Factory Relationships (Has Many)

> If you're not using Laravel 8+ just skip this guide

In the previous lesson we've moved `Company::factory()` as well as `Contact::factory()` from `CompaniesTableSeeder` and `ContactsTableSeeder` to `DatabaseSeeder`. Our `run` method now looks as follow.

```
class DatabaseSeeder extends Seeder
{
    public function run()
    {
        Company::factory()->count(10)->create();
        Contact::factory()->count(50)->create();
    }
}
```

Since `Company` model has `hasMany` relationship to `Contact` model we can utilize Factory relationship which simplified above code.

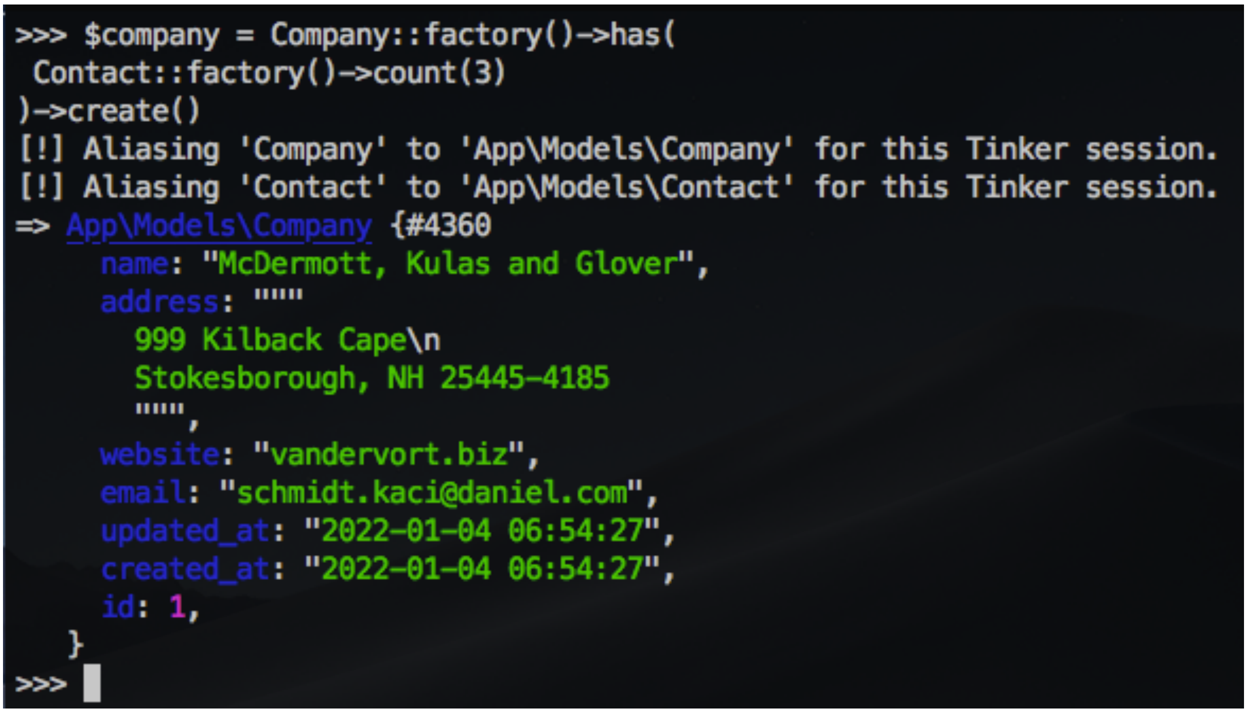
Open your terminal, then enter tinker by saying:

```
php artisan tinker
```

Let's say we want to create a company that has 3 contacts. Here we can use `has` method that provided by the Laravel's factories. The `has` method receives a factory instance, in this case we can specify the Contact factory instance.

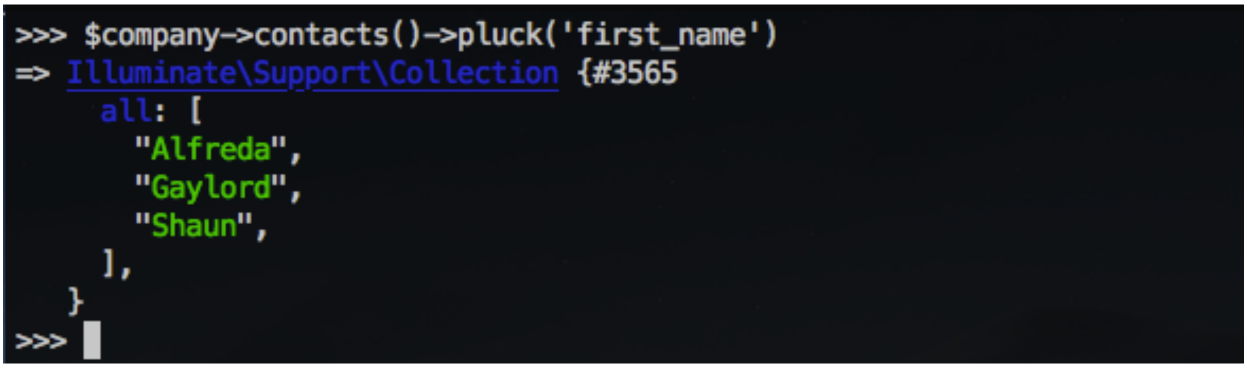
```
$company = Company::factory()->has(
  Contact::factory()->count(3)
)->create()
```

If we execute above code, we'll see a company instance printed out.



```
>>> $company = Company::factory()->has(
 Contact::factory()->count(3)
)->create()
[!] Aliasing 'Company' to 'App\Models\Company' for this Tinker session.
[!] Aliasing 'Contact' to 'App\Models\Contact' for this Tinker session.
=> App\Models\Company {#4360
     name: "McDermott, Kulas and Glover",
     address: """
       999 Kilback Cape\n
       Stokesborough, NH 25445-4185
       """,
     website: "vandervort.biz",
     email: "schmidt.kaci@daniel.com",
     updated_at: "2022-01-04 06:54:27",
     created_at: "2022-01-04 06:54:27",
     id: 1,
   }
>>> 
```

If we now see its contacts, we'll get 3 contacts.



```
>>> $company->contacts()->pluck('first_name')
=> Illuminate\Support\Collection {#3565
     all: [
       "Alfreda",
       "Gaylord",
       "Shaun",
     ],
   }
>>> 
```

Now, to create let's say 10 companies, and for each company we want to generate 3 contacts, we can call `count` method before `create` method call and specify the number.

```
Company::factory()->has(
   Contact::factory()->count(3)
)
   ->count(10)
   ->create()
```

We can make above code much simpler by using magic method `has[relationship]` like so.

```
Company::factory()->hasContacts(3)->count(10)->create()
```

With Factory relationship our `DatabaseSeeder` now looks like this.

```php
class DatabaseSeeder extends Seeder
{
    public function run()
    {
        Company::factory()->hasContacts(5)->count(10)->create();
    }
}
```

Please check out this link to get more information about Factory Relationships.

https://laravel.com/docs/8.x/database-testing#factory-relationships