How do I create a custom dashboard lens?

To create a dashboard lens you need to create a class that extends `Aero\Admin\AdminLens` and implements the **data** method.

data Method

This method is executed every time the lens needs data (when the dashboard loads or when the dashboard date range is changed). This method must return an **Illuminate\Http\JsonResponse** because it is called on the frontend by javascript.

data Method Parameters

Type	Description
Request	The first parameter is the request.
Array	The second parameter is an array that holds the start and end date selected for the dashboard stats. This array has a start and end key that has the dates.
Array	The third parameter is an array that holds information about the comparison dates. This array has a type key that is 1 for the same period prior, 2 for the same period 1 year ago, or 3 for a custom period. This array has a date key that has the same structure as the second parameter. This array also has a text key that holds text to be displayed to explain the comparison.

Here is the code for our top shipping countries implementation:

```
<?php

namespace Aero\Admin\Lenses;

use Aero\Admin\AdminLens;
use Aero\Cart\Models\Order;
use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;
```

```
use Illuminate\Support\Str;

class TopShippingCountriesLens extends AdminLens
{
   protected $title = 'Top Shipping Countries';

   protected static $permission = 'reports.orders';

   protected static $containerClass = 'row-span-2';

   protected $view = 'admin::lenses.percentage-list';

   public function data(Request $request, array $date, array $compare): JsonResponse
   {
       $countries = Order::visible()->with('shippingAddress.country')->whereHas('shippingAddress')
           ->whereBetween('ordered_at', [$date['start'], $date['end']])->get()
           ->groupBy('shippingAddress.country.name')
           ->map(function ($group, $key) {
               return [
                   'name' => $key,
                   'count' => $group->count(),
                   'percentage' => 0,
               ];
           })->sortByDesc('count')->take(5);

       $total = $countries->reduce(function ($count, $country) {
           return $count + $country['count'];
       }, 0);

       $countries->transform(function ($country) use ($total) {
           $country['value'] = number_format(($country['count'] / $total) * 100, 2);
           $country['text'] = $country['count'].' '.Str::plural('order', $country['count']);
           $country['percentage'] = $country['value'].'%';

           return $country;
       });

       return response()->json([
           'items' => $countries,
       ]);
```

```
    }
}
```

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