

Services

The services this package provides, their interfaces, exceptions, and how to wire them with Symfony's dependency injection

Anonymous · 12/10/2026

Services

Available Services

Interface	Class	Purpose
<code>Contract\SerializerInterface</code>	<code>Service\Serializer</code>	Turn a document into a json/yaml/xml string.
<code>Contract\DeserializerInterface</code>	<code>Service\Deserializer</code>	Turn a json/yaml/xml string into a document.
<code>Contract\ValidatorInterface</code>	<code>Service\SchemaValidator</code>	Validate a document's XML against the official XSD.
<code>Contract\FormatHandlerFactoryInterface</code>	<code>Service\FormatHandlerFactory</code>	Resolve a format name to the handler that implements it; register a custom one.

Every class is namespaced under `Derafu\Ubl\`. See **Formats** for `Serializer/Deserializer` usage and **Validation** for `SchemaValidator`.

Exceptions

Every exception this package throws extends `Derafu\Ubl\Exception\UblException`, itself a `Derafu\Translation\Exception\Core\TranslatableException`:

Exception	Thrown when
<code>UnsupportedFormatException</code>	A format name is not <code>json</code> , <code>yaml</code> or <code>xml</code> (or a custom registered one).
<code>SerializationException</code>	A document cannot be turned into the requested format.
<code>DeserializationException</code>	Content cannot be turned into a document (invalid JSON/YAML, an unknown document type, ...).
<code>ValidationException</code>	A document's XML does not validate against the schema. Carries the real <code>LibXML_Error</code> list, see Validation .

Catching the base `UblException` catches any of them:

```
use Derafu\Ubl\Exception\UblException;

try {
    $document = (new Deserializer())->deserialize($content, 'yaml');
} catch (UblException $e) {
    // SerializationException, DeserializationException, ValidationException, ...
}
```

Dependency Injection

For an application using `symfony/dependency-injection`, `resources/config/ubl-services.yaml` wires every service above to its interface, ready to autowire:

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

With that imported, a service can simply type-hint the interface it needs:

```
use Derafu\Ubl\Contract\SerializerInterface;

final class QuotationController
{
    public function __construct(
        private readonly SerializerInterface $serializer,
    ) {
    }
}
```

Last updated on 12/10/2026