
Django reCAPTCHA

Django reCAPTCHA form field/widget integration app.

downloads 205k/month downloads 205k/month coverage 99% coverage 99%

[!NOTE] django-recaptcha supports Google reCAPTCHA V2 - Checkbox (Default), Google reCAPTCHA V2 - Invisible and Google reCAPTCHA V3. Please look at the widgets section for more information.

Django reCAPTCHA uses a modified version of the Python reCAPTCHA client which is included in the package as `client.py`.

- Requirements
- Installation
- Usage
 - Fields
 - Widgets
 - reCAPTCHA V3 Score
 - reCAPTCHA V3 Action
 - Local Development and Functional Testing
- Credits

Requirements

Tested with:

- Python: 3.8, 3.9, 3.10, 3.11, 3.12
- Django: 3.2, 4.1, 4.2, 5.0
- You can view the Python-Django support matrix [here](#)

This package only supports modern, “evergreen” desktop and mobile browsers. For IE11 support, make sure to add a polyfill for `Element.closest`.

Installation

1. Sign up for reCAPTCHA.
2. Install with `pip install django-recaptcha`.
3. Add `'django_recaptcha'` to your `INSTALLED_APPS` setting.

```
1 INSTALLED_APPS = [  
2     ...,  
3     'django_recaptcha',  
4     ...,  
5 ]
```

4. Add the Google reCAPTCHA keys generated in step 1 to your Django production settings with `RECAPTCHA_PUBLIC_KEY` and `RECAPTCHA_PRIVATE_KEY`. Note that omitting these settings will default to a set of test keys refer to Local Development and Functional Testing for more information.

For example:

```
1 RECAPTCHA_PUBLIC_KEY = 'MyRecaptchaKey123'  
2 RECAPTCHA_PRIVATE_KEY = 'MyRecaptchaPrivateKey456'
```

These can also be specified per field by passing the `public_key` or `private_key` parameters to `ReCaptchaField` - see field usage below.

5. (OPTIONAL) If you require a proxy, add a `RECAPTCHA_PROXY` setting (dictionary of proxies), for example:

```
1 RECAPTCHA_PROXY = {'http': 'http://127.0.0.1:8000', 'https': 'https  
://127.0.0.1:8000'}
```

6. (OPTIONAL) In the event `www.google.com` is not accessible the `RECAPTCHA_DOMAIN` setting can be changed to `www.recaptcha.net` as per the reCAPTCHA FAQ:

```
1 RECAPTCHA_DOMAIN = 'www.recaptcha.net'
```

This will change the Google JavaScript api domain as well as the client side field verification domain.

Usage

Fields

The quickest way to add reCAPTCHA to a form is to use the included `ReCaptchaField` field class. A `ReCaptchaV2Checkbox` will be rendered by default. For example:

```
1 from django import forms  
2 from django_recaptcha.fields import ReCaptchaField  
3  
4 class FormWithCaptcha(forms.Form):
```

```
5 captcha = ReCaptchaField()
```

Be sure to include the captcha field in your forms. There are many ways to add fields to forms in Django. We recommend you refer to the form rendering options and rendering fields manually sections of the official Django documentation for forms.

To allow for runtime specification of keys you can optionally pass the `private_key` or `public_key` parameters to the constructor. For example:

```
1 captcha = ReCaptchaField(  
2     public_key='76wtgdfsjsydt7r5FFGFhgsdfytd656sad75fgh',  
3     private_key='98dfg6df7g56df6gdfgdfg65JHJH656565GFGFGs',  
4 )
```

If specified, these parameters will be used instead of your reCAPTCHA project settings.

Widgets

There are three widgets that can be used with the `ReCaptchaField` class:

- `ReCaptchaV2Checkbox` for Google reCAPTCHA V2 - Checkbox
- `ReCaptchaV2Invisible` for Google reCAPTCHA V2 - Invisible
- `ReCaptchaV3` for Google reCAPTCHA V3

To make use of widgets other than the default Google reCAPTCHA V2 - Checkbox widget, simply replace the `ReCaptchaField` widget. For example:

```
1 from django import forms  
2 from django_recaptcha.fields import ReCaptchaField  
3 from django_recaptcha.widgets import ReCaptchaV2Invisible  
4  
5 class FormWithCaptcha(forms.Form):  
6     captcha = ReCaptchaField(widget=ReCaptchaV2Invisible)
```

The reCAPTCHA widget supports several data attributes that customize the behaviour of the widget, such as `data-theme`, `data-size`, etc. You can forward these options to the widget by passing an `attrs` parameter to the widget, containing a dictionary of options. For example:

```
1 captcha = fields.ReCaptchaField(  
2     widget=widgets.ReCaptchaV2Checkbox(  
3         attrs={  
4             'data-theme': 'dark',  
5             'data-size': 'compact',  
6         }  
7     )  
8 )
```

```
9 # The ReCaptchaV2Invisible widget
10 # ignores the "data-size" attribute in favor of 'data-size="invisible"'
```

The reCAPTCHA api supports several parameters. To customise the parameters that get sent along pass an `api_params` parameter to the widget, containing a dictionary of options. For example:

```
1 captcha = fields.ReCaptchaField(
2     widget=widgets.ReCaptchaV2Checkbox(
3         api_params={'hl': 'cl', 'onload': 'onLoadFunc'}
4     )
5 )
6 # The dictionary is urlencoded and appended to the reCAPTCHA api url.
```

By default, the widgets provided only supports a single form with a single widget on each page.

The language can be set with the 'hl' parameter, look at language codes for the language code options. Note that translations need to be added to this package for the errors to be shown correctly. Currently the package has error translations for the following language codes: es, fr, nl, pl, pt_BR, ru, zh_CN, zh_TW

However, the JavaScript used by the widgets can easily be overridden in the templates.

The templates are located in:

- `django_recaptcha/includes/js_v2_checkbox.html` for overriding the reCAPTCHA V2 - Checkbox template
- `django_recaptcha/includes/js_v2_invisible.html` for overriding the reCAPTCHA V2 - Invisible template
- `django_recaptcha/includes/js_v3.html` for overriding the reCAPTCHA V3 template

For more information about overriding templates look at Django's template override

reCAPTCHA V3 Score

As of version 3, reCAPTCHA also returns a score value. This can be used to determine the likelihood of the page interaction being a bot. See the Google documentation for more details.

To set a project wide score limit use the `RECAPTCHA_REQUIRED_SCORE` setting.

For example:

```
1 RECAPTCHA_REQUIRED_SCORE = 0.85
```

For per field, runtime, specification the score can also be passed to the widget:

```
1 captcha = fields.ReCaptchaField(
```

```
2     widget=ReCaptchaV3(required_score=0.85)
3 )
```

In the event the score does not meet the requirements, the field validation will fail as expected and an error message will be logged.

reCAPTCHA V3 Action

Google's reCAPTCHA V3 API supports passing an action value. Actions allow you to tie reCAPTCHA validations to a specific form on your site for analytical purposes, enabling you to perform risk analysis per form. This will allow you to make informed decisions about adjusting the score threshold for certain forms because abusive behavior can vary depending on the nature of the form.

To set the action value, pass an `action` argument when instantiating the ReCaptcha widget. Be careful to only use alphanumeric characters, slashes and underscores as stated in the reCAPTCHA documentation.

```
1 captcha = fields.ReCaptchaField(
2     widget=widgets.ReCaptchaV3(
3         action='signup'
4     )
5 )
```

Setting an action is entirely optional. If you don't specify an action, no action will be passed to the reCAPTCHA V3 API.

Local Development and Functional Testing

If `RECAPTCHA_PUBLIC_KEY` and `RECAPTCHA_PRIVATE_KEY` are not set, django-recaptcha will use Google's test keys instead. These cannot be used in production since they always validate to true and a warning will be shown on the reCAPTCHA. Google's test keys only work for reCAPTCHA version 2.

To bypass the security check that prevents the test keys from being used unknowingly add `SILENCED_SYSTEM_CHECKS = [..., 'django_recaptcha.recaptcha_test_key_error', ...]` to your settings, here is an example:

```
1 SILENCED_SYSTEM_CHECKS = ['django_recaptcha.recaptcha_test_key_error']
```

If you want to mock the call to Google's servers altogether, have a look at `test_fields.py`:

```
1 from unittest.mock import patch
2 from django.test import TestCase
```

```
3 from django_recaptcha.client import RecaptchaResponse
4
5 class TestFields(TestCase):
6     @patch("django_recaptcha.fields.client.submit")
7     def test_client_success_response(self, mocked_submit):
8         mocked_submit.return_value = RecaptchaResponse(is_valid=True)
9         ...
```

Credits

Originally developed by Praekelt Consulting.

Inspired Marco Fucci's blogpost titled Integrating reCAPTCHA with Django.

`client.py` taken from `recaptcha-client` licensed MIT/X11 by Mike Crawford.

reCAPTCHA copyright 2012 Google.