www.jates.org

Alkalmazott Műszaki és Pedagógiai tudományos folyóirat
szak- és mérnökképzési, műszaki és környezeti aspektusok

ISSN 2560-5429
8. évfolyam, 1. szám

doi: 10.24368/jates.v8i1.26
http://doi.org/10.24368/jates.v8i1.26

JATES
Journal of Applied Technical and Educational Sciences

Route-planning based on county tickets for android operating system

Melinda Mátrai[a]

[a]*University of Dunaújváros,Táncsics M. út 1/A, Dunaújváros 2400, Hungary, matrai.melinda@gmail.com*

Abstract

It is not easy to make a route planner software, because so many developers have already made similar programs, so many have high expectations about them. This article presents an application, which is different from those, it is specially designed for Hungarian highways, and their usage possibilities. Because in Hungary, it is possible to buy county tickets, which are much cheaper than those tickets, which are valid for the whole country. The paper summarize the structure and implementation of the application.

Keywords: own development; route planner; Hungarian county tickets; Android

Meglévő autópálya-matricáktól függő útvonaltervezés Android operációs rendszerre

Mátrai Melinda[a]

[a]*Dunaújvárosi Egyetem, Táncsics M. út 1/A, Dunaújváros 2400, matrai.melinda@gmail.com*

Absztrakt

Útvonaltervező szoftvert nem egyszerű írni, sok fejlesztő készített már hasonló programot, így sokaknak van alapelvárásuk ezekkel szemben. Ez a cikk egy olyan alkalmazást mutat be, amely merőben más, mint a többi abban a tekintetben, hogy kifejezetten a magyar autópályákra lett tervezve, az itteni autópálya használati lehetőségeket veszi figyelembe. Magyarországon ugyanis egyedülálló módon lehet megyei matricákat is vásárolni, melyekhez jóval olcsóbban lehet hozzájutni, mint az egész ország területére érvényes matricákhoz. A cikk összefoglalja az alkalmazás felépítését, megvalósítását.

Kulcsszavak: saját fejlesztés; útvonaltervező; megyei matricák; Android

1. Bevezetés

A mai világban végbement technológiai fejlődésnek köszönhetően elengedhetetlen használati kellék lett a mobiltelefon. Az utcán közlekedve gyakori látványt nyújt a gyalogosok tömegénél kézben levő telefon, sőt, ha megpillantunk egy-két autóban utazót, akkor többnyire náluk is láthatunk felfüggesztve egy készüléket, amit többnyire navigáció gyanánt használnak menet közben. Az IDC Quarterly Mobile Phone Tracker szerint a készülékgyártók 2017 második negyedévében 341,6 millió (Smartphone Volumes Decline Slightly in the Second Quarter of 2017 Amid Anticipation of Strong Second Half Product Launches, 2017) okostelefont szállítottak le, mely eszközök között piacvezető operációs rendszer az Android. A 2017-es májusi statisztika szerint ezen év első negyedévében az eladott készülékek 85%-án ilyen rendszer fut (IDC: Smartphone OS Market Share, 2017) (1. táblázat).

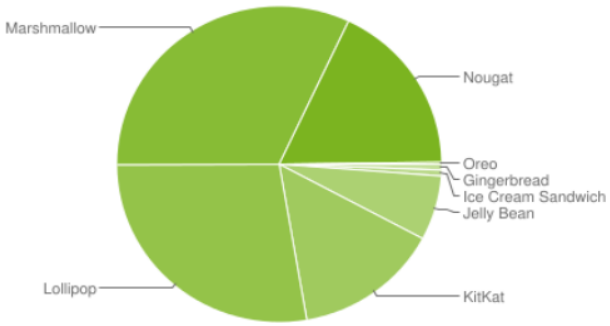
1. táblázat Az okostelefonok operációs rendszerekre bontott világpiaci részesedése (forrása: https://www.idc.com/promo/smartphone-market-share/os)

Period	Android	iOS	Windows Phone	Others
2016Q1	83.4%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%
2016Q3	86.8%	12.5%	0.3%	0.4%
2016Q4	81.4%	18.2%	0.2%	0.2%
2017Q1	85.0%	14.7%	0.1%	0.1%

Source: IDC, May 2017

Ez nem meglepő, hiszen az Androidos készülékek széles kínálata közül lehet kiválasztani a számunkra legmegfelelőbb példányt, mindenki megtalálhatja a számára legalkalmasabb árkategóriában lévő eszközt. Az Android Developers 2017. október 2-ai felméréséből kiderül, hogy a jelenlegi legelterjedtebb Android disztribúció a 6.0-ás verzió, ami a Marshmallow fantázianevet viseli, amely a készülékek 32%-án érhető el, és 2015-ben jelent meg. Nem sokkal van lemaradva ettől az 5.1-es verziójú Lollipop (21%, megjelenés: 2015), és még számottevő mennyiségben van jelen a 7.0-ás Nougat (15,8%, megjelenés: 2016), illetve a 4.4-es KitKat-tel rendelkező mobilok (14,5%, 2013-ban jelent meg). (Dashboards, 2018) (1. ábra)

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.6%
4.1.x	Jelly Bean	16	2.3%
4.2.x		17	3.3%
4.3		18	1.0%
4.4	KitKat	19	14.5%
5.0	Lollipop	21	6.7%
5.1		22	21.0%
6.0	Marshmallow	23	32.0%
7.0	Nougat	24	15.8%
7.1		25	2.0%
8.0	Oreo	26	0.2%



Data collected during a 7-day period ending on October 2, 2017.
Any versions with less than 0.1% distribution are not shown.

1. ábra Android verziók, kódnevük, API szintjük, eloszlásuk
(https://developer.android.com/about/dashboards/index.html)

A jelenlegi legfrissebb, 2017-ben megjelent disztribúció az Oreo nevet viseli, ez a 8.0-ás verzió. Megállapítható, hogy az Android rohamosan fejlődik, megújul, alkalmazkodik, és hosszú távra váltott jegyet a technológiai fejlődés megállíthatatlan, szélsebesen robogó vonatára.

Alapvetően, ha egy átlagos felnőtt megvesz egy ilyen okostelefont, ha nem is használja ki a benne rejlő lehetőségeket, de esetleg az alapfunkcióin felül egy navigációs alkalmazást nagy valószínűséggel használni fog. A jelenleg elérhető hasonló applikációk nem magyar kézből kerültek ki, így nem mondható el, hogy a magyar utakra lennének tervezve. Ezért ebben a témakörben nyitott lehetőségek rejlenek, amiket érdemes megfogni.

Sajnos nem elérhető olyan alkalmazás, amely adott megyére vonatkozó matricával rendelkezők számára megmondja, mikor kell elhagyni az autópályát. Így az útdíj.hu oldalról kell megnézni, melyik csomópontnál kell letérni, és a navigáció során erre is nagymértékű figyelmet kellett fordítani. Erre a problémára született meg a R-út-R („Rúter”), egy egyszerűen, egyértelműen használható szoftver a magyar utakra. Android operációs rendszerre készült, ezen belül a minimális igénye a KitKat, és a legújabb (Oreo) disztribúció lett a célplatform.

2. Specifikáció

A program telepítésekor létrejön két beépített adatbázis: az egyik a VÉDA szupertraffipaxok koordinátáit, a másik pedig az autópálya-matrica típusok jellemzőit tartalmazza. A szoftver elindulásakor, ha szükséges, engedélyeket kér a felhasználótól a használni kívánt Android szolgáltatások végett. Csak akkor indul el, ha megkap minden jogot, hiszen ezek megléte nélkülözhetetlen. Továbbá jelzi, ha esetleg nem kapcsolódik valamilyen hálózathoz (Wi-Fi vagy mobilhálózat), illetve felhívja a figyelmet arra is, hogy a GPS-t be kell kapcsolni.

A menübe belépve megtaláljuk a matricák kezelése, járművek kezelése, előzmények, valamint a beállítások menüpontot. Amikor a felhasználó rögzíteni kíván egy matricát, meg kell adnia annak típusát (10 napos, havi, éves, megyei), ha megyei, akkor melyik megyére szól, valamint az érvényességének kezdetét. Az alkalmazás ezen adatok megadása után kikalkulálja a lejárat idejét. Megadható továbbá, hogy kér-e értesítést a felhasználó, ha elkövetkezik a lejárati idő, illetve az is, hogy mennyivel előbb jelenjen meg a figyelmeztetés (1 héttel, 5-4-3-2-1 nappal). Természetesen, ha lejár a matrica, akkor is küld figyelmeztetést az alkalmazás. Az elmentett adat a saját matricák adatbázisába kerül.

A térképnézet induláskor a készülék aktuális pozícióját mutatja. Található itt egy szövegmező, ahol megadható a célállomás. A kiindulási pont a készülék aktuális pozíciója, így ezt nem kell megadni. Ha a felhasználó elnavigál az aktuális térképnézetről, akkor könnyedén vissza tud térni a saját pozíciójára, ha megnyomja a jobb alsó sarokban található saját pozíció megjelenítésére alkalmas gombot. Ha a szövegmezőbe elkezdi írni a kívánt célállomást, akkor a szoftver kiadja az összes lehetséges találatot egy listában, ahol könnyedén kiválasztható az úti cél. Ha már vannak előzmények, akkor azok közül is lehet keresni. Ezután az „Tervezés” gombra kattintva kikalkulálja a program a legoptimálisabb útvonalat úgy, hogy figyelembe veszi az adatbázisban rögzített, érvényes vignettákat. Ha talál olyan optimális utat, ahol nem kell/kellenek a saját matricák és semmilyen másik sem, akkor azt ajánlja fel. Amint rákattintottunk a „Tervezés” gombra, kirajzolódik az útvonal a térképre. Ha mégsem a megadott célba akar eljutni, akkor a készüléke „Vissza” gombjával tud visszajutni a tervezéshez.

Miután végbement a tervezés, az „Indulás” gombra kattintva elindul a navigáció, melynek során folyamatos jelentéseket kap az alkalmazás az aktuális közlekedési viszonyokról úgy, hogy kirajzolja azokat a térképre. Navigálás közben a térképen láthatók lesznek a VÉDA-t jelző markerek is az út folyamán, illetve, ha a felhasználó engedélyezte a figyelmeztetést, akkor kiírja, ha VÉDA közeledik.

3. A fejlesztőkörnyezet bemutatása

Az Android Studio egy integrált fejlesztőkörnyezet, amely 2013-ban jelent meg. Letölthető Windows-ra, Mac OS X-re, és Linux-ra is. Én a fejlesztést Windows operációs rendszeren hajtottam végre, melynél előírás, hogy rendelkezzen a gépem minimum 3 GB RAM-mal, de 8 GB RAM az ajánlott, illetve dedikálni kell minimum 1 GB RAM-ot az Android Emulator számára. Továbbá minimum 2 GB elérhető lemezkapacitást kér, de 4 GB ajánlott számára. Megkötést tesz arra is, hogy minimum 1280 x 800-as felbontással rendelkezzen az adott képernyő.

Az Android Studio a 3.0-ás verzióig a Java programozási nyelvet használta elsődlegesen, mindamellett fejleszthetünk C++ támogatással is, és 2017. októberétől már a Kotlin nyelvet is támogatja, amelyet a JetBrains fejlesztői alkottak meg. Ennek a cégnek az IntelliJ IDEA szoftverére épült a Studio. (Why JetBrains needs Kotlin, 2018) (IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains, 2018)

A Studio Grandle alapú build támogatást használ, ami egy olyan projektépítő eszköz, amely lehetővé teszi a projektkonfiguráció deklarációját a hagyományos XML formátum helyett. Ezt több alprojektes projekt összeállítására tervezték, ami esetenként óriásira nőhet.

Az Android Studio elindulásakor meg kell határozni a készülő alkalmazás és a vállalati domain nevet, valamint a projekt helyét is lokalizálni kell. A vállalati domain, és az alkalmazás neve adja a csomagnevet, ami végigkíséri a projektet, összetartja azt. Továbbá megadható a C++, és/vagy Kotlin támogatás. Innen továbblépve megjelölhető, hogy telefonra és táblagépre, vagy kiegészítőre, esetleg TV-re, vagy autóra, esetleg úgynevezett Android Things-re készül az alkalmazás. Meg lehet adni a minimum SDK-t, amire a fejlesztés készül, vagyis azt, hogy melyik legyen az az Android verzió, amely még futtatni képes a programot. [12] Itt felajánl egy lehetőséget, miszerint, ha az általa felkínált célplatform kerül kiválasztásra, akkor a készülékek nagy részén futni fog az alkalmazás. Továbblépve, kiválasztható az alap Activity, amihez kódsablont is legenerál a fejlesztőkörnyezet. Ezzel egy kezdőlökés születik, ami a szoftver kinézetét érinti, hiszen ez lesz a kiválasztott alapkinézete az alkalmazásnak. Az utolsó lépésben a kiválasztott Activity, valamint az elrendezés neve adható meg, amivel a későbbiekben hivatkozni lehet rájuk.

Rengeteg előre elkészített csomagot lehet belefordítani a fejlesztés alatt álló projektbe, amelyek megkönnyítik a munkát, hiszen ezek mások által már letesztelt, működő programrészek. A fejlesztőkörnyezet előre elkészített dizájnelemeket is tartalmaz, amelyeket tovább lehet formázni, amennyiben szükséges.

Ha fejlesztés közben felmerülnek kérdéseink a Java-val, vagy a fejlesztőkörnyezettel kapocslatban, a https://developer.android.com oldalon rengeteg információ áll rendelkezésre, megannyi lépésről-lépésre haladó példakóddal.

4. A felhasznált technológiák bemutatása

4.1. HERE Android SDK

A R-út-R első lélegzetvételei Google Maps API-val történtek, de nagyobb utánajárás után kiderült, hogy a Google nem engedélyezi, ha az általa elkészített komponensekből valaki összeállítson egy hasonló tulajdonságokkal rendelkező applikációt, mint a Google Maps. [14] Így hamar más alternatíva után kellett néznem. Szerencsére nem tartott sokáig ez a procedúra, mivel ráakadtam a HERE Android SDK-ra.

A HERE Andoid SDK rengeteg programozási interfészt biztosít a fejlesztő számára. Az SDK közreműködésével olyan helymeghatározási szerepekben bővelkedhet, mint az útvonaltervezés, interaktív térképek, és globális helykeresés. Az SDK tartalmaz egy szofisztikált motort a térképadatok és a kikalkulált útvonal összetársításához.

Természetesen, a HERE-nek is megvannak a rendszerkövetelményei, mégpedig meghatározza a 4.1.x-es verziójú Jelly Bean-t, mint minimum Android verzió, az Android Studio 2.3.2-es, vagy frissebb változatát kell használni, a minimum RAM mennyisége 60 MB az egyszerűbb alkalmazások esetén, viszont a navigációs szoftverek ennél többet igényelnek. Minimum 25 MB szabad területet vár applikációnként, és további 50 MB-ot a térképadatok tárolása végett. És persze internetkapcsolat is szükséges a legfrissebb térképadatok letöltéséhez. A Google ajánlása szerint x86 alapú emulátort kell létrehozni, mert ez tízszer gyorsabb, mint az ARM-alapú. (System Requirements - HERE Android SDK., 2018)

A HERE Android SDK sokféle alkalmazásprogramozási interfészt kínál, mint például:

- HERE Geocoder Autocompletion API
- Geocoder API
- Routing API
- Traffic API
- Maps API for JavaScript
- Venue Maps API
- Places API
- Weather API
- Public Transit API
- Positioning API

Ezek közül többet is magába foglal a Premium SDK. [18]

4.2. JSON

A JSON (JavaScript Object Notation) az adatcsere és az adattárolás szintaxisa. Olyan szöveges fájl, amely JavaScript objektumokat reprezentál. [19] Amikor adatcsere megy végbe a kliens és a szerver között, akkor az adat csak szöveg lehet. Minden JavaScript objektumot át lehet alakítani JSON formátummá, és így lehet a keletkezett szöveges fájlt a szervernek továbbítani. A folyamat visszafelé is működik: ha a kliens kap egy JSON fájlt, akkor azt vissza tudja alakítani JavaScript objektummá. Ezáltal elkerülhető a JavaScript objektumok komplikált elemzése és lefordítása. [20]

A R-út-R JSON formátumban kapja meg a szervertől a kérésre érkezett választ. Ezek mindig ugyanúgy épülnek fel, tehát hasonló metódusokkal lehet őket kezelni, feldolgozni, majd kinyerni a szükséges információkat.

4.3. XML

Az XML (eXtensible Markup Language) egy szoftver- és hardverfüggetlen jelölőnyelv, amely adatok tárolására és szállítására szolgál. Hasonló a felépítése a HTML-hez, ami szintén egy leíró nyelv, de az XML az adatokra koncentrál, a HTML pedig a megjelenítésre. [21] További különbség az, hogy a HTML esetén előre definiált jelölők vannak, míg az XML-nél bármi előfordulhat. [22] Az Android Studio ilyen formátumban tárolja konfigurációs állományait, és a program felületeit.

4.4. SQLite

Az SQLite egy nyílt forráskódú, önálló, kisméretű relációs adatbáziskezelő rendszer és adatbázismotor. Nem egy különálló folyamat, mint a kliens-szerver kommunikáció, hanem a program részét alkotja, egy programkönyvtár benne. Az adatbázis egy platformfüggetlen fájlban tárolódik az eszközön, így innen gyorsabban el lehet érni a kívánt adatokat, mintha azokat egy távoli szerver elérésén keresztül kéne kinyerni. [23] Az Android Studio SQLite API-t használ, ami csomagként beimportálható. Ezzel a módszerrel tárolom az adatokat a telefon belső tárhelyén, hogy gyorsan, nehézségek nélkül el lehessen érni a szükséges adatokat.

5. Tervezés

Egy program fejlesztésénél az egyik legfontosabb fázis a tervezés. Ekkor a részletes specifikációt követően kialakul egy még pontosabb összkép a születő szoftverről. Elsődleges

szempont az, hogy könnyen használható, egyszerűen áttekinthető legyen. Az egyes programelemeket megfelelően el kell határolni egymástól, a gomboknak kényelmesen megnyomhatónak kell lenniük, de nem lehetnek túl nagyok, az esztétikailag nem jó. A felhasználónak egyértelművé kell tenni a használatot, minden résznek adnia kell, mi a feladata.

A kiindulási képernyőn a térképnek kell lennie, mert ez adja a fő funkcionalitást, ezen pedig elegendő, ha egy szövegmező, egy kereső gomb, és a saját pozíció megtalálására alkalmas gomb található. A menüt, ami egymás alatt felsorolja a menüpontokat, a képernyő bal oldaláról lehet behúzni. Itt is figyelni kell a betűméretre, az egyértelmű menüpontokra, amelyeknek elég távol kell lenniük egymástól, ezzel csökkentve a mellényomás lehetőségét.

Adatbázis

5.1. Védák

Előre legenerált adattábla, a felhasználónak nem adok jogot, hogy szerkessze. Egyszerű szerkezetű, nem szükséges kettőnél több attribútumot eltárolni benne, hiszen csak a VÉDA-k koordinátáira van szükség.

5.2. Megyei matricák területi érvényessége

Minden megyei matricának van területi határa, amelyek többnyire fedik egymást. Számos oldalon elérhető ezen határok leírása úgy, hogy meg van határozva az első, illetve az utolsó érvényességi csomópont, ennek neve, és az útvonal neve.

5.3. Saját matricák

A felhasználó által rögzített, törölt, illetve a lejárt vignettákat lehet megállapítani belőle.

5.4. Előzmények

Amikor a felhasználó az applikáció szövegmezőjébe begépel valamit, majd a legördülő listából kiválasztja a megfelelő célállomást, és megérinti a „Tervezés” gombot, ebbe a táblába bekerül egy új sor.

5.5. Rendszámok

Ha valaki több gépjárművel rendelkezik, akkor rögzíteni tudja azok rendszámait, hogy utána egyszerűen ki tudja választani, melyikhez vásárolta a matricát.

6. Megvalósítás

6.1. Android Studio telepítése, beüzemelése

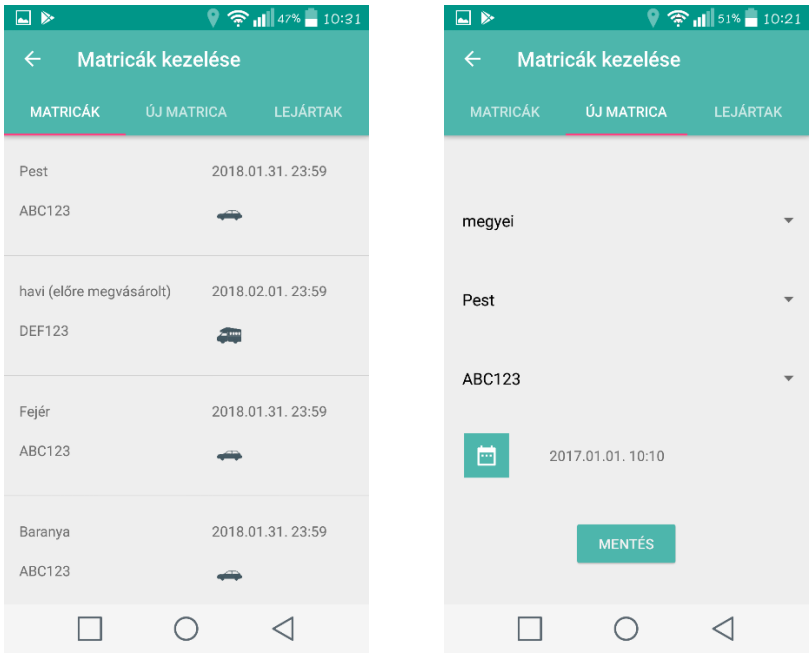
A megvalósítás első lépéseként telepítenem kellett az Android Studio-t, majd konfigurálnom kellett azt. Már a telepítés során is akadtak problémák, mivel a program nem képes olyan könyvtárakat használni, amelyek nevében az angol ábécétől eltérő karakter van, illetve a white space-t sem kezeli megfelelően. [24] Ezekkel csak az a probléma, hogy a Windows előszeretettel használja a felhasználó teljes nevét a gép beüzemeléséhez, így az alapértelmezett Users/felhasználónév könyvtár legtöbbször nem használható az SDK telepítési helyeként.

Beüzemelésnél is adódott probléma az Android Studio-val, amikor ugyanis telepítettem SDK-kat, hibát írt ki a folyamat legvégén. Az interneten kutatva megtaláltam, hogy meg kell növelnem a heap méretét. [25] Amikor feltelepítettem a gépemre, az Android Studio általa ajánlott minden értéket elfogadtam, de még a felhasználható RAM mennyiségének megnövelése után is hiba keletkezett.

Ezek után az emulátor beüzemelése következett. Az első ütemben, amikor még Google-termékekkel dolgoztam, úgy állítottam be az emulált eszközöket, ahogy azt a hivatalos dokumentáció ajánlotta. Viszont a HERE nem kompatibilis az x86-os eszközökkel, így ezen a beállításon változtatnom kellett. Sajnos, ez a folyamat egyébként sem szélsebes, de ezzel az új beállítással végképp megnőtt a tesztidő, az emulálási folyamat majdnem tízszer lassabb lett.

6.2. A létrejött programelemek bemutatása

Matricák kezelése



2. ábra (a) Matricák fül (b) Új matrica fül

Ebben a menüpontban a matricákat tudjuk kezelni (2. ábra). Tabos elrendezést kapott, az első fülön a meglévő matricákat, a másodikon új felvitelt lehet kezdeményezni, míg a harmadikon a lejártakat lehet megtekinteni. Az új matricánál egy legördülő menüből meg kell adni a típusát (10 napos, heti, havi, megyei), majd, ha az került kiválasztásra, hogy megyei típusú, akkor azt is meg kell adni egy legördülő listából, melyik megyére szól, illetve, az érvényesség kezdetét is, amit dátumválasztó segítségével lehet kiválasztani. Ha 10 napos, havi vagy éves került kiválasztásra, akkor csak az érvényesség kezdete mező fog látszódni, mert csak ez az információ releváns ekkor. Ha a felhasználónak több gépjárműve van, akkor azt is kiválaszthatja, melyik rendszámhoz vásárolta az adott vignettát (a rendszámot a Járművek kezelése menüpontban viheti fel). A „Mentés” gombra kattintva létrejön egy adatbázis-bejegyzés a „Saját matricák” táblában. Ettől a ponttól kezdve a „Meglévő matricák” fül alatt megjelenik ez a rekord. Ha esetleg több megyei vignetta kerül be, akkor ezeket jól elkülönítve ábrázolja az alkalmazás.

Törölni úgy lehet, ha hosszan megérintjük a kiválasztott elemet, majd megjelenik egy párbeszéd ablak, amely megkérdezi, biztosan törölni akarjuk a tételt. Amelyik matrica lejár, az automatikusan átkerül a „Lejárt matricák” fül alá, itt is időrendi sorrendben látható az összes eddigi rekord, a legfrissebbel a tetején.

Járművek kezelése

Felvihető a rendszerbe számos rendszám is, arra az esetre, ha a felhasználó több gépjárművel rendelkezik. Ezeket itt lehet kezelni (rögzítés, törlés). Rögzítéskor rá kell kattintani a megjelenő oldal alján található gombra, amin egy + szimbólum látható. Meg kell adni a rendszámot, fűzhető hozzá egy rövid megjegyzés, illetve kiválasztható egy, a gépjármű típusára jellemző ikon. Amint rögzítünk egy járművet, láthatóvá válik a törlés gomb, amelyiken egy X szerepel. Törléshez ki kell választani egy tételt, majd meg kell nyomni a törlés gombot.

Előzmények

A felhasználó által beírt címeket tartalmazza egy keresőmezővel. Innen esetleg gyorsabban meg tudja határozni a célállomás címét, illetve, ha nem emlékszik pontosan egy előzőleg keresettre, akkor itt megtalálja azt. Az előzmények listájáról bármelyik elem kitörölhető.

Beállítások

Itt lehet megadni, hogy jelezzen-e az applikáció, ha egy VÉDA felé közelít a gépjármű, illetve azt, hogy ezt mekkora távolságon belül tegye meg. A jelzés kérését egy csúszkával kapcsolhatja be a felhasználó, ami az egész alkalmazásra jellemző beviteli mód az igaz/hamis esetekre. Ha

bekapcsolt állapotban van, akkor elérhetővé válik egy legördülő lista, amiből kiválasztható, mekkora távolságon belül küldjön figyelmeztetést (1000 m, 500 m, 300 m, 200 m).

Következő lehetőségként arról lehet nyilatkozni, küldjön-e értesítést a szoftver, ha lejár a matrica, valamint a legördülő menüből megadható, mikor tegye (1 héttel, 5-4-3-2-1 nappal a lejárat előtt).

Minden módosított érték akkor tárolódik el, amikor a felhasználó megnyomja a „Mentés” gombot.

Térkép

A térkép a HERE saját tulajdona, használni csak úgy lehet, ha a licensz jogokért fizet a fejlesztő, illetve ingyenesen kipróbálható 90 napos próbaverzió keretében. A HERE által kínált navigációs szoftver ellenben díjmentesen használható bármilyen platformon. A vállalat régóta foglalkozik térképészettel, folyamatosan fejlesztik, bővítik a meglévő terméküket, hogy naprakész legyen.

A HERE térkép sok lehetőséget nyújt, többek között hozzá lehet adni markereket, amik olyan objektumok, amelyek egy adott koordinátát jelölnek a térképen. Így adtam hozzá a VÉDA-k koordinátáihoz egy saját készítésű jelölőt, amelyek kirajzolódnak, amikor a program elindul, és betölt a térkép. A térképen természetesen kirajzolható útvonal, de akár kör vagy egyéb alakzat is. Lehet éjszakai, lehet nappali módba állítani, és persze többfajta nézet közül lehet választani, mint például a műholdas vagy a hibrid. Be lehet állítani különböző dőlésszögeket, zoomolási szinteket is. Az aktuális forgalmat is reprezentálni lehet, bár ennek frissességéhez mindenképpen szükséges meglévő internetkapcsolat. Ekkor a forgalom függvényében különböző színű vonalakkal jelöli az egyes útszakaszokat:

- zöld: normál forgalom
- sárga: nagy
- piros: nagyon nagy
- fekete: blokkoló

Megannyi módon lehet állítani a térképet, érdemes kihasználni ezeket a lehetőségeket, hogy minél rugalmasabb, testre szabhatóbb, minél látványosabb térképet nyújtson a felhasználó számára.

7. A navigációig elvezető lépések

A program legfőbb funkciója a navigáció. Minden jogosultságot, ami ehhez szükséges, definiáltam az AndroidManifest.xml-ben. Végül ezeket engedélyeztetni kell a felhasználó által.

Ha nem engedélyezi ezeket, akkor a program nem indul el. A szükséges jogosultságok között szerepel a helymeghatározáshoz, az interneteléréshez (Wifi és mobilinternet egyaránt), az internetkapcsolat módosításához, a külső tároló írásához, és a telefonállapothoz való hozzáférés. Ezekután betöltődik a kezdőképernyő, amin megjelenik a térkép, egy keresőmező, ennek bal-, illetve jobboldalán a menü, és egy törlés gomb, valamint látható lesz egy saját helyzetet meghatározó gomb. Természetesen, a program elindulásakor eleve a készülék aktuális helyzete látható a térképen. A keresőmezőbe beírt hely lesz a célállomás, amit az automatikus szövegkiegészítő funkció segítségével lehet megadni. Ez a funkció úgy működik, hogy minden billentyűleütésnél elindul egy kérés a szerver felé, amelyben egy speciális URL szerepel. Ebben megtalálható a szerver címe, illetve azonosítani kell magam, hogy jogos-e a kérés, meg kell határozni a kérés nyelvét, illetve magát a keresendő karaktersorozatot. A szerver ezt feldolgozza, majd JSON formátumban válaszol vissza. A válaszban található több JSON objektum is, minden olyan helység, amelyre valamilyen módon ráilleszthető a keresett kifejezés. Természetesen, minél több karakter kerül a keresőmezőbe, annál kevesebb, annál pontosabb választási lehetőség adódik. Amint betűk kerülnek a keresőmezőbe, láthatóvá válik az útvonal tervezését elindító gomb, amely a megtervezett útvonal kirajzolása után a navigáció elindítását teszi lehetővé, a keresett célállomás pedig eltárolódik az „Előzmények” menüpontban. Innen gyorsabban vissza lehet keresni az eddigi célcímeket. Ha innen kiválasztásra kerül egy cím, akkor a kijelző visszaváltja a térképnézetre, a cím beíródik a keresőmezőbe, az adott hely koordinátái pedig beállítódnak utazási célpontnak. Az így kapott koordinátákat is a szerver felé indított kéréssel lehet meghatározni. Hasonló elven működik, mint az automatikus szókiegészítő, csak itt némileg módosul az URL, például ezeket a kéréseket egy másik szerver szolgálja ki. A szerver válasza szintén JSON formátumban érkezik vissza, majd ezt parse-olni kell, hogy az így elkészült objektumból könnyedén kinyerhető legyen a fontos információ. Így már van egy kiinduló, illetve egy célkoordináta is, ezután következhet az útvonal megtervezése.

8. Útvonaltervezés algoritmusa

A projekt célja, hogy olyan szoftver jöjjön létre, amely úgy tervez útvonalat, hogy figyelembe veszi a felhasználó megvásárolt és rögzített autópálya-matricáit. Az autópálya-matricáknak több típusa van: lehet hetit, havit, éveset vagy megyeit vásárolni. A megyei matricának éves díja van, egészen addig éri meg, amíg az ember fixen közlekedik ugyanazon 8 megye között.

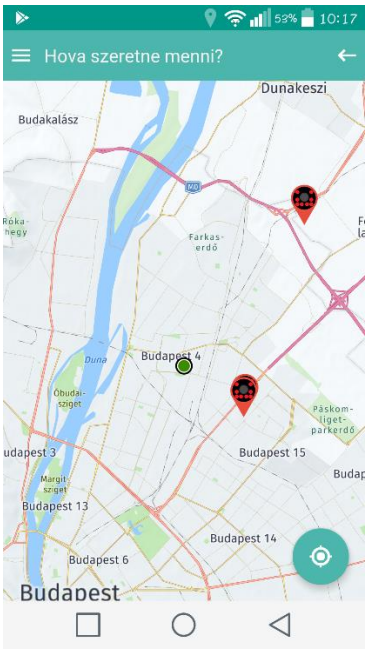
A megvásárolt matricákat a saját matricák adattáblában tárolom el, csak azokat gyűjtöm ki vizsgálat gyanánt, amelyik nem járt le, illetve nem törölt állapotú. Ha a kigyűjtött matrica heti, havi vagy éves, akkor nincs egyéb teendő, kikalkulálom a két bemenő koordináta között úgy az

útvonalat, hogy használhatja az autópályákat, illetve fizetős útszakaszokat. Amennyiben megyei matricája van az illetőnek, akkor gondosan szemügyre kell venni a lehetőségeit. Lehet egy, vagy akár több megyei matricája ugyanarra a gépjárműre. Az ilyen típusú vignetták esetére az általam létrehozott adatbázisban eltároltam az adott megye fizetős útvonalára jellemző szakaszkezdő-, illetve végcsomópont koordinátáját, tehát azt a pontot, ahol fel tud már hajtani az autópályára, továbbá azt a csomópontot, ahol el kell hagynia azt.

Miután kigyűjtöttem a megfelelő matricák adatait egy tömbbe, kikalkulálom a lehető leggyorsabb útvonalat az adott kezdő- és végkoordináta között. Megvizsgálom a kapott útvonalat, kigyűjtöm belőle a fizetős útszakaszokat egy tömbbe, azon koordinátákkal, ahol útszakaszváltás történik. Megnézem, mennyi eleme van ennek a tömbnek, vagyis hány különböző fizetős útszakaszon akar végigmenni. Kiveszem a tömbből az első elemet, a hozzátartozó koordinátát visszaalakítva címmé megkapom, melyik megyében van. Ekkor ellenőrzöm, van-e ilyen megyei matrica az érvényesek között. Ha van, akkor az az útszakasz használható. Ha nincs több ilyen típusú vignetta, akkor nincs is más dolog hátra. Megvan a leggyorsabb útvonal, és ez használható, elindul a navigáció. Viszont, ha egy vignetta van, de a felhasználó több megyén is keresztül akar menni, akkor az utolsó használható koordináta mentén levezeti az autópályáról a felhasználót, és az autópálya használata nélkül kikalkulált második útszakaszon folytathatja tovább az útját.

Abban az esetben, ha a „saját matricáim” tömbnek több eleme van, és több megyén is keresztül akar utazni a felhasználó, akkor több vizsgálatra van szükség, több útvonal-kalkuláció is történhet a végső útvonal meghatározásáig.

9. A program működése



3. ábra Az alkalmazás kezdőképernyője

A program elindulásakor a térképnézet jelenik meg (3. ábra). Látható a keresősáv, ahol megadhatjuk az úticélunk, továbbá megtalálható a saját pozíciónk megjelenítésére alkalmas gomb, valamint a VÉDA-kat jelző ikonok és a menügomb. Betöltés után az aktuális pozíciónkra áll be a térkép, egy külön markerrel jelöli meg a helyzetünket. Ha a keresőmezőbe elkezdünk gépelni, automatikusan megjelenít találatokat a program, amelyek közül kiválaszthatjuk a számunkra megfelelő célpontot. A kiválasztott úticélok bekerülnek az Előzmények menüpont alá.

Első indításkor első lépésnek gépjárművet kell rögzíteni a rendszerben, ezután lehet matricát. A matricához hozzá kell rendelni egy gépjárművet, ezért ez a sorrend. A főmenüben, a Rendszám résznél lehet kiválasztani, melyik gépjárművel kívánjuk megkezdeni az utazást, és az ahhoz tartozó matricával vagy matricákkal fogja az alkalmazás megtervezni az útvonalat.

A célállomás megadása után a Saját pozíció gomb helyén a navigáció elindítására szolgáló gomb jelenik meg. Navigáció közben pedig a szintén itt megjelenő Stop gomb segítségével szakíthatjuk meg a folyamatot.

Navigálásnál egyértelmű utasításokat mutat, hogyan kell végighaladnunk az útvonalon, hogy elérhessük a célállomást. Ha beállítottuk, hogy jelezzen, mikor VÉDA közeledik, akkor a beállított távolságon belül megjelenik egy távolságot visszaszámláló felirat, hogy jelezze, hány méter múlva érünk el a szupertraffipaxig. A navigálást bármikor le lehet állítani, ekkor a saját pozíciónkat jeleníti meg a kijelzőn, és újabb célállomást lehet megadni.

10. Összefoglalás

A fejlesztés sikeresen végbement, a produktum egy működőképes alkalmazás lett. Számos lehetőség adott még a továbbfejlesztésére. Többek között, a teljes offline működést valósítanám meg, illetve, az útvonal újratervezése is praktikus funkció az ilyen típusú programok esetén. Ha voltak is nehézségek a munka folyamán, a végeredmény egy jól működő, a specifikációnak megfelelő, piacképes szoftver lett.

11. Köszönetnyilvánítás

A projekt Új Nemzeti Kiválóság Program támogatásával valósult meg.

Irodalomjegyzék

Smartphone Volumes Decline Slightly in the Second Quarter of 2017 Amid Anticipation of Strong Second Half Product Launches, According to IDC. (n.d.). Retrieved April 04, 2018, from https://www.idc.com/getdoc.jsp?containerId=prUS42935817

IDC: Smartphone OS Market Share. (n.d.). Retrieved April 04, 2018, from https://www.idc.com/promo/smartphone-market-share/os

Dashboards. (2018, February 05). Retrieved April 04, 2018, from https://developer.android.com/about/dashboards/index.html

Why JetBrains needs Kotlin. (2015, May 29). Retrieved April 04, 2018, from https://blog.jetbrains.com/kotlin/2011/08/why-jetbrains-needs-kotlin/

IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains. (n.d.). Retrieved April 04, 2018, from https://www.jetbrains.com/idea/?fromMenu

System Requirements - HERE Android SDK. (n.d.). Retrieved April 04, 2018, from https://developer.here.com/documentation/android-premium/topics/system-requirements.html