

Laravel

Laravel で開発をするためには、開発環境を用意する必要がある。一人で開発するときだけでなく、複数人で開発する際には「同じ開発環境」を揃えることが大切である。

更に、フレームワークやライブラリを多用する場合、個々の PC 環境によって細かいバージョンの差異や動作具合に影響が生じることがある。

主な Laravel の開発環境には以下のようなものがある。

- ローカル環境（XAMPP など）
- ローカル上の仮想マシン（vagrant など）
- 仮想コンテナ（Docker など）
- クラウド上の仮想マシン（AWS Cloud9, Paiza Cloud など）

今回は仮想コンテナを用いて開発を行う。

【Mac の人】

下記の順に進める。

1. 「【Mac 編】環境構築」
2. 「Laravel プロジェクト作成」

【Windows の人】

下記の順に進める。

1. 「【Windows 編】環境構築」
2. 「Laravel プロジェクト作成」

【Mac 編】環境構築

Docker を使用するための準備を進めていく。

必要な OS

- 最新の MacOS (2023 年 08 月時点では Ventura)

必要なもの

- Docker Desktop
- ターミナル (Mac に入っているもので OK)

Docker Desktop のダウンロードとインストール

下記の URL にアクセスし、必要なファイルをダウンロードする。

Intel CPU と M1 CPU で異なるので、ダウンロード時は注意すること。

<https://hub.docker.com/editions/community/docker-ce-desktop-mac/>

1. ダウンロードした `Docker.dmg` をダブルクリックしてインストーラを起動したら、アプリケーション・フォルダに Docker アイコンをドラッグする。
2. アプリケーション・フォルダ内にある `Docker.app` をダブルクリックし、Docker を起動する。
3. トップステータスバーにある Docker メニュー (クジラのマーク) で、Docker Desktop が実行中で、ターミナルからアクセスできる状態になる。

Docker の動作確認

ターミナルを開いて下記のコマンドを実行し、`docker` のバージョンが表示されれば準備完了 (バージョンなどの数値は資料より新しくても問題ない)。

```
$ docker -v
Docker version 20.10.8, build 3967b7d
```

バージョンが表示されたら、「Laravel プロジェクト作成」項に進もう！

【Windows 編】環境構築

本講座では Docker を使用して開発を行うため、準備を進めていく。

必要な OS

- Windows10 最新版もしくは Windows11 の最新版
- 上記より古い場合は進める前に OS アップデートしておこう。

必要なもの

- Docker Desktop
- Windows ターミナル
- Ubuntu 20.04
- WSL2
- PowerShell (Windows にもともと入っている)

WSL の準備

1. Powershell を管理者権限で実行する。
2. 下記のコマンドを実行する。完了には時間がかかる場合もあるので気長に待つ。

```
wsl --install
```

実行結果

```
PS C:\WINDOWS\system32> wsl --install
```

インストール中: 仮想マシン プラットフォーム

仮想マシン プラットフォーム はインストールされました。

インストール中: Linux 用 Windows サブシステム

Linux 用 Windows サブシステム はインストールされました。

インストール中: Ubuntu

Ubuntu はインストールされました。

要求された操作は正常に終了しました。変更を有効にするには、システムを再起動する必要があります。

3. PC を再起動する。

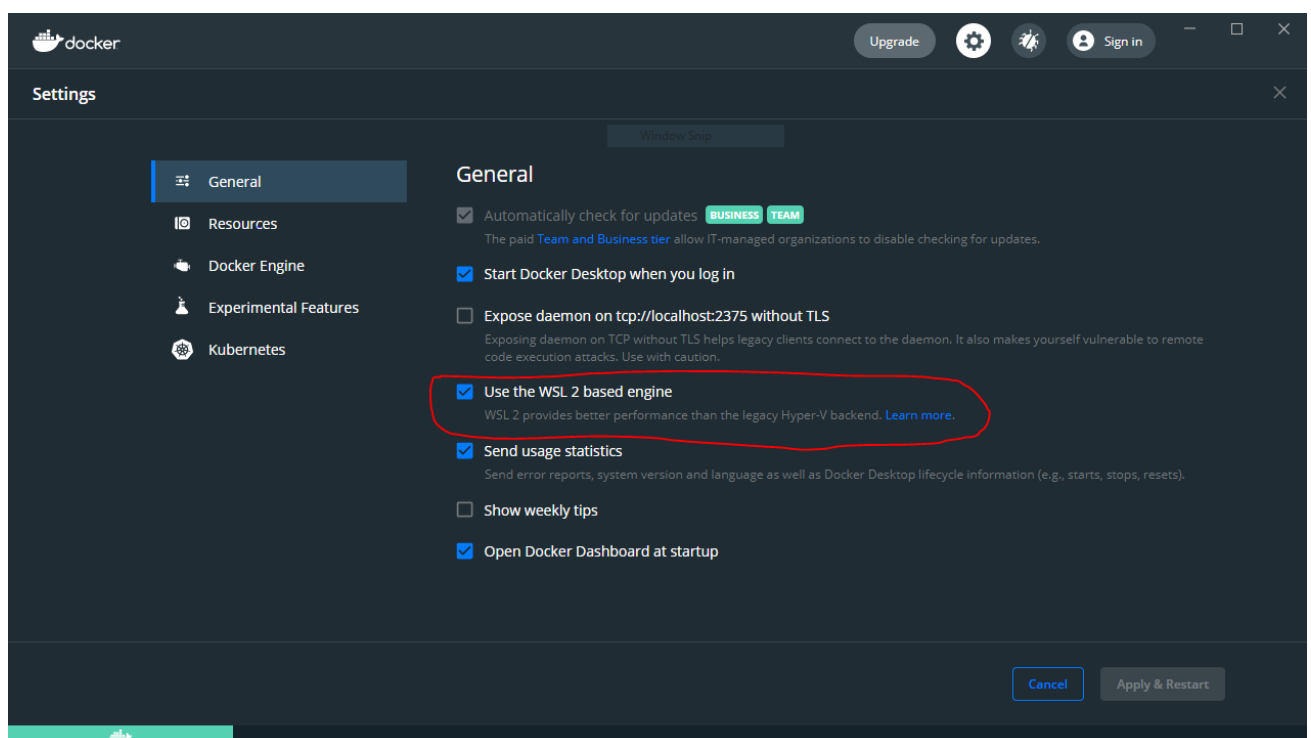
- 再起動すると ubuntu のターミナルが自動的に立ち上がるのでユーザ名とパスワードを設定する。
 - パスワードは画面に表示されないので注意.
 - 任意の値で構わない
 - ユーザ名とパスワードは後ほど使用するので必ず控えておくこと.
- Microsoft store で 「Windows ターミナル」 をインストールする.

Docker Desktop のインストール

- 下記の URL にアクセスし, ダウンロード → インストールする.
 - <https://www.docker.com/products/docker-desktop/>
- 完了したら画面の指示にしたがって再起動する.

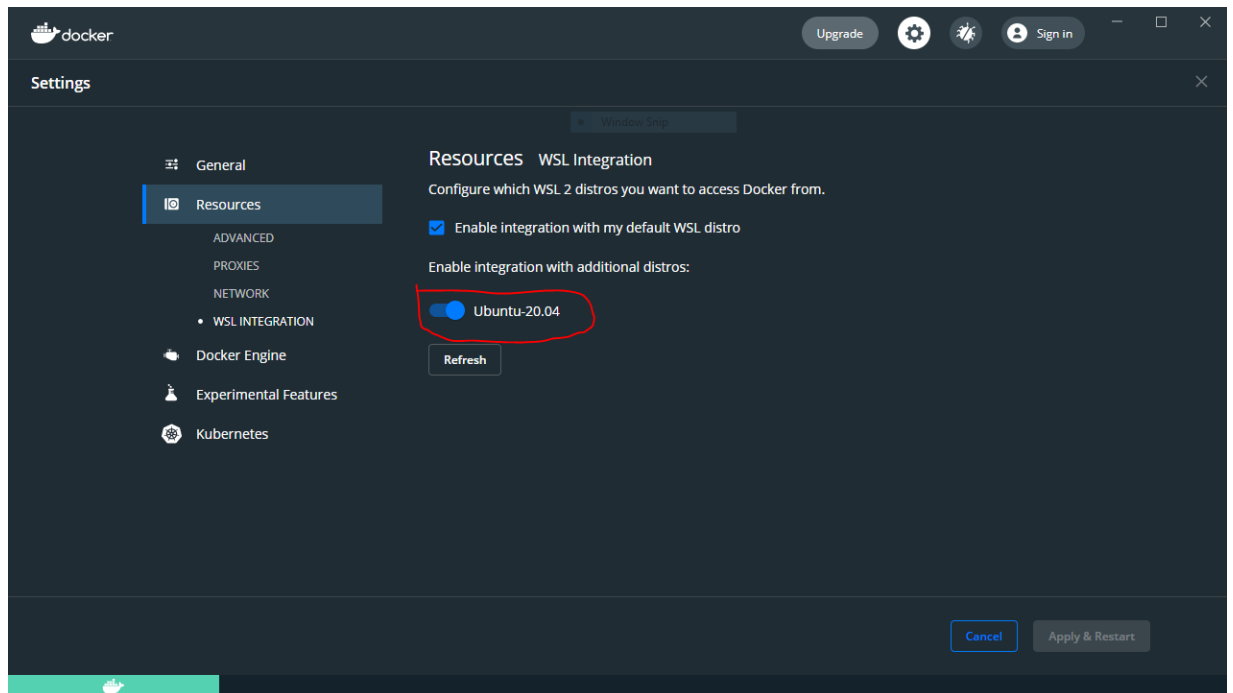
Docker の動作確認

- Docker Desktop を起動する. アカountのログインはしなくて OK.
- 「Setting (画面上部の ⚙️ アイコン)」 -> 「General」 -> 「Use the WSL2 based engine」 にチェックを入れる. すでにチェックが入っていてユーザが操作できない状態となっている場合があるが, その場合はそのまま OK.



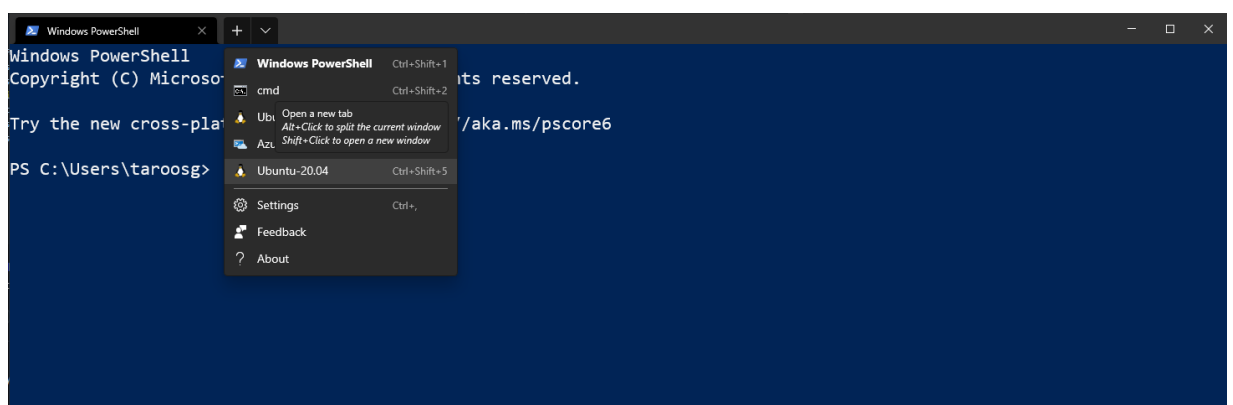
3. 「Setting (画面上部の ⚙️ アイコン)」 -> 「Resources」 -> 「WSL INTEGRATION」の Enable Integration ... にチェックを入れ, 「Ubuntu」のトグルをオンにする (下記画像では Ubuntu 20.04 だが, Ubuntu になっているはず). オンにしたら「Apply & Restart」をクリックする.

- トグルが出てこない場合は PC を再起動する.

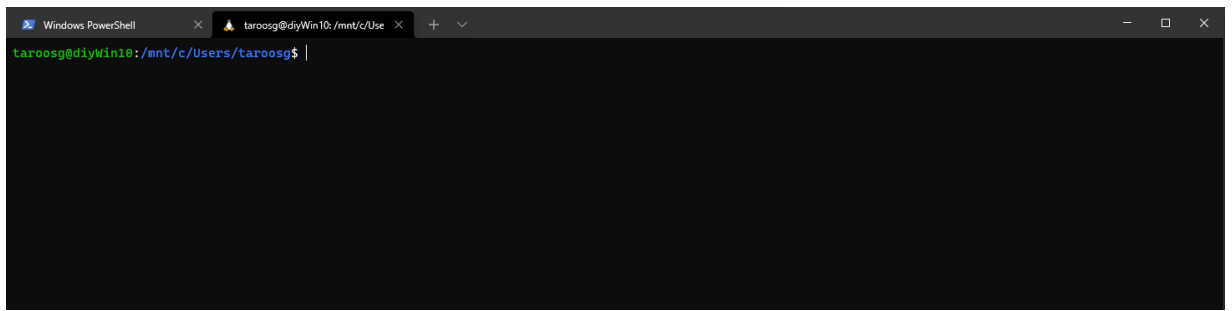


4. Windows ターミナルを Ubuntu で動かす.

- ↓ Windows ターミナルを開いて上部の「v」をクリックするとメニューが出るので, ここで Ubuntu を選ぶ. 出てこない場合は PC を再起動する (画像では Ubuntu 20.04 となっているが Ubuntu 表記になっているはず).



- ↓ このようになっていればコマンドが実行できる. 次項以降もこの状態でコマンドを実行していく.



5. 下記のコマンドを実行し、 `docker` のバージョンが表示されれば準備完了（バージョンなどの数値は資料より新しくても問題ない）.

```
$ docker -v  
Docker version 20.10.8, build 3967b7d
```

バージョンが表示されたら、「Laravel プロジェクト作成」項に進もう！

Laravel プロジェクト作成

ここでやりたいこと

- Laravel Sail を使用して、Laravel 開発環境をセットアップする.
- phpMyAdmin を使用して、Laravel アプリケーションのデータベースを管理できるようにする.

Laravel Sail でのプロジェクトセットアップ

このセクションでは、Laravel Sail を使用して「Laratter」という名前の新しい Laravel プロジェクトを作成し、開発環境をセットアップする。Laravel Sail は、Docker を使用した Laravel の開発環境を提供するツールで、開発の初期段階での環境構築を簡単にする。

Laravel Sail のインストール

Laravel Sail をインストールするには、以下のコマンドを実行する。このコマンドは、新しい Laravel プロジェクト「Laratter」を作成し、必要な Docker の設定を含む Laravel Sail をセットアップする。

コマンドを実行する際には Mac の場合はターミナル、Windows の場合は Windows ターミナルを使用する。ホームディレクトリに移動した後、以下のコマンドを実行する。

コマンドは任意のディレクトリで実施できるが、今回はできる限り他の影響を受けないようにホームディレクトリで実行する。

参考 URL

【補足】Laravel バージョンの指定

上記のコマンドでプロジェクトを作成した場合は最新バージョンが選択される。何らかの要因で最新でないバージョンを使用したい場合は [Tips](#) を参照。

phpMyAdmin の導入

Laratter アプリケーションのデータベースをブラウザ上で簡単に管理するために、phpMyAdmin をセットアップする。phpMyAdmin は、MySQL データベースを管理するためのフリーでオープンソースなツールである。

`docker-compose.yml` ファイルを開き、phpMyAdmin を追加する。追加場所は `services:` 直下。yml ファイルは**インデントが重要**なので注意すること。

```
# docker-compose.yml

services:
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    links:
      - mysql:mysql
    ports:
      - 8080:80
    environment:
      MYSQL_USERNAME: "${DB_USERNAME}"
      MYSQL_ROOT_PASSWORD: "${DB_PASSWORD}"
      PMA_HOST: mysql
    networks:
      - sail
```

記述した内容を保存したら以下のコマンドでコンテナを起動する。実行するときは、`laratter` ディレクトリで実行すること。

```
cd ~/laratter
./vendor/bin/sail up -d
```

Point

このコマンドはローカルサーバを立ち上げるコマンド。作業するときは必ず実行する。

注意

XAMPP などで開発を行っていた場合、同時に立ち上げるとポートの競合が発生して起動できない場合がある。XAMPP のサーバを停止してから上記コマンドを実行する。

同様に、コンテナを立ち上げていると XAMPP のサーバが立ち上がらない場合がある。XAMPP を使用したい場合は後述の「コンテナを終了するとき」のコマンドでコンテナを終了してから XAMPP を立ち上げる。

ユーザテーブルのマイグレーション

プロジェクトを作成すると、ユーザ認証に必要なユーザテーブルのマイグレーションファイルが自動的に生成される。これらのマイグレーションを実行して、データベースにユーザテーブルを作成する。

```
./vendor/bin/sail php artisan migrate
```

Point

Laravel では、テーブル（とカラム）の作成や変更はマイグレーションファイルを作成して行う。

例えば、テーブルを作成する場合の流れは以下の通りとなる。

1. コマンドでマイグレーションファイルを作成する。
2. マイグレーションファイルに、作成したいカラムの情報を記述する。
3. コマンドでマイグレーションを実行するとテーブルが作成される。

詳しい実行時の流れは [Tips](#) を参照。

動作確認

下記 2 点が確認できたら Laravel Sail を使用した開発環境の基本的なセットアップが完了である。

Laravel アプリケーション

ブラウザで `http://localhost` にアクセスして、Laravel のデフォルトページが表示されることを確認する。



Documentation

Laravel has wonderful, thorough documentation covering every aspect of the framework. Whether you are new to the framework or have previous experience with Laravel, we recommend reading all of the documentation from beginning to end.



Laracasts

Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.



Laravel News

Laravel News is a community driven portal and newsletter aggregating all of the latest and most important news in the Laravel ecosystem, including new package releases and tutorials.



Vibrant Ecosystem

Laravel's robust library of first-party tools and libraries, such as [Forge](#), [Vapor](#), [Nova](#), and [Envoyer](#) help you take your projects to the next level. Pair them with powerful open source libraries like [Cashier](#), [Dusk](#), [Echo](#), [Horizon](#), [Sanctum](#), [Telescope](#), and more.

[Shop](#) [Sponsor](#)

Laravel v8.20.1 (PHP v8.0.0)

phpmyadmin

ブラウザで `http://localhost:8080` にアクセスして、phpMyAdmin のログインページが表示されることを確認する。下記の情報でログインできることを確認しておく。

username: sail
password: password

ログインしたら、`laravel` データベース内に `users` テーブルが作成されていることを確認しておく。テーブル内にはまだデータが入っていない状態となる。

コンテナを終了するとき

コンテナを終了するときは下記コマンドを実行する。動作確認ができたなら終了させておく。

```
# 必ず laratter ディレクトリで実行する
cd ~/laratter
./vendor/bin/sail down
```

Point

このコマンドはローカルサーバを終了するコマンド。作業終了時に実行する。

Laravel Sail コンテナ内でのコマンド実行

Laravel のコマンドは `php artisan ...` のような形のものが多いが、Laravel Sail コンテナ内でコマンドを実行するときは先頭に `./vendor/bin/sail` をつける必要がある。下記では `php artisan migrate` を実行する例を示す。

```
./vendor/bin/sail php artisan migrate
```