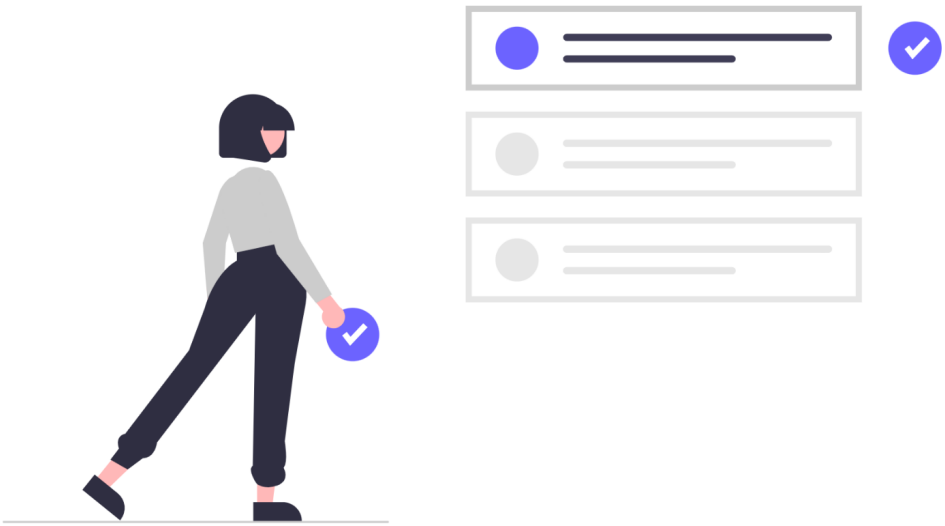
WCAG-Erfolgskriterien praxisnah erklärt:



Robustheit

Überwachungsstelle des Bundes für
Barrierefreiheit von Informationstechnik

Robustheit: Inhalte technologisch zuverlässig und zukunftssicher gestalten

Das WCAG-Prinzip **Robustheit** stellt sicher, dass digitale Inhalte von möglichst vielen Geräten, Browsern und assistiven Technologien zuverlässig verarbeitet werden können. Ziel ist es, Inhalte so technisch sauber und standardkonform aufzubauen, dass sie auch bei Weiterentwicklung von Technologien zugänglich bleiben.

Warum ist Robustheit so wichtig?

Technische Fehler in HTML-Strukturen oder falsche Angaben zu Rollen, Zuständen oder Namen von Elementen führen dazu, dass Hilfsmittel wie Screenreader Informationen nicht korrekt interpretieren können. Auch moderne Browser oder zukünftige Assistenzsysteme könnten Inhalte dann nicht mehr zuverlässig darstellen.

Was ist bei der Umsetzung zu beachten?

Robuste Inhalte setzen voraus, dass der Quellcode fehlerfrei und standardkonform ist. Dazu gehört z. B., dass HTML-Elemente korrekt verschachtelt und geschlossen sind und Rollen und Zustände bei interaktiven Elementen eindeutig ausgezeichnet sind. Auch ARIA-Attribute müssen korrekt verwendet werden.

Im Folgenden werden die Erfolgskriterien des Prinzips **Robustheit** praxisnah erläutert. Die Umsetzungsempfehlungen enthalten konkrete Hinweise für Redaktion, Entwicklung und Gestaltung. Dabei wird jedes Erfolgskriterium zunächst kurz erklärt. Im Anschluss folgen jeweils Hinweise zur technischen Umsetzung.

Die Erfolgskriterien sind in drei Stufen unterteilt:

- Stufe A umfasst grundlegende Anforderungen, die nötig sind, damit Webauftritte und Anwendungen überhaupt zugänglich sind.
- Stufe AA ist der empfohlene Standard für gute digitale Zugänglichkeit.
- Stufe AAA enthält zusätzliche Anforderungen für ein sehr hohes Maß an Barrierefreiheit.

Da öffentliche Stellen gemäß BITV 2.0 und Unternehmen gemäß BFSG zur Umsetzung der Stufen A und AA verpflichtet sind, konzentriert sich dieser Leitfaden ausschließlich auf diese beiden Konformitätsstufen. Die Kriterien der Stufe AAA können ergänzend berücksichtigt werden, wenn ein besonders hoher Barrierefreiheitsstandard erreicht werden soll.

Zur besseren Verständlichkeit findet ihr am Ende dieses Dokuments ein Glossar mit kurzen Erklärungen zentraler Begriffe rund um die folgenden Erklärungen.

Erfolgskriterien des Prinzips Robustheit

4.1 Kompatibel

4.1.1 Parsing (Stufe A):
Der Code einer Webseite sollte fehlerfrei sein und korrekt geparst werden können. Fehlerhafter HTML-Code kann dazu führen, dass Screenreader oder andere assistive Technologien Inhalte nicht oder falsch interpretieren. Jede ID auf einer Webseite darf nur einmal vergeben werden. Wird dieselbe ID mehrfach verwendet, kann das dazu führen, dass Bildschirmleseprogramme, Tastatursteuerungen oder andere Hilfsmittel nicht mehr eindeutig erkennen, welches Element gemeint ist.

Umsetzungsempfehlung: Nutzt HTML-Validatoren wie den W3C Markup Validation Service. Bezieht auch ARIA-Rollen und -Attribute in die Prüfung mit ein. Verwendet valide HTML-Syntax gemäß dem verwendeten Doctype.

4.1.2 Name, Rolle, Wert (Stufe A):
Alle benutzbaren Komponenten (z. B. Formulare, Links, Schaltflächen, Widgets) müssen programmatisch korrekt identifizierbar sein: mit Name, Rolle und aktuellem Status- oder Wertangabe. Screenreader und andere Hilfsmittel müssen erkennen, was ein Element ist (z. B. ein Button), wie es heißt („Suche starten“) und was sein aktueller Zustand ist (z. B. aktiviert/deaktiviert, ausgewählt, geöffnet). Ohne diese Informationen können Nutzende das Element nicht richtig bedienen oder einordnen.

Umsetzungsempfehlung: Nutzt ARIA-Rollen und -Attribute gezielt und korrekt: z. B. role="button", aria-expanded="true". Beschriftet Bedienelemente mit sichtbarem und programmatischem Namen (label, aria-label, aria-labelledby). Stellt sicher, dass der Zustand dynamischer Komponenten aktualisiert wird (aria-checked, aria-selected usw.)

4.1.3 Statusmeldungen (Stufe AA):
Statusmeldungen, die sich ohne Benutzerinteraktion ändern (z. B. Erfolgs- oder Fehlermeldungen), müssen von assistiven Technologien automatisch erkannt und übermittelt werden. Nutzende, die mit Screenreadern arbeiten, bekommen visuelle Änderungen oft nicht mit, z. B. wenn nach Absenden eines Formulars ein Hinweis erscheint: „Ihre Nachricht wurde gesendet.“ Ohne programmatische Erkennbarkeit ist die Information für sie nicht zugänglich.

Umsetzungsempfehlung: Verwendet ARIA-Live-Regionen für dynamische Statusupdates und platziert Live-Regionen semantisch sinnvoll.

Weiterführende Informationen aus der BFIT:

Barrierefreie Gestaltung von User Interface-Elementen

Tools zur Prüfung der Robustheit

1. HTML- und Code-Validierung:
    a. W3C Validator, prüft, ob der HTML- oder XHTML-Code valide und standardkonform ist. Link: https://validator.w3.org

    b. Nu HTML Checker; Alternative W3C-Prüfung (auch per Datei-Upload), erkennt strukturelle und semantische Fehler: Link: https://validator.w3.org/nu

2. ARIA-Rollen und Zustände überprüfen
    a. Tollwerk Bookmarklet: „ARIA-Struktur anzeigen“. Visualisiert ARIA-Rollen und Landmarken. Link: https://bitvtest.de/bookmarklets/landmarks.html

    b. WAVE Evaluation Tool: Das Tool unterstützt die Prüfung robuster technischer Umsetzung, indem es aufzeigt, wo der Code problematisch für die maschinelle Interpretation ist.. Link: https://wave.webaim.org/

Diese Liste erhebt nicht den Anspruch vollständig zu sein, es gibt noch viele andere Tools und Werkzeuge. Dies ist lediglich ein Einstieg.

Glossar:

- **ARIA-Label:** Ein Attribut zur barrierefreien Beschriftung von Elementen, insbesondere von Icons oder Schaltflächen, die keinen sichtbaren Text enthalten.
- **Assistive Technologien:** Hilfsmittel wie Screenreader, Braillezeilen oder Sprachausgabe, die Menschen mit Beeinträchtigungen die Nutzung digitaler Inhalte ermöglichen.
- **BFSG**: Barrierefreiheitsstärkungsgesetz.
- **BITV:** Barrierefreie-Informationstechnik-Verordnung.
- **Browser:** Programme zur Darstellung von Webseiten, z. B. Firefox, Chrome, Safari oder Edge.
- **Doctype:** Der Dokumenttyp (z. B. <!DOCTYPE html>) kennzeichnet den verwendeten HTML-Standard einer Webseite.
- **geparst / Parsing:** Das Einlesen und Interpretieren des Codes (z. B. HTML oder CSS) durch den Browser oder durch assistive Technologien.
- **HTML:** Hypertext Markup Language. Strukturierungssprache für Webseiten, z. B. zur Auszeichnung von Überschriften, Absätzen und Formularen.
- **HTML-Syntax:** Die formalen Regeln, nach denen HTML geschrieben wird.
- **ID:** Ein eindeutiger Bezeichner (Attribut id="...") innerhalb des HTML-Dokuments, mit dem einzelne Elemente gezielt angesprochen werden können.
- **JavaScript**: Skriptsprache zur Umsetzung dynamischer Inhalte und Interaktionen auf Webseiten. Muss barrierefrei integriert und steuerbar sein.
- **Programmatisch:** Der Begriff bezieht sich auf Informationen im Quellcode, die maschinell ausgelesen werden können, z. B. von Screenreadern.
- **Rollen, Zuständen oder Namen von Elementen**: Informationen, die assistive Technologien benötigen, um interaktive Elemente zu erkennen und zu beschreiben. Rollen (z. B. „Button“), Zustände (z. B. „aktiviert“, „deaktiviert“) und Namen (z. B. „Suchen“) können programmatisch über HTML oder ARIA vergeben werden.
- **Standardkonform:** Bedeutet, dass der Code den offiziellen Webstandards entspricht.
- **UX-Design:** Kurz für „User Experience Design“. Gestaltungsansatz, der auf gute Benutzerführung, Verständlichkeit und barrierefreie Bedienbarkeit abzielt.
- **Validatoren:** Werkzeuge zur Überprüfung des HTML- und CSS-Codes auf Einhaltung von Standards.
- **WCAG:** Die Web Content Accessibility Guidelines sind internationale Richtlinien zur digitalen Barrierefreiheit.