Entwicklung barrierefreier Software

Inhaltsverzeichnis

Entwicklung barrierefreier Software

Nachdem in den vorhergehenden Abschnitten die besonderen Bedürfnisse von Menschen bzw. die Anforderungen, die an barrierefreie Soft- und Hardware gestellt werden, beschrieben wurden, sollen die nächsten Kapitel einen kleinen Einblick in die tatsächliche Umsetzung dieser Forderungen bieten.

Grundlegende Programmierregeln für barrierefreie Software

Obwohl die Palette an unterschiedlichen Programmiersprachen und Entwicklungsumgebungen sehr breit gefächert ist und jede einzelne Sprache ihre Eigenheiten besitzt, gibt es dennoch ein paar grundlegende Richtlinien die immer beachtet werden sollten. In der Folge sollen ein paar dieser Grundprinzipien näher erläutert werden.

Standardkonformität

Neben dem großen Fehler zu glauben, es reicht Hard- und vor allem Software für den „Standarduser" zu entwickeln, scheint auch die Annahme, dass Barrierefreiheit etwas sehr kostspieliges ist, weit verbreitet zu sein. Obwohl diese Vermutung für manche Entwicklungen zwar durchaus eintreten kann, reicht für den Abbau vieler Hürden zumeist bereits ein Entwicklungskonzept, welches sich an Normen und offizielen Standards orientiert.
Ein gutes Beispiel hierfür sind die unterschiedlichen Bedürfnisse blinder und mobiler User bezüglich der Nutzung von Webseiten. Während die von blinden Menschen genutzten Screenreader eine semantisch korrekte HTML-Seite benötigen, ist es für mobile User besonders wichtig, dass Inhalt und Layout getrennt behandelt werden. Die Bedürfnisse beider Nutzergruppen können schlicht und einfach durch Standardkonformität gelöst werden, ohne eine individuelle Lösung erarbeiten zu müssen oder eine Kostenexplosion zu verursachen.[1]

Alternativen für Multimediainhalte

Multimediainhalte, wie Video und Audio, sind für bestimmte Nutzergruppen nicht zugänglich. Daher ist es wichtig diesen Nutzern die Inhalte und Informationen, welche nicht erfasst werden können, auf andere Art und Weise näherzubringen. Dies bedeutet, dass eine zusätzliche und alternative Variante der Informationsvermittlung zur Verfügung gestellt werden muss. Im konkreten Fall kann dies bedeuten, dass akustische Informationen, wie beispielsweise Warntöne, durch zusätzliche visuelle Hinweise signalisiert werden. Dasselbe gilt auch für

[1] Vgl. ebd., Seite 32.

Videos. Wenn die Inhalte auf akustischem Weg nicht wahrgenommen werden können, müssen die Inhalte zusätzlich in Bild- oder Textform, zum Beispiel durch Untertitel, beschrieben werden. Eine weitere nützliche Funktion, welche vielen Usern den Umgang mit der Software erleichtert, ist die Möglichkeit gewisse Einstellungen, wie etwa die Lautstärke, selbst regulieren zu dürfen.[2]

Steuerung mittels Tastatur

Genau wie bei den „normalen" Usern ist es auch einem Großteil der Nutzer mit speziellen Handicaps möglich die Tastatur als Standard-Eingabemedium zu benutzen. Daher lässt sich durch eine zusätzliche bzw. alternative Steuerungsmöglichkeit mittels Tastatur eine große Zielgruppe ansprechen, welche davon profitieren würde.

Um eine alternative Steuerung mittels Tastatur zu ermöglichen, ist es zunächst nötig für jede mögliche Interaktionen zwischen Mensch und Maschine eine bestimmte Taste bzw. Tastenkombination zu belegen. Hierzu müssen unter anderem Standardtasten definiert werden um beispielsweise den Tastaturfokus auf einen gewissen Bereich der Software zu lenken oder um Sub-Menüs öffnen zu können. Besonders hilfreich sind hier „mnemonics" (Merkhilfen), welche ein rasches Zugreifen auf bestimmte Ressourcen erlauben. Mnemonics werden zumeist durch das Unterstreichen des ersten Buchstabens gekennzeichnet (z.B.: Datei) und können durch eine bestimmte Tastenkombination aktiviert werden (z.B.: ALT + D).

Eine besondere Schwierigkeit stellen Mausgesten und spezielle durch Maussteuerung ausgeführte Funktionen, wie zum Beispiel „Drag&Drop" dar. Ein optisches Drag&Drop wie man es von der Maussteuerung kennt, ist durch Steuerung mittels Tastatur nicht sinnvoll reproduzierbar, dennoch ist es ohne größere Probleme möglich die Funktionsweise von Drag&Drop mittels Tastatureingabe zu reproduzieren.

Auch bei der Auswahl der Tasten, welche eine barrierefreie Software navigierbar machen sollen, ist Vorsicht geboten. Zunächst muss darauf geachtet werden, dass Tasten, die bereits für spezielle Accessibilityfeatures des Betriebssystems belegt sind, nicht nochmals belegt werden, da dies sonst zu Komplikationen führen würde. Zusätzlich sollte man sich auch an systemweite Konventionen bei der Belegung von Tasten und Kombinationen halten um Verwirrung beim Umgang mit der Software zu verhindern. Es wäre daher beispielsweise falsch die „ESC-Taste" mit einer anderen Funktion, als jener für die sie bekannt ist, zu belegen.

[2] Vgl. Schwerdtfeger, Richard S. (2006): IBM Guidelines for Writing Accessible Applications Using 100% Pure Java: Online im Internet: URL: http://www-03.ibm.com/able/guidelines/java/snsjavag.html, [22.10. 2011].

Abschließend ist noch zu erwähnen, dass eine digitale Dokumentation der Navigation mittels Tastatur vorhanden sein sollte, da vielen Nutzern die Informationen eines Handbuches nicht zugänglich sind.[3]

Programmfokus

Der Programmfokus wurde zwar bereits im Abschnitt „Steuerung mittels Tastatur" angesprochen, soll an dieser Stelle jedoch etwas genauer beschrieben weden. Der Programmfokus beschreibt jenen Bereich einer Software, welcher den nächsten Schritt des aktuellen Anwendungsprozesses darstellt. Daher soll auch der Fokus des Nutzers auf diesen Abschnitt gelenkt werden. Der Anwender muss also durch die einzelnen Schritte eines Prozesses geführt werden, damit dieser zu jedem Zeitpunkt darüber in Kenntnis ist, was er an der aktuellen Prozessstelle zu tun hat. Der Programmfokus muss daher gut definiert und deutlich hervorgehoben sein, um dem Anwender den Prozessfluss leichter zugänglich zu machen.
Bei der Planung des Programmfokus ist zusätzlich darauf zu achten, dass etwaige Wechsel des Fokus von assistiven Technologien erfasst werden können. Besonders kritisch ist diese Planung bei Dialogen und Warnhinweisen, da diese zumeist nach einer Interaktion des Anwenders verlangen und unter Umständen auch unerwartet auftreten können. Wird beim Entwurf der Anwendungsprozesse auf diese Umstände keine Rücksicht genommen, wird die Anwendung für Nutzer von assistiven Technologien zwangsläufig unbrauchbar.[4]

Gestaltung von Inhalten

Die Art und Weise in welcher die Informationen an die Endnutzer transportiert werden, ist ein wesentlicher Faktor um eine Anwendung barrierefrei zu machen. Wie der bisherige Inhalt dieser Arbeit gezeigt hat, ist bei der Präsentation von Informationen auf viele unterschiedliche Details zu achten. In der Folge sollen einige wichtige Faktoren genauer beschrieben werden.

Layout

Ein wichtiger Faktor um eine sinnvolle Steuerung des Programmfokus zu ermöglichen, ist das Programmlayout. Wie bereits in einem vorhergehenden Punkt beschrieben ist die Navigation mittels Tastatur oder auch mittels Sprache für viele Nutzer unumgänglich. Die Navigation mittels Tastatur oder Sprache funktioniert sehr linear, es wird schlichtweg von einem Abschnitt zum nächsten gesprungen, das sogenannte „tabbing". Daher müssen die unterschiedlichen

[3] Vgl. ebd.
[4] Vgl. ebd.

Komponenten einer Software in jener Reihenfolge zugänglich gemacht werden in welcher sie verwendet werden sollen. Hierbei ist es wichtig, dass die Sprünge zwischen den Elementen inhaltlich nachvollziehbar und logisch sind.

Beschreibung von Inhalten, Gruppen und Komponenten

Jede Software besteht aus unterschiedlichen Komponenten, welche in einer gewissen Form verwendet werden sollen und dadurch auch spezifische Anforderungen an den Nutzer stellen. Durch unterschiedliche Handicaps ist es dem Nutzer allerdings eventuell nicht möglich die Anforderungen einer Komponente, wie beispielsweise eines Textfeldes, problemlos erfassen zu können. Daher ist es nötig diese Elemente eindeutig zu benennen und eine alternative Beschreibung der Inhalte bzw. Anforderungen, beispielsweise durch Tooltips oder „alt-Texte“, zur Verfügung zu stellen. Das Ziel eines Programmes solltes es sein, dem Nutzer zu jeder Zeit Information und Hilfestellung bezüglich der an ihn gestellten Anforderungen zu geben.

Im Zusammenhang mit der Erfassung von Anforderungen ist es auch notwendig logisch zusammengehörige Gruppen als solche zu definieren. Ein simples Beispiel hierfür stellt eine Reihe von zusammengehörigen Check-Boxen dar, welche durch einen Gruppennamen als logisch zusammengehörig deklariert werden muss. Auf diese Art und Weise können verschiedene Optionen oder Funktionen als zusammengehörige Gruppen, wie zum Beispiel „Suchoptionen“ oder „Audio-Einstellungen“, erkannt werden und erleichtern dem Anwender somit die Navigation.

Allgmein ist festzuhalten, dass jede einzelne Schaltfläche und Grafik einer Anwendung über eine alternative Beschreibung verfügen sollte. Dies ist nicht nur für Menschen mit Handicaps vorteilhaft, sondern auch für unerfahrene Anwender, die mit der Programmlogik und den Funktionen nicht vertraut sind.

Schriftart, Farbgestaltung und Größe

Schriftliche Inhalte sind ein wichtiger Bestandteil jeder Software und verlangen daher nach besonderer Aufmerksamkeit. Viele essentielle Informationen werden über schriftliche Inhalte vermittelt, wie beispielsweise bei Menü- und Schaltflächen oder digitalen Bedienungsanleitungen. Um hier keine unnötigen Hürden für die Endnutzer zu erzeugen, sollte man sich an ein paar Grundregeln halten.

Zunächst sollte es immer möglich sein die System-Einstellungen bezüglich Schriftgröße, Farbe und Darstellung auch für die Software zu übernehmen. Moderne Betriebssysteme bieten den Anwendern bereits vielfältige Möglichkeiten um das Erscheinungsbild schriftlicher Informationen an die jeweiligen Bedürfnisse anzupassen. Es ist daher nicht benutzerfreundlich

die Vorlieben der Anwender zu ignorieren und ausschließlich eigene Einstellungsmöglichkeiten zur Individualisierung anzubieten.

Wenn man sich dazu entscheidet auch eigene Einstellungen zum Verändern des Erscheinungsbildes in die Software einzubinden sollte man dem Anwender immer mehrere Auswahlmöglichkeiten bieten, da unterschiedliche User unterschiedliche Kontraststärken oder Farbmischungen benötigen. Zusätzlich ist darauf zu achten, dass diese Einstellungen für alle Bereiche einer Software übernommen werden und nicht nur für ein Menü oder einen Teil des Bildschirms.

Eine andere häufige Fehlerquelle beim Erstellen von Software ist die farbliche Gestaltung von Inhalten. Obwohl der richtige Einsatz von Farbe und Kontrasten den Informationsgehalt und die Übersichtlichkeit deutlich verbessern, können bei schlechter Gestaltung große Probleme für die Anwender entstehen. Grundsätzlich sollten Farbe bzw. Kontrast nur als zusätzliches Mittel zur Informationsvermittlung verwendet werden und die Information nicht allein transportieren. Beispielsweise ist es nicht sinnvoll den Anwender aufzufordern eine rote Schaltfläche anzuklicken, da Menschen, welche unter Farbenblindheit leiden, diese Information nicht problemlos verarbeiten können. Die Information muss also zusätzlich noch über einen anderen Weg transportiert werden.[5]

Animierte Bildschirminhalte

Animierte Bildschirminhalte sind ein wesentlicher Bestandteil vieler Programme, für welche grundsätzlich dieselben Richtlinien, die bereits im vorhergehenden Punkt beschrieben wurden, gelten.

Obwohl Animationen das Erscheinungsbild eines Screens aufpolieren können, sollten sie nur dann eingesetzt werden, wenn sie zur Informationsvermittlung notwendig sind bzw. diese signifikant unterstützen. Ist dies nicht der Fall, dann ist es vermutlich sinnvoller auf andere Gestaltungsmethoden zurückzugreifen. Diese Vorgangsweise ist vor allem deswegen nötig, da animierte Informationen von Hilfsmitteln wie Screenreadern oder Braillezeilen nicht problemlos erfasst werden können und daher ist es besonders wichtig diese Informationen auch auf andere Art und Weise verfügbar zu machen.

Blinkende Texte bzw. Objekte sollten generell vermieden werden. Diese irritieren einerseits den Fokus des Nutzers und erzeugen andererseits auf Dauer auch ein unangenehmes Erlebnis beim Betrachter, was schlussendlich bei Überreizung auch zu epileptischen Anfällen führen kann.[6]

[5] Vgl. ebd.
[6] Vgl. ebd.

Barrierefreie PDFs

PDF (Portable Document Format) hat sich in vielen unterschiedlichen Unternehmen, Bildungseinrichtungen und Behörden als Standardmedium zur Weitergabe von Daten entwickelt. Dies liegt vorrangig daran, dass PDF-Files auf jedem gängigen System in unverfälschter und layoutgetreuer Form wiedergegeben werden können. Zudem ist es durch die Einbindung von digitalen Signaturen auch möglich rechtlich verbindliche Unterschriften in digitaler Form zu übermitteln. Dies bedeutet in weiterer Folge, dass viele Verpflichtungen, wie beispielsweise Behördenwege, von zuhause aus erledigt werden können, was für Menschen mit körperlichen Einschränkungen von Vorteil sein sollte.

Was hier in der Theorie gut klingt, scheitert leider an der Realität, in welcher viele PDF-Dokumente schlichtweg nicht barrierefrei gestaltet werden. Obwohl die Grundvoraussetzungen für Barrierefreiheit bereits durch die Lesesoftware „Acrobat Reader“, welcher viele Funktionen für Menschen mit Handicap bietet, gegeben wären, scheitert die letztendliche Umsetzung an der Unzugänglichkeit der Dokumente selbst.

Um ein PDF-Dokument zugänglich zu machen, ist es wichtig, dass die Dokumentenstruktur erkennbar ist und die Text- von den Grafikelementen unterschieden werden können. Ein gutes Beispiel hierfür sind eingebettete Textgrafiken oder eingescannte Dokumente. Diese sind für sehende Menschen ohne größere Probleme zu verstehen, für einen Screenreader aber unmöglich zu verarbeiten. Von mindestens ebenso hoher Bedeutung sind die Strukturinformationen des Dokuments, welche im Endeffekt als Navigationshilfen dienen. Dies ist umso wichtiger je größer das Dokument ist. Hierzu lässt sich grundsätzlich festhalten, dass ein Dokument welches strukturiert, unter der Verwendung von Formatvorlagen, erstellt worden ist, bereits einen wesentlichen Schritt in Richtung Zugänglichkeit gemacht hat.[7]

Der Schlüssel um PDFs nun endgültig barrierefrei zu machen, nennt sich „tagged PDF“. Tagged PDF ist seit der Version Adobe Reader 5 in das Programm integriert und hat sich seit dem stetig weiterentwickelt. Die Funktionalität und Schreibweise von tagged PDF ähnelt jener von (X)HTML, welche ebenfalls auf die Kennzeichnung unterschiedlicher Elemente abzielt. Damit ist es möglich die Strukturierung des Dokuments zu beschreiben, Alternativtexte für Bilder beizufügen und die Navigationmöglichkeit mittels Lesezeichen bereitzustellen.[8]

---

[7] Vgl. Heuwinkel, Roland (2003): PDF-Dokumente – lesbar für alle, Online im Internet: URL: http://www.einfach-fuer-alle.de/artikel/pdf_barrierefrei/download/pdf_barrierefrei.pdf, [09.11.2011].
[8] Vgl. Hellbusch, Jan Eric (2005): Gestaltung barrierefreier PDF-Dokument, Online im Internet: URL: http://www.einfach-fuer-alle.de/artikel/pdf-barrierefrei-umsetzen/Gestaltung-barrierefreier-PDF.pdf, [09.11.2011].

Die Initiative „Einfach für Alle“[9] von Der „Aktion Mensch“ hat eine Checkliste zur Erstellung von barrierefreien PDFs zusammengestellt, welche unter folgendem Link zu finden ist:

> http://www.einfach-fuer-alle.de/artikel/checkliste-barrierefreie-pdf/Checkliste-BarrierefreiesPDF.pdf

Im Zuge dieses Dokuments werden in mehreren Schritten die wichtigsten Informationen bezüglich tagged PDF vermittelt und zusätzlich auch detailierte Informationen zur Umsetzung mittels Acrobat Reader gegeben.

Barrierefreiheit mit Java

Bei Java handelt es sich um eine objektorientierte Programmiersprache. Sie wurde 1995 von Sun Microsystems veröffentlicht und ist Basis bzw. grundlegender Bestandteil von vielen Millionen Anwendungen, aber auch unterschiedlichsten Geräten wie beispielsweise Fernsehern. Zusätzlich ist Java in Form von Applets auch auf vielen Internetseiten vertreten.
Um die Zugänglichkeit von Java-Programmen zu gewährleisten, wurden spezielle Programmierschnittstellen geschaffen. Diese Accessibility-Klassen ermöglichen dem User die Software neben den herkömmlichen auch mit alternativen Eingabemedien zu bedienen.[10]

Die Barrierefreiheit von Java wird durch folgende Elemente realisiert:

- ***„Java Accessibility API (JAAPI):*** *Ermöglicht die Kommunikation zwischen Java-Anwendungen und Unterstützungstechnologien (wie Screenreader oder Braille-Zeile).*
- ***Java Accessibility-Dienstprogramme:*** *Damit können Informationen aus einer Anwendung erfasst und für die Anzeige mit speziellen Ausgabegeräten weiterverarbeitet werden. Unterstützungstechnologien können Ereignisse überwachen und zusätzliche Informationen über die graphische Nutzeroberfläche erhalten (geöffnete Fenster, Mausposition etc.).*

[9] http://www.einfach-fuer-alle.de/
[10] Vgl. Lemay, Laura/ Cadenhead, Rogers: Java 2 in 21 Tagen, Markt + Technik Verlag, München, 2003, Seite 698f.

- ***Java Access Bridge (JAB):*** *Ermöglicht die Kommunikation zwischen der in Microsoft Windows integrierten Unterstützungstechnologien mit der JAAPI.*
- ***Java Foundation Classes (JFC):*** *Dies ist eine Bibliothek von Komponenten der graphischen Benutzeroberfläche, in die JAAPI vollständig implementiert ist.“*[11]

Um eine Java-Anwendung nun barrierefrei zu gestalten, ist vor allem die Schnittstelle „javax.accessibility.Accessible“ von Bedeutung. Diese wird von allen Komponenten implementiert, die alternativ zugänglich sein sollen, wie etwa „JButton“ oder „JLabel“. Über diese neue Schnittstelle wird die Methode „getAccessibleContext()“ verfügbar, welche ein „AccessibleContext“- Objekt erzeugt. Dieses Objekt verwaltet Informationen bezüglich der Barrierefreiheit der Komponente. Hierzu zählen beispielsweise Namen oder Beschreibungen die von assistiven Technologien verarbeitet werden können.[12]

Hier ein Beispiel für eine barrierefreie Schaltfläche in Java:

```
„importjava.awt.*;
import javax.swing.JButton;
import javax.swing.JPanel;
import javax.swing.JFrame;
import javax.accessibility.*;
public class AccessSimpleButton extends JPanel {
public AccessSimpleButton() {
JButtonaButton = new JButton("Button");
aButton.getAccessibleContext().
setAccessibleName("Button");
String desc = "Dies ist ein einfacher Knopf";
aButton.getAccessibleContext().
setAccessibleDescription(desc);
add(aButton);
}
public static void main(String[] args) {
JFrame frame = new JFrame(
```

[11] Aktionsbündnis für barrierefreie Informationstechnik (AbI): Java und Barrierefreiheit, Online im Internet: URL: http://www.wob11.de/java-und-barrierefreiheit.html, [21.12.2011].
[12] Vgl. Ullenboom, Christian (2011): Java ist auch eine Insel. Das umfassende Handbuch, Online im Internet: URL: http://www.eshca.net/java/books/javainsel9/javainsel_19_027.htm#mje3e5b4b4ce8439d1223c2d7d608e6c17, [21.11.2011].

```
"AccessSimpleButton");
frame.getContentPane().
add(new AccessSimpleButton(),
BorderLayout.CENTER);
frame.pack();
frame.setVisible(true);
}
}"[13]
```

Um „nicht-barrierefreien" Javacode zu identifizieren, bietet sich der Java Accessibilty Helper an. Dieser durchsucht den Programmcode nach möglichen Problemen und bietet verschiedene Test-Funktionen an.

Barrierefreiheit mit HTML, CSS und Javascript

Im Gegensatz zu Java bieten HTML, CSS und Javascript keine Schnittstelle, welche eine Anwendung um Funktionen zur Barrierefreiheit erweitert. Alle Möglichkeiten zur barrierefreien Programmierung sind bereits vorhanden und müssen nur in der richtigen Form ausgeführt werden.
Hierbei ist es ratsam sich an die Prinzipien des WCAG zu halten: Wahrnehmbarkeit, Bedienbarkeit, Verständlichkeit und Robustheit. Genau diesen Weg wählen auch die unterschiedlichen Gesetze und Regelungen bezüglich Barrierefreiheit. Ein gutes Beispiel hierfür ist die am 12. 09. 2011 veröffentlichte BITV 2.0, anhand welcher in der Folge die unterschiedlichen Ansprüche an barrierefreie Webseiten beschrieben werden.

Die BITV, definiert verschiedene Bedingungen welche je nach betroffener Funktionalität einer von zwei Prioritätskategorien zugeordnet werden. Die erste Kategorie beschreibt Bedingungen, die auf jeden Fall umgesetzt werden müssen, während die zweite Anforderungen auflistet, welche zumindest von zentralen Navigations- und Einstiegsangeboten eingehalten werden sollten.[14]

[13] Mikhalenko, Peter V. (2007): JAAPI: Barrierefreiheit für Java-Anwendungen, Online im Internet: URL: http://www.zdnet.de/magazin/39152431/p-2/jaapi-barrierefreiheit-fuer-java-anwendungen.htm, [21.12.2011].
[14] Vgl. Bundesministerium der Justiz/ Juris GmbH (2011): Verordnung zur Schaffung barrierefreier Informationstechnik nach dem Behindertengleichstellungsgesetz (Barrierefreie-Informationstechnik-Verordnung – BITV 2.0), Online im Internet: URL: http://www.gesetze-im-internet.de/bundesrecht/bitv_2_0/gesamt.pdf, [31.12.2011].

Im Zuge dieser Arbeit werde ich ausschließlich auf Inhalte der ersten Prioritätskategorie eingehen, da die darin beschriebenen Anforderungen für den Einsatz im Schulbereich meiner Meinung nach ausreichend sind und die zusätzlichen Informationen im weiteren Verlauf der Arbeit daher keine Anwendung finden würden.
Für die anschließend genannten Bedingungen ist zudem noch anzumerken, dass für fast alle Punkte Ausnahmen bezüglich besonderer Anwendungszwecke oder Situationen beschrieben werden, in denen dieser Punkt nicht erfüllt werden muss. Da eine Auflistung jeder einzelnen Ausnahmesituation die Beschreibung der Anforderungen unnötig verkomplizieren würde, werde ich dies nur in wirklich wichtigen Fällen tun.

Die vollständigen Inhalte der BITV 2.0 sind unter folgendem Link abrufbar:
http://www.gesetze-im-internet.de/bundesrecht/bitv_2_0/gesamt.pdf

Anforderungen nach BITV 2.0

Allgemein

Die Internetseite ist für assistive Technologien zugänglich und alle Inhalte können von ihnen verarbeitet werden. Hierzu muss die Seite nach allgemein akzeptierten Standards und Richtlinien der verwendeten Programmier- bzw. Skriptsprache gestaltet sein. Von assistiven Technologien benötigte Metadaten sowie Zusatzinformationen zu Seitenelementen, wie etwa „alt-Texte“, müssen zur Verfügung gestellt werden.

Formatierung von Inhalten

Die Text-Inhalte der Seite sind in einfacher, verständlicher Sprache zu formulieren und die verwendete Sprache der jeweiligen Textpassagen ist durch assistive Technologien abfragbar.
Bei der Formatierung von Farbinhalten ist darauf zu achten, dass der Kontrast zwischen Vorder- und Hintergrund ausreichend ist um alle Inhalte problemlos wahrnehmen zu können. Zusätzlich darf die Farbgestaltung eines Elements nicht als einziges Mittel zur Informationsvermittlung genutzt werden, da dies bestimmten Usergruppen die Nutzung der Seiteninhalte erschwert.
Audio-Inhalte, die eine Abspielzeit von mehr als 3 Sekunden aufweisen, müssen bestimmte Navigationsmittel zur Verfügung stellen. Beispielsweise muss die Lautsärke der Wiedergabe vom Anwender reguliert werden können und er muss auch die Möglichkeit besitzen sie komplett zu stoppen.

Schriftliche Inhalte einer Seite müssen skalierbar gestaltet sein. Es sollte eine Vergrößerung von 200% möglich sein, ohne damit die Bedienbarkeit der Website einzuschränken. Schriftgrafiken sollten vollkommen vermieden werden, außer sie können vom Nutzer an dessen individuelle Bedürfnisse angepasst werden.

Nicht-Text-Inhalte

Für alle Inhalte einer Website, die nicht aus Text bestehen, also beispielsweise Bilder, Grafiken, Buttons oder Textfelder, müssen Textequivalente bereitgestellt werden.

Für die Erstellung von Textäquivalenten ist anzumerken, dass diese nur für Elemente erstellt werden müssen, die Information für den Nutzer transportieren. Es ist daher beispielsweise nicht nötig Alternativtexte für rein dekorative Bildschirminhalte oder Elemente, welche bereits durch ihren Namen eindeutig beschrieben werden, anzulegen.

Zeitgesteuerte Medien

Dem Nutzer muss immer die Möglichkeit geboten werden die Inhalte von zeitgesteuerten Medien, zusätzlich zur bereits gebotenen, auch in mindestens einer alternativen Form aufnehmen zu können. Bei Audio-Dateien könnte dies eine Text-Alternative sein und im Fall von Video-Dateien auch eine Tonspur. Wichtig ist hierbei immer die Gleichwertigkeit der Inhalte zu berücksichtigen. Bei genauerer Analyse dieser Bedingung ergeben sich nun einige unterschiedliche Möglichkeiten um diese Inhalte barrierefrei zu gestalten.

Eine Lösungsmöglichkeit, welche sowohl bei Audio- als auch bei Video-Inhalten Anwendung finden kann, ist eine alternative Volltext-Version der Informationen. Hier muss allerdings darauf geachtet werden, dass nicht nur der sprachlich kommunizierte Inhalt, sondern auch anderweitige Interaktion, die während der Spielzeit getätigt wurde, beschrieben wird. Eine ähnliche Lösungsvariante sind Untertitel, welche im Prinzip denselben Zweck erfüllen, aber zusätzlich noch an die zeitliche Komponente des Hauptmediums gekoppelt sind. Für Video-Inhalte gibt es auch noch die Möglichkeit eine Audio-Deskription bereitzustellen.

Zeitbezogene Anforderungen

Falls die Website zeitbezogene Anforderungen aufweist, gelten für diese besondere Regelungen. Sie müssen mindestens eine der folgenden Anforderungen erfüllen: die Begrenzung muss deaktivierbar sein, sie kann um das mindestens Zehnfache verlängert werden oder die Nutzer werden diesbezüglich vorgewarnt und erhalten anschließend 20 Sekunden Zeit um per Tastendruck zu reagieren um diese Frist zu verlängern (muss mindestens 10 mal möglich sein).

Animierte Seiteninhalte

Blinkende oder sich bewegende Seiteninhalte, die automatisch einsetzen und länger als fünf Sekunden andauern, müssen den Nutzern die Möglichkeit geben sie anzuhalten, zu beenden oder auszublenden. Häufig aufblitzende Elemente sind generell zu vermeiden. Als Richtwerte gelten hier spezielle „general flash“ und „red flash“-Schwellen.

Layout und Präsentation

Inhalte, die wesentliche Informationen durch die Form in der sie präsentiert werden vermitteln, müssen diese entweder für assistive Technologien zugänglich machen oder zusätzlich in Textform zur Verfügung stellen. Dies ist beispielsweise der Fall wenn Beziehungen zwischen Elementen nur durch optische Gestaltung markiert werden, die Reihenfolge der Elemente Auswirkungen auf deren Inhalt hat oder Unterschiede zwischen Inhalten durch spezielle Gestaltungsmerkmale, wie Größe, Farbe oder Platzierung, definiert werden.

Bedienbarkeit der Seite

Um die Bedienbarkeit zu gewährleisten, ist es nötig der Seite einen sinnvollen Titel zu geben, der bereits erste Informationen zum Inhalt gibt. Zusätzlich sollen alle Inhalte durch Überschriften und Labels gekennzeichnet bzw. beschrieben werden, damit dem Anwender zu jedem Zeitpunkt klar ist, wo genau er sich gerade befindet und wie er weiternavigieren kann. Hierzu ist es auch wichtig die Funktion bzw. das Ziel von Links deutlich zu deklarieren. Dies kann entweder durch die Namensgebung oder durch zusätzliche Kontext-Informationen gemacht werden. Weiters muss das aktuell fokussierte Element immer deutlich erkennbar und die durch Fokuswechsel durchgeschalteten Elemente durchdacht und sinnvoll gereiht sein. Die Bedienung der Inhalte soll somit vorhersehbar und vor allem nachvollziehbar gemacht werden. Dies wird auch durch die Verwendung von einheitlichen Navigationsmechanismen bzw. einheitlichen Namen für gleiche Elemente unterstützt.

Tastatur

Die gesamte Internetseite, inklusive all ihrer Funktionen, muss auch mittels Tastatur steuerbar sein. Dem Nutzer muss es also möglich sein die Seiteninhalte durch Tastenanschläge navigieren und bestimmte Elemente direkt ansteuern zu können. Jedes Element, das auf diese Weise ansteuerbar ist, muss auch wieder verlassen werden können. Falls hierfür andere als die Standard-, Pfeil- oder Tab-Tasten benötigt werden, muss dem Anwender dies in einer entsprechenden Form kommuniziert werden.

Fehlervermeidung bzw. -identifizierung

Dem Anwender müssen Funktionen zur Vermeidung von Eingabefehlern bereitgestellt werden. Werden Fehler automatisch identifiziert, müssen diese dem Nutzer deutlich aufgezeigt und ihm Informationen diesbezüglich, wie beispielsweise Korrekturvorschläge, in Textform bereitgestellt werden.

Wenn von einer Seite besonders heikle Informationen, wie etwa rechtliche oder finanzielle Belange, behandelt werden, muss dem Nutzer eine der folgenden Kontrollmöglichkeiten geboten werden: Die durchgeführte Aktion kann rückgängig gemacht werden, die Fehlerfreiheit der Aktion wird überprüft und Fehler können anschließend korrigiert werden oder die Informationen werden dem Anwender, bevor er sie endgültig bestätigt, nochmals gesammelt vorgelegt.

Validierungstools

Da das Internet mittlerweile zu einem zentralen und vielgenutzten Instrument unserer Gesellschaft geworden ist, scheint es gerade hier wichtig zu sein für Barrierefreiheit zu sorgen. Um Entwickler hierbei zu unterstützen gibt es inzwischen eine Vielzahl an Validierungstools, welche Programmcode auf Barrierefreiheit testen können.

Ein gutes Beispiel hierfür ist „AChecker“[15]. Bei AChecker handelt es sich um eine Webseite, welcher man HTML-Dateien oder gleich direkte Links zu einer Homepage übergeben kann. Die übergebene Quelldatei wird anschließend hinsichtlich Barrierefreiheit untersucht und mögliche Fehlerquellen detailreich aufgelistet. Zusätzlich werden der Fehleranalyse auch gleich Lösungsvorschläge beigefügt.

Ein anderes Online-Tool, welches sich im Prinzip genauso verhält wie AChecker ist „Hera“[16]. Hera bietet aber zusätzlich noch das nützliche Feature, dass es bereits gesichtete Seiten für sieben Tage in einer Datenbank speichert, sodass der Anwender durch einen speziellen Link in dieser Zeit nochmals auf die Auswertung zurückgreifen kann.

Ein drittes Werkzeug um Internetseiten auf ihre Barrierefreiheit zu überprüfen ist „Wave“[17]. Wave hebt sich von den anderen beiden Online-Tools dadurch ab, dass die Auswertung der Fehler direkt an der Internetseite passiert und nicht in Tabellenform ausgegeben wird. Dies birgt allerdings auch den Nachteil, dass die Auswertung durch die Vielzahl an Warnhinweisen schwer und vor allem unübersichtlich werden kann.

[15] http://achecker.ca/checker/index.php
[16] http://www.sidar.org/hera/
[17] http://wave.webaim.org/

Synchronized Multimedia Integration Language (SMIL)

Bei Synchronized Multimedia Integration Language (SMIL) handelt es sich um eine auf XML basierende Auszeichnungssprache. Sie wird vom W3C entwickelt und befindet sich aktuell in der Version 3.0, welche am 1. Dezember 2008 als neuer Standard veröffentlicht wurde. SMIL ermöglicht es Multimediainhalte, also beispielsweise Bilder, Audiodateien, Videos oder Hyperlinks, miteinander zu verknüpfen um somit neue interaktive Inhalte zu generieren.[18]

Die Inhalte lassen sich hierbei nicht einfach nur miteinander verknüpfen, sondern können mittels SMIL auch manipuliert oder erweitert werden. Hier liegt wohl das größte Potential dieser Auszeichnungssprache, zumindest im Bezug auf Barrierefreiheit. SMIL ermöglicht es nämlich Informationen auf verschiedenen Kommunikationsebenen zeitsynchron zu übertragen.[19] Dies bedeutet, dass es beispielsweise möglich ist ein Video abzuspielen, für welches eine alternative Tonspur mit mehr Informationen und zusätzlich auch noch zuschaltbare Untertitel bereitgestellt werden. Zusätzlich bietet SMIL auch noch viele andere Möglichkeiten um Barrieren abzubauen und Multimediainhalte zugänglicher zu machen. Einige andere potentielle Anwendungsgebiete wären eine Sprachauswahl für die Audiospur, barrierefreie Picture- bzw. Video-Maps, oder das automatische Anpassen der Inhalte an die Systemsettings des jeweiligen Users.[20]

Um Applikationen mit SMIL entwickeln zu können, bedarf es keiner besonderen Voraussetzungen. Wie auch bei HTML lassen sich die Anwendungen schon mit einem simplen Texteditor erstellen. Um die Multimediainhalte anschließend abspielen zu können, wird allerdings ein kompatibler Player benötigt. Ein Beispiel hierfür ist der „Ambulant Player“ [21] welcher zu SMIL 3.0 kompatibel ist und für den auch Versionen für alle gängigen Betriebssysteme, sowie Plugins für ausgewählte Browser existieren.[22]

---

[18] Vgl. Thierry Michel (2011), Synchronized Multimedia, Online im Internet: URL: http://www.w3.org/AudioVideo/, [04. 01. 2012].
[19] Vgl. World Wide Web Consortium (2008), SMIL 3.0 verbessert Standard für Synchronisation bei Multimedia. W3C integriert Industrie- und Benutzererfahrung in ein Set von Eigenschaften, Online im Internet: URL: http://w3c.de/Press/2008/smil3-pressrelease.html.de.html, [04. 01. 2012].
[20] Vgl. Koivunen, Marja-Ritta/ Jacobs, Ian (1999): Accessibility Features of SMIL. W3C NOTE 21 September 1999, Online im Internet: URL: http://www.w3.org/TR/SMIL-access/, [04. 01. 2012].
[21] http://www.ambulantplayer.org/
[22] Vgl. Thierry Michel (2011), Synchronized Multimedia, Online im Internet: URL: http://www.w3.org/AudioVideo/, [04. 01. 2012].