VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

INOVATIVNÍ PŘEHRÁVAČ HUDBY PRO CHYTRÉ TELEFONY A PC

INNOVATIVE MUSIC PLAYER FOR SMARTPHONES AND PC

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. ROMAN RICHTER**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing. ADAM HEROUT, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Richter Roman, Bc.**
Program: Informační technologie Obor: Informační systémy
Název: **Inovativní přehrávač hudby pro chytré telefony a PC**
Innovative Music Player for Smartphones and PC
Kategorie: Uživatelská rozhraní
Zadání:

1. Prostudujte problematiku vývoje aplikací - jak pro mobilní platformy, tak pro PC, včetně komunikace takovýchto aplikací.
2. Zhodnoťte vlastnosti stávající verze přehrávače Supernova Music Player. Identifikujte prostor pro vylepšení.
3. Vylepšujte přehrávač SMP, zaměřte se na uživatelskou zkušenost a použitelnost přehrávače.
4. Analyzujte možnosti vytvoření kompatibilního přehrávače pro PC. Zaměřte se na komunikaci přehrávačů, bezpečnost, soukromí a uživatelskou zkušenost a použitelnost.
5. Navrhněte přehrávač hudby pro PC, který umožní maximalizovat užitek ve spolupráci s mobilním přehrávačem.
6. Implementujte přehrávač pro PC a rozhraní pro komunikaci mezi přehrávači.
7. Iterativně vylepšujte svoje řešení. Získejte uživatele, kteří budou fungovat jako testeři a veďte s nimi dialog - testujte vytvářené řešení a iterativně je vylepšujte.
8. Zhodnoťte dosažené výsledky a navrhněte možnosti pokračování projektu; vytvořte plakátek a krátké video pro prezentování projektu.

Literatura:

- Steve Krug: Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability, ISBN: 978-0321965516
- Steve Krug: Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability, ISBN: 978-0321657299
- Joel Marsh: UX for Beginners: A Crash Course in 100 Short Lessons, O'Reilly 2016
- Susan M. Weinschenk: 100 věcí, které by měl každý designér vědět o lidech, Computer Press, Brno 2012
- Android Developers: https://developer.android.com/index.html

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 4, značné rozpracování bodů 5 a 6.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Herout Adam, prof. Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cieľom tejto práce je vytvorenie hudobného prehrávača ako pre chytré telefóny, tak pre PC, ktorý pracuje s lokálnymi hudobnými súbormi v užívateľovom zariadení a ktorý sa dokáže naučiť, ktoré pesničky má užívateľ rád a to na základe jeho činností počas počúvania hudby. Prehrávač si okrem iného dokáže zapamätať, ktoré pesničky užívateľ preskočil, kedy zvýšil hlasitosť prehrávania alebo koľkokrát bola daná pesnička prehraná. Každá pesnička má skóre, ktoré sa vypočíta na základe týchto činností. Čím väčšie je skóre, tým väčšia je aj šanca, že daná pesnička bude v budúcnosti prehraná. Výsledkom mojej práce sú dve plnohodnotné verzie hudobného prehrávača, ktoré dokážu medzi sebou komunikovať pre zaistenie synchronizácie skóre pesničiek. Hlavným prínosom tejto práce je zlepšenie užívateľskej skúsenosti počas počúvania hudby, ktoré je dosiahnuté vlastným algoritmom pre výber pesničiek a minimalistickým užívateľským rozhraním.

Abstract

The goal of this thesis is to create a music player for smartphones as well as PCs that works with local music files in the user's device and which can learn which songs does the user like based on their actions during listening to music. The music player can, among other things, remember which songs were skipped by the user, when was volume turned up, or how many times was a certain song played. Each song has a score that is calculated based on these actions. With a higher score, there is also a higher chance of playing the song in the future. The results of my thesis are two full-featured versions of music player, which are capable of communication with each other to ensure synchronization of song scores. The main benefit of this thesis is an improvement of user experience during listening to music, which is achieved by the application's algorithm for song selection and minimalistic user interface.

Kľúčové slová

chytrý prehrávač hudby, výber pesničiek podľa činností užívateľa, Android, UWP

Keywords

smart music player, song selection based on user's actions, Android, UWP

Citácia

RICHTER, Roman. *Inovativní přehrávač hudby pro chytré telefony a PC*. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing. Adam Herout, Ph.D.

Inovativní přehrávač hudby pro chytré telefony a PC

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána prof. Adama Herouta. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Roman Richter
14. mája 2021

Poďakovanie

Rád by som sa poďakoval vedúcemu mojej diplomovej práce prof. Adamovi Heroutovi, ktorý mi umožnil pracovať na zvolenom zadaní diplomovej práce. Rád by som mu taktiež poďakoval za jeho postrehy a nápady, ktoré mi pomohli výrazne zvýšiť užívateľskú skúsenosť a použiteľnosť prehrávača, ako aj za výpomoc pri testovaní aplikácie a pri písaní tejto práce.

Obsah

Kapitola 1

Úvod

Táto práca popisuje návrh a implementáciu inovatívneho hudobného prehrávača pre chytré telefóny a pre osobné počítače. Hudobný prehrávač pracuje výlučne s lokálnymi hudobnými súbormi v užívateľovom zariadení a dôvodom jeho vytvorenia je fakt, že mnoho existujúcich hudobných prehrávačov fungujúcich na tomto princípe vyberá pesničky jednoduchým spôsobom – buď je vybraná nasledujúca pesnička zo zoznamu, alebo je výber pesničiek čisto náhodný. Užívateľ ale môže mať vo svojom zariadení aj pesničky, ktoré sa mu po nejakej dobe prestali páčiť alebo pesničky, ktoré boli súčasťou albumu a užívateľ o ne nikdy nemal väčší záujem, no väčšina ostatných prehrávačov hudby mu ich bude prehrávať a to aj v prípade, keď užívateľ tieto pesničky vždy preskočí.

Obsahom tejto práce je popis toho, ako vytvoriť hudobný prehrávač, ktorý je schopný riešiť tento problém. To zahŕňa všetko od ukladania užívateľových dát cez algoritmus výberu pesničiek a ďalšie algoritmy zabezpečujúce chod prehrávača až po tvorbu grafického užívateľského rozhrania. Všetky tieto časti sú popísané ako pre platformu Android, ktorá je použitá na vývoj prehrávača pre chytré telefóny, tak pre platformu Universal Windows Platform (UWP), ktorá bola zvolená na vývoj pre osobné počítače. Text tejto práce ďalej obsahuje popis spôsobu komunikácie medzi týmito dvoma platformami, popis testovania a získavania nových užívateľov, ako aj popis vylepšení existujúcej mobilnej verzie, pričom jej základ bol výstupom mojej bakalárskej práce. Táto práca pozostáva z pomerne rozsiahleho počtu strán, no v texte sa nachádza množstvo obrázkov (najmä v implementačných kapitolách), ktoré zobrazujú užívateľské rozhranie prehrávača na chytrom telefóne a aj na počítači.

Hudobný prehrávač je pre chytré telefóny voľne dostupný v obchode aplikácií Google Play pod názvom „Supernova Music Player“ (https://play.google.com/store/apps/details?id=com.starko.supernovaMP). Pre počítače je hudobný prehrávač dostupný pod rovnakým názvom v obchode aplikácií Microsoft Store (https://www.microsoft.com/sk-sk/p/supernova-music-player/9nc35kzh1pts).

Kapitola 2

Existujúce hudobné prehrávače

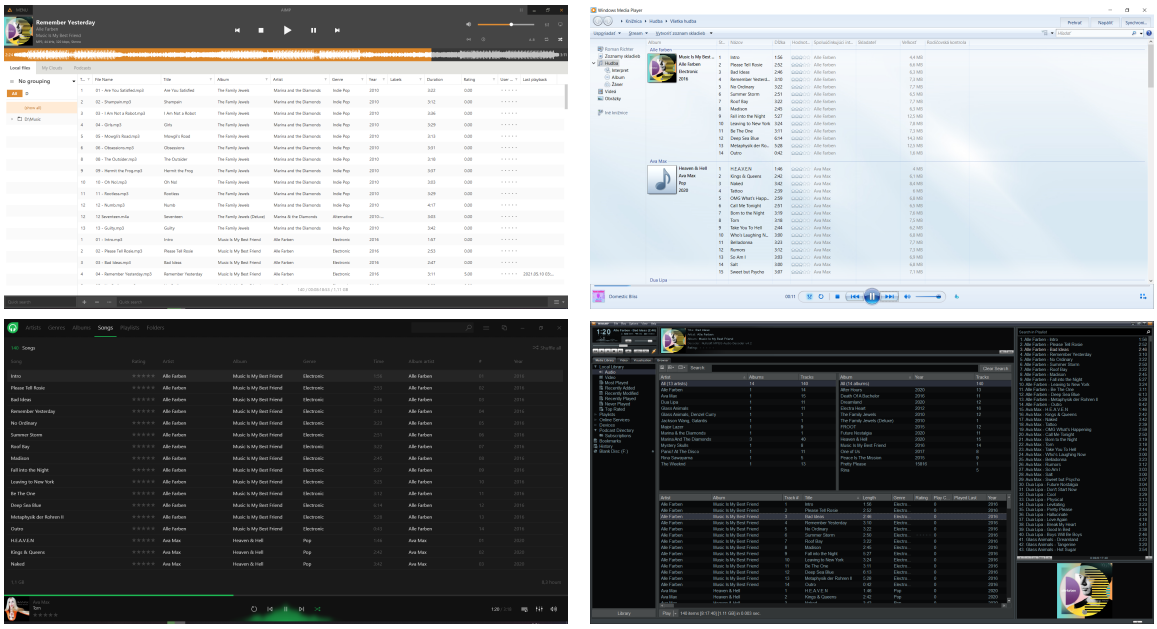
Predtým, ako som začal prehrávač hudby implementovať, tak som sa snažil zistiť, či už neexistuje prehrávač s podobným správaním. Väčšina hudobných prehrávačov, ktorá pracuje s lokálnymi hudobnými súbormi v zariadení, sa od seba ale často odlišuje len po vizuálnej stránke, prípadne doplnkovými funkciami, akými je ekvalizér, zobrazenie textu pesničky alebo úprava metadát daného hudobného súboru. Väčšina hudobných prehrávačov, ktoré som analyzoval, dokázala prehrávať hudbu len sekvenčne alebo čisto náhodne bez toho, aby sa do úvahy brali preferencie užívateľa. To, že sa daná pesnička v užívateľovom zariadení nachádza, nemusí vždy znamenať, že si ju bude chcieť vypočuť rovnako často ako všetky ostatné pesničky. Preferencie užívateľa berú do úvahy online prehrávače hudby, pre ktoré to je takmer nevyhnuté, keďže väčšinou obsahujú obrovské množstvo pesničiek z celého sveta. Tento typ prehrávačov je ale práve kvôli tejto skutočnosti postavený na úplne inom princípe ako hudobný prehrávač, ktorý je výstupom mojej diplomovej práce, a preto sa im ani táto kapitola bližšie nevenuje.

2.1 Hudobné prehrávače pre PC

Pred implementáciou môjho hudobného prehrávača pre počítače som analyzoval nasledujúce existujúce riešenia: AIMP, Dopamine, foobar2000, Groove Music, MediaMonkey, MusicBee, VLC media player, Winamp a Windows Media Player. Každý z týchto hudobných prehrávačov má špecifické užívateľské rozhranie a rozsiahlu sadu funkcií. Okrem samotného prehrávania hudby umožňujú tieto prehrávače aj upravovať zvuk cez ekvalizér, upravovať metadáta hudobných súborov a získavať chýbajúce metadáta z internetu, vytvárať a upravovať zoznamy skladieb, zobrazovať text pesničiek, napaľovať súbory na disk alebo prehrávať podcasty. Na druhej strane pre aplikácie všeobecne platí, že čím viac funkcií poskytujú, tak tým ťažšie môže byť naučiť sa ich ovládať [1]. Všetky tieto prehrávače poskytujú jednoduché vypnutie a zapnutie náhodného výberu pesničiek a nastavenie opakovania. Hudobné prehrávače MediaMonkey a MusicBee navyše poskytujú funkciu s názvom „Auto-DJ“, ktorá umožňuje prehrávať hudbu na základe zadefinovaných kritérií.

Základom užívateľského rozhrania každého prehrávača je zoznam pesničiek a panel na ovládanie prehrávania (obr. 2.1). Okrem toho obsahujú užívateľské rozhrania týchto prehrávačov záložky alebo iný spôsob, akým je možné filtrovať interpretov, albumy, hudobné žánre a podobne. Prehrávače taktiež často umožňujú zmeniť vzhľad aplikácie, či už úpravou farieb alebo zvolením medzi svetlým alebo tmavým režimom aplikácie. Hudobné prehrávače ako Winamp alebo MediaMonkey sa snažia zobraziť čo najviac informácií na jednej obrazovke.

Tá okrem zoznamu pesničiek a ovládacieho panelu obsahuje aj zoznam všetkých interpretov a zoznam všetkých albumov. MediaMonkey dokonca zobrazuje na svojej hlavnej obrazovke aj zoznam všetkých hudobných žánrov a text práve prehrávanej pesničky. Užívateľ má síce všetky informácie na jednom mieste, no zároveň môže takéto užívateľské rozhranie pôsobiť preplnene a neprehľadne. Neprehľadné bývajú aj nastavenia mnohých hudobných prehrávačov. Tie zvyknú obsahovať značne veľké množstvo možností nastavenia prehrávača a kvôli tomu môže mať nový užívateľ problém nájsť to, čo potrebuje.



Obr. 2.1: Obrázky zobrazujú užívateľské rozhranie hudobných prehrávačov pre počítače, ktorého základom je zoznam pesničiek a ovládací panel. V hornom rade je zľava doprava zobrazený hudobný prehrávač AIMP a Windows Media Player. V dolnom rade je v rovnakom poradí zobrazený hudobný prehrávač Dopamine a Winamp.

2.2 Hudobné prehrávače pre chytré telefóny

Pri analýze existujúcich hudobných prehrávačov pre chytré telefóny som sa zameral na dve skupiny prehrávačov:

1. najpopulárnejšie hudobné prehrávače, ktoré sa užívateľovi zobrazia, keď do vyhľadávania v obchode Google Play zadá heslo „music player“,

2. chytré hudobné prehrávače, ktoré sa užívateľovi zobrazia, keď do vyhľadávania v obchode Google Play zadá heslo „smart music player“.

Z prvej skupiny som pre ukážku vybral nasledujúce prehrávače: Music Player – MP3 Player[1], Pulsar Music Player – Mp3 Player, Audio Player[2] a hudobný prehrávač s obecným názvom Music player[3], pričom počet stiahnutí každej z týchto aplikácií je možné počítať v miliónoch.

[1] https://play.google.com/store/apps/details?id=com.shaiban.audioplayer.mplayer
[2] https://play.google.com/store/apps/details?id=com.rhmsoft.pulsar
[3] https://play.google.com/store/apps/details?id=com.media.music.mp3.musicplayer

Okrem týchto troch hudobných prehrávačov som zanalyzoval množstvo ďalších populárnych prehrávačov hudby, no ich princíp fungovania a užívateľské rozhranie je vo väčšine prípadov veľmi podobné tomu, aké majú zvolené hudobné prehrávače.

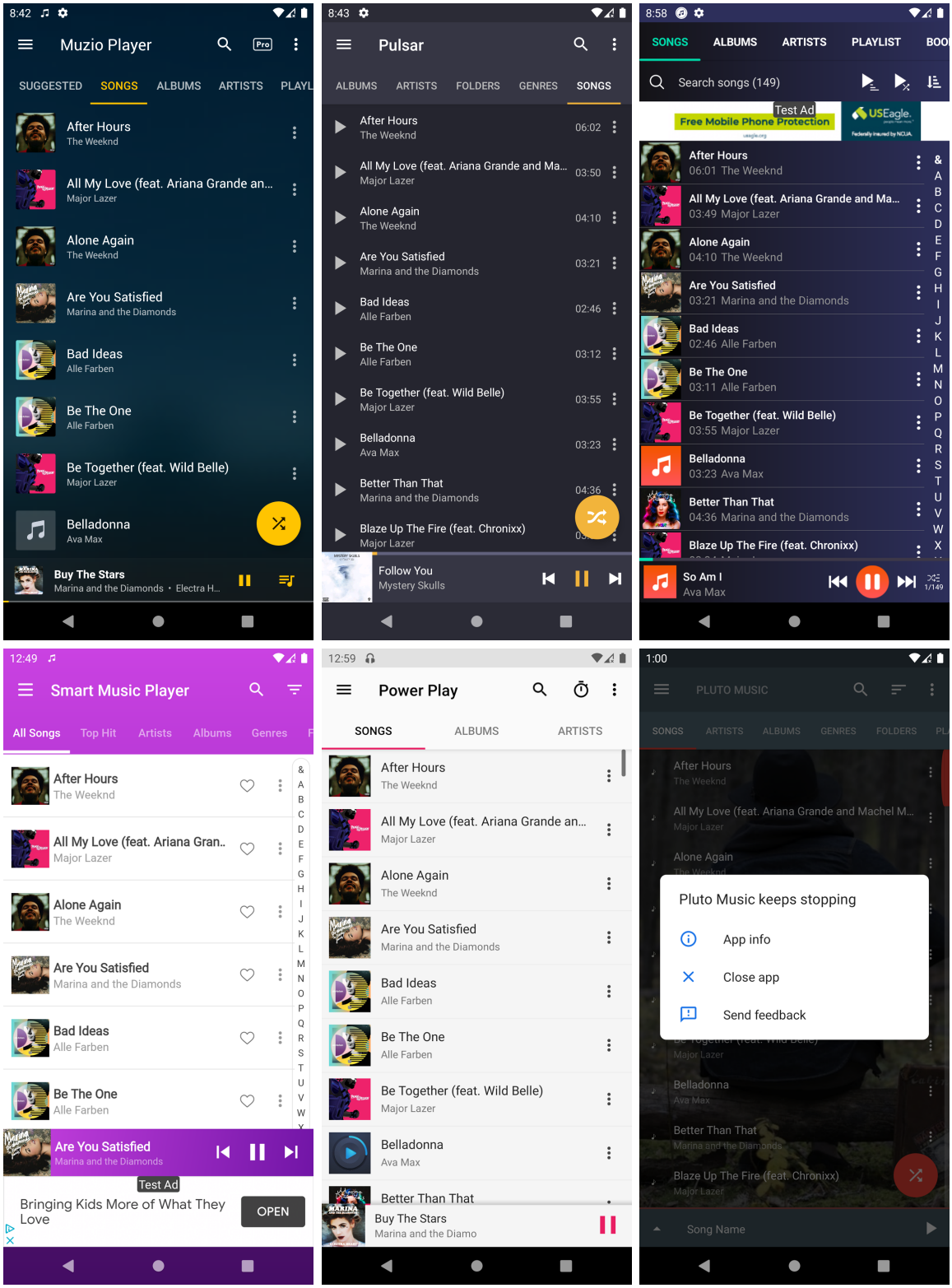
Medzi spoločné kľúčové prvky všetkých troch hudobných prehrávačov patrí podpora rôznych hudobných formátov, ekvalizér, zobrazenie textu pesničky, zmena metadát pesničky, časovač vypnutia, widget, vytváranie a úprava vlastných zoznamov pesničiek alebo zdieľanie hudobných súborov cez Bluetooth, e-mail alebo sociálne siete. Spoločnou vlastnosťou všetkých troch prehrávačov je aj to, že podporujú len dva spôsoby prehrávania – sekvenčný a náhodný, pričom u obidvoch je dodatočne možné nastaviť rôzne možnosti opakovania pesničiek. Všetky tri aplikácie majú aj niečo, čo býva označované ako „chytré zoznamy pesničiek“, no v skutočnosti sa jedná len o štyri pomerne jednoduché zoznamy:

- zoznam nedávno prehratých pesničiek,
- zoznam najnovšie pridaných pesničiek,
- zoznam pesničiek s najväčším počtom prehratí,
- zoznam obľúbených pesničiek, ktorý ale musí naplniť sám užívateľ pomocou tlačidla, ktoré ma zvyčajne tvar srdca.

Podobne vyzerá aj užívateľské rozhranie všetkých troch hudobných prehrávačov (obr. 2.2). To pozostáva zo záložiek zvlášť pre pesničky, albumy, interpretov, hudobné žánre, priečinky obsahujúce hudobné súbory alebo zoznamy pesničiek. Pod týmito záložkami sa zobrazuje obsah práve zvolenej záložky a na dolnej časti obrazovky sa nachádza panel so základnými informáciami o práve prehrávanej pesničke a tlačidlá pre základné ovládanie prehrávania. Po kliknutí na tento panel sa zobrazí obrazovka samotného prehrávania, ktorá typicky zobrazuje detailnejšie informácie o práve prehrávanej pesničke, poskytuje rozšírené možnosti ovládania prehrávania, ako aj ďalšiu dodatočnú funkcionalitu. Všetky tieto aplikácie taktiež umožňujú užívateľovi zmeniť ich vzhľad využitím rôznych farebných tém. Mnohé populárne hudobné prehrávače sa teda od seba líšia len v doplnkových funkciách (napr. zmena rýchlosti prehrávania alebo umožnenie nastavenia pesničky ako zvonenia) a v detailoch užívateľského rozhrania (napr. použité farby alebo veľkosť písma).

Pre ukážku druhej skupiny prehrávačov som zvolil Smart Music Player[4], Power Play – Smart Music Player For Smart People[5] a Pluto Smart Music Player[6]. Prvé dva hudobné prehrávače, ako aj väčšina ostatných prehrávačov z tejto skupiny, avšak neobsahujú žiadnu špeciálnu funkcionalitu, ktorá by z nich robila skutočne chytré hudobné prehrávače. Tieto aplikácie sa nijak výrazne nelíšia od aplikácií z prvej skupiny a to ani vo funkcionalite a ani v užívateľskom rozhraní (obr. 2.2). Výnimkou je Pluto Smart Music Player, ktorý sa snaží naučiť, akú hudbu má prehrávač užívateľovi prehrávať ráno, cez deň alebo večer, pričom sa do úvahy berie aj aktuálne počasie vo zvolenej lokalite. Tento prehrávač má na rozdiel od ostatných prehrávačov z tejto skupiny zaujímavý koncept, no má jednu značnú nevýhodu a tou je časté padanie aplikácie. Na zariadeniach s operačným systémom Android 10 a 11 sa mi ani nepodarilo prehrávanie pesničiek spustiť. Prehrávanie sa mi síce podarilo spustiť na zariadení s operačným systémom Android 6, no aplikácia v tomto prípade spadla vždy, keď som sa snažil prejsť do nastavení. Následkom toho je to, že nie je možné zmeniť lokalitu, pre ktorú sa zisťuje aktuálne počasie, keďže aplikácia nevyužíva pre zistenie polohy systém GPS, ale užívateľ musí zadať svoju polohu do textového poľa v nastaveniach.

[4] https://play.google.com/store/apps/details?id=com.fatihturker.smartmusicplayer
[5] https://play.google.com/store/apps/details?id=power.play
[6] https://play.google.com/store/apps/details?id=com.mp3music.music.player



Obr. 2.2: Horný rad obrázkov zobrazuje niektoré z najpopulárnejších prehrávačov hudby v obchode Google Play – zľava doprava: Music Player – MP3 Player, Pulsar Music Player a Music player. Spodný rad zobrazuje aplikácie, ktoré sa prezentujú ako chytré prehrávače hudby, no nijak špeciálne nevylepšujú prehrávanie hudby alebo prehrávanie hudby cez ne ani nie je možné spustiť – zľava doprava: Smart Music Player, Power Play a Pluto Smart Music Player.

Kapitola 3

Vývoj aplikácií pre Universal Windows Platform

Universal Windows Platform (UWP) je jedna z mnohých platforiem pre vývoj aplikácií a hier pre osobné počítače. Ide o vysoko prispôsobiteľnú platformu, ktorá využíva značkovací jazyk XAML pre oddelenie užívateľského rozhrania (prezentačná vrstva) od kódu (logická vrstva). UWP je vhodným riešením pre desktopové aplikácie, pretože poskytuje automatickú podporu pre vstupné zariadenia akými je klávesnica, myš, dotykové zariadenia alebo dokonca gamepad. Platformu UWP je možné využiť nielen pre vývoj desktopových aplikácií, ale aj pre vývoj aplikácií pre Xbox alebo HoloLens a vďaka platforme Uno je možné spustiť UWP kód aj na operačných systémoch iOS, Android, macOS alebo Linux [2, Choose your Windows app platform[1]]. Prvky užívateľského rozhrania UWP reagujú na veľkosť a DPI obrazovky, na ktorej je aplikácia spustená. Automaticky sa upravuje ich rozloženie a veľkosť, no zároveň je možné navrhnúť užívateľské rozhranie pre špecifickú veľkosť obrazovky alebo pre špecifické zariadenie. Značnou výhodou vývoja na tejto platforme je možnosť zverejnenia aplikácie cez online obchod aplikácií Microsoft Store, odkiaľ si môžu užívatelia vždy stiahnuť aktuálnu verziu aplikácie, ktorá im bude súčasne aj automaticky nainštalovaná. Vývojár tu tiež môže zvoliť, či je jeho aplikácia určená pre všetky dostupné zariadenia alebo len pre skupinu vybraných. Okrem toho tu sú dostupné aj rozličné analýzy, ktoré mu pomáhajú v porozumení užívateľom a vo vylepšovaní aplikácie. UWP aplikácia môže byť vytvorená pomocou jedného z nasledujúcich programovacích jazykov: C#, Visual Basic, C++ alebo JavaScript [2, What's a Universal Windows Platform (UWP) app?[2]].

3.1 Základné stavebné prvky aplikácie a ich životný cyklus

Trieda `Application` zapuzdruje aplikáciu a poskytuje nasledujúce služby:

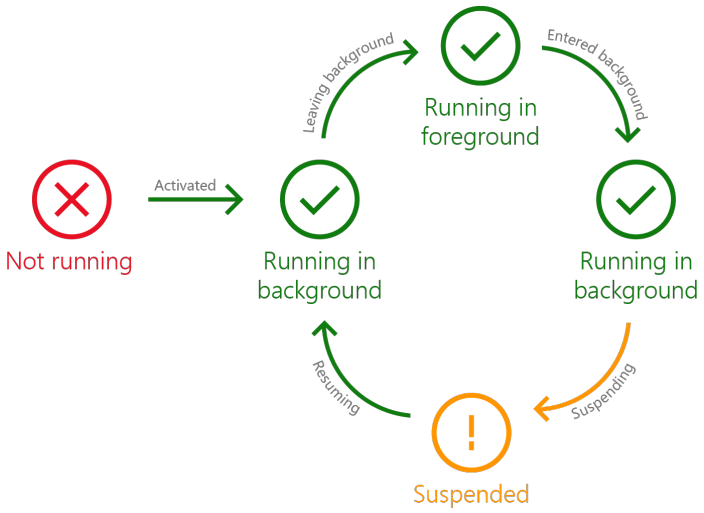
- vstupný bod aplikácie,
- správa životného cyklu aplikácie,
- správa zdrojov aplikácie,
- detekcia neošetrených výnimiek.

[1] https://docs.microsoft.com/en-us/windows/apps/desktop/choose-your-platform

[2] https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide

Visual Studio, primárne vývojové prostredie pre tvorbu UWP aplikácií, automaticky vygeneruje triedu `App`, ktorá dedí z `Application` a ktorá slúži ako vstupný bod, do ktorého je možné pridať inicializačný kód [2, Application Class[3]].

Životný cyklus UWP aplikácie (obr. 3.1) začína a končí v stave, keď aplikácia nebeží. Aplikácia môže byť v tomto stave, pretože nebola ešte spustená odvtedy, ako užívateľ spustil zariadenie alebo odvtedy, ako sa do neho prihlásil. V tomto stave sa môže nachádzať aj z toho dôvodu, že aplikácia bežala, ale následne kvôli nečakanej chybe zlyhala alebo pretože ju vypol sám užívateľ. Aplikácia môže byť buď spustená užívateľom, alebo aktivovaná systémom, čím prechádza do behu na pozadí. Beh na pozadí je predvolený stav, keď je aplikácia spustená, aktivovaná alebo znova obnovená. Tento stav je charakteristický tým, že užívateľské rozhranie aplikácie nie je viditeľné, čo znamená, že do tohto stavu sa môže aplikácia dostať aj jej minimalizovaním alebo prepnutím na inú aplikáciu. Aplikácia prechádza do stavu beh na popredí hneď ako sa užívateľské rozhranie zobrazí alebo keď sa užívateľ sám k aplikácii vráti. Aplikácia prejde do pozastaveného stavu krátko po tom, ako užívateľ minimalizuje aplikáciu alebo prejde na inú bez návratu k pôvodnej do niekoľkých sekúnd a keď súčasne nie je spustená žiadna úloha na pozadí (napríklad prehrávanie multimédií). Aplikácia je taktiež pozastavená pri zobrazení obrazovky uzamknutia. Pri pozastavení sú zastavené vlákna aplikácie a aplikácia a jej dáta sú ponechané v pamäti, pokiaľ to systém umožňuje. Výsledkom toho je úspora energie a väčšie množstvo dostupných zdrojov pre aplikáciu, ktorá je práve na popredí. Pri nedostatku dostupných zdrojov systém pozastavenú aplikáciu ukončí, pričom ale aplikácia stále zostáva viditeľná na paneli úloh. Keď na ňu užívateľ klikne, aplikácia musí obnoviť stav, v ktorom bola pred ukončením, pretože užívateľ si nie je vedomý toho, že bola ukončená systémom. Keď je aplikácia obnovená z pozastaveného stavu, tak sa vracia do behu na pozadí, kde systém obnoví jej stav, čo sa užívateľovi javí ako keby celý čas bežala [2, Windows 10 universal Windows platform (UWP) app lifecycle[4]].



Obr. 3.1: Životný cyklus UWP aplikácie. Kruhy zobrazujú stavy životného cyklu, v ktorých sa môže UWP aplikácia nachádzať. Šípky znázorňujú zmenu medzi jednotlivými stavmi. Obrázok prevzatý z https://docs.microsoft.com/en-us/windows/uwp/launch-resume/app-lifecycle.

[3] https://docs.microsoft.com/en-us/uwp/api/windows.ui.xaml.application?view=winrt-19041
[4] https://docs.microsoft.com/en-us/windows/uwp/launch-resume/app-lifecycle

Aplikáciu je možné chápať ako kolekciu stránok [2, Navigation design basics for Windows apps[5]]. Stránka slúži na prezentáciu obsahu v aplikácii, pričom jednej stránke je možné priradiť práve jeden XAML súbor. Väčšina stránok potom obsahuje viac ako jeden element užívateľského rozhrania. Priamy potomok elementu stránky v XAML je typicky niektorý druh panelu (tabuľka 3.1), ktorý umožňuje to, aby sa stránka skladala z viacerých elementov užívateľského rozhrania. Pri vytváraní stránky sa okrem XAML súboru vytvorí trieda s rovnakým názvom vo vybranom programovacom jazyku, ktorá má na starosti kódovú časť stránky [2, Page Class[6]].

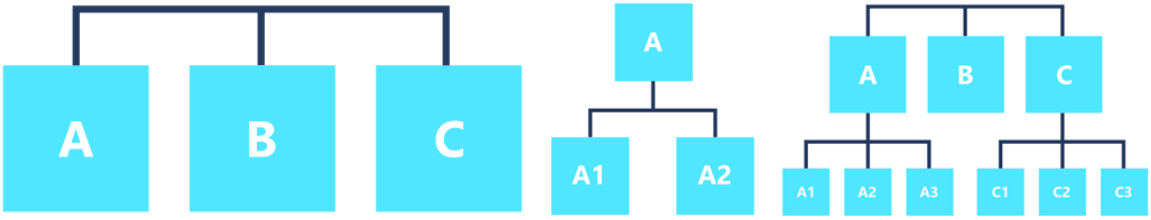
Rámec má na starosti samotné zobrazovanie inštancií stránok, podporu prechodu na nové stránky a udržiavanie histórie navigácie pre podporu prechodu v smere dopredu a späť. Predvoleným nastavením je, že každým prechodom sa vytvára nová inštancia danej stránky a odstraňuje sa inštancia predchádzajúcej stránky. Toto sa deje aj pri prechode smerom späť na naposledy navštívenú stránku alebo keď typ novej stránky je zhodný s typom predchádzajúcej stránky. Aplikácie, ktorých súčasťou sú časté prechody na rovnaké stránky si môžu uložiť inštancie týchto stránok do pamäti cache a znova ich použiť pre zvýšenie efektivity [2, Frame Class[7]]. S navigáciou v aplikácii úzko súvisí jej štruktúra. V UWP sa rozlišujú tri typy štruktúry aplikácie (obr. 3.2), ktoré sú popísané nasledujúcim zoznamom.

- Plochá/Horizontálna štruktúra – stránky existujú vedľa seba a je možné medzi nimi prechádzať v ľubovoľnom poradí. Táto štruktúra je vhodná pre aplikácie s menším počtom stránok, ktoré sú navyše od seba dostatočne rozlíšiteľné.
- Hierarchická štruktúra – stránky sú usporiadané do stromovej štruktúry. Každá detská stránka má práve jedného rodiča, ale jeden rodič môže mať môže mať jednu alebo viacero detských stránok. Hierarchická štruktúra je vhodná na organizáciu komplexného obsahu, ktorý má rozsah na viacero stránok. Nevýhodou je, že čím hlbšia je štruktúra, tak tým viac klikov je potrebné vykonať pre prechod z jednej stránky na inú.
- Kombinovaná štruktúra – kombinácia oboch predchádzajúcich štruktúr. Aplikácia môže používať plochú štruktúru pre stránky na najvyššej úrovni, ktoré môžu byť prehliadané v ľubovoľnom poradí a hierarchické štruktúry pre stránky, ktoré majú zložitejšie vzťahy [2, Navigation design basics for Windows apps[5]].

3.2 Ukladanie dát

UWP aplikácia rozlišuje dva typy dát – dáta aplikácie a dáta užívateľa. Dáta aplikácie sú premenlivé dáta, ktoré sú vytvárané a spravované špecifickou aplikáciou. To zahŕňa dáta o stave aplikácie, nastavenia aplikácie, preferencie užívateľa, referenčný obsah (napríklad slovníkové definície pre aplikáciu slovníka) a podobne. Dáta užívateľa sú dáta, ktoré užívateľ sám vytvára a spravuje, keď danú aplikáciu používa. Tu patria dokumenty, multimediálne súbory, emaily, databázové záznamy s obsahom vytvoreným užívateľom a ďalšie dáta, ktoré chce užívateľ manipulovať alebo prenášať nezávisle na aplikácii.

[5] https://docs.microsoft.com/en-us/windows/uwp/design/basics/navigation-basics
[6] https://docs.microsoft.com/en-us/uwp/api/Windows.UI.Xaml.Controls.Page?view=winrt-19041
[7] https://docs.microsoft.com/en-us/uwp/api/Windows.UI.Xaml.Controls.Frame?view=winrt-19041



Obr. 3.2: Obrázky zobrazujú tri možné typy štruktúr UWP aplikácie. Na obrázku vľavo je zobrazená plochá štruktúra, ktorá sa taktiež nazýva horizontálna. Na strednom obrázku je hierarchická štruktúra a pravý obrázok zobrazuje kombinovanú štruktúru. Obrázky boli prevzaté z https://docs.microsoft.com/en-us/windows/uwp/design/basics/navigation-basics.

3.2.1 Dáta aplikácie

Dáta aplikácie sa ďalej delia na dva druhy – nastavenia a súbory. Nastavenia sa využívajú na ukladanie preferencií užívateľa a pre uloženie informácií o stave aplikácie. Nastavenia dokážu ukladať základné dátové typy ako sú čísla, znaky, reťazce, boolovské hodnoty a prípadne ďalšie jednoduché typy objektov ako je dátum, čas alebo bod. Súbory sa používajú pre ukladanie binárnych dát alebo pre serializáciu vlastných typov. Keď je aplikácia nainštalovaná, systém jej vyčlení vlastné dátové úložisko pre nastavenia a súbory. Systém je taktiež zodpovedný za správu tohto fyzického úložiska a za zabezpečenie toho, aby dáta boli izolované od iných aplikácií a užívateľov. Systém taktiež uchováva obsah týchto dátových úložísk, keď užívateľ spustí inštaláciu aktualizácie aplikácie a rovnako má na starosti odstránenie obsahu týchto dátových úložísk, keď je aplikácia odinštalovaná. V rámci dátového úložiska má každá aplikácia tri systémové koreňové adresáre: jeden pre lokálne súbory, jeden pre tzv. roaming súbory a jeden pre dočasné súbory. Aplikácia môže pridať nové súbory a kontajnery do každého z týchto koreňových adresárov. Namiesto priamej práce s adresármi sa využívajú kontajnery, ktoré pomáhajú organizovať nastavenia a súbory.

Lokálne aplikačné dáta by mali byť použité pre ľubovoľné informácie, ktoré musia byť uchované medzi jednotlivými behmi aplikácie a ktoré súčasne nie sú použiteľné na iných zariadeniach. Veľkosť lokálnych dát nie je nijak obmedzená, a preto by malo byť lokálne dátové úložisko použité hlavne pre tie dáta, ktoré nemá zmysel prenášať do iných zariadení a pre veľké sady dát.

Roaming dáta slúžia na synchronizáciu medzi viacerými zariadeniami. Ak si užívateľ nainštaluje aplikáciu na viacero zariadení, tak systém uchováva dáta aplikácie synchronizované, čím sa redukuje množstvo práce, ktoré musí užívateľ vykonať po inštalácii na ďalšom zariadení. Keď sú roaming dáta aktualizované, systém ich skopíruje na cloudové úložisko a dáta sú následne synchronizované na ostatných zariadeniach, ktoré majú aplikáciu nainštalovanú. Systém limituje veľkosť dát, ktoré sa môžu prenášať, a keď aplikácia dosiahne tento limit, žiadne ďalšie dáta nebudú skopírované na cloudové úložisko a to dovtedy, kým množstvo prenášaných dát nie je znova menšie ako daný limit. Z tohto dôvodu by sa mali roaming dáta používať len na preferencie užívateľa, odkazy a súbory malej veľkosti. Roaming dáta aplikácie sú okrem toho dostupné na cloudovom úložisku len po určitý časový interval. Ak užívateľ nespustí aplikáciu počas tohto intervalu, tak budú jeho roaming dáta z cloudového úložiska odstránené. Pri odinštalovaní aplikácie sú roaming dáta zachované na cloudovom úložisku taktiež len po dobu daného časového intervalu. Roaming dáta avšak nie sú určené pre súčasné použitie na viacerých zariadeniach. Pri vzniku konfliktu počas

synchronizácie z dôvodu, že určité dáta boli pozmenené na dvoch zariadeniach, systém vždy uprednostní tú hodnotu, ktorá bola zapísaná ako posledná. Tento prístup zabezpečí to, že aplikácia bude využívať najaktuálnejšie informácie.

Dočasné dáta aplikácie fungujú ako pamäť cache. Tieto dáta nie sú prenášané do iných zariadení a môžu byť hocikedy odstránené či už systémom alebo užívateľom. Dočasné dáta môžu byť použité pre ukladanie dočasných informácií počas behu aplikácie, pričom tu ale nie je žiadna garancia toho, že tieto dáta pretrvajú aj po skončení behu aplikácie [2, Store and retrieve settings and other app data[8]].

3.2.2 Dáta užívateľa

Dáta užívateľa je možné ukladať buď v súboroch, alebo pomocou databázových záznamov. Aplikácie vyvíjané na platforme UWP môžu pristupovať k niektorým lokáciám súborového systému voľne, no pri iných je nutné deklarovanie špecifických oprávnení alebo prístup cez dialógové okno pre výber súborov. Medzi lokácie, ku ktorým majú prístup všetky UWP aplikácie patrí:

- adresár inštalácie aplikácie – priečinok, do ktorého je aplikácia nainštalovaná v systéme užívateľa,
- adresáre aplikačných dát – priečinky, do ktorých aplikácia ukladá svoje dáta a ktoré sú vytvorené pri inštalácii,
- adresár stiahnutých súborov – aplikácia ako taká má prístup len k tým súborom a priečinkom v adresári užívateľa, do ktorého sa mu predvolene ukladajú stiahnuté súbory, ktoré v ňom sama vytvorila,
- vymeniteľné zariadenia – dodatočne môže aplikácia pristúpiť ku niektorým súborom na pripojenom zariadení. Toto je povolené, ak aplikácia používa rozšírenie na automatické spustenie, keď užívateľ pripojí zariadenie ako fotoaparát alebo USB kľúč do systému. Aplikácia má prístup len k súborom určitých typov, ktoré sú špecifikované v manifeste aplikácie.

Aplikácia môže získať prístup ku všetkým súborom a priečinkom dostupným užívateľovi volaním dialógového okna pre výber súborov. Týmto spôsobom užívateľ svojím výberom sám povolí aplikácii prístup ku zvolenému súboru alebo priečinku. Pomocou tohto dialógového okna môže užívateľ taktiež sám zvoliť, kde bude súbor s jeho dátami uložený, aký bude jeho názov a typ. Keď aplikácia získa súbor alebo priečinok takýmto výberom, tak prístup k tejto položke má len dokým nie je aplikácia ukončená. Raz získaný súbor alebo priečinok je ale možné vložiť do špeciálneho zoznamu, vďaka čomu je možné automaticky získať prístup aj v budúcnosti. Pomocou deklarovania oprávnení v manifeste môže aplikácia získať prístup k ďalším dodatočným lokáciám. K takýmto lokáciám patria priečinky s dokumentami, hudbou, obrázkami, videami alebo všetky priečinky a súbory, ku ktorým ma prístup samotný užívateľ [2, File access permissions[9]].

Ďalším spôsobom uloženia dát na zariadení užívateľa je buď použitie SQLite databázy, alebo priame prepojenie aplikácie s SQL databázovým serverom. Použitie SQLite databázy pre lokálne úložisko má určité výhody – jedná sa o knižnicu s kódom bez žiadnych ďalších

[8] https://docs.microsoft.com/en-us/windows/uwp/design/app-settings/store-and-retrieve-app-data

[9] https://docs.microsoft.com/en-us/windows/uwp/files/file-access-permissions

závislostí a pre jej použitie nie je potrebná žiadna konfigurácia. SQLite nepoužíva žiadny databázový server – klientská aj serverová časť bežia v tom istom procese. SQLite databázu je možné používať zdarma a distribuovať ju spolu s aplikáciou, pričom funguje naprieč mnohými platformami a architektúrami. Pre pridávanie a získavanie dát z SQLite databázy je potrebné vytvoriť triedu pre prístup k dátam, ktorá obsahuje metódu pre inicializáciu takejto databázy. V tej sa definuje názov databázy a vykoná sa príkaz pre jej vytvorenie a uloženie v lokálnom úložisku dát. Ďalej je nutné vytvoriť metódy pre samotné ukladanie a získavanie dát, ako aj užívateľské rozhranie, ktoré užívateľovi umožní tieto činnosti vykonať [2, Use a SQLite database in a UWP app[10]].

Okrem použitia SQLite databázy je možné pripojiť sa priamo ku klasickému SQL databázovému serveru, akým je napríklad MySQL alebo MongoDB, a následne ukladať a získavať dáta z databázy. Dôležitou súčasťou pripojenia k databázovému serveru je vytvorenie reťazca, ktorý obsahuje meno alebo adresu databázového serveru a ďalšie informácie potrebné pre úspešné pripojenie [2, Use a SQL Server database in a UWP app[11]].

3.3 Grafické užívateľské rozhranie

3.3.1 XAML

XAML (Extensible Application Markup Language) je deklaratívny jazyk, pomocou ktorého je možné vytvoriť viditeľné prvky užívateľského rozhrania s využitím deklaratívnych XAML značiek. Využitím jazyka XAML je možné inicializovať objekty a nastaviť ich vlastnosti pomocou jazykovej štruktúry, ktorá zobrazuje hierarchické vzťahy medzi viacerými objektmi. Každému XAML súboru je možné priradiť separátny súbor s kódom, ktorý je schopný reagovať na vzniknuté udalosti a manipulovať s objektmi, ktoré sú v XAML deklarované. Každý projekt má typicky aspoň jeden XAML súbor, ktorý reprezentuje úvodnú stránku aplikácie. Ďalšie XAML súbory môžu deklarovať dodatočné stránky alebo zdroje, akými sú šablóny alebo štýly. Základná syntax jazyka XAML je postavená na jazyku XML – validný XAML súbor musí byť taktiež validný XML súbor. XAML je jazykom, v ktorom sa rozlišujú veľké a malé písmená. Toto je jeden z dôsledkov toho, že jazyk XAML je založený na XML, v ktorom to funguje tak isto. XAML má naviac syntaktické koncepty, ktorým je pridelený iný a úplnejší význam, no zároveň stále platný aj v XML.

Jedným z konceptov jazyka XAML je použitie menných priestorov. Menný priestor je organizačný koncept, ktorý určuje ako sú identifikátory pre programovacie entity interpretované. Použitie menných priestorov umožňuje odlíšiť vývojármi deklarované identifikátory od tých, ktoré boli deklarované frameworkom alebo presadzovať rôzne pravidlá. XAML používa rezervovaný XML atribút `xmlns` pre deklaráciu menných priestorov. Hodnotou tohto atribútu je typicky URI (Uniform Resource Identifier), čo je konvencia zdedená z XML. XAML využíva pre deklarovanie a odlíšenie menných priestorov prefixy, ktoré sa následne používajú v elementoch a atribútoch, čím sa odkazuje na daný menný priestor. Napríklad prefixom „x“ býva označený menný priestor jazyka XAML, ktorý zahŕňa elementy a koncepty definované špecifikáciou jazyka XAML. XAML súbor má navyše takmer vždy v koreňovom elemente deklarovaný predvolený menný priestor. Ten definuje, ktoré elementy je možné deklarovať bez použitia prefixu. Definície menných priestorov sú dedené z rodičovského elementu na detský vrátane všetkých jeho atribútov.

[10] https://docs.microsoft.com/en-us/windows/uwp/data-access/sqlite-databases
[11] https://docs.microsoft.com/en-us/windows/uwp/data-access/sql-server-databases

Rozšírenia značiek sú ďalším často používaným konceptom jazyka XAML. Často reprezentujú istý typ „skratky“, ktorá umožňuje XAML súboru pristúp k určitej hodnote alebo správaniu. Niektoré rozšírenia značiek dokážu napríklad naplniť vlastnosti objektov bežnými reťazcami alebo vnorenými elementmi s cieľom zefektívnenia syntaxe. Použitie rozšírenia značiek je v XAML syntaxi atribútov vyjadrené množinovými zátvorkami, čo pri spracovaní XAML súboru umožňuje zavolanie kódu, ktorý poskytne odpovedajúce správanie pre dané rozšírenie značiek. Rozšírenia značiek môžu mať argumenty, ktoré nasledujú za názvom rozšírenia a sú taktiež obsiahnuté vnútri množinových zátvoriek. Rozšírenia značiek typicky poskytujú návratovú hodnotu, ktorá je vložená na tú pozíciu v strome objektov, kde bolo použité rozšírenie značiek v zdrojom XAML.

XAML je deklaratívny jazyk pre objekty a ich vlastnosti, no taktiež zahŕňa syntax pre zabezpečenie spracovania udalostí objektov v značkách. Názov danej udalosti je špecifikovaný ako názov atribútu v objekte, ktorý udalosť spracováva. Hodnotou tohto atribútu je názov funkcie, ktorá spracuje udalosť v súbore s kódom. Jednoduchým príkladom je tlačidlo podporujúce udalosť stlačenia a definujúce funkciu, ktorá bude mať túto udalosť na starosti: `<Button Click="showUpdatesButton_Click">Show updates</Button>` [2, XAML overview[12]].

3.3.2 Prvky užívateľského rozhrania

Pre zobrazenie a určenie pozície viac ako jedného vizuálneho objektu je najskôr nutné tieto objekty vložiť do panelu alebo iného kontajnerového objektu. XAML poskytuje rozličné triedy panelov, ktoré slúžia ako kontajnery. Panely umožňujú usporiadať prvky užívateľského rozhrania vnútri nich a zvoliť ich pozíciu. Tabuľka 3.1 popisuje základné triedy panelov. Do panelu je možné následne vnoriť ďalší panel alebo priamo viditeľné prvky užívateľského rozhrania. Medzi najpoužívanejšie prvky užívateľského rozhrania patria tlačidlá, prvky pre zobrazenie, zadanie alebo úpravu textu, zoznamy, prvky pre zobrazenie obrázkov a prvky, ktoré užívateľovi umožnia vybrať z viacerých možností. Každý prvok užívateľského rozhrania má vlastnú množinu vlastností, no väčšina prvkov obsahuje vlastnosti, ktoré nastavujú výšku a šírku daného prvku, jeho viditeľnosť alebo zarovnanie v rodičovskom kontajneri [2, Responsive layouts with XAML[13], Layout panels[14]].

3.3.3 Väzba dát

Väzba dát je spôsob, akým užívateľské rozhranie aplikácie zobrazuje dáta a poprípade aj zostáva v synchronizácii s týmito dátami. Väzba dát umožňuje oddeliť dáta ako také od užívateľského rozhrania, čoho výsledkom je jednoduchší konceptuálny model, ako aj lepšia čitateľnosť, testovateľnosť a udržovateľnosť aplikácie. Každá väzba pozostáva z nasledujúcich častí:

- zdroj väzby – zdroj dát pre väzbu – inštancia ľubovoľnej triedy, ktorej hodnoty členov sa majú zobrazovať v užívateľskom rozhraní,
- cieľ väzby – vlastnosť prvku užívateľského rozhrania, ktorá má obsahovať dáta na zobrazenie,

[12] https://docs.microsoft.com/en-us/windows/uwp/xaml-platform/xaml-overview
[13] https://docs.microsoft.com/en-us/windows/uwp/design/layout/layouts-with-xaml
[14] https://docs.microsoft.com/en-us/windows/uwp/design/layout/layout-panels

- objekt väzby – časť, ktorá prenáša hodnoty dát od zdroja k cieľu a prípadne aj od cieľa späť do zdroja.

Väzbu dát je možné použiť pre jednoduché zobrazenie hodnôt zdroja dát, keď je užívateľské rozhranie prvýkrát zobrazené, ale ďalej nereagovať na zmeny týchto hodnôt. Tento režim väzby sa nazýva jednorazový a funguje dobre pre hodnoty, ktoré sa počas behu aplikácie nemenia. Alternatívou je „pozorovanie“ hodnôt a aktualizácia užívateľského rozhrania pri ich zmene. Tento režim väzby sa nazýva jednosmerný a funguje dobre pre dáta, ktoré sú určené len na čítanie. Hodnoty v užívateľskom rozhraní je možné okrem pozorovania aj meniť, pričom zmeny, ktoré užívateľ z užívateľského rozhrania vykoná sa automaticky premietnu do zdroja dát. Tento režim sa nazýva obojsmerný a funguje dobre pre dáta, ktoré nie sú určené len na čítanie, ale aj na zápis.

Na platforme UWP sa rozlišujú dva rôzne druhy väzby dát a to nezávisle na zvolenom režime väzby. Obidva z nich sú typicky deklarované pomocou rozšírenia značiek. Je možné si zvoliť medzi rozšírením značiek `{x:Bind}` alebo `{Binding}`, pričom v aplikácii je možné súčasné použitie oboch. Druh väzby dát `{x:Bind}` je novší a má lepší výkon [2, Data binding in depth[15]].

[15] https://docs.microsoft.com/en-us/windows/uwp/data-binding/data-binding-in-depth

Tabuľka 3.1: Prehľad všetkých základných tried panelov, ktoré slúžia ako kontajnery pre ďalšie prvky užívateľského rozhrania.

Typ panelu	Popis panelu
Relatívny panel	Prvky užívateľského rozhrania sú usporiadané vzhľadom na vzťah k okraju alebo stredu panelu a vzhľadom na vzťah k ostatným prvkom. Pozícia prvkov je určená na základe množstva vlastností, ktoré riadia zarovnanie v celom paneli, ako aj zarovnanie a polohu na základe súrodeneckých prvkov.
Zásobníkový panel	Jednotlivé prvky zásobníkového panelu sú usporiadané do jednej línie, ktorá môže byť orientovaná buď vertikálne, alebo horizontálne a to v tom poradí, v akom boli prvky deklarované.
Mriežka	Prvky sú usporiadané do riadkov a stĺpcov, pričom môžu presahovať niekoľko riadkov a stĺpcov. Počet riadkov a stĺpcov musí byť explicitne definovaný, ináč bude mriežka pozostávať len z jedného riadku a stĺpca. Dodatočne je možné špecifikovať aj ich veľkosť. Pre mriežku je špecifická možnosť použitia symbolu hviezdy namiesto konkrétnej hodnoty pri zadávaní veľkostí. Tým sa dosiahne to, že využiteľný priestor bude rozdelený medzi riadky a stĺpce mriežky podľa váhových proporcií. Mriežka sa dokáže taktiež zmenšiť, zväčšiť alebo preusporiadať, čím reaguje na dostupný vizuálny priestor na obrazovke zariadenia.
Mriežka premennej veľkosti	Prvky sú usporiadané do riadkov alebo stĺpcov, pričom keď je dosiahnutý zadaný maximálny počet prvkov v jednom riadku alebo stĺpci, tak ďalšie prvky sa automaticky vložia do ďalšieho riadku, respektíve stĺpca. Orientácia je predvolene nastavená ako vertikálna. To znamená, že mriežka pridáva prvky zhora nadol, dokým nie je stĺpec plný. Následne sa prejde na ďalší stĺpec. Keď je orientácia nastavená ako horizontálna, tak mriežka pridáva položky zľava doprava a potom prejde na ďalší riadok.
Plátno	Plátno umiestňuje svoje prvky pomocou pevných súradnicových bodov od horného ľavého rohu. Použitie pevných súradníc môže avšak spôsobiť to, že užívateľské rozhranie bude menej adaptívne na zmeny veľkosti okna aplikácie. Objekty v plátne sa môžu prekrývať. Plátno predvolene vykresľuje objekty v tom poradí, v ktorom boli deklarované, takže posledný objekt je vykreslený navrchu. Zmenu tohto poradia je možné dodatočne vykonať nastavením vlastnosti pre súradnicu z. Prvok s najväčšou hodnotou súradnice z je vykreslený ako posledný a tým pádom je vykreslený nad každým iným prvkom, ktorý by ho nejakým spôsobom prekrýval alebo zdieľal rovnaký priestor.

Kapitola 4

Vývoj aplikácií pre Android

Android je open source platforma, ktorá bola navrhnutá predovšetkým pre mobilné zariadenia, akými sú chytré telefóny alebo tablety. Platforma Android pozostáva z operačného systému, ktorý je založený na jadre operačného systému Linux, middlevéru, prívetivého užívateľského rozhrania a aplikácií [3]. Aplikácie pre platformu Android môžu byť napísané pomocou programovacieho jazyka Kotlin, Java alebo C++. Súprava nástrojov pre vývoj softvéru preloží kód vo zvolenom jazyku do archívu s koncovkou `.apk` spolu s ďalšími ľubovoľnými dátami a zdrojovými súbormi. Medzi zdrojové súbory patria obrázky, audio súbory a všetko, čo súvisí s vizuálnou prezentáciou aplikácie. S tou súvisia hlavne XML súbory definujúce užívateľské rozhranie. Používanie zdrojov umožňuje jednoduché zmeny rôznych vlastností aplikácie bez nutnosti úpravy kódu. Poskytnutie sady alternatívnych zdrojov zase umožňuje optimalizáciu aplikácie pre rôzne konfigurácie zariadení, akými sú jazykové nastavenia alebo veľkosti obrazoviek. Jeden APK súbor obsahuje všetok obsah Android aplikácie a jedná sa o súbor, ktorý zariadenia s operačným systémom Android používajú pre inštaláciu [4, Application Fundamentals[1]]. Android aplikácie je možné zverejňovať a sťahovať ako z oficiálneho obchodu aplikácií Google Play, tak aj z obchodov aplikácii výrobcov chytrých telefónov, medzi ktoré patrí Samsung Galaxy Store alebo Huawei AppGallery.

4.1 Základné stavebné prvky aplikácie a ich životný cyklus

Platforma Android rozlišuje štyri základné typy komponentov aplikácie:

1. aktivita,
2. služba,
3. prijímač broadcastu,
4. poskytovateľ obsahu.

Komponenty aplikácie sú základnými stavebnými blokmi Android aplikácie. Každý komponent je vstupným bodom, cez ktorý môže systém alebo užívateľ vstúpiť do aplikácie. Niektoré komponenty závisia na iných, no každý typ má vlastný účel a životný cyklus, ktorý určuje, akým spôsobom je komponent vytvorený a zničený.

Predtým, ako môže systém Android spustiť nejaký komponent, tak systém musí vedieť, že tento komponent skutočne existuje. Na tento účel slúži súbor reprezentujúci manifest

[1] https://developer.android.com/guide/components/fundamentals

aplikácie. Aplikácia musí deklarovať všetky svoje komponenty v tomto súbore, ktorý musí byť umiestnený v koreňovom adresári projektu aplikácie. Manifest slúži na množstvo ďalších účelov, ako napríklad:

- identifikuje ľubovoľné užívateľské povolenia, ktoré aplikácia vyžaduje (ako je prístup na internet alebo ku kontaktom),
- deklaruje hardvérové a softvérové funkcie používané alebo vyžadované aplikáciou (ako je fotoaparát alebo Bluetooth),
- deklaruje knižnice, s ktorými musí byť aplikácia prepojená,
- deklaruje názov a ikonu aplikácie.

4.1.1 Aktivita

Aktivita slúži ako vstupný bod pre interakciu aplikácie s užívateľom, pričom reprezentuje jednu obrazovku s užívateľským rozhraním [4, Application Fundamentals[1]]. Užívateľská skúsenosť pri používaní mobilných aplikácií sa oproti požívaniu desktopových aplikácií líši v tom, že interakcia užívateľa s aplikáciou nemusí vždy začínať z rovnakého miesta. Aktivita je navrhnutá tak, aby umožnila takýto prístup. Keď jedna aplikácia spustí inú, tak spúšťajúca aplikácia v skutočnosti spustí aktivitu druhej aplikácie namiesto spustenia aplikácie ako celku.

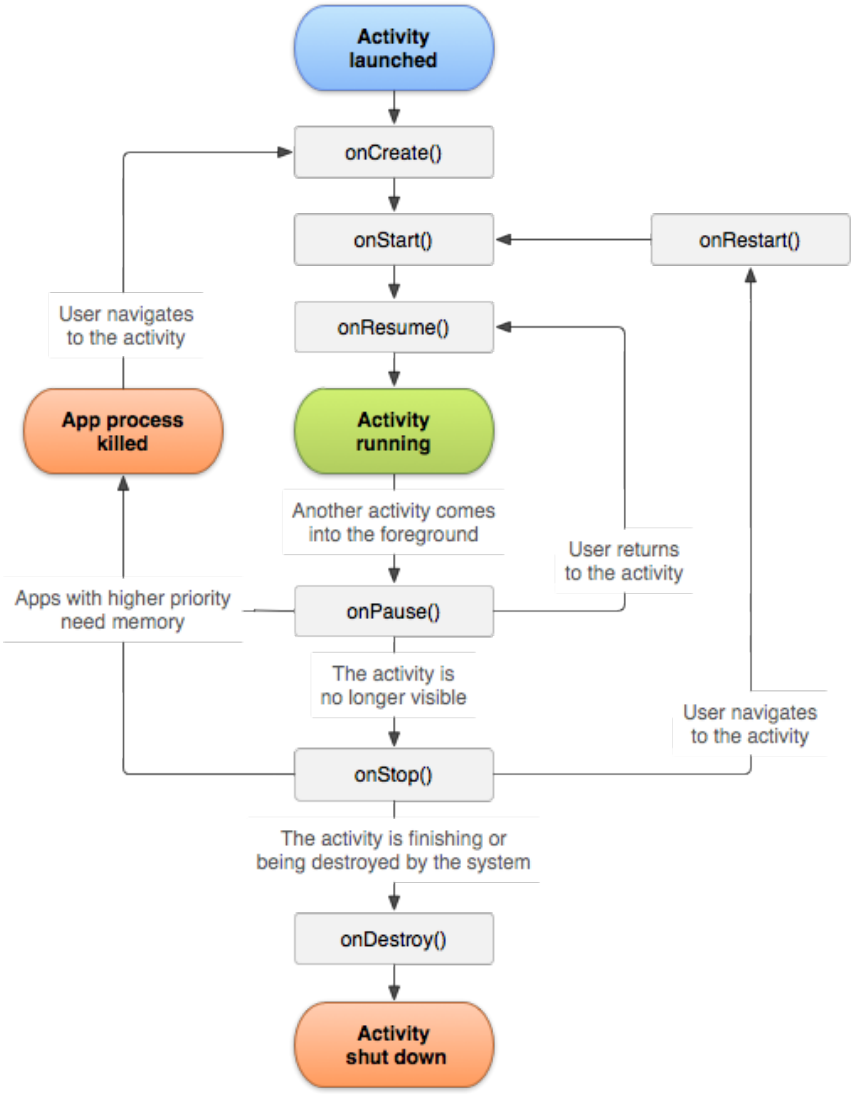
Aktivita poskytuje okno, v ktorom aplikácia vykreslí svoje užívateľské rozhranie. Toto okno zvyčajne vypĺňa celú obrazovku, ale môže byť aj menšie a vznášať sa nad inými oknami. Vo všeobecnosti platí, že jedna aktivita implementuje jednu obrazovku aplikácie. Väčšinu aplikácií tvorí viacero obrazoviek a tým pádom aj viacero aktivít. Jedna z týchto aktivít je typicky špecifikovaná ako hlavná aktivita, ktorej obrazovka sa užívateľovi zobrazí ako prvá pri spustení aplikácie. Každá aktivita môže spustiť inú aktivitu pre vykonanie rozličných akcií. Napriek tomu, že aktivity spolupracujú pre vytvorenie kohéznej užívateľskej skúsenosti, tak každá aktivita je len voľne viazaná na iné aktivity. Závislosti medzi aktivitami aplikácie sú zvyčajne minimálne. Naopak, aktivity jednej aplikácie môžu často spúšťať aktivity patriace iným aplikáciám [4, Introduction to Activities[2]].

Aktivity sú spravované systémom na princípe zásobníka. Keď je nová aktivita spustená, tak sa zvyčajne vloží na vrchol aktuálneho zásobníka a stane sa z nej práve bežiaca aktivita. Predchádzajúca aktivita zostane v zásobníku vždy pod ňou a znova neprejde do popredia, pokiaľ nová aktivita existuje. Na základe životného cyklu aktivity sa rozlišujú 4 stavy, v ktorých sa môže aktivita nachádzať (obr. 4.1).

1. Ak je aktivita na popredí (nachádza sa na vrchu zásobníka a jej obrazovka je viditeľná), tak sa označuje ako práve bežiaca alebo aktívna. Toto je zvyčajne aktivita, s ktorou užívateľ práve interaguje.
2. Ak aktivita stratila fókus, ale stále sa užívateľovi zobrazuje, tak sa hovorí, že je viditeľná. Tento stav môže napríklad nastať vtedy, keď na vrchu zásobníka je aktivita, ktorá je transparentná alebo ktorá nezaberá celú obrazovku.
3. Ak je aktivita úplne prekrytá ďalšou aktivitou, tak je skrytá alebo zastavená. Aktivita v tomto stave nie je viac viditeľná pre užívateľa, a preto býva často ukončená systémom pri nedostatku pamäti.

[2] https://developer.android.com/guide/components/activities/intro-activities

4. Systém môže uvoľniť aktivitu z pamäti a to buď tým, že ju požiada o ukončenie, alebo jednoducho ukončí jej proces. Následne je aktivita zničená a keď má byť znova zobrazená užívateľovi, tak musí byť kompletne reštartovaná a navrátená do svojho predchádzajúceho stavu [4, Activity[3]].



Obr. 4.1: Diagram zobrazujúci životný cyklus aktivity. Obdĺžniky reprezentujú metódy, v ktorých môžu byť implementované rôzne operácie počas toho, ako aktivita mení stavy. Farebné ovály predstavujú hlavné stavy, v ktorých sa môže aktivita nachádzať. Obrázok prevzatý z https://developer.android.com/reference/android/app/Activity.

Porozumením životného cyklu aktivity a jeho správnou implementáciou je možné vyhnúť sa:

- padaniu aplikácie, keď užívateľ príjme telefónny hovor alebo prejde na inú aplikáciu,
- spotrebe zdrojov systému, keď užívateľ aplikáciu nepoužíva,

[3] https://developer.android.com/reference/android/app/Activity

- strate užívateľových dát pri opustení aplikácie a neskoršom návrate,
- padaniu aplikácie alebo strate dát pri zmene orientácie obrazovky [4, Understand the Activity Lifecycle[4]].

4.1.2 Služba

Služba je univerzálny vstupný bod, ktorý zabezpečuje beh aplikácie na pozadí. Služba pritom predvolene beží v hlavnom vlákne hostiteľského procesu – nevytvára si vlastné vlákno a nebeží ani v separátnom procese. Jedná sa o komponent, ktorý beží na pozadí, aby vykonal dlhotrvajúcu operáciu alebo aby vykonal prácu pre vzdialené procesy. Služba neposkytuje žiadne užívateľské rozhranie. Životný cyklus služby (obr. 4.2) začína tým, že iný komponent, akým je aktivita, môže službu spustiť a nechať ju bežať alebo sa k nej pripojí, aby s ňou mohol interagovať. Na základe tohto sa rozlišujú dva sémanticky odlišné typy služieb.

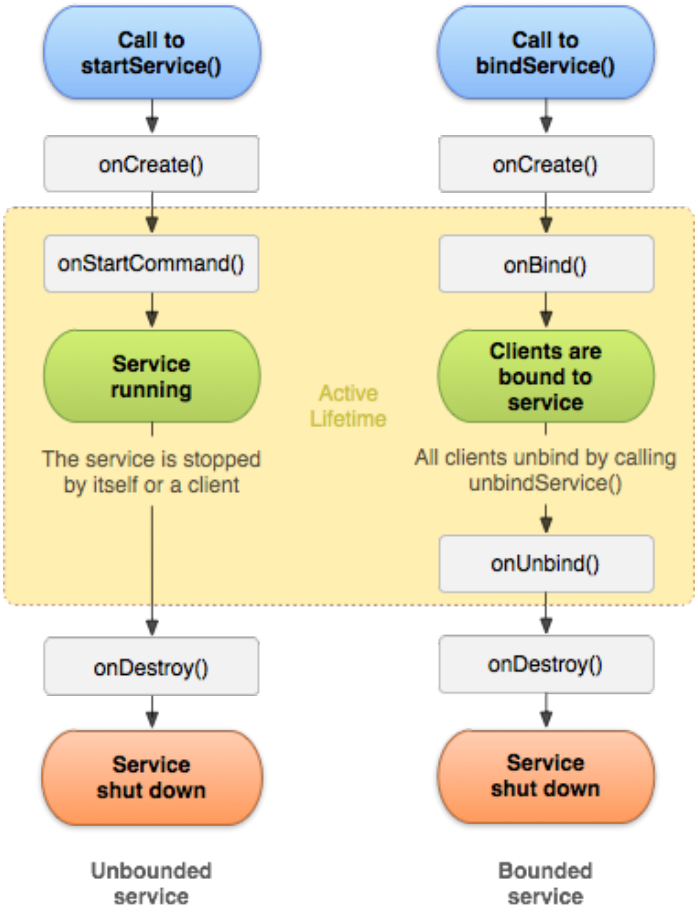
- Spustená služba beží dovtedy, dokým nedokončí svoju prácu. To znamená, že jej životný cyklus je nezávislý na komponente, ktorý ju spustil a služba existuje dovtedy, kým nie je zastavená sama sebou alebo iným komponentom. Zastavenú službu systém zničí. Spustené služby sa ďalej delia na dva odlišné typy.
  - Služba na popredí vykonáva tie operácie, ktorých si je užívateľ vedomý. Tento typ služby musí preto užívateľovi zobrazovať notifikáciu. Notifikácia môže byť odstránená len vtedy, keď je služba pozastavená alebo odstránená z behu na popredí. Služba na popredí beží aj vtedy, keď užívateľ s aplikáciou neinteraguje. Pre zabezpečenie dobrej užívateľskej skúsenosti býva tento typ služby ukončený systémom len výnimočne.
  - Služba na pozadí vykonáva také operácie, ktoré si užívateľ priamo nevšimne a vďaka tomu má systém viac voľnosti v spravovaní jej procesu. Tento typ služby môže byť ukončený systémom pri nedostatku pamäti RAM a neskôr reštartovaný [4, Application Fundamentals[1]].
- Služba sa nazýva viazaná, keď sa k nej pripojí komponent aplikácie volaním špeciálnej metódy. Viazaná služba poskytuje klient-server rozhranie, ktoré umožňuje komponentom interagovať so službou, posielať jej požiadavky, prijímať výsledky a to dokonca aj medzi procesmi. Viazaná služba beží len tak dlho, kým je k nej iný komponent aplikácie pripojený. Viacero komponentov sa môže pripojiť k jednej službe naraz, no keď sa všetky komponenty odpoja, tak je daná služba zničená.

Napriek tomuto rozdeleniu môže služba pracovať oboma spôsobmi – môže byť spustená a bežať neurčite dlhú dobu a zároveň povoľovať prepojenie s inými komponentmi [4, Services overview[5]].

4.1.3 Prijímač broadcastu

Prijímač broadcastu je komponent, ktorý umožňuje systému doručovať aplikácii správy o udalostiach a ktorý umožňuje aplikácii reagovať na systémové oznámenia. Pretože sú prijímače broadcastu ďalším vstupným bodom do aplikácie, tak systém môže doručovať správy aj tým aplikáciám, ktoré nie sú práve spustené. Mnoho broadcastových správ vytvára

[4] https://developer.android.com/guide/components/activities/activity-lifecycle
[5] https://developer.android.com/guide/components/services



Obr. 4.2: Diagramy zobrazujúce možné životné cykly služby. Diagram vľavo zobrazuje životný cyklus spustenej služby a diagram vpravo zobrazuje životný cyklus viazanej služby. Napriek tomu, že obrázok oddeľuje tieto dva typy služieb, tak služba môže byť spustená a súčasne mať pripojené komponenty. Obrázok prevzatý z https://developer.android.com/guide/components/services.

samotný systém. Príkladom môže byť správa oznamujúca, že sa vypla obrazovka alebo že batéria je takmer vybitá. Aplikácia môže taktiež vytvárať vlastné správy a to napríklad kvôli tomu, aby dala ostatným aplikáciám vedieť, že sťahovanie dát bolo dokončené a dáta môžu samy použiť. Prijímače broadcastu môžu zobrazovať notifikáciu, ktorou upozornia užívateľa na to, že nastala nejaká udalosť a to aj napriek tomu, že zvyčajne tieto komponenty užívateľské rozhranie nezobrazujú. Častejšie sú ale prijímače broadcastu len bránou k iným komponentom a sú určené na vykonávanie minimálneho množstva práce [4, Application Fundamentals[1]].

Aplikácie sa môžu zaregistrovať pre získavanie špecifických správ. Keď je správa poslaná, systém automaticky odošle správu tým aplikáciám, ktoré sa prihlásili na odber daného typu správy. Aplikácie môžu prijímať broadcastové správy dvoma spôsobmi:

1. cez prijímače deklarované v manifeste – takýto typ prijímača je zaregistrovaný, keď je aplikácia nainštalovaná s tým, že z prijímača sa stane samostatný vstupný bod do aplikácie a systém môže spustiť aplikáciu a doručiť správu aj vtedy, keď aplikácia práve nebeží,

2. cez prijímače zaregistrované v rámci daného kontextu – tieto typy prijímačov sú schopné prijímať správy tak dlho, pokým je ich registračný kontext platný. Ak bol prijímač zaregistrovaný v rámci kontextu aktivity, tak správy je možné prijímať dovtedy, dokým nie je aktivita zničená. Ak bol prijímač zaregistrovaný v rámci kontextu aplikácie, tak správy je možné prijímať pokiaľ je aplikácia spustená. Registráciu týchto prijímačov je možné pomocou kódu zrušiť a aplikácia následne nebude prijímať daný typ správ [4, Broadcasts overview[6]].

4.1.4 Poskytovateľ obsahu

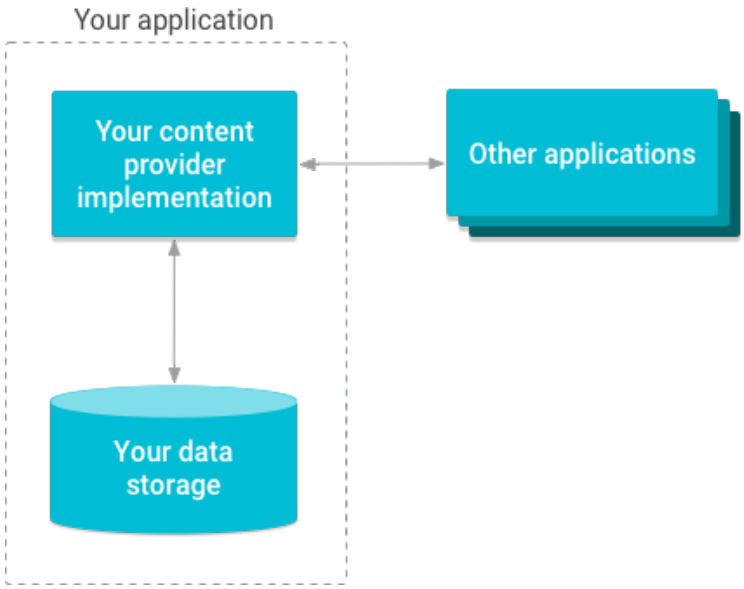
Poskytovateľ obsahu spravuje zdieľanú množinu aplikačných dát, ktorá môže byť uložená v súborovom systéme, SQLite databáze, na internete alebo na ľubovoľnom inom perzistentnom úložisku, ku ktorému má aplikácia prístup. Pomocou poskytovateľa obsahu môžu ostatné aplikácie získavať alebo modifikovať dáta ďalšej aplikácie (obr. 4.3), ak to poskytovateľ obsahu povolí. Ako príklad je možné uviesť poskytovateľa obsahu, ktorý spravuje informácie o kontaktoch v telefóne užívateľa. Každá aplikácia so správnymi povoleniami môže využiť tohto poskytovateľa obsahu na čítanie a zápis informácií o určitej osobe v kontaktoch. Pre systém je poskytovateľ obsahu vstupný bod aplikácie pre zverejnenie pomenovaných dátových položiek, ktoré sú identifikované schémou URI. Aplikácia sa teda môže sama rozhodnúť, ako chce mapovať svoje dáta na menný priestor URI a ako predá tieto URI ostatným entitám, ktoré ich môžu použiť na prístup k dátam. Poskytovateľ obsahu môže byť taktiež užitočný pre čítanie a zápis dát, ktoré nie sú zdieľané, ale privátne pre danú aplikáciu [4, Application Fundamentals[1]]. Poskytovateľ obsahu prezentuje dáta externým aplikáciám ako jednu alebo viacero tabuliek, ktoré sú podobné tabuľkám relačnej databázy. Jeden riadok tabuľky reprezentuje jednu inštanciu určitého dátového typu a každý stĺpec riadku reprezentuje jednotlivý údaj danej inštancie [4, Content provider basics[7]]. Platforma Android navyše poskytuje poskytovateľov obsahu, ktorý spravujú dáta ako audio súbory, video súbory a obrázky. S určitými obmedzeniami sú títo poskytovatelia obsahu prístupní všetkým aplikáciám na platforme Android [4, Content providers[8]].

4.2 Ukladanie dát

Android používa súborový systém, ktorý sa podobá na súborové systémy založené na diskoch na iných platformách. Systém poskytuje niekoľko možností pre ukladanie dát aplikácie:

- úložisko aplikácie – ukladá súbory, ktoré sú určené len pre použitie danou aplikáciou a to do vyhradených adresárov na internom alebo externom úložisku, pričom interné úložisko by malo byť použité pre uloženie citlivých informácií,
- zdieľané úložisko – ukladá súbory, ktoré má aplikácia v úmysle zdieľať s inými aplikáciami – to zahŕňa multimediálne súbory, dokumenty a ostatné súbory,
- preferencie – ukladajú privátne primitívne dáta v dvojiciach kľúč-hodnota,
- databázy – ukladajú štruktúrované dáta v privátnej databáze.

[6] https://developer.android.com/guide/components/broadcasts
[7] https://developer.android.com/guide/topics/providers/content-provider-basics
[8] https://developer.android.com/guide/topics/providers/content-providers



Obr. 4.3: Obrázok zobrazuje to, že pomocou poskytovateľa obsahu môžu ostatné aplikácie získavať alebo modifikovať dáta ďalšej aplikácie bezpečným spôsobom. Obrázok prevzatý z https://developer.android.com/guide/topics/providers/content-providers.

Platforma Android poskytuje dva typy fyzického úložiska: interné úložisko a externé úložisko. Na väčšine zariadení má interné úložisko menšiu veľkosť ako externé úložisko. Interné úložisko má taktiež obmedzený priestor pre ukladanie dát špecifických pre aplikáciu. Na druhej strane je interné úložisko vždy dostupné na všetkých zariadeniach a to z neho robí spoľahlivejšie miesto pre uloženie dát, na ktorých aplikácia závisí. Aplikácie samotné sa predvolene ukladajú na toto úložisko. Ďalšou výhodou interného úložiska je, že dáta sú pred užívateľmi skryté. Odstrániteľné médiá, akými sú SD karty, sú v súborovom systéme považované za časť externého úložiska.

Aplikácie na starších verziách systému Android museli deklarovať povolenie pre čítanie a zápis súborov nachádzajúcich sa mimo adresára aplikácie na externom úložisku. Pre určenie toho, či aplikácia bude mať schopnosť čítať a zapisovať do daného súboru sa na novších verziách systému Android spolieha viac na účel súboru ako na jeho umiestnenie. Tento model úložiska založený na účele súboru zlepšuje súkromie užívateľov, pretože aplikáciám je povolený prístup len k tým častiam súborového systému zariadenia, ktoré skutočne používajú. Všetky súbory aplikácie nachádzajúce sa v úložisku aplikácie, preferenciách a databázach sú pri odinštalovaní aplikácie zo zariadenia odstránené. Jediným spôsobom ako uchovať súbory vytvorené aplikáciou aj po jej odinštalovaní je ich uloženie na zdieľanom úložisku [4, Data and file storage overview[9]].

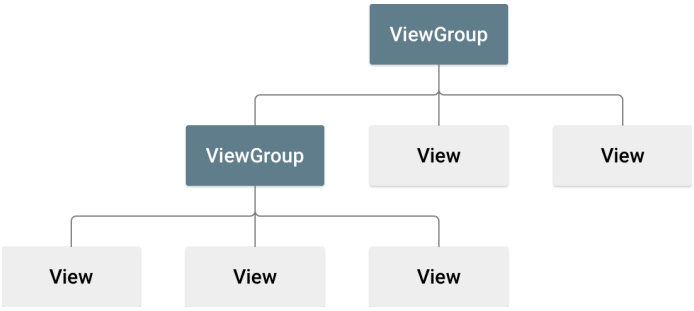
4.3 Grafické užívateľské rozhranie

Užívateľské rozhranie aplikácie je všetko, čo môže užívateľ vidieť a s čím môže interagovať. Android poskytuje množstvo predpripravených komponentov užívateľského rozhrania, akými sú objekty štruktúrovaného rozloženia a ovládacie prvky užívateľského rozhrania, ktoré umožňujú zostaviť grafické užívateľské rozhranie aplikácie. Platforma Android tak-

[9] https://developer.android.com/training/data-storage

tiež poskytuje ďalšie moduly užívateľského rozhrania pre špeciálne rozhrania, akými sú dialógové okná, upozornenia a menu aplikácie [4, User Interface & Navigation[10]].

Trieda `View` reprezentuje základný stavebný blok pre komponenty užívateľského rozhrania. Každý objekt tejto triedy zaberá obdĺžnikovú plochu na obrazovke a zároveň je zodpovedný za vykreslenie daného prvku užívateľského rozhrania, ako aj za spracovanie jeho udalostí. Trieda `ViewGroup`, ktorá je podtriedou triedy `View`, je základnou triedou pre rozloženia. Rozloženia sú neviditeľné kontajnery, ktoré môžu obsahovať ďalšie objekty triedy `View` alebo `ViewGroup` a ktoré definujú ich rozmiestnenie. Všetky prvky užívateľského rozhrania sú usporiadané do jedinej stromovej štruktúry (obr. 4.4). Pridávanie prvkov užívateľského rozhrania je možné buď z kódu aplikácie, alebo špecifikovaním stromovej štruktúry prvkov užívateľského rozhrania v jednom alebo viacerých XML súboroch [4, View[11]]. Priradenie XML súboru k aktivite sa vykonáva z kódu, typicky pri jej spustení. Každý XML súbor musí obsahovať práve jeden koreňový element, pričom ten musí byť objektom triedy `View` alebo `ViewGroup`. Deklarovanie užívateľského rozhrania v XML umožňuje oddeliť prezentačnú časť aplikácie od kódu, ktorý riadi jej správanie. Používanie XML súborov taktiež umožňuje jednoducho poskytnúť rôzne rozmiestnenia prvkov užívateľského rozhrania pre rôzne veľkosti obrazoviek a pre rôzne orientácie, v ktorých sa môže zariadenie nachádzať. Pomocou kódu je zase možné meniť užívateľské rozhranie aj za behu aplikácie.



Obr. 4.4: Obrázok zobrazuje príklad stromovej štruktúry, ktorá definuje rozmiestnenie prvkov užívateľského rozhrania. Obrázok prevzatý z https://developer.android.com/guide/topics/ui/declaring-layout.

Každý objekt typu `View` a `ViewGroup` podporuje svoju vlastnú sadu XML atribútov. Niektoré atribúty sú špecifické pre daný prvok užívateľského rozhrania (napríklad veľkosť alebo farba textu pre textové pole) a iné sú spoločné pre všetky prvky. Jedným z takýchto atribútov, ktorý majú všetky prvky užívateľského rozhrania, je atribút `id`. Tento atribút reprezentuje celočíselnú identifikáciu, pomocou ktorej je možné identifikovať prvok užívateľského rozhrania v stromovej štruktúre. Keď sa aplikácia preloží, tak sa na túto identifikáciu síce odkazuje pomocou celého čísla, no atribút `id` sa v XML súbore typicky zadáva v podobe textového reťazca. Vďaka tomuto textovému reťazcu je následne možné pristupovať k prvku užívateľského rozhrania aj z kódu. Medzi ďalšie spoločné atribúty patrí výška a šírka alebo nastavenie okrajov. Výšku a šírku je možné nastaviť tak, aby prvok užívateľského rozhrania:

- zaberal len toľko miesta, koľko potrebuje na základe svojho obsahu,
- zaberal čo najviac miesta, tak ako to jeho rodičovský prvok dovoľuje,

[10] https://developer.android.com/guide/topics/ui
[11] https://developer.android.com/reference/android/view/View

- mal špecifickú veľkosť zadanú v relatívnych alebo konkrétnych jednotkách [4, Layouts[12]].

Každý prvok užívateľského rozhrania rovnako podporuje svoju vlastnú sadu udalostí, na ktoré dokáže reagovať. Všetky prvky užívateľského rozhrania dokážu reagovať na získanie a stratu fókusu a pre tlačidlá je zase typická udalosť krátkeho alebo dlhého stlačenia. Pre zachytenie a spracovanie danej udalosti je nutné z kódu nastaviť danému prvku špeciálny objekt (tzv. `listener`), ktorý bude upozornený, keď udalosť nastane [4, View[11]].

Platforma Android poskytuje množstvo rôznych typov rozložení. Medzi bežné a často používané patrí:

- lineárne rozloženie – organizuje svoje detské prvky do jediného horizontálneho alebo vertikálneho riadku, pričom ak jeho veľkosť presiahne veľkosť obrazovky, tak sa na kraji obrazovky automaticky zobrazí posuvná lišta,
- relatívne rozloženie – umožňuje špecifikovať umiestnenie detských prvkov na základe vzťahov medzi sebou (napr. detský prvok A je naľavo od detského prvku B) alebo na základe vzťahu k rodičovskému prvku (napr. detský prvok zaberá hornú časť svojho rodičovského prvku) [4, Layouts[12]],
- rámcové rozloženie – všetky detské prvky sú vykreslené na princípe zásobníka, pričom sa môžu navzájom prekrývať,
- rozloženie mriežky – umiestňuje prvky užívateľského rozhrania do riadkov a stĺpcov [5].

Keď je obsah rozloženia dynamický alebo ho nie je možné vopred určiť, tak je možné použiť adaptér, ktorý naplní rozloženie za behu aplikácie. Adaptér sa správa ako prostredník medzi zdrojom dát a rozložením – adaptér získa dáta zo zdroja, akým môže byť pole alebo databáza a následne prevedie každú položku zo zdroja na prvok užívateľského rozhrania, ktorý môže byť ďalej vložený do rozloženia. Adaptér najčastejšie pracuje so zoznamami, ktoré bývajú usporiadané do jedného stĺpca alebo mriežky [4, Layouts[12]].

[12] https://developer.android.com/guide/topics/ui/declaring-layout

Kapitola 5

Návrh hudobného prehrávača

Napriek tomu, že obidve verzie hudobného prehrávača sú implementované na rôznych platformách, tak obe musia zdieľať rovnakú základnú funkcionalitu. Toto je obzvlášť dôležité pre užívateľskú skúsenosť, ktorou sa rozumie celkový dojem vytvorený interakciami a percepciami, ktoré užívateľ má pri používaní aplikácie. Tento pojem býva často spájaný s návrhom užívateľského rozhrania, no užívateľská skúsenosť zahŕňa okrem užívateľského rozhrania aj užívateľove ciele a motivácie, kontext použitia aplikácie a hardvér [6]. Bez ohľadu na to, ktorú verziu hudobného prehrávača začal užívateľ používať ako prvú, tak pri prvom stretnutí s druhou verziou prehrávača by mu malo byť jasné, na akom princípe prehrávač funguje a to aj napriek zmenám v grafickom užívateľskom rozhraní medzi verziami. Táto kapitola približuje tie najdôležitejšie princípy fungovania obidvoch verzií hudobného prehrávača.

5.1 Algoritmus výpočtu skóre pesničky

Jednou z prvých akcií, ktoré prehrávač vykoná pri svojom spustení, je načítanie hudobných súborov zo zariadenia. Táto akcia je detailne popísaná v kapitole 6.2. Prehrávač následne jednotlivo prejde všetkými získanými hudobnými súbormi, vyberie si z nich tie informácie, ktoré potrebuje pre svoju činnosť a vytvorí zoznam pesničiek. Ten sa podľa preferencií užívateľa zoradí podľa názvu pesničky, hodnoty skóre alebo dátumu pridania pesničky do zariadenia a zobrazí sa užívateľovi. Najdôležitejšou informáciou z hudobného súboru je jeho identifikátor, na základe ktorého dokáže prehrávač jednoznačne určiť, aká pesnička sa práve prehráva, aké pesničky sa prehrali pred touto pesničkou a aké pesničky sa budú prehrávať po tejto pesničke a to dokonca aj vtedy, keď sa zmení obsah zoznamu pesničiek, napríklad pri vyhľadávaní. Medzi ďalšie dôležité informácie, ktoré sa získavajú zo súboru s hudbou patrí: názov pesničky a jej autor, názov albumu, na ktorom sa pesnička nachádza, dĺžka trvania pesničky, dátum pridania hudobného súboru do zariadenia a cesta k tomuto súboru.

Počas získavania všetkých týchto informácií o pesničke sa taktiež vypočíta jej skóre, na ktorom je postavený celý princíp fungovania hudobného prehrávača. Základom výpočtu skóre pesničky sú hodnotenia, ktoré pesnička získa na základe akcií užívateľa počas používania prehrávača a počúvania hudby. Všetky typy hodnotení, ktoré môže pesnička získať sú bližšie popísané v tabuľke 5.1. Hodnotenia každej pesničky sú uložené v záznamoch, ktoré majú tvar kľúč-hodnota, kde kľúčom je identifikátor hudobného súboru a hodnotou sú samotné hodnotenia udelené danej pesničke v podobe textového reťazca, pričom jednotlivé hodnotenia sú od seba oddelené čiarkou. Ak bola pesnička pridaná do zariadenia len nedávno, tak sa jej práve tu, pri výpočte skóre, vytvorí nový záznam a priradí sa jej

inicializačné hodnotenie. Algoritmus výpočtu skóre si teda získa všetky hodnotenia danej pesničky na základe jej identifikátoru a postupne ich začne prechádzať, čím súčasne začína výpočet skóre.

Každému hodnoteniu sa priradí konkrétna číselná hodnota na základe toho, o aký typ hodnotenia sa jedná. Špeciálnym prípadom je hodnotenie za prechod na nasledujúcu pesničku – číselná hodnota tu nezávisí len na samotnom type hodnotenia, ale hlavne na tom, aká časť pesničky sa stihla prehrať. V prípade, keď sa stihlo prehrať 85 % pesničky alebo viac, tak číselná hodnota hodnotenia bude 0. Dôvodom pridelenia nulovej hodnoty v takomto prípade je to, že užívateľ si už vypočul väčšinu pesničky a pesnička mohla mať na konci už len nejakú inštrumentálnu alebo tichú časť. Ak sa pri prechode na nasledujúcu pesničku prehralo menej ako 85 % pôvodnej pesničky, tak sa číselná hodnota vypočíta pomocou nasledujúceho vzorca:

$$v = -\left(2 - 2 \cdot \frac{p}{100}\right), \tag{5.1}$$

pričom v je konkrétna číselná hodnota za prechod na nasledujúcu pesničku a p je počet prehratých percent. Podobným prípadom je typ hodnotenia, ktorým sa hodnota skóre upraví presne o takú hodnotu, ktorú užívateľ sám zadá cez užívateľské rozhranie. Tento typ hodnotenia teda závisí čisto na užívateľovi a tým pádom môže byť jeho číselná hodnota rôzna pre každú pesničku. Ak sa v reťazci daný typ hodnotenia za sebou bezprostredne opakuje, tak sa číselná hodnota daného hodnotenia upraví podľa nasledujúceho vzorca:

$$n = o \cdot \left(1 + \frac{c}{10}\right), \tag{5.2}$$

kde n je nová číselná hodnota, o je pôvodná číselná hodnota daného hodnotenia a c je počet opakovaní. Ak má hodnotenie „X“ číselnú hodnotu 1 a v reťazci s hodnoteniami existuje podreťazec „Y,X,X,X,Z“, tak hodnota prvého „X“ bude 1, druhé „X“ bude mať hodnotu 1,1 a tretie 1,2. Následne sa číselná hodnota každého hodnotenia sčíta do sumy. Táto suma je obzvlášť dôležitá pre užívateľa, pretože jej výsledná hodnota je to, čo užívateľ vidí ako skóre pri každej pesničke.

Okrem samotných hodnotení ovplyvňujú reálnu hodnotu skóre pesničky ďalšie faktory so špecifickým významom pre fungovanie prehrávača. Veľkosť zmeny spôsobená týmito faktormi na hodnotu skóre je premenlivá a mení sa s každým prehratím danej pesničky. Tieto zmeny môžu byť drastické, a preto sa užívateľovi nezobrazujú. Reálna hodnota skóre pesničky má ale dôležitý význam, pretože sa používa pri výbere pesničky, ktoré je popísané v nasledujúcej podkapitole.

Čím viac sa prehrávač používa, tým viac hodnotení pesničky získavajú. Po nejakej dobe sa môže stať, že užívateľ prestane mať rád pesničku, ktorú kedysi počúval často. Aj keď ju začne vždy na začiatku preskakovať, tak jej skóre môže byť nejakú dobu stále vysoké, čo má za následok, že sa užívateľovi často prehráva, aj keď sa mu už nepáči. To isté platí aj naopak – keď užívateľ začne mať rád pesničku, ktorú kedysi preskakoval, tak jej môže dosť dlho trvať, kým sa nezačne prehrávať častejšie, keďže jej skóre ovplyvňujú staré negatívne hodnotenia tak isto ako tie nové kladné. Prvým zo spomínaných faktorov je teda váha hodnotenia, ktorou prehrávač rieši tento problém tým, že mení číselnú hodnotu hodnotenia na základe jeho pozície v reťazci. Predtým, ako sa hodnotenie pridá do sumy, ktorá bude tvoriť reálne skóre pesničky, tak sa jeho hodnota upraví podľa tohto vzorca:

$$n = o \cdot 0{,}55 \cdot \log_{0{,}25}(p) + 3, \tag{5.3}$$

kde n je nová číselná hodnota, o je pôvodná číselná hodnota daného hodnotenia a p je pozícia hodnotenia od konca textového reťazca s hodnoteniami. Táto úprava sa nevzťahuje

Tabuľka 5.1: Prehľad všetkých možných typov hodnotení, ktoré môžu byť uložené v zázname pesničky vrátane odôvodnenia ich názvu a bližšieho popisu. Symbol hviezdy (*) na konci textového popisu hodnotenia znamená, že tento typ hodnotenia sa môže v zázname pesničky vyskytovať len raz.

Typ hodnotenia	Bližší popis hodnotenia
C (Create)	inicializačné hodnotenie pridelené pri vytvorení záznamu *
F (Full)	užívateľ si vypočul celú pesničku
F+ (Full bonus)	užívateľ si vypočul celú pesničku prvýkrát po tom, ako prechádzal na nasledujúce pesničky
NX (Next)	užívateľ prešiel na nasledujúcu pesničku, pričom X označuje konkrétnu pozíciu prehrávania v percentách v čase prechodu
P (Previous)	užívateľ sa vrátil na predchádzajúcu pesničku
S (Selected)	užívateľ spustil prehrávanie pesničky kliknutím na jej položku v zozname
UX (User)	užívateľ explicitne zmenil hodnotu skóre pesničky, pričom X označuje konkrétnu hodnotu zmeny skóre; toto hodnotenie sa vždy ukladá na začiatok záznamu pre zabezpečenie jeho rýchlej a jednoduchej detekcie *
SW (Swipe)	užívateľ pridal pieseň do zoznamu nasledujúcich pesničiek
SB (SeekBar)	užívateľ presunul pozíciu prehrávania na začiatok pesničky, pričom rozdiel od začiatočnej pozície posunu do konečnej pozície bol minimálne 25 % dĺžky pesničky
V+ (Volume up)	užívateľ zvýšil hlasitosť aspoň raz počas prehrávania pesničky
B (Blocked)	užívateľ pridal pieseň do zoznamu blokovaných pesničiek *

na prvé dve hodnotenia zo začiatku reťazca, pretože na prvých dvoch pozíciách sa môže vyskytovať hodnotenie typu „UX“ a „C“, ktorých hodnota by mala byť nemenná. Táto zmena hodnoty hodnotenia sa užívateľom nezobrazuje, pretože po udelení kladného hodnotenia užívateľ očakáva, že sa skóre pesničky zväčší, no pri tejto úprave to tak vždy byť nemusí. Zobrazované skóre by sa mohlo naopak aj zmenšiť, čo by mohlo užívateľa zmiasť. Takáto situácia nastáva hlavne v prípade, keď sú na konci reťazca hodnotenia s vysokou kladnou číselnou hodnotou a keď sa za ne pridá hodnotenie s nízkou kladnou číselnou hodnotou. Toto najnovšie hodnotenie sa síce prenásobí najväčšou váhou, ale všetky predchádzajúce hodnotenia budú súčasne vynásobené nižšími váhami, čím vznikne rozdiel a celkové skóre sa zmenší aj po udelení kladného hodnotenia.

Druhým z týchto faktorov je počet prehratí pesničky. Počet prehratí pesničky je daný počtom hodnotení za vypočutie celej pesničky a počtom hodnotení za prechod na ďalšiu pesničku (typy hodnotenia „F“ a „NX“) v textovom reťazci s hodnoteniami. Pomocou počtu prehratí je možné jednoducho identifikovať nové pesničky v zariadení užívateľa a dočasne zvýšiť skóre týchto pesničiek tak, aby sa prehrávali častejšie a čo najskôr získali hodnotenia priamo od užívateľa. K reálnemu skóre pesničky sa na základe počtu prehratí pričíta hodnota vyjadrená nasledujúcim vzorcom:

$$p = 0{,}8^c \cdot 3 \cdot a, \tag{5.4}$$

kde p je hodnota, o ktorú sa zvýši reálne skóre pesničky na základe počtu prehratí, c je samotný počet prehratí danej pesničky a a je priemerné skóre. Priemerné skóre je dané sumou kladných hodnôt užívateľského skóre všetkých nezablokovaných pesničiek, ktorá je

vydelená počtom pesničiek v užívateľovom zariadení. Ak by bola hodnota priemerného skóre menšia ako desať, tak sa priemerné skóre nastaví práve na túto hodnotu. Pri výpočte priemerného skóre sa používa suma tých skóre, ktoré vidí užívateľ, pretože toto skóre je stabilnejšie ako reálne skóre pesničky. Pri použití reálneho skóre pesničky na výpočet priemerného skóre sa môže stať to, že hodnota priemerného skóre bude exponenciálne narastať s každým zapnutím prehrávača a to aj bez toho, aby boli udelené nové hodnotenia.

Ďalším takýmto faktorom je výskyt pesničky v zozname predchádzajúcich pesničiek. Ak sa daná pesnička v tomto zozname nachádza, tak je jej skóre vynásobené hodnotou získanou týmto vzorcom:

$$l = \frac{i}{\sqrt{12^2 + i^2}}, \tag{5.5}$$

pričom l je hodnota, ktorou je reálne skóre pesničky vynásobené a i je index pozície pesničky v zozname predchádzajúcich pesničiek. Ak sa pesnička v tomto zozname vyskytuje viackrát, tak sa vždy použije index s najmenšou hodnotou. Po tom ako je pesnička prehratá, prípadne preskočená, tak sa vloží na začiatok zoznamu prechádzajúcich pesničiek. To znamená, že v takejto situácii bude reálne skóre pesničky vynulované, keďže sa zoznamy indexujú od nuly. Tým, že sa prehrávajú ďalšie pesničky a index v zozname narastá, tak aj reálne skóre danej pesničky sa eventuálne vráti na svoju pôvodnú hodnotu. Tomu dopomáha aj to, že pri zapnutí prehrávača je tento zoznam vždy vytvorený nanovo a je do neho vložených len pätnásť pesničiek, ktoré sa nachádzali na začiatku zoznamu pri vypnutí aplikácie. Tento princíp slúži k tomu, aby nedochádzalo k neustálemu opakovaniu rovnakých pesničiek a aby dostali šancu na prehranie aj pesničky s nižším skóre.

Posledným faktorom je zvolený mód prehrávania. Užívateľ si môže v nastaveniach aplikácie zvoliť jeden z troch módov prehrávania: predvolený mód, mód pre najnovšie pesničky a mód pre najobľúbenejšie pesničky. Predvolený mód hodnotu reálneho skóre pesničky nijak viac neupravuje a reprezentuje základné správanie prehrávača. Predvolený mód už v sebe má zakomponované isté zvýhodnenie pre nové pesničky, no ak užívateľ chce, aby sa najnovšie pesničky prehrávali omnoho častejšie oproti tým starším, tak môže v nastaveniach zvoliť mód pre najobľúbenejšie pesničky. Tento mód upravuje reálne skóre pesničky nasledujúcim spôsobom:

$$n = 0{,}3^i \cdot o^2 \cdot 0{,}85^c, \tag{5.6}$$

kde n je nová hodnota reálneho skóre pesničky, i je index dátumu pridania pesničky do zariadenia v zoradenom zozname všetkých takýchto dátumov, o je pôvodná hodnota reálneho skóre a c je počet prehratí danej pesničky. Touto úpravou je dosiahnuté to, že všetky pesničky s kladným skóre majú stále šancu na prehranie, ale to aká veľká je táto šanca závisí hlavne na premennej i. Tá umožňuje určenie novosti pesničky v porovnaní s ostatnými pesničkami na základe dátumu pridania pesničky do zariadenia. V prípade, že užívateľ preniesol všetky svoje pesničky spolu s ich hodnoteniami zo starého zariadenia do nového, tak prehrávač stále dokáže detegovať nové pesničky pomocou počtu ich prehraní (premenná c) a to aj napriek tomu, že premenná i je v takomto prípade pre všetky pesničky rovnaká. Ako jeho názov napovedá, tak mód pre najobľúbenejšie pesničky slúži na to, aby sa užívateľovi častejšie prehrávali jeho obľúbené skladby, pričom obľúbenosť je v tomto prípade odvodená od skóre pesničky. Tento mód upravuje reálne skóre len vtedy, keď má skóre, ktoré vidí užívateľ, nezápornú hodnotu a to nasledujúcim vzorcom:

$$n = 2^{\sqrt{u}}, \tag{5.7}$$

kde n je nová hodnota reálneho skóre pesničky a u je skóre, ktoré sa zobrazuje užívateľom. Skóre, ktoré sa zobrazuje užívateľom, sa tu používa hlavne z toho dôvodu, že toto skóre sa

na rozdiel od reálneho skóre po prehratí pesničky nenuluje. To umožňuje častejšie opakovanie najobľúbenejších pesničiek, ktorých reálne skóre je navyše v tomto móde niekoľkokrát zväčšené vďaka použitej exponenciálnej funkcii. Podobne ako v predchádzajúcom móde, tak aj tu majú pesničky s nízkym skóre stále šancu na to, že sa prehrajú, no táto šanca je značne menšia v porovnaní s pesničkami s vyšším skóre.

Následne algoritmus vygeneruje tri dôležité hodnoty, ktoré bude ďalej využívať algoritmus výberu pesničky:

1. dolná hranica skóre – predstavuje hodnotu sumy reálnych skóre doposiaľ spracovaných pesničiek bez pridania reálneho skóre aktuálnej pesničky. Ak je hodnota reálneho skóre pesničky menšia ako nula alebo je pesnička blokovaná, tak sa táto hodnota nastaví na -1. To sa vykonáva kvôli tomu, že pesničky, ktorých reálne skóre je nula alebo menej sa neprehrávajú, tak isto ako ani blokované pesničky, a práve toto to umožňuje.

2. Finálna hodnota reálneho skóre pesničky – ak je táto hodnota väčšia ako nula a zároveň nie je pesnička blokovaná, tak sa pridá do sumy reálnych skóre doposiaľ spracovaných pesničiek.

3. Horná hranica skóre – predstavuje hodnotu sumy reálnych skóre doposiaľ spracovaných pesničiek po pridaní reálneho skóre aktuálnej pesničky. Z rovnakých dôvodov ako pri dolnej hranici skóre platí, že ak je hodnota reálneho skóre pesničky menšia ako nula alebo ak je pesnička blokovaná, tak sa táto hodnota nastaví na -1.

5.2 Algoritmus výberu pesničky

Po tom ako sa načíta skóre a ďalšie atribúty všetkých pesničiek pri spustení prehrávača, tak sa rovnako načítajú dôležité nastavenia z predchádzajúceho behu aplikácie. Medzi tieto nastavenia patrí:

- pätnásť naposledy prehratých pesničiek, ktoré sa vložia do zoznamu predchádzajúcich pesničiek,

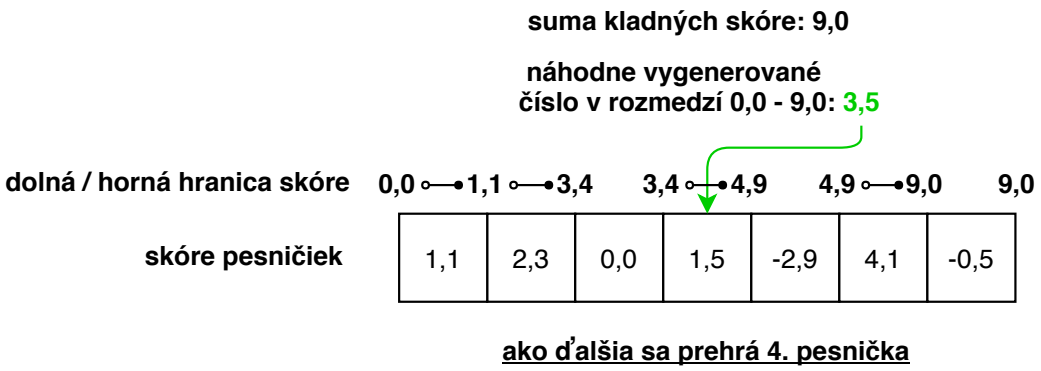
- päť prvých pesničiek, ktorých prehrávanie bolo naplánované do budúcnosti a ktoré sa vložia do zoznamu nasledujúcich pesničiek,

- pesnička, ktorá bola práve prehrávaná pri predchádzajúcom vypnutí aplikácie, vrátane pozície jej prehrávania. Táto pesnička sa načíta a pripraví sa na prehrávanie.

Ak bol ale prehrávač spustený prvýkrát po inštalácii alebo ak nastala pri spustení nejaká chyba, tak prehrávač musí byť sám schopný vybrať niektorú z pesničiek na základe jednotlivých skóre.

Pri každom výbere novej pesničky je dôležité, aby najskôr prebehlo prepočítanie skóre pre aktuálne zobrazený zoznam pesničiek (ktorý sa môže meniť napríklad pri vyhľadávaní) a to kvôli tomu, aby sa správne nastavila dolná a horná hranica skóre každej pesničky aktuálneho zoznamu. Následne sa skontroluje, či je suma všetkých reálnych skóre väčšia ako nula. Ak toto neplatí, tak to môže znamenať, že sa v zariadení nenachádza žiadny hudobný súbor, všetky pesničky majú negatívne skóre, sú zablokované alebo sa jediná nezablokovaná pesnička s kladným skóre práve prehrala. Ak ide o posledný prípad, tak sa táto jedna pesnička bude prehrávať dookola, ináč sa prehrávanie pozastaví. V prípade, že hodnota sumy bola skutočne kladná, tak algoritmus vygeneruje náhodné číslo v rozmedzí od nuly

až po hodnotu sumy všetkých reálnych skóre. Na základe dolnej a hornej hranice skóre je potom možné jednoducho určiť ktorá pesnička sa má prehrať – ak je dolná hranica skóre pesničky menšia ako náhodné číslo a zároveň pre tú istú pesničku platí, že jej horná hranica skóre je väčšia alebo zhodná s náhodným číslom, tak práve táto pesnička sa vyberie a prehrá (obr. 5.1). Platí teda, že čím väčšie je reálne skóre pesničky, tak tým má pesnička väčšiu šancu na to, že náhodné číslo sa bude nachádzať práve v jej rozmedzí dolného a horného skóre a to znamená, že bude vybratá na prehranie.



Obr. 5.1: Výber novej pesničky závisí hlavne na vygenerovanom čísle a súčasne na hornej a dolnej hranici skóre. Obrázok znázorňuje tento proces, ako aj fakt, že pesničky, ktorých skóre je nula alebo menej sa neprehrávajú.

Správanie algoritmu pre výber pesničky sa mierne líši v prípade, keď sa pesnička prehrá celá alebo keď sa užívateľ sám rozhodne prejsť na ďalšiu pesničku. Pesnička, ktorej prehrávanie práve skončilo, sa vloží do zoznamu predchádzajúcich pesničiek, udelí sa jej odpovedajúce hodnotenie a algoritmus skontroluje, či je zoznam nasledujúcich pesničiek prázdny. Ak má tento zoznam nejaké položky, tak algoritmus z neho vyberie pesničku nachádzajúcu sa na prvej pozícii, ktorá sa vzápätí začne prehrávať. V opačnom prípade algoritmus vyberie novú pesničku podľa postupu z predchádzajúceho odseku. Algoritmus sa správa obdobne pri návrate na predchádzajúcu pesničku. Aktuálna pesnička sa vloží do zoznamu nasledujúcich pesničiek a skontroluje sa, či má zoznam prechádzajúcich pesničiek nejaké položky. Ak nejaké má, tak sa z neho vyberie pesnička nachádzajúca sa na začiatku tohto zoznamu a udelí sa jej hodnotenie za to, že sa k nej užívateľ vrátil. Ak je zoznam prázdny, tak sa vyberie pesnička podľa postupu vyššie. Užívateľ môže toto správanie dodatočne upraviť v nastaveniach a to takým spôsobom, že ak sa prehralo viac ako 5 % pesničky, tak namiesto návratu na predchádzajúcu pesničku sa pozícia prehrávania vráti späť na začiatok aktuálnej pesničky. Súčasne platí, že ak sa pred návratom na začiatok prehralo aspoň 25 % pesničky, tak sa pesničke udelí hodnotenie za návrat na jej začiatok. Prechod na predchádzajúcu pesničku sa pri takto upravenom správaní vykoná len vtedy, ak sa prehralo menej ako 5 % pesničky.

Alternatívne môže užívateľ v nastaveniach vypnúť náhodné prehrávanie. Algoritmus v takomto prípade stále berie do úvahy obsah zoznamu predchádzajúcich a nasledujúcich pesničiek, no ak sú tieto zoznamy prázdne, tak sa jednoducho vyberie nasledujúca pesnička z hlavného zoznamu pesničiek. Po prehratí poslednej pesničky zo zoznamu sa automaticky začne prehrávať zoznam od začiatku. Rovnako začne algoritmus prehrávať pesničku zo začiatku, ak sa hlavný zoznam výrazne zmenší vyhľadávaním.

5.3 Algoritmus vyhľadávania

Vyhľadávanie ovplyvňuje prehrávanie tým spôsobom, že pri použití vyhľadávania sa prehrávajú len jeho výsledky. Vďaka tomu nie sú v aplikácii potrebné záložky a špeciálne zoznamy pesničiek zvlášť pre skladby, albumy, interpretov a podobne – užívateľ jednoducho do vyhľadávania zadá, čo chce počúvať a prehrávač mu bude prehrávať len to. Aplikácia podporuje aj viacnásobné vyhľadávanie, čiže keď chce užívateľ poslúchať pesničky od dvoch alebo viacerých rôznych interpretov, tak jediné, čo musí urobiť je zadať ich mená do vyhľadávania a oddeliť ich symbolom zvislej čiary („|“).

Proces vyhľadávania začína, keď užívateľ zadá do príslušného políčka v užívateľskom rozhraní prvý znak, pričom s každým ďalším pridaným alebo odobraným znakom sú výsledky vyhľadávania aktualizované. Keď je toto políčko prázdne, aplikácia prehráva celý zoznam pesničiek. Každý vstup od užívateľa je pre lepšiu užívateľskú skúsenosť prevedený na malé písmená a sú z neho odstránené diakritické znamienka. Vyhľadávanie takto vráti požadované výsledky aj v prípade, keď užívateľ nezadal správnu veľkosť písmen alebo keď nepoužil diakritiku. Pri viacnásobnom vyhľadávaní sa vstup od užívateľa rozdelí na jednotlivé časti, keďže pre každú časť prebehne samotné vyhľadávanie v zozname pesničiek zvlášť. Každá časť sa následne rozdelí na slová. Algoritmus vyhľadávania skontroluje výskyt prvého zadaného slova v názve každej skladby zo zoznamu pesničiek, ako aj v názve jej interpreta a albumu, pričom tieto hodnoty sú taktiež prevedené na malé písmená a je z nich odstránená diakritika. Ak názov pesničky, interpreta alebo albumu obsahuje hľadané slovo, tak je táto pesnička pridaná do nového výstupného zoznamu. Ďalšie slovo v poradí sa už nehľadá v celom zozname pesničiek, ale v tom, ktorý bol získaný ako výsledok vyhľadávania prvého slova. Tento postup sa opakuje, až kým sa neprejde všetkými slovami z danej časti. Použitie tohto princípu zabezpečuje to, že čím viac slov je zadaných a čím sú slová dlhšie, tak tým presnejšie sú aj výsledky vyhľadávania. Výsledky všetkých častí viacnásobného vyhľadávania sú na záver spojené do spoločného zoznamu. Tento finálny zoznam sa ešte zoradí podľa aktuálne zvoleného usporiadania a po zoradení sa zobrazí užívateľovi. Ak je finálny zoznam prázdny, a teda sa na základe zadaného textu vyhľadávania nenašla zhoda v žiadnom názve pesničky, jej interpreta ani albumu, tak sa po prehratí aktuálnej pesničky prehrávanie hudby pozastaví.

Kapitola 6

Implementácia hudobného prehrávača

Táto kapitola bližšie popisuje implementačné detaily pre vytvorenie hudobného prehrávača ako na platforme Android, tak na platforme UWP. V prvých podkapitolách sú popísané spoločné prvky pre obe platformy a vo všeobecnosti všetko to, čo je nutné riešiť pri implementácii každého hudobného prehrávača bez ohľadu na zvolenú platformu. To zahŕňa grafické užívateľské rozhranie, získavanie hudobných súborov, ich prehrávanie a správu dát, ktoré prehrávač vytvára. Okrem toho majú obidve platformy ďalšie špecifické prvky, ktoré zlepšujú užívateľskú skúsenosť pri počúvaní hudby alebo poskytujú dodatočnú funkcionalitu. Všetky tieto špecifické prvky sú v tejto kapitole popísané tiež.

Hudobný prehrávač pre platformu Android je možné vytvoriť vo viacerých programovacích jazykoch. Medzi tie najpoužívanejšie patria programovacie jazyky Java, Kotlin, C#, C++ alebo poprípade JavaScript [7]. Pre moje riešenie som zvolil programovací jazyk Java a to hlavne z toho dôvodu, že sa jedná o oficiálny jazyk pre vývoj na platforme Android. Jeho použitie pre túto platformu je teda riadne zdokumentované a tak isto je podporovaný vývojovým prostredím Android Studio. Programovací jazyk Java je taktiež široko používaný a sám som s ním mal mnoho skúseností ešte pred vytváraním popisovaného hudobného prehrávača. Aplikácie pre platformu UWP môžu byť vytvorené pomocou jedného z nasledujúcich programovacích jazykov: C#, Visual Basic, C++ alebo JavaScript, pričom všetky tieto jazyky sú podporované vývojovým prostredím Visual Studio [2, What's a Universal Windows Platform (UWP) app?[1]]. Jazyk C# som zvolil z toho dôvodu, že sa jedná o vysoko-úrovňový jazyk, ktorý je syntakticky aj sémanticky podobný programovaciemu jazyku Java, v ktorom bola vytvorená mobilná verzia hudobného prehrávača. Ďalším dôvodom je, že dokumentácia pre vývoj klasických typov aplikácií na platforme UWP, akými sú aj hudobné prehrávače, je zväčša dostupná len pre tento programovací jazyk. Z dostupných jazykov som mal taktiež práve s týmto programovacím jazykom najväčšie skúsenosti.

6.1 Grafické užívateľské rozhranie

To, že na obrazovke počítača je možné zobraziť podstatne väčšie množstvo informácií ako na obrazovke chytrého telefónu spôsobuje, že vytvorenie kompletne totožného grafického užívateľského rozhrania pre obe verzie hudobného prehrávača by bolo neefektívne a tým

[1] https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide

pádom sú nevyhnutné určité zmeny v užívateľskom rozhraní medzi jednotlivými verziami. Grafické užívateľské rozhranie oboch verzií však spĺňa základnú spoločnú vlastnosť, ktorou je vytvorenie čo najjednoduchšieho, minimalistického užívateľského rozhrania. Dôraz je taktiež kladený na použiteľnosť. To znamená, že osobe s priemernými (alebo dokonca aj podpriemernými) schopnosťami a skúsenosťami bude po spustení aplikácie ihneď jasné, ako má spustiť a ovládať prehrávanie hudby bez toho, aby pozornosť tejto osoby odvádzalo množstvo rôznych záložiek a tlačidiel. Užívateľ by mal mať vždy po ruke tie funkcie, ktoré používa často ako aj tie, ktoré potrebuje rýchlo nájsť a spustiť. Práve preto medzi kľúčové prvky užívateľského rozhrania oboch verzií hudobného prehrávača patrí hlavný zoznam pesničiek a panel s ovládacími prvkami, ktoré intuitívne umožňujú okamžité spustenie prehrávania hudby [1].

6.1.1 Grafické užívateľské rozhranie mobilnej verzie prehrávača

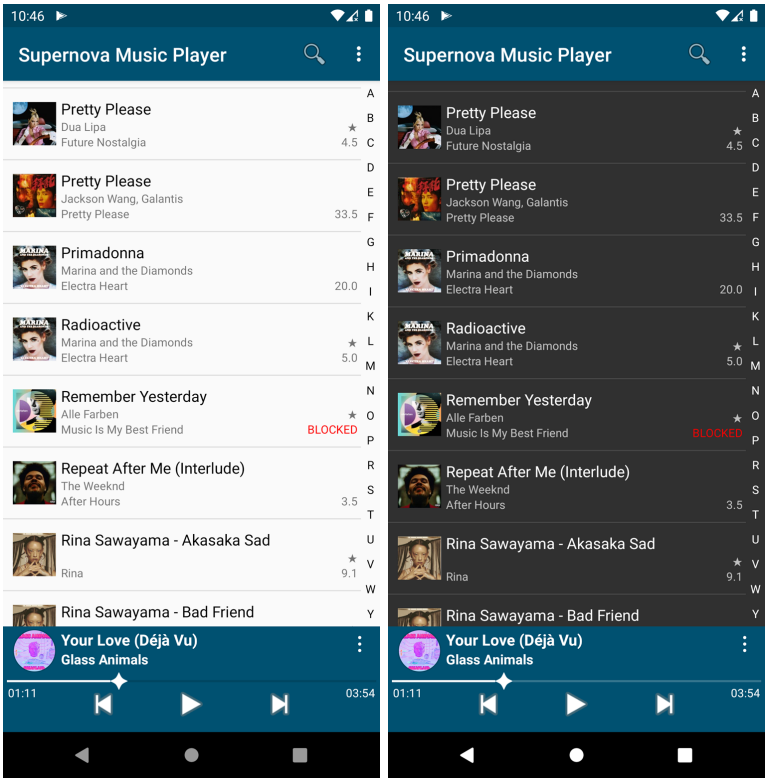
Grafické užívateľské rozhranie mobilnej aplikácie pozostáva z viacerých obrazoviek:

- hlavná obrazovka,
- obrazovka nastavení,
- obrazovka so zoznamom predchádzajúcich a nasledujúcich pesničiek,
- obrazovka so zoznamom blokovaných pesničiek.

Hlavnú obrazovku (obr. 6.1) je možné zhora nadol rozdeliť na tri časti:

- titulný panel aplikácie,
- hlavný zoznam pesničiek,
- panel s ovládacími prvkami.

Titulný panel (obr. 6.2) je súčasťou každej obrazovky, pričom plní základné účely navigácie: pomáha užívateľovi nájsť to, čo hľadá, zobrazuje informáciu o tom, na akej obrazovke sa práve nachádza a umožňuje návrat z danej obrazovky na hlavnú [1]. Titulný panel hlavnej obrazovky je špecifický tým, že namiesto názvu obrazovky zobrazuje názov aplikácie a ďalej aj tým, že obsahuje hlavné menu aplikácie (obr. 6.2). Hlavné menu plní dve základné úlohy – tou prvou je, že umožňuje prechod na všetky ostatné obrazovky aplikácie a tou druhou je, že poskytuje prístup k najdôležitejším dodatočným funkciám, ktoré môže užívateľ často potrebovať. Medzi tieto funkcie konkrétne patrí: vyhľadávanie, usporiadanie hlavného zoznamu pesničiek, vymazanie obsahu zoznamu nasledujúcich pesničiek, nastavenie, spustenie a prípadne aj zrušenie časovača pre vypnutie aplikácie a taktiež aj okamžité vypnutie aplikácie. Pre vytvorenie hlavného menu je potrebné vytvoriť XML súbor, ktorý bude definovať štruktúru menu. Koreňovým prvkom tohto súboru musí byť prvok typu `menu`, ktorý reprezentuje celé menu a ktorý môže v sebe obsahovať prvky typu `item`, ktoré reprezentujú jednotlivé položky menu. Položkám menu je možné prideliť identifikátor a zobrazovaný názov. Každá položka môže ďalej obsahovať vnorený prvok typu `menu`, čím vznikne podmenu. Pre zobrazenie hlavného menu je nutné v príslušnej aktivite prepísať metódu `onCreateOptionsMenu()`, v ktorej sa objektu triedy `Menu` pridelí štruktúra na základe vytvoreného XML súboru. Pre definovanie správania a ošetrenie udalosti kliknutia niektorej položky z hlavného menu je nutné prepísať metódu `onOptionsItemSelected()`,

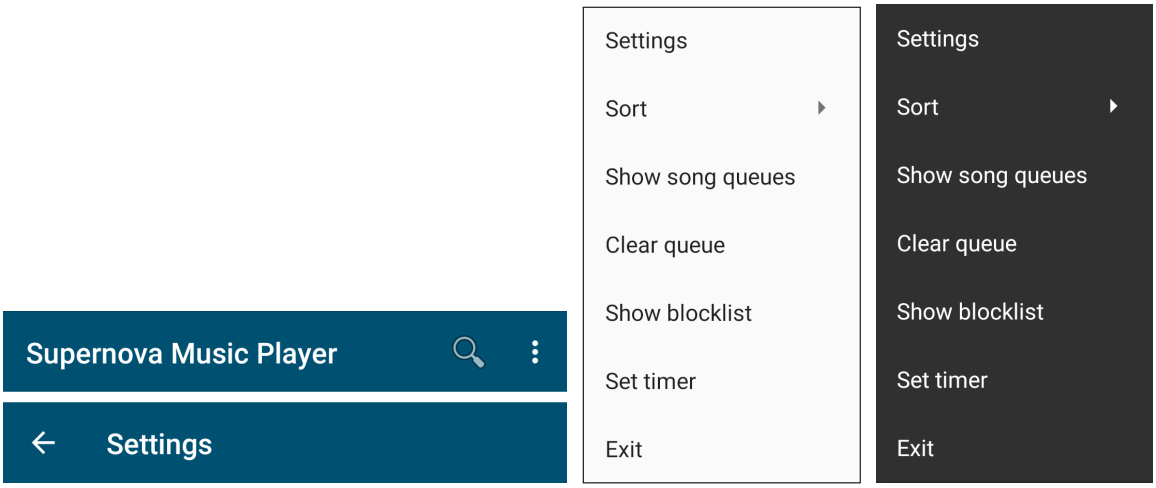


Obr. 6.1: Obrázky zobrazujú celú hlavnú obrazovku mobilnej verzie hudobného prehrávača, ktorá sa skladá z titulného panelu, hlavného zoznamu pesničiek a z ovládacieho panelu. Na ľavom obrázku je zobrazená hlavná obrazovka vo svetlom režime a na pravom obrázku je hlavná obrazovka v tmavom režime.

pričom na základe identifikátoru definovanom v XML súbore je možné jednoznačne určiť, ktorá položka menu bola stlačená [4, Menus[2]].

Špecifickou položkou hlavného menu je vyhľadávanie (obr. 6.3), ktoré je implementované ako `SearchView`. Aby bolo vyhľadávanie funkčné, tak je nutné najskôr vytvoriť XML súbor s konfiguráciou pre vyhľadávanie. Následne musí byť tento súbor nastavený v manifeste aplikácie pri aktivite s vyhľadávaním. Pri tejto aktivite je okrem toho nutné explicitne vyznačiť, že táto aktivita dokáže spracovať akciu vyhľadávania. V metóde `onCreateOptionsMenu()` tejto aktivity je potom nutné získať položku vyhľadávania z hlavného menu, z ktorej sa následne vytvorí objekt triedy `SearchView`. Pre `SearchView` sa tu nastaví ďalší objekt, ktorý bude reagovať na udalosť zmeny textu vyhľadávania a ktorý obsahuje samotnú logiku vyhľadávania popísanú v kapitole 5.3 [4, Creating a Search Interface[3]]. Vyhľadávanie je špecifické tým, že sa jeho ikona zobrazuje samostatne od ostatných položiek hlavného menu (obr. 6.2), aby k nej mal užívateľ čo najrýchlejší prístup. Toto správanie je dosiahnuté nastavením atribútu položky vyhľadávania `showAsAction` v XML súbore menu na hodnotu `always|collapseActionView` a nastavením obrázku ikony atribútom `icon`. Po kliknutí na ikonu vyhľadávania sa táto ikona skryje spolu s názvom aplikácie v titulnom paneli a namiesto nich sa zobrazí pole, do ktorého môže užívateľ zadať text vyhľadávania cez klávesnicu, ktorá sa po kliknutí automaticky zobrazí tiež (obr. 6.3). Ak užívateľ zadal

[2] https://developer.android.com/guide/topics/ui/menus
[3] https://developer.android.com/guide/topics/search/search-dialog

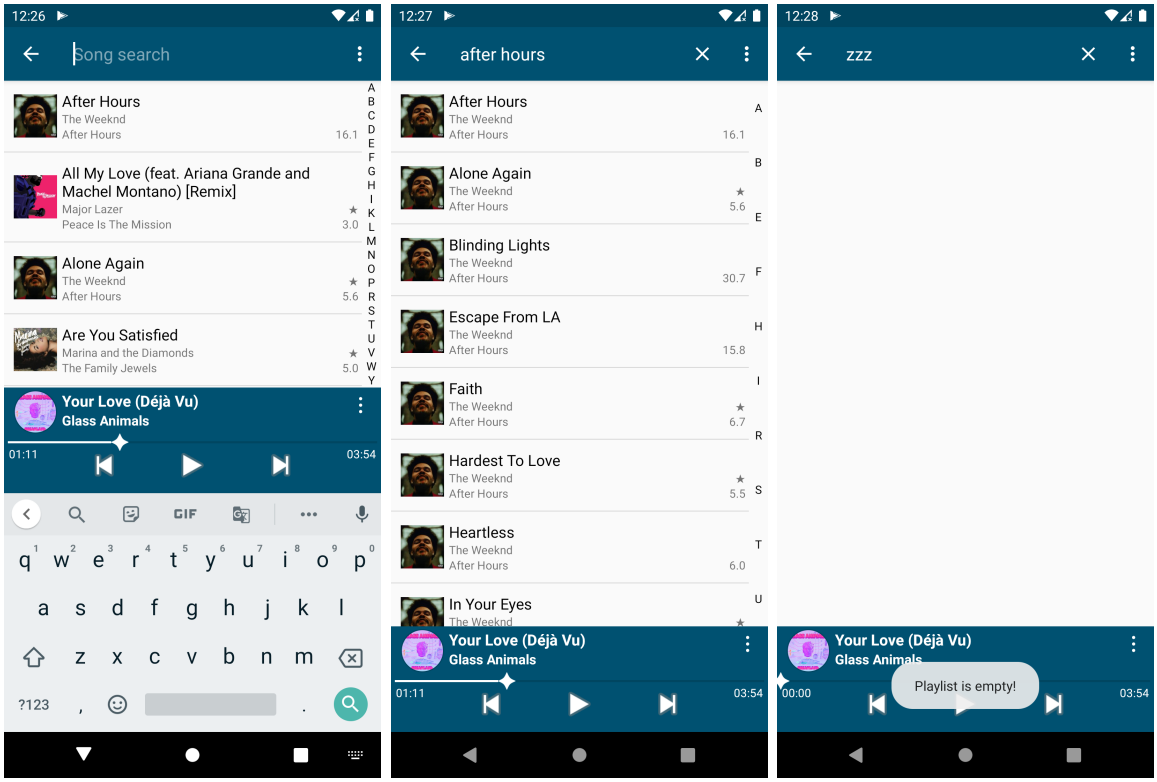


Obr. 6.2: Horný obrázok vľavo zobrazuje titulný panel aplikácie z hlavnej obrazovky, na ktorom sa nachádza názov aplikácie a ktorý umožňuje spustiť vyhľadávanie alebo otvoriť hlavné menu. Spodný obrázok vľavo zobrazuje titulný panel z obrazovky nastavení, ktorý zobrazuje názov obrazovky a ktorý umožňuje návrat na hlavnú obrazovku. Všetky ostatné obrazovky taktiež zdieľajú tento typ titulného panelu. Obrázok v strede a vpravo zobrazuje otvorené hlavné menu vo svetlom a tmavom režime. Hlavné menu umožňuje vykonať prechod z hlavnej obrazovky na všetky ostatné a taktiež ponúka rôzne dodatočné funkcie.

do tohto poľa aspoň jeden znak, tak sa v pravom rohu poľa zobrazí krížik, ktorým je možné vymazať celý text vyhľadávania. Súčasne sa po zadaní textu vyhľadávania premietajú jeho výsledky priamo do hlavného zoznamu pesničiek, ktorý je umiestnený hneď pod titulným panelom. Počas vyhľadávania je stále k dispozícii prístup k ostatným položkám hlavného menu, takže užívateľ môže aj počas vyhľadávania zmeniť usporiadanie zoznamu pesničiek alebo prejsť do nastavení, pričom po návrate na hlavnú obrazovku je vyhľadávanie stále aktívne. Okrem samotného poľa na vyhľadávanie sa naľavo od neho zobrazí šípka, pomocou ktorej je možné ukončiť vyhľadávanie. Kliknutie na túto šípku spôsobí, že sa daná šípka, ako aj pole pre vyhľadávanie skryje a znova sa zobrazí názov aplikácie a ikona pre vyhľadávanie. Ak užívateľ niečo vyhľadával, tak sa po ukončení vyhľadávania taktiež znova zobrazí celý zoznam pesničiek.

Typy usporiadania hlavného zoznamu pesničiek sú skryté do podmenu (obr. 6.4), aby hlavné menu po zobrazení nezaberalo príliš veľkú časť obrazovky. Samotné usporiadanie sa vykonáva pomocou objektu triedy `Collator`. Tento objekt sa používa pre porovnávanie reťazcov, pričom berie do úvahy lokalizačné nastavenia chytrého telefónu [4, Collator[4]]. Pri každej zmene usporiadania si prehrávač túto zmenu uloží, aby pri spustení prehrávača bol zoznam zoradený tým istým spôsobom, akým bol zoradený pri predchádzajúcom vypnutí. Prvým a predvoleným typom usporiadania je usporiadanie podľa názvu pesničiek (obr. 6.4). Pri usporiadaní zoznamu podľa názvu pesničiek sú ich názvy pri porovnávaní prevedené na malé písmená, aby bol zoznam pesničiek zoradený abecedne správne bez ohľadu na veľkosť písmen. Keď je zvolený tento typ usporiadania, tak sa napravo od zoznamu pesničiek zobrazí panel s počiatočnými písmenami z názvu pesničiek, ktorý umožňuje rýchly presun k pesničkám začínajúcim daným písmenom. Druhým typom usporiadania je usporiadanie podľa skóre pesničiek (obr. 6.4). Pri tomto usporiadaní sa zoznam pesničiek najskôr uspo-

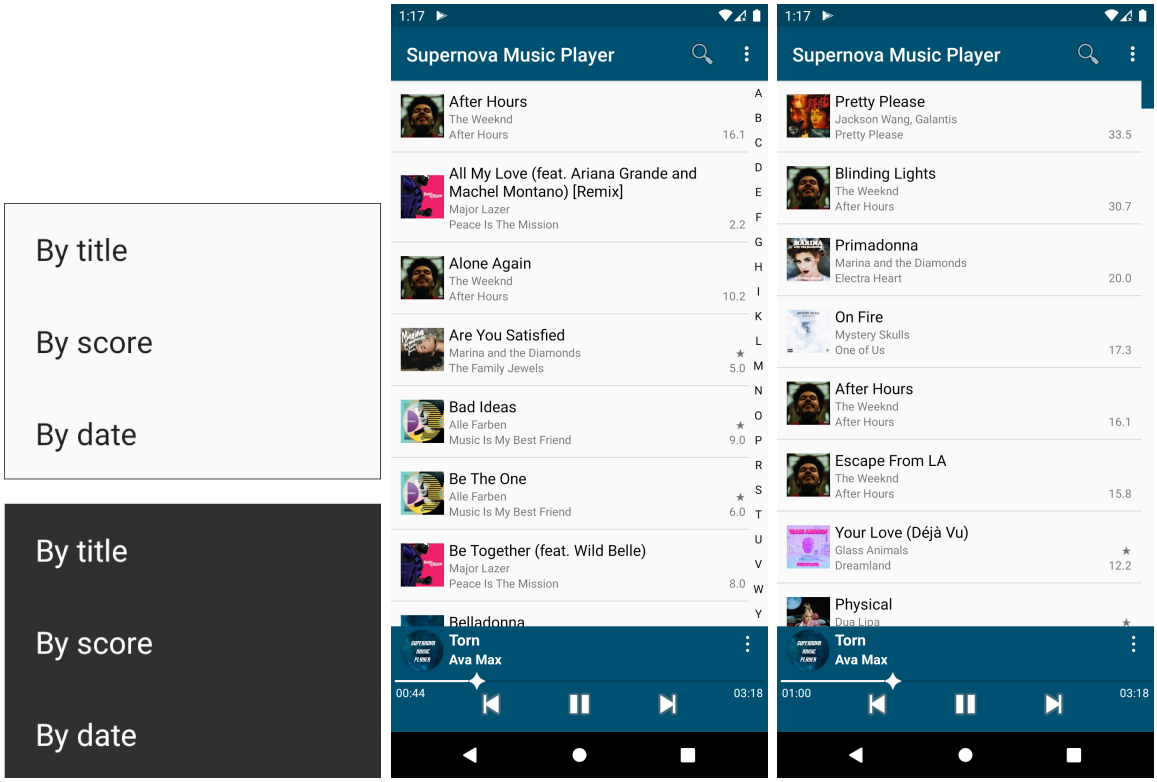
[4] https://developer.android.com/reference/java/text/Collator



Obr. 6.3: Na ľavom obrázku je zobrazená situácia bezprostredne po kliknutí na ikonu vyhľadávania z titulného panelu, pri ktorej sa zobrazí textové pole pre vyhľadávanie, šípka pre ukončenie vyhľadávania a klávesnica. Na strednom obrázku je možné vidieť ako sa výsledky vyhľadávania zobrazujú priamo v hlavnom zozname pesničiek a obrázok vpravo zobrazuje situáciu, keď sa pri vyhľadávaní nenájde žiadna zhoda. Prehrávanie hudby je v takomto prípade pozastavené a užívateľ je o tejto skutočnosti informovaný.

riada podľa názvu a až potom podľa skóre, ktoré je zobrazené užívateľom. Tým sa dosiahne to, že zoznam je primárne usporiadaný podľa skóre, ale keď má viacero pesničiek rovnaké skóre, tak je táto skupina pesničiek dodatočne zoradená podľa ich názvu. Rovnaký postup je aplikovaný aj pri poslednom type usporiadania. Usporiadanie podľa dátumu pridania pesničiek do zariadenia najskôr usporiada zoznam podľa názvu pesničky a až potom podľa dátumu, aby pesničky s rovnakým dátumom boli zoradené podľa ich názvu.

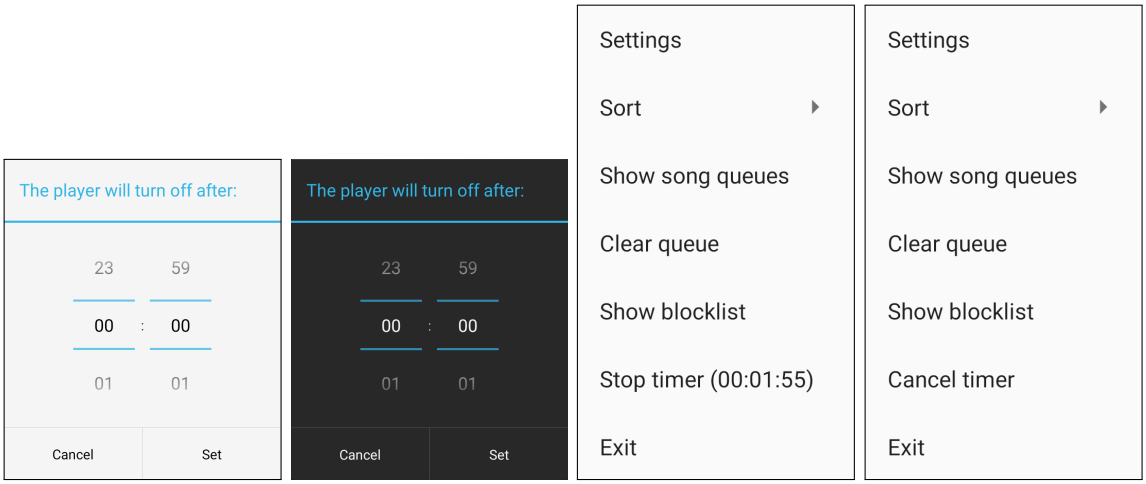
Ako názov napovedá, tak funkcia z hlavného menu pre vymazanie obsahu zoznamu nasledujúcich pesničiek jednoducho vyprázdni tento zoznam v prípade, keď užívateľ naplnil tento zoznam pesničkami, ktoré už viac nemá záujem počúvať. Po odstránení všetkých položiek z tohto zoznamu je o tejto skutočnosti užívateľ informovaný cez objekt triedy `Toast`, ktorý sám po niekoľkých sekundách zmizne. Implementačne zaujímavejšou funkciou je nastavenie časovača pre vypnutie aplikácie. Po kliknutí na odpovedajúcu položku v hlavom menu sa užívateľovi zobrazí dialógové okno typu `TimePickerDialog` (obr. 6.5), z ktorého môže nastaviť, za akú dobu (v hodinách a minútach) sa má prehrávač vypnúť. Užívateľ môže nastavovanie časovača zrušiť kliknutím na ľavé tlačidlo dialógového okna alebo kliknutím mimo dialógové okno. Keď užívateľ spustí časovač stlačením pravého tlačidla dialógového okna, tak sa zvolená doba prepočíta na milisekundy, vytvorí a spustí sa objekt triedy `CountDownTimer`, ktorý sa daným počtom milisekúnd inicializuje. Po spustení časovača sa



Obr. 6.4: Ľavý obrázok zobrazuje podmenu so všetkými možnosťami usporiadania hlavného zoznamu pesničiek, ktorý sa zobrazí po kliknutí na položku usporiadania v hlavnom menu. Horný obrázok zobrazuje toto podmenu vo svetlom režime a spodný obrázok ho zobrazuje v tmavom režime. Na strednom obrázku sa nachádza zoznam pesničiek zoradený podľa názvu pesničiek. Pri tomto usporiadaní sa na boku zoznamu zobrazuje panel s písmenami, ktorý umožňuje rýchly prechod k pesničkám, ktorých názov začína zvoleným písmenom. Na obrázku vpravo je zoznam pesničiek zoradený podľa skóre, pričom namiesto panelu s písmenami sa tu zobrazuje klasická posuvná lišta, tak isto ako pri usporiadaní podľa dátumu pridania pesničiek do zariadenia.

názov položky menu zmení z „nastaviť časovač“ na „zastaviť časovač“ a v zátvorke za tým sa zobrazí zostávajúci čas (obr. 6.5). Keď teda užívateľ znova stlačí tlačidlo časovača pred uplynutím zostávajúceho času, tak sa časovač zastaví, k vypnutiu aplikácie nedôjde a text položky menu sa nastaví späť na „nastaviť časovač“. Ak uplynul všetok zostávajúci čas, tak je vyvolaná špeciálna ukončujúca udalosť, pomocou ktorej sa prehrávač okamžite vypne bez ohľadu na to, na akej obrazovke sa užívateľ práve nachádzal. Užívateľ môže toto správanie upraviť z nastavení a to takým spôsobom, že prehrávač sa nevypne hneď po uplynutí zostávajúceho času, ale až po tom ako sa po uplynutí zostávajúceho času dokončí prehrávanie aktuálnej pesničky. Po uplynutí zostávajúceho času sa v takomto prípade zmení text položky menu zo „zastaviť časovač“ na „zrušiť časovač“ (obr. 6.5). Užívateľ môže teda pred dokončením prehrávania aktuálnej pesničky stále zabrániť vypnutiu aplikácie opakovaným stlačením položky menu, čím sa text položky menu nastaví späť na „nastaviť časovač“. `CountDownTimer` má ešte ďalší inicializačný parameter, ktorým je dĺžka intervalu v milisekundách a po uplynutí ktorého je vyvolaná ďalšia udalosť, ktorá periodicky informuje o tom, že ubehol zvolený časový interval z celkového počtu milisekúnd. Hodnota tohto in-

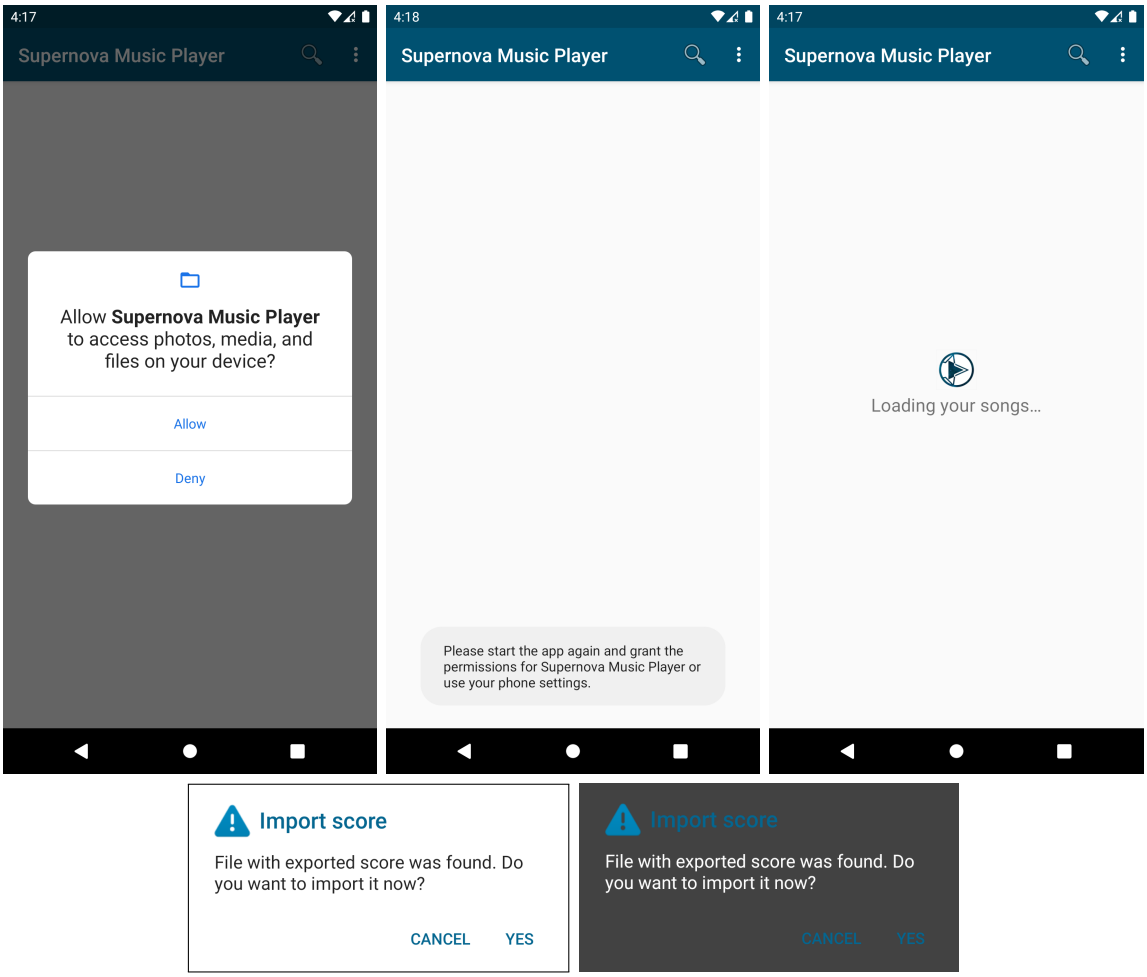
tervalu je v implementácii môjho hudobného prehrávača pevne nastavená na hodnotu 1 000, čo zabezpečuje, aby sa každú sekundu vyvolala daná udalosť, pomocou ktorej sa nastaví aktuálny zostávajúci čas v položke menu. Predtým, ako sa aplikácia vypne, či už cez časovač, položku okamžitého vypnutia hlavného menu, notifikáciu alebo systémový zoznam naposledy spustených aplikácií, tak sa najskôr pozastaví prehrávanie a nastaví sa ikona pre spustenie prehrávania vo widgete. Následne sa uložia všetky dôležité informácie pre ďalšie spustenie aplikácie, ktoré sú bližšie popísané na začiatku kapitoly 5.2 a na záver sa odstráni notifikácia a uvoľnia sa všetky potrebné zdroje.



Obr. 6.5: Prvé dva obrázky zľava zobrazujú dialógové okno pre nastavenie a spustenie časovača vo svetlom aj v tmavom režime, ktorý slúži na vypnutie prehrávača. Tretí obrázok zľava zobrazuje zmenenú položku hlavného menu po spustení časovača, ktorá zobrazuje zostávajúci čas do vypnutia, pričom opakované kliknutie na ňu spôsobí zrušenie časovača. Obrázok vpravo zobrazuje takisto zmenenú položku hlavného menu pre zrušenie časovača. Táto položka sa ale zobrazuje v prípade, keď zostávajúci čas do vypnutia už ubehol, ale vypnutie nastane až po dokončení prehrávania aktuálnej pesničky.

Pri prvom spustení prehrávača sa namiesto zoznamu pesničiek a panelu s ovládacími prvkami zobrazí okno, ktoré užívateľa požiada, aby povolil aplikácii prístup k súborom v zariadení (obr. 6.6). Toto povolenie je nevyhnutné pre získanie hudobných súborov v zariadení, a preto ak ho užívateľ odmietne udeliť, tak aplikácia pomocou objektu triedy `Toast` vyzve užívateľa k tomu, aby dané povolenie udelil buď pri ďalšom spustení prehrávača, alebo cez nastavenia telefónu (obr. 6.6) a následne sa aplikácia po troch sekundách sama vypne. Keď užívateľ povolenie udelí, tak sa mu zobrazí obrazovka načítavania (obr. 6.6), ktorá obsahuje logo prehrávača a ktorá ho informuje o tom, že prebieha načítavanie jeho pesničiek. Táto obrazovka načítavania sa zobrazuje pri každom spustení, no dlhšie je zobrazená len v dvoch prípadoch: pri prvom spustení prehrávača a pri spustení prehrávača po tom, ako bolo do zariadenia pridané väčšie množstvo nových pesničiek. Obrazovka načítavania sa v oboch týchto prípadoch zobrazuje dlhšie hlavne z toho dôvodu, že všetkým (novým) pesničkám sa v tomto okamžiku vytvára záznam, do ktorého budú ukladané ich hodnotenia, čo určitý čas trvá. Ihneď po tom ako sa dokončí načítavanie pesničiek sa obrazovka načítavania skryje a užívateľovi sa zobrazí zoznam pesničiek spolu s panelom s ovládacími prvkami. Ak sa pri prvom spustení aplikácie nachádza na úložisku zariadenia priečinok s názvom prehrávača obsahujúci textový súbor s názvom „score“, ktorý obsahuje exportované hodno-

tenia všetkých pesničiek z predchádzajúcej inštalácie prehrávača alebo z iného zariadenia, tak aplikácia je schopná tento súbor nájsť a pomocou dialógového okna typu `AlertDialog` (obr. 6.6) túto skutočnosť oznámiť užívateľovi. Zároveň sa ho aplikácia cez toto dialógové okno opýta, či chce tieto hodnotenia znova načítať. Ak užívateľ zvolí, že chce hodnotenia načítať, tak sa spustí proces importu, ktorý je bližšie popísaný v kapitole 6.4.

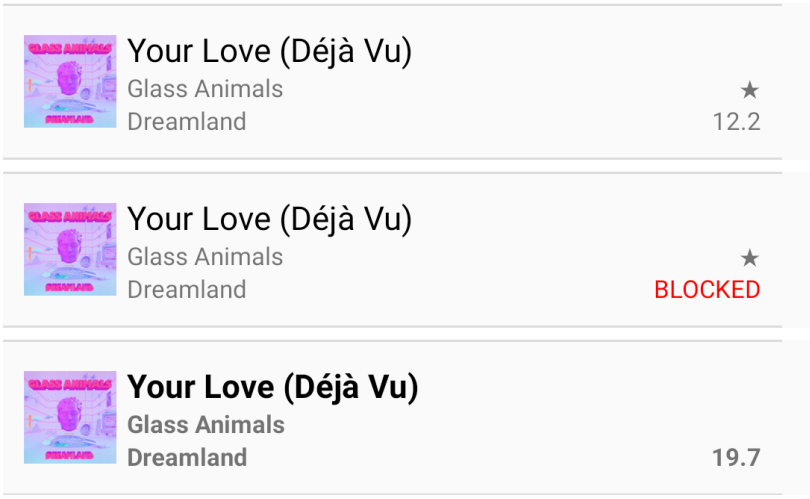


Obr. 6.6: Vľavo hore sa zobrazuje žiadosť o povolenie prístupu k súborom pri prvom spustení prehrávača. Vedľa je zobrazená reakcia aplikácie, keď užívateľ odmietne toto povolenie udeliť. Po udelení daného povolenia sa spustí načítavanie pesničiek zo zariadenia, čo je zobrazené vpravo hore. Ak sa v zariadení pri prvom spustení prehrávača nachádza súbor s hodnoteniami, tak sa prehrávač opýta, či ich má načítať cez dialógové okno zobrazené dolu.

Najväčšiu plochu hlavnej obrazovky zaberá hlavný zoznam pesničiek, ktorý je implementovaný ako `RecyclerView`. `RecyclerView` umožňuje efektívne zobrazenie veľkého množstva dát tým, že dynamicky vytvára položky len vtedy, keď sú potrebné. Tento princíp zlepšuje výkon aplikácie, jej schopnosť reagovať a súčasne znižuje spotrebu energie [4, Create dynamic lists with RecyclerView[5]]. Po načítaní hudobných súborov zo zariadenia je ich zoznam predaný adaptéru, ktorého hlavnou úlohou je to, aby sa pre každý hudobný súbor v zariadení

[5] https://developer.android.com/guide/topics/ui/layout/recyclerview

vytvorila položka v hlavnom zozname pesničiek v užívateľskom rozhraní. Ak sa v zariadení nenachádza žiaden hudobný súbor, tak aj hlavný zoznam pesničiek zostane prázdny, o čom je užívateľ dodatočne informovaný (podobne ako na obr. 6.3). Ak sa v zariadení nachádza aspoň jeden alebo viac hudobných súborov, tak sa pre každý dynamicky vytvorí položka zoznamu (obr. 6.7), ktorej štruktúra je daná samostatným XML súborom. Každá položka hlavného zoznamu pesničiek obsahuje v ľavom rohu obrázok pesničky. Obrázky sa v adaptéri načítavajú pomocou knižnice `Glide` s využitím cesty ku danému hudobnému súboru. V prípade, že pesnička vlastný obrázok nemá, tak sa namiesto toho použije predvolený obrázok aplikácie, ktorý zobrazuje jej názov. Napravo od obrázku sa zhora nadol zobrazuje názov pesničky, názov jej interpreta a názov albumu, na ktorom sa pesnička nachádza. Ak hudobný súbor nemá v metadátach uvedený názov albumu, tak systém automaticky namiesto toho vráti názov priečinku, v ktorom sa hudobný súbor nachádza. Text názvu pesničky má tmavšiu farbu a väčšie písmo ako ostatné dva údaje, aby v zozname viac vynikol. V pravom rohu sa oproti názvu albumu zobrazuje užívateľské skóre pesničky, ktoré je pre jednoduchosť a lepšiu čitateľnosť orezané na jedno desatinné miesto. V prípade, ak užívateľ danú pesničku zablokoval, tak sa mu na tomto mieste zobrazuje namiesto skóre text „BLOCKED“, ktorý je dodatočne zvýraznený červenou farbou (obr. 6.7). Pri odblokovaní pesničky sa tu znova zobrazí jej skóre v pôvodnej farbe textu. Ak bola pesnička prehratá len trikrát alebo menej, tak sa nad skóre zobrazí symbol hviezdy („★“). Vďaka tomuto symbolu je možné v zozname rozpoznať tie pesničky, ktorých skóre je relatívne nízke z toho dôvodu, že ešte neboli veľakrát prehrané od tých, ktoré majú nízke skóre z toho dôvodu, že dostali negatívne hodnotenia za preskočenie. Adaptér dokáže rozpoznať práve prehrávanú pesničku, pričom všetky jej textové údaje zvýrazní tučným písmom, aby bolo túto pesničku možné lepšie a rýchlejšie nájsť v celom zozname pesničiek (obr. 6.7). Medzi dôležité úlohy adaptéru ďalej patrí aktualizácia hodnoty skóre v položke po udelení nového hodnotenia, zobrazenie zmeny usporiadania zoznamu alebo zobrazenie výsledkov vyhľadávania. Adaptér má taktiež na starosti získanie počiatočných písmen z názvov pesničiek, ktoré sa zobrazujú na bočnom paneli, keď je zoznam pesničiek usporiadaný podľa názvu pesničiek.



Obr. 6.7: Na obrázku hore je možné vidieť klasickú položku pesničky v hlavnom zozname pesničiek. Na obrázku v strede je znázornená položka zablokovanej pesničky a na dolnom obrázku sa nachádza položka práve prehrávanej pesničky. Na dolnom obrázku už bola daná pesnička prehratá aspoň štyrikrát, a preto sa u nej už symbol hviezdy nezobrazuje.

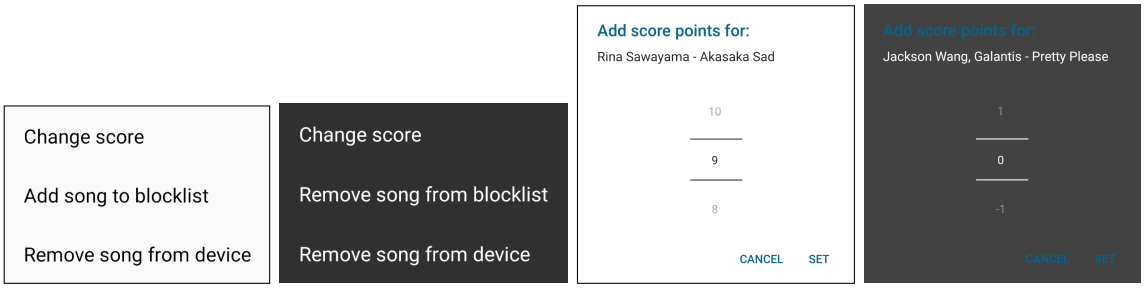
Každá položka hlavného zoznamu dokáže reagovať na tri udalosti:

- krátke kliknutie na položku,
- dlhé kliknutie na položku,
- potiahnutie položky zľava doprava.

Pri krátkom kliknutí na položku zoznamu sa začne prehrávať pesnička, na ktorú sa kliklo. Tento proces pozostáva z niekoľkých krokov. Ak pesnička na ktorú užívateľ klikol nie je súčasne práve prehrávanou pesničkou, tak sa práve prehrávaná pesnička vloží na začiatok zoznamu predchádzajúcich pesničiek. Táto kontrola sa vykonáva z toho dôvodu, aby rovnaká pesnička nebola viackrát za sebou v zozname predchádzajúcich pesničiek. Niečo také by spôsobovalo, že pri prechode na predchádzajúcu pesničku by sa daná pesnička opakovala toľkokrát, koľkokrát na ňu užívateľ klikol. Užívateľ ale skôr očakáva, že pri prechode na predchádzajúcu pesničku sa mu daná pesnička prehrá len raz, aj keď na ňu predtým klikol niekoľkokrát. Zvolenej pesničke sa následne udelí hodnotenie za to, že na ňu užívateľ klikol a jej prehrávanie sa spustí. V prípade, ak pesnička na ktorú užívateľ klikol je súčasne práve prehrávanou pesničkou, tak sa tejto pesničke udelí nové hodnotenie za kliknutie a jej prehrávanie sa spustí od začiatku.

Pri dlhom kliknutí sa na základe pozície položky na obrazovke zobrazí vyskakovacie menu (`PopupMenu`) pod alebo nad danou položkou (obr. 6.8). Užívateľ môže vykonať jednu z troch akcií, ktoré vyskakovacie menu ponúka alebo ho zavrieť stlačením ľubovoľnej časti obrazovky mimo menu. Prvá akcia umožňuje užívateľovi explicitne zmeniť hodnotu skóre danej pesničky. Keď užívateľ zvolí túto akciu, tak sa mu zobrazí dialógové okno (obr. 6.8), ktoré obsahuje titulok informujúci užívateľa o tom, akú akciu zvolil, ako aj názov interpreta a pesničky, aby bolo užívateľovi vždy jasné, ktorú pesničku vybral dlhým kliknutím. Okrem toho obsahuje toto dialógové okno aj objekt triedy `NumberPicker`, ktorý umožňuje užívateľovi zvoliť hodnotu v intervale od -50 do 50, o ktorú sa zvýši alebo zníži hodnota skóre. Po zvolení želanej hodnoty môže užívateľ svoju voľbu potvrdiť pravým tlačidlom v dolnej časti dialógového okna, čím sa daná hodnota uloží a skóre pesničky sa na jej základe upraví. Užívateľ môže taktiež vykonávanie tejto akcie zrušiť ľavým tlačidlom dialógového okna alebo kliknutím mimo toto okno. Ak užívateľ nikdy explicitne nemenil skóre danej pesničky, tak `NumberPicker` bude inicializovaný na hodnotu nula. V opačnom prípade sa pred zobrazením dialógového okna tento objekt inicializuje na naposledy uloženú hodnotu. Keď užívateľ nechce, aby sa mu nejaká pesnička viac prehrávala, tak môže túto pesničku pomocou druhej akcie vyskakovacieho menu zablokovať (obr. 6.8). To prakticky znamená, že sa danej pesničke udelí hodnotenie za zablokovanie. Ak už je pesnička zablokovaná, tak je možné ju pomocou tejto akcie odblokovať, čím sa hodnotenie za zablokovanie z reťazca s hodnoteniami vymaže. Užívateľ môže pomocou poslednej akcie vyskakovacieho menu natrvalo odstrániť vybraný hudobný súbor zo zariadenia. Keďže je vymazanie súboru na platforme Android nezvratné, tak predtým, ako k nemu dôjde sa užívateľovi zobrazí dialógové okno (obr. 6.15), ktoré ho upozorní, že sa snaží odstrániť danú pesničku zo zariadenia a požiada ho, aby túto akciu potvrdil. Ak užívateľ potvrdí vymazanie pesničky pravým tlačidlom dialógového okna, tak dôjde k samotnému procesu vymazania hudobného súboru zo zariadenia, ktorý je detailnejšie popísaný v kapitole 6.2. Ak užívateľ zvolil túto akciu náhodou alebo ak si odstránenie pesničky rozmyslel, tak môže túto akciu zrušiť ľavým tlačidlom dialógového okna alebo kliknutím mimo toto okno.

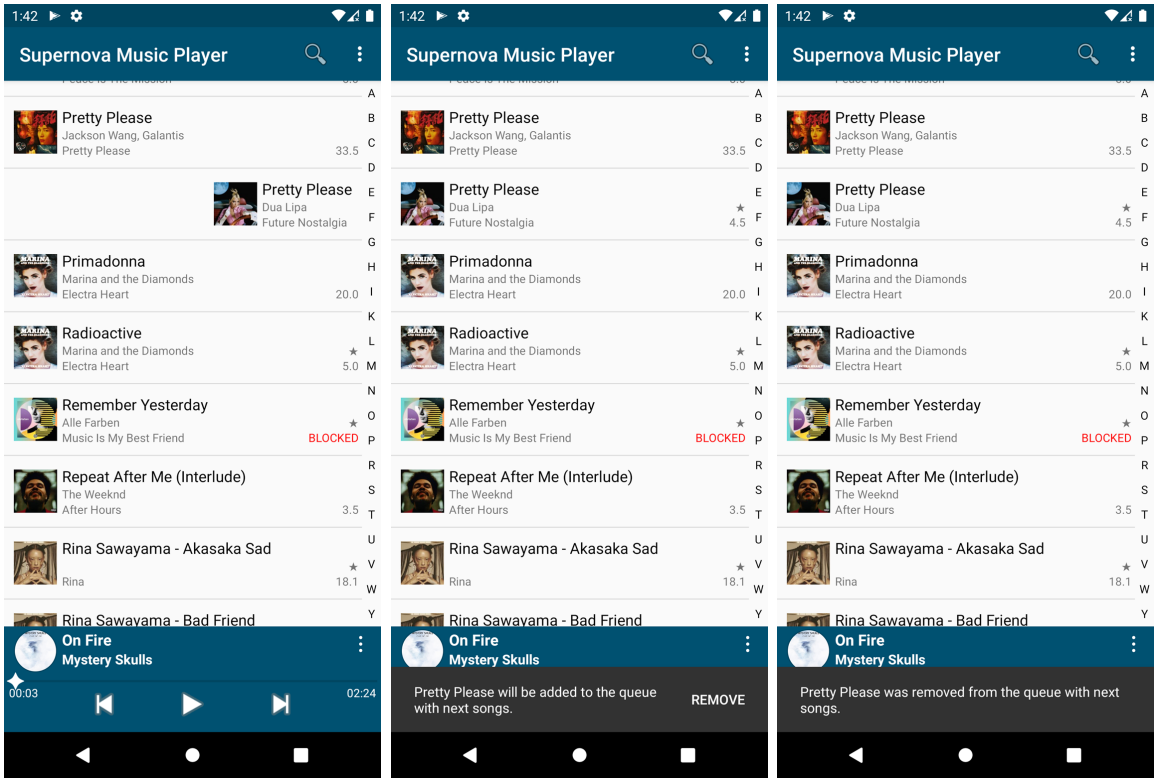
Posledným typom podporovanej udalosti pre položku hlavného zoznamu pesničiek je jej potiahnutie zľava doprava (obr. 6.9). Prehrávač dokáže reagovať ako na samotné potiahnu-



Obr. 6.8: Prvý obrázok vľavo znázorňuje vyskakovacie menu vo svetlom režime, ktoré sa zobrazí pre nezablokovanú pesničku pri dlhom kliknutí na jej položku v hlavnom zozname. Druhý obrázok znázorňuje to isté menu, len v tmavom režime a pre zablokovanú pesničku. Posledné dva obrázky vpravo zobrazujú dialógové okno, ktorým môže užívateľ upraviť skóre pesničky podľa vlastného uváženia.

tie, tak aj na udalosť zdvihnutia prstu pri ukončení potiahnutia. Ak užívateľ zdvihol prst s potiahnutou položkou zoznamu mimo jej pôvodnej pozície, tak sa položka do tejto pozície automaticky vráti, zvolená pesnička sa pridá na koniec zoznamu nasledujúcich pesničiek a užívateľ je o tejto skutočnosti informovaný cez `Snackbar` zobrazený v spodnej časti obrazovky (obr. 6.9). `Snackbar` obsahuje v pravom rohu tlačidlo, ktorým môže užívateľ vloženie do zoznamu nasledujúcich pesničiek zrušiť v prípade, ak nechcene potiahol položkou zoznamu. Ak užívateľ toto tlačidlo stlačí, tak sa pesnička zo zoznamu nasledujúcich pesničiek odstráni a zobrazí sa nový `Snackbar`, ktorý to oznámi užívateľovi (obr. 6.9). V opačnom prípade sa pôvodný `Snackbar` po pár sekundách sám skryje a pesničke je udelené hodnotenie za jej potiahnutie a vloženie do zoznamu nasledujúcich pesničiek. Špecifickým prípadom je situácia, keď užívateľ potiahol položkou zoznamu, no následne ju vrátil do pôvodnej pozície a až potom zdvihol svoj prst. Tento prípad je alternatívou pre zrušenie potiahnutia a žiadna akcia nebude v takomto prípade vykonaná.

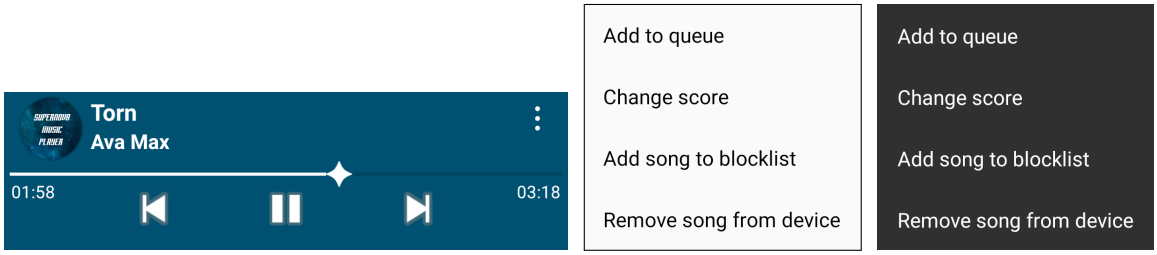
Spodnú časť hlavnej obrazovky tvorí panel s ovládacími prvkami (obr. 6.10). Ako jeho názov napovedá, tak tento panel slúži primárne na ovládanie prehrávania hudby. Okrem toho zabezpečuje tento ovládací panel zobrazenie najdôležitejších informácií o aktuálne prehrávanej pesničke a umožňuje s touto pesničkou vykonávať rôzne akcie bez toho, aby bolo nutné hľadať jej položku v zozname. Ovládací panel má rovnako ako titulný panel kontrastnú tmavomodrú farbu z toho dôvodu, aby ich bolo možné jednoznačne odlíšiť od zoznamu s pesničkami. Panel s ovládacími prvkami dokáže automaticky reagovať na zobrazenie klávesnice a to vďaka tomu, že má nastavenú pevnú výšku. Toto umožňuje, že ovládanie prehrávania je k dispozícii aj počas zadávania textu vyhľadávania (obr. 6.3). V ľavom hornom rohu ovládacieho panelu sa zobrazuje obrázok pesničky v kruhovom výseku a napravo od neho je pod sebou vypísaný jej názov a interpret. Rovnako ako pri položke zoznamu, tak aj tu má názov pesničky o niečo väčšiu veľkosť textu ako názov interpreta, no obidve tieto informácie sú vyznačené tučným písmom. V pravom hornom rohu sa nachádza tlačidlo, pomocou ktorého je možné zobraziť vyskakovacie menu pre aktuálne prehrávanú pesničku (obr. 6.10). Toto vyskakovacie menu obsahuje rovnaké akcie ako vyskakovacie menu položky zo zoznamu pesničiek a navyše pridáva akciu pre vloženie pesničky do zoznamu nasledujúcich pesničiek, ktorou sa tu nahrádza udalosť potiahnutia položky pesničky zľava doprava. Pod týmito prvkami ovládacieho panelu sa nachádza posuvná lišta, pomocou ktorej môže užívateľ meniť pozíciu prehrávania a ktorá je implementovaná ako `SeekBar`. Ako býva v hudobných prehrávačoch zvykom, tak v ľavom rohu pod posuvnou lištou sa



Obr. 6.9: Na obrázku vľavo je zobrazená samotná udalosť potiahnutia položky hlavného zoznamu pesničiek zľava doprava. Po potiahnutí sa v spodnej časti obrazovky zobrazí `Snackbar`, ktorý užívateľa informuje o tom, že pesnička bude vložená do zoznamu nasledujúcich pesničiek, čo je zobrazené na obrázku v strede. Užívateľ môže tomuto zabrániť stlačením tlačidla v pravej časti `Snackbaru`. Ak užívateľ toto tlačidlo stlačí, tak sa pesnička zo zoznamu nasledujúcich pesničiek odstráni a zobrazí sa nový `Snackbar`, ktorý to oznámi užívateľovi. Presne to zobrazuje obrázok vpravo.

zobrazuje aktuálna pozícia prehrávania v minútach a sekundách a na opačnej strane vpravo sa zobrazuje celková dĺžka pesničky rovnako vyjadrená v minútach a sekundách. Aktuálna pozícia prehrávania je taktiež vyjadrená symbolom štvorcípej hviezdy na posuvnej lište, pričom týmto symbolom je možné pohybovať smerom doprava alebo doľava, čím dochádza k zmene pozície prehrávania. Aktuálna pozícia prehrávania sa aktualizuje každých sto milisekúnd, aby bola zaistená plynulá animácia zmeny tejto hodnoty v ovládacom paneli. Okrem toho sú všetky dáta v ovládacom paneli aktualizované pri každej zmene pesničky. Ku aktualizácii dochádza aj keď je užívateľ na inej obrazovke, keď používa inú aplikáciu a aj keď má obrazovku uzamknutú. Zmenu pozície prehrávania je možné dosiahnuť taktiež kliknutím na ľubovoľnú pozíciu na posuvnej lište. Aby dokázal prehrávač reagovať na zmenu pozície prehrávania, tak je nutné pre `SeekBar` nastaviť špeciálny objekt, ktorý bude reagovať a spracovávať udalosť zmeny pozície prehrávania z posuvnej lišty. Súčasťou spracovania tejto udalosti je okrem samotnej zmeny pozície aj udelenie hodnotenia typu „SB“, ktoré sa ale pesničke udelí len vtedy, keď dôjde k posune na začiatok pesničky a ak súčasne platí, že pred zmenou pozície sa prehralo minimálne 25 % pesničky. V dolnej časti ovládacieho panela sa v jednej horizontálnej línii nachádzajú tri dôležité tlačidlá pre ovládanie prehrávania hudby:

- tlačidlo pre prechod na predchádzajúcu pesničku,
- tlačidlo pre spustenie a pozastavenie prehrávania, ktoré automaticky mení svoju ikonu na základe toho, či sa práve hudba prehráva,
- tlačidlo pre prechod na nasledujúcu pesničku.



Obr. 6.10: Obrázok vľavo zobrazuje celý panel s ovládacími prvkami hudobného prehrávača, ktorý zobrazuje základné informácie o práve prehrávanej pesničke a umožňuje ovládať prehrávanie hudby. Vyskakovacie menu, ktoré je možné zobraziť pomocou tlačidla vpravo hore umožňuje s touto pesničkou vykonávať rôzne akcie bez toho, aby bolo nutné hľadať jej položku v zozname. Toto vyskakovacie menu je zobrazené na obrázku v strede a vpravo.

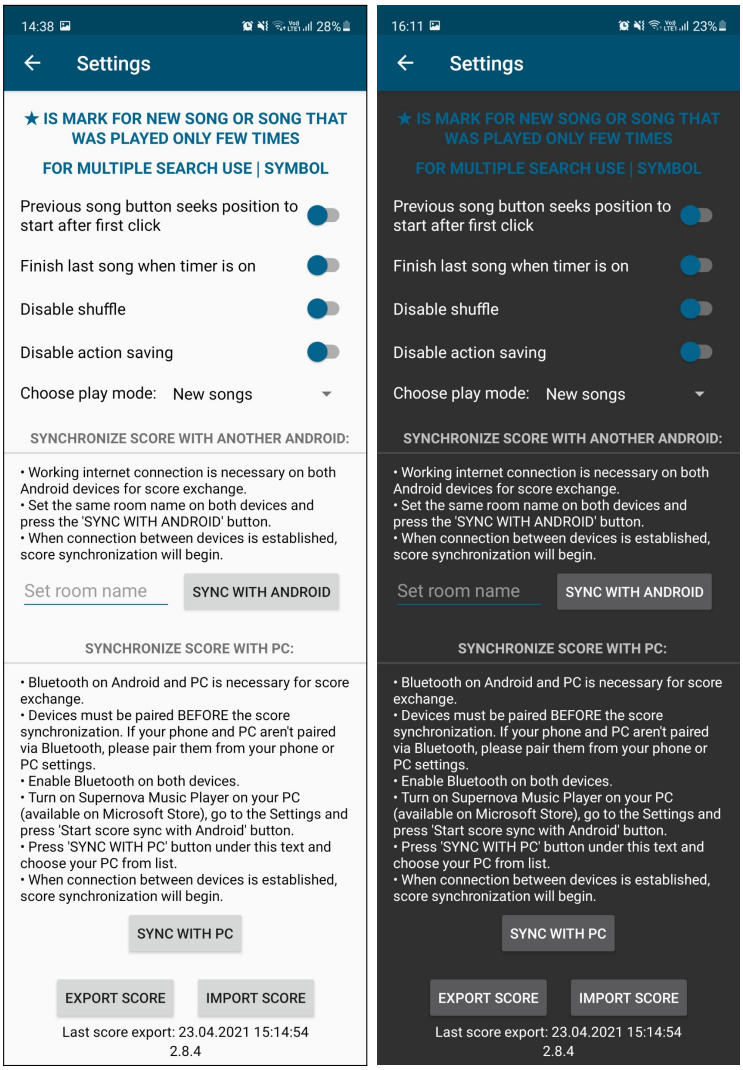
Obsah obrazovky nastavení (obr. 6.11) je možné rozdeliť do štyroch častí:

- časť s dodatočnými informáciami o fungovaní prehrávača,
- časť pre samotné nastavenia prehrávača,
- časť popisujúca a umožňujúca synchronizáciu skóre s iným Android zariadením a s počítačovou verziou prehrávača,
- časť pre import a export skóre.

Na vrchu obrazovky nastavení sa nachádzajú dve informácie o fungovaní prehrávača, ktoré by užívateľovi nemuseli byť jasné, keby neboli niekde explicitne uvedené. Prvá z týchto informácii vysvetľuje význam symbolu hviezdy („★“), ktorý sa môže vyskytovať nad hodnotou skóre v položke hlavného zoznamu pesničiek. Druhá informácia vysvetľuje, akým spôsobom je možné vykonávať viacnásobné vyhľadávanie.

Druhá časť obrazovky nastavení sa skladá hlavne z prepínačov, ktoré sú implementované ako `SwitchCompat`. Tieto prepínače môžu byť v jednom z dvoch stavov – vypnutom alebo zapnutom. Všetky prepínače sú predvolene vo vypnutom stave, no užívateľ môže kliknutím zmeniť stav prepínača, pričom nový stav si prehrávač uloží, aby pri každom zobrazení obrazovky nastavení bol vždy zobrazený aktuálny stav prepínača. Každý prepínač má špecifický význam pre fungovanie prehrávača, ktorý je popísaný naľavo od samotného prepínača. Konkrétne je na obrazovke nastavení zobrazený:

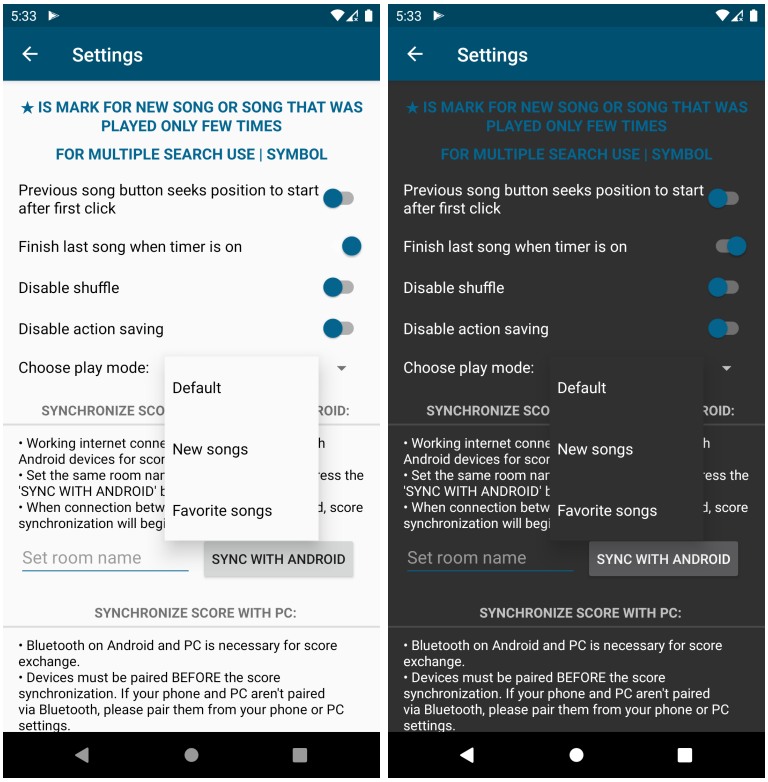
- prepínač pre úpravu správania tlačidla pre prechod na predchádzajúcu pesničku,
- prepínač pre úpravu správania časovača,
- prepínač pre vypnutie náhodného prehrávania,
- prepínač pre zabránenie ukladania nových hodnotení.



Obr. 6.11: Obrázky zobrazujú celú obrazovku nastavení vo svetlom a v tmavom režime. Táto obrazovka obsahuje časť s informáciami o fungovaní prehrávača, časť pre samotné nastavenia, časť pre synchronizáciu skóre a časť pre import a export skóre.

Pod prepínačmi sa nachádza rozbaľovací zoznam, pomocou ktorého je možné zmeniť mód prehrávania, implementovaný ako objekt triedy `Spinner` (obr. 6.12). Pre `Spinner` je nutné vytvoriť a nastaviť adaptér, ktorý s využitím viacerých XML súborov definuje aké položky sa budú v rozbaľovacom zozname nachádzať a tak isto aj to, ako budú tieto položky vyzerať. `Spinner` má predvolene zvolenú prvú položku rozbaľovacieho zoznamu. Po kliknutí na rozbaľovacie menu sa zobrazia všetky tri dostupné módy prehrávania, pričom užívateľ môže ďalším kliknutím jeden z nich vybrať alebo rozbaľovací zoznam skryť kliknutím mimo neho. Rovnako ako pri prepínačoch si prehrávač každú zmenu uloží a pri každom zobrazení obrazovky nastavení sa zobrazí naposledy zvolená hodnota.

Treťou a najväčšou časťou obrazovky nastavení je časť pre synchronizáciu skóre. Táto časť sa ďalej skladá z dvoch sekcií – jedna pre synchronizáciu skóre s ďalším Android zariadením a jedna pre synchronizáciu skóre s počítačom. Obidve sekcie obsahujú špecifické inštrukcie a ovládacie prvky pre spustenie daného typu synchronizácie. Keďže inštrukcie

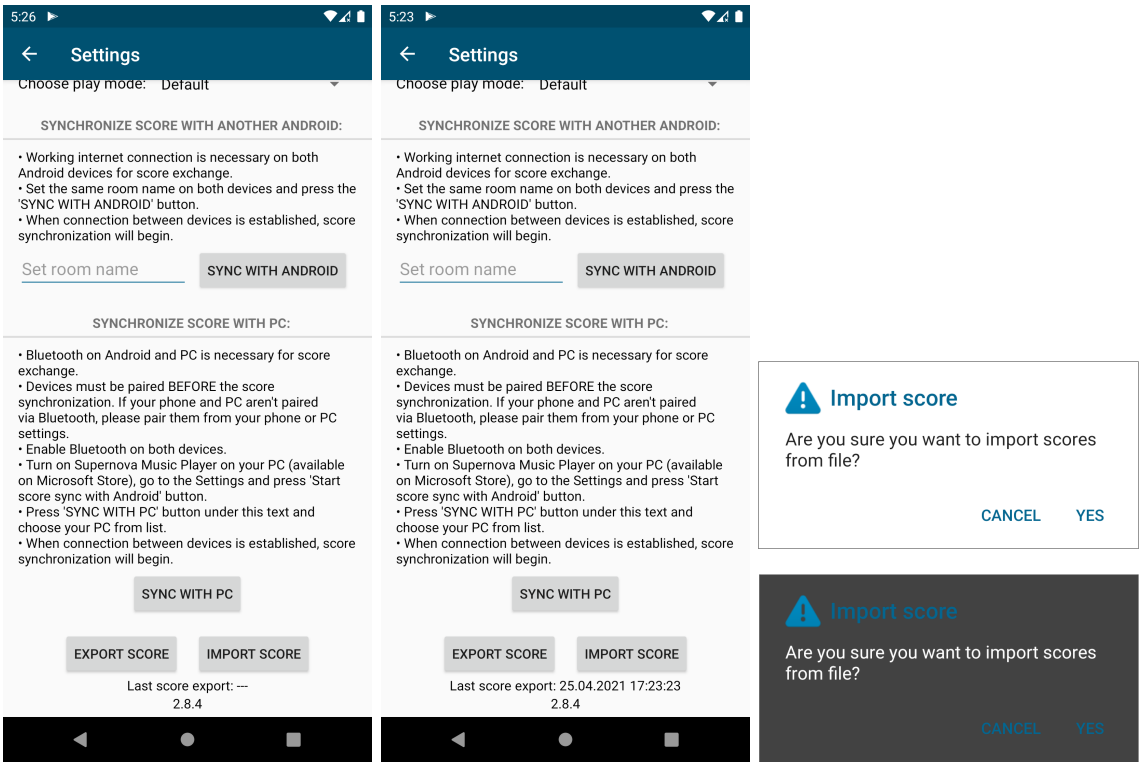


Obr. 6.12: Obrázok zobrazuje otvorený rozbaľovací zoznam vo svetlom aj tmavom režime, pomocou ktorého môže užívateľ zmeniť mód prehrávania.

pre spustenie synchronizácie zaberajú relatívne veľkú plochu obrazovky nastavení, tak je nutné pre túto obrazovku nastaviť v jej XML súbore `ScrollView`, ktorý umožňuje rolovanie obrazovky a teda zobrazenie celého jej obsahu. Princíp fungovania synchronizácie skóre je detailne popísaný v kapitole 7.

Poslednou časťou na obrazovke nastavení je časť pre import a export skóre. Základom tejto časti sú tlačidlá pre import a export skóre. Keď užívateľ stlačí tlačidlo exportu, tak sa spustí proces exportovania hodnotení všetkých pesničiek do súboru, ktorý je bližšie popísaný v kapitole 6.4. Užívateľ je informovaný ako o úspešnom exporte, tak aj v prípade vzniku chyby počas exportu pomocou objektu triedy `Toast`. Po úspešnom exporte sa pod tlačidlami taktiež zobrazí dátum a čas posledného exportu. Ak nebol export ešte nikdy vykonaný, tak sa tu namiesto dátumu zobrazuje text „—“, ktorý o doposiaľ nevykonanom exporte užívateľa informuje (obr. 6.13). Ak z úložiska nie je možné čítať alebo ak sa na úložisku nenachádza priečinok s názvom prehrávača obsahujúci súbor s názvom „score“ a užívateľ stlačí tlačidlo importu, tak je import okamžite ukončený a užívateľ informovaný o konkrétnej chybe cez `Toast`. V opačnom prípade sa užívateľovi zobrazí dialógové okno, ktoré užívateľa žiada o to, aby potvrdil vykonanie importu hodnotení zo súboru (obr. 6.13). Toto dialógové okno sa zobrazuje hlavne z toho dôvodu, že užívateľ môže toto tlačidlo stlačiť omylom a keby nemal možnosť túto akciu potvrdiť, tak by sa mohlo stať, že by mohol nechceným importom prísť o množstvo najnovšie uložených hodnotení, ak nebol export vykonaný dlhší čas. Užívateľ môže import potvrdiť stlačením pravého tlačidla dialógového okna alebo ho zrušiť kliknutím na ľavé tlačidlo dialógového okna, ako aj kliknutím na ľubovoľnú časť obrazovky nastavení mimo dialógové okno. Ak užívateľ túto akciu potvrdí, tak sa spustí proces importu, ktorý

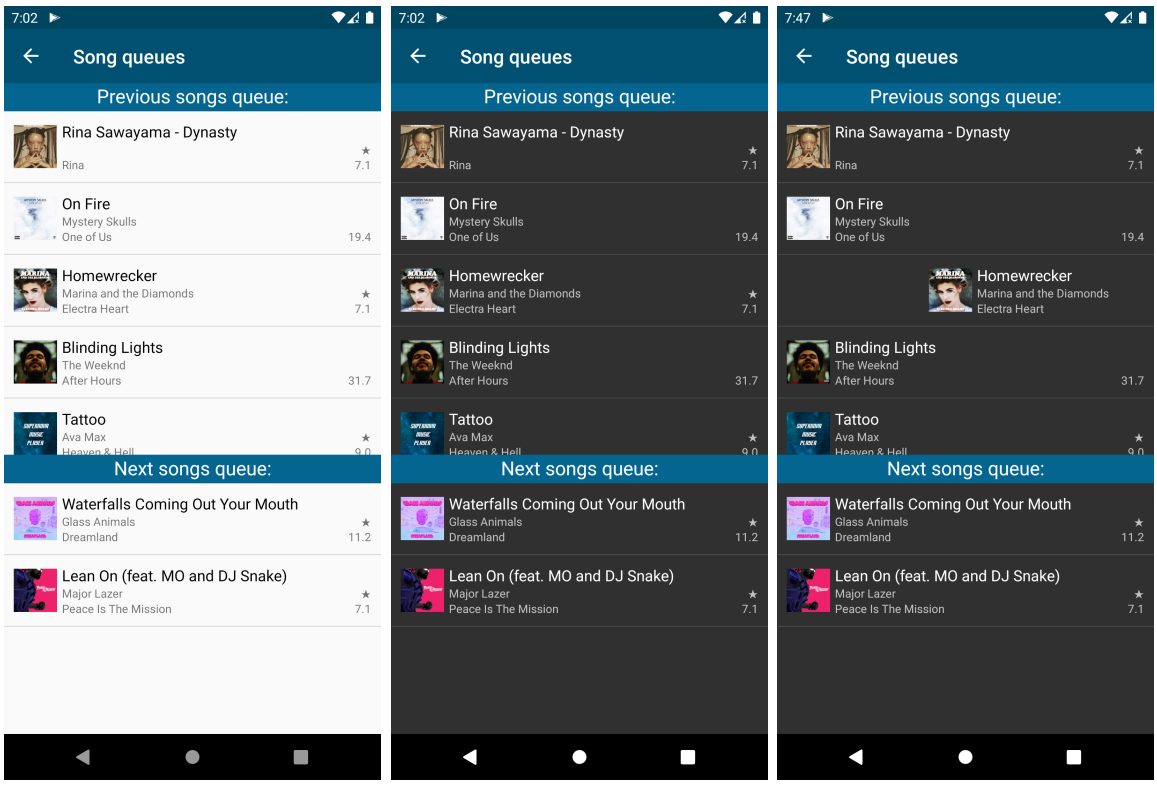
je rovnako ako export popísaný v kapitole 6.4. Pod dátumom posledného exportu sa ešte dodatočne nachádza číslo verzie hudobného prehrávača, aby mohol užívateľ skontrolovať to, akú verziu má aktuálne nainštalovanú.



Obr. 6.13: Ľavý obrázok zobrazuje situáciu, keď nebol ešte nikdy export hodnotení do súboru vykonaný a namiesto dátumu sa pod tlačidlami importu a exportu zobrazuje výplňový text. Stredný obrázok zobrazuje situáciu po vykonaní exportu, keď sa namiesto výplňového textu zobrazuje pod tlačidlami dátum a čas posledného vykonania exportu. Obrázky vpravo zobrazujú dialógové okno pre import, ktoré sa zobrazuje z toho dôvodu, aby sa import hodnotení nevykonal omylom.

Keď užívateľ z hlavného menu aplikácie zvolí, že chce zobraziť obrazovku so zoznamom predchádzajúcich a nasledujúcich pesničiek, tak predtým ako k tomuto prechodu z hlavnej obrazovky dôjde, aktivita hlavnej obrazovky získa tieto zoznamy zo služby, ktorá zabezpečuje prehrávanie a s ktorou táto aktivita priamo komunikuje. Aktivita hlavnej obrazovky následne predá tieto zoznamy zvolenej aktivite, ktorá ich na svojej obrazovke zobrazí (obr. 6.14). V hornej časti obrazovky je zobrazený zoznam predchádzajúcich pesničiek a v spodnej časti obrazovky je zobrazený zoznam nasledujúcich pesničiek, pričom toto rozdelenie je oznámené užívateľovi textom nad každým zoznamom. Zoznamy sú na obrazovke zobrazené v pomere 4:3 a to z toho dôvodu, že zoznam predchádzajúcich pesničiek sa na rozdiel od zoznamu nasledujúcich pesničiek napĺňa automaticky. Tým pádom má zoznam predchádzajúcich pesničiek vo väčšine prípadov viac položiek na zobrazenie ako zoznam nasledujúcich pesničiek, ktorý svojimi explicitnými akciami napĺňa užívateľ. Tieto dva zoznamy sú rovnako ako hlavný zoznam pesničiek implementované ako `RecyclerView`, naplnené cez rovnaký typ adaptéru a aj ich položky majú rovnakú štruktúru. Zoznamy majú okrem toho totožné aj správanie pre udalosť krátkeho kliknutia a pesnička na ktorej položku sa kliklo sa začne prehrávať. Zmenou oproti hlavnému zoznamu je, že položky týchto zozna-

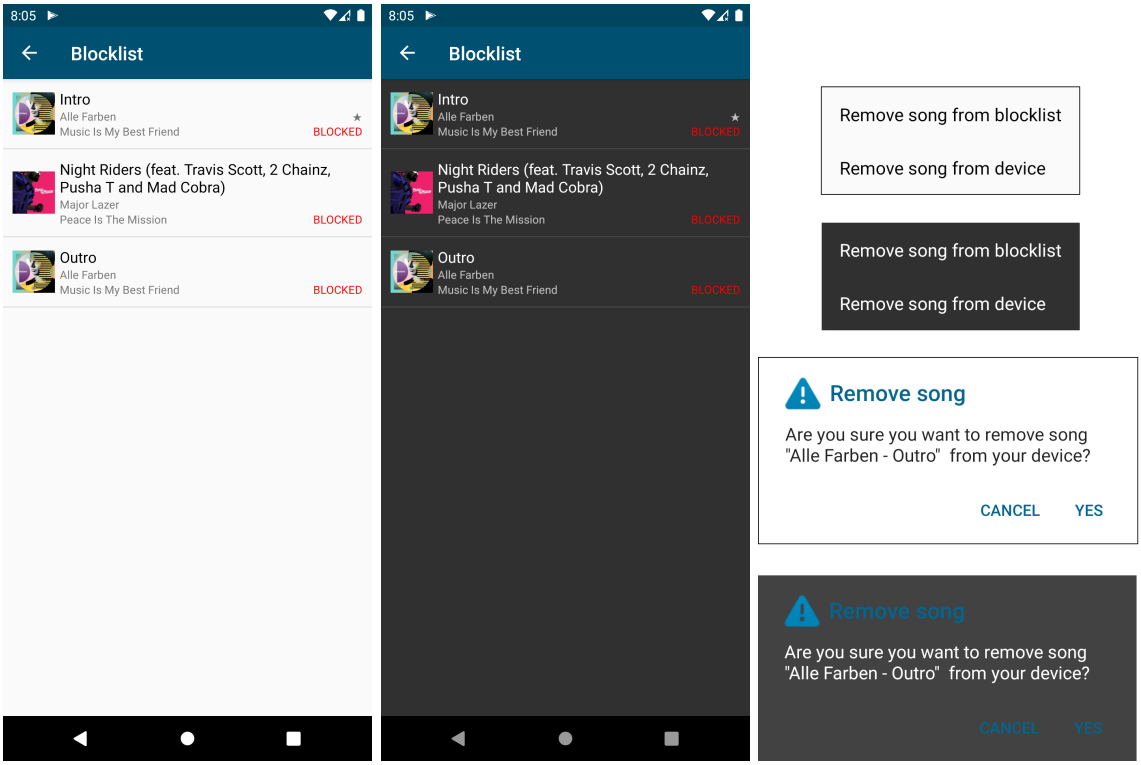
mov nijak nereagujú na udalosť dlhého kliknutia. Tieto dva zoznamy sa od seba navzájom odlišujú v správaní pri udalosti potiahnutia položky zľava doprava (obr. 6.14). Potiahnutie položky zoznamu nasledujúcich pesničiek spôsobí to, že daná položka bude z tohto zoznamu odstránená. Týmto spôsobom môže užívateľ zo zoznamu odstrániť tie pesničky, u ktorých si ich budúce prehranie rozmyslel. Pri potiahnutí položky zoznamu predchádzajúcich pesničiek dôjde k rovnakému správaniu ako v hlavnom zozname a pesnička teda bude pridaná do zoznamu nasledujúcich pesničiek, čo je na tejto obrazovke, na rozdiel od tej hlavnej, aj viditeľné. K aktualizácii obsahu obidvoch zoznamov dochádza taktiež pri každej zmene práve prehrávanej pesničky.



Obr. 6.14: Obrázky zobrazujú obrazovku so zoznamom prechádzajúcich a nasledujúcich pesničiek. Obrázok vpravo znázorňuje to, že položky obidvoch zoznamov podporujú okrem udalosti krátkeho kliknutia aj udalosť potiahnutia položky zľava doprava, pričom táto udalosť má pre každý zoznam vlastné správanie.

Obrazovka so zoznamom blokovaných pesničiek (obr. 6.15) pozostáva výlučne z tohto zoznamu, ktorý sa získa tým, že sa z hlavného zoznamu pesničiek vyberú tie pesničky, ktoré sú zablokované. Účelom tejto obrazovky je zobraziť všetky blokované pesničky pokope, aby ich užívateľ nemusel jednotlivo hľadať v hlavnom zozname, ktorý môže obsahovať veľký počet pesničiek. Zoznam blokovaných pesničiek je implementovaný tým istým spôsobom ako v predchádzajúcich prípadoch. Od ostatných zoznamov sa líši správaním pri vzniknutých udalostiach. Tento zoznam nijak nereaguje na udalosť krátkeho kliknutia a udalosť posunutia položky zľava doprava ani nepodporuje. Pri dlhom kliknutí sa zobrazí vyskakovacie menu (obr. 6.15) podobné tomu z hlavnej obrazovky. Rozdielom je to, že toto vyskakovacie menu umožňuje vykonať len dve akcie a to konkrétne odblokovanie a odstránenie zvolenej pes-

ničky zo zariadenia. Zvolenie týchto akcií spustí správanie, ktoré je totožné ako to z hlavnej obrazovky. Navyše sa po vykonaní zvolenej akcie daná položka zo zoznamu odstráni.



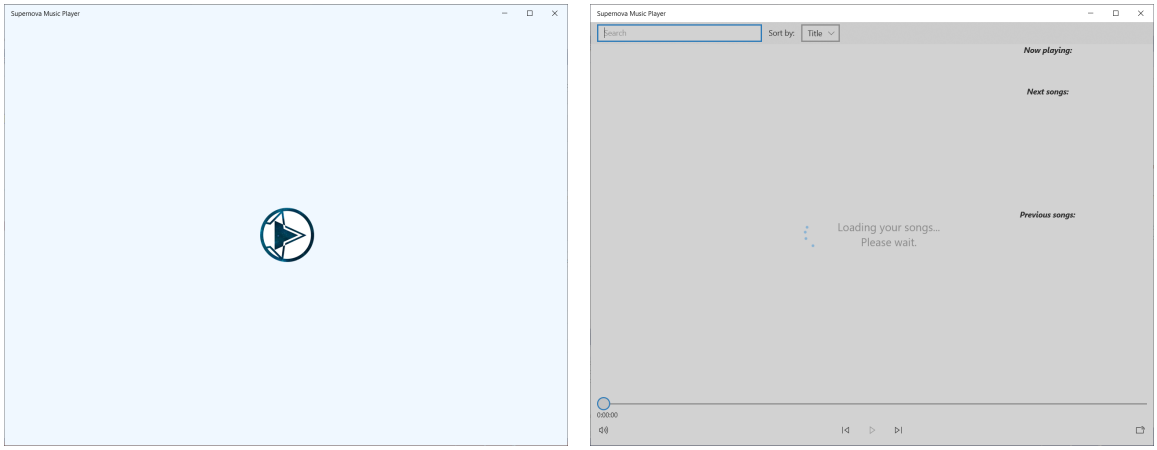
Obr. 6.15: Obrázok vľavo a v strede zobrazuje obrazovku so zoznamom blokovaných pesničiek. Prvé dva obrázky vpravo hore zobrazujú vyskakovacie menu, ktoré sa zobrazí po dlhom kliknutí na položku tohto zoznamu. Na spodných dvoch obrázkoch vpravo je dialógové okno, ktoré sa zobrazí, keď užívateľ zvolí akciu pre vymazanie pesničky zo zariadenia z vyskakovacieho menu a cez ktoré musí užívateľ svoju voľbu potvrdiť, ak chce pesničku zo zariadenia skutočne vymazať.

6.1.2 Grafické užívateľské rozhranie počítačovej verzie prehrávača

Vo všeobecnosti platí, že na obrazovke počítača je možné zobraziť väčšie množstvo informácii ako na obrazovke chytrého telefónu a práve preto tvoria grafické užívateľské rozhranie počítačovej aplikácie len dve obrazovky:

1. hlavná obrazovka,
2. obrazovka nastavení.

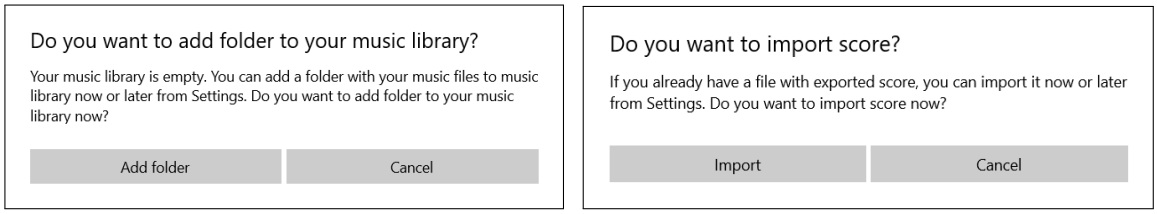
Predtým, ako sa po spustení aplikácie zobrazí hlavná obrazovka, tak sa na pár sekúnd zobrazí automaticky vygenerovaná úvodná obrazovka s logom aplikácie (obr. 6.16). Keď sa táto obrazovka skryje, tak sa síce zobrazí hlavná obrazovka, ktorá ale ešte nemá načítané potrebné dáta, a tak sa nad ňou zobrazí priehľadná obrazovka načítavania, ktorá oznámi užívateľovi, že prebieha načítanie pesničiek (obr. 6.16). Špeciálnym prípadom je prvé spustenie aplikácie po inštalácii. Ak hudobná knižnica daného zariadenia neobsahuje žiadne priečinky s hudbou, tak sa zobrazí dialógové okno (`ContentDialog`), ktoré na to užívateľa



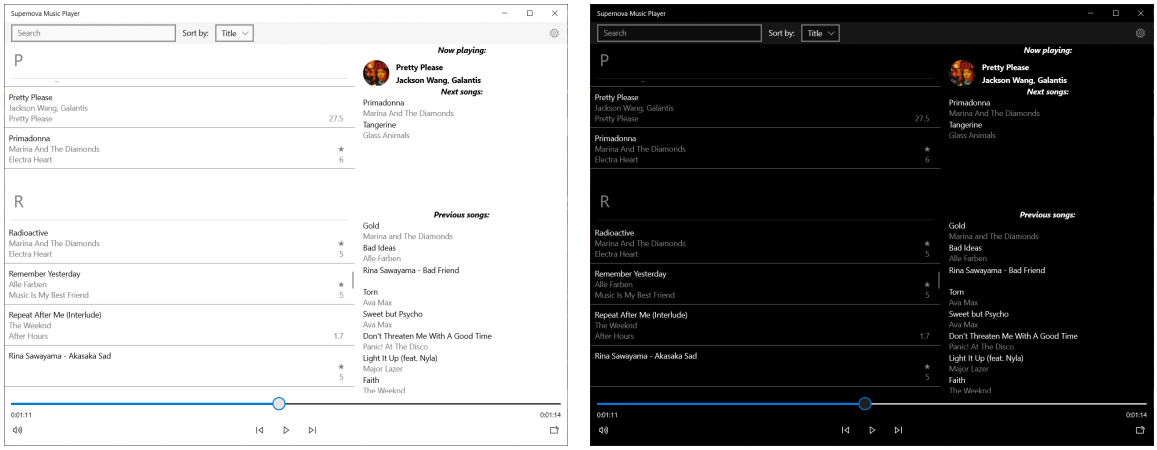
Obr. 6.16: Na ľavom obrázku sa nachádza úvodná obrazovka s logom aplikácie, ktorá sa objaví hneď po spustení aplikácie. Vedľa nej je zobrazená hlavná obrazovka počítačovej verzie hudobného prehrávača, ktorá ešte ale nemá načítané potrebné dáta, a tak sa nad ňou zobrazuje obrazovka načítavania.

upozorní a opýta sa ho, či chce do knižnice nejaký priečinok pridať (obr. 6.17). Ak sa užívateľ rozhodne pridať priečinok do hudobnej knižnice, tak sa mu zobrazí nové dialógové okno (obr. 6.30), ktoré mu to umožní a pesničky z vybraného priečinku budú načítané. Po dokončení načítavania všetkých potrebných dát sa nimi hlavná obrazovka naplní a obrazovka načítavania sa skryje. Ak bola z počítača načítaná aspoň jedna pesnička, tak sa pri prvom spustení prehrávača po tomto kroku zobrazí ešte jedno dialógové okno. Pomocou tohto dialógové okna môže užívateľ vykonať import hneď pri prvom spustení prehrávača, ak už má súbor s exportovanými hodnoteniami (obr. 6.17). Plne načítanú hlavnú obrazovku počítačovej verzie prehrávača (obr. 6.18) je možné rozdeliť vertikálne na 4 časti:

- systémový titulný panel,
- titulný panel aplikácie,
- panel s pesničkami,
- ovládací panel.



Obr. 6.17: Obrázok vľavo zobrazuje dialógové okno, ktoré sa zobrazí pri prvom spustení aplikácie v takom prípade, keď hudobná knižnica zariadenia neobsahuje žiadne priečinky, pričom cez toto okno je možné priečinok s hudbou do knižnice pridať. Obrázok vpravo zobrazuje dialógové okno, ktoré sa zobrazí pri prvom spustení aplikácie a pomocou ktorého môže užívateľ hneď vykonať import, ak už má súbor s exportovanými hodnoteniami.

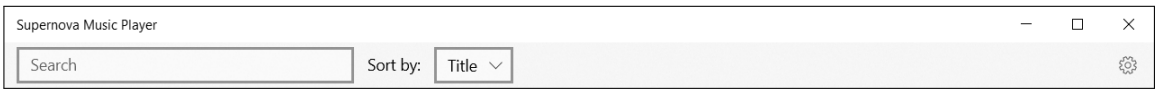


Obr. 6.18: Obrázky zobrazujú celú hlavnú obrazovku počítačovej verzie hudobného prehrávača vo svetlom a v tmavom režime. Hlavná obrazovka pozostáva zhora nadol zo systémového titulného panelu, titulného panelu aplikácie, panelu s pesničkami a z ovládacieho panelu. Všetky tieto časti sú detailnejšie zobrazené na ďalších obrázkoch.

Systémový titulný panel (obr. 6.19) sa automaticky zobrazuje na vrchu každej obrazovky. V pravom hornom rohu zobrazuje názov aplikácie a v ľavom rohu tri základné systémové tlačidlá. Konkrétne sa jedná o:

- tlačidlo pre minimalizovanie aplikácie,
- tlačidlo pre maximalizovanie okna alebo pre zobrazenie okna vo veľkosti, ktorú naposledy nastavil sám užívateľ,
- tlačidlo pre vypnutie aplikácie.

Keď užívateľ stlačí tlačidlo pre vypnutie, tak sa pred samotným vypnutím vyvolá v triede reprezentujúcej aplikáciu udalosť `OnSuspending`, ktorá umožňuje uložiť všetky dáta potrebné pre ďalší beh aplikácie. Jedná sa o rovnaké dáta ako pri vypnutí mobilnej verzie prehrávača a ich bližší popis sa nachádza v úvode kapitoly 5.2.



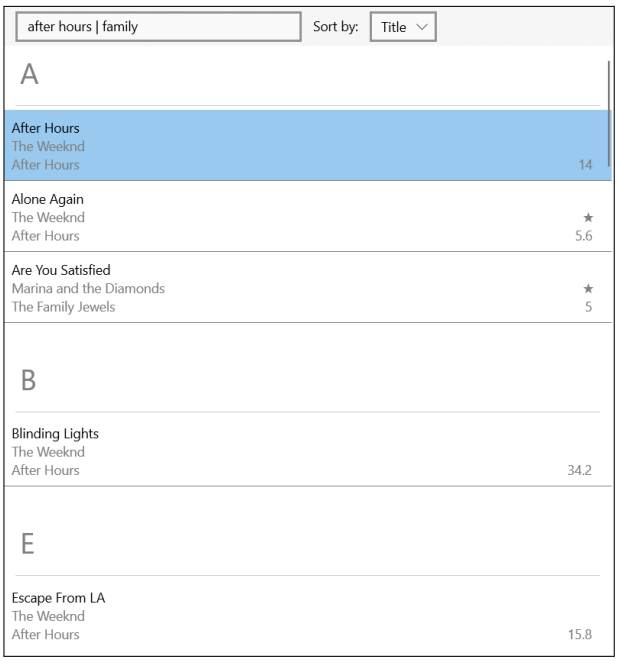
Obr. 6.19: Obrázok detailne zobrazuje dve horné časti hlavnej obrazovky – systémový titulný panel a titulný panel aplikácie. Systémový titulný panel zobrazuje názov aplikácie a umožňuje minimalizovať, maximalizovať alebo vypnúť aplikáciu. Titulný panel aplikácie je zobrazený pod ním a umožňuje spustenie vyhľadávania, usporiadanie zoznamu pesničiek a prechod do nastavení.

Titulný panel aplikácie (obr. 6.19) je implementovaný ako `NavigationView`, čo znamená, že sa nachádza hneď pod systémovým titulným panelom. Pre prehrávač poskytuje 3 dôležité funkcie:

1. vyhľadávanie,
2. usporiadanie hlavného zoznamu pesničiek,

3. prechod na obrazovku nastavení.

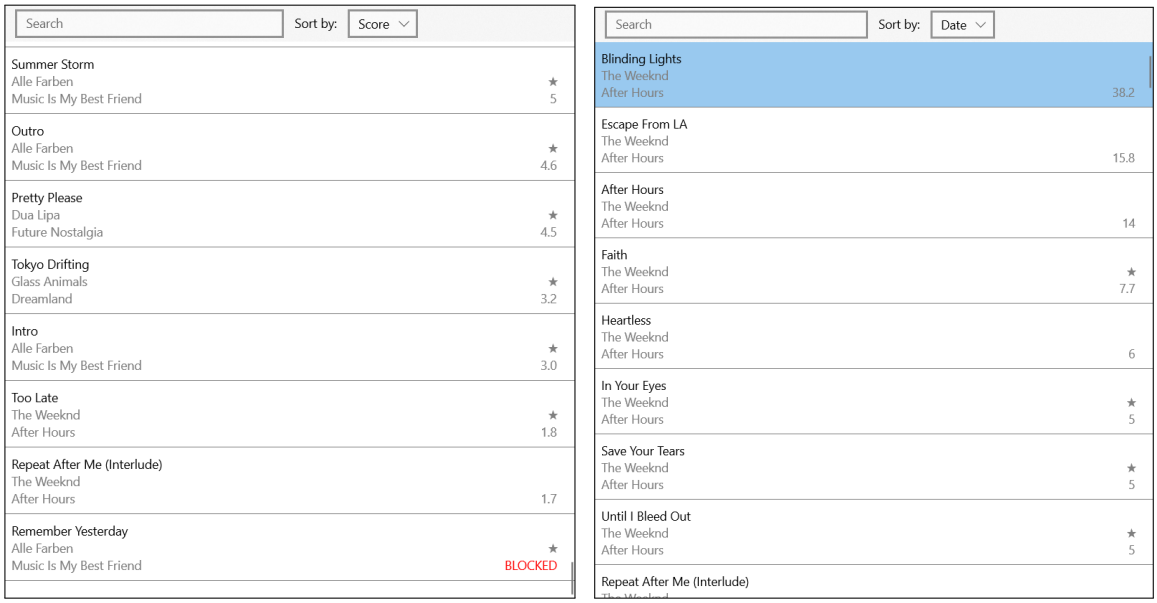
Vyhľadávanie (obr. 6.20) je jednoducho implementované pomocou textového poľa (`TextBox`), ktoré pomocou udalosti `TextChanged` dokáže reagovať na zmenu textu v tomto poli. Po zadaní prvého znaku sa spustí proces vyhľadávania a v pravom rohu textového poľa sa automaticky zobrazí krížik, ktorým je možné obsah vyhľadávacieho poľa zmazať a vyhľadávanie tým ukončiť. Výsledky vyhľadávania sú zoradené podľa aktuálneho usporiadania a zobrazené v hlavnom zozname pesničiek, tak isto ako v mobilnej verzii. Pre zlepšenie užívateľskej skúsenosti bolo nutné v počítačovej verzii implementovať to, aby sa zoznam s výsledkami vyhľadávania aktualizoval len vtedy, keď sa po zmene textu vyhľadávania zmenia aj výsledky vyhľadávania.



Obr. 6.20: Obrázok zobrazuje viacnásobné vyhľadávanie, ktoré je realizované cez textové pole umiestnené v titulnom paneli aplikácie, pričom výsledky sú zobrazované v hlavnom zozname pesničiek pod ním. Obrázok rovnako znázorňuje usporiadanie zoznamu pesničiek podľa ich názvu. Pri tomto usporiadaní je zoznam rozdelený na sekcie podľa počiatočného písmena názvu pesničky, pričom každá sekcia má svoju hlavičku.

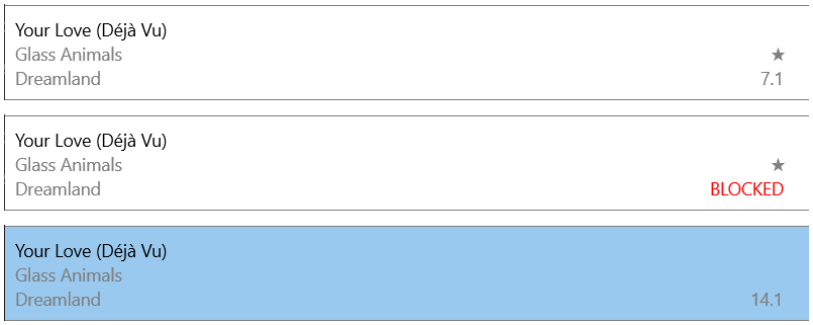
Usporiadanie hlavného zoznamu pesničiek je možné cez rozbaľovací zoznam (`ComboBox`). Tento zoznam má ako predvolenú hodnotu nastavené zoradenie podľa názvu pesničky. Užívateľ môže zoznam kliknutím rozbaliť a ďalším kliknutím vybrať iný typ usporiadania. V takomto prípade sa vyvolá udalosť `SelectionChanged`, novú hodnotu si prehrávač uloží a zoznam pesničiek sa podľa nej zoradí okamžite a aj pri ďalšom spustení aplikácie. Pri usporiadaní podľa názvu pesničky (obr. 6.20) sa zoznam rozdelí na sekcie podľa počiatočného písmena názvu pesničky. Každá sekcia má hlavičku, ktorá výrazne informuje o tom, aké počiatočné písmeno bude ďalej v zozname nasledovať. Na vrchu hlavného zoznamu pesničiek sa pri tomto usporiadaní zobrazuje špeciálna hlavička, ktorá vždy užívateľa informuje o tom, na akom písmene sa aktuálne nachádza (možné vidieť na obr. 6.23). Vzhľad týchto hlaviček je definovaný v XAML súbore priamo vnútri elementu hlavného zoznamu pesničiek a to pomocou jeho detských elementov `ListView.GroupStyle` a `ListView.ItemsPanel`. Uspo-

riadanie podľa skóre pesničky (obr. 6.21) je v počítačovej verzii špecifické tým, že blokované pesničky sa zobrazujú až na konci zoznamu pesničiek, čo je spôsobené mierne odlišnou implementáciou objektov reprezentujúcich pesničku. Pri usporiadaní hlavného zoznamu podľa dátumu pridania hudobného súboru do zariadenia (obr. 6.21) sa okrem samotného dátumu berie do úvahy aj čas pridania daného súboru do zariadenia. Poslednou funkciou titulného panelu aplikácie je prechod na obrazovku nastavení. Tento prechod umožňuje tlačidlo s ikonou ozubeného kolieska v pravom rohu titulného panelu aplikácie. Keď užívateľ toto tlačidlo stlačí, tak špeciálny objekt triedy `Frame` zabezpečí prechod do nastavení.



Obr. 6.21: Ľavý obrázok zobrazuje usporiadanie pesničiek hlavného zoznamu podľa ich skóre, pričom zablokované pesničky sa pri tomto usporiadaní zobrazujú až na konci. Pravý obrázok zobrazuje usporiadanie toho istého zoznamu podľa dátumu a času pridania hudobného súboru do zariadenia. Na tomto obrázku teda vidno, ktorý album bol do zariadenia pridaný ako posledný.

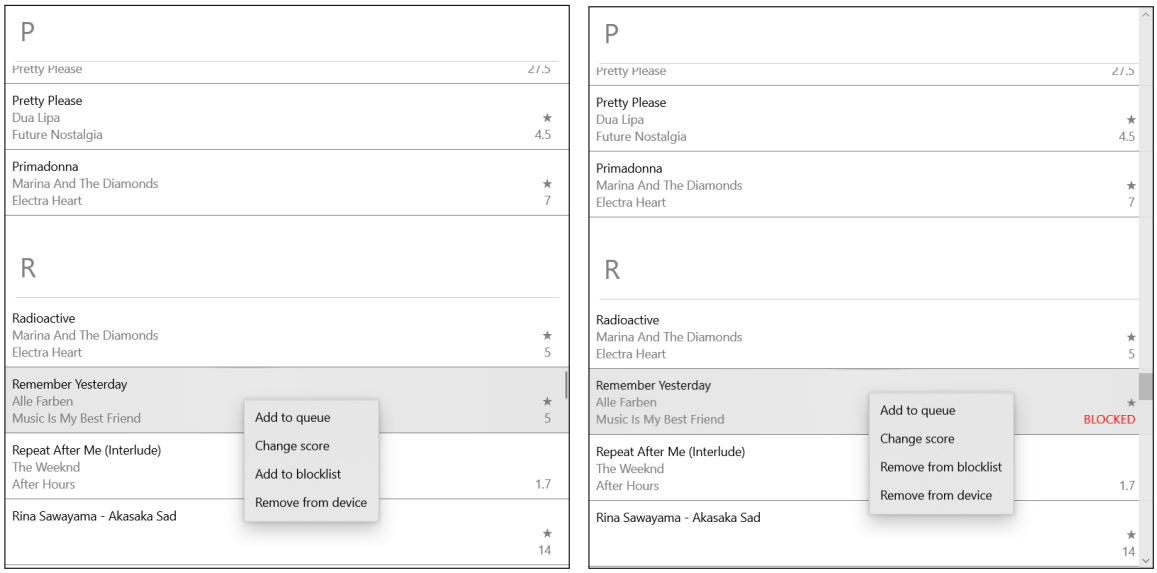
Panel s pesničkami je možné rozdeliť na dva stĺpce. Celý širší ľavý stĺpec zaberá hlavný zoznam pesničiek. V pravom stĺpci sa zhora nadol zobrazujú informácie o práve prehrávanej pesničke, zoznam nasledujúcich pesničiek a zoznam predchádzajúcich pesničiek. Hlavný zoznam pesničiek je implementovaný tak isto ako všetky ostatné zoznamy pesničiek cez `ListView`, pričom dáta sa napĺňajú cez väzbu dát pomocou XAML atribútu `ItemsSource`. Definícia štruktúry položky zoznamu je súčasťou každého `ListView` a vykonáva sa v súbore XAML pomocou jeho detského elementu `ListView.ItemTemplate`. Každá položka hlavného zoznamu (obr. 6.22) obsahuje vo svojej ľavej časti nasledujúce informácie zoradené pod sebou: názov pesničky, názov jej interpreta a albumu. V pravom dolnom rohu sa zobrazuje skóre pesničky s maximálne jedným desatinným číslom, nad ktorým môže byť zobrazený symbol hviezdy označujúci pesničky s nízkym počtom prehratí. Názov pesničky je rovnako ako vo verzii pre Android vyznačený čiernou farbou, kým farba textu pre všetky ostatné informácie je sivá. Výnimkou sú len zablokované pesničky, u ktorých sa namiesto skóre zobrazuje červený text informujúci užívateľa o tom, že daná pesnička je zablokovaná a tým pádom sa nebude prehrávať (obr. 6.22). Rámčeky položiek a ďalšie zmeny vzhľadu položky ako celku sú nastavené cez detský element zoznamu `ListView.ItemContainerStyle`.



Obr. 6.22: Na obrázku hore je možné vidieť klasickú položku pesničky v hlavnom zozname pesničiek. Na obrázku v strede je znázornená položka zablokovanej pesničky a na dolnom obrázku sa nachádza zvýraznená položka pesničky, na ktorú užívateľ klikol ľavým tlačidlom myši. Na dolnom obrázku už bola daná pesnička prehratá aspoň štyrikrát, a preto sa u nej už symbol hviezdy nezobrazuje.

Položka hlavného zoznamu dokáže reagovať na niekoľko udalostí, ktoré sú vyvolané počítačovou myšou, touchpadom alebo iným podobným zariadením. Pri jednoduchom kliknutí ľavého tlačidla myši sa daná položka zvýrazní modrým pozadím (obr. 6.22). Užívatelia počítačových aplikácií sú väčšinou zvyknutí, že určitú akciu spustí až dvojklik tohto tlačidla. Spustenie prehrávania vybranej pesničky zo zoznamu a všetkých akcií s tým spojených vyvolá práve až udalosť dvojkliku ľavého tlačidla myši. Keď sa kurzor myši nachádza nad hlavným zoznamom pesničiek a keď užívateľ klikne na stredné tlačidlo myši, tak sa ikona kurzoru zmení, čím sa dá užívateľovi najavo, že teraz môže zoznamom rolovať ťahaním myši smerom nadol alebo nahor. Ikona kurzoru v tvare šípky sa mení s každou zmenou smeru, pričom vždy zobrazuje aktuálny smer rolovania. Rolovanie zoznamu pesničiek sa ukončí ľubovoľným kliknutím myši alebo keď kurzor opustí okno aplikácie. Rýchlo prechádzať zoznamom je možné aj pomocou posuvnej lišty, ktorá sa automaticky zobrazí, keď užívateľ prejde kurzorom na pravý okraj zoznamu (obr. 6.23).

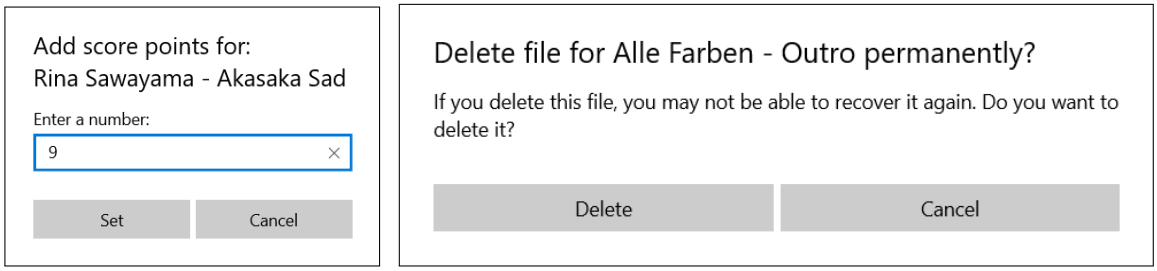
Po kliknutí na pravé tlačidlo myši sa pri položke zoznamu objaví vyskakovacie menu (obr. 6.23), ktoré je implementované ako objekt triedy `MenuFlyout` a jeho štruktúra je definovaná vnútri ďalšieho detského elementu zoznamu `ListView.Resources`. Toto menu obsahuje presne tie isté akcie ako vyskakovacie menu v ovládacom paneli mobilnej aplikácie: pridanie pesničky do zoznamu nasledujúcich pesničiek, explicitná zmena skóre užívateľom, zablokovanie alebo prípadné odblokovanie pesničky a odstránenie pesničky zo zariadenia. Pridanie pesničky do zoznamu nasledujúcich pesničiek je nutné vykonať z vyskakovacieho menu, keďže platforma UWP nepodporuje udalosť posunutia položky zoznamu v počítačových aplikáciách. Zároveň sa tento typ udalosti v počítačových aplikáciách ani nezvykne používať v takom rozsahu ako v mobilných aplikáciách. Pesnička sa po pridaní do zoznamu nasledujúcich pesničiek zobrazí v tomto zozname v pravej časti hlavnej obrazovky a je jej udelené odpovedajúce hodnotenie. Keď užívateľ zvolí z menu úpravu skóre, tak sa mu v strede okna aplikácie zobrazí dialógové okno (obr. 6.24), ktoré pozostáva z troch častí. Na vrchu dialógového okna je text, ktorým je užívateľ informovaný o tom, že upravuje skóre pre zvolenú pesničku. Pod týmto textom sa nachádza textové pole, ktoré je inicializované buď nulou, alebo hodnotou, ktorú naposledy nastavil užívateľ. Užívateľ môže do tohoto poľa zadať hodnotu, o ktorú chce zväčšiť alebo zmenšiť skóre pesničky, pričom toto pole je implementované takým spôsobom, aby do neho bolo možné zadať len čísla a znamienko mínus. Keď užívateľ nechá toto pole prázdne alebo keď zadá hodnotu, ktorá nereprezen-



Obr. 6.23: Obrázky zobrazujú vyskakovacie menu, ktoré sa zobrazí po kliknutí na položku hlavného zoznamu pesničiek pravým tlačidlom myši. Obrázok vľavo zobrazuje toto menu pred zablokovaním danej pesničky a obrázok vpravo zobrazuje rovnaké menu po zablokovaní pesničky. Na obrázku vpravo je možné taktiež vidieť zobrazenú posuvnú lištu pre rýchly prechod zoznamom.

tuje celé číslo (napr. „-0-0-“), tak prehrávač automaticky pri potvrdení nastaví hodnotu na nulu. Pod textovým poľom sa vľavo nachádza tlačidlo pre potvrdenie a uloženie zadanej hodnoty a vpravo sa nachádza tlačidlo pre zrušenie vykonávania zmeny skóre. Zobrazovanie dialógových okien je taktiež možné skryť pomocou klávesy „Esc“. Ďalšou akciou vyskakovacieho menu je zablokovanie alebo odblokovanie pesničky, pričom vo vyskakovacom menu sa vždy zobrazuje relevantný text podľa toho, či je zvolená pesnička zablokovaná alebo nie (obr. 6.23). Samotná akcia len udelí alebo odstráni hodnotenie pre blokovanie a aktualizuje sa informácia o skóre v danej položke. Pri odstránení pesničky zo zariadenia sa zobrazí dialógové okno (obr. 6.24), ktoré varuje užívateľa, že ak sa rozhodne odstrániť túto pesničku, tak jej súbor nemusí byť možné znovu získať. Súčasne toto dialógové okno žiada užívateľa, aby svoju voľbu potvrdil ľavým tlačidlom alebo zrušil pravým. Ak užívateľ vymazanie potvrdí, tak sa spustí samotný proces odstránenia tohto súboru, ktorý je popísaný v kapitole 6.2.

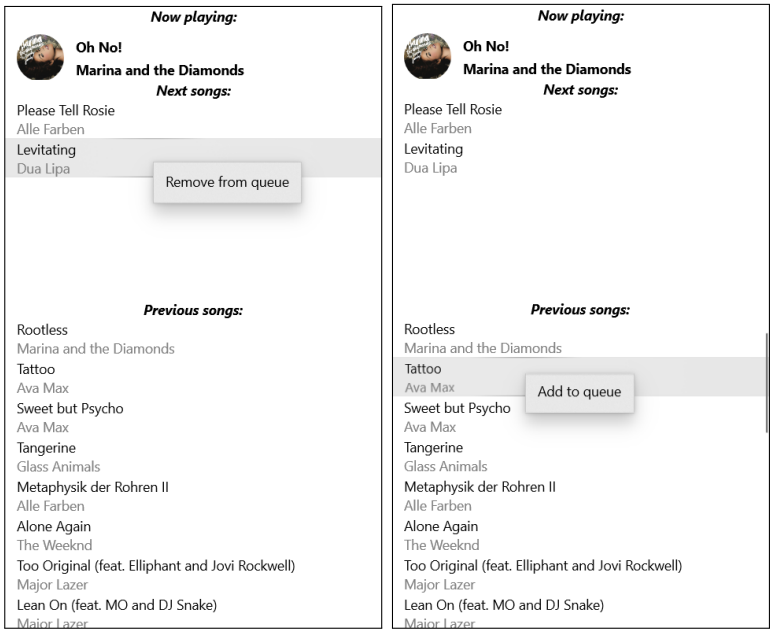
Každá časť pravého stĺpca panelu s pesničkami (obr. 6.25) je označená nadpisom, ktorý určuje obsah danej časti. Pre každú časť tohto stĺpca taktiež platí, že ich obsah je aktualizovaný pri každej zmene práve prehrávanej pesničky a to bez ohľadu na to, či má užívateľ práve zobrazenú hlavnú obrazovku alebo nie. Na vrchu pravého stĺpca panelu s pesničkami sa zobrazujú informácie o práve prehrávanej pesničke. Všetky tieto informácie sa získajú a zobrazia tesne predtým, ako sa pesnička začne prehrávať. Medzi tieto informácie patrí názov pesničky a názov jej interpreta, ktoré sú vyznačené tučným písmom. Naľavo od týchto informácii sa v tejto časti zobrazuje aj obrázok pesničky v kruhovom výseku. Pod touto časťou sa nachádza zoznam s nasledujúcimi pesničkami. Od hlavného zoznamu pesničiek sa líši hlavne tým, že jeho položky obsahujú výlučne názov pesničky a interpreta, aby mohlo byť naraz zobrazených čo najviac položiek tohto zoznamu. Zoznam nasledujúcich pesničiek reaguje totožne ako hlavný zoznam pri vzniku udalosti jednoduchého kliknutia a dvojkliku ľavého tlačidla myši na položku zoznamu. Tento zoznam nepodporuje rolovanie zoznamom



Obr. 6.24: Vľavo sa nachádza dialógové okno, pomocou ktorého môže užívateľ zmeniť hodnotu skóre pesničky a to zadaním celého čísla do textového poľa a potvrdením cez príslušné tlačidlo. Vpravo je dialógové okno, ktoré sa užívateľovi zobrazí pri pokuse o vymazanie pesničky zo zariadenia. Toto dialógové okno varuje užívateľa, že ak sa rozhodne odstrániť túto pesničku, tak jej súbor nemusí byť možné znovu získať a súčasne ho žiada, aby svoju voľbu potvrdil alebo zrušil tlačidlom.

po stlačení stredného tlačidla myši, no užívateľ má stále k dispozícii posuvnú lištu na pravom okraji zoznamu pre rýchly prechod zoznamom, ak má zoznam dostatočne veľký počet položiek. Po kliknutí pravým tlačidlom na položku zoznamu sa zobrazí vyskakovacie menu, no toto menu umožňuje vykonať len jedinú akciu – odstrániť danú pesničku zo zoznamu nasledujúcich pesničiek (obr. 6.25). Spodnú časť pravého stĺpca vypĺňa zoznam predchádzajúcich pesničiek. Tento zoznam zaberá väčšiu plochu stĺpca ako zoznam nad ním a to z toho dôvodu, že pre väčšinu užívateľov bude obsahovať vo väčšine prípadov viac položiek ako zoznam nasledujúcich pesničiek. Ďalším a súčasne aj posledným rozdielom medzi týmito dvoma zoznamami je akcia, ktorú je možné vykonať z vyskakovacieho menu, ktoré sa zobrazí po kliknutí pravým tlačidlom na položku tohto zoznamu. Položka zoznamu predchádzajúcich pesničiek umožňuje akciou z vyskakovacieho menu vložiť zvolenú pesničku do zoznamu nasledujúcich pesničiek, ak si chce užívateľ danú pesničku vypočuť znova (obr. 6.25).

Ovládací panel (obr. 6.26) zaberá celú šírku dolnej časti hlavnej obrazovky a tvorí ho jediný prvok – `MediaPlayerElement`. Ten pozostáva z niekoľkých menších ovládacích prvkov, pre zobrazenie ktorých je nutné v XAML súbore hlavnej obrazovky nastaviť atribút `AreTransportControlsEnabled` prvku `MediaPlayerElement` na kladnú pravdivostnú hodnotu. Na vrchu ovládacieho panelu sa nachádza posuvná lišta, ktorou je možné zmeniť aktuálnu pozíciu prehrávania či už kliknutím ľavým tlačidlom myši na novú pozíciu alebo potiahnutím kruhu, ktorý na tejto lište graficky znázorňuje aktuálnu pozíciu prehrávania. Aby bolo možné pesničke udeliť hodnotenie za posun aktuálnej pozície prehrávania na začiatok pesničky, tak je nutné získať túto posuvnú lištu v kóde ako objekt triedy `Slider`, ktorému sa následne pridelí požadované správanie na udalosť zmeny aktuálnej pozície prehrávania. `MediaPlayerElement` automaticky zobrazuje vľavo pod posuvnou lištou aká časť pesničky sa už prehrala v hodinách, minútach a sekundách. Na opačnej strane sa zobrazuje v rovnakom formáte čas zostávajúci do dokončenia prehrávania danej pesničky. Tieto informácie, ako aj aktuálna pozícia prehrávania na posuvnej lište sa aktualizujú automaticky a to aj v prípade, keď aplikácia beží na pozadí alebo keď má užívateľ otvorenú obrazovku nastavení. V ľavom dolnom rohu sa nachádza tlačidlo pre úpravu hlasitosti prehrávania (obr. 6.26). Hlasitosť je možné pomocou neho úplne stlmiť alebo nastaviť jej hodnotu v rozmedzí od nuly po sto, pričom predvolená hodnota je päťdesiat. Platforma UWP nepodporuje prístup k hodnote systémovej hlasitosti a k jej zmenám, no ak užívateľ zvýši hlasitosť prehrávania pomocou tohto tlačidla, tak je pesničke maximálne raz za jej aktuálne prehrávanie udelené



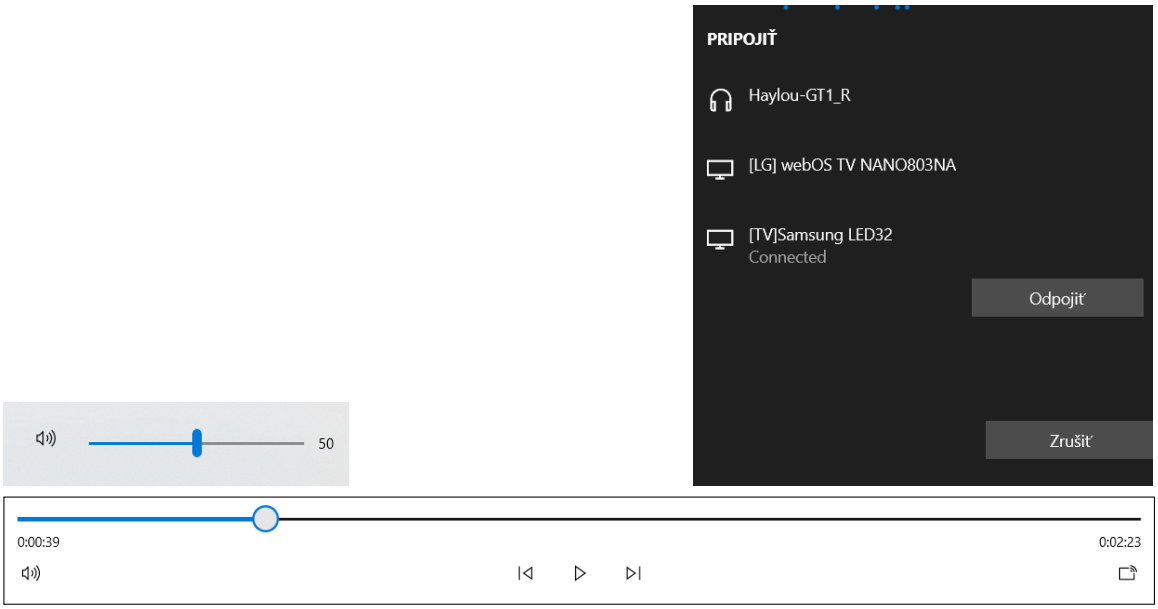
Obr. 6.25: Obrázky zobrazujú pravý stĺpec panelu s pesničkami, ktorý obsahuje informácie o práve prehrávanej pesničke, zoznam nasledujúcich pesničiek a zoznam predchádzajúcich pesničiek. Ľavý obrázok navyše zobrazuje vyskakovacie menu, ktoré sa zobrazí po kliknutí pravým tlačidlom myši na položku zoznamu nasledujúcich pesničiek. Pravý obrázok zobrazuje to isté len pre zoznam predchádzajúcich pesničiek.

hodnotenie za zvýšenie hlasitosti prehrávania. V strede spodnej časti ovládacieho panelu sa nachádzajú tri základné tlačidlá pre ovládanie prehrávania:

- tlačidlo pre prechod na predchádzajúcu pesničku,
- tlačidlo pre spustenie alebo pozastavenie prehrávania,
- tlačidlo pre prechod na nasledujúcu pesničku.

Predvolené správanie tlačidiel pre prechod na prechádzajúcu alebo nasledujúcu pesničku je, že sa prehrá predchádzajúca alebo nasledujúca pesnička v poradí. Aby bolo možné definovať správanie týchto tlačidiel podľa algoritmu výberu pesničky (kapitola 5.2), tak je najskôr nutné vytvoriť novú triedu, ktorá bude dediť z triedy `MediaTransportControls` a ktorá sprístupní udalosť kliknutia na tieto dve tlačidlá. Následne sa táto trieda nastaví v XAML súbore vnútri elementu `MediaPlayerElement.TransportControls`, čo je detský element prvku `MediaPlayerElement`. Vďaka sprístupneným udalostiam je možné tlačidlá správne zobraziť a v kóde definovať ich správanie podľa vyššie popísaného algoritmu. V pravom dolnom rohu ovládacieho panelu sa zobrazuje tlačidlo, ktoré umožňuje prehrávať hudbu cez iné zariadenie v okolí, napríklad cez chytrý televízor (obr. 6.26). Po kliknutí na toto tlačidlo sa zobrazí zoznam zariadení, ktoré by mali byť schopné prehrávať hudbu a ktoré sa buď nachádzajú v blízkom okolí a sú práve zapnuté, alebo sú spárované s používateľovým počítačom. Keď užívateľ zvolí niektoré zo zariadení, tak prebehne pokus o pripojenie. Ak tento pokus zlyhá, tak sa v zozname pri danom zariadení zobrazí chybová hláška. V opačnom prípade sa spustí prehrávanie na pripojenom zariadení, pričom ovládať prehrávanie je možné ako cez počítač, tak cez pripojené zariadenie. Keď chce užívateľ prehrávanie cez pripojené

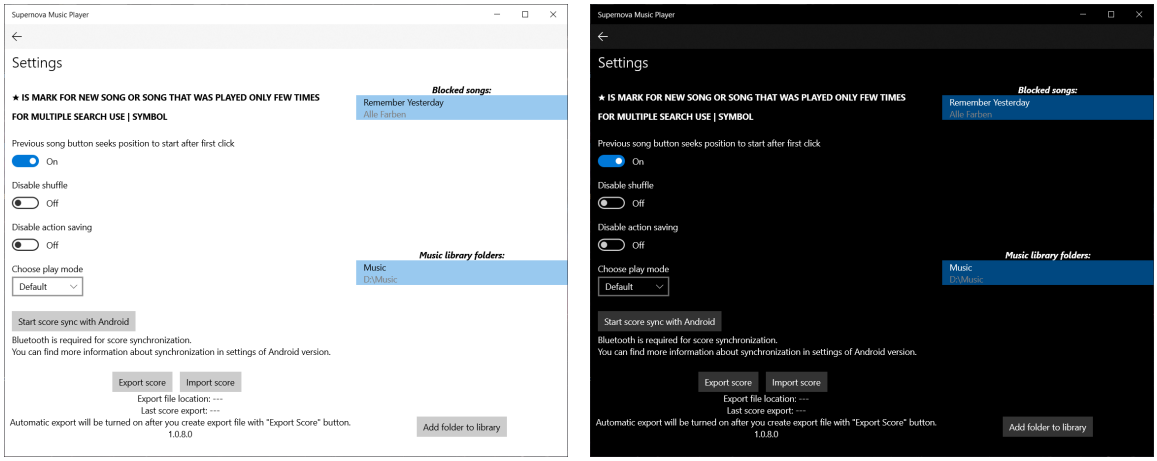
zariadenie zrušiť, tak je nutné sa od tohto zariadenia odpojiť cez zoznam zariadení a následne sa bude hudba prehrávať znova cez počítač. Všetku túto funkcionalitu zabezpečuje `MediaPlayerElement` automaticky sám.



Obr. 6.26: Obrázok zobrazuje ovládací panel, ktorý slúži na ovládanie prehrávania hudby. Okrem ovládania prehrávania hudby umožňuje tento panel upraviť hlasitosť prehrávania. Keď užívateľ klikne na tlačidlo vľavo dolu, tak sa zobrazí malé dialógové okno pre samotnú úpravu hlasitosti, ktoré je zobrazené nad ovládacím panelom vľavo. Keď užívateľ klikne na tlačidlo, ktoré sa na ovládacom paneli nachádza vpravo dolu, tak sa zobrazí zoznam zariadení, ku ktorým by malo byť možné sa pripojiť a prehrávať cez ne hudbu. Tento zoznam zariadení je zobrazený nad ovládacím panelom vpravo.

Obrazovka nastavení (obr. 6.27) má rovnako ako hlavná obrazovka na svojom vrchu systémový titulný panel a titulný panel aplikácie. Kým systémový titulný panel obrazovky nastavení sa v ničom nelíši od toho na hlavnej obrazovke, tak titulný panel aplikácie obrazovky nastavení slúži výlučne pre návrat na hlavnú obrazovku pomocou tlačidla s ikonou šípky v ľavom rohu (obr. 6.28). Návrat na hlavnú obrazovku je okrem toho možný aj stlačením bočného tlačidla myši alebo stlačením kombinácie klávesy „Alt“ s klávesou ľavej šípky, pričom samotný prechod späť vykoná znova objekt triedy `Frame`. Pod titulnými panelmi je zobrazený názov obrazovky pre lepšiu orientáciu a pod ním sa nachádza už samotný obsah obrazovky nastavení, ktorý je možné znova rozdeliť na ľavý a pravý stĺpec. Ak nie je možné naraz zobraziť celý obsah obrazovky nastavení kvôli nízkej výške okna aplikácie, tak je možné obsah prechádzať kolieskom myši alebo posuvnou lištou, ktorá sa zobrazí pri pravom okraji okna aplikácie vždy, keď je to nutné.

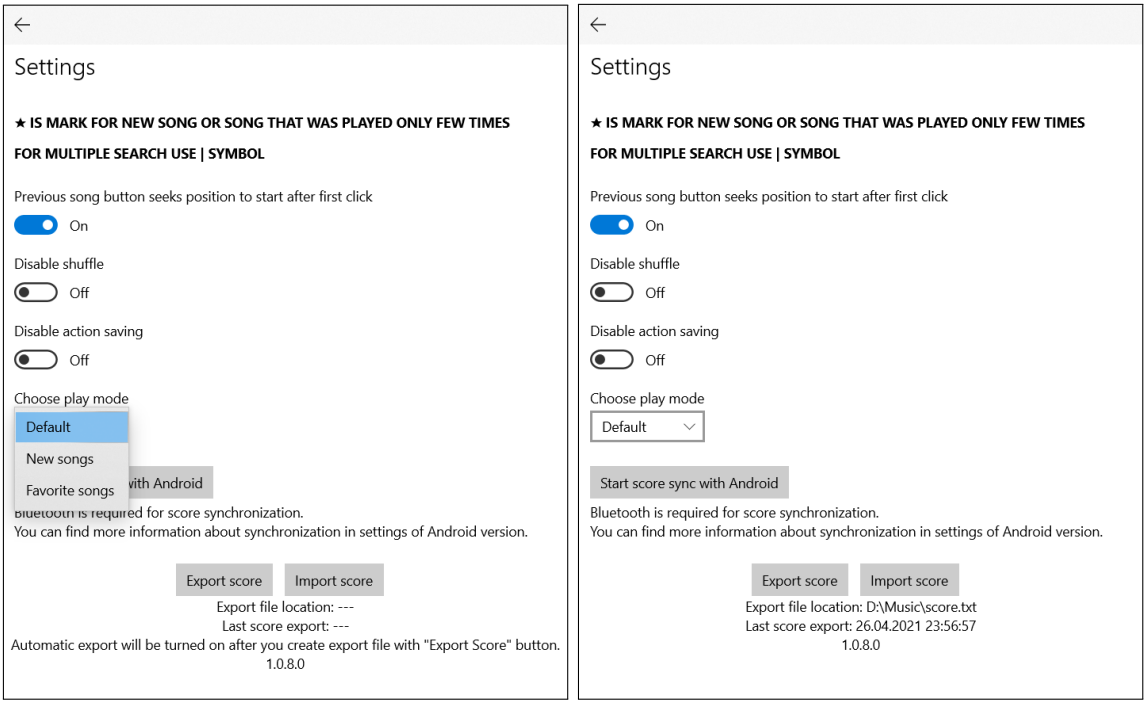
Ľavý stĺpec obrazovky nastavení (obr. 6.28) je obsahovo takmer totožný s obrazovkou nastavení v mobilnej verzii prehrávača. Na jeho vrchu sú tučným písmom vyznačené informácie o fungovaní prehrávača, pod ktorými sa nachádzajú prepínače pre ovplyvnenie správania hudobného prehrávača, ktoré sú implementované ako `ToggleSwitch`. Konkrétne sa tu nachádza prepínač pre ovplyvnenie správania tlačidla pre prechod na predchádzajúcu pesničku, prepínač pre vypnutie náhodného prehrávania a prepínač pre zabránenie ukladania nových hodnotení. Každému prepínaču je možné nastaviť hlavičku, ktorá sa au-



Obr. 6.27: Obrázky zobrazujú celú obrazovku nastavení hudobného prehrávača vo svetlom a v tmavom režime. Obrazovku nastavení tvoria predovšetkým prepínače pre zmenu správania prehrávača, časť pre synchronizáciu skóre, časť pre import a export skóre, zoznam blokovaných pesničiek a zoznam priečinkov, ktoré tvoria hudobnú knižnicu. Túto obrazovku je možné rozdeliť na dva stĺpce, pričom každý z nich je detailnejšie zobrazený na ďalších obrázkoch.

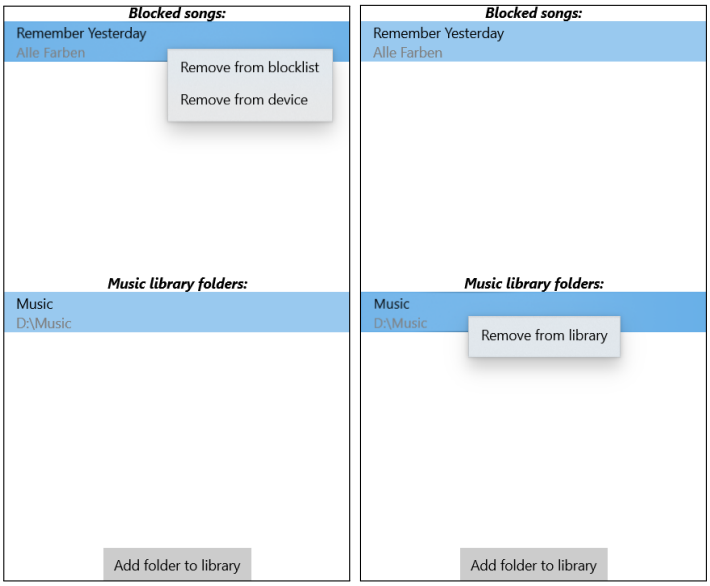
tomaticky zobrazuje nad samotným prepínačom a ktorá informuje o jeho význame. Všetky prepínače sú predvolene vo vypnutom stave, no pri každej zmene sa nový stav uloží pre okamžité ovplyvnenie daného správania, ako aj pre správne zobrazenie stavu prepínača. Z rovnakého dôvodu sa ukladá aj zmena zvolenej položky v rozbaľovacom zozname pre mód prehrávania, ktorý sa nachádza pod prepínačmi. Rozbaľovací zoznam je na platforme UWP implementovaný ako objekt triedy `ComboBox` a je mu tak isto možné nastaviť hlavičku. Jeho položky, ktoré reprezentujú jednotlivé módy prehrávania, sú definované ako jeho detské elementy, pričom predvolene je zvolená prvá položka. Pod rozbaľovacím zoznamom sa nachádza časť pre synchronizáciu skóre s Android zariadením. Tá pozostáva z tlačidla, ktorého stlačenie umožňuje spustenie tohto procesu a z textu, ktorý informuje užívateľa o tom, že pre synchronizáciu skóre medzi verziami prehrávača je Bluetooth nevyhnutný. Tento text mu taktiež oznamuje, že viac informácii o synchronizácii je dostupných v Android verzii prehrávača. Pod týmto textom sa môže v prípade problému pri synchronizácii zobraziť ďalší text, ktorý o danom probléme informuje užívateľa výraznou červenou farbou. Poslednou časťou ľavého stĺpca je sekcia pre import a export skóre. Táto sekcia pozostáva hlavne z dvoch tlačidiel, ktoré umožňujú spustiť príslušnú akciu. Popis obidvoch akcií sa nachádza v kapitole 6.4. Pod týmito tlačidlami sa nachádza text, ktorý informuje užívateľa o tom, kedy bol export vykonaný naposledy a kde sa súbor s exportovanými hodnoteniami nachádza. Ak nebol export ešte vykonaný alebo ak bol súbor s exportovanými hodnoteniami z disku odstránený, tak sa namiesto týchto údajov zobrazuje hodnota „—“. Súčasne sa tu v takomto prípade zobrazuje informácia o tom, že export sa bude vykonávať automaticky až po tom, ako export vykoná prvýkrát sám užívateľ. Po vytvorení súboru s exportovanými hodnoteniami sa tento text skryje a do predchádzajúcich sú doplnené konkrétne hodnoty (obr. 6.28). Na záver sa v tejto sekcii zobrazuje taktiež číslo aktuálnej verzie prehrávača.

Pravý stĺpec obrazovky nastavení (obr. 6.29) pozostáva z dvoch oddelených častí – zo zoznamu zablokovaných pesničiek a zo zoznamu priečinkov, ktoré tvoria hudobnú knižnicu. Zoznam zablokovaných pesničiek má mnoho rovnakých vlastností ako zoznam predchá-



Obr. 6.28: Obrázok zobrazuje ľavý stĺpec obrazovky nastavení, ktorý obsahuje dodatočné informácie o fungovaní prehrávača, prepínače a rozbaľovací zoznam pre úpravu správania prehrávača, časť pre synchronizáciu a časť pre import a export skóre. Obrázky taktiež zobrazujú titulný panel aplikácie, ktorý slúži pre návrat na hlavnú obrazovku. Obrázok vľavo zobrazuje tento stĺpec obrazovky nastavení pred vykonaním exportu a obrázok vpravo ho zobrazuje po vykonaní exportu hodnotení do súboru. Obrázok vľavo taktiež zobrazuje otvorený rozbaľovací zoznam s módmi prehrávania.

dzajúcich a nasledujúcich pesničiek z hlavnej obrazovky. Jediným výrazným rozdielom je udalosť kliknutia pravým tlačidlom myši na položku zoznamu. Pri tejto udalosti sa zobrazí vyskakovacie menu (obr. 6.29), ktoré ponúka akciu pre odblokovanie a odstránenie pesničky zo zariadenia. Správanie týchto akcií je identické ako správanie im odpovedajúcich akcií dostupných z vyskakovacieho menu hlavného zoznamu pesničiek. Spodnú časť pravého stĺpca zaberá zoznam priečinkov, ktoré spolu so svojimi súbormi tvoria hudobnú knižnicu. Položka tohto zoznamu obsahuje názov priečinku, ktorý je vyznačený čiernou farbou. Pod názvom priečinku sa nachádza cesta k tomuto priečinku, ktorej text má menej výraznú sivú farbu. Položka zoznamu reaguje na jedinú udalosť, ktorou je kliknutie na zvolený priečinok pravým tlačidlom myši. Pri tejto udalosti sa zobrazí vyskakovacie menu s jednou akciou (obr. 6.29). Tá umožňuje odstránenie tohto priečinku z hudobnej knižnice, čo znamená, že pesničky z tohto priečinku sa nebudú viac prehrávať a ani zobrazovať v prehrávači. K tejto časti taktiež patrí tlačidlo pod zoznamom priečinkov, ktorý naopak umožňuje pridávanie ďalších priečinkov do hudobnej knižnice. Správou hudobnej knižnice sa bližšie zaoberá nasledujúca kapitola.



Obr. 6.29: Na obrázku sa nachádza pravý stĺpec obrazovky nastavení, ktorý obsahuje zoznam blokovaných pesničiek a zoznam priečinkov, ktoré tvoria hudobnú knižnicu. Ku spodnému zoznamu patrí taktiež tlačidlo, ktorým môže užívateľ pridávať ďalšie priečinky do hudobnej knižnice. Vľavo je zobrazené vyskakovacie menu, ktoré sa zobrazí po kliknutí na položku zoznamu blokovaných pesničiek pravým tlačidlom myši. Vpravo sa zobrazuje to isté, ale pre zoznam s priečinkami hudobnej knižnice.

6.2 Správa hudobných súborov

Získanie hudobných súborov zo zariadenia je základom každého funkčného prehrávača bez ohľadu na zvolenú vývojovú platformu. Princíp, akým sa tieto súbory zo zariadenia získavajú sa ale medzi platformou Android a UWP značne líši. Hudobný prehrávač pre chytré telefóny získava hudobné súbory pomocou objektu `ContentResolver`. `ContentResolver` je jediná, globálna inštancia v aplikácii, ktorá poskytuje prístup k poskytovateľom obsahu (kapitola 4.1.4). Tento objekt prijíma požiadavky od klienta, ktoré následne predá poskytovateľovi obsahu s príslušnou zodpovednosťou. `ContentResolver` zahŕňa metódy pre vytváranie, čítanie, aktualizovanie alebo mazanie dát daného poskytovateľa obsahu, pričom každej tejto metóde je predané URI, ktorým sa špecifikuje poskytovateľ obsahu, s ktorým má `ContentResolver` interagovať [8]. Získanie hudobných súborov prebieha pomocou metódy `query()` tohto objektu, ktorej parameter `uri` sa nastaví na hodnotu `android.provider.MediaStore.Audio.Media.EXTERNAL_CONTENT_URI`. Použitím tohto URI bude `ContentResolver` interagovať s tým poskytovateľom obsahu, ktorý má na starosti všetky audio súbory. Aby sa z telefónu získali len pesničky a nie audioknihy, podcasty, tóny zvonenia a podobne, tak je nutné nastaviť taktiež parameter `selection` vyššie spomínanej metódy na hodnotu `MediaStore.Audio.Media.IS_MUSIC + "!= 0"`. Takto nastavená metóda vráti prehrávaču všetky súbory pesničiek, ktoré sa na zariadení nachádzajú. Prehrávač týmito súbormi prejde pomocou objektu triedy `Cursor`, získa z nich potrebné dáta a zobrazí ich v hlavnom zozname pesničiek tak, ako to je popísané v kapitolách 5.1 a 6.1.1.

Najväčším rozdielom pri získavaní hudobných súborov z počítača je to, že aplikácie vyvíjané na platforme UWP väčšinou nemajú prístup ku všetkým súborom, ktoré sa na počítači nachádzajú. Prístup ku všetkým súborom užívateľa je síce možný cez špeciálne

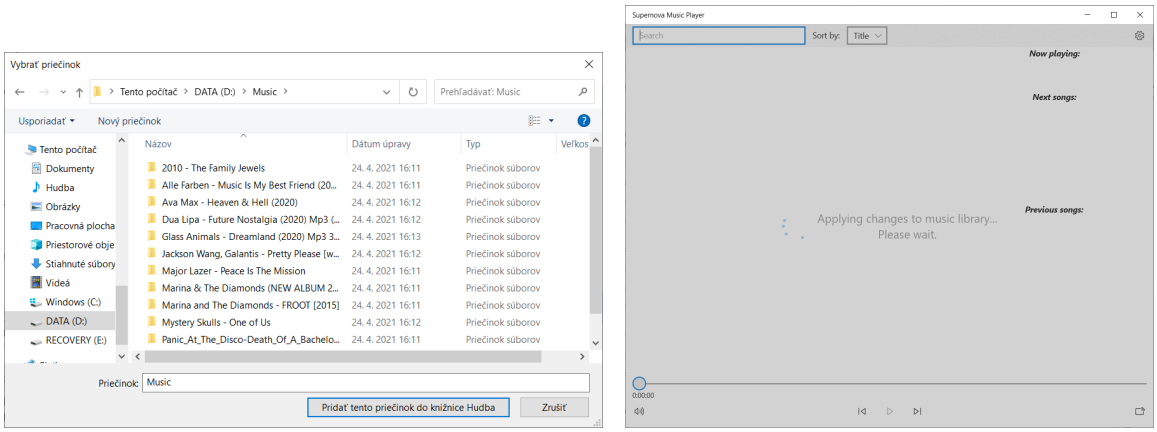
oprávnenie `broadFileSystemAccess`, no toto oprávnenie by malo byť použité len v tých prípadoch, kedy aplikácia toto povolenie nevyhnutne potrebuje pre svoju činnosť. Užívateľ môže udelenie tohto oprávnenia kedykoľvek zrušiť zo systémových nastavení a súčasne použitie tohto oprávnenia musí byť schválené spoločnosťou Microsoft pred vydaním aplikácie na Microsoft Store, pričom vývojár musí uviesť prečo jeho aplikácia toto oprávnenie potrebuje a k akému účelu ho využije [2, App capability declarations[6]]. Namiesto toho je pre získanie hudobných súborov možné použiť hudobnú knižnicu. Knižnica predstavuje na platforme UWP virtuálnu kolekciu priečinkov, ktorá automaticky zahŕňa systémový priečinok danej knižnice (v tomto prípade systémový priečinok „Hudba“) a všetky ďalšie priečinky, ktoré do knižnice pridal užívateľ [2, Files and folders in the Music, Pictures, and Videos libraries[7]]. Hudobná knižnica umožňuje aplikácii získať prístup ku všetkým jej súborom a to bez dodatočnej interakcie s užívateľom. Pre prístup k hudobnej knižnici je stále nutné nastaviť oprávnenie v manifeste aplikácie či už cez užívateľské rozhranie, ktoré poskytuje vývojové prostredie Visual Studio, alebo priamo v XML súbore manifestu pridaním nasledujúceho riadku: `<Capabilities> <uap:Capability Name="musicLibrary"/> </Capabilities>`. Rozdiel oproti oprávneniu `broadFileSystemAccess` je v tom, že oprávnenie pre hudobnú knižnicu môže využívať každá aplikácia voľne bez toho, aby bolo nutné schválenie jej použitia pri vydávaní aplikácie v Microsoft Store [2, App capability declarations[6]]. Ďalšou výhodou tohto prístupu je to, že hudobná knižnica je zdieľaná medzi všetkými aplikáciami, ktoré ju používajú. Ak ju užívateľ naplnil priečinkami s hudbou v jednej aplikácii, tak všetky hudobné súbory z týchto priečinkov budú automaticky načítané aj v ďalších aplikáciách, ktoré tento princíp používajú a užívateľ ju tým pádom nemusí napĺňať pre každú aplikáciu zvlášť. To isté platí aj o zmenách v hudobnej knižnici – ak užívateľ pridá alebo odstráni niektorý priečinok z hudobnej knižnice v jednej aplikácii, tak sa tieto zmeny automaticky prejavia aj vo všetkých ostatných aplikáciách.

Pred samotným získaním súborov z hudobnej knižnice sa vytvorí objekt triedy `QueryOptions`, pomocou ktorého sa nastaví zoradenie získaných súborov, ako aj to, aký typ hudobných súborov sa získa – počítačová verzia hudobného prehrávača momentálne podporuje nasledujúce typy: mp3, mp4, wma, m4a a flac. Ak priečinok hudobnej knižnice obsahuje vo svojom vnútri ďalšie priečinky s hudbou, tak sa pomocou tohto objektu nastaví, aby sa hudobné súbory získali aj zo všetkých týchto podpriečinkov. Hudobné súbory sa z knižnice získajú ako zoznam objektov triedy `StorageFile` pomocou nasledujúceho výrazu: `KnownFolders.MusicLibrary.CreateFileQueryWithOptions().GetFilesAsync()`, pričom vytvorený objekt triedy `QueryOptions` sa použije ako parameter metódy `CreateFileQueryWithOptions()`. Prehrávač po získaní súborov z hudobnej knižnice nimi začne prechádzať a získavať z nich dáta potrebné pre svoje fungovanie. Objekt triedy `StorageFile` avšak sám o sebe neobsahuje tú najdôležitejšiu informáciu pre fungovanie prehrávača – jedinečný identifikátor súboru. Identifikátor súboru sa napriek tomu dá získať volaním funkcie `GetFileInformationByHandle()`, ktorú je ale nutné naimportovať z Win32 knižnice s názvom „kernel32.dll“. Každý `StorageFile` pri tom slúži na vytvorenie parametru pre danú funkciu. Získaný identifikátor je reprezentovaný číselnou hodnotou a jeho výhoda spočíva v tom, že táto číselná hodnota zostáva rovnaká ako po zmene metadát hudobného súboru, tak aj po ľubovoľnej zmene umiestnenia súboru v rámci daného disku. Hlavný zoznam pesničiek sa zobrazí na hlavnej obrazovke prehrávača po získaní všetkých dát zo všetkých súborov hudobnej knižnice tak, ako to je znázornené v kapitole 6.1.2.

[6] https://docs.microsoft.com/en-us/windows/uwp/packaging/app-capability-declarations

[7] https://docs.microsoft.com/en-us/windows/uwp/files/quickstart-managing-folders-in-the-music-pictures-and-videos-libraries

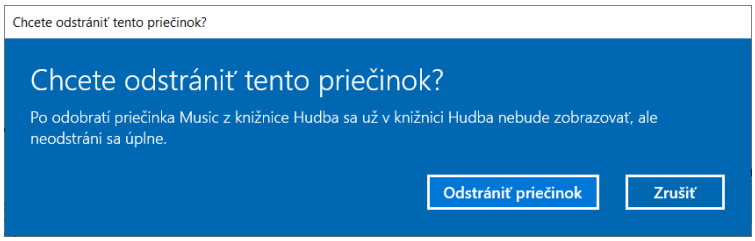
Užívateľ môže pridať priečinky do hudobnej knižnice stlačením tlačidla, ktoré sa nachádza v pravom dolnom rohu obrazovky nastavení (obr. 6.29) alebo stlačením ľavého tlačidla dialógového okna, ktoré sa užívateľovi zobrazí pri prvom spustení prehrávača, ak je hudobná knižnica prázdna (obr. 6.17). Užívateľovi sa po tomto úkone zobrazí systémové dialógové okno (obr. 6.30), ktoré mu umožní pridať vybraný priečinok do hudobnej knižnice alebo túto akciu zrušiť. O zobrazenie a skrytie tohto dialógového okna, ako aj o samotné pridanie zvoleného priečinku do hudobnej knižnice sa automaticky postará systém po zavolaní metódy `RequestAddFolderAsync()` objektu hudobnej knižnice, ktorý je možné získať výrazom: `StorageLibrary.GetLibraryAsync(KnownLibraryId.Music)`. Ak užívateľ pridal priečinok do hudobnej knižnice pomocou tlačidla z obrazovky nastavení, tak sa tento nový priečinok hneď zobrazí v zozname priečinkov hudobnej knižnice na danej obrazovke. Po návrate na hlavnú obrazovku sa v tomto prípade zobrazí obrazovka načítavania (obr. 6.30), ktorá užívateľovi oznamuje, že boli vykonané zmeny v hudobnej knižnici a že prehrávač tieto zmeny práve aplikuje. Prehrávač znovu získava pesničky z hudobnej knižnice počas toho, ako je táto obrazovka načítavania užívateľovi zobrazená a to z toho dôvodu, aby bol obsah hlavného zoznamu pesničiek na hlavnej obrazovke vždy aktuálny. Ak pred naplnením hudobnej knižnice z nastavení nebolo možné prehrať žiadnu pesničku z toho dôvodu, že hlavný zoznam pesničiek bol prázdny, tak po návrate na hlavnú obrazovku prehrávač automaticky vyberie pesničku na prehranie podľa algoritmu výberu pesničky (kapitola 5.2).



Obr. 6.30: Obrázok vľavo zobrazuje dialógové okno pre pridanie priečinku do hudobnej knižnice. Obrázok vpravo zobrazuje obrazovku načítavania, ktorá sa zobrazí po návrate z nastavení, ak boli v nastaveniach vykonané zmeny v priečinkoch hudobnej knižnice.

Aplikácia umožňuje pre kompletnú správu hudobnej knižnice priečinky z nej aj odstraňovať. Túto akciu môže užívateľ vykonať cez vyskakovacie menu na obrazovke nastavení (obr. 6.29), ktoré sa mu zobrazí, keď pravým tlačidlom myši klikne na priečinok v zozname, ktorý chce z knižnice odstrániť. Keď užívateľ zvolí túto akciu z vyskakovacieho menu, tak sa zavolá metóda objektu hudobnej knižnice `RequestRemoveFolderAsync()`, pričom jej parametrom je zvolený priečinok na odstránenie. Táto metóda zobrazí systémové dialógové okno (obr. 6.31), ktoré užívateľovi oznámi, že ak sa rozhodne zvolený priečinok odstrániť ľavým tlačidlom, tak sa tento priečinok odstráni len z hudobnej knižnice a nie z disku. Ak užívateľ túto akciu potvrdí, tak systém automaticky odstráni daný priečinok z hudobnej knižnice, čo sa okamžite prejaví aj v príslušnom zozname na obrazovke nastavení. Po návrate na hlavnú obrazovku sa zobrazí totožná obrazovka načítavania ako po pridaní nového priečinku do knižnice (obr. 6.30). Pesničky, ktoré sa viac v hudobnej knižnici nenachádzajú z dôvodu od-

stránenia ich priečinku z hudobnej knižnice budú po dokončení načítavania odstránené ako z hlavného zoznamu pesničiek, tak aj zo všetkých ostatných zoznamov aplikácie. Ak užívateľ všetko z hudobnej knižnice odstráni, tak po návrate z nastavení bude hlavná obrazovka prázdna až na časť s práve prehrávanou pesničkou, ktorá zostane v prehrávači načítaná do vypnutia aplikácie a je možné ju prehrávať v takomto prípade aj opakovane. Pri ďalšom spustení prehrávača už bude hlavná obrazovka kompletne prázdna tak, ako to je možné vidieť na obrázku 6.30 vpravo (bez obrazovky načítavania).



Obr. 6.31: Obrázok zobrazuje dialógové okno vygenerované systémom, ktoré užívateľovi oznamuje, že pri odstránení priečinku z hudobnej knižnice nedochádza k jeho odstráneniu z disku. Zároveň sa ho pýta, či chce zvolený priečinok skutočne z hudobnej knižnice odstrániť.

Obidve verzie hudobného prehrávača dokážu súbory pesničiek odstrániť. Túto možnosť je možné využiť hlavne vtedy, keď užívateľ nechce, aby súbory neobľúbených pesničiek zaberali miesto na úložisku jeho zariadenia. Vymazanie súboru pesničky je dostupné v mobilnej verzii prehrávača pomocou vyskakovacieho menu hlavného zoznamu pesničiek a ovládacieho panelu na hlavnej obrazovke (obr. 6.8 a 6.10), ako aj cez vyskakovacie menu zoznamu na obrazovke blokovaných pesničiek (obr. 6.15). Po potvrdení dialógového okna pre vymazanie pesničky (obr. 6.15) sa spustí proces pre odstránenie súboru zo zariadenia. Hudobný súbor je možné zo zariadenia odstrániť dvoma spôsobmi – cez `ContentResolver` a využitím objektu triedy `File`, pričom najlepším riešením je kombinácia obidvoch spôsobov. Pri vymazaní súboru cez `ContentResolver` je najskôr nutné zvolený súbor získať metódou `query()`. Keďže sa jedná o prácu s audio súbormi, tak URI poskytovateľa obsahu je rovnaké ako pri získavaní súborov. Zmena nastáva od parametru `selection`. Ten sa nastaví na hodnotu `MediaStore.Audio.Media.DATA + "=? "`, ktorou sa dá poskytovateľovi obsahu vedieť, že aplikácia chce získať konkrétny hudobný súbor na základe jeho umiestnenia. Samotné umiestnenie zvoleného súboru sa nastaví pomocou parametru `selectionArgs`. Pre vymazanie súboru cez `ContentResolver` nie je potrebné získať celý súbor, ale stačí získať jeho identifikátor. Toto je umožnené nastavením parametru `projection` na hodnotu `MediaStore.Audio.Media._ID`. Keď sa zvolený súbor nájde, tak sa k nemu pristúpi pomocou objektu triedy `Cursor` a získa sa jeho identifikátor. URI pre zmazanie daného súboru sa vytvorí metódou `ContentUris.withAppendedId()` s využitím získaného identifikátoru a URI poskytovateľa obsahu, ktorý má na starosti audio súbory. Samotné zmazanie prebehne pomocou metódy `delete()` objektu `ContentResolver`, pričom URI pre zmazanie súboru sa nastaví ako jej parameter. Pred týmto zmazaním sa ešte vytvorí objekt triedy `File`, ktorého inicializačným parametrom je umiestnenie zvoleného súboru. Po dokončení odstránenia cez `ContentResolver` sa využitím metódy `exists()` objektu `File` skontroluje, či sa súbor fyzicky stále nachádza v úložisku chytrého telefónu. Ak táto metóda vráti kladnú pravdivostnú hodnotu, tak prebehne zmazanie súboru cez objekt `File` zavolaním jeho metódy `delete()`. Ak dôjde pri niektorom pokuse o vymazanie k chybe, tak je o nej užívateľ

informovaný cez objekt triedy `Toast`. Po úspešnom zmazaní pesničky je jej položka odstránená z hlavného zoznamu pesničiek a aj zo všetkých ostatných zoznamov aplikácie, vrátane výsledkov vyhľadávania. Ak bola zmazaná práve prehrávaná pesnička, tak sa automaticky spustí prehrávanie ďalšej pesničky podľa algoritmu výberu pesničky (kapitola 5.2).

Odstránenie pesničky z počítača je užívateľovi umožnené cez vyskakovacie menu hlavného zoznamu pesničiek na hlavnej obrazovke (obr. 6.23) a cez vyskakovacie menu zoznamu zablokovaných pesničiek na obrazovke nastavení (obr. 6.29). V obidvoch prípadoch sa zobrazí totožné dialógové okno (obr. 6.24), ktoré užívateľa žiada o potvrdenie zmazania. Keď užívateľ zmazanie potvrdí, tak prehrávač získa odpovedajúci súbor pesničky ako objekt triedy `StorageFile` zo zoznamu súborov, ktorý aplikácia získala pri načítavaní pesničiek z hudobnej knižnice. Samotné zmazanie hudobného súboru je vykonané volaním metódy `DeleteAsync()` získaného objektu triedy `StorageFile`. Po vymazaní pesničky je jej položka odstránená zo všetkých zoznamov prehrávača, čo sa prejaví aj v užívateľskom rozhraní. V prípade zmazania práve prehrávanej pesničky sa automaticky začne prehrávať ďalšia pesnička tak isto ako v mobilnej verzii.

6.3 Prehrávanie hudby

Najdôležitejším komponentom pre fungovanie mobilnej verzie hudobného prehrávača je služba. Jednou z najdôležitejších úloh služby je práve prehrávanie hudby a jeho riadenie. Služba sa pre prehrávanie hudby používa hlavne z toho dôvodu, že umožňuje prehrávať hudbu aj na pozadí, keď užívateľ s aplikáciou priamo nepracuje. Podporu pre prehrávanie hudby a riadenie tohto prehrávania poskytuje na platforme Android trieda `MediaPlayer`. Objekt tejto triedy je vytvorený vzápätí po vytvorení samotnej služby. Po vytvorení objektu triedy `MediaPlayer` sa zavolá jeho metóda `setWakeMode()`, ktorá zabezpečí to, aby systém nezrušil prehrávanie hudby z dôvodu šetrenia batérie, keď prehrávač prehráva hudbu na pozadí a keď súčasne užívateľ so svojím telefónom práve aktívne nepracuje. Aby vôbec bolo možné túto metódu použiť, tak musí byť v manifeste aplikácie deklarované povolenie s názvom `android.permission.WAKE_LOCK`. Okrem toho sa ešte `MediaPlayer` nastaví na prehrávanie hudby a taktiež sa mu nastavia metódy, ktoré sú vyvolané v prípade, keď:

- je dokončená príprava hudobného súboru na prehrávanie,
- je dokončené prehrávanie aktuálnej pesničky,
- vznikne nejaká chyba.

`MediaPlayer` sa celý čas od svojho vzniku až po svoj zánik nachádza v jednom z viacerých stavov (obr. 6.32). V každom stave je možné vykonať len určité operácie, čo má za následok to, že ak sa vykoná nejaká operácia v nesprávnom stave, tak systém môže vyvolať výnimku alebo spôsobiť iné neželané správanie. Toto je jeden z viacerých spôsobov, akým sa môže `MediaPlayer` dostať do chybového stavu. Pri prechode do tohto stavu sa automaticky vyvolá metóda `onError()`. Pre zotavenie z chyby a znovupoužitie objektu `MediaPlayer` je nutné zavolať jeho metódu `reset()`, pomocou ktorej sa `MediaPlayer` dostane späť do svojho počiatočného stavu. Po svojom vytvorení sa `MediaPlayer` taktiež nachádza v počiatočnom stave. V tomto stave sa získa objekt pesničky, ktorá sa má prehrať. Táto pesnička bola buď vybraná algoritmom výberu pesničky, alebo načítaná ako pesnička, ktorá sa prehrávala pri poslednom vypnutím prehrávača. Na základe identifikátoru tejto pesničky sa vytvorí URI, ktoré sa použije pre inicializáciu objektu triedy `MediaPlayer`.

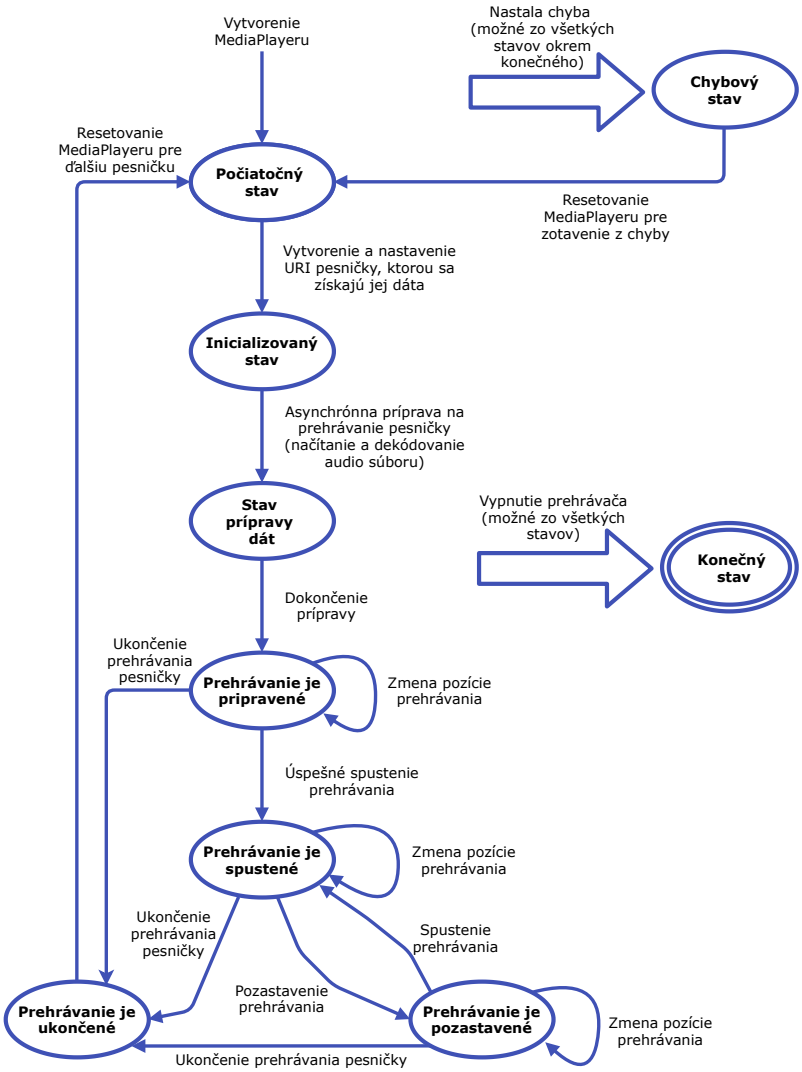
Samotná inicializácia prebieha volaním metódy `setDataSource()` tohto objektu, pričom vytvorené URI je jej parametrom. Vykonaním tejto metódy sa `MediaPlayer` dostáva do inicializovaného stavu. Pred spustením prehrávania pesničky musia byť jej dáta načítané a dekódované. Tento proces sa spúšťa volaním metódy `prepareAsync()`. Ako jej názov napovedá, tak táto metóda sa vykonáva asynchrónne a to z toho dôvodu, že načítanie a dekódovanie dát pesničky môže určitý čas trvať. Pri synchrónnom vykonávaní by tak počas tohto procesu mohlo dochádzať k nereagovaniu užívateľského rozhrania, čo spôsobuje zlú užívateľskú skúsenosť. O dokončení tejto prípravy je prehrávač informovaný vyvolaním metódy `onPrepared()`, čo pre `MediaPlayer` znamená, že sa nachádza v stave pripravenom na spustenie prehrávania. Ak bola táto metóda vyvolaná prvýkrát po spustení prehrávača, tak sa pozícia prehrávania nastaví metódou `seekTo()` na to miesto, kde prehrávanie pri predchádzajúcom vypnutí aplikácie skončilo. V ostatných prípadoch začne prehrávanie vždy od začiatku. Keď `MediaPlayer` dosiahne tento stav, tak sú súčasne aktualizované dáta o práve prehrávanej pesničke v ovládacom paneli hlavnej obrazovky, v notifikácii a prípadne aj vo widgete. Prehrávanie sa spustí zavolaním metódy `start()`. `MediaPlayer` môže v stave po spustení prehrávania meniť aktuálnu pozíciu prehrávania alebo prejsť do stavu, v ktorom je prehrávanie pozastavené zavolaním metódy `pause()`. V tomto stave sa môže tak isto meniť pozícia prehrávania alebo sa môže prehrávanie znova spustiť metódou `start()`, čím sa `MediaPlayer` vráti do predchádzajúceho stavu. Prehrávanie aktuálnej pesničky môže byť ukončené jedným z nasledujúcich spôsobov:

- prehrávanie aktuálnej pesničky bolo dokončené – pesnička sa prehrala celá, pričom sa vyvolá metóda `onCompletion()`,
- užívateľ zvolil prechod na nasledujúcu pesničku,
- užívateľ zvolil prechod na predchádzajúcu pesničku,
- užívateľ spustil prehrávanie inej pesničky kliknutím na jej položku v niektorom zozname pesničiek, ktorý túto akciu podporuje.

`MediaPlayer` je vo všetkých týchto prípadoch nutné zresetovať metódou `reset()`, čím sa `MediaPlayer` znova dostane do svojho počiatočného stavu, kde je možné nastaviť URI ďalšej pesničky, ktorá bola zvolená buď algoritmom výberu pesničiek, alebo samotným užívateľom. Pri vypnutí aplikácie sa `MediaPlayer` dostane zavolaním metódy `release()` do konečného stavu a zdroje, ktoré tento objekt využíval sú uvoľnené. Keďže vypnutie aplikácie môže nastať hocikedy, tak aj do tohto stavu sa môže `MediaPlayer` dostať zo všetkých ostatných. Okrem samotného riadenia prehrávania umožňuje `MediaPlayer` nastaviť hlasitosť prehrávania a poskytuje prístup k informáciám o aktuálnej pozícii prehrávania, celkovej dĺžke trvania práve prehrávanej pesničky alebo o tom, či sa hudba práve prehráva alebo nie [4, MediaPlayer[8], MediaPlayer overview[9]].

Prehrávanie hudby na platforme UWP funguje na podobnom princípe. Táto platforma rovnako používa na prehrávanie hudby a jeho ovládanie objekt triedy s názvom `MediaPlayer`, ktorý sa pri spustení hudobného prehrávača vytvorí a prepojí s prvkom `MediaPlayerElement` z užívateľského rozhrania, ktorý slúži ako ovládací panel (obr. 6.26). Prepojenie sa vykoná metódou `SetMediaPlayer()` tohto objektu a vytvorený `MediaPlayer` sa použije ako parameter. `MediaPlayer` sa potom nastaví atribútom `AudioCategory` na prehrávanie hudby,

[8] https://developer.android.com/reference/android/media/MediaPlayer
[9] https://developer.android.com/guide/topics/media/mediaplayer



Obr. 6.32: Obrázok zobrazuje stavy, v ktorých sa môže objekt triedy `MediaPlayer` mobilnej verzie prehrávača nachádzať, pričom tento objekt zabezpečuje prehrávanie hudby a jeho ovládanie.

aby systém vedel, aký typ média aplikácia prehráva. Ďalej sa mu zaregistrujú metódy, ktoré budú vyvolané pri vzniku rôznych udalostí. Medzi tieto udalosti patrí:

- `MediaOpened` – udalosť, keď je pesnička načítaná a pripravená na prehranie,
- `MediaEnded` – udalosť dokončenia prehrávania aktuálnej pesničky,
- `CommandManager.NextReceived` – udalosť prechodu na nasledujúcu pesničku,
- `CommandManager.PreviousReceived` – udalosť prechodu na predchádzajúcu pesničku,
- `VolumeChanged` – udalosť zmeny hlasitosti prehrávania z ovládacieho panelu.

Po dokončení načítavania súborov z hudobnej knižnice sa získa pesnička, ktorá sa prehrávala pri ukončení predchádzajúceho behu aplikácie. Spolu s touto pesničkou sa načíta aj jej

predchádzajúca pozícia prehrávania, ktorá sa pre `MediaPlayer` rovno nastaví využitím jeho atribútu `PlaybackSession.Position`. Ak sa jedná o prvé spustenie prehrávača po inštalácii, ak sa daná pesnička viac v počítači nenachádza alebo ak došlo ku nejakej inej chybe, tak sa vyberie nová pesnička pomocou algoritmu výberu pesničky a jej pozícia prehrávania sa nastaví na začiatok. Následne sa získa hudobný súbor zvolenej pesničky ako objekt triedy `StorageFile` zo zoznamu súborov hudobnej knižnice. Objekt `StorageFile` je potom možné previesť na objekt triedy `MediaSource` metódou `CreateFromStorageFile()` tejto triedy, ktorej parametrom je práve získaný `StorageFile`. Takto vytvoreným objektom triedy `MediaSource` je možné inicializovať objekt triedy `MediaPlaybackItem` pri jeho vytváraní. Súbor pesničky musí byť prevedený na tento dátový typ, pretože objekty triedy `MediaPlaybackItem` sa môžu nastaviť ako zdroj dát pre `MediaPlayer`, čím sa začne príprava danej pesničky na prehrávanie. Zdrojom dát pre `MediaPlayer` síce môže byť aj `MediaSource`, no `MediaPlaybackItem` je nevyhnutý pre správne zobrazenie informácií o práve prehrávanej pesničke v systéme, čomu sa bližšie venuje kapitola 6.6.1. Keď sa príprava dokončí, tak sa vyvolá udalosť `MediaOpened` a prehrávanie môže byť spustené. Metóda, ktorá má na starosti spracovanie tejto udalosti zabezpečí aktualizáciu informácií o práve prehrávanej pesničke na hlavnej obrazovke prehrávača. Po spustení prehrávania môže byť prehrávanie pozastavené a následne znova spustené. Tak isto je možné meniť aj pozíciu prehrávania. Prehrávanie aktuálnej pesničky môže byť ukončené celým prehraním aktuálnej pesničky, prechodom na nasledujúcu alebo prechádzajúcu pesničku a taktiež dvojklikom na položku určitých zoznamov pesničiek. Ak nebola nová pesnička vybraná dvojklikom, tak sa získa algoritmom výberu pesničky. Prehrávač získa hudobný súbor zvolenej pesničky ako objekt triedy `StorageFile` a celý proces sa opakuje. `MediaPlayer` sa taktiež používa pre zistenie aktuálnej pozície prehrávania a celkovej dĺžky trvania práve prehrávanej pesničky. Tieto informácie sa využívajú pre správne fungovanie prehrávača. Pri vypnutí aplikácie sa zavolá metóda `Dispose()` objektu triedy `MediaPlayer`, ktorou sa uvoľnia zdroje využívané týmto objektom [2, Play audio and video with MediaPlayer[10], Media items, playlists, and tracks[11]].

Aplikácia na platforme UWP nemá predvolene povolené prehrávať hudbu na pozadí – čiže vždy, keď užívateľ aplikáciu minimalizuje, keď sa vráti na domovskú obrazovku alebo keď opustí aplikáciu iným spôsobom. Pre umožnenie prehrávania hudby na pozadí je nutné toto oprávnenie zadefinovať v manifeste aplikácie cez užívateľské rozhranie vývojového prostredia Visual Studio alebo priamo v XML súbore manifestu pridaním nasledujúceho riadku: `<Capabilities> <uap3:Capability Name="backgroundMediaPlayback"/> </Capabilities>`. Bez tohto oprávnenia bude síce prehrávač pokračovať v prehrávaní hudby naďalej aj na pozadí, no systém v takomto prípade automaticky stlmí hlasitosť prehrávania. Pomocou udalostí životného cyklu aplikácie (`EnteredBackground` a `LeavingBackground`) dokáže aplikácia zistiť, či sa práve nachádza na popredí alebo na pozadí. Spracovanie obidvoch týchto udalostí sa vykonáva v automaticky vygenerovanom súbore aplikácie (`App.xaml.cs`), keďže sa jedná o udalosti životného cyklu aplikácie [2, App capability declarations[6], Play media in the background[12]].

[10] https://docs.microsoft.com/en-us/windows/uwp/audio-video-camera/play-audio-and-video-with-mediaplayer

[11] https://docs.microsoft.com/en-us/windows/uwp/audio-video-camera/media-playback-with-mediasource

[12] https://docs.microsoft.com/en-us/windows/uwp/audio-video-camera/background-audio

6.4 Správa dát

Aplikácia vytvára ako v mobilnej verzii, tak aj v počítačovej dva rozdielne typy dát. Prvým týmto typom sú hodnotenia pesničiek, ktoré môžu byť každej pesničke udelené v prípadoch popísaných tabuľkou 5.1. Ukladaniu nových hodnotení je možné zabrániť pomocou prepínača z obrazovky nastavení. Výnimkou je uloženie explicitného hodnotenia od užívateľa (typ „UX“) a hodnotenia pre zablokovanie pesničky (typ „B“), ktoré je možné udeliť vždy. Hodnotenie pre zablokovanie pesničky je taktiež špecifické tým, že ho je možné po udelení odstrániť. Druhým typom sú dáta, ktoré aplikácia potrebuje pre zabezpečenie korektného fungovania a ktoré je možné označiť ako nastavenia aplikácie. Nastavenia aplikácie obsahujú v obidvoch verziách prehrávača nasledujúce informácie:

- informácia určujúca to, či sa jedná o prvé spustenie aplikácie,
- hodnota priemerného skóre pesničiek,
- aktuálne usporiadanie hlavného zoznamu pesničiek,
- dátum vykonania posledného exportu skóre,
- aktuálny stav všetkých prepínačov a rozbaľovacieho zoznamu z nastavení,
- informácie o stave prehrávača pred jeho vypnutím, ktoré sú potrebné pre nasledujúci beh aplikácie – identifikátor práve prehrávanej pesničky a jej aktuálna pozícia prehrávania, identifikátory pätnástich naposledy prehratých pesničiek zo zoznamu predchádzajúcich pesničiek a identifikátory prvých piatich pesničiek zo zoznamu nasledujúcich pesničiek.

Keďže ani jeden z týchto dvoch typov dát neobsahuje štruktúrované dáta, tak obidva typy je možné ukladať vo forme dvojíc kľúč-hodnota. Kľúčom pre hodnotenia pesničky je identifikátor jej hudobného súboru a hodnotou sú samotné hodnotenia udelené danej pesničke v podobe textového reťazca, pričom jednotlivé hodnotenia sú od seba oddelené čiarkou. Kľúčom pre nastavenia aplikácie je vždy názov daného nastavenia a hodnotou týchto nastavení je buď číslo, pravdivostná hodnota, alebo textový reťazec.

Pre ukladanie dvojíc kľúč-hodnota sa na platforme Android používajú zdieľané preferencie (`SharedPreferences`). Zdieľané preferencie umožňujú vytvoriť súbor, ktorý bude obsahovať dvojice takéhoto tvaru a zároveň poskytujú jednoduché metódy pre ich čítanie a zápis. Každý takýto súbor spravuje sám framework a môže byť súkromný alebo zdieľaný. Pri prvom spustení aplikácie sa pre každý typ dát vytvorí a získa samostatný súbor volaním metódy `getSharedPreferences()`, ktorej prvým parametrom je názov daného súboru („SongScore“ alebo „Settings“) a druhým parametrom sa súbor nastaví ako privátny hodnotou `Context.MODE_PRIVATE`. Vďaka tomu bude mať k týmto súborom prístup výlučne len samotný prehrávač. Volanie tejto metódy pri ďalšom spustení aplikácie umožní získanie týchto súborov ako objektov dátového typu `SharedPreferences`. Pre zápis do súboru zdieľaných preferencií je najskôr nutné vytvoriť objekt `SharedPreferences.Editor` volaním metódy `edit()` daného súboru. Tomuto objektu sa následne predá kľúč a hodnota na uloženie pomocou metód `putInt()`, `putString()` a podobne. Na záver je pre samotné uloženie zmien nutné zavolať metódu editoru `apply()` alebo `commit()`. Metóda `apply()` vykonáva zmeny v objekte `SharedPreferences`, ktorý je načítaný v pamäti okamžite, no zmeny sa na disk zapisujú asynchrónne. Metóda `commit()` zapisuje dáta na disk synchrónne. To môže spôsobiť nereagovanie užívateľského rozhrania, a preto sa táto metóda používa v prehrávači

len vtedy, keď je to nevyhnutné, napríklad pri ukladaní dát tesne pred vypnutím prehrávača. Pre čítanie dát sa nad objektom súboru zdieľaných preferencií zavolá metóda ako `getInt()`, `getString()` alebo podobne. Metódy pre čítanie dát majú dva parametre. Prvým je kľúč hodnoty, ktorá sa má získať a druhým je predvolená hodnota, ktorá sa vráti, ak súbor daný kľúč neobsahuje [4, Save key-value data[13]].

Dvojice kľúč-hodnota ukladá aplikácia na počítači ako lokálne aplikačné dáta (predstavené v kapitole 3.2.1). Aplikácia musí najskôr získať úložisko lokálnych aplikačných dát predtým, ako môže čítať alebo zapisovať dáta. Toto úložisko je možné získať výrazom `ApplicationData.Current.LocalSettings`. Pomocou získaného úložiska sa následne pre každý typ dát prehrávača vytvorí a získa vlastný kontajner metódou `CreateContainer()`. Prvým parametrom tejto metódy je názov kontajneru („scoreContainer“ alebo „settingsContainer“) a druhým parametrom je hodnota `ApplicationDataCreateDisposition.Always`, ktorá zabezpečí to, že daná metóda vždy vráti objekt triedy `ApplicationDataContainer`. Ak kontajner s daným názvom neexistuje, tak sa kontajner vytvorí práve nastavením druhého parametru na túto hodnotu. Platforma UWP umožňuje využitím kontajnerov organizovať dáta aplikácie a samotný kontajner umožňuje ich zápis a čítanie. Zápis dát sa vykonáva atribútom `Values` kontajneru a to nasledujúcim spôsobom: `kontajner.Values[kľúč] = hodnota`. Pre čítanie dát sa používa ten istý atribút nasledujúcim spôsobom: `string x = (string)kontajner.Values[kľúč]`. Ak sa v kontajneri daný kľúč nenachádza, tak kontajner vráti pri čítaní hodnotu `null` [2, Store and retrieve settings and other app data[14]].

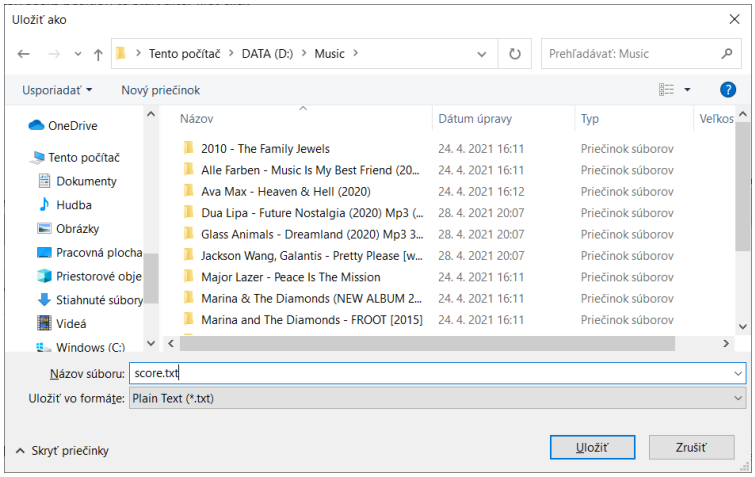
Súčasťou správy dát je aj import a export skóre. Export skóre znamená, že sa všetky hodnotenia všetkých pesničiek z hlavného zoznamu uložia do súboru na úložisku zariadenia. Užívateľ tak môže získať prístup k informáciám, ktoré sú ináč prístupné len aplikácii. Väčší význam má export skóre pre zálohovanie jednotlivých hodnotení a pre ich uchovanie aj po odinštalovaní aplikácie. Ďalším významným využitím exportu skóre je prenos hodnotení z jedného zariadenia do druhého, napríklad keď si užívateľ kúpi nové zariadenie. Užívateľ môže export vykonať stlačením tlačidla na obrazovke nastavení (obr. 6.13 a 6.28). Na chytrých telefónoch sa po stlačení tohto tlačidla vytvorí priečinok s názvom aplikácie na predvolenom úložisku a v ňom sa vytvorí textový súbor s názvom „score“ využitím triedy `File`. Ak už daný súbor existuje, tak bude jeho obsah prepísaný. Ak daný priečinok alebo súbor nie je možné vytvoriť, tak je užívateľ o tejto chybe informovaný cez objekt triedy `Toast` a proces exportu sa ukončí.

Keďže aplikácie vyvíjané na platforme UWP obvykle nemajú voľný prístup k úložisku zariadenia, tak sa po kliknutí tlačidla pre export zobrazí užívateľovi dialógové okno, ktoré ho vyzve, aby sám zvolil umiestnenie a názov súboru s hodnoteniami (obr. 6.33). Toto dialógové okno je implementované ako objekt triedy `FileSavePicker`. Atribútom `SuggestedStartLocation` tohto objektu sa nastaví predvolené umiestnenie súboru na pracovnú plochu a atribútom `SuggestedFileName` sa nastaví predvolený názov súboru na „score.txt“. Cez toto dialógové okno sa taktiež atribútom `FileTypeChoices` nastaví to, aby daný súbor mohol byť uložený len v textovom formáte. Ak užívateľ zruší toto dialógové okno tlačidlom v pravom dolnom rohu, tak sa export skóre ukončí. Po potvrdení ľavým tlačidlom získa aplikácia daný súbor ako `StorageFile` metódou dialógového okna `PickSaveFileAsync()`. Vykonaním metódy `CachedFileManager.DeferUpdates()`, kde získaný súbor bude jej parametrom, sa docieli to, že iné aplikácie nebudú môcť vyko-

[13] https://developer.android.com/training/data-storage/shared-preferences

[14] https://docs.microsoft.com/en-us/windows/uwp/design/app-settings/store-and-retrieve-app-data

návať zmeny v tomto súbore a to až dovtedy, kým sa export skóre nedokončí. Aplikácia ďalej vloží získaný súbor do špeciálneho systémového zoznamu aplikácie `MostRecentlyUsedList`, aby bolo v budúcnosti možné voľne pristupovať k tomuto súboru aj bez interakcie s užívateľom. Po vložení súboru do tohto zoznamu sa aplikácii vráti token, ktorý umožní voľný prístup k súboru a ktorý si aplikácia uloží do kontajneru nastavení. Aplikácia nestratí prístup k tomuto súboru ani keď sa zmení jeho umiestnenie v rámci disku a ani keď sa zmenia jeho metadáta [2, Save a file with a picker[15], Track recently used files and folders[16]].



Obr. 6.33: Obrázok zobrazuje dialógové okno, pomocou ktorého musí užívateľ počítačovej verzie prehrávača vybrať súbor, do ktorého budú uložené hodnotenia pesničiek pri exporte skóre.

Aplikácia začne prechádzať zoznamom pesničiek hneď ako získa prístup k súboru, do ktorého sa budú hodnotenia ukladať. Z každej pesničky sa získa názov jej súboru, samotný názov pesničky, názov jej interpreta a dĺžka jej trvania. Tieto informácie sa v uvedenom poradí spoja dokopy a táto nová hodnota bude slúžiť ako identifikátor pesničky pre export. Pôvodný číselný identifikátor pesničky v tomto prípade nie je možné použiť už len z toho dôvodu, že tento identifikátor môže mať pre danú pesničku na každom zariadení inú hodnotu. Následkom použitia pôvodného identifikátoru by bolo to, že hodnotenia by nebolo možné na inom zariadení správne naimportovať. Za tento nový identifikátor sa pridá reťazec „&&“, ktorý slúži ako oddeľovač identifikátoru pesničky od jej hodnotení. Následne sa na základe pôvodného číselného identifikátoru pesničky získajú jej hodnotenia zo zdieľaných preferencií alebo z kontajnera a pridajú sa za oddeľovač. Za hodnotenia sa ešte pridá znak nového riadku, čím vznikne riadok danej pesničky v súbore s hodnoteniami. Tieto riadky sa postupne spájajú dokopy a po prejdení celého zoznamu pesničiek vznikne obsah súboru s exportovanými hodnoteniami.

Zápis do súboru sa na platforme Android vykonáva objektom triedy `BufferedWriter`, ktorý sa inicializuje nasledujúcim spôsobom: `new BufferedWriter(new FileWriter(f))`, kde `f` je objekt súboru, do ktorého sa hodnotenia uložia. Samotný zápis sa vykoná metódou `write()`, ktorej parametrom je vytvorený obsah súboru. Pre správne zobrazenie priečinku aplikácie a súboru s exportovanými hodnoteniami v úložisku je nutné na platforme Android vykonať ešte nasledujúce kroky:

[15] https://docs.microsoft.com/en-us/windows/uwp/files/quickstart-save-a-file-with-a-picker
[16] https://docs.microsoft.com/en-us/windows/uwp/files/how-to-track-recently-used-files-and-folders

- vytvoriť URI pre daný súbor na základe jeho umiestnenia,
- vytvoriť nový objekt triedy `Intent`, ktorého prvým inicializačným parametrom bude hodnota `Intent.ACTION_MEDIA_SCANNER_SCAN_FILE` a druhým parametrom bude vytvorené URI,
- zavolať metódu `sendBroadcast()`, pričom ako jej parameter sa nastaví `Intent` vytvorený v predchádzajúcom kroku.

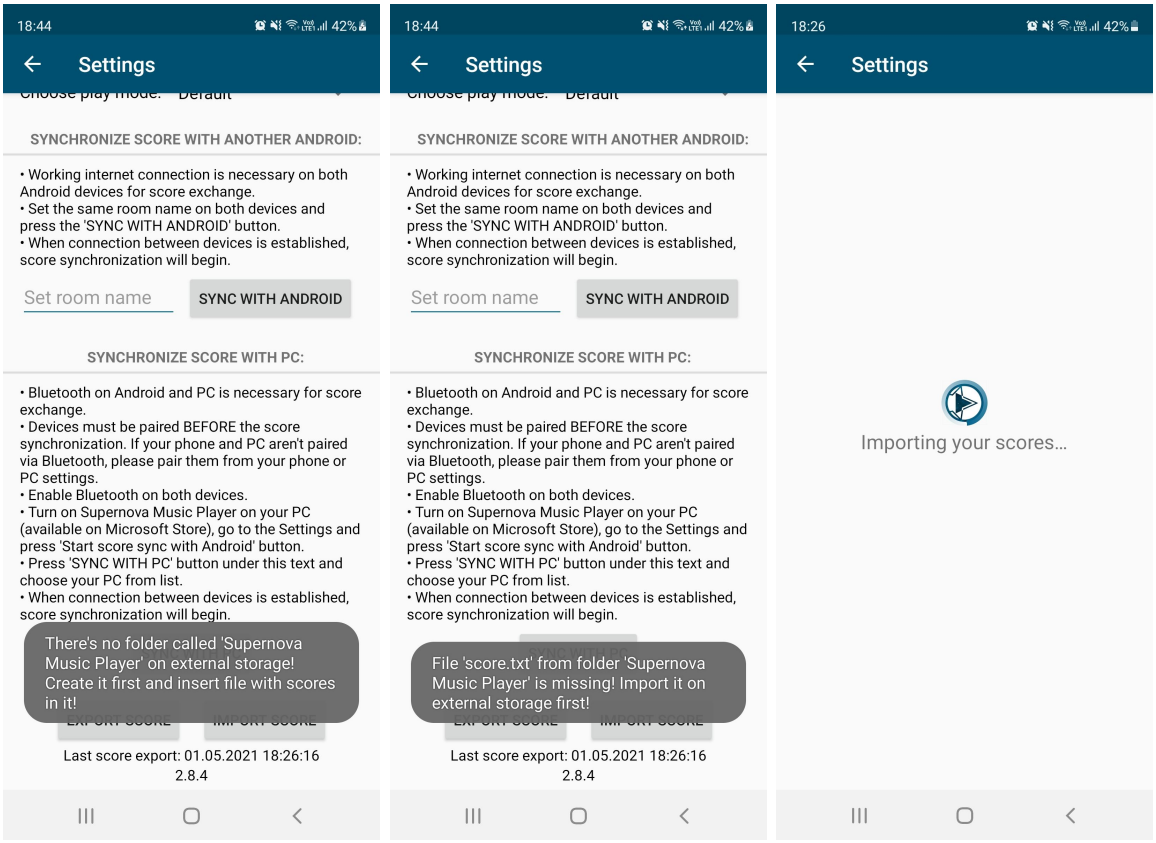
Následne sa už len uloží dátum a čas vykonania exportu do zdieľaných preferencií, ten istý údaj sa zobrazí v spodnej časti obrazovky nastavení a užívateľovi sa zobrazí objekt triedy `Toast`, ktorý mu oznámi umiestnenie súboru s exportovanými hodnoteniami.

Zápis exportovaných hodnotení do súboru sa na platforme UWP vykoná metódou `FileIO.WriteTextAsync()`, pričom jej prvým parametrom je objekt daného súboru a druhým parametrom je jeho budúci obsah. Po zápise do súboru sa dá systému vedieť, že ostatné aplikácie môžu znova vykonávať zmeny s týmto súborom a to zavolaním metódy `CachedFileManager.CompleteUpdatesAsync()`, ktorej parametrom je objekt súboru s hodnoteniami. Na záver sa dátum a čas vykonania exportu uloží do kontajneru nastavení a zároveň sa tento údaj zobrazí na obrazovke nastavení. Na obrazovke nastavení sa taktiež zobrazí aj umiestnenie tohto súboru, vrátane jeho názvu.

Aplikácia dokáže vykonávať export aj automaticky. V mobilnej verzii sa automatický export skóre vykonáva vždy pri spustení aplikácie, ak od posledného vykonania exportu alebo od prvého spustenia aplikácie ubehli štyri dni alebo viac. Jediným rozdielom od manuálneho exportu je to, že užívateľ nie je v takomto prípade informovaný o vzniknutej chybe ani o úspešnom dokončení exportu. Užívateľa by totiž mohlo zmiasť, že mu prehrávač zobrazuje hlásenia o exporte, keď ho sám nespustil. Počítačová verzia prehrávača môže vykonávať automatický export až po tom, ako export vykoná sám užívateľ. O tejto skutočnosti je užívateľ informovaný na obrazovke nastavení. Predtým totiž nemá aplikácia k dispozícii žiaden token, cez ktorý by mohla k nejakému súboru pristúpiť a vykonať export skóre sama. Počítačová aplikácia stratí schopnosť vykonávať automatický export z rovnakého dôvodu aj vtedy, keď užívateľ odstráni súbor s exportovanými hodnoteniami z disku. Zároveň tiež platí, že automatický export skóre sa vykoná pri spustení aplikácie len vtedy, keď od posledného vykonania exportu ubehli štyri dni alebo viac. Pri automatickom exporte sa najskôr získa inštancia zoznamu `MostRecentlyUsedList`. Súbor s hodnoteniami sa potom získa zavolaním metódy `GetFileAsync()` tohto zoznamu, pričom ako jej parameter sa použije token, ktorý sa predtým načíta z kontajneru nastavení. Zvyšok exportu prebehne rovnako ako u manuálneho exportu s tým, že obsah získaného súboru bude prepísaný.

Import skóre je komplementárnou operáciou exportu skóre. Úlohou importu je získať hodnotenia zo súboru a prideliť ich odpovedajúcim pesničkám v prehrávači. Import skóre je teda možné využiť pre načítanie hodnotení po každom exporte – či už po odinštalovaní a následnom nainštalovaní aplikácie alebo po prenose hodnotení z iného zariadenia. Import skóre môže užívateľ vykonať, rovnako ako export, pomocou tlačidla z obrazovky nastavení (obr. 6.13 a 6.28). Okrem toho je možné vykonať import aj pomocou dialógového okna (obr. 6.6 a 6.17), ktoré sa zobrazí pri prvom spustení aplikácie. V mobilnej verzii prehrávača sa toto dialógové okno zobrazí pri prvom spustení len ak sa na úložisku nachádza v čase spustenia priečinok s názvom aplikácie, ktorý obsahuje textový súbor s názvom „score“. Ak sa takto uložený súbor nenachádza v chytrom telefóne ani po spustení importu z nastavení, tak sa užívateľovi cez `Toast` zobrazí chybová hláška (obr. 6.34) a vykonávanie importu sa ukončí. Import z nastavení mobilnej verzie bude taktiež ukončený, ak užívateľ nepotvrdí

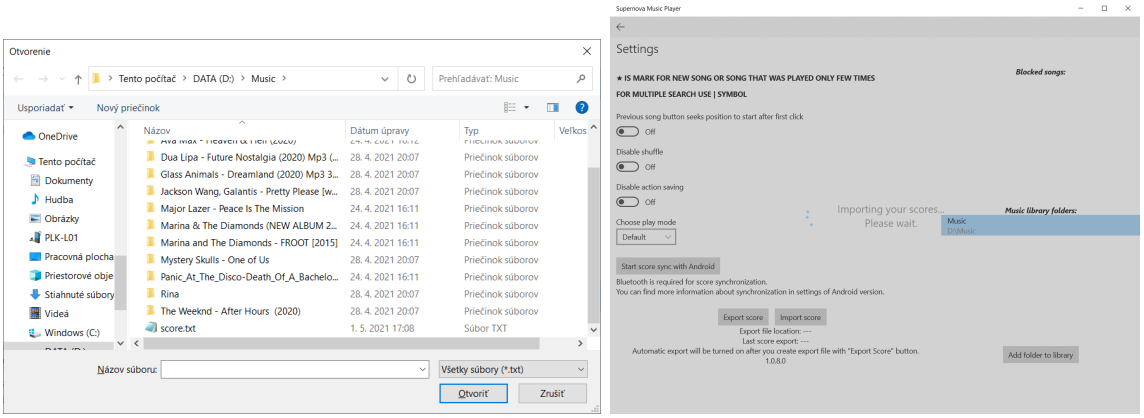
jeho vykonanie cez dialógové okno (obr. 6.13), ktoré sa zobrazí po stlačení príslušného tlačidla. V opačnom prípade aplikácia získa daný súbor na základe jeho umiestnenia ako objekt triedy `File` a užívateľovi sa zobrazí obrazovka načítavania (obr. 6.34).



Obr. 6.34: Obrázok vľavo a v strede zobrazuje výpis chybovej hlášky, ktorá sa užívateľovi zobrazí, keď sa pokúsi vykonať import skóre z nastavení mobilnej verzie prehrávača a keď sa súčasne na úložisku nenachádza priečinok s názvom aplikácie a v ňom textový súbor s názvom „score“. Na obrázku vpravo sa nachádza obrazovka načítavania, ktorá sa užívateľovi zobrazí na chytrom telefóne po úspešnom spustení importu skóre a počas ktorej prebieha načítavanie hodnotení pesničiek zo súboru.

Po potvrdení dialógového okna importu pri prvom spustení alebo po stlačení tlačidla z nastavení sa užívateľovi na počítači zobrazí dialógové okno (obr. 6.35), ktorým musí znova sám vybrať súbor s exportovanými hodnoteniami, aby k nemu mohla aplikácia získať prístup. Toto dialógové okno pre výber súboru je taktiež znova implementované ako `FileOpenPicker`, no zvolený súbor sa pre čítanie hodnotení získa metódou `PickSingleFileAsync()`. Po získaní súboru sa aj na počítači zobrazí užívateľovi obrazovka načítavania (obr. 6.35). Ak sa v prehrávači nenachádza veľké množstvo pesničiek a ak ani získaný súbor neobsahuje veľké množstvo dát, tak sa môže stať, že sa import vykoná tak rýchlo, že sa obrazovka načítavania ani nestihne zobraziť.

Po zobrazení obrazovky načítavania sa začne vykonávať samotný proces importu skóre zo súboru do aplikácie. Tento proces sa vykonáva asynchrónne, aby nedochádzalo k nereagovaniu užívateľského rozhrania. Na platforme Android sa pre zabezpečenie asynchrónneho vykonávania používa `AsyncTask`. Na platforme UWP stačí nastaviť kľúčové slovo `async` me-



Obr. 6.35: Ľavý obrázok zobrazuje dialógové okno, ktorým musí užívateľ počítačovej verzie prehrávača vybrať súbor obsahujúci exportované hodnotenia, aby mohol začať proces importu skóre. Vpravo sa nachádza obrazovka načítavania, ktorá sa zobrazí po úspešnom výbere takéhoto súboru a ktorou aplikácia oznamuje užívateľovi, že import skóre práve prebieha.

tóde, ktorá import vykonáva. Obsah súboru sa na platforme Android získa objektom triedy `BufferedReader`, ktorý sa inicializuje nasledujúcim spôsobom: `new BufferedReader(new FileReader(f.getAbsolutePath()))`, kde `f` je objekt získaného súboru. `BufferedReader` dokáže čítať súbor len po znakoch alebo po riadkoch, a preto je nutné pre získanie celého obsahu súboru celým týmto objektom prejsť cyklom `while` a získať každý riadok metódou `readLine()`. Počítačová verzia získa celý obsah súboru zavolaním metódy `FileIO.ReadTextAsync()`, pričom jej parametrom je objekt súboru s hodnoteniami.

Prehrávač sa snaží pre každú pesničku nájsť jej odpovedajúci riadok v súbore s hodnoteniami tým, že začne prechádzať zoznamom pesničiek hneď ako získa celý obsah daného súboru. Pre každú pesničku sa získa názov jej súboru, názov danej pesničky, názov jej interpreta a dĺžka jej trvania. Spojením týchto hodnôt sa vytvorí rovnaký identifikátor pesničky ako pri exporte. Za tento identifikátor sa pridá ešte oddeľovač („&&“). Takýto reťazec sa následne vyhľadá v získanom súbore. Ak sa tento reťazec v súbore nenájde, tak sa z reťazca odstráni ako oddeľovač, tak dĺžka trvania pesničky a vyhľadávanie sa spustí znova. Ten istý hudobný súbor totiž môže mať rôznu dĺžku trvania na rôznych zariadeniach, pričom tieto rozdiely bývajú typicky v niekoľkých milisekundách. Pri testovaní importu skóre som zistil, že pri prenose hudobného súboru z jedného zariadenia do druhého môže dokonca nastať zmena veľkosti písmen v názve pesničky a jej autora. Ak teda ani predchádzajúci pokus o vyhľadanie neuspeje, tak sa identifikátor bez dĺžky trvania pesničky prevedie na veľké písmená. Okrem toho sa vytvorí kópia obsahu súboru s hodnoteniami, ktorá sa taktiež prevedie na veľké písmená a vykoná sa posledný pokus o vyhľadanie identifikátoru v súbore. Ak ani tento posledný pokus nenájde žiadnu zhodu, tak sa v mobilnej verzii vytvorí špeciálny súbor v priečinku aplikácie. Do tohto súboru uloží prehrávač informáciu o tom, že sa pre danú pesničku nenašiel záznam v súbore s hodnoteniami alebo nastala iná chyba a import sa pre túto pesničku nevykonal. V počítačovej verzii sa tento súbor nevytvára, pretože užívateľ by musel znova vybrať ďalší súbor cez dialógové okno, do ktorého by sa tieto informácie uložili. Keď sa pesničku nepodarí v súbore s hodnoteniami vyhľadať, tak prehrávač automaticky prejde na ďalšiu pesničku zo zoznamu. V prípade, že sa niektorá verzia identifikátoru pesničky v súbore nájde, tak prehrávač získa celý jej riadok z tohto súboru.

Z riadku sa následne získajú hodnotenia, ktoré sa ďalej uložia buď do zdieľaných preferencií, alebo do kontajnera pomocou pôvodného číselného identifikátoru pesničky. Pôvodná hodnota v zdieľaných preferenciách alebo v kontajneri pri tom bude kompletne prepísaná hodnoteniami zo súboru. Pre zrýchlenie vykonávania importu prehrávač ešte odstráni spracovaný riadok z obsahu súboru, presunie sa na ďalšiu pesničku zo zoznamu a tento proces sa pre ňu opakuje. Po prejdení celého zoznamu pesničiek sa import skóre ukončí tým, že sa skryje obrazovka načítavania, aktualizujú sa hodnoty skóre v hlavnom zozname pesničiek a mobilná aplikácia dodatočne zobrazí `Toast`, ktorým užívateľa informuje, že import skóre prebehol úspešne. So správou dát aplikácie súvisí aj synchronizácia skóre, no tej sa venuje samostatná kapitola číslo 7.

6.5 Špecifické prvky platformy Android

6.5.1 Audio focus

`Audio focus` je koncept platformy Android, ktorým je vyjadrená skutočnosť, že užívateľ sa dokáže sústrediť len na jeden zdroj zvuku. Dve alebo aj viaceré aplikácie môžu súčasne prehrávať zvuk s tým, že systém zmixuje všetko dokopy. Takéto správanie ale spôsobuje zlú užívateľskú skúsenosť, ktorá môže užívateľov odradiť od používania aplikácie. Cez `audio focus` sa platforma Android snaží zaistiť to, aby vždy vydávala zvuk len jedna aplikácia. Keď chce aplikácia prehrávať zvuk, tak by mala požiadať o `audio focus` a keď ho získa, tak môže prehrávanie spustiť. To, že aplikácia `audio focus` získa avšak neznamená, že jej zostane už nastálo. Ostatné aplikácie si môžu vyžiadať `audio focus` pre seba a aplikácia, ktorá ho vlastní by ho mala ihneď uvoľniť a prehrávanie pozastaviť alebo prípadne stíšiť hlasitosť prehrávania. Platforma Android rozlišuje tri typy `audio focusu`:

- trvalý (`AUDIOFOCUS_GAIN`) – aplikácia plánuje prehrávať zvuk neurčite dlhý čas, pričom aplikácia, ktorá `audio focus` vlastnila svoje prehrávanie ukončí,
- dočasný (`AUDIOFOCUS_GAIN_TRANSIENT`) – aplikácia plánuje prehrávať zvuk len dočasne a aplikácia, ktorá `audio focus` vlastnila svoje prehrávanie počas toho pozastaví,
- krátky (`AUDIOFOCUS_GAIN_TRANSIENT_MAY_DUCK`) – aplikácia prehrá len krátky zvuk, pričom predchádzajúca aplikácia môže počas toho v prehrávaní pokračovať so zníženou hlasitosťou.

Prehrávač žiada o `audio focus` vždy tesne predtým, ako má byť spustené prehrávanie hudby a to metódou `requestAudioFocus()` objektu triedy `AudioManager`. Ten je možné získať metódou `getSystemService()` s parametrom `Context.AUDIO_SERVICE`. Metóda `requestAudioFocus()` má dve mierne odlišné implementácie, pričom konkrétna implementácia sa zvolí na základe verzie operačného systému zariadenia. V obidvoch implementáciách sa ale dáva systému vedieť, že aplikácia vyžaduje `audio focus` pre prehrávanie hudby a že pre svoje fungovanie potrebuje trvalý `audio focus`. Ak bola žiadosť o `audio focus` úspešná, tak táto metóda vráti hodnotu `AUDIOFOCUS_REQUEST_GRANTED`. Ak daná metóda nevráti túto hodnotu, tak prehrávanie hudby nebude spustené. Prehrávač sa dobrovoľne vzdá `audio focusu` vždy, keď užívateľ prehrávanie pozastaví. Pre vykonanie tejto akcie existujú taktiež dve rozdielne implementácie. Na zariadeniach s Androidom vo verzii 8.0 a vyššie sa pre odovzdanie `audio focusu` použije metóda `abandonAudioFocusRequest()` a na ostatných zariadeniach sa zavolá metóda `abandonAudioFocus()`.

Aby mohol prehrávač koncept `audio focusu` používať, tak musí implementovať rozhranie `AudioManager.OnAudioFocusChangeListener`. Toto rozhranie obsahuje jedinú metódu `onAudioFocusChange()`, ktorá sa vyvolá, keď prehrávač `audio focus` získa alebo stratí. Trvalú stratu `audio focusu` (`AUDIOFOCUS_LOSS`) najčastejšie spôsobuje spustenie iného hudobného prehrávača, spustenie prehrávania videa alebo podcastu. Pri tejto strate `audio focusu` sa prehrávanie hudby pozastaví a znova ho bude môcť spustiť len užívateľ cez užívateľské rozhranie aplikácie, notifikáciu alebo widget. Prehrávanie sa pozastaví aj pri dočasnej strate `audio focusu` (`AUDIOFOCUS_LOSS_TRANSIENT`). Tá môže byť spôsobená zvonením budíka, automatickým spustením krátkeho videa pri používaní inej aplikácie alebo telefónnym hovorom. Keď aplikácia, ktorá tento typ straty spôsobila, skončí s prehrávaním, tak prehrávač získa `audio focus` automaticky späť. Prehrávač si pri takejto strate musí uložiť nasledujúce informácie, aby dokázal určiť, či má po získaní `audio focusu` znova prehrávanie spustiť alebo nie:

- informácia o tom, či bolo pred dočasnou stratou prehrávanie spustené,
- informácia o tom, či boli k chytrému telefónu pred dočasnou stratou pripojené Bluetooth slúchadlá, ktorú je možné získať z objektu triedy `BluetoothHeadset`,
- informácia o tom, či boli k chytrému telefónu pred dočasnou stratou pripojené klasické drôtové slúchadlá, ktorú je možné získať pomocou prijímača broadcastu (kapitola 4.1.3) `ACTION_HEADSET_PLUG`.

Krátku stratu `audio focusu` (`AUDIOFOCUS_LOSS_TRANSIENT_CAN_DUCK`) typicky spôsobujú upozornenia iných aplikácií. Prehrávač bude v takejto situácii pokračovať v prehrávaní hudby, no hlasitosť prehrávania sa na krátky okamih zníži metódou `setVolume()` objektu triedy `MediaPlayer` tak, aby užívateľ dokázal zaregistrovať zvuk upozornenia [4, AudioFocusRequest[17], Managing audio focus[18]].

Pri získaní `audio focusu` (`AUDIOFOCUS_GAIN`) sa hlasitosť prehrávania nastaví na pôvodnú nezníženú hodnotu a prehrávanie sa znova spustí v prípade, ak platí, že:

- prehrávanie je pozastavené,
- prehrávač pred stratou `audio focusu` hudbu prehrával,
- počas straty `audio focusu` neboli odpojené slúchadlá.

Užívateľ pri počúvaní hudby cez slúchadlá očakáva, že po ich odpojení sa prehrávanie automaticky pozastaví namiesto toho, aby hudba začala hrať cez vstavané reproduktory chytrého telefónu. Presne túto úlohu vykonáva prijímač broadcastu `ACTION_AUDIO_BECOMING_NOISY`. Jeho nevýhodou je, že nie je pomocou neho možné zistiť, či sú slúchadlá pripojené v konkrétnom časovom okamžiku. Používanie len tohto prijímača broadcastu by mohlo viesť k tomu, že prehrávanie by po získaní `audio focusu` po dočasnej strate bolo pozastavené, aj keď by malo byť spustené a naopak. Aby bolo možné za každých okolností správne určiť, či sa má prehrávanie po získaní `audio focusu` spustiť, tak je znova nutné získať informácie o pripojení slúchadiel tak, ako to je popísané vyššie a porovnať tieto hodnoty pred stratou `audio focusu` a po jej opätovnom získaní [4, Handling changes in audio output[19]].

[17] https://developer.android.com/reference/android/media/AudioFocusRequest
[18] https://developer.android.com/guide/topics/media-apps/audio-focus
[19] https://developer.android.com/guide/topics/media-apps/volume-and-earphones

6.5.2 Notifikácia

Keďže aplikácia využíva na prehrávanie hudby službu spustenú na popredí (kapitola 4.1.2), tak po jej vytvorení sa musí okamžite vytvoriť notifikácia, ktorá sa následne zobrazí volaním metódy `startForeground()`. Notifikácia sa vo všeobecnosti používa na platforme Android pre zobrazenie informácii z aplikácie mimo jej užívateľského rozhrania. Užívateľ môže kliknutím na notifikáciu otvoriť aplikáciu alebo vykonávať pomocou nej ďalšie rôzne akcie. Predtým, ako sa notifikácia zobrazí, je nutné pre zariadenia s operačným systémom Android 8.0 a vyššie vytvoriť a nastaviť kanál notifikácie. Bez prideleného kanálu sa notifikácia na týchto zariadeniach ani nezobrazí. Keď aplikácia zobrazuje rôzne typy notifikácií, tak práve pomocou kanálov je možné rozdeliť ich do kategórií, pričom jeden kanál reprezentuje jednu kategóriu. Užívateľ má takýmto spôsobom možnosť zabrániť zobrazovaniu jednotlivých kanálov zo systémových nastavení namiesto zabránenia zobrazovaniu všetkých notifikácií naraz. Keď aplikácia beží na zariadení s operačným systémom Android 7.1 a nižšie, tak sa správa tak, ako keby mala len jediný kanál a všetky jej notifikácie môžu byť buď povolené, alebo zakázané. Kanál notifikácie je implementovaný ako objekt triedy `NotificationChannel`, ktorému sa pri vytváraní pridelí identifikátor, názov a jeho dôležitosť. Dôležitosť kanálu určuje to, akým vizuálnym a zvukovým spôsobom sa dá užívateľovi vedieť, že notifikácia daného kanálu práve vznikla. Dôležitosti kanálu odpovedá na zariadeniach s Androidom 7.1 a nižšie priorita notifikácie. Keďže samotné vytvorenie notifikácie hudobného prehrávača väčšinou nebýva sprevádzané žiadnym zvukovým efektom, tak sa dôležitosť kanálu pre notifikáciu v mojom hudobnom prehrávači nastaví na hodnotu `IMPORTANCE_LOW`. Kanálu je následne možné okrem týchto inicializačných parametrov nastaviť ďalšie vlastnosti, akými je napríklad popis daného kanálu alebo zobrazenie na uzamknutej obrazovke. Vytvorený kanál je nutné zaregistrovať metódou `createNotificationChannel()` objektu triedy `NotificationManager`, kde objekt kanálu bude jediným parametrom tejto metódy. `NotificationManager` je možné získať zavolaním metódy `getSystemService()` s parametrom `Context.NOTIFICATION_SERVICE` [4, Notifications Overview[20]].

Notifikácia vytvorená hneď po spustení služby sa zobrazuje prázdna, keďže prehrávač ešte nemá potrebné informácie, ktorými by ju mohol naplniť. Predtým, ako služba získa tieto informácie sa musí vytvoriť a inicializovať objekt triedy `MediaSessionCompat`. Tento objekt umožňuje spracovanie akcií z notifikácie, widgetu a z diaľkových ovládaní medzi ktoré patria napríklad: drôtové aj bezdrôtové slúchadla, chytré hodinky alebo autá so systémom Android. Využitím tohto objektu dokáže aplikácia reagovať aj na hlasové príkazy, ktoré zariadeniu užívateľ udelí cez hlasového asistenta Google. Pomocou objektu triedy `MediaSessionCompat.Callback` sa definuje správanie pre jednotlivé akcie, pričom tento objekt dokáže spracovať nasledujúce akcie:

- spustenie a pozastavenie prehrávania,
- prechod na nasledujúcu alebo prechádzajúcu pesničku,
- zmena aktuálnej pozície prehrávania a
- ukončenie prehrávania hudby.

Po vytvorení sa tento objekt nastaví objektu `MediaSessionCompat` pomocou jeho metódy `setCallback()`. `MediaSessionCompat` okrem toho zabezpečuje zobrazenie aktuálnych in-

[20] https://developer.android.com/guide/topics/ui/notifiers/notifications

formácií o práve prehrávanej pesničke (metóda `setMetadata()`) ako aj informácií o aktuálnom stave prehrávania (metóda `setPlaybackState()`) v notifikácii a v zariadeniach s obrazovkou a možnosťou diaľkového ovládania prehrávania. Hudobný prehrávač vykonáva aktualizáciu týchto informácii pri každej zmene práve prehrávanej pesničky a pri každej zmene stavu prehrávania [4, Using a media session[21]].

Hneď ako prehrávač dokončí načítavanie všetkých potrebných dát sa zobrazí aktualizovaná verzia notifikácie (obr. 6.36), ktorá vo svojej nerozbalenej podobe obsahuje:

- ikonu a názov aplikácie,
- názov pesničky,
- názov interpreta,
- tri základné tlačidlá pre ovládanie prehrávania,
- obrázok pesničky.

Užívateľ môže pomocou špeciálneho tlačidla v tvare šípky alebo potiahnutím prstom smerom nadol zobraziť rozbalenú podobu notifikácie. Po rozbalení pribudne v notifikácii tlačidlo pre vypnutie prehrávača a v zariadeniach s novšou verziou operačného systému sa taktiež zobrazí posuvná lišta pre úpravu pozície prehrávania spolu s informáciami o aktuálnej pozícii prehrávania a o celkovej dĺžke trvania práve prehrávanej pesničky. Notifikácie sa vytvárajú objektom triedy `NotificationCompat.Builder`, ktorým sa nastaví kanál notifikácie, jej obsah, vzhľad a aj správanie. Po vykonaní metódy `build()` tohto objektu vznikne objekt triedy `Notification`, ktorý použije `NotificationManager` spolu s číselným identifikátorom notifikácie v metóde `notify()`. Vykonaním tejto metódy sa zobrazovaná notifikácia aktualizuje [4, Create a Notification[22]].

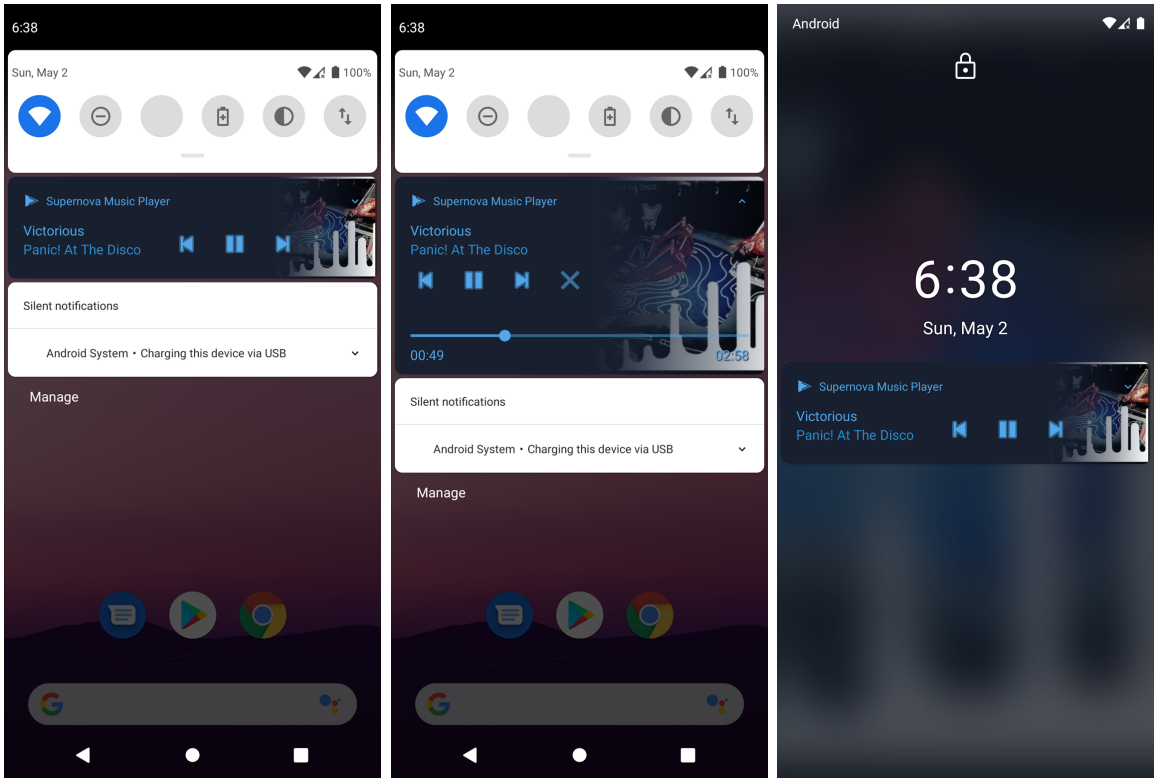
6.5.3 Widget

Widget (obr. 6.37) predstavuje na platforme Android akúsi miniaplikáciu, ktorú je možné vložiť na domovskú obrazovku zariadenia a ktorú je možné pravidelne aktualizovať. Základom vytvorenia widgetu je:

- konfiguračný XML súbor,
- XML súbor popisujúci štruktúru widgetu,
- trieda widgetu, ktorá rozširuje triedu `AppWidgetProvider`.

Koreňovým elementom konfiguračného súboru musí byť element `appwidget-provider`, ktorého atribútmi sa definujú základné vlastnosti widgetu aplikácie. Týmito vlastnosťami je možné nastaviť okrem iného: minimálnu výšku a šírku widgetu, umiestnenie súboru popisujúceho štruktúru widgetu, frekvenciu aktualizácie widgetu alebo umiestnenie obrázka, ktorý bude slúžiť ako ukážka pri výbere widgetu (obr. 6.37). XML súbor popisujúci štruktúru widgetu sa vytvára podobným spôsobom, akým sa vytvárajú ostatné obrazovky aplikácie. Rozdielom je, že štruktúra widgetu môže byť tvorená len určitými prvkami užívateľského rozhrania a súčasne sa musí brať do úvahy obmedzená výška a prípadne aj šírka widgetu. Na ľavom kraji widgetu môjho hudobného prehrávača sa nachádza obrázok pesničky, ktorý

[21] https://developer.android.com/guide/topics/media-apps/working-with-a-media-session
[22] https://developer.android.com/training/notify-user/build-notification



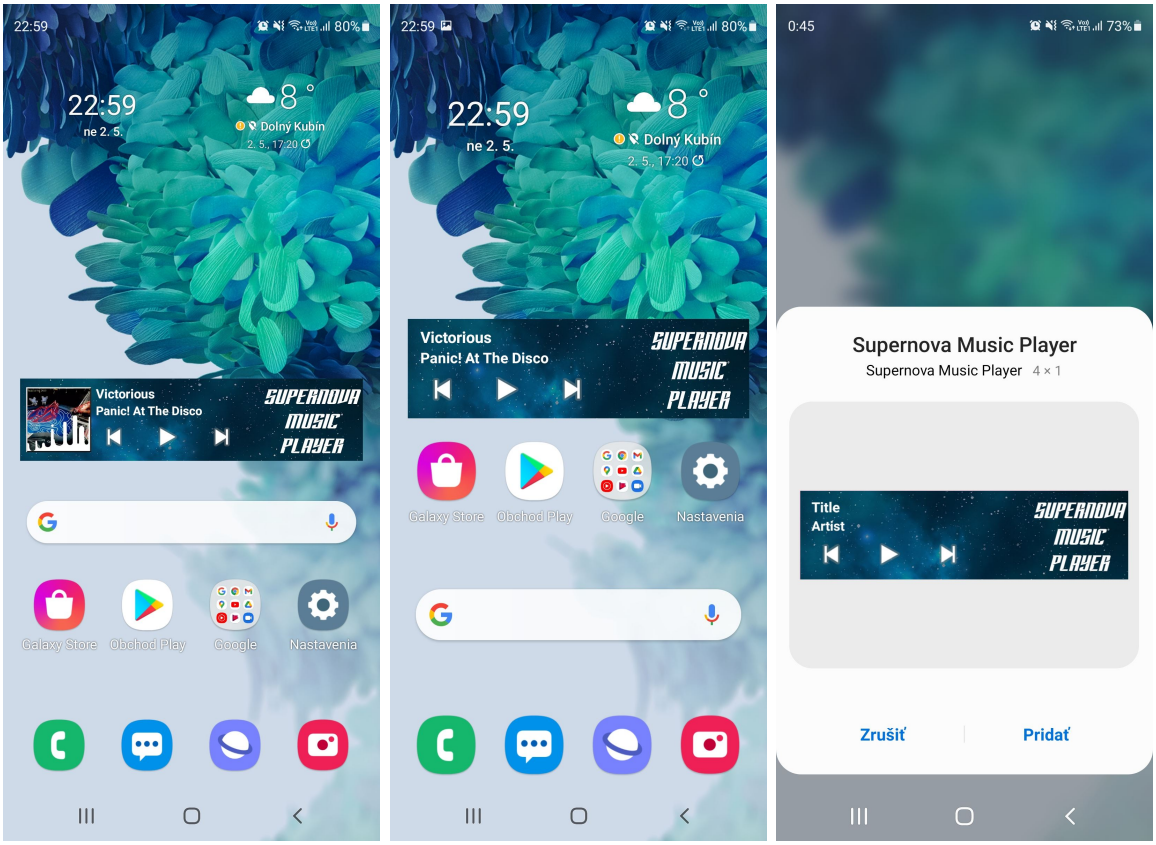
Obr. 6.36: Obrázky zobrazujú načítanú notifikáciu hudobného prehrávača na zariadení Pixel 3a s operačným systémom Android 10. Ľavý obrázok zobrazuje notifikáciu v jej nerozbalenej podobe. Obrázok v strede zobrazuje tú istú notifikáciu, ale v rozbalenej podobe a obrázok vpravo zobrazuje notifikáciu na uzamknutej obrazovke tohto zariadenia.

je predvolene skrytý a zobrazuje sa len vtedy, keď je obrazovka dostatočne veľká na jeho zobrazenie. Vo widgete sa vždy zobrazuje názov práve prehrávanej pesničky, jej interpret a tri základné tlačidlá pre ovládanie prehrávania. Keď užívateľ klikne na ľubovoľnú časť widgetu okrem týchto troch tlačidiel, tak sa aplikácia otvorí na hlavnej obrazovke. Widget má taktiež cez tento súbor nastavené špecifické pozadie s názvom hudobného prehrávača.

Poslednou základnou časťou je trieda widgetu. Po tom ako sa táto trieda vytvorí, tak je nutné ju zaregistrovať v manifeste aplikácie ako `receiver`, pričom sa mu tu cez jeho metadáta pridelí vytvorený konfiguračný súbor. Okrem toho mu je nutné nastaviť akciu `APPWIDGET_UPDATE`, ktorou systém dáva widgetu vedieť, že sa má jeho obsah aktualizovať. Najdôležitejšou časťou triedy widgetu je metóda `onUpdate()`, ktorá sa vyvolá, keď:

- ubehne interval definovaný atribútom `updatePeriodMillis` v konfiguračnom súbore,
- užívateľ pridá widget na domovskú obrazovku,
- je dokončené spúšťanie zariadenia.

Ako názov tejto metódy napovedá, tak jej úlohou je aktualizovať widget. Pre túto akciu potrebuje widget hudobného prehrávača získať informácie o práve prehrávanej pesničke, jej obrázok a aktuálny stav prehrávania pre správne zobrazenie ikony stredného tlačidla. Túto metódu však môže systém vyvolať aj keď nie je samotný prehrávač spustený a to po spustení



Obr. 6.37: Na obrázku vľavo je widget, ktorý zobrazuje základné informácie o práve prehrávanej pesničke a ktorý umožňuje ovládanie prehrávania hudby. Obrázok v strede zobrazuje ten istý widget, keď ale nie je možné zobraziť obrázok pesničky, pretože na obrazovke nie je dostatok voľného miesta. Obrázok vpravo zobrazuje ukážku widgetu, ktorú môže užívateľ vidieť predtým, ako vloží widget na domovskú obrazovku svojho zariadenia.

telefónu alebo keď užívateľ vloží widget na domovskú obrazovku. Widget teda získava tieto dáta dvoma spôsobmi:

1. ak je prehrávač spustený, tak widget získa informácie priamo od služby volaním jej statických metód a premenných,

2. ak prehrávač spustený nie je, tak widget získa zo zdieľaných preferencií identifikátor pesničky, ktorá sa prehrávala tesne pred posledným vypnutím prehrávača. Na základe tohto identifikátoru sa cez metódu `query()` objektu triedy `ContentResolver` vyhľadá daný hudobný súbor. Parameter `uri` bude mať znova hodnotu `MediaStore.Audio.Media.EXTERNAL_CONTENT_URI`. Parameter `selection` sa nastaví na hodnotu `MediaStore.Audio.Media._ID + "=?"`, ktorou sa dá poskytovateľovi obsahu vedieť, že widget chce získať konkrétny hudobný súbor na základe identifikátoru zo zdieľaných preferencií, ktorý sa uvedie ako parameter `selectionArgs`. Ak sa tento súbor nájde, tak widget z neho cez `Cursor` získa názov pesničky, názov interpreta a umiestnenie daného súboru. Objektom triedy `MediaMetadataRetriever` sa získa obrázok pesničky na základe umiestnenia jej súboru. Ak daná pesnička žiadny obrázok nemá, tak sa získa predvolený obrázok aplikácie. Stav prehrávania je v tomto prípade automaticky pozastavený.

V obidvoch prípadoch sa následne kombináciou objektov triedy `Intent` a `PendingIntent` nastaví to, aby widget reagoval ako na stlačenie tlačidiel pre ovládanie prehrávania, tak aj na kliknutie zvyšnej plochy widgetu pre otvorenie aplikácie. Pri stlačení tlačidiel na ovládanie prehrávania sa zvolená akcia odošle službe, kde ju príjme metóda `onStartCommand()` a následne ju spracuje objekt `MediaSessionCompat`. Ak služba pre prehrávanie hudby pri stlačení tlačidla nebeží, tak ju widget sám spustí. V metóde `onUpdate()` sa ešte skontroluje veľkosť widgetu a ak je dostatočne veľký, tak sa povolí zobrazenie obrázku pesničky. Samotnú aktualizáciu vykoná objekt triedy `AppWidgetManager`, ktorý je metóde predaný ako jej parameter. Okrem tejto metódy sa v triede widgetu môjho hudobného prehrávača nachádza metóda `onAppWidgetOptionsChanged()`. Túto metódu vyvolá systém vtedy, keď užívateľ vloží widget na domovskú obrazovku a vždy, keď užívateľ zmení veľkosť widgetu. Správanie tejto metódy je totožné so správaním metódy `onUpdate()`. Widget je aktualizovaný aj zo služby a to rovnako ako notifikácia – pri každej zmene práve prehrávanej pesničky a pri každej zmene stavu prehrávania. Aktualizácia sa zo služby vykonáva rovnakým spôsobom ako z widgetu s tým rozdielom, že služba vždy získa všetky potrebné informácie využitím svojich metód a premenných [4, Build an App Widget[23]].

Dôležitým poznatkom je, že pri každej aktualizácii sa musia vždy aktualizovať všetky časti widgetu, ináč sa môže stať to, že widget nebude zobrazovať správne informácie alebo nebude reagovať na kliknutie. V mojej implementácii som pri zmene práve prehrávanej pesničky aktualizoval len obrázok pesničky a textové informácie o práve prehrávanej pesničke. Rovnako sa pri spustení alebo pozastavení prehrávania aktualizovala len ikona stredného tlačidla. Reštartovanie telefónu a vypnutie alebo zapnutie systémového tmavého režimu spôsobilo pri takejto implementácii popísané neželané správanie. Aby widget znova správne fungoval, tak ho bolo nutné z domovskej obrazovky odstrániť a nanovo ho do nej vložiť. Táto skutočnosť navyše nie je nikde oficiálne zdokumentovaná a príčinu neželaného správania sa mi podarilo zistiť až po dlhom testovaní. Ďalšou podobnou nezdokumentovanou skutočnosťou je, že obrázok vo widgete môže mať len určitú veľkosť. Prekročenie tejto veľkosti spôsobilo pád hudobného prehrávača, ale len v prípade, ak sa widget nachádzal na domovskej obrazovke. Tento problém bol vyriešený tým, že všetky obrázky sa pred ich zobrazením zmenšujú metódou `Bitmap.createScaledBitmap()`.

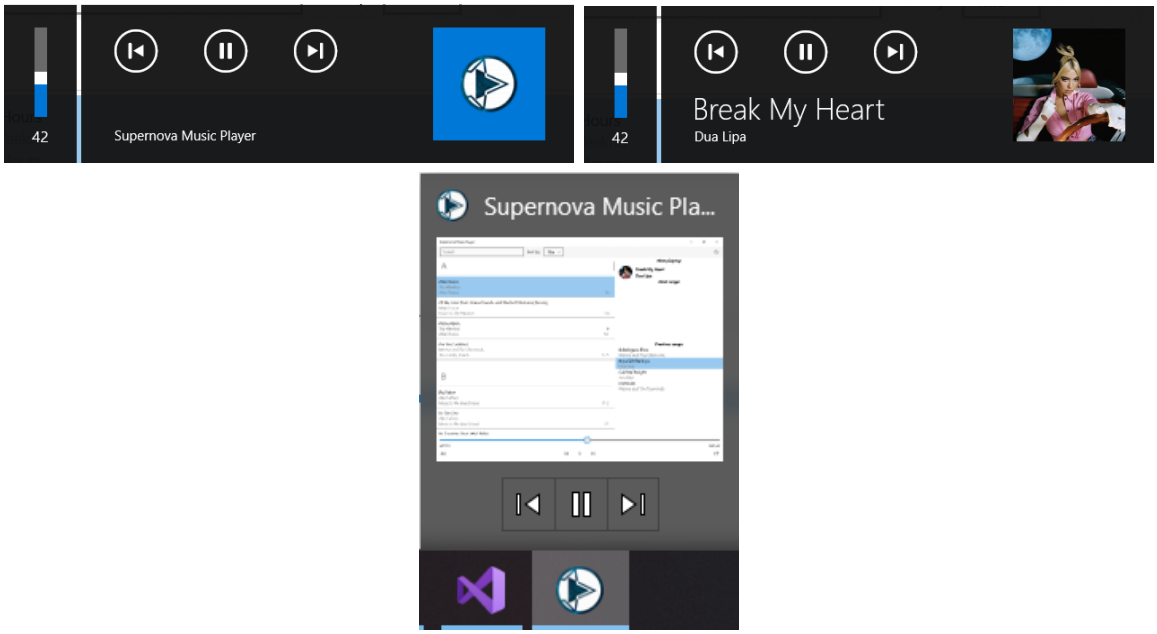
6.6 Špecifické prvky platformy UWP

6.6.1 System Media Transport Controls

System Media Transport Controls (SMTC) je sada ovládacích prvkov, ktoré umožňujú užívateľom ovládať prehrávanie médií konzistentným spôsobom. Ovládacie prvky SMTC umožňujú integráciu aplikácie so vstavaným systémovým užívateľským rozhraním, v ktorom je možné zobraziť metadáta ako názov pesničky alebo názov interpreta. Cez toto užívateľské rozhranie môže užívateľ pozastaviť a znova spustiť prehrávanie, ako aj prejsť na nasledujúcu alebo predchádzajúcu pesničku. Aplikácie, ktoré pre prehrávanie médií používajú objekt triedy `MediaPlayer` sú so systémom SMTC integrované automaticky. Aby bol tento systém funkčný, tak stačí v aplikácii vytvoriť `MediaPlayer` a nastaviť mu zdroj dát. V takomto prípade uvidí užívateľ v SMTC názov aplikácie, jej logo a 3 základné tlačidlá pre ovládanie prehrávania (obr. 6.38). Zobrazenie týchto informácií v SMTC je síce dostačujúce, no lepšiu užívateľskú skúsenosť je možné dosiahnuť nastavením a zobrazením informácií o práve prehrávanej pesničke. Keď prehrávač získa súbor pesničky ako objekt

[23] https://developer.android.com/guide/topics/appwidgets

triedy `MediaPlaybackItem`, tak sa z neho získa objekt `MediaItemDisplayProperties` metódou `GetDisplayProperties()`. Tento objekt sa naplní metadátami pesničky, ktorej prehrávanie sa za okamih spustí. To zahŕňa jej názov, názov jej interpreta a aj jej obrázok. Tieto zmeny sa aplikujú zavolaním metódy `ApplyDisplayProperties()` objektu triedy `MediaPlaybackItem`, ktorý sa vzápätí nastaví ako zdroj dát pre `MediaPlayer` [2, Integrate with the System Media Transport Controls[24]]. SMTC sa zobrazí užívateľovi, keď dôjde využitím klávesy „Fn“ k zmene systémovej hlasitosti, k zmene stavu prehrávania alebo k zmene práve prehrávanej pesničky. Tento systém taktiež umožňuje ovládať prehrávanie cez Bluetooth slúchadlá alebo z miniatúry aplikácie, ktorá sa zobrazí po prechode myšou na ikonu prehrávača na paneli úloh (obr. 6.38).



Obr. 6.38: Na horných obrázkoch sa nachádzajú ovládacie prvky SMTC, ktoré sa užívateľovi zobrazia, keď dôjde využitím klávesy „Fn“ k zmene systémovej hlasitosti, k zmene stavu prehrávania alebo k zmene práve prehrávanej pesničky. Na ľavom hornom obrázku je SMTC, ktoré by sa zobrazovalo, keby nedošlo k jeho naplneniu informáciami o práve prehrávanej pesničke. Na pravom hornom obrázku je SMTC, ktoré je takýmito informáciami naplnené. Na obrázku dole je zobrazená miniatúra aplikácie, ktorá sa zobrazí po prechode myšou na ikonu prehrávača na paneli úloh a pomocou ktorej je taktiež možné ovládať prehrávanie cez SMTC.

[24] https://docs.microsoft.com/en-us/windows/uwp/audio-video-camera/integrate-with-systemmediatransportcontrols

Kapitola 7

Synchronizácia dát medzi zariadeniami

Užívatelia môžu počúvať hudbu ako cez svoj chytrý telefón, tak cez počítač, čím na každom zariadení vzniká rozdielna sada hodnotení. Pre zjednotenie hodnotení môže síce užívateľ preniesť hodnotenia z jedného zariadenia do druhého využitím exportu a importu skóre, no nevýhodou je, že pri importe sa všetky uložené hodnotenia na druhom zariadení prepíšu hodnoteniami z prvého zariadenia a užívateľ musí navyše sám vykonať prenos súboru s hodnoteniami. Z týchto dôvodov je vhodné vykonávať import skóre len jednorazovo a to po inštalácii aplikácie. Synchronizácia skóre slúži na vyriešenie týchto problémov. Jej cieľom je získať dáta z iného zariadenia čo najmenej zaťažujúcim spôsobom pre užívateľa, nájsť rozdiely medzi získanými dátami a dátami z daného zariadenia a tieto rozdiely spojiť. Užívateľ tak môže používať aplikáciu na rôznych zariadeniach a následne synchronizáciou skóre dosiahnuť taký efekt, ako keby počúval hudbu stále len cez jedno zariadenie. Užívateľ môže synchronizáciu skóre spustiť pomocou ovládacích prvkov z nastavení (obr. 6.13 a 6.28).

7.1 Prenos dát medzi dvoma zariadeniami

Pre prenos dát medzi dvoma Android zariadeniami sa používa technológia WebRTC. Použitie tejto technológie bolo pôvodne zamýšľané aj na platforme UWP, keďže pre obidve platformy existujú knižnice, ktoré umožňujú použitie tejto technológie. Na platforme UWP sa mi však z neznámeho dôvodu nepodarilo vytvoriť spojenie medzi dvoma zariadeniami. Dôkladne som preštudoval dokumentáciu knižnice tejto technológie pre platformu UWP a na základe toho si nemyslím, že by moja implementácia obsahovala nejakú chybu, obzvlášť keď som sa pri implementácii okrem danej dokumentácie riadil aj už existujúcou a funkčnou implementáciou pre platformu Android. Keďže sa mi môj problém nepodarilo vyriešiť ani kontaktovaním samotných vývojárov danej knižnice, tak som bol nútený zvoliť pre komunikáciu s počítačovou aplikáciou inú technológiu. Musel som teda nájsť inú technológiu, ktorú podporujú obidve platformy, pričom som sa snažil zvoliť technológiu s podobnými vlastnosťami, aké má technológia WebRTC. Nakoniec som sa rozhodol pre prenos dát medzi mobilnou a počítačovou verziou prehrávača použiť technológiu Bluetooth.

7.1.1 Prenos dát medzi dvoma Android zariadeniami

Prenos dát medzi dvoma Android zariadeniami môže prebehnúť medzi dvoma chytrými telefónmi, ale aj medzi telefónom a tabletom alebo iným zariadením so systémom Android,

ktorý umožňuje prehrávanie hudby. Ako bolo spomenuté, tak tento prenos sa vykonáva využitím technológie WebRTC. WebRTC umožňuje webovú komunikáciu zariadení v reálnom čase. Táto technológia sa často používa pre prenos audiovizuálnych dát pre uskutočňovanie videohovorov, no je možné ju použiť aj na prenos ľubovoľných dát medzi dvoma zariadeniami spôsobom peer-to-peer. Aplikácia využívajúca túto technológiu musí vykonať nasledujúce činnosti:

- získať dáta,
- získať informácie o sieti (akými sú IP adresy a porty) a vymeniť ich s druhým zariadením pre umožnenie vytvorenia spojenia,
- koordinovať signalizačnú komunikáciu na hlásenie chýb a začatie alebo ukončenie spojenia,
- vymeniť si s druhým zariadením informácie o prenášaných dátach a o schopnostiach tohto druhého zariadenia,
- preniesť dáta.

WebRTC pozostáva z troch základných častí:

1. `MediaStream` – zabezpečuje získanie audiovizuálnych dát cez mikrofón a kameru zariadenia,
2. `RTCPeerConnection` – umožňuje vytvorenie spojenia medzi zariadeniami a prenos audiovizuálnych dát,
3. `RTCDataChannel` – umožňuje prenos ľubovoľných dát.

Pre zaistenie bezpečného prenosu dát používa implementácia technológie WebRTC bezpečnostné protokoly, akými je DTLS[1] a SRTP[2]. Okrem toho je šifrovanie povinné pre všetky komponenty WebRTC, vrátane signalizačného mechanizmu [9].

Predtým, ako samotný prenos dát vôbec začne, tak aplikácia na platforme Android zistí, či je dané zariadenie pripojené na internet. Aby aplikácia vôbec mala prístup na internet a aby mohla zisťovať stav pripojenia, tak je nutné definovať odpovedajúce povolenia v manifeste aplikácie. Stav pripojenia sa získa objektom triedy `ConnectivityManager`. Ten sa získa metódou `getSystemService()`, ktorej parametrom bude hodnota `Context.CONNECTIVITY_SERVICE`. Ak zariadenie nie je pripojené na internet, tak aplikácia vyzve cez `Toast` užívateľa, aby si zapol Wi-Fi alebo mobilné dáta. Ak zariadenie prístup na internet má, tak prebehne pripojenie ku signalizačnému serveru. Hoci WebRTC umožňuje peer-to-peer komunikáciu, tak sú stále potrebné servery, cez ktoré si zariadenia vymenia metadáta pre koordináciu tejto komunikácie. Tento proces výmeny sa nazýva signalizácia (obr. 7.1). Signalizáciou sa vymieňajú nasledujúce informácie:

- riadiace správy pre spustenie alebo ukončenie komunikácie,
- správy o chybách,
- metadáta prenášaných médií,

[1] https://en.wikipedia.org/wiki/Datagram_Transport_Layer_Security
[2] https://en.wikipedia.org/wiki/Secure_Real-time_Transport_Protocol

- dáta obsahujúce kľúče, ktoré umožňujú nadviazať bezpečné spojenie,
- nastavenia siete.

Signalizačný mechanizmus avšak nie je implementovaný technológiou WebRTC a vývojár si musí sám zvoliť, akým spôsobom ho implementuje. Spoločnosť Google poskytuje voľný prístup k zdrojovému kódu signalizačného serveru[3], ktorý pre komunikáciu využíva knižnicu Socket.IO na platforme Node.js. Keďže je táto knižnica dostupná aj pre platformu Android, tak som pre signalizáciu zvolil práve toto riešenie. Ďalšou výhodou tejto knižnice je koncept miestností [10]. Každá miestnosť má svoj názov, pomocou ktorého je možné identifikovať tie zariadenia, ktoré sa majú prepojiť. Názov tejto miestnosti môže nastaviť sám užívateľ cez textové pole, ktoré sa nachádza na obrazovke nastavení. Miestnosť na tomto signalizačnom serveri je nastavená tak, aby sa v nej mohli nachádzať najviac dve zariadenia [11]. Prehrávač si po pripojení na signalizačný server uloží informáciu o tom, či dané zariadenie miestnosť vytvorilo (ďalej bude toto zariadenie označované ako zariadenie A) alebo či sa len pripojilo do miestnosti, ktorú vytvorilo druhé zariadenie (ďalej označované ako zariadenie B). Ak užívateľ zvolí na svojich zariadeniach taký názov miestnosti, v ktorej sa už dve iné zariadenia nachádzajú, tak ho prehrávač o tomto fakte informuje cez `Toast` a súčasne ho vyzve nech zvolí iný názov miestnosti. Aby bolo možné signalizačný server vôbec použiť, tak je nutné ho niekde nasadiť. Pre tento účel som zvolil platformu Heroku. Heroku je platforma ako služba (PaaS), ktorá umožňuje vývojárom vytvárať, spúšťať a prevádzkovať aplikácie výlučne v cloude. Po dokončení nasadenia vygeneruje platforma Heroku URL adresu daného serveru. Signalizačnému serveru pre moju aplikáciu bola pridelená nasledujúca adresa: `https://blooming-dusk-15865.herokuapp.com/`. Tou sa v aplikácii inicializuje objekt triedy `Socket` knižnice Socket.IO, ktorý umožňuje komunikáciu s týmto serverom [12].

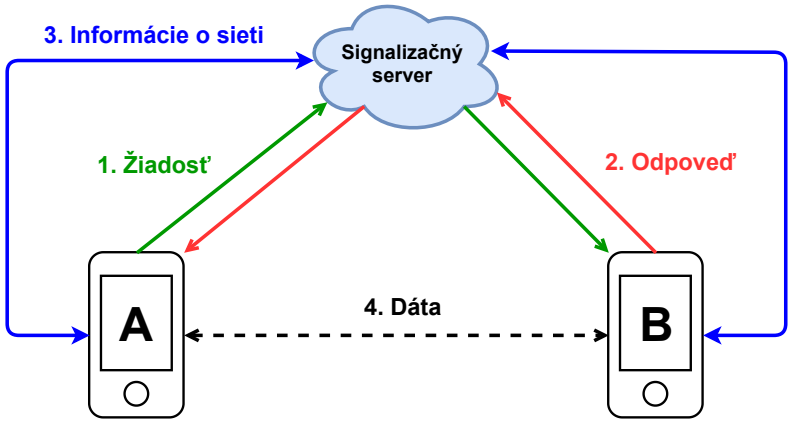
Po úspešnom pripojení ku signalizačnému serveru sa vytvorí a inicializuje objekt triedy `PeerConnectionFactory`. Týmto objektom je možné vytvoriť objekt triedy `PeerConnection`, ktorý odpovedá triede `RTCPeerConnection` v originálnej implementácii pre internetové prehliadače. Pre vytvorenie a inicializáciu tohto objektu sú nutné ďalšie tri rôzne objekty. Prvým z nich je konfiguračný objekt triedy `PeerConnection.RTCConfiguration`, ktorý je nutné inicializovať adresou aspoň jedného STUN serveru. Aplikácie využívajú STUN servery z jediného dôvodu – pre získanie verejnej IP adresy a čísla portu, keďže verejná adresa je nevyhnutná pre komunikáciu zariadení. Aplikácia hudobného prehrávača využíva verejný STUN server spoločnosti Google, ktorého adresa je: „stun:stun.l.google.com:19302“. Druhým inicializačným objektom je objekt triedy `MediaConstraints`, ktorým sa nastaví druh prenášaných médií, ako aj použitie protokolov DTLS a SRTP. Tretím a posledným inicializačným objektom je objekt triedy `PeerConnection.Observer`. Pomocou tohto objektu dokáže aplikácia reagovať na vznik rôznych udalostí, akou je napríklad zmena stavu pripojenia. Vytvorený objekt `PeerConnection` sa použije pre vytvorenie objektu triedy `DataChannel`, ktorý slúži na prenos ľubovoľných dát. Tomuto objektu sa ešte zaregistruje objekt triedy `DataChannel.Observer`, ktorým je možné reagovať na zmenu jeho stavu alebo na prichádzajúce dáta.

Keď sa dokončí vytváranie všetkých týchto objektov, tak aplikácia dá signalizačnému serveru vedieť, že je pripravená na komunikáciu s ďalším zariadením tým, že mu odošle správu s textom „got user media“. Keď túto správu odošle signalizačnému serveru aj druhé zariadenie, tak prebehne zvyšná časť procesu signalizácie (obr. 7.1):

[3] `https://codelabs.developers.google.com/codelabs/webrtc-web/#2`

- zariadenie A vytvorí cez `PeerConnection` žiadosť o spustenie komunikácie metódou `createOffer()`,
- obsah tejto žiadosti si zariadenie A uloží metódou `setLocalDescription()` a následne ju odošle zariadeniu B cez signalizačný server,
- zariadenie B si uloží obsah žiadosti metódou `setRemoteDescription()`,
- zariadenie B vytvorí odpoveď na túto žiadosť metódou `createAnswer()`, pričom obsah odpovedi si uloží cez `setLocalDescription()`,
- zariadenie B odošle odpoveď zariadeniu A cez signalizačný server,
- zariadenie A si uloží obsah odpovedi metódou `setRemoteDescription()`,
- obidve zariadenia si následne musia vymeniť informácie o sieti, ktoré sú reprezentované objektom triedy `IceCandidate`,
- keď zariadenie takýto objekt vytvorí, tak sa vyvolá metóda `onIceCandidate()` objektu `PeerConnection.Observer`, ktorou sa tento objekt odošle druhému zariadeniu cez signalizačný server,
- druhé zariadenie si tieto informácie uloží metódou `addIceCandidate()`.

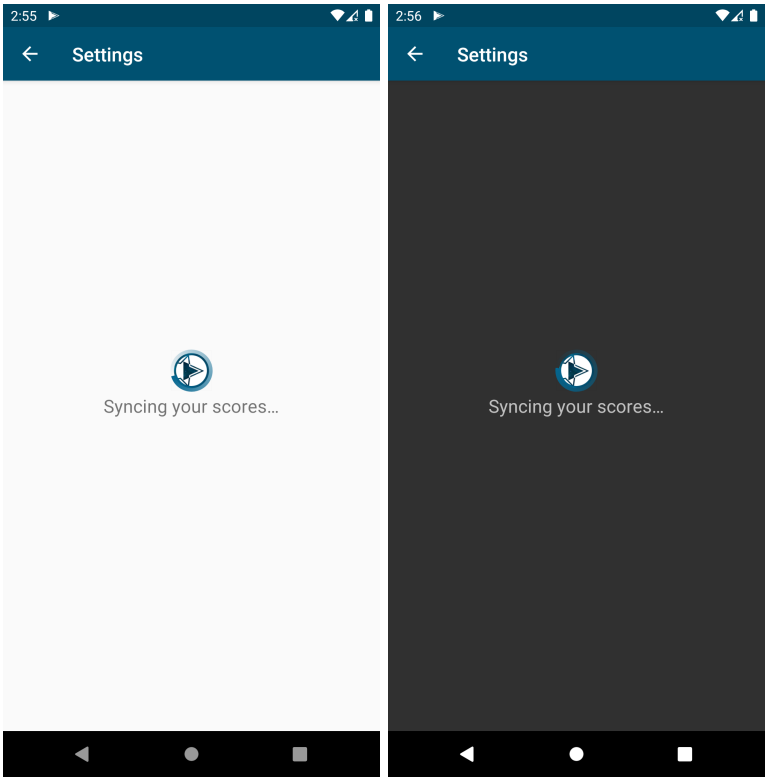
Po úspešnom dokončení procesu signalizácie je možné dáta prenášať priamo medzi zariadeniami [9, 10].



Obr. 7.1: Obrázok zobrazuje proces signalizácie, ktorého výsledkom je vytvorenie peer-to-peer spojenia cez technológiu WebRTC medzi dvoma Android zariadeniami. Zariadenie A sa k signalizačnému serveru pripojilo ako prvé, a tak vytvorí a pošle žiadosť o spustenie komunikácie. Zariadenie B túto žiadosť príjme a vytvorí na ňu odpoveď. Následne si zariadenia medzi sebou vymenia informácie o sieti. Potom sa už môžu zariadenia priamo prepojiť a prenášať medzi sebou aplikačné dáta.

Pri prepojení dvoch zariadení sa vyvolá metóda `onIceConnectionChange()` objektu `PeerConnection.Observer`, pomocou ktorej sa užívateľovi zobrazí obrazovka načítania pre synchronizáciu skóre (obr. 7.2). Súčasne sa po nadviazaní spojenia vyvolá metóda `onStateChange()` objektu triedy `DataChannel`. Keď tento objekt prejde do stavu „OPEN“, tak to znamená, že je možné vykonať prenos dát medzi zariadeniami. Pri synchronizácii skóre si

zariadenia automaticky vymenia všetky hodnotenia všetkých pesničiek, ktoré sa na danom zariadení nachádzajú. Práve preto sa prenášané dáta získajú takým istým spôsobom a v takom istom formáte ako pri vykonaní exportu skóre (kapitola 6.4). Výhodou synchronizácie je to, že užívateľ nemusí vyberať ani vytvárať žiadny súbor, keďže tieto dáta sa nebudú ukladať na úložisko ale posielať druhému zariadeniu. Všetky dáta sú pred odoslaním uložené v jednej premennej ako textový reťazec. Tým je možné určiť veľkosť prenášaných dát ako počet znakov tohto reťazca. Toto číslo je zaslané druhému zariadeniu pred samotným prenosom hodnotení, aby druhé zariadenie vedelo, koľko dát má prijať. Každý prenos dát sa vykonáva metódou `send()` objektu triedy `DataChannel`. Parametrom tejto metódy je objekt triedy `DataChannel.Buffer`, ktorý je inicializovaný dátami, ktoré sa majú preniesť a ktoré sa prevedú z textového reťazca na postupnosť bytov. Textový reťazec s hodnoteniami sa následne rozdelí na bloky o veľkosti maximálne 30 000 znakov a každý blok sa samostatne odošle druhému zariadeniu. Bez takéhoto rozdelenia by sa nemuseli správne preniesť všetky dáta. Daná hodnota bola určená po testovaní na viacerých zariadeniach. Príchod dát je aplikácii oznámený vyvolaním metódy `onMessage()` objektu `DataChannel.Observer`, pričom samotné dáta sa tejto metóde predajú parametrom. Pri prvom vyvolaní tejto metódy po nadviazaní spojenia získa aplikácia veľkosť dát, ktoré má prijať. Pri každom ďalšom vyvolaní sa ukladajú prijaté dáta do špeciálnej premennej a keď je počet znakov v tejto premennej rovnaký ako veľkosť dát získaná pri prvom vyvolaní, tak aplikácia vie, že ďalšie dáta už neprídu a že môže začať spracovávať prijaté dáta s hodnoteniami pesničiek.



Obr. 7.2: Na obrázku sa nachádza obrazovka načítavania, ktorá sa užívateľovi zobrazí po úspešnom prepojení dvoch zariadení a ktorá ho informuje o tom, že prebieha synchronizácia skóre. Vľavo je zobrazená táto obrazovka vo svetlom režime a vpravo v tmavom režime.

7.1.2 Prenos dát medzi Android a UWP aplikáciou

Prenos dát medzi mobilnou a počítačovou verziou prehrávača prebieha využitím technológie Bluetooth, ktorá umožňuje ich bezdrôtovú výmenu. Obidve vývojové platformy umožňujú použitie dvoch rôznych verzií tejto technológie:

1. klasický Bluetooth (RFCOMM) – táto verzia je vhodná pre operácie náročnejšie na batériu, akými je prenos médií a komunikácia medzi zariadeniami,

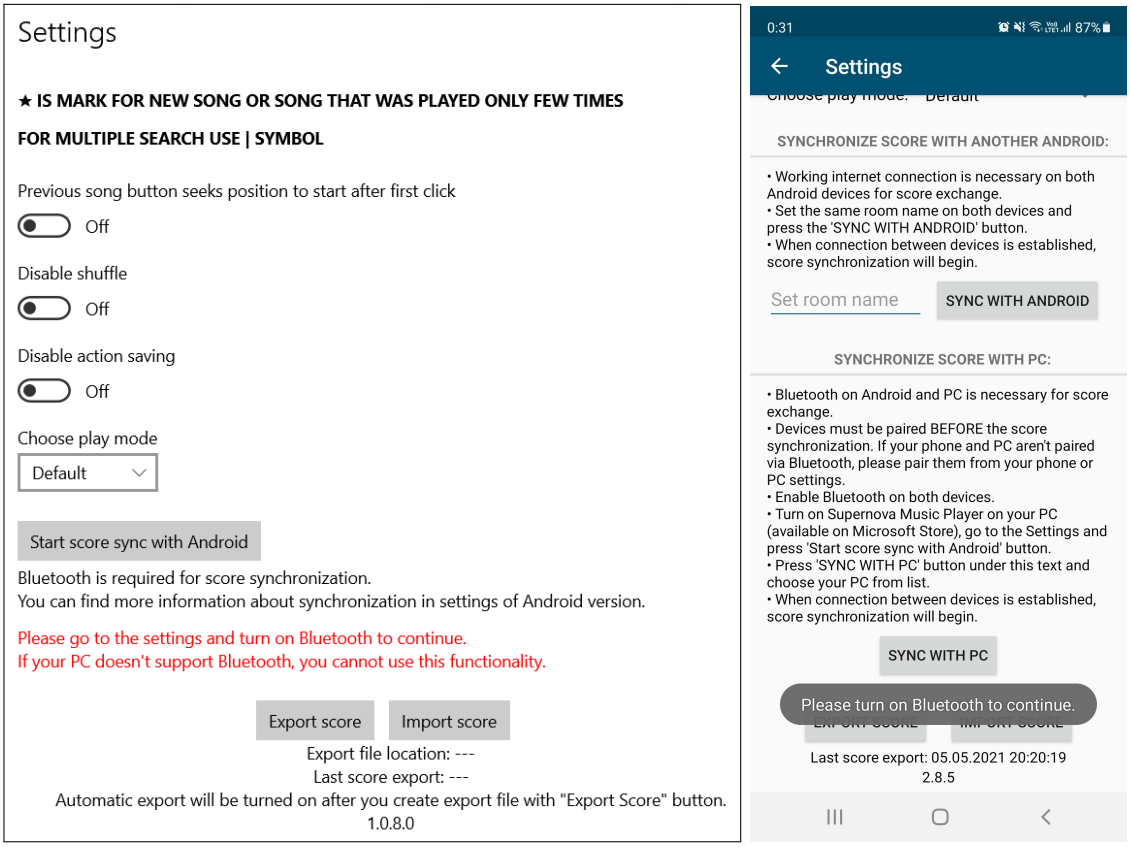
2. Bluetooth LE (Low-Energy) – táto verzia sa používa pre komunikáciu medzi zariadeniami, ktoré majú požiadavku na efektívne využívanie energie [4, Bluetooth overview[4]] [2, Bluetooth[5]].

Pre prenos dát bol zvolený klasický Bluetooth, keďže sa hudobný prehrávač bežne nepoužíva na zariadeniach, ktoré vyžadujú efektívne využívanie energie. Aby mohli obidve verzie prehrávača Bluetooth využívať, tak sa musia v manifeste aplikácie deklarovať príslušné povolenia. Vykonanie synchronizácie skóre cez technológiu Bluetooth má v mojej aktuálnej implementácii podmienku, že zariadenia musia byť pred synchronizáciou spárované. Spárovať zariadenia môže byť teoreticky pre niektorých užívateľov problém, no mobilná aplikácia cez pokyny na obrazovke nastavení (obr. 6.13) užívateľovi vysvetľuje, že dve zariadenia je možné spárovať cez systémové nastavenia počítača alebo chytrého telefónu. Výhodou použitia technológie Bluetooth je to, že synchronizáciu skóre je možné vykonať aj bez pripojenia na internet a komunikácia prebieha výlučne medzi dvoma spárovanými zariadeniami.

Kým komunikácia medzi dvoma Android zariadeniami prebieha na princípe peer-to-peer, tak komunikácia medzi Android zariadením a počítačom prebieha na princípe klient-server s tým, že počítačová aplikácia sa vždy správa ako server, ktorý po spustení čaká na to až sa k nemu pripojí klient – mobilná aplikácia. Spustenie Bluetooth serveru prebehne po stlačení tlačidla synchronizácie z obrazovky nastavení počítačovej verzie aplikácie. Bezprostredne po stlačení tlačidla sa na platforme UWP vykoná pokus o vytvorenie objektu triedy `RfcommServiceProvider` metódou `CreateAsync()`, ktorý predstavuje inštanciu lokálnej RFCOMM služby. Parametrom tejto metódy musí byť hodnota `RfcommServiceId.ObexObjectPush`. Ak pri volaní tejto metódy vznikne výnimka, tak to znamená, že užívateľ buď nemá na svojom zariadení zapnutý Bluetooth, alebo že jeho zariadenie túto technológiu ani nepodporuje. Na obrazovke nastavení sa zobrazí červený text, ktorý užívateľa na tento fakt upozorní (obr. 7.3). V opačnom prípade sa vytvorí objekt triedy `StreamSocketListener`. Cez tento objekt sa nastaví metóda, ktorú systém vyvolá pri nadviazaní spojenia s ďalším zariadením. Následne sa tieto dva objekty prepoja metódou `BindServiceNameAsync()` a objektu `RfcommServiceProvider` sa ešte nastavia dodatočné atribúty. Server môže byť potom spustený metódou `StartAdvertising()`.

Hneď ako je Bluetooth server spustený, tak sa k nemu môže užívateľ pripojiť cez svoj chytrý telefón. Po stlačení príslušného tlačidla z nastavení sa na platforme Android získa objekt triedy `BluetoothAdapter` metódou `getDefaultAdapter()` tejto triedy, ktorý je pre prácu s technológiou Bluetooth nevyhnutný. Keď táto metóda vráti hodnotu `null`, tak to znamená, že dané zariadenie Bluetooth nepodporuje a užívateľovi sa cez `Toast` oznámi, že synchronizáciu skóre nie je možné z tohto dôvodu vykonať. V opačnom prípade sa cez `BluetoothAdapter` skontroluje, či je Bluetooth zapnutý. Ak užívateľ pred spustením synchronizácie Bluetooth nezapol, tak sa mu využitím metódy `startActivityFor-`

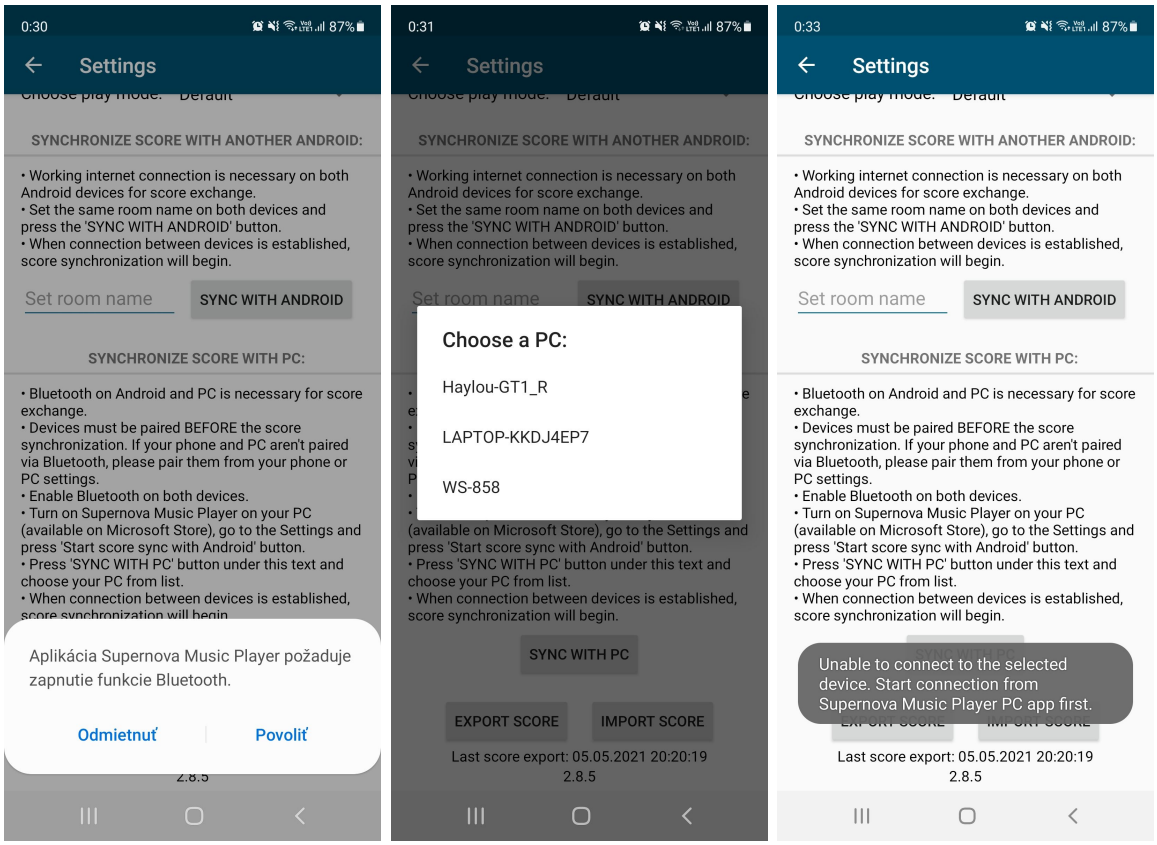
[4] https://developer.android.com/guide/topics/connectivity/bluetooth
[5] https://docs.microsoft.com/en-us/windows/uwp/devices-sensors/bluetooth



Obr. 7.3: Na obidvoch obrázkoch sa zobrazuje chyba, ktorá vznikne, keď užívateľ nezapne Bluetooth pre vykonanie synchronizácie skóre medzi Android zariadením a počítačom. Pravý obrázok zobrazuje túto chybu na chytrom telefóne a ľavý obrázok ju zobrazuje na počítači s tým, že na počítači sa zobrazí daná chyba aj vtedy, keď zariadenie nepodporuje Bluetooth.

`Result()` zobrazí systémové dialógové okno, ktorým sa ho systém opýta, či chce Bluetooth zapnúť (obr. 7.4). Aplikácia získa výsledok užívateľovej akcie pomocou metódy `onActivityResult()`. Ak užívateľ odmietne Bluetooth zapnúť, tak ho aplikácia cez `Toast` požiada, aby ho zapol a vykonávanie synchronizácie sa ukončí (obr. 7.3). Ak užívateľ povolí zapnutie alebo ak bol Bluetooth zapnutý už pred spustením synchronizácie, tak aplikácia získa zoznam spárovaných zariadení metódou `getBondedDevices()` objektu `BluetoothAdapter`. Užívateľovi sa zobrazí dialógové okno typu `AlertDialog` so zoznamom názvov spárovaných zariadení (obr. 7.4), z ktorého môže vybrať počítač, na ktorom je spustený Bluetooth server alebo proces synchronizácie ukončiť kliknutím mimo toto dialógové okno. Keď užívateľ klikne na niektorú položku zo zoznamu, tak prebehne pokus o nadviazanie spojenia s vybraným zariadením, ktoré je reprezentované ako objekt triedy `BluetoothDevice`. Cez objekt vybraného zariadenia sa metódou `createRfcommSocketToServiceRecord()` získa objekt triedy `BluetoothSocket`, ktorý na platforme Android slúži na vytvorenie spojenia s ďalším zariadením cez Bluetooth. Parametrom tejto metódy musí byť univerzálne unikátny identifikátor (`UUID`) s hodnotou: „00001105-0000-1000-8000-00805f9b34fb“. Táto hodnota totiž odpovedá použitej hodnote `RfcommServiceId.ObexObjectPush` na platforme UWP [2,

RfcommServiceId Class[6]]. Mobilná aplikácia potom iniciuje vytvorenie spojenia zavolaním metódy `connect()`. V prípade, ak užívateľ zvolil nesprávne zariadenie zo zoznamu alebo ak najskôr nespustil server z počítačovej aplikácie, tak vznikne výnimka, synchronizácia sa ukončí a užívateľ je o zlyhaní pokusu o pripojenie informovaný cez `Toast` (obr. 7.4). Ak užívateľ zvolil správne zariadenie, tak sa tieto dve zariadenia prepoja.



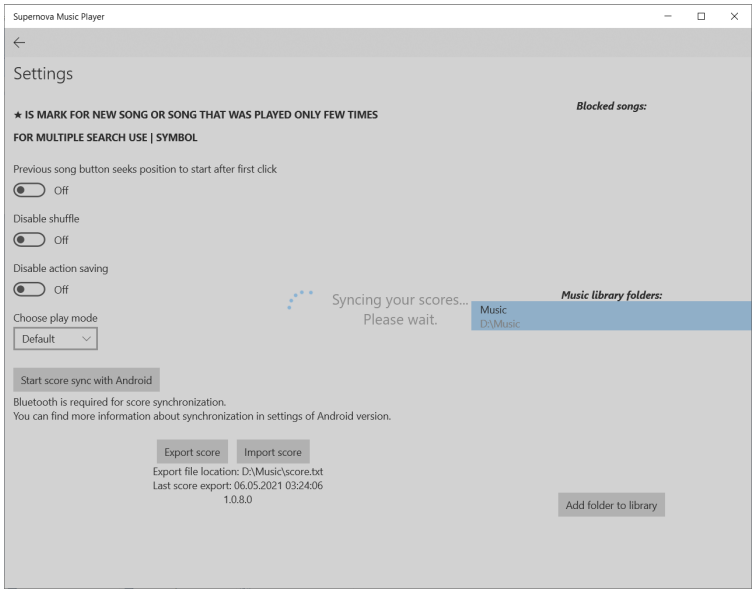
Obr. 7.4: Obrázok vľavo zobrazuje systémové dialógové okno, ktoré sa užívateľovi zobrazí, keď spustí synchronizáciu skóre medzi Android zariadením a počítačom bez toho, aby najskôr spustil Bluetooth. Na obrázku v strede je možné vidieť dialógové okno so zoznamom spárovaných zariadení, z ktorých môže užívateľ vybrať to, ku ktorému sa chce pripojiť. Obrázok vpravo zobrazuje chybovú hlášku, ktorá sa zobrazí vtedy, keď užívateľ zvolil zo zoznamu nesprávne zariadenie alebo nespustil Bluetooth server z počítačovej aplikácie.

Počítačová aplikácia získa socket pre komunikáciu s mobilným zariadením ako objekt triedy `StreamSocket` z parametru metódy, ktorá sa vyvolá pri prepojení zariadení. Z tohto socketu sa po úspešnom nadviazaní spojenia získa na obidvoch platformách objekt, ktorý umožní čítanie prichádzajúcich dát (`InputStream`) ako aj objekt, ktorý umožní odoslanie dát (`OutputStream`). Na platforme UWP sa tieto objekty ešte prevedú na objekt triedy `DataReader` a `DataWriter`. Následne obidve aplikácie vytvoria a spustia samostatné vlákno pre zaslanie dát a samostatné vlákno pre čítanie dát. Užívateľovi sa taktiež v tomto okamžiku zobrazí obrazovka načítavania, ktorá ho informuje o tom, že práve prebieha synchronizácia skóre (obr. 7.2 a 7.5). Prenos dát prebieha podobne ako pri použití technológie WebRTC. Na obidvoch platformách sa zo zariadenia získajú dáta obsahujúce hodnotenia

[6] https://docs.microsoft.com/en-us/uwp/api/windows.devices.bluetooth.rfcomm.rfcommserviceid

pesničiek rovnako ako pri exporte skóre (kapitola 6.4) a pred prenosom sa tieto dáta uložia do textového reťazca. Po získaní dát sa zistí ich veľkosť ako počet znakov tohto textového reťazca, pričom táto informácia sa odošle druhému zariadeniu ešte pred samotnými dátami. Výhodou technológie Bluetooth je, že všetky prenášané dáta je možné odoslať naraz. Prijímajúca strana sa dokáže vysporiadať aj s veľkým množstvom prichádzajúcich dát. Pri zaslaní dát druhému zariadeniu teda stačí previesť dáta textového reťazca na postupnosť bytov a všetky ich naraz odoslať. Pre odoslanie dát sa na platforme UWP využívajú metódy `WriteBytes()` a `StoreAsync()` objektu `DataWriter`. Na platforme Android sa pre tento účel používa metóda `write()` objektu `OutputStream`.

Vo vlákne pre čítanie dát sa cez cyklus `while` čaká na dáta s hodnoteniami z druhého zariadenia. Pri prvom príchode dát sa získa veľkosť prenášaných dát a prijímajúce zariadenie sa pokúsi prijať všetky dáta naraz, no to sa pri veľkom množstve dát nemusí vždy podariť. Získané dáta sa prevedú späť na textový reťazec a uložia sa do špeciálnej premennej. Pri ďalšej iterácii cyklu sa na základe celkovej veľkosti dát a počtu znakov v špeciálnej premennej zistí, či aplikácia získala všetky dáta. Ak neboli prijaté všetky dáta, tak aplikácia zistí koľko znakov ešte nebolo prijatých a znova sa pokúsi prijať všetky zvyšné dáta naraz. Tento cyklus sa vykonáva dovtedy, kým sa neprijmú všetky dáta. Pri testovaní som zistil, že dáta o veľkosti 37 000 znakov prijala aplikácia hneď v prvej iterácii, no pre úplné prijatie dát o veľkosti 170 000 znakov bolo nutné vykonať tri iterácie cyklu. Hneď ako aplikácia získa všetky dáta, tak sa vykonávanie cyklu ukončí a aplikácia začne spracovávať textový reťazec s prijatými hodnoteniami. Pre príjem dát sa na platforme UWP používa `DataReader` a jeho metódy `LoadAsync()`, `ReadInt32()` a `ReadString()`. Pre prijatie veľkosti prenášaných dát sa na platforme Android využíva metóda `read()` objektu `InputStream` a pre prijatie dát s hodnoteniami sa používa objekt triedy `DataInputStream` a jeho metóda `readFully()` [4, Bluetooth overview[4]].



Obr. 7.5: Obrázok znázorňuje obrazovku načítavania, ktorá sa užívateľovi zobrazí na počítači po nadviazaní spojenia s chytrým telefónom pri vykonávaní synchronizácie skóre cez Bluetooth.

7.2 Detekcia rozdielov a synchronizácia skóre

Synchronizácia skóre prebieha rovnakým spôsobom pre obidve technológie prenosu dát. Spracovanie prijatých dát musí na oboch zariadeniach prebehnúť totožným spôsobom, aby sa zaistilo, že synchronizáciou vzniknú totožné hodnoty skóre na obidvoch stranách. Toto spracovanie prebieha podobným spôsobom ako pri importe skóre (kapitola 6.4). Každej pesničke sa vytvorí identifikátor na základe jej umiestnenia, názvu, názvu interpreta a celkovej dĺžky trvania. Aplikácia sa potom snaží vyhľadať tento identifikátor v textovom reťazci s dátami z druhého zariadenia. Ak sa tento identifikátor nenájde ani keď sa z neho odstráni dĺžka trvania pesničky a ani po úprave veľkosti písmen, tak synchronizácia skóre pre túto pesničku nebude vykonaná a prehrávač automaticky prejde na ďalšiu pesničku. Ak sa identifikátor pesničky v prijatých dátach z druhého zariadenia nájde, tak prehrávač z nich získa hodnotenia danej pesničky. Synchronizácia sa od importu odlišuje tým, že prehrávač následne získa hodnotenia tej istej pesničky zo svojich zdieľaných preferencií alebo kontajnera. Obidvoma reťazcami s hodnoteniami sa následne začne prechádzať cyklom, ktorý zisťuje, či sú medzi reťazcami nejaké rozdiely. Počas toho sa vytvára nový reťazec s hodnoteniami, ktorý prípadné rozdiely zlúči a ktorý je výsledkom synchronizácie skóre pre danú pesničku.

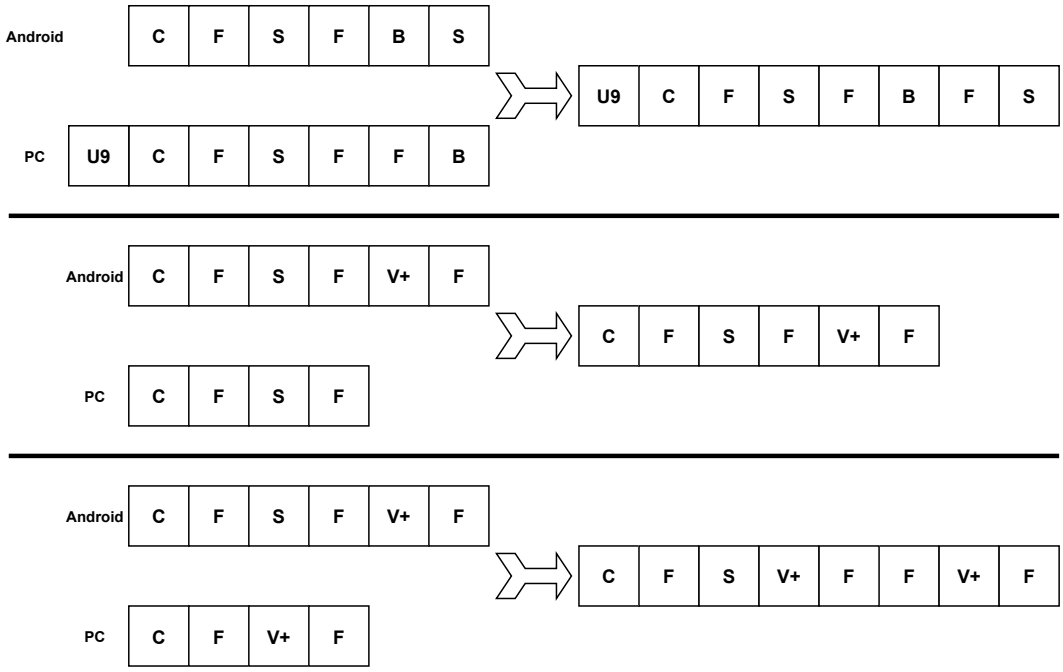
Pri vytváraní nového reťazca s hodnoteniami musí algoritmus synchronizácie špeciálne sledovať výskyt troch typov hodnotení, ktoré sa v tomto reťazci môžu vyskytovať len raz – inicializačné hodnotenie (typ „C“), explicitné hodnotenia od užívateľa (typ „UX“) a hodnotenie pre zablokovanie pesničky (typ „B“). Hodnotenie typu „UX“ je možné jednoducho detegovať ešte pred spustením cyklu, pretože ak danej pesničke bolo toto hodnotenie udelené, tak sa vždy nachádza na začiatku reťazca s hodnoteniami. Ak bolo toto hodnotenie udelené pesničke len na jednom zariadení, tak ho pesnička na druhom zariadení prevezme tiež (obr. 7.6). Ak ale má pesnička tento typ hodnotenia na obidvoch zariadeniach, tak sa na každom z nich ponechá pôvodná hodnota (ktorú reprezentuje písmeno „X“ v názve typu hodnotenia), keďže nie je možné určiť, ktorá z týchto dvoch hodnôt bola pesničke udelená neskôr a je tým pádom viac aktuálna. Udelenie hodnotenia typu „UX“ s rôznou hodnotou na obidvoch zariadeniach je teda jediný spôsob, akým môže aj po synchronizácii vzniknúť odlišné skóre tej istej pesničky. Po vložení tohto hodnotenia na začiatok nového reťazca sa toto hodnotenie z pôvodných reťazcov odstráni. Dôvodom je, že hodnotenie typu „C“ sa vždy nachádza za týmto hodnotením, keďže hodnotenie „C“ je udelené každej pesničke pri vytvorení jej záznamu v zdieľaných preferenciách alebo v kontajneri. Obidva pôvodné reťazce s hodnoteniami budú teda vždy začínať rovnako a to hodnotením typu „C“ a je možné nad nimi spustiť cyklus.

Počas cyklu sa prechádza obidvoma reťazcami po jednotlivých hodnoteniach a algoritmus synchronizácie sleduje nasledujúce:

- či sa už v niektorom z pôvodných reťazcov s hodnoteniami vyskytlo hodnotenie typu „B“, ktoré sa na rozdiel od predchádzajúcich dvoch typov hodnotení môže v reťazci vyskytovať na ľubovoľnej pozícii,
- či sú hodnotenia na aktuálnej a všetkých predchádzajúcich pozíciách rovnaké alebo či už vznikol medzi hodnoteniami rozdiel.

Pri prvom výskyte sa hodnotenie typu „B“ vloží do nového reťazca s hodnoteniami a všetky ďalšie výskyty tohto typu hodnotenia sú ignorované (obr. 7.6). Dokým sa na aktuálnej pozícii v pôvodných reťazcoch s hodnoteniami nachádza rovnaké hodnotenie, tak sa toto hodnotenie vždy pridá na koniec nového reťazca s hodnoteniami. Správanie sa mení po detekcii prvého rozdielu v týchto reťazcoch. Detekcia prvého rozdielu je dôležitá z toho hľadiska, že

na jej základe je možné určiť, ktoré hodnotenia získala pesnička vykonaním importu alebo predchádzajúcej synchronizácie a ktoré hodnotenia získala pesnička až potom. Detekcia prvého rozdielu nastane vtedy, keď sa na aktuálnej pozícii v reťazcoch nachádzajú rôzne hodnotenia alebo vtedy, keď je jeden reťazec dlhší ako druhý. Keďže po detekcii rozdielu nedokáže aplikácia určiť, ktoré z dvoch hodnotení na aktuálnej pozícii bolo udelené pesničke skôr, tak sa musí pre synchronizáciu určiť iný princíp, na základe ktorého sa hodnotenia vložia do nového reťazca. Pri synchronizácii skóre medzi chytrým telefónom a počítačom sa do nového reťazca vždy ako prvé vloží to hodnotenie, ktoré pochádza z chytrého telefónu a zaň sa vloží hodnotenie z počítača. Pri synchronizácii medzi dvoma Android zariadeniami sa do nového reťazca s hodnoteniami vloží ako prvé to hodnotenie, ktoré pochádza zo zariadenia, ktoré na signalizačnom serveri vytvorilo miestnosť (zariadenie A) a až potom sa vloží hodnotenie, ktoré pochádza zo zariadenia, ktoré sa do vytvorenej miestnosti len pripojilo (zariadenie B). Po tom ako sa detegoval prvý rozdiel sa týmto princípom spájajú všetky nasledujúce hodnotenia a to aj v prípade, ak sa na danej pozícii nachádza rovnaké hodnotenie. Keď je jeden reťazec s hodnoteniami dlhší ako druhý, tak sa všetky hodnotenia dlhšieho reťazca na pozíciách presahujúcich dĺžku kratšieho reťazca automaticky pridajú na koniec nového reťazca s hodnoteniami (obr. 7.6). Cyklus sa ukončí po prejdení všetkých hodnotení z obidvoch pôvodných reťazcov a nový reťazec s hodnoteniami nahradí ten pôvodný v zdieľaných preferenciách alebo v kontajneri. Následne sa pre zrýchlenie vykonávania synchronizácie skóre odstráni riadok danej pesničky z textového reťazca s hodnoteniami z druhého zariadenia a prehrávač prejde na ďalšiu pesničku. Po prejdení všetkých pesničiek sa obrazovka načítavania skryje, aktualizujú sa zobrazené hodnoty skóre v hlavnom zozname pesničiek a mobilná aplikácie ešte zobrazí `Toast`, ktorým užívateľovi oznámi, že skóre pesničiek bolo úspešne zosynchronizované.



Obr. 7.6: Obrázok zobrazuje tri rôzne prípady synchronizácie skóre pesničky medzi Android zariadením a počítačom. Keď sa vykonáva synchronizácia skóre medzi takýmito dvoma zariadeniami, tak sa po detekcii prvého rozdielu do nového reťazca najskôr vloží hodnotenie z Androidu a zaň sa vloží hodnotenie z počítača.

Kapitola 8

Vylepšenia existujúcej mobilnej verzie hudobného prehrávača

Základ mobilnej verzie hudobného prehrávača bol výstupom mojej bakalárskej práce. Hudobný prehrávač síce poskytoval už v tomto stave žiadanú funkcionalitu a dokázal sa naučiť, ktoré pesničky má užívateľovi prehrávať na základe jeho preferencií, no aplikácia mala stále dostatok priestoru pre jej ďalšie vylepšovanie. Úlohou tejto kapitoly je predstaviť tieto vylepšenia, pričom implementačné detaily sú bližšie popísané v predchádzajúcich kapitolách. Vylepšenia existujúcej mobilnej verzie hudobného prehrávača je možné rozdeliť do dvoch kategórií:

1. vylepšenia funkcionality prehrávača,
2. vylepšenia užívateľského rozhrania prehrávača.

Keďže počítačová verzia začala vznikať až po dokončení mobilnej verzie hudobného prehrávača, tak všetky relevantné vylepšenia boli v počítačovej verzii aplikované už pri jej vývoji.

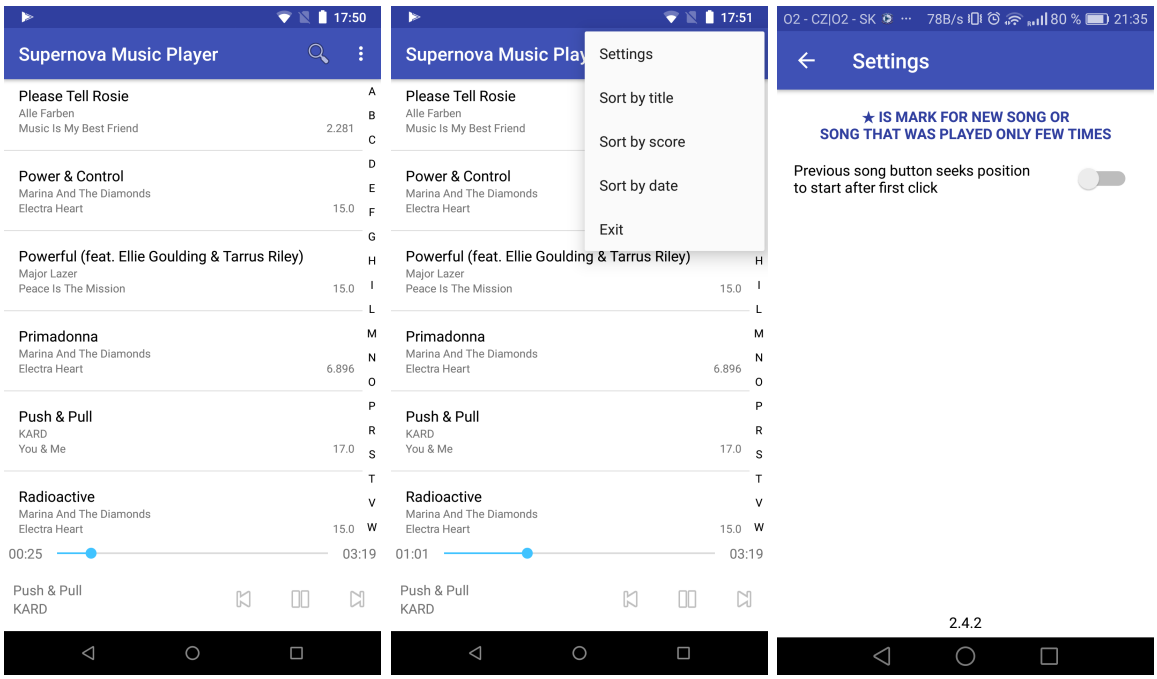
Vylepšenia funkcionality prehrávača je možné ďalej rozdeliť na tri typy:

- úpravy a vylepšenia existujúcej funkcionality,
- pridanie novej funkcionality,
- opravy chýb.

Najväčšie úpravy existujúcej funkcionality boli vykonané v algoritme výpočtu skóre pesničky, ktorý tvorí základ celého hudobného prehrávača. Jednou z týchto úprav je, že pesničkám môžu byť udelené tri nové typy hodnotení: typ „F+“, typ „UX“ a typ „B“, pričom ich význam je detailne popísaný v tabuľke 5.1. Ďalšou úpravou je, že pri výpočte skóre sa začala využívať priemerná hodnota skóre, ktorá sa aktualizuje pri každom spustení prehrávača. Algoritmus výpočtu skóre pesničky bol upravený aj tým spôsobom, že novozískaným hodnoteniam je prideľovaná väčšia váha ako tým hodnoteniam, ktoré boli pesničke udelené hneď po inštalácii aplikácie. Hodnota určitého typu hodnotenia sa taktiež zvyšuje, keď sa daný typ hodnotenia bezprostredne za sebou opakuje. Vďaka všetkým týmto zmenám dokáže prehrávač lepšie a rýchlejšie reagovať na zmeny v užívateľových preferenciách. Užívateľ môže dokonca sám ovplyvniť správanie tohto algoritmu a to výberom jedného z pridaných módov prehrávania. Užívateľ taktiež môže vykonávať viacnásobné vyhľadávanie, vypnúť náhodné prehrávanie alebo zabrániť ukladaniu nových hodnotení. Úpravou prešiel aj koncept

`audio focusu`, ktorý bol vylepšený tak, aby sa prehrávanie znova nespustilo, keď užívateľ počas krátkej straty `audio focusu` odpojí slúchadlá. Prehrávač si v aktuálnej verzii pri vypnutí neukladá desať, ale až pätnásť pesničiek zo zoznamu predchádzajúcich pesničiek a navyše aj päť pesničiek zo zoznamu nasledujúcich pesničiek.

Najväčšia zmena v aplikácii bola spôsobená pridaním widgetu. Pôvodne sa vykonávalo načítavanie pesničiek zo zariadenia a výpočet skóre v aktivite hlavnej obrazovky, keďže práve táto aktivita predstavovala vstupný bod aplikácie. Po načítaní pesničiek a výpočte ich skóre sa získaný zoznam pesničiek predal službe, ktorá mohla spustiť prehrávanie. Widget ale musí byť schopný spustiť prehrávanie aj keď aplikácia nebeží, pričom by mal komunikovať len so službou, keďže tá zabezpečuje ovládanie prehrávania. Služba pri spustení prehrávania z widgetu nemala pri takejto implementácii k dispozícii zoznam pesničiek, keďže aplikácia nebola spustená cez aktivitu, čo spôsobovalo pád aplikácie. Widget síce teoreticky môže spustiť aktivitu hlavnej obrazovky, ktorá následne spustí službu a predá jej zoznam s pesničkami, no to by ale spôsobovalo zlú užívateľskú skúsenosť, pretože spustenie aktivity je vždy sprevádzané zobrazením jej užívateľského rozhrania. Účelom widgetu je avšak rýchle spustenie a ovládanie prehrávania bez toho, aby bolo nutné zobraziť celé užívateľské rozhranie aplikácie. Potencionálnym riešením by mohlo byť získavanie pesničiek v triede widgetu, no takéto riešenie by len vytvorilo duplicitu kódu. Lepším a zvoleným riešením bol presun získavania pesničiek a výpočtu skóre priamo do služby. Služba sa teda sama postará o to, aby mala k dispozícii zoznam pesničiek, bez ohľadu na to, či bolo prehrávanie spustené cez widget alebo aktivitu. Ďalšími novými vylepšeniami prehrávača je okrem widgetu aj export a import skóre, synchronizácia skóre, časovač vypnutia alebo možnosť vymazať pesničku zo zariadenia.



Obr. 8.1: Na obrázkoch je zobrazené pôvodné užívateľské rozhranie hudobného prehrávača. Na ľavom a strednom obrázku je zobrazená hlavná obrazovka a na obrázku vpravo je zobrazená obrazovka nastavení.

Pôvodná verzia hudobného prehrávača (obr. 8.1) pozostávala len z dvoch obrazoviek – z hlavnej obrazovky a z obrazovky nastavení. Obidve tieto obrazovky prešli rôznymi zmenami (obr. 6.1 a 6.11) a v aplikácii navyše pribudli ďalšie dve obrazovky – obrazovka so zoznamom predchádzajúcich a nasledujúcich pesničiek (obr. 6.14) a obrazovka so zoznamom blokovaných pesničiek (obr. 6.15). Do položky hlavného zoznamu pesničiek bol pridaný obrázok danej pesničky a pri dlhom kliknutí na položku tohto zoznamu sa užívateľovi zobrazí vyskakovacie menu, ktoré mu umožňuje vykonať rôzne akcie s danou pesničkou. Ďalšou zmenou je, že položka práve prehrávanej pesničky sa v zozname dodatočne zvýrazní tučným písmom. Prehrávač si v aktuálnej verzii taktiež ukladá každú zmenu usporiadania hlavného zoznamu pesničiek, takže zoznam bude pri spustení zoradený podľa toho istého usporiadania, v akom sa nachádzal pri vypnutí aplikácie. Mnohými zmenami prešiel aj ovládací panel. Pre túto časť hlavnej obrazovky boli vykonané zmeny jej vzhľadu a rozloženia jednotlivých prvkov užívateľského rozhrania pre zabezpečenie lepšieho ovládania a pre výraznejšie oddelenie tejto časti od hlavného zoznamu pesničiek. Okrem toho tu taktiež pribudol obrázok práve prehrávanej pesničky a tlačidlo, ktorého stlačením sa zobrazí vyskakovacie menu, cez ktoré je možné s práve prehrávanou pesničkou vykonávať rôzne akcie bez toho, aby bolo nutné ju nájsť v hlavnom zozname pesničiek. Zmeny nastali aj v obsahu hlavného menu. Veľkou zmenou prešla aj obrazovka nastavení, v ktorej pribudlo množstvo nových prvkov užívateľského rozhrania, ktoré súvisia s pridaním novej funkcionality. Výrazným vylepšením užívateľského rozhrania je aj to, že aplikácia podporuje systémový tmavý režim ako to je možné vidieť na obrázkoch v celej kapitole 6.1.1. Zmeny nastali aj mimo aplikácie – do notifikácie hudobného prehrávača bolo pridané tlačidlo pre vypnutie prehrávania a na zariadeniach s novšou verziou operačného systému je taktiež možné správne zobraziť a meniť aktuálnu pozíciu prehrávania (obr. 6.36).

Výsledkom všetkých týchto zmien je aj to, že aplikácia prešla z otvorenej bety do produkčnej verzie. Tento prechod vyžadoval kompletné prerobenie stránky aplikácie v obchode Google Play. Na stránku aplikácie bol pridaný nový text v angličtine popisujúci všetky základné vlastnosti prehrávača, ako aj aktuálne obrázky zobrazujúce užívateľské rozhranie prehrávača.

Kapitola 9

Testovanie aplikácií a získavanie nových užívateľov

Pri zverejnení obidvoch verzií prehrávača v obchodoch aplikácií boli aplikácie najskôr vydané v režime pre beta testovanie. Aplikácie v tomto štádiu okrem mňa testoval aj vedúci tejto diplomovej práce a aj moji známi z okruhu rodiny, priateľov a spolužiakov. Týchto testerov som pravidelne kontaktoval, aby som sa uistil, či aplikácia funguje správne, prípadne ma kontaktovali aj sami pri vzniku nejakého problému. Okrem toho som od nich zisťoval, čo im v aplikácii chýba alebo ako by prehrávač vylepšili. Výsledkom toho je napríklad pridanie časovača vypnutia v mobilnej verzii alebo umožnenie viacnásobného vyhľadávania v obidvoch verziách hudobného prehrávača, pričom každá zmena alebo pridaná funkcionalita bola znova otestovaná. Svojich známych som taktiež požiadal, aby moju aplikáciu odporučili svojím priateľom, vďaka čomu získala moja aplikácia ďalších nových používateľov.

Aplikácie prešli aj testom použiteľnosti. Test použiteľnosti spočíva v sledovaní ľudí pri používaní aplikácie so zámerom zistenia toho, ako uľahčiť jej používanie alebo so zámerom dokázať, že aplikácia už ľahko použiteľná je. Tento typ testu prebieha tak, že užívateľom sa zadá niekoľko úloh, pričom sa od nich vyžaduje, aby nahlas premýšľali počas toho, ako zadané úlohy vykonávajú. Test použiteľnosti umožňuje odhalenie problémov na základe toho, že vývojári vedia ako má aplikácia fungovať, ale užívatelia väčšinou nie. Kvôli pandemickým opatreniam som vykonával toto testovanie len s rodinnými príslušníkmi, no pre odhalenie najzávažnejších problémov postačujú traja užívatelia, ktorých som k dispozícii mal. Tým som zadal najzákladnejšie úlohy, akými je spustenie prehrávania alebo prechod na ďalšiu pesničku, ale aj zložitejšie úlohy, akými je explicitná úprava skóre užívateľom alebo zmena módu prehrávania. Pomocou tohto testu bolo odhalené napríklad to, že užívatelia nevedeli, ako vykonať viacnásobné vyhľadávanie, a tak bola do nastavení pridaná poznámka, ktorá im to vysvetľuje [13].

Mobilná verzia aplikácie bola obzvlášť dôkladne otestovaná na týchto zariadeniach:

- Honor 7,
- Huawei P9 Lite 2017,
- Nokia 1,
- Pixel 3a,
- Samsung Galaxy Note10+,

- Samsung Galaxy Note20 Ultra,
- Samsung Galaxy S20 FE 5G,
- Samsung Galaxy S8+,
- Xiaomi Mi 10.

Počítačová verzia bola dôkladne otestovaná na zariadeniach s operačným systémom Windows 10. Konkrétne sa jedná o stolové počítače a notebooky od výrobcov HP, Lenovo, Asus, Dell a Acer.

Pre overenie toho, že aplikácie pracujú na obidvoch platformách správne pravidelne kontrolujem záznamy aplikácií cez platformy Google Play Console a Microsoft Partner Center, cez ktoré sa môžu vývojári dozvedieť o chybách, ktoré v aplikácii vznikli a čo ich spôsobilo. Vždy som sa snažil odstrániť tieto chyby najrýchlejšie ako to šlo a vydať novú verziu aplikácie, aby k nim viac nedochádzalo. Najviac chýb pri vývoji aplikácie vzniklo tým, že som na platforme Android zvýšil cieľovú verziu operačného systému (atribút `targetSdkVersion` v súbore s názvom „build.gradle“) bez toho, aby som najskôr preštudoval všetky zmeny, ktoré boli vykonané v operačnom systéme medzi pôvodnou a novou cieľovou verziou. To spôsobilo napríklad to, že sa v aplikácii nezobrazovali správne obrázky, nebolo možné vykonať import a export a aplikácia taktiež nedokázala pesničku zo zariadenia vymazať.

Po odstránení všetkých týchto chýb mohla byť mobilná verzia prehrávača vydaná v produkčnej verzii. Po dokončení implementácie počítačovej verzie a po jej otestovaní bola aj táto verzia hudobného prehrávača vydaná mimo beta testovanie. Po vydaní obidvoch verzií prehrávača v produkčnej verzii som sa snažil aplikáciu spropagovať a získať nových užívateľov aj mimo okruh mojich známych. Ako prvé ma napadlo kontaktovať webové portály, ktoré sa venujú mobilným a počítačovým aplikáciám a to špeciálne tie, ktoré už v minulosti vydali články, recenzie alebo porovnania iných hudobných prehrávačov. Vytvoril som teda text popisujúci všetky základné vlastnosti môjho hudobného prehrávača a odoslal ho nasledujúcim portálom:

- https://appauthority.com/,
- https://www.techradar.com/,
- https://fossbytes.com/,
- https://www.makeuseof.com/,
- https://viebly.com/,
- https://www.androidauthority.com/,
- https://www.androidcentral.com/,
- https://www.androidpolice.com/,
- https://www.mojandroid.sk/.

Keď som ani po dlhšej dobe nedostal z týchto portálov žiadnu spätnú väzbu, tak som tieto portály kontaktoval znova, no v čase písania tejto práce som stále žiadnu reakciu nedostal. Okrem kontaktovania týchto portálov som sa snažil priamo osloviť aj samotných užívateľov aplikácií. Popis aplikácie spolu s odkazmi na stiahnutie som teda vložil na fórum webového

portálu xda-developers[1], kde vývojári môžu prezentovať svoje aplikácie potencionálnym užívateľom a ďalším vývojárom. To isté som spravil aj na internetovom portáli reddit, konkrétne vo vlákne s názvom „r/androidapps“[2]. Aplikácia bola na týchto portáloch prijatá kladne – ľudia v komentároch chválili zaujímavý koncept aplikácie, dali mi vedieť, že si aplikáciu stiahli a poskytli mi taktiež spätnú väzbu. Napríklad mi pre získanie a udržanie ďalších užívateľov bolo odporučené viac experimentovať s užívateľským rozhraním. Ďalším nápadom, ktorý ma zaujal, je umožnenie vyhľadávania pesničiek podľa ich hudobného žánru.

[1] https://forum.xda-developers.com/f/android-apps-and-games.530/
[2] https://www.reddit.com/r/androidapps/

Kapitola 10

Záver

Výsledkom mojej diplomovej práce je hudobný prehrávač, ktorý dokáže na chytrom telefóne a na počítači prehrávať hudbu z daného zariadenia inovatívnym spôsobom. Tento inovatívny spôsob prehrávania hudby berie do úvahy preferencie užívateľa. Akcie užívateľa počas počúvania hudby totiž priamo ovplyvňujú to, ktoré pesničky sa budú prehrávať častejšie, ktoré pesničky sa budú prehrávať menej často a ktoré vôbec. Existujúca mobilná verzia hudobného prehrávača prešla mnohými zmenami, či už v oblasti funkcionality, tak aj v oblasti užívateľského rozhrania. Tieto zmeny aplikáciu zatraktívnili, zlepšili fungovanie samotného inovatívneho spôsobu prehrávania hudby a pridali do prehrávača nové funkcie, z ktorých sú mnohé obľúbené u užívateľov aj v iných prehrávačoch hudby. Na základe prehrávača pre chytré telefóny bol vytvorený kompatibilný prehrávač pre počítače. Ten funguje na rovnakom princípe, no súčasne je jeho užívateľské rozhranie prispôsobené počítačovým obrazovkám. Jedným z hlavných úspechov mojej diplomovej práce je, že aplikácie medzi sebou dokážu komunikovať a vymieňať si dáta bezpečným spôsobom. Pre komunikáciu medzi mobilnou verziou prehrávača a počítačovou verziou sa používa technológia Bluetooth a pre komunikáciu medzi dvoma Android zariadeniami sa používa technológia WebRTC. Podarilo sa mi taktiež získať nových užívateľov, ktorí vytvorený prehrávač hudby aktívne používajú. Aplikácia bola riadne otestovaná a zverejnená v obchodoch aplikácií Google Play[1] a Microsoft Store[2].

V budúcnosti chcem ďalej vylepšovať funkcionalitu prehrávača. Jedným z plánovaných vylepšení je automatické napĺňanie zoznamu nasledujúcich pesničiek. Prehrávač do tohto zoznamu automaticky pridá zopár pesničiek, aby mohol užívateľ dopredu zistiť, aké pesničky pre neho prehrávač vybral. Pesničky vybrané prehrávačom budú v tomto zozname odlíšené od tých, ktorých prehrávanie naplánoval sám užívateľ a keď užívateľ takúto pesničku zo zoznamu odstráni, tak sa jej udelí nový typ hodnotenia so zápornou hodnotou. Taktiež plánujem pridať nový mód prehrávania, ktorý by prehrával menej obľúbené pesničky, aby mohol užívateľ zvýšiť skóre tým pesničkám, ktorých skóre je nízke z toho dôvodu, že na ne užívateľ len v danom okamihu nemal náladu a preskočil ich. Skutočne neobľúbeným pesničkám sa týmto módom prehrávania skóre ešte viac zníži, čo môže byť pre užívateľa znamenie, že takúto pesničku môže zo zariadenia odstrániť. Značnú časť mojej budúcej pozornosti si zaslúži aj užívateľské rozhranie aplikácie, ktorému chcem vytvoriť modernejší a prívetivejší vzhľad, no stále zachovať minimalistický dizajn.

[1] https://play.google.com/store/apps/details?id=com.starko.supernovaMP
[2] https://www.microsoft.com/sk-sk/p/supernova-music-player/9nc35kzh1pts

Literatúra

[1] Krug, S.: *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders, 2014, ISBN-13: 978-0321965516.

[2] Microsoft: *Windows Developer*.
URL https://docs.microsoft.com/en-us/windows/uwp/

[3] Open Handset Alliance: *Industry Leaders Announce Open Platform for Mobile Devices*. [Online; navštívené 25.12.2020].
URL http://www.openhandsetalliance.com/press_110507.html

[4] Google Developers: *Android Developers*.
URL https://developer.android.com/

[5] Lake, I.: Layouts, Attributes, and you. *Android Developers*, 01 2016.

[6] Buley, L.: *The User Experience Team of One: A Research and Design Survival Guide*. Rosenfeld Media, 2013, ISBN-13: 978-1-933820-18-7.

[7] Sims, G.: I want to develop Android apps — What languages should I learn? *Android Authority*, 08 2019.

[8] Lockwood, A.: Content Providers & Content Resolvers. *Android Design Patterns*, 06 2012.

[9] Dutton, S.: Get Started with WebRTC. *HTML5 Rocks*, 11 2020.

[10] Dutton, S.: Build the backend services needed for a WebRTC app. *HTML5 Rocks*, 11 2020.

[11] Chanddru, V.: PeerConnection — Getting Started with WebRTC — Part 2. *Medium*, 04 2017.

[12] Maindola, S.: Step by Step Guide to Build WebRTC Native Android App. *Medium*, 05 2020.

[13] Krug, S.: *Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability Problems*. New Riders, 2010, ISBN-13: 978-0-321-65729-9.