SOP 23

Sichtbarer Fokusindikator

Projektleitung | Geschäftsführung | IT | Recht

Was du hier machst	Du gestaltest einen sichtbaren, markenkonformen Fokus-Indikator, der die Tastaturnavigation barrierefrei macht und prüfst seine korrekte Anwendung.
Voraussetzung	Du arbeitest in Entwicklung, UX, Design oder koordinierst Barrierefreiheitsmaßnahmen als EAA-Verantwortliche:r oder Projektleitung.
Dauer / Aufwand	Initialaufwand: 2–3 Stunden (für Einrichtung des Kanals, Erstellung der Tracking-Liste, Formulierung von Textbausteinen). Laufend: 15–30 Minuten pro eingegangener Meldung.
Gesetzlich verpflichtend	Ja – das Barrierefreiheitsstärkungsgesetz (BFSG), der European Accessibility Act (EAA) und die WCAG 2.1 (insbesondere 1.3.1 und 2.4.6) fordern eine verständliche und maschinenlesbare Struktur. Die DSGVO (Art. 12 Abs. 1) verlangt, dass Informationen in leicht zugänglicher Form bereitgestellt werden.
Nächster Schritt	SOP 24: Link- und Navigationstexte

🗒️ SOP 23: Gestalte sichtbare Fokus-Indikatoren

Der geheime Bonus für Tastatur-Nutzer und andere Menschen, die gerade nicht die Maus bedienen können

Wer sollte diese SOP lesen?

Was du hier machst

Du gestaltest einen sichtbaren, markenkonformen Fokus-Indikator, der die Tastaturnavigation barrierefrei macht und prüfst seine korrekte Anwendung.

🧠 Voraussetzung

Du arbeitest in Entwicklung, UX, Design oder koordinierst Barrierefreiheitsmaßnahmen als EAA-Verantwortliche:r oder Projektleitung.

Ergebnis

Eine klar strukturierte, logisch aufgebaute Textseite, bei der alle Inhalte technisch korrekt für Screenreader formatiert sind und eine sichtbare sowie maschinenlesbare Struktur aufweisen.

⏱️ Dauer

Initialaufwand: 2–3 Stunden (für Einrichtung des Kanals, Erstellung der Tracking-Liste, Formulierung von Textbausteinen). Laufend: 15–30 Minuten pro eingegangener Meldung.

🩹 Gesetzlich verpflichtend?

Ja – das Barrierefreiheitsstärkungsgesetz (BFSG), der European Accessibility Act (EAA) und die WCAG 2.1 (insbesondere 1.3.1 und 2.4.6) fordern eine verständliche und maschinenlesbare Struktur. Die DSGVO (Art. 12 Abs. 1) verlangt, dass Informationen in leicht zugänglicher Form bereitgestellt werden.

📍 Nächster Schritt

SOP 24: Link- und Navigationstexte

Warum machst du das überhaupt?

Wenn du eine Website nicht mit der Maus nutzen kannst (weil du z.B. motorisch eingeschränkt bist oder eine Muskelerkrankung hast), musst du zum Navigieren deine Tastatur nutzen.

Wenn du dich also z.B. auf einer Website befindest und dort die Shift-Taste mehrmals drückst, springst du mit Hilfe der Tastur/Shift-Taste automatisch von einem Element zum nächsten.

Damit du siehst, welches Element du gerade ansteuerst (du kannst es dann durch durch Pressen der Enter-Taste öffnen) wird um das jeweils ausgewählte Element ein Rahmen angezeigt.

Und hier fängt das Problem an: Es muss nicht standardmäßig auf deiner Seite existieren, viele Entwickler:innen entfernen diese Funktion sogar.

Ist dies der Fall, hast du damit eine echte Barrierere auf deiner Website – und die möchtest du natürlich loswerden. Weil du dadurch nicht nur mehr potenzielle Kund:innen ansprichst, sondern auch die gesetzlichen Vorgaben erfüllst.

Wie? Indem du diese SOP liest und ausführst.

Der wichtigste Punkt in dieser SOP ist die Regel Nr.1:

Es gibt keine Entschuldigung, den Fokus-Indikator zu entfernen!

Am Ende hast du einen klar definierten, durchgehend sichtbaren und markenkonformen Fokus-Stil für alle interaktiven Elemente deiner Website. Die Tastaturnavigation ist damit nicht nur möglich, sondern auch intuitiv und klar nachvollziehbar.

Es gibt mit dieser SOP keine Dokumentation … wenn du nicht willst. Ein Gesetz, das dies von dir erfordern würde,gibt es jedenfalls nicht. Aber falls du es wirklich gründlich und lückenlos machen möchtest, kannst du dir die Anweisung natürlich in dienem Ordner speichern – und so gegenüber Behörden nachweisen, dass du den Fokus-Indikator berückscihtigst und implementiert hast.

- Initialaufwand: 2–3 Stunden (für Einrichtung des Kanals, Erstellung der Tracking-Liste, Formulierung von Textbausteinen).
- Laufend: 15–30 Minuten pro eingegangener Meldung.
- DU (Projektleitung, EAA-Verantwortliche:r): Steuerst und überwachst den Prozess, führst die Triage durch und verantwortest die Kommunikation mit den Nutzer:innen.
- GF / Recht: Werden bei kritischen Meldungen oder potenziellen Eskalationen informiert und treffen strategische Entscheidungen.
- IT / Entwicklung / Agentur: Analysieren die gemeldete Barriere technisch und sind für die Behebung verantwortlich.
- Kundenservice: Kann als erste Anlaufstelle dienen und leitet Meldungen standardisiert an dich weiter.

Wann musst du das machen?

- Immer. Dieses Thema ist nicht optional.
- Bei der initialen Erstellung des globalen CSS oder des Styleguides.
- Vor jedem Launch und bei der Qualitätssicherung jedes einzelnen interaktiven Elements (Navigation, Buttons, Formulare, Links).
- Wenn du eine Tastatur zur Hand nimmst und versuchst, deine eigene Seite zu bedienen.

Welches Gesetz verlangt das?

- European Accessibility Act (EAA) / Barrierefreiheitsstärkungsgesetz (BFSG): Eine Website muss bedienbar sein, wozu die Tastaturnavigation fundamental dazugehört.
- WCAG 2.1 , als zentrales Kriterium:
- Die neueren WCAG 2.2 konkretisieren dies weiter und sind zukunftsweisend.

Der Prozess im Detail

1 Verstehe das Grundproblem …%22outline: none ist dein Feind%22″

Früher hatten Webseiten oft ein Problem. Der Fokus-Indikator (meist ein blauer oder schwarzer Rahmen, den der Browser automatisch gesetzt hat) war auch dann zu sehen, wenn jemand mit der Maus geklickt hat. Und das fanden viele Webdesigner „hässlich“, weil es ihrer Meinung nach das Design gestört hat. Deshalb haben sie oft einfach diesen Rahmen ganz weggemacht, mit dem Befehl outline: none; oder outline: 0; .

Das ist aber, als würde man bei einem Auto die Blinker abbauen, nur weil sie im Stand vielleicht nicht schön aussehen. Sobald du fährst, brauchst du sie aber dringend!

Deshalb wollen wir im ersten Schritt sicher stellen, dass auf jeden Fall Deine Tastatur-User den Rahmen als Highlight sehen können.

Die moderne und korrekte Lösung für dieses „Problem“ heißt :focus-visible. „:focus visible“ bedeutet dabei einfach,dass der der Fokus-Stil (also dieser Rahmen, der zeigt, wohin man gerade navigiert) nur dann angezeigt wird , wenn jemand mit der Tastatur navigiert – genau dann, wenn er wirklich gebraucht wird (Barrierefreiheit!).

Diese CSS-Pseudoklasse sorgt dafür, dass dein benutzerdefinierter Fokus-Stil nur dann angezeigt wird, wenn der Nutzer per Tastatur navigiert – nicht aber beim Mausklick.

Und das machen wir in den folgenden Schritten.

2 Optimiere deinen Fokus-Indikator

Bevor du deinen sichtbaren Fokus-Indikator implementierst, willst du sicherstellen, dass er maximal nützlich ist. Die Grunderegel dabei lautet: Ein guter Fokus-Indikator ist nicht dezent, er ist offensichtlich .

Konkret bedeutet es:

- Kontrast: Er muss einen Kontrast von mindestens 3:1 zum Hintergrund haben.
- Größe und Dicke: Er sollte dick genug sein, um sofort ins Auge zu fallen (z. B. mindestens 2px dick).
- Keine Verdeckung: Stelle sicher, dass der Indikator niemals von anderen Elementen wie Sticky Headern, Footern oder Cookie-Bannern verdeckt wird.
- Klarheit: Er muss sich deutlich von anderen Zuständen wie dem Hover-Effekt (Mouseover) unterscheiden.
- Konsistenz: Sorge für einheitliche Fokus-Stile über die gesamte Website, um die Vorhersehbarkeit für Nutzer zu erhöhen.

Im Code könnte das konkret folgendermaßen aussehen:

```
*/

:focus-visible {

outline: 3px solid #0057B8; /* Beispiel: Eine kräftige, kontrastreiche Farbe */

outline-offset: 2px;

border-radius: 4px; /* Optional: Passt den Radius an deine Buttons an */

}
```

3 Individualisere deinen Fokus-Indikator

Du solltest natürlich nicht nur deinen Fokus-Indikator optimieren, sondernihn auch auf deinen Corporate Design anpassen. Und zwar wie folgt

- Dicker Rahmen (outline): Die beste Methode. Sie verändert das Layout des Elements nicht.
- Beispiel: Ein 3px dicker, durchgehender Rahmen in einer deiner Markenfarben mit etwas Abstand zum Element (outline-offset).
- Box-Schatten (box-shadow): Eine flexible Alternative, die ebenfalls das Layout nicht beeinflusst. Beachte jedoch, dass Schatten in einigen „erzwungene Farben“-Modi (High Contrast Mode) nicht sichtbar sind; kombiniere ihn dann am besten mit einem transparenten outline .

Beispiel: box-shadow: 0 0 0 3px #DeineMarkenfarbe;

- Hintergrund- und Textfarbe ändern: Eine weitere Option, bei der du aber sicherstellen musst, dass die neuen Farben untereinander und zum Hintergrund immer noch genügend Kontrast haben.
- Wovon abzuraten ist: Kleine Änderungen wie eine dünne Unterstreichung oder eine leichte Farbänderung des Textes sind oft nicht ausreichend sichtbar.<(li>

4 Implementiere :focus-visible

Nun, da du den Fokus-Indikator designedhast, möchtest du ihn natürlich auf deiner Seite implementieren.+

Ganz egal,, ob du es selbst machst oder mit Hilfe eines Entwicklers – das ist der Prozess: Du musst diesen Code in dem Quelltext (bzw.CSS) deiner Website integrieren:

CSS

```
/* Moderner und empfohlener Ansatz:
Dieser Stil wird nur bei Tastatur-Navigation angewendet,
nicht bei Mausklicks. Er ersetzt den Standard-Stil.
*/
:focus-visible {
outline: 3px solid #0057B8; /* Beispiel: Eine kräftige, kontrastreiche Farbe */
outline-offset: 2px;
border-radius: 4px; /* Optional: Passt den Radius an deine Buttons an */
}
```

Dieser Code-Block ist die Grundlage. Weise deine Entwickler:innen an, jegliche outline: none oder outline: 0 aus dem Code zu entfernen und durch diesen Block zu ersetzen.

5 **Testen, testen, testen**

Du weißt jetzt, dass du deinen Fokus-Indikator auf deiner Website und damit eine extrem wichtige Barriere abgebaut hast. Nun willst du sichergehen, dass es auch funktioniert. Und da ist der einzige sichere Weg eine manuelle Prüfung.

Das heißt:Der einzige Weg, um sicherzugehen, ist eine manuelle Prüfung.

Und die führen wir wie folgt durch:

- Lege die Maus weg.
- Navigiere deine komplette Website von oben bis unten nur mit der Tab-Taste (und Shift + Tab zum Zurückspringen).
- Stelle bei jedem Schritt diese Fragen:
- Sehe ich sofort und eindeutig , wo ich bin?
- Ist der Fokus-Indikator immer sichtbar?
- Springt der Fokus in einer logischen Reihenfolge?
- Nutze ggf. ein Tool zur Visualisierung der Tab-Reihenfolge (z.B. in den Browser-Entwicklertools wie Chrome Accessibility Pane , Firefox Accessibility Panel, oder spezialisierte Browser wie Polypane).
- Wird der Fokus von einem anderen Element (Header, Chat-Widget) verdeckt?
- Funktioniert es in allen interaktiven Elementen: Links, Buttons, Menüpunkten, Formularfeldern, Slidern?

Du hast nun:

- Eine Website, die für alle Tastatur-Nutzer:innen vollständig und stressfrei bedienbar ist.
- Ein zentrales und kritisches WCAG-Kriterium zuverlässig erfüllt.
- Einen professionellen, markenkonformen Look, der Inklusivität signalisiert.
- Den häufigsten und unnötigsten Barrierefreiheitsfehler aus deiner Codebase eliminiert.

Copy-Paste-Baustein: Fokusindikator-Code

Und weil wir mittlerweile raus aus der strategischen Planung sind und uns inmitten der operativen Umsetzung befinden, hier mal keine E-Mail, die du an Kolleg:innen raussenden kannst, sondern der Code, der gründlich und umfassend diese Barriere in deinem Unternehmen enfert

Schnelle Implementierung: Dein CSS-Code zum Kopieren

Dieser Code ist die Basis für einen sichtbaren, barrierefreien und modernen Fokus-Indikator . Kopiere ihn und gib ihn an deine Entwickler:innen weiter oder füge ihn direkt in dein globales CSS ein. Er ersetzt den oft fehlerhaften outline: none; und sorgt dafür, dass der Fokus-Indikator nur dann angezeigt wird, wenn er gebraucht wird (bei Tastaturnavigation).

/* Empfohlener Standard-Stil für Fokus-Indikatoren.

Dieser Stil wird nur bei Tastatur-Navigation angewendet (:focus-visible).

Er sorgt für optimale Sichtbarkeit und ist WCAG-konform.

*/

:focus-visible {

outline: 3px solid #0057B8; /* Helle, gut sichtbare Farbe deiner Wahl – idealerweise eine Markenfarbe */

outline-offset: 2px; /* Sorgt für etwas Abstand zum Element */

border-radius: 4px; /* Optional: Passt den Radius an deine Buttons/Elemente an */

/* Tipp: Für zusätzlichen Kontrast könntest du auch einen box-shadow hinzufügen:

box-shadow: 0 0 0 1px rgba(0, 0, 0, 0.2);

*/

}

/* Wichtig: Entferne alle anderen ‚outline: none;‘ oder ‚outline: 0;‘ in deinem Code! */

/* Wenn spezifische Elemente eigene Fokus-Stile haben, stelle sicher,

dass diese die Eigenschaften von :focus-visible überschreiben und sichtbar bleiben. */

Fragen & Antworten

Warum ist der Kontrast von 3:1 für Fokus-Indikatoren anders als die 4.5:1 für normalen Text?

Der Kontrast von 4.5:1 gilt für normalen Text, um dessen Lesbarkeit zu gewährleisten . Für grafische Objekte und Benutzeroberflächen-Komponenten, zu denen auch der Fokus-Indikator gehört, fordert die WCAG (Web Content Accessibility Guidelines) einen Mindestkontrast von 3:1 zum angrenzenden Hintergrund . Das liegt daran, dass der Fokus-Indikator nicht primär gelesen, sondern visuell erkannt werden muss.

Ich habe gehört, dass man den outline-Stil nicht verändern oder entfernen sollte, weil er wichtig ist. Stimmt das?

Es stimmt, dass outline: none; ein Problem ist, weil es den Fokus-Indikator für Tastaturnutzer komplett entfernt. Deine Darum ist es empfehlenswert, den Standard-outline nicht zu entfernen, sondern ihn mit :focus-visible individuell anzupassen und zu optimieren. So bleibt die Funktionalität erhalten, aber das Design wird verbessert. Das ist der Kerngedanke von :focus-visible .

Gibt es Fälle, in denen ein Element fokussierbar ist, aber kein sichtbarer Fokus-Indikator angezeigt wird, auch mit :focus-visible?

outline -Eigenschaften, die für den Fokus-Indikator verwendet werden, könnten in bestimmten „erzwungene Farben“-Modi (wie dem Windows High Contrast Mode) nicht sichtbar sein, wenn sie nicht entsprechend gestaltet sind.

Wenn du box-shadow verwendest, kann dieser in diesen Modi ebenfalls unsichtbar sein. Daher empfiehlt es sich, box-shadow mit einem transparenten outline zu kombinieren, da transparente outline -Farben in diesen Modi sichtbar werden können.

Warum sollte ich tabindex mit positiven Werten (z.B. tabindex=%221%22) vermeiden?

Positive tabindex -Werte sind eine schlechte Praxis, weil sie die natürliche Tab-Reihenfolge im HTML-Dokument überschreiben . Das macht die Navigation für Tastaturnutzer unvorhersehbar und verwirrend, da der Fokus nicht mehr der visuellen oder logischen Reihenfolge folgt . Der Browser priorisiert Elemente mit positiven tabindex -Werten zuerst, bevor er den Rest des Dokuments in seiner ursprünglichen Reihenfolge durchgeht .

Die manuelle Prüfung mit der Tab-Taste kann bei großen Seiten sehr zeitaufwendig sein. Gibt es dazu eine Empfehlung?

Das stimmt, die manuelle Prüfung ist unerlässlich, um die Nutzererfahrung zu gewährleisten. Du kannst den Prozess jedoch durch automatisierte Tests ergänzen, die „Low-Hanging Fruits“ (leicht erkennbare Fehler) finden.

Tools wie „axe DevTools“ oder „WAVE“ können helfen, grundlegende Probleme zu identifizieren, bevor du zur manuellen Prüfung übergehst. Denke daran, dass automatisierte Tests niemals eine manuelle Prüfung ersetzen können.