

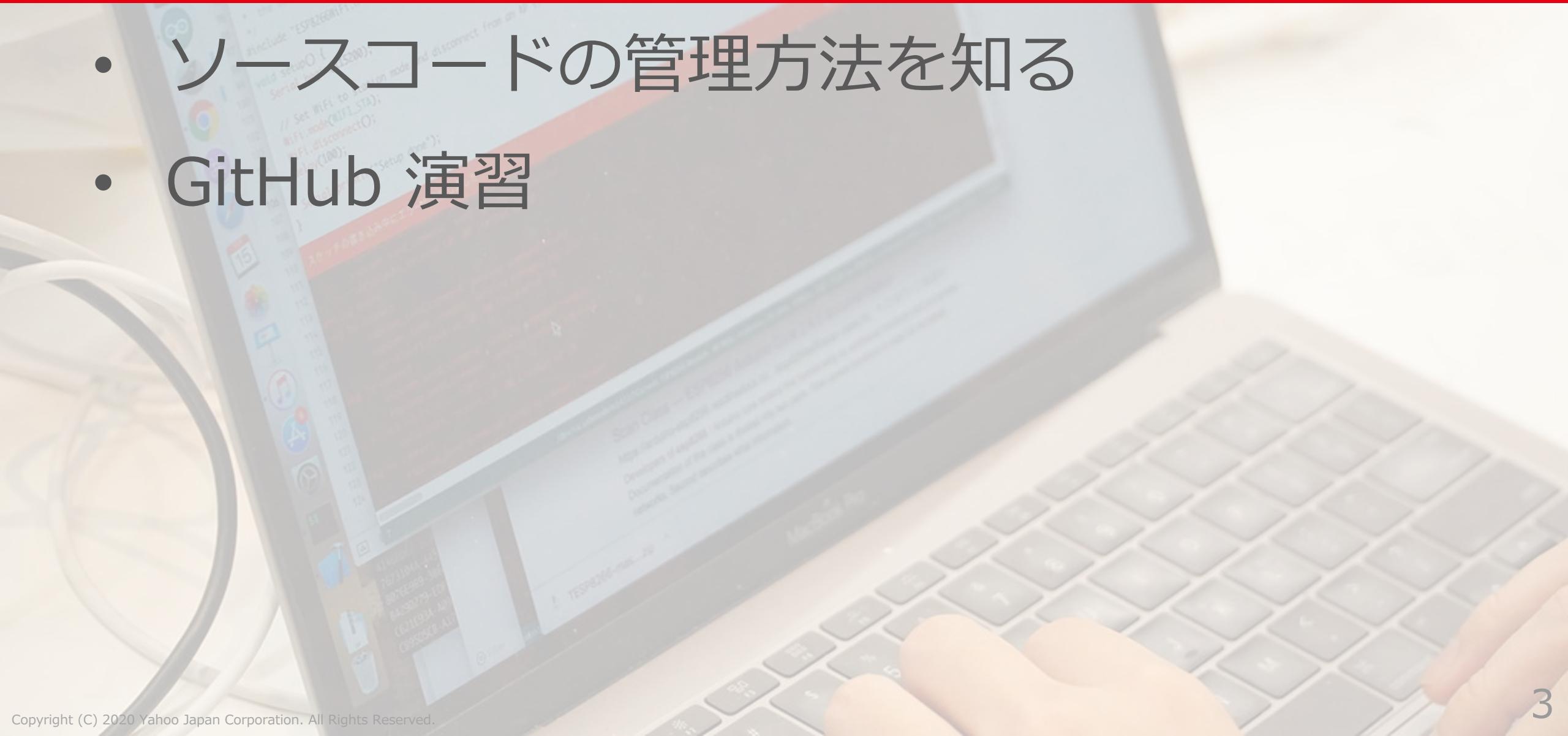
GitHub 入門編

GUI版



この資料で学ぶこと

- ・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

- ・ 各機能の解説
- ・ ツールの使い方
 - ・ 説明と一緒に進めてください
- ・ ハンズオン
- ・ まとめ
- ・ 懇親会

Git使ったことある人



YAHOO!
JAPAN

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

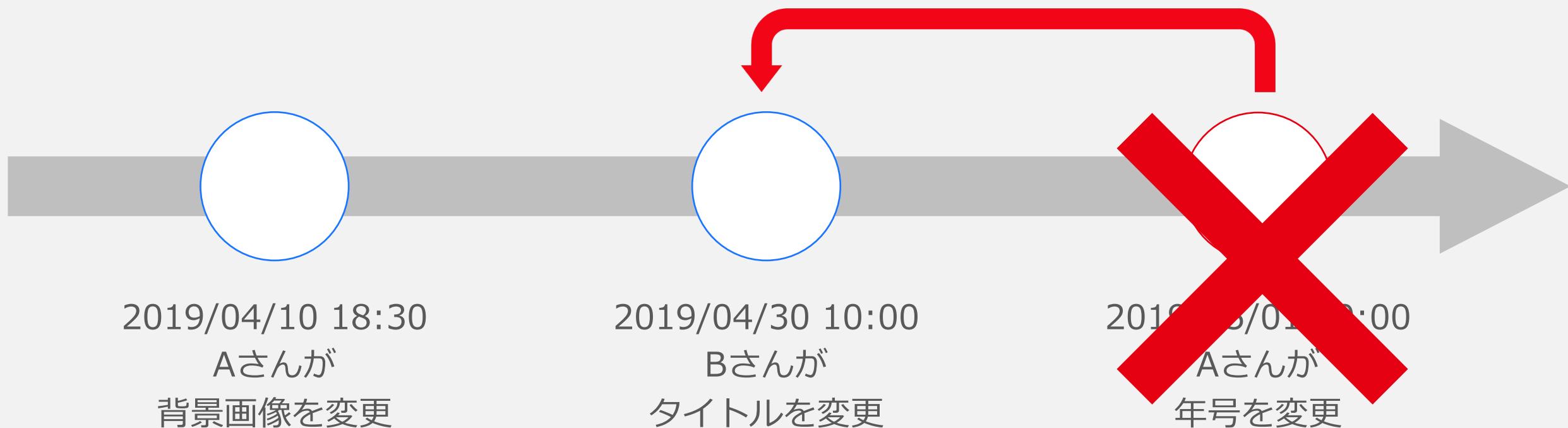
過去の状態との差分がわかる



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 :**

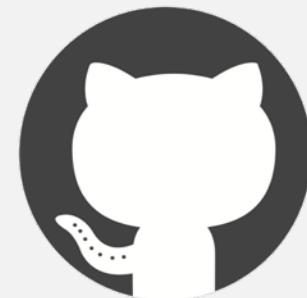
色々な人が共有できるようにしたリポジトリ
GitHub上にあるものはリモートリポジトリ

- **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 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

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

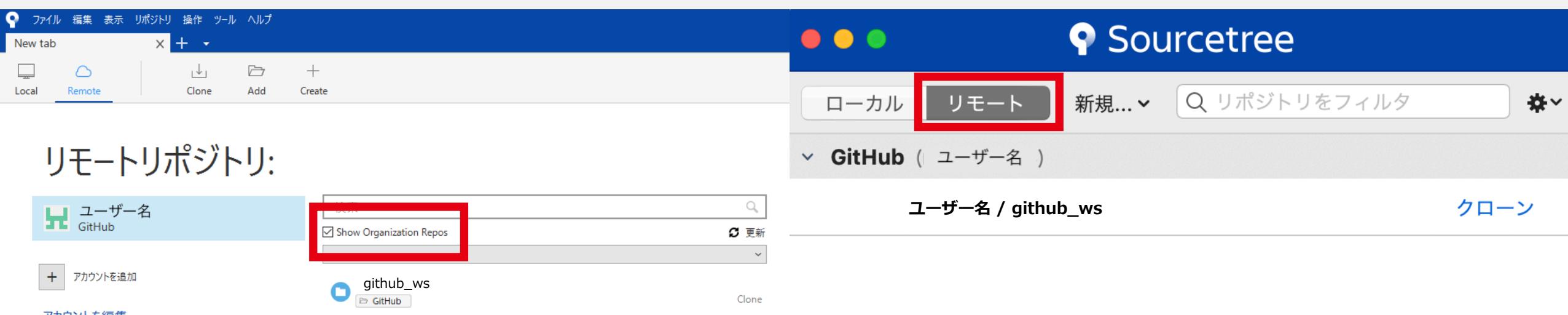
今回は Sourcetree という
GUIツールを使います

セットアップ
お済みですか？

YAHOO!
JAPAN

Sourcetree と GitHub を連携

setup.pdf を参考にセットアップしてください。
Sourcetree で「リモート」を選択して、
作ったリポジトリが表示されていればOKです。
(Windowsの人は Show Organization Repos のチェックを1回外すと出てきます)



ローカルリポジトリの準備

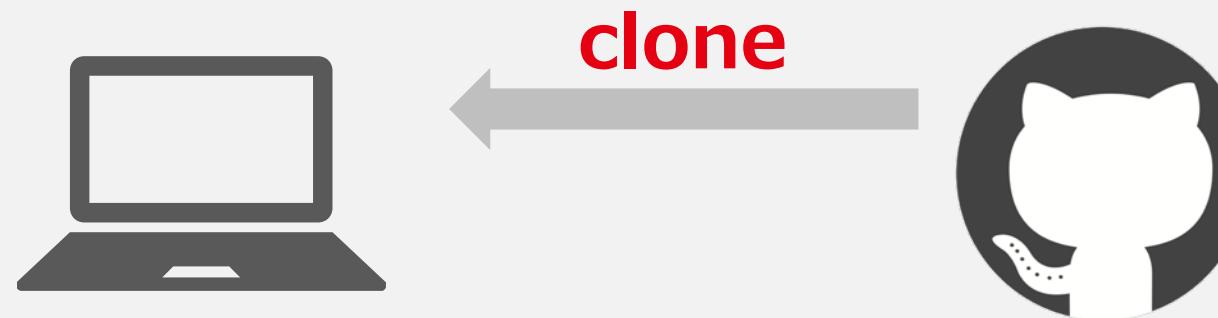
YAHOO!
JAPAN

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

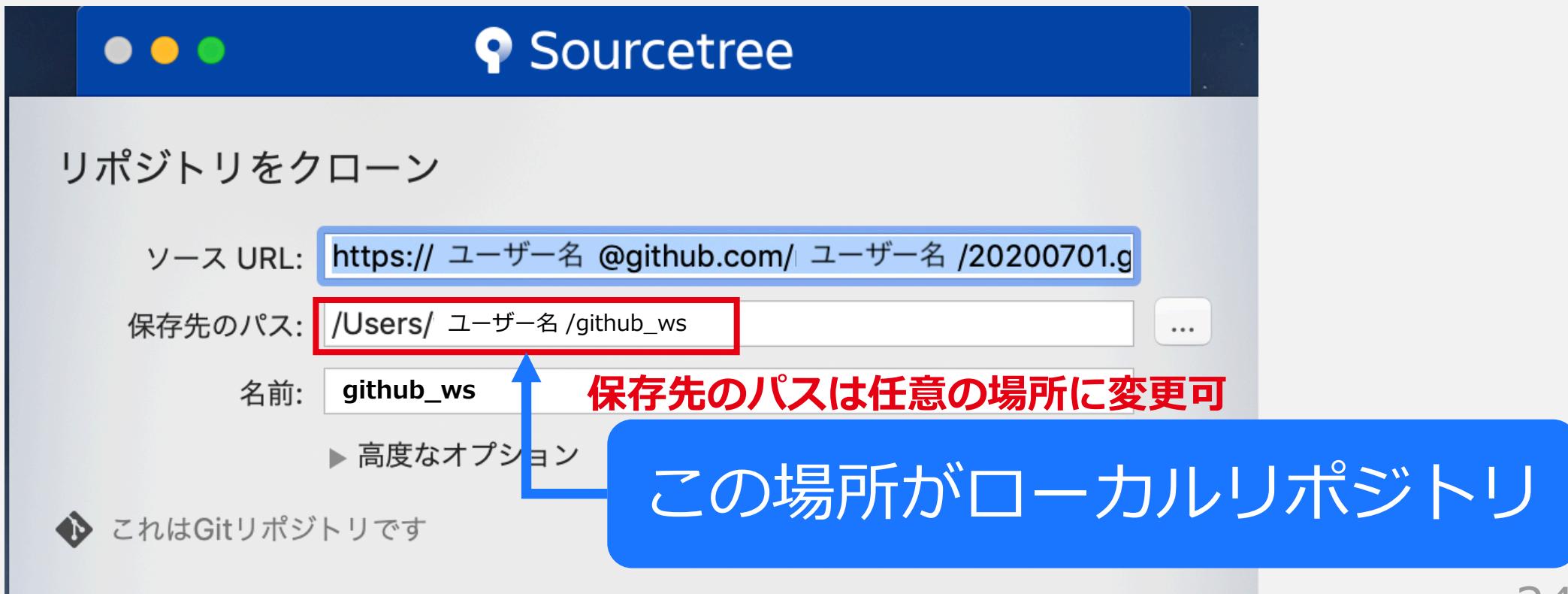
clone

- リモートに作成したリポジトリがあるので、その横にある「クローン」ボタンを押す

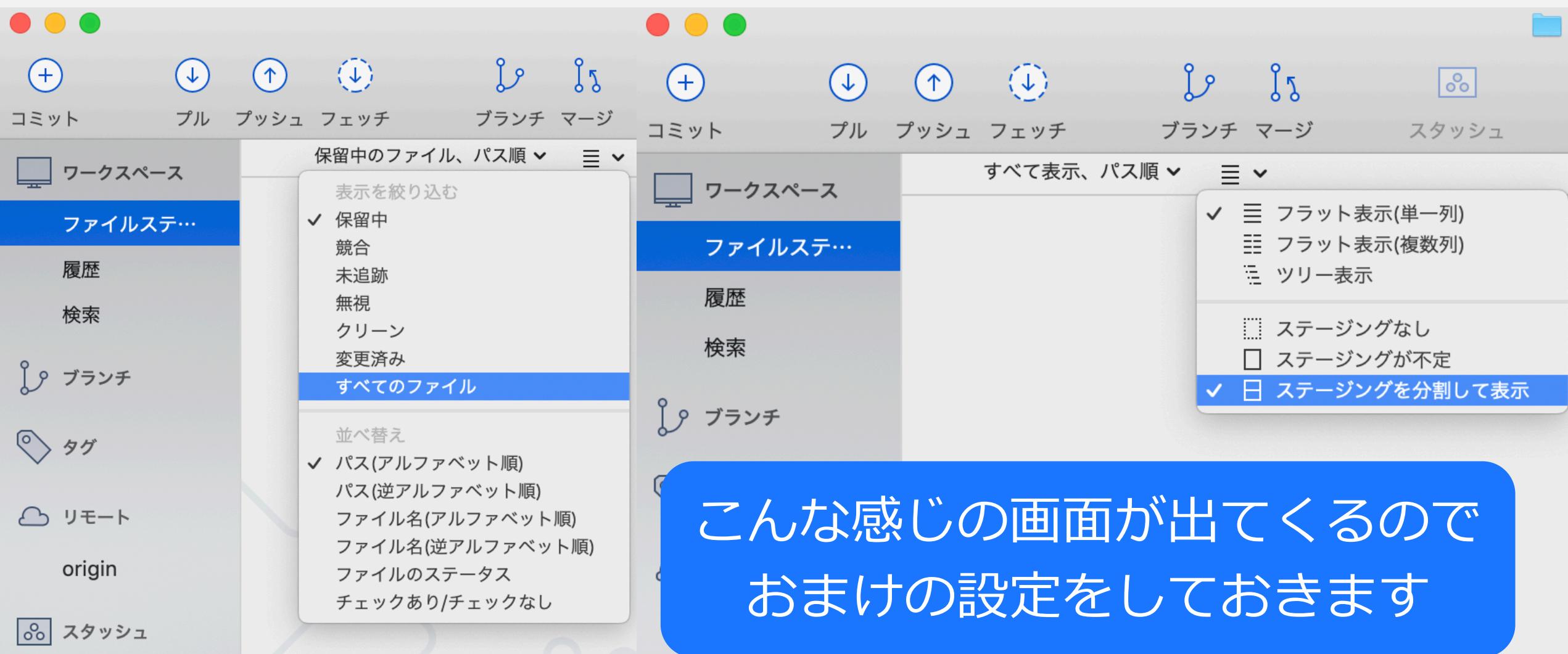


clone

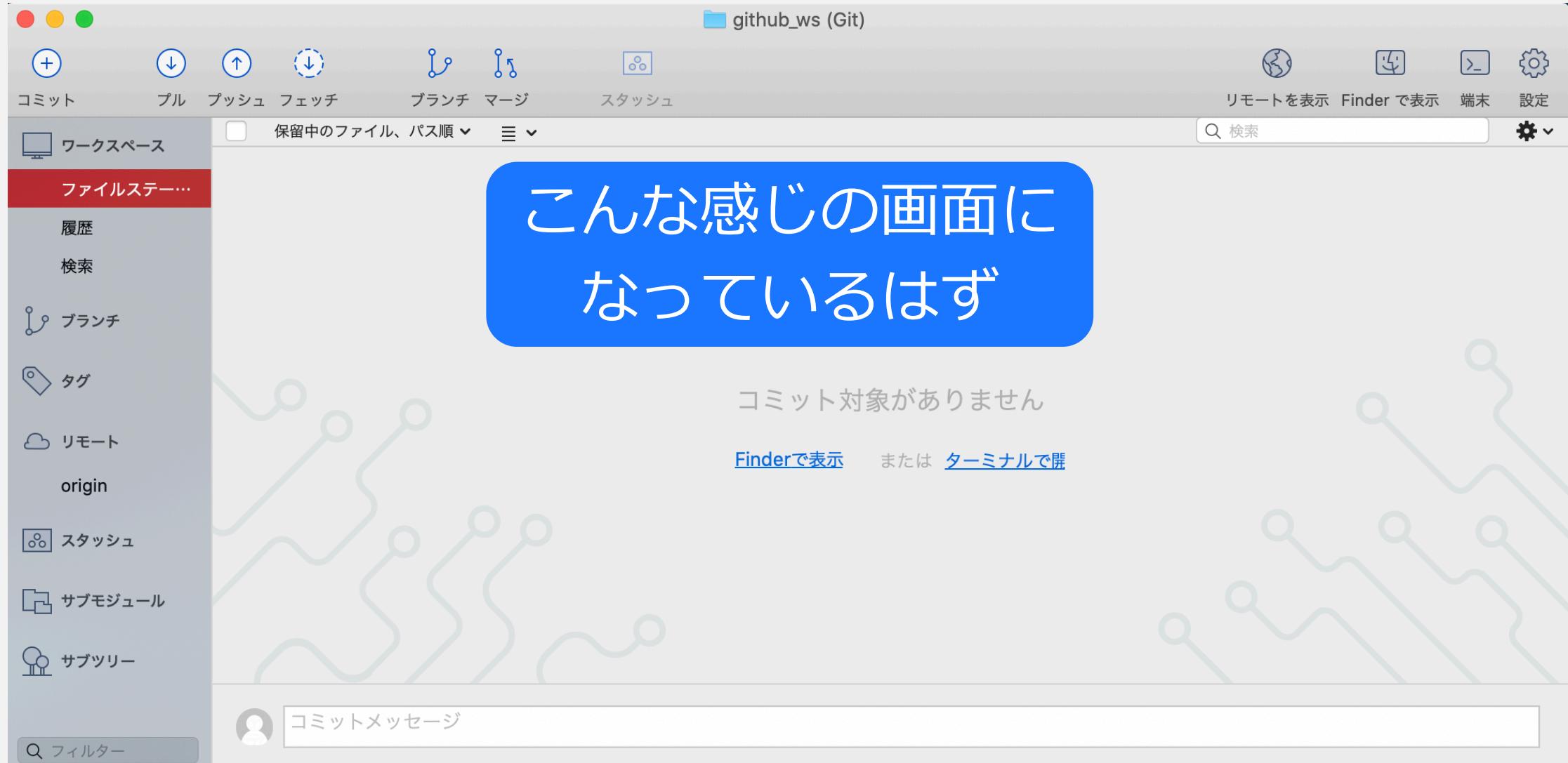
- リモートに作成したリポジトリがあるので、その横にある「クローン」ボタンを押す



clone

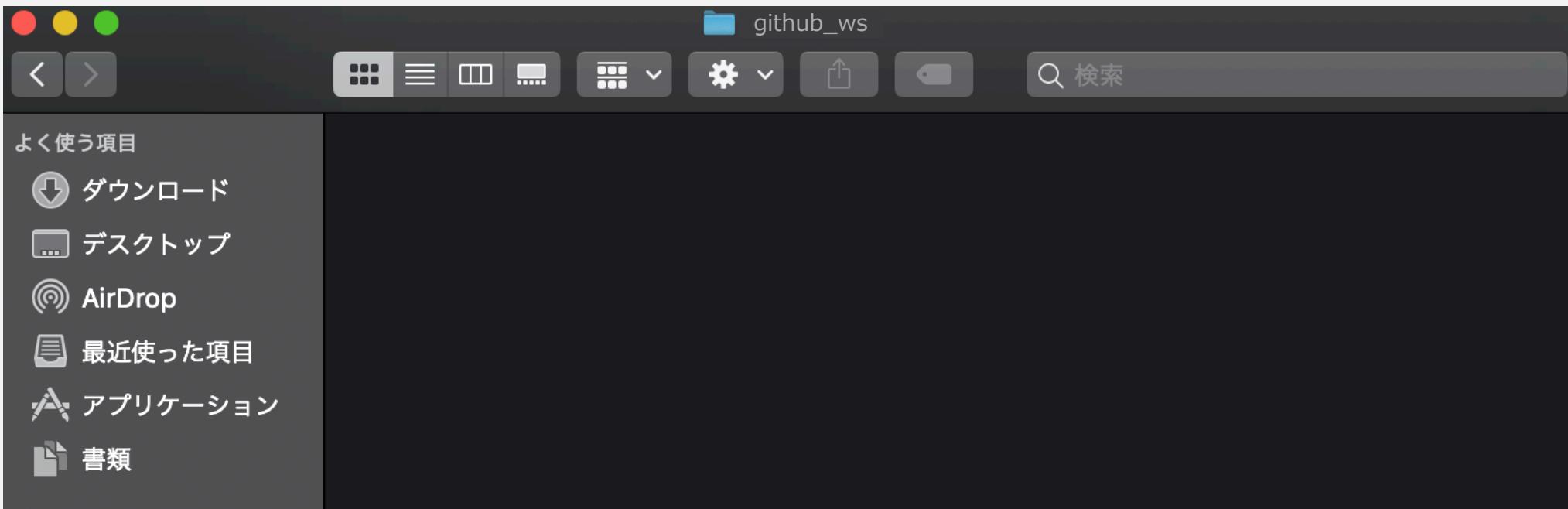


clone



ローカルリポジトリを確認する

- 先程指定したパスを開いてみましょう
 - ここが「ローカルリポジトリ」です
※ 中身は空のはず



ファイルを追加する

- GitHub の Search or jump to … から「hackujp/github_tutorial」を検索
- リポジトリ内の ver_GUI から下記ファイルを取得
 - index.html
 - src
 - index.js
 - styles.css

※ フォルダごとではなくファイルのみ

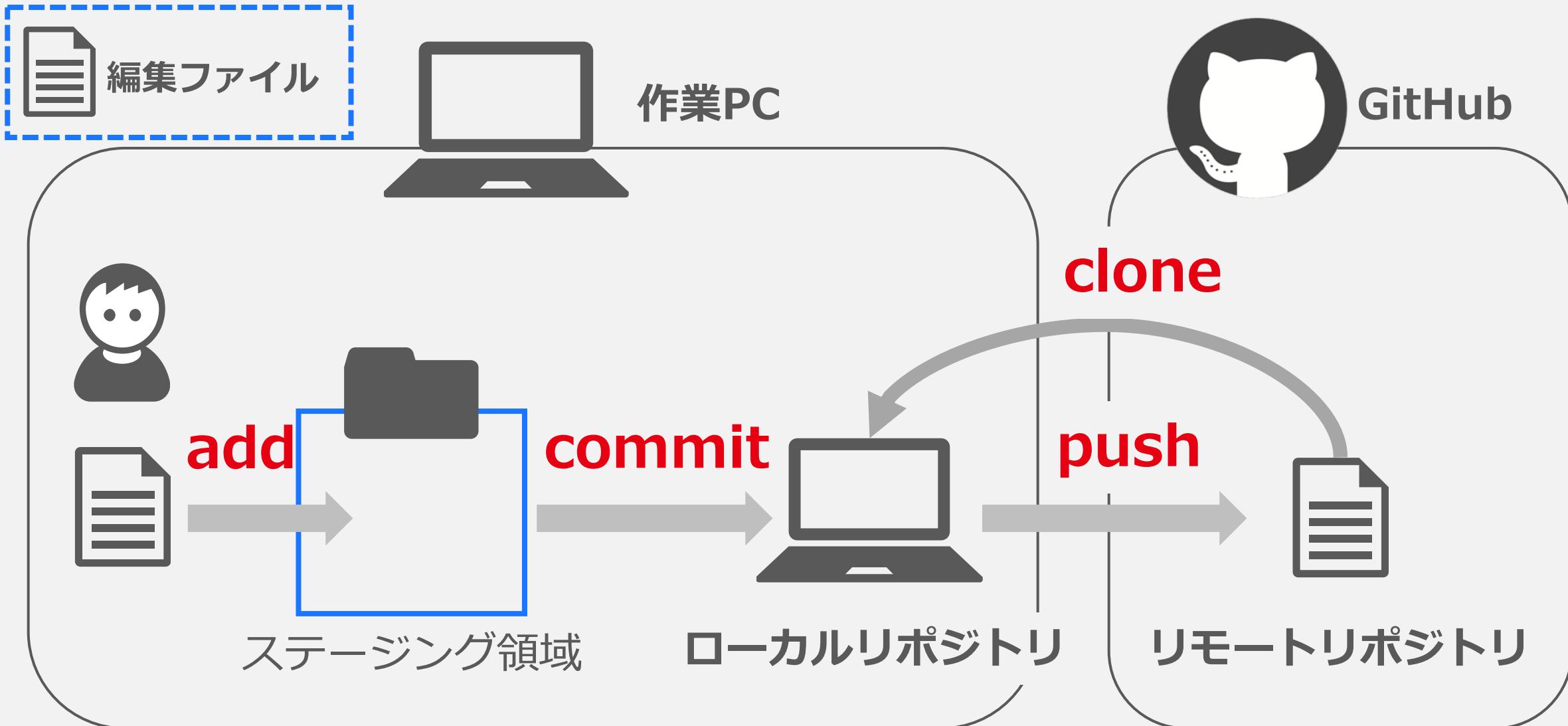


Step.1

GitHub の基本操作

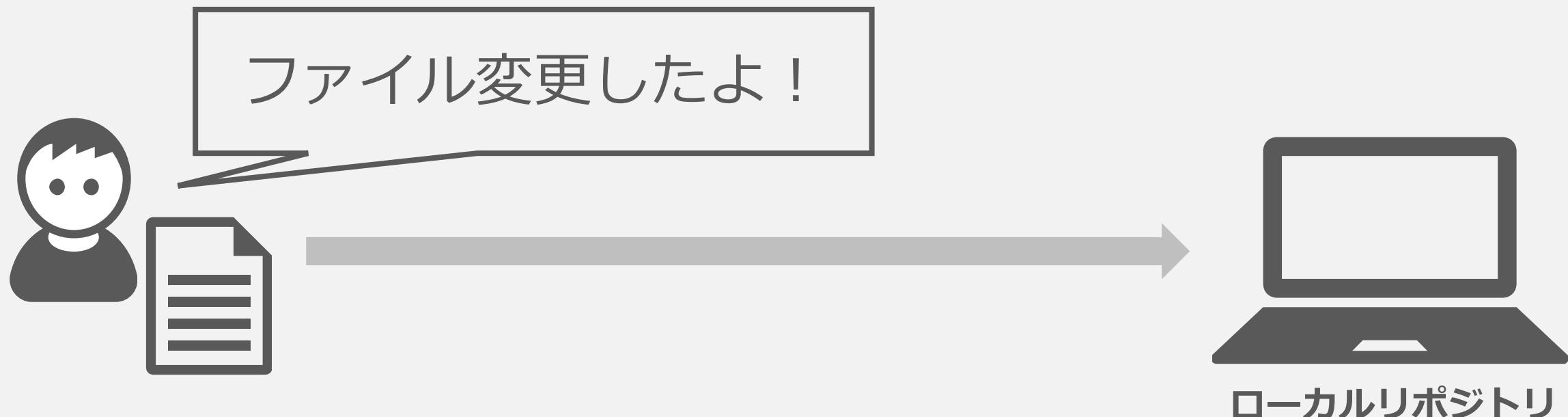


GitHub の基本操作

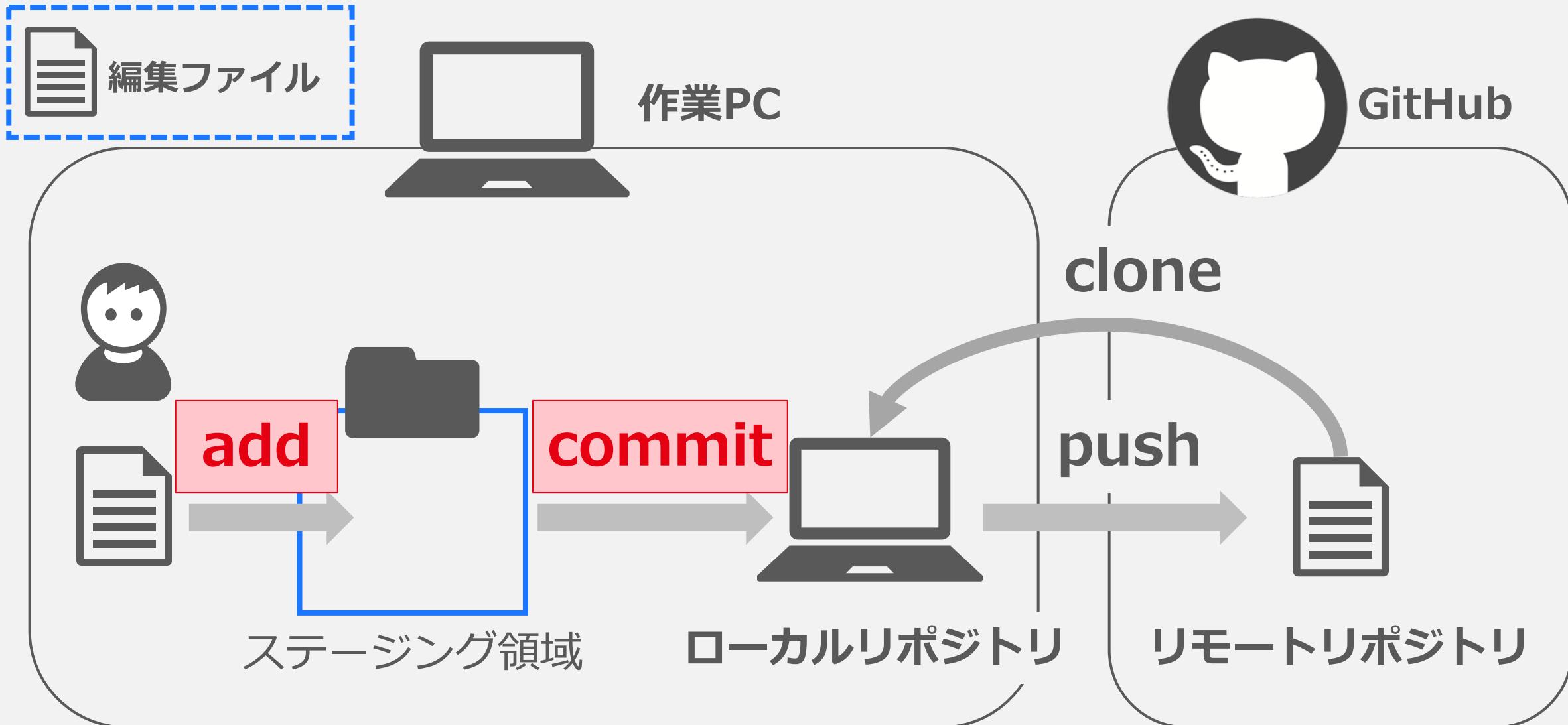


ファイルを修正してGitで扱う

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

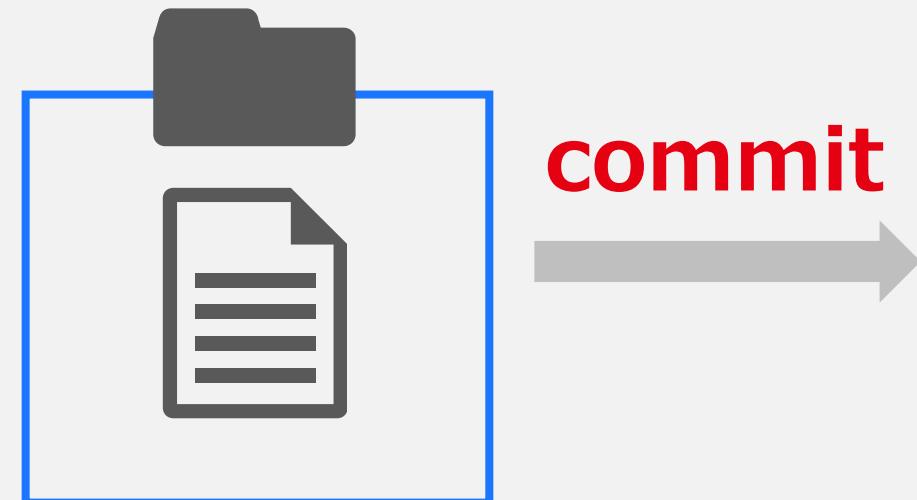
どのファイルの変更をしたかをステージング領域に一度登録する → **add** という



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

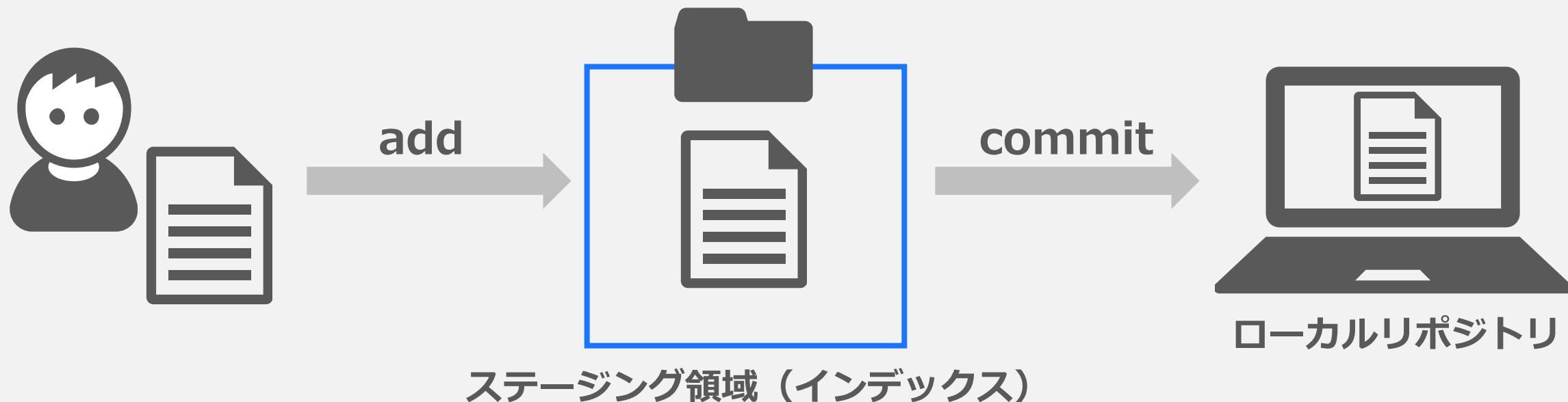


ステージング領域（インデックス）

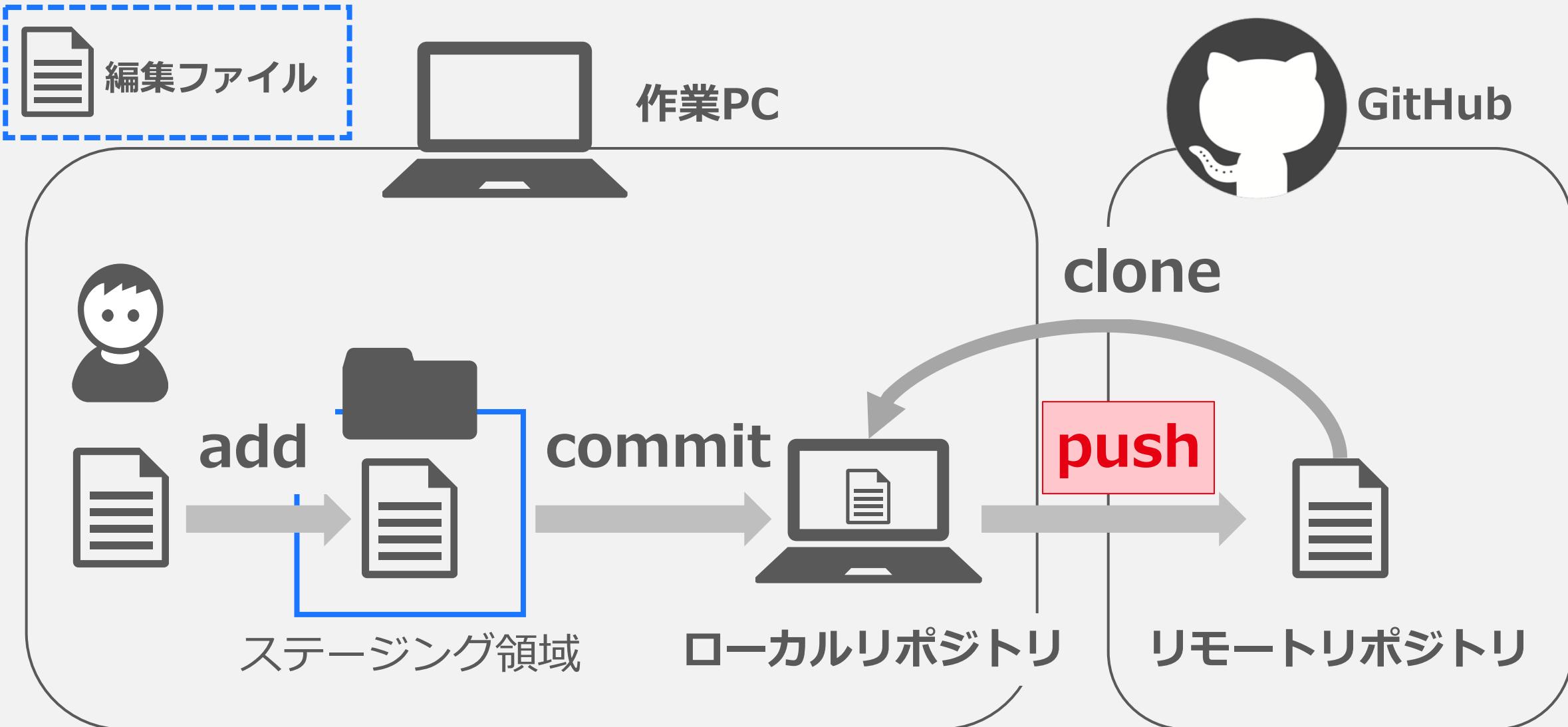


ファイルを修正してGitで扱う

- 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

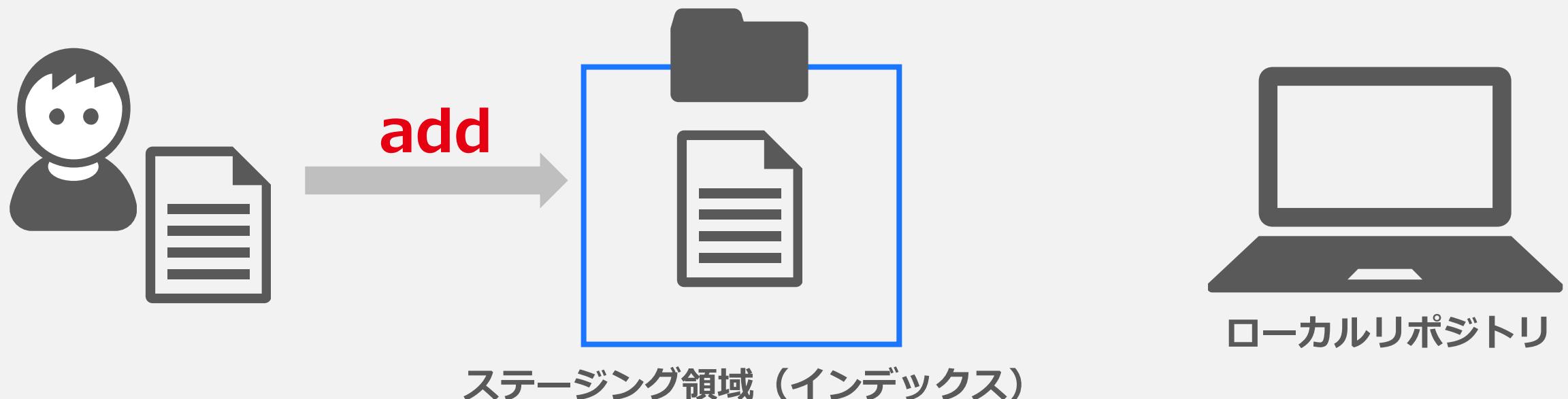
Sourcetreeで add する

YAHOO!
JAPAN

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

The screenshot shows the GitHub Desktop application interface. On the left is a sidebar with icons for Commit, Pull, Push, Fetch, Branch, Merge, Workspace, File Status (which is selected), History, Search, Branches, Tags, Remotes, origin, Stash, Submodule, and Subtree. At the bottom are Filter and Commit Message fields. The main area shows a file list with three files: index.html, src/index.js, and src/styles.css. The first file, index.html, is highlighted with a red background and has a blue border around it, indicating it is selected or being staged. To the right is a code editor window titled "github ws (Git)" showing the following HTML and JavaScript code:

```
<h2>演習で自由に編集してください</h2>
<!-- 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 -->
<!--<div id="target">ここをクリックするとアラート発生</div>-->
<!--<div id="ok">GitHub完全に理解した</div>-->
<script src=".src/index.js"></script>
</body>
</html>
```

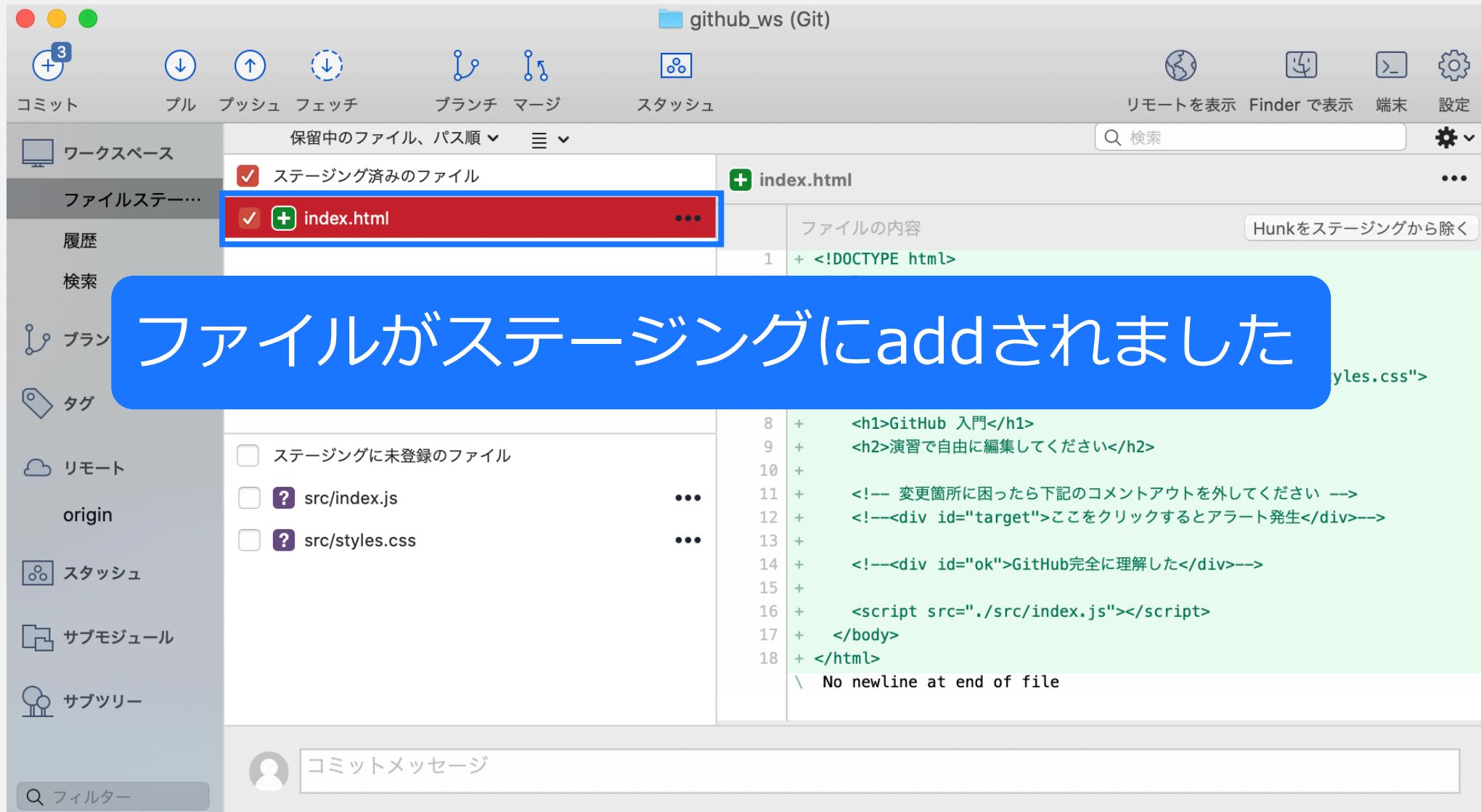
A large blue callout bubble on the right side contains the following text:

各ファイルを

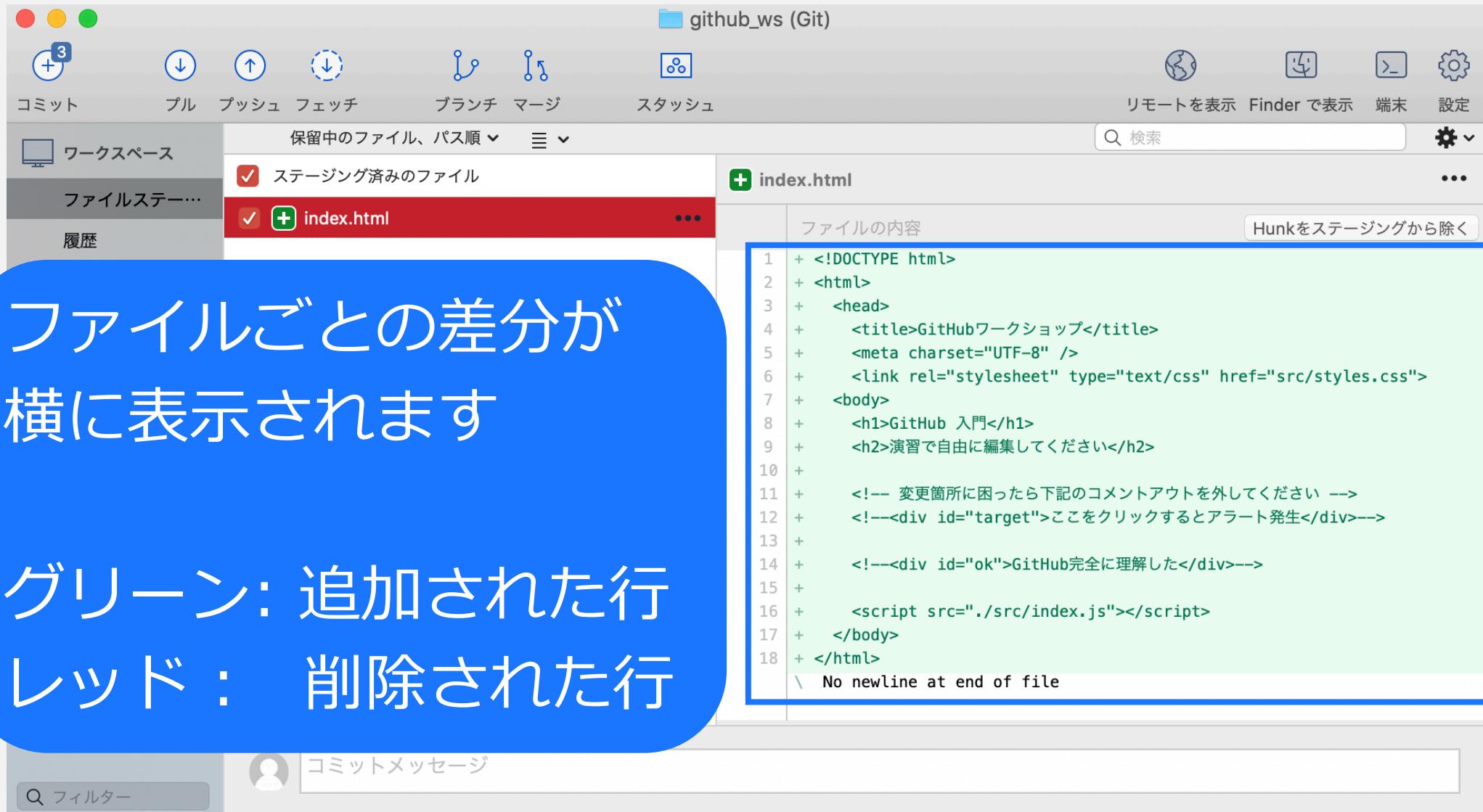
- ・チェックする
- ・上にドラッグ&ドロップする

のいずれかの操作で add される

ステージングにaddする



ステージングにaddする



Sourcetree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



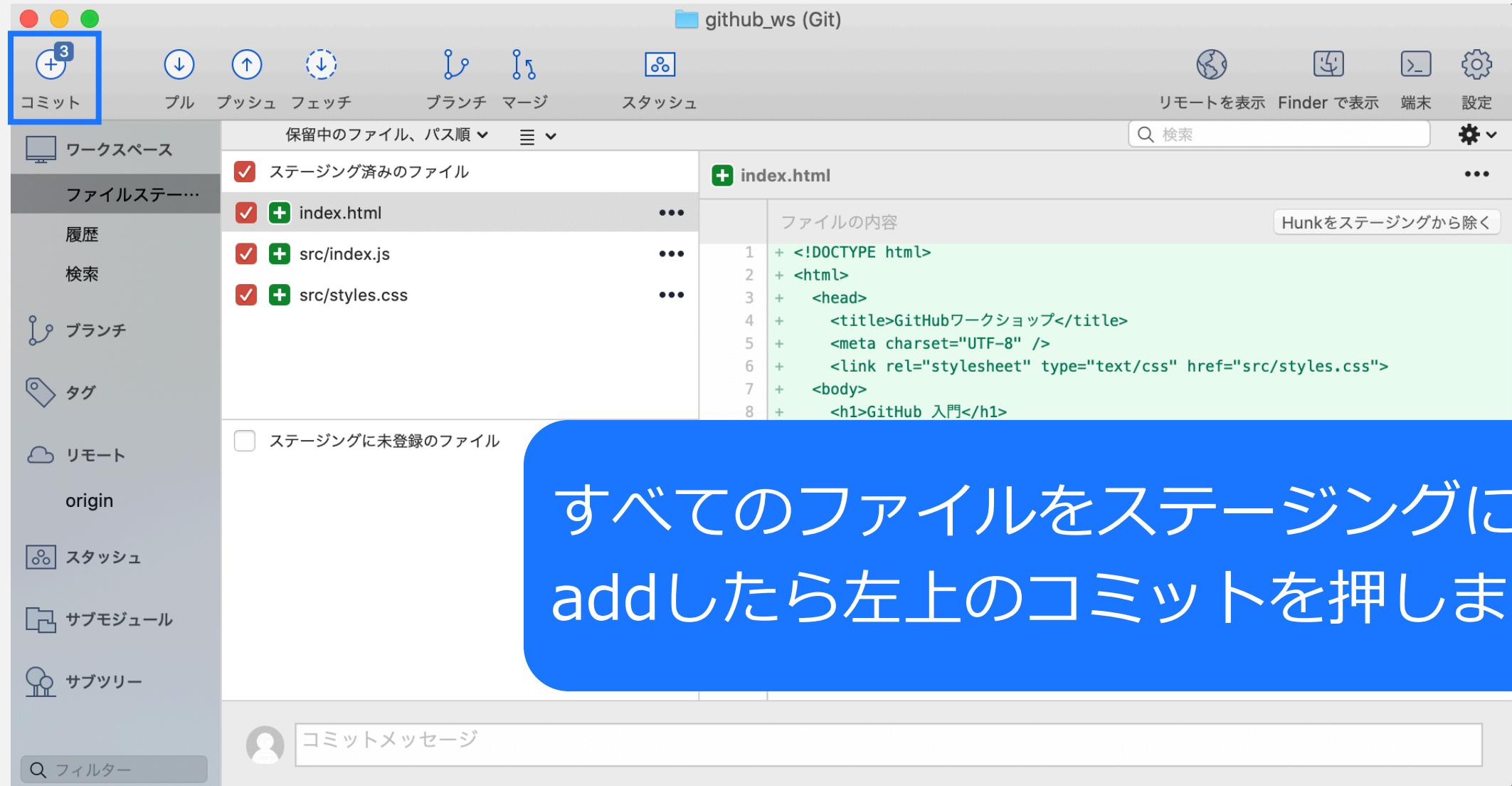
commit



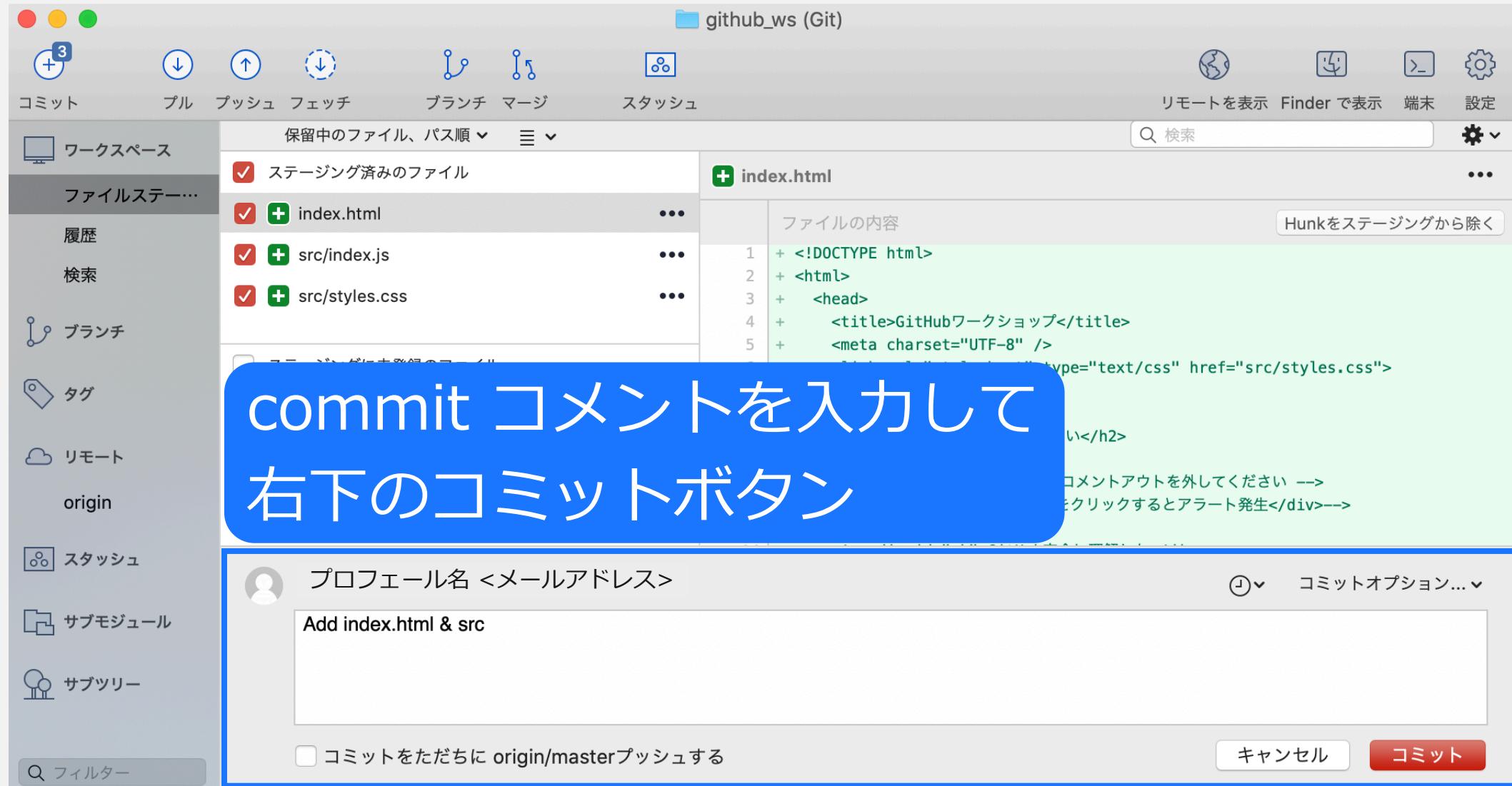
ローカルリポジトリ

ステージング領域（インデックス）

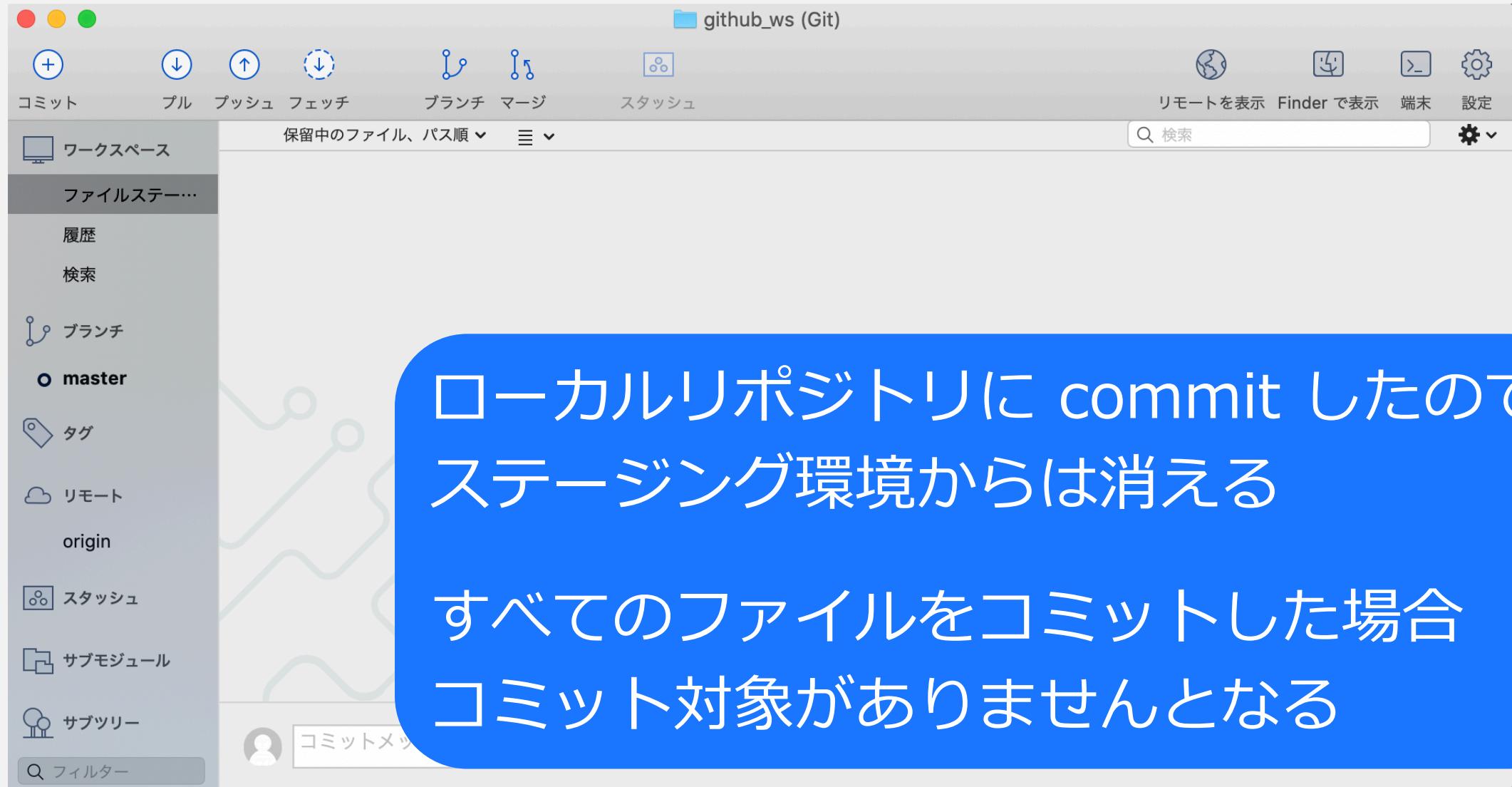
ローカルリポジトリにcommitする



ローカルリポジトリにcommitする



ローカルリポジトリにcommitする



Sourcetreeで push する



自分のPCからGithubに反映する

push :

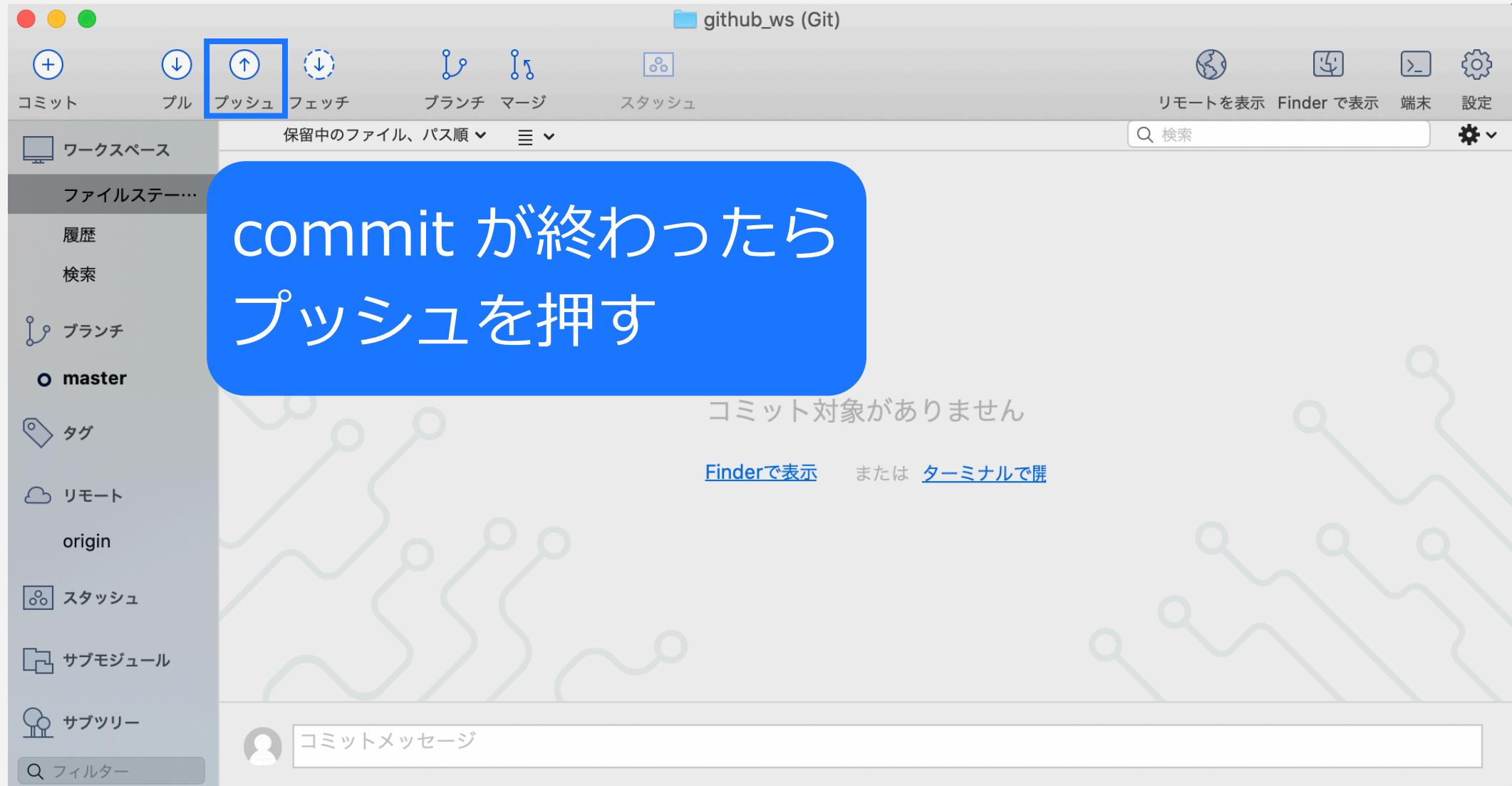
ローカルリポジトリの変更をリモートリポジトリに反映



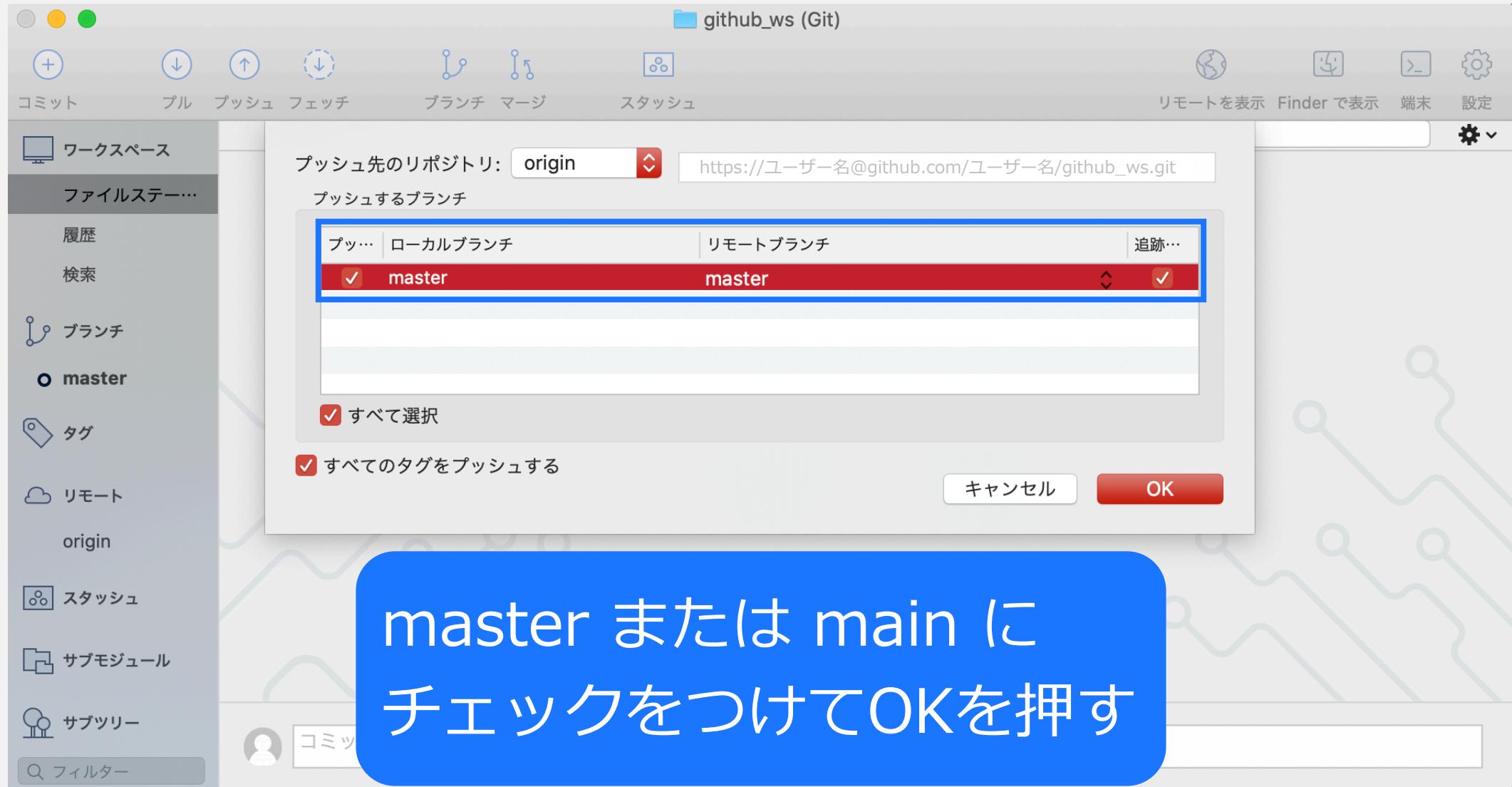
ローカルリポジトリ

リモートリポジトリ

リモートリポジトリにpushする



リモートリポジトリにpushする



master または main に
チェックをつけてOKを押す

Github に反映されているか確認

Screenshot of a GitHub repository page for a user named "github_ws".

The repository details:

- Code: master branch, 1 branch, 0 tags
- Last commit: 50cdc45, 14 minutes ago, 1 commit
- Commit details:
 - src: Add index.html & src, 14 minutes ago
 - index.html: Add index.html & src, 14 minutes ago
- File status: 1 file added
- Contributors: 1 commit from 1 author

Repository statistics:

- Unwatched (1)
- Starred (0)
- Forked (0)

About section:

No description, website, or topics provided.

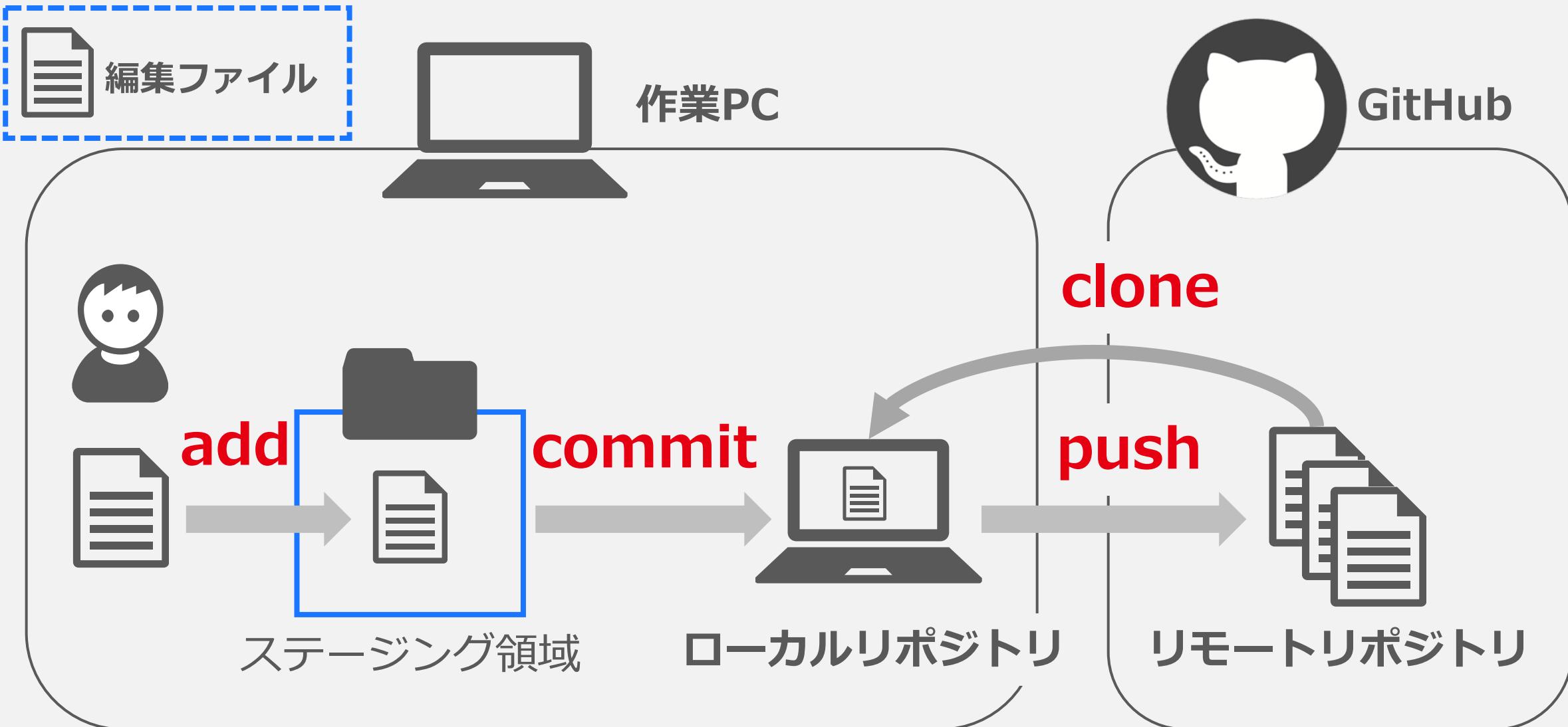
Releases section:

No releases published

Add a README

Header navigation: Pull requests, Issues, Marketplace, Explore, Notifications, +, Profile.

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 の作成 / index.html を編集する
 - 2. 作成/変更したファイルを add する
 - 変更したファイルの差分を確認しよう

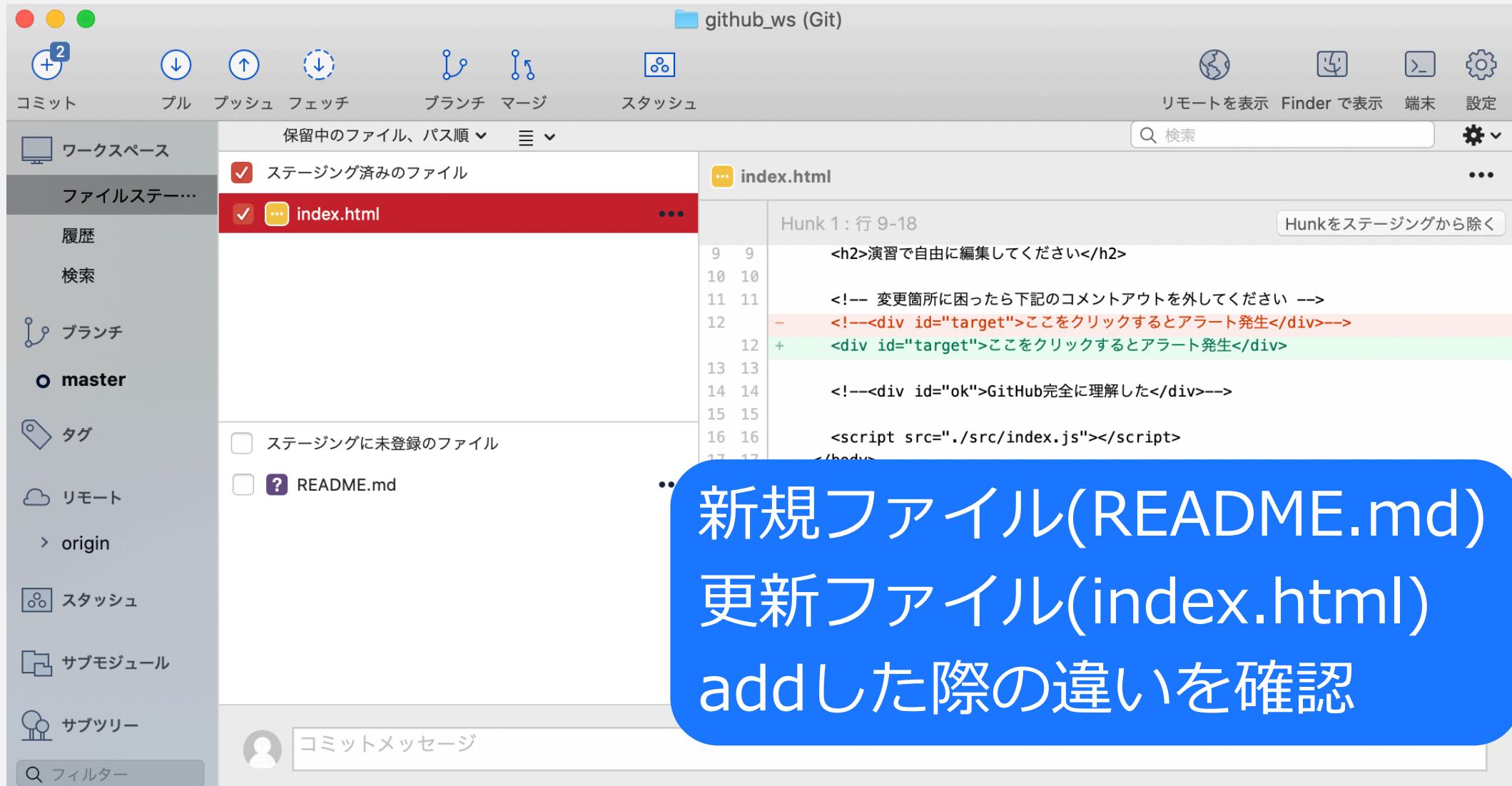
ハンズオンの作業内容 その1

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
 - コミットツリーを確認してみよう

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
- **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

ファイル差分の確認

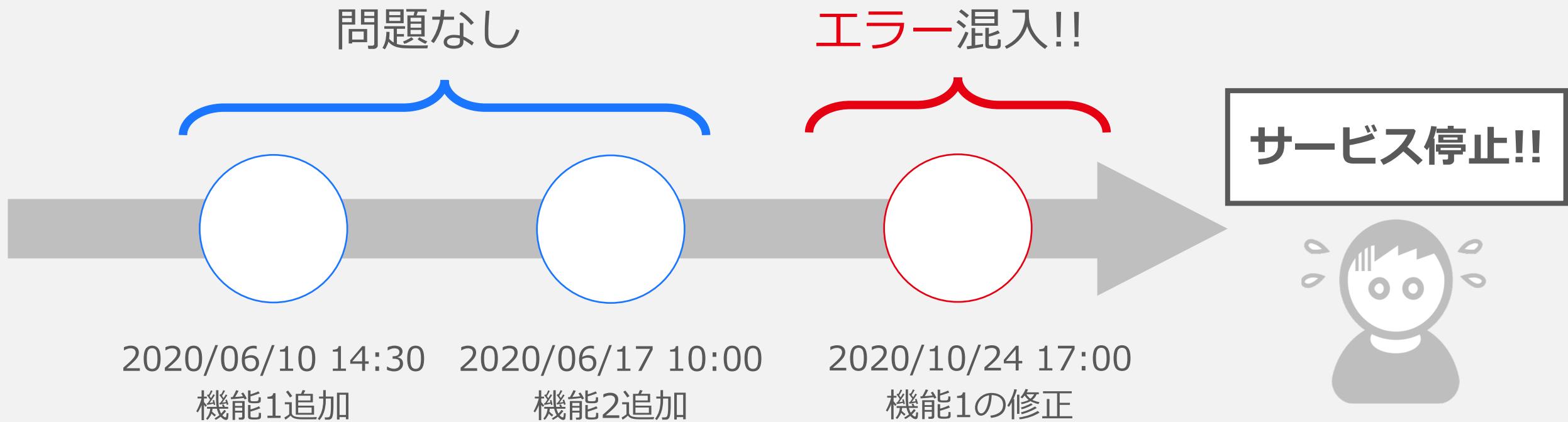


Step.2-1

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

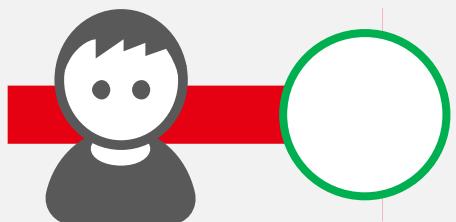
機能2追加

機能1の修正

止!!

ブランチ

最新の状態



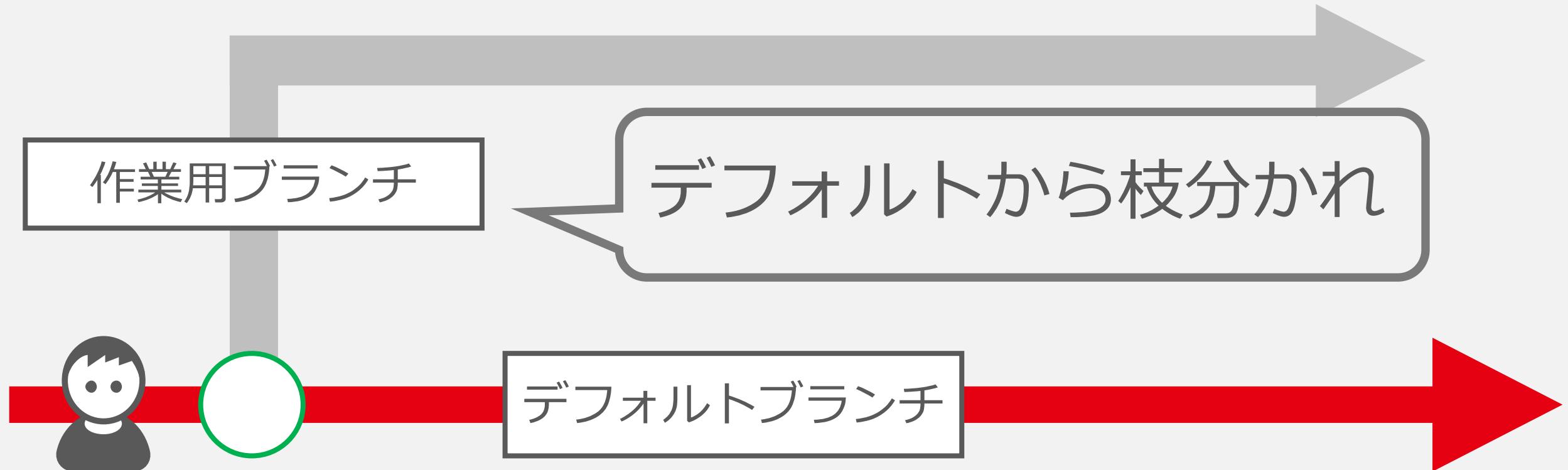
大元のブランチ

→ 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 `master` を使用

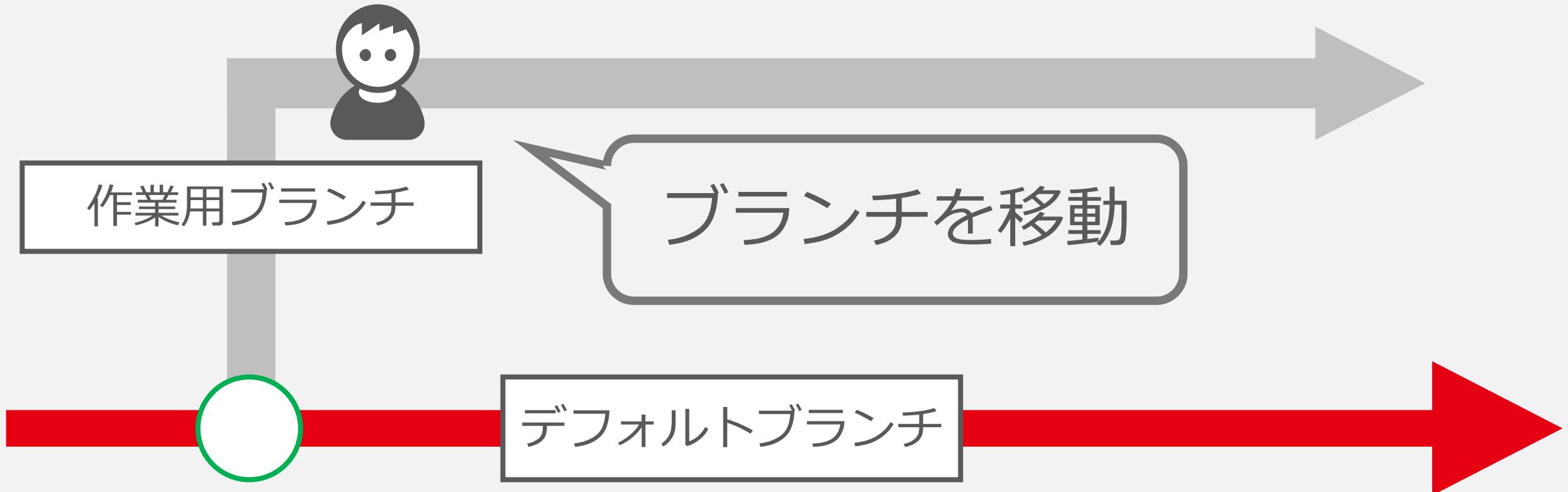
ブランチ

作業用ブランチを作成する



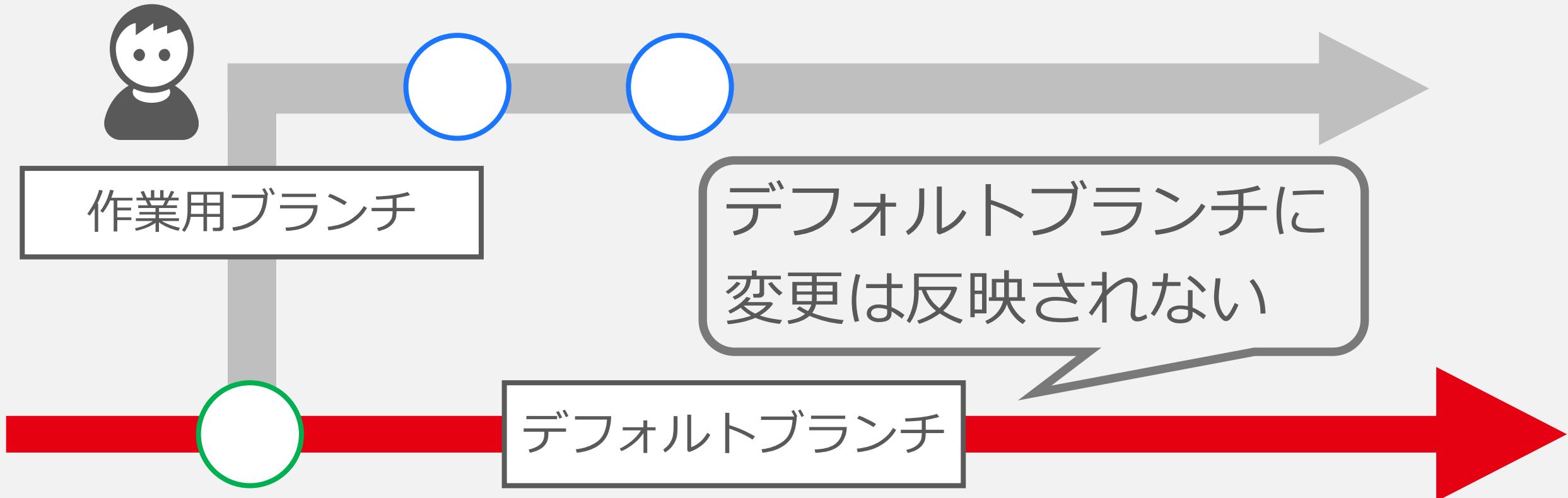
ブランチ

作業用ブランチにチェックアウトをする



ブランチ

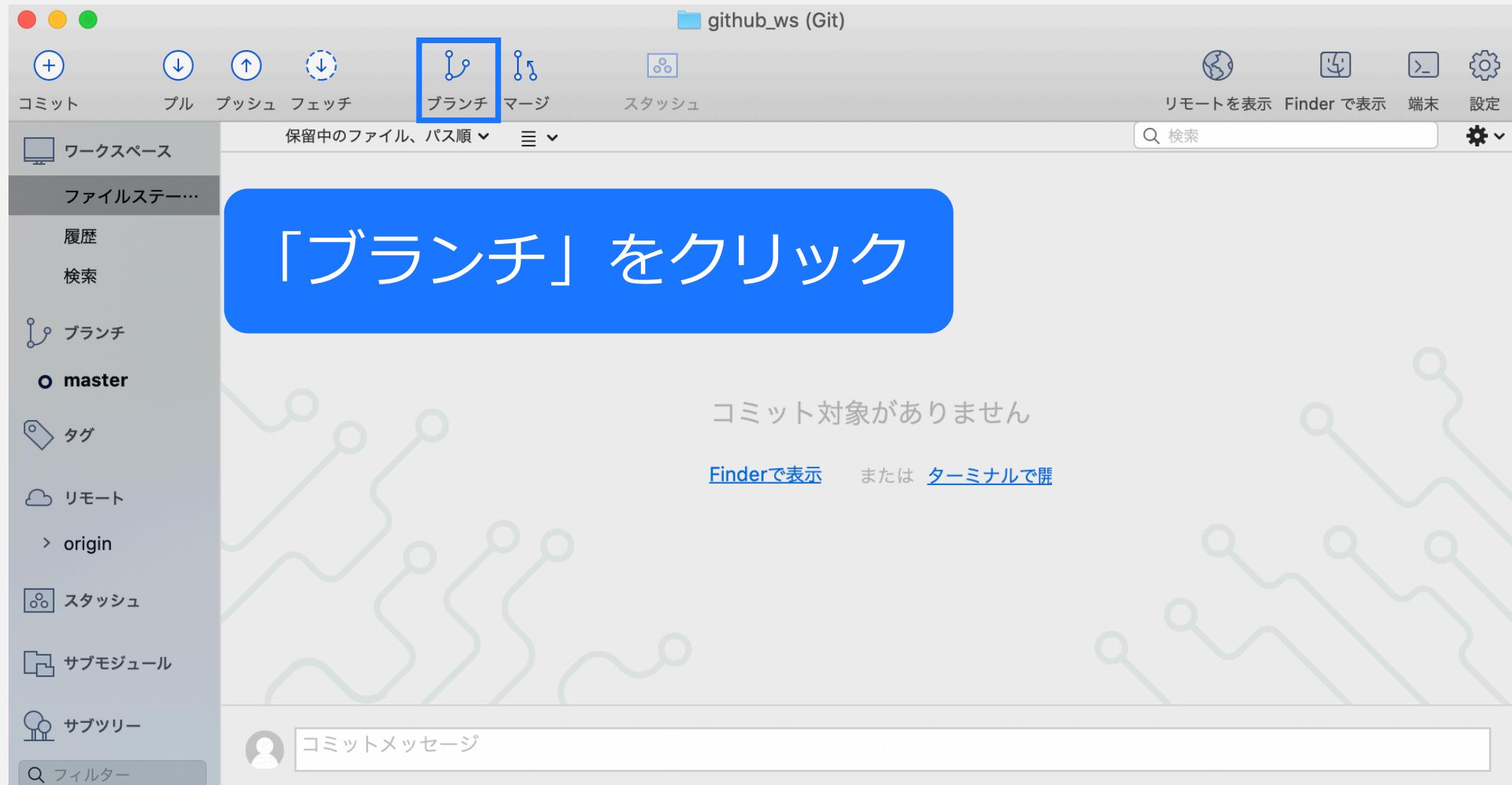
作業ブランチで同じように commit/push する



Sourcetreeで ブランチを作成する



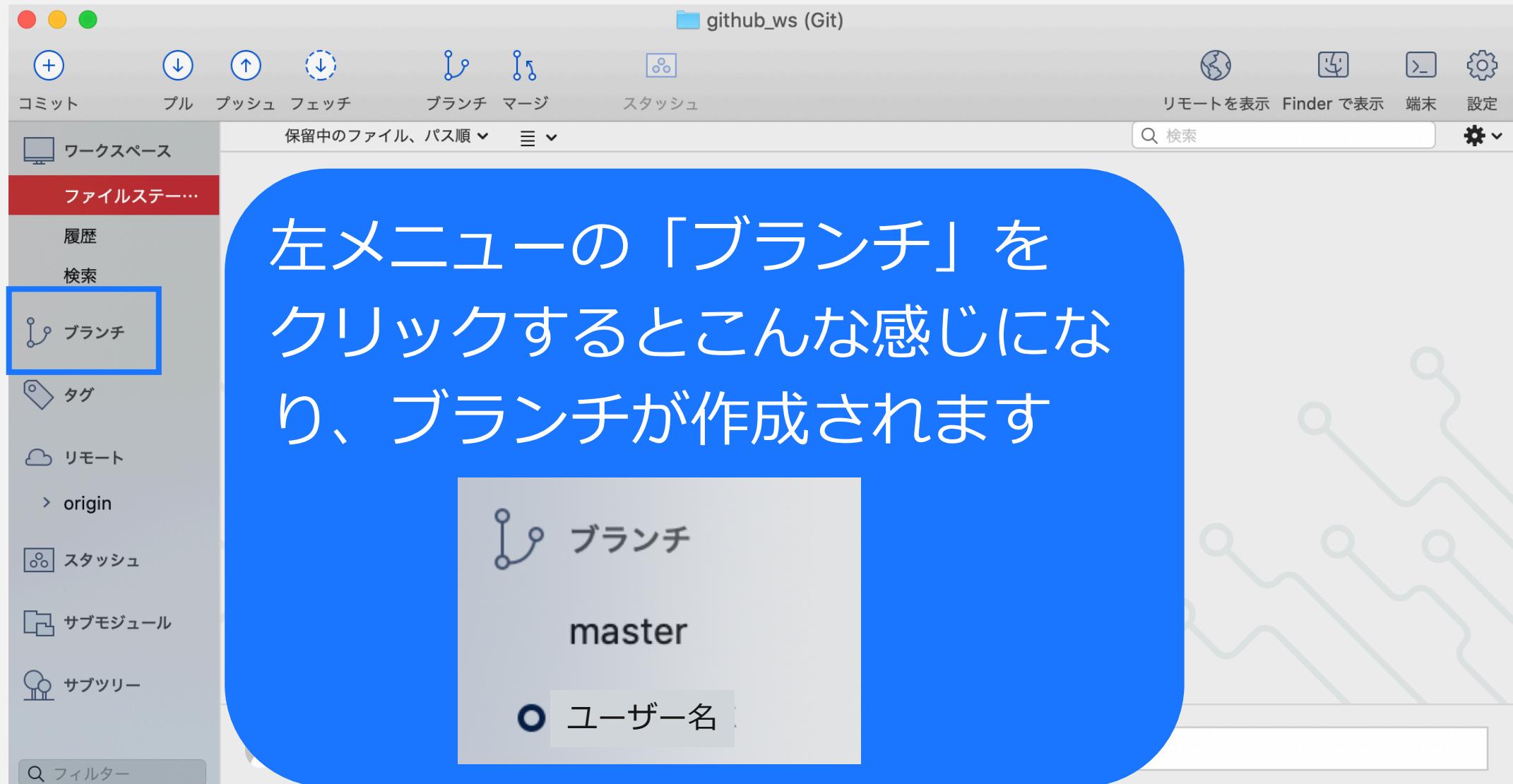
ブランチを作成する



ブランチを作成する



ブランチを作成する



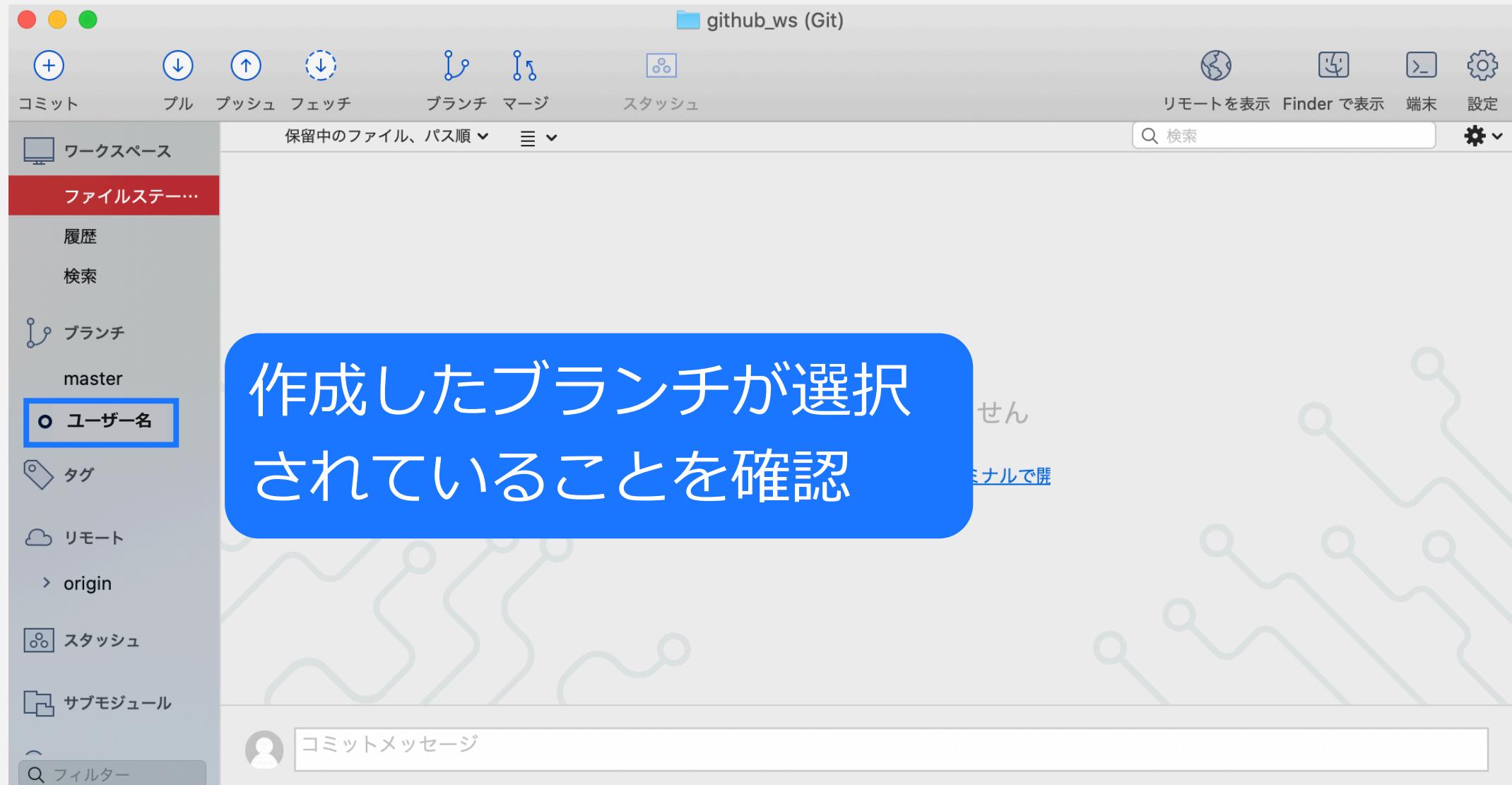
ブランチを指定して
ファイル変更



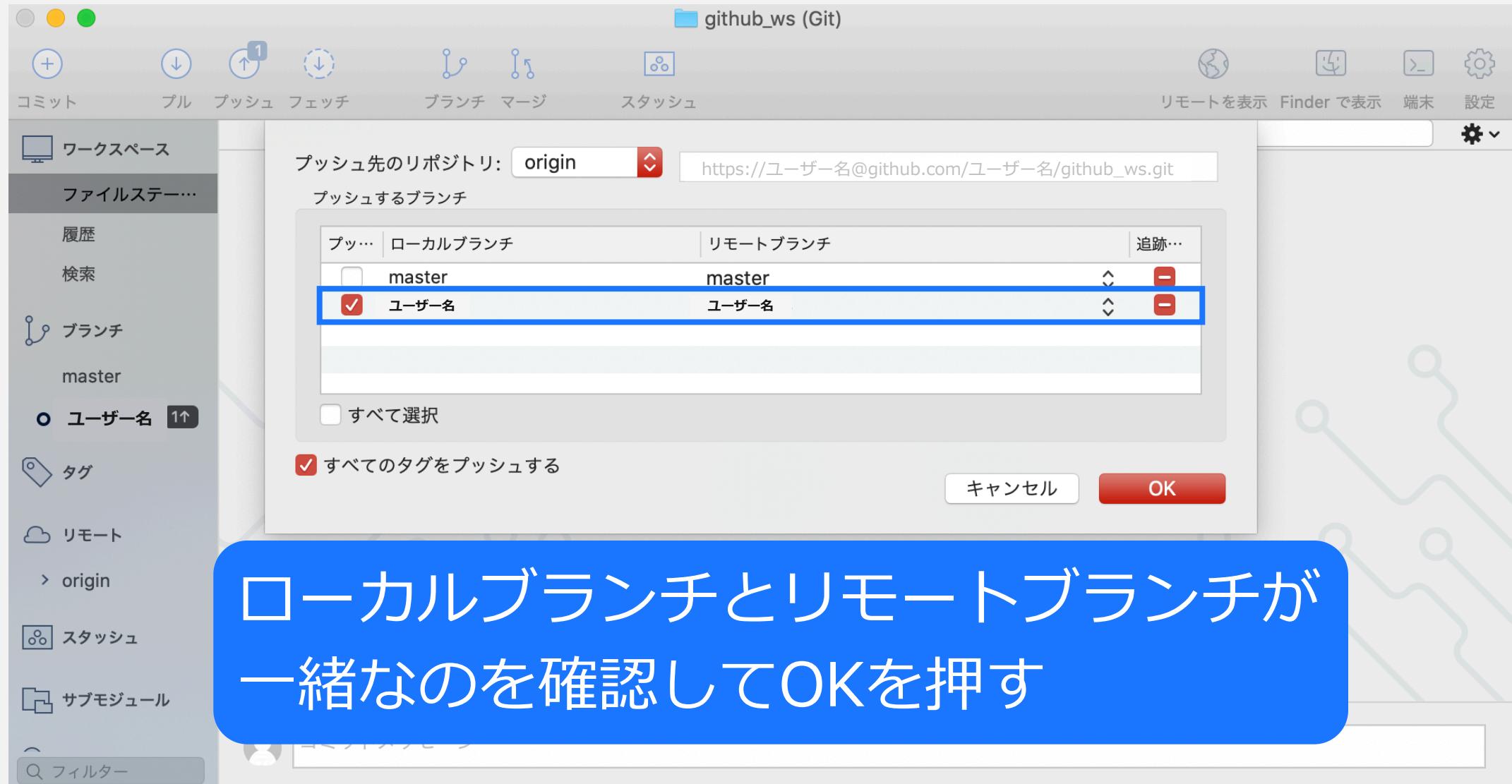
ブランチを作成してファイルを Push

- ファイルを変更して add → commit
 - この時リポジトリが master では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
 - リモートブランチに同じ名前で Push
 - master にブランチに Push しないように

ファイル変更前にブランチを確認



任意ブランチに Push する



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, Issues, Pull requests, Actions, Projects, Wiki, and Security. The 'Code' tab is highlighted with a blue border and a red underline. Below the navigation bar, there is a yellow banner with a user icon and the text 'ユーザ名 had recent pushes less than a minute ago'. To the right of the banner is a green button labeled 'Compare & pull request'. Underneath the banner, there are several buttons: a dropdown menu set to 'master', a link to '2 branches', a link to '0 tags', 'Go to file', 'Add file', and a 'Code' dropdown menu. A modal window titled 'Switch branches/tags' is open, containing a search bar with the placeholder 'Find or create a branch...' and tabs for 'Branches' and 'Tags'. The 'Branches' tab is selected and highlighted with a blue border. Below the tabs, there is a list of branches: 'master' (marked with a checkmark) and 'ユーザ名'. At the bottom of the modal, there is a blue button labeled 'View all branches'. In the bottom right corner of the modal, there is a small edit icon. The overall background of the page is white.

Code → master をクリック

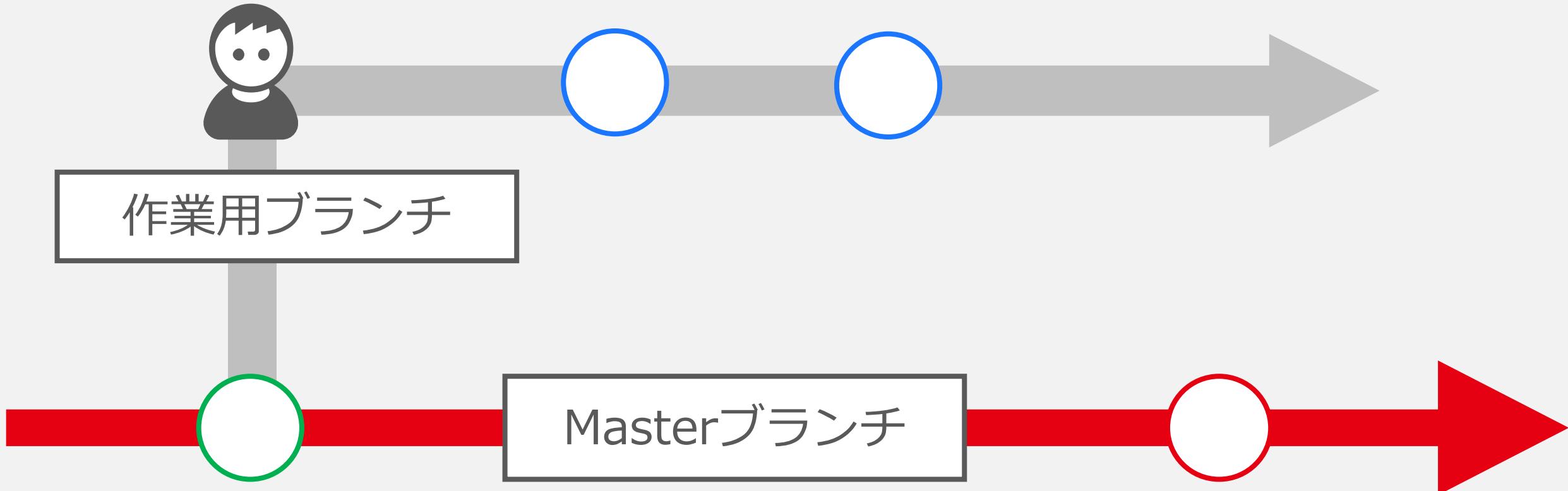
Branches にアカウント名があればOK

Step.2-2

masterに変更を反映する

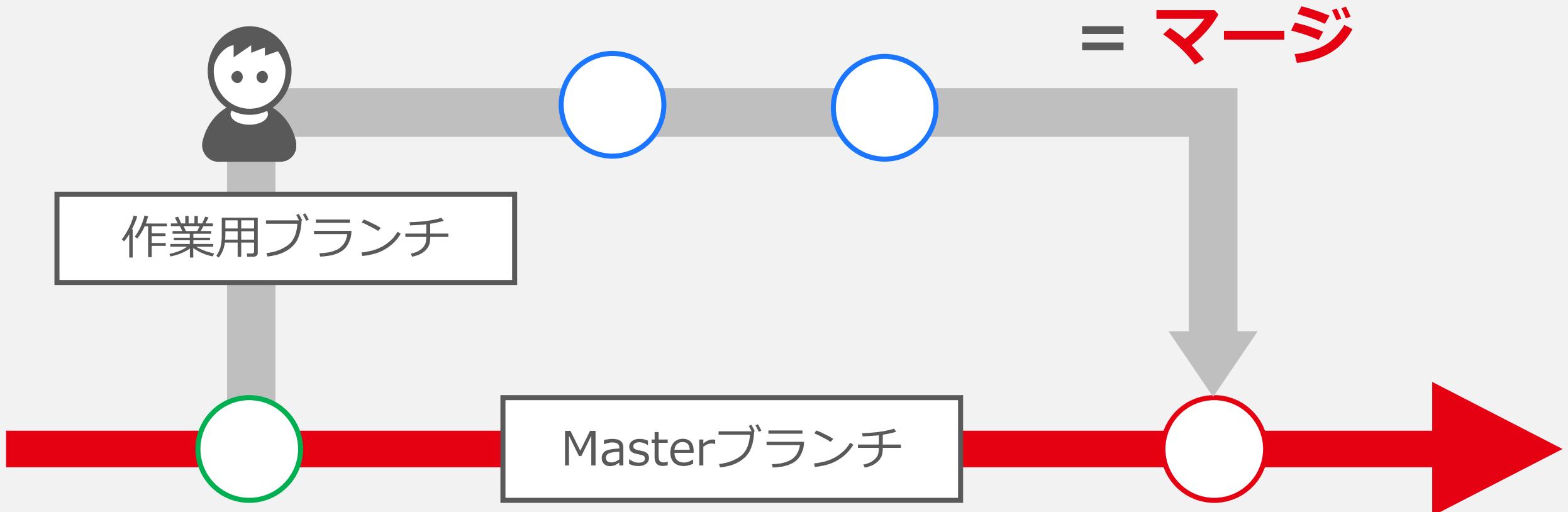
マージ

作業用ブランチの変更をMasterに反映する



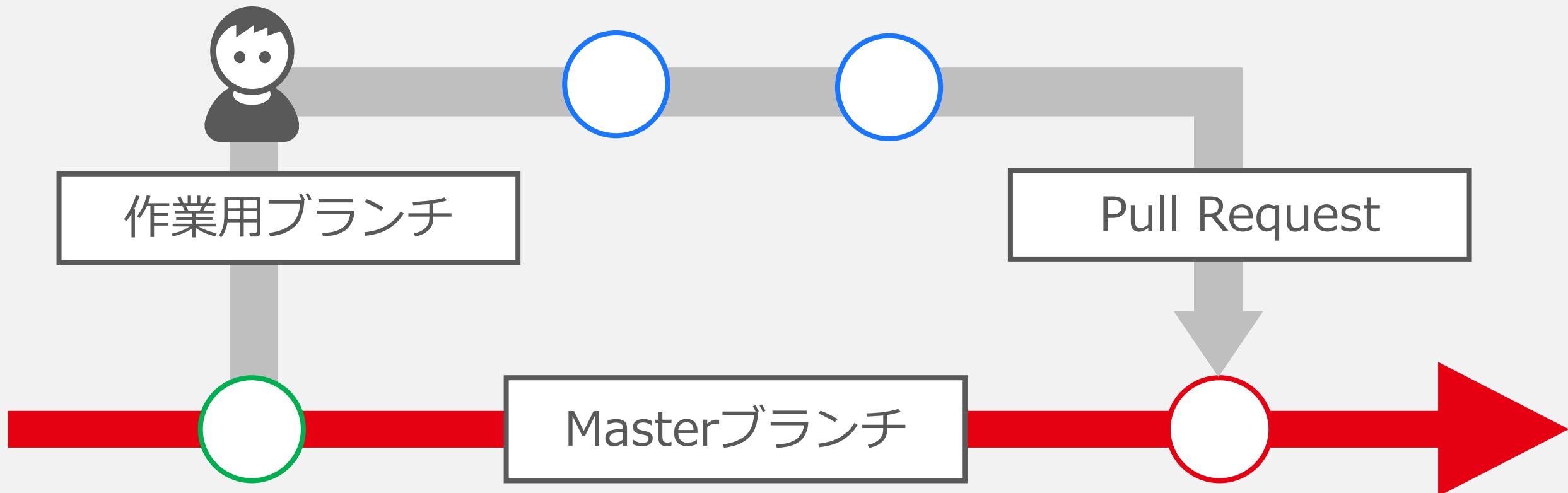
マージ

作業用ブランチの変更をMasterに反映する



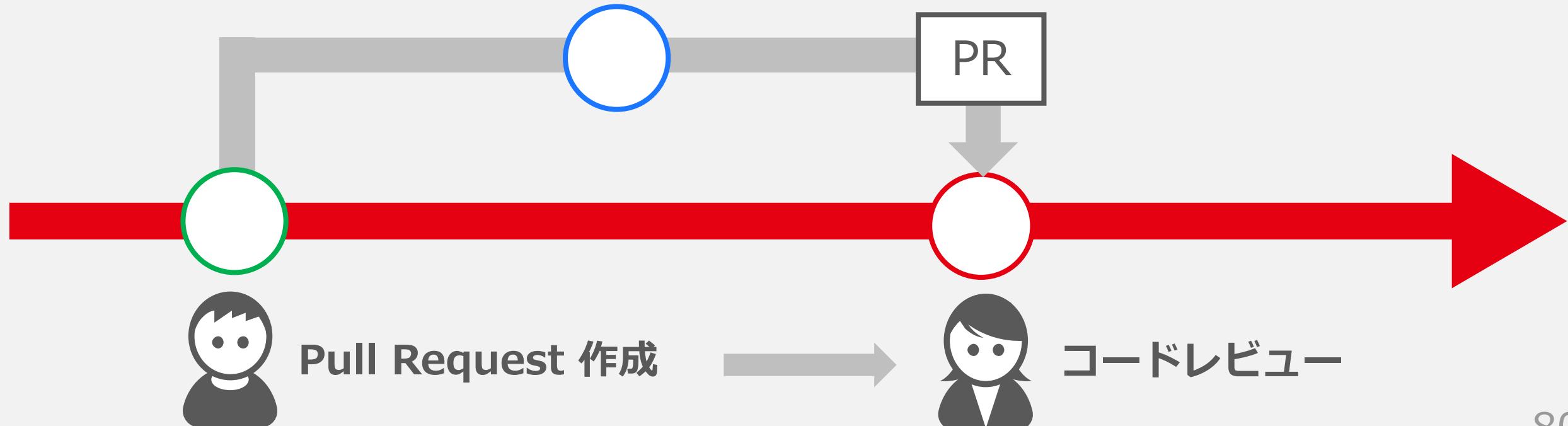
マージ

マージするために Pull Request 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 Requests (PR) を作成
 - 追加/変更した内容を記載
- Pull Requests 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 master 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 interface. 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 "master" (with a dropdown arrow), "2 branches", "0 tags", "Go to file", "Add file" (with a dropdown arrow), and "Code" (with a dropdown arrow). The main content area shows a list of commits. The first commit is from "ユーザーネーム" at "a2b7d68 39 minutes ago", adding "README.md" and "2 commits". The second commit is at "1 hour ago", adding "index.html & src" to the "src" folder. A blue callout bubble with white text overlays the bottom-left portion of the commit list, containing the Japanese text: "GitHub にいくとこのような画面が出てきます (青枠注文)".

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

Compare & pull request

master ▾ 2 branches 0 tags Go to file Add file ▾ Code ▾

ユーザーネーム Add README.md a2b7d68 39 minutes ago 2 commits

src Add index.html & src 1 hour ago

README.md index.html

GitHub にいくとこのような画面が
出てきます (青枠注文)

README.md

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

master ▾ 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

File	Commit Message	Time
src	Add index.html & src	1 hour ago
README.md	Add README.md	39 minutes ago
index.html	Add README.md	39 minutes ago

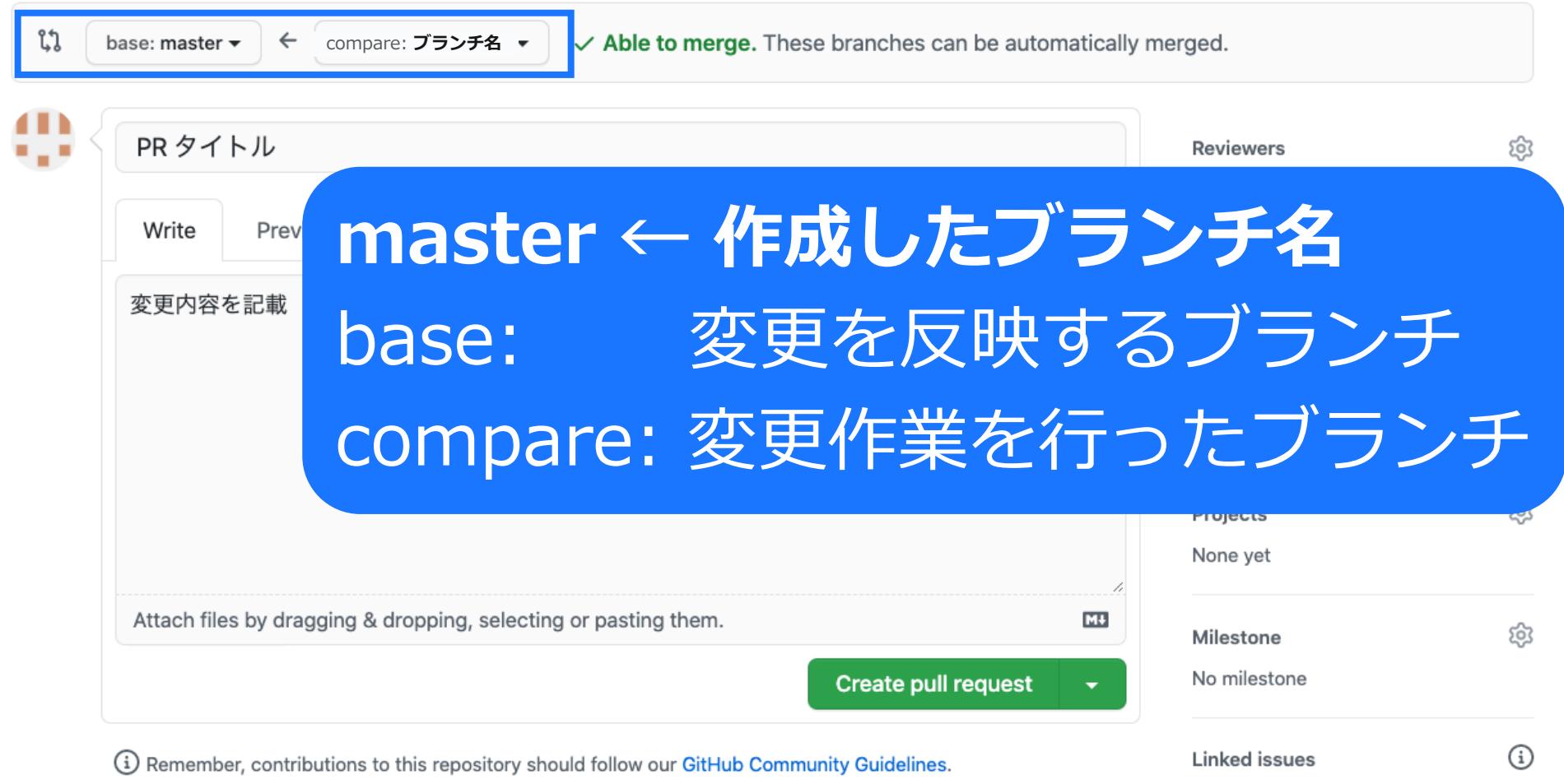
README.md

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: master' and 'compare: ブランチ名'. A green checkmark indicates 'Able to merge. These branches can be automatically merged.' Below this, a large blue callout box contains Japanese text explaining the parameters:

- master ← 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 includes fields for 'PR タイトル' (Title), 'Write' (content area), 'Reviewers', 'Projects' (None yet), 'Milestone' (No milestone), and 'Linked issues'. A 'Create pull request' button is at the bottom.

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, there are dropdown menus for 'base: master' 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 input field containing 'PR タイトル', which is highlighted with a blue border. To the left of the title field is a red octocat icon. To the right are sections for 'Reviewers' (No reviews), 'Assignee' (Assign yourself), 'Projects' (None yet), 'Milestone' (No milestone), and 'Linked issues'. A large blue callout box is overlaid on the interface, containing the Japanese text 'Pull Request のタイトル' and '→ 概要がひと目でわかるように', explaining the purpose of the title field.

PR タイトル

Pull Request のタイトル
→ 概要がひと目でわかるように

base: master ▾ ← 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

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

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: master" 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" tabs. The main body area has a placeholder "変更内容を記載". To the right, there are sections for "Reviewers" (No reviews), "Assignees" (No one—assign yourself), "Labels" (None yet), and "Milestone" (No milestone). At the bottom, there's a large blue button with the Japanese text "タイトルと概要入力したらクリック" (Click after entering title and summary). The "Create pull request" button 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 master 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
- 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 Conflicts:** This branch has no conflicts with the base branch. Merging can be performed automatically.
- Actions:** Merge pull request (button) or open this in GitHub Desktop or 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 master 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

3 README.md

2 index.html

Files changed に変更差分が掲載される

...
11	11	11
12	12	12
13	13	13
14	-	-><div id="ok">GitHub完全に理解した</div><-->
	14	+><div id="ok">GitHub完全に理解した</div>
15	15	15
16	16	<script src=".src/index.js"></script>
17	17	</body>

Pull Request の確認

A screenshot of a GitHub Pull Request review interface. At the top, there's a commit history showing a single commit: "Update index.html" with hash "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 is followed by a message: "Continuous integration has not been set up" with a note that "GitHub Actions and several other apps can be used to automatically catch bugs and enforce style." Another message indicates "This branch has no conflicts with the base branch" and "Merging can be performed automatically." A green button labeled "Merge pull request" is visible. Below this, a comment section is shown with a "Write" tab active. The text area contains the Japanese text "レビューや気になった内容はコメントできる". A placeholder at the bottom says "Attach files by dragging & dropping, selecting or pasting them." At the bottom right are two buttons: "Close with comment" and a larger green "Comment" button.

Update index.html
78149a0

Add more commits by pushing to the ユーザー名 branch on ユーザー名 / github_ws.

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

Write Preview

レビューや気になった内容はコメントできる

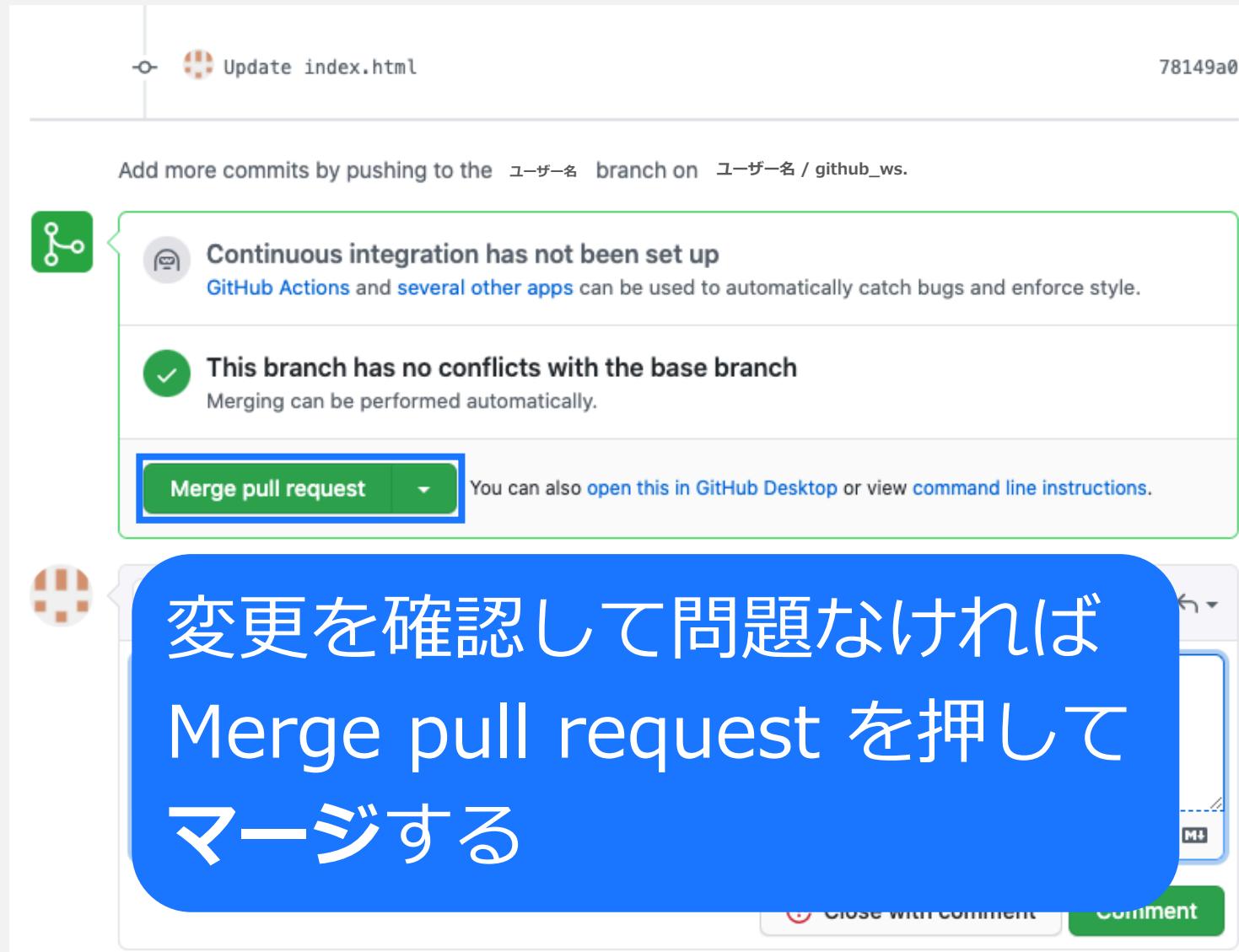
Attach files by dragging & dropping, selecting or pasting them.

Close with comment Comment

Pull Request の確認



Pull Request をマージ



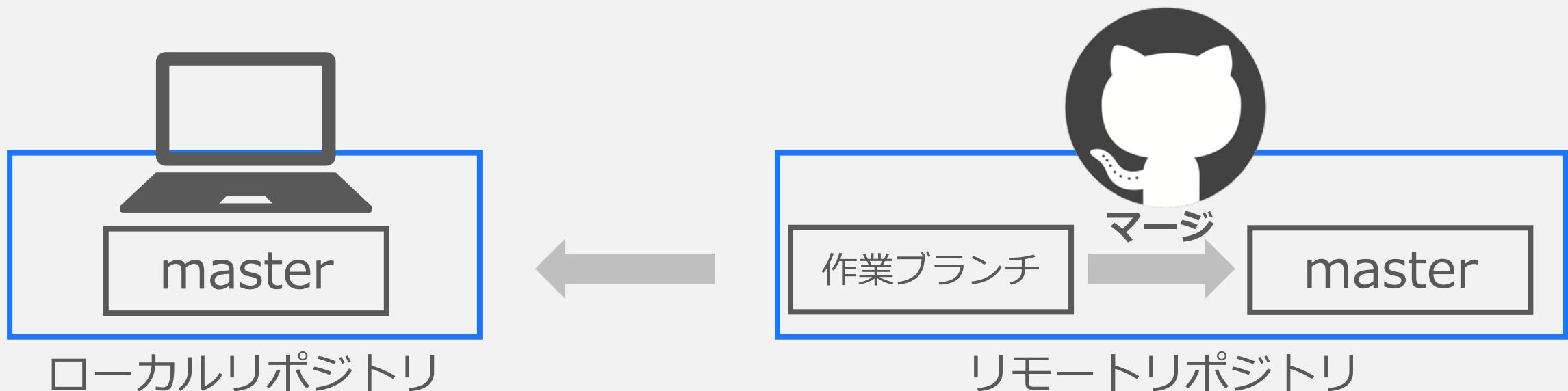
ローカルリポジトリに
Pull する



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

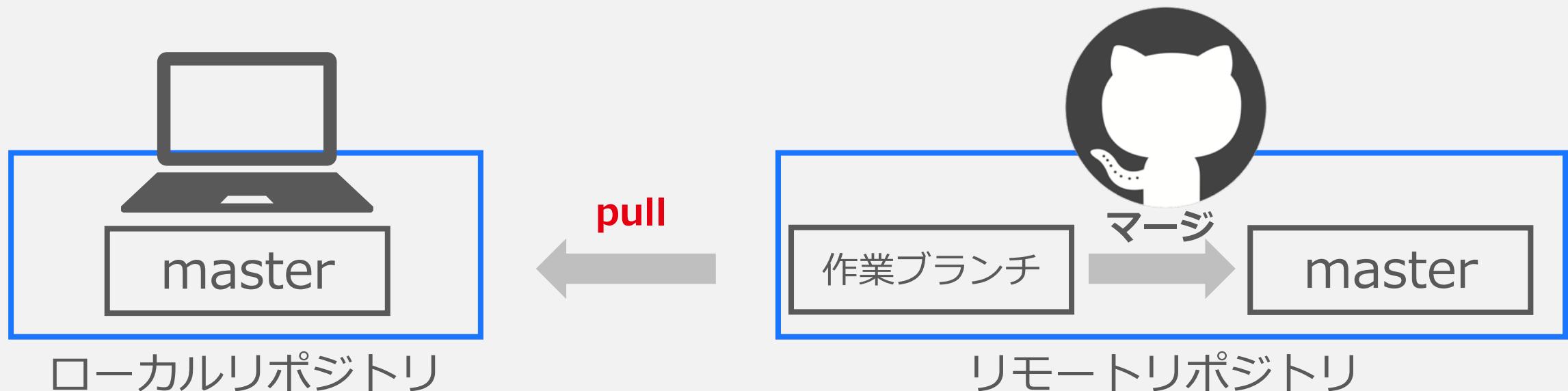
- ・ リモートの master には反映される
- ・ ローカルの master には反映されていない



マージされたものをローカルに反映

リモートの master の内容をローカルの master に反映

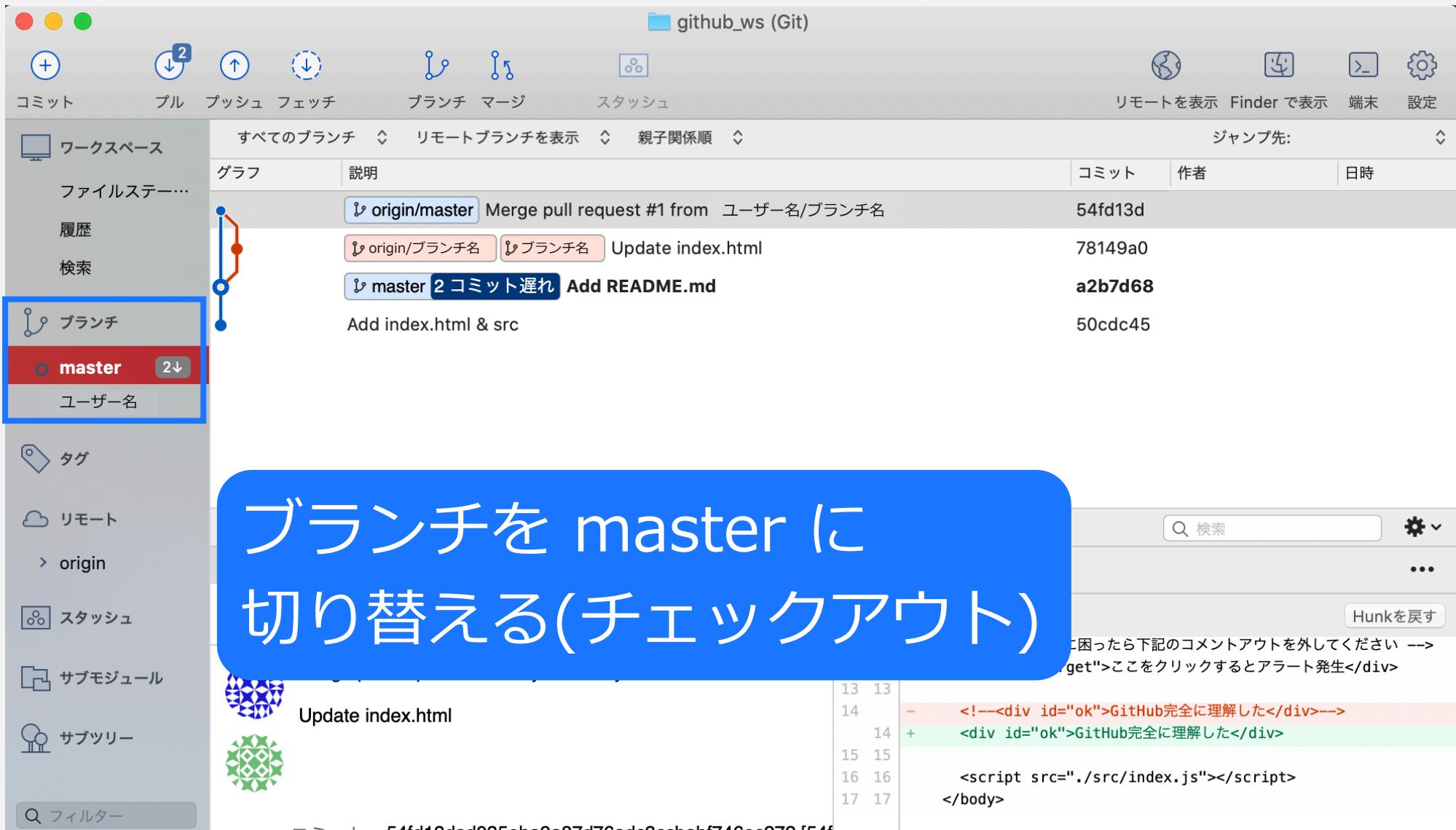
→ ローカルに **pull** する



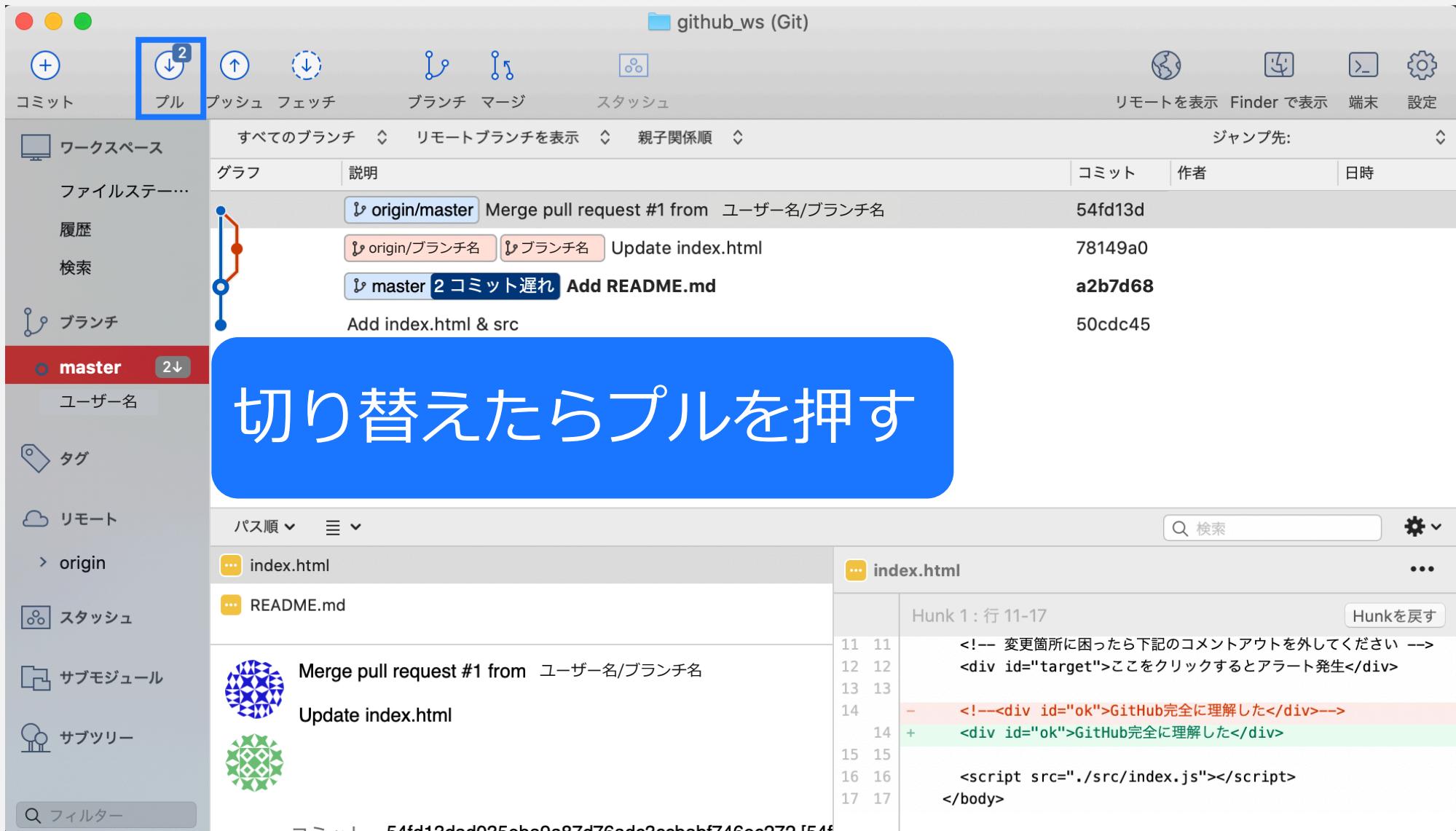
Sourcetreeで Pull する



ローカルに Pull する



ローカルに Pull する



ローカルに Pull する



マージされたものをローカルに反映

The screenshot shows the GitHub Desktop application interface. On the left, a sidebar lists repository branches like 'master', 'origin/master', and 'origin/branch-name'. The main area displays a timeline of commits. A specific commit from 'origin/master' is highlighted with a blue border, showing a merge pull request from a user named 'ユーザー名/プランチ名' that updated 'index.html' and added 'README.md'. A large blue callout bubble in the center of the screen contains the text: '作成されたブランチがマージされたのがわかる' (You can tell that the created branch was merged). Below the commit details, a diff view shows the changes made to 'index.html' and 'README.md'. The diff highlights additions in green and deletions in red, with a note at the top: '!-- 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 -->'. The bottom right corner of the diff view has a button labeled 'Hunkを戻す' (Revert Hunk).

github_ws (Git)

コミット ブル ブッシュ フェッチ ブランチ マージ スタッシュ リモートを表示 Finder で表示 端末 設定

ワークスペース ファイルステー... リモートブランチを表示 親子関係順

すべてのブランチ グラフ 説明

master origin/master Merge pull request #1 from ユーザー名/プランチ名
origin/プランチ名 ブランチ名 Update index.html

Add README.md
Add index.html & src

54fd13d
78149a0
a2b7d68
50cdc45

ユーザー名

タグ

リモート origin

パス順

index.html
README.md

検索

... Hunkを戻す

Merge pull request #1 from ユーザー名/プランチ名
Update index.html

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>
13 13
14 - <!--<div id="ok">GitHub完全に理解した</div>-->
14 + <div id="ok">GitHub完全に理解した</div>
15 15
16 16
17 17 <script src=".src/index.js"></script>
 </body>

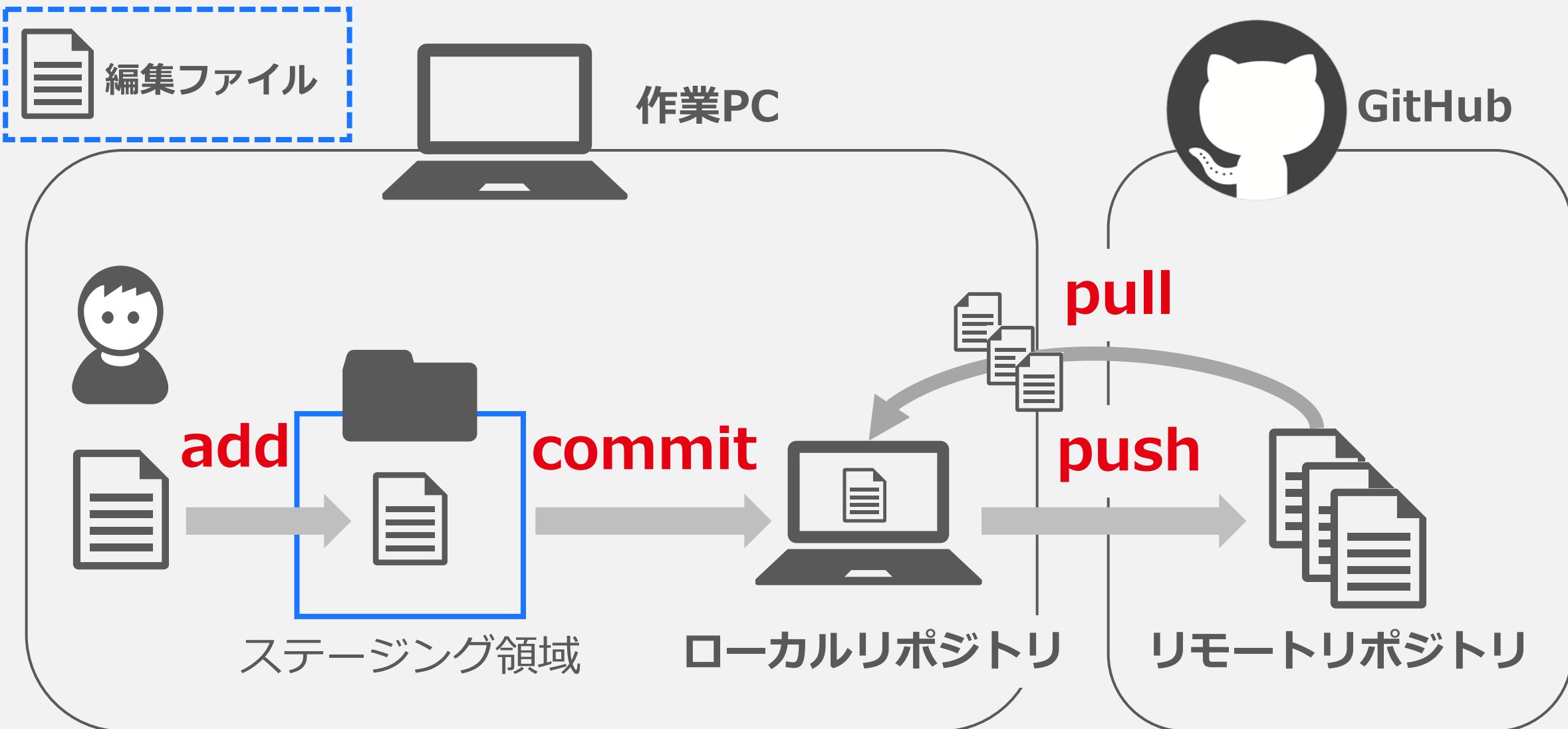
Copyright (C) 2020 Yahoo Japan Corporation. All Rights Reserved.

ローカルで変更を確認する

master の該当ファイルを開く

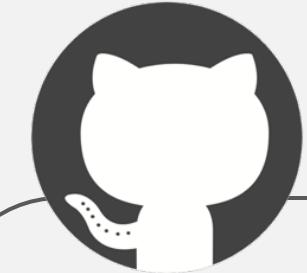
→ ファイルの変更箇所を確認

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

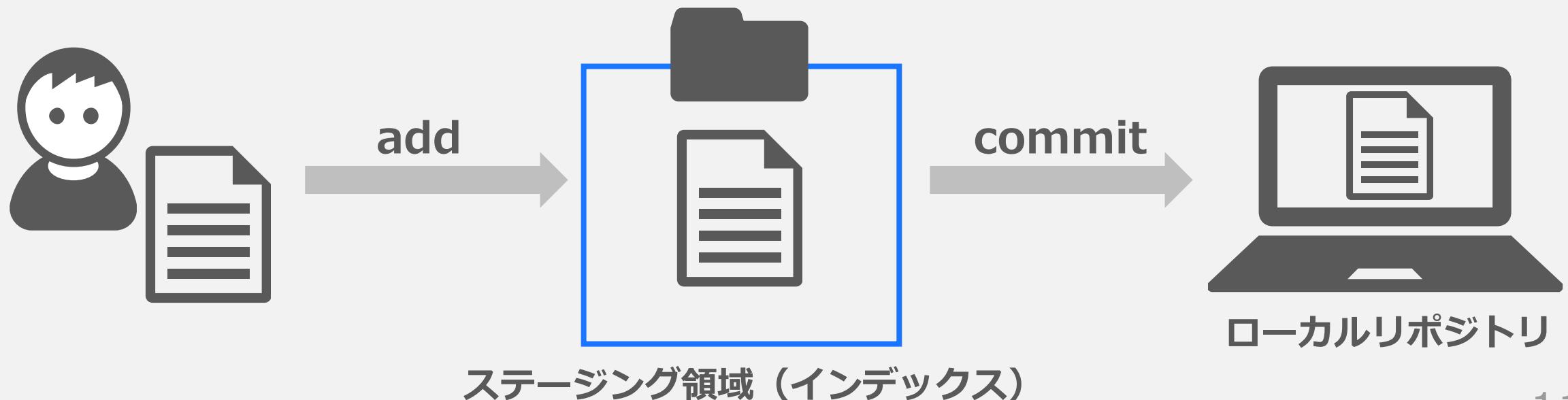
- Pull Request を作成して master 反映
 - 1. Pull Request を作成
 - こだわった点をコメントしよう
 - 他人のPRにコメントしよう
 - 2. Pull Request を merge
 - 3. ローカルの master に Pull
 - 変更が反映されてるかファイル内容を確認しよう

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**
- ・ commit したあとは**必ずコメントを残す**

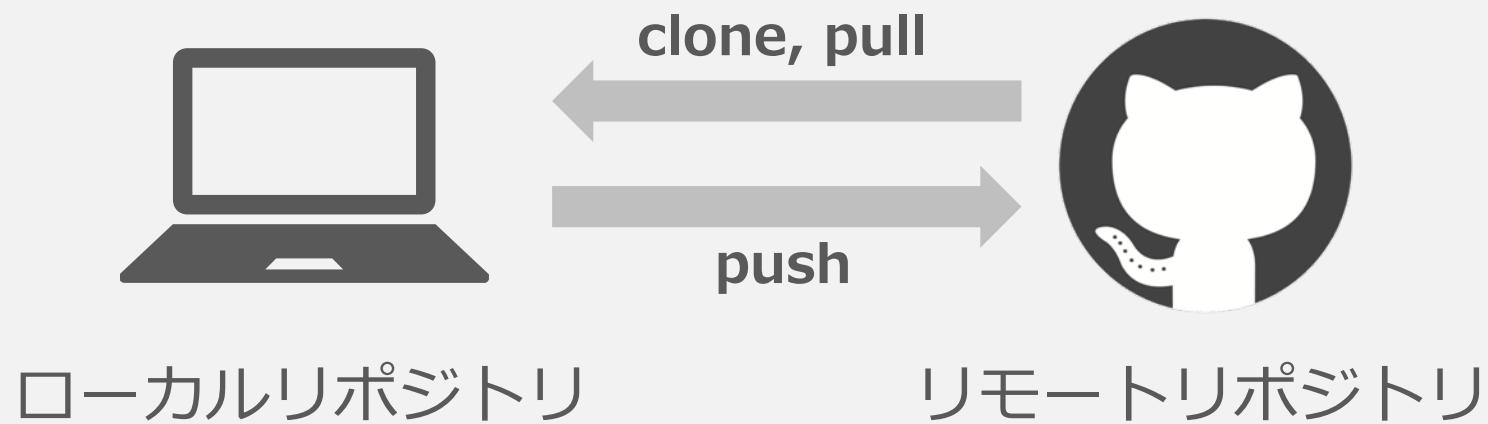


【おさらい】ローカルとリモートの関係

clone : リモートをローカルにコピー

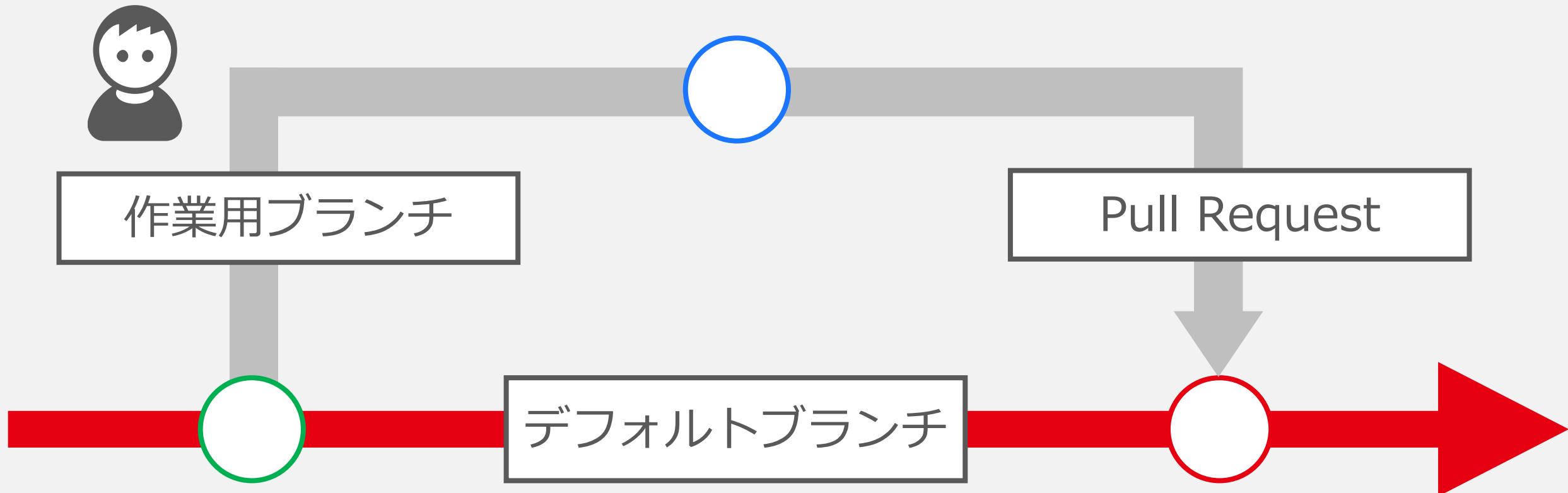
pull : リモートの変更をローカルに反映

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。

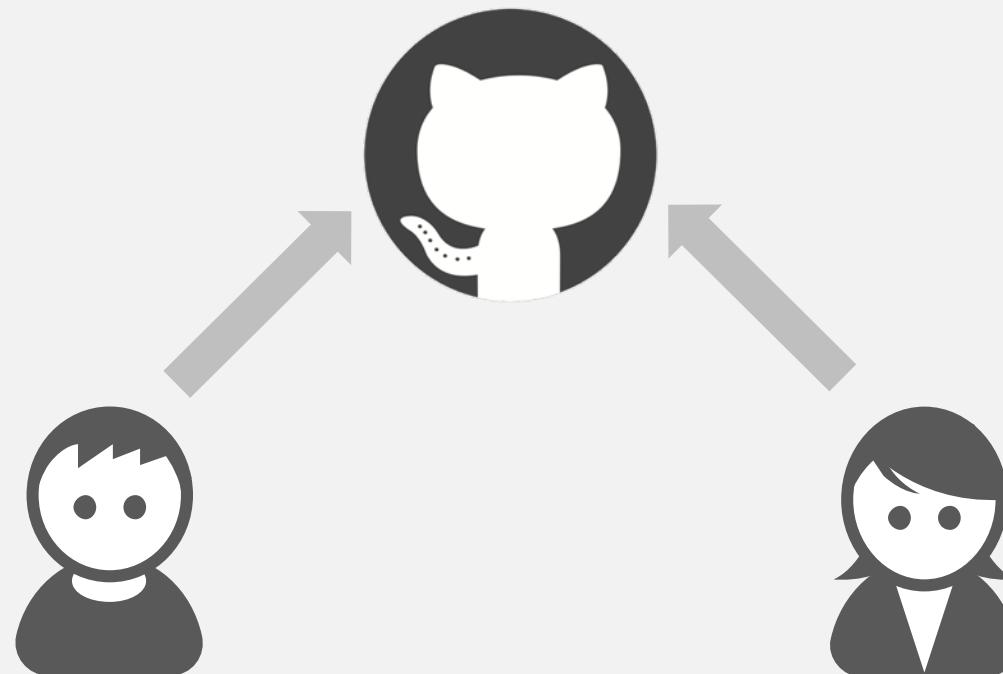
まずは基本的な使い方に慣れる。
使っていく中で調べながら使う。

GitHub を複数人で使う

YAHOO!
JAPAN

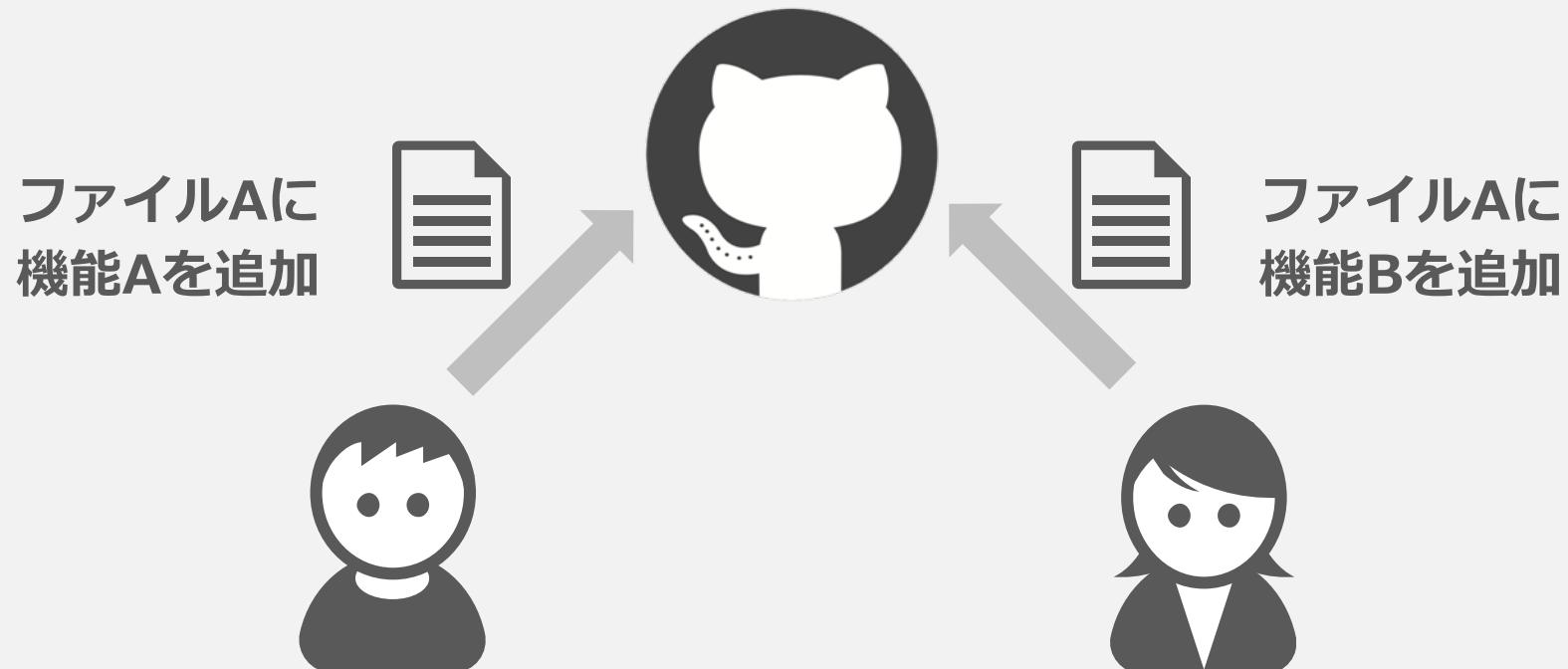
チーム開発では

- 1つのリポジトリに対して複数人で作業する



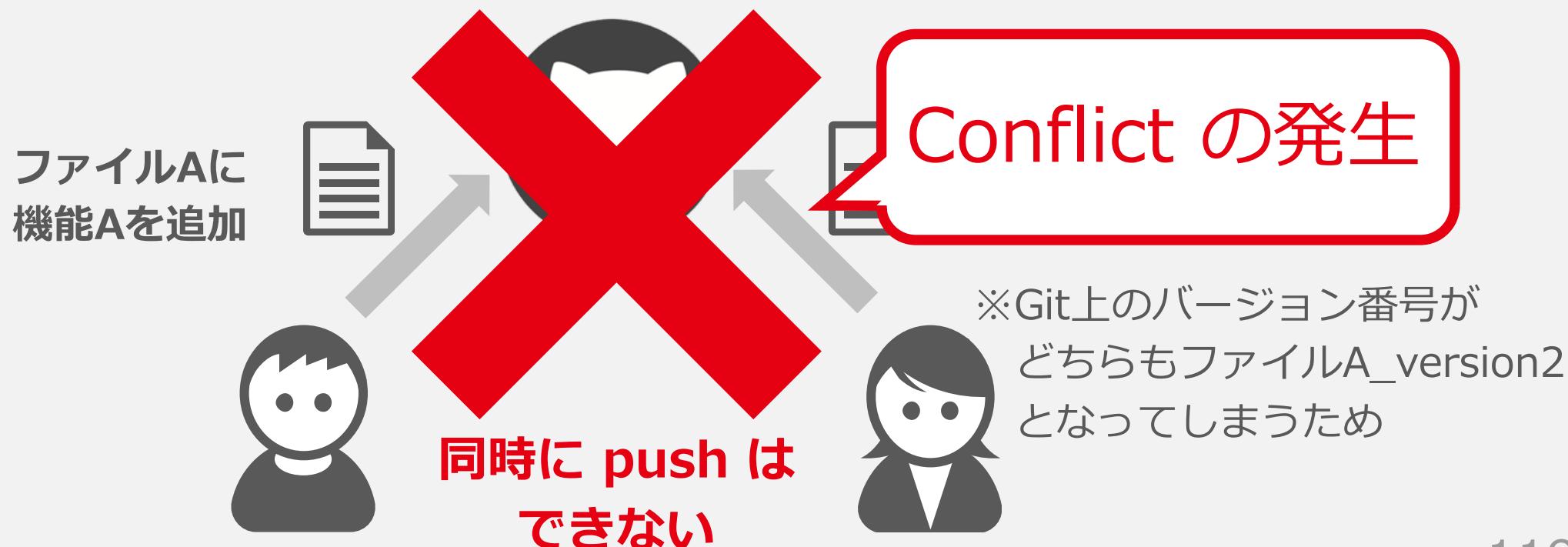
チーム開発では

- ・ 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



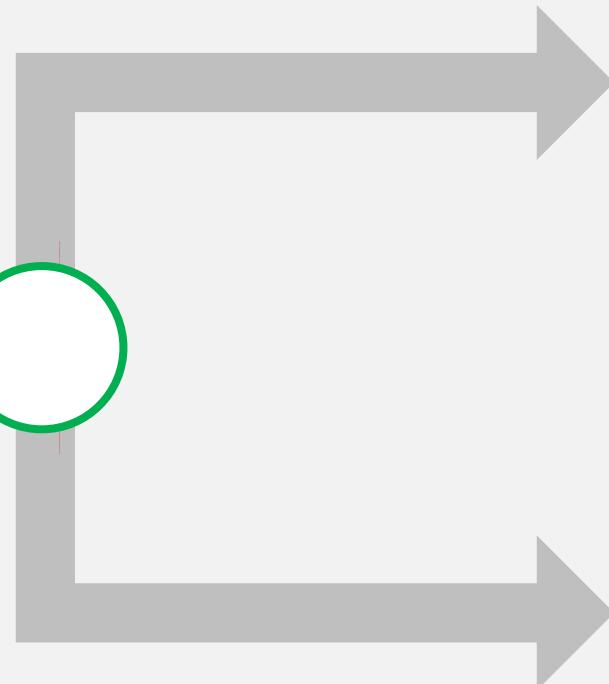
同時に push は
できない



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

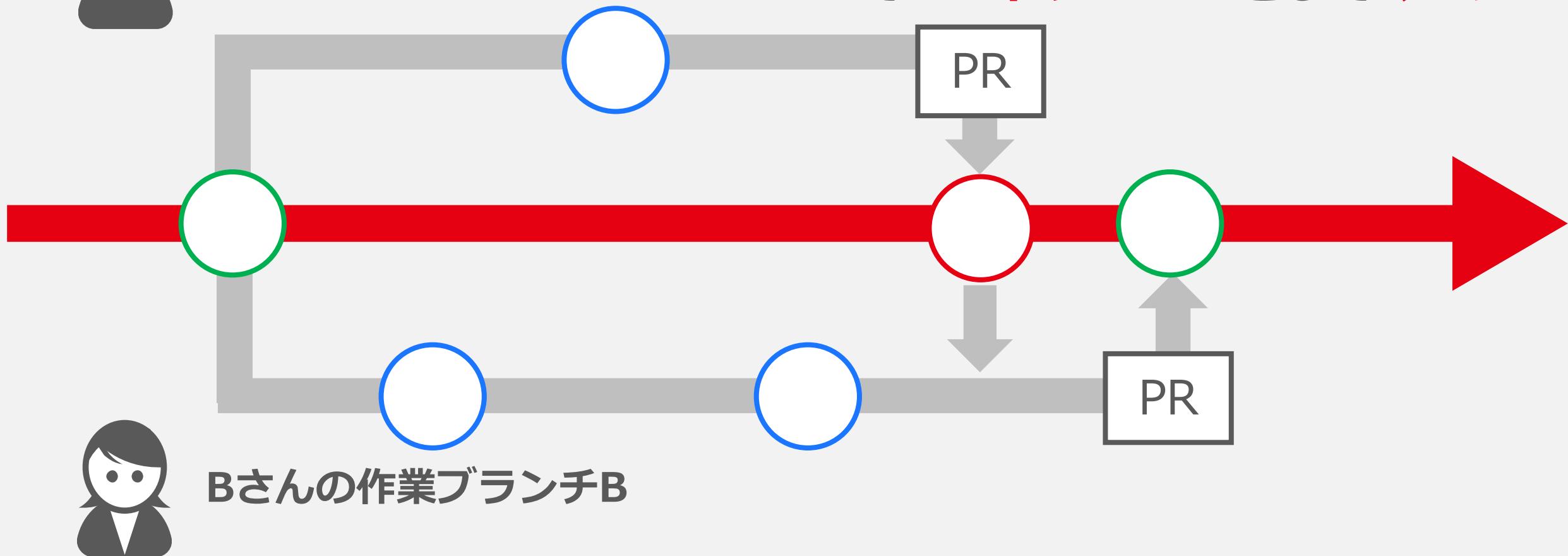
作業者がそれぞれブランチを作成

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



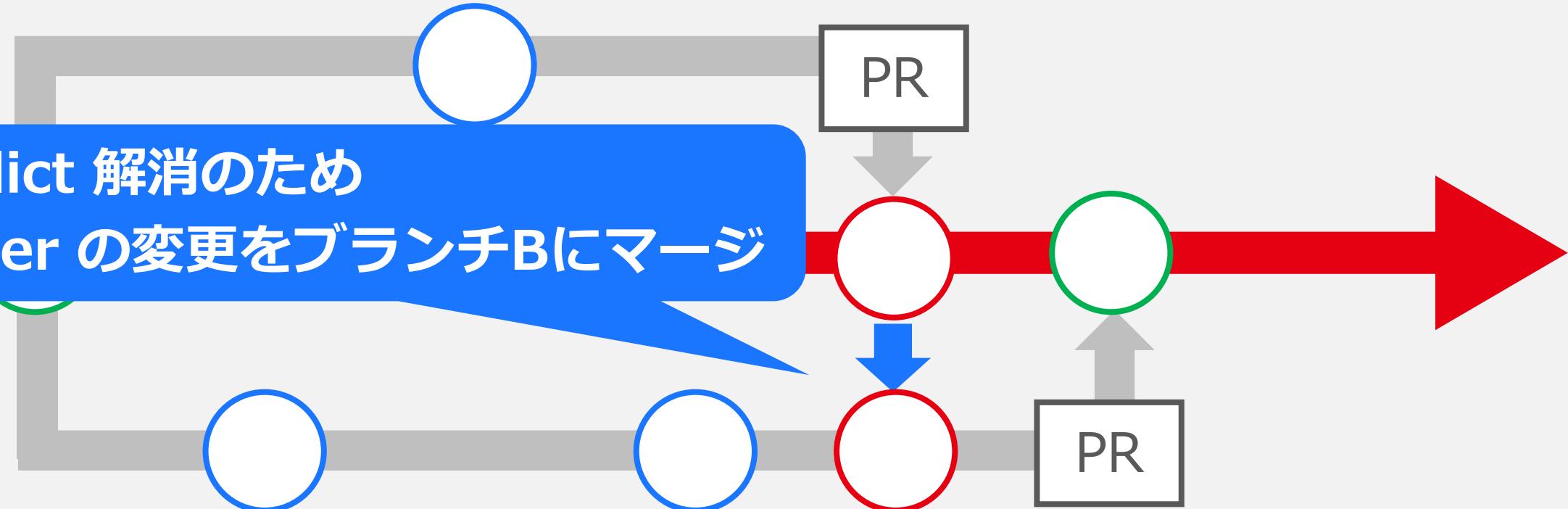
Aさんの作業ブランチA

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
master 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 **compass/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 master
```

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