جلسه ۳۰ Autoloading - Extraction و

عنوان: *PHP For Beginners, Ep 30 - Autoloading and Extraction*

موضوع اصلی:

۱. تغییر root پروژه به پوشه `public`
۲. ساخت تابع `base_path`
۳. استفاده از `extract()` برای ارسال داده به view ها
۴. اضافه کردن `spl_autoload_register` برای لود خودکار کلاس‌ها

---

1 تغییر root پروژه به پوشه `public`

دو روش معرفی شد:

- **روش با سرور داخلی PHP:**

```
php -S localhost:8888 -t public
```

با این دستور، ریشه‌ی سایت به `public` تغییر می‌کنه.

- **روش با XAMPP یا MAMP:**

نیازی به دستور بالا نیست، فقط در مرورگر آدرس رو اینطور می‌زنی:

```
http://localhost/laracast-php/public
```

📌نکته:

`__DIR__` مسیر پوشه‌ای که فایل فعلی داخلشه رو برمی‌گردونه. اگر با `-t` مسیر رو تغییر بدی، اون مسیر رو به عنوان root در نظر می‌گیره.

---

2 تابع `base_path`

در `function.php`:

```
function base_path($path) {
    return BASE_PATH . $path;
}
```

در index.php (داخل public):

```
const BASE_PATH = __DIR__ . '/../';
require_once BASE_PATH . "function.php";

$routes = require_once base_path('routes.php');
```

مزیتش اینه که همه‌ی مسیرها به صورت یکدست و وابسته به BASE_PATH ساخته می‌شن.

---

3 استخراج متغیرها در view با extract()

در function.php:

```
function view($path, $attributes = []) {
    extract($attributes);
    require base_path("controller/views/" . $path);
}
```

توضیح:

extract() کلیدهای آرایه رو به متغیر با همان نام تبدیل می‌کنه.

مثلاً:

```
view("index.view.php", [
    'heading' => "Dashboard"
]);
```

نتیجه:

- $heading → "Dashboard"
- در index.view.php **می‌توان مستقیم نوشت:**

```
<h1><?= $heading ?></h1>
```

---

4 ساخت منوی ناوبری (nav.php)

```
$pages = [
    "/laracast-php/public/" => "Dashboard",
    "/laracast-php/public/about" => "About",
    "/laracast-php/public/contact" => "Contact",
    "/laracast-php/public/notes" => "Notes"
];
?>
<div class="hidden md:block">
    <div class="ml-10 flex items-baseline space-x-4">
        <?php foreach ($pages as $key => $value): ?>
            <a href="<?= $key ?>" class="rounded-md bg-
gray-900 px-3 py-2 text-sm font-medium text-white"><?=
$value ?></a>
        <?php endforeach; ?>
    </div>
</div>
```

این بخش فقط لینک‌ها رو به صورت داینامیک تولید می‌کنه.

---

5 Autoloading با spl_autoload_register

هدف: حذف `require`های دستی برای کلاس‌ها.

در `index.php`:

```
spl_autoload_register(function ($class) {
    $path = base_path("core/") . $class . '.php';
    if (file_exists($path)) {
        require_once $path;
    }
});
```

ساختار پوشه:

```
project/
```

```
├── core/
│   ├── Validator.php
│   ├── Response.php
│   ├── Database.php
├── public/
│   └── index.php
└── function.php
```

حالا هرجا یک کلاس جدید صدا بزنی:

```
new Validator();
```

PHPخودش میره و `core/Validator.php`رو لود می‌کنه.

📌یادآوری: باید همه‌ی `require`هایی که قبلاً برای این کلاس‌ها نوشته بودیم رو حذف کنیم.

6نکته در مورد روت‌ها

در `routes.php`، مسیرها به فایل‌های کنترلر نگاشت می‌شن:

```
return [
    "/laracast-php/public/" => "controller/index.php",
    "/laracast-php/public/about" =>
"controller/about.php",
];
```

و در `index.php`بررسی می‌کنیم:

```
if (array_key_exists($url, $routes)) {
    require base_path($routes[$url]);
} else {
    http_response_code(404);
    echo "404 - Page not found";
}
```

📌جمع‌بندی مفهومی

- **base_path()** → مسیر مطلق بر اساس BASE_PATH می‌سازه.
- **extract()** → کلیدهای آرایه رو به متغیر تبدیل می‌کنه) برای ارسال داده به viewها.(
- **spl_autoload_register()** → بدون requireدستی، فایل کلاس‌ها رو خودکار لود می‌کنه.
- **ساختار پوشه‌ها** → برای امنیت، فقط publicبه عنوان root اجرا می‌شه.

با استفاده از دستور php –S localhost:8888 –t public برای تغییر ریشه سایت به پوشه publicدر اینجا یه پوشه public ساخته و فایل index.php را درون آن میریزیم بعد با دستور بالا میگیم سرور از پوشه public اجرا بشه. البته اگه از سرور داخلی خود php storm استفاده کنیم این دستور به درد میخوره و گرنه در صورت استفاده از xampp همین داخل مرورگر بزنی localhost/laracast-php/public

__DIR__ اشاره به پوشه جاری داره اگه با دستور بالا با t– پوشه جاری رو تغییر دهیم اشاره به اون پوشه میکنه.

در ابتدا یک فایل function.php ایجاد کرده و کد زیر را در آن مینویسیم :

```php
<?php
    function base_path($path){
        return BASE_PATH.$path;
    }
```

حالا به فایل index.phpکه در پوشه public هست رفته و کد زیر را مینویسیم فقط برای فایل function.php به صورت دستی آدرس رو میدیم تا تابعbase_path رو بیاره ولی برای بقیه آدرس دهی ها از تابع base_path استفاده میکنیم.

```
<?php
    const BASE_PATH = __DIR__ . '/../';
    echo BASE_PATH;
    echo '<br>';
    require_once BASE_PATH."function.php";
    $routes = require_once base_path('routes.php');
    $url =parse_url($_SERVER['REQUEST_URI'])['path'];
    echo $url;
    if(array_key_exists($url,$routes)){

        require $routes[$url];
    }else
    {
            http_response_code(404);
            echo "404 - Page not found";
            exit;
        }
```

به فایل nav.php رفته و کد زیر را اعمال میکنیم:

```
<?php
    $pages =[
        "/laracast-php/public/" => "Dashboard",
        "/laracast-php/public/about" => "About",
        "/laracast-php/public/contact"=> "Contact",
        "/laracast-php/public/notes"=> "Notes"
    ]
?>
<div class="hidden md:block">
    <div class="ml-10 flex items-baseline space-x-4">
        <!-- Current: "bg-gray-900 text-white", Default: "text-gray-300
hover:bg-gray-700 hover:text-white" -->
        <?php foreach ($pages as $key => $value): ?>
            <a href=<?=$key ?> aria-current="page" class="rounded-md bg-gray-
900 px-3 py-2 text-sm font-medium text-white"><?= $value ?></a>
        <?php endforeach; ?>
    </div>
</div>
```

حالا برای اینکه متغییر heading درست استفاده بشه به فایل function.php رفته و تابع view را تغییر میدهیم:

```
function view($path , $attributes = []){
    extract($attributes);
    require base_path("controller/views/".$path);
}
```

در تابع بالا صفت attributes که آرایه هست را اضافه کردیم و سپس داخل extract نوشتیم.

حالا به فایل index,about,contact رفته و کد زیر را مینویسیم:

اینجا تابع extract() کاری می‌کند که کلیدهای آرایه $attributes را به شکل **متغیر** در همان اسکوپ تعریف کند.

در کدی که دادی:

```
function view($path , $attributes = []){
    extract($attributes);
    require base_path("controller/views/".$path);
}
```

وقتی توی index.php این صدا زده میشه:

```
 view("index.view.php", [
    'heading' => "index"
]);
```

آرایه‌ای مثل این وارد تابع میشه:

```
$attributes = [
    'heading' => "index"
];
```

حالا با اجرای:

```
extract($attributes);
```

این اتفاق میفته:

- کلید heading تبدیل میشه به یک متغیر $heading
- مقدار "index" هم بهش اختصاص داده میشه

یعنی بعد از extract($attributes)، شما داخل فایل index.view.php می‌توانید مستقیماً بنویسید:

```
<h1><?= $heading ?></h1>
```

و بدون اینکه قبلاً $heading رو دستی ساخته باشی، مقدار "index" رو نمایش میده.

📌خلاصه:
extract() آرایه را به متغیرهایی با نام کلیدهایش تبدیل می‌کند.
در اینجا باعث شده کلید 'heading' به متغیر $heading تبدیل شود تا بتوانی راحت‌تر در view استفاده کنی.

حالا با استفاده از spl_autoload_register باید کاری کنیم که دیگه کلاسها رو require نکنیم و خود برنامه تشخیص بده قبلش یه پوشه با نام core درست کرده و تمام فایلهای کلاس رو یعنی function-validator-response-database رو داخل ان ریخته و ...

به فایل index.php داخل پوشه public رفته و کد مربوط به spl_auto_load را به شکل زیر اضافه میکنیم

```php
<?php
    const BASE_PATH = __DIR__ . '/../';
    echo BASE_PATH;
    echo '<br>';
    require_once BASE_PATH."function.php";
    spl_autoload_register(function ($class) {
        // بساز کلاس نام اساس بر را کلاس فایل مسیر
        $path = base path("/core/"). $class . '.php';

        if (file_exists($path)) {
            require_once $path;
        }
    });

    $routes = require_once base_path('routes.php');
    $url =parse_url($_SERVER['REQUEST_URI'])['path'];
    echo $url;

    if(array_key_exists($url,$routes)){

        require $routes[$url];
    }else
    {
            http_response_code(404);
            echo "404 - Page not found";
            exit;
        }
```

فقط در هرجای پروژه که کلاس ها را require کردیم باید غیرفعالش کنیم چون با تعریف دستور بالا نیازی به require کردن نیست.

۱ .دستور زیر چه کاری انجام می‌دهد؟

```
php -S localhost:8888 -t public
```

جواب:
یک سرور داخلی PHP را روی پورت ۸۸۸۸ اجرا می‌کند و ریشه سایت را روی پوشه `public`قرار می‌دهد.

---

۲ .ثابت `__DIR__`در PHP چه مقداری برمی‌گرداند؟
جواب:
مسیر کامل پوشه‌ای که فایل فعلی داخل آن قرار دارد.

---

۳ .خروجی این کد چیست؟

```
const BASE_PATH = __DIR__ . '/../';
echo BASE_PATH;
```

(فرض کنیم فایل در `C:\xampp\htdocs\laracast-php\public\index.php` است)
جواب:

```
C:\xampp\htdocs\laracast-php\
```

چون `__DIR__`مسیر `public`را برمی‌گرداند و با `/..` یک پوشه بالاتر می‌رود.

---

۴ .تابع زیر چه کاری انجام می‌دهد و مزیتش چیست؟

```
function base_path($path) {
```

```
    return BASE_PATH . $path;
}
```

جواب:

مسیر کامل یک فایل یا پوشه را بر اساس BASE_PATHمی‌سازد. مزیت: همه مسیرها از یک نقطه مرجع ساخته می‌شوند و تغییر ساختار پوشه آسان می‌شود.

---

۵ .این کد چه نتیجه‌ای دارد؟

```
extract(['heading' => 'Home Page']);
echo $heading;
```

جواب:

Home Page

چون extract()کلید headingرا به متغیر $headingتبدیل می‌کند.

---

۶ .در تابع زیر، پارامتر $attributesچه کارکردی دارد؟

```
function view($path, $attributes = []) {
    extract($attributes);
    require base_path("controller/views/" . $path);
}
```

جواب:

آرایه‌ای از داده‌ها را از کنترلر به فایل view ارسال می‌کند و با extract()آن‌ها را به متغیرهای قابل استفاده در view تبدیل می‌کند.

---

۷ spl_autoload_register() .چه مشکلی را حل می‌کند؟

جواب:

به‌طور خودکار فایل کلاس‌ها را هنگام استفاده از آن‌ها لود می‌کند و نیاز به requireیا includeدستی هر کلاس را از بین می‌برد.

---

۸ .در کدAutoloading ، این خط چه می‌کند؟

```
$path = base_path("core/") . $class . '.php';
```

جواب:

مسیر کامل فایل کلاس را با استفاده از نام کلاس و پوشه `core`می‌سازد.

---

۹ .چرا در فایل `routes.php`مسیرها به صورت آرایه تعریف می‌شوند؟

جواب:

برای نگاشت آدرس‌های URL به فایل‌های کنترلر مربوطه، تا بتوان با جستجو در آرایه مشخص کرد کدام کنترلر باید اجرا شود.

---

۱۰ .این کد چه زمانی `404 - Page not found`را نمایش می‌دهد؟

```
if (array_key_exists($url, $routes)) {
    require base_path($routes[$url]);
} else {
    http_response_code(404);
    echo "404 - Page not found";
}
```

جواب:

وقتی آدرس درخواست شده (`$url`) در آرایه `$routes`تعریف نشده باشد.