Hauptseminar-Arbeit

Vue.js

Prüfungsleitung des Moduls
CS1025 Hauptseminar

von

Maximilian Biebl
`Matrikelnr.: 5323481`

am 26. April 2023

Dozent: Sebastian Süß

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Literatur und Hilfsmittel erstellt zu haben.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Gießen, 26. April 2023

Zusammenfassung

Heutzutage ist es standard, dass man mit einer Webseite interagieren kann und sich die Inhalte der Webseite dynamisch anpassen. Eine Webanwendung mit einer komplexen Benutzeroberfläche ohne die Verwendung eines Frontend-Frameworks zu entwickeln, ist aufwendig und fehleranfällig. Zudem besteht ein erhöhtes Risiko von Inkonsistenzen in Bezug auf Inhalt und Qualität, was die spätere Wartung erschwert. Für eine Webanwendung wird in der Regel ein Frontend-Framework verwendet, die aktuell gängigsten dieser Frameworks sind React, Angular und Vue.js. Diese Frontend-Frameworks bilden das Bindeglied zwischen der Logic in JavaScript und der Darstellung in HTML und CSS. In dieser Ausarbeitung möchten wir uns genauer mit dem Framework Vue.js beschäftigen und einen Vergleich zu React und Angular ziehen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Frontend Frameworks

In diesem Kapitel möchte ich zunächst allgemein in die JavaScript-Frontend-Frameworks einführen. Die Drei aktuell gängigsten Frontend-Frameworks React.js, Angular und Vue.js werden vorgestellt.
An eine moderne Webapplikation werden hohe Anforderungen an den Funktionsumfang durch den Benutzer gestellt. Die Entwickler einer solchen Webapplikation erwarten eine einheitliche Codequalität und einheitliche Strukturen für eine bessere Wartbarkeit der Webapplikation. Um bei dem Erstellen neuer Webapplikationen den Entwickler dabei zu unterstützen diese Ziele zu erfüllen, werden unter anderem entsprechende Frontend Frameworks verwendet.

> Front-end frameworks determine the logic, structure, design, behavior, and animation of every element you see on-screen when you interact with websites, web applications, and mobile apps. [1]

Man kann zwischen UI-Frameworks in HTML und CSS sowie JavaScript-Frameworks unterscheiden. Letztere dienen als Verbindung zwischen der Darstellung und der Logik im Frontend.

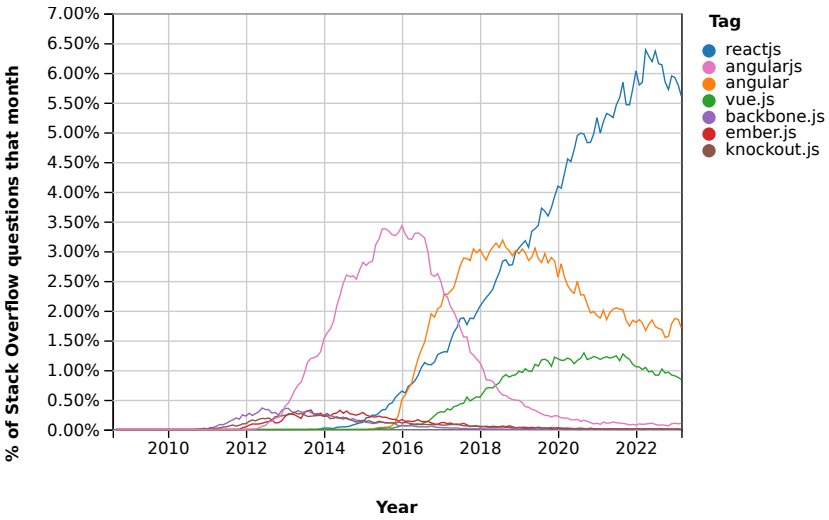

Abbildung 1.1: Stackoverflow Statistik zur Häufigkeit von nach Framework

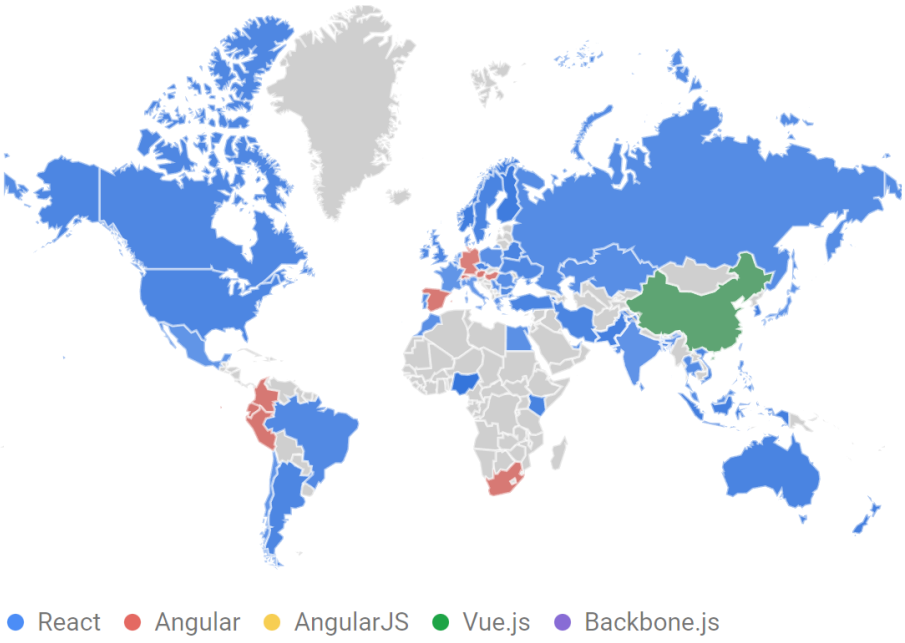

Abbildung 1.2: Googlge Trends Weltweiteverteilung von Suchanfragen nache Framework der letzten 5 Jahre

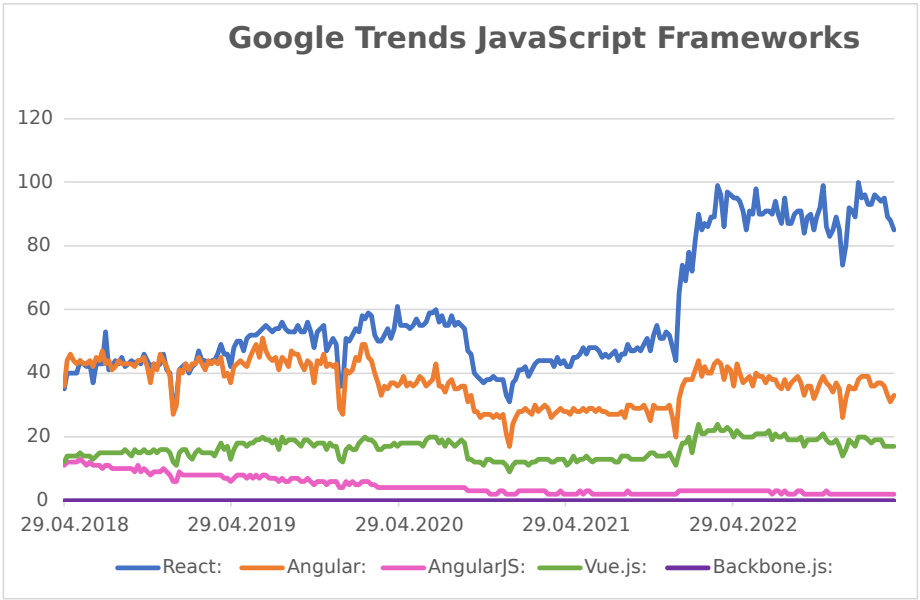

Abbildung 1.3: Googlge Trends zur Häufigkeit von Suchanfragen nache Framework der letzten 5 Jahre

1.1 Etwas Geschichte – oder ein paar Geschichten

Die ersten Bände von *The Art of Computer Programming* (TAOCP) von Donald Knuth wurden im Bleisatz gesetzt. In den 1970er Jahren aber starb der Bleisatz aus und wurde durch den Fotosatz ersetzt. Knuth war mit den damaligen Fotosatz-Systemen sehr unzufrieden, weil sie mathematische Formeln nicht gut darstellen konnten. Das brachte ihn auf die Idee selbst ein Satzsystem zu entwickeln, das für Texte der Mathematik und Informatik typografisch hochwertige Ergebnisse erreichen sollte. Zunächst dachte er, ein solches Programm könne in einem Jahr oder so erstellt werden. Es dauerte dann doch etwa länger – siehe Tabelle **??**.[1]

Knuths Idee [3] bestand darin, dass Text, Formeln und Layout gewissermaßen programmiert werden. Das Satzsystem konsumiert ein solches „Programm“ und macht daraus ein Dokument — heutzutage in der Regel ein PDF-Dokument. Das System sollte natürlich erweiterbar und anpassbar sein, weshalb die Sprache, die Knuth entwickelt hat, im Grunde eine Makrosprache ist und es deshalb auch erlaubt eigene Befehle zu schreiben. Knuth hat einen Satz an Makros entwickelt, den man Plain TEX nennt – diese Makros sind sehr nahe am Kern von TEX, *low level* sozusagen.

Leslie Lamport[2] hat eine Menge von Makros entwickelt, die auf TEX aufbauen und das Setzen von Büchern, Berichten und Artikeln erheblich vereinfachen. Diese Makros nannte er LATEX [5]. Leslie Lamport erzählt über die Entstehung von LATEX (http://research.microsoft.com/en-us/um/people/lamport/pubs/pubs.html#latex):

> In the early 80s, I was planning to write the Great American Concurrency Book. I was a TEX user, so I would need a set of macros. I thought that, with a little extra effort, I could make my macros usable by others. Don Knuth had begun issuing early releases of the current version of TEX, and I figured I could write what would become its standard macro package. That was the beginning of LATEX.
> ...
> Meanwhile, I still haven't written the Great American Concurrency Book.

LATEX trennt im Grunde die Befehle, die den Inhalt und die Struktur eines Texts betreffen von den Befehlen, die für die typografische Gestaltung sorgen. LATEX stellt eine Reihe von sogenannten Dokumentklassen bereit, die das Layout von Artikeln, Berichten, Büchern und Briefen bereits enthalten, so dass man sich beim Schreiben nicht mehr darum kümmern muss.

Die Dokumentklassen von LATEX erzeugen eine Typografie, die an amerikanischen Konventionen orientiert ist. KOMA-Script ist eine Sammlung von Dokumentklassen, die sich an europäischer Typografie orientieren. Frank Neukam entwickelte Anfang der 1990er Jahr

[1] Die Entwicklung von TEX ist auch ein interessanter Fall von Software-Engineering, siehe Donald E. Knuth *The Errors of TEX* in: Tom de Marco und Timothy Lister (Hrsg.) *Software State-of-the-Art: Selected Papers* New York NY: Dorste Publishing House, 1990.

[2] Leslie Lamport hat wichtige Beiträge zur Theorie verteilter Systeme geleistet.

Script, das Markus Kohm zu KOMA-Script weiterentwickelte [4]. Wir verwenden für die Vorlage die Dokumentklasse von KOMA-Script für das Setzen von Büchern.

1.2 Der Aufbau einer LaTeX-Datei

Eine LaTeX-Datei beginnt mit der Angabe der Dokumentklasse mitsamt ihren Optionen. Darauf folgt die sogenannte Präambel, in der benötigte Pakete angegeben, Einstellungen festgelegt und auch eigene Makros definiert werden. Darauf folgt in der Umgebung `document` der eigentliche Text.

1.2.1 Dokumentklasse

Wir verwenden für die Abschlussarbeit die Dokumentklasse `scrbook` von KOMA-Script. Diese Dokumentklasse ist vorgesehen für den Satz von Büchern und sorgt dafür, dass Kapitel immer auf einer rechten (also vorderen) Seite beginnen.

Die Dokumentklasse hat in der Regel geeignete Voreinstellungen für eine Abschlussarbeit. Deshalb ändern wir nur einige wenige der Optionen von `scrbook`. In [6] werden weitere Optionen erläutert.

1.2.2 Präambel

In der Präambel binden wir benötigte Pakete ein. Pakete sind vorgefertigte Sammlungen von Makros für LaTeX. Es gibt für nahezu jede denkbare Aufgabe solche Pakete, die man im TeX Catalogue http://texcatalogue.ctan.org finden kann.

Unsere Vorlage bindet die wichtigsten und für eine Arbeit im Feld der Informatik in der Regel benötigten Pakete ein. Auch hierzu findet man weitere Informationen in [6].

Darüber hinaus stehen Einstellungen in der Präambel, bei uns z.B. zur Nummerierungstiefe.

Eigene Makros kann man auch in der Präambel unterbringen.

1.2.3 Die Umgebung `document`

Nach der Präambel kommt die Umgebung `document`. In LaTeX nennt man Blöcke, die durch `\begin{umgebung}` und `\end{umgebung}` begrenzt sind *Umgebung*. In der Regel werden zu Beginn der Umgebung bestimmte Einstellungen ein- und am Ende der Umgebung wieder ausgeschaltet.

Der eigentliche Text der Arbeit steht also in der Klammer

```
\begin{document}

% hierher kommt der eigentliche Text

\end{document}
```

Es ist ratsam, den eigentlichen Text in verschiedene Dateien zu verteilen, etwa pro Kapitel eine Datei. Diese Dateien können dann in die führende Datei eingebunden werden, bei uns durch folgende Befehle:

```
\input{aufbau}      % Kapitel 1
\input{gliederung}  % Kapitel 2
\input{elemente}    % Kapitel 3
```

Literaturverzeichnis

[1] Thomas Sigdestad *Front-end frameworks: What is important right now?*, https://enonic.com/blog/front-end-frameworks-what-is-important, (abgerufen 26.04.2023)

[2] Marco Daniel, Patrick Gundlach, Walter Schmidt, Jörg Knappen, Hubert Partl und Irene Hyna *LaTeX 2ε-Kurzbeschreibung*, http://mirror.unicorncloud.org/CTAN/info/lshort/german/l2kurz.pdf, 2015.

[3] Donald E. Knuth *The TeXbook*, Reading, MA: Addison-Wesley, 1986.

[4] Markus Kohm *Die Anleitung KOMA-Script*, http://www.komascript.de/~mkohm/scrguide.pdf 2016.

[5] Leslie Lamport *LaTeX: a document preparation system*, 2nd edition, Reading, MA: Addison-Wesley, 1994.

[6] Günter Partosch *Anforderungen an wissenschaftliche Abschlussarbeiten und wie sie mit LaTeX gelöst werden können*, https://www.staff.uni-giessen.de/partosch/unterlagen/abschlussarbeit.pdf, 2015.

[7] Max Mustermann *Bib Testen*, https://googl.de, 2023.