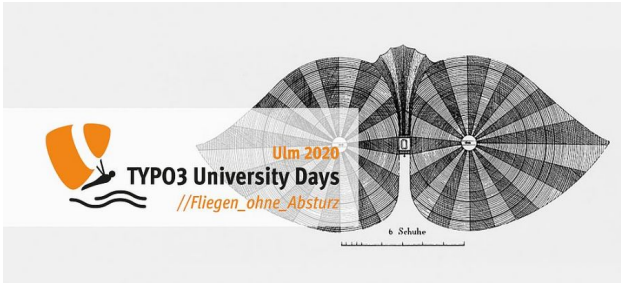
Den TYPO3 Seitenbaum barrierefrei machen



16. September 2020

Michael Telgkamp

mindscreen.
München | Berlin

Michael Telgkamp

- Zuhause in München
- Webentwickler bei **mindscreen**
  Fokus auf Barrierefreiheit
- Projekte mit TYPO3 seit 2008
- Seit 2019 Lead der **TYPO3 Accessibility Initiative**
- Im Jahr 2020 das **TYPO3 Accessibility Team gegründet**

Über das TYPO3 Accessibility Team

- Der Fokus des Accessibility Teams ist die Barrierefreiheit von TYPO3 zu verbessern
  - Wir teilen unsere Expertise
  - Wir suchen Unterstützer für ein barrierefreies TYPO3
- Initiative Gegründet 2016 und 2019 wiederbelebt
- Team gegründet 2020

Ihr könnt uns gerne ansprechen!

Der Status des Seitenbaums

Der TYPO3 Seitenbaum ist seit TYPO3 v9 ein SVG Element

- Keine Links, nur SVG Gruppen und Elemente mit click-handler für die Interaktionen mit der Maus
- Keine Möglichkeit den Seitenbaum mit der Tastatur zu nutzen (noch nicht einmal fokussierbar)
- Frühere Versionen von TYPO3 nutzen einen <a> Tag für jede Seite (nicht sehr gut nutzbar)

Das Issue auf Forge

Issue “Page tree is not accessible via tab key anymore” erstellt 2018-10-31
https://forge.typo3.org/issues/86818

Ein erster Patch hat einen <a> Tag für jede Seite hinzugefügt

- Das hätte das alte Verhalten wiederhergestellt
- Bei vielen Seiten hätte man aber für jede Seite einmal den TAB key drücken müssen
- Schließen und öffnen von Unterbäumen wäre mit der Tastatur nicht möglich

Ich stellte fest, dass ich das nicht alleine schaffen würde

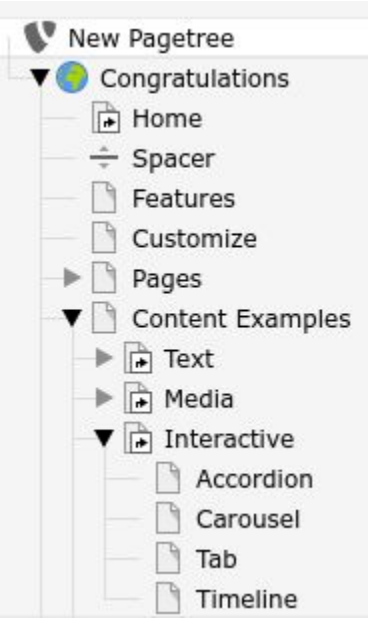
- Anfang 2019: die ersten Idee, dass ich den Seitenbaum so verbessern möchte, dass es jedem hilft (der mit der Tastatur arbeitet)
- Erste Kontakte auf Slack geknüpft
- Auf Events wie dem TYPO3camp Munich mit anderen gesprochen
  https://www.typo3camp-munich.de/ 
- Im September 2019 die TYPO3 Accessibility Initiative wiederbelebt
- Im September 2020 das Accessibility Team gegründet.
  https://typo3.org/community/teams/accessibility/ 

Der Plan den Seitenbaum wirklich zu fixen

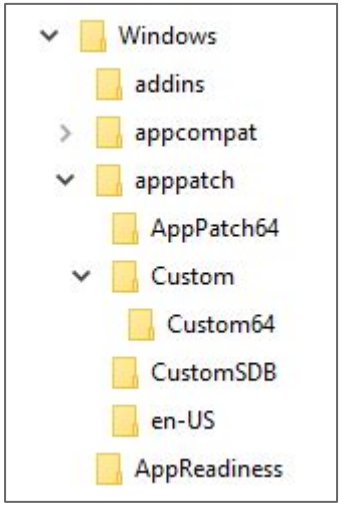
Im Oktober 2019 habe ich bei der T3INIT19 teilgenommen.
https://t3init19.typo3.com/

Meine Vorstellung war **den Seitenbaum an einem Tag während der Woche zu reparieren**.

Der Seitenbaum und Barrierefreiheit



- Ich schaute mich nach der besten Methode um und fand das Umsetzungsbeispiel Tree View in WAI-ARIA w3.org/TR/wai-aria-practices-1.1/#TreeView
- Das Beispiel verhält sich wie der Windows Dateibrowser
- Bekannte Bedienung für die meisten Nutzer
- Intuitiv, benutzt die Pfeiltasten zur Navigation



Begriffe für die Baumansicht

- **Node** Ein Knoten im Baum.
- **Root Node** Der Basisknoten im Baum.
- **Child Node** Ein Knoten mit einem Elternknoten.
- **End Node** Ein Knoten ohne Kinder.
- **Parent Node** Ein Knoten mit einem oder mehreren Kindern.
- **Open Node** Ein offener Elternknoten, seine Kinder sind sichtbar.
- **Closed Node** Ein geschlossener Elternknoten, unsichtbare Kinder.

Was erreicht werdem soll

Alles, was man mit der Maus machen kann, soll auch mit der Tastatur möglich sein.

- Man muss Elternknoten öffnen und schließen können
- Wenn man einen Knoten auswählt, soll der Inhalt im rechten Frame angezeigt werden
- Man muss jeden Knoten mit der Tastatur fokussieren können

Die Struktur, die ich beim Seitenbaum erwartete

- New Pagetree
  - Congratulations
    - Home
    - Spacer
    - Features
    - Customize
    - Pages
      - Default
      - 2 Columns
      - ...

Warum es schwerer wurde als gedacht

Der Baum ist ein SVG Bild, nicht als Baum strukturiert

Es gibt einfache Listen von SVG Elementen

```
<svg version="1.1" ...>
    <g class="nodes-wrapper" ...>
        <g class="nodes-bg">
            <rect width="100%" height="20" ... rx="0" ry="0">
            ...
        </g>
        <g class="links">
            <path class="link" d="M0 0 V20 H18"></path>
            ...
        </g>
        <g class="nodes" role="tree">
            <g class="node identifier-0_0" ...>...</g>
            ...
        </g>
    </g>
</svg>
```

Es gibt drei Gruppen von DOM Elementen aber keine Informationen über Ebenen, Eltern- oder Kindknoten

- **Nodes-bg** eine flache Liste von SVG Rechtecken <rect>, die für das hervorheben der aktuellen Auswahl und die Erkennung von Mausklicks genutzt werden
- **Links** eine flache Liste von SVG Linien <path>, die eine Linie für den Seitenbaum zeichnen
- **Nodes** eine flache Liste von SVG Gruppen <group>, die Symbole, Zusatzbilder auf Symbolen (overlays), den Text des Seitentitels und anderes enthalten

Warum es noch schwerer wurde

Jede Gruppe enthält nur die Knoten im DOM, die im Viewport des Browsers die sichtbar sind

- Nicht alle Knoten sind für den Browser verfügbar, man kann das DOM nicht nutzen um den Accessibility Tree korrekt aufzubauen

Es gibt keine strukturelle Gruppierung für Kind Elemente. Es ist nicht möglich herauszufinden ob die Nachbarknoten Kinder, Eltern oder Geschwister sind

- Das Javascript des Seitenbaums wird auch für den Kategoriebaum genutzt, der eine Mehrfachauswahl nutzt (es wäre schön, wenn das auch mit der Tastatur möglich ist)

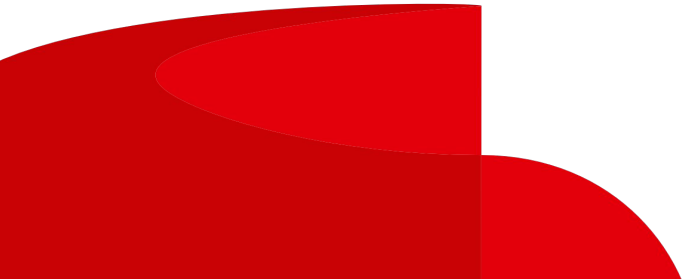
Nur der Seitenbaum? Geht eigentlich nicht

Das Javascript des Seitenbaums wird auch für den Kategoriebaum genutzt.

Für den Kategoriebaum gelten leicht veränderte Regeln

- Kein anzeigen der Kategorie, stattdessen Auswahl
- Mehrfachauswahl möglich

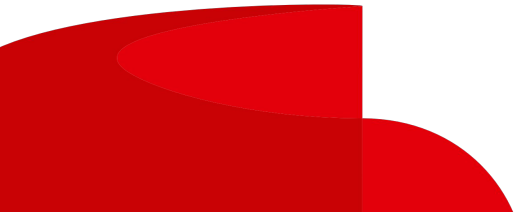
Wie man die Liste für den Browser zum Baum macht



Die Probleme

- Wir können die Liste nicht (einfach genug) zu einem Baum umbauen
- Es sind nur die Knoten im DOM, die aktuell sichtbar sind, wir müssen die korrekten Zahlen den Knoten zuweisen
- Es gibt keine Baumstruktur, der Browser kann also nicht erkennen, ob er Kinder hat

Wir müssen Lösungen finden, die dem Browser alle notwendigen Informationen zur Verfügung stellen.



Welche Informationen brauchen wir?

- Bei jedem Knoten
  - Die Ebene im Baum
  - Die Anzahl der Geschwister
  - Die Position in den Geschwisterknoten
- Zusätzlich für Elternknoten
  - Offen oder geschlossen
  - Die Gruppe der zugehörigen Kinderknoten
- Zusätzlich für die Kindknoten
  - Den Elternknoten

Dem Browser die neue Struktur beibringen

WAI-ARIA hilft auch hier um die Struktur zu redefinieren

- Man kann Elementen neue Rollen zuweisen
  https://www.w3.org/TR/wai-aria/#roles

  ACHTUNG! Wenn man mit ARIA Rollen zuweist,
  muss man das Verhalten selbst mit JavaScript implementieren

- Man kann mit ARIA Elementen Zustände und Eigenschaften zuweisen um dem Browser Informationen zu geben, die für die zugewiesenen Rollen notwendig sind
  https://www.w3.org/TR/wai-aria/#states_and_properties

Welche Elemente brauchen welche Rollen

- Wir können die nodes Gruppe als Baum nutzen
- Jeder node bekommt dann die Rolle treeitem

```
<g class="nodes" role="tree">
    <g class="node identifier-0_65" role="treeitem" ...
```

- Aber jetzt haben wir ein Problem, wenn der Knoten ein Elternknoten ist. Ihm muss eine Gruppe welche die Kinder enthält zugewiesen werden. Wir haben aber nur weitere <g> Elemente mit der role="treeitem" im DOM...

Kapern der **link** Elemente als Gruppen

- Da jedes Elternelement einen link path im links Abschnitt des SVG hat, können wir diesen kapern um die Rolle group dort zu definieren.

Das aria-owns Attribut der group sammelt alle Geschwister in einer Gruppe

```
<path class="link" id="group-identifier-0_65" role="group"
aria-owns="identifier-0_78 ... identifier-0_72 ...
identifier-0_66" d="M32 120 V140 H50"></path>
```

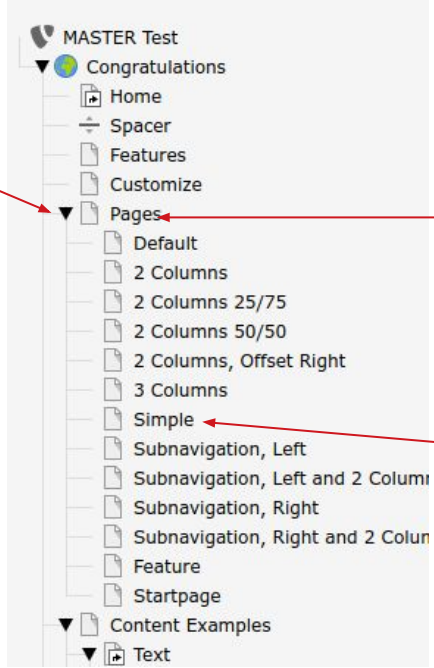
Hinzufügen der Informationen

Das aria-owns Attribut eines Elternelements verknüpft das Element mit der Gruppe der Kinder

```
<g class="nodes" role="tree">
    <g class="node identifier-0_65" id="identifier-0_65"
        role="treeitem" aria-owns="group-identifier-0_65"
        aria-level="2" aria-setsize="10" aria-posinset="5"
        aria-expanded="false" transform="translate(32,120)"
        data-state-id="0_65" title="id=65" data-table="pages"
        data-context="tree" tabindex="-1"><text ...
    </g>
</g>
```

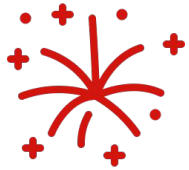
Werte für die Seiten “Pages” und “Simple”

id="group-identifier-0_65"
role="group"
aria-owns="identifier-0_78 ...
identifier-0_72 ... identifier-0_66"



id="identifier-0_65"
role="treeitem"
aria-owns="group-identifier-0_65"
aria-level="2"
aria-setsize="10"
aria-posinset="5"
aria-expanded="true"

id="identifier-0_72"
role="treeitem"
aria-level="3"
aria-setsize="13"
aria-posinset="7"



Der Browser kennt den Baum!

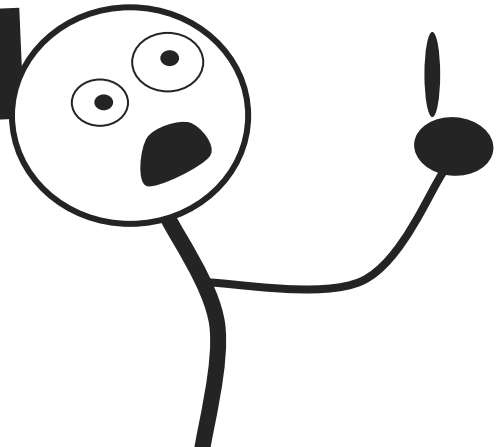
Wir sind fertig!

Der Browser kennt den Baum!

Sind wir fertig?

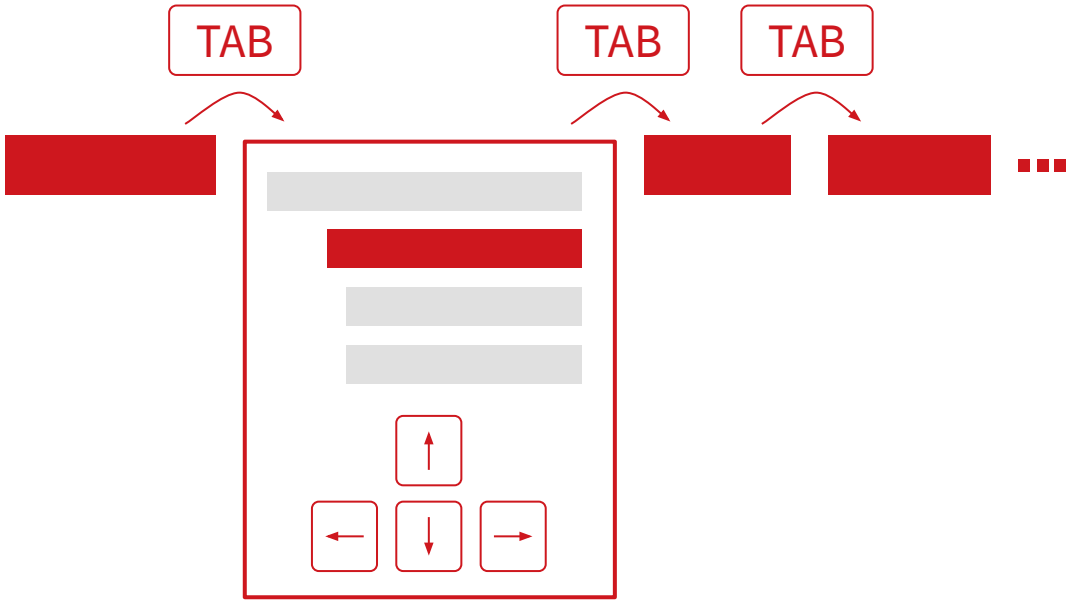
Moment!

Die Baumstruktur ist nicht genug!



Fokuswechsel und Tastaturnavigation

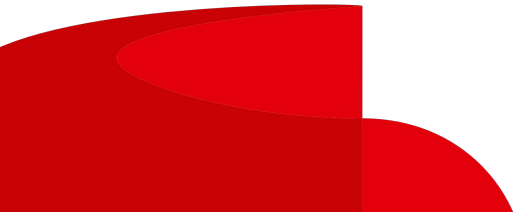
Mit der Taste TAB kann der Nutzer den Fokus auf ein Element im Seitenbaum setzen.



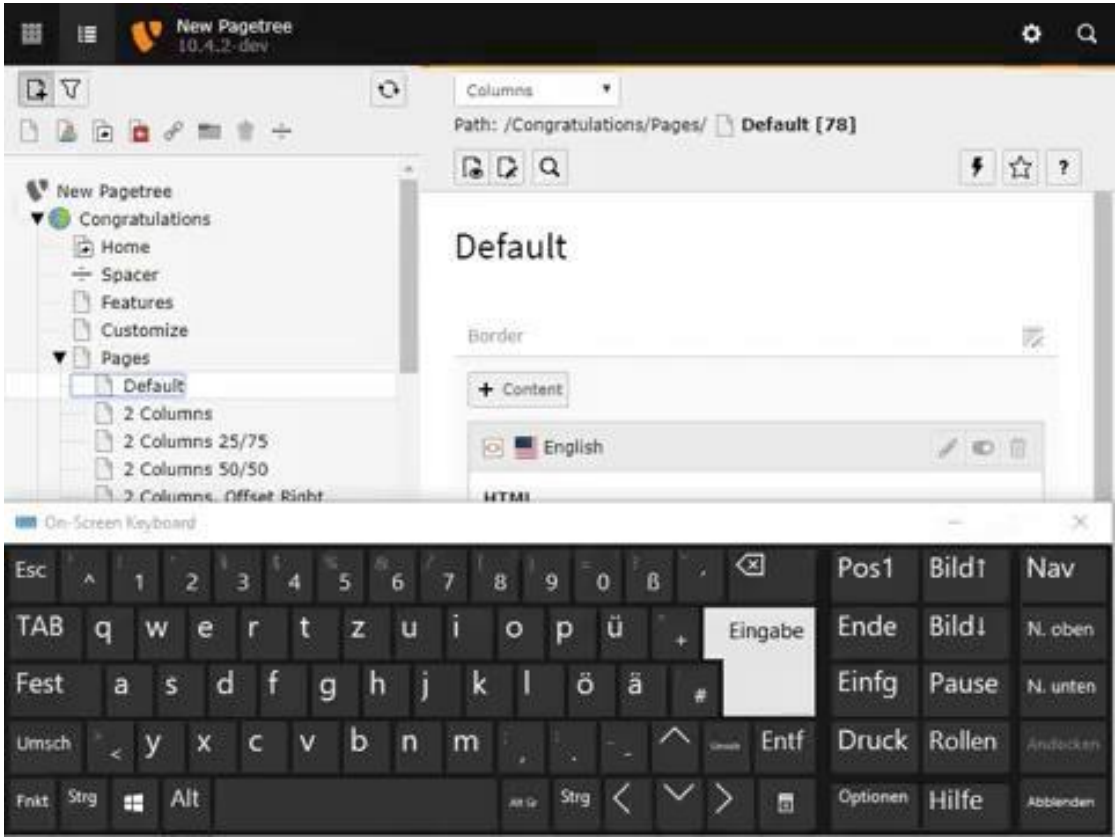
Innerhalb vom Baum Widget wird der Fokus mit den Pfeiltasten geändert.

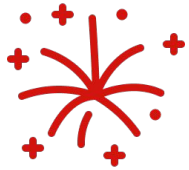
Anforderungen an die Tastaturbedienung

- Behandlung der Pfeiltasten: up down left right
- Behandlung der Home und End Tasten
- Behandlung der Enter Taste
- Behandlung der Space Taste, besonders im Auswahl Baum



Demo Tastaturbedienung





Wir sind barrierefrei!

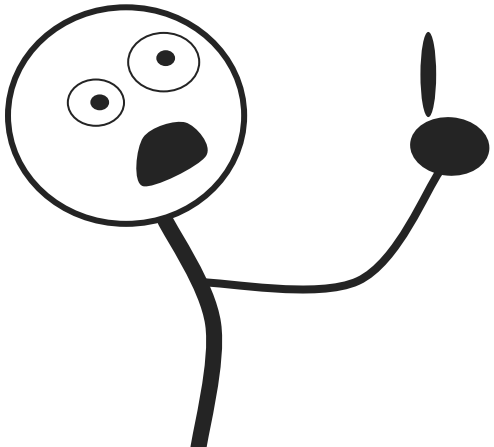
Wir sind fertig!

Wir sind barrierefrei!

Sind wir fertig?

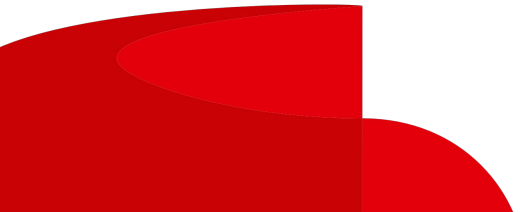
Moment!

Wir sind noch nicht fertig!



Sicherstellen, dass es auch in der Zukunft funktioniert

- Bei großen Open Source Projekten wie TYPO3 gibt es hunderte Entwickler die Code beitragen
- Wir müssen sicherstellen, dass die Tastaturbedienbarkeit des Seitenbaums durch neuen Code nicht wieder beeinträchtigt wird, ohne jede Änderung manuell begutachten zu müssen



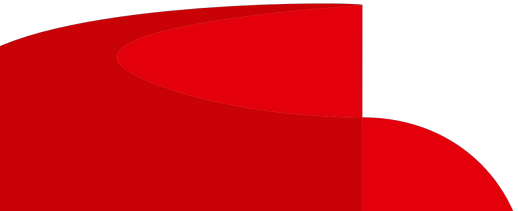
Akzeptanztests hinzufügen

- Bei TYPO3 wird jeder Merge-Request an Bamboo übergeben, dort werden alle automatisierten Test ausgeführt:
  - https://bamboo.typo3.com/
- Die Tests werden mit codeception ausgeführt
- Die Implementierung der Tests mit Tastatureingaben war schwierig
  - Eine Taste mehrfach schnell hintereinander zu drücken war nicht stabil, es wurden Tastendrücke vergessen. Nach einigen Anpassungen, auch im Test-Setup, hat dann aber alles geklappt.

Notwendige Frontend-Tests

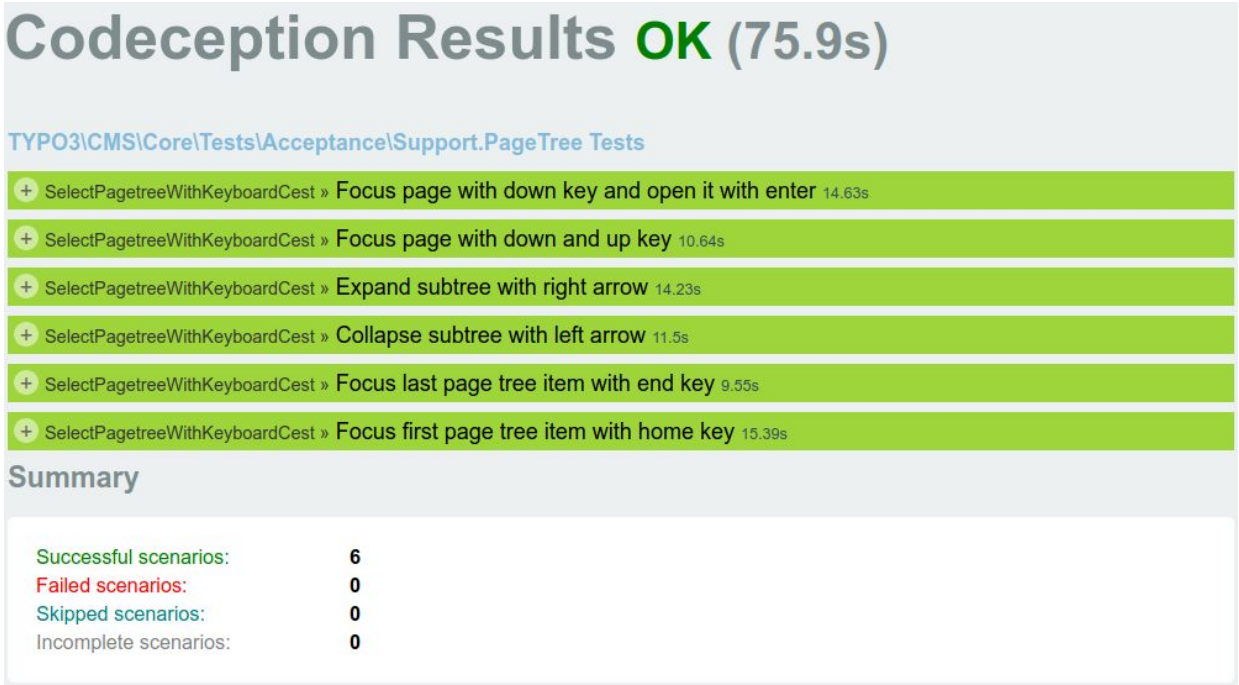
- Wir testen nur die Basisfunktionalität, da die Tests bei jedem Commit getestet werden
- Manuelles ausführen der Tests im Development Setup:

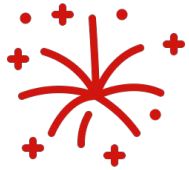
```
Build/Scripts/runTests.sh -s acceptancePagetree
```



Frontend Tests sind Teil der Aufgabe

Hier ein Screenshot der Tests, die von uns ausgeführt werden.



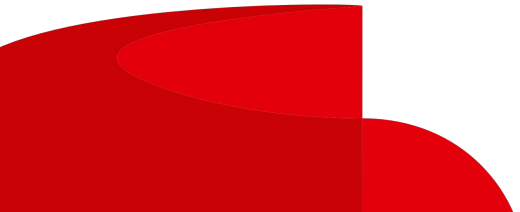


Wir sind barrierefrei!

Wir sind fertig!

Wie ich es hinbekommen habe

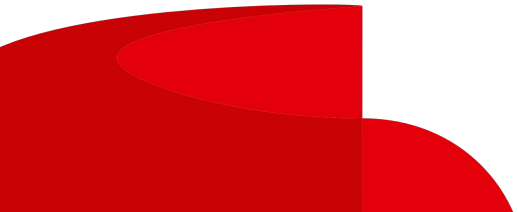
- Kontakt mit der Community aufgenommen (TYPO3 barcamps, Slack channel)
- Bei der Initiative Week 2019 teilgenommen (T3INIT19)
  - Während der Woche hatte ich sehr viel Hilfe
  - Wir hatten viele Diskussionen wie es gemacht werden kann (und sollte)
  - Wir haben alle notwendigen Informationen in aria Attributen hinzugefügt
  - Wir haben alle notwendigen Tastaturinteraktionen hinzugefügt
  - Wir haben die Tests hinzugefügt um alles am laufen zu halten



Die Idee “den Baum an einem Tag während der Initiative Woche zu korrigieren” hat nicht funktioniert.

Aber am Ende der Woche waren wir so weit, dass der Seitenbaum mit der Tastatur bedienbar war.

Am Ende des Jahres war dann endlich alles fertig und hat wie erwartet funktioniert, dass es in den master gemergt wurde.

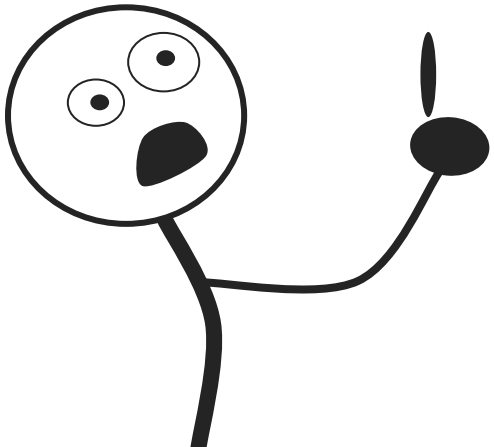


Wir sind barrierefrei!

Sind wir fertig?

Moment!

Das war der Seitenbaum!

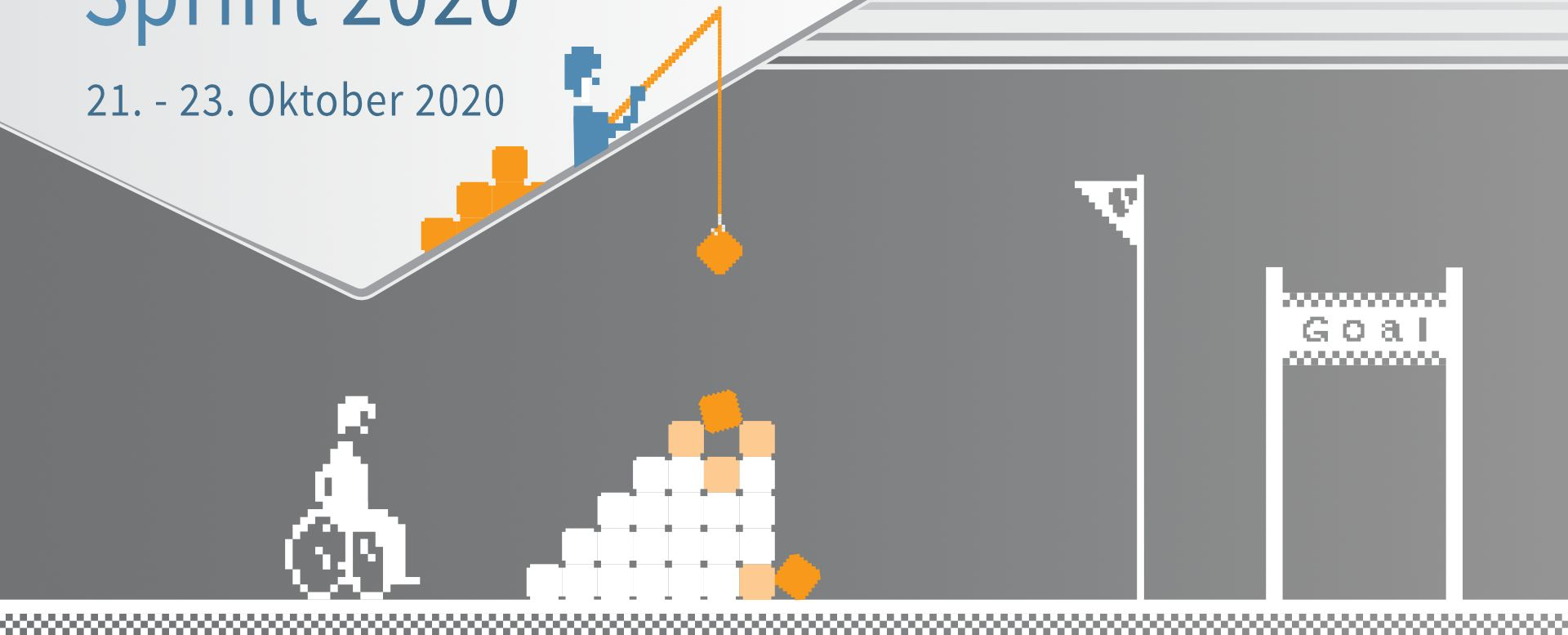


Accessibility

Sprint 2020

21. - 23. Oktober 2020

TYPO3



Vielen Dank!

Michael Telgkamp

michael.telgkamp@mindscreen.de