CEPTCHA Administrator Guide

Introduction

For now, CEPTCHA is in beta. You can already test it on tour website and provide us with feedback!

First of all, make sure you have registered as a beta user and you have obtained an API key. The service will only work for the domain(s) you requested.

The service works as follows:

- A user tries to submit a form
- The CEPTCHA shows
- The users selects the right punchline
- The form is submitted (to your server)
- Your server makes a request to the CEPTCHA service to validate the code
    - When the code is valid: You should continue as normal, i.e. send the contactforum
    - When the code is invalid: It's up to you want you want to do, let the user try again, laugh really hard, or try to talk in computerlanguage

Disclaimer

We try to do our best, but robots can learn, so it's not 100% foolproof, it's not our responsibility when there are false positives or false negatives.

The client side

Just before </body> add this line of code:

```
<script src="https://service.ceptcha.com/ceptcha.js"></script>
```

That's it, now Ceptcha will be enabled on all forms.

Content-Security-Policy

If you use a Content-Security-Policy (you should!) then your website won't load the ceptcha.js file, you need to add the service URL to default-src, img-src, style-src, connect-src

- **default-src** *(for loading the javascript file)*
- **img-src** *(for loading the CEPTCHA logo)*
- **style-src** *(for loading the stylesheet)*
- **connect-src** (f*or connecting to the service)*

Disable CEPTCHA on a form

When you want to disable CEPTCHA on some forms, just add a custom data property to the form:

```
<form data-ceptcha="false">
```

Language

At the moment the jokes are available in English and Dutch. There are several ways to change the language.

Using the html5 language on the page:

```
<html lang="nl">
```

Using the html5 form language:

```
<form lang="nl">
```

Using a custom data property on the form:

```
<form data-ceptcha-lang="nl">
```

The server side

When you receive a form from a user, you will receive an additional field with the name CEPTCHA this is the token you need to send back to the service.

When this field is missing, you can be sure this is spam, the CEPTCHA form was never displayed or never answered.

Make a POST request to https://service.ceptcha.com/ and send a JSON body with your API key and the CEPTCHA token:

```
	{
		key: "my-super-secret-key",
		token: "the-token-i-received"
	}
```

You will get a 200 status back when everything is fine with a JSON response:

```
{
	valid: true
}
```

When valid is true, the user selected the right punch line and is not a known spammer
When valid is false, the user selected the wrong punch line or is a known spammer

When something is going wrong you will get a 500 error back with an error message what went wrong:

```
{
	error: "Request expired",
	valid: false
}
```

Example NodeJS

An example validation using NodeJS, Express and Axios

```
// set up
========================================================================
const serviceURL    = 'https://service.ceptcha.com';
const websitePORT   = 3000;
const apiKey        = 'my-super-secret-key';

const express       = require('express');
const app           = express();
const bodyParser    = require('body-parser');
const axios         = require('axios');

app.enable('strict routing');
app.use(bodyParser.urlencoded({extended: true}));

app.use(function (req, res, next) {
    res.setHeader(
        'Content-Security-Policy',
                `default-src 'self'; script-src 'self' ${serviceURL};
img-src 'self' ${serviceURL}; font-src 'self' data:; style-src 'self' $
{serviceURL} 'unsafe-inline' ${serviceURL}; connect-src ${serviceURL};`
    );
    next();
});

// Just an ugly sample form
app.get('/', (req, res) => {
    const form = `<!DOCTYPE html>
<html lang="en">
<head>
    <title>Ceptcha Example</title>
</head>
<body>
    <form method="post">
        <input type="email" name="email" placeholder="Your Email"
autocomplete="on" required>
        <button type="submit">Submit</button>
    </form>

    <script src="https://service.ceptcha.com/ceptcha.js"></script>
</body>
</html>`;

    res.status(200).send(form);
});

// Check all posts for a valid Ceptcha
app.post('*', (req, res, next) => {

    if (!req.body || !req.body.ceptcha) {
        res.status(401).send('Invalid Ceptcha');
        return;
    }
    const token = req.body.ceptcha;
    // Don't need to keep the Ceptcha in the body
    delete req.body.ceptcha;

    // Make a POST request to the service
    axios.post(serviceURL+'/validate', {
            key: apiKey,
            token: token
        })
        .then(function(response) {
            if (response.data && response.data.valid) {
                // Yes, the Ceptcha is valid, just continue to the next
route
                next();
            } else {
                // Nope, the Ceptcha was invalid
                res.status(200).send("No spam allowed");
            }

        })
        .catch(function(error) {
            if (error.response && error.response.data &&
error.response.data.error) {
                // Got an error from the service
                res.status(401).send(error.response.data.error);
            } else {
                // Got an other error (network?)
                res.status(error.response.status ||
500).send(error.message);
            }
        });

});

// Here you can handle the normal POST request
app.post('/', (req, res) => {
    res.status(200).send('Yay, it works!');
});

// launch
======================================================================
app.listen(websitePORT, () => {
    console.log(`Server started on port ${websitePORT}`)
})
```