

GIT Branches Lab

Author: Om Luitel (10th Sept, 2023)

Git provides a set of commands for working with branches, allowing you to create, switch between, manage, and merge branches in your Git repository. Here are some common Git branching commands:

1. Create a New Branch:

- `git branch <branch-name>`: Creates a new branch with the specified name but does not switch to it.
- `git checkout -b <branch-name>`: Creates a new branch with the specified name and immediately switches to it.
- `git switch -c <branch-name>`: A newer way to create and switch to a new branch in one command (Git 2.23 and later).

2. List Branches:

- `git branch`: Lists all local branches in your repository. The current branch is highlighted with an asterisk (*).

3. Switch Between Branches:

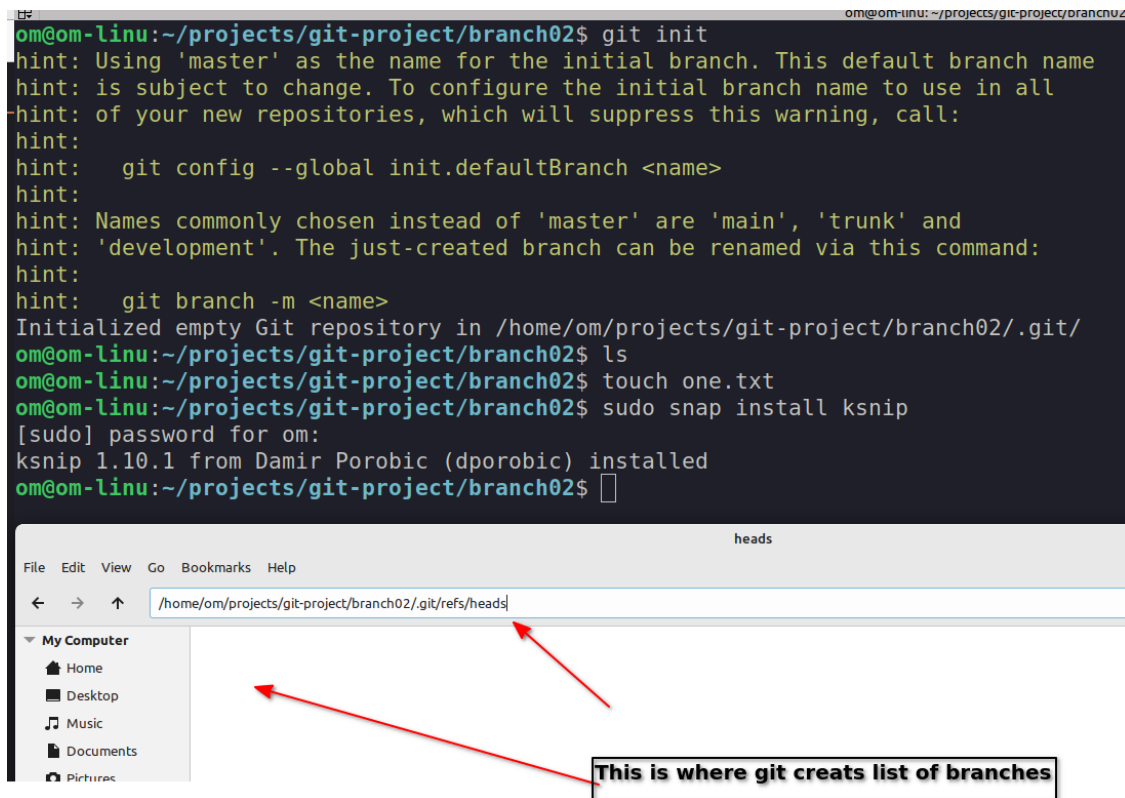
- `git checkout <branch-name>`: Switches to the specified branch.
- `git switch <branch-name>`: A more user-friendly way to switch between branches (Git 2.23 and later).

Steps:

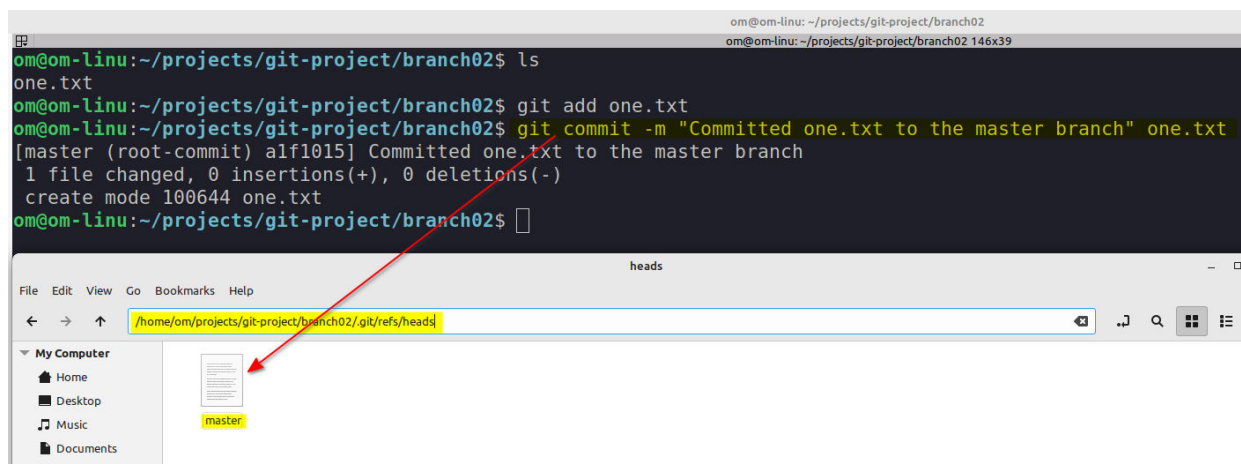
I have already created a folder named 'branch02' and executed 'git init.'

Git stores branch information under `.git/refs/heads/`.

Currently, no files are visible under the 'heads' directory because I haven't made any commits yet. We will observe what happens after the first commit in a few seconds.



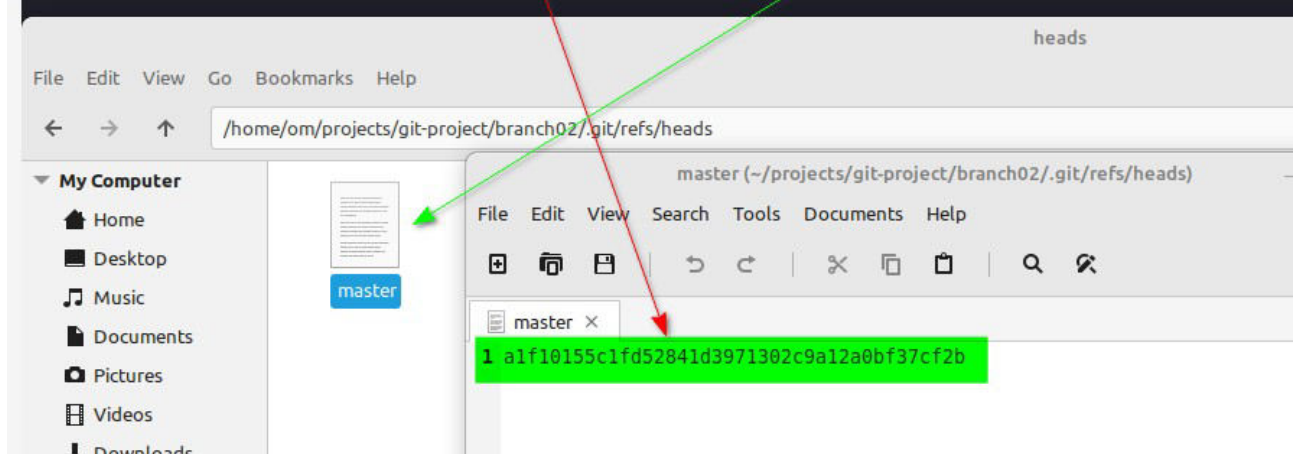
After creating a file named 'one.txt,' I've added it to the staging area and committed it with the message 'Committed one.txt file to the master branch,' as 'master' serves as the default primary branch in Git. Following the initial commit, I noticed a file named 'master' has been generated under the 'heads' directory



Now, let's examine the contents of the 'master' file, which stores the most recent commit ID. Please note that recent commit was done to master branch/default git branch.

```
om@om-linu:~/projects/git-project/branch02$ ls
one.txt
om@om-linu:~/projects/git-project/branch02$ git add one.txt
om@om-linu:~/projects/git-project/branch02$ git commit -m "Committed
[master (root-commit) alf1015] Committed one.txt to the master branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 one.txt
om@om-linu:~/projects/git-project/branch02$ git log
commit alf10155c1fd52841d3971302c9a12a0bf37cf2b (HEAD -> master)
Author: omluitel <omprakashluitelessm@gmail.com>
Date: Sun Sep 10 13:47:53 2023 +1000

    Committed one.txt to the master branch
om@om-linu:~/projects/git-project/branch02$
```



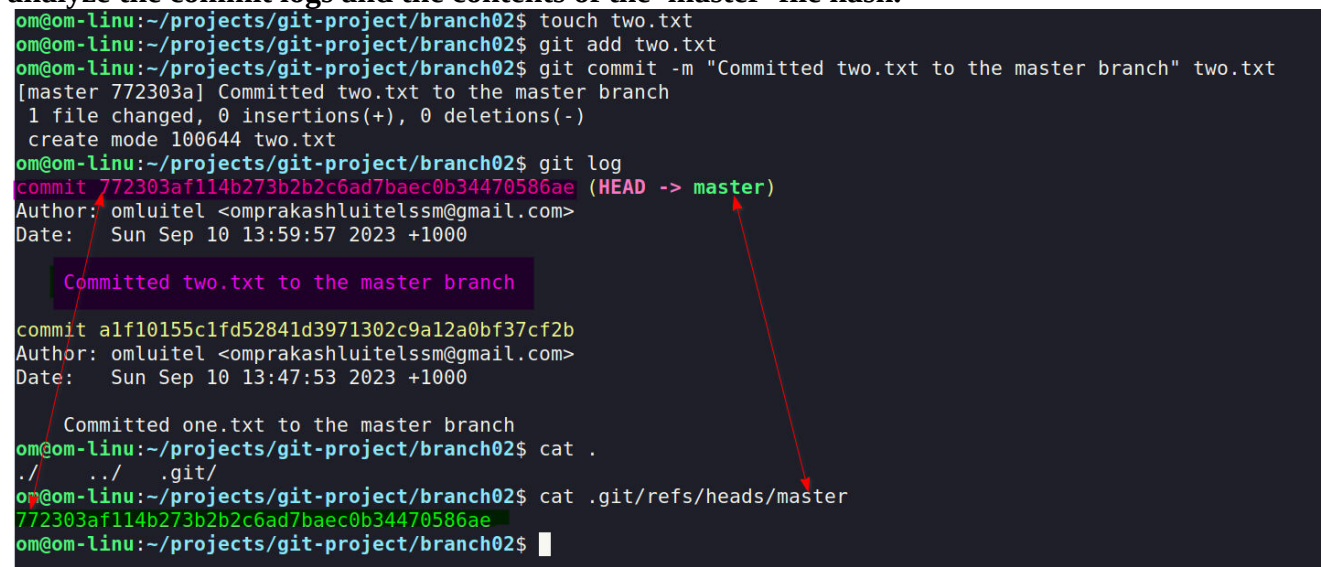
I am making another commit just to ensure that the 'master' file always references the most recent commit. I am creating a file called 'two.txt,' adding it to the staging area, and committing it with the message 'Committed two.txt to the master branch.' Afterward, we will analyze the commit logs and the contents of the 'master' file hash.

```
om@om-linu:~/projects/git-project/branch02$ touch two.txt
om@om-linu:~/projects/git-project/branch02$ git add two.txt
om@om-linu:~/projects/git-project/branch02$ git commit -m "Committed two.txt to the master branch" two.txt
[master 772303a] Committed two.txt to the master branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 two.txt
om@om-linu:~/projects/git-project/branch02$ git log
commit 772303af114b273b2b2c6ad7baec0b34470586ae (HEAD -> master)
Author: omluitel <omprakashluitelessm@gmail.com>
Date: Sun Sep 10 13:59:57 2023 +1000

    Committed two.txt to the master branch

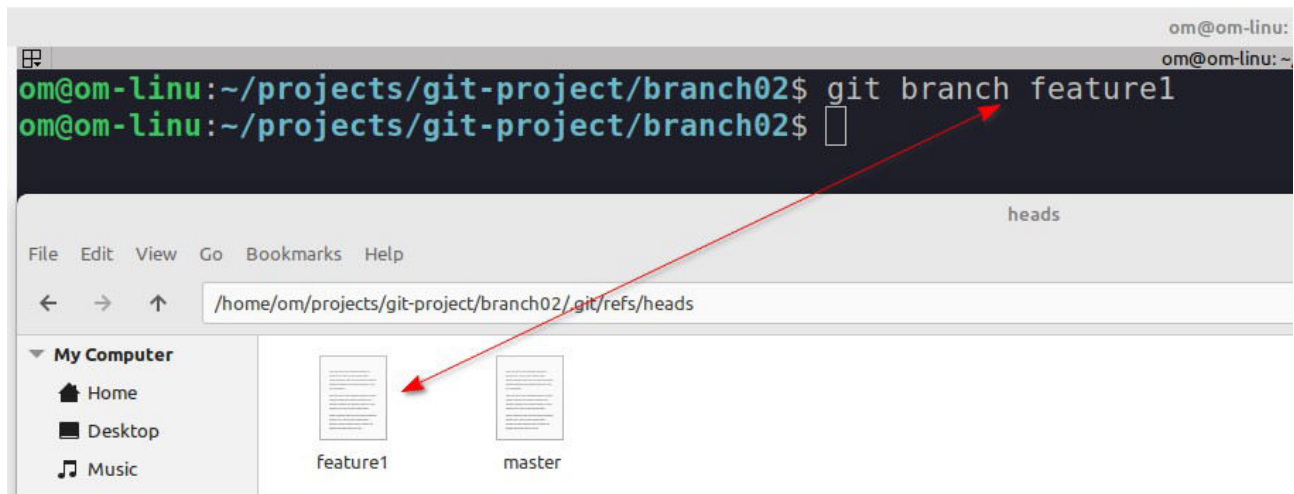
commit alf10155c1fd52841d3971302c9a12a0bf37cf2b
Author: omluitel <omprakashluitelessm@gmail.com>
Date: Sun Sep 10 13:47:53 2023 +1000

    Committed one.txt to the master branch
om@om-linu:~/projects/git-project/branch02$ cat .
./ .. .git/
om@om-linu:~/projects/git-project/branch02$ cat .git/refs/heads/master
772303af114b273b2b2c6ad7baec0b34470586ae
om@om-linu:~/projects/git-project/branch02$
```

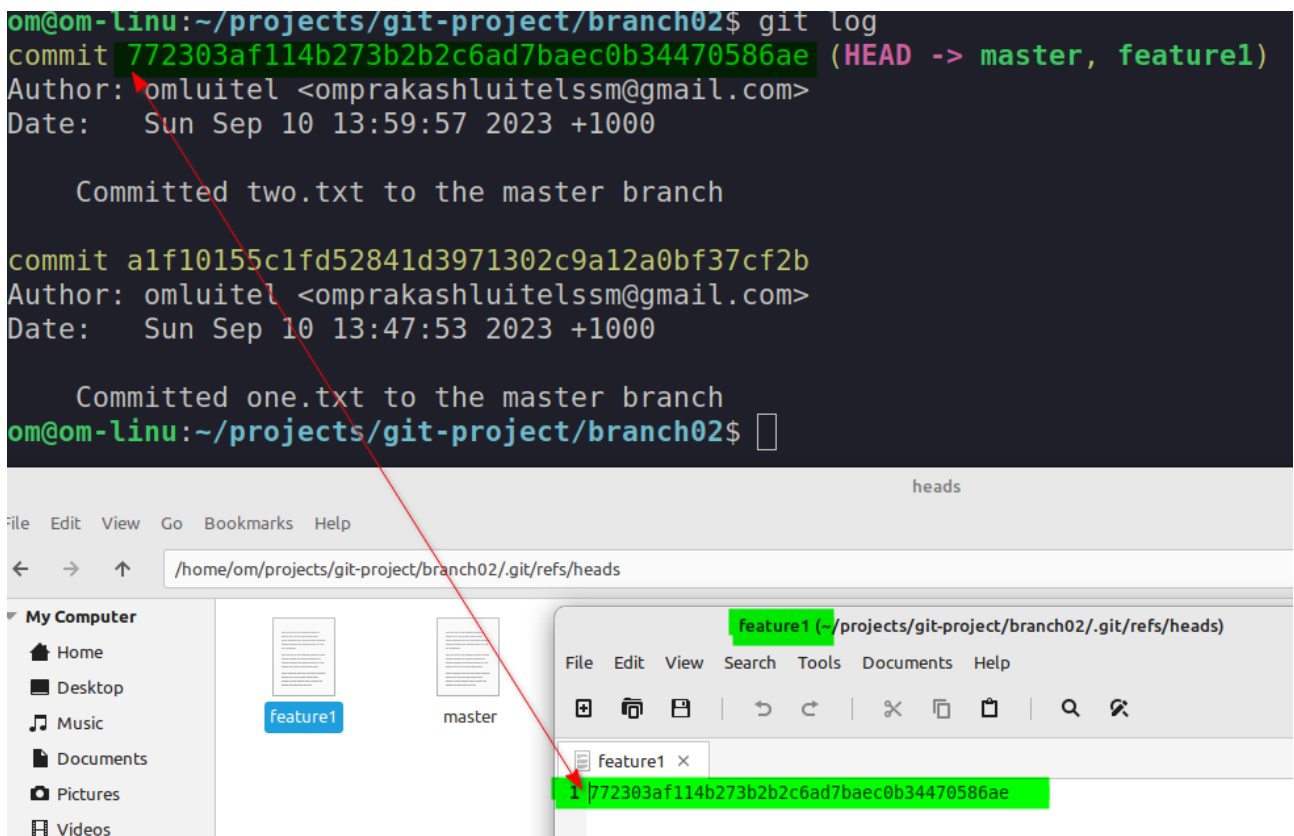


This confirms master branch is pointing to recent commit **772303af114b273b2b2c6ad***

Now, I am creating a branch called 'feature1' and will analyze what happens inside the 'heads' directory.



As soon as I created the 'feature1' branch, I noticed a file named 'feature1' created inside the 'heads' directory. Now, let's analyze the content inside it.



I can confirm feature1 is pointing to last commit of master branch.

Question: What will happen if I make a new commit? Will it go to the master or feature branch?

Answer: It will go to the master branch because the current active branch is master, and we haven't switched/checked out to feature yet. To determine which branch is active, you can run the `git branch` command.

```
om@om-linu:~/projects/git-project/branch02$ git branch
feature1
* master
om@om-linu:~/projects/git-project/branch02$
```

Let's create a file called 'three.txt' and commit it to the master branch with the commit message 'Committed three.txt to the master branch.' Once that is done, we will use 'ls' to see the list of visible files. Afterward, we will switch to the feature branch and examine the list of files we can see. This will confirm how branching works.

```
om@om-linu:~/projects/git-project/branch02$ touch three.txt
om@om-linu:~/projects/git-project/branch02$ git add three.txt
om@om-linu:~/projects/git-project/branch02$ git commit -m " Committed three.txt to the master branch " three.txt
[master feb9c14] Committed three.txt to the master branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 three.txt
om@om-linu:~/projects/git-project/branch02$ tree
.
├── one.txt
├── three.txt
└── two.txt

0 directories, 3 files
om@om-linu:~/projects/git-project/branch02$ git switch feature1
Switched to branch 'feature1'
om@om-linu:~/projects/git-project/branch02$ tree
.
├── one.txt
└── two.txt

0 directories, 2 files
om@om-linu:~/projects/git-project/branch02$
```

In the master branch, we observe a list of three files, as the most recent commit was pushed to the master branch. However, on the feature1 branch, we can only see two files. If we perform a new commit on the feature1 branch and then switch back to the master branch and list the files, we will not be able to see the file committed on the feature1 branch. This behavior will also apply if we switch from the master branch to another branch.


```
om@om-linu:~/projects/git-project/branch02$ git switch feature1
Switched to branch 'feature1'
om@om-linu:~/projects/git-project/branch02$ tree
.
├── one.txt
└── two.txt

0 directories, 2 files
om@om-linu:~/projects/git-project/branch02$ git branch
* feature1
  master
om@om-linu:~/projects/git-project/branch02$ touch featurefile01.txt
om@om-linu:~/projects/git-project/branch02$ git add featurefile01.txt
om@om-linu:~/projects/git-project/branch02$ git commit -m "This commit is for
[feature1 a30518e] This commit is for feature1 branch featurefile01.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 featurefile01.txt
om@om-linu:~/projects/git-project/branch02$ tree
.
├── featurefile01.txt
├── one.txt
└── two.txt

0 directories, 3 files
om@om-linu:~/projects/git-project/branch02$ git switch master
Switched to branch 'master'
om@om-linu:~/projects/git-project/branch02$ tree
.
├── one.txt
├── three.txt
└── two.txt

0 directories, 3 files
om@om-linu:~/projects/git-project/branch02$
```