

```
Homestead — vagrant@homestead: ~/Code/laravel — ssh ◀
vagrant@homestead:~/Code/laravel$ php artisan inspire
It is quality rather than quantity that matters. — Lucius Annaeus Seneca
vagrant@homestead:~/Code/laravel$
```

# 40 (and counting) Laravel Quick Tips

**Prepared by:**

Povilas Korop / LaravelDaily Team

[www.laraveldaily.com](http://www.laraveldaily.com)

[povilas@laraveldaily.com](mailto:povilas@laraveldaily.com)

**Last updated:**

October 2018

## Tip 1. Invokable Controllers

From Laravel 5.6.28 - if you want to create a controller with just one action, you can use `__invoke()` method and even create "invokable" controller.

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

use App\User;
use App\Http\Controllers\Controller;

class ShowProfile extends Controller
{
    /**
     * Show the profile for the given user.
     *
     * @param int $id
     * @return Response
     */
    public function __invoke($id)
    {
        return view('user.profile', ['user' => User::findOrFail($id)]);
    }
}
```

Routes:

```
Route::get('user/{id}', 'ShowProfile');
```

Artisan command to generate this controller:

```
php artisan make:controller ShowProfile --invokable
```

---

## Tip 2. Unsigned Integer

For foreign key migrations instead of `integer()` use `unsignedInteger()` type or `integer()->unsigned()`, otherwise you may get SQL errors.

```
Schema::create('employees', function (Blueprint $table) {
    $table->unsignedInteger('company_id');
    $table->foreign('company_id')->references('id')->on('companies');
    // ...
});
```

---

### Tip 3. OrderBy on Eloquent relationships

You can specify `orderBy()` directly on your Eloquent relationships.

```
public function products()
{
    return $this->hasMany(Product::class);
}

public function productsByName()
{
    return $this->hasMany(Product::class)->orderBy('name');
}
```

---

### Tip 4. Order of Migrations

If you want to change the order of DB migrations, just rename the file's timestamp, like from `2018_08_04_070443_create_posts_table.php` to `2018_07_04_070443_create_posts_table.php` (changed from `2018_08_04` to `2018_07_04`). They run in alphabetical order.

---

### Tip 5. Raw DB Queries

You can use RAW DB queries in various places, including `havingRaw()` function after `groupBy()`.

```
Product::groupBy('category_id')->havingRaw('COUNT(*) > 1')->get();
```

---

## Tip 6. \$loop variable in foreach

Inside of foreach loop, check if current entry is first/last by just using \$loop variable.

```
@foreach ($users as $user)
    @if ($loop->first)
        This is the first iteration.
    @endif

    @if ($loop->last)
        This is the last iteration.
    @endif

    <p>This is user {{ $user->id }}</p>
@endforeach
```

There are also other properties like `$loop->iteration` or `$loop->count`.

More here: <https://laravel.com/docs/master/blade#the-loop-variable>

---

## Tip 7. Eloquent where date methods

In Eloquent, check the date with functions `whereDay()`, `whereMonth()`, `whereYear()`, `whereDate()` and `whereTime()`.

```
$products = Product::whereDate('created_at', '2018-01-31')->get();
$products = Product::whereMonth('created_at', '12')->get();
$products = Product::whereDay('created_at', '31')->get();
$products = Product::whereYear('created_at', date('Y'))->get();
$products = Product::whereTime('created_at', '=', '14:13:58')->get();
```

---

## Tip 8. Route group within a group

in Routes, you can create a group within a group, assigning a certain middleware only to some URLs in the "parent" group.

```
Route::group(['prefix' => 'account', 'as' => 'account.'], function() {  
  
    Route::get('login', 'AccountController@login');  
    Route::get('register', 'AccountController@register');  
  
    Route::group(['middleware' => 'auth'], function() {  
        Route::get('edit', 'AccountController@edit');  
    });  
  
});
```

---

## Tip 9. Increments and decrements

if you want to increment some DB column in some table, just use `increment()` function. Oh, and you can increment not only by 1, but also by some number, like 50.

```
Post::find($post_id)->increment('view_count');  
User::find($user_id)->increment('points', 50);
```

---

## Tip 10. Does view file exist?

You can check if View file exists before actually loading it.

```
if (view()->exists('custom.page')) {  
    // Load the view  
}
```

You can even load an array of views and only the first existing will be actually loaded.

```
return view()->first(['custom.dashboard', 'dashboard'], $data);
```

---

## Tip 11. No timestamp columns

If your DB table doesn't contain timestamp fields `created_at` and `updated_at`, you can specify that Eloquent model wouldn't use them, with `$timestamps = false` property.

```
class Company extends Model
{
    public $timestamps = false;
}
```

---

## Tip 12. Migration fields with timezones

Did you know that in migrations there's not only `timestamps()` but also `timestampsTz()`, for the timezone?

```
Schema::create('employees', function (Blueprint $table) {
    $table->increments('id');
    $table->string('name');
    $table->string('email');
    $table->timestampsTz();
});
```

Also, there are columns `dateTimeTz()`, `timeTz()`, `timestampTz()`, `softDeletesTz()`.

---

## Tip 13. Eloquent has() deeper

You can use Eloquent `has()` function to query relationships even two layers deep!

```
// Author -> hasMany(Book::class);
// Book -> hasMany(Rating::class);
$authors = Author::has('books.ratings')->get();
```

---

## Tip 14. Database migrations column types

There are interesting column types for migrations, here are a few examples.

```
$table->geometry('positions');
$table->ipAddress('visitor');
$table->macAddress('device');
$table->point('position');
$table->uuid('id');
```

See all column types: <https://laravel.com/docs/master/migrations#creating-columns>

---

## Tip 15. Artisan command help

To check the options of artisan command, Run artisan commands with `--help` flag. For example, `php artisan make:model --help` and see how many options you have:

Options:

```
-a, --all          Generate a migration, factory, and resource controller for
the model
-c, --controller  Create a new controller for the model
-f, --factory     Create a new factory for the model
--force          Create the class even if the model already exists.
-m, --migration   Create a new migration file for the model.
-p, --pivot       Indicates if the generated model should be a custom
intermediate table model.
-r, --resource    Indicates if the generated controller should be a resource
controller.
-h, --help        Display this help message
-q, --quiet       Do not output any message
-V, --version     Display this application version
--ansi           Force ANSI output
--no-ansi        Disable ANSI output
-n, --no-interaction Do not ask any interactive question
--env[=ENV]      The environment the command should run under
-v|vv|vvv, --verbose Increase the verbosity of messages: 1 for normal output, 2
for more verbose output and 3 for debug
```

---

## Tip 16. Default Timestamp

While creating migrations, you can use `->timestamp()` column type with option `->useCurrent()`, it will set `CURRENT_TIMESTAMP` as default value.

```
$table->timestamp('created_at')->useCurrent();  
$table->timestamp('updated_at')->useCurrent();
```

---

## Tip 17. Set logged in user with Observers

Use `make:observer` and fill in `creating()` method to automatically set up `user_id` field for current logged in user.

```
class PostObserver  
{  
    /**  
     * Handle to the post "creating" event.  
     */  
    * @param \App\Post $post  
    * @return void  
    */  
    public function creating(Post $post)  
    {  
        $post->user_id = auth()->id();  
    }  
}
```

---

## Tip 18. Soft-deletes: multiple restore

When using soft-deletes, you can restore multiple rows in one sentence.

```
Post::withTrashed()->where('author_id', 1)->restore();
```

---



## Tip 19. Has Many. How many exactly?

In Eloquent `hasMany()` relationships, you can filter out records that have X amount of children records.

```
// Author -> hasMany(Book::class)
$authors = Author::has('books', '>', 5)->get();
```

---

## Tip 20. Image validation

While validating uploaded images, you can specify the dimensions you require.

```
'photo' => 'dimensions:max_width=4096,max_height=4096'
```

---

## Tip 21. Wildcard subdomains

You can create route group by dynamic subdomain name, and pass its value to every route.

```
Route::domain('{username}.workspace.com')->group(function () {
    Route::get('user/{id}', function ($username, $id) {
        //
    });
});
```

---

## Tip 22. Exact Laravel version

Find out exactly what Laravel version you have in your app, by running command  
`php artisan --version`

---

## Tip 23. Testing email into laravel.log

If you want to test email contents in your app but unable or unwilling to set up something like Mailgun, use `.env` parameter `MAIL_DRIVER=log` and all the email will be saved into `storage/logs/laravel.log` file, instead of actually being sent.

---

## Tip 24. Error code Blade pages

If you want to create a specific error page for some HTTP code, like 500 - just create a blade file with this code as filename, in `resources/views/errors/500.blade.php`, or `403.blade.php` etc, and it will automatically be loaded in case of that error code.

---

## Tip 25. Factory callbacks

While using factories for seeding data, you can provide Factory Callback functions to perform some action after record is inserted.

```
$factory->afterCreating(App\User::class, function ($user, $faker) {  
    $user->accounts()->save(factory(App\Account::class)->make());  
});
```

---

## Tip 26. Artisan command parameters

When creating Artisan command, you can ask the input in variety of ways: `$this->confirm()`, `$this->anticipate()`, `$this->choice()`.

```
// Yes or no?  
if ($this->confirm('Do you wish to continue?')) {  
    //  
}  
  
// Open question with auto-complete options  
$name = $this->anticipate('What is your name?', ['Taylor', 'Dayle']);  
  
// One of the listed options with default index  
$name = $this->choice('What is your name?', ['Taylor', 'Dayle'],  
$defaultIndex);
```

---

## Tip 27. Preview Mailables

If you use Mailables to send email, you can preview the result without sending, directly in your browser. Just return a Mailable as route result:

```
Route::get('/mailable', function () {  
    $invoice = App\Invoice::find(1);  
  
    return new App\Mail\InvoicePaid($invoice);  
});
```

---

## Tip 28. Don't create Controllers

If you want route to just show a certain view, don't create a Controller method, just use `Route::view()` function.

```
// Instead of this  
Route::get('about', 'TextsController@about');  
// And this  
class TextsController extends Controller  
{  
    public function about()  
    {  
        return view('texts.about');  
    }  
}  
  
// Do this  
Route::view('about', 'texts.about');
```

---

## Tip 29. Blade @auth

Instead of if-statement to check logged in user, use @auth directive.

Typical way:

```
@if(auth()->user())
    // The user is authenticated.
@endif
```

Shorter:

```
@auth
    // The user is authenticated.
@endauth
```

---

## Tip 30. Model all: columns

When calling Eloquent's `Model::all()`, you can specify which columns to return.

```
$users = User::all(['id', 'name', 'email']);
```

---

## Tip 31. Localhost in .env

Don't forget to change `APP_URL` in your `.env` file from `http://localhost` to real URL, cause it will be the basis for any links in your email notifications and elsewhere.

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:9PHz3TL5C4YrdV6Gg/Xkkmx9btaE93j7rQTUZWm2MqU=
APP_DEBUG=true
APP_URL=http://localhost
```

---

## Tip 32: What's behind the routes?

Want to know what routes are actually behind `Auth::routes()`? Check the file `/vendor/laravel/framework/src/illuminate/Routing/Router.php`.

Note that in Laravel 5.7 it also has a new option for verification emails.

```
public function auth(array $options = [])
{
    // Authentication Routes...
    $this->get('login', 'Auth\LoginController@showLoginForm')->name('login');
    $this->post('login', 'Auth\LoginController@login');
    $this->post('logout', 'Auth\LoginController@logout')->name('logout');

    // Registration Routes...
    if ($options['register'] ?? true) {
        $this->get('register',
'Auth\RegisterController@showRegistrationForm')->name('register');
        $this->post('register', 'Auth\RegisterController@register');
    }

    // Password Reset Routes...
    $this->get('password/reset',
'Auth\ForgotPasswordController@showLinkRequestForm')->name('password.request');
    $this->post('password/email',
'Auth\ForgotPasswordController@sendResetLinkEmail')->name('password.email');
    $this->get('password/reset/{token}',
'Auth\ResetPasswordController@showResetForm')->name('password.reset');
    $this->post('password/reset',
'Auth\ResetPasswordController@reset')->name('password.update');

    // Email Verification Routes...
    if ($options['verify'] ?? false) {
        $this->emailVerification();
    }
}

public function emailVerification()
{
    $this->get('email/verify',
'Auth\VerificationController@show')->name('verification.notice');
    $this->get('email/verify/{id}',
'Auth\VerificationController@verify')->name('verification.verify');
    $this->get('email/resend',
'Auth\VerificationController@resend')->name('verification.resend');
}
```

## Tip 33. To Fail or not to Fail

In addition to `findOrFail()`, there's also Eloquent method `firstOrFail()` which will return 404 page if no records for query are found.

```
$user = User::where('email',  
'povilas@laraveldaily.com')->firstOrFail();
```

---

## Tip 34. Column name change

in Eloquent Query Builder, you can specify "as" to return any column with a different name, just like in plain SQL query.

```
$users = DB::table('users')  
->select('name', 'email as user_email')  
->get();
```

---

## Tip 35. Logging with parameters

You can write `Log::info()`, or shorter `info()` message with additional parameters, for more context about what happened.

```
Log::info('User failed to login.', ['id' => $user->id]);
```

---

## Tip 36. Default Model

You can assign a default model in `belongsTo` relationship, to avoid fatal errors when calling it like `{{ $post->user->name }}` if `$post->user` doesn't exist.

```
/**  
 * Get the author of the post.  
 */  
public function user()  
{  
    return $this->belongsTo('App\User')->withDefault();  
}
```

---

## Tip 37. Use hasMany to create Many

If you have `hasMany()` relationship, you can use `saveMany()` to save multiple "child" entries from your "parent" object, all in one sentence.

```
$post = Post::find(1);
$post->comments()->saveMany([
    new Comment(['message' => 'First comment']),
    new Comment(['message' => 'Second comment']),
]);
```

---

## Tip 38. More convenient DD

Instead of doing `dd($result)`; you can put `->dd()` as a method directly at the end of your Eloquent sentence, or any Collection.

```
// Instead of
$users = User::where('name', 'Taylor')->get();
dd($users);

// Do this
$users = User::where('name', 'Taylor')->get()->dd();
```

---

## Tip 39. Map query results

After Eloquent query you can modify rows by using `map()` function in Collections.

```
$users = User::where('role_id', 1)->get()->map(function (User $user) {
    $user->some_column = some_function($user);
    return $user;
});
```

---

## Tip 40. Custom validation error messages

You can customize validation error messages per **field**, **rule** and **language** - just create a specific language file `resources/lang/xx/validation.php` with appropriate array structure.

```
'custom' => [  
    'email' => [  
        'required' => 'We need to know your e-mail address!',  
    ],  
],
```

# To be continued...

Follow [@DailyLaravel](https://twitter.com/DailyLaravel) on Twitter for updates

Or subscribe to our weekly newsletter:

<http://bit.ly/laravel-newsletter>