

VSCode
Git

VSCodeの 拡張機能

VSCodeのプラグイン

Dracula theme / Material Icon Theme

Bracket Pair Colorizer 2 / Highlight Matching Tag

Laravel Snippets

Laravel Blade Snippets

Laravel Extra Intellisense

Laravel-goto-Controller

Laravel goto view

PHP InteliPhense ->追加設定必要

Tailwindcss Intellisense

DotEnv

Git

Git バージョン管理システム

特徴

- ・ 更新履歴の確認
- ・ 保存した(コミット)ポイントに戻せる ・ ・
セーブポイント
- ・ 保存場所の切換(ブランチ)
- ・ 共同作業しやすい(マージ・プルリク)

Git と GitHubの違い

Git ・ ・ バージョン管理の仕組み

GitHub ・ ・ Gitをweb上で使える仕組み(ウェブサイト)

Pull Request(プルリク)ができる
→コード差分を確認し問題なければ取り込みで
きる。

フォーク、マージ

Gitのインストール

Gitのインストール

Gitのサイト

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

brew -v (homebrewのバージョン確認)

brew install git (gitのインストール)

git --version (バージョン確認)

最新バージョンになっていればOK

Gitのパスを通す

`sudo vi ~/.bash_profile`

(viエディタは aやiで挿入モード

Escで閲覧モード

閲覧モード時に `:wq` で保存して終了)

パス設定

`PATH=/usr/local/bin/git:$PATH`

`export PATH`

変更の反映 `source ~/.bash_profile`

GitHubの登録

GitHubのサイト

<https://github.com/>

GitとGitHubの連動

https または ssh

2021年8月～ パスワード認証廃止

トークンベースの認証必須

(SSHベース、2要素認証は影響しない)

<https://gigazine.net/news/20201219-github-token-git-operations/>

Gitの初期設定

誰がコミットしたかの記録のため

`git config —global user.name “ユーザ名”`

`git config —global user.email “メール”`

Gitのイメージ

ローカルとリモート リポジトリ

リモート



リポジトリ(貯蔵庫)

ローカル



リポジトリ(貯蔵庫)

git init (ローカル生成)
リモートはGitHub上で作成

リモートリポジトリの登録

リモート



リポジトリ(貯蔵庫)

ローカル

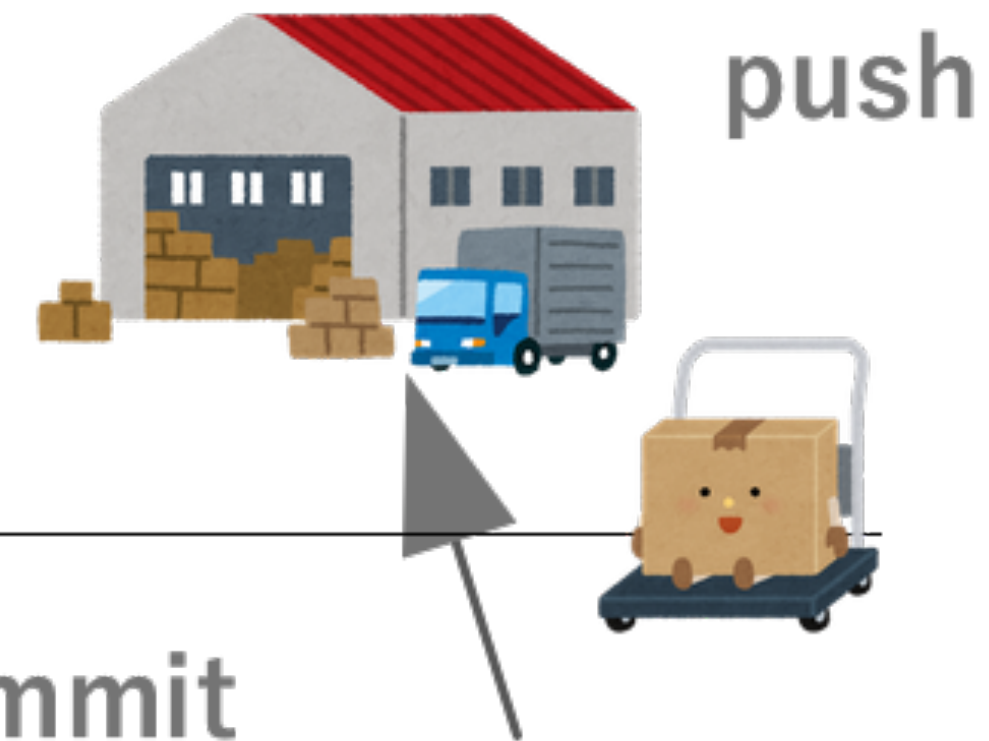


リポジトリ(貯蔵庫)

git remote add ローカル側 リモート側
git remote (一覧表示)

add, commit, push

リモート



ローカル



`git add -A` . (インデックス(ステージ)に全て追加)

`git commit -m “コメント”` (セーブポイント)

`git push origin main` (-u 設定すると `git push` のみでもOK)

初回登録の補足

git branch -M ブランチ名を強制書換

(以前はmaster

master/slave

人権問題により mainに変更

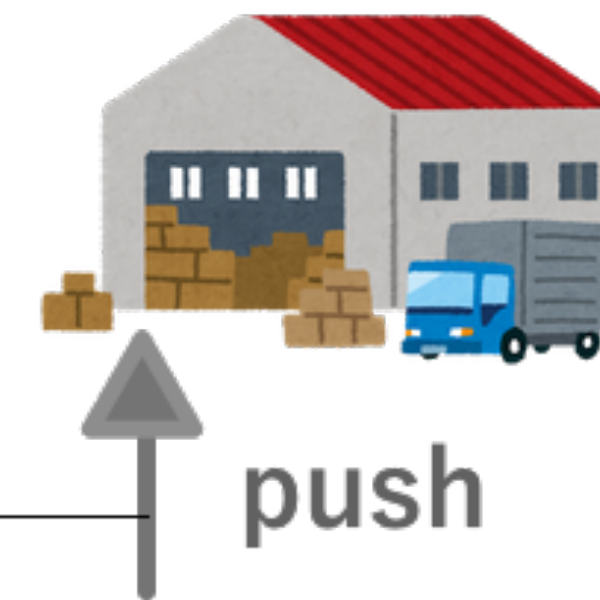
git push -u origin main

—set-upstreamと同じ

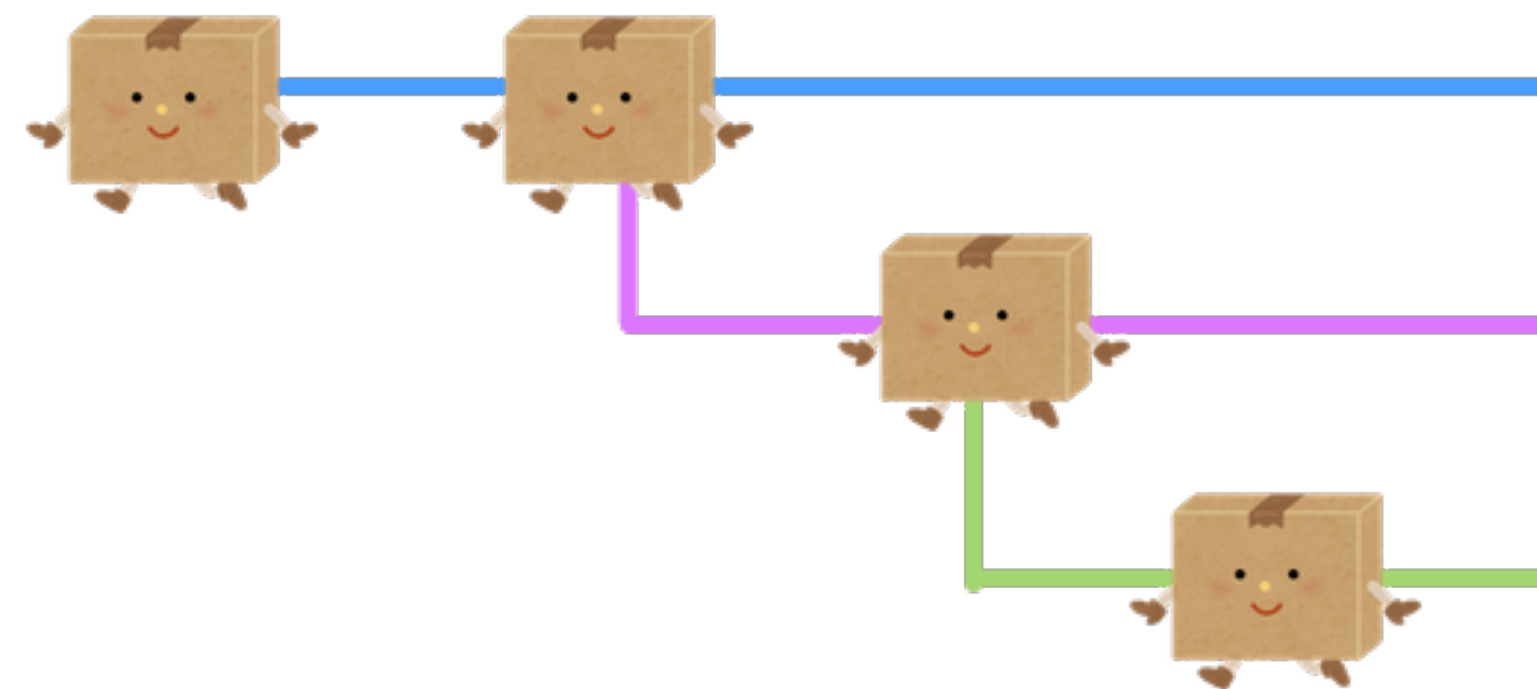
次からは git push だけでOKになる

ブランチ (枝)

リモート



ローカル



公開用/開発用/バグ修正用など
用途に合わせて枝分かれできる
(ワーキングツリー、先頭 HEAD)

ブランチ(枝)

git branch -a (一覧)

ブランチ作成

git branch ブランチ名

git checkout -b ブランチ名

git switch -c ブランチ名(git 2.23～) 作成&切替

ブランチ切替

git checkout ブランチ名

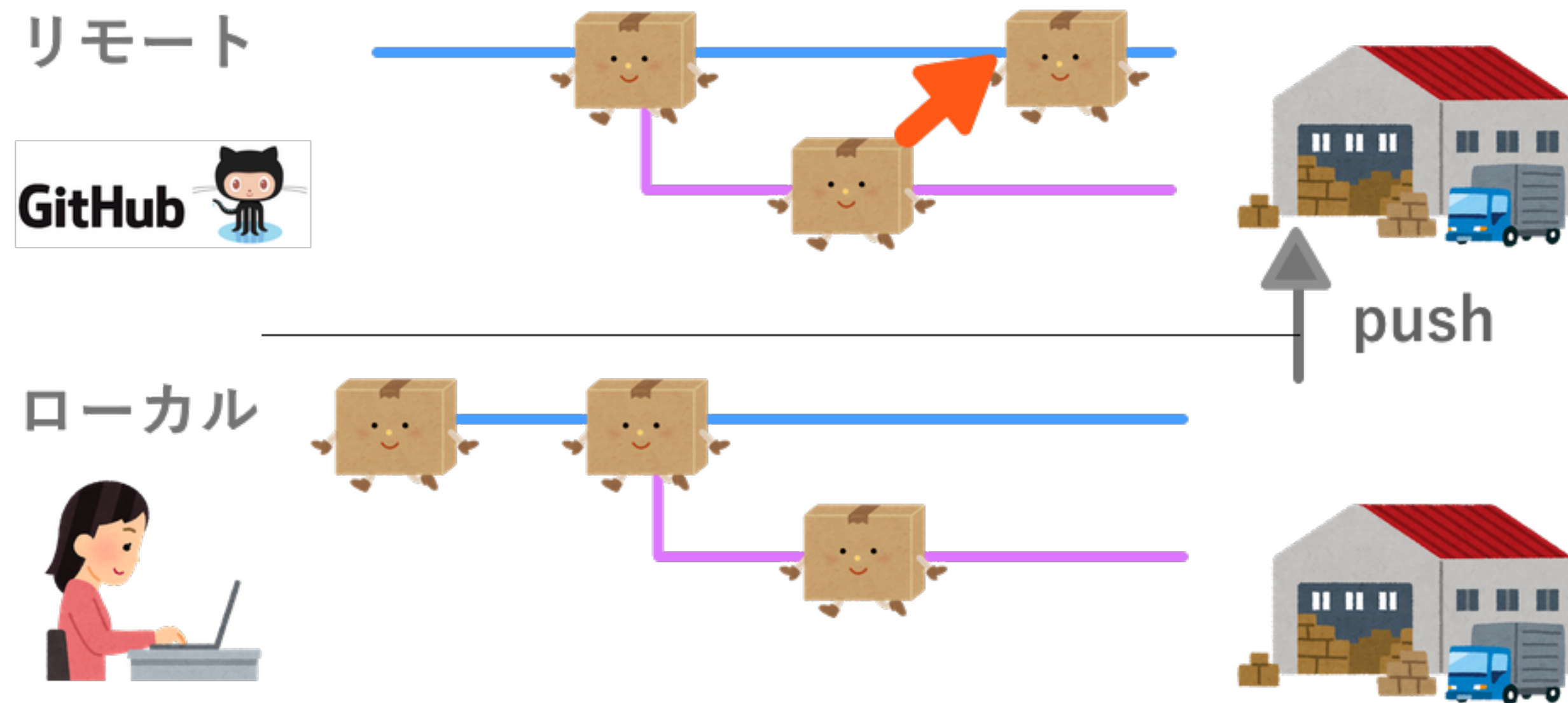
git switch ブランチ名(git 2.23～)

ブランチを切替後にpush

```
git add -A .  
git commit -m “コメント”  
git push origin ブランチ名
```

gitHub側で pull Request作成
コードをチェックして問題なければ
merge(マージ 融合)も可能

プルリクとマージ



gitHub側で pull Request作成 (コードの差分)
管理者がコードをチェックして問題なければ
merge(マージ 融合)も可能

その他の機能(抜粋)

status 前回コミットからの差分

diff インデックスとワーキングツリーの差異表示

merge ブランチを元のブランチに統合


reset コミットを取り消す

rebase ブランチを元のブランチにくっつける

stash コミットしていない内容を一時退避

pull request 変更をリポジトリに取り込んでもらうよう要求(GitHubの機能)

<https://qiita.com/2m1tsu3/items/6d49374230afab251337>



講座コードの ダウンロード& インストール方法

講座コードのダウンロード方法

```
git clone https://github.com/aokitashipro/  
laravel\_umarche.git
```

ブランチ指定してダウンロードする場合

```
git clone -b ブランチ名 https://github.com/  
aokitashipro/laravel\_umarche.git
```

講座コードのインストール方法

```
cd laravel_umarche // フォルダ移動  
composer install // バック側ライブラリインストール  
npm install // フロント側ライブラリインストール  
npm run dev // フロント側コンパイル
```

.env.example を .envにコピー

DB情報を変更

講座コードのインストール方法

.env

DB情報を変更（開発環境によって変更してください）

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=laravel_umarche

DB_USERNAME=umarche

DB_PASSWORD=password123

講座コードのインストール方法

.envを作成した後に

```
php artisan key:generate //ユニークキー生成
```

```
php artisan migrate:fresh -seed //DBテーブル・ダミー生成
```

```
php artisan serve // 簡易サーバー立ち上げ
```

講座コードのインストール方法

画像もダミー設定するなら
`php artisan storage:link`

`public/images/sample1.jpg~sample6.jpg` をコピー

Storage/app/public配下に
shops フォルダ products フォルダを生成し
それぞれのフォルダに画像をペースト