

Github

2019/10/23

木南 貴志

1. Github とは

Githubとは

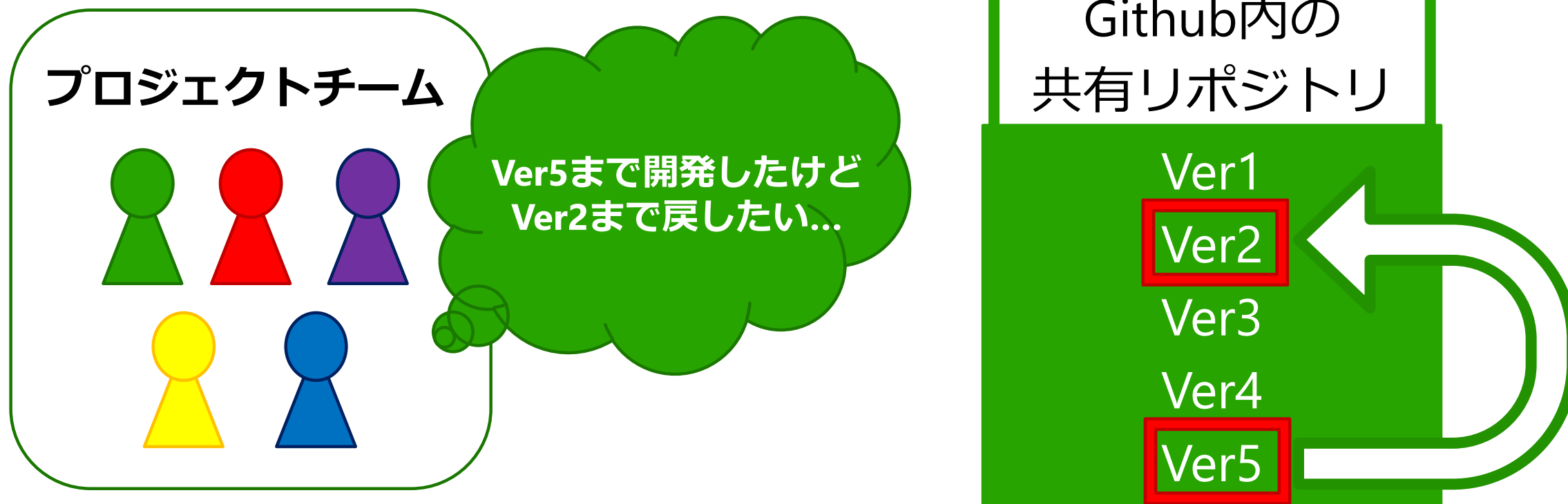
ソフトウェア開発のプラットフォーム

複数人でプログラムを開発するときによく用いられる



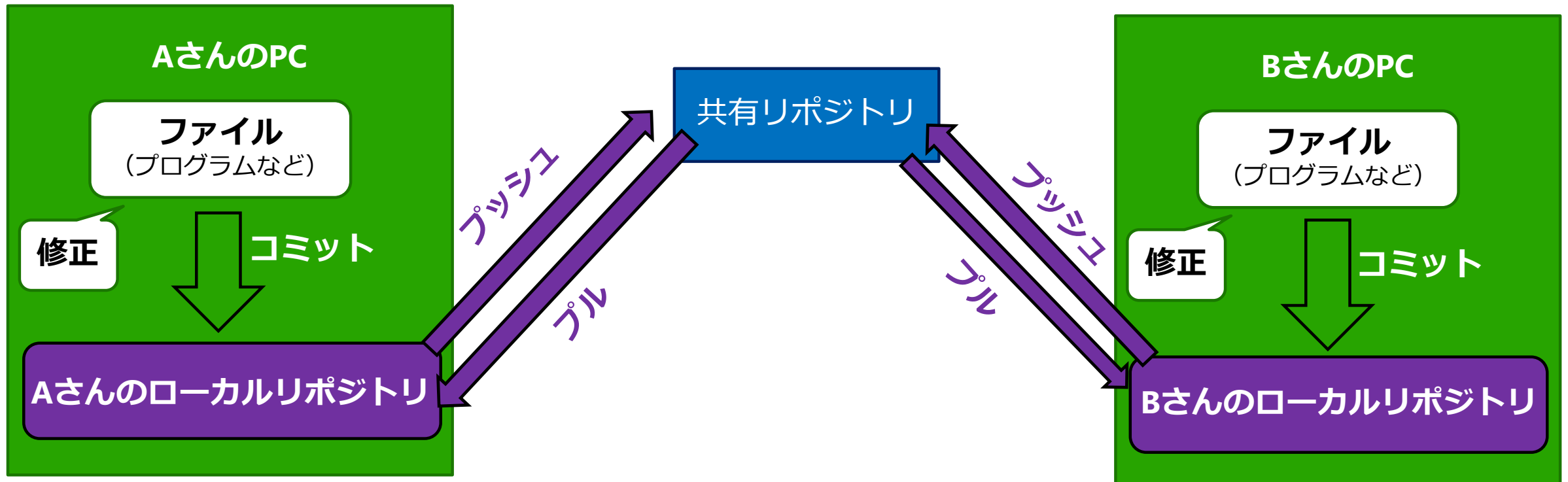
Githubとは

バージョン管理システムがあり、ファイルの履歴を残しておくことで以前の状態に復元可能



分散型バージョン管理システム

- リポジトリ：ファイルやディレクトリを保存する場所
- コミット：ローカルリポジトリに記録すること
- プッシュ：ローカルリポジトリの内容を共有リポジトリに反映すること
- プル：共有リポジトリの内容をローカルリポジトリに反映すること



Githubを使うための準備

- gitのインストール

```
$ sudo apt-get update  
$ sudo apt-get install git
```

- ユーザの設定 (.gitconfigの設定)

```
$ git config --global user.name "NAME"  
$ git config --global user.email "E-MAIL"
```

※NAMEとE-MAILは各自で設定 (Githubの公式サイトで登録したもの)

エディタの設定 (エディタをvim形式に変更)

```
$ git config --global core.editor "vim -f"
```

2. リポジトリ、プッシュ、プル

リポジトリの種類

- Publicリポジトリ
 - すべての人に公開できる
 - **機密情報は含めないようにする**
- Privateリポジトリ
 - 基本的に有料
 - 権限を持っている人に対してのみ公開される

リモート（共有）リポジトリ

- GithubのWebページへ <https://github.co.jp/>

①リポジトリを作る場所を選択

kinami-takashi ▾

今回は自分のアカウント
の中に作成します

- cu-milab/semi
- cu-milab/semi-homework-kinami-...
- platypus5384/learn_python
- cu-milab/pro-ena-games
- kinami-takashi/learn_python
- cu-milab/prac-github
- cu-milab/ra-kinami-2019

Show more

【②New repository】
をクリック

New repository

Import repository

New gist

New organization

New project

Learn Git and GitHub without any
code!

Using the Hello World guide, you'll create a repository, start a
branch, write comments, and open a pull request.

Read the guide

Start a project



Securin

Introduc

prevent

your cod



Welcome to the new dashboard. Get
closer to the stuff you care about most.

Explore repositories

styled-components/styled-components-website

The styled-components website and documentation

JavaScript ★ 369

knsv/mermaid

Generation of diagram and flowchart from text in a

リモート（共有）リポジトリ



Create a new repository

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

Owner

kinami-takashi

Repository name *

test

リポジトリの名前

Great repository names are short, lowercase, and contain only numbers, lowercase letters, and hyphens. Your new repository will be created as test- about expert-disco?

Description (optional)

test用のリポジトリ

リポジトリの説明



Public

Anyone can see this repository. You choose who can commit.

公開・非公開の設定



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.

Readmeとライセンスの設定

Add .gitignore: None

Add a license: None



Create repository

リモート（共有）リポジトリ

kinami-takashi / test リポジトリが作成されているか確認

<> Code 0 Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH **https://github.com/kinami-takashi/test-.git** **リポジトリのURL**

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# test-" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/kinami-takashi/test-.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/kinami-takashi/test-.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

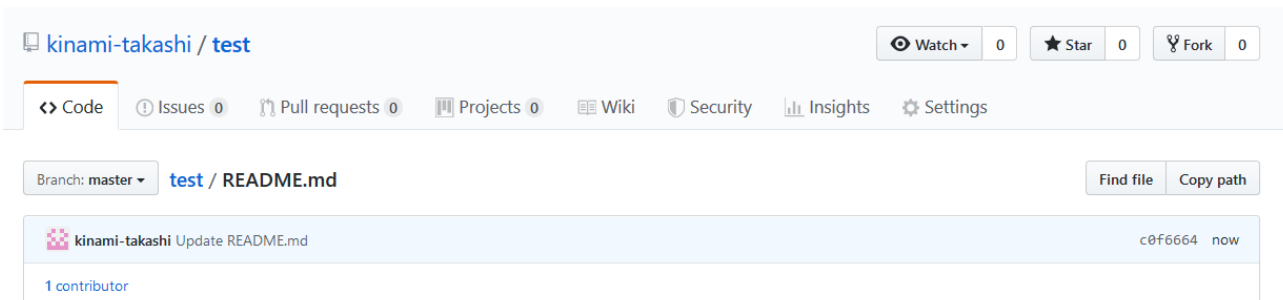
ローカルリポジトリから
リモートリポジトリにプッシュ

リモートリポジトリから
リモートリポジトリにプッシュ

他のバージョン管理方法からイ
ンポート

作業用ディレクトリの作成

今回は作業用ディレクトリ内にREADME.mdを作成
-ディレクトリのトップにREADME.mdを作成するとリポジトリのトップに表示される



README.mdに作成した内容はここに表示される



作業用ディレクトリの作成

今回は作業用ディレクトリ内にREADME.mdを作成

-ディレクトリのトップにREADMEを作成するとトップに表示される

```
$ mkdir test
```

リモートリポジトリ（11ページで作成したもの）と同じ名前にする

```
$ cd test
```

```
$ vim README.md
```

作業用ディレクトリの作成

README.mdの記入例)

test

====

これはテスト用のリポジトリです

README.mdとは

README.md内にはリポジトリの説明を記入します

README.mdには特有の書き方が存在します

たとえば【## タイトル】と記入すると見出しを作成することができます

13ページの例を再現

test

これはテスト用のリポジトリです

README.mdとは

README.md内にはリポジトリの説明文を記入します

READMEには特有の書き方が存在します

例えば【## タイトル】と記入すると見出しを作成することができます

ローカルリポジトリの作成

共有したいディレクトリに移動

```
$ git init
```

ローカルリポジトリの作成が成功すると.gitディレクトリが作成されるのでlsで確認

※-aは隠しファイルを見るコマンド

```
$ ls -a
```

```
... .git test
```

インデックスに追加

- ・ 作業ディレクトリのファイル等をインデックスに追加
(gitで管理したいファイル等を登録すること)
- ・ ファイルを一つだけ指定する方法

```
$ git add [file]
```

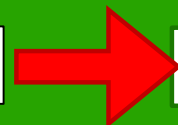
- ・ 全てのファイル（新規作成、変更されたもの）を指定する方法

```
$ git add .
```

今回はこっち

PC

変更・追加したファイル

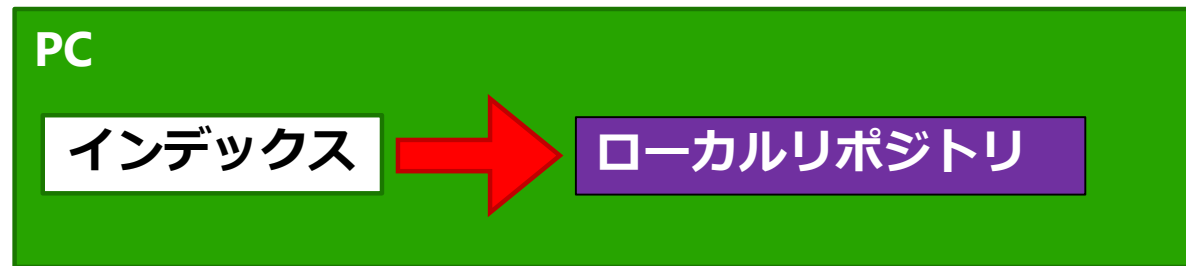


インデックス

インデックスに記録されたデータをコミット

- ・ ローカルリポジトリに記録
 - ・ -mはコメント機能、必須ではないが記述するのが通例

```
$ git commit -m "コメント"
```



インデックスの取り消しと確認

間違えて登録した時は取り消すこともできる

- ・登録したものの中で一つだけ取り消す

```
$ git reset HEAD [file]
```

- ・登録したものの全てを取り消す

```
$ git reset HEAD
```

- ・インデックスに登録されているファイルを確認

```
$ git status
```

リモートリポジトリを登録

- ・ URLと識別子（名前）をつける

※新しくリポジトリを作成した時のみ実行する

```
$ git remote add [short-name] [URL]
```

- ・ 登録の例

よくつかわれる名前（違う名称でも可）

```
$ git remote add origin https://github.com/kinami-takashi/test.git
```

リモートリポジトリ作成時のURL

上記の場合 origin = https://github.com/kinami-takashi/test.git

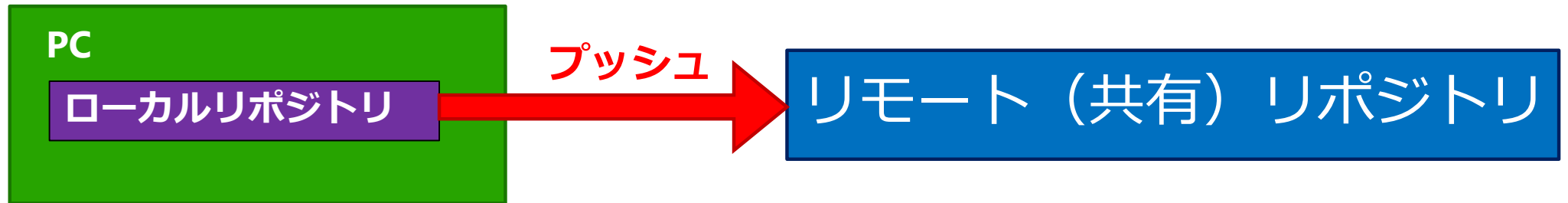
※以降の説明では識別子はすべてoriginとして説明

リモートリポジトリへ送信（プッシュ）

Githubにアップロード

```
$ git push -u origin master
```

※ **master**=メインのブランチ



リモートリポジトリを取得（プル）

ローカルリポジトリを更新

- originが設定済みの場合のみ実行可能

```
$ git pull origin master
```



Bさんが、ローカルリポジトリのファイルを追加・修正しプッシュして、リモートリポジトリを更新
⇒Aさんが更新内容を確認するためにリモートリポジトリをプルして自身のローカルリポジトリを更新

Organization

organizationはチームのようなもの

同じチームで色々なリポジトリを共有したい時などに使用される

イメージ)

challenge-site-kamonohashi (organization)

**learn_python
(repository1)**

**micro_mouse
(repository2)**

3. クロージン

リポジトリのクローン

常に作成してあるリポジトリが欲しい ➡ クローン

The screenshot shows the GitHub interface for the repository 'platypus5384 / learn_python'. The repository description is 'repository for fleshmans learning python'. It has 12 commits, 1 branch, 0 releases, and 2 contributors. A modal window titled 'Clone with HTTPS' is open, showing the URL 'https://github.com/platypus5384/learn_py' and options to 'Open in Desktop' or 'Download ZIP'. The modal is highlighted with a red rectangle. Below the repository details, there is a table of files and their commit hashes.

File	Commit
homework/1	test pullrequest
README.md	modify README
kinami.txt	aaaaaa

learn_python

欲しいリポジトリのURLをコピー

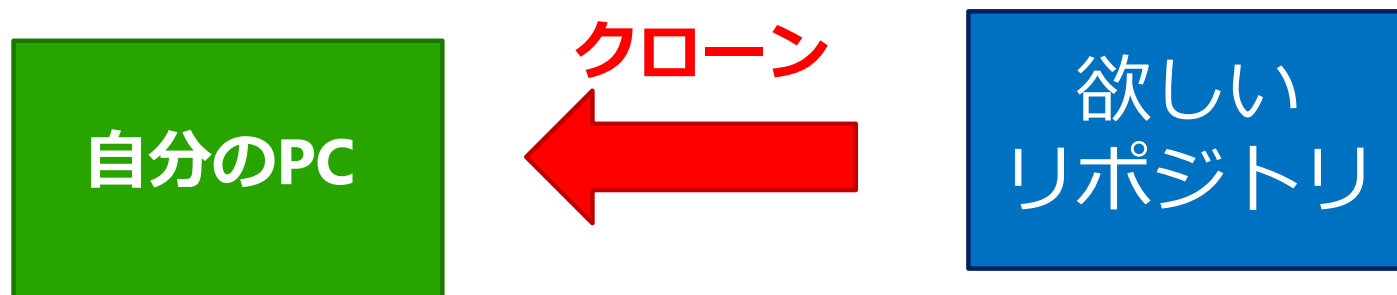
クローンの仕方

クローンの方法

```
$ git clone [URL]
```

今回は以下を実行

```
$ https://github.com/challenge-site-kamonohashi/learn_python.git
```

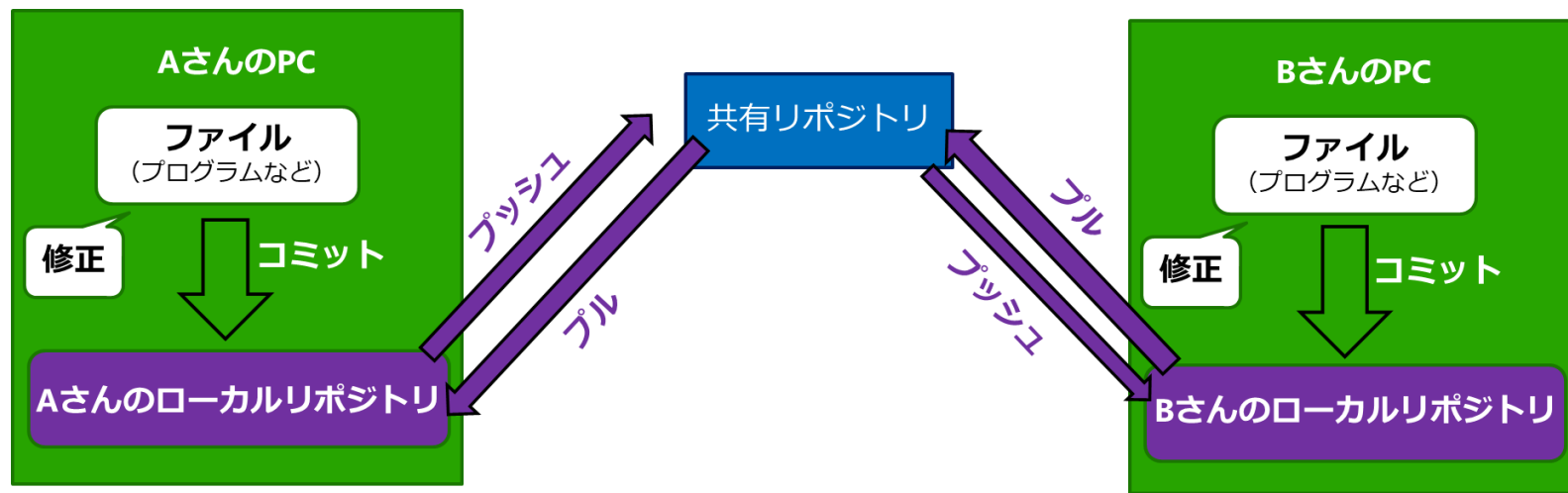


4. ブランチ

ブランチとは

前述ではファイル内の変更した内容をmasterでプッシュしていた

➡複数人で作業する時に全員がmasterでプッシュすると不具合の可能性



全員が好き勝手にpush

- ・プログラムが動かなくなった
- ・変更点が分からなくなった など

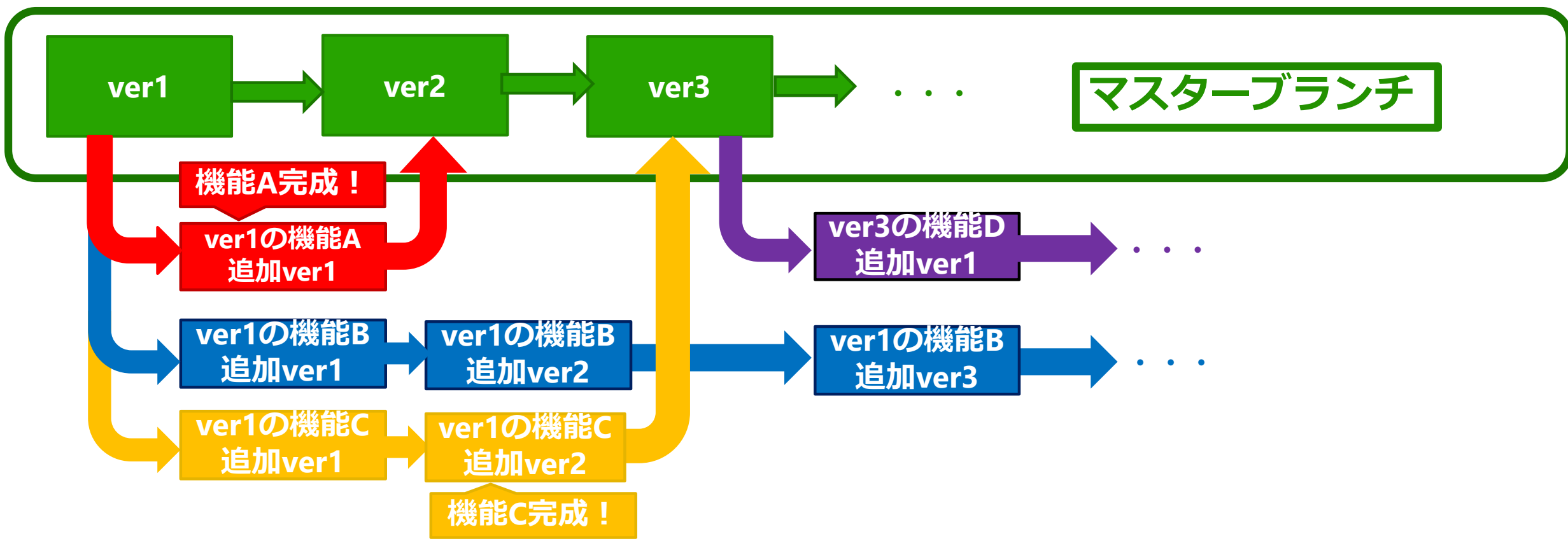


ブランチを使用する

ブランチとは

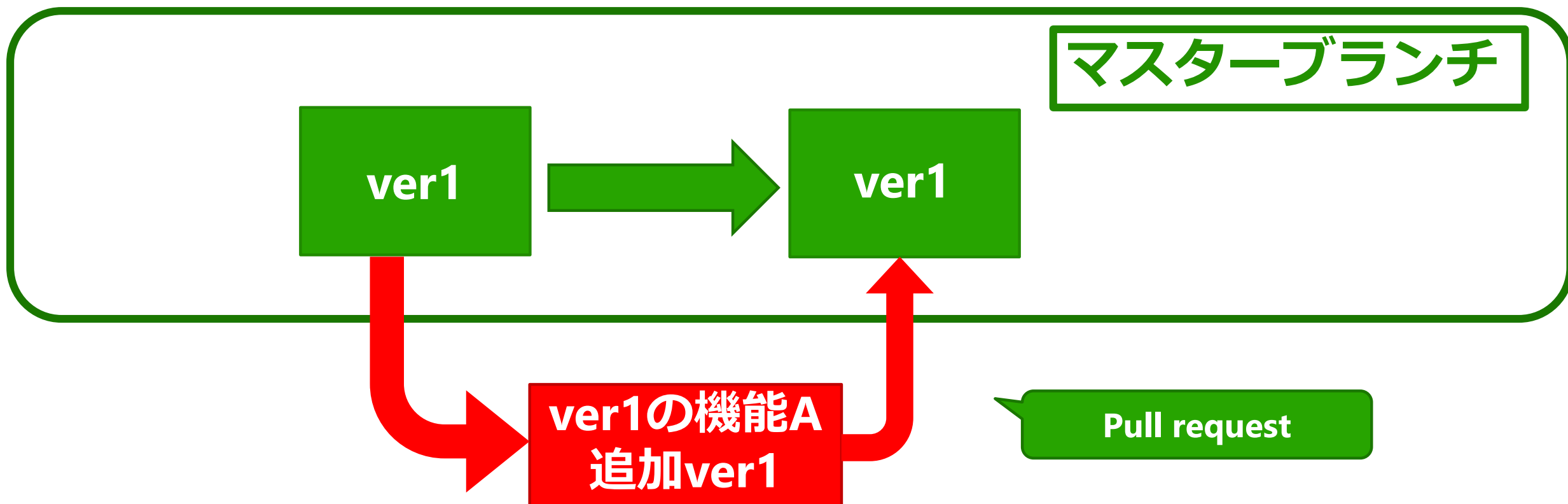
ブランチは履歴の流れを分岐して記録するもの

そのなかでもメインのブランチをマスターブランチという



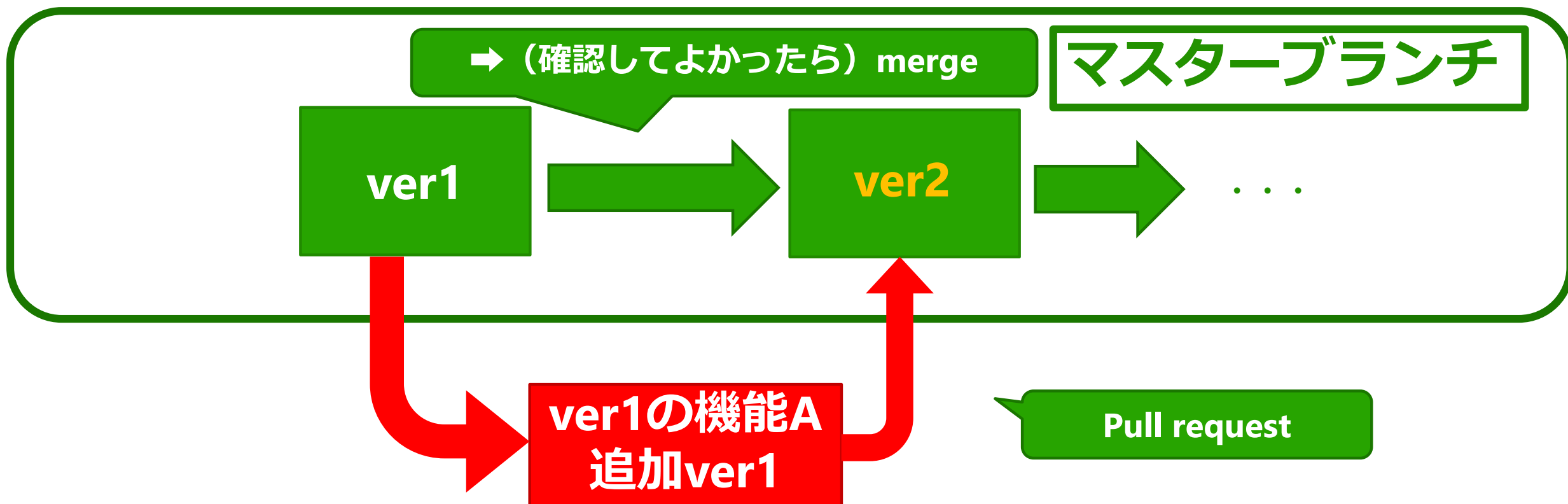
プルリクエスト

ブランチの作業内容をマスターに反映させるにはブランチから**プルリクエスト**を行う
内容を確認してよかったらマージ（統合）する



プルリクエスト

ブランチの作業内容をマスターに反映させるにはブランチから**プルリクエスト**を行う
内容を確認してよかったらマージ（統合）する



ブランチのコマンド

ブランチの一覧確認

```
$ git branch
```

ブランチ作成

```
$ git branch [ブランチ名]
```

ブランチ切り替え

```
$ git checkout [ブランチ名]
```

ブランチ作成 & 切り替え

```
$ git checkout -b [ブランチ名]
```

ブランチをpush

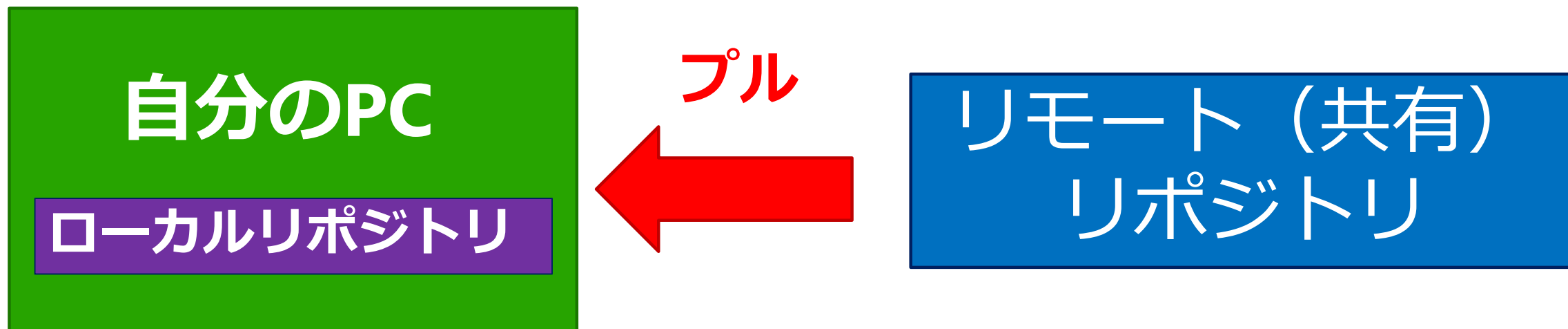
```
$ git push origin [ブランチ名]
```

プルリクエストの例

例) ①リポジトリをpullして自分のローカルリポジトリを最新にする

※常にリポジトリが存在する場合

```
$ git pull origin master
```



プルリクエストの例

例) ②ローカルリポジトリの中身を変更（下の例ではhello.pyを変更）

自分のPC

ローカルリポジトリの中身

- README.md
- hello.py

```
print("hello")
```

プルリクエストの例

例) ②ローカルリポジトリの中身を変更（下の例ではhello.pyを変更）

自分のPC

ローカルリポジトリの中身

- README.md
- hello.py

```
print("hello")  
print("good mornig")
```

一行追加

プルリクエストの例

例) ③ブランチを作成してリポジトリをプッシュ

※作成するブランチには任意の名前をつけられる 下の例ではadd-greetingという名前のブランチ

<code>\$git checkout -b [ブランチ名]</code>	ブランチ作成&切り替え
<code>\$git branch</code>	ブランチが切り替わったか確認
<code>\$git add .</code>	ファイルの登録
<code>\$git commit -m "コメント"</code>	ファイルのコミット
<code>\$git push origin [ブランチ名]</code>	ファイルのプッシュ

自分のPC

変更した
ローカルリポジトリ



ブランチ
add-greeting

現在のブランチは下図のように緑文字で表示されます
意図していないブランチを更新するといけなないので切り替わっているか必ず確認しましょう

```
kinami@kinami-ZenBook-13-UX331UAL:~/learn_python$ git branch
aaaaa
* add-greeting
master
```

プルリクエストの例

例) ④ ブランチにpushしたらgithubのサイトでプルリクエストする

ブランチにプッシュすると下記のように
プルリクエストをするように表示される

platypus5384 / learn_python

Watch 1 Star 0 Fork 11

Code Issues 0 Pull requests 1 Projects 0 Wiki Security Insights

repository for fleshmans learning python

12 commits 1 branch 0 releases 2 contributors

kinami-takashi: add-greeting (12 minutes ago) Compare & pull request

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

powdersnow1442 test pullrequest	Latest commit dd47731 13 days ago
homework/1	test pullrequest 13 days ago
README.md	modify README 19 days ago
kinami.txt	aaaaaa 20 days ago

表示されなかった場合は
【New pull request】をクリックする

platypus5384 / learn_python

Watch 1 Star 0 Fork 11

Code Issues 0 Pull requests 2 Projects 0 Wiki Security Insights

repository for fleshmans learning python

12 commits 1 branch 0 releases 2 contributors

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

powdersnow1442 test pullrequest		Latest commit dd47731 13 days ago
homework/1	test pullrequest	13 days ago
README.md	modify README	19 days ago
kinami.txt	aaaaaa	20 days ago

プルリクエストの例

例) ④ ブランチにpushしたらgithubのサイトでプルリクエストする

Open a pull request

リクエストされる側 changes across two branches. If you need to, you can also [compare across forks](#).

base: master → compare: add-greeting ✓ Able to merge. These branches can be automatically merged.

リクエストする側

変更点等を記入

Add greeting

Write Preview

挨拶にhelloを追加しました

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

Create pull request

記入したらクリック⇒【pull request】完了

Reviewers: No reviews

Assignees: No one—[assign yourself](#)

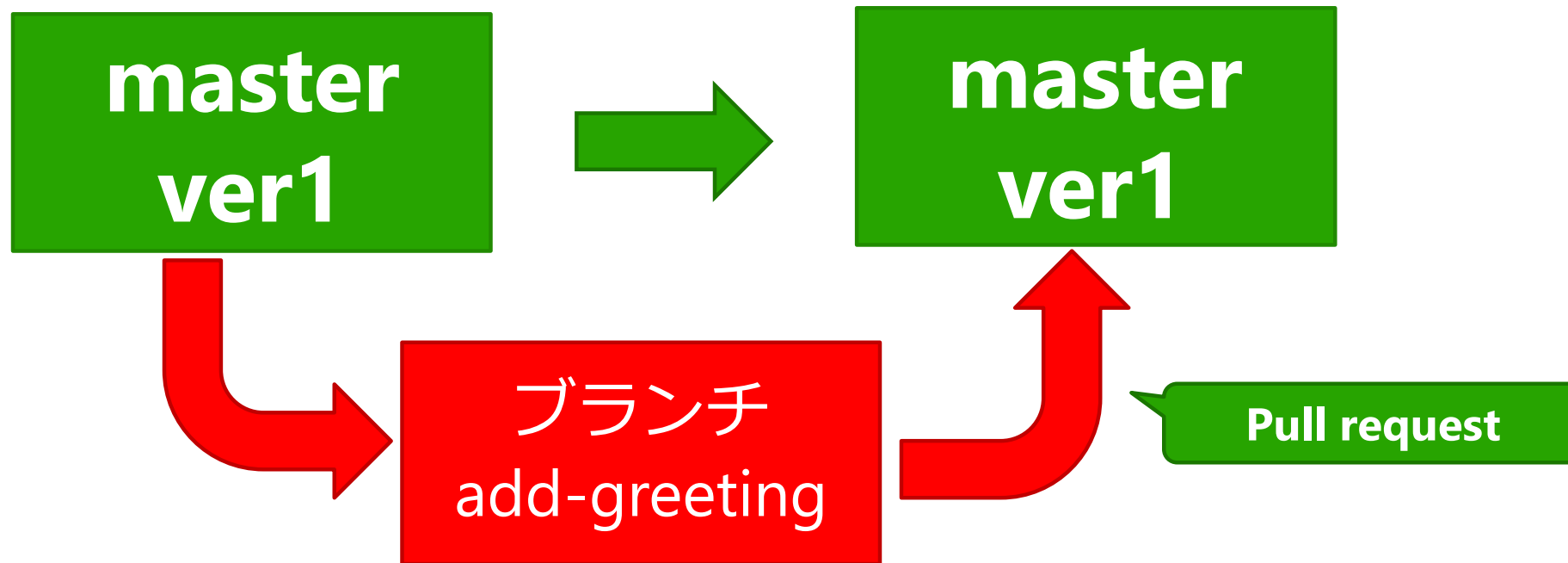
Labels: None yet

Projects: None yet

Milestone: No milestone

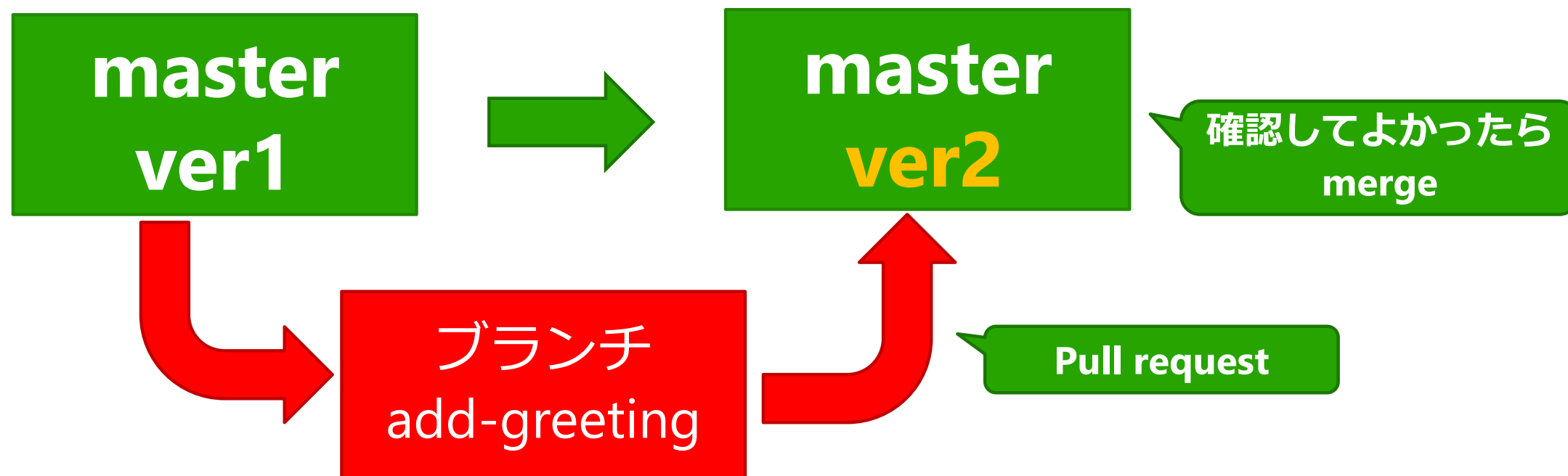
プルリクエストの例

例) ⑤masterにプルリクエストが承認されたらマージ（統合）される



プルリクエストの例

例) ⑤masterにプルリクエストが承認されたらマージ（統合）される



コンフリクト（競合）

同じファイルの同じ箇所を編集してプルリクエストを行うとコンフリクト（競合）が起こる



コンフリクト（競合）

コンフリクトはなるべく起こらないほうがよいので

- ・ あらかじめ役割分担して同じ箇所を編集しないようにする
- ・ マスターが変更されたら変更をブランチに取り込む（下記に方法を示す）
等の対策をとるとよい

```
$git checkout master
```

ローカルのマスターに移動してプル

```
$git pull origin master
```

```
$git checkout [ブランチ名]
```

ブランチに移動してマージ

```
$git merge master
```

```
$git add .
```

```
$git commit -m "Merge"
```

```
$git push origin [ブランチ名]
```

4. イシュー

issue

issueは課題を出題する時や、コードの内容を議論したりするときに使用

例) 2019/10/23 課題 #4

 Open kinami-takashi opened this issue 39 minutes ago · 1 comment



kinami-takashi commented 39 minutes ago

Member + 😊 ...

今週の課題
課題をpull requestしたらコメントしてください



Tanaka-taro commented 23 seconds ago

Author Member + 😊 ...

課題を pull request しました
[#3](#)



Write

Preview

AA B i “ <> 🔗 ☰ ☷ ☰ @ 📎 ↶

Leave a comment

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



 Close issue

Comment

issueの作成方法

①タブ【Issues】をクリック ➡ 【New issue】をクリック

① 【issues】をクリック

The screenshot shows the GitHub repository page for 'learn_python'. The 'Issues' tab is selected and highlighted with a green box. Below the repository header, there is a notification banner about labeling issues for new contributors. The 'Filters' section shows 'is:issue is:open'. The 'New issue' button is highlighted with a green box. The list of issues shows one open issue titled 'テスト' (Test) created 27 minutes ago by kinami-takashi.

Label issues and pull requests for new contributors
Now, GitHub will help potential first-time contributors [discover issues](#) labeled with `good first issue`

Filters Labels 9 Milestones 0

☐ 1 Open ✓ 2 Closed Author Labels Projects Milestones Assignee Sort

☐ 🟢 テスト #2 opened 27 minutes ago by kinami-takashi 1

② 【New issue】をクリック

issueの作成方法

②issueのタイトルと概要を書く

The screenshot shows the GitHub interface for creating a new issue. The repository is 'challenge-site-kamonohashi / learn_python'. The 'Issues' tab is selected. A green box highlights the title input field containing '2019/10/23 課題'. Another green box highlights the body text area containing '今週の課題' and '課題をpull requestしたらコメントしてください'. A third green box highlights the 'Submit new issue' button at the bottom right.

challenge-site-kamonohashi / learn_python

Watch

Code Issues Pull requests Projects Wiki Security Insights Settings

①issueのタイトルを記入

2019/10/23 課題

Related Issues Beta Try it.

②issueで議論する内容の詳細を記入

今週の課題
課題をpull requestしたらコメントしてください

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

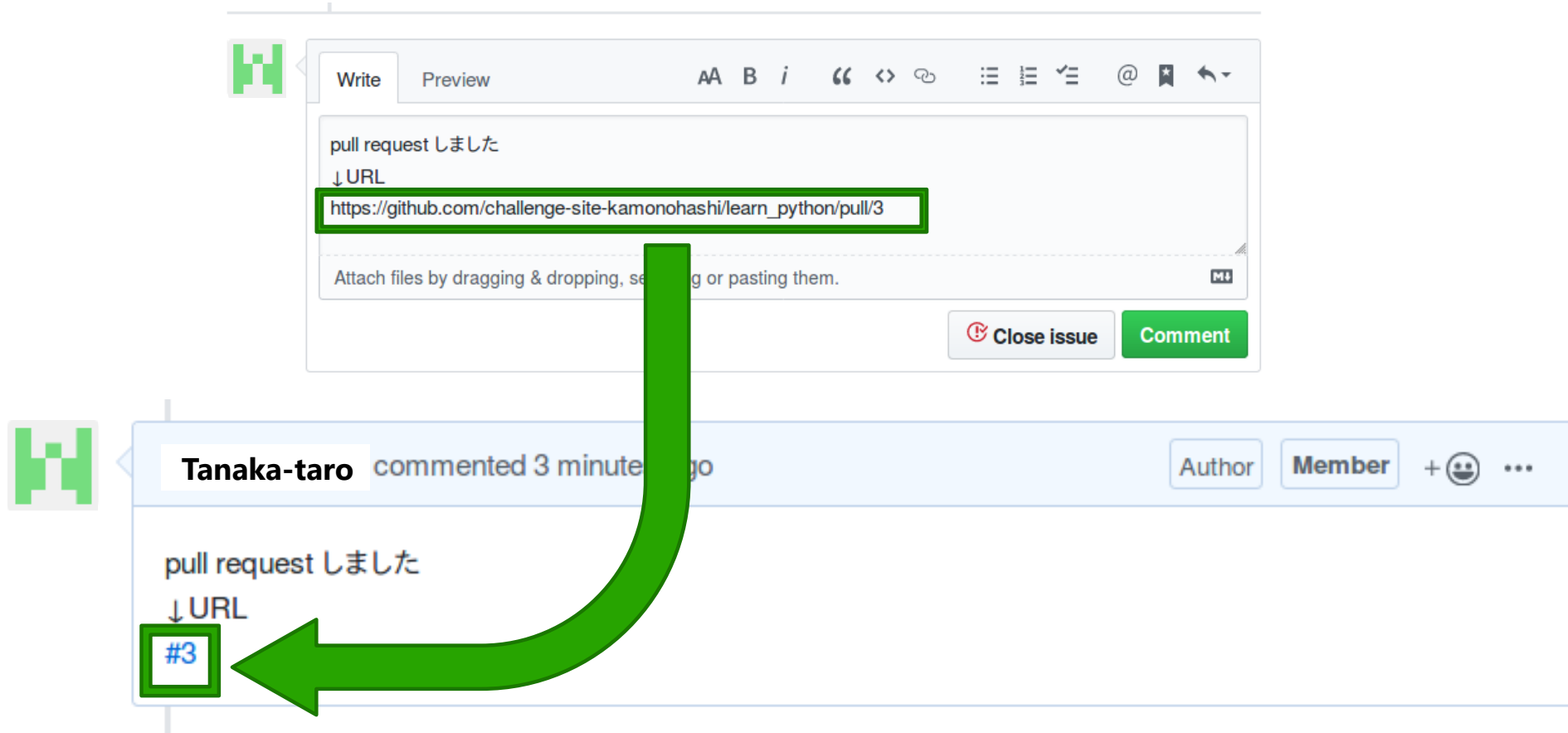
Styling with Markdown is supported

Submit new issue

③記入が終わったら
クリック

Issueでのコメントの例

例えばpull requestした時のURLをコメントするといい感じに短縮されます



issueのその他の機能

issueには

担当者を明確にする「Assignee」機能

スケジュールを管理する「Milestone」機能

増えたIssueを整理する「Label」機能

など 管理するのに便利な機能が色々あります

参考サイト <https://seleck.cc/647>

課題

- ①リポジトリ【learn_python】内の【kadai】フォルダ内の【day1】フォルダ内に【2019_10_23_（自分の名前）.py】作成する。（.pyの中身は何も記述しなくてもよい）
- ② ①で変更されたリポジトリをブランチにpushする
(ブランチ名は自分の名前とする)
- ③プルリクエストをする
- ④プルリクエストをしたらissue【2019/10/23課題】にてコメントする
(46ページのようにURLも記述すること)

最後に

challenge-site-kamonohashi 内では何をしてもいいので（最悪ファイル等が消えてしまっても大丈夫です）githubで他にも試してみたいことがあったら色々試してみてください