VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

MOBILNÍ PŘEHRÁVAČ PRO PŘÍJEM A ZPRACOVÁNÍ MULTIMEDIÁLNÍCH DAT

MOBILE PLAYER FOR MULTIMEDIA CONTENT PRESENTATION AND PROCESSING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. TOMÁŠ KRÓNER
AUTHOR

VEDOUCÍ PRÁCE Ing. JAKUB MÜLLER
SUPERVISOR

BRNO 2009

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Tomáš Króner	***ID:***	83574
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Mobilní přehrávač pro příjem a zpracování multimediálních dat

POKYNY PRO VYPRACOVÁNÍ:

Seznámte se s možnostmi návrhu mobilních aplikací. Navrhněte vhodnou realizaci aplikace pro mobilní zařízení schopnou příjmout a přehrávat multimediální obsah s využitím platformy JavaME.

DOPORUČENÁ LITERATURA:

[1] BURGET, R., KOMOSNY, D. Real-time control protocol and its improvements for Internet Protocol Television. International Transaction on Computer Science and Engineering, ISSN 1738-6438, 2006, roč. 2006, č. 31, s. 1 - 12.

[2] KOMOSNY D., NOVOTNY V. Tree Structure for Source-Specific Multicast with feedback Aggregation, in ICN07 - The Sixth International Conference on Networking . Martinique, 2007, ISBN 0-7695-2805-8

[3] GOYAL V., Pro Java ME MMAPI: Mobile Media API for Java Micro Edition

Termín zadání:	9.2.2009	***Termín odevzdání:***	26.5.2009

Vedoucí práce: Ing. Jakub Müller

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práve třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

ABSTRAKT

Diplomová práce pojednává o vývoji multimediální aplikace umožňující přehrávání a příjem multimediálních souborů, jako jsou audio a video. V úvodu práce je v několika kapitolách shrnutá teorie ohledně programovacího jazyka J2ME, jeho částmi a dělením. Vývoj aplikace začíná výběrem cílové skupiny a zároveň skupinou zařízení, které vycházejí ze současných trendů mobilních zařízení. Další část se zabývá volbou vývojového nástroje NetBeans IDE na základě požadavku pro vývoj multimédií a také volbou vývojového kitu Nokia SDK a jeho krátkým popisem. Následně je stručně charakterizována aplikace a její základní třídy a metody. Z hlediska rozdělení popisu vývoje, jsou kapitoly tříd a metod dělené podle úrovně algoritmu. Jednotlivé funkce jsou obohaceny o vývojové diagramy pro doplnění textu, na kterých jsou zakreslené důležité body programu. Hlavní kapitoly popisující části algoritmu začínají popisem přehrávání audia a videa, v následujících stranách zahrnují popis síťového spojení a přenosu dat. V každé části je nejdřív rozebrána programátorská část, algoritmus a následně také obecné možnosti a funkce dané oblasti. V části přehrávání ze souboru, je to výběr souboru a postup k zahájení přehrávání s možností ovládání různých funkcí. Při síťovém připojení je to přístup na síť a práce s daty, případně vzájemná komunikace v hierarchii klient – server a následné přehrávání multimediálního obsahu. Čtenář je v závěru popisu vývoje obeznámený s ošetřením možných výjimek programu, které jsou neoddělitelnou součástí každé aplikace pracující se vstupně-výstupními daty. Další kapitola je věnována budoucímu vývoji a návrhu, vyplývajícího z dosavadního zpracování. V práci je k nalezení také popis aplikace z hlediska běžného uživatele, návod na obsluhu a řešení vzniklých problémových situací. V návodu jsou obsažené snímky popisovaných částí programu. Dosažené výsledky a zhodnocení je shrnuté v závěru práce.

KLÍČOVÁ SLOVA

Java Micro Edition, CLDC, MIDP 2.0, přehrávač, audio, video

ABSTRACT

Master's thesis deals with developing of multimedia application that allow play and receive multimedia files as audio and video. In thesis introduction, there is theory overview about programming language J2ME, its parts and divisions in couple of chapters. The applications developing begin by choosing focus groups and also group of devices that agree with present trends about mobile devices. The next part undertakes selecting development tool NetBeans IDE based on requirements for multimedia develop and also undertakes selecting software development kit Nokia SDK and its short definition. Afterwards, the application and its basic classes and methods are briefly characterized. The chapters of classes and methods are divided according algorithm level, following development description dividing aspect. Program functions are enriched with flow diagrams completing written text, which have marked important points of program. The main chapters describing algorithm parts begin with define audio and video play. The other pages include description of network connection a data transfers. In every part the programmer part, algorithm is analyzed first and than general options and functionality of this area. In file play part, it is selection of file and proceeding to launch playing with function control possibility. In case of network connection, it is access to network and data handling, eventually communication with each other in client - server hierarchy and consequential multimedia content playing. In the end of development part, reader is cognizant of application fault handling that is inherently part of every application working with I/O data. Next chapter is devoted to future development a design resulting to present process. In the thesis it can be found also application description from user's point of view, user manual a solving incurred problem situations. In the instructions, there are contained screenshots of described parts of program. Reached results and reviews are summarized on the end of the thesis

KEYWORDS

Java Micro Edition, CLDC, MIDP 2.0, player, audio, video

BIBLIOGRAFICKÉ CITACE MÉ PRÁCE

KRÓNER, T. *Mobilní přehrávač pro příjem a zpracování multimediálních dat*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 64 s. Vedoucí diplomové práce Ing. Jakub Müller.

PREHLÁSENIE

Prehlasujem, že svoju diplomovú prácu na tému „Mobilní přehrávač pro příjem a zpracování multimediálních dat“ som vypracoval samostatne pod vedením vedúceho diplomovej práce a s použitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky uvedené v zozname literatúry na konci práce.

Ako autor uvedeného diplomovej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, predovšetkým som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a som si plne vedomí následkov porušenia ustanovenia § 11 a nasledujúcich autorských zákonov č. 121/2000 Zb., vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovení § 152 trestného zákona č. 140/1961 Zb.

V Brne dňa ....

podpis autora

POĎAKOVANIE

Ďakujem vedúcemu diplomovej práce Ing. Jakubovi Müllerovi, za veľmi užitočnú metodickú pomoc a cenné rady pri spracovaní diplomovej práce.

V Brne dňa ....

podpis autora

OBSAH

Úvod **11**

1. Technológia java a j2ME **12**

1.1. Java Micro Edition 12
1.2. Konfigurácia 13
1.3. Profily 15
1.4. MIDlet 17
1.4.2. MIDlet užívateľské rozhranie 18
1.4.3. High Level 19
1.4.4. Low Level 20
1.5. JAR a JAD súbory 20
1.6. Mobile Media API 21

2. Mulimediálny prehrávač **23**

2.1. Výber cieľovej skupiny 23
2.2. Vytvorenie projektu 23
2.2.1. NetBeans IDE 23
2.2.2. Voľba vývojového kitu 24
2.3. Úvod do aplikácie 25
2.4. Základné metódy a triedy aplikácie 25
2.4.1. Trieda MobilePlayer 25
2.4.2. Trieda ButtonCanvas 32
2.5. Prehrávanie audia zo súboru 34
2.5.1. Trieda PlayVlakno 34
2.5.2. Dostupné funkcie pri prehrávaní audia 34
2.6. Prehrávanie videa zo súboru 35
2.6.1. Trieda PlayVideo 35
2.6.2. Dostupné funkcie pri prehrávaní videa 35
2.7. Audio stream 36
2.7.2. Trieda URLaddress 37
2.7.3. Record Store –Záznam adries 39
2.7.4. Prehrávanie audio streamu – triedy a metódy 41
2.8. Odoslanie a príjem video sekvencie 44
2.8.2. Záznam a odoslanie videa – triedy a metódy 46
2.8.3. Príjem a prehranie videa – triedy a metódy 49
2.9. Výnimky aplikácie a ich ošetrenie 53
2.9.1. Audio prehrávanie zo súboru 53
2.9.2. Video prehrávanie zo súboru 53
2.9.3. Audio stream 53
2.9.4. Záznam, prenos a príjem videa 54
2.10. Budúce rozšírenie aplikácie 55
2.11. Popis multimediálneho prehrávača 56
2.11.2. Chybové hlásenia 59

3. Záver **61**

Literatúra **62**

ZOZNAM OBRÁZKOV

Obr. 1.1. Prehľad architektúry pracovného prostredia J2ME 13
Obr. 1.2. Profily J2ME 15
Obr. 1.3. Životný cyklus MIDlet 17
Obr. 1.4. Rozdelenie High Level a Low Level 19
Obr. 2.1. Emulátory S60 3rd Edition FP1 a FP2 24
Obr. 2.2. Vývojový diagram metódy commandAction 27
Obr. 2.3. Vývojový diagram metódy getFileMng 29
Obr. 2.4. Návratové hodnoty tlačidiel 33
Obr. 2.5. Vývojový diagram audio streamu. 37
Obr. 2.6. Vývojový diagram metódy OpenURLStore 40
Obr. 2.7. Vzájomná komunikácia pri prenose video sekvencie 46
Obr. 2.8. Vývojový diagram záznamu a prenosu video sekvencie 52
Obr. 2.9. Úvodná obrazovka 56
Obr. 2.10. a) Hlavná obrazovka s menu b) Súborový manažér 57
Obr. 2.11. a) Prehrávač videa b) Prehrávač audia 58
Obr. 2.12. a) Záznam z kamery b) Príjem záznamu 59

ZOZNAM TABULIEK

Tab. 2.1. Zoznam parametrov na prístup k zariadeniam v mobilnom zariadení 47

ÚVOD

V dnešnej dobe je pre každého úplnou samozrejmosťou mať mobilný telefón. Je to odvetvie spotrebnej elektroniky, ktoré prechádza najrýchlejším vývojom čo nám, užívateľom, prináša mnohé klady.

Štandardom poslednej doby je použitie programovacieho jazyku Java v mobilných telefónoch. Táto implementácia otvára celú radu možností, ako rozšíriť funkcie mobilného telefónu podľa potrieb jeho používateľa. Mobilný telefón sa tak stáva akousi prenosnou, vreckovou alternatívou počítača. Samozrejme, nemá také možnosti ako PC, ale ako ukazuje vývoj možností mobilných zariadení, k dispozícií je stále viac aplikácií napísaných v programovacom jazyku J2ME. J2ME je skratka Java 2 Micro Edition a predstavuje rozšírenie javy pre vývoj aplikácií vyhovujúcich štandardu podporovaného mobilnými zariadeniami.

Cieľom tejto práce je vytvorenie aplikácie, ktorá dokáže prijímať a prehrávať multimediálny obsah, tzn. video a audio, v mobilnom telefóne. To v poslednej dobe dospieva k veľkej obľube a to hlavne rastom možností a výkonového potenciálu mobilných zariadení. Práca sa ďalej zaoberá sieťovým pripojením na vzdialené servery a komunikáciou medzi zariadeniami pripojenými do ethernetu.

Aplikácia je určená pre mobilné telefóny s podporou Javy (J2ME) s farebným displejom, s dostatočným výkonom pre dekódovanie videa, CLDC 1.0 konfiguráciou, ktorá je primárne určená pre mobilné telefóny a MIDP 2.0 profilom. Určité časti aplikácie boli vyvíjané pomocou vývojového kitu pre zariadenia s operačným systémom Symbian S60 verzie 3. Sieťové pripojenie a praktické využitie bolo testované na mobilnom telefóne Nokia N95.

V nasledujúcich kapitolách sa práca zaoberá technológiou Java a jej mikroedíciou J2ME a to obecným popisom a tiež voľbou vývojového prostredia.

1. TECHNOLÓGIA JAVA A J2ME

Java je objektovo orientovaný programovací jazyk, ktorý vyvinula firma Sun Microsystems a predstavila 23.5.1995. Java je jedným z najpoužívanejších programovacích jazykov na svete. Vďaka svojej prenositeľnosti je používaný pre programy, ktoré majú pracovať na rôznych systémoch počínajúc čipovými kartami (platforma JavaCard), cez mobilné telefóny a rôzne zabudované zariadenia (platforma Java ME), aplikácie pre desktopové počítače (platforma Java SE) až po rozsiahle distribuované systémy, pracujúce na množstve spolupracujúcich počítačov rozmiestnených po celom svete (platforma Java EE).

1.1. Java Micro Edition

Jazyk Java™ bol pôvodne navrhnutý pre použitie na malých zariadeniach, v spotrebnej elektronike, ako sú televízory, videá, audio súpravy apod. Ale počas svojho vývoja viac a viac smeroval k použitiu na desktopových systémoch. Aplikácie obyčajne potrebujú k svojej prevádzke veľké množstvo knižníc. Ich počet sa stále rozširuje a ďalej sa prispôsobujú potrebám serverových aplikácií. Tento vývoj je určite dobrý, ale Java™ týmto prestala byť vhodnou pre použitie v „malých“ zariadeniach s obmedzenými zdrojmi. Preto bol v roku 1998 vo firme Sun Microsystems zahájený vývoj novej verzie, určenej pre „veľmi malé“ zariadenia so silne obmedzenými zdrojmi. Tieto prístroje majú často len niekoľko desiatok až stoviek kB pamäte. Vznikla špecifikácia jazyka označovaného ako Java ME™ (Java Mobile Edition, skôr tiež označovaná Java2ME™).

J2ME poskytuje riešenie sieťových aplikácií pre malé zariadenia a tiež výrobcom poskytuje možnosť vyvíjať nové aplikácie pre svojich zákazníkov. Veľkou výhodou J2ME je jej kompatibilita s rôznymi prostrediami. V princípe pohľadu J2ME poskytuje nasledovné komponenty:

- séria javovských virtuálnych strojov, každý sa uplatňuje na inom type zariadenia
- skupina knižníc a API, ktoré sú spustiteľné na každom virtuálnom stroji
- nástroje pre vývoj a nastavenie zariadení

Prvé dve komponenty tvoria pracovné prostredie Javy. Jeho centrum tvorí javovský virtuálny stroj, ktorý pracuje na hostiteľskom operačnom systéme. Nad ním je špecifická konfigurácia J2ME zložená z knižníc. Tie zaisťujú základné funkcie vychádzajúce zo zdrojov daného zariadenia. Vrchol konfigurácie tvoria jeden či viac profilov.[4]

Obr. 1.1. Prehľad architektúry pracovného prostredia J2ME

1.2. Konfigurácia

Konfigurácia definuje množinu základných programových knižníc, ktoré sú ovplyvnené technickými zdrojmi (veľkosť ROM a RAM pamäte, typ a frekvencia procesoru, zdroj napájania, typ sieťového prepojenia) daného zariadenia. Teda zariadenie s podobnými technickými vlastnosťami disponujú rovnakou konfiguráciou.

Konfigurácia špecifikuje tri základné elementy:

- Množinu podporovaných rysov programovacieho jazyka Java
- Vlastnosti virtuálneho stroja Javy
- Podporované javovské knižnice a aplikačné programové rozhrania (API)

V súčasnej dobe existujú dva typy konfigurácií. Jedna je určená pre nízko úrovňové zariadenia, kde spadá väčšina mobilných telefónov. Tejto konfigurácií sa hovorí CLDC (Connected Limited Device Configuration). Druhá konfigurácia sa nazýva CDC (Connected Device Configuration) a je určená pre inteligentnejšie zariadenia, ako sú napríklad PDA, navigačné zariadenia pre automobily a pod.

Pri tvorbe aplikácie prehrávača bola uvažovaná konfigurácia CLDC, pretože sa jedná o konfiguráciu, ktorá sa v prevažnej väčšine používa pri mobilných telefónoch.

a) CDC

Konfigurácia CDC sa vyskytuje u výkonnejších zariadení (inteligentné PDA, komunikátory, digitálne Set-Top-Boxy). CDC podporuje všetky vlastnosti virtuálneho stroja Java 2 a preto je tu k dispozícií väčšina vlastností knižnice J2SE.

Virtuálny stroj pre konfiguráciu CDC sa nazýva CVM (Compact Virtual Machine). Pre programátora aplikácií určených pre mobilné telefóny nemá súčasnej dobe veľký význam.

Požiadavky na zdroje pre CDC sú nasledujúce:

- Zariadenie je riadené 32-bitovým mikroprocesorom
- Prostredie pre beh Javy zaberá 2MB RAM a 2,5 MB ROM.
- Zariadenie je pripojiteľné k niektorému typu siete.

b) CLDC

Konfigurácia CLDC je z pohľadu programátora mobilných aplikácií zaujímavejšia, pretože sa vyskytuje v mobilných telefónoch. Ďalej ju môžeme nájsť napríklad v pageroch a starších PDA zariadeniach. Konfigurácia CLDC je teda zameraná na zariadenia, ktoré sú výkonnostne na slabšej úrovni než zariadenia s konfiguráciou CDC. Zariadenia s touto konfiguráciou disponujú virtuálnym strojom, ktorý má veľkosť rádovo stovky kilobajtov. Tento virtuálny stroj sa nazýva KVM (Kilo Virtual Machine).

Typické vlastnosti pre CLDC konfiguráciu sú nasledujúce:

- Zariadenie je riadené 16-bitovým alebo 32-bitovým procesorom a minimálnou taktovacou frekvenciou 25MHz.
- Prostredie pre beh Javy potrebuje minimálne 32KB RAM a 160 KB ROM (bežne je celková pamäť 160 KB až 512 KB).
- Zariadenie môže mať obmedzený zdroj energie (batéria).
- Zariadenie je prepojiteľné s niektorým typom siete, pomalšie (typický dátový tok 9,6 Kbps), prerušiteľné spojenie.

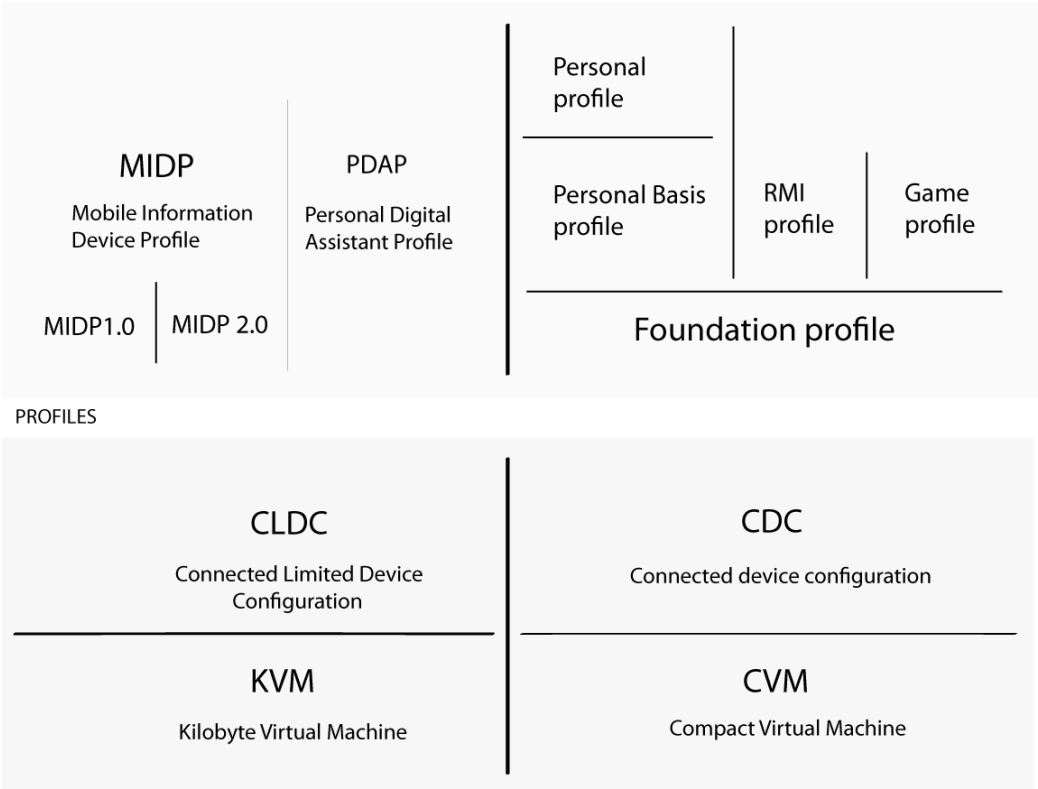
S danými obmedzeniami poskytuje v súčasnosti CLDC zariadeniam tieto funkcie:

- Súbor základných knižníc Javy (`java.lang, java.util`)
- Základný vstup / výstup (`java.io`)
- Základnú podporu sietí (`javax.microedition.io`)
- Zabezpečenie na úrovni virtuálneho stroja, zabezpečenie na úrovni aplikácie (aplikácie môžu volať jedine triedy podporované samotným zariadením – sandbox)[1]

1.3. Profily

Profil je ďalšou vrstvou pracovného prostredia J2ME, ktorá doplňuje programovacie knižnice využívajúce príbuzných funkcií v podobných zariadeniach. Jedná sa o rozlíšenie rôznych produktov rovnakej úrovne (napríklad mobilné telefóny disponujú jednou konfiguráciou – CLDC, ale líšia sa v profiloch).

Pod záštitou Java Community Process bolo vytvorených niekoľko profilov, ktoré sú špecifické nielen pre produkty rôznej úrovne, ale aj pre určité množiny zariadení na rovnakej úrovni (pre danú úroveň existuje viacero profilov). Profily obsiahnuté v J2ME viď Obr. 1.2.



Obr. 1.2. Profily J2ME

Z profilov J2ME sa pre mobilné telefóny uvažujú profily MIDP 1.0 a MIDP 2.0. MIDP 1.0 je na použitie univerzálnejší, pretože je podporovaný každým mobilným telefónom. Naopak pri MIDP 2.0 to neplatí. Preto je voľba profilu pri tvorbe aplikácie veľmi dôležitá z dôvodu kompatibility. Súčasný trh s mobilnými telefónmi však ponúka väčšinovú podporu profilu MIDP 2.0.

Z dôvodu využitia rôznych pokročilých techník v aplikácií, ktoré podporujú len mobilné telefóny novej generácie, ako napríklad bezdrôtové pripojenie k sieti a podpora video kodekov, bola aplikácia navrhnutá pre mobilné telefóny s podporou profilu MIDP 2.0

a) MIDP 1.0

MIDP 1.0 je základným profilom pre mobilné telefóny, ktorý doplňuje programové knižnice pre CLDC konfiguráciu. Pokiaľ je žiaduca maximálna kompatibilita medzi mobilnými zariadeniami je nutné sa držať tohto profilu.

MIDP 1.0 definuje nasledujúce požiadavky na dané zariadenie:

- Minimálny rozmer obrazovky 96x54 px
- Minimálne 1 bitová farebná hĺbka
- Klávesnica pre jednu alebo dve ruky, prípadne dotyková obrazovka
- 128 KB stálej pamäti pre MIDP komponenty
- 8 KB pamäti pre dlhodobo ukladané dáta
- 32 KB nestálej pamäti pre beh Javy (JRE, memory heap)
- Obojsmerná bezdrôtová komunikácia

MIDP 1.0 špecifikuje nasledujúce aplikačné rozhranie:

- Správa priebehu aplikácií (`javax.microedition.midlet`)
- Užívateľské rozhranie (`javax.microedition.lcdui`)
- Perzistentné uchovanie dát – RMS (`javax.microedition.rms`)
- Obecné sieťové pripojenie (`javax.microedition.io`)

b) MIDP 2.0

Profil MIDP 2.0 je rozšírením profilu MIDP 1.0 a je spätne kompatibilný. To znamená, že mobilný telefón s MIDP 2.0 bezproblémovo spustí aplikáciu napísanú pre MIDP 1.0.

Keďže obsahuje takmer dvakrát toľko tried než MIDP 1.0, vzrastá požiadavka na pamäť. Jedná sa o vzrast 128 KB RAM (celkom 256 KB) a o 96 KB viac pre beh JRE (celkom 128 KB).

Medzi významné rozšírenia programového vybavenia patrí:

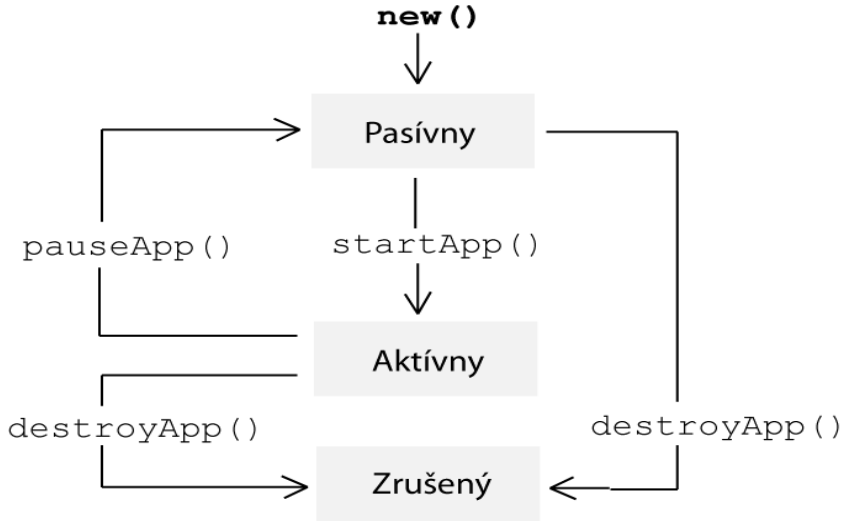
- Pribudli rozhrania pre ďalšie sieťové protokoly (http, datagram, soket a iné)
- Herné užívateľské rozhranie a ovládanie zvuku štandardným rozhraním

1.4. MIDlet

Pri tvorbe mobilnej aplikácie je nutné zdediť abstraktnú triedu `MIDlet`, ktorá sa nachádza v balíčku `javax.microedition.midlet`. Preto sa často mobilné aplikácie nazývajú midlety.

Pri spustení aplikácie je trieda, ktorá dedí triedu MIDLet, vytvorená zavolaním svojho verejného konštruktoru. Stavy, v ktorých sa môže midlet nachádzať, a prechod medzi týmito stavmi, ukazuje Obr. 1.3. Jedná sa o stav pasívny, aktívny a zrušený. Beh aplikácie a prechod medzi stavmi riadi *Aplikačný manažér*.

Pri volaní metódy `startApp()` MIDlet inicializuje všetky objekty potrebné pre aktívny stav a nastaví odpovedajúcu obrazovku. Metóda `pauseApp()` preruší všetky aktívne vlákna. Volaním metódy `destroyApp()` by sa mali uvoľniť, alebo zavrieť všetky zdroje.



Obr. 1.3. Životný cyklus MIDlet

Príklad kostry MIDletu:

```
import javax.microedition.midlet.*;

public class newMIDlet extends MIDlet {

      public newMIDlet () { //konstruktor }

      public void startApp() { //aktivacia midletu }

      public void pauseApp() { //prerusenie midletu }

      public void destroyApp (Boolean unconditional ) {
//zrusenie midletu) }
}
```

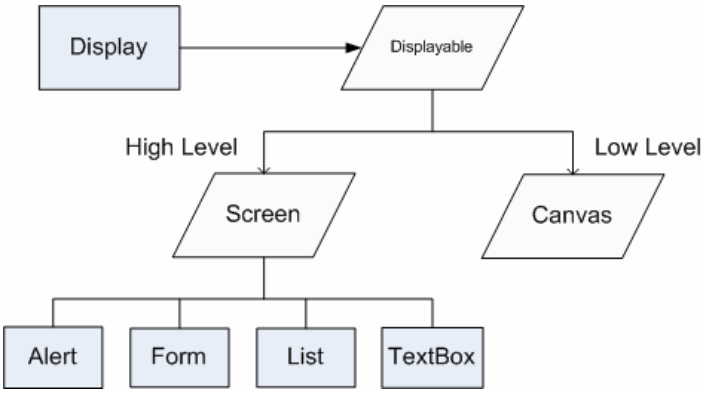
1.4.2. MIDlet užívateľské rozhranie

a) High Level a Low Level

MIDlety sú určené na prenosné zariadenia širokého druhu s rôznymi možnosťami vstupu údajov a s obrazovkami odlišných typov. Je potrebné vytvoriť rozhranie pre jednoduché pagery a mobilné telefóny, ktoré umožňujú zobraziť jednoduché symboly. Zároveň pre smartphony a PDA, ktoré majú polohovacie zariadenie a klávesnicu, z čoho vyplýva oveľa komfortnejší vstup údajov a teda aj prípadné ovládanie zariadenia. Samozrejme zobrazovacia časť je prispôsobená tejto úrovni a tým je umožnené vytvoriť kvalitnejšie grafické rozhranie, na ktoré sa kladú vyššie požiadavky. Vývojári tohto rozhrania zvažovali všetky tieto aspekty. A rozhodovali sa nasledovne, spraviť zjednodušené AWT alebo Swing zo štandardných knižníc J2SE. Bohužiaľ táto možnosť sa zamietla najmä z dôvodu výpočtovej náročnosti výpočtovej náročnosti a náročnosti na zobrazovanie GUI. V dobe vytvorenia J2ME bolo len veľmi málo zariadení, ktoré by dokázali adekvátne využiť túto úroveň. Na druhej strane ale bolo množstvo jednoduchých prenosných zariadení s malým, väčšinou čiernobielym displejom a preto bolo potrebné vytvoriť jednoduché a užitočné grafické užívateľské rozhranie, ktoré nesie označenie High Level.

High Level sa vykresľuje na obrazovku (Screen). Sú definované ovládacie prvky ako napríklad textové pole, dátum, výberová skupina, atď. Výhodou je, že tieto prvky sa pomerne ľahko implementujú. Nevýhodou je, že sa ťažko prispôsobujú konkrétnejším požiadavkám. Toto API sa používa najmä v prostredí kancelárskych aplikácii. Podrobnejší popis je v kapitole 1.4.3.

Pre vytvorenie vlastného, prispôsobeného prostredia sa využíva druhá možnosť, označuje sa ako Low Level. V tomto API sa kreslí priamo na plátno (Canvas). Využívajú sa ti základné obrazce ako priamka, štvorec, kružnica. Z týchto útvarov je potom možné vytvoriť komplikovanejšie obrazce. Prípadne využívať externé obrázky a implementovať do plátna. Vytváranie takto prívetivého prostredia je však dosť časovo náročné. Výhodou je však možnosť vytvoriť akúkoľvek obrazovku, napríklad farebnú a grafickú. Je možné ju použiť na zobrazovanie čohokoľvek v rámci možností použitého zariadenia. Viac v kapitole 1.4.4.[2]



Obr. 1.4. Rozdelenie High Level a Low Level

1.4.3. High Level

Implementácia aplikácie v High Level režime je veľmi prívetivá. Vykresľuje sa na triedu Screen (obrazovku) zdedenú od abstraktnej triedy `Displayable`. Sú definované ďalšie triedy zdedené od Screen, ktoré slúžia na výpis určitých informácií.

- `List` Vypisuje zoznam, môže byť s výberom položky
- `TextBox` Výpis jednoduchého textu na obrazovku
- `Alert` Upozornenie. Vykreslí sa vždy na spodnú časť obrazovky. Používa sa v prípade oznamovania chyby a pod.
- `Form` Formulár. Najširšia trieda, lebo umožňuje vkladanie prvkov typu Item. Medzi najznámejšie patria:
    (a) ChoiceGroup (výberová skupina)
    (b) StringItem(reťazec)
    (c) DateField(položka dátumu)
    (d) TextItem(text)
    (e) ImageItem(obrázok)
    (f) Spacer(vytváranie medzier, len od MIDP 2.0)

Ich výhodou je, že sú už vopred nadefinované a majú určitú formu. Takže programátor sa môže venovať viac využitiu ich funkcií. Na druhú stranu sú však málo graficky prispôsobivé.

1.4.4. Low Level

V tomto režime sa kreslí na plátno (Canvas), čo dáva programátorovi do rúk nespútanú voľnosť vytvoriť si prostredie podľa vlastných predstáv a požiadaviek. Tento mód sa väčšinou používa pri tvorbe hier a grafických aplikácií, kde je potrebné vytvorenie obrázkov, pohyblivých elementov, prípadne videa. Vývojár má nad všetkým plnú moc a kontrolu. Bohužiaľ z toho vyplýva, že všetko si vývojár musí kontrolovať a obstarávať. Sú k dispozícií základné grafické elementy ako sú čiara, kružnica, obdĺžnik, vykreslenie reťazca, obrázku atď. Dajú sa nastavovať ich parametre ako sú veľkosť, dĺžka, šírka, výška a tiež farba a mnoho iného. Umožňuje zachytávať návratové hodnoty podporovaných kláves a ďalej ich spracovať. Vhodné napríklad na rozšírené ovládanie, kde je potrebné čítať z celej klávesnice zariadenia. Vytvoriť v tomto režime použiteľnú aplikáciu je časovo dosť náročné.

1.5. JAR a JAD súbory

Predtým než sa programátor bude snažiť spustiť MIDlet aplikáciu na mobilnom telefóne, bude z nej musieť vytvoriť JAR a JAD súbory. Podmienkou je, aby JAR súbor vo vnútri obsahoval aj tzv. manifest súbor. Podrobnejší popis jednotlivých súborov je v nasledujúcich kapitolách.

JAD (Java Application Descriptor) je štandardný textový súbor s príponou *.jad. Je to externý súbor pre JAR súbor, podobne ako manifest. Pozostáva z niekoľkých atribútov, ktoré opisujú MIDlet v MIDlet suite. JAD súbor sa používa pre manažérsky software k nastaveniu MIDletov.

Medzi povinné atribúty JAD súboru patrí napríklad atribút s veľkosťou JAR súboru v bajtoch, URL, ktorej je možné JAR stiahnuť, názov MIDlet, verzia, formát a iné.

JAR má najdôležitejšiu úlohu udržiavať skompilované triedy celého projektu a to v adresárovej štruktúre podľa balíčkov navrhnutých v projekte. Je vhodné pripomenúť, že JAR súbor je komprimovaný algoritmom ZIP. Dôležitou súčasťou je manifest súbor, ktorého cestou je konštrukcia META-INF/MANIFEST.MF. Súbor MANIFEST.MF obsahuje totožné položky ako JAD súbor. Napríklad URL s adresou, názov MIDlet a ďalšie.[2]

1.6. Mobile Media API

Vkladanie multimédií do mobilných aplikácií je ďalším krokom v evolúcií MIDletov. Sun Microsystem zistil už dávnejšie, že audio a video je budúcnosť mobilných zariadení a predstavil Mobile Media API (MMAPI) ako súčasť Java Specification Request (JSR) 135. Najväčšie požiadavky MMAPI špecifikácie boli zabezpečiť kompatibilitu s malými mobilnými zariadeniami a zároveň vytvoriť špecifikáciu, ktorá by bola využiteľná v budúcnosti pre pravdepodobne viac inteligentné zariadenia. Tento účel sa MMAPI podarilo splniť.

MMAPI je doplnkový API, ktorý vývojári používajú na vloženie pokročilých multimediálnych funkcií do zariadenia podporujúceho J2ME. Pokiaľ sú používané funkcie na prehrávanie multimédií, napríklad audia v profile MIDP 2.0, využívajú určitú podskupinu MMAPI.

Mobile Media API umožňuje vytvárať aplikácie, ktoré budú prehrávať niekoľko formátov audio a video súborov zo siete, budú disponovať úložným priestorom (`record store`), budú mať pokročilú kontrolu nad prehrávaním multimediálnych súborov, zachytávanie audia a videa, zachytávanie snímok, prehrávanie MIDI súborov, generovanie a prehrávanie zvukových tónov, prehrávanie audio streamu zo siete a ešte veľa ďalšieho.

V súčasnosti však existuje mnoho mobilných telefónov, ktoré podporujú niektoré základné protokoly a formáty. Napríklad, väčšina dovoľuje prístup do siete cez http a prehrávanie WAV súborov a MP3 súborov, prípadne videa. Ak zariadenie podporuje niektoré z týchto typov, MMAPI môže riadiť určité ovládacie prvky pomocou funkcií týchto formátov, za účelom vytvoriť jednotnosť ovládacích prvkov pre rôzne mobilné zariadenia.

Síce bol MMAPI vyvinutý pre mobilné zariadenia, ale jeho prevedenie je všeobecné, takže z neho môže čerpať akékoľvek zariadenie, podporujúce Javu. V nasledujúcom zozname je niekoľko pravidiel API:

- API pre malé zariadenia: Pretože hlavným cieľom tejto API sú mobilné zariadenia, ktoré sú často obmedzené dostupnou pamäťou, MMAPI musí podporovať prehrávanie multimédií v dostupnej pamäti. A to z dôvodu, ako bolo už spomenuté, že mobilné telefóny podporujú CLDC konfiguráciu, ktorej dostupná pamäť je v rozsahu 128KB až 512KB. Táto pamäť je k dispozícií pre virtuálny stroj, softwarové knižnice, vytvorený MIDlet a MMAPI.

- Podpora niekoľkých multimediálnych typov: Definíciou jadra API ako súboru rozhraní je, že MMAPI nemá jasnú definíciu protokolu ani formátu. Zariadenia sú od výroby opatrené ich vlastnou implementáciou MMAPI a rozhraní, ktoré podporujú multimediálne funkcie ich vlastných zariadení. To umožňuje široký rozsah protokolov a formátov, ktoré budú podporované priamo v zariadení, bez nejakej pevnej implementácie protokolov a formátov priamo do samotného API. To robí z MMAPI nesmierne dostupný nástroj pre výrobcov zariadení, ktorý vyvíjajú stále nové formáty a implementujú ich do svojich výrobkov.[3]

2. MULIMEDIÁLNY PREHRÁVAČ

V tejto kapitole sa pojednáva o návrhu a riešení aplikácie Multimediálny prehrávač pre mobilné telefóny. Popis jednotlivých častí je riešené z hľadiska programátora. Užívateľská časť, popis aplikácie je uvedený v kapitole 2.11.

2.1. Výber cieľovej skupiny

Pred začiatkom vývoja akejkoľvek aplikácie pre mobilné zariadenie je potrebné prehodnotiť hardwarovú náročnosť a zvoliť si cieľovú skupinu mobilných telefónov. To znamená, ako už bolo spomenuté, zvoliť vhodnú konfiguráciu, profil, prispôsobiť aplikáciu rozlíšeniu displeja a iné.

Navrhovaný prehrávač je zameraný na mobilné telefóny, ako to vyplýva zo zadania práce, takže konfigurácia je volená CLCD. Z profilov MIDP 1.0 a MIDP 2.0 aplikácia využíva vlastnosti, ktoré ponúka profil MIDP 2.0, keďže sa jedná o prehrávanie multimediálneho obsahu. Ďalším aspektom je rozlíšenie displeja, ktoré je zvolené 320 x 240 px, čo je štandardné rozlíšenie v súčasnosti obľúbených smartphonov od Nokie, Apple a iných výrobcov, ktoré sú cieľovou skupinou tejto aplikácie.

2.2. Vytvorenie projektu

Po zvolení cieľovej skupiny, kedy je teda známe čo sa bude pri projekte využívať, prichádza na rad vytvorenie projektu s vybranými parametrami.

2.2.1. NetBeans IDE

Prvým krokom je zvoliť si pracovný nástroj. Pre vývoj mobilnej aplikácie je dôležitá podpora platformy Java mobile vývojového nástroja. Jeden z najrozšírenejších nástrojov je nepochybne vývojové prostredie NetBeans IDE. Je to Open Source projekt, ktorý je možné bezplatne používať ku komerčným aj nekomerčným účelom a jej zakladateľom je priamo Sun Microsystems. Vývojové prostredie NetBeans IDE je nástroj, pomocou ktorého programátor môže písať, prekladať, ladiť a výsledné aplikácie tiež distribuovať.

2.2.2. Voľba vývojového kitu

Ďalším dôležitým bodom je aj voľba emulátoru. Emulátor je spojka medzi vývojovým prostredím a mobilným telefónom. Slúži k ladeniu vyvíjanej aplikácie, poskytuje náhľad a testovanie aplikácie na PC pred inštaláciou na fyzické zariadenie. Programátor má možnosť zvoliť medzi univerzálnym emulátorom, ktorý poskytuje Sun Java Wireless Toolkit, alebo zvoliť od niektorého z výrobcov mobilných telefónov. Pre aplikáciu multimediálneho prehrávača sú zvolené dve verzie emulátorov Nokia S60 3rd Feature Pack 1 a 2 SDK pre mobilné telefóny Nokia na platforme Symbian. Vďaka týmto SDK je možný vývoj a implementácia MIDP aplikácie pre platformu S60 na PC. Spolu s vývojovým rozhraním IDE, SDK poskytuje potrebné súčasti pre funkčnosť, vrátane ukážkových programov a dokumentácie potrebnej na vývoj nových midletov pre S60 platformu. Obsahuje softwarové knižnice, rozhranie pre programovanie aplikácií API, emulátor s platformou *S60 3rd Edition*, ktorý podporuje vývojové prostredie NetBeans 5.0 a CLDC 1.1 a MIDP 2.0. Poskytuje grafické rozhranie reálneho telefónu so základnou funkcionalitou telefónu na testovanie vyvíjaných aplikácií. Užívateľské rozhranie emulátoru je navrhnuté pre jednoduchšie použitie. Z užívateľského hľadiska, pravdepodobne najdôležitejšia časť platformy je práve toto rozhranie, ako je veľká farebná obrazovka, prípadne rôznorodé funkčné tlačidlá. Princíp jednoduchšieho použitia a veľkej farebnej obrazovky umožňuje vývojárom aplikácií prezentovať atraktívny obsah a poskytnúť jednoduchú navigáciu.

Obr. 2.1. Emulátory S60 3rd Edition FP1 a FP2

2.3. Úvod do aplikácie

Telo programu tvorí hlavná trieda `MobilePlayer`, ktorá obsahuje povinné metódy `startApp()`, `pauseApp()` a `destroyApp()`. Ďalej sa tu nachádza inicializácia atribútov, konštruktor a ostatné, vytvorené metódy. V programe sú tiež naimportované knižnice, ktoré poskytujú rôzne funkcie ako prehrávanie multimediálnych súborov ako audio a video, prístup k súborom, vstupno-výstupné funkcie, sieťové pripojenie a iné. Súčasťou sú aj triedy na vytvorenie vlákien a časovača. Program ďalej obsahuje triedu pre zavedenie grafického rozhrania `ButtonCanvas`, triedu pre http pripojenie a načítavanie do vyrovnávacích pamätí `HttpConn`. Ďalej triedu `URLaddress`, ktorá vytvára formulár na manažovanie adries, `VideoStream` triedu na zachytávanie obrazu kamery, príjmu dát a následného prehratia a nakoniec triedu `RecieveVideo`, ktorá zachytené video odosiela. Bližšie popísané triedy a ich metódy a funkcie sú v kap.

Spomínaná skladba tried a metód má za úlohu prehrať hudobné súbory a videá formátu MPEG-4. Umožňuje tiež prehrávať audio v reálnom čase zo vzdialeného servera, audio stream. Tiež zachytávať obraz zo vstavanej kamery, ukladať ho a následne odoslať na zariadenie s rovnakou aplikáciou v prijímacom móde.

2.4. Základné metódy a triedy aplikácie

2.4.1. Trieda MobilePlayer

Je to hlavná trieda, ktorá riadi cyklus celej aplikácie. Okrem povinných metód obsahuje aj väčšinu metód vytvárajúcich časť aplikácie.

Vykonaním štartovacej metódy `startApp()` sa inicializuje displej a zobrazí sa splashscreen s úvodným obrázkom a názvom aplikácie. Naopak metóda `destroyApp()` má na starosti ukončenie aplikácie.

a) Metóda getDisplay

Metóda má návratovú hodnotu typu `Display`. Touto metódou sa vykonáva spomínaná inicializácia displeja.

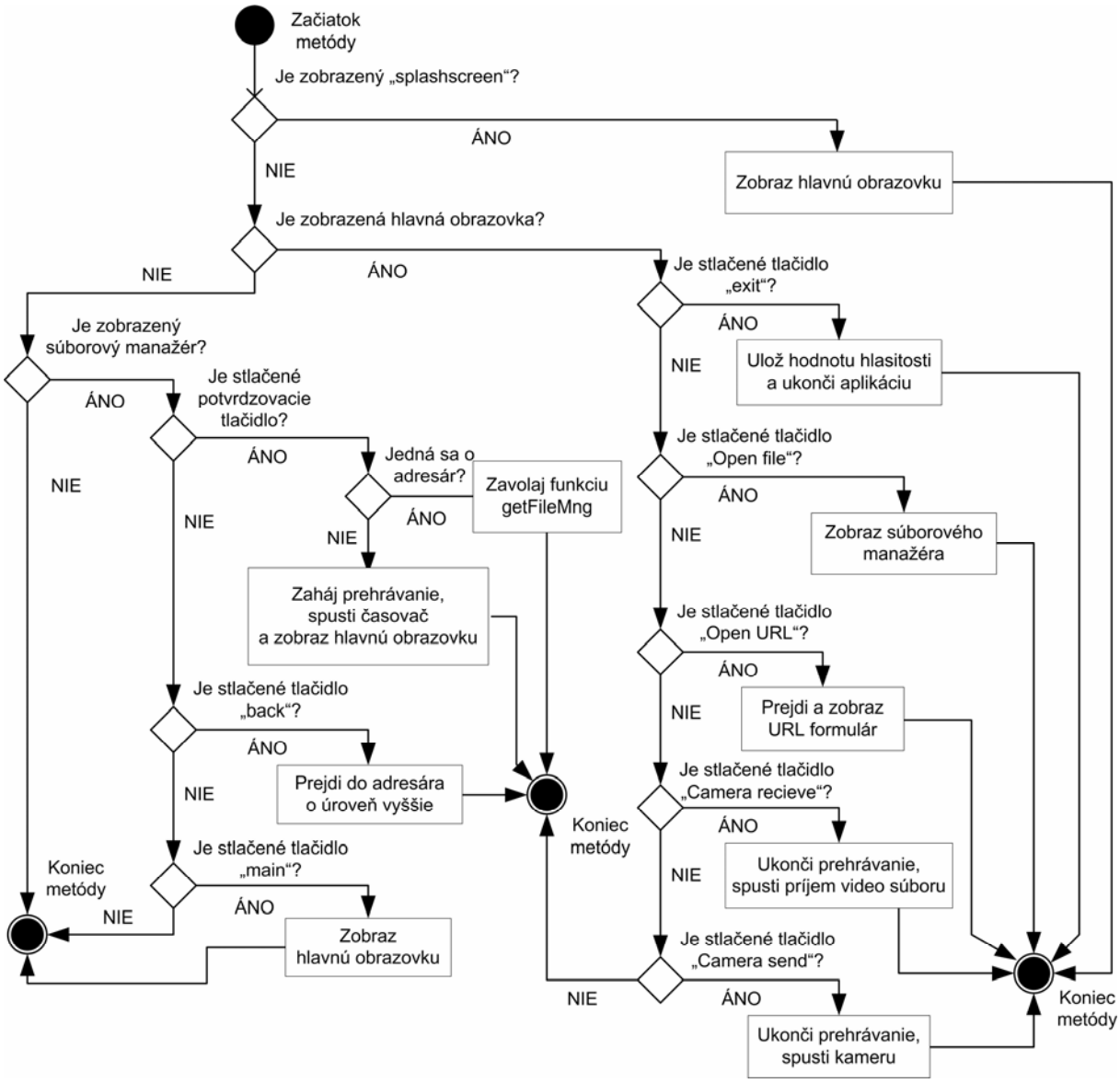
b) Metóda commandAction

Metóda zaradená pod hlavnú triedu `MobilePlayer` zachytáva stlačenie tlačidla a vykonáva funkcie podľa typu tlačidla. `ComandAction` metóda pracuje s dvoma vstupnými parametrami. Jedna z nich obsahuje hodnotu práve stlačenej klávesy a druhá hodnotu

aktuálneho zobrazenia. To znamená, že každá obrazovka má vlastné menu. Táto metóda vyberá medzi hlavnou obrazovkou, súborovým manažérom a tiež zabezpečuje akciu, ktorá nastáva po uplynutí časového intervalu úvodnej obrazovky (splashscreenu). Akciou sa rozumie zobrazenie hlavnej obrazovky na displeji. Hlavná obrazovka je ešte rozdelená podľa toho, ktorá časť programu sa vykonáva. Obsahuje základné zobrazenie, prehrávanie audio súboru, prehrávanie video súboru, prehrávanie audia v reálnom čase, prijímaciu časť videa a časť zachytávania obrazu z kamery. O ostatné časti sa starajú metódy `commandAction` v iných triedach.

Na výber z tlačidiel sú jednotlivé položky v možnostiach „Options“ a ukončenie aplikácie tlačidlom s označením „Exit“. Po stlačení prvej položky v možnostiach v hlavnej obrazovke, „Open file“, sa prepne zobrazenie do súborového manažéra. Druhou v poradí je položka „Open URL“, ktorá zobrazí výber IP adresy, z ktorej bude prehrávaný obsah. Po výbere „Camera Recieve“ sa zastaví každý z prehrávačov a volá sa metóda na príjem video súboru. No a poslednou položkou v možnostiach je „Camera Send“. Po výbere sa opäť zastaví prípadný bežiaci prehrávač a volá sa metóda ktorá zachytáva obraz z kamery.

Metóda `commandAction` je pre lepšie pochopenie zobrazená na vývojovom diagrame (Obr. 2.2).



Obr. 2.2. Vývojový diagram metódy commandAction

Za zmienku ešte stoja príkazy, ktoré sú vykonávané, ak je zobrazený súborový manažér. Po stlačení potvrdzovacieho tlačidla sa rozhodovacou metódou zistí, či sa jedná o adresár, alebo súbor.

Pokiaľ je zvolenou položkou adresár, zapíše sa do atribútu *currDirName* jeho názov a zavolá sa funkcia getFileMng. Ak uskutočnený výber odkazuje na súbor, do atribútu *song_URL* sa vloží kompletná cesta k súboru. Podľa koncovky súboru sa rozhodne, či sa jedná o formát „mp3“ a zavolá sa metóda prehrávania audio súborov `PlayMusic` a časovač `getTime` a zobrazí sa hlavná obrazovka, alebo ide o video formát „mp4“, kedy sa zavolá metóda prehrávania video súborov a taktiež zobrazenie prejde do hlavnej obrazovky.

c) Metóda getCanvas

`GetCanvas` je metóda, ktorá vytvorí objekt triedy `ButtonCanvas`. Priradí jej tlačidlá a aktivuje odchytávanie stlačenia. Prakticky to znamená, že sa vytvorí hlavná obrazovka, tlačidlám sa priradia funkcie, vytvorí sa menu so súborom možností aplikácie a ukončením aplikácie. Vstupnými parametrami funkcie `commandAction` budú položky tohto menu. Menu tvoria položky:

- Open file – *Fopen* (otvorenie súborového manažéra)
- Open URL – *URLopen* (otvorenie IP manažéra)
- Camera recieve – *Vrecieve* (príjem videa z druhého zariadenia)
- Camera send – *Vsend* (zachytávanie videa pre odoslanie)
- Exit (ukončenie aplikácie)

d) Metóda stopAllPlay

Pri spustení ktoréhokoľvek z prehrávačov sa nastaví príznak konkrétne spúšťaného prehrávača. Jedná sa o atribút typu boolean. Sú to atribúty hudobného prehrávača, video prehrávača a prehrávača audio streamu.

Po zavolaní tejto metódy sa podľa spomínaného príznaku zistí, ktorý z prehrávačov bol aktívny. Následne uloží metódou `SetVolLevel` jeho hodnotu hlasitosti a prehrávač ukončí a prekreslí hlavnú obrazovku (Canvas). V prípade zložitejšieho prehrávača audio streamu sa nastaví aktuálny čas prehrávania na celkový čas skladby, to znamená, že sa skončí prehrávanie, zaháji ukončenie vlákna vyrovnávacej pamäte a následne po ukončení sa neguje príznak a prekreslí hlavná obrazovka do základného režimu.

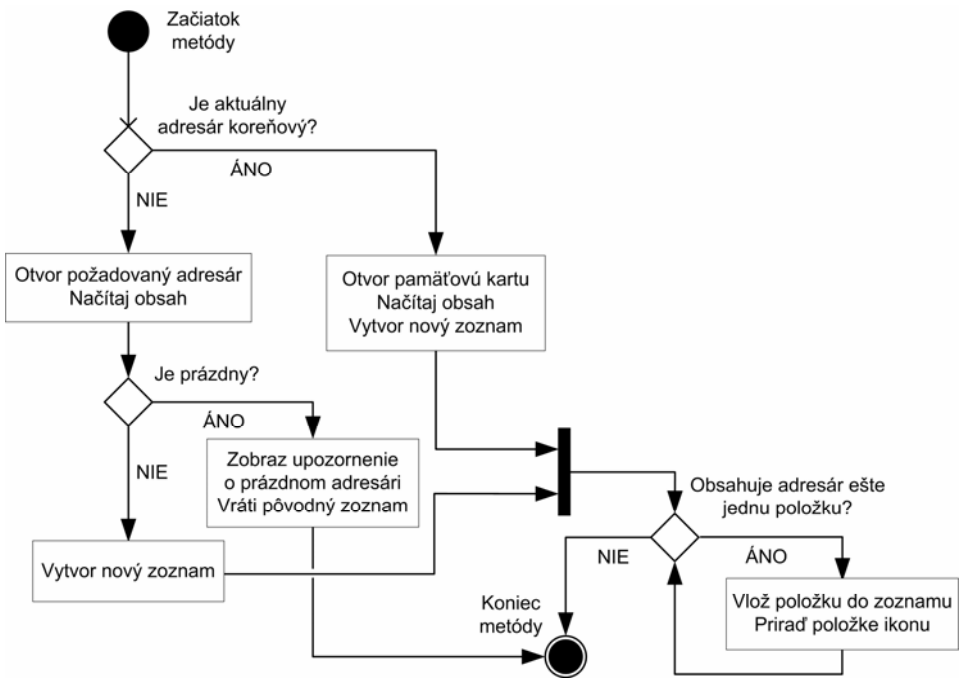
e) Metóda getFileMng

Táto metóda má za úlohu vytvoriť súborový manažér. Teda vypísať zoznam adresárov a súborov a umožniť pohyb v pamäti mobilného telefónu. Keďže aplikácia je určená k mobilným telefónom na platforme Symbian, predpokladá sa, že bude obsahovať aj pamäťovú kartu. Táto je potom vo väčšine prípadov zdrojom multimediálnych dát. Takže aj súborový manažér má koreňový adresár nastavený na pamäťovú kartu. V prípade, že pamäťová karta nie je vložená, zobrazí sa varovanie o chýbajúcom zdroji dát, súborový manažér sa zavrie a na displeji sa nastaví naspäť hlavná obrazovka.

Metóda pred výpisom obsahu karty zistí, či sa jedná o koreňový adresár alebo nie. Pokiaľ sa jedná už o adresár na pamäťovej karte, zisťuje, či nie je prázdny, prípadne

neprístupný. V prípade, že je prázdny, je zobrazené upozornenie informujúce o tom, že je požadovaný adresár je prázdny. Ak je problémom chyba, ktorá neumožní prístup k obsahu, adresár sa jednoducho neotvorí. V opačnom prípade, ak nenastal žiadny problém, metóda načíta obsah zložky a vypíše ich zoznam. Ku každej položke je pridaná ikona, podľa toho, či sa jedná o adresár, súbor s koncovkou „mp3“, „mp4“, alebo iný, nepodporovaný typ súboru. V súborovom manažéri je vytvorené menu obsahujúce položky:

- Main (vráti sa do hlavnej obrazovky)
- Back (postúpi do adresára o úroveň vyššie)



Obr. 2.3. Vývojový diagram metódy getFileMng

f) Metóda PlayMusic

Jedná sa o metódu, ktorá vytvorí objekt triedy `PlayVlakno`. Táto trieda obsahuje algoritmus, ktorý sa stará o vytvorenie prehrávača a samotné prehrávanie. Viac v kapitole 2.5. V ďalšom kroku sa vytvorí nové vlákno, v ktorom bude prehrávanie bežať.

Po zavolaní metóda zistí pomocou ďalšej metódy `stopAllPlay`, či sa momentálne nevykonáva prehrávanie súboru. V prípade, že áno, prehrávanie zastaví, zavrie vytvorenú funkciu prehrávača, čím sa ukončí aj pred tým vytvorené vlákno. Až keď je prehrávanie zastavené, alebo prehrávač ešte nebol vytvorený vykonajú sa vyššie spomínané úkony na vytvorenie vlákna.

g) Metóda PlayVideo

Metóda podobná metóde `PlayMusic`. Opäť sa po ukončení prípadného bežiaceho prehrávača vytvorí objekt triedy, tento krát `VideoThread`. Tá vytvorí video prehrávač, získa kontrolu zvuku a videa a spustí prehrávanie na plátne *canvas*. Viac v kapitole 2.4.2. Následne sa z objektu triedy vytvorí nové vlákno, ktoré bude prehrávanie obsluhovať.

h) Metóda GetVolLevel

Po vytvorení prehrávača sa nastaví aj jeho hodnota hlasitosti. Aby táto hodnota nebola po každom spustení rovnaká, bez možnosti zásahu užívateľa, je po každom spustení načítaná posledná uložená hodnota hlasitosti. Načítanie tejto hlasitosti je práve funkciou metódy `GetVolLevel`.

V prvom kroku sa nastaví štandardne hlasitosť na hodnotu 50. Ak bola aplikácia spustená prvý krát po inštalácii, alebo nebola korektne ukončená, metóda vracia práve túto hodnotu. V opačnom prípade sa program pokúsi načítať naposledy uloženú hodnotu a túto následne vráti.

i) Metóda SetVolLevel

Pred korektným ukončením aplikácie tlačidlom end sa zavolá metóda `SetVolLevel`. Jej vstupným parametrom je aktuálna hodnota hlasitosti prehrávača. Funkciou tejto metódy je túto hodnotu uložiť do pamäte, aby bola použitá pri ďalšom spustení aplikácie.

j) Metóda getTime

Každý správny prehrávač multimédií by mal obsahovať ukazovateľ odohraného času. Základom tohto je časovač, ktorý vytvára `getTime` metóda.

Metóda vytvorí a spustí časovač a následne mu pridá funkciu, aby v časovom intervale 1000ms prekresľoval hlavnú obrazovku, na ktorej je vypisovaný aktuálny odohraný čas. Týmto sa docieli, že na obrazovke pobeží čas prehrávaného súboru.

k) Metóda getOpenFile

Táto metóda sa volá pri vytváraní prehrávača. Slúži na otvorenie súboru za účelom čítania. Po otvorení, metóda súbor načíta do tzv. „InputStreamu“, inými slovami ho pretypuje a takto načítaný súbor je vrátený k prehrávaniu.

Ak by sa z nejakých príčin súbor nepodarilo otvoriť, na obrazovke sa zobrazí upozornenie, že nie je možné požadovaný súbor otvoriť.

l) Metódy upozornení

V programe sa používajú štyri druhy upozornení. Sú to metódy typu Alert. Sú to:

```
getAlertFolder ()    getAlertRTPStream()    getAlertConnect()
getAlertFormat()     getInfoConnected()    getAlertConnectTo()
getAlertSource()      getInfoRTPplay()      getAlertSended()
getAlertOpenfile()   getAlertRTPcontent()
```

Prvé z nich, getAlertFolder, je volaná v prípade, že požadovaný adresár je prázdny. Jedná sa o upozornenie informatívneho typu a to „Selected folder is EMPTY!“. Dĺžka zobrazenia je nastavená na 2000 ms.

GetAlertFormat upozorňuje na to, že sa prehrávaču sa nepodarilo otvoriť súbor pretože je nepodporovaný. Informatívne hlásenie „Unsupported file FORMAT!“ sa zobrazí po dobu 2000 ms.

Ďalším z použitých upozornení je getAlertSource, ktoré hlásením „The SOURCE is missing!“ informuje o chýbajúcom pamäťovom médiu opäť po dobu 2000ms. Je volané prípade neúspešného prístupu ku koreňovému adresáru pri otváraní súborového manažéra.

Posledné upozornenie getAlertOpenfile sa zobrazí v prípade, keď sa program pokúša otvoriť súbor, ktorý má ochranu proti čítaniu, prípadne je poškodený. Vypísaný je reťazec „Can not OPEN the file!“ štandardne po dobu 2000 ms.

Ostatné metódy upozornení vytvárajú rovnako hlásenia o rôznych chybách a situáciách. Doba vypísania v reťazca je v každom hlásení v rozmedzí 1200 až 2000ms. Výpis a vysvetlenie chybových hlásení je popísaný v užívateľskej časti v kapitole 2.11.2.

m) Metóda EmptyFolder

Jedná sa len o pomocnú metódu, ktorá obsahuje volanie upozornenia `getAlertFolder`, v prípade prístupu do prázdneho adresára a metódy `UpFolder`.

n) Metóda UpFolder

V prípade, že je potrebné vystúpiť v súborovom manažéri o adresár vyššie je volaná metóda `UpFolder`. Z atribútu *currDirName* s cestou aktuálne zobrazeného adresára zistí najbližšiu vyššiu zložku a spomínaný atribút ňou prepíše.

o) Metóda getSplashImg

Ide o vytvorenie obrázka pre úvodnú obrazovku tak, aby bol použiteľný v programovacom jazyku Java. To znamená, že do atribútu typu Image je vložený požadovaný obrázok.

p) Metóda getSplashScreen

Hneď po spustení aplikácie je zobrazená úvodná obrazovka s obrázkom po dobu niekoľkých sekúnd a to pred tým, než je zobrazené hlavné okno. Táto metóda úvodnú obrazovku vytvorí, načíta pripravený obrázok a stanový časový limit zobrazenia.

2.4.2. Trieda ButtonCanvas

`ButtonCanvas` dedí možnosti triedy `Canvas`, na čom je postavená hlavná obrazovka aplikácie. Hlavnou úlohou tejto triedy je vytvoriť kresliace plátno (Canvas). Na tomto plátne budú zobrazované hlavné udalosti aplikácie, ako napríklad odohraný čas, hlasitosť, grafické pozadie, názov skladby atď. Ďalšou funkciou tejto triedy je zachytávanie kláves. Toto je využívané na ovládanie hlasitosti a na spustenie a pozastavenie prehrávania. Táto trieda obsahuje niekoľko nasledujúcich metód.

a) Metóda paint

Metóda `paint` je základnou abstraktnou metódou triedy Jej funkciou je vykresľovanie na plátno. Sú v nej nastavené hodnoty farby textu a grafiky, implementuje obrázok do pozadia, pomocou metódy je vypisovaný text. Ak je potrebné plátno prekresliť v prípade nejakej zmeny v texte, stačí jednoducho zavolať funkciu plátna `repaint()`.

b) Metóda getPlayTime

Náplňou metódy `getPlayTime` je prevod času v mikrosekundách na štandardný zobraziteľný čas formátu „mm:ss“. Vstupným atribútom je čas získaný z prehrávača, buď aktuálny čas alebo celkový, ktorý sa obsiahnutým algoritmom prevedie na reťazec uvedeného formátu.

c) Metódy Image

Tieto metódy naimportujú obrázky bežného formátu do atribútov typu Image, aby boli použiteľné v kóde jazyka Java. Sú to metódy:

- `getImgVol` (obrázok so stupnicou hlasitosti)
- `getImgRepro` (obrázok označujúci, že je prehrávaný audio súbor)
- `getImgHardrock` (obrázok pozadia)

d) Metóda keyPressed

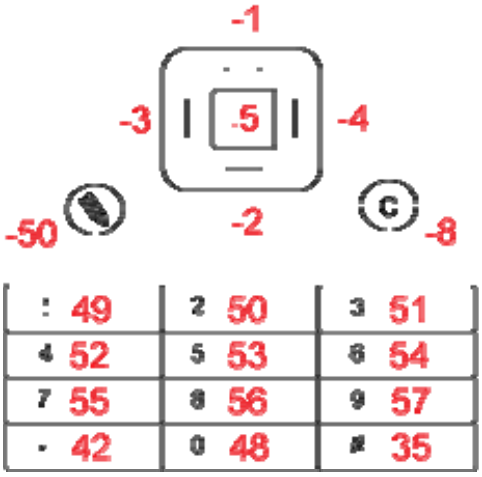
Touto metódou sa vykonáva spomínané zachytávanie návratových hodnôt kláves a následné vykonanie priradených funkcií. Použité sú klávesy šípky z hora na dol, na hlasitosť a stredové potvrdzovacie tlačidlo na funkciu play/stop.

Tlačidlom hornej šípky sa zvyšuje hodnota hlasitosti a jej návratová hodnota je -1. Po každom stlačení sa hodnota hlasitosti zvýši o 10 až pokiaľ nedosiahne hranicu maxima 100. Naopak tlačidlo dolnej šípky hodnotu hlasitosti znižuje o 10 až po hodnotu minima 0. Jej návratová hodnota je -2.

Stredové potvrdzovacie tlačidlo navráti hodnotu -5 a obsahuje funkciu zastavenia skladby na aktuálnej pozícií. Opätovným stlačením sa skladba opäť spustí od miesta, kde bola pozastavená. Po spustení skladby, alebo videa je hlasitosť pozvoľne zvyšovaná na pôvodnú hodnotu, z dôvodu komfortu počúvania.

Tlačidlo „c“ s návratovou hodnotou -8 sa používa vo video prehrávači na prepínanie medzi režimami zobrazenia.

Všetky návratové hodnoty tlačidiel pre mobilné zariadenie Nokia N95 sú zobrazené na Obr. 2.4.



Obr. 2.4. Návratové hodnoty tlačidiel

2.5. Prehrávanie audia zo súboru

2.5.1. Trieda PlayVlakno

Po výbere skladby v súborovom manažéri je možné si ju na mobilnom zariadení vypočuť pomocou tejto triedy. V nej sa skladba začne prehrávať, prehrávaný zvuk sa dostane na výstup, nastaví sa hlasitosť a pod. V nasledujúcich riadkoch je popis algoritmu prehrávania.

Pre prehratie audio súboru sa pre možnosť naďalej používať a ovládať aplikáciu, vytvára paralelné spúšťanie kódu pomocou tzv. multithreadingu. Jedná sa o spustenie niekoľkých častí kódu naraz, každú časť v samostatnom vlákne. V prípade, že by sa celá aplikácia spustila v hlavnom behu programu, niektoré časti by nefungovali, pretože jedna časť by zabrala pre seba celé hlavné vlákno. Java má vo východiskovom stave jedno vlákno, ktoré vytvára základný beh programu a označuje sa ako primárne (hlavné) vlákno, na ktorom beží každá aplikácia [5].

Túto triedu tvorí nové vlákno, v ktorom sa vykonáva program obsiahnutý v implicitnej triede `run()`. Trieda `run()` obsahuje algoritmus prehrávania. Pomocou java triedy `Manager` z balíčka `javax.microedition.media`, sa vytvorí nový prehrávač. Jedným zo vstupných parametrov prehrávača je otvorený multimediálny súbor. Ten sa musí otvoriť a previesť na typ vhodný na prehrávanie. To má za úlohu metóda `getOpenFile()` v triede `MobilePlayer`.

```
player = Manager.createPlayer(getOpenFile(),"audio/mpeg");
```

Získa sa kontrola nad zvukovým výstupom a nastaví sa úroveň hlasitosti pomocou metódy `GetVolLevel` a spustí sa prehrávanie. Následne sa prekreslí hlavná obrazovka, zobrazí sa názov skladby, aktuálna hlasitosť a čas skladby pomocou triedy `ButtonCanvas`.

2.5.2. Dostupné funkcie pri prehrávaní audia

Spustením prehrávača sa nastaví príznak prehrávania a je možné prehrávanie ovládať. Stredové tlačidlo slúži na pozastavenie skladby a opätovným stlačením sa spustí od zastaveného miesta. Tlačidlami hore a dole, ktoré sú súčasťou krížového tlačidla, sa pridáva a uberá hlasitosť prehrávaného obsahu v rozmedzí 0 až 100.

2.6. Prehrávanie videa zo súboru

2.6.1. Trieda PlayVideo

Formáty videa, ktoré je možné prehrať sa líšia od použitého MMAPI a s tým súvisiaceho zariadenia. Táto aplikácia je vyvíjaná na zariadenie podporujúce formát MPEG-4.

Ak bol vybratý súbor s koncovkou „mp4“, je predpoklad, že ide o video súbor spomínaného formátu. V tom prípade je zavolaná metóda na vytvorenie vlákna v triede `MobilePlayer`, v ktorom sa bude prehrávať zvolené video. Toto prehrávanie má na starosti práve trieda `PlayVideo`, ktorá patrí k vytvorenému vláknu.

Na začiatku vypíše na obrazovku správu o nahrávaní súboru z pamäte a následne sa vykonajú opatrenia, ktoré zastavia proces časovača z prehrávania audia, ak bolo pred tým spustené. Následne sa vytvorí prehrávač, rovnakým spôsobom ako pri vytváraní audia, s jediným rozdielom, že druhým vstupným parametrom je tento krát „video/mpeg4“. Získa sa kontrola nad zobrazením videa a hlasitosti zvuku. Nastaví sa hlasitosť pomocou metódy `GetVolLevel()`, ktorá načíta uloženú hodnotu z predošlého prehrávania. Pred prehraním samotného videa, sa musí inicializovať displej. V tejto aplikácií je použité nezávislé plátno Canvas, na ktoré sa zobrazí prehrávané video. Z toho vyplýva, že mód zobrazenia bude USE_DIRECT_VIDEO. Ten vykreslí video na určitý objekt. Týmto objektom je plátno Canvas.

```
VC.initDisplayMode(VC.USE_DIRECT_VIDEO, canvas);
```

Ďalej sa zobrazenie videa aktivuje, dá do módu celej obrazovky („fullscreen“) a spustí sa prehrávanie.

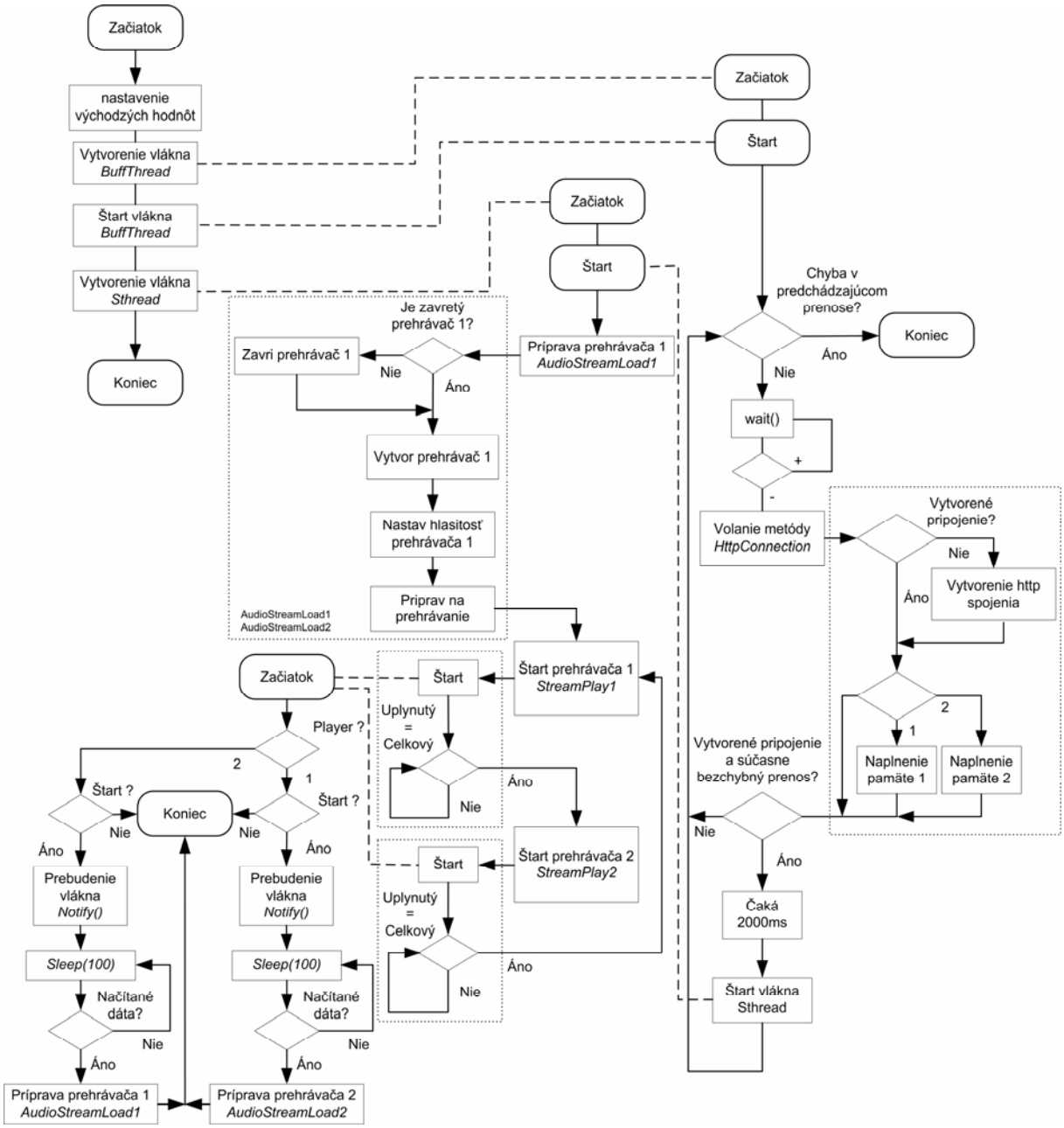
2.6.2. Dostupné funkcie pri prehrávaní videa

Následné prehrávanie je možné zastaviť a následne ďalej spustiť stredovým tlačidlom. Po zastavení sa ukazovateľ stavu zmení z „playing“ (prehráva) na „paused“ (zastavený). Taktiež krížovými tlačidlami v smere hore a dole sa zvýši a zníži hlasitosť v rozsahu 0 až 100. Hodnota aktuálnej hlasitosti sa tiež zobrazí na displeji v celoobrazovkovom zobrazení. Tlačidlom „C“ sa prepína medzi režimom zobrazenia na celú obrazovku a východzím režimom, kedy je k dispozícií menu a stav telefónneho signálu a batérie. Tieto funkcie tlačidiel má na starosti trieda `ButtonCanvas` (kapitola 2.4.2).

2.7. Audio stream

Všetky mobilné zariadenia, ktoré sú cieľovou skupinou pre túto aplikáciu majú možnosť pripojenia do bezdrôtovej siete k internetu. Práve táto výsada umožňuje obohatiť tento multimediálny prehrávač o prehrávanie hudby v reálnom čase z určitého vzdialeného servera. Či sa už jedná o niektorú z komerčných rádiostaníc, alebo vlastného servera, vysielajúceho svoj obsah, napríklad pomocou známeho programu VLC player.

Táto časť aplikácie náleží voľbe „Open URL“ v možnostiach v menu aplikácie. Po zvolení tejto položky sa otvorí formulár, v ktorom sa nachádza pole na zadanie adresy vysielajúceho servera. Pod ním sa nachádza položka „Save“ umožňujúca vyššie zadanú URL adresu uložiť. Ak však požadovaná adresa uložená je, stačí si ju zvoliť zo zoznamu uložených adries v spodnej časti formulára a označiť políčko „Open from saved“. Tým sa načíta adresa zo zoznamu. Načítaná adresa sa použije k prehrávaniu. Voľbou ľavého tlačidla telefónu, ktorému patrí položka „Open“, sa spustí prehrávanie. Popis kódu tejto časti aplikácie je rozpísaná a vysvetlená v nasledujúcich kapitolách. Pre lepšie pochopenie celku na Obr. 2.5 vývojový diagram audio streamu.



Obr. 2.5. Vývojový diagram audio streamu.

2.7.2. Trieda URLaddress

Táto trieda vytvorí vyššie spomínaný formulár na zadanie a správu URL adries pre prehrávanie audio streamu.

Pri zadaní zachytení voľby „Open URL“ metódou `commandAction` v triede `MobilePlayer` sa zavolá metóda `CreateURLform`, ktorá patrí do popisovanej triedy `URLaddress`.

a) Metóda CreateURLform

Táto metóda ma za úlohu len pár funkcií, ale o to dôležitejších. Vytvorí inštanciu formulára `Form` s názvom URLform, priradí mu názov „Open URL“, zhodný s názvom v menu a povkladá do neho jednotlivé prvky, ako napríklad textové pole na zadanie URL adresy a ostatné. Ďalej sa priradia tlačidlám funkcie a to, ľavému funkcia na otvorenie adresy a pravému „Back“ na návrat do hlavnej obrazovky. Následne sa prepne zobrazenie do vytvoreného formulára.

b) Metódy prvkov Formu

Sú to metódy, vytvárajúce prvky, ktoré sa vložia do formuláru *URLform*.

Prvá metóda s názvom `getURLfield()` vytvára textové pole s možnosťou zadať len znaky, ktoré môže obsahovať URL adresa. Jej nadpis je „IP address“ a obsahuje predpísanú časť adresy „http://“, pretože väčšina serverov je založená na http prenose a aj aplikácia vytára http pripojenie so zadaným serverom.

Druhý prvok vytvorí metóda `getSaveCheckBox()` a je to zaškrtávacie políčko s názvom „Save“ bez pridanej grafiky, obrázku. Implicitne je neoznačené.

Tretia metóda `getOpenChoiceGroup()` vytvorí prvok s rovnakým políčkom, ale s názvom „Open from saved“, opäť bez pridaného obrázku a opäť bez implicitne neoznačená

No a poslednou metódou je `getSavedURLGroup()`. Tá vytvorí políčka s uloženými URL adresami, ktoré pomocou ďalšej metódy `OpenURLStore()` sa načítajú zo súboru, vytvoreného a uloženého v mobilnom zariadení. Tiež pridá do menu vo formulári položku na vymazanie záznamu, „Delete“. Tá sa však objaví len po nadídení kurzorom nad túto skupinu políčok a po odídení zmizne. To je spôsobené iným typom ochytávania príkazov po vybratí položky v menu. Jedná sa o `ItemCommandListener`, čo je typ prijímajúci oznámenia o vybratej položke (príkazu), ktoré boli vyvolané v rámci nejakého prvku. V tomto prípade sa jedná o prvok so skupinou adries. Popis ukladania URL adries je popísaný v kapitole 2.7.3.

c) Metóda commandAction

Táto metóda reprezentuje poslucháča udalostí. Tieto udalosti sú spôsobené výberom niektorej z položiek menu, stlačením tlačidla. Pri zachytení udalosti je zavolaná metóda `commandAction` so vstupnými parametrami command, teda príkaz, ktorý bol vykonaný a parameter displayable, teda aktuálne zobrazenie, v našom prípade to môže byť canvas alebo Form Keďže táto metóda je volaná v rámci udalostí triedy `URLaddress`, v ktorej je vytvorený jedine formulár Form, testuje sa len vykonaný príkaz.

Ak bol zavolaným príkazom príkaz „Open“, v podstate sa spustí audio stream s vybratou adresou a prípadne sa adresa uloží. Z hľadiska algoritmu sa načíta obsah poľa so zadanou URL adresou a zistí sa, či bolo označené políčko „Save“. Ak áno, je volaná metóda `SaveURLStore()`, ktorá načítanú adresu uloží. Následne sa zisťuje, či nebolo zvolené políčko „Open from saved“, čím sa prepíše už zapísaná adresa z poľa URL adresy. Onačená adresa zo skupiny uložených adries sa prečíta a vloží do atribútu URL. Nakoniec sa otvorí adresa, začne načítavanie do vyrovnávajúcej pamäte a následné prehrávanie a to zavolaním metódy `PlayRTS(String URLadd)` a zobrazí sa naspäť hlavná obrazovka.

V prípade, že sa jedná o udalosť po stlačení pravého tlačidla „Back“, nevykoná sa žiadna operácia, len sa prepne zobrazenie na hlavnú obrazovku.

2.7.3. Record Store –Záznam adries

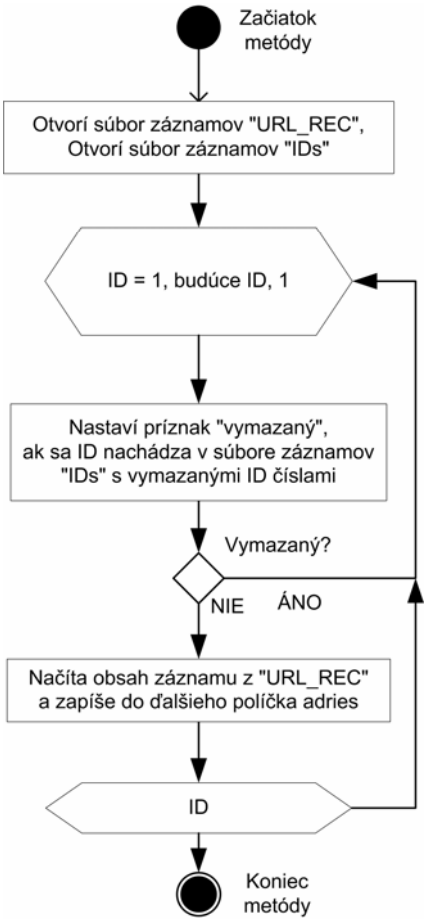
RMS (record management system) poskytuje mechanizmus, pomocou ktorého je možné permanentne uchovať dáta a neskôr ich dostať naspäť. RMS sa nachádza v zabezpečenej časti mobilného zariadenia. Nie je možné sa k záznamu dostať inak, než pomocou aplikácie. RMS zahrňuje Record Store. Ten predstavuje jednu kolekciu záznamov, ktoré zostávajú permanentne uložené v rámci triedy. Dá sa povedať, že ide o trvácnejší atribút, ktorého hodnota zostane nezmenená aj po ukončení aplikácie, vrátane reštartu systému, výmeny batérie atď. Každý záznam je vložený do Record Store dostane jedinečné identifikačné číslo. Tieto čísla sa priradzujú postupne od jedna a s každým pridaným záznamom rastie. ID číslo nie je používané viac krát. Teda aj keď sa záznam vymaže, ID sa už nezopakuje. To znamená, že ak chceme pristúpiť k uloženej položke, musíme poznať identifikačné číslo. V tejto časti je opísaná práca so záznamom. Ide o jeho využitie na uloženie URL adries zadaných k prehrávaniu streamu, teda zápis, načítanie a vymazanie.

Najjednoduchšou časťou je zápis adresy. Zapisuje sa pomocou metódy `SaveURLStore()`. Tu sa otvorí súbor záznamov s názvom URL_REC, ktorý ak nebol pred tým vytvorený, tak sa vytvorí. A to v prípade prvého inštalovania aplikácie. URL adresa sa pretypuje na byte[] a uloží sa do spomínaného záznamu.

Vymazanie adresy zo súboru záznamu sa vykoná po zvolení položky „Delete“ v rámci skupiny políčok. Pri vymazávaní sa zistí síce index vymazávaného záznamu (poradie od vrchu, začínajúce 0) vo výberovom poli adries vo formulári. Ale to nemusí sedieť s ID vymazávaného záznamu. Preto sa vedie druhý Record Store zoznam s názvom „IDs“, kde sa pri vymazaní zapíše ID vymazaného záznamu a vedie sa prehľad vymazaných ID čísel.

Pri otváraní súboru záznamov metódou `OpenURLStore()` sa načítavajú jeho položky pomocou spomínaných ID čísel. Tie sa ukladajú od nuly po rade, ale pri vymazávaní sa to poradie poruší a záznam s ID x nie je už v poradí x-tý, ale x-1.Existuje možnosť zistiť nasledujúce budúce priradené ID, z čoho vychádza maximálny počet možných položiek. To je využité pri získavaní položky zo záznamu. Prechádza sa každé ID od nuly po ich zistený maximálny počet a porovnáva sa s ID zo záznamu vymazaných. Ak sa nezhodujú, znamená to, že záznamu s týmto ID bol vytvorený, ale ešte nebol vymazaný, tak sa použije ako vstupný parameter pre načítanie obsahu záznamu. Ten sa následne vloží, ako ďalšie políčko s URL adresou. Toto sa opakuje až pokiaľ sa neprejdú všetky doposiaľ vytvorené ID čísla v URL_REC súbore. Pre lepšiu predstavu je znázornený vývojový diagram na Obr. 2.6.

Výraz „budúce ID“ vo vývojovom diagrame reprezentuje metódu `getNextRecordID()`, ktorá, ako už bolo spomínané, vráti nasledujúce budúce priradené ID.



Obr. 2.6. Vývojový diagram metódy OpenURLStore

2.7.4. Prehrávanie audio streamu – triedy a metódy

V tomto štádiu má užívateľ vybratú adresu stream servera, z ktorého chce prehrávať hudobný obsah. To je základný parameter na zahájenie množstva procedúr potrebných k tomu, aby sa do audio výstupu mobilného zariadenia dostali prvé tóny.

Prehrávanie audio streamu má na starosti niekoľko dôležitých častí. Časť pripojenia sprístupní dátový obsah zo servera, ktorým sa začnú plniť striedavo dve vyrovnávacie pamäte. O prehranie obsahu vyrovnávacích pamätí sa starajú dva prehrávače, ktoré striedavo prehrávajú obsah naplnenej vyrovnávacej pamäte.

Na každú časť sú vytvorené zvlášť vlákna. Základný beh programu obstaráva základné funkcie, ako tlačidlové udalosti, ukončovanie a vytváranie vlákien atď. V prvom vlákne sa vytvárajú prehrávače a spúšťa sa prehrávanie a druhé vlákno plní vyrovnávaciu pamäť. Tieto činnosti sú rozdelené do vlákien, aby sa zlepšila stabilita a rýchlosť aplikácie a aby bolo možné ošetriť prípadné výnimky a zlyhania toku dát.

Aplikácia sa v prvom kroku pokúsi otvoriť zvolenú URL adresu. Tomu nasleduje požiadavka zariadenia na pripojenie do siete. Po úspešnom pripojení sa otvorí dátový tok zo servera a naplní prvú vyrovnávaciu pamäť. Po naplnení sa toto dátové pole, vyrovnávacia pamäť, použije k vytvoreniu prehrávača. Ten spustí prehrávanie a vtedy sa rozdvojujú procesy vykonávané aplikáciou. Prvý prehráva obsah z vyrovnávacej pamäte a druhý, v momente štartu prehrávača, začína ukladať do druhej vyrovnávacej pamäte prichádzajúce dáta. Po naplnení sa vytvorí druhý prehrávač a čaká, až dohrajú dáta z prvej vyrovnávacej pamäte. V momente, kedy sa prehrávanie takmer ukončí, spúšťa vytvorený druhý prehrávač. Ten prvý dobrá a začnú sa prichádzajúce dáta plniť do prvého dátového poľa pre prvý prehrávač. Takýmto kolobehom sú dáta prichádzajúce zo servera prehrávané v aplikácií. V nasledujúcom texte sú bližšie opísané jednotlivé kroky algoritmu.

a) Metóda PlayRTS

Je základnou metódou celého prehrávania audio streamu. Je volaná v prípade, že je známa adresa URL a je udalosťou požadované spustenie prehrávania.

V úvode metódy sa zastaví všetok doterajší beh prípadných prehrávačov známou metódou `stopAllPlay()`. Všetky príznaky, ktoré sa používajú na zisťovanie stavu vykonávaných aj nevykonávaných častí, sa nastavujú na východzie hodnoty, pre prípad, že by nebol stream korektne ukončený, alebo by sa prerušil, prípadne by nastal jeden z ďalších mnohých možných stavov. Následne sa zavolá metóda vytárajúca vlákno *BuffThread*, z triedy

pre sieťovú komunikáciu a prácu s dátovými poľami. Vlákno sa zobudí, pretože ihneď po spustení je zastavené. Vlákno začne vykonávať zadaný algoritmus, pripojenie a príjem dát, a v opisovanej triede sa v ďalšom kroku vytvorí vlákno na prehrávanie. To je však spustené neskôr, po naplnení prvej pamäte vo vlákne *BuffThread*.

b) Trieda BufferingThread

Táto trieda sa vykonáva vo vlákne číslo dva, zaobstarávajúcom sieťové spojenie. Ihneď po spustení sa vlákno a teda aj vykonávanie algoritmu triedy zastaví metódou `wait()` a čaká na zavolanie zobudením. Vlákno sa budí v momente, kedy je potrebné prijať ďalšie dáta do nasledujúcej vyrovnávacej pamäte. Po zobudení nasleduje volanie metódy `HttpConnection`, ktorá v rámci vlákna vytára prvotné spojenie, prijíma a plní vyrovnávacie pamäte pre prehrávače. Po prebehnutí sa zisťuje príznak prehrávania a prípadnej chyby pri prenose. Ak chyba nenastala a tento algoritmus sa vykonáva prvý krát, to znamená, že prehrávače ešte nie sú vytvorené a spojenie bolo práve nadviazané, vlákno sa na 2s uspí. Je to pre vytvorenie časovej rezervy medzi prehrávaním a prijímaním dát. Po uplynutí dvoch sekúnd sa spustí vlákno prehrávania a okrem nastavenia príznakov, že bolo nadviazané spojenie a bol rozbehnutý stream, sa v hlavičke hlavnej obrazovky zobrazí nápis „audio stream“.

c) Metóda HttpConnection

Tvorí hlavnú časť sieťovej komunikácie. Je to metóda, ktorá sa vykonáva v druhom z vlákien. Ihneď po zavolaní sa nastaví príznak načítavania na pozitívny. Nasledujúcim krokom je test, či sa vykonáva tento algoritmus po prvý krát a to atribútom typu boolean. Za predpokladu, že sa ešte nevykonal, vytvoria sa dve bajtové polia o dĺžke 100000 bajtov. Tie sa priradia vyrovnávacej pamäti, poľu, vhodného na prehrávanie. Vytvorí sa požiadavka na otvorenie spojenia s danou URL adresou. A je úspešné, pripojí sa na dátový tok servera.

```
rtp = (HttpConnection) Connector.open(URLadr);
dis = rtp.openInputStream();
```

Pri prvom spustení je príznak vyrovnávacej pamäte nastavený na prvú z nich, tak sa začne čítať dátový tok a ukladať prvých 100000 bajtov do prvého dátového poľa. Následne sa overia prípadné chyby na prenose a príznak načítavania na negatívny.

d) Trieda RTPThread

Trieda pre prehrávanie vytvára vlákno číslo jedna, ktoré sa spustí po naplnení prvej vyrovnávacej pamäte v triede `BufferingThread`. A to z dôvodu, že vlákno slúži na

prehratie dát z vyrovnávacej pamäte, preto je vlákno spustené až po jej naplnení. Následne sa zavolá metóda `AudioStreamLoad1()`, teda príprava prvého prehrávača a potom metóda `StreamPlay1()` vykonávajúca samotný štart. Tým sa úloha triedy končí, pretože prepínanie medzi prehrávaním riešia už spomínané triedy samé.

e) Metódy AudioStreamLoad 1 a 2

Sú to identické metódy, ktoré sa líšia jedine spracúvajúcou vyrovnávacou pamäťou. Metódy pripravujú prijaté dáta na prehranie vytvorením prehrávača, nastavením hlasitosti a rozkódovaním formátu prijatého hudobného obsahu. Fakt, že sa začína pripravovať jeden z prehrávačov znamená, že ten druhý je v tej chvíli už spustený. A preto, pre zvýšenie spoľahlivosti, sa ten predošlí prvý, ktorý sa ide opäť vytvárať a ešte nie je uzavretý, uzavrie hneď na začiatku. Po vytvorení prehrávača sa zistí a nastaví hlasitosť buď z predošlého, alebo ak je prehrávanie spustené po prvý krát, načíta sa uložená hodnota metódou `GetVolLevel`. Posledným krokom pred samotným rozkódovaním je nastavenie poslucháča udalostí na vytváraný prehrávač. Udalosťami je napríklad štart, zastavenie alebo ukončenie prehrávania. Poslucháč `PlayerListener` je definovaný metódou `playerUpdate`, ktorá je implicitne vytváraná pri priradení poslucháča.

Takto vytvorený prehrávač má alokované zvukové zariadenia a zásobníky naplnené zvukovými dátami a čaká na zahájenie prehrávania v jednej z metód `StreamPlay1()` a `StreamPlay2()`.

f) Metódy StreamPlay 1 a 2

Týmto metódam musí predchádzať vytvorený prehrávač, pretože zahŕňajú predovšetkým akciu štart. Okamžite po zavolaní, prvým príkazom v oboch metódach, sa spustí prehrávanie jedného z prehrávačov. Tomu predchádza ukončenie predošlého prípadne vytvorenie nového prehrávača, v prípade, že prehrávanie ešte nebolo zahájené. Zistí sa dĺžka prehrávaného úseku v milisekundách následne sa každých 50ms testuje, či sa aktuálna doba prehrávaného úseku nerovná celkovej dĺžke, teda či sa nedosiahlo konca úseku. Za čas prehrávania tohto úseku, sa naplní druhá vyrovnávacia pamäť a pripraví sa druhý prehrávač. Po detekovaní ukončenia, sa spúšťa metódou `StreamPlay1()` alebo `StreamPlay2()` prehrávanie druhého prehrávača. Na odstránenie krátkej pauzy, ktorá vznikne medzi ukončením predošlého a spustením nasledujúceho úseku sa detekuje koniec úseku o 60ms skôr, než sa skutočne ukončí.

g) Metóda playerUpdate

Metóda so vstupnými parametrami `playerUpdate(Player player, String string, Object eventData)` je volaná, keď nastane nejaká udalosť v triede *Player*. Jedna z udalostí použitá v tejto metóde je štart prehrávača.

Po zahájení štartu je vstupným parametrom prehrávač, v ktorom zachytená udalosť prebehla. Hneď v prvom kroku algoritmu sa testuje, ktorý z prehrávačov spustil udalosť a podľa toho sa vykonáva následne kód. Akcie na štart prehrávača sú v podstate identické, len sú zmenené parametre pre konkrétnu časť prehrávania. V nasledujúcom kroku sa vyfiltrujú udalosti a pokračuje sa len v prípade, že sa jedná o štart. Následne sa nastaví príznak čísla vyrovnávacej pamäte. Ak bol spustený prehrávač číslo jedna, zahajuje sa plnenie pamäte číslo dva. Pointer na zápis do dátového pola sa nastaví na začiatok funkciou reset. Okrem zabezpečení proti možným výnimkám, ktoré sú popísané v kapitole 2.9.3, sa zobudí druhé vlákno. To bolo zastavené po načítaní do predošlej vyrovnávacej pamäte. Implicitnou metódou `notify()` sa opäť rozbehne a naplní dátový blok k prehrávaniu. Testuje sa príznak plnenia v metóde `HttpConnection` a ukončení sa spustí metóda `AudioStreamLoad1()` alebo `AudioStream2()`, ktorá hudobné dáta opäť pripraví k prehrávaniu.

2.8. Odoslanie a príjem video sekvencie

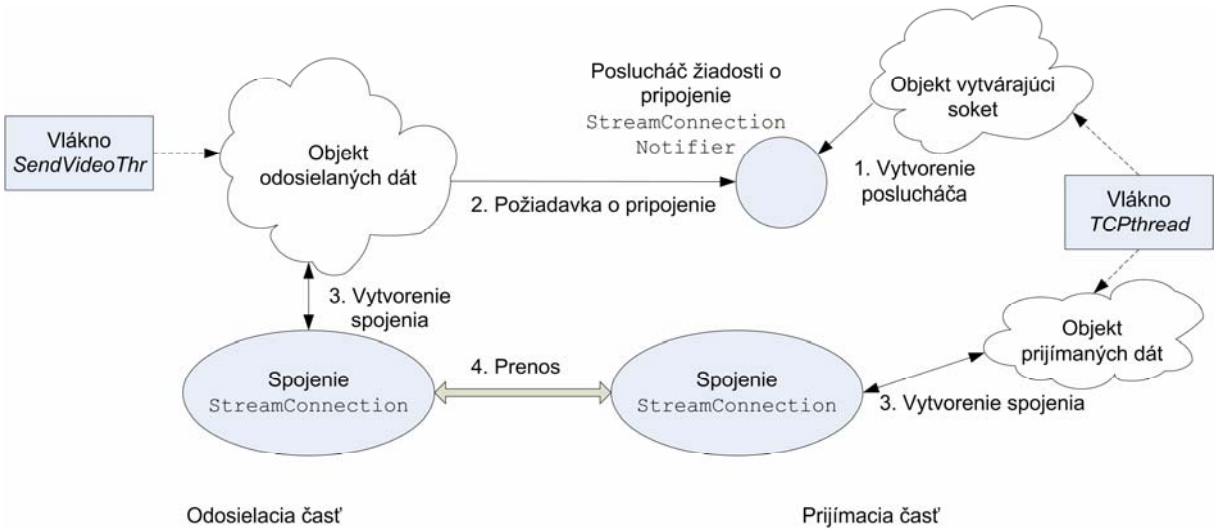
Možnosť využitia vstupného video zariadenia, kamery, podnietilo vývoj jeho využitia. Po mnohých testovaniach bol nakoniec multimediálny prehrávač rozšírený o možnosť záznamu a výmeny video záznamu medzi zariadeniami pomocou sieťového pripojenia. Prenos je možný jedine v prípade, že na dvoch zariadeniach pripojených v sieti je nainštalovaný tento prehrávač a obe zariadenia ho plne podporujú. Vyvíjané na mobilných telefónoch Nokia N95. Na jednom zo zariadení je aplikácia v móde čakajúcom na príjem „Camera recieve“ a druhá, vysielacia, v časti „Camera send“, zaznamená a pošle video sekvenciu. Záznam a odosielanie už nie je možné spustiť v ktoromkoľvek emulátore. To zapríčinilo, že obe časti aplikácie boli vyvíjané v mobilnom telefóne, čo je časovo veľmi náročné.

Časť aplikácie, ktorá ma na starosti zachytenie videa a následné odoslanie sa skladá z dvoch okien. Prvé po spustení začne prehrávať obraz zachytávaný sekundárnym vstupným obrazovým zariadením. Sekundárne z dôvodu, že je vždy dostupné. Naopak primárna kamera má ochranný kryt. A po jeho otvorení je implicitne spúšťaná aplikácia dodávaná v mobilnom zariadení na záznam videa, ktorá spoločne s Java aplikáciou, multimediálnym prehrávačom,

občas koliduje. Následne sa voľbou menu obraz začne ukladať do dátového poľa. Takto uložené pole je pripravené na prenos. Vtedy sa zobrazí formulár s časťou pre zadanie IP adresy druhého prijímacieho zariadenia. Na túto IP adresu je následne súbor odoslaný. Aplikácia vytvorí TCP spojenie medzi zariadeniami, odošle dátovú dĺžku súboru a po potvrdení odošle dátové pole. Po úspešnom odoslaní je zobrazené hlásenie a zobrazenie prejde naspäť k prehrávaniu kamery s položkou v menu „Send“ k opätovnému odoslaniu. Pre nový záznam je nutné sa vrátiť do hlavnej obrazovky stlačením tlačidla „Back“ a následne vybrať položku zachytávania z kamery „Camera send“.

Ďalšia časť aplikácie slúži na príjem zachytenej video sekvencie z kamery druhého zariadenia a tiež na následné prehranie. Spustí sa položkou „Camera recieve“ v menu aplikácie. Vtedy sa hlavná obrazovka mení na obrazovku prijímania súboru. V prvom kroku čaká aplikácia na zahájenie naslúchania na porte. Po zvolení položky „Recieve“ si aplikácia vyhradí port 50000. Tento port sa radí do skupiny portov, využívaných pre komunikáciu klienta so serverom (porty 49152 až 65535). Nepatrí teda medzi registrované a „well know“ (dobre známe) porty, ktoré sú využívané v komerčnej sfére, prípadne operačným systémom. Na tomto porte čaká na požiadavku na spojenie a na obrazovke sa zmení stav z „NOT CONNECTED“ na „WAITING...“. V hornej lište je vypísaná dôležitá informácia a to IP adresa tohto čakajúceho zariadenia, na ktoré má opačná strana dáta odoslať. Prípad, že IP adresa je v tvare 127.0.0.1, znamená, že sa jedná o loopback zariadenia a telefón nie je pripojený do siete a aplikácia nie je schopná prijať dáta z vonku. Po prijatí požiadavky o spojenie sa vytvorí TCP spojenie medzi oboma zariadeniami a táto strana začne prijímať obrazové dáta. Opäť sa zmení stav z „WAITING...“ na „RECIEVING...“, čím upovedomí o prebiehajúcom prenose. Po úspešnej výmene dát je tento blok otvorený vo video prehrávači aplikácie.

Spojenie medzi aplikáciami je zabezpečované pomocou vytvoreného sieťového soketu na strane poslucháča. Soket je v sieťovej terminológií chápaný ako spojovo orientovaný sieťový mechanizmus na transportnej vrstve, ktorý obyčajne implementuje pár TCP/IP protokolov v danom prostredí internetovej siete. Sokety nemajú zavedenú žiadnu štruktúru na dátový tok. Na rozdiel od datagramov, ktoré jednoducho prenášajú sekvenciu bytov. Služba soketu definuje formát, syntax a sémantiku prenášaných dát. Klient sa následne musí držať týchto definicí, aby mohol danú službu využívať. [6] Obe časti aplikácie vytvárajú spoločne komunikáciu, ktorá je znázornená na Obr. 2.7.



Obr. 2.7. Vzájomná komunikácia pri prenose video sekvencie

2.8.2. Záznam a odoslanie videa – triedy a metódy

Táto časť má algoritmus rozdelený do dvoch tried. V prvej triede `VideoStream` sa nachádza vytvorenie vlákna na prehrávanie obsahu z kamery a tiež záznam videa a volanie metódy na vytvorenie formulára, ktorý sa už nachádza vo vlastnej triede `SendVideoSeq`, spolu s nadviazaním spojenia a odoslaním súboru. V nasledujúcich riadkoch sú vysvetlené dôležité triedy, metódy a ich funkcie.

a) Subtrieda VideoStreamThread v triede VideoStream

Z objektu tejto triedy je vytvorené nové vlákno, preto trieda implementuje `Runnable` interface. Úlohou tohto vlákna je zobraziť na výstupe, teda obrazovke Canvas, video zachytávané kamerou. Tomu predchádza zrušenie časovača, ktorý mohol pretrvávať z prípadného prehrávania hudobného súboru. Vytvorí sa menu a to spôsobom, že sa odstránia všetky položky hlavnej obrazovky a pridá sa len funkcia „Start“ a „Exit“ na návrat do hlavnej obrazovky. Následne na to sa metódou `StartCameraPlay()` spustí prehrávanie obrazu z kamery na displej. Vlákno sa zastaví implicitnou metódou `wait()`, prehrávanie naďalej beží a čaká sa na zahájenie akcie nahrávania zvolením možnosti „Start“. Program vykresľuje a počúva príkazy v triede `MobilePlayer`, čomu svedčí fakt, že sa nemenila obrazovka a funkcie tlačidiel sa len odstránili a následne pridali. Udalosti sú zachytávané tiež poslucháčom v pôvodnej triede.

Po zachytení udalosti na akciu stlačenia „Start“ sa z hlavnej triedy zobudí vlákno a vykoná sa následný algoritmus. Vymení sa príkaz „Start“ za príkaz „Stop“ a metódou

`RecordBuff1()` sa začne nahrávanie, ukladaním dát z kamery pamäte. Tu sa nasledujúcim príkazom nastavil cieľ video sekvencie.

```
rc.setRecordStream(Voutput1);
```

Vlákno sa opäť zastaví čaká na ukončenie nahrávania funkciou „Stop“. Po výbere tejto funkcie sa pomocou `notify()` vlákno zobudí a zavolá vytvorenú metódu StopRecordBuff1, v ktorej metóda `commit()` ukončí nahrávanie a nahraté surové dáta v pamäti sa obalia hlavičkou a zapíšu sa do dátového poľa *Voutput1*. Počas nahrávania bola aplikácia zbavená možnosti návratu do hlavného okna, pre zníženie náročnosti algoritmu, ktorý by musel byť rozšírený o ďalšie vlákno. V tomto kroku je video sekvencia uložená v použiteľnom atribúte, dátovom poli. A čaká sa na prechod do časti prenosu. Opäť je vlákno pozastavené a opäť sa funkcia tlačidla zmenila, zo „Stop“ na „Send“.

b) Metóda StartCameraPlay

Úlohou tejto časti aplikácie je pripojiť sa na kameru zabudovanú v mobilnom zariadení. Z dôvodov spomínaných na začiatku kapitoly je zvolená druhá, menšia kamera pre jednoduchší prístup. Následne zostaví prehrávač, ktorý bude obraz z kamery prehrávať na obrazovke aplikácie. Samotná stavba prehrávača je z veľkej časti totožná so stavbou video prehrávača v kapitole 2.6.

Hlavným rozdielom je zostavenie prehrávača v prvom kroku. Na prehrávanie videa z kamery sa vstupný parameter zadáva len jeden a nepoužije sa žiadny dátový blok s multimediálnym obsahom, ale priamo cesta prístupu na internú kameru zariadenia. Cesta pre sekundárne video zariadenie použité v aplikácií je "capture://devcam1". Zoznam ďalších možných parametrov na prístup k rôznym zariadeniam je v Tab. 2.1.

Tab. 2.1. Zoznam parametrov na prístup k zariadeniam v mobilnom zariadení

"capture://video"	zachytávanie videa z východiskového zariadenia
"capture://devcam0"	zachytávanie videa zo primárneho zariadenia
"capture://devcam1"	zachytávanie videa zo sekundárneho zariadenia
"capture://radio?f=105.1&st=stereo"	zachytávanie audia z FM tuneru s frekvenciou 105,1MHz v móde stereo
"capture://audio"	zachytávanie audia z východiskového zariadenia

Ostatné časti metódy sa takmer zhodujú so spomínaným video prehrávačom. Rovnako sa získava kontrola nad zobrazovaním videa, ale na druhej strane sa nezískava kontrola zvuku a teda zvuk nie je ani prehrávaný. Mohla by vznikať spätná väzba, prípadne len nepríjemná

ozvena zachytených zvukov. Následne je inicializovaný displej, prehrávanie zvolené na plátno Canvas. Ďalšou zmenou oproti pôvodnému prehrávaču je získanie možnosti zachytávať prehrávaný obsah. Slúži k tomu interface RecordControl, ktorý ovláda nahrávanie média z prehrávača. Spusteniu nahrávania predchádza zadanie dátového poľa. Zadanie poľa ako aj samotné nahrávanie je však vytvorené v časti záznamu audio sekvencie.

c) Trieda SendVideoSeq

Po zvolení možnosti „Send“ je zavolaná metóda na vytvorenie formulára, ktorá sa nachádza v ďalšej triede časti aplikácie, záznam a odosielanie. Táto má na starosti výber IP adresy, spojenie s poslucháčom na vzdialenom servery a prenos obsahu dátového poľa.

Vytvorenie nového formuláru *IPform* typu Form, ktorému sa priradí textové pole na zadanie IP adresy. Vytvorí sa menu priradením príkazu „Back“ a „Send to“ do formulára. Tieto príkazy sa vytvárajú v konštruktore triedy. Táto trieda obsahuje aj vlastného poslucháča `commandListener`, ktorý je priradený tomuto formuláru. Akcie na udalosti príkazov sú vykonávané opäť metódou `commandAction`, obsiahnutou v triede.

Po vyplnení textového poľa IP adresou a potvrdením tlačidla sa zachytí udalosť v metóde `commandAction` na príkaz „Send to“ a z poľa s adresou sa zapíše kombinácia do atribútu *IP*. Následne sa volá metóda `SendThread()`, ktorá pošle nahratý obsah.

Táto metóda vytvára pomyselného klienta vytváraného spojenia. Prvým krokom je požiadavka klienta o pripojenie na známy „well-known“ soket odoslaním klientskej požiadavky v štandardnom formáte. Nasledujúci príkaz reprezentuje spomínanú požiadavku.

```
StreamConnection soc = (StreamConnection) Connector.open
                       ("socket://" + IP + ":50000");
```

Musí sa uviesť meno naslúchajúceho druhého zariadenia, pomyselného servera, ktorého v tomto prípade označuje IP adresa a číslo portu reprezentujúceho známy soket servera, ktorý je pre komunikáciu zvolený z najvyššieho rozsahu. Týmto sa vytvorí objekt `StreamConnection`, pomocou ktorého budú obe strany posielať a prijímať dáta. Po nadviazaní spojenia sa na obrazovke zobrazí hlásenie o pripojení a v hlavičke formulára stav prenosu, „Sending...“. Následne sa vytvoria atribúty typu OutputStream a InputStream, ktoré budú spracovávať tok dát zo zdroja a do zdroja sieťového pripojenia ako sekvenciu surových bajtov.

Po vytvorení spojenia sa zaháji prenos video sekvencie. Zistí sa veľkosť dátového poľa sekvencie v bajtoch a tá sa odošle druhej aplikácií. Avšak zistená veľkosť je

v celočíselnom tvare a musí sa previesť do bajtovej hodnoty. To zabezpečuje metóda `intToByte(int i)`, ktorá navráti číslo v 4 bajtoch, čo predstavuje hraničnú veľkosť celočíselného typu atribútu. Tieto 4 bajty sa pomocou nasledujúcej funkcie zapíšu do odchádzajúceho dátového toku.

```
OutData.write(lengthInByte, 0, 4);
```

Metódou `flush()` sa okamžite vypustia do prenosu všetky zhromaždené dáta, ktoré čakali vo vyrovnávajúcej pamäti na doplnenie a ešte neboli odoslané. Následne sa čaká na potvrdzovací bajt, ktorý odošle druhá strana po prijatí dát s dĺžkou prenášaného hlavného úseku. Po potvrdení sa na výstup začnú za sebou posielať bajty z bajtové poľa typu ByteArrayOutputStream. Po zapísaní celej veľkosti, sa opäť vypustia do prenosu prípadné čakajúce dáta. Po potvrdení bajtom z druhého zariadenia sa prenos uzavrie a zobrazí sa hlásenie o úspešnom odoslaní. Zobrazenie sa prepne do prehrávania obrazu z kamery s možnosťou opätovného prenosu.

2.8.3. Príjem a prehranie videa – triedy a metódy

Na prijatie súboru je určená položka „Camera recieve“ v menu hlavnej obrazovky. Príjem a prehrávanie prijatého videa je algoritmicky menej rozsiahla časť, pretože nepotrebuje vytvárať novú triedu pre vlákno. Obsahuje len triedu na vytvorenie spojenia a príjem, ktorá je podtriedou hlavnej triedy `VideoStream`. Následné prehratie je ponechané na pôvodnom prehrávači videa, opísanom v kapitole 2.6.

a) Metódy zobrazenia

Po udalosti na tlačidlo prijatia video sekvencie je zavolaná metóda `VideoServer()` z hlavnej triedy `MobilePlayer`. V hlavnej triede je ponechaný aj poslucháč udalostí tlačidiel a teda všetky akcie sú volané z metódy `commandAction` v hlavnej triede. V tejto metóde sa ponechá zobrazenie pôvodného plátna, zobrazí stav spojenia, „NOT CONNECTED“ a vymení sa ponúkané menu a to odobratím položiek hlavnej obrazovky a priradením nových, ktorými sú na začiatok „Recieve“ a „Back“. Funkcia tlačidla „Back“ podľa nastavovaných príznakov spojenia buď len priradí spätne plátnu pôvodné menu, alebo aj ukončí spojenie a vráti sa do hlavnej obrazovky. Tlačidlom „Recieve“ sa volá metóda `RecieveVideo()`.

Obsah tejto metódy ako prvé vytvorí objekt triedy `ConnectionThread`, ktorá obstaráva sieťové spojenie. Na to sa vytvorí nové vlákno, v ktorom budú bežať procesy

z triedy sieťového spojenia. Následne sa vlákno spustí a na obrazovke sa aktualizuje stav spojenia na „WAITING...“, čomu odpovedá súčasne vytvárané spojenie v bežiacom vlákne.

b) Subtrieda ConnectionThread v triede VideoStream

V triede `ConnectionThread` je definované sieťové spojenie s odosielacou časťou a následná komunikácia a príjem video sekvencie. Spustením novovytvoreného vlákna sa pristúpi k algoritmu v tejto podtriede.

Ako prvé je nutné poznať IP adresu prijímacieho zariadenia, na ktoré sa má odoslať požadovaný dátový blok. To rieši dočasné vytvorenie soketu pomocou `ServerSocketConnection`. Objekt tejto triedy umožňuje zistiť lokálnu IP adresu a port pomocou metódy `getLocalAddress()`. Po získaní pridelenej IP adresy a portu sa vypíšu za sebou do hlavičky a následne sa vytvorený soket uzavrie.

Následne vytvorí opäť soket, ktorý bude tvoriť základ pre komunikáciu a je označený ako „dobre známy“ (well-know). Tento soket sa ďalej priradí k portu, na ktorom bude počúvať prichádzajúce požiadavky. Syntax vytvorenia soketu je nasledovný:

```
StreamConnectionNotifier notifier = (StreamConnectionNotifier)
                Connector.open ("socket://:50000");
```

Zápis adresy, na ktorej sa poslucháč vytvorí je `<protokol>://<cieľ>`. Protokol komunikácie sa zvolil soket. Cieľ sa zapisuje v tvare [<hostiteľ>]:<port>, kde hostiteľ predstavuje platný DNS názov, alebo IP adresu a číslo portu platného systémového portu. Absencia alebo prítomnosť hostiteľa v zápise cieľa určuje či je spojenie servera, alebo klienta. V tomto prípade je hostiteľ vynechaný, čiže je soket vytváraný serverom. Takto vytvorený poslucháč počúva na vytvorenom pripojení a čaká žiadosť o pripojenie klienta so syntaxom:

```
StreamConnection sc = notifier.acceptAndOpen();
```

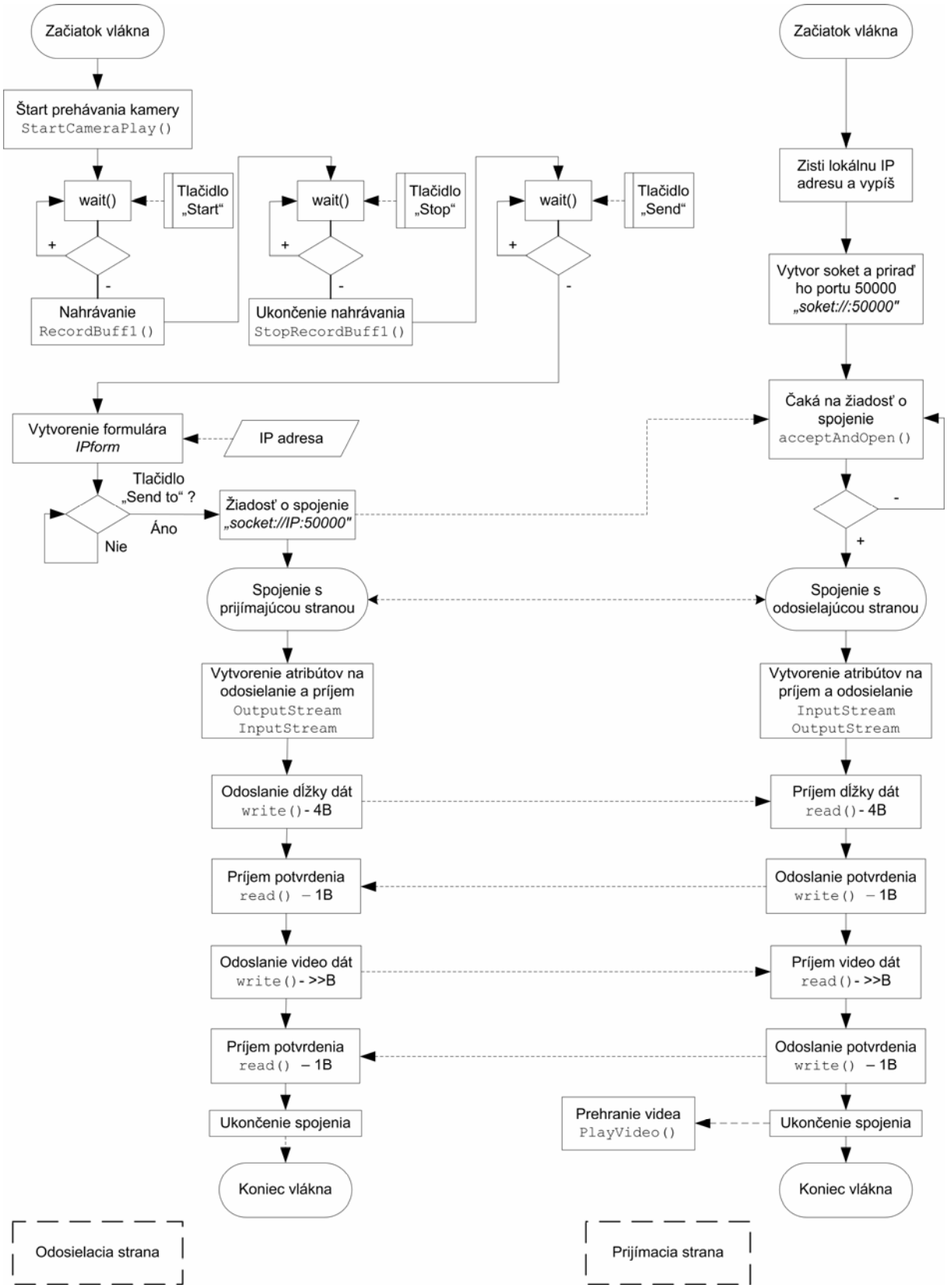
Tento príkaz zablokuje vykonávanie algoritmu v danom vlákne pokiaľ nepríde požiadavka od odosielajúcej strany na pripojenie. Po obdržaní požiadavky na pripojenie, metóda `acceptAndOpen()` spracuje požiadavku ešte pre vykonaním akejkoľvek ďalšej akcie. Spracovaním požiadavky sa rozumie:

- Akceptovanie požiadavky na pripojenie
- Vytvorenie nového objektu pripojenia
- Priradenie spojenia nevyužitému soketu
- Informovanie odosielacej strany, klienta o novom soketovom spojení

Preto ten názov `StreamConnectionNotifier`. Serverová časť je informovaná („notified“) o žiadosti o pripojenie. Metóda `acceptAndOpen()` následne vráti objekt stream spojenia `StreamConnection`, cez ktorý bude prijímacia strana komunikovať s odosielacou. Po vytvorení spojenia sa zobrazí hlásenie a stav „WAITING...“ sa zmení na „RECIEVING...“. Následne sa rovnako ako v odosielacej časti, vytvoria reprezentanti odchádzajúcich a prichádzajúcich dát `OutputStream` a `InputStream`. Ďalším krokom je očakávanie 4 bajtového slova s veľkosťou prijímaného dátového toku s video obsahom. Po prijatí 4 bajtov sú pomocou metódy `byteToInt()` prevedené na celočíselný typ Integer a podľa toho je alokované pole. Pole o veľkosti očakávaného súboru je následne metódou `read()` naplnené z prichádzajúceho dátového toku. Po prijatí celej dĺžky dátového poľa je odoslaný kontrolný bajt prijímacej strane. Následne sa nastavia príznaky prenosu na negatívne a prenos sa uzavrie.

Po uzavretí prenosu sú dáta video sekvencie celé na strane prijímateľa a pripravené na prehrávanie. To je uskutočnené pomocou pôvodného prehrávača videa a preto sa nastaví príznak prehrávania na prehrávanie video sekvencie, aby sa nepokúšal otvoriť súbor z karty a zaháji sa prehrávanie. Pre samotným spustením inštancie video prehrávača sa ešte upraví zobrazenie výmenou tlačidiel menu, za pôvodné.

Popri slovnom opise, je na Obr. 2.8 zobrazený vývojový diagram, znázorňujúci beh programu pre záznam, odoslanie a príjem video sekvencie.



Obr. 2.8. Vývojový diagram záznamu a prenosu video sekvencie

2.9. Výnimky aplikácie a ich ošetrenie

V každej aplikácií, nielen multimediálneho charakteru, môžu nastať stavy, ktoré sa vymykajú normálu a nie sú ideálne pre beh a potreby programu. Konkrétnejšie, v tejto aplikácií patria medzi výnimky všetky situácie, ktoré nedokáže spracovať základný kód a sú nepodporované. Či už sú to vstupno-výstupné dáta, alebo spojenie k nim. V nasledujúcich riadkoch sú popísané možné situácie a ich riešenie v závislosti na konkrétnej časti aplikácie.

2.9.1. Audio prehrávanie zo súboru

Pri výbere a prehrávaní nie je toľko možností zlyhania aplikácie, ako v iných častiach. Avšak výber nepodporovaného súboru je jedným z výnimiek. Po výbere zo súborového manažéra sa prehrávač pokúsi vytvoriť, ale neúspešne. V programe je výnimka média odchytená a na základe chyby sa zavolá metóda vracajúca upozornenie typu Alert, upozorňujúce na nepodporovaný formát.

Naspäť v súborovom manažéri sú tiež ošetrené situácie, kedy nie je prístupná pamäťová karta a program by sa bez ošetrenia náhle uzavrel. Taktiež pri pristupovaní do položky, ktorá je poškodená, alebo neprístupná k čítaniu.

Ošetrením by sa dalo nazvať aj to, že pri prehrávaní iného prehrávača vo vlastnom vlákne a súčasnom spustení audio prehrávania, sa predošlí prehrávač ukončí a následne aj jeho vlákno a až potom sa pristúpi k prehrávaniu v audio prehrávači.

2.9.2. Video prehrávanie zo súboru

Prehrávanie videa obnáša podobné problémy ako audio prehrávanie. Opäť je to nekompatibilita formátu súboru a požadovaného formátu prehrávačom. Pri spustení video prehrávača sa uzavrie tiež prípadné prehrávanie a ukončí sa vlákno. V prípade audio streamu je to okrem ukončenia prehrávania aj uzavretie vlákna, ktoré zabezpečuje príjem dát.

2.9.3. Audio stream

V rámci prehrávania audio streamu je nutné brať do úvahy nie len prehrávače, ale množstvo ďalších, na zlyhanie veľmi náchylných okolností sieťového pripojenia a príjmu dát.

Pri nadväzovaní spojenia je možné, že zadaná URL adresa užívateľom nie je správna, prípadne nevysiela a teda nie je sa možné s takýmto serverom spojiť. To je nad rámec metódy zaisťujúcej pripojenie a rieši sa ošetrenie výnimky. To spôsobí, že v takomto prípade aplikácia nezlyhá, ale vypíše chybové hlásenie o nedostupnosti zadanej URL adresy. Ďalšou z možností je, že v počas čítania dát a súčasného prehrávania sa preruší spojenie, či už stratou

wi-fi signálu, alebo inými okolnosťami mimo pripojenú sieť. Vtedy sa nenaplní vyrovnávacia pamäť do plnej kapacity a aplikácia sa pokúsi o opätovné načítanie, v prípade, že by sa jednalo len o dočasný výpadok. Ak sa nepodarí naplniť buffer ani po druhý krát je nastavený príznak, ktorý nedovolí prehrávaniu spustiť druhý prehrávač a prehrávanie sa ukončí a tiež sa uzavrie vlákno sieťového spojenia. Do hlavičky sa vypíše správa „connection error“ upozorňujúca na problém so spojením.

Môže nastať aj prípad, kedy telefón ostane pripojený a aplikácia sa snaží prečítať dáta, avšak zariadenie prestalo dáta prijímať. Vtedy sa uplatní časovač plnenia každej vyrovnávacej pamäti, ktorý ak sa neobnoví príznak zaplnenia buffera každých 15s, ukončí prehrávanie a uzavrie spojenie. Ošetrený je aj prípad, že by sa dočasne znížila rýchlosť pripojenia zoslabnutím signálu bezdrôtového pripojenia a to vytvorením časovej rezervy 2000ms medzi spustením prvého prehrávania a čítaním dát do pamäte. To umožňuje načítavať vyrovnávaciu pamäť o 2000ms dlhšie než sa v skutočnosti prehrá tá predošlá. Ďalšou z výnimiek je tá, ak sa nejakým pričinením ani s touto časovou rezervou nepodarí naplniť vyrovnávaciu pamäť, tak okrem toho, že vyhlási chybu, že nie je možné prehrávať hudobný stream, tak nedovolí aplikácií ani začať napĺňať druhý buffer a uzavrie sa spojenie.

Po každej zo spomínaných možných situácií je nutné dať prehrávače, ich príznaky a tiež sieťové spojenia, vyrovnávacie pamäte a ich príznaky, do pôvodnej podoby. To znamená korektne uzavrieť obe vlákna a predísť zlyhaniu aplikácie pri opätovnom pripájaní.

2.9.4. Záznam, prenos a príjem videa

V tejto časti aplikácie sa opäť nachádza vytvorenie prehrávača, ktoré síce prehráva podporovaný obsah, ale to len na mobilnom zariadení, ktoré dovoľuje pripojenie na sekundárnu kameru zadaným príkazom, prípade či vôbec zariadenie sekundárnu kameru obsahuje. Pri akomkoľvek probléme pri zavádzaní kamery na displej sa zobrazí chybové hlásenie oznamujúce nemožnosť zobrazenia.

Ďalší problém môže nastať, pri nadväzovaní spojenia s druhou stranou, kedy nemusí byť korektne zadaná IP adresa, alebo nemusí byť zariadenie dostupné či už z dôvodu, že nebola aplikácia spustená, alebo zariadenie nedostalo verejnú IP adresu, v prípade že nie je premostené, alebo v rovnakej sieti. Vtedy sa vypíše na obrazovku upozornenie o nedostupnosti druhého zariadenia. V prípade, že sa zahájil prenos a užívateľ sa odpojil, prípadne nastal iný problém pri prenose, je možné ukončiť príjem položkou „Back“ v menu prijímacej, alebo odosielacej strany.

2.10. Budúce rozšírenie aplikácie

Aplikácia multimediálneho prehrávača je síce spracovaná v súčasnej podobe nad rozsah zadania práce, avšak do úrovne v rámci časových možností daných na vývoj. Do budúcnosti je možné ju prepracovať do väčších detailov a tiež rozšíriť o niekoľko častí.

V prvom rade ide o riešenie sieťovej komunikácie, ktorá by sa stala univerzálnejšou a by bolo možné spojiť aj zariadenia iných značiek a typov, medzi ktorými z nezistených príčin v súčasnej podobe neprebieha komunikácia. Taktiež sa berie do úvahy možnosť lepšie kontrolovať stav spojenia a prenosu a zväčšiť možnosti užívateľa pri nahrávaní videa.

Aplikácia bola nad rámec zadania vyvíjaná aj smerom, kedy by bola schopná video dáta nie len nahrávať a následne odosielať, ale záznam odosielať v reálnom čase a prijímať v štandardizovaných prehrávačoch typu Windows Media Player, alebo VLC Player. Vývoj dospel k http spojeniu aplikácie prehrávača s programom VLC Player. Avšak posielané dáta je potrebné zložito prekódovávať a vkladať synchronizáciu, čo vďaka náročnosti z časového hľadiska nebolo možné.

Ďalšou realizáciou, ktorú nebolo možné z časového hľadiska uskutočniť je video stream. Ten však nedovoľuje dáta jednoducho prijímať, zapisovať do vyrovnávacej pamäte a následne prehrávať, ako tomu bolo u audio streamu. Pri budúcom rozširovaní je nutné rozobrať prijímané dáta a upraviť ich na formát, dovoľujúci prehrávanie v mobilnom zariadení. Eventuálne zvoliť iný typ streamu, napríklad pomocou protokolu RTSP.

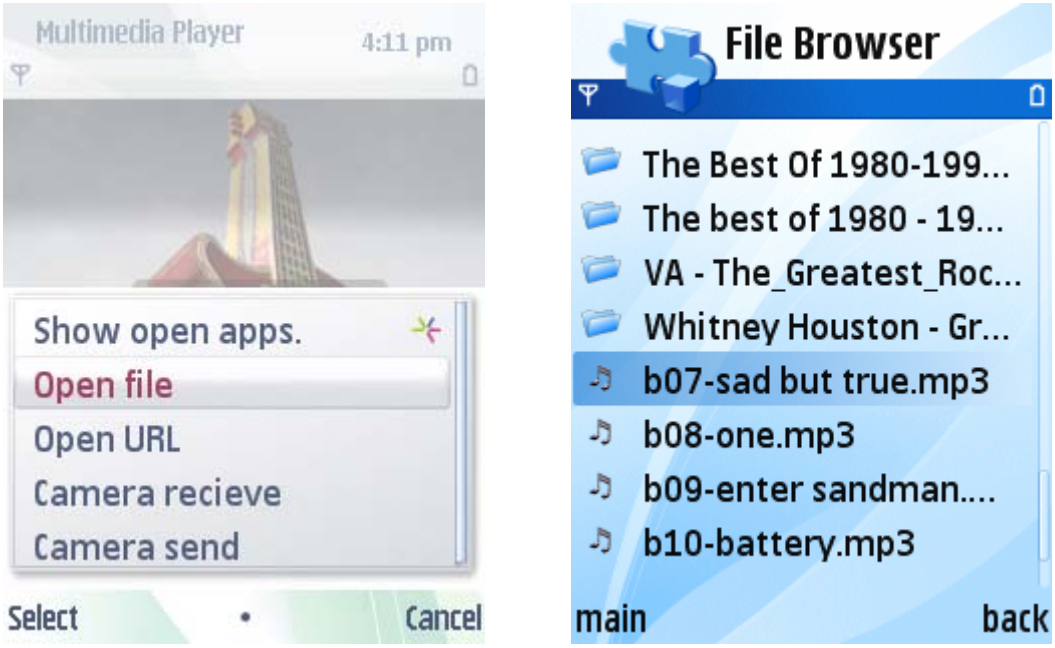
2.11. Popis multimediálneho prehrávača

Po nainštalovaní a spustení sa zobrazí úvodný obrázok v režime „fullscreen“ (celá obrazovka) s plným rozlíšením displeja mobilného telefónu Nokia rady S60 a to 240*320 pixelov. Obrazovka, ktorú je množné vidieť na Obr. 2.9 zotrvá zobrazená 4 sekundy.

Obr. 2.9. Úvodná obrazovka

Následne na to sa spustí hlavná obrazovka, na ktorej je zobrazené menu s položkami „Options“ (Možnosti) a „Exit“ (Ukončiť) a grafické pozadie. Výberom možnosti „Options“ sa zobrazia jeho položky. Sú to:

- Open file (Otvor súbor)
- Open URL (Otvor URL adresu)
- Camera recieve (Príjem zo vzdialenej kamery)
- Camera send (Posielanie záznamu kamery)

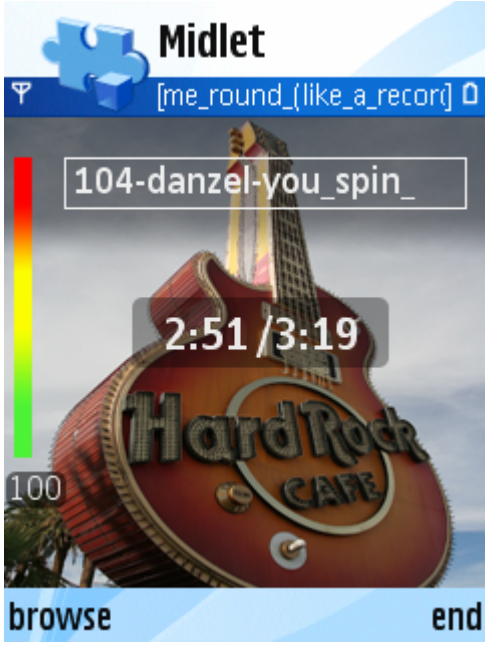


a) b)

Obr. 2.10. a) Hlavná obrazovka s menu b) Súborový manažér

Prvou možnosťou sa otvorí súborový manažér a užívateľ si prezerá obsah pamäťovej karty, kde sa šípkami pohybujem hore a dole v adresári, potvrdzovacím tlačidlom vstupuje do vybraného adresára, prípadne otvorí vybraný súbor. Návrat o úroveň vyššie zabezpečuje pravé tlačidlo s funkciou späť („back“). Návrat na úvodnú obrazovku sa realizuje ľavým „main“ tlačidlom. Po vybratí súboru sa otvorí buď prehrávač audia, alebo videa. Podľa toho aký formát bol zvolený. Pri otvorení skladby sa na hlavnej obrazovke zobrazí ukazovateľ hlasitosti, názov skladby a odohraný a celkový čas. Funkcie hlavných tlačidiel sa nezmenia. Stredové tlačidlo slúži na zastavenia a opätovné spustenie (play/pause) a šípkami hore a dolu sa ovláda hlasitosť prehrávača. Pri otvorení video súboru sa video zobrazí na čiernom pozadí v režime „fullscreen“. Na ľavej strane ukazuje hlasitosť v rozmedzí 0 až 100, na pravej strane „playing“ pri prehrávaní a po zastavení „paused“. Hlasitosť a tiež zastavenie a spustenie sa ovláda rovnako ako u audia. Prechod z režimu celej obrazovky a naspäť je realizovaný tlačidlom „c“.

a)

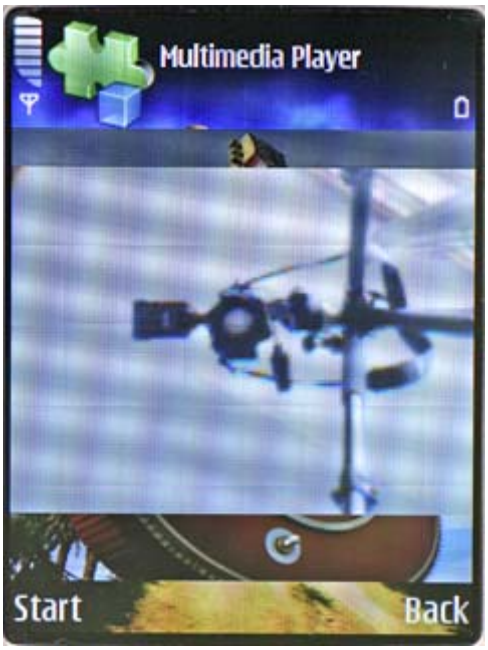


b)

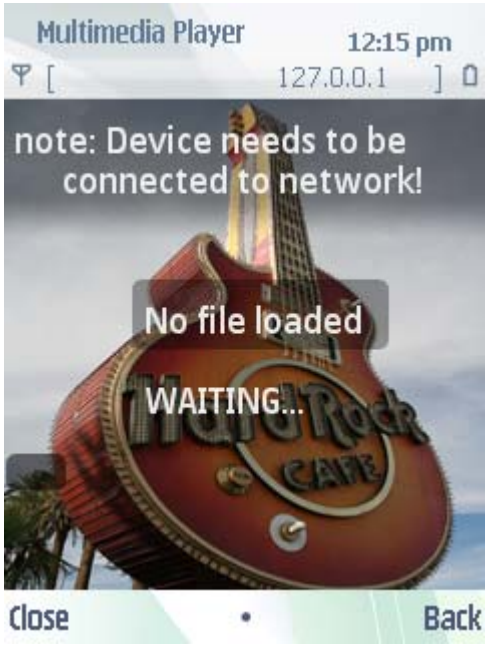
Obr. 2.11. a) Prehrávač videa b) Prehrávač audia

Výberom nasledujúcej položky „Open URL“ v zozname možností sa zobrazí formulár, kde sa zadá adresa internetového rádia (audio servera). Adresu je možné uložiť do zoznamu označením políčka „Save“, prípadne označiť v zozname už uloženú adresu a označením políčka „Open from saved“ ju použiť. Tiež je možná kombinácia všetkých prvkov. To znamená, že je vyplnená adresa, že je označené políčko „Save“ a tiež políčko „Open from saved“. Vtedy sa zadaná adresa nepoužije, ale uloží a otvorí sa adresa zo zoznamu. Spustenie zvolenej adresy sa vykoná stlačením tlačidla „Open“. Naopak opustenie formuláru tlačidlom „Back“.

Voľbou poslednej z možností „Camera send“ sa zobrazí obraz z sekundárnej kamery, teda tej, ktorá sa používa pri video hovoroch a je umiestená nad displejom. Tento obraz sa voľbou „Start“ začne zaznamenávať a voľbou „Stop“ sa záznam ukončí (Obr. 2.12). Voľbou „send“ sa zadadá IP adresa mobilného zariadenia, kde musí byť spustená táto aplikácia, ale v režime „Camera recieve“ a ktorá v tomto režime adresu vypíše. Po zadaní IP adresy sa voľbou „Send to“ záznam odošle na druhé mobilné zariadenie.



a)



b)

Obr. 2.12. a) Záznam z kamery b) Príjem záznamu

Spomínaná voľba „Camera recieve“ zobrazí stav prijímania. V prvom rade musí byť užívateľ pripojený do siete a potom spustiť tlačidlom „Recieve“ naslúchanie. Vtedy sa v hornej časti obrazovky vypíše IP adresa zariadenia, ktorú odosielateľ zadá. Po prijatí súboru sa spustí prehrávanie vo video prehrávači.

Po zvolení funkcie pravého tlačidla v hlavnom okne „Exit“ sa aplikácia ukončí. Ostanú však uložené záznamy IP adries z „Open URL“ a tiež hlasitosť prehrávania.

2.11.2. Chybové hlásenia

Chybové hlásenia môžu nastať pri zadávaní a výbere nesprávnych súborov, adries a iného zo strany užívateľa. Alebo zo strany sieťovej, kedy je možných aspektov zlyhania viacero. V nasledujúcich riadkoch je vysvetlenie situácií a k nim priradené chybové hlásenia.

- „The SOURCE is missing!“ – hlásenie, ktoré sa zobrazí užívateľovi v prípade, že pamäťová karta nie je vložená, prípadne nie je dostupná a upozorní na chýbajúci zdroj dát.
- „Selected FOLDER is empty!“ – hlásenie oznamujúce, že požadovaný adresár v súborovom manažéri je prázdny.

- „Unsupported file FORMAT!“ – vybraný súbor je nesprávneho formátu, teda iný než mp3 alebo mp4 s kompresiou MPEG-4 a rozlíšením väčším než 320x240 pixelov.
- „Can not OPEN the file!“ – vybraný súbor nie je možné otvoriť. Je to možné z dôvodu, že je poškodený, alebo nie je prístupný pre čítanie.
- „Can not OPEN URL address!“ – upozorňuje, pri otváraní zadanej URL adresy na nedostupné sieťové pripojenie, alebo na neplatnú adresu. Môže to nastať aj v prípade, že je nedostupný požadovaný server.
- „Not valid content“ – hlásené v prípade, že dáta audio streamu prijaté zo servera nie sú správneho formátu. Môže byť spôsobené chybou v prenose, prípadne nepodporovaným obsahom.
- „Connection lost“ – Hlásenie upozorňujúce užívateľa, že sa dostal z dosahu sieťového pripojenia, alebo došlo k prerušeniu. Prerušenie mohlo byť spôsobené medzi mobilným zariadením a sieťou, alebo niekde v ceste od servera, alebo sa odmlčal samotný server.
- „Can not connect to other device“ – upozornenie v prípade, že sa nepodarilo pripojiť aplikácií na druhé zariadenie, pri odosielaní video sekvencie. A to v prípade, že nebola zadaná správna adresa druhého mobilného zariadenia, druhá strana nemá spustenú aplikáciu, prípadne aplikácia nie je v móde „waiting...“ kedy očakáva pripojenie.
- „Video successfully SENDED“ – nejedná sa o chybové hlásenie ako v predošlých prípadoch, ale o oznámenie, že zaznamenaná video sekvencia bola úspešne odoslaná a prijatá druhým zariadením.

3. ZÁVER

Cieľom práce bolo, zoznámiť sa s možnosťami návrhu mobilných aplikácií a navrhnúť vhodnú realizáciu aplikácie pre mobilné zariadenie schopnú prijímať a prehrávať multimediálny obsah s využitím platformy Java ME. Prvým cieľom bolo teda vybrať vhodné prostredie, ktoré podporuje platformu J2ME a čo najviac skvalitňuje a uľahčuje prácu a vývoj. Jedným z najvhodnejších kandidátov bol NetBeans IDE, ktoré sa mimo iné vyvíja v Českej Republike. Spomedzi nespočetného množstva použiteľných zariadení som v rámci možností vybral zariadenie, ktoré svojimi schopnosťami najviac vyhovuje použitiu pre prehrávanie multimédií a to Nokiu na platforme Symbian S60.

Vývoj aplikácie bol realizovaný spočiatku na vývojovom kite pre zariadenia pod operačným systémom Symbian S60 a to na verziách FP1 a FP2. Následným využívaním prekrývania audio stôp, zachytávaním videa cez kameru a nutnosťou spojiť dve aplikácie v jednej sieti, vývoj a testovanie aplikácie prebiehal priamo na zariadení Nokia N95, bez možnosti krokového sledovania behu aplikácie. Tento mobilný telefón splňuje všetky požiadavky, ktoré boli kladené na úroveň aplikácie.

V práci sa nachádza popis každej časti vytvorenej aplikácie. Prvou je vytvorený audio prehrávač, ktorý umožňuje prehrať hudobný súbor z pamäťovej karty zariadenia. Táto časť je z aplikácie najuniverzálnejšia, pretože hudobné súbory formátu mp3 dokáže v dnešnej dobe prehrať už takmer každé moderné zariadenie. Ďalšiu časť tvorí prehrávač, ktorý otvorí video súbor a prehrá ho na obrazovke. Táto možnosť je už obmedzená len na niektoré typy a tiež výber a formát súboru závisí na vybranom zariadení. Aplikácia popisovaná v práci je ladená na spomínanú Nokiu N95 a podporuje súbor s koncovkou „.mp4“ formátu MPEG-4. V práci je ďalej popísaný princíp pripájania do siete za účelom prijatia dát zo vzdialeného servera. Taktiež ich spracovanie a následné prehratie. Touto časťou sú splnené podmienky zadania.

Nad rámec je v práci riešený algoritmus prístupu na video zariadenie v telefóne, kde sa z hľadiska jednoduchšieho prístupu volila sekundárna kamera. Video z kamery je zaznamenávané a je popísané riešenie jeho prenosu do druhého zariadenia. Medzi niekoľkými možnosťami sa z pohľadu flexibility a možnej vzdialenosti zvolilo pripojenie do siete ethernet a prípadná vzdialenosť je riešená prenosom súboru cez sieť internet.

Celá aplikácia bola vyvinutá pre mobilný telefón Nokia N95, ktorý v súčasnosti predstavuje jedného z najvhodnejších kandidátov na aplikáciu tohto zamerania.

LITERATÚRA

[1] PUMPRLA, Ondřej. *Sudoku na mobilním telefonu*. [s.l.], 2007. 46 s. Vysoké učení technické v Brně. Bakalářská práce.

[2] LACO, Dušan. *Správa financií pomocou mobilného telefónu*. [s.l.], 2006. 77 s. České vysoké učení technické. Bakalářská práce.

[3] GOYAL, Vikram. *Pro Java MMAPI*. [s.l.] : [s.n.], c2006. 267 s. ISBN 1-59059-639-0.

[4] JÁCHYM, Jakub. *Co je to JAVA?* [online]. 2007 [cit. 2008-10-24]. Dostupný z WWW: <http://clanky.katalog-notebookov.sk/slovnik-pojmov-notebooky/1331-co-je-to-java/>.

[5] ALBAHARI, Joseph. *Threading in C#*. [online], 2009. 77 s. O'Reilly Media. Dostupný z WWW: <http://www.albahari.com/threading/threading.pdf>.

[6] PIROUMIAN, Vartan. *Wireless J2ME™ Platform Programming*. [s.l.] : [s.n.], 2002. 400 s. ISBN 0-13-044914-8.

[7] RIGGS, Roger; TAIVALSAARI, Antero; PEURSEM, Jim Van; HUOPANIEMI, Jyri; PATEL, Mark; UOTILA, Aleksi. *Programming Wireless Devices with the Java 2 Platform, Micro Edition, Second Edition*. [s.l.] : [s.n.], 2003. 464 s. ISBN 0-321-19798-4.

[8] WELLS, Martin J. *J2ME™ Game Programming*. [s.l.] : [s.n.], 2004. 803 s. ISBN 1-59200-118-1.

[9] MORRISON, Michael. *Sams Teach Yourself Wireless Java with J2ME in 21 Days*. [s.l.] : [s.n.], 2001. 549 s. ISBN 0-672-32142-4.

[10] MAHMOUD, Qusay. *Learning wireless Java*. [s.l.] : [s.n.], 2001. 262 s. ISBN 0-59600-243-2.

ZOZNAM SKRATIEK

J2ME	Java 2 Micro Edition – platforma jazyka Java určená k programovaniu mobilných zariadení
CDC	The Connected Device Configuration – konfigurácia výkonnejších mobilných zariadení
CLDC	The Connected Limited Device Configuration – konfigurácia mobilných telefónov
PDA	A personal digital assistant – osobný digitálny asistent
MIDP	Mobile Information Device Profile – profil mobilného zariadenia
JAD	Java Application Description – štandardný textový súbor s koncovokou „.jad“
JAR	Java Archive – súbor obsahujúci skompilovaný program
MMAPI	Mobile Media Application Programming Interface – Rozhranie na programovanie multimediálnych aplikácií
IDE	Integrated Development Environment – integrované vývojové prostredie
mp3	MPEG-1 Audio Layer 3 – formát stratovej kompresie zvukových súborov, založený na kompresnom algoritme MPEG
mp4	Formát súborov špecifikovaný v medzinárodnom štandarde ISO/IEC MPEG-4 (ISO 14496-14)

PRÍLOHY

A OBSAH PRILOŽENÉHO DVD

Adresáre:

- Mobile player – projekt vytvorený podľa zadania diplomovej práce
- S60 3rd SDK FP1 – Vývojový kit SDK použitý na prehrávanie audia
- S60 3rd SDK FP2 – Vývojový kit SDK použitý na prehrávanie videa
- NetBeans IDE 6.1 – Vývojové prostredie aplikácie

Súbory:

- Diplomová práca – dokument s diplomovou prácou formátu pdf