Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Anže Marinko

Učenje klavirja za najmlajše
ANDROID APLIKACIJA

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana 2013

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

[1] V dogovorju z mentorjem lahko kandidat diplomsko delo s pripadajočo izvorno kodo izda tudi pod katero izmed alternativnih licenc, ki ponuja določen del pravic vsem: npr. Creative Commons, GNU GPL. V tem primeru na to mesto vstavite opis licence, na primer tekst [**?**]

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: 00415 / 2013
Datum: 5.4.2013

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

Kandidat: **ANŽE MARINKO**

Naslov: **UČENJE KLAVIRJA ZA OTROKE NA PLATFORMI ANDROID**

LEARNING PIANO FOR CHILDREN ON ANDROID PLATFORM

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

V okviru diplomske naloge preučite stanje na področju izobraževalnih aplikacij za učenje klavirja na mobilnih platformah. Izdelajte lastno aplikacijo, ki bo na privlačen način otroka vodila skozi proces učenja igranja enostavnih melodij.

Mentor:

doc. dr. Matija Marolt

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Anže Marinko, z vpisno številko **63090264**, sem avtor diplomskega dela z naslovom:

Učenje klavirja za najmlajše

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Matije Marolta,
- 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 30. avgust 2013 Podpis avtorja:

Zahvalil bi se svoji družini in dekletu, ki so me spodbujali in podpirali med študijem. Posebna zahvala gre tudi mentorju doc. dr. Matija Marolt, ki mi je pomagal z nasveti in me usmerjal pri samem delu.

Posvečeno družini

Kazalo

Povzetek

Diplomsko delo opisuje načrtovanje in implementacijo Android aplikacije, katera bi mlajše otroke lahko naučila igranja klavirja na zabaven način. Sama aplikacija naj bi spodbujala otroke k igranju, poleg tega pa bi se skozi igro tudi učili. Veliko aplikacij je narejenih za otroke, a le malo je takih, kateri bi otroke skozi igro tudi učili.

V začetku diplomskega dela je predstavljena mobilna tehnologija in uporabljena razvojna orodja. Temu sledi načrtovanje aplikacije, kateri je razdeljen na tri področja: ideja, videz in namen same aplikacije. Tu je opisano kako je sama aplikacija nastajala. Načrtovanju sledi opis implementacije same aplikacije. Diploma pa je zaključena z pregledom narejenega in idejami za nadaljnje delo.

Ključne besede:
Android, klavir, aplikacija, učenje

Abstract

Graduation thesis describes the design and implementation of Android apps, which would younger children can learn to play the piano in a fun way. Application itself should encourage children to play, in addition to also learn through play. Many applications are made for children, but few of such, which would also be children learn through play.

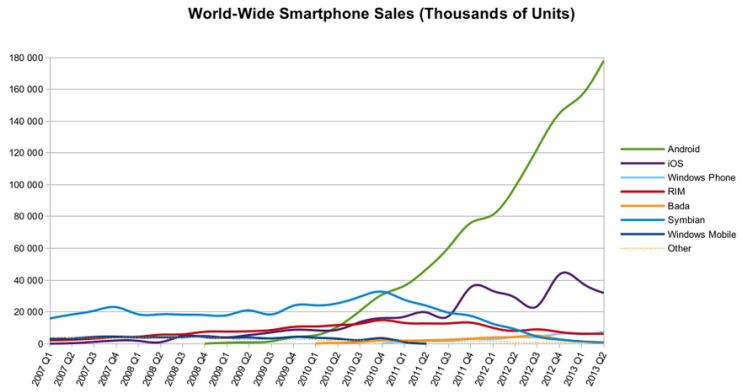
In the early graduation thesis is presented and used mobile technology development tool. This is followed by application design, which is divided into three areas: the idea of the appearance and purpose of the application itself. Here's how the application itself created. Planning followed by a description of the implementation of the application itself. Graduation thesis is completed with the examination done and ideas for future work.

Keywords:
Android, piano, application, learning

Poglavje 1

Uvod

Mobilne tehnologije so v zadnjih nekaj letih zelo napredovale in so postale zelo razširjene med uporabniki, saj ima že skoraj vsak od nas mobilno napravo, katere pa so vse zmogljivejše in se dandanes po določenih karakteristikah primerjajo celo z računalniki. Z povečanjem števila naprav pa se je povečala tudi zmogljivost in zanesljivost aplikacij.



Slika 1.1: Prodaja telefonov z določenim operacijskim sistemovi[3]

Kot lahko razberemo iz grafa, je najbolj uporabljeni mobilni operacijski sistem Android.[3] Android je odprtokoden in povsem brezplačen, kar za uporabnike to pomeni, da jim bodo na voljo cenejše in bolj funkcionalne mobilne naprave ter večje število inovativnih storitev. Proizvajalcem mobilnih

telefonov Android omogoča znižanje stroškov razvoja programske opreme ter hitrejšo realizacijo svojih konceptov, prav tako pa jim ob fleksibilnosti sistema še vedno prinaša možnost krojenja lastne identitete z razvojem samosvojih rešitev.[1]

Večina ljudi, ki so že slišali za Android, zmotno misli, da je Android, tako kot je večina preostalih operacijskih sistemov za pametne telefone, namenjen zgolj zahtevnejšim uporabnikom, ki imajo veliko predhodnega znanja in izkušenj z napravami, kot so namizni in prenosni računalniki, mobilni telefoni ter drugimi sodobnimi stvaritvami, ki zahtevajo upravljanje prek interaktivnih sprogramiranih vmesnikov. A to ne drži. Android je namenjen različnim uporabnikom ne glede na spol, starost in predznanje. Je enostaven in prilagodljiv, a hkrati kompleksen in zmogljiv.

Pri aplikaciji sta uporabljena programska jezika Java in XML, za razvojno orodje sem uporabil Eclipse s potrebnimi dodatki za izdelavo aplikacij za operacijski sistem Android. Eclipse je razvojno okolje, namenjeno različnim programskim jezikom, njegov naravni jezik pa je Java.[2] Če pa želimo delati Android aplikacije moramo uporabiti tudi Android SDK, ki nam omogoča namestitev testne aplikacije. Za testiranje aplikacije je bil uporabljen emulator, kateri se prav tako nahaja v Android SDK. Z emulatorjem lahko razvijalec Android aplikacij dobi predstavo, kako bo celotna aplikacija delovala in izgledala.

Za izdelavo aplikacije je bilo potrebno znanje iz mobilnih tehnologij in poznavanje programskega jezika Java. Potrebno se je bilo ukvarjati tudi s samim zvokom, saj je potrebno poznavanje tako glasbe kot tudi tega, kako samo glasbo uporabljati v Android aplikaciji.

Aplikacije, katere so namenjene učenju klavirja, lahko najdemo na spletni strani Google play[9], kjer razvijalci ogjavljajo svoje Android aplikacije. Aplikacije za učenje klavirja so recimo: Pianist HD - Finger Tap Piano[10], Piano Melody Free[11] in Piano Instructor (Lite) [12]. Program Pianist HD - Finger Tap Piano in Piano Instructor (Lite) delujeta na ta način, da si izberemo melodijo, katero se želimo naučiti, potem pa se toni kateri morajo

biti pritisnjeni v nekakšnem zaporedju spuščajo na tipkovnico, uporabnik pa jih mora pritisniti. A zna biti sama aplikacija za mlajše otroke nekoliko prezahtevna, saj je potrebno pritisniti tudi po več tipk na enkrat, uporabnik pa si tudi težje zapomni tone, katere so pritisnjen .Tako si uporabnik težko zapomni zaporedje tonov in če bi želel igrati klavir v prostem načinu, bi to znala biti težave, saj si uporabnik ne bi uspel zapomniti celega zaporedja tonov. Program Piano Melody Free pa obarva tiste tipke, katere je potrebno pritisniti, a je sama hitrost zaporedja prehitra in prevelika količina tonov, tako da lahko tudi tu nastane problem, da se uporabnik ne nauči skladbe dejansko zaigrati na klavir.

Zato je bil naš cilj narediti čim bolj preprosto aplikacijo za učenje klavirja, saj če je prezahtevna, se otrok lahko hitro naveliča, naš namen pa je, da bi se otrok, ko bi se želel sprostiti ali ko bi bil recimo na avtobusu ali v kakšni čakalnici, namesto igranja igric, raje svoj čas uporabil za učenje klavirja. Potrebno je narediti preprost grafični vmesnik, da ne bo preveč zahteven in hkrati samo aplikacijo narediti tako, da se otrok še kaj nauči. Vse zgoraj naštete aplikacije so v angleškem jeziku, aplikacija Igranje klavirja za najmlajše je po vsej verjetnost edina, ki je v slovenskem jeziku in ki ponuja pesmi, katere slovenski otroci poznajo.

Poglavje 2

Razvojna orodja

Za razvoja aplikacije sta bila uporabljena programska jezika Java in XML, razvojno orodje pa Eclipse s potrebnimi dodatki za izdelavo aplikacije za operacijski sistem Android.

2.1 Programski jeziki

2.1.1 Java

Programski jezik Java je objektno usmerjen programski jezik, katerega je razvil James Gosling s sodelavci v podjetju Sun Microsystems. Projekt se je na začetku imenoval Oak, kar v prevodu pomeni hrast, začetki pa segajo v leto 1991. Ko so programerji odkrili, da programski jezik s tem imenom že obstaja, so Oak preimenovali v Javo (ime kave, ki so jo programerji pili). Skodelica kave je zato tudi eden od grafičnih znakov, s katerim ponazarjamo programski jezik Java. Različica Java 1.0 je bila objavljena leta 1996. Javo danes vzdržuje in posodablja podjetje Oracle - Sun Microsystems.

Java je danes eden najpomembnejših sodobnih programskih jezikov, razlogov za to pa je več. Prvi razlog je ta, da je Java preprost jezik, saj je za razliko od C++ očiščen mnogih nepotrebnih elementov. Naslednji razlog je, da je predmetno usmerjen (temu se reče tudi objektno orientiran). To pomeni, da je program razbit na množico modulov, ki se imenujejo objekti

in zato je objektno orientiran pristop moderen pristop za razvoj kompleksnih programskih sistemov. Tretji razlog je ta, da je Java robusten jezik, saj napake v programih ne povzročajo sesutja celotnega sistema tako pogosto, kot je to pri nekaterih drugih programskih jezikih. Prednost Jave je tudi v tem, da je neodvisna od platforme, saj lahko program brez spreminjanja uporabljamo na vseh sistemih. Je pa Java tudi varen programski jezik, ki omogoča pisanje varnih programov na spletu, ker pozna več stopen varnosti. Je pa Java kljub vsem naštetim lastnostim dovolj zmogljiv programski jezik, da z njim lahko naredimo skoraj vse in to na zelo preprost način.[5]

2.1.2 XML

XML (Extensible Markup Language) je programski jezik, ki ga pogosto srečate, če brskate po Internetu. XML je preprost računalniški jezik podoben HTML-ju, ki nam omogoča format za opisovanje strukturiranih podatkov ali arhitektura za prenos podatkov in njihovo izmenjavo med več omrežji. XML spreminja mnogo aspektov računalništva, še posebej na področju komuniciranja aplikacij in strežnikov. Da pa se ga tudi razširiti, saj ima namreč to možnost, da si lahko sami izmislimo imena etiket (angleško TAG). Zelo je uporaben za komunikacije, saj ima zelo preprosto in pregledno zgradbo.

XML je razdeljen na 3 dele:

• podatkovni (vanj shranimo podatke v neki obliki z želenimi etiketami (tag)),

• deklarativni (skrbi za to, da lahko pri dodajanju novih podatkov vidimo kaj kakšna etiketa predstavlja),

• predstavitveni (z njim oblikujemo izpis podatkov).

Razvijalci XML povečujejo vsebino tega jezika in s tem njegovih standardov tehnologije, ki vsebujejo podatke, ki se jih da enostavno preoblikovati in zamenjati v neenakih sistemih.

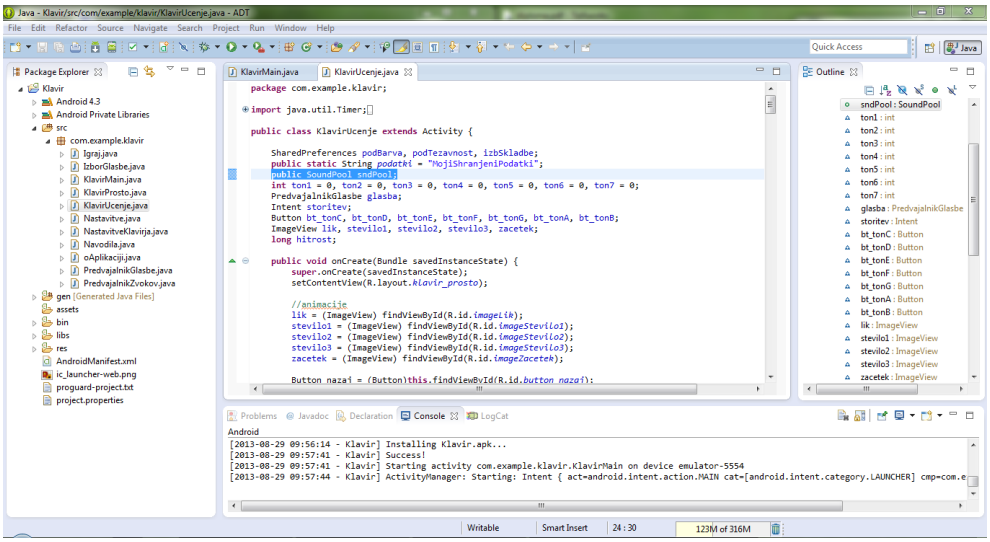
Obstaja več dobrih lastnosti uporabljanja jezika XML:

• XML razdeli podatke za lokalno obdelavo. Podatki so lahko brani v XML obliki, potem pa prenešeni v lokalno aplikacijo, kot je na primer brskalnik za nadaljnje gledanje ali procesiranje. Podatki so lahko preneseni tudi skozi skripto ali druge programske jezike s pomočjo XML objektnega modela.
• Uporabnikom da XML možnost primernega vpogleda v strukturirane podatke. Podatki prenešeni na namizje so lahko predstavljeni v več možnih variantah. Lokalni podatki so lahko predstavljeni na takšen način kot to najbolj ustreza uporabniku.
• Omogoča integracijo strukturiranih podatkov iz več virov v logične in preproste poglede v podatke. Običajno so bili uporabniki navajeni integrirati podatke, iz strežniških baz in ostalih aplikacij na medmrežnih strežnikih, tako da so bili podatki uporabni za pošiljanje na ostale strežnike za nadaljnjo procesiranje, obdelavo in distribucijo.
• Opisuje podatke iz različnih aplikacij. Ker je XML obsežen jezik se lahko uporablja za opisovanje podatkov v široki variaciji aplikacij, od opisovanja kolekcij spletnih strani do podatkovnih zapisov. Ker so podatki samo opisni (self-describing), so lahko sprejeti in procesirani brez potrebe, da so še dodatno opisani.
• Omogoča boljši pretok skozi parcialno granularno popravljanje (granular updates). Izvajalcem ni potrebno poslati celotnih strukturiranih podatkov vsakokrat, ko v njih pride do spremembe. Z granularnimi popravki, se morajo distribuirati samo spremenjeni elementi poslani od strežnika do klienta. Spremenjeni podatki so tako lahko predstavljeni brez ponovnega osveževanja celotne strani ali namizja.[6]

2.2 Razvojno orodje Eclipse

Eclipse je razvojno orodje, namenjeno različnim programskim jezikom, njegov naravni jezik pa je Java. Eclipse ima zelo dobro oblikovano sistematičnost grafičnega vmesnika, saj je le ta sila preprost in pregleden. Če imamo

nameščen osnovno različico, je le ta dovolj za bolj osnovne programe. Če pa želimo delati kaj specifičnega, pa je potrebno namestiti dodatke. Če želimo aplikacijo delati v Javi, je najprej po potrebno namestiti Java SDK (Software Development Kit). V Eclipsu je bistvena uporaba perspektiv. Vsaka perspektiva zase ima svojo množico urejevalnikov in pogledov, katere lahko uporabnik sam nastavlja, glede na to, kaj potrebuje in s čim se ukvarja. [2][8]



Slika 2.1: Program Eclipse

2.3 Razvojno orodje ADT

ADT je na splošno razvojno orodje Eclipse, le da ima že vse potrebno, kar programer potrebuje za izdelavo Android aplikacij. Pri navadnem Eclipsu je potrebno namestiti Android SDK, ki nam omogoča namestitev testne aplikacije in uporabo emulator, kateri se prav tako nahaja v Android SDK, pri razvojnem orodju ADT pa je vse to že vključeno.

Slika 2.2: Izgled emulatorja

2.4 Operacijski sistem Android

Android je odprtokodni programski jezik in operacijski sistem za pametne telefone, ter ostale prenosne naprave. Zgrajen je na Linuxovem jedru. Za razvoj androida je najzaslužnejši Google, ki je ravno v ta namen ustanovil poslovno združenje več podjetij, imenovano Open Handset alliance(OHA), ter pod svoje okrilje vzel hitro rastoče podjetje Android Inc. Poslovno združenje so ustanovili oz. predstavili javnosti 5. novembra 2007, s prizadevanjem skupnega razvoja odprtih standardov na področju telefonije ter ostalih prenosnih naprav, saj poslovno združenje teži k razvoju inovacij na področju mobilne telefonije, ter prenosnih naprav, prav tako pa želi približati te telefone uporabnikom z vedno cenejšimi in boljšimi pametnimi telefoni, ter prenosnimi napravami.[1]

Prednosti uporabe Android platforme:

• ker je Android odprtokoden, omogoča cenejše in lažje razvijanje programov. Občutno prednost tu občutijo tudi uporabniki, saj so programi za ta operacijski sistem večinoma zastonjski

• omogoča cenejše, lažje in hitrejše razvijanje pametnih telefonov (proizvajalcem ni potrebno več razvijati operacijskih sistemov, lahko pa razvijajo posamezne komponente sistema)

• je enostaven, odziven in omogoča večopravilnost

• se samodejno sinhronizira z Googlovimi storitvami

Zgodovina Androida:

Android 1.0

Prva različica operacijskega sistema, ki je prišla na trg s prvimi pametnimi telefoni 9. februarja 2009. Operacijski sistem je vseboval funkcije, kot so budilka, testni prikaz (demo) uporabniškega vmesnika, pregledovalnik za internet, kamero, itd. Ta različica operacijskega sistema je bila zelo okrnjena.

Android 1.5 (Cupcake)

Popravek za različico 1.0. Ta različica operacijskega sistema je sprožila pravo poplavo pametnih telefonov na tržišču in med uporabniki. Poleg različice

1.0 so tej različici dodatno dodali še možnosti dodajanja medijskih datotek neposredno na internet, možnost bluetooth povezave, animacije na ekranu, itd. Operacijski sistem je nastal na platformi Linux Kernel 2.6.27.

Android 1.6 (Donut)

Splavili so ga 15. septembra, 2009. Dodan mu je bil nov, preglednejši, uporabnejši Android Market za prenos programov, skupaj z Open Handset Alliance so pripravili telefon HTC Hero in na njem uspešno zagnali to različico operacijskega sistema. Na ta način so prvič pritegnili pozornost ostalih večjih proizvajalcev mobilnih telefonov. Nastal je na platformi Linux Kernel 2.6.29.

Android 2.0 (Eclair)

Android 2.0 je nastal v rekordnem času od zadnjega popravka, 26. oktobra 2009. Ta različica je bila nekaj novega, saj so jo pričeli izdelovati od samega začetka in ni popravek. Zaradi ponovne izdelave, so se vsem uporabnikom prejšnjih različic operacijskega sistema telefoni nadgradili v celoti. Posledice so bile občutna pohitritev odzivnega časa operacijskega sistema, novi uporabniški vmesniki, bluetooth 2.1 in podobno.

Android 2.2 (Froyo)

Popravek se je sprva pojavil na telefonu HTC Nexus One, sčasoma pa tudi na ostalih pametnih telefonih. Ključne značilnosti popravka so bile nalaganje aplikacij na spominsko kartico telefona, vizualno popravljeni in spremenjeni uporabniški vmesniki, ter spremenjen Android market, ki je sedaj omogočal samodejne posodobitve aplikacij.

Android 2.3 (Gingerbread)

Pojavil se je 6. decembra 2010. Ker so se pri tem popravku večinoma osredotočili na strojno opremo, vsebuje popravek dva nova senzorja (giroskop, barometer). Dodana so tudi orodja za kopiranje in lepljenje datotek. Prav tako so ta popravek razširili tudi za širše ločljivosti zaslonov (XVGA ali večje).

Android 4.0 (Ice cream sandwich)

Na trg je prispel skupaj z telefonom Galaxy Nexus 19. okrobra 2011. Izvorno kodo je Google objavil na spletu 14. novembra 2011. V obilici popravkov

in dodatnih funkcionalnosti sistema izstopajo strojno pospešen grafični vmesnik, prenova grafičnega vmesnika, odklep z prepoznavo obraza, nov spletni brskalnik, izboljšana aplikacija za kamero, itd.[7]

Poglavje 3

Načrt aplikacije

3.1 Ideja

Ideja aplikacije je bila ta, da naredimo preprosto aplikacijo, ki bo uporabniku na nek preprost način omogočila učenje klavirja, bi pa lahko s določenimi nastavitvami samega programa lahko spremenil težavnost do stopnje, katera bi uporabniku odgovarjala. Aplikacija je zelo povezana z glasbo in zvoki, zato je v njej uporabljeni razred SoundPool in razred MediaPlayer. Prvi je uporabljen za zvoke, razred MediaPlayer pa za glasbo.

Sama aplikacija je bila mišljena tako, da uporabnik posluša skladbo, katero se želi naučiti, potem pa jo poskusi zaigrati še sam. Pri tem bo imel na voljo pomoč, ki mu bo pokazala, katera tipka sledi. Na koncu zaigrane skladbe pa bi uporabniku na zaslon prikazala rezultat, v koliki meri mu je uspelo pravilno zaigrati skladbo.

Uporabnik bi imel na voljo tudi možnost nastavljanja nastavitev določenih parametrov kod so: nastavljanje barve klavirja, težavnost in glasnost igranja.

3.2 Videz

Ideja o videzu aplikacije je bil najprej narisan na list papirja, nato pa je s programom Adobe Photoshop dobil končno podobo. Sam videz aplikacije je

preprost, saj mora za otroke tak tudi biti, je pa dovolj zanimiv, da pritegne uporabnika. S programom Adobe Photoshop je bila narejena tudi skica klavirja in sicer dve različici le tega: črno-bela kombinacija tipk in barvna kombinacija tipk. Sledna bi bila lahko za otroke še posebno zanimiva, saj so klasične tipke za otroke mogoče preveč dolgočasne.

Velik poudarek je tudi na videz gumbov, saj so klasični gumbi, katere ponuja program, precej dolgočasni. Zato so v sami aplikaciji nekoliko spremenjeni, saj so živahnih barv in rahlo elipsaste oblike.

3.3 Namen

Namen aplikacije je, da se otroci na zabaven način naučiti igranja klavirja, s tem pa tudi uporabe mobilne tehnologije. S tem, ko uporabnik lahko izbere prosto igranje, pa se pri otroku povečuje tudi ustvarjalnost, saj lahko s ustvarjajo svoje melodije.

Poglavje 4

Implementacija

V tem poglavju bom pojasnil, kako je nastajala celotna aplikacija. Celotna aplikacija je sestavljena iz trinajstih razredov, ti razredi pa dostopajo do desetih xml datotek, katera nam povedo, kakšne videza so same strani aplikacije.

4.1 Izdelava začetnega menija

Glavni meni je začetna stran aplikacije, katera se ponudi uporabniku, ko odpre aplikacijo. Tu ima uporabnik na voljo različne možnosti gumbov, kateri prevedejo na naslednje strani. Tako uporabnik lahko izbira med gumbi »Igraj«, »Nastavitve«, »Navodila«, »O avtorju« in »Izhod«. Ob pritisku gumba »Izhod« se je aplikacija zaprla in prenehala z vsemi aktivnostmi.

Za izdelavo začetne strani je bilo potrebno najprej narediti xml datoteko, v kateri smo podali, kakšnega videza naj bo začetni meni. Tako je na namizju dodanih pet gumbov, kateri pa so povezani z xml datoteko button.xml, v kateri je podan videz vsakega gumba. Potem je potrebno povezati razred KlavirMain.java (to je naš začetni program) povezati z primerno xml datoteko (v našem primeru je to activity_klavir_main.xml). Vse aktivnost (razrede) je potrebno vpisati tudi v AndroidMenifest.xml .

Približno tak postopek se je uporabil pri vseh nadaljnjih razredih, le pri

Slika 4.1: Začetni meni

denimo klavirju, je nekoliko drugače, saj se več razredov sklicuje na isto xml datoteko, saj ima prosti klavir in igranje klavirja različne razrede, ne pa tudi xml datoteko, saj je videz klavirja pri obeh enak.

4.1.1 Glasba

Ves čas v ozadju igrala glasba, katera se naključno izmenjava. Za to je poskrbljeno z razred imenovan PredvajalnikGlasbe.class. V njem se v tabelo shranijo vse skladbe, katere so predhodno shranjene v mapo raw. V razred

PredvajalnikGlasbe.class tudi nekaj metod, katere se uporabljajo za samo predvajanje oziroma ustavljanje glasbe.

V glavnem razredu je bilo potem potrebno naključno generirati številko od 1 do 10 (v sami aplikaciji je bilo 10 skladb). To naključno generirano število se je uporabilo za zaporedno število pesmi. Da je to delovalo brez zapletov je bilo vse pesmi potrebno preimenovati in sicer v naslednjo obliko: vse pesmi so imele v imenu besedo *zvok*, temu pa sledi število.

Ker se to izvrši vedno, ko pridemo v aplikacijo, se vedno predvaja druga skladba in s tem se izognemo, da bi uporabnik poslušal vedno eno skladbo.

```
//predvajanje glasbe - naključne
int nakljucnaSkladba = (int) (Math.random() * 9);
PredvajalnikGlasbe.stevilka = nakljucnaSkladba;
storitev = new Intent(getBaseContext(),PredvajalnikGlasbe.class);
startService(storitev);
```

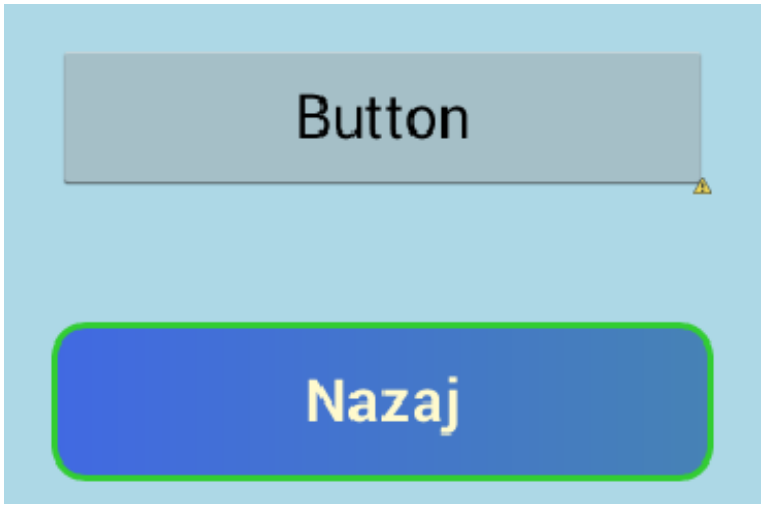
4.1.2 Gumbi

Veliko časa je bilo posvečeno videzu gumbov, saj klasični gumbi, katere ponudi program, niso bili najbolj primerni, saj je aplikacija namenjena otrokom, kateri pa morda želijo malo bolj zaobljene oblike gumbov. Zato je bila napisana datoteka button.xml, v kateri so nastavljeni določeni parametri gumba, da le ta ne bo videti tako monotono. Tu je nastavljena barva gumba, barva obrobe gumba in tudi rob je nekoliko zaobljen.

Datoteka button.xml je bila uporabljena pri vseh nadaljnjih menijih, da so vsi gumbi v aplikaciji vizualno enaki.

4.2 Izdelava navodil

Ob pritisku gumba »Navodila« se nam odpre novo okno, kjer lahko uporabnik izbira med dvema videoma. V prvem video si lahko ogleda, kaj se naredi pri določenih nastavitvah, drugi video pa, kako poteka samo aplikacija učenje in kaj predstavlja pri učenju določen gumb. Videa sta bila posneta s progra-



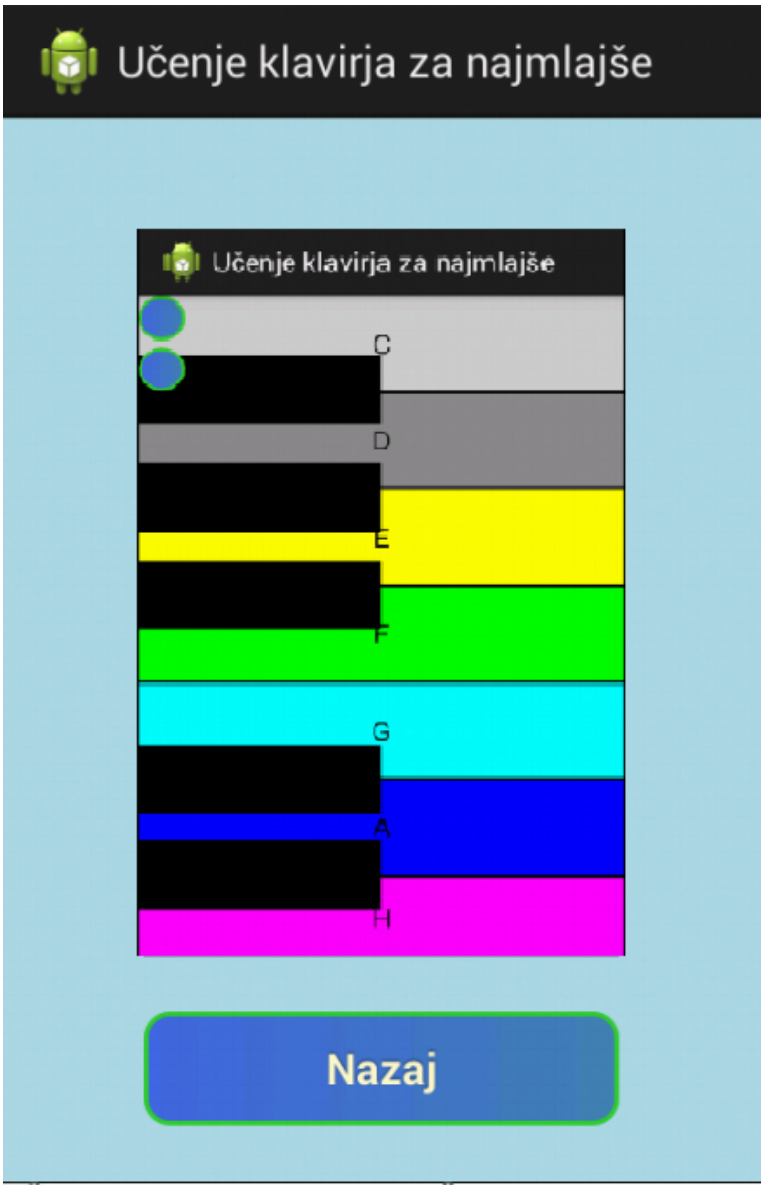
Slika 4.2: Razlika med navadnim gumbom in gumbom uporabljenim v aplikaciji

mom Windows Media Encoder, s katerim lahko posnamemo dele zaslona in tako je posneto okno emulatorja, v katerem je prikazana določena tema. S Windows Media Encoder pa ne moremo dodajati napisov, lahko pa zato napise dodamo z Windows Movie Maker, s katerim sta videa dobila tudi pisne navodila za uporabo.

4.3 Implementacija nastavitev

Pri izdelavi okna za nastavitve je bilo največ težav, saj se morajo vsi nastavljeni parametri v nastavitvah shraniti in jih potem nekako prikazati (upoštevati) pri samem klavirju. Nastavitve, katere ima uporabnik na voljo so sledeče: nastavljanje glasnosti zvoka in glasbe, videz klavirja in težavnostjo samega igranja. Pri videzu klavirja lahko uporabniki izbira med klasičnim in barvnim klavirjem, pri težavnosti pa ima tri stopnje, kako hitro bo določena skladba zaigrana.

Ko uporabnik nastavi določen parametre, se le ti shranijo. Tako uporabniku, ko ponovno zažene aplikacijo, ni potrebno še enkrat nastavljati pa-



Slika 4.3: Navodila

rametrov. Za to je poskrbljeno s tem, da se v tabelo shranijo podatki o nastavljenih vrednostih in se ob vsakem zagonu aplikacije le te preberejo iz tabele, in nastavijo parametre. Ta metoda je uporabljena tudi pri klavirju.

Pri samih nastavitvah je uporabljenih kar nekaj različnih gradnikov. Tako je za nastavljanje glasnosti uporabljen gradnik SeekBar. To je drsnik, s katerim lahko uporabnik regulira jakost zvoka. Za izbiro barve in težavnost pa je uporabljen gradnik RadioGroup, ki pa je sestavljen iz več gradnikov RadioButton. S tem uporabnik lahko izbere le en RadioButton (to je na nek način gumb) znotraj RadioGroup (skupina gumbov).

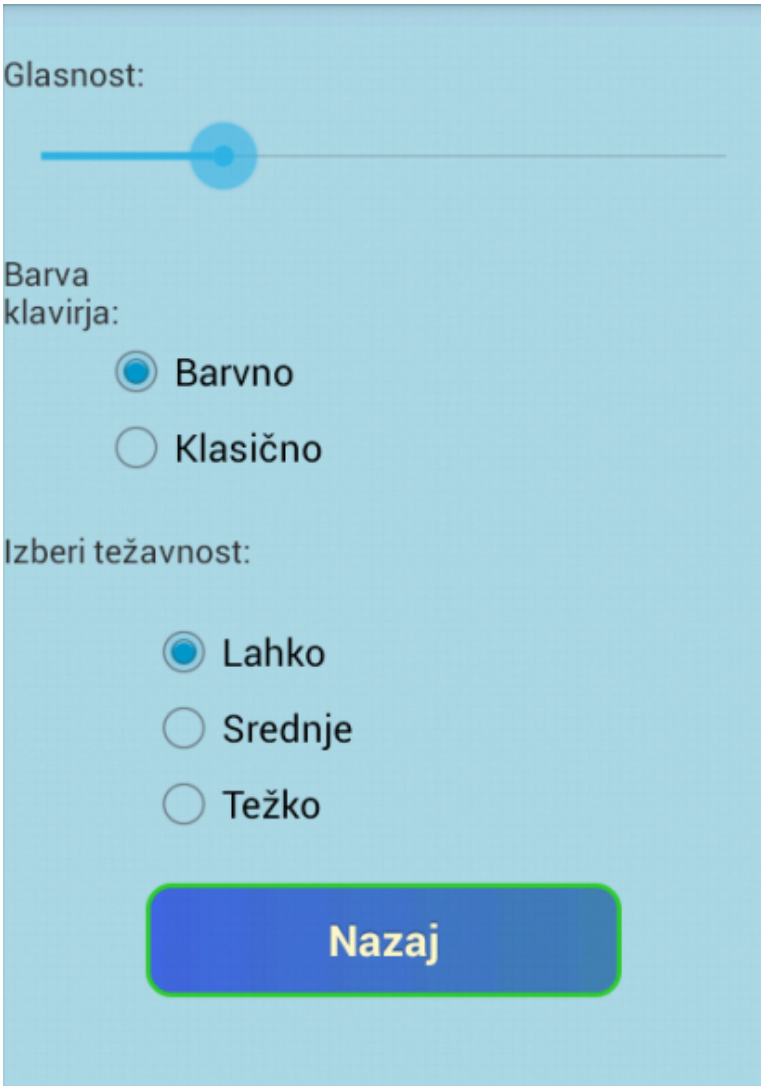
4.4 Prosto igranje

Pri prostem igranju lahko uporabnik poljubno pritiska tipke klavirja. Sam videz klavirja je odvisen od nastavitev, katere smo nastavili. Tako je sama barva klavirja lahko klasična, to pomeni črno - bele tipke, ali pa barvna, pri kateri je vsaka tipka klavirja različne barve in je mogoče zaradi tega za otroke bolj privlačna. Ko se nam sam klavir prikaže, se glasba, katera nas je spremljala skozi menije, ugasne. Na tipkah klavirja so, ker se nekateri učijo po notah, napisani tudi toni.

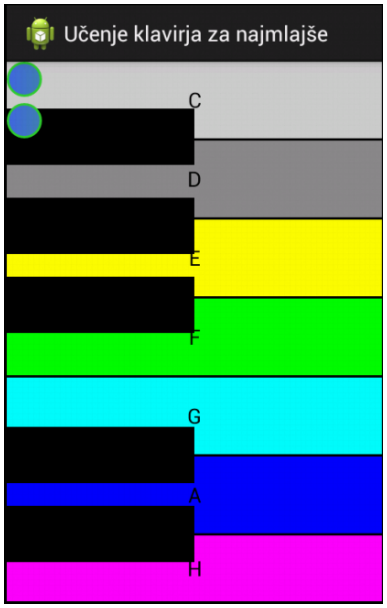
Ob samem pritisku tipke, se nam tipka, ki je v bistvu gumb, za pol sekunde obarva v rdečo barvo in s tem uporabnik ve, katera tipka je pritisnjena. Za samo implementacijo tega, da se tipka obarva v rdečo barvo je potrebno, kar precej preverjanja, saj je potrebno preveriti, katera tipka je pritisnjena in na kakšne so nastavitve klavirja, saj se mora po pol sekunde tipka spremeniti v začetno obliko. Ob pritisku je bilo potrebno poskrbeti, da se je ob pritisku slišal pravi ton.

4.5 Izbor skladbe za učenje

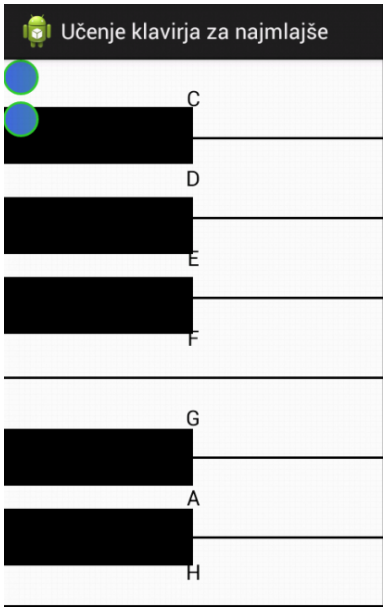
Ko v meniju »Igraj« izberemo učenje, se nam pojavi novo okno, v katerem lahko uporabnik lahko izbere skladbo, katero se želi učiti igrati. Ko le to



Slika 4.4: Nastavitve



Slika 4.5: Barvni klavir



Slika 4.6: Črno - beli klavir

naredi, se mu pokaže klavir, le da je tokrat nekoliko drugače kot pri prostem igranju.

Tu je bilo potrebno poskrbeti za naslednje stvari:

1. stvar je, da se ob izboru katere koli skladbe, glasba v ozadju ugasne.
2. stvar je, da odvisno katero skladbo bo uporabnik izbral, jo mora aplikacija v naslednjem koraku zaigrati uporabniku in zato sem moral to nekako shraniti.
3. stvar pa je, da se ob pritisku na gumb pojavi klavir.

Za to je uporabljen podprogram, kateri napravi vse potrebne stvari nadaljnjo delo.

```
public void izborSkladbe(Button a, final int st){
    a.setOnClickListener(new OnClickListener(){
    public void onClick(View v){
        Intent intent = new Intent(v.getContext(), KlavirUcenje.class);
        StringBuilder skladba = new StringBuilder(100);
        skladba.append(škladba");
        skladba.append(st);
        startActivityForResult(intent, 0);
        SharedPreferences.Editor editor = izbSkladbe.edit();
        editor.putString("izborSkladbe", skladba.toString());
        editor.commit();
        stopService(storitev);
        }
    });
}
```

4.6 Učenje skladbe

Pri učenju skladbe je potrebno biti pozoren na vse nastavitve, katere je uporabnik izbral. Pri prostem igranju je bilo dovolj, da smo bili pozorni le na barvo klavirja, tu pa moramo upoštevati še težavnost, saj le ta vpliva na igranje samih tonov klavirja. Tudi tu se tipka, katera je pritisnjena obarva,

Slika 4.7: Izbira skladbe

le da tu to dobi še drugi pomen kot pri prostem igranju. Tu lahko uporabnik vidi, katero tipko mora pritisniti, da bo dobil željen ton.

Ko v meniju »Učenje« pritisnemo določeno skladbo in če je le ta skladba kratka, se nam pojavi klavir, na samem klavirju pa se nam začno odštevati tri sekunde, po katerih se nam zaigra izbrana skladba. Ta animacija je naredil tako, da se uporabijo štiri slike, narisane v programu Adobe Photoshop. Slike so velikosti 500x500 slikovnih točk. Te štiri slike imajo prosojno ozadje, na njih pa so bile: na eni sliki število 3, na drugi sliki število 2, na tretji sliki število 1 in na četrti sliki napis start. Da so se slike lepo izmenjavale in na nek način uporabnika opozarjale, naj se pripravi, je poskrbel podprogram, kateri se je zagnal pri vsaki izbiri pesmi.

```
public void prikaziSliko(final long delay, final int i, final ImageView slika){
    final Handler zakasnitev = new Handler();
    Timer t = new Timer();
    t.schedule(new TimerTask() {
        public void run() {
            zakasnitev.post(new Runnable() {
                public void run() {
                    if(i == 0)
                        slika.setVisibility(View.VISIBLE);
                    else
                        slika.setVisibility(View.INVISIBLE);
                }
            });
        }
    }, delay);
}
```

Ko je z zaslona izginila slika, na kateri je bil napis start, je klavir pričel igrati izbrano skladbo. Sama hitrost predvajanja je bila odvisna od težavnosti, katero smo izbrali. Če je uporabnik izbral težavnost *težko*, je ton trajal 0,7s, pri izbrani težavnost *srednje* 1,2s in pri izbrani težavnost *lahko* pa ton traja 1,7s.

```
//težavnost bo pomenila hitrost samega izvajanja pesmi
podTezavnost = getSharedPreferences(podatki, Context.MODE_PRIVATE
);
final String vrnjenaVrednostTezavnosti = podTezav-
nost.getString("podatekTezavnost", "klasicno");
//System.out.println(vrnjenaVrednostTezavnosti);
if(vrnjenaVrednostTezavnosti.equals("lahko")){
    hitrost = 1700;
}
else if(vrnjenaVrednostTezavnosti.equals(šrednje")){
    hitrost = 1200;
}
else{
    hitrost = 700;
}
```

Ob vsakem tonu, kateri je bil zaigran, se poleg zvoka tona obarva tudi tipka klavirja, kateremu določen ton pripada in s tem si uporabnik lahko vizualno in slušno zapomni, kako poteka melodija skladbe. Pri sami implementaciji tona je uporabljen Android razred SoundPool, kateri je namenjen predvsem igranju kratkih zvočnih efektov. Same tone sem dobil na spletu.[4]

```
public SoundPool sndPool;
int ton1 = 0, ton2 = 0, ton3 = 0, ton4 = 0, ton5 = 0, ton6 = 0, ton7 = 0;
Button bt_tonC, bt_tonD, bt_tonE, bt_tonF, bt_tonG, bt_tonA, bt_tonB;
...
sndPool = new SoundPool(7, AudioManager.STREAM_MUSIC, 0);
    ton1 = sndPool.load(this, R.raw.c, 1);
    ton2 = sndPool.load(this, R.raw.d, 1);
    ton3 = sndPool.load(this, R.raw.e, 1);
    ton4 = sndPool.load(this, R.raw.f, 1);
    ton5 = sndPool.load(this, R.raw.g, 1);
    ton6 = sndPool.load(this, R.raw.a, 1);
    ton7 = sndPool.load(this, R.raw.b, 1);
...
//tu je ton tisti, kateri je pritisnjen
sndPool.play(tonSound, 1, 1, 0, 0, 1);
```

Samo igranje skladbe poteka nekako tako, da program v zaporedju zaigra že napisano pesem.

Ko program zaigral skladbo do konca, se pojavi simpatični lik, kateri je bil sem prav tako narisal v programu Adobe Photoshop, kateri uporabniku pove, da je aplikacija z igranjem zaključila in da lahko sedaj še sam poskusi zaigrati skladbo. Videz lika je moral biti tak, da uporabnika pritegne in ga razveseli, zato so uporabljene živahne barve, lik pa je na videz prijazen in zabaven.

Ko uporabnik pritisne toliko tipk, kolikor je dolga skladba, se mu izpiše obvestilo, katero mu čestita za odigrano pesem in ga vpraša, ali želi igrati v prostem načinu ali pa bi se mogoče želel vrniti v meni »Učenje« in tam zopet poskusil s učenjem te ali katere druge skladbe.

Pri samem klavirju je vključena tudi tipko pomoč, s katero aplikacija uporabniku pomaga, kateri ton sledi in ga s tem usmeri, kaj ton mora pritisniti. To sem naredil tako, da program v spremenljivko tipa string zapisoval vse tone, kateri so v pesmi. To naredi z vsakim tonom, kateri je bil zaigran in

Slika 4.8: Lik

tako dobimo zaporedje tonov. Potem pa aplikacija preveri, koliko tonov je uporabnik že pritisnil in mu prikazal na zaslon, kateri je naslednji ton.

```
public void pomoc(){
    String steviloTon = "ton"+ (stevecPritisnjenihTonov+1);
    String naslednjiTon = ;
    final String vrnjenaVrednostZadetkov = pritisnjeni-
Toni.getString(steviloTon.toString(), );
    if(vrnjenaVrednostZadetkov.equals("1")){
        naslednjiTon = "ton C";
        ... Tu so opisane še ostale tipke
        AlertDialog.Builder opozorilo = new AlertDialog.Builder(this);
        opozorilo.setMessage("Naslednji ton, keterega morate pritisniti je:
"+ naslednjiTon);
        opozorilo.setTitle("Pomoč");
        opozorilo.setCancelable(true);
        opozorilo.setPositiveButton("Ok",
        new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {
            //dismiss the dialog
        }
    });
    opozorilo.create().show();
}
```

To je bilo v zadnji implementaciji spremenjeno, saj vsi otroci še ne znajo brati in se namesto tega, da se pojavi obvestilo na zaslon, pojavi lik na gumbu, katerega mora otrok pritisniti.

4.7 Skladbe

V sami aplikaciji sta uporabljeni dve ljudski skladbi in sicer Kuža pazi in Mojster Jaka. Prva pesem je kratka (Kuža pazi) in se jo uporabnik lahko nauči v celoti, druga pa je daljša (Mojster Jaka) in je zato razdelil v več različnih sektorjev.

4.7.1 Kuža pazi

Kuža pazi je preprosta skladba za učenje klavirja. Ima 15 tonov in dovolj lahko melodijo, da si lahko zapomnimo vse tone. Sama pesem je zasnovana tako, da se za vsak ton izvrši podprogram zaigrajTipko, v katerem se glede na pritisnjeno tipko zaigra ton in obarva tipko, vse skupaj pa se zgodi z določenim zamikom (v sami kodi je to predstavljeno s besedo delay).

```
private Runnable kuzaPaziPotek = new Runnable(){
    long delay = 7000;
    @Override
    public void run(){
        zacetekIgranja();
        Button bt_tonC = (Button) findViewById(R.id.button_tonC);
        Button bt_tonD = (Button) findViewById(R.id.button_tonD);
        Button bt_tonE = (Button) findViewById(R.id.button_tonE);
        zaigrajTipko(ton1, delay, bt_tonC, "tonc");
        delay = delay + hitrost;
        zaigrajTipko(ton1, delay, bt_tonC, "tonc");
        delay = delay + hitrost;
        ... tu sledi še več tonov
        zaigrajTipko(ton1, delay, bt_tonC, "tonc");
        delay = delay + hitrost;
        zaigrajTipko(ton1, delay, bt_tonC, "tonc");
        delay = delay + hitrost;
        zaigrajTipko(ton1, delay, bt_tonC, "tonc");
        delay = delay + hitrost;
        prikaziSliko(delay, 0, lik);
        delay = delay + 3000;
        prikaziSliko(delay, 1, lik);
    }
};
```

4.7.2 Mojster Jaka

Mojster Jaka je prav tako dokaj preprosta skladba, a bi se jo bilo zaradi svoje dolžine težko naučiti v celoti, zato je razdeljena na štiri različne sektorje, uporabnik pa lahko poskusi zaigrati tudi celotno skladbo.

Sam sektor je sestavljen prav tako kot je sestavljena skladba Kuža pazi.

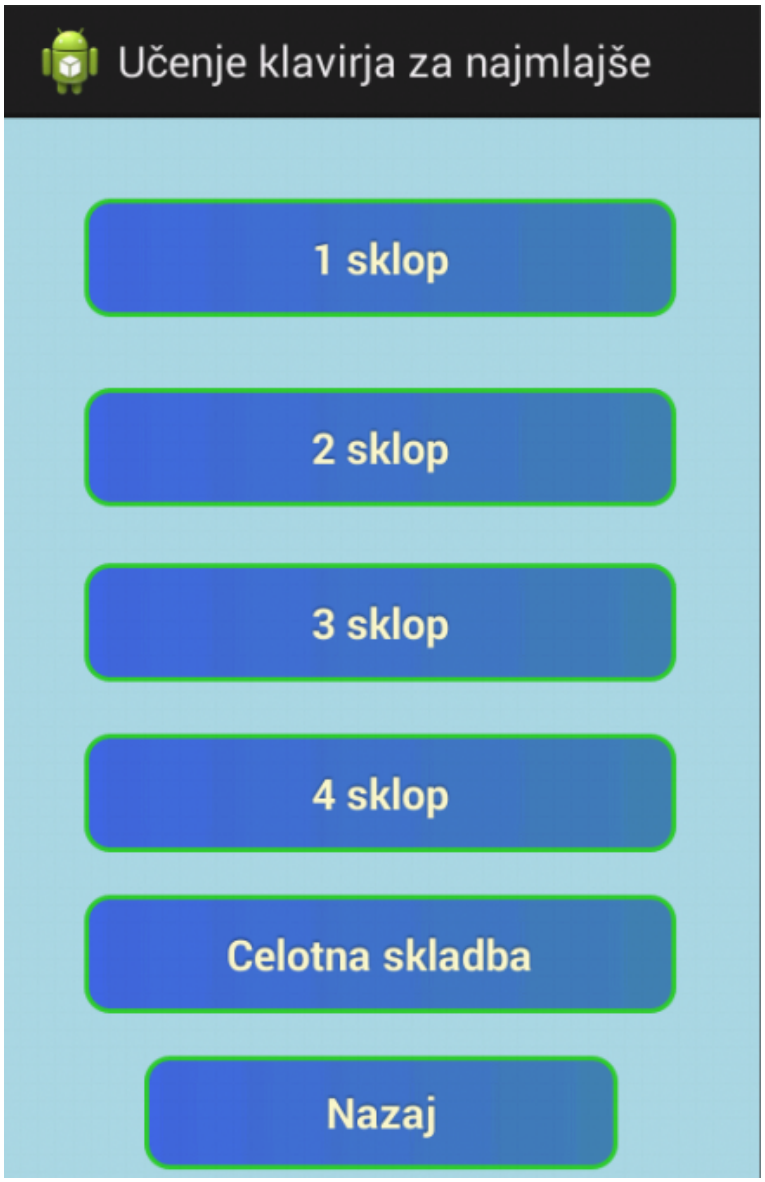
4.8 Rezultati igranja

Ko končamo z učenjem, pa nam aplikacija vrne, koliko tonov smo zaigrali pravilno. To je implementirano tako, da imamo neko spremenljivko, ki je nastavljena na 0 in potem se z vsakim zaigranim tonom vrednost poveza za 1. Istočasno pa se v spremenljivko tipa SharedPreferences shranjuje tudi tone, kateri so bili v določeni skladbi pritisnjeni.

```
stevecZaigranihTonov = stevecZaigranihTonov + 1;
//System.out.println(stevecZaigranihTonov);
String steviloTon = "ton"+ stevecZaigranihTonov;
SharedPreferences.Editor editorTon = pritisnjeniToni.edit();
editorTon.putString(steviloTon.toString(), +ton);
editorTon.commit();
```

Ko je skladba v celoti zaigrana (to pomeni po tolikem številu tonov kot jih je v sami skladbi ali sklopu), se nam pojavi okno z rezultatom, kateri nam pove, koliko tonov je bilo pravilno zaigranih. Samo preverjanje poteka tako, da se ton, katerega uporabnik pritisne, preveri z tonom, katerega smo shranili v spremenljivko tipa SharedPreferences (vsak ton posebej). To pomeni, da se recimo 12 ton že zaigrane skladbe preveri z 12 tonom pritisnjene tipke.

Samo opozorilo pa je sestavljeno tako, da poda, koliko tonov je bilo pravilno zaigranih (če je uporabnik pravilno zaigral 12 od 15 tono, se izpis glasi: *Čestitamo: Pravilno ste zaigrali 12 od 15 tonov!*). Uporabnik lahko v samem opozorilu izbere prosto igranje ali se odloči, da bo nadaljeval z učenjem druge skladbe.



Slika 4.9: Izbor sektorjev skladbe Mojster Jaka

```
stevecPritisnjenihTonov = stevecPritisnjenihTonov + 1;
primerjajaZadetke(stevecPritisnjenihTonov, tonSound);
if(stevecPritisnjenihTonov == stevecZaigranihTonov)
    prikaziOpozorilo(0);
```

To je bilo v zadnji implementaciji spremenjeno, saj vsi otroci še ne znajo brati in se namesto tega prikažejo zvezdice, katere nam bi povedale rezultat igranja. Samih zvezdic je deset, potem pa se le te obarvajo glede na procent uspešnosti zaigrane skladbe.

Poglavje 5

Sklepne ugotovitve

Sama izdelava Android aplikacije je lahko težavna, a ko enkrat ugotoviš, kako se kakšno stvar implementira, je vse dosti lažje. Določene probleme, katere nastanejo pri izdelavi določene problema pa je možno rešiti na več načinov. Ne obstaja samo ena rešitev, ampak je mogoče problem rešiti tudi drugače, ne da bi bil pri tem rezultat različen.

Aplikacija je bila namenjena otrokom, da bi se na nek drug način naučili igranje klavirja. S privlačnim in preprostim videzom je aplikacija namenjena vsem, ki bi se želeli nekoliko poigrati s igranjem virtualnega klavirja, s nastavljanjem različnih nastavitev pa je aplikacija namenjena tudi nekoliko starejšim, saj si s tem lahko otežijo ali pa olajšajo samo učenje klavirja.

Ideje za nadaljnje delo:
1. Prevod v več različnih jezikov in s tem razumljivost večjemu številu ljudi.
2. Dodajanje na spletno stran, ki uporabniku ponuja prenos aplikacije na mobilno napravo.
3. Dodajanje skladbe samega uporabnika. Tu bi lahko recimo starši dodali pesem, katero bi potem otrok vadil. Ali pa bi nekdo lahko svojo pesem samo shranil in bi jo potem nekdo drug lahko vadil.
4. Možno bi bilo narediti večje število tipk, le tu pa se lahko pojavi problem, saj so mobilni telefoni še vedno kar majhni, lahko pa bi bilo to uporabljeno recimo na tabličnih računalnikih.

Slike

Literatura

[1] (2013) Android operacijski sistem
Dostopno na: http://sl.wikipedia.org/wiki/Android_(operacijski_sistem)

[2] (2013) Eclipse (software)
Dostopno na: http://en.wikipedia.org/wiki/Eclipse_(software)

[3] (2013) Mobile operating system
Dostopno na: http://en.wikipedia.org/wiki/Mobile_operating_system

[4] (2013) University of Iowa Electronic Music Studios
Dostopno na: http://theremin.music.uiowa.edu/MISpiano.html

[5] (2013) Programski jezik java
Dostopno na: http://sl.wikipedia.org/wiki/Programski_jezik_java

[6] (2013) XML
Dostopno na: http://sl.wikipedia.org/wiki/XML

[7] (2013) Kratka zgodovina Androida
Dostopno na: http://slo-android.si/prispevki/clanki/item/1007-kratka-zgodovina-androida.html

[8] (2013) About the Eclipse Foundation
Dostopno na: http://www.eclipse.org/org/

[9] (2013) Google play
Dostopno na: https://play.google.com/store

[10] (2013) Pianist HD - Finger Tap Piano
Dostopno na: https://play.google.com/store/apps/details?id=com.rubycell.pianisthd&h

[11] (2013) Piano Melody Free
Dostopno na: https://play.google.com/store/apps/details?id=com.veitch.themelodymas

[12] (2013) Piano Instructor (Lite)
Dostopno na: https://play.google.com/store/apps/details?id=com.okythoos.android.pia