

# **webmap - Automatische Generierung von Websites zur Interaktiven Datenveranschaulichung**

## **DIPLOMARBEIT**

verfasst im Rahmen der

**Reife- und Diplomprüfung**

an der

**Höhere Lehranstalt für Informationstechnologie,  
Ausbildungsschwerpunkt Medientechnik**

Eingereicht von:

Jonas Dorfinger  
Sebastian scholl

Betreuer:

Dietmar Steiner

Projektpartner:

Christopher Stelzmüller, triply GmbH

Leonding, April 2022

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

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

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2022

J. Dorfinger & S. Scholl

Zur Verbesserung der Lesbarkeit wurde in diesem Dokument auf eine geschlechtsneutrale Ausdrucksweise verzichtet. Alle verwendeten Formulierungen richten sich jedoch an alle Geschlechter.

# Abstract

Brief summary of our amazing work. In English.

This is the only time we have to include a picture within the text. The picture should somehow represent your thesis. This is untypical for scientific work but required by the powers that are. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.



# Zusammenfassung

Zusammenfassung unserer genialen Arbeit. Auf Deutsch. Das ist das einzige Mal, dass eine Grafik in den Textfluss eingebunden wird. Die gewählte Grafik soll irgendwie eure Arbeit repräsentieren.

Das ist ungewöhnlich für eine wissenschaftliche Arbeit aber eine Anforderung der Obrigkeit. *Bitte auf keinen Fall mit der Zusammenfassung verwechseln, die den Abschluss der Arbeit bildet!* Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.



# Inhaltsverzeichnis

<b>1 Einleitung</b>	<b>1</b>
1.1 Kurzbeschreibung . . . . .	1
1.2 Aufgabenstellung . . . . .	1
1.3 Zielsetzung . . . . .	1
1.4 Geplantes Ergebnis . . . . .	1
<b>2 Technologien</b>	<b>2</b>
2.1 Versionierung . . . . .	2
2.2 Projektkoordination . . . . .	5
2.3 Frontend Framework/Library . . . . .	7
2.4 Map Frameworks . . . . .	16
2.5 Static Site Generators . . . . .	16
2.6 CI/CD Pipeline . . . . .	16
2.7 Backend . . . . .	16
2.8 Webserver . . . . .	16
2.9 Reverse Proxy . . . . .	16
2.10 Containerization . . . . .	16
<b>3 Umsetzung</b>	<b>17</b>
3.1 Frontend Implementierung . . . . .	17
<b>4 Evaluation des Projektverlaufs</b>	<b>18</b>
<b>Literaturverzeichnis</b>	<b>VI</b>
<b>Abbildungsverzeichnis</b>	<b>VIII</b>
<b>Tabellenverzeichnis</b>	<b>IX</b>
<b>Quellcodeverzeichnis</b>	<b>X</b>
	<b>III</b>



# **1 Einleitung**

## **1.1 Kurzbeschreibung**

Triply entwickelt hochwertige Softwarelösungen, die dabei helfen, bestehende Mobilitäts-situationen (Verkehrsanalysen, Besucherströme) zu verstehen. Eine der Kernfunktionen liegt darin, Ergebnisse der Analysen einfach und verständlich darzustellen. Dafür soll eine Software geschaffen werden, die interaktive Kunden-Demos einfach und schnell erzeugen kann.

## **1.2 Aufgabenstellung**

Bei dem Diplomarbeitsprojekt Webmap handelt es sich um die Implementierung eines Generators, welcher interaktive Webseiten automatisch erstellt. Für die erwartete Benutzung ist es erforderlich, dass die ganze Software vollständig im Webbrower und somit plattformunabhängig funktionieren wird.

## **1.3 Zielsetzung**

Mitarbeitern von triply soll es mit der Datenvisualisierungs-Pipeline möglich sein, schnell und einfach interaktive Websites auf der Basis von komplexen Datensätzen zu erstellen, um diese potentiellen Kunden anschaulich zu präsentieren.

## **1.4 Geplantes Ergebnis**

Entwicklung eines effizienten Generators für interaktive Datenveranschaulichung. Weiters ist eines unserer Ziele die Benutzung der Software so leicht und intuitiv wie möglich zu gestalten, zusätzlich dazu soll der gesamte Prozess vom Starten der Konfiguration bis zur laufenden Website möglichst wenig Zeit in Anspruch nehmen.

# **2 Technologien**

## **2.1 Versionierung**

Der Begriff Versionierung beschreibt im Allgemeinen einen Prozess welcher zur Dokumentation von Änderungen an Dokumenten oder Dateien stattfindet. Nach jeder Änderung wird im System eine aktuelle Version abgespeichert und mit einer eindeutigen Versionsnummer versehen. Die Versionsnummer wird meist automatisch durch das Versionierungstool vergeben. Zusätzlich wird auch gespeichert, wer welche Einträge wann und wo gemacht hat, so werden Änderungen transparent mitprotokolliert. Eine der wichtigsten Vorteile einer Versionierung ist jedoch die Funktion, dass man alte Stände von Dateien wiederherstellen kann. Eine Versionsverwaltungssoftware kann auch die gleichzeitigen Zugriffe auf eine Datei von mehrere Personen koordinieren. Versionierung ist nicht nur in der Softwareentwicklungs Branche Standard, sondern auch in etlichen weiteren Bereichen stark verbreitet.

### **2.1.1 automatische vs. manuelle Versionierung**

Wenn während dem Schreiben einer Arbeit Sicherheitskopien anlegt und diese mit zum Beispiel "arbeit-neu", "arbeit-fertig" oder "arbeit-fertig2" benannt werden, dann ist das nichts anderes als eine manuelle Versionierung, weil auch hier eine eindeutige Versionsnummer für jede Datei vergeben wurde. Das birgt aber auch große Risiken, da eine manuelle Versionierung viel Zeit und Disziplin erfordert und immer die Gefahr besteht, dass man eine wichtige Version überschreibt. Die meisten Probleme werden von der automatischen Versionierung gelöst. Bei dieser Variante werden automatisch, wenn man einen Stand als "fertig" markiert, diese mit einer Versionsnummer versehen und im Archiv schreibgeschützt abgelegt. Dadurch hat man die Risiken der manuellen Versionierung auf ein Minimum reduziert [1].

## 2.1.2 git

git ist *das* Versionierungssystem in der Softwareentwicklung. Es wurde designt um kleine aber auch extrem große und komplexe Projekte effizient zu verwalten. Grundsätzlich als CLI (Command Line Interface) konzeptioniert, gibt es mittlerweile auch etliche Grafische Implementierungen, um git interaktiver und intuitiver zu gestalten. Laut einer Umfrage von Stackoverflow nutzen mehr als 87% der Entwickler weltweit git [2].

### Der git Workflow

Der git Workflow beschreibt die effizienteste Arbeitsweise mit git. Dabei gibt es vier Stufen zwischen denen mit verschiedenen Konsolen Befehlen gewechselt werden kann.

**1. Remote Repository** Das Remote Repository befindet sich auf einem externen Server (zum Beispiel Github Server 2.1.4). Mit einem einfache *git clone* Befehl in der Konsole, kann man das Repository klonen und somit ein lokales Repository erstellen. Auf dieses Archiv können mehrere Personen zugreifen und so gemeinsam an einem Projekt arbeiten.

**2. Lokales Repository** Das Lokale Repository ist eine Kopie der Version im Remote Repository, mit dem Unterschied, dass die lokalen Änderungen in diesem Repository gesammelt werden und erst durch den Befehl *git push* wieder zurück ins Remote Repository gelangen.

**3. Staging Area** In der Staging Area sind alle lokal modifizierten Files, welche für den nächsten Commit auserwählt sind, gesammelt. Durch den Befehl *git commit -m <message>* gelangen ausgewählte Änderungen in des lokale Repository. Ein Commit repräsentiert dabei eine Version des Programmes, welche mit einer eindeutigen Nummer im System gespeichert wird.

**4. Working Directory** Im Working Directory wird tatsächlich programmiert, alle Änderungen der Files werden hier durchgeführt. Mit dem Befehl *git add <filename>* können einzelne Files gezielt in die Staging Area verschoben werden um diese dann anschließend ins lokale Repository zu committen.

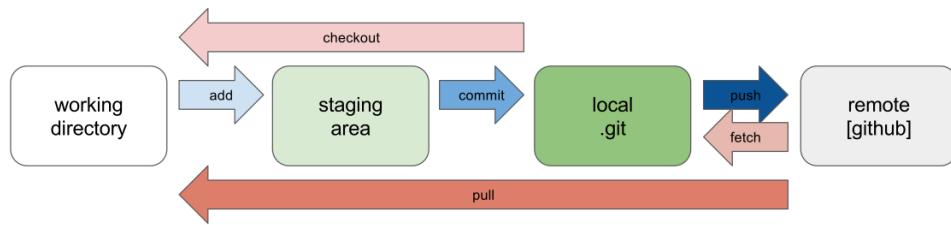


Abbildung 1: git workflow [3]

### 2.1.3 Semantische Versionierung

Eine semantische Versionsnummer besteht aus drei Teilen:

MAJOR.MINOR.PATCH

Und sollen nach Änderungen in der entsprechenden Kategorie in einser Schritten erhöht werden.

**MAJOR** Wenn API Änderungen durchgeführt werden, welche die API mit der Vorgängerversion inkompatibel machen

**MINOR** Wenn Funktionalität hinzugefügt wird, welche mit älteren Versionen noch kompatibel sind

**PATCH** Wenn kleine Bug fixes durchgeführt werden, welche auch mit älteren Versionen kompatibel sind

Zusätzlich können auch labels für *pre-release* oder *build metadata* angegeben werden:

MAJOR.MINOR.PATCH format

[4]

### 2.1.4 GitHub

GitHub Inc. ist ein amerikanisches gewinnorientiertes Softwareunternehmen, welches git Repositories in der Cloud anbietet, also das Remote Repository. Dadurch können Privatpersonen aber auch große Unternehmen und Konzerne ganz leicht git als Versionierungstool verwenden. Durch viele Features von GitHub wird das arbeiten mit git so stark erleichtert, dass auch Anfänger bereits sehr professionell mit git arbeiten und

lernen können. Normalerweise verwendet man git mit der Kommandozeile des jeweiligen Betriebssystems, was hohes technisches Verständnis voraussetzt. Man kann einen kostenlosen Account erstellen und gratis Repositories anlegen und verwenden, deshalb ist Github in der Open-Source Community sehr stark verbreitet. Doch mittlerweile kann diese Software mehr als "nur" Projekte zu versionieren. Es gibt project-tracking-tools (GitHub Projects, Issues, Milestones, Labels, etc.), Pipelines für Continues Integration and Delivery, gratis Website Hosting und noch vieles mehr [5].

Es gibt aber auch andere Programme welche zu GitHub gehören, zum einen ist es *Atom*, ein kostenloser und Open-Source Editor für Entwickler. Weiters gibt es zum Beispiel auch *Electron*, das ist ebenfalls ein Open-Source JavaScript Framework um Web-Anwendungen zu Desktopanwendungen zu machen.

Unternehmen können kostenpflichtige Organisationen anlegen, um alle Repositories einer Firma gesammelt an einem Ort zu verwalten. Dazu gibt es wieder etliche Features welche die Sicherheit, Rollen und vieles mehr betreffen.

Seit 2018 gehört GitHub zu Microsoft, welche für die Übernahme \$7,5 Milliarden US-Dollar auf den Tisch gelegt haben [16].

## 2.2 Projektkoordination

Da Triply über eine GitHub Organisation verfügt, ist es naheliegend, dass für webmap ein eigenes Repository angelegt wird.

Um Übersicht über den Projektverlauf zu behalten, ist die Entscheidung aufgrund von internen Vereinbarungen von triply sowie die Erfahrung des Teams auf GitHub Projects gefallen. Man kann ein Project Board ganz einfach über die GitHub Website erstellen und dabei auswählen, ob eine Vorlage ausgewählt werden soll.

Es gibt folgende Templates zur Auswahl: [?]

**None** Es wird ein völlig leeres Project Board erstellt, alle Spalten und Einstellungen müssen manuell gemacht werden.

**Basic Kanban** Ein Project Board mit den Spalten *To do*, *In progress* und *Done* wird erstellt.

**Automated Kanban** Es wird ein Basic Kanban Board erstellt, mit der zusätzlichen Funktion von eingebauten Triggern, welche automatisch Issues und Pull Requests zwischen den Spalten hin und her schieben.

**Automated Kanban with Reviews** Das Template *Automated Kanban with Reviews* erweitert das Automated Kanban Template um 2 weitere Spalten und einen Trigger , für die Pull Request Reviews.

**Bug triage** Dieses Template wird verwendet, um Bugs zu priorisieren und diese geordnet anzusehen, dabei gibt es folgende Spalten:

- To do
- High priority
- Low priority
- Closed

### 2.2.1 Issues

Issues haben die Möglichkeit, die anstehende Arbeit zu dokumentieren und planen.

[?]

### 2.2.2 Automatisierung

Durch die Verwendung spezieller Keywords in Commit messages oder in einem Kommentar bei einem Issue, kann der Zustand dieses Issues verändert werden. Dies geht einher mit der Automatisierung von Project Boards.

#### Keywords:

- close
- closes
- closed
- fix
- fixes

- fixed
- resolve
- resolves
- resolved

## Verwendung

<Keyword> <Issue Nummer> <Commit Message>

[?]

## 2.3 Frontend Framework/Library

Webmap ist als Webanwendung konzeptioniert, deshalb ist es wichtig für das Frontend eine Technologie zu wählen, welche nicht nur langlebig ist, sondern auch eine gute Übersicht bietet. Das beinhaltet unter anderem, dass eine große Community hinter dem Framework oder der Library steht, aber auch die Anforderungen des Auftraggebers dürfen nicht unbeachtet bleiben. Anhand dieser definierten Kriterien wurden die in 2021 beliebtesten Frameworks beziehungsweise Libraries verglichen und für eines entschieden.

### 2.3.1 Framework vs. Library

Sowohl Frameworks als auch Libraries sind Codes, welche oft auftretende Probleme einfach lösen sollen. Um das konkret zu veranschaulichen, kann man dies mit einer Metapher besser beschreiben.

Eine Library ist wie der Besuch in einem Möbelhaus, man hat zwar grundsätzlich ein Haus aber keine Einrichtung, man braucht also eine leichte Lösung um das Haus zu möblieren. Einen Tisch selbst zu zimmern wäre zu aufwendig, deswegen wählt man einen Tisch aus der großen Auswahl in dem besuchten Geschäft.

Das Verwenden eines Frameworks ist wie der Bau eines Hauses, man hat ein paar Blaupausen und Vorgaben zum Design und zur Architektur des neuen Gebäudes. Man bekommt ein Grundgerüst zur Verfügung gestellt und darin kann man den Regelkonform frei agieren.

Zusammengefasst sind Frameworks Gerüste, welche durch Entwickler projektspezifisch erweitert werden müssen und Libraries fertige Pakete, mit denen Programmierer Anwendungen um spezielle Funktionen erweitern.

### 2.3.2 Angular

Angular wurde 2016 von Google ins Leben gerufen, seitdem ist es ein beliebtes Framework für die Webentwicklung. Nur auf TypeScript basierend füllt es die Lücke zwischen der wachsenden Nachfrage von Technologien und traditionellen Ideen zur Performance Optimierung. Die Entwicklerplattform bietet etliche Services und Features an, dazu gehören nicht nur der Cross-Platform Support, sondern auch eine built-in Geschwindigkeits- und Performanceoptimierung. Auch für Unternehmen mit sehr hohen Nutzerzahlen ist Angular ein gern gewähltes Tool. Ein großer Vorteil von Angular ist zudem, dass Angular unter einer Open-Source-Lizenz veröffentlicht ist und somit nicht von dem Angular-Google Team, sondern auch von der Community betreut und erweitert wird. Angular hat mit dem Two-Way-Binding einen großen Vorteil gegenüber React 2.3.3. Dieses Feature stellt sicher, dass die View und das Model in Echtzeit miteinander synchronisiert werden. Das bedeutet, dass eine Änderung im Model direkt in der View angezeigt wird und umgekehrt auch. [6, 7]

#### Vorteile

- Features und Funktionalität wie das Two-Way-Binding sind direkt inkludiert
- Direkt eingebaute Features zum Updaten der View oder des Models.
- Komfortable Wiederverwendung von Components durch Dependency Injection
- Große Community zum Lernen und zur Fehlersuche

#### Nachteile

- Schwieriger zu lernen, da Angular so groß ist und es viele mögliche Lösungen gibt
- Durch die riesigen und komplexen Strukturen kann es sein, dass bei großen dynamischen Applikationen Performance Probleme auftreten
- Angular ist ein full-package Framework, weshalb es für kleine Applikationen ungeeignet ist

### 2.3.3 React

ReactJS ist die größte Konkurrenz von Angular, mit einem Vielfachen an Marktanteilen. Im Gegensatz zu Angular wird React von Facebook gewartet und betreut und verfügt über die Funktionalität des virtuellen Document Object Models (DOM). Das virtuelle DOM verbessert die Performance und verringert die Dauer von Render Prozessen, da nur die Inhalte verwendet werden, welche sich auch tatsächlich verändert haben und nicht die gesamte View neu gerendert wird. Durch dieses Feature wird eine Plattform geschaffen, welche sich sehr gut für Applikationen mit großem Traffic eignet. Ein Alleinstellungsmerkmal ist dazu auch noch, dass React Server-Side Rendering unterstützt. Für SPAs, also Single Page Applications ist es empfehlenswert sich für React zu entscheiden, durch das Wiederverwenden der Komponenten kann man in kurzer Zeit gute und interaktive User Interfaces entwickeln.

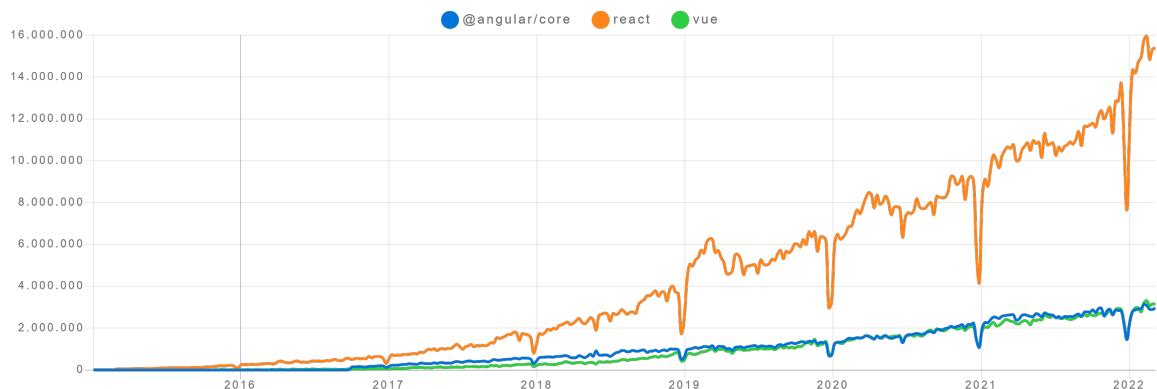


Abbildung 2: wöchentliche Downloads [8]

		Stars	Issues	Version	Updated	Created	Size	
	@angular/core	<a href="#">npm</a>	<a href="#">GitHub</a>	-	-	13.2.6	7 days ago	<a href="#">minzipped size 73.3 KB</a>
	react	<a href="#">npm</a>	<a href="#">GitHub</a>	-	-	17.0.2	a year ago	<a href="#">minzipped size 2.8 KB</a>
	vue	<a href="#">npm</a>	<a href="#">GitHub</a>	28.174	564	3.2.31	a month ago	<a href="#">minzipped size 33.7 KB</a>

Abbildung 3: Direkter Vergleich der Frameworks [8]

## Vorteile

- Durch das virtuelle DOM ist die Performance sehr optimiert und kann gut mit vielen gleichzeitigen Zugriffen zureckkommen
- Unabhängige Components sind leicht zu implementieren und zu wiederverwenden
- React Developer Tools sind sehr ausgereift mit einer großen Nutzbarkeit

## Nachteile

- Durch etliche Updates ist die Dokumentation lückenhaft
- Schnelle Änderungen in kurzer Zeit, neu gelerntes ist in naher Zukunft bereits wieder "alt"
- Inhalte wie JSX sind komplex und schwer zu lernen
- ReactJS setzt einiges an Erfahrung in JavaScript oder TypeScript voraus

### 2.3.4 Vue

Vue.js ist ein kleines, aber sehr einfach funktionierendes Frontend System und wird immer beliebter. Es ist gut sehr darin, Probleme mit denen Angular Developer umgehen müssen zu vermindern oder gar zu lösen. Vue.JS ist außerdem sehr leichtgewichtig und klein, hat aber trotzdem viele nützlich Features wie das Two-Way-Binding out-of-the-box dabei, auch die Verwendung als PWA (Progressive Web Apps) ist möglich. Der wohl größte Nachteil ist die niedrige Trucknumber [9, 10]. Die Trucknumber bei Projekten beschreibt, wie viele Personen ausfallen müssen, dass das Projekt weiterlaufen kann. Je höher die Trucknumber, desto mehr Personen könnten ausfallen und das Projekt würde also weiterlaufen. Im Gegensatz dazu, je kleiner die Trucknumber, desto weniger Personen können ausfallen, das geht so weit, dass das gesamte Projekt an einer einzigen Person hängt. Das letzte Beispiel ist fast bei Vue.js gegeben, in das Core Repository von Vue committed regelmäßig nur der Gründer von Vue, Evan You. VueJS kann in beiden, TypeScript und JavaScript geschrieben werden.

## Vorteile

- Sehr leicht zu lernen, vor allem mit basis JavaScript Wissen
- Flexibles App Layout
- Umfangreiche und ausführliche Dokumentation

## Nachteile

- Sehr kleine Community (Wartung und Support)
- Sehr geringe Truck Number
- Stabilitätsprobleme bei großen Projekten

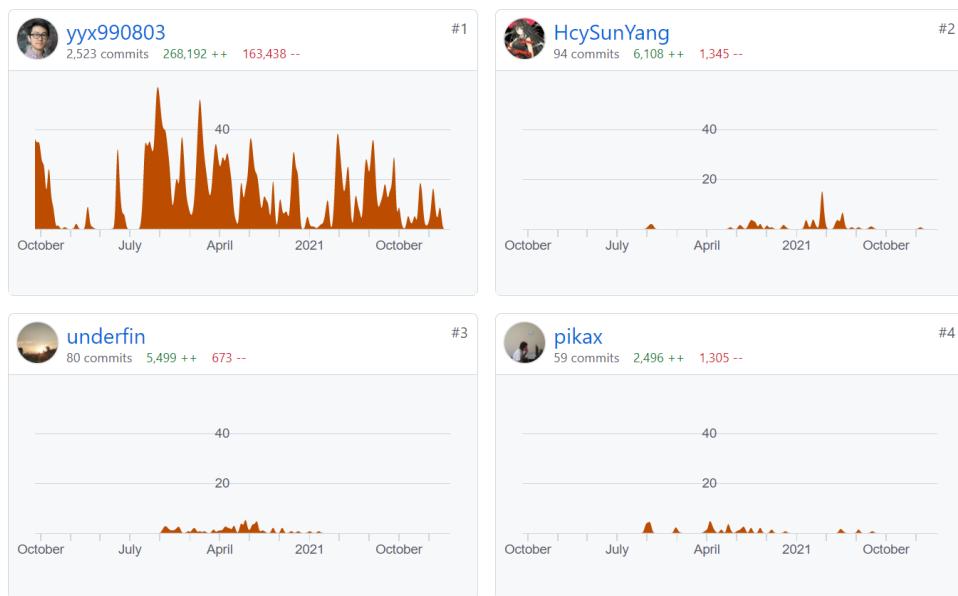


Abbildung 4: Vue.js Core Repository Commits im Zeitraum 16. September 2018 - 9. Februar 2022 [11]

[12, 13, 14]

### 2.3.5 Verwendung von Angular

#### Warum Angular?

Einer der Entwickler hat bereits mehr als zweieinhalb Jahre Erfahrung in der Entwicklung von UserInterfaces und Web-Oberflächen mit Angular. Zusätzlich dazu wird Angular in der HTL Leonding ab dem 4. Jahrgang in der Abteilung Medientechnik unterrichtet, wodurch die Wahl auf Angular naheliegend war. Unabhängig von der Schule und den Entwicklern ist es außerdem eine Anforderung der Betreuerfirma triply Angular zu verwenden, weil triply bei allen Frontend Projekten auf Angular vertraut und um diese Konsistenz zu bewahren ist Angular die logische Wahl.

#### Angular installieren und verwenden

**Node.js** Node.js ist eine asynchrone open-source Event gesteuerte JavaScript Entwicklerumgebung. Node.js ist Plattformunabhängig, das heißt, dass in node geschriebene Anwendungen sowohl auf Windows oder Linux als auch auf macOS problemlos funktionieren. JavaScript wird grundsätzlich im Browser ausgeführt, durch node kann JavaScript jedoch auch serverseitig ausgeführt werden. Dabei erhält man durch die Node.js API umfangreiche Funktionen, um mit dem lokalen Betriebssystem zu interagieren. Node.js wird verwendet, um skalierbare Netzwerk Applikationen zu designen. Im folgenden

Beispiel wird gezeigt, wie man einen einfachen Node.js Webserver implementiert. Durch diese Implementierung können etliche Verbindungen gleichzeitig behandelt werden. Bei jeder Verbindung wird die Callback Funktion ausgeführt. Werden keine Verbindungen aufgebaut, so geht Node.js in den Wartezustand. Wenn man Node.js verwendet, muss man sich keine Sorgen über dead-locks machen, weil es durch die Asynchronität keine locks gibt. Node.js Funktionen führen kaum I/O Prozesse durch, dadurch kann ein Prozess nie blockieren außer es wird die I/O Funktion synchron über die Node.js Library aufgerufen. Weil keine Prozesse blockieren, ist es naheliegend skalierende Systeme in Node.js zu entwickeln [15].

```

1  const http = require('http');
2
3  const hostname = '127.0.0.1';
4  const port = 3000;
5
6  const server = http.createServer((req, res) => {
7    res.statusCode = 200;
8    res.setHeader('Content-Type', 'text/plain');
9    res.end('Hello World');
10 });
11
12 server.listen(port, hostname, () => {
13   console.log(`Server running at http://$${hostname}:$${port}/`);
14 });

```

**npm** npm oder auch ausgeschrieben der Node-Package-Manager ist eine Software um Packages zu Node.js Applikationen hinzuzufügen. npm wird standardmäßig mit Node.js mitinstalliert, es handelt sich um eine CLI (Command-Line-Interface) welche auf die online registry zugreift. Die npm-registry ist eine Online-Datenbank welche public (open-source) und private (auch kostenpflichtige) Packages zur Verfügung stellt. Die Packages werden über die CLI installiert und können über die npm Website gesucht werden. npm ist ein Produkt von Github, Github wiederum gehört seit 2018 zu Microsoft [16].

**TypeScript** TypeScript ist eine typisierte Programmiersprache welche auf JavaScript aufbaut. TypeScript ist außerdem eine Übermenge von JavaScript. Das bedeutet, dass TypeScript alle Eigenschaften von JavaScript hat und aber noch eigene Funktionalität mitbringt. Folgende Features werden im Gegensatz zu JavaScript unterstützt:

- Typisierung von Variablen (auch während des Kompilierens)
- Type inference
- Type erasure
- Interfaces
- Enumerated types

- Generics
- Namespaces
- Tuples
- Async/await
- Classes
- Modules
- Verkürzte Syntax von anonymen arrow-functions
- Optionale Parameter and Defaultwerte für Parameter

Wenn man ein TypeScript Programm ausführt, dann wird der Quellcode von TypeScript Code in JavaScript Code transpiled. Dies wird erzielt, indem eine von mehreren möglichen Optionen ausgewählt wird. Die Auswahl beschränkt sich dennoch auf den TypeScript Checker oder Babel, beide Varianten sind stark verbreitet. TypeScript kann sowohl Clientseitig, als auch Serverseitig (zum Beispiel in Kombination mit Node.js) eingesetzt werden.

### **Beispiel: Funktionen mit und ohne Types**

JavaScript Funktion zum Addieren von 2 Zahlen (ohne types)

```

1  function add(number1, number2) {
2      return number1 + number2;
3  }
4
5
6  add(6, 9); // returns 15
7
8  add("Hallo ", "Welt"); // returns "Hallo Welt"

```

Bei JavaScript gibts es keine typisierten Funktionen, deshalb verhält sich die Funktion immer den Parametern entsprechend.

TypeScript Funktion zum Addieren von 2 Zahlen (mit types)

```

1  function add(number1: number, number2: number): number {
2      return number1 + number2;
3  }
4
5  add(6, 9); // returns 15
6
7  add("Hallo ", "Welt"); // Error: Argument of type 'string' is not assignable to
                           // parameter of type 'number'.

```

Da TypeScript Typisierungen unterstützt, wird bereits zur Compilezeit sichergestellt, dass die richtigen Daten als Parameter übergeben werden. Deshalb tritt beim zweiten Aufruf ein Error auf, weil ein String nicht als Nummer verwendbar ist.

[0, 0]

### Wie funktioniert Angular?

*Ivy* ist die neue Compilation und Rendering Pipeline für Angular. Seit der Angular Version 9 wird standardmäßig die neue Pipeline verwendet und ersetzt damit die Alter Pipeline, welche unter dem Namen *View Engine* bekannt ist.

Da *Ivy* nicht nur eine Rendering Engine, sondern auch eine Compile Engine ist, öffnen sich ganz neue Türen was die Optimierung für und über das gesamte Framework angeht.

Anstatt Template Daten zu generieren, diese in einen Interpreter zu geben, der wiederum entscheidet, welche Operationen ausgeführt werden, wird direkt eine Sammlung an *Template Instructions* generiert. Die *Template Instructions* beinhalten die Logik, welche Components instanziert, DOM Nodes erstellt und Change Detection in Echtzeit ausführt.



Abbildung 5: Ivy Pipeline

**Incremental DOM** Jede Component wird in eine eigene Sammlung an Instructions compiled, diese erstellen DOM Trees und aktualisieren die Daten an Ort und Stelle, wenn diese sich ändern. Zum Erstellen des *Document Object Models* gibt es drei verschieden Funktionen:

- *elementStart* (opening tag)
- *text* (inner html)
- *elementEnd* (closing tag)

Beim Aktualisieren des DOMs wird nicht automatisch jedes Mal ein komplett neuer Tree erstellt, sondern es werden alle Nodes verglichen und nur jene verändert, welche auch tatsächlich verändert wurden. Das spart nicht nur Arbeitsspeicher, sondern auch Zeit.

Der Grund warum Incremental DOM verwendet sind die Ziele welche erreicht werden möchten. Der Fokus liegt auf einer besseren Performance von Web-Applikationen auf Mobilen Endgeräten (Handys, Tablets). Um dies zu erreichen, ist es erforderlich zwei Dinge zu optimieren, *bundle size* und *memory footprint*.

**Shakable Tree** Bei Verwendung der Incremental DOM Strategie wird nicht die Component interpretiert, sondern jede Component verweist auf verschiedene Instructions. Wenn kein Verweis auf eine Instruction zu finden ist, dann wird dieser Teil nicht verwendet und ist somit überflüssig. Durch die Vorteile von Incremental DOM ist dies bereits zur Compile Zeit bekannt, weshalb alle nicht benützten Instructions vom bundle entfernt, was wiederum die *bundle size* beträchtlich verringern kann.

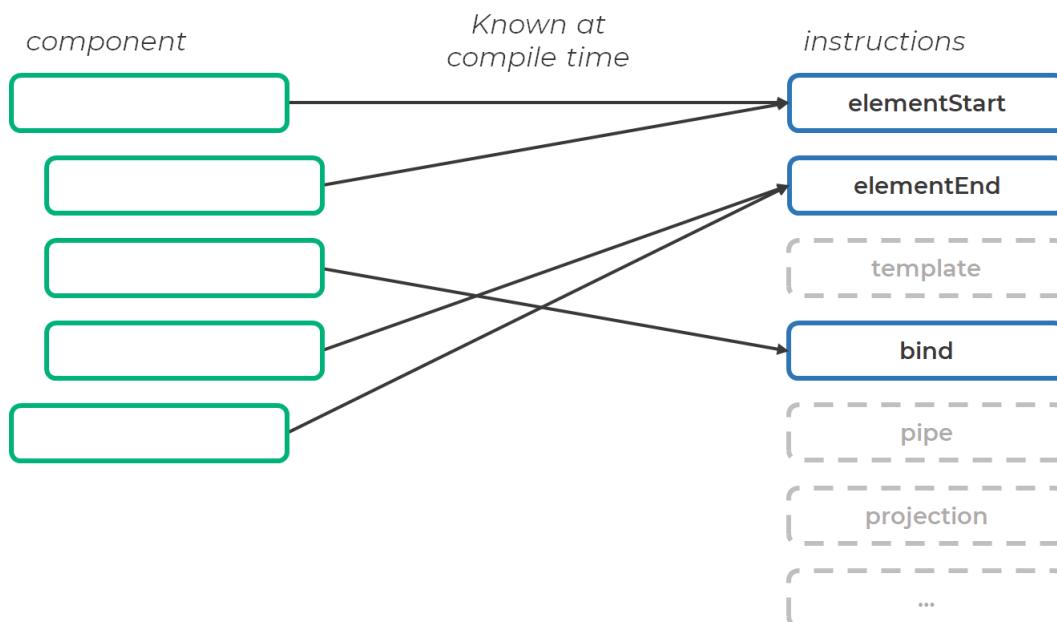


Abbildung 6: Skakable Tree

[?, ?]

## Angular Material

Angular Material ist eine Component-Library von Angular, diese beinhaltet stylistisch aufeinander abgestimmte Elemente. Diese Komponenten können ganz einfach importiert werden, durch das Responsive Design der Inhalte, eignen sich diese sehr gut für die Umsetzung einer Web-Oberfläche, da nicht viel Zeit für das Design verloren geht, sondern der Fokus voll und ganz auf die Funktionalität gelegt werden kann.

Angular Material kann jederzeit über einen Konsolenbefehl zu einem Angular Projekt hinzugefügt werden:

```
ng add @angular/material
```

Bei der Installation kann man bereits vordefinierte Themes von Angular auswählen, zur Auswahl stehen insgesamt 4 verschiedene, zwei Dark- und zwei Lighttheme Varianten [?]:

- Deep Purple & Amber (Light)
- Indigo & Pink (Light)
- Pink & Blue-grey (Dark)
- Purple & Green (Dark)

Weiters hat man die Möglichkeit eigene Themes zu erstellen. Mithilfe von Custom Themes können die von Angular bereitgestellten Komponenten für jedes Projekt individuell eingefärbt werden. Um das zu erreichen, wählt man beim Installieren die fünfte Auswahlmöglichkeit *Custom* aus. Dies erstellt, den dafür benötigten Code in dem *styles.scss* File. Man kann sich über diverse Internet Seiten Farbpaletten generieren lassen, diese werden benötigt, um ein eigenes Theme zu gestalten. Ein Angular Theme hat genau "drei" verschiedenen Farben, wobei es eher Farbpaletten sind.

**primary-palette** Die Primary Palette legt die Hauptfarbe fest, welche in fast allen Komponenten standardmäßig verwendet wird. Man kann allerdings auch bei den meisten Komponenten individuell als Parameter angeben, welche der drei Paletten verwendet werden soll.

**accent-palette** Die Accent Palette definiert eine Akzentfarbe, welche einen hohen Kontrast zur Hauptfarbe haben soll, um bestimmte Elemente besser hervorheben zu können.

**warn-palette** Die Warn Palette besteht meist aus einem Rot, welcher für Fehlermeldungen verwendet wird.

Eine Farbpalette besteht aus einer Basisfarbe und mehrere abstufungen dieser Farbe, um auch hier unterschiedliche Töne für Hervorhebungen zu haben.

Ein Beispiel einer Farbpalette für eine primary-palette:

```

1 $primary-palette: (
2   50 : #e2f3f4,
3   100 : #b8e2e4,
4   200 : #88ced2,
5   300 : #58babf,
6   400 : #35acb2,
7   500 : #119da4,
8   600 : #0f959c,
9   700 : #0c8b92,
10  800 : #0a8189,
11  900 : #056f78,
12  A100 : #a7f7ff,
13  A200 : #74f3ff,
14  A400 : #41efff,
15  A700 : #27ecff,
16  contrast: (
17    50 : #000000,
18    100 : #000000,
19    200 : #000000,
20    300 : #000000,
21    400 : #000000,
22    500 : #fffffff,
23    600 : #fffffff,
24    700 : #fffffff,
25    800 : #fffffff,
26    900 : #fffffff,
27    A100 : #000000,
28    A200 : #000000,
29    A400 : #000000,
30    A700 : #000000,
31  )
32 );

```

Insgesamt hat man die Auswahl aus 36 verschiedenen Komponenten [?, ?, ?].

Die Komponenten werden ganz einfach über einen Custom HTML Tag eingebunden.

## triply Material Library

Triply hat in den letzten Monaten einige eigene Angular Komponenten erstellt und in einer internen Bibliothek gesammelt, vor allem auf diese Inhalte wurde zurückgegriffen, um zum Beispiel einen Slider oder einen true/false Switch zu verwenden. Der Unterschied zu Angular Material liegt darin, dass die Firmeninterne Library aus Files besteht, welche in das Projekt kopiert werden, vereinfach gesagt, man verwendet dabei Custom-Components.



Abbildung 7: Triply Slider und Triply Switch

### Model-View-Controller Design Pattern

Dieses Pattern wird oft für die Entwicklung von User Interfaces verwendet, dabei wird die Logik in drei verschiedenen Elementen (Files) aufgeteilt. Das wird gemacht, um interne Abbildungen und Referenzen von Daten beziehungsweise Informationen in einer Art und Weise aufzuteilen. Das Model ist von der restlichen Anwendung komplett unabhängig. Im Gegensatz dazu ist die View stark abhängig von der zu programmierenden Anwendung. Die Aufgabe der View beschränkt sich rein auf die Visualisierung der durch den Controller bereitgestellten Daten. Der Controller hat eine Vermittlerrolle zwischen Model und View. In Angular ergeben alle drei Files eine gemeinsame Component.

**Model** Das Model File ist das Herzstück der Component, es beinhaltet alle Daten Strukturen und den Aufbau, dabei werden Daten, Logik und Regeln der Component festgelegt und verwaltet. Bei Angular sind das entweder Interfaces oder Klassen ohne Logik. [0]

**View** Die View beinhaltet alle Details zur Darstellung der Daten und Informationen. Es kann mehrere Views für die gleiche Information geben, zum Beispiel ein Balkendiagramm für den Manager oder eine Tabelle für die Buchhaltung.

**Controller** Der Controller ist das Gehirn einer Component, er ist das Verbindungsstück zwischen View und Model. Dabei werden die Daten mittels Getter-Funktionen aus dem Model geladen und so verändert beziehungsweise so manipuliert, sodass die View diese richtig anzeigen kann. Wenn Änderungen der Daten durch die gemacht werden, speichert der Controller diese Änderung mittels Setter-Funktionen im Model.

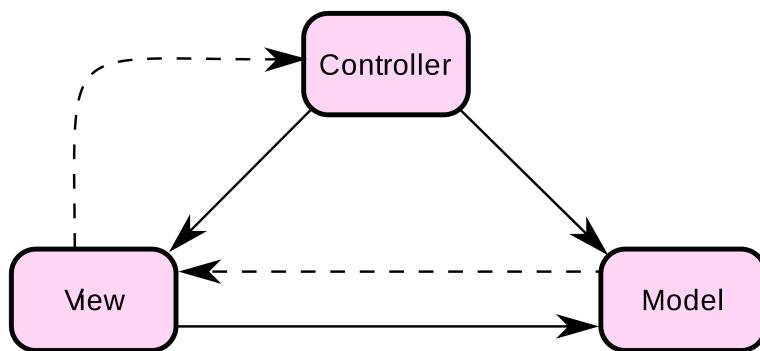


Abbildung 8: Model-View-Controller-Pattern

## 2.4 Map Frameworks

Geo-Daten sind sehr komplexe Daten, welche nur sehr schwer und mit viel Erfahrung ohne Visualisierung interpretiert werden können. Um solche Daten visualisieren zu können werden Map Frameworks benötigt, diese können auf digitalen Karten, die Koordinaten grafisch darstellen, doch auch interaktive Darstellungen sind möglich. Drei der bekanntesten Frameworks sind:

- Mapbox
- Leaflet
- Google Maps

### 2.4.1 Mapbox

Mapbox ist ein amerikanisches Unternehmen, welches sich auf Custom-Maps spezialisiert und die Nische stark vergrößert hat. Dabei geht es um die Darstellung von Geo-Daten auf digitalen Kartensystemen und deren Individualisierung. Weiters ist Mapbox Contributor zum OpenStreetMap (OSM) Projekt, das ist eine Geo-Datenbank mit Kartendaten und Informationen für den gesamten Globus. Große Konzerne wie Amazon, Facebook oder Snapchat setzen auf OpenStreetMap für ihre eigenen Angebote und Services den Kunden gegenüber. Unternehmen wie Tesla, mit einer eigenen Navigationssoftware verwenden die OSM-Datenbank ebenfalls für ihre eigenen Zwecke [?, ?].

#### Vorteile

- OpenSource (bis Version 2.0.0) [?]
- Gutes Handling von großen Datensätzen

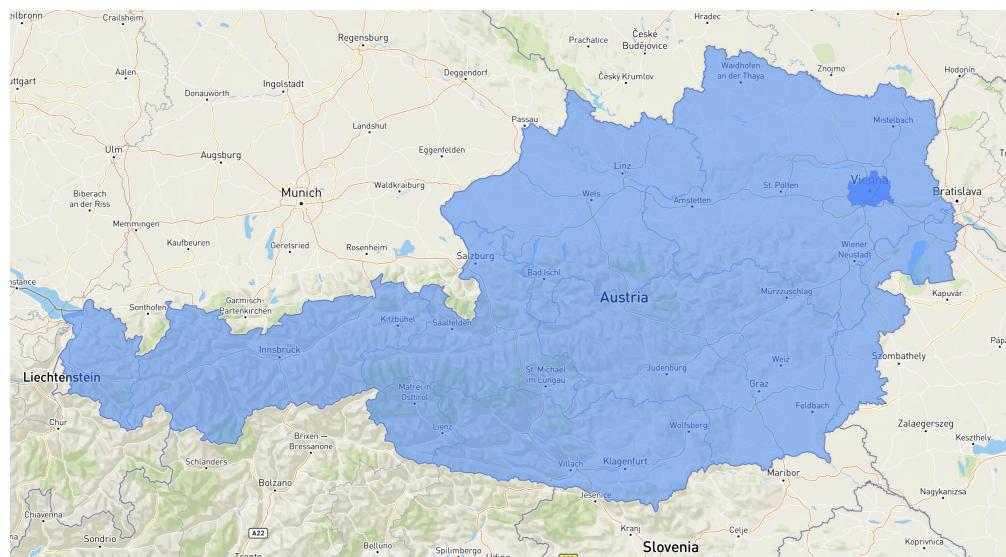


Abbildung 9: Österreich mit Bundesländern in Mapbox

- Individualisierbar
- Gute Unterstützung von komplexen Strukturen

### Nachteile

- Ausbaufähige Dokumentation
- Kein ausgereifter Angular Wrapper

### 2.4.2 Leaflet

Leaflet ist das laut eigenen Angaben die führende Open-Source JavaScript Library, wenn mobile und interaktive Karten gefordert sind. Durch die sehr gute API Dokumentation können ganz leicht Geo-Daten auf Karten dargestellt und interaktive Elemente eingebaut werden. Der Fokus liegt dabei auf den Grundfunktionalitäten, das beinhaltet auch eine Performance Optimierung für fast alle Geräte und Plattformen. Dank einer leichten Anbindungsmöglichkeit für Plugins, gibt es etliche davon im Internet, die man einfach verwenden kann, um die Funktionalitäten zu erweitern. Leaflet wird auch von den ganz großen verwendet: GitHub, Pinterest, Facebook, European Commission, The Washington Post und noch viele mehr. [?]

### Vorteile

- Open-Source (keine Restriktion oder Terms of Service)

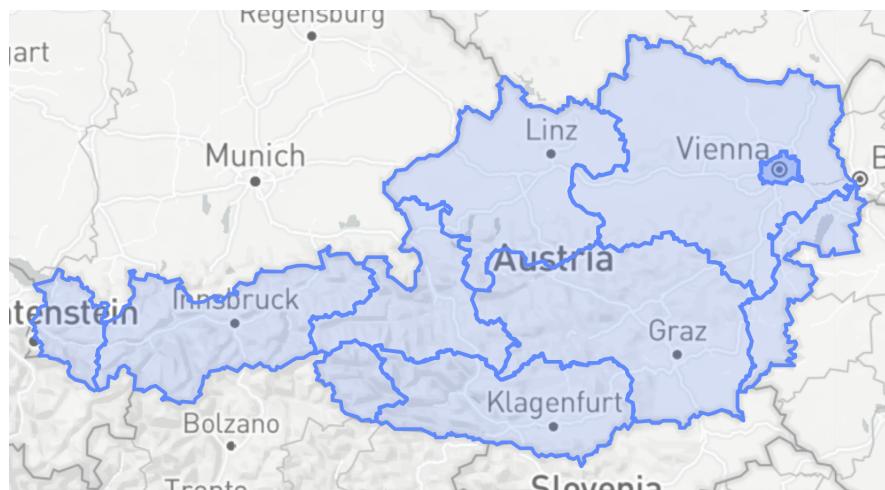


Abbildung 10: Österreich mit Bundesländern in Leaflet

- Hervorragende Dokumentation
- Leichtgewichtig (39KB)
- Plugin Support
- Angular Wrapper
- Bereits von triply in Verwendung

### Nachteile

- Offizielle Dokumentation beinhaltet nur grundlegende Beispiele
- Möglicherweise verwenden von GIS Programmen (zum Beispiel QGIS) notwendig, um Informationen aufzubereiten

### 2.4.3 Google Maps

Google Maps ist der Kartendienst von Google und der Kartenprovider mit den meisten Daten, welche zur Verfügung stehen. OpenStreetMap ist vergleichsweise auf dem Globus nicht deckend, das liegt, jeder kann aber Daten oder Orte beisteuern. Google Maps hingegen, hat Daten bis in jede noch so kleine Straße, das ermöglicht eine sehr gute Informationslage. In Kombination mit Places API und der Maps images API kann niemand mit Google mithalten. Das liegt auch daran, dass Google mehr als Maps betreut und deshalb viel mehr Möglichkeiten als zum Beispiel OpenStreetMap hat. Der wohl größte Nachteil liegt an der kostenpflichtigen Nutzung ab einer gewissen

Nutzerzahl. Ab 1000 Anfragen pro Monat ist Google Maps nicht mehr gratis und damit frei zu nutzen [?].

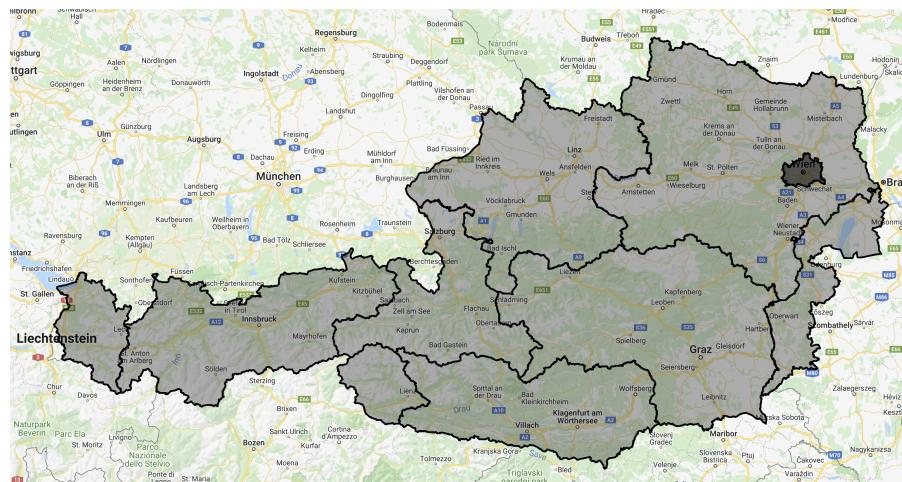


Abbildung 11: Österreich mit Bundesländern in Google Maps

## Vorteile

- Google Ökosystem
- hohe Abdeckung des Globus
- Angular Wrapper

## Nachteile

- Kann kostenpflichtig werden
- Terms of Use und Restriktionen
- Google Logo immer sichtbar
- Farbschema kann kaum verändert werden.

#### 2.4.4 Performance im Vergleich

Um eine Wahl auf ein bestimmtes Framework treffen zu können, wurden mithilfe der Chrome-Developer-Tools die Ladezeiten unter bestimmten Voraussetzungen der einzelnen Frameworks getestet. In diesen Tools gibt es die Möglichkeit zwischen einem schnellem (fast), einem mittel-schnellem (mid-tier) und einem langsamen (low-end) Gerät zu wählen. Dadurch änderten sich die Ladezeiten teils sehr drastisch. Folgende Ergebnisse wurden bei dem Versuch erhalten:

	Leaflet	Mapbox	Google Maps
<b>low-end device</b>	132s	138s	192s
<b>mid-tier device</b>	39,5s	41,69s	56,12s
<b>fast device</b>	1,2s	3,9s	3,23s

## 2.5 Static Site Generators

### 2.5.1 Next.js

### 2.5.2 Jekyll

### 2.5.3 Scully

## 2.6 Deployment Pipeline

Nach dem Download des generierten Projekts sollte die Möglichkeit bestehen, kleine Änderungen daran vorzunehmen und es dann schnell und einfach auf einen Server zu deployen. Der Server ist dabei entweder eine Linux Virtual Machine oder eine *Firebase Hosting* Instanz. Dazu gab es mehrere Ansätze: Ist das Projekt ein Git-Repository, kann eine Automatisierungssoftware, wie Jenkins oder GitHub Actions verwendet werden, die bei jedem *Push*-Event das Projekt buildet und auf den Server deployed. Die zweite Möglichkeit ist ein Node.js Skript, das im Projekt enthalten ist und von der Kommandozeile aus ausgeführt wird.

### 2.6.1 Jenkins

Jenkins ist ein Open-Source Automatisierungsserver. Das Projekt wird in Java entwickelt und kann mit Hilfe von Plugins an spezifische Anforderungen angepasst werden. Der Server wird vor Allem zur Automatisierung von Aufgaben, wie das Builden, Testen und Deployen von Softwareprojekten genutzt. Jenkins muss selbst gehosted werden. Dazu wird ein offizielles Docker Image im Docker Hub bereitgestellt. Die Konfiguration einer Pipeline wird in einem *Jenkinsfile* vorgenommen, das sich im Git-Repository befindet. Um die Dateien zur Virtual Machine zu senden, wurde das Plugin *Publish Over SSH* verwendet. Für das Deployen zu *Firebase Hosting* steht kein Plugin zur Verfügung. Stattdessen wurde auf die *Firebase CLI* zurückgegriffen. Ein großer Nachteil von Jenkins war, dass für jedes Projekt erst ein GitHub Repository und eine neue Pipeline im Web-Interface von Jenkins erstellt werden musste, bevor es automatisch deployed werden konnte.

## 2.6.2 GitHub Actions

GitHub Actions ist eine Automatisierungssoftware, die von GitHub zur Verfügung gestellt wird. Im Gegensatz zu Jenkins werden die Server dabei von GitHub gehosted. Die Konfiguration von sogenannten Actions wird in *YAML*-Dateien im Repository vorgenommen. Auch in GitHub Actions können Plugins verwendet werden, um verschiedenste Aufgaben zu automatisieren. Sowohl für das Deployen auf eine Virtual Machine, als auch zu *Firebase Hosting* stehen Plugins zur Verfügung, was die Konfiguration einfacher als bei Jenkins macht. Wie auch bei einer Jenkins Pipeline muss für jedes Projekt jedoch auch beim Ansatz mit GitHub Actions ein GitHub Repository erstellt werden.

# 2.7 Backend

## 2.7.1 JavaScript

## 2.7.2 Typescript

# 2.8 Webserver

# 2.9 Reverse Proxy

# 2.10 Containerization

## 2.10.1 Docker

## 2.10.2 Docker Compose

# **3 Umsetzung**

## **3.1 Frontend Implementierung**

### **3.1.1 Aufbau und Struktur**

- Grafik von Angular Architektur

### **3.1.2 Sidebar**

### **3.1.3 Mapview**

### **3.1.4 Global Datasources**

### **3.1.5 Custom Color Schemes**

### **3.1.6 Navigation**

# 4 Evaluation des Projektverlaufs

Aufzählungen:

- Itemize Level 1
  - Itemize Level 2
    - Itemize Level 3 (vermeiden)
- 1. Enumerate Level 1
  - a. Enumerate Level 2
    - i. Enumerate Level 3 (vermeiden)

**Desc** Level 1

**Desc** Level 2 (vermeiden)

**Desc** Level 3 (vermeiden)



# Literaturverzeichnis

- [1] C. Ostendarp, „Versionierung von Dokumenten: So behalten Sie den Überblick über Stand und Status Ihrer Dokumente!“ 2018, letzter Zugriff am 17. März 2022. Online verfügbar: <https://www.d-velop.de/blog/prozesse-gestalten/versionierung-von-dokumenten-behalten-sie-den-ueberblick/>
- [2] S. Overflow, „Developer Survey Results 2018,” 2018, letzter Zugriff am 14. März 2022. Online verfügbar: <https://insights.stackoverflow.com/survey/2018/#work--version-control>
- [3] E. Will, „Basic Git Workflow,” 2017, letzter Zugriff am 17. März 2022. Online verfügbar: <https://uidaholib.github.io/get-git/3workflow.html>
- [4] T. Preston-Werner, „Semantic Versioning,” 2022, letzter Zugriff am 18. März 2022. Online verfügbar: <https://semver.org/>
- [5] G. Inc., „The tools you need to build what you want.” 2022, letzter Zugriff am 15. März 2022. Online verfügbar: <https://github.com/features>
- [6] Angular Team, „FEATURES BENEFITS,” 2022, letzter Zugriff am 9. Februar 2022. Online verfügbar: <https://angular.io/features>
- [7] ——, „What is Angular?” 2022, letzter Zugriff am 9. Februar 2022. Online verfügbar: <https://angular.io/guide/what-is-angular>
- [8] John Potter, „@angular/core vs react vs vue,” 2022, letzter Zugriff am 16. März 2022. Online verfügbar: <https://www.npmtrends.com/@angular/core-vs-react-vs-vue>
- [9] Guilherme Avelino, Marco Tulio Valente, Andre Hora, „What is the Truck Factor of popular GitHub applications? A first assessment,” 2017, letzter Zugriff am 15. März 2022. Online verfügbar: <https://peerj.com/preprints/1233v3/>
- [10] Sébastien Dubois, „What’s the bus factor and 7 ways to increase it,” 2020, letzter Zugriff am 14. März 2022. Online verfügbar: <https://medium.com/management-matters/whats-the-bus-factor-of-your-team-and-how-to-increase-it-8bdfb63361fc>
- [11] Association for Progressive Communications, „Wireless technology is irreplaceable for providing access in remote and scarcely populated regions,” 2022, letzter Zugriff am 9. Februar 2022. Online verfügbar: <https://github.com/vuejs/core/graphs/contributors>
- [12] Adarsh Tripathi, „Best Front End Frameworks for Web Development of 2021: The Complete Guide,” 2021, letzter Zugriff am 14. März 2022. Online verfügbar: <https://medium.com/geekculture/best-front-end-frameworks-for-web-development-of-2021-the-complete-guide-ec30098fd1d0>
- [13] Natallia Sakovich, „Top Most Popular Frontend Frameworks 2022,” 2021, letzter Zugriff am 14. März 2022. Online verfügbar: <https://www.sam-solutions.com/blog/best-frontend-framework/>

- [14] Aris Pattakos, „Angular vs React vs Vue 2022,” 2022, letzter Zugriff am 14. März 2022. Online verfügbar: <https://athemes.com/guides/angular-vs-react-vs-vue/>
- [15] Node.js, „About Node.js®,“ letzter Zugriff am 17. März 2022. Online verfügbar: <https://nodejs.org/en/about/>
- [16] Paul Sawers, „Microsoft confirms it will acquire GitHub for 7.5billion,” 2018, *letzter Zugriff am 16. März 2022.* Online verfügbar: <https://www.typescriptlang.org/>

„TypeScript is JavaScript with syntax for types.” 2022, letzter Zugriff am 16. März 2022. Online verfügbar: <https://www.typescriptlang.org/>

TypeScript Tutorial Website, „What is TypeScript,” 2022, letzter Zugriff am 16. März 2022. Online verfügbar: <https://www.typescripttutorial.net/typescript-tutorial/what-is-typescript/>

Angular-Enterprise.com, „Design patterns to know in Angular,” 2022, letzter Zugriff am 10. März 2022. Online verfügbar: <https://angular-enterprise.com/en/ngpost/courses/design-patterns/>

# **Abbildungsverzeichnis**

1	git workflow [3] . . . . .	4
2	wöchentliche Downloads [8] . . . . .	9
3	Direkter Vergleich der Frameworks [8] . . . . .	9
4	Vue.js Core Repository Commits im Zeitraum 16. September 2018 - 9. Februar 2022 [11] . . . . .	11
5	Model-View-Controller-Pattern . . . . .	15

# **Tabellenverzeichnis**

# Quellcodeverzeichnis

hello-world.js . . . . .	12
app.js . . . . .	13
app.ts . . . . .	13

# **Anhang**