

Translations

Translations

Anonymous · 08/10/2026

Translations

The exceptions of `derafu/etl` can be translated. This page explains what is translatable, what ships out of the box and how to activate it in your application.

What's Translatable

Every exception the package throws is translatable (see [derafu/translation](#)):

- `Derafu\ETL\Exception\ETLException`, and so the exceptions of the pipeline.
- The errors of the databases, the schema sources and the schema targets, which use `TranslatableRuntimeException` and `TranslatableInvalidArgumentException`.

Messages are written in ICU MessageFormat, with the English text as the translation id and named placeholders:

```
use Derafu\ETL\Database\DatabaseManager;

try {
    (new DatabaseManager())->connect('/no/such/dir/data.xlsx', ['createIfNotExists' =>
false]);
} catch (\Throwable $e) {
    echo $e->getMessage();
    // "File does not exist and creation is disabled for this file:
/no/such/dir/data.xlsx"
}
```

Without a translator configured, `getMessage()` already returns the formatted English text, so the package works as before whether or not you set up translation.

The command line is not translated

The `derafu:etl` command (its description, the help of its arguments and options, and its output) is written in English on purpose, like the commands of Symfony: it is a tool for whoever administers the application, not text for end users.

What Ships

- `resources/translations/errors+intl-icu.es.php`: a Spanish translation for every message of the package.
- `Derafu\ETL\Translation\EtlTranslationResourceProvider`: a `TranslationResourceProviderInterface` pointing at that directory.
- `resources/config/etl-services.yaml`: a recipe for applications that use `symfony/dependency-injection`.

None of them does anything by itself: `derafu/etl` is a library, it does not build a `Translator`.

Activating It (Plain PHP)

```
use Derafu\ETL\Translation\EtlTranslationResourceProvider;
use Derafu\Translation\TranslatorFactory;

$translator = TranslatorFactory::create(
    defaultLocale: 'es',
    fallbackLocales: ['es', 'en'],
    resourceProviders: [new EtlTranslationResourceProvider()],
);
```

Activating It (Dependency Injection)

```
imports:
  - { resource: '../vendor/derafu/translation/resources/config/translation-services.yaml' }
  - { resource: '../vendor/derafu/etl/resources/config/etl-services.yaml' }

services:
  Symfony\Component\Translation\Translator:
    factory: ['Derafu\Translation\TranslatorFactory', 'create']
    arguments:
      $defaultLocale: 'es'
      $fallbackLocales: ['es', 'en']
      $resourceProviders: !tagged_iterator derafu_translation.resource_provider
```

`etl-services.yaml` tags `EtlTranslationResourceProvider` with `derafu_translation.resource_provider`, so it is picked up automatically.

Checking It in Tests

The package checks its own messages with the lint of `derafu/translation` (`Derafu\Translation\Lint\TranslationAudit`), which reads the code. A new message without an

entry in the catalogue, an entry nobody uses, or an exception that is not translatable fails the test instead of showing in English when the error happens:

```
$report = (new TranslationAudit())->audit(
    __DIR__ . '/../../src',
    new EtlTranslationResourceProvider()
);

$this->assertFalse($report->nothingFound);
$this->assertSame([], $report->describe($report->missingTranslations));
```

See [the lint guide](#) for the full report. It needs `nikic/php-parser`, listed in `require-dev` of the package.

Last updated on 08/10/2026