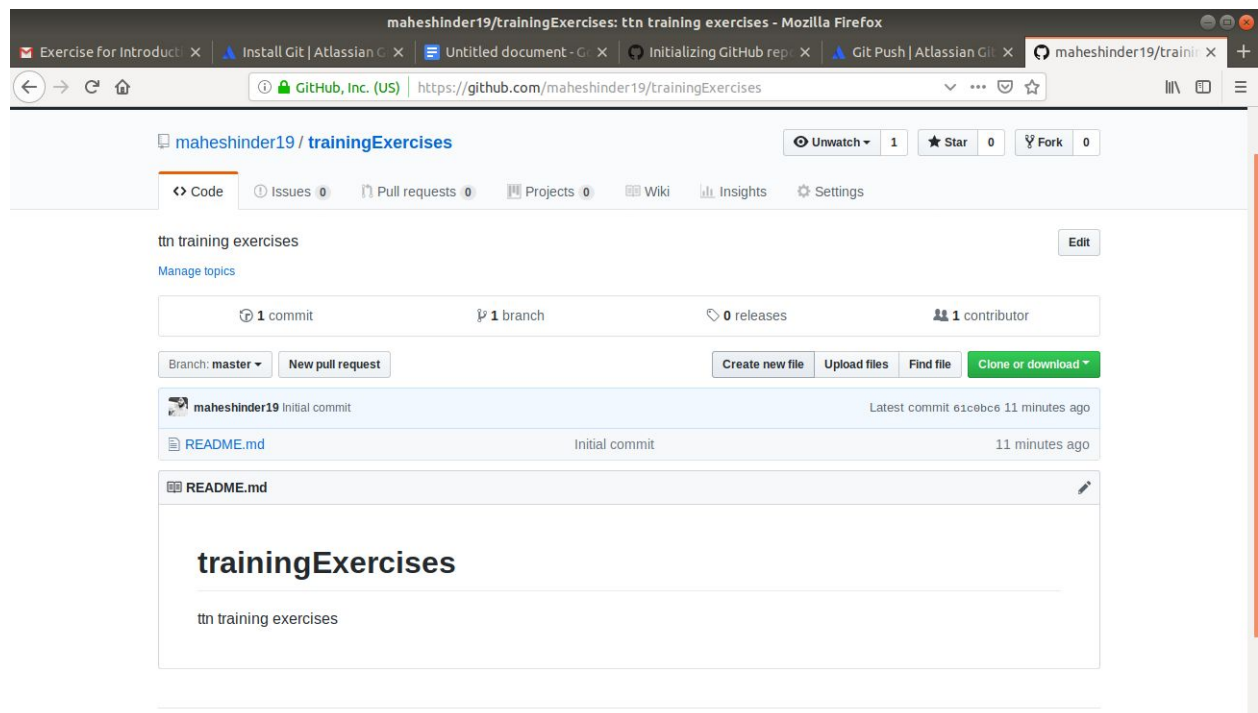
- We can initialise a git repo by using “git init” command from the terminal.

```
ttn@TTN: ~  
→ ~ git init trainingExercises  
Initialized empty Git repository in /home/ttn/trainingExercises/.git/
```

- Now by using “git remote” command, route the repository to the desired remote repository.

```
→ trainingExercises git:(master) git remote add origin https://github.com/maheshinder19/trainingExercises.git  
→ trainingExercises git:(master) █
```

- Check the online repository.

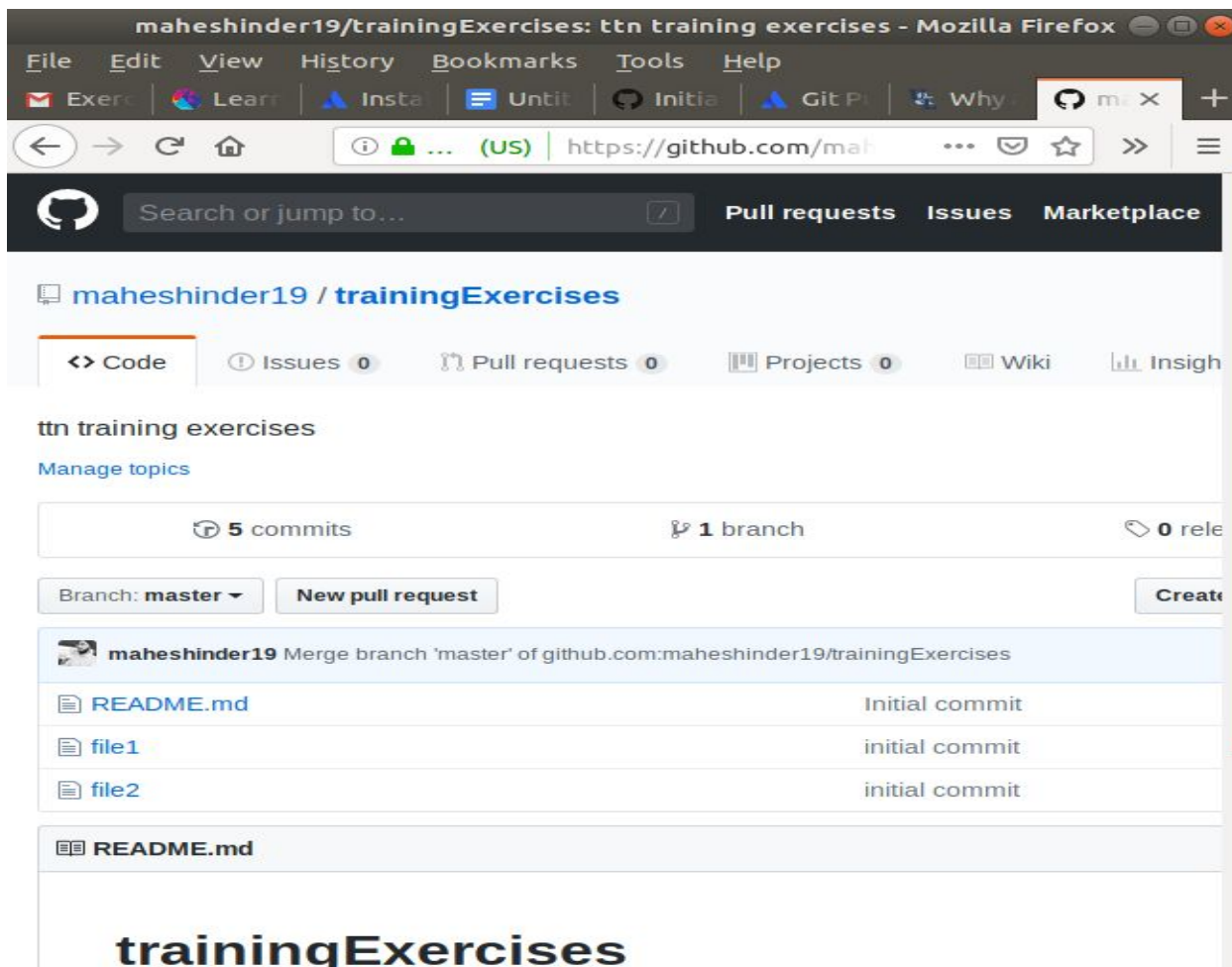


3. Add files to the repository.

```
ttn@TTN: ~/trainingExercises

→ trainingExercises git:(master) touch file1 file2
→ trainingExercises git:(master) git add file1 file2
→ trainingExercises git:(master) git commit -m "initial commit"
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
→ trainingExercises git:(master) git push
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 514 bytes | 514.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:mareshinder19/trainingExercises.git
  561bfe6..b5d950a  master -> master
→ trainingExercises git:(master) █
```



3. Unstage 1 file

- A. A file can be unstaged by using the command “git --cached fileName”

```
ttn@TTN: ~/trainingExercises

→ trainingExercises git:(master) git rm --cached file1
rm 'file1'
→ trainingExercises git:(master) x git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    deleted:    file1

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    file1

→ trainingExercises git:(master) x
```

4. Commit the file

- A. To commit a file, a file is staged first using “git add” command and then committed using “git commit”.

```
ttn@TTN: ~/trainingExercises

→ trainingExercises git:(master) git add file1
→ trainingExercises git:(master) x git commit -m "file1 commit"
[master 6bce929] file1 commit
 1 file changed, 1 insertion(+)
→ trainingExercises git:(master) x
```


5. Add a remote

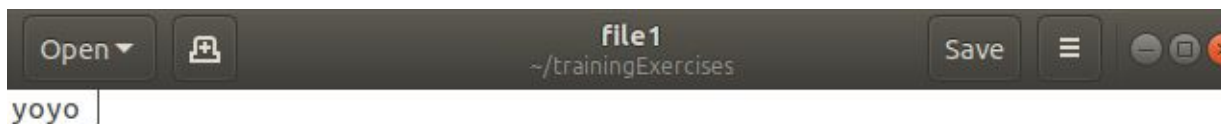
- A. We can add remote by using the “git remote add/set-url” command followed by the url to the remote repo.

```
→ trainingExercises git:(master) git remote add origin https://github.com/maheshinder19/trainingExercises.git
→ trainingExercises git:(master) █
```

6. Undo changes to a particular file.

- A. Changes in a file can be undone by using “revert” command as shown below:

Suppose we have a file named “file1” with the last commit as follows.



We need to find the commit hashcode for the above commit to undo it.

```
ttn@TTN: ~/trainingExercises
→ trainingExercises git:(master) git log --oneline
```

By looking at the log we can get the informations about the commits.

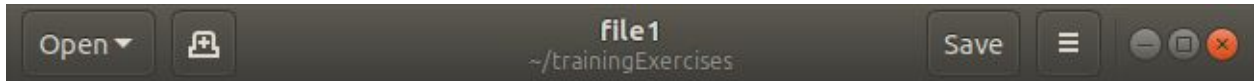
```
git log --oneline

6bce929 (HEAD -> master) file1 commit
b5d950a (origin/master, origin/HEAD) Merge branch 'master' of github.com:maheshinder19/trainingExercises
e4ad7d2 initial commit
561bfe6 Delete introToGit
5a571bb initial commit
61c0bc6 Initial commit
(END)
```

Now, to undo the changes of the commit "6bce929", we need to use "revert".

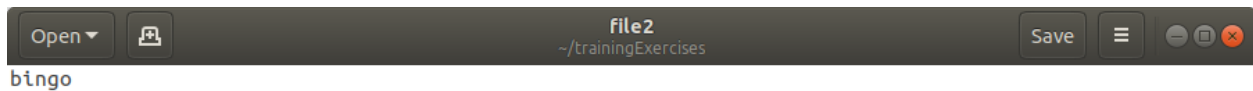
```
→ trainingExercises git:(master) git revert 6bce929
[master 26f1814] Revert "file1 commit"
1 file changed, 1 deletion(-)
→ trainingExercises git:(master) █
```

Changes get reflected on the file "file1"(text "yoyo" deleted).



7. Push changes to Github

- Making changes in "file2"



- Committing changes and pushing to Github repo.

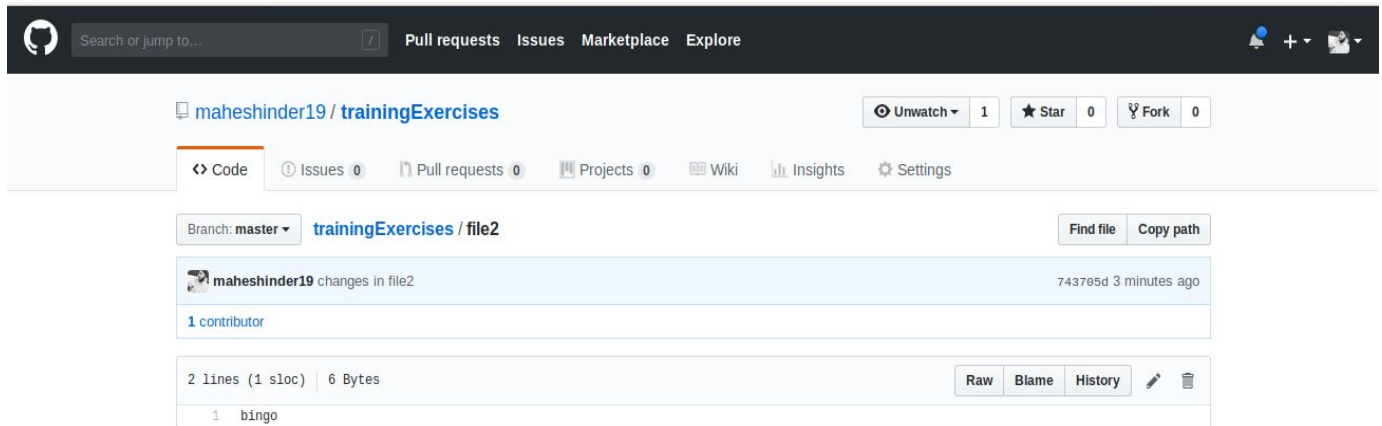
```
ttn@TTN: ~/trainingExercises

→ trainingExercises git:(master) x git add file2
→ trainingExercises git:(master) x git commit -m "changes in file2"
[master 743705d] changes in file2
2 files changed, 2 insertions(+)
→ trainingExercises git:(master) █

ttn@TTN: ~/trainingExercises

→ trainingExercises git:(master) git push origin master
Counting objects: 8, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 766 bytes | 766.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To github.com:mareshinder19/trainingExercises.git
b5d950a..743705d master -> master
→ trainingExercises git:(master) █
```

- Push successful.



8. Clone the repository

- A. One can clone any repository by using “git clone” command followed by the url of the git repo.

```
ttn@TTN: ~/Desktop
→ Desktop git clone git@github.com:mareshinder19/trainingExercises.git
Cloning into 'trainingExercises'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 19 (delta 1), reused 15 (delta 1), pack-reused 0
Receiving objects: 100% (19/19), 2.57 KiB | 2.57 MiB/s, done.
Resolving deltas: 100% (1/1), done.
→ Desktop ls
trainingExercises
→ Desktop
```

9. Add changes to one of the copies and pull the changes in the other.

- Making changes in the copy of repo in Desktop and pushing the changes.

```
ttn@TTN: ~/Desktop/trainingExercises

→ Desktop git clone git@github.com:mareshinder19/trainingExercises.git
Cloning into 'trainingExercises'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 19 (delta 1), reused 15 (delta 1), pack-reused 0
Receiving objects: 100% (19/19), 2.57 KiB | 2.57 MiB/s, done.
Resolving deltas: 100% (1/1), done.
→ Desktop ls
trainingExercises
→ Desktop training Exercises
zsh: command not found: training
→ Desktop trainingExercises
→ trainingExercises git:(master) ls
file1 file2 README.md
→ trainingExercises git:(master) touch file3
→ trainingExercises git:(master) x git add file3
→ trainingExercises git:(master) x git commit -m "file3changes"
[master bf8214d] file3changes
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file3
→ trainingExercises git:(master) git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 265 bytes | 132.00 KiB/s, done.
Total 3 (delta 1), reused 1 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:mareshinder19/trainingExercises.git
743705d..bf8214d master -> master
→ trainingExercises git:(master) ls
file1 file2 file3 README.md
→ trainingExercises git:(master) █
```

- Changing working repo to the original one and pulling the changes.

```
→ ~ trainingExercises
→ trainingExercises git:(master) ls
file1 file2 README.md
→ trainingExercises git:(master) git pull origin master
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 1), reused 3 (delta 1), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:mareshinder19/trainingExercises
* branch      master      -> FETCH_HEAD
743705d..bf8214d master    -> origin/master
Updating 743705d..bf8214d
Fast-forward
 file3 | 0
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file3
→ trainingExercises git:(master) ls
file1 file2 file3 README.md
→ trainingExercises git:(master) █
```


10. Check differences between a file and its staged version

- A. I have made some in changes in file and didn't stage it. Let's see the differences between the staged and unstaged version by using "git diff" command.

```
→ trainingExercises git:(master) git diff
diff --git a/file3 b/file3
index e69de29..f53ca2e 100644
--- a/file3
+++ b/file3
@@ -0,0 +1 @@
+hello i m an unstaged file.
(END)
```

Staging prepares a file to get committed. If a file is unstaged, no changes will reflect in it after the commit.

```
→ trainingExercises git:(master) x git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   file3

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

11. Ignore a few files to be checked in.

- A. A file named "gitignore" is used to specify the names of the files that need not to be tracked by Git.

Suppose we want file4 to be untracked.

```
ttn@TTN: ~/trainingExercises
→ trainingExercises git:(master) x touch file4
→ trainingExercises git:(master) x vi .gitignore
```

Now adding file4 to "gitignore" file.

```
vi .gitignore
*.file4
```

Checking git status.

```
→ trainingExercises git:(master) x git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   file3

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    .gitignore
    file4
```

12. Create a new branch.

A. A new branch can be created by the following command:

“git checkout -b branchName”

```
→ trainingExercises git:(master) x git checkout -b develop
M   file3
Switched to a new branch 'develop'
→ trainingExercises git:(develop) x █
```

13. Diverge them with commits.

A. Committing changes in branch “develop” (adding “file5”)

```
→ trainingExercises git:(develop) x ls
file1 file2 file3 file4 README.md
→ trainingExercises git:(develop) x touch file5
→ trainingExercises git:(develop) x git add file5
→ trainingExercises git:(develop) x git commit -m "file5 commit"
[develop 19f83b4] file5 commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file5
→ trainingExercises git:(develop) x ls
file1 file2 file3 file4 file5 README.md
```

Checking "master" branch:

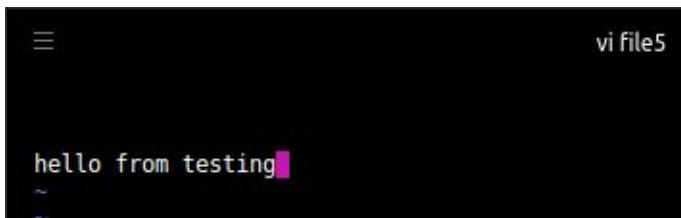
```
→ trainingExercises git:(develop) x git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
→ trainingExercises git:(master) x ls
file1 file2 file3 file4 README.md
```

As we can see both the branches differ.

14. Edit the same file at the same line on both branches and commit.

- Editing file5 in testing branch.

```
→ trainingExercises git:(testing) x vi file5
```



```
→ trainingExercises git:(testing) x git add file5
→ trainingExercises git:(testing) x git commit -m "file5 updation"
[testing 957c758] file5 updation
1 file changed, 3 insertions(+)
create mode 100644 file5
→ trainingExercises git:(testing) x git push origin testing
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 288 bytes | 288.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'testing' on GitHub by visiting:
remote:   https://github.com/maheshinder19/trainingExercises/pull/new/testing
remote:
To github.com:maheshinder19/trainingExercises.git
* [new branch]   testing -> testing
```

- Editing file5 in develop

```
→ trainingExercises git:(testing) x git checkout develop
Switched to branch 'develop'
→ trainingExercises git:(develop) x vi file5
```

```

≡ vi file5

hello from develop
~

```

```

→ trainingExercises git:(develop) x git add
Nothing specified, nothing added.
Maybe you wanted to say 'git add .'
→ trainingExercises git:(develop) x git add .
→ trainingExercises git:(develop) x git commit .
Aborting commit due to empty commit message.
→ trainingExercises git:(develop) x git commit -m "file5 updates"
[develop fb2ebfc] file5 updates
3 files changed, 8 insertions(+)
create mode 100644 .gitignore
create mode 100644 file4
→ trainingExercises git:(develop) git push origin develop
Counting objects: 10, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (10/10), 865 bytes | 288.00 KiB/s, done.
Total 10 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/maheshinder19/trainingExercises/pull/new/develop
remote:
To github.com:maheshinder19/trainingExercises.git
* [new branch]   develop -> develop

```

15. Try merging and resolve merge conflicts.

- On merging develop and testing branches we get conflicts as the file called "file5" has same text on same line.

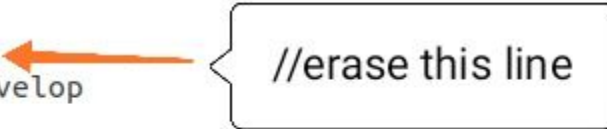
```

→ trainingExercises git:(develop) git merge testing
Auto-merging file5
CONFLICT (add/add): Merge conflict in file5
Automatic merge failed; fix conflicts and then commit the result.

```


To resolve the conflicts we need to look at the “file5” and erase the “HEAD”.

```
<<<<<< HEAD
hello from develop
=====
hello from testing
>>>>>> testing
```



Pushing the changes and check if the conflict is resolved.

```
→ trainingExercises git:(develop) ✕ git add .
→ trainingExercises git:(develop) ✕ git commit -m "solving merging conflicts"
[develop 315b3f7] solving merging conflicts
→ trainingExercises git:(develop) git merge testing
Already up to date.
```

16. Stash the changes and pop them.

- A. We can stash the changed if we do not want to commit them and do not want them to create any interruption in pushing.
- Creating a file “stashdemo”.

```
→ trainingExercises git:(develop) touch stashdemo
→ trainingExercises git:(develop) ✕ git status
On branch develop
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    stashdemo

nothing added to commit but untracked files present (use "git add" to track)
```

As we can see “stashdemo” is unstaged. Let's save it in stash.

```
→ trainingExercises git:(develop) ✕ git stash save -u
Saved working directory and index state WIP on develop: 315b3f7 solving merging conflicts
→ trainingExercises git:(develop) git status
On branch develop
nothing to commit, working tree clean
```

- Now let's pop it.

```

→ trainingExercises git:(develop) git stash pop
Already up to date!
On branch develop
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    stashdemo

nothing added to commit but untracked files present (use "git add" to track)
Dropped refs/stash@{0} (d90c9517f0e2139d372a6205cfbc91a41f095762)

```

File “stashdemo” is again unstaged for changes to be made.

17. Add the following code to your .bashrc file : color_prompt="yes"

```

parse_git_branch() {
git branch 2> /dev/null | sed -e '/^[/d' -e 's/* \(.*)/(1)/'
}
if [ "$color_prompt" = yes ]; then
PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\]
$(parse_git_branch)\[\033[00m\]\$ '
else
PS1='\u@\h:\W $(parse_git_branch)\$ '
fi
unset color_prompt force_color_prompt

```

A. Opening bashrc using vi.

```

→ ~ vi ~/.bashrc

```

Adding code.

```

parse_git_branch() {
git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
}
if [ "$color_prompt" = yes ]; then
PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]
\w\[\033[01;31m\]$(parse_git_branch)\[\033[00m\]\$ '
else
PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w$(parse_git_branch)\$ '
fi
unset color_prompt force_color_prompt
RED="\e[0;31m"
GREEN="\e[0;92m"
BLACK="\e[m"
YELLOW="\e[0;93m"
export PS1='\[\e[0;96m\]@ [\[\e[0;94m\]\u\[\e[0;96m\]\]\w\[\e[m\]\
$(echo $_git_ps1 "\["'$GREEN'\]' git:([\['$RED'\]%s\['$GREEN'\])")\ \["'$YELLOW'\]->\['$B
LACK'\]' '

# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
    *) . ~/.bashrc ;;
esac

"~/.bashrc" 158 lines, 4404 characters

```