Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Rok Volarič

Mobilna podpora iskanju parkirišč v Ljubljani

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Dejan Lavbič

Ljubljana 2012

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00054/2012
Datum: 05.09.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **ROK VOLARIČ**

Naslov: **MOBILNA PODPORA ISKANJU PARKIRIŠČ V LJUBLJANI**
MOBILE PARKING ASSISTANT IN LJUBLJANA

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

Iskanje prostega parkirnega mesta je za voznike v Ljubljani pogosto izziv, saj je potrebno upoštevati številne dejavnike. Glede na različne scenarije in situacije, v katerih se znajde uporabnik, je včasih pomembna oddaljenost od parkirišča, cena parkiranja na uro, število prostih mest ali celo vremenska napoved. V okviru diplomske naloge je potrebno raziskati omenjene scenarije in na podlagi javno dostopnih virov pripraviti mobilno podporo iskanju parkirišč v Ljubljani.

Mentor:

doc. dr. Dejan Lavbič

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Rok Volarič, z vpisno številko **63080104**, sem avtor diplomskega dela z naslovom:

Mobilna podpora iskanju parkirišč v Ljubljani

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Dejana Lavbiča,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 4. septembra 2012 Podpis avtorja:

Ob tej priliki se zahvaljujem mentorju doc. dr. Dejanu Lavbiču, za nasvete pri izdelavi diplomske naloge in sošolcem za čudovita študijska leta. Zahvaljujem se tudi staršem za podporo pri študiju.

Kazalo

Seznam uporabljenih kratic in simbolov

3D - Three Dimensional; tri dimenzionalno
ADT - Android Development Tools; razvojna orodja za Android
API - Application Programming Interface; programski vmesnik
AVD - Android Virtual Device; navidezna naprava za Android
DDMS - Dalvik Debug Monitor Server; strežnik za nadzor razhroščevanja
EII - Enterprise Information Integration; integracija v realnem času
ETL - Extract, Transform, Load; zajem, transformacija, prenos
GPS - Global Positioning System; globalni sistem za pozicioniranje
HD - High Definition; visoka ločljivost
IDC - International Data Corporation; podjetje, ki se ukvarja z analizo trga
LCD - Liquid Crystal Display; zaslon s tekočimi kristali
LPT - Ljubljanska parkirišča in tržnice d.o.o.
OS - Operating System; operacijski sistem
SD - Secure Digital; vrsta pomnilniških kartic
SDK - Software Development Kit; paket za razvoj programske opreme
SIM - Subscriber Identity Module; kartica, ki omogoča omrežno povezavo
SMS - Short Message Service; storitev za pošiljanje kratkih sporočil
XML - eXtensible Markup Language; razširljiv označevalni jezik

Povzetek

Z razvojem pametnih mobilnih naprav se je razširila uporaba mobilnih aplikacij, ki danes pokrivajo že prav vsa področja življenja in nam pomagajo pri reševanju vsakdanjih problemov. Cilj diplomske naloge je bil izdelati mobilno aplikacijo za mobilno platformo Android, s katero bi si uporabnik pomagal pri izbiri najprimernejšega parkirišča glede na trenutno lokacijo. Aplikacija vključuje vsa javna parkirišča v Ljubljani. V začetnem delu je predstavljen operacijski sistem Android in same komponente razvojnega okolja Android SDK v navezi z orodjem Eclipse. V nadaljevanju sledi predstavitev obstoječih rešitev na področju parkirišč in njihova primerjava z razvijajočo aplikacijo. Osrednji del pa je opis celotnega razvoja aplikacije od opredelitve ideje do končnega testiranja. Aplikacija izpiše seznam parkirišč glede na izbran pogoj iskanja. Uporabnik lahko izbira med oddaljenostjo od parkirišča, ceno in številom prostih mest. Na razvrščanje vpliva tudi vreme, saj se ob napovedi slabega vremena svetuje uporaba parkirnih hiš. Podatki o vremenu in zasedenosti se pridobijo iz spletnih virov. Trenutno lokacijo pridobi s pomočjo GPS sprejemnika. Aplikacija omogoča še prikaz lokacije posameznega parkirišča na zemljevidu. V zaključnem delu so opisane možnosti za razširitev funkcionalnosti aplikacije.

Ključne besede

Android, mobilna aplikacija, GPS, parkirišče, razvoj

Abstract

With the development of smartphones, the use of mobile applications today already covers a lot of everyday life areas and also help us to solve our everyday problems. The objective of the thesis was to create a mobile application for the Android mobile platform, which would help the user in selecting the most suitable parking space. The application includes all public parking places in Ljubljana. At the beginning the Android operating system is presented and the individual component of Android SDK development environment in conjunction with tool Eclipse. Then come presentation of the existing solutions in the area of parking places applications and their comparison with the developing application. The central part is the description of the entire application development from the definition of an idea to final testing. The application list parking places based on the selected search criteria. The user can choose between distance from the parking place, the price and the number of free locations. Selection of parking places is also impacted by the weather. In case of bad weather forecast the use of parking garages is advised. Information's on weather and parking places occupancy are obtained from internet sources. The current location is obtained by the help of GPS receiver. The application can also demonstrate the location of each parking place on the map. In the final section the options for extending application functionality are described.

Keywords

Android, mobile application, GPS, parking, development

Poglavje 1

Uvod

Mobilni telefon je tako kot druge elektronske naprave postal pomemben in nepogrešljivi del v našem vsakdanjem življenju. Danajšnji način življenja nas je prisilil, da smo vseskozi na dosegu informacij iz vsega sveta. Mobilni telefoni uporabniku omogočajo mobilnost in dostop do virov podatkov kjerkoli na svetu se nahajajo.

V današnjem času je razvoj mobilnih tehnologij iz leta v leto vedno hitrejši. Na to vpliva razvijanje novih omrežnih tehnologij, mobilnih naprav in same potrebe uporabnikov [2]. Razvoj omogoča tudi veliko število uporabnikov zaradi vedno večje dostopnosti mobilnih naprav. Trend širjenja se kaže tudi v Sloveniji, kjer večina prebivalstva uporablja mobilni telefon. Leta 2008 je bilo že 90 % uporabnikov mobilnih naprav [5].

Tehnološki napredek je omogočil, da je zmogljivost mobilnih naprav vedno bolj primerljiva z zmogljivostjo namiznih računalnikov. Že dalj časa se telefoni ne uporabljajo le za opravljanje klicev, temveč omogočajo dostop do svetovnega spleta in s tem do vira vseh informacij ter uporabo aplikacij, ki si jih lahko uporabnik po želji namesti na napravo. Telefon tako združuje funkcionalnosti več elektronskih naprav, kot so televizija, fotoaparat, kamera in računalnik. Govorimo lahko o tako imenovanih pametnih telefonih. V zadnjih letih so se še posebej razširili telefoni občutljivi na dotik. Največji proizvajalci mobilnih naprav na trgu v prvi polovici leta 2012 so Samsung,

Apple in Nokia. Skupaj pokrivajo kar 63 % svetovnega trga, od tega kar polovico obsega proizvajalec Samsung [6]. Z razvojem strojne opreme se sočasno razvijajo mobilni operacijski sistemi. Trenutno najbolj razširjen je operacijski sistem Android, ki mu sledijo še iOS, BlackBerry OS, Windows Mobile in ostali. Nekoč najbolj razširjen operacijski sistem Symbian je danes v zatonu in predstavlja le še manjši delež uporabe. Ta razvoj je tako omogočil različnim razvijalcem izdelavo mobilnih aplikacij in objavo le teh na spletnih storitvah kot so Apple App Store, Google Play, Windows Phone Marketplace in BlackBerry App World. Vse te aplikacije so v plačljivi ali zastonjski obliki dosegljive uporabniku za namestitev na mobilni napravi. Namestitev je za uporabnika hitra in zelo enostavna. Aplikacije danes pokrivajo prav vsa področja od prikaza vremenskih podatkov, lokacijskih storitev, do opravljanja bančnih storitev... Raziskava podjetja comScore je pokazala, da je uporaba mobilnih aplikacij postala bolj priljubljena od deskanja po spletu [7].

Mobilne aplikacije nam pomagajo pri reševanju vsakdanjih problemov. Eden takšnih primerov je mobilna podpora iskanju parkirišč. Voznikom je v večjih mestih velikokrat problem najti primerno parkirno mesto, še posebej ob delovnih dneh, ko se v službo v mesto vozijo ljudje, ki živijo v bližnji in daljni okolici. V veliko pomoč voznikom bi bil prikaz oddaljenosti od najbljižjih parkirišč ter njihovi zasedenosti. S tem bi se skrajšal čas, ki ga voznik porabi za iskanje prostega parkirnega mesta. Prednost mobilne naprave je, da ne zaseda veliko prostora, zato jo imamo vedno pri roki. Prav tako je hitra za uporabo. Te lastnosti so ključne pri voznikih. V Sloveniji je takšna podpora iskanja parkirišč še slabo razvita.

V diplomskem delu je predstavljen razvoj mobilne aplikacije za platformo Android. Najprej je krajša predstavitev samega operacijskega sistema Android in razvojno okolje Android SDK. Aplikacija pridobiva podatke iz različnih spletnih strani, zato je opisana tudi integracija podatkov. Glavni del diplomske naloge je razvoj aplikacije Parkiraj, ki uporabniku pomaga pri iskanju primernejšega parkirišča glede na trenutno lokcijo. Najprej so predstavljene že obstoječe aplikacije na trgu na tem področju, nato sledi opis

razvojnega cikla aplikacije, od načrtovanja ideje do izdelave uporabniškega vmesnika in končnega testiranja. V zadnjem delu so predstavljene zaključne ugotovitve in možne izboljšave aplikacije.

Poglavje 2

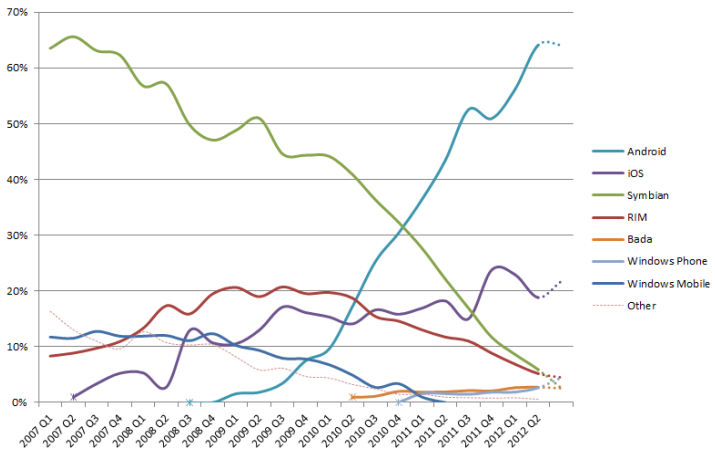
Mobilna platforma Android

2.1 Splošno o Androidu

Android je odprtokodni operacijski sistem za mobilne naprave, ki temelji na operacijskem sistemu Linux. Prvotno ga je razvilo podjetje Android Inc., ki ga je finančno podpiralo podjetje Google. Leta 2005 je podjetje Google odkupilo platformo Android od prvotnih razvijalcev v želji, da bi postal odprtokoden in prosto dostopen za vse, ki bi ga želeli uporabljati. V ta namen je bila objavljena izvorna koda pod licenco za brezplačno programsko opremo Apache License [3].

Skozi razvoj je Android doživel kar nekaj izboljšav. Trenutna verzija platforme je 4.1 pod kodnim imenom Jelly Bean, vendar je najbolj razširjena verzija med uporabniki še vedno Android 2.3.3 pod kodnim imenom Gingerbread. Poleg same verzije platforme je pomembna tudi verzija ogrodja API (angl. Application Programming Interface), ki določa delovanje aplikacij na samih napravah. Funkcije, ki jih podpira Android so shranjevanje podatkov, povezljivost, pošiljanje sporočil, podpora medijskim storitvam, spletni brskalnik, večopravilnost in večdotičnost zaslona, različna strojna oprema (GPS, kamera, digitalni kompas, merilnik pospeška...) in omogoča privezovanje (deljenje internetne povezave z drugimi napravami).

Operacijski sistem Android dosega vedno večji uspeh na trgu, kot pri-



Slika 2.1: Primerjava razširjenosti mobilnih operacijskih sistemov na svetovnem trgu po podatkih podjetja Gartner in IDC (angl. International Data Corporation) [16]. Po napovedi IDC naj bi tržni delež Androida naraščal skozi vse leto 2012, potem pa bi začel postopoma upadati vse do leta 2016.

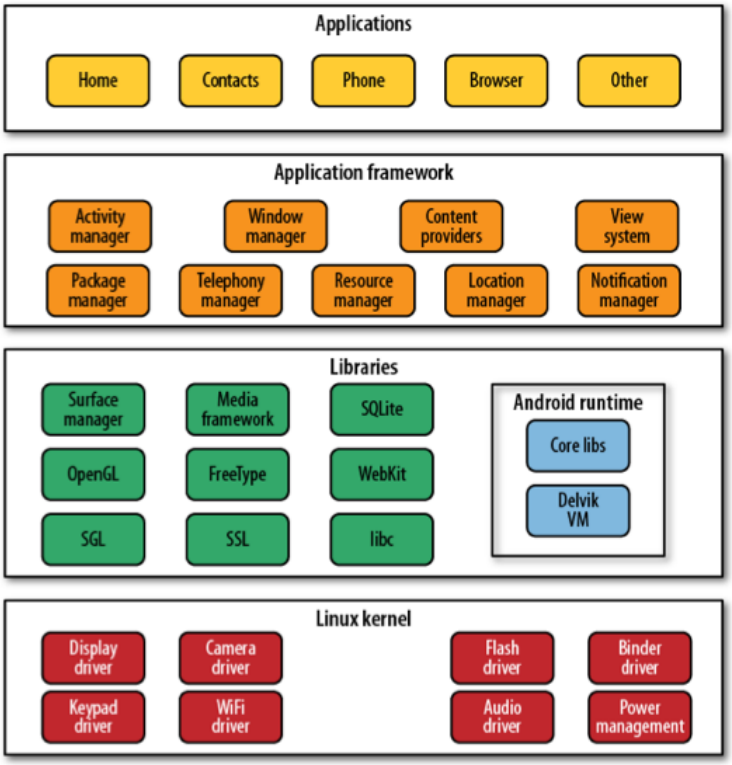
kazuje slika 2.1. K takšnem vzponu je pripomogel velik nabor mobilnih aplikacij, ki so dostopne uporabniku preko spletne trgovine Android Market.

2.2 Arhitektura

Zgradba operacijskega sistema Android je predstavljena iz štirih plasti, ki jih sestavljajo jedro Linux, knjižnice, izvajalnik kode, aplikativno ogrodje in aplikacije. Slika 2.2 prikazuje kako so sestavljene posamezne plasti.

Linux jedro

Android temelji na jedru Linux 2.6, novejše različice pa na Linux jedru 3.0. Ta plast je namenjena upravljanju s pomnilnikom in drugimi procesi sistema. Prav tako vsebuje gonilnike naprave. Uporabo sistema Linux omogoča dobro prenosljivost, varnost in vsebuje veliko uporabnih funkcij [1].



Slika 2.2: Arhitektura operacijskega sistema Android.

Programske knjižnice

Naslednji nivo sestavljajo programske knjižnice C/C++ in izvajalnik kode (angl. Android runtime). Knjižnice zagotavljajo vse pomembne funkcije operacijskega sistema. Med njimi spadajo OpenGL (podpora 3D grafiki), Webkit, SQLite (podatkovna baza) in druge. Izvajalnik kode pa vsebuje navidezni stroj Dalvik in potrebne knjižnice za izdelavo aplikacij (angl. Android apps). Namen navideznega stroja Dalvik je, da omogoča delovanje aplikacij v vsak svojem procesu operacijskega sistema.

Aplikacijsko ogrodje

Temu sledi aplikacijsko ogrodje, kjer najdemo različne upravljalnike (angl. managers) in prav tako zagotavlja razvijalcem storitve za izdelavo aplikacij

in vmesnik API že vgrajenih aplikacij.

Aplikacijska plast

Na najvišjem nivoju je aplikacijska plast in je uporabniku najpomembnejša, saj vključuje vse mobilne aplikacije, ki so ustvarjene s strani razvijalcev, kot tudi tiste, ki so že vključene v napravi (klici, kontakti, spletni brskalnik...).

2.3 Android SDK

Je paket (angl. Software Development Kit), ki združuje vsa orodja za razvoj mobilnih aplikacij [8]. Vanj so vključena emulator, knjižnice, razhroščevalnik in dokumentacija. Podprt je na operacijskih sistemih Windows, Mac OS X in Linux. Za sam razvoj aplikacij je potrebno še razvojno okolje Eclipse z vtičnikom ADT ali pa se uporablja SDK orodja kar preko ukazne vrstice.

2.3.1 Razhroščevalnik

Del Android SDK je razhroščevalnik, ki se imenuje Dalvik Debug Monitor Server oziroma DDMS. Deluje na emulatorju kakor tudi na računalnik priključeni mobilni napravi. Pri DDMS gre za program, ki komunicira z napravo preko ADB (angl. Android Debug Bridge).

DDMS razvijalcu omogoča

- pregled podatkov o skadu, nitih in procesih naprave,
- zajem slike zaslona naprave,
- logcat (dnevnik sistemskih izhodnih podatkov),
- simuliranje trenutne lokacije,
- opravljanje klicev in pošiljanje SMS sporočil.

2.3.2 Emulator

Emulator je program, ki omogoča delovanje virtualne mobilne naprave kar na lastnem računalniku. Posnema vse funkcionalnosti prave fizične naprave. To daje razvijalcem možnost razvoja in testiranja svojih aplikacij brez uporabe pravih mobilnih naprav. Kot je razvidno iz slike 2.3 zagotavlja prikaz zaslona naprave kot tudi tipkovnico in navigacijske tipke.

Slika 2.3: Android virtualna naprava.

Preden lahko emulator uporabljamo je potrebno nastaviti različne lastnosti AVD (angl. Android Virtual Device). AVD kreiramo v AVD Managerju, kjer določimo katero različico Android platforme naj uporablja, izgled uporabniškega vmesnika in nastavitve strojne opreme naprave (merilec pospeška, GPS, itd.). Nekatere funkcije, ki jih podpira emulator:

- Qwerty tipkovnica,
- GSM modem,
- simulacija SIM kartice,
- podpora SD kartice,

- 16-bitni LCD zaslon
- uporaba kamere,
- podpora različnim senzorjem naprave, itd.

Emulator zaženemo preko razvojnega okolja Eclipse ali preko ukazne vrstice. Vsak AVD deluje kot povsem samostojna naprava. Na emulator nato namestimo razvijajočo aplikacijo, preko katere lahko dostopamo do interneta in drugih senzorjev naprave.

Poglavje 3

Integracija podatkov

Namen razvijajoče aplikacije je razvrstitev javnih parkirišč od najboljše možnosti do najslabše. Na to razvrstitev vplivajo različni dejavniki (oddaljenost, vreme, cena, zasedenost parkirišč). Potrebno je imeti dovolj informacij o teh dejavnikih, zato moramo pridobiti več podatkov iz različnih spletnih virov in jih nato ustrezno uporabiti. V nadaljevanju bo predstavljeno takšno pridobivanje podatkov, ki mu pravimo integracija podatkov.

Integracija podatkov pomeni pridobivanje podatkov iz več virov, njihovo povezovanje in združevanje v neko smiselno celoto. Podatki se lahko nahajajo v različnih oblikah. Uporabniku se tako zagotavlja ustrezen pregled nad veliko količino podatkov. Profesor M. Lenzerini je v svojem delu [4] opisal integracijo podatkov kot problem združevanja podatkov, ki se nahajajo na različnih virih in zagotavljanja uporabniku enotnega pregleda nad temi podatki.

Z razvojem podatkovnih baz se je pojavila potreba po integraciji podatkov. V današnjem času je ta potreba postala vedno večja, saj se skozi podjetja prenašajo vedno večje količine podatkov, ki za odločanje o svojem poslovanju potrebujejo nek urejen sistem. "Kakovostno povezani in učinkovito predstavljeni podatki so temelj za pridobivanje informacij - s tem odpirajo možnost večjega izkoristka poslovnega sistema!" [9]

Do danes se je razvilo več tehnologij za integracijo podatkov, med katerimi

so nspogostejše zajem, transformacija in prenos (angl. Extract, Transform and Load), integracija v realnem času (angl. Enterprise Information Integration) in integracija aplikacij (angl. Enterprise Application Integration).

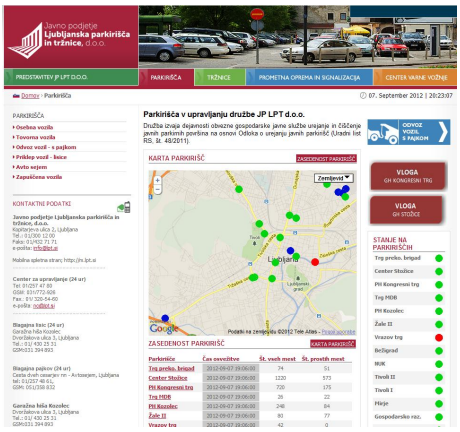
Extract, Transform, Load

ETL je v zadnjem času najpogostejša tehnologija, ki se uporablja za integracijo podatkov [10, 11]. Sestavljen je iz treh procesov:

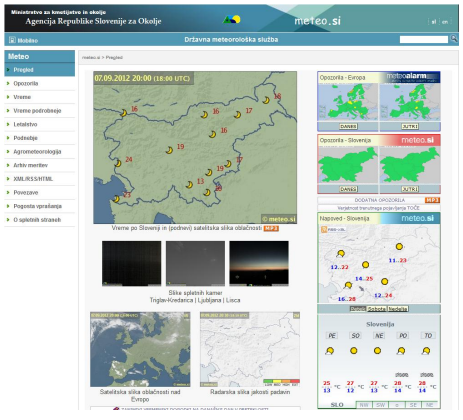
- **zajema podatkov** iz zunanjih virov, arhivskih sistemov,
- **transformacija**, ki preoblikuje podatke v obliko primerno za nadaljno uporabo in jih pri tem filtrira in validira,
- **prenos podatkov** na ustrezno mesto (podatkovne baze, skladišča).

Enterprise Information Integration

EII tehnologija omogoča pridobivanje podatkov, ki se nahajajo na različnih zunanjih virih v realnem času [11]. Poleg podatkov, ki se nahajajo na relacijskih podatkovnih bazah lahko dostopa do XML datotek in drugih nerelacijskih podatkov.



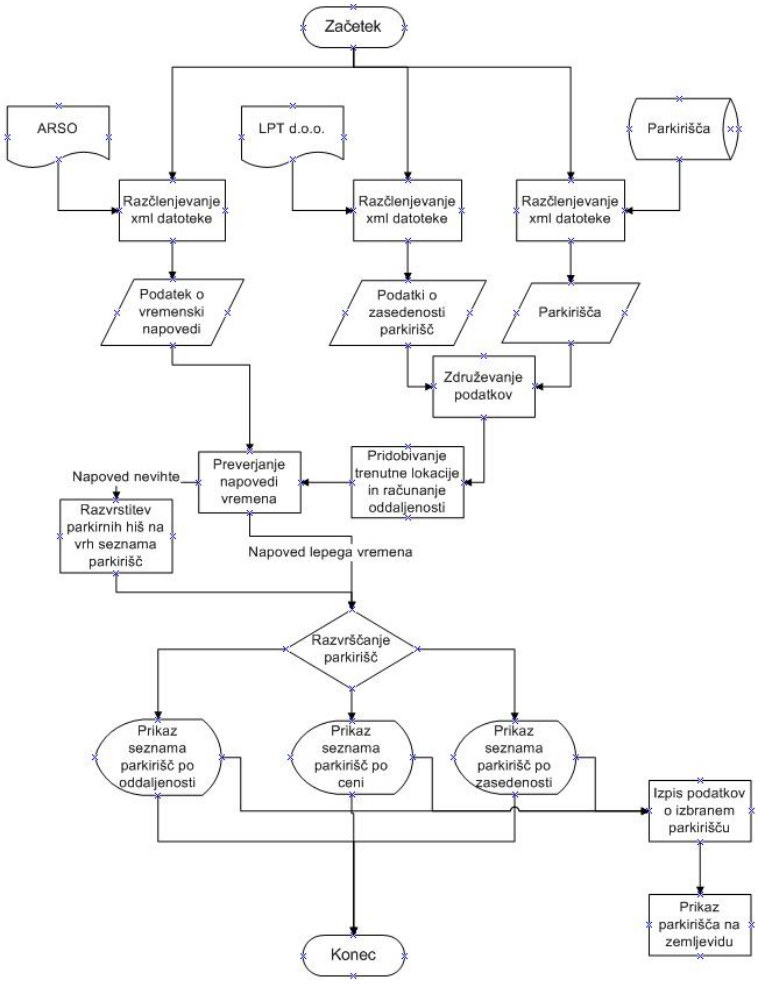
(a) Podjetje LPT d.o.o.



(b) Agencija RS za okolje.

Slika 3.1: Spletni viri od koder se črpajo podatki.

Na principu tehnologije EII deluje tudi aplikacija Parkiraj. Aplikacija iz različnih virov pridobi informacije iz spleta v realnem času, ko uporabnik zažene aplikacijo oziroma ob pritisku na gumb za osvežitev zaslonske maske. Spletni viri od koder se pridobivajo podatki so prikazani na sliki 3.1. Podatke se pridobi z ustreznih XML datotek. S spletne strani podjetja LPT d.o.o. se pridobijo podatki o zasedenosti parkirišč, z Agencije Republike Slovenije za okolje pa napoved vremena za okolico Ljubljane. Ti podatki se nato uporabijo pri izbiri in razvrstitvi parkirišč (slika 3.2).



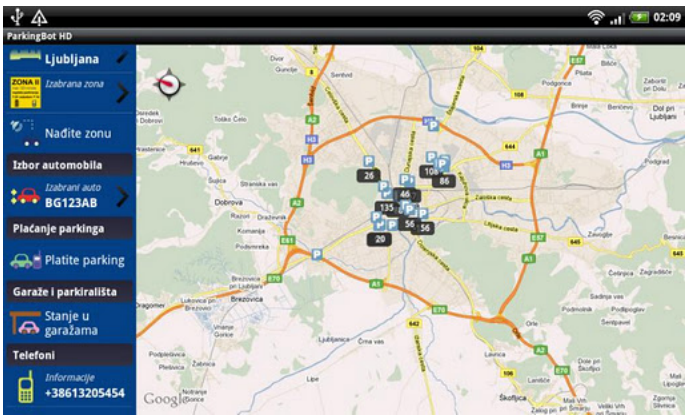
Slika 3.2: Diagram delovanja aplikacije.

Poglavje 4

Obstoječe rešitve

S področja parkirišč in parkiranja obstaja že kar nekaj mobilnih aplikacij, vendar je kljub temu območje Slovenije na tem področju še slabo razvito. Vse te aplikacija so predstavljene na spletni prodajalni podjetja Google, katere naslov je Google Play. Od tu lahko uporabnik po želji kupi aplikacijo in jo namesti na svojo mobilno napravo [12].

ParkingBot HD



Slika 4.1: Aplikacija ParkingBot HD.

Aplikacija je zasnovana za uporabo na tabličnih napravah. Vsebuje informacije o parkiriščih večjih mest v Srbiji, Hrvaški in osnovne informacije

o Sloveniji. Omogoča iskanje najbližjih parkirišč preko GPS, pomnjenja lokacije parkiranega vozila (uporabno v primerih, ko uporabnik pozabi mesto, kjer je parkiral svoje vozilo) in plačevanje parkiranja preko SMS sporočil.

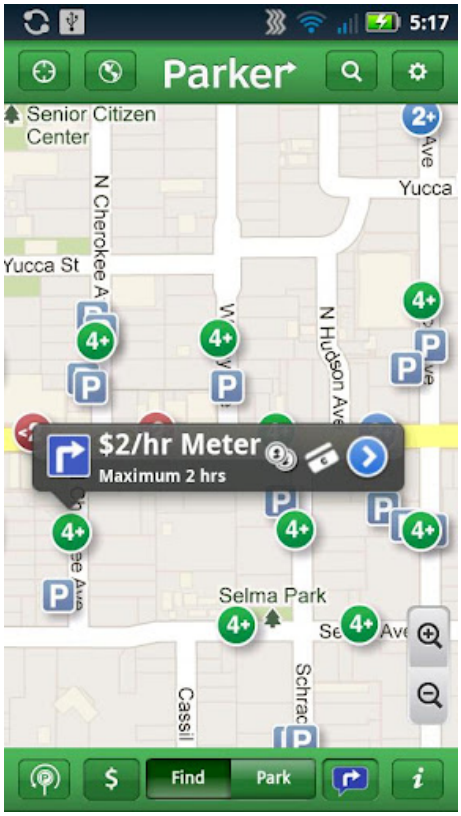
Parkirišča

Aplikacija Parkirišča je ena redkih slovenskih aplikacij v kategoriji parkirišč. Pokriva področje javnih parkirišč v mestu Ljubljana. Omogoča pregled nad zasedenostjo javnih parkirišč, o podatkih zaseženih parkirišč zaradi napačnega parkiranja in ponuja informacije o cenah in kontaktnih podatkov službe, ki opravlja z ljubljanskimi parkirišči.

Slika 4.2: Aplikacija Parkirišča.

Parker

Trenutno pokriva le določena območja Združenih držav Amerike. Omogoča prikaz razpoložljivih parkirnih mest na zemljevidu, rezervacijo mesta in plačilo še preden parkiraš. Uporabnik lahko vključi tudi glasovno vodenje do izbranega parkirnega mesta. Vsako parkirno mesto je označeno s senzorjem, ki zaznava prisotnost vozila.



Slika 4.3: Aplikacija Parker.

4.1 Primerjava aplikacij

V kategoriji iskanja parkirišč obstaja veliko različnih aplikacij, vsaka s svojimi funkcionalnostmi. Skupno vsem je prikaz števila prostih mest na parkiriščih.

Uporabniški vmesnik je ponavadi enostaven, da lahko uporabnik hitro pride do željenih podatkov. Aplikacije omogočajo prikaz najbližjih parkirišč. Večji del aplikacij omogoča prikaz parkirišč na zemljevidu. Kar nekaj aplikacij podpira plačevanje parkirnega mesta preko mobilnega telefona, prav tako pomnjenje lokacije, kjer si parkiral svoje vozilo. Nekatere druge funkcije, ki jih še omogočajo so: navigacija, rezervacija parkirnega mesta, prikaz najpomembnejših lokacij v bližini parkirišča, itd.

Za diplomsko nalogo razvita aplikacija omogoča nekatere zgoraj naštete funkcionalnosti. Omogoča prikaz podatkov o trenutni zasedenosti parkirišč in prikaz parkirišča na zemljevidu. Aplikacija se od drugih razlikuje po samem razvrščanju parkirišč, saj omogoča uporabniku izbiro med več načinov razvrščanja. Omogoča razvrščanje po oddaljenosti, ceni in številu prostih mest. Druge aplikacije ponavadi omogočajo le razvrščanje po oddaljenosti. Najbolj pa se aplikacija od drugih razlikuje pri upoštevanju vremenske napovedi. Aplikacija obvešča uporabnika o stanju vremena in v primeru slabega vremena svetuje uporabo parkirne hiše.

Poglavje 5

Implementacija iskanja parkirišč v Ljubljani

5.1 Ideja

Voznikom je v mestu velikokrat problem najti primerno parkirno mesto, še posebej ob delovnih dneh, ko se v službo vozijo v mesto ljudje iz bližnje in daljne okolice. Dobro je tako imeti nek pregled nad zasedenostjo javnih parkirišč in njihovo oddaljenost od uporbnikove lokacije, kar bi marsikomu prihranilo nepotrebne minute iskanja za prosto parkirno mesto.

Uporabniški vmesnik bi bil enostaven, da se lahko uporabnik hitro pomika skozi zaslonske maske in tako pride kar najhitreje do željenih informacij. Aplikacija bi preverila trenutno pozicijo uporabnika in razvrstila parkirišča po oddaljenosti. Uporabnik bi tako imel informacije o še nezasedenih parkiriščih v bližnji okolici. Seveda bi si lahko uporabnik v nastavitvah nastavljal tudi druge pogoje za razvrstitev parkirišč. S klikom na posamezen element seznama bi dostopali do še bolj podrobnih informacij glede izbranega parkirišča. Parkirišča bi bila za boljšo predstavo uporabnika ustrezno prikazana tudi na zemljevidu.

5.2 Načrtovanje aplikacije

Po opredeljeni ideji je bilo potrebno natančneje določiti način delovanja aplikacije. Namenjena bi bila za uporabnike mobilnih naprav z operacijskim sistemom Android. Aplikacija bi se razvijala za najnovejšo verzijo Androida. Trenutno najbolj razširjena verzija je Android 2.3.3, zato bi se upoštevala pri razvoju podpora starejšim verzijam. Pripraviti je bilo potrebno nek uporabniški vmesnik s katerim bo upravljal uporabnik. Aplikacija mora imeti glavno okno, ki bi predstavljalo seznam parkirišč, preko katerega bi lahko dostopal še do drugih zaslonskih mask. Uporabnik bi imel možnost v nastavitvah nastaviti na kakšen način se naj razvrstijo parkirišča. Delovanje aplikacije se je tekom razvoja spreminjalo in dopolnjevalo z dodatnimi funkcionalnostmi.

Pred samim začetkom razvoja je bilo potrebno pripraviti razvojno okolje. Uporabil se je Eclipse 4.2, kateremu je bilo potrebno dodati vtičnik ADT. Ob kreiranju projekta je bilo potrebno nastaviti za katero verzijo SDK se bo razvijalo in kakšna bo minimalna verzija SDK, ki ga bo aplikacija podpirala. Izbrana je bila najnovejša verzija (Android 4.1, API 16), s podporo do verzije Android 2.2, API 8.

Android projekt je sestavljen iz več direktorijev in avtomatsko generiranih datotek. V mapi src se nahajajo vsi javanski razredi potrebni za delovanje naše aplikacije, v res so vsi viri, ki jih uporablja aplikacija, glede na tip vira (ikone gumbov, xml datoteka s podatkih o parkiriščih, datoteke za oblikovanje, itd.) in druge.

Projekt vsebuje tudi xml datoteko AndroidManifest. Ta datoteka vsebuje vse nujne informacije o komponentah aplikacije, ki jih potrebuje operacijski sistem za zagon programske kode aplikacije [13]. Vanjo vnesemo potrebna dovoljenja in knjižnice, ki jih potrebuje naša aplikacija.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<uses-library android:name="com.google.android.maps" />
```

5.3 Priprava podatkov

Aplikacija hrani osnovne podatke o ljubljanskih javnih parkiriščih. Vsako parkirišče vsebuje podatke o imenu parkirišča, odpiralnem času, ceni, o številu vseh parkirnih mest in številu mest za invalide. Hrani se tudi krajši opis parkirišča in koordinate njegove lokacije.

Potrebe po podatkovni bazi ni bilo, saj podatkov ni veliko. Podatki se nahajajo v xml datoteki, katere zgradba je prikazana v spodnji kodi.

```
<seznamParkirisc>
  <parkirisce>
     <id_parkirisca>2</id_parkirisca>
     <ime>PH Kozolec</ime>
     <delovnik>24 ur (ponedeljek - nedelja)</delovnik>
     <cena></cena>
     <st_mest>248</st_mest>
     <invalidi_st_mest>4</invalidi_st_mest>
     <opis>Parkirna hisa je avtomatizirana. ...</opis>
     <tip>parkirna hisa</tip>
     <longitude>14.505016</longitude>
     <latitude>46.056917</latitude>
  </parkirisce>
  ...
</seznamParkirisc>
```

Ta xml datoteka se nahaja v za to namenjeni mapi xml, ta pa se nahaja znotraj mape res, kjer se hranijo vsi potrebni podatki, glede na njihov tip.

Podatki so bili pridobljeni s spletnega mesta podjetja Ljubljanska parkirišča in tržnice, d.o.o.. Geografska širina in dolžina pa s pomočjo spletne storitve Google Maps.

5.4 Uporabniški vmesnik

Zaslonske maske aplikacije se lahko gradijo programsko, torej v kodi samega programa. Drugi, bolj praktičen način, ki smo ga uporabili tudi sami, pa omogoča izdelavo mask s pomočjo postavitvenih xml datotek. Vsaka aktivnost ima svojo xml datoteko, ki definirajo njihov izgled na zaslonu. Te datoteke se nahajajo v mapi layout direktorija res. Primer teh datotek je

prikazan v spodnji kodi.

```
<?xml version="1.0" encoding="UTF-8"?>
<LinearLayout>
    ...
    <ListView
        android:id="@+id/listView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>
    ...
</LinearLayout>
```

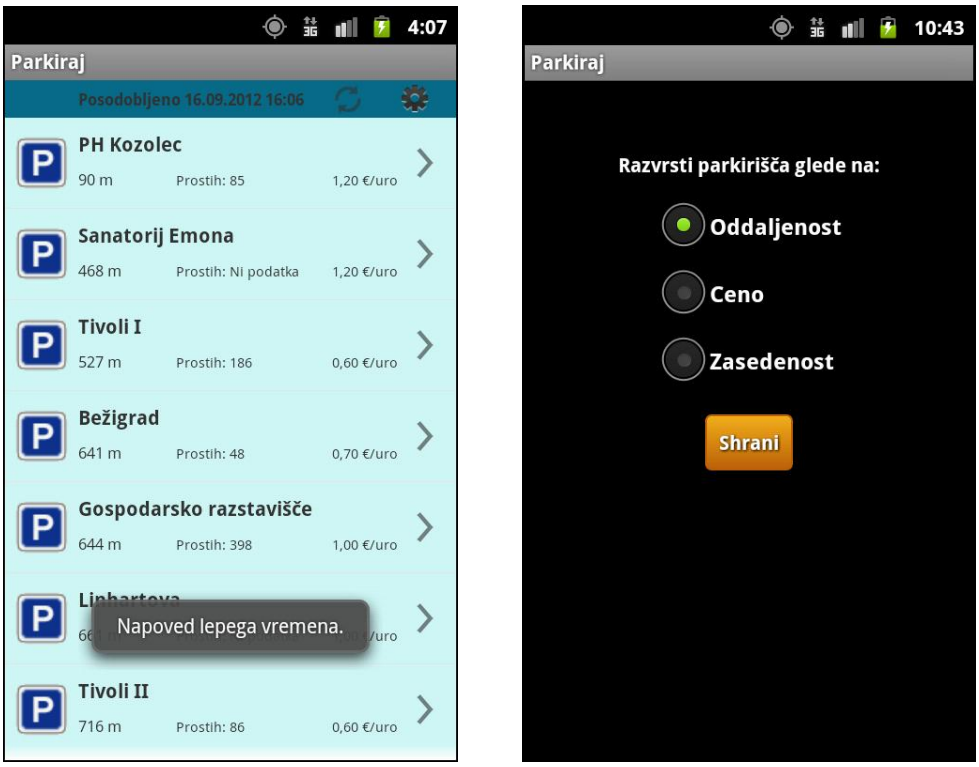
S pisanjem ustreznih xml zaznamkov smo hitro zgradili željen uporabniški vmesnik. Gradnikom tako določamo njihov položaj in lastnosti. Obstaja več glavnih tipov postavitev zaslonske maske od katerih smo uporabili LinearLayout, RelativeLayout in TableLayout. Določajo razvrstitev vseh ostalih gradnikov znotraj njih. Za bolj kompleksnejšo obliko smo uporabili gnezdenje glavnih tipov postavitev.

Aplikacijo Parkiraj sestavljajo štiri aktivnosti, vsaka s svojo zaslonsko masko.

Glavni zaslon in nastavitve

Glavna aktivnost aplikacije je seznam javnih parkirišč, ki se prikaže ob zagonu aplikacije (slika 5.1a). Za vsako parkirišče so podani podatki o oddaljenosti, ceni in zasedenosti parkirišča. Vsa parkirišča so razvrščena od najboljše možnosti na vrhu do najslabše na dnu seznama. Privzet način razvrščanja je po oddaljenosti uporabnika od trenutne lokacije. Uporabnik s pritiskom na zgornji desni gumb (v obliki zobatega kolesa) dostopa do nastavitev (slika 5.1b), kjer lahko izbira med drugimi načini razvrščanja (cena, zasedenost). Prav tako aplikacija obvešča uporabnika o vremenu in na zaslonu izpiše sporočilo o napovedi vremena. Vreme prav tako vpliva na samo razvrstitev parkirišč, saj v primeru slabega vremena aplikacija priporoča uporabo parkirnih hiš. Na vrhu zaslona je obvestilo kdaj je bil zaslon nazadnje osvežen.

V xml datoteki, kjer je določen izgled maske je na najvišjem nivoju element LinearLayout z vertikalno postavitvijo elementov. Znotraj glavnega tipa postavitve so vgnezdeni LinearLayout za prikaz okvirja na vrhu zaslona, kjer se nahajata dva gumba ImageButton za osveževanje in prehod na novo aktivnost, ki je namenjena nastavitvam. Za prikaz seznama parkirišč se uporabi ListView, katerega izgled posamezne vrstice je določen v novi xml datoteki.



(a) Glavni zaslon. (b) Nastavitve.

Slika 5.1: Prikaz seznama parkirišč in nastavitve razvrščanja.

Parkirišče

S klikom na gumb v obliki puščice določenega parkirišča v seznamu odpremo novo aktivnost, kjer so prikazane podrobnejše informacije o izbranem parkirišču (slika 5.2). V spodnjem delu zaslona se nahaja gumb za prehod na zaslonsko masko z zemljevidom.

Poleg že vseh omenjenih grafičnih elementov se je tu uporabil še TableLayout za predstavitev števila parkirnih mest.

Slika 5.2: Podrobnejše informacije izbranega parkirišča.

Prikaz zemljevida

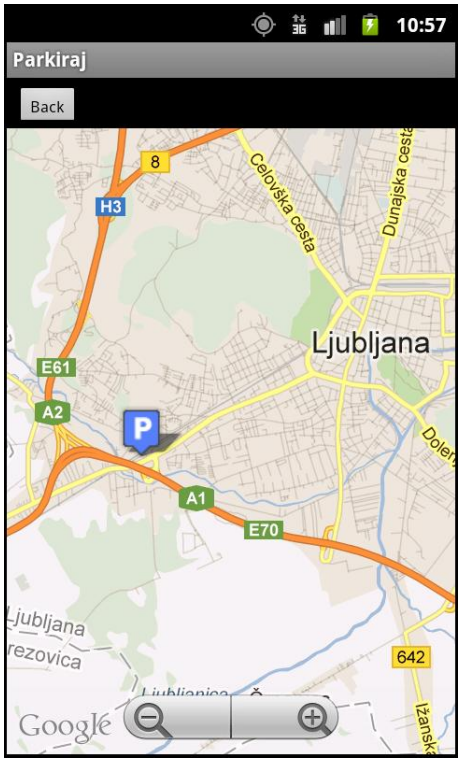
Aktivnost za prikaz lokacije izbranega parkirišča na zemljevidu mesta Ljubljana. Po zemljevidu se lahko premikamo s prstom in po želji približujemo in oddaljujemo pogled (slika 5.3).

V postavitveni xml datoteki se nahaja element MapView. Ta prikaže zemljevid, ki ga pridobi preko storitve Google Maps. Zgradba elementa MapView je prikazana v spodnji kodi.

```
<com.google.android.maps.MapView
        android:id="@+id/mapView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true"
        android:layout_below="@+id/navigate_tab2"
        android:clickable="true"
        android:apiKey="0Vvp-2nxQa70VUD1pu_58-ubPOrtHoaSRm1sJgQ" >
</com.google.android.maps.MapView>
```

Preden lahko uporabljamo zemljevid, moramo registrirati certifikat s storitvijo Maps. Pomemben je element API Key, ki ga pridobimo za vsak registriran certifikat. Gre za enolični identifikator, s katerim nas lahko z vsakim dostopom do storitve Maps, server prepozna in omogoči prikaz zemljevida. V nasprotnem primeru se bo aplikacija še vedno uspešno zagnala vendar se bo izrisal prazen zaslon [14].



Slika 5.3: Lokacija parkirišča na zemljevidu.

5.5 Programska logika

Za delovanje same aplikacije je potrebna v ozadju neka programska logika, ki skrbi da se podatki prikažejo na zaslonu. Javanska koda je tista, ki poganja aplikacijo.

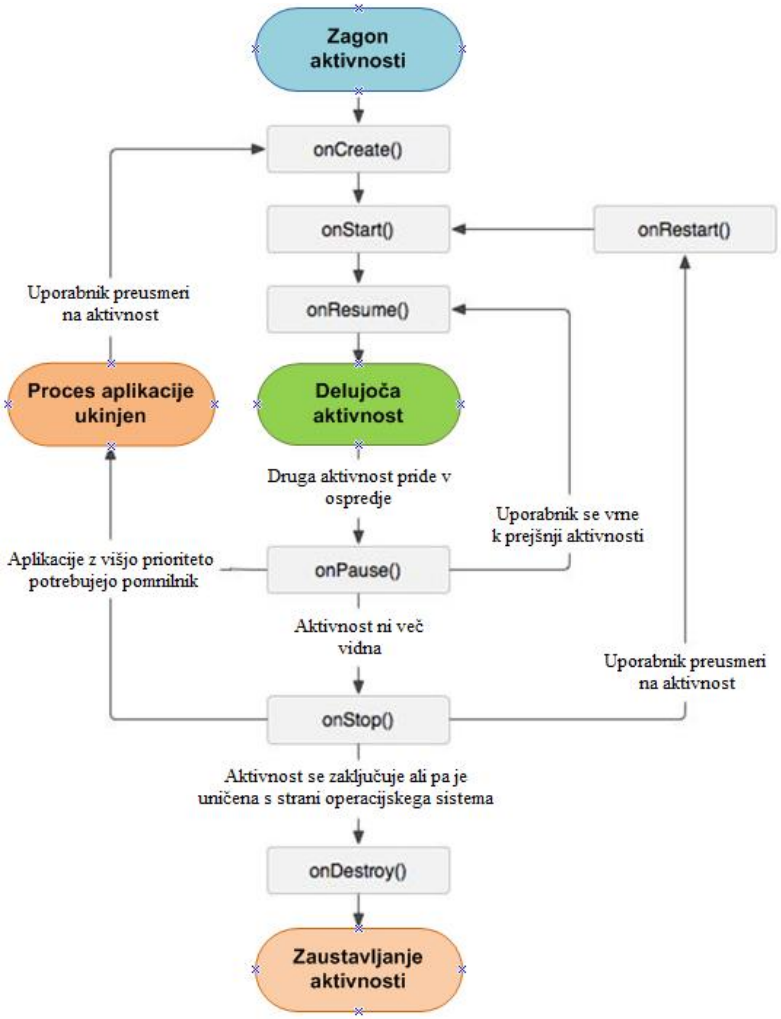
5.5.1 Aktivnost

Aktivnost je komponenta aplikacije, ki uporabniku zagotavlja uporabniški vmesnik s katerim lahko upravlja [1, 15]. Aplikacija je običajno sestavljena iz več aktivnosti med katerimi lahko uporabnik poljubno prehaja. Aplikacijo Parkiraj sestavljajo štiri aktivnosti MainActivity, SettingsActivity, ParkirisceActivity in ZemljevidActivity. Prav tako ima aplikacija glavno aktivnost, ki se prikaže ob samem zagonu aplikacije (to je aktivnost MainActivity). Vsaki aktivnosti pripada okno v katerem izriše vsebino zaslonske maske.

Življenski cikel aktivnosti

Vsaka aktivnost ima svoj življenski cikel, ki je prikazan na sliki 5.4. Cikel se začne z metodo onCreate() za inicializacijo aktivnosti. V glavni aktivnosti je metoda namenjena pripravi vseh potrebnih podatkov. S spleta se preberejo podatki o vremenu in zasedenosti parkirišč. Prav tako se preberejo podatki o parkiriščih iz lokalne datoteke in se ob tem kreirajo objekti tipa parkirisce. Metodo onCreate() je potrebno vedno implementirati. Cikel se konča z metodo onDestroy() s katero se sprostijo vsi uporabljeni viri. Ta metoda v našem primeru ni bila implementirana.

Notranji cikel predstavljajo metode onStart(), onStop in onRestart(), v katerem je aktivnost dejansko vidna za uporabnika na ekranu. To je cikel, ko si uporabnik oglеduje seznam parkirišč ter preklaplja med aktivnostmi za vpogled v nastavitve, podrobnejših informacij o parkirišču ter ogled zemljevida. V glavni aktivnosti se v metodi onStart() pridobi koordinate trenutne lokacije uporabnika, izračuna oddaljenost od parkirišč in prikaže urejen seznam na zaslonu.



Slika 5.4: Življenski cikel aktivnosti.

Prožilec komponent

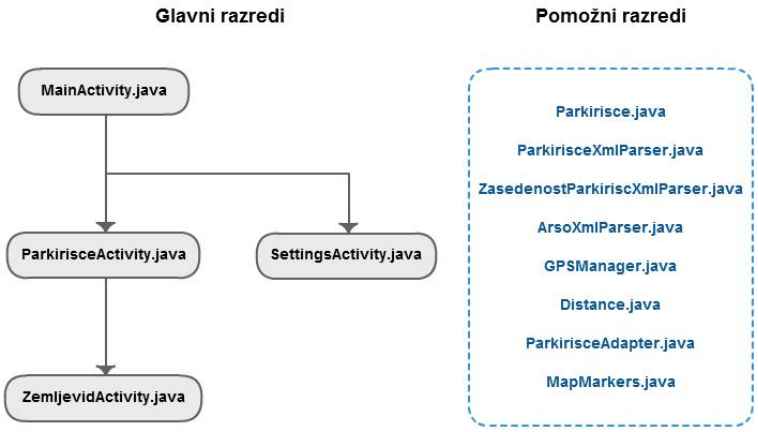
Za zaganjanje drugih aktivnosti uporabimo asinhronski prožilec oziroma namero (angl. Intent). Gre za objekt, ki vsebuje vsebino sporočila in določa akcijo, ki se bo zahtevala [1].

Prožilec podamo metodi startActivity() s katero poženemo novo aktivnost kot so SettingsActivity, ParkirisceActivity in ZemljevidActivity. To metodo kličemo ob pritisku na gumb. Klic metode je prikazan v spodnji kodi za odpiranje nastavitev.

```
public void openSettings(View v) {
  Intent intent = new Intent(MainActivity.this, SettingsActivity.class);
  MainActivity.this.startActivity(intent);
}
```

5.5.2 Razredi aplikacije

Aplikacijo sestavlja več javanskih razredov. Vsi razredi so prikazani na sliki 5.5. Imamo štiri glavne razrede, ki predstavljajo aktivnosti aplikacije. Z usmerjenimi črtami je ponazorjeno kako si sledijo aktivnosti po pomembnosti in kako uporabnik prehaja med aktivnostmi. Iz slike je razvidno, da se ob zagonu aplikacije zažene aktivnost MainActivity. Od tu lahko naprej dostopamo do aktivnosti za nastavitve SettingsActivity ter ParkirisceActivity za podrobnejši opis izbranega parkirišča. Preko slednje aktivnosti dostopamo do ZemljevidActivity za prikaz parkirišča na zemljevidu. Vse te aktivnosti za svoje delovanje potrebujejo dodatne razrede, ki jih najdemo na desni strani slike.



Slika 5.5: Programski razredi aplikacije Parkiraj.

Najpomembnejša aktivnost je torej MainActivity, katere glavna naloga je prikaz urejenega seznama javnih parkirišč. V ta namen je potrebno podatke o zasedenosti parkirišč in napovedi vremena pridobiti iz spletnih virov. Za

dostop do interneta je bilo potrebno v manifest datoteko vnesti spodnja dovoljenja.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Pred dostopom do interneta se je v metodi checkNetworkConection() preverilo, če je internetna povezava mogoča, saj je lahko naprava izven obsega omrežja ali pa ima uporabnik izklopljen prenos podatkov. Za prenos podatkov in izvedbo GET metode se je uporabil HttpURLConnection klient.

Vsi podatki so bili shranjeni v xml formatu. Razreda ZasedenostParkiriscXmlParser in ArsoXmlParser poskrbita, da se podatki o zasedenosti parkirišč in vremenu pridobijo v primerni obliki. Željeni podatki so se iz xml datotek pridobili s pomočjo parserja XmlPullParser. Parser izlušči podatke iz tistih mest, ki smo mu jih določili da jih želimo, ostalo pa pusti pri miru. Spodaj je koda metode s pomočjo katere se pomikamo med xml zaznamki.

```
xmlParser.require(XmlPullParser.START_TAG, namespace, "ROOT");
while(xmlParser.next()!=XmlPullParser.END_TAG) {
        if(xmlParser.getEventType()!=XmlPullParser.START_TAG) {
             continue;
    }
        String tagZasedenost = xmlParser.getName();
        if(tagZasedenost.equals("ZASEDENOST")) {
            prostaMesta(xmlParser, parkirisca);
        } else {
            skipTag(xmlParser);
        }
}
```

Za podatke o parkiriščih se je dostopalo do datoteke parkirisca.xml, ki se nahaja lokalno v mapi res (mapa kjer se nahajajo vsi tipi podatkov). Razred ParkirisceXmlParser poskrbi, da se kreirajo objekti tipa Parkirisce.

Izračun oddaljenosti parkirišč

Ena od možnosti je razvrstitev parkirišč glede na oddaljenost od uporabnika. Aplikacija ob zagonu pridobi trenutno lokacijo uporabnika v razredu GPSManager ob pomoči razreda LocationManager. Dostop do lokacijskih

podatkov je bilo potrebno omogočiti z vpisom ustrezne pravice v manifest datoteki. Potrebno je bilo dodati naslednjo vrstico.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Lokacija se pridobiva preko GPS sprejemnika. Najprej z metodo isProviderEnabled(LocationManager.GPS_PROVIDER) preverimo ali je GPS omogočen in v primeru, da je vključen se s klicem metode requestLocationUpdates() posodobi trenutna lokacija. Zadnja znana lokacija uporabnika pa se pridobi s spodnjo kodo.

```
locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
```

Trenutna lokacija je tipa Location. Geografsko širino in dolžino dobimo s pomočjo metod getLatitude() in getLongitude().

Ko končamo z uporabo lokacijskih podatkov moramo prekiniti posodabljanje lokacije, da zmanjšamo nepotrebno porabo električne energije. To storimo v metodi glavne aktivnosti onStop() s klicem removeUpdates(), ki se nahaja v razredu GPSManager, kot prikazuje spodnja koda.

```
protected void onStop() {
  super.onStop();
  if(gpsManager!=null) gpsManager.disableGPS();
}

public void disableGPS() {
  if(locationManager != null) {
    locationManager.removeUpdates(GPSManager.this);
  }
}
```

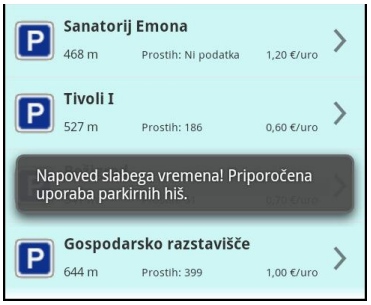
V razredu Distance se izračunajo oddaljenosti vseh parkirišč od prej pridobljene lokacije uporabnika. Razdalja med dvema lokacijama se računa z metodo distanceBetween(), ki vrne razdaljo v metrih.

V primeru, da GPS ni vključen, aplikacija uporabnika o tem obvesti s sporočilom na zaslonu. Za prikaz sporočila je bil uporabljen razred AlertDialog. Uporabniku je ponujena možnost, da odpre nastavitve mobilnega telefona, kjer lahko omogoči GPS. Dostop do nastavitev omogoča spodnja koda.

```
private void enableLocationSettings() {
    Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
    startActivity(intent);
}
```

Seznam parkirišč se ureja s pomočjo metode Collections.sort() in vmesnika Comparator. Comparator se prenese na metodo sort, da se omogoči še natančnejši nadzor nad ureditvijo vrstnega reda. Poleg oddaljenosti se parkirišča razvršča še glede na ceno in število prostih mest (slika 5.1b).

Za prikaz seznama na zaslonu poskrbi razred ParkirisceAdapter. Vsaka vrstica gradnika ListView predstavlja določeno parkirišče (slika 5.6). Na koncu vrstice nas gumb preusmeri na zaslonsko masko z informacijami izbranega parkirišča.



Slika 5.6: Seznam parkirišč in obvestilo o vremenu.

Sporočilo o stanju vremena se kreira s pomočjo razreda Toast. Gre za sporočilo, ki se prikaže na zaslonu brez, da se prekine interakcija s trenutno aktivnostjo (slika 5.6).

Prenos podatkov med aktivnostmi

V novo aktivnost se prenesejo podatki preko prožilca. Podatki se prenašajo v obliki parov ključ in vrednost. Iz glavne aktivnosti se v ParkirisceActivity prenese objekt tipa Parkirisce izbranega parkirišča za katerega se nato izbišejo podrobnejše informacije. V aktivnost ZemljevidActivity je potrebno

prenesti ime parkirišča, ki ga želimo prikazati na zemljevidu ter njegove koordinate kar lahko vidimo v spodnji kodi.

```
public void onClickPrikaziZemljevid(View v) {
  Intent intent = newIntent(ParkirisceActivity.this,ZemljevidActivity.class);
  intent.putExtra("imePark", parkirisce.getIme());
  intent.putExtra("latitude", parkirisce.getLatitude());
  intent.putExtra("longitude", parkirisce.getLongitude());
  ParkirisceActivity.this.startActivity(intent);
}
```

Za shranjevanje nastavitev oziroma načina razvrščanja parkirišč v aktivnosti SettingsActivity se je uporabil razred SharedPreferences, katerega uporaba je prikazana v spodnji kodi. Ti podatki so vedno dosegljivi znotraj aplikacije.

```
public void saveSettings(View v) {
  SharedPreferences sharedPref=getSharedPreferences("myPrefs",MODE_PRIVATE);
  SharedPreferences.Editor editor = sharedPref.edit();
  editor.putString("nastavi_razvrstitev", buttonText);
  editor.commit();
  finish();
}
```

Zemljevid

Ob zagonu aktivnosti ZemljevidActivity se na zaslonu izriše zemljevid. Za prikaz zemljevida je potrebna zunanja knjižnica za Google Maps storitev. V manifest datoteko je potrebno vnesti naslednjo vrstico:

```
<uses-library android:name="com.google.android.maps" />
```

Z metodo setBuiltInZoomControls() se omogoči že vgrajene kontrole za približevanje in oddaljevanje pogleda. Zemljevid je bilo potrebno nastaviti, da se vedno izriše območje Ljubljane.

```
List<Overlay> mapOverlays = mapView.getOverlays();
Drawable drawable=this.getResources().getDrawable(R.drawable.parkirisce_icon)
MapMarkers markers = new MapMarkers(drawable, this);
point = new GeoPoint((int) (latitude * 1E6),(int) (longitude * 1E6));
OverlayItem overlayItem=new OverlayItem(point, imeParkirisca, "Lokacija");
markers.addOverlay(overlayItem);
mapOverlays.add(markers);
```

Kako se prikaže ikono za parkirišče na zemljevidu ponazarja zgornja koda.

5.6 Testiranje

Testiranje mobilne aplikacije je potekalo na emulatorju. Testi so se izvajali že tekom razvoja aplikacije. AVD kreiramo in ga poženemo v AVD Managerju, do katerega lahko dostopamo preko razvojnega orodja Eclipse. Potrebno je bilo kreirati več navideznih naprav AVD z različnimi nastavitvami. Pri vsaki napravi je bilo potrebno uporabiti API z knjižnico Google Maps, ter omogočiti podporo za GPS.

Testiranje na emulatorju je bilo uspešno, vendar so se pri tem pokazale nekatere slabosti. Zaganjanje virtualne naprave je počasno, prav tako je na trenutke počasno delovanje emulatorja. Ker gre za virtualno napravo ima tudi nekaj omejitev. Težje je testirati spreminjanje trenutne lokacije. To se je testiralo s pomočjo telnet klienta preko katerega smo dostopali do emulatorja in s pomočjo ukaza geo fix spreminjali trenutni položaj uporabnika.

```
telnet localhost 5554
geo fix 14.50588 46.057464
```

Poglavje 6

Sklepne ugotovitve

Mobilna tehnologija je v današnjem času v vedno hitrejšem razvoju. Pametni telefoni so tako postali med najbolj priljubljenimi elektronskimi napravami. Prav to je bil eden glavnih razlogov za izoblikovanje ideje o razvoju mobilne aplikacije. Cilj diplomske naloge je bil izdelati mobilno aplikacijo za pomoč pri iskanju javnih parkirišč v mestu Ljubljana. Aplikacija je bila izdelana za operacijski sistem Android.

Končna različica aplikacije vključuje vse funkcionalnosti, ki so bile predvidene v ideji. Aplikacija ima uporabno vrednost in služi kot dober pripomoček za iskanje primernega parkirišča v bližini uporabnika. Aplikacija ne poišče samo najbljižja parkirišča, ampak ga tudi opozarja na slabo vreme in svetuje, če je bolj primerna uporaba parkirne hiše. Nihče si ne želi imeti vozila zunaj na prostem v času toče.

Na slovenskem trgu je malo mobilnih aplikacij na tem področju, kar daje večjo možnost, da bi se lahko aplikacija dobro razširila med uporabniki. Omejitev pa je, da je trenutno namenjena le uporabi v Ljubljani.

Seveda obstaja še kar nekaj izboljšav s katerimi bi še dodatno nadgradili funkcionalnost obstoječe aplikacije. Vsebina aplikacije bi se lahko razširila še na parkirišča drugih večjih mest v Sloveniji. V aplikacijo bi bilo lahko vključeno glasovno vodenje do same lokacije parkirišča. Dobra bi bila možnost rezervacije mesta in plačilo parkirnine preko telefona. Zanimiva bi bila tudi

možnost spremljanja parkirnih mest za kratkoročno parkiranje, vendar je to že bolj zapleteno, saj bi moralo biti vsako parkirno mesto opremljeno s senzorjem.

Glede na to, da se trend uporabe mobilnih aplikacij vedno bolj povečuje, lahko pričakujemo, da bo v prihodnosti tudi na področju parkirišč velik nabor različnih aplikacij.

Slike

Literatura

[1] M. Gargenta, *Learning Android*, Sebastopol: O'Reilly Media, Inc., 2011.

[2] R. Kalakota, M. Robinson *M-business : the race to mobility*. New York: McGraw-Hill, cop., 2002.

[3] W. Lee, *Beginning Android^{TM} Application Development*, Indianapolis: Wiley Publishing, Inc., 2011.

[4] M. Lenzerini, *Data Integration: A Theoretical Perspective*, Rim: Universita di Roma "La Sapienza", 2002.

[5] (2012) Mobilna telefonija. Dostopno na: http://www.ris.org/index.php

[6] (2012) Samsung May Have Just Become The King Of Mobile Handsets, While S&P Downgrades Nokia To Junk. Dostopno na: http://techcrunch.com/2012/04/27/samsung-may-have-just-become-the-king-of-mobile-handsets-while-sp-downgrades-nokia-to-junk/

[7] (2012) comScore: In U.S. Mobile Market, Samsung, Android Top The Charts; Apps Overtake Web Browsing. Dostopno na: http://techcrunch.com/2012/07/02/comscore-in-u-s-mobile-market-samsung-android-top-the-charts-apps-overtake-web-browsing/

[8] (2012) Android SDK. Dostopno na: http://developer.android.com/sdk/index.html

[9] (2012) Integracija podatkov. Dostopno na: http://integracija-podatkov.com/opis-tematike-integracije-podatkov/predstavitev-integracije-podatkov

[10] (2012) Integracija podatkov. Dostopno na: http://www.crmt.com/si/1/258/Integracija-podatkov

[11] (2012) Understanding the Three E's of Integration EAI, EII and ETL. Dostopno na: http://www.information-management.com/issues/20050401/1023893-1.html

[12] (2012) Android Market. Dostopno na: https://play.google.com/store

[13] (2012) The AndroidManifest.xml File. Dostopno na: http://developer.android.com/guide/topics/manifest/manifest-intro.html

[14] (2012) Google Maps Android API. Dostopno na: https://developers.google.com/maps/documentation/android/

[15] (2012) Activities. Dostopno na: http://developer.android.com/guide/components/activities.html

[16] (2012) Mobile operating system. Dostopno na: http://en.wikipedia.org/wiki/Mobile_operating_system