How to add a custom field to the address forms?

This short tutorial will walk you through adding an address line 3 field to all of the address forms (found in the admin, account-area, and checkout).

Migrations

The first step is to add your field to all of the address tables so that it can be stored. To do this we'll create and register a migration that adds a `line_3` field to the `addresses`, `order_addresses`, and `fulfillment_addresses` tables.

Migration Code

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class AddLine3ToAddressTables extends Migration
{
   /**
    * Run the migrations.
    *
    * @return void
    */
   public function up()
   {
       Schema::table('addresses', function (Blueprint $table) {
           $table->string('line_3')->nullable()->after('line_2');
       });
```

```
        Schema::table('order_addresses', function (Blueprint $table) {
            $table->string('line_3')->nullable()->after('line_2');
        });

        Schema::table('fulfillment_addresses', function (Blueprint $table) {
            $table->string('line_3')->nullable()->after('line_2');
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::table('fulfillment_addresses', function (Blueprint $table) {
            $table->dropColumn('line_3');
        });

        Schema::table('order_addresses', function (Blueprint $table) {
            $table->dropColumn('line_3');
        });

        Schema::table('addresses', function (Blueprint $table) {
            $table->dropColumn('line_3');
        });
    }
}
```

Service Provider Code

```
<?php

namespace Acme\MyModule;

use Aero\Common\Providers\ModuleServiceProvider;
```

```
class ServiceProvider extends ModuleServiceProvider
{
   public function setup()
   {
       if ($this->app->runningInConsole()) {
           $this->loadMigrationsFrom(__DIR__.'/../database/migrations');
       }
   }
}
```

Views

The next step is to create a view for the line 3 field. To achieve this we'll create and register a Twig view for the field. The view will include the `forms::components.input` view to get a simple input field.

View Code

```
{% include "forms::components.input" with {
   type: 'text',
   error: errors.first((inputName ?? group) ~ '.line_3'),
   half: false,
   label: 'Address Line 3 (Optional)',
   class: (inputName ?? group) ~ '-' ~ 'line-3',
   name: (inputName ?? group) ~ '[line_3]',
   value: old((inputName ?? group) ~ '.line_3', address.line_3),
   autocomplete: type ? type ~ ' line_3' : 'line_3',
   required: true
} only %}
```

Service Provider Code

```
<?php

namespace Acme\MyModule;

use Aero\Common\Providers\ModuleServiceProvider;
```

```
class ServiceProvider extends ModuleServiceProvider
{
   public function setup()
   {
       if ($this->app->runningInConsole()) {
           $this->loadMigrationsFrom(__DIR__.'/../database/migrations');
       }

       $this->loadViewsFrom(__DIR__.'/../resources/views', 'my-module');
   }
}
```

Adding the Field to the Forms

Now we need to add the field to the address forms. To do this we'll make the field fillable and add it to the validation requests using the `Aero\Common\Helpers\Address::addField()` helper method. After that we'll extend `Aero\Forms\AddressForm` to add the field to the frontend address forms (checkout and account-area) and `Aero\Admin\Http\Forms\AdminAddressForm` to add the field to the admin address forms.

```
<?php

namespace Acme\MyModule;

use Aero\Admin\Http\Forms\AdminAddressForm;
use Aero\Common\Helpers\Address;
use Aero\Common\Providers\ModuleServiceProvider;
use Aero\Forms\AddressForm;

class ServiceProvider extends ModuleServiceProvider
{
   public function setup()
   {
       if ($this->app->runningInConsole()) {
           $this->loadMigrationsFrom(__DIR__.'/../database/migrations');
       }
```

```
        $this->loadViewsFrom(__DIR__.'/../resources/views', 'my-module');

        Address::addField('line_3');

        AddressForm::extend(function ($form) {
            $form->addSectionAfter('line3', 'my-module::line-3-field', 'line2');
        });

        AdminAddressForm::extend(function ($form) {
            $form->addSectionAfter('line3', 'my-module::line-3-field', 'line2');
        });
    }
}
```

Adding the Field to the Rendered Addresses

This step is optional and will add the line 3 field to the address when it's rendered (such as when viewing an order or viewing your addresses in the account area). To do this we'll extend `Aero\Address\Pipelines\AddressFormatter` and `Aero\Address\Pipelines\AddressStringFormatter`.

```
<?php

namespace Acme\MyModule;

use Aero\Address\Pipelines\AddressFormatter;
use Aero\Address\Pipelines\AddressStringFormatter;
use Aero\Admin\Http\Forms\AdminAddressForm;
use Aero\Common\Helpers\Address;
use Aero\Common\Providers\ModuleServiceProvider;
use Aero\Forms\AddressForm;

class ServiceProvider extends ModuleServiceProvider
{
    public function setup()
    {
        if ($this->app->runningInConsole()) {
            $this->loadMigrationsFrom(__DIR__.'/../database/migrations');
```

```
        }

        $this->loadViewsFrom(__DIR__.'/../resources/views', 'my-module');

        Address::addField('line_3');

        AddressForm::extend(function ($form) {
            $form->addSectionAfter('line3', 'my-module::line-3-field', 'line2');
        });

        AdminAddressForm::extend(function ($form) {
            $form->addSectionAfter('line3', 'my-module::line-3-field', 'line2');
        });

        AddressFormatter::extend(function ($formatter) {
            $formatter->putAfter('line3', $formatter->fields['line_3'], 'line2');
        });

        AddressStringFormatter::extend(function ($formatter) {
            $formatter->putAfter('line3', $formatter->fields['line_3'], 'line2');
        });
    }
}
```

Revision #1
Created 2026-09-02 13:27:56 UTC by Michael Price
Updated 2026-09-02 13:27:56 UTC by Michael Price