Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Domen Selinec

Razvoj aplikacije RouteTracker za mobilni operacijski sistem Android

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Ljubljana 2013

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Domen Selinec

Razvoj aplikacije RouteTracker za mobilni operacijski sistem Android

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: izr. prof. dr. Branko Šter

Ljubljana 2013

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: 00561 / 2013
Datum: 15.9.2013

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

Kandidat: **DOMEN SELINEC**

Naslov: **RAZVOJ APLIKACIJE ROUTETRACKER ZA MOBILNI OPERACIJSKI SISTEM ANDROID**
DEVELOPMENT OF APPLICATION ROUTETRACKER FOR MOBILE OPERATING SYSTEM ANDROID

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Raziščite in predstavite arhitekturo in način delovanja mobilnega operacijskega sistema Android. Implementirajte mobilno aplikacijo za sistem Android, ki temelji na razpoznavanju lokacije in risanju sledi gibanja mobilne naprave. Opišite uporabniški vmesnik in preizkusite delovanje implementirane aplikacije.

Mentor:

izr. prof. dr. Branko Šter

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Domen Selinec, z vpisno številko **63010129**, sem avtor diplomskega dela z naslovom:

Razvoj aplikacije RouteTracker za mobilni operacijski sistem Android

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom izr. prof. dr. Branka Štera,
- 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 13. novembra 2013 Podpis avtorja:

Rad bi se zahvalil vsem, ki so mi skozi vsa leta študija stali ob strani in me spodbujali. Še posebej bi se rad zahvalil mami Veri in očetu Marjanu, ki so me kljub neizpolnjevanju pričakovanj, tako moralno kot materialno podpirali. Za spodbudo bi se rad zahvalil tudi sestrama Katji ni Ani. Mentorju dr. Branku Šteru bi se rad zahvalil za vso strokovno podporo in pomoč pri izdelavi diplomskega dela.

Kazalo

Slike

Seznam uporabljenih kratic in simbolov

1080p	angl. *High definition video* video visoke ločljivosti
1G	angl. *First Generation* prva generacija mobilnih omrežij
2G	angl. *Second Generation* druga generacija mobilnih omrežij
3G	angl. *Third Generation* tretja generacija mobilnih omrežij
3D	angl. *Three Dimensional* geometrični model s tremi parametri
4G	angl. *Fourth Generation* četrta generacija mobilnih omrežij
AMOLED	angl. *Active Matrix Organic Light Emitting Diode* tehnologija LED zaslonov
API	angl. *Application Programming Interface* aplikacijski programski vmesnik
ADT	angl. *Android Development Tools* Eclipse vtičnik, ki vsebuje razvojna orodja za platformo Android
DVM	angl. *Dalvik Virtual Machine* navidezni stroj Dalvik
ENIAC	angl. *Electronic Numerical Integrator And Computer*

	prvi elektronski računalnik
GPU	angl. *Graphics Processing Unit*
	grafična procesna enota
GPS	angl. *Global Positioning System*
	sistem globalnega pozicioniranja
GUI	angl. *Graphical User Interface*
	uporabniški vmesnik
IDE	angl. *Integrated Development Enviroment*
	integrirano razvojno okolje
JIT	angl. *Just In Time compiler*
	prevajalnik, ki sproti prevaja bitno kodo
JDK	angl. *Java Development Kit*
	paket razvojnih orodij za programski jezik Java
JRE	angl. *Java Runtime Enviroment*
	javansko izvajalno okolje
JVM	angl. *Java Virtual Machine*
	javanski navidezni stroj
iOS	angl. *iPhone Operating System*
	Applov prenosni operacijski sistem
IPS	angl. *In Plane Switching*
	tehnologija LCD zaslonov
LED	angl. *Light Emitting diode*
	svetleče diode
LCD	angl. *Liquid Crystal Displays*
	zaslon s tekočimi kristali
LTE	angl. *Long Term Evolution*
	standard četrte generacije mobilnih omrežij
MAC	angl. *Media access control*
	MAC address - strojni naslov
mAh	angl. *Milli Ampere Hour*
	miliamperska ura

PDA	angl. *Personal Digital Assistant* osebni organizator
QWERTY	standardna postavitev tipk, ki je prisotna pri osebnih računalnikih
SDK	angl. *Software Development Kit* programski razvojni paket
SSL	angl. *Secure Socket Layer* kriptografski protokol, ki omogoča varno komunikacijo na medmrežju
SQL	angl. *Structured Query Language* strukturirani povpraševalni jezik za delo s podatkovnimi bazami
TFT	angl. *Thin Film Transistor* tehnologija LCD zaslonov
USB	angl. *Universal Seiral Bus* univerzalno serijsko vodilo
WiFi	angl. *Wireless Network* brezžično omrežje
WiMAX	angl. *World Wide Interoperability for Microwave Access* mobilno omrežje četrte generacije

Povzetek

Cilj diplomske naloge je izdelava mobilne aplikacije RouteTracker, ki temelji na zaznavanju lokacije in risanju sledi gibanja mobilne naprave. Hkrati je predstavljen tudi mobilni operacijski sistem Android ter razvoj aplikacij zanj. V prvem delu naloge so opisane mobilne tehnologije in mobilne naprave ter orodja, uporabljena pri razvoju. Sledi opis arhitekture in način delovanja mobilnega operacijskega sistema Android. Na koncu uvodnega dela je podan opis sistema za globalno pozicioniranje (GPS) in Googlovih mobilnih naprav. V drugem delu so opisane zahteve in arhitektura aplikacije, ki je nastala na podlagi podanih zahtev. Sledi podroben opis uporabniškega vmesnika s priloženimi slikami in način uporabe aplikacije. Na koncu so podane sklepne ugotovitve ter predlagane možne izboljšave in nadgradnje aplikacije.

Ključne besede:
Android, mobilne naprave, operacijski sistem, GPS, RouteTracker, mobilne aplikacije.

Abstract

The aim of the thesis is to create a RouteTracker mobile application which is based on device location detection and visual representation of recorded route. At the same time, the Android mobile operating system and its application development are presented. The first part of the thesis discusses mobile technologies, mobile devices and tools used for development. Furthermore, the operation of Android mobile operating system and its architecture are described. Following the introduction, a global positioning system (GPS) and Google mobile devices are presented. The second part describes the application's requirements and architecture which was created in compliance with these requirements. A detailed description of user interface including images and instructions on how to use the application is provided. In conclusion, the application's suggested improvements and upgrading are presented.

Keywords:
Android, mobile devices, operating system, GPS, RouteTracker, mobile applications.

Poglavje 1

Uvod

Razvoj na področju računalništva je od svojega začetka v strmem porastu. Posledično so računalniki in njim podobne naprave kljub vedno manjšim dimenzijam in porabi zmogljivejše. Leta 1946 je ameriško obrambno ministrstvo predstavilo prvi elektronski računalnik ENIAC, ki je bil tisočkrat hitrejši od do tedaj znanih mehanskih računalnikov. Računalnik je v površino meril okoli 60 kvadratnih metrov [6]. V osemdesetih letih so razcvet doživeli osebni računalniki, ki jih v podobni obliki poznamo še danes. V devetdesetih letih prejšnjega stoletja so postali popularni prenosni računalniki, ki so v prvem desetletju 21. stoletja pri osebni uporabi izpodrinili osebne računalnike. V zadnjih petih letih pa so izjemno popularni postali pametni mobilni telefoni in tablični računalniki. Nekatera podjetja so šla korak naprej in predstavila pametne ure ter pametna očala. Cilj je pametne naprave in mobilne tehnologije človeku čimbolj približati in jih integrirati v njegovo življenje. Z razvojem mobilnih naprav se razvijajo tudi mobilne tehnologije. Kombinacija uporabe mobilnih tehnologij in mobilnih naprav uporabniku nudi izjemno uporabniško izkušnjo, ki mu je dostopna kadarkoli in kjerkoli. Ker so mobilne naprave vedno zmogljivejše, pri nekaterih opravilih nadomeščajo tako osebne kot prenosne računalnike.

Zaradi vedno večjih zmogljivosti je možnosti za uporabo mobilne naprave veliko. S pojavom mobilnih operacijskih sistemov in razvojnih okolij za-

nje, število mobilnih aplikacij strmo narašča. Sposobnost zaznavanja lokacije mobilne naprave je še posebej zanimiva, ker odpira povsem nove dimenzije v svetu mobilnih aplikacij. Najbolj popularni so navigacijski sistemi in sledilci. Na trgu je lepo število sledilcev, namenjenih predvsem športu in rekreaciji. Idejo za aplikacijo RouteTracker sem dobil, ker nisem zasledil nobene enostavne mobilne aplikacije, ki bi sledila napravi, hkrati pa omogočala tudi fotografiranje in bi fotografijo umestila na zajeto sled. Nekatere enostavne aplikacije, ki že obstajajo in omogočajo sledenje so: My Tracks, RunKeeper in SolidSync [10]. Nobena od naštetih aplikacij pa ne ponuja fotografiranja in prikazovanja teh fotografij na sledi naprave. V nadaljnjih poglavjih so opisane mobilne tehnologije, mobilne naprave, operacijski sistem Android ter rezultat diplomskega dela, mobilna aplikacija RouteTracker.

Poglavje 2

Uporabljene tehnologije in orodja

2.1 Mobilne tehnologije

Človek skozi svojo celotno zgodovina obstoja stremi k udobju in napredku. Posledica tega je vedno hitrejši razvoj novih naprav, storitev in tehnologij. Ta je še posebej očiten pri razvoju računalnikov in njim sorodnih tehnologij in naprav. Med te sodijo tudi mobilne tehnologije.

Mobilne tehnologije so tehnologije, ki uporabniku med uporabo omogoča mobilnost. Razvoj mobilnih tehnologij je iz leta v leto hitrejši, saj človek stremi k čim večji mobilnosti, hkrati pa tudi neoviranemu dostopu do mobilnih storitev, kot so pogovori, dostop do podatkov ter storitev na svetovnem spletu, sporočila in nadzor oddaljenih naprav. Mobilne tehnologije se v grobem delijo na:

- mobilna oziroma brezžična omrežja,
- mobilne naprave,
- mobilne storitve.

Slika 2.1: Mobilne naprave.

Mobilna oziroma brezžična omrežja so omrežja, ki za komunikacijo namesto žičnih povezav uporabljajo radijske valove. Predstavljajo osnovno infrastrukturo, ki uporabnikom mobilnih naprav omogoča uporabo mobilnih storitev. Mobilno omrežje najnovejše generacije je omrežje LTE, ki velja za omrežje 4G, čeprav ne izpolnjuje vseh zahtev, ki so določene v standardu za to omrežje [12].

Mobilne naprave so mobilni telefoni, tablični računalniki in prenosni računalniki. Na sliki 2.1 so predstavljeni vsi trije tipi mobilnih naprav. Med uporabo mobilnega telefona ali tabličnega računalnika se uporabnik lahko prosto giba, med uporabo prenosnega računalnika pa je z gibanjem omejen. Naprava, ki je največ v uporabi, je mobilni telefon.

Med mobilnimi telefoni imajo pametni telefoni vse večji delež, ki je v nekaterih državah že več kot 50% [4].

Mobilne storitve so storitve, ki jih uporabnik koristi z uporabo mobilne naprave v mobilnem omrežju. Najbolj pogosto uporabljena mobilna storitev je prenos govora, s prihodom pametnih telefonov pa je vedno bolj uveljavljena storitev prenosa podatkov. Slednja nudi uporabniku nešteto možnosti uporabe, še posebej v kombinaciji s senzorji pametnega telefona.

Slika 2.2: Motorola DynaTAC, prvi prototip mobilnega telefona.

2.1.1 Zgodovina mobilnih telefonov

Prvi prototip ročnega mobilnega telefona je leta 1973 predstavilo podjetje Motorola. Pred tem so bili mobilni telefoni vgrajeni samo v avtomobile, vlake in letala ali pa so imeli ogromne akumulatorje, ki so bili vgrajeni v posebnih kovčkih. Prvi klic z ročnim mobilnim telefonom je opravil Motorolin inženir Martin Cooper. Klical je svojega kolega iz Bell Labs, dr. Joela S. Engela. Prototip, iz katerega je klical Cooper, je tehtal 1,1 kilograma, v višino je meril 23 centimetrov, v širino 4,45 centimetrov ter v globino 13 centimetrov. Baterija je ob 10-urnem polnjenju zagotavljala za 30 minut pogovora [11]. Kakor je razvidno iz slike 2.2, je bil prvi mobilni telefon brez zaslona.

Prvi mobilni telefon, ki je bil namenjen komercialni uporabi, je bil Motorola DynaTAC 8000x. Predstavljen je bil leta 1983 [11].

Prvo komercialno mobilno omrežje je bilo predstavljeno na Japonskem leta 1979 s strani podjetja Nippon Telegraph and Telephone. Leta 1981 mu

Slika 2.3: Simon Personal Communicator, prvi prototip pametnega telefona.

je sledilo podjetje Nordic Mobile Telephone, ki je postavilo mobilno omrežje v skandinavskih državah. Do sredine osemdesetih let je mobilna omrežja uvedlo še nekaj drugih držav.

Leta 1991 je podjetje Nippon Telegraph and Telephone predstavilo mobilno omrežje druge generacije 2G, deset let pozneje pa omrežje tretje generacije 3G. Okoli leta 2009 je postalo jasno, da je zaradi povečane uporabe pametnih telefonov mobilno omrežje tretje generacije postalo prepočasno. Zaradi tega se je začelo razvijati mobilno omrežje četrte generacije 4G. Prvi dve komercialno uporabni omrežji sta WiMAX in LTE. WiMAX je v uporabi v Ameriki, LTE pa v Evropi.

2.1.2 Pametni telefon

Pametni telefoni (angl. *smartphone*) so telefoni, ki združujejo funkcije telefona in funkcije ročnih računalnikov, kot so dlančniki in osebni organizatorji (*PDA*).

Pojem pametni telefon se je pojavil leta 1997, ko je švedsko podjetje Ericsson predstavilo svoj koncept pametnega telefona GS 88. Prvi mobilni telefon, ki je združeval funkcije telefona in osebnega organizatorja, je bil predstavljen s strani podjetja BellSouth leta 1994. Simon Personal Commu-

nicator (slika 2.3) je poleg klicanja omogočal pošiljanje elektronske pošte in faksov. Imel je imenik, koledar, opomnik, računalo, elektronsko beležnico ter navidezno tipkovnico na zaslonu občutljivem na dotik. Čeprav pojem pametnega telefona takrat še ni bil poznan, je ta naprava sodila v to kategorijo telefonov [16].

Leta 1996 je finsko podjetje Nokia predstavilo model Nokia 9000, ki je bil del njihove linije Nokia Communicator. Telefon je združil dve napravi, telefon in osebni organizator podjetja HP. Telefon je imel zaslon ločljivosti 640x200 točk ter fizično QWERTY tipkovnico. Na telefonu je tekel operacijski sistem GEOS V3.0. Nokii so sledila še podjetja HP, Palm in HTC [16]. Revolucijo na trgu pametnih telefonov je leta 2007 naredilo podjetje Apple s svojim modelom iPhone prve generacije. Od takrat dalje je prodaja pametnih telefonov začela strmo naraščati.

Pametni telefoni poleg klicanja nudijo še veliko ostalih storitev, kot so dostop do spleta, fotografiranje, pregledovanje in kreiranje dokumentov, navigacija in še veliko drugih. Dobro uporabniško izkušnjo pri uporabi teh storitev omogočajo zaradi svojih tipičnih lastnosti [18].

Lastnosti pametnih telefonov:

- Velik zaslon, občutljiv na dotik.
  Pametni telefoni imajo vgrajene velike zaslone, občutljive na dotik. Diagonala zaslona pri nekaterih modelih dosega tudi 5,5 palcev ter ločljivost 1080p, kar je 1920x1080 točk. Tehnologije matrik zaslonov so različne, TFT v telefonih nižjega cenovnega razreda ter IPS in AMOLED v telefonih srednjega in višjega cenovnega razreda. Zaradi velikih in visokoločljivostnih zaslonov je s pametnim telefonom mogoče brskati po spletu, urejati dokumente in slike ter gledati filme.

- Operacijski sistem.
  Ena izmed glavnih značilnosti pametnega telefona je mobilni operacijski sistem (angl. *mobile operating system*), ki med drugim uporabniku

omogoča tudi nalaganje poljubnih mobilnih aplikacij. Mobilne operacijske sisteme najdemo tudi na tabličnih računalnikih, osebnih organizatorjih in ostalih ročnih napravah. V zadnjem času lahko mobilni operacijski sistem najdemo tudi v televizijskih sprejemnikih. Tukaj še posebej izstopata Googlov Android in Applov iOS.

- Aplikacije.
  Običajni telefoni imajo naložene le nekatere osnovne aplikacije, kot so imenik kontaktov, koledar, računalo, igrice, budilka, itd. Pametni telefoni zaradi mobilnega operacijskega sistema poleg privzetih aplikacij omogočajo nalaganje naprednejših aplikacij, kot so urejevalniki besedil in slik, navigacija, 3D igrice, itd. Izbira aplikacij za pametne mobilne telefone je zelo velika in pestra. Na telefon se nalagajo iz spletne trgovine dotičnega operacijskega sistema. Aplikacije so lahko plačljive in zelo zmogljive ali zastonjske in bolj enostavne. Prve so največkrat razvite s strani podjetij, v nasprotju z drugimi, ki jih navadno razvijajo posamezni zanesenjaki. V zadnjem času se v mobilnih telefonih pojavljajo tudi napredne storitve, kot so pretvarjanje govora v besedilo in obratno (angl. *speech to text, text to speech*) in prepoznava slik. Nekatere najbolj priljubljene aplikacije za operacijski sistem Android so Facebook, Viber, Skype, Gmail in Google Maps [14].

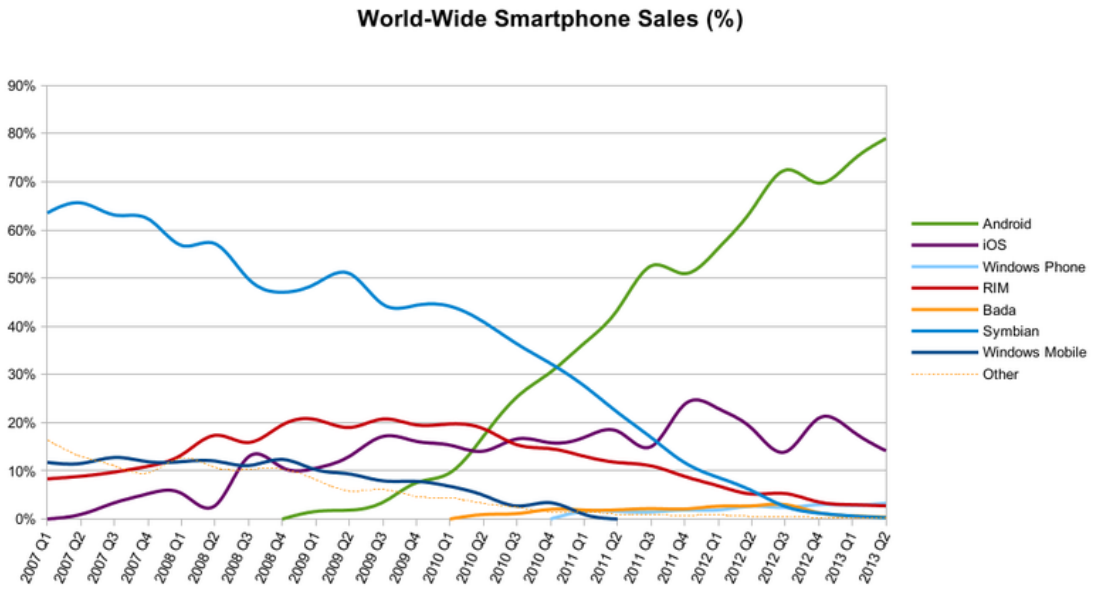
- Dostop do spleta.
  Pametni telefoni omogočajo dostop do spleta bodisi preko mobilnih omrežij bodisi preko WiFi dostopnih točk. Za dostop preko mobilnega omrežja se uporablja tehnologija 3G, medtem ko nekateri novejši modeli že podpirajo tehnologijo 4G. Zaradi hitrega prenosa podatkov je mogoče tekoče brskanje po spletu, poslušanje glasbe, gledanje videoposnetkov ter vse ostalo, kar svetovni splet nudi.

- QWERTY tipkovnica.
  V nasprotju z običajnimi mobilnimi telefoni imajo pametni mobilni telefoni QWERTY tipkovnico. V večini primerov tipkovnica ni fizična,

ampak navidezna, upodobljena na delu zaslona, občutljivega na dotik. Po fizičnih QWERTY tipkovnicah so znani predvsem telefoni linije BlackBerry proizvajalca BlackBerry Limited, vendar se tudi ta proizvajalec v zadnjem času poslužuje navideznih tipkovnic. Prednost navidezne tipkovnice je predvsem v tem, da je prikazana samo takrat, ko je potrebno. V primeru, da tipkovnica ni prikazana, je vidno polje zaslona neprimerno večje, kot pri telefonih s fizično tipkovnico.

- Zmogljiva strojna oprema [3].
  Poleg kvalitetnih zaslonov, ki so občutljivi na dotik, in tehnologij za dostop do spleta imajo pametni mobilni telefoni vgrajeno tudi ostalo zelo zmogljivo strojno opremo. Najzmogljivejši modeli imajo štirijedrne procesorje s taktom do 2,26 GHz, do 64 GB zunanjega pomnilnika, ki je pri nekaterih modelih razširljiv z zunanjo pomnilniško kartico, in 2 GB delovnega pomnilnika. Imajo po dve kameri, katerih ločljivost pri nekaterih modelih presega 13 milijonov točk, obenem pa zmorejo snemati posnetke v ločljivosti 1080p. Pametni mobilni telefoni imajo vgrajeno posebno grafično enoto, GPU, ki omogoča tekoče igranje 3D iger in gledanje videoposnetkov. Za povezljivost skrbijo standardi WiFi, 3G, 4G, Bluetooth in USB.
  Poleg naštete strojne opreme so v pametnih mobilnih telefonih prisotni tudi naslednji senzorji: GPS sprejemnik, pospeškomer (angl. *accelerometer*), kompas, senzor svetlobe (angl. *light sensor*), senzor bližine (angl. *proximity sensor*) in žiroskop (angl. *gyroscope*).
  Da vsa ta zahtevna strojna oprema pravilno deluje in nudi dobro uporabniško izkušnjo, je potrebna dovoljšnja zaloga energije, ki je shranjena v baterijah, katerih kapaciteta pri nekaterih modelih presega 3000 mAh.

Zaradi vseh naštetih lastnosti pametni mobilni telefoni na veliko področjih izpodrivajo osebne in prenosne računalnike. Zaradi vgrajenih senzorjev nudijo odlično uporabniško izkušnjo. Tukaj še posebej izstopajo zelo uporabne



Slika 2.4: Prodajni delež mobilnih operacijskih sistemov.

aplikacije, ki delujejo na podlagi lokacije telefona (angl. *location aware applications*).

2.1.3 Mobilni operacijski sistemi

Operacijski sistem je najpomembnejša programska oprema pametnega mobilnega telefona, ki je zadolžena za upravljanje z strojnimi in programskimi viri naprave. Operacijski sistem je platforma za razvoj in poganjanje mobilnih aplikacij. Najbolj uveljavljeni operacijski sistemi so Android, iOS, Windows Phone, BlackBerry in Symbian.

Kot je razvidno s slike 2.4, ima največji prodajni delež operacijski sistem Android. Pred prihodom le-tega na trg mobilnih operacijskih sistemov je prevladoval operacijski sistem Symbian. Konec leta 2010 je Android, ki mu delež hitro raste, po prodaji prehitel padajoči Symbian. Prodajni delež Applovega iOSa od prihoda na trg konstantno raste, medtem ko prodaja BlackBerrya pada, Microsoftov Windows Phone pa je še razmeroma svež

operacijski sistem, kar je tudi eden od razlogov za njegov majhen prodajni delež.

Android

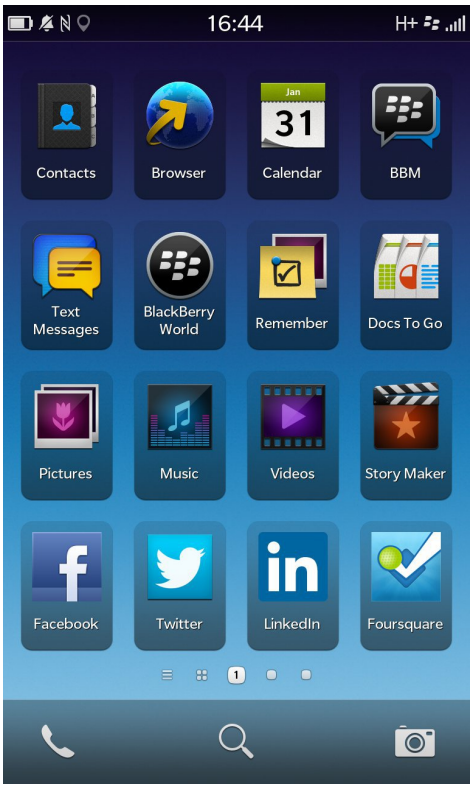
Android (slika 2.5) je zastonj in odprtokodni (angl. *open source*) operacijski sistem. Od njegovega začetka leta 2003 je za njegov razvoj skrbelo podjetje Android Inc., ki je bilo v večini financirano s strani Googla. Ta je leta 2005 kupil to podjetje, tako da je trenutno najbolj popularen mobilni operacijski sistem postal njihova last. Prvi pametni telefon z operacijskim sistemom Android je bil HTC Dream. Odkar je Android leta 2008 prišel na trg, njegov tržni delež hitro raste [13].

Slika 2.5: Operacijski sistem Android.

Slika 2.6: Operacijski sistem iOS.

iOS

iOS (slika 2.6) je produkt in last podjetja Apple in je namenjen izključno Applovim produktom iPhone, iPad, iPod in Apple TV. Prva verzija je prišla na trg leta 2007 z napravo iPhone prve generacije.



Slika 2.7: Operacijski sistem BlackBerry.

Slika 2.8: Operacijski sistem Windows Phone.

BlackBerry

BlackBerry (slika 2.7) je produkt in last podjetja BlackBerry Limited. Trenutna verzija operacijskega sistema je BlackBerry 10. Ta operacijski sistem poganjajo samo njihove lastne naprave BlackBerry, kar je glavni razlog za njegov zelo majhen tržni delež [13]. Prvi pametni telefon z operacijskim sistemom BlackBerry je bil Pager BlackBerry 580, ki je prišel na trg leta

Slika 2.9: Operacijski sistem Symbian.

1999 [2].

Windows Phone

Windows Phone (slika 2.8) je mlad operacijski sistem, saj je, kot produkt podjetja Microsoft, prišel na trg v začetku leta 2010. Je zaprtokodni operacijski sistem [13].

Symbian

Symbian (slika 2.9) izhaja iz operacijskega sistema EPOC, ki je last podjetja Psion. Podjetje Symbian Ltd. je leta 1998 ustanovilo partnerstvo med Ericssonom, Motorolo in Nokio. Leta 2008 je Nokia odkupila delež od ostalih partneric in tako postala 100-odstotni lastnik podjetja Symbian. Hkrati z nakupom je odprla kodo, kot odgovor odprtokodnemu operacijskemu sistemu Android, kar pa se je izkazalo za napačno potezo, saj je danes Symbian

mrtev. Zadnji pametni telefon s tem operacijskim sistemom je bil Nokia 808 PureView, ki je prišel na trg v začetku leta 2013 [17].

2.2 Mobilni operacijski sistem Android

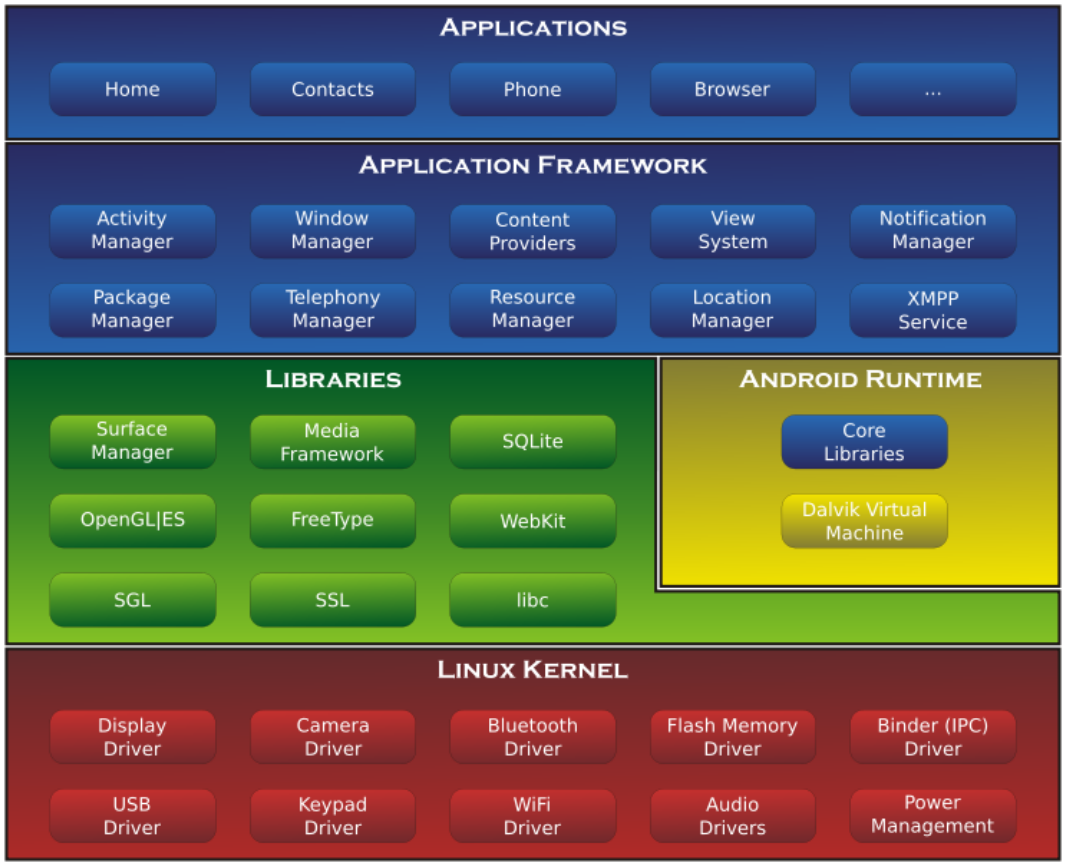
V tem poglavju je opisana arhitektura operacijskega sistema Android in njegovo delovanje. Na koncu poglavja so podrobno opisane ključne komponente in tehnologije sistema ter tiste, ki so bile uporabljene pri razvoju aplikacije RouteTracker.

Android je trenutno najbolj razširjen operacijski sistem za mobilne naprave. Prisoten je na več sto milijonov napravah v več kot 190 državah. Dnevno se število novih naprav poveča za skoraj milijon. Ker je Android zastonj in odprtokoden sistem ter nudi odlično podprto in dokumentirano platformo za razvoj in distribucijo aplikacij, ima največjo in najhitreje rastočo bazo aplikacij, ki so plačljive ali zastonjske [1].

2.2.1 Arhitektura

Mobilni operacijski sistem Android je sestavljen iz jedra odprtokodnega operacijskega sistema Linux in zbirke knjižnic, napisanih v programskem jeziku C/C++, ki prek različnih programskih ogrodij (angl. *application framework*) nudijo programske vmesnike in servise aplikacijam.

Kot je razvidno s slike 2.10, arhitektura operacijskega sistema Android sestoji iz petih nivojev, ki je vsak posebej sestavljen iz večih programskih komponent. Nivoji od spodaj navzgor so jedro operacijskega sistem Linux (angl. *Linux kernel*), knjižnice (angl. *libraries*), izvajalno okolje (angl. *Android runtime*), aplikacijsko ogrodje (angl. *application framework*) in aplikacije (angl. *applications*). V nadaljevanju sledi opis posameznih nivojev.



Slika 2.10: Arhitektura operacijskega sistema Android.

Jedro operacijskega sistema Linux

Na dnu arhitekture Androida se nahaja jedro Linuxa. Ta plast predstavlja temelj operacijskega sistema. Prva verzija Androida je temeljila na jedru verzije 2.6., zadnja različica pa vsebuje jedro 3.4.

Za potrebe delovanja Androida je s strani Googla jedro doživelo nekatere arhitekturne spremembe.

Glavne naloge jedra so upravljanje s pomnilnikom, procesi in porabo energije, skrb za varnost, podpora omrežnim povezavam, itd. Poleg naštetega, jedro s svojo zbirko gonilnikov predstavlja abstrakcijsko plast med strojno opremo in ostalimi deli operacijskega sistema.

Knjižnice

Nad jedrom operacijskega sistema se nahaja plast knjižnic. Android vsebuje veliko različnih C/C++ knjižnic, ki aplikacijam na višjih plasteh nudijo dostop do strojne opreme in funkcionalnosti jedra. Nekateri pomembni paketi knjižnic so:

- Knjižnici SSL in WebKit nudita podporo spletnemu brskalniku in varnosti.
- Standardna knjižnica programskega jezika C, libc.
- Grafični knjižnici SGL in OpenGL sta namenjeni grafični podpori.
- Podpora podatkovnim bazam SQL, SQLite, zagotavlja podporo za dostop, kreiranje in spreminjanje podatkovnih baz.
- Multimedijske knjižnice zagotavljajo kodeke (angl. *media codecs*), ki skrbijo za pravilno predvajanje zvoka in slike.

Izvajalno okolje

Izvajalno okolje se nahaja v isti plasti kot knjižnice. Vsebuje jedrne knjižnice (angl. *Core Libraries*) in navidezni stroj Dalvik (DVM), s čimer poskrbi za iz-

vajanje aplikacij in skupaj s predhodno opisanimi C/C++ knjižnicami tvori osnovo za aplikacijsko ogrodje. Jedrne knjižnice nudijo večino funkcionalnosti, ki so specifične za Android in ki so na voljo v standardnih javanski knjižnicah.

Navidezni stroj Dalvik je zasnovan tako, da na isti napravi nudi sočasno učinkovito poganjanje več aplikacij ali več instanc iste aplikacije. Vsaka aplikacija v operacijskem sistemu Android teče v lastnem procesu, ki ga upravlja posamezna instanca navideznega stroja Dalvik. S tem je vsaka aplikacija izolirana od ostalih, s čimer se doseže visoka stopnja varnosti, učinkovito upravljanje s pomnilnikom in večnitno delovanje. To ni javanski navidezni stroj (JVM), ampak Googlov posebej razvit in prilagojen navidezni stroj, ki je sestavni del operacijskega sistem Android. DVM poganja izvajalne datoteke (angl *Dalvik executables*) v formatu .dex, ki je prilagojen za učinkovito rabo pomnilnika. Za svoje učinkovito delovanje koristi v jedru Linuxa implementirane funkcionalnosti, kot so varnost, upravljanje s pomnilnikom in komuniciranje s strojno opremo.

Aplikacijsko ogrodje

Aplikacijsko ogrodje je sestavljeno iz javanskih razredov, ki nudijo vso podporo razvijalcu pri razvoju aplikacij. Razvijalec lahko kreira in dodaja nove razrede ali nadgrajuje obstoječe. Ogrodje predstavlja tudi API vmesnik za dostop do strojne opreme, upravljanje z uporabniškim vmesnikom (GUI) ter aplikacijskimi viri. Android obravnava prednameščene ter dodatno nameščene in razvite aplikacije enako, saj vse aplikacije za svoje delovanje uporabljajo razrede iz aplikacijskega ogrodja.
Nekaj pomembnejših razredov aplikacijskega ogrodja:

- Upravnik aktivnosti (angl. *Activity Manager*) - skrbi za življenjski cikel aktivnosti, njihovih fragmentov (angl. *fragments*) ter za upravljanje s skladom aktivnosti in fragmentov.

- Pogledi (angl. *Views*) - so posamezni gradniki uporabniškega vme-

snika aktivnosti in njihovih fragmentov. Razvijalec lahko uporablja obstoječe poglede, lahko kreira svoje, bodisi prilagodi obstoječe bodisi kreira popolnoma nove.

- Upravnik z obvestili (angl. *Notification Manager*) - zagotavlja nemoteč mehanizem za upravljanje z obvestili in njihov prikaz.

- Ponudniki vsebin (angl. *Content Providers*) - omogočajo souporabo podatkov med aplikacijami.

- Upravljavec z viri (angl. *Resource manager*) - upravlja z zunanjimi viri aplikacij, kot so slike, ikone, nizi znakov, zvoki, animacije, itd.

- Namere (angl. *Intents*) - mehanizem za prenašanje podatkov in ukazov med aplikacijami in njihovimi komponentami. Je neke vrste medprocesni komunikacijski sistem (angl. *interprocess communication*).

Aplikacije

Vse aplikacije, prednameščene s strani Androida in dodatno nameščene s strani uporabnika, se nahajajo v plasti aplikacije in uporabljajo enako verzijo APIja. Aplikacije tečejo v izvajalnem okolju in uporabljajo razrede in servise iz aplikacijskega ogrodja.

2.2.2 Podroben opis komponent in tehnologij, uporabljenih pri razvoju aplikacije RouteTracker

V tem poglavju so poleg ključnih gradnikov in tehnologij sistema Android opisani tudi tisti, ki sem jih uporabil pri razvoju mobilne aplikacije RouteTracker. Vsaka aplikacija sestoji iz izvorne kode Java, ki je zgrajena iz ključnih gradnikov, zunanjih virov (angl. *resources*), kot so slike, ikone, nizi znakov, itd. in datoteke AndroidManifest.xml.

AndroidManifest.xml

V datoteki AndroidManifest.xml so zapisane vse potrebne informacije, ki jih Android rabi za zagon aplikacije. Datoteka se nahaja v korenu projekta aplikacije. Na njeni podlagi deluje tudi sistem nalaganja aplikacij iz Googlove spletne trgovine Google Play. Če ciljna naprava ne ustreza vsem zahtevam, ki so določene v datoteki AndroidManifest.xml, storitev Google Play onemogoči nalaganje aplikacije na to napravo. V nadaljevanju so naštete nekatere informacije in zahteve, ki so zapisane v datoteki AndroidManifest.xml.

- Ime javanskega paketa, v katerem se aplikacija nahaja
  Ime določi edinstven identifikator aplikacije v sistemu.

- Opis vseh komponent aplikacije
  Vse komponente, ki so del aplikacije, morajo biti navedene v manifest datoteki. Komponente so lahko aktivnosti, storitve, ponudniki vsebin ter sprejemniki namer. Poleg navedbe posamezne komponente je potrebno določiti še razred, ki implementira to komponento, in lastnosti komponente, kot so: namere, ki jih lahko komponenta sprejme, lastnosti uporabniškega morebitnega vmesnika, ime in identifikator komponente, itd.

- Filtri namer
  V filtrih namer so določene operacije, ki jih aplikacija zna postreči.

- Proces, ki gosti komponente aplikacije
  Vsaka aplikacija v sistemu Android se starta v svojem procesu. Vse komponente aplikacije živijo v tem procesu, razen če v manifest datoteki ni drugače določeno.

- Pravice aplikacije
  Aplikacija mora izrecno podati pravice, do katerih funkcij operacijskega sistema in strojne opreme dostopa med svojim delovanjem. Med pravicami so določene tudi pravice, ki jih morajo imeti aplikacije, ki želijo imeti dostop do posameznih komponent aplikacije.

- Minimalna verzija *API* vmesnika
  S tem je določena minimalna verzija operacijskega sistema Android, na kateri aplikacija še pravilno deluje.

- Knjižnice
  Zunanje knjižnice, ki jih aplikacija rabi za svoje delovanje.

Namera

Namera (angl. *intent*) je pasiven objekt, ki je namenjen zaganjanju novih aplikacij, komponent aplikacij ali prenašanju podatkov. Pasiven je zato, ker sam po sebi nič ne naredi, ampak proži neko aplikacijo ali komponento, ki je zmožna opraviti nalogo, ki jo namera vsebuje. Namera lahko poleg naloge vsebuje tudi podatke, ki so potrebni za uspešno izvršitev naloge.

Namere, ki so namenjene zaganjanju aplikacij ali komponent, se delijo v dve kategoriji:

- Eksplicitne namere
  Namere eksplicitnega tipa imajo ciljno komponento določeno z imenom. Uporabljajo se za pošiljanje podatkov in ukazov znotraj aplikacije. Eksplicitna namera lahko zažene določeno aktivnost aplikacije, servis, itd.

- Implicitne namere
  Implicitne namere vsebujejo informacijo o operaciji, ki jo je potrebno izvršiti. Te namere so namenjene pošiljanju ukazov s podatki po celotnem sistemu Android. Android na podlagi manifest datotek vseh nameščenih aplikacij ter na podlagi filtrov namer določi, katera aplikacija oziroma komponenta aplikacije je primerna za izvršitev te operacije. Če je možnih aplikacij več, se izbira ponudi uporabniku.

Poleg zaganjanja aplikacij in komponent namera lahko prenaša podatke ali dogodke po sistemu, kot je sprememba lokacije mobilne naprave. Za to funkcionalnost se uporablja čakajoče namere (anlg. *pending intent*). Aplikacija čakajočo namero s določenim podatkom posreduje neki drugi aplikaciji.

Ko ima ta druga aplikacija zahtevane podatke na voljo, jih prva aplikacija dobi prek sprejemnika namer.

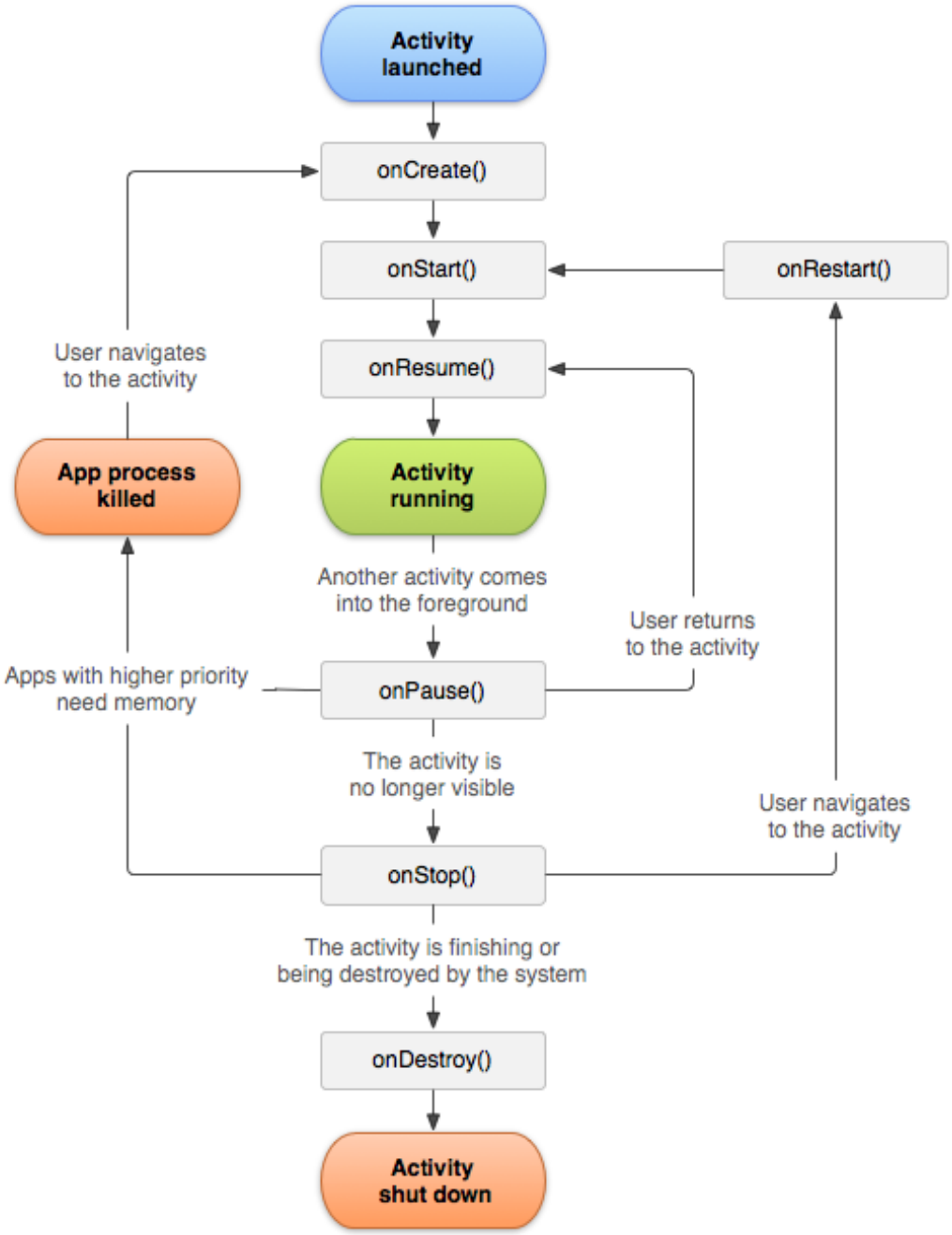
Aktivnost

Aktivnost (angl. *activity*) je ključen gradnik aplikacije, katerega glavna značilnost je uporabniški vmesnik, ki omogoča uporabniku interakcijo z aplikacijo. Uporabniški vmesnik je definiran z xml datoteko, ki pripada zunanjim virom (angl. *external resources*). Aplikacija je največkrat sestavljena iz večih aktivnosti, saj je vsak zaslon aplikacije svoja aktivnost. Aktivnosti med seboj komunicirajo z namerami (angl. *intents*). Če je aktivnost dialog, zasede le del zaslona, v nasprotnem primeru celega. Vsaka aktivnost mora biti navedena v datoteki AndroidManifest.xml, ki pripada aplikaciji, katere del je aktivnost.

Ker je aktivnost najbolj osnovna aplikacijska komponenta, je vedenje aplikacije tesno povezano z življenjskim ciklom aktivnosti, ki je prikazan na sliki 2.11. Življenjski cikel je sestavljen iz stanj Created, Started, Resumed, Paused, Stopped in Destroyed. Prehodi med temi stanji so implementirani v povratnih (angl. *callback*) funkcijah, ki ji kliče sistem Android, ko se stanje aktivnosti spremeni. Primer takega klica je ob pritisku gumba za nazaj (angl. *back button*), ko se pokličeta funkciji onPause() in onStop(). Stanje aktivnosti se spremeni iz Resumed v Paused in nato v Stopped. Aktivnost se v temu primeru shrani na sklad aktivnosti (angl *back stack*). V nadaljevanju so opisane povratne funkcije, ki so vidne na sliki 2.11. Naloga razvijalca je, da po potrebi dopolni funkcije tako, da bo aplikacija pravilno delovala in pravilno uporabljala strojne in programske vire naprave.

Funkcija **onCreate()** se pokliče, kadar je aktivnost prvič kreirana. V tej funkciji se inicializirajo vse potrebne komponente aplikacije in kreirajo pogledi (angl. *views*) uporabniškega vmesnika, ki je definiran v xml datoteki.

Funkcija **onRestart()** se pokliče ob ponovnem zagonu aktivnosti, potem



Slika 2.11: Življenjski cikel aktivnosti.

ko je bila ustavljena.

Funkcija **onStart()** se pokliče, tik preden aktivnost postane vidna.

Funkcija **onResume()** funkcija se pokliče, tik preden postane aktivnost uporabna za uporabnika.

Funkcija **onPause()** se pokliče, kadar je Android tik pred tem, da zažene novo ali aktivnost, ki je na pavzi.

Funkcija **onStop()** funkcija se pokliče, kadar aktivnost uporabniku ni več vidna. Aktivnost se shrani na sklad aktivnosti.

Funkcija **onDestroy()** funkcija se pokliče, kadar je aktivnost uničena. Aktivnost se odstrani s sklada aktivnosti. To se zgodi v primeru pomanjkanja delovnega spomina.

Aktivnosti so lahko sestavljene iz več fragmentov (angl. *fragments*). Fragmenti in njihova povezava z aktivnostmi so opisani v enem izmed naslednjih poglavij.

Storitev

Storitev (angl. *service*) je komponenta, ki teče v ozadju in opravlja opravila, ki za svoje delovanje ne rabijo uporabniškega vmesnika ali pa so del oddaljenega procesa. V nasprotju z aktivnostjo storitev nima uporabniškega vmesnika. Storitev se zažene s strani aktivnosti in ostane živa tudi potem, ko je aktivnost ustavljena. Poleg tega se lahko katerakoli druga aktivnost priklopi na storitev in si z njo izmenjuje podatke. Primer uporabe storitev je poslušanje glasbe in hkratno brskanje po spletu. Storitev v ozadju predvaja glasbo, medtem ko aktivnost prikazuje podatke na spletu.

Ponudniki vsebin

Ponudniki vsebin (angl. *content providers*) omogočajo izmenjevanje podatkov med aplikacijami ter shranjevanje in branje podatkov. Podatke se lahko shrani na datotečni sistem, splet ali bazo SQL. Tipični primer uporabe ponudnikov vsebin je dostopanje do imenika telefonskih podatkov in spreminjanje kontakta.

Sprejemnik namer

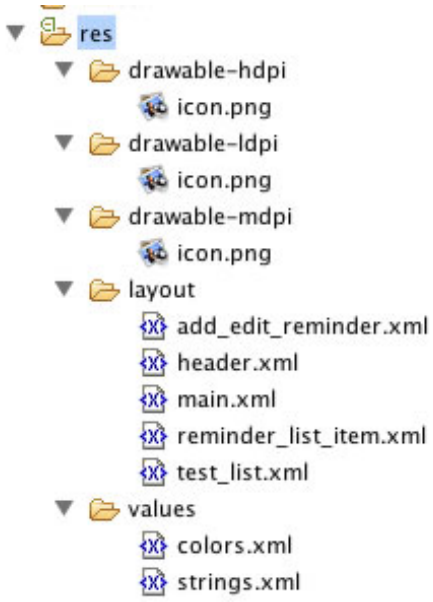
Sprejemnik namer (angl. *broadcast receiver*) je komponenta, ki sprejema in se odzove na dogodke (namere) v sistemu, kot so prazna baterija, prejet klic, prejeto sporočilo, itd. Namero lahko generira tudi katerakoli druga aplikacija, ki s tem pove, da se je nek dogodek zgodil. Tak primer je aplikacija RouteTracker, ki ima implementiran sprejemnik, ki sprejema nove lokacije naprave. Sprejemniki namer nimajo uporabniškega vmesnika, lahko pa kreirajo opozorilo v statusni vrstici.

Fragment

Kot je že bilo omenjeno v podpoglavju 2.2.2, lahko aktivnost sestavlja več framgmentov (angl. *fragments*). Fragmenti so deli uporabniškega vmesnika, ki so lahko uporabljeni v več aktivnostih. Izgled fragmenta je določen v njegovi xml datoteki, njegovo obnašanje pa je implementirano v razredu, ki implementira fragment. Tako kot aktivnost, ima tudi fragment svoj življenjski cikel, ki je tesno povezan z življenjskim ciklom aplikacije, v kateri gostuje.

Zunanji aplikacijski viri

Kot je že bilo omenjeno, aplikacijo sestavlja izvorna koda, manifest datoteka in zunanji aplikacijski viri (angl. *Application Resources*). Ti viri so slike, animacije, ikone, melodije, definicije menijev, uporabniških vmesnikov, fragmentov, barv, stilov, itd. Definicije se nahajajo v ustreznih xml datotekah.



Slika 2.12: Drevesna struktura zunanjih aplikacijskih virov.

Kot je razvidno s slike 2.12, se viri v organizirani drevesni strukturi v korenu projekta aplikacije.

V mapah *drawable* se nahajajo ikone aplikacije. Različne mape so za različne ločljivosti zaslonov. Datoteke xml, ki definirajo izgled uporabniških vmesnikov aktivnosti in fragmentov, so v mapi *layout*, v mapi *values* pa so definicije barv, stilov in nizov. V času prevajanja projekta vsak vir dobi enoličen identifikator, prek katerega se dostopa do vira med izvajanjem aplikacije. Na ta način lahko menjamo ikone, beremo podatke, ki jih uporabnik vnaša, itd.

Google Maps

Knjižnica Google Maps je na voljo kot del storitev Google Play. V primeru uporabe Google Maps v aplikaciji je na razvojnem računalniku potrebno imeti naložen Google Play services SDK in ga pravilno vključiti v projekt aplikacije. Za pravilno delovanje Googlovih zemljevidov mora imeti aplikacija dostop do strežnikov Google Maps, zato pa rabi aplikacija Maps API

ključ. Potrebni ključ se pridobi v Google API konzoli (angl. *console*) na naslovu *https://code.google.com/apis/console/* na podlagi digitalnega ključa, s katerim je podpisana aplikacija. Poleg pridobljenega ključa je potrebno v manifest datoteko aplikacije dodati še nekatere pravice za pravilno delovanje Googlovih zemljevidov in zahtevo za podporo knjižnici OpenGL.

Google Maps se doda v projekt kot fragment. S tem je poskrbljeno za vso interakcijo z zemljevidom. V aplikaciji je dovolj samo definirati, kje naj bo zemljevid prikazan, oziroma, kje naj se fragment nahaja. S fragmentom je poskrbljeno tudi za osnovno interakcijo z zemljevidom, kot je približevanje, rotiranje in premikanje. Prek vmesnika API lahko aplikacija sama kontrolira vse te dogodke, riše poljubne objekte na zemljevid in še veliko več.

2.3 Programska orodja

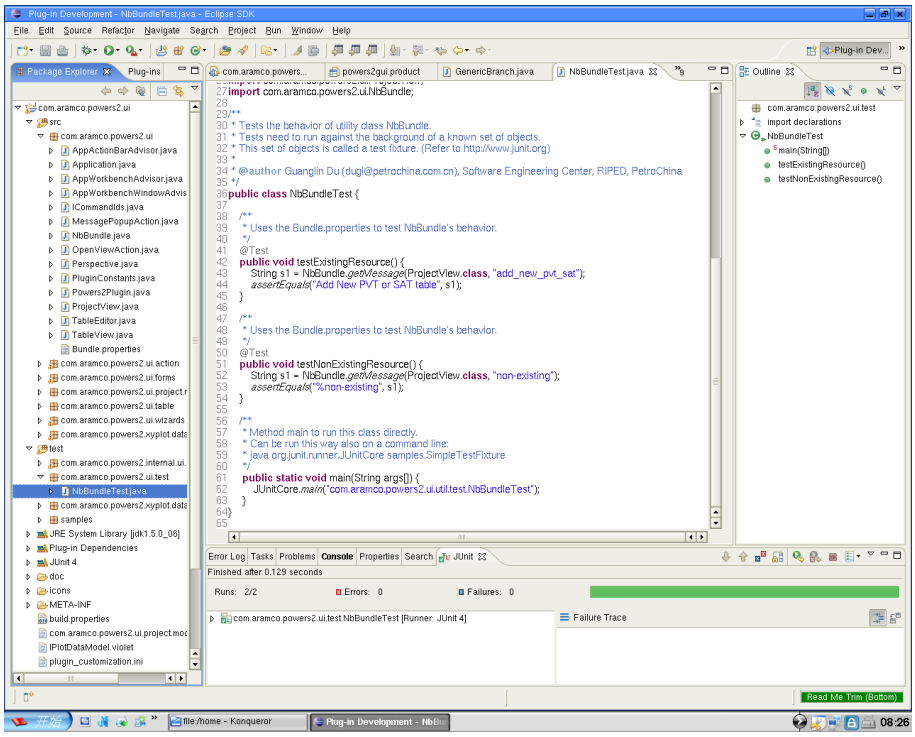
V tem poglavju so opisana nekatera nujna orodja in tehnologije, ki jih rabimo za razvoj aplikacije za operacijski sistem Android.

Aplikacije operacijskega sistema Android so napisane v programskem jeziku Java, zato je potrebno imeti na računalniku naložen razvijalski paket za Javo (angl. *Java software development kit - JDK*). Poleg paketa za Javo je potrebno namestiti tudi paket (angl. *Android SDK*) , ki je potreben za razvoj na platformi Android. Razvijalec lahko piše programsko kodo v poljubnem urejevalniku besedil, prevede, testira in razhroščuje pa jo z orodji iz paketa Android SDK.

Pri razvijalcih je zelo popularno razvojno okolje Eclipse z nameščenim vtičnikom ADT, ki nudi profesionalno okolje za razvoj aplikacij sistema Android. Leta 2013 je Google predstavil svoje razvojno okolje Android Studio, ki je po njihovih besedah najboljše razvojno okolje za platformo Android.

2.3.1 Eclipse

Eclipse (slika 2.13) je odprtokodno integrirano razvojno okolje (angl. *Integrated Development Enviroment - IDE*), ki temelji na razširljivem sistemu



Slika 2.13: Razvojno okolje Eclipse.

vtičnikov (angl. *plug-ins*). Zaradi svoje razširljivosti je večjezikovno razvojno okolje, kar je poleg odprtokodnosti velik razlog za njegovo veliko popularnost. IBM je v začetku razvijal Eclipse kot novo razvojno okolje za platformo Java, vendar se je vodstvo kasneje odločilo, da projekt odpre in ga ponudi javnosti. Tako je bila januarja leta 2004 ustanovljena fundacija Eclipse Foundation [5].

Za potrebe razvoja aplikacij za platformo Android je Google razvil vtičnik ADT (angl. *Android Development Tools*), ki v Eclipse integrira orodja, ki omogočajo hitrejši in udobnejši razvoj aplikacij. ADT vsebuje grafične vmesnike za orodja iz paketa Android SDK, emulator androidnih naprav (slika 2.14) ter grafični vmesnik za oblikovanje uporabniškega vmesnika (GUI).

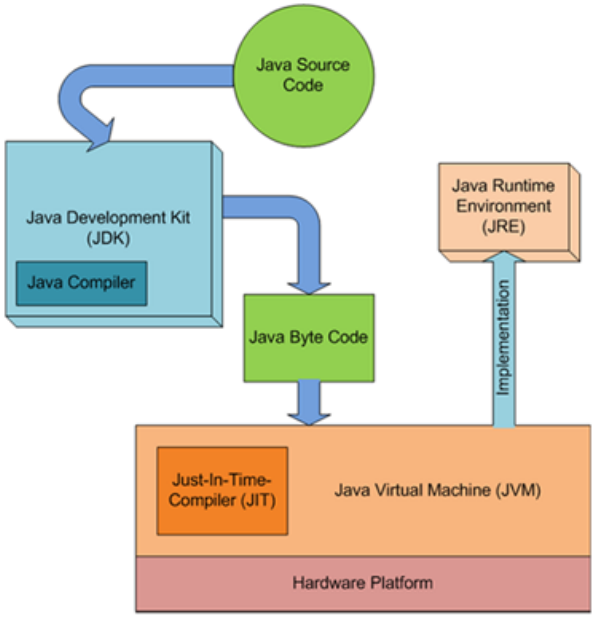
Slika 2.14: Emulator naprav Android.

2.3.2 Java

Java je objektno usmerjen programski jezik, ki ga je razvilo podjetje Sun Microsystems. Razvoj projekta, ki je bil zasnovan kot zamenjava za programski jezik C++, se je začel leta 1991. Prva verzija Jave je bila objavljena leta 1996. Trenutna verzija Jave je 6.0 in je izšla leta 2007 [15].

Ena izmed značilnosti in prednosti programskega jezika Java je prenosljivost. Program, napisan v Javi, deluje na katerikoli strojni platformi, ki ima nameščen javanski navidezni stroj (angl. *Java Virtual Machine*).

Na sliki 2.15 je predstavljena platforma Java. Izvorna koda programa je s prevajalnikom, ki je del javanskega razvojnega paketa (angl. Java Development Kit), prevedena v bitno kodo. Datoteka z bitno kodo ima končnico .class. Ta se izvaja na javanskem navideznem stroju, ki bitno kodo interpretira v strojno kodo konkretnega procesorja, na katerem se program poganja, in v javanskem izvajalnem okolju (angl. *Java Runtime Enviroment*). Interpretiranje je razlog, da so javanski programi počasnejši od tistih, ki so napisani v C/C++. Mehanizem sprotnega prevajanja (JIT) izvajanje pohitri, saj se interpretirana koda shranjuje v pomnilnik. Ob naslednjem klicu



Slika 2.15: Platforma Java.

že interpretirane kode se njena interpretacija prebere iz pomnilnika.

2.3.3 Android SDK

Paket orodij za razvoj androidnih aplikacij (Android SDK) je osnova in vse, kar razvijalec potrebuje za razvoj aplikacij za platformo Android. Paket je tako kot Android brezplačen. V nadaljevanju so opisane nekatere ključne komponente paketa.

- **Knjižnice Android API** so knjižnice, ki se nahajajo v plasti Programska ogrodja. Razvijalcem aplikacij omogočajo dostop do komponent operacijskega sistema. Z vsako novo verzijo Androida je na voljo tudi nova verzija knjižnic API, ki vsebujejo nove ali izboljšane funkcionalnosti.

- **Razvojna orodja** (anlg. *development tools*) so orodja, ki so v pomoč razvijalcu pri razvoju, prevajanju, razhroščevanju, optimiziranju in di-

stribuiranju aplikacij.

- **Dokumentacija** delovanja konstruktov ter mehanizmov in knjižnic Androida. Vključuje vse informacije o tem, kaj posamezni paketi in knjižnice vsebujejo ter kako se jih uporablja. V dokumentaciji so opisana tudi navodila za kreiranje izgleda aplikacije in dobre prakse programiranja aplikacij.

- **Demonstracijski primeri** so enostavne aplikacije za Android, ki prikazujejo uporabo nekaterih najbolj uporabljenih konstruktov operacijskega sistema Android.

2.4 Ostalo

V tem poglavju je opisana tehnologija za pridobivanje lokacije in mobilna naprava, na kateri sem testiral aplikacijo RouteTracker. Ker ne tehnologija za pridobivanje lokacije ne naprava ne spadata med programska orodja, mobilne tehnologije ali v operacijski sistem Android, sem ju opisal v posebnem poglavju.

2.4.1 Tehnologije za pridobivanje lokacije

Mobilne naprave imajo ponavadi dva sistema za pridobivanje trenutne lokacije: prvi je sistem globalnega pozicioniranja (angl. *Global Positioning System - GPS*), drugi pa sistem pridobivanja lokacije na podlagi mobilnega omrežja ali brezžičnih dostopnih točk. Vsak sistem ima svoje prednosti in slabosti. Sistem GPS je zelo točen in hiter, vendar ne deluje v zaprtih prostorih in je velik porabnik energije. Sistem, ki deluje na podlagi mobilnega ali brezžičnega omrežja, v nasprotju s sistemom GPS nemoteno deluje tudi v zaprtih prostorih, porabi manj energije, vendar so njegove meritve manj točne.

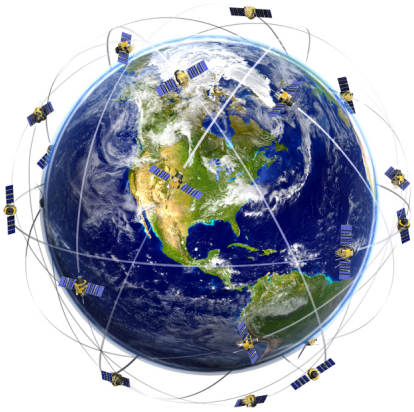
Sistem globalnega pozicioniranja - GPS

Sistem globalnega pozicioniranja (angl. *Global Positioning System - GPS*) je satelitski navigacijski sistem, ki se uporablja za določanje točne lege in časa kjerkoli na Zemlji. Za vojaške namene ga je sprva zasnovalo obrambno ministrstvo Združenih držav Amerike, nato pa ga je ponudilo v brezplačno uporabo vsakomur, ki ima ustrezen sprejemnik GPS [7]. S prihodom pametnih mobilnih naprav je postal dostopen praktično vsakomur. Pred pametnimi mobilniki je bila njegova uporaba omejena samo na drage namenske naprave.

Sistem sestavlja najmanj 24 satelitov v šestih ravninah tirnic. Slika 2.16 prikazuje razporeditev satelitov okrog Zemlje. Sateliti so okrog Zemlje razporejeni tako, da so v vsakem trenutku z vsake lokacije na Zemlji vidni najmanj štirje. Vsak od njih Zemljo obkroži dvakrat dnevno na višini 20200 metrov in ima nameščeno atomsko uro, ki zagotavlja točen čas. Satelit neprestano oddaja čas in podatke o tirnici gibanja. Za pridobitev podatkov o lokaciji, ki je sestavljena iz zemljepisne širine in dolžine, nadmorske višine ter točnega časa, sprejemnik potrebuje signale najmanj štirih satelitov. Iz razlike med časom sprejema signala in časom njegove oddaje lahko določimo razdaljo med sprejemnikom in satelitom. Nato iz njihovih signalov in notranje baze podatkov ugotovimo mesta satelitov. Sprejemnik se torej nahaja na sferi, katere središče je satelit in katere polmer je določen z razdaljo med satelitom in sprejemnikom. Ker sprejemnik sprejema signale z večih satelitov, se njegovo točno lokacijo določi kot presečišče vseh sfer satelitov, s katerimi sprejemnik komunicira.

Sistem za pridobivanje lokacije na podlagi mobilnega omrežja ali brezžičnih dostopnih točk

Sistem za pridobivanje lokacije na podlagi mobilnega omrežja deluje na podlagi lokacij oddajnikov. Naprava pridobi lokacijo na osnovi IDja oddajnika, na katerega je povezana. ID trenutnega oddajnika in preteklih oddajnikov se pošlje Googlovi lokacijski storitvi (angl. *Google location service*), ki vrne



Slika 2.16: Sateliti GPS.

lokacijo telefona. Ker se lokacija določa s triangulacijo, je rezultat bolj točen, če se pošlje IDje vsaj treh oddajnikov.

Na podobnem principu, kot deluje sistem pridobivanja lokacije na podlagi mobilnih oddajnikov, deluje sistem za pridobivanje lokacije na podlagi brezžičnih dostopnih točk. Mobilna naprava pošlje strojni naslov (MAC) dostopne točke, na katero je povezana, Googlovi lokacijski storitvi, ki na podlagi seznama dostopnih točk vrne lokacijo dostopne točke s podanim strojnim naslovom.

Googlova lokacijska storitev ima svojo bazo podatkov z IDji oddajnikov in njihovimi lokacijami. Isto velja za dostopne točke. Bazo teh podatkov vzdržujejo vse mobilne naprave z operacijskim sistemom Android. Ko uporabnik želi vklopiti pridobivanje lokacije na podlagi mobilnega omrežja ali brezžičnih dostopnih točk, se mora strinjati s tem, da mobilna naprava samodejno pošilja Googlovim strežnikom lokacijo, pridobljeno s sistemom GPS in IDje oddajnikov ali strojne naslove dostopnih točk. Če se s tem ne strinja, storitve ne more vklopiti.

2.4.2 Googlove mobilne naprave

Pri razvoju aplikacije RouteTracker sem za testiranje uporabljal Googlovo napravo Nexus 4. Zato sta v tem poglavju podrobno opisana Googlova linija naprav Nexus in mobilni telefon Google Nexus 4.

Google Nexus 4 je razvilo in proizvedlo podjetje LG. Je najnovejši mobilni telefon iz linije Nexus. Ima 4.7 palčni zaslon z ločljivostjo 768x1280 pik in IPS matriko. Poganjata ga 4-jedrni procesor ARM Cortex-A15 s taktom 1.5 GHz proizvajalca Qualcomm in grafična enota (GPU) Adreno 320. Zunanji pomnilnik je kapacitete 16GB in ni razširljiv z razširitveno kartico. Delovnega pomnilnika ima 2 GB [9].

Google Nexus je linija mobilnih naprav, katerih naročnik je podjetje Google. Na napravah Nexus je naložen operacijski sistem Android, brez modifikacij, kot smo jih navajeni pri napravah drugih proizvajalcev. Naprave iz linije Nexus so v prvi vrsti namenjene razvijalcem, saj Android 100-odstotno podpira njihovo strojno opremo in so kot prve deležne nadgradenj in posodobitev operacijskega sistema Android. Ker Google ni proizvajalec strojne opreme, naprave Nexus izdelujejo različni proizvajalci mobilnih naprav, s katerimi Google sklene pogodbo.

Seznam do sedaj izdanih naprav linije Nexus [8]:

- Mobilni telefoni
  - **Nexus One**
    Na trg je prišel maja 2010. Proizvedlo ga je podjetje HTC. Na njem je tekel Android 2.1 Eclair.
  - **Nexus S**
    Izdelal ga je Samsung, predstavljen je bil decembra 2010. Naložen je imel Android 2.3 Gingerbread.
  - **Galaxy Nexus**
    Telefon je tako kot Nexus S plod dela podjetja Samsung. Naprava, ki je bila novembra 2011 je imela naložen Android 4.0 Ice Cream Sandwich.

  - **Nexus 4**

    Novembra 2012 je bil predstavljen zadnji mobilni telefon iz linije Nexus, Nexus 4. Telefon, ki je plod dela podjetja LG, je poganjal operacijski sistem Android 4.2 Jelly Bean.

- Tablični računalniki

  - **prva generacija Nexus 7**

    Junija 2013 je Google predstavil prvi tablični računalnik iz njihove linije Nexus. Na tablici z zaslonom diagonale 7 palcev, ki jo je proizvedlo podjetje ASUS, je tekel Android 4.1.

  - **druga generacija Nexus 7**

    Leto po predstavitvi tablice Nexus 7 prve generacije, julija 2013, je luč sveta ugledala tablica Nexus 7 druge generacije, ki jo je prav tako proizvedlo podjetje ASUS. Naložen je bil operacijski sistem Android 4.3.

  - **Nexus 10**

    Prva in zaenkrat edina 10-palčna tablica iz linije Nexus je bila predstavljena oktobra 2012. Proizvedlo jo je podjetje Samsung, poganjala pa je Android 4.2.

Poglavje 3

Aplikacija RouteTracker

Razvoj aplikacije, ki sem jo razvil v razvojnem okolju Eclipse in testiral na realni napravi Google Nexus 4, je potekal v več stopnjah. Najprej sem definiral zahteve za aplikacijo, na osnovi katerih sem v grobem naredil raziskavo tehnologij in komponent operacijskega sistema Android, kot so sprejemnik namer (angl. *boradcast receiver*), storitev (angl. *service*), Googlovi zemljevidi ter opravilna vrstica z zavihki. Za vsako od naštetih komponent sem ustvaril demonstracijski projekt, v katerem sem preizkusil tehnologijo oziroma komponento. Na podlagi dobljenih rezultatov in zahtev za aplikacijo sem definiral arhitekturo aplikacije. Kodiranje aplikacije in implementacija uporabniškega vmesnika sta potekala v več fazah.

Prva faza je predstavljala implementacijo orodne vrstice z zavihki in vključitev Googlovih zemljevidov v projekt. Druga faza je zajemala implementacijo sistema za pridobivanje trenutne lokacije in risanje poti na zemljevidu. V tretji fazi pa je bilo implementirano shranjevanje poti na datotečni sistem, zajem fotografij ter ogled shranjenih poti in fotografij.

V naslednjih poglavjih sledi opis zahtev za aplikacijo, arhitektura aplikacije, sistem za pridobivanje trenutne lokacije naprave ter uporabniški vmesnik. Na koncu so navedene in opisane nekatere možne izboljšave aplikacije.

3.1 Zahteve za aplikacijo

Osnovni namen aplikacije RouteTracker je zajemanje lokacije naprave in risanje sledi gibanja naprave na zemljevidu. Sledi je možno dodati fotografijo, ki je prikazana na točki, na kateri je bila posneta. Sled in morebitne fotografije se shranijo na datotečni sistem. Začetek in konec zajemanja ter fotografiranje uporabnik proži iz uporabniškega vmesnika. Ob koncu zajemanja uporabnik izbere ime posnete poti, ki se shrani na datotečni sistem. Uporabnik lahko kasneje shranjene poti pregleduje in briše. V primeru prekinitve izvajanja aplikacije zaradi telefonskega klica ali kakšne druge aplikacije, ki preide v ospredje, se mora zajemanje lokacij in shranjevanje le-teh nemoteno nadaljevati. Ko uporabnik znova preklopi na aplikacijo RouteTracker in sproži konec snemanja, se zajemanje konča.

3.2 Arhitektura

Na podlagi v poglavju 3.1 naštetih zahtev je aplikacija RouteTracker smiselno razdeljena na tri komponente: sprejemnik lokacije, uporabniški vmesnik in generator poti. Vsaka komponenta ima svoje specifične naloge. Z izjemo slikanja in ogleda slike, ki se proži prek Androidovega mehanizma namer, so vse druge funkcionalnosti povezane med sabo z funkcijami, ki so definirane v vmesnikih komponent. Za upravljanje s kamero in ogledovanje slik so uporabljene Androidove ali dodatno nameščene aplikacije.

3.2.1 Sprejemnik lokacije

Ključna naloga aplikacije je zbiranje in shranjevanje lokacij mobilne naprave. V ta namen se uporabi sistemska storitev (angl. *system service*) Androida za pridobivanje lokacij naprave.

Sprejemnik lokacije je implementiran z Androidovo komponento sprejemnik namer (angl. *broadcast receiver*). Tipični lastnosti sprejemnika namer sta, da nima kontrole nad uporabniškim vmesnikom z izjemo statusne vrstice

```
//  get system location service
locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
```

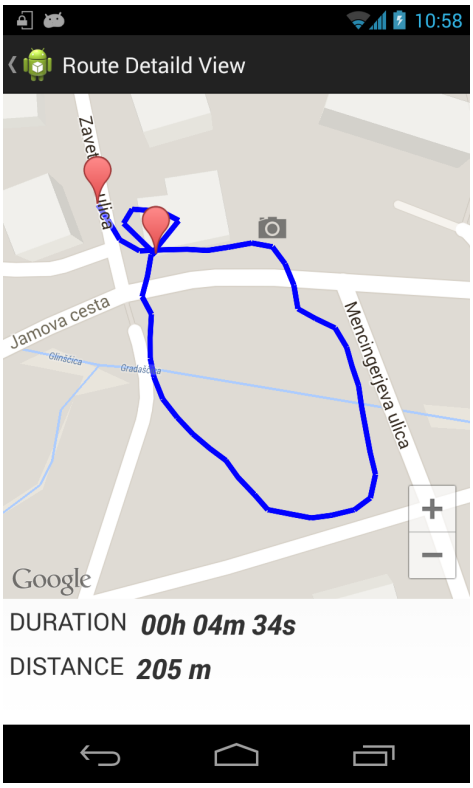
Slika 3.1: Pridobitev ročice upravitelja lokacij.

```
private void requestLocationUpdates(){

    if (!tracking)
    {
        pendingIntent = createPendingIntent();

        Criteria criteria = new Criteria();
        criteria.setAccuracy(Criteria.ACCURACY_COARSE);

        for (String provider : ((HomeScreen) parentActivity).getLocationManager().getProviders(criteria, true))
        {
            Log.d(TAG, "Enabling provider " + provider);
            ((HomeScreen) parentActivity).getLocationManager().requestLocationUpdates(provider,
                    300,                //  minimum time interval between location updates = 300 ms
                    5,                  //  minimum distance between location updates = 5 m
                    pendingIntent);
        }

        tracking = true;
    }
}
```

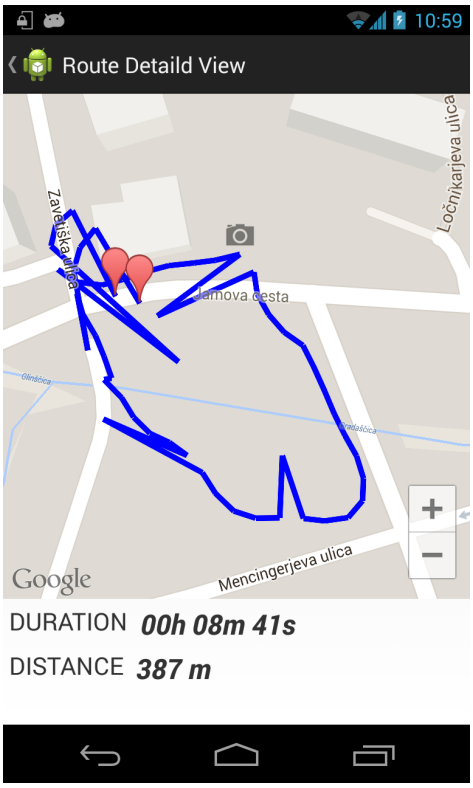
Slika 3.2: Zahteva za osveževanje lokacij.

in da nemoteno teče v ozadju, kar omogoča neprekinjeno zajemanje lokacij tudi ob prekinitvi zaradi recimo telefonskega klica.

Ob kreaciji aplikacije je potrebno v funkciji onCreate() pridobiti ročico do Androidovega upravljavca lokacijske storitve (slika 3.1). Ob začetku zajemanja lokacij naprave je od upravljavca potrebno zahtevati osveževanje podatka trenutne lokacije. Kot je razvidno s slike 3.2, se zahtevi poda minimalen čas med lokacijami in minimalno razdaljo med lokacijami. Ob koncu zajemanja je potrebno sprejemnik lokacij odjaviti iz Androidovega lokacijskega sistema. Android pošilja sveže lokacije sprejemniku namer v naši aplikaciji. Ta ima implementiran filter, ki na podlagi točnosti lokacije lokacijo zavrže ali pa jo posreduje naprej generatorju poti in uporabniškemu vmesniku, ki lokacijo prikaže na zemljevidu in nariše premico do prejšnje lokacije. Sliki 3.3 in 3.4 prikazujeta isto pot, prva je nastala s filtriranjem lokacij, druga brez filtriranja.



Slika 3.3: Primer poti s filtriranjem lokaciji.



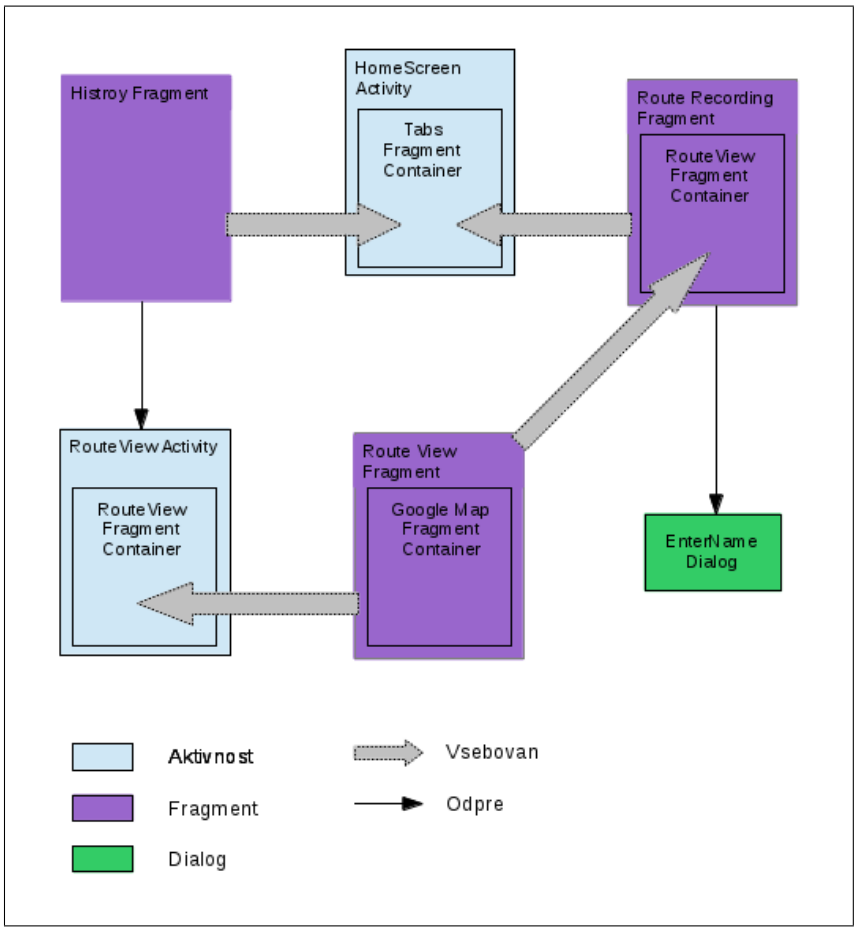
Slika 3.4: Primer poti brez filtriranja lokaciji.

3.2.2 Uporabniški vmesnik

Uporabniški vmesnik (GUI) je ključna komponenta aplikacije RouteTracker. Prek uporabniškega vmesnika uporabnik proži začetek in konec zajemanja poti ter dodajanje fotografije k poti. Iz uporabniškega vmesnika se lahko shranjene poti pregledujejo in brišejo. Ker je uporabniški vmesnik obsežna in ključna komponenta aplikacije, je opisan v poglavju 3.3.

3.2.3 Generator poti

Generator poti je enostavna komponenta, ki od sprejemnika lokacije prejme dovolj točno lokacijo in jo shrani v trenutno pot. Od uporabniškega vmesnika lahko prejme fotografijo, ki jo ravno tako shrani v pot, obenem pa jo poveže



Slika 3.5: Razdelitev uporabniškega vmesnika.

s trenutno lokacijo. Pot, ki jo ustvari generator poti, se ob koncu zajemanja zapiše v datoteko xml, ki se nato shrani na datotečni sistem.

3.3 Uporabniški vmesnik

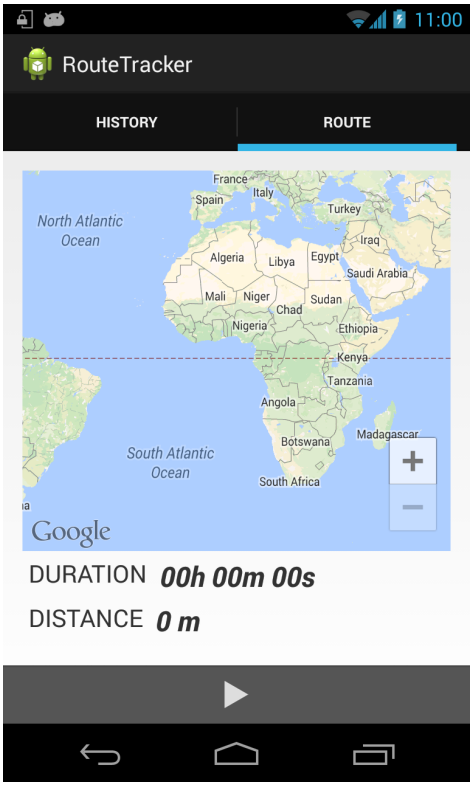
Uporabniški vmesnik je komponenta, prek katere uporabnik upravlja z aplikacijo in pregleduje rezultate aplikacije. Ker je najobsežnejša komponenta, je opisan v posebnem poglavju.

Kot je vidno na sliki 3.5, je uporabniški vmesnik aplikacije sestavljen iz dveh aktivnosti, treh fragmentov in enega dialoga.

Aplikacija RouteTracker je logično razdeljena na dva dela, snemanje poti ter pregledovanje in urejanje posnetih poti. Na podlagi te razdelitve je zasnovan tudi uporabniški vmesnik, ki ima dva zavihka, Route in History. Podroben opis zavihkov sledi v nadaljevanju.

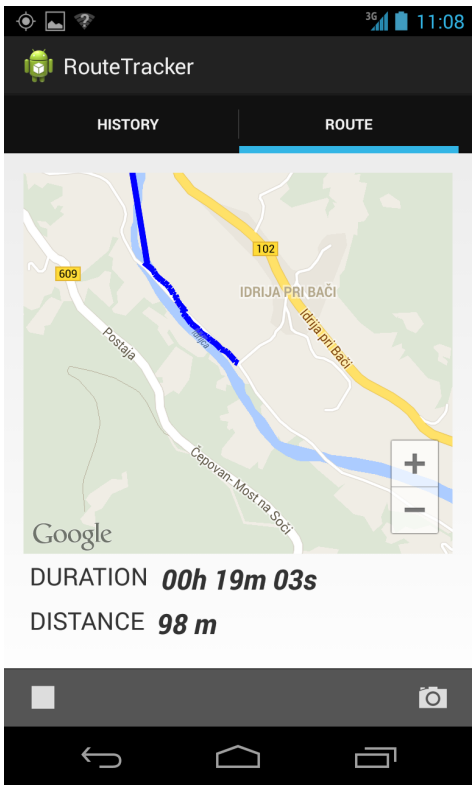
Aktivnost HomeScreen je vstopna točka aplikacije. Aktivnost ima samo en element in sicer TabsFragmentContainer, ki odvisno od izbranega zavihka vsebuje ali History ali RouteRecording fragment. Zavihki so implementirani s pomočjo opravilne vrstice, ki je razdeljena na dva dela. Na vrhu zaslona so zavihki, na spodnjem delu pa akcije, ki so specifične za zavihek, ki je prikazan. Če zavihek nima akcije, spodnji del opravilne vrstice ni prikazan. Opravilno vrstico se razčleni na dva dela z vnosom `android:uiOptions="splitActionBarWhenNarrow"` v datoteko AndroidManifest.xml.

Slika 3.6: Zavihek zgodovine.

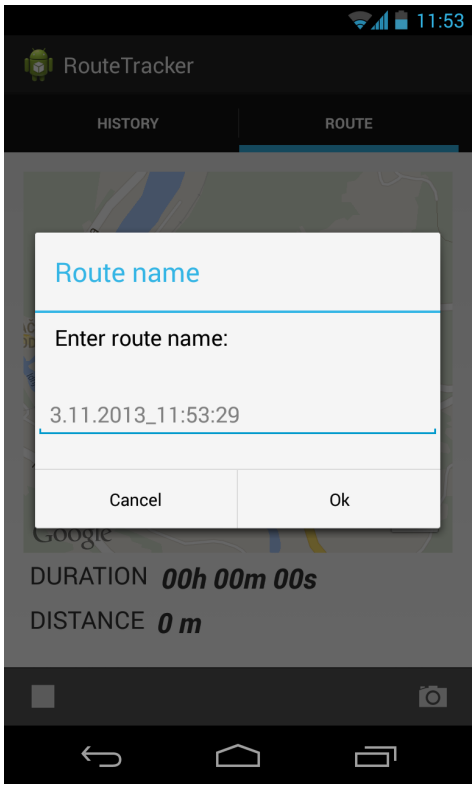


Slika 3.7: Zavihek poti.

3.3.1 Zavihek Route



Slika 3.8: Snemanje poti.



Slika 3.9: Dialog za izbiro imena poti.

Zavihek Route je namenjen upravljanju snemanja poti in dodajanju fotografij k poti. Fragment RouteRecordingFragment ima samo element RouteViewFragmentContainer. Ta je namenjen vgradnji RouteViewFragmenta, ki je sestavljen iz fragmenta Google Maps ter dveh prikazovalnikov teksta (angl. *TextView*), ki sta namenjena prikazovanju dolžine in trajanja poti. V spodnjem delu opravilne vrstice je na začetku na voljo samo akcija pričetka snemanja. Začetni zaslon zavihka Route je prikazan na sliki 3.7. Ko uporabnik prične s snemanjem poti, se od upravljavca lokacijske storitve zahteva osveževanje lokacije naprave. Sistem Android tako pošilja sprejemniku namer trenutne lokacije mobilne naprave. Opravilna vrstica z akcijami se, kot je prikazano na sliki 3.8, posodobi, tako da prikazuje akciji za konec snemanja

```
private void takePicture(){

    //  dispatch take picture intent
    Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    startActivityForResult(takePictureIntent, REQUEST_CODE_TAKE_PICTURE);
}
```

Slika 3.10: Implementacija dodajanja fotografije.

in za dodajanje fotografij.

Ko aplikacija dobi prvo lokacijo telefona, na zemljevid nariše znak za začetek poti in premakne sredino zemljevida na to lokacijo. Ob uporabnikovem pritisku na ikono za dodajanje fotografije aplikacija kreira namero in od sistema Android, zahteva naj zažene ustrezno aplikacijo, ki zna postreči zahtevo v nameri. Ta je v našem primeru fotografiranje. Kreiranje namere in pošiljanje je prikazano na sliki 3.10.

Aplikacija, ki posname fotografijo, le-to vrne aplikaciji RouteTracker, ki jo shrani na datotečni sistem, poveže s trenutno lokacijo telefona in prikaže ikono fotografije na poti. Ikona je vidna na sliki 3.3.

Ko se ob pritisku na akcijo stop snemanje zaključi, se prikaže dialog za izbiro imena poti. Privzeto ime je trenutni čas in datum. Uporabnik lahko to ime spremeni. Ob pritisku na gumb OK se pot zapiše v datoteko xml, katere oblika je prikazana na sliki 3.11, in shrani na datotečni sistem. Dialog je prikazan na sliki 3.9.

3.3.2 Zavihek History

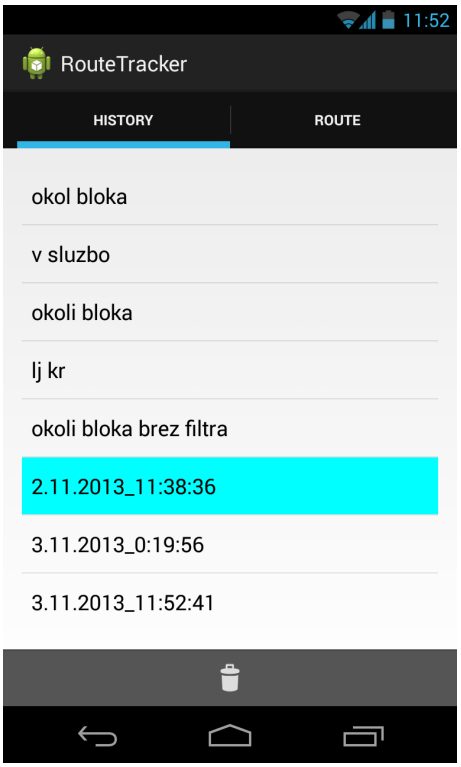
Zavihek History je namenjen prikazovanju na datotečnem sistemu shranjenih poti. Fragment, ki prikazuje zavihek History, ima samo element ListView. Ta je namenjen prikazovanju seznama shranjenih poti. Seznam je viden na sliki 3.6. Ob izbiri zavihka History se preberejo na datotečnem sistemu shranjene poti in prikažejo v seznamu. Ob izbiri poti iz seznama se prikaže aktivnost RouteViewActivity, ki vsebuje fragment RouteViewFragment. Aktivnost prikaže pot z začetkom, koncem, fotografijami, časom in

```
<route name="okoli bloka" distance="205" duration="274">
 <point>
   <lat>46.0417505</lat>
   <long>14.4809939</long>
   <acc>42.24300003051758</acc>
  </point>
 <point>
   <lat>46.04182522</lat>
   <long>14.48088326</long>
   <acc>17.0</acc>
  </point>
 <point>
   <lat>46.04185892</lat>
   <long>14.48091239</long>
   <acc>11.0</acc>
  </point>
```

Slika 3.11: Oblika datoteke xml, ki predstavlja pot.

dolžino. Ob pritisku na ikono, ki označuje fotografijo, se sistemu Android prek namere ukaže, naj zažene prikazovalnik fotografij in prikaže fotografijo. Primer prikaza poti je predstavljen na sliki 3.3, primer prikaza med potjo posnete fotografije pa na sliki 3.13.

Shranjeno pot je mogoče tudi zbrisati iz datotečnega sistema. Uporabnik z dolgim klikom na vnos v seznamu izbere pot za brisanje. Kot je prikazano na sliki 3.9, se vnos obarva, v opravilni vrstici pa se prikaže akcija za brisanje. Ob izbiri te akcije se pot zbriše iz datotečnega sistema. Nato se vsebina datotečnega sistem prebere in še enkrat prikaže v seznamu, ki tokrat ne vsebuje prej zbrisane poti.



Slika 3.12: Brisanje shranjene poti.

Slika 3.13: Prikaz posnete fotografije.

Poglavje 4

Sklepne ugotovitve

Pri izdelavi diplomskega dela sem spoznal delovanje mobilnega operacijskega sistema Android in razvoj aplikacij zanj. Najprej sem se poglobil v teoretično delovanje sistema in postavil razvojno okolje. Pri samem razvoju zaradi izredno dobre dokumentacije razvojnih orodjih in postopkov nisem imel večjih težav. Najprej sem definiral zahteve za aplikacije in na podlagi teh zahtev izbral komponente in tehnologije, ki so implementirane v aplikaciji. Po uspešnem preizkusu vseh komponent in tehnologij sem se lotil pisanja programske kode in ostalih stvari, povezanih z razvojem. Rezultat je delujoča aplikacija RouteTracker, katere glavna funkcionalnost je sledenje mobilni napravi in risanje sledi na zemljevidu. Posebej uporabna se mi zdi funkcionalnost, ki omogoča fotografiranje ter shranjevanje fotografij na posneto sled gibanja. Pri preizkusu aplikacije sem ugotovil, da so lokacije, pridobljene prek mobilnih omrežij ali brezžičnih dostopnih točk, zaradi slabe točnosti za ta tip aplikacije neuporabne.

Aplikacijo bi bilo mogoče nadgraditi, tako da bi bilo mogoče poleg fotografij zajemati tudi video posnetke, zvok ali napisati beležko. Vse te priponke bi se, tako kot sedaj fotografije, prikazale na sledi gibanja. Posneto sled s priponkami bi lahko zapakirali in prenesli na prenosni ali osebni računalnik, kjer bi si jo v posebnem programu lahko ogledali. Lahko bi jo naložili v Googlovo oblačno storitev Google Drive ter jo tako delili z izbranimi ose-

bami. Aplikacija je dobro izhodišče in temelj za razvoj kompleksnejših in zmogljivejših aplikacij.

Literatura

[1] (2013) Android, the world's most popular mobile platform. Dostopno na:
http://developer.android.com/about/index.html

[2] (2013) BlackBerry OS. Dostopno na:
http://en.wikipedia.org/wiki/BlackBerry_OS#1.0

[3] (2013) Comparison of smartphones 2013. Dostopno na:
http://en.wikipedia.org/wiki/Comparison_of_smartphones#2013

[4] (2013) Delež pametnih telefonov na trgu mobilnih telefonov. Dostopno na:
http://www.ris.org/db/27/12565/Raziskave/V_EU5_je_94_vec_pametnih_telefonov_kot_pred_letom/

[5] (2013) Eclipse (software). Dostopno na:
http://en.wikipedia.org/wiki/Eclipse_%28software%29#History

[6] (2013) ENIAC. Dostopno na:
http://en.wikipedia.org/wiki/ENIAC

[7] (2013) Global Positioning System. Dostopno na:
http://en.wikipedia.org/wiki/Global_Positioning_System

[8] (2013) Google Nexus. Dostopno na:
http://en.wikipedia.org/wiki/Google_Nexus

[9] (2012) Google Nexus 4. Dostopno na:
http://www.gsmarena.com/lg_nexus_4_e960-5048.php

[10] (2012) Google Play track. Dostopno na:
https://play.google.com/store/search?q=track&hl=sl

[11] (2013) History of mobile phones. Dostopno na:
http://en.wikipedia.org/wiki/History_of_mobile_phones

[12] (2013) LTE. Dostopno na:
http://en.wikipedia.org/wiki/LTE_(telecommunication)

[13] (2013) Mobile operating system. Dostopno na:
http://en.wikipedia.org/wiki/Mobile_operating_system

[14] (2013) Most popular Android mobile applications. Dostopna na:
https://play.google.com/store/apps/collection/topselling_free

[15] (2013) Programski jezik Java. Dostopno na:
http://sl.wikipedia.org/wiki/Programski_jezik_java

[16] (2013) Smartphone. Dostopno na:
http://en.wikipedia.org/wiki/Smartphone

[17] (2013) Symbian officially dead. Dostopno na:
http://www.gsmarena.com/the_808_pureview_is_the_last_symbian_by_nokia_an_end_of_an_era-news-5400.php

[18] (2013) What is smartphone. Dostopno na:
http://cellphones.about.com/od/smartphonebasics/a/what_is_smart.htm