

Git及びGitHub勉強会

濱口皓樹

soiwebでは

DropBoxで管理

- 容量に制限
- 現状ごちゃごちゃしていて作り直したい

現状

GitHubで管理

- 公開されているので就職時にアピール
- バージョン管理(変更履歴)
- README.mdで詳細が書ける
- 誰が仕事したかわかる

これから

Agenda

1. Gitについて

2. GitHubについて

3. 使ってみる

その前に...

Q. 古いコードの復元するには？

以前は

「フォルダのコピーを取っておく」

「Ctrl+Zを押しまくる」

「覚えておく（脳筋）」

Gitを使うと簡単

名前
120525_ドキュメント_最新.txt
120602_ドキュメント.txt
120604_ドキュメント.txt
120605_ドキュメント_修正版.txt
120605_ドキュメント_江口.txt
120605_ドキュメント_最新のコピー.txt
120605_ドキュメント_最新.txt
120605_ドキュメント.txt
ドキュメント_会議用.txt

Gitでできること

- ファイルの変更履歴が管理できる
- いつでも過去のファイルに戻せる
- なんでも入る
- バックアップにもなる
- チームでファイルを共有できる



Gitとは

分散型バージョン管理システム

「誰が」「いつ」「どのような変更をしたか」
を管理

複数人で開発する上で必須

時代は

集中型(Subversion[SVN]) から **分散型**(Git) へ

Git ? GitHub ?

GitHubとは？

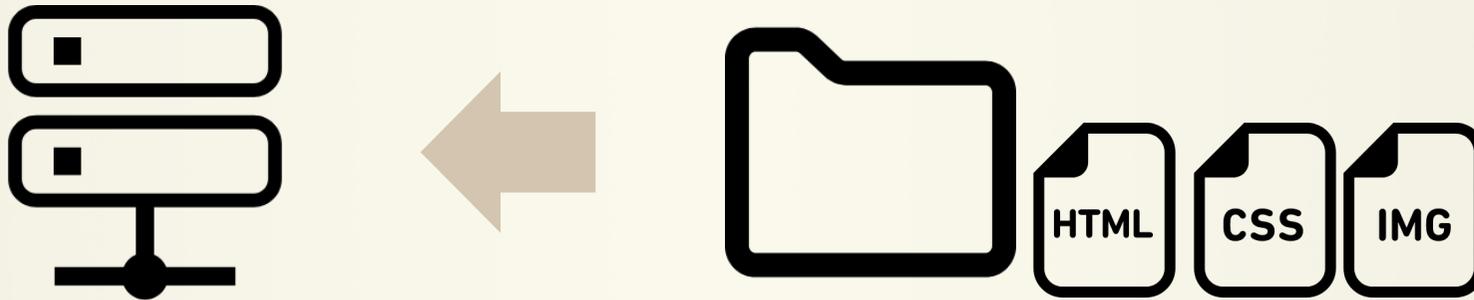
- 「Git」の「ハブ：拠点・中心・集まり」
- Gitの仕組みを利用して、世界中の人々が自分の作品を保存、公開できるウェブサービス

Gitを便利に使うためのツール

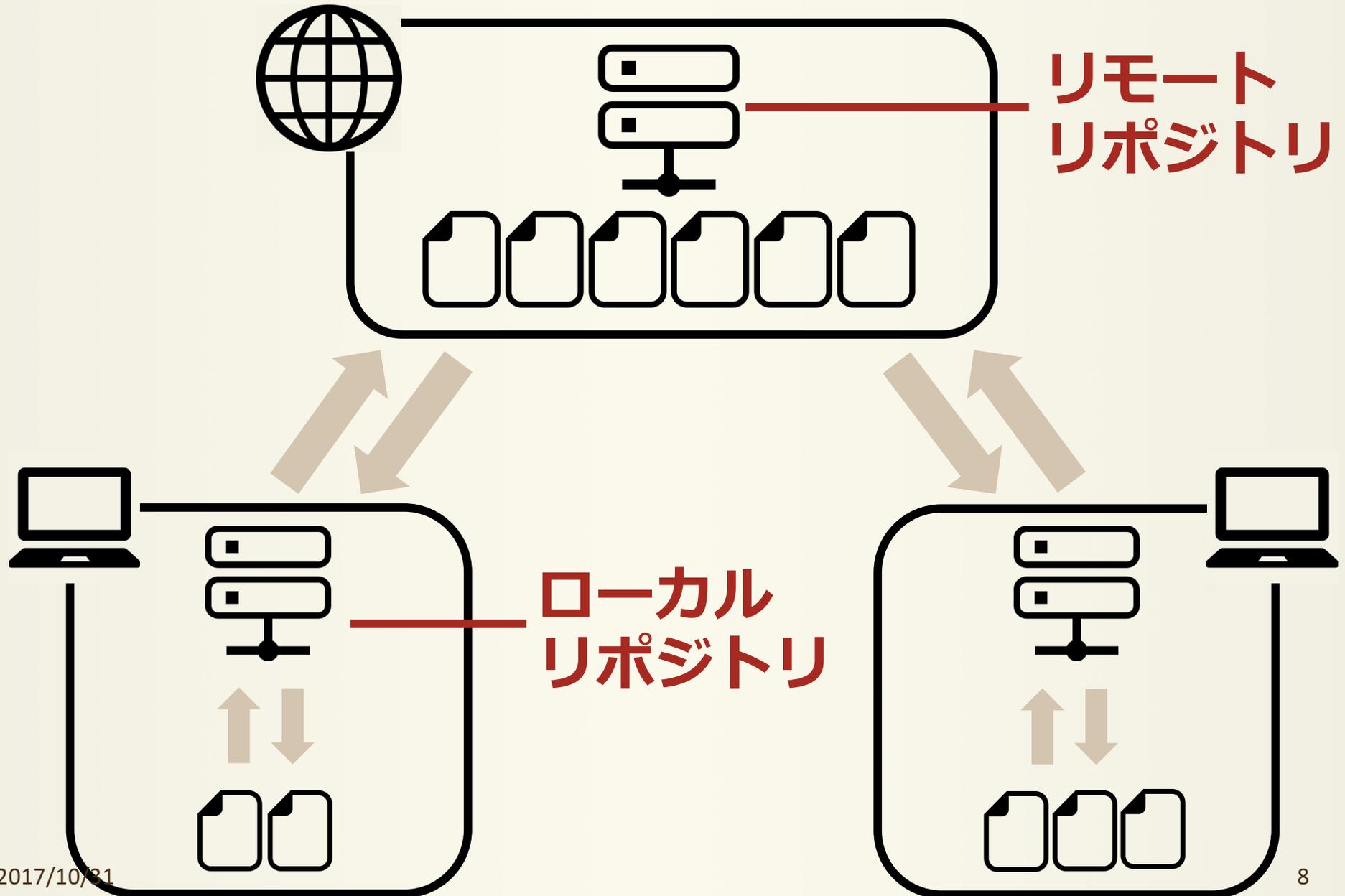
Gitとは

リポジトリ

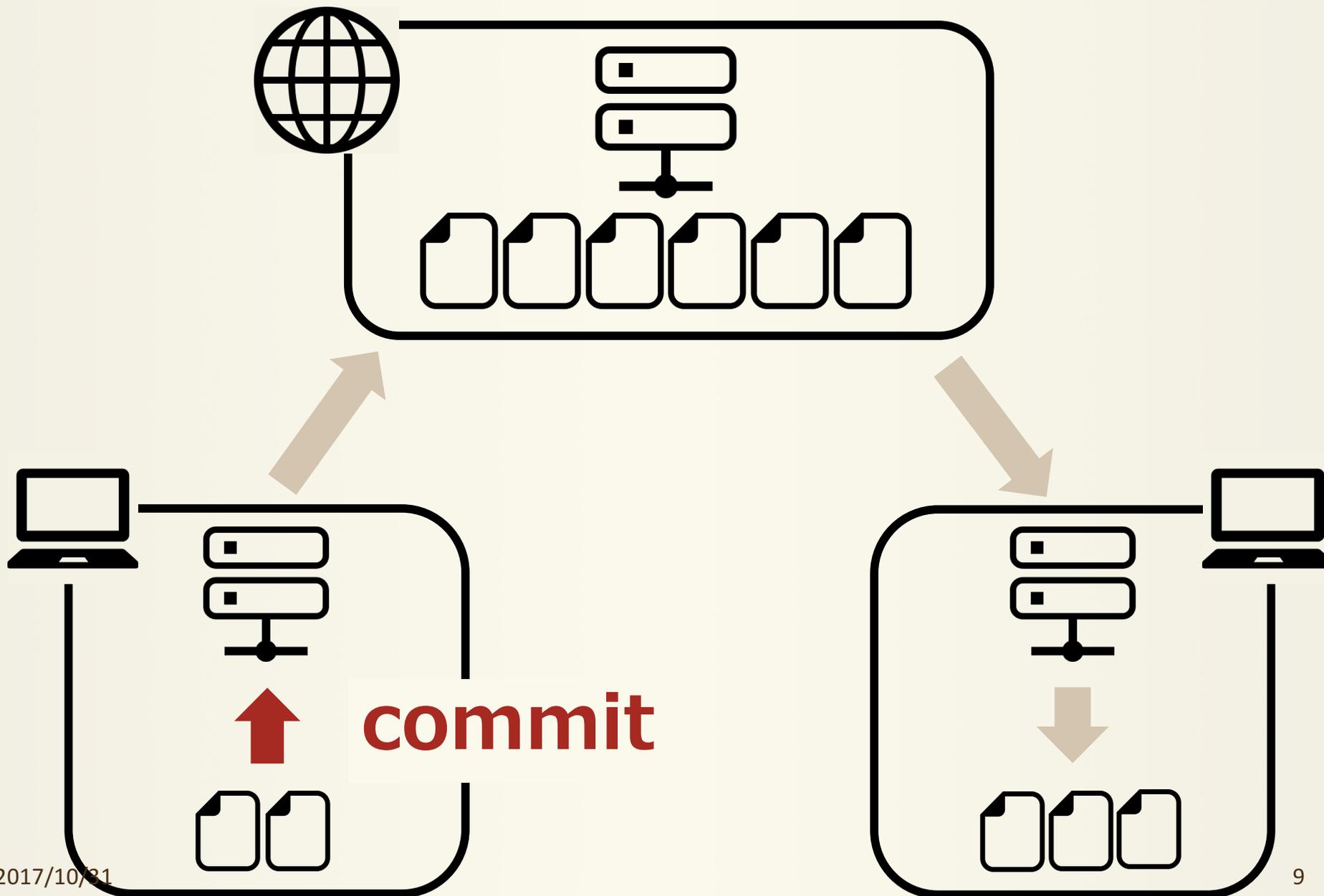
ファイルやディレクトリの状態を保存する場所



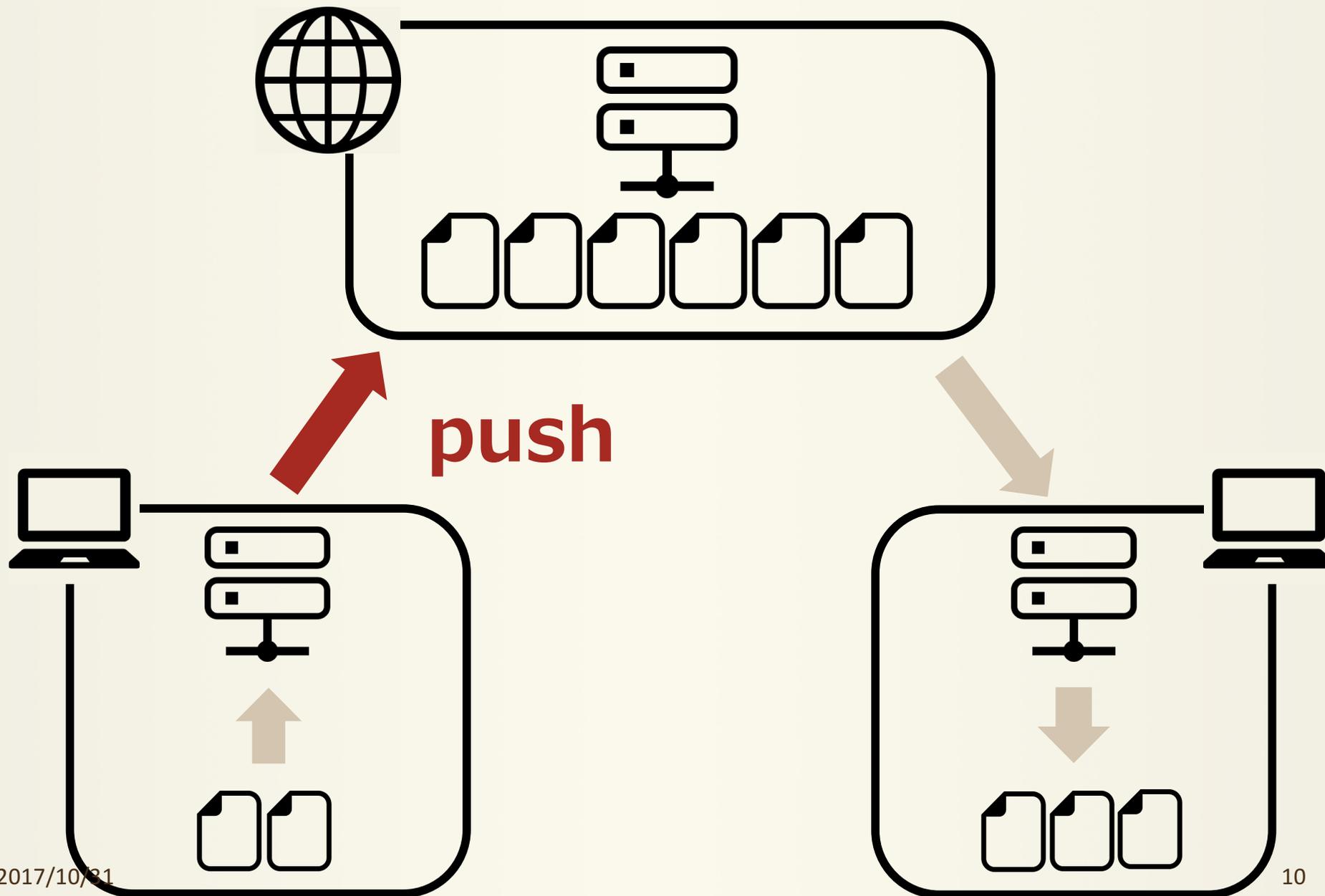
Gitの全体構成



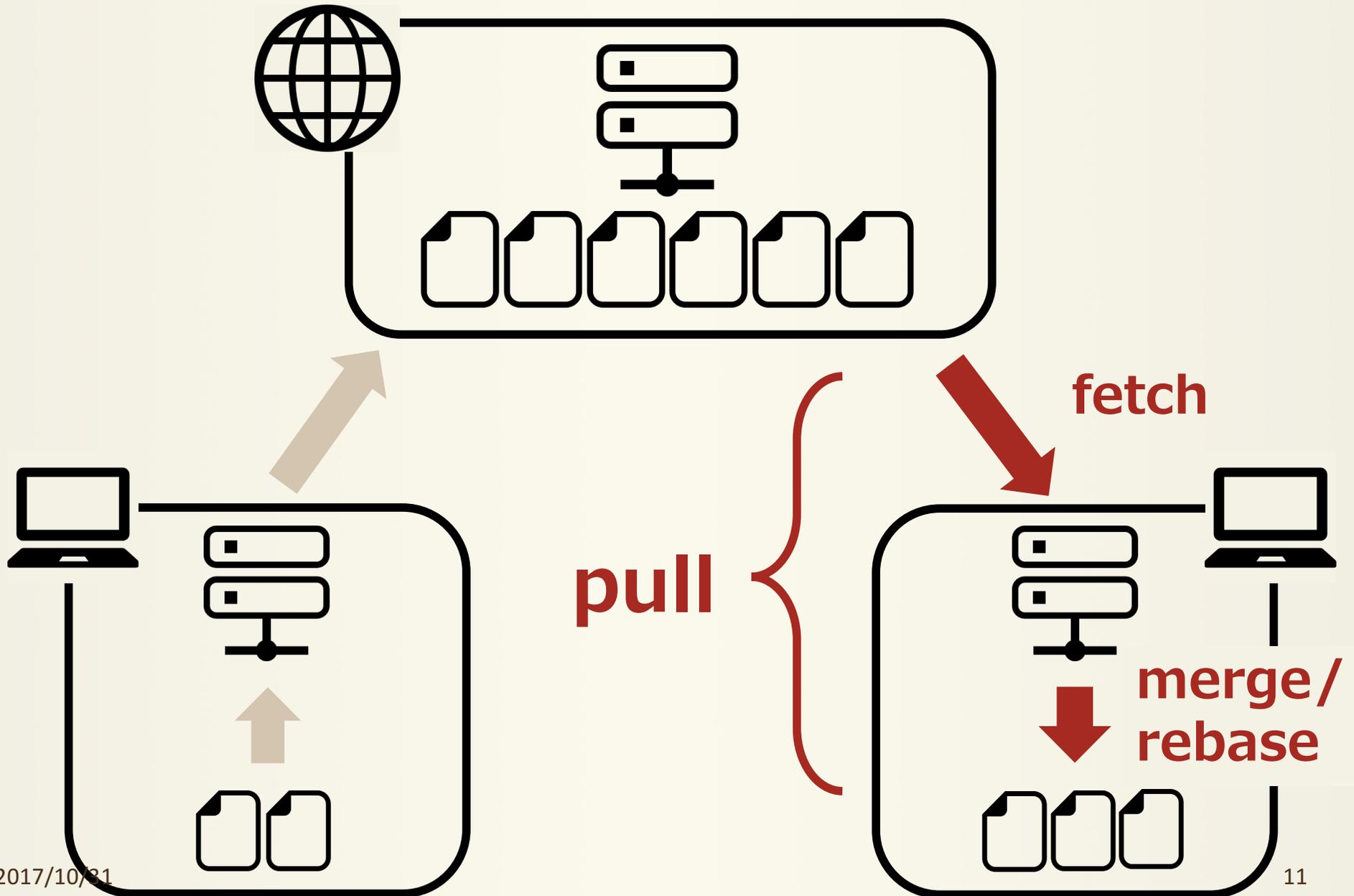
Gitの全体構成



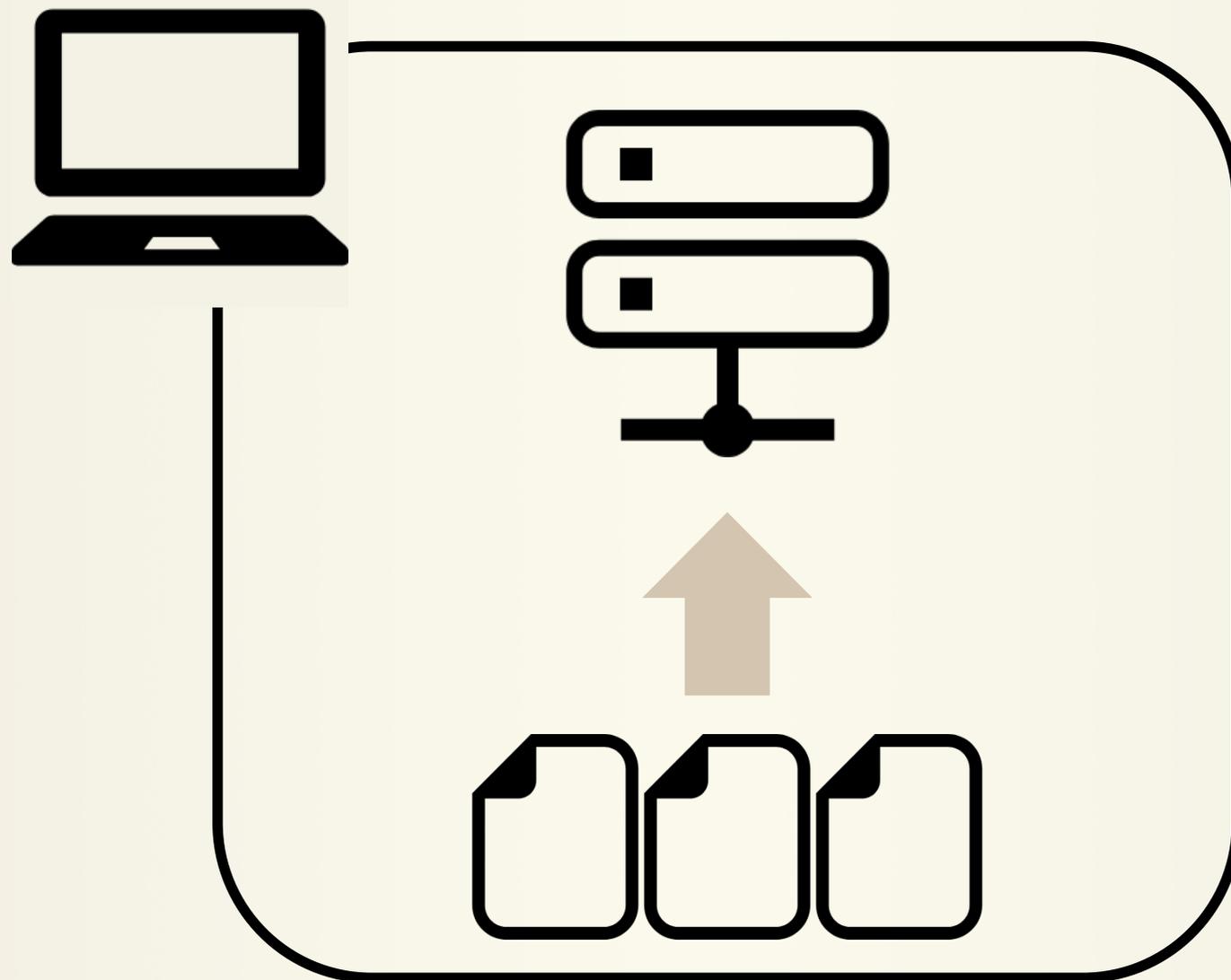
Gitの全体構成



Gitの全体構成



ローカル環境の内部構成



ローカル環境の内部構成



例



git_root



hello.txt



parent



child.txt



.git

作業ディレクトリ
(ワークツリー)

実際に作業中の
ディレクトリ

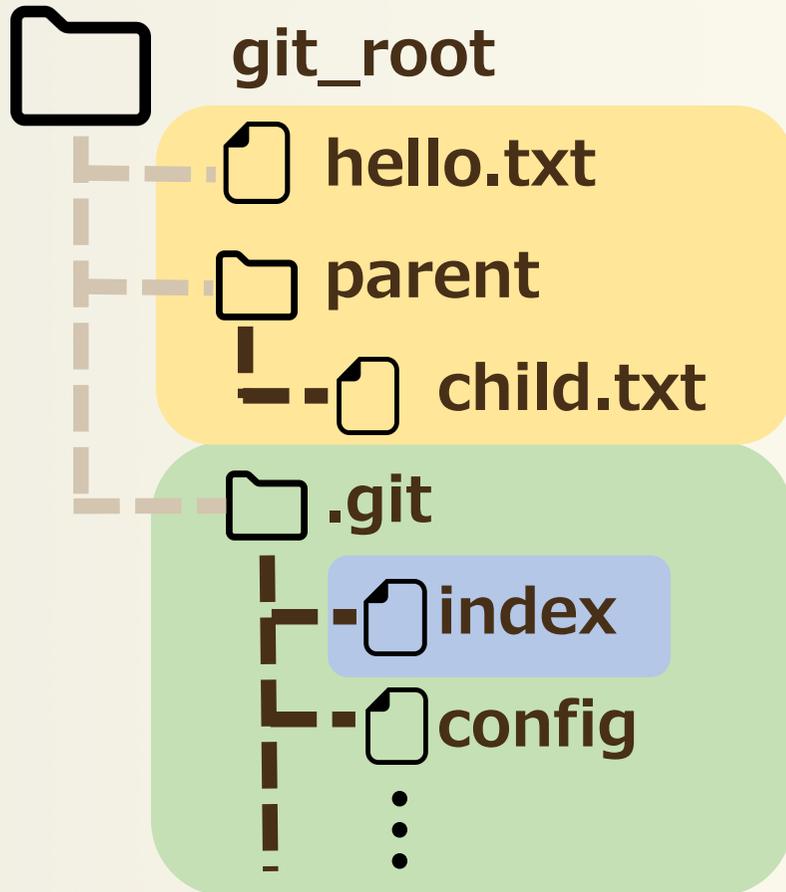
Gitディレクトリ

Gitが管理している
ディレクトリ

ローカル環境の内部構成



例



作業ディレクトリ
(ワークツリー)

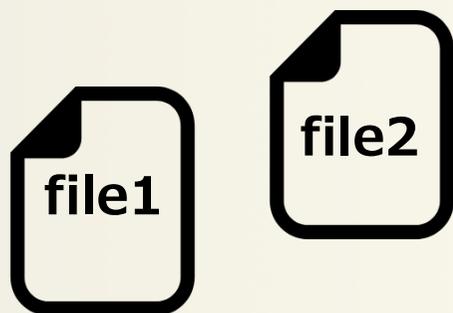
ステー징・エリア
(インデックス)

Gitディレクトリ

基本の流れ

Gitでの管理開始前

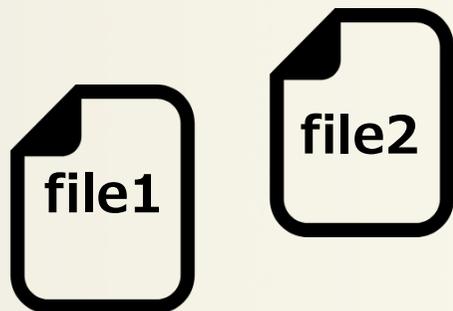
作業
ディレクトリ



基本の流れ

\$ git init

作業
ディレクトリ



ステージング
エリア

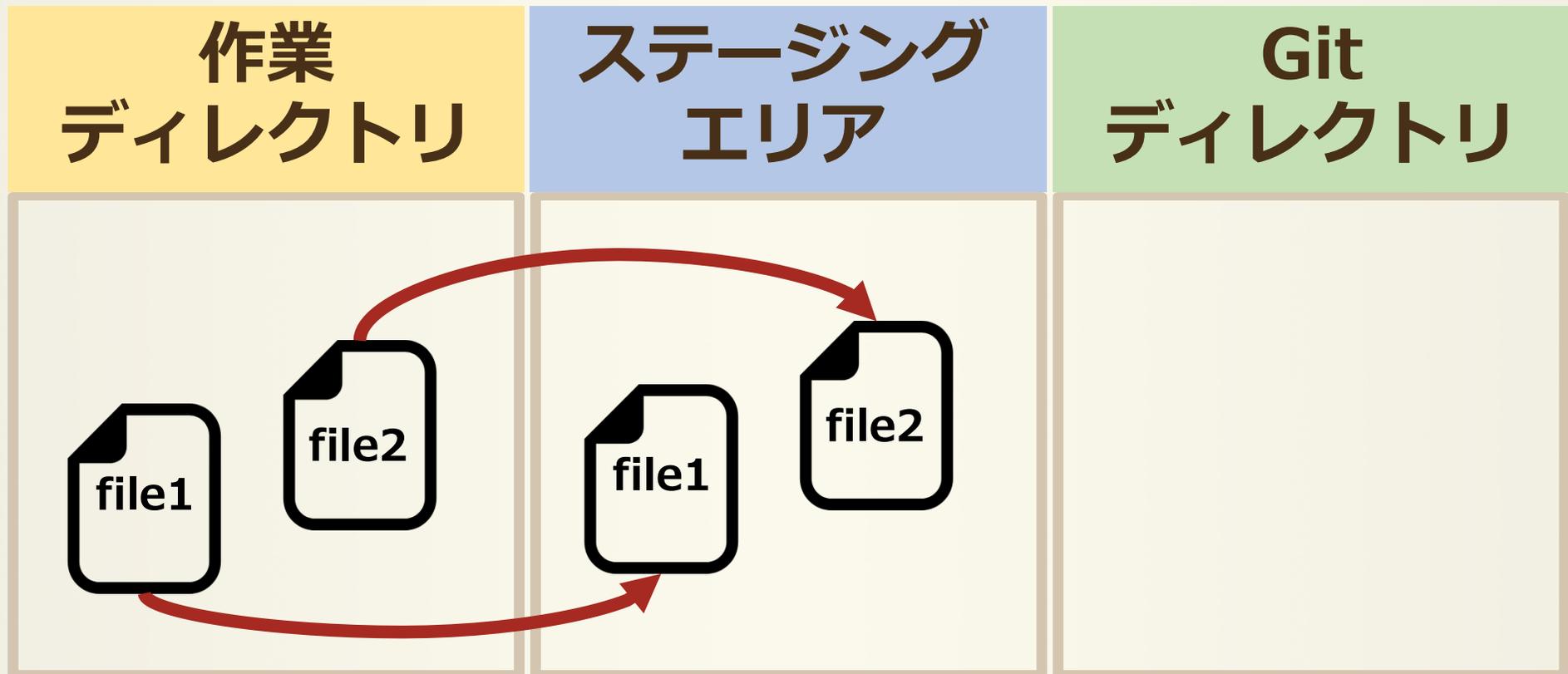


Git
ディレクトリ



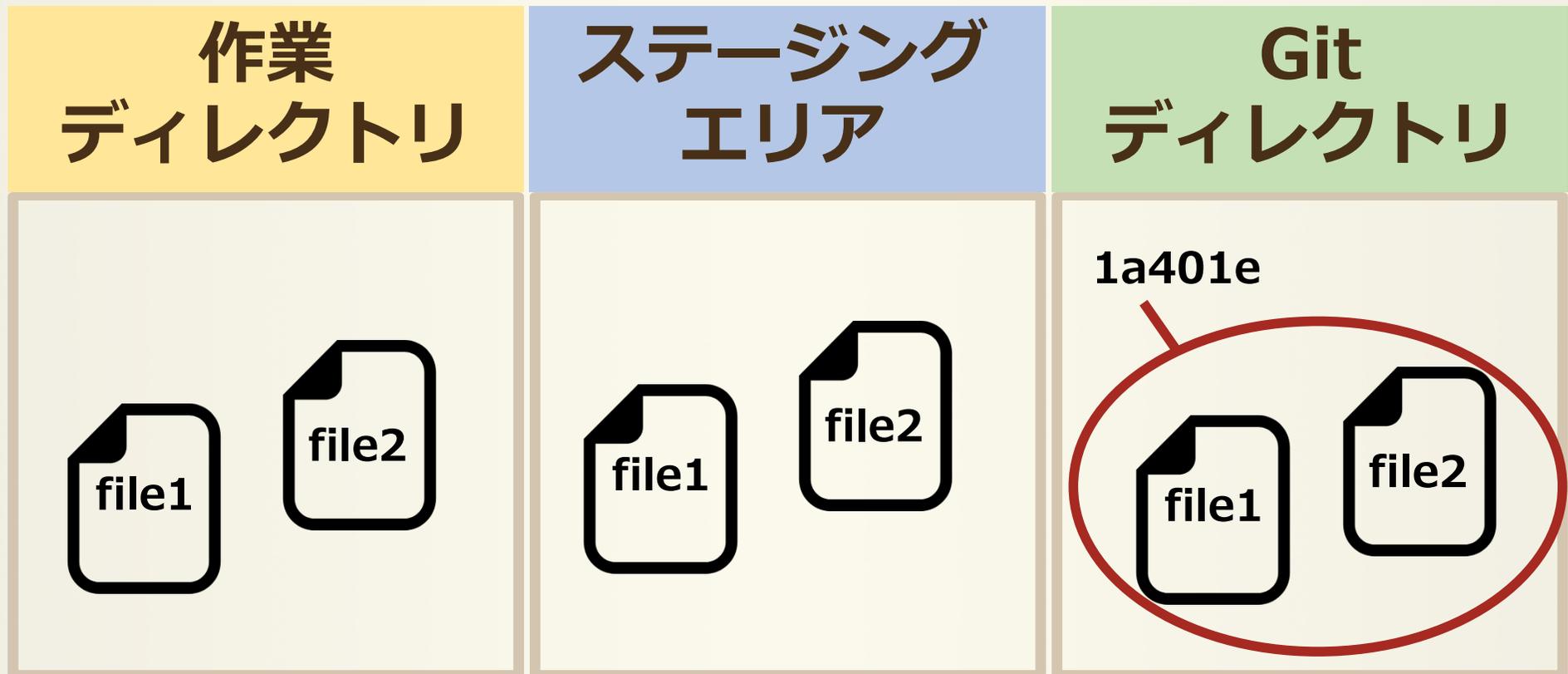
基本の流れ

\$ git add file1 file2 **ファイル名**



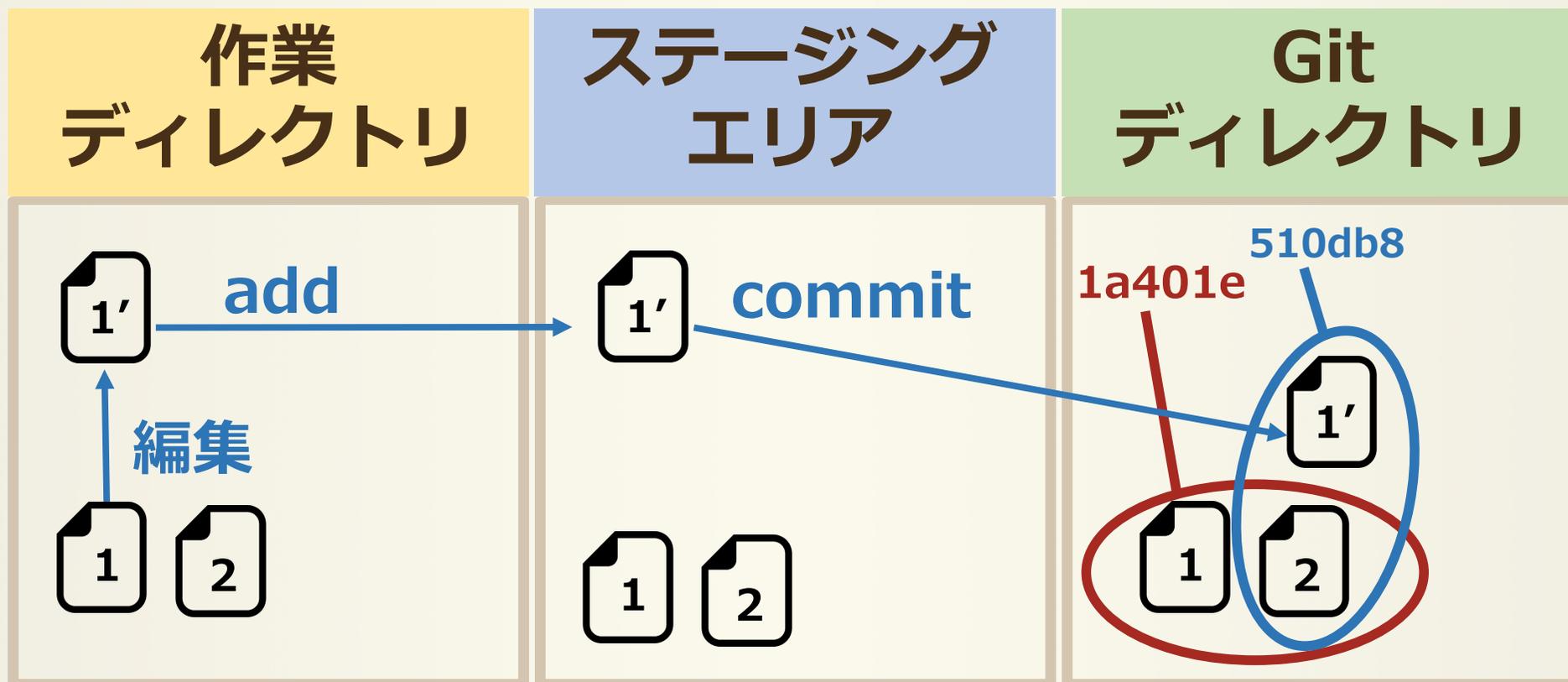
基本の流れ

```
$ git commit -m 'アレ更新した'
```



基本の流れ

編集しても...



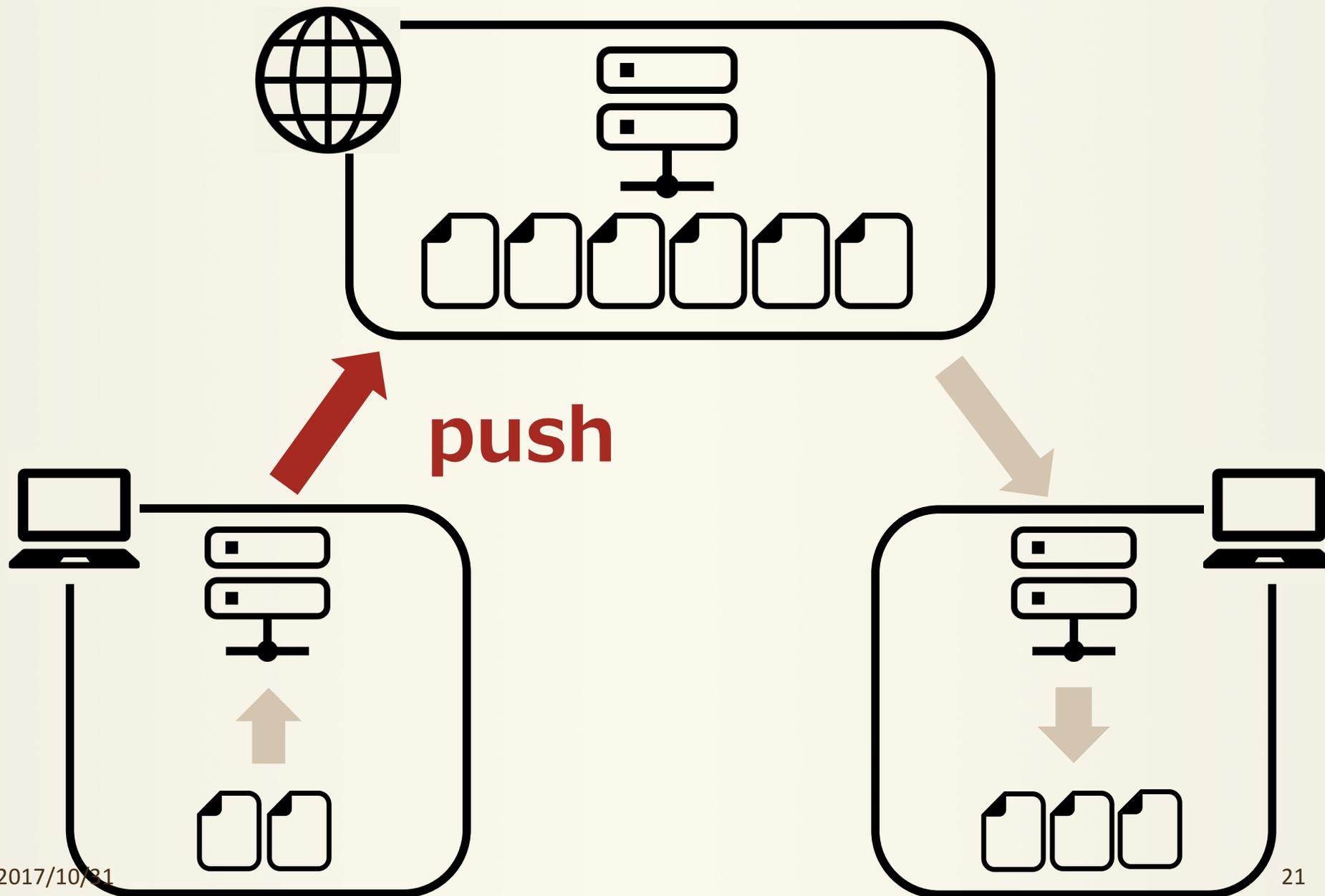
基本の流れ(未)

削除の場合...

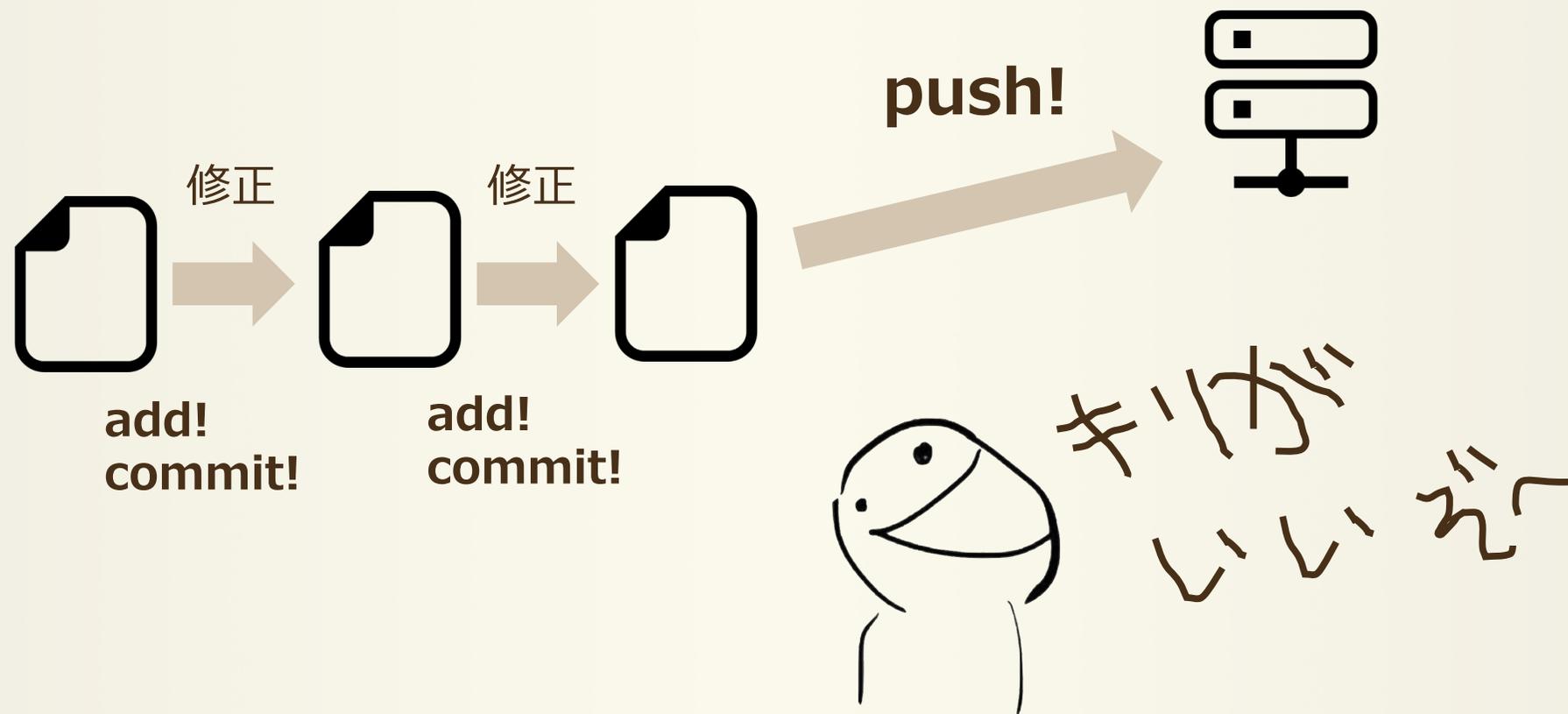
消したファイルも
コミット必要



基本の流れ



基本の流れ



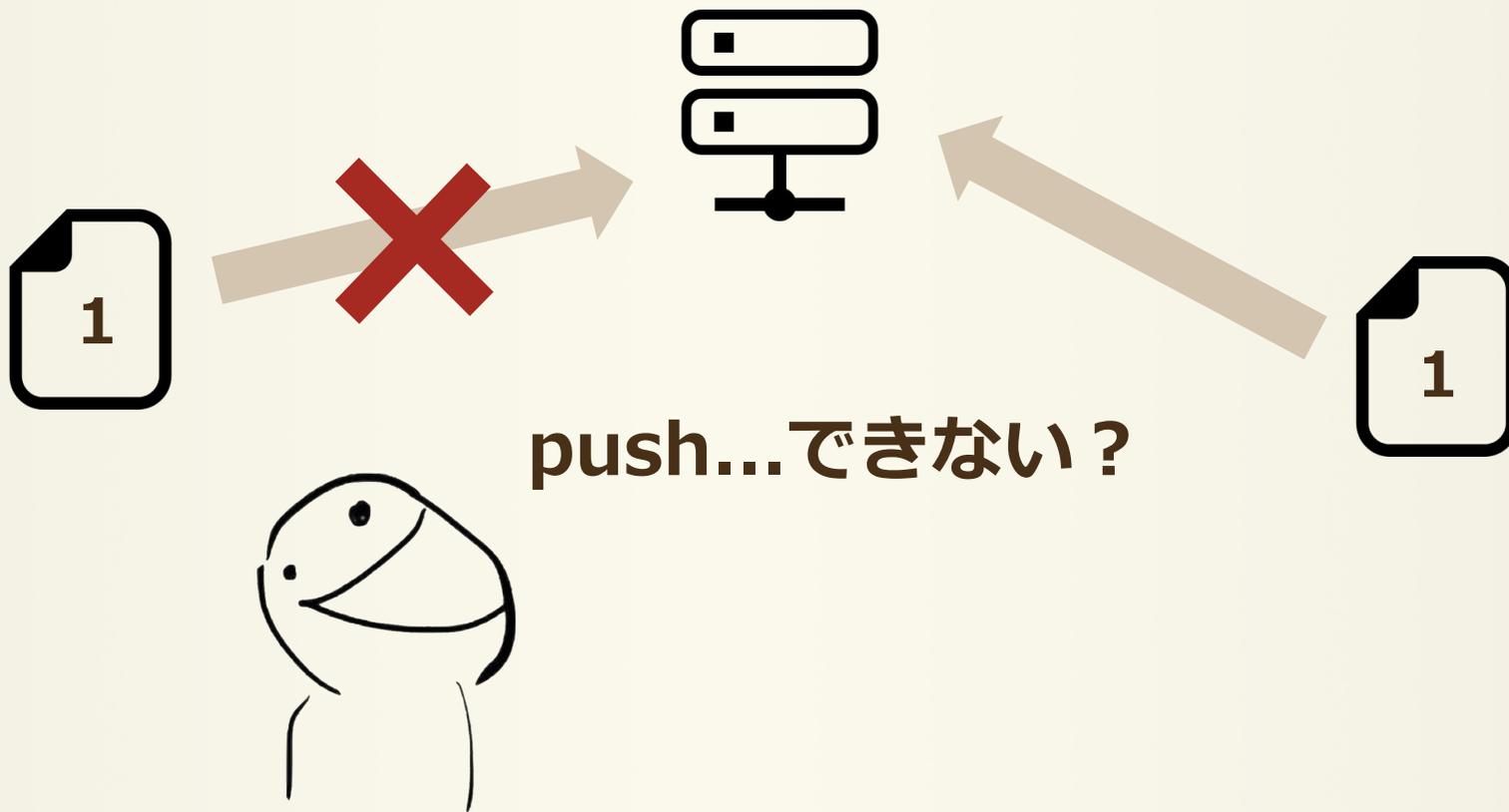
衝突

同じ箇所を別々のブランチで変更



どっちの変更を適用すればいいかわからない！

基本の流れ



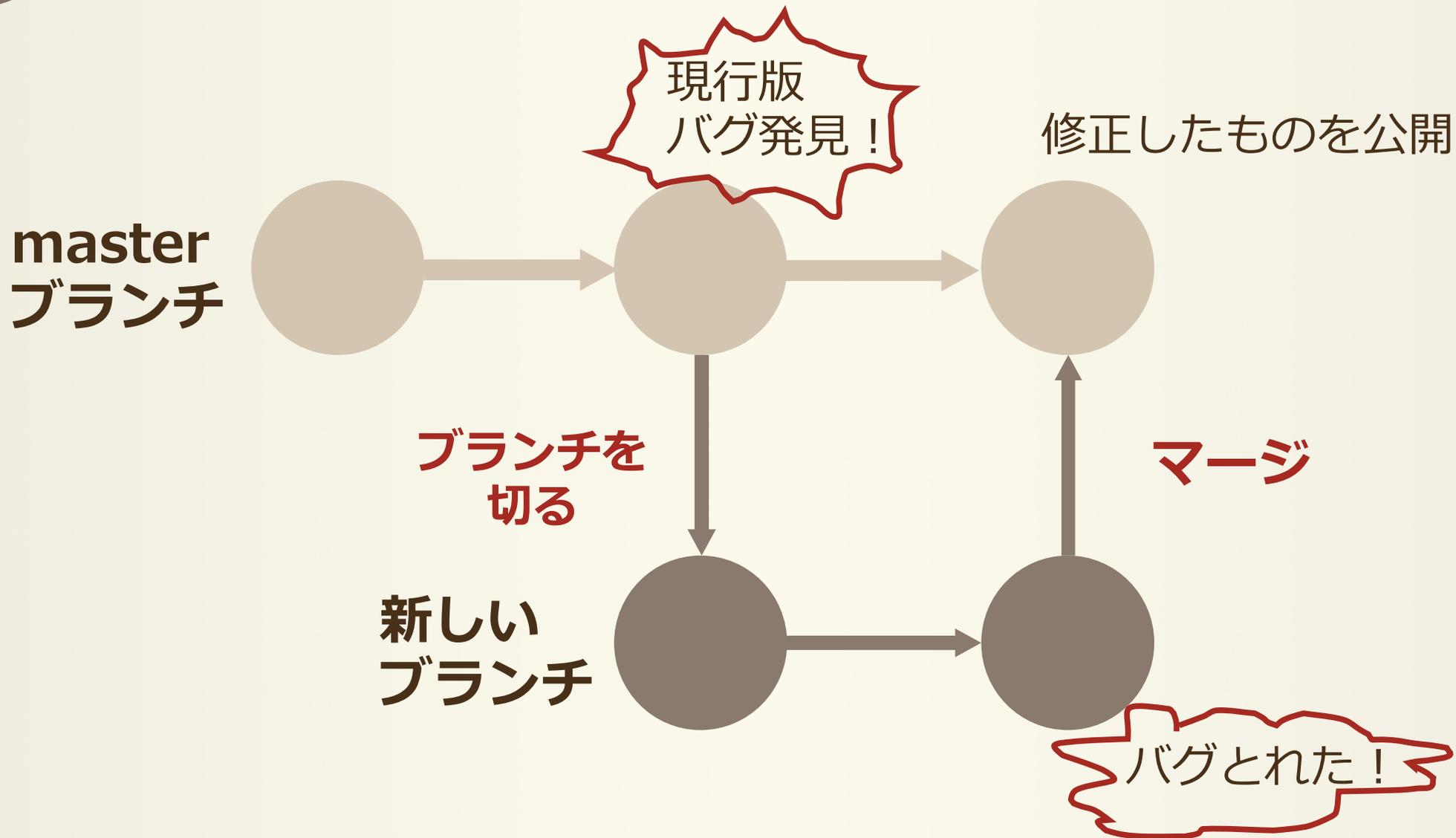
ランチ

どんな時に使う？

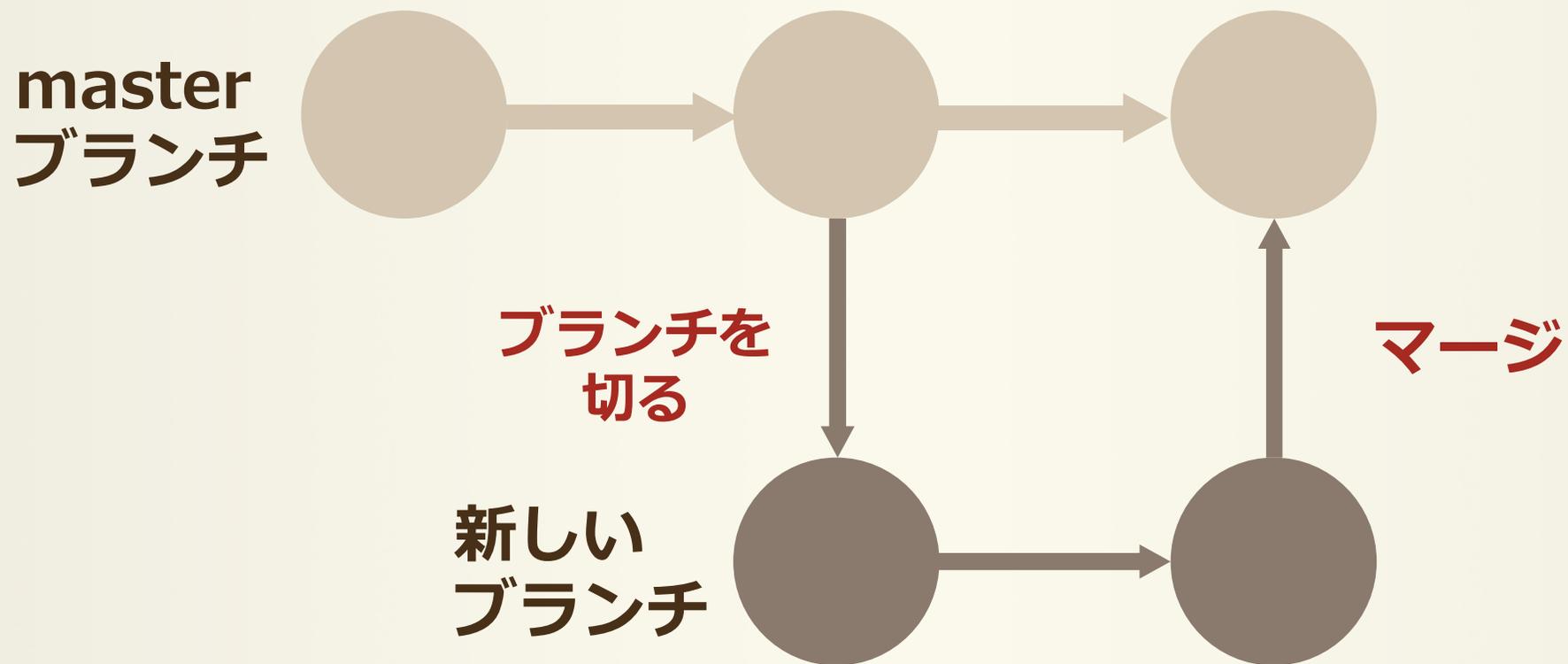
- 別の機能の開発
- バグ修正

いじって変になったら戻すのめんどくさーい

ブランチを用いた開発の例



ブランチ



ブランチ

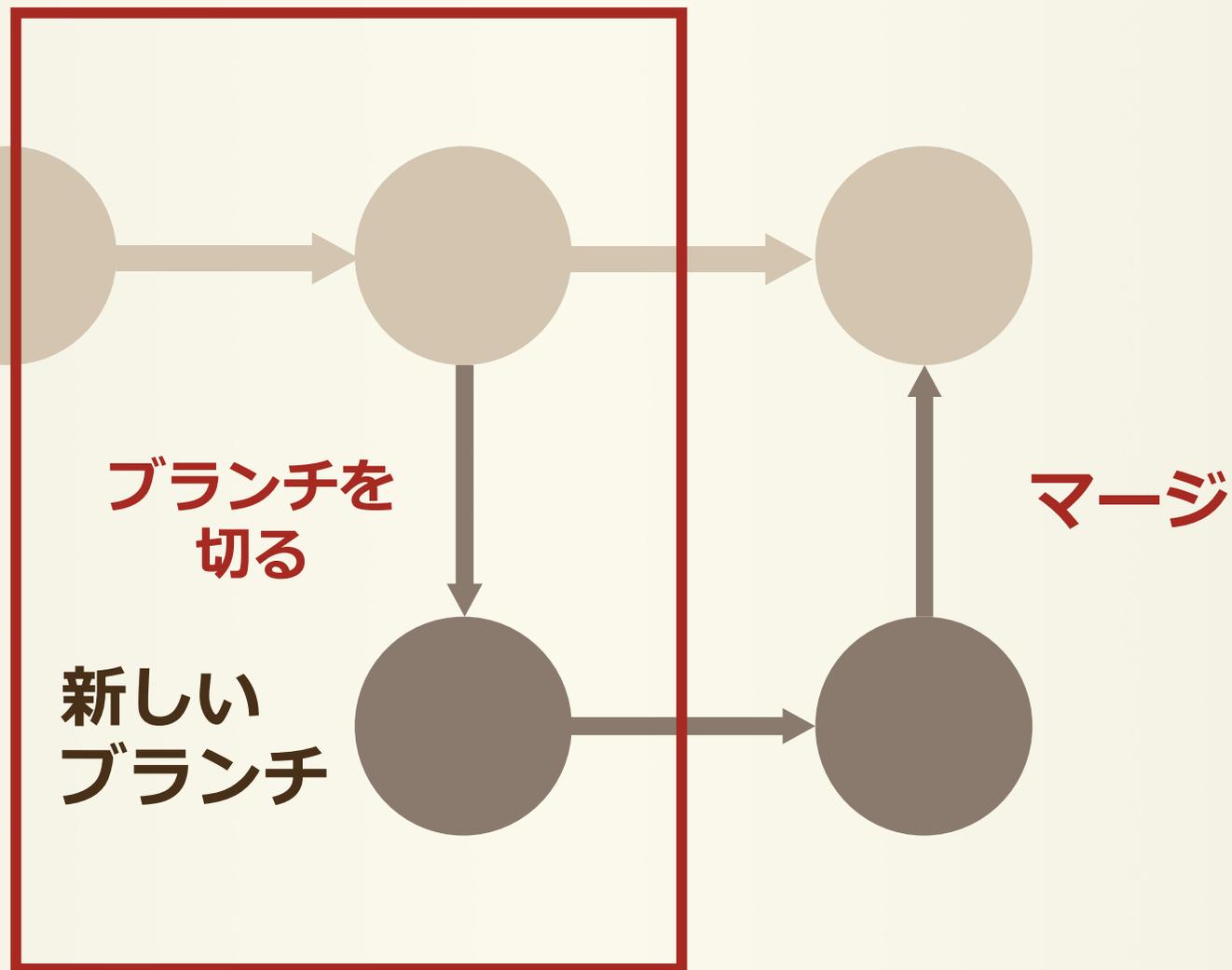
- **git branch**

現在のブランチを確認

```
kut-db:main huncro$ git branch  
  bpushtest  
  gucci  
* master  
  tekito
```

ブランチ

master
ブランチ



ブランチ

- `git branch` ブランチ名
ブランチを作成(切る)

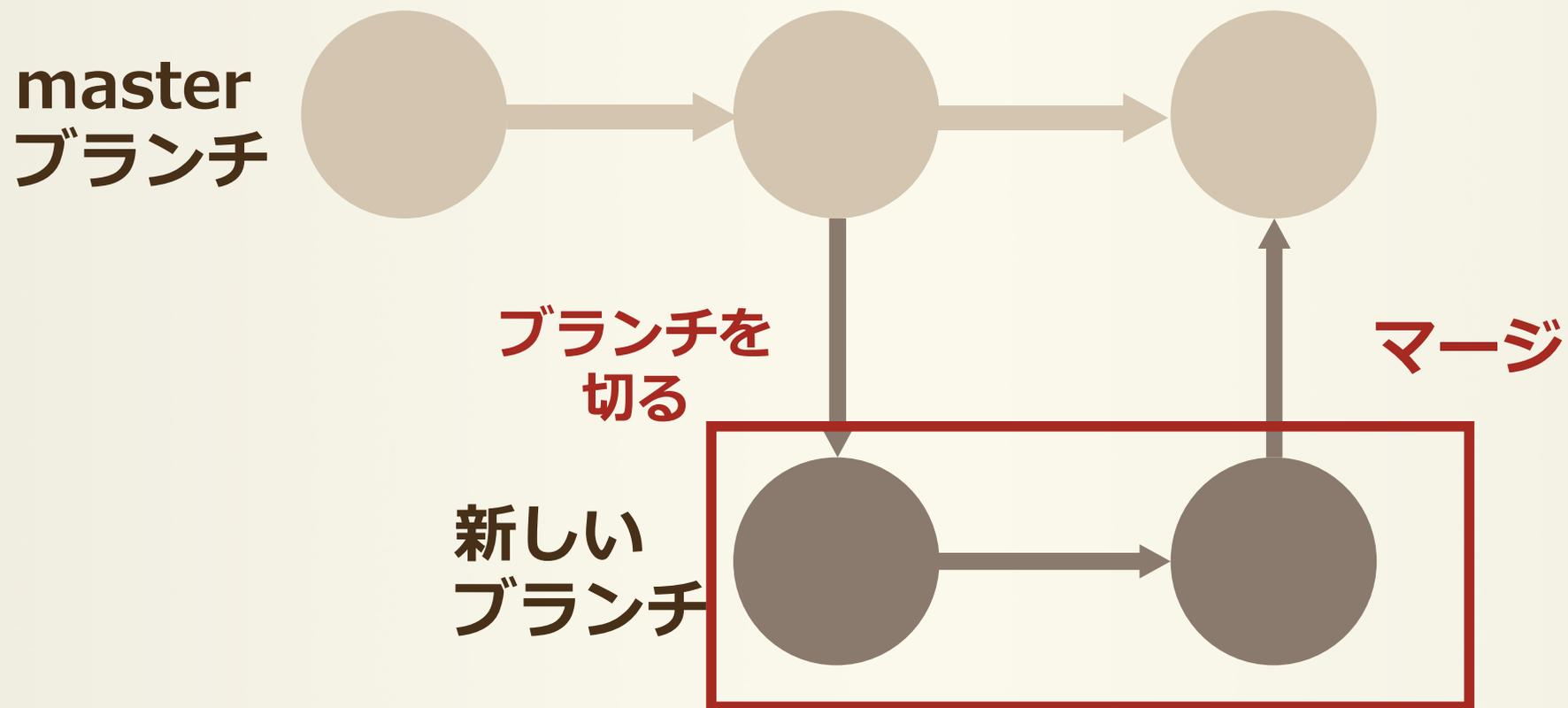
```
kut-db:main huncro$ git branch gucci_debug
kut-db:main huncro$ git branch
  bpushtest
  gucci
  gucci_debug
* master
  tekito
```

ブランチ

- **git checkout ブランチ名**
[ブランチ名]にブランチを移動

```
kut-db:main huncro$ git branch gucci_debug
kut-db:main huncro$ git checkout gucci_debug
M      .DS_Store
Switched to branch 'gucci_debug'
kut-db:main huncro$ git branch
  bpushtest
  gucci
* gucci_debug
  master
  tekito
```

ブランチ



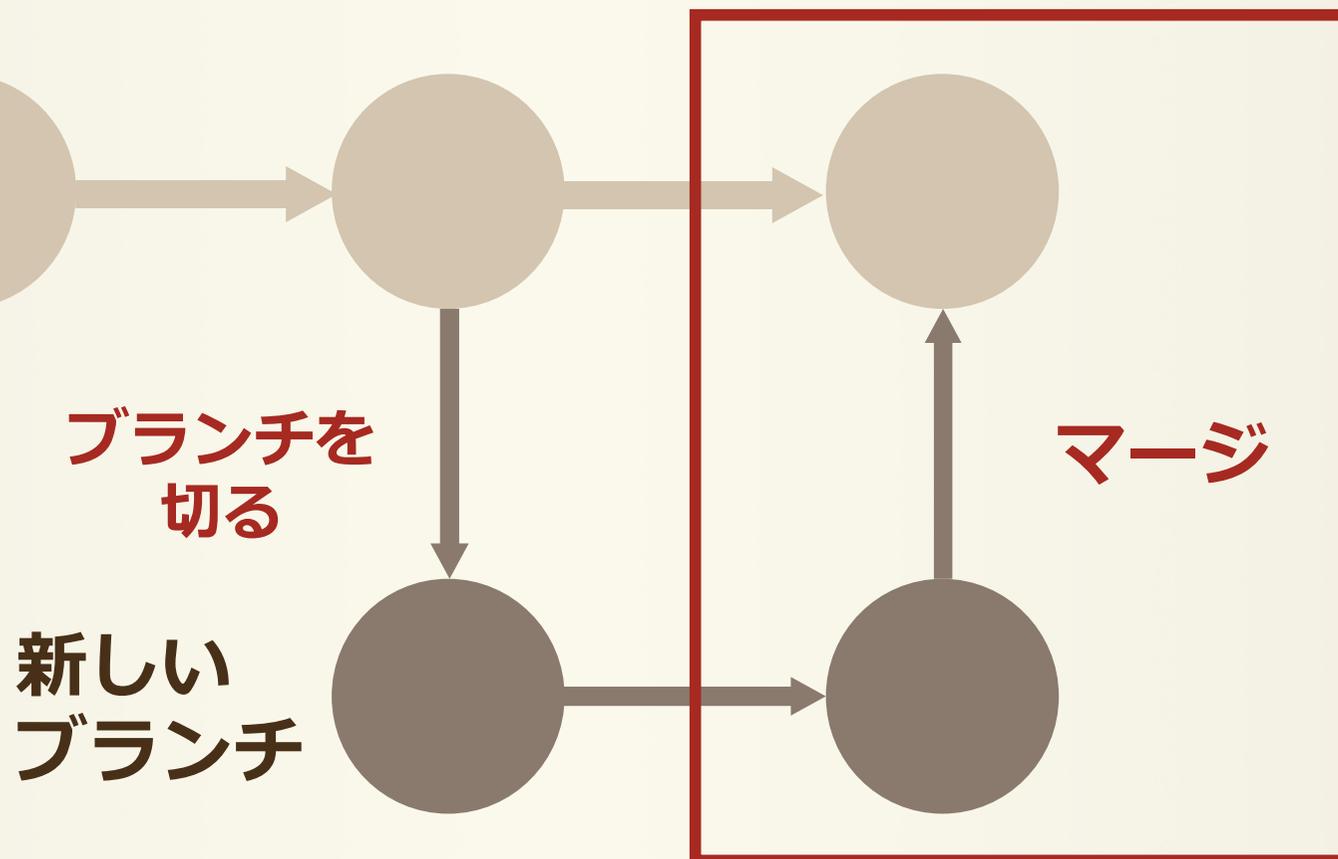
ブランチ

ファイル作成, add, commit, push

```
kut-db:main huncro$ ls
MSP      README.md  aiueo.html  index.html  plactice.html  web      yeah.html
kut-db:main huncro$ vim debug.txt
kut-db:main huncro$ ls
MSP      README.md  aiueo.html  debug.txt  index.html  plactice.html  web      yeah.html
kut-db:main huncro$ git add debug.txt
kut-db:main huncro$ git commit -m 'debug test'
[gucci_debug 6695d0b] debug test
 1 file changed, 1 insertion(+)
 create mode 100644 debug.txt
kut-db:main huncro$ git push origin gucci_debug
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 266 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: This repository moved. Please use the new location:
remote:  https://github.com/KUT-soiweb/main.git
To https://huncro@github.com/KUT-soiweb/temporary-directory.git
* [new branch]      gucci_debug -> gucci_debug
```

ブランチ

master
ブランチ



ブランチ

- **git merge ブランチ名**

現在のブランチに[ブランチ名]での変更をマージ

```
kut-db:main huncro$ git branch
  bpushtest
  gucci
  gucci_debug
* master
  tekito
kut-db:main huncro$ ls
MSP      README.md  aiueo.html  index.html  plactice.html  web          yeah.html
kut-db:main huncro$ git merge gucci_debug
Updating 5d700ac..6695d0b
Fast-forward
 debug.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 debug.txt
kut-db:main huncro$ ls
MSP      README.md  aiueo.html  debug.txt  index.html  plactice.html  web          yeah.html
```

Agenda

1. Gitについて

2. GitHubについて

3. 使ってみる

GitHubでできること

- コードなどの成果物の公開
- ソースコードを通じて他の開発者とコミュニケーション
- Wiki、タスク管理
- 活動状況の表示

etc



GitHub

- 「Git」の「ハブ：拠点・中心・集まり」
- Gitの仕組みを利用して、世界中の人々が自分の作品を保存、公開できるウェブサービス

GitHub

- **プルリクエスト(pull request)**
コードレビュー機能
修正を取り込む際に内容の確認
- **フォーク(fork)**
他の開発者のコードを複製し、編集
- **イシュー(issue)**
課題管理機能

Pull Request

- プルリクエスト(pull request)
コードレビュー機能
修正を取り込む際に内容の確認

こんな変更したから、
masterブランチに取り込んでくれ～



別の人チェック

Pull Request手順(1/5)

KUT-soiweb / main

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 5 Projects 0 Wiki Insights Settings

No description, website, or topics provided. Edit

Add topics

29 commits 11 branches 0 releases 9 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

Kusakalzumi Merge branch 'master' of https://github.com/KUT-soiweb/main Latest commit 5d700ac 6 days ago

MSP	'testです'	6 days ago
plactice.html	練習用のファイルを作成しました	6 days ago
web	ファイル片付け	6 days ago
.DS_Store	削除	6 days ago
README.md	README	8 months ago
aiueo.html	あいうえお	6 days ago
index.html	index編集	6 months ago
yeah.html	イエイ	6 months ago

README.md

Pull Request手順(2/5)

Compare changes

Compare changes across branches, commits, tags, and more below. If you need to, you can also [compare across forks](#).



base: master ▾

...

compare: master ▾

Create pull request

Choose different branches or forks above to discuss and review changes.



Compare and review just about anything

Branches, tags, commit ranges, and time ranges. In the same repository and across forks.

EXAMPLE COMPARISONS

gucci_debug	2 hours ago
rensyu2	on 23 Jun
rensyu1	on 23 Jun
bpushtest	on 31 May
test	on 19 Apr
master@{1day}...master	24 hours ago

gucci_debugというブランチでプルリクを送ってみる

Pull Request手順(3/5)

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

 base: **master** ... compare: **gucci_debug** ✓ Able to merge. These branches can be automatically merged.

 **Create pull request** Discuss and review the changes in this comparison with others. 

 2 commits  1 file changed  0 commit comment  1 contributor

-  Commits on Nov 06, 2017
-  **huncro** debug test 6695d0b
 -  **huncro** 修正箇所の追加 9aea2cc

 Showing 1 **changed file** with 4 additions and 0 deletions. Unified Split

4  debug.txt View  

```
... @@ -0,0 +1,4 @@  
1 +デバッグする箇所を記述していくぞ~  
2 +  
3 +わーい  
4 +ねむーい
```

No commit comments for this range

Pull Request手順(4/5)

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](#).

base: master ... compare: gucci_debug ✓ Able to merge. These branches can be automatically merged.

Gucci debug

Write Preview

修正箇所の記述をしました

debug -> デバッグ
栗原研究室A313 -> A360

どのような変更したか

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

Styling with Markdown is supported

Create pull request

2 commits 1 file changed 0 commit comment 1 contributor

Commits on Nov 06, 2017

huncro debug test 6695d0b

Pull Request手順(5/5)

The screenshot shows a GitHub Pull Request interface for 'Gucci debug #11'. At the top, it indicates 'huncro wants to merge 2 commits into master from gucci_debug'. Below this, there are statistics for 'Conversation' (0), 'Commits' (2), and 'Files changed' (1), along with a progress bar showing '+4 -0' changes. A comment from the owner 'huncro' is visible, stating '修正箇所の記述をしました' (I added descriptions for the correction points) and listing changes: 'debug -> デバッグ' and '栗原研究室A313 -> A360'. The commit history shows two commits: 'debug test' and '修正箇所の追加'. A green box highlights a message: 'This branch has no conflicts with the base branch. Merging can be performed automatically.' Below this is a 'Merge pull request' button. At the bottom, there is a 'Write' section with a text area for comments and a 'Comment' button. On the right side, there are settings for 'Reviewers', 'Assignees', 'Labels', 'Projects', and 'Milestone', all currently set to 'None yet'. A 'Notifications' section shows an 'Unsubscribe' button. At the bottom right, there is a '1 participant' section and a 'Lock conversation' button.

プルリク完了
レビュー待ち

Agenda

1. Gitについて

2. GitHubについて

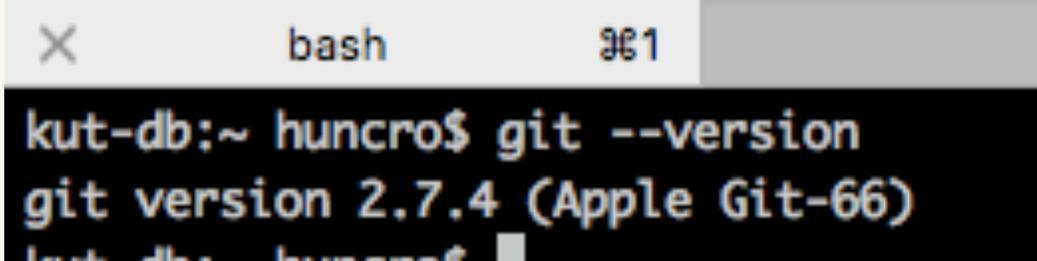
3. 使ってみる

Gitのインストール

• Macの場合

初めから入っていることが多い
ターミナルなどで確認

\$ git --version



```
× bash 981
kut-db:~ huncro$ git --version
git version 2.7.4 (Apple Git-66)
kut-db:~ huncro$
```

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

Downloads -> Mac OS X

pkgファイルを開いて手順に従う

Gitのインストール

- Windowsの場合

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

Downloads -> Windows

exeファイルを開いて手順に従う

コマンドプロンプト (または Git Bash) で確認

```
$ git --version
```

実際に使ってみる

1. mkdir ディレクトリ名
2. cd ディレクトリ名

ここはgitで管理する
ディレクトリということが
わかればなんでもいい

Gitの初期設定

- ユーザ名の設定

```
git config --global user.name "ユーザ名"
```

- メールアドレスの設定

```
git config --global user.email メールアドレス
```

→ 確認

```
git config -l
```

**GitHubでも
同じものを使う**

GitHubの登録

- アカウント作成

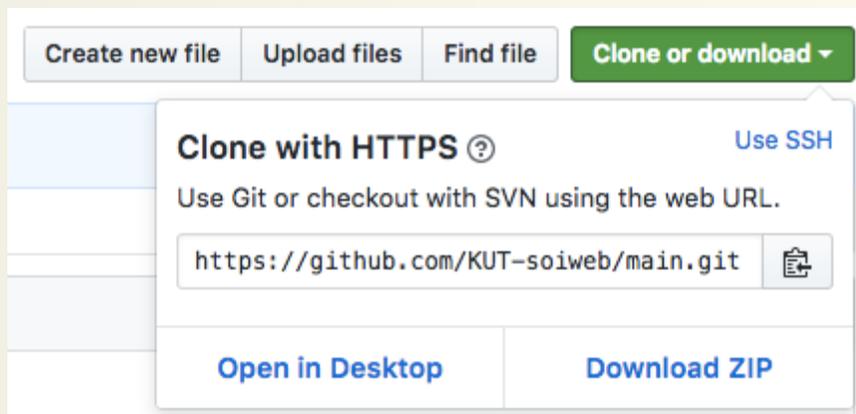
<https://github.com/>

取得先によって変わる

- soiwebのリポジトリを取得

`git clone https://ユーザ名@github.com/~`

登録したユーザ名



実際に使ってみる(ブランチなし)

1. clone してきたフォルダに入る (今回は cd main)

2. ファイル作成

git pull

→ 他の誰かがpushしていた場合
適宜必要

3. git add ファイル名

4. git commit -m 'どのような編集をしたのか'

5. git push origin master

実際に使ってみる

その他のそこそこ使うコマンド

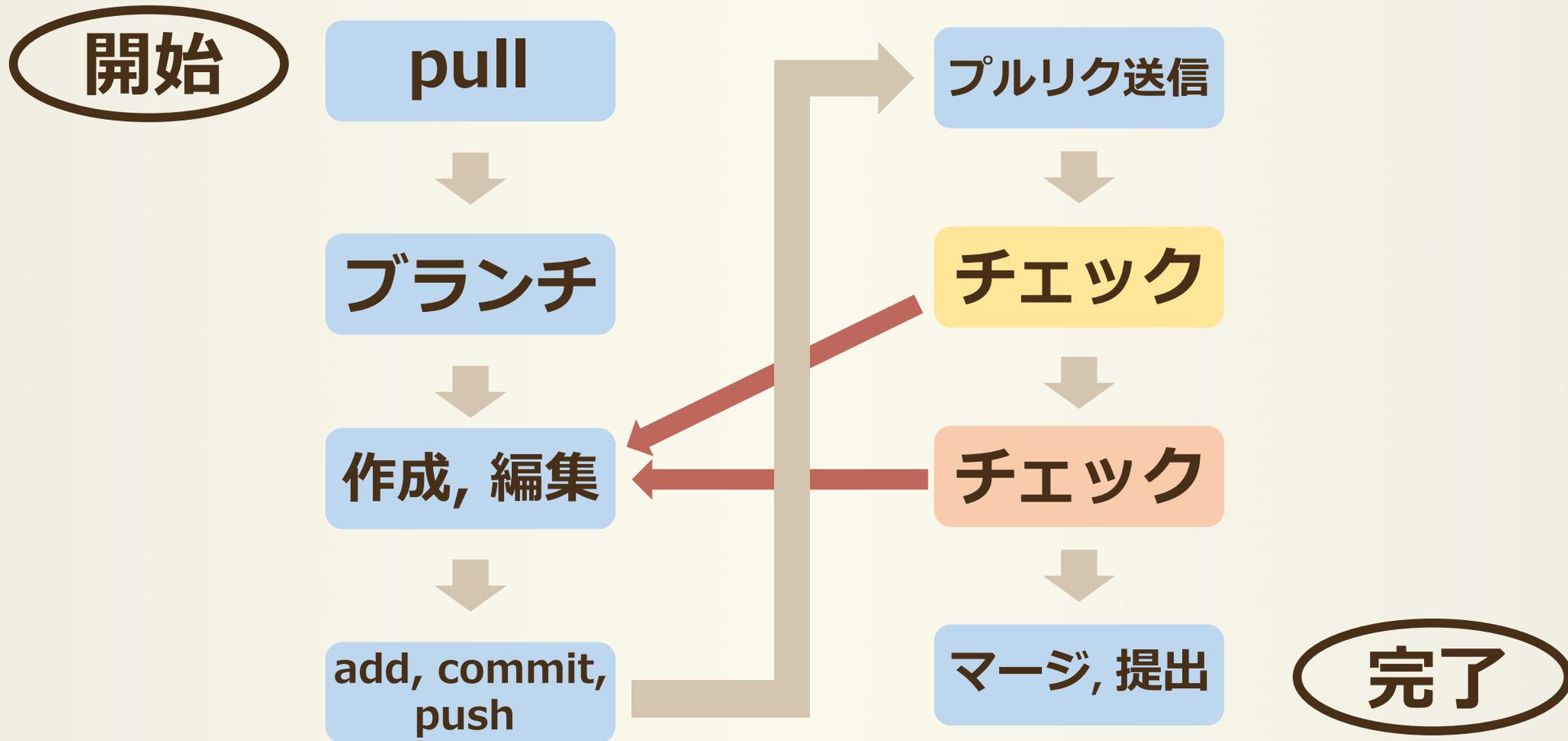
- `git log`
- `git status`
- `git reset`
- `git rm`

各自調べてくれ

実際に使ってみる(ブランチあり)

次の流れに従って使ってみよ～

そういうえぶふるーちやーと



そういうえぶふるーちやーと

pull

`git pull origin master`

- リモートリポジトリの変更を反映
これを忘れると、
旧バージョンのファイルをいじっている状態

そういうえぶふるーちやーと

pull



ブランチ

git branch 新ブランチ名
git checkout 新ブランチ名

- ブランチの作成及び移動

そういうえぶふるーちやーと

pull



ブランチ



作成,編集

エディタ名 ファイル名

エディタ名 = vim, atom, emacs など

- ターミナルからファイルを開く
環境があることが前提, (windowsもできたはず)

そういうえぶふるーちやーと

pull



ブランチ



作成,編集



add, commit,
push

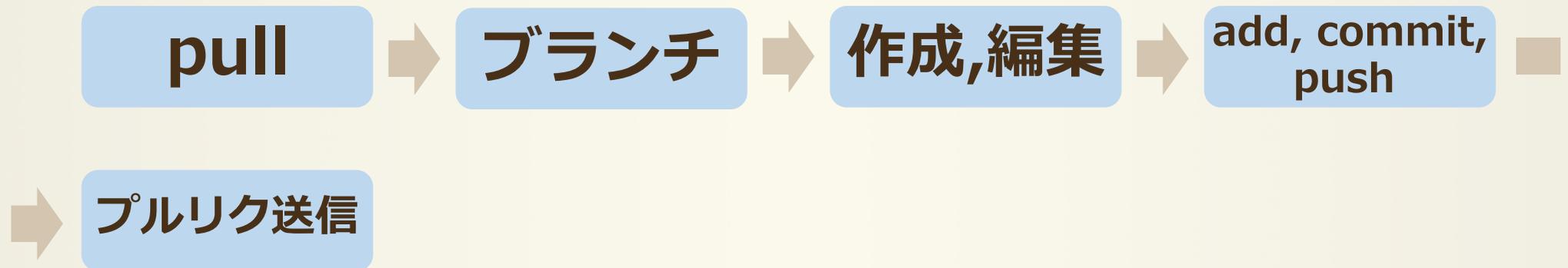
git add ファイル名

git commit -m '内容'

git push origin ブランチ名

- コミットの内容は「どのような変更したか」、など
- ブランチ名でpush

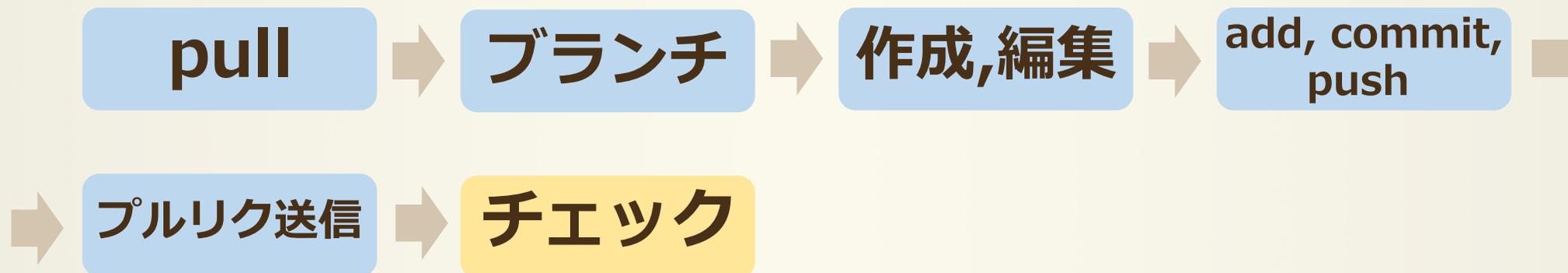
そういうえぶふるーちやーと



GitHubのページ上からPull Requestを送信する

- P.41～ Pull Requestの手順参照

そういうえぶふるーちやーと



git pull origin master

git checkout origin/ブランチ名

↓ Pull Requestを
送ってきたブランチ名

• このブランチの状況を見る

ファイルを開いて確認

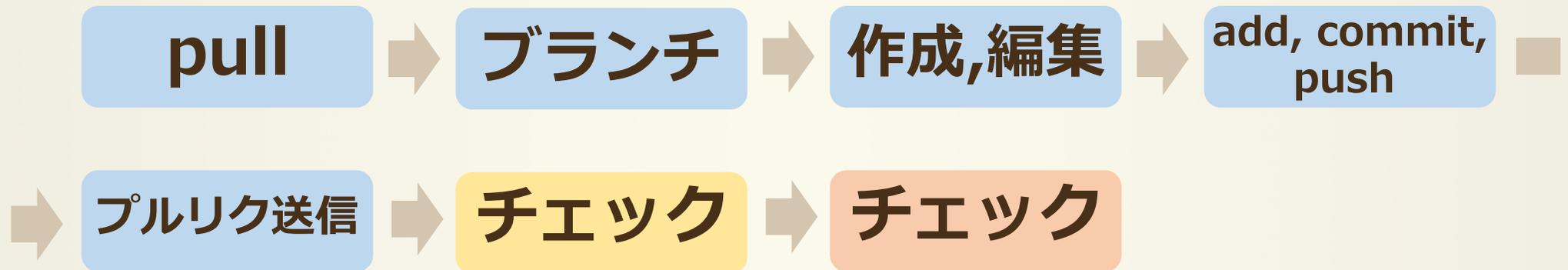
- open ファイル名
- ファイル名

(mac)

(windows)

HTMLファイルなどを
ブラウザから開けたはず

そういうえぶふるーちやーと



一つ前の手順と同じ

- ソースコード見るにはブランチ移動して
エディタ名 ファイル名

そういうえぶふろーちやーと



チェック後

- slackから先生にファイル提出
- GitHubからマージ
- (ブランチ削除)

まとめ

- 細かいコマンドはまだまだある
- リモートとのやりとりは、push, pull
- 定期的にpullしてリモートとのズレを抑える