# Kapitel 2

# Verwendete Technologien

## 2.1 Ionic 3

Ionic 3 ist ein frei verfügbares Open-Source SDK zur Entwicklung von nativen und progressiven Web-Apps. Der besondere Fokus des Frameworks liegt auf mobilen Endgeräten. So kann eine Anwendung gleichzeitig für iOS, Android und WindowsPhone Geräte entwickelt werden. Entwickelt werden die Apps auf Basis von HTML5, CSS3, Angular 4 und TypeScript. Um auf den einzelnen Geräten die integrierten Hardwarefunktionen benutzen zu können wird Apache-Cordova verwendet, diese werden in Ionic als Native-Plugins bezeichnet.
Das Ionic Framework steht unter der MIT-Lizenz, wodurch es sowohl privat als auch geschäftlich kostenlos verwendet werden kann.

## 2.2 Angular 5

Angular ist ein Web-Framework zur Erstellung von Single-Page Applications (SPA). Es basiert auf TypeScript. Angular verwendet das MVVM-Pattern (Model-View View-Model). Die Komponente (Component) entspricht hier dem ViewModel, welches die UI-Logik enthält, sie tauscht Daten mit dem Model aus und stellt Methoden und Dienste bereit. Die View wird durch Databinding an das ViewModel gebunden und ist somit einfach austauschbar. Sie wird unter Verwendung von HTML5 und CSS erzeugt. Das Model, enthält alle Inhalte die vom Benutzer angezeigt und verändert werden können. Angular ist seit November 2017 in der Version 5 erhältlich. Das Framework ist Open-Source und Google ist an der Entwicklung beteiligt. Unit-Tests und End-to-End Tests werden von Angular unterstützt.
Hier der Link zur Angular-Hompage: `https://angular.io/`

## 2.3 Typescript

TypeScript ist eine freie Open-Source Programmiersprache, die von Microsoft entwickelt wird. Bei der Programmiersprache handelt es sich um eine typisierte Obermenge von JavaScript. Sie basiert auf den ECMAScript 6 Standard und erweitert die JavaScript Sprache mit einer statischen Typisierung, wie auf der Abbildung 2.1 zu sehen ist. Um die Kompatibilität zu gewährleisten, ist JavaScript Code gülitger Typescript Code.

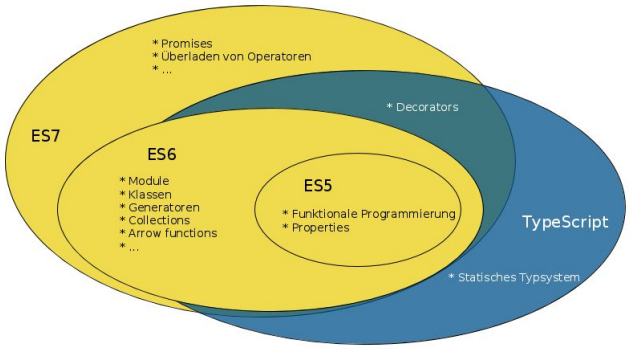


Abbildung 2.1: *TypeScript*

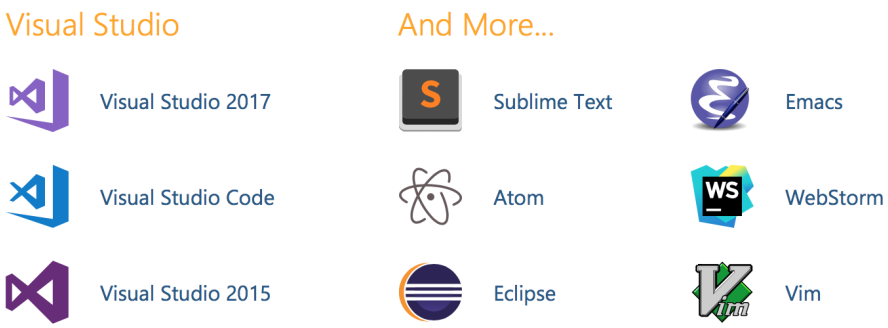


Abbildung 2.2: *Empfohlene IDEs für Typescript*

## 2.4 Cordova

Apache Cordova ist ein Open-Source Entwicklungs-Framework für mobile Plattformen. Mit Cordova ist es möglich moderne Web-Technologien, wie HTML5, CSS3 und JavaScript (hier TypeScript) für plattformübergreifende Entwicklung zu nutzen. Um auf die Sensoren und Daten der einzelnen Geräte zu zugreifen werden Standard APIs der einzelnen Betriebssysteme verwendet. Cordova dient als Mittelschicht zwischen

Anwendung und Gerät. Abbildung 2.3 zeigt den allgemeinen Aufbau der Architektur.

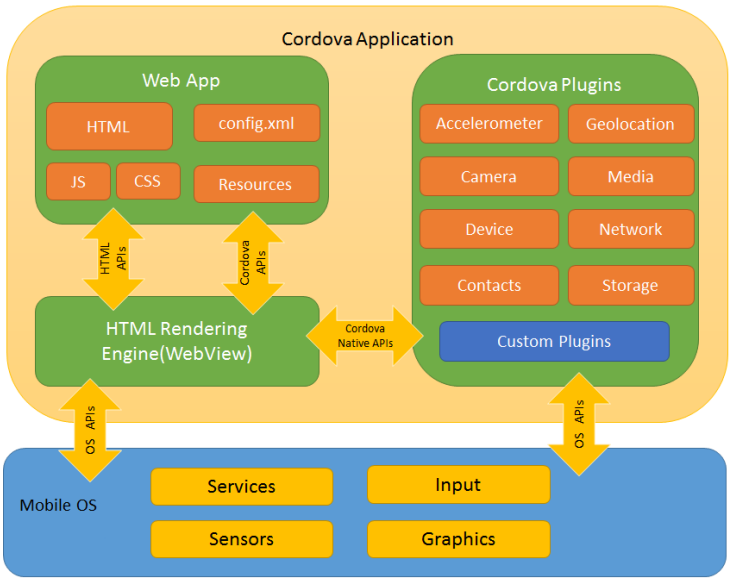


Abbildung 2.3: *Cordova Architektur*

## 2.5 iBeacon

Sogenannte iBeacons sind Bluetooth Low Energy Sender, die nach einem proprietärem von Apple spezifizierten Protokoll, dem iBeacon Protokoll, arbeiten. Die Idee war es, Lokalisation in Gebäuden durchzuführen. Beacons eignen sich aber auch für den Gebrauch im Freien, damit können unabhängig von GPS-Signal Ereignisse, wie das Erreichen oder Verlassen eines bestimmten Bereichs detektiert werden. Zur Identifizierung eines einzelnen Beacons werden bei Bustracker vier Parameter verwendet.

**UUID**

Die UUID gibt in diesem Falle die sogenannte Region an. Mehrere iBeacons können die gleiche UUID haben. Alle Beacons mit der gleichen UUID gehören zur gleichen Region. Zum Beispiel könnten 100 Beacons mit der gleichen UUID einem Busunternehmen zugeordnet sein.

**Major**

Der Major-Parameter unterteilt die Beacons einer UUID. Zum Beispiel könnte dies eine Linie (Strecke) des Busunternehmens sein.

**Minor**

Der Minor-Parameter unterteilt eine Major-Gruppe in einzelne Beacons. Dies könnte einer Haltestelle auf einer Linie entsprechen.

**Identifier**

Identifier ist ein String um ein Beacon zu beschreiben. Zum Beispiel könnte das hier der Name der Haltestelle sein.

Zur konkreten Anwendung kommt diese Technologie im Abschnitt 4.2.3.2.

## 2.6 REST

REST bedeutet Representational State Transfer und Beschreibt eine Architektur. Eine REST API ist **Stautslos**, d.h. es gibt keine Nutzersessions. Jede Anfrage erzeugt eine neue Ressource, alle Daten werden nochmals generiert. Jede dieser Ressourcen ist über eine spezifische URI adressierbar. Ressourcen können in unterschiedlichen Repräsentationen vorliegen. Die gleiche Information kann in unterschiedlichen Formaten abgerufen werden, z.b. HTML und JSON. [1]

## 2.7 Werkzeuge

### 2.7.1 WebStorm

Aus einem vorhergehenden Projekt war die Integrierte Entwicklungsumgebung WebStorm bereits bekannt.
In WebStorm sind die Technologien HTML5, TypeScript, Angular und PhoneGap/-Cordova bereits integriert. Die IDE lässt sich durch ein breites Angebot an Plugins erweitern. Webstorm basiert auf der IntelliJ IDEA Plattform und verwendet Konzepte, die bereits durch IDEs wie AndroidStudio bekannt sind. WebStorm ist auf JavaScript spezialisiert und bietet darauf zugeschnittene Funktionen wie z.B. Codevervollständigung, automatisches Importieren, Signaturinformationen bei Funktionsaufruf und kontextbezogene Hervorhebung von Syntax. Die Integration von Git/Github erwies sich ebenfalls als sehr hilfreich. Es wird kein weiteres Programm neben der IDE benötigt.
Das Interface ist in Abbildung 2.4 zu sehen.

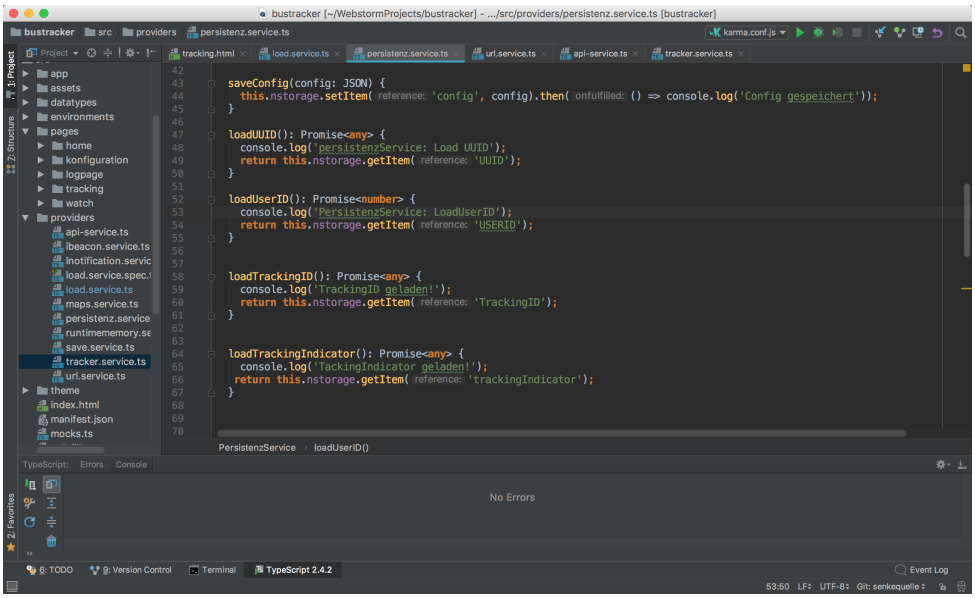


Abbildung 2.4: *WebStorm IDE*

### 2.7.2 compodoc

Zu jeder Software sollte es eine Dokumentation geben. Der Dokumentationsprozess ist in der Regel aufwendig und bedeutet für die Entwickler zusätzliche Arbeit. Es gibt jedoch Tools, die diesen Prozess automatisieren. **Compodoc** ist ein solches Tool.
Compodoc ist Kommandozeilen-basiert und erzeugt eine statische Dokumentation von Angularanwendungen. Die Dokumentation ist in Art einer Website ausgeführt. Es besteht die Auswahl zwischen verschiedenen Themes bzw. Templates. Die Templates sind bereits responsiv und unterstützen moderne Webtechnologien.
Diese „Website“ wird lokal gehostet und benötigt somit keinen Server. Eine mächtige Such-Engine namens „lunr.js“ sorgt dafür, dass die Dokumentation effizient durchsucht werden kann. Compodoc parsed den Code lediglich, es ist keine TypeScript-Kompilierung notwendig. Durch das Parsen enthält die Dokumentation alle Elemente der Anwendung automatisch. Sie sind im Inhaltsverzeichnis gelistet. Es besteht die Möglichkeit zu jeder Komponente den Quellcode direkt einzusehen. Kommentare die im JSDoc-Format vorliegen, werden von Compodoc „verstanden“ und in die Dokumentation mit aufgenommen. (Abbildung 2.5)

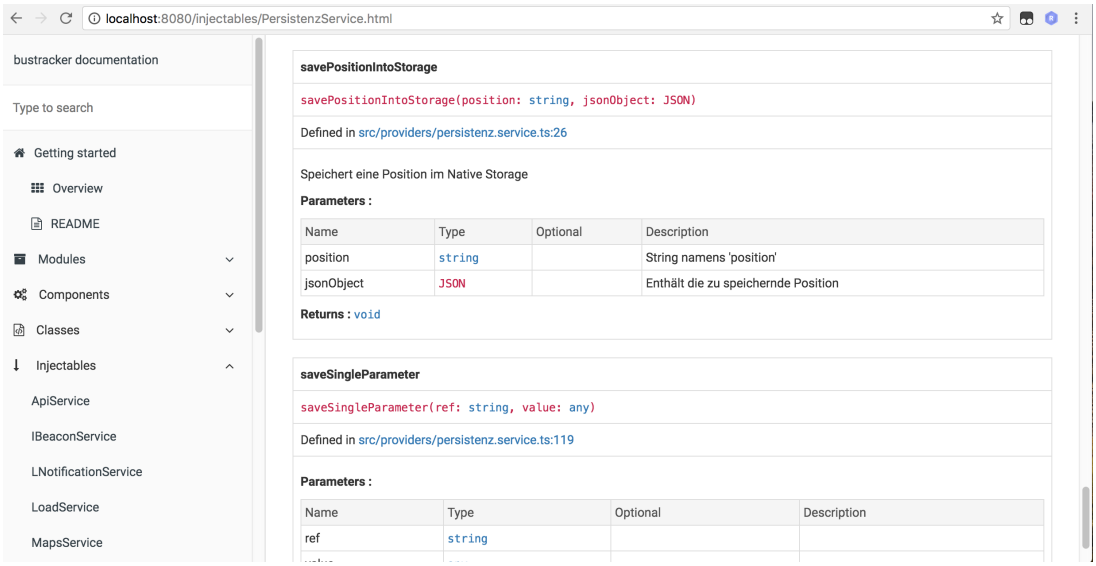


Abbildung 2.5: *Compodoc-Dokumentation*