How do I add custom data to a transformer?

To add custom data you need to use the static **add()**method on the relevant transformer. This method expects a closure that will return an array of the field(s) to merge into the transformers array. The closure will be passed an array of data that you can use.

This code example adds a **customField**that will have the value of the products ID multiplied by two. This field will then be usable on the product new/edit page directly in Vue like **product.customField**. It’s important to note that this is used on the new product page where the product will not have an ID and some other data which is why the **?? 0**is used.

```
<?php

namespace Acme\MyModule;

use Aero\Admin\Transformers\ProductTransformer;
use Aero\Common\Providers\ModuleServiceProvider;

class ServiceProvider extends ModuleServiceProvider
{
   public function setup()
   {
       ProductTransformer::add(function ($data) {
           /* @var \Aero\Catalog\Models\Product */
           $product = $data['product'];

           return [
               'customField' => ($product->id ?? 0) * 2,
           ];
       });
   }
}
```

Casts

When the admin page fails validation and redirects back with the old data, the old data is casted to ensure it’s correct using the transformer. This is particularly important for booleans where a checkbox may set the value as 1 and it therefore should be casted to true to avoid problems.

Available Cast Types

You can currently use the following casts: boolean, array, string, and integer.

Adding a Cast

To add a cast you need to use the static **addCast()**method on the relevant transformer. This method expects two parameters, the key of the field and the value that the field should be casted to.

This code example ensures the **customField**is casted as an **integer**.

```
<?php

namespace Acme\MyModule;

use Aero\Admin\Transformers\ProductTransformer;
use Aero\Common\Providers\ModuleServiceProvider;

class ServiceProvider extends ModuleServiceProvider
{
   public function setup()
   {
       ProductTransformer::addCast('customField', 'integer');

       ProductTransformer::add(function ($data) {
           /* @var \Aero\Catalog\Models\Product */
           $product = $data['product'];

           return [
               'customField' => ($product->id ?? 0) * 2,
           ];
       });
   }
```

```
}
```

Revision #1
Created 2026-09-02 13:27:31 UTC by Michael Price
Updated 2026-09-02 13:27:31 UTC by Michael Price