Digitale Barrierefreiheit

Die Barrierefreiheit von Webseiten ist tief im HTML-Standard verankert. Dennoch sind viele Webseiten für Menschen nicht nutzbar, wenn sie keine Standardhardware bedienen, nicht gut sehen, hören, komplexe Sachverhalte verstehen oder sich nicht gut konzentrieren können. Das sind viel mehr Nutzer:innen als landläufig angenommen. Bisher wurden Innovationszyklen von Webseiten vor allem visuell von Brandagenturen und Marketing oder durch technische Lösungen vorangetrieben, bei denen nur auf die „normalen“ Interaktionsmöglichkeiten Rücksicht genommen wurde. Da der Markt die Barrierefreiheit in 30 Jahren nicht vorangetrieben, sondern eher verhindert hat, sind in den USA und der EU die Gesetzgeber eingesprungen.

Der Stichtag ist der 28. Juni 2025 – das Datum liegt bei Erscheinen dieses Artikels noch knapp zwei Jahre in der Zukunft [1]. Ab diesem Tag müssen alle neu erstellten Webseiten barrierefrei sein, es sei denn, es handelt sich um ein Dienstleistungsangebot einer KMU mit weniger als 10 Mitarbeitenden und weniger als 2 Millionen Euro Jahresumsatz bzw. Jahresbilanzsumme. Für alle anderen Webseiten gilt eine Frist von 5 Jahren. Wenn in der Zwischenzeit das Angebot auf diesen Seiten verändert wird, gilt die Pflicht ab der Veröffentlichung dieser Veränderung. Alle anderen Seiten müssen ebenfalls bis zum 27. Juni 2030 barrierefrei überarbeitet werden [2]. Das will die EU mit der Richtlinie (EU) 2019/882 [3], auch EAA (European Accessibility Act) genannt, erreichen.

Warum wird digitale Barrierefreiheit zur Pflicht?

„Der Bedarf an barrierefreien Produkten und Dienstleistungen ist groß, und die Zahl der Menschen mit Behinderungen wird voraussichtlich noch deutlich steigen. Ein Umfeld mit besser

zugänglichen Produkten und Dienstleistungen ermöglicht eine inklusivere Gesellschaft und erleichtert Menschen mit Behinderungen ein unabhängiges Leben. Dabei sollte berücksichtigt werden, dass in der Union mehr Frauen als Männer eine Behinderung haben.“ So heißt es im zweiten Absatz der aufgeführten Gründe, die der Richtlinie vorangestellt wurden.

Weiterhin geht es darum, dass der Flickenteppich an Richtlinien, die derzeit in den Mitgliedsstaaten gelten, vereinheitlicht werden soll. Dadurch soll es einfacher werden, Produkte innerhalb der EU grenzüberschreitend zu vermarkten. Es geht auf der einen Seite also darum, Menschen mit Behinderung bewusst an der digitalen Welt teilhaben zu lassen. Das ist gut, wichtig und mehr als überfällig. Begründet wird dies aber auch mit dem wirtschaftlichen Aspekt von größeren Zielgruppen und Märkten. Dabei gibt es drei Richtlinien:

1. *WCAG* des W3C
2. *EN 301 549* der EU enthält viele der WCAG-Kriterien und ergänzt sie
3. *BITV* enthält die Übertragung der EN 301 549 in deutsches Recht

WCAG

WCAG steht für „Web Content Accessibility Guidelines“. Version 2.1 [4] gilt seit dem 05.06.2018. Version 2.2 hat den Status „Candidate Recommendation Draft“ [5]. Die finale Version wird noch 2023 erwartet.

Die WCAG 2.1 [6] enthält 78 Erfolgskriterien, die vier Prinzipien zugeordnet und in drei Konformitätsstufen unterteilt sind. Level A ist das absolute Minimum, das erfüllt werden muss, Level AA ist der Standard und Level AAA betrifft nur bestimmte Inhalte, die nicht auf allen Seiten Anwendung

finden.

In Tabelle 1 sehen Sie die Matrix, wie die Prinzipien und die Konformitätsstufen sich anhand der Quick-Referenz der WCAG 2.2 überschneiden [7].

4 Prinzipien	**14 Unterkapitel**	**78 Erfolgskriterien**	**3 Konformitätsstufen**		
			A	**AA**	**AAA**
1) perceivable/ wahrnehmbar	Textalternativen	1	1	–	–
	zeitbasierte Medien	9	4	1	4
	anpassbar	6	3	2	1
	unterscheidbar	13	2	7	4
	wahrnehmbar gesamt	**29**	**10**	**10**	**9**
2) operable/ bedienbar	mit der Tastatur zugänglich	4	3	–	1
	genug Zeit	6	2	–	4
	Krämpfe und physische Reaktionen	3	1	–	2
	navigierbar	10	4	3	3
	Eingabemodalitäten	6	4	–	2
	bedienbar gesamt	**29**	**14**	**3**	**12**
3) understandable/ verständlich	lesbar	6	1	1	4
	vorhersagbar	5	2	2	1
	Eingabehilfen	6	2	2	2
	verständlich gesamt	**17**	**5**	**5**	**7**
4) robust*	kompatibel	3	2	1	–
	robust gesamt	**3**	**2**	**1**	**0**
Gesamt		**78**	**31**	**19**	**28**

* robust: alle Inhalte müssen mit einer Vielzahl an Geräten und Browsern auch dann wahrnehmbar und bedienbar sein, wenn assistive Technologien, z. B. Screenreader, Braillezeilen, Bildschirmlupen, unterschiedliche Eingabemethoden (Maus/Touch/Fokus) genutzt werden. Außerdem gibt es unzählige adaptive Technologien/Einstellungsmöglichkeiten in Standardhardware, auf die dieser Grundsatz zutrifft.

Tabelle 1: Die WCAG-Erfolgskriterien mengenmäßig den Prinzipien und Konformitätsstufen zugeordnet

EN 301 549

EN steht für „Europäische Norm“. Die Nummer 301 549 enthält die „Accessibility requirements for ICT products and services“, die „Barrierefreiheitsanforderungen für IKT-Produkte und -Dienste“. IKT steht für Information- und Kommunikationstechnik. Version 3.2.1 gilt seit dem 21.05.2021.

BITV 2.0

BITV steht für „Barrierefreie-Informationstechnik-Verordnung“. Version 2.0 gilt seit dem 21.05.2019.

Die Prüfschritte der BITV 2.0 [8] orientieren sich an der EN 301 540 Version 3.1.1. und 3.2.1. Die Nummern von BITV-Prüfschritten, die auf der EN 301 549 basieren, sind weitgehend mit der Nummerierung dort identisch. Manche der EN-301-549-Prüfschritte werden in der BITV jedoch in mehrere Einzelschritte unterteilt.

Auf der Seite barrierefreiheit-dienstekonsolidierung.bund.de wird klargestellt: Die WCAG-Erfolgskriterien der Konformitätsstufen A und AA sind damit mit dem Standard EN 301 549 verbindlich einzuhalten. Die WCAG-Erfolgskriterien der Konformitätsstufe AAA werden im Standard EN 301 549 informatorisch beziehungsweise als erweiterte Kriterien aufgelistet, die nicht für alle Inhalte einer Webseite relevant sind (Tabelle 2) [9].

Kapitel	Kriterien	EN 3.1.1	EN 3.21	WCAG 2.1 Level		
				A	AA	AAA
1. allgemeine Anforderungen	3	3	–	–	–	–
2. Zwei-Wege-Sprachkommunikation	16	14	2	–	–	–
3. Videofähigkeiten	9	7	2	–	–	–
4. Textalternativen	4 (1 nur BITV)	=	=	3	–	–
5. zeitbasierte Medien	5 (1 nur BITV)	=	=	2	2	–
6. anpassbar	12	=	=	10	2	–
7. unterscheidbar	9 (1 A + AA)	=	=	2	8	–
8. per Tastatur zugänglich	3	=	=	3	–	–
9. ausreichend Zeit	2	=	=	2	–	–
10. Anfälle	1 (2*AAA extra)	=	=	1	–	(2)
11. navigierbar	7	=	=	5	2	–
12. Eingabemodalitäten	4	=	=	4	–	–
13. lesbar	2	=	=	1	1	–
14. vorhersehbar	4	=	=	2	2	–
15. Hilfestellung bei der Eingabe	4	=	=	2	2	–
16. kompatibel	3	=	=	2	1	–
17. benutzerdefinierte Einstellungen	1	1	=	–	–	–
18. Autorenwerkzeuge	4	4	=	–	–	–
19. Dokumentation und Support	5	5	=	–	–	–
Gesamt	**98**	**34**	**4**	**37**	**20**	**(2)**

Tabelle 2: Die BITV-Prüfkriterien mengenmäßig den Kapiteln und

den EN-301-549- und WCAG-Konformitätsstufen zugeordnet

Die EN 301 549 und die BITV enthalten einige Punkte, die in der WCAG nicht enthalten sind und umgekehrt. Zum Beispiel fehlen in der BITV einige Punkte der WCAG zur Verständlichkeit. Diese wiederum sind in Leichter Sprache sehr wichtig (Kasten: „Beispiel Leichte Sprache“).

Beispiel Leichte Sprache

In Bezug auf die Leichte Sprache und wie diese generell gut für die Zielgruppe, aber auch grundsätzlich barrierefrei eingebunden werden kann, sind die Regeln nicht ausgereift. Die gesetzlichen Vorgaben dazu sind derart detailliert festgeschrieben, dass sie die allgemeine Barrierefreiheit im Web komplett torpedieren: „Texte werden linksbündig ausgerichtet. Jeder Satz beginnt mit einer neuen Zeile. Der Hintergrund ist hell und einfarbig.“

Das berücksichtigt weder den Bedarf nach Dark Mode, noch die Tatsache, dass von der Zielgruppe eher Smartphones als Computer für den Webzugang genutzt werden, auf denen die oft genutzten PDF-Dateien zu klein dargestellt und visuelle Einstellungen der Browser ignoriert werden.

Leider sind die BITV- und EN-301-549-Kriterien, die nicht in der WCAG auftauchen, keinen Konformitätsstufen zugeordnet. Dennoch gilt es, sie ebenfalls zu erfüllen, um den BITV-Test zu bestehen, sofern sie auf die jeweils geprüfte Seite anwendbar sind.

Sind Angebote, die nach der Prüfung barrierefrei sind, auch garantiert inklusiv?

Nein.

Zum einen entwickelt sich die Technik schneller weiter als es in Regulierungen fixiert werden kann. So sind z. B. automatische Anpassungen an Einstellungen der Nutzenden möglich, die von der WCAG auch in Version 2.2 noch nicht erfasst wurden, jedoch in der BITV als Prüfschritt 11.7 „Benutzerdefinierte Einstellungen“ [10] enthalten sind, leider ohne Konformitätsstufe. Zum anderen kommen auch immer wieder neue Erkenntnisse hinzu.

Ein paar Dinge sind aktuell weder gesetzlich gefordert noch automatisch anpassbar: Es gibt keine Vorgabe, dass alle aktiven Elemente mit nur einer ggf. in der Beweglichkeit stark eingeschränkten Hand vom Bildschirmrand aus gut erreichbar sein müssen.

Die Prüfschritte basieren oft auf einzelnen Einschränkungen. Die Anforderungen, die Mehrfachbehinderungen mit sich bringen, werden nicht durchgängig abgedeckt. So wird zwar berücksichtigt, dass Menschen, die weder sehen noch hören können, mit der Braillezeile auf Inhalte zugreifen können müssen. Aber dass das auch für Texte in Leichter Sprache gelten sollte, wird bisher nicht beachtet. Menschen, die gebärden und auf eine weniger komplexe oder stärker kontrastierte Gebärdensprache angewiesen sind, fallen gänzlich durchs Raster.

Auch was die Verständlichkeit von interaktiven Systemen allgemein betrifft, muss die Lücke zwischen Usability und Barrierefreiheit noch stärker erforscht werden.

Wie groß ist die Zielgruppe?

Die Schätzungen, wie viele Menschen mit einer permanenten Behinderung leben, gehen weit auseinander. Die meisten Zahlen liegen zwischen 10 und 15 Prozent der Weltbevölkerung, in Industrieländern mehr als in Entwicklungsländern. In den USA ging das CDC, das Centers for Disease Control and Prevention, kürzlich gar von 27 Prozent aus [11].

Nur 3-4 Prozent der Behinderungen sind angeboren. Alle anderen Behinderungen werden erst im Laufe des Lebens erworben bzw. treten erst später in Erscheinung. Dabei kann es sich um Unfälle, Krankheitsfolgen, Umwelteinflüsse und nicht zuletzt zunehmendes Alter handeln.

Das Statistische Bundesamt hat im Juni 2022 eine Quote von 9,4 Prozent veröffentlicht – als schwerbehindert gelten dabei „Personen, denen die Versorgungsämter einen Behinderungsgrad von mindestens 50 zuerkannt sowie einen gültigen Ausweis ausgehändigt haben.“ [12]. Genaue Zahlen sind das nicht. Das Beispiel Blindheit zeigt das Problem ganz gut: „Blinde und sehbehinderte Menschen werden in Deutschland nicht gezählt.“ [13].

Manchmal sind es auch nicht die primär anerkannten Behinderungen, die zu einer Einschränkung führen, die für die digitale Barrierefreiheit relevant ist. Auch Menschen mit kurzzeitigen Einschränkungen, zum Beispiel Verletzungen, die wieder komplett ausheilen, und Menschen die sich situativ in ihren Bewegungen, in ihrer Sicht, ihrer Hörfähigkeit oder anderweitig eingeschränkt sind, profitieren von Maßnahmen, die im Rahmen der digitalen Barrierefreiheit umgesetzt wurden.

Von einer (einheitlichen) Zielgruppe kann in diesem Zusammenhang also nicht gesprochen werden. Aber das ist sekundär. Die Vergangenheit zeigt immer wieder Beispiele, dass Erfindungen, die ursprünglich für Menschen mit bestimmten Behinderungen erfunden wurden, allen Menschen nutzen bzw. von vielen Menschen angenommen werden. Dieses Phänomen wird „Curb Cut Effect“ [14] genannt, weil abgesenkte Bordsteinkanten und großzügigere Bewegungsflächen im öffentlichen Raum allen Menschen zugutekommen, die sich dort bewegen. Barrierefreiheit, wenn sie gut umgesetzt wurde, ist für uns alle gut.

Was sind die häufigsten Barrieren im Web?

WebAIM („Web Accessibility In Mind“) ermittelt seit 2019 in der Studie „The WebAIM Million“ [15] jährlich, wie viele der Top-1 000 000-Webseiten Barrieren aufweisen und welche am häufigsten sind. 96,3 Prozent der untersuchten Seiten haben 2023 Probleme in der Barrierefreiheit aufgewiesen. Seit 2019 ist der Wert nur um 1,5 Prozent (von 97,8 Prozent) gesunken.

Hier die Top-6-Fehlerquellen, die auf den Seiten zu finden waren:

1. 83,6 Prozent der Seiten hatten Schrift mit zu geringem Farbkontrast zum Hintergrund.
2. 58,2 Prozent der Seiten enthielten Bilder ohne Alternativtexte, die den Inhalt beschreiben.
3. 50,1 Prozent der Seiten enthielten „leere“ Links, die Icons darstellen, aber keinen Text.
4. 45,9 Prozent der Seiten enthielten Eingabefelder, die nicht korrekt beschriftet waren.
5. 27,5 Prozent der Seiten enthielten Buttons, die leer bzw. nicht korrekt beschriftet waren.
6. 18,6 Prozent der Seiten haben keine korrekte Sprachauszeichnung enthalten.

Die genannten Fehlerquellen beziehen sich in erster Linie auf Fehler, die sich vor allem für Menschen negativ auswirken, die nicht über 100 Prozent Sehkraft verfügen. Barrieren für Nutzer:innen mit Behinderungen, die das Hören, kognitive oder psychische Fähigkeiten oder die Feinmotorik (diese Liste ließe sich fortsetzen) betreffen, tauchen in dieser Liste noch nicht mal auf.

Der Curb Cut Effect im Internet – angenehmere Nutzung für alle!

Die wirklich gute Nachricht ist, dass digitale Barrierefreiheit auch auf andere Aspekte des Internets einen guten Effekt hat. Da wäre zum Beispiel: Stress. Die Internetagentur Cyber Duck hat 2020 eine Studie mit 1 100 „gesunden“ User:innen durchgeführt, die alle nichts mit dem Erstellen von Webseiten zu tun und sich als souveräne Anwender:innen beschrieben haben [16]. Im Test wurde der Anstieg des systolischen Blutdrucks als Stressindikator gemessen, wenn die Testwebseiten bestimmte Probleme aufgewiesen haben. Diese lassen sich fast alle auch Barrierefreiheitskriterien zuordnen (Tabelle 3).

Ungewollte Wechselwirkungen mit anderen Themenbereichen

Generell ist bei der Barrierefreiheit oft des einen Freud des anderen Leid, wie Sie an den folgenden drei Beispielen sehen:

- *Beispiel Schriftgröße:* Es ist nicht einfach damit getan, die Schrift auf einer Webseite doppelt so groß wie üblich zu machen. Das würde vor allem für Menschen, die nur mühsam scrollen können, das Nutzungserlebnis verschlechtern, wenn sie gut sehen können oder gar kleine Schrift bevorzugen. Hier soll es aber auch vor allem um Wechselwirkungen mit Themenbereichen aus Sicht der Betreibenden gehen, denn manchmal geraten die einzelnen Themen miteinander in Konflikt.
- *Beispiel SEO:* Wenn man SEO-Alternativtexte von Bildern für das Ranking nutzen möchte, der SEO-Text aber für blinde Nutzer:innen keinen Informationswert [17] enthält, da sie eine objektive Beschreibung bevorzugen, gilt es abzuwägen, was wichtiger ist. Hier kann ggf. das

HTML-Element <*figure*> weiterhelfen und die <*figcaption*> hinzugezogen werden, um Informationen für SEO einem Bild hinzuzufügen.

- *Beispiel DSGVO/Cookie-Banner-Pflicht:* Cookie-Banner nerven uns alle. Vor allem, wenn sie als Pop-up daherkommen, führen sie zu vermehrtem Stress. Für viele sind sie einfach lästig. Wer sich um die eigenen Daten sorgt, möchte oft nicht zustimmen und hat so manches verwirrende Interaktionsmuster gesehen, mit dem Nutzer:innen doch noch ein Einverständnis abgerungen werden soll. Abgesehen davon, dass hier oft schon sogenannte Dark Patterns, speziell das „Privacy Zuckering“ [18] zum Einsatz kommen, gibt es noch ganz konkrete Probleme mit der Barrierefreiheit, speziell mit der Tastaturnavigation. Da der Code für das Overlay oft am Ende der Seite eingebunden wird, müssen erst alle Links, die visuell hinter dem Overlay liegen, übersprungen werden, bis das Overlay erreicht wurde und durch Auswahl einer Option entfernt werden kann. Blinde Nutzer:innen können die Inhalte dabei normal vom Screenreader vorlesen lassen. Sehende Menschen, die nur die Tastaturnavigation nutzen können, können hingegen die Inhalte derweil nicht sehen. Egal wie sorgfältig vorher auf die Barrierefreiheit geachtet wurde: Einmal ein falsches Plug-in gewählt und schon ist die Seite wieder eine einzige Barriere.

Kann eine Webseite nachträglich barrierefrei gemacht werden?

Bedingt.

Es hängt vor allem davon ab, mit welcher Technik die Seite erstellt wurde. Am einfachsten ist es bei handgeschriebenen Seiten, die nicht von der Barrierefreiheit der verwendeten Frameworks abhängig sind. Webseiten, die auf Content-

Management-Systemen wie WordPress aufgesetzt sind, werden immer nur so barrierefrei sein wie die verwendeten Themes und Plug-ins. Hier sind die Anbieter gefragt. Leider ist es möglich, ein gutes Level an Barrierefreiheit mit der Wahl eines einzigen falsch gewählten Plug-ins zurück auf komplett nicht barrierefrei zu setzen. Das gilt auch für nicht geprüfte Updates und insbesondere auch für Overlay-Tools.

Zum Beispiel mit einem Overlay-Tool?

Nein.

Overlay-Tools sind kein Garant für Barrierefreiheit! Oftmals können Overlay-Tools Seiten, die bereits recht barrierearm waren, sogar komplett unzugänglich machen, wie die Seite Overlay Factsheet zusammengestellt hat [19].

Außerdem kann es zusätzlich zu Verstößen gegen die DSGVO kommen, wenn das eingesetzte Tool Userdaten zum Beispiel außerhalb der EU sammelt.

Beispiel für einen misslungenen Einsatz eines Overlay-Tools

Eine Seite mit hektischen Videos erfüllt die Level-A-Kriterien BITV 9.2.2.2 „Bewegte Inhalte abschaltbar“ [20] und ggf. auch 9.2.3.1 „Verzicht auf Flackern“ [21] nicht, wenn es nicht pausierbar ist und darin Elemente häufiger als dreimal die Sekunde aufblitzen. Das kann für Menschen unangenehm werden, die durch Flackern ein Anfallrisiko haben. Aber auch Menschen mit Seheinschränkungen, Neurodivergenz und andere, die von Bewegungen stark abgelenkt werden, können damit Probleme haben. Es fällt ihnen schwer, den Link zum Öffnen des Overlays überhaupt zu identifizieren, um dort die Einstellung zu finden, die das Video pausiert.

Was genau ist ein Overlay-Tool und wie funktioniert es?

Ein Overlay-Tool verspricht, Webseiten dadurch barrierefrei zu machen, dass sie um Funktionen wie Schrift- und Farbeinstellungen ergänzt werden. Diese Einstellungen können teilweise in Profilen gespeichert werden. Problematisch ist vor allem, dass

- es sich um einen proprietären Ansatz handelt – welche Lösung eingebunden wird, ist vom Anbietenden abhängig, nicht von der Wahl der Nutzenden.
- weiterer Code übertragen und ausgeführt werden muss.
- ein zusätzlicher Log-in nötig ist.
- bestehende adaptive Einstellungen der Nutzer:innen ignoriert werden.
- die Overlays die Barrierefreiheit letztlich nicht garantieren können, sondern den Zugang eher erschweren.

Die Message hinter Overlay-Tools ist daher oft: „Wir wissen, dass unsere Website barrierefrei sein sollte, darum haben wir das Overlay eingebunden – aber eigentlich ist uns egal, ob sie wirklich für alle nutzbar ist.“ Besser:

- auf das Video verzichten
- das Video nicht automatisch starten lassen, vor allem nicht, wenn im Code nicht abgefragt wird, ob „Bewegungen reduzieren“ im Betriebssystem aktiviert wurde.
- ein Overlay-Tool nur dann ergänzend einsetzen, wenn alle automatischen Adaptionsmöglichkeiten ausgeschöpft wurden

Warum wurde meine Webseite nicht

längst barrierefrei umgesetzt?

Im Idealfall werden Webseiten zum Zeitpunkt ihrer Erstellung zum dann gültigen Stand der Technik umgesetzt. Leider sehe ich auch noch über 13 Jahre nach der Veröffentlichung des Artikels „Responsive Web Design [22]“, dass neue Webseiten nicht responsiv umgesetzt werden und damit auf verschiedenen Geräten unbrauchbar sind.

Laut BITV verstoßen sie damit nicht nur gegen gängige Marktstandards, sondern auch gegen 9.1.4.10 „Inhalte brechen um“ [23] (AA) und 9.1.3.4 „Keine Beschränkung der Bildschirmausrichtung“ [24] (AA).

Dass das responsive Internet letztlich seinen Durchbruch hatte, lag nicht zuletzt daran, dass Google mobile Webseiten ab dem 21. April 2015 bei der mobilen Suche bevorzugt hat [15] und ab März 2021 auf Mobile-First-Index [26] umgestellt hat.

Aber das ist nur die Spitze des Eisbergs: Auf der einen Seite wurde das Studium von Medieninformatiker:innen durch die Einführung des Bachelors um ein Jahr verkürzt. Auf der anderen Seite kamen zusätzlich zu den Veränderungen der Standards von HTML und CSS auch jährlich neue Frameworks für CSS (z. B. Bootstrap, Tailwind) und JavaScript (z. B. jQuery) auf den Markt. Teilweise wurden sie sofort an den Hochschulen gelehrt, die Standards darüber vernachlässigt. Kompatibilität zwischen verschiedenen Browsern (Stichwort Vendor-Prefix, Modernizr) und immer neue gestalterische Ideen zu ermöglichen waren lange Zeit einfach schicker als die Berücksichtigung von Barrierefreiheitsprinzipien.

Was genau muss geändert werden?

Das kommt ganz auf Ihre Seite an. Manchmal sind es nur Kleinigkeiten, manchmal Kleinigkeiten, die sich läppern, teilweise lautet die Antwort „am besten neu kodieren“,

manchmal aber auch „es muss alles ab dem Konzept neu“. Wie Sie das anfangen und auf welche Details geachtet werden muss, wird an dieser Stelle nach und nach Thema sein.

Wer soll das alles umsetzen?

Wir sind alle gefragt! Wir müssen die Menschen, mit denen wir zusammenarbeiten bzw. die wir beauftragen, zunächst für das Thema sensibilisieren. Dann müssen sich alle passend zu ihrer Rolle weiterbilden:

- *Alle Rollen* müssen sich mit den Prüfkriterien und den realen Anforderungen der digitalen Barrierefreiheit vertraut machen, um entsprechend darauf reagieren zu können. Es ist wichtig, dass dieses Thema nicht nur an einer Person im Team liegt, die dann immer nur sagen kann, was falsch ist.
- *Product-Owner:innen* müssen Barrierefreiheit mit in die Akzeptanzkriterien der User Stories aufnehmen, damit alle Teammitglieder das Thema immer berücksichtigen lernen.
- *UX-Designer:innen* müssen sich bewusst machen, welche Interaktionspatterns barrierefrei sind.
- *UI-Designer:innen* müssen nicht nur für unterschiedliche Bildschirmgrößen gestalten, sondern auch für unterschiedliche Größen der Schriften und Klickflächen, sowie für unterschiedliche Farbvarianten etc.
- *Frontend-Entwickler:innen* müssen sich präziser mit UX/UI-Designer:innen abstimmen, um sicherzustellen, ob die Übergabewerte korrekt sind und was bereits berücksichtigt wurde. Außerdem muss das bestehende Wissen um HTML-Elemente, Attribute, CSS-Properties und JavaScript-Methoden mit den Anforderungen der Barrierefreiheit abgeglichen werden. Nicht alle Methoden, die zwischendurch (inoffizieller) Industriestandard waren, können für barrierefreie

Webseiten eingesetzt werden.

- *Backend-Entwickler:innen* müssen wissen, wie sie bestimmte Elemente verfügbar machen können. Zum Beispiel, damit Redakteur:innen nur eine H1 anlegen können oder Text, der fett dargestellt wird, nicht als <*b*>, sondern als <*strong*> ausgezeichnet wird. Schließlich heißt es in den BITV-Prüfkriterien: „Wenn es sich bei der zu testenden Webanwendung um ein Autorenwerkzeug handelt, soll die Anwendung die Erstellung von barrierefreien Dokumenten erlauben und den Nutzer dabei unterstützen.“ [27]
- *SEO-Expert:innen* müssen den Spagat zwischen barrierefreien und SEO-optimierten Titeln und Bildbeschreibungen im Alt-Text hinbekommen. An vielen Stellen können sie nun aber auch darauf verweisen, dass ihre Arbeit für SEO und Barrierefreiheit gut und wichtig ist.
- *Redakteur:innen* müssen lernen, Dokumente in Word und anderen Programmen so anzulegen, dass daraus barrierefreie PDFs erzeugt werden können. Sie müssen wissen, welche semantischen Auszeichnungsmöglichkeiten es gibt und sie konsequent anwenden. Außerdem benötigen Sie Kontakte zu Übersetzer:innen für Leichte Sprache und/oder einen Zugang zu entsprechender KI [28]. Gleiches gilt für Videos, die den Text in Deutsche Gebärdensprache (DGS) übersetzen, bzw. in die Gebärdensprachen der Zielländer. Auch hier sind KI-gestützte Avatare in der Vorbereitung, es gilt jedoch neben dem wirtschaftlichen Aspekt auch die Akzeptanz der Gebärdenden zu berücksichtigen, die dieser Technik 2023 noch ablehnend gegenüberstehen [29].
- *Bildredakteur:innen* müssen nicht nur lernen, Bilder nach deren inhaltlicher Verständlichkeit zu bewerten, sondern auch, wie Personen deutlicher hervorgehoben werden können und wie optimale Alt-Texte geschrieben werden. Diese konsequent auch für Social-Media-Bilder einzusetzen, muss zur Gewohnheit werden.

- *Qualitätstester:innen* müssen wissen, wie sie die unterschiedlichen Barrieren ausfindig machen können, um sie melden und die Korrektur prüfen zu können.

Nicht zu vernachlässigen ist eine Dokumentation der Maßnahmen, die getroffen wurden. Zum einen wird es ab 2025 im Rahmen der EAA für manche Seitentypen Pflicht sein, sie zu führen, zum anderen ermöglicht es Ihnen auch eine einfachere Übergabe von Projekten an andere oder neue Teammitglieder.

Shopify ist dies bereits einmal misslungen. Die Plattform wird in einem Artikel über aria-current als gutes Beispiel genannt, weil dort die aktuelle Seite in der Navigation korrekt als *aria-current="page"* [30] ausgezeichnet wurde. Dem ist inzwischen nicht mehr so. Nur noch die aktuell ausgewählte Sprache und Region wird im Footer mit *aria-current="true"* ausgezeichnet. Welche Seite im Menü die derzeit angezeigte ist, geht für Screenreader-Nutzer:innen nur aus dem *<title>* hervor.

Fazit

Sie werden nicht um das Thema herumkommen. Sie können jetzt langsam mit digitaler Barrierefreiheit anfangen, oder es in zwei Jahren unter Zeitdruck tun, wenn die Konkurrenz an Ihnen vorbeizieht. Dabei ist es egal, ob Ihr Produkt eine Komponente, ein Tool oder eine Plattform ist. Je eher Sie anfangen, umso größer wird Ihr Vorsprung sein. Nicht zu vernachlässigen ist auch der soziale Aspekt: Im Idealfall hilft es uns als Gesellschaft, mehr Verständnis füreinander und unsere unterschiedlichen Voraussetzungen zu erhalten, die bisher einfach als Unzulänglichkeit Einzelner abgetan wurden.

Problem	**%**	**Widerspricht BITV … (Level nach WCAG)**
langsam ladende Seiten	21	–

Problem	%	Widerspricht BITV … (Level nach WCAG)
mehrere Pop-ups		9.1.4.13 Eingeblendete Inhalte bedienbar (AA)
automatisch abspielende Musik	20	9.1.4.2 Ton abschaltbar (A)
kaputte Seiten (404 Error)	17	WCAG-Prinzip Robust (A und AA)
automatisch abspielende Videos	mArKeD mArKeD mArKeD 1 mArKeD6	9.2.2.2 Bewegte Inhalte abschaltbar (A) 9.2.3.1 Verzicht auf Flackern (A/AA)
mArKeD mArKeD mArKeD nicht klickbare Button mArKeDs	14	9.1.1.1a Alternativtexte für Bedienelemente (A) 9.4.1.1 Korrekte Syntax (A)
schwer lesbare Schrift	13	9.1.4.3 Kontraste von Texten ausreichend (AA)
Bilder, die nicht laden (und keine Alt-Tags haben)	12	WCAG-Prinzip Robust (A und AA) 9.1.1.1b Alternativtexte für Grafiken und Objekte (A)
Bilderkarussells	10	9.1.1.1a Alternativtexte für Bedienelemente (A) 9.4.1.2 Name, Rolle, Wert verfügbar (A)
ablenkende Animationen	5	9.2.3.1 Verzicht auf Flackern (A) 9.2.2.2 Bewegte Inhalte abschaltbar (A)

Tabelle 3: Die Stressfaktoren im Internet den Anforderungen der Barrierefreiheit zugeordne

Annika Brinkmann, Web-Designerin seit 2003, sensibilisiert und schult Teams, die an Konzeption, Gestaltung, Programmierung und Redaktion barrierefreier Webseiten beteiligt sind. Auf Barrieren-fasten.de bietet sie

Entwickler:innen einen niedrigschwelligen Einstieg.

Links & Literatur

[1] https://online-accessibility-countdown.eu/

[2] https://gehirngerecht.digital/digitale-barrierefreiheit-pflicht-wissen/

[3] https://eur-lex.europa.eu/legal-content/DE/TXT/HTML/?uri=CELEX:32019L0882

[4] https://www.w3.org/TR/WCAG21/

[5] https://www.w3.org/TR/WCAG22/

[6] https://www.w3.org/WAI/WCAG21/quickref/ wurde bereits 2018 finalisiert

[7] https://www.w3.org/WAI/WCAG22/quickref/ wird noch um neue Anforderungen ergänzt

[8] https://ergebnis.bitvtest.de/pruefverfahren/bitv-20-web

[9] https://www.barrierefreiheit-dienstekonsolidierung.bund.de/Webs/PB/DE/gesetze-und-richtlinien/en301549/en301549-node.html

[10] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/11-7-benutzerdefinierte-einstellungen

[11] https://www.cdc.gov/ncbddd/disabilityandhealth/infographic-disability-impacts-all.html

[12] https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Behinderte-Menschen/_inhalt.html

[13] https://www.dbsv.org/zahlen-fakten.html

[14] https://accessibleweb.com/civil-rights/the-curb-cut-effect-7-ways-the-ada-is-for-everyone/

[15] https://webaim.org/projects/million/

[16] https://www.netimperative.com/2020/12/09/blood-pressure-study-which-website-issue-cause-users-the-most-stress/ leider ist nur noch Sekundärliteratur online, bei Cyber-duck selbst taucht die Studie nicht mehr auf: https://www.cyber-duck.co.uk/

[17] https://www.dbsv.org/bildbeschreibung-4-regeln.html

[18] https://de.wikipedia.org/wiki/Dark_Pattern#Beispiele_f%C3%BCr_Dark_Patterns

[19] https://overlayfactsheet.com/

[20] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/9-2-2-2-bewegte-inhalte-abschaltbar

[21] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/9-2-3-1-verzicht-auf-flackern

[22] https://alistapart.com/article/responsive-web-design/ von Ethan Marcotte erschien am 25. Mai 2010

[23] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/9-1-4-10-inhalte-brechen-um

[24] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/9-1-3-4-

keine-beschraenkung-der-bildschirmausrichtung

[25] https://www.googlewatchblog.de/2015/02/mobile-websuche-apps-ranking/

[26] https://ebakery.de/google-mobile-first-index/

[27] https://ergebnis.bitvtest.de/pruefschritt/bitv-20-web/11-8-2-barrierefreie-erstellung-von-inhalten

[28] https://summ-ai.com/

[29] https://www.br.de/nachrichten/deutschland-welt/gehoerlose-aeussern-kritik-an-gebaerdensprach-avataren,TfMrXDf?mc_cid=f07dedcb3d&mc_eid=712cfaf1f0

[30] https://www.aditus.io/aria/aria-current/

Progressive Web Apps barrierefrei entwickeln

entwickler.de – entwickler.de Deine Wissensplattform

[…]Weiterlesen…
von Markus Lemcke

Öffentliche Stellen des Bundes sind seit September 2016 gesetzlich dazu verpflichtet, barrierefreie Apps

einzusetzen [1]. Progressive Web Apps haben dabei eine großen Vorteil gegenüber nativen Apps: Bei einer nativen App gibt es für drei unterschiedliche Betriebssysteme drei unterschiedliche Richtlinien. Wenn PWAs barrierefrei nach der EN 301 549 [2] entwickelt werden, kann eine App auf drei unterschiedlichen Betriebssystemen eingesetzt werden und die Barrierefreiheit ist auf allen dreien gleich gut.

Seit dem 1. Mai 2002 gibt es das Behindertengleichstellungsgesetz (BGG). Dieses Gesetz wurde im Jahr 2016 überarbeitet und wurde im September 2016 neu verabschiedet. Seit September 2016 sind öffentliche Stellen des Bundes nach § 12a Barrierefreie Informationstechnik Absatz 1 zur Barrierefreiheit bei Apps verpflichtet. Entsprechend haben auch die drei großen Unternehmen Google [3], Apple [4] und Microsoft [5] Richtlinien zur barrierefreien Appentwicklung veröffentlicht.

Bei der Überlegung, wie viele Menschen von barrierefreier Appentwicklung profitieren, stellt man fest, dass es nicht nur 7,8 Millionen schwerbehinderte Menschen [6] betrifft, denn auch einigen der 18,3 Millionen Senioren [7] können barrierefreie Apps das Leben erleichtern.

Vorteile von PWAs

Es gibt native Apps und Progressive Web Apps (PWAs). Als native Apps werden Anwendungen auf mobilen Endgeräten bezeichnet, die speziell für das Betriebssystem des jeweiligen Endgeräts konzipiert und entwickelt wurden. Sie werden meist über die App Stores, die an das Betriebssystem gekoppelt sind, als kostenfreie oder auch kostenpflichtige Anwendungen vertrieben. PWAs hingegen werden mit HTML, CSS und JavaScript entwickelt, sind somit plattformunabhängig und können auf Desktop- und Mobile-Betriebssystemen eingesetzt werden.

Richtlinien, die dafür sorgen, dass Apps von allen Menschen mit unterschiedlichen körperlichen Einschränkungen bedient

werden können, haben so viele Prüfungskriterien, wie notwendig sind, um alle körperlichen Beeinträchtigungen zu berücksichtigen. Die Anzahl der Prüfungsschritte bei den Richtlinien von Google, Apple und Microsoft sind unterschiedlich. Das bedeutet, wenn bei einer Richtlinie Prüfungsschritte fehlen, werden Menschen mit bestimmten körperlichen Beeinträchtigungen ausgrenzt.

PWAs sind plattformunabhängig, sie können also auf allen Betriebssystemen eingesetzt werden. Wenn eine PWA nach der EN 301 549 barrierefrei entwickelt wird, ist sie auf allen Betriebssystemen für alle Menschen mit körperlichen Beeinträchtigungen barrierefrei [8].

https://phpconference.com/session-qualification/ipc-webentwicklung/?layout=contentareafeed&widgetversion=1&utmtrackerversion=1&seriesId=oLGXGfBxxeHKh6rMj
Um es auf den Punkt zu bringen: Die Entwicklung von barrierefreien PWAs sorgt dafür, dass alle Menschen mit unterschiedlichen körperlichen Beeinträchtigungen eine App auf jedem Betriebssystem bedienen können.

Richtlinie EN 301 549

Der Standard für Barrierefreiheit von Webinhalten, den das W3C 1999 eingeführt hat und der seitdem global angewendet wird, sind die Web Content Accessibility Guidelines (WCAG). Diese internationale Richtlinie wird kontinuierlich weiterentwickelt und erfuhr 2018 mit den WCAG 2.1 [9] ihre vorerst letzte Aktualisierung.

Die WCAG in ihrer aktuellen Fassung ist die Grundlage für die europäische Norm EN 301 549. Allerdings geht es in der EN 301 549 nicht nur um Barrierefreiheit im Web, sie beinhaltet ebenso Barrierefreiheit bei

- Hardware
- Nicht-Web-Dokumente

- Software
- Dokumentation und unterstützende Dienste

Die EN 301 549 ist also die erste Richtlinie, die der Komplexität des Themas digitale Barrierefreiheit Rechnung trägt und ist deswegen umfassender.

Um den Unterschied zwischen EN 301 549 und WCAG 2.1 deutlich zu machen, hier ein Beispiel: Wenn eine Webseite in Frankreich barrierefrei gemacht werden soll, dann kommt die EN 301 549 zur Anwendung. Bei einer Webseite in Amerika kommt hingegen die WCAG 2.1 zur Anwendung.

Die EN ist eine Richtlinie für Webprogrammierung. Deswegen ist es keine gute Idee, native Apps, die mit Java oder Swift entwickelt werden, nach der EN 301 549 barrierefrei zu entwickeln. PWAs nach der EN 301 549 barrierefrei zu machen, passt perfekt, weil diese mit HTML, CSS und JavaScript entwickelt werden.

Einzelne Kriterien der EN 301 549 umgesetzt

In den folgenden Abschnitten schauen wir uns anhand von praktischen Anwendungsbeispielen an, wie bestimmte Prüfungsschritte umgesetzt werden können.

Screenreadertauglichkeit

Ein Screenreader ist eine Vorlesefunktion für blinde Menschen. Da PWAs plattformunabhängig sind, gibt **Tabelle 1** einen Überblick über Screenreader, die für unterschiedliche Betriebssysteme zur Verfügung stehen, und wie diese aktiviert werden können.

Screenreader	Betriebssystem	Menü
Sprachausgabe	Windows 11	Einstellungen | Barrierefrei-heit | Sprachausgabe [10]
NVDA	Windows 11	Downloadlink https://nvda.bhvd.de/
Talkback	Android 12	Einstellungen | Bedienungshilfen | Talkback [11]
Voice Over	IOS 15.5	Einstellungen | Bedienungshilfen | Voice Over
Voice Over	macOS 10.15	Systemeinstellungen | Bedienungshilfen | Voice Over [12]
Orca	Ubuntu 22.04	Einstellungen | Barrierefreiheit | Bildschirmleser [13]

Tabelle 1: Überblick über unterschiedliche Screenreader

Screenreadertauglichkeit bedeutet, dass eine App-Oberfläche so entwickelt ist, dass sie von Screenreadern vorgelesen werden kann. Um das responsive Webdesign besser verwirklichen zu können, können Div-Container als Schaltflächen verwendet werden. Warum dies wichtig ist, wird im Abschnitt „Motorische Einschränkungen“ erklärt. Damit der Screenreader weiß, dass der Div-Container eine Schaltfläche ist, muss das *role*-Attribut [14] den Wert *button* bekommen. Um einen Text festzulegen, der von Screenreadern vorgelesen wird, muss das Attribut *aria-label* verwendet und ihm ein Wert zugewiesen werden. Hier ein HTML-Beispiel:

```
<div role="button" aria-label="Quiz starten">START</div>
```

Mit den Attributen *role* und *aria-label* wird der Div-Container zur screenreadertauglichen, responsiven Schalfläche.

PWAs können nicht nur in mobilen Betriebssystemen, Android und

IOS, sondern auch auf Desktopbetriebssystemen (Windows, Ubuntu und macOS) ausgeführt werden. Diese Möglichkeit ist für Menschen interessant, bei denen aufgrund einer Behinderung eine motorische Einschränkung in den Händen vorhanden ist. Blinde Menschen können ebenso den Wunsch haben, eine PWA auf einem Computer oder Laptop bedienen zu können. Sobald eine PWA auf einem Computer ausgeführt wird, ist es wichtig, dass sie komplett ohne Maus, also nur per Tastatur bedienbar ist [15]. Deswegen sollten alle Bedienelemente per Tabulatortaste erreichbar sein. Dem Div-Container, der als Schaltfläche verwendet wird, wird deswegen das Attribut *tabindex* hinzugefügt. Hier ein HTML-Beispiel:

```
<div tabindex="0">START</div>
```

Nach dem Hinzufügen des Attributs sind die Div-Schalter per Tabulatortaste erreichbar und somit ist die Grundvoraussetzung des Tastaturbedienbarkeit erfüllt. Dazu gehört auch, dass wichtige Funktionen per Tastenkürzel ausgeführt werden können. Das hilft blinden und sehbehinderten Menschen. Tastenkürzel können mit JavaScript wie folgt realisiert werden:

```
document.addEventListener("keydown", (varevent) => {
  switch (varevent.key) {
    case "s":
      document.getElementById("btnStart").click();
      break;
    case "w":
      document.getElementById("btnWeiter").click();
      break;
  }
});
```

Der JavaScript-Code sorgt dafür, dass der Schalter Start durch Drücken des Buchstaben *s* und der Schalter Weiter mit dem Buchstaben *w* ausgeführt werden kann.

Sichtbarkeit des Tastaturfokus

Dieses Thema bekommt dann große Bedeutung, wenn eine PWA auf einem Computer oder Laptop ausgeführt wird. Menschen mit einer Sehbehinderung haben Probleme, zu erkennen, welches Bedienelement den Tastaturfokus hat. In Eingabefeldern wird der Textcursor oft als schmaler senkrechter Strich dargestellt. Das ist für Menschen mit einer Sehbehinderung sehr schwer zu erkennen. Microsoft hat im Betriebssystem Windows 11 hierfür eine Lösung. In Einstellungen | Barrierefreiheit | Textcursor kann bei Textcursor-Indikator [16] die Darstellung des Textcursors angepasst werden.

In PWAs kann der App-Entwickler dafür sorgen, dass aktive Bedienelemente die Farbe Gelb als Hintergrundfarbe zugewiesen bekommen. Wenn ein Bedienelement bei Aktivierung eine gelbe Hintergrundfarbe bekommt, ist das für Menschen mit Sehbehinderung sofort sichtbar. Das kann mit CSS sehr einfach gelöst werden:

```
#btnStart:focus{background-color: yellow; color: black;}
```

Barrierefreier Farbkontrast

Menschen mit einer Farbfehlsichtigkeit können nicht immer einer Farbe den richtigen Namen zuordnen. Ihnen fehlt das Gefühl, welche Farben zusammenpassen. Ein Text mit einer dunklen Schriftfarbe auf einer dunklen Hintergrundfarbe ist für sie ebenso wenig erkennbar wie Text mit einer hellen Schriftfarbe auf einer hellen Hintergrundfarbe. Für diese Menschen ist es wichtig, dass eine App-Oberfläche einen barrierefreien Farbkontrast zwischen Schriftfarbe und Hintergrundfarbe [17] hat. Die Überprüfung des Farbkontrasts auf Barrierefreiheit einer App-Oberfläche kann mit der kostenlosen Software Colour Contrast Analyser [18] vorgenommen werden.

Zur generellen Frage der Farbgestaltung von PWA-Oberflächen ist es auch möglich, sich Anregungen von Material Design [19] von Google zu holen.

Motorische Einschränkungen

Motorische Einschränkungen betreffen „kleine Bewegungen“ (Feinmotorik) und „große Bewegungen“ (Grobmotorik). Menschen mit motorischen Einschränkungen in den Händen können Probleme haben, zu kleine Schaltflächen anzutippen. Google führt deswegen in seinen Richtlinien zur barrierefreien Appentwicklung das Kriterium „Use large, simple controls“ auf [20]. Hier empfiehlt Google eine Mindestgröße von Schaltflächen von 48dpx48dp. Bei der Entwicklung von PWAs wird empfohlen, diese Mindestgröße umzusetzen, weil sie auch von dem Accessibility Scanner (so heißt das Überprüfungstool von Google) kontrolliert wird.

Menschen mit motorischen Einschränkungen können auf die Idee kommen, eine PWA auf einem Tablet, iPad oder Computer zu bedienen, mit der Hoffnung, dass dort die Schaltflächen größer dargestellt werden. Wie im Abschnitt Screenreadertauglichkeit erklärt, ist das der Grund, warum die Schaltflächen nicht als JavaScript-Buttons

```
<button onclick="myFunction()">Click me</button>
```

sondern als Divs

```
<div role="button" aria-label="Quiz starten" tabindex="0">START</div>
```

definiert werden.

Mit Media Queries (gehört zu Cascading Style Sheets) kann dafür gesorgt werden, dass bei einer Displaygröße von 768 x 1024 die Schaltflächen größer dargestellt werden. Folgender CSS-Code zeigt, wie es geht:

```
@media only screen and (min-width: 768px){
```

```
  .schalter{
    height: 6.0rem;
  }
}
```

Es wird die Displaygröße 1024 × 768 abgefragt, die bei Tablets und iPads oft anzutreffen ist. Die Abfrage funktioniert ebenfalls bei Bildschirmauflösungen von Computern und Laptops. Wenn die Displaygröße zutrifft, werden die Div-Container, welche eine CSS-Klasse *Schalter* besitzen, in der Höhe auf 6.0 rem angepasst. Somit werden die Schaltflächen größer und Menschen mit motorischen Einschränkungen in den Händen haben es leichter, eine Schaltfläche mit dem Finger oder der Computermaus anzutippen.

Übernahme von Einstellungen des Betriebssystems

Bestimmte Personengruppen mit körperlichen Einschränkungen nehmen grundsätzliche Anpassungen im Betriebssystem vor mit der Erwartung, dass die Apps diese Einstellungen übernehmen. Sehbehinderte Menschen passen im Betriebssystem die Schriftgröße an. Menschen mit einer Farbfehlsichtigkeit können im Betriebssystem den hohen Kontrast aktivieren.

PWAs sollen, wenn installiert, so plattformnah wie möglich aussehen und sich auch so verhalten. Deswegen ist dieses Thema etwas komplex. Für Menschen mit einer Sehbehinderung ist es wichtig, dass die App-Oberfläche vergrößert bzw. gezoomt werden kann. Wie dies auf unterschiedliche Betriebssysteme umgesetzt wird, zeigt Tabelle 2.

Betriebssystem	**App-Oberfläche zoomen**
Windows	Im App-Menü (3 senkrechte Punkte) gibt es einen Menüpunkt Zoomen
Ubuntu	Im App-Menü (3 senkrechte Punkte) gibt es einen Menüpunkt Zoomen

Betriebssystem	App-Oberfläche zoomen
macOS	Im App-Menü (3 senkrechte Punkte) gibt es einen Menüpunkt Zoomen
Android	Einstellungen | Bedienungshilfen | Text und Anzeige | Anzeigegrösse
IOS	Bedienungshilfen | Zoom

Tabelle 2: Vergrößern der App-Oberfläche auf unterschiedlichen Betriebssystemen

An diesem Beispiel wird deutlich, dass es keine einheitliche Regel gibt. Bei drei Betriebssystemen wird das Zoomen in der App-Oberfläche umgesetzt und bei zwei Betriebssystemen wird die Einstellung im Betriebssystem übernommen. Dieses Wissen ist wichtig, wenn App-Entwickler bestimmte Barrierefreiheitsfunktionen auf unterschiedlichen Betriebssystemen testen möchten.

Progressive Web Apps auf Barrierefreiheit validieren

Nach der App-Entwicklung mit Barrierefreiheit im Blick muss zum Schluss herausgefunden werden, ob alles tatsächlich wie gewünscht funktioniert. Es gibt zwei grundsätzliche Methoden, PWAs auf Barrierefreiheit zu testen: ein automatisierter Test oder ein Test von Hand aufgrund der Richtlinien EN 301 549.

Zunächst wird der automatisierte Test mit dem Accessibility Scanner [21] betrachtet. Dieser ist für Smart-phones und Tablets ab Android 6.0 geeignet [22]. Auf einem Tablet mit Android 12 ist er erfreulicherweise schon vorinstalliert. Mit dem Accessibility Scanner können native Apps und PWAs auf Barrierefreiheit überprüft werden. Aktiviert wird er in Einstellungen | Bedienungshilfen | Accessibility Scanner. Zunächst wird eine Erlaubnis benötigt, dass der Scanner über anderen Apps eingeblendet werden darf. In den Einstellungen können das Textkontrastverhältnis, das Bildkontrastverhältnis

und die Größe des Berührungsbereichs angepasst werden.

Es gibt zwei Möglichkeiten, mit dem Accessibility Scanner eine PWA auf Barrierefreiheit zu überprüfen: „Aufnehmen“ und „Snapshot“. „Aufnehmen“ macht jedes Mal ein Screenshot, wenn sich die App-Oberfläche ändert. Mit dieser Methode kann man sehr schnell eine komplette PWA auf Barrierefreiheit überprüfen. Bei „Snapshot“ kann der App-Entwickler festlegen, wann er die App-Oberfläche überprüfen möchte. Wenn der Accessibility Scanner Fehler findet, dann sind die Fehlermeldungen leicht verständlich formuliert, sodass der App-Entwickler weiß, was er zur Behebung tun muss.

Eine weitere Möglichkeit, die Barrierefreiheit einer PWA zu überprüfen, ist das kostenlose Tool Google Lighthouse [23]. Google Lighthouse ist im Browser Google Chrome in den Entwickler-Tools zu finden. Folgende Schritte müssen umgesetzt werden, um eine PWA mit Google Lighthouse auf Barrierefreiheit zu überprüfen: Die PWA in Google Chrome öffnen. Die Entwickler-Tools öffnen über das Menü Weitere Tools | Entwickler-Tools oder mit der Tastenkombination STRG + UMSCHALT + I. Jetzt das Menü „>>“ aktivieren und dann das Menü Lighthouse auswählen. Hier den Modus auf Navigation (Default) lassen. Bei Gerät die Option Mobil auswählen. Bei Categories die Option Bedienungshilfen auswählen. Jetzt mit Aktivieren des Schalters Analyze page load die Analyse auf Barrierefreiheit starten.

Wenn die Analyse beendet ist, wird ein Kreis angezeigt, in dem eine Zahl steht. Die Zahl 100 ist die beste Bewertung, die erreicht werden kann. Als Erstes werden Prüfungskriterien angezeigt, die nicht bestanden wurden. Weiter unten werden Elemente angezeigt, die manuell geprüft werden müssen. Noch weiter unten werden *Bestandene Prüfungen* und ganz unten *Nicht zutreffend* angezeigt. In der Kategorie *Zusätzliche Elemente zur manuellen Überprüfung* gibt es noch einen Link zur Seite „How To Do an Accessibility Review“ [24]. Hier gibt Google-Mitarbeiter Rob Dodson Tipps in Form eines YouTube-Videos und

Text, wie eine Überprüfung auf Barrierefreiheit durchgeführt werden kann.

Die dritte Möglichkeit ist, die PWA nach den 98 Prüfungsschritten der EN 301 549 von Hand zu prüfen. Zusätzlich ist es hilfreich, die App von Menschen mit Behinderungen testen zu lassen. Ob beispielsweise eine PWA für blinde Menschen bedienbar ist, weiß ein blinder Mensch am besten.

Fazit

Das Entwickeln von barrierefreien PWAs ermöglicht es, barrierefreie Apps zu entwickeln, die auf allen Betriebssystemen den gleichen Standard in Sachen Barrierefreiheit zu Verfügung stellen. Das bedeutet, dass es keine Personengruppe gibt, die von der Nutzung einer App auf einem bestimmten Betriebssystem ausgeschlossen wird.

Markus Lemcke ist seit elf Jahren selbstständig im Bereich Barrierefreiheit von Webseiten, Software und Betriebssystemen. Er ist Dozent an Hochschulen und schreibt als Autor für Fachmagazine. Sein Schwerpunkt ist die barrierefreie Softwareentwicklung mit Java und C#.

Links & Literatur

[1] https://www.gesetze-im-internet.de/bgg/__12a.html

[2] https://www.etsi.org/deliver/etsi_en/301500_301599/301549/03.0

2.01_60/en_301549v030201p.pdf

[3] https://developer.android.com/guide/topics/ui/accessibility/apps

[4] https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/iPhoneAccessibility/Making_Application_Accessible/Making_Application_Accessible.html

[5] https://docs.microsoft.com/de-de/windows/apps/design/accessibility/developing-inclusive-windows-apps

[6] https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Behinderte-Menschen/_inhalt.html

[7] https://www.destatis.de/DE/Themen/Querschnitt/Demografischer-Wandel/Aeltere-Menschen/bevoelkerung-ab-65-j.html

[8] https://www.bitvtest.de/bitv_test/das_testverfahren_im_detail/pruefschritte.html

[9] https://www.w3.org/TR/WCAG21/

[10] https://www.youtube.com/watch?v=vAwf4WFq1jg

[11] https://www.youtube.com/watch?v=ABdOTZUGZR0

[12] https://www.youtube.com/watch?v=6MjR498CyMM

[13] https://www.youtube.com/watch?v=HIKGTi8O9KQ

[14] https://webtest.bitv-test.de/index.php

[15] https://webtest.bitv-test.de/index.php?a=di&iid=267&s=n

[16] https://www.youtube.com/watch?v=cjl0gTkAVc8

[17] https://webtest.bitv-test.de/index.php?a=di&iid=260&s=n

[18] https://www.tpgi.com/color-contrast-checker/

[19] https://material.io/design/color/the-color-system.html

[20] https://developer.android.com/guide/topics/ui/accessibility/apps

[21] https://support.google.com/accessibility/android/answer/6376570?hl=de

[22] https://www.youtube.com/watch?v=GRV1kucMqIo

[23] https://developers.google.com/web/tools/lighthouse

[24] https://web.dev/how-to-review/