



JavaScript & Vue

# Inhalt

- Theorie
- Installation
- Praktisches
- After Work Vollsuff im HQ

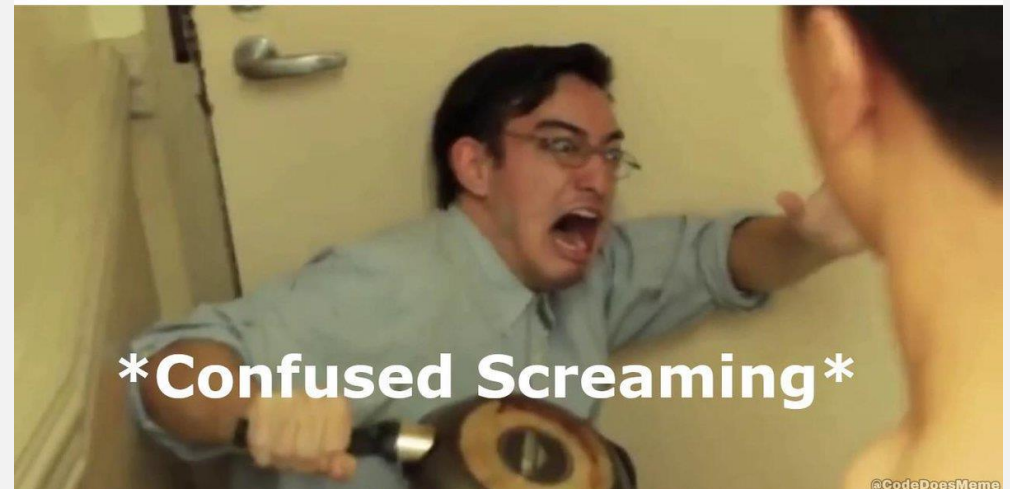


Theorie

# Was ist JavaScript (JS)

- Problem: HTML ist keine Programmiersprache
  - 1995 wollte man eine „Art“ Java im Browser haben
- Standardisiert unter dem Namen „ECMAScript“ (ES)
  - Aktuelle Version: ECMAScript 2019
  - Problem: Browserkompatibilität
  - Nutzbare Version: ES2015 (ES6)

Me: \*Tries to make website\*  
Internet Explorer:



# Wie sieht das aus?

```
<!DOCTYPE html>
<html>

<body>

  <h2>JavaScript in Body</h2>

  <p id="demo">A Paragraph.</p>

  <button type="button" onclick="myFunction()">Try it</button>

  <script>
    function myFunction() {
      document.getElementById("demo").innerHTML = "Paragraph changed.";
    }
  </script>

</body>

</html>
```

# JavaScript Eigenschaften

- Keine expliziten Typen !!!!11 (string, int, ...)
  - Alles ist nach außen ein Objekt, intern aber nicht
  - Alles kann verglichen werden: Funktionen, Klassen, Variablen
  - JavaScript konvertiert selbst intern so, dass es passt

```
> 20000 + "miles"
< "20000miles"
> true + "false"
< "truefalse"
> "R" + 2 + "D" + 2
< "R2D2"
>
```

undefined==false	false
null==false	false
undefined==0	false
null==0	false
undefined==''	false
null==''	false
false==''	true
''==0	true
false==0	true
null==undefined	true

# JavaScript nutzen

- If abfragen immer immer mit `===` oder `!==` nutzen
  - Typensicherer, da wenig „automatische“ Konvertierung stattfindet
- Variable deklarieren: `var meinText = „hallo“;`

# JSON

- Datenspeicherformat ähnlich zu XML
- Kann fast 1zu1 in JS genutzt werden

- `var variable = {...};`
- `Inhaber.Alter = 19;`
- `if(user[0].name === „Max“){`  
    `...`  
    `}`

```
{
  "Gruppenname": "Mitglieder",
  "user": [
    { "id": 1, "name": "Max" },
    { "id": 10, "name": "Mark" },
    { "id": 11, "name": "Manuel" },
    { "id": 11, "name": "Marcel" },
    { "id": 100, "name": "Moriz" }
  ]
}
```

```
{
  "Herausgeber": "Xema",
  "Nummer": "1234-5678-9012-3456",
  "Deckung": 2e+6,
  "Waehrung": "EURO",
  "Inhaber":
  {
    "Name": "Mustermann",
    "Vorname": "Max",
    "maennlich": true,
    "Hobbys": ["Reiten", "Golfen", "Lesen"],
    "Alter": 42,
    "Kinder": [],
    "Partner": null
  }
}
```



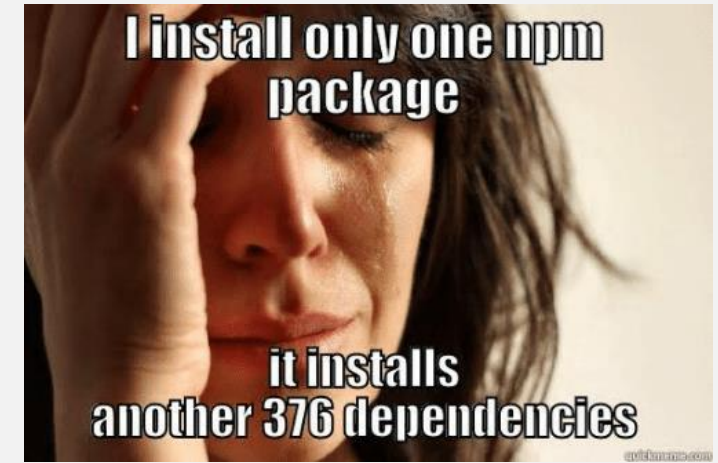
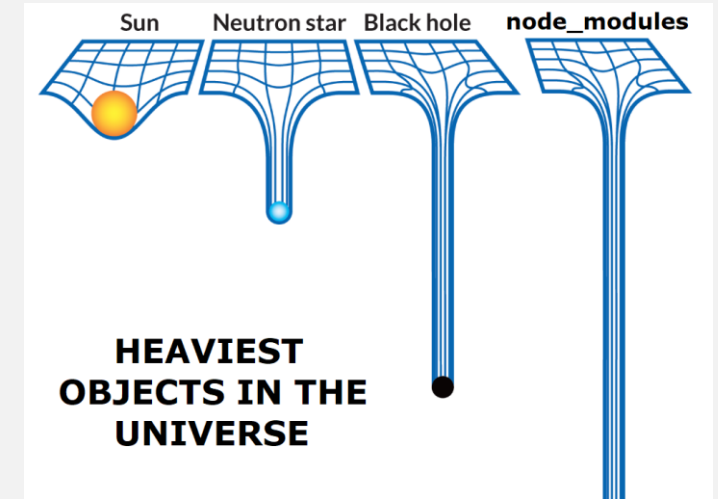
# NPM

- Das Maven der JavaScript Welt
  - nur in geil
- Befehl:  
`npm install <package-name>`
- Erzeugt eine package.json config (die nicht Kacke ist!)



# NPM

- Für viele nur eine andere Art Krebs
- Da alle Quelldateien wiederum JavaScript Dateien sein müssen eskaliert die Anzahl oftmals.
- Beispiel OPhIT
  - über 900 Pakete
  - Über 50.000 Dateien



# Und was soll Node.js sein?

- JavaScript als Server ohne Browser
- Chrome Engine als Basis im Hintergrund
  - Und damit fast auf jedem System lauffähig
- In Node.js läuft Vue
- Interessiert uns eigentlich nicht

# Vue

- Vue ist ein JavaScript „Framework“
  - In JS geschrieben und erzeugt JS, HTML und CSS Dateien
  - Erweitert HTML, sodass wir mehr abstrakten Code schreiben können
- Beispiel:
  - Irgendwo im Code haben wir die Variable `var user = { „name“: „max“ }`
  - Wir können nun einen HTML Button bauen der nur angezeigt wird, wenn dieser User existiert:

```
<button v-if=„user.name === ‚max‘“>  
  Click Me  
</button>
```

# Aufbau

HTML

JavaScript

CSS style code

```
<template>
  <div>Ganz normaler Text</div>
</template>

<script>
export default {
  data: {
    user: [
      { id: 1, name: "Max" },
      { id: 10, name: "Mark" },
      { id: 11, name: "Manuel" },
      { id: 11, name: "Marcel" },
      { id: 100, name: "Moriz" }
    ]
  },
  methods: {
    clickFunction() {
      console.log("click happend!");
      return "you clicked me";
    }
  }
};
</script>

<style>
div {
  margin: 8px;
}
</style>
```

# Aufbau II

Variablen

Methoden

```
<template>
  <div>Ganz normaler Text</div>
</template>

<script>
export default {
  data: {
    user: [
      { id: 1, name: "Max" },
      { id: 10, name: "Mark" },
      { id: 11, name: "Manuel" },
      { id: 11, name: "Marcel" },
      { id: 100, name: "Moriz" }
    ]
  },
  methods: {
    clickFunction() {
      console.log("click happend!");
      return "you clicked me";
    }
  }
};
</script>

<style>
div {
  margin: 8px;
}
</style>
```

# Loop (Array) in Vue

- Temporär wird eine variable erstellt, mit der dann in den Unterobjekten mit `{{tempVariable}}` zugegriffen werden kann `v-for=„tempVariable in Variable“`
- Unterscheidung der Objekte `:key=„identifier“`
- Hohzählender Index `v-for=„(user, i) in users“`

```
<template>
  <div>
    <div v-for="user in users" :key="user.id">
      Username: {{user.name}}
    </div>
  </div>
</template>

<script>
export default {
  data: {
    users: [
      { id: 1, name: "Max" },
      { id: 10, name: "Mark" },
      { id: 11, name: "Manuel" },
      { id: 11, name: "Marcel" },
      { id: 100, name: "Moriz" }
    ]
  },
  methods: {
    clickFunction() {
      console.log("click happend!");
      return "you clicked me";
    }
  }
};
</script>

<style>
div {
  margin: 8px;
}
</style>
```

# Events

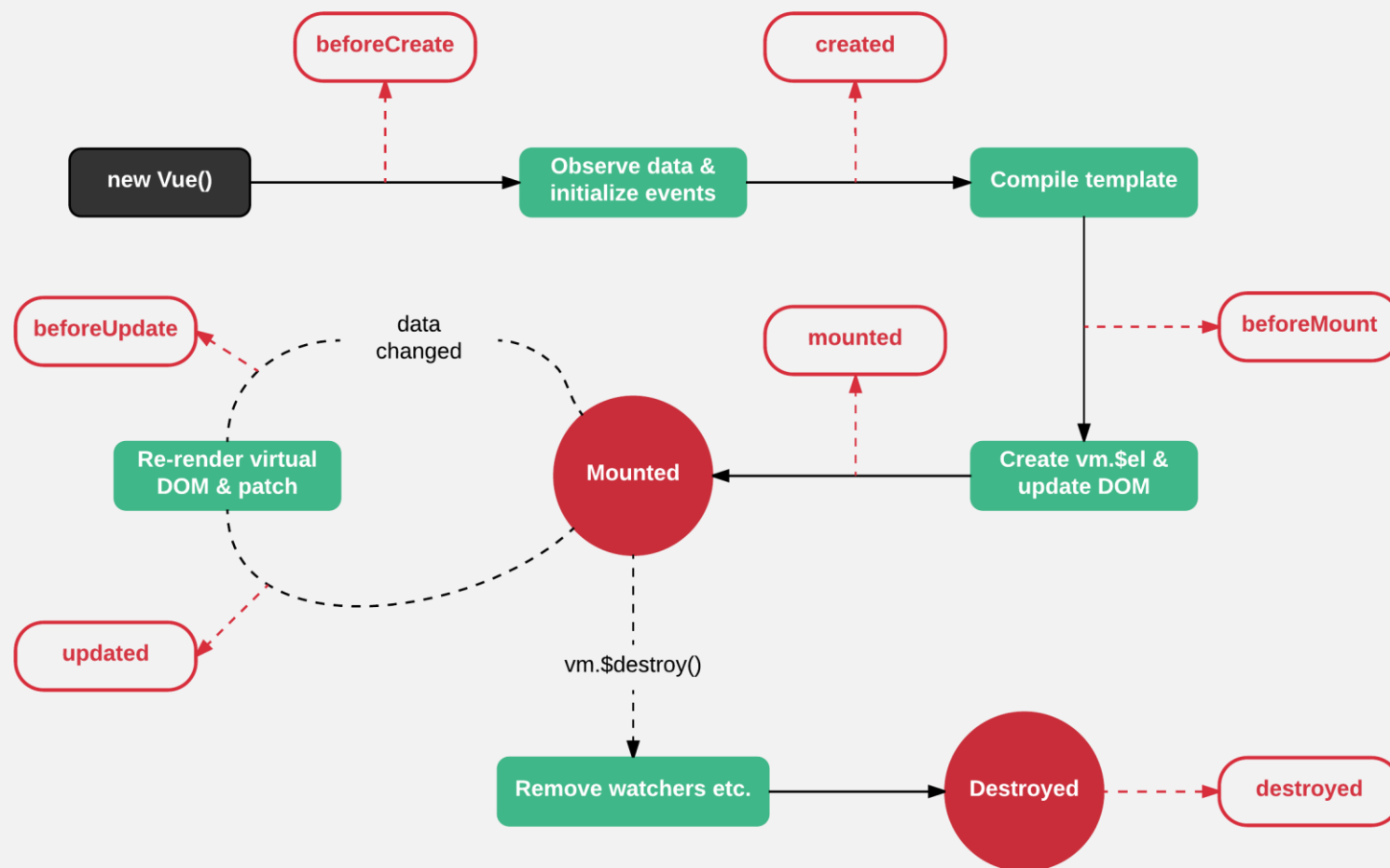
- @EventName=,,FunktionsName“

```
<template>
  <div>
    <button @click="clickFunction">Click me</a>
  </div>
</template>

<script>
export default {
  methods: {
    clickFunction() {
      console.log("click happend!");
      return "you clicked me";
    }
  }
};
</script>
```



# Lifecycle Events

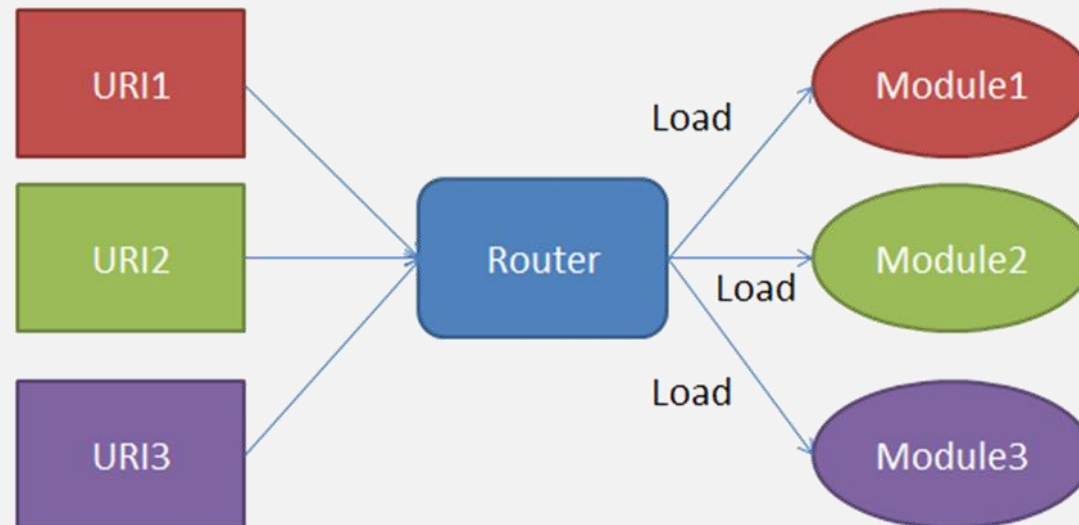


# Nur Theorie: Store

- Problem: Wenn die Seite neu geladen wird, sind alle Daten weg
- Lösung: Alles in Cookies speichern
- Ein Store verwaltet diesen Cookie-Speicher für euch
- Viele Unsicherheiten
  - Löschar
  - Nicht aktiv
  - Datenschutz des EuGH<sub>uso</sub>

# Nur Theorie: Router

- Problem: Wir können nicht unser ganzes Programm auf eine Seite klatschen
- Lösung: Wir nutzen einen Router, der für uns die Navigation übernimmt



# Nur Theorie: Code Splitting

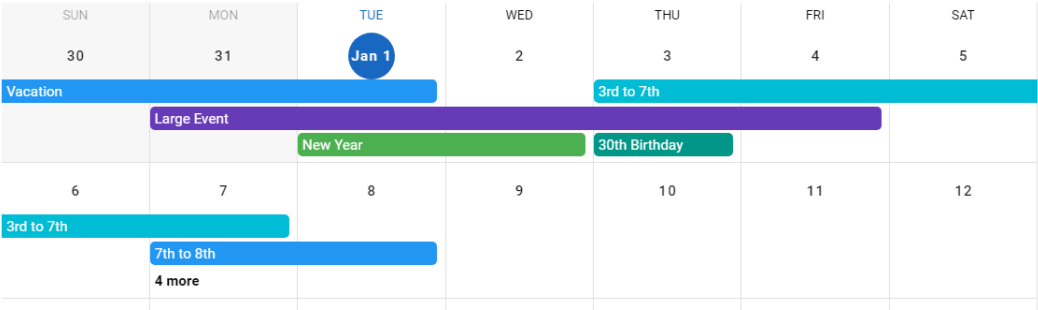
- Problem: Wenn wir große Webseiten schreiben, dann braucht unsere Seite zu lange, um zu laden
- Lösung: Nur den Code laden, den wir brauchen
  - Den Rest laden wir nach
- Passiert oft schon automatisch
- Führt dazu, dass viele kleine unleserliche Dateien erstellt werden

	chunk-c849f9f2.0670aa22.css	200	text/css	dashboard	(disk ca...	53 ms		
	chunk-cadd89f0.389f93e4.css	200	text/css	dashboard	(disk ca...	46 ms		
	chunk-ce71eb4e.0670aa22.css	200	text/css	dashboard	(disk ca...	63 ms		
	chunk-e0d25da6.3cac2c65.css	200	text/css	dashboard	(disk ca...	63 ms		
	chunk-e1f6287c.f7cecf9d.css	200	text/css	dashboard	(disk ca...	75 ms		
	chunk-f516c544.4cabaf1f.css	200	text/css	dashboard	(disk ca...	79 ms		
	chunk-04fcc3e8.50d37c4.js	200	javascri...	dashboard	(disk ca...	89 ms		
	chunk-0862df6c.5ce10d94.js	200	javascri...	dashboard	(disk ca...	122 ms		
	chunk-13002858.18ed6768.js	200	javascri...	dashboard	(disk ca...	103 ms		
	chunk-200df3b1.72cdb960.js	200	javascri...	dashboard	(disk ca...	230 ms		
	chunk-24b2b7e2.81926997.js	200	javascri...	dashboard	(disk ca...	116 ms		
	chunk-2c23259b.14933dab.js	200	javascri...	dashboard	(disk ca...	116 ms		
	chunk-2d0738d6.ead5694c.js	200	javascri...	dashboard	(disk ca...	228 ms		
	chunk-2d0c1517.78a80ca8.js	200	javascri...	dashboard	(disk ca...	118 ms		
	chunk-2d224ca0.42135bec.js	200	javascri...	dashboard	(disk ca...	117 ms		
	chunk-2ed48cf7.a022b3c1.js	200	javascri...	dashboard	(disk ca...	111 ms		
	chunk-30de4778.ed9a30c4.js	200	javascri...	dashboard	(disk ca...	111 ms		
	KFOICnqEu92Fr1MmSU5f8Bc4...	200	font	dashboard	(memo...	0 ms		
	materialdesignicons-webfont....	200	font	dashboard	(memo...	0 ms		
	chunk-312619c5.3dcf9c7d.js	200	javascri...	dashboard	(disk ca...	95 ms		
	chunk-313100bd.bafc2930.js	200	javascri...	dashboard	(disk ca...	187 ms		
	chunk-33900897.2d0f2808.js	200	javascri...	dashboard	(disk ca...	94 ms		
	chunk-33bb3e05.498c33bd.js	200	javascri...	dashboard	(disk ca...	95 ms		
	chunk-33c899d4.186ad993.js	200	javascri...	dashboard	(disk ca...	95 ms		
	chunk-33de3fce.a4c1fe7f.js	200	javascri...	dashboard	(disk ca...	95 ms		
	chunk-3f2945ec.e8a2a01e.js	200	javascri...	dashboard	(disk ca...	94 ms		
	chunk-4f8783c8.09032e52.js	200	javascri...	dashboard	(disk ca...	95 ms		
	chunk-4febb969.379fb3bb.js	200	javascri...	dashboard	(disk ca...	94 ms		
	chunk-5e052c30.daed3f19.js	200	javascri...	dashboard	(disk ca...	175 ms		
	chunk-5f3ceb5a.ea35025f.js	200	javascri...	dashboard	(disk ca...	97 ms		
	chunk-607fbb10.30086001.js	200	javascri...	dashboard	(disk ca...	98 ms		
	chunk-63148ef0.038eda1b.js	200	javascri...	dashboard	(disk ca...	145 ms		
	chunk-6bff4d4c.3f7e75c9.js	200	javascri...	dashboard	(disk ca...	144 ms		
	chunk-7017beca.6d6a3f56.js	200	javascri...	dashboard	(disk ca...	123 ms		
	chunk-77728fd1.66c31241.js	200	javascri...	dashboard	(disk ca...	124 ms		
	chunk-7a22daac.8edb412b.js	200	javascri...	dashboard	(disk ca...	125 ms		
	chunk-c41a20d6.56a06f91.js	200	javascri...	dashboard	(disk ca...	125 ms		
	chunk-c459b8d6.3819abce.js	200	javascri...	dashboard	(disk ca...	126 ms		
	chunk-c849f9f2.96e2c703.js	200	javascri...	dashboard	(disk ca...	126 ms		
	chunk-cadd89f0.172a98d0.js	200	javascri...	dashboard	(disk ca...	126 ms		
	chunk-ce71eb4e.a66804dc.js	200	javascri...	dashboard	(disk ca...	127 ms		
	chunk-e0d25da6.42252e24.js	200	javascri...	dashboard	(disk ca...	127 ms		
	chunk-e1f6287c.4a7cf7c5.js	200	javascri...	dashboard	(disk ca...	128 ms		
	chunk-f516c544.3c0c3b5e.js	200	javascri...	dashboard	(disk ca...	127 ms		
	chunk-313100bd.0670aa22.css	200	stylesh...	app.17630fdb.j...	(prefet...	3 ms		
	chunk-313100bd.bafc2930.js	200	script	app.17630fdb.j...	(prefet...	3 ms		
	logov4_dark.3bcc03ca.png	200	png	chunk-vendors...	(memo...	0 ms		
	favicon.png	200	png	Other		96.8 KB 129 ms		

78 requests | 98.5 KB transferred | 749 KB resources | Finish: 1.33 s | DOMContentLoaded: 982 ms | Load: 1.14 s

# Vuetify

- Noch ein JavaScript Framework
- Baut auf zwingend auf Vue auf
- Dokumentation: <https://vuetifyjs.com/en/>
  - Gar nicht mal so schlecht



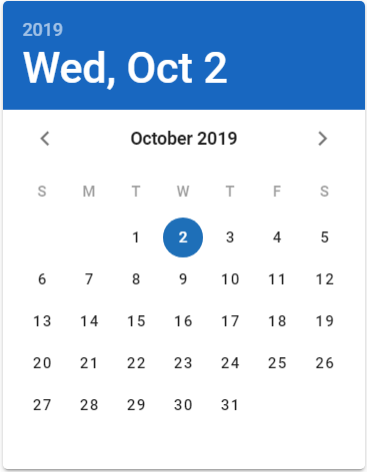
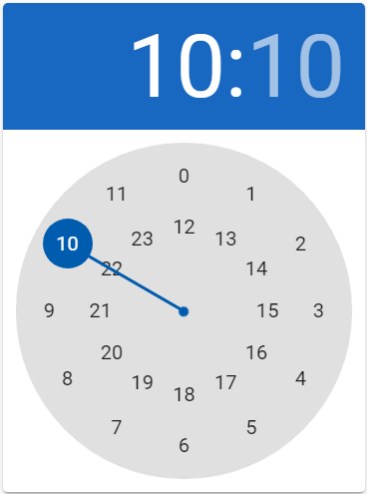
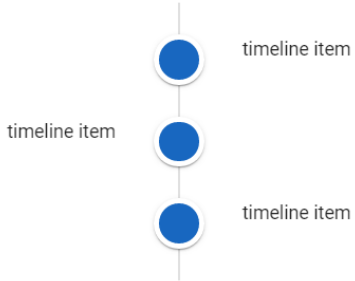
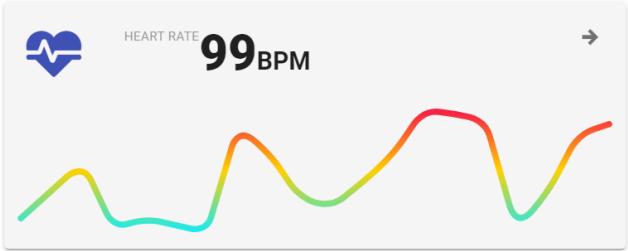
☰

Page title

🔍 ⋮

ITEM 1ITEM 2ITEM 3

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



ⓘ

I'm a dense alert with a **type** of info

I'm a dense alert with the **text** prop and a **type** of success

! I'm a dense alert with the **border** prop and a **type** of warning

⚠ I'm a dense alert with the **outlined** prop and a **type** of error

I'm a title

I'm card text

CLICK

Select Country

☐ Bahamas, The

☐ Bahrain

☐ Bangladesh

☐ Barbados

☐ Belarus

☐ Belgium

☐ Belize

☐ Benin

☐ Bhutan

CLOSE SAVE

Application

subtext

🏠

Dashboard

🖼️

Photos

?

About

NORMAL

PRIMARY

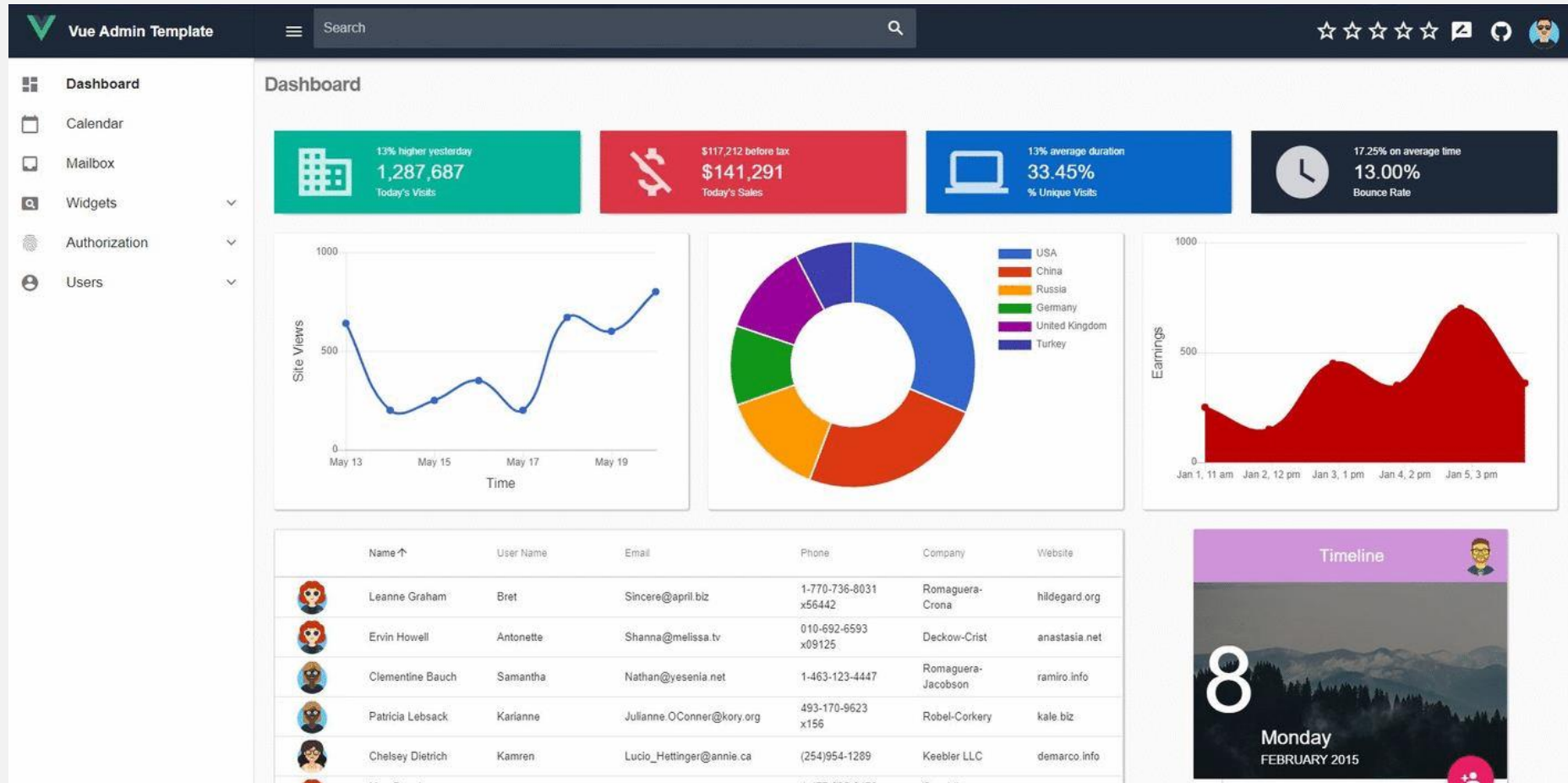
ERROR

DISABLED

Dessert (100g serving)	Calories	Fat (g)	Carbs (g)	Protein (g)	Iron (%)
Frozen Yogurt	159	6	24	4	1%
Ice cream sandwich	237	9	37	4.3	1%
Eclair	262	16	23	6	7%
Cupcake	305	3.7	67	4.3	8%
Gingerbread	356	16	49	3.9	16%

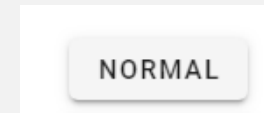
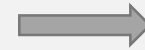
Rows per page: 5 1-5 of 10 < >

# Beispiel

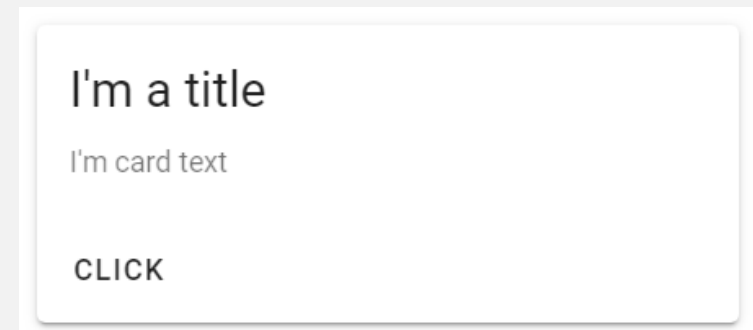
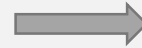


# Nutzung von Vuetify

`<v-btn>Normal</v-btn>`



```
<v-card>
  <v-card-title>I'm a title</v-card-title>
  <v-card-text>I'm card text</v-card-text>
  <v-card-actions>
    <v-btn text>Click</v-btn>
  </v-card-actions>
</v-card>
```







# Installation

# Installation

- Das solltet ihr installiert haben:
  - Visual Studio Code (VSC)
  - Node.js
- Ablauf
  - Konsole öffnen (cmd)
  - Befehl zum installieren: `npm install -g @vue/cli`
  - Befehl zum starten: `vue ui`
  - (Bonus) Plugins *Prettier* und *Vetur* in VSC installieren
  - (Bonus) Chrome Plugin *Vue Devtools* installieren

# Installation Teil II

- Projekt anlegen in Vue
  - Tab „Neu“
  - Ordner auswählen
  - Name: Egal
  - Voreinstellung: Standard
- Vuetify installieren
  - Plugins -> Plugin hinzufügen -> „vue-cli-plugin-vuetify“
  - Preset: Default
  - Am Ende die Änderungen mit einer Nachricht akzeptieren (weil GIT)
  - Lädt im Hintergrund eigentlich ein npm package

# Installation Teil III

- Projekt starten
  - Aufgabe -> serve -> Aufgabe Starten
  - Warten -> „App öffnen“-Button klicken

*Fertig*

# Code angucken

- Öffnet den Code in VSC und guckt euch die App.vue Datei an.
- Verändert den Code. Die Webseite ändert sich sofort.
  - Das nennt man Hot-Module-Replacement (HMR)
- Öffnet mit einem Rechtsklick die Entwicklereinstellungen in eurem Browser. Guckt mal, was Vue für einen unleserlichen Mist zusammengestellt hat.
  - Viele unnötige und wiederholende Abschnitte
  - Code selbst schreiben ist immer besser
  - Aber eigentlich immer unmöglich



# Praktisches Beispiel

# Praktisches Beispiel

- Wir bauen ein ~~eine Bierpong-App~~ User-Verwaltungs-Seite



Abschließende Worte



# Abschließende Worte

- JavaScript ist eine Art „C“ für die Web-Welt
  - Niemand nutzt die reine Low-Level Sprache, weil sie nervt
  - Alle Nutzen aber Tools, die am Ende JavaScript Code produzieren
- Vue ist erfolgreich geworden, weil es nicht so kompliziert wie **Angular** und mächtiger als **React** ist.
  - Angular (Google) und React (Facebook) werden aber viel häufiger genutzt
- Zukünftig werden alle Apps Progressive Web Apps (PWA) sein
  - JavaScript kann alles, was man braucht
  - Auch viele Hardware-Zugriffe (zB. Kamera) sind möglich
  - Performance oft schlecht, aber der Entwicklungsaufwand ist kleiner
  - Apples Safari ist der neue Internet-Explorer, weil er auf viele Standards schießt
- Einzige Server- und Frontend- Sprache die überall läuft

# Dateien

- GIT Repository mit allen Daten:  
<https://github.com/LightSnowDev/JavaScriptVueTutorial->



Danke für's kommen.



Ab in's HQ für n' Bier!