Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aleš Watzak

Mobilna aplikacija za branje aktualnih novic iz več virov

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Rok Rupnik

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: 00381/2013
Datum: 02.04.2013

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

Kandidat: **ALEŠ WATZAK**

Naslov: **MOBILNA APLIKACIJA ZA BRANJE AKTUALNIH NOVIC IZ VEČ VIROV**

MOBILE APPLICATION FOR READING LATEST NEWS FROM MULTIPLE SOURCES

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Izdelajte aplikacijo, ki bo uporabnikom mobilnih naprav z operacijskim sistemom Android omogočala spremljanje aktualnih novic. Aplikacija naj bo sestavljena iz dveh delov. Strežniški del naj predstavlja spletna aplikacija, razvita v skriptnem jeziku PHP. Njena naloga naj bo združevanje novic večih spletnih virov na enem mestu, shranjevanje le teh v podatkovno bazo MySQL in serviranje novic odjemalcem. Drugi del naj predstavlja mobilna aplikacija razvita za mobilno platformo Android. Slednja naj uporabnikom omogoča tudi izbiro virov in kategorij novice po svojem izboru. Za osveževanje novic uporabite storitev Google Cloud Messaging, ki omogoča realizacijo osveževanja v določenem intervalu.

Mentor:

doc. dr. Rok Rupnik

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Aleš Watzak, z vpisno številko **63980179**,
sem avtor diplomskega dela z naslovom:

Mobilna aplikacija za branje aktualnih novic iz več virov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Roka Rupnika,
- 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 1. julij 2013 Podpis avtorja:

Rad bi se zahvalil vsem, ki so mi ves čas stali ob strani in me podpirali na poti do uspešne izvedbe diplomskega dela. Še posebej bi se zahvalil svoji mami in bratu za potrpežljivost in vzpodbudo. Velika zahvala gre tudi mentorju, doc. dr. Roku Rupniku in zvestim prijateljem Špeli, Tomi in Filipu.

Kazalo

Seznam uporabljenih kratic in simbolov

Povzetek

Abstract

1 Uvod **1**

2 Razvojna orodja in tehnologije **3**

2.1 Android 3

2.2 Različice platforme Android 4

2.3 Podrobnosti arhitekture Androida 5

2.3.1 Jedro Linux 5

2.3.2 Knjižnice 6

2.3.3 Zagonsko okolje Android 6

2.3.4 Aplikacijsko ogrodje 7

2.3.5 Aplikacije 7

2.4 Komponente Androida 7

2.4.1 Aktivnosti 8

2.4.2 Storitve 8

2.4.3 Prejemniki oddajanja 8

2.4.4 Ponudniki vsebine 10

2.5 Google Cloud Messaging 10

2.6 Java 11
2.7 PHP 11
2.8 Apache 11
2.9 MySQL 12
2.10 Android Development Tools 12
2.10.1 phpMyAdmin 12
2.10.2 SQLite Manager 12

3 Aplikacija za spremljanje novic 13
3.1 Analiza 14
3.1.1 Diagram primerov uporabe 14
3.2 Načrtovanje in razvoj 15
3.2.1 Načrtovanje podatkovne baze 15
3.2.2 Struktura projekta 18
3.2.3 Dovoljenja 20
3.2.4 Uporabniški vmesnik 21
3.2.5 Komponente aplikacije 22
3.2.6 Aktivacija in prehodi med komponentami 24
3.2.7 Osveževanje vsebin s pomočjo storitve GCM 25
3.3 Testiranje 26

4 Strežnik novic 27

5 Zaključek 29

Seznam slik 31

Seznam tabel 32

Literatura 33

Seznam uporabljenih kratic in simbolov

ADT - Android Development Tools; razvojna orodja za platformo Android

GCM - Google Cloud Messaging; Googlova storitev v oblaku za pošiljanje sporočil

API - Application Programming Interface; programski vmesnik

HTML - Hypertext Markup Language; označevalni jezik za izdelavo spletnih vsebin

HTTP - Hypertext Transfer Protocol; komunikacijski protokol za prenos informacij na spletu

ID - Identifier; enolični identifikator

IDE - Integrated Development Enviroment; integrirano programsko okolje

JSON - JavaScript Object Notation; format za izmenjavo podatkov

PHP - Hypertext Preprocessor; skriptni jezik za razvoj spletnih aplikacij

RSS - Rich Site Summary; zgoščeni povzetek strani

SDK - Software Development Kit; paket razvojnih orodij

SQL - Structured Query Language; strukturirani povpraševalni jezik za delo s podatkovnimi bazami

XHTML - Extensible HyperText Markup Language; označevalni jezik iz družine XML za izdelavo spletnih vsebin

XML - Extensible Markup Language; strukturiran format podatkov za izmenjavo dokumentov

URL - Uniform Resource Locator; niz znakov, ki predstavlja spletni naslov

Povzetek

Diplomska naloga opisuje razvoj mobilne aplikacije za naprave z Android operacijskim sistemom. Omogoča spremljanje aktualnih novic, vremenske napovedi ter prometnih informacij. Spremljanje novic je na voljo na vseh lokacijah z dostopom do spleta in ni omejena samo na fizične lokacije kot je pisarna z namiznim računalnikom. V ta namen smo razvili dve aplikaciji. Prva, predstavlja strežniški del in se izvaja na spletnem strežniku Apache. Drugi del aplikacije je odjemalec, katerega uporabnik namesti na svoji Android mobilni napravi.

V začetku diplomskega dela predstavimo uporabljena razvojna orodja in tehnologije. Temu sledi podrobnejša predstavitev mobilne aplikacije na uporabnikovi strani. Naloga odjemalca je prenos novic in drugih informacij s strežnika ter prikaz le teh na zaslonu uporabnikove naprave. Strežnik za prenos novic uporablja format XML, deloma tudi XHTML zato smo implementirali ustrezno razčlenjevanje take strukture podatkov. Za hitro dostavo novic ob izbranem času smo uporabili storitev Google Cloud Messaging s push tehnologijo. Strežnik sledi večim virom novic in razčleni njihove zgoščene povzetke strani RSS. Novice nato kategorizira in shrani v bazo podatkov MySQL. Podobno obdela vremenske in prometne podatke. Odjemalec lahko do teh dostopa kadarkoli. Diplomo zaključimo z nekaj idejami za nadaljnje izboljšave.

Ključne besede:

Android, aplikacija, GCM, mobilna naprava, strežnik, odjemalec, novice

Abstract

Thesis covers development of an application for mobile devices running Android operating system. It enables users to keep up to date with the latest news, weather and traffic report. News delivery service is available wherever there is Internet access and not limited to a physical location like an office with desktop computer. To this end we developed two applications. First represents server part of the application and runs on Apache web server. The second part is a client application which users install on their Android mobile devices.

We start this thesis with introduction of development tools and technology used during the development followed by a detailed presentation of our client side mobile application. Client transfers news articles and other data from the server and presents it on the users screen. Server delivers news in XML and partly in XHTML format so a proper parsing of this data is implemented. We used Google Cloud Messaging and its push technology for prompt delivery of any new stories at chosen intervals. Server keeps track of multiple news sources and parses their RSS feeds to get latest news. It then categorizes articles and saves them in MySQL database. Similarly it processes and saves weather and traffic data. Client application can then access all this data at any time. We conclude the thesis with some ideas for future upgrades.

Key words:

Android, application, GCM, mobile device, server, client, news

Poglavje 1

Uvod

Hiter razvoj računalništva, pa tudi tehnologije na splošno, zagotavlja da tako razvijalcem kot uporabnikom še zdaleč ni dolgčas. Če so še kakšno desetletje ali dve ljudje večinoma pridobivali informacije iz tiskanih medijev in televizijskih ekranov, pa še to le občasno v vnaprej določenih intervalih, je danes precej drugače. Dostop do spleta je uporabnikom prinesel raznorazne vsebine praktično na dom ali v pisarno. Tako na pomembne novice ali vremensko napoved ni potrebno čakati več ur do televizijske oddaje ali izida časopisa, pač pa so omenjene vsebine na voljo ob vsakem času, dostop do njih pa le nekaj klikov računaniške miške stran.

Količina informacij danes tako ni problem, saj se število novičarskih spletnih portalov, polnih aktualnih novic, vztrajno veča. A napredek na tem področju je s prihodnom mobilnih naprav še vedno mogoč. Mobilna aplikacija lahko uporabniku omogoča dostop do novic in drugih informacij v vsakem trenutku na vsakem kraju. Potrebo po takšni aplikaciji sem sam pogosto začutil na poti, še posebej kadar sem pričakoval objavo določene novice, a dostopa do računalnika in spleta na poti nisem imel. Morda še bolj pomembne pa so v določenem trenutku lahko izredne novice, vremenska napoved ali prometne informacije. Podatek o tem, da je na cesti kjer smo se vozili prišlo do nesreče, ali pa da je zaradi snežnih zametov zaprt določen mejni prehod, je

namreč veliko bolj uporaben, če zanj izvemo pravočasno, morda še preden smo se podali na pot. Ravno zato sem se odločil razviti aplikacijo, ki bo omogočila praktično takojšen dostop do izrednih novic kot tudi do ostalih aktualnih informacij v trenutku ko sam kot uporabnik to želim. Naš cilj je tako uporabniku omogočiti mobilni dostop do novic in drugih informacij. Današnje mobilne naprave omogočajo prav to, saj ima večina od nas takšno napravo vedno s seboj. Z ustreznim operacijskim sistemom lahko razvijalci programske opreme za uporabnika razvijemo uporabne aplikacije s prijaznim uporabniškim vmesnikom. Eno takšnih aplikacij za mobilne naprave z operacijskim sistemom Android smo razvili tudi mi.

Kot uporabnik Androidnih mobilnih naprav, sem se osločil da razvijem aplikacijo za ta operacijski sistem, čeprav sem si za strežniški del zadal nalogo, da ne sme biti omejen le na serviranje vsebine Androidnim napravam pač pa tudi ostalim, z drugimi operacijskimi sistemi, ki nudijo podobne funkcionalnosti kot Android.

Poglavje 2

Razvojna orodja in tehnologije

2.1 Android

Android je eden on najpopularnejših mobilnih platform. Je operacijski sistem namenjen predvsem malim napravam kot so mobilni telefoni in tablice, širi pa se tudi na tv vmesnike ter postopoma tudi druge naprave kot so ure in podobno. Mobilni telefoni in tablice so danes pravzaprav pravi mali računalniki in je na njih moč opraviti marsikatero opravilo, ki jih sicer večina opravlja na namiznih računalnikih. Android uporabnikom ponuja vrsto različnih aplikacij, iger, pa tudi zabave vsebine, filme, glasbo, knjige. Največja prednost mobilnih naprav je prav mobilnost. Uporabniki lahko svoje priljubljene aplikacije uporabljajo kjerkoli se v določenem trenutku nahajajo.

Google je Android oznanil javnosti konec leta 2007, ko so razvijalci dobili v uporabo preizkusne različice razvojnih orodij. Že v nekaj mesecih je ta orodja v svoj računalnik preneslo več kot milijon razvijalcev. Podobno uspešna je bila tudi prodaja prvih mobilnih telefonov v Združenih državah Amerike, kjer so jih jeseni leta 2008 v le nekaj mesecih prodali več sto tisoč[2]. Nič čudnega torej, da je ta platforma zelo priljubljena tudi med razvijalci.

Google je skupaj z nekaj deset drugimi podjetji ustanovil Open Handset Allience, združenje, ki vsak s svojimi prispevki skrbi za nadaljni razvoj platforme

in razvijalcem tako omogoča razvoj najrazličneših aplikacij, iger in drugih vsebin za Android.

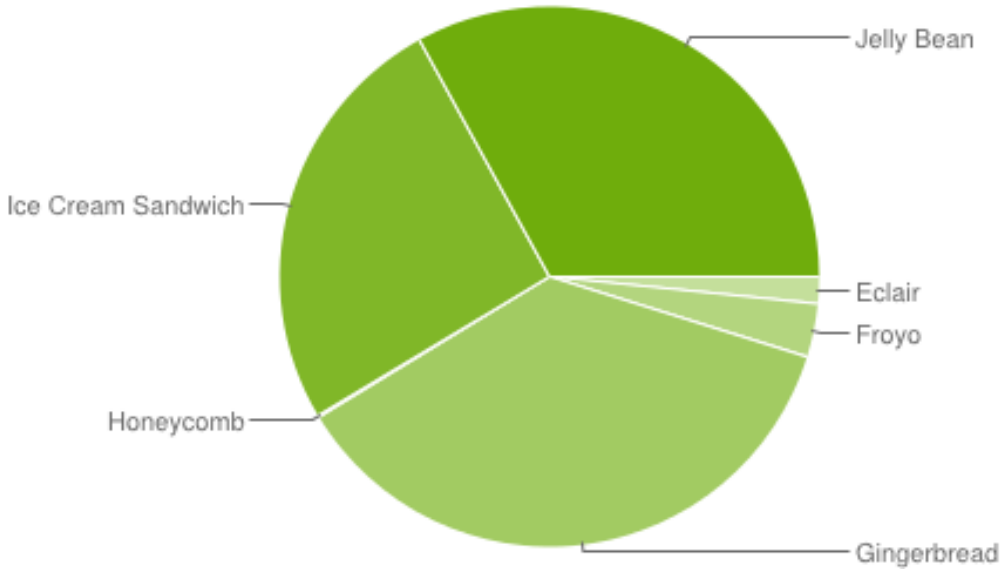
2.2 Različice platforme Android

Danes je Android prisoten na več sto milijonov mobilnih naprav v skoraj 200 državah sveta[4]. Število naprav s posamezno različico platforme se s časom precej spreminja in je odvisna od tega, kako hitro uporabniki menjajo svoje naprave z novejšimi na katerih je naložena tudi višja različica operacijskega sistema Android.

različica	kodno ime	API	razširjenost
1.6	Donut	4	0,1%
2.1	Eclair	7	1,5%
2.2	Froyo	8	3,2%
2.3 - 2.3.2	Gingerbread	9	0,1%
2.3.3 - 2.3.7	Gingerbread	10	36,4%
3.2	Honeycomb	13	0,1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	25,6%
4.1.x	Jelly Bean	16	29,0%
4.2	Jelly Bean	17	4,0%

Tabela 2.1: Različice platforme Android in njena razširjenost.

Google vsakih 14 dni objavi sveže podatke o razširjenosti posamezne različice platforme. Podatki se zbirajo tako da vsaka naprava sporoči nameščeno različico ob tem ko uporabnik obišče Googlovo trgovino z aplikacijami (Google Play Store). Podatki za obdobje 14 dni, ki se je končalo 3. junija 2013 so vidni v tabeli 1, grafično pa stanje prikazuje slika 1.



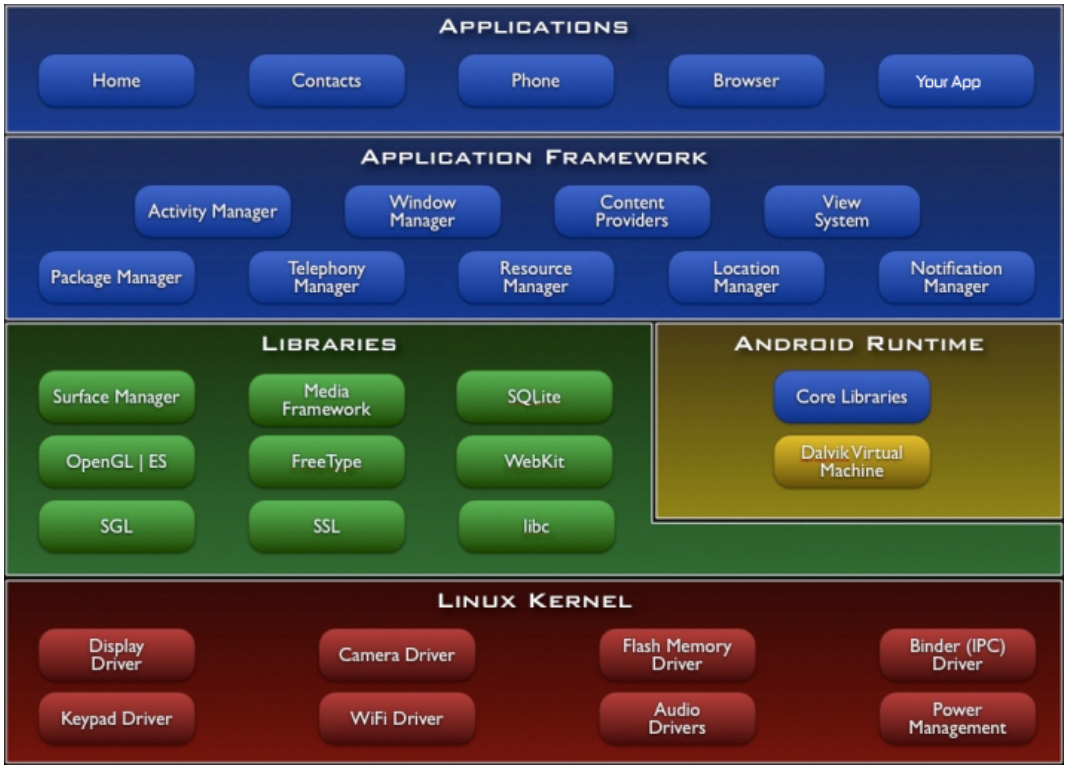
Slika 2.1: Razširjenost različic platforme Android.

2.3 Podrobnosti arhitekture Androida

Za razvijalce je pomembno, da spoznamo Android še nekoliko bolj podrobno, kot le to da je to operacijski sistem na katerem tečejo med drugim tudi naše aplikacije. Android sestavlja pet nivojev, vsak nivo vsebuje več programskih komponent.

2.3.1 Jedro Linux

Najnižji nivo predstavlja temelj celotne arhitekture. Ta nivo je kar Linux jedro, ki skrbi za interakcijo s strojno opremo v napravi. V ta namen vsebuje vse potrebne gonilnike, ki komunicirajo s strojno opremo in jo upravljajo. Primer je npr. gonilnik za kamero (angl. Camera Driver), ki omogoča uporabo v mobilno napravo vgrajene kamere. Linux jedro skrbi tudi za osnovne funkcionalnosti, kot je upravljanje s procesi in pomnilnikom, mrežo in podobno. Prav ta nivo zaradi široke podpore različni strojni opremi, omogoča uspešno integracijo le te skupaj s platformo Android v posamezne mobilne naprave.



Slika 2.2: Arhitektura platforme Android.

2.3.2 Knjižnice

Nivo višje od jedra so knjižnice razvite v programskem jeziku C in C++. Njihov namen je omogočiti napravi obdelavo različnih tipov podatkov. Tako npr. za predvajanje in snemanje multimedijskih vsebin skrbi ogrodje Media. Za prikaz spletnih vsebin v brskalniku se uporablja WebKit, za shranjevanje vsebine v bazo podatkov pa SQLite.

2.3.3 Zagonsko okolje Android

Zagonsko okolje poskrbi za ustrezno okolje v katerem tečejo aplikacije. To okolje ima zaradi majhnosti naprav kar nekaj omejitev, npr. velikost pomnilnika, zmogljivost procesorja in baterije. Te omejitve poskuša reševati navidezni stroj

Dalvik, ki aplikacije optimizira in jih v najboljši meri pripravi za poganjanje v okoljih z omejenimi viri. Zagonsko okolje vsebuje tudi nekaj osnovnih knjižnic napisanih v Javi.

2.3.4 Aplikacijsko ogrodje

Aplikacijsko ogrodje vsebuje množico komponent s pomočjo katerih razvijamo svoje aplikacije. Sem sodijo med drugim vizualne komponente kot so gumbi, stikala, seznami pa tudi različni upravitelji aktivnosti, paketov, oken, telefonije in drugo. Kot primer lahko vzamemo upravitelja lokacije (angl. Location Manager). Ta omogoča aplikaciji določitev trenutne lokacije naprave s pomočjo vgrajene GPS naprave ali pa npr. omrežja mobilnega operaterja.

2.3.5 Aplikacije

V ta nivo sodi zbirka osnovnih aplikacije, vključnenih v operacijski sistem Android. Primeri so aplikacije za elektronsko pošto, spletni brskalnik, SMS, koledar in druge. Sem sodi tud naša aplikacija, ki jo predstavljamo v nadaljevanju.

2.4 Komponente Androida

V Androidu je aplikacija vsebnik (angl. container) večih komponent. Štiri glavni tipi komponent so:

- aktivnosti (angl. Activities);
- storitve (angl. Services);
- prejemniki oddajanja (Broadcast Receivers);
- ponudniki vsebine (Content Provides);

2.4.1 Aktivnosti

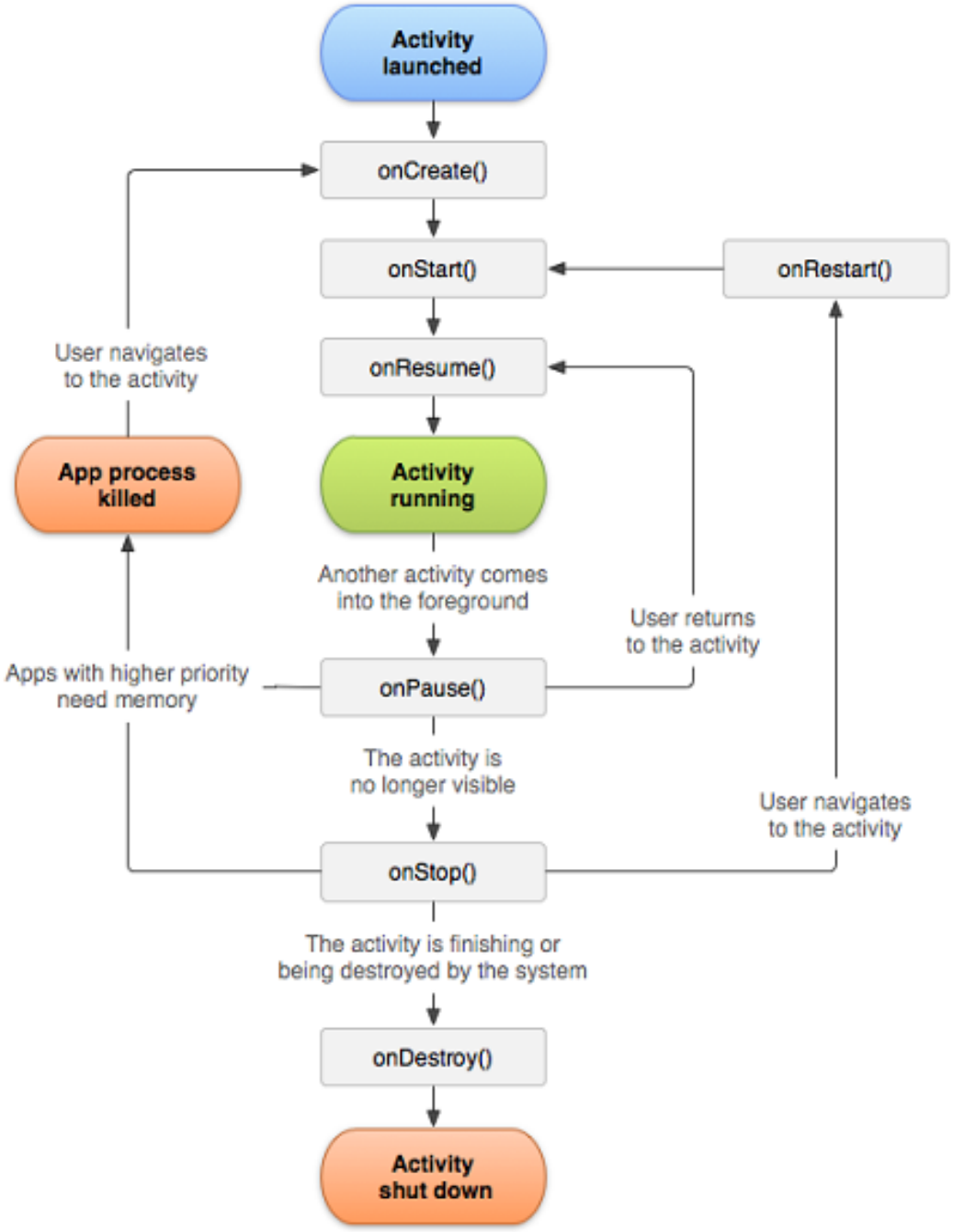
Aktivnost je edina vizuelna komponenta aplikacije. Ko jo uporabnik požene v resnici požene kar aktivnosti, ki med drugim vsebuje tudi uporabniški vmesnik. V tej aktivnosti imamo lahko npr. seznam novic z naslovom in kratkim povzetkom vsebine. Za podrobnejši opis novice se uporabnik dotakne posamezne novice v seznamu, kar ga preusmeri na novo aktivnost, ki je namenjena prikazu ene same novice z vsemi podrobnostmi. Ob tem je pomembno vedeti, da ima vsaka aktivnost svoj življenski cikel. Ob zagonu aplikacije se aktivnost kreira in tako prične svojo življensko pot. Takoj, ko jo uporabnik na zaslonu zapusti, pa se njeno življenje postopoma konča. Ni pa nujno, da se v vsakem primeru popolnoma razgradi. Če jo naprimer le deloma prekrije druga, se prva le začasno zaustavi, potisne na sklad in njeno izvajanje nadaljuje takoj, ko se druga razgradi. Za programerja je izredno pomembno dejstvo, da nad stanjem aktivnosti nima direktne kontrole, pač pa izkorišča prehode iz enega stanja v drugega. Ob vsakem prehodu Android namreč sam sproži klic določenih metod, v katere implementiramo želeno funkcionalnost. Življenski cikel aktivnosti prikazuje slika 2.3.

2.4.2 Storitve

Storitve so namenjene opravilom, ki se izvajajo v ozadju in lahko trajajo dlje časa. Najpogosteje storitve uporabljajo predvajalniki glasbe. Takšne aplikacije predvajanje prepustijo storitvi kar uporabnikom omogoča nemoteno interakcijo z uporabniškim vmesnikom in prehode med aktivnostmi ob sočasnem predvajanju glasbe.

2.4.3 Prejemniki oddajanja

Prejemniki oddajanja se odzivajo na prihajajoča sporočila in spremembe stanj v mobilni napravi. Primer je npr. prejem obvestila o spremembi lokacije s strani GPS naprave ali pa zatemnitev ekrana. Podobno lahko sprejemajo



Slika 2.3: Življenski cikel aktivnosti.

sporočila drugih aplikacij, npr. ko želijo da se neka vsebina osveži, prejme SMS sporočilo itd.

2.4.4 Ponudniki vsebine

Ponudniki vsebine so namenjeni deljenju podatkov z drugimi aplikacijami. Za primer vzemimo telefonski imenik oseb in podjetij. En razvijalec bi lahko razvil aplikacijo namenjeno fizičnim osebam. Temu bi prilagodil svoj uporabniški vmesnik, vir podatkov pa bi bil ponudnik vsebine imenik. Spet drugi razvijalec bi lahko razvil drugačno aplikacijo za poslovne uporabnike. Aplikacija bi se v uporabniškem vmesniku razlikovala, podatke pa črpala iz istega vira, imenika.

2.5 Google Cloud Messaging

Google Cloud Messaging (GCM) je storitev, ki omogoča pošiljanje sporočil uporabnikom naše mobilne aplikacije. Sporočila so lahko velika le 4 KB, kar pomeni da gre pravzaprav za sporočila, ki predstavljajo drezljaj aplikaciji. Le ta se nanj odzove in opravi neko nalogo. Največkrat se uporablja za sprožitev osveževanja podatkov, npr. e-pošte, novic in podobno.

GCM deluje na platformi Android 2.2 in novejših. Za pošiljanje uporablja PUSH (potisno) tehnologijo, ki deluje tako, da odjemalcu ni potrebno zahtevati podatkov ob vnaprej določenem intervalu, kot je to npr. pri elektronski pošti. Strežnik ob prejemu sporočila namenjenemu odjemalcu tega sam pošlje, (potisne) proti uporabniku.

Za uporabo GCM storitve potrebujemo:

- lasten aplikacijski strežnik
- Googlov GCM strežnik
- odjemalca (napravo z Androidom)

Običajno je naloga aplikacijskega strežnika ta, da ponuja uporabnikom neko storitev. Vzemimo za primer, da se na strežniku izvaja spletna aplikacija za komunikacijo med uporabniki. Ko prvi uporabnik drugemu pošlje sporočilo, želi aplikacijski strežnik prejemnika obvestiti, da ga čaka novo sporočilo. To stori tako, da GCM strežniku pošlje obvestilo o novem sporočilu skupaj s podatki prejemnika. GCM strežnik to sporočilo posreduje naprej odjemalcu, aplikaciji na mobilni napravi. Če je sporočilo dovolj majhno je lahko kar vsebovano v obvestilu, sicer pa obvestilo služi le kot opomnik odjemalcu, da ga na na aplikacijskem strežniku čaka novo sporočilo. Še predno lahko odjemalec prejema obvestila od GCM strežnika se mora registrirati. Registracijo opravi kar odjemalec, to je aplikacija na Androidni napravi.

2.6 Java

Java je objektni programski jezik, katerega uporabljamo za razvoj namiznih, poslovnih in drugih aplikacij. Je tudi priporočen programski jezik za razvoj Androidnih aplikacij.

2.7 PHP

PHP je skriptni jezik, ki se izvaja na strežniku. Z njim je možno zgraditi zelo zmogljive, dinamične spletne aplikacije. Mi smo ga uporabili za razvoj strežniškega dela naše aplikacije, ki bere novice z virov RSS ter nato shranjene novice streže odjemalcem.

2.8 Apache

Apache je spletni strežnik, ki odjemalcem streže spletne strani. Uporabili smo ga v strežniškem delu aplikacije.

2.9 MySQL

Za shranjevanje podatkov se uporablja relacijska podatkovna baza MySQL.

2.10 Android Development Tools

Za čimbolj enostaven razvoj Androidnih aplikacij je Google poskrbel s programsko zbirko razvojnih orodij. Ta vsebuje:

- Java Software Development Kit
- Eclipse IDE
- Android SDK
- Android Development Tools Plugin for Eclipse

Eclipse je zelo zmogljivo integrirano okolje in razvijalcu med drugim omogoča pisanje izvorne kode, gradnjo uporabniškega vmesnika ter tudi testiranje aplikacije. Gradnja uporabniškega vmesnika je še posebej enostavna saj omogoča vizualno gradnjo po sistemu povleci/spusti (angl. drag and drop). Tako lahko gradnike kar povlečemo direktno na sliko zaslona, ki prikazuje kako bo uporabniški vmesnik izgledal.

Za integracijo Androidnih orodij v Eclipse poskrbi vtičnik ADT.

2.10.1 phpMyAdmin

Za pregledovanje in urejanje baze podatkov MySQL na strežniku, smo uporabili pripročni programski paket phpMyAdmin.

2.10.2 SQLite Manager

SQLite Manager je razširitev za spletni brskalnik Firefox in je namenjen pregledu in upravljanju z bazo podatkov SQLite.

Poglavje 3

Aplikacija za spremljanje novic

Slika 3.1: Aplikacija Aktualne Novice v emulatorju.

Aktualne Novice je mobilna aplikacija razvita za operacijski sistem Android. Uporabniku olajša spremljanje aktualnih novic, vremenske napovedi in prome-

tnih informacij z večih spletnih virov. Glavna prednost je prav v mobilnosti, torej dosegljivosti storitve spremljanja novic na vsaki lokaciji ob vsakem času. Mobilna naprava uporabnika pa za delovanje potrebuje dostop do spleta, saj aplikacija kot odjemalec novice pridobiva s strežnika in jih shrani v podatkovno bazo SQLite.

3.1 Analiza

Naša želja je bila izdelati aplikacijo, ki bi bila uporabnikom na voljo kjerkoli in kadarkoli. En najbolj razširjenih opracijskih sistemov na mobilnih napravah je prav Android, zato smo se odličili razvijati zanj. Sicer pa je strežniški del razvit tako, da bi brez večjih težav odejamec razvili tudi za druge platforme, npr IOS.

Da bi bila storitev spremljanja novic koristna čim večjemu krogu uporabnikov, smo na strežniški strani skupaj na eno mesto zbrali novice več spletnih novičarskih portalov novice pa tudi ustrezno kategorizirali. Uporabnik s tem dobi možnost prilagoditi nastavitve aplikacije po svoji meri. Spremlja lahko le izbrane vire novic, ali pa vse, lahko pa se odloči tudi za omejitev prikaza, filtriranje novic določene kategorije (npr. le šport, tehnika).
Da so novice in druge informacije kar se da sveže, pa smo uporabili Googlovo storitev v oblaku (Google Cloud Messaging - GCM), ki omogoča prenos novih novic s strežnika takoj ko so te na voljo. Uporabnik pa lahko interval prilagodi po svoji želji.

Ker so novice na strežniku predstavljene v formatu XML, smo v odjemalec vgradili razčlenjevalnik XML, ki razčleni posamezne dele novice (naslov, vsebina, fotografija, povezave) in jo shrani v lokalno podatkovno bazo SQLite.

3.1.1 Diagram primerov uporabe

Za lažje razumevanje funkcionalnosti aplikacije so le te prikazane v diagramu primerov uporabe 3.2. Akter je v našem primeru uporabnik aplikacije, ki na

zaslonu svoje mobilne naprave izvaja zaporedje akcij, ki ga pripelje do nekega rezultata, npr. seznama novic, podrobnosti ene novice in podobno.

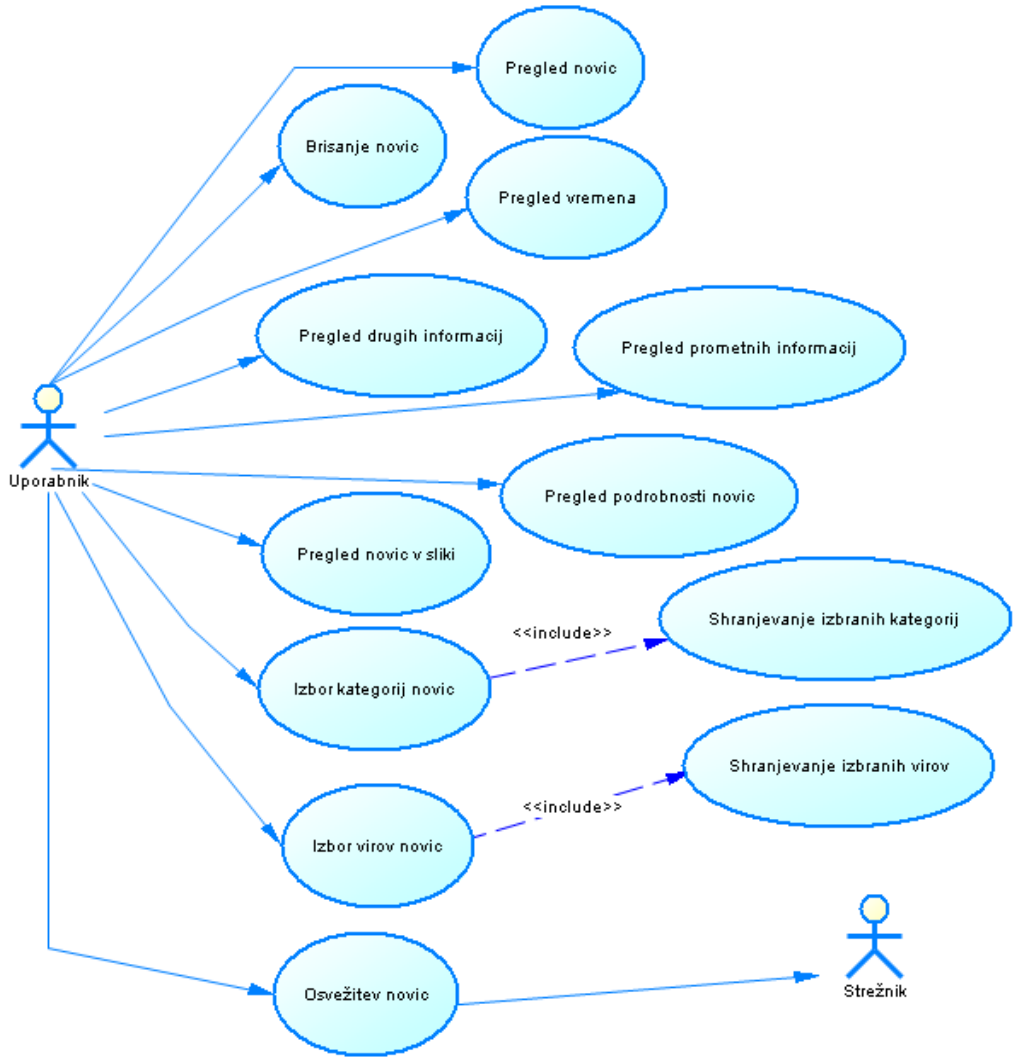
Uporabniku se s klikom na ikono prikaže seznam povzetkov novic, vremenska napoved ali prometne informacije. Z dotikom povzetka novice se prikažejo podrobnosti novice. Posamezno novico ali pa vse naenkrat lahko uporabnik izbriše. V vsakem trenuku lahko s strežnika prenese nove novice, ki jih še ni prebral. Prikaz novic lahko prilagodi z izbiro le želenih kategorij ali virov novic. Vse spremembe se shranijo. Ob vklopu filtra se prikažejo le izbrane kategorije ali viri novic. Če je filter izklopljen se prikažejo vse. Uporabnik lahko vklopi avtomatično osveževanje novic in ob tem nastavi časovni interval osveževanja.

3.2 Načrtovanje in razvoj

3.2.1 Načrtovanje podatkovne baze

Naša aplikacija, po prenosu novic s strežnika, le te skupaj z drugimi podatki, shrani v mobilno napravo. V ta namen uporablja vgrajeno bazo podatkov SQLite. Entitete, ki smo jih izdelali so:

- Kategorija
- Novica
- Nastavitev
- Vir
- Promet
- Vreme



Slika 3.2: Diagram primerov uporabe.

Kategorija

Hrani kategorije novic, ki jih odjemalcu nudi strežnik. Primeri: Slovenija, svet, gospodarstvo, šport,..

- catid - enolični identifikator kategorije
- prioriteta - pomembnost kategorije

- kategorija - naziv kategorije
- vidnost - vidnost kategorije

Novica

Hrani vse podatke o novicah kot so naslov, povzetek, vir in drugo.

- nid - enolični identifikator
- kategorija - kategorija novice (tuji ključ)
- naslov - naslov novice
- povzetek - povzetek novice
- novica - besedilo novice
- objava - datum in čas objave
- povezava - povezava do podrobnosti
- slika - povezava do pripadajoče fotografije
- vir - vir, ki je objavil novico (tuji ključ)

Nastavitev

Hrani osnovne nastavitve, prilagoditve aplikacije potrebi uporabnika.

- nid - enolični identifikator
- nastavitev - naziv nastavitve
- vrednost - vrednost nastavitve

Vir

Hrani vire novic, ki so dostopni na strežniku.

- idv - enolični identifikator
- prioriteta - pomembnost vira za uporabnika
- vir - naziv vira
- vidnost - vidnost vira

Promet
Hrani podatke o stanju v cestnem, železniškem in letalskem prometu.

- pid - enolični identifikator
- vsebina - vsebina prometnih informacij

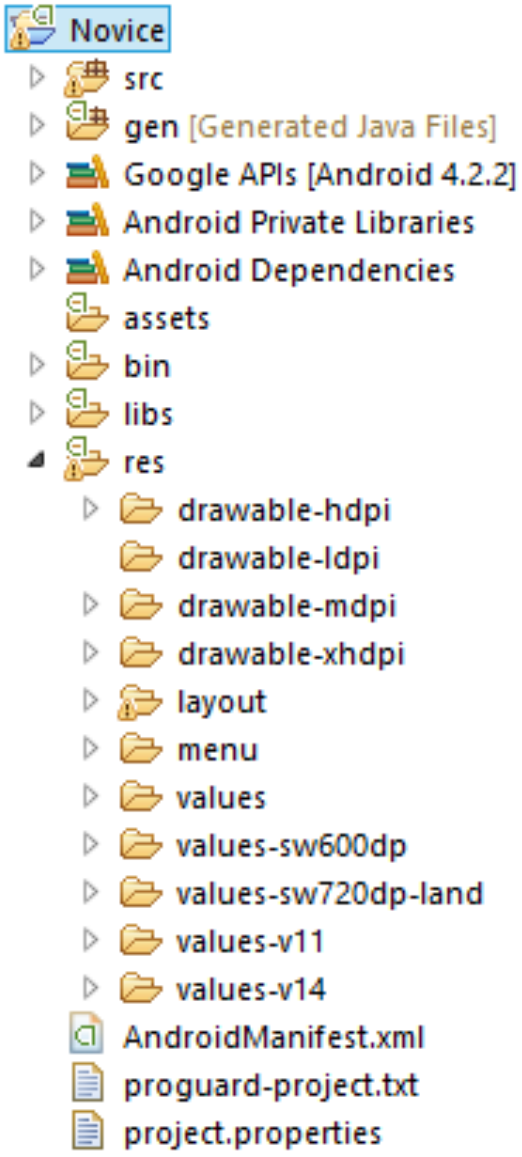
Vreme
Hrani podatke vremenskih razmerah.

- pid - enolični identifikator
- vsebina - vsebina, vremenski podatki

3.2.2 Struktura projekta

Ob kreiranju projekta, to je nove aplikacije, Android zahteva določeno strukturo map in datotek. To sistemu omogoča, da črpa posamezne vire kot so besedila, slike, ikone in druge datoteke, ki jih uporablja aplikacija. Razvijalcu je v veliko pomoč razvojno orodje Android Development Tools, natančneje Eclipse, ki osnovno strukturo postavi kar sam. Vgrajen vtičnik skupaj z razvojnimi orodji Android SDK poskrbi za avtomatično generiranje določenih datotek potrebnih za normalno delovanje aplikacije. Struktura projekta je vidna na sliki 3.3.

Najpomembnejša datoteka, ki je prisotna v vsaki aplikaciji je AndroidManifest.xml. Poleg zapisa enoličnega naziva paketa, opisuje tudi vse bistvene



Slika 3.3: Struktura projekta Android aplikacije.

komponente aplikacije, sistem pa jo prebere že ob namestitvi in kasneje pred zagonom. V tej datoteki so shranjena tudi dovoljenja, ki jih mora uporabnik sprejeti za pravilno delovanje aplikacije. Vso Javansko izvorno kodo smo shranili v mapo src, grafične komponente, torej ikone ter slike pa sistem pričakuje

v mapah drawable znotraj mape res, ki predstavlja vire (angl. resources). Datoteke, ki opisujejo uporabniški vmesnik smo vstavili v mapo layout, vrednosti nizov, velikosti črk pa v mapo values.

3.2.3 Dovoljenja

Naša aplikacija za delovanje potrebuje dostop v internet, zato zahteva ustrezno dovoljenje:

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

Nekoliko več dovoljenj zahteva storitev GCM s pomočjo katere aplikacija osvežuje seznam novic v svoji podatkovni bazi. Za različice starejše od Android 4.0.4, storitev Google Cloud Messaging potrebuje dostop do računa Google na mobilni napravi:

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

Da ob prememu sporočila procesor ne preide v stanje spanja zagotavlja:

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

Za prejem sporočil se mora aplikacija storitvi GCM registrirati, za registraciji in prejemanje sporočil pa je potrebno dovoljenje:

```
<uses-permission
android:name="com.google.android.c2dm.permission.RECEIVE" />
```

Registracijo in prejem sporočil dovolimo le naši aplikaciji:

```
 <permission
 android:name="si.wrs.android.frinovice.permission.C2D_MESSAGE"
```

```
 android:protectionLevel="signature" />

<uses-permission
android:name="si.wrs.android.frinovice.permission.C2D_MESSAGE" />
```

3.2.4 Uporabniški vmesnik

Grafični uporabniški vmesnik (UV) lahko gradimo na dva načina. Priporočen način je deklaracija grafičnih komponent z uporabo formata XML. Te datoteke nato shranimo v mapo layout. Drugi način gradnje UV pa je direktna implementacija v izvorni kodi s programskim jezikom Java. Ta način ni priporočljiv prav zaradi poskusa ločitve izvorne kode of deklaracije grafičnih komponent. Kljub temu se večinoma uporablja kar kombinacija obeh načinov. Začetno stanje UV deklariramo v formatu XML, kasnejše spremembe pa kar v izvorni kodi. Gradniki UV razširjajo razreda ViewGroup in View pri čemer so objekti View posamezni gradniki kot npr. vnosno polje, gumb, gradnik besedila. Objekti ViewGroup pa predstavljajo razporeditvene gradnike, ki jih na zaslonu aplikacije ne vidimo, saj je njihova naloga le ustrezno razporediti posamezne gradnike uporabniškega vmesnika. V naši aplikaciji smo uporabili naslednje razporeditvene gradnike, ki razširjajo razred ViewGroup:

- LinearLayout
- RelativeLayout
- GridLayout

Opis uporabniškega vmesnika

Glavna aktivnost aplikacije vsebuje tudi uporabniški vmesnik, ki se uporabniku prikaže na zaslonu takoj po zagonu aplikacije. Uporabniški vmesnik sestavljajo naslednji gradniki, ki razširjajo razred View in so vsebovani v razporeditvenih gradnikih naštetih zgoraj. Na vrhu zaslona smo horizontalno postavili pet

gradnikov tipa ImageView s prikazom ikon za novice, vreme, promet, slike in drugo. Pod njimi je vrstica z gradnikom TextView, ki z besedo nakaže v katerem delu aplikacije se uporabnik trenutno nahaja. Pod to vrstico sledi še več gradnikov TextView z napisi pozameznih kategorij (npr. Slovenija, gospodarstvo, šport,...). Ti gradniki se nahajajo v HorizontalScrollView, ki omogoča, da lahko kategorije pomikamo horizontalno. Sredino zaslona pokriva gradnik ListView, ki prikazuje povzetke novic skupaj z manjšo sliko. Na spodnji rob zaslona smo postavili še nekaj gradnikov ImageView, ki izvajajo različne funkcije (npr. brisanje novic, osveževanje, filtriranje novic,..). Uporabniški vmesnik prikazuje slika 3.1 na začetku poglavja.

3.2.5 Komponente aplikacije

Našo aplikacijo sestavljajo tri glavne komponente aktivnosti, storitev ter prejemnik oddajanja. V Androidu je potrebno vsako od teh komponent predstaviti sistemu tako, da jih vpišemo v datoteko AndroidManifest.xml.

Aktivnosti:

- glavna aktivnost skrbi za prikaz seznama novic in komunikacijo s servisom
- aktivnost kategorij omogoča izbiro kategorij novic
- aktivnost promet prikaže prometne informacije
- aktivnost vreme prikaže vremenske podatke
- aktivnost slike prikaže novice v sliki
- aktivnost osveževanja omogoča vklop/izklop ter nastavitev avtomatičnega osveževanja

Storitev:

Aplikacija vsebuje eno storitev, ki razširja razred Service. Omogoča osveževanje novic v ozadju in pri tem ne moti uporabnika v času interakcije z drugimi gradniki uporabniškega vmesnika. V ozadju pomeni, da je opravilo uporabniku nevidno, ne pomeni pa da se storitev izvaja v ločenem procesu ali niti. Storitev poskrbi za prenos novice, razčlenitev v formatu XML zapisanih podatkov o novici ter shranjevanje v lokalno bazo podatkov. Storitev je aktivna dokler je pripeta kakšni od aktivnosti oz. dokler jo ne ustavimo z s klicem stopself(), če je bila pognana z metodo startService().

Prejemnik oddajanja:

Prejemnik razširja razred BroadcastReceiver in je namenjen prejemanju sporočil storitve Google Cloud Messaging. Ko želi naš strežniški del aplikacije obvestiti odjemalca, da so na voljo nove novice o tem obvesti strežnik storitve GCM. Slednji odjemalcu pošlje t.i. push sporočilo, katerega sprejme prejemnik oddajanja. Ob tem se požene storitev, ki opravi vse naloge osveževanja novic.

Naš prejemnik smo vpisali v AndroidManifest.xml in zapisali da želi prejemati sporočila storitve GCM:

```
<receiver
    android:name=".GCMBroadcastReceiver"
    android:permission="com.google.android.c2dm.permission.SEND" >
    <intent-filter>
<action android:name="com.google.android.c2dm.intent.RECEIVE" />
<category android:name="ime.paketa.aplikacije" />
    </intent-filter>
</receiver>
```

Prva vrstica vsebuje naziv našega prejemnika oddajanja. Z vrstico spodaj zahtevamo, da le Googlov strežnik lahko prejemniku pošilja sporočila. Sledita še vrstici s filtrom s katerim omogočimo dejansko branje prejetih sporočil.

3.2.6 Aktivacija in prehodi med komponentami

V vseh aplikacijah, kjer obstaja več kot ena glavna komponenta (npr. dve ali več aktivnosti) se pojavi potreba po prehodu iz ene v drugo aktivnost. Podobno lahko v aplikaciji želimo npr. aktivirati storitev. To naredimo z uporabo sporočil, ki jim pravimo namen (angl. Intent). V naši aplikaciji jih imamo kar nekaj. V glavni aktivnosti, to je tista, ki jo uporabnik najprej vidi na zaslonu, smo na vrh postavili gradnike ImageView. To so ikone, ki omogočajo ogled novic, vremena, prometa in drugih informacij. Prehod oz. aktivacijo druge aktivnosti sproži explicitni namen:

```
Intent intent = new Intent(this,ImeAktivnosti.class);
startActivity(intent);
```

Eksplicitni namen je takšen, ki določa točno katera aktivnost naj se požene.

Uporabili pa smo tudi impliciten namen v primeru, ko smo želeli, da želi uporabnik prebrati celotno novico na spletni strani vira.

```
Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(link));
startActivity(intent);
```

V tem primeru gre za impliciten namen, ki ne določa točno katera aplikacija bo odprla spletno stran z novico. Če ima uporabnik nameščenih več brskalnikov bo lahko izbral kateri naj se uporabi. Pri tem smo kreirali nov objekt razreda Intent, mu kot parameter podali ime ciljne aktivnosti ter poklicali metodo startActivity, ki novo aktivnost požene. Uporabniku se po omenjenem klicu na zaslonu prikaže nova aktivnost s svojim uporabniškim vmesnikom in svojo vsebino. Namesto metode startActivity() lahko uporabimo tudi startActivityForResult(). Pri tem druga aktivnost prvi posreduje nek zahtevan rezultat.

Aktiviramo lahko tudi storitev. Pri tem imamo dve možnosti, v naši aplikaciji smo uporabili obe. Storitev, ki smo jo implementirali skrbi za osveževanje

novic. Nove novice prebere s strežnika in jih trajno shrani v bazo podatkov. Ker aplikacija omogoča tako ročno kot avtomatično osveževanje smo za aktivacijo storitve za prvi način uporabili bindService(), drugi avtomatični del pa aktivira storitev z metodo startService(). Avtomatično osveževanje sproži naš strežnik, ki zahtevo pošlje storitvi GCM ta pa uporabnikovi mobilni napravi. Zahtevo sprejme naš prejemnik oddajanja, ki takoj požene storitev:

```
Intent intent = new Intent(context, StoritevOsvezevanja.class);
context.startService(intent);
```

3.2.7 Osveževanje vsebin s pomočjo storitve GCM

Za uporabo storitve Google Cloud Messaging je potrebnih kar nekaj korakov:
- v prvem koraku potrebujemo Googlov račun za dostop do Google API nadzorne plošče, kjer storitev GCM aktiviramo
- kreiramo nov projekt, številko projekta pa uporabimo za identifikacijo pošiljatelja (GCM sender ID)
- kreiramo API ključ, ki ga bo uprabljal aplikacijski strežnik

Registracija odjemalca

Za pošiljanje GCM sporočil potrebujemo registracijsko številko. Ta številka identificira mobilno napravo, dobimo pa jo s klicem metode register() objekta GoogleCloudMessaging:

```
GoogleCloudMessaging gcm =
GoogleCloudMessaging.getInstance(context);
String registrationId = gcm.register(senderID);
```

SenderID je številka projekta, ki smo ga kreirali v nadzorni plošči Google API. GCM strežnik nam ob uspešni registraciji vrne niz znakov, ki predstavljajo registracijsko številko.
Ker bo sporočila pošiljal naš aplikacijski strežnik, moramo registracijsko številko

posredovati in shraniti v strežniku. Slednji jo uporabi vsakič, ko želi napravi poslati sporočilo.

Ob prejemu sporočila sistem aktivira prejemnika oddajanja in pokliče metodo onReceive(). V tej metodi obdelamo vsa prejeta sporočila. Prejeto sporočilo je lahko regularno sporočilo, ki ga GCM deklarira kot tip MESSAGE_TYPE_MESSAGE, lahko pa je tudi obvestilo o napaki.

```
GoogleCloudMessaging gcm =
GoogleCloudMessaging.getInstance(context);
String msgType = gcm.getMessageType(intent);
if (msgType.equals(GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE))
  // obdelava regularnega sporočila in sprožitev osveževanja
```

3.3 Testiranje

V pomoč pri testiranju aplikacije smo uporabili emulator, orodje, ki je del razvojnih orodij za platformo Android. Emulator je navidezna naprava, ki poskuša čimbolj natančno simulirati fizično mobilno napravo. Izkaže se, da v veliki meri uspešno opravlja svojo nalogo in razvijalca pravilno opozarja na morebitne pomanjkljivosti. Vendar proizvajalci mobilnih naprav, zaradi odprtosti platforme, prilagajajo Android svojim potrebam, kar v določenih primerih onemogoča dobro, pravilno simulacijo teh naprav. Tako še vedno velja, da moramo za 100% kompatibilnost aplikacije le to preizkusiti kar na fizičnih mobilnih napravah.

Aplikacijo smo testirali na treh fizičnih mobilnih napravah s sicer že starejšimi različicami operacijskega sistema, to je Android 2.2, 2.3 in 3.2. Na omenjenih napravah je aplikacija delovala stabilno.

Poglavje 4

Strežnik novic

Pomembno vlogo pri naši nalogi ima tudi strežniški del. Strežnik konstantno pregleduje zgoščene povzetke RSS spletnih novičarskih portalov, iz njih izlušči dele novice (naslov, besedilo novice, kategorija, povezava do podrobnosti,...) in jih shrani v podatkovno bazo MySQL. Naslovi RSS virov so shranjeni v podatkovni bazi. Dodamo lahko nove naslove virov ter spremenimo ali zbrišemo obstoječe. Izkaže se, da spletni portali pogosto spreminjajo spletne naslove do povzetkov RSS, zato je ta funkcionalnost zelo dobrodošla na strežniku. Uporabnikom Androidne aplikacije namreč zato ni potrebno nadgrajevati aplikacij zaradi morebitnih sprememb virov.

Primer novice zapisane v formatu XML enega od spletnih portalov:

```
<item>
  <title>Ruska raketa takoj po izstrelitvi strmoglavila</title>
  <link>http://24ur.com/novice/svet/ruska-raketa.html</link>
  <description><![CDATA[ <img src="http://images.24ur.com/265.jpg" />
  V Kazahstanu je takoj po izstrelitvi strmoglavila ruska raketa
  s tremi navigacijskimi sateliti.]]></description>
  <pubDate>Tue, 02 Jul 2013 06:54:35 +0200</pubDate>
  <category>Svet</category>
</item>
```

Vse novice v formatu XML strežnik s pomočjo skriptnega jezika PHP razčleni. Uporabili smo funkcijo simplexml_load_file(), ki je del PHP razširitve SimpleXML. Slednji na zelo enostaven način omogoča razčlenjevanje XML dokumenta.

Novice strežnik posreduje v XML obliki kar omogoča kompatibilnost praktično na vseh operacijskih sistemih. Tako lahko razvijalci razvijejo aplikacijo za npr. operacijski sistem IOS ali pa Windows in prav tako uporabljajo to storitev.

Za odjemalce hrani tudi podatke o vseh virih novic, ki so na voljo pa tudi kategorije novic v katere so slednje razvrščene. Strežnik hrani tudi podatke o že prenešenih novicah s čimer se izognemo nepotrebnemu ponovnemu prenosu istih novic k odjemalcu. V kolikor to uporabnik želi, lahko strežniku preko odjemalca sporoči tudi interval osveževanja novic. Pri tem bo strežnik ob izbranih intervalih preveril, če za določenega uporabnika obstajajo nove novice in jih posredoval aplikaciji le v primeru da te obstajajo.

Poglavje 5

Zaključek

V sklopu diplomske naloge smo razvili aplikacijo za spremljanje aktualnih novic, vremenske napovedi in prometnih informacij za operacijski sistem Android. Podatke streže strežniški del aplikacije, ki je izdelan tako, da so odjemalci lahko tudi naprave z drugimi operacijskimi sistemi. Tako bi lahko podobno aplikacijo izdelali za mobilne naprave IOS ali pa npr. namizne računalnike z operacijskim sistemom Windows.

Naša aplikacija za Android je razvita za različice 2.2 in novejše pri čemer smo se omejili na manjše naprave z ločljivostjo 480x800 pik. Aplikacija na napravah z večjo ločljivostjo sicer deluje, vendar je potrebno dograditi uporabniški vmesnik. Na napravah z Android 2.1 in starejšimi storitev Google Cloud Messaging ni podprta za to je možna le ročna osvežitev novic. Možna pa je nadgradnja, ki bi omogočala pull način osveževanja podatkov.

Uporabniški vmesnik smo razvili tako, da je pregleden in enostaven za uporabo, saj je vsa funkcionalnost vidna že na prvem zaslonu. Zaradi velikega števila različnih naprav smo se morali omejiti le na naprave, kjer smo aplikacijo lahko tudi fizično preizkusili. Pri samem razvoju je bila najbolj zahtevna implementacija seznama novic, ki se prikaže uporabniku takoj po zagonu aplikacije. Uporabljen gradnik ListView v osnovi vsebuje le TextView gradnike, sami pa smo morali izdelati povsem svoj kombiniran gradnik. V eni sami ce-

lici seznama imamo namreč več gradnikov za prikaz slike ob novici, naslov novic, povzetek ter za podatke o viru in času objave. Pravi izziv je predstavljalo tudi drsenje po novicah, ki se je zelo rado zatikalo, saj je sistem moral vnaprej pripraviti vse podatke skupaj s prenosom slike. Pri velikem številu elementov seznama Android v pomoč ponuja tako imenovani adapter, ki poskrbi za pravočasno, sprotno pripravo potrebnih podatkov brez nepotrebnega dodatnega prenašanja podatkov, ki jih ne potrebujemo.

V prihodnjih nadgradnjah bi bilo koristno implementirati funkcijo iskanja po novicah, ki bi bila pravzaprav filtriranje novic glede na vpisane ključne besede. Tako bi lahko uporabniki poleg virov in kategorij zožili izbor novic še po vsebovanih ključnih besedah. Podobno koristna funkcija bi bila tudi shranjevanje fotografij za pregledovanje v času ko povezava v internet ni mogoča.

Slike

2.1 Razširjenost različic platforme Android. 5
2.2 Arhitektura platforme Android. 6
2.3 Življenski cikel aktivnosti. 9

3.1 Aplikacija Aktualne Novice v emulatorju. 13
3.2 Diagram primerov uporabe. 16
3.3 Struktura projekta Aktualne Novice. 19

Tabele

2.1 Različice platforme Android. 4

Literatura

[1] M.L. Murphy. *The Busy Coder's Guide to Android Development.* CommonsWare LLC, 2008-2011.

[2] R. Rogers, J. Lombardo, Z. Mednieks, B. Meike. *Android Application Development.* O'Reilly, 2009, str 1-13.

[3] (2013) Android Developers.
Dostopno na: http://developer.android.com

[4] (2013) Android Developers Dashboards.
Dostopno na: http://developer.android.com/about/dashboards/index.html

[5] (2013) PHP.
Dostopno na: http://www.php.net/manual/en/

[6] (2013) Google Cloud Messaging.
Dostopno na: http://developer.android.com/google/gcm/index.html