



Comprendre GIT ?



Xavier ROBERT (xavier.robert@ird.fr) ; 27/02/2024

```
$ git init
Initialized empty Git repository in /tmp/tmp.IMBYSY7R8Y/.git/
$ cat > README << 'EOF'
> Git is a distributed revision control system.
> EOF
$ git add README
$ git commit
[master (root-commit) e4dcc69] You can edit locally and push
to any remote.
 1 file changed, 1 insertion(+)
 create mode 100644 README
$ git remote add origin git@github.com:cdown/thats.git
$ git push -u origin master
```

Qu'est-ce GIT ?

<https://git-scm.com/>



- logiciel de gestion de versions :
 - Conserver l'*historique des modifications*
 - ***Synchronisation*** dossier local / serveur
 - Coordination, ***travail collaboratif***
- Créé en 2005 (Linus Torvalds)
- **Modèle distribué** : projet hébergé sur un serveur distant, chaque utilisateur télécharge et héberge l'intégralité du projet sur sa propre machine
- Autres logiciels : CVS, SVN, Mercurial, Bazaar,...

Pourquoi utiliser un système de gestion des versions ?

- Retour à d'anciennes versions possible !
- Travail sur plusieurs versions simultanément
- Travail en équipe en même temps sur plusieurs versions
- Fusion des modifications
- Sécurité

Pourquoi utiliser GIT ?

- Simple d'utilisation
- Flexible
- Interface graphique possible
- Rapide
- Sécurisé
- Travail hors ligne possible, pas de nécessité de serveur distant
- Open source !

Git : notions principales

Dépôt / **Repository**

- répertoire caché **.git**
- contient toutes les données dont GIT a besoin pour gérer l'historique
- Pas de modifications manuelles de ce dossier
- Modifications avec les commandes GIT

```
10:57:49 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$ la
.                .git                CITATION.cff                MANIFEST.in                dist
..              .gitattributes        LICENCE.txt                README.rst                setup.py
.DS_Store       .gitignore            LICENSE                    To_Do.txt                simple_swath
10:57:52 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$ la .git
.                COMMIT_EDITMSG  HEAD                config                hooks                info                objects
..              FETCH_HEAD          branches            description            index                logs                refs
10:58:40 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$
```

Git : notions principales

Commit

- **Séquence de « snapshots/instantané »** (= commits) concernant l'état de tous les fichiers du projet
- Possède :
 - 1 date
 - 1 Auteur
 - 1 description textuelle
 - 1 lien vers le(s) commit(s) précédent(s)
- Stocké dans une base de données locale

Git : notions principales

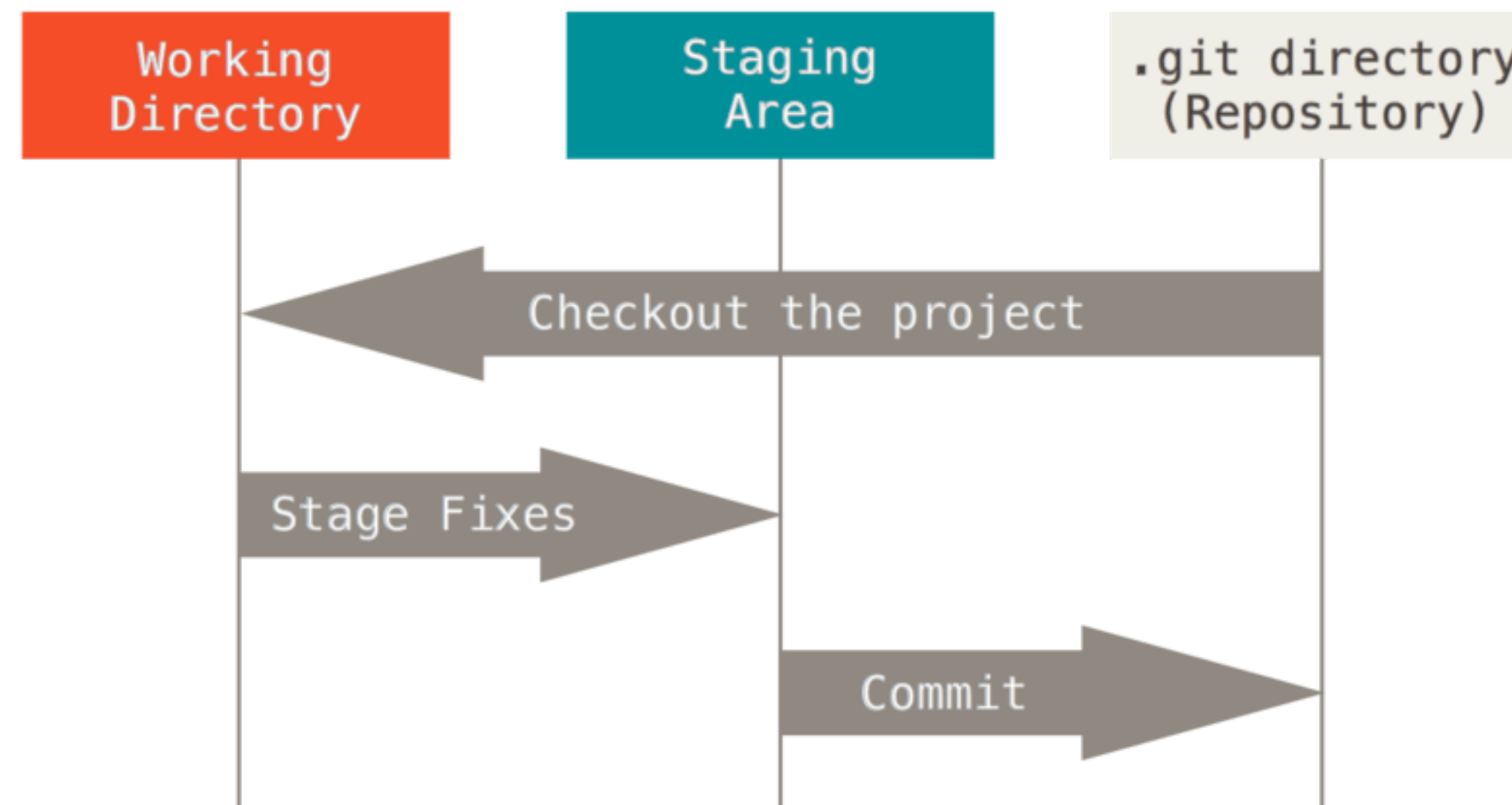
Commit

- Ne stocke que les fichiers modifiés par rapport au commit précédent
 - —> Efficace
- l'ensemble peut être compressé pour réduire la redondance (\$ *git gc*)
- Etat « *committed* » = validé = stocké dans la base de données de l'instantané

Git : notions principales

Copie de travail / **Working copy**

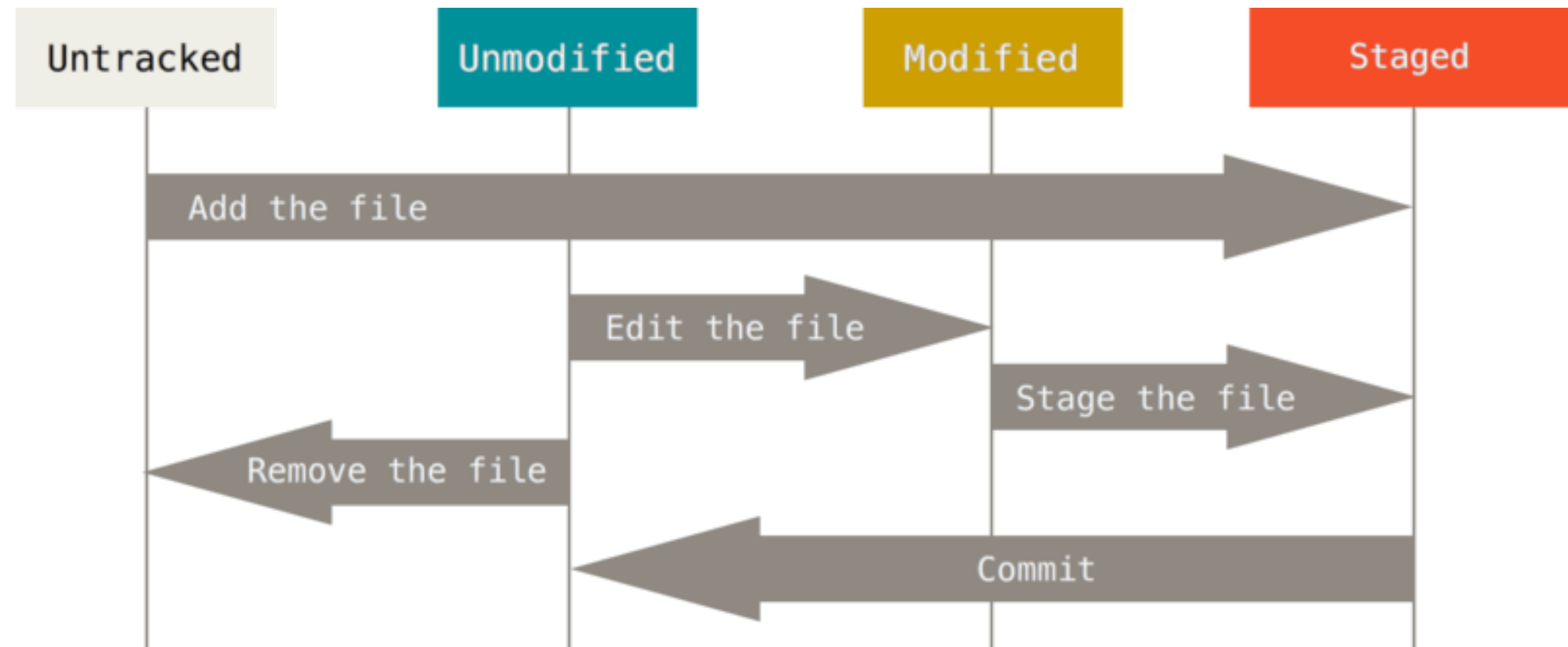
- Fichiers effectivement présents dans le répertoire géré par GIT
- Etat peut-être différent du dernier commit de l'historique
- Correspond à une extraction unique (*checkout*) d'une version du projet



Git : notions principales

Index

- Espace temporaire concernant les modification prêtes à être ***commitées***
- Modifications =
 - création de fichier(s)
 - modification de fichier(s)
 - suppression de fichier(s)



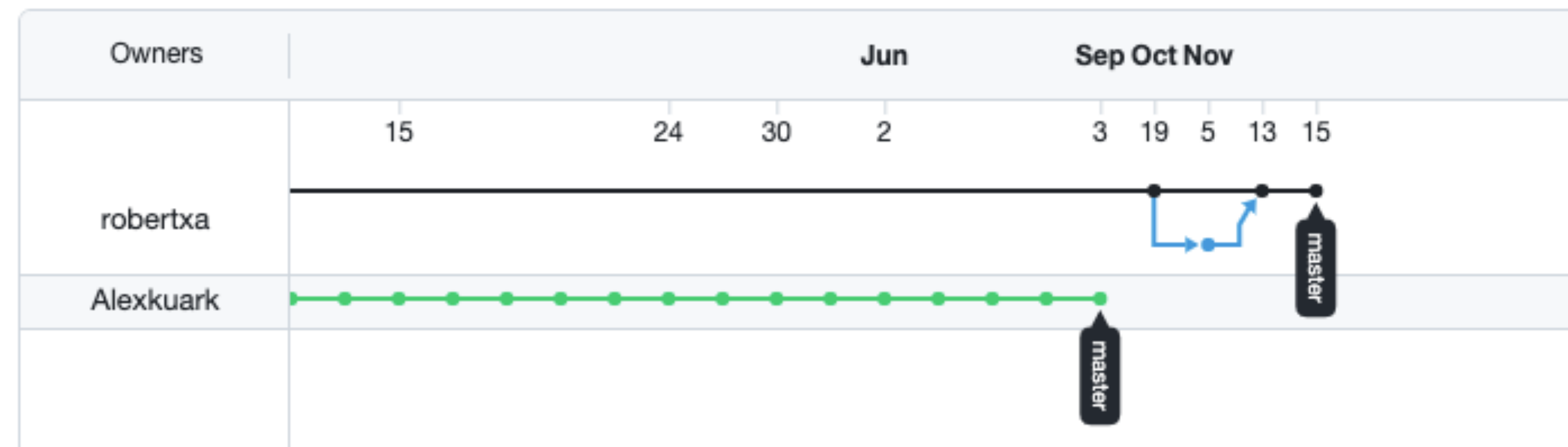
Git : notions principales

Branches

- Plusieurs versions divergentes du projet
- N'impacte pas le projet de base
- Convergence/fusion possible (Merge)
- Branche par défaut = **master**

Network graph

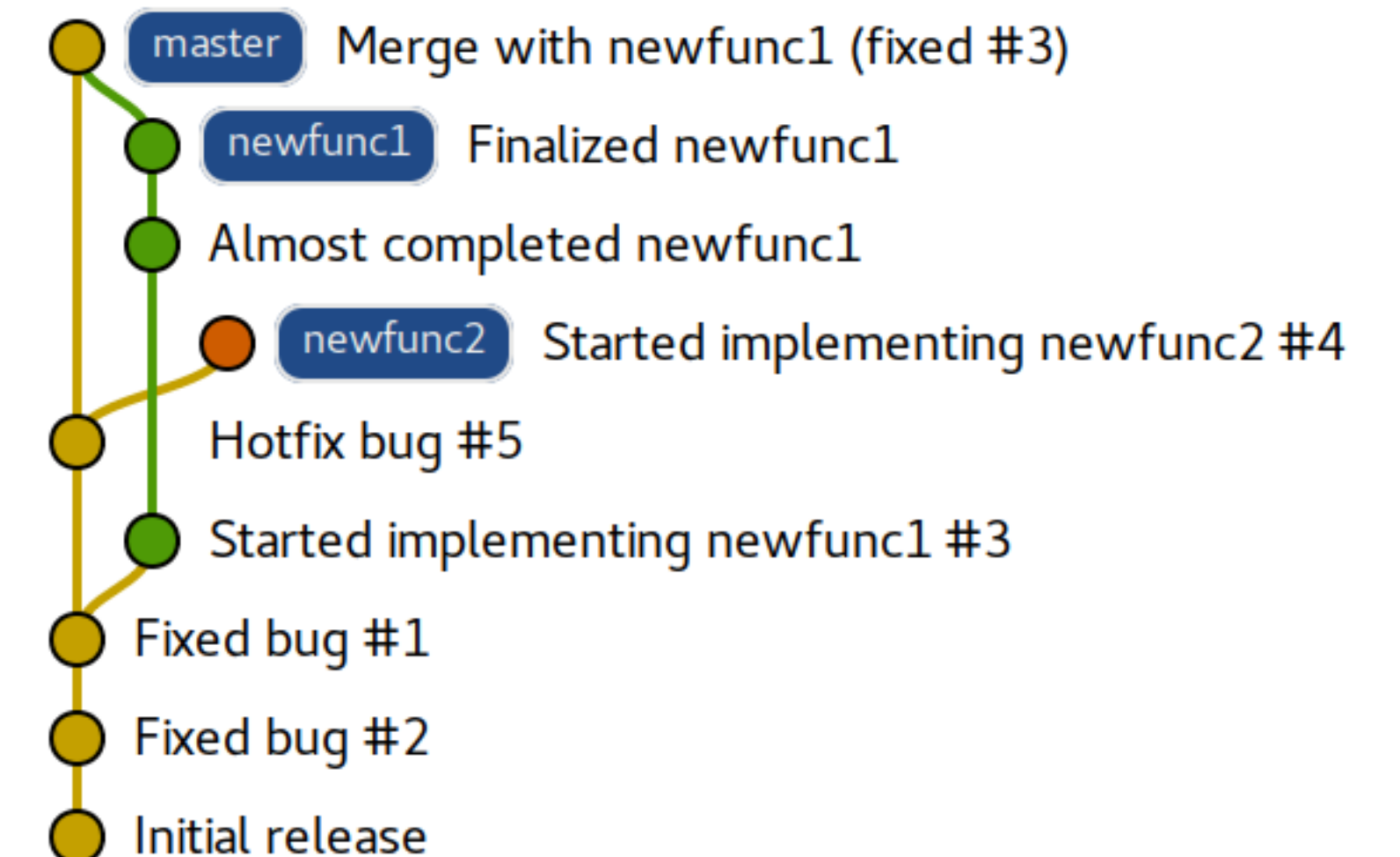
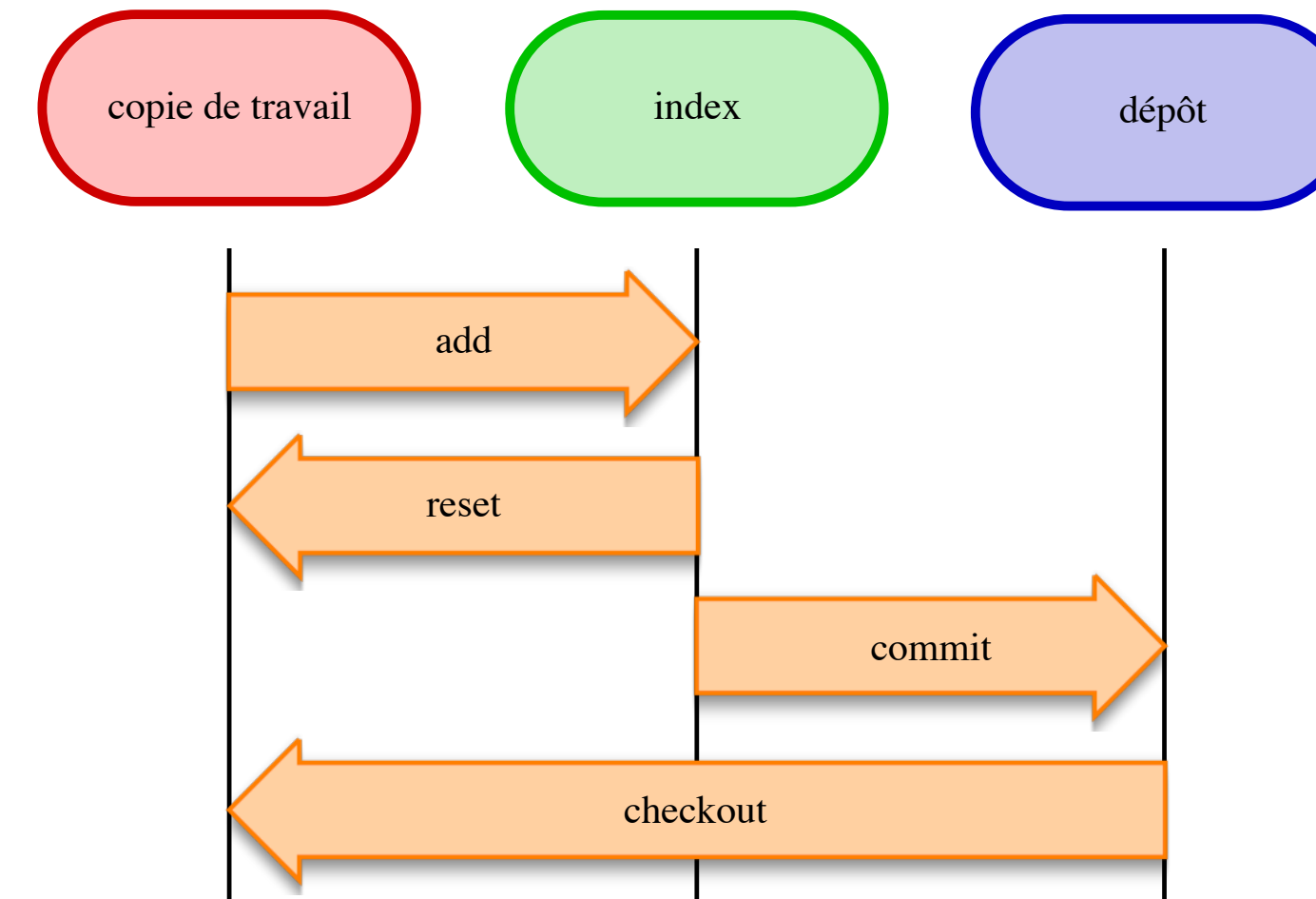
Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



Git : Mise en oeuvre

Workflow

1. Création d'un dépôt
2. Modifier le dépôt
3. 1er commit
4. Créer une nouvelle branche
5. Commiter les modifications
6. Proposer une pull-request
7. Faire un Merge
8. Synchronisation



Git : Mise en oeuvre

Comment ?

- Soit en **lignes de commandes** (<https://git-scm.com/>)
 - Git doit être installé sur votre ordinateur
- Soit avec une **interface graphique** (<https://git-scm.com/downloads/guis>)
 - git-gui,
 - gitk,
 - GitHub Desktop,
 - ...

Configuration des outils

- Définir le nom à associer à toutes les opérations :

\$ git config --global user.name "Xavier"

- Définir l'email :

\$ git config --global user.email 'xavier.Robert@ird.fr'

Création d'un dépôt

- ***Initialise la gestion de version dans un répertoire***
- Création du sous-répertoire .git
- Pour créer un nouveau dépôt:
 - Aller dans le répertoire (*\$ cd mon_repertoire*)
 - *\$ git init* # initialise un dépôt vide —> Faut indexer les fichiers
- Pour utiliser un dépôt Git existant sur le cloud :
 - *\$ git clone <url>*

Committer

- Rappel : **Message informatif du commit !**
 - Doit expliquer la modification/apport du commit
- En ligne de commandes :
 - Ajouter un fichier dans l'index : `$ git add <filename>` **PENSER à le faire systématiquement**
 - Supprimer un fichier de l'index : `$ git reset <filename>`
 - Suppression d'un fichier : `$ git rm <filename>`
 - Voir l'état des modifications indexées : `$ git status` (résumé) ou `$ git diff`
 - Commiter : `$ git commit` ou `$ git commit -m <mon_message>`

Consulter un historique

- Afficher la liste des commits : `$ git log`
- Afficher le détail d'un commit spécifique : `$ git show <id_commit>`

```
[pierres-macbook-pro:projet-git pierre$ git log -p
commit 4e4193fa9576cee497ff1862b6a3f38f6ae1111b (HEAD -> master)
Author: Pierre Giraud <pierre.giraud@edhec.com>
Date:   Sat Oct 26 08:55:57 2019 +0200

    Sauvergarde fichier2

diff --git a/fichier2.txt b/fichier2.txt
new file mode 100644
index 0000000..27e7cf9
--- /dev/null
+++ b/fichier2.txt
@@ -0,0 +1 @@
+Ajout de texte

commit c7a4dd2e66608a062cc8afe53adbfd9380f81184
Author: Pierre Giraud <pierre.giraud@edhec.com>
Date:   Thu Oct 24 09:48:43 2019 +0200

    Abandon du suivi de fichier2.txt

diff --git a/fichier2.txt b/fichier2.txt
deleted file mode 100644
index e69de29..0000000
```


Ecraser et remplacer un commit

- Pour annuler une validation (commit) et la remplacer par une autre
- `$ git commit --amend`

Annuler des modifications apportées à 1 fichier

- Pour revenir à un état antérieur

\$ git checkout --<filename>

ou

\$ git restore <filename>

Empêcher indexation de fichiers spécifiques

- Créer un fichier **.gitignore** avec les noms de fichiers/dossiers à ne pas indexer

```
[11:20:19 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$ la
.                .git                CITATION.cff          MANIFEST.in           dist
..              .gitattributes      LICENCE.txt           README.rst            setup.py
.DS_Store       .gitignore          LICENSE              To_Do.txt             simple_swath
[12:03:27 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$ head .gitignore
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
12:03:43 [robertxa1@eduroam-085002.grenet.fr ~/Documents/CODES-backup/My_tools/Simple_Swath]$
```

Créer une branche

- Crée un nouveau pointeur vers le dernier commit effectué :

\$ git <nom_nouvelle_branche>

- Basculer dans la nouvelle branche :

\$ git checkout <nom_nouvelle_branche>

- Le prochain *\$ git commit* affectera la nouvelle branche

Intégration des différentes branches dans Master

Fusion

- Se positionner sur le master :

\$ git checkout master

- Faire la fusion :

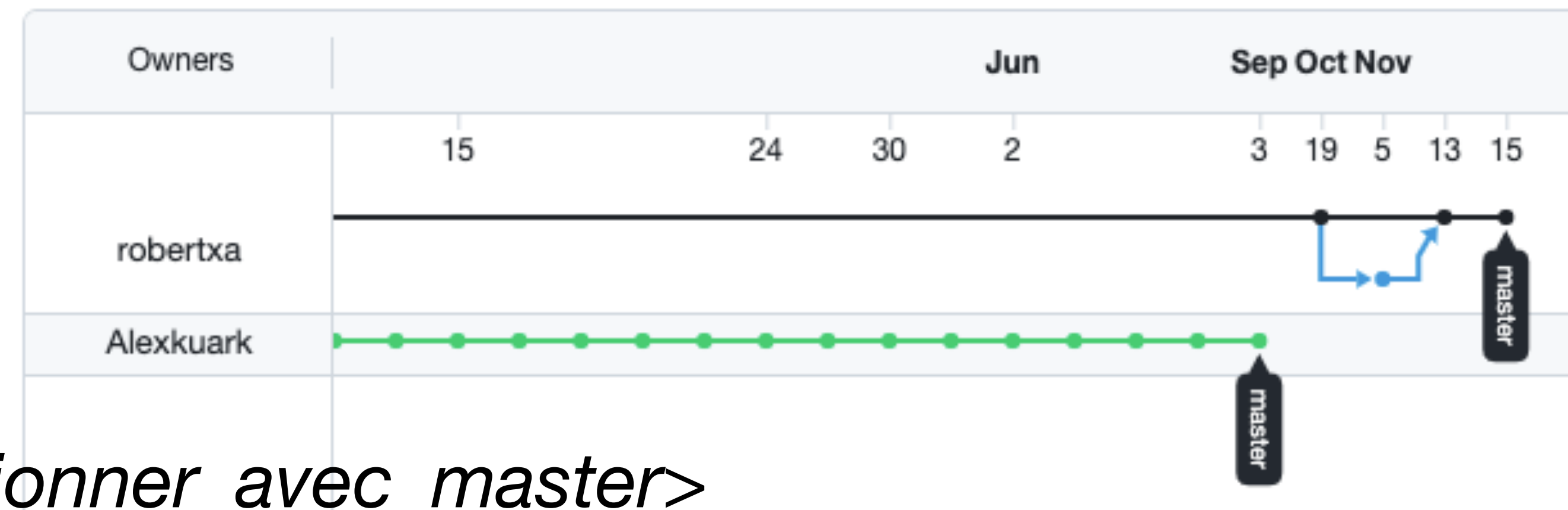
\$ git merge <nom_branche_a_fusionner_avec_master>

- Effacer branche adjacente

\$ git branch -d

Network graph

Timeline of the most recent commits to this repository and its network ordered by most recent



Intégration des différentes branches dans Master

Rebaser

- Git part du dernier commit commun
- Récupère les modifications sur la branche à rapatrier
- Les applique sur la branche de base

\$ git rebase

- Même résultat que fusion, mais historique plus clair
- **ATTENTION** : ne jamais rebaser des commits poussés sur un dépôt public !!!

Synchroniser avec une plateforme distante

- Ajout URL distant à configuration

\$ git remote add <nom_distant> <url_repository>

- Pousser modification locale vers plateforme distante

\$ git commit

puis

\$ git push <nom_distant> <nom_local>

- Récupérer en local dernières modifications :

\$ git pull <nom_distant> <nom_local>

**Vous savez tout sur git
(ou presque !)**

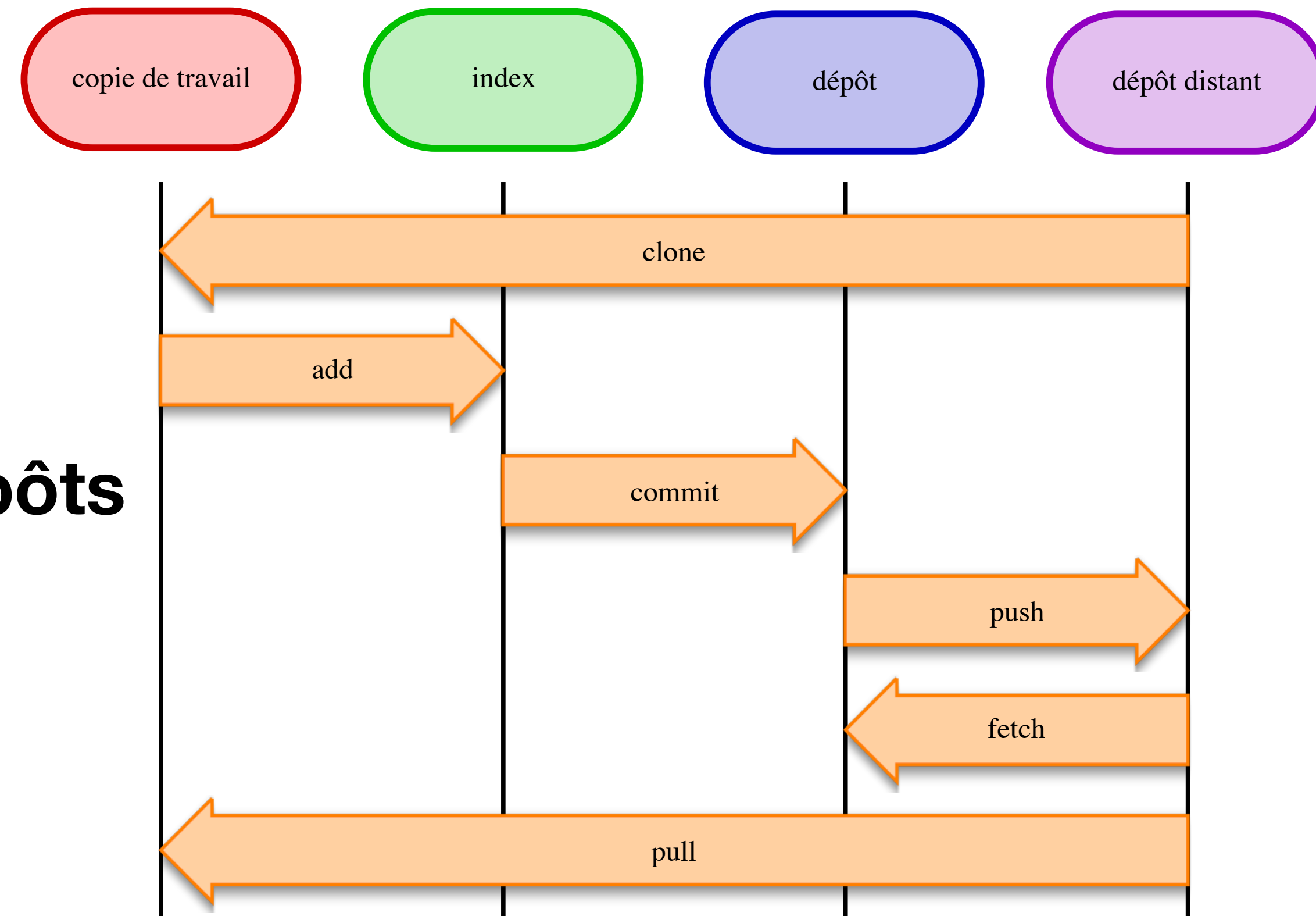
Qu'est-ce que GitHub

<https://github.com/>

- **Service en ligne d'hébergement des dépôts**
- Le plus grand hébergeur du monde
- Une partie publique
- Racheté par Microsoft



GitHub



GitHub

- Éléments importants :
- Readme.rst
- Release
- DOI
- Licence
- Citation
- ...

robertxa / Th-Config-Xav

Type / to search

>_

+ ▾

🕒

🔗

📧

👤

<> Code

🕒 Issues

🔗 Pull requests

🎬 Actions

📁 Projects

📖 Wiki

🛡 Security

📊 Insights

⚙ Settings

👤 Th-Config-Xav

Public

📌 Pin

👁 Watch 2 ▾

🔗 Fork 0 ▾

★ Star 1 ▾

🔗 master ▾

🔗 1 Branch

🏷 1 Tags

🔍 Go to file

t

Add file ▾

<> Code ▾

About ⚙

👤 robertxa

Update config.thc

dd74ec2 · 10 months ago

🕒 39 Commits

📁 TEST_THERION	update Config.th2	5 years ago
📁 Template_cave	Add vertical scalebar	4 years ago
📄 .gitignore	Add a Test cases folder	5 years ago
📄 History.txt	Update transparency	3 years ago
📄 README.rst	Add points by Juraj Halama	4 years ago
📄 ToDo.txt	Update transparency	3 years ago
📄 config.thc	Update config.thc	10 months ago

📖 README

✎

☰

Therion Config file

Version 3.1 - Apr. 27, 2020

Created by: / Créé par : Xavier Robert

Licence

Released under a Creative Commons Attribution-ShareAlike-NonCommercial License (except indications in thconfig.thc):

Therion Config file to simplify *.thconfig files and associated (and commented) template files

📖 Readme

📈 Activity

★ 1 star

👁 2 watching

🔗 0 forks

Releases 1

📦 Zenodo DOI

Latest

on May 10, 2021

Packages

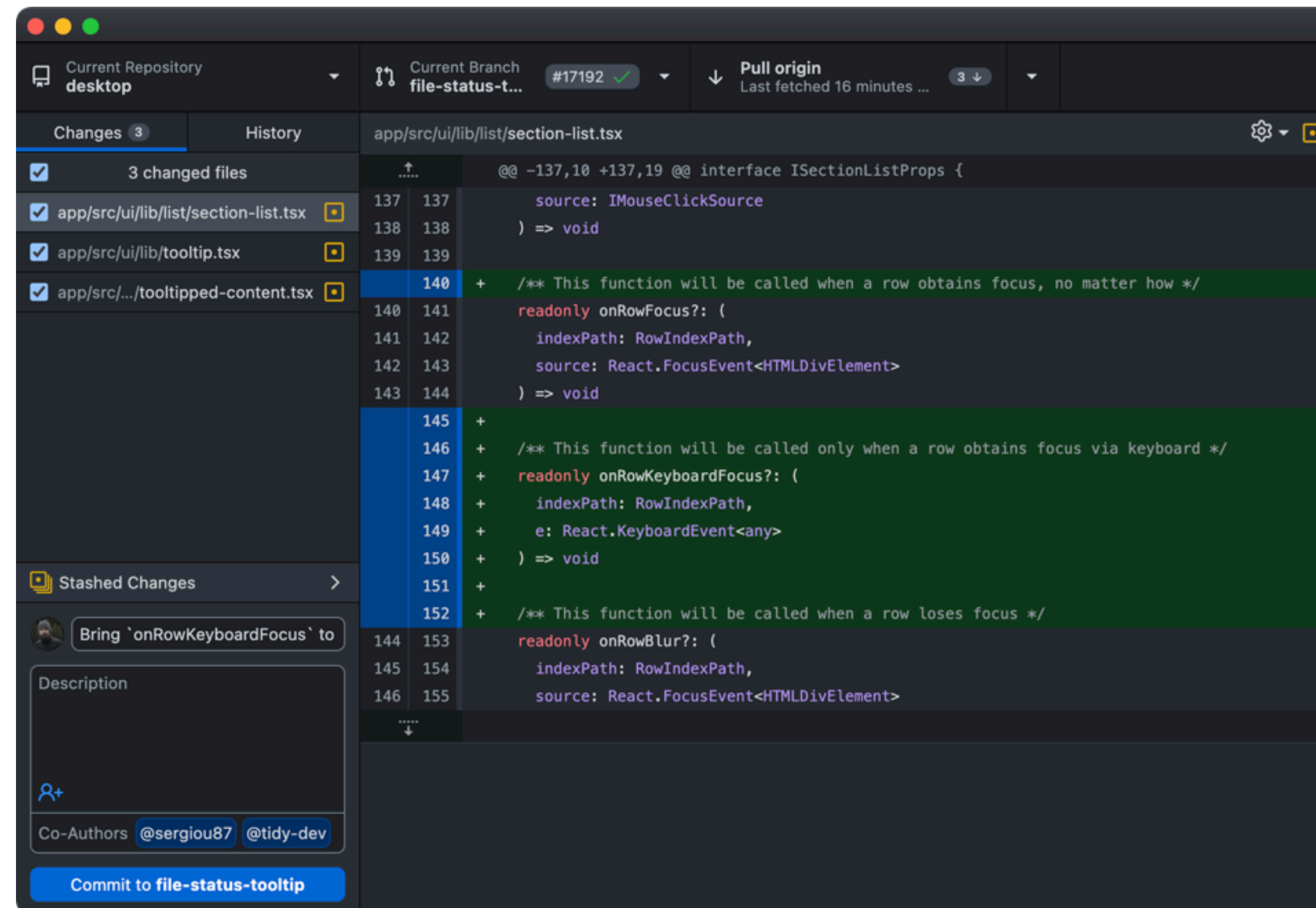
No packages published

[Publish your first package](#)

Github Desktop

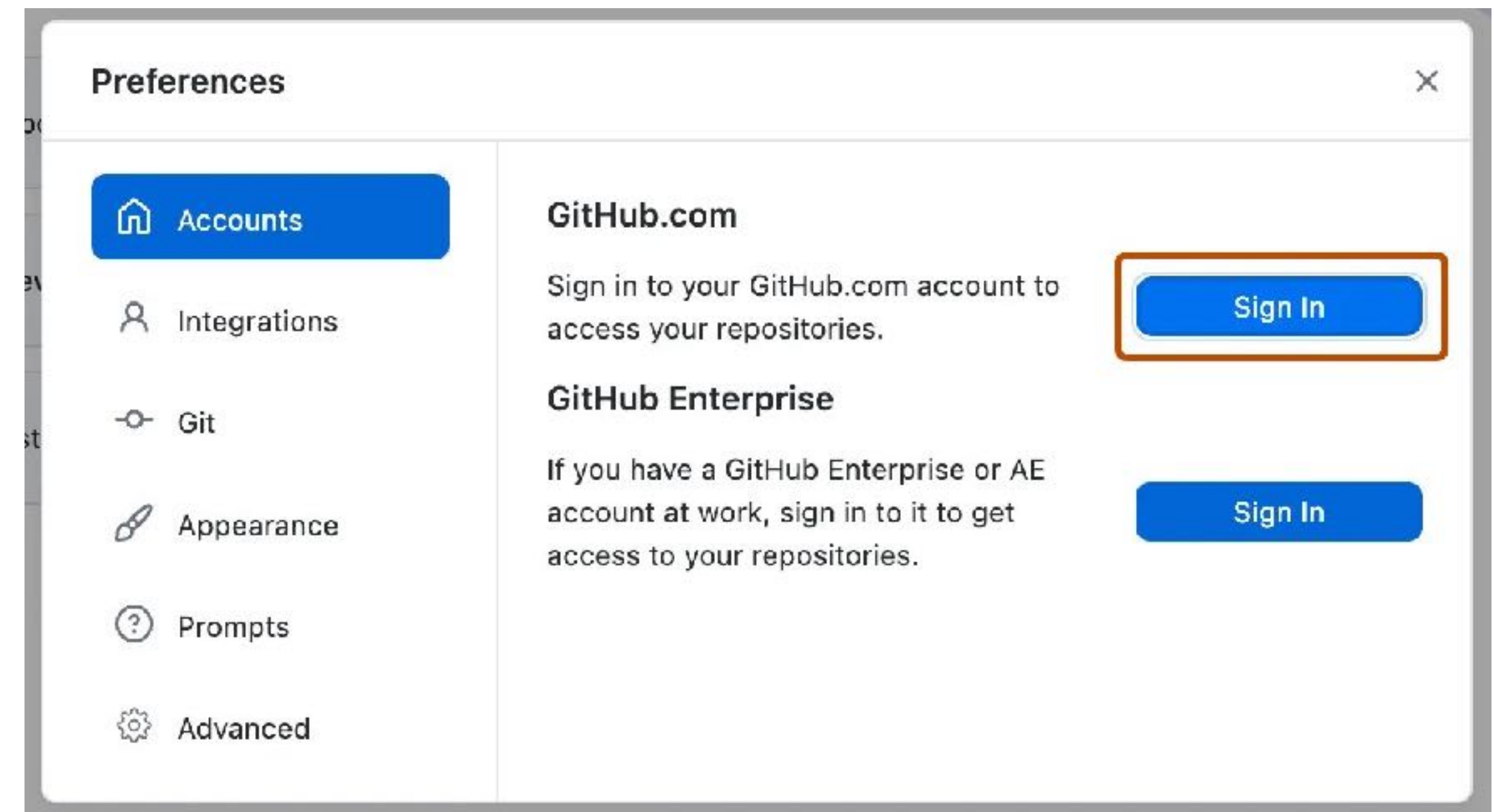
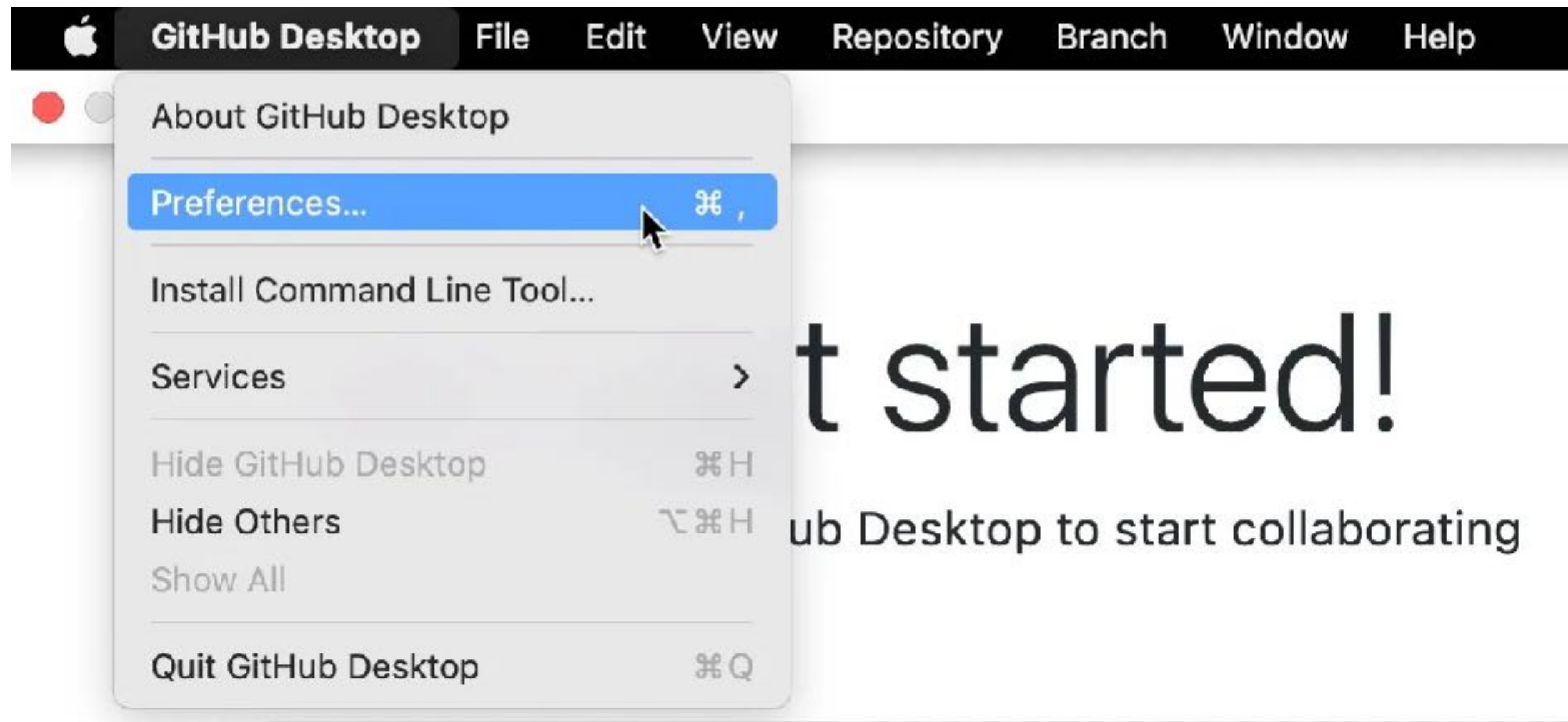
<https://desktop.github.com/>

- Interface graphique dédiée à Git et GitHub



Github Desktop

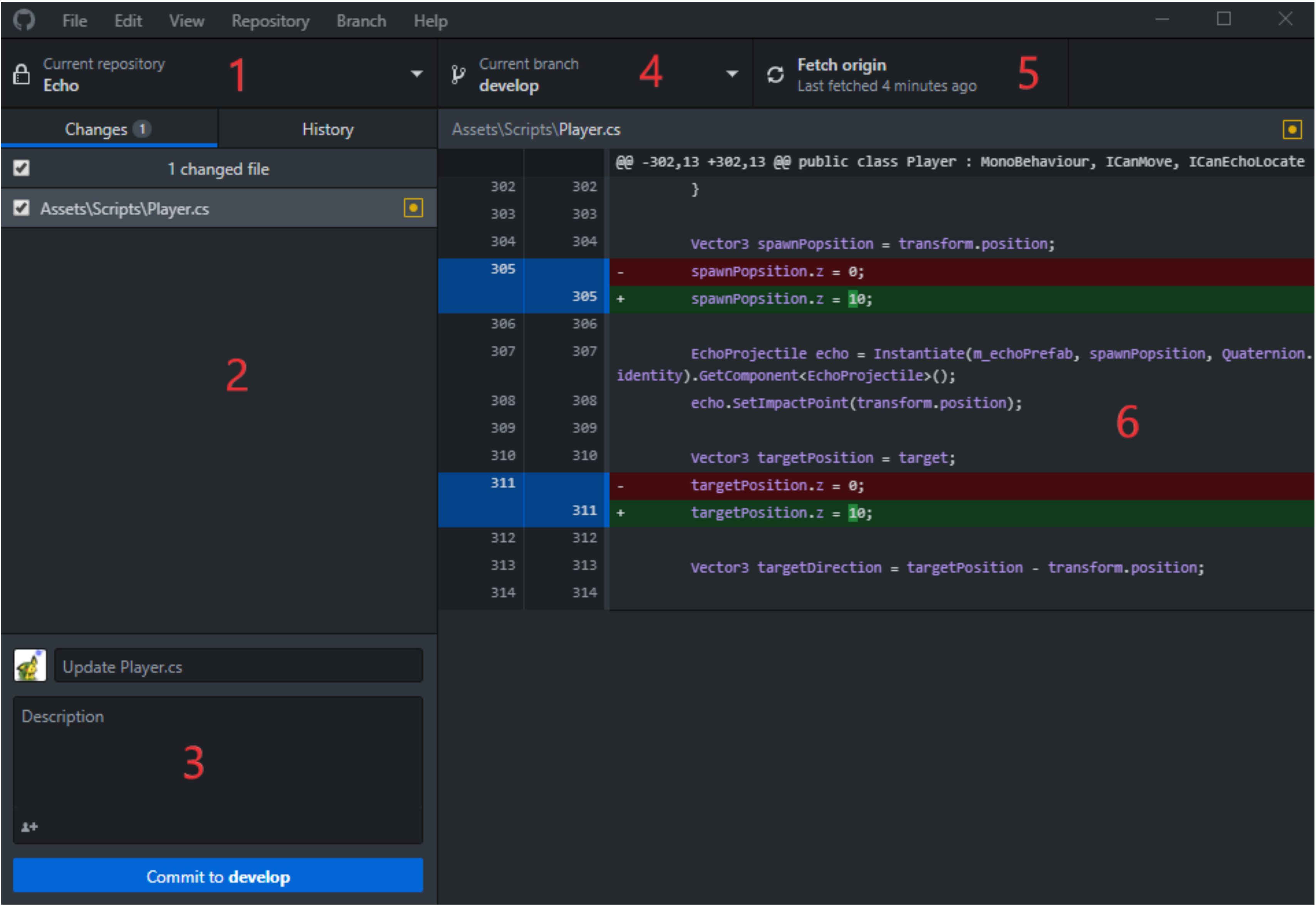
Configuration



Github Desktop

Interface

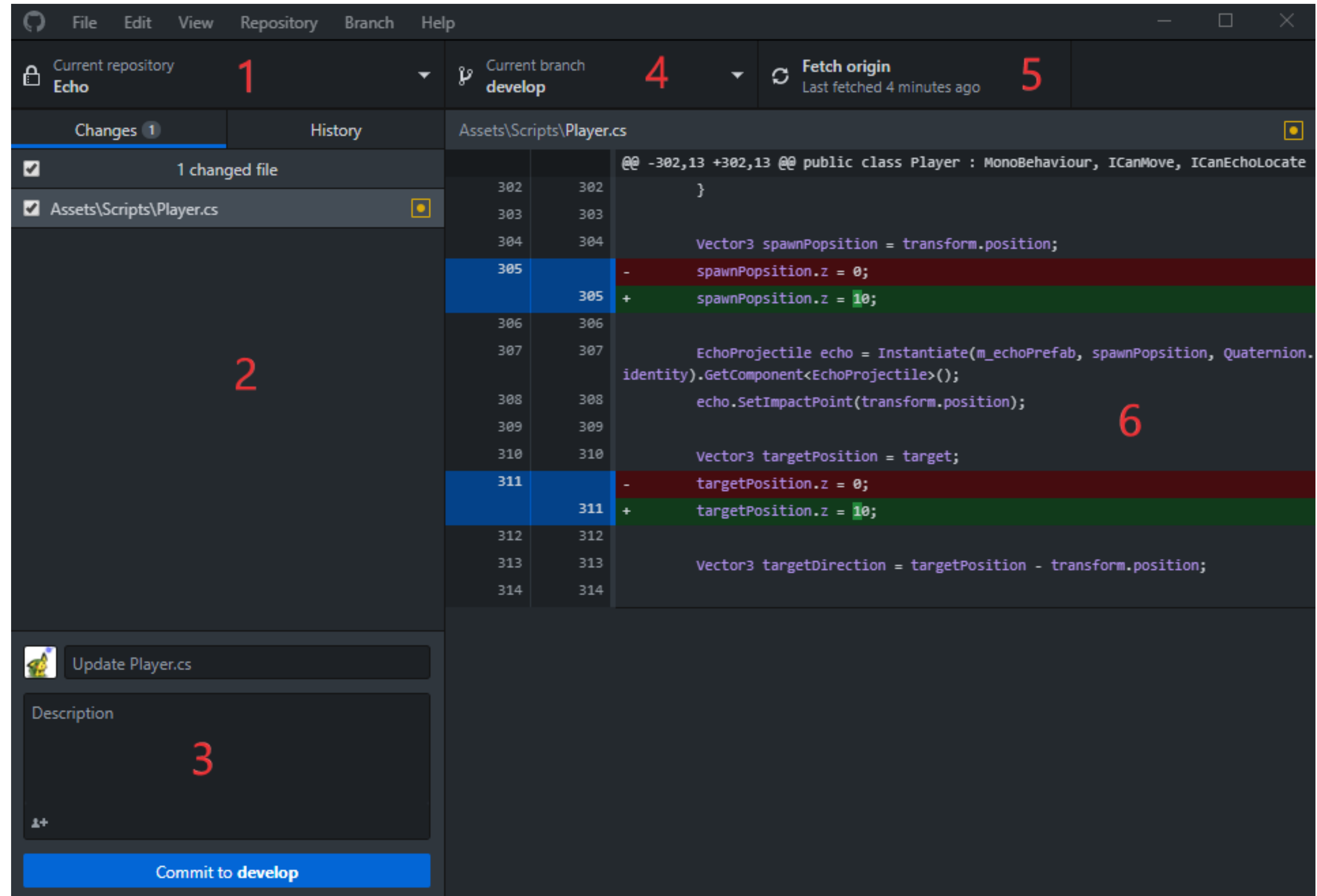
1. Explorateur de projets



Github Desktop

Interface

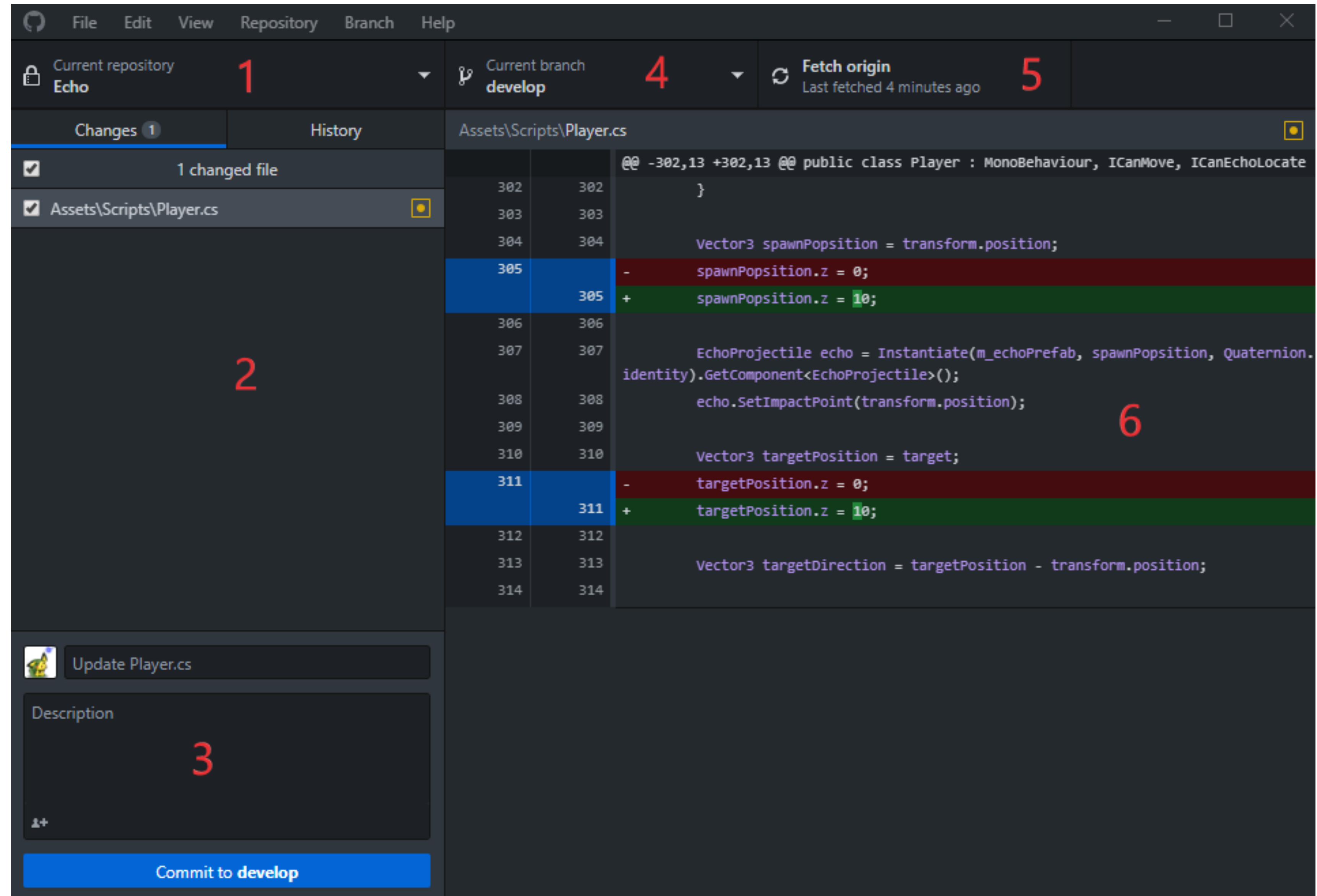
1. Explorateur de projets
2. Explorateur de fichiers



Github Desktop

Interface

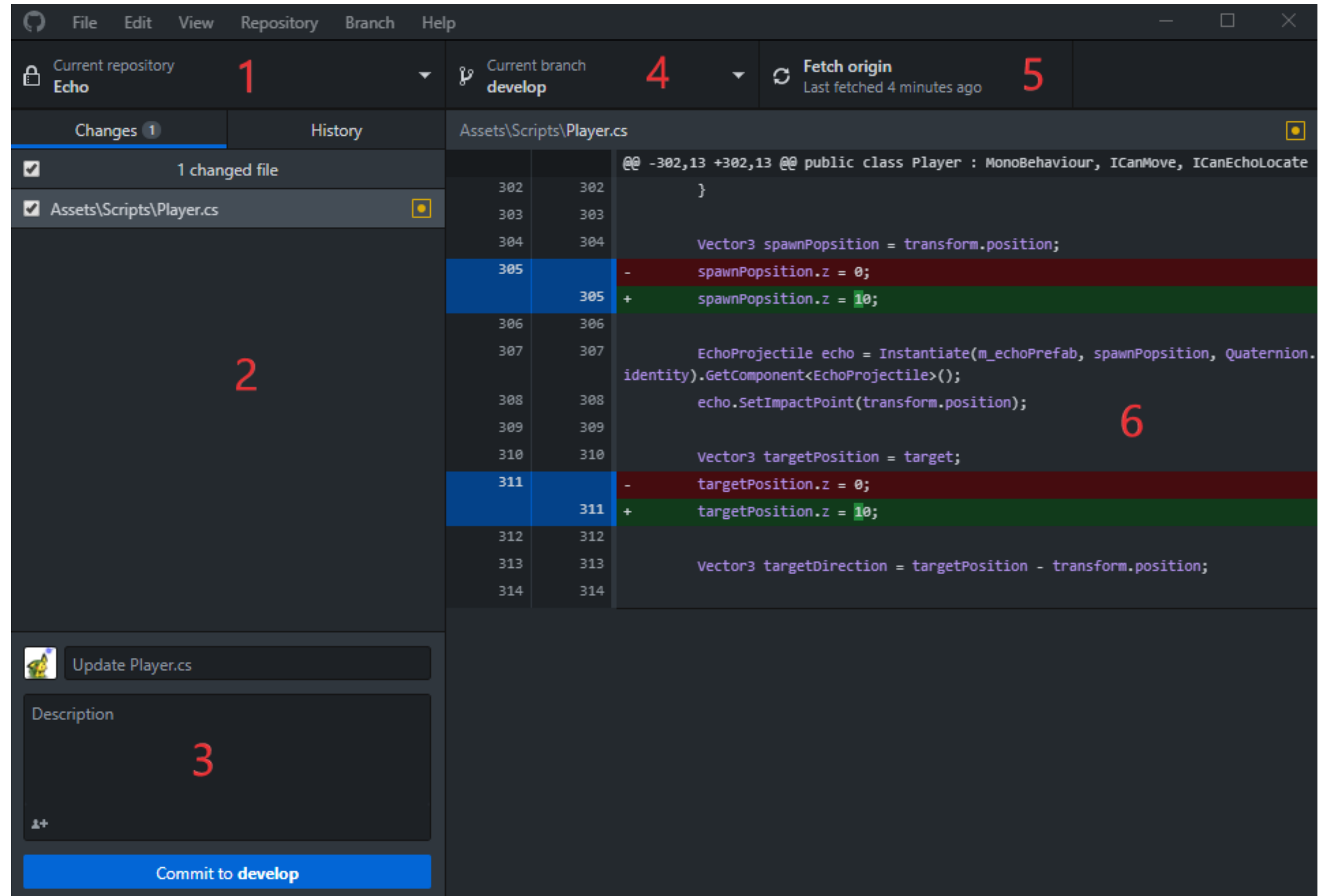
1. Explorateur de projets
2. Explorateur de fichiers
3. Gestion du commit



Github Desktop

Interface

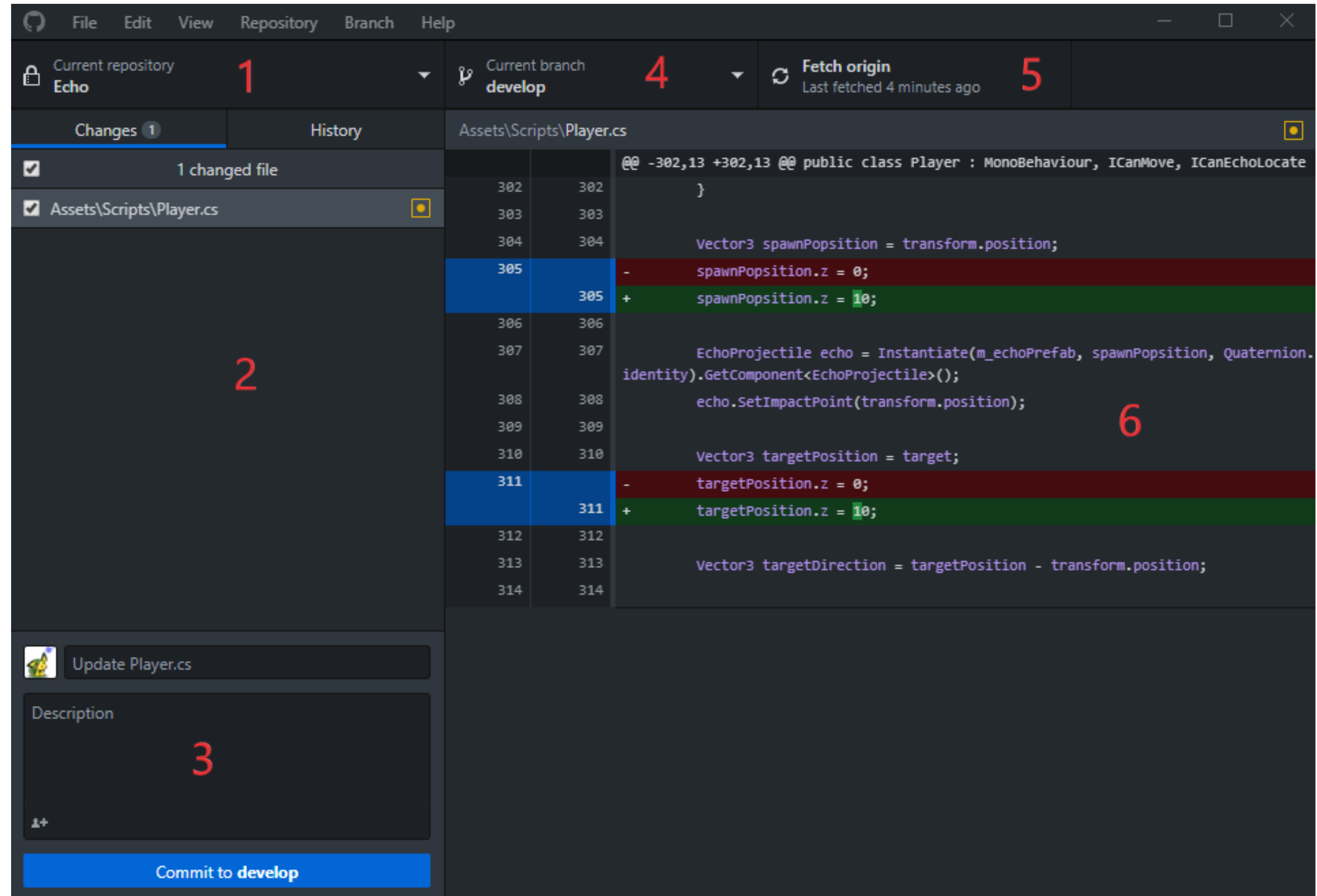
1. Explorateur de projets
2. Explorateur de fichiers
3. Gestion du commit
4. Explorateur de branches



Github Desktop

Interface

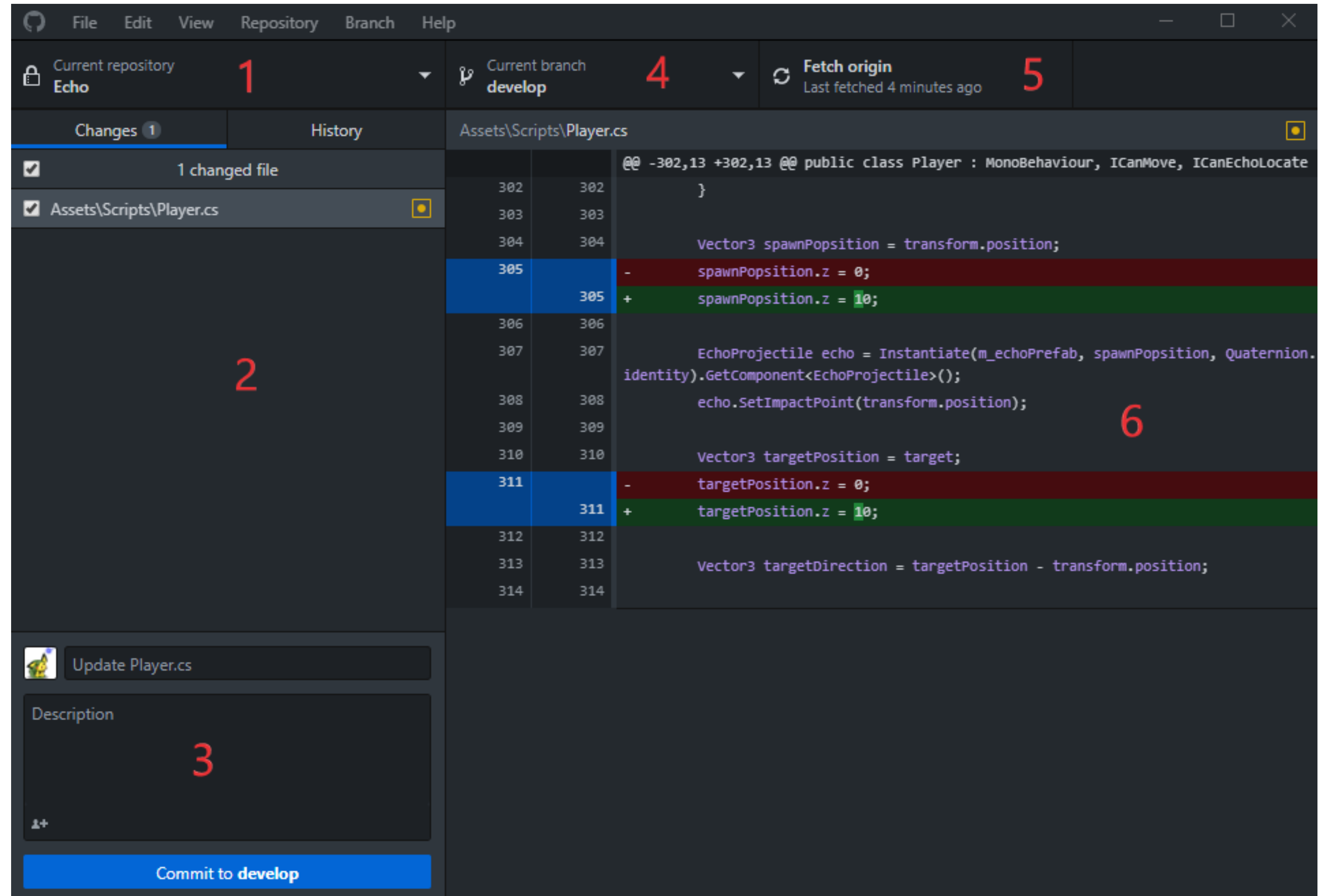
1. Explorateur de projets
2. Explorateur de fichiers
3. Gestion du commit
4. Explorateur de branches
5. Bouton Fetch/Pull/Push



Github Desktop

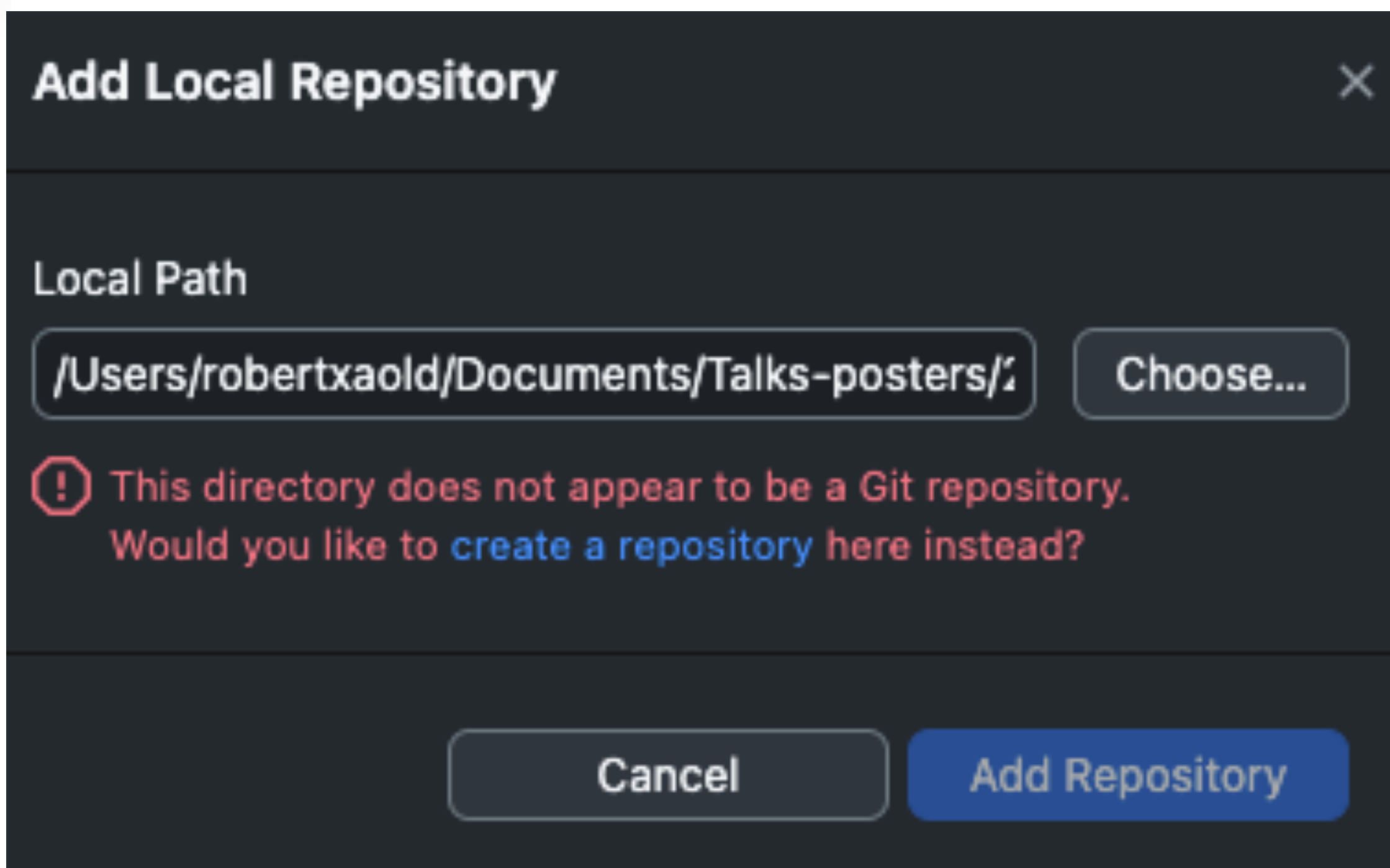
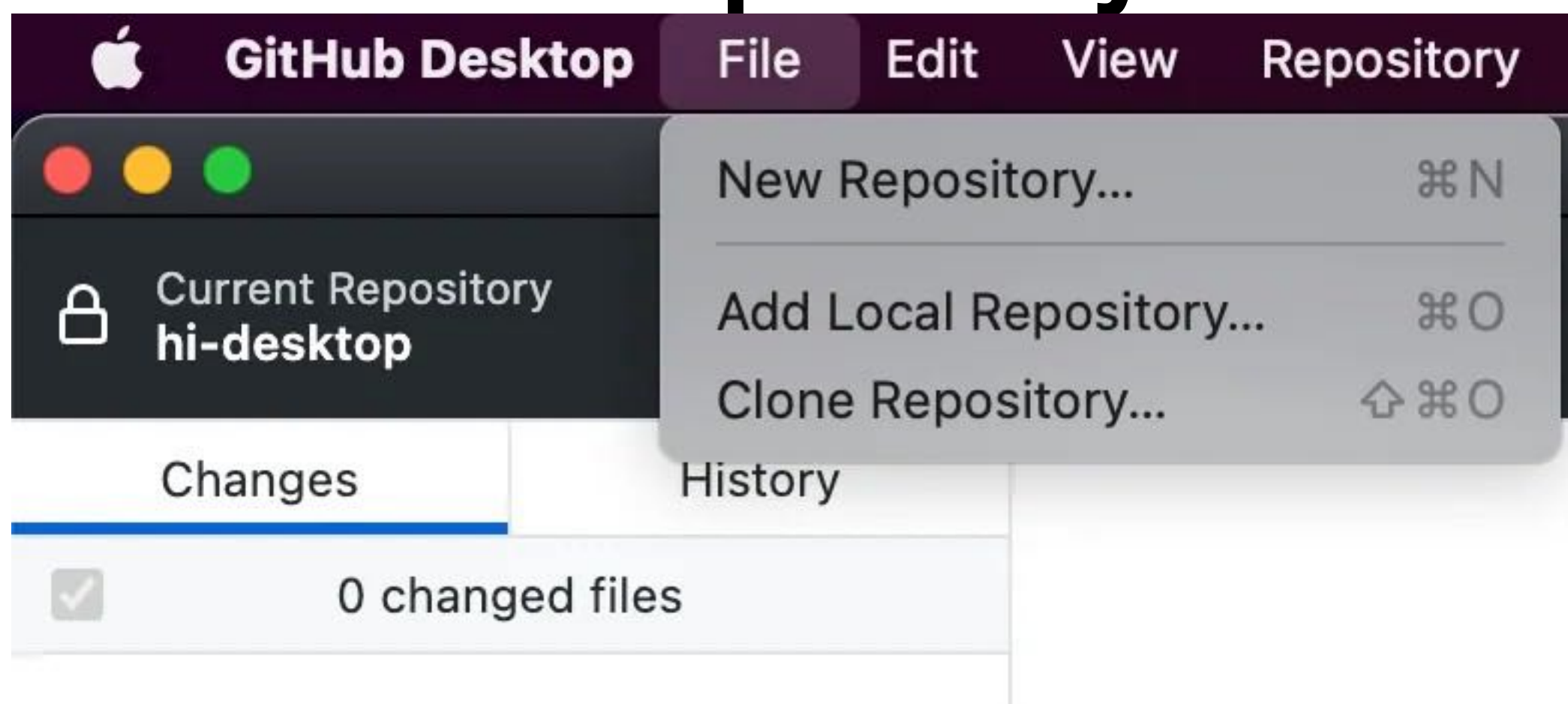
Interface

1. Explorateur de projets
2. Explorateur de fichiers
3. Gestion du commit
4. Explorateur de branches
5. Bouton Fetch/Pull/Push
6. Explorateur des modifications



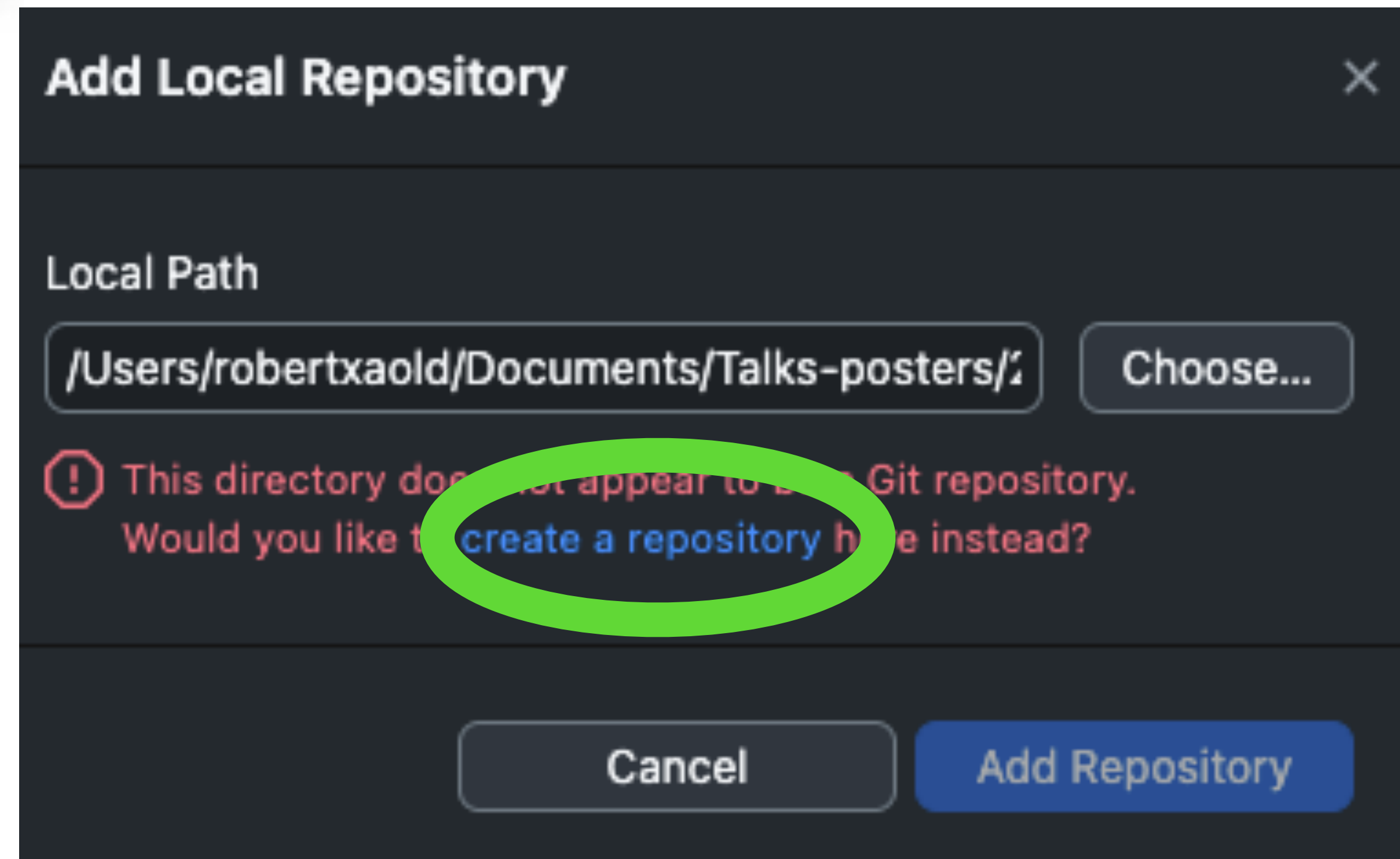
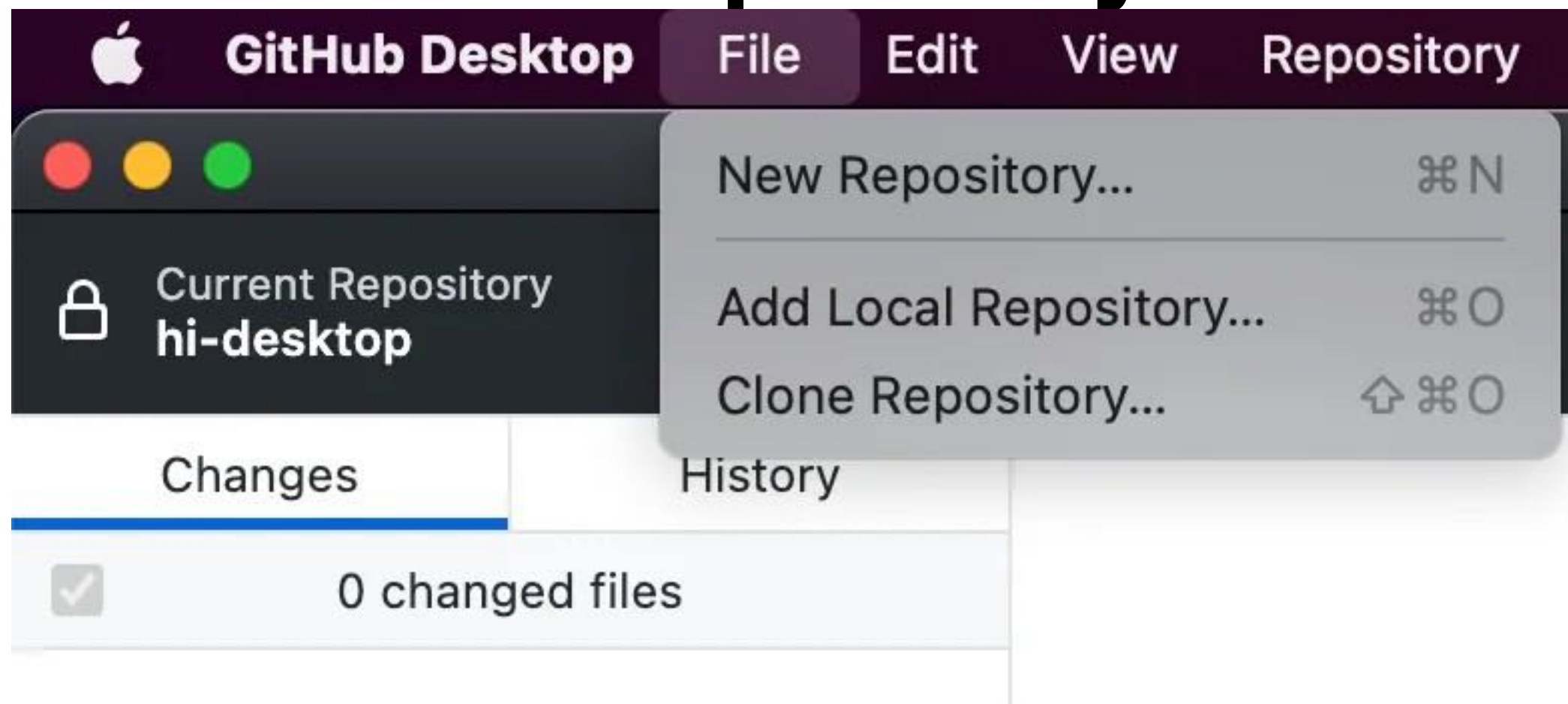
Github Desktop

Créer un repository



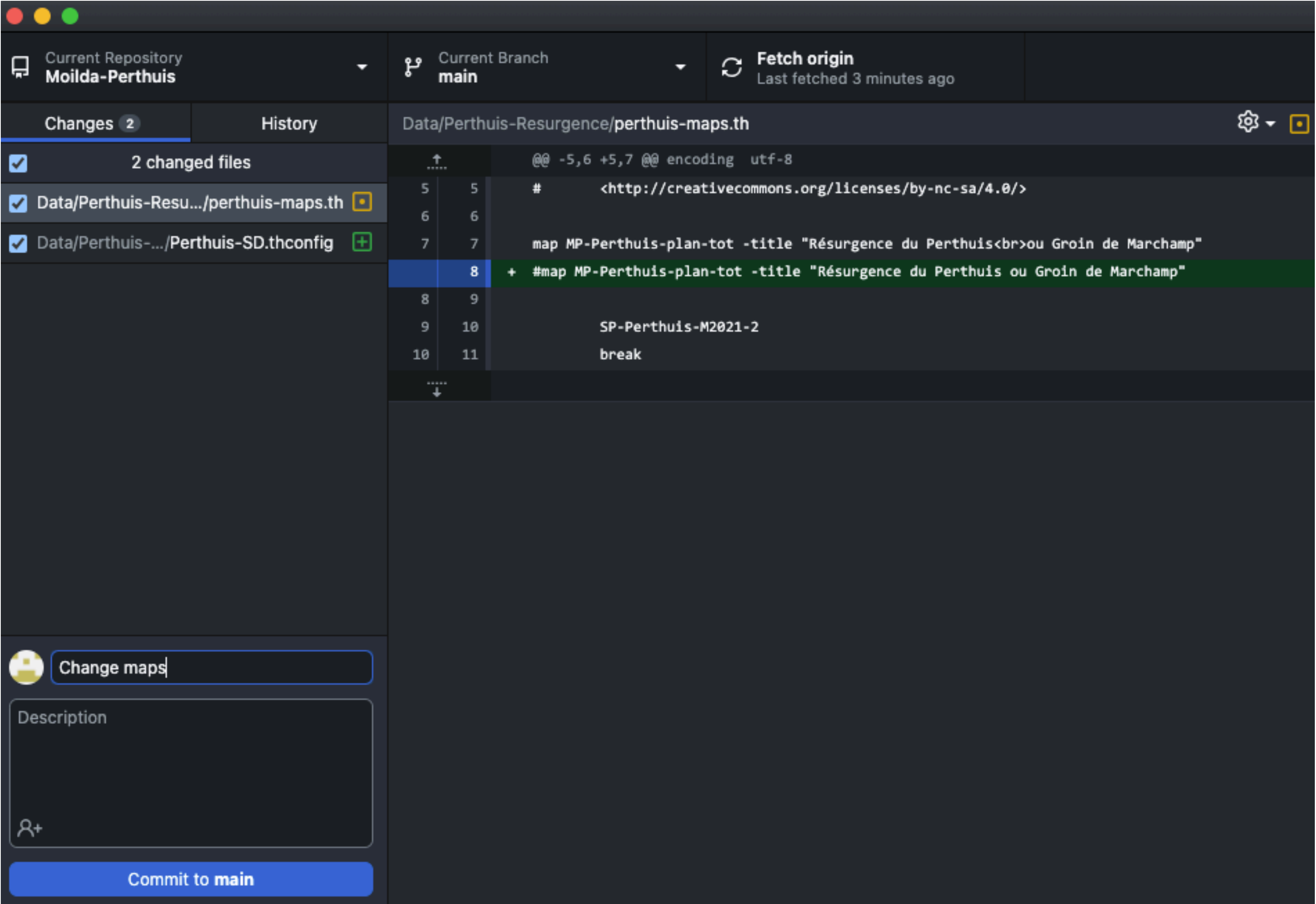
Github Desktop

Créer un repository



Github Desktop

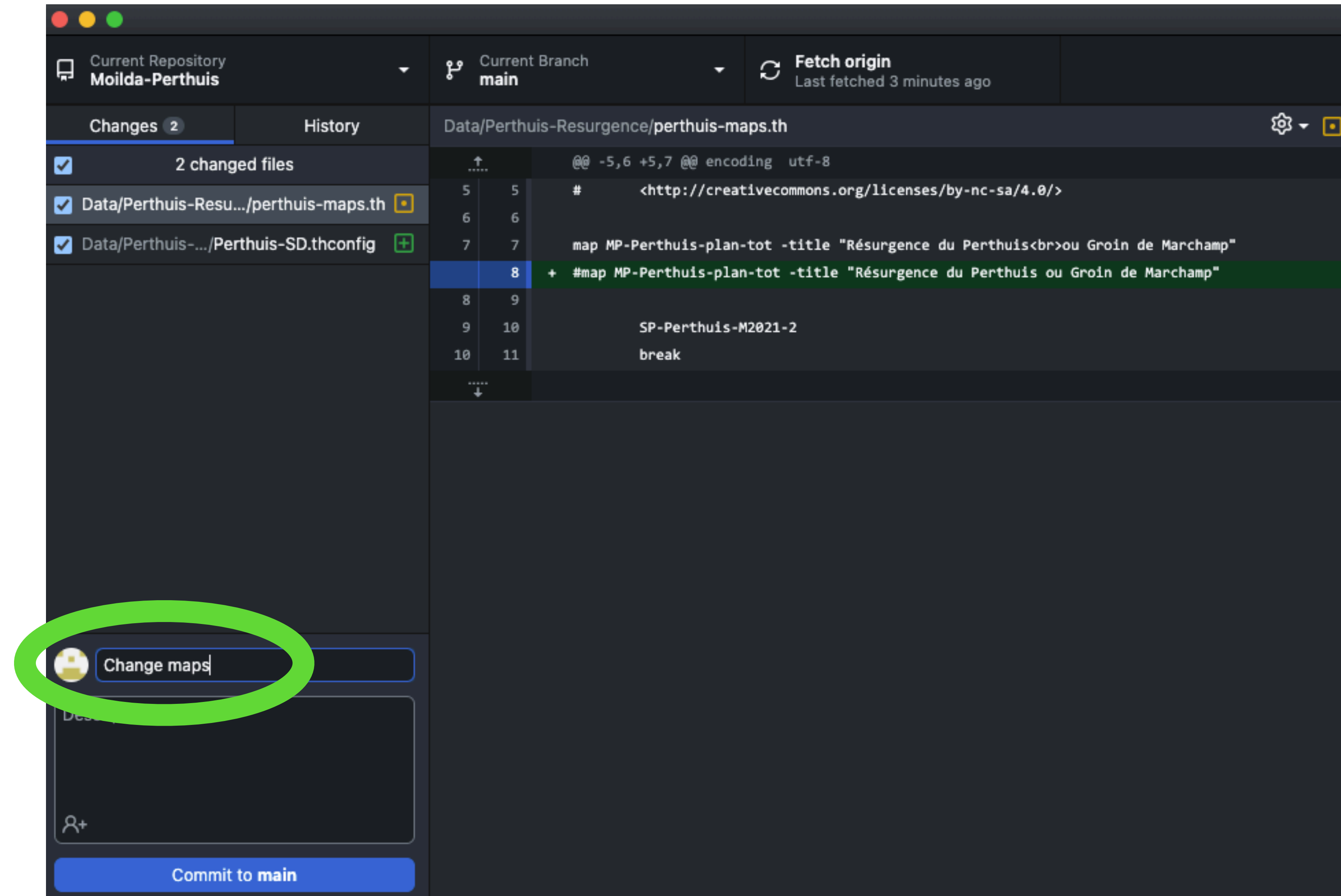
Faire un commit



Github Desktop

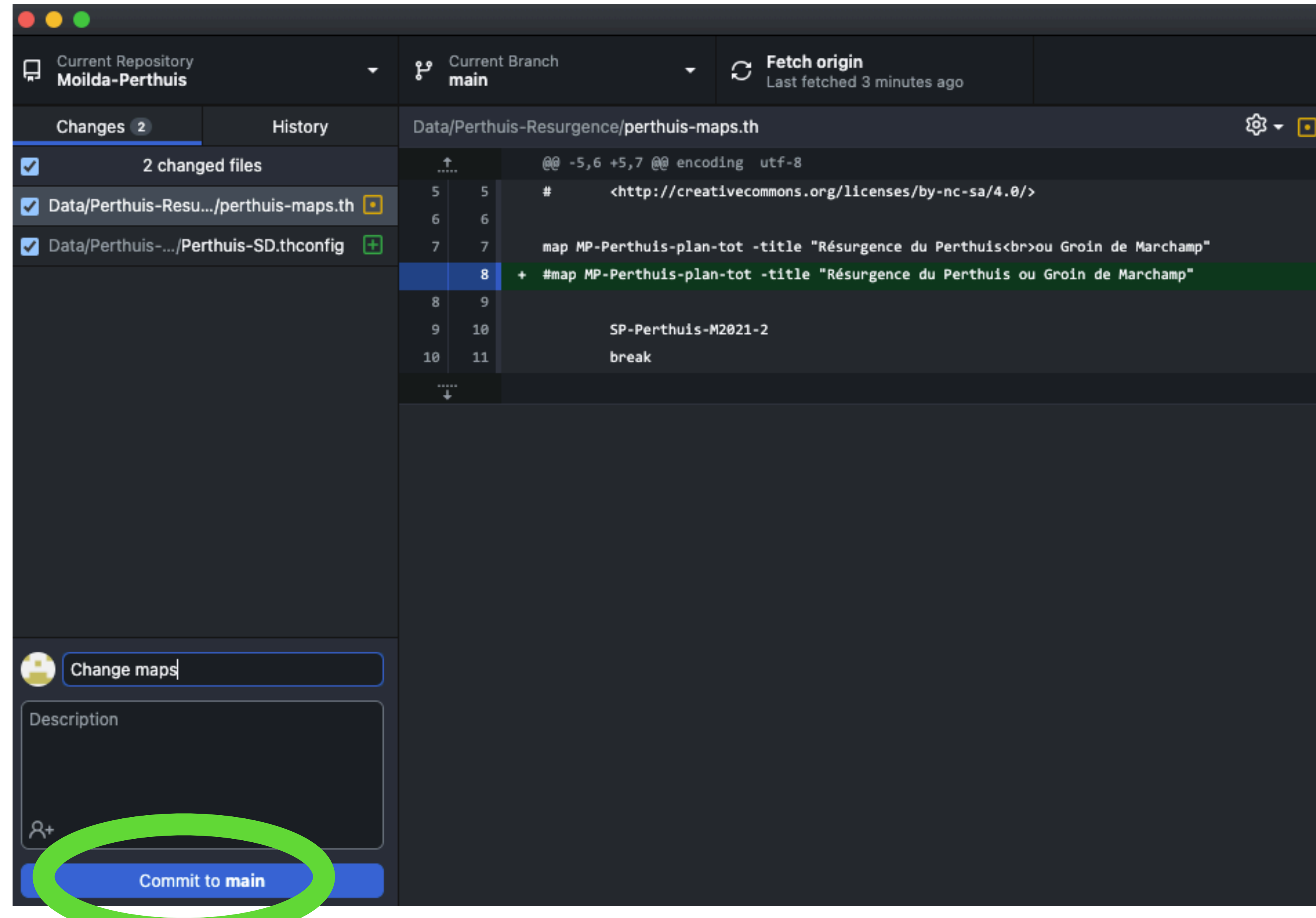
Faire un commit

- Donner un nom et une description



Github Desktop

Faire un commit

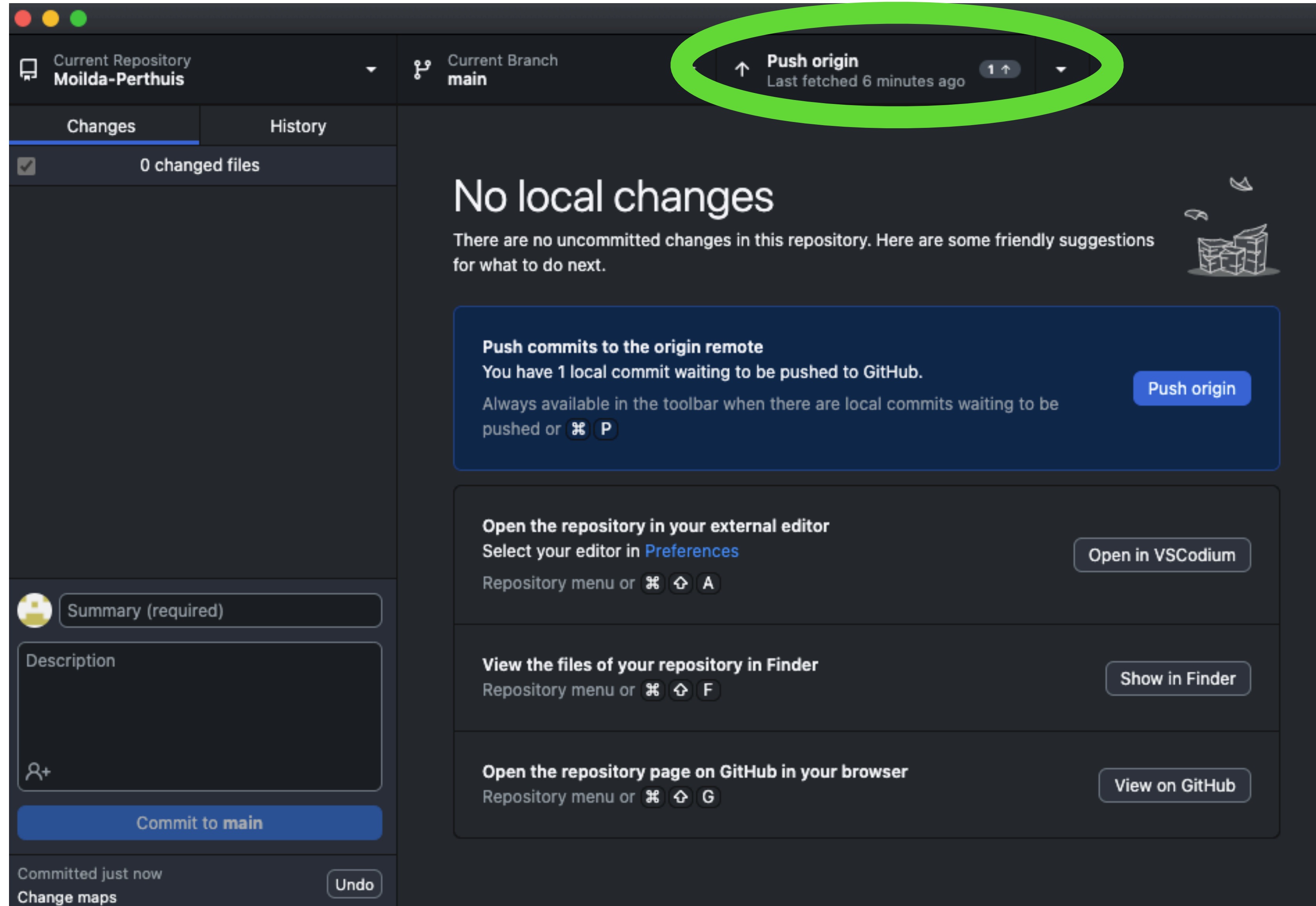


- Commit to **main**

Github Desktop

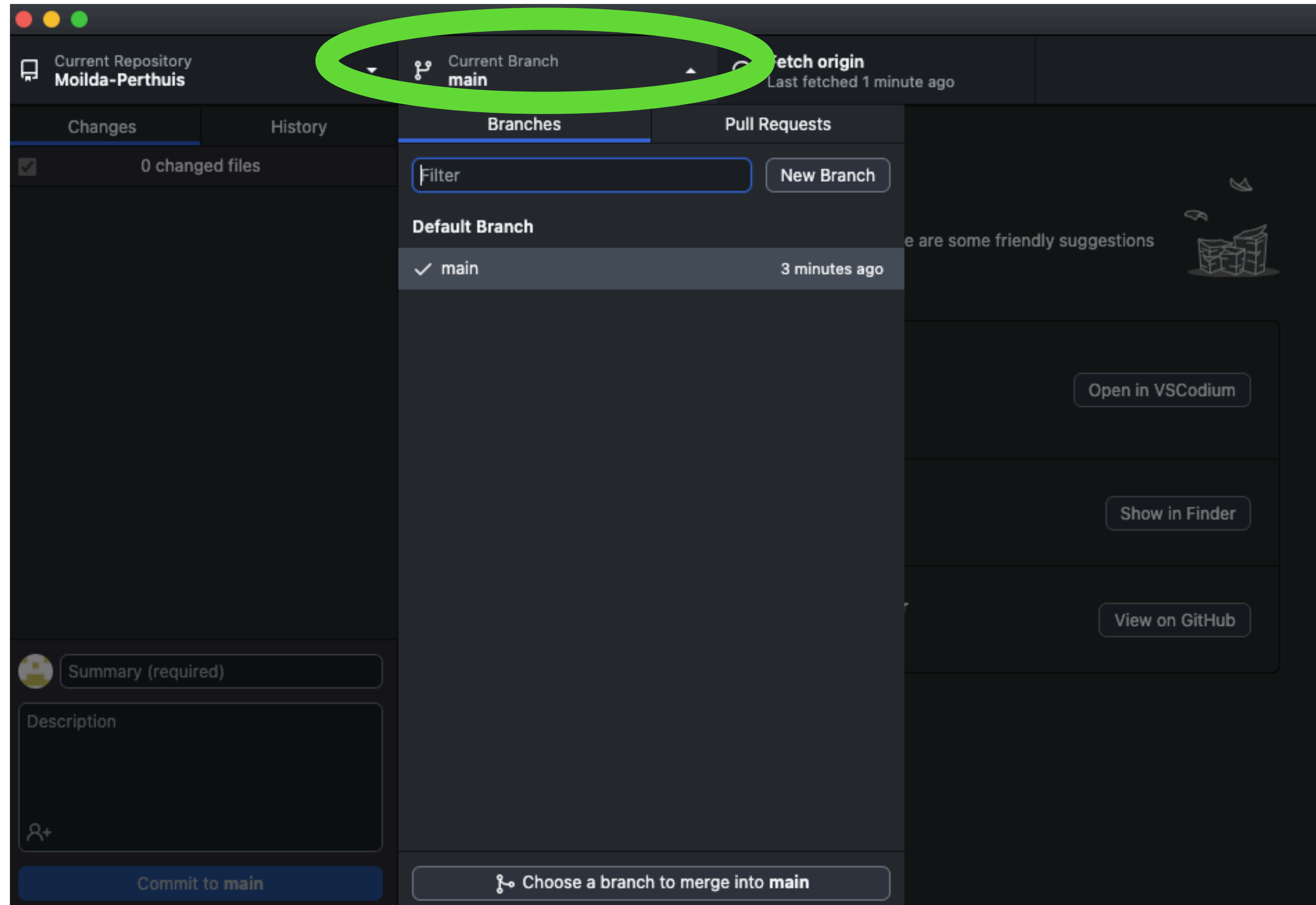
Faire un commit

- Push Origin



Github Desktop

Nouvelle Branche

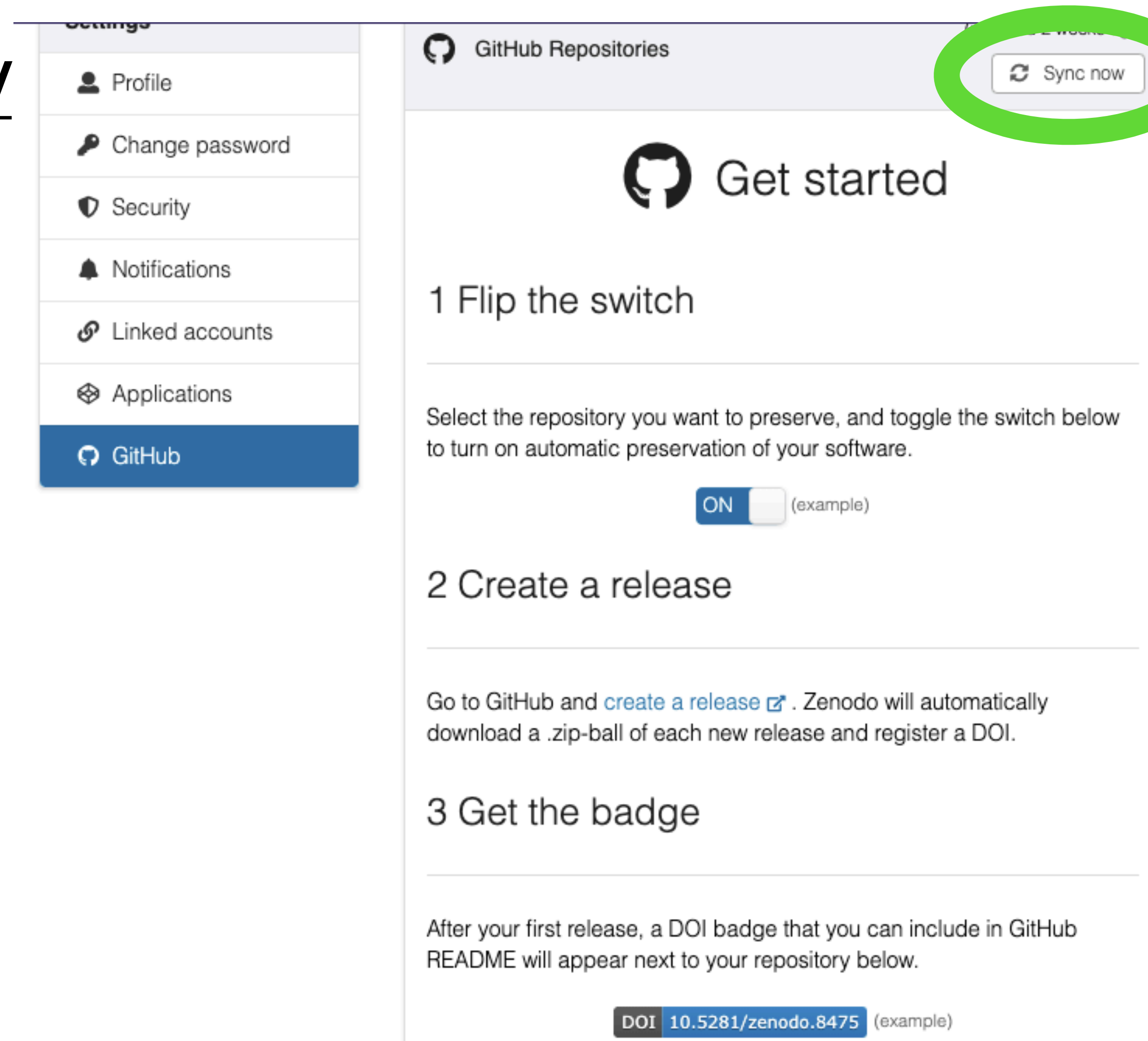


**Vous savez tout sur GitHub Desktop
(ou presque !)**

GitHub et DOI

Avec Zenodo : <https://zenodo.org/>

- Zenodo —> My Account —> GitHub
- Sync
- Bouger le bouton
- Créer une release sous Github
- Et le DOI est créé !



The screenshot shows the GitHub 'Settings' page. On the left, the 'Settings' menu is open, and the 'GitHub' option under 'Linked accounts' is selected. On the right, the 'GitHub Repositories' section is visible. A 'Sync now' button is highlighted with a green circle. Below this, the 'Get started' section contains three steps: 1. Flip the switch, 2. Create a release, and 3. Get the badge. Each step includes instructions and an example.

Settings

- Profile
- Change password
- Security
- Notifications
- Linked accounts
- Applications
- GitHub**

GitHub Repositories

Get started

1 Flip the switch

Select the repository you want to preserve, and toggle the switch below to turn on automatic preservation of your software.

ON (example)

2 Create a release

Go to GitHub and [create a release](#). Zenodo will automatically download a .zip-ball of each new release and register a DOI.

3 Get the badge

After your first release, a DOI badge that you can include in GitHub README will appear next to your repository below.

DOI 10.5281/zenodo.8475 (example)

GitHub et DOI

Avec Zenodo : <https://zenodo.org/>

- Zenodo —> My Account —> GitHub
- Sync
- Bouger le bouton
- Créer une release sous Github
- Et le DOI est créé !

DOI Badge

This badge points to the latest released version of your repository. If you want a DOI badge for a specific release, please follow the DOI link for one of the specific releases and grab badge from the archived record.

DOI

10.5281/zenodo.10606463

Markdown

```
[[DOI]](https://zenodo.org/badge/751342655.svg){https://zenodo.org/doi/10.5281/zenodo.10606462}
```

reStructuredText

```
.. image:: https://zenodo.org/badge/751342655.svg
   :target: https://zenodo.org/doi/10.5281/zenodo.10606462
```

HTML

```
<a href="https://zenodo.org/doi/10.5281/zenodo.10606462"></a>
```



My account

Settings

Profile

Change password

Security

Notifications

Linked accounts

Applications

GitHub

Repositories >



robertxa/Simple_Swath

ON

Releases

Create release

DOI 10.5281/zenodo.10606463

1.0.1 simple_swath, a simple Python code to extract swath profile using a shapefile

Published


DOI: 10.5281/zenodo.10606463

2 weeks ago


First release

GitHub et documentation

Avec Read The Docs




Read the Docs

 **TUTORIALS**


- Read the Docs tutorial
- Getting started with Sphinx
- Getting started with MkDocs

☒ **Importing your documentation**

- Automatically import your docs
- Manually import your docs
- Building your documentation

 **EXPLANATION**

- Choosing between our two platforms
- Continuous Documentation Deployment
- Understanding offline formats

 Read the Docs v: stable ▼

🏠 / Importing your documentation

 [Edit on GitHub](#)

Importing your documentation

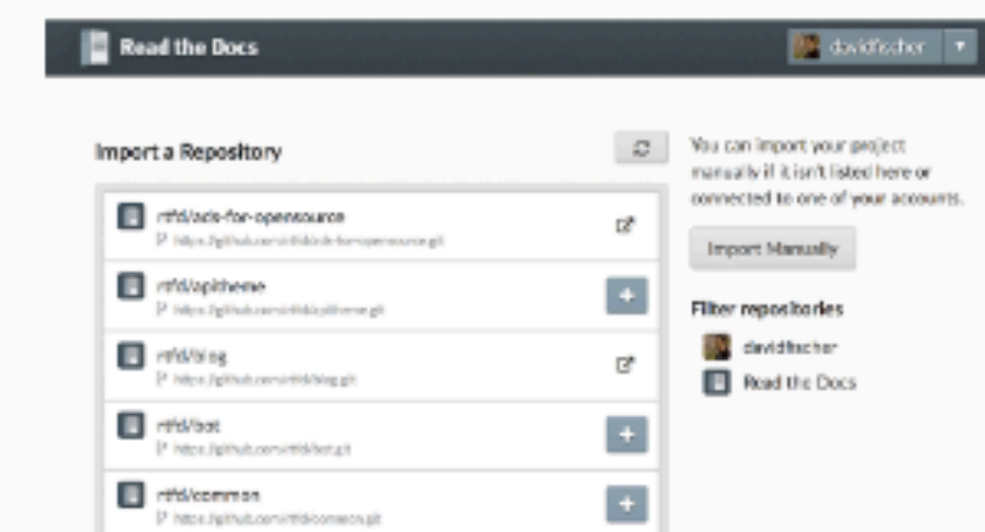
To import a public documentation repository, visit your [Read the Docs dashboard](#) and click [Import](#). For private repositories, please use [Read the Docs for Business](#).

Automatically import your docs

If you have [connected your Read the Docs account](#) to GitHub, Bitbucket, or GitLab, you will see a list of your repositories that we are able to import. To import one of these projects, just click the import icon next to the repository you'd like to import. This will bring up a form that is already filled with your project's information. Feel free to edit any of these properties, and then click [Next to build your documentation](#).

Manually import your docs

If you have not [connected a Git provider account](#), you will need to select **Import Manually** and enter the information for your repository yourself. You will also need to manually configure the webhook for your repository as well. When importing your project, you will be asked for the repository URL, along with some other information for your new



Importing a repository

Bilbio non exhaustive

- ***Git***

- <https://git-scm.com/>
- <https://www.pierre-giraud.com/git-github-apprendre-cours/>
- <https://www.atlassian.com/fr/git/glossary#commands>
- <https://perso.liris.cnrs.fr/pierre-antoine.champin/enseignement/intro-git/>

- ***GitHub***

- <https://desktop.github.com/>
- <https://gist.github.com/Marsgames/2eb2e0321302640efafa4067b483b427>
- <https://docs.github.com/fr/desktop/overview/getting-started-with-github-desktop>
- <https://docs.github.com/fr/desktop>