

---

# Lab: Git code versioning

Sprint 1 - Week 2

*INFO 9023 - Machine Learning Systems Design*

*2024 H1*

Thomas Vrancken ([t.vrancken@uliege.be](mailto:t.vrancken@uliege.be))  
Matthias Pirlet ([matthias.pirlet@uliege.be](mailto:matthias.pirlet@uliege.be))

# Introduction

This tutorial is designed for absolute beginners with Git.

If you're already familiar with Git, this tutorial may be a good reminder.

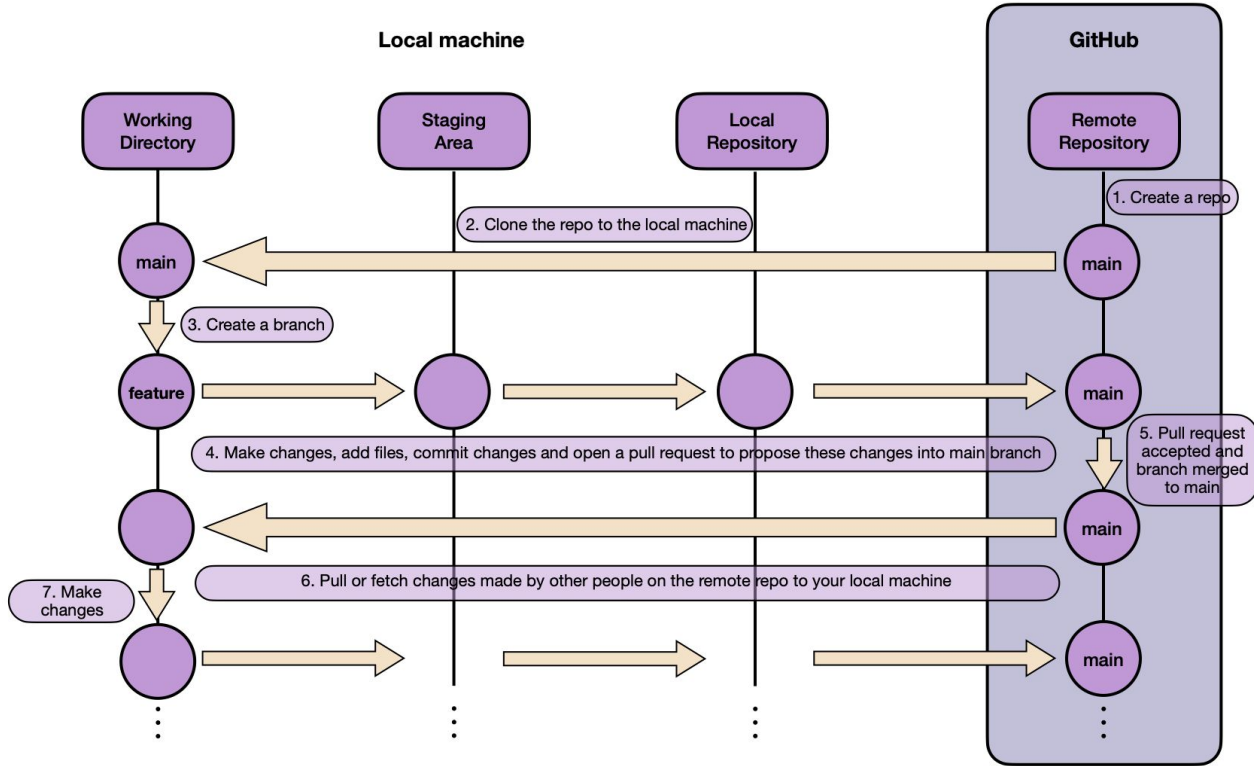


```
final_code.py  
final_code_v2.py  
final_code_last_modif.py  
real_final_code.py  
ready_to_submit_code.py
```

# Installation

## Requirements to make connection with GitHub work

1. Download and install “Git” from <https://git-scm.com/>
2. Check the installation by running “git --version” in your terminal
3. Create your GitHub account on <https://github.com/>
4. Set Up GitHub SSH Key (mandatory to access and write data in repositories on GitHub.com)
  - a. following this tutorial: <https://docs.github.com/en/authentication/connecting-to-github-with-ssh>



# Basic Git commands

**git init** : initializes a brand-new Git repository (locally) and begins tracking an existing directory. It adds a hidden subfolder within the existing directory that houses the internal data structure required for version control (i.e., tracking every change in the different files).

**git clone** : this command creates a local copy of a project that already exists remotely. The clone includes all the project's files, history, and branches.

**git add** : stages a change. Git tracks changes to a developer's codebase, but it's necessary to stage and take a snapshot of the changes to include them in the project's history. This command performs staging, the first part of that two-step process. Any changes that are staged will become a part of the next snapshot and a part of the project's history. Staging and committing separately gives developers complete control over the history of their project without changing how they code and work.

**git commit** : saves the snapshot to the project history and completes the change-tracking process. This is the **second part** of the two-step process. In short, a commit functions like taking a photo. Anything that's been staged with **git add** will become a part of the snapshot with **git commit**.

# Basic Git commands (1)

`git status` : shows the status of changes as untracked, modified, or staged.

`git branch` : shows the branches being worked on locally.

`git checkout` : switch to another branch and check it out into your working directory.

`git merge` : merges lines of development together. This command is typically used to combine changes made on two distinct branches. For example, a developer would merge when they want to combine changes from a feature branch into the main branch for deployment.

`git pull` : updates the local line of development with updates from its remote counterpart. Developers use this command if a teammate has made commits to a branch on a remote, and they would like to reflect those changes in their local environment.

`git push` : updates the remote repository with any commits made locally to a branch.

## Basic Git commands (2)

`git rebase [base_branch]` : modifies the commit history to create a linear project history. It integrates changes from one branch into another while maintaining a clean and chronological sequence of commits. `git rebase [base_branch]` transfers changes from the current branch onto another branch (`base_branch`).

Useful for keeping the commit history clean and avoiding unnecessary merge commits to prepare a clean branch for a pull request.

**Caution:** Rewriting history can cause conflicts and should be used with care, especially in shared branches.

`git fetch` : retrieves changes from a remote repository without merging them into the local branch. When downloading content from a remote repo, `git pull` and `git fetch` commands are available to accomplish the task. You can consider `git fetch` the 'safe' version of the two commands.

## Basic Git commands (3)

**pull request** : A pull request (PR) is a mechanism for proposing changes to a codebase hosted on platforms like GitHub, GitLab, or Bitbucket. This facilitates collaboration and code review in a distributed development environment. The workflow is the following:

1. Developer forks the repository or creates a branch.
2. Developer makes changes in their branch.
3. Developer opens a pull request to propose the changes.
4. Team members review the changes, discuss, and suggest modifications.
5. Once approved, changes are merged into the target branch.

Cheat sheet: <https://education.github.com/git-cheat-sheet-education.pdf>

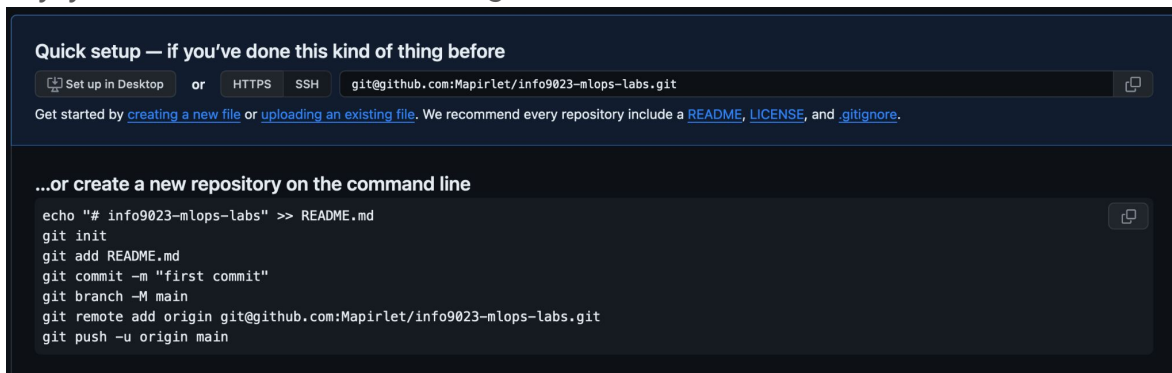


# Start a new repository and publish it on GitHub





1. Open “Terminal”
2. Go to your newly created folder where you will put your codes, data, ... , e.g.

```
cd /Users/Username/Desktop/Projects/GitHub/info9023-mlops-labs
```

3. On GitHub create a new repository e.g., “info9023-mlops-labs”
4. In this repository, you should see the following:



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

 Set up in Desktop or  HTTPS  SSH `git@github.com:Mapirlet/info9023-mlops-labs.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 "# info9023-mlops-labs" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:Mapirlet/info9023-mlops-labs.git
git push -u origin main
```

# Start a new repository and publish it on GitHub (1)

5. Select SHS and not HTTPS and then copy and paste the given commands:

```
echo #info9023-mlops-labs">>README.md
git init
git add README.md
git commit -m "Add first commit"
git branch -M main
git remote add origin git@github.com:Username/info9023-mlops-labs.git
git push -u origin main
```

# Contribute to an existing repository

1. Fork the project.
2. Clone this project to get it on your computer and be able to modify it.

```
git clone https://github.com/Username/repo.git  
cd repo  
git branch my-branch  
git checkout my-branch  
git add file1.md file2.md  
git commit -m "my snapshot"  
git push --set-upstream origin my-branch
```

3. Create a pull request to push your modifications on the project



Playlist publique

# The GIT Playlist

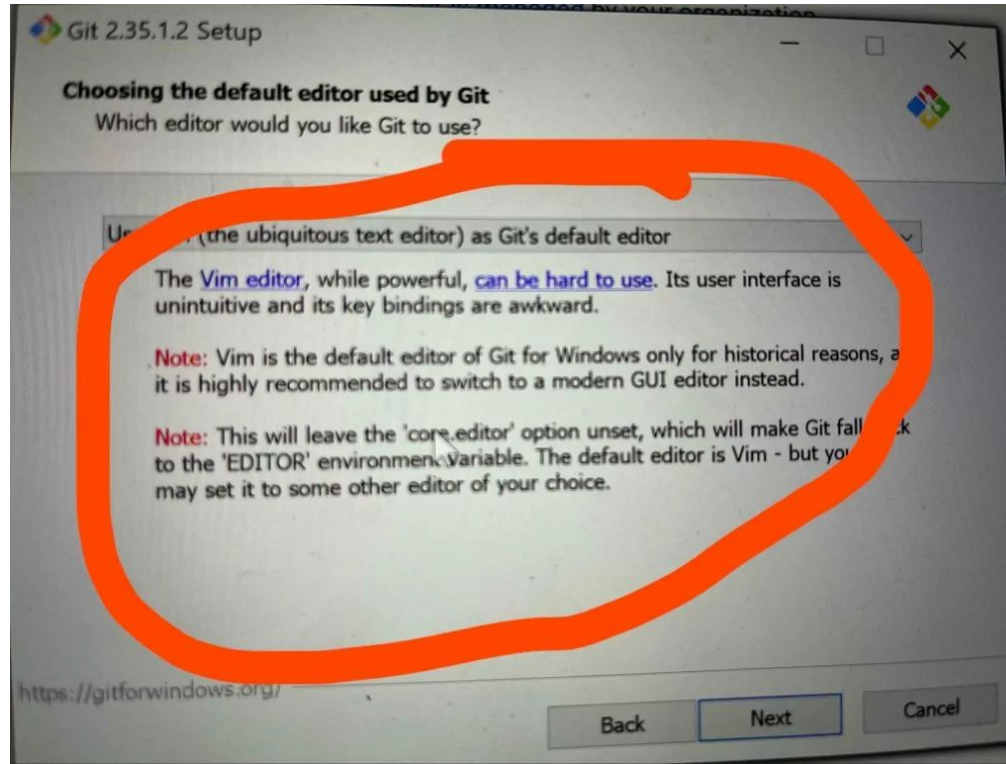
Matthias Pitt • 5 likes • 13 titres, 43 min 48 s



Tri personnalisé

#	Titre	Album	Date d'ajout	
1	<b>Init</b> Full dub	Rewind	il y a 3 jours	2:38
2	<b>Fork</b> e Chainz	B.O.A.T.S. II #METIME (Deluxe)	il y a 3 jours	3:47
3	<b>Clone</b> Tracelab, The Blaze, Leo Boy, Lexi, LiTP	Clone	il y a 3 jours	4:07
4	<b>Fetch</b> S-Type	Fetch	il y a 3 jours	2:39
5	<b>Pull</b> Pull	Pull	il y a 3 jours	3:28
6	<b>add</b> rody dünyada	add	il y a 3 jours	2:26
7	<b>Commit</b> Ten Bulls	The Physician's Magician	il y a 3 jours	2:35
8	<b>Push (feat. Omah Lay)</b> Naomi Sharon, Omah Lay	Obsidian	il y a 3 jours	2:43
9	<b>Merge</b> Basscannon, Rexalted	Merge	il y a 3 jours	6:17
10	<b>Conflict</b> Harmeet Aulakh, Rav Hanjra, The Kidd	Conflict	il y a 3 jours	2:38
11	<b>Pull Request</b> Alarm! Alarm!	Whatever...	il y a 3 jours	1:46
12	<b>Rebase</b> LaCargo	Tunnel ClassIX, Vol. 2	il y a 2 jours	6:13
13	<b>REVERT</b> Shiv Rathi	REVERT	il y a 3 jours	2:31

# Git UIs



# Git UIs

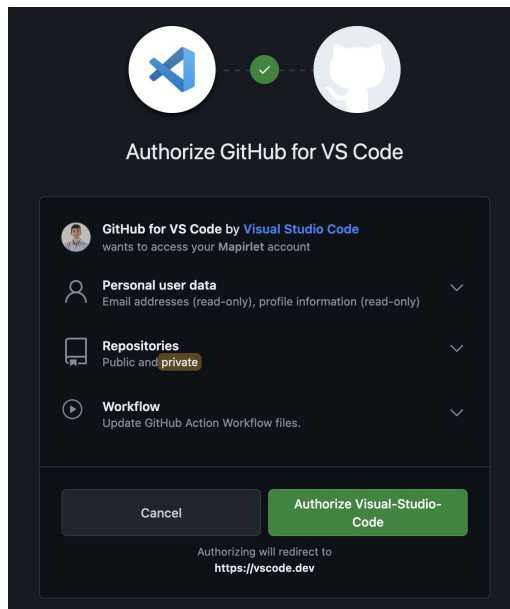
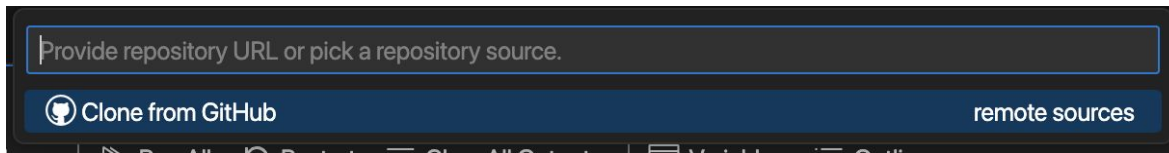
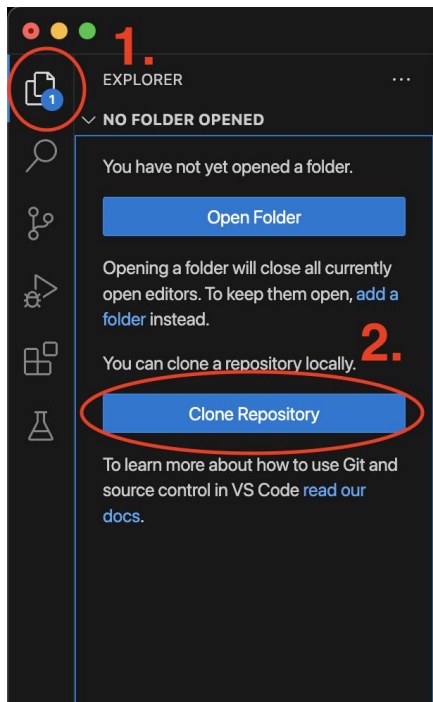
<https://desktop.github.com/>

PyCharm for code versioning with GitHub:

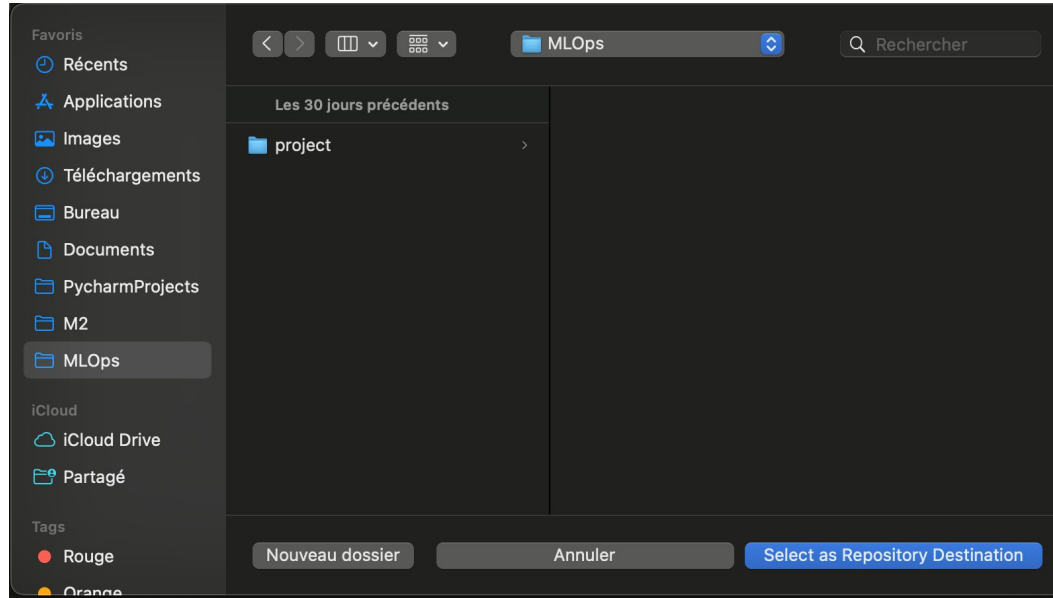
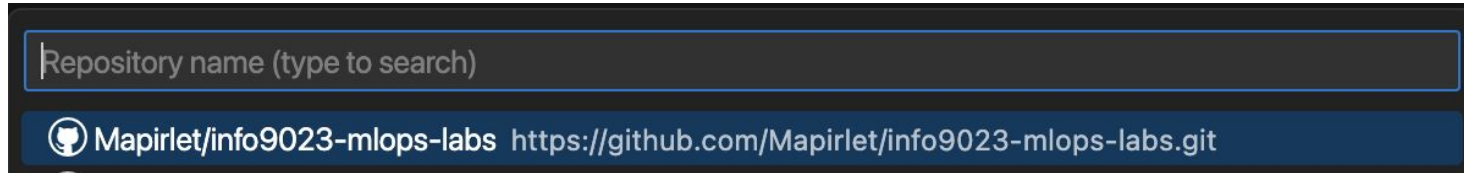
<https://www.jetbrains.com/help/pycharm/manage-projects-hosted-on-github.html>

VSCode for code versioning with GitHub: <https://vscode.github.com/>

# VSCode: clone a project

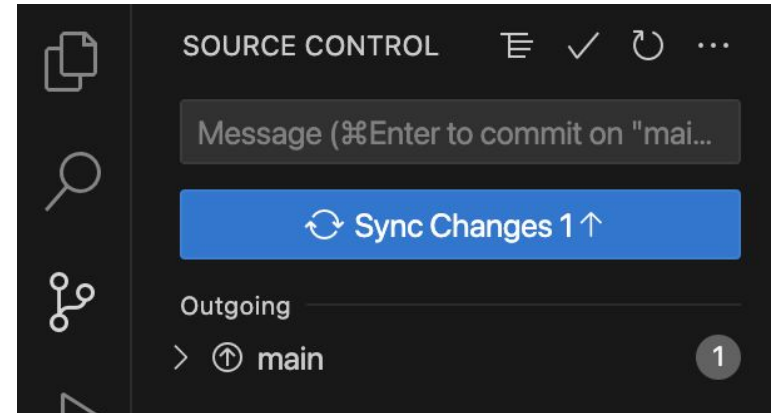
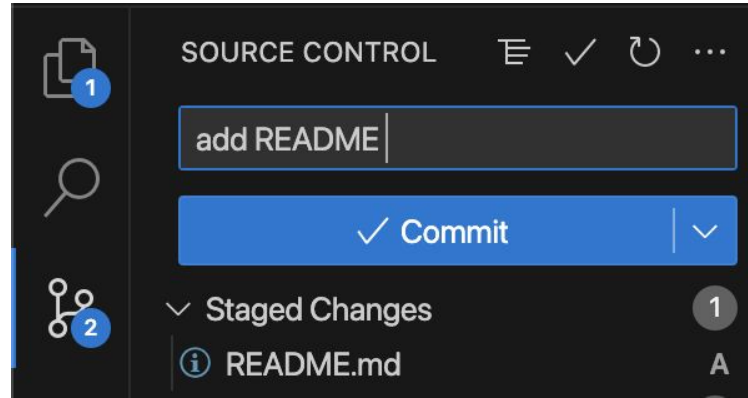
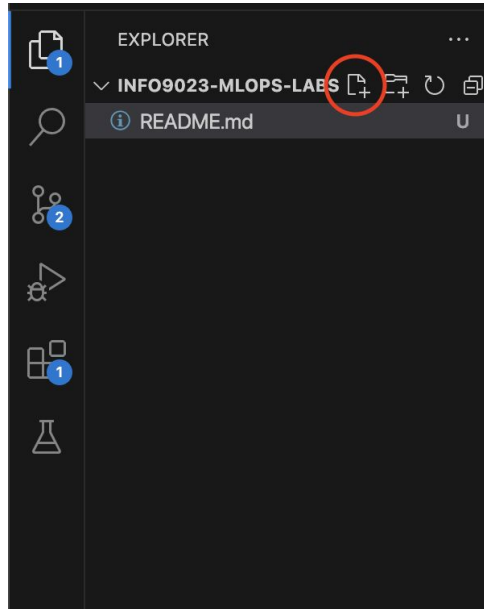


# VSCode: clone a project (1)

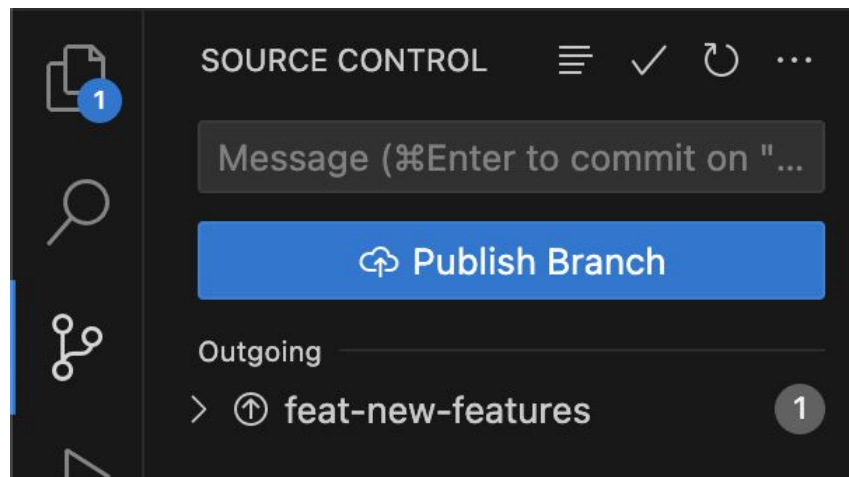
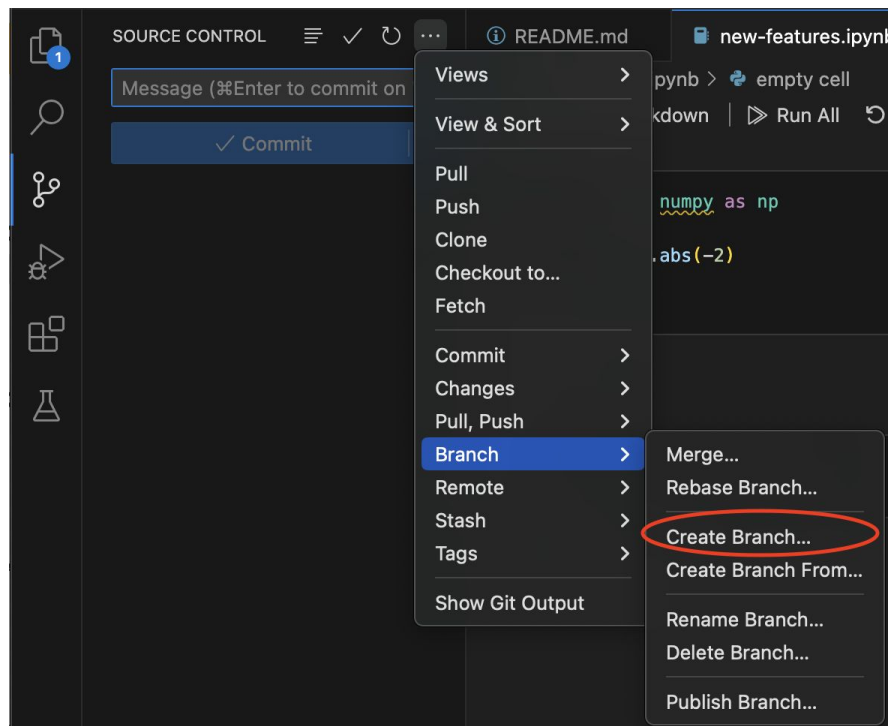




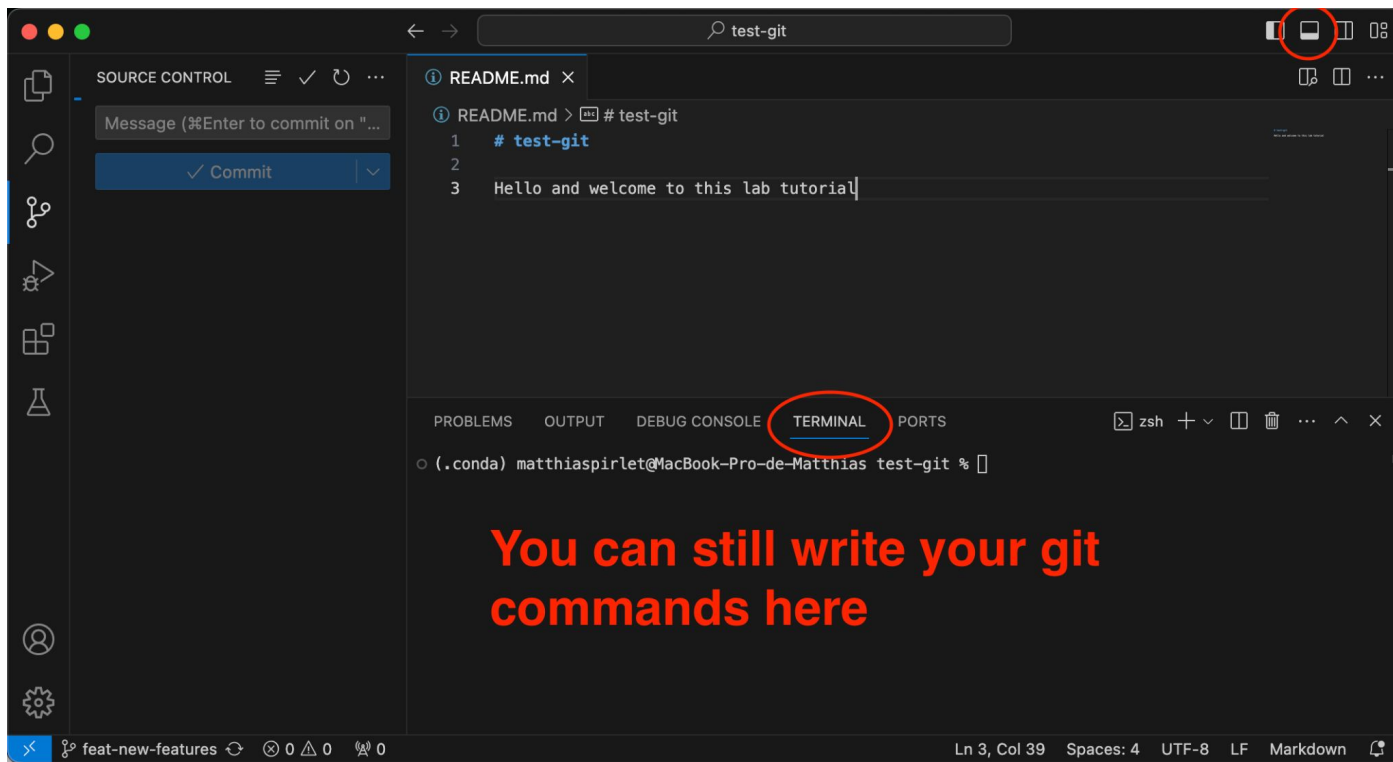
## VSCode: clone a project (2)



# VSCode: create a new branch



# VSCode: terminal and git commands



# In the event of false manipulation or error

If you've done something you shouldn't have:

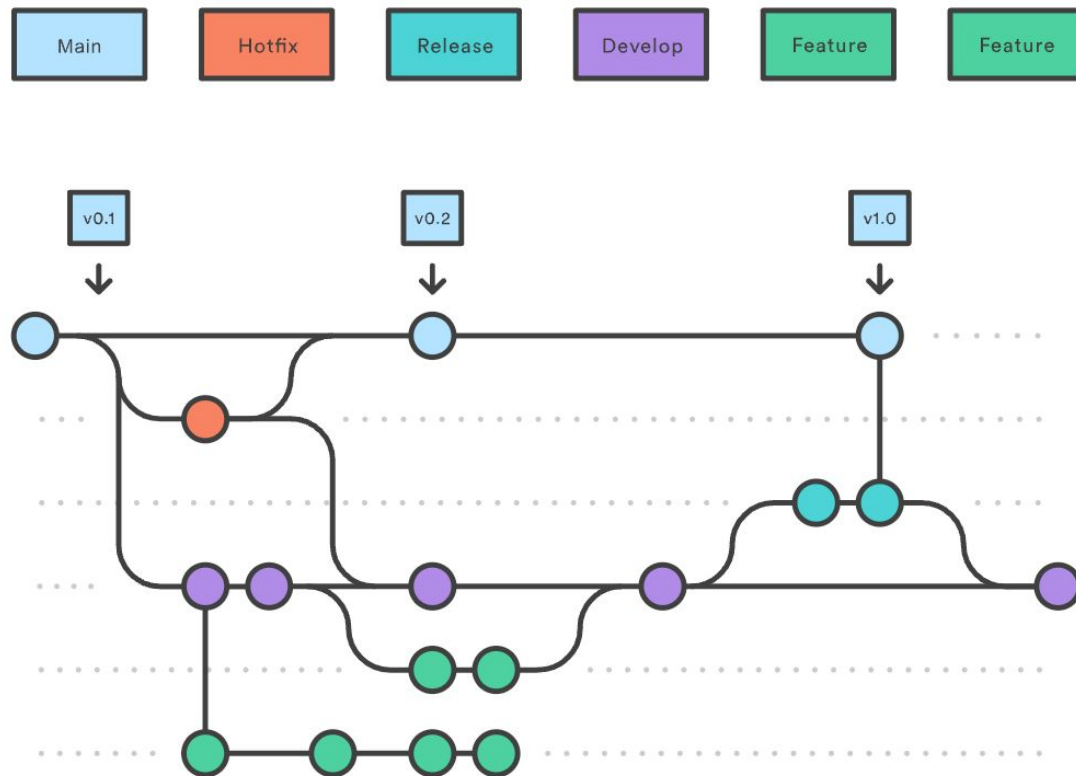
<https://ohshitgit.com/>

This is the solution to all your problems.



# GitFlow

## Reminder



# GitFlow (1)

## Installation

- Windows : already installed when you install latest version of Git.
- MacOS : `$ brew install git-flow`
- Linux : `$ apt-get install git-flow`

# GitFlow (2)

## Initialization

Create your project in GitHub, create your local folder and

```
$ cd my/project
```

```
$ git init
```

```
$ git add .
```

```
$ git commit -am "Initial commit"
```

```
$ git remote add origin git@github.com:username/Project-Name.git
```

```
$ git push -u origin main
```

# GitFlow (3)

## Features

Prepare repository for development :

```
$ git checkout -b develop
```

```
$ git push -u origin develop
```

Setup Git Flow :

```
$ git flow init -d
```

Developing features :

```
$ git flow feature start my-feature
```

```
$ git commit -am "Pithy commit message"
```



# GitFlow (4)

## Release

When finished :

```
$ git flow feature finish my-feature
```

## Releasing

Before releasing, check for the existence of known tags so you **know** what to name the release :

```
$ git fetch --tags
```

Start release :

```
$ git flow release start 1.0
```

# GitFlow (5)

## Release

Update README and commit using appropriate commit message

```
$ git commit -am "Bumped version number to 1.0"
```

Make any last minute updates/changes. NO FEATURE DEVELOPMENT!:

```
$ git flow release finish 1.0
```

Update remotes :

```
$ git push origin main
```

```
$ git push origin develop
```

```
$ git push --tags
```

# GitFlow (6)

## Hotfixes

If you need to make a fix on a live version

```
$ git flow hotfix start 1.0.1
```

Update the README file and commit:

```
$ git commit -am "Bumped version number to 1.0.1"
```

Fix the issue and commit those changes :

```
$ git commit -am "Fixed major issue with site!"
```

Finish the hotfix :

```
$ git flow hotfix finish 1.0.1
```

# GitFlow (7)

## Hotfixes

Finally, update the remotes :

```
$ git push origin main
```

```
$ git push origin develop
```

```
$ git push --tags
```