



Laravel 5.1 : Model Factories

En versiones anteriores de Laravel usábamos componentes como Faker para crear datos de prueba y así poder crear una gran cantidad de registros de forma automática en nuestro proyecto de una forma muy sencilla, sin embargo se trataba de un componente externo que debíamos agregar o incluir en el proyecto. Desde ahora esta funcionalidad viene integrada en el núcleo de Laravel 5.1.

Model factories

Son una muy buena forma para crear datos de prueba en la aplicación, esta clase recibe una instancia de la clase Faker que permite generar una gran variedad de valores de forma aleatoria.

Puedes ver más sobre Faker en la [pagina oficial del proyecto en GitHub](#).

Por defecto Laravel 5.1 trae consigo un ejemplo del uso de este nuevo componente, si vamos al archivo *database/factories/ModelFactory.php* veremos algo como esto :

```
1 $factory->define(App\User::class, function ($faker) {  
2     return [  
3         'name' => $faker->name,  
4         'email' => $faker->email,  
5         'password' => str_random(10),  
6         'remember_token' => str_random(10),  
7     ];  
8 });
```

Es aquí donde podemos definir los datos de prueba y retornarlos en forma de un array.

```
1 $factory->define();
```

Recibe como primer parámetro el modelo de la base de datos que deseamos usar, en ocasiones puede ocurrir el caso de que se desee crear datos de prueba de varios tipos como por ejemplo una serie de post activos o inactivos. para ello se puede usar.

```
1 $factory->define(App\Post::class, 'active');  
2 $factory->define(App\Post::class, 'inactive');
```

Por supuesto siempre seguido del tercer parámetro que seria la función que devuelve el array de datos.

Seeders y Model Factories

Si queremos hacer el llamado a esta función para llenar la base de datos con información de prueba,

podemos hacerlo desde el archivo *database/seeds/DatabaseSeeder.php*

```
1 public function run()  
2 {  
3     factory('App\User', 50)->create();  
4 }
```

Solo debemos hacer el llamado mediante

```
1 factory('App\User', 50)
```

Enviando como primer parámetro el modelo para el cual hemos definido los datos de prueba anteriormente y como segundo parámetro la cantidad de registros que deseamos crear.

De igual forma si quieres llamar a un tipo específico que hayas definido, como en el ejemplo de los post activos e inactivos, puedes enviar ese valor como segundo parámetro y la cantidad de registros como un tercero.

```
1 $post = factory('App\Post', 'active')->make();
```

Veamos un ejemplo

Primero que todo recuerda configurar correctamente las credenciales de acceso a tu base de datos en el archivo *.env* de la aplicación.

Creamos un nuevo modelo con

```
1 $ php artisan make:model Post
```

Editamos el archivo *app/Post.php*

```
1 class Post extends Model  
2 {
```

```

3      /**
4       * The database table used by the model.
5       *
6       * @var string
7       */
8      protected $table = 'posts';
9
10     /**
11      * The attributes that are mass assignable.
12      *
13      * @var array
14      */
15     protected $fillable = ['title', 'body', 'active'];
16 }

```

Creamos una nueva migración con

```

1 $ php artisan make:migration create_posts_table

```

Editamos el archivo de la migración

```

1      public function up()
2      {
3          Schema::create('posts', function (Blueprint $table) {
4              $table->increments('id');
5              $table->string('title')->unique();
6              $table->text('body');
7              $table->boolean('active');
8              $table->timestamps();
9          });
10     }
11     public function down()
12     {
13         Schema::drop('posts');
14     }

```

Para crear un nuevo Model Factory vamos al archivo *database/factories/ModelFactory.php*

```
1 $factory->defineAs(App\Post::class, 'active', function ($faker) {
2     return [
3         'title' => $faker->sentence(),
4         'body'  => $faker->text(),
5         'active' => true,
6     ];
7 });
8 $factory->defineAs(App\Post::class, 'inactive', function ($faker) {
9     return [
10        'title' => $faker->sentence(),
11        'body'  => $faker->text(),
12        'active' => false,
13    ];
14 });
```































En este caso hemos creado dos tipos (active, inactive), ahora podemos agregar al archivo *database/seeds/DatabaseSeeder.php*

```
1 public function run()
2 {
3     Model::unguard();
4     $post = factory('App\Post', 'active', 5)->create();
5     $post = factory('App\Post', 'inactive', 5)->create();
6     Model::reguard();
7 }
```

Finalmente para verificar que todo funcione ejecutamos la migración y el seeder con

```
1 $ php artisan migrate --seed
```

Ahora si verificamos la base de datos vemos que se han creado 10 registros para la tabla Posts, 5 activos y 5 inactivos.

←T→			id	title	body	active	created_at	updated_at	
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1	Cupiditate dolores perspiciatis maxime dolorum vel...	Omnis quia molestias voluptas debitis id. Mollitia...	1	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2	Ipsum velit omnis aut dignissimos ea esse porro.	Perferendis placeat quo ex. Facilis porro doloribu...	1	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3	Quod ducimus sint eum asperiores.	Et qui atque ut magnam. Et et dolorum est quasi re...	1	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4	Architecto optio libero itaque consequatur omnis m...	Tempora rerum voluptatem dicta amet possimus et. Q...	1	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	5	Magni temporibus quia nihil qui dicta nihil.	Nisi ullam expedita error numquam expedita ducimus...	1	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	6	Voluptatibus quaerat maxime veniam nobis qui vel d...	Provident laboriosam nulla et ut pariatut hic. Aut...	0	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	7	Id a velit nemo natus.	Expedita doloremque sunt adipisci dolores minus ul...	0	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	8	Quas voluptatem et quia.	Non non consequatur at excepturi. Ut quo repellat ...	0	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	9	Quisquam voluptatem enim est deleniti quis quis du...	Atque quos omnis iste. Aperiam quia totam asperior...	0	2015-06-19 15:29:02	2015-06-19 15:29:02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	10	Tempora omnis eum est neque reprehenderit earum su...	Reprehenderit officiis qui esse. Non ullam similiq...	0	2015-06-19 15:29:02	2015-06-19 15:29:02

Como puedes ver es una forma muy sencilla de agregar información a tu base de datos, sin el uso de componentes externos.