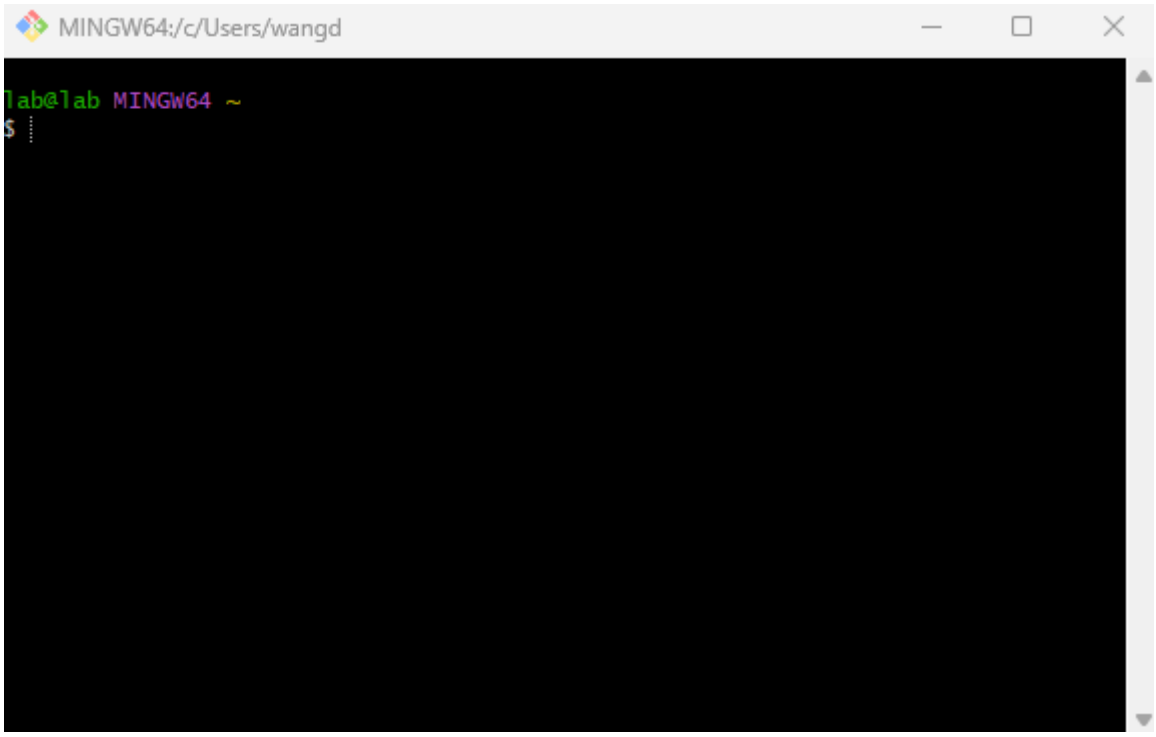
# Introduction to Git

This lab tutorial is about how to use git to manage our projects.

## Open a terminal

After successfully installing git from https://git-scm.com, you should be able to start a "git bash" from your start up menu if you are using windows:



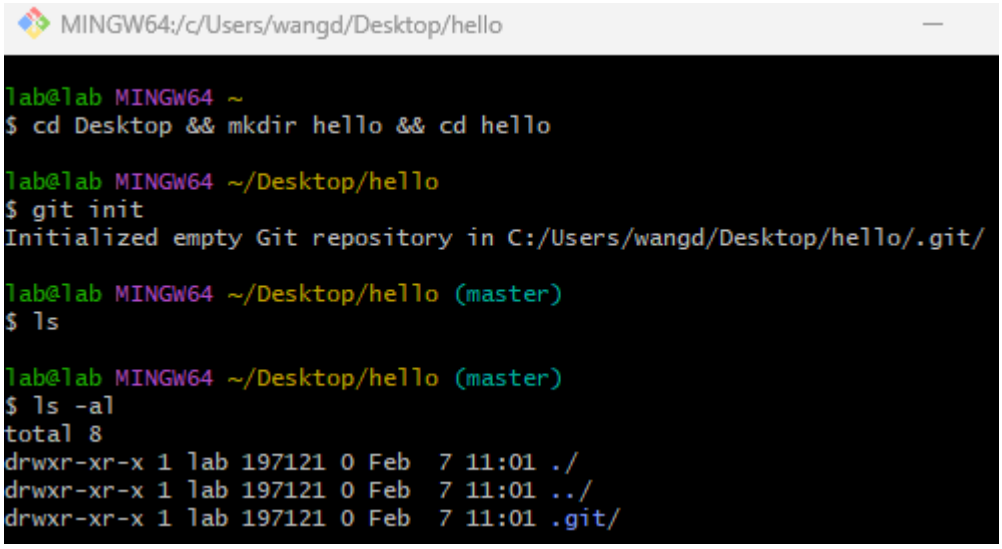

If you are using linux then you only need to open a terminal.

## Create a project

Now create a project with `git init`:

```
git init
```

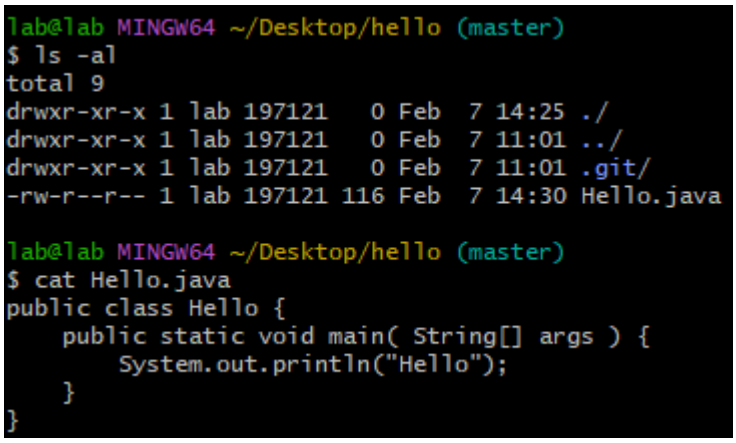
After that the ".git" directory should be created:




The ".git" directory shows that this directory is a git project.

# Add and commit

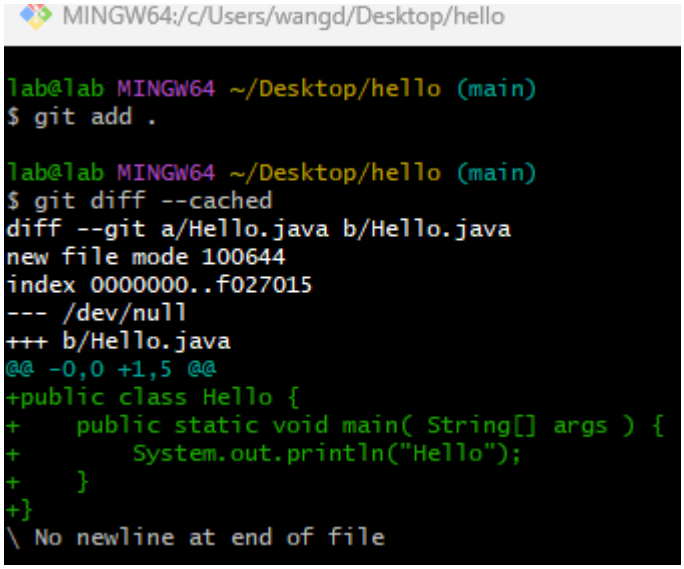
Let's first create a new file in the project:




Then you need to take a snapshot to the staging area:

```
git add .
```

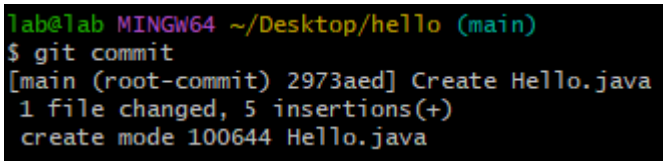
You can use `git diff --cached` to see what is added to your project:




Note that the `git diff` command with or without the `--cached` argument is different. See git document https://git-scm.com/docs/git-diff for more details.

This is not over, you must commit the changes to finish this step with:

```
git commit
```




The terminal will prompt for you to enter the "commit message", it is better if you know how to use "vim".

You can also specify the commit message when doing the commit, so that you don't need to type the commit message in the next step:


# See log

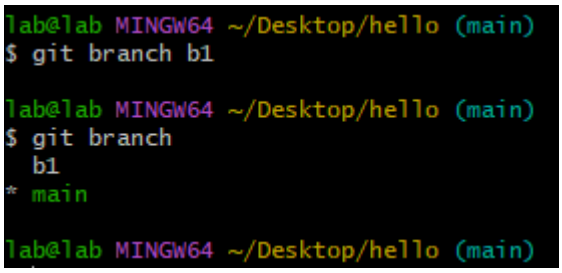
Now we have done one or two commits. If there are many commits in the repository, you may want to see the log to know the history.

You can choose one of the following command or refer to https://www.git-scm.com/docs/git-log for a detailed demostration.

```
git log
git log -p
git log --stat --summary
```

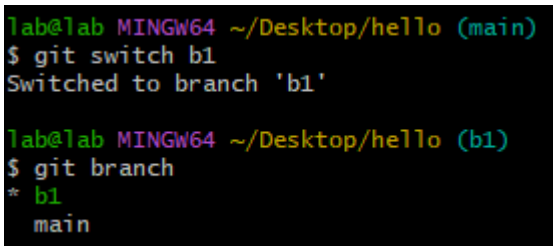
# Making branches

You can have multiple branches in your project.

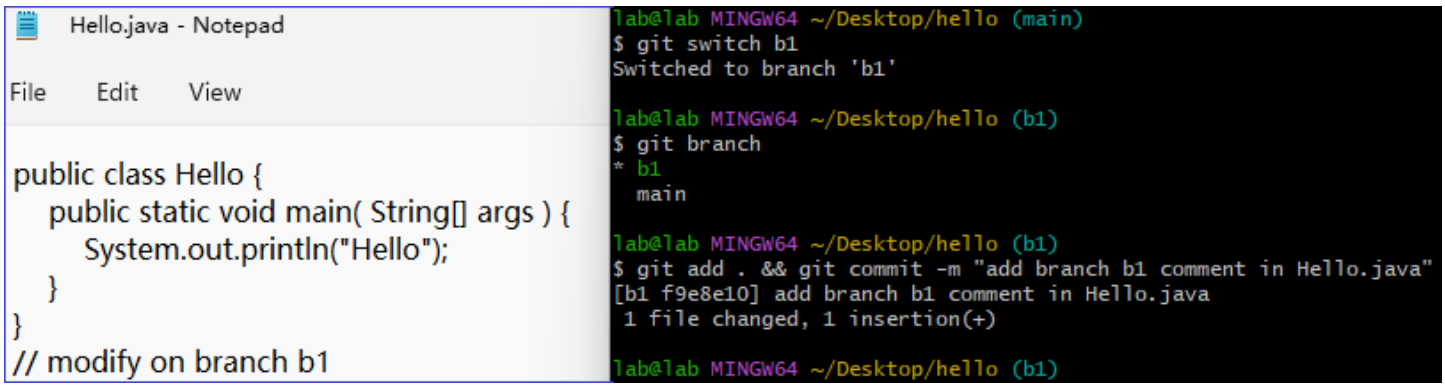



In the above example, we created a new branch named "b1" with command `git branch b1`. Then we use `git branch` to show the branches and we see two branches there: "b1" and "main". The "main" branch is the default one. We also see that the current selected branch is "main".

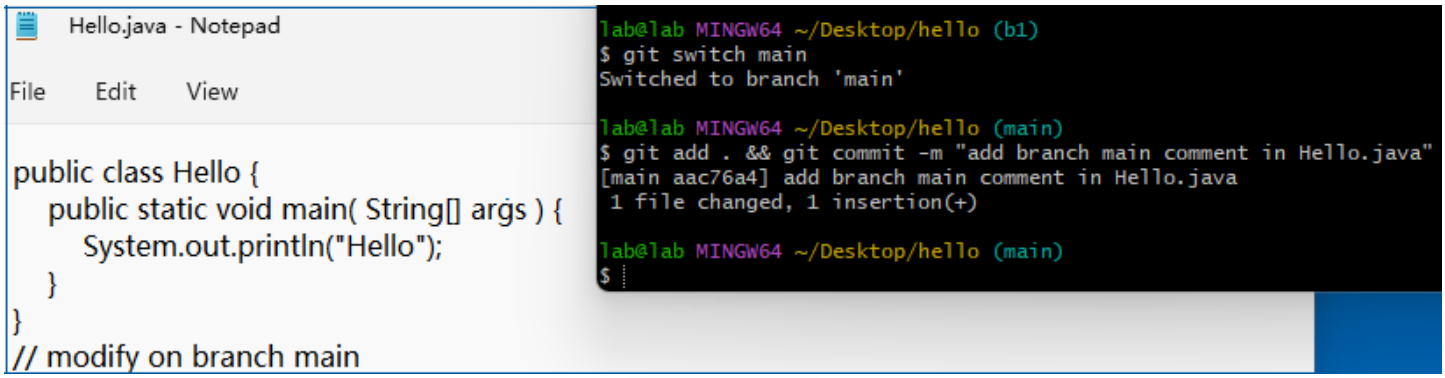
You can switch between branches:



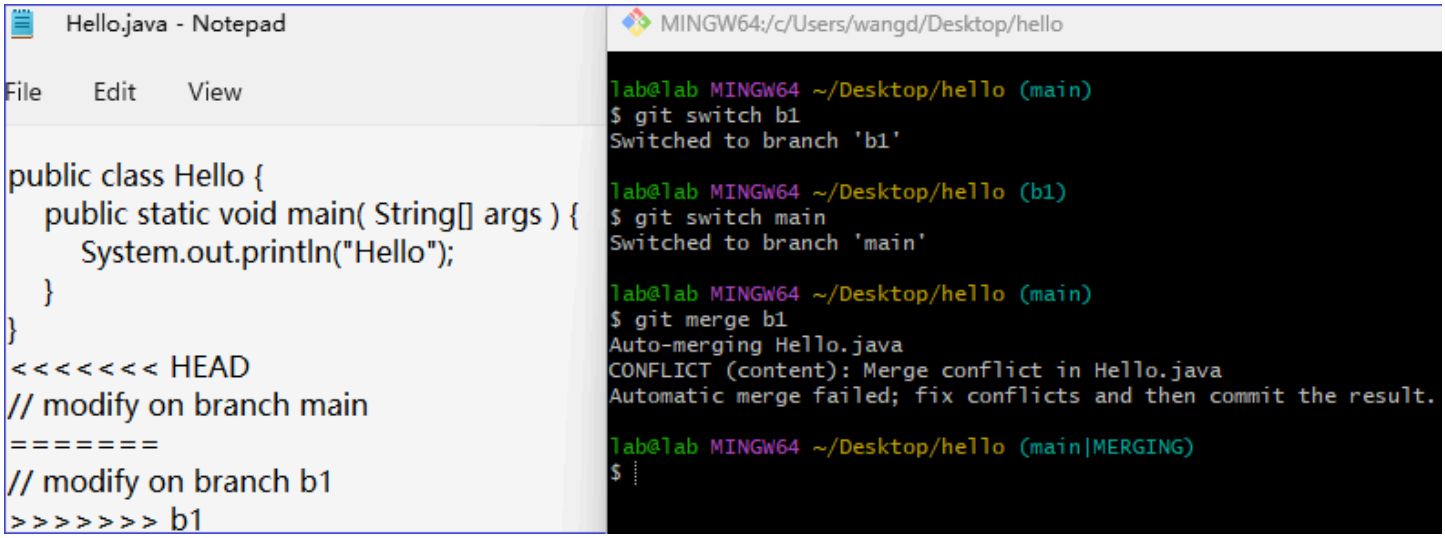

Now let's do some changes and commit on branch "b1":




Go back to "main" and make some changes:

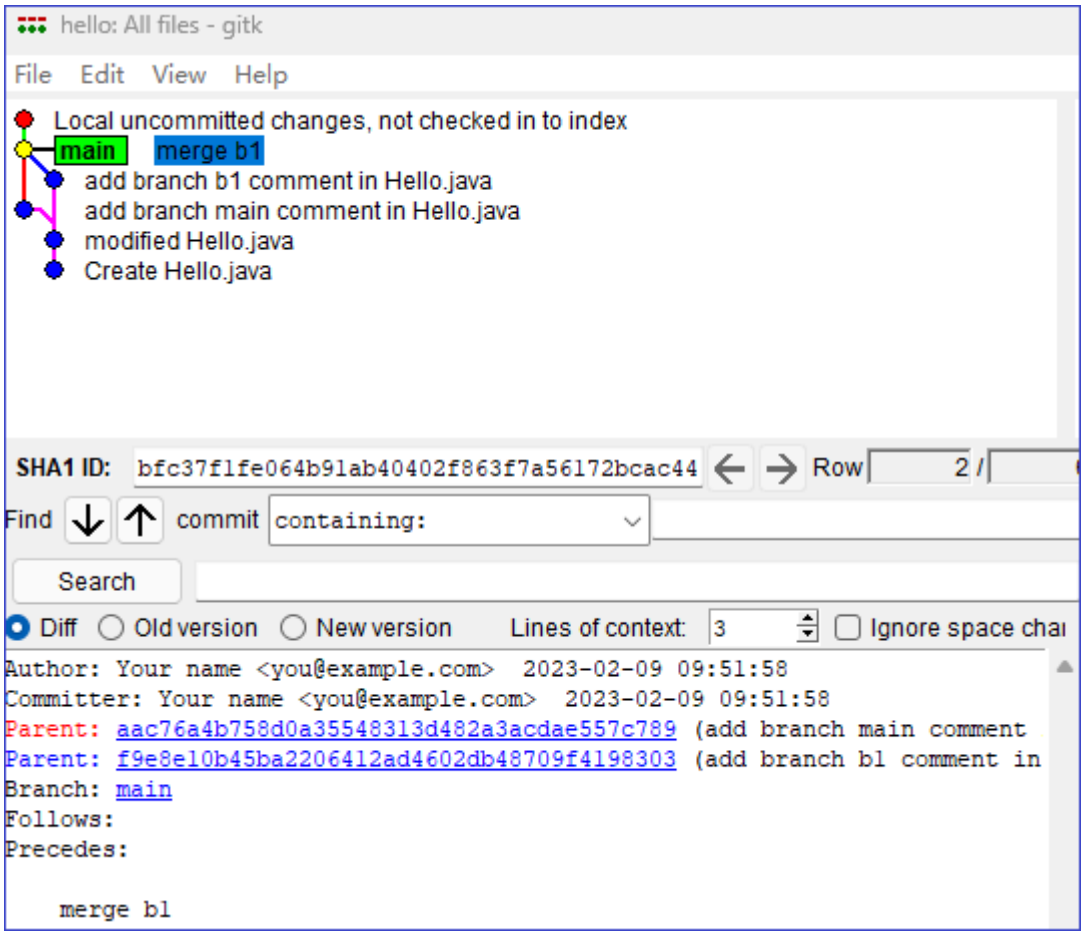



Merge branch "b1" from branch "main" with `git merge b1`. If there is a conflict, as follows:



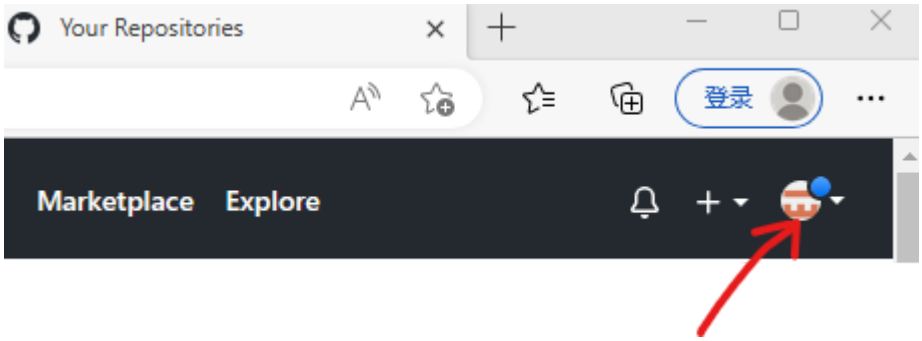

you should manually solve the conflict and do a commit. Then everything is fine and you can remove branch "b1" if it is not useful anymore with `git branch -d b1`. Argument "-d" and "-D" are different, refer to git document for more details.

Now everything is fine and you can execute `gitk` to see a graphical view of the change log:

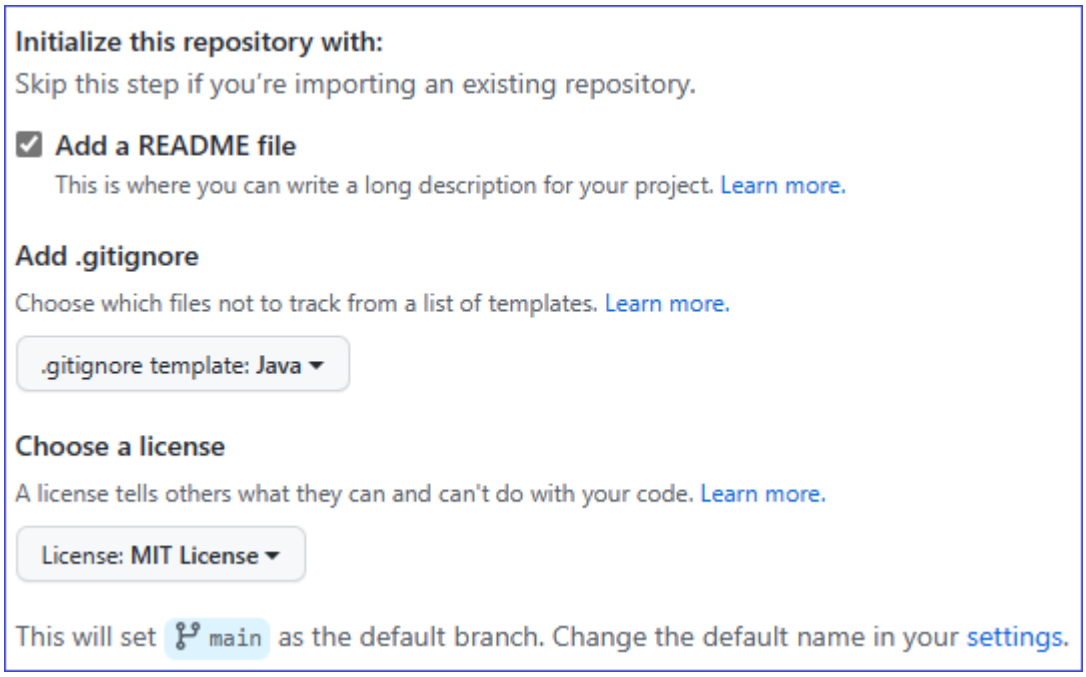



# Using github

Go to https://github.com and register an account if you haven't already done it.



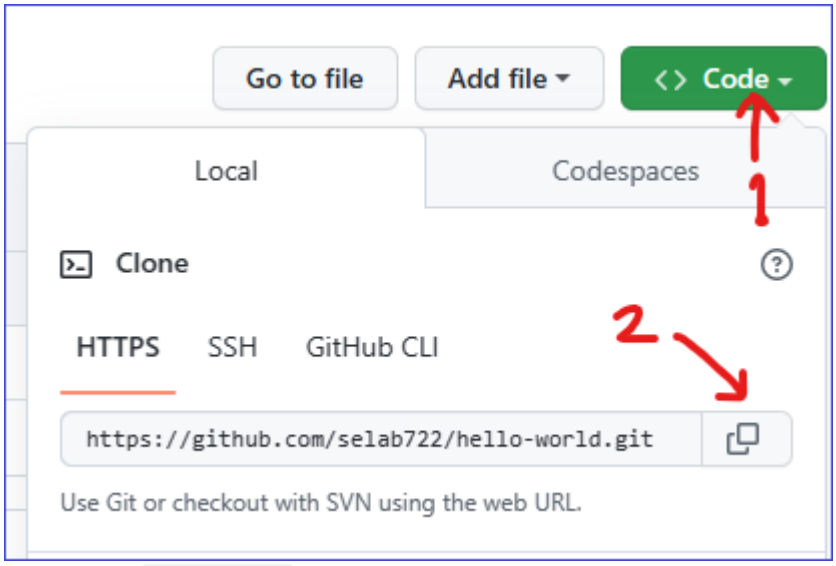

Click the button above, click "repositories", "new" to create a new repository. When github ask you to set the initial configuration, you can choose to add "readme", "gitignore" and "licence" file to you repo:



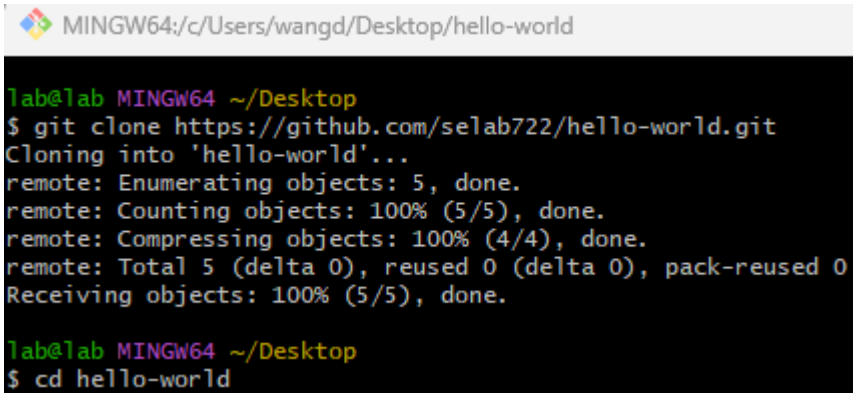

Click "Create repository" and your repository will be created.

You can directly edit some files using github but sometimes you still want to edit them locally, specially when you need an IDE.

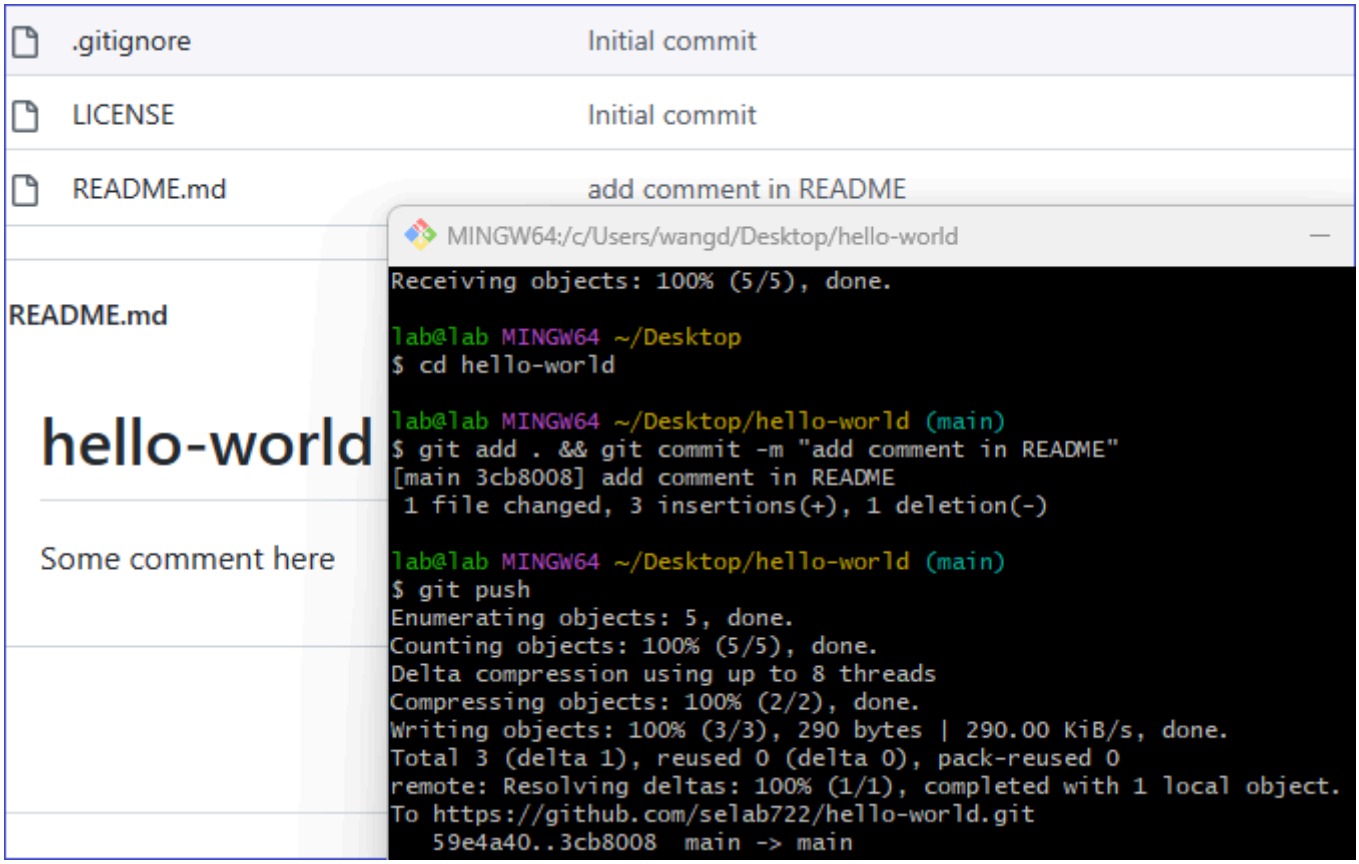
So click "code" and "clone" to copy the link (like https://github.com/username/hello-world.git) to the click board:




Then run `git clone` locally to clone the repository to your local disk:



You can now do any modification as a local project (remember to "cd" into the project directory). You can also use `git pull` to "pull" updates from the github repo to your local. You can use `git push` to "push" local updates to the github repo.




# References

- https://git-scm.com/docs/gittutorial
- https://git-scm.com/docs/user-manual