


CaptureBarriereFrei - Barrierefreier Formulare mit Bot-Schutz

 CaptureBarriereFrei Logo (Optional: Logo hinzufügen)

Übersicht

CaptureBarriereFrei ist eine moderne, barrierefreie Alternative zu herkömmlichen CAPTCHA-Systemen, die Formulare vor automatisierten Bot-Zugriffen schützt und gleichzeitig vollständige Barrierefreiheit gewährleistet. Im Gegensatz zu herkömmlichen CAPTCHAs, die oft erhebliche Zugänglichkeitsprobleme für Menschen mit Behinderungen darstellen, nutzt diese Lösung eine Kombination aus verschiedenen nicht-intrusiven Techniken.

Hauptfunktionen

- **Barrierefreier Bot-Schutz:** Schützt Formulare ohne Beeinträchtigung der Zugänglichkeit
- **Benutzerinteraktions-Analyse:** Erkennt menschliches Verhalten durch Maus-, Tastatur- und Scroll-Interaktionen
- **Honeypot-Technik:** Unsichtbare Felder für Bots als Falle
- **Zeit-Analyse:** Prüfung der Ausfüllzeiten von Formularen
- **Barrierefreie "Ich bin kein Roboter"-Checkbox:** Semantisch korrekt und mit Screenreadern kompatibel
- **Modulares System:** Einfach erweiterbar und anpassbar
- **E-Mail-Versand:** Integrierter E-Mail-Versand mit Fallback-Optionen

Installation

Voraussetzungen

- Webserver mit PHP (für E-Mail-Funktionalität)
- Moderner Browser mit JavaScript-Unterstützung

Einrichtung

1. Projekt-Dateien in Ihr Webverzeichnis kopieren:

```
git clone [repository-url] /path/to/webserver/mailTest  
# oder manuell die Dateien hochladen
```

2. Überprüfen Sie die PHP-Mail-Konfiguration mit mail_diagnose.php:

```
http://localhost/mailTest/mail_diagnose.php
```

3. Passen Sie die E-Mail-Empfängeradresse in der JavaScript-Konfiguration an (index.html):

```
recipient: 'ihre.email@domain.de',
```

Systemarchitektur

CaptureBarriereFrei ist modular aufgebaut und besteht aus folgenden Komponenten:

Frontend-Komponenten

- **Kernmodul (core.js)**: Hauptklasse und Steuerungszentrale
- **Bot-Erkennung (botDetection.js)**: Analyse von Benutzerverhalten
- **Formularschutz (formProtection.js)**: Implementierung von Schutzmaßnahmen
- **UI-Komponenten (ui.js)**: Barrierefreie Benutzeroberfläche
- **Formular-Validierung (formValidation.js)**: Client-seitige Validierung
- **Hilfsfunktionen (utils.js)**: Allgemeine Hilfsfunktionen
- **Logger (logger.js)**: Protokollierung und Debugging

Backend-Komponenten

- **send_mail.php**: Verarbeitet Formularübermittlungen und sendet E-Mails
- **mail_diagnose.php**: Diagnose-Tool für PHP-Mail-Funktionalität

Bot-Erkennung und Schutzmaßnahmen

CaptureBarriereFrei nutzt mehrere Techniken zur Erkennung von Bots:

1. Verhaltensanalyse

Die Bibliothek analysiert natürliches Benutzerverhalten anhand von:

- Mausbewegungen und Klicks
- Tastatureingaben und Timing
- Scrollverhalten
- Interaktionsbalance (Verhältnis zwischen verschiedenen Interaktionsarten)

2. Zeitanalyse

Bots füllen Formulare typischerweise sehr schnell aus. Die Bibliothek:

- Misst die Zeit zwischen Formularanzeige und Absenden
- Vergleicht mit einer konfigurierbaren Mindestzeitspanne
- Wertet zu schnelle Eingaben als verdächtig

3. Honeypot-Felder

Versteckte Felder, die für Menschen unsichtbar sind, aber von Bots ausgefüllt werden:

- Für Screenreader unsichtbar gemacht (aria-hidden="true")
- Visuell aus dem Bildschirm positioniert
- Aus dem Tab-Index entfernt

4. Barrierefreie Verifizierung

Eine zugängliche Alternative zur CAPTCHA-Checkbox:

- Vollständig mit Tastatur bedienbar
- Mit korrekten ARIA-Attributen für Screenreader
- Visuelles Feedback für sehende Benutzer

5. Punktevergabe und Score-Berechnung

Der Bot-Erkennungsmechanismus vergibt Punkte für verschiedene Benutzerinteraktionen. Je höher der Score, desto wahrscheinlicher ist ein menschlicher Benutzer:

Interaktion/Faktor	Bedingung	Punkte
Mausbewegungen	> 4 Ereignisse	+8
Tastatureingaben	> 3 Ereignisse	+12
Scrollverhalten	> 0 Ereignisse	+7
"Ich bin kein Roboter" Checkbox	Angehakt	+25
Ausgewogenes Verhältnis zwischen Maus/Tastatur	Ratio zwischen 0.3 und 3	+5
Honeypot-Feld ausgefüllt	Beliebiger Inhalt	-150 (sofortiger Bot-Verdacht)

Beispielrechnung für einen typischen menschlichen Benutzer:

- 10 Mausbewegungen: +8 Punkte
- 5 Tastatureingaben: +12 Punkte
- 2 Scrollaktionen: +7 Punkte
- Checkbox angehakt: +25 Punkte
- Ausgewogenes Interaktionsverhältnis: +5 Punkte
- **Gesamtscore: 57 Punkte** (weit über dem Standard-Schwellenwert von 5)

Beispielrechnung für einen einfachen Bot:

- 0 Mausbewegungen: 0 Punkte
- 1 Tastatureingabe: 0 Punkte
- 0 Scrollaktionen: 0 Punkte
- Checkbox nicht angehakt: 0 Punkte
- Honeypot-Feld ausgefüllt: -150 Punkte
- **Gesamtscore: -150 Punkte** (deutlich unter dem Schwellenwert)

Der Standardschwellenwert für eine erfolgreiche Validierung beträgt 5 Punkte und kann über die Konfigurationsoption `thresholdScore` angepasst werden.

Barrierefreiheits-Features

1. Screenreader-Unterstützung

- Semantisch korrekte HTML-Struktur
- ARIA-Live-Regionen für dynamische Inhalte
- Aussagekräftige Fehlermeldungen

2. Tastaturzugänglichkeit

- Skip-Links zum Hauptinhalt
- Logische Tab-Reihenfolge
- Fokus-Indikatoren für alle interaktiven Elemente

3. Robuste Formularvalidierung

- Fehlerrückmeldungen sind mit dem jeweiligen Feld verknüpft
- Validierungsstatus wird über ARIA-Attribute kommuniziert
- Fehlerkorrektur mit klaren Anweisungen

Konfigurationsoptionen

CaptureBarriereFrei-Konfiguration

```
const captureConfig = {  
  autoProtect: true,           // Automatischer Schutz aller  
  Formulare  
  formSelectors: 'form.protected', // CSS-Selektor für zu schützende  
  Formulare  
  botScoreFieldName: 'security-score', // Name des versteckten  
  Sicherheitsfeld  
  minTimeToFill: 3000,         // Minimale Zeit zum Ausfüllen in  
  Millisekunden  
  thresholdScore: 10,          // Mindest-Score für erfolgreiche  
  Validierung  
  enableLogging: false         // Debug-Ausgaben aktivieren  
};
```

MailSender-Konfiguration

```
const mailConfig = {  
  recipient: 'empfaenger@beispiel.de', // Empfänger-E-Mail-Adresse  
  subject: 'Neue Nachricht vom Kontaktformular',  
  formSelector: '#kontaktFormular',    // Formular-Selektor  
  endpoint: 'send_mail.php',            // PHP-Endpunkt für E-Mail-  
  Versand  
  method: 'POST',                      // HTTP-Methode  
  fallbackToMailto: true,              // Bei Fehlern auf mailto:  
  ausweichen  
  debug: true,                        // Debug-Modus aktivieren  
  preserveFormHandlers: true           // Bestehende Event-Handler
```

```
erhalten  
};
```

Verwendungsbeispiele

Grundlegende Einbindung

```
<!-- CSS einbinden -->  
<link rel="stylesheet" href="assets/css/style.css" />  
  
<!-- Formular mit protected-Klasse -->  
<form id="kontaktFormular" method="post" class="protected">  
  <!-- Formularfelder -->  
</form>  
  
<!-- Module einbinden -->  
<script type="module" src="assets/js/captureBarriereFrei/index.js">  
</script>  
<script src="assets/js/mailSender.js"></script>  
<script>  
  document.addEventListener('DOMContentLoaded', function() {  
    // CaptureBarriereFrei initialisieren  
    window.captureBarriereFreiInstance = new CaptureBarriereFrei({  
      formSelectors: 'form.protected',  
      minTimeToFill: 3000  
    });  
  
    // MailSender initialisieren  
    const mailSender = new MailSender({  
      recipient: 'empfaenger@beispiel.de',  
      formSelector: '#kontaktFormular'  
    });  
  });  
</script>
```

Erweitertes Beispiel mit benutzerdefinierten Validierungen

Siehe [index.html](#) für ein vollständiges Beispiel mit benutzerdefinierten Validierungen und Konfigurationen.

Anpassungsmöglichkeiten

Anpassen der Styles

Die visuellen Aspekte der Komponenten können über CSS angepasst werden. Die Hauptstile befinden sich in:

- [assets/css/style.css](#) - Allgemeine Styles
- [ui.js](#) - Spezifische Komponenten-Styles (intern)

Eigene Validierungsregeln hinzufügen

Sie können eigene Validierungsregeln erstellen, indem Sie die `validateField`-Funktion erweitern oder überschreiben:

```
// Benutzerdefinierte Validierung hinzufügen
const originalValidateField = captureBarriereFreiInstance.validateField;
captureBarriereFreiInstance.validateField = function(input, errorElement)
{
    // Originale Validierung durchführen
    const isValid = originalValidateField.call(this, input, errorElement);

    // Eigene Validierungslogik hinzufügen
    if (isValid && input.name === 'customField') {
        // Benutzerdefinierte Validierung...
    }

    return isValid;
};
```

Technische Details

Ereignisfluss bei der Formularverarbeitung

1. Initialisierung:

- CaptureBarriereFrei wird initialisiert und scannt nach Formularen
- Event-Listener für Maus, Tastatur und Scroll werden eingerichtet
- Formulare werden mit Schutzmaßnahmen versehen

2. Benutzer-Interaktion:

- Interaktionen werden erfasst und Sicherheits-Score aktualisiert
- Formularfelder werden bei Fokus/Blur validiert
- Honeypot-Felder und "Ich bin kein Roboter"-Checkbox werden überwacht

3. Formular-Übermittlung:

- Formular-Submit-Event wird abgefangen
- Alle Felder werden validiert
- Sicherheits-Score wird final berechnet
- Bei positivem Score wird das Formular an `send_mail.php` gesendet
- Erfolgs- oder Fehlermeldung wird angezeigt

4. E-Mail-Versand:

- `send_mail.php` empfängt die Daten
- E-Mail wird mit `PHP mail()` gesendet
- JSON-Antwort mit Erfolgs-/Fehlerstatus wird zurückgegeben
- Bei Fehler wird ggf. der `mailto:-Fallback` aktiviert

Fehlerbehebung

Formulardaten werden nicht gesendet

- Überprüfen Sie die Browser-Konsole auf JavaScript-Fehler
- Testen Sie die PHP-Mail-Funktionalität mit mail_diagnose.php
- Stellen Sie sicher, dass der Sicherheits-Score ausreichend hoch ist (debug: true)

E-Mail-Versand funktioniert nicht

- Überprüfen Sie die Servereinstellungen für PHP mail()
- Testen Sie alternative Mail-Konfigurationen wie SMTP
- Prüfen Sie Firewall-Einstellungen für ausgehende E-Mails

Probleme mit der Barrierefreiheit

- Testen Sie mit einem Screenreader (z.B. NVDA, JAWS, VoiceOver)
- Überprüfen Sie den Kontrast und die Lesbarkeit aller Elemente
- Stellen Sie sicher, dass alle interaktiven Elemente mit der Tastatur bedienbar sind

Lizenz

Dieses Projekt ist unter der MIT-Lizenz lizenziert - siehe die LICENSE.md-Datei für Details.

Mitwirkende

- JP. Böhm - Initialer Autor und Maintainer
- Claude 3.7 Sonnet Thinking (Struktur + Botdetection)

Danksagungen

- Besonderer Dank an alle, die zur Verbesserung der Barrierefreiheit im Web beitragen
- Inspiration durch etablierte Bot-Schutz-Systeme, mit dem Ziel, diese barrierefrei zu gestalten