

GitHubワークショッップ セットアップ資料

Hack U Project



この資料のゴール

- ソースコードエディタの **Visual Studio Code** (以降 **VSCode**) のセットアップ
- **Git Bash**のセットアップとVSCodeとの連携
- ソースコード管理サービス**GitHub**のアカウント作成
- GitHubへの疎通確認

もくじ

1. VSCodeのセットアップ

- [Windows/macOS] VSCodeのインストール

2. VSCodeのターミナルを設定

- [Windows] Git Bashのインストール

3. VSCodeの使い方

4. Gitの環境構築

- GitHubアカウントの作成
- Repositoryの作成

5. GitHubへの疎通確認/Gitの基本操作

注意

ワークショップでは、**VSCodeとGit Bash/ターミナルを推奨環境**としています。

別のエディタやコマンドラインツールを使用している場合、最低限コマンドラインツールでGitコマンドが使えることを確認してください。

Gitコマンドでの操作は巻末に資料を載せています。

1.VSCodeのセットアップ

ソースコードエディタとは

- 代表的なソースコードエディタ
 - vi/Vim
 - Emacs
 - VSCode
 - Atom …etc
- ソースコードを編集することに特化したシステム
 - 構文強調 (シンタックスハイライト)
 - 自動補完/自動整形
 - 括弧のマッチング …etc

VSCode とは

Microsoft が開発したOSSのソースコードエディタ

動作環境	Windows, Linux, macOS
特徴機能	構文強調 デバッグ 関数ジャンプなどの拡張機能追加 GitHubの連携 CLIサポート

VSCodeのインストール

- 公式サイトからインストーラーをダウンロード

URL: <https://code.visualstudio.com/download>

- 自身のOSに合わせてダウンロードしてください

[Windows]

VSCodeのインストール

[Windows] VSCodeのインストール

1. VSCodeUserSetup-x64-[version] を実行
2. 使用許諾契約書の同意
→ 同意する(A) を選択して、 次へ(N) >
3. インストール先の指定
→ 次へ(N) > を選択
4. スタートメニューフォルダーの指定
→ 次へ(N) > を選択

[Windows] VSCodeのインストール

5. 追加タスクの選択

→ その他の全てにチェックを付けて 次へ(N) >

6. インストール準備完了

→ 内容を確認して インストール(I)

7. セットアップウィザードの完了

→ 完了(F) を選択VSCode が起動します

[macOS]

VSCodeのインストール

[macOS] VSCodeのインストール

1. VSCode-darwin-universal.zip を解凍
2. Visual Studio Code.app を
アプリケーションに移動させてダブルクリック
 - WEBからダウンロードしたものです.. には**信頼する**を選択

2.VSCodeのターミナルを設定

Git Bashとは

Git Bash

- WindowsでGitコマンドを扱うために作られた
コマンドラインツール
- VSCodeのターミナルとして設定可能

※ **MacOS**ではデフォルトのターミナルが使用できるため、必要ありません。

※ すでにGitが使えるコマンドラインツールがある場合は、この項目は飛ばしても構いません。

[Windows]

Git Bashのインストール

Git Bashのインストール

1. 公式サイトよりインストーラをダウンロード
 - **URL:** <https://gitforwindows.org>
2. ダウンロードした Git-2.31.1-64-bit.exe をクリック
3. 「このアプリがデバイスに変更を加えることを許可しますか」→ はい を選択
4. 「Information」→ Next > を選択

Git Bashのインストール

4. 「Select Destination Location」

→ インストール先を確認し、問題なければ **Next >** を選択

5. 「Select Components」

下記の3つのみチェックをつけて、**Next >** を選択

- Git Bash Here
- Associate .git* configuration files with the default text editor
- Associate .sh files to be run with Bash

Git Bashのインストール

6. 「**Select Start Menu Folder**」
→ 特に変更せずに **Next >** を選択
7. 「**Choosing the default editor used by Git**」
→ 「Use Visual Studio Code as Git's default editor」を選択して **Next >**
8. 「**Adjusting the name of the initial branch in new repositories**」
→ 「Override the default branch name for new repositories」を選択し、
テキストボックス内が「**main**」となっていることを確認

Git Bashのインストール

9. 「Adjusting your PATH environment」

→ 個人の環境によります

※基本は **b** を推奨

a. PowerShell等でUNIX環境を整えてる場合

2つ目の「Git from the command line and also from 3rd-party software」を選択して **Next >**

b. 何も環境設定してない場合

3つ目の「Use Git and optional Unix tools from the Command Prompt」を選択して **Next >**

Git Bashのインストール

10. 「**Choosing HTTPS transport backend**」
→ 1つ目の「Use the OpenSSL library」を選択して **Next >**
11. 「**Configuring the line ending conversions**」
→ 3つ目の「Checkout as-is, commit as-is」を選択して **Next >**
12. 「**Configuring the terminal emulator to use with Git Bash**」
→ 1つ目の「Use MinTTY(the default terminal of MSYS2)」を選択して **Next >**

Git Bashのインストール

12. 「**Choose the default behavior of `git pull`**」
→ 1つ目の「Default (fast-forward or merge)」を
選択して **Next >**
13. 「**Choose a credential helper**」
→ 1つ目の「Git Credential Manager Core」を
選択して **Next >**
14. 「**Configuring extra options**」
→ **全ての項目**を選択して **Next >**

Git Bashのインストール

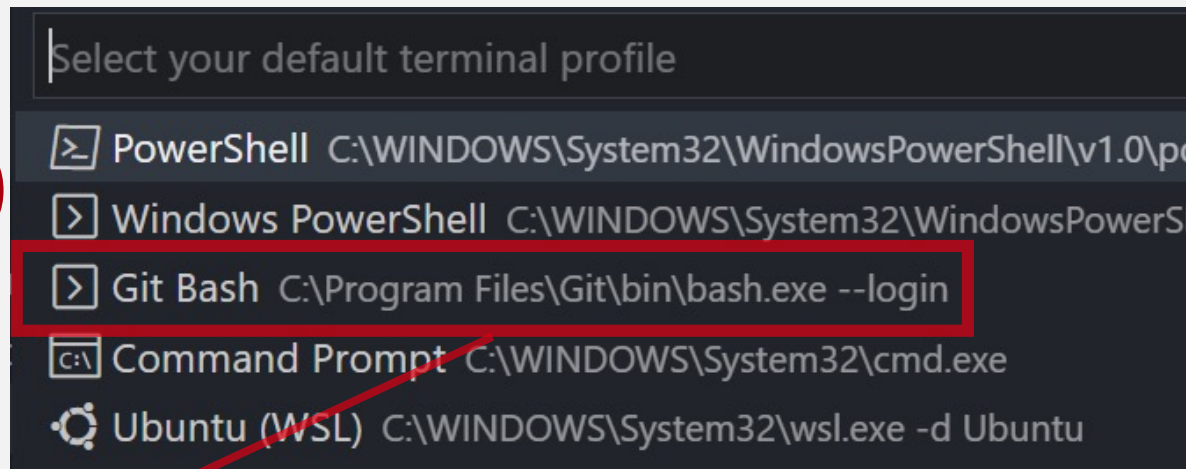
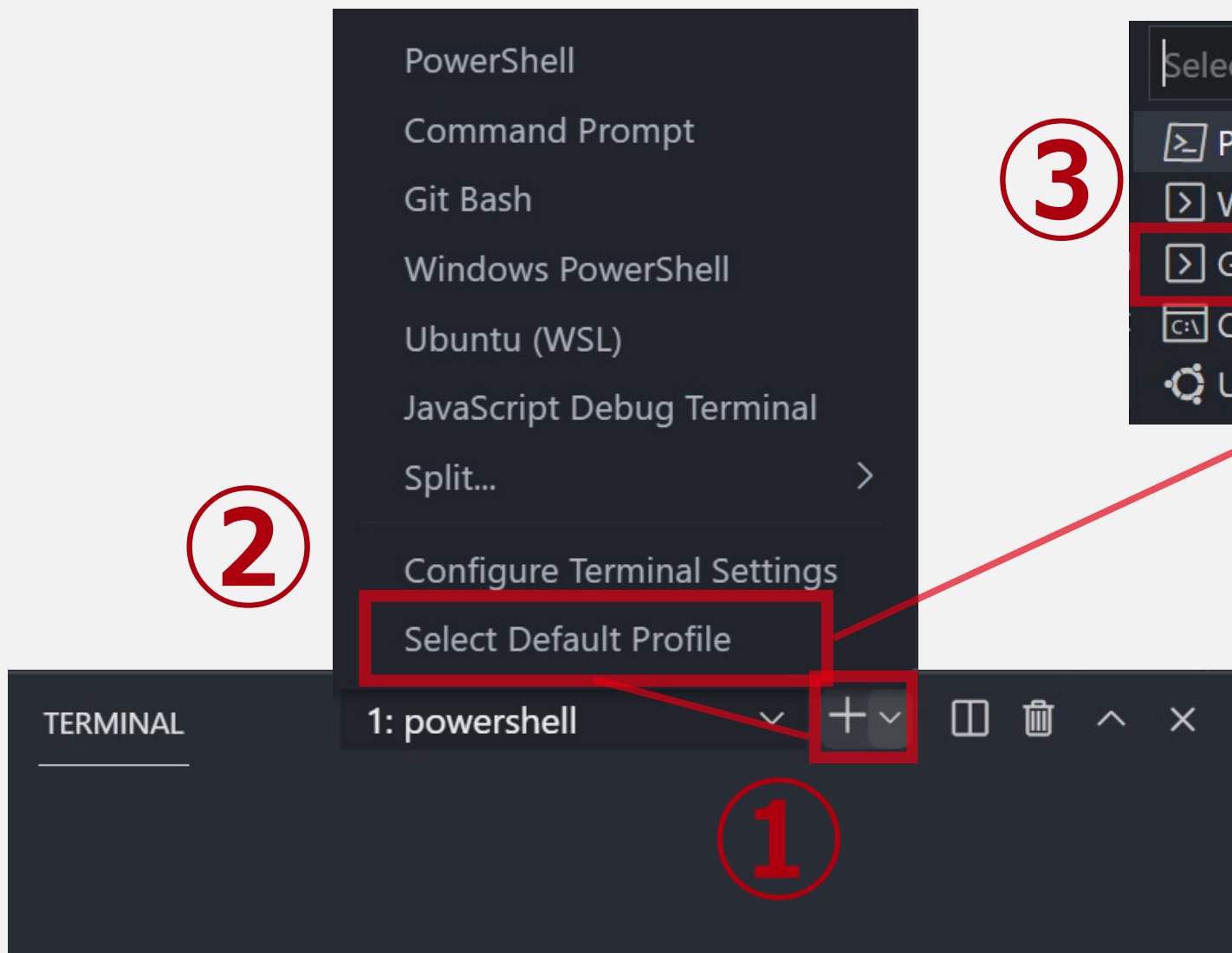
14.「Configuring experimental options」

→ 何も選択せずに **Install** を選択

15.「Completing the Git Setup Wizard」

→ 全てのチェックを外して **Next >** を選択

VSCodeにGit Bashを連携



ターミナルの設定画面から、
Select Default Profile
を選び
Git Bashを選んでください

3.VSCodeの使い方

VSCodeにライブラリを追加する方法

- 機能を拡張するライブラリを追加
 - 使用する言語のLintツール
 - カラーテーマ …等
1. メニューから **View** → **Extensions** を選択
 2. テキストボックスでライブラリの検索
 3. ほしいライブラリをクリックして詳細画面を表示
 4. **Install** ボタンからインストール

Workspaceを設定

1. メニューから「File」→「Open Workspace...」を選択
2. 開発物の保存場所を選択して、**Open** を選択
 - 成果物を保存する用のディレクトリを作成しておくことをおすすめします

ターミナルを開く

- メニューから「Terminal」
→ 「New Terminal」を選択
 - 場合により開くディレクトリを選択

ファイルの編集

- 新規ファイルの作成
 - File → New File
- 既存ファイルの編集
 - File → Open File
 - 該当ファイルを右クリックで **Codeで開く** を選択

※ どの操作もキーボードショートカットがあるので覚えると便利です。
メニューの右側に表示されているので参考にしてください

例：新規ファイルの作成 → **⌘+N** (Mac) または **Ctrl+N**(Windows)

4.Gitの環境構築

バージョン管理とは

- 代表的なバージョン管理システム
 - Git
 - Subversion …etc
- **ファイルの変更を記録**しておくシステム
 - 誰が
 - いつ
 - どのファイルを
 - どう変更したか

バージョン管理とは

ファイルの変更を記録する場所

→ **リポジトリ**



APIサーバ
リポジトリ



Androidアプリ
リポジトリ



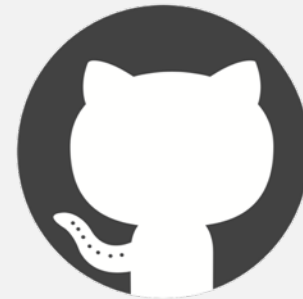
iPhoneアプリ
リポジトリ

バージョン管理とは

- **リモートリポジトリ**：
色んな人が共有できるようにしたリポジトリ
GitHub上にあるものはリモートリポジトリ
- **ローカルリポジトリ**：
自分のPC上にだけ存在するリポジトリ



ローカルリポジトリ



リモートリポジトリ

Git と GitHub

Git	バージョン管理の方式 (バージョン管理とは、いつ、誰が、どのファイルを、どう変更したかを記録すること)
GitHub	Gitでバージョン管理をしやすくする Webサービス ブラウザから変更履歴やコードを共有するのがかんたんにできる

Gitを使うには

- コマンドラインでGitコマンドを使う
- GUIツールを使う
- Visual Studio CodeでGitコマンドを使う
(以後 VSCode と表現します)

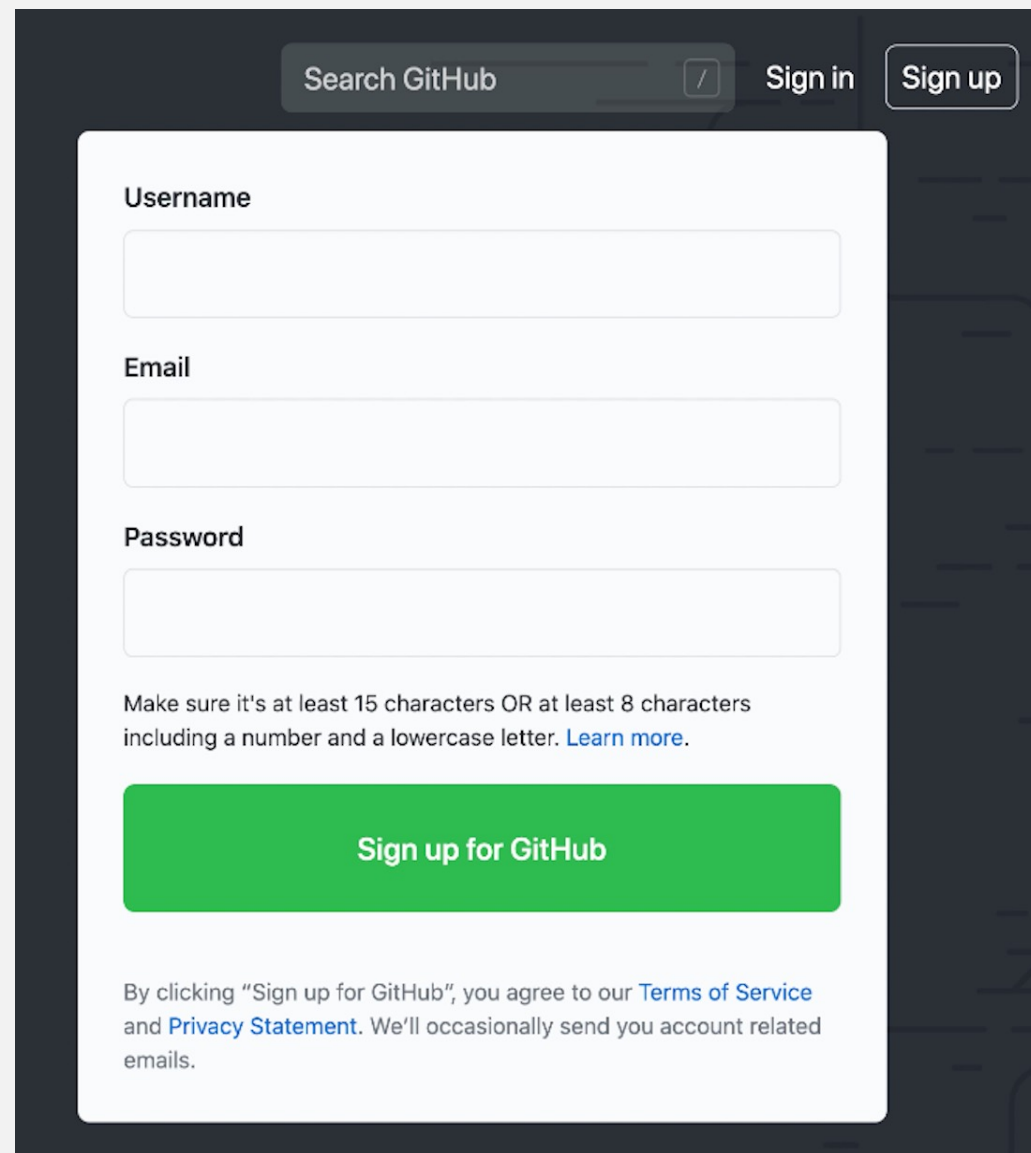
今回はVSCodeで
Gitコマンドを使います

GitHubのアカウント作成

GitHubのアカウントを作る

- 右上のSign upをクリック
 - <https://github.com>
- 必要事項を記入する
 - アカウント名
 - メールアドレス
 - パスワード

アカウント名は公開されます



The image shows the GitHub sign-up page. At the top, there is a search bar labeled 'Search GitHub' and two buttons: 'Sign in' and 'Sign up'. Below these is a white sign-up form with the following fields: 'Username', 'Email', and 'Password'. Each field has a corresponding input box. Below the 'Password' field, there is a note: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)'. At the bottom of the form is a large green button labeled 'Sign up for GitHub'. Below the button, there is a disclaimer: 'By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.'

GitHubのアカウントを作る

- 認証
- Welcome to GitHub
 - 仕事は何か
 - どのくらいプログラミングができるか
 - 何のためにGitHubを使うか
 - 興味があるもの
- メールを確認 “Verify email address” でアクセス

GitHubのアカウントを確認する

- プロフィールページにアクセス
 - `https://github.com/アカウント名`

※アカウント作成されていない場合、404になります

Git設定ファイルを作成

- Gitを使う際には設定ファイルの作成が必要です
1. VSCode等でTerminalを開いて下記コマンドを実行
 - `git config --global user.name “作成したアカウント名”`
 - `git config --global user.email “登録したメールアドレス”`
 2. Terminalで `cat ~/.gitconfig` を実行し、
設定した**アカウント名とメールアドレス**表示されればOK

リポジトリの作成


リポジトリを作る

Create repositoryをクリック

Owner には先程作った
アカウント名が入る

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner *
 アカウント名

Repository name *

Great repository names are short and memorable. Need inspiration? How about [scaling-eureka?](#)

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.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▼

Add a license: **None** ▼ 

Create repository

リポジトリを作る

Create repositoryをクリック

Owner には先程作った
アカウント名が入る


Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * /

Great repository names are short and memorable. Need inspiration? How about [scaling-eureka?](#)

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.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

[i](#)

今回は「**github_ws**」
というリポジトリ名で
作りましょう

リポジトリを作る

Create repositoryをクリック


Owner には先程作った
アカウント名が入る

リポジトリを誰でも参照
可能にするかを設定する
(今回は **public**)

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * Repository name *

 アカウント名 /

Great repository names are short and memorable. Need inspiration? How about [scaling-eureka?](#)

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.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

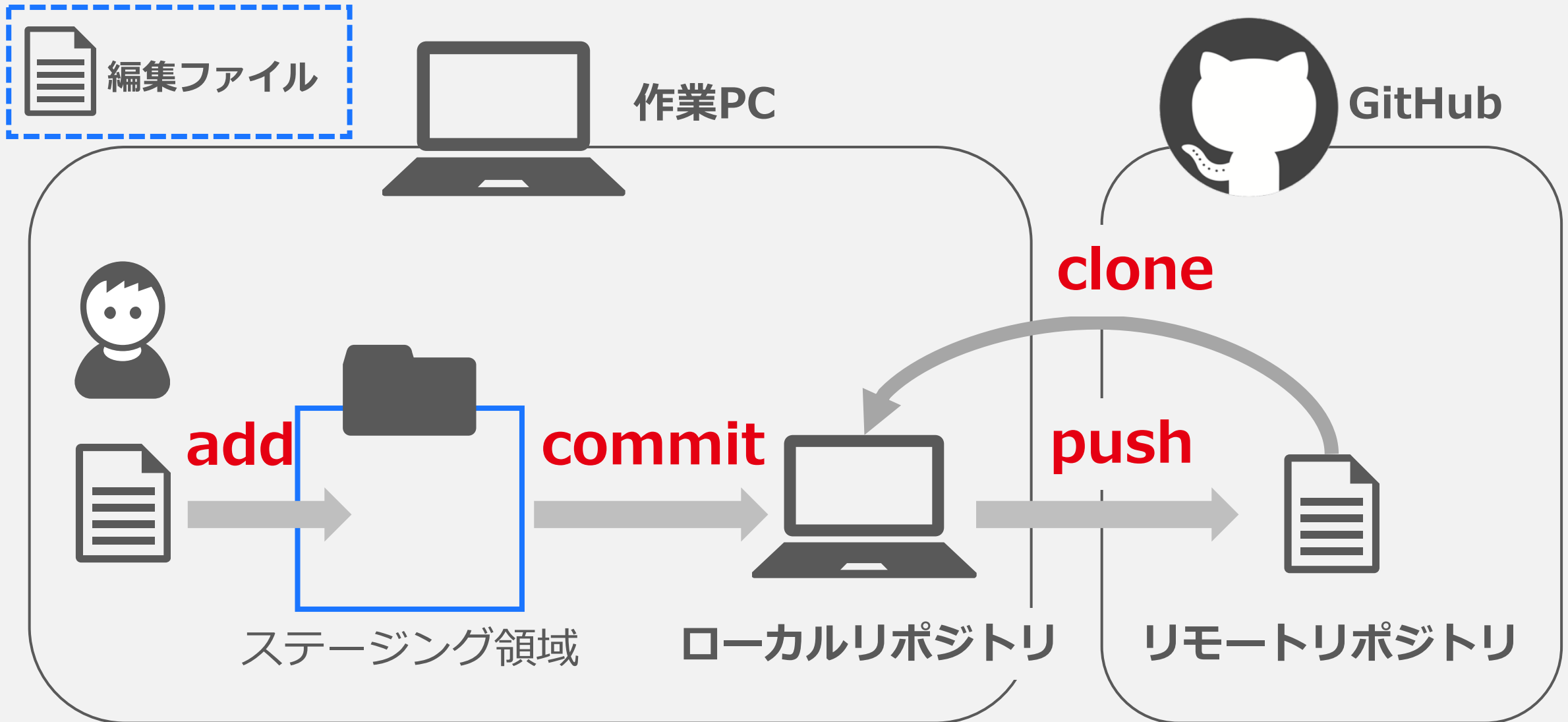
Add .gitignore: **None** ▼ Add a license: **None** ▼ [?](#)

Create repository

今回は「**github_ws**」
というリポジトリ名で
作りましょう

5.GitHubの疎通確認

GitHubの基本操作



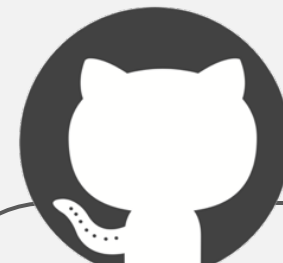
GitHubの基本操作



編集ファイル



作業PC



GitHub

疎通確認のため、
一通りのGitの基本操作を行います

詳しい解説は、
メイン資料に記載しています

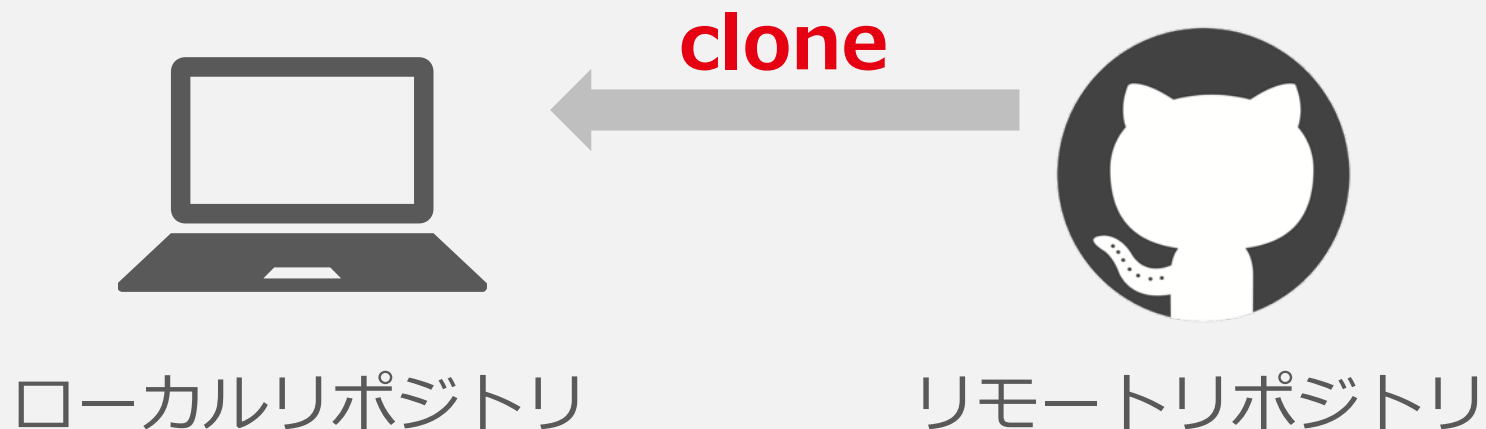
VSCodeで
cloneする

GitHub から自分のPCに持ってくる

- **clone** :

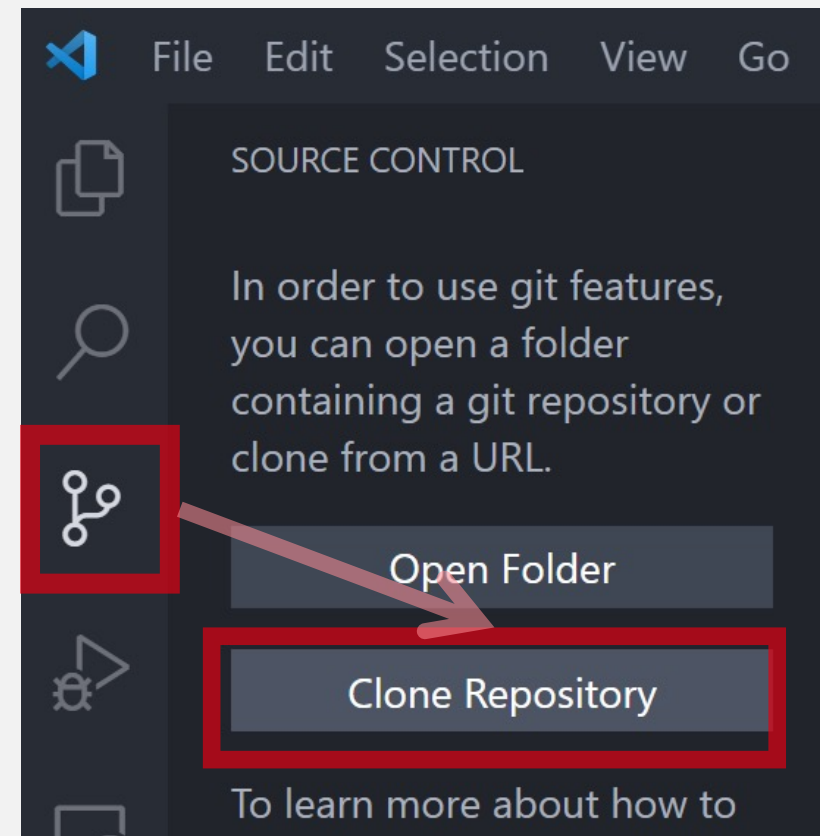
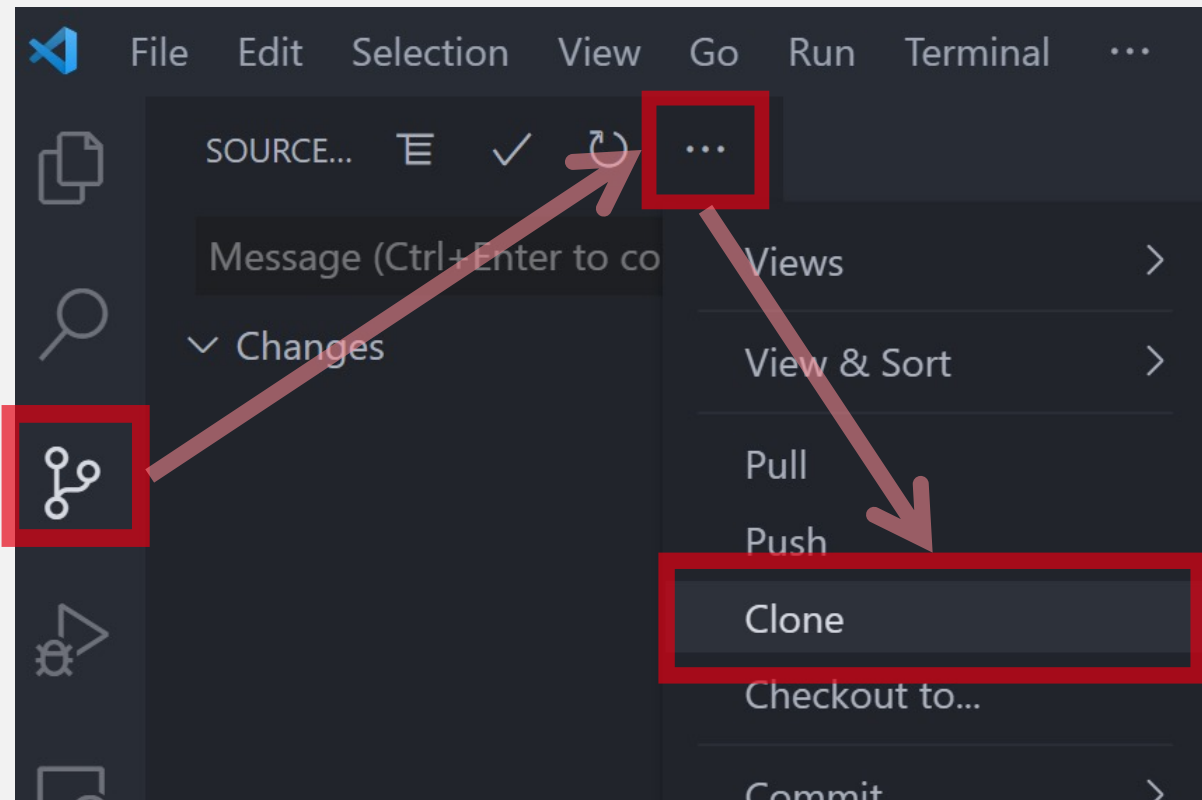
リモートリポジトリを自分のローカルリポジトリにコピーする

※ 初回のみ操作



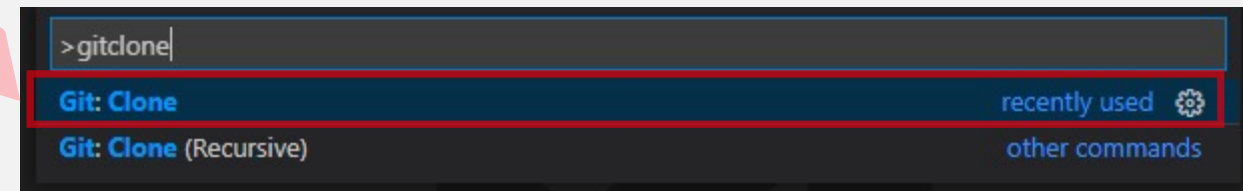
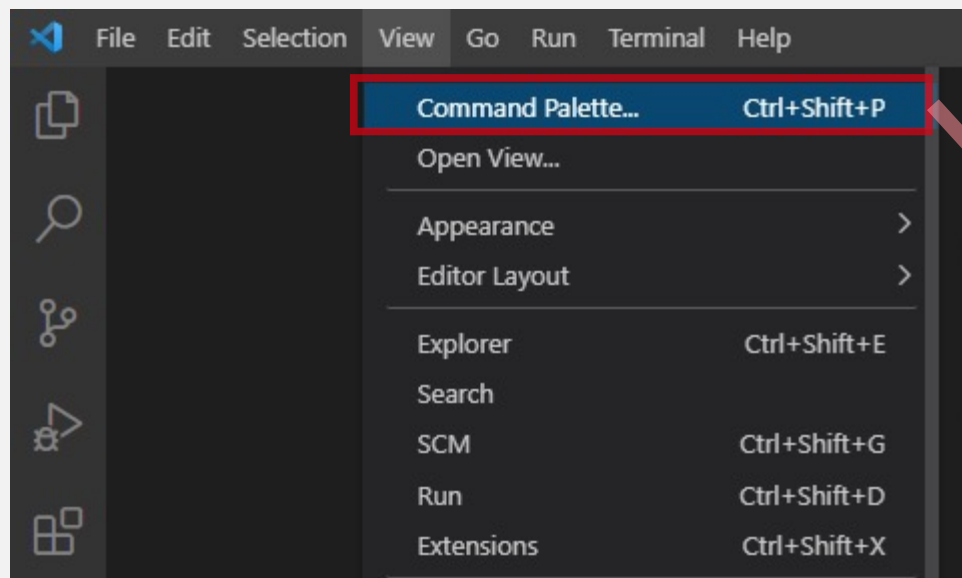
clone ①

- その時のVSCodeの状態によって、
下2つのどちらかの手順でcloneを行えます



clone ① 補足

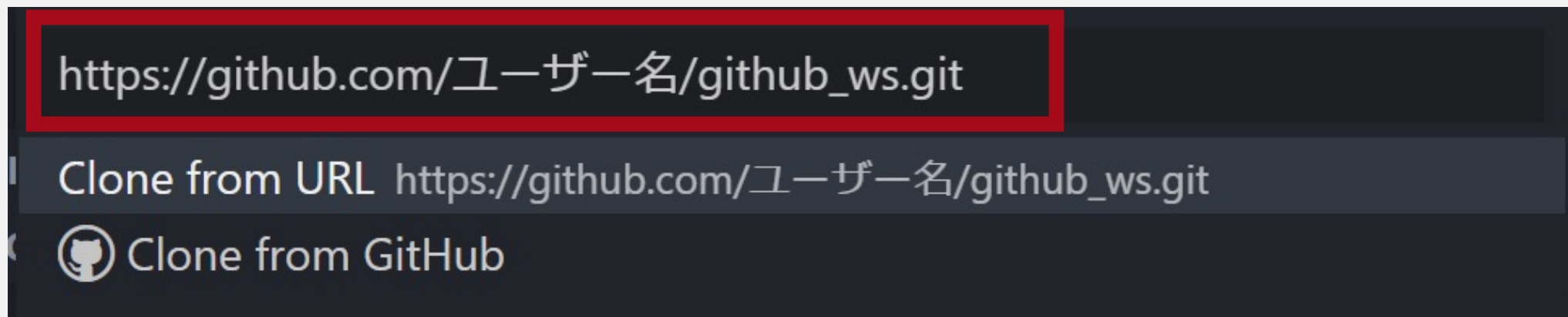
- VSCodeのUIが異なる際は、「Command Palett...」から検索します



検索欄に「**gitclone**」と入力して、
出てきた「Git: Clone」をクリック

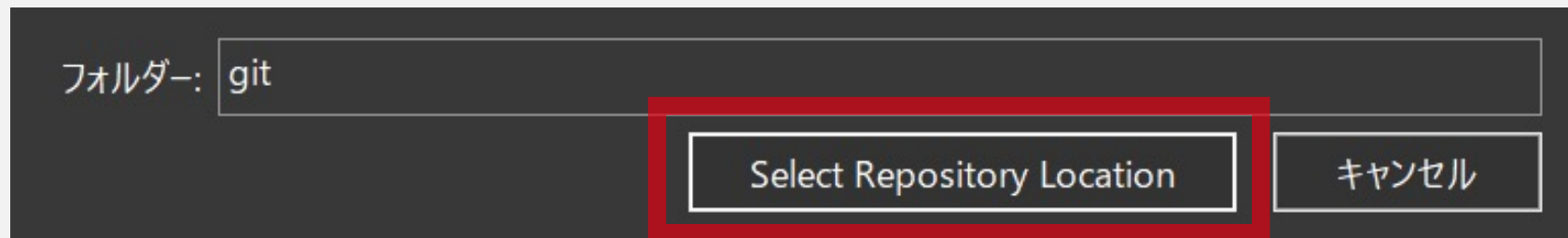
clone ②

- 出てきた入力欄に、
自分で作成したリポジトリのURL
「**https://github.com/ユーザー名
/github_ws.git**」を入力し、Enterを押します

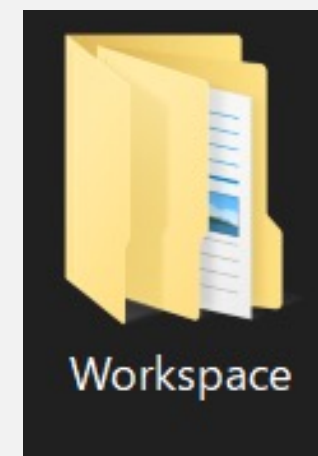
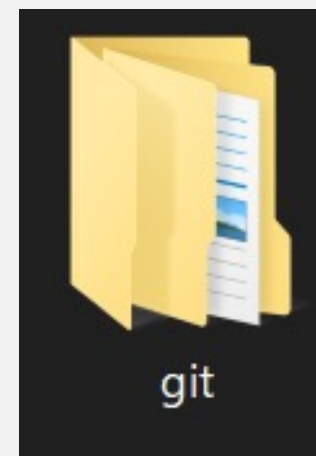


clone ③

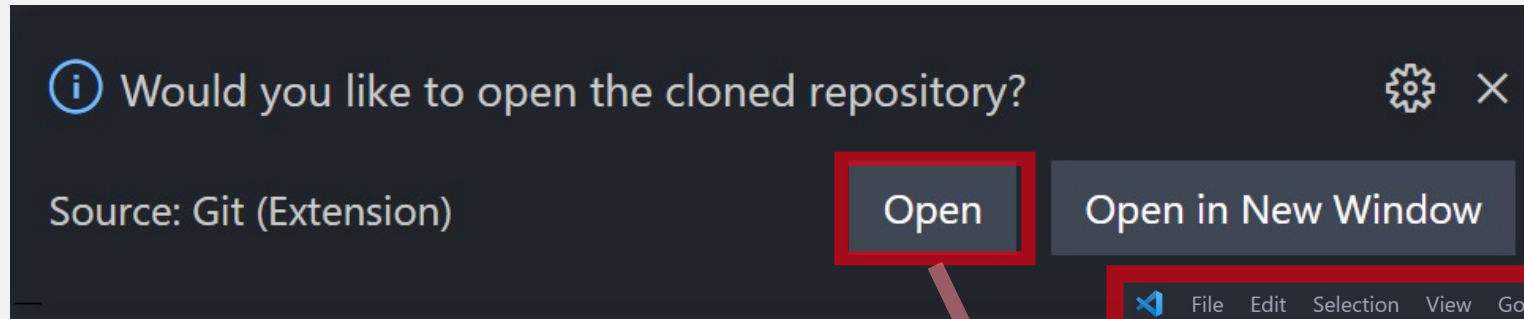
- 適当なディレクトリを指定して
[Select Repository Location]を押す



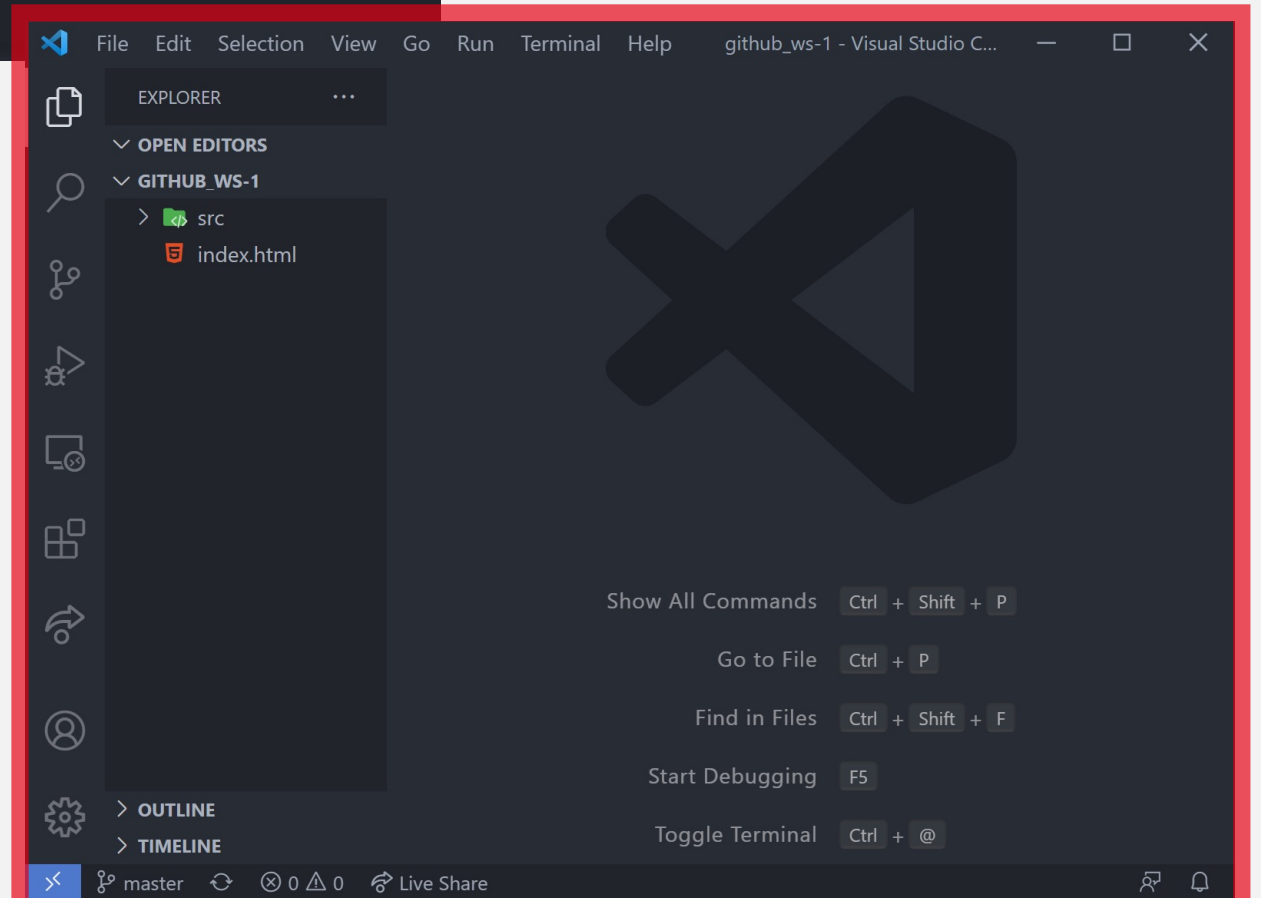
- 「git」や「workspace」などのディレクトリ下にcloneするとリポジトリが管理しやすいです



clone ④



- Openを押したら
Clone完了です

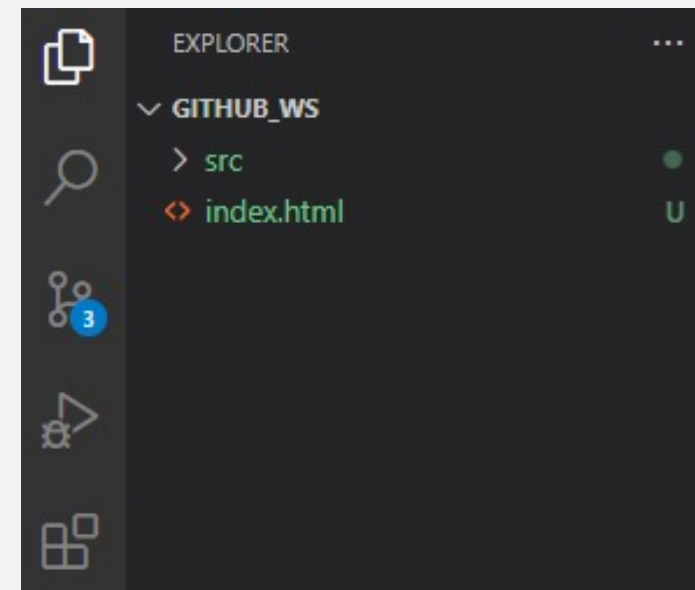


サンプルファイルの準備

- GitHub の Search or jump to ... から「**hackujp/github_tutorial**」を検索
- リポジトリ内の ver_VSCode/hands-on.zip を開く
 - Download ボタンよりダウンロード
- hands-on.zip を解凍

サンプルファイルの準備

- 先程Cloneした「github_ws」に展開したファイルを移動させる
- 移動させる対象は以下の通り
 - **index.html**
 - **src**
- 右図のようになっていればOKです



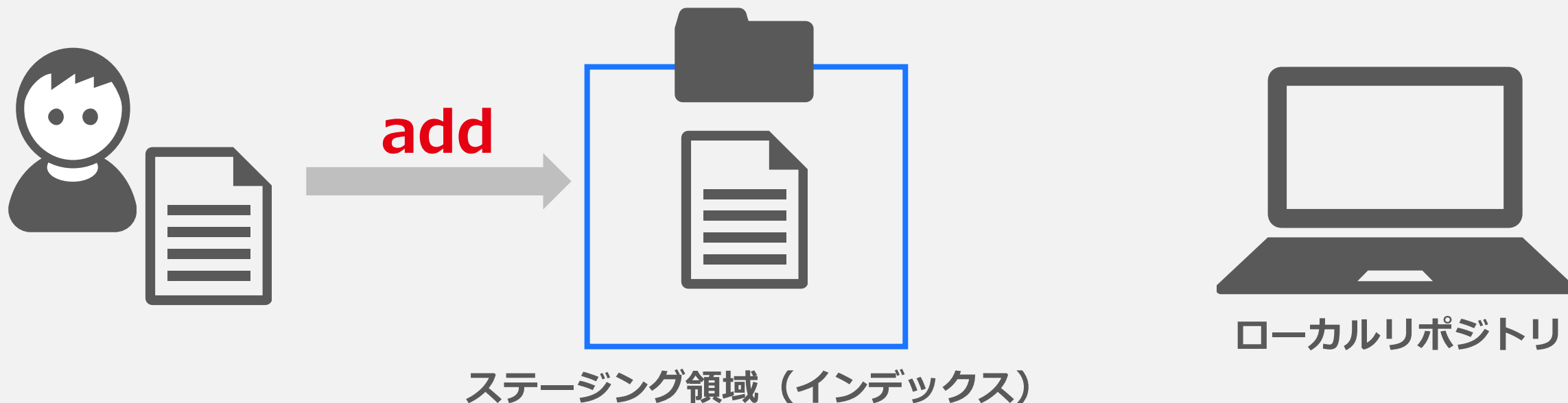
※以降の作業では、上記のファイルに対してGitの操作を行ってください

VSCodeで
addする

追加したファイルをGitで扱う

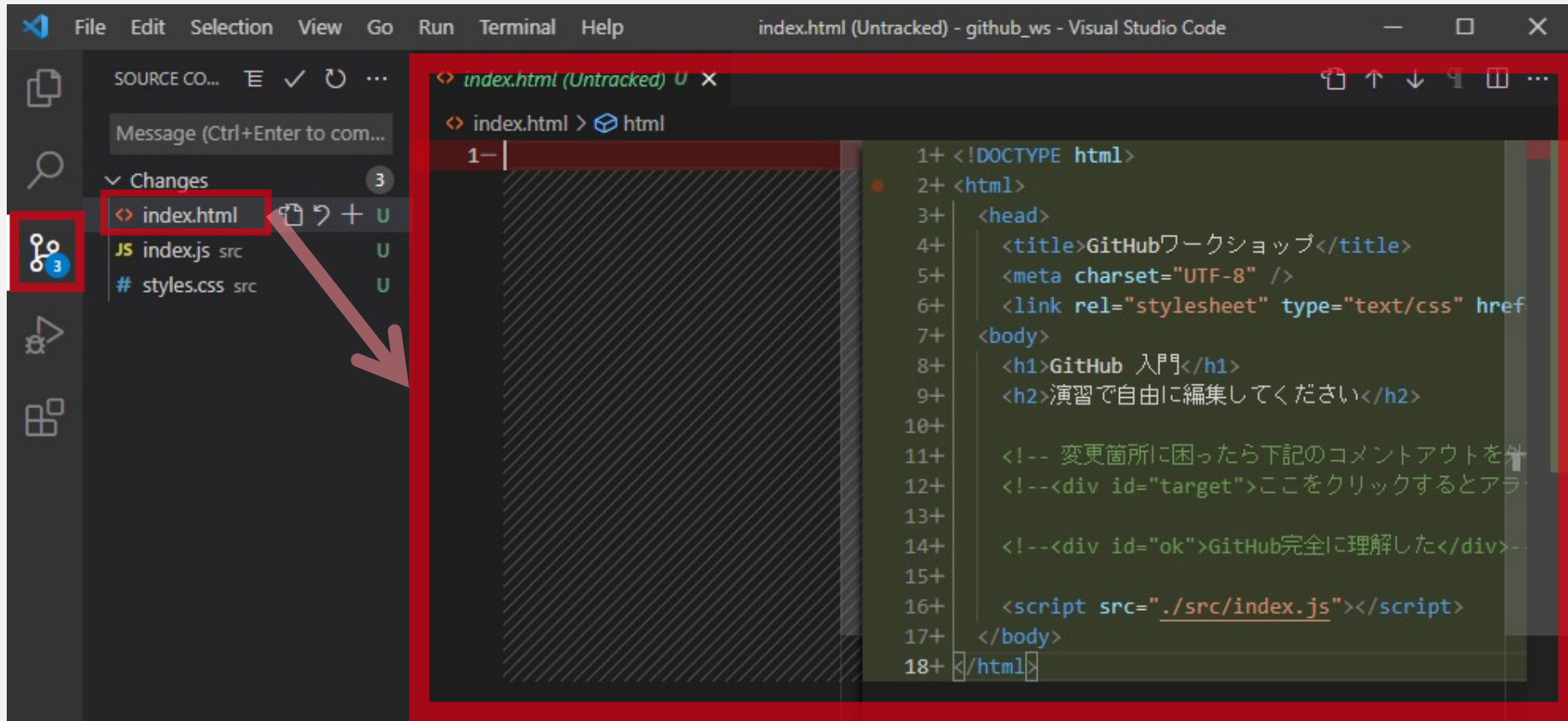
Step.1

どのファイルの追加・変更をしたかをステージング領域に一度登録する → **add** という



ステージングにaddする

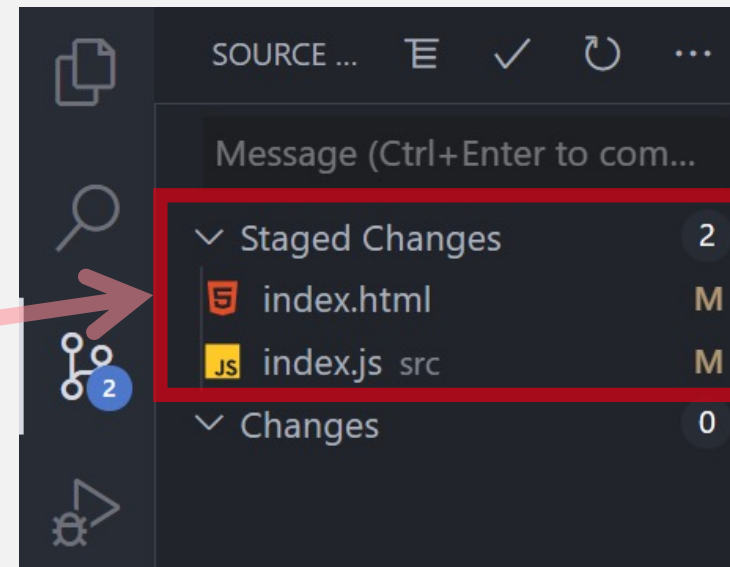
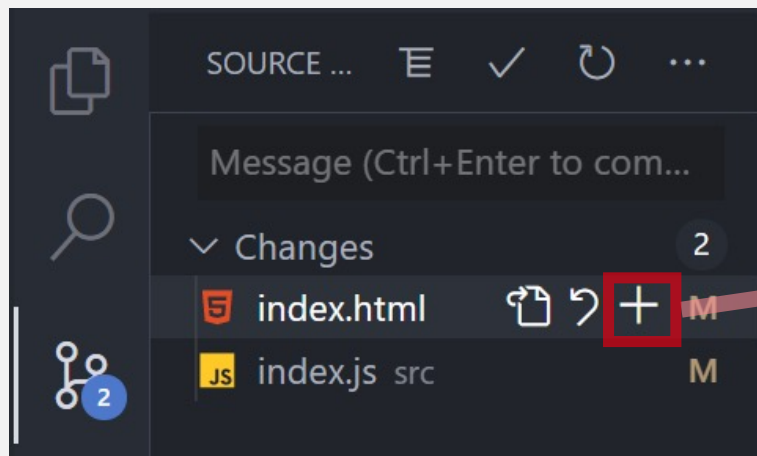
- addする前にVSCodeのメニューから追加・変更内容を確認できます



ステージングにaddする

- addしたいファイルの「+」をクリックすると、「Changes」から「Staged Changes」にファイル名が移動します

全てファイルを「Staged Changes」に入れましょう



add する際の注意点

- add するファイルの内容を確認
 - 個人情報
 - パスワード
 - API Key などの鍵情報

リモートリポジトリに push したファイルは
消すことができません

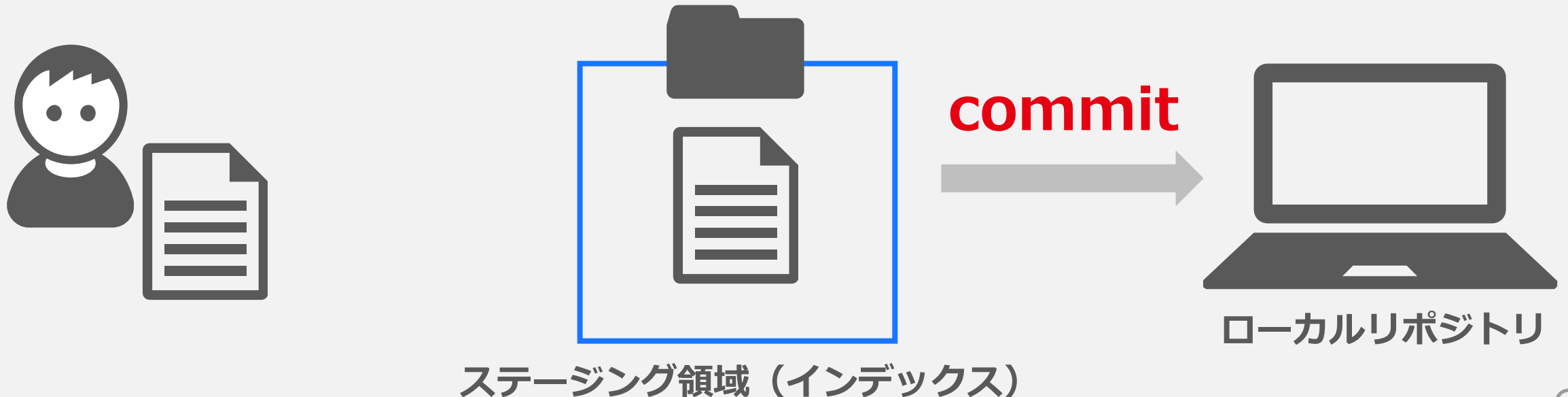
これらの情報を含んだファイルは add しない
ように注意しましょう

**VSCodeで
commitする**

ファイルを修正してGitで扱う

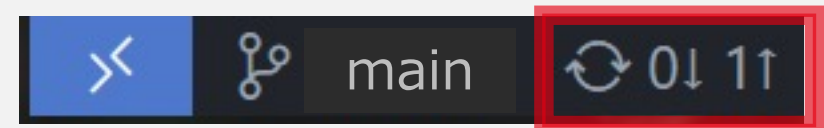
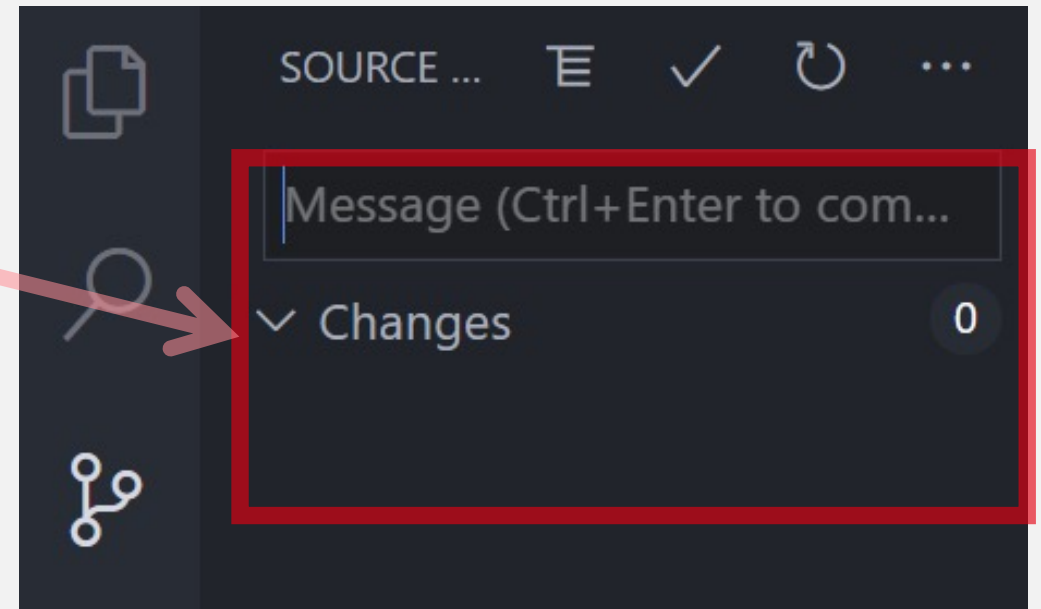
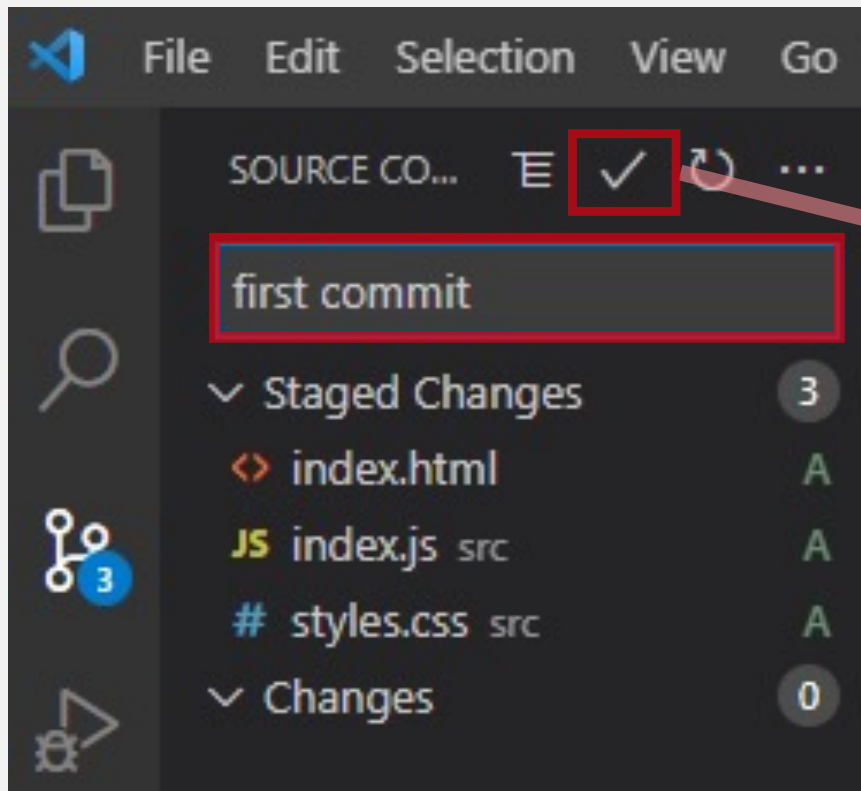
Step.2

ステージング領域に登録した変更内容をローカルリポジトリに登録する → **commit** という




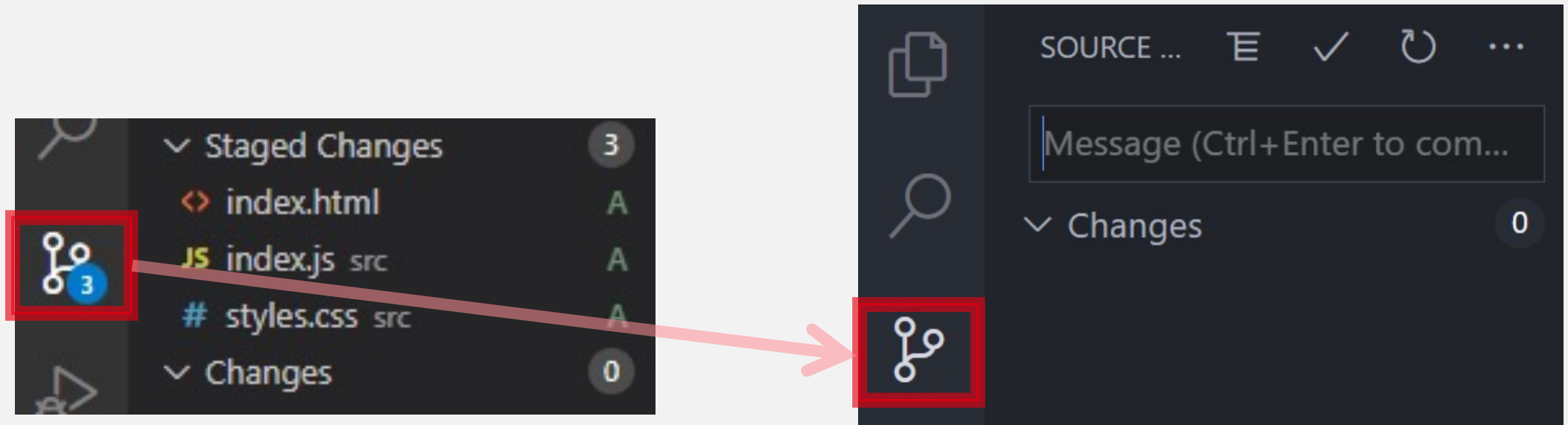
ローカルリポジトリにcommitする

- 任意のメッセージを入力し✓をクリックすると表示が更新されて画面下が「0↓1↑」となります



ローカルリポジトリにcommitする

- OSの違いやGit設定によっては画面下が「0↓0↑」のままである場合があります
- その場合は  の数字が消えていればOKです

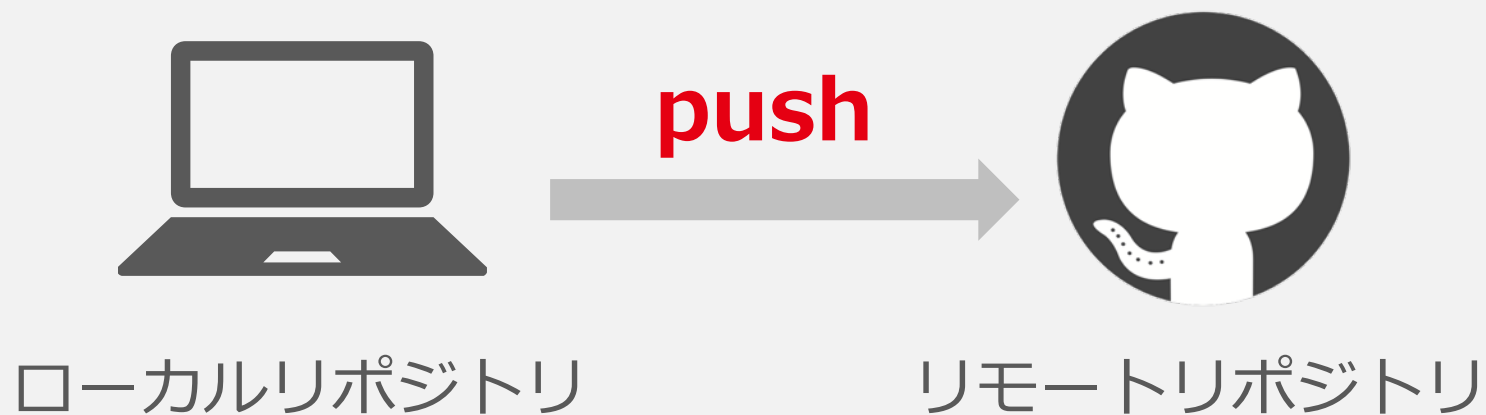


**VSCodeで
push する**

自分のPCからGitHubに反映する

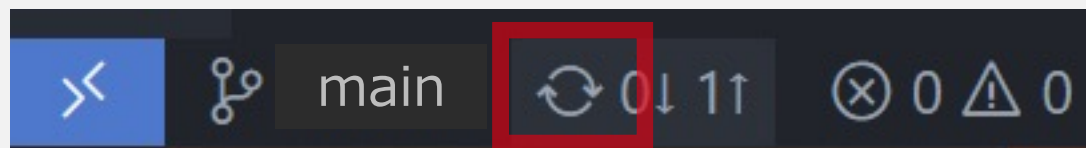
push :

ローカルリポジトリの変更をリモートリポジトリに反映



GitHubへの疎通確認（push）

- 更新ボタン（赤枠のところ）を押すとpushされます

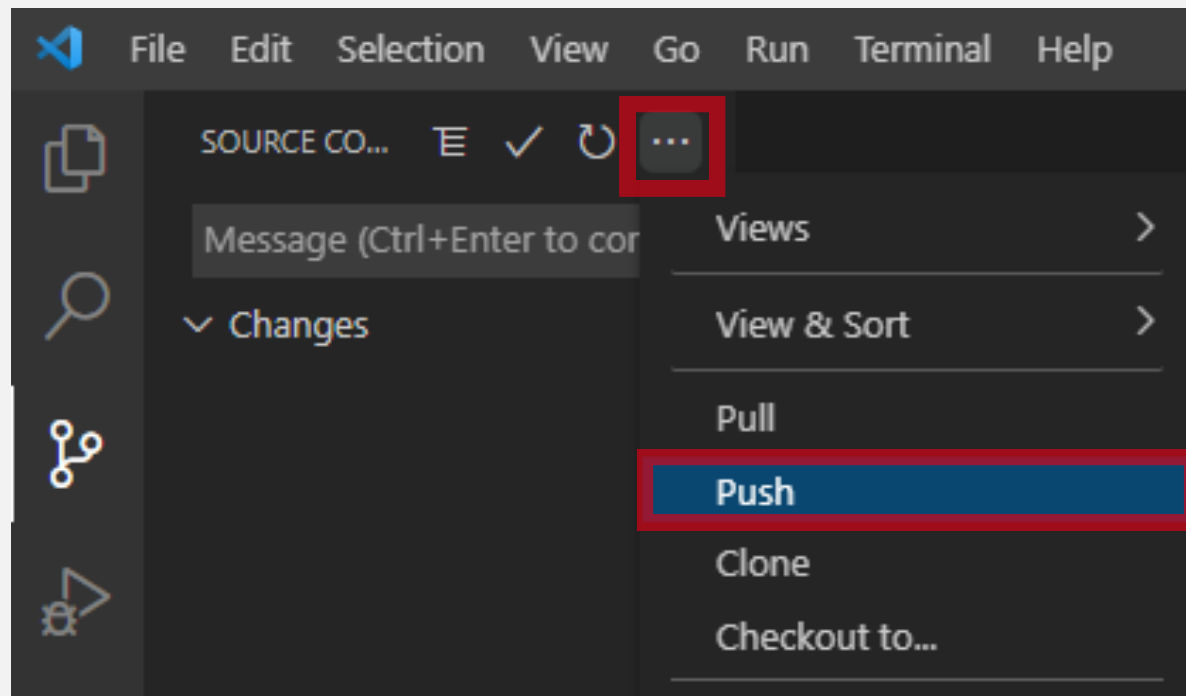


- 更新ボタンは押すことでpushの他にpullも行うことができます

pullができるときは「1↓0↑」のように表示されます（状況により表示されない場合もあります）

GitHubへの疎通確認（push） 補足

- 更新ボタンでのpushがうまく行かないときは、
下図の「…」から「Push」を選択してください



GitHubに反映されているか確認

The screenshot shows the GitHub interface for a repository named 'github_ws' under the username 'ユーザー名'. The top navigation bar includes links for Pull requests, Issues, Marketplace, and Explore. The repository header shows 1 Unwatch, 0 Stars, and 0 Forks. The main content area displays the 'Code' tab with a commit history table. The commit table has three rows: a commit by 'ユーザー名' adding 'index.html & src' 14 minutes ago with 1 commit, and two file-level commits for 'src' and 'index.html' also adding 'index.html & src' 14 minutes ago. A green 'Add a README' button is visible at the bottom of the commit list. The right sidebar contains the 'About' section (no description) and the 'Releases' section (no releases published).

Search or jump to... Pull requests Issues Marketplace Explore

ユーザー名 / github_ws Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

master 1 branch 0 tags Go to file Add file Code

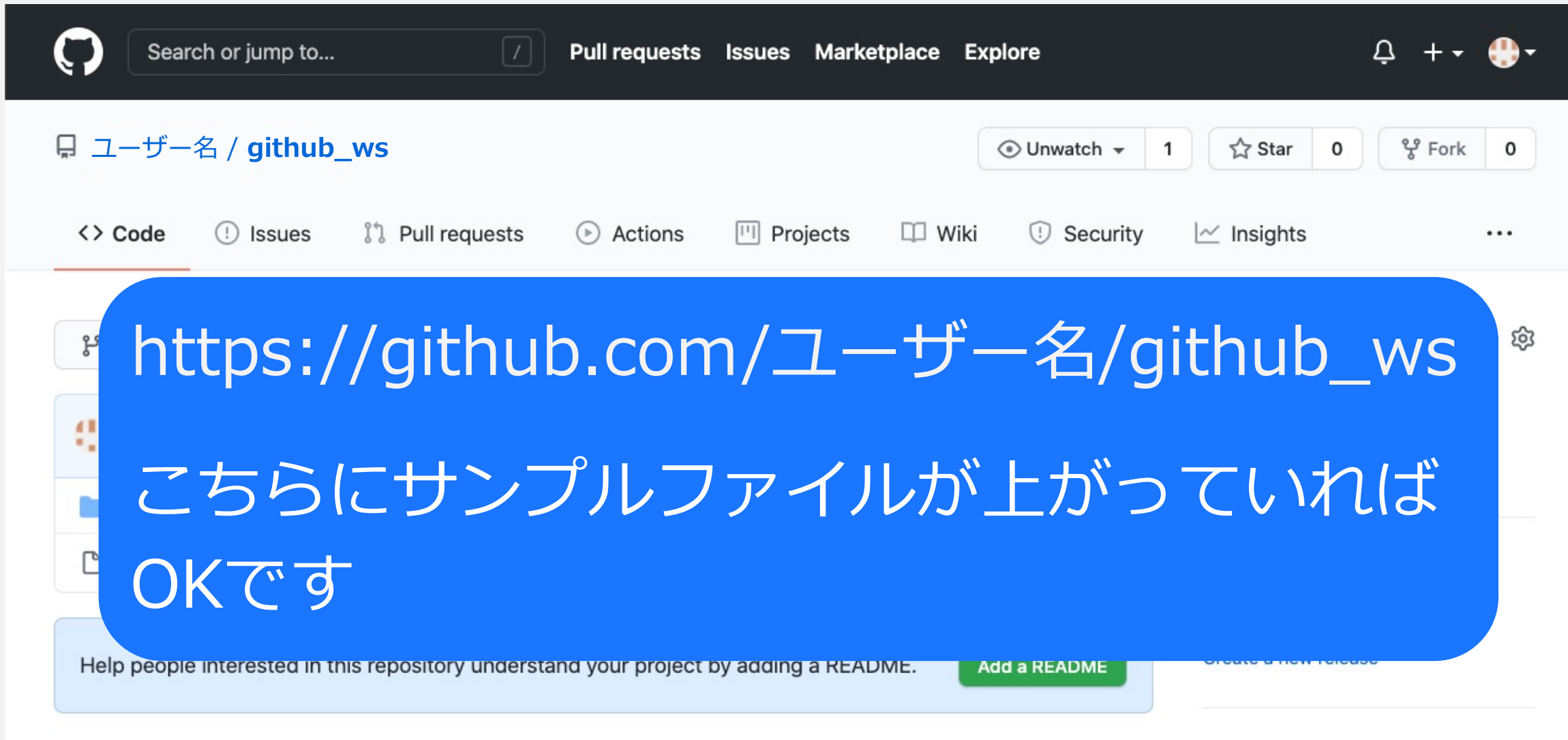
ユーザー名	Add index.html & src	50cdc45 14 minutes ago	🕒 1 commits
src	Add index.html & src	14 minutes ago	
index.html	Add index.html & src	14 minutes ago	

Help people interested in this repository understand your project by adding a README. Add a README

About No description, website, or topics provided.

Releases No releases published Create a new release

GitHubに反映されているか確認



The screenshot shows the GitHub interface for a repository named 'github_ws' owned by a user named 'ユーザー名'. The repository has 1 watch, 0 stars, and 0 forks. The 'Code' tab is selected, and a blue overlay box contains the URL 'https://github.com/ユーザー名/github_ws' and the text 'こちらにサンプルファイルが上がっていればOKです' (If sample files are uploaded here, it's OK). Below the overlay, there is a prompt to 'Add a README'.

GitHub repository page for `github_ws` by `ユーザー名`.

URL: https://github.com/ユーザー名/github_ws

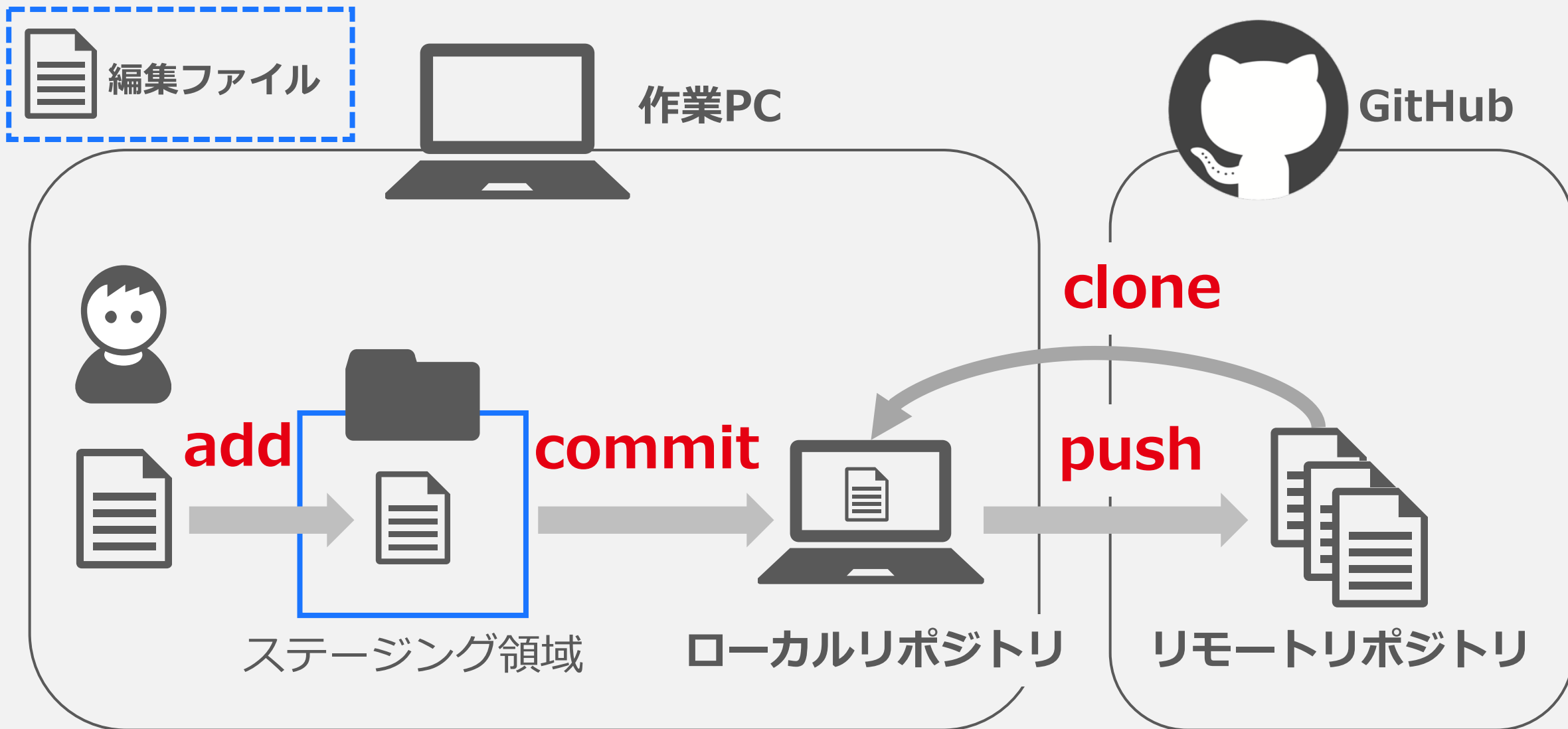
こちらにサンプルファイルが上がっていればOKです

Help people interested in this repository understand your project by adding a README. [Add a README](#)

Pushがうまくいかない時①

- ネットワークの問題
 - 学校のWifiなどを使用していると、ネットワークの設定で制限がかけられ、Pushができない場合があります。
 - 別のネットワークやスマホのテザリングなどで再度試してみましょう

基本操作まとめ



セットアップ資料は以上です

Appendix


VSCodeライブラリ

- **Japanese Language Pack for Visual Studio Code**
 - VSCodeの日本語化
- **Auto Rename Tag**
 - HTMLの開始タグと終了タグを同時に修正
- **Beautify**
 - コード整形
- **VSLiveShare**
 - 複数人でコードの編集・デバッグが行える拡張機能

Gitコマンドでの操作

clone

- 適当なディレクトリに移動して、← この場所がローカルリポジトリ
- 「git clone https://github.com/ユーザー名/github_ws.git」

A terminal window with a dark background and light gray text. The prompt is '~ \$' and the command being entered is 'git clone https://github.com/ユーザー名/github_ws.git'. The window has three colored window control buttons (red, yellow, green) in the top left corner.

```
~ $ git clone https://github.com/ユーザー名/github_ws.git
```

- 「git」や「workspace」などのディレクトリ下に clone するとリポジトリが管理しやすいです

add

- addする前に「git status」で変更されたファイルを確認できます。

```
~/github/github_ws $ git status
On branch master
Your branch is up to date with 'origin/main' .

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    index.html
    src/

nothing added to commit but untracked files present (use "git add" to track)
```

add

- 「git add .」 (最後にピリオドが必要です) で addされます。「git status」でファイルごとの変更内容が出ていることを確認してください。

```
~/github/github_ws $ git add .  
~/github/github_ws $ git status  
On branch master  
Your branch is up to date with 'origin/main'.  
  
Changes to be committed:  
  (use "git reset HEAD <file>..." to unstage)  
  
    new file:   index.html  
    new file:   src/index.js  
    new file:   src/styles.css
```

ファイルごとの
変更内容が表示され
ます

commit

- 「git commit -m "<任意のメッセージ>"」でコミットされます。
- commit後は「git status」すると変更分は消えていることが確認できます。

```
[~/github/github_ws $ git commit -m "first commit"]
[main d55fbdf] first commit
3 files changed, 50 insertions(+)
create mode 100644 index.html
create mode 100644 src/index.js
create mode 100644 src/styles.css
```

push

- 「git push origin main」でpushします
(ブランチがmainの場合はmaster -> main。
commit時の表示を確認してください)
- GitHubのユーザー名とパスワードを入力する必要があります。

```
~/github/github_ws $ git push origin main
```