VueJS 101 / 1

SAE – 08.09.2020

Simon Auer

Willkommen

Kurze Vorstellung

- Wie heiße ich?
- Woher bin ich?
- Hobbies
- Haustiere
- Ausrichtung
- Motivation / Ziel
- Lieblings Hausarbeit

VueJS

Wie spricht man es aus?

- **wü** (“heast oida, wos **wü**st?)
- **wu** (“**Vou**lez-**vou**s coucher avec moi ce soir?”)
- **view** (“this window has an amazing **view**”)
  [siehe Model View ViewModel (MVVM)]

Die Antwort ist natürlich “view”!

VueJS

Geschichte von vuejs:

- 2014 von Evan You
- Ehemaliger Mitarbeiter bei Google verwendete AngularJS
- => Suche nach “more lightweight alternative to Angular”
- => VueJS

Was ist VueJS

- Javascript frontend framework
- MVVM pattern (Model–view–viewmodel)
- Teilung von grafischem Interface (View) und business logic (Model)
- (two-way) data-binding

MVVM

MODEL VIEW VIEWMODEL

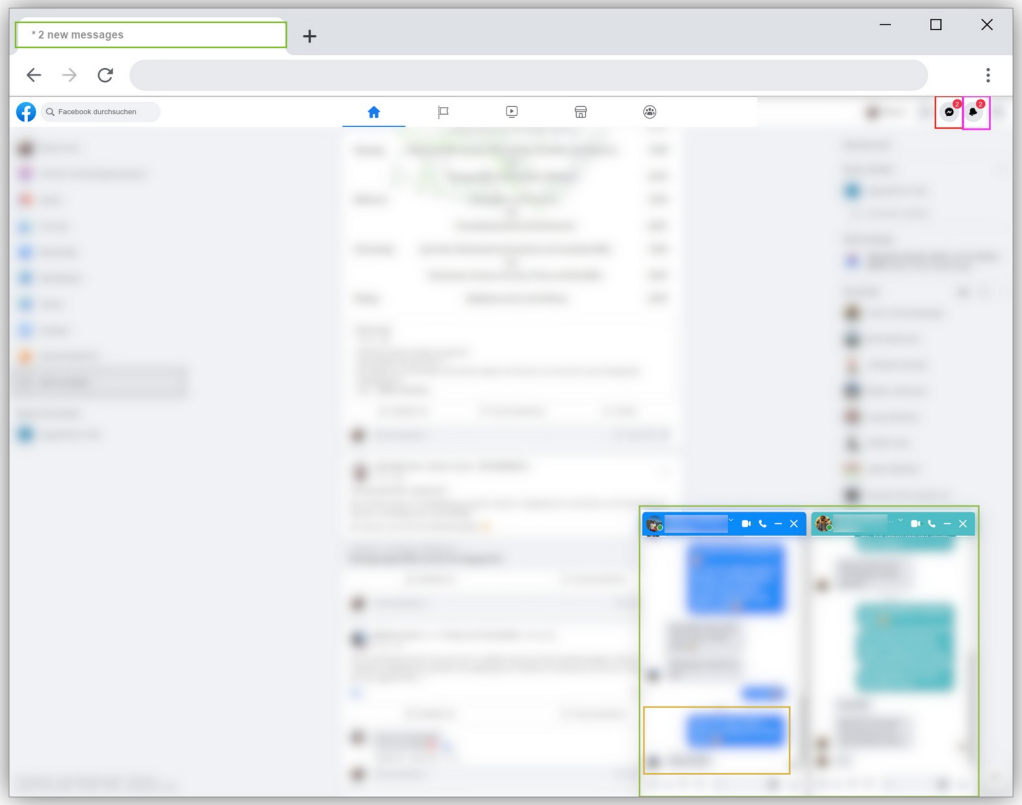
Evolution der Frontendentwicklung

- **Statische HTML Webseiten**
  - reine XML Files ohne dynamische Teile
- **Server side gerenderte Scripts/Applikationen**
  - ab Mitte 90er
  - z.Bsp: Perl, Php, Java, Python, Ruby (on Rails), …
- **The flash-y age**
  - ***bling bling***
- **Enhanced HTML** (AJAX – Asynchronous JavaScript and XML)
  - ab 2005
  - Dynamisches Nachladen einzelner Inhalte
- **Enhanced HTML Extreme** – JQuery
  - ab 2006
  - Vereinfachte Manipulation des DOMs AJAX
- **Frontends reinvented**
  - ab 2008
  - Mobile Versionen
  - Erste richtige Frontend Frameworks
  - Dojo, Backbone, Knockout, ...
- **Das Zeitalter der Single Page Applications**
  - ab 2013
  - Angular, Ember, React, Vue, ...

Warum moderne Frontend Frameworks gebraucht werden!

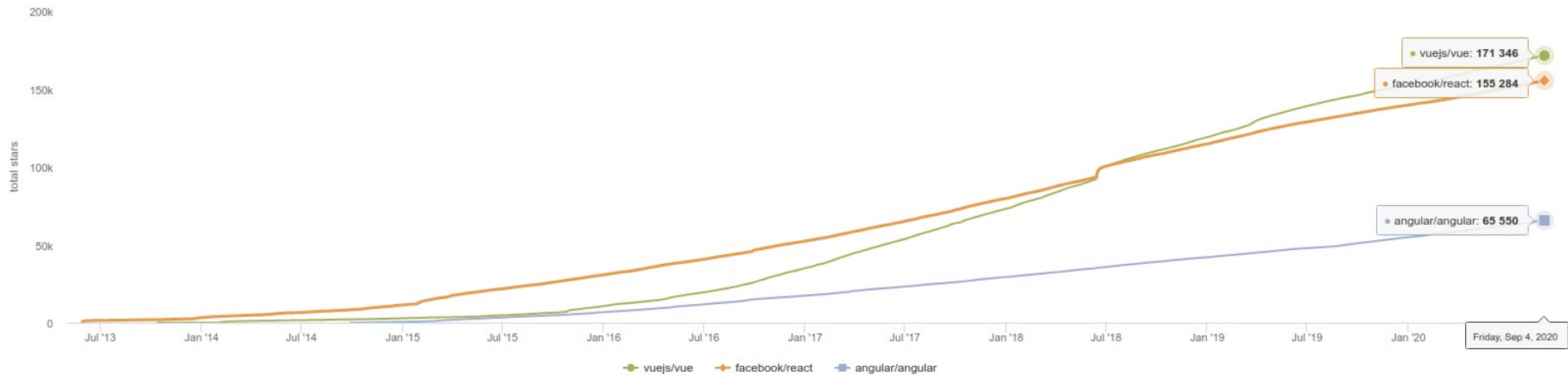
Moderne Applikationen erfordern viele zusammenhängende Updates des UIs ohne Reloads.

Siehe anbei – Facebook, wenn eine neue Nachricht eintrifft
=> 5 unterschiedliche Änderungen im User Interface bei einer neuen Nachricht



VueJS Verbreitung

- Vergleich zu Angular & React: noch nicht so mainstream
- Aber aktuell stark wachsend
- Aktuelle Einsatzgebiete:
  - Angular: mehr enterprise Projekte
  - React: mehr Startups (aber auch langsam enterprise)
  - VueJS: aktuell sehr Startup lastig, aber immer beliebter, da gut kombinierbar



1.) Anzahl github stars der “main repositories” am 06.09.2020

Unsere erste VueJs App

Legen wir mit unseren ersten kleinen Beispielen los:

- Clone https://github.com/SimonErich/sae-vuejs-101-0920
  - git clone https://github.com/SimonErich/sae-vuejs-101-0920 sae-vuejs
- Cd into dir
  - cd sae-vuejs/

VueJS einbinden

- beliebiges HTML Dokument (git - example0)
- Einbindung vuejs via CDN *

- **einfache Möglichkeit:**
  - Füge das vuejs Javascript direkt vor </head> ein:

```
<head>
    <title>What a wonderful view 0</title>

    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js" defer></script>
</head>
```

(* Nur für Entwicklung. Optimale Einbindung für Production Release in einer späteren Einheit)

VueJS intialisieren

- Next step: Neues VueJS Objekt intialisieren
- Nimmt ein Javascript Objekt mit Optionen als Parameter

```
// app.js
const app = new Vue({
    el: '#app' // initialisiere vue app in div container mit id “app”
    ....
});
```

Verbinde nun die app.js mit deiner HTML Datei.
Folgender Script Tag muss in die index.html (unter vuejs-Script Tag aus letztem Schritt)

```
    <script src="js/app.js" defer></script>
</head>
```

Unser erstes Vue Template (example1)

- “Variablen” werden im Vue Object unter “data” definiert.
- Wie in Laravel (blade) mit “{{“ “}}” dargestellt
  (nur ohne $-Zeichen)
- Wichtig: Sämtlicher Code, den wir hier schreiben MUSS innerhalb von “ …… ” sein!
  (denn nur das wird von VueJS verarbeitet) – siehe “el: ‘#app’” unten (hier wird das von uns festgelegt)

app.js

```
const app = new Vue({
    el: '#app',
    data: {
        firstname: 'Maxi',
        lastname: ‘Mustermann’,
    }
});
```

index.html:

```
    <div id="app">
        Hallo, ich bin {{firstname}} {{lastname}}
    </div>
```

Aufgabe I

- Binde die “Variable” favoriteFood im Template nach “Liebelingsessen: “ ein.
- Füge nun noch folgende zwei Variablen hinzu: Hobby (hobby) und Haustier (pet) und zeige sie im Satz an.
  (jeweils in einem eigenen)

Direktiven

- sind Kontrollfunktionen und spezielle Mechanismen in vue
- if, for, bindings, …
- Sind spezielle html-parameter, die mit “v-” starten

```
<div v-text='Direktive"></div>
```

Binding key: v-if, v-for, v-model, …

Binding Wert: je nach Direktive unterschiedliche Syntax (siehe folgende Slides)

If-direktive (v-if)

- Zeigt ein div an oder nicht, abhängig von der Bedingung.

```
// Beispiel für korrekte Bedingung
<div v-if="condition === true">Sichtbar</div>

// Beispiel für falsche Bedingung
<div v-if="condition !== true">Nicht sichtbar</div>

// Beispiel für "ODER" ("||") Verbindung von Bedingungen
<div v-if="wohnort == 'Wien' || plz < 2000">
    Du lebst in Wien!
</div>
```

If Bedingung (selbe Syntax wie bei js, php, ...)

If-direktive – Aufgabe I

- Verwende die if Direktive, um “Hobby” auszublenden, falls kein Hobby definiert wurde (die Variable leer - ‘’ - ist)
- Probiere die Variable “hobby” zu leeren und dann wieder zu füllen und sieh wie sich die Ausgabe verändert

If-direktive – Aufgabe II

- Erstelle eine neue Variable “Ort” (location) mit deinem Wohnort
- Zeige den Wohnort im view an
- Stelle den Wohnort so ein, dass er nur sichtbar ist, wenn dieser **weder** St. Pölten **noch** nicht Wien ist. (Es sollen also alle Orte außer Wien und St. Pölten angezeigt werden)

For-direktive (v-for)

- Wiederholt den Tag (hier ein “div”) für jedes item in “myListArr”.

```
// const myListArr = ['item 1', 'item 2', 'item 3'];

<div v-for="listItem in myListArr">{{ listItem }}</li>
```

kann auch mit Objekten verwendet werden:

```
// const myTodoList = [
//   { title: ‘Wäsche waschen’, description: ‘Waschen ist cool’ },
//   { title: ‘Staubsaugen’, description: ‘Ich & mein Dyson <3’ },
// ];
<ul>
  <li v-for="todoItem in myTodoList">
    <b>{{ todoItem.title }}</b><br />
    {{ todoItem.description }}
  </li>
</ul>
```

For-direktive – Aufgabe I (example2)

- Stelle dieses Array als Liste mit <ul> ... {{title}} … im view dar
- Zeige das “checkmark” Bild nur für todos mit dem Status “done”

Model-direktive (v-model)

- Verknüpft Eingabefelder mit Variablen (data-binding)
- Updated alle Vorkommen der Variable automatisch

```
<input type="text" v-model="theVariable" />
theVariable: {{theVariable}}
```

Model-direktive – Aufgabe I (example 3)

- Verwende die Variable “dogname” an allen Stellen anstatt “DOGNAME”
- Ändere den Standardwert von “dogname” auf einen tollen Hundenamen
- Verknüpfe das input Feld mit der Variable “dogname”
- Teste, ob Eingaben im Feld überall übernommen werden

Übungsaufgabe / Wiederholung – Einheit 1

1) Clone das git Repository neu (clean start) – siehe Slide 9
2) Binde vuejs ein und initialisiere es – siehe Slides 10 & 11
3) Füge nun die folgenden Variablen hinzu: (app.js)
    1) firstname
    2) lastname
    3) favoriteAnimal
4) Füge diese Variablen nun in die index.html statt FIRSTNAME, LASTNAME, … ein (siehe Slide 12)
5) Erstelle nun in vuejs – data (app.js) ein array 3 Hunden

```
const app = new Vue({
    el: '#app',
    data: {
        dogs: [
            { name: 'bello', color: 'braun' },
            { name: 'rufus', color: 'weiß' },
            { name: 'max', color: 'pink' },
        ]
    },
});
```

6) Erstelle eine unordered list (<ul>) mit v-for und wiederhole das <li> für jeden Hund und gib dabei den Namen und die Farbe des jeweiligen Hundes aus – siehe Slide 18
7) Füge ein v-if ein, um nur Hunde anzuzeigen, die **braun** sind. - siehe Slide 15