Android aplikacija

Legović, Boris

Undergraduate thesis / Završni rad

2015

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:170510

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-08-07**

Repository / Repozitorij:

Digital Repository Juraj Dobrila University of Pula

Sveučilište Jurja Dobrile u Puli

Fakultet ekonomije i turizma

Dr. Mijo Mirković

BORIS LEGOVIĆ

Android aplikacija

Where2Go

Završni rad

Pula, 2015.

Sveučilište Jurja Dobrile u Puli

Fakultet ekonomije i turizma

Dr. Mijo Mirković

BORIS LEGOVIĆ

Android aplikacija

Where2Go

Završni rad

JMBAG: 0303022489, redovni student

Studij, smjer: Informatika

Kolegij: Projektiranje informacijskih sustava

Mentor: Prof. dr. sc. Vanja Bevanda

Pula, 2015.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Boris Legović, kandidat za prvostupnika Informatike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student:

U Puli, __. __. 2015. _____

Sadržaj

Uvod 6

1. Mobilno poslovanje 7

2. Poslovni modeli mobilnog poslovanja 8

3. Mobilni Operacijski sustavi 10

3.1. Operacijski sustav Android 10

3.2 Arhitektura sustava Android 10

3.3 Virtualni stroj 12

3.4. Koncept Android aplikacija 13

3.5 Osnovne komponente aplikacije 14

4. Razvoj aplikacija za Android 20

4.1 Korisničko sučelje 20

4.2 Izbornici 21

5.3 Dijalozi 22

4.4 Događaji unutar korisničkog sučelja 23

4.5 Podatkovni resursi 23

4.6 Upotreba resursa u kodu 24

4.7 Upotreba resursa u drugim podatkovnim resursima 24

4.8 Upotreba sistematskih resursa 24

5. Prikaz razvoja aplikacije 25

5.1 Aplikacija Where2Go 25

5.2 Where2go kao socijalni model 26

5.3 Aplikacija Where2Go u poslovnom modelu 26

5.4 Početna aktivnost 27

5.5 Sekundarna aktivnost – detaljni opis 27

5.6 Aktivnost prijava 28

5.7 Prijavljena aktivnost 28

5.8 Registracijska aktivnost 30

5.9 Sigurnosna Aktivnost 30

5.10 Where2Go aktivnost.....30

6. Zaključak 32

Literatura:.....33

Popis slika.....34

Popis tablica:.....34

Uvod

Mobilna tehnologija svoje početke gradi 1973., koja se smatra i početkom mobilne ere (izvor: mob.hr). Prvi mobilni uređaji imali su jednu funckiju – obavljanje poziva. S napretkom tehnologije mobilni uređaji postaju jeftini i dostupni širokoj masi, dok se mogućnosti koje uređaji pružaju, eksponencijalno rastu, a time i njihova popularnost. Vrhunac popularnosti doživljava izlaskom Apple-ovog pametnog telefona 2007. godine.[1] Nakon toga, velike kompanije poput Microsofta, Google-a izbacuju svoje inačice operativnih sustava, kao i mnoge manje poznate tvrtke. Uskoro se na tržištu pojavio niz softvarea za izradu aplikacija. Danas Google-ov operativni sustav Android obuhvaća 82.8% tržišta pametnih telefona.[2]

Osnovna hipoteza ovoga rada je da mobilne aplikacije povećavaju razinu podjele informacija, poboljšavaju poslovanje i kvalitetu življenja. Cilj rada je razviti aplikaciju za prikaz informacija o interaktivnim objektima. U ovome radu prikazat će se proces nastajanja aplikacije za Android, od poslovnog plana do praktične izrade. Prvi dio završnog rada govorit će o mobilnom poslovanju, te prikazati nekoliko poslovnih modela. Drugi dio opisivat će Android arhitekturu i njene komponente dok se u trećem dijelu obrađuje sama aplikacija i poslovni model koji će se primjeniti.

1 Povijest mobilne tehnologije, ŠTO SE DOGAĐALO 40 GODINA, dostupno na: http://mob.hr/povijest-mobilne-telefonije-sto-se-dogadalo-u-40-godina/ (01.09.2015)

2 Udio operativnih sustava na tržištu, SMARTPHONE OS MARKETSHARE 2015, dostupno na: http://www.idc.com/prodserv/smartphone-os-market-share.jsp (02.09.2015)

1. Mobilno poslovanje

Mobilno se poslovanje može definirati kao korištenje mobilnih tehnologija u razmjenu dobara, usluga, informacija i znanja. Mobilno poslovanje je izvršavanje transakcijaobavljanih pomoću pokretne opreme putem mobilnih mreža, koje mogu biti bežićne ili javne mreže[3] . Mobilno poslovanje predstavlja proces primjene mobilnih tehnologija u poslovne svrhe, za pružanje usluga, trgovine i vršenje plaćanja. Ono zapravo predstavlja elektronsko poslovanje koje se odvija u bežičnom okruženju uz pomoć bežičnih uređaja koji su prešli u svakodnevicu. Cilj mobilnog poslovanja je što bolja poslovna efikasnost koja se odnosi na niže troškove poslovanja i bolju konkurentsku poziciju na tržištu. Prva generacija mobilnih usluga donosi komunikacijske čvorove koji su omogućavali prebacivanje iz jednog geografskog područja u drugo bez mjenjanja fekvence i bili su povezani centralnom razmjenom. Druga generacija donosi GSM - *Global System for Mobile Communications* koja omogućava slanje poruka. Razvitak treće generacije mobilnih usluga označio je mogućnost zz bolju propusnost raspoloživih podataka i uveden je novi spektar usluga. Te se usluge kreću u tri pravca, a odnose se na:

- usluge mreže (network services) – obuhvaćaju usluge telefonije kroz održavanje mreže,
- dodatne usluge vezane za mrežu – SMS i MMS poruke, glasovna pošta, audio – video konferencije i slično,
- usluge vezane za sudjelovanje treće strane, npr. proces vršenja mobilnih financija uz učešće poduzeća i banaka.
- Mobilno poslovanje razlikuje sljedeće ovlasti: mobilna poslovna komunikacija, mobilna trgovina i mobilna plaćanja. Prva ovlast pokriva odnos poduzeća komunikaciju između zaposlenih unutar samog poduzeća, dok druge dvije ovlasti reprezentiraju odnose između poduzeća i potrošača.

Prednost mobilnog poslovanja je u tome što se ono može odvijati bilo gdje i bilo kada uz primjenu Internet tehnologije.[4]

3 Panian, Ž., **Elektroničko poslovanje druge generacije**, Zagreb, 2013., str 125.

4 Generacije mobilnih usluga, MOBILNI UREĐAJI, dostupno na: http://www.informatika.buzdo.com/pojmovi/mobile-1.htm (03.09.2015)

2. Poslovni modeli mobilnog poslovanja

Pri planiranju izrade mobilne aplikacije, potrebno je razviti plan zarade. Načini zarade planiraju se u samim počecima razvoja projekta. Proces nastaje u trenutku nastanka ideje, a nekada i paralelno. Kao i svaki drugi proces, politika cijena ima neke svoje zakonitosti koje se moraju poštovati. Prvi korak kreće od istraživanja tržišta. Potrebno je definirati zakone koji vladaju tržištem i na temelju analize planirati daljnji razvoj. Za planiranu prodaju postoji formula kojom se izračunava zarada od aplikacije. Ona glasi : broj stanovnika s obzirom na lokaciju, te uzima postotak korisnika koji koriste namijenjeni operativni sustav * cijena aplikacije – 30% (provizija za playstore).

Jedan od jednostavnijih modela je *Model kupnje* (eng. *Paid model*) u kojemu sami određujemo cijenu ovisno o vlastitoj procjeni aplikacije. Ako aplikacija ima bežičnu ulogu povremenog rješavanja nekog problema, cijena se kreće u visini jedne kave. Ako aplikacija rješava neki veći problem, cijena može biti nešto viša, no samim time rezultirati će smanjenim brojem preuzimanja. Kod definiranja cijene, potrebno je evaluirati vrijednost, budući da nakon postavljanja cijene, ona samo može ići dolje. Osim u slučajevima nadogradnje, cijena se može povećavati.

Sve je popularniji model besplatnih aplikacija je *Besplatni model (eng. Freemium model*) kojima se naplaćuje otključavanje pune funkcionalnosti. Ograničenja mogu biti vremenskog tipa gdje se aplikacija ograničava na probni period ili se neka mogućnost nudi nekoliko puta dnevno, mjesečno itd. Danas, pogledom na "google play" ili "app store" moguće je vidjeti da prevladava ovakav tip aplikacija.

Model koji obilježavaju reklame ili plaćeni oglasi profit ostvaruju na reklamama koje se pojavljuju tijekom korištenja. Da bi se ostvarila zarada na ovakvom modelu potrebno je imati veliki broj preuzetih aplikacija jer zarada po prikazu iznosi nekoliko lipa. Često se može kombinirati s Freemium modelom, te se po naplati reklame maknu.[5]

Početkom 2000 – ih godina dolazi do popularizacije društvenih mreža. Danas je iz toga proizašao socijalni model. Društvene mreže postale su vrhunac interesa više dobnih generacija te njihove aplikacije zauzimaju sam vrh aplikacijske trgovine. ‘’Nema tih argumenata kojima bi negirali jačanje utjecaja socijalnih medija, ne samo na poslovanje već i na društvo općenito. S prosječno 1.400 minuta – ili gotovo 24 sata – koje dnevno provedu na Facebooku svakog mjeseca, oko 550 milijuna američkih korisnika zapravo na socijalne medije troši oko 30% svojeg slobodnog

5 Naplatni modeli, MONETIZACIJA MOBILNIH APLIKACIJA: KAKO ODABRATI NAPLATNI MODEL?, dotupno na: http://www.pokreniposao.hr/monetizacija-mobilnih-aplikacija-kako-odabrati-naplatni-model/ (04.09.2015)

vremena . No, nisu samo frapantni pokazatelji kvantitativnog rasta socijalnih medija i mreža ono što treba izazvati pozornost. Još više od toga, upozoravajuća je činjenica to što se ne mijenja samo tehnologija. Mijenjaju se ljudi pa onda i društvo u cjelini koje se sve više okreće virtualnom svijetu. Možda je bizarna, ali je i vrlo upečatljiva činjenica da se jedan od šest američkih parova koje se vjenčao u posljednje dvije godine upoznao *on-line*. Diljem svijeta, socijalni mediji i novi mobilni uređaji postaju ekstenzijom osobnih odnosa na način koji čini teškim razdvajanje tehnologije od privatnog života i neformalnih odnosa u društvenim mrežama. U tom svijetlu treba gledati i potrebe današnjeg zaposlenog stanovništva. Oni i na poslu očekuju nesmetanu i trenutnu komunikaciju i pristup aplikacijama koje im u privatnom životu pomažu da budu učinkoviti. Za njih su osobni mobilni uređaji i društvene mreže idealni alati za izgradnju poslovnih odnosa i dobro obavljenog posla".[6]

6 Panian, Ž., **Elektroničko poslovanje druge generacije**, Zagreb, 2013., str. 355.

3. Mobilni Operacijski sustavi

Mobilni operacijski sustavi postoje od nastanka prvih mobitela. No, prvi pametni sustavi nastaju tek 2007. godine kada Apple predstavlja svoj Iphone i Ios operacijski sustav. Nakon niza inovacija koje je donio, ostale velike kompanije izbacuju svoje inačice pametnih operacijskih sustava. Oni su i danas aktualni, pa tako na tržištu vladaju Apple-ov Ios, Google-ov Android i Microsoft-ov Windows phone. Android je najpopularniji sustav sa udjelom od 82,8% , slijedi Ios sa 13,9% , Windows phone 2,6% tržišta, te ostali manje popularni sustavi. [7]

3.1. Operacijski sustav Android

Mala kompanija, čija je osnovna djelatnost bila razvoj programske potpore za pametne mobilne uređaje pokrenula je operacijski sustav Android. Kasnije ju kupuje Google, a 2007. godine inicira osnivanje konzorcija OHA (*Open Handset Alliance*) s ciljem stvaranja otvorenih standarda za mobilne uređaje, promociju inovacija i prilagodba uređaja korisniku s poboljšanom izvedbom i pristupačnom cijenom. Odmah zatim, udruzi su pristupile 34 firme različitih djelatnosti iz mobilne industrije poput proizvođača mobilnih telefona, programera aplikacija, mobilnih operatera i sličnih. Udruga je ubrzo brojila 80 članova u kojima se trenutno nalaze T – Mobile, HTC, Intel, Motorola, Qualcomm i drugi. OHA je odmah po osnivanju udruženja predstavila mobilnu platformu Android.

Operacijski sustav Android baziran je na programskom jeziku Java te ima ograničene resurse u pogledu procesne moći i memorijskih kapaciteta u odnosu na osobna računala. Jezgra i arhitektura sustava prilagođeni su za pokretanje i rad u ograničenim uvjetima kakvo je mobilno okruženje.

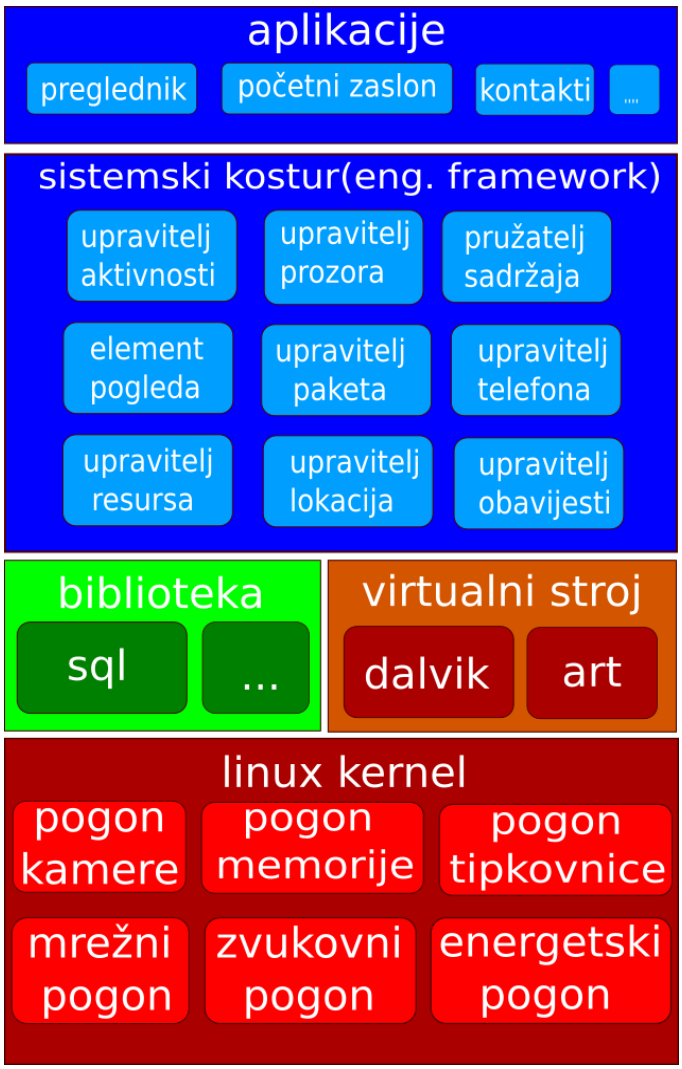
Što se tiče pristupa mogućnostima uređaja, Android ne razlikuje jezgrene aplikacije i aplikacije neovisnih proizvođača. Stoga, i jedni i drugi mogu u potpunosti iskoristiti sve resurse i mogućnosti koje mu pruža uređaj na kojemu je instaliran. Android sadrži 2D i 3D grafičke biblioteke s podrškom za hardversku akceleraciju, SQLite sustav za upravljanje i pohranu relacijskih baza podataka, omogućuje izvođenje pozadinskih procesa te pruža bogato korisničko sučelje.

3.2 Arhitektura sustava Android

Sustav Android bazira se na Linux jezgrama (eng. *Kernel*) verzije 2.6 i višim. One se koriste kao sloj apstrakcije hardvera (HAL, eng. *Hardware Abstraction Layer*). Dokazana pogonska podrška (eng. *Driver model*), mogućnost upravljanja memorijom i procesima, sigurnosni model, mrežni sustav te dokazana robusnost, konstantni razvoj i unapređivanje sustava razlog su za korištenje jezgre operacijskog sustava Linux.

7 Udio mobilnih operativnih sustava na tržištu, SMARTPHONE OS MARKET SHARE 2015, dotupno na: http://www.idc.com/prodserv/smartphone-os-market-share.jsp (04.09.2015)

Kao što je prikazano na slici (Slika 1), komponente operacijskog sustava Android dijele se u pet cjelina.



Slika 1 Arhitektura operacijskog sustava Android, izvor: izradio autor

Sloj aplikacija čini skup osnovnih aplikacija: e-mail klijent, kalendar, Internet preglednik i drugi. Sve aplikacije napisane su u programskom jeziku Java. Arhitektura aplikacije je napravljena na način da pojednostavi ponovnu upotrebu komponenti. U aplikacijski okvir spadaju:

- skup elemenata prikaza (eng. *Views*) koji se koriste pri izradi aplikacije kao što su liste, gumbi, polja za unos teksta itd.,
- pružatelji sadržaja (eng. *Content Providers*) koji aplikacijama omogućuje pristup podacima drugih aplikacija,

- upravitelj resursa (eng. *Resource Manager*) koji omogućuje pristup ostalim resursima kao što su slike ili dokumenti koji određuju izgled,
- upravitelj obavijesti (eng. *Notification Manager*) koji pruža aplikacijama mogućnost prikaza obavijesti u traci stanja,
- upravitelj aktivnosti (eng. *Activity Manager*) koji upravlja životnim ciklusom aplikacije i brine se o navigacijskom stogu.

Sloj biblioteka sadrži skup C/C++ koje koriste razne komponente operacijskog sustava poput System C biblioteka, 3D biblioteke, SQLite i ostale. U *Android Runtime* sloj spadaju osnovne biblioteke. One omogućuju većinu funkcionalnosti osnovnih biblioteka programskog jezika Java. Svaka aplikacija odvija se u vlastitom procesu. Također, svaka aplikacija ima vlastitu instancu Dalvik virtualnog stroja koji se oslanja na Linux jezgru.

3.3 Virtualni stroj

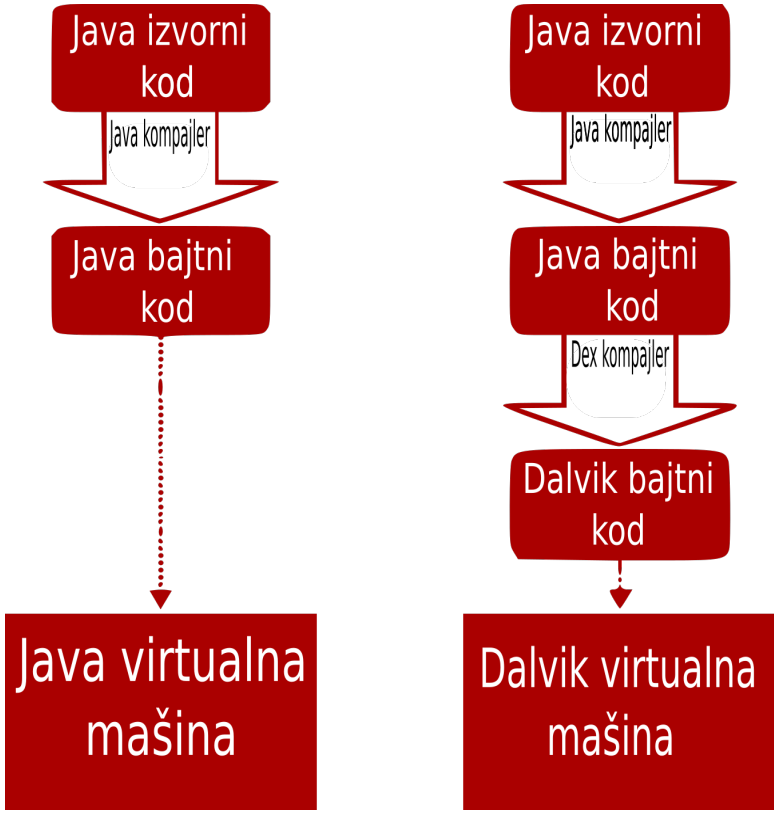
Google nije koristio standardni Java 2 Micro Edition (J2ME) kao mehanizam za pokretanje Javinih aplikacija na pokretnim uređajima, već je razvio vlastiti virtualni stroj za Android. Dalvik virtualni stroj (DVM) je najvjerojatnije razvijen kako bi se zaobišla problematika s dozvolama za korištenje Sun-ovog J2ME.

Osnovna razlika je u tome što su Sun Java virtualni strojevi stožno bazirani virtualni strojevi, a DVM je registarski baziran virtualni stroj. Međukod (eng. *bytecode*) Dalvik virtualnog stroja transformira se pomoću alata dex (koji je sastavni dio Android SDK-a) iz Javinih kasnih datoteka (eng. *Java class file*) prevedenih Javinim prevoditeljem u novu klasu *.dex (eng. *Dalvik Executable*) formata.

Međukod kojeg izvršava DVM nije Javin međukod, već upravo spomenuti .dex oblik. Transformacija formata omogućava bolju prilagodbu za rad na manjim procesorima boljim iskorištavanjem raspoložive memorije i procesorske snage. Rezultat svega je mogućnost višestrukog instanciranja samog virtualnog stroja. To znači da se svaka Androidova aplikacija pokreće kao zasebni proces s vlastitom instancom virtualnog stroja Dalvik.

Danas je Dalvika zamjenio Android Runtime (ART). Art također pokreće međukod, stoga su sustavi kompatibilni. On predstavlja novi oblik kompilacije koda poznat kao “ahead –of-time (AOT)” koji poboljšava performance aplikacije. Također, vrijeme instalacije je kraće nego kod Dalvika. Nekompatibilnosti se pojavljuju kod instalacijskog procesa u kojem ART kompilira

aplikacije koristeći on-device dex2oat alat. Alat kao ulaz prima dex datoteke i generira egzekutivne programe za ciljani uređaj. Alat može kompilirati sve validne dex podatke. No, neki post-procesni alati znaju generirati nevaljane datoteke koje Art neće moći kompilirati, dok Dalvik neće imati problema s istim.



Slika 2 Razlika između Sun Java i Dalvik virtualnog stroja, izvor: izradio autor

Struktura .dex datoteke sastoji se od: jednostavnog zaglavlja, identifikatora nizova, tipova, prototipova, polja i metoda, definicija klasa i podataka. Time se osigurava da se konstante koje se ponavljaju u klasnim datotekama pojavljuju samo jednom u datoteci .dex kako bi se sačuvala memorija.

3.4. Koncept Android aplikacija

Sve Androidove aplikacije pisane su u programskom jeziku Java. Aplikacijom se smatra kompletni kod paketa upakiran pomoću AAPT (eng. *Android Asset Packaging Tool*) alata koji kao rezultat stvara .apk datoteku. Aplikacije se distribuiraju na mobilni uređaj upravo u ovom formatu.

Većina stvorenih aplikacija pripada jednoj od sljedećih kategorija:

- aktivnost koja se izvršava u prvom planu (eng. *Foreground activity*) – aplikacije bez korisne

pozadinske aktivnosti. Kada su izvan fokusa aplikacije ove vrste se uglavnom privremeno zaustavljaju,

- pozadinska usluga (eng. *Background service*) – aplikacije ograničene interakcije s korisnikom koje se uglavnom izvršavaju u pozadini,
- aktivnost s prekidima – aplikacije koje podrazumijevaju određeni stupanj interakcije s korisnikom, ali se uglavnom odvijaju u pozadini.

Za razliku od aplikacija na drugim sustavima, Androidove aplikacije nemaju samo jednu pokretnu točku (konkretnije – ne postoji samo jedna *main()* funkcija). Razlog je zapravo karakteristika sustava koja zastupa ideju međusobne interakcije dvije ili više različitih aplikacija. Da bi to bilo ostvarivo, sustav mora biti u mogućnosti pokrenuti proces pojedine aplikacije i kada se zatraži rad samo nekog njenog određenog dijela te instancirati Javin objekt za dotični dio.

3.5 Osnovne komponente aplikacije

Aktivnost (eng. *Activity*) jedna je od četiri komponenata koje čine aplikaciju. Ona predstavlja komponentu aplikacije koja se uglavnom može poistovjetiti s jednim prozorom aplikacije u kojemu je korisnik u mogućnosti izvršiti određenu radnju. Aplikacija može sadržavati jednu ili više definiranih aktivnosti. Jedna od njih je uvijek definirana kao primarna aktivnost. Dizajn i namjena aplikacije utjecat će na to koliko će aplikacija imati komponenti aktivnosti ili bilo kojih drugih osnovnih komponenti te kako će one surađivati.

Prijelaz između aktivnosti odvija se tako što aktualna aktivnost invocira novu. S obzirom da više aktivnosti tvori jedno kompaktno korisničko sučelje, treba imati na umu da su one međusobno nezavisne. Pri izradi, svakoj je aktivnosti dodijeljen njezin vlastiti prozor koji je inicijalno postavljen da prekrije cijeli ekran. To se naravno može i promijeniti. Svaka aktivnost može koristiti dodatne elemente prikaza. To su primjerice *pop-up* prozori koji zahtijevaju korisnikovu interakciju ili prozor za prikaz detaljnih informacija o trenutnom odabranom elementu.

Vizualni sadržaj prozora ima hijerarhijski organiziran sadržaj elemenata prikaza izvedenih iz osnovne klase *View*. Svaki element pripada određenom prostoru pravokutnog oblika unutar prozora. Pojam 'hijerarhijski' odnosi se na elemente prikaza pa tako element prikaza 'roditelj' organizira i sadrži izgled i raspored elemenata prikaza 'djece'. Postoji skup gotovih elemenata prikaza sadržanih u Androidu kao što su gumbi, polja za unos teksta, izbornici, *check-boxovi* i drugi.

Namjera (eng. *Intent*) predstavlja namjeru za obavljanje određene radnje, odnosno omogućuje prijelaz između zaslona aplikacija. U radu aplikacija i sustava pojavljuje se mnogo obavijesti poput: obavijesti da ponestaje napona u bateriji, da se promijenila vremenska zona, da su aplikacijama koje ih trebaju dostupni podaci prikupljeni s Interneta itd. Stoga treba spomenuti i *Broadcast receivere* koji primaju obavijesti o višeodredišnom odašiljanju i reagiraju na njih. Aplikacija može imati bilo koji broj *Broadcast Receivera* koji mogu odgovarati na sve informacije koje se smatraju važnima. Ove komponente nemaju grafičko sučelje, ali mogu pokrenuti određenu aktivnost kao odgovor na primljenu informaciju ili umjesto toga mogu koristiti upravitelja obavijesti. Takve obavijesti mogu se prikazati na različite načine, npr. kao treperenje pozadinskog svjetla, vibracija pokretnog uređaja, reprodukcija određenog zvuka.

Usluga (eng. *Service*) predstavlja proces bez vidljie korisničke interakcije. Uglavnom se izvršavaju u pozadini kroz nedefinirani period vremena, služe za obnavljanje podatkovnih resursa, vidljivih aktivnosti i signalizacijskih obavijesti. Svaka usluga je potklasa osnovne klase *Service*.

Pružatelj sadržaja (eng. *Content Provider*) omogućava uzajamno korištenje podataka između različitih aplikacija i njihovih procesa. Ti se podaci mogu pohraniti u datotečni sustav kao datoteka ili u obliku SQLite baze podataka. Pružatelj sadržaja nasljeđuje klasu *ContentProvider*. Ona sadrži skup metoda koje omogućavaju aplikacijama preuzimanje ili pohranu tipova podataka s kojima rade. Aplikacije ne pozivaju te metode izravno, već pomoću objekta ContentResolver. Takav objekt može komunicirati sa svakim *Content Providerom* na način da on upravlja komunikacijom.

Operacijski sustav se brine o tome je li proces pokrenut ili ga treba pokrenuti na svaki zahtjev koji trenutno neaktivna aplikacija treba obraditi. Također, operacijski sustav sam stvara instance koje su mu potrebne, a dio su aplikacije koja treba odraditi zahtjev.

Svaka aplikacija sadrži i opisnu datoteku *AndroidManifest.xml*. Datoteka se nalazi u korijenskom direktoriju paketa i sadrži bitne informacije o aplikaciji te informacije koje sustav mora imati prije pokretanja bilo kojeg dijela aplikacije. Između ostalog, manifest sadrži sljedeće informacije:

- naziv paketa koji služi kao jedinstveni identifikator aplikacije,
- opis komponenti – aktivnosti, usluge, pružatelji sadržaja, filtere namjere, *Broadcast Receivers* i ostali,
- odredbe o tome koji će procesi sadržavati programske komponente,

- deklaracija dozvola za pristup aplikacijskim komponentama od strane drugih aplikacija,
- minimalna razina Android API-ja koju zahtijeva aplikacija,
- popis biblioteka s kojima aplikacija treba biti povezana,
- popis instrumentacijskih klasa koje osiguravaju oblikovanje i ostale informacije dok je aplikacija aktivna – ove deklaracije prisutne su u *AndroidManifest.xml* datoteci prilikom razvoja i testiranja te se izbacuju prije njenog objavljivanja.

U nastavku prikazani su osnovni elementi datoteke AndroidManifest.xml čije je značenje sljedeće:

- <*uses-sdk*> - definira razine Android API-ja na kojima se može izvoditi aplikacija,
- <*uses-permission*> - dio sigurnosnog modela koji sadrži sigurnosne dozvole dodijeljene od strane korisnika, a odnose se na funkcije aplikacije. Dozvole se definiraju prilikom instaliranja aplikacije,
- <*uses-library*> - specifira biblioteku s kojom aplikacija mora biti povezana,
- <*service*> - definira usluge. Sadržan u <application> bloku, a može sadržavati <*intent-filter*> i <meta-data>,
- <*receiver*> - definira *Broadcast Receiver* koji omogućuje prihvat namjera i u slučaju kada aplikacija i njene komponente nisu pokrenute,
- <*provider*> - definira pružatelja sadržaja,
- <*permission-tree*> - određuje stablo dozvola,
- <*permission*> - deklarira sigurnosne dozvole nad komponentama kojima pristupaju druge aplikacije,
- <*meta-data*> - par ime-vrijednost dodatnih metapodataka koji mogu biti isporučeni roditeljskoj komponenti,
- <*manifest*> - definira komponente, sigurnosne postavke i testne klase koje čine aplikaciju,
- <*intent-filter*> - odreĎuje vrstu namjere na koju reagiraju aktivnost, usluga ili *Broadcast Receiver,*

- <*instrumentation*> - deklarira koje su instrumentacijske komponente raspoložive za testiranje funkcionalnosti paketa,
- <grant-uri-permission> - deklarira koje su instrumentacijske komponente raspoložive za testiranje funkcionalnosti paketa,
- <*data*> - određuje vrstu podatka Intent filtera. To mogu biti opcionalni, ali međusobno zavisni atributi MIME ili URI,
- <category> vrsta kategorije Intent filtera,
- <*application*> - korijenski element s opisom dubine stabla kojeg čine komponente paketa. Također, može sadržavati globalne i/ili unaprijed zadane aplikacijske atribute kao što su tema, ikona, dozvole itd. Manifest datoteka može sadržavati samo jedan element ove vrste,
- <*activity-alias*> - zamjenska aktivnost. Ciljanu aktivnost predstavlja kao nezavisni entitet s vlastitim skupom filtera namjera koji umjesto definiranja namjera za izravno pokretanje ciljane aktivnosti određuju namjere koji ju pokreću preko zamjenske aktivnoti,
- <*activity*> - obavezni deklarirajući element za svaku aktivnosti aplikacije. Podržava podelement <intent-filter> kojim se specifiraiju pojedine namjere aktivnosti. Pokretanje aktivnosti koja nije definirana u AndroidManifest.xml datoteci rezultira dojavom greške pri izvođenju,
- <*action*> opisuje akciju dodijeljenu <intent-filter> elementu. Ukoliko se ne definira Intent objekti ne prolaze kroz filtere.

Dozvola predstavlja sigurnosni pristup memoriji odabranog procesa. Definiraju se zbog hardverski ograničenog pristupa memoriji jednog procesa od strane drugog te zato što je svakoj aplikaciji dodijeljen korisnički ID. Neke od najčešće korištenih dozvola su:

- INTERNET – pristup Internetu,
- READ_CONTACT – dozvola za čitanje podataka kontakata,
- WRITE_CONTACTS – dozvola za pisanje podataka kontakata,
- RECEIVE_SMS – nadzor pristigle SMS poruke.

Postoje tri stanja u kojima se aktivnost može naći:

- aktivno stanje ili stanje izvođenja – kada je prikazana na ekranu (to je stanje kada korisnik koristi njezine mogućnosti),
- pauzirano stanje – još uvijek je vidljiva korisniku, no trenutno ga ne koristi. To je slučaj kada je druga aktivnost koja je transparentna ili ne pokriva cijeli ekran „iznad" njega. U slučaju da operacijski sustav ima premalo slobodne memorije onda može zaustaviti takvu aktivnost,
- zaustavljeno stanje – druga aktivnost ju je potpuno prekrila. Još uvijek su sačuvane sve informacije o stanju. Ovakve aktivnosti se često zaustavljaju kada sustavu treba memorije.

Prelazak iz jednog stanja u drugo događa se pozivom jedne od ovih metoda:

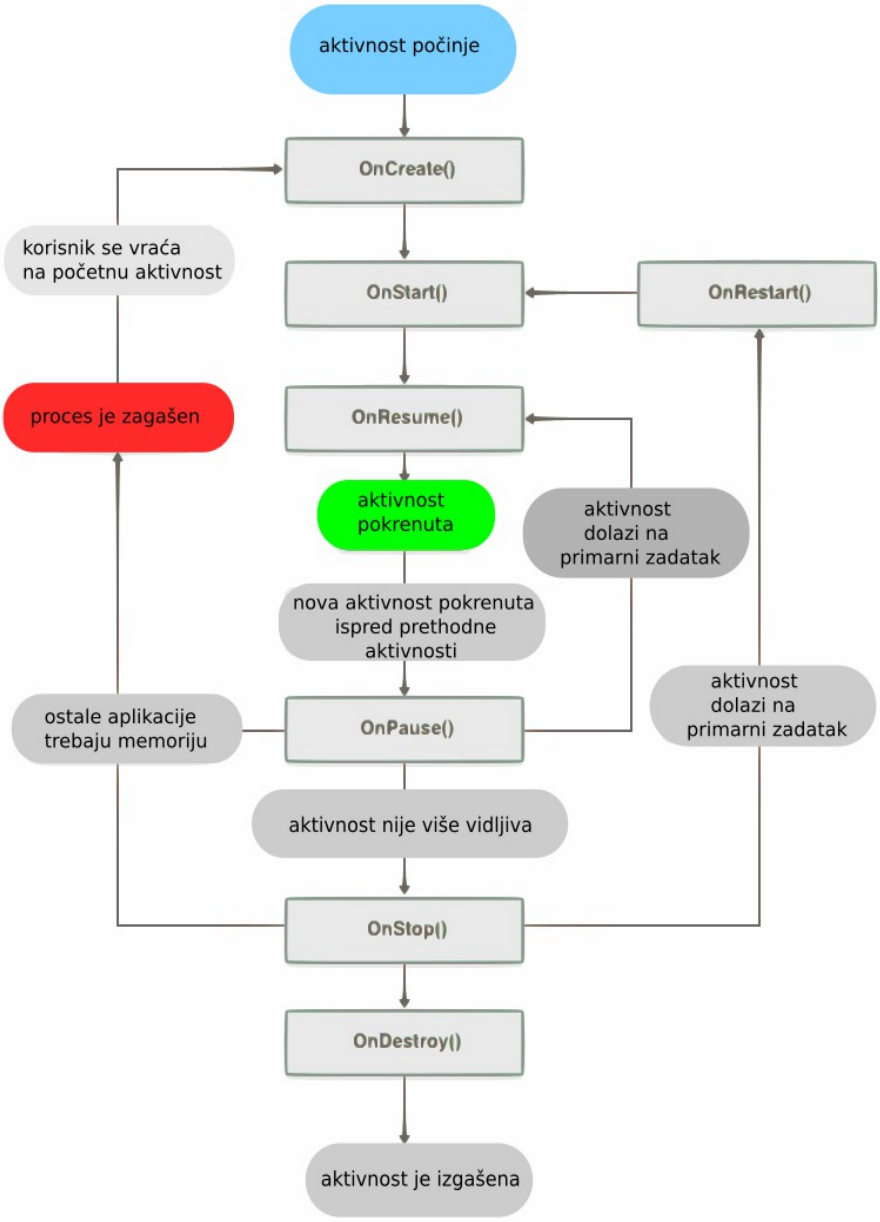
- void onCreate(),
- void onStart(),
- void onRestart(),
- void onResume(),
- void onPause(),
- void onStop(),
- void onDestroy().

Sve prethodno navedene metode mogu se nadjačati (izvršava se metoda koja koristi oznaku nadjačavanja te se izvršava umjesto prethodne metode), a i lako se može definirati što treba učiniti kada se promijeni stanje. Jedina nužna metoda je *onCreate()* jer se u njoj definiraju početne postavke.

Aktivnost se stvara metodom *onCreate()*, a pokreće metodom *onStart()*. Tada je aktivnost u aktivnom stanju, što znači da radi. Metodom *onPause()* prelazi u pauzirano stanje iz kojeg se u aktivno stanje može vratiti pozivom metode *onResume()*. Da bi se vratio u prvi plan iz stanja kada se ne vidi na ekranu, mora se pozvati metoda *onRestart()*. Metodom *onDestroy()* se gasi.

Operacijski sustav se brine za pohranu stanja aktivnosti što podrazumijeva gašenje aktivnosti kako bi se oslobodila memorija. Da bi to bilo moguće, treba implementirati metodu

onSaveInstanceState() koju Android poziva kada postoji mogućnost da će aktivnost biti srušena – dakle, prije poziva *onPause()* metode. Stvara se *Bundl*e objekt u koji se pohranjuje trenutno stanje aktivnosti u obliku parova ime-vrijednost.



Slika 3 Prikaz stanja aktivnosti Android mobilnog sustava, izvor: izradio autor prema uz pomoć sheme preuzete sa http://jcavar.me/androidworkshop/2014/04/09/class2/

Kada se ta aktivnost ponovno pokrene, Bundle objekt se predaje metodama *onCreate()* i *onRestoreInstanceState()* (koja se izvršava nakon *onStart()* metode) kako bi se moglo vratiti stanje u kojem je aktivnost bila prije nego je srušena.

4. Razvoj aplikacija za Android

Razvijanje aplikacija sve je popularniji posao među mladima. Prvenstveno ovaj tip tržišta nastaje 2007. godine, te bilježi kontinuirani rast, često ga zovi i zanimanjem budućnosti. Trenutno Google dućan bilježi 1.600.000,00 aplikacija.[8]

4.1 Korisničko sučelje

Prvi način dizajniranja korisničkog sučelja je proceduralni dizajn. Takva vrsta dizajniranja odnosi se na pisanje Java koda, dok pisanje XML koda (eng. *Extensibe Markup Language*) se naziva deklarativno dizajniranje. U praksi se za kreiranje grafičkog sučelja uglavnom koristi deklarativno dizajniranje.

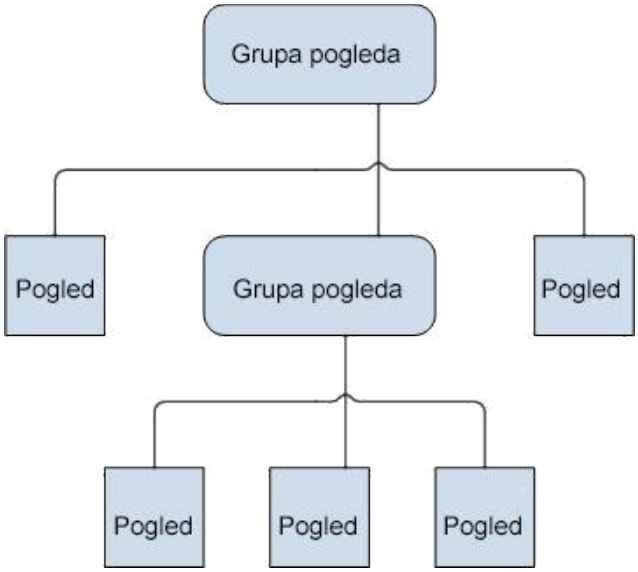
Kreiranjem sučelja, aktivnosti dobivaju svoju funkcionalnost, odnosno vidljivost na zaslonu uređaja. Na taj se način omogućava interakcija s korisnikom. Osnovne jedinice korisničkog sučelja su objekti pogled (eng. *View*) i grupa pogleda (eng. *ViewGroup*):

- pogled – objekt čija podatkovna struktura u sebi nosi zapis izgleda i sadržaja određenog pravokutnog područja na zaslonu te upravlja iscrtavanjem elemenata, pomicanjem sadržaja na zaslonu i ostalim faktorima koji utječu na izgled definiranog dijela zaslona. U hijerarhijskom stablu objekti pogled su listovi stabla. Android raspolaže s već gotovim skupovima objekata ove vrste kao što su gumbi, kvadratići za odabir i slično. Ovi objekti nasljeđuju klasu *View*;

- grupa pogleda – posebna vrsta objekta pogled koji sadrži i upravlja skupinom zavisnih objekata pogleda i grupa pogleda čime je omogućena kompleksnost prikaza korisničkog sučelja. Objekti ove vrste su instance klase *ViewGroup*.

Na slici (Slika 3) je prikazana hijerarhijska ovisnost objekata pogleda i grupa pogleda. Iscrtavanje elemenata stabla započinje od korijena stabla tako što aktivnost poziva svoju *setContentView()* metodu i Android sustavu predaje referencu na korijenski objekt. Svaki podčvor iscrtava se sam pozivanjem *draw()* metode i to pod uvjetom da čvor sam postavlja zahtjev za lokacijom i veličinom. Roditeljski čvor (grupa pregleda) donosi konačnu odluku o njegovoj lokaciji na zaslonu te o veličini prostora za iscrtavanje podčvorova.

8 Količina aplikacija na tržištu, NUMBER OF APPS AVAILABLE IN LEADING APP STORES AS OF JULY 2015, dotupno na: http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/ (01.09.2015)

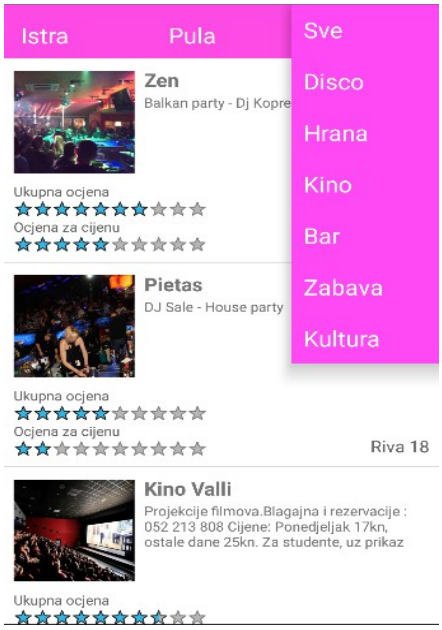


Slika 4 Hijerarhija prikaza elemenata korisničkog sučelja, izvor*: http://www.fer.unizg.hr*

Na slici (Slika 3) je prikazana hijerarhijska ovisnost objekata pogleda i grupa pogleda. Iscrtavanje elemenata stabla započinje od korijena stabla tako što aktivnost poziva svoju *setContentView()* metodu i Android sustavu predaje referencu na korijenski objekt. Svaki podčvor iscrtava se sam pozivanjem *draw()* metode i to pod uvjetom da čvor sam postavlja zahtjev za lokacijom i veličinom. Roditeljski čvor (grupa pregleda) donosi konačnu odluku o njegovoj lokaciji na zaslonu te o veličini prostora za iscrtavanje podčvorova.

4.2 Izbornici

Aplikacijski izbornici često su korišten dio korisničkog sučelja pomoću kojeg se mogu koristiti funkcije aplikacije i mijenjati njene postavke. Struktura im se stvara pomoću hijerarhije elemenat prikaza, no nju se tako ne definira. Umjesto toga, definiraju se *onCreateOptionsMenu()* ili *onCreateContextMenu()* metode za aktivnost i deklariraju se elementi koji se žele uključiti u izbornik. Pri pokretanju, Android će automatski stvoriti hijerarhiju elemenata prikaza i nacrtati svaki element izbornika. Također, postoje mnoge mogućnosti izrade vlastitog izbornika, kao u slučaju moje aplikacije koje sam izgradio pomoću spinner elemenata ili padajućih izbornika.

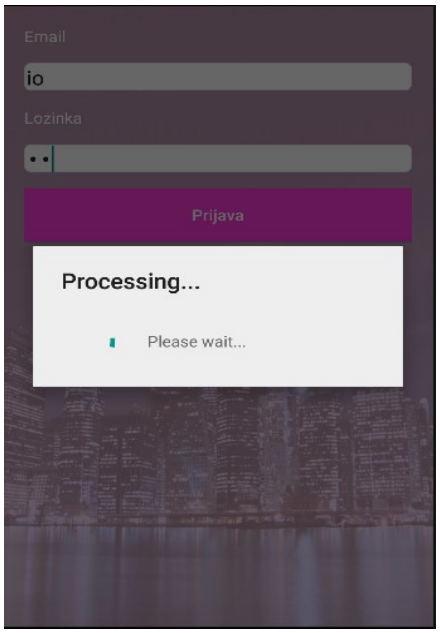


Slika 5 Primjer izbornika, izvor: izradio autor

Elementi izbornika i izgled aplikacije mogu se definirati u XML datoteci. Primjer izbornika može se vidjeti pri vrhu slike (Slika 4).

5.3 Dijalozi

Dijalog (Slika 5) je obično mali prozor koje se pojavi iznad trenutno aktivne aktivnosti koja odlazi u stanje pauze dok dijalog ne preuzme svu interakciju s korisnikom.



Slika 6 Primjer dijaloga, izvor: izradio autor

Dijalog se koristi za obavijesti i kratke aktivnosti koje su u neposrednom odnosu s aplikacijom koje se izvodi.

4.4 Događaji unutar korisničkog sučelja

Postoje dva načina na koji se može saznati da je korisnik uopće odabrao neku od aktivnosti ili stavku u izborniku sučelja, a potrebno je i definirati način na koji korisnik vrši interakciju s određenim komponentama sučelja:

• definiranje slušača događaja (eng. *Event listener*) – najčešći način osluškivanja događaja. *View* klasa sadrži kolekciju ugniježđenih sučelja *On<event>Listener* od kojih svako sadrži povratnu metodu *On<event>()*,

• nadjačavanje (eng. *Override*) postojeće povratne metode za *View* klasu – koristi se u slučaju implementiranja vlastite *View* klase i osluškivanja posebnih događaja koji se u njoj pojavljuju.

4.5 Podatkovni resursi

Podatkovni resursi podrazumijevaju vanjske podatke koji se koriste u aplikaciji, a nisu dio programskog koda. Podatkovni resursi u aplikacijskom paketu smješteni su u vlastitom direktoriju pod nazivom *res/*. Takvo odjeljivanje se prvenstveno provodi zbog lakšeg upravljanja, ažuriranja i manipulacije podacima. U tablici koja slijedi (Tablica 2) prikazana je ovisnost tipa podatkovnog resursa i lokacije u paketu.

Treba napomenuti da se XML datoteke pretvaraju u binarni zapis kako bi se zbog veće efikasnosti mogle brže učitati.

Tablica 1 Ovisnost tipa podatkovnog resursa i lokacije u paketu, izvor: izradio autor

LOKACIJA	PODATKOVNI RESURS
/res/layout/	Datoteke prikaza
/res/drawable/	Slike
/res/values/	Jednostavne vrijednosti (boje, stilovi)
/res/raw/	Sirovi (eng. *Raw*) podaci

Mala slova, brojevi, točka i podvlaka dozvoljeni su znakovi u zapisu podatkovnih resursa. Prilikom prevođenja stvara se posebna klasa naziva *R.java* koja sadrži identifikatore podatkovnih resursa.

Standardni Java podatkovni resursi referencirani su po ključevima tipa String. Ova klasa sastoji se od niza potklasa, svaka za različitu vrstu podataka.

4.6 Upotreba resursa u kodu

Podaci se u kodu pozivaju preko identifikatora sintaksa *findViewById(R.id.naziv* idetifikacije). Neke podatke, kao npr. *TextView* nije potrebno pozivati u kodu, tj. pozivaju se samo oni s kojima se ima namjera ostvariti neka promjena ili reakcija.

4.7 Upotreba resursa u drugim podatkovnim resursima

Podaci se mogu referencirati i iz drugih resursa (npr. XML datoteke). Takav način referencije koristi se ponajviše prilikom izrade stilova i datoteka prikaza. Za referenciranje jednog resursa iz drugog koristi se notacija "@".

4.8 Upotreba sistematskih resursa

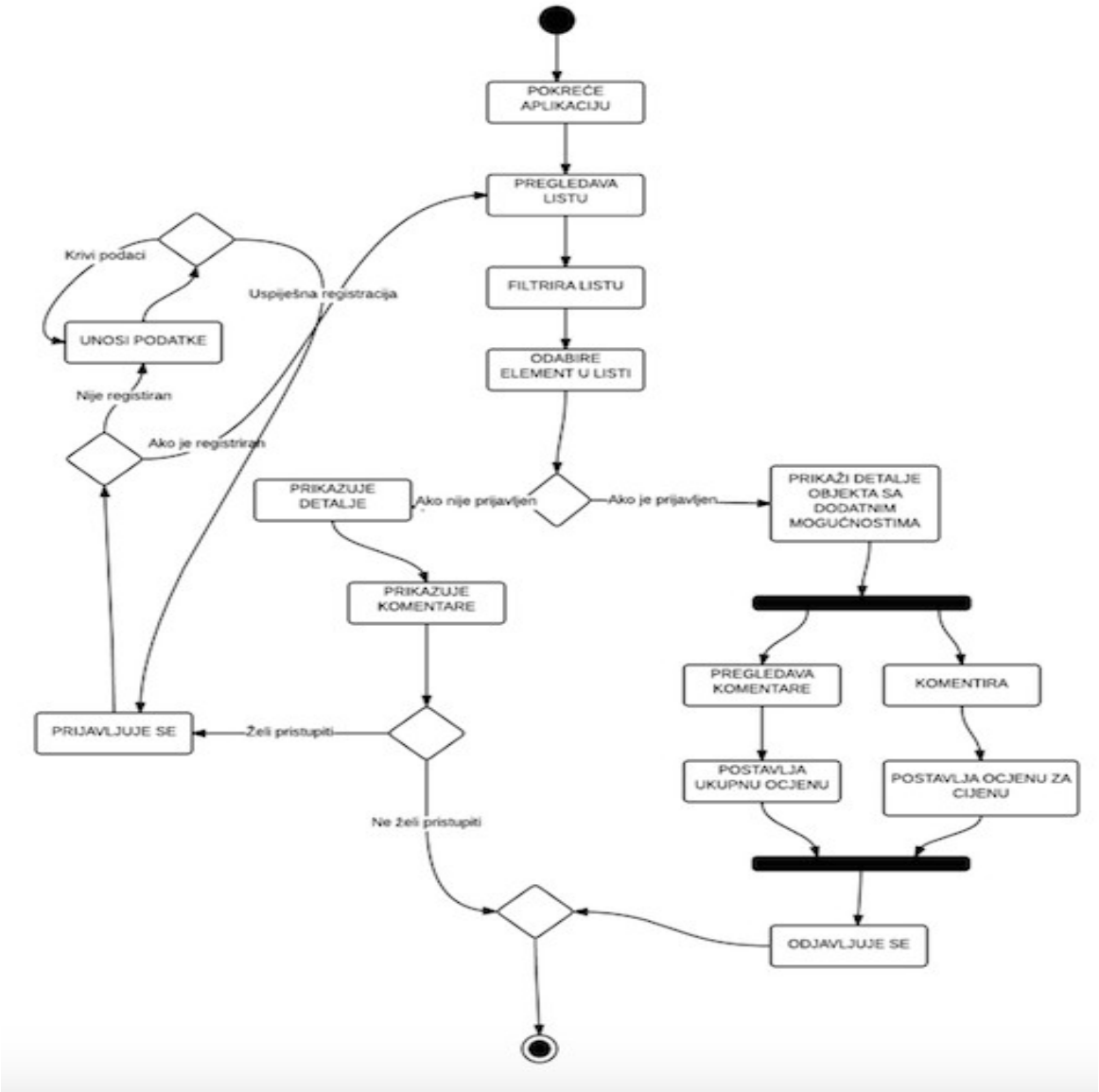
Upotreba sistematskih resursa odnosi se na upotrebu resursa koje već koriste neke izvorne aplikacije. Korištenje ovih podatkovnih resursa slično je korištenju vlastitih resursa s razlikom da im se pristupa u klasi *android.R*.

5. Prikaz razvoja aplikacije

Program za izradu aplikacije korišten je Android studio API. Nastao je 2013. godine i namjenjen je isključivo programiranju Android aplikacija. Netom prije koristio se Eclipse s ugrađenim paketima kao dodatak za Android aplikacije. Android studio je besplatan software i dostupan je za preuzimanje putem weba.

5.1 Aplikacija Where2Go

Aplikacija je namijenjena svim dobnim generacijama od omladine na dalje, a ponajviše onima koji vole putovati i posjetiti poznate lokale, kafiće, pubove i mjesta koja nude ugodnu atmosferu i/ili dobru zabavu.



Slika 7 Prikaz dijagrama aktivnosti, izvor: izradio autor

Sam koncept aplikacije rađen je s ciljanim dizajnom u kojem nema puno texta, pa se većina korisnika može snalaziti iako ne vlada jezikom (aplikacija je na hrvatskom jeziku). Rađena je za lokalno tržište, no nije isključena mogućnost da bude korištena od strane turista koji provremeno borave u Hrvatskoj. U sljedećim poglavljima objašnjeno je korisničko sučelje i funkcionalnost.

Nakon otvaranja same aplikacije (i registracije po želji), korisnik otvara padajući izbornik koji na izbor nudi regije Republike Hrvatske. Odabirom željene regije, otvara se mogućnost izbora grada koji se nalazi u izabranoj regiji. Nakon što korisnik odabere grad koji posjećuje ili želi posjetiti, na zaslonu mu se pojavljuju kafići, pubovi, poznati lokali i mjesta koja može posjetiti. Otvaranjem jednog od ponuđenih objekata, može se vidjeti fotografija i opis objekta, te ocjena i komentari drugih korisnika koji su odabrani objekt već posjetili. Korisnik tako dobiva pregled (ne)zadovoljnih korisnika i detaljan uvid u objekte koje želi posjetiti. Na temelju toga može odlučiti hoće li objekt zapravo i posjetiti ili mu više odgovara atmosfera nekog drugog objekata. Najvažnije, nakon posjeta nekome od tih mjesta, korisnik može u komentaru ostaviti povratnu informaciju. Time će i sami vlasnici tih objekata moći vidjeti komentare svojih korisnika. To im daje mogućnost uvida u zadovoljstvo korisnika i prostor za preinake u poslovanju kako bi korisnike i zadržali.

5.2 Where2go kao socijalni model

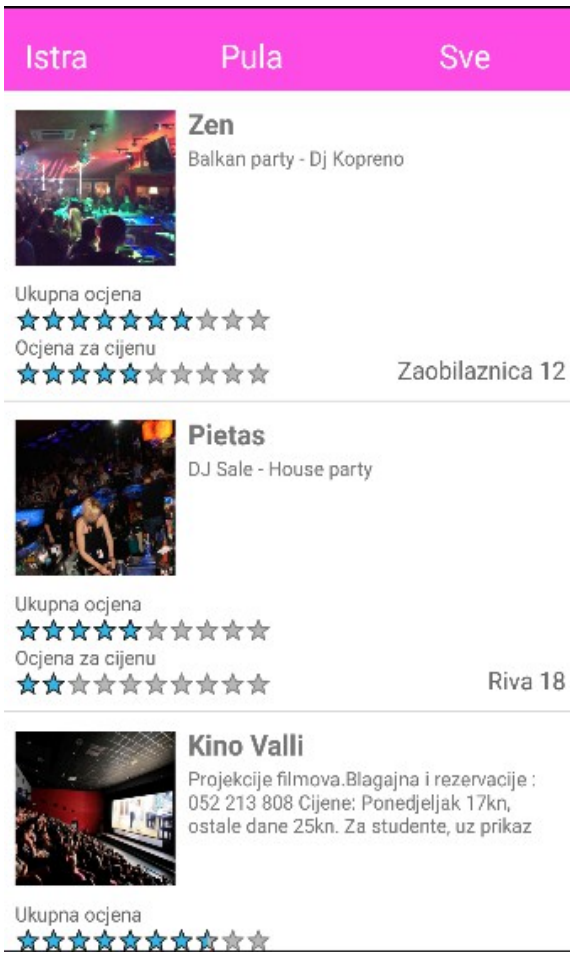
Budući da je Where2Go aplikacija zamišljena kao mala socijalna mreža, neki sastavni dijelovi mogu se usporediti s popularnim društevim mrežama. Sama aplikacija može biti i alat koji će poslužiti vlasnicima objekata da promoviraju svoj posao kroz kvalitetne usluge, ali i pozitivne recenzije svojih gostiju. Budućnost Where2Go aplikacije predviđa se u hibridnom socijalnom modelu, gdje bi postojala sporedna aplikacija preko koje bi vlasnici objekata sami uređivali podatke o objektu, dok bi se pretplata odrađivala po SaaS modelu (*Software as a service*).

5.3 Aplikacija Where2Go u poslovnom modelu

Aplikacija Where2Go može se predstaviti kao jedan model mobilnog poslovanja. Sam naziv aplikacije (Where to go – gdje ići) već asocira na njezinu funkciju u stvarnome životu. Pretpostavlja se da će aplikacija zainteresirati srednji broj korisnika, specifično mladih osoba koje vole putovati Hrvatskom i istraživati nova, zanimljiva mjesta. Tako će lakše moći razmjenjivati informacije o mjestima koja su posjetili i dati preporuke drugima. Vlasnici lokala moći će koristiti aplikaciju kako bi svoje objekte oglašavali. Model zarade biti će ostvaren putem reklama koje će se pojavljivati nakon ulaska u aplikaciju.

5.4 Početna aktivnost

Početna aktivnost je prvi prikaz koji korisnik vidi na ekranu kada uključuje aplikaciju. Prva aktivnost prikazuje listu sa objektima s obzirom na filter. Korisnik pomicanjem prsta po ekranu lista podatke. Klikom na gornji menu, filtrira podatke po odabiru, dok klikom na objekt, pali se sljedeća aktivnost.



Slika 8 Prikaz početne aktivnosti, izvor: izradio autor

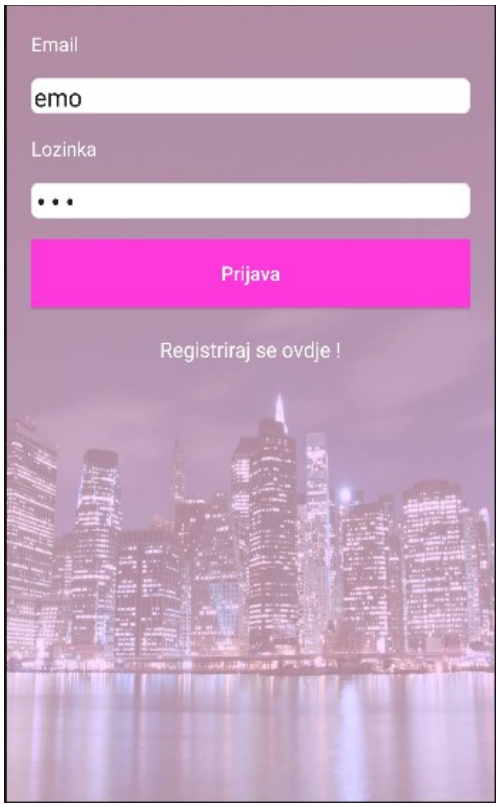
Korisnik ima opći uvid u fotografiju koja predstavlja objekt, ime, kratki opis, adresu i ocjene koje su rezultat recenzija korisnika.

5.5 Sekundarna aktivnost – detaljni opis

U sljedećoj aktivnosti koja se otvara putem klika na željeni objekt, pokreće se detaljniji opis objekta te se korisniku nude opcije «Prijava», gumb ikone koja otvara recenzije te text *Where2Go* koja klikom vodi na novu aktivnost. Prvi dvije opisane aktivnosti služe samo kao pregled podataka. Korisnik koji nije prijavljen nema mogućnost ostvariti interakciju sa ostalim korisnicima.

5.6 Aktivnost prijava

Aktivnost nudi mogućnost prijave u sustav unosom polja za e-mail i lozinke. Ako su lozinka i e-mail ispravni, pokrenuti će se nova aktivnost. U slučaju krive lozinke ili e-maila, korisnik dobiva poruku kojom je obaviješten da nije unio dobre podatke za e-mail tj. lozinku.

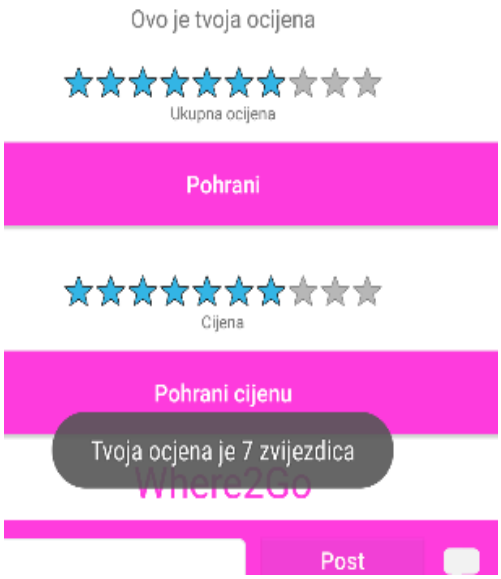


Slika 9 Prikaz aktivnosti prijave, izvor: izradio autor

Ako internet konekcija nije dostupna, pojaviti će se poruka upozorenja. U slučaju da korisnik nije registriran, ispod gumba za prijavu, nalazi se text: «*Registriraj se ovdje* !». klikom na taj text otvara se aktivnost za registraciju.

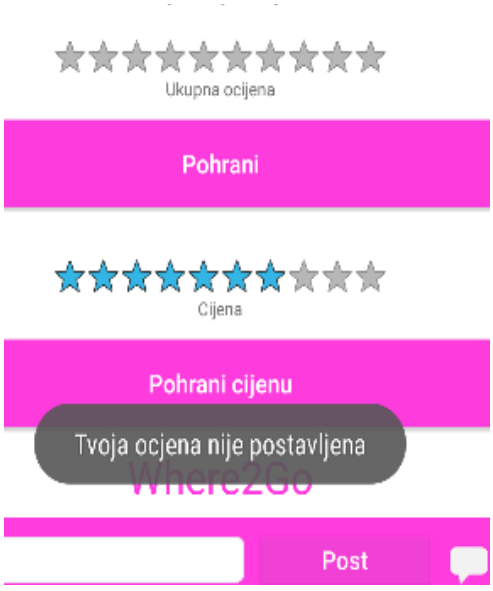
5.7 Prijavljena aktivnost

Ova aktivnost otvara se kada je korisnik uspješno prijavljen u sustav. Slična je kao sekundarna aktivnost opisa, osim što sada postoje neke dodatne mogućnosti. To su ocjenjivanje i postavljanje recenzije.



Slika 10 Ocjenjivanje objekata, izvor: izradio autor

Prvim *zvjezdanim redom (eng. Rating bar)* ocjenjuje se ukupan dojam lokala, dok drugi *zvjezdani red* označava ocjenu korisnika za cijene objekta. U slučaju da je korisnik ocijenio dotični objekt, ocjena će biti prikazana u *zvjezdanom redu*.

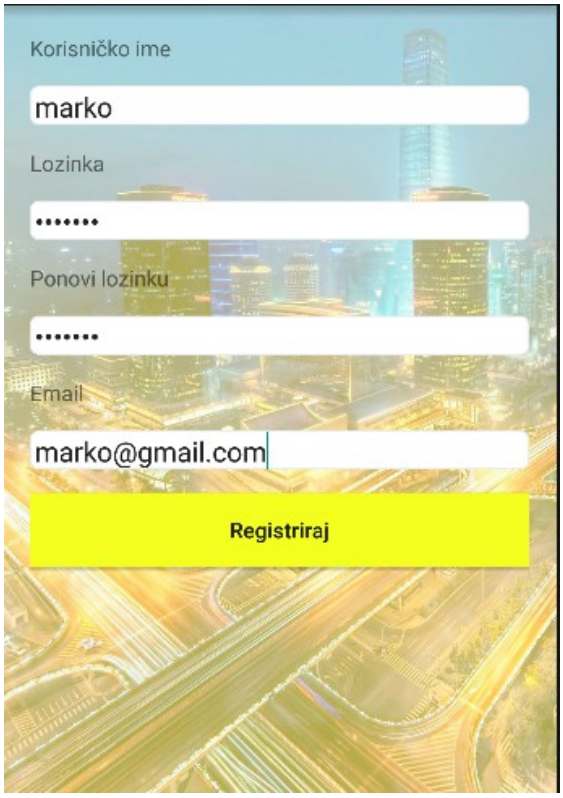


Slika 11 Upozorenje kada korinik želi pohraniti ocjenu koju nije postavio, izvor: izradio autor

Klikom na bar, ocjena se mjenja, te je promjena moguća u bilo kojem trenutku. Svaki put kada se promijeni ocjena ili se postavi nova, sustav će obavijestiti korisnika. Tek nakon klika na gumb za ocjenjivanje, ocjena postavljena u *zvjezdanom redu* bit će poslana na bazu.

5.8 Registracijska aktivnost

Registracijska aktivnost pokreće se iz prijavljene aktivnosti klikom na text koji inicira registraciju. Ovdje se pojavljuju tri polja za unos, ime – koje će biti prikazano pri recenziji, e-mail adresa i lozinka koja zahtjeva ponavljanje radi izbjegavanja greške.



Slika 12 Prikaz registracijske aktivnosti, izvor: izradio autor

Klik na gumb šalje podatke na bazu i provjerava valjanost unesenih podataka. U slučajevima praznog polja, unosa krive lozinke, korisniku se šalje namijenjeno upozorenje. Ako je registracija uspješna, korisniku se na e-mail šalje sigurnosna lozinka koju korisnik mora unijeti pri prvoj prijavi. Nakon uspješne prijave opet se otvara prijavna aktivnost.

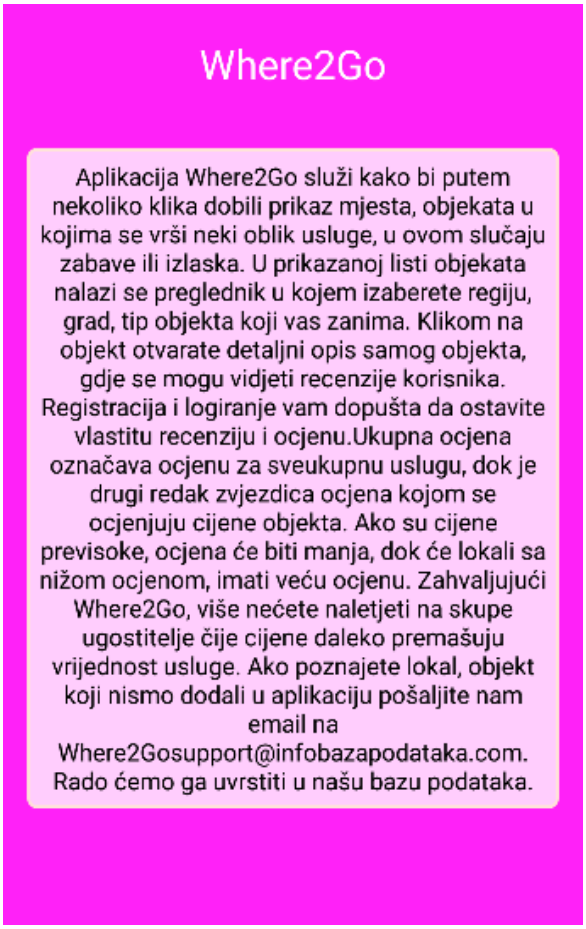
5.9 Sigurnosna Aktivnost

Sigurnosna aktivnost pali se samo kada se korisnik prvi puta prijavljuje. U aktivnosti se traži upis sigurnosnog koda koji je poslan korisniku pri registraciji. Nakon uspješnog unosa sigurnosnog koda, otvara se početna aktivnost te je korisnik prijavljen.

5.10 Where2Go aktivnost

U ovoj aktivnosti, koja se pokreće klikom na *text*, nalaze se upute za korištenje aplikacije i e-mail preko kojeg se može kontaktirati administracija aplikacije. Služi kako bi se korisnici bolje

upoznali sa mogućnostima aplikacije, ali i sudjelovali u popunjavanju baze objekata. Budući da nije moguće doći do informacija o svakom pojedinom registriranom objektu, korisnici aplikacije bit će dodatna pomoć.



Slika 13 Prikaz Where2Go aktivnosti, izvor: izradio autor

Preko e-mail adrese moguće je direktno stupiti u kontakt sa administracijom aplikacije. Ovo nije uobičajena opcija u Android aplikacijama, no može povećati zadovoljstvo korisnika, jer sami mogu dati prijedloge i indirektno sudjelovati u daljnjem razvoju.

6. Zaključak

U ovome završnom radu opisan je proces izrade aplikacije za Android operativni sustav korištenjem alata Android studio. Prikazani su osnovni elementi alata za izradu, te osnovne komponente operativnog sustava Android. Android je operacijski sustav široke primjene, budući da je kompatibilan sa različitim hardverskim komponentama. Upravo je to razlog zašto je najpopularniji operativni sustav pametnih telefona.

Proces izrade aplikacije obuhvaća i planiranje poslovnog modela, kao i odabir najpovoljnijeg modela s obzirom na funkcije aplikacije (Besplatni model). Hipoteza završnog rada temelji se na pretpostavci da mobilne aplikacije povećavaju razinu podjele informacija, poboljšavaju poslovanje i kvalitetu življenja. Funkcije ostvarene u izradi aplikacije Where2Go potvrđuju hipotezu – željene informacije postaju dostupne u nekoliko klika na mobilnom uređaju. Aplikacija je namjenjena svim dobnim skupinama koje koriste telefone sa Android operativnim sustavom na području Republike Hrvatske.

Android kontinuirano poboljšava perfomanse sustava, radi na optimizaciji i uvođenju novih stavki kako bi se unaprijedilo korisničko iskustvo. Prema tome moguće je pretpostaviti da će i dalje uvjerljivo držati poziciju vodećeg operativnog sustava na mobilnim telefonima.

Literatura:

Knjige:

1. Panian, Željko, *Elektroničko poslovanje druge generacije*, Zagreb, 2013.

Internet izvori:

1. Radić, Drago: Mobilni uređaji, s Interneta, http://www.informatika.buzdo.com/pojmovi/mobile3.htm (08.09.2015)
2. Vogella: Android Development, s Interneta, http://www.vogella.com/android.html (05.09.2015)
3. Povijest mobilne tehnologije, ŠTO SE DOGAĐALO 40 GODINA, dostupno na: http://mob.hr/povijest-mobilne-telefonije-sto-se-dogadalo-u-40-godina/ (01.09.2015)
4. Udio operativnih sustava na tržištu, SMARTPHONE OS MARKETSHARE 2015, dostupno na: http://www.idc.com/prodserv/smartphone-os-market-share.jsp (02.09.2015)
5. Generacije mobilnih usluga, MOBILNI UREĐAJI, dostupno na: http://www.informatika.buzdo.com/pojmovi/mobile-1.htm, (03.09.2015).
6. Količina aplikacija na tržištu,NUMBER OF APPS AVAILABLE IN LEADING APP STORES AS OF JULY 2015, dotupno na: http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/ (05.09.2015)
7. Naplatni modeli, MONETIZACIJA MOBILNIH APLIKACIJA: KAKO ODABRATI NAPLATNI MODEL?, dotupno na: http://www.pokreniposao.hr/monetizacija-mobilnih-aplikacija-kako-odabrati-naplatni-model/ (04.09.2015)

Popis slika

Slika 1 Arhitektura operacijskog sustava Android.....11

Slika 2 Razlika između Sun Java i Dalvik virtualnog stroja.....13

Slika 3 Prikaz stanja aktivnosti Android mobilnog sustava.....19

Slika 4 Hijerarhija prikaza elemenata korisničkog sučelja.....21

Slika 5 Primjer izbornika.....22

Slika 6 Primjer dijaloga.....22

Slika 7 Dijagram stanja aktivnosti.....25

Slika 8 Prikaz početne aktivnosti.....27

Slika 9 Prikaz aktivnosti prijave.....28

Slika 10 Ocjenjivanje objekata.....29

Slika 11 Upozorenje kada korinik želi pohraniti ocjenu koju nije postavio.....29

Slika 12 Prikaz registracijske aktivnosti.....30

Slika 13 Prikaz Where2G aktivnosti.....31

Popis tablica:

Tablica 1 Ovisnost tipa podatkovnog resursa i lokacije u paketu.....23