

# GitHub 入門編

## vSCode版

Hack U Project

# Hack U について

Hack U とは

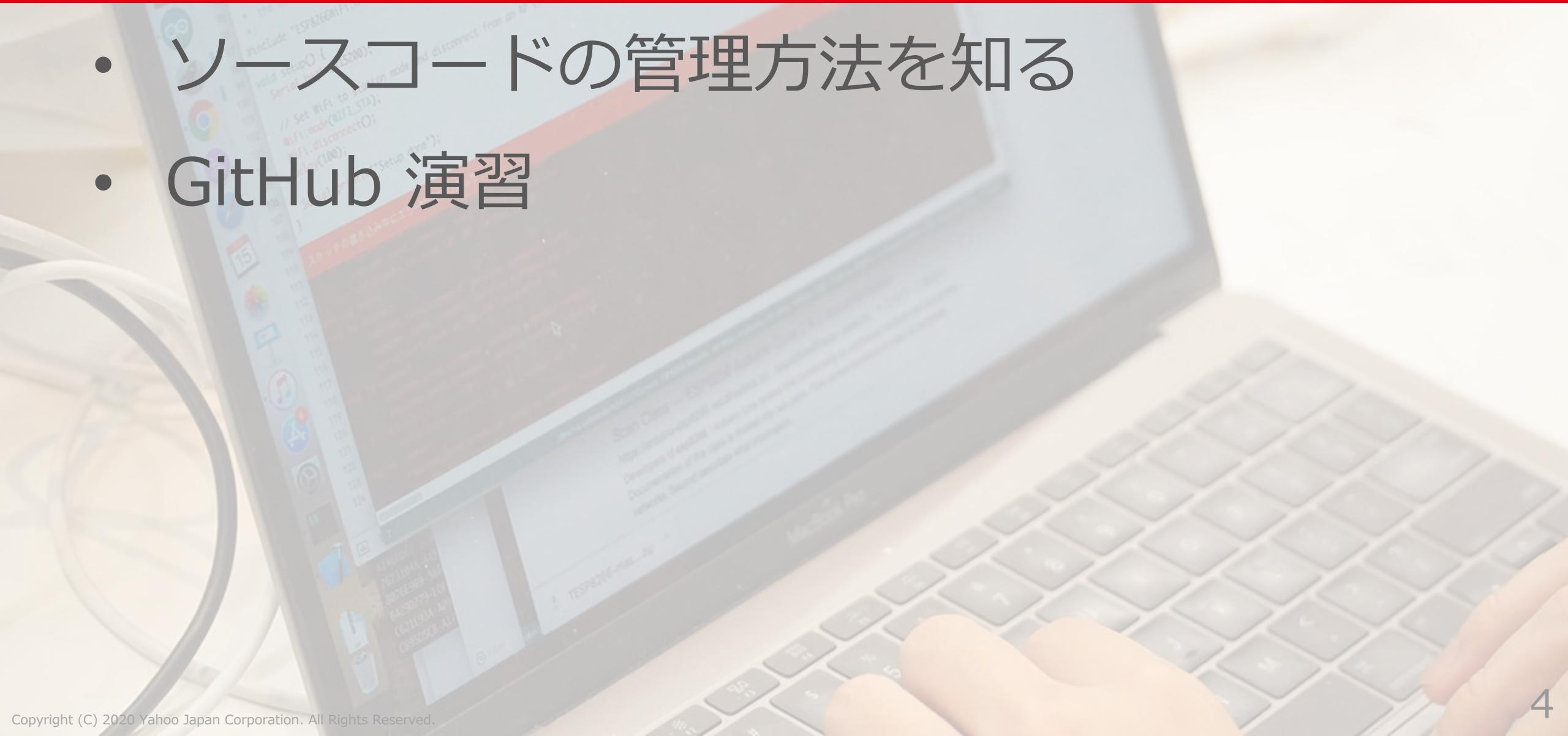
「学生からクリエイターになるきっかけ」となる場作りをし、  
「ものづくりの楽しさ」を伝えるヤフー株式会社の取り組みです。

毎年、学生を対象としたハッカソンを開催しています。  
また、初心者歓迎の講座イベントも開催しています。  
(今回のイベントはこちら!)

ものづくりが初心者の方も、今回のような講座をきっかけに  
ぜひハッカソンへの出場を考えてみてください！

# イベントで学ぶこと

- ・ソースコードの管理方法を知る
- ・GitHub 演習



# このような経験をしていませんか？

よし、先週の開発の続きをやるぞ！



index  
.html



index(1)  
.html



最新版  
index.html



index\_final  
.html

最後に編集したファイルどれだっけ？



# このような経験をしていませんか？



こここの機能開発しておいたよ。  
ソースコードはメールで添付して送るから取り込んでね。

ありがとう！早速コピペして・・・  
あ！自分が作ってた部分が上書きされちゃった



# このような経験をしていませんか？



こここの機能開発しておいたよ。

Git / GitHub を使って  
解決しましょう！

ありが

あ！自分が作ってた部分が上書きされちゃった

んでね。



# このイベントのゴール

- ・ コード管理の大切さを体験してもらう
- ・ 自分がつくったものをWebに公開できるようになってもらう
- ・ オープンソースを探しだして最先端の技術を見つけられるようになつてもらう

# イベントの進め方

- ・ セットアップの確認
- ・ 各機能の解説
- ・ ツールの使い方
- ・ ハンズオン
- ・ まとめ
- ・ 懇親会

セットアップ  
お済みですか？

YAHOO!  
JAPAN

# GitHubへの疎通確認

The screenshot shows a GitHub repository page for the user 'github\_ws'. The repository name is 'github\_ws'. The 'Code' tab is selected. The commit history shows a single commit from 'github\_ws' adding 'index.html & src' files. A large blue callout box contains the Japanese text 'add/commit/pushを行い 演習ファイルが上がつていればOK' (After performing add/commit/push, if the exercise file has been uploaded, it's OK). The bottom right corner of the callout box has a small white arrow pointing up and to the left.

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

ユーザー名 / [github\\_ws](#)

<> Code Issues Pull requests A

main 1 branch 0 tags

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](#)

topics provided.

Releases

No releases published [Create a new release](#)

add/commit/pushを行い  
演習ファイルが上がつていればOK

# Gitとは



# 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

## リポジトリ内のバージョン管理



# バージョン管理とは

過去の状態との差分がわかる



# バージョン管理とは

過去の状態に戻せる

過去の状態を最新として扱う



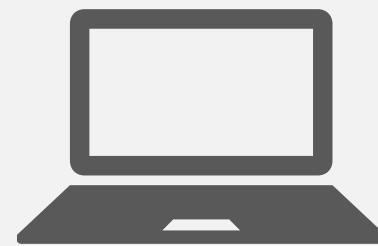
# バージョン管理とは

- **リモートリポジトリ :**

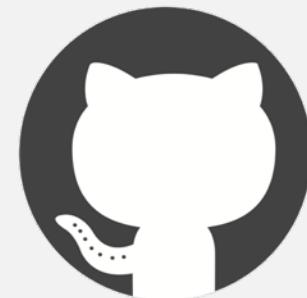
色々な人が共有できるようにしたリポジトリ  
GitHub上にあるものはリモートリポジトリ

- **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
GitHub	Gitでバージョン管理をしやすくする <b>Webサービス</b> ブラウザから変更履歴やコードを共有するのが簡単にできる

# 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)

# 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)

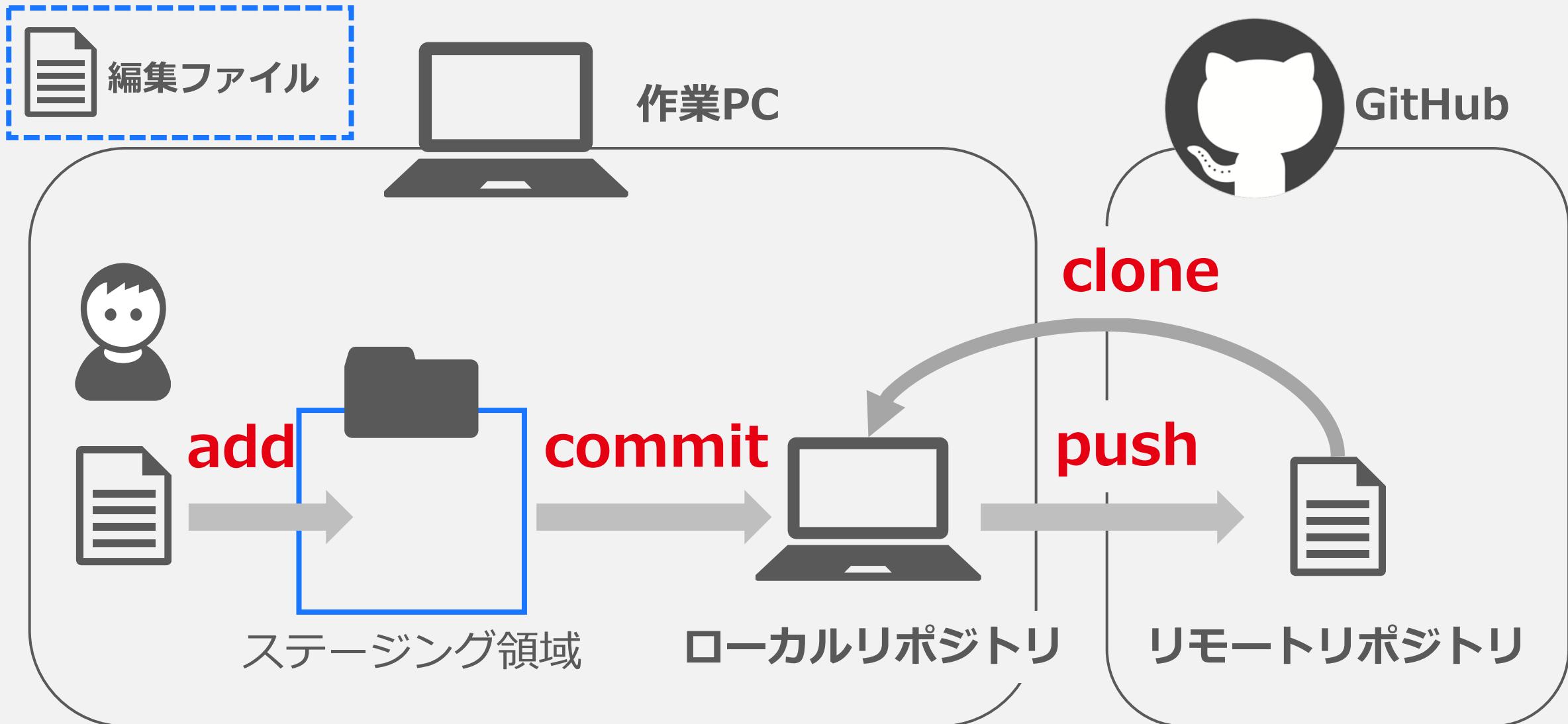
今回は VSCode で  
Gitコマンド を使います

# Step.1

# GitHubの基本操作



# GitHubの基本操作

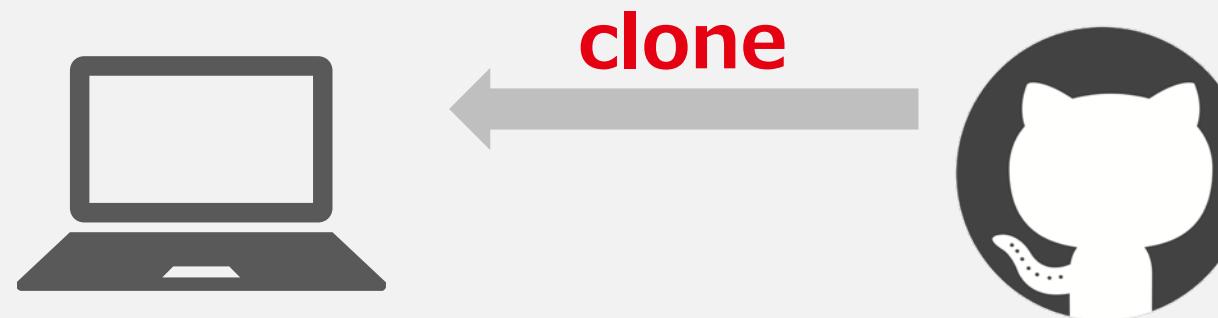


# GitHubから自分のPCに持ってくる

- **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

※ 初回のみの操作

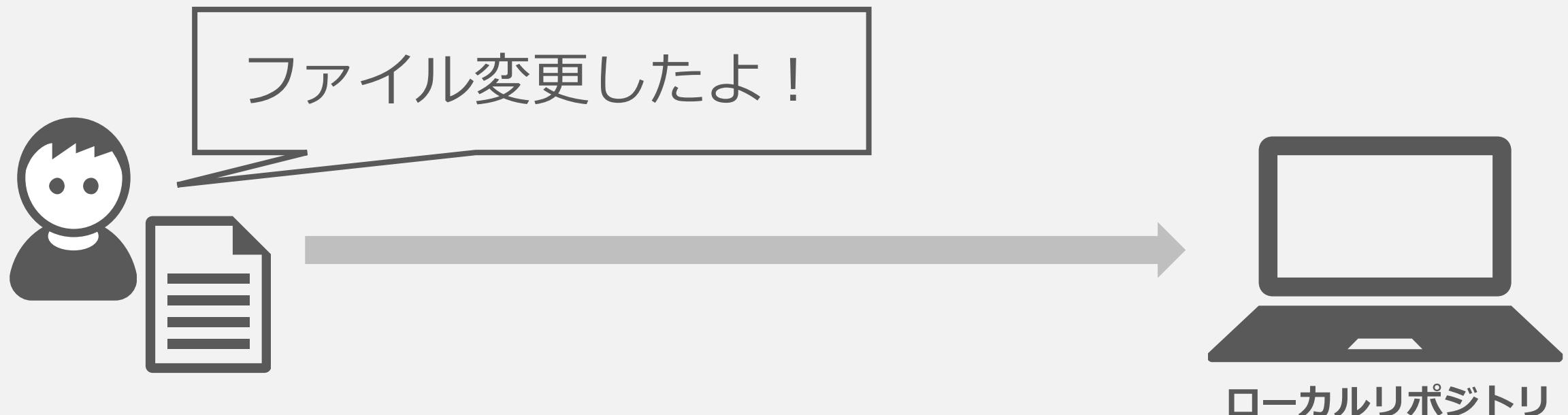


ローカルリポジトリ

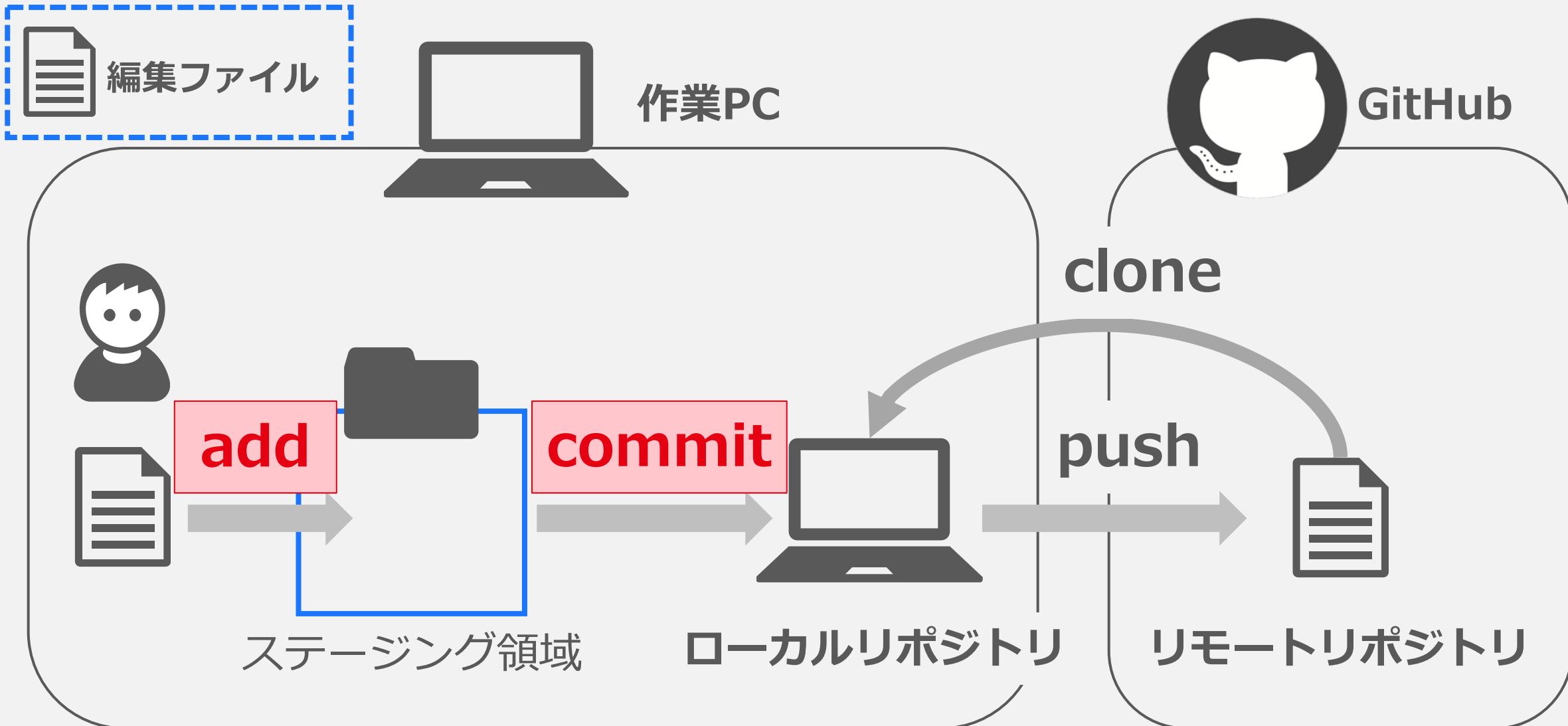
リモートリポジトリ

# ファイルを修正してGitで扱う

ファイルの変更をローカルリポジトリに  
登録するには**2つのステップ**が必要になります



# GitHub の基本操作



# ファイルを修正してGitで扱う

## Step.1

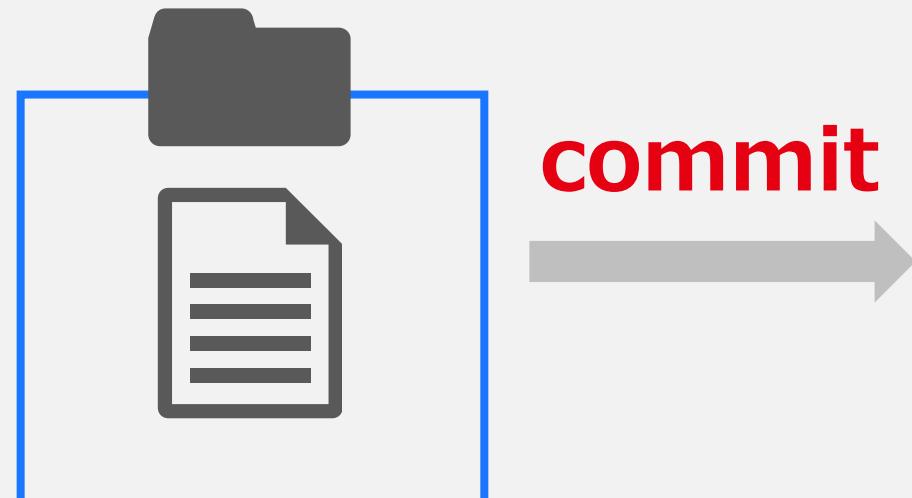
どのファイルの変更をしたかをステージング領域に  
一度登録する → **add** という



# 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



ステージング領域（インデックス）



ローカルリポジトリ

# ファイルを修正してGitで扱う

## Step.2の注意点

commitした時には必ず**どういう変更をしたかを  
コメント**で残す必要がある



ステージング領域（インデックス）

commit  
→

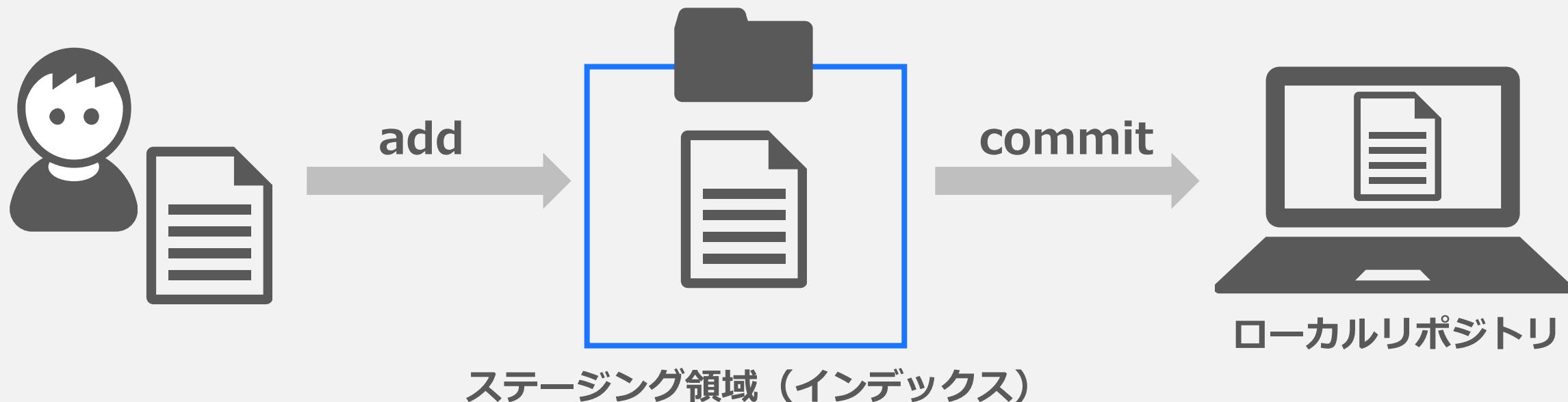
Commitコメント



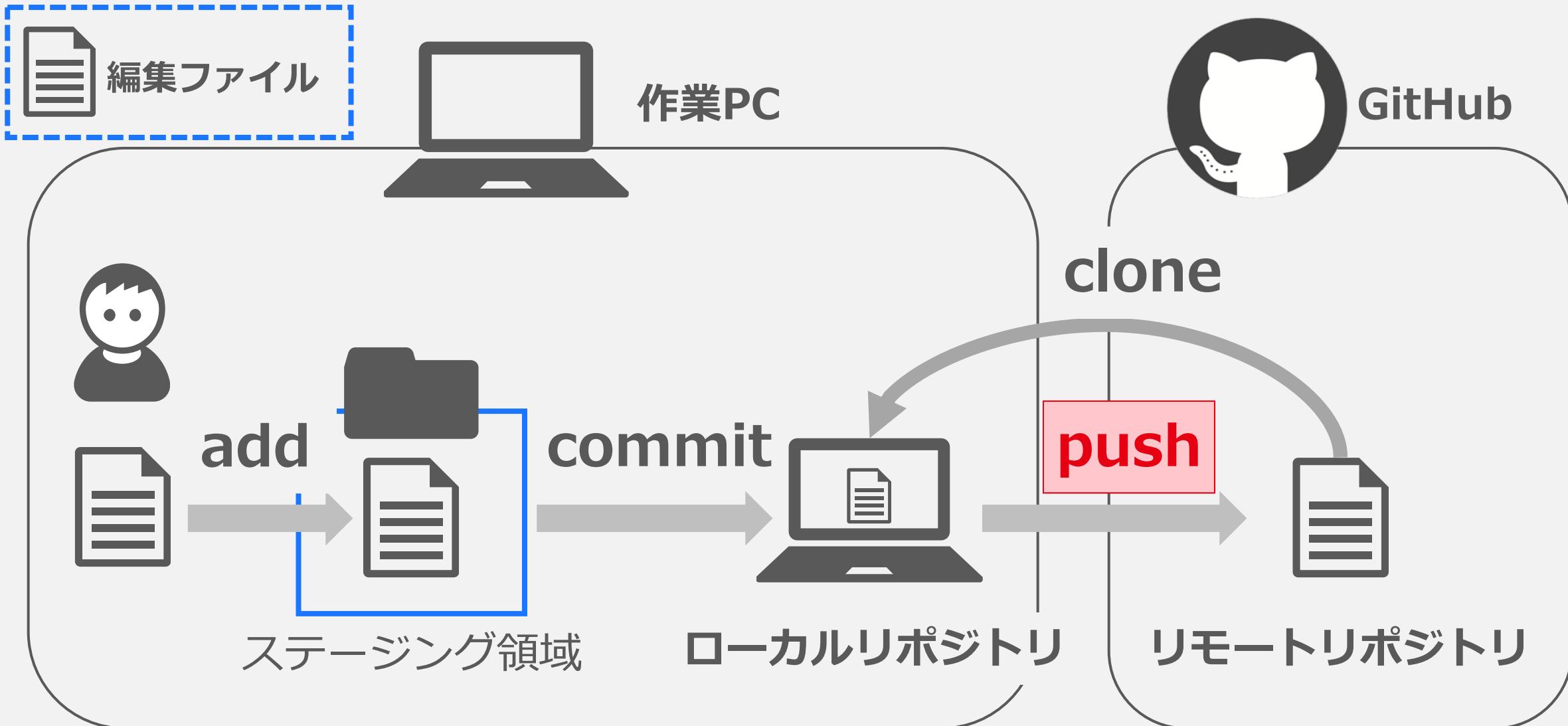
ローカルリポジトリ

# ファイルを修正してGitで扱う

- ファイルの変更は**add**して**commit**
- commitした時は**必ずコメントを残す**



# GitHub の基本操作



# 作業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



ローカルリポジトリ

リモートリポジトリ

VSCodeで  
cloneする

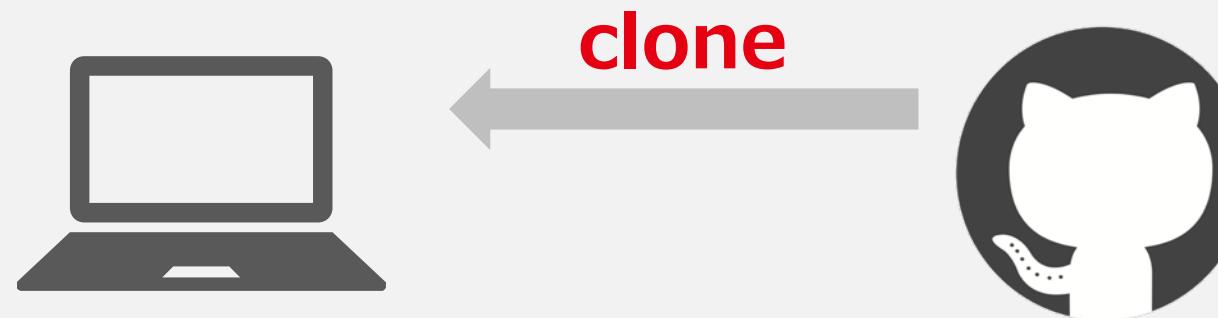


# GitHub から自分のPCに持ってくる

- **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

※ 初回のみの操作

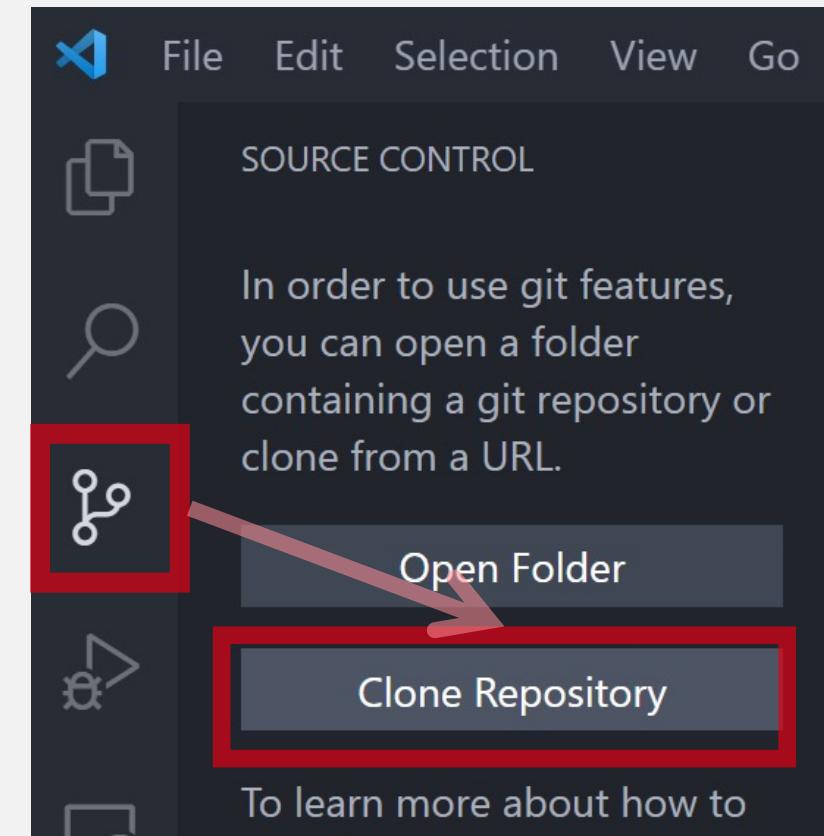
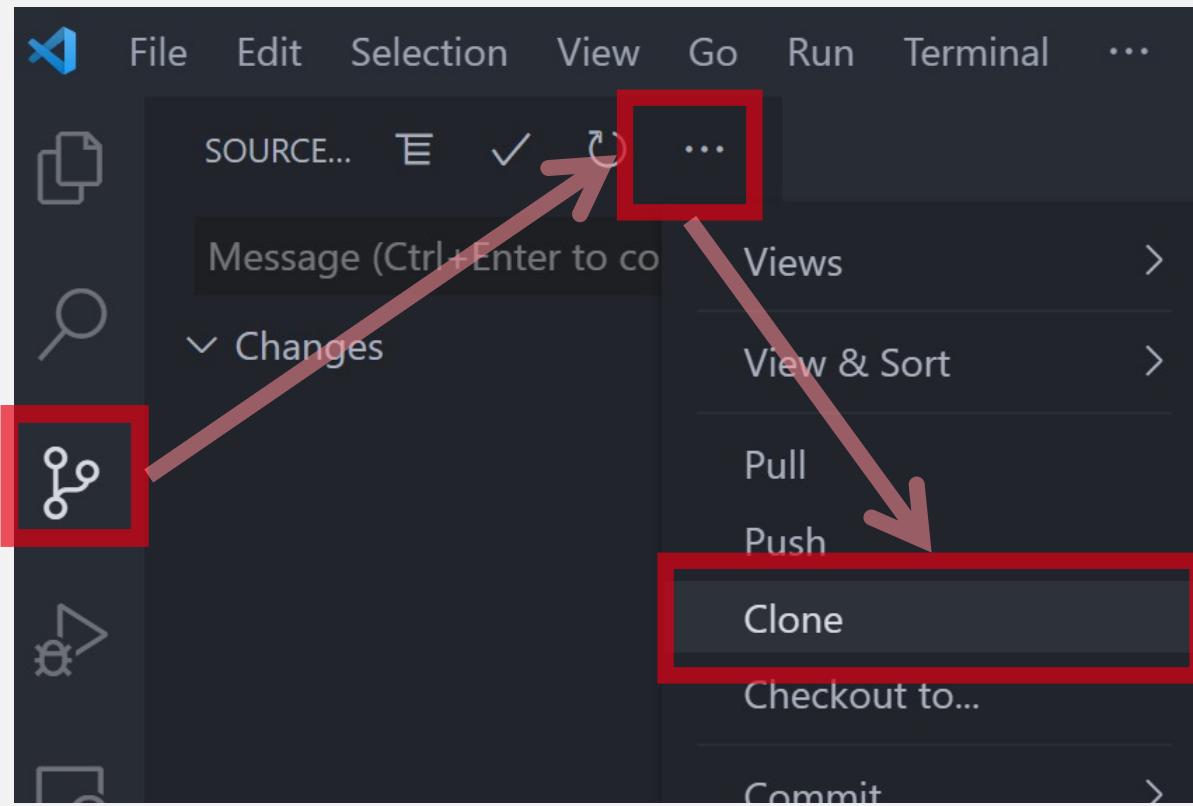


ローカルリポジトリ

リモートリポジトリ

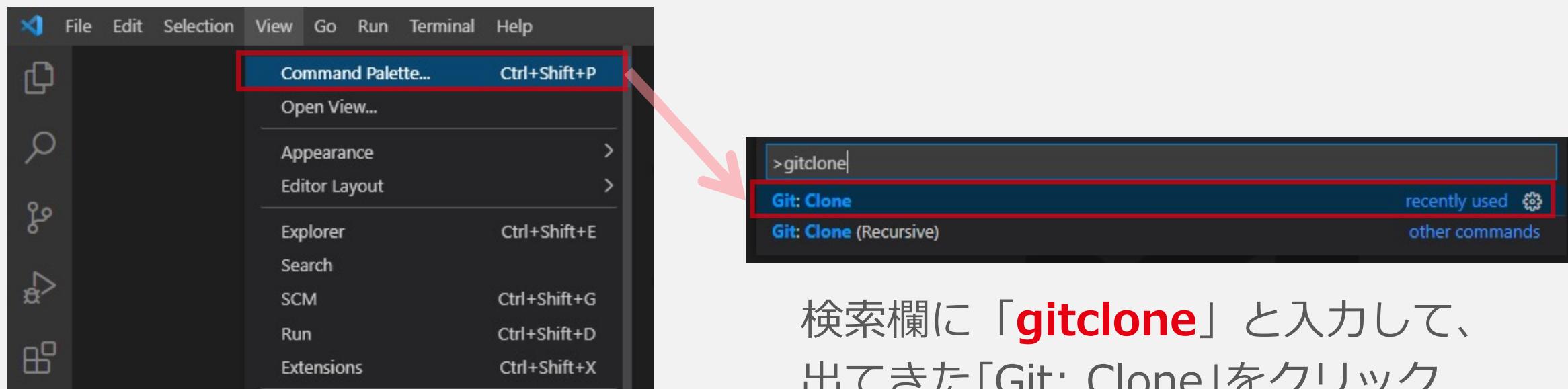
# clone ①

- 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

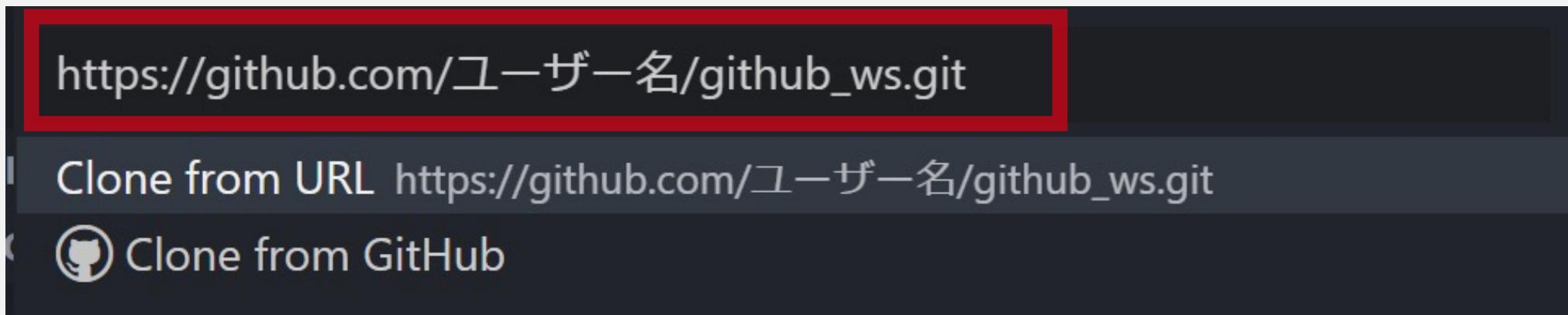
- VSCodeのUIが異なる際は、「Command Palett...」から検索します



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

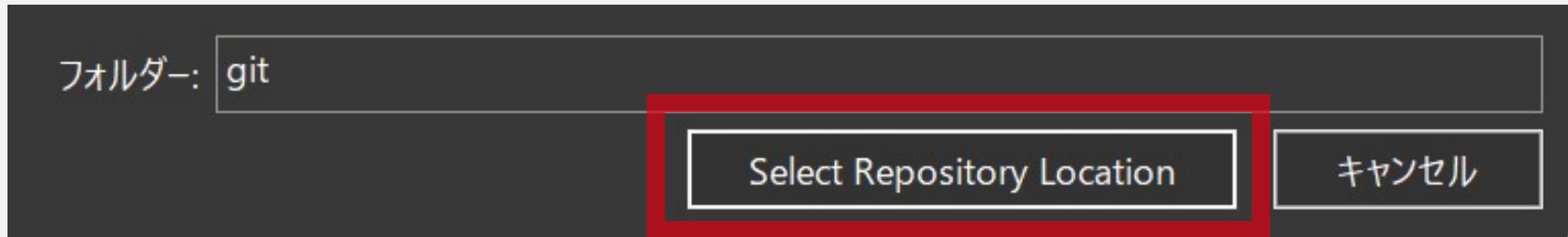
# clone ②

- 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](https://github.com/ユーザー名/github_ws.git)」** を入力し、Enterを押します

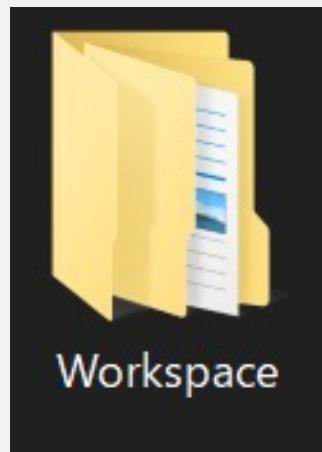
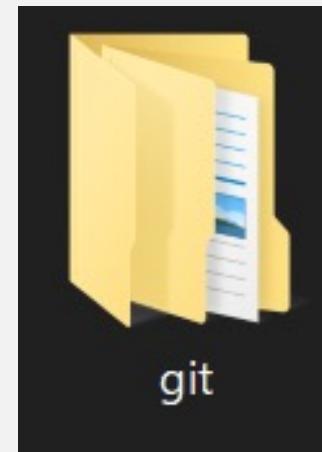


# clone ③

- 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

The screenshot shows a Visual Studio Code interface. A modal dialog box is open in the center, asking "Would you like to open the cloned repository?". It has two buttons: "Open" (highlighted with a red box) and "Open in New Window". Below the dialog, the main VS Code window displays the "EXPLORER" view. The "OPEN EDITORS" section shows a folder named "GITHUB\_WS-1" containing files "src" and "index.html". The status bar at the bottom shows the current branch is "master". A large red arrow points from the text "Openを押したら Clone完了です" (Press Open to finish cloning) down to the "Open" button in the dialog.

① Would you like to open the cloned repository?

Source: Git (Extension)

Open      Open in New Window

- Openを押したら  
Clone完了です

EXPLORER

OPEN EDITORS

GITHUB\_WS-1

src

index.html

Show All Commands  $\text{Ctrl} + \text{Shift} + \text{P}$

Go to File  $\text{Ctrl} + \text{P}$

Find in Files  $\text{Ctrl} + \text{Shift} + \text{F}$

Start Debugging  $\text{F5}$

Toggle Terminal  $\text{Ctrl} + @$

master

Live Share

VSCodeで  
addする



# ファイルを修正してGitで扱う

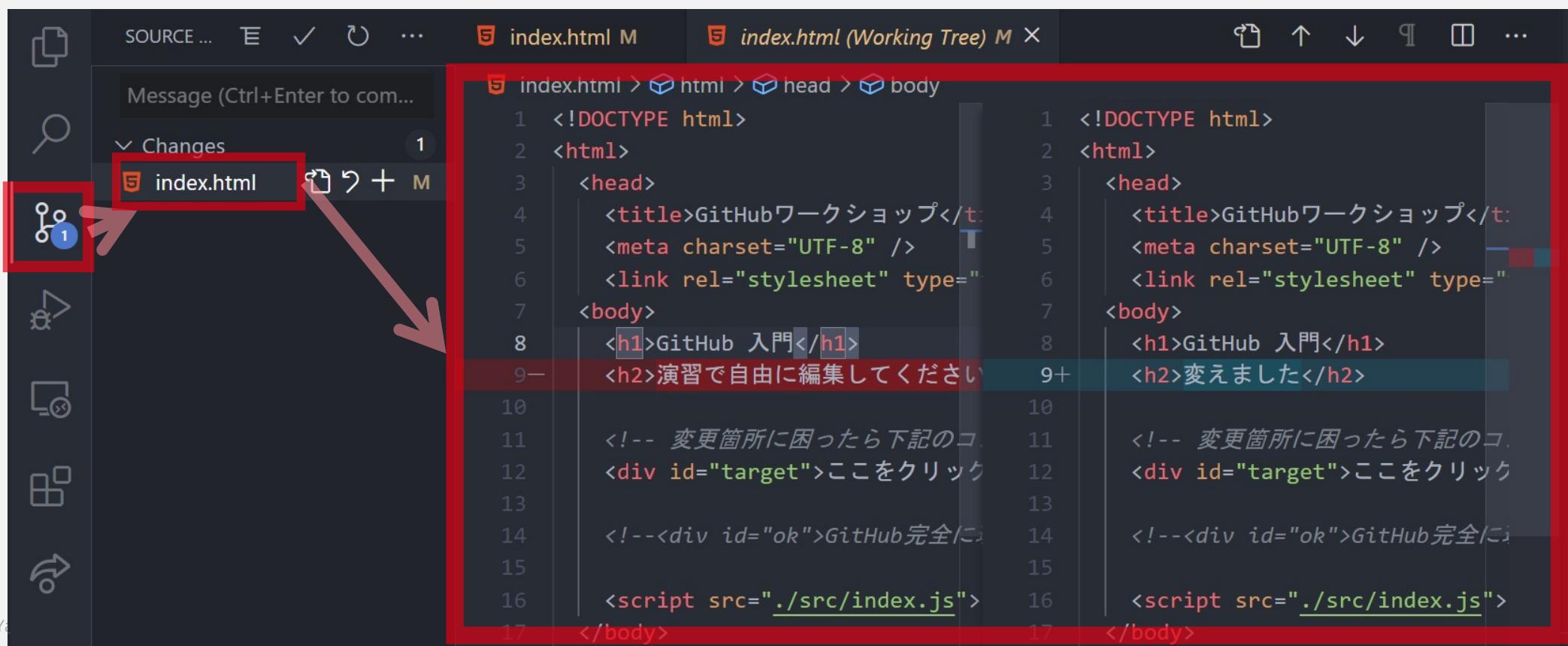
## Step.1

どのファイルの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



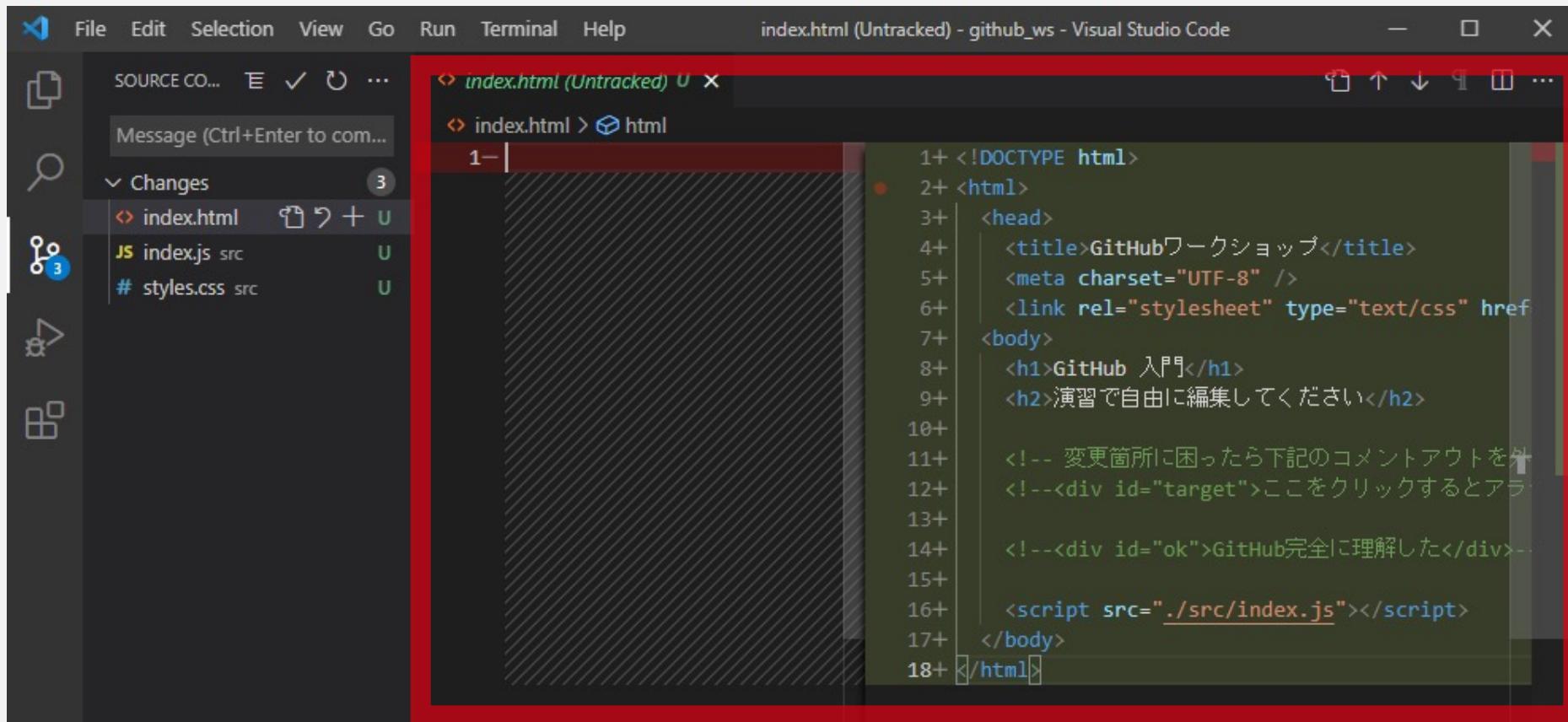
# ステージングにaddする

- addする前にVSCodeのメニューから変更内容を確認できます



# ステージングにaddする

- 新規ファイルをaddする際は下記のようになります



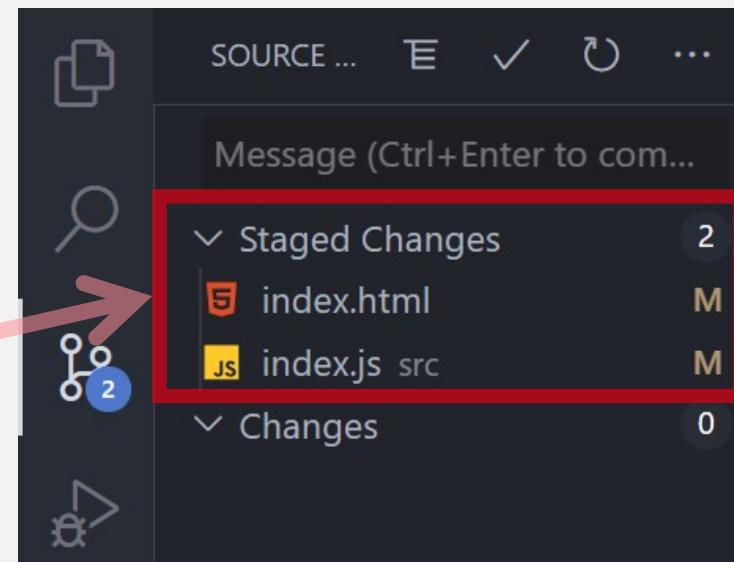
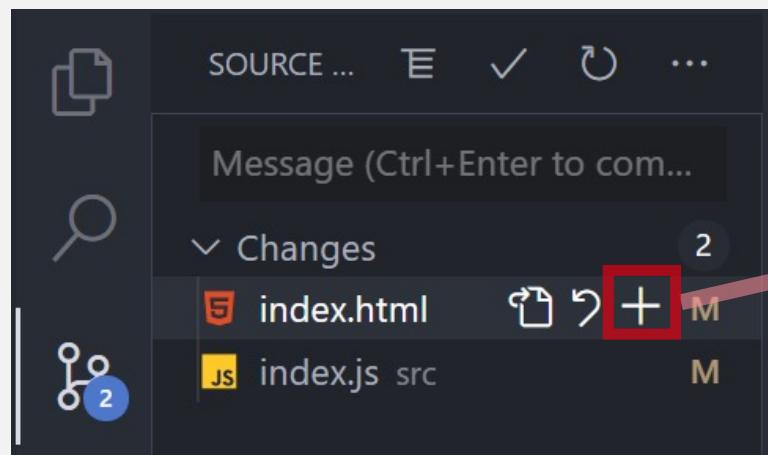
The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** index.html (Untracked) - github\_ws - Visual Studio Code.
- Sidebar:** Shows a tree view with "Changes" expanded, listing "index.html" (Untracked), "index.js src" (Untracked), and "# styles.css src" (Untracked). A red box highlights this sidebar area.
- Editor Area:** The main editor shows the content of index.html. The code is displayed with line numbers on the left. A red box highlights the entire editor area.

```
1+ <!DOCTYPE html>
2+ <html>
3+   <head>
4+     <title>GitHubワークショップ</title>
5+     <meta charset="UTF-8" />
6+     <link rel="stylesheet" type="text/css" href="styles.css" />
7+   <body>
8+     <h1>GitHub 入門</h1>
9+     <h2>演習で自由に編集してください</h2>
10+
11+    <!-- 変更箇所に困ったら下記のコメントアウトを外す -->
12+    <!--<div id="target">ここをクリックするとアラートが表示されるよ！</div>-->
13+
14+    <!--<div id="ok">GitHub完全に理解した</div>-->
15+
16+    <script src=".src/index.js"></script>
17+  </body>
18+ </html>
```

# 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。  
addするファイルを全て「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

YAHOO!  
JAPAN

# 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



**commit**

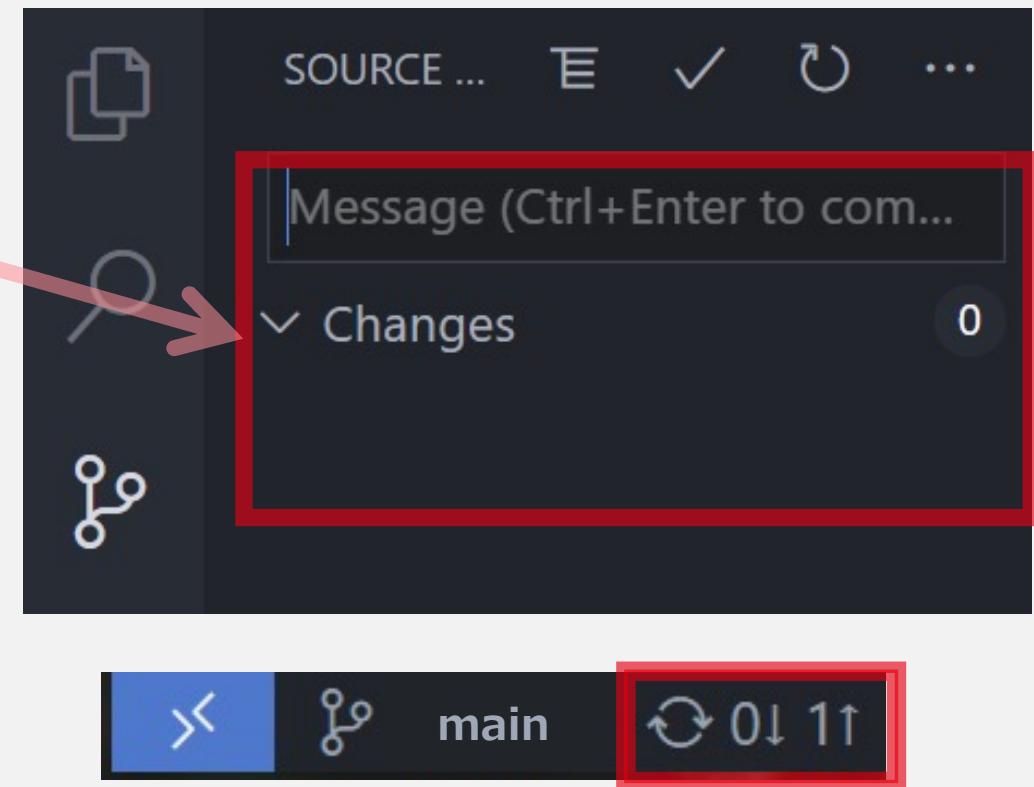
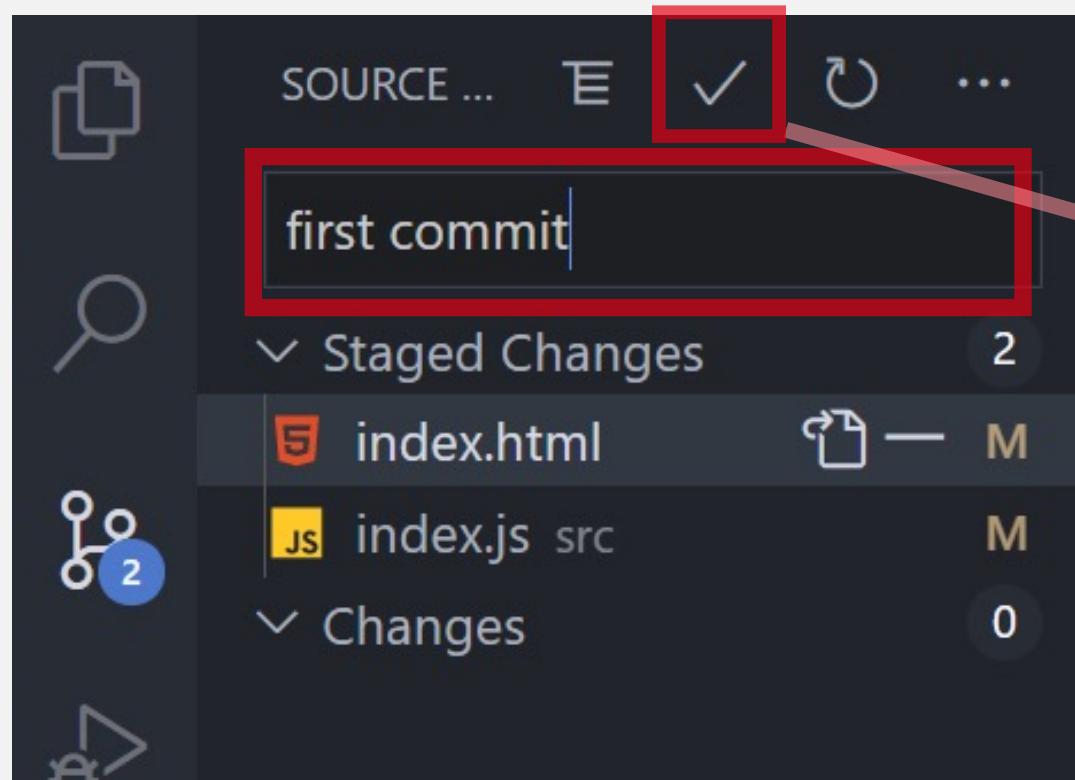


ローカルリポジトリ

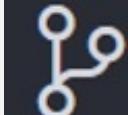
ステージング領域（インデックス）

# 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「 $0 \downarrow 1 \uparrow$ 」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

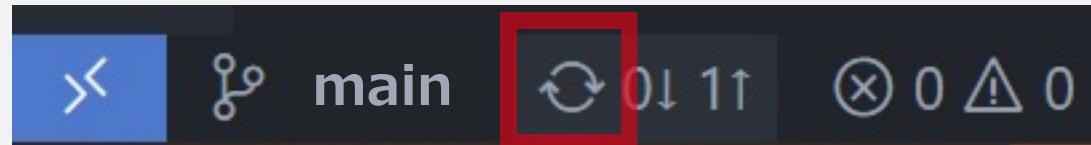


ローカルリポジトリ

リモートリポジトリ

# GitHubへの疎通確認 (push)

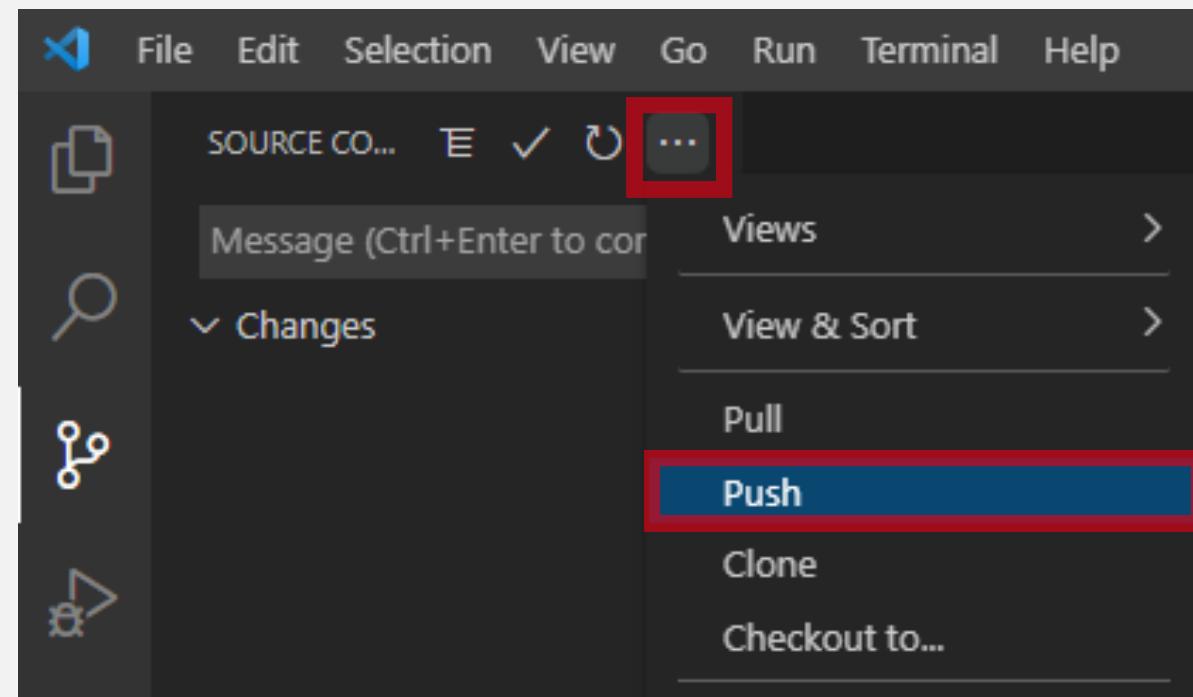
- 更新ボタン（赤枠のところ）を押すとpushされます



- 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「 $1\downarrow 0\uparrow$ 」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 a GitHub repository page for a user named 'github\_ws'. The repository has 1 branch and 0 tags. A recent commit was made 14 minutes ago, adding 'index.html & src' to the 'src' directory. The commit hash is 50cdc45. The repository has 1 commit in total. There is a note at the bottom encouraging the addition of a README file.

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

Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights ...

main 1 branch 0 tags Go to file Add file Code

About No description, website, or topics provided.

Releases No releases published Create a new release

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

# GitHubに反映されているか確認

The screenshot shows a GitHub repository page for a user named 'github\_ws'. The page includes a navigation bar with links for Pull requests, Issues, Marketplace, and Explore. Below the navigation bar, there are buttons for Unwatch (1), Star (0), and Fork (0). The main content area displays tabs for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. A large blue callout box is overlaid on the page, containing the URL [https://github.com/ユーザー名/github\\_ws](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 a sample file is uploaded here, it's OK). At the bottom of the callout box, there is a note to 'Add a README'.

https://github.com/ユーザー名/github\_ws

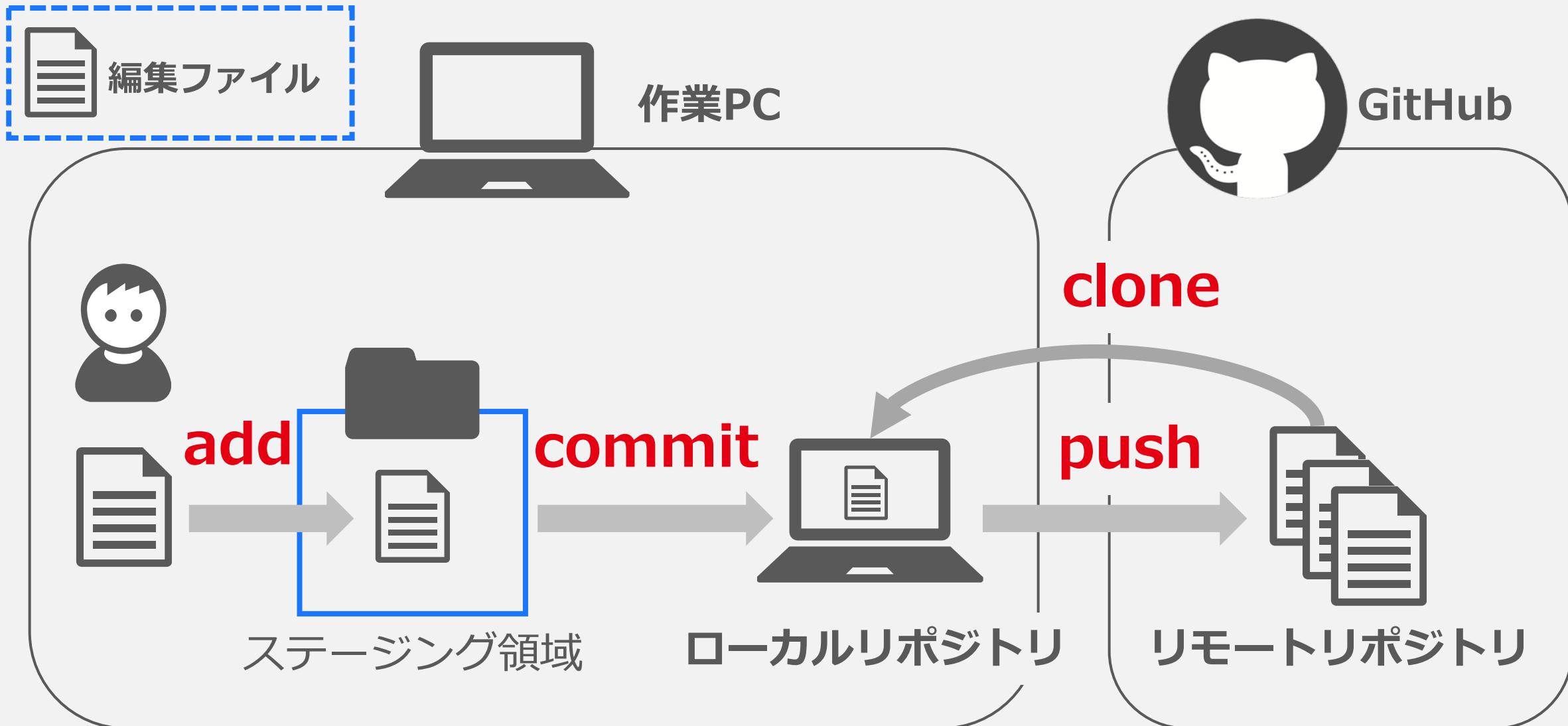
こちらにサンプルファイルが上がっていれば  
OKです

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



# Step.1

## ハンズオン



# ハンズオンの作業内容 その1

- ファイルを追加/変更して、リモートにpush
1. README.mdの新規作成 / index.htmlを編集する
  2. 作成/変更したファイルをaddする
    - ファイルの差分を確認しよう

# ハンズオンの作業内容 その2

3. インデックスされたファイルをcommitしてみよう
  - コミットコメントは変更内容がわかるように記述しよう
4. ローカルリポジトリの変更をpushしてみよう
5. リモートリポジトリを確認してみよう

# README.md とは

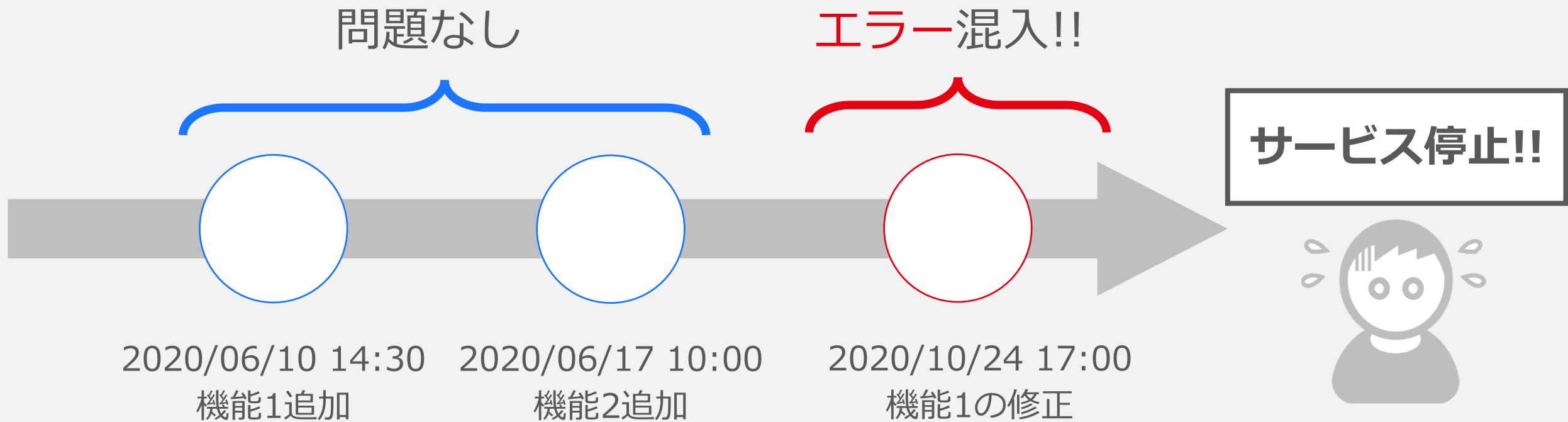
- **リポジトリで最初に見るべき資料**
  - 使用方法
  - 開発ルール
  - 変更履歴 ...など
- **自分のリポジトリの説明を追加しましょう**
  - Markdown記法で記述できます

# Step.2-1

## 作業用ブランチの作成

# ブランチ

リポジトリに動かないコードが上がってしまう



# ブランチ

リポジトリに動かないコードが上がってしまう

既存のコードにバグを含まないよう

作業環境を分けましょう

機能1追加

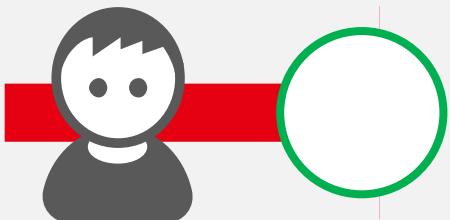
機能2追加

機能1の修正

止!!

# ブランチ

最新の状態



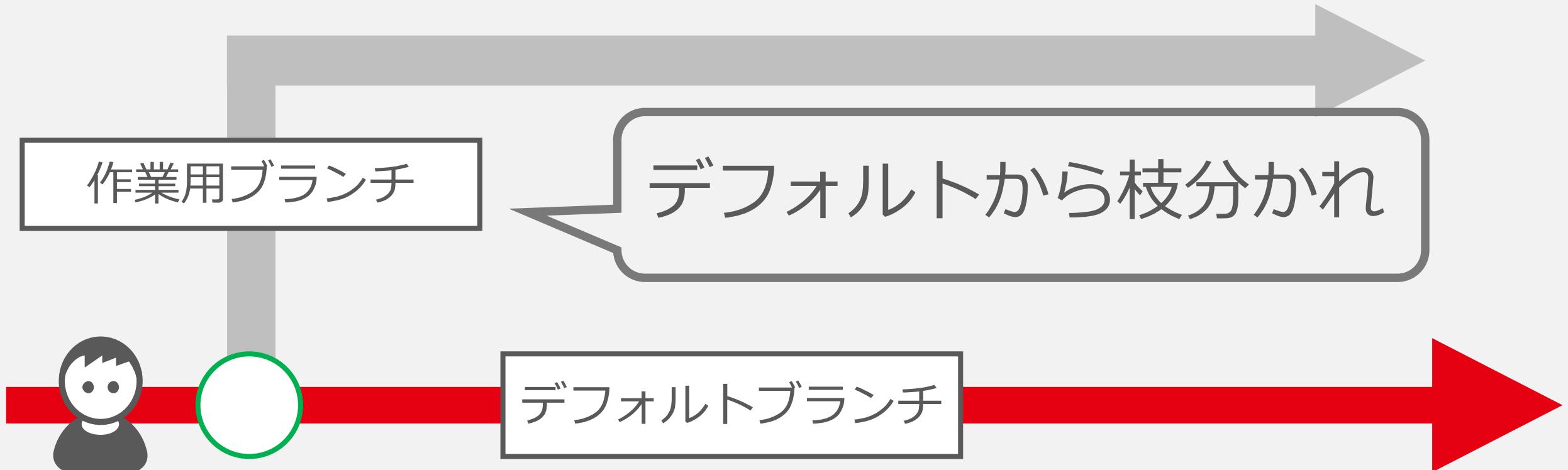
大元のブランチ

→ **デフォルトブランチ**

- ブランチ = 枝  
**デフォルトブランチ**から  
枝を生やす
  - デフォルトブランチの名称は  
変更可能
- ※この資料では**main**を使用します

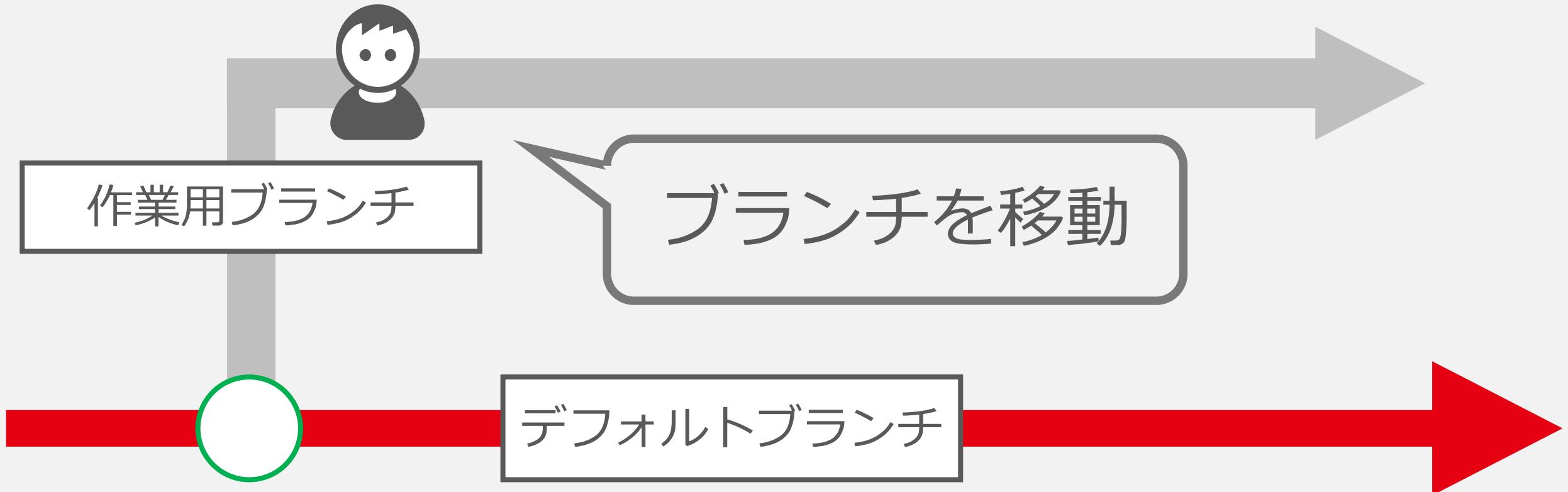
# ブランチ

作業用ブランチを作成する



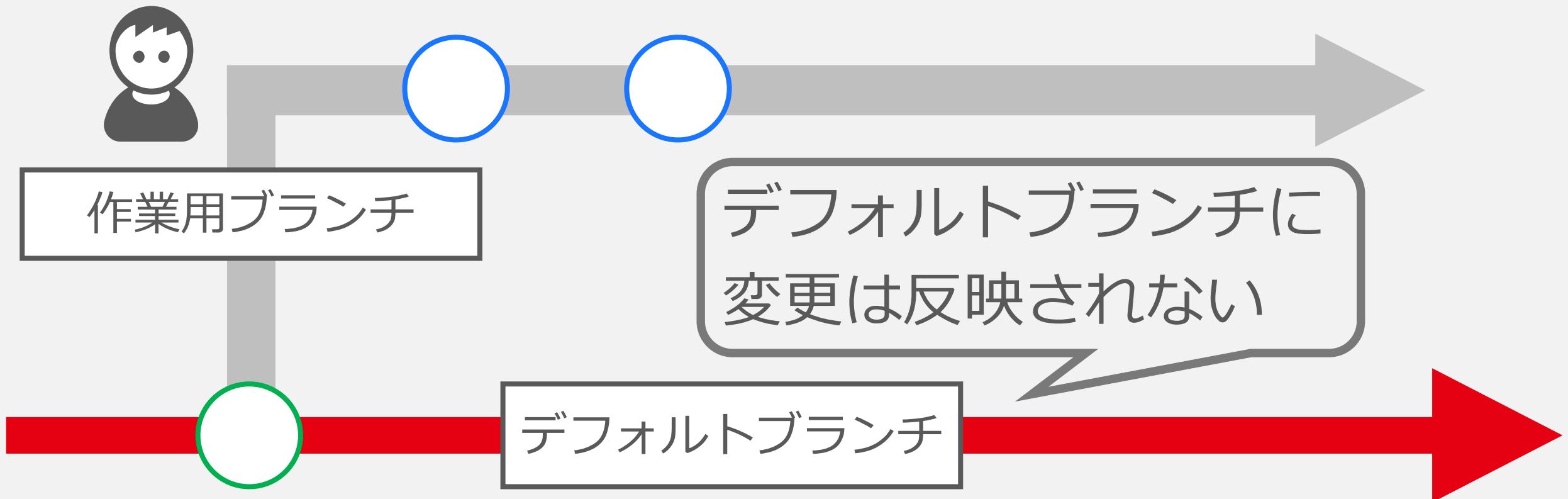
# ブランチ

作業用ブランチにチェックアウトをする



# ブランチ

作業ブランチで同じようにcommit/pushする



# VSCodeで ブランチを作成する



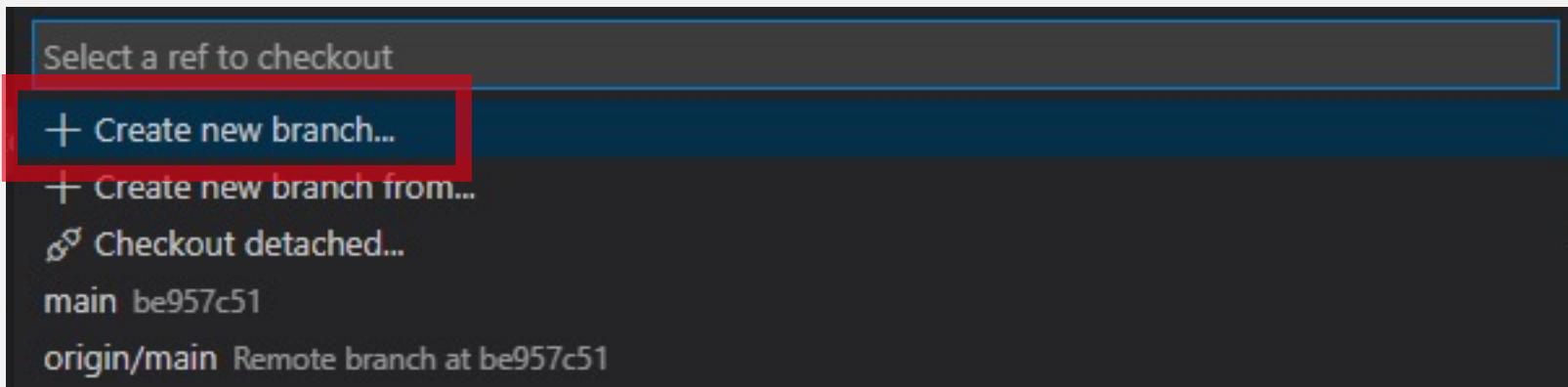
# ブランチを作成する

- GitHubの「ユーザーネーム」でブランチを作成します

- 画面下のブランチ名をクリック



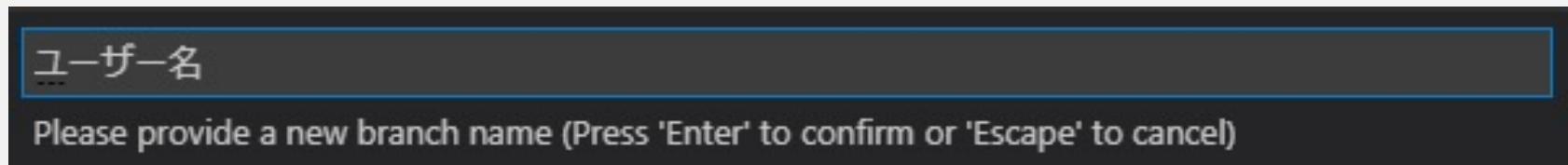
- 「Create new branch...」を選択



# ブランチを作成する

- GitHubの「ユーザー名」でブランチを作成します

- 「Branch name」の入力欄にユーザー名を入力

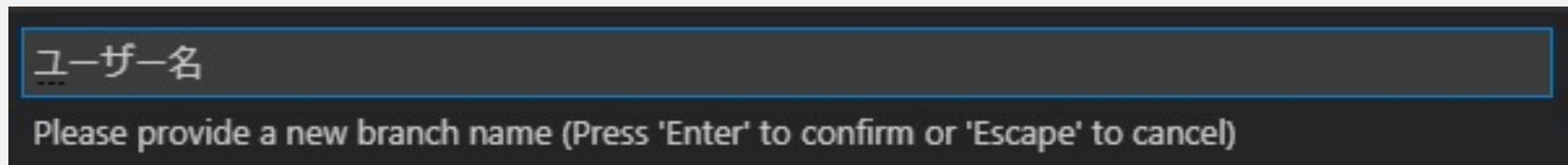


- 画面下のブランチ名が変更されたのを確認



# ブランチを作成する

- GitHubの「ユーザー名」でブランチを作成します
- 「Branch name」の入力欄にユーザー名を入力



- 画面下のキー、矢印キーもしくはEnterキーを確認

ブランチ名は自分で好きに決められますが、  
今回は新規ブランチ名をGitHubアカウント  
と同じにしましょう。

ブランチを指定して  
ファイル変更



# ブランチを作成してファイルをPush

- ファイルを変更してadd → commit
  - addする前にブランチがmainでないことを必ず確認しましょう
- commitしたら作成したブランチをリモートにpushします
  - mainブランチでpushしないように注意

# GitHub でブランチを確認

The screenshot shows a GitHub repository interface. At the top, there is a navigation bar with tabs: Code (highlighted with a blue border), Issues, Pull requests, Actions, Projects, Wiki, and Security. Below the navigation bar, there is a yellow banner with a user icon and the text "ユーザー名 had recent pushes less than a minute ago". To the right of the banner is a green button labeled "Compare & pull request". In the center, there is a dropdown menu showing "main" (highlighted with a blue border), "2 branches", and "0 tags". To the right of the dropdown are three buttons: "Go to file", "Add file", and "Code" (highlighted with a green border). Below the center, there is a modal window titled "Switch branches/tags" with a search bar containing "Find or create a branch...". Under the search bar, there are two tabs: "Branches" (highlighted with a blue border) and "Tags". The "Branches" tab shows a list with "main" (marked with a checkmark) and "ユーザー名". A small "default" label is next to "main". At the bottom of the modal, there is a blue button labeled "View all branches". At the very bottom of the page, there is a small "README.md" link.

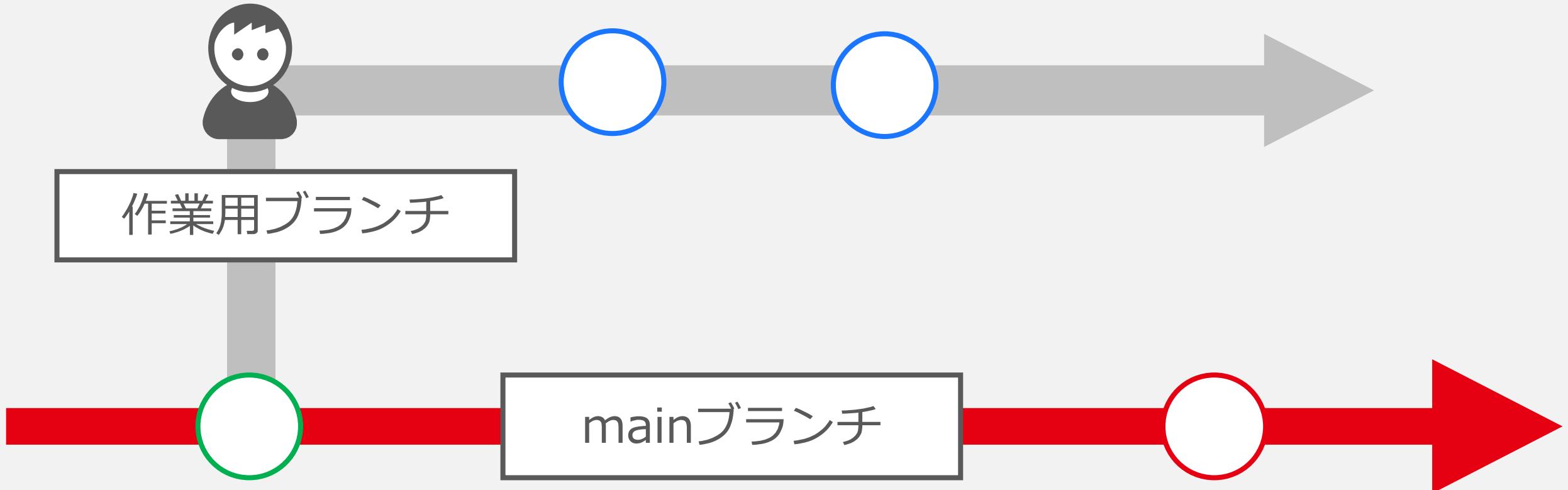
Code → main をクリック  
Branches にユーザー名があればOK  
クリックするとそのブランチの内容を確認できます

# Step.2-2

## mainに変更を反映する

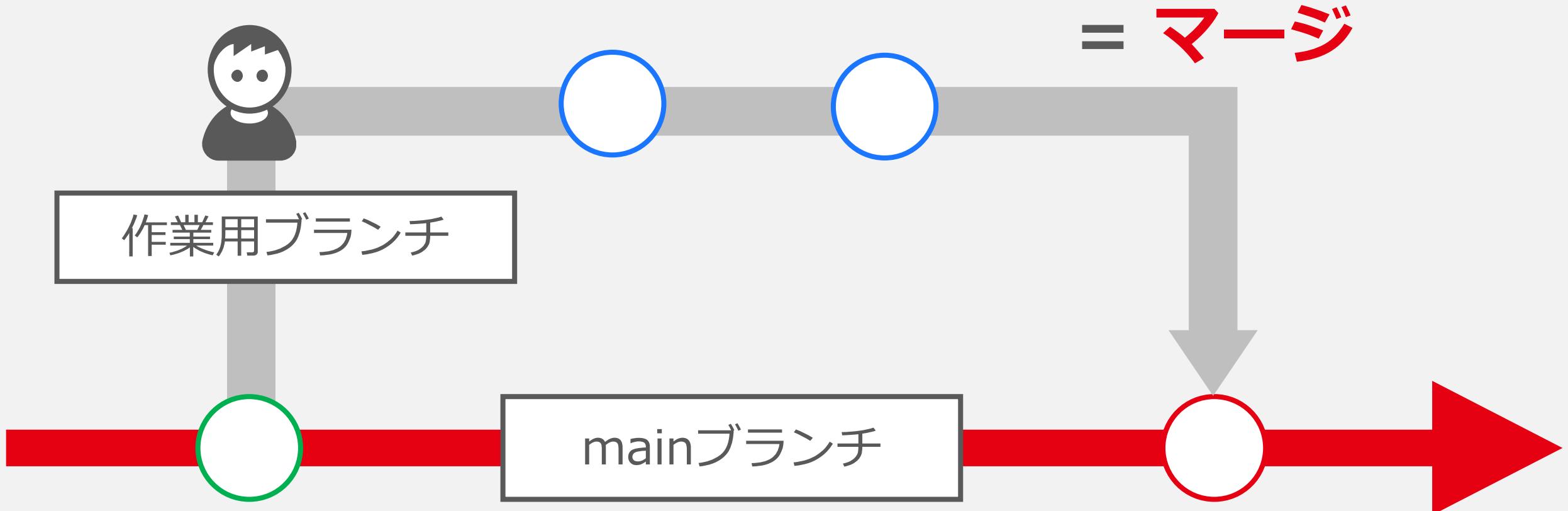
# マージ

作業用ブランチの変更をmainに反映する



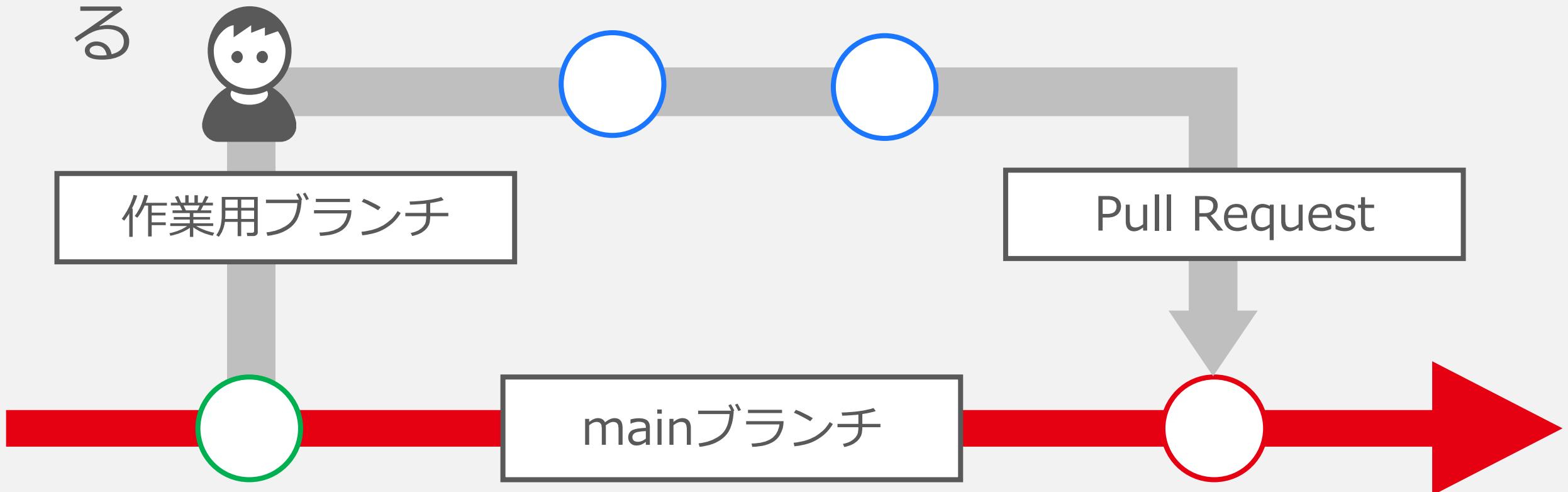
# マージ

ブランチの変更を別のブランチに反映する



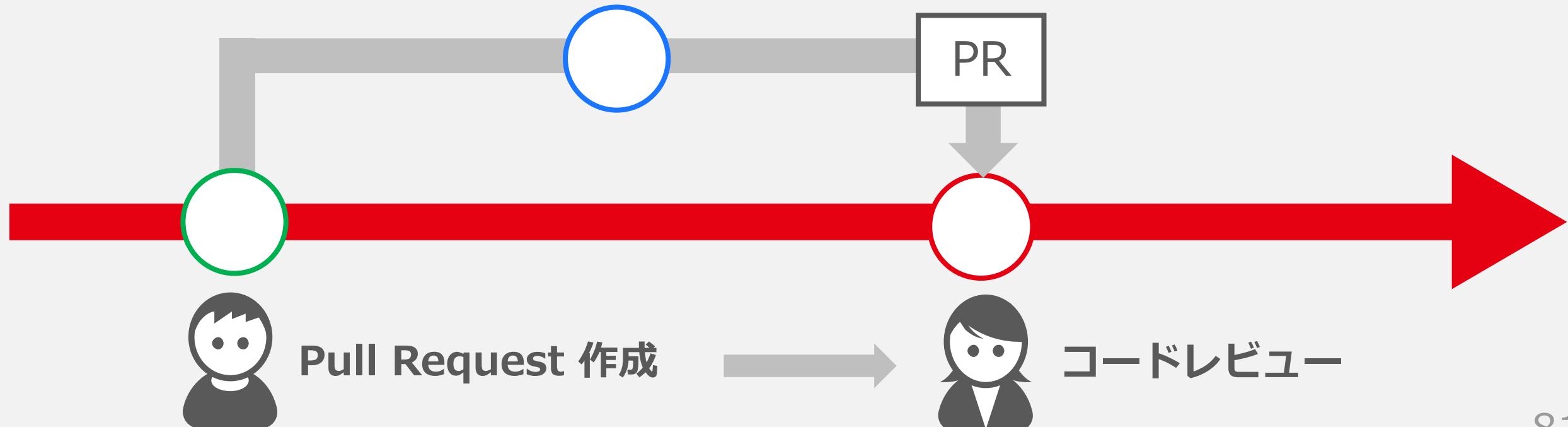
# マージ

マージするために Pull Request (PR) を作成する



# マージ

- **Pull Request** を作成する
  - GitHub の機能
  - マージしてOKかコードレビューをする



# Pull Requestの作成



# Pull Request作成してマージする

- GitHubでPull Requestを作成
  - 追加/変更した内容を記載
- Pull Requestをマージ
  - 変更内容を確認して問題なければ**Confirm merge**
  - リモートリポジトリの main が更新されているのを確認

# Pull Requestを作成方法

The screenshot shows a GitHub repository page. At the top, there is a navigation bar with links: Code (highlighted), Issues, Pull requests, Actions, Projects, Wiki, and Security. Below the navigation bar, a yellow banner displays a message: "ユーザーネーム had recent pushes less than a minute ago" and a green button labeled "Compare & pull request". Underneath the banner, there are buttons for "main" (with a dropdown arrow), "2 branches", "0 tags", "Go to file", "Add file", and "Code". On the left side, there is a sidebar with icons for a user profile, "Add R", "src", "README.md", and "index.html". A large blue callout box contains Japanese text: "GitHubにいくとこのような画面が出てきています (青枠注文)" and "※もし出でない場合はPull requestsのページから作成できます".

ユーザーネーム had recent pushes less than a minute ago

Compare & pull request

main

2 branches

0 tags

Go to file

Add file

Code

ユーザーネーム Add R

src

README.md

index.html

README.md

GitHubにいくとこのような画面が  
出てきています (青枠注文)

※もし出でない場合はPull requestsの  
ページから作成できます

# Pull Requestを作成方法

Code Issues Pull requests Actions Projects Wiki Security

ユーザーネーム had recent pushes less than a minute ago Compare & pull request

main 2 branches 0 tags Go to file Add file Code

ユーザーネーム Add REA/ Compare & pull requestを押す

src	Add index.html & src	1 hour ago
README.md	Add README.md	39 minutes ago
index.html	Add README.md	39 minutes ago

README.md

# Pull Requestの作成方法

# Pull Requestの作成方法

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

The screenshot shows the GitHub interface for creating a pull request. At the top, there are dropdown menus for 'base' set to 'main' and 'compare' set to 'ブランチ名'. A green success message indicates 'Able to merge. These branches can be automatically merged.' Below this, there's a large blue callout box containing Japanese translations for the fields:

- main ← 作成したブランチ名**
- base: 変更を反映するブランチ**
- compare: 変更作業を行ったブランチ**

The interface also includes fields for 'PR タイトル' (Title), 'Write' (content area), 'Reviewers', 'Projects' (None yet), 'Milestone' (No milestone), and 'Linked issues'.

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

# Pull Requestの作成方法

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

The screenshot shows the GitHub interface for creating a pull request. At the top, it says "base: main" and "compare: ブランチ名". A green checkmark indicates "Able to merge. These branches can be automatically merged." Below this, there's a text input field labeled "PR タイトル" (PR Title) which is highlighted with a blue border. A large blue callout box contains the Japanese text "Pull Requestのタイトル" and "→ 概要がひと目でわかるように". To the right of the title input, there are sections for "Reviewers" (No reviews), "Assignee" (Assign yourself), "Projects" (None yet), "Milestone" (No milestone), and "Linked issues". At the bottom, there's a note about following GitHub Community Guidelines and a "Create pull request" button.

PR タイトル

PR のタイトル  
→ 概要がひと目でわかるように

base: main compare: ブランチ名 ✓ Able to merge. These branches can be automatically merged.

Reviewers

No reviews

Assignee

Assign yourself

Projects

None yet

Milestone

No milestone

Linked issues

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Create pull request

① Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).



# Pull Requestの作成方法

Open a pull request

Create a new pull request

Pull Requestの概要  
→ どのような変更をしたか説明を記載

Write

Preview



変更内容を記載

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

No reviews

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked issues

① Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).



# Pull Requestの作成方法

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

The screenshot shows the GitHub interface for creating a pull request. At the top, it says "base: main" and "compare: ブランチ名". A green checkmark indicates "Able to merge. These branches can be automatically merged." Below this, there's a title field labeled "PR タイトル" and a rich text editor toolbar with "Write" and "Preview" buttons. The main body area has a placeholder "変更内容を記載". On the right side, there are sections for "Reviewers" (No reviews), "Assignees" (No one—assign yourself), "Labels" (None yet), "Milestone" (No milestone), and "Linked issues". A large blue button at the bottom with the text "タイトルと概要入力したらクリック" (Click after entering title and summary) covers the "Create pull request" button. The "Create pull request" button itself is highlighted with a red box.

タイトルと概要入力したらクリック

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

① Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Copyright (C) 2020 Yahoo Japan Corporation. All Rights Reserved.

# Pull Requestの作成方法

The screenshot shows a GitHub pull request page with the following details:

- Title:** PR タイトル #1
- Status:** Open
- Merge Status:** ユーザー名 wants to merge 1 commit into main from ユーザー名
- Activity:** Conversation 0, Commits 1, Checks 0, Files changed 2.
- Comment:** ユーザー名 commented now: 変更内容を記載
- Commit:** Update index.html by ユーザー名 at 78149a0
- Push Instructions:** Add more commits by pushing to the ユーザー名 branch on ユーザー名 / github\_ws.
- CI Status:** Continuous integration has not been set up. GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.
- Branch Conflict:** This branch has no conflicts with the base branch. Merging can be performed automatically.
- Buttons:** Merge pull request, Open this in GitHub Desktop, View command line instructions.

# Pull Requestの作成方法

PR タイトル #1

Open ykato524 wants to merge 1 commit into main from ykato524

Conversation 0 Commits 1 Checks 0 Files changed 2 +4 -1 Review changes

Changes from all commits ▾ File filter... ▾ Jump to... ▾ 1 / 2 files viewed Review changes

「Files changed」に変更差分が掲載される

...	...	...
11	11	<!-- 変更箇所に困ったら下記のコメントアウトを外してください -->
12	12	<div id="target">ここをクリックするとアラート発生</div>
14	-	<!--<div id="ok">GitHub完全に理解した</div>-->
	+	<div id="ok">GitHub完全に理解した</div>
15	15	
16	16	<script src=".src/index.js"></script>
17	17	</body>

# Pull Requestの確認

The screenshot shows a GitHub Pull Request interface. At the top, there's a commit history with one entry: "Update index.html" by user "78149a0". Below the commit, a message says "Add more commits by pushing to the ユーザー名 branch on ユーザー名 / github\_ws." A green icon with a gear and a plus sign indicates CI status.

**Continuous integration has not been set up**  
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

**This branch has no conflicts with the base branch**  
Merging can be performed automatically.

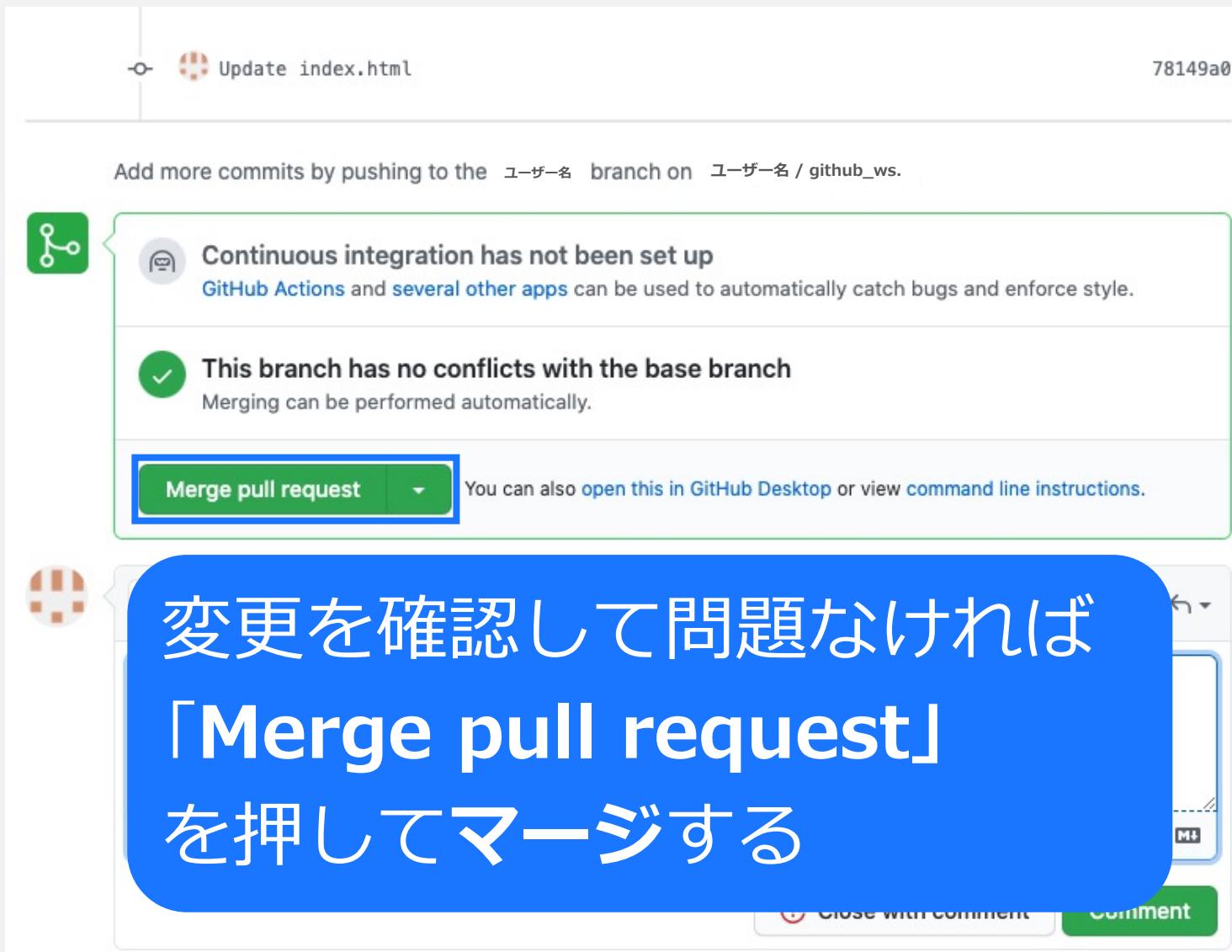
**Merge pull request** You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

A comment box is open, showing the text "レビューや気になった内容はコメントできる". The comment area includes a rich text editor toolbar and a file attachment instruction. At the bottom are "Close with comment" and "Comment" buttons.

# Pull Requestの確認



# Pull Requestをマージ



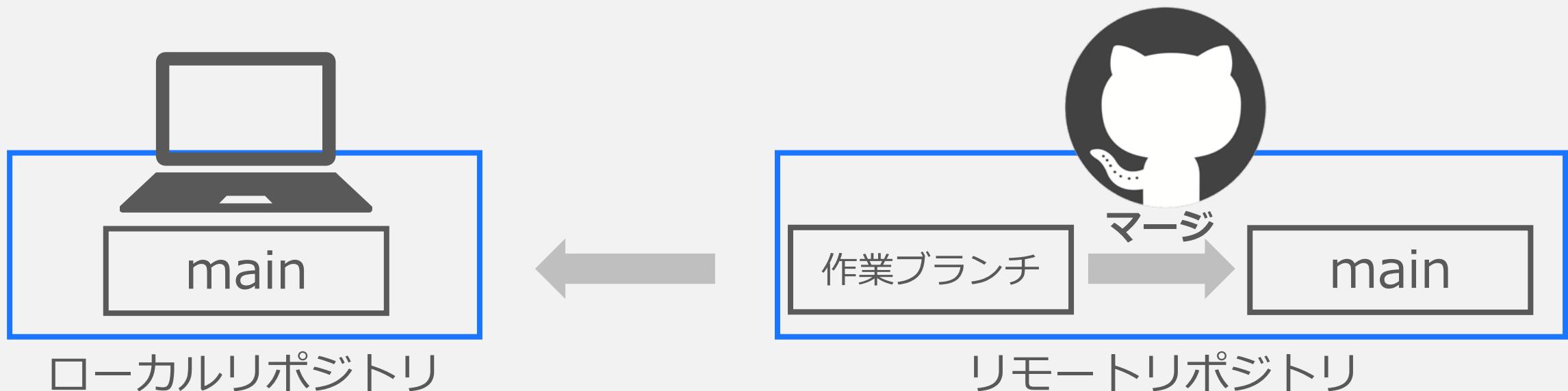
ローカルリポジトリに  
pullする



# マージされたものをローカルに反映

Pull Requestがマージされると

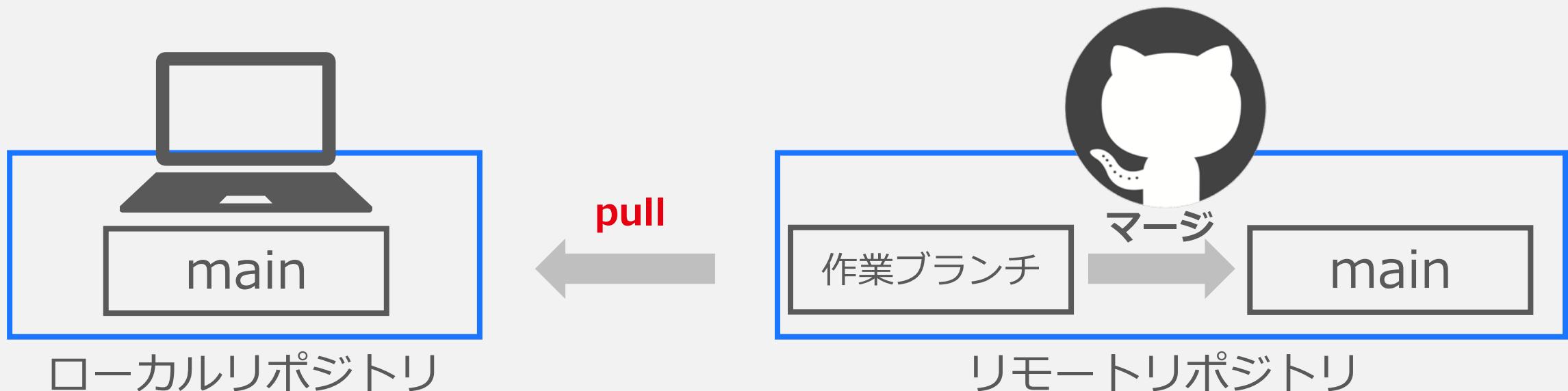
- ・ リモートのmainには反映される
- ・ ローカルのmainには反映されていない



# マージされたものをローカルに反映

リモートのmainの内容をローカルのmainに反映

→ ローカルに**pull**する

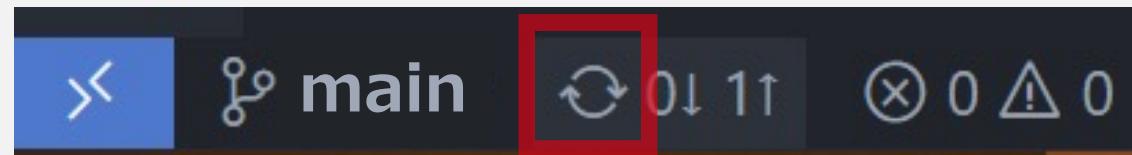


VSCodeで  
pullする



# ローカルにpullする

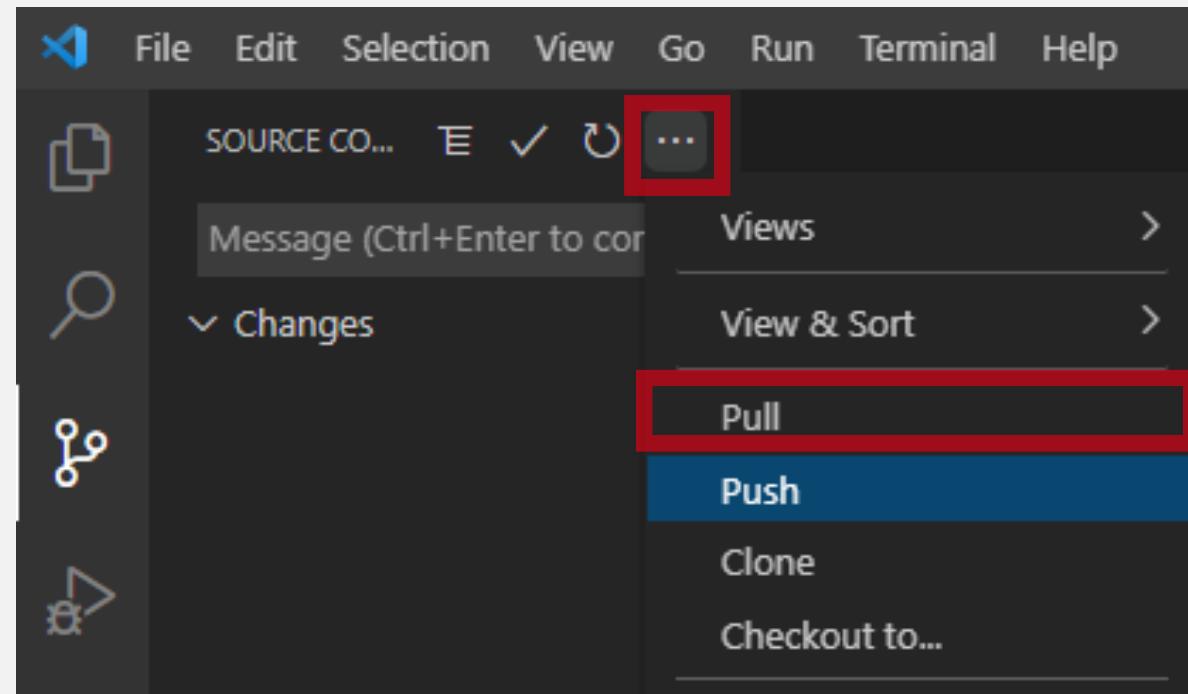
1. 画面下のブランチ名をクリックして、mainをクリックしてブランチを移動
2. 「1↓0↑」のように表示されているとき更新ボタンを押すとpullされます  
(状況により表示されない場合もあります)



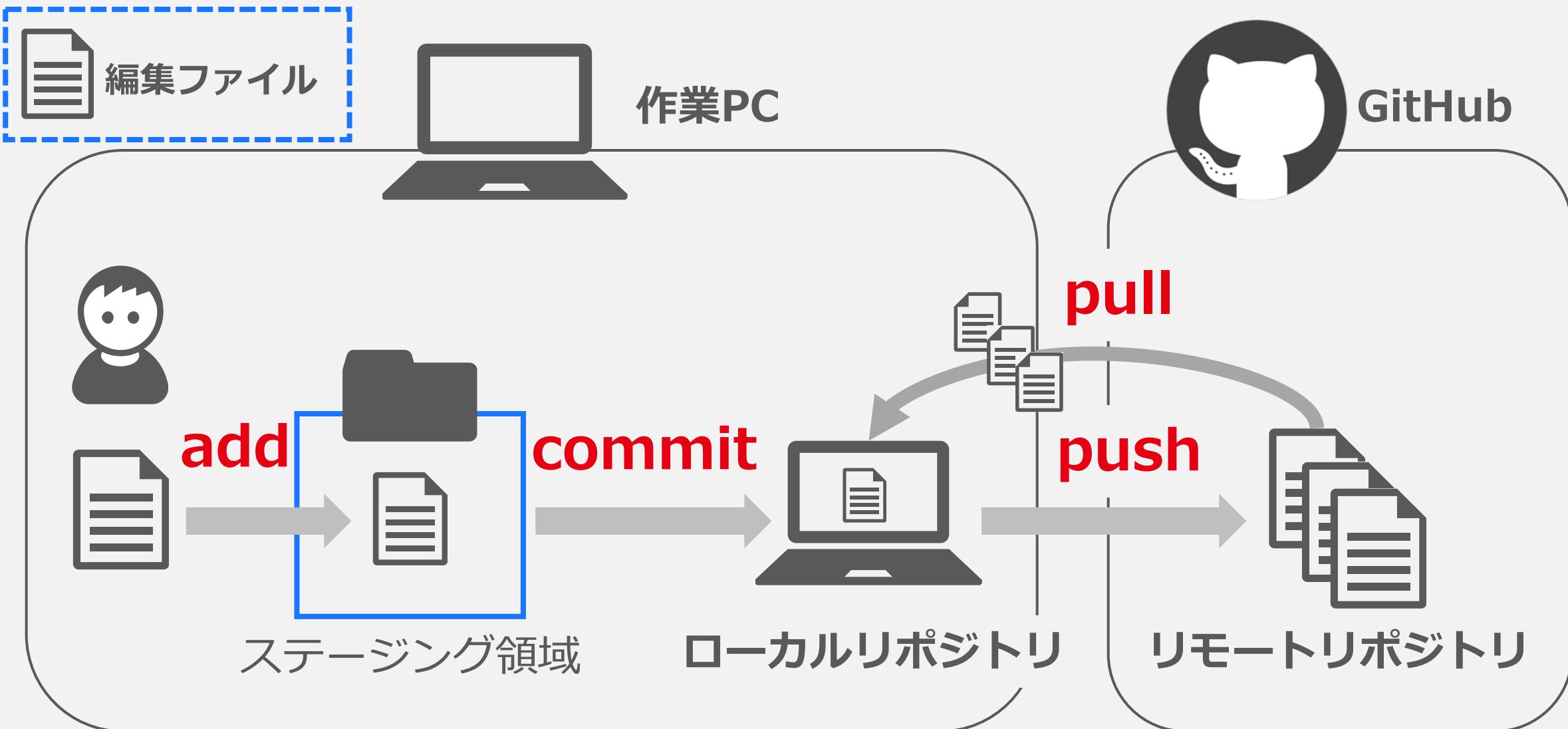
3. ローカルファイルが更新されていくことを確認

# ローカルにpullする

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

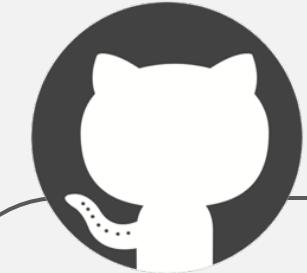


# ローカルリポジトリの更新



# ローカルリポジトリの更新

**clone** : リモートをローカルにコピー  
**pull** : リモートの変更をローカルに反映



GitHub



# Step.2

## ハンズオン

# ハンズオンの作業内容 その1

- 新たにブランチを作成し、ファイルをpush
  1. 別のブランチを作成
  2. ファイルを編集して作成したブランチにpushしてみよう
  3. GitHubで作成したブランチとpushしたファイルを確認

# ハンズオンの作業内容 その2

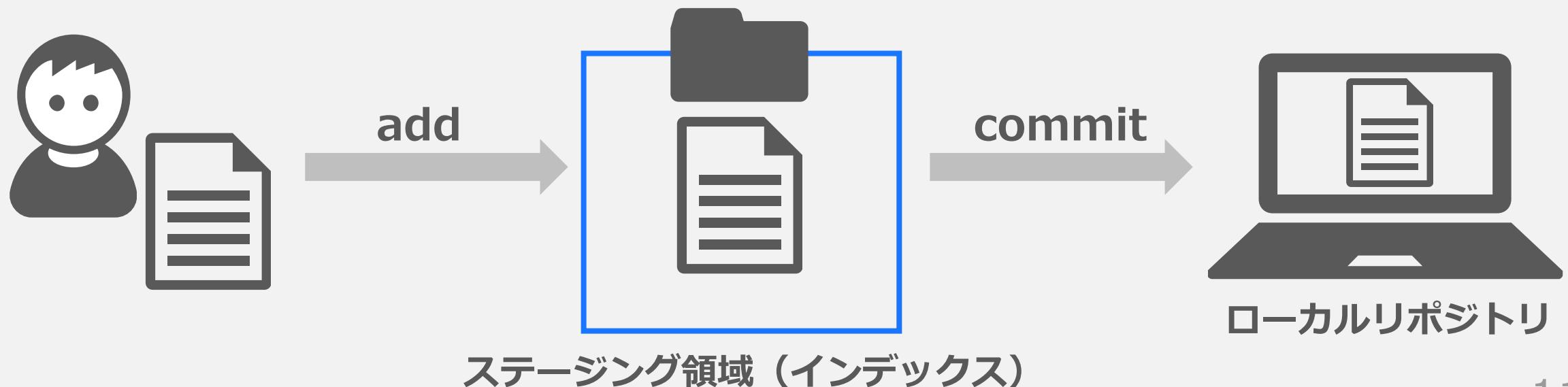
- Pull Requestを作成してmainに反映
  - 1. Pull Requestを作成
    - 変更点をコメントしよう
    - 他人のPRにコメントしよう
  - 2. Pull Requestをマージ
  - 3. ローカルのmainにpull
    - 変更が反映されてるかファイル内容を確認しよう

おさらい

YAHOO!  
JAPAN

# 【おさらい】ローカルリポジトリの作業

- ・ ファイルの変更は**add**して**commit**
- ・ addするファイルの内容に注意
- ・ commitしたあとは**必ずコメントを残す**

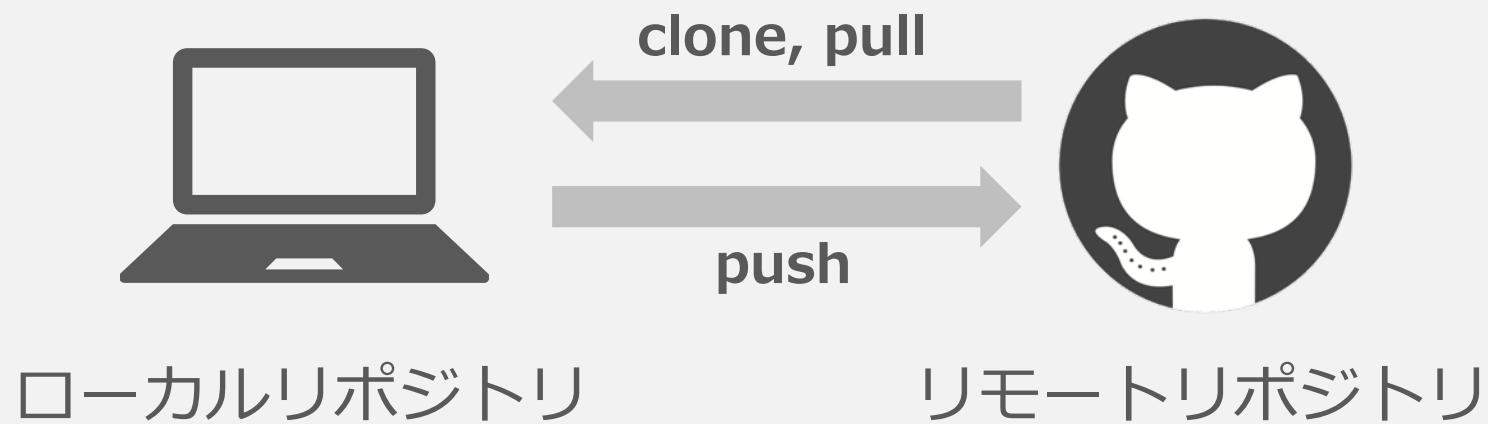


# 【おさらい】ローカルとリモートの関係

**clone** : リモートをローカルにコピー

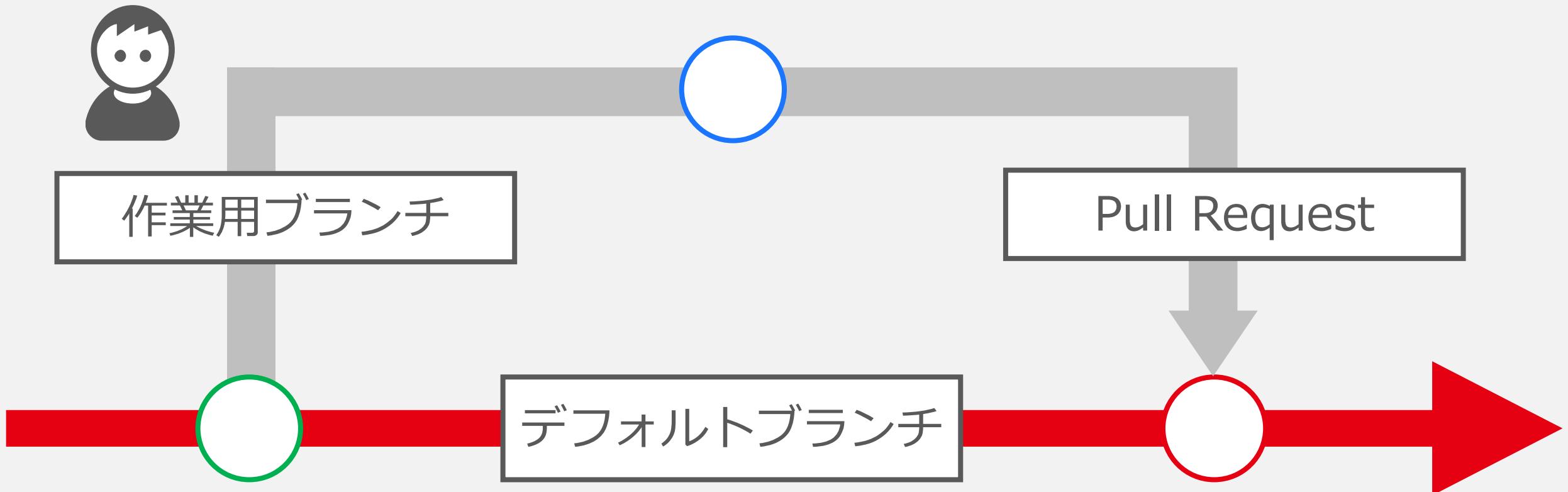
**pull** : リモートの変更をローカルに反映

**push** : ローカルの変更をリモートに反映



# 【おさらい】 ブランチ

作業ブランチを作つてPull Requestをする



# まとめ

- ・ この資料で扱った機能は基礎中の基礎
- ・ GitとGitHubにはこれ以外にも便利な機能があるので、必要な場合は検索してみましょう。

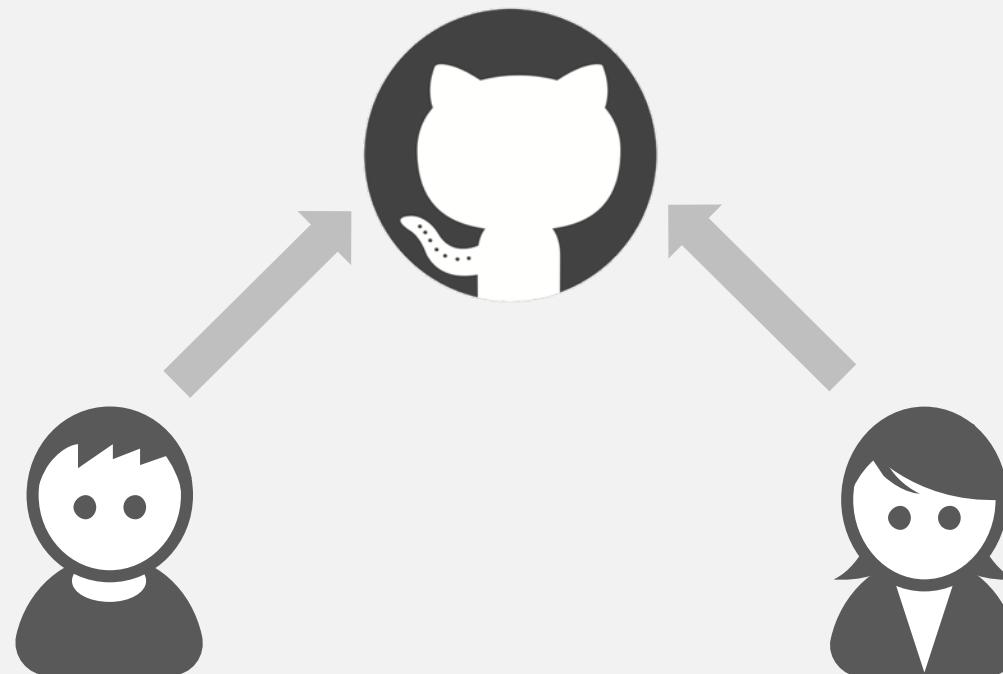
まずは基本的な使い方に慣れる。  
使っていく中で調べながら使う。

# GitHubを複数人で使う



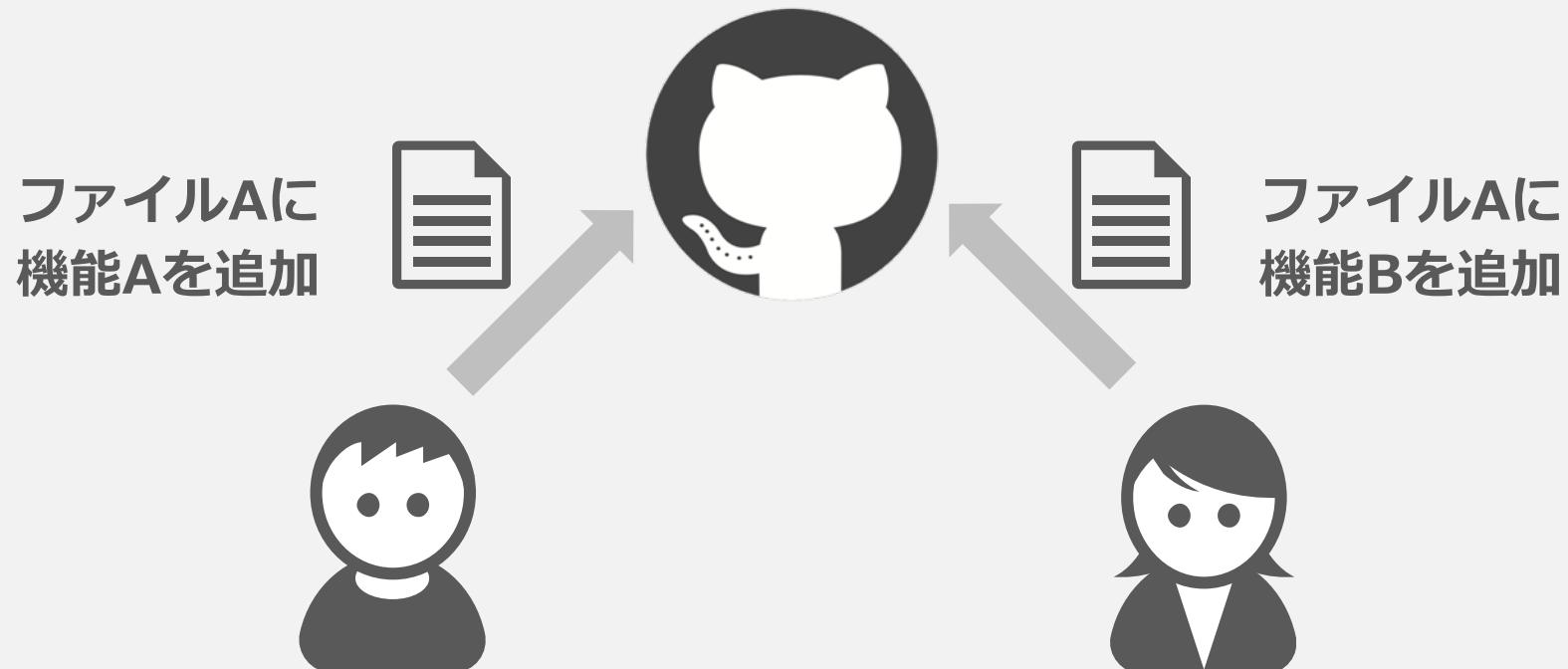
# チーム開発では

- 1つのリポジトリに対して複数人で作業する



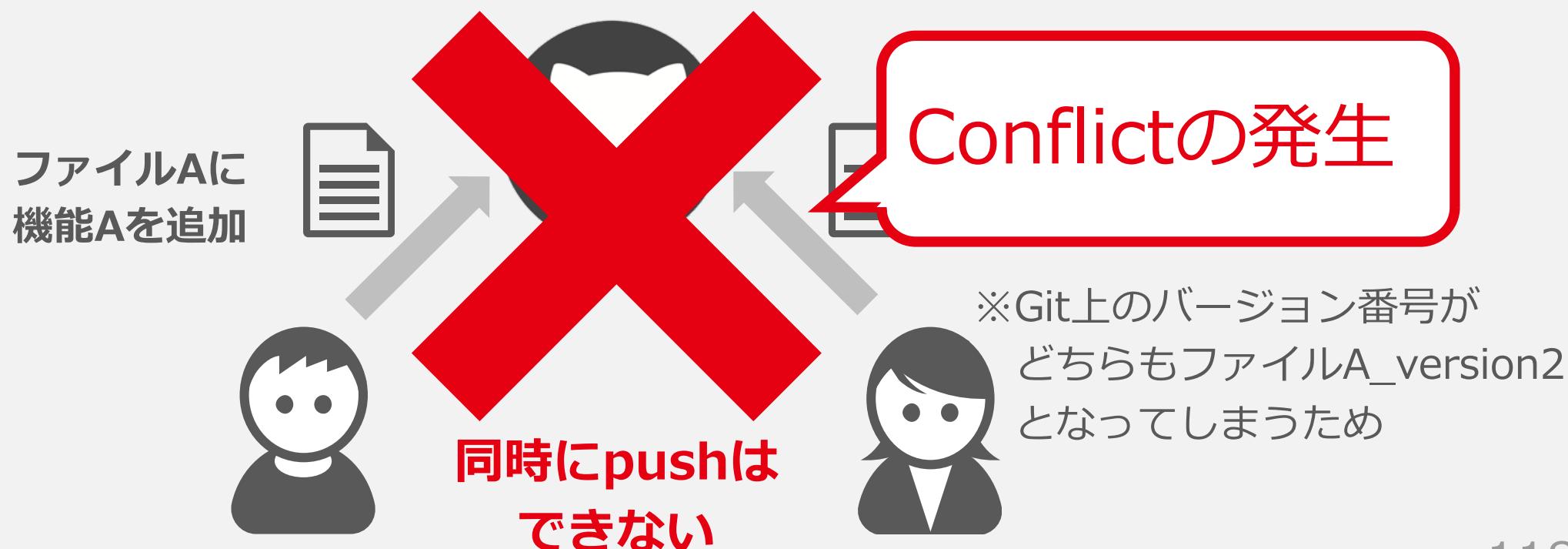
# チーム開発では

- ・ 1つのリポジトリに対して複数人で作業する
- ・ 同じファイルを同時に修正してしまう可能性も



# チーム開発では

- ・ 1つのリポジトリに対して複数人で作業する
- ・ 同じファイルを同時に修正してしまう可能性も



# チーム開発では

- ・ 1つのリポジトリに対して複数人で作業する
- ・ 同じコードを同時に修正してしまう可能性もある

Conflictを避けるため

作業者ごとに

ブランチを分けましょう



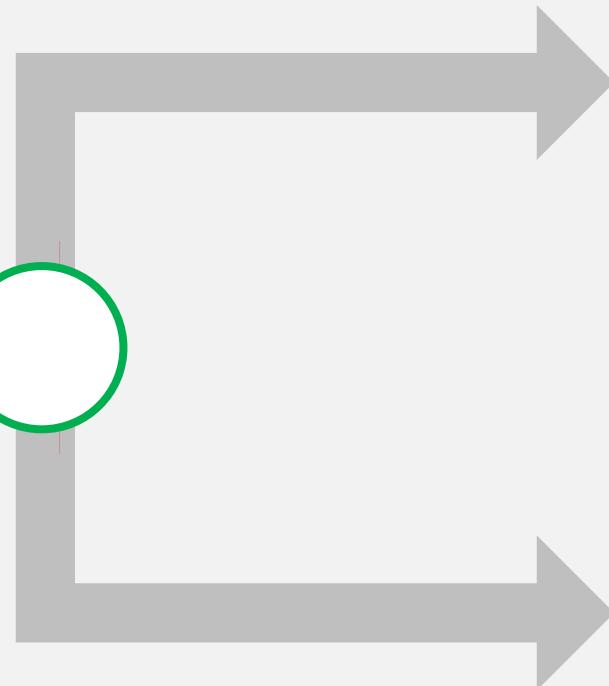
同時に push は  
できない



# チーム開発では



Aさんの作業ブランチ



Bさんの作業ブランチ

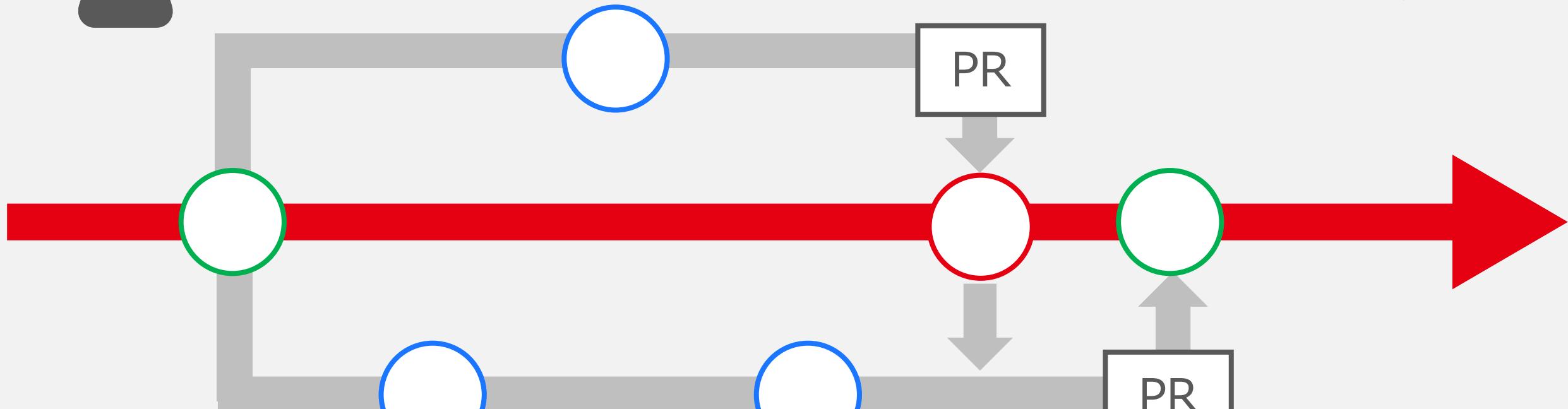
作業者がそれぞれブランチを作成

# チーム開発では



Aさんの作業ブランチ「A」

作業者がそれぞれブランチを作成  
→ PRでコードレビューをしてマージ



Bさんの作業ブランチ「B」

# チーム開発では



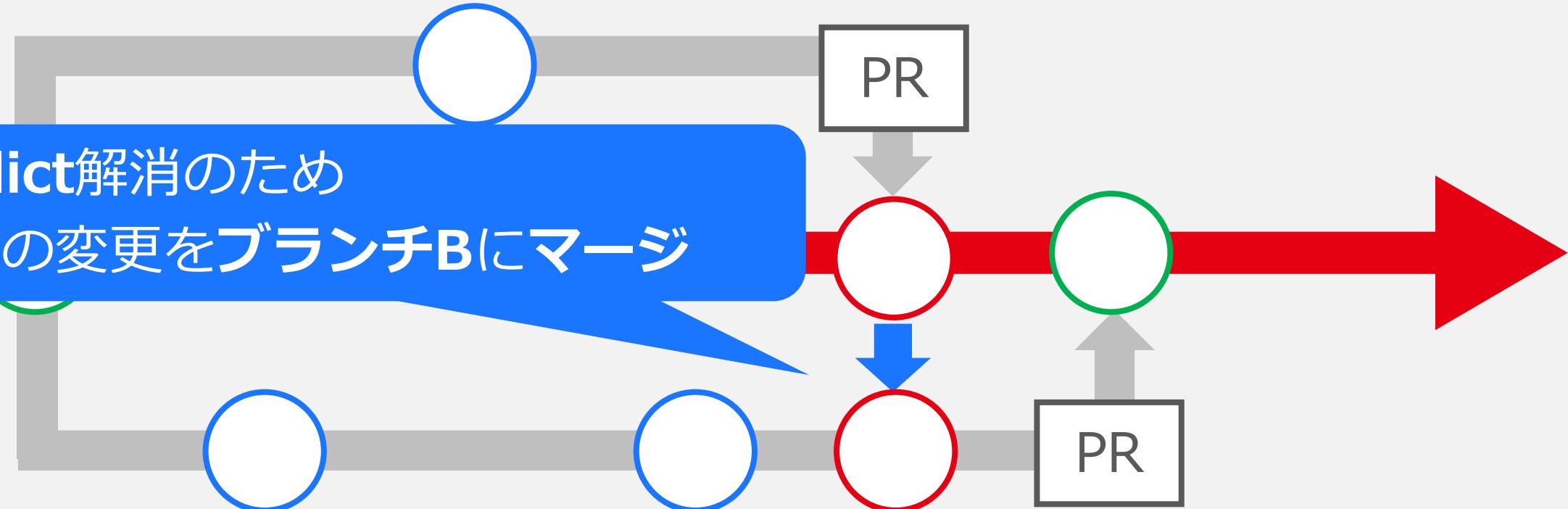
Aさんの作業ブランチ「A」

作業者がそれぞれブランチを作成  
→ PRでコードレビューをしてマージ

Conflict解消のため  
mainの変更をブランチBにマージ



Bさんの作業ブランチ「B」



# GitHub入門編

以上



# おまけ

- **Hack U をフォローしよう**

1. <https://github.com/hackujp> にアクセス
2. hackujp を Follow
3. hackujp/github\_tutorial を Watch & Star

※どちらも任意です

最新情報は **connpass/Twitter(@hackujp)** をフォロー

# Appendix



# Gitコマンドでの操作



# Gitコマンドでの操作

- add

```
$ git add ファイルパス
```

- commit

```
$ git commit -m "commitコメント"
```

# Gitコマンドでの操作

- ブランチ作成  
\$ git branch ブランチ名
- 今のブランチを確認  
\$ git branch
- ブランチの切り替え  
\$ git checkout ブランチ名
- ブランチの作成 + 切り替え  
\$ git checkout -b ブランチ名

# Gitコマンドでの操作

- clone

```
$ git clone リポジトリURL
```

- pull

```
$ git pull origin pullしたいブランチ名
```

- push

```
$ git push origin pushしたいブランチ名
```

# Gitコマンドでの操作

- “origin” とは？
  - リモートリポジトリのURLを指している
  - 毎回打つのが面倒なので“origin”という略称で代用できるように設定されている

例) これらは同じ内容を表す

```
$ git pull origin main
```

```
$ git pull git@github.com:ユーザ名/リポジトリ名.git main
```