

Captcha

The captcha of the forms of derafu/form, with the provider of the application

Anonymous · 08/10/2026

Captcha

A form can ask for a **captcha**: the form renders a widget, and when its data is processed what the visitor solved is checked. `derafu/form` does not know which captcha service it is, nor its keys: that is what a **captcha provider** knows, and the application gives one (for example the ones of [derafu/captcha](#)).

The option of the form

It is an option of the form, in the `options` of its definition, next to `csrf_protection` and `translation_domain`:

```
# resources/forms/contact.form.yaml
options:
  captcha_protection: true
```

Value	What happens
absent	The form is not protected with the captcha. Nothing is rendered or checked, even if the application has a provider.
false	The same, written down.
true	The form is protected with the captcha , always. With a provider available the widget is rendered and the data is checked, and the form is not valid if the visitor did not solve it, or if the service could not be asked (it fails closed). If the application did not configure a captcha, it is an error : the form can not be rendered nor processed, and the message says what to configure.

A form is never left open because nobody thought of the captcha. A package that ships forms (a contact form, a login) declares `captcha_protection: true`, and the application configures a captcha once; no variable per form is needed. The simplest one needs no account: `ALTCHA`, with `CAPTCHA_PROVIDER=altcha` and a secret key (see [derafu/captcha](#)).

The application can also decide, **on purpose**, not to have a captcha (`CAPTCHA_PROVIDER=none` in `derafu/captcha`): then the forms that are protected have none and there is no error. It is not the same as not configuring anything, which fails.

The provider

```
namespace Derafu\Form\Contract\Captcha;

interface CaptchaProviderInterface
{
    // Whether the application has configured a captcha.
    public function isAvailable(): bool;

    // Whether the application decided, on purpose, not to have one.
    public function isDisabled(): bool;

    // The name of the field that the widget writes in the form.
    public function getResponseField(): string;

    // The HTML of the widget, with its scripts.
    public function getWidget(string $formId): string;

    // Whether what the visitor solved is the proof of a person for that form.
    // It throws CaptchaUnavailableException if the service could not be asked.
    public function verify(string $token, string $formId): bool;
}
```

The id of the form is the name of its schema, or `form` if it has none (`$form->getId()`): the services that have an action (reCAPTCHA v3, Turnstile) use it, so what was solved for a form is not good for another.

Rendering

`form_captcha(form)` renders the widget. `form.html.twig` and `renderBody` call it, after the fields and before the CSRF token, so a form that is rendered whole has it. A template that builds the form by hand writes it where it wants, before the button:

```
{{ form_start(form) }}
    {{ form_element(form.uiSchema, form) }}
    {{ form_captcha(form) }}
    {{ form_csrf(form) }}
    <button type="submit">Send</button>
{{ form_end(form) }}
```

It renders nothing when the form is not protected with the captcha or the application disabled it on purpose.

Processing

`FormDataProcessor` checks the captcha according to the option of the form, **never according to what arrives in the POST**: a request that is made by hand without the field is not valid.

- What the visitor solved is read from the raw data, in the field of the provider, and it is **not part of the processed data**.
- The service is asked last: after the CSRF token and after the fields, and only if they are valid, so it is not asked about a form that is not going to be accepted.
- When it is not valid, the result is not valid and has a message in the **errors of the form** (`getFormErrors()`). There are two messages: “The captcha is not valid. Try again.” and “The captcha could not be verified. Try again in a moment.” (the service did not answer). Both are translatable, in the domain `errors`.

Wiring

With the services of the package (`form-services.yaml`) the provider is **optional**: if the application has a service that implements `CaptchaProviderInterface`, the renderer and the processor use it. Without one, the forms that are protected with the captcha fail. When they are created by hand, it is given in the options:

```
$renderer = FormRendererFactory::create(['captcha_provider' => $provider]);

$processor = new FormDataProcessor($resolver, $dataProcessor, captchaProvider:
    $provider);
```

Last updated on 08/10/2026