Github Desktop を
利用したチーム開発手法

1. 導入編

Git・Github・Github Desktop
について知ろう！

1-1. Gitとは何か？

Gitとは何か？

> Git（ギット）は、プログラムのソースコードなどの変更履歴を記録・追跡するための分散型バージョン管理システムである。
> 出典: https://ja.wikipedia.org/wiki/Git

分散型……バージョン管理システム……？

ゲームのセーブデータみたいに

今の状態を保存しておくもの

これをリポジトリ、と呼びます

作業が一段落したら、今の状態を保存します

履歴が溜まっていきますね
すると……？

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

いつ、何の機能を実装したのか分かる

履歴には説明文をつけるので、
分かりやすいです

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

バグが発生した時も、
元に戻せる

過去のデータに、一瞬で元通り

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

変なデータ、必要なし

- program.c
- program(1).c
- program(2).c
- program(3)_これが最終版.c
- program_提出用.c

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

その上、複数人で作業するための機能も沢山

編集の重複を防いだり、差分を分かりやすくしたり……

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

Gitとは何か？

ソースコードをいい感じに管理するやつ

1-2. Githubとは何か？

Githubとは何か？

> GitHub（ギットハブ）は、ソフトウェア開発のプラットフォームであり、ソースコードをホスティングする。コードのバージョン管理システムにはGitを使用する。
> 出典: https://ja.wikipedia.org/wiki/GitHub

ソースコードを……ホスティングする……？

つまり？

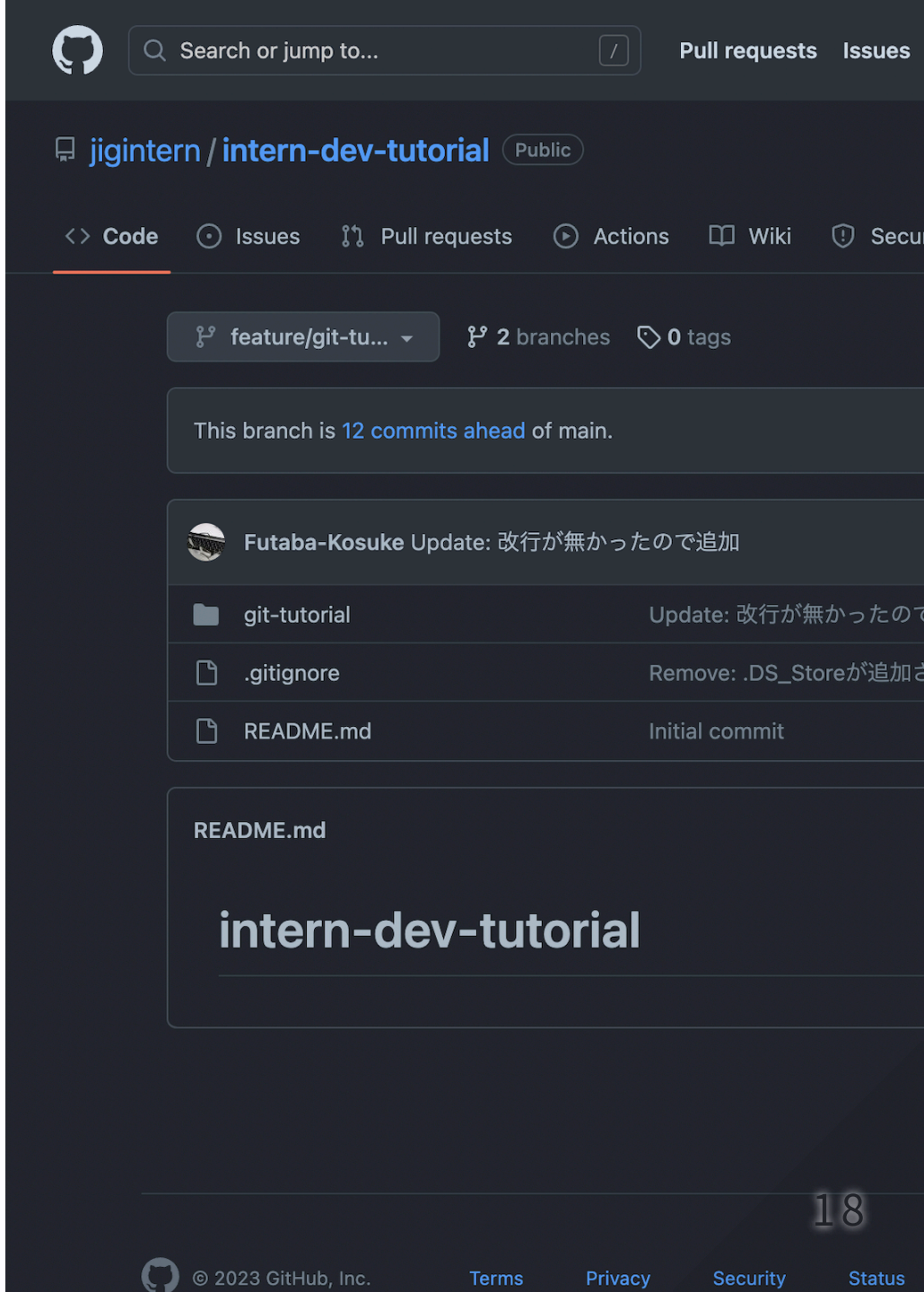
Gitをオンラインで管理するやつ

Gitのデータをアップロードして共有・バックアップするためのサービスです

https://github.com/

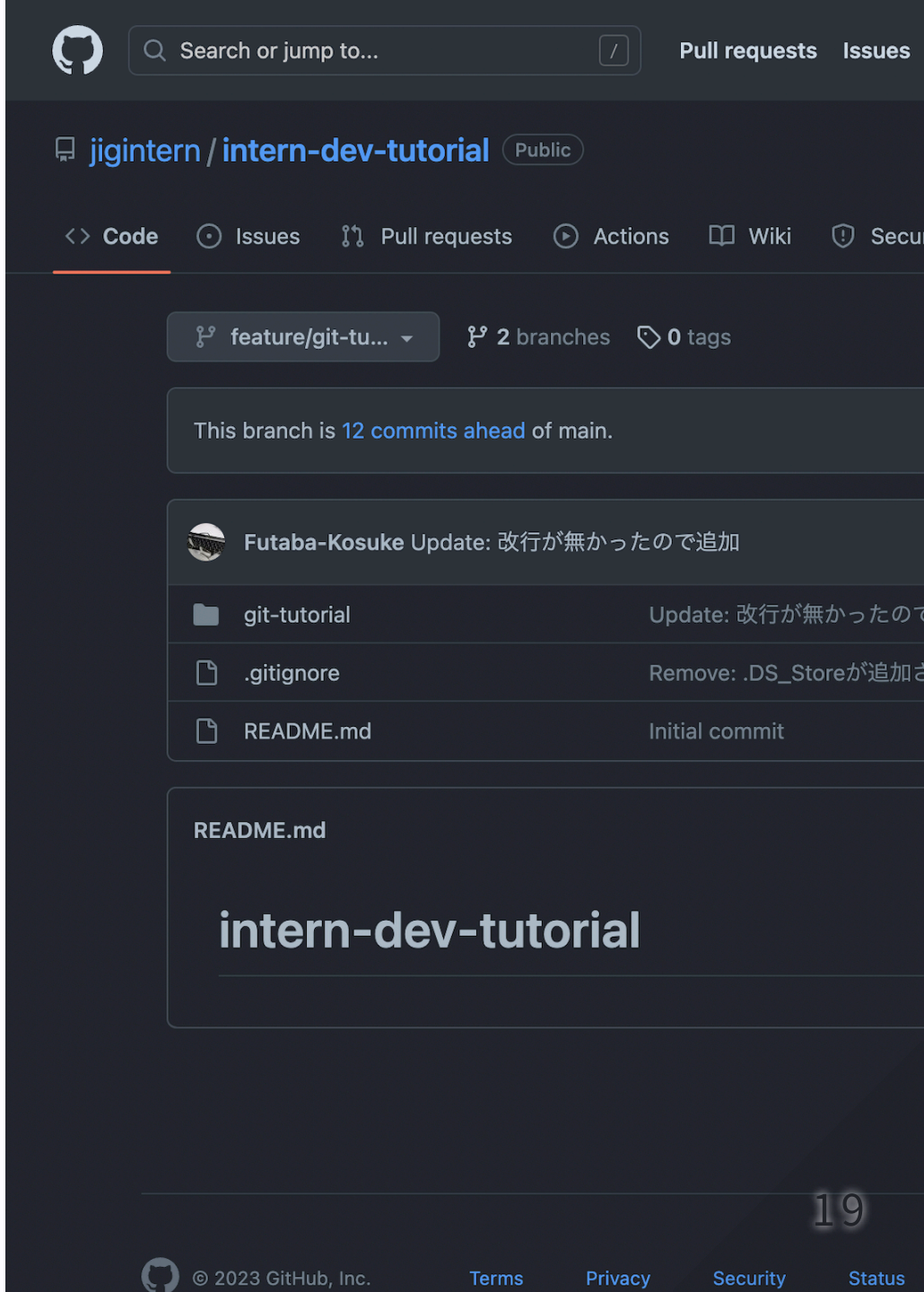
記録がある程度溜まったら、アップロード

PC上のデータが吹っ飛んでも、
オンラインに残ります



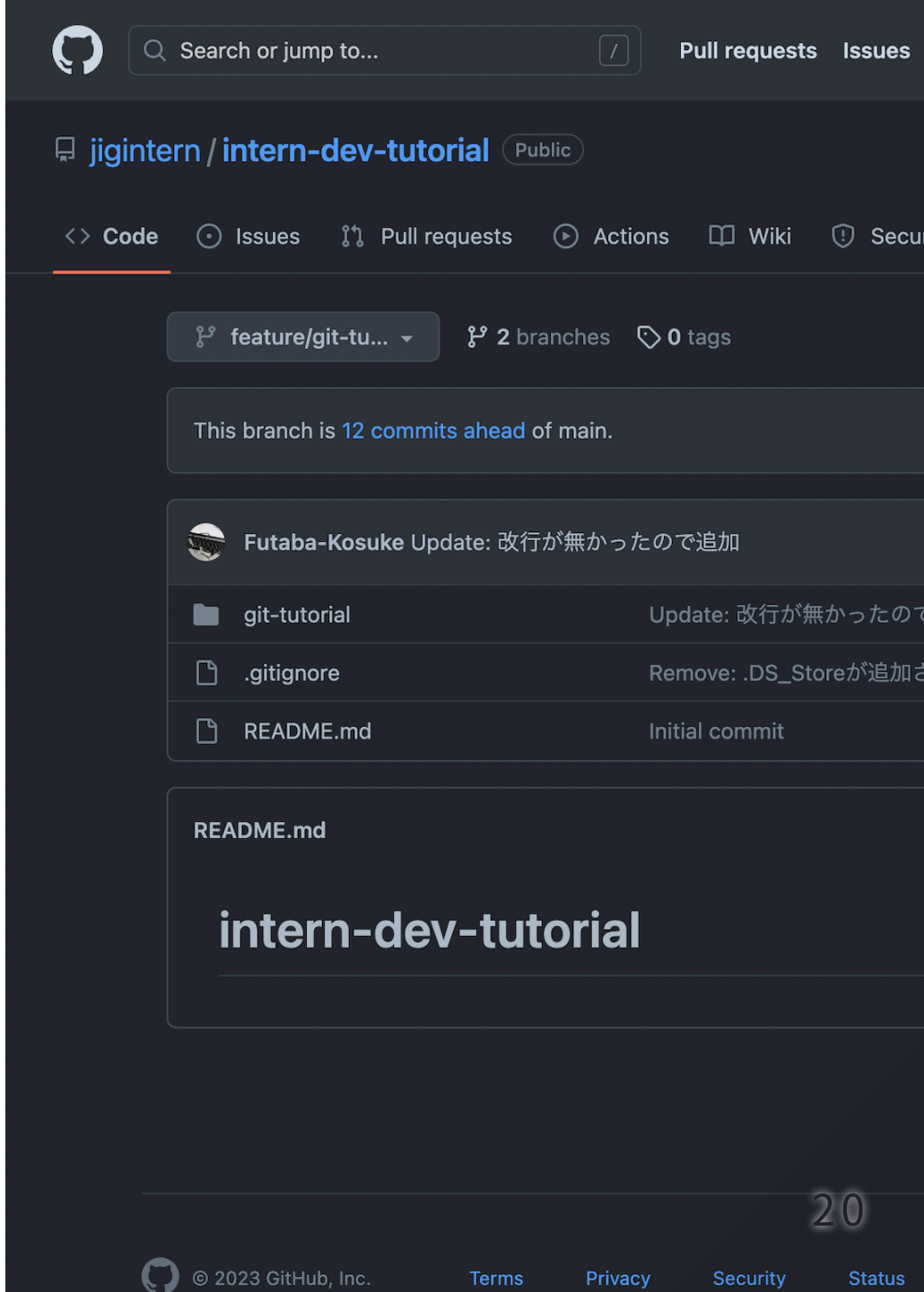
複数人で、データ共有

クラウドでの共有は不要です



他の人も、閲覧可能

共同開発の場としても、能力を
アピールする場としても



Githubとは何か？

gitをオンラインで管理するやつ

1-3. Github Desktopとは何か？

Github Desktopとは何か？

> GitHub Desktop では、コマンド ラインや Web ブラウザーではなく GUI を使用して GitHub と対話できます。
> 出典: https://docs.github.com/ja/desktop

Gitは、コマンドライン で操作するのが一般的

→

```
* commit 0a3390048cecd60b09f74ae0108c41425823e572 (HEAD -> featu
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:55:15 2023 +0900
|
|     Update: 改行が無かったので追加
|
* commit 0b3e9504485cefa8a5392cab495ea922b6ec448a
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:16:53 2023 +0900
|
|     Move: 命名を変更
|
* commit d9a89524f7e21d00f4bc232dd211cb8f84c153a5
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 15:11:16 2023 +0900
|
|     Remove: .DS_Storeが追加されてしまっていたので削除
|
* commit cb4d7e8827b0f854d8c9d91d3308c3ee059a7f08
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:53:06 2023 +0900
|
|     Update: <br>を削除
|
* commit d34628768c833533172f3b85f3daf5b9e0ac3d36
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:47:16 2023 +0900
|
|     Update: mermaidによるGit図解を追加
|
* commit dfcf7fa4d632543aff79f323074f237832ec1b09
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Wed Jun 14 14:39:39 2023 +0900
|
|     Update: Github Desktopハンズオンの内容を記載
|
* commit afe159403e4a7b83491319bcdcd68eb03ff76154
| Author: niba2828 <k.futaba2828@gmail.com>
| Date:   Tue Jun 13 16:24:58 2023 +0900
|
|     Update: Github Desktopを利用したチーム開発手法、の本編の基
|
* commit 4d89428a91675ecbcb0f9850ed153c7ed9d03d1c
:
```

Github Desktopを使えば、コマンド無しで操作できます

環境構築もインストールするだけ！
事前にインストールして貰いました

UIはこんな感じ

表示されているのは、この資料を書いている時の様子です

Current Repository
intern-dev-tutorial

Current Branch
feature/git-tutorial

Fetch origin
Last fetched 1 minute ago

An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

Changes 3 | History

3 changed files

- git-tutorial/img.../01_git_tree.png
- git-tutorial/imgs.../02_github.png
- git-tutorial/slide.md

Summary (required)

Description

Commit to feature/git-tutorial

git-tutorial/slide.md

```
@@ -0,0 +1,200 @@
+---
+marp: true
+theme: uncover
+paginate: true
+---
+
+# **Github Desktop** を
+# 利用したチーム開発手法
+
+---
+
+## 1. 導入編
+
+Git・Github・Github Desktop
+について知ろう！
+
+---
+<!--
+header: 1-1. Gitとは何か？
+-->
+
+## 1-1. Gitとは何か?
+
+---
+
+### Gitとは何か?
+
+> Git（ギット）は、プログラムのソースコードなどの変更履歴を記録・追跡するための分散型バージョン管理システムである。
+> 出典: https://ja.wikipedia.org/wiki/Git
+
+
+---
+
+### 分散型……バージョン管理システム……?
+
+---
+
```

今回はGithub Desktopを使って開発します

お気に入りのGitの操作方法があれば、そちらで進めていただいても問題ありません！

2. Github Desktopを触ってみよう

Github Desktopを実際に使ってみよう！

2-1. clone: Githubリポジトリを丸ごとダウンロードしよう

Githubからリポジトリをダウンロードしてみましょう

これをclone（クローン）と呼びます

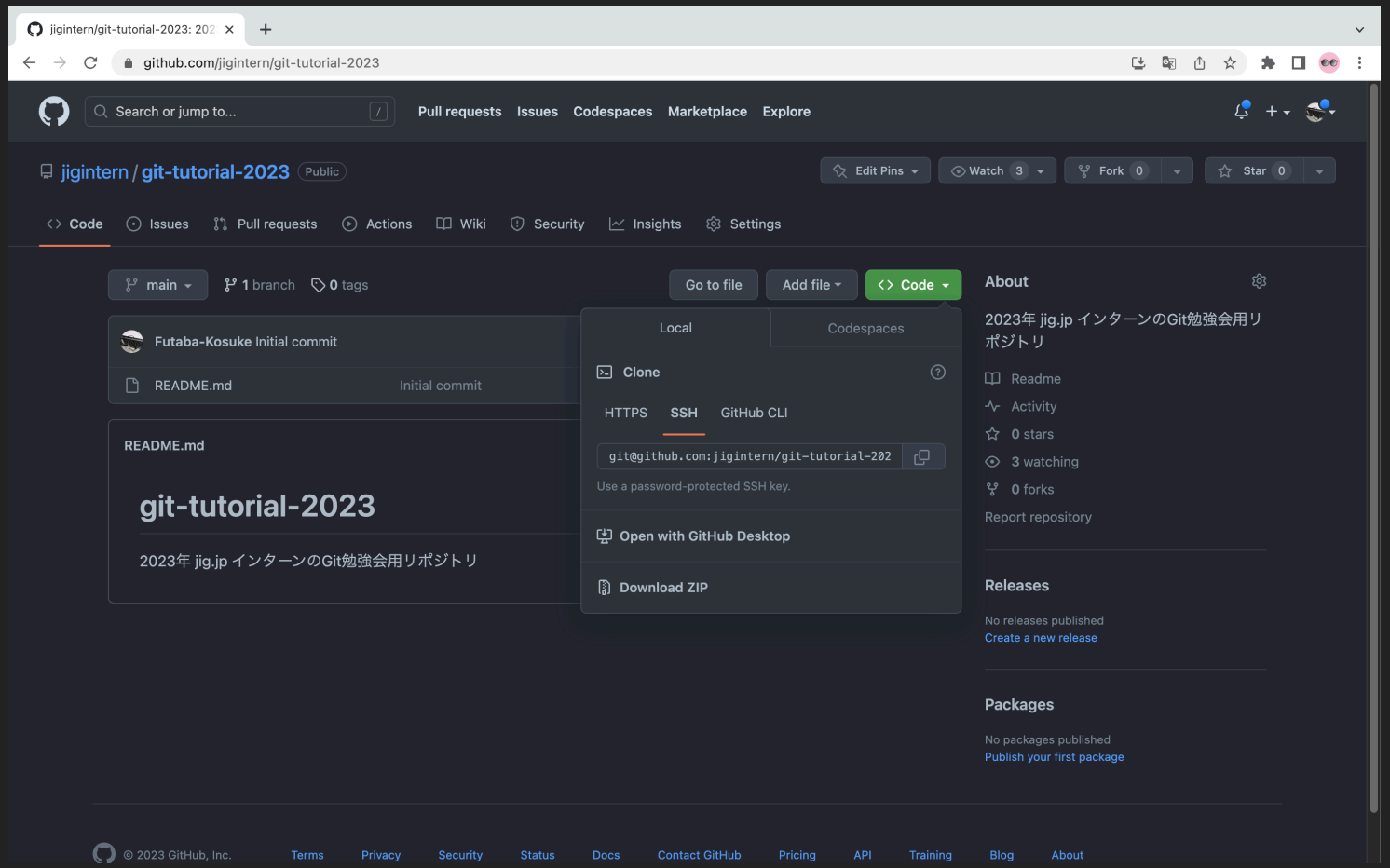
今回はこちらで事前にリポジトリを作成しました

ブラウザで以下URLを開いてください

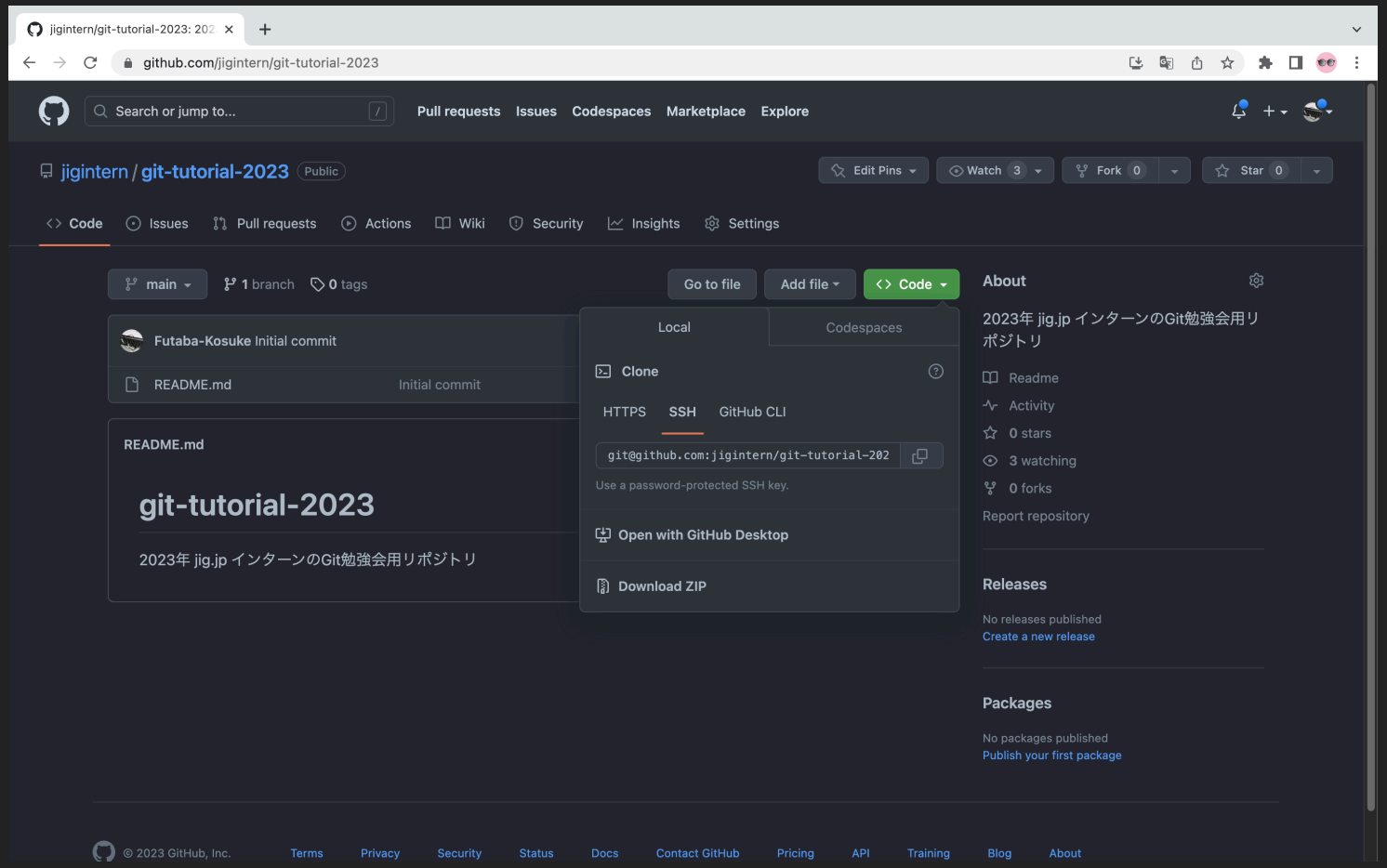
Zoomのチャット欄にも記載します

https://github.com/jigintern/git-tutorial-2025

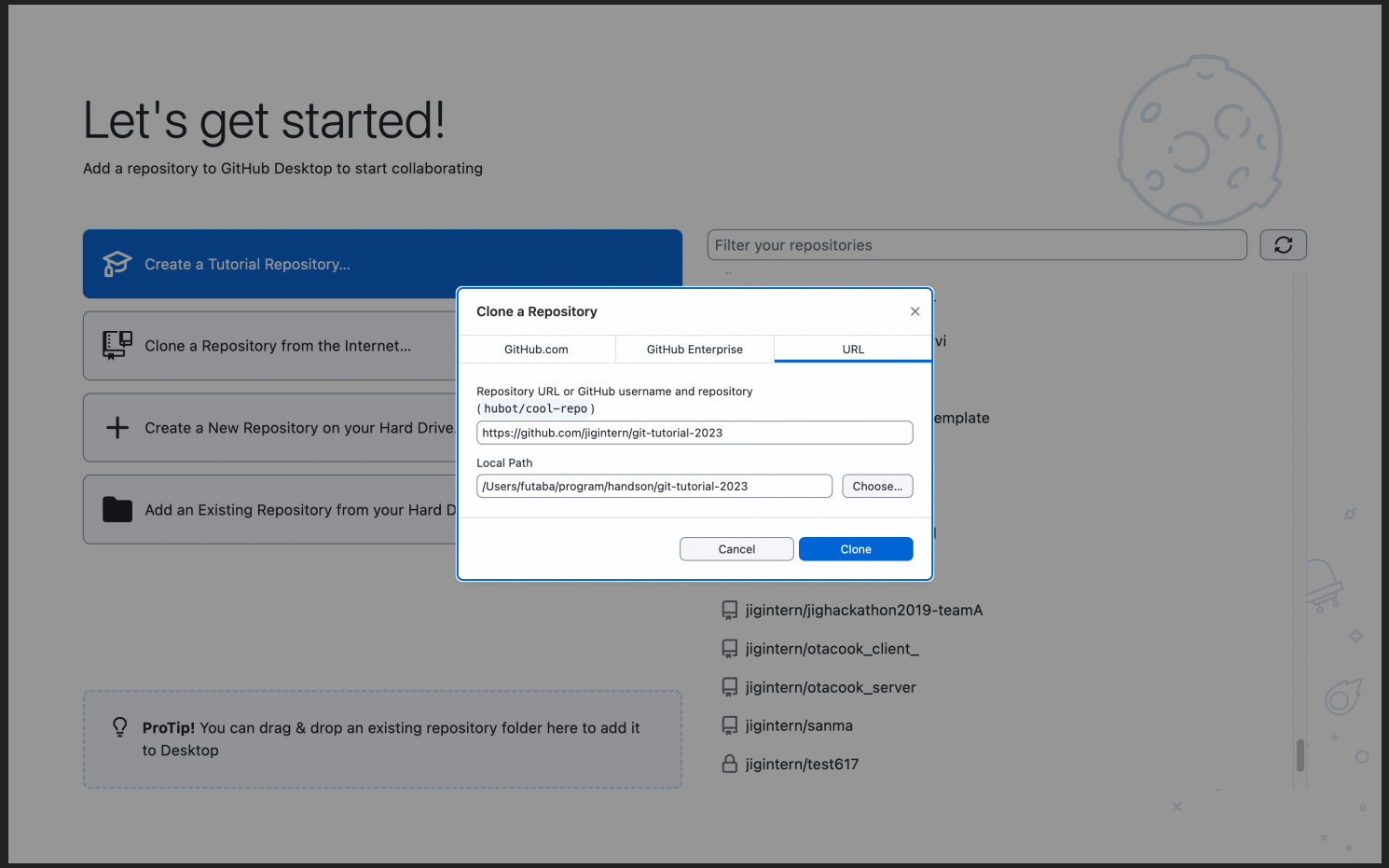
1. 「<> Code」のボタンをクリックします



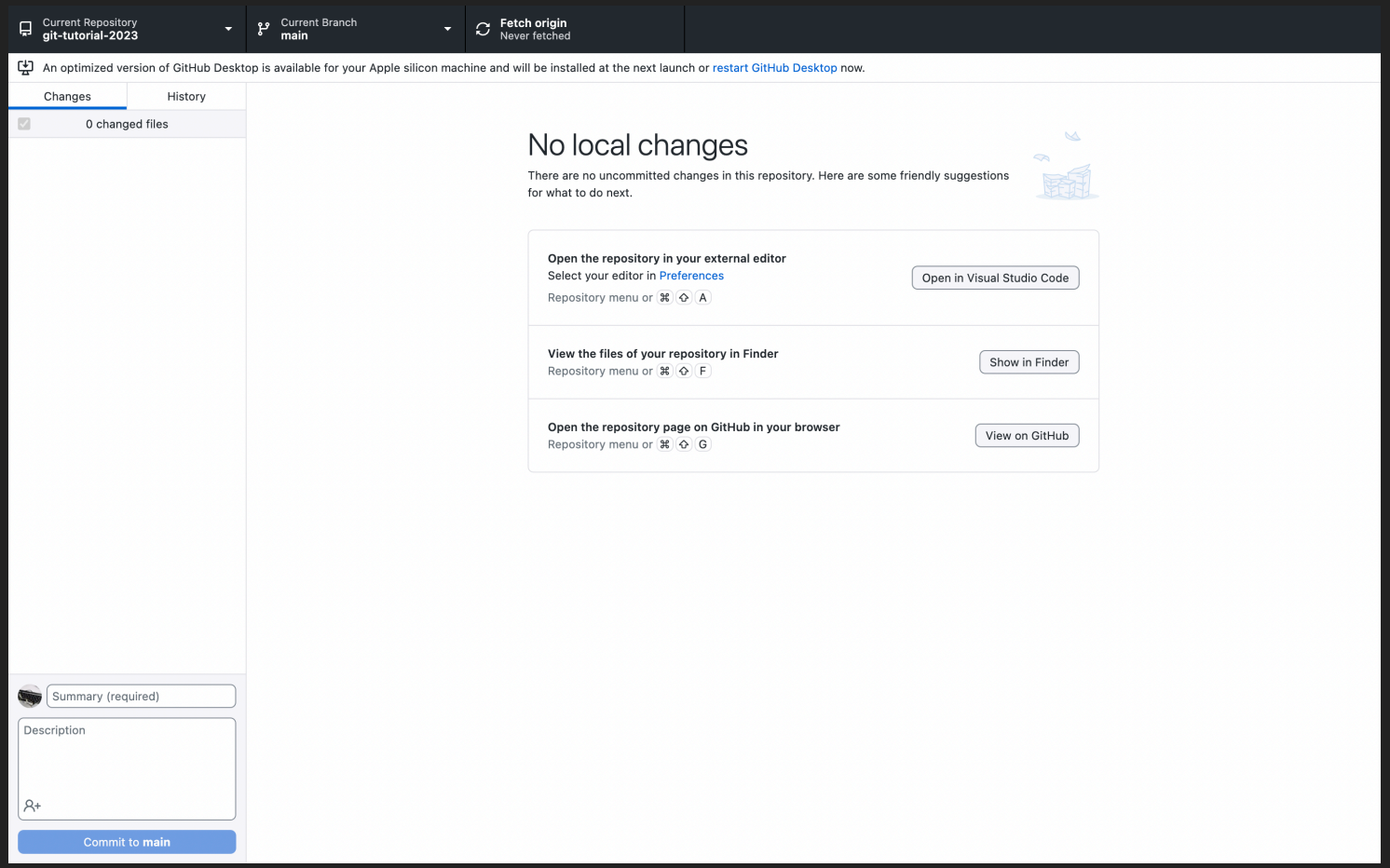
2. 表示されたウィンドウの「Open with Github Desktop」をクリックして、Github Desktopを開きます



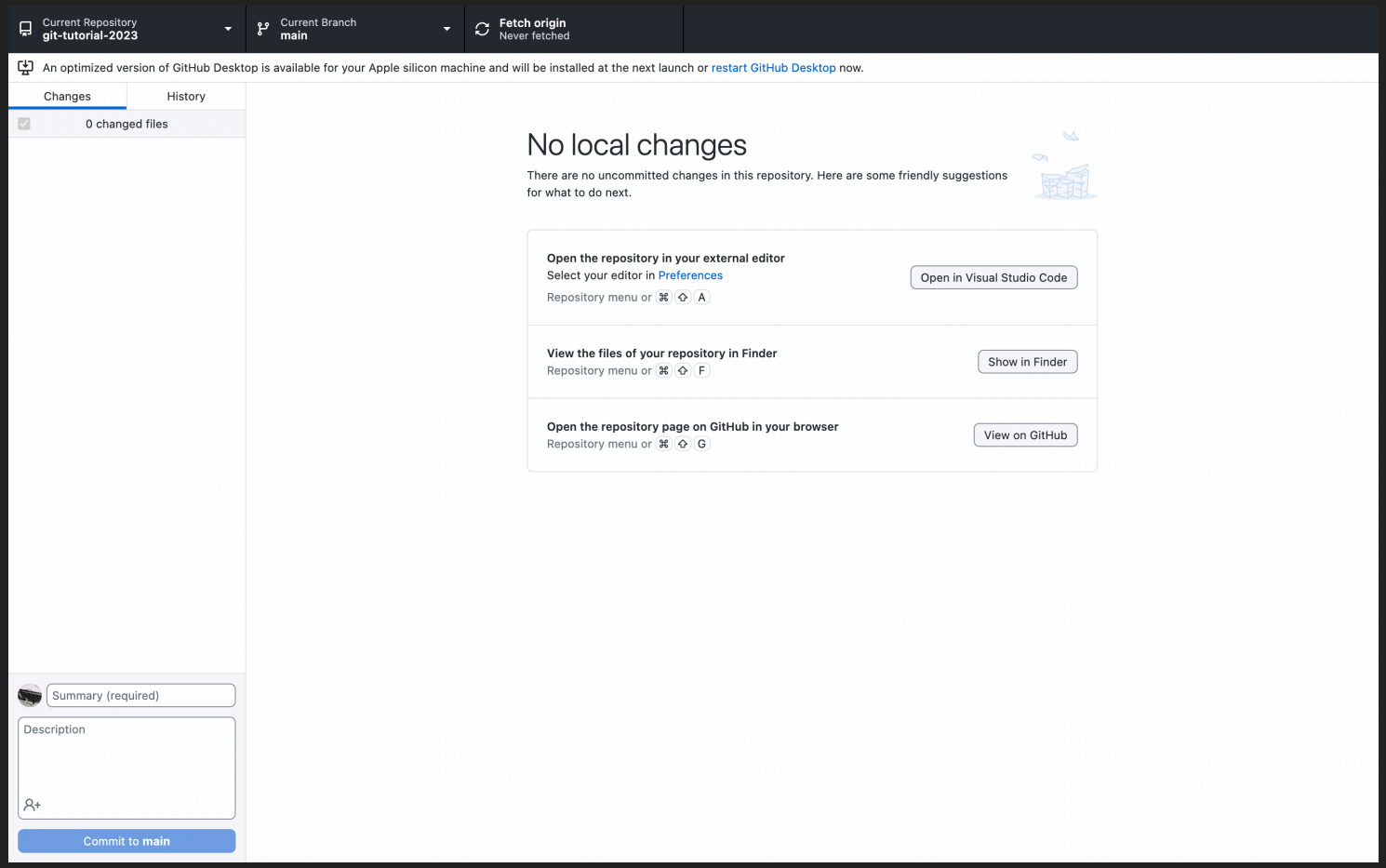
3.「Local Path」にPC上での保存先を設定し、「Clone」をクリックします

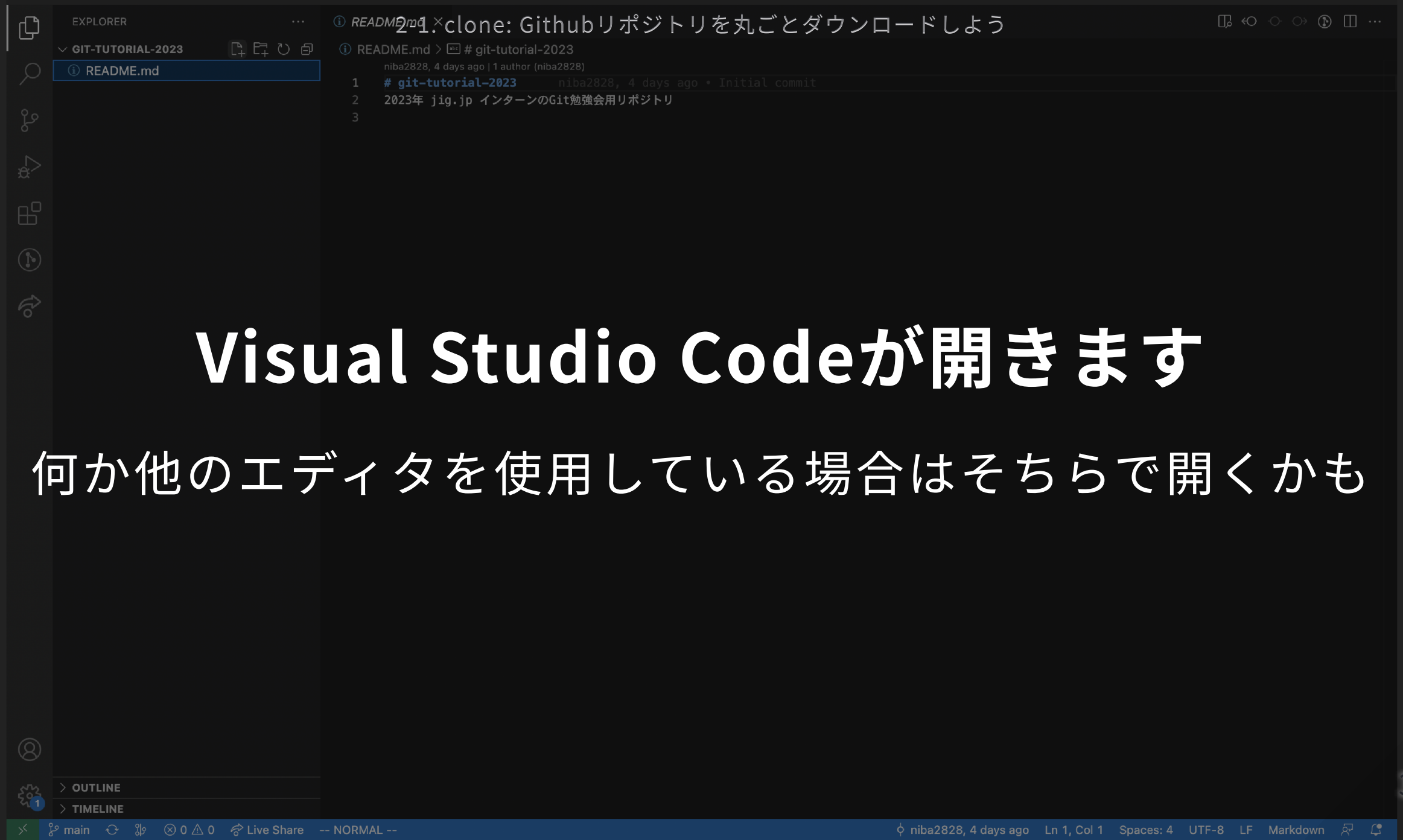


4. リポジトリがクローンできたことを確認します



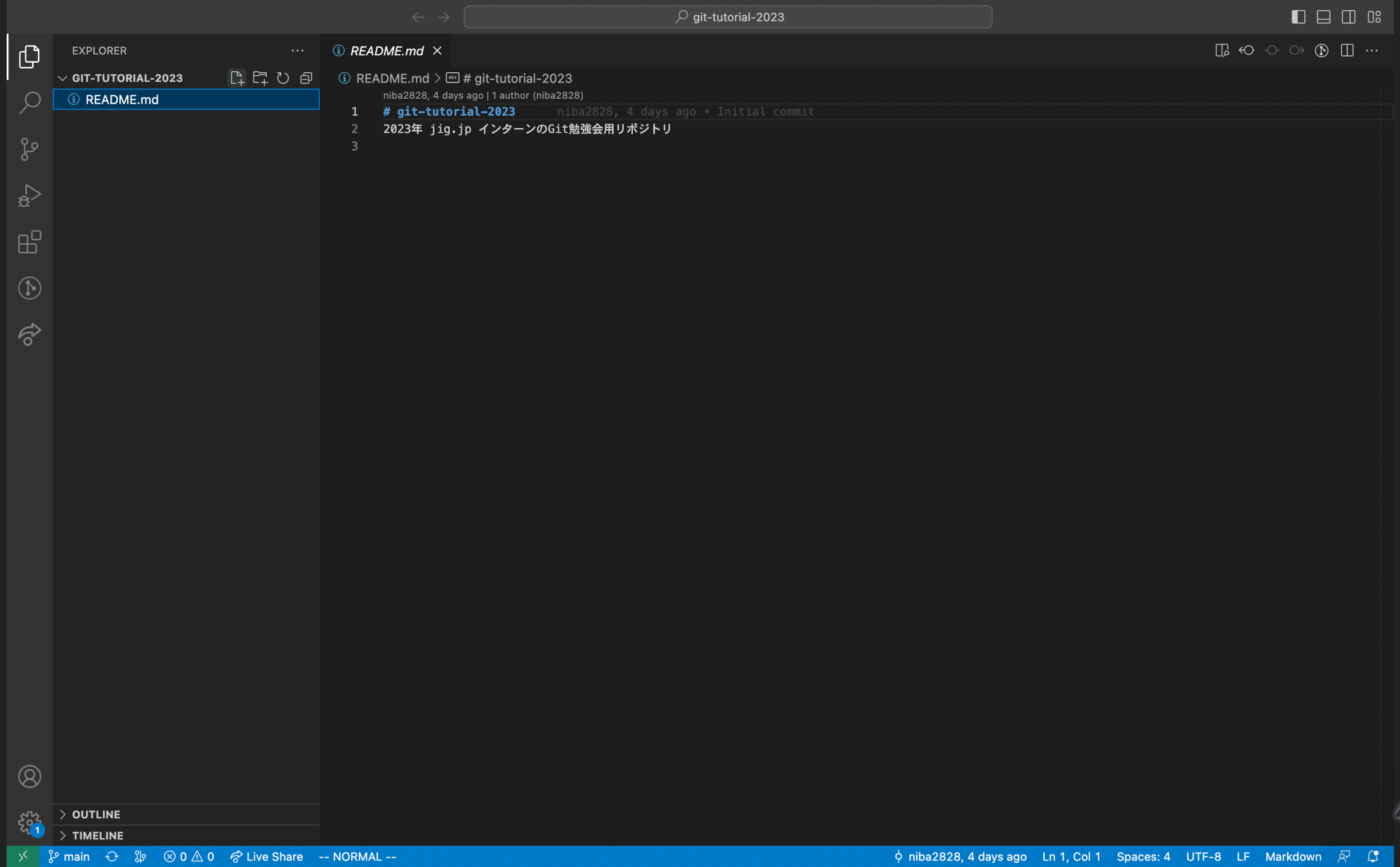
5.「Open in Visual Studio Code」をクリックして、リポジトリをVisual Studio Codeで開きます

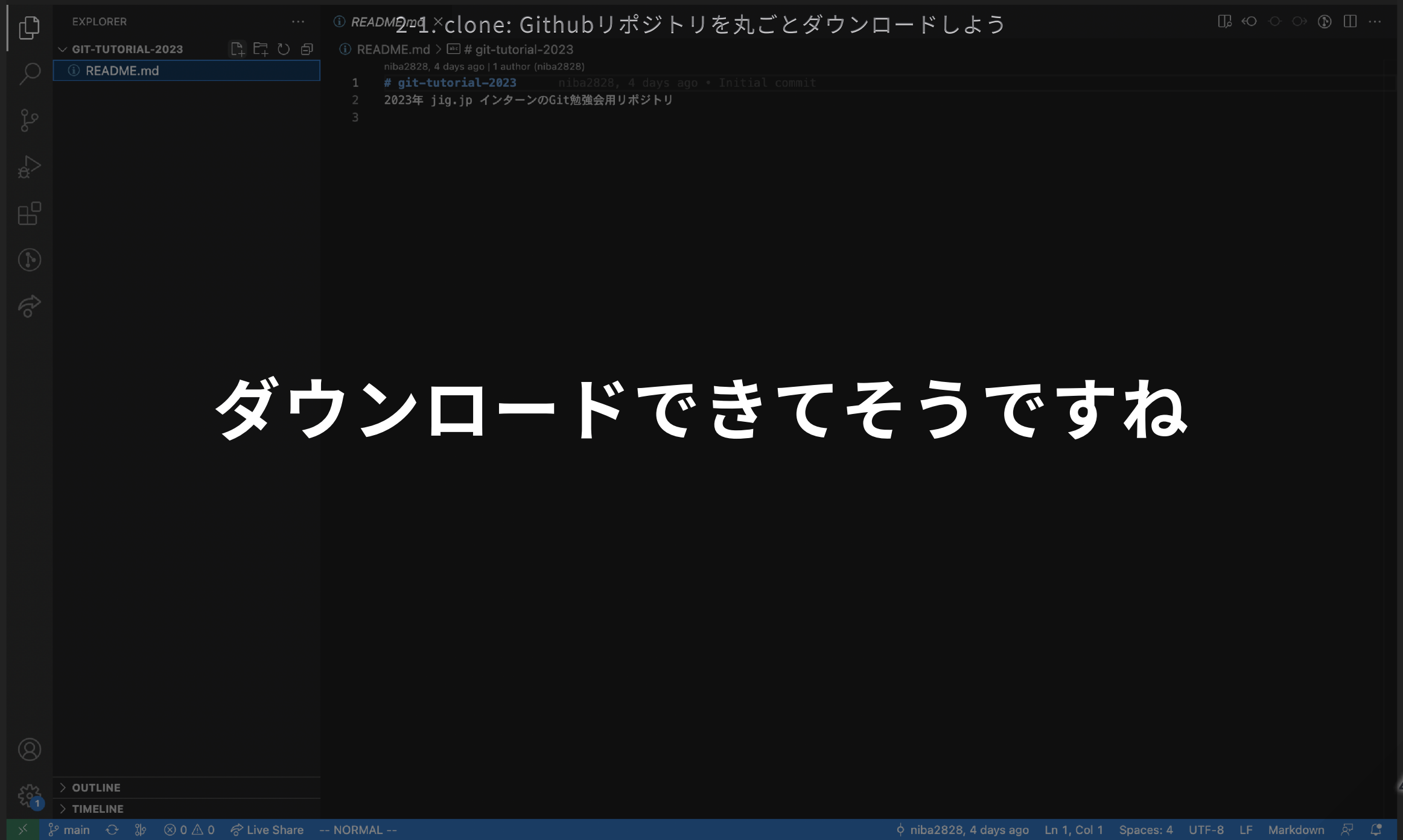




Visual Studio Codeが開きます

何か他のエディタを使用している場合はそちらで開くかも


git-tutorial-2023
EXPLORER
GIT-TUTORIAL-2023
README.md
README.md
git-tutorial-2023
niba2828, 4 days ago | 1 author (niba2828)
1 # git-tutorial-2023 niba2828, 4 days ago • Initial commit
2 2023年 jig.jp インターンのGit勉強会用リポジトリ
3
OUTLINE
TIMELINE
main
0 0
Live Share
-- NORMAL --
niba2828, 4 days ago
Ln 1, Col 1
Spaces: 4
UTF-8
LF
Markdown



ダウンロードできてそうですね

2-2. branch: 作業を枝分かれさせよう

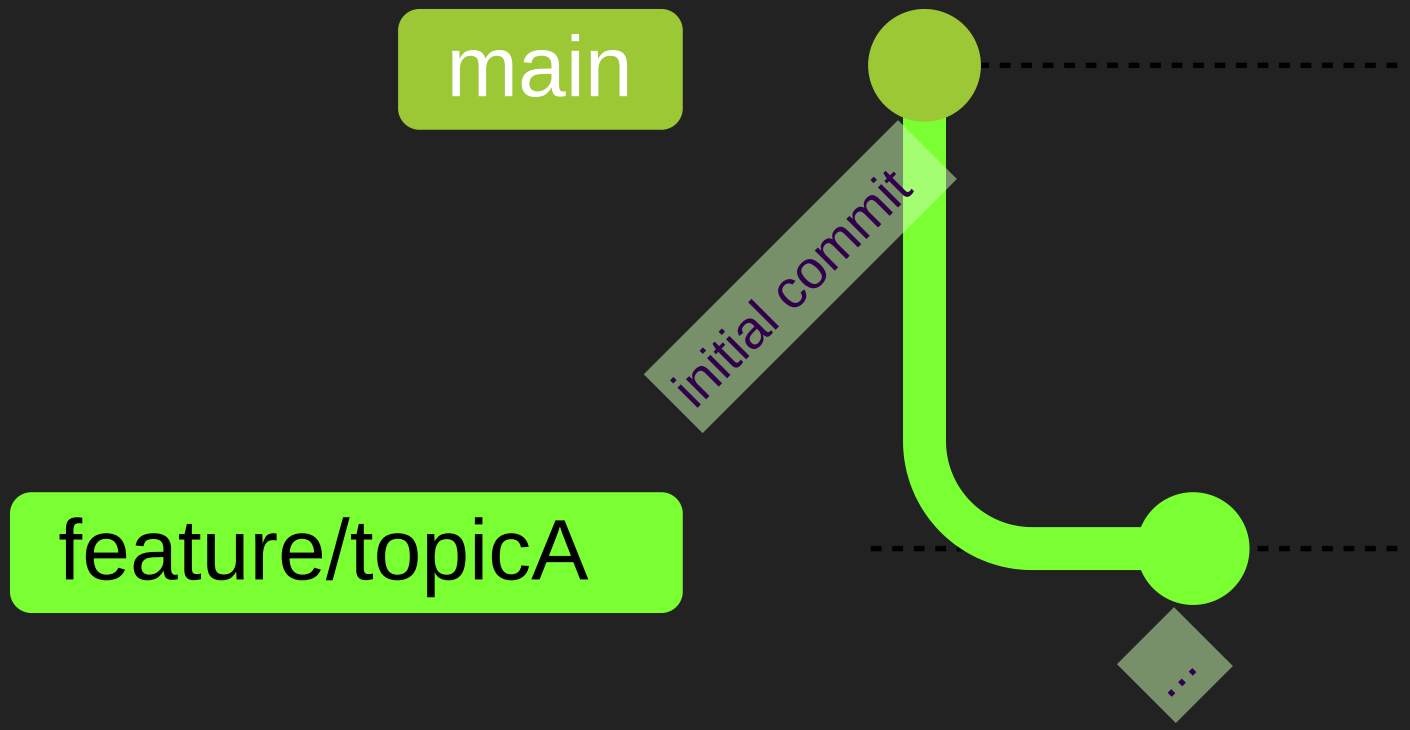
作業を枝分かれさせてみましょう

これをbranch（ブランチ）と呼びます

クラウドサービスでファイル共有すると……

編集が重複したりして、片方の編集内容が消されることも

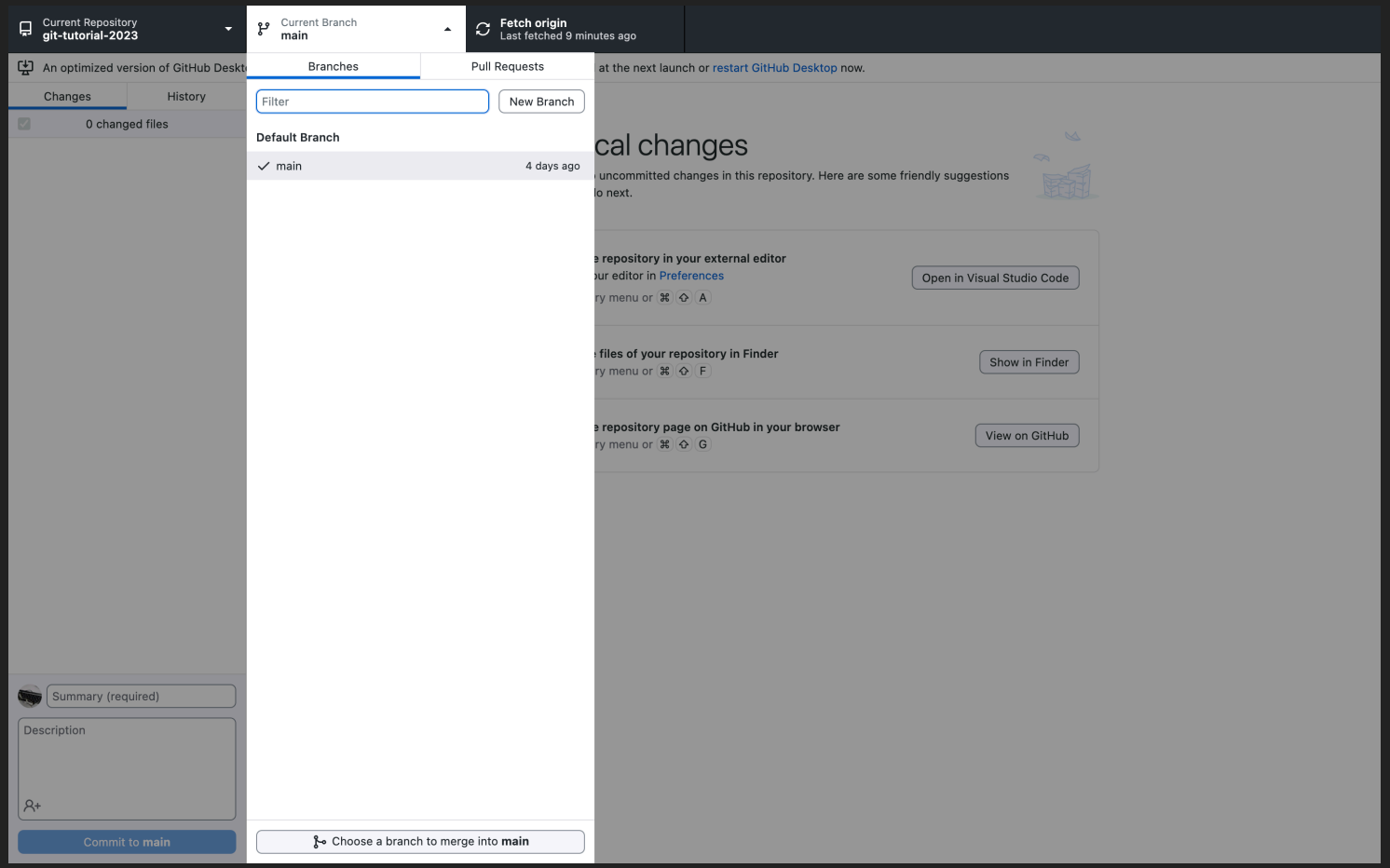
枝分かれすると、複数人で同じファイルを触っても（ちょっと）安心です



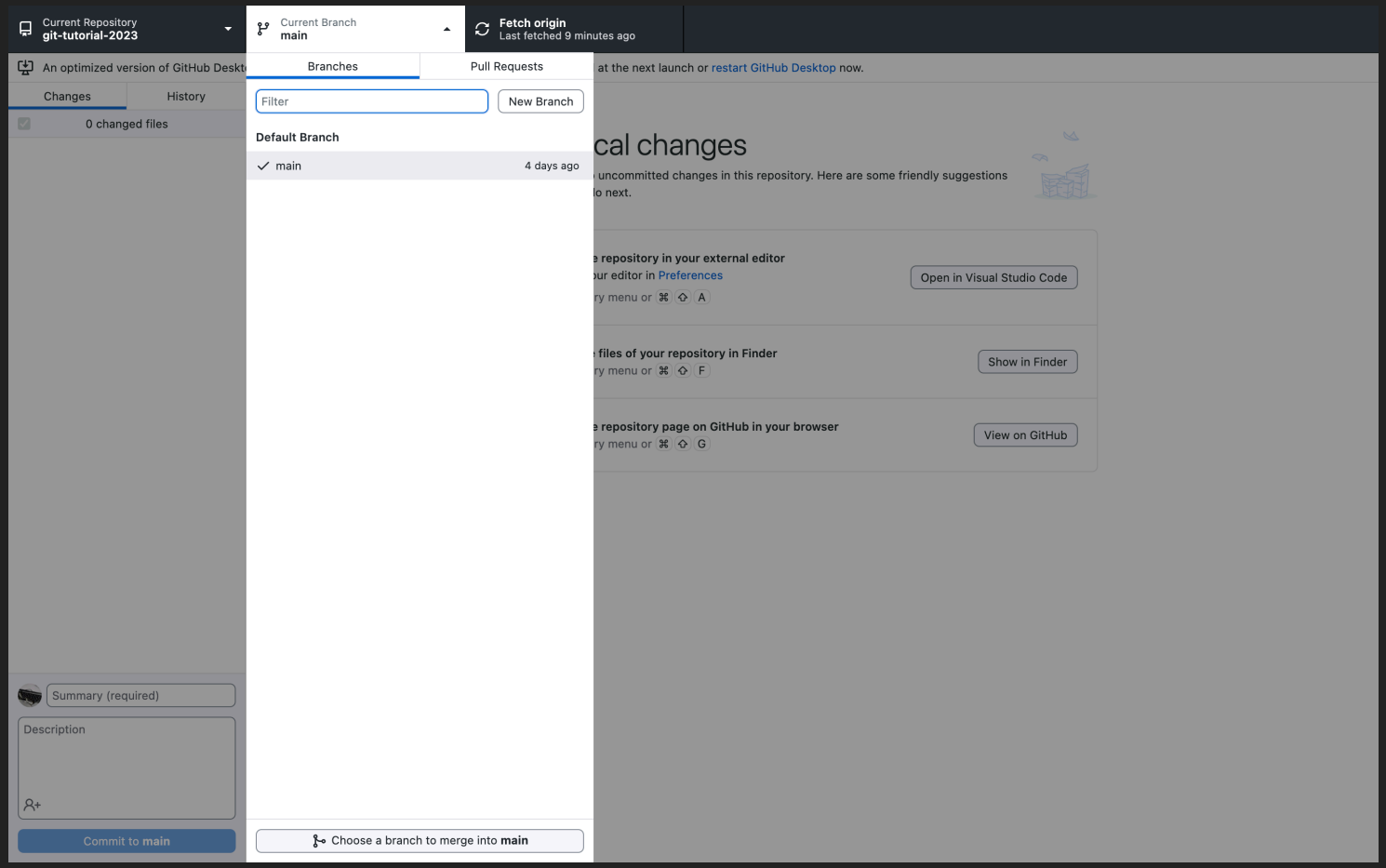
では、ブランチを切ってみましょう

ブランチを作ることを、慣習的に「ブランチを切る」、といいます

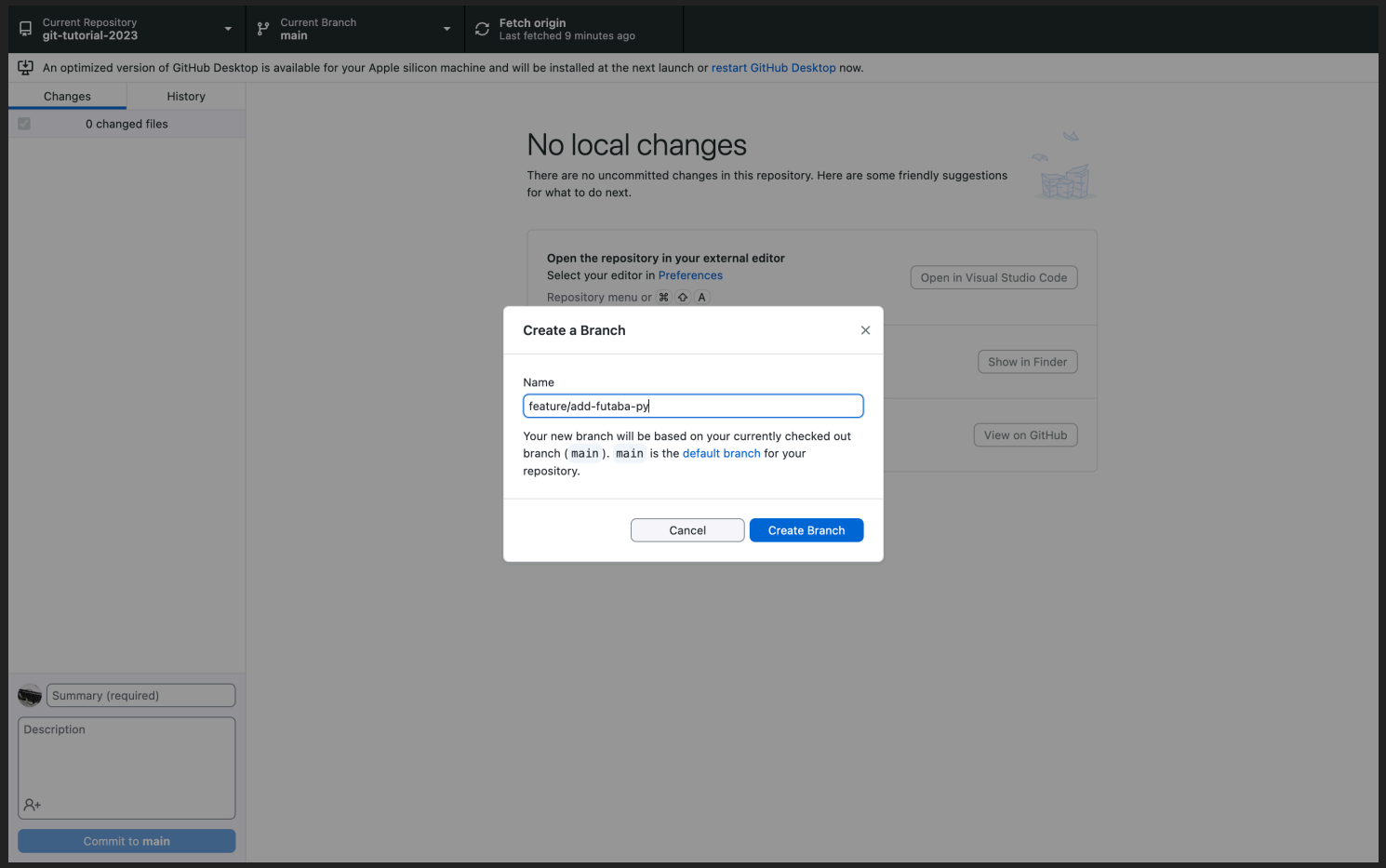
1. ブランチ操作用のフォームを開きます



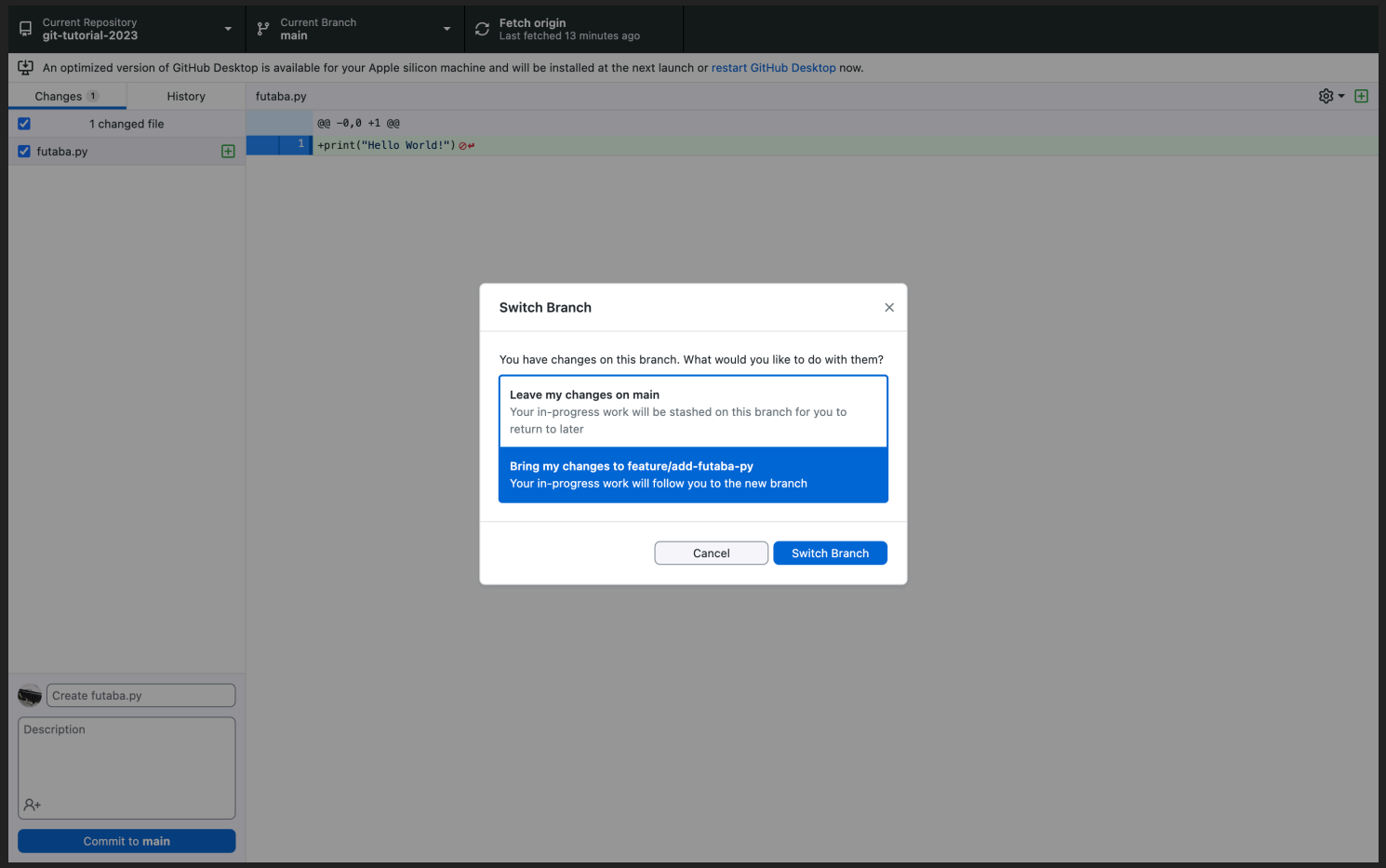
2.「New Branch」をクリックして、新規ブランチの作成ウィンドウを開きます



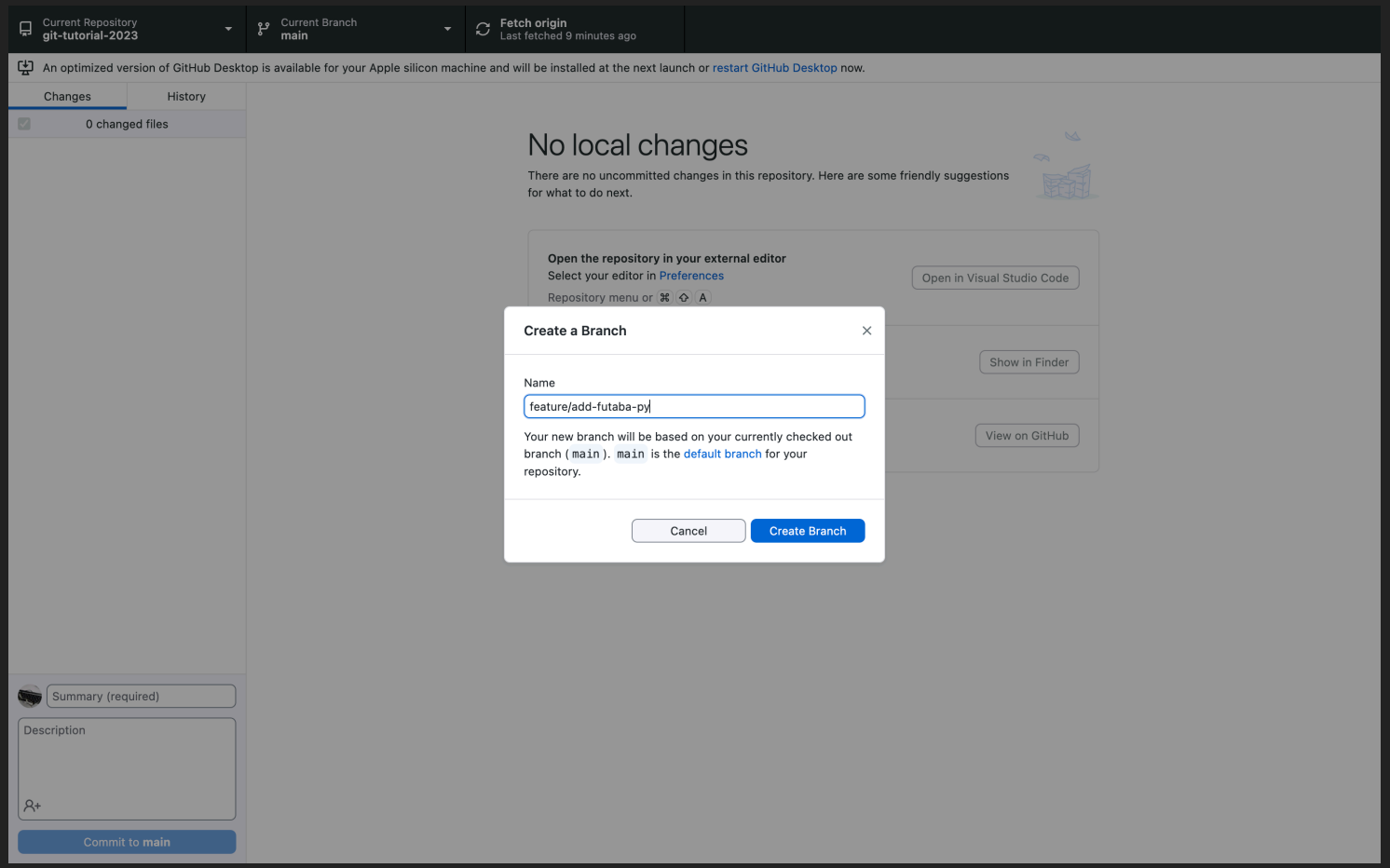
3. 新規ブランチの名前を入力して、「Create Branch」をクリックします。ブランチ名の重複が発生しないように注意してください



ex. 既にブランチに変更を加えている場合は、以下のような画面が表示されるので、「Bring my changes to ...」を選択してください



4. 新しいブランチができるので、作業に取り掛かることができます。



2-3. commit: 作業前と後の差分を記録しよう

作業前と後の差分を記録してみましょう

これをcommit（コミット）と呼びます

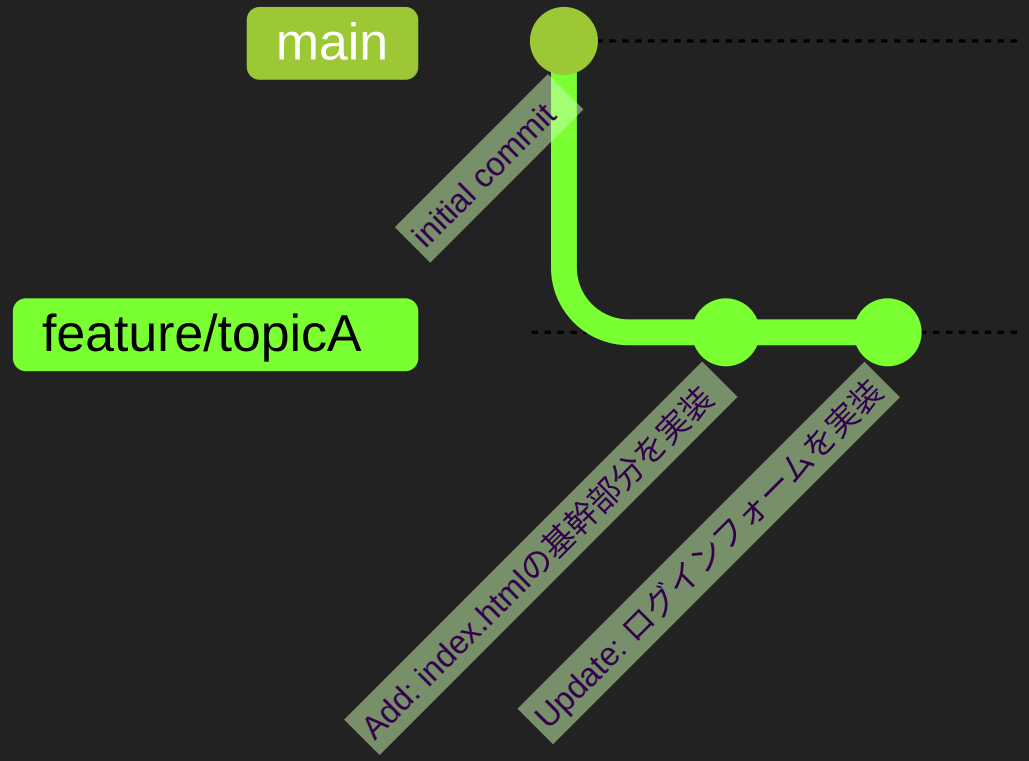
Gitはファイルの現在の内容を保存することを繰り返してログを作ると言いましたが……

実は、全てを保存するわけではありません

作業を重ねる程にデータが膨大になってしまいます

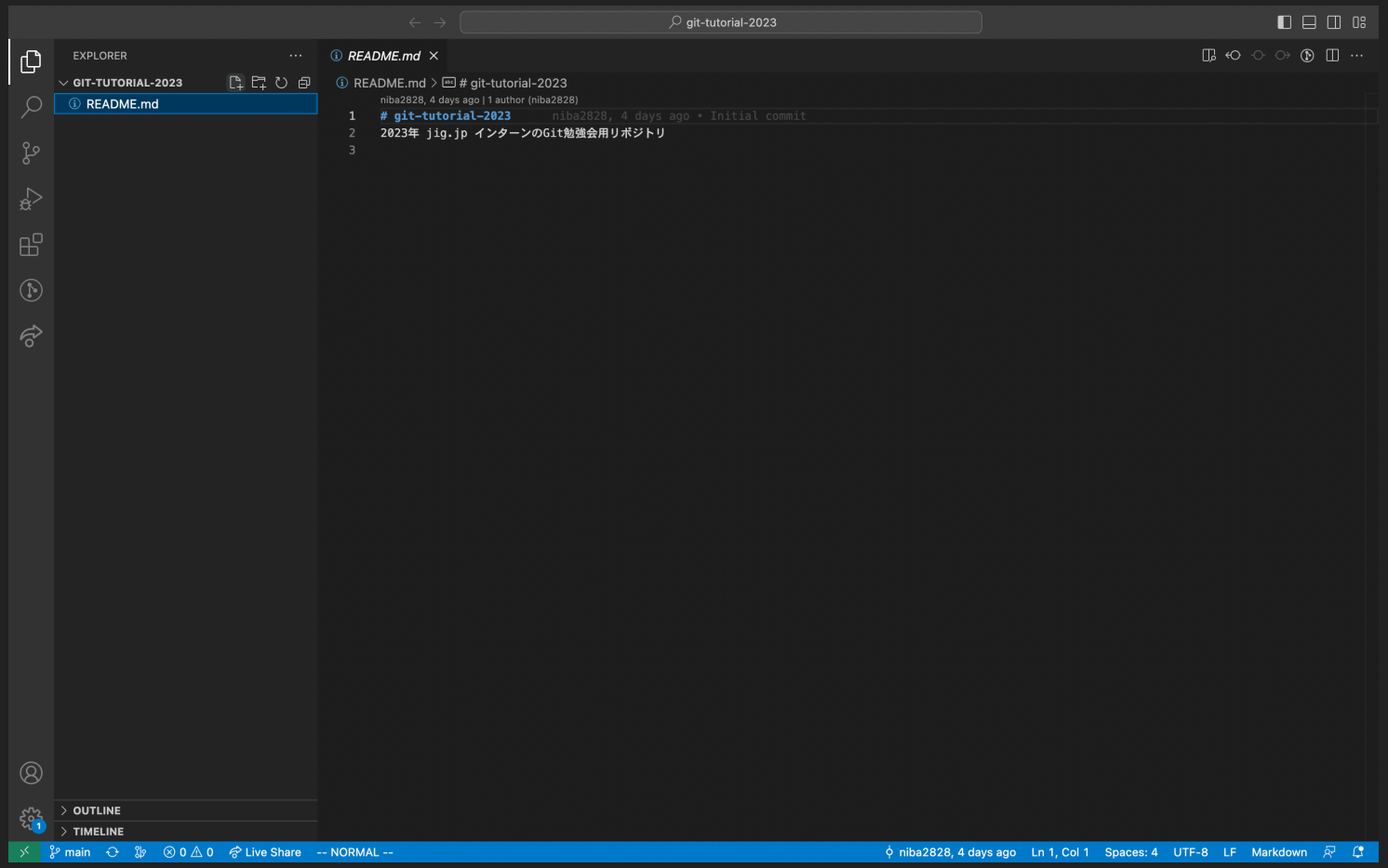
これでは、いけませんね。しかし、解決する方法があります

データの保存時には、作業前と後の**差分を記録**する

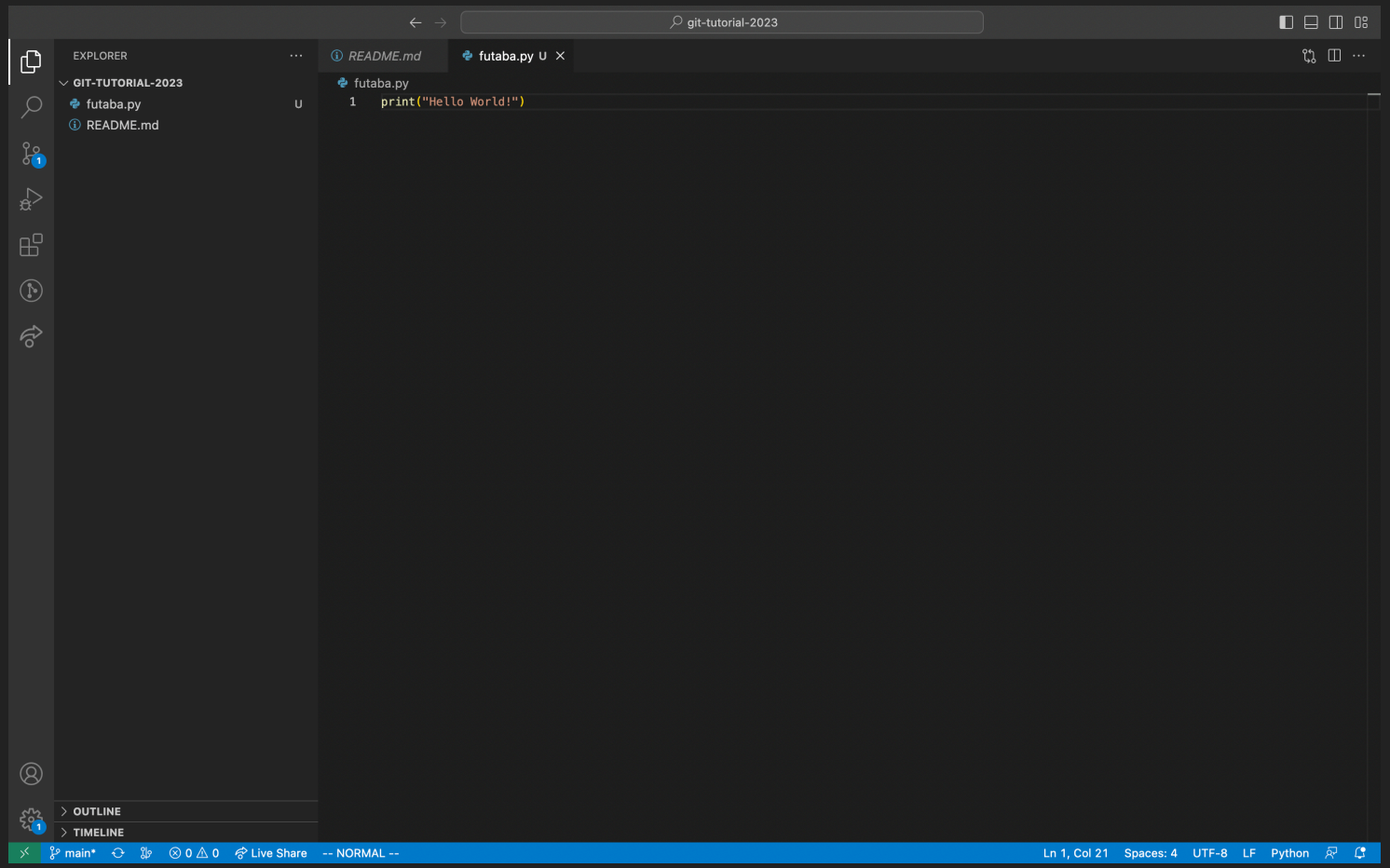


では、コミットしてみましょう

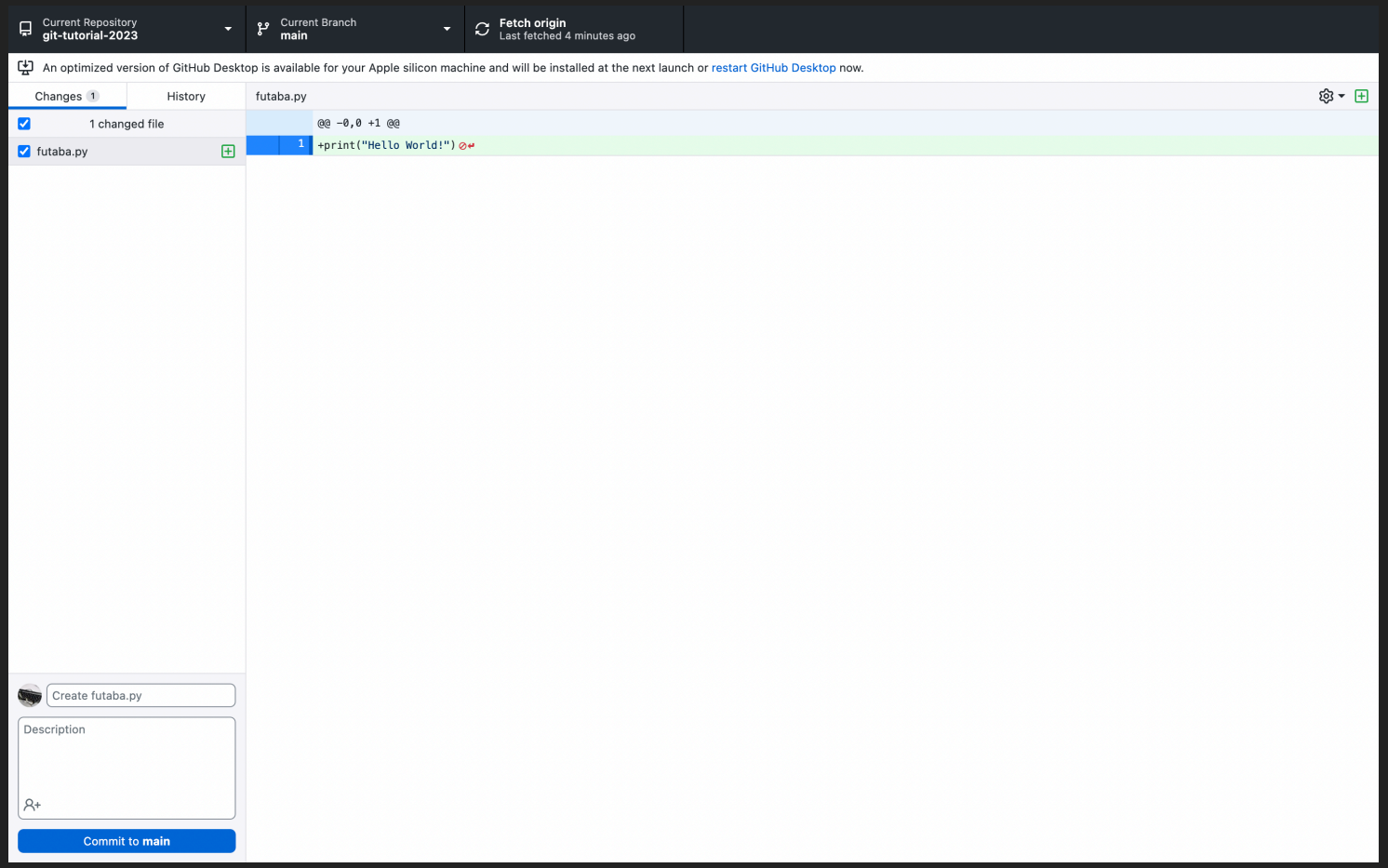
1. Visual Studio Codeを開いて、ファイルを新規作成します。ファイル名が重複しないよう、自分の名前などを半角英字でつけてください



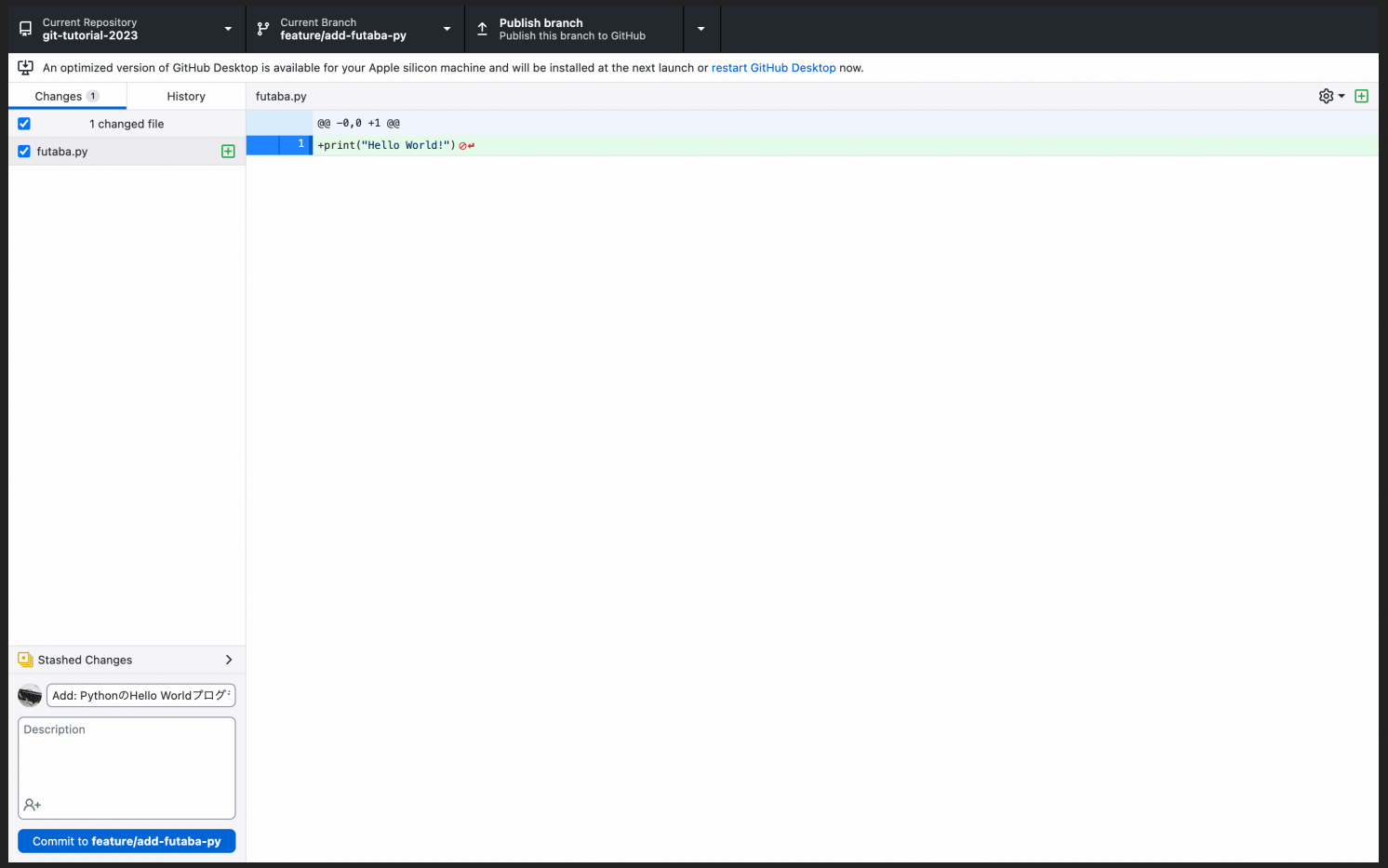
2. 作成したファイルに、適当なプログラムを書き込みます



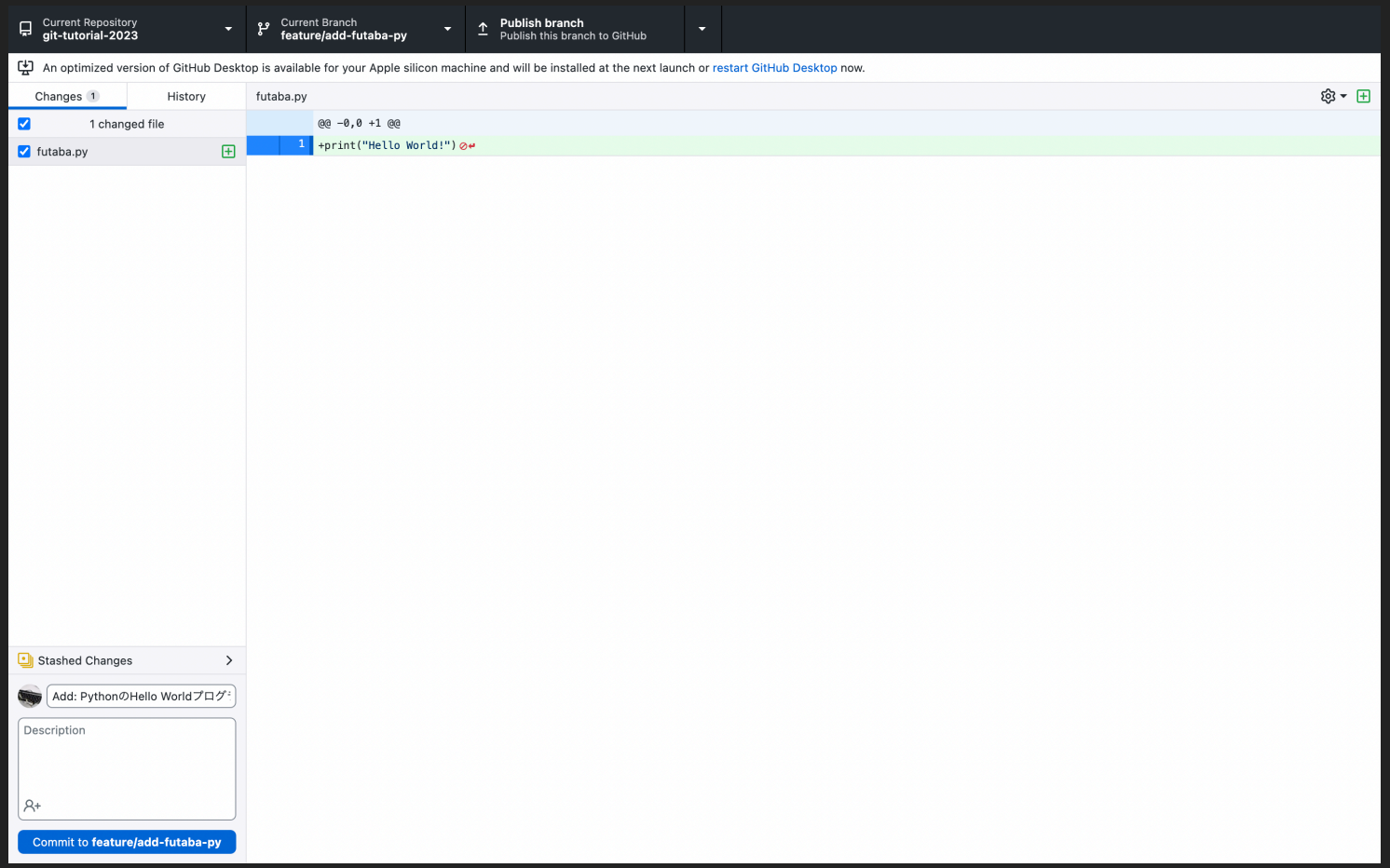
3. Github Desktopを開いて、差分が表示されていること、その差分が正しいことを確認します



4. 画面左下のフォームから、変更内容についての説明文（コミットメッセージ）を記載します



5. 「Commit to ...」をクリックします。これで完了です

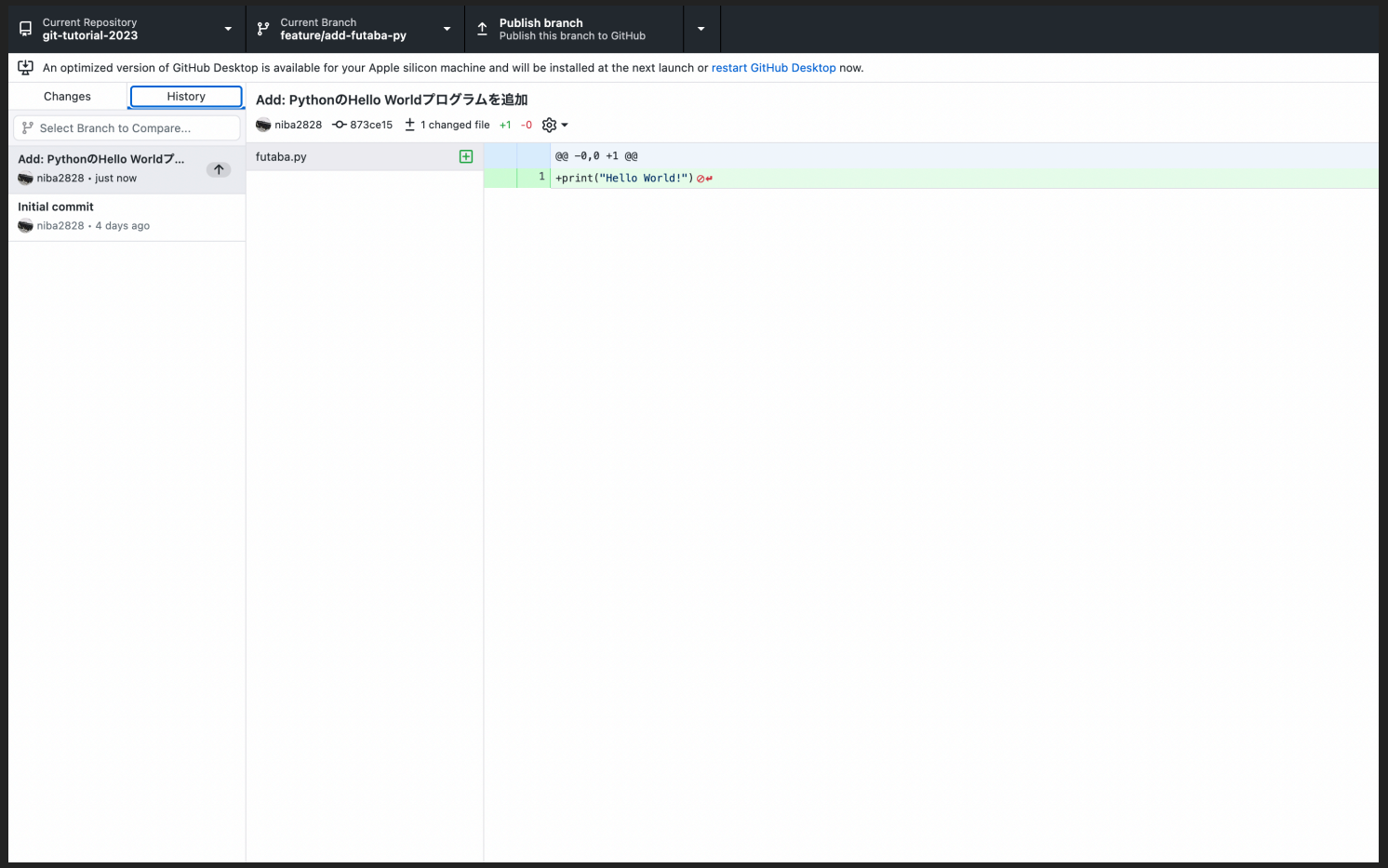


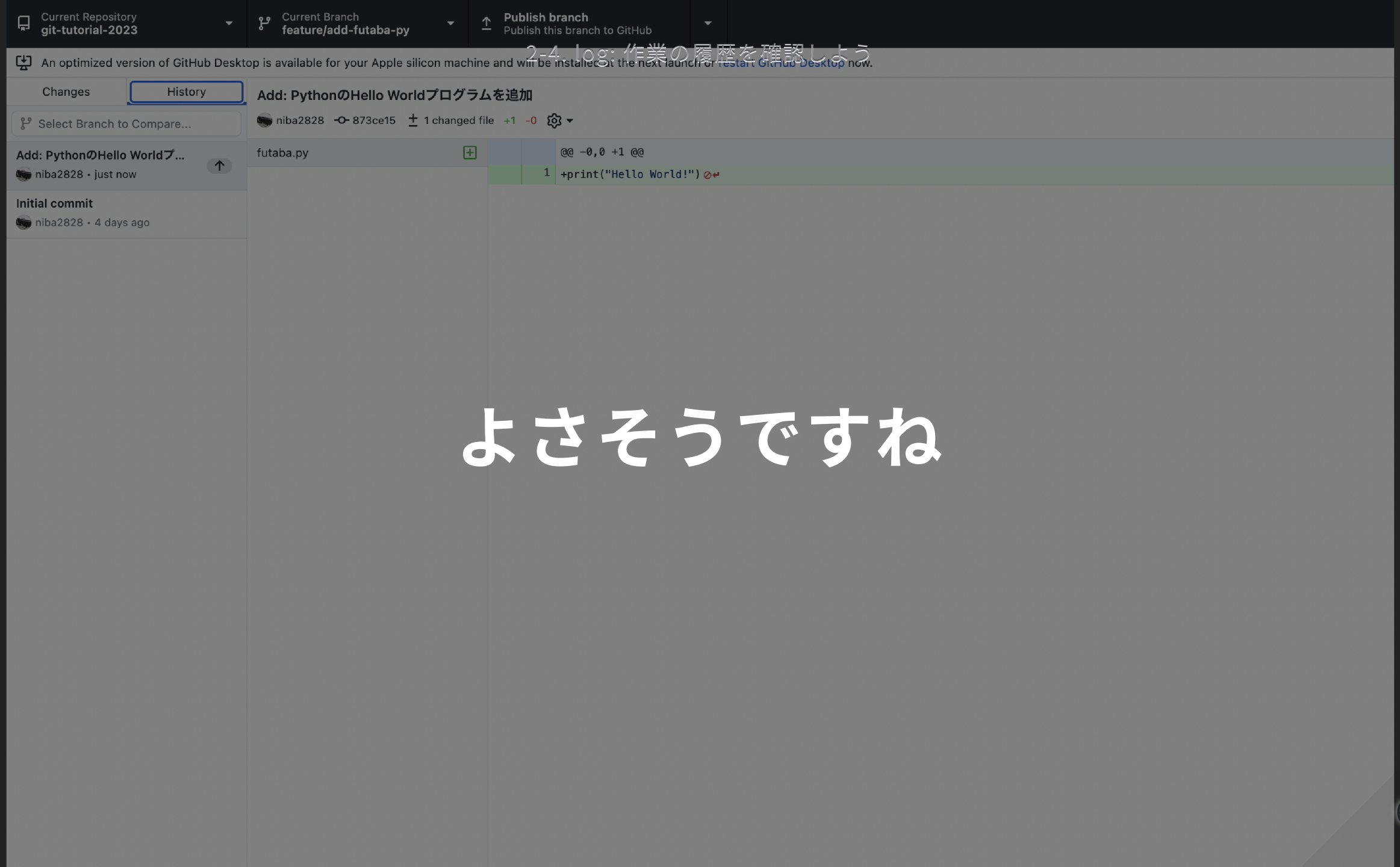
2-4. log: 作業の履歴を確認しよう

作業の履歴を確認してみましょう

この履歴をlog（ログ）と呼びます

「History」をクリックして、コミットの履歴が表示されることを確認します




Current Repository
git-tutorial-2023
Current Branch
feature/add-futaba-py
Publish branch
Publish this branch to GitHub
An optimized version of GitHub Desktop is available for your Apple silicon machine and will be installed at the next launch or Restart GitHub Desktop now.
Changes
History
Select Branch to Compare...
Add: PythonのHello Worldプ...
niba2828 • just now
Initial commit
niba2828 • 4 days ago
Add: PythonのHello Worldプログラムを追加
niba2828 873ce15 1 changed file +1 -0
futaba.py
@@ -0,0 +1 @@
1 +print("Hello World!")

よさそうですね

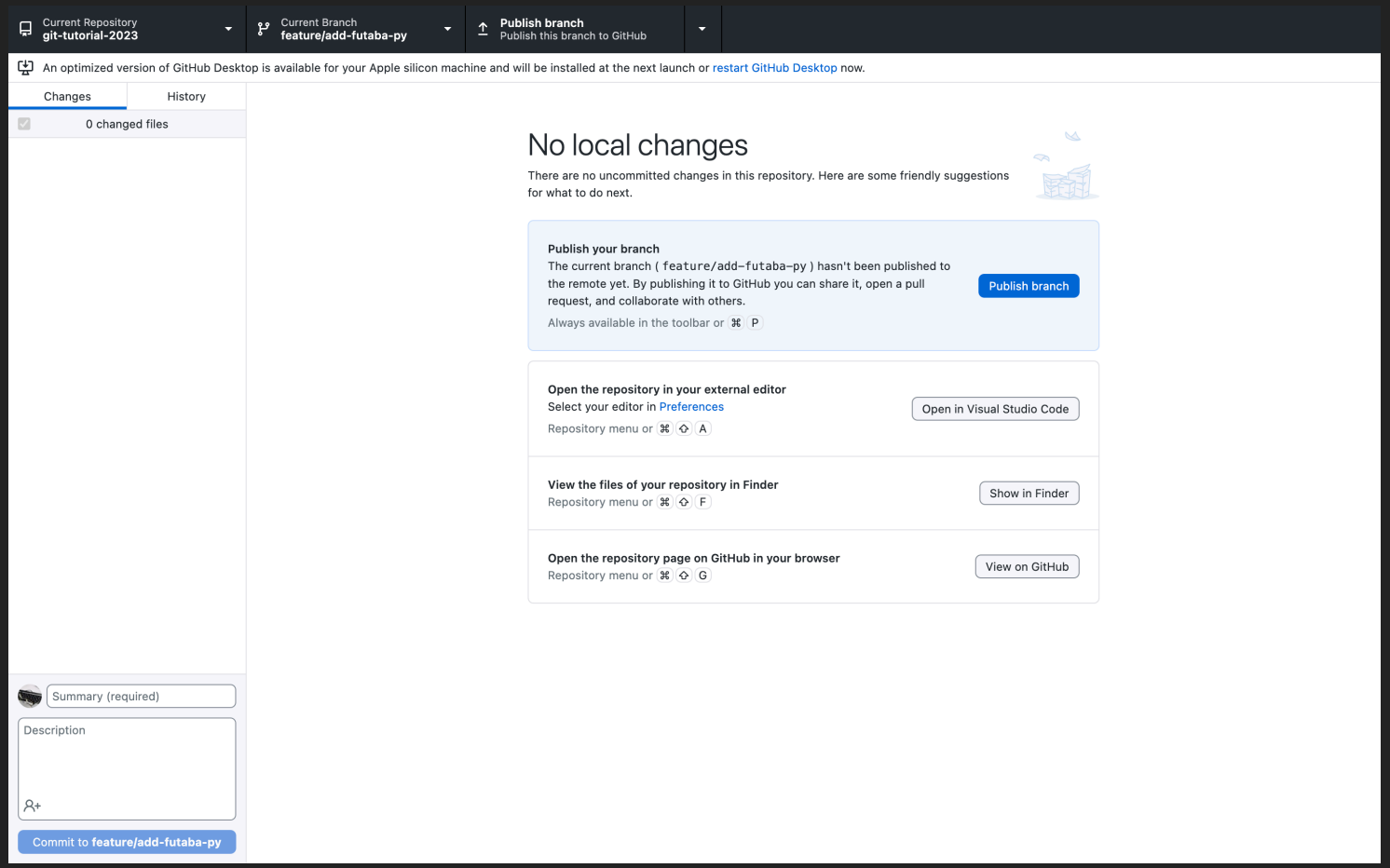
2-5. push: 作業の成果をGithubにアップロードして共有しよう

作業の成果をGithubにアップロードしてみましょう

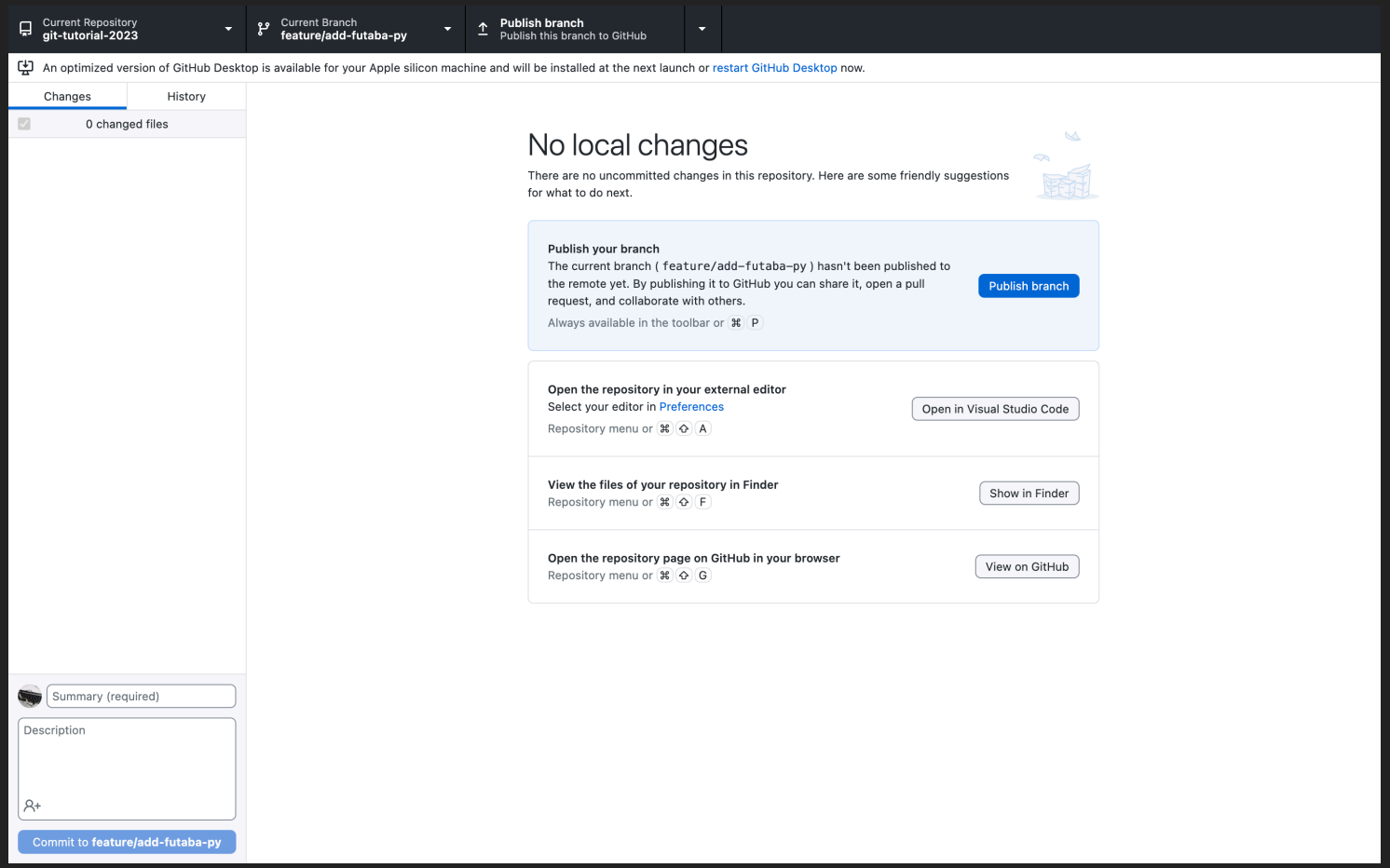
これをpush（プッシュ）と呼びます

早速、プッシュしてみましょう

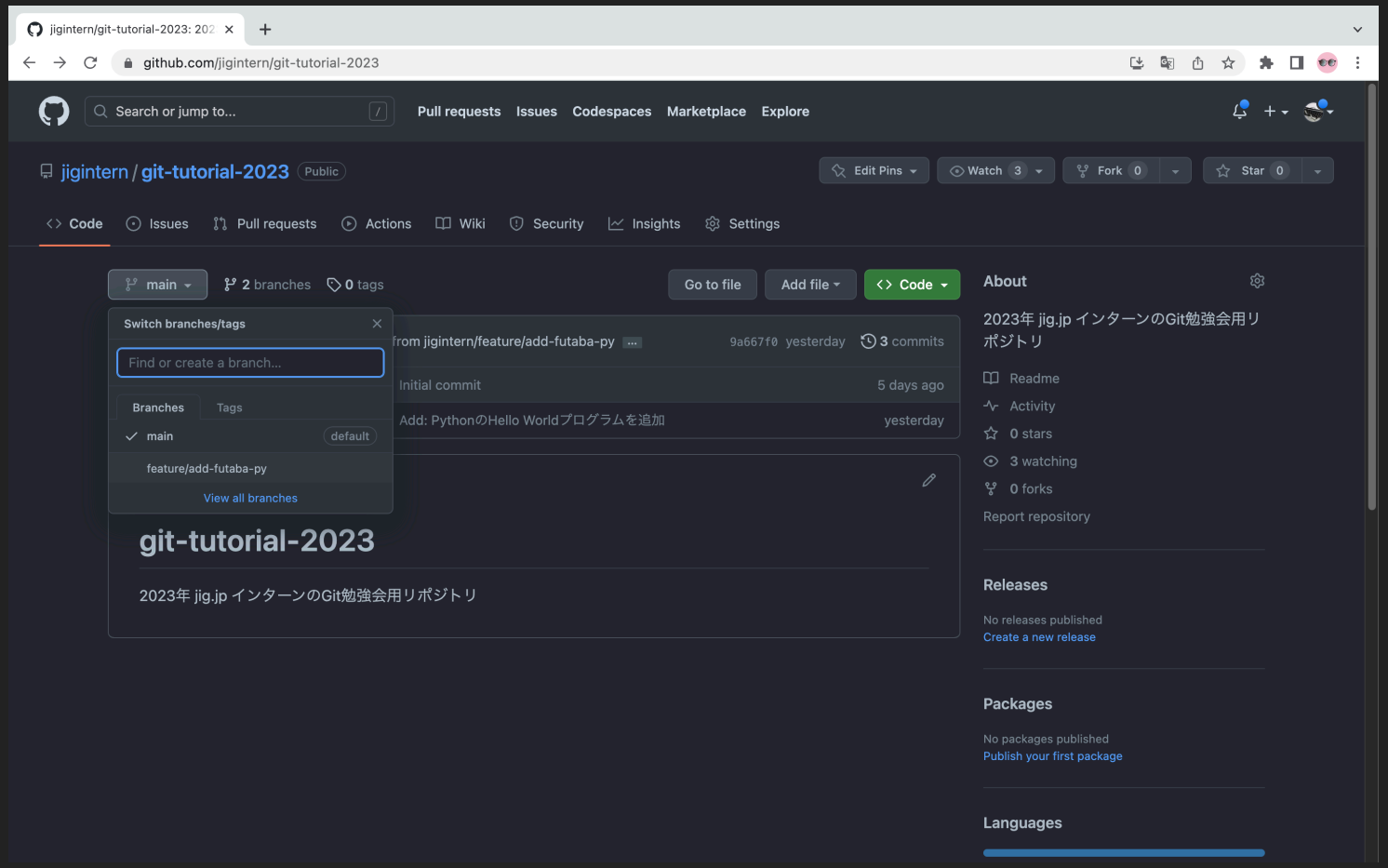
1. 「Changes」をクリックして、元の画面に戻ります



2. 「Publish branch」をクリックして、変更内容をプッシュします



3. ブラウザでGithubを開き、プッシュしたブランチが正しく反映されていることを確認します



2-6. Pull Request: 枝分かれした成果を結合しよう

ブランチ機能で枝分かれさせた作業を結合させましょう

これをmerge（マージ）と呼びます

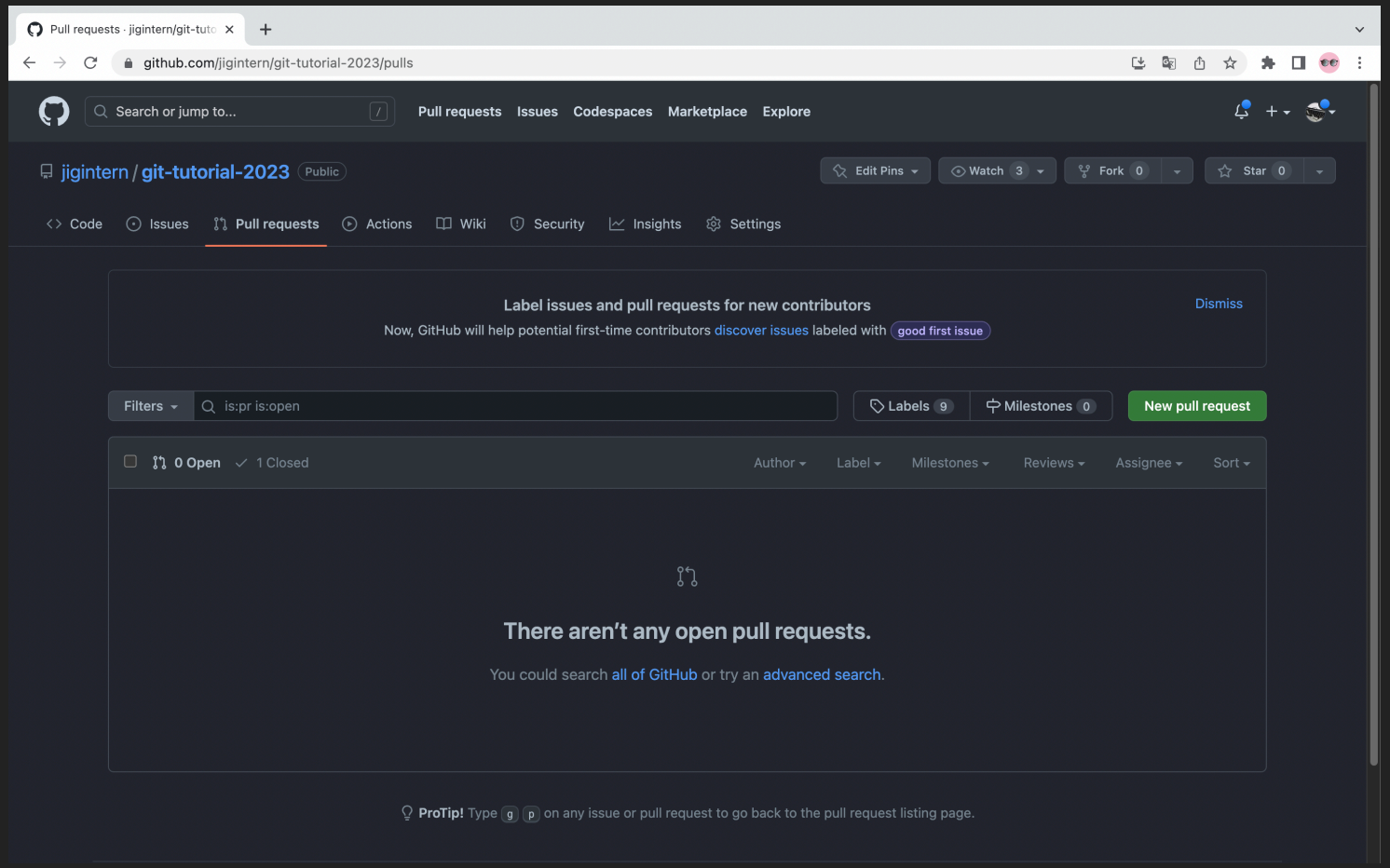
一応、手元でマージもできますが……

マージする時は、バグ等の防止のため、他の人に確認して貰いたいです

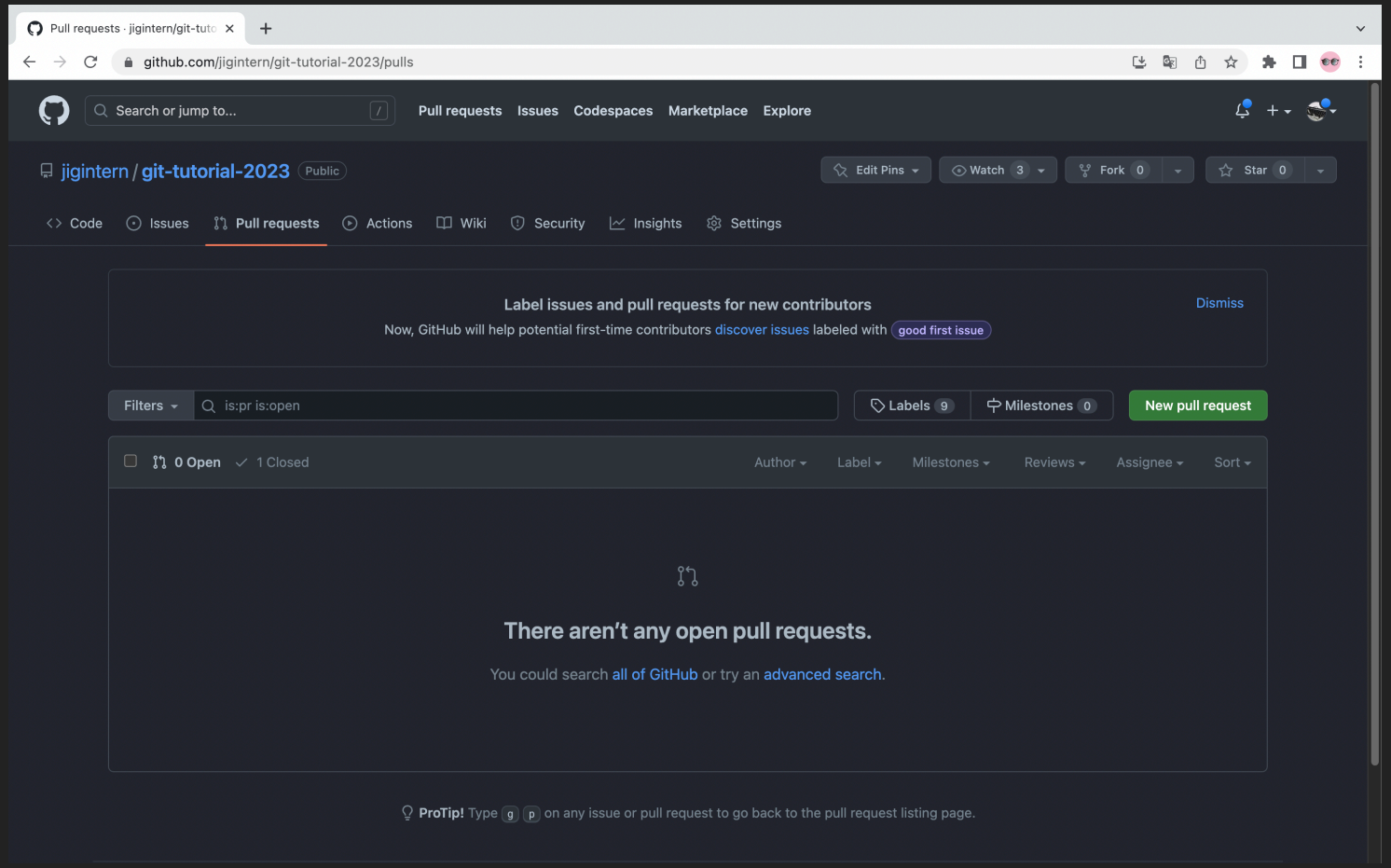
GithubのPull Request（プルリクエスト）機能を使いましょう！

プルリクエストでマージしましょう

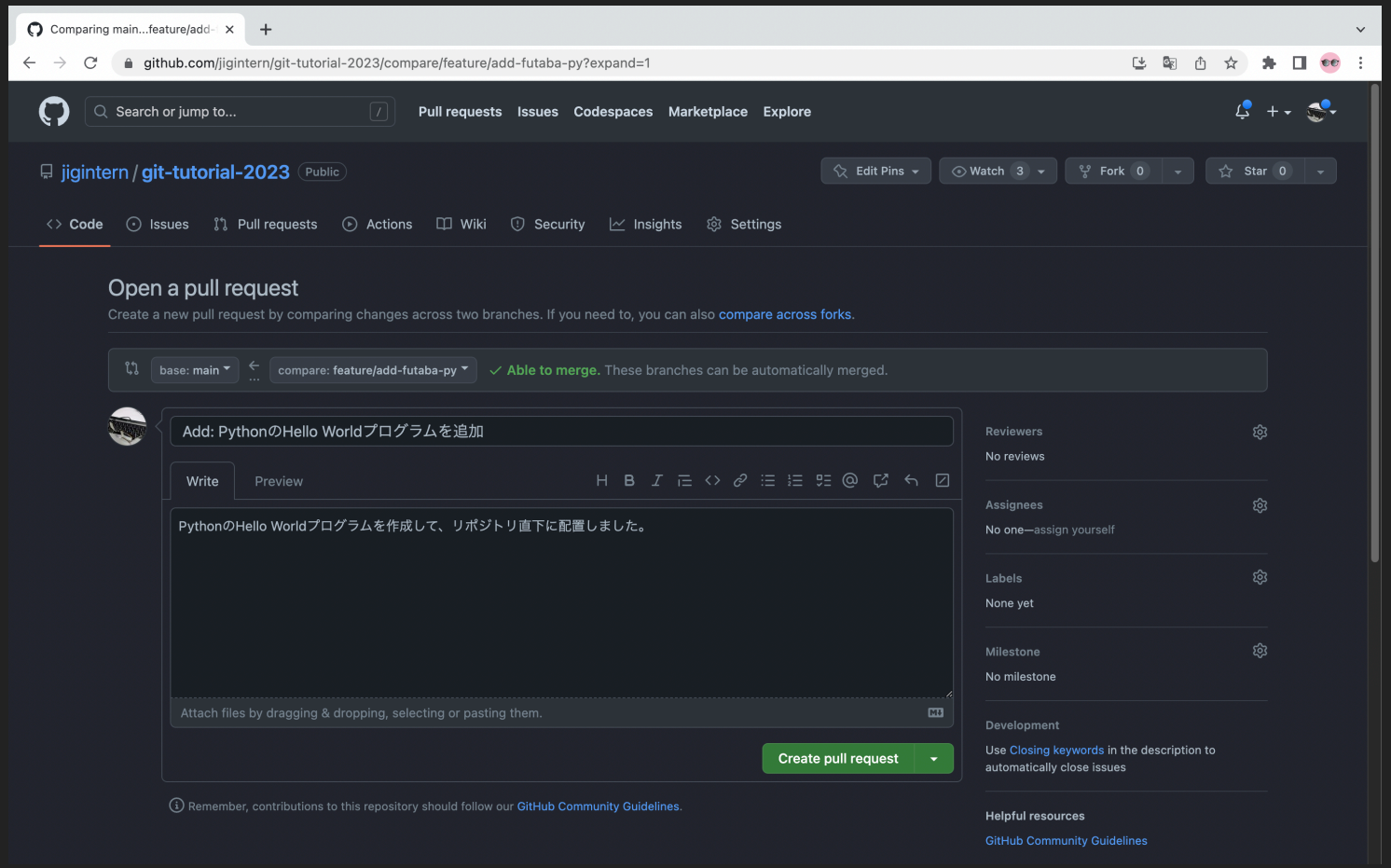
1. Githubで「Pull requests」のタブをクリック



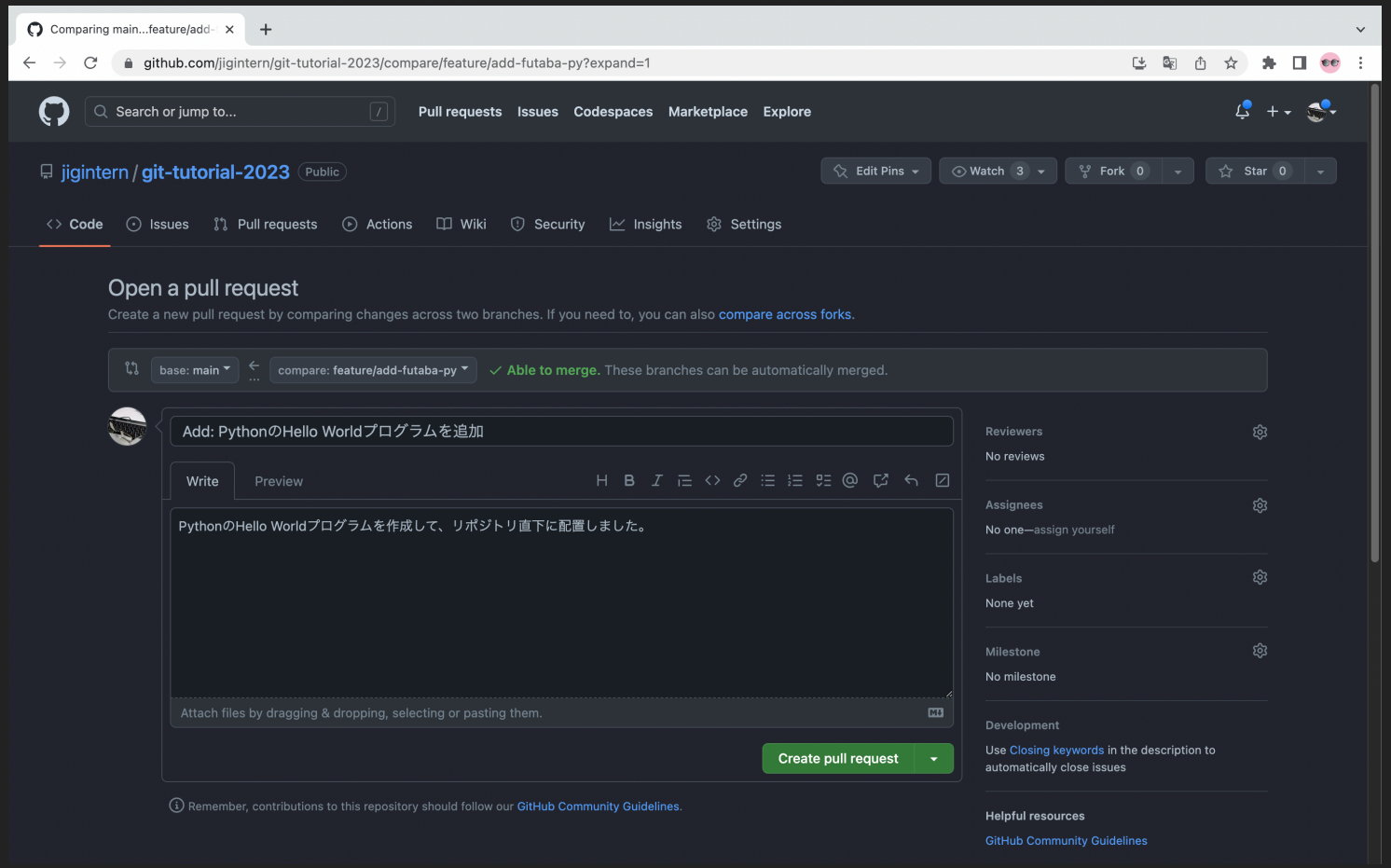
2. 「New pull request」をクリックします



3. 新規ブランチの名前、Pull Requestのタイトル、本文を入力します。作業内容が理解しやすい内容にすると良いです



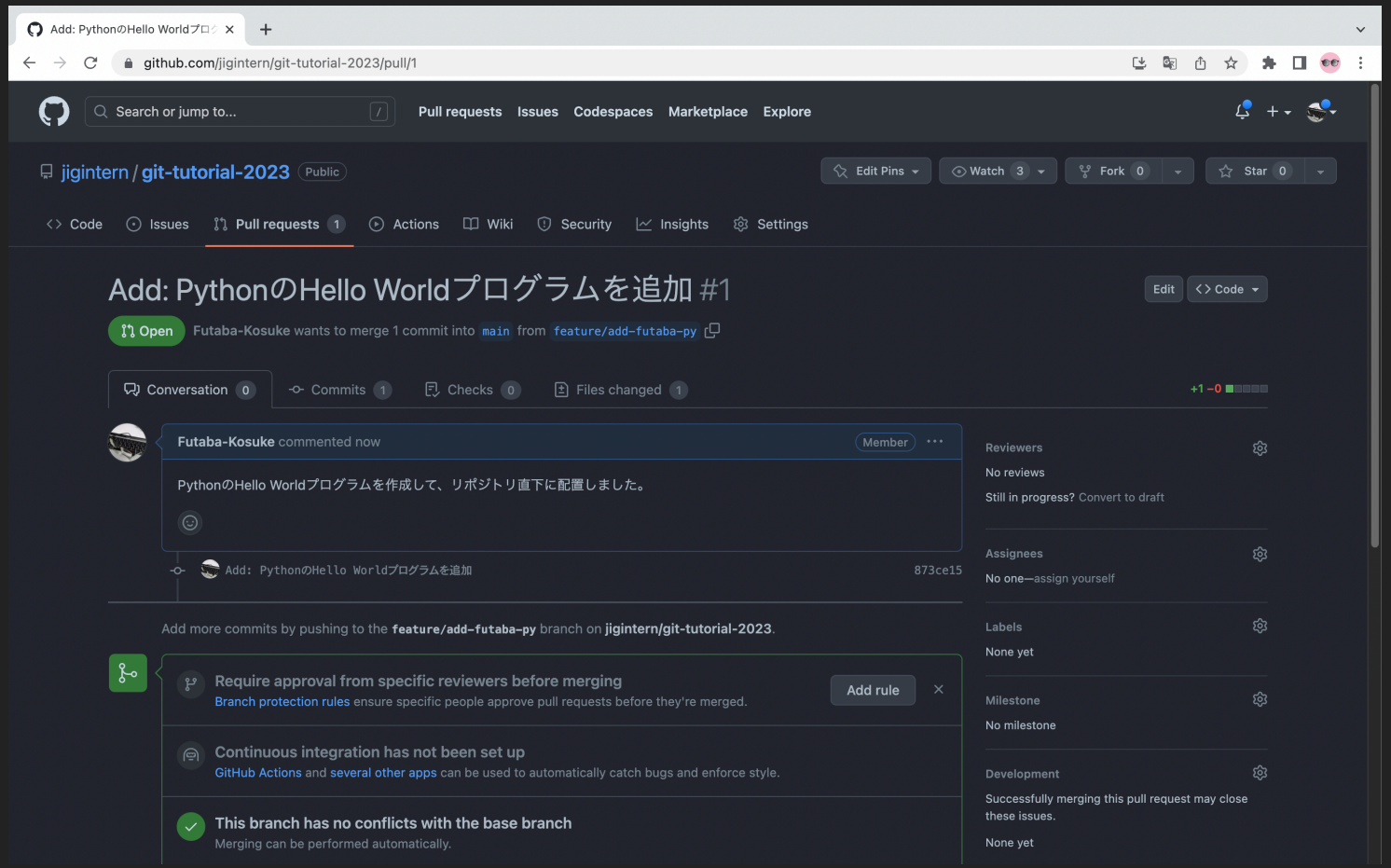
4. 「Create pull request」をクリックすると、Pull Requestが作成されます



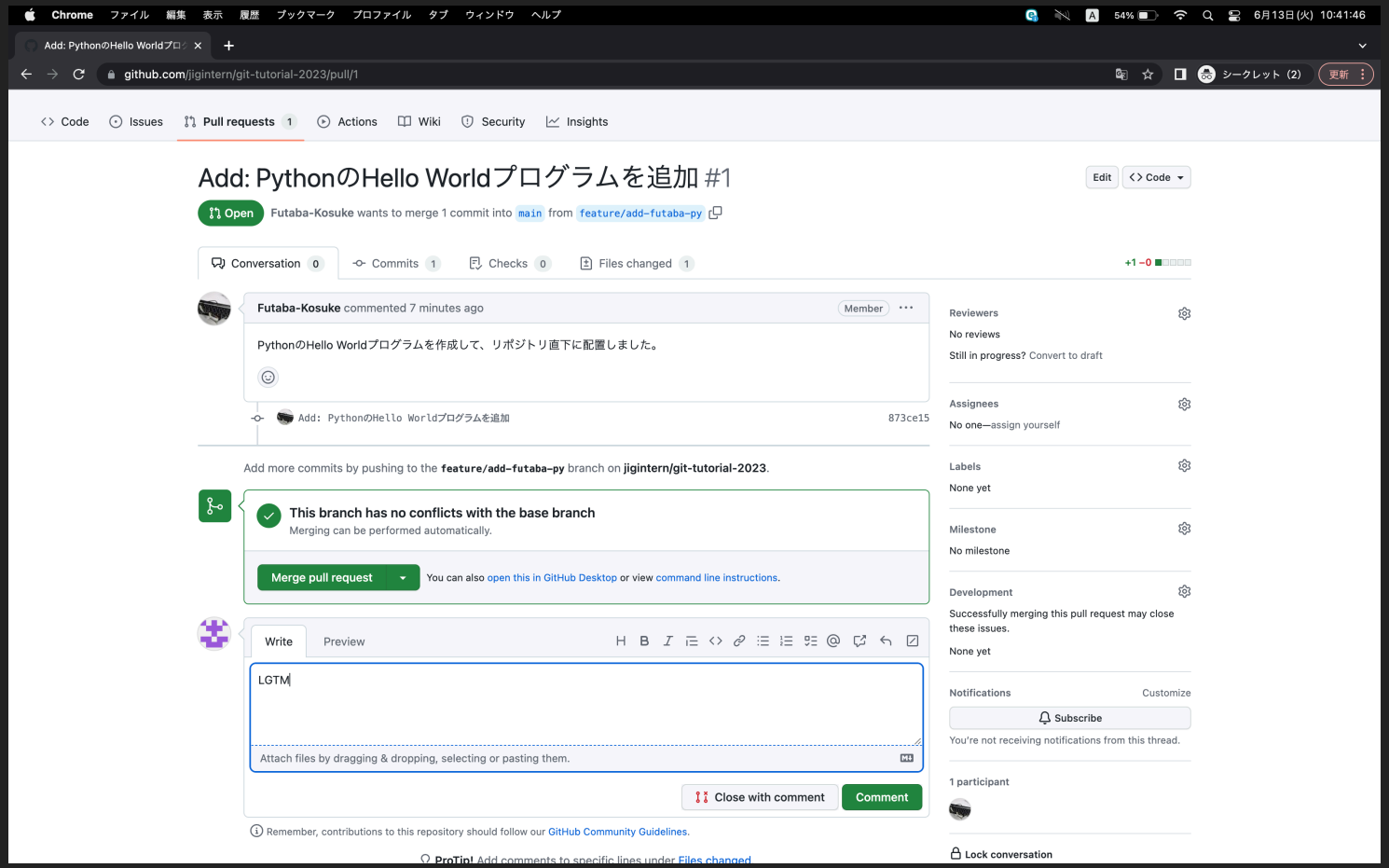
5. 他開発者に、Pull Requestの確認を依頼します

SlackやGithub上でのコメントなど、適宜チーム内で決定した方法で依頼しましょう

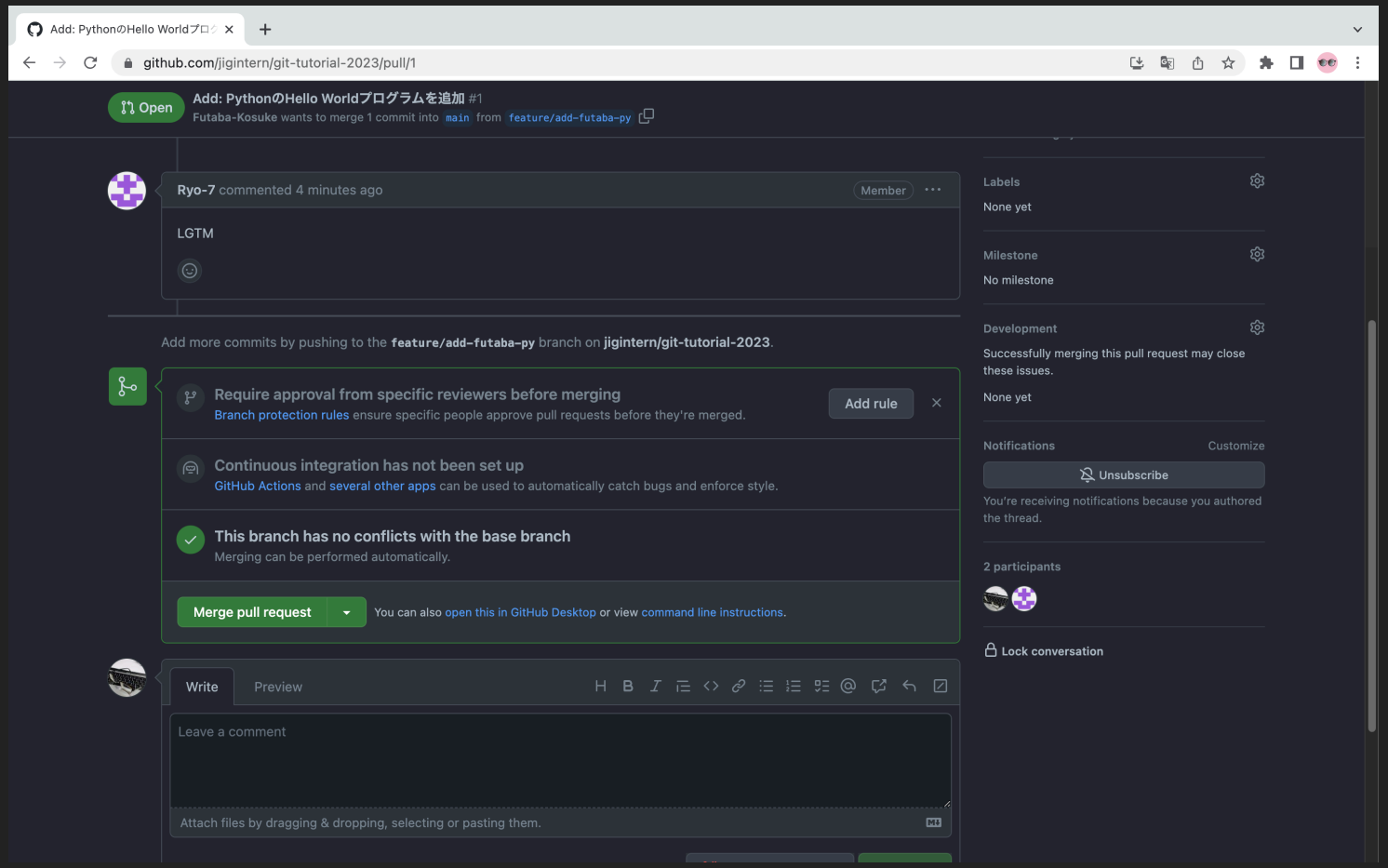
6-1. 確認を依頼された人は、Pull Requestの変更内容等を確認して、問題箇所があればコメント等で指摘します



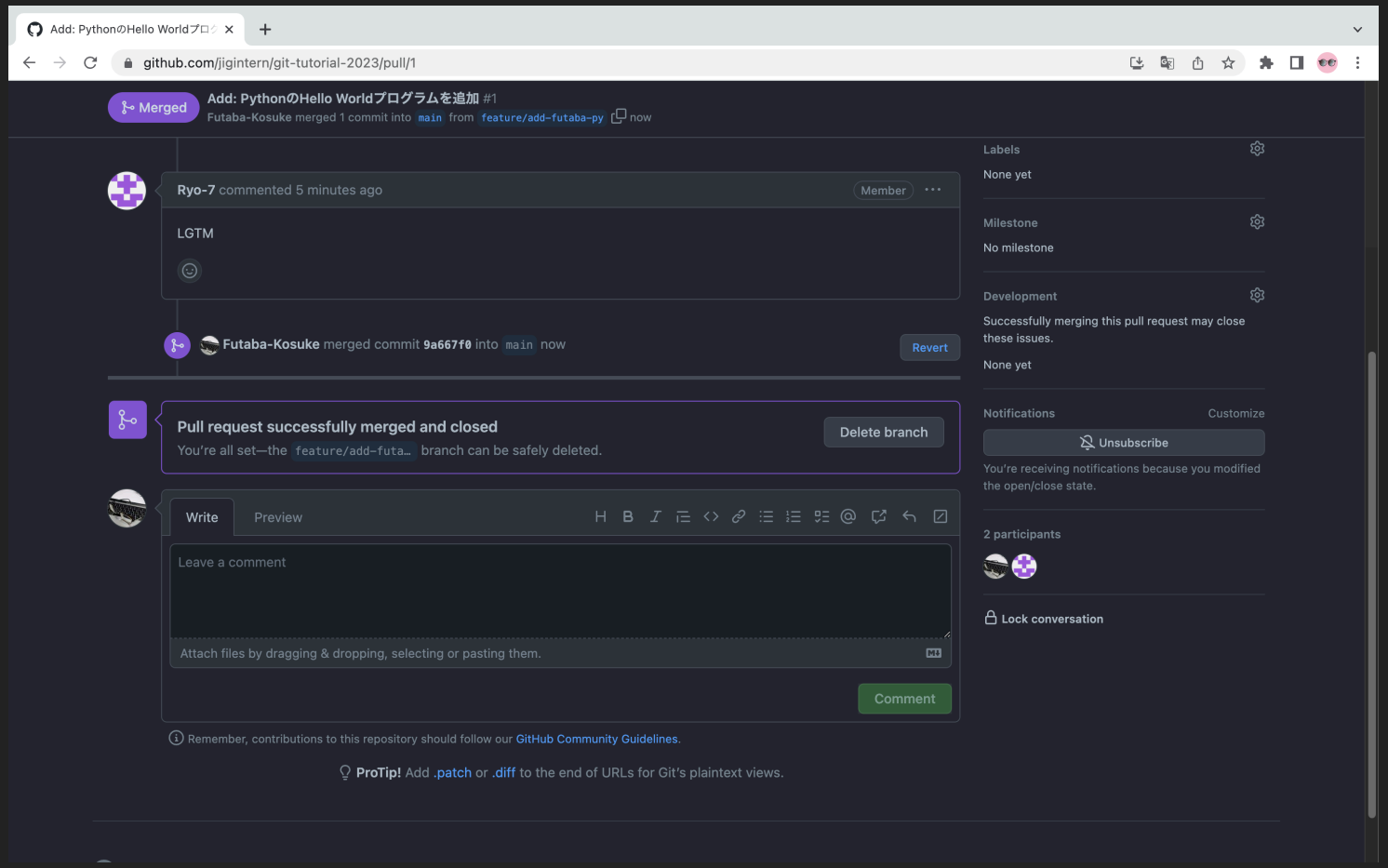
6-2. 問題箇所が無い場合は、LGTM（Looks Good To Me: 私は良いと思います）等のコメントをつけて確認したことを報告しましょう



7. Pull Requestの作成者は、「Merge pull request」をクリックして、Pull Requestを元のブランチに結合します



7. Pull Requestの作成者は、「Merge pull request」をクリックして、Pull Requestを元のブランチに結合します



8. mainブランチを確認して、変更内容が正しく取り込まれていることを確認します

Githubで開いてみましょう

2-7. fetch / pull: Githubリポジトリの変更部分をダウンロードしよう

Githubの更新を、PC上に取り込みましょう

これをfetch（フェッチ） / pull（プル）と呼びます

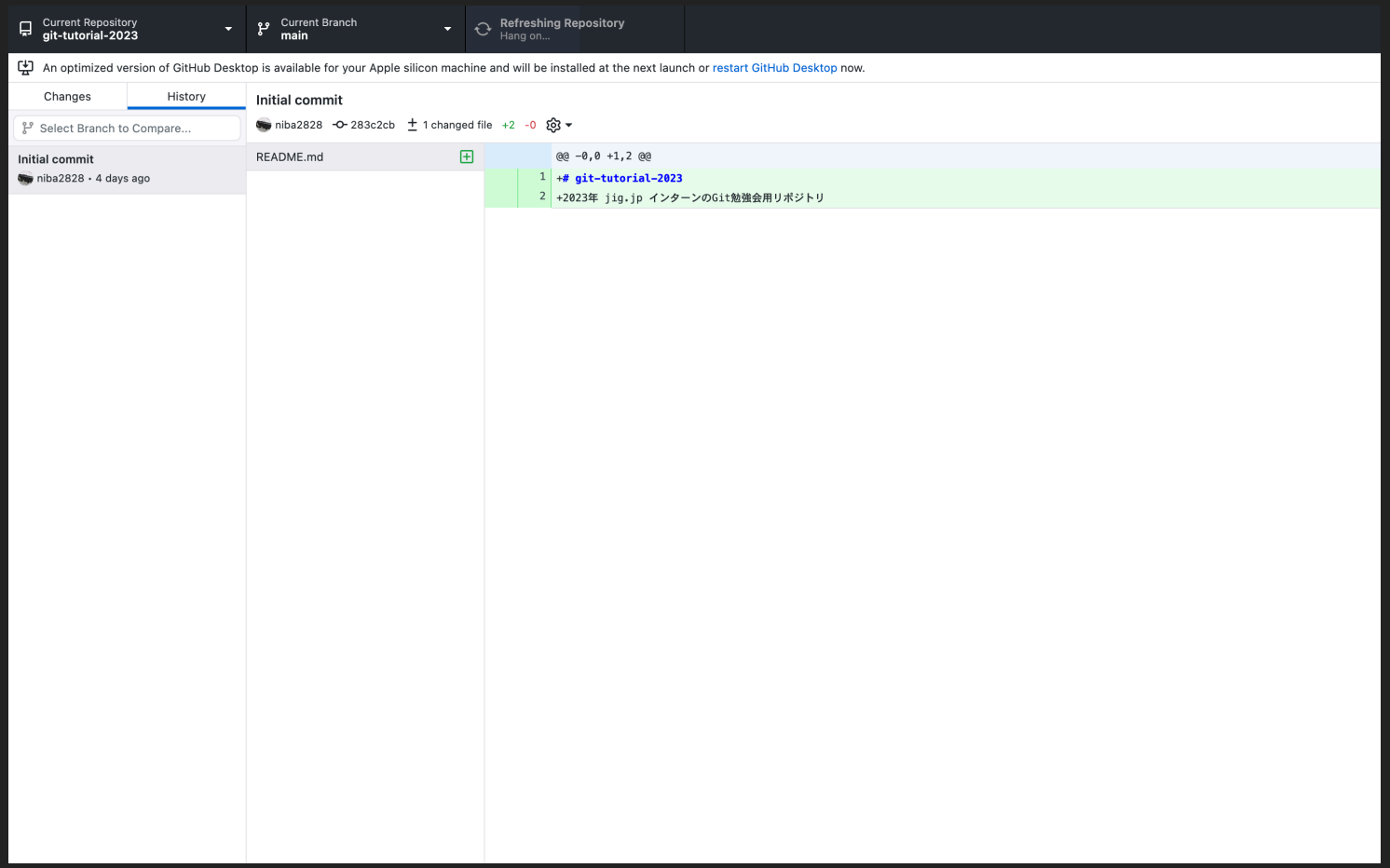
fetch（フェッチ）は、Github上のリモートリポジトリの**変更を確認**する操作です。
pull（プル）は、Github上のリモートリポジトリの**変更をダウンロード**する操作です。

1. fetchしてGithubの変更を確認して
2. pullしてその変更を取り込む

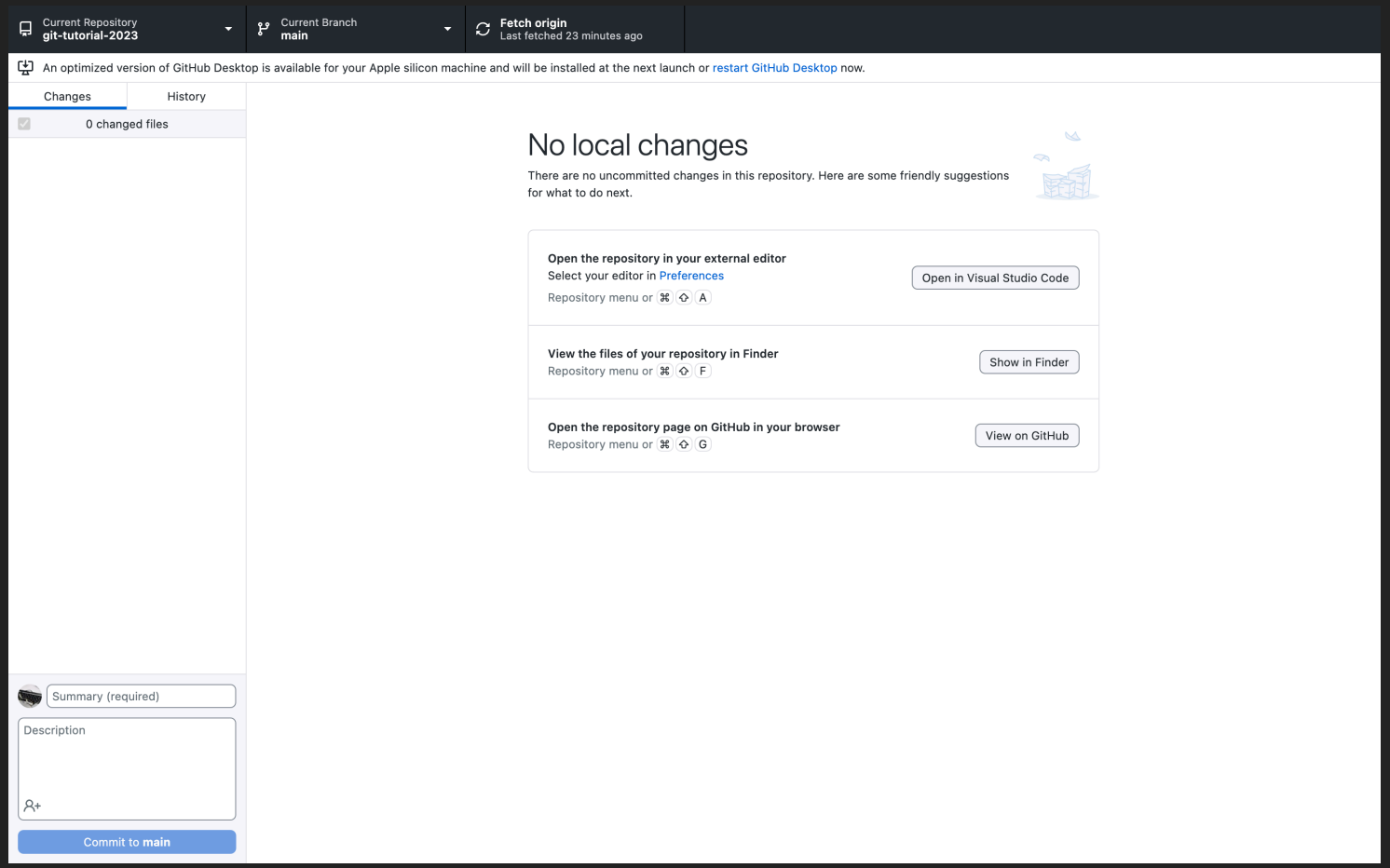
というイメージになります

実際に取り込んでみましょう

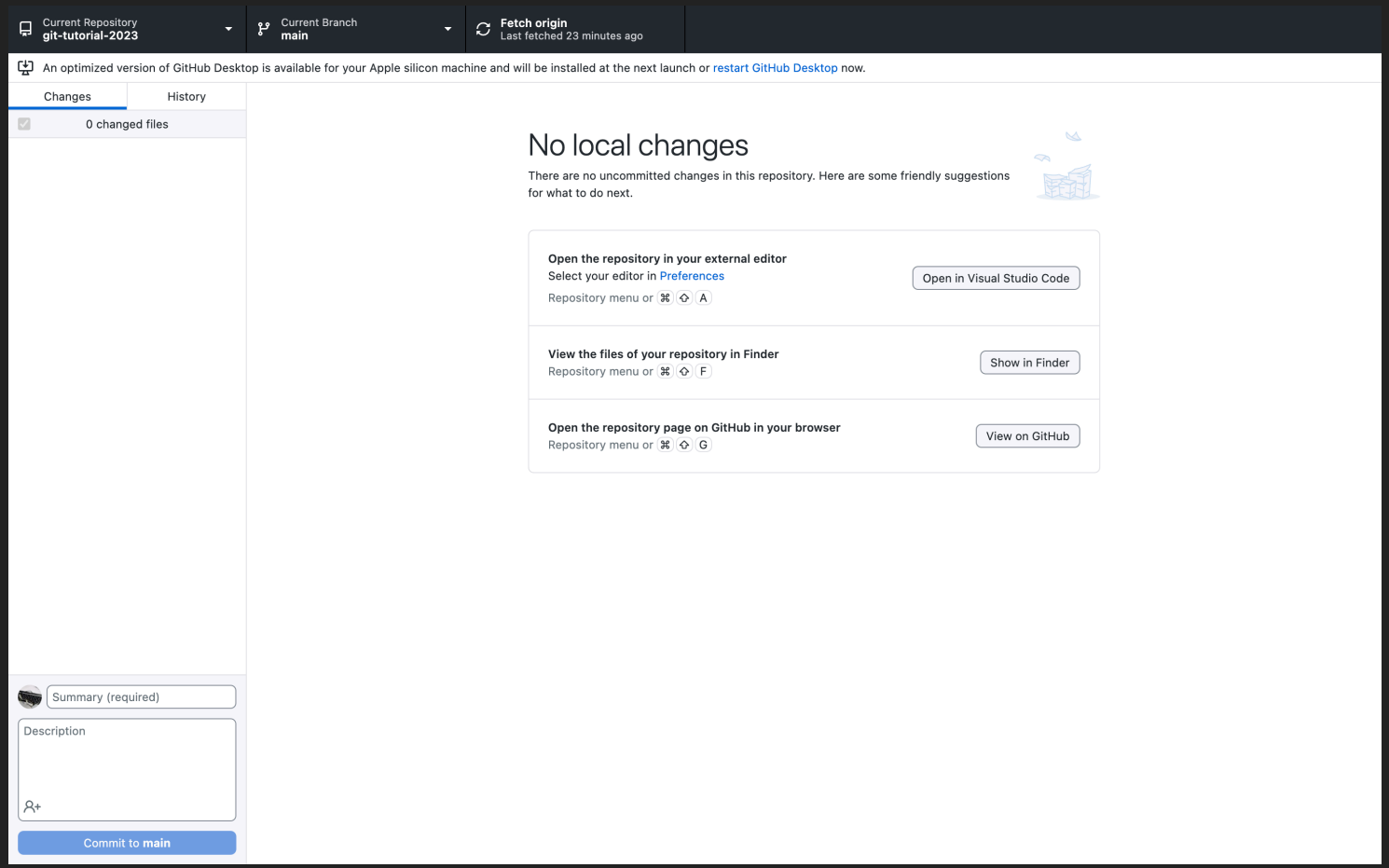
1. mainブランチに移動します



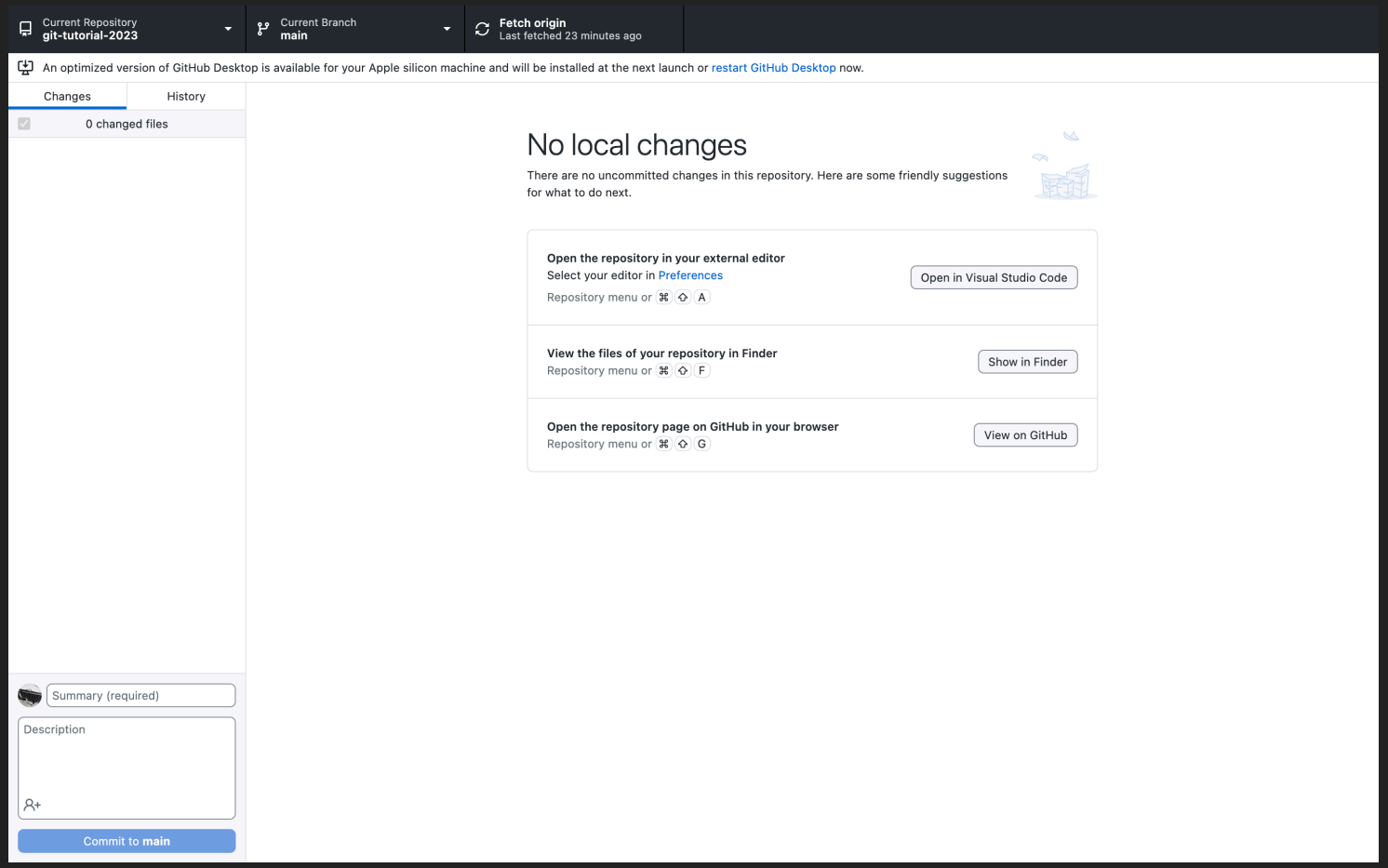
2-1. 画面上部の「Fetch origin」をクリックします



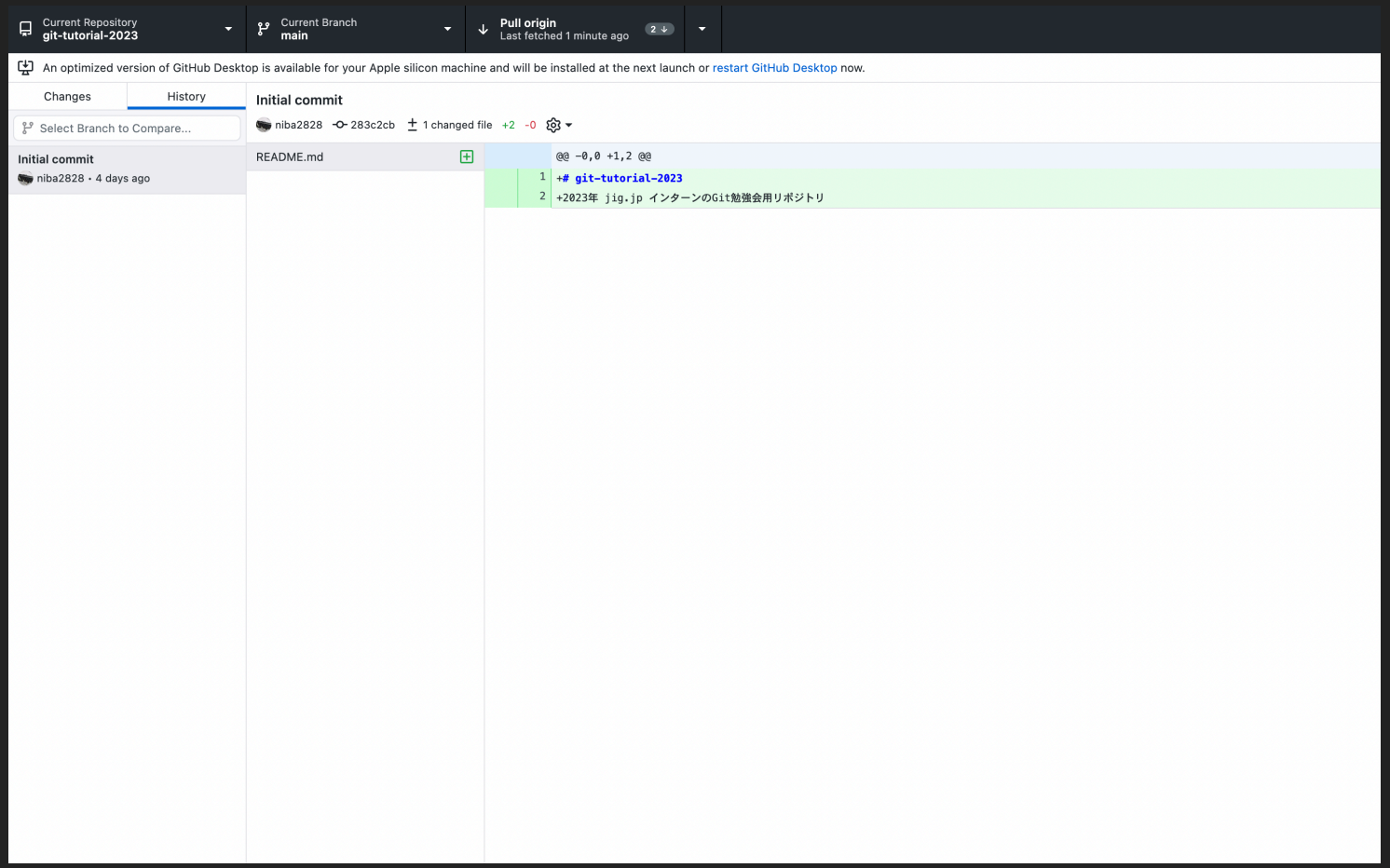
2-2. これだけでは変更内容は取り込まれませんが、Github上で変更があったことをGithub Desktopが認識します



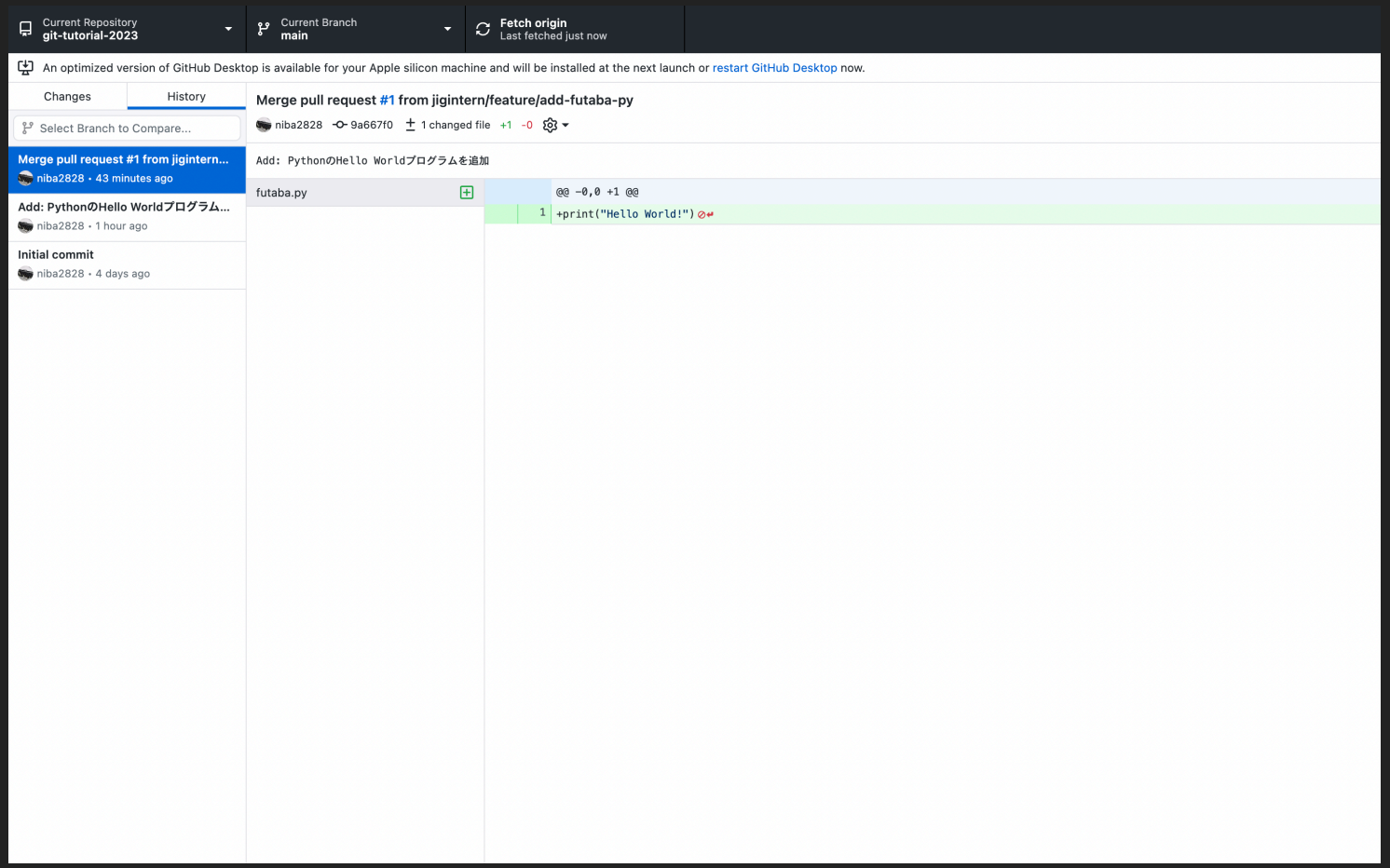
2-3.（Github Desktopの自動更新がONになっている場合、既に変更が取り込まれている場合があります）



3. 画面上部の「Pull origin」をクリックします



4. 変更内容が取り込まれていることを、「History」から確認します



3. まとめ

できるようになったことを、振り返ろう

1. clone: Githubリポジトリを丸ごと丸ごとダウンロードする
2. branch: 作業を枝分かれさせて、他の開発者との衝突を防止する
3. 通常通りに、プログラムを書く
4. commit: 作業前と後の**差分を記録**しよう
5. push: 作業の成果をGithubにアップロードして共有する
6. Pull Request: 枝分かれした成果を結合する
7. review: 他の開発者の作業内容を確認する

これでGit入門は完璧です

お疲れ様でした！