# DBAF-Dokumentation Vorbereitung

## Die Idee

* Daten von der Deutsche Bahn API für den Nutzer einfach und übersichtlich laden und darstellen
* Funktionalitäten für verschiedenen Sichtungsmöglichkeiten der API-Daten: alle Stationsdaten deutschlandweit, Fahrplanauskunft für alle Anfahrten und Abfahrten an einem Bahnhof an einem vom Nutzer festgelegten Datum, Routenplaner zur Darstellung verschiedener Daten für eine Route von Startbahnhof A zu Startbahnhof B, Karte für alle Bahnhöfe in Deutschland, welche nach Bundesland gefiltert werden können

## Verwendete Programme und Hilfsmittel

* Visual Studio Code (<https://code.visualstudio.com/>) zur Entwicklung des Source-Codes
* XAMPP für die Ausführung des Quellcodes (<https://www.apachefriends.org/>)
* Die API <https://v6.db.transport.rest/> zum Laden der Daten, welche intern auf die Deutsche Bahn API zugreift, wofür kein Zugangs-Schlüssel benötigt wird
* DevExtreme JavaScript Komponenten (<https://js.devexpress.com/>) von DevExpress zur einfachen Erstellung der Benutzeroberfläche und zur übersichtlichen Anzeige aller Daten für den Nutzer

## Installationshinweise

* Alle Quelldateien, diese Dokumentation sowie der Arbeitsnachweis befinden sich auf GitHub unter <https://github.com/chfalkk/DBAF>
* Alle Dateien im Ordner „src“ können entweder per XAMPP ausgeführt werden oder direkt per Ausführung der index.html, da das gesamte Projekt über JavaScript gesteuert wird, wofür kein Zusatzprogramm benötigt wird, außer der Browser
* Hinweis: Für die Ausführung muss JavaScript im Browser aktiviert sein

## Funktionalitäten

### Stations-Daten

(BILD IST TODO)

* Alle Bahn-Stationen in Deutschland werden in einer Tabelle anzeigt
* Können gefiltert werden nach Bezeichnung, Stadt, Postleitzahl, Straße
* Standardmäßig nach Bundesland gruppiert – Gruppierungen können auch vom Nutzer festgelegt werden
* Anzahl der Stationen insgesamt und nach Bundesland wird angezeigt

### Fahrplanauskunft

(BILD IST TODO)

* Station und Datum können angegeben werden
* Abfahrten und Ankünfte an diesem Datum werden angezeigt
* Filtern und Sortierung nach Ankunftsprognose, genauer Ankunft, Herkunft/Richtung und Plattform

### Routenplaner

(BILD IST TODO)

* Start- und Zielbahnhof können angegeben werden
* Nächster Startzeitpunkt kann angegeben werden
* Berechnung der Reise und Zwischenstopps durch die API
* Anzeige einer Übersicht mit allen wichtigen Daten sowie alle Daten zu den Zwischenstopps
* Für Richtigkeit der Daten wird nicht garantiert, da sich Fahrpläne ändern, verzögern oder gestört werden können

### Karte

(BILD IST TODO)

* Anzeige aller Stationen deutschlandweit
* Filterung nach Bundesland möglich
* Kartenansicht kann vom Nutzer eingestellt werden
* Anzeige von Bahnhofsdaten mit Klick auf einen Marker
* Weiterleitung zur Fahrplanauskunft mit Klick auf „Abfahrten und Ankünfte zu diesem Bahnhof anzeigen“, nachdem ein Marker ausgewählt wurde

## Funktionalitäten für Fortgeschrittene

### DevExtreme-Komponenten

* von uns genutzt zur einfachen Erstellung von übersichtlichen Oberflächen
* Über 50 Komponenten werden im DevExtreme-Library mitgeliefert, inklusive Icons, Skins bzw. Designs (dunkler Stil, heller Stil, Farben für die Seitendarstellung), Spracheinstellungen und weiteren nützlichen Funktionen

(BEISPIELBILD VON GRID-KOMPONENTE SOURCE CODE IN ROUTEPLANNING.JS)

* Beispiel hier: Fahrplan-Tabelle im Routenplaner
* Komponenten werden über Properties eingestellt
* Daten werden in einer Variable gespeichert (hier: eGridJourneys), wodurch einzelne Einstellungen während der Laufzeit geändert oder gelesen werden können (z.B. Eingabewerte in Formularen)
* Wichtige Einstellungsmöglichkeiten bei der Tabelle:
  + „dataSource“ (Datenquelle)
  + „keyExpr“ (Primärschlüssel der Datenquelle)
  + „columns“ (Spalten) 🡪 „dataField“ (Name des Datenfeldes in der Datenquelle), „visible“ (Sichtbarkeit), „caption“ (Anzeigename), „sortOrder“ (Sortierung)
  + Und viele Weitere
* Übersicht aller weiteren Komponenten: <https://js.devexpress.com/Overview/Widgets/>
* Alle Dateien des Libraries befinden sich im Ordner „dx\_22.2.4“

### Selbstdefinierte Funktionalitäten

* Alle selbstdefinierten JavaScript-Funktionalitäten befinden sich im Ordner „js“.
* Diese beinhalten:

|  |  |  |
| --- | --- | --- |
| **Dateiname** | **Funktion** | **Beschreibung** |
| dbaf.js | - | Konstanten, Grundeinstellungen der Seiten (Deutsche Sprache für die Komponenten, Dunkles Design, maximale Anfragegrenze) |
| request.js | Klasse DBAF\_Request mit Konstruktur constructor(RequestParam, Async, OnSuccess) |  |
| request.js | IsAPIAvailable() |  |
| request.js | GetAllStations(FederalState, OnSuccess = null) |  |
| request.js | GetAllDepartures(StationID, TargetDate, OnSuccess = null) |  |
| request.js | GetAllArrivals(StationID, TargetDate, OnSuccess = null) |  |
| request.js | ConvertMinutesToStr(Minutes, ShowZero) |  |
| request.js | GetAllJourneys(StartStationID, EndStationID, StartDateTime, OnSuccess = null) |  |
| toolbar.js | - | Einstellung der Navigationsleiste |
| ui.js | StretchToClient(ElementID) |  |
| ui.js | StickToClient(ElementID) |  |

## Erweiterungsmöglichkeiten

* Anzeige der Fahrstühle nicht Stationsweise, sondern gesondert für jedes Gleis in der Tabelle (Abfrage an die FaSta – Station Facility Status der Deutsche Bahn API)
* Filterung der bevorzugten Transportmittel beim Routenplaner (Bus, Bahn, U-Bahn, S-Bahn, Regionalbahn, ICE) – zurzeit wird die optimalste Reiseroute angegeben, ohne die Transportmittel zu beachten

## Arbeitsnachweis

(TODO: wird am Ende eingefügt, ist aber schon vorbereitet)

## Quellenverzeichnis

|  |  |  |
| --- | --- | --- |
| **Bezeichnung** | **URL** | **Aufrufdatum** |
| DevExtreme-Komponenten | <https://js.devexpress.com/> | 28.02.2023 |
| V6-REST-API zur Abfrage aller Daten | <https://v6.db.transport.rest/> | 06.03.2023 |
| Fontawesome (für die Icons auf index.html) | <https://fontawesome.com/start> | 22.04.2023 |