PHP programozás

Laravel: MVC, Routing

Rostagni Csaba

2025. szeptember 22.

Tartalom

MVC

- A model-view-controller (modell-nézet-vezérlő MNV) egy programtervezési minta
- Célja, hogy az adatok kezelése és azok mejelenítése legyen egymástól szétválasztva
- **Model (Modell)**: Adatok kezelése, kommunikáció az adatbázissal
- **View (Nézet)**: Az adatok megjelenítéséért felelős. Ugyanaz az adat több formában is megjelenhet
- **Contoller (Vezérlő)**: Események feldolgozása, szükséges adatok összegyűjtése, ezek alapján a nézet felhasználásával előállítja a kimenetet

Linkek:

- Modell-Nézet-Vezérlő - Wikipedia

Tartalom

Tartalom

Routing

- Megszabja, hogy milyen útvonalakat lehet elérni
- Paraméterezhető (pl id átadható)
- A paraméterek ellenőrizhetőek
- A `routes` mappában találhatóak a fájlok
- Az oldalak elérhetőségeit a `web.php` fájlban találjuk

Tartalom

A Route osztály

A routoláshoz a `Illuminate\Support\Facades\Route` osztályt használjuk. Megadhatjuk a teljes nevét,

routes/web.php

```
Illuminate\Support\Facades\Route::get('/', function () {
    return 'Főoldal';
});
```

vagy rövidíthetünk a use használatával:

routes/web.php

```
use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return 'Főoldal';
});
```

Linkek:

- Routing (Laravel dokumentáció)

A Route osztály metódusai

A különböző HTTP metódusokhoz 1-1 **osztályszintű** függvény létezik

PHP
```php
Route::get($uri, $callback);
Route::post($uri, $callback);
Route::put($uri, $callback);
Route::patch($uri, $callback);
Route::delete($uri, $callback);
Route::options($uri, $callback);
```

Továbbá egy speciális, ami egyszerre többet is le tud kezelni

PHP
```php
Route::match()
```

match() és any()

PHP
```php
Route::match($array, $uri, $callback);
```

- A `Route::match()` elsőként egy string tömböt vár, ahol megadhatjuk mely metódust kezelje

PHP
```php
Route::any($uri, $callback);
```

- A `Route::any()` bármelyik metódust kezeli

Sorrend

A fájlban elsőként a pontosabb meghatározás szerepeljen (`get()`, `post()`, stb...) különben az általános meghatározás `match()`, vagy `any()` hamarabb életbe lépne!

Tartalom

Egyszerű szöveges válasz

routes/web.php

```php
use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return 'Főoldal';
});
```

- Az első paraméter az útvonal.
  - A példán a weboldal gyökere, azaz a főoldal (http://example.com)
- A második paramétere lehet egy névtelen függvény
  - Itt egy szöveget ad vissza, a teljes HTML kód hiányozni fog

Linkek:

- Anonymous functions (PHP dokumentáció)

Egyszerű szöveges válasz paraméter alapján

routes/web.php

```php
use Illuminate\Support\Facades\Route;

Route::get('/hello/{name}', function ($name) {
    return "Hello $name";
});
```

- Az útvonal paraméterezhető. A példán `/hello/name` aloldal látható, ahol a `name` egy paraméter (http://example.com/hello/Peti)
- A függvény a bemeneten egy `$name` paramétert kap
- Egyszerű szöveget ad vissza (pl: "Hello Peti")

Figyelem!

Az útvonal paraméter neve (`'{name}'`) és a névtelen függvény paraméterének neve (`$name`) legyen ugyanaz!

Egyszerű szöveges válasz több paraméterrel

routes/web.php

```php
use Illuminate\Support\Facades\Route;

$uri = '/{lang}/hello/{first}/{last}';
Route::get($uri, function ($lang,$first,$last) {
    if($lang == "hu") {
        return "Szia $last $first";
    }
    else {
        return "Hello {$first} {$last}";
    }
});
```

- Több paramétert is megadhatunk (http://example.com/hu/hello/Andor/Kiss)

Figyelem!

Az útvonalban meghatározott sorrendben jelenjenek meg a függvényben is a paraméterek!

Opcionális paraméter

routes/web.php

```php
use Illuminate\Support\Facades\Route;

Route::get('/hello/{name?}', function ($name = null) {
    if(is_null($name)) {
        return "Hello";
    }
    else {
        return "Hello {$name}";
    }
});
```

- Az útvonal paraméter esetén a ? jelzi, hogy opcionális: (`'{name?}'`)
- A függvény bemenetének kell alapértelmezett értéket megadni ! (`$name = null`)

Tartalom

Megszorítások

- Szükség lehet validálni az URL-ben kapott értékeket
- A reguláris kifejezések erre alkalmasak
- Nem a legkönnyebb, de van egyszerűbb megoldás

where()

routes/web.php

```
Route::get('/hello/{username}', function ($username) {
    return "Hello $username"
})->where('username', '[A-Za-z]+');
```

- A `$username` csak szöveget tartalmazhat, a példában még szóközt sem

routes/web.php

```
Route::get('/hello/{username}', function ($username) {
    return "Hello $username"
})->whereAlpha('username');
```

- A `whereAlpha()` metódus ugyanazt engedélyezi

Egy útvonal, több feltétel

routes/web.php

```
Route::get('/user/{nev}/{kor}', function ($nev, $kor) {
    // ...
})->whereAlpha('nev')->whereNumber('kor');
```

- A szűrő metódusok egymás után láncolhatóak, köztük és kapcsolat lesz érvényben

PHP

```
Route::get('/user/{nev}/{kor}', function ($nev, $kor) {
    // ...
})->where(['nev' => '[a-z]+', 'kor' => '[0-9]+']);
```

- A `where()` metódus paramétereként egy egyszerű string helyett kaphat egy asszociatív tömböt, ahol a kulcs a paraméter neve, az érték pedig a regex

Előre definiált ellenőrzők

Az előre definiált metódusok csak bizonyos feltételeknek megfelelő paramétereket fogadnak el, ehhez még regex tudás sem szükséges

- A `whereAlpha()` csak a betűket
- A `whereNumber()` csak a számokat
- A `whereAlphaNumeric()` csak az betűket és a számokat
- A `whereUuid()` csak az UUID-kat

Linkek:

- CreatesRegularExpressionRouteConstraints osztály

Tartalom

Elnevezett útvonalak

- A `name()` metódus segítségével elnevezhetjük az adott útvonalat
- Egy paramétert vár, ami legyen szöveg, az útvonal neve
- A per jelek helyett célszerű pontot alkalmazni
- A szóközöket is illik kerülni, ezek kötőjelekkel helyettesítendőek

routes/web.php

```
Route::get('/hello', function () {
    return "Hello"
})->name('hello');
```

routes/web.php

```
use Illuminate\Support\Facades\Route;

Route::get('/hello/{name}', function ($name) {
    return "Hello $name";
})->name("hello.nev");
```

Tartalom

URL generálása egyszerű útvonal alapján

- Az elnevezett útvonalakra a Laravel tud linkeket generálni
- Paraméterként az elnevezett útvonal nevét kell megadni

PHP
```php
route('hello');
```

routes/web.php
```php
Route::get('/hello', function () {
    return "Hello"
})->name('hello');
```

URL generálása paraméterezéssel

- Az elnevezett útvonalakra a Laravel tud linkeket generálni
- Első paraméterként az elnevezett útvonal nevét kell megadni
- A második paraméter az útvonal paramétereinek a tömbje

PHP
```php
route('hello.nev', ['name' => 'Robi',])
```

routes/web.php
```php
Route::get('/hello/{name}', function ($name) {
    return "Hello $name";
})->name("hello.nev");
```

Tartalom

Tartalom

Controller bevezető

- Ha mindent a route fájlban írunk meg átláthatatlanná válik
- Egy-egy témakörhöz tartozó tennivalókat vezérlőkbe (controller) csoportosítjuk
- A különböző tennivalókat eseményeknek (action) nevezzük
  - Autók listázása (index vagy list)
  - Egy autó megjelenítése (view vagy show)
  - Autók beszúrása (create, insert vagy store)
  - Egy autó módosítása (update vagy edit)
  - Egy autó törlése (delete, remove), vagy destroy
- A webes részhez tartozó vezérlők az `app/Http/Controllers` mappában találhatóak
- Létrehozhatunk controllert a konzolhoz is

Linkek:

- Laravel dokumentáció: Controllers

Controller generálása artisan segítségével

CMD

```
php artisan make:controller CarController
```

- Az artisan a `make:controller` alkalmas controller generálására
- A végén a controller nevét kell megadni
- Az osztály postfixe (utótagja): `Controller`
- A parancs hatására az `app/Http/Controllers` mappában létrejön egy `CarController.php` fájl, benne az osztállyal.

CarController

App/Http/Controllers/CarController.php

```php
<?php
namespace App\Http\Controllers;

use Illuminate\Http\Request;

class CarController extends Controller
{
    // ...
}
```

- Beállítja a névteret
- A létrejött osztály, az azonos névtérben lévő `Controller` osztályból származik

Tartalom

Egyszerű szöveges action

routes/web.php

```
Route::get('/cars', function () {
    return "Autók listázása";
})
```

- Emlékeztetőül hogyan nézett ki a route fájlban, ezt kell átalakítani
  - A route-ban lévő függvény a `CarController` osztályba kerül
  - A láthatóságának `public`-nak kell lennie
  - A névtelen függvény nevet is kap

Egyszerű szöveges action

App/Http/Controllers/CarController.php

```
class CarController extends Controller {
    public function index() {
        return "Autók listázása";
    }
}
```

- Ezzel az átalakítással elérhetetlenné tettük

Egyszerű szöveges action

routes/web.php

```php
use App\Http\Controllers\CarController;

Route::get('/cars', [CarController::class, "index"])
```

- A második paraméter itt nem a lefuttatandó függvény lesz, hanem egy tömb, aminek
  - az első eleme a `CarController` osztálytól lekéri a nevét a `::class` segítségével
  - a második eleme az action lesz, azaz a lefuttatandó metódus neve
- A Controllert célszerű megadni a use-ban