

# Inhaltsverzeichnis

## **1 Einleitung**

- 1.1 Vorwort
- 1.2 Das Unternehmen
- 1.3 Kurzbeschreibung der Projekte
- 1.4 Arbeitsumgebung

## **2 Hauptteil**

- 2.1 Einarbeitung
- 2.2 Der Graph Editor
  - 2.2.1 Allgemeines
  - 2.2.2 Der Form-Editor
  - 2.2.3 Themes
  - 2.2.4 Module
    - 2.2.4.1 SAWS-Receiver
    - 2.2.4.2 Process-Container
    - 2.2.4.3 Image-Module
- 2.3 Die Zeitleiste
  - 2.3.1 Vorgaben
  - 2.3.2 Die Parent-Komponente
  - 2.3.3 Die Child-Komponente
  - 2.3.4 Dokumentation
- 2.4 Betreuung und Hilfestellung
- 2.5 Kontrolle und Abnahme der Arbeit
- 2.6 Anwendung von Vorwissen
- 2.7 Zukunft der Projekte

## **3 Schluss**

- 3.1 Fazit
- 3.2 Literaturverzeichnis

## 1 Einleitung

### 1.1 Das Unternehmen

Die SAWS GmbH & Co. KG ist ein mittelständisches Unternehmen und seit dem Jahr 1999 im IT-Projektgeschäft, spezialisiert auf Shopsysteme für die Vermarktung von Produkten oder Dienstleistungen, sowie Software-Entwicklung für Produktion und Logistik in den Bereichen „Business to Business“ und „Business to Customer“. Seit 2005 sind sie Implementierungspartner der Firma CONTENTSERV. Mit ihrem Cloud-basierten Produktinformationsmanagement Programm wird eine Vielzahl an Produktinformationen, die an vielen verschiedenen Stellen benötigt werden, übersichtlich und effizient verwaltet.



Abbildung 1: Das Firmenlogo

### 1.2 Kurzbeschreibung der Projekte

Das erste Projekt war ein Web Graph Editor, mit dem man Prozessabläufe und deren Events dynamisch visualisieren kann. Dies war auch die Hauptaufgabe meines Praxissemesters von Anfang Oktober bis Mitte Januar. Es ist mit VueJs programmiert worden, einem JavaScript-Webframework. Die Idee hinter dem Produkt ist, dass Kunden ihre einzelnen Produktionsabläufe vom Einkauf bis Verkauf graphisch darstellen und diese auf einem Blick nachvollziehen können. Durch die live Visualisierung von Events wird es dem Kunden möglichst einfach gemacht, zu wissen, welche Schritte derzeit bearbeitet werden. Das Projekt wurde erfolgreich abgeschlossen. Im Falle von weiteren Kundenwünschen oder Ideen, die umgesetzt werden sollen, kann das Programm entsprechend erweitert werden.

Das zweite Projekt ist eine Zeitleisten-Komponente, an der man Events anhand von Punkten zeitlich ablesen kann. An dieser Zeitleiste kann der Kunde erkennen, wann Änderungen an Abläufen oder Daten vorgenommen wurden. Die Punkte, die die Events repräsentieren, sind prozentual zeitlich aufeinander abgestimmt sein. Die Zeitleiste ist dynamisch anpassbar und scrollable. Zudem besteht noch die Möglichkeit der Komponente mehrere Zeitleisten zu übergeben, die dann übereinander abgebildet und zeitlich voneinander stimmig sind.

### 1.3 Arbeitsumgebung

In der SAWS wird eigenständiges Arbeiten und Zeitmanagement erwartet. Den Mitarbeitern und Mitarbeiterinnen wird viel Vertrauen entgegengebracht und entsprechende Freiräume gewährt. Die Arbeitszeiten sind frei wählbar und werden auch nicht in einem Zeiterfassungssystem erfasst. Jeder kommt und geht nach eigenem Ermessen. Das übliche gemeinsame Mittagessen war wegen Corona beschränkt.

Allgemein wird mit NetBeans oder Visual Studio Code auf Windows 10 oder Linux Rechnern mit JavaScript oder PHP programmiert und jeder arbeitet zunächst an seinem Rechner im Entwicklungsmodus, bevor die Änderungen auf die live Server committed werden. Es gibt regelmäßige, kleinere Meetings mit dem Vorgesetzten, um sich zu beraten, wie gut das bisher geleistete war und was an Arbeit noch bevorsteht.

## 2 Hauptteil

### 2.1 Einarbeitung

Die ersten zwei Wochen habe ich mich darauf fokussiert, mich in die Firma einzufinden, die Kollegen besser kennen zu lernen und Programme zu lernen. Für mein vorgesehene Projekt musste ich mich in drei neue Bereiche der Informatik simultan einarbeiten, was zeitlich die meiste Einarbeitungszeit beansprucht hat. Diese waren CSS (Cascading Style Sheets), eine Stylesheet-Sprache für elektronische Dokumente [1], HTML (Hypertext Markup Language), eine Sprache für die semantische Beschreibung von verschiedenen Dokumententypen [2], und VueJs, ein progressives Framework für JavaScript (eine Programmiersprache für „die Entwicklung dynamischer Webseiten“ [3]) zum Erstellen von Benutzeroberflächen [4]. Diese stellen die Grundlage für das World-Wide-Web dar und sind somit unweigerlich miteinander verknüpft. Da ich für ein komplett neues Projekt eingeteilt wurde, deshalb musste ich mich nicht in bestehende Codes einarbeiten.

### 2.2 Der Graph Editor

#### 2.2.1 Allgemeines

Wie in der Einleitung schon erwähnt, war das erste Projekt ein typischer Graph Editor. Von einer Seitenleiste kann eine Form oder verschiedene Module in den Editor gezogen werden. Diese Module und Formen kann man mit Pfeilen verbinden und durch verschiedene Editoren bearbeiten. Dabei sollen die Formen verschiedene Geschäfte, Filialen, Werkstätten oder ähnliches darstellen, während die Module für verschiedene Aufgaben zuständig sind, zum Beispiel die Visualisierung von Events oder das Anzeigen von Zusatzinformationen. Wenn fertigen Graphen erstellt sind, können diese für Mitarbeiter freigegeben werden. Mit dem Aufrufen der Graphen kann der wesentliche Zustand eines Produktes oder eines Geschäftsablaufes übersichtlich dargestellt werden. Meine Hauptaufgabe war die Programmierung der grafische Benutzeroberfläche sowie die visuelle Gestaltung.

Für den Graph Editor wurde mxGraph verwendet, eine Open Source-Softwarekomponente zum Zeichnen von Grafiken. Dadurch, dass die SAWS auch international aktiv ist, wird das Interface nicht nur auf Deutsch, sondern auch auf Englisch gepflegt.

### 2.2.2 Der Form-Editor

Mit diesem Editor sind Formen individuell visuell veränderbar. Es kann zwischen verschiedenen Formen wie Dreieck, Kreis oder Wolke gewechselt werden. Dies ist mit einer Select Box realisiert. In dieser Select Box wird ein Vorschaubild und daneben der Name der Form wiedergegeben (vgl. Abbildung 2).

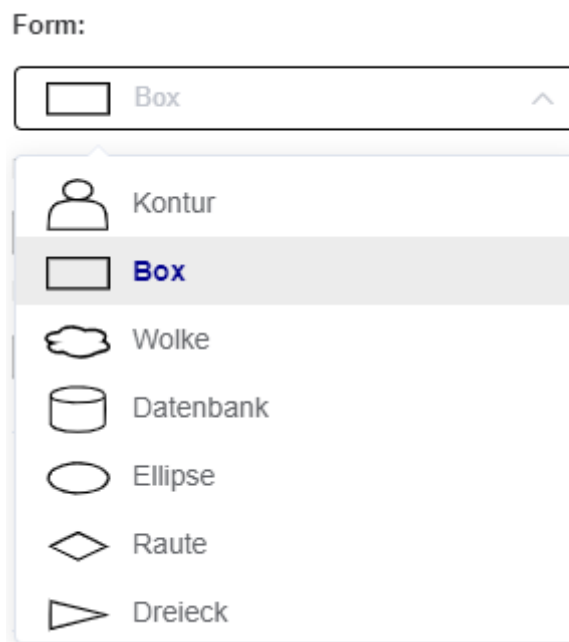


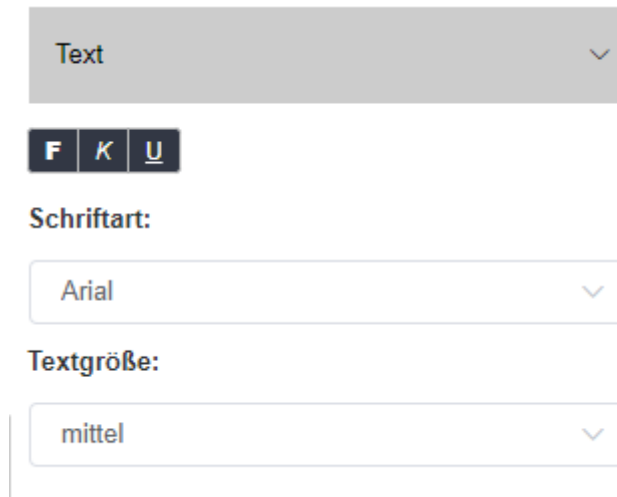
Abbildung 2: Select Box für die Auswahl der Form

Direkt darunter befinden sich drei Schaltflächen, mit der die Dicke der Ränder bestimmen werden. Es wurde zuerst ein Schieberegler dafür verwendet, dann aber in die genannten drei Schaltflächen umgewandelt. Der Hauptgrund war nicht zu viel Individualisierung für dieses Programm anzubieten, sodass alles möglichst schnell und einfach gestaltet werden kann. Der Kunde hat mit dieser begrenzten Auswahl immer noch genug Freiraum für kleinere visuelle Änderungen (vgl. Abbildung 3). Außerdem gibt es noch drei weitere Schaltflächen, die die Abrundung der Ecken bestimmen (vgl. Abbildung 3).



Abbildung 3: Buttons um die Stärke der Rundung und Randbreite einer Form zu bestimmen

In diesen Formen ist es möglich Texte einzufügen. Diesen Text kann man, genauso wie bei anderen Texteditoren, anpassen. Es gibt drei Buttons für Fett, Kursiv und Unterstrichen und zwei Select Boxen für Schriftarten und Schriftgrößen. Im Gegensatz zu anderen Texteditoren besteht hier nur eine übersichtliche Auswahl von fünf Schriftarten und fünf Schriftgrößen, um den Kunden eine schnellere Gestaltung des Graphen anzubieten (vgl. Abbildung 4).



The image shows a user interface for text formatting. At the top is a grey header bar with the word 'Text' and a downward arrow. Below this are three buttons labeled 'F' (Bold), 'K' (Italic), and 'U' (Underline). Then, there is a label 'Schriftart:' followed by a dropdown menu currently showing 'Arial'. Below that is a label 'Textgröße:' followed by a dropdown menu currently showing 'mittel'.

Abbildung 4: In diesem Abschnitt des Form Editors kann man seinen Text umgestalten

Als letztes gibt es eine Auswahl an verschiedenen Farben für verschiedene Bereiche. Bis zu sieben Farben stehen für den Text, die Formfüllung und den Rand der Form zur Verfügung (vgl. Abbildung 5).



The image shows a user interface for color selection. At the top is a grey header bar with the word 'Farbe' and a downward arrow. Below this are three sections of color swatches. The first section is labeled 'Füllfarbe:' and contains seven swatches: blue, dark blue, light blue, medium blue, dark blue, light blue, and white. The second section is labeled 'Randfarbe:' and contains five swatches: light blue, dark blue, blue, light blue, and white. The third section is labeled 'Textfarbe:' and contains five swatches: light blue, yellow, dark blue, blue, and white.

Abbildung 5: die Farbauswahl für eine Form

Bei der Auswahl einer Form sollen die ausgewählten Punkte hervorgehoben werden. Wenn mehrere Formen auf einmal ausgewählt werden, dann muss über jede Form iteriert werden, um

zu vergleichen, ob sie dieselben Eigenschaften haben. Nur wenn jede Form dieselbe Eigenschaft ausgewählt hat, wird diese Eigenschaft angezeigt (vgl. Abbildung 6).

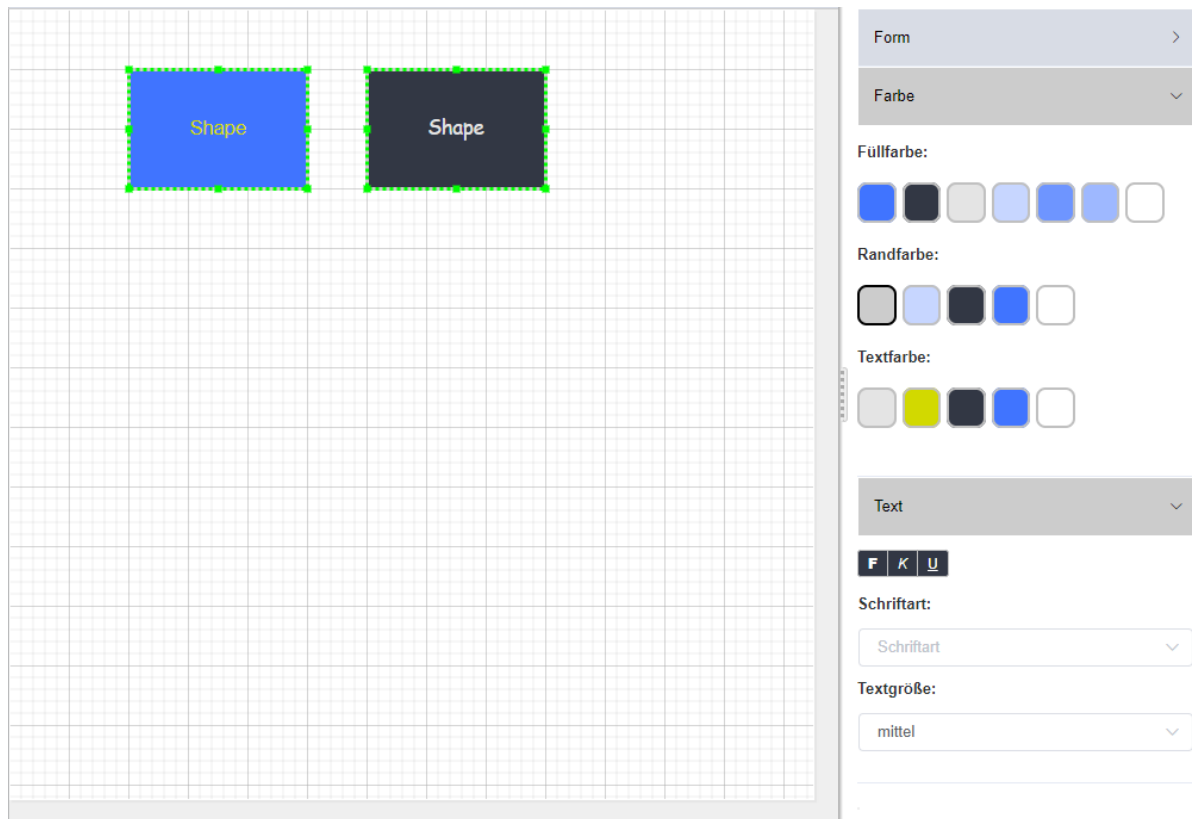


Abbildung 6:

Zwei Formen sind selektiert, es werden aber nur die Eigenschaften angezeigt, die sie gemeinsam haben. Beispielsweise haben sie dieselbe Textgröße, aber nicht dieselbe Schriftart.

Neben Formen gibt es auch Pfeile, die editiert werden können. Da Pfeile ebenfalls als Form interpretiert werden und eine sehr ähnliche Struktur der Visualisierung haben, fällt diese Aufgabe auch auf den Form-Editor. Pfeile sind in verschiedene Formen wie rund, eckig oder gerade darstellbar. Neben einer normalen Pfeilspitze kann man dies auch zu einer Box oder einem Kreis verändern. Man kann Pfeilen ebenfalls eine Beschriftung geben und sie können auch verschiedene Farben und Breiten haben. Der einzige Unterschied ist, dass ein Pfeil keine Füllfarbe hat. Dafür ist es möglich bei einem Pfeil die Art zu bestimmen, also ob er komplett oder gestrichelt sein soll. Das dynamische Anzeigen der Felder erfolgt über eine einfache if-Anweisung, die prüft, ob eine Form oder ein Pfeil ausgewählt ist.

The image shows a configuration panel for an arrow. At the top is a dropdown menu labeled 'Pfeil' with a downward arrow. Below it is the label 'Pfeilfrom:' followed by a dropdown menu showing a right-angle icon and the text 'Eckig'. Next is the label 'Linie:' followed by a dropdown menu showing a thick black horizontal line. Below that is the label 'Pfeilspitze:' followed by two dropdown menus: 'Start:' and 'Ende:'. The 'Ende:' dropdown shows a right-pointing arrow icon. At the bottom is the label 'Randbreite:' followed by three buttons: 'klein', 'mittel', and 'groß'.

Abbildung 7: Auswahlmöglichkeiten um einen Pfeil zu editieren

### 2.2.3 Themes

Um zu gewährleisten, dass die farbliche Gestaltung ebenfalls schnell und einfach ist, wurde beschlossen, Themes einzufügen. Ein Theme bestimmt, welche Farben in der Farbauswahl und welche Schriftarten zur Verfügung stehen. Für eine ansprechende Darstellung kann man Farben beliebig kombinieren. Eine wichtige Rolle spielen die Komplementärfarben. „Komplementärfarben liegen sich im Farbkreis gegenüber und sind sogenannte Ergänzungsfarben. Sie löschen sich gegenseitig aus, wenn sie miteinander gemischt werden.“ [5]. Dies war wichtig, weil der Text immer leserlich sein sollte und somit die Textfarbe sich von der Füllfarbe abheben muss. Die Textfarbe nicht zu hell sein, ansonsten wird der Text unleserlich. Anders verhielt es sich bei der Randfarbe, denn bei ihr waren sowohl Komplementärfarben als auch ähnliche Farben gut erkennbar. Beim Erstellen eines neuen Themes werden zuerst mehrere zueinander passende Füllfarben gesucht und anhand dieser die passenden Rand- und Textfarben bestimmt.

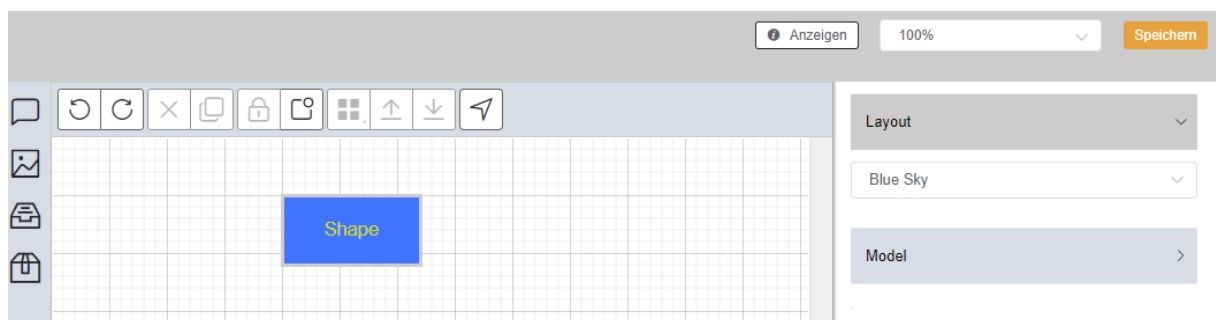


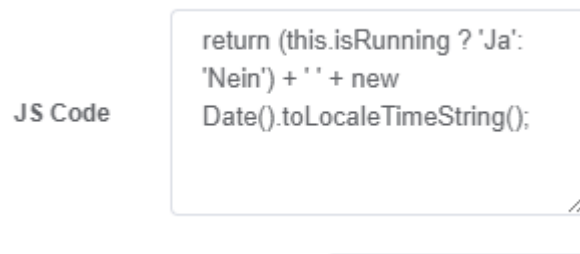
Abbildung 8: Theme – Blue Sky



## 2.2.4 Module

### 2.2.4.1 SAWS-Receiver

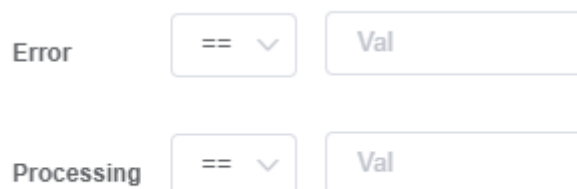
Das erste Modul ist der SAWS-Receiver. Seine Aufgabe ist es Event-Objekte zu empfangen und darzustellen. Es gibt drei verschiedene Zustände von Events. Neben „inaktiv“ können sie im „Processing“ oder im „Error“ Zustand sein. Um ein Event darzustellen, muss man im SAWS-Receiver ein neues Event-Ereignis mit mehreren Eigenschaften deklarieren, wie zum Beispiel einen Event-Namen. Anschließend wird bestimmt, bei welchen Event-Werten dieses Event-Ereignis ausgelöst wird und wann dieses Event-Ereignis in den „Processing“ oder in den „Error“ Zustand kommt und angezeigt werden soll. Dafür gibt es das Feld „JS Code“ mit einem hinterlegten kurzen Code, welcher Wert von dem Event-Objekte entscheidend ist und für die Zustandsprüfung verwendet wird. Zusätzlich kann man mit diesem Wert noch kleinere Änderungen mittels JavaScript Code vornehmen (vgl. Abbildung 9). Die Felder „Error“ und „Processing“ bestehen jeweils aus einer Select Box und einem Eingabefeld (vgl. Abbildung 10). Über eine Select Box wird ausgesucht, wann das Event bei welchem „JS Code“ Rückgabewert in den jeweiligen Zustand kommen soll, wie zum Beispiel „is true“ oder „is null“ (vgl. Abbildung 11). Das Eingabefeld wird nur beachtet, falls in der Select Box eine Vergleichsoperation wie „equal“ oder „less than“ ausgewählt ist (vgl. Abbildung 12).



JS Code

```
return (this.isRunning ? 'Ja':  
'Nein') + ' ' + new  
Date().toLocaleTimeString();
```

Abbildung 9: Eingabefeld JS Code



Error	== ▾	Val
Processing	== ▾	Val

Abbildung 10: Eingabefelder Error und Processing

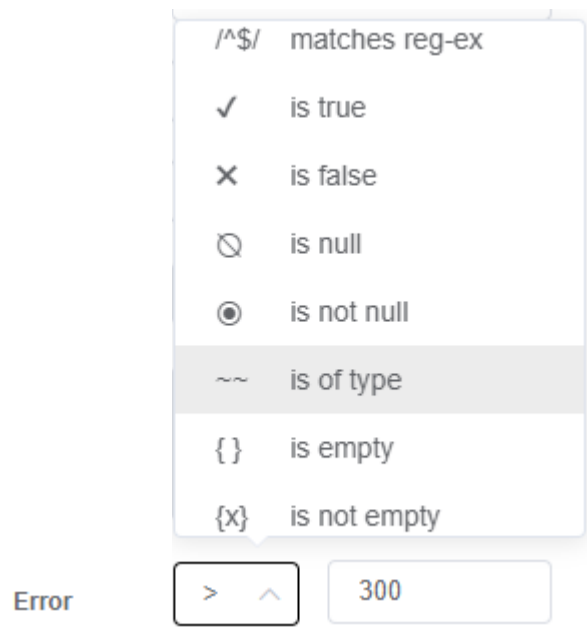


Abbildung 11: true or false Operationen für die Zustände Error und Processing

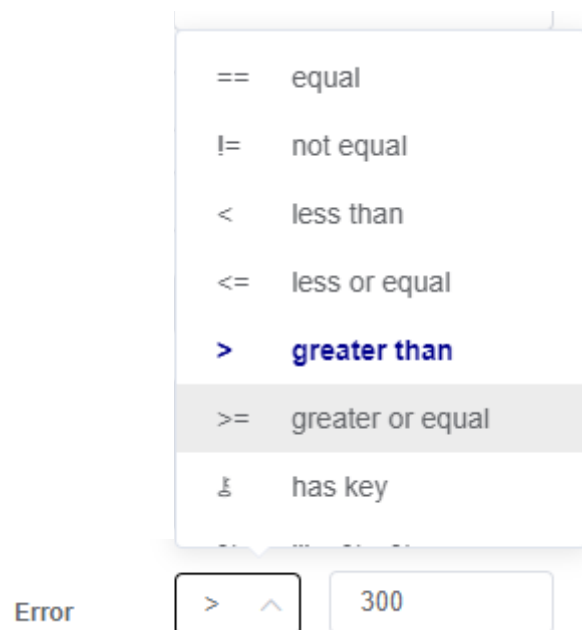


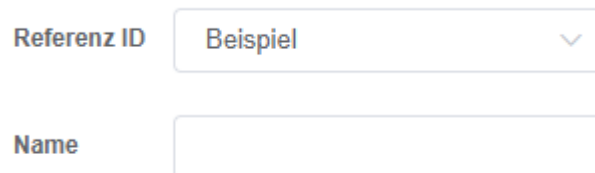
Abbildung 12: Vergleichsoperationen für die Zustände Error und Processing

Im Graph entsteht unter der SAWS-Receiver Form eine kleine Liste mit einer Aufzählung der Event-Ereignisse und deren aktueller Wert (vgl. Abbildung x). Wenn ein Event-Ereignis im „Processing“ Modus ist, wird die Form und alle anliegenden Pfeile grün und im „Error“ Modus rot umrandet. Ein Klick auf die Form lässt die Liste zusammenklappen oder ausfahren. Zudem kann die Liste in vier verschiedenen Richtungen von der Form ausgerichtet werden.

Außerdem gibt es noch das Auswahlfeld „Visible“. Darüber erfolgt die Bestimmung, ob dieses Event-Ereignis gar nicht, nur in diesem Graphen oder in allen Graphen angezeigt werden soll. Für das Anzeigen in anderen Graphen ist der Process-Container zuständig.

#### 2.2.4.2 Process-Container

Der Process-Container ist eine Referenz zu einem Graphen. Im Process-Container-Editor kann man einen bereits erstellten Graphen referenzieren (vgl. Abbildung 13) und bekommt dann alle Event-Ereignisse von allen SAWS-Receiveern, die im „Visible“ Feld auf „In allen Ansichten“ gestellt sind. Das sind die „Globalen Events“ des jeweiligen Graphen und werden sowohl im Editor als auch im Graphen auf die gleiche Weise angezeigt wie im SAWS-Receiver mit dem Unterschied, dass es nicht möglich ist die Felder zu bearbeiten.



The image shows a form with two input fields. The first field is labeled 'Referenz ID' and contains the text 'Beispiel' with a dropdown arrow. The second field is labeled 'Name' and is empty.

Abbildung 13:

Die Referenz ID ist der Name des Graphen, der Referenziert werden soll; Name ist der Name, der in diesem Graphen für die Referenz verwendet werden soll.

#### 2.2.4.3 Image-Module

Das Image-Module ist eine Form, die anstatt einer Füllfarbe ein Bild hat. Dabei soll das Bild von der Form ausgeschnitten werden (vgl. Abbildung 14). Es wird entweder per Upload oder mit einem Link dem Graphen hinzugefügt. Für das Bild gibt es verschieden einstellbare Größen (vgl. Abbildung 15). „passend“ bedeutet, dass das Bild sein Original Seitenverhältnis auf maximaler Größe in der Form beibehält, während „gestreckt“ die ganze Form ausfüllt. Die Prozentzahlen beziehen sich auf die originale Auflösung des Bildes. Das Bild wird immer mittig der Form platziert. Die Bildgröße ist limitiert auf 2 Megabyte und erlaubt sind nur die Dateiformate „.png“, „.jpg“, „.jpeg“ und „.gif“.



Abbildung 14: Ein Bild, dass von der ausgewählten Form ausgeschnitten wird.

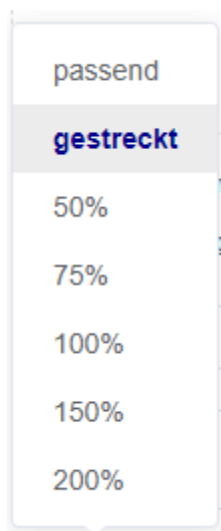


Abbildung 15:

Die verschieden auswählbaren

Bildgrößen

Form

☒ Bild

---

File

Dateien mit weniger als 2 MB  
.png / .jpg / .jpeg / .gif

Titel

Oben

Link

Größe

gestreckt

Abbildung 16:

Der gesamte Editor des Image-Modules

## 2.3 Die Zeitleiste

### 2.3.1 Vorgaben

Das zweite Projekt war die Programmierung einer Zeitleiste, die Events in einem Zeitstrahl in Form von Punkten darstellt. Jeder Punkt hat zeitlich prozentual den richtigen Abstand zu den anderen Punkten. Das heißt, dass Punkte, die nur wenige Tage voneinander entfernt sind näher zusammen liegen als Punkte die mehrere Wochen trennen. Unter jedem Punkt steht das dazugehörige Datum und darüber ein Name. Die Komponente kann mehrere Zeitleisten untereinander darstellen (vgl. Abbildung 17) und sind ebenfalls zeitlich miteinander stimmig. Wenn bei zwei Zeitleisten die erste beispielsweise später anfängt als die zweite, dann darf sie nicht auf derselben Höhe anfangen, sondern muss weiter rechts sein. Durch einen Klick-Event auf einen Punkt muss das dazugehörige Objekt an das Programm, dass diese Komponente einbindet, gesendet werden.

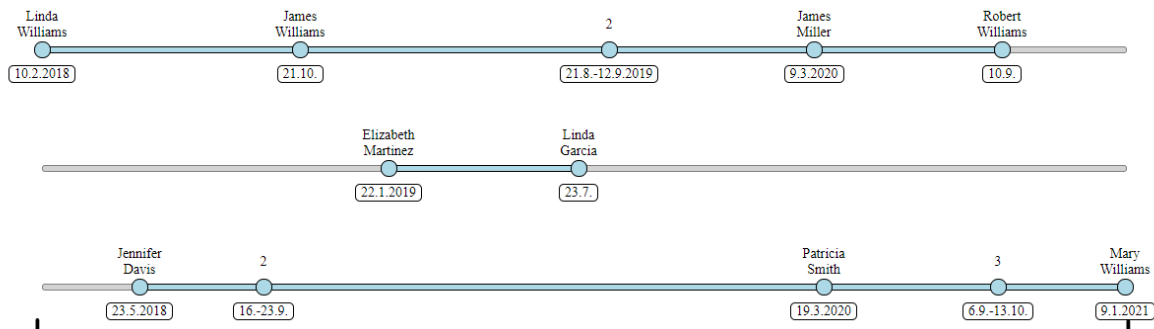


Abbildung 17: Eine Zeitleiste

### 2.3.2 Die Parent-Komponente

Die Komponente wurde in zwei Teilkomponenten aufgeteilt. Die erste Teilkomponente, die Parent-Komponente, bekommt ein Array aus Objekten als Property übergeben. Dabei erstellt sie für jedes Objekt eine Zeitleiste mit Hilfe der zweiten Teilkomponente. Diese Objekte haben zwei Werte. Zum einen haben diese Objekte einen Wert namens „content“. Das ist ebenfalls ein Array aus Objekten, aber diesmal repräsentiert jedes Objekt einen Punkt auf der Zeitleiste und braucht einen Objektwert namens „date“, das vom Objekt-Typen Date sein muss. Alternativ kann es auch vom Typ String sein, wenn man aus diesem String mit dem Date Konstruktor ein neues JavaScript Date Objekt machen kann. Neben dem „content“ Wert gibt es auch noch einen „settings“ Wert von Type Objekt. In diesem ist es möglich bestimmte Grundwerte vorzudefinieren, wie die Zeitleistenlänge oder den Mindestabstand, den die Punkte zueinander einhalten müssen. Die Werte „content“ und „settings“ werden der zweiten Teilkomponente, die Child-Komponente, als Property übergeben. Hierbei sei erwähnt, dass eine „shallow copie“ von den „content“ Arrays gemacht wird und diese dann übergeben werden, weil man in JavaScript Properties nicht verändern sollte. Eine „shallow

copie ist eine bitweise Kopie eines Objekts. Es wird ein neues Objekt erstellt, das eine exakte Kopie der Werte im Originalobjekt enthält. Wenn eines der Felder des Objekts Verweise auf andere Objekte sind, werden nur die Referenzadressen kopiert, d.h. nur die Speicheradresse wird kopiert.“ [6]. Für die Länge des Zeitstrahls gibt es zwei Optionen. Sie kann der Komponente in den „settings“ übergeben werden oder automatisch eine passende Länge von der Parent-Komponente abhängig von der Anzahl an übergebenen Events berechnen lassen. Damit die Länge nicht bei zu vielen Events explodiert, gibt es eine maximale Länge, die ebenfalls durch die „settings“ anpassen werden kann. Diese Länge ist unabhängig von der Zeitleiste vom ältesten Datum bis zum jüngsten Datum (vgl. Abbildung 17, schwarzer Strich am unteren Rand). Falls den Zeitleisten unterschiedliche Längen mitgegeben werden, wird die längste verwendet. Das jüngste und älteste Datum von allen Zeitleisten wird mit der Länge jeder Zeitleiste ebenfalls der Child-Komponente übergeben. Ein weiterer Wert, dem man einer Zeitleiste über die „settings“ übergeben kann, bestimmt, wie viele Tage zu einem Punkt zusammengefasst werden. Nimmt man als Beispiel den Wert zehn, dann geht das Programm so vor, dass es sich das älteste Datum anschaut und alle weiteren Einträge des Arrays, die innerhalb von zehn Tagen liegen, zu einem eigenen Array macht und dies dann die Einträge im kopierten „content“ Array ersetzt. Danach schaut es sich den nächsten Eintrag an und macht dasselbe. Das geschieht solange, bis es das ganze kopierte „content“ Array durch iteriert hat. Die Child-Komponente weiß, was sie machen muss, wenn sie im „content“ Array auf Arrays trifft.

### 2.3.3 Die Child-Komponente

Um die Abstände zwischen den Events zu berechnen, wird zunächst das Array nach Datum sortiert, sodass das älteste Datum den ersten und das jüngste Datum den letzten Platz belegt. Indem man dann das älteste Datum vom jüngsten Datum abzieht, kann die gesamte Zeit berechnet werden, die für die Darstellung benötigt wird. Danach iteriert man über das Array und mit der Differenz zum jüngsten Datum, multipliziert mit 100 und geteilt durch die gesamte Zeit, hat man den prozentualen Anteil des Punkts im Vergleich zur Gesamtzeit. Diese Prozentzahl wird dann nur noch auf die Breite der Zeitleiste verwendet, um den exakten Standort zu ermitteln. Das kann bei der Abbildung großer Zeitspannen allerdings zu Überlappungen führen, wenn die Punkte nur eine kurze Zeitspanne voneinander entfernt liegen oder die übergebene Breite nicht groß genug ist, um die Punkte nicht überlappend darzustellen. Deshalb werden Punkte, die nicht den Mindestabstand einhalten können, zu einem Punkt zusammengefasst. Dann sieht man statt einem Namen über dem Punkt eine Zahl, die die Anzahl der Punkte repräsentiert, die in dem Punkt zusammengefasst wurden. Bei einem Klick auf so einen Punkt erscheint ein kleines Fenster unter der Zeitleiste, die eine kleinere Zeitleiste mit den zusammengefassten Punkten beinhaltet (vgl. Abbildung 17). Ein weiteres Ziel war das Bearbeiten des „content“ Arrays, ohne dass alles neu gerendert

werden muss. Würde einfach das „content“ Array verändert, müsste jeder Eintrag überprüft werden, ob er sich verändert hat und immer noch derselbe ist, oder ob etwas dazu gekommen und/oder etwas weggefallen ist, weil die Komponente nicht weiß, ob nur der letzte Eintrag weggefallen ist oder ob es ein komplett neues Array mit einem Eintrag weniger ist. Bei einem einfachen Ersetzen des Arrays würde alles neu gerendert werden, was die Performanz verschlechtern würde. Deshalb fiel die Entscheidung, der Komponente eigene Events zu geben, die nur für das Bearbeiten des Arrays zuständig sind. So gibt es zum Beispiel ein „pop“ Event, dass den letzten Eintrag löscht und zurückgibt, oder ein „shift“ Event, dass den ersten Eintrag löscht und zurückgibt. Mit diesen Events weiß die Komponente, dass wirklich nur ein Eintrag wegfällt und sich nicht mehr im Array verändert. Neben der Bearbeitung des „content“ Arrays kann man auch zwischen zwei Daten filtern, sodass nur Punkte zwischen diesen Daten dargestellt werden. Theoretisch könnte man das „content“ Array auch außerhalb der Komponente filtern und neue übergeben, aber dann müsste wieder alles neu gerendert werden. Deshalb gibt es auch dafür ein Event. Es ist nicht möglich eine einzelne Zeitleiste zu filtern, sondern es werden immer alle gefiltert. Dadurch, dass das „content“ Array bereits sortiert ist, muss man nur noch die passenden Indizes herausfinden, von wo bis wo das Array angezeigt werden soll. Damit nicht immer von vorne nach hinten durch das Array iterieren werden muss, um die gültigen Punkte herauszufinden, wurde ein eigener Algorithmus erstellt. Der Algorithmus hat immer vier Daten zur Verfügung, nämlich die Daten vom ersten Punkt, vom letzten Punkt und von den zuletzt gefilterten Daten. Aber er speichert nicht diese Daten ab, sondern die Indizes, die sie im „content“ Array besitzen. Falls noch nicht gefiltert wurde, entsprechen die zuletzt gefilterten Daten dem ersten und letzten Punkt. Wenn nun neue Daten zum Filtern übergeben werden, berechnet der Algorithmus, welches Datum am nächsten von diesen neuen Daten ist und benutzt dann deren Index, um von dort aus der Suche zu beginnen. Dieser Algorithmus ist am effizientesten, wenn alle Daten vom ältesten bis zum jüngsten in etwa gleich verteilt sind. Wenn sie das nämlich nicht sind, können ineffiziente Berechnungen entstehen. Nehmen wir als Beispiel ein Array mit 10 Elementen. Die ersten neun Elemente sind jeweils nur einen Tag voneinander entfernt. Das letzte Element ist allerdings ein Jahr vom neunten Element entfernt. Wenn nach dem Datum des neunten Elements gefiltert wird, fängt der Algorithmus nicht vom zehnten Element an, sondern vom ersten, da das zeitlich näher ist, als das letzte.

#### 2.3.4 Dokumentation

Eine gute Dokumentation ist genauso wichtig wie die Lesbarkeit des Programms. Besonders bei einer Komponente, da diese von Personen verwendet wird, die nicht bei der Entwicklung dabei waren. Damit die Personen die Komponente in ihr Programm einbinden können, müssen sie verstehen, was die Komponente kann und wie man sie verwendet und einbindet. Dafür wurde eine Textdatei angelegt, in der genau beschrieben wird, welche Übergabewerte

die Komponente haben muss und welche optional sind. Zudem wird beschrieben, was die einzelnen Werte machen, welchen Datentype sie benötigen und was der Standardwert ist. Außerdem gibt es auch Beispiele, wie sie sinnvoll benutzt werden können. Ein weiterer Aspekt, der angesprochen wird, sind die CSS-Klassen. Da wird gezeigt, wie der Benutzer seine Zeitleiste individualisieren kann.

#### 2.4 Betreuung und Hilfestellung

Sowohl die Mitarbeiter als auch der Abteilungsleiter standen mir zu jeder Zeit mit Rat und Tat zur Seite, wenn ich Fragen hatte und gaben sich immer viel Mühe mit mir alles in verständlichen Worten zu besprechen. Zudem bekam ich immer genug Zeit und Freiraum, um Dinge selber auszuprobieren und zu experimentieren. Das Arbeitsklima war stets gelassen. Für die beiden Projekte wurden mir unterschiedliche Mitarbeiter zugeteilt. Gerade bei dem ersten Projekt in den ersten zwei Wochen habe ich noch viel Unterstützung gebraucht und durfte auch beim live Codieren zuschauen.

#### 2.5 Kontrolle und Abnahme der Arbeit

Es gab regelmäßige Besprechungen mit meinem Tutor und mit dem Vorgesetzten. In den Besprechungen mit dem Vorgesetzten haben wir das Scrum-Verfahren angewandt. Dazu haben wir zuallererst Brainstorming betrieben und unsere Ideen auf Zetteln festgehalten. Danach wurden die verworfenen Ideen aussortiert und die restlichen nach Priorität sortiert. Abschließend wurden die Aufgaben auf meinen Tutor und mich verteilt, indem sie auf eine Tafel geklebt wurden. Die Tafel hatte eine Tabelle und diese besaß mehrere Zeilen und Spalten. Zwei Zeilen für meinen Tutor und mich und drei Spalten, die „noch offen“, „in Bearbeitung“ und „Abgeschossen“ darstellen. Die Zettel wurden dann immer an die richtige Spalte geklebt. Allerdings ging es nicht immer nur vorwärts voran, denn manchmal mussten schon bereits abgeschlossene Zettel zurück in die „in Bearbeitung“ Spalte geklebt werden aufgrund nicht ausreichender Tests oder andere Zwischenfälle.

#### 2.6 Anwendung von Vorwissen

Die bereits in der Hochschule erlernten Programmiersprachen C, Java und Assembler konnte ich in der Arbeit nicht verwenden, da ausschließlich mit VueJs, ein progressives Framework für JavaScript, programmiert wurde, für das ich kein Vorwissen mitbrachte. Allerdings habe ich abseits des Programmierens Vorwissen aus einem Fach nutzen können.



Wie im Abschnitt 2.5 „Kontrolle und Abnahme der Arbeit“ erwähnt, haben wir das Scrum-Verfahren verwendet. Dies war ein Teil der Vorlesung Projektmanagement aus dem 4. Semester.

## 2.7 Zukunft der Projekte

Der Graph Editor kann schon Testweise von den Kunden verwendet werden. Derzeit ist dieses Projekt beendet. Sollte es aber neue Features benötigen, würde dieses Projekt fortgesetzt werden.

Die Zeitleiste wird schon in das neue, noch nicht erschienene Produkt eingebunden. Sie ist allerdings noch nicht fertig und wird auch noch nach dem Praxissemester weiterentwickelt.

### 3 Schluss

#### 3.1 Fazit

Das Praxissemester hat mir sehr gut gefallen. Es war eine willkommene Abwechslung zur Hochschule. Ich habe tiefe Blicke in die Abläufe eines Unternehmens bekommen und konnte viel neues Wissen sammeln das mir im weiteren Verlauf meines Studiums und in meiner zukünftigen Berufslaufbahn sehr nützlich sein wird. Webdevelopment ist ein Bereich, der heutzutage unverzichtbar für eine Firma ist. Zudem hat mir dieses Praktikum nochmal verdeutlicht, dass ich im richtigen Studiengang bin, weil es mir sehr Spaß gemacht hat.

Das heißt allerdings nicht, dass das Praxissemester keine Herausforderung für mich war. JavaScript, HTML und CSS waren mir komplett neu. Da alle drei Sprachen sehr einsteigerfreundlich sind, konnte ich sie mir schnell und eigenständig aneignen. In den ersten Wochen war es dennoch schwierig und umfangreich, zeitgleich drei komplett neue Sprachen zu erlernen.

Außerdem musste ich mich auch außerhalb der Komfortzone eines Programmierers begeben, da ich mich nicht nur um die Programmierung einer graphischen Benutzeroberfläche kümmern musste, sondern sowohl um ihr Design als auch das Farbdesign der Themes.

#### 3.2 Literaturverzeichnis

[1]

drafts.csswg, 13.02.2021, CSS Values and Units Module Level 4 – Abstract vom 02.02.2021

<https://drafts.csswg.org/css-values/>

„CSS is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, etc.“

[2]

w3, 13.02.2021, HTML5 A vocabulary and associated APIs for HTML and XHTML - 1 Introduction, 1.1 Background vom 28.10.2014

<https://www.w3.org/TR/2014/REC-html5-20141028/introduction.html#introduction>

„HTML was primarily designed as a language for semantically describing scientific documents, although its general design and adaptations over the years have enabled it to be used to describe a number of other types of documents.“

[3]

wiki.selfhtml, 13.02.2021, JavaScript, zuletzt bearbeitet am 19.01.2021

<https://wiki.selfhtml.org/wiki/JavaScript>

[4]

vuejs, 13.02.2021, Introduction - What is Vue.js?,

<https://vuejs.org/v2/guide/>

„Vue (pronounced /vju:/, like view) is a progressive framework for building user interfaces.“

[5]

themen.rainbowprint, 14.02.2021, Rainbowprint Blog – Komplementärfarben vom 01.03.2018 von Elena Kälber

<https://themen.rainbowprint.de/komplementaerfarben/#:~:text=Komplement%C3%A4rfarben%20liegen%20sich%20im%20Farbkreis,wenn%20sie%20miteinander%20gemischt%20werden.>

[6]

medium, 17.02.2021, Understanding Deep and Shallow Copy in Javascript - Shallow copy vom 08.12.2016 von Manjula Dube

<https://medium.com/@manjuladube/understanding-deep-and-shallow-copy-in-javascript-13438bad941c>

„Shallow copy is a bit-wise copy of an object. A new object is created that has an exact copy of the values in the original object. If any of the fields of the object are references to other objects, just the reference addresses are copied i.e., only the memory address is copied.“

Unterschriften zwecks Bestätigung der inhaltlichen Richtigkeit:

---

Unterschrift der Firma

---

Unterschrift des Studenten