Routing

LaraMind

Sidrit Trandafili

@sidis405

@strandafili

- Sviluppatore PHP 15+ anni
- Laravel da 4 anni
- Laravel Certified
- Back-end
  - php
  - java
  - node
  - python
  - dart
- Front-end
  - VueJS
  - Vanilla JS
  - TailwindCSS
  - Sass
  - dart
- Devops + Sysops da 14 anni

Cosa sono le Route?

Le Routes

Una route è un meccanismo che serve a configurare la nostra applicazione in modo da rispondere a un certo URL.
Si tratta di una modalità per per creare un URL richiesta (request URL) al quale la nostra applicazione possa dare un output.

Questi url non devono necessariamente corrispondere a file specifici esistenti sul nostro server. Stiamo parlando di url dinamici e riscrivibili che possono essere costruiti in qualsiasi modo, anche per essere ottimizzati in ottica SEO.

Le Routes

In Laravel 5.6, le routes vengono create e gestite dentro la cartella routes. Le routes per le interazioni web, dunque le iterazioni che avvengono sul browser sono salvate in web.php . Alternativamente, se con Laravel stiamo costruendo un API, queste routes vengono salvate in api.php .

Un'istallazione default di Laravel appena scaricata, nel file web.php contiene una sola route, quella che mostra la pagina iniziale di benvenuto.

Le Routes

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

Questa route è definita per rispondere alla homepage. Ogni volta che Laravel riceve una richiesta per la pagina / , risponderà con
la view welcome . (un file di markup, html, che discuteremo nell'articolo dedicato a Blade).

Le Routes

Costruire una route è facilissimo. Apriamo il file desiderato (web.php oppure api.php), e iniziamo a scrivere Route:: .
Questa istruzione viene seguita dal nome del tipo di richiesta che vogliamo associare a questa route (verbi HTTP), e alla funzione che vorremmo eseguire al ricevimento della richiesta stessa.

I metodi base che Laravel offre per le routes sono:

- get
- post
- put
- delete
- patch
- options

Come creare nuove Routes?

La più comune: GET Route

Prendiamo con un semplicissimo esempio. La tabella multipli del numero 2 .

```
Route::get('/tabella-multipli', function () {
    for(range(1, 10) as $x){
        echo "$x * 2 = ". $x * 2 ."<br>";
    }
});
```

In questo esempio abbiamo creato una route di richiesta GET per l'url /tabella-multipli .

Come creare nuove Routes?

Come dovremmo strutturare la nostra route se l'utente volesse specificare il numero per il quale creare la tabellina?

```
Route::get('/tabella-multipli/{numero}', function ($numero) {
    for(range(1, 10) as $x){
        echo "$x * $numero = ". $x * $numero ."<br>";
    }
});
```

Abbiamo creato una route che adesso accetta un parametro e lo passa alla funzione da eseguire.

Ogni volta che un utente esegue una richiesta a /tabella-multipli e inserisce un numero (es. /tabella-multipli/7), la tabella di moltiplicazione di quel numero verrà stampata su schermo.

Come creare nuove Routes?

Notiamo che in questo momento stiamo gestendo due routes che fanno praticamente la stessa cosa. Una che mostra la tabellina del 2, e una che accetta un numero per il quale mostrare la tabellina.
Come facciamo a condensare queste due dinamiche in una route sola?

Laravel ci viene incontro con i Parametri Opzionali. Se aggiungiamo un ? dopo il parametro che vogliamo rendere opzionale e forniamo un valore di default, il sistema eseguirà quella route in presenza o meno del parametro stesso.

Come creare nuove Routes?

```
Route::get('/tabella-multipli/{numero?}', function ($numero = 2) {
        for(range(1, 10) as $x){
            echo "$x * $numero = ". $x * $numero ."<br>";
        }
});
```

Questa è l'unica route che ci serve. Se un numero viene fornito, verrà passato alla funzione, altrimenti la funzione andrà in esecuzione come se il valore passatogli fosse 2 .

Espressioni Regolari per controllare i parametri delle Routes

Cosa succederebbe se invece di passare un numero come parametro, scrivessimo qualcos'altro?
Cosa succederebbe nel caso che la nostra (semplicissima) applicazione ricevesse una richiesta per la pagina

/tabella-multipli/test ? Questo causerebbe un errore.

Nel sistema routing di Laravel è possibile definire quando una Route deve essere eseguita, non solo con URL e verbo HTTP, ma anche con espressioni regolari tramite la funzione where.

Espressioni Regolari per controllare i parametri delle Routes

Riscriviamo la nostra Route in modo da accettare solo numeri come input.

```
Route::get('/tabella-multipli/{numero?}', function ($numero = 2) {
    for(range(1, 10) as $x){
        echo "$x * $numero = ". $x * $numero ."<br>";
    }
})->where('numero',  '[0-9]+');
```

Se il nostro url non contiene un numero intero (definito dall'espressione regolare [0-9]+),
il server risponderà con un 404,
ovvero (in termini Laravel), un errore NotFoundHttpException .

Routing e il Controller

In Laravel è possibile assegnare il metodo di un Controller a una Route. Ogni volta che l'utente richiede la route in questione, le operazioni non si svolgeranno più nella funzione della Route, ma nel metodo del Controller assegnato.

```
Route::get('/index',  'IlNostroController@nomeFunzione');
```

Come per le Route semplici, anche questa istruzione inizia con Route:: seguita dal tipo di metodo (get, post, put eccetera).
Bisogna poi specificare l'url al quale si vuole rispondere (in questo caso '/index'), seguito dal nome del Controller creato da noi, il simbolo @, e il metodo (la funzione) che abbiamo creato nel controller e che svolge il lavoro desiderato.

Assegnare nomi alle Route

In Laravel è possibile assegnare nomi alle Route, cosa che si rivela molto utile in vari scenari.

È possibile usare questa funzionalità tramite la funzione name 'appesa' alle nostre Routes.

```
Route::get('/tabella-multipli/{numero?}', function ($numero = 2) {
        for(range(1, 10) as $x){
            echo "$x * $numero = ". $x * $numero ."<br>";
        }
    })->where('numero',  '[0-9]+')->name('tabellina);
```

In questo modo possiamo generare velocemente le URL per queste route

```
$url  =  route('tabellina');
```

```
return  redirect()->route('tabellina');
```