

Laravel 資料1 環境構築とページ作成

0. はじめに

本資料は以下の内容について書かれている。

1. Composer の導入
2. Laravel プロジェクトの作成
3. login ページの作成

■ 注意

XAMPP は **PHP のバージョン** が **7.x 系列** の物を使用すること。

また、XAMPP のパスは著者の環境に依存している為

それぞれの環境に合わせて修正すること。

1. Composer のインストール

[composer公式](#) 

Getting Started から飛ぶ



A Dependency Manager for PHP

Latest: **1.9.0**

[Getting Started](#)

[Download](#)

[Documentation](#)

[Browse Packages](#)

[Issues](#)

[GitHub](#)

[Windows用インストーラリンク](#) 

Composer-Setup.exe をクリックし、インストーラをダウンロードし、実行する。

Installation - Windows

Using the Installer

This is the easiest way to get Composer set up on your machine.

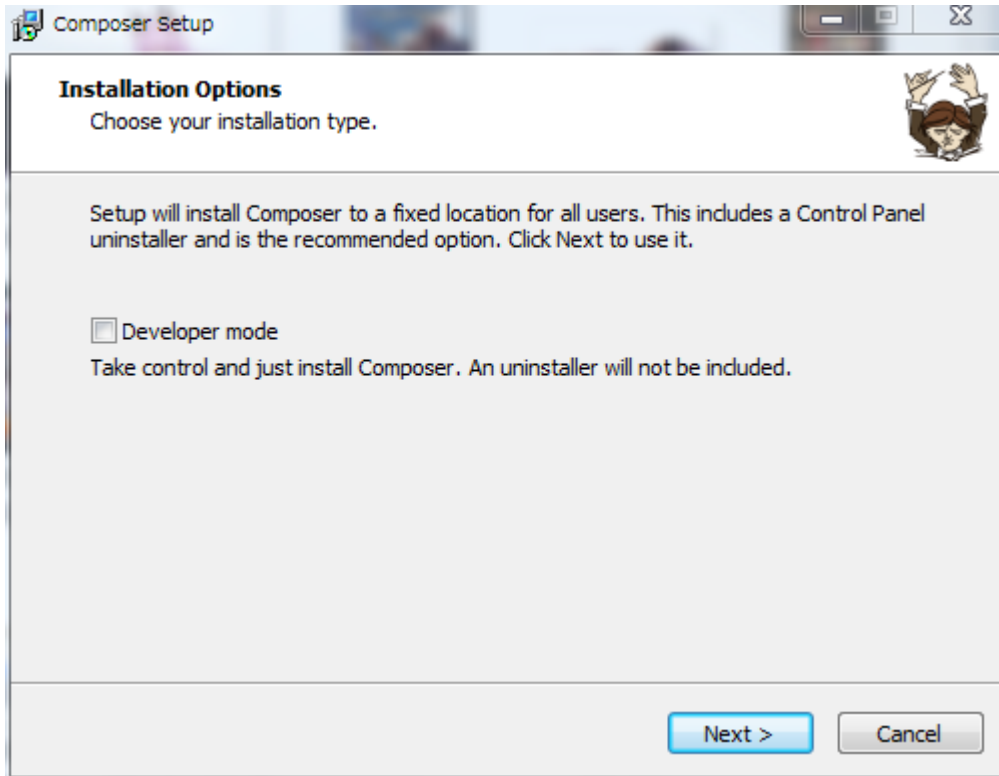
Download and run [Composer-Setup.exe](#). It will install the latest Composer version and set up your PATH so that you can call `composer` from any directory in your command line.

Note: Close your current terminal. Test usage with a new terminal: This is important since the PATH only gets loaded when the terminal starts.

■ ステップ 1

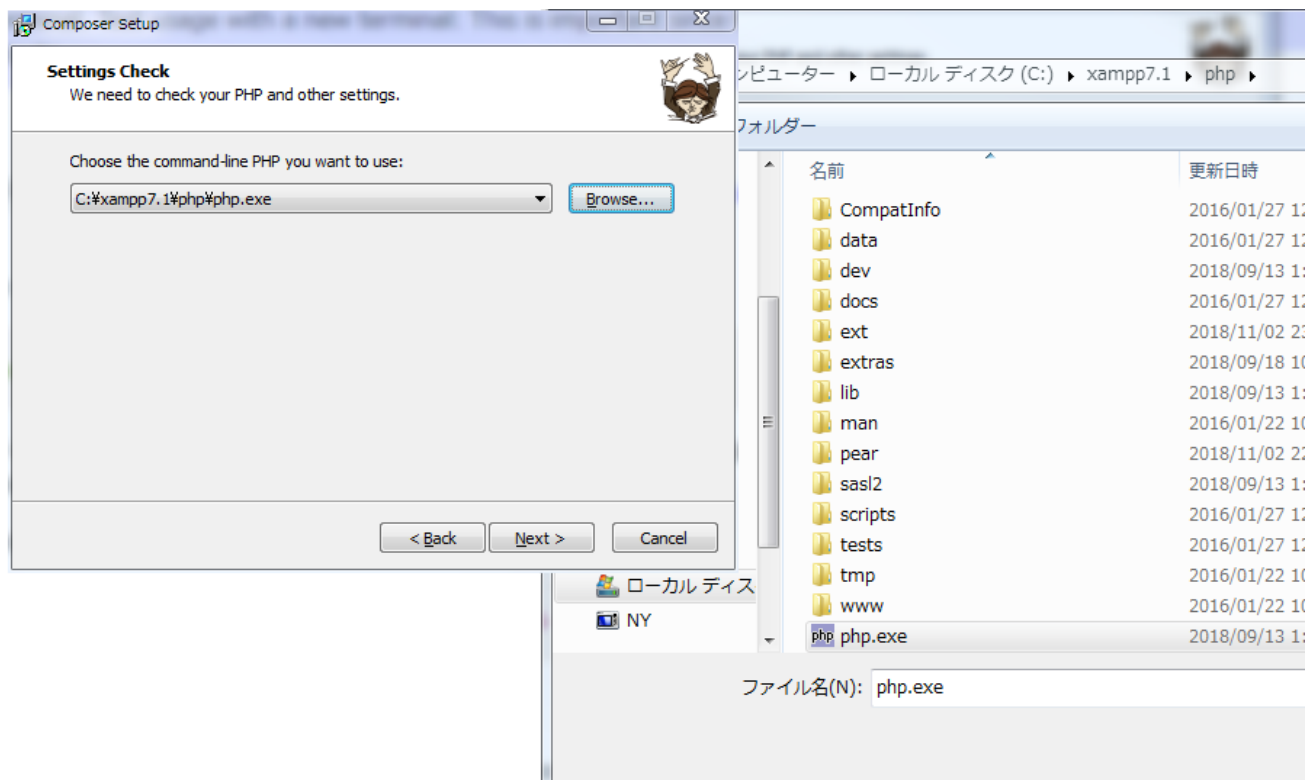
チェックはせず、次へ。

※チェックするとインストール時に「アンインストール用ファイル」が含まれなくなる



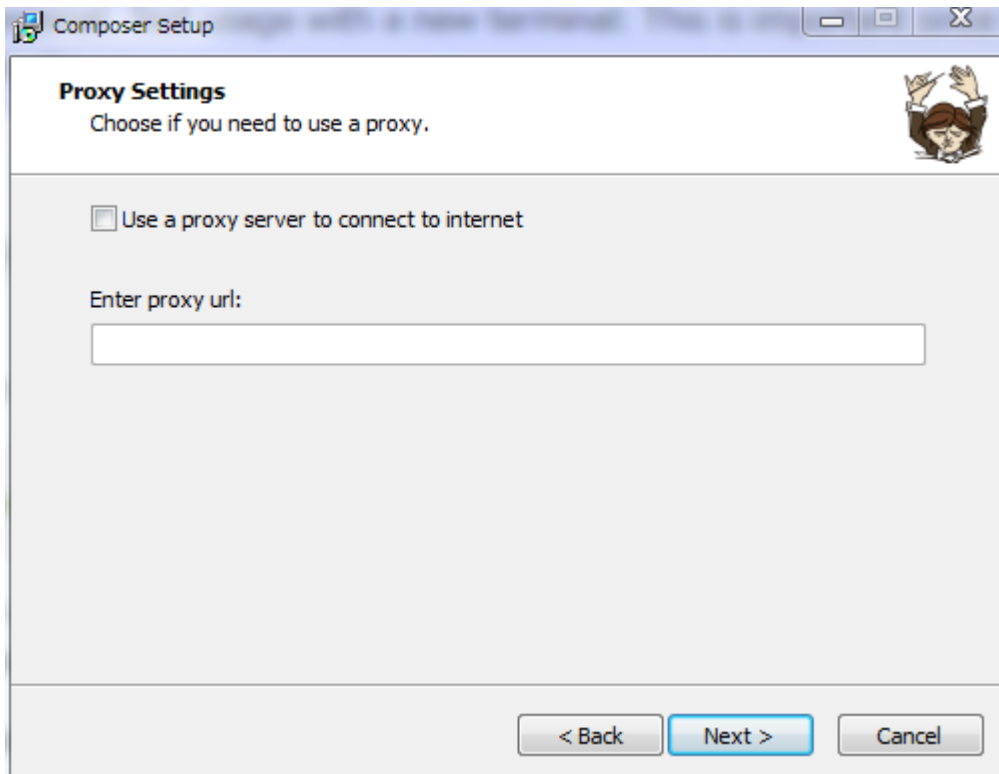
■ ステップ 2

今回使う予定の XAMPP 内の php.exe を選択し、次へ。



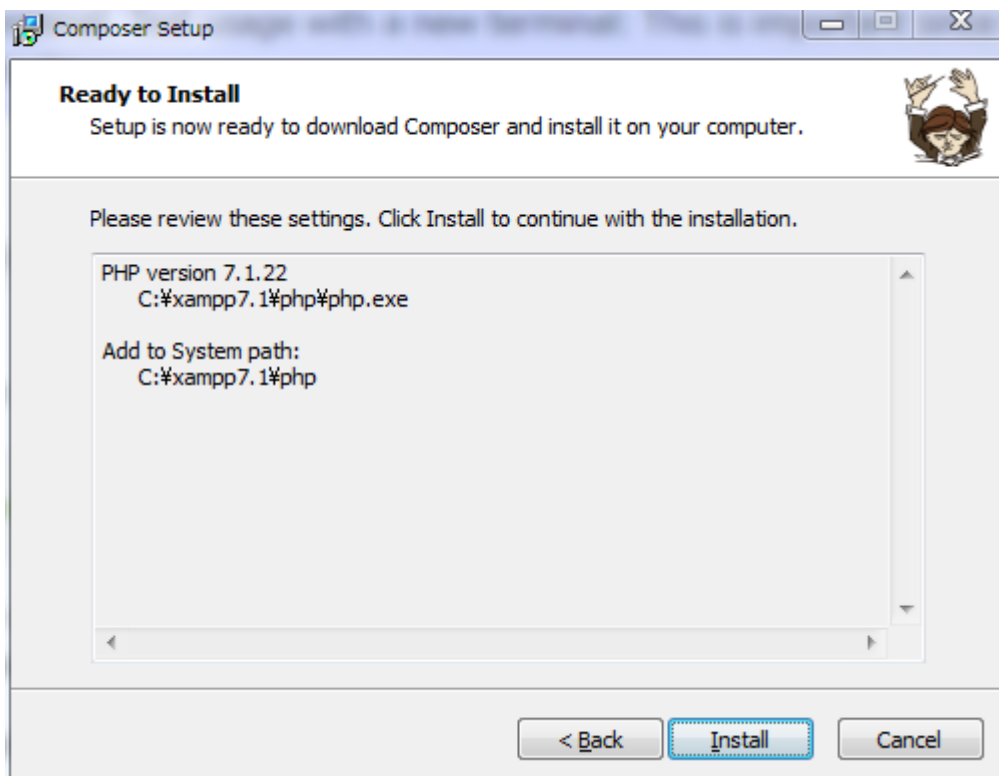
■ ステップ 3

プロキシ設定。無視して次へ。



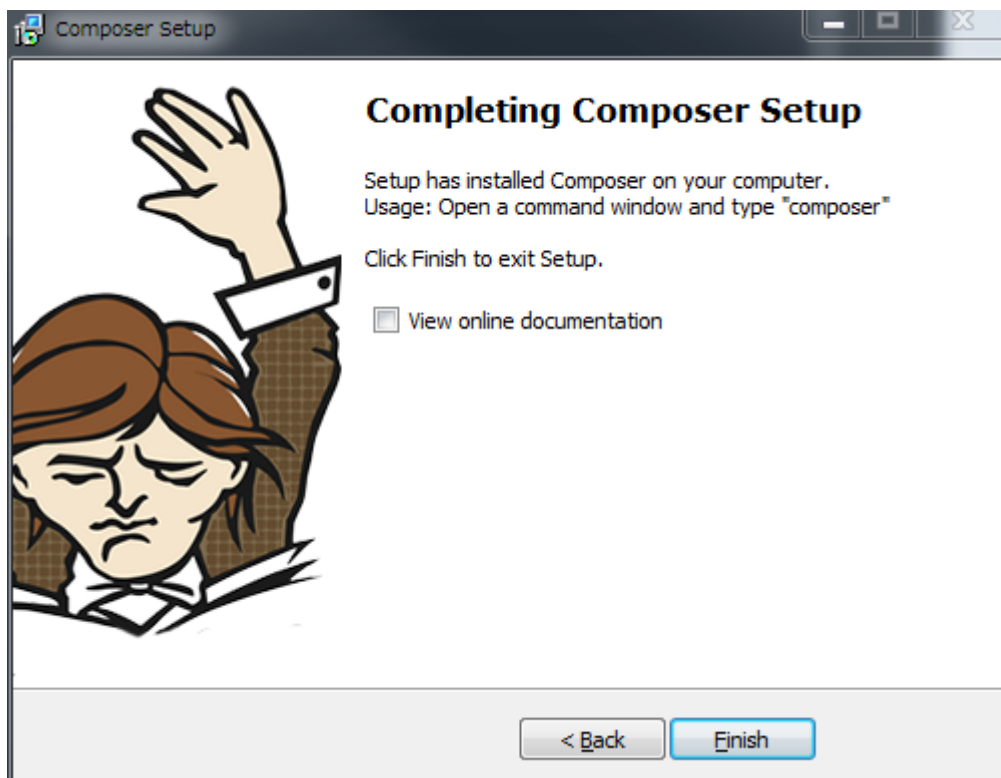
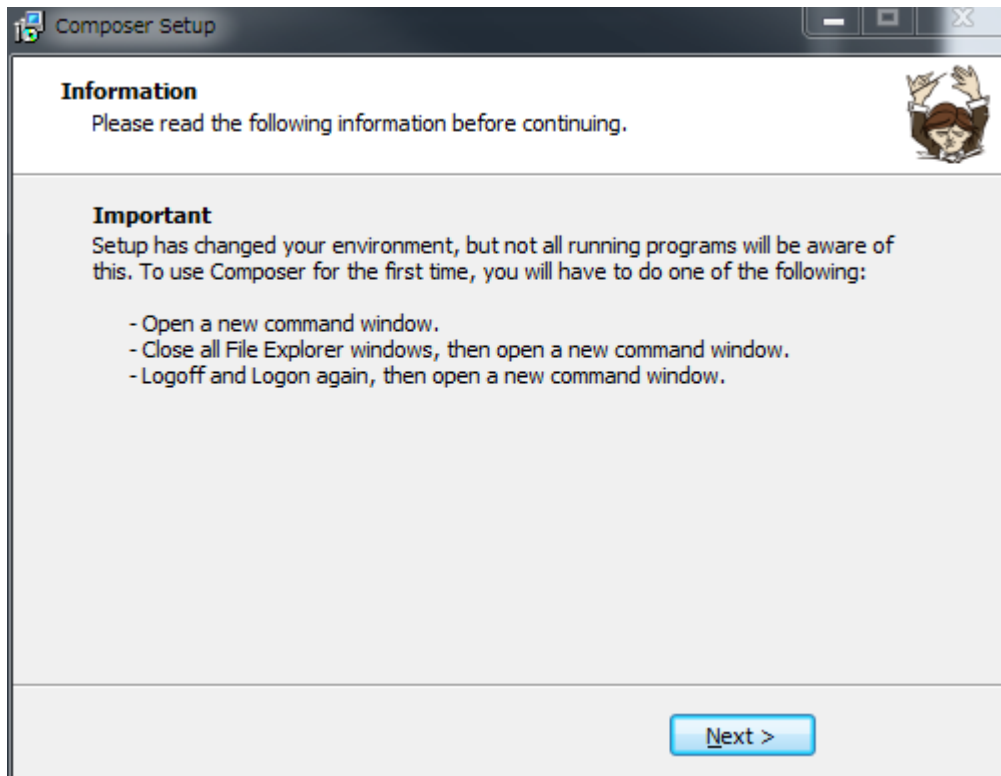
■ ステップ 4

「Install」からインストールを開始する。



■ ステップ 5

インストール完了。

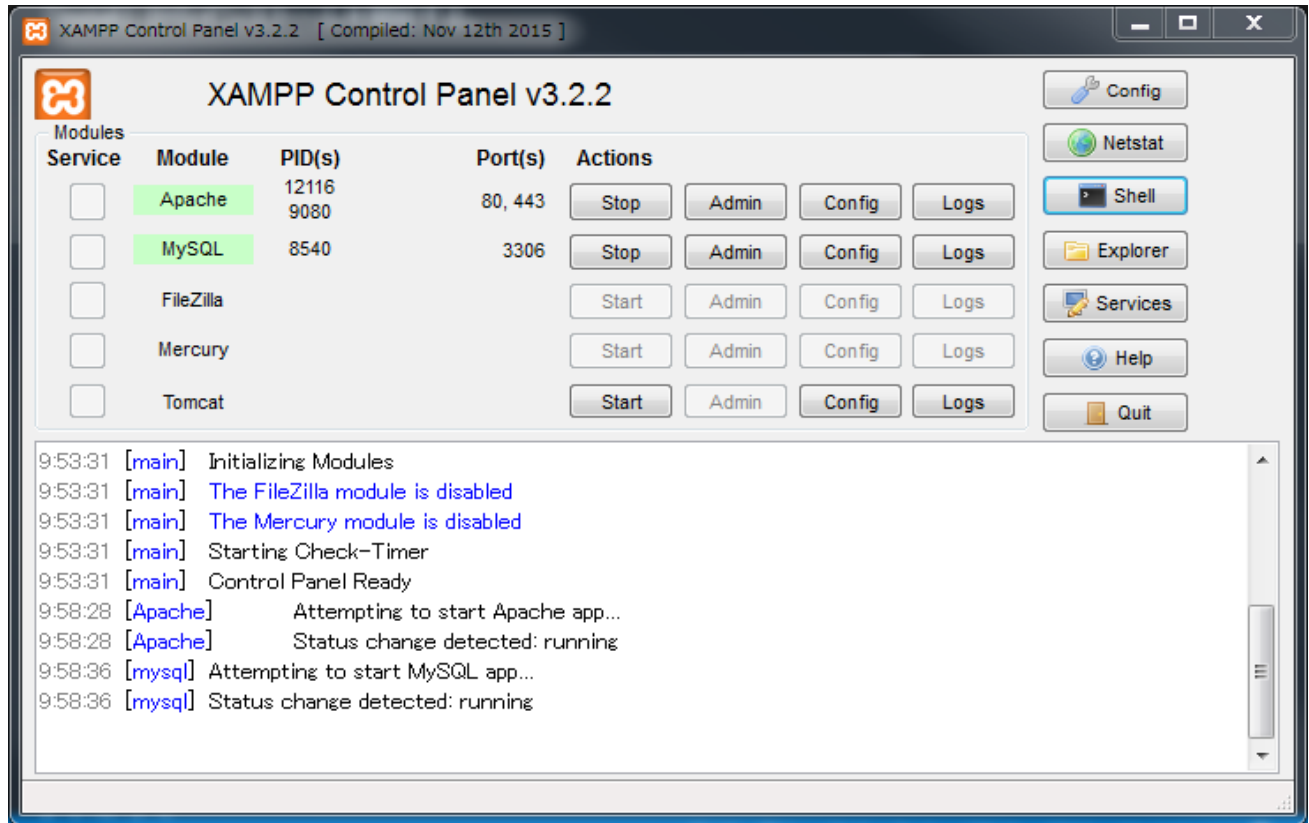


Laravel プロジェクトの作成

XAMPP の `htdocs` 内に Laravel のプロジェクトを作成する。

■ shell の起動

XAMPP を起動し、図のボタンから shell を起動する。



■ プロジェクトの作成

起動後は `C:\xampp` から始まるため

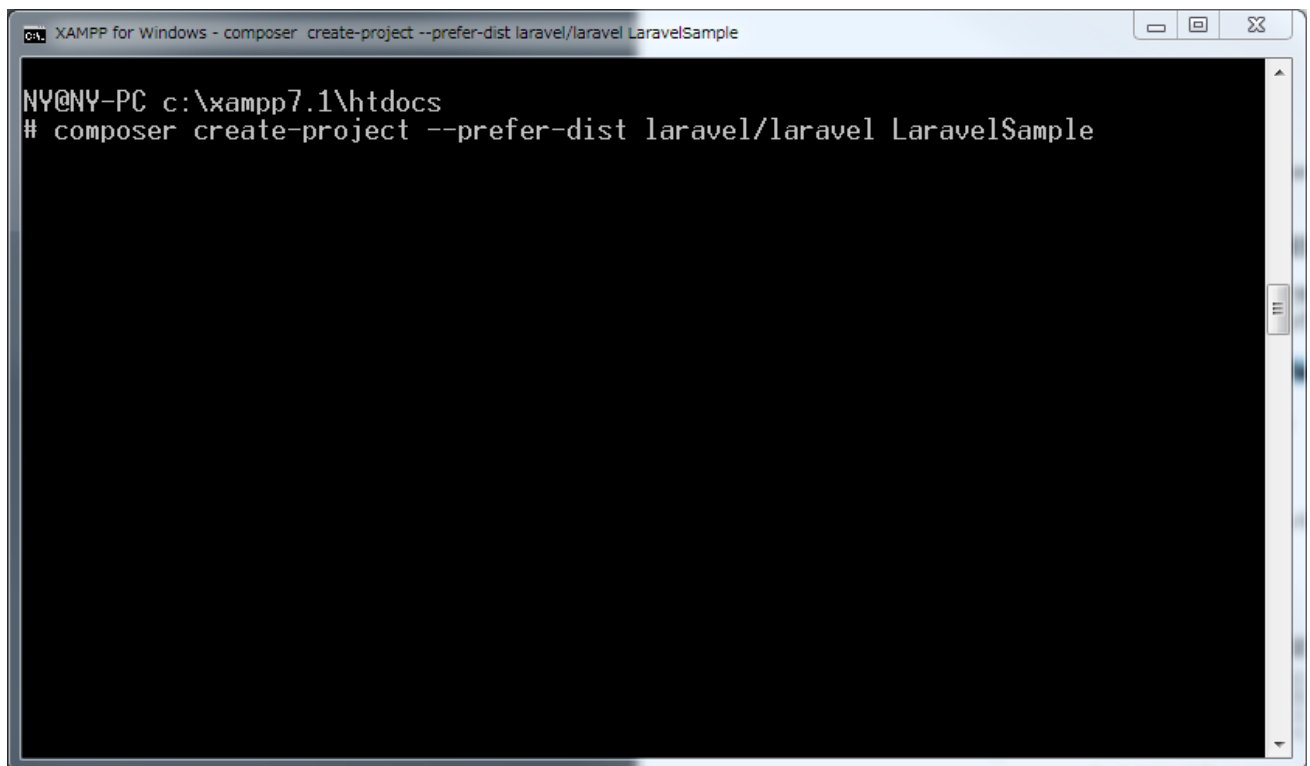
以下のコマンドで `C:\xampp\htdocs` フォルダへ移動。

```
cd htdocs
```

移動を確認後、以下のコマンドを実行する。

```
composer create-project "laravel/laravel=5.4.*" LaravelSample
```

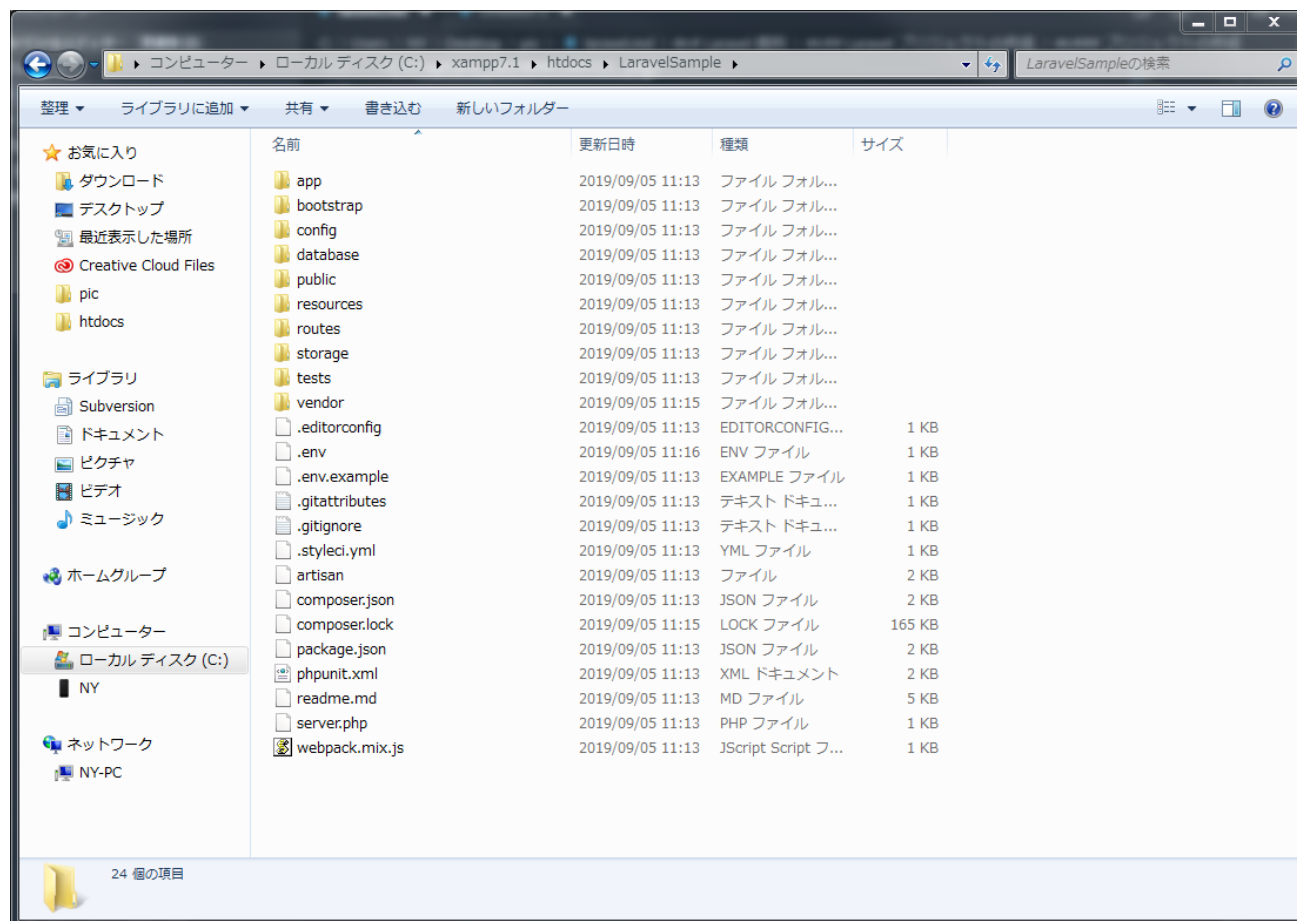
※今回使用する Laravel のバージョンは 5.4 とする



```
NY@NY-PC c:\xampp7.1\htdocs
# composer create-project --prefer-dist laravel/laravel LaravelSample
```

`Application key set successfully.` と出たら完了。

XAMPP の htdocs をエクスプローラから確認すると、フォルダが作成されている。



■ サーバーの起動

shell で以下のコマンドを実行し、プロジェクトフォルダへ移動する。

```
cd c:\xampp\htdocs\LaravelSample
```

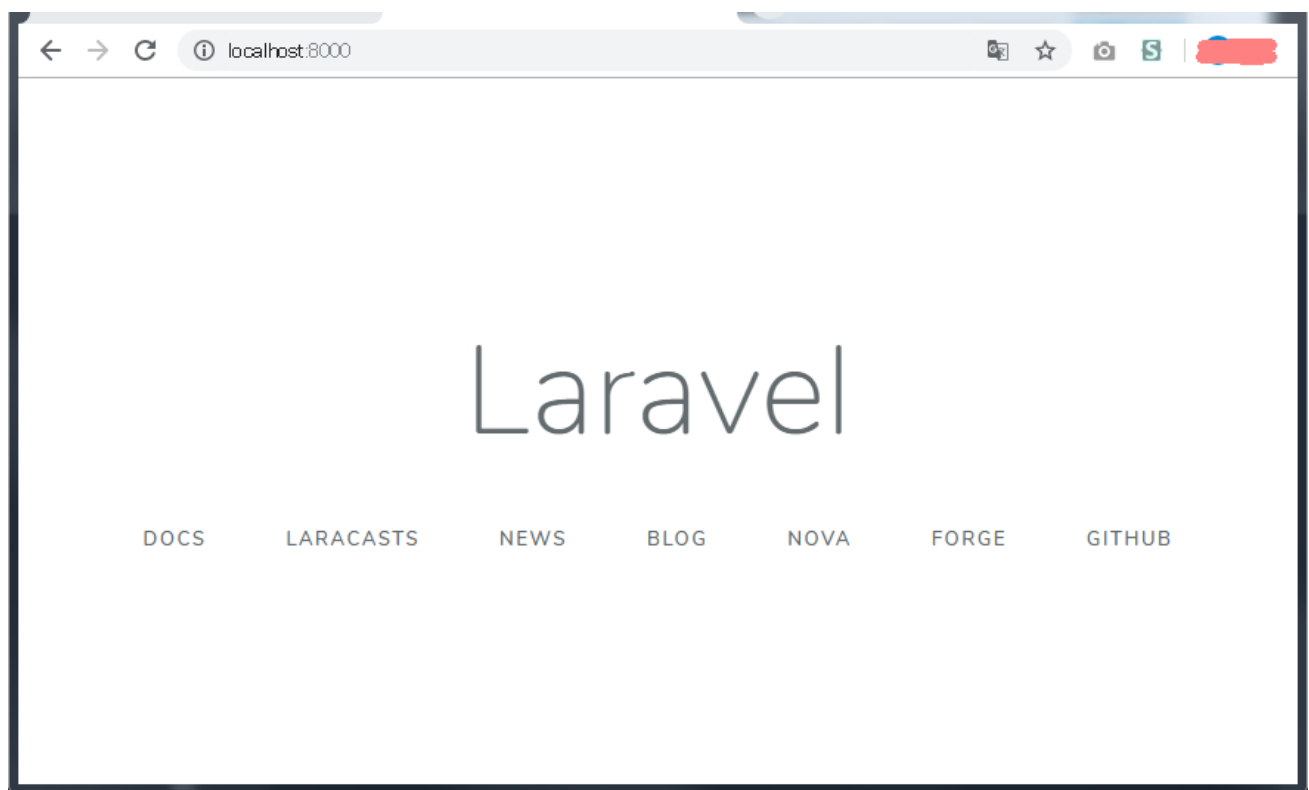
移動を確認後、以下のコマンドを実行しサーバーを起動する。

```
php artisan serve
```

shell 上に `Laravel development server started: <http://127.0.0.1:8000>` と表示された後

ブラウザにて <http://localhost:8000> を開き

以下の画面が表示されたらOK。



■ サーバーの停止

サーバーを止める際には shell を閉じればOK。

ログイン画面の作成

ブラウザ上で `localhost:8000/login` と叩いた際にログイン画面が出るよう

以下の流れでログイン画面のトップページを表示させる。

1. ルーティングに関わる `web.php` を編集する
2. 項目 1 にて「`web.php` にて呼んだコントローラー名」でコントローラーを作成
3. 画面表示用に View ファイル (拡張子 `.blade.php`) を作成

新たなページを作る際には、上記の流れを再度行うことになる。

ルーティングの編集

`routes/web.php` を編集する。

```
Route::get('/login', 'LoginController@getIndex');
```

本ルーティングファイルは「どのメソッドで」「どのURL (URI) が叩かれたら」

「どのコントローラの」「どの関数を呼ぶか」を指定する。

今回の場合は「GET メソッドで」「/login にアクセスした場合」

「LoginController.php 内の」「getIndex() 関数を呼ぶ」という意味になる。

もちろん「呼ぼうとしているファイルが存在しない」とエラーを吐くため

これから順番に作成していくことになる。

ちなみに URL から直接ページを開く際には `Route::get()` を使い

フォームの値の受け渡しの際には `Route::post()` を使う。

コントローラーの生成

本項目ではコマンドライン上から「LoginController.php」ファイルを生成する。

■ プロジェクトフォルダへ移動

shell を起動し `cd c:\xampp\htdocs\LaravelSample` を実行し

プロジェクトフォルダへ移動する。

■ LoginController の生成

移動後、以下のコマンドを実行。

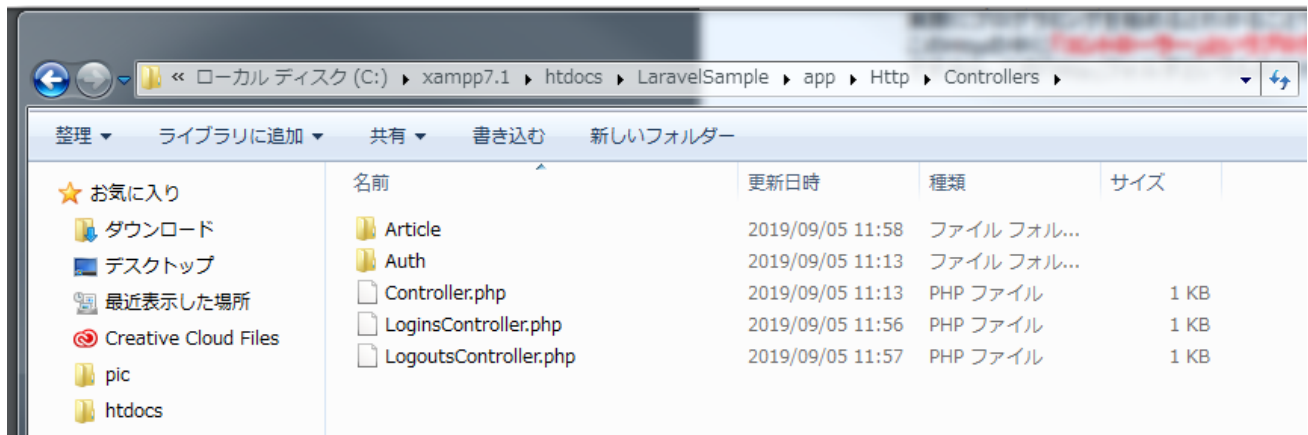
```
php artisan make:controller LoginController
```

実行後、`Controller created successfully.` と出たらOK.

■ ファイルの確認

C:\xampp\htdocs\LaravelSample\app\Http\Controllers を開くと

以下の通り、コントローラーファイルが生成されている。



■ LoginController の編集

ファイルの作成に成功したら、 web.php にて記述した関数

`getIndex()` を作成する。

【app¥Http¥Controllers¥LoginController.php】

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class LoginController extends Controller
{
    function getIndex() {
        // view ファイルを返却
        return view('login/login');
    }
}
```

テンプレートファイルの作成

[公式リンク](#)

Blade (ブレード) という Laravel 固有のテンプレートエンジンを利用し

拡張子 `.blade.php` ファイルを動かす。内容は html + php。

MVC の V ... View にあたる。

■ layout.blade.php の作成

以下の通り、テンプレートとなる view ファイルを作成する。

配置場所は `LaravelSample/resources/views/layout` の中。

※layout フォルダは自分で作成すること。

【layout.blade.php】

```
<!doctype html>
<html lang="ja">
<head>
<meta charset="utf-8">
<link rel="stylesheet" href="{{ asset('/css/bootstrap.min.css') }}">
<link rel="stylesheet" href="{{ asset('/css/app.css') }}">
<!-- js -->
<script src="https://ajax.googleapis.com/ajax/libs/
jquery/1.11.3/jquery.min.js"></script>
<script src="{{ asset('/js/bootstrap.min.js') }}">
</script>
<script>
{{--@yield('script')--}}
</script>
</head>
<body>
  
  <div class="container">
    @yield('content')
  </div>
</body>
</html>
```


■ login.blade.php の作成

配置場所は `LaravelSample/resources/views/login` の中。

※login フォルダは自分で作成すること。

【login.blade.php】

```
@extends('layout/layout')
@section('content')
<!-- form -->
<form method="post" action="/login">
{{ csrf_field() }}
<!-- input type="text" string -->
<div class="form-group">
    <label>名前</label>
    <input type="text" name="loginid" class="form-control">
</div>

<!-- input type="text" int -->
<div class="form-group" >
    <label>パスワード</label>
    <input type="password" name="loginpassword" class="form-control">
</div>

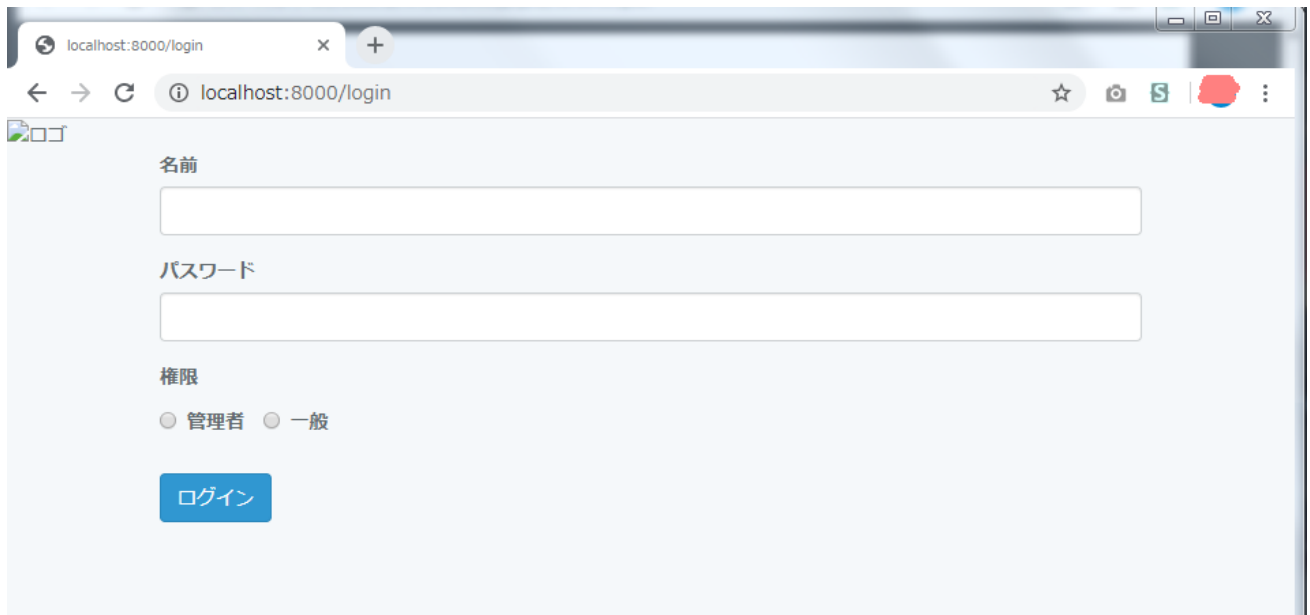
<!-- radio -->
<p><b>権限</b></p>
<div class="radio-inline">
    <label>
        <input type="radio" name="authority" value="1" >管理者
    </label>
</div>

<div class="radio-inline">
    <label>
        <input type="radio" name="authority" value="2">一般
    </label>
</div>

<br/><br/>
<input type="submit" value="ログイン" class="btn btn-primary">
</form>
@stop
```

画面を確認

少々画面がアレだが、表示には成功した。



The screenshot shows a web browser window with the address bar displaying 'localhost:8000/login'. The page content includes:

- A label '名前' (Name) above a text input field.
- A label 'パスワード' (Password) above a text input field.
- A label '権限' (Permissions) above two radio buttons: '管理者' (Administrator) and '一般' (General).
- A blue button labeled 'ログイン' (Login).