Entwicklerdokumentation

Gefördert durch

netidee
PROJEKTE

Philipp Toscani, Nicolaus Piso, Julius Piso

27.11.2020

Zusammenfassung

eTrax | rescue ist eine Cloud-Software zur Abwicklung von Personensucheinsätzen. Die Software wurde initial für den Einsatz durch Rettungs-/Suchhunde Organisationen entwickelt. Die Software wurde zwar spezifisch für diese Zielgruppe entwickelt, soll aber nicht auf diese beschränkt sein. Die standardisierten Abläufe sind für die Abwicklung von Personensuchen entwickelt und erprobt.
Durch die Unterstützung durch die Internet Privatstiftung Austria (netIdee Projekt 4546) steht **eTrax | rescue** als MIT lizenzierte OpenSource Software für die Allgemeinheit kostenfrei zur Verfügung. Dieses Handbuch enhält die wichtigsten Hinweise zur technischen Dokumentation von **eTrax | rescue** für die Weiterentwicklung. Da die Software laufend weiterentwickelt wird, ist auf die Übereinstimmung der Dokumentation mit der Softwareversion zu achten. Generell wird empfohlen, den aktuellsten Stand dieser Dokumentation aus dem **eTrax | rescue Repository** auf GitHub zu verwenden (siehe https://github.com/etrax-rescue).

Viel Erfolg bei der Weiterentwicklung wünschen

Philipp Toscani
Nicolaus Piso
Julius Piso

Dokument kompiliert: 27. November 2020
Für Softwareversion: 1.0.0

Inhaltsverzeichnis

Webserver

Development Environment Setup

Wichtig: Für die Entwicklung von eTrax | rescue muss node.js installiert sein

- Repository auschecken
- Alle nötigen **node-Pakete** sind im **package.json**-File angeführt
- Development Files liegen im **dev** Verzeichnis
- Beim Building-Prozess werden die erzeugten Dateien in ein Verzeichnis mit dem Namen **v5** geschrieben
- Der Building-Prozess wird durch **gulp** gesteuert
  - **webpack-stream** importiert die js Files in ol.js (import jQuery from 'jquery')

Installation des Development Environments

- Um die benötigten **node-Pakete** zu installieren ein Terminal Fenster öffnen und **npm install** eingeben

Anstoßen des build Prozesses

- Zum Builden im Terminal **npm run build** eingeben
- Mit **npm run watch** wird die **watcher** Funktion von **gulp** aktiviert, die Änderungen in den Development Files verfolgt und den Build anstößt wenn Änderungen gespeichert werden

Setup der Webapplikation

Systemvoraussetzungen

Voraussetzungen zur Installation:

- MySQL (5.7.28 oder höher) - 1 Datenbank
- PHP (7.3 oder höher)
- Zur Nutzung der APP bzw. BOS-Schnittstelle muss jeweils 1 Subdomain angelegt werden können

Download

- Aktuellste Release im Repository `https://github.com/etrax-rescue/webapp/releases` oder
- Builden der aktuellen Version (siehe oberhalb)

Setup

Vorbereitung

Der Inhalte des release Verzeichnisses muss in ein Verzeichnis am Server gelegt werden (z.B. webroot), sodass auf gleicher Ebene beim Setupvorgang noch ein Verzeichnis erstellt werden kann.

Level 1	Level 2	Level 3
webroot		
	bos_import	
		data
	v5	
		css
		datenschutz
		...
		vendor
secure		
	data	
		info.inc.php
		secret.php

Inhalt und Verzeichnis 'secure' werden vom Installationsskript automatisch angelegt. Der Name und die relative Lage des Verzeichnisses darf nicht verändert werden, da sonst die relativen Bezüge nicht mehr stimmen.

Domain und Subdomain einrichten

- **Startseite**: Das Hauptverzeichnis für die Applikation ist **/v5**
- **BOS-Schnittstelle:** Nutzung ist optional; Weiterleitung auf **/bos_import**

Setup durchführen

1. Adresse für die Startseite aufrufen und **/install** hinzufügen (z.B. https://etrax.at/install)
2. Das Setupskript ausführen
3. Das Verzeichnis **/install** vom Webserver löschen

Im Zuge des Setups wird das Verzeichnis **/secure** angelegt. Darin gibt es das Verzeichnis **/data**, in welchem für jeden Einsatz ein eigenes Verzeichnis mit der ID des Einsatzes angelegt wird. In der Datei **info.inc.php** sind die Zugangsdaten für die MySQL Datenbank sowie der Google Places API Key für die Adressuche gespeichert. Die Datei **secret.php** enthätl den private Key für die Verschlüsselung der Daten. **Das Gesamte Verzeichnis /secure muss außerhalb des webzugriffes liegen.**

Wichtigste Anpassungsmöglichkeiten

Startseitentexte

Auf der Startseite gibt es die **Lizenzhinweise**, **Datenschutzhinweis** und **Impressum**. Diese drei Seiten öffnen als B**ootstrap Modal**. Die angezeigten Inhalte können im Verzeichnis **/v5/inc/startseitentexte.php** bearbeitet werden.

Kontakt E-mailaresse:

Bei den Datenschutzhinweisen und im Impressum wird eine E-mailadresse angezeigt. Diese wird aus einem Array aufgebaut. Die Aufteilung muss genau wie im Beispiel unterhalb erfolgen (3 Teile vor dem @, 2 Teile danach).

```
$text["email"] = array('sup','po','rt','@','etr','ax.at');
```

Startseitentexte:

```
$text["datenschutz"] = ''; //Information zum Datenschutz im öffentlich sichtbaren Bereich
        der Seite (Startseite)
$text["datenschutz_user"] = ''; //Information zum Datenschutz, die User nach dem Login
        angezeigt bekommen
$text["impressum"] = ''; //Impressum der Website.
$text["license"] = ''; //Information zu den verwendeten Lizenzen.
```

Die Formatierung kann mittels HTML und Bootstrap erfolgen.

Kartenmaterial

Verfügbares Kartenmaterial kann in der Datei **/v5/inc/include.php** definiert werden. Für jedes Kartenmaterial sind folgende Informationen einzutragen (am Beispiel der Open Topomap):

```
$path['opentopomap'] = array('path' => 'https://opentopomap.org/', //wie url,
          aber ohne z, x, y Verzeichnis
    'url' => '//opentopomap.org/{z}/{x}/{y}.png', //Link zum Tile ohne http: oder https:
    'name' => 'OpenTopoMap',
    'name_js' => 'otm', //Kürzel, unique
    'printname' => 'opentopomap', //Selber Name wie der Key
    'printable' => true,
    'dir' => 'xy', //Kann xy oder yx sein - siehe url Key ({x}/{y}.png --> xy
    'type' => 'xyz',
    'copyright' => 'Grundkarte: opentopomap.org - CC-BY-SA', //Text, der bei den Karten-
         ausdrucken erscheint
    'attributions' => "Kartendaten: &copy; <a href='https://openstreetmap.org/copyright'>
         OpenStreetMap</a>-Mitwirkende, <a href='http://viewfinderpanoramas.org'>SRTM
        </a> | Kartendarstellung: &copy; <a href='https://opentopomap.org'>OpenTopoMap
        </a> (<a href='https://creativecommons.org/licenses/by-sa/3.0/'>CC-BY-SA</a>",
        //Text der im Kartenfenster angezeigt wird
    'zlim' => 17, //Maximales Zoom Level
    'format' => 'png', //meist PNG oder JPEG
    'land' => 'world', //Für weltweite Karte 'world' oder einen Wert aus dem Array $org_land
    'org' => '', //Wenn nur für gewisse Organisationen verfügbar dann OIDs mit ; separiert anführen
    'demotile' => 'https://opentopomap.org/15/17859/11347.png'); //Für die Kartenauswahl
         im Adminbereich
```

Ressourcen

Die für die Organisationen verfügbaren Kategorien von Ressourcen können in der Datei **/v5/inc/include.php** definiert werden.

```
$ressource["NEU"] = array('typ_kurz' => "NEU", 'typ_lang' => "Neu angelegte Ressource");
         //Muss vorhanden sein
$ressource["R-KFZ"] = array('typ_kurz' => "R-KFZ", 'typ_lang' => "Kraftfahrzeug");
```

Jede angelegte Ressource steht als neue Auswahl im Select zur Verfügung. Der `'typ_kurz'` muss unique sein.

App Einstellungen im Administrationsbereich

Die für die Organisationen verfügbaren Werte für die individuelle Konfiguration der App können in der Datei **/v5/inc/include.php** definiert werden.

```
//Aufzeichnungshäufigkeit von Trackingpunkten
$app_position["15"] = array('time' => "15", 'text' => "15 Sekunden");
$app_position["30"] = array('time' => "30", 'text' => "30 Sekunden");
...
// Minimalabstand zwischen Trackingpunkten
$app_distance["25"] = array('distance' => "25", 'text' => "25 Meter");
$app_distance["50"] = array('distance' => "50", 'text' => "50 Meter");
...
//Aktualisierungshäufigkeit der Informationen zur vermissten Person
$app_suchinfo["15"] = array('time' => "15", 'text' => "15 Minuten");
$app_suchinfo["30"] = array('time' => "30", 'text' => "30 Minuten");
...
```

Anpassung von Karten bzw. Einsatzbericht

Alle Berichte und Karten, die im Format PDF erstellt werden, sind im Verzeichnis **/v5/pdf** zu finden. Die PDF Erstellung erfolgt mittels TCPDF. Die Formatierung erfolgt mittels css Styles, die jeweils im Anfangsbereich der Dateien definiert sind.

- **einsatzberichtpdf.php:** Alle Elemente des Einsatzberichtes.
- **handzettel.php**: Handzettel mit Informationen zur vermissten Person
- **mapawesome.php:** Lagekarte bestehend aus mehreren Seiten
- **suchgebiet_als_pdf_senden.php:** Suchgebietskarte welche per E-mail verschickt wird
- **suchgebietpdf.php:** Suchgebietskarte für den Ausdruck

App

Die *eTrax | rescue* App ist der offizielle Client des *eTrax | rescue* Projektes, dessen Ziel es ist die Verwaltung von Personensuchen für Hilfsorganisationen zu vereinfachen, sowie die zentrale Koordination von Suchteams aus der Einsatzleitung zu ermöglichen. Die App ist sowohl für Android als auch für iOS verfügbar.

Funktionsumfang

- **Anmeldung/ Rückmeldung zu einem Einsatz**
- **Statusmeldungen:** mit der App kann der Einsatzleitung der derzeitige Status gemeldet werden (In Anreise, Am Einsatzort, ...)
- **Live-Location-Tracking:** die App greift bei gewissen (von der Organisation festgelegten) Statusmeldungen auf den Gerätestandort zu und stellt diesen der Einsatzleitung zur Verfügung.
- **Setzen von Points of Interest (POIs):** die App ermöglicht es mit der Handykamera ein Foto aufzunehmen und dieses mit dazugehörigem Standort und einer kurzen Beschreibung zur Einsatzleitung zu schicken.
- **QuickActions:** Bestimmte Statusmeldungen werden in einer schnellzugriffs Schaltfläche verfügbar gemacht, damit diese mit nur wenigen Interaktionen zurückgemeldet werden können.

Installation

Über die App Stores

Manuell

> Die App wurde in Dart geschrieben und verwendet das Flutter UI Toolkit. Um die App selbst zu builden muss daher die entsprechende Entwicklungsumgebung installiert sein.

1. git Repository herunterladen:

```
git clone https://github.com/etrax-rescue/etrax-rescue-app.git
cd etrax-rescue-app
```

2. Dependencies installieren/ updaten

```
flutter pub upgrade
```

3. App kompilieren

```
flutter run --release
```

App Architektur

Die Architektur der App basiert auf einer vereinfachten Version des Clean Architecture Prinzips. Für die Implementierung wurde insbesondere die Version dieses Architekturprinzips von Matt Rešetár (mit dessen freundlicher Genehmigung) als starke Inspiration herangezogen.

Background Location Plugin

Da die Ortungsfunktionalität plattformspezifische Funktionen benötigt (Android/iOS) wurde sie als separates Plugin entwickelt, welches in folgendem Repository zu finden ist: `https://github.com/etrax-rescue/flutter_background_location`

Flutter Background Location Plugin

This plugin brings background location updates to Flutter apps. It is based on the flutterlocation plugin, which was extensively modified under the hood to enable background location access on Android and iOS (while retaining some API compatability with the original plugin).

Getting Started

Installation

Add the following lines to your app's *pubspec.yaml* dependencies section and run *flutter pub get*.

```
background_location:
  git:
    url: https://github.com/etrax-rescue/flutter_background_location.git
    ref: main
```

Android

On Android the background service has to be registered. One simply has to add the following lines to the app's AndroidManifest.xml file under the *application* section.

```
<service android:name="at.etrax.background_location.BackgroundService"
         android:foregroundServiceType="location"
         android:exported="false"/>
```

The *uses-permission* statements are inherited from the plugin's manifest.

iOS

To use the plugin on iOS, you have to add the following permissions to your app's *Info.plist* file.

```
NSLocationAlwaysAndWhenInUseUsageDescription
NSLocationWhenInUseUsageDescription
```

Additionally the *Location updates* background mode has to be enabled. When a *url* is provided during *startUpdates*, the *Background fetch* background mode is needed as well.

Public Methods

Return Type	Description
Future<PermissionStatus>	**hasPermission()** Returns a PermissionStatus to know if the backgroud location permission has been granted by the user.
Future<PermissionStatus>	**requestPermission()** Request the background location permission. Returns a PermissionStatus to know if the background location permission has been granted by the user.

Return Type	Description
Future<bool>	**serviceEnabled(LocationAccuracy accuracy = LocationAccuracy.high, int interval = 1000, double distanceFilter = 0)** Returns a boolean indicating whether the location services are enabled or not. The supplied parameters are only used on Android and offer a more fine-grained control over the requirements for the location services.
Future<bool>	**requestService(LocationAccuracy accuracy = LocationAccuracy.high, int interval = 1000, double distanceFilter = 0)** Shows an alert dialog asking the user to turn on the location services. On Android this dialog directly offers a button to turn on the location services, while on iOS the user is asked to go to the Settings and manually turn on the location services.
Future<bool>	**startUpdates(LocationAccuracy accuracy = LocationAccuracy.high, int interval = 1000, double distanceFilter = 0, String notificationTitle = , String notificationBody = , bool notificationClickable = false, String url = , Map<String, String> header = const {}, String label =)** Starts the location update service. The *accuracy* argument is controlling the precision of the *LocationData*. The *interval* (milliseconds) and *distanceFilter* (meters) are controlling how often the location is updated. *notificationTitle* and *notificationBody* can be used to customize the persistent notification on Android. When *notificationClickable* is true, tapping the notification on Android will open the app. The *label* is stored in the location cache alongside each recorded location and will be used to identify the locations which were captured during this run. When the locations should be sent to a server, the label is used to retrieve those locations of this run which are not yet uploaded. If a *url* is provided, the plugin will try to send the recorded locations as a JSON payload to the given URL. The *header*t property can be used to customize the headers of this reques (e.g. for authentication headers). If either the permission has not yet been granted or the location services are disabled, the function returns false. If the location updates were started sucessfully true is returned.
Future<bool>	**stopUpdates()** Stops the location updates.
Future<bool>	**updatesActive()** Returns a boolean to know if the location updates are active.
Stream<LocationData>	**onLocationChanged** Returns a stream of the user's location.
Stream<LocationData>	**getLocationUpdateStream(String label)** Returns a stream of the user's location, but only for updates with the specified label.
Future<LocationData>	**getLastLocation()** Retrieves the last known location from the location cache.
Future<List<LocationData»	**getLocations(List<String> labels, int n = -1)** Returns a list of LocationData corresponding to the given labels from the intenal location cache. The parameter n can be used to limit the number of locations. If n is omitted, all locations matching the given labels are returned.
Future<bool>	**deleteLocations(List<String> labels)** Deletes the locations with the supplied labels from the location cache.
Future<bool>	**clearLocationCache()** Clears the location cache.

App Schnittstelle

Diese Serverapplikation ist die Schnittstelle zwischen der *eTrax | rescue* webapp und der *eTrax | rescue* App. Sie implementiert die API Spezifikation die sich die App erwartet, wenn sie mit dem Server kommuniziert.

Installation

Die Applikation wurde mithilfe des Lumen Frameworks in php entwickelt. Bevor die Serverapplikation auf einem Server aufgesetzt werden kann muss composer installiert sein.

```
git clone https://github.com/etrax-rescue/etrax-app-interface.git
cd etrax-app-interface/

# Installieren der php Dependencies
composer install
```

Konfiguration

Die Konfiguration der Applikation wird mit einem Environment File (.env) vorgenommen. Um dieses einzurichten kann das Beispielfile *.env.example* kopiert und anschließend editiert werden.

```
cp .env.example .env
```

Folgende Variablen müssen angepasst werden, bevor das App Interface einsatzbereit ist:

Variable	Funktion
APP_KEY	Der Schlüssel der für die Datenbankverschlüsselung verwendet wird (liegt in secure/secret.php)
ETRAX_BASE_PATH	Pfad (relativ zum *public* Verzeichnis) zum eTrax | rescue server webroot
STATUS_UPDATE_URL	URL des BOS Interfaces
SECURE_PATH	Relativer Pfad zum *secure* Verzeichnis der eTrax | rescue Installation
TOKEN_MAX_AGE	Maximal zulässige Gültigkeitsdauer des Zugriffstokens in Sekunden
MAX_CACHE_TIME	Maximaler Zeitrahmen in Sekunden in dem Bildressourcen in der App gecached werden
DB_*	Konfiguration der Verbindung mit der Datenbank auf der auch die Daten der eTrax | rescue Webapp gespeichert sind