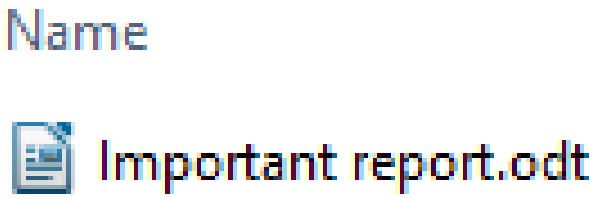
Versioning with git

BTM 2017 – Tom Boissonnet

EMBL

Bad versioning



What's the problem there ?

- Every piece of the file is duplicated, even the unedited parts
- Modifications are dispersed across the versions
- Need to think about new names every time
- Not synchronised across computers

What could we do

- Instead of duplicating all files, keeping the only original file
- Having beside a journal to keep track of all modifications
- Share just these two files instead of a whole librairy

Ever heard about Version control software ?

- Git the most popular today, with the plateform GitHub

Git origin

- Linus Torvald, creator of Linux, key figure of OpenSource
- Git created in 2005 to initially develop the linux kernel
- WWCVSND What would CVS not do ? If in doubt, make the exact opposite decision

Git functionalities

- Keep track of modifications: new files, additions, deletions
- Work on different branches
- Revert back and forth to other versions
- Allow editing by multiple person: Authentification

GitHub functionalities

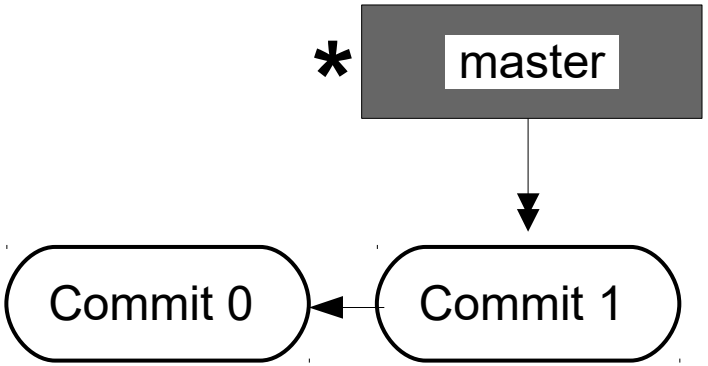
- Cloud storage for files and the git tracking
- Graphic interface
- Code release, wiki, statistics, issues, projects

git init

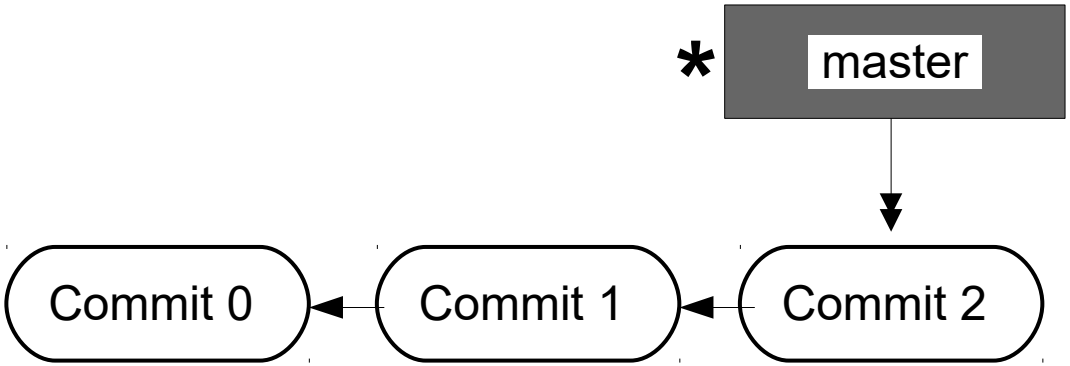
Computer

git_btm2017

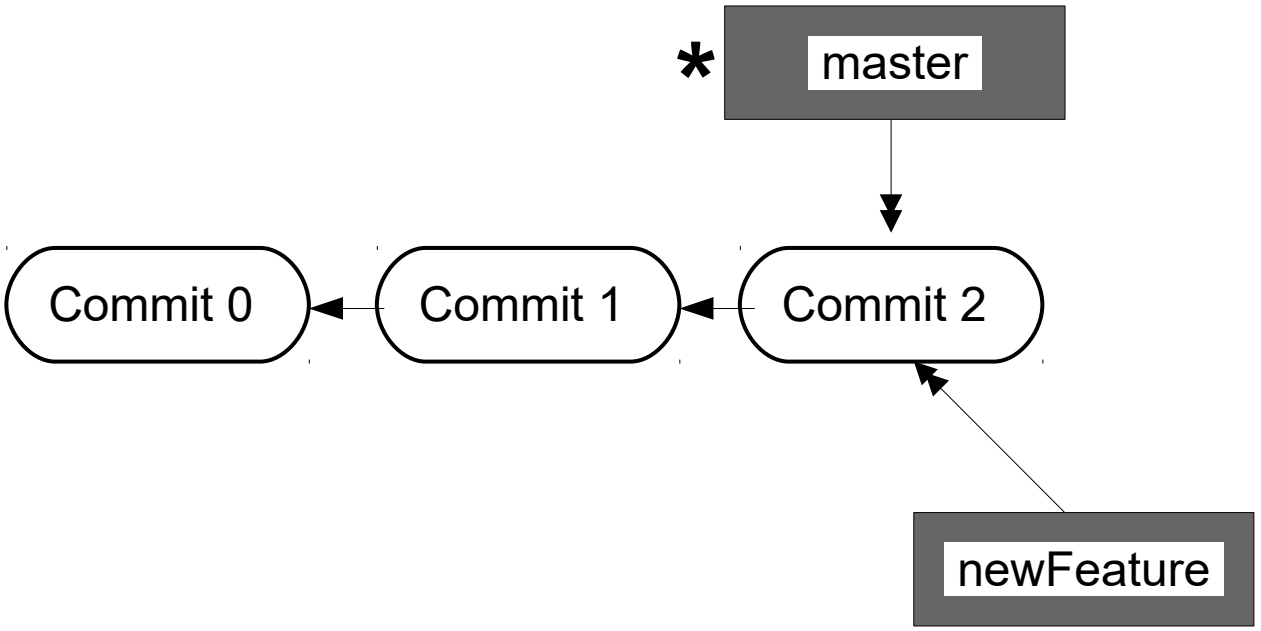
```
C:\Users\tombo\btm2017>git init
Initialized empty Git repository in C:/Users/tombo/btm2017/.git/
```


*
master
Commit 0
Commit 1

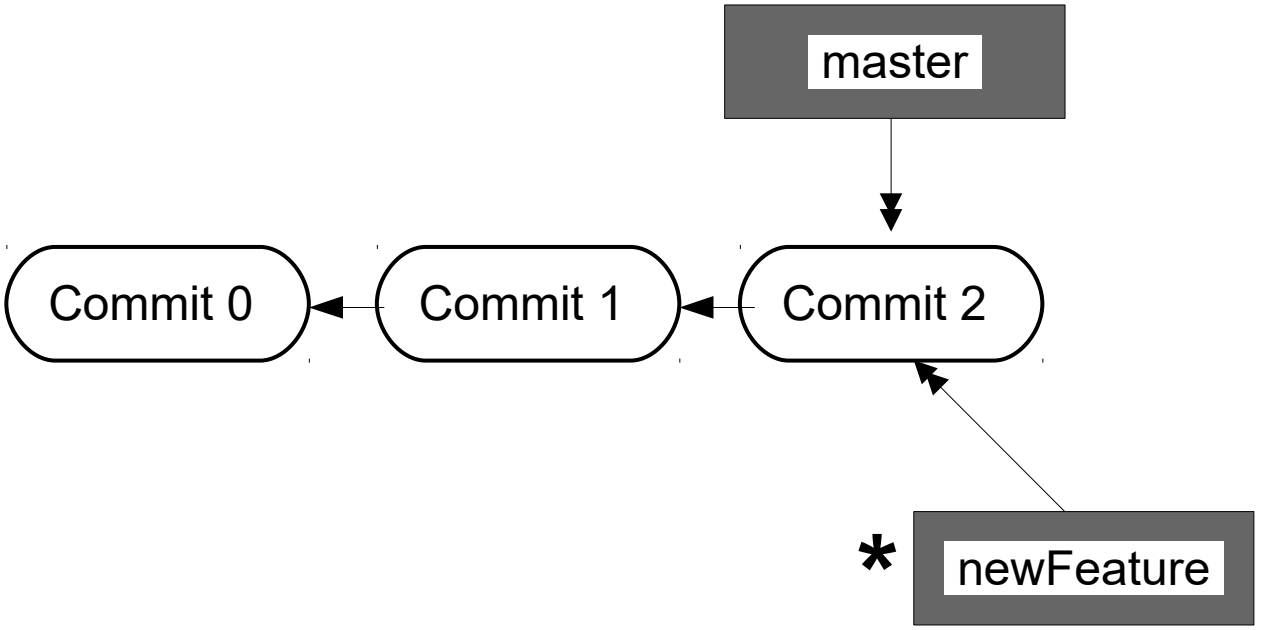
git commit -m ‘Modif message 2’



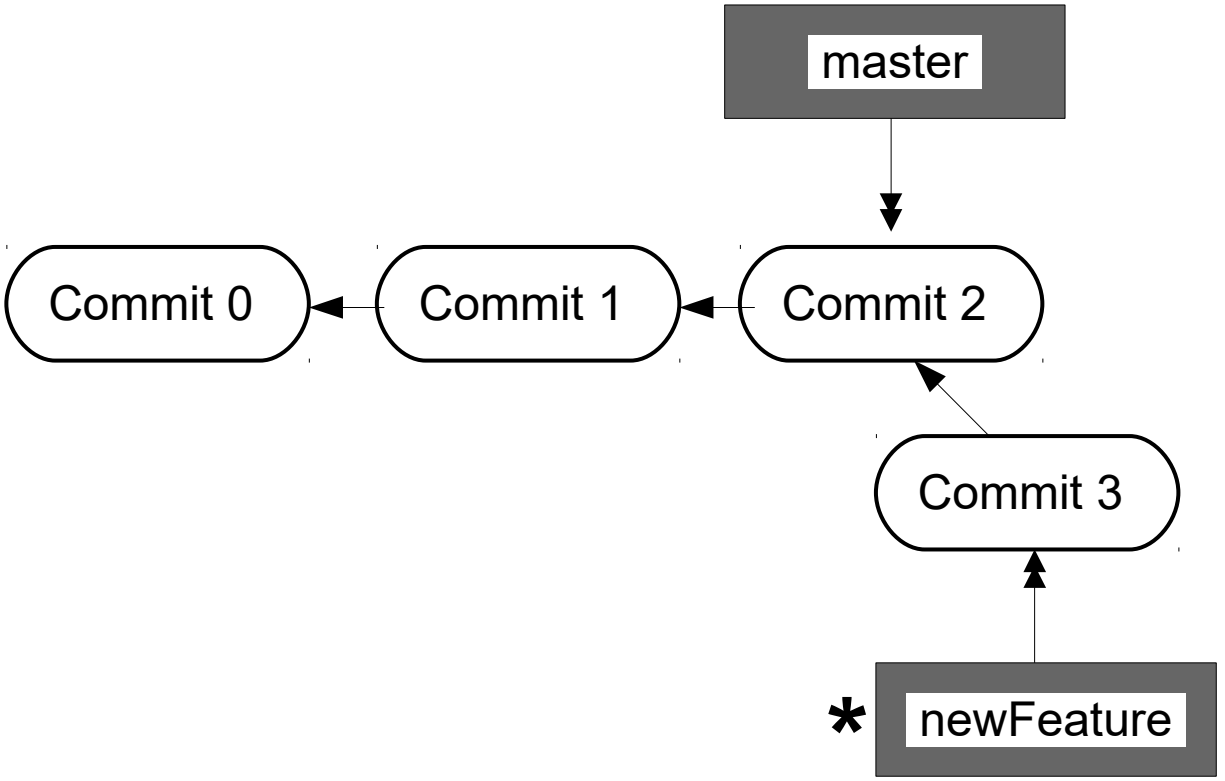
git branch newFeature



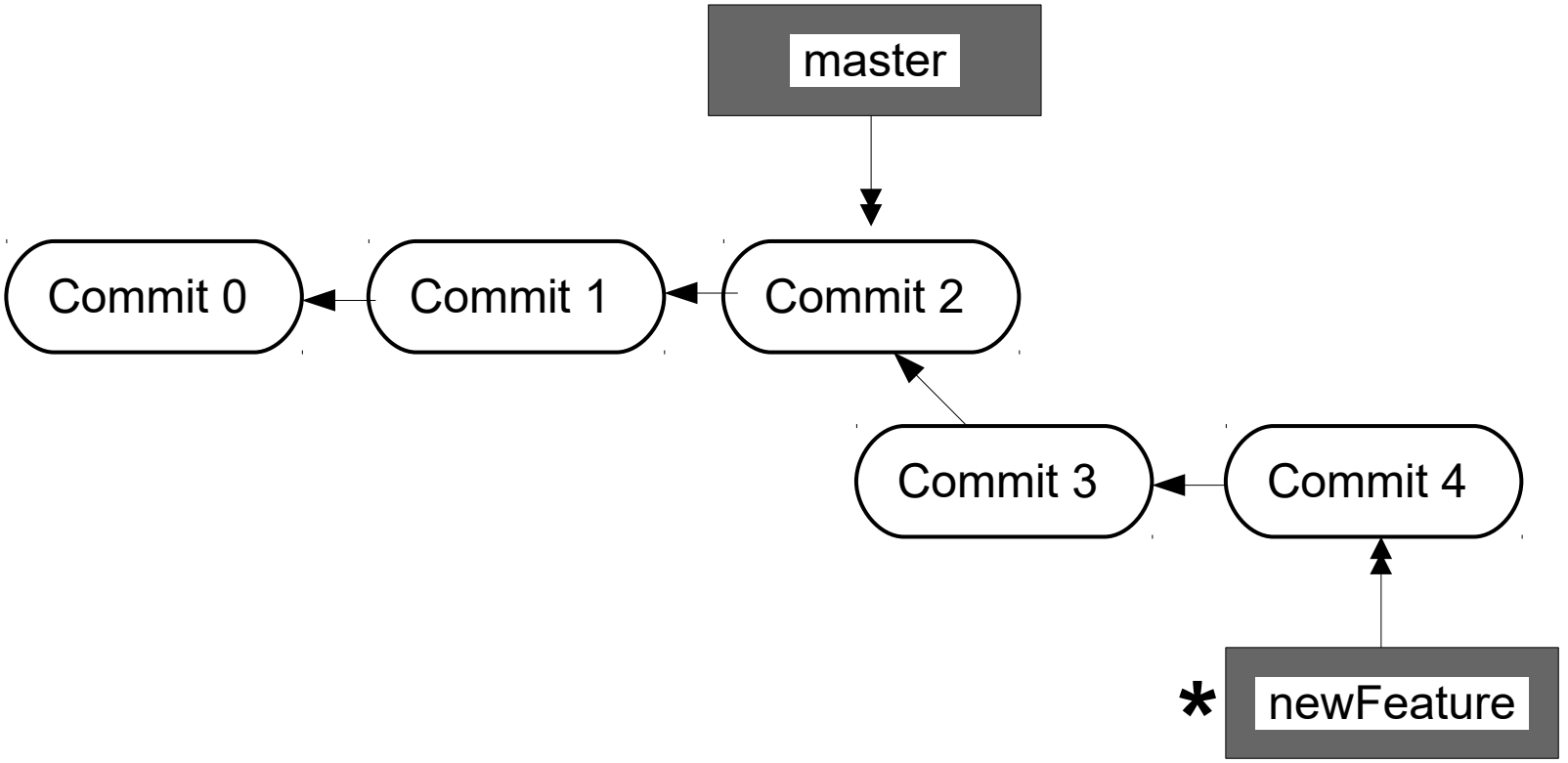
git checkout newFeature



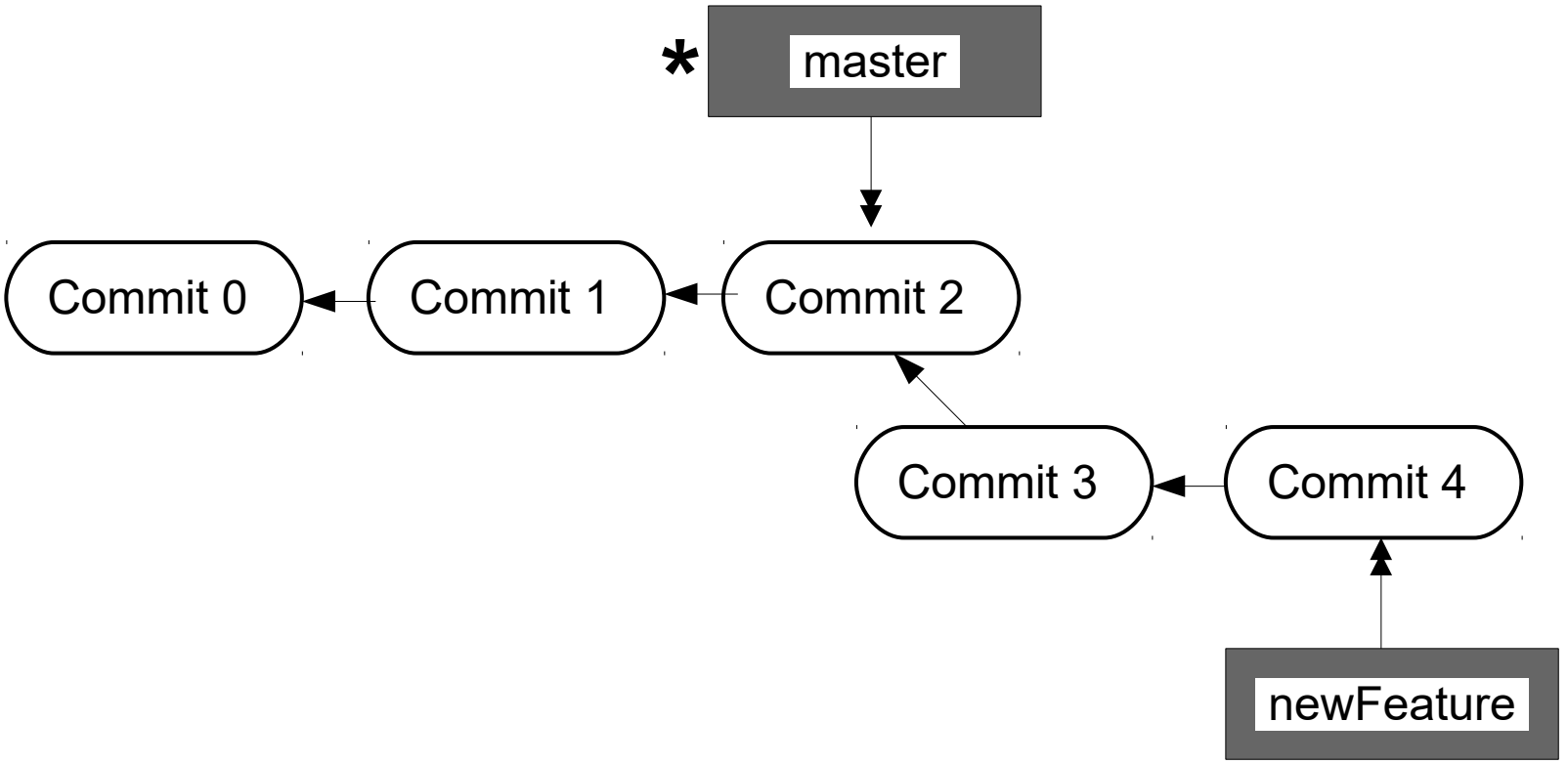
git commit -m ‘Modif message 3’



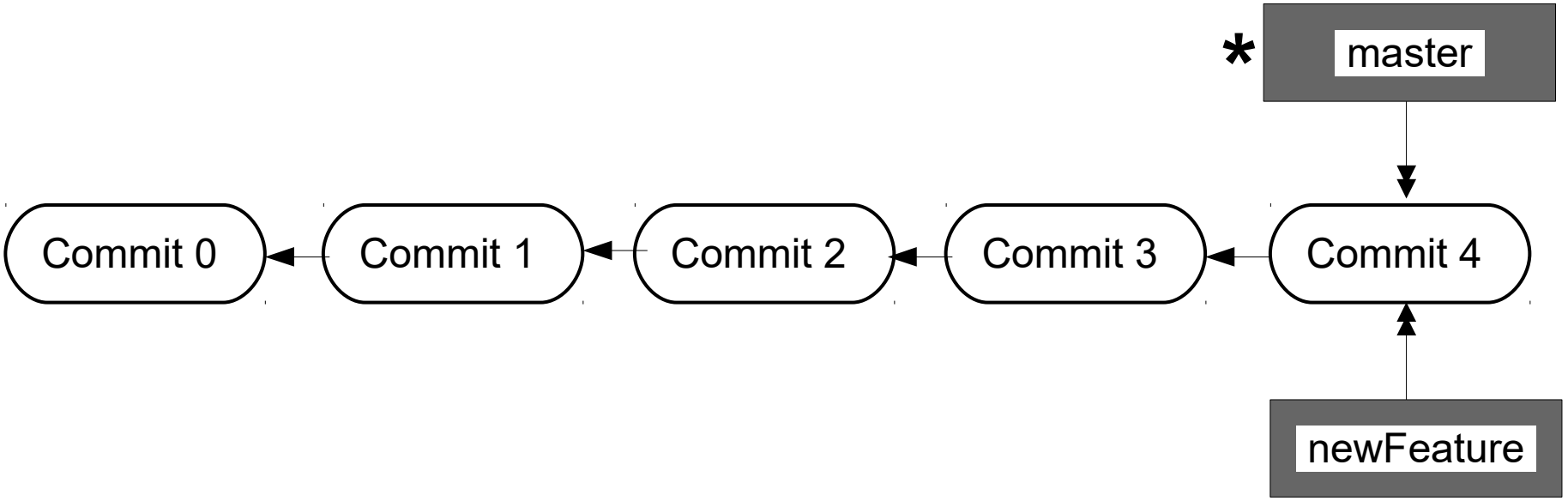
git commit -m ‘Modif message 4’



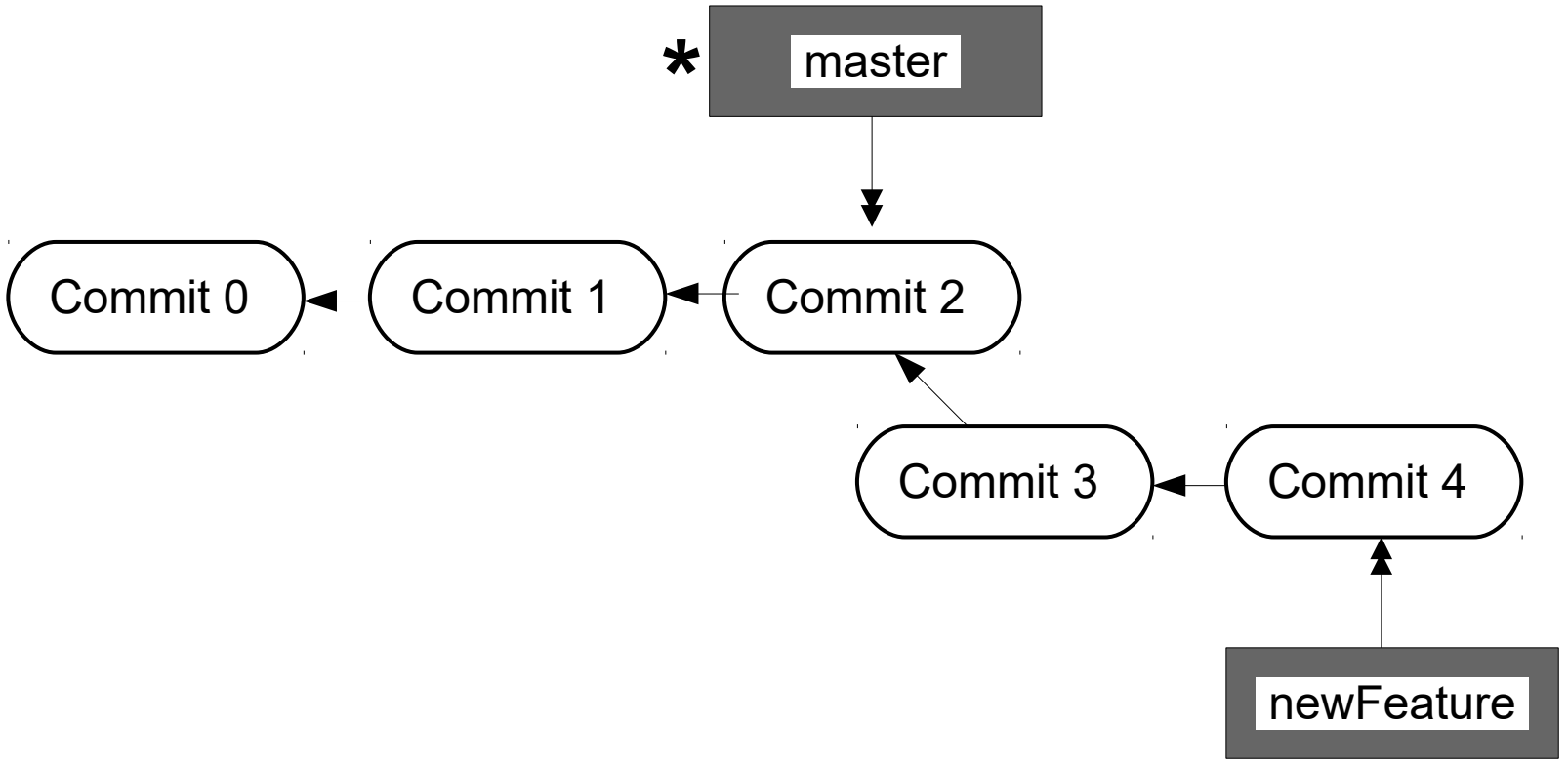
git checkout master



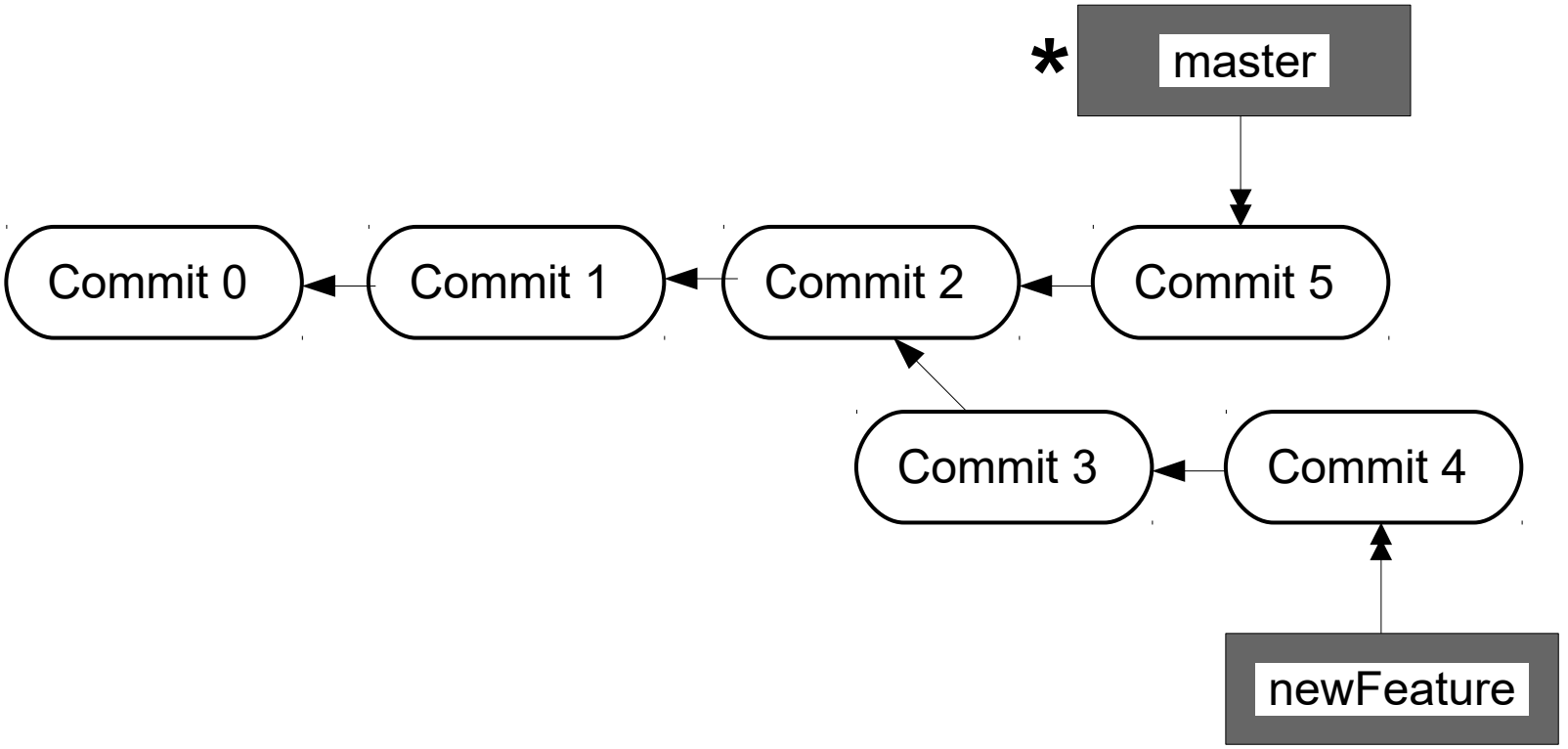
git merge newFeature



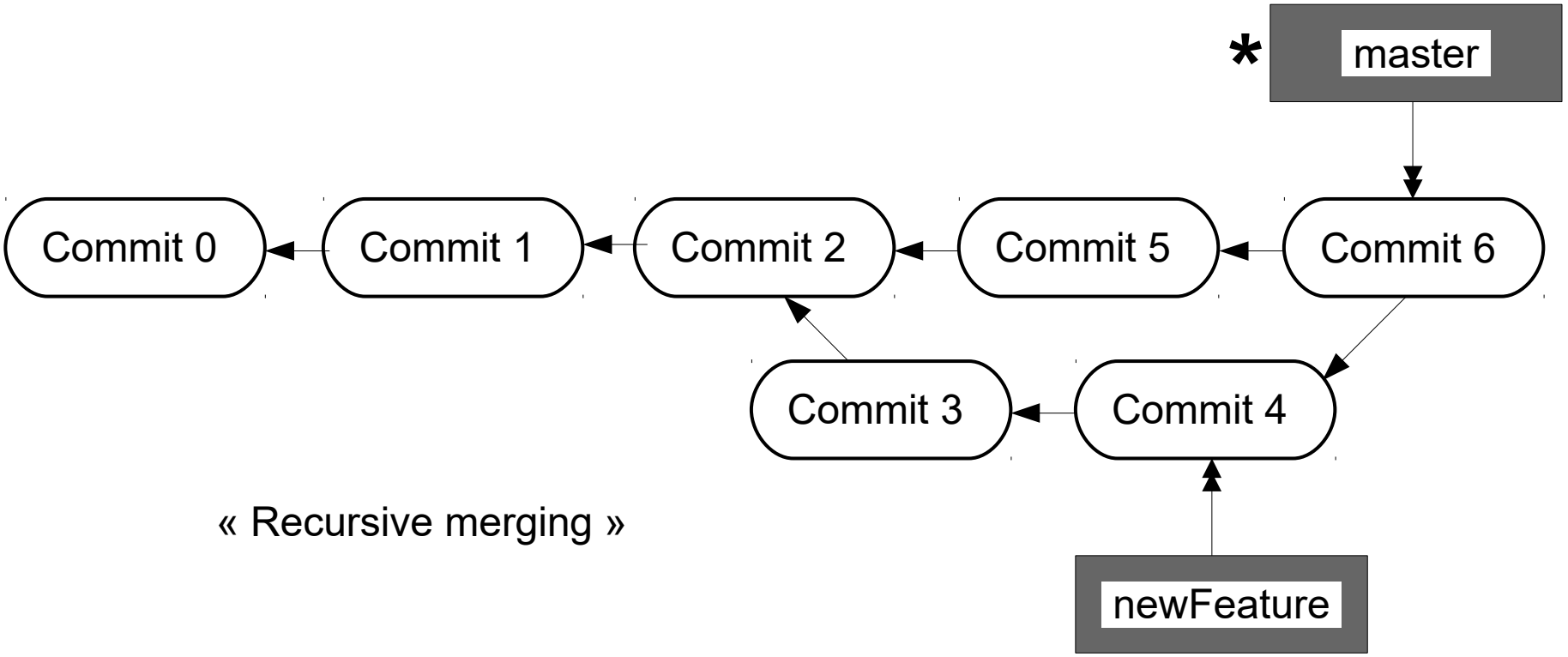
« Fastforward merging »


*
master
Commit 0
Commit 1
Commit 2
Commit 3
Commit 4
newFeature

git commit -m ‘Modif message 5’



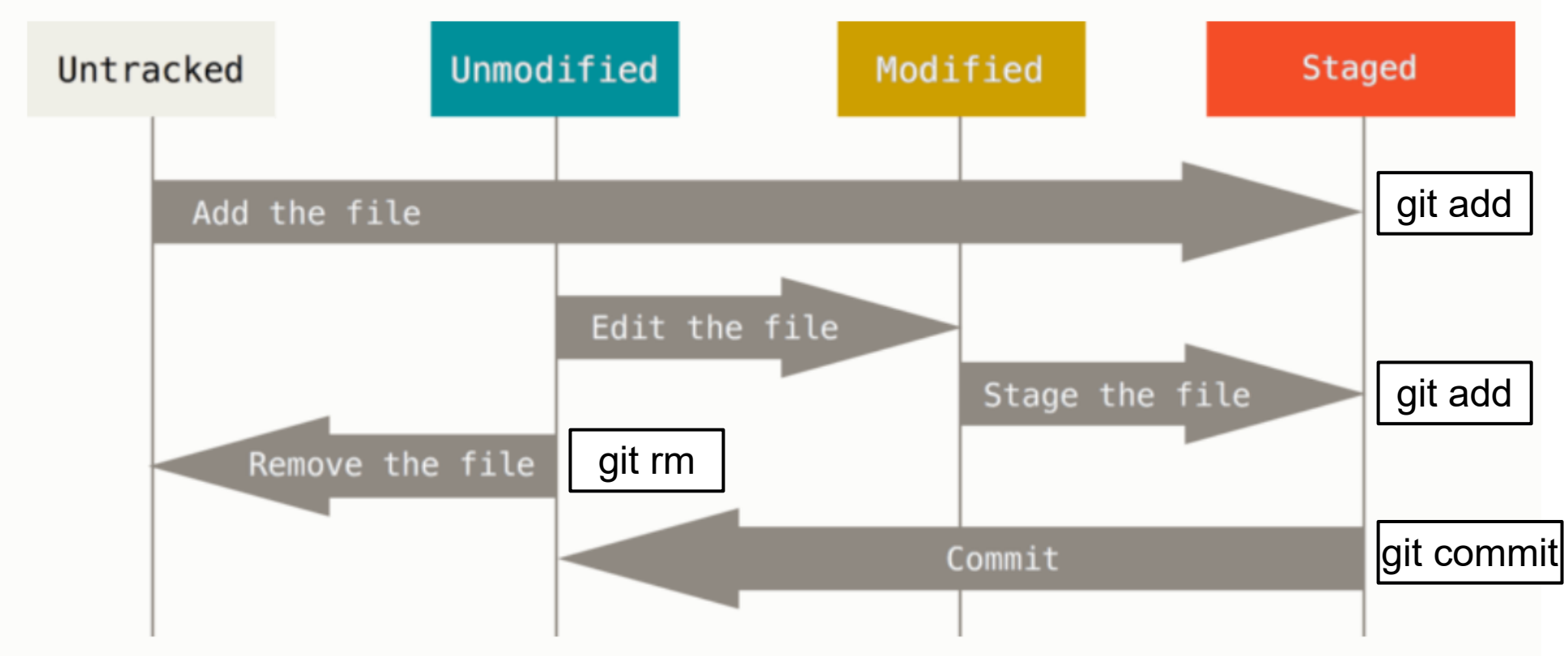
git merge newFeature



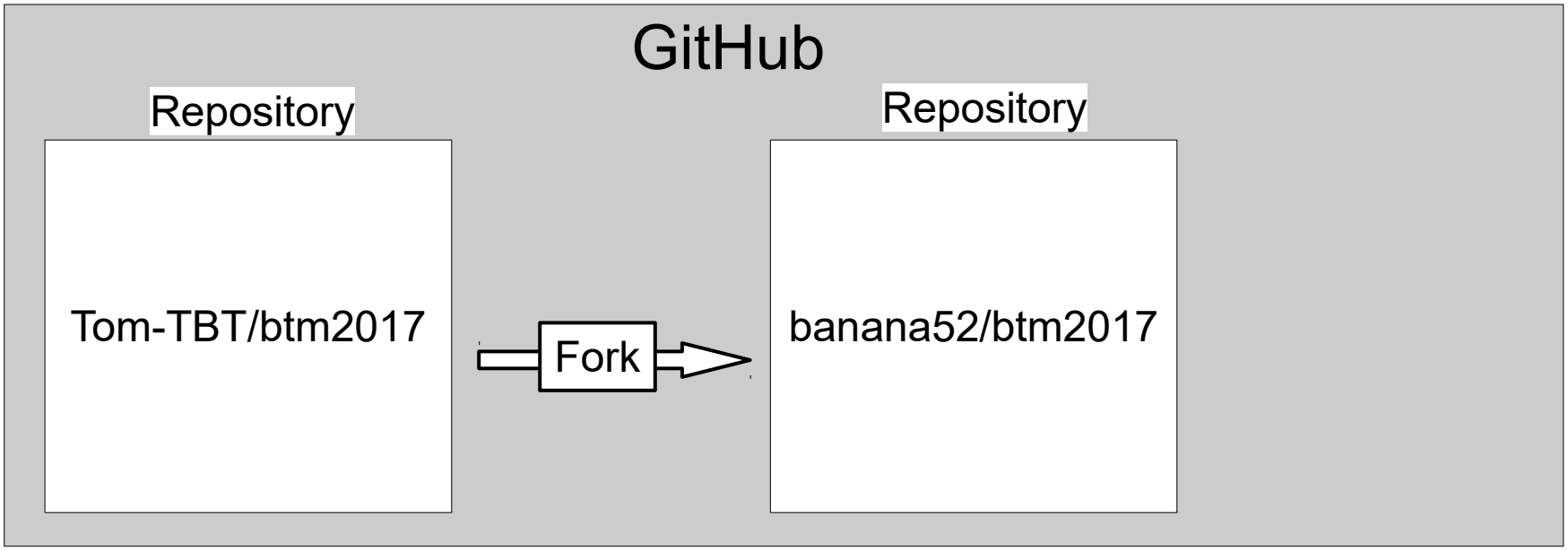
« Recursive merging »

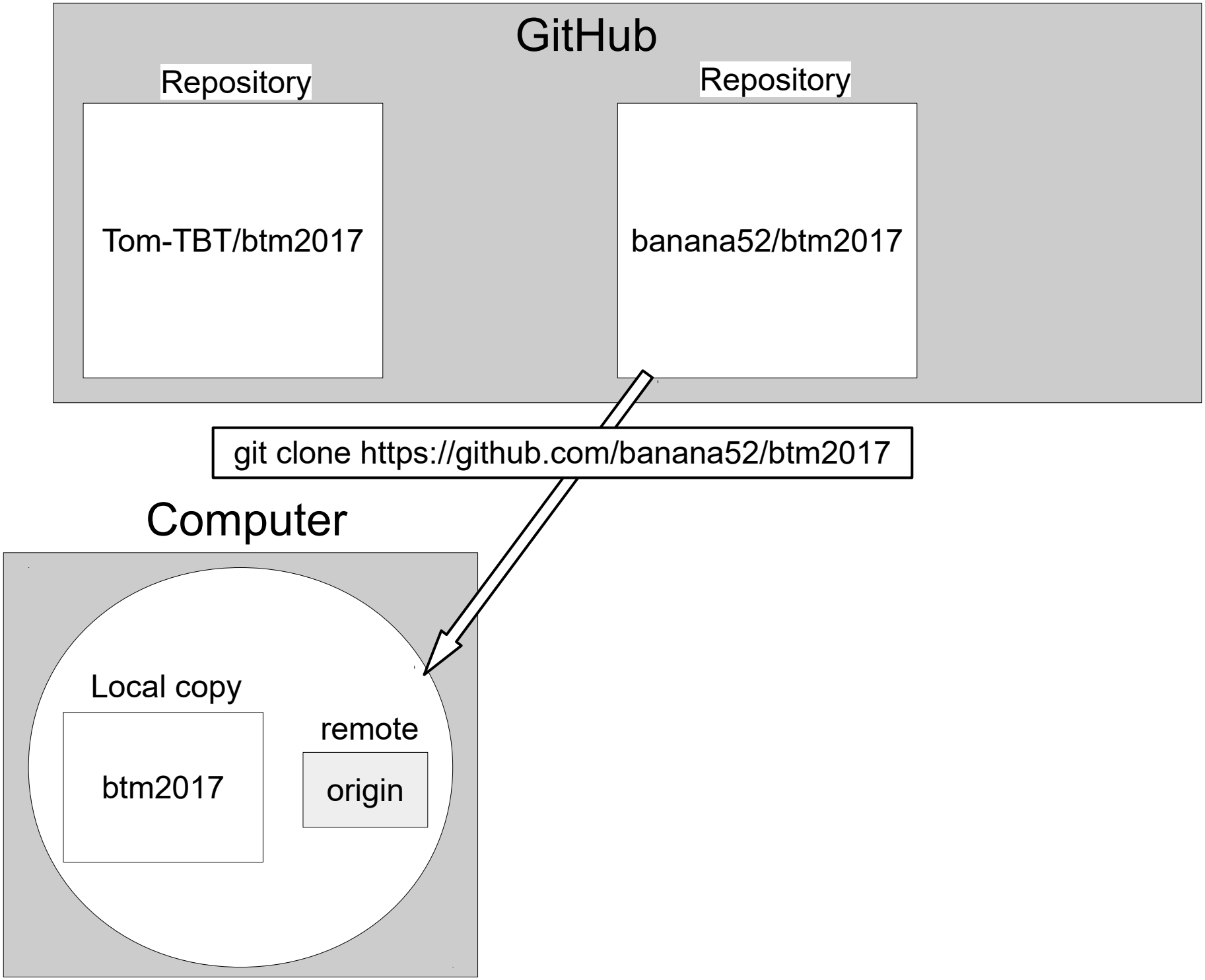
No git

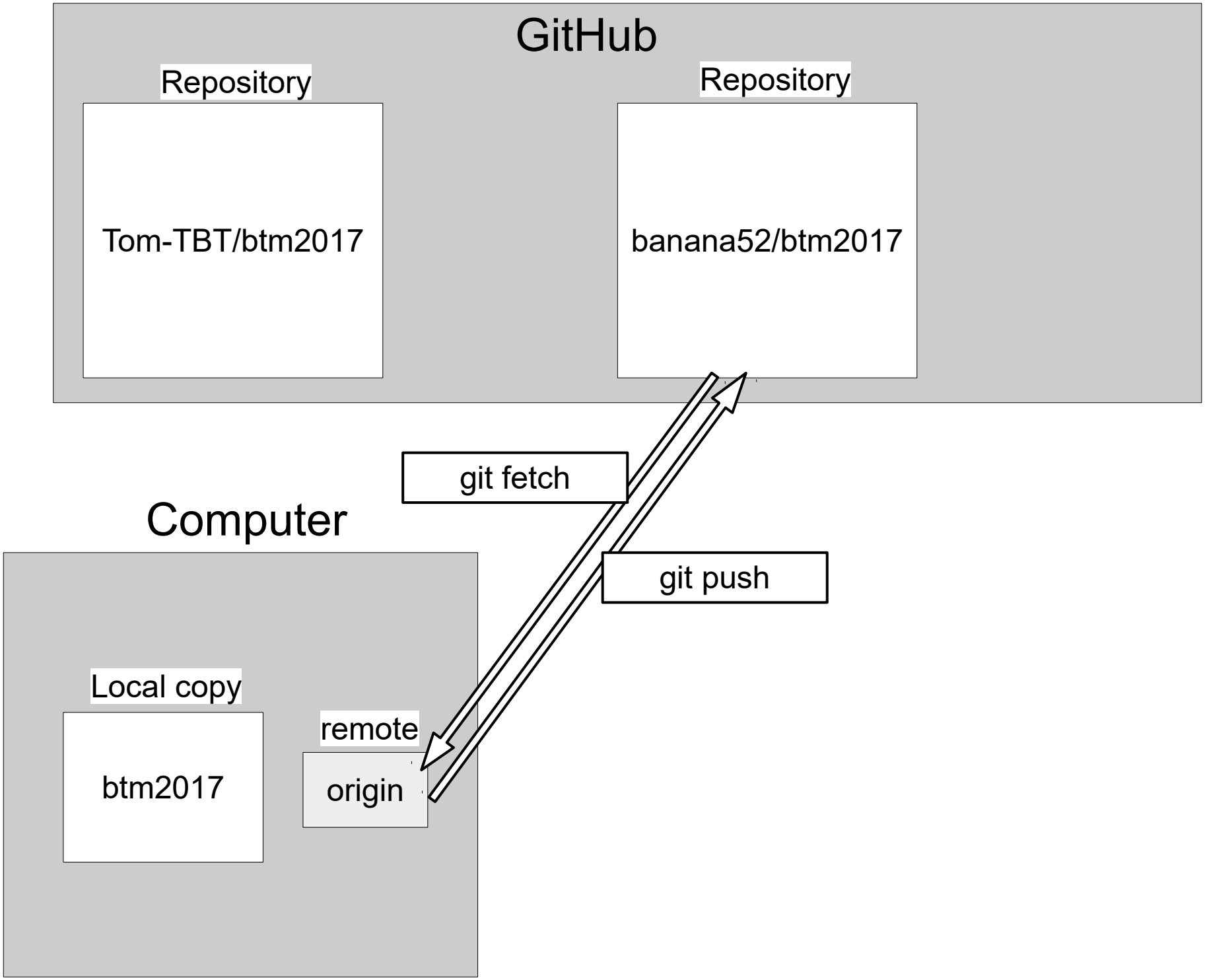
git



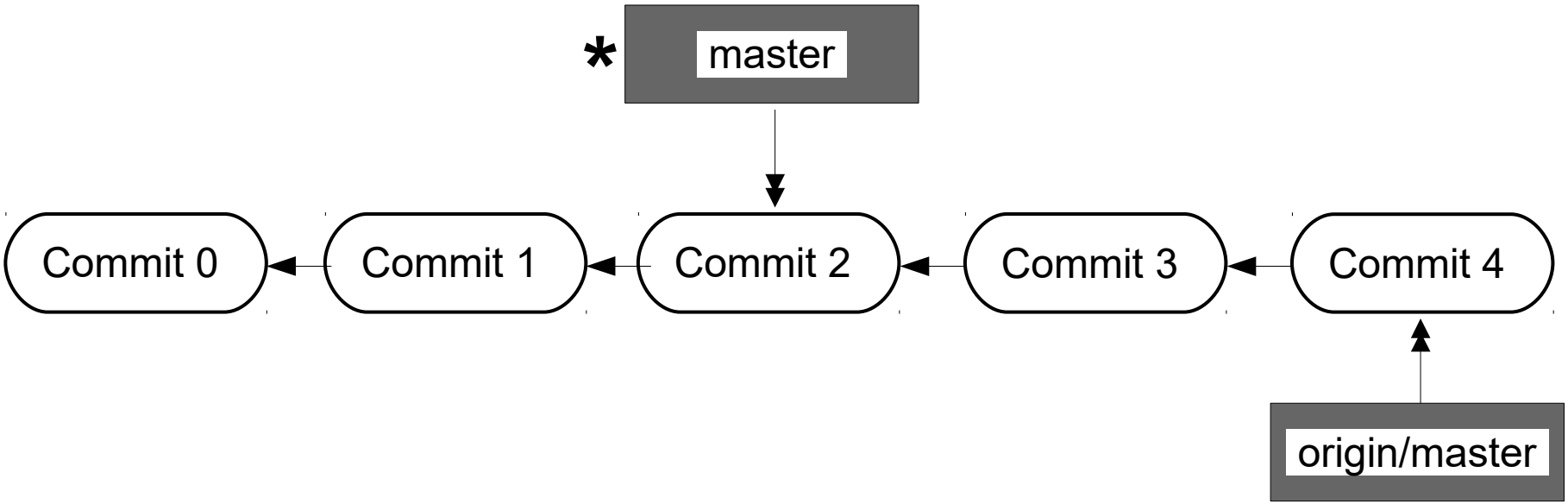
The life cycle of your files Git documentation


GitHub
Repository
Repository
Tom-TBT/btm2017
Fork
banana52/btm2017

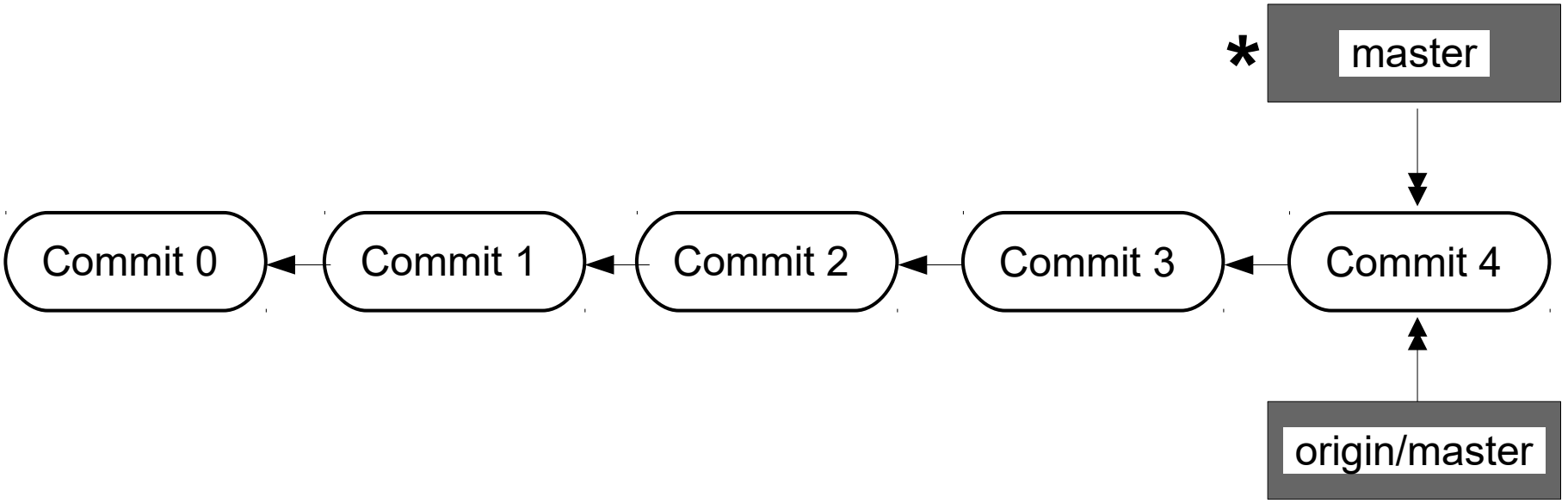

GitHub
Repository
Tom-TBT/btm2017
Repository
banana52/btm2017
git clone https://github.com/banana52/btm2017
Computer
Local copy
btm2017
remote
origin


GitHub
Repository
Tom-TBT/btm2017
Repository
banana52/btm2017
git fetch
git push
Computer
Local copy
btm2017
remote
origin

git fetch origin



git merge origin/master



Looking at git info :

git status
git log
git log --pretty=oneline –graph
git diff <file>
git remote show

Staging and commiting :

git add <file> >>>>Add <file> to stage
git rm <file> >>>> Remove <file> from tracking and delete it
git mv <orig> <dest> >>>> Move <orig> to <dest> (useful for renaming)
git commit -m "Don’t forget the message" >>>> Commit the staged changes

Playing with branches :

git branch <branch> >>>> Create a branch named <branch>
git checkout <branch> >>>> Change current branch to <branch>
git merge <branch> >>>> Merge <branch> onto the current branch
git fetch <remote> >>>> Ask to <remote> to import online changes (but don’t merge
→ merge it with git merge <remote/branch>)
git push >>>> Push local changes to online repository (you need permission)

Undos :

git commit –ammend >>>> Add staged modif to the last commit (if you forgot something)
git --reset HEAD <file> >>>> Unstage a file
For other undos, it can be dangerous. Ask google.

Before starting

- Same file can be modified by several person at a time
- What if the file is edited at the same place at the same time ?
- What modification is saved ?

  -> Owner of the file will need to solve conflicts

Let's create something collaborative !

Extra links:

https://try.github.io

https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

https://www.google.it/search?q=git