

Barrierefreie-Navigation

Web-Seiten sollte von Menschen mit Einschränkungen gelesen werden können.

1) Einschränkungen

- Visuell
 - Maus nicht immer bedienbar
- Motorisch
 - Tastatur muss herhalten
 - Mauseinsatz kommt infrage
- Kognitiv
 - Schlechte Erfassungsgabe, ...

2) Hilfsmitteln

- Mauseinsatz
 - Kopfmaus (beispielsweise eviacam)
 - Tastatur Mausemulation (beispielsweise orca)
- Spracheingabe
- Screenreader (beispielsweise orca)
 - Braille-Zeile
 - Sprachausgabe

3) Rechner und Smartphone / Tablett

- Für Rechner Bedienung über die bereits erwähnten Hilfsmitteln.
- Für reine Touchgeräte im OS eingebauten Screenreader
 - Android Talkback
 - iOS voiceOver

Das Konzept und die Funktionsweise sind sehr unterschiedlich. Bei der Touchgeräte werden nur Klicks und Tastatur-Eingabe in Elementen wie Input, Links und Schaltelementen weitergereicht. Das Fokus wird nicht immer behandelt.

Auf Rechner werden die Tastatur und Mauseingabe den Browser, gegebenenfalls nach Vorverarbeitung weitergereicht.

Die Unterschieden erlauben, auf Smartphone oder Tabletten kein saubere Reaktion der GUI, manche Events fehlen.

4) Tastatur Navigation mit und ohne Screen-Reader

- Ohne Screen-Reader:
 - Tab: von fokussierbare Element zur fokussierbare Element
 - Space: Auswählen, Aus- Einklappen
 - Wagen Rücklauf: Ein- Ausklappen, in Formularen posten.
 - Cursor hoch/runter Bei Selectionsbox nutzbar

- Mit Screen-Reader
 - Wie ohne
 - Cursor Tasten:
 - Buchstabenweise in einer Zeile bewegen.
 - Eine Zeile hoch oder runter

5) Caret/Browser Navigation

- Die Caret Navigation erlaubt einen Zeichen- oder Zeilenweise Navigation in der Applikation. Mittels F7 kann sie manchmal ein- ausgeschaltet werden.
- Die Browser Navigation verschiebt nur das Ausgabefenster.
- Screenreader wie Orca oder NVDA (Windows) sorgen dafür, dass eine Caret Navigation im Browser verwendet wird,

6) Navigation Titel zu Titel

- Orca und NVDA können von Titelzeile zur Titelzeile springen (Taste H)
- oder gezielt zum nächste Titel (H1 bis H6) mittels die Tasten 1 bis 6 zu springen.

7) Tastatur Eingabe

- Am PC wie gewohnt
- Auf Smartphone oder Tabletten wird nach Doppelklick auf das Bildschirm die Tastatur eingeblendet.
 - Mit Verschieben des Fingers, werden die erreichten Tasten angesagt, durch heben und zweifach klicken erfolgt die Übernahme des Zeichens.

8) Sinnvoller Aufbau der Webseite

- Reduzieren der Anzahl an Tastenbetätigungen.
- Skiplinks vorsehen, Aber nicht viel das Hauptinhaltsbereich sollte sofort erreichbar sein.
- Mit H Elementen die Navigation zu relevante Abschnitte erleichtern.
 - H1 entspricht Beginn der Seite.
 - H2 Zeigt auf Bereiche wie Inhalt.
 - H3 Zeigt auf den nächsten Untertitel unterhalb von h2
 - ...
- Touch Geräte wandern von Abschnitt zum Abschnitt, der Browser bekommt es nicht mit-

8.1) Navigationsmenü einer Webseite

Zahlreiche Seiten können erreicht werden. Unterseiten sind vorhanden.

- Hauptkategorien vorsehen, sie sollen erst nach Auswahl Unterkategorien/Seiten anbieten.
- Es sollte auch ohne Javascript soweit funktionieren.

9) Realisierung eines Navigationsmenüs

9.1) Details

- Details sind Elemente, die es ermöglichen, die unterhalb eingefügten Abschnitte ein-/auszublenden.
- Das erste Element ist ein `<summary>` und beinhaltet die Bezeichnung des Abschnittes (oder Menü-Kategorie).

9.1.1) Beispiel HTML5

```
<nav>
  <details>
    <summary aria-label="Hauptmenü" id="menu-top">Menü;</summary>
    <ul>
      <li><a href='.' >HOME</a></li>
      <li>
        <details><summary>Navigation</summary>
        <ul>
          <li><a href="./?1">Seite 1</a></li>
          <li>
            <details><summary>Seite 3.X</summary>
            <ul>
              <li><a href="./?3,1">Seite 3.1</a></li>
            </ul>
          </li>
        </ul>
      </li>
      <li>
        <details><summary>Aktion</summary>
        <ul>
          <li><button id='action' aria-label="Hell/Dunkel"
            type="submit">Ausführen</button></li>
        </ul>
      </li>
    </ul>
  </details>
</nav>
```

Das Menü kann unterschiedlichen Untermenü-Ebenen besitzen. Jede Ebene kann ohne Javascript aus- oder eingeklappt werden.

Für den ausreichend sehenden ist es leider nicht so komfortabel, das Schließen einer Ebene bedarf immer einer Handlung.

Dies kann mit wenig Javascript verbessert werden. Klicks außerhalb des Menüs oder Tastatur-Kursor-Veränderungen müssen das gesamte Menü oder Untermenü automatisch schließen.

9.1.2) Events und Zeitliche Reihenfolge

- Tabulator Navigation
 - **Fokus In** auf angesprungenen Objekt **A**
 - **Fokus Out** auf **A**; **Fokus In** auf Objekt **B**
 - Verlassen per Cursor-Taste eventuell **Nichts**
- Maus Navigation
 - **Fokus In** auf **A**; **klick** auf **A**
 - **Fokus Out** auf **A**; **Fokus In** auf **B**; **Klick** auf **B**

◦ Klick Außerhalb Fokus Out

Als Events eignen sich die Events **Focusout**, **keydown** und **Click**. Die Aktion für **Focusout** muss allerdings erst nach dem Ausführen der von den anderen **Events** erfolgen, sonst kann nicht gezielt die entsprechenden **Parent** Elemente behandelt werden. Dies geht einfach, durch Aufruf der entsprechende Funktion mittels:

```
function focusOut(e) { setTimeout(toogleDetails, 1); }
```

Diese Funktion wird, sobald, einen der Navigationselemente der Fokus verliert durch einem passenden EventListener aufgerufen. Dies wird erreicht durch:

```
document.querySelectorAll('nav summary, nav a, nav button').forEach(a => {  
    a.addEventListener('focusout', focusOut);});
```

Falls das Menü durch ein Cursor-Taste verlassen wird, muss es erkannt werden. Ein einfacher Weg, ist das Abfangen des **keyup** Events auf **Body** Ebene und zu prüfen ob das Ursprung Element ein Kind des Navigationsbereiches ist. Klick Events können ebenso ausgewertet werden.

In unserem Script werden die Events **click** (Mausbedienung) und **keyup** ausgewertet.

Da viele Events, vom Browser auf Touchgeräte nicht empfangen werden, entspricht die Bildschirmdarstellung nicht unbedingt das Erwartete,

9.1.3) Beispiel Javascript für den Sehenden

```
var allDetails = null;  
function closeAll() {allDetails.forEach( det => { det.open = false;});}  
window.addEventListener('DOMContentLoaded',function() {  
    var menu = null; var navi = null;  
    function closeOnlyOpen() {  
        allDetails.forEach( det => {  
            if ( !det.contains(document.activeElement) ) det.open = false;});  
    }  
    function toogleDetails() {  
        if ( document.activeElement ) closeOnlyOpen(document.activeElement);  
        else closeAll();  
    }  
    function outHandler(e) {  
        if ( navi.contains(e.target) && e.key == 'Escape' ) {  
            e.preventDefault(); closeAll(); menu.focus();  
        } else if ( !navi.contains(e.target) ) closeAll();  
    }  
    allDetails = document.querySelectorAll('nav details');  
    menu = document.querySelector('#menu-top');  
    navi = document.querySelector('nav');  
    document.querySelector('body').addEventListener('keyup', outHandler);  
    document.querySelector('body').addEventListener('click', function (e) {  
        if ( !navi.contains(e.target)) closeAll();  
        else closeOnlyOpen();  
    });  
});
```

9.2) Hamburger Menü

Hamburger Menüs werden auf Smartphone und auf einige Webseiten verwendet

- Pro:
 - Für normalen Anwender Untermenüs schließt sich nicht selbständig.
 - Für nicht Maus-Benutzer weniger Klicks.
 - Nur ein Layout (sowohl für mobile Geräte als f+r PC).
- Contra:
 - Für normalen Anwender ein wenig mehr Klicks.

9.2.1) Hamburger Menü Symbol

Für Smartphone haben sich Hamburger Menüs bewährt. Das Menü Eintrittspunkt besteht aus ein Symbol bestehend aus drei, vertikal ausgerichtet, horizontale Strich.

Solch ein Symbol kann leicht realisiert werden

CSS:

```
summary svg {  
  width:1em;  
  height:1em;  
  vertical-align:middle;  
  stroke: #00a;  
}
```

HTML:

```
<summary>  
  <svg viewBox="0 0 100 100" width="16" height="16">  
    <line x1="5" y1="85" x2="95" y2="85" stroke-width="10" />  
    <line x1="5" y1="50" x2="95" y2="50" stroke-width="10" />  
    <line x1="5" y1="15" x2="95" y2="15" stroke-width="10" />  
  </svg>  
</summary>
```

Die Svg Beschreibungssprache ist XML, deswegen müssen in sich geschlossene Elementen, hier <line /> entsprechend beendet werden.

Ein Arial Label für das Summary Element ist imperativ.