Mehr Reichweite, weniger Barrieren – Dein Content-Boost

Workshop zum TTWW – Mai 2025

SEO & Barrierefreiheit – wo liegt die Verbindung?

Barrierefreiheit und Suchmaschinenoptimierung (SEO) verfolgen unterschiedliche Ziele, treffen sich aber an vielen Stellen. Barrierefreiheit zielt darauf ab, Inhalte für alle Menschen zugänglich zu machen – insbesondere für Menschen mit Behinderungen. SEO hingegen sorgt dafür, dass Inhalte von Suchmaschinen besser gefunden und eingeordnet werden können. **Beide profitieren von klar strukturierten, verständlichen und technisch sauberen Inhalten.**

Gemeinsame Prinzipien

1. **Semantische HTML-Struktur**
   Überschriften, Listen, Absätze, Tabellen: Wenn Inhalte korrekt mit HTML ausgezeichnet sind, können sowohl Screenreader als auch Suchmaschinen die Inhalte besser verstehen.

2. **Alternative Texte für Bilder**
   Alt-Texte helfen blinden oder sehbehinderten Nutzer*innen beim Verstehen von Bildern – und sie liefern Google gleichzeitig Kontext für die Bildersuche.

3. **Klare Navigation & verständliche Linktexte**
   Eine gut strukturierte Website mit sprechenden Links verbessert sowohl die Nutzerführung für alle Menschen als auch das Crawling durch Suchmaschinen.

4. **Mobile Optimierung & Lesbarkeit**
   Barrierefreie Seiten sind oft auch mobilfreundlich und gut lesbar – das wirkt sich positiv auf die Nutzererfahrung und die SEO-Bewertung aus.

5. **Page Speed & saubere Technik**
   Technische Barrierefreiheit (z. B. keine blockierenden Skripte) fördert schnelle Ladezeiten und eine funktionierende Indexierung.

Fazit

Barrierefreiheit verbessert die User Experience (UX) für alle. Und genau das belohnen Suchmaschinen. Wer also auf digitale Barrierefreiheit achtet, arbeitet gleichzeitig an seiner SEO. Es ist kein Entweder-oder, sondern ein Win-Win für Reichweite und Inklusion.

Checklisten

Inhalte

✅ Checkliste: Barrierefreie Inhalte

- **Überschriften-Hierarchie korrekt** (<h1> bis <h6>, keine Sprünge)
- **Alt-Texte für alle relevanten Bilder** (alt="" bei rein dekorativen Bildern)
- **Einfache, klare Sprache verwenden**
- **Keine Informationen nur durch Farbe vermitteln**
- **Genügender Farbkontrast** (Text ↔ Hintergrund)
- **Verlinkter Text ist aussagekräftig** („Mehr erfahren über XY“, nicht nur „Hier klicken“)
- **Links mit Unterstrich kennezeichnen** (text-decoration: underline)
- **Tabellen nur für tabellarische Daten, mit th und scope**
- **Videos mit Untertiteln / Audiodeskription (wenn nötig)**
- **PDFs oder Downloads sind barrierefrei oder gekennzeichnet**
- **Keine flackernden / sich schnell bewegenden Inhalte** (→ Epilepsiegefahr)
- **Sprache des Inhalts ist im <html lang=""> korrekt gesetzt**
- **Fremdsprachige Begriffe mit lang-Attribut ausgezeichnet**

Nützliche Kennzeichnungen für barrierefreie Inhalte

HTML-Element / Attribut	Zweck	Beispiel
<abbr>	Abkürzungen mit Erklärung	<abbr title="World Health Organization">WHO</abbr>
<cite>	Zitatquelle (z. B. Buchtitel)	<cite>Buchtitel</cite>
<blockquote>	Längeres Zitat	<blockquote>Text</blockquote>
<q>	Kurzes, eingebettetes Zitat	Er sagte: <q>Hallo</q>

Tastatur-Bedienung

✅ **Tastaturbedienbarkeit – Checkliste**

- **Tab-Reihenfolge ist logisch**
- **Alle interaktiven Elemente sind fokussierbar** (button, a, input, etc.)
- **Keine „Tastaturfallen“** (Fokus kann nicht steckenbleiben)
- **Fokus ist sichtbar** (z. B. durch Outline)
- **tabindex korrekt eingesetzt** (0 für zusätzliche Elemente, -1 für programmatischen Fokus)
- **Tastatursteuerung für benutzerdefinierte Komponenten** (Enter, Space, Pfeiltasten etc.)
- **Sprunglinks vorhanden** (z. B. „Zum Inhalt springen“)
- **Modale Dialoge sind vollständig per Tastatur bedienbar**
- **Keine Maus-Only-Funktionen** (z. B. Hover-Menüs ohne Tastaturalternative)
- **ARIA-Attribute unterstützen Tastaturnavigation** (z. B. aria-expanded, aria-hidden)
- **Escape schließt Modale/Dialoge**
- **Rollen und Zustände sind programmatisch verfügbar**

Formulare

✅ Checkliste: Barrierefreie Formulare

- **<label> ist immer vorhanden** und mit for-Attribut verknüpft
- **Sichtbare Beschriftung entspricht der Screenreader-Beschriftung**
- **Pflichtfelder sind klar gekennzeichnet** (z. B. mit * **und** Hinweis per aria-required oder im Text)
- **Fehlermeldungen sind eindeutig, sichtbar und per Screenreader erfassbar** (aria-describedby)
- **Tab-Reihenfolge ist logisch**
- **Fokus springt bei Fehlern zum ersten fehlerhaften Feld**
- **Hinweise oder Formatvorgaben stehen vor dem Eingabefeld**
- **Gruppierte Felder (z. B. Radiobuttons) mit fieldset + legend umgeben**
- **Buttons sind klar beschriftet („Absenden“, nicht „OK“)
- **Autocomplete verwendet, wo sinnvoll** (autocomplete="email", etc.)
- **Keine Placeholder-only-Beschriftung** (Placeholders **ergänzen**, nicht ersetzen Labels)
- **Fehlermeldungen nicht nur farblich kennzeichnen** (z. B. zusätzlich Symbol oder Text)
- **Barrierefreie CAPTCHA-Alternativen anbieten** (z. B. logische Frage statt Bild)

WAI-ARIA-Rollen

✅ Was sind WAI-ARIA Rollen überhaupt?

WAI-ARIA (Web Accessibility Initiative – Accessible Rich Internet Applications) erweitert HTML, damit dynamische Inhalte und komplexe UI-Komponenten für assistive Technologien verständlich werden.

🔑 Wofür braucht man sie?

- Für **barrierefreie Interaktionen**, die HTML allein nicht ausreichend beschreiben kann
- Z. B. bei Slidern, Dialogen, Akkordeons, Tabs, Modalen usw.

🧱 Arten von WAI-ARIA Rollen:

1. **Landmark-Rollen**
   z. B. banner, navigation, main, complementary, contentinfo
   → *Erlauben schnelle Orientierung im Screenreader*
2. **Widget-Rollen**
   z. B. button, dialog, tab, tabpanel, slider
   → *Beschreiben interaktive Komponenten*
3. **Dokument-Rollen**
   z. B. article, heading, list, listitem

⚠️ Häufige Fehler & Missverständnisse

- ❌ **Rollen mehrfach kombinieren (role="button link")** → Nur **die erste** zählt, der Rest wird ignoriert
- ❌ **Rollen ersetzen vorhandenes HTML nicht automatisch** → Ein braucht **immer noch JS und Tastaturhandling**
- ❌ **ARIA ist kein Ersatz für sauberes HTML!**
  → *“First, use native HTML. Then ARIA if you must.”*

✨ Gute Praxis

- Nutze **semantisches HTML** zuerst – z. B. <button> statt div + role="button"
- Setze aria-label, aria-labelledby oder aria-describedby, um Rollen sprachlich zu erklären
- Verwende aria-roledescription nur, wenn die native Rolle nicht reicht

Kriterien für Barrierefreiheit im Überblick

1. Bedienbarkeit

- **Alle Funktionen mit Tastatur bedienbar** (Tab, Enter, Esc etc.)
- **Fokus sichtbar** (z. B. Umrandung beim Tabben durch die Seite)
- Keine „Maus-only"-Funktionen (Hover-Effekte auch per Tastatur nutzbar)

2. Wahrnehmbarkeit

- **Gute Kontraste** (Text vs. Hintergrund, Buttons, Links)
  Empfohlen: Mind. 4.5:1 für normalen Text
- **Alternativtexte (alt-Tags)** für Bilder, Icons, Logos
- **Keine rein visuellen Inhalte ohne Beschreibung** (z. B. Infografiken → kurze Erklärung)

3. Struktur & Orientierung

- **Korrekte HTML-Überschriftenstruktur** (H1, H2, H3 … logisch aufgebaut)
- **Navigation ist einheitlich & logisch** (Header, Menü, Footer)
- **Sprache der Seite definiert** (<html lang="de">)

4. Mobil & Zoom-fähig

- Seite funktioniert **responsiv** auf mobilen Geräten
- Inhalte bleiben **lesbar bis 200 % Zoom**
- Keine Inhalte überlappen oder verschwinden beim Vergrößern

5. Screenreader & Assistenzsysteme

- Formulare haben **beschriftete Felder** (<label>)
- Links und Buttons haben **klare Bezeichnungen** (nicht nur „Hier klicken")
- Keine „leeren" Buttons oder Links (z. B. nur mit einem Icon)

6. Multimedia & Bewegung

- Videos haben **Untertitel oder Transkripte**
- Animationen können **pausiert oder deaktiviert** werden
- Kein automatisches Abspielen von Ton/Videos beim Laden

7. Technisches (Bonus)

- Sauberes HTML / ARIA-Rollen (für komplexere Komponenten)
- Verwendung semantischer Tags (<main>, <nav>, <footer>)
- Kein übermäßiger Einsatz von JavaScript für essentielle Inhalte

Quellen: https://www.fruits-harvest.de/barrierefreiheit-jimdo-wordpress-shopify/

https://www.barrierefreies-webdesign.de/richtlinien/wcag-2.2/erfolgskriterien/

Theme-Auswahl für WordPress

Die wichtigsten Kriterien, die sich mit Test-Tools prüfen lassen:

1. Themes Basis

- Navigation, Farbkontraste, Fokusmarkierungen, ARIA-Labels und die semantische Struktur prüfen.

2. Tastaturnavigation und Fokussteuerung sicherstellen

- Alle interaktiven Elemente (Menüs, Formulare, Buttons) sollten mit der Tab-Taste erreichbar und steuerbar sein.
- Visuelle Fokus-Indikatoren dürfen **nicht entfernt** oder versteckt werden.

3. Screenreader-Kompatibilität

- Nutze semantisches HTML: Überschriftenhierarchie korrekt anwenden (<h1> bis <h6>), Listen, Tabellen, Buttons statt divs.
- Ergänze nötige **ARIA-Rollen** und -Attribute, z.B. aria-label, aria-live, aria-expanded.

4. Formulare optimieren

- Alle Eingabefelder sollten ein zugeordnetes <label> haben.
- Validierungsfehler müssen **verständlich und visuell wie akustisch** kommuniziert werden.
- Visuelle Hinweise (z.B. Pflichtfeldsterne) zusätzlich **textlich beschreiben**.

5. Farben & Kontraste prüfen

- Der Kontrast zwischen Text und Hintergrund sollte mindestens **4.5:1** betragen (WCAG AA).
- Tools: WAVE, Contrast Checker

6. Alternative Texte für Bilder

- Jeder img-Tag braucht ein **aussagekräftiges alt-Attribut**.
- Bei dekorativen Bildern: alt=""

7. Barrierefreiheitstools & Plugins verwenden

- **WP Accessibility**: Hilft bei typischen Fehlern, z.B. Skip-Links, Alt-Text-Warnungen, Kontrastverbesserung.
- **Accessibility Checker** von Equalize Digital: Gibt sofort Feedback im Editor.

8. Regelmäßige Tests durchführen

- Tools: **axe DevTools**, **NVDA Screenreader**, **VoiceOver (Mac)**
- **manuelle Tests** zusätzlich zu automatisierten Scans

Was sind Overlays

Overlays sind digitale Elemente oder Technologien, die über eine bestehende Website „gelegt“ werden, um bestimmte Funktionen hinzuzufügen – ohne den Quellcode der Website direkt zu verändern. Sie werden häufig eingesetzt, um Barrierefreiheit oder zusätzliche Benutzerfunktionen bereitzustellen.

Im Kontext Barrierefreiheit meint man mit „Overlays“:

Softwarelösungen, die versprechen, eine Website barrierefrei zu machen, indem sie z. B. eine Toolbar mit folgenden Funktionen hinzufügen:

- Kontrast erhöhen / Farben anpassen
- Schriftgröße ändern
- Vorlesen von Inhalten
- Tastaturnavigation erleichtern

Diese Overlays werden meist per JavaScript eingebunden und arbeiten clientseitig.

Kritik an Overlays:

- **Scheinsicherheit**: Sie suggerieren Barrierefreiheit, ohne strukturelle Probleme (z. B. fehlende Alternativtexte, unlogische Überschriftenstruktur, schlechte Tastaturbedienbarkeit) zu lösen.
- **Unzuverlässigkeit**: Viele Assistenztechnologien (z. B. Screenreader) funktionieren nicht gut mit Overlays oder ignorieren sie komplett.
- **Rechtlich problematisch**: In Ländern mit gesetzlichen Anforderungen (z. B. EU, USA) reichen Overlays oft nicht aus, um Konformität mit Barrierefreiheitsrichtlinien wie WCAG zu gewährleisten.
- **Ableismus durch Kontrolle**: Nutzer:innen wird oft eine spezielle Oberfläche aufgezwungen, anstatt ihnen zu erlauben, ihre eigenen Assistenztechnologien zu verwenden.

Beispiele für bekannte Overlay-Anbieter:

- accessiBe
- UserWay
- EqualWeb

Fazit:

Overlays können **kleine Komfortfunktionen** bieten, ersetzen aber **keine echte Barrierefreiheit**. Eine barrierefreie Website erfordert sauberen, semantischen Code, gute Struktur und umfassendes Testing – nicht nur ein Plugin.

Helfen Web Accessibility Overlays oder schaffen sie neue Probleme?

Diverse Zitate dazu:

„Entgegen anderslautender Versprechungen sind Accessibility Overlays Stand heute nicht in der Lage, eine Webseite von außen und quasi auf Knopfdruck umfassend barrierefrei zu gestalten. Sie versagen insbesondere im Bereich der Zugänglichkeit für blinde Menschen. Schlecht programmierte Overlay-Tools können im Gegenteil neue Barrieren erzeugen.

Vor diesem Hintergrund empfiehlt der DBSV, die Website selbst barrierefrei nach vorhandenen Normen und Standards zu gestalten. Ist dies gewährleistet, können Accessibility Overlays einen Mehrwert für spezifische Zielgruppen haben. Dabei muss das Overlay-Tool selbst und die mit ihm erzeugte Darstellung den Standards genügen.“

https://www.dbsv.org/aktuell/barrierefreiheit-overlay-tools-achtung-vor-falschen-werbeversprechen.html

---

„Accessibility-Overlay-Anbieter nutzen Unwissenheit und die Hoffnung auf eine schnelle Lösung aus und verfolgen dabei zumeist aggressives Marketing mit falschen Versprechungen. „

„Menschen mit Behinderung brauchen keine Accessibility-Overlays

Menschen mit Behinderung nutzen zumeist Hilfstechnologien oder spezielle Einstellungen in Betriebssystem oder im Browser. Sie brauchen diese Einstellungen auf jeder Website und wollen diese Einstellung nicht jedes Mal neu vornehmen.“

https://barrierekompass.de/aktuelles/detail/accessibility-overlays-aggressives-marketing-kaum-nutzen.html

---

„Overlays können viele wichtige Barrieren nicht beheben

Dies sind nur einige Beispiele. Overlays

- *fügen keine Untertitel hinzu,*
- *beheben keine ARIA-Fehler bei der Auszeichnung dynamischer Elemente,*
- *verbessern nicht die Tastaturinteraktion,*
- *verbessern keine Fehlermeldungen.“*

https://bitvtest.de/blog/detail/overlays-fuer-mehr-barrierefreiheit-warum-das-keine-gute-idee-ist