

Prise en main de *VsCode*, compilation/exécution d'un programme C

Sébastien Jean

IUT de Valence
Département Informatique

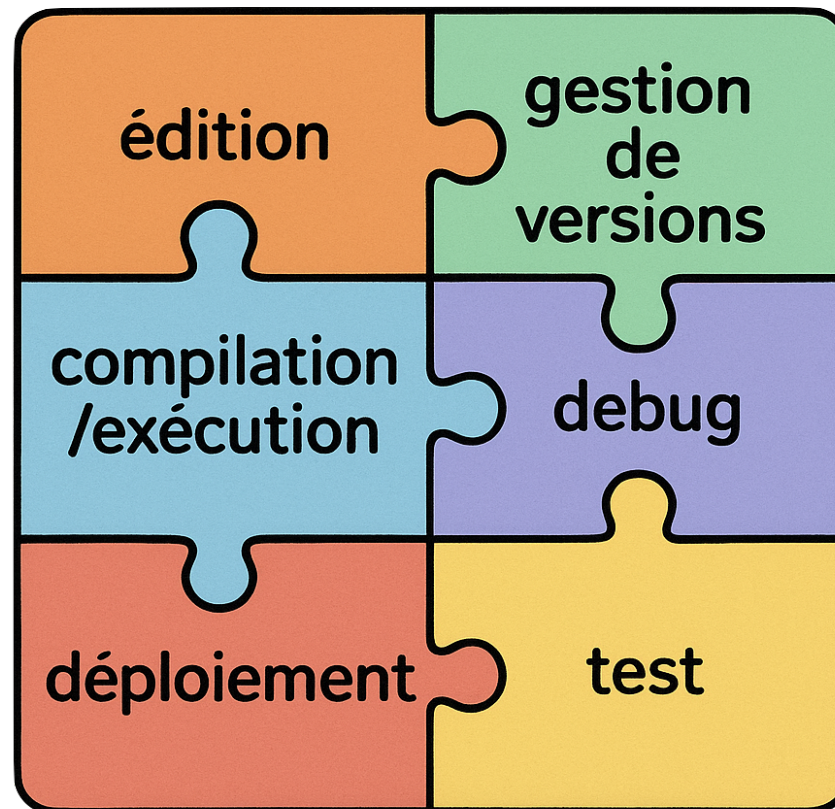
v1.0, 17 septembre 2025

- Qu'est ce qu'un environnement de développement intégré (**IDE**) ?



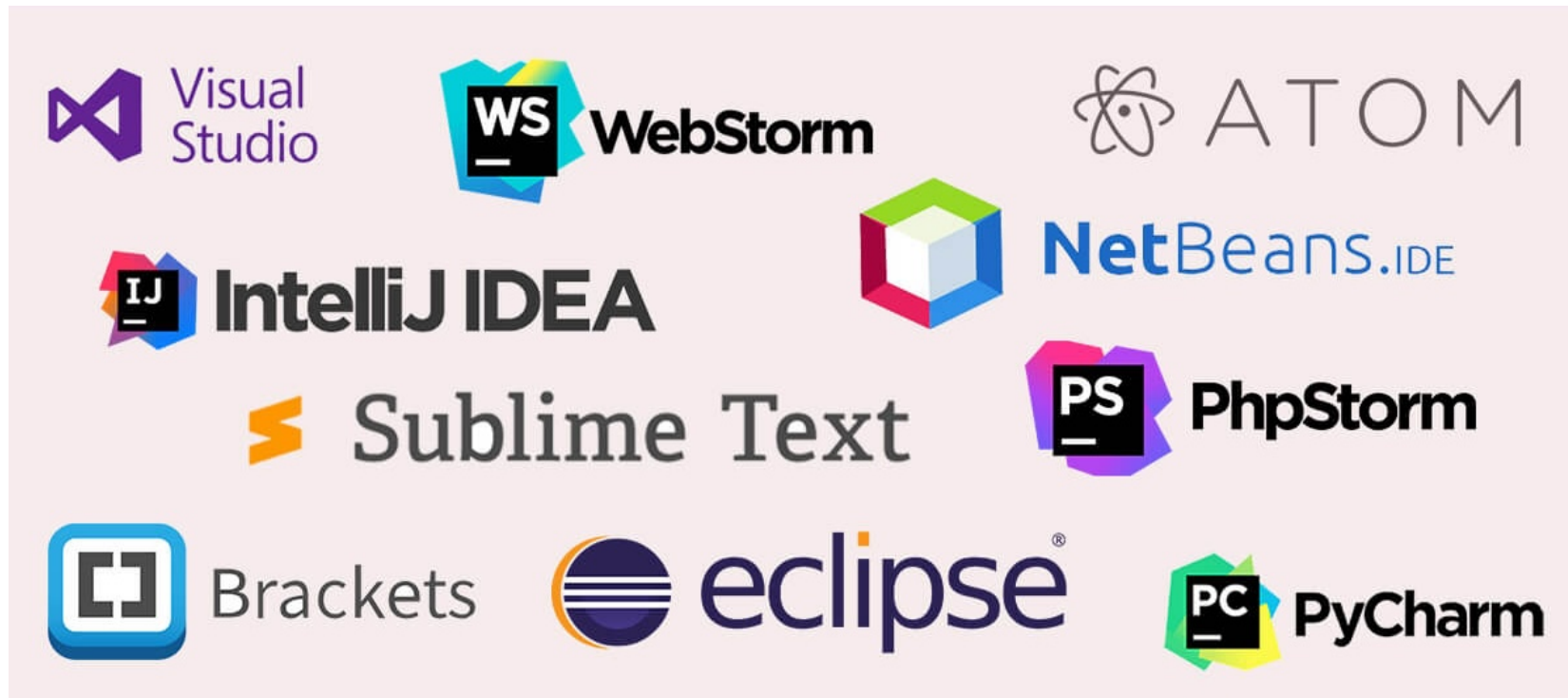
IDE

- Un IDE est un **atelier** qui **rassemble** les principaux **outils** permettant la **production de logiciel**



- Un IDE est **configurable** et **extensible**

Quelques IDE ou simples

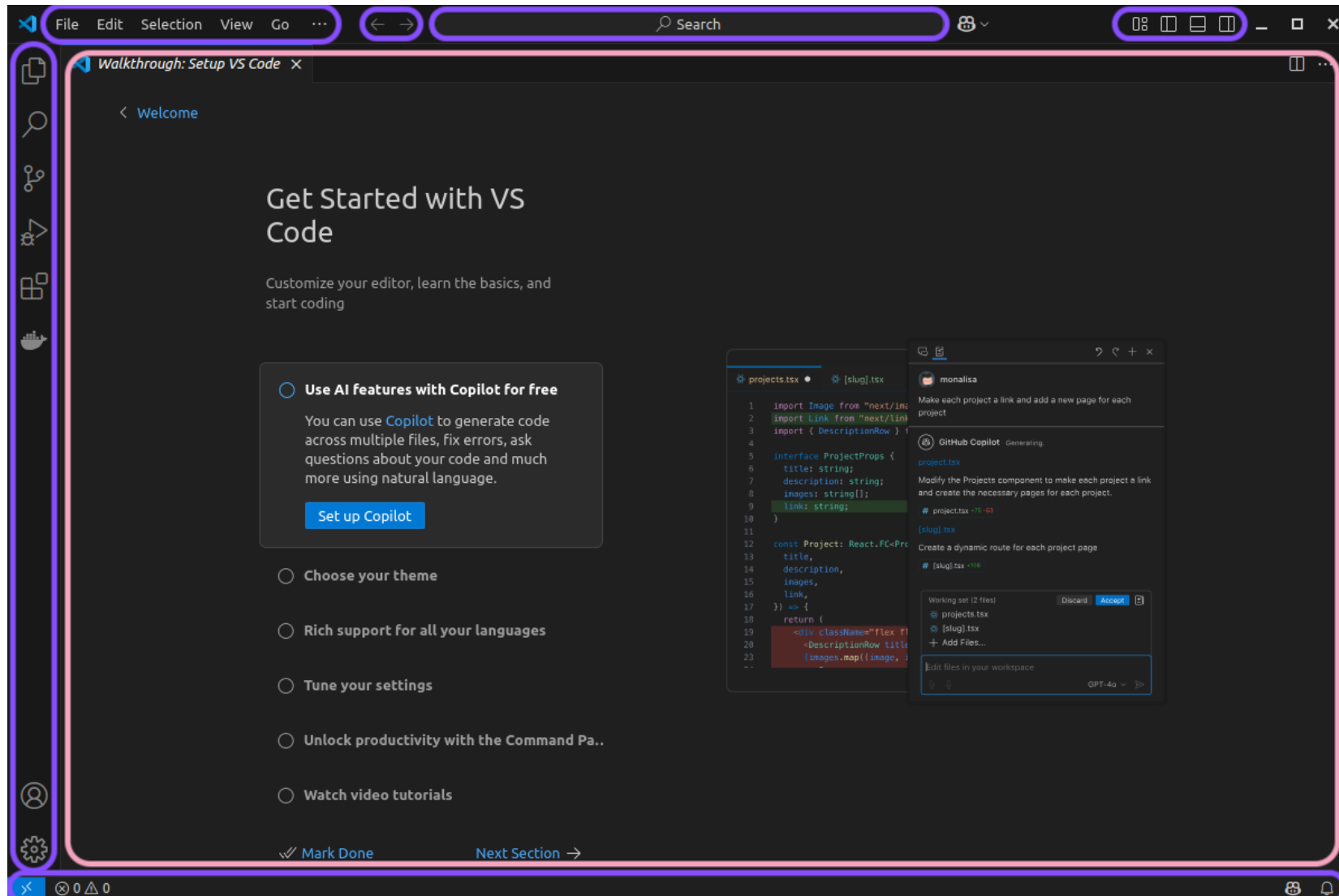




- **Exécuter** VsCode

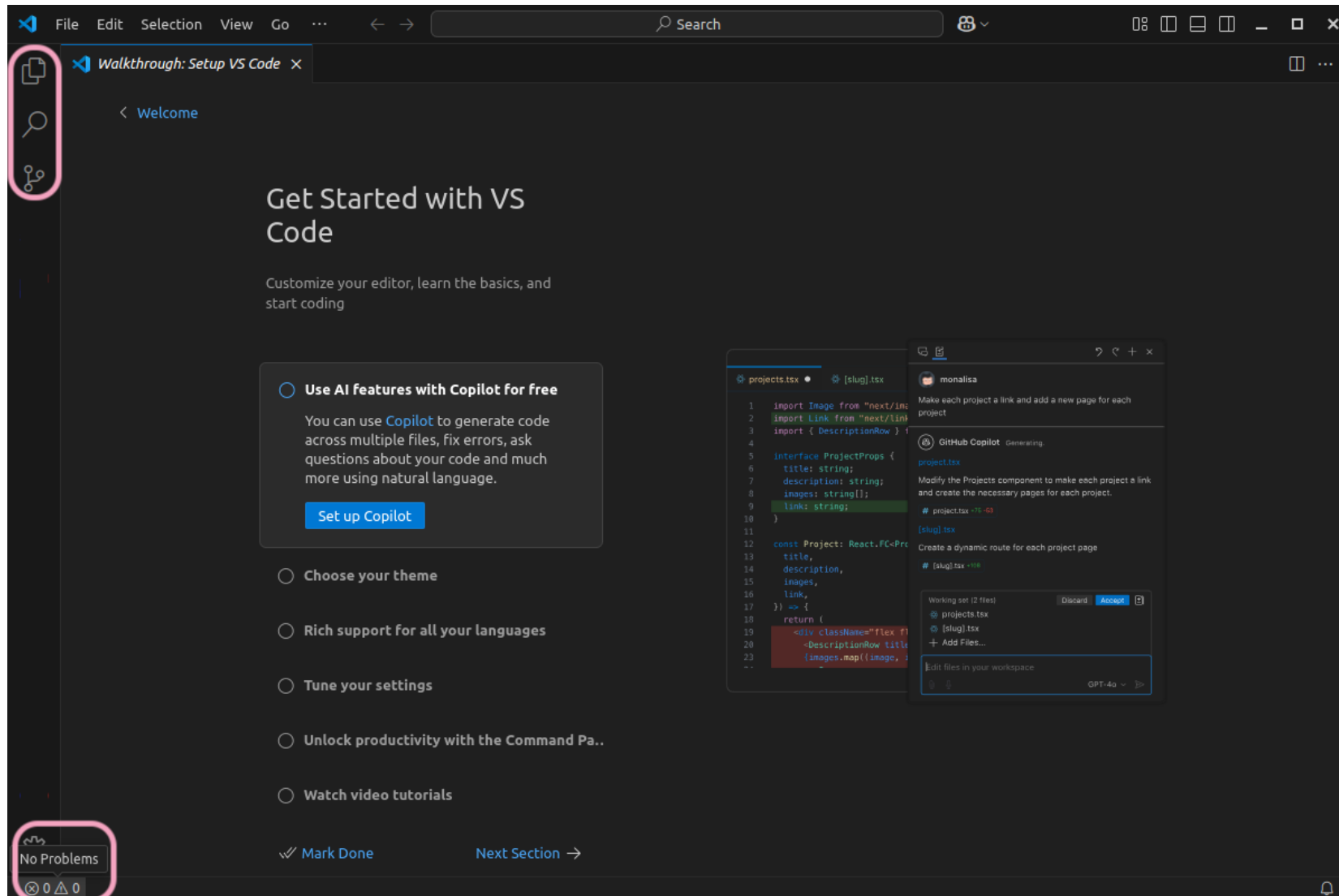
Exécuter VsCode pour la première fois

- Tour d'horizon des différentes zones de l'interface

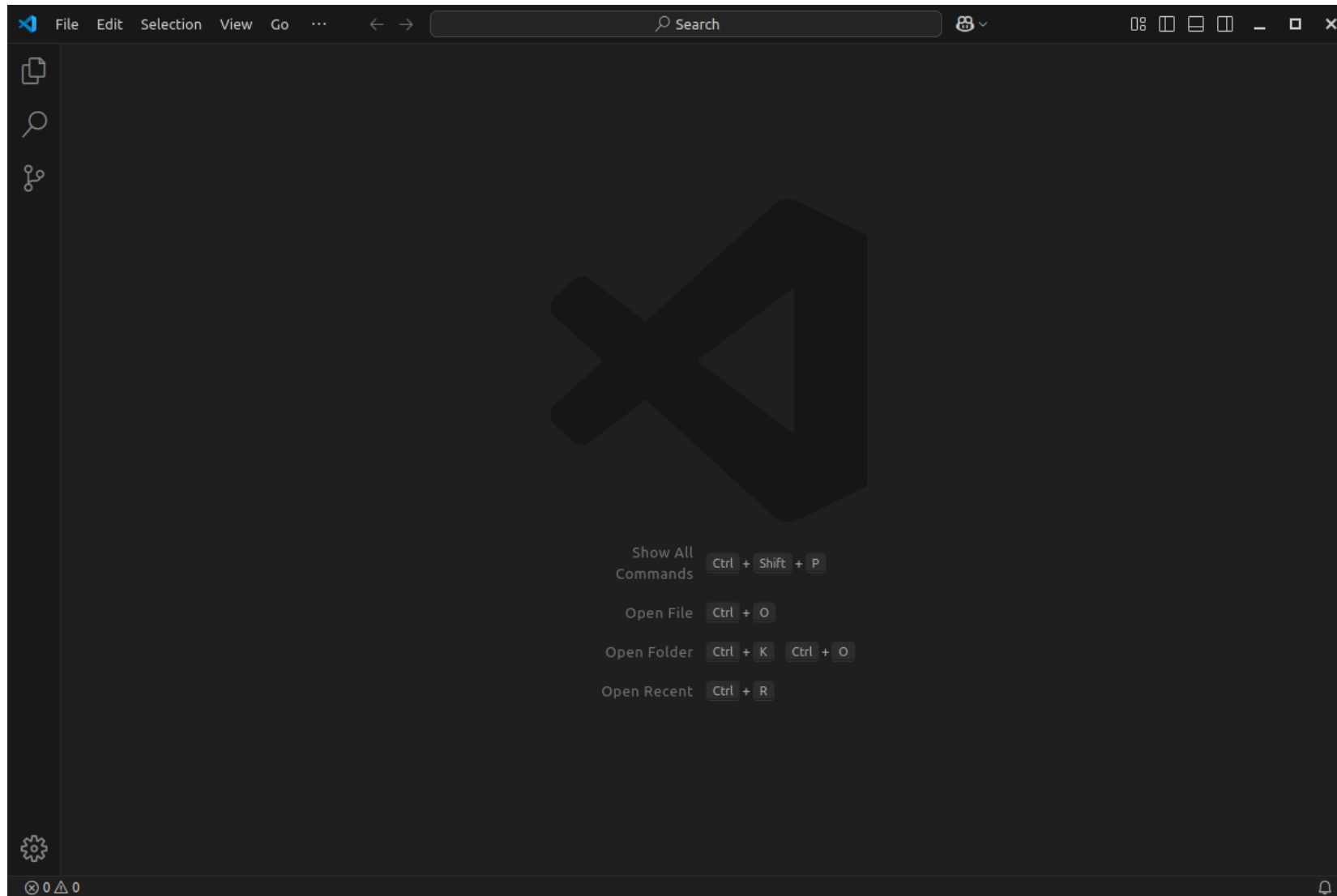


Simplifier le menu latéral

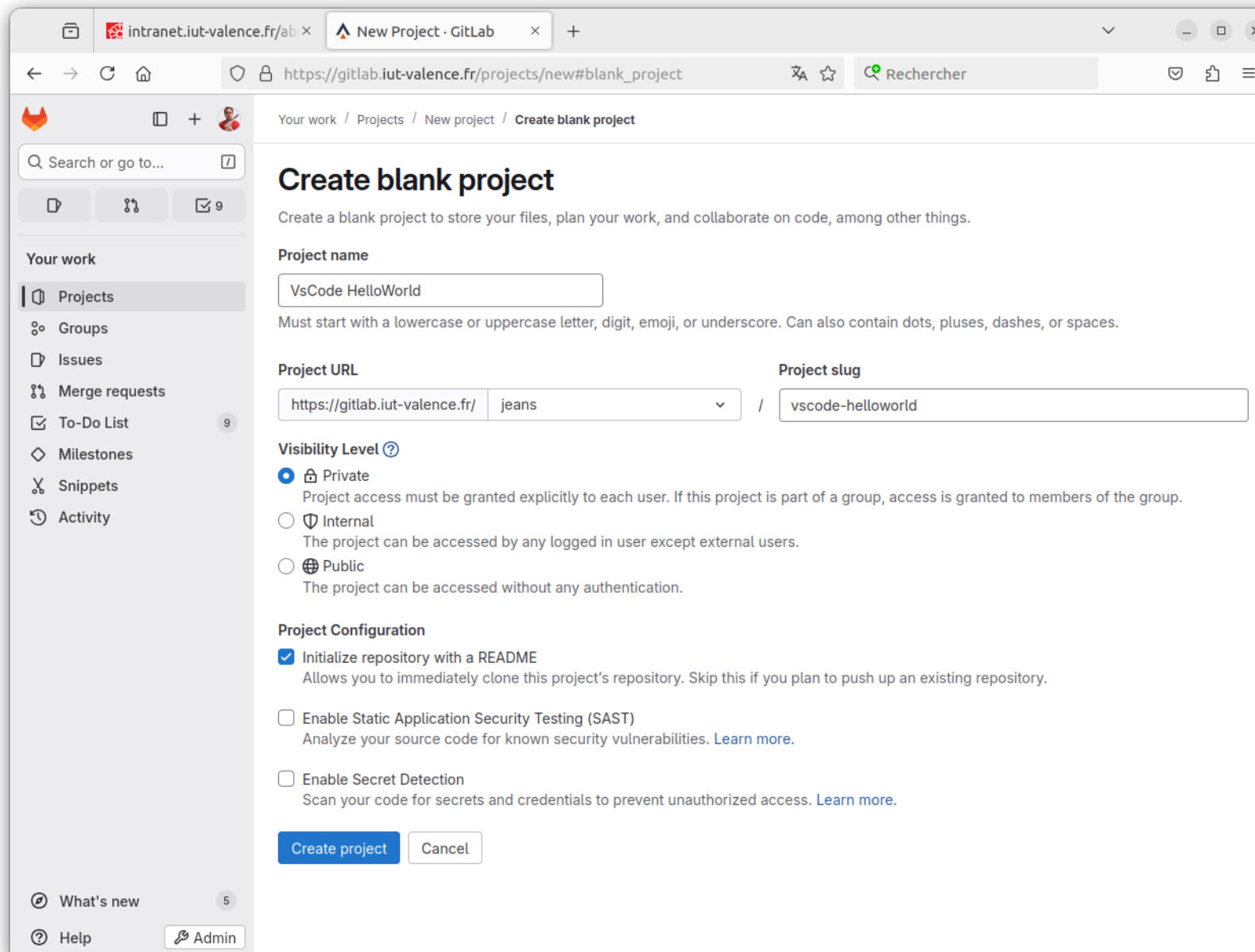
- Se débarrasser des fonctions *inutiles* pour le moment



Etre prêt à coder



Créer d'un dépôt gitlab



intranet.iut-valence.fr/ab x New Project - GitLab x +

https://gitlab.iut-valence.fr/projects/new#blank_project

Rechercher

Your work / Projects / New project / Create blank project

Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name

VsCode HelloWorld

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL **Project slug**

https://gitlab.iut-valence.fr/ jeans / vscode-helloworld

Visibility Level

☒ Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

☐ Internal
The project can be accessed by any logged in user except external users.

☐ Public
The project can be accessed without any authentication.

Project Configuration

☒ Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

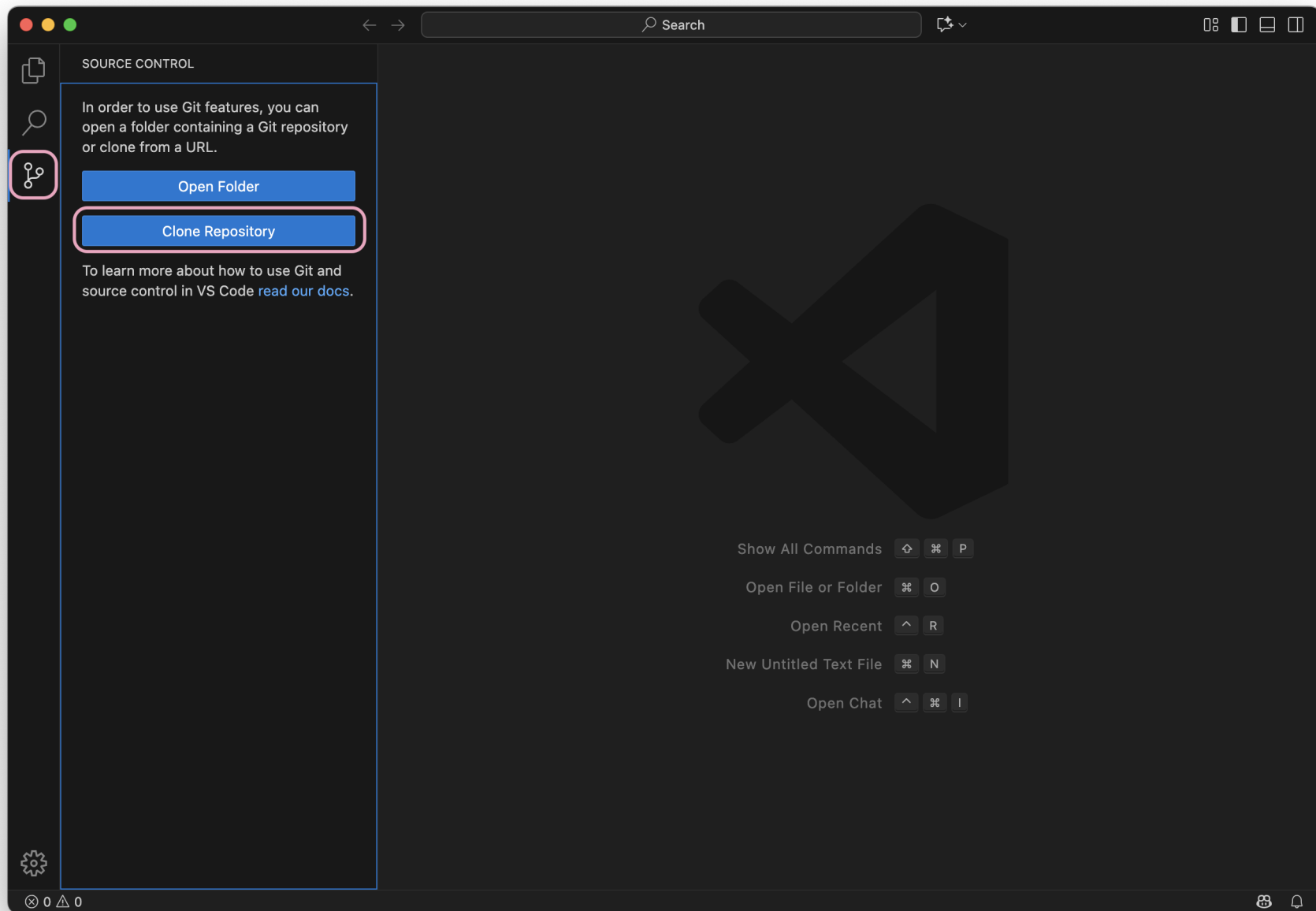
☐ Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. [Learn more.](#)

☐ Enable Secret Detection
Scan your code for secrets and credentials to prevent unauthorized access. [Learn more.](#)

Create project Cancel

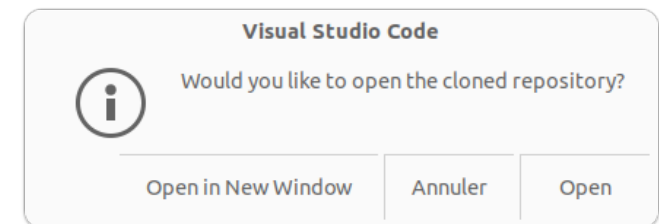
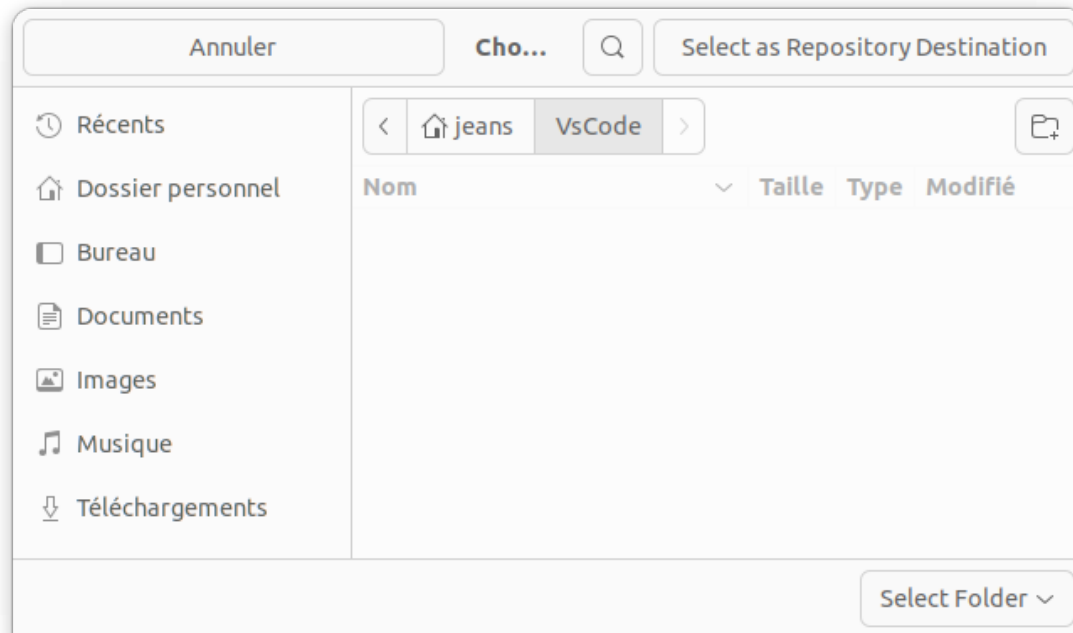


Cloner un dépôt *GitLab* depuis *VsCode*



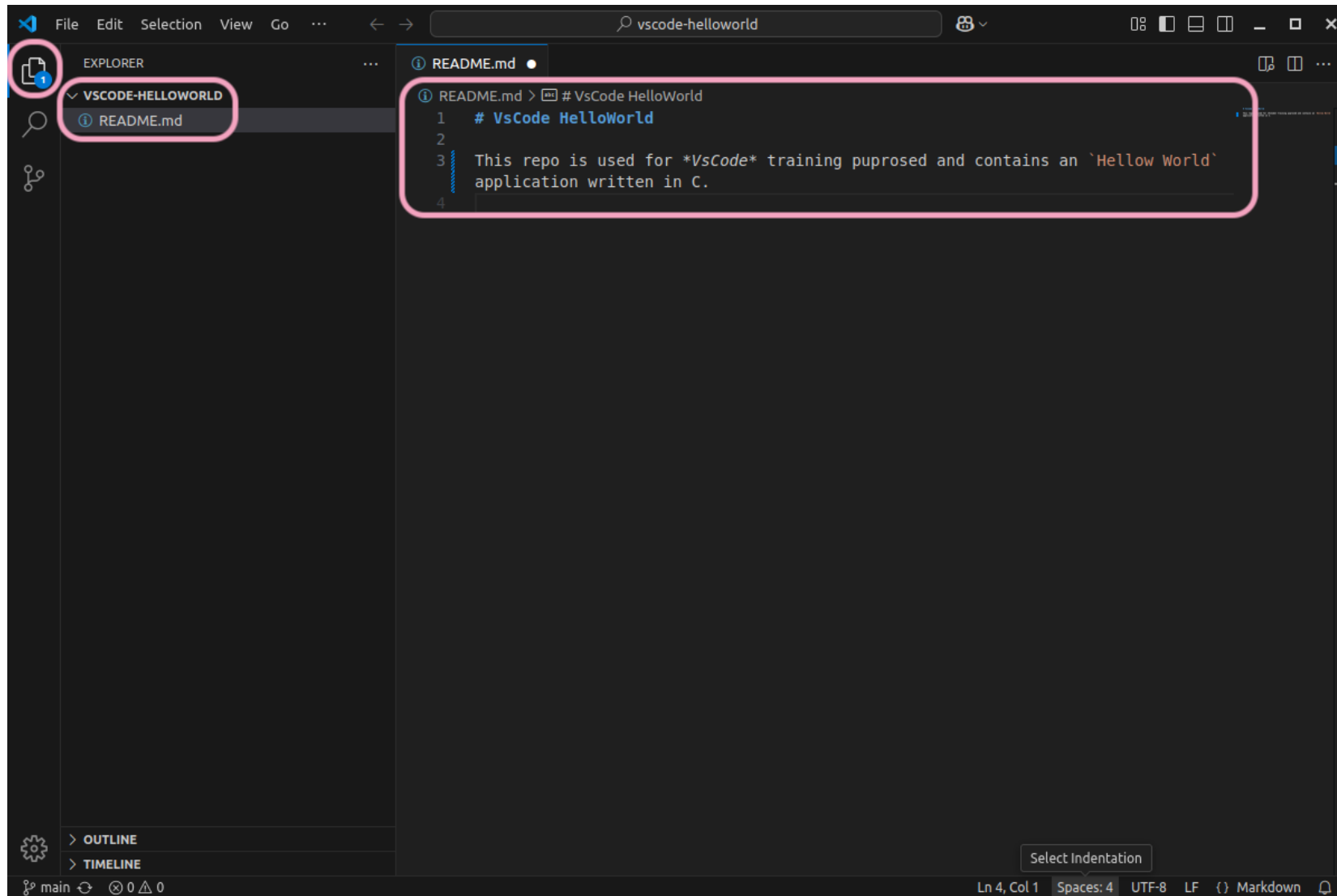
Cloner un dépôt *GitLab* depuis *VsCode* (fin)

- Créer un **répertoire d'accueil** pour le clone
- Ouvrir le dossier dans *VsCode*



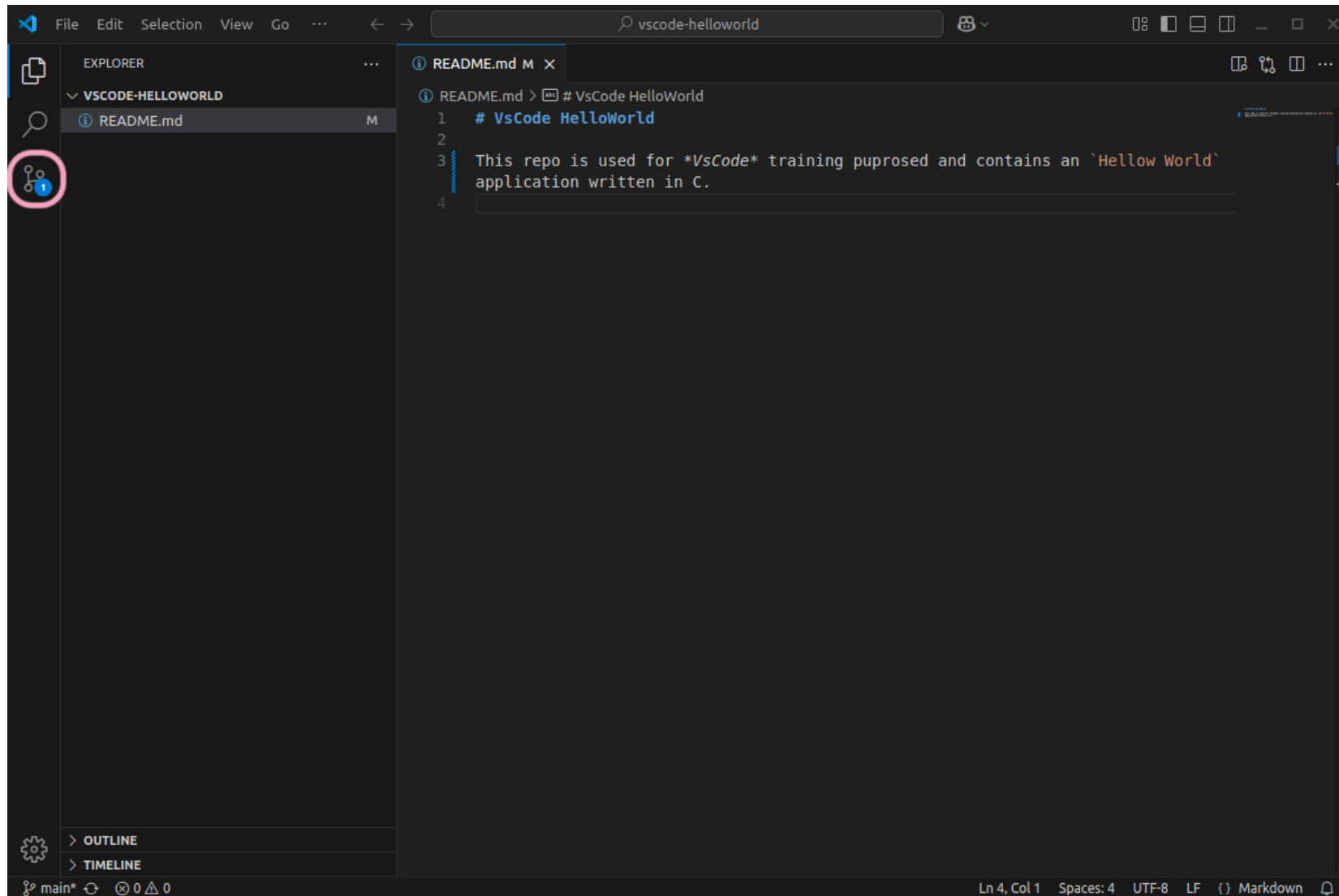
Editer (sans sauvegarder) le fichier README

- N.B. : le fichier n'est pas automatiquement sauvegardé



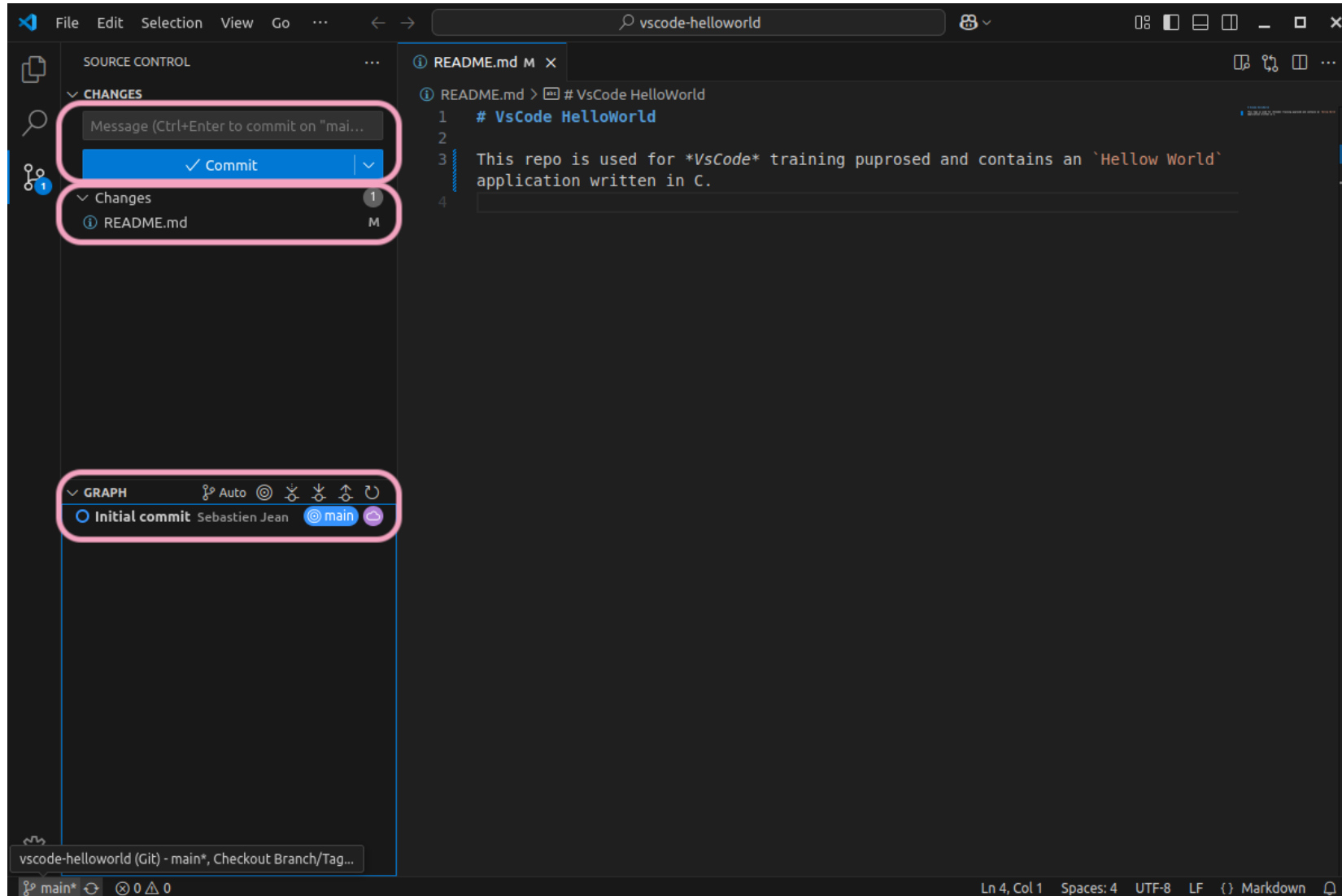
Sauvegarder le fichier README

- L'apparition du fichier est **observée par git**

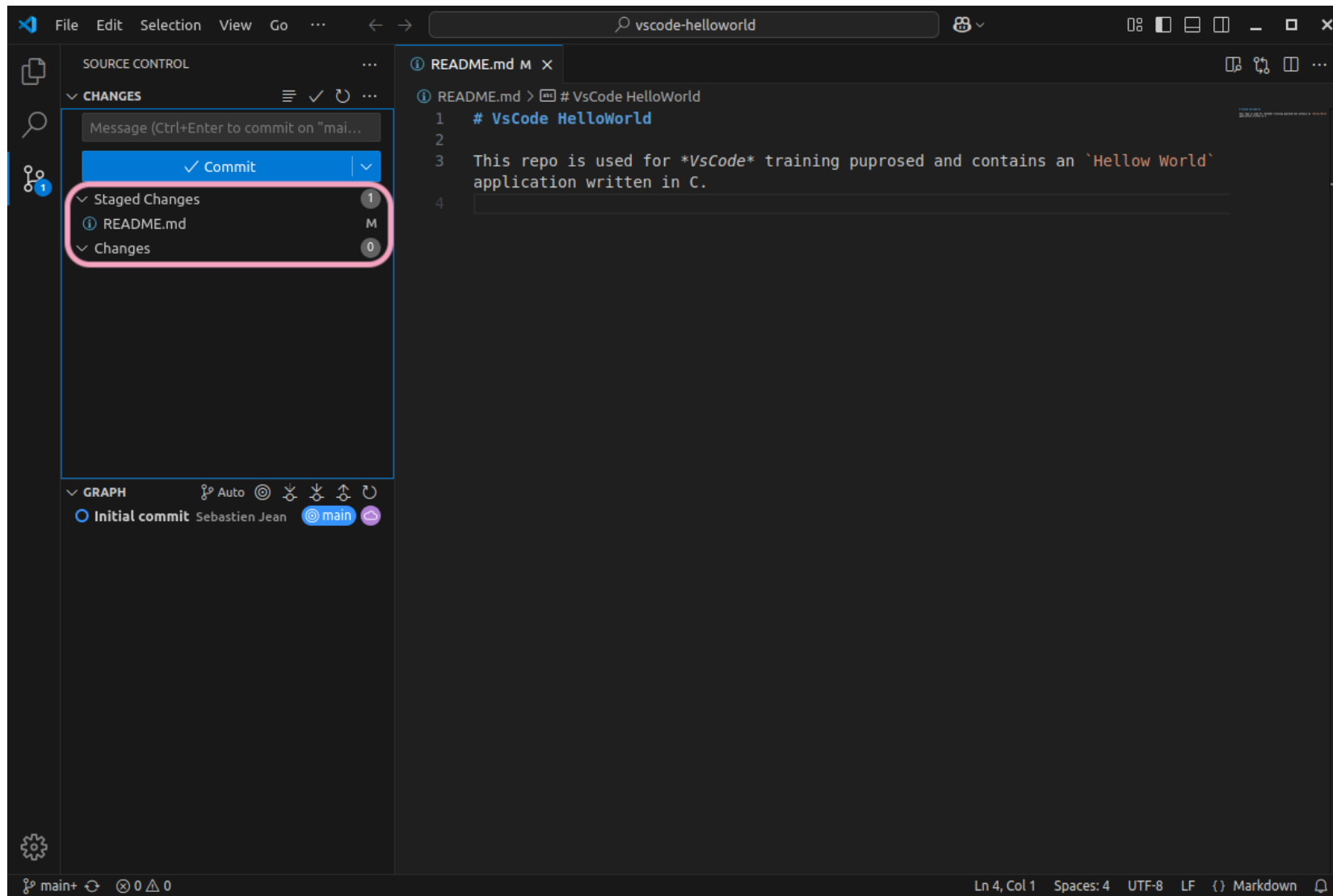


Vue "gestion de versions"

- Tour d'horizon (et liens avec les commandes en ligne)

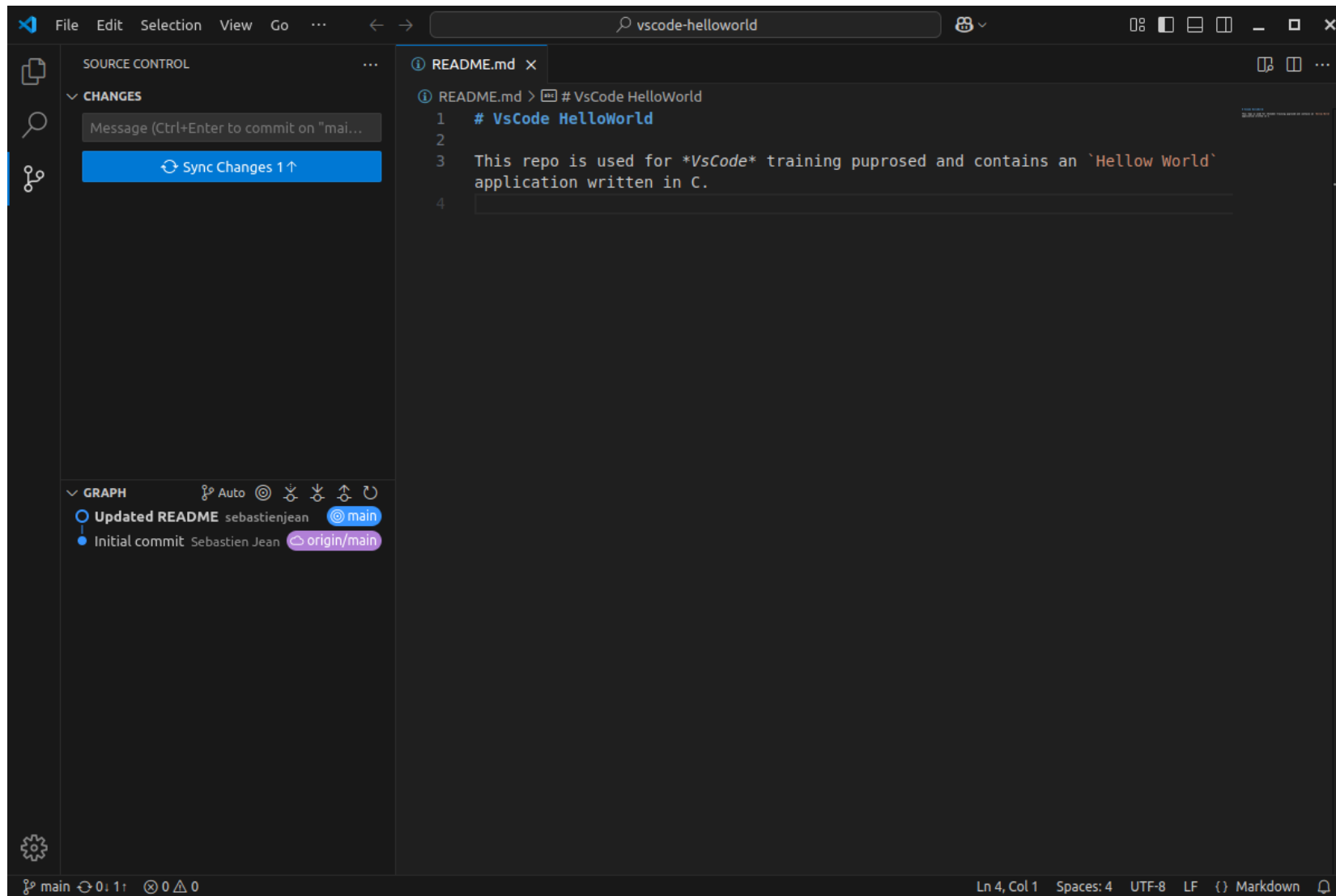


Préparer la prochaine version (*unstaged* → *staged*)



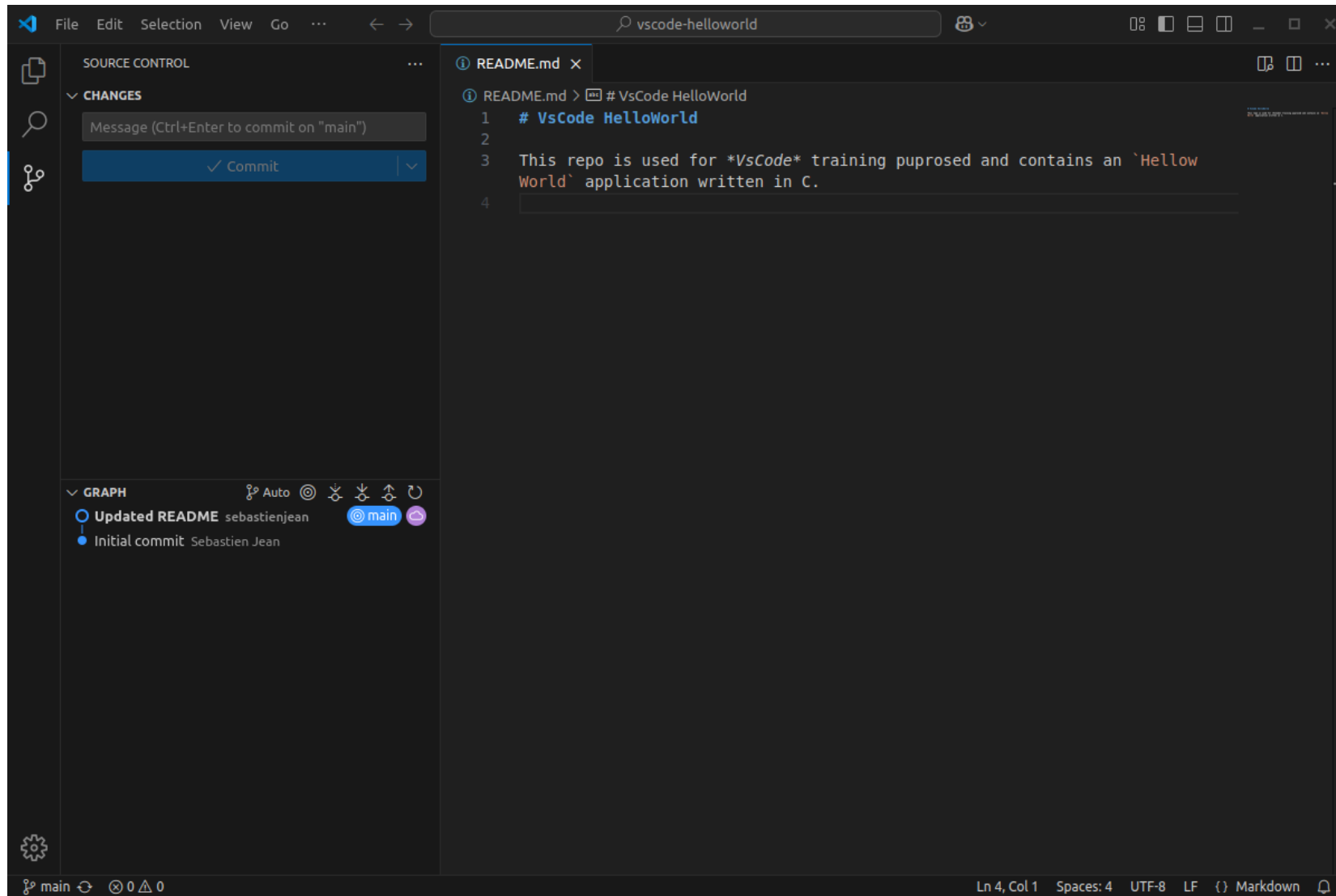
- Quelle est la commande *git* correspondante ?

Produire une nouvelle version (*commit*)



- Sans surprise le commit est toujours local

Pousser le *commit* sur le serveur



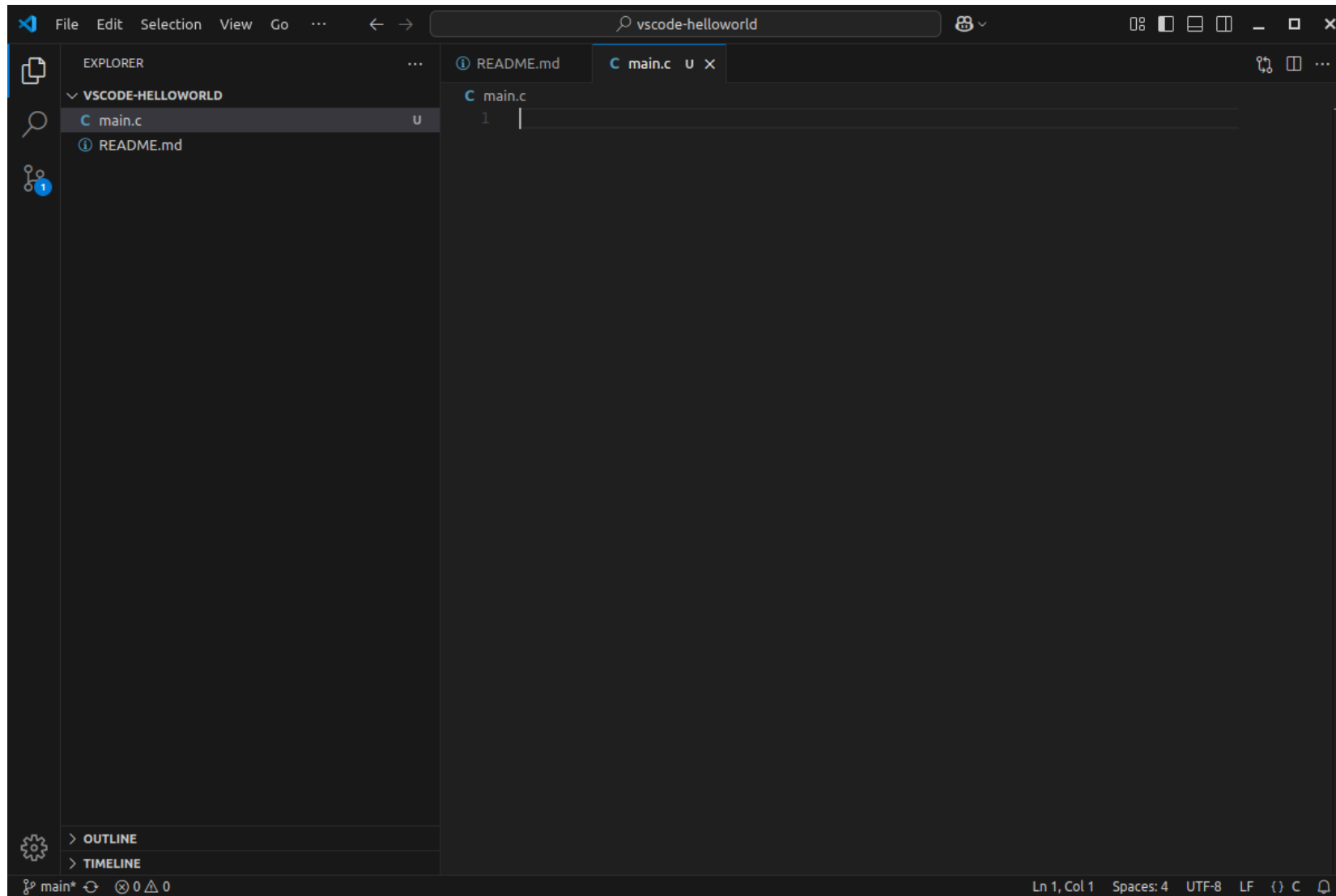
Pousser le *commit* sur le serveur (fin)

The screenshot shows a web browser window with the URL `https://gitlab.iut-valence.fr/jeans/vscode-helloworld`. The page title is "Sebastien Jean / VsCode HelloWorld". The repository name "VsCode HelloWorld" is prominently displayed. Below the name, there's a section for the latest commit: "Updated README" by "sebastienjean" 1 minute ago, with a commit hash "ab92c55a" and a "History" link. A table lists the files in the repository:

Name	Last commit	Last update
README.md	Updated README	1 minute ago

Below the table, the README content is shown, starting with "VsCode HelloWorld" and "This repo is used for VsCode training puprosed and contains an Hello World application written in C." On the right, "Project information" shows: 1 Commit, 1 Branch, 0 Tags, and 4 KiB Project Storage. A list of actions is provided: + Add LICENSE, + Add CHANGELOG, + Add CONTRIBUTING, + Enable Auto DevOps, + Add Kubernetes cluster, + Set up CI/CD, + Add Wiki, and + Configure Integrations. The bottom right indicates the repository was "Created on September 16, 2025". The left sidebar contains navigation links: Project, Pinned, Issues (0), Merge requests (0), Manage, Plan, Code, Build, Secure, Deploy, Operate, Monitor, Analyze, Settings, What's new (5), and Help (Admin).

Créer un fichier .c



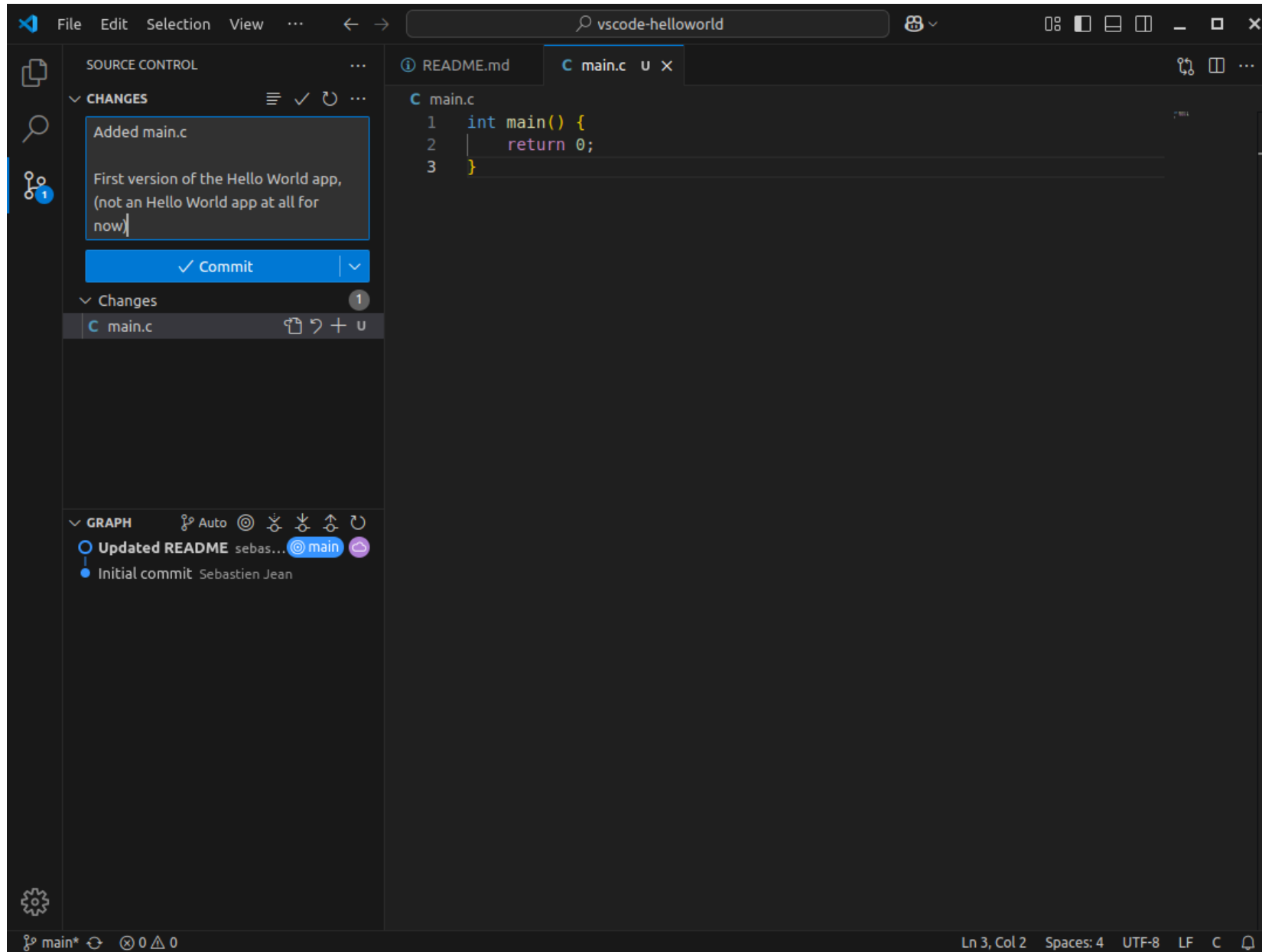
Hello world!, V0

Code C

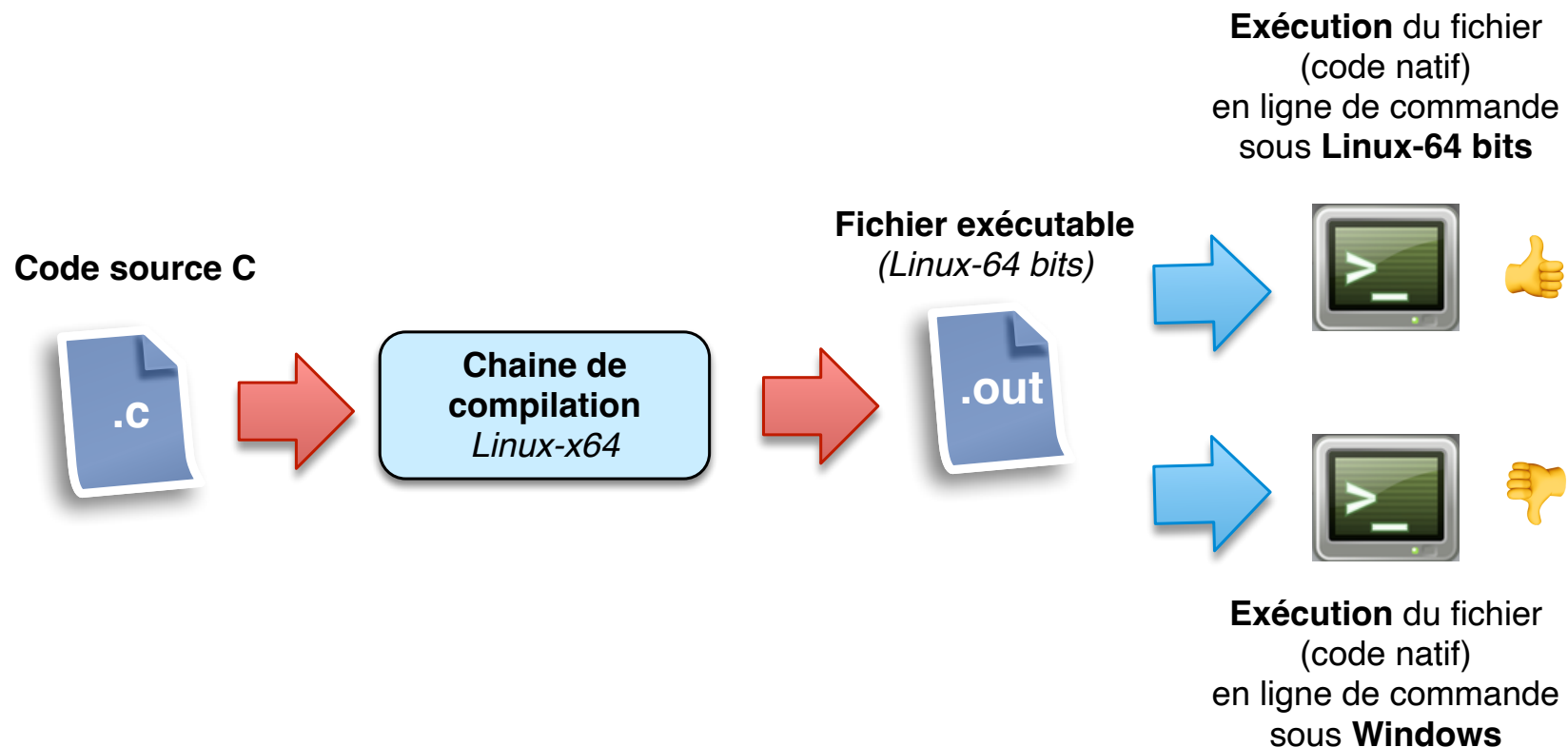
```
int main() {  
    return 0;  
}
```

- N.B. : Le programme est destiné à être exécuté depuis un terminal
- La **valeur retournée** par la fonction main est un **statut de terminaison**
 - $0 \rightarrow$ normal,
 - $\neq 0 \rightarrow$ anormal,

Hello world!, V0 : commit



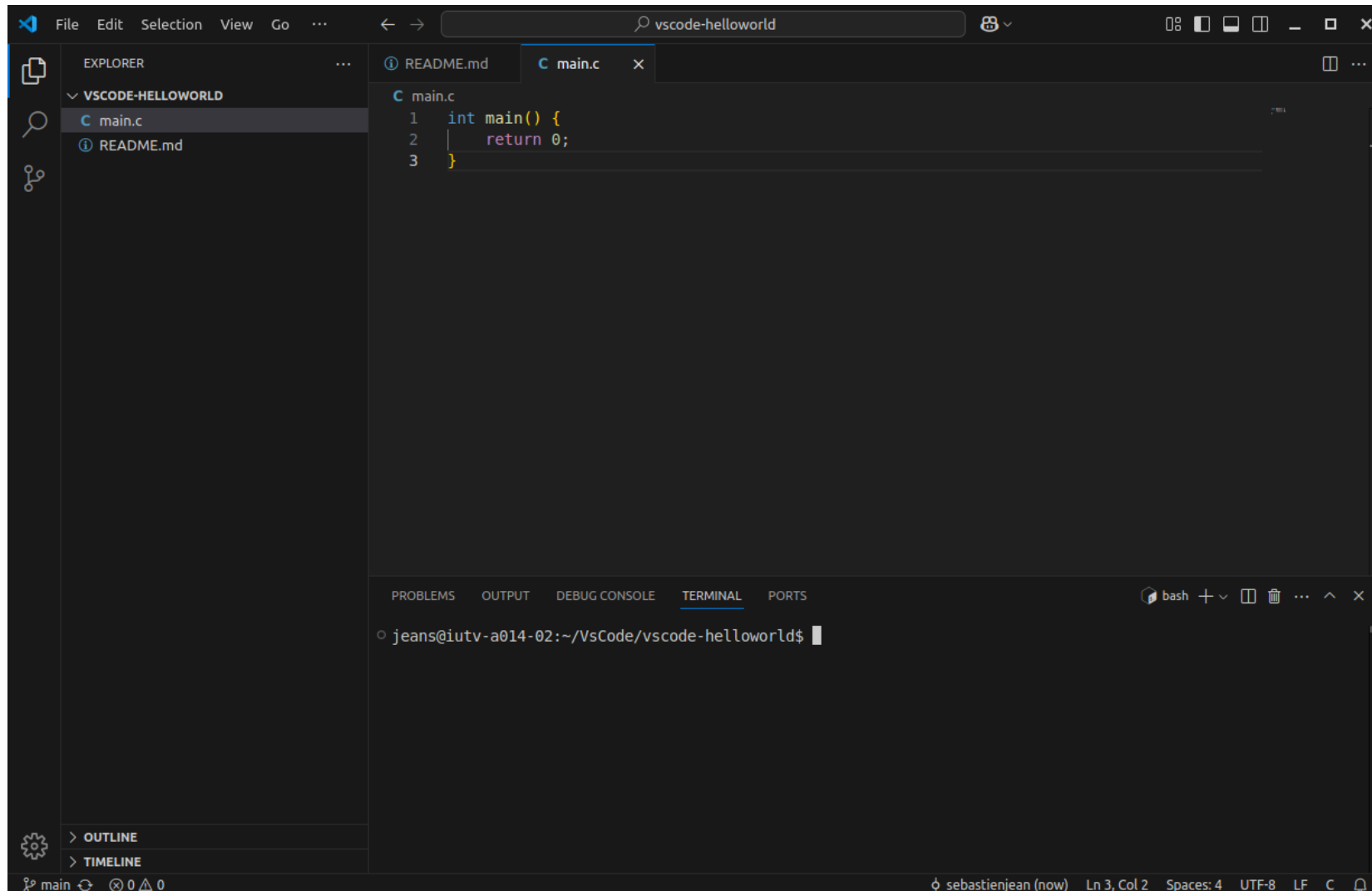
Compiler et exécuter un programme



- N.B. : version très simplifiée
 - La **chaîne de compilation** est une **suite d'outils** comprenant le compilateur mais pas que ...
 - Le processus de "compilation" comporte un certain nombre d'étapes intermédiaires

Vue *Terminal* de *VsCode*

- Terminal intégré (dans le répertoire ouvert)



Compiler et exécuter le programme



- Depuis le terminal, **compiler** le fichier `main.c` à l'aide de `gcc`
- **Observer** le contenu détaillé (`ls -al`) du répertoire
- **Caractériser** `a.out`
- **Exécuter** `a.out`

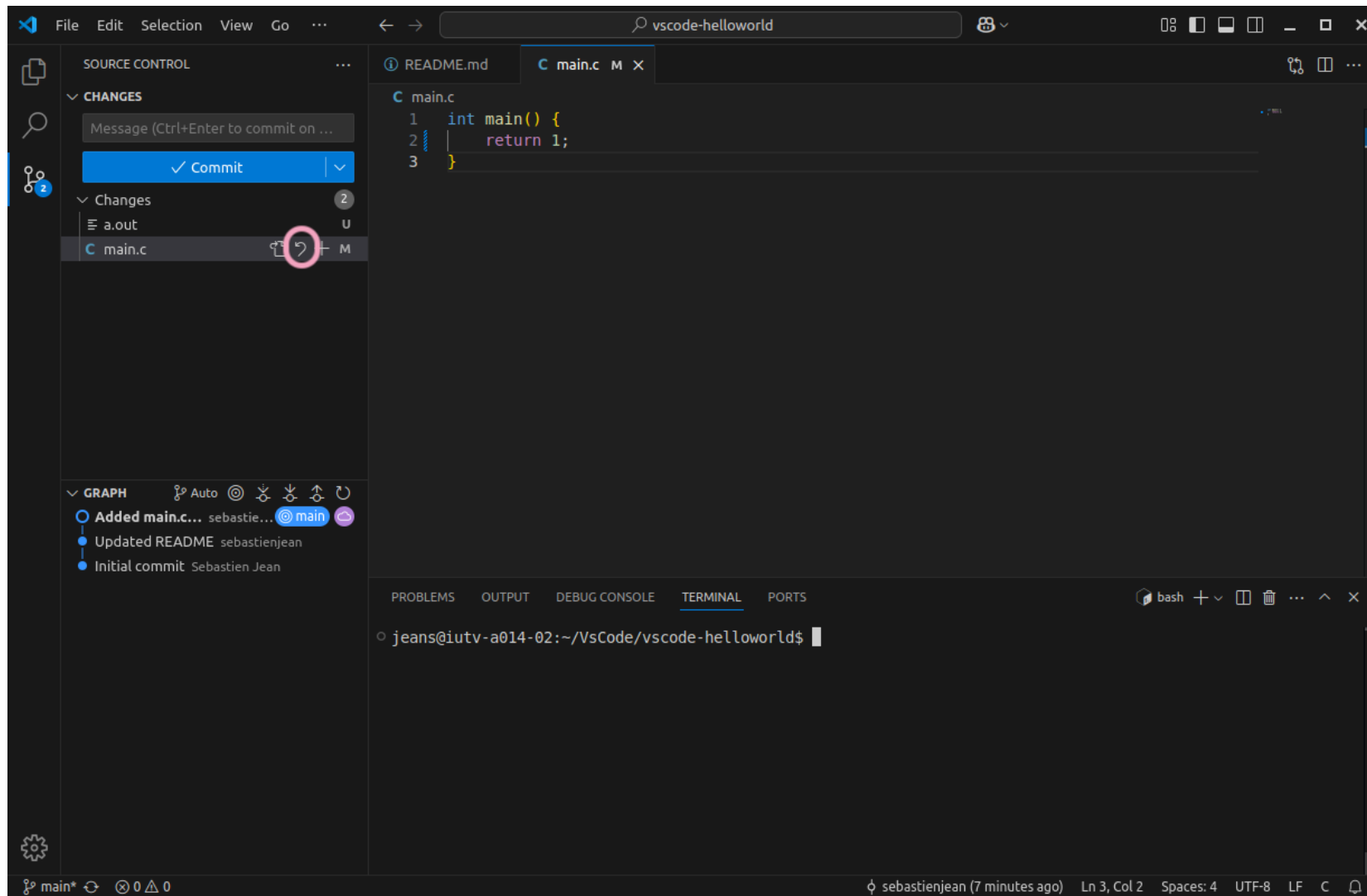
Compiler et exécuter le programme (fin)

- N.B. : dans le *shell Linux* (bash) la **variable d'environnement \$?** contient le statut de terminaison de la dernière commande exécutée



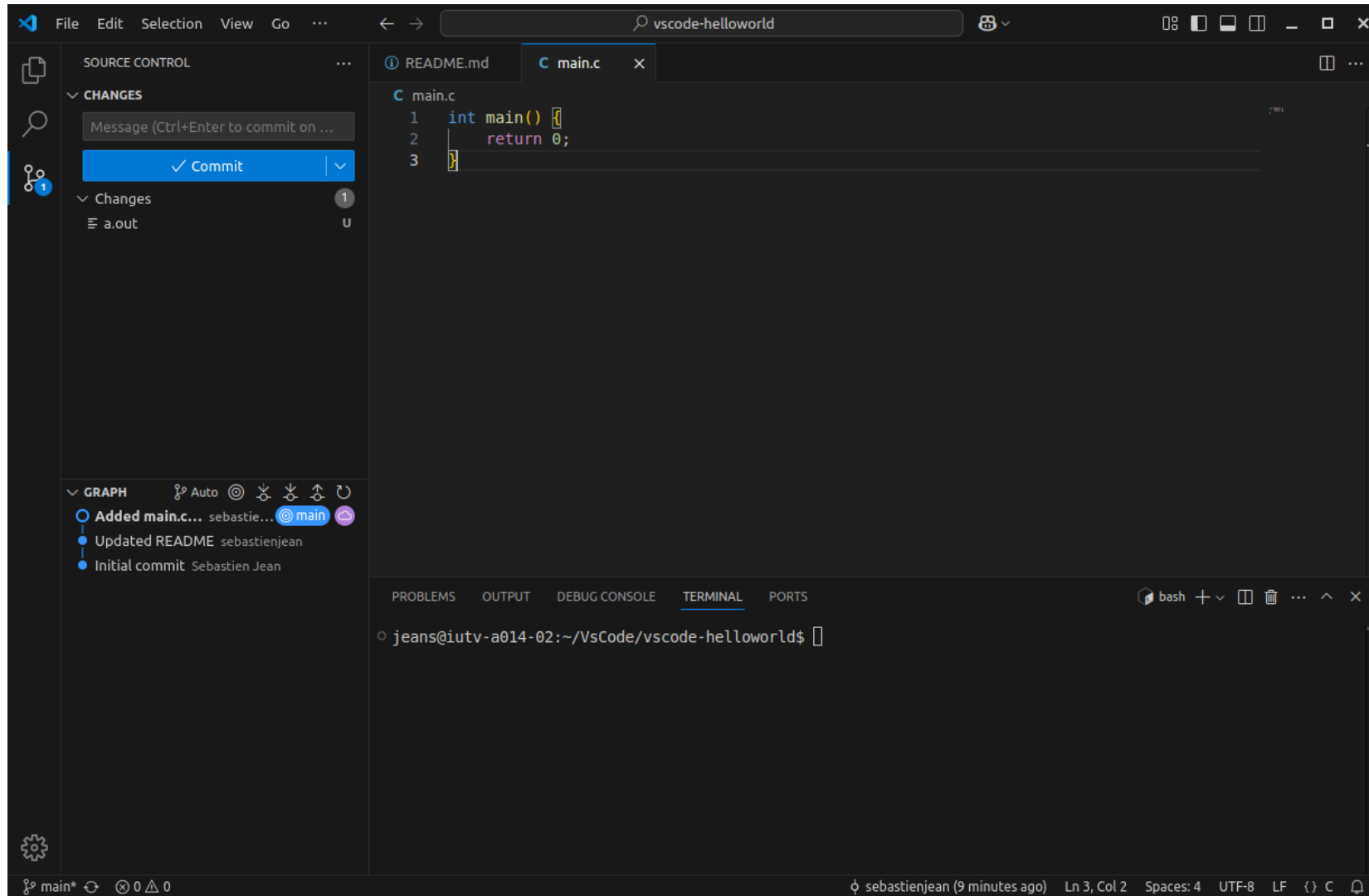
- **Afficher** le statut de terminaison de l'exécutable `a.out`
- **Modifier** le code source pour changer le statut de terminaison
- **Recompiler** `main.c`
- **Afficher** le statut de terminaison de l'exécutable `a.out`

Annuler les modifications du fichier



- Quelle est la commande *git* correspondante ?

Annulation les modifications du fichier (fin)



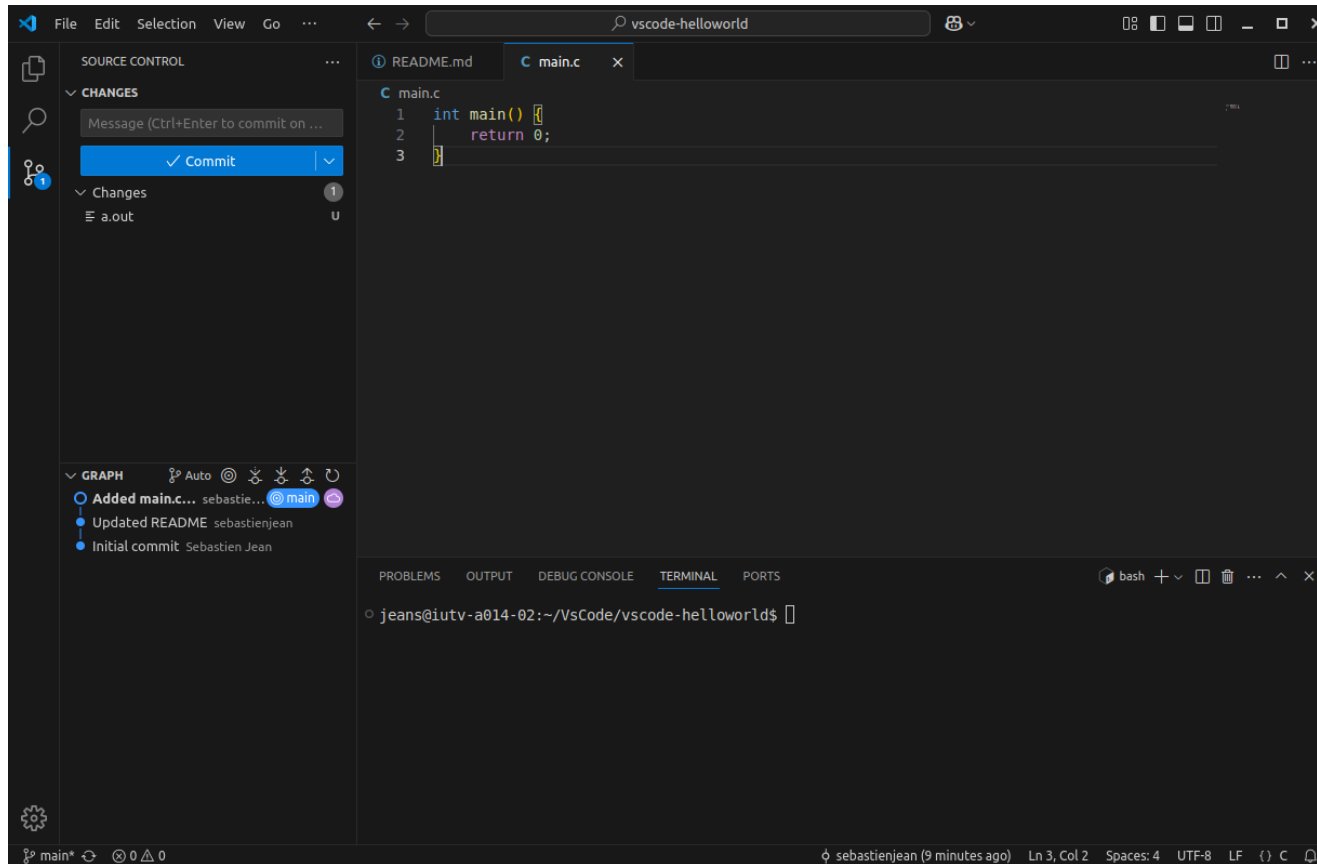
Ignorer le suivi d'un fichier

- N.B. : On peut ajouter dans n'importe quel répertoire du dépôt un fichier `.gitignore` qui spécifie les **ressources (fichiers, répertoires) à ignorer** dans ce répertoire et en dessous (il est possible d'utiliser des *jokers* pour spécifier des ensembles de ressources)



- En utilisant les menus contextuels, **Ignorer** le fichier a.out
- **Observer** l'état des changements
- Question subsidiaire : **pourquoi ignorer ce fichier ?**

Ignorer le suivi d'un fichier (fin)



- **Effectuer un commit** du fichier `.gitignore`

Hello world!, V1

Code C, V1

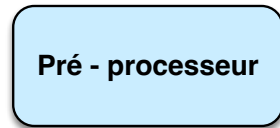
```
#include <stdio.h>

int main() {
    printf("Hello World!\n");
    return 0;
}
```

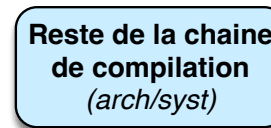
- N.B. : la fonction `printf` est une **fonction prédéfinie de la librairie standard C** (*libC*)
 - Elle **affiche** sur la **sortie standard** la chaîne de caractères passée en paramètres (sans passer à la ligne)
 - Le caractère spécial `\n` représente un **saut de ligne**

Compiler et exécuter un programme (suite)

Code source C



Code source C
(intégrant les définitions)



Fichier exécutable
(arch/syst)



Header C
(définition de
fonctions, ...)

- N.B. : version toujours très simplifiée

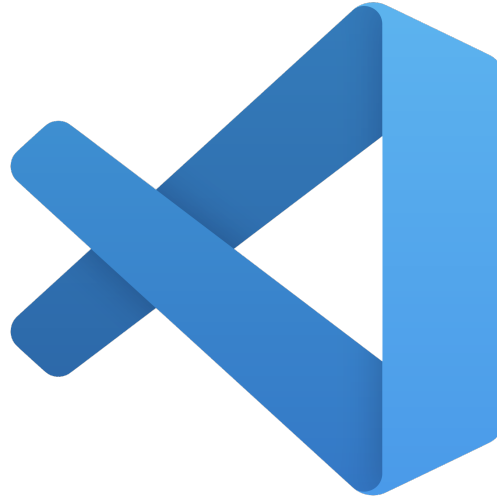
- Un fichier .c faisant référence à une fonction externe doit **inclure le fichier de définition .h** contenant sa signature
- Le **pré-processeur** résout notamment les **inclusions** de fichiers .h
- A la sortie du pré-processeur, le fichier .c résulte de la **fusion** du fichier .c original et des fichiers .h inclus

Compiler et exécuter le programme



- Depuis le terminal, **compiler** le fichier `main.c` à l'aide de **gcc**
 - Utiliser l'option **-o** pour que l'exécutable soit nommé `main` et non plus `a.out`
- **Exécuter** `a.out`, **observer** la sortie sur le terminal
- **Ignorer** le fichier `main`, **effectuer un commit** et **pousser le commit** sur le serveur

Compléments sur l'utilisation de *VsCode*



- Palette de commandes
- Chercher/remplacer
- Multi-curseurs
- . . .