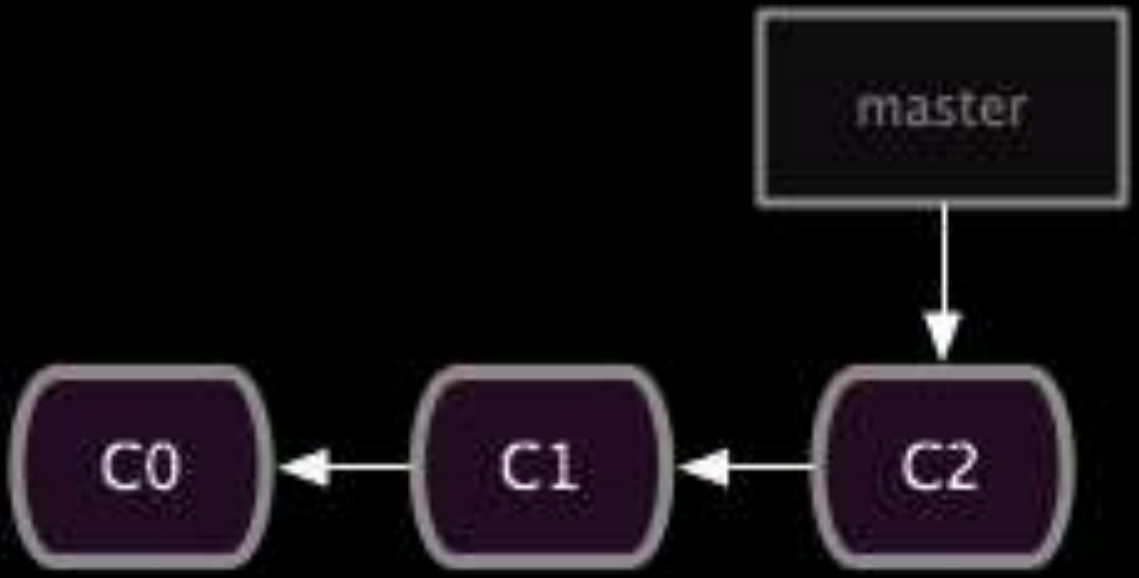
Version Control with Git

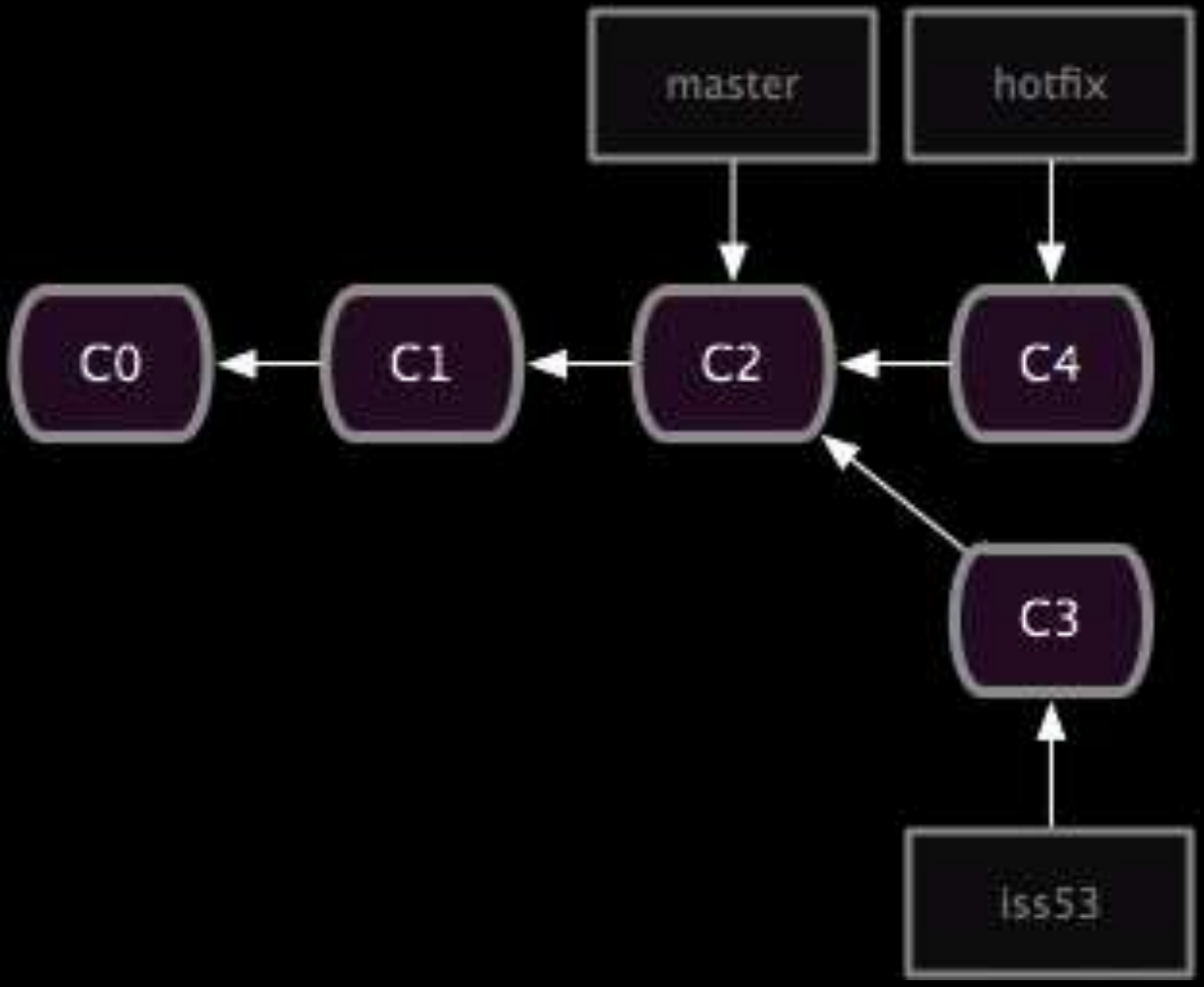
Why track/manage revisions?

Backup: Undo or refer to old stuff



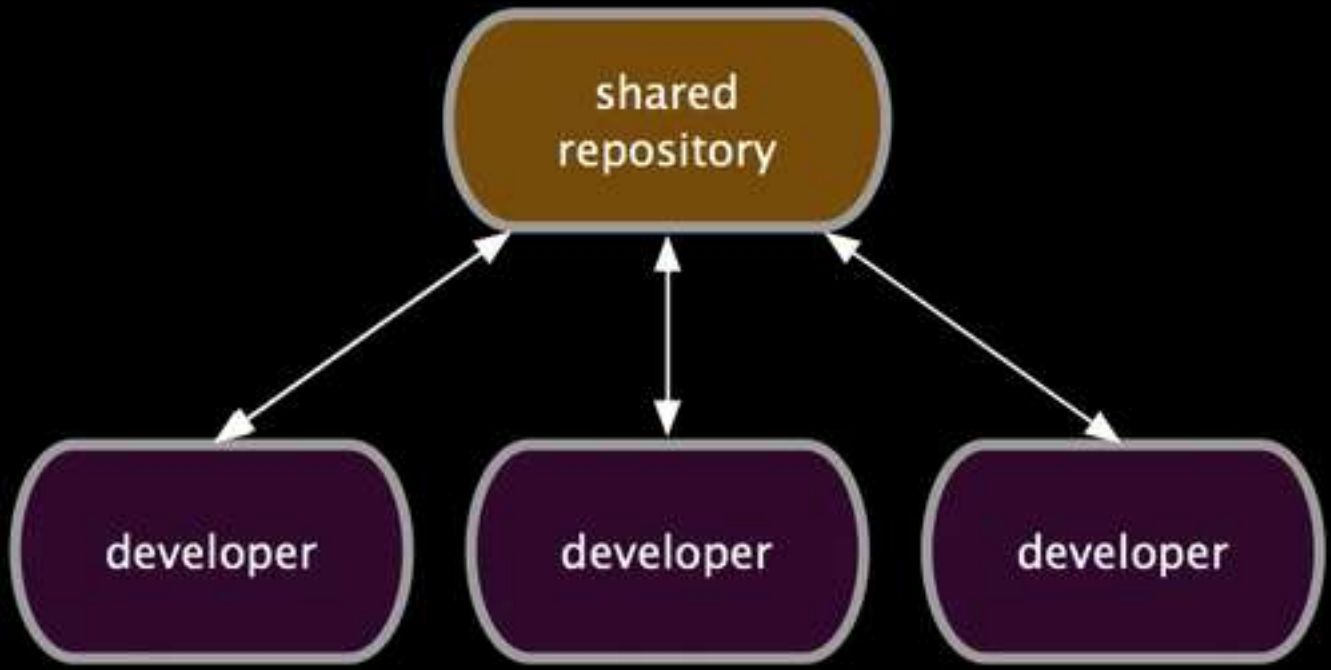
http://git-scm.com/book/en/Git-Branching-Basic-Branching-and-Merging

Branch: Maintain old release while working on new



http://git-scm.com/book/en/Git-Branching-Basic-Branching-and-Merging

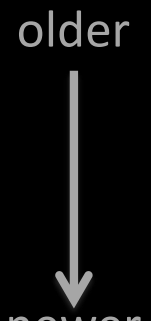
Collaborate: Work in parallel with teammates



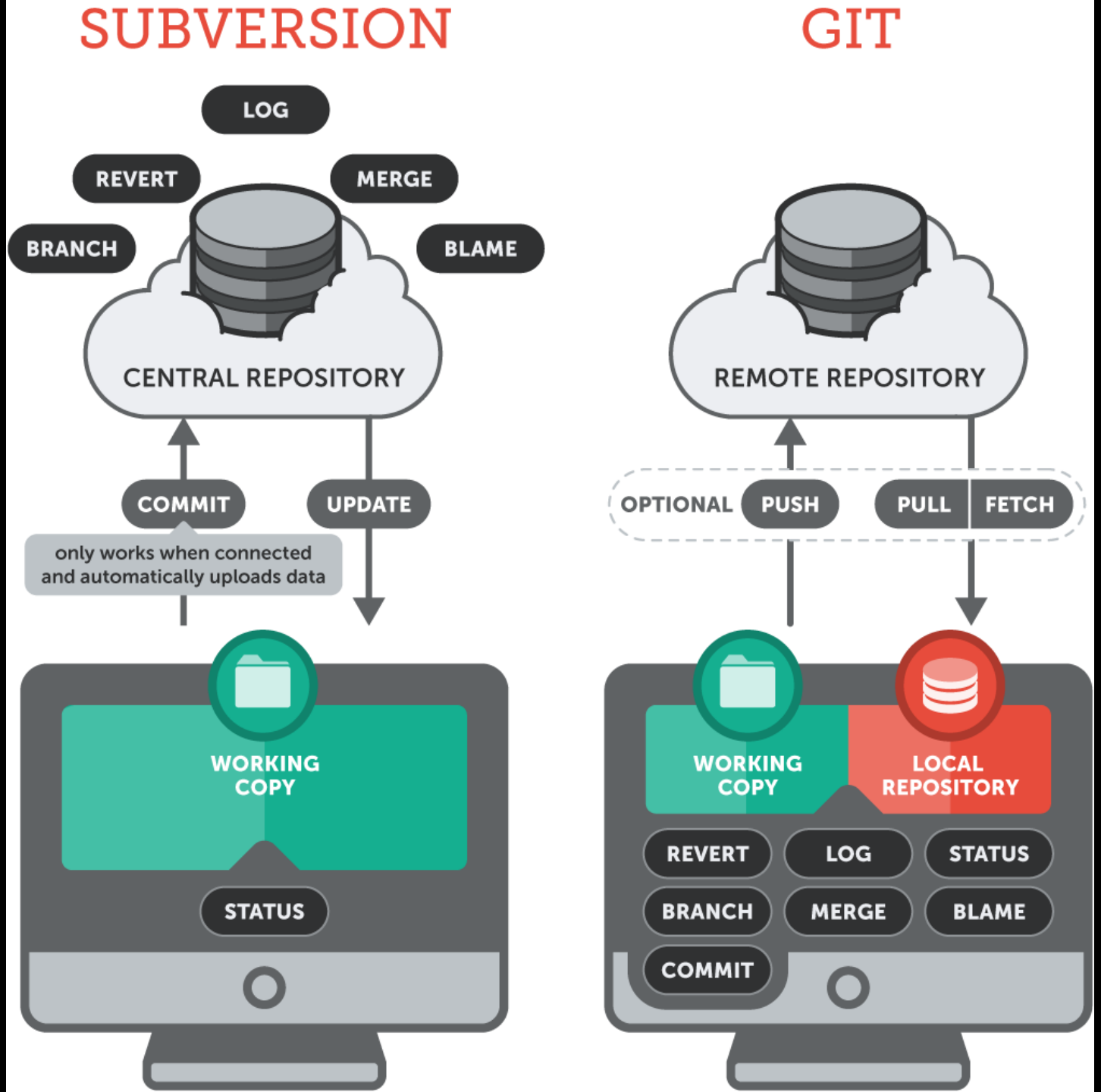
http://git-scm.com/book/en/Distributed-Git-Distributed-Workflows

Version Control Systems (VCSs)

- Help you track/manage/distribute revisions
- Standard in modern development
- Examples:
  - Revision Control System (RCS)
  - Concurrent Versions System (CVS)
  - Subversion (SVN)
  - Git



Our focus


SUBVERSION
LOG
REVERT
MERGE
BRANCH
BLAME
CENTRAL REPOSITORY
COMMIT
UPDATE
only works when connected
and automatically uploads data
WORKING
COPY
STATUS
GIT
REMOTE REPOSITORY
OPTIONAL
PUSH
PULL
FETCH
WORKING
COPY
LOCAL
REPOSITORY
REVERT
LOG
STATUS
BRANCH
MERGE
BLAME
COMMIT

https://git-scm.com

Secure | https://git-scm.com/downloads

git --distributed-even-if-your-workflow-isnt

Search entire site...

About
Documentation
Blog
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Downloads

Mac OS X Windows

Linux Solaris

Older releases are available and the Git source repository is on GitHub.

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

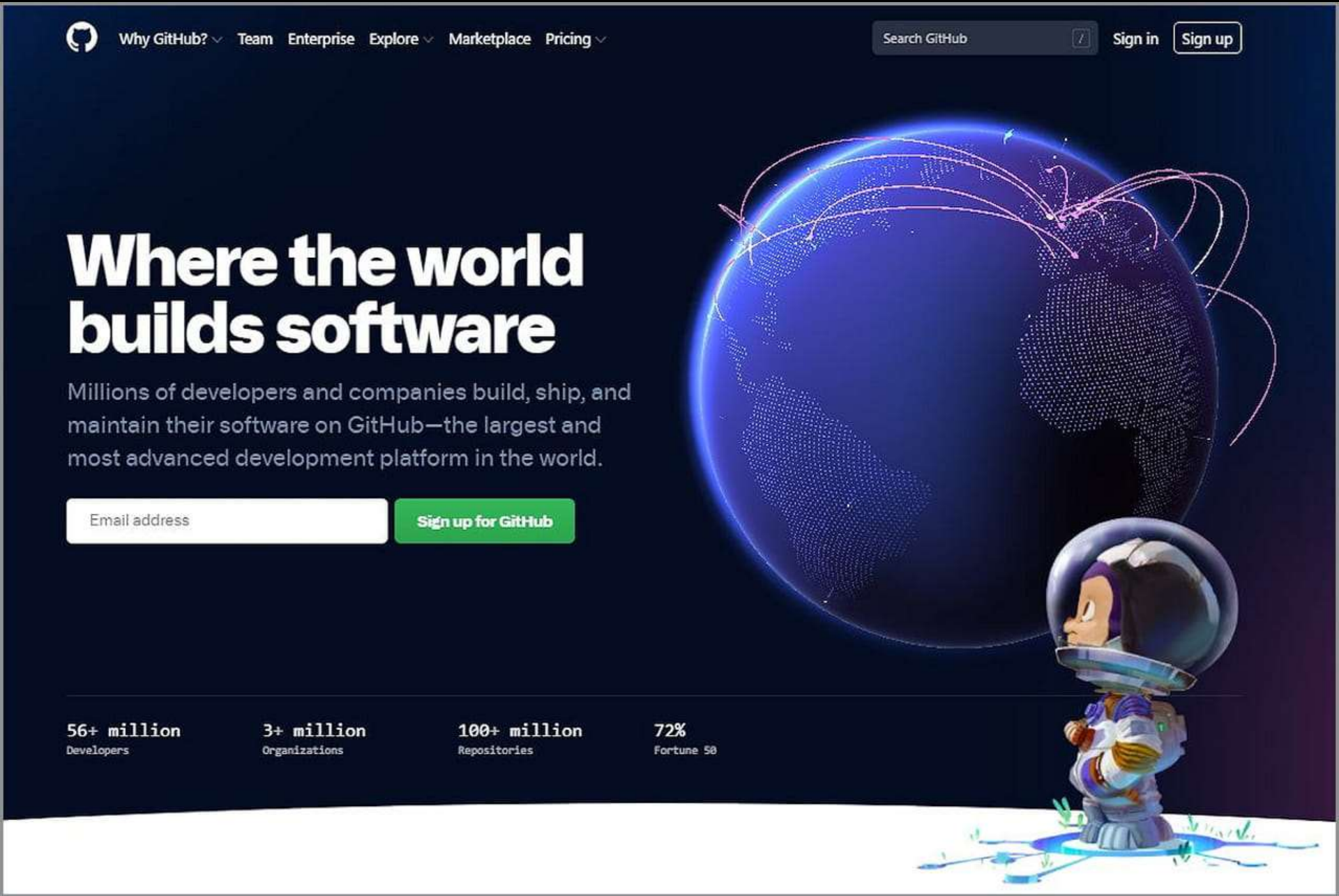
View GUI Clients →

Logos

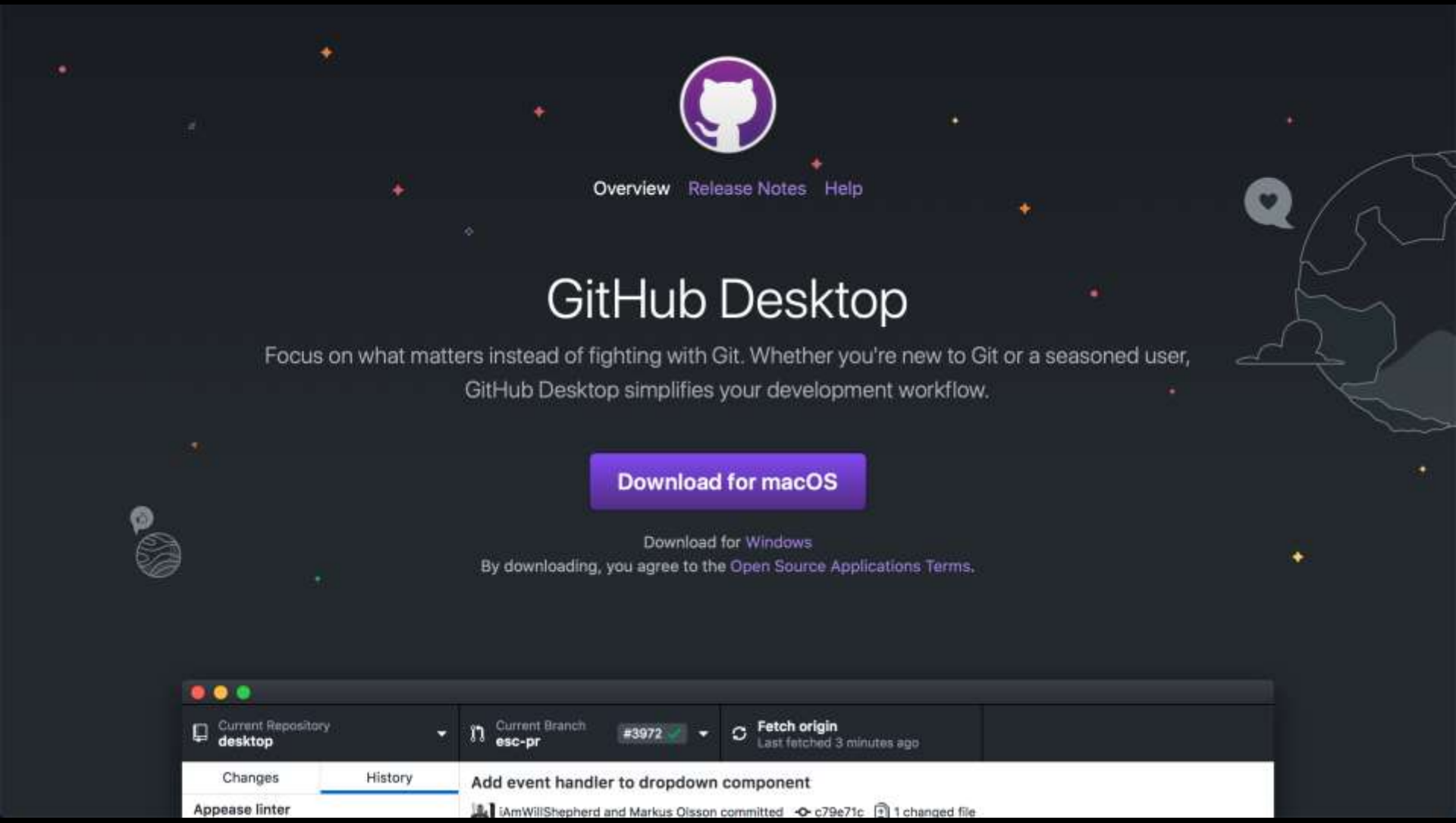
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

https://github.com



https://desktop.github.com



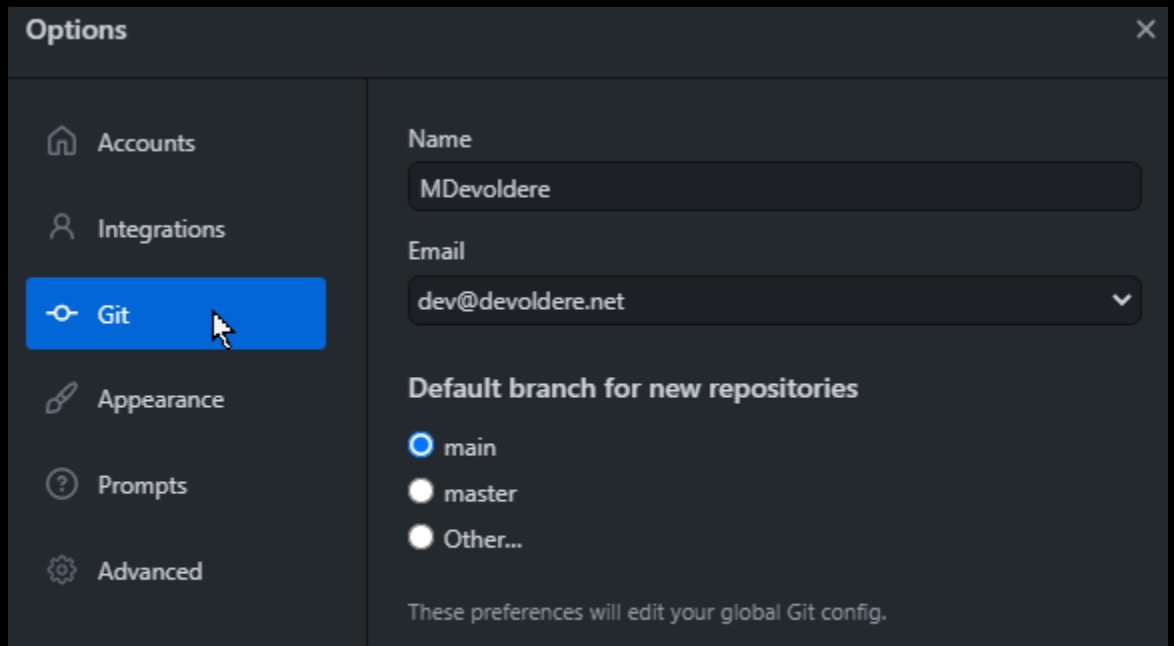
Configure your Git client

- Check config info:

```
$ git config --list
user.name=MDevoldere
user.email=mdevoldere@arfp.asso.fr
```

- Fix if necessary:

```
$ git config --global user.name "John Doe"
$ git config --global user.email jdoe@example.com
```

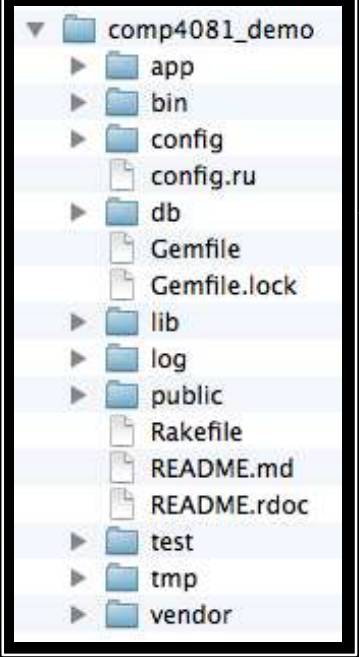


GitHub-User Perspective

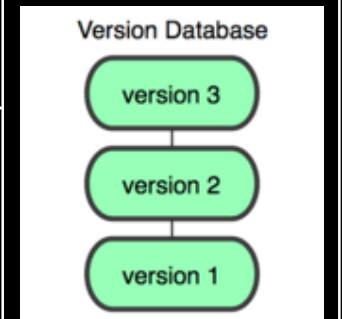
You

GitHub

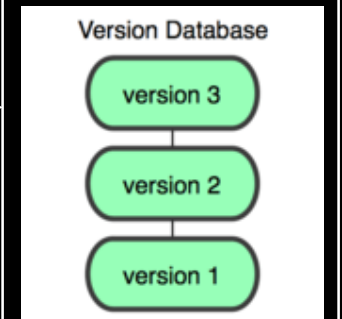
Working Dir



Local Repos



Remote Repos



Let’s begin with an example…

You

GitHub

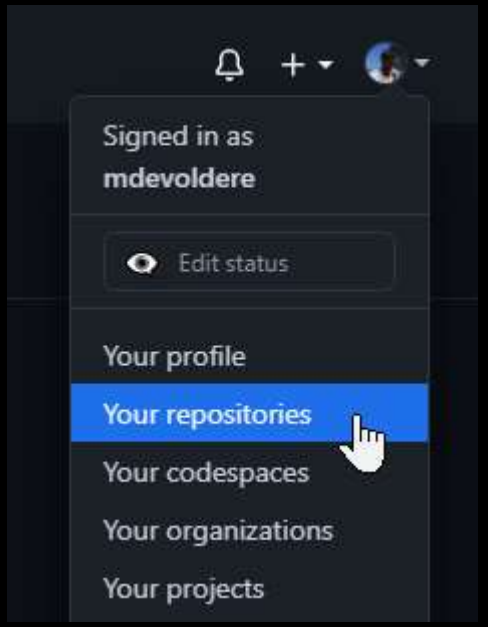
Log into GitHub and create a repos
(with add README option)

You

GitHub

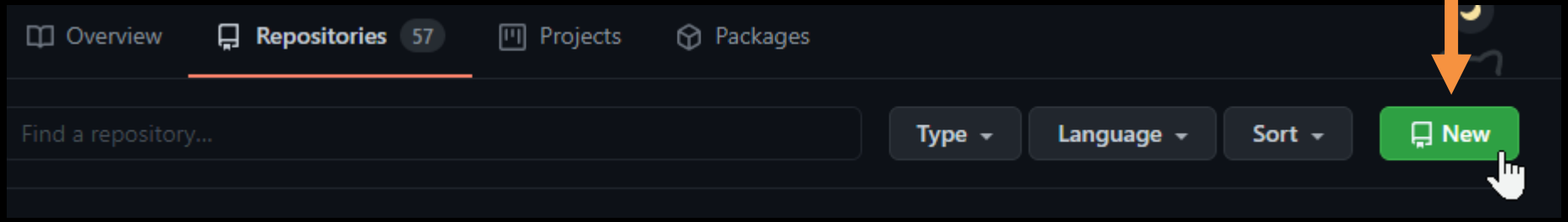
Remote Repos

https://github.com



1. Go to your Repos page

2. On Repos page, click « New »



Repository template
Start your repository with a template repository's contents.

No template

Owner * Repository name *

mdevoldere / my-repo

Great repository names are short and memorable. Need inspiration? How about special-journey?

Description (optional)

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:
Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. Learn more.

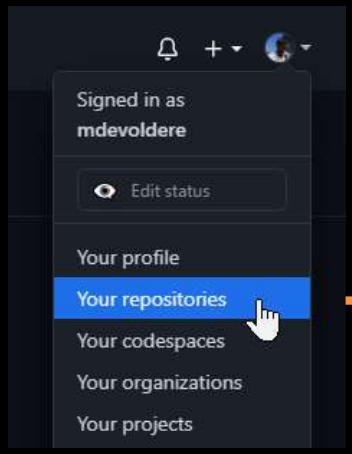
Add .gitignore
Choose which files not to track from a list of templates. Learn more.

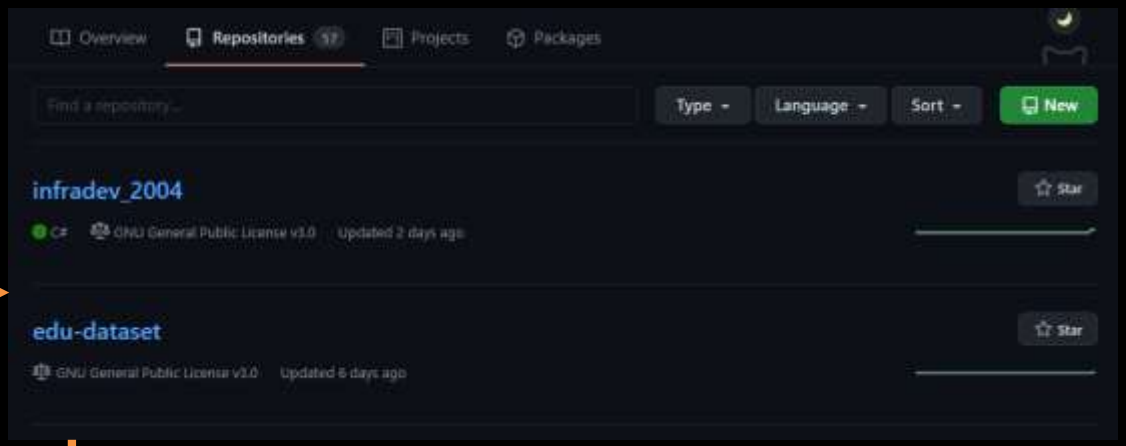
.gitignore template: VisualStudio

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

This will set `main` as the default branch. Change the default name in your settings.

Create repository





mdevoldere / edu-dataset

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

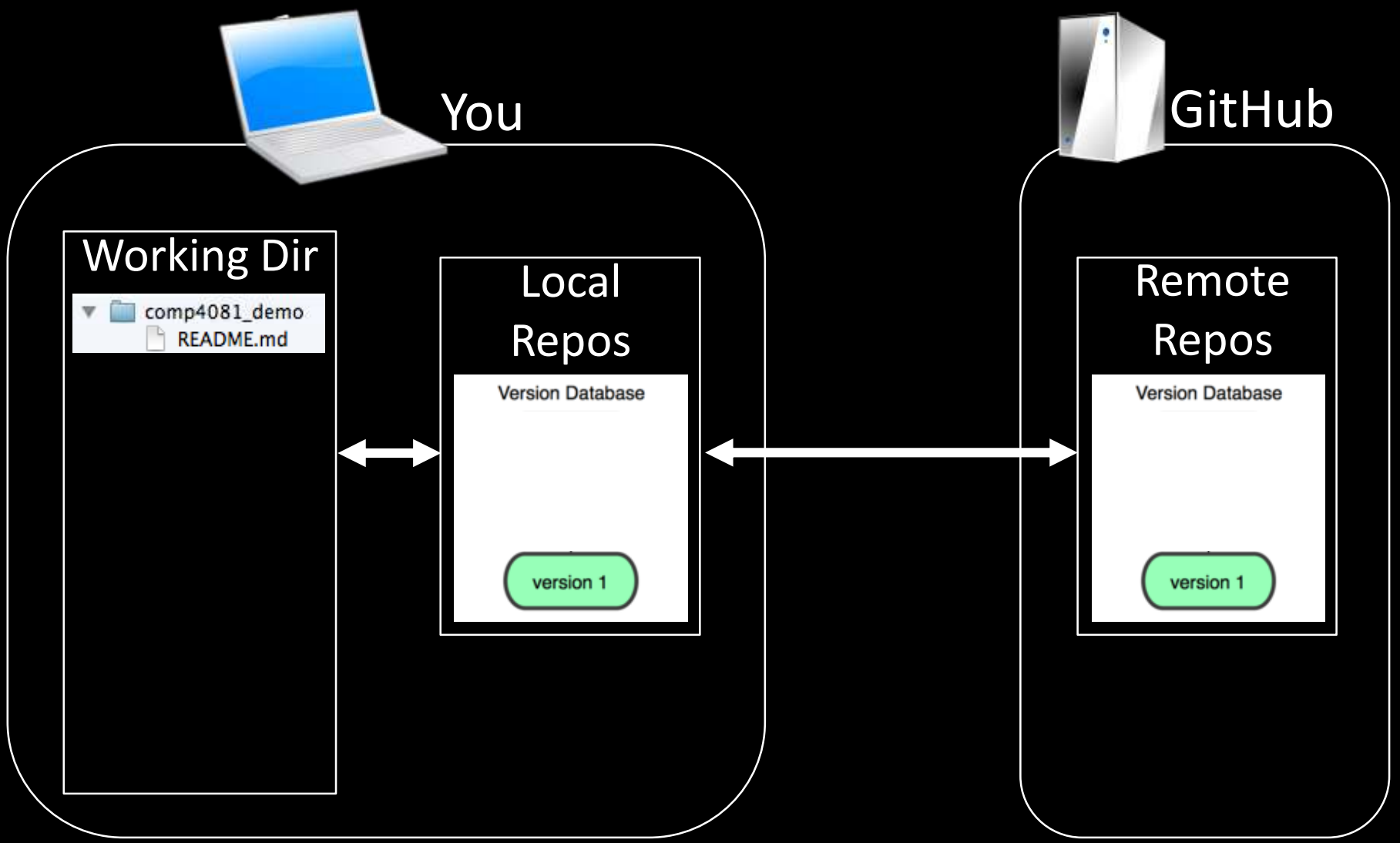
main | 3 branches | 0 tags | Go to file | Add file | Code

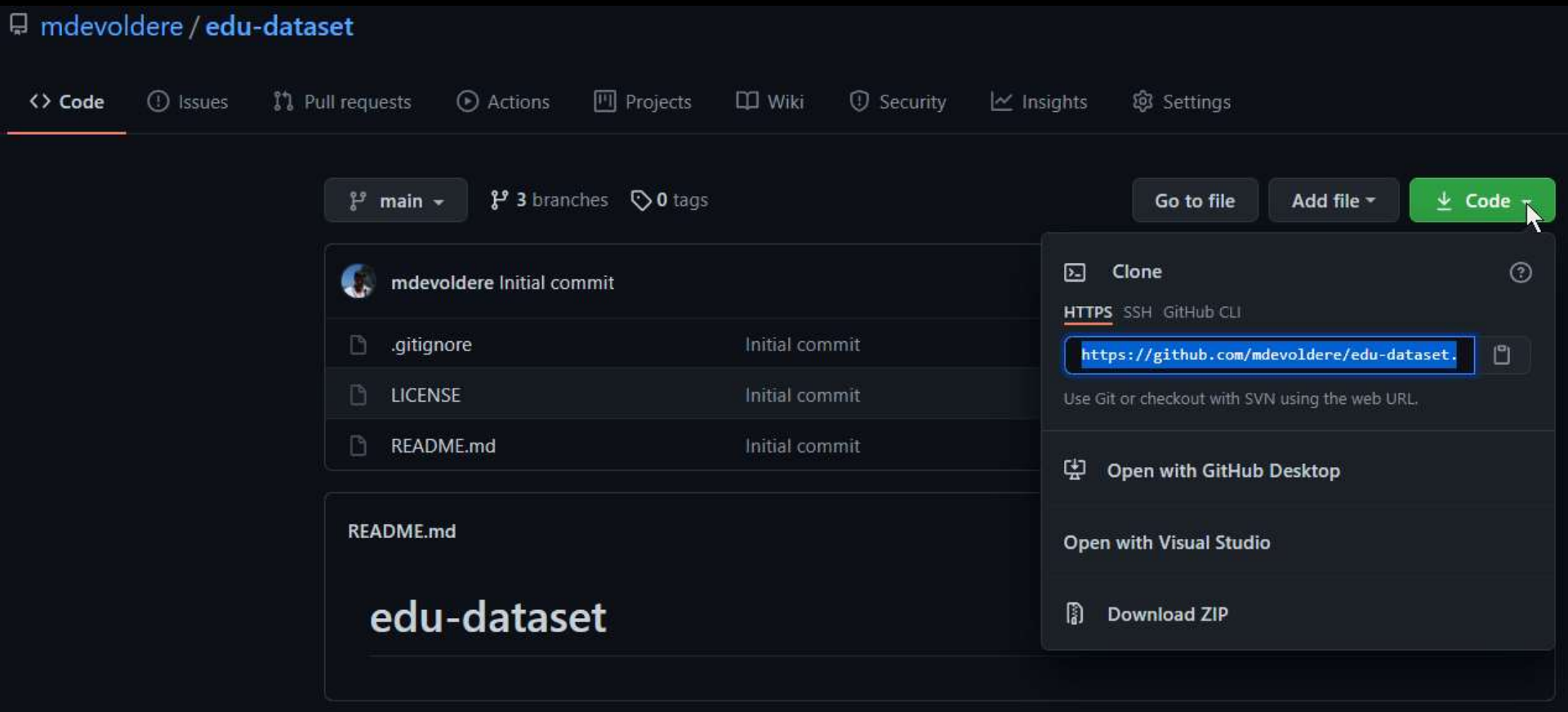
mdevoldere Initial commit	41960d5	6 days ago · 1 commit
.gitignore	Initial commit	6 days ago
LICENSE	Initial commit	6 days ago
README.md	Initial commit	6 days ago

README.md

edu-dataset

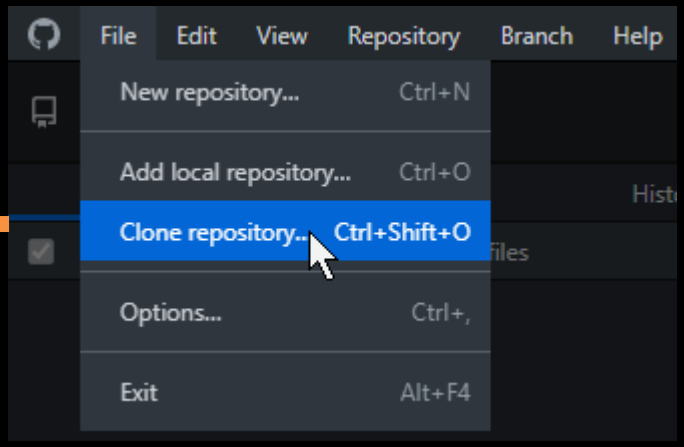
```
$ git clone https://github.com/arfp/comp4081_demo.git
```

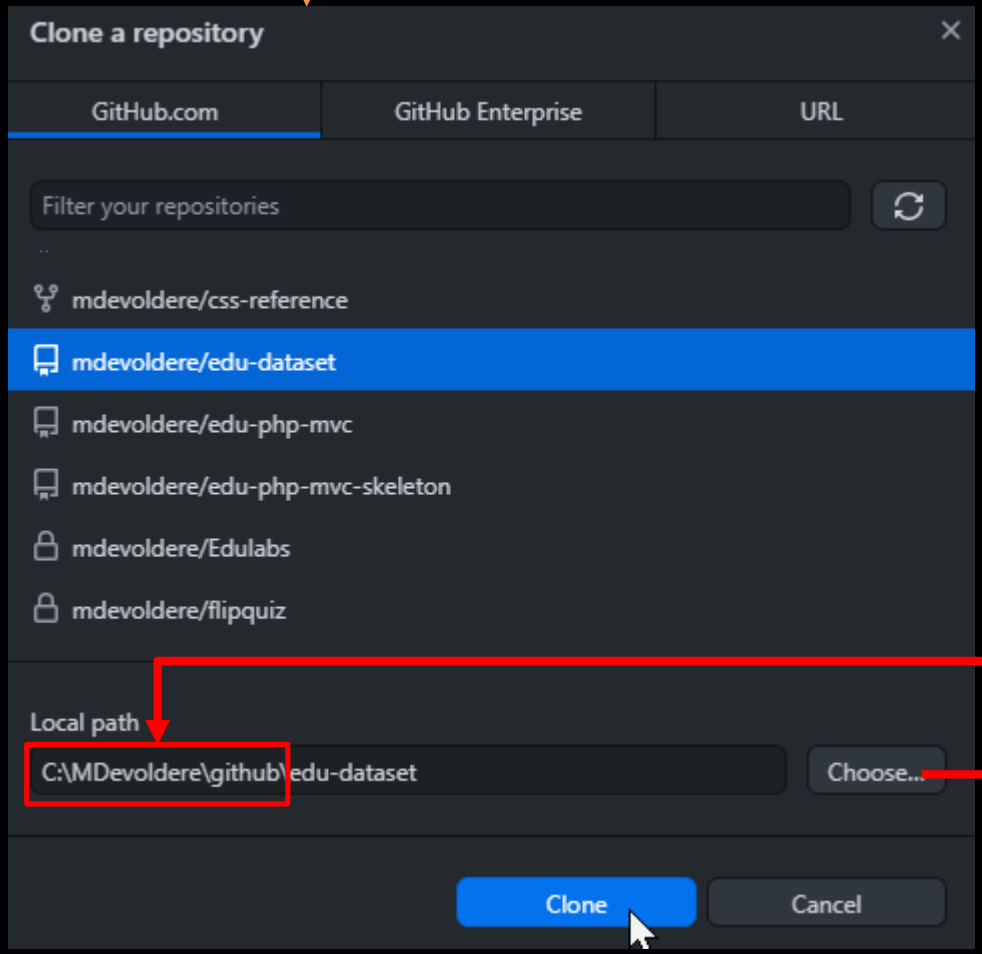


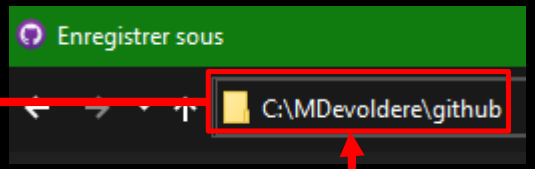


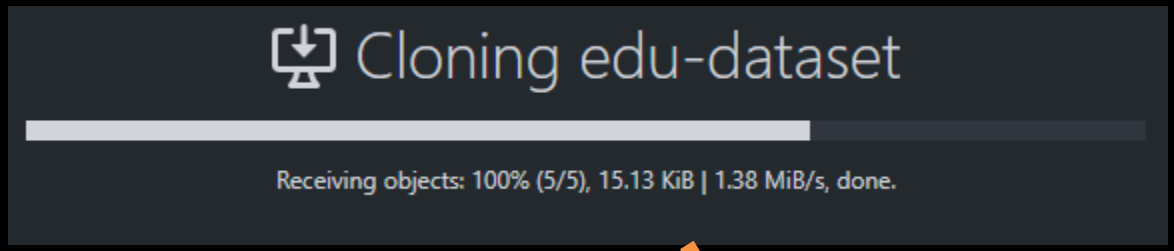

```
git clone https://github.com/mdevoldere/edu-dataset.git
```

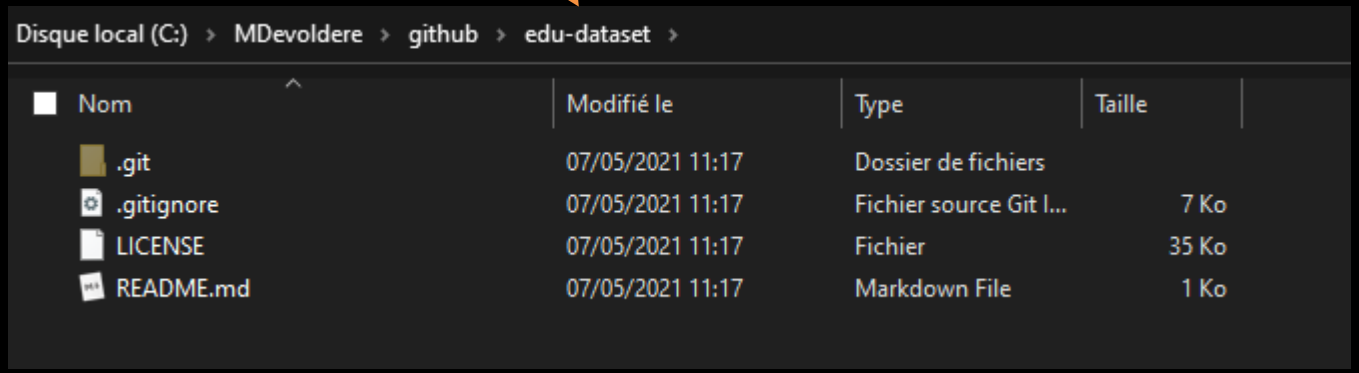
GitHub Desktop



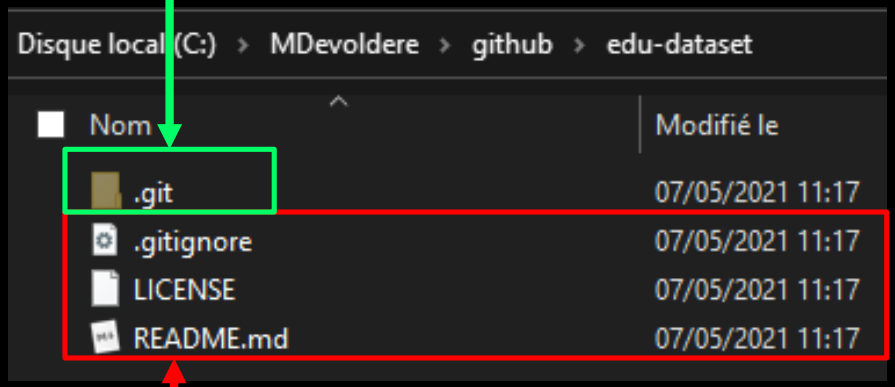




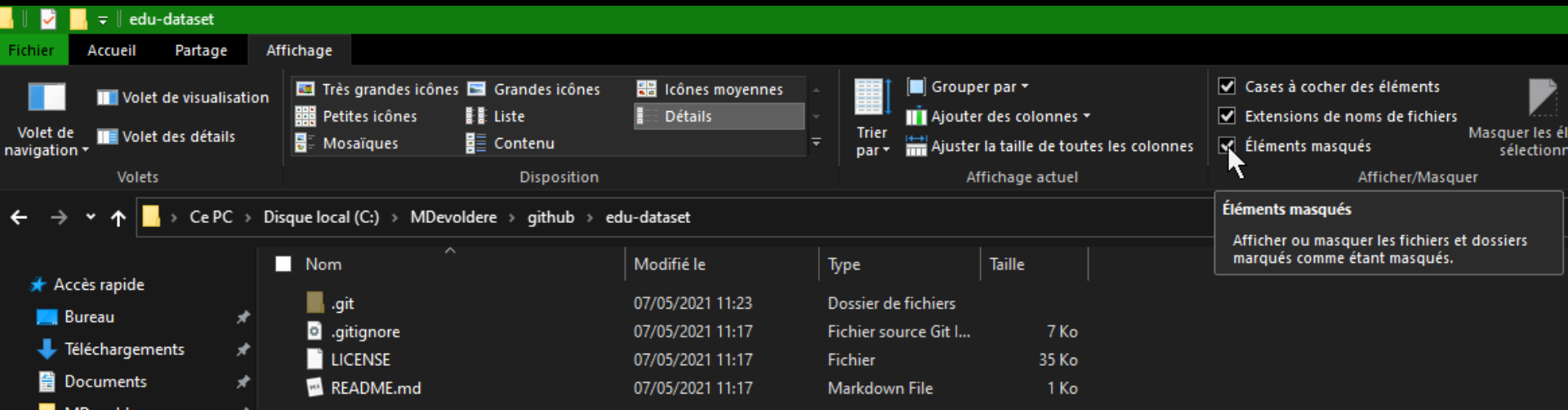

Cloning edu-dataset
Receiving objects: 100% (5/5), 15.13 KiB | 1.38 MiB/s, done.


Disque local (C:) › MDevoldere › github › edu-dataset ›
Nom
Modifié le
Type
Taille
.git
07/05/2021 11:17
Dossier de fichiers
.gitignore
07/05/2021 11:17
Fichier source Git I...
7 Ko
LICENSE
07/05/2021 11:17
Fichier
35 Ko
README.md
07/05/2021 11:17
Markdown File
1 Ko

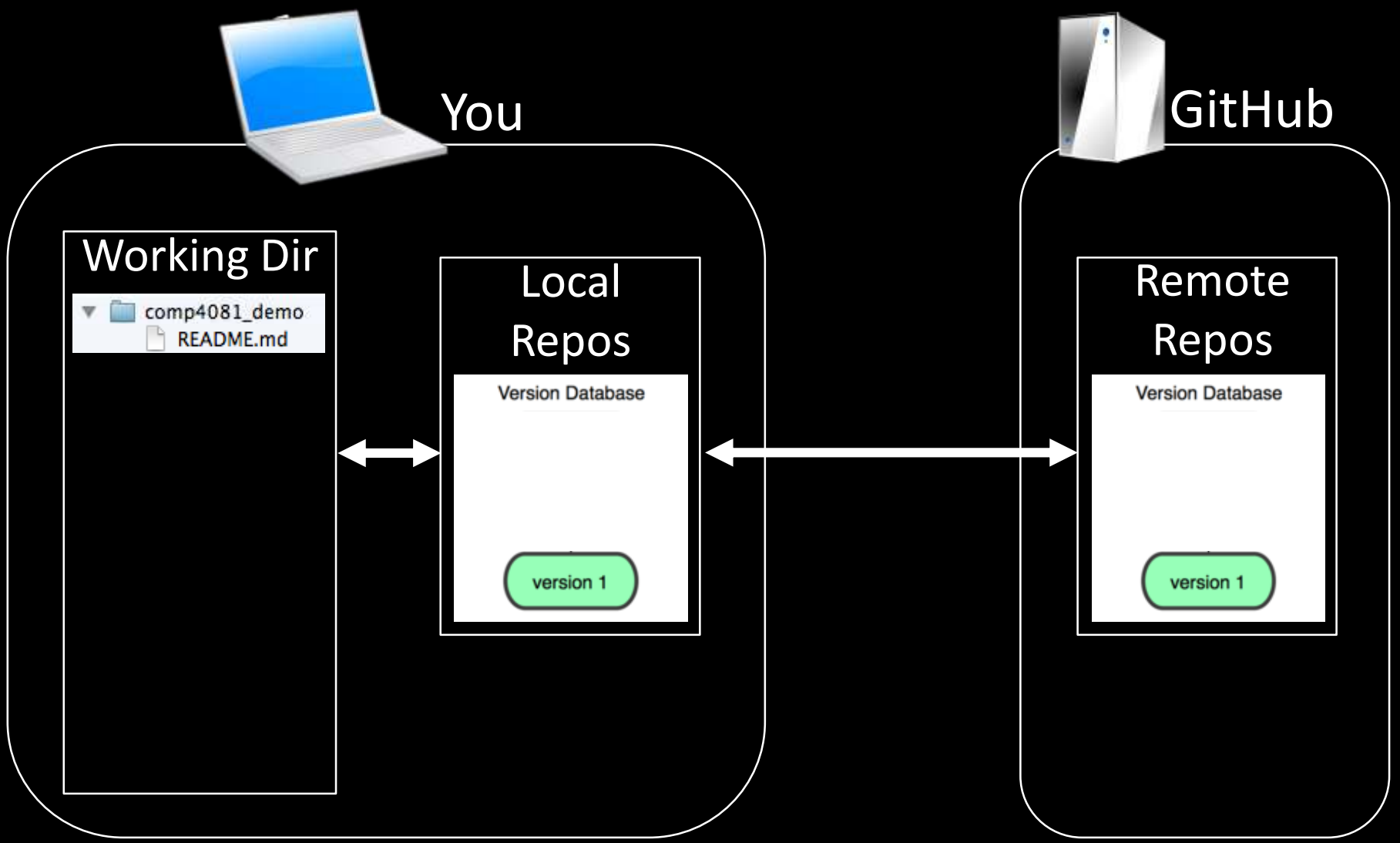
Local Repository



Working Directory (the files you are working on)



```
$ git clone https://github.com/arfp/comp4081_demo.git
```

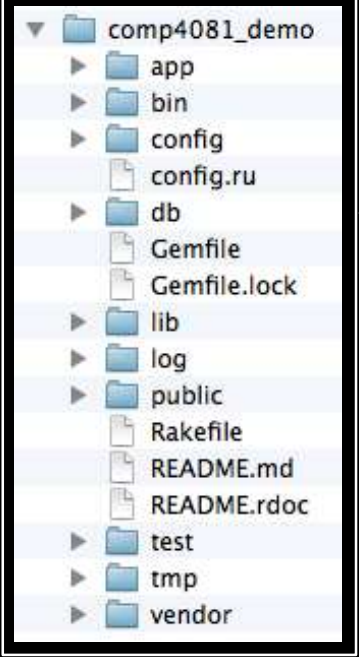


```
$ cd comp4081_demo
// Add/edit files
$ git add -A
$ git commit -m "Created project skeleton"
```

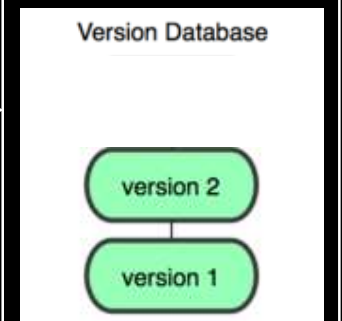
You

GitHub

Working Dir

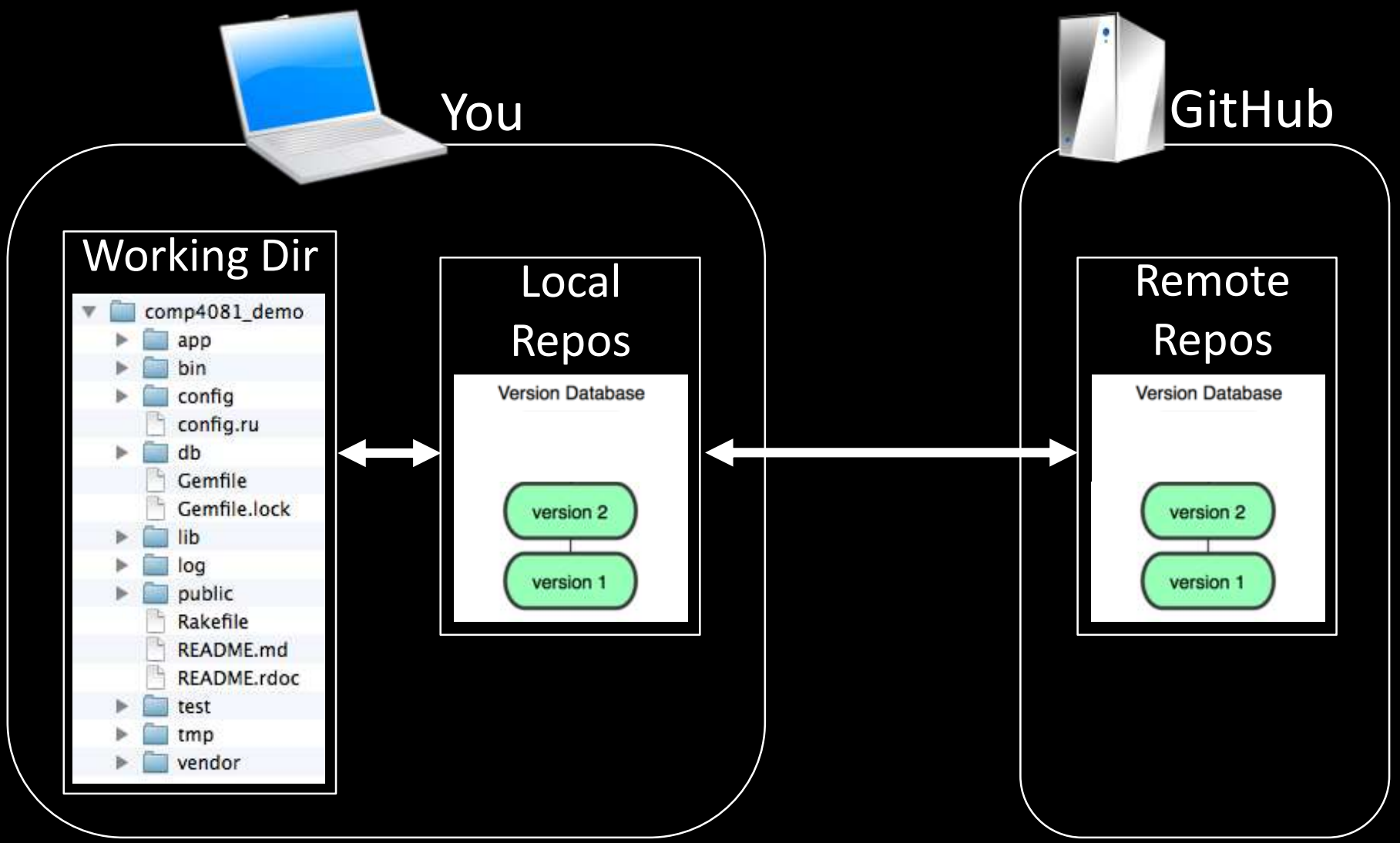


Local Repos

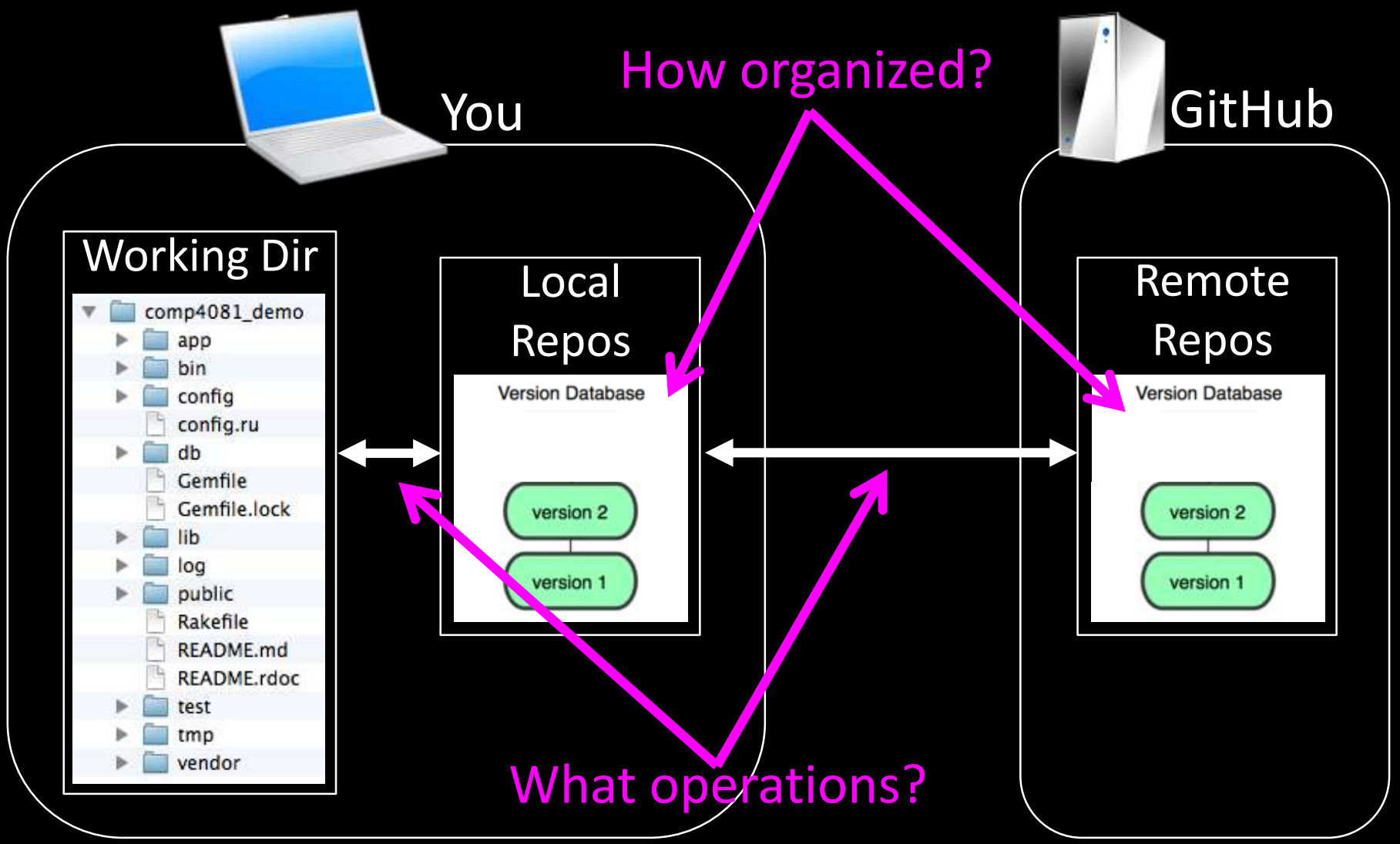


Remote Repos

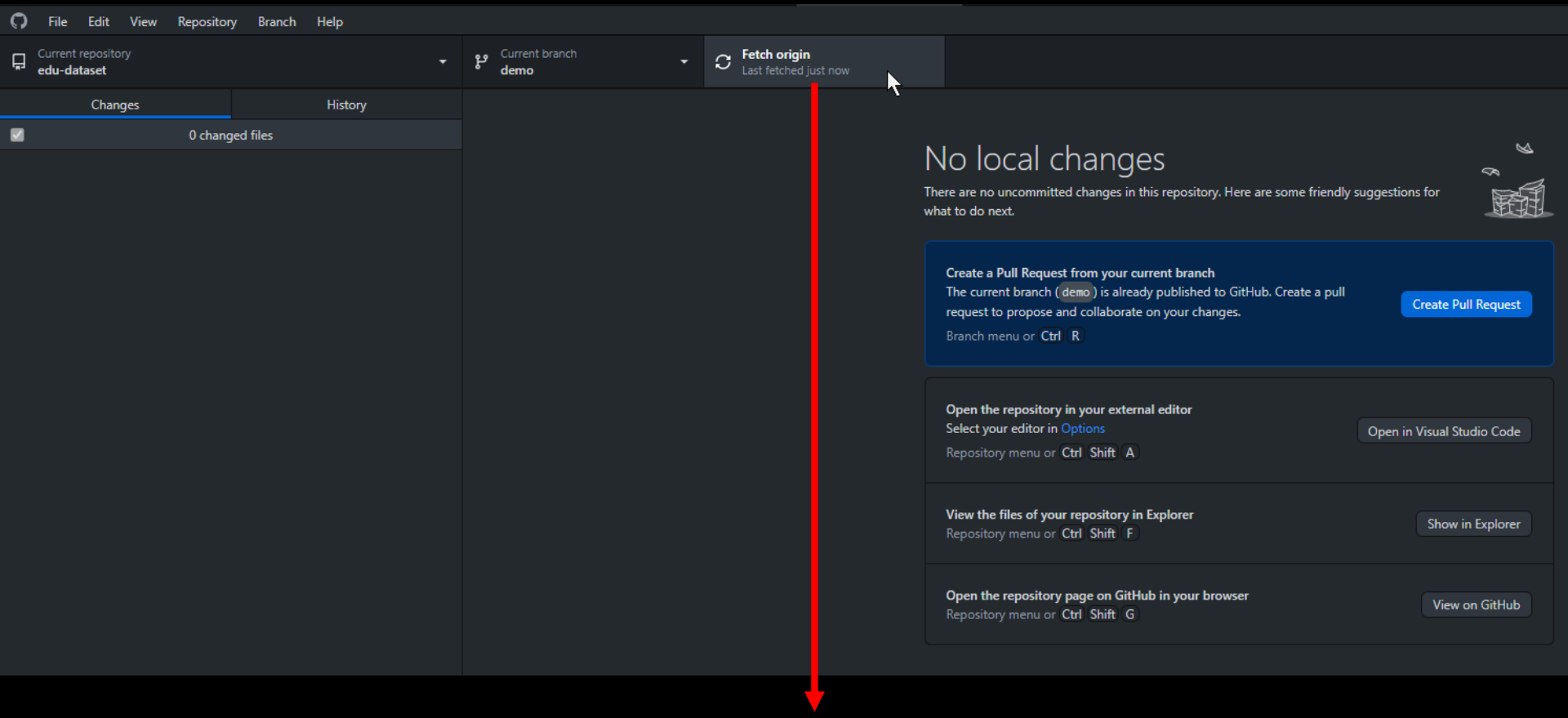
```
$ git push
```


You
GitHub
Working Dir
comp4081_demo
app
bin
config
config.ru
db
Gemfile
Gemfile.lock
lib
log
public
Rakefile
README.md
README.rdoc
test
tmp
vendor
Local Repos
Version Database
version 2
version 1
Remote Repos
Version Database
version 2
version 1

Questions to answer



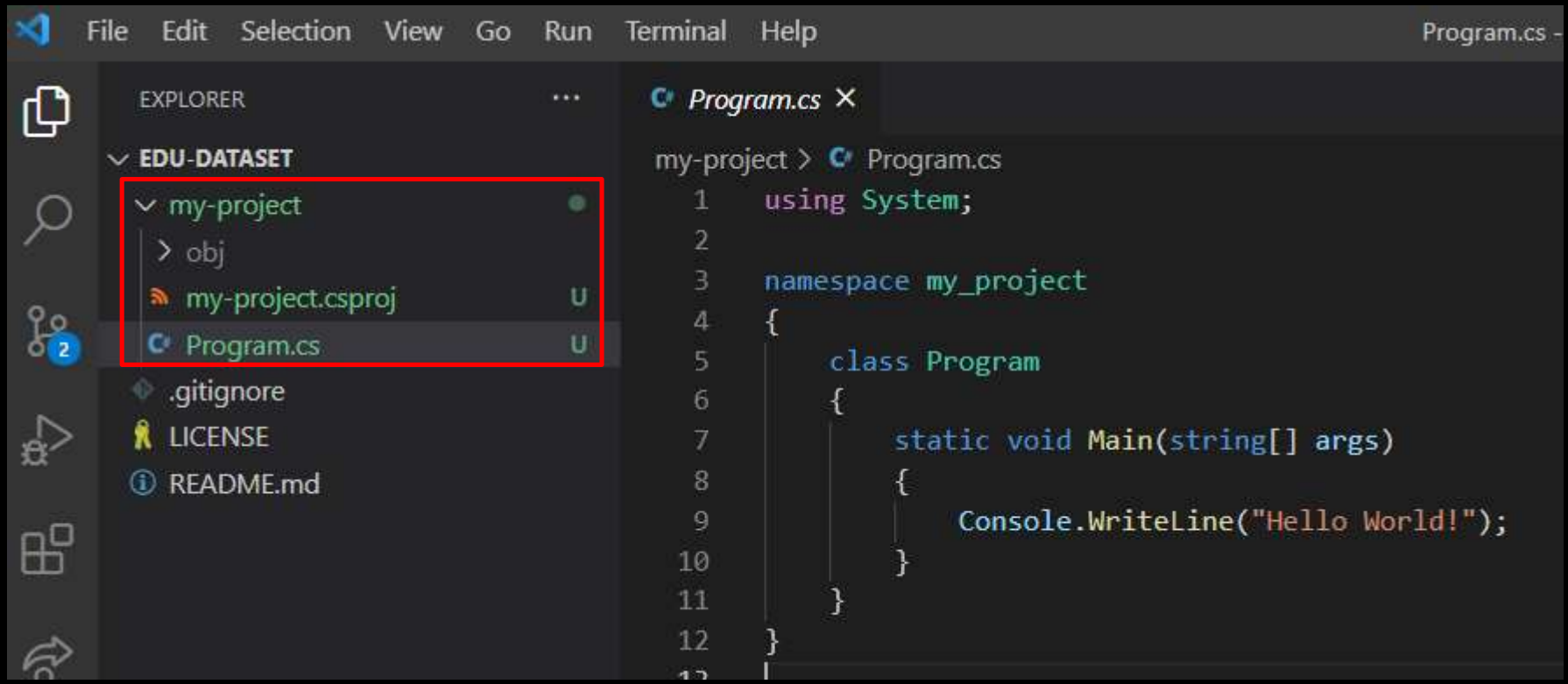
Important: before starting to work

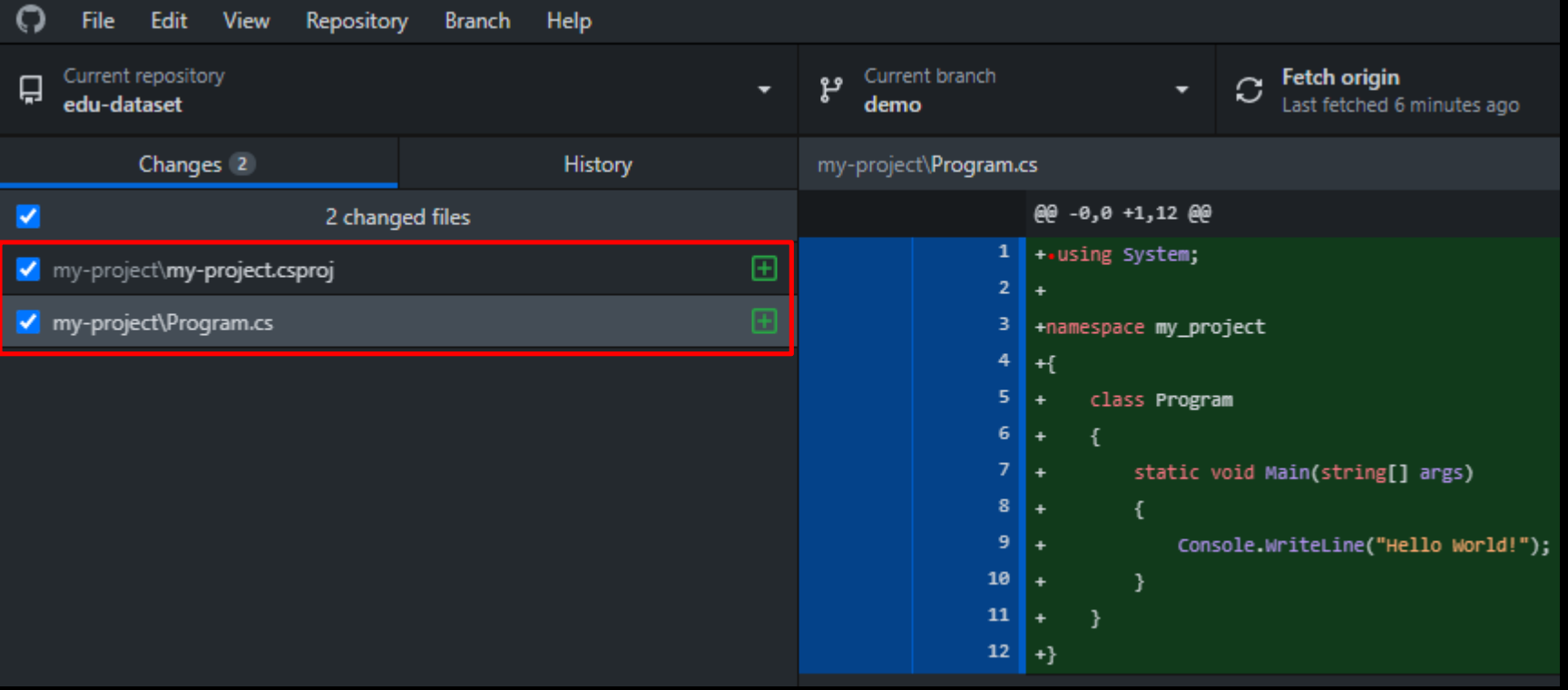


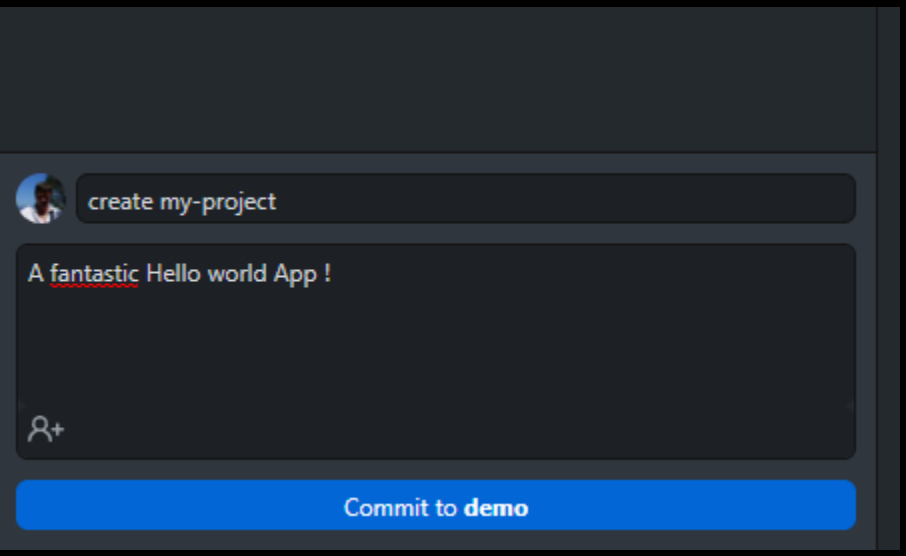
Update your local repository
to make sure you're in sync
with the remote repository

```
git fetch origin
```

Add / Edit files




File Edit View Repository Branch Help
Current repository
edu-dataset
Current branch
demo
Fetch origin
Last fetched 6 minutes ago
Changes 2
History
2 changed files
my-project\my-project.csproj
my-project\Program.cs
my-project\Program.cs
@@ -0,0 +1,12 @@
+using System;
+
+namespace my_project
+{
+ class Program
+ {
+ static void Main(string[] args)
+ {
+ Console.WriteLine("Hello World!");
+ }
+ }
+}

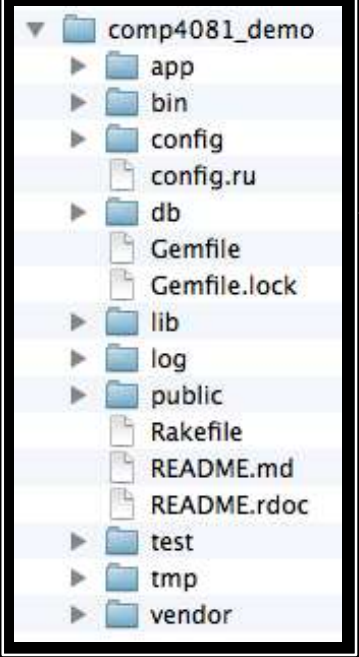

create my-project
A fantastic Hello world App !
Commit to demo

```
$ cd comp4081_demo
// Add/edit files
$ git add -A
$ git commit -m "Created project skeleton"
```

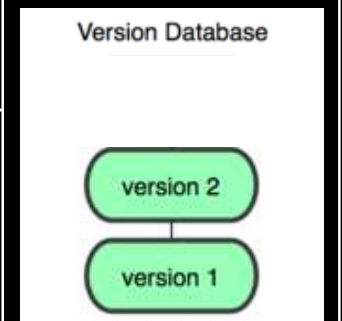
You

GitHub

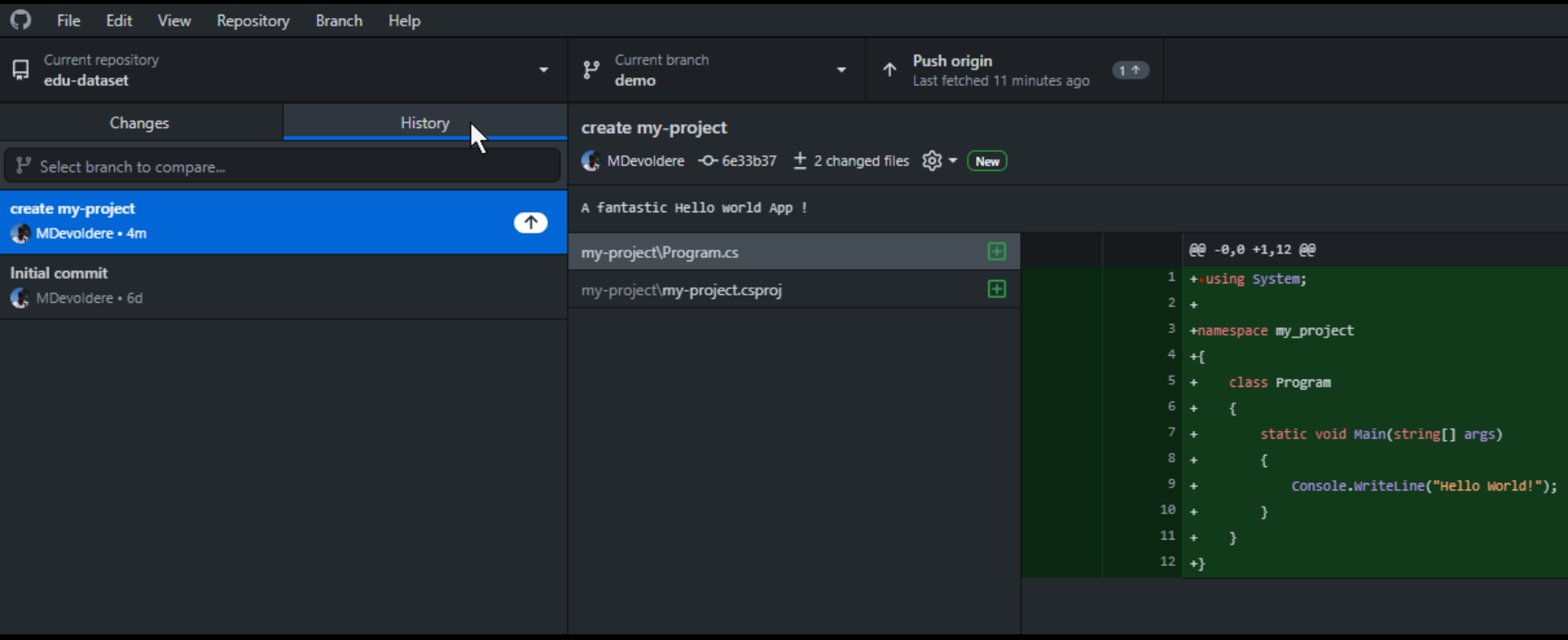
Working Dir

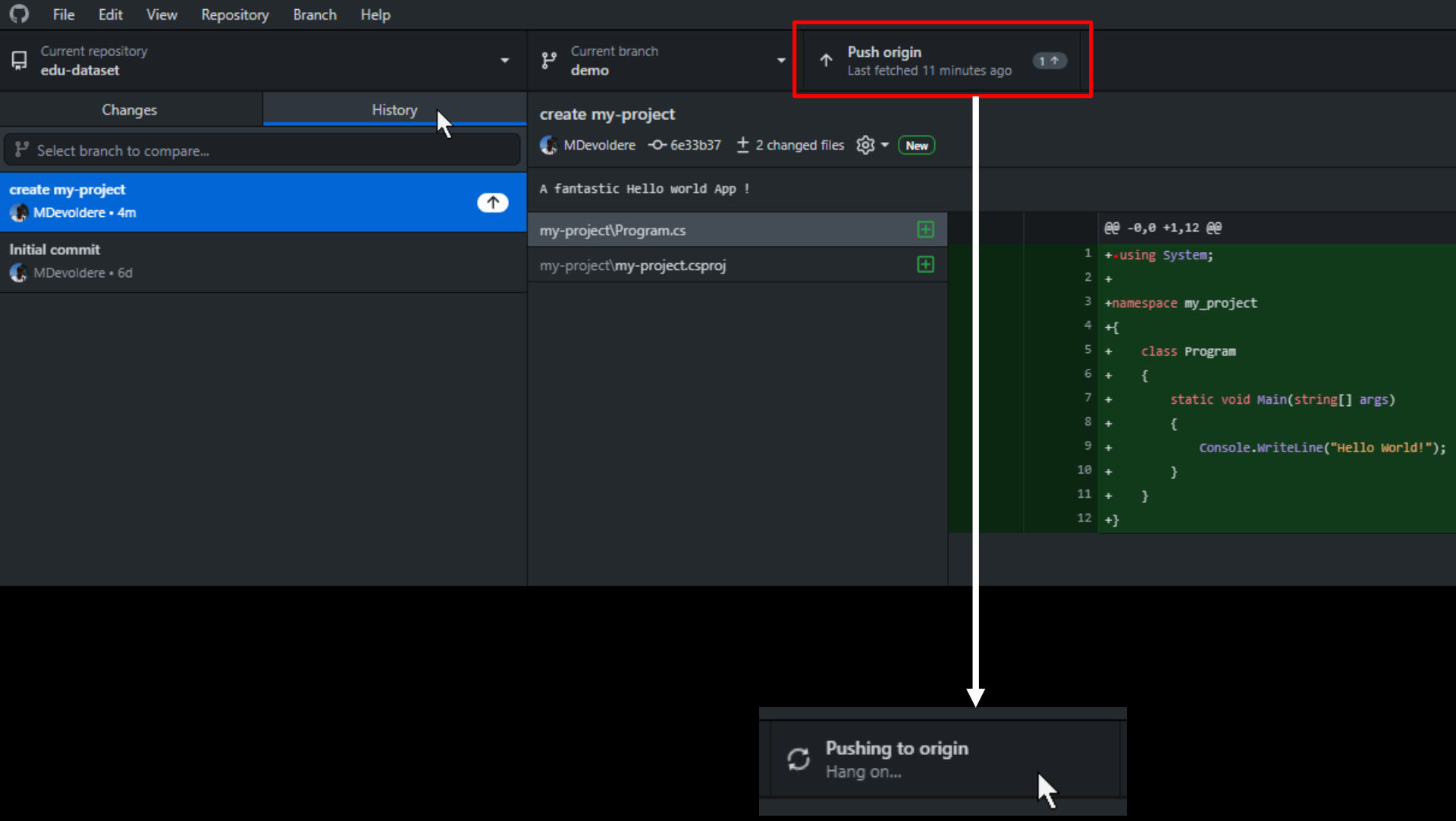


Local Repos



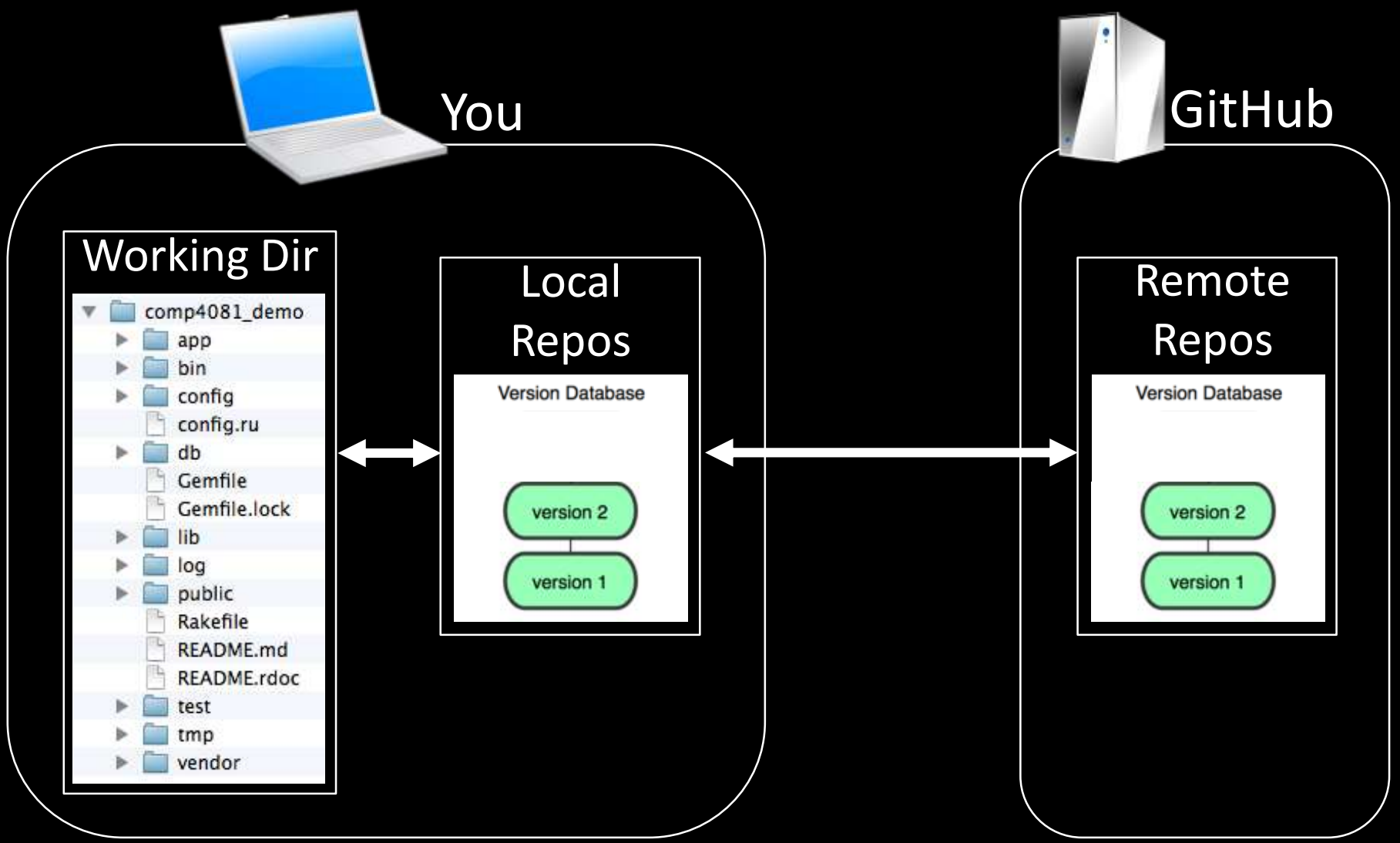
Remote Repos

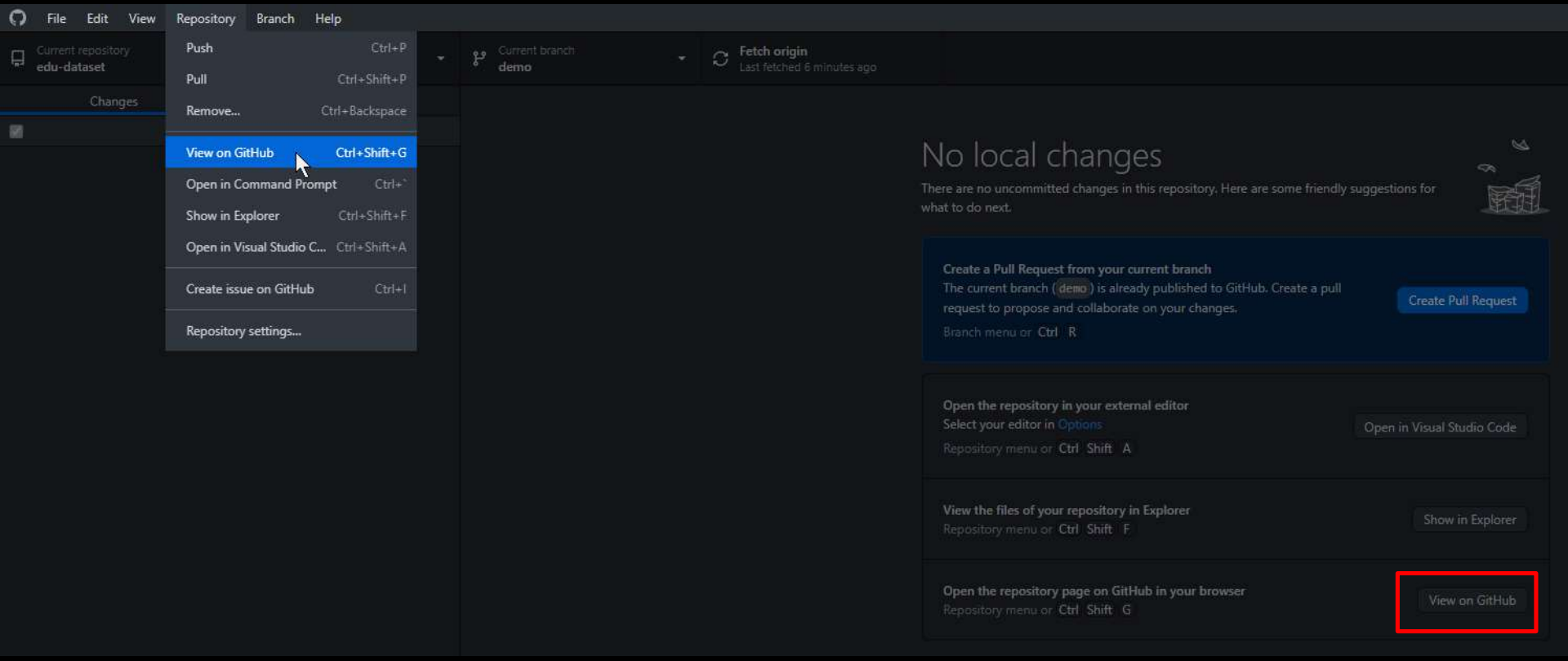

File
Edit
View
Repository
Branch
Help
Current repository
edu-dataset
Current branch
demo
Push origin
Last fetched 11 minutes ago
1
Changes
History
Select branch to compare...
create my-project
MDevoldere • 4m
Initial commit
MDevoldere • 6d
create my-project
MDevoldere
6e33b37
2 changed files
New
A fantastic Hello world App !
my-project\Program.cs
my-project\my-project.csproj
@@ -0,0 +1,12 @@
+using System;
+
+namespace my_project
+{
+ class Program
+ {
+ static void Main(string[] args)
+ {
+ Console.WriteLine("Hello World!");
+ }
+ }
+}


File
Edit
View
Repository
Branch
Help
Current repository
edu-dataset
Current branch
demo
Push origin
Last fetched 11 minutes ago
1
Changes
History
Select branch to compare...
create my-project
MDevoldere • 4m
Initial commit
MDevoldere • 6d
create my-project
MDevoldere
6e33b37
2 changed files
New
A fantastic Hello world App !
my-project\Program.cs
my-project\my-project.csproj
@@ -0,0 +1,12 @@
+using System;
+
+namespace my_project
+{
+ class Program
+ {
+ static void Main(string[] args)
+ {
+ Console.WriteLine("Hello World!");
+ }
+ }
+}

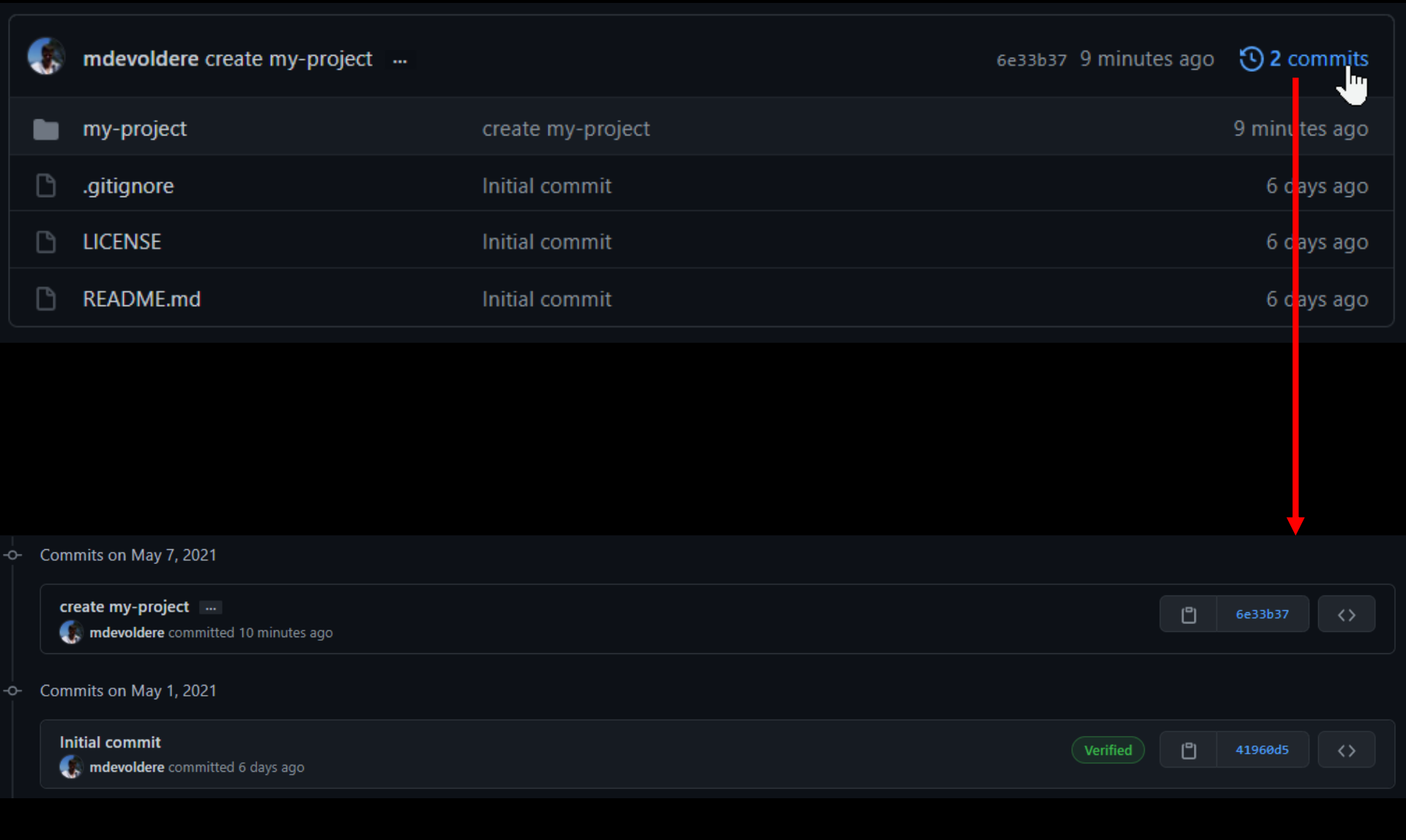
Pushing to origin
Hang on...

```
$ git push
```

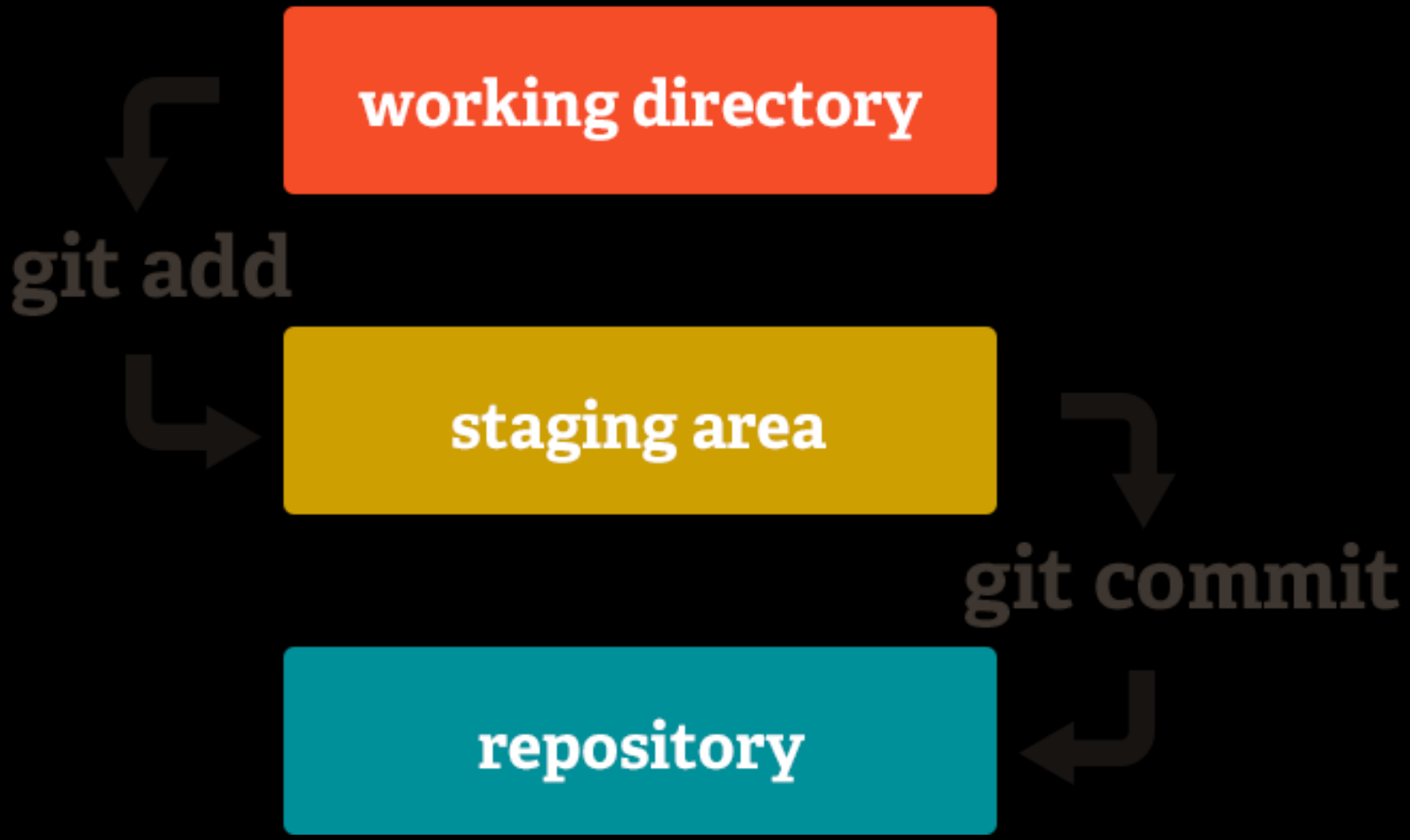

You
GitHub
Working Dir
comp4081_demo
app
bin
config
config.ru
db
Gemfile
Gemfile.lock
lib
log
public
Rakefile
README.md
README.rdoc
test
tmp
vendor
Local Repos
Version Database
version 2
version 1
Remote Repos
Version Database
version 2
version 1

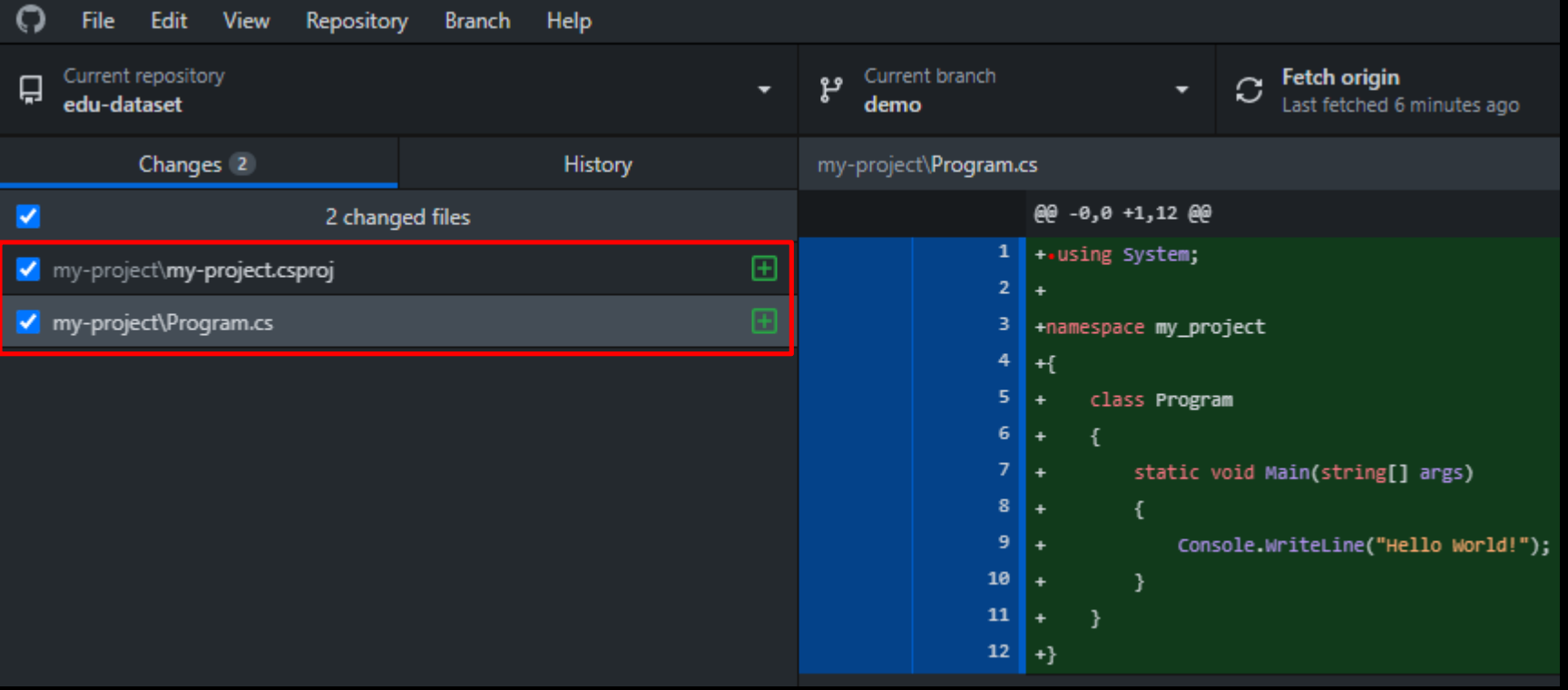

File
Edit
View
Repository
Branch
Help
Push Ctrl+P
Pull Ctrl+Shift+P
Remove... Ctrl+Backspace
View on GitHub Ctrl+Shift+G
Open in Command Prompt Ctrl+`
Show in Explorer Ctrl+Shift+F
Open in Visual Studio C... Ctrl+Shift+A
Create issue on GitHub Ctrl+I
Repository settings...
Current repository
edu-dataset
Changes
Current branch
demo
Fetch origin
Last fetched 6 minutes ago
No local changes
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.
Create a Pull Request from your current branch
The current branch (demo) is already published to GitHub. Create a pull request to propose and collaborate on your changes.
Branch menu or Ctrl R
Create Pull Request
Open the repository in your external editor
Select your editor in Options
Repository menu or Ctrl Shift A
Open in Visual Studio Code
View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer
Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub

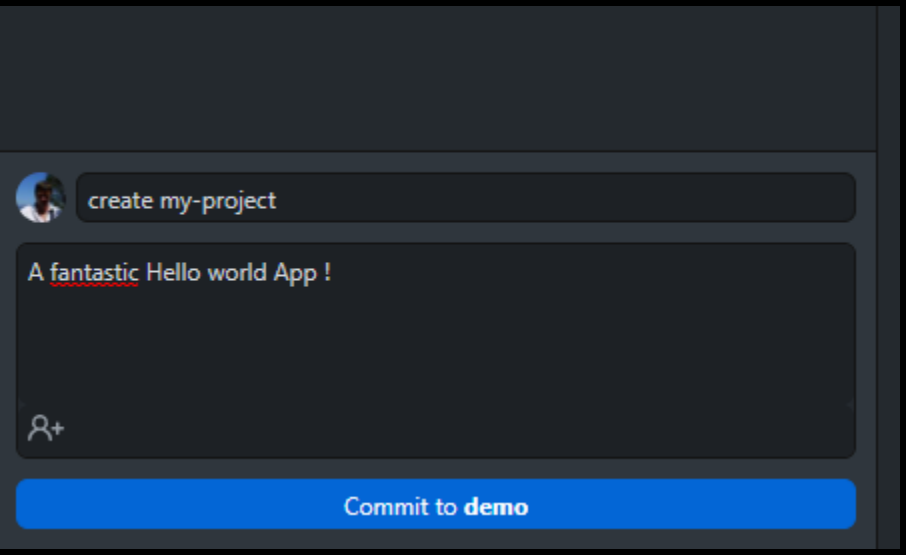
https://github.com/mdevoldere/edu-dataset

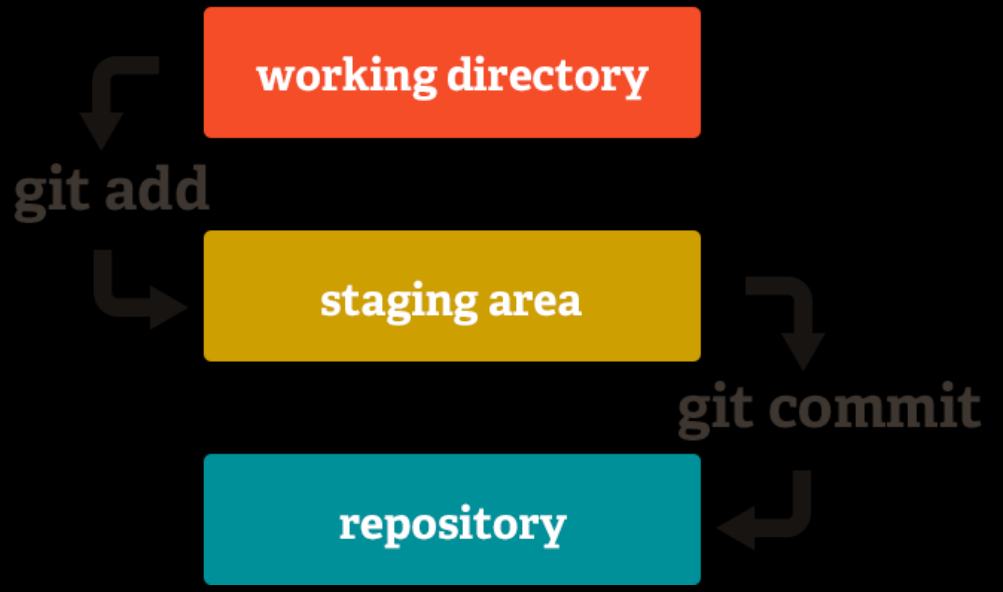


How the repos is organized

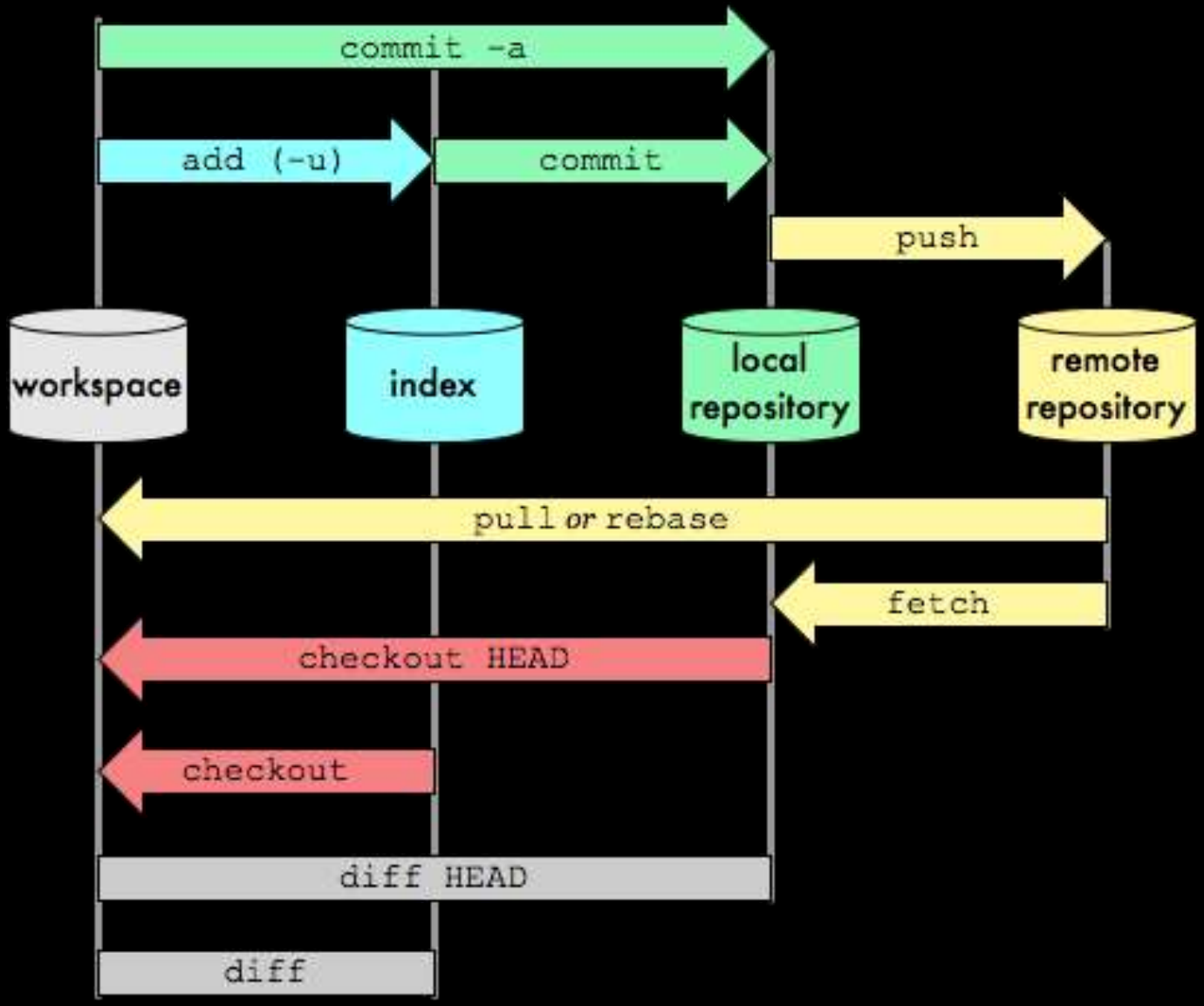



File Edit View Repository Branch Help
Current repository
edu-dataset
Current branch
demo
Fetch origin
Last fetched 6 minutes ago
Changes 2
History
my-project\Program.cs
2 changed files
my-project\my-project.csproj
my-project\Program.cs
@@ -0,0 +1,12 @@
+using System;
+
+namespace my_project
+{
+ class Program
+ {
+ static void Main(string[] args)
+ {
+ Console.WriteLine("Hello World!");
+ }
+ }
+}

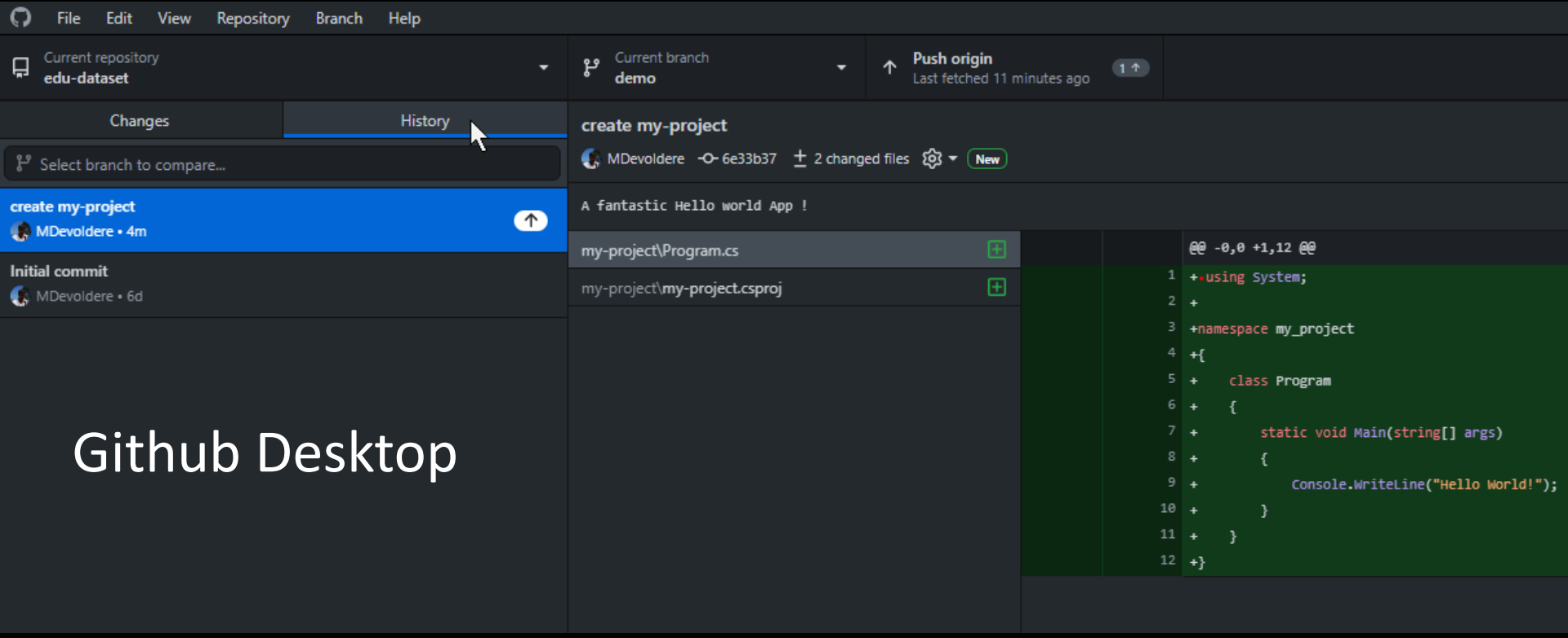

create my-project
A fantastic Hello world App !
Commit to demo


working directory
git add
staging area
git commit
repository

How the repos is organized

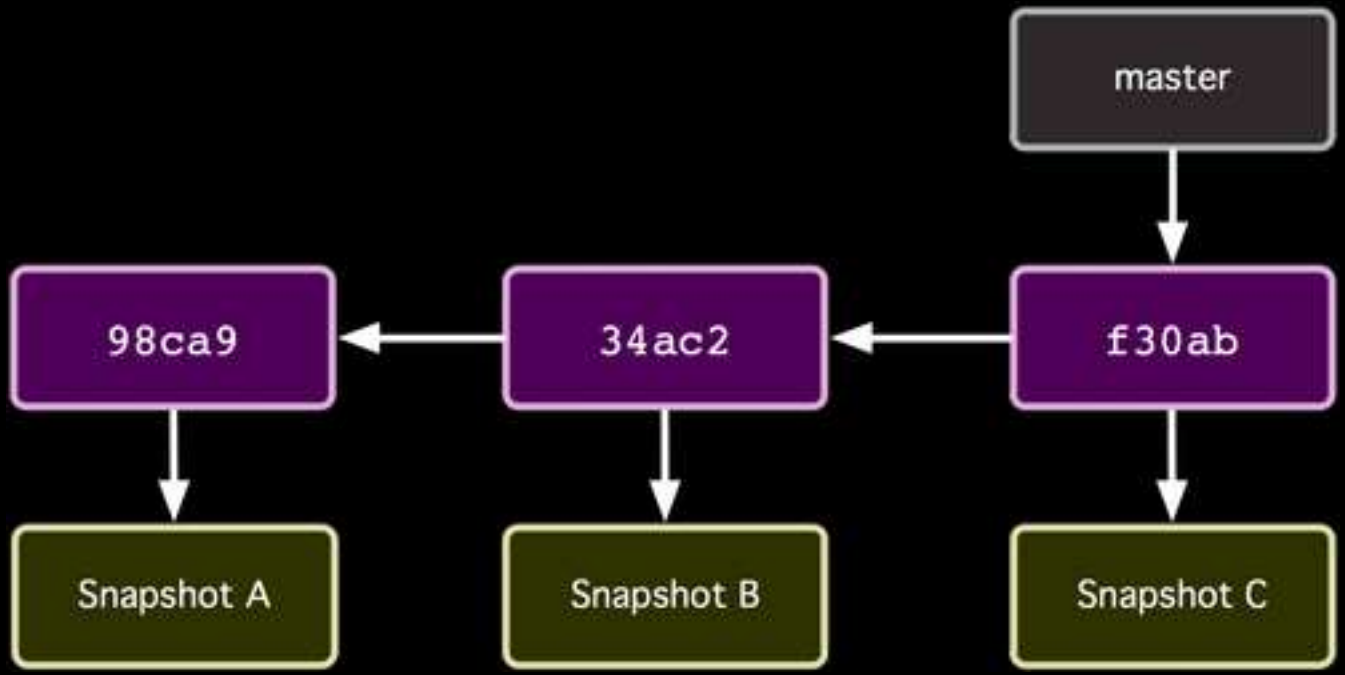


http://git-scm.com/book/



How are commits organized?

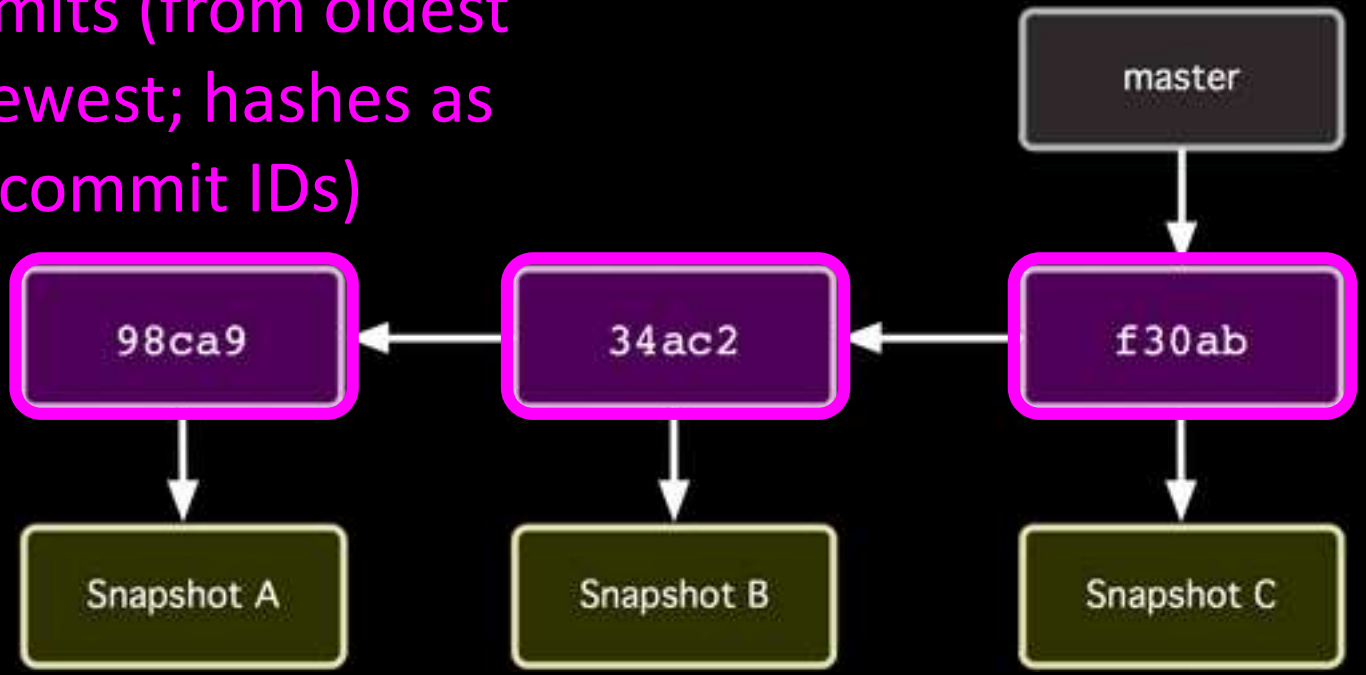
How the repos is organized



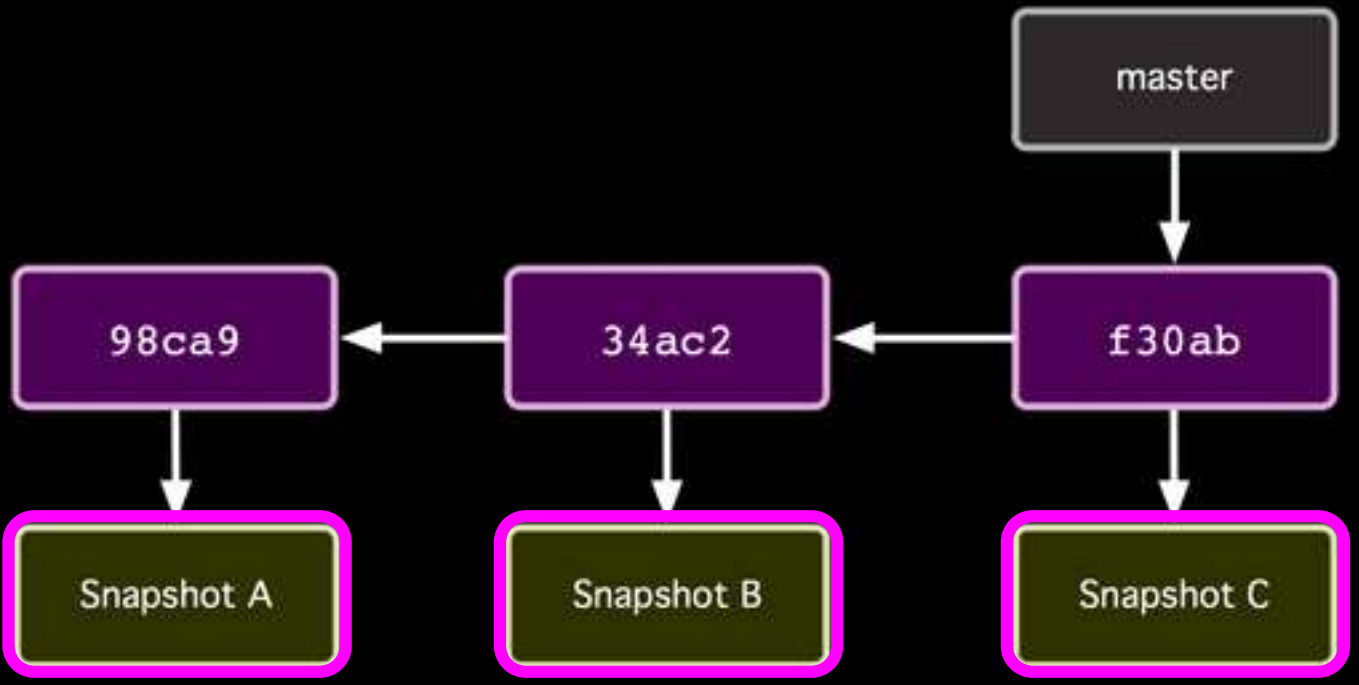
http://git-scm.com/book/

How the repos is organized

Commits (from oldest to newest; hashes as commit IDs)

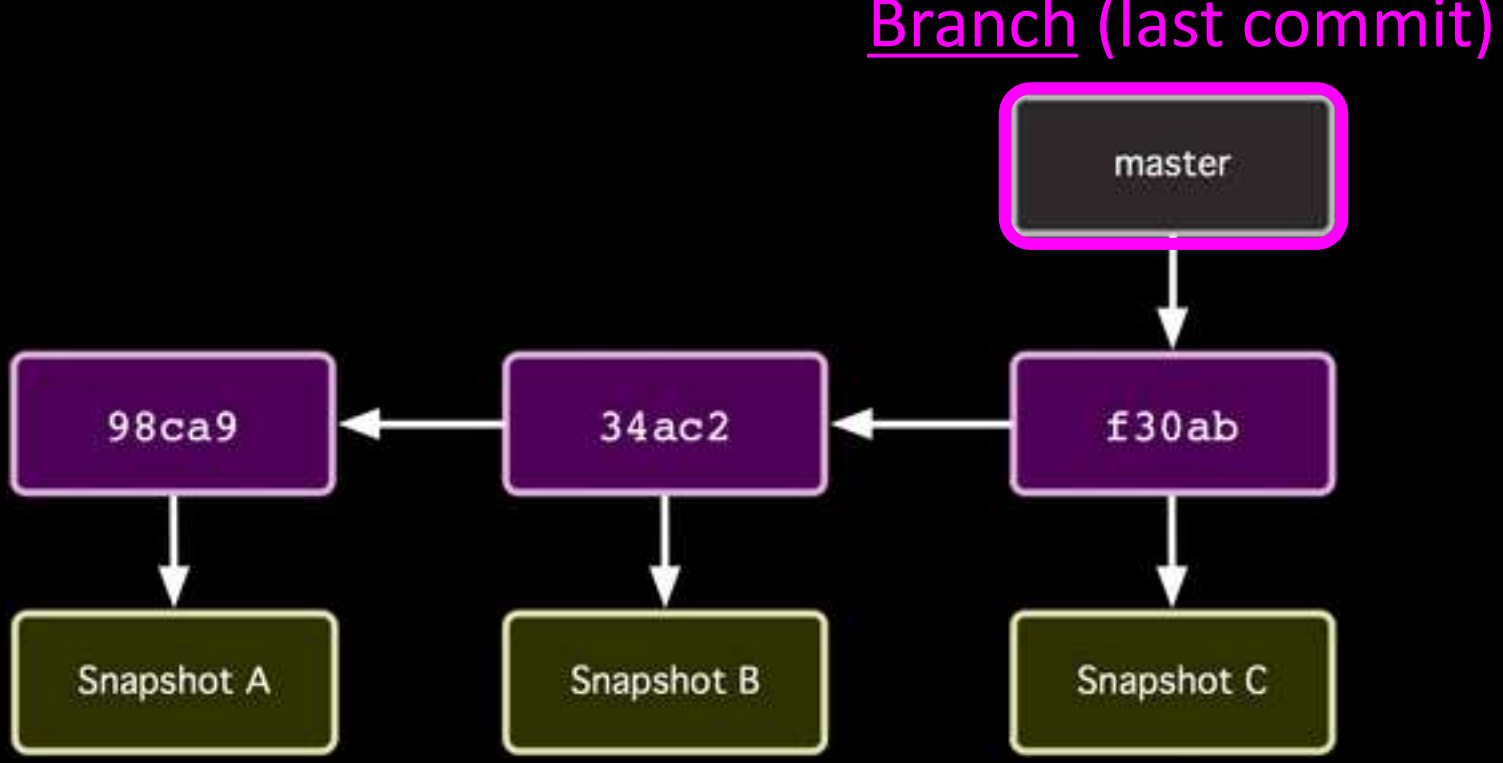


How the repos is organized



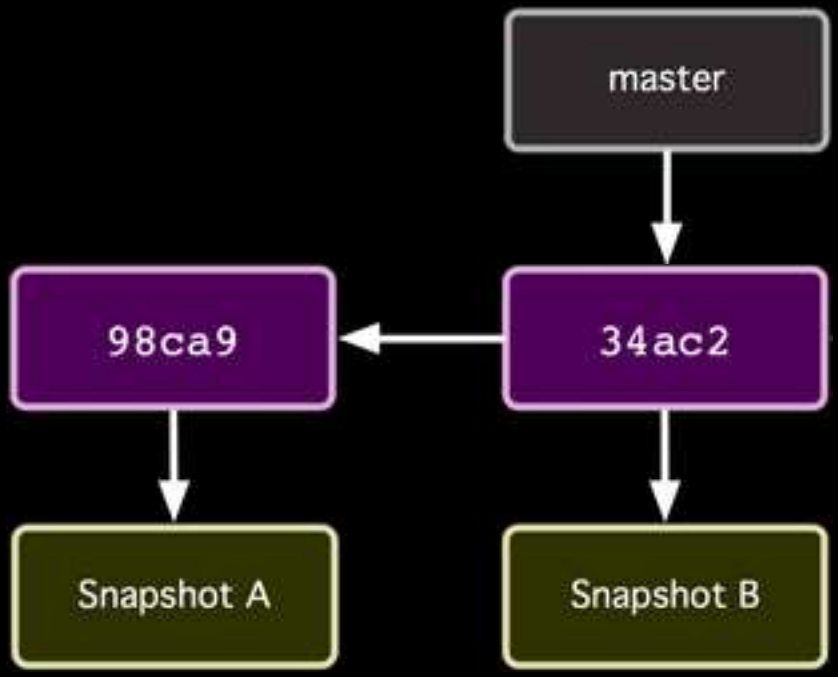
Snapshot of all files
at each commit

How the repos is organized

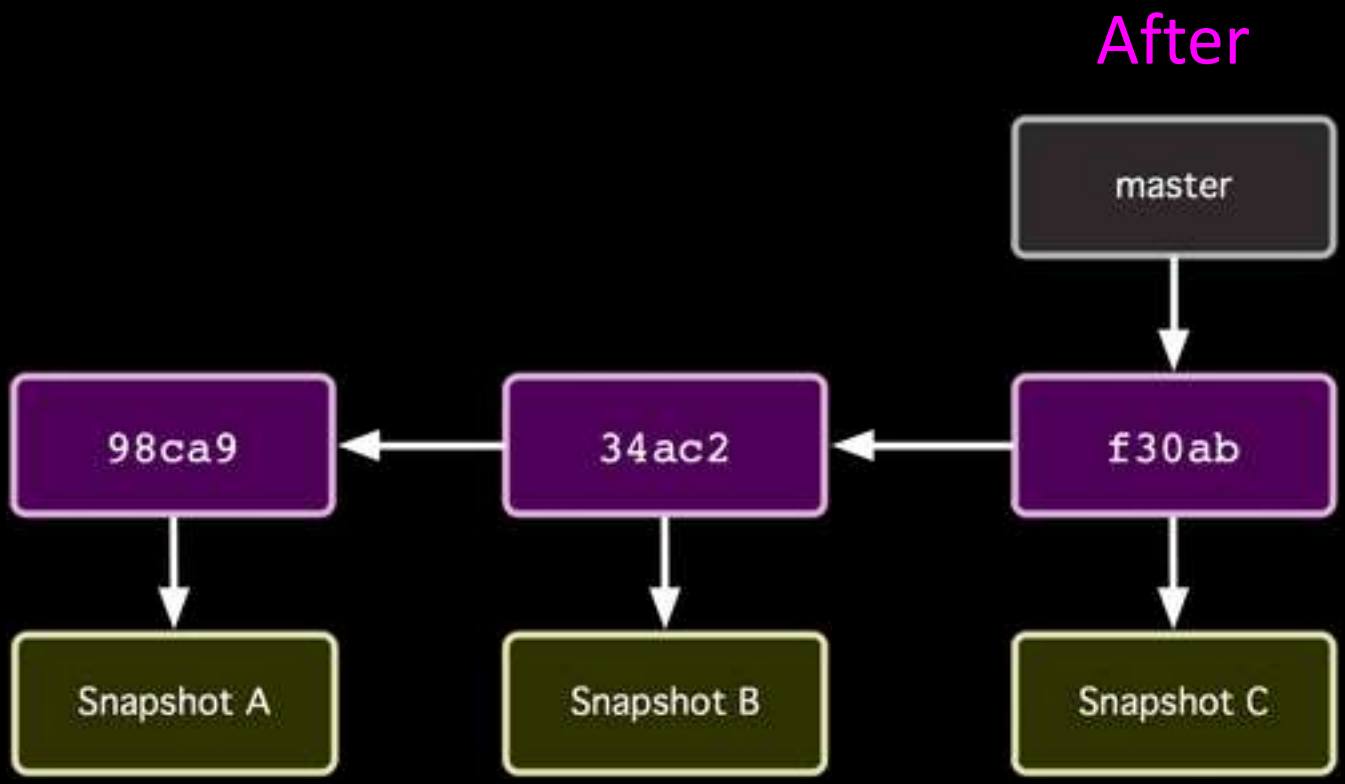


How commit works

Before



How commit works



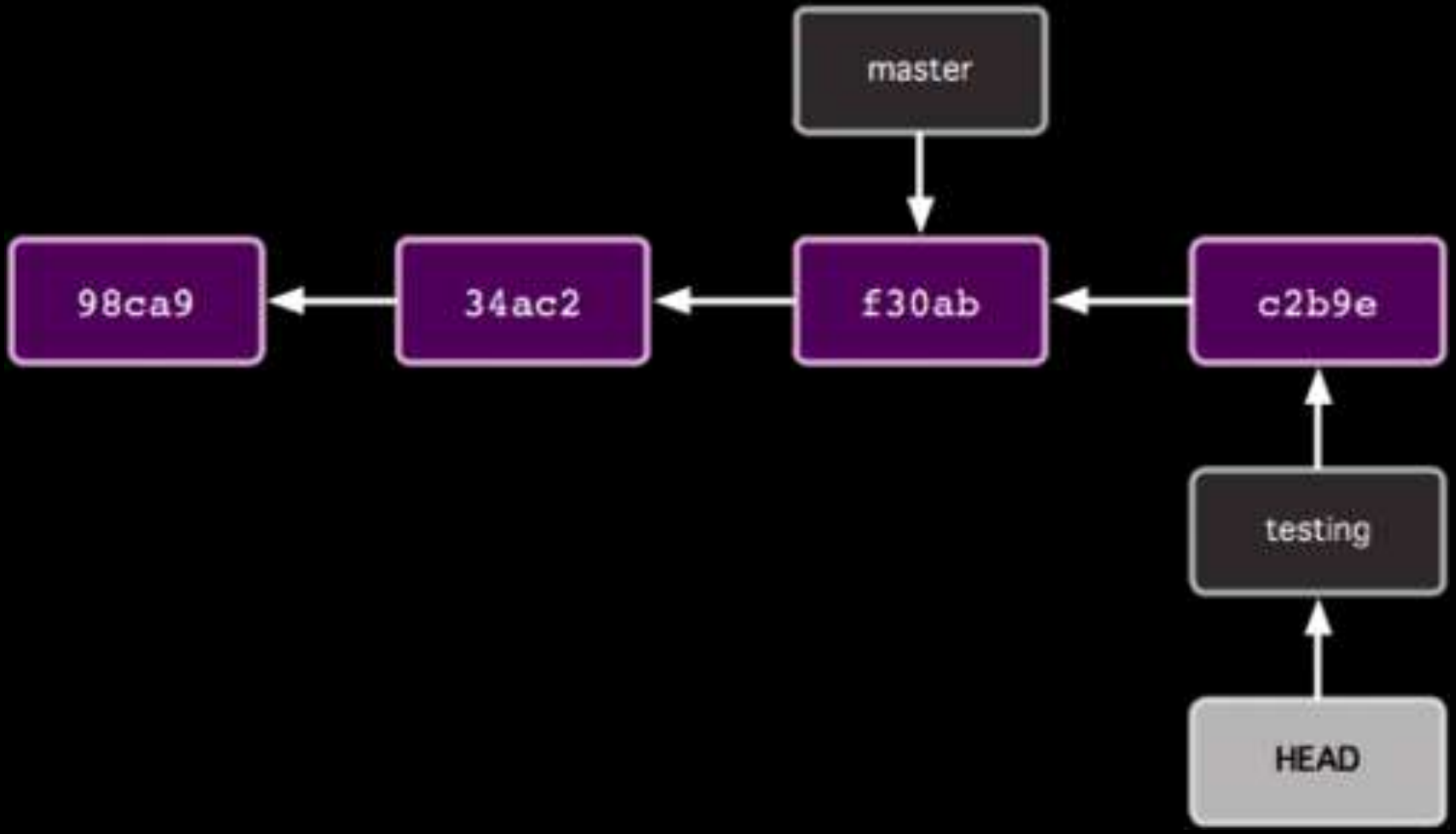
Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

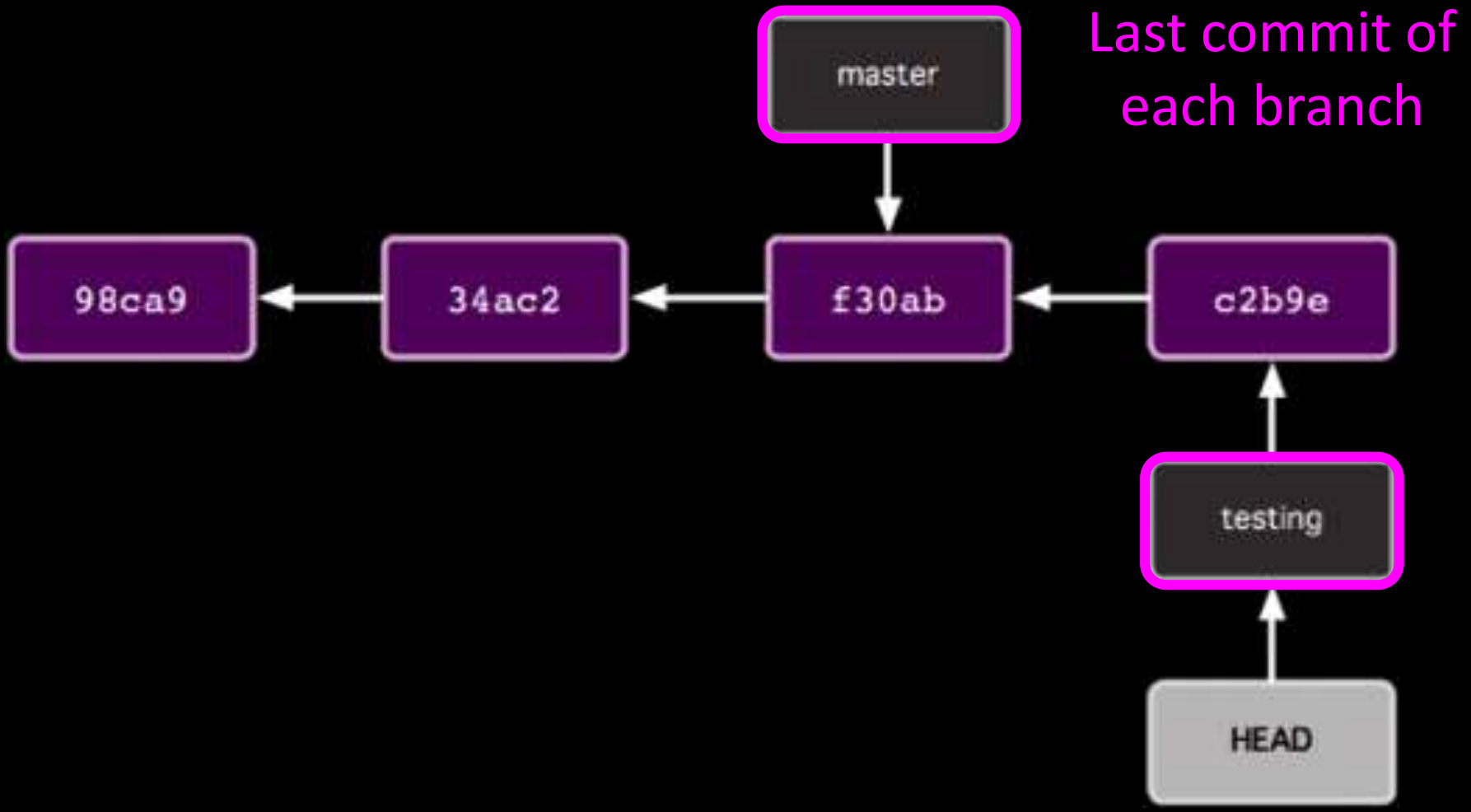
Make changes in local branch

Merge with GitHub repos

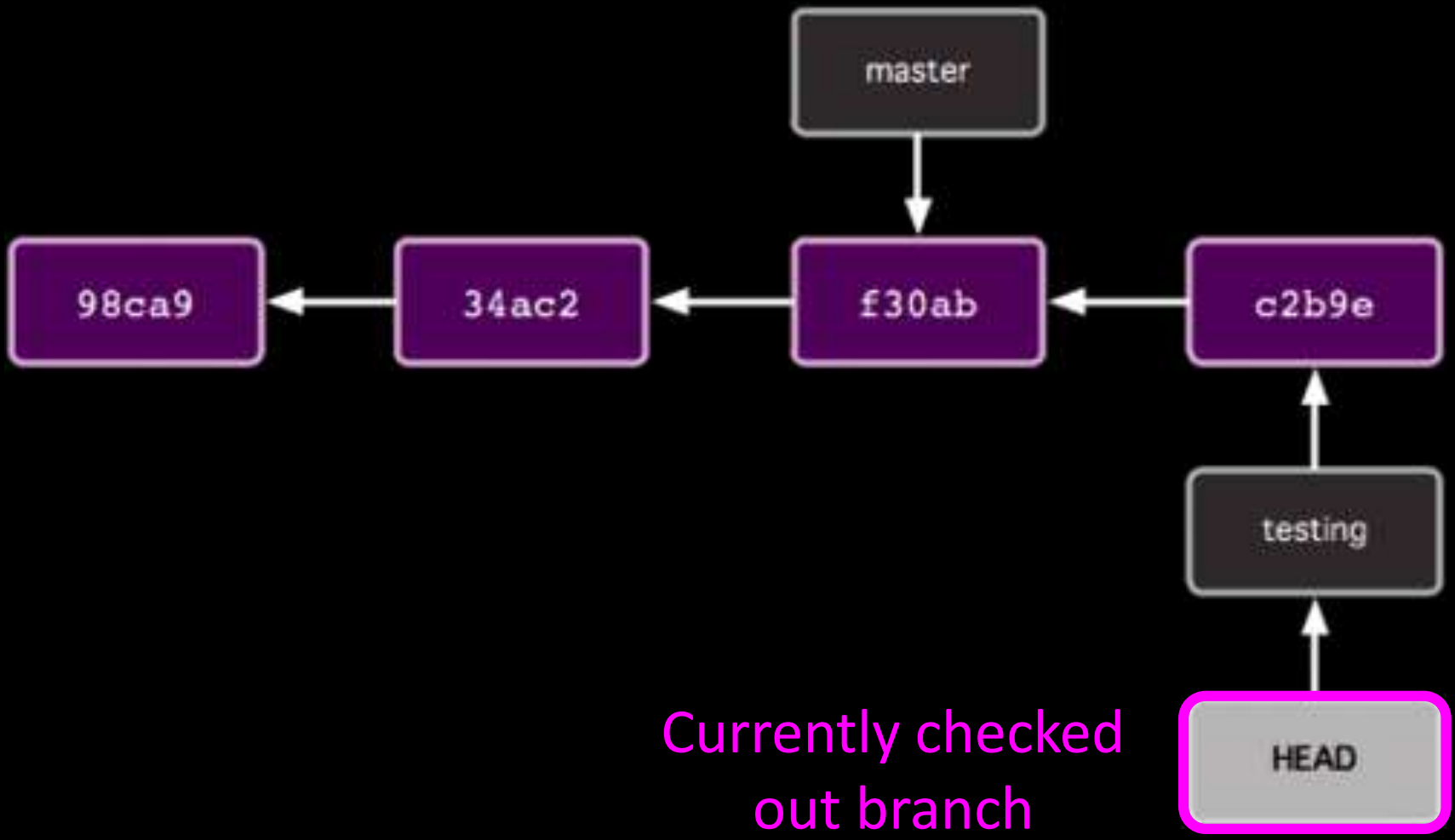
Organization with two branches



Organization with two branches

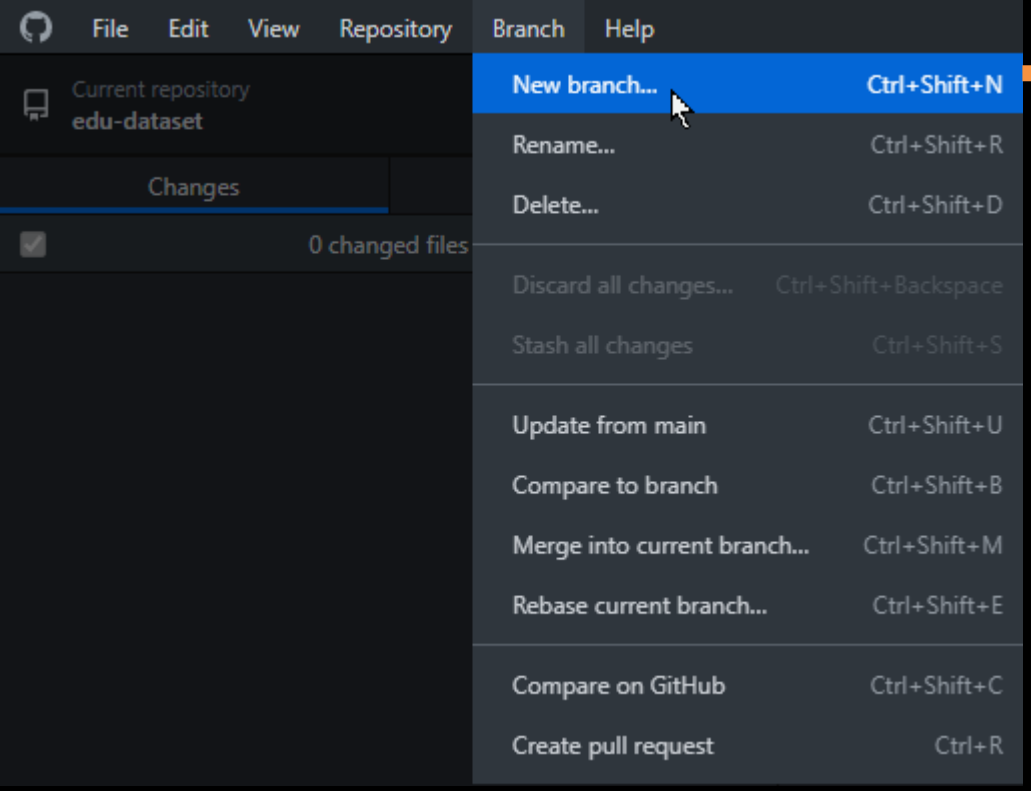


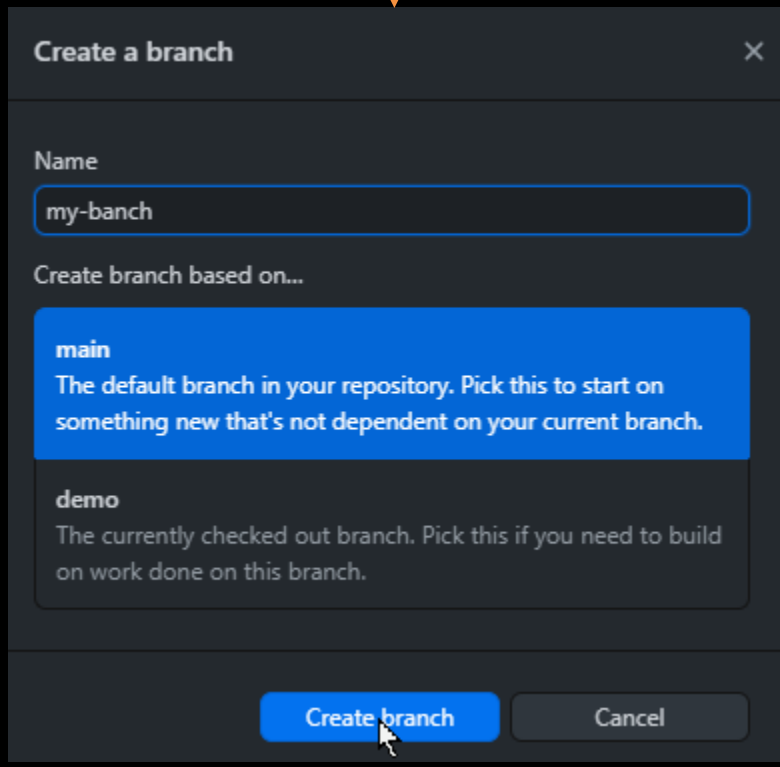
Organization with two branches



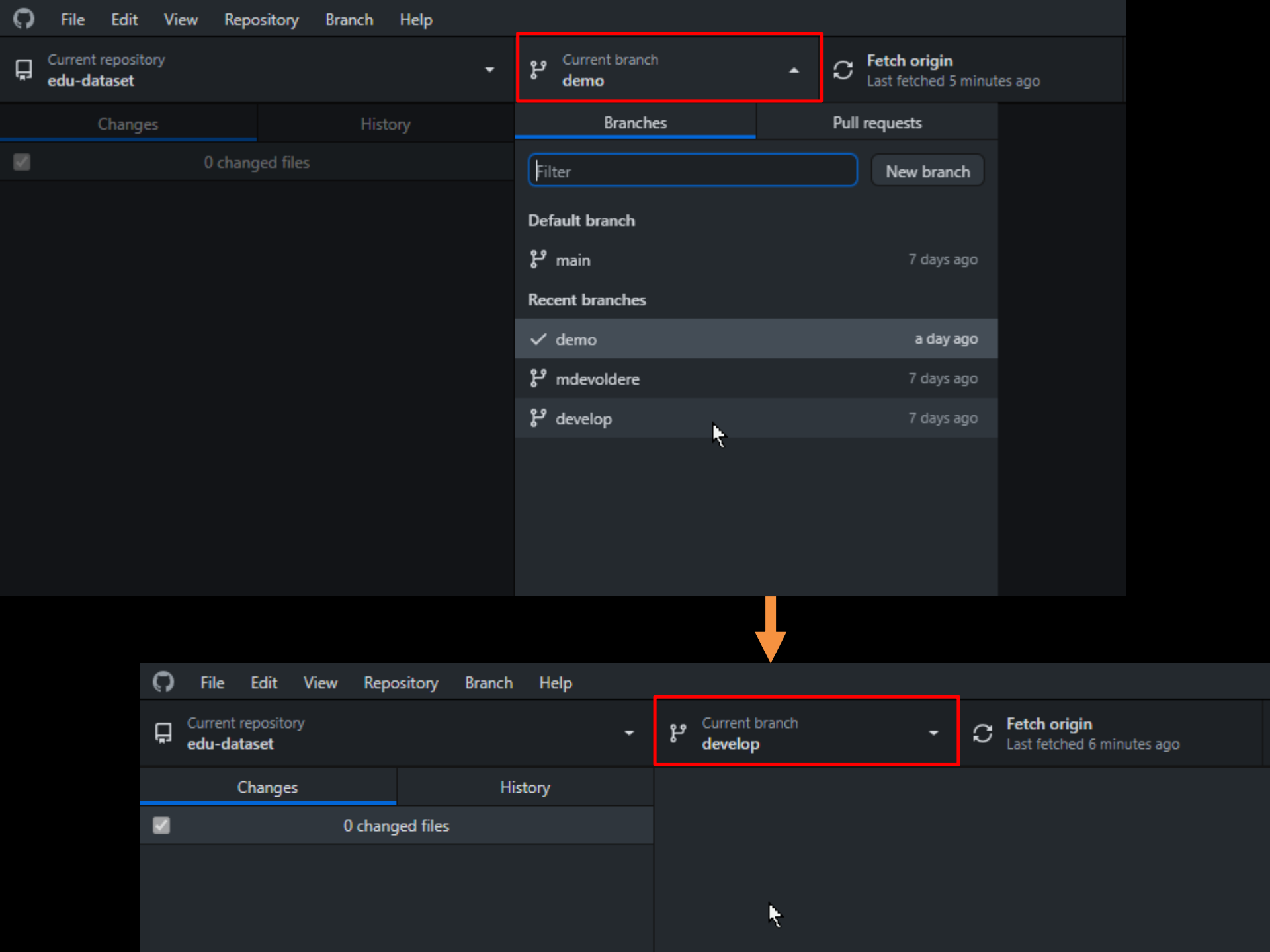
Common Workflow

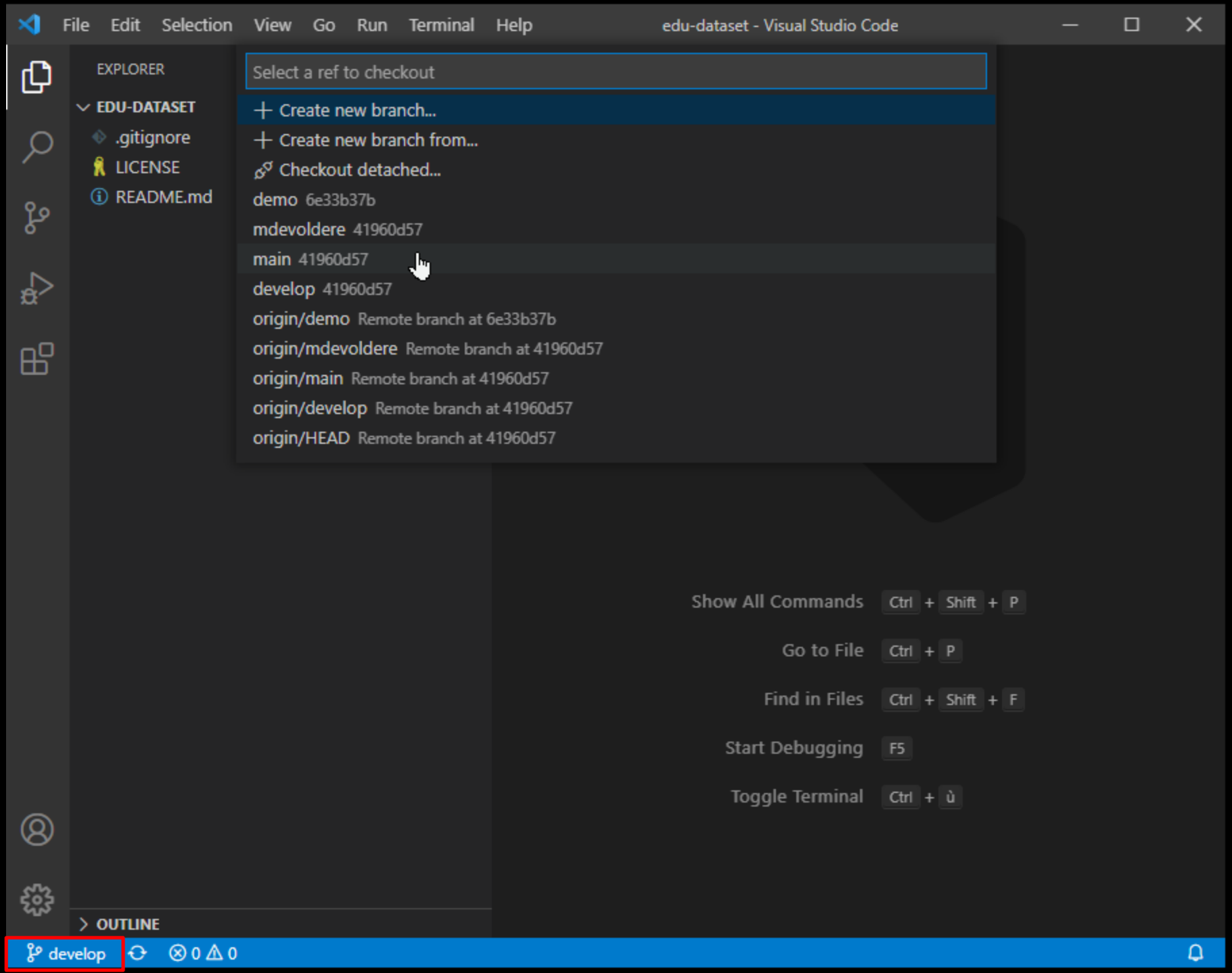
1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos





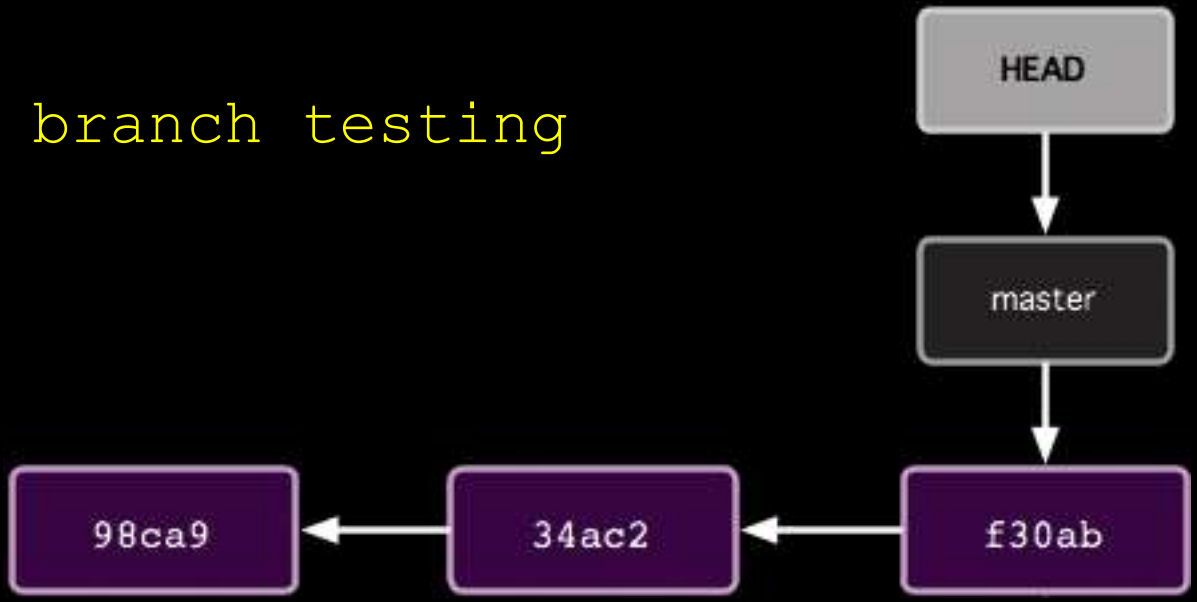

```
$ git branch my-branch
$ git checkout my-branch
ou
$ git checkout –b my-banch
```


File
Edit
View
Repository
Branch
Help
Current repository
edu-dataset
Current branch
demo
Fetch origin
Last fetched 5 minutes ago
Changes
History
Branches
Pull requests
0 changed files
Filter
New branch
Default branch
main
7 days ago
Recent branches
demo
a day ago
mdevoldere
7 days ago
develop
7 days ago
File
Edit
View
Repository
Branch
Help
Current repository
edu-dataset
Current branch
develop
Fetch origin
Last fetched 6 minutes ago
Changes
History
0 changed files


File Edit Selection View Go Run Terminal Help
edu-dataset - Visual Studio Code
EXPLORER
EDU-DATASET
.gitignore
LICENSE
README.md
Select a ref to checkout
Create new branch...
Create new branch from...
Checkout detached...
demo 6e33b37b
mdevoldere 41960d57
main 41960d57
develop 41960d57
origin/demo Remote branch at 6e33b37b
origin/mdevoldere Remote branch at 41960d57
origin/main Remote branch at 41960d57
origin/develop Remote branch at 41960d57
origin/HEAD Remote branch at 41960d57
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + ù
OUTLINE
develop
0 0

How git branch works

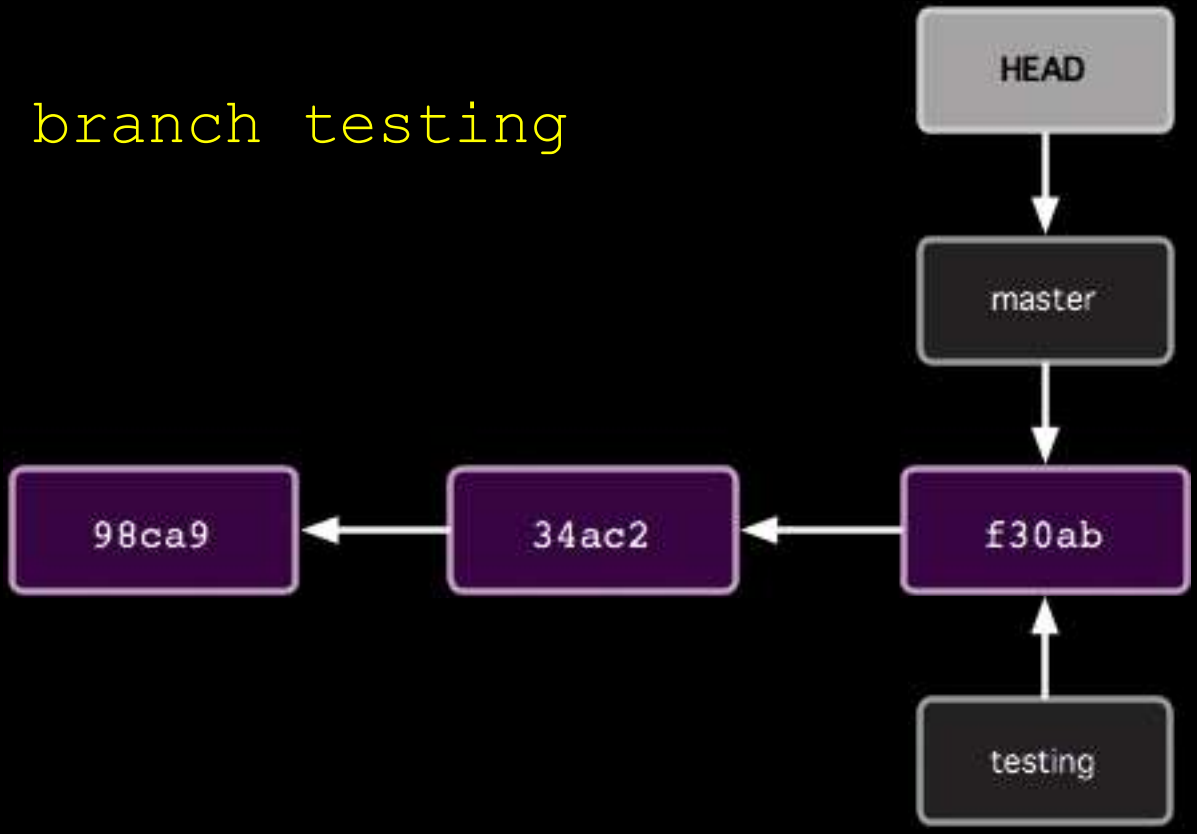
```
$ git branch testing
```



Before

How git branch works

```
$ git branch testing
```



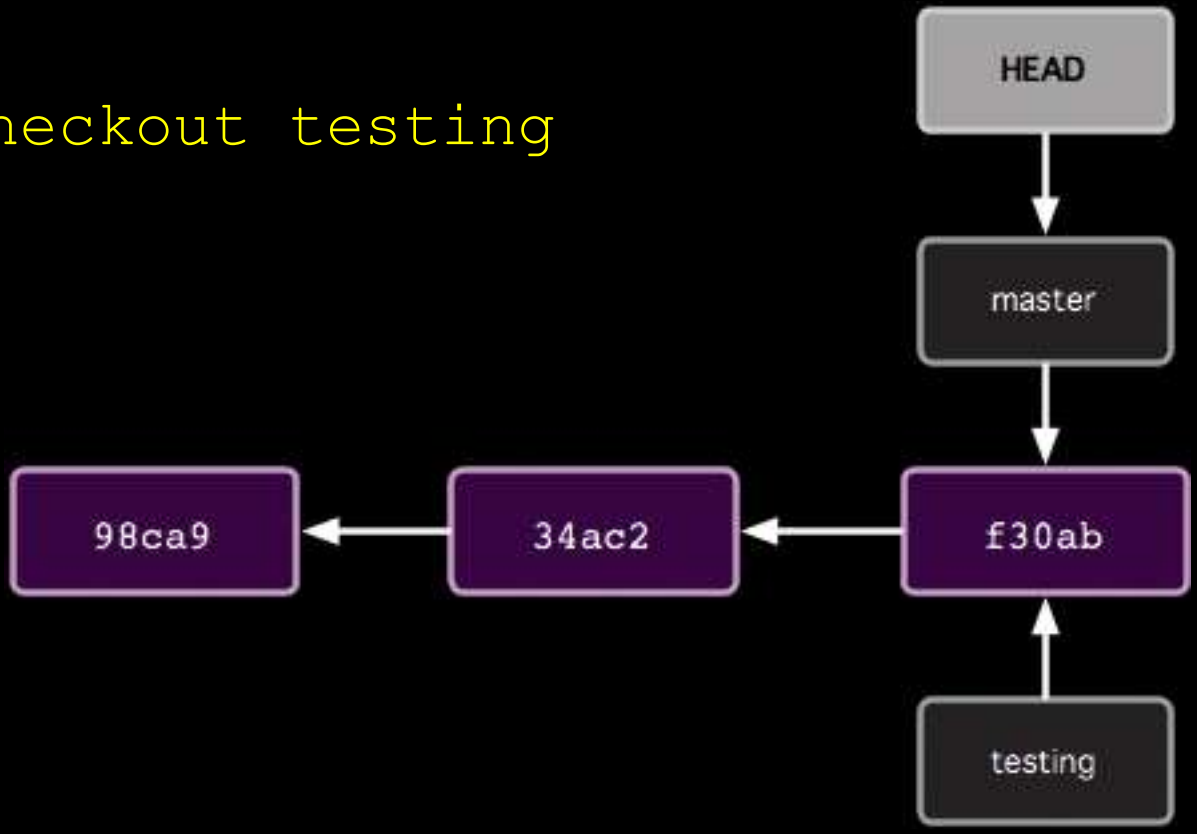
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git checkout works

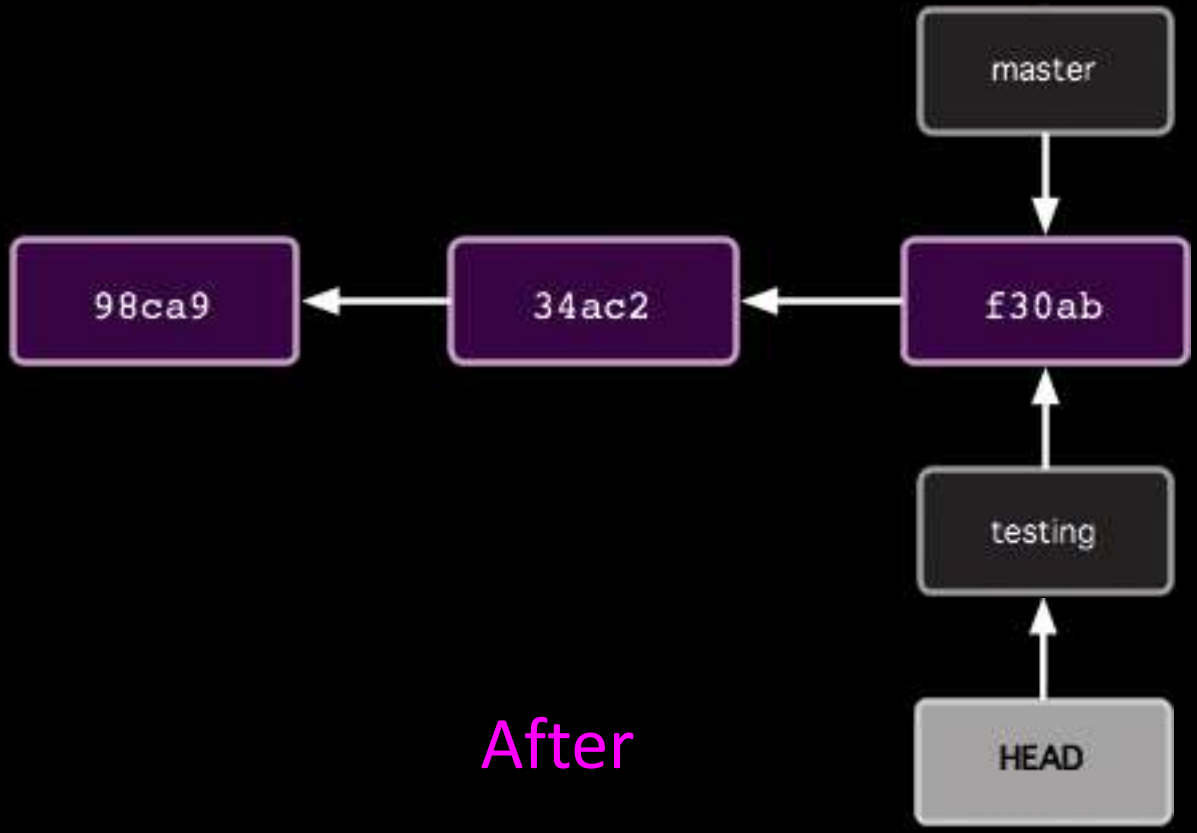
```
$ git checkout testing
```



Before

How git checkout works

```
$ git checkout testing
```



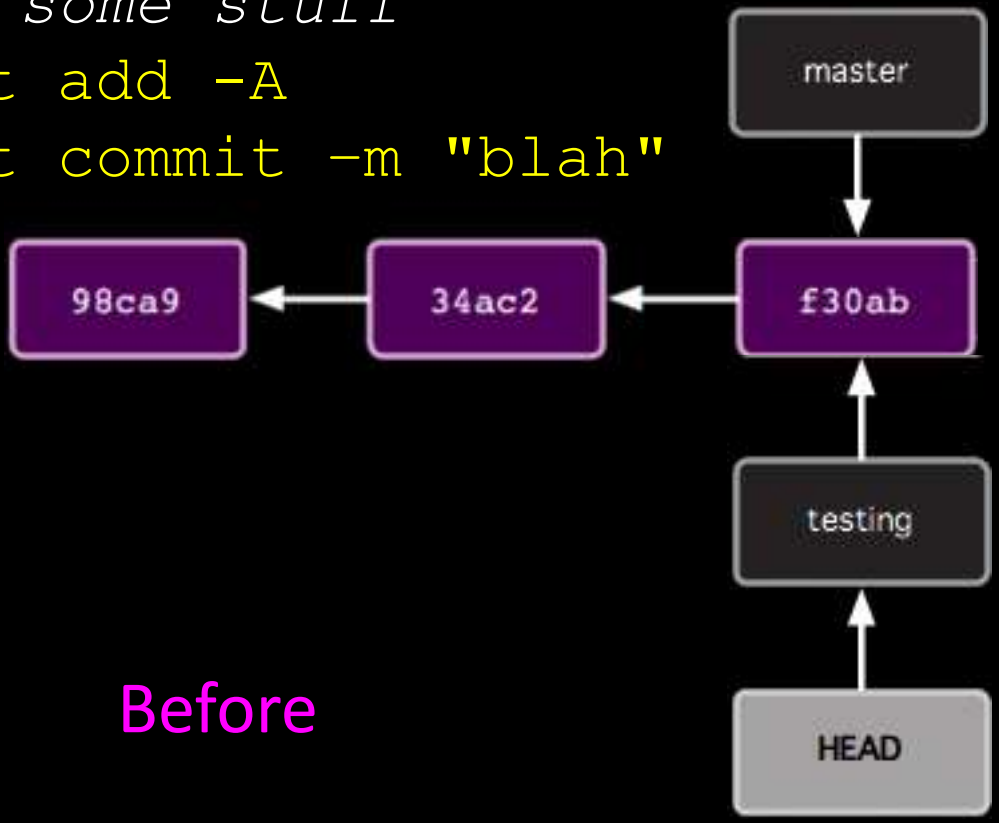
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git commit works with multiple branches

```
Edit some stuff
$ git add -A
$ git commit -m "blah"
```

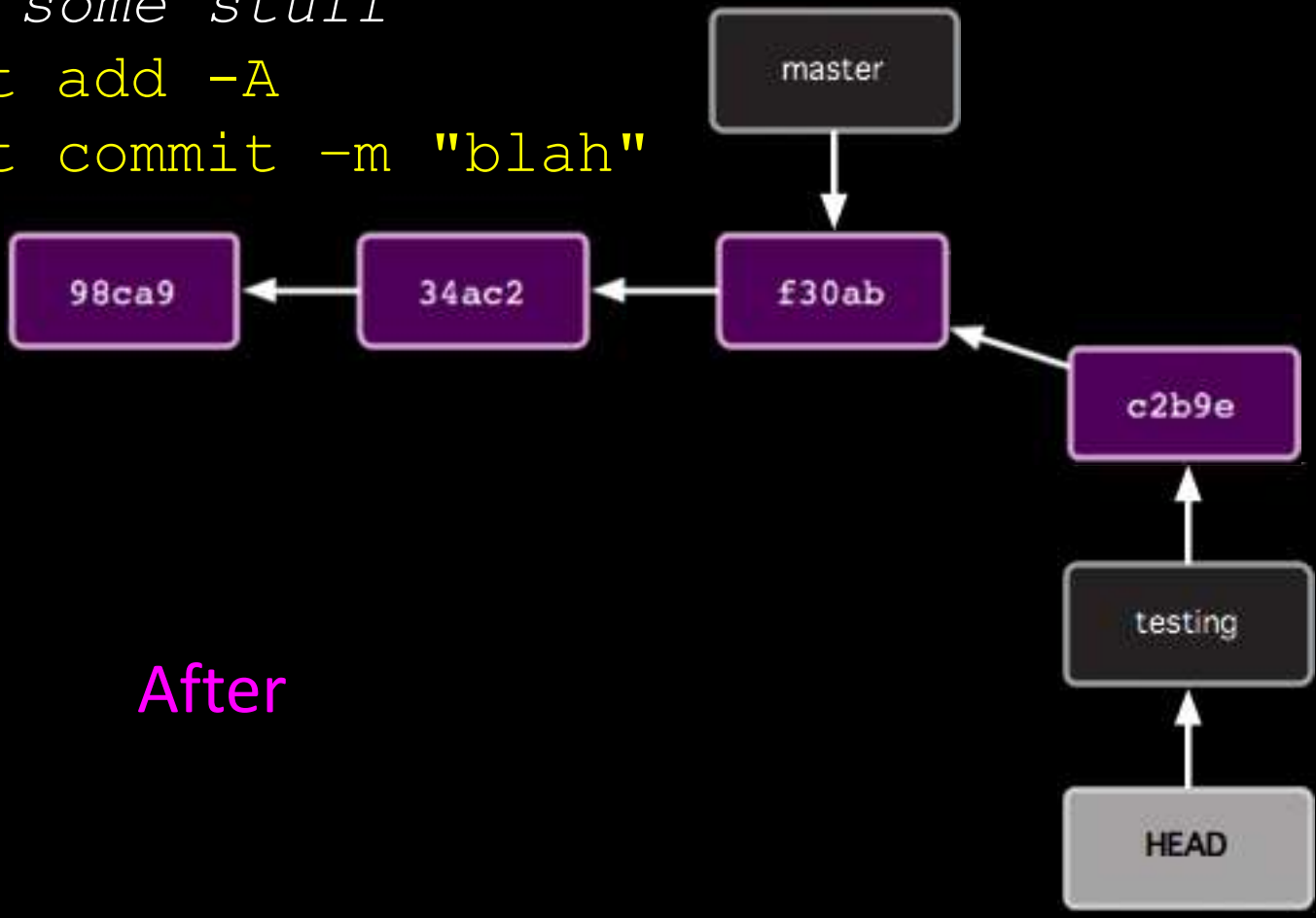


Before

How git commit works with multiple branches

Edit some stuff

```
$ git add -A
$ git commit -m "blah"
```



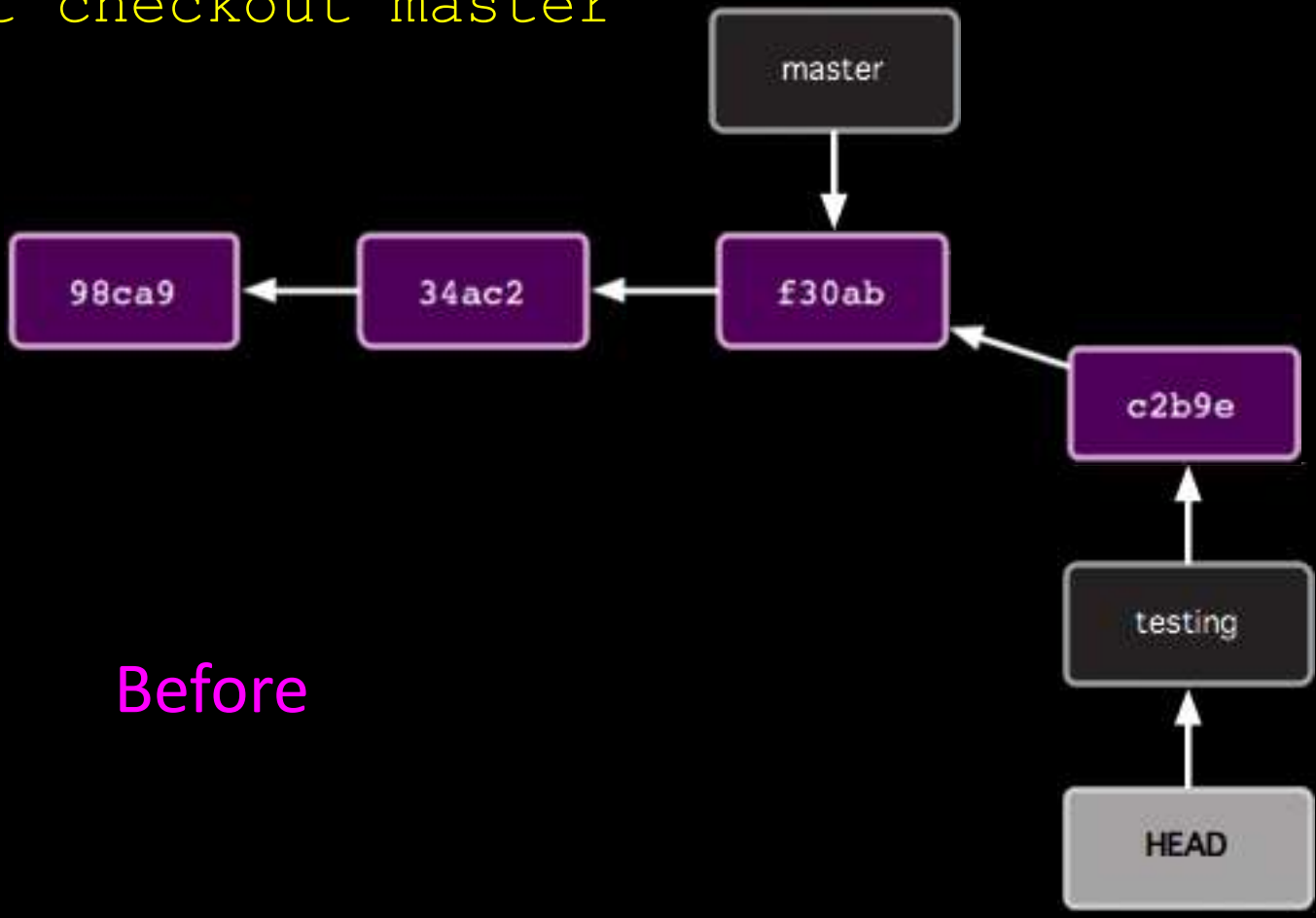
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git checkout works

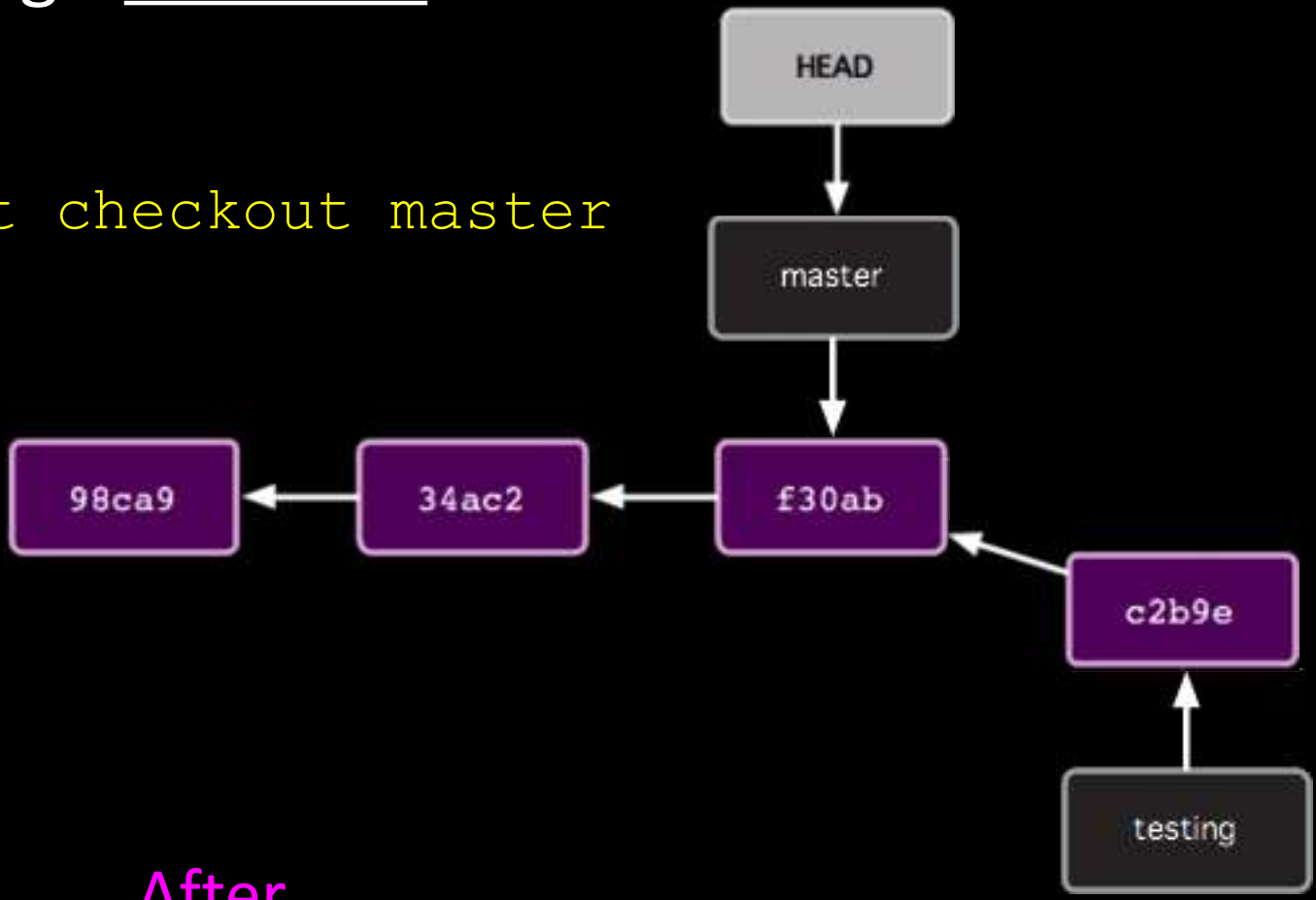
```
$ git checkout master
```



Before

How git checkout works

```
$ git checkout master
```



After

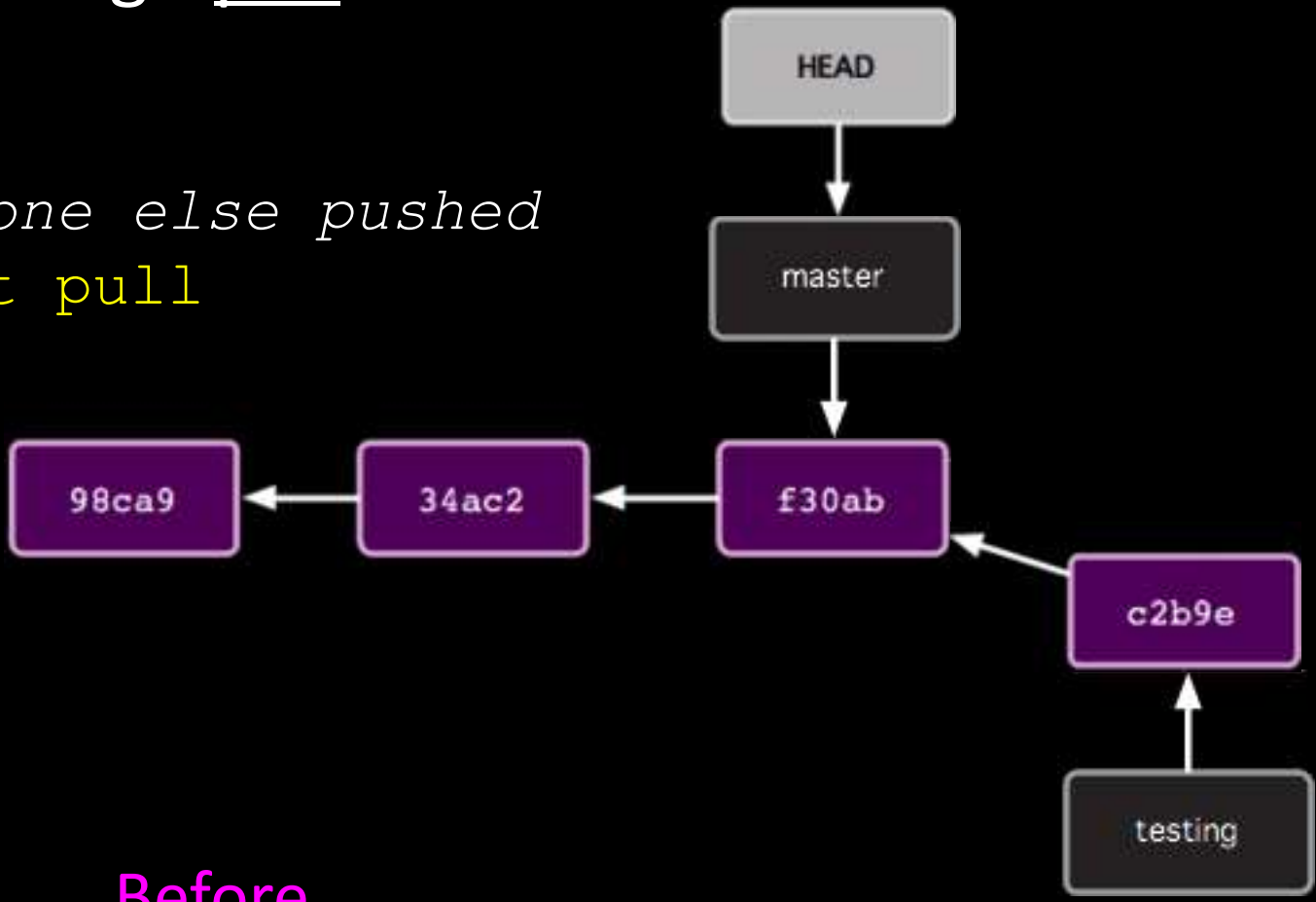
Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git pull works

Someone else pushed

```
$ git pull
```

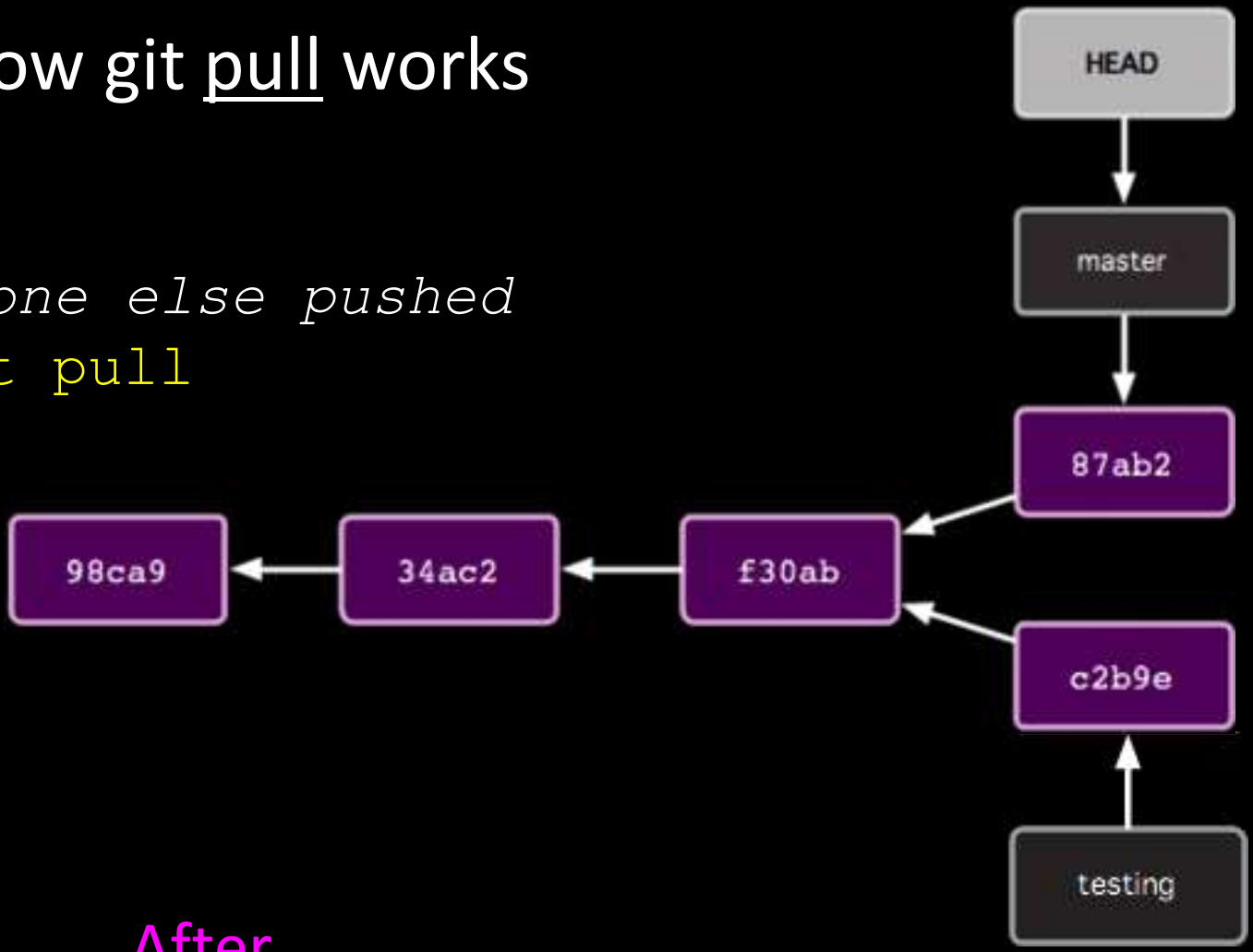


Before

How git pull works

Someone else pushed

```
$ git pull
```



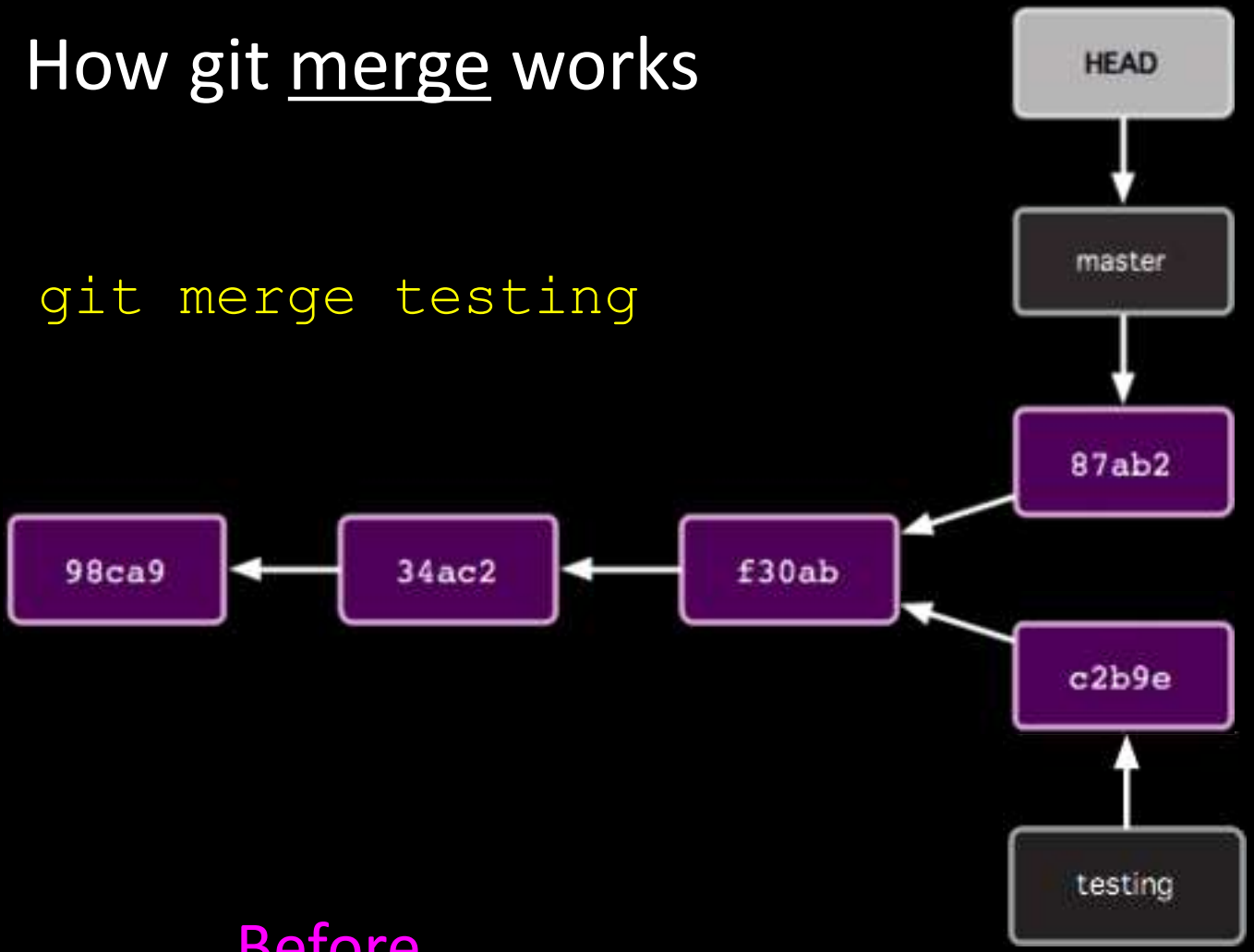
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git merge works

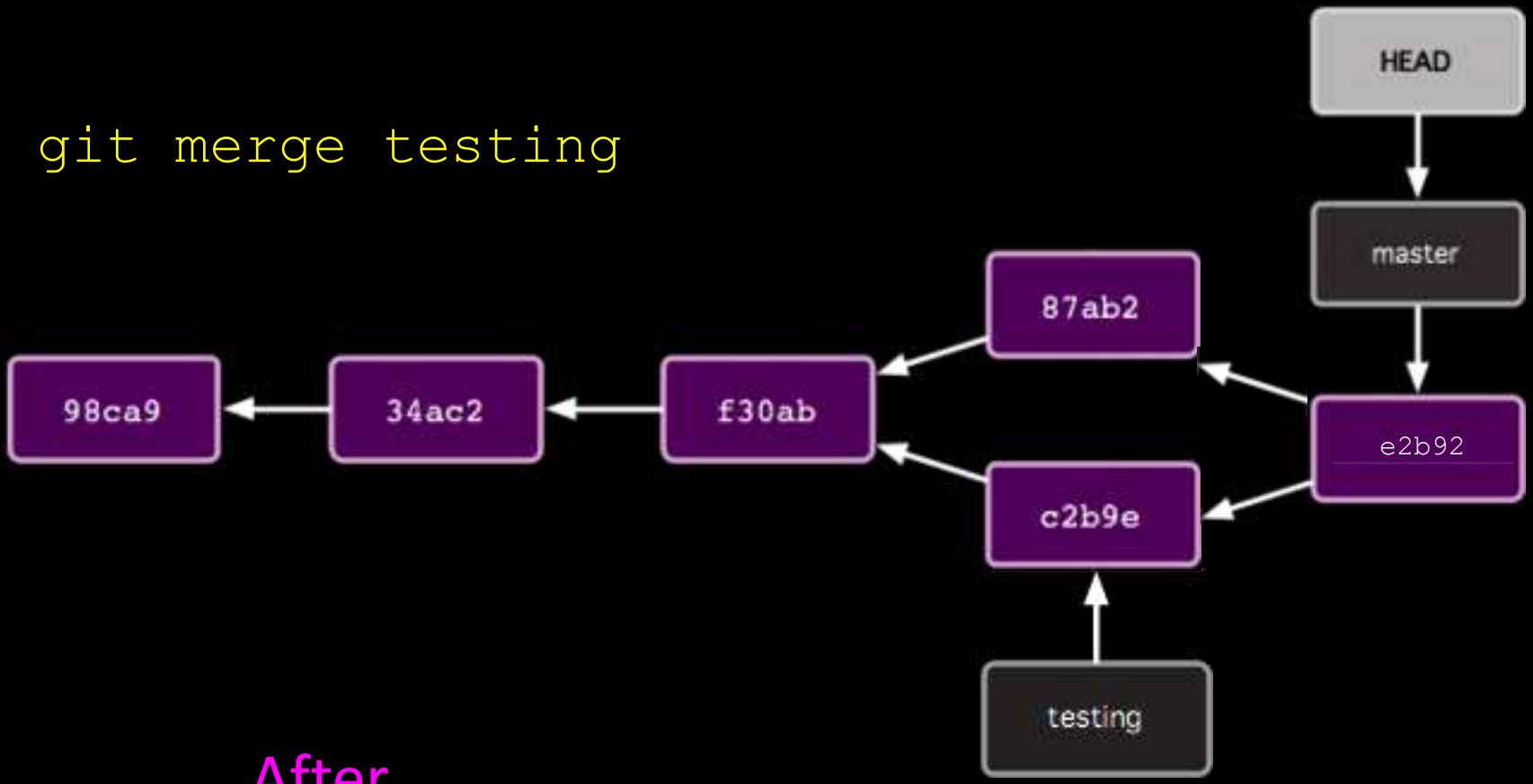
```
$ git merge testing
```



Before

How git merge works

```
$ git merge testing
```



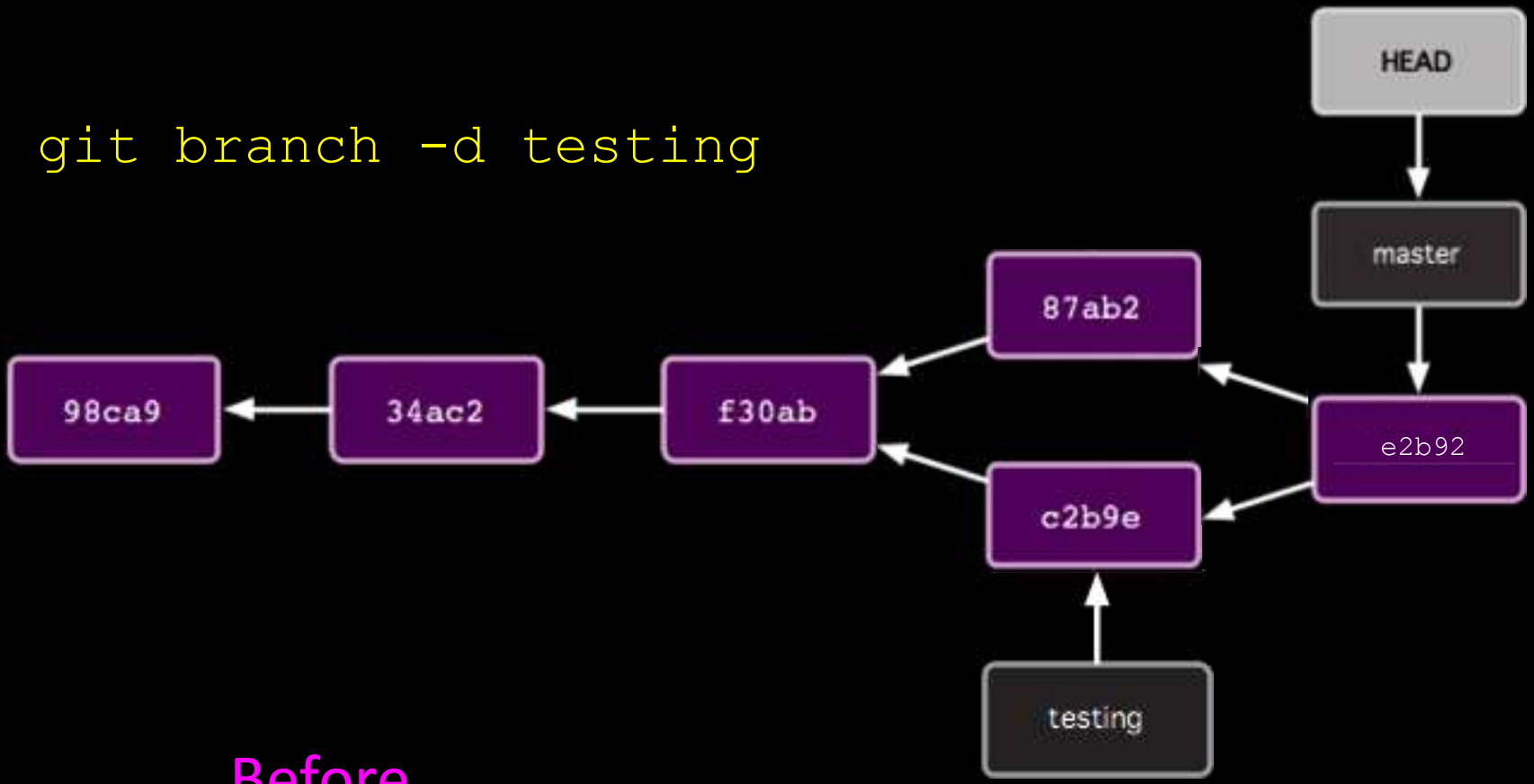
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How to delete branches

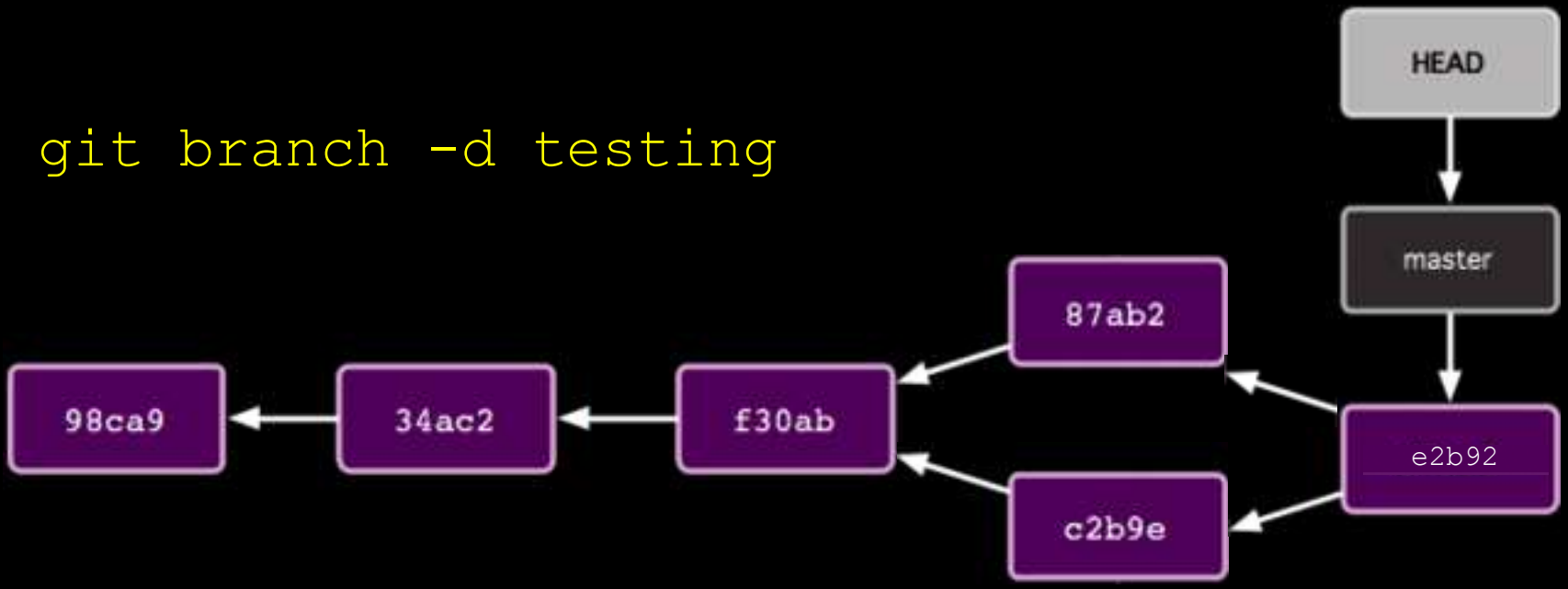
```
$ git branch -d testing
```



Before

How to delete branches

```
$ git branch -d testing
```



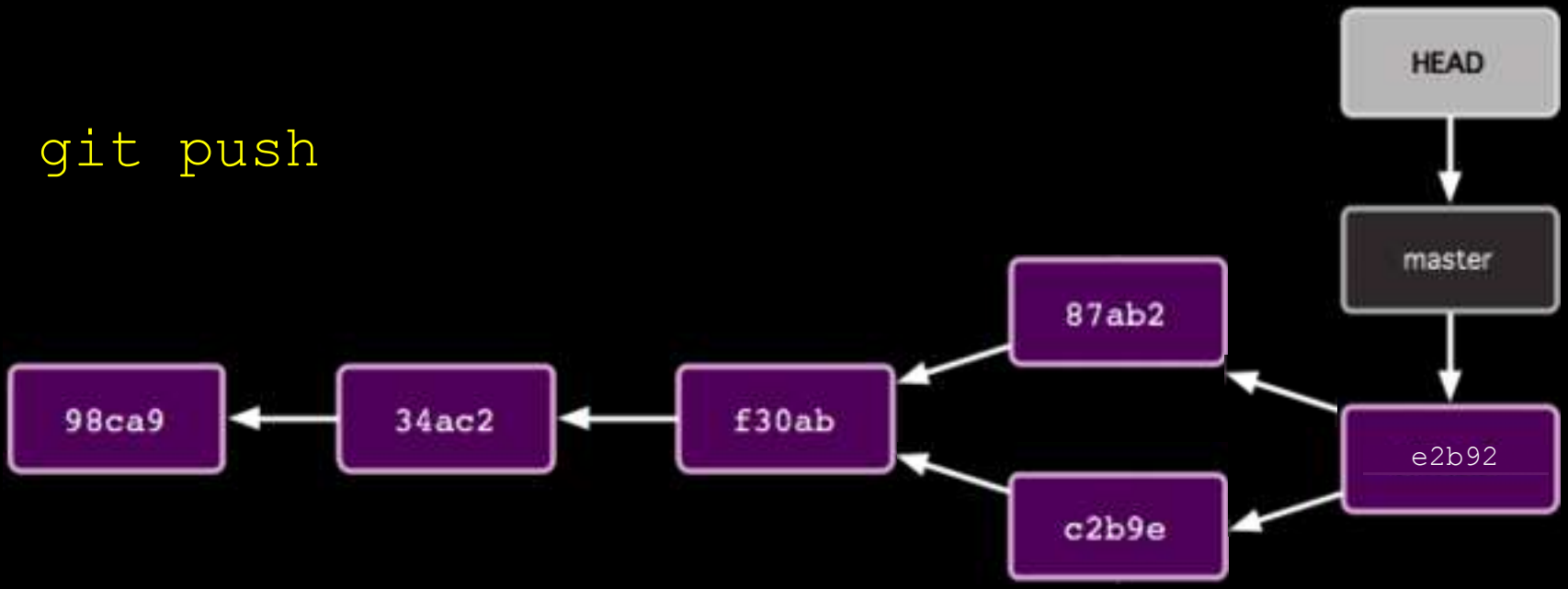
After

Common Workflow

1. Create temp local branch
2. Checkout temp branch
3. Edit/Add/Commit on temp branch
4. Checkout master branch
5. Pull to update master branch
6. Merge temp branch with updated master
7. Delete temp branch
8. Push to update server repos

How git push works

```
$ git push
```



Should update server repos

(if no one else has pushed commits to master branch since last pull)

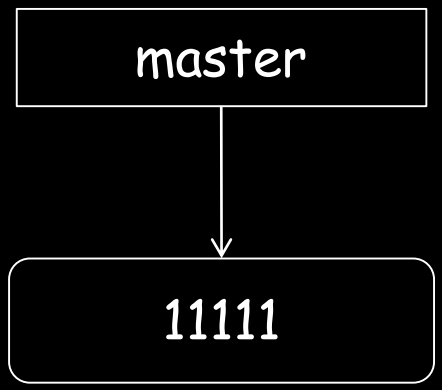
Tips

- git output contains lots of hints
  - git status is your friend!
- Merging may not be as easy as showed
  - E.g.: Multiple collabs updated same parts of file
- Pull before starting temp branch
- Team communication important !

Pop Quiz

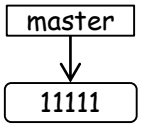
- 5 questions
- Update diagram in each
  - Commit nodes
  - Branch nodes
- Based on actions of Alice and Bob
  - Collaborating via GitHub repo

Start like this



Scott Fleming SF 1

GitHub



Alice

Bob

Question 1

- Alice:
  - $ git clone https://github.com/whatever.git
  - $ cd whatever
- Bob:
  - $ git clone https://github.com/whatever.git
  - $ cd whatever

(include the HEAD node)

Question 2

- Alice:
  - $ git branch myfix
  - $ git checkout myfix
- (Alternatively)
  - $ git checkout -b myfix

Question 3

- Alice:
  - $ rails generate scaffold User …
  - $ git add -A
  - $ git commit -m "Added User" # 22222
- Bob:
  - $ rails generate scaffold Micropost …
  - $ git add -A
  - $ git commit -m "Added Micropost" # 33333

Question 4

- Bob:
  - git push

Question 5

- Alice:
  - git pull

Appendix

What if…

Alice did this:

app/models/micropost.rb

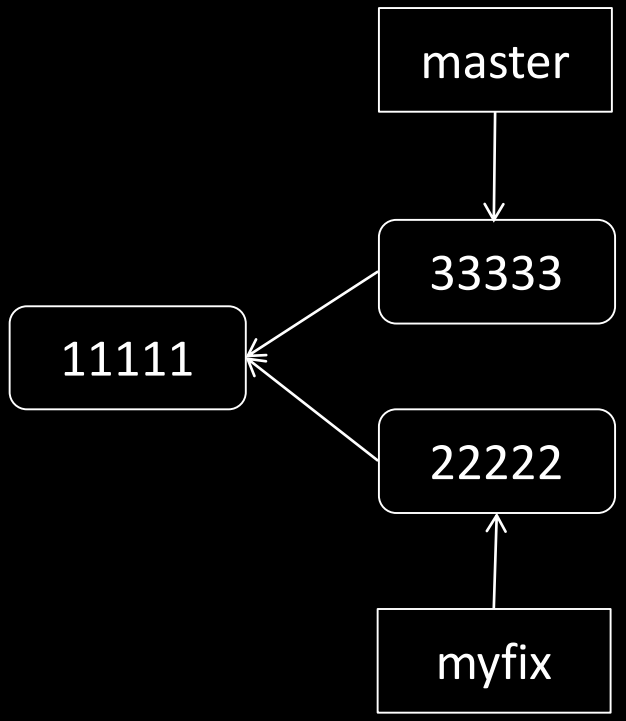
```
class Micropost < ActiveRecord::Base
  validates :content, length: { maximum: 140 }
end
```

Bob did this:

app/models/micropost.rb

```
class Micropost < ActiveRecord::Base
  validates :content, length: { maximum: 120 }
end
```

What if Alice did this?




```
$ git checkout master
$ git merge myfix
```

$ git merge myfix
Auto-merging app/models/micropost.rb
Automatic merge failed; fix conflict and then commit result.

app/models/micropost.rb

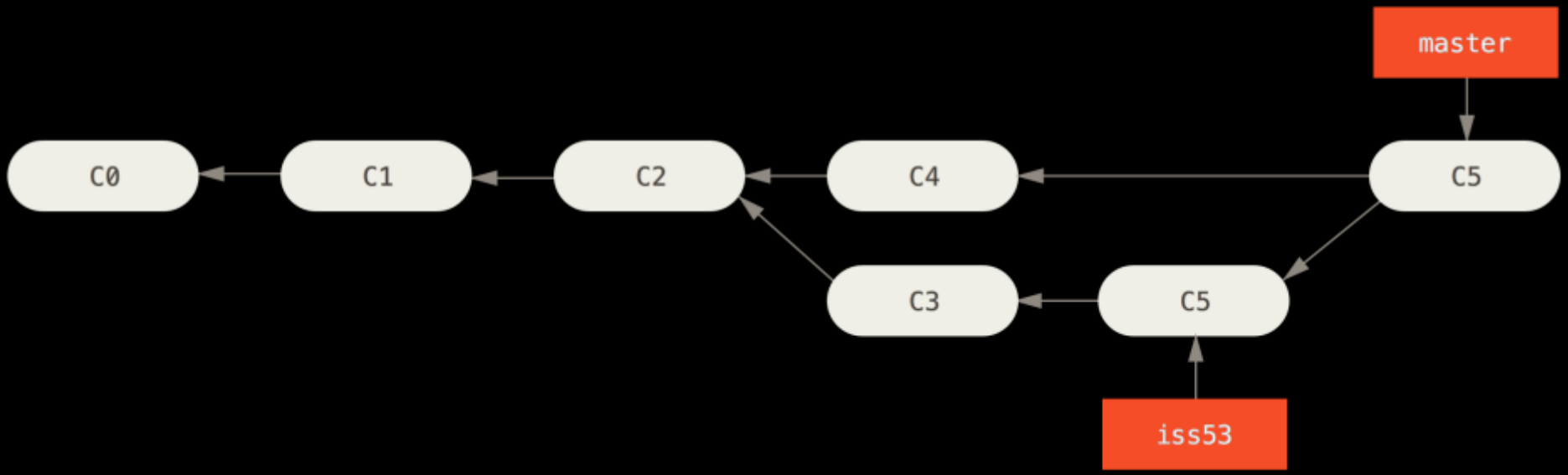
```
class Micropost < ActiveRecord::Base
<<<<<<< HEAD
  validates :content, length: { maximum: 140 }
=======
  validates :content, length: { maximum: 120 }
>>>>>>> myfix
end
```

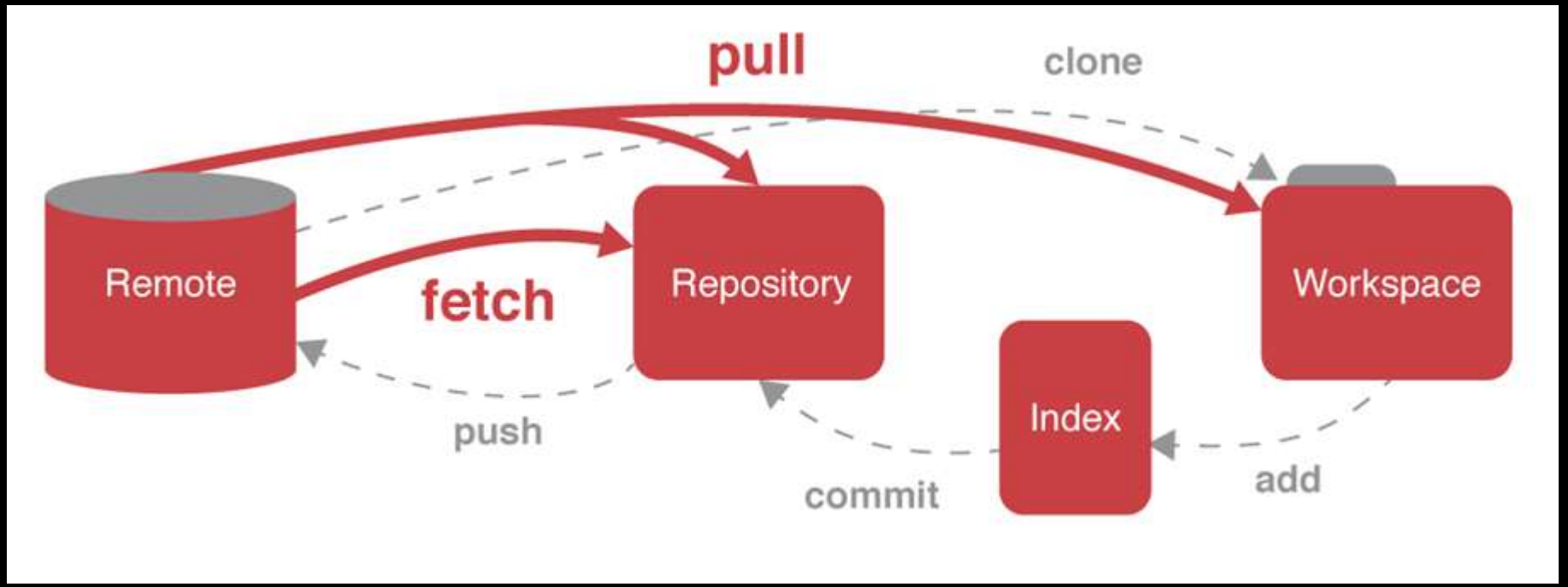
To resolve:
Manually fix the file; git add and commit

Reality

Reality

End


master
C0
C1
C2
C4
C5
C3
C5
iss53


pull
clone
Remote
fetch
Repository
Workspace
push
Index
commit
add