# Git for scientific projects:
# manage your code and work in teams

Dimitri Marinelli

UBICS

## My experience with git

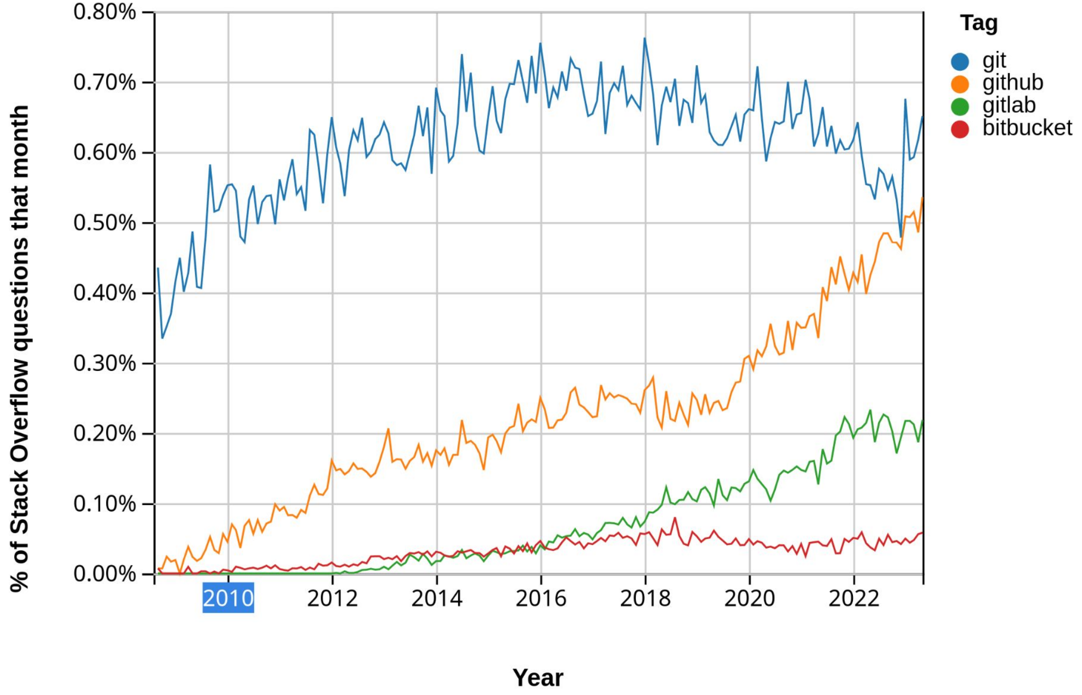
- Before 2016, I used git only for small personal projects.
- Postdoc on deep learning in research project Deep Riemann (we used git for all the projects, common and private repositories)
- Marie Skłodowska-Curie in German financial companies (not always, but when developer teams were involved we always used git)
- Data science and data engineering consultant (DevOps pipeline - git was given for granted)

It is years, but…

**I am not**

## What I actually do:

Google + stack overflow



## A moment of clarity: two different things

The software

git

Services

GitHub

GitLab*

ATLASSIAN
Bitbucket

* is also a software for in premises repository for git

## The starting point: a folder in your computer

You want to track what is happening within the files in the folder on my local machine.

All the files? … well, only the interesting ones.

- Git is very good in tracking **text files** (like code, latex, mark down… )
- Can track binaries or any other file (images, pdfs, mathematica notebooks)

IT IS OPTIMAL TO TRACK YOUR WORK NOT THE WORK OF THE COMPUTER:

**Large files (e.g. over 100MB) can make your life not easy.**

**[ There are other tools, if you really want it, e.g. DVC, Git LFS ]**

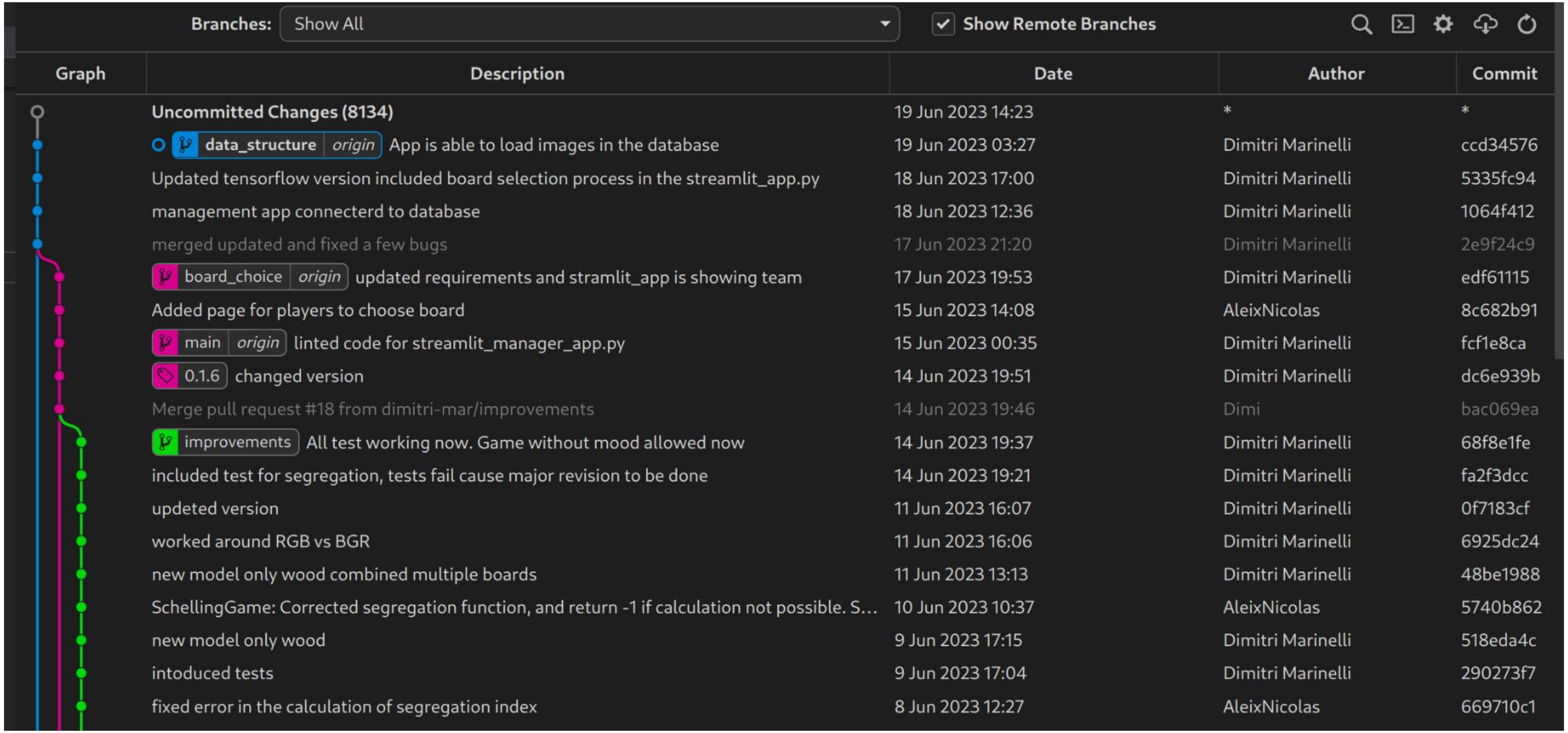
## From your command line, let’s track code in a directory.

Git commands are usually composed of

```
git command --options -o
```

```
> cd git_course/
git_course> git init
    Initialized empty Git repository in /home/dimitri/UB/git_course/.git/
git_course (main)> git status
    On branch main

    No commits yet

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

# The history is made of commits:

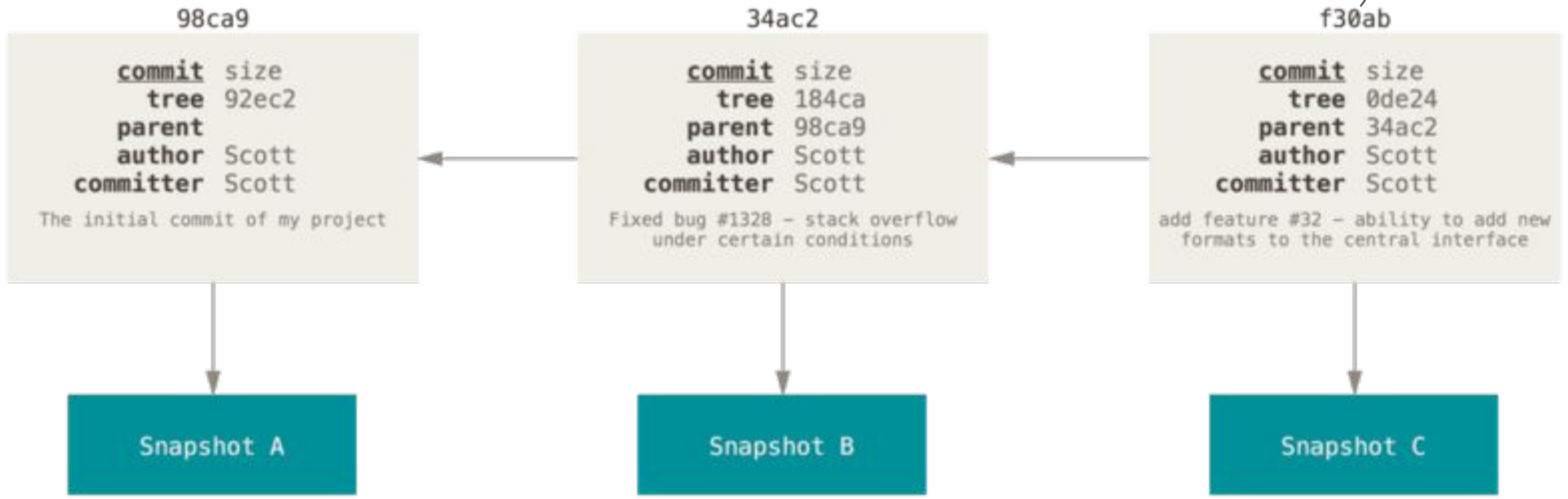


Branches: Show All    Show Remote Branches

| Graph | Description | Date | Author | Commit |
|---|---|---|---|---|
| | Uncommitted Changes (8134) | 19 Jun 2023 14:23 | * | * |
| | data_structure origin App is able to load images in the database | 19 Jun 2023 03:27 | Dimitri Marinelli | ccd34576 |
| | Updated tensorflow version included board selection process in the streamlit_app.py | 18 Jun 2023 17:00 | Dimitri Marinelli | 5335fc94 |
| | management app connecterd to database | 18 Jun 2023 12:36 | Dimitri Marinelli | 1064f412 |
| | merged updated and fixed a few bugs | 17 Jun 2023 21:20 | Dimitri Marinelli | 2e9f24c9 |
| | board_choice origin updated requirements and stramlit_app is showing team | 17 Jun 2023 19:53 | Dimitri Marinelli | edf61115 |
| | Added page for players to choose board | 15 Jun 2023 14:08 | AleixNicolas | 8c682b91 |
| | main origin linted code for streamlit_manager_app.py | 15 Jun 2023 00:35 | Dimitri Marinelli | fcf1e8ca |
| | 0.1.6 changed version | 14 Jun 2023 19:51 | Dimitri Marinelli | dc6e939b |
| | Merge pull request #18 from dimitri-mar/improvements | 14 Jun 2023 19:46 | Dimi | bac069ea |
| | improvements All test working now. Game without mood allowed now | 14 Jun 2023 19:37 | Dimitri Marinelli | 68f8e1fe |
| | included test for segregation, tests fail cause major revision to be done | 14 Jun 2023 19:21 | Dimitri Marinelli | fa2f3dcc |
| | updeted version | 11 Jun 2023 16:07 | Dimitri Marinelli | 0f7183cf |
| | worked around RGB vs BGR | 11 Jun 2023 16:06 | Dimitri Marinelli | 6925dc24 |
| | new model only wood combined multiple boards | 11 Jun 2023 13:13 | Dimitri Marinelli | 48be1988 |
| | SchellingGame: Corrected segregation function, and return -1 if calculation not possible. S... | 10 Jun 2023 10:37 | AleixNicolas | 5740b862 |
| | new model only wood | 9 Jun 2023 17:15 | Dimitri Marinelli | 518eda4c |
| | intoduced tests | 9 Jun 2023 17:04 | Dimitri Marinelli | 290273f7 |
| | fixed error in the calculation of segregation index | 8 Jun 2023 12:27 | AleixNicolas | 669710c1 |

## Each commit points to a previous commit

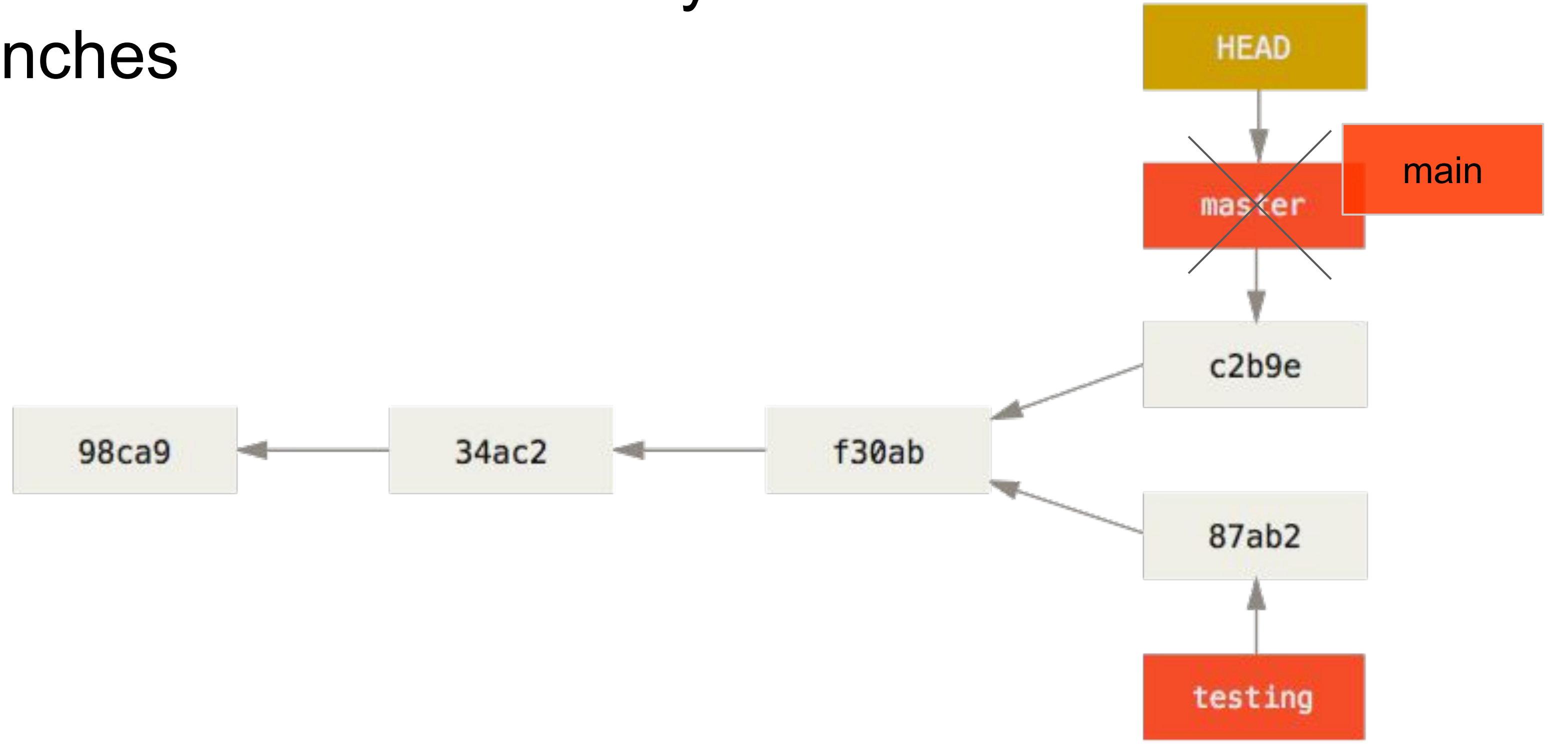
Unique id of the commit



In the folder you do not see anything. All the logic takes place in a hidden folder `git_course/.git`

[many images are taken from [https://git-scm.com/book/en/v2](https://git-scm.com/book/en/v2) Pro Git book]

## This data structure naturally allows branches



You do not have to worry of the data structure

# Let’s create our first commit:

```
git_course (main)> ls
        README.md

git_course (main)> git add README.md

git_course (main)> git status
        On branch main

        No commits yet

        Changes to be committed:
         (use "git rm --cached ..." to unstage)
            new file:   README.md

git_course (main)> git commit -m "first commit"

        [main (root-commit) 714918b] first commit
        1 file changed, 1 insertion(+)
        create mode 100644 README.md
```

The file is “staged”

## One of my personal opinions:

- There are shortcuts:
  ```
  git add .
  git add *
  ```

  or even:
  `git commit -a -m “another version”` (without staging)

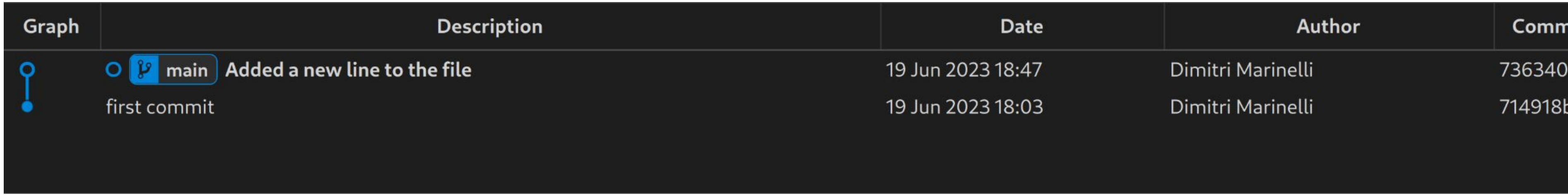
**I suggest: do not to use them.**
Our folders are full of plots, spreadsheets,  csv files, file generated by the simulations, scripts inherited but never used…
You do not want to accidentally commit everything.

**UN-DOing is** always **longer** than writing one extra command.

## We can modify our tracked files.

```
# I can add a new line to README.md
git_course (main)> git status
On branch main
Changes not staged for commit:
 (use "git add ..." to update what will be committed)
 (use "git restore ..." to discard changes in working directory)
       modified:   README.md
no changes added to commit (use "git add" and/or "git commit -a")
git_course (main)> git add README.md
git_course (main)> git commit -m "Added a new line to the README file"
[main 7363406] Added a new line to the file
1 file changed, 2 insertions(+), 1 deletion(-)
```

The message should be meaningful



| Graph | Description | Date | Author | Comm |
|---|---|---|---|---|
| | main Added a new line to the file | 19 Jun 2023 18:47 | Dimitri Marinelli | 736340 |
| | first commit | 19 Jun 2023 18:03 | Dimitri Marinelli | 714918b |

Another personal opinion: <u>commit often!</u>

…. But it is also good if you commit something that work-ish.

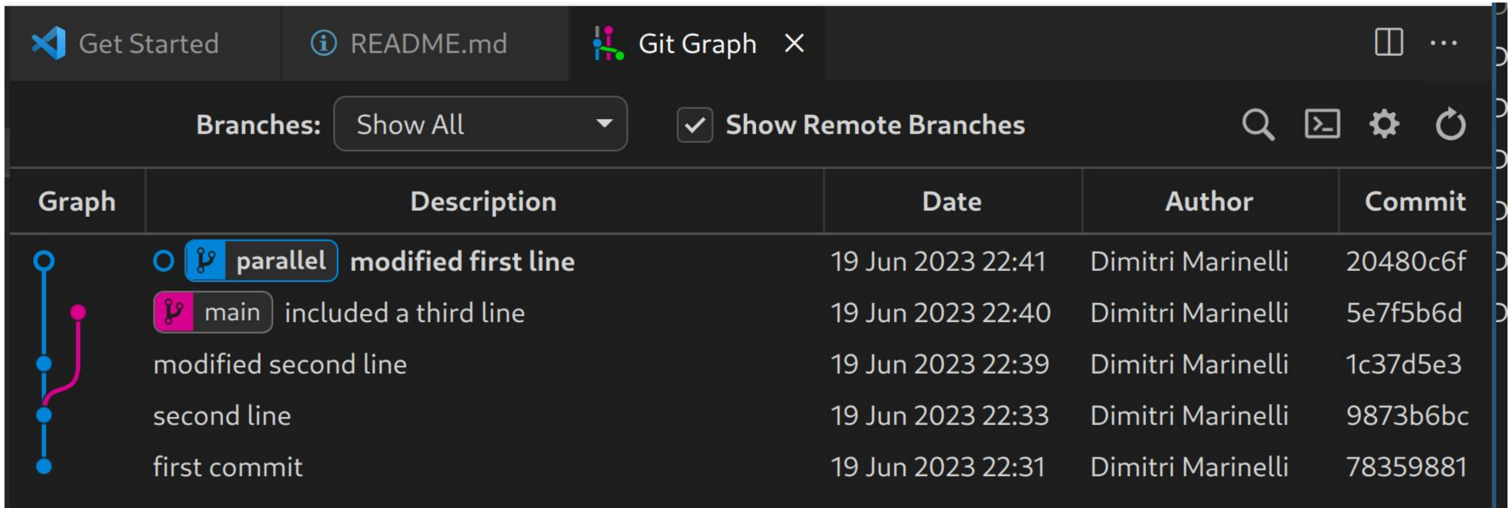
# git diff - to compare different point in history

```
diff --git a/README.md
b/README.md
index 6dd8be0..184a4f9 100644
--- a/README.md
+++ b/README.md
@@ -1,2 +1,4 @@
First line. I modify also the first line!
-A second line. Change the second
line
\ No newline at end of file
+A second line. Change the second
line
+
+A forth line
\ No newline at end of file
(END)
```

# Create a new branch

`git checkout where_you_want_to_go`
is the command used for navigating the history.

`git checkout -b new_branch`
(it prepares the folder to changes that will appear in the commits of the new branch)



| Graph | Description | Date | Author | Commit |
|---|---|---|---|---|
| | parallel modified first line | 19 Jun 2023 22:41 | Dimitri Marinelli | 20480c6f |
| | main included a third line | 19 Jun 2023 22:40 | Dimitri Marinelli | 5e7f5b6d |
| | modified second line | 19 Jun 2023 22:39 | Dimitri Marinelli | 1c37d5e3 |
| | second line | 19 Jun 2023 22:33 | Dimitri Marinelli | 9873b6bc |
| | first commit | 19 Jun 2023 22:31 | Dimitri Marinelli | 78359881 |

## If we want to look at what is different between the two branches

GOOGLE IT! - an examples I never remember.

It is important to know that git is integrated with “diff” software to compare text files.

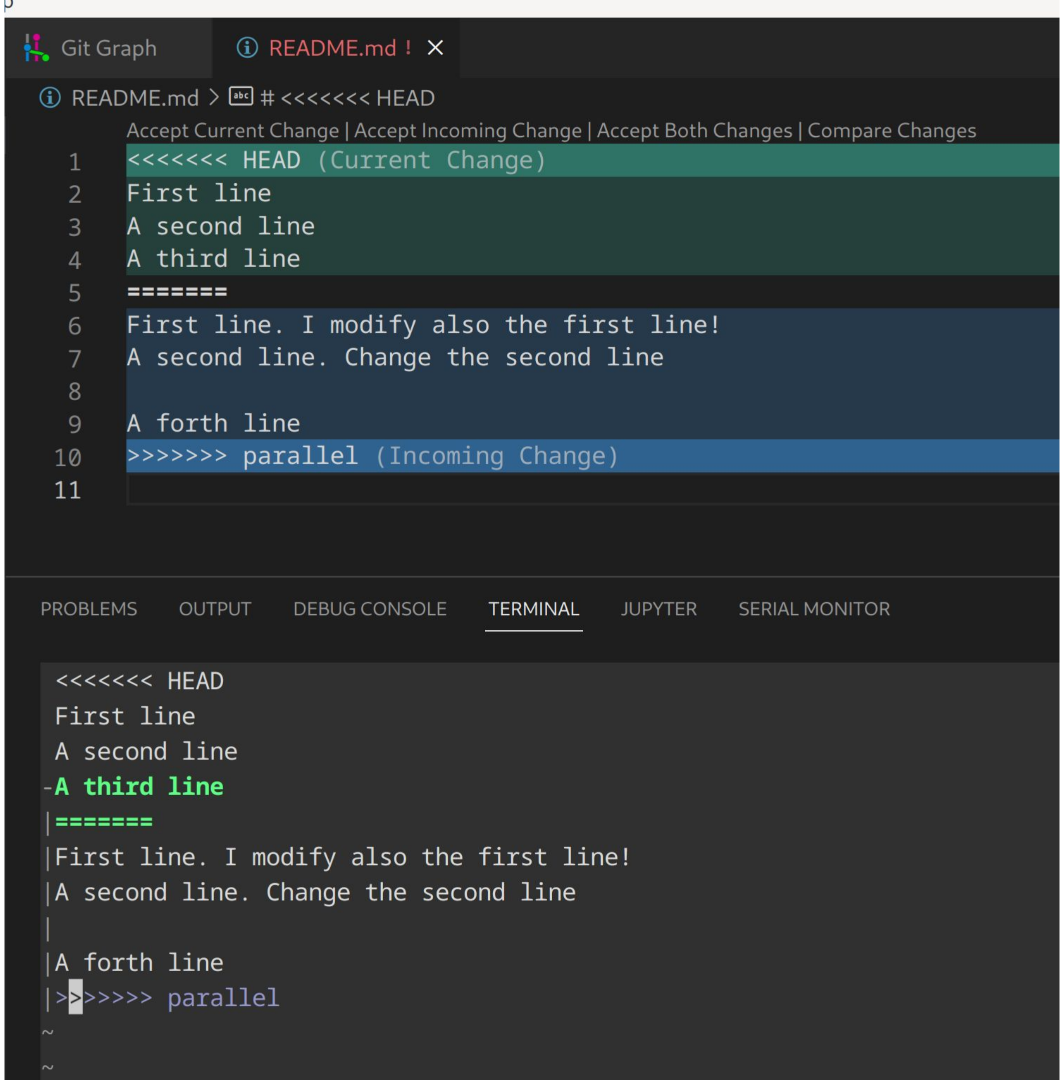
## We want to update “main” with the changes of “parallel”

## We want to update “main” with the changes of “parallel”

```
git_course (parallel)> git checkout main

Switched to branch 'main'

git_course (main)> git merge parallel

Auto-merging README.md

CONFLICT (content): Merge conflict in README.md

Automatic merge failed; fix conflicts and then commit the result.

git_course (main|MERGING) [1]> git status


[ solve the conflicts …]

git_course (main|MERGING)> git commit -m "a merge"

[main 3ec9e91] a merge

git_course (main)>
```

How the README.md looks:

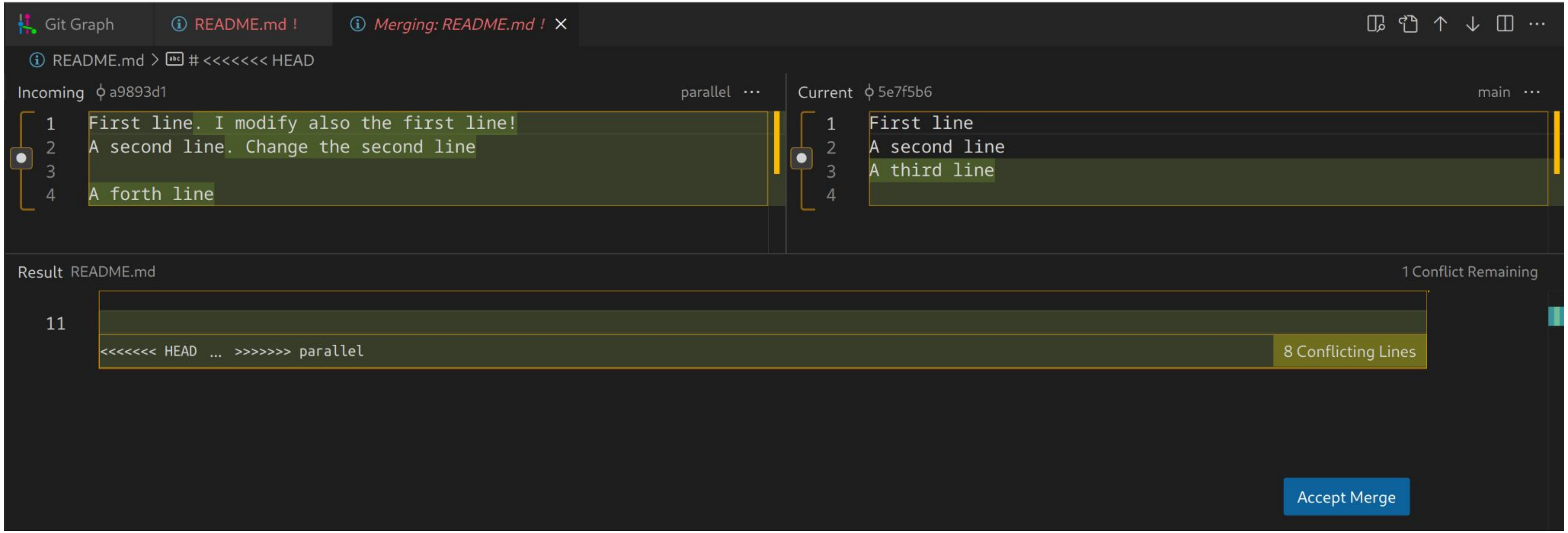
- Above in VScode
- Below in pure txt


Git Graph
README.md !
README.md > # <<<<<<< HEAD
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
SERIAL MONITOR

```
<<<<<<< HEAD (Current Change)
First line
A second line
A third line
=======
First line. I modify also the first line!
A second line. Change the second line

A forth line
>>>>>>> parallel (Incoming Change)
```

```
<<<<<<< HEAD
First line
A second line
-A third line
|=======
|First line. I modify also the first line!
|A second line. Change the second line
|
|A forth line
|>>>>>>> parallel
~
~
```

# Another format



Incoming:

```
First line. I modify also the first line!
A second line. Change the second line

A forth line
```

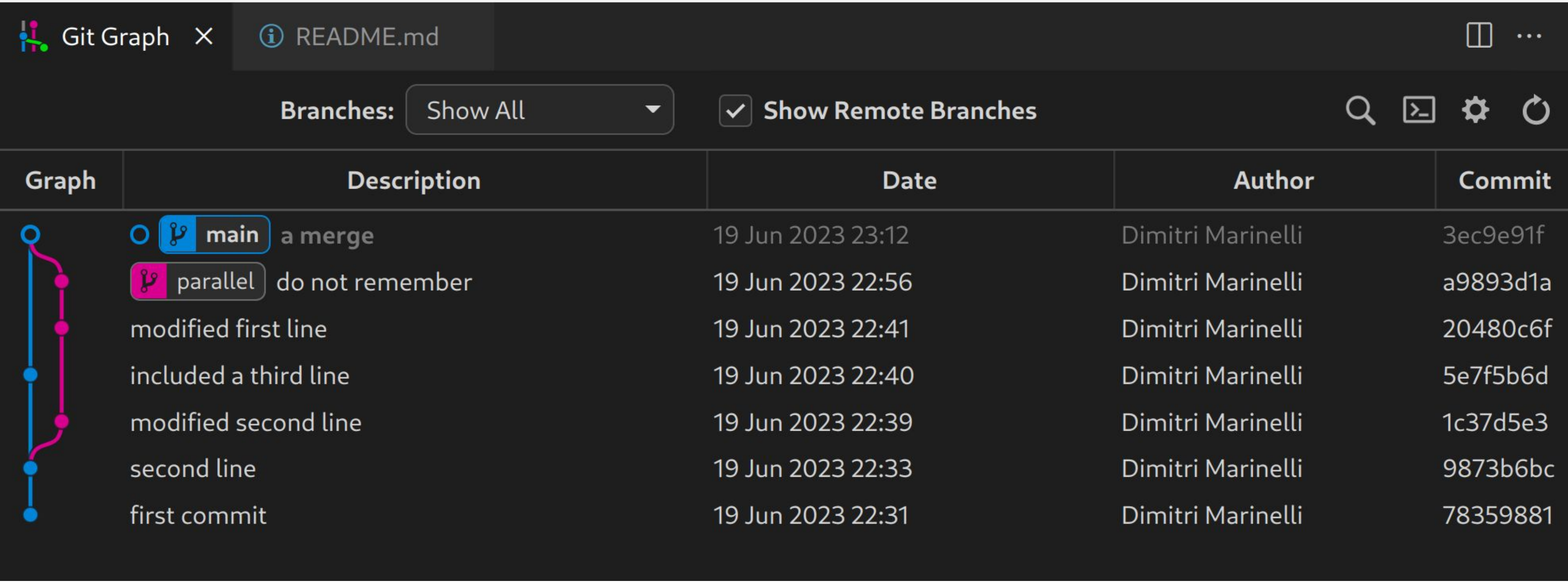
Current:

```
First line
A second line
A third line

```

Result:

```
11
<<<<<<< HEAD … >>>>>>> parallel
```

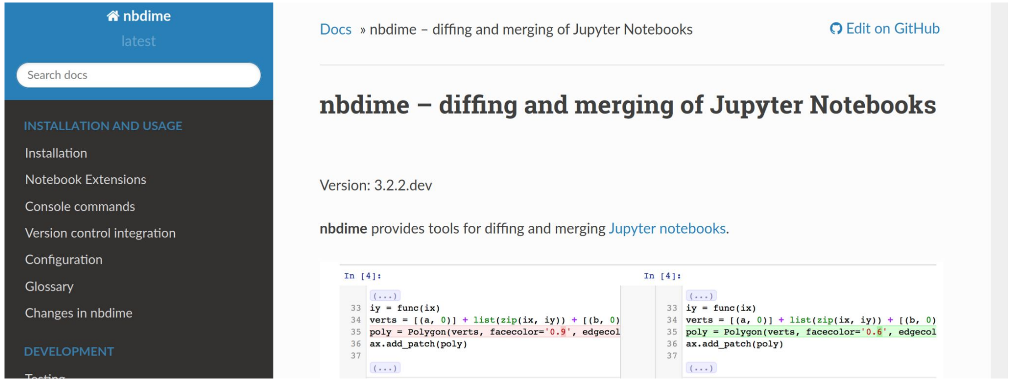
## The merge is accomplished with a commit.



| Graph | Description | Date | Author | Commit |
|---|---|---|---|---|
| | main a merge | 19 Jun 2023 23:12 | Dimitri Marinelli | 3ec9e91f |
| | parallel do not remember | 19 Jun 2023 22:56 | Dimitri Marinelli | a9893d1a |
| | modified first line | 19 Jun 2023 22:41 | Dimitri Marinelli | 20480c6f |
| | included a third line | 19 Jun 2023 22:40 | Dimitri Marinelli | 5e7f5b6d |
| | modified second line | 19 Jun 2023 22:39 | Dimitri Marinelli | 1c37d5e3 |
| | second line | 19 Jun 2023 22:33 | Dimitri Marinelli | 9873b6bc |
| | first commit | 19 Jun 2023 22:31 | Dimitri Marinelli | 78359881 |

## Warning with notebooks and binary stuff

Jupyter needs its own tool



nbdime

latest

Search docs

INSTALLATION AND USAGE

- Installation
- Notebook Extensions
- Console commands
- Version control integration
- Configuration
- Glossary
- Changes in nbdime

DEVELOPMENT

Docs » nbdime – diffing and merging of Jupyter Notebooks    Edit on GitHub

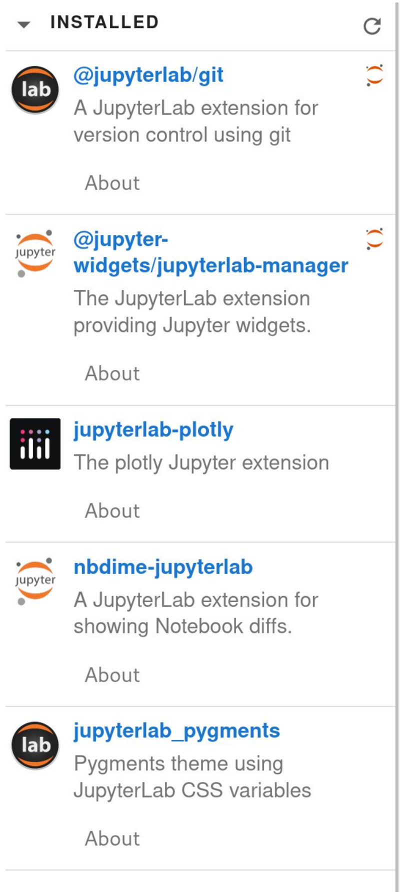
### nbdime – diffing and merging of Jupyter Notebooks

Version: 3.2.2.dev

**nbdime** provides tools for diffing and merging Jupyter notebooks.

```
In [4]:
(...)
iy = func(ix)
verts = [(a, 0)] + list(zip(ix, iy)) + [(b, 0)
poly = Polygon(verts, facecolor='0.9', edgecol
ax.add_patch(poly)

(...)
```

```
In [4]:
(...)
iy = func(ix)
verts = [(a, 0)] + list(zip(ix, iy)) + [(b, 0)
poly = Polygon(verts, facecolor='0.6', edgecol
ax.add_patch(poly)

(...)
```



INSTALLED

**@jupyterlab/git**
A JupyterLab extension for version control using git
About

**@jupyter-widgets/jupyterlab-manager**
The JupyterLab extension providing Jupyter widgets.
About

**jupyterlab-plotly**
The plotly Jupyter extension
About

**nbdime-jupyterlab**
A JupyterLab extension for showing Notebook diffs.
About

**jupyterlab_pygments**
Pygments theme using JupyterLab CSS variables
About

## There are many other commands….

```
git stash

git rebase                git branch

git tag                   git blame
```

…

But they are for an advanced course, and, honestly, I use them rarely.

## And one useful file

```
.gitignore
```

## Work in teams

(or synchronize your folder with a remote repository)

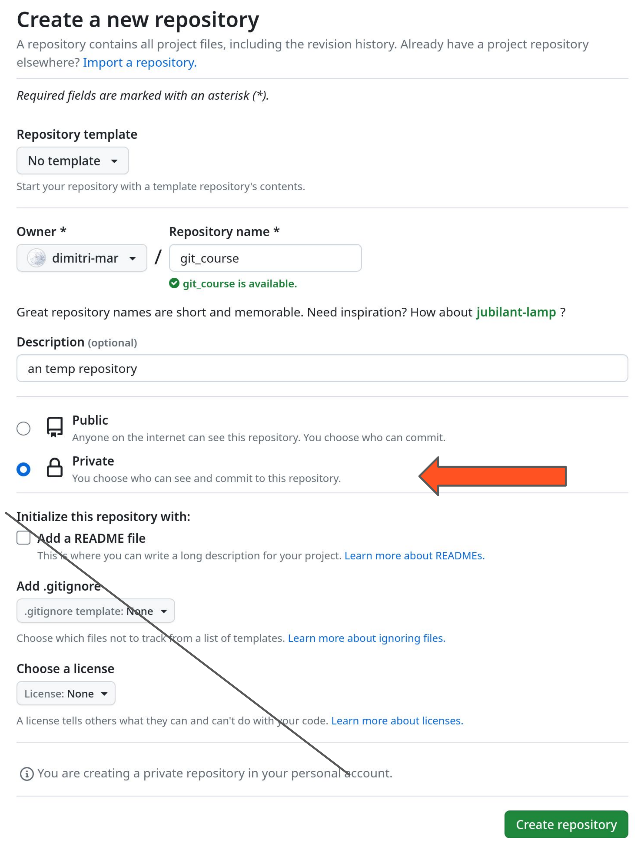
## Ingredient 1. You need a remote repository

GitHub

GitLab

ATLASSIAN
Bitbucket

# Ingredient 1. You need a remote repository



## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

*Required fields are marked with an asterisk (*).*

**Repository template**

No template

Start your repository with a template repository's contents.

**Owner *** dimitri-mar / **Repository name *** git_course

git_course is available.

Great repository names are short and memorable. Need inspiration? How about jubilant-lamp ?

**Description** (optional)

an temp repository

( ) **Public**
Anyone on the internet can see this repository. You choose who can commit.

(✓) **Private**
You choose who can see and commit to this repository.

**Initialize this repository with:**

[ ] **Add a README file**
This is where you can write a long description for your project. Learn more about READMEs.

**Add .gitignore**

.gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

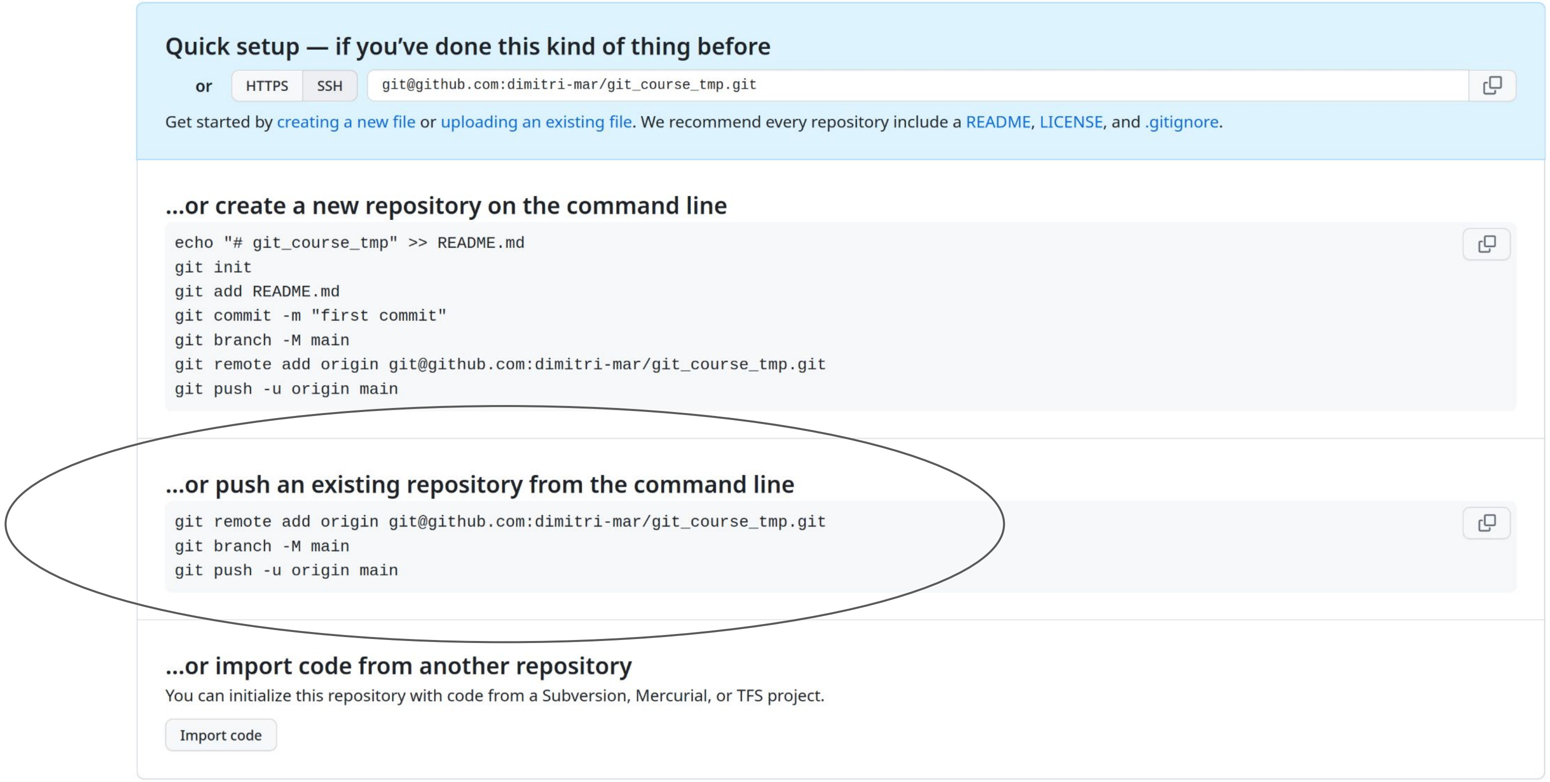
**Choose a license**

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

You are creating a private repository in your personal account.

## Ingredient 2. You need to set the local folder to talk with the remote repository



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

```
git@github.com:dimitri-mar/git_course_tmp.git
```

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

**...or create a new repository on the command line**

```
echo "# git_course_tmp" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:dimitri-mar/git_course_tmp.git
git push -u origin main
```

**...or push an existing repository from the command line**

```
git remote add origin git@github.com:dimitri-mar/git_course_tmp.git
git branch -M main
git push -u origin main
```

**...or import code from another repository**

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

## Ingredient 2. You need to set the local folder to talk with the remote repository

```
git remote add origin git@github.com:dimitri-mar/git_course_tmp.git
```

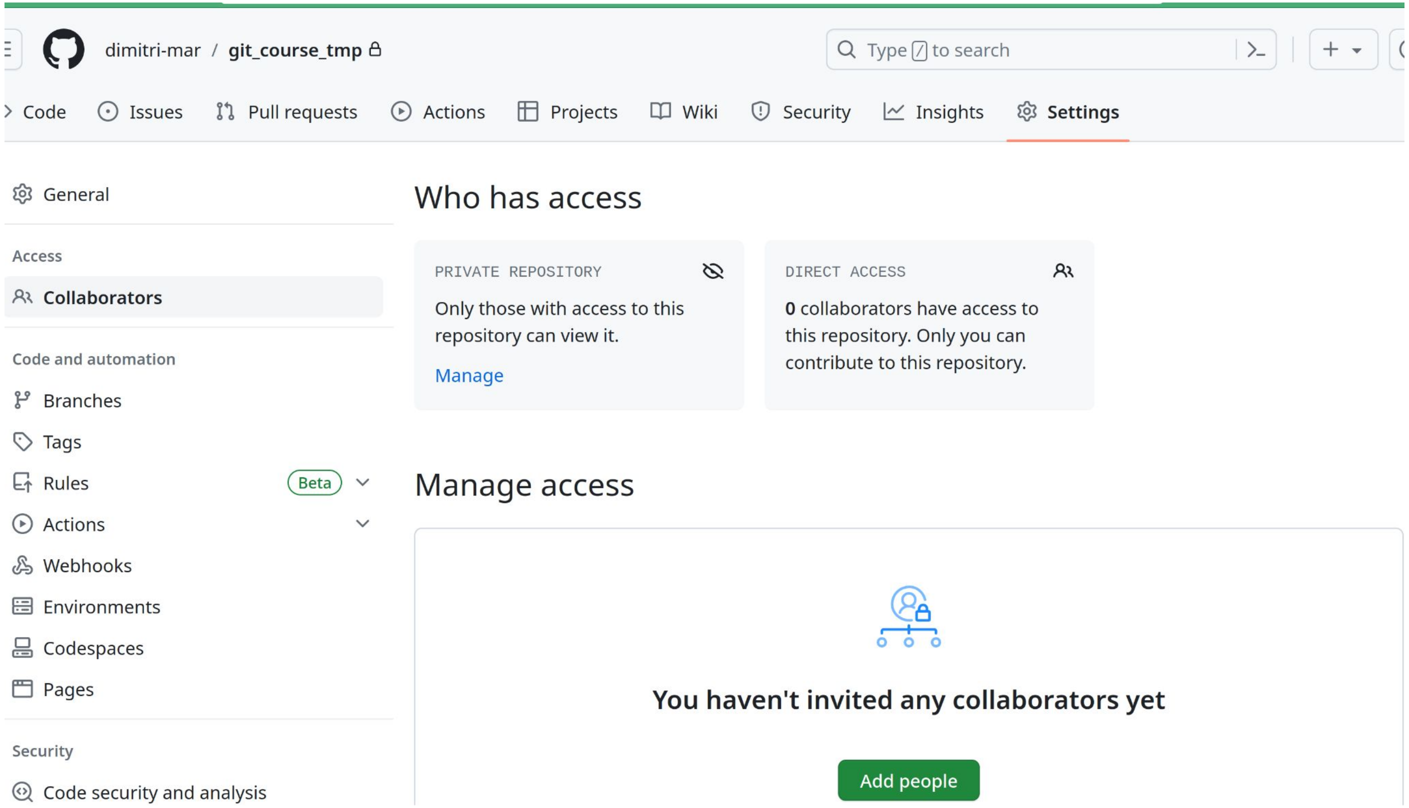
It means: to my local repository (my folder) add a remote repository that locally we will call “`origin`” with the address: **`git@github.com:dimitri-mar/git_course_tmp.git`**

The address is unique to the remote repository.

## Ingredient 3. We want to **push** what we have in our directory into the remote repository (github, gitlab, etc)

```
git_course (main)> git push

fatal: The current branch main has no upstream branch.
To push the current branch and set the remote as upstream, use

   git push --set-upstream origin main

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

# Ingredient 4. You can add collaborators.


dimitri-mar / git_course_tmp

Type / to search

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

General

Access

Collaborators

Code and automation

Branches

Tags

Rules Beta

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security and analysis

## Who has access

PRIVATE REPOSITORY

Only those with access to this repository can view it.

Manage

DIRECT ACCESS

**0** collaborators have access to this repository. Only you can contribute to this repository.

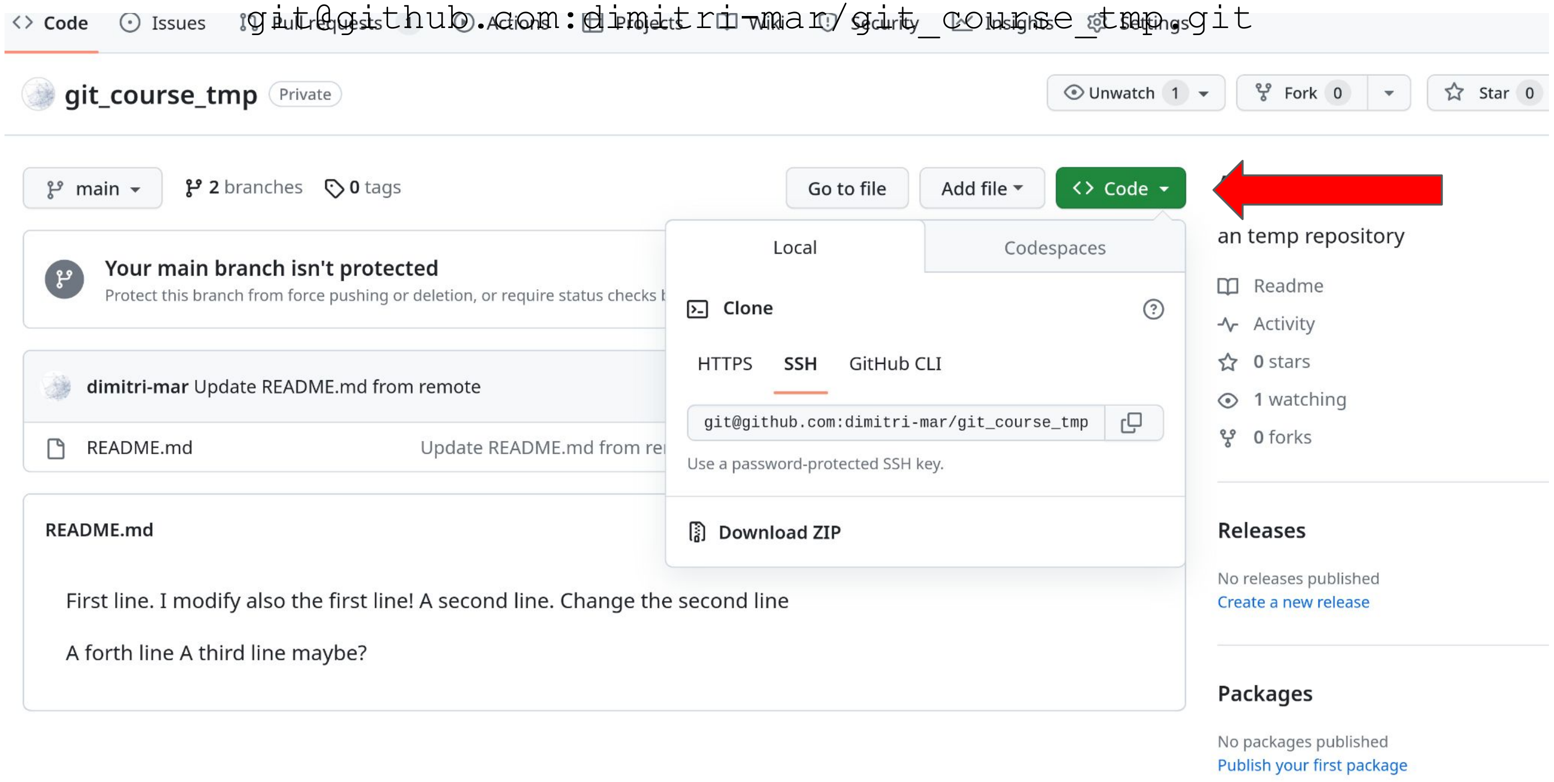
## Manage access

**You haven't invited any collaborators yet**

Add people

## Ingredient 5. Collaborators can **clone** your repository.

```
git clone
git@github.com:dimitri-mar/git_course_tmp.git
```



Your main branch isn't protected

Protect this branch from force pushing or deletion, or require status checks b

dimitri-mar Update README.md from remote

README.md Update README.md from re

git@github.com:dimitri-mar/git_course_tmp

Use a password-protected SSH key.

an temp repository

No releases published

No packages published

README.md

First line. I modify also the first line! A second line. Change the second line

A forth line A third line maybe?

## Last commands: **pull** and **fetch**

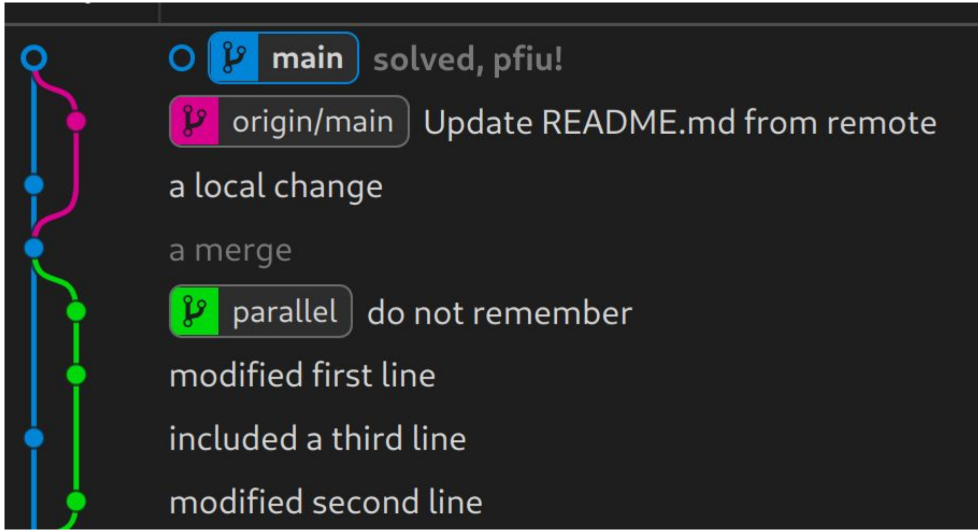
Somebody changed a branch in the remote repository,

- `git pull` from your local branch will download the changes and update the folder
- `git fetch` will download the changes and update the folder

# A (not so) bad scenario: divergent branches

You did not listened my advice! You changed “main” in your local, and at the same time your collaborator changed “main” in “origin”, your remote repository.

```
git_course (main)> git status

On branch main

Your branch is ahead of 'origin/main' by 1 commit.

 (use "git push" to publish your local commits)



nothing to commit, working tree clean
```



```
main solved, pfiu!
origin/main Update README.md from remote
a local change
a merge
parallel do not remember
modified first line
included a third line
modified second line
```

Actually is equivalent to any other branch. You can do a `git merge`

The first time, you will need to tell git what to do, I prefer git merge: `git config pull.rebase false  # merge`

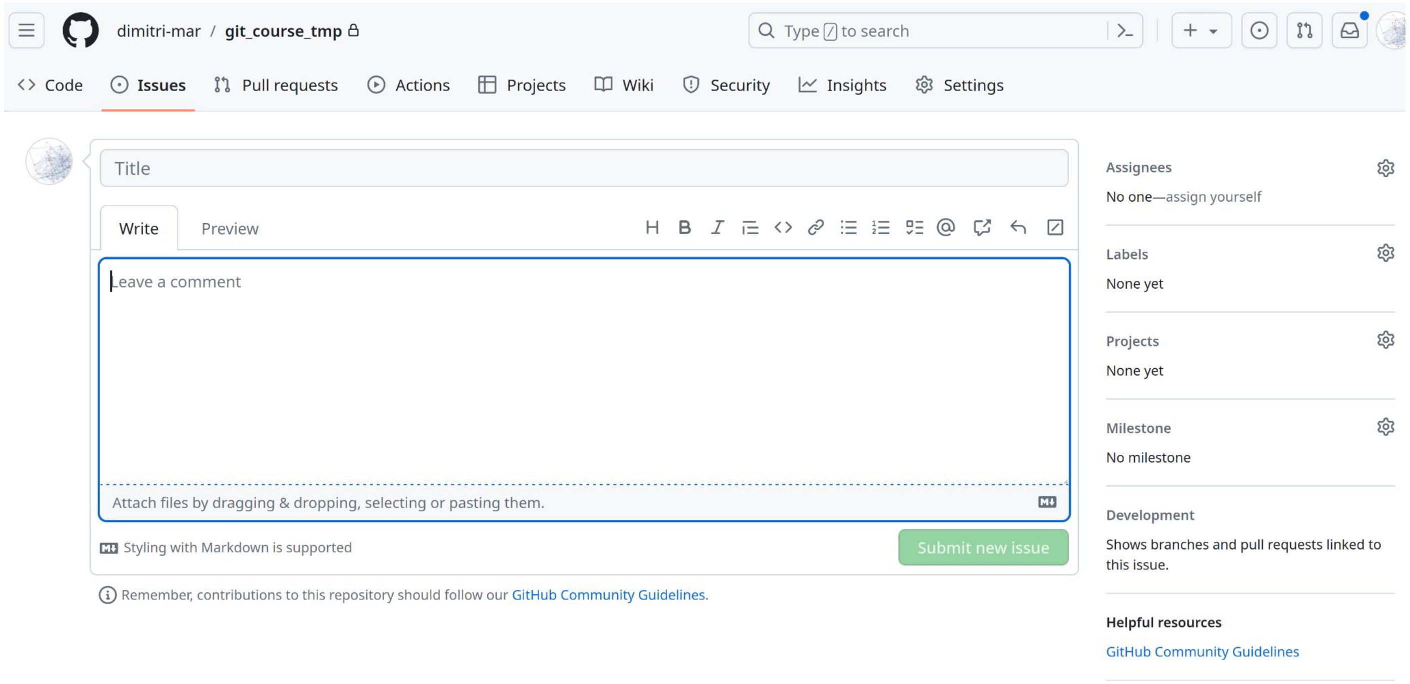
## The first who **push**es does less work! Unless…

# Configure git the first time

(or every time you get a new computer)

## How to collaborate?
## (quick intro)

## You found a bug, but you can't fix it now. Create an issue


dimitri-mar / git_course_tmp
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Title
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Styling with Markdown is supported
Submit new issue
Remember, contributions to this repository should follow our GitHub Community Guidelines.
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Shows branches and pull requests linked to this issue.
Helpful resources
GitHub Community Guidelines

The issue is a space where also non-developers can write and interact. Users, managers, … supervisors ;)

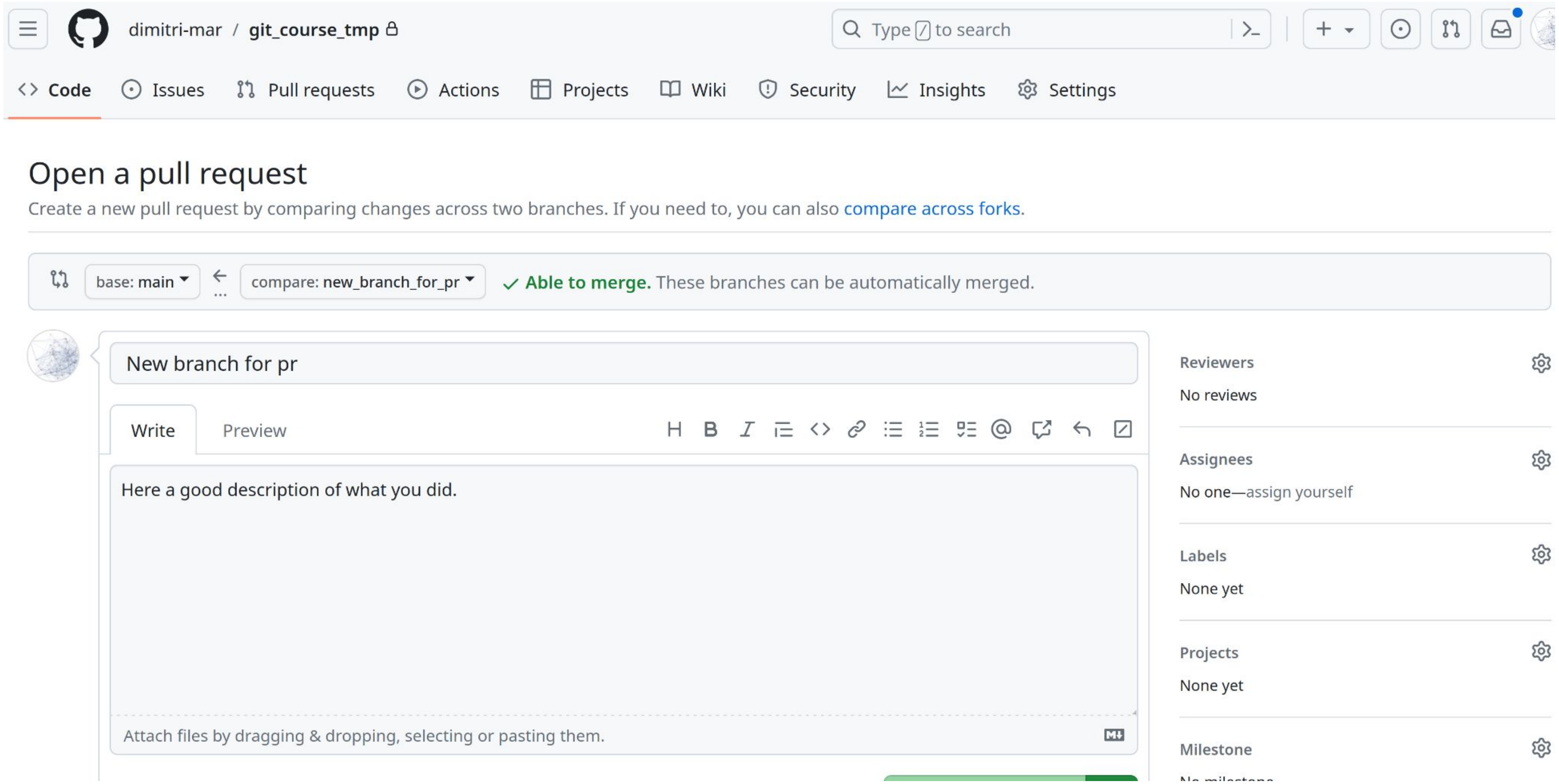
Not only for bugs:

- Proposals
- New features
- User stories

## Pull request

You finish what you are doing in your branch, and want to push it to the main branch but:

- Somebody else is responsible for the **main** branch
- You want to keep track of what you did in your branch when you merge in main.
- You want somebody else **review and approve** and merge your branch
- You are proposing a change in an open source project owned by somebody else.



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: main ← compare: new_branch_for_pr ✓ Able to merge. These branches can be automatically merged.

New branch for pr

Here a good description of what you did.

Attach files by dragging & dropping, selecting or pasting them.

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone

## Now let’s try it out!