

# 使用 Controller

范聖佑 Shengyou Fan  
新北市樹林國小 (2015/07/09)

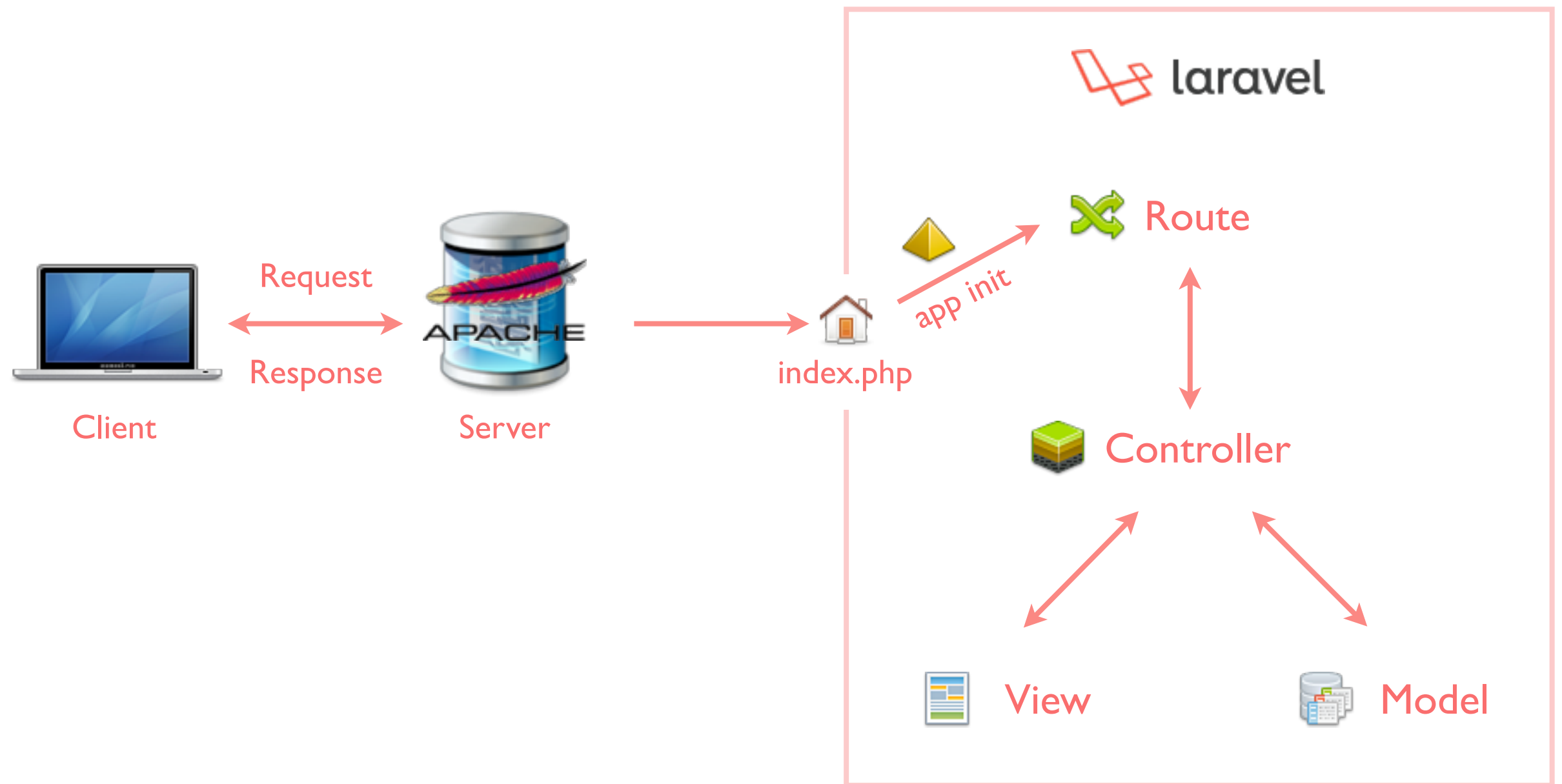
```
1  /**
2   * Display the specified resource.
3   *
4   * @param int $id
5   * @return Response
6   */
7  public function show($id)
8  {
9      $post = Post::with('comments')->where('id', $id)->first();
10
11      return View::make('post.show')->with('post', $post);
12  }
```

# 單元主題

- 什麼是 Controller ？
- 目前實作方式有什麼問題？Controller 可以幫助我們什麼？
- 如何透過 `artisan` 產生 Controller 類別？
- Controller 類別如何依照 `RESTful` 慣例實作各方法
- 示範將工作坊實作專案裡，原本寫在 `routes.php` 裡的動作搬至對應的 Controller 類別內

# Controller 簡介

# Laravel 的 MVC 分工



# 在有 Controller 之前...

- 截至目前的示範，我們把所有應用程式邏輯都寫在 `app/Http/routes.php` 裡，雖然可以正常運作，但會讓 `routes.php` 的內容一直長、一直長...
- 各頁面的程式邏輯沒有清楚的分類，若是其他人接手時，也很難找出整個應用程式的運行脈絡，這樣寫下去程式遲早難以維護...
- 總結來說，就是目前的範例只用到 M 及 V，該是 C 介入的時候了！

# 什麼是 MVC 的 Controller ？

- MVC 設計架構裡，**M** 代表 **Model** 負責處理資料；**V** 代表 **View** 負責顯示頁面；**C** 代表 **Controller** 負責控制 M 及 V 間的溝通並回傳結果
- 在此架構設計下，**Controller** 的功能就是負責控制 **M** 及 **V**，也就是接受來自 **Route** 的指令，將所需要顯示的資料和頁面整理好回傳。也可以思考成將動作分類的一種方式
- 如此即可讓**商業邏輯** (M)、**顯示** (V) 與**程式控制** (C) 三者獨立開來，彼此分工、互相合作

# Laravel 的 Controller 機制

- 在 Laravel 的 `app/Http/routes.php` 裡，可以分別將不同的 Route 動作指定給不同的 Controller 來處理，而每一個 Controller 內也可以有不同的方法，處理來自不同 Route 傳來的請求
- 一般來說，Controller 內的動作也有慣例，即依照 **RESTful** 的動作名稱來做對應，讓不同的 Controller 及各 Controller 內的動作都有規則可循。除了可以依賴 **artisan** 產生程式碼外，也可提升開發速度

# 產生 Controller



# 產生 Controller 類別

- `artisan` 內建就有幫我們產生 Controller 的指令，直接呼叫 `make:controller` 即可
- 預設產生出來的 Controller 就支援 `RESTful` 的設計，若要產生出「空白 (plain)」的 Controller 則要額外下參數

```
$ [php] artisan make:controller \
    {name}
```

產生 controller 檔案

# artisan make:controller

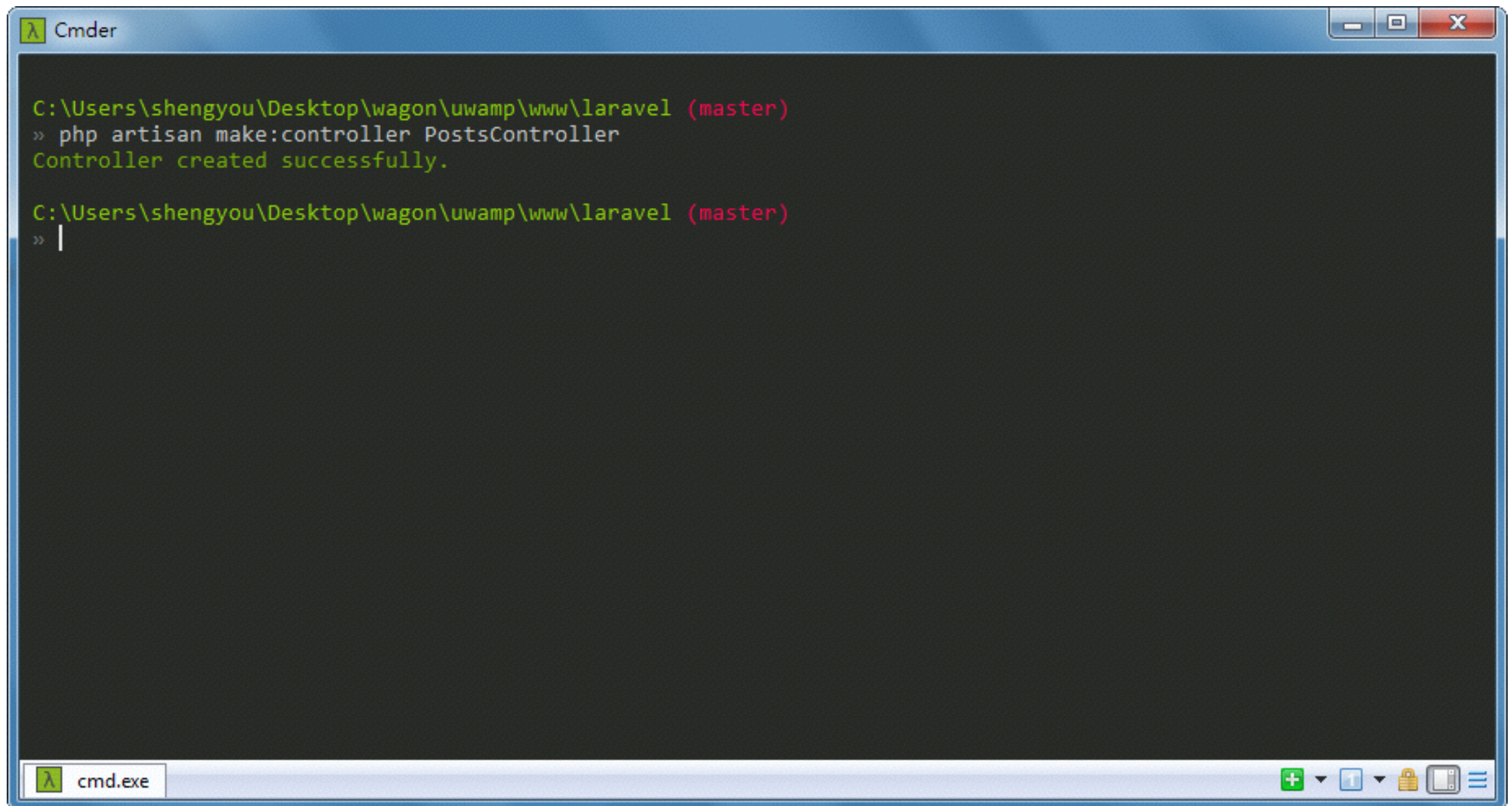
- 透過 `artisan` 產生 `Controller` 類別
  - `artisan` 會依照給予的名稱，產生類別檔案
  - `--plain` 預設指令會自動在 `Controller` 類別裡產生對應的 `RESTful` 的各種方法，若不希望自動產生的話，可加上這個參數
- 範例：

```
$ php artisan make:controller PostsController
```

```
$ php artisan make:controller PostsController --plain
```

# 產生 Controller 檔案

使用 artisan 產生 Controller 檔案



```
C:\Users\shengyou\Desktop\wagon\uwamp\www\laravel (master)
>> php artisan make:controller PostsController
Controller created successfully.

C:\Users\shengyou\Desktop\wagon\uwamp\www\laravel (master)
>> |
```

The screenshot shows a Windows Command Prompt window titled 'Cmder'. The command prompt is at the directory `C:\Users\shengyou\Desktop\wagon\uwamp\www\laravel` and is in the `(master)` branch. The user has entered the command `php artisan make:controller PostsController`, and the output is `Controller created successfully.`. The prompt is now waiting for the next command.

# 命名與指令慣例

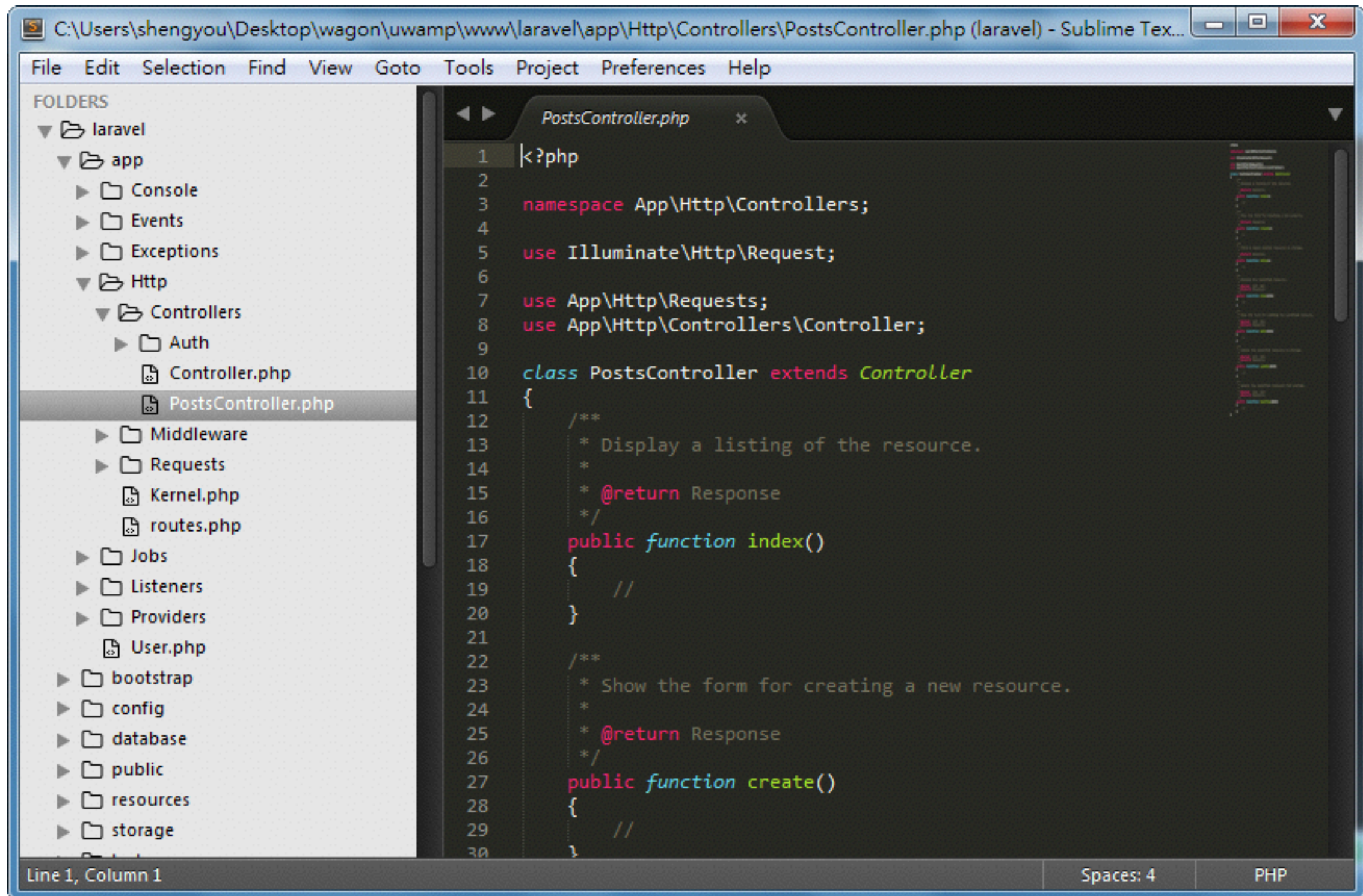
- 雖然 Laravel 沒有強制規定，但一般慣例會將 Controller 以 {大駝峰、資源複數名稱}Controller 來命名，如 PostsController
- 預設 Controller 檔案會放置在 `app/Http/Controllers/` 資料夾底下
- 假如要變更 Controller 檔案的放置位置，別忘了要一併更改 `namespace`

撰寫 Controller 內容

# Controller 結構

- Laravel 的 Controller 會先繼承自 `BaseController` (`Illuminate\Routing\Controller`) 以繼承大多數的 Controller 功能成  
`App\Http\Controllers\Controller.php`
- 而產生出來的每一個 Controller 而其又繼承自 `App\Http\Controllers\Controller.php` 若是在應用程式層級需要有共同的屬性與方法時，可以統一集中寫在這裡，讓其他子 Controller 可以透過繼承就有相同的能力
- 各 Controller 再依 RESTful 慣例實作各方法的內容

# RESTful Controller 範例



The screenshot shows a Sublime Text editor window with the file path `C:\Users\shengyou\Desktop\wagon\uwamp\www\laravel\app\Http\Controllers\PostsController.php (laravel) - Sublime Tex...`. The left sidebar displays the project structure, with the `PostsController.php` file selected under the `Http/Controllers` directory. The main editor area shows the following PHP code:

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 use App\Http\Requests;
8 use App\Http\Controllers\Controller;
9
10 class PostsController extends Controller
11 {
12     /**
13      * Display a listing of the resource.
14      *
15      * @return Response
16      */
17     public function index()
18     {
19         //
20     }
21
22     /**
23      * Show the form for creating a new resource.
24      *
25      * @return Response
26      */
27     public function create()
28     {
29         //
30     }
```

The status bar at the bottom indicates the cursor is at Line 1, Column 1, with 4 spaces and PHP syntax highlighting.



# RESTful 動作對應

- Laravel 的 RESTful Controller 動作對應依以下表格：

動詞	路徑	動作	名稱
GET	/resource	index	resource.index
GET	/resource/create	create	resource.create
POST	/resource	store	resource.store
GET	/resource/{id}	show	resource.show
GET	/resource/{id}/edit	edit	resource.edit
PUT/PATCH	/resource/{id}	update	resource.update
DELETE	/resource/{id}	destroy	resource.destroy

# 設定 Route

- 在 Route 裡設定將請求轉給 `Controller@action`，並依照 RESTful 慣例為 Route 命名
- Route 命名慣例：`resources.verb`

```
// app/Http/routes.php
```

```
Route::get('posts', ['as' => 'posts.index', 'uses' => 'PostsController@index']);
```

```
Route::get('posts/create', ['as' => 'posts.create', 'uses' => 'PostsController@create']);
```

```
Route::get('posts/{id}', ['as' => 'posts.show', 'uses' => 'PostsController@show']);
```

```
Route::get('posts/{id}/edit', ['as' => 'posts.edit', 'uses' => 'PostsController@edit']);
```

# 將查詢結果送到 View

- 在 Controller 裡把該頁要顯示的資料用 Model 查詢出來，並送到 Views 裡

```
// app/Http/Controllers/PostsController.php
public function index()
{
    $posts = \App\Post::all();
    $data = compact('posts');
    return view('posts.index', $data);
}

public function show($id)
{
    $post = \App\Post::find($id);
    $data = compact('post');
    return view('posts.show', $data);
}
```

# 在 View 裡顯示 Model 資料

- 當 Controller 把資料準備好送到 View 後，在 View 裡就可以直接取得 Model 或 Collection 來顯示資料

```
// resources/views/posts/index.blade.php
// $posts 是 Collection
// $post 是 Model
@foreach($posts as $post)
    <h2>{{ $post->title }}</h2>
    <h3>{{ $post->sub_title }}</h3>
    <p>{!! $post->content !!}</p>
@endforeach
```

```
// resources/views/posts/show.blade.php
// $post 是 Model
<h1>{{ $post->title }}</h1>
<h2>{{ $post->sub_title }}</h2>
<p>{!! $post->content !!}</p>
```

實作專案 Controller

# 產生專案所需 Controller

- 依專案需求建立各 Controller
- 把 app/Http/routes.php 裡的邏輯搬到對應的 Controller
- 用 Model 將資料從資料庫查詢出來，在 Controller 裡把查詢結果傳到 View 做顯示
- 測試所有頁面是否正確顯示

# 存檔點

- 試著把現在已經可以運作的程式碼加入版本控制內
- 流程提醒：
  - working directory > staging area > commit

# 單元總結

- 在這個單元裡我們學到了些什麼？
  - MVC 裡的 Controller
  - 使用 `artisan` 產生 Controller 類別
  - 依照 `RESTful` 慣例實作 Controller 裡的各方法
  - 依照 `RESTful` 慣例將 Route 與 Controller 接起來
  - 實作專案所需的 Controller，並整理 Route 設定





歡迎提問討論