SOP 43

Screenreader-Workflow

Projektleitung | Geschäftsführung | IT | Recht

So erlebst du deine Website wie ein:e blinde:r Nutzer:in und findest die wirklich kritischen Barrieren

Weil dies der ultimative Realitäts-Check für Barrierefreiheit ist. Automatisierte Tools finden nur technische Fehler im Code. Ob deine Website aber wirklich logisch, verständlich und ohne Frust bedienbar ist, kann dir keine Software sagen.

Nur ein manueller Test mit einem Screenreader deckt die schwerwiegendsten Probleme auf: eine unlogische Lesereihenfolge, unverständliche Link-Bezeichnungen, fehlender Kontext in Formularen oder eine nicht funktionierende Navigation in einem Slider.

Diesen Schritt auszulassen bedeutet, im Blindflug zu agieren und die tatsächliche Nutzererfahrung der Menschen zu ignorieren, für die du diese Maßnahmen ergreifst. Eine Seite, die den Screenreader-Test nicht besteht, ist nicht barrierefrei. Punkt.

Wann musst du das machen?

- Als zentraler Bestandteil jedes manuellen Testzyklus (siehe SOP 38).
- Vor jedem Launch eines neuen Seitentemplates, eines neuen Features oder eines Kernprozesses (z. B. Checkout).
- Immer, wenn du komplexe, interaktive Komponenten wie modale Dialoge, Slider oder Akkordeons implementierst.
- Wenn du einen Fehler gemeldet bekommst und ihn nachvollziehen musst.

Welches Gesetz verlangt das?

- EAA / BFSG / WCAG 2.1: Dieser Test ist die einzige Möglichkeit, die Einhaltung unzähliger WCAG-Kriterien der Prinzipien Wahrnehmbar, Bedienbar und Verständlich wirklich zu validieren. Er prüft unter anderem: 1.3.1 (Info & Beziehungen), 2.4.3 (Fokus-Reihenfolge), 2.4.4 (Link-Zweck), 3.3.2 (Beschriftungen), 4.1.2 (Name, Rolle, Wert) und 4.1.3 (Statusmeldungen).

Der Prozess im Detail

1 🟧 Phase 1: Werkzeuge einrichten und Grundbefehle lernen

Du brauchst keine Vorkenntnisse, nur die richtige Software und die wichtigsten Befehle.

Werkzeug

Beschreibung & Installation

NVDA

Der weltweit meistgenutzte, kostenlose Screenreader für Windows . Lade ihn von www.nvaccess.org herunter und installiere ihn.

VoiceOver

Auf jedem Mac, iPhone und iPad bereits vorinstalliert. Aktiviere ihn unter Systemeinstellungen > Bedienungshilfen > VoiceOver oder mit dem Kurzbefehl Cmd + F5.

Die wichtigsten Tastaturbefehle (dein Handwerkszeug):

Aktion

NVDA (Windows)

VoiceOver (macOS)

Nächstes Element lesen

Pfeil nach unten

VO + Pfeil rechts (VO = Ctrl + Option)

Element aktivieren

Enter oder Leertaste

VO + Leertaste

Nächste Überschrift

H

VO + Cmd + H

Nächster Link / Button

K / B

VO + Cmd + L / VO + Cmd + B

Nächstes Formularfeld

F

VO + Cmd + J

Elemente-Liste öffnen

Insert + F7

VO + U (Rotor öffnen, dann auswählen)

2 🟨 Phase 2: Der Orientierungs-Test – Wie ist der erste Eindruck?

Simuliere, wie ein:e Nutzer:in sich einen Überblick über die Seite verschafft.

- Seite laden & zuhören: Schließe die Augen. Was wird als Erstes vorgelesen? Wird der Seitentitel (<title>) angesagt und ist er aussagekräftig?
- Mit Überschriften navigieren: Springe nur mit der H-Taste (NVDA) oder über den Rotor (VoiceOver) durch die Überschriften.
- Ergibt sich eine logische Gliederung?
- Gibt es genau eine <h1>?
- Kannst du die Seiteninhalte grob erfassen, nur anhand der Überschriften?
- Mit Landmarks navigieren: Springe mit D (NVDA) oder über den Rotor (VoiceOver) durch die Seitenregionen.
- Werden header, nav, main und footer korrekt erkannt?
- Kannst du direkt zum Hauptinhalt (main) springen und die Navigation überspringen?

3 🟩 Phase 3: Der Interaktions-Test – Kann ich die Seite wirklich nutzen?

Jetzt geht es ans Eingemachte.

- Formulare ausfüllen: Navigiere zu einem Formular.
- Wird zu jedem Eingabefeld ein klares <label> vorgelesen?
- Werden Hinweise zum Format oder Pflichtfelder angesagt?
- Sind Fehlermeldungen verständlich und werden sie vorgelesen, wenn du das Formular falsch abschickst?
- Links & Buttons bedienen:
- Ist der vorgelesene Name eines Links auch ohne den umgebenden Text verständlich? (z.B. „Jahresbericht herunterladen“ statt „hier klicken“)
- Beschreibt der Button-Name klar seine Funktion? (z.B. „Anfrage jetzt absenden“ statt nur „Absenden“)
- Komplexe Komponenten testen: Öffne ein Akkordeon oder einen Slider.
- Wird der Zustand angesagt (z.B. „eingeklappt“, „Slide 2 von 5“)?
- Kannst du alle Bedienelemente erreichen und nutzen?

4 🟥 Phase 4: Inhalte & Medien prüfen – Was sehe ich nicht?

- Bilder: Werden aussagekräftige Alternativtexte vorgelesen? Ist der Alt-Text eine nützliche Beschreibung oder nur nutzloser Fülltext?
- Tabellen: Navigiere in eine Datentabelle. Werden die Spalten- und Zeilenüberschriften beim Navigieren durch die Zellen mit vorgelesen, sodass du immer den Kontext der Daten kennst?
- Videos: Kannst du den Videoplayer erreichen und die Steuerelemente (Play, Pause, Lautstärke) bedienen?

5 🟦 Phase 5: Fehler präzise dokumentieren

Ein guter Bug-Report ist der halbe Fix.

- Sei spezifisch: Schreibe nicht „Der Link ist kaputt“. Schreibe: „Mit NVDA auf der Kontaktseite: Der Link im Footer mit dem Text ‚Datenschutz‘ wird nur als ‚Link‘ vorgelesen. Der Linktext fehlt in der Ausgabe.“
- Gib Schritte zur Reproduktion an: 1. Gehe zu URL. 2. Starte NVDA. 3. Drücke K, um zum Link zu springen. 4. Höre auf die Ausgabe.
- Erwartetes vs. tatsächliches Ergebnis: Formuliere klar, was du erwartet hast und was stattdessen passiert ist.
- Mache eine kurze Aufnahme: Ein 15-sekündiges Video mit dem Screenreader-Ton ist für Entwickler:innen oft hilfreicher als eine lange Beschreibung.

Ergebnis

Am Ende dieser SOP hast du:

- Die Fähigkeit, die häufigsten und kritischsten Barrieren aufzudecken, die automatisierte Tools übersehen.
- Ein tiefes Verständnis für die tatsächliche Nutzererfahrung von Menschen, die auf Hilfstechnologien angewiesen sind.
- Präzise und nachvollziehbare Fehlerberichte, die dein Entwicklungsteam direkt umsetzen kann.
- Die Sicherheit, dass deine Website nicht nur technisch konform, sondern auch praktisch nutzbar ist.

Und hier noch ein hoffentlich nützlicher E-Mail-Baustein, mit dem du darüber informierst, dass nun DSGVO und Barrierefreiheit synchronisiert sind:

Titel: Screenreader-Bug: [Kurze Beschreibung des Problems]

URL: [Link zur betroffenen Seite] Screenreader: [NVDA / VoiceOver] Browser: [Chrome / Firefox / Safari]

Problembeschreibung: [Detaillierte, spezifische Beschreibung des Problems.]

Schritte zur Reproduktion: 1. 2. 3.

Erwartetes Ergebnis: [Was sollte der Screenreader tun/sagen?]

Tatsächliches Ergebnis: [Was tut/sagt der Screenreader stattdessen?]

(Optional) Anhang: [Link zu Screenshot oder Video-Aufnahme]

Fragen & Antworten

Wie kann ich über den seitenbasierten Test hinausgehen und ganze Benutzerabläufe (User Journeys) testen?

Um komplexe Prozesse wie einen Checkout oder eine Registrierung zu testen, sollten Sie szenariobasiert vorgehen. Statt nur eine einzelne Seite zu prüfen, folgen Sie dem gesamten Prozess von Anfang bis Ende. Achten Sie dabei besonders auf Übergänge zwischen den Seiten, Statusmeldungen und die logische Abfolge von Formularen oder Schritten. Spezielle Herausforderungen sind hier die korrekte Ansage von dynamisch nachgeladenen Inhalten und das persistente Speichern von Eingaben über mehrere Seiten hinweg.

Wie unterscheidet sich der Test auf mobilen Geräten von dem auf dem Desktop?

Das mobile Testing mit Screenreadern wie VoiceOver auf iOS oder TalkBack auf Android unterscheidet sich erheblich. Nutzer verwenden Gesten (Wischen, Tippen) anstelle von Tastatur-Shortcuts. Sie müssen daher die entsprechenden Touch-Befehle beherrschen. Achten Sie darauf, wie Elemente in der „Wisch-Reihenfolge“ angeordnet sind und ob interaktive Elemente wie Slider oder komplexe Menüs auch mit Fingergesten korrekt bedienbar sind.

Inwiefern kann Automatisierung bei Screenreader-Tests helfen, obwohl der manuelle Test als unverzichtbar gilt?

Manuelle Screenreader-Tests sind unerlässlich, um die tatsächliche Nutzererfahrung zu prüfen. Automatisierte Tools können jedoch als Ergänzung dienen, um wiederkehrende, technische Checks durchzuführen. Beispielsweise können Tools wie axe-core in ein CI/CD-System integriert werden, um bei jeder Codeänderung auf häufige technische Fehler wie fehlende ARIA-Attribute oder unzureichenden Farbkontrast zu prüfen. So können Sie sicherstellen, dass die grundlegenden Kriterien stets erfüllt sind und sich der manuelle Test auf die komplexeren, nutzerzentrierten Aspekte konzentrieren kann.

Was ist der Zusammenhang zwischen den WCAG-Kriterien und ARIA-Attributen, und warum muss ich als Tester beides verstehen?

Viele WCAG-Kriterien, insbesondere die, die das Verhalten von dynamischen oder interaktiven Komponenten betreffen, werden durch ARIA-Attribute (Accessible Rich Internet Applications) im Code umgesetzt. Ein grundlegendes Verständnis von ARIA-Rollen (role), Zuständen (state) und Eigenschaften (property) ermöglicht es Ihnen als Tester, nicht nur festzustellen, dass etwas nicht funktioniert, sondern auch die wahrscheinliche Ursache (z.B. ein fehlendes aria-label bei einem Button ohne sichtbaren Text) zu benennen. Das macht die Fehlerbeschreibung für Entwickler präziser und effektiver.

Welche speziellen Probleme treten beim Testen von Single Page Applications (SPAs) auf?

Bei SPAs, die mit Frameworks wie React oder Vue.js erstellt wurden, liegt die größte Herausforderung in der dynamischen Natur der Inhalte. Screenreader müssen über Änderungen auf der Seite, wie etwa den Wechsel zu einem neuen Seitenbereich oder die Aktualisierung von Inhalten nach einer Nutzerinteraktion, informiert werden. Hier sind korrekte ARIA-Live-Regionen entscheidend, um sicherzustellen, dass Statusmeldungen oder neue Inhalte vorgelesen werden, ohne dass der Nutzer die Seite manuell neu laden muss.

Wie kann man Entwickler am besten in den Screenreader-Workflow integrieren?

Das Testen mit Screenreadern sollte nicht alleinige Aufgabe des QA-Teams sein. Am effektivsten ist es, wenn Entwickler lernen, ihre eigenen Komponenten direkt nach der Implementierung kurz mit einem Screenreader zu testen. Planen Sie kurze Demos oder Workshops, in denen Entwickler direkt miterleben können, wie ihre Arbeit von einem Screenreader wahrgenommen wird. Dies fördert das Verständnis und die Sensibilität für das Thema und führt zu weniger Fehlern von vornherein.

Die Dokumentation der Fehler ist im SOP-Text beschrieben. Gibt es Tools, die darüber hinaus helfen, Barrierefreiheits-Bugs zu verwalten?

Ja, über die reine Ticket-Erstellung hinaus gibt es spezialisierte Tools und Frameworks, die bei der Verwaltung von Accessibility-Fehlern unterstützen. Plattformen wie Axe DevTools oder Pa11y bieten nicht nur automatisierte Tests, sondern oft auch Dashboards, um den Fortschritt zu verfolgen, Bugs zu priorisieren und Berichte zu erstellen. Diese Lösungen können die manuelle Dokumentation ergänzen und den gesamten Prozess effizienter gestalten.