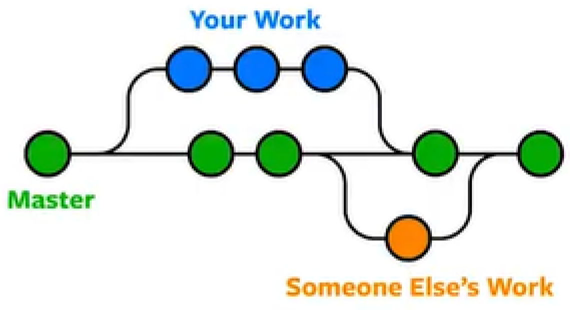
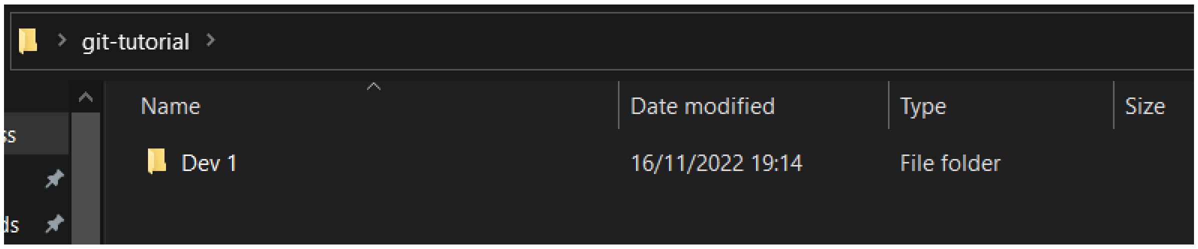
git - tutorial



1. Set directory :

   set the cmd to the folder we want to collaborate/work with git

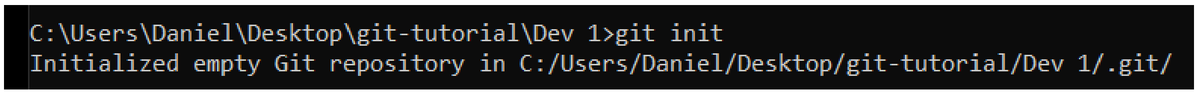
   example : Desktop/git-tutorial/Dev 1/



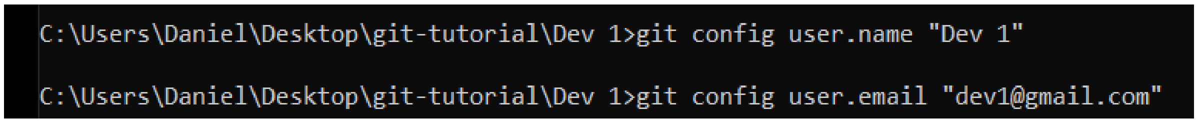
Name	Date modified	Type	Size
Dev 1	16/11/2022 19:14	File folder	

2. initiate git

   `git init` → `git config` `user.name` `'name'` → `git config` `user.email` `'email'`

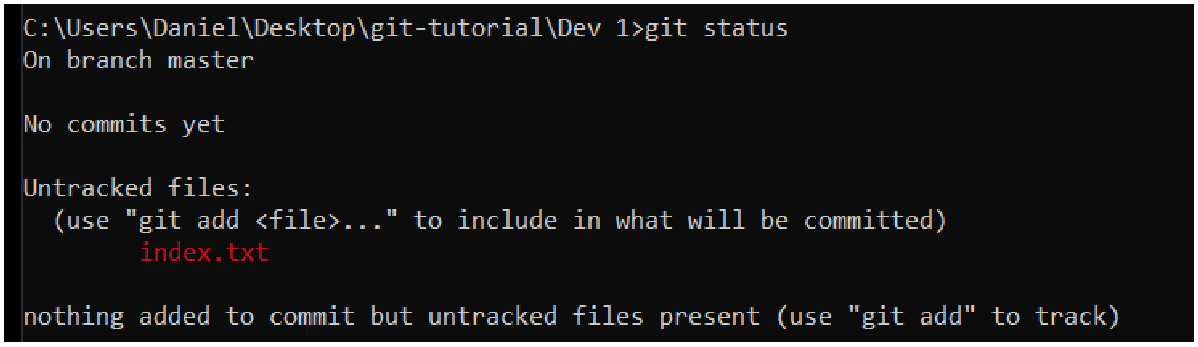


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git init
Initialized empty Git repository in C:/Users/Daniel/Desktop/git-tutorial/Dev 1/.git/
```



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git config user.name "Dev 1"

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git config user.email "dev1@gmail.com"
```

to check your git status use `git status` , and the output is :



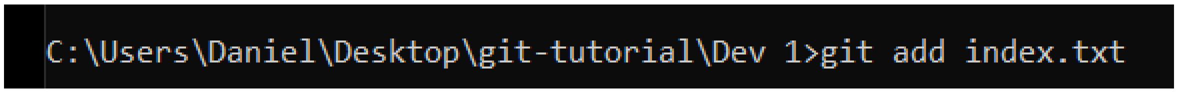
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master

No commits yet

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

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

3. track the file and add it

💡 by using add, we **staging** it not creating new version yet!

coz it said there is untracked file → red color (happened because we create the file before initiate git), so we can add the file to our git

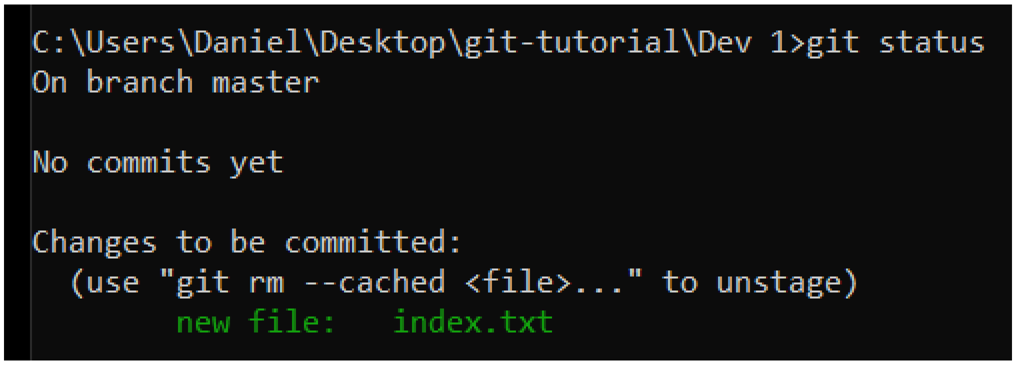
by using this command : `git add index.txt` / `git add .`



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git add index.txt
```

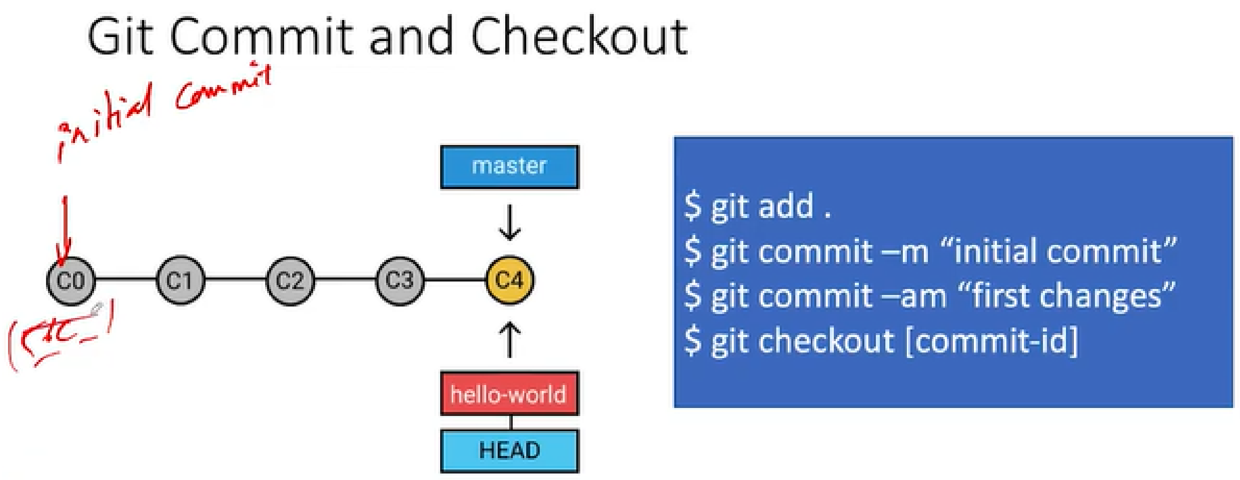
- if only . in your git add command line, it means you want to add all the files on that particular folder

and if you check the status again



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.txt
```

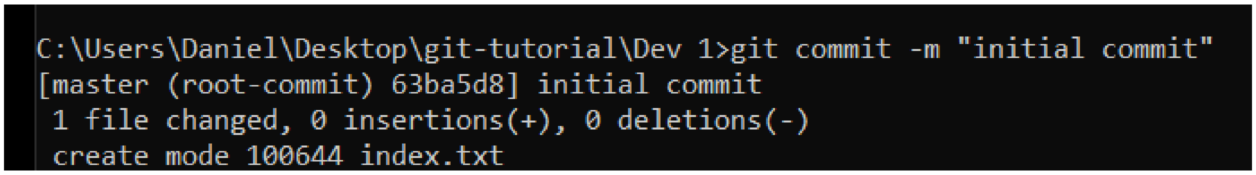
4. commit



Git Commit and Checkout

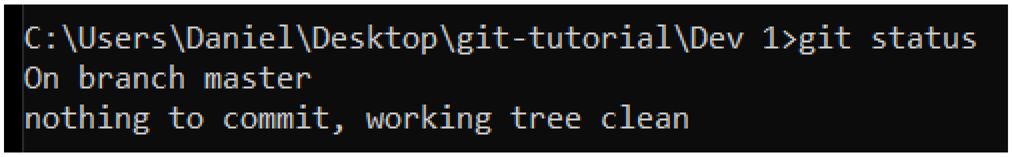
```
$ git add .
$ git commit –m “initial commit”
$ git commit –am “first changes”
$ git checkout [commit-id]
```

-m mean message, so you can put a note on it

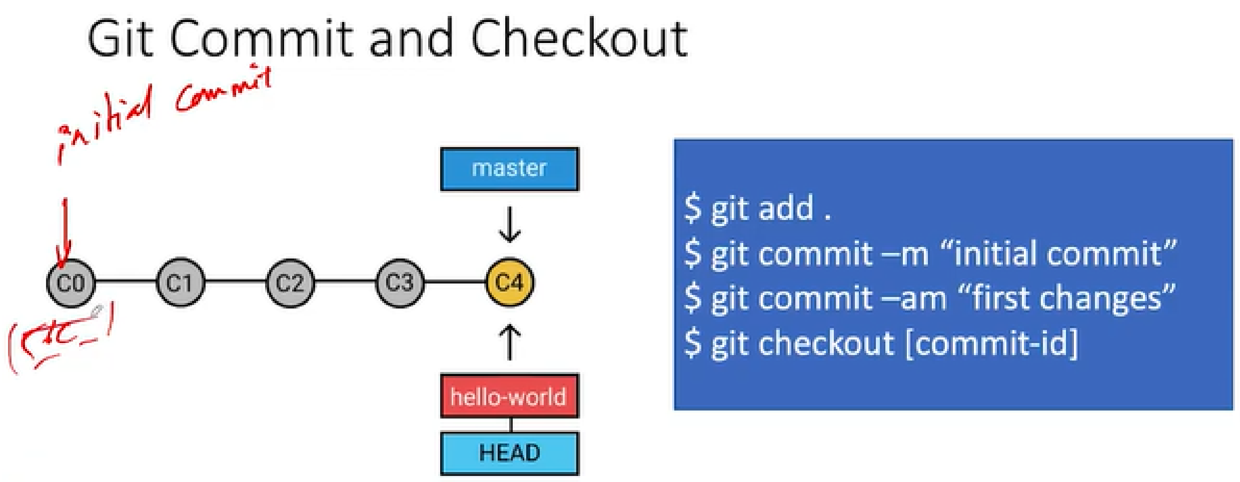


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -m "initial commit"
[master (root-commit) 63ba5d8] initial commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.txt
```

the file that on the staging phase by using add command, is already commited now.



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master
nothing to commit, working tree clean
```

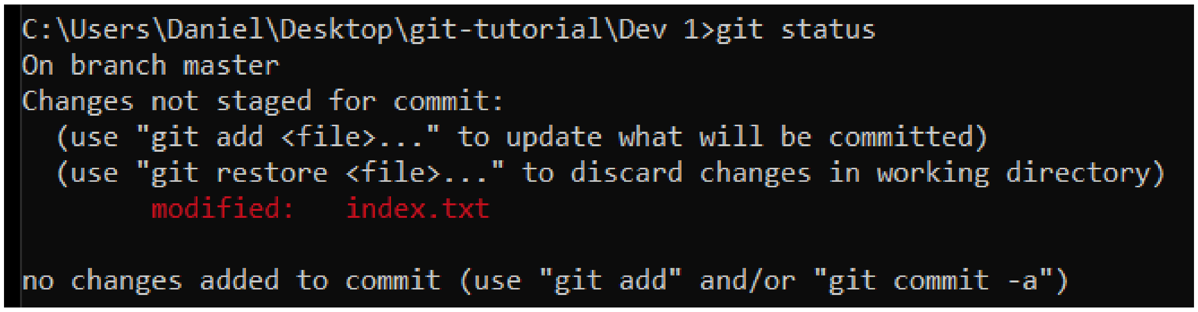


Git Commit and Checkout

```
$ git add .
$ git commit –m “initial commit”
$ git commit –am “first changes”
$ git checkout [commit-id]
```

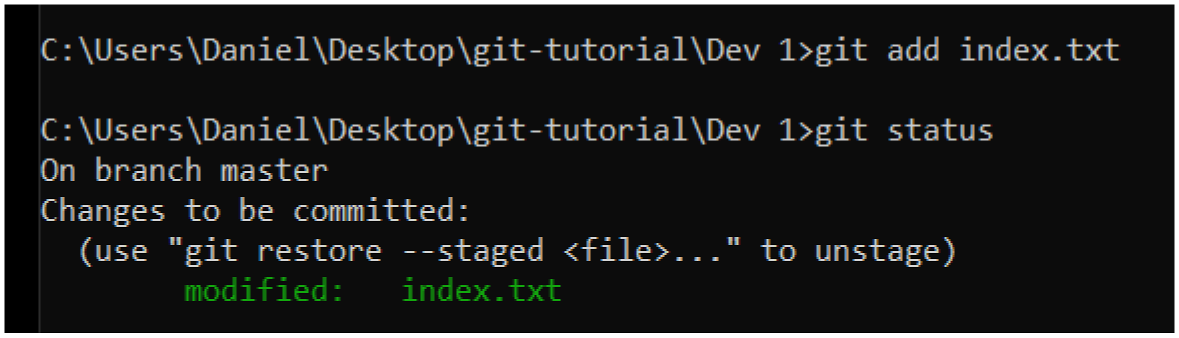
5. Moving between versions

   when i change the file, then check the git status again :



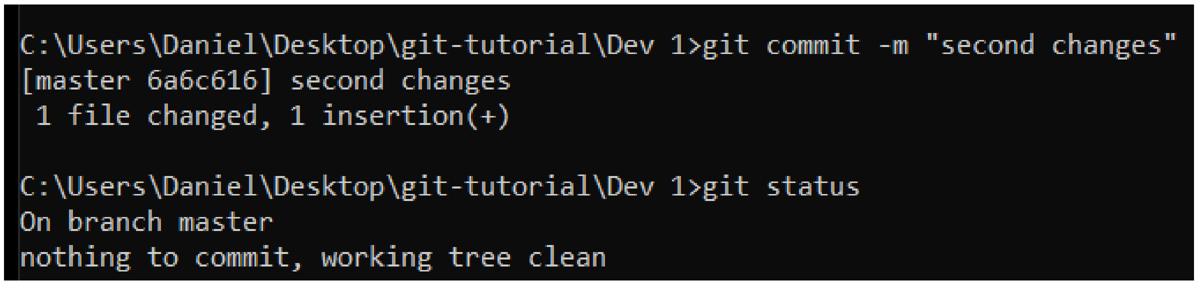
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.txt

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

so we can simply put the newest version of the file into staging area by using add command:

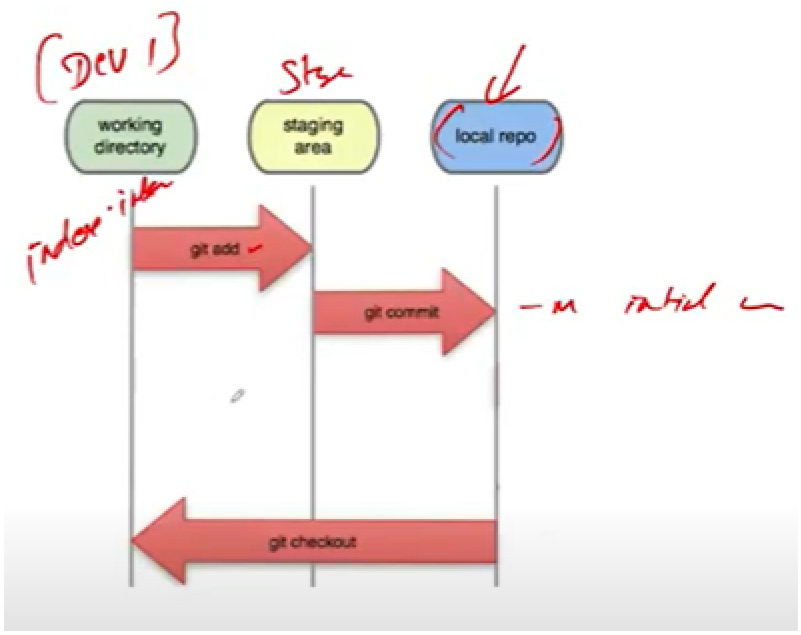


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git add index.txt

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.txt
```

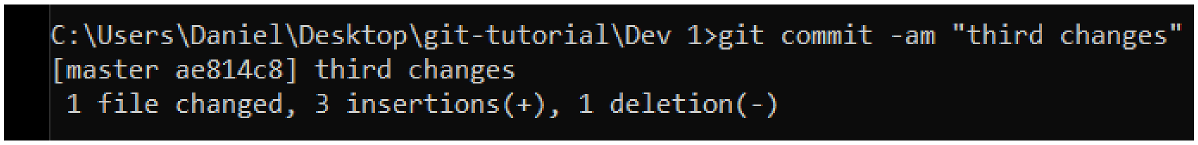
then commit it!



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -m "second changes"
[master 6a6c616] second changes
 1 file changed, 1 insertion(+)

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch master
nothing to commit, working tree clean
```



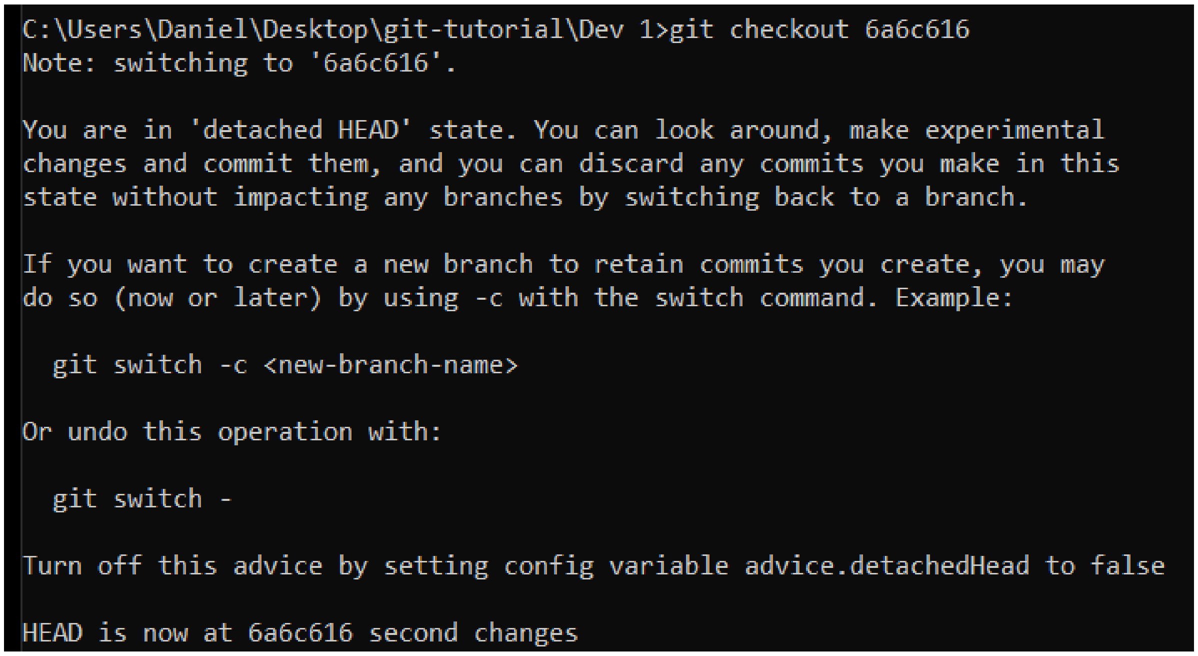
-am let you do add and commit in the same line



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -am "third changes"
[master ae814c8] third changes
 1 file changed, 3 insertions(+), 1 deletion(-)
```

how to go back to another version (not the latest) → `git checkout`

we need the commit id

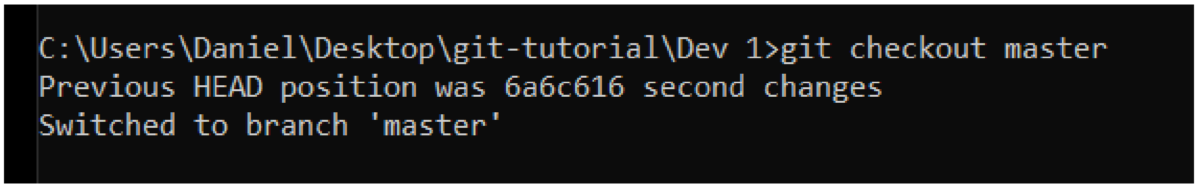


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git checkout 6a6c616
Note: switching to '6a6c616'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 6a6c616 second changes
```

and magically when you open your file again, it back to the second one, not the latest!

and if you want to go back again to the latest version you can use the same command plus the commit id **OR** you can use

`git checkout master`

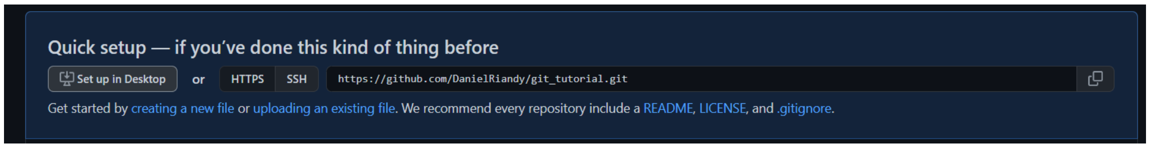


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git checkout master
Previous HEAD position was 6a6c616 second changes
Switched to branch 'master'
```

7. Working remotely with Github

- Connect with remote repository

  first, you can create repository on your github and get the directory for github repo

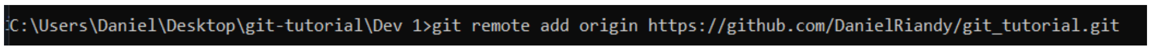


Quick setup — if you've done this kind of thing before

https://github.com/DanielRiandy/git_tutorial.git

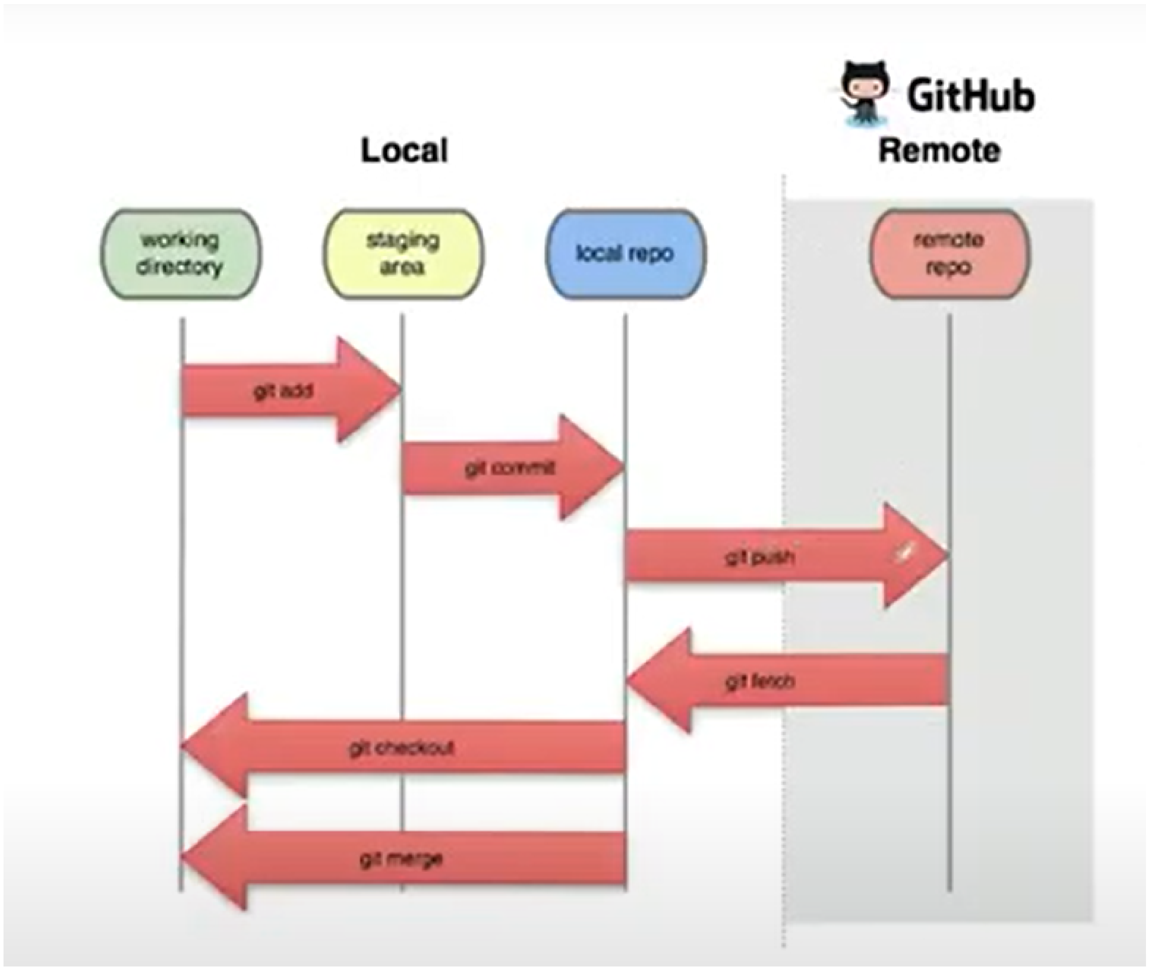
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

and we can use this command (`git remote add origin [link]`)



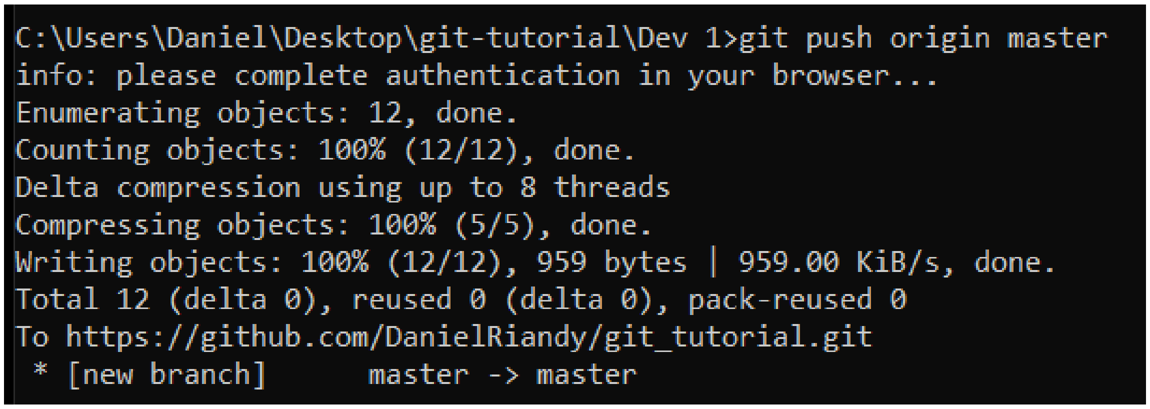
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git remote add origin https://github.com/DanielRiandy/git_tutorial.git
```

- Push the file already create locally the remote github Repo



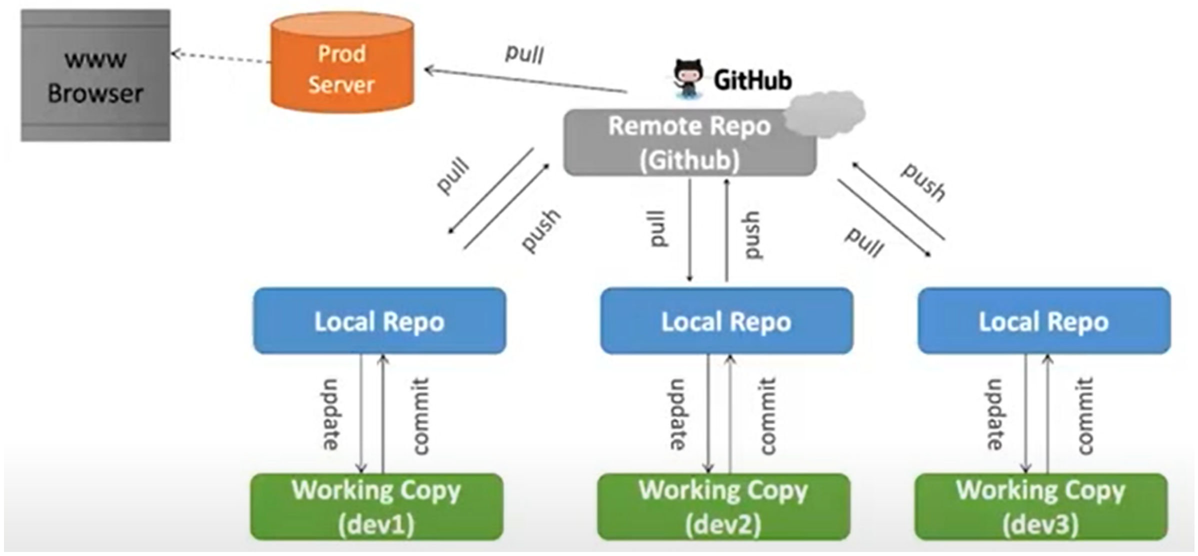
and use this command to push the master branch to the remote repo

`git push origin master`



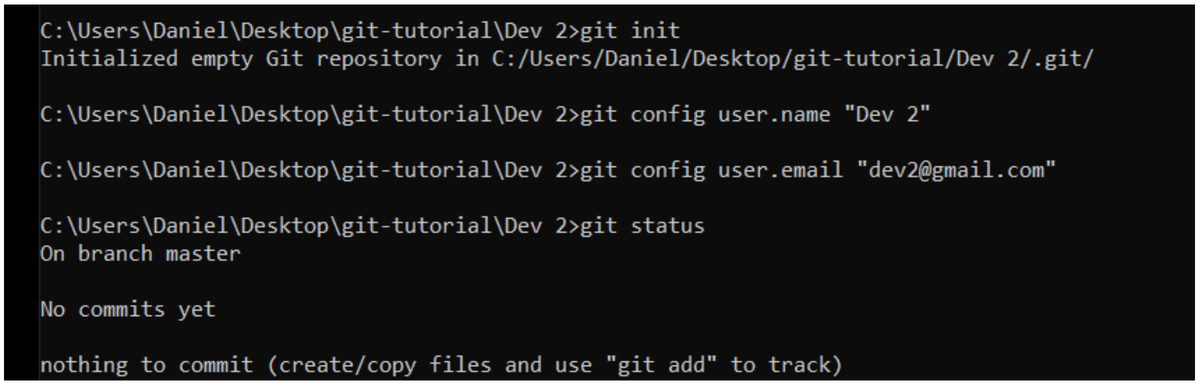
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git push origin master
info: please complete authentication in your browser...
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (12/12), 959 bytes | 959.00 KiB/s, done.
Total 12 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/DanielRiandy/git_tutorial.git
 * [new branch]      master -> master
```

7. Team collaboration with git



first we define another person, we named it “Dev 2”, which this person got zero file in his folder. For this case, Dev 2 want to start working with file that Dev 1 created, that is what collaboration should be.

then, we have to initiate the git, config the name and email



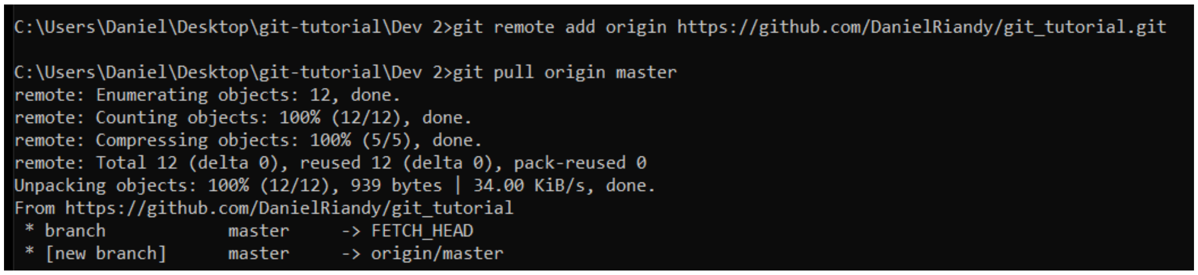
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git init
Initialized empty Git repository in C:/Users/Daniel/Desktop/git-tutorial/Dev 2/.git/

C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git config user.name "Dev 2"

C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git config user.email "dev2@gmail.com"

C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

after that, we can connect to remote repo by the same link with the previous one Dev 1 has already accessed, and pull the file

using these commands :

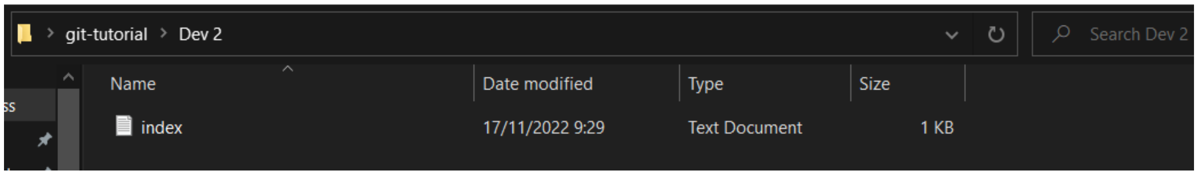
`git remote add origin [repo link]`

`git pull origin master`



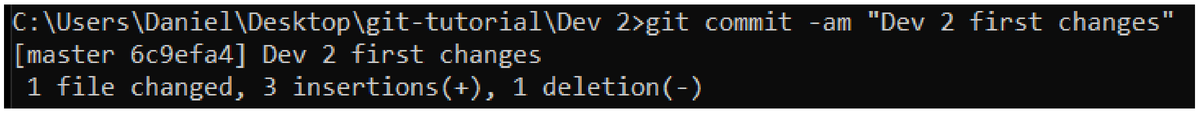
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git remote add origin https://github.com/DanielRiandy/git_tutorial.git

C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git pull origin master
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 12 (delta 0), reused 12 (delta 0), pack-reused 0
Unpacking objects: 100% (12/12), 939 bytes | 34.00 KiB/s, done.
From https://github.com/DanielRiandy/git_tutorial
 * branch            master     -> FETCH_HEAD
 * [new branch]      master     -> origin/master
```

and we can check Dev 2 local, the file that done by Dev 1 is already there



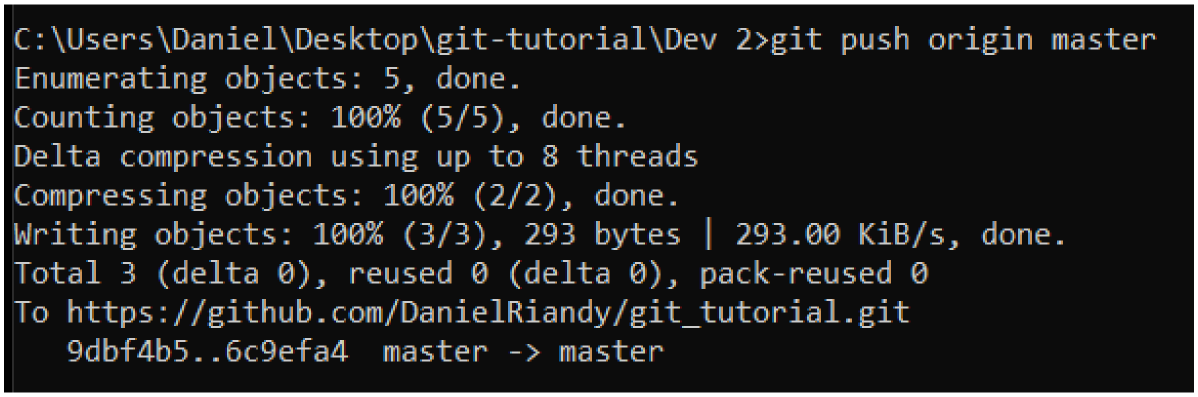
Name	Date modified	Type	Size
index	17/11/2022 9:29	Text Document	1 KB

now Dev 2 already finish the work on the same file and want to push it into remote repo, first Dev to has to commit the file to master branch by using commit command



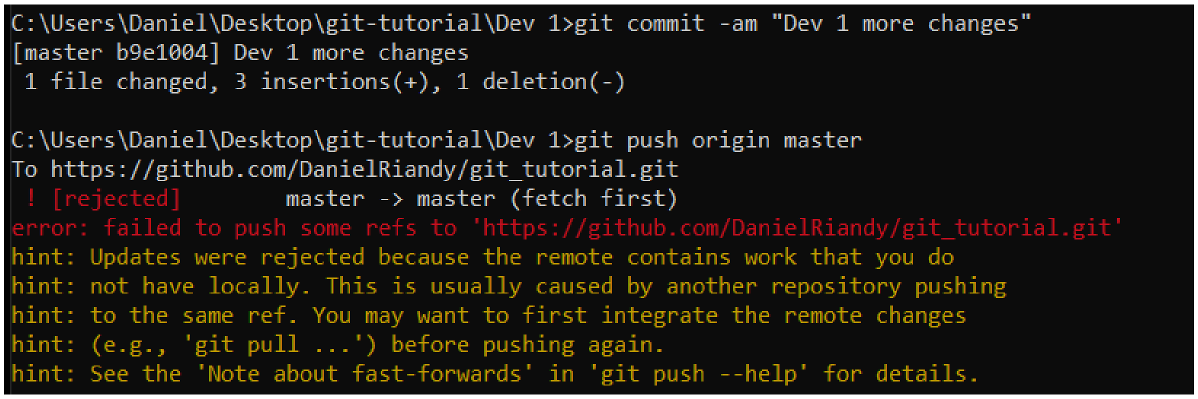
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git commit -am "Dev 2 first changes"
[master 6c9efa4] Dev 2 first changes
 1 file changed, 3 insertions(+), 1 deletion(-)
```

then push it to remote repo by using push origin master command



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 2>git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/DanielRiandy/git_tutorial.git
   9dbf4b5..6c9efa4  master -> master
```

Okay, now let say Dev 1 have another changes on his code, and want to push it to remote before pull the latest modified one by Dev 2

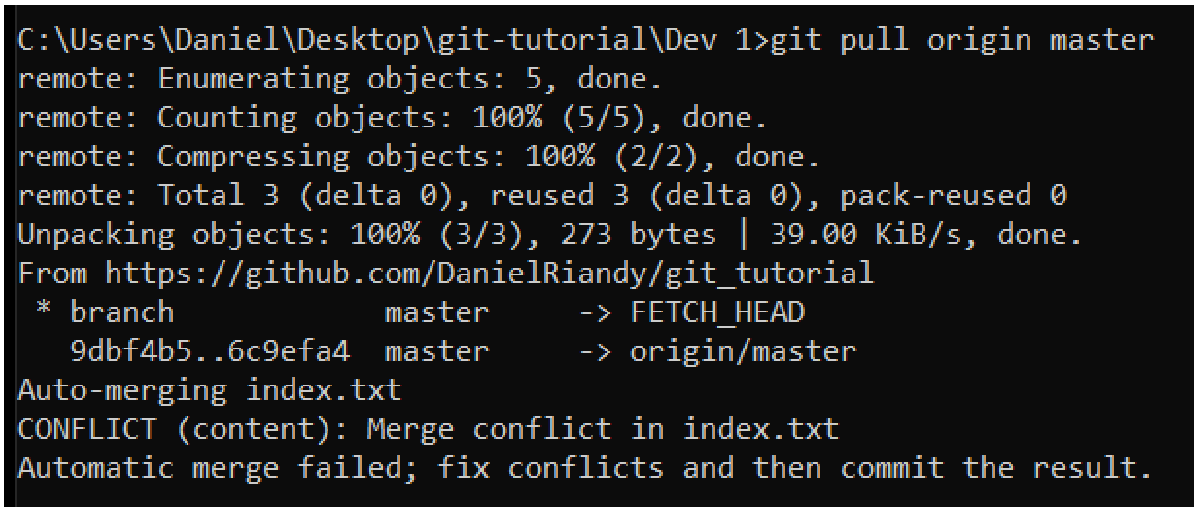


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -am "Dev 1 more changes"
[master b9e1004] Dev 1 more changes
 1 file changed, 3 insertions(+), 1 deletion(-)

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git push origin master
To https://github.com/DanielRiandy/git_tutorial.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/DanielRiandy/git_tutorial.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

REJECTED!! because, there is new modified file by another user, so you have to pull it first, and then modify the code begin from that new modified file by other user

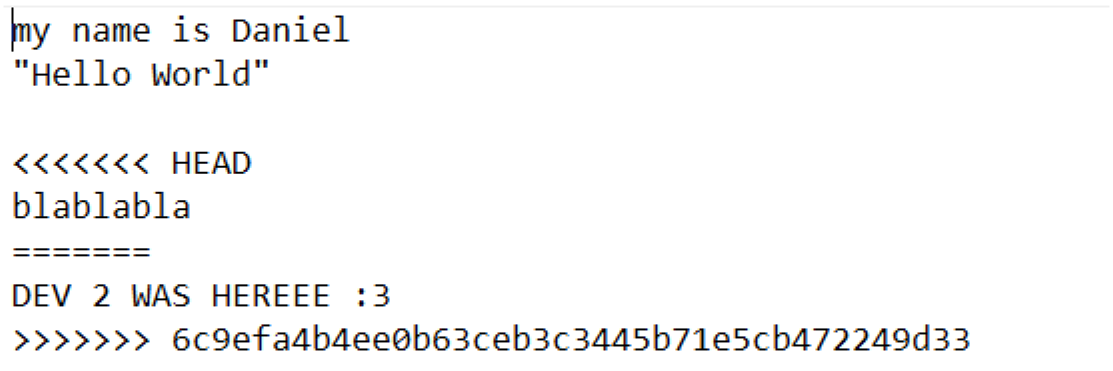
💡 Make sure before you modify your file/code, the file is already the latest version!

so, we have to pull the latest version done by another user first



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 273 bytes | 39.00 KiB/s, done.
From https://github.com/DanielRiandy/git_tutorial
 * branch            master     -> FETCH_HEAD
   9dbf4b5..6c9efa4  master     -> origin/master
Auto-merging index.txt
CONFLICT (content): Merge conflict in index.txt
Automatic merge failed; fix conflicts and then commit the result.
```

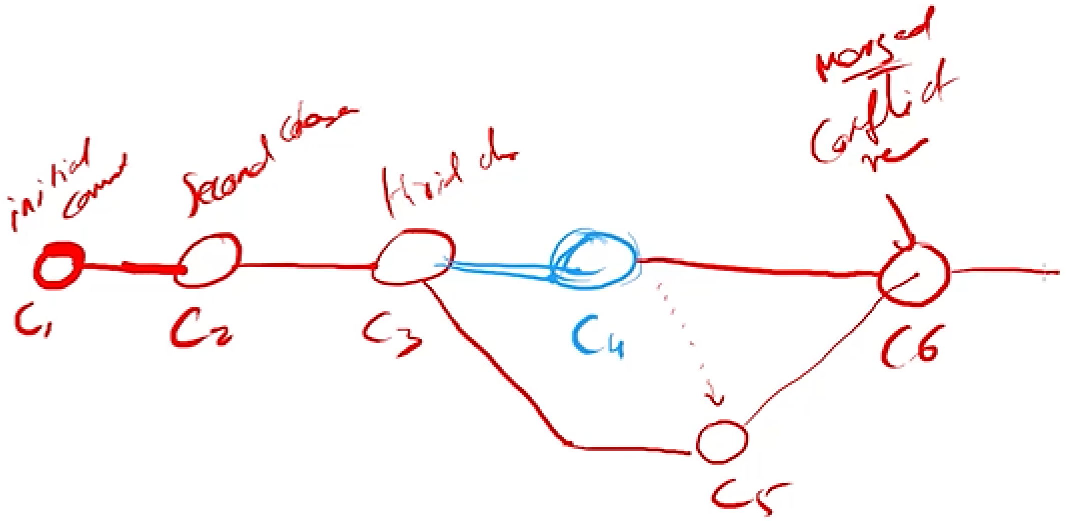
and the file will automatically merged

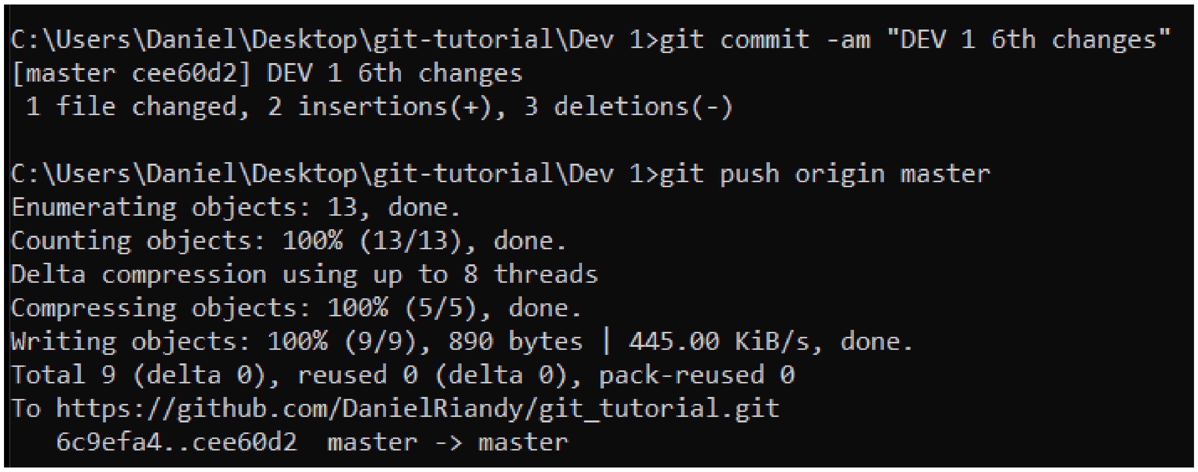


```
my name is Daniel
"Hello World"

<<<<<<< HEAD
blablabla
=======
DEV 2 WAS HEREEE :3
>>>>>>> 6c9efa4b4ee0b63ceb3c3445b71e5cb472249d33
```

blablabla is done by Dev 1 locally, and DEV 2 WAS HEREE :3 is the latest version from Dev 2

the merge process is not good enough, so we have to edit it manually on Dev 1’s local, then commit the changes, and pull it again to remote repo

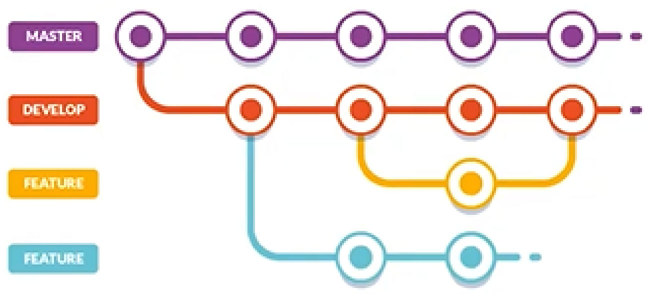


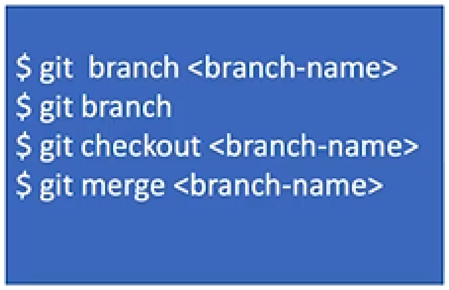


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -am "DEV 1 6th changes"
[master cee60d2] DEV 1 6th changes
 1 file changed, 2 insertions(+), 3 deletions(-)

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git push origin master
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (9/9), 890 bytes | 445.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/DanielRiandy/git_tutorial.git
   6c9efa4..cee60d2  master -> master
```

8. Create Branches

Git Branches



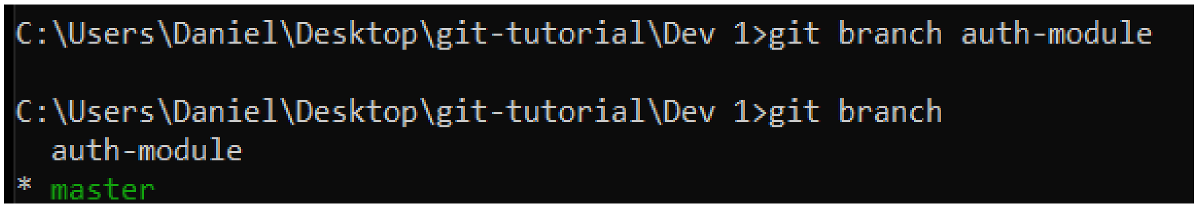


```
$ git  branch <branch-name>
$ git branch
$ git checkout <branch-name>
$ git merge <branch-name>
```

this is separate feature in the same project, in case you want to work on the same file but don’t want to getting disturbed by the changes on master branch

using this command

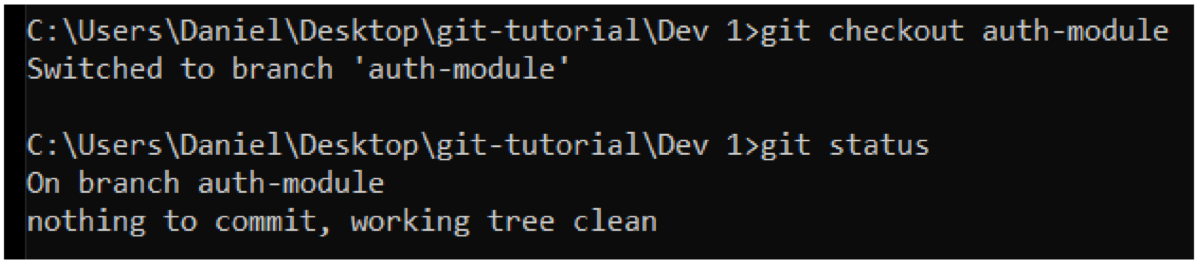
`git branch [branch name]`



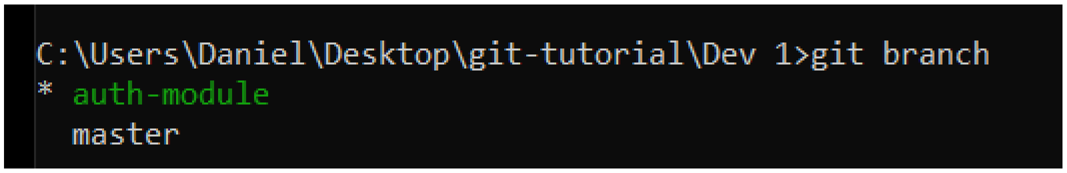
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git branch auth-module

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git branch
  auth-module
* master
```

as you can see, now we have 2 branch, Master and auth-module

now we switch the branch to another branch by using checkout

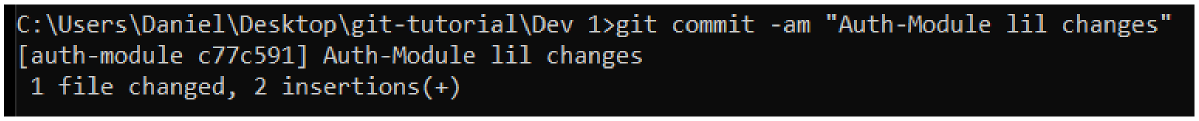


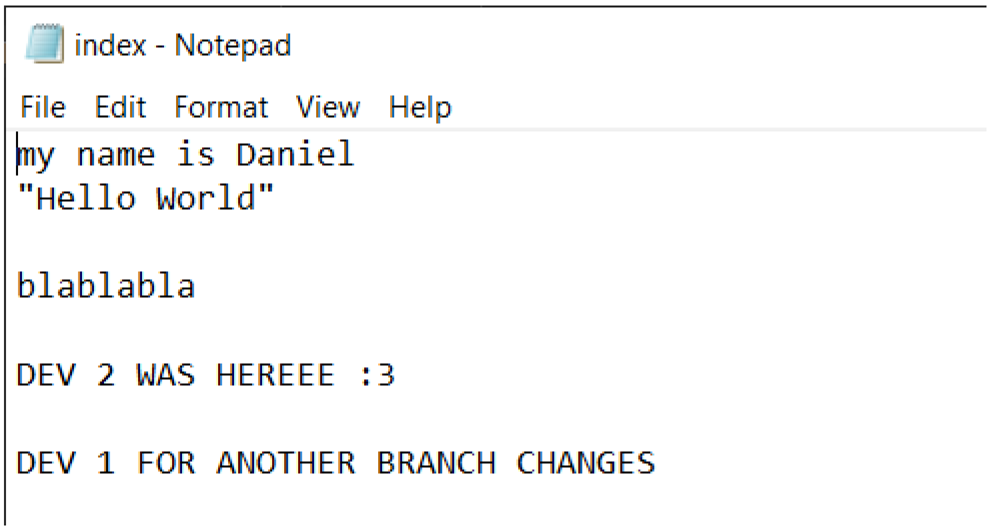
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git checkout auth-module
Switched to branch 'auth-module'

C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git status
On branch auth-module
nothing to commit, working tree clean
```


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git branch
* auth-module
  master
```

now this another developer want to modify the code, and merge it to master branch

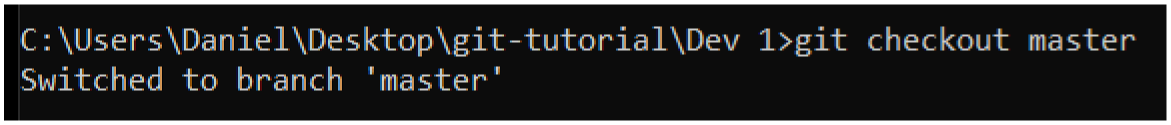
this can be done by commit the changes first

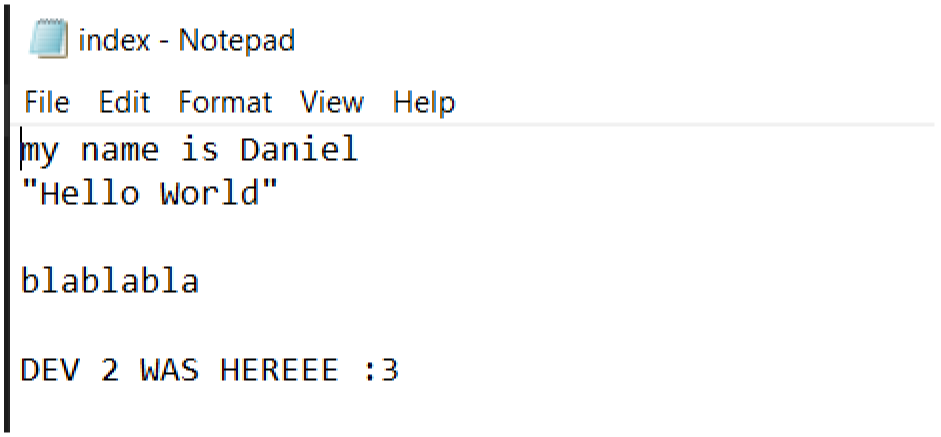

```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git commit -am "Auth-Module lil changes"
[auth-module c77c591] Auth-Module lil changes
 1 file changed, 2 insertions(+)
```



```
my name is Daniel
"Hello World"

blablabla

DEV 2 WAS HEREEE :3

DEV 1 FOR ANOTHER BRANCH CHANGES
```

to make sure it works, let go back to master branch and see the file change or not


```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git checkout master
Switched to branch 'master'
```



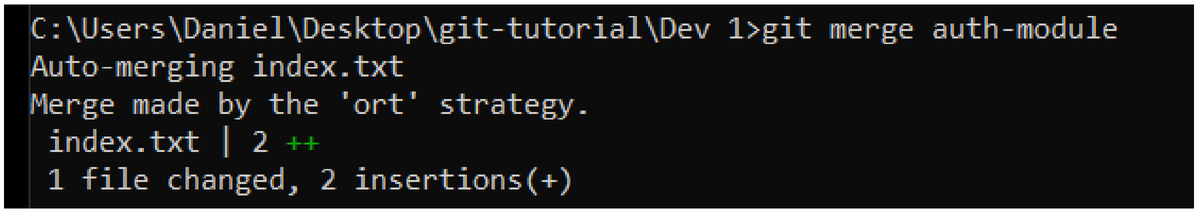
```
my name is Daniel
"Hello World"

blablabla

DEV 2 WAS HEREEE :3
```

the file is the same with the latest one on the master branch, it works perfectly!

now on the master branch will appear changes, now we want to merge it with the one on auth-module branch.

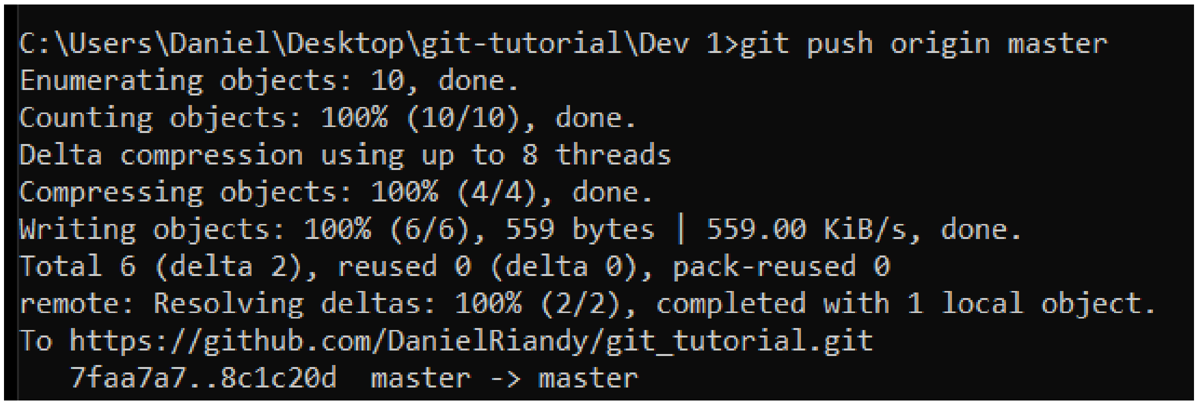
we can use

`git merge [branch name we want to merged with]`



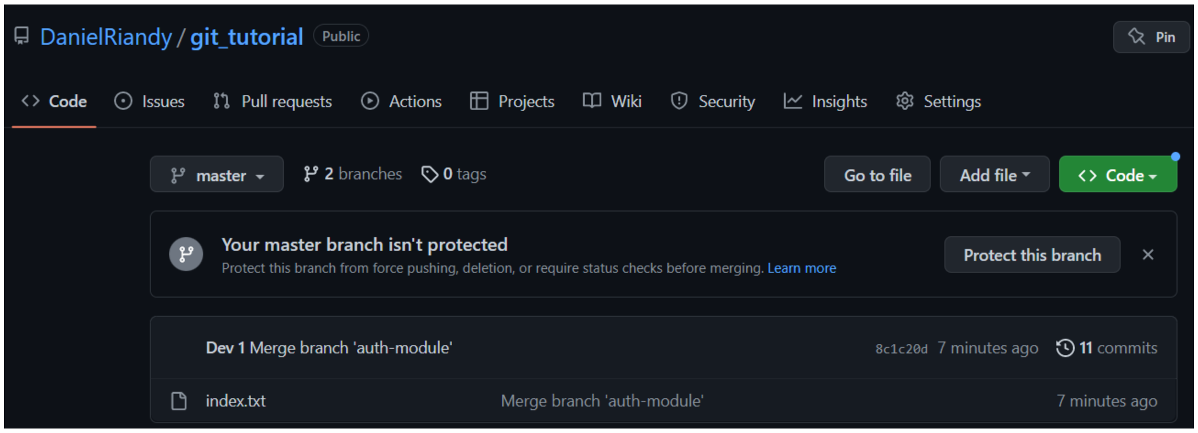
```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git merge auth-module
Auto-merging index.txt
Merge made by the 'ort' strategy.
 index.txt | 2 ++
 1 file changed, 2 insertions(+)
```

then push it to remote repo



```
C:\Users\Daniel\Desktop\git-tutorial\Dev 1>git push origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 559 bytes | 559.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/DanielRiandy/git_tutorial.git
   7faa7a7..8c1c20d  master -> master
```

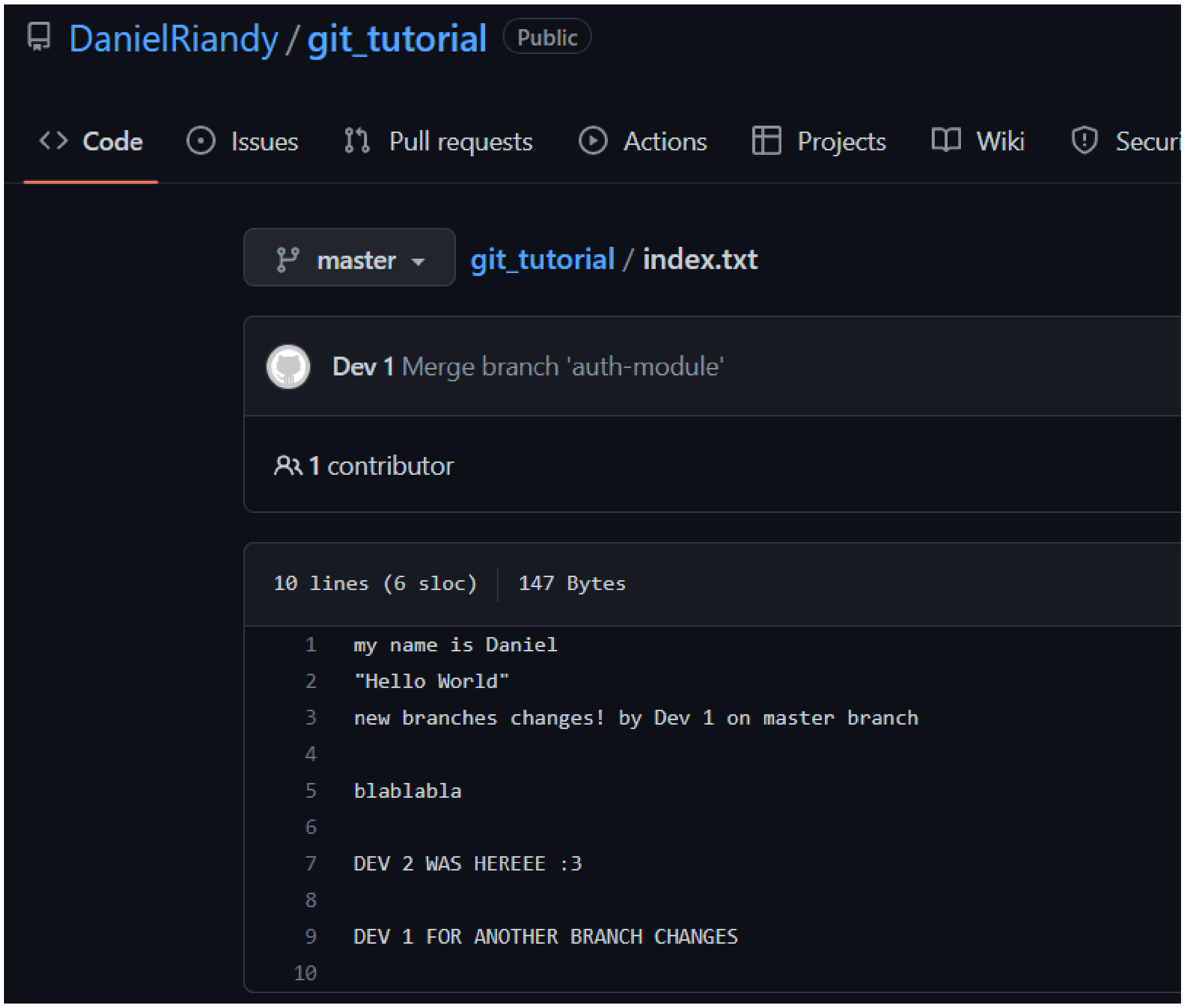
DONE! 🙂


DanielRiandy / git_tutorial Public

Your master branch isn't protected
Protect this branch from force pushing, deletion, or require status checks before merging. Learn more

Dev 1 Merge branch 'auth-module' 8c1c20d 7 minutes ago 11 commits

index.txt Merge branch 'auth-module' 7 minutes ago


DanielRiandy / git_tutorial Public

master git_tutorial / index.txt

Dev 1 Merge branch 'auth-module'

1 contributor

```
10 lines (6 sloc) | 147 Bytes

my name is Daniel
"Hello World"
new branches changes! by Dev 1 on master branch

blablabla

DEV 2 WAS HEREEE :3

DEV 1 FOR ANOTHER BRANCH CHANGES

```