Laravel News

20 Laravel Eloquent Tips and Tricks

Tutorials April 13th, 2018

Eloquent ORM seems like a simple mechanism, but under the hood, there's a lot of semi-hidden functions and less-known ways to achieve more with it. In this article, I will show you a few tricks.

1. Increments and Decrements

Instead of this:

```
$article = Article::find($article_id);
$article->read_count++;
$article->save();
```

You can do this:

```
$article = Article::find($article_id);
$article->increment('read_count');
```

Also these will work:

```
Article::find($article_id)->increment('read_count');
Article::find($article_id)->increment('read_count', 10); // +10
Product::find($produce_id)->decrement('stock'); // -1
```

---

2. XorY methods

Eloquent has quite a few functions that combine two methods, like "please do X, otherwise do Y".

Example 1 – **`findOrFail()`**:

Instead of:

```
$user = User::findOrFail($id);
```

Example 2 – `firstOrCreate()`:

Instead of:

```
$user = User::where('email', $email)->first();
if (!$user) {
  User::create([
    'email' => $email
  ]);
}
```

Do just this:

```
$user = User::firstOrCreate(['email' => $email]);
```

---

3. Model boot() method

There is a magical place called `boot()` in an Eloquent model where you can override default behavior:

```
class User extends Model
{
    public static function boot()
    {
        parent::boot();
        static::updating(function($model)
        {
```

generate UUID field at that moment.

```
public static function boot()
{
  parent::boot();
  self::creating(function ($model) {
    $model->uuid = (string)Uuid::generate();
  });
}
```

---

4. Relationship with conditions and ordering

This is a typical way to define relationship:

```
public function users() {
    return $this->hasMany('App\User');
}
```

But did you know that at this point we can already add **`where`** or **`orderBy`**?
For example, if you want a specific relationship for some type of users, also ordered by email, you can do this:

```
public function approvedUsers() {
```

```
class User extends Model {
    protected $table = 'users';
    protected $fillable = ['email', 'password']; // which fielc
    protected $dates = ['created_at', 'deleted_at']; // which f
    protected $appends = ['field1', 'field2']; // additional va
}
```

But wait, there's more:

```
protected $primaryKey = 'uuid'; // it doesn't have to be "id"
public $incrementing = false; // and it doesn't even have to be
protected $perPage = 25; // Yes, you can override pagination cc
const CREATED_AT = 'created_at';
const UPDATED_AT = 'updated_at'; // Yes, even those names can b
public $timestamps = false; // or even not used at all
```

And there's even more, I've listed the most interesting ones, for more please check out the code of default abstract Model class and check out all the traits used.

---

6. Find multiple entries

```
$users = User::where('approved', 1)->get();
```

Into this:

```
$users = User::whereApproved(1)->get();
```

Yes, you can change the name of any field and append it as a suffix to "where" and it will work by magic.

Also, there are some pre-defined methods in Eloquent, related to date/time:

```
User::whereDate('created_at', date('Y-m-d'));
User::whereDay('created_at', date('d'));
User::whereMonth('created_at', date('m'));
User::whereYear('created_at', date('Y'));
```

9. Eloquent::when() – no more if-else's

Many of us write conditional queries with "if-else", something like this:

```
if (request('filter_by') == 'likes') {
    $query->where('likes', '>', request('likes_amount', 0));
}
if (request('filter_by') == 'date') {
    $query->orderBy('created_at', request('ordering_rule', 'des
}
```

But there's a better way – to use **`when()`**:

```
$query = Author::query();
```

But what if the author is deleted, or isn't set for some reason? You will get an error, something like “property of non-object”.

Of course, you can prevent it like this:

```
{{ $post->author->name ?? '' }}
```

But you can do it on Eloquent relationship level:

```
public function author()
{
    return $this->belongsTo('App\Author')->withDefault();
```

```
1  $clients = Client::orderBy('full_name')->get(); // doesn't work
```

The solution is quite simple. We need to order the results **after** we get them.

```
1  $clients = Client::get()->sortBy('full_name'); // works!
```

```
    ->whereRaw('price > IF(state = "TX", ?, 100)', [200])
    ->get();

```

Laravel News Partners

LOADFORGE

TIGHTEN

Kirschbaum

Scout APM

SimpleBackups

SHIFT

DevSquad

Curotec

Newsletter

Join 33,000+ others and never miss out on new tips, tutorials, and more.

Your email address

Subscribe

RingCentral

View All Jobs

Laravel Full Stack Developer

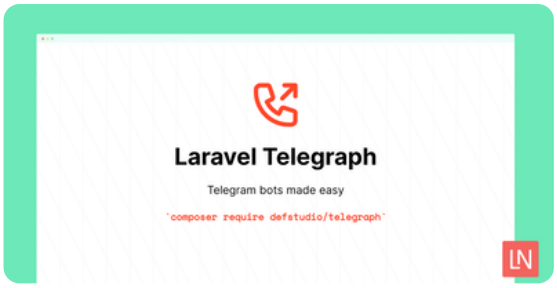
Software Engineer PHP (m/f/d) - o

Full Stack Engineer Senior PHP/

Senior Full Stack Level Developer

aimeos

Most recent



March 1st, 2022

February 28th, 2022

Subscribe to the Email Newsletter

Your email address

Subscribe

Follow Laravel News on

Facebook Twitter LinkedIn Instagram

Design & development by

Blog Newsletter Tutorials Packages Training Resources

Partners Advertising Archive Login Contact

© 2012 - 2022 Laravel News
A division of dotdev inc.