UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Audio soubory a jejich meta data

Václav Harapes

Diplomová práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Václav Harapes**
Osobní číslo:	**I11376**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Základní formáty audio souborů, praktická analýza a příklady používání dostupných knihoven v C++**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Popis současných formátů audio souborů, hlavně mp3, ogg vorbis, wav a další včetně metody jejich komprese.
Řešitel navrhne a implementuje v aplikaci zobrazení informací o audio souborech, editaci uživatelských informací. Aplikace bude poskytovat komplexní možnosti pro správu hudební sbírky. Bude napsána v jazyce C++ s využitím multiplatformní knihovny QT4.
Řešitel navrhne teoretické další možnosti open source knihoven C++ v problematice práce s audio soubory.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

ŠEBESTA, Ondřej a David MORKES. MP3 a vše o něm. 1. vyd. Praha: Grada, 2001. ISBN 80-247-0013-1.
NOKIA CORPORATION AND/OR ITS SUBSIDIARIES. Online Reference Documentation [online]. 2011 [cit. 2012-10-03]. Dostupné z: http://doc.qt.digia.com/

Vedoucí diplomové práce: **Ing. Jaroslav Štroch**
Katedra informačních technologií

Datum zadání diplomové práce: **31. října 2012**
Termín odevzdání diplomové práce: **17. května 2013**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2012

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Cerhýnkách dne 15. srpna 2013 Václav Harapes

Poděkování

Rád bych poděkoval Ing. Jaroslavu Štrochovi, vedoucímu mé diplomové práce, za jeho cenné rady a čas. Dále bych ještě chtěl poděkovat svému otci za duševní i materiální pomoc po celou dobu mého studia.

Anotace

Tato práce se zabývá v současnosti používanými tagovacími formáty. Klade si za cíl teoreticky popsat a srovnat tyto formáty z různých kritérií. V další části popisuje vybrané open source knihovny pro práci s meta daty těchto souborů. V praktické části je implementována aplikace pro správu sbírky audio souborů s pomocí popsaných knihoven. Mimo to se v práci ještě nacházejí dvě malé, avšak ne nepodstatné kapitoly – první pojednává o vlastních formátech souborů a druhá, spíše rešeršní, popisuje použití dalších knihoven pro práci s audio soubory.

Klíčová slova

audio, hudba, tag, tagovací knihovny, mp3, wav, ogg vorbis, flac

Title

Audio files and their meta data

Annotation

This thesis deals with current audio tagging formats. The goal is to theoretically describe and compare wide audio formats. There is also overview of some of the most currently used open source tagging libraries. In practical part is implemented application for management music collection with some of the mentioned libraries. There are also two little chapters – the first describes some audio formats and the second one is researching another open source library used to work with audio files.

Keywords

audio, music, tag, tagging libraries, mp3, wav, ogg vorbis, flac

Obsah

Seznam zkratek

BSD	Berkley Software Distribution
COM	Component Object Model
FLAC	Free Lossless Audio Codec
GCC	GNU C Compiler
GNU	GNU's Not Unix!
GPL	General Public Licence
ISO	International Organization for Standardization
LAME	LAME Ain't an Mp3 Encoder
LGPL	Lesser General Public License Library General Public License
MPEG	Moving Picture Experts Group
MPL	Mozilla Public License
POSIX	Portable Operating Systém Interface
XML	Extensible Markup Language

Seznam obrázků

Seznam tabulek

Úvod

V posledních letech se stále více rozmáhá poslech hudby na cestách z nejrůznějších přenosných zařízení od specializovaných tzv. MP3 přehrávačů až po mobilní telefony. Nároky uživatelů ne úložné místo se stále zvyšují, navíc některé mobilní telefony dnes již nedisponují slotem pro paměťovou kartu a výrobci počítají s tím, že uživatelé budou mít svá data v cloudu. Vzhledem k rychlostem mobilních přenosů je komprese těchto dat, byť z jiných důvodů než kdysi, stále důležitá. Novější standardy, jakými jsou např. 3G nebo LTE, sice mají dostatečnou přenosovou kapacitu pro audio, ale operátoři používají tzv. FUP – omezují množství dat, které uživatel může přenést za delší období.

Multimediální přehrávače dnes již počítají s vyplněnými meta daty v audio souborech. S organizací pomocí složek již většinou nepočítají. Tím se do popředí dostává problém, jak spravovat tato meta data.

Práce se zaměřuje primárně na vytvoření aplikace, která umožní tato meta data získat z názvů složek a souborů, a to jen pomocí jednoduchých, uživatelsky zadaných, výrazů. Protože implementace zápisu vlastních meta dat je dostupná ve formě svobodně dostupných knihoven, bude aplikace napsána tak, aby některou, nebo některé z nich využívala.

Vlastní teoretická část pak v omezené míře popíše vybrané formáty audio souborů a datových struktur uchovávajících jejich meta data.

Vzhledem k různým překladům do českého jazyka budou všechny názvy uvedeny v angličtině a v poznámce budou přeloženy. Ve zbytku práce pak bude použit tento přeložený tvar.

1 Popis formátů souborů

V průběhu času bylo vyvinuto mnoho formátů pro ukládání zvuku. Jedním z hlavních rozdílů je typ použité komprese. Existují dva hlavní druhy komprese, a to ztrátová a bezztrátová. Jak již názvy napovídají, liší se hlavně tím, zda při kódování vynechávají některá data či nikoli. Ztrátová komprese bývá navržena tak aby posluchač nerozeznal rozdíl mezi originálem a zkomprimovanými daty.

Mělo by zde být zdůrazněno, že zde budou primárně popisovány nejrozšířenější formáty kontejnerů (souborů) nikoli kompresní kodeky. Vlastní kodek není z hlediska přístupu k meta datům většinou významný, a proto bude komprese popsána jen u formátu MP3. Jiné kodeky buď využívají podobný princip, nebo by jejich popis byl až příliš rozsáhlý.

1.1 MP3

MP3, známý též jako MPEG 1 resp. MPEG 2 Layer III je jeden z nejznámějších a nejpoužívanějších formátů pro ukládání zvukových dat, a to hlavně mezi laickou veřejností. Byl vytvořen společností Moving Picture Experts Group. [1]

MP3 soubory mají hlavičku o velikosti 32 bitů, a popisuje ji tabulka 1. Podrobnější popis lze nalézt literatuře, vzhledem ke komplexnosti jsou uvedeny pouze základní informace.

Tabulka 1 – Hlavička MP3 souboru [2], [3]

Název pole	Od bitu	Délka pole [b]	Poznámka
FrameSync	0	11	Vždy 1111 1111 111
AudioVersionID	11	2	Verze audia
LayerIndex	13	2	Vrstva
ProtectionBit	15	1	Obsahuje CRC součet
BitrateIndex	16	4	Index datového toku
SamplingRateIndex	20	2	Index vzorkovací frekvence
PaddingBit	22	1	Výplň
PrivateBit	23	1	Může být použit pro potřeby konkrétní aplikace.
ChannelMode	24	2	Režim kanálů
ModeExtension	26	2	Rozšířený režim, pouze pro Joint Stereo
CopyrightBit	28	1	Copyright
OriginalBit	29	1	Originál
Emphasis	30	2	Informace pro reekvivalizaci, téměř nevyužíváno

Některé položky nemusí být na první pohled úplně jasné. Jedná se hlavně o verzi audia, ta může nabývat čtyř hodnot, kde jednotlivé znamenají 00 → MPEG v2.5 (neoficiální verze), 01 → rezervováno, nepoužívá se, 10 → MPEG v2 a 11 → MPEG v1.

Položka udává, jaká MPEG vrstva se nachází v souboru, konkrétně 00 → rezervováno, nepoužívá se, 01 → Layer III, 10 → Layer II a 11 → Layer I. Pro MP3 soubory je typická vrstva III.

Index datového toku a vzorkovací frekvence závisí na počtu kanálů, resp. na konkrétní verzi audia. Podrobněji rozepsány nebudou.

Režim kanálů uvádí, zda se jedná o stereo (00), joint stereo (01), dual channel stereo (10) – jedná se o jiný způsob uložení stereofonní zvuku – nebo mono (11).

Rozšířený režim taktéž závisí na verzi audia, a z prostorových důvodů nebude popsán.

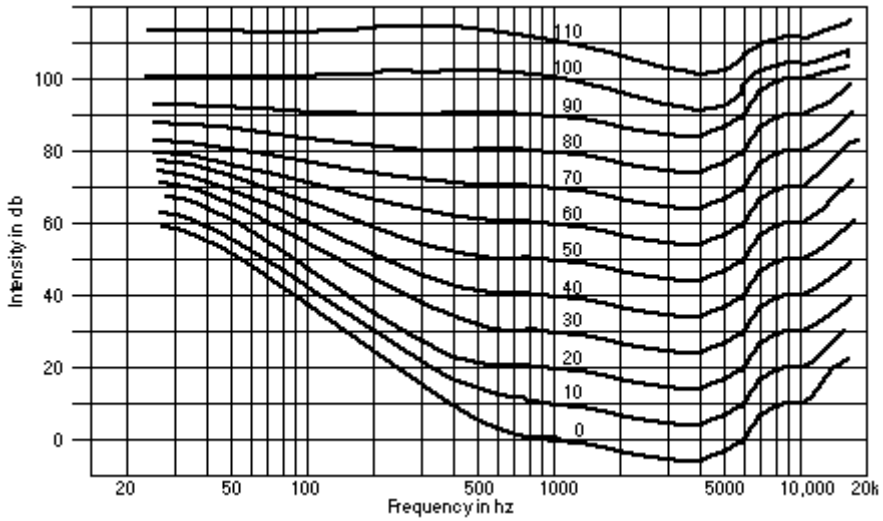
Komprese

MP3 využívá několik technik pro redukci datového toku, mezi ně patří hlavně:

- odstranění zvuků ležících mimo frekvenční rozsah lidského ucha,
- odstranění zvuků, které jsou překryty jinými zvuky,
- Joint Stereo kódování a
- Huffmanovo kódování. [4]

Odstranění zvuků ležících mimo frekvenční rozsah lidského ucha

Lidský sluch nevnímá zvuk o stejné intenzitě a různé frekvenci jako stejně intenzivní. Tuto závislost vyjadřují Fletcher-Munsonovy křivky (viz obrázek 1–1).



Obrázek 1–1 – Fletcher-Munsonovy křivky[1]

Pro běžný poslech není třeba ukládat zvuky umístěné mimo hranice slyšitelnosti, protože je posluchač stejně neuslyší. [4]

Odstranění zvuků, které jsou překryty jinými zvuky

Tato metoda je založena na tom, že v hlasitých pasážích nejsou slyšet tiché zvuky a proto tyto mohou být vypuštěny. [4]

Joint Stereo kódování

I tato technika je založena na nedokonalosti lidského ucha. Je použitelná pouze u dvou a vícekanálového zvuku.

Skládá se ze dvou hlavních částí.

1. Intensity Stereo využívá toho, že lidské ucho neumí u nízkých ani vysokých frekvencí dostatečně rozlišit místo původu zvuku. Tím pádem mohou tyto frekvence být kódovány a přehrávány pouze monofonně.

[1] zdroj: http://www.webervst.com/fm.htm

2. Mid/Side stereo je oproti výše uvedenému intensity stereo použitelné na všech frekvencích, ale zas je použitelné pouze v případě, že v levém a pravém kanálu je podobný signál. Potom může místo každého kanálu být uložen pouze jejich součet a rozdíl, což zabírá ve výsledném souboru méně místa. Při přehrávání jsou z těchto složek zpětně dopočítány původní hodnoty pro oba signály.

Tato technika má jednu významnou slabinu – pokud je Joint Stereo použito, musí být použito v celém souboru, což snižuje možnost jeho využití. [4]

Huffmanovo kódování

Huffmanovo kódování je použito až na konci komprese, a je to jediný bezztrátový kompresní algoritmus použitý při kompresi.

Základní princip Huffmanova kódování spočívá v tom, že často opakujícím se sekvencím jsou přiřazeny krátké kódy, kdežto méně opakující se sekvence jsou vyjádřeny delšími kódy. Velkou výhodou této metody je rychlost kódování i dekódování.

U zvukových souborů je tato metoda relativně efektivní a dokáže zmenšit výsledná data téměř o 20 %. [4]

Zásobník bitů

Kromě výše uvedených technik je používána ještě jedna, která ale není počítána mezi kompresní. Jedná se o to, že části vyžadující vyšší datový tok, si tento tok mohou „půjčit“ od částí, které plně svůj datový tok nevyužily.

Tato technika umožňuje snížit nevýhodu ohledně limitu maximální šířky rámce (320 $kb \cdot s^{-1}$), ale v určitých případech ani toto nestačí a při kompresi dochází k degradaci kvality skladby. [4]

1.2 Waveform audio

Waveform audio je formát od společnosti Microsoft, který se rozšířil primárně díky operačnímu systému Windows. Formát je založen na obecném standardu RIFF (Resource Interchange File Format) se specializací pro ukládání audia. [5]

Tento formát používá metodu LPCM (Linear pulse-code modulation – lineární pulzně-kódová modulace). Ta je založena na jednoduchém principu, kdy vstupní signál je vzorkován v pravidelných intervalech. Na rozdíl od ostatních PCM metod využívá LPCM lineární kvantizaci. Ač Waveform audio používá bezztrátovou kompresi, tak výstupní signál při přehrávání WAV souboru nebude identický s originálem právě z důvodu převodu analogového signálu na digitální (a při přehrávání zas zpět na analogový). [5]

Základní formát souboru se skládá ze dvou částí („fmt “ a „data“) a popisuje jej tabulka 2 (hodnoty uvedené v uvozovkách indikují řetězce). Většina datových položek v souboru je ukládána pomocí Little Endian.

Tabulka 2 – Hlavička souboru Waveform audio [6], [7]

Název pole	Endian	Od bajtu	Délka pole [B]	Poznámka
ChunkID	big	0	4	„RIFF“
ChunkSize	little	4	4	Velikost souboru bez prvních dvou polí
Format	big	8	4	„WAVE“
Subchunk1ID	big	12	4	„fmt “
Subchunk1Size	little	16	4	16 (pro PCM)
AudioFormat	little	20	2	1 pro LPCM, jiná hodnota v případě komprimovaného souboru
NumChannels	little	22	2	Počet kanálů
SampleRate	little	24	4	[Hz]
ByteRate	little	28	4	SampleRate · NumChannels · BitsPerSample ÷ 8
BlockAlign	little	32	2	NumChannels · BitsPerSample ÷ 8
BitsPerSample	little	34	2	8, 16…
Subchunk2ID	big	36	4	„data“
Subchunk2Size	little	40	4	NumSamples · NumChannels · BitsPerSample ÷ 8
Data	little	44		

Všechny položky jsou dostatečně popsány přímo v tabulce, a proto nepotřebují další upřesnění.

Kromě toho ještě může soubor obsahovat jiné části, konkrétně „fact“ (informace o kompresi), „wavl“ (specifikace alternativních slnt a data částí), „slnt“ (uvádí délku předchozích dat, slouží ke kompresi) apod. [6]

Waveform audio podporuje různé kompresní kodeky, a to např. i dříve uvedený MP3.

1.3 Ogg

Ogg je moderní kontejner s plně otevřenou specifikací od společnosti Xiph.org. Ogg je často používán v kombinaci se ztrátovým kodekem Vorbis, a to tak často, že se nesprávně používá označení Ogg Vorbis, ale Vorbis může být použit i s kontejnerem FLAC, a existuje i verze využívající kodek WebM od firmy Google. Zajímavostí je, že nejnovější verze Ogg je 0. [8]

Kontejner Ogg podporuje mimo uložení více datových proudů i detekci chyb a pravidelné ukládání časového razítka, vhodné pro vyhledávání v souborech s proměnnou šířkou datového toku. Podporuje několik audio či video proudů (které mohou být komprimovány různými kodeky), titulky a meta data v jednom souboru. [8]

Na rozdíl od předchozích uvedených formátů není Ogg kontejner navržen jako monolitický, ale skládá se z částí – tzv. stránek. Každá stránka může být velká až 64 kB, ale běžně se vyskytují stránky o velikosti do 8 kB. Dělení na stránky umožňuje přeskočit chybnou část, což je vhodné zejména při streamování. [9]

Stránky se skládají ze segmentů. Každá stránka jich může obsahovat až 255. Segmenty jsou logicky slučovány do paketů, což jsou nejmenší logické celky, jež má smysl dekódovat. [9]

Podrobnější formát stránky uvádí tabulka 3. Všechny číselné hodnoty jsou ve formátu little endian.

Tabulka 3 – Stránka Ogg kontejneru [10]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Header	0	27	Hlavička
Segment Table	27	0–255	Segmentová tabulka
Data		Max. 65 025	Data

Zde je potřeba popsat jednotlivé prvky. Hlavička má velikost 27 bajtů a podrobně ji popisuje tabulka 4. Segmentová tabulka udává délky jednotlivých paketů. Jejich počet je definován v hlavičce. Ve zbytku stránky se nacházejí vlastní data. Těmi mohou být jak meta data, tak vlastní zvuková data. [9]

Tabulka 4 – Hlavička stránky Ogg kontejneru [10]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Capture Pattern	0	4	„OggS“
Stream Structure Version	4	1	„\0“ pro současnou v0
Header Type Flag	5	1	Příznaky: abc0 0000
Absolute Granule Position	6	8	Časový údaj o poloze
Stream Serial Number	14	4	Číslo proudu
Page Sequence No	18	4	Číslo stránky
Page Checksum	22	4	Kontrolní součet stránky
Page Segments	26	1	Počet segmentů

I zde je potřeba podat podrobnější vysvětlení. Capture Pattern slouží pro synchronizaci přehrávače, např. pokud je soubor nebo stream částečně poškozen.

Příznaky uvádějí podobnější informace o paketech ve stránce. Pokud je nastaven příznak

a jedná se o poslední stránku logického proudu bitů,
b jedná se o první stránku logického proudu bitů,
c pak paket pokračuje z předchozí stránky. [10]

Časový údaj o poloze v podstatě určuje délku souboru od jeho počátku do posledního paketu této stránky (pakety pokračující na další stránce se nepočítají). Konkrétní implementace závisí na daném kodeku. [10]

Jak bylo popsáno výše, tento kontejner umožňuje ukládat více logických proudů. Typickým příkladem využití může být více audio stop, což bývá využíváno spíše u filmů. Aby mohl dekodér poznat, která stránka patří ke kterému proudu, byla zavedena položka Číslo proudu. [10]

Číslo stránky, jak již název pole napovídá, udává číslo stránky od začátku souboru. Toto je vhodné např. při streamování, kdy dekodér může určit, zda nebyla některá stránka ztracena, či zda stránky nebyly přijaty v jiném pořadí. [10]

Kontrolní součet je spočítán algoritmem CRC-32. Součet je vypočítán z celé stránky včetně celé hlavičky, a to i včetně tohoto pole. Aby jej bylo možné vůbec vypočítat je v době výpočtu celé pole pokládáno za vynulované. [10]

Poslední pole v hlavičce udává, na kolik segmentů se stránka dělí. Těch může být od nuly do 255. Pokud má stránka 255 segmentů, pak to znamená, že paket pokračuje

v následující stránce. Pokud by náhodou logický proud končil přesně v 255. segmentu, pak následující stránka musí mít nula segmentů. Pokud je segmentů méně než 255, pak paket končí na této stránce a v další začíná nový (nebo se dekodér nachází na konci souboru). [10]

2 Popis formátů uložení metainformací

Meta data neboli tagy jsou datová struktura uložená na začátku (resp. za hlavičkou), případně na konci souboru. Původní formáty tagů obsahovaly pouze jednotky základních informací (např. interpret, album, název, stopa a žánr), moderní tagy již obsahují desítky, často umožňují definovat i uživatelské. K nejznámějším formátům patří ID3v1, ID3v2, APE tagy a Xiph komentáře.

Některé formáty umožňují ukládat i časové údaje. Pro jejich popis budou použity samo vysvětlující písmena, ale pro jistotu zde bude tento formát popsán. Pokud je písmeno uvedeno vícekrát, značí to minimální délku údaje, pokud není hodnota tak dlouhá, je zleva doplněna nulami na příslušnou délku. Význam jednotlivých písmen je následující:

- r – rok,
- M – měsíc (číselně),
- d – den měsíce,
- h – hodina,
- m – minuta,
- s – sekunda.

Jakýkoli jiný znak je považován za řetězcovou konstantu.

Tagem bude v této kapitole myšlena celá struktura, rámec bude označovat jednu konkrétní položku (informaci).

2.1 ID3v1

ID3v1 tag může být kromě MP3 kontejneru použit i v jiných formátech. Vytvořil jej Eric Kemp v roce 1996. Z důvodu dodržení zpětné kompatibility byla tato struktura umístěna v posledních 128 bajtech souboru. Její podrobný formát znázorňuje tabulka 5. [11]

Tabulka 5 – Formát ID3v1 tagu

Název pole	Od bajtu	Délka pole [B]	Poznámka
Preambule	0	3	„TAG“
Title	3	30	Název
Artist	33	30	Interpret
Album	63	30	
Year	93	4	Rok vydání alba
Comment	97	30	Komentář
Genre	127	1	Žánr

V případě, že nějaká položka má kratší hodnotu než uvedenou délku je zbytek pole vyplněn znakem „\0“. V opačném případě musí být daná hodnota oříznuta na maximální povolenou délku. [11]

Všechny hodnoty jsou uložené čistě textově vyjma Žánru. Každému žánru bylo přiřazeno číslo, které jej reprezentuje. Seznam podporovaných žánrů uvádí příloha C. [11]

Za povšimnutí stojí, že i rok je implementován jako čtyřbajtová textová hodnota, ačkoli by stačily pouze dva bajty, ve kterých by byl rok uložen jako 16bitový integer (ať už znaménkový nebo bezznaménkový).

Později byl ID3v1 tag vylepšen Michaelem Mutschlerem, který doplnil položku ukládající číslo stopy. Z důvodu požadavku na zpětnou kompatibilitu nebylo dodatečné místo získáno z položky Rok, ale byl o dva bajty zkrácen Komentář. Jeden z těchto bajtů byl využit pro uložení znaku „\0“, díky kterému aplikace pro práci s ID3v1 tagy přestaly se čtením pole, a v druhém bajtu byla uložena vlastní informace – Číslo stopy. Podobnější formát uvádí tabulka 6. [11]

Tabulka 6 – ID3v1.1 tag

Název pole	Od bajtu	Délka pole [B]	Poznámka
Preambule	0	3	„TAG“
Title	3	30	Název
Artist	33	30	Interpret
Album	63	30	
Year	93	4	Rok vydání alba
Comment	97	28	Komentář
Oddělovač	125	1	„\0“
Track	126	1	Číslo stopy
Genre	127	1	Žánr

2.2 ID3v2

Nepružnost ID3v1 tagů je řešena ve verzi ID3v2. V době psaní této práce je k dispozici nejnovější verze 2.4.0 z 1. listopadu 2000. ID3v2 tag je na rozdíl od předchozí verze navržen jako pružný a rozšiřitelný. Toto je také první verze, která oficiálně podporuje UTF-8. Formát se používá hlavně v MP3 souborech, ale může být využit i jinde. [12]

Je preferováno, aby tag byl umístěn na začátku souboru z důvodu využití těchto informací přehrávači při online streamování. Nicméně může být i rozdělen na dvě části – v první části jsou pak umístěny základní informace – minimálně hodnota „SEEK“, která určuje přesnou pozici druhé části tagu, která je až za vlastním audio daty. Třetí možností je umístit celý tag až za audio data, ale před jakékoli další tagy. [12]

V této kapitole budou často zmiňovány pojmy jako synchronizace (anglicky synchronization), desynchronizace (unsynchronization), případně synchsafe integer. Jedná se o to, že aplikace nepodporující ID3v2 tagy mohou považovat kombinaci 11 nastavených bitů za sebou jako začátek vlastních audio dat, a tedy „přehrávat“ i tag. Toto je vyřešeno tak, že za každý bajt s hodnotou 0xFF je přidán bajt navíc s hodnotou 0x00. Toto není vhodné, pokud poslední bajt libovolného pole má hodnotu 0xFF. [12]

V případech hodnot s pevnou délkou (typicky velikosti) tento postup také není vhodný. V tom případě se používají tzv. synchsafe integery. Nejvýznamnější bit každého bajtu je povinně nulový, a tím pádem se nemůže za sebou vyskytnout více jak sedm nastavených bitů. Sice maximální hodnota 32bitové integeru klesne z přibližně 4,3 miliard na 268 milionů, nicméně při typickém použití nenastává praktické omezení. [12]

Tabulka 7 – Formát ID3v2 tagu

Název pole	Délka pole [B]	Poznámka
Header	10	Hlavička
Extended header	Proměnná	Rozšířená hlavička (volitelná)
Frames	Proměnná	Rámce
Padding	Proměnná	Výplň (volitelná)
Footer	10	Patička (volitelná)

Základní strukturu uvádí tabulka 7. Je nutné zmínit, že tag nemůže zároveň obsahovat výplň i patičku. [12]

Tabulka 8 – Hlavička ID3v2 tagu

Název pole	Od bajtu	Délka pole [B]	Poznámka
File identifier	0	3	„ID3“
Version	3	2	Verze, 0x04 0x00 pro v. 2.4.0 apod.
Flags	5	1	Příznaky, bitově abcd0000
Size	6	4	Velikost tagu uložená pomocí tzv. synchsafe integeru

Téměř všechny položky hlavičky jsou samo vysvětlující, ale u příznaků a velikosti bude uveden komentář. Hlavička obsahuje čtyři příznaky (a další čtyři příznaky jsou rezerva do budoucna, v současné verzi musí být vynulovány), které uvádí, zda tag:

a obsahuje desynchronizované rámce.
b obsahuje rozšířenou hlavičku.
c je v experimentální verzi.
d obsahuje patičku. [12]

Po hlavičce může volitelně následovat rozšířená hlavička (viz tabulka 9). Pokud soubor obsahuje rozšířenou hlavičku, pak její nejmenší možná velikost je šest bajtů. [12]

Tabulka 9 – Rozšířená hlavička ID3v2 tagu

Název pole	Od bajtu	Délka pole [B]	Poznámka
Extended header size	0	4	Velikost rozšířené hlavičky, uložená pomocí tzv. synchsafe integeru
Number of flag bytes	4	1	Velikost příznaků v bajtech
Extended flags	5	1	Příznaky, bitově 0abc0000
Flag's data	6		Data příznaků

Příznaky a jejich data budou opět vysvětleny podrobněji. Data příznaků jsou seřazeny ve stejném pořadí, v jakém jsou uvedeny v příznacích. Pokud není příznak nastaven, pak jeho data nesmí být uvedena. Každá datová položka příznaku obsahuje v prvním bajtu velikost dat v bajtech a v následujících bajtech vlastní data. [12]

Pokud je nastaven příznak „a“, znamená to, že tento tag je pouze aktualizací staršího tagu nalezeného v souboru. Tento příznak nemá žádná data, pouze je uvedena, že velikost dat je nulová. [12]

Příznak „b“ značí, zda soubor obsahuje svůj vlastní hash pro detekci poškození. Metoda CRC-32 je aplikována na všechna data mezi hlavičkou a patičkou (čili včetně výplně), ale bez rozšířené hlavičky. Velikost CRC-32 součtu se ukládá jako 35bitový synchsafe integer, velikost těchto dat je tedy 5 bajtů. [12]

Poslední použitý příznak, „c“, uvádí některá omezení. Velikost jeho dat je 1 bajt a další bajt uvádí velikost). Bitově jej lze zapsat jako „ddeffghh“. [12]

a Tento příznak omezuje velikost tagu (00 → nejvíce 128 rámců a velikost tagu menší nebo rovna jednomu MB, 01 → nejvíce 64 rámců a 128 kB, 10 → nejvíce 32 rámců a 40 kB a 11 → nejvíce 32 rámců a čtyři kB). [12]

b Příznak udává, jakými způsoby mohou být uloženy texty. Pokud je příznak nastaven, pak všechny řetězce musí být v ISO-8859-1 (Latin 1) nebo UTF-8. Pokud je nulový, pak tag není ohledně kódových stránek nijak omezen. [12]

c Omezuje maximální velikost každého jednotlivého textového řetězce (00 → bez omezení, 01 → nejvíce 1024 znaků, 10 → nejvíce 128 znaků a 11 → nejvíce 30 znaků). [12]

d Předposlední příznak udává, v jakých formátech mohou být vložené obrázky. Pokud je nastaven, tak tyto obrázky mohou být pouze typu PNG nebo JPEG, jinak pro ně žádné omezení neplatí. [12]

e Poslední příznak uvádí omezení ohledně velikosti obrázků (00 → bez omezení, 01 → žádný z obrázků není větší jak 256×256 px, 10 → nejvíce 64×64 px a 11→ přesně 64×64 px. [12]

Po rozšířené hlavičce následují teprve vlastní datové rámce tagu. Formát každého rámce uvádí tabulka 10. [12]

Tabulka 10 – Formát ID3v2 rámce [12]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Frame ID	0	4	Identifikátor rámce
Size	4	4	Velikost dat (synchsafe)
Flags	8	2	Příznaky, bitově 0abc0000 0d00efgh
Data	10		Vlastní informace

Rámce mohou být uloženy v libovolném pořadí, konkrétní hodnotu určuje jejich identifikátor, typicky se vyskytujícími zástupci mohou být:

- TALB – název alba / filmu / představení,
- TCOM – skladatel,
- TRCK – číslo stopy,
- TIT2 – název písně… [13]

Pro potřeby konkrétní aplikace je možné využít identifikátory začínající písmeny „X“, „Y“ nebo „Z“, a to i bez nastavení příznaku experimental v hlavičce. Cílem práce rozhodně není vypisovat všechny identifikátory, ty čtenář může nalézt kdekoli na internetu, případně přímo v použité literatuře. [12]

Příznaky se dělí do dvou skupin:

- příznaky hlavičky rámce (a–c),
- formát dat rámce (d–g).

a Příznak „a“ značí, že pokud je identifikátor pro danou aplikaci, která mění tag, neznámý, pak by měl být celý rámec odstraněn. [12]
b Tento příznak je podobný příznaku „a“, ale platí pro aplikace, které mění jakoukoli část souboru. [12]
c Pokud je příznak nastaven, pak by neměl být rámec měněn, protože mohou nastat určité problémy (např. nebude souhlasit CRC-32 součet). V případě změny tohoto rámce musí být tento bit vynulován. [12]
d Nastavením tohoto příznaku je určeno, že rámec patří do větší skupiny rámců. K rámci se přidá jeden bajt, který je shodný pro všechny rámce v dané skupině.
e Příznak udává, zda je na obsah rámce použita komprese. Pokud ano, pak musí být příznak „h“ nastaven též. [12]
f Pokud je obsah rámce šifrován, pak tento příznak musí být nastaven, stejně jako příznak „h“. Šifrování by mělo být provedeno až po kompresi (pokud je použita). [12]

g Pokud je tento příznak nastaven, pak jsou data v rámci považována za desynchronizovaná. Příznak „h“ by měl být při použití tohoto příznaku nastaven také, ale není to povinné. [12]

h Příznak „h“ se nazývá indikátor délky dat. Pokud je použita komprese, šifrování nebo synchronizace, pak se skutečná velikost dat nemusí shodovat s uloženými daty. V tom případě je tento bit nastaven na „1“ a k rámci je přidán další údaj o skutečné velikosti ve formátu synchsafe integeru. [12]

Data mohou být ukládány jako řetězce, číselné hodnoty, nebo časové údaje. Ty jsou ukládány v podmnožině formátů, které definuje norma ISO 8601. Patří mezi ně:

- rrrr,
- rrrr-MM,
- rrrr-MM-dd,
- rrrr-MM-ddThh,
- rrrr-MM-ddThh:mm a
- rrrr-MM-ddThh:mm:ss. [12]

Po rámcích následuje výplň. Všechny bity musí být nulové. Používá se pro budoucí rozšíření tagu tak, aby nemusel být uložen celý soubor znovu. Je zakázáno, aby konkrétní soubor měl výplň i patičku zároveň. [12]

Pro zvýšení rychlosti nalezení tagu při prohledávání od konce (viz ID3v1) se používá patička. Ta má stejný obsah jako hlavička, jen s mírně upraveným identifikátorem – místo „ID3“ obsahuje „3DI“. Pokud je ID3v2 tag na konci souboru, musí být patička uvedena. [12]

2.3 APE Tag

APE tag je první plně svobodný formát popsaný v této práci pro ukládání meta informací. Původně byl vyvinut pro formát Mousepack, ale je používán i v jiných formátech, typicky Monkey’s Audio, WavPack či optimFROG. Kromě těchto spíše exotických formátů jej lze využít např. i v MP3 souborech. [14]

Existují dvě verze – v1 a v2. Z hlediska čtení a zápisu dat jsou téměř identické, druhá verze přinesla podporu UTF-8 (první verze podporovala pouze ASCII). Druhá verze též přidává hlavičku, která usnadňuje nalezení vlastního tagu. [14]

Je doporučeno APE tag umisťovat na konec souboru. Potom musí obsahovat alespoň patičku. V případě, že je umístěn na začátek souboru, což ale není doporučeno, potom musí mít alespoň hlavičku. V případě, že je použit u MP3 kontejneru, potom je doporučeno jej umístit na konec audio dat – před ID3v1 tag. Je možné celý tag nebo jeho část umisťovat do souboru i vícekrát, to je vhodné např. při online streamování. [14]

V této práci bude popsána primárně druhá verze, jak vyplývá z výše zmíněného, obě verze se liší pouze minimálně. Případné rozdíly budou okomentovány. Strukturu uvádí tabulka 11. [14]

Tabulka 11 – Základní formát APE tagu [14]

Název pole	Od bajtu	Délka pole [B]	Poznámka
APE Tag Header	0	32	Hlavička
APE Tag Item 1–n	32	Min 10	Jednotlivé datové položky
…			
APE Tag Footer		32	Patička

Jednotlivé datové položky je doporučeno řadit dle důležitosti, ale není to povinné. [14]

Hlavičku, o velikosti 32 bajtů, ukazuje tabulka 12.

Tabulka 12 – Formát hlavičky APE tagu

Název pole	Od bajtu	Délka pole [B]	Poznámka
Preambule	0	8	„APETAGEX“
Version Number	8	4	Verze
Tag Size	12	4	Velikost tagu
Item Count	16	4	Počet datových položek tagu
Tags Flags	20	4	Globální příznaky
Reserved	24	8	Pro budoucí použití, musí být vynulováno

Některá pole budou podrobněji vysvětlena. Specifikace v době psaní práce uvádí pouze dvě možné verze. Pokud je v příslušném poli uvedeno číslo 1 000 jedná se o verzi 1.0, pokud 2 000 pak je tag ve verzi 2.0. [14]

Velikost tagu zahrnuje pouze datové položky a patičku. Velikost hlavičky není k tagu přičítána z důvodu kompatibility obou verzí. [14]

Globální příznaky jsou novinkou v APEv2, v APEv1 musí být tato sekce vynulována. Ač je pro ně rezervováno 32 bitů, tak druhá verze APE tagu využívá pouze 6 bitů (31. (a), 30. (b), 29. (c), 2.–1. (d), 0. (e)), ostatní (28.–3.) musí být vynulovány. [14]

a Tento příznak uvádí, zda tag obsahuje hlavičku. [14]

b Příznak „b“ má podobnou funkci, ale pro patičku – také určuje její (ne)přítomnost, nicméně tento příznak je implementován inverzní logikou (0 → obsahuje, 1 → neobsahuje). [14]
c Udává, zda jde o hlavičku (1), či patičku (0). [14]
d Dvoubitový příznak d určuje typ datových položek (00 → text v kódování UTF-8, 01 → informace v binárním formátu (např. vložené obrázky), 10 → informace odkazují externě, ať už absolutně či relativně, lokálně nebo vzdáleně, 11 → v současné verzi nespecifikováno). Odkazy na externí zdroje jsou také ve formátu UTF-8. [14]
e Udává, zda je tag určen pouze pro čtení.

Po hlavičce již následují jednotlivé datové položky. Každá z nich má strukturu, kterou uvádí tabulka 13.

Tabulka 13 – Formát položky APE tagu [14]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Size of the Item Value	0	4	Velikost hodnoty
Item Flags	4	4	Příznaky
Item Key	12		Název pole
0x00		1	„\0“
Item Value			Hodnota

I zde je formát na první pohled zřejmý, ale některá pole budou upřesněna. Velikost hodnoty je udávána v bajtech, a je v ní uložena pouze velikost hodnoty (poslední položky) a nic jiného. [14]

Příznaky jsou naprosto shodné s příznaky v hlavičce uvedené výše s jediným rozdílem – příznaky v hlavičce platí pro všechny datové položky, příznaky v položce platí pro konkrétní položku. [14]

Název pole je řetězec o délce od dvou do 255 znaků obsahující libovolné znaky s ASCII kódem od 32 do 126. Typicky se vyskytují názvy o délce 2–16 znaků, která se skládají pouze z malých a velkých písmen, číslic a mezer. Vzhledem ke kompatibilitě s jinými tagovacími formáty nejsou dovoleny názvy „ID3“, „TAG“, „OggS“ a „MP+“. Specifikace uvádí, že názvy jsou závislé na velikosti písmen, ale zároveň zakazuje používat názvy, které se liší jen velikostí písmen, a zároveň doporučuje, aby software pro práci s tagy nerozlišoval velikost písmen. [14]

Vlastní názvy jsou uživatelsky mnohem přívětivější než ty, které jsou použité u ID3v2 tagu. Příkladem jsou:

- Album – název alba,
- Composer – skladatel,
- Track – číslo stopy,
- Title – název písně… [14]

Hodnota může obsahovat různá data, např.

- celočíselné hodnoty,
- číselné hodnoty s plovoucí desetinnou čárkou,
- UTF-8 řetězce,
- výčet UTF-8 řetězců (oddělený znakem „\0“),
- binární data… [14]

Datum je i zde ukládán v souladu s normou ISO 8601, ale na rozdíl od ID3v2 je datum a čas oddělován nikoli znakovou konstantou „T“, ale mezerou, čili typický zápis využívá jednu z následujících možností:

- rrrr,
- rrrr-MM,
- rrrr-MM-dd,
- rrrr-MM-dd hh,
- rrrr-MM-dd hh:mm nebo
- rrrr-MM-dd hh:mm:ss. [14]

Patička APE tagu je stejná hlavička, s výjimkou jednoho příznaku, kterým již byl zmíněn, proto nebude popsána zvlášť. [14]

2.4 Vorbis comment

Vorbis comment (též vorbis komentář, xiph komentář apod.) je další z otevřených formátů pro ukládání meta dat ve zvukových souborech. Původně byl vytvořen pro formát Vorbis (kontejner ogg), ale rozšířil se i do jiných formátů společnosti Xiph.org – FLAC, Theora (video formát) nebo Speex. Dle autorů by v tagu měly být pouze základní údaje, ostatní patří do vlastního logického proudu, jako příklad uvádějí formát XML, který umožňuje lepší strojové parsování. [15]

Vzhledem k tomu, že implementace v jednotlivých kontejnerech mají mírně rozdílné hlavičky, bude v této kapitole primárně popisován Ogg Vorbis komentář. Specifikaci ostatních hlaviček lze dohledat v oficiální dokumentaci. [15]

Vlastní struktura s informacemi se nachází v druhém paketu od začátku souboru. Na rozdíl od prvního paketu s hlavičkou souboru se v druhém paketu nemusí nacházet jen jedna struktura – tag. Hlavička (umístěná v prvních 16 bajtech) je povinná, i když soubor neobsahuje vůbec žádné meta data. Všechny číselné hodnoty jsou uloženy v podobě bezznaménkového 32bitového integeru způsobem little endian. [15]

Tabulka 14 – Formát Vorbis komentáře [15]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Vendor String Length	0	4	Velikost identifikátoru formátu v B
Vendor String	4	8	Identifikátor formátu („vorbis “)
Number of comment fields	12	4	Počet položek
Comment field 1–n	16		Položky 1–n
…			
Framing bit		1 bit	Ukončovací bit „1“

Téměř všechny hodnoty jsou dostatečně popsány v tabulce. Ukončovací bit musí být nastaven na jedničku a soubor jej musí obsahovat. Pokud jej neobsahuje, nebo je nulový, pak se jedná o poškozený soubor. [15]

Položka Vorbis komentáře má také jednoduchou strukturu a popisuje jej tabulka 15.

Tabulka 15 – Položka Vorbis komentáře [15]

Název pole	Od bajtu	Délka pole [B]	Poznámka
Comment field length	0	4	Velikost pole v B
Field name	4	Velikost pole	Jméno pole
Delimiter			„=“
Field value			Hodnota pole

Jména polí jsou, podobně jako u APE tagů, také uživatelsky přívětivá. Pro srovnání budou i zde uvedeny stejné pole jako u předchozích formátů, konkrétně:

- ALBUM – název alba,
- základní výčet polí neobsahuje položku skladatel,
- TRACKNUMBER – číslo stopy,
- TITLE – název písně… [15]

Vorbis komentáře neobsahují tak vysoký počet základních polí jako tomu je u dříve popsaných. Nicméně vzhledem k tomu, že dovolují definovat vlastní pole, není problém tyto položky dodefinovat. Problémem tedy není technické omezení ale spíše nekompatibilita mezi aplikacemi různých výrobců. [15]

2.5 Srovnání

Na závěr kapitoly bude uvedeno srovnání. Autor se zde snažil vybrat ke srovnání takové hodnoty, které jsou objektivní, nicméně jsou tu i některé, které vycházejí z autorových zkušeností a názorů a jsou tedy subjektivní. Typickým příkladem je složitost implementace, která by určitě měla být zahrnuta ve srovnání, ale není možné ji určit exaktně. Srovnání uvádí tabulka 16.

Tabulka 16 – Srovnání tagovacích formátů

	ID3v1	ID3v2	APE Tag	Vorbis komentář
Podpora UTF-8	Ne	Ano	Ano	Ano
Otevřenost	Ne	Ne	Ano	Ano
Podpora více kontejnerů	Ano	Ano	Ano	Ano
Počet uložených informací	Malý	Velký	Velký	Střední
Vlastní informace	Ne	Ano	Ano	Ano
Komplexnost	Malá	Velká	Velká	Velká
Složitost implementace	Nízká	Vysoká	Střední	Nízká
Celkové hodnocení	6	10	11	12

Logické hodnoty (ano / ne) přidávají k celkovému hodnocení 1 resp. 0 bodů. Hodnoty malý / nízký přidávají jeden bod, střední dva body a velký / vysoký tři body (u složitosti implementace jsou bodové hodnoty obráceně).

Jak lze z hodnocení vidět, tak starší formáty ID3 nedosáhly tak vysokého hodnocení jako modernější otevřené formáty. Nicméně je nutno zdůraznit, že ve srovnání není žádný moderní uzavřený formát, a to primárně kvůli tomu, že autor nenašel dostatečně podrobnou dokumentaci jejich struktur. I spousta otevřených knihoven, které budou popsány později, umožňuje v těchto formátech číst a zapisovat pouze základní informace.

3 Přehled C / C++ open source knihoven pro práci s tagy

Na internetu je k dispozici několik knihoven pro manipulaci s tagy otevřeným zdrojovým kódem. Budou zde popsány pouze ty, které něčím autora zaujaly. Některé z popsaných knihoven pak budou použity v praktické části práce.

3.1 TagLib

TagLib je open source knihovna distribuovaná pod GNU LGPL a MPL licencí. V době psaní práce je na adrese http://taglib.github.io/ k dispozici verze 1.8 (vydána 5. září 2012). Pyšní se podporou širokého spektra formátů od nejběžnějších ID3 tagů (v obou verzích), APE tagů, FLAC tagů, Xiph komentářů až po MP4 a WMA tagy od společnosti Microsoft. Bohužel podpora posledních dvou, i vzhledem k tomu že existují proprietární implementace, je omezená. [16]

Obsahuje jak abstraktní rozhraní pro práci s obecným multimediálním souborem, tak možnost přistoupit ke konkrétnímu typu na neabstraktní úrovni – takto je řešeno například vkládání obrázků do souborů. [16]

Mezi další výhody patří podpora Unicode, přehledná dokumentace a nezávislost na žádné jiné knihovně (kromě obecně používané knihovně zlib určené ke kompresi dat). [16]

Knihovna využívá vlastní implementaci datového typu String místo použití standardních datových typů, které nabízí C++ (std::string, resp. std::wstring pro Unicode data). Výhodou je, že s běžnými typy stringů v C++ pracuje transparentně, ba dokonce obsahuje makra pro konverzi z / na QString (implementace stringu ve frameworku Qt). [16]

Zajímavostí pro české a slovenské vývojáře je, že do aktivního vývoje této knihovny se zapojuje i Slovák Lukáš Lalinský. [16]

Příklad použití:

```
TagLib::FileRef f("song.mp3");
TagLib::Tag* tag = f.tag();

TagLib::String artist = tag->artist(); // get artist

tag->setAlbum("Album"); // set album „Album“

f.save();
```

3.2 libmtag

Knihovna libmtag není sama o sobě tagovací knihovnou, ke své funkci spolupracuje s knihovnou TagLib popsanou na předchozích řádcích, a funguje jako její

frontend pro jazyky C, Python a Ruby. Je distribuovaná pod licencí GNU/LGPL. [13] Tato knihovna podtrhává popularitu knihovny TagLib. Je k dispozici na stránce http://code.google.com/p/libmtag/. [17]

Příklad použití:

```
mtag_file_t* f;
mtag_tag_t* tag;

f = mtag_file_new("song.mp3");
tag = mtag_file_tag(f);

char* artist = mtag_tag_set(tag, "artist"); // get artist

mtag_tag_set(tag, "album", "Album"); // set album „Album“

mtag_file_save(file);
mtag_file_free(file);
```

3.3 MediaInfo

MediaInfo patří je další otevřená knihovna pro práci s meta daty audio souborů. V době psaní práce je k dispozici verze 0.7.64. Knihovna je k dispozici na adrese http://mediaarea.net/cs/MediaInfo uvolněná pod licencí BSD. [14]

Knihovna je určena pro čtení informací nejen ze zvukových, ale i z video souborů. Unicode je podporováno. [14] Bohužel v současné verzi neumí zapisovat žádné informace, v API jsou připraveny signatury metod, ale v dokumentaci je upozornění, že nejsou implementovány, a zatím by neměly být používány. [15]

Knihovna nabízí velmi širokou podporu kontejnerů, a to od těch nejrozšířenějších až k těm méně známým. Z hlediska formátů meta dat jsou k dispozici obligátní ID3 tagy obou verzí, APE tagy, Xiph.org komentáře a v omezené míře i RIFF a Windows Media tagy. [14]

Tuto knihovnu je nutné linkovat proti knihovnám ZenLib a zlib.

Příklad použití:

```
MediaInfo mi;
mi.Open("song.mp3");
MediaInfoLib::String artist = mi.Get(MediaInfoLib::Stream_General, 0,
TEXT("Performer")); //get artist
mi.Close();
```

3.4 id3lib

Knihovna id3lib je další ze známých tagovacích knihoven. Je uvolněna pod licencí GNU LGPL. Jak již její název napovídá je použitelná pouze pro ID3 tagy, a to v obou verzích. Jak sami její vývojáři uvádějí, jejím cílem je plná podpora ID3v2 standardů a nezávislost na konkrétní platformě. V době psaní práce je na stránce http://id3lib.sourceforge.net/ k dispozici nejnovější verze 3.8.3 z 2. února 2003. [16] Vývoj této knihovny se zastavil a s novými kompilátory ji často nelze ani zkompilovat.

Knihovna je k dispozici pro jazyky C, C++, a je ji dokonce možné využít jako COM knihovnu (pomocí id3com wrapperu) v jazycích, které toto rozhraní podporují – typicky jazyky využívající platformu .NET. [16]

Všechny řetězce jsou interně zpracovány v kódování Unicode. [16]

Knihovna není závislá na žádné externí knihovně, což usnadňuje vlastní kompilaci. [16]

Příklad použití:

```
ID3_Tag tag("song.mp3");
ID3_Frame* frame;
ID3_Field* field;
char data[1024];
frame = tag.Find(ID3FID_ARTIST);
field = myFrame->GetField(ID3FN_TEXT);
field->Get(data, 1024); // get artist
frame = tag.Find(ID3FID_ALBUM);
field = myFrame->GetField(ID3FN_TEXT);
field->Set("Album"); // set album „Album“
tag.Update();
```

3.5 Srovnání

Po přehledu knihoven následuje jejich srovnání. Hodnoceny budou jak objektivní hlediska (množství podporovaných formátů, podpora Unicode…), tak subjektivní hlediska, jakými jsou jednoduchost rozhraní, či kvalita dokumentace. Srovnání uvádí tabulka 17.

Tabulka 17 – Srovnání open source tagovacích knihoven

	TagLib	libmtag	MediaInfo	id3lib
Podpora Unicode	Ano	Ano	Ano	Ano
Množství podporovaných formátů	Vysoké	Vysoké	Vysoké	Nízké
Zápis i čtení meta dat	Ano	Ano	Ne	Ano
Další informace o souboru (datový tok, vzorkovací frekvence,…)	Ano	Ano	Ano	Ne
Snadnost použití	Vysoká	Vysoká	Střední	Střední
Kvalita dokumentace	Vysoká	Vysoká	Nízká	Vysoká
Celkové hodnocení	12	12	8	8

Hodnocení probíhalo podobně jako v předchozí kapitole. U položek ano / ne byl přičten jeden bod za „ano“, nula bodů za „ne“. U ostatních položek byl zvolen systém nízké (jeden bod) / střední (dva body) / vysoké (3 body). Není náhoda že TagLib a libmtag dosáhly stejného výsledku – libmtag je knihovna pro jazyk C postavené na TagLib, což jen potvrzuje kvalitu této knihovny.

V hodnocení nebyl brán v potaz fakt, že knihovna MediaInfo je teprve ve vývoji, a částečně i proto ji chybí kvalitní dokumentace. Pokud se jejím autorům podaří splnit plány, pak by se z ní mohla stát velmi kvalitní knihovna a přímý konkurent TagLibu.

4 Ostatní knihovny pro práci s audio soubory

V této kapitole bude provedena rychlá rešerše ostatních open source knihoven pro práci s audio soubory. Čtenář by ji spíše měl považovat za jakýsi rozcestník než za podrobný popis.

4.1 Lame

LAME je knihovna pro konverzi souborů do formátu MP3. Je dostupná na adrese http://lame.sourceforge.net/index.php pod licencí GNU LGPL. V době psaní práce je nejnovější dostupná verze 3.99 z října roku 2011. LAME lze zkompilovat na asi dvou desítkách platforem, ale bohužel ve výčtu nebyla uvedena žádná mobilní platforma. [17]

Přímo její autoři vyzdvihují její rychlost a kvalitu, která se vyrovná, ne-li přímo porazí komerční konkurenci. To potvrzuje velké množství aplikací, které tuto knihovnu používají. [17]

4.2 MAD: MPEG Audio Decoder

MAD: MPEG Audio Decoder je knihovna distribuovaná pod licencí GNU GPL. případně i pod komerční licencí. Slouží k dekódování MPEG I, MPEG II a prakticky i MPEG 2.5 souborů. Podporuje vrstvy Layer I, Layer II i Layer III (kterou využívají MP3 soubory). Knihovna je k dispozici na stránce http://www.underbit.com/products/mad. [18]

Jde o novou implementaci ISO standardu, a proto, jak uvádějí její autoři, není zatížena chybami vzniklými změnami tohoto standardu. Při výpočtech jsou využívány pouze celočíselné datové typy, což zrychluje výpočty, obzvlášť na platformách bez matematického koprocesoru. [18]

4.3 ccAudio

Knihovna ccAudio je framework pro práci s audio soubory pod licencí GNU GPL v2. Cílem jeho autorů je vytvořit pro C++ takový framework, jakým jsou dále popsané audiofile a snadfile pro klasické C. Nejnovější verze 2.0.5 z 27 března 2011 je k dispozici na adrese http://savannah.gnu.org/projects/ccaudio. Knihovna je určena pro operační systémy využívající rozhraní POSIX (např. GNU/Linux) a Microsoft Windows. [19]

Knihovna převádí interní formát jednotlivých souborů do podoby vzorků, se kterými poté může být manipulováno pomocí různých operací. [19]

4.4 Audiofile

Audiofile je GNU LGPL knihovna primárně určená pro jazyk C. Nejnovější verze 0.3.6 je dostupná na adrese http://audiofile.68k.org/. [20]

Knihovna převádí audio soubor do podoby vzorků, s kterými pak lze manipulovat v paměti bez ohledu na vnitřní strukturu jednotlivých formátů. Ze známějších formátů jsou podporovány AIFF, WAVE, nebo FLAC. [20]

4.5 libsndfile

Knihovnu libsndfile je možné použít pro rychlé konverze mezi audio soubory, které ukládají audio jako soubor vzorků. Poslední verze 1.0.25 z 13. července 2011 je dostupná na adrese http://www.mega-nerd.com/libsndfile/. Je distribuována pod licencí GNU LGPL. Knihovnu lze zkompilovat na 32 i 64bitových Windows, dále na Linuxu, Mac OS X, ale i Androidu, a několika dalších. [21]

Mezi rozšířené podporované formáty patří RIFF, AIFF a FLAC. Kromě konverze mezi audio formáty nabízí i konverzi do souborů matematického softwaru Octave. [21]

5 Implementace

V praktické části bude implementována grafická aplikace pro správu tagů v souborech. Kromě toho bude umožňovat i různá dávková přejmenování souborů. Pro nízko úrovňovou práci se soubory budou použity některé z popsaných knihoven, které budou zaváděny jako pluginy.

5.1 Prostředí

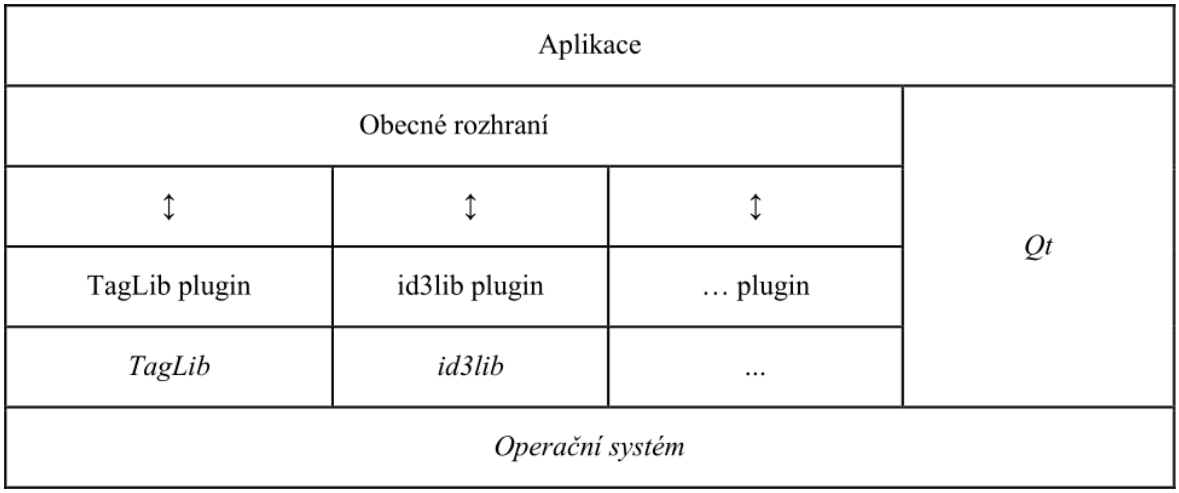
Pro implementaci bude použit programovací jazyk C++ a multiplatformní nadstavbová knihovna Qt 4. Tato kombinace má hned několik pozitiv – výsledné aplikace v C++ bývají většinou rychlejší než jejich kolegyně psané v jiných jazycích. [22]

Další výhodou je seznam podporovaných platforem, mezi něž patří desktopové Microsoft Windows, Linux, ale i mobilní zařízení se systémy Symbian, Android [23], nebo Apple iOS. To znamená, že jednou napsanou aplikaci je možno přeložit i pro jiný operační systém pouze s menšími úpravami grafického rozhraní tak, aby respektovalo designové zvyklosti dané platformy a nemátlo uživatele (platí hlavně pro mobilní operační systémy).

Pro kompilaci bude použit open source kompilátor GCC, resp. jeho varianta pro C++.

5.2 Návrh aplikace

Vzhledem k nemalému počtu open source knihoven pro práci s meta daty bude vytvořeno speciální rozhraní pro komunikaci mezi programem a knihovnou. Knihovny budou dynamicky zaváděny jako pluginy. To umožní přechod na jinou knihovnu bez nutnosti jakékoli úpravy programu – postačí napsat příslušný plugin. Obrázek 5–1 ukazuje schematicky celou aplikaci. Normálním řezem jsou vyznačeny komponenty, které bude nutné naprogramovat, kurzíva naznačuje knihovny třetích stran, nebo jiné, již hotové moduly. Na všechny tyto moduly bude dále v textu referováno jako „komponenty třetích stran“.



Obrázek 5–1 – Schéma aplikace

Zjednodušený pracovní proces aplikace se skládá z kroků:

1. otevření souboru (plugin),
2. přečtení dat (plugin),
3. zobrazení dat s možností editace (aplikace),
4. uložení dat (plugin),
5. zavření souboru (plugin).

V závorce je uvedena poslední komponenta před komponentami třetích stran, která se bude o danou činnost starat. Proces je kvůli přehlednosti velmi zjednodušen, a téměř libovolný krok vyjma otevření a uzavření souboru lze vynechat.

5.3 Návrh rozhraní

Z hlediska této kapitoly jsou nejdůležitější body 1, 2, 4 a 5. Z hlediska návrhu budou k implementaci potřeba nejméně dvě třídy.

- Třída Tag, což bude datová třída obsahující víceméně jen gettery a settery. Třída bude konvertovat datové typy (např. std::string na QString a obráceně). Typickou metodou takovéto třídy by mohlo být:

```
QString Tag::getArtist() {
    // getTag() vrací instanci třídy „Tag“ definované
    // konkrétní knihovnou
    return QString(this->getTag()->get(ARTIST));
}
```

- Třída Soubor, která se bude starat o otevření a uzavření vlastního souboru a předání instance třídy Tag.

Třída Tag

Jak již čtenář mohl pochopit z příkladu metody z třídy Tag, bude existovat asociace mezi komponentami knihovny a komponentami rozhraní. Z hlediska aplikace půjde dokonce o kompozici (ale ne už z hlediska knihovny). Mimo jiné to znamená, že knihovna a aplikace budou plně odstíněny, a nebudou spolu moci komunikovat jinak než přes toto rozhraní.

Při vlastní implementaci třídy Tag se nabízejí dvě resp. tři možná řešení, a to

a) samostatné metody – getArtist(), setAlbum("album")…
b) vícefunkční metody – get(ARTIST), set(ALBUM, "album")…

Postup a je z hlediska návrhu neflexibilní – při změně rozhraní je nutné znovu překompilovat jak plugin, tak vlastní rozhraní, které je součástí aplikace. Navíc metody budou obsahovat podobný kód. Výhodou je jednoduchost pro uživatele rozhraní, a plná možnost využití refaktorování.

Druhou možností je použít vícefunkční metody. Tato možnost může být rozdělena na dvě samostatné možnosti, a to podle toho zda budou pole identifikována položkou výčtového typu nebo konstantou (ať už řetězcovou či číselnou).

Položky výčtového typu mají výhodu ve své bezpečnosti. Při předání něčeho jiného než položky výčtového typu dojde ke zkompilování jen za použití explicitního přetypování. Na druhou stranu je nutno podotknout, že v C++ bývají výčtové typy definovány v hlavičkových souborech, a pokud by byl daný výčtový typ změněn, je možné že bude nutné znovu zkompilovat jak rozhraní (a aplikaci) tak plugin. V novém standardu C++11 již mohou být položky výčtového typu definovány až v souborech se zdrojovým kódem, ale Qt 4 není s novými možnostmi C++11 kompatibilní.

Druhou možností je využít konstanty pro identifikaci daného pole. Pomocí tohoto způsobu se lze vyhnout znovu kompilování aplikace při změně rozhraní.

Každopádně u dynamických metod nastává ještě jeden problém, konkrétně s návratovým typem. Moderní tagovací formáty umožňují ukládat řetězce, čísla, časové údaje i binární data (např. obrázky), ale jazyk C++ nedovoluje definovat více metod se stejným názvem, které se liší jen návratovým typem. Pro rozlišení návratových typů by musely být použity metody ve stylu getString(), getInt(), getData() apod. Tyto názvy metod nejsou z hlediska budoucího rozvoje také příliš vhodné, dle posledních doporučení by metody ani proměnné neměly mít v názvu svůj typ.

Po zvážení uvedených výhod a nevýhod byla zvolena koncepce samostatných metod.

Třída Soubor

Třída Soubor bude natolik jednoduchá, že bude popsána pouze stručně. Bude existovat asociace mezi příslušnou interní třídou knihovny a vlastní třídou Soubor. Rozhraní bude překládat operace, mezi které bude patřit:

- otevření souboru,
- získání instance třídy Tag,
- uložení souboru,
- zavření souboru.

Třída by mohla být v budoucnu rozšířena např. o smazání všech meta dat, získání podrobných informací o souboru, a to i obecných (datum modifikace, přístupová práva…)

Třída AudioVlastnosti

Mimo to může být rozhraní doplněno o třídu, která umožní získat další informace o souboru, kterými mohou být např. počet kanálů, datový tok, vzorkovací frekvence…

Závěr

Hlavní částí práce bylo popsat datové struktury, ve kterých jsou uchovávány meta data audio souborů (tagy). Ty byly nakonec porovnány s výsledkem, že APE tagy a Vorbis komentáře se snadno používají a mají velké možnosti. Naopak ID3v2 tagy z tohoto porovnání vyšly kvůli své složitosti nelichotivě. ID3v1 tagy by, vzhledem k jejich omezeným možnostem, měly být používány pouze v kombinaci s novějšími formáty kvůli kompatibilitě se staršími přehrávači.

V další části, která se již dotýká implementace, byla provedena analýza několika otevřených knihoven pro práci s těmito informacemi. Byly porovnány jak knihovny, které již nejsou ve vývoji, tak ty, které jsou stále aktivně vyvíjeny. Z porovnání vítězně vyšla knihovna TagLib, která se snadno používá a zároveň má široké možnosti. Knihovně MediaInfo bohužel chyběla podpora zápisu informací do souboru, ale je nutno vzít v potaz, že se jedná o novou knihovnu, která je ve vývoji. Naopak většinu jiných knihoven nešlo na operačním systému Windows 7 pomocí GCC zkompilovat.

Cílem praktické části byla implementace aplikace využívající právě tyto knihovny pro úpravu informací v audio souborech. Tyto knihovny jsou dynamicky zaváděny jako plugin. Výhoda tohoto řešení je, že pro použití jiné knihovny je nutné napsat jen plugin, který převede rozhraní knihovny na rozhraní aplikace. Vzhledem k problémům s kompilací knihoven byla použita jen knihovna TagLib. Aby mohlo být demonstrováno přepínání pluginů za běhu aplikace byl napsán ještě jeden plugin, který si zajišťuje operace se soubory sám. Bohužel tento plugin umí jen ID3v1.1 tagy.

Aplikace umožňuje kromě dávkového i čtení a zápis tag do jednotlivých souborů. Informace do tagů mohou být načteny z názvů souborů i složky, ve které se nacházejí, a naopak soubory i složka mohou být přejmenovány podle informací v tagu.

Literatura

1. **ŠEBESTA, Ondřej a MORKES, David.** *MP3 a vše o něm.* 1. vyd. Praha : Grada, 2001. 80-247-0013-1.

2. MPEG audio frame header (mp3 format). *DataVoyage.* [Online] 22. 12 1999. [Citace: 08. 07 2013.] http://www.datavoyage.com/mpgscript/mpeghdr.htm.

3. **WINDSZUS, Konrad.** MPEG Audio Frame Header. *Code Project.* [Online] 12. 05 2007. [Citace: 08. 07 2013.] http://www.codeproject.com/Articles/8295/MPEG-Audio-Frame-Header.

4. **BOUVIGNE, Gabriel.** MP3' Tech. *MP3' Tech.* [Online] (c) 1997–2001. [Citace: 03. 04 2013.] http://www.mp3-tech.org/.

5. WAVE Audio File Format with LPCM audio. *Digital Preservation.* [Online] 26. 10 2012. [Citace: 08. 07 2013.] http://www.digitalpreservation.gov/formats/fdd/fdd000002.shtml.

6. **Sonic Spot.** Wave File Format. *The Sonic Spot.* [Online] © 1999–2007. [Citace: 08. 07 2013.] http://www.sonicspot.com/guide/wavefiles.html.

7. **WILSON, Scott.** WAVE PCM soundfile format . *Center for Computer Research in Music and Acoustics.* [Online] 20. 01 2003. [Citace: 08. 07 2013.] https://ccrma.stanford.edu/courses/422/projects/WaveFormat/.

8. **Xiph.org.** Vorbis – Xiph.org. *Vorbis.* [Online] 20. 05 2013. [Citace: 29. 07 2013.] https://wiki.xiph.org/Vorbis.

9. —. Ogg bitstream overview. *Ogg Documentation.* [Online] (c) 1994–2005. [Citace: 29. 07 2013.] http://xiph.org/ogg/doc/oggstream.html.

10. —. Ogg logical bitstream framing. *Ogg Documentation.* [Online] (c) 1994–2005. [Citace: 29. 07 2013.] http://xiph.org/ogg/doc/framing.html.

11. ID3v1. *ID3.org.* [Online] 08. 10 2012. [Citace: 21. 07 2013.] http://id3.org/ID3v1.

12. **NILSSON, M.** ID3 tag version 2.4.0 – Main Structure. *id3.org.* [Online] 01. 11 2000. [Citace: 29. 07 2013.] http://id3.org/id3v2.4.0-structure.

13. —. ID3 tag version 2.4.0 – Native Frames. *id3.org.* [Online] 01. 11 2000. [Citace: 29. 07 2013.] http://id3.org/id3v2.4.0-frames.

14. Monkey's Audio. *HydrogenAudio.* [Online] 28. 06 2013. [Citace: 20. 07 2013.] http://wiki.hydrogenaudio.org/index.php?title=Monkey%27s_Audio.

15. **Xiph.Org.** vorbis.com. *vorbis.com.* [Online] Xiph.Org, © 1994–2008. [Citace: 29. 12 2012.] http://www.vorbis.com/.

16. **CONTRERAS, Felipe.** libmtag – Library for music tagging. *libmtag – Library for music tagging.* [Online] [Citace: 29. 12 2012.] http://code.google.com/p/libmtag/.

17. What is MediaInfo? *MediaInfo.* [Online] [Citace: 29. 07 2013.] http://mediaarea.net/en/MediaInfo.

18. More information about the programming interface. *MediaInfo.* [Online] [Citace: 29. 07 2013.] http://mediaarea.net/cs/MediaInfo/Support/SDK/More_Info.

19. id3lib. *id3lib – The ID3v1/ID3v2 tagging library.* [Online] [Citace: 20. 07 2013.] http://id3lib.sourceforge.net/.

20. About LAME. *LAME MP3 Encoder.* [Online] [Citace: 2013. 07 25.] http://lame.sourceforge.net/about.php.

21. **Underbit Technologies.** MAD: MPEG Audio Decoder. [Online] [Citace: 25. 07 2013.] http://www.underbit.com/products/mad.

22. GNU ccAudio – Summary. [Online] (c) 2013. [Citace: 25. 07 2013.] http://savannah.gnu.org/projects/ccaudio.

23. **PRUETT, Michael.** Audio File Library. [Online] 06. 03 2013. [Citace: 25. 07 2013.] http://audiofile.68k.org/.

24. **CASTRO LOPO, Eric de.** libsndfile. *libsndfile.* [Online] [Citace: 25. 07 2013.] http://www.mega-nerd.com/libsndfile/.

25. Computer Language Benchmarks Game. *The Computer Language Benchmarks Game.* [Online] [Citace: 01. 03 2013.] http://benchmarksgame.alioth.debian.org/.

26. k **Desktop Environment incorporated society.** KDE Necessitas project. *KDE Necessitas project.* [Online] k Desktop Environment incorporated society. [Citace: 01. 03 2013.] http://necessitas.kde.org/.

27. **TAUFER, Ivan, KOTYK, Josef a JAVŮREK, Milan.** *Jak psát a obhajovat závěrečnou práci bakalířskou, diplomovou, rigorózní, disertační, habilitační.* Pardubice : Univerzita Pardubice, 2009. ISBN 978-80-7395-157-3.

28. **Qt Project Hosting.** Documentation | Qt Project. *Qt Project.* [Online] Qt Project Hosting, 2011. [Citace: 29. 12 2012.] http://qt-project.org/doc/.

29. **WHEELER, Scott.** TagLib. *TagLib Audio Meta-Data Library.* [Online] [Citace: 29. 12 2012.] http://taglib.github.com/api/.

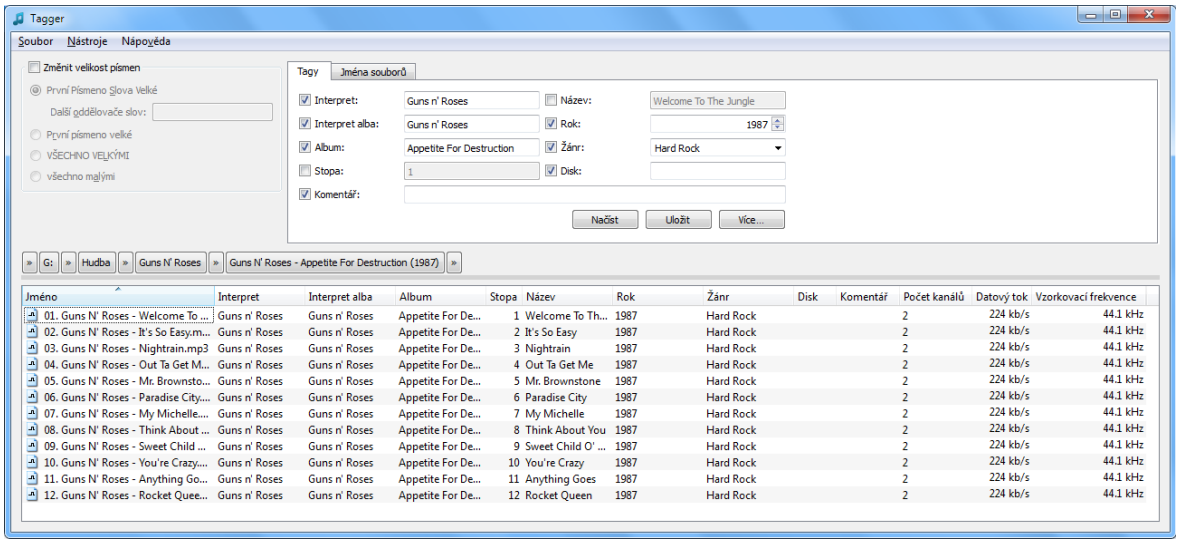
30. **MAHONEY, Dirk.** id3lib. *The ID3v1/ID3v2 Tagging Library.* [Online] [Citace: 29. 12 2012.] http://id3lib.sourceforge.net/.

Příloha A – Uživatelská příručka aplikace

Aplikace Tagger slouží k úpravám tagů v mediálních souborech. K přístupu k souborům využívá systém tzv. pluginů. Na těch mj. závisí, které a kolik formátů umí aplikace spravovat.

V této kapitole bude zmíněn pojem označené soubory. Je nutno zmínit, že pojem je zjednodušený. Aplikace považuje za označené soubory buď ty skutečně vybrané, nebo v případě, že žádné vybrané nejsou, všechny soubory.

Vzhled hlavního okna aplikace je na obrázku 1 této přílohy.



Příloha A obrázek 1 – Hlavní okno programu

Úplně nahoře se nachází aplikační menu (menu). Pod ním vlevo je měnič velikosti písmen, vpravo oblast dávkového zpracování souborů. Pod těmito oblastmi se nachází adresní řádek, a úplně vespod je seznam souborů.

Menu

Nahoře se nachází klasické menu, které sestává z položek:

- Soubor,
  - Konec,
- Nástroje,
  - Nastavení,
  - Generovat čísla stop,
- Nápověda,
  - Qt,
  - O Aplikaci.

Položku konec jistě není nutno blíže vysvětlovat, po jejím zvolení se uzavře program.

Menu nástroje má dvě položky, „Nastavení…“ otevře dialogové okno umožňující nastavení aplikace. To bude podrobněji popsáno dále. Položka „Generovat čísla stop“ umožňuje automaticky vyplnit čísla stop dle aktuálního seřazení seznamu souborů. Má čtyři podpoložky, které mají na první pohled nejasné názvy, proto budou uvedeny. Pro složitější názvy položek menu budou uvedeny příklady. V příkladech bude použit rozsah 1 – 15.

- „x“ generuje čísla 1 – 15.
- „0x“ generuje čísla 01 –15.
- „x/0x“ generuje čísla 1/15 – 15/15.
- „0x/0x“ generuje čísla 01/15 – 15/15.

Nápověda obsahuje dvě položky. „O Qt…“ zobrazí informace o multiplatformní knihovně Qt, kterou tato aplikace interně využívá, a autorovi se zdálo vhodné na to upozornit uživatele. „O Aplikaci…“ zobrazuje velmi základní informace, jakou je název, shrnutí k čemu je vhodná, název zavedené knihovny, která se používá pro nízko úrovňovou práci se soubory a seznam podporovaných souborů. Mimo to ještě zobrazuje jméno autora a jeho email.

Měnič velikosti písmen

Měnič velikosti písmen umístěný vlevo nahoře umožňuje upravit velikosti (malá / velká) písmen. Pokud je aktivní, pak je použit na všechny ukládané informace tzn., pokud bude zvolena položka „VŠECHNO VELKÝMI“ (stojí za povšimnutí, že názvy jsou zároveň příklady), tak budou převedeny na velká písmena i položky, které nejsou vpravo zaškrtnuty, a dokonce i položky, které nejsou v pravé části vůbec uvedeny.

Dávková úprava tagů

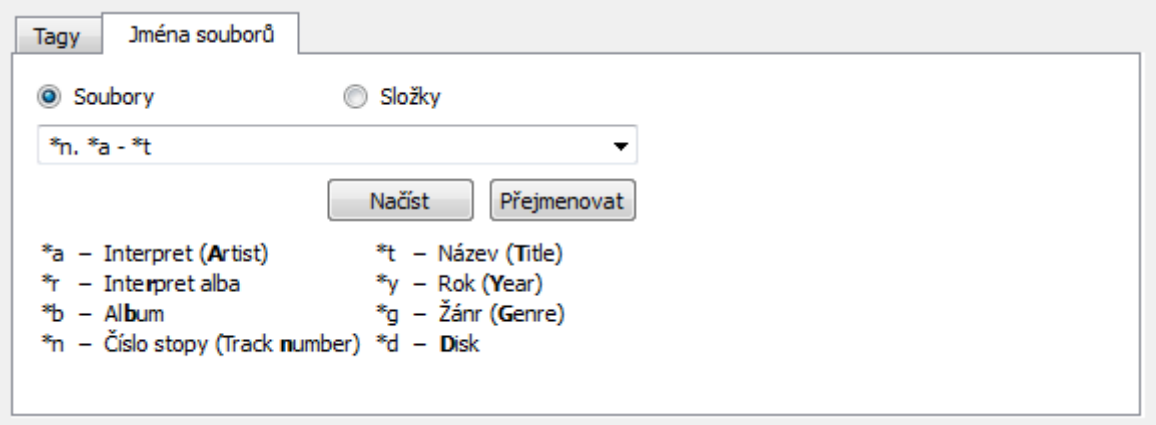
Vpravo od měniče velikosti písmen se nachází dva panely – první slouží k dávkové úpravě tagů a druhá, umožňuje přejmenování souborů, resp. aktuální složky a zároveň získání informací z těchto názvů.

Vlevo od každé položky je zaškrtávací pole. Nezaškrtnuté nedovoluje editaci příslušného textového pole a signalizuje, že tato informace nebude zapsána do žádného ze souborů.

Tlačítko „Načíst“ získá a vyplní informace do všech textových polí, tedy i těch neaktivních dle prvního vybraného souboru. Tlačítko „Uložit“ zapíše informace z nezašedlých polí do vybraných souborů. Tlačítko „Více…“ zobrazí dialogové okno, které umožní editovat další informace. Vzhledem k tomu, že logika ovládání tohoto dialogového okna je podobná právě popsanému principu, nebude toto okno více popsáno. Ale bude popsáno podobné okno, které se používá při editaci více informací v jednom souboru.

Dávkové přejmenování a dávkové načtení informací z názvů

Tento panel zobrazený na obrázku 2 této přílohy slouží k úpravě názvu dle informací v tagu a obráceně. Úplně nahoře je přepínač, který nastavuje, zda se bude pracovat s názvy souborů nebo názvem složky.



Příloha A obrázek 2 – Panel práce se jmény

Pod přepínačem se nachází výběrové pole s možností editace, které umožňuje pomocí zástupných znaků definovat formát jména. Pokud se v poli objeví znak, který není zástupným symbolem, pak je tento považován za konstantu. Položky výběrového pole lze upravit v nastavení. Pod tímto polem se nachází tlačítka „Načíst“ a „Přejmenovat“ jejichž funkce je zřejmá.

Adresní řádek

Přibližně uprostřed okna se nachází adresní řádek. Slouží k zobrazení a aktuální adresy. Lze ho používat podobně jako adresní řádek známý z Průzkumníka Windows Vista a Windows 7.

Stiskem některého z tlačítek se jménem složky je možné přejít do libovolné nadřazené složky. Tlačítka se šipkou slouží k zobrazení podsložek složky uvedené vlevo od tohoto tlačítka. Tlačítko se šipkou umístěné úplně vlevo má význam jen na operačních systémech s více kořenovými složkami (typicky Microsoft Windows). Při kliknutí mimo tlačítka se zobrazí textové pole s možností ruční editace adresy.

Pod adresním řádkem se nachází úzký tmavě šedý proužek. Pokud je tlačítek se složkami tolik, že se nevejdou do okna, pak tento proužek funguje jako posuvník.

Seznam souborů

Dole umístěný seznam souborů slouží k zobrazení podporovaných souborů a složek. U složek je možné dvojklikem přejít do této složky. U souborů jsou vyjma názvů zobrazeny i informace z nich načtené. Všechny operace se provádí pouze na vybraných souborech. Jednotlivé sloupce mohou být uživatelsky skryty pomocí kontextového menu

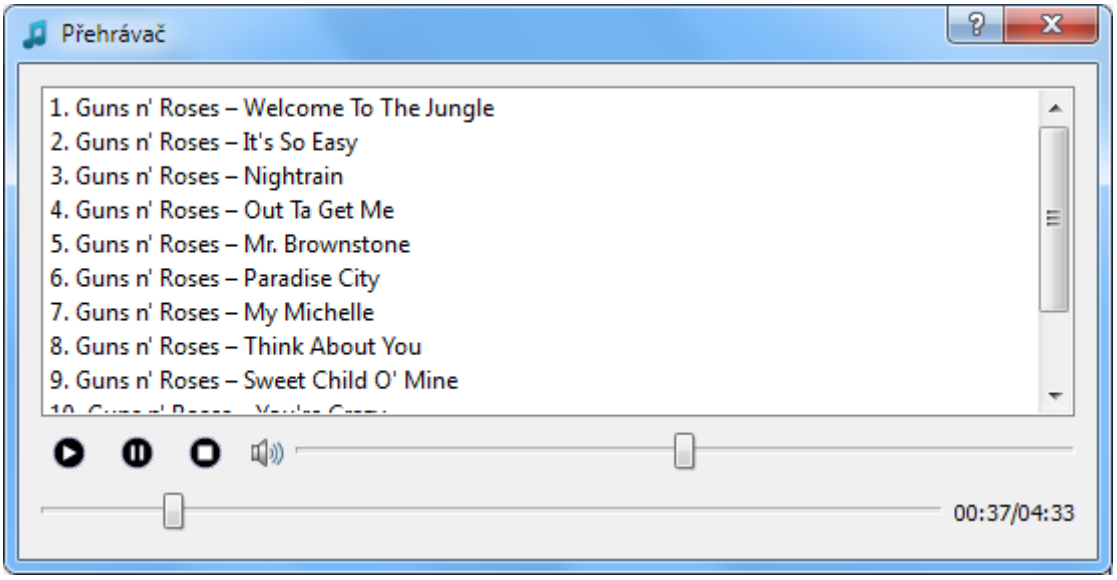
horního řádku s názvy. Jednotlivým sloupům je možné nastavit i šířku. Mimo to mohou být i uživatelsky setříděny.

Řádky se soubory mají také k dispozici kontextové menu. To má čtyři položky, které poskytují následující funkce:

- přehrát v interním přehrávači,
- přehrát v externím přehrávači,
- upravit informace,
- přejmenovat.

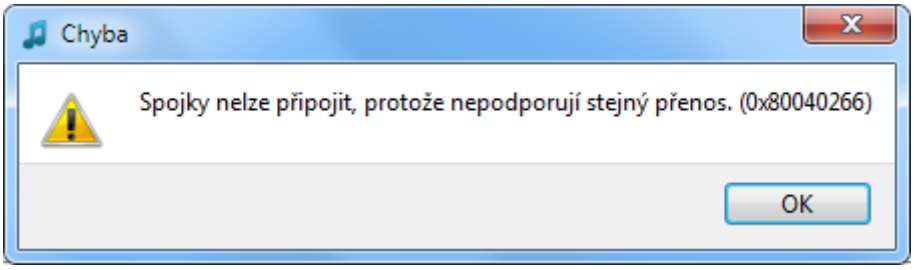
Přehrávač

Aplikace obsahuje jednoduchý přehrávač souborů vyobrazený na obrázku 3. Obsahuje pouze základní uživatelské prvky, kterými jsou seznam k přehrání, tlačítka přehrávání, pauza, zastavení, ztlumení hlasitosti a dvou posuvníku – horní slouží k regulaci hlasitosti, dolní k přesunu na jiný čas skladby.



Příloha A obrázek 3 – Přehrávač

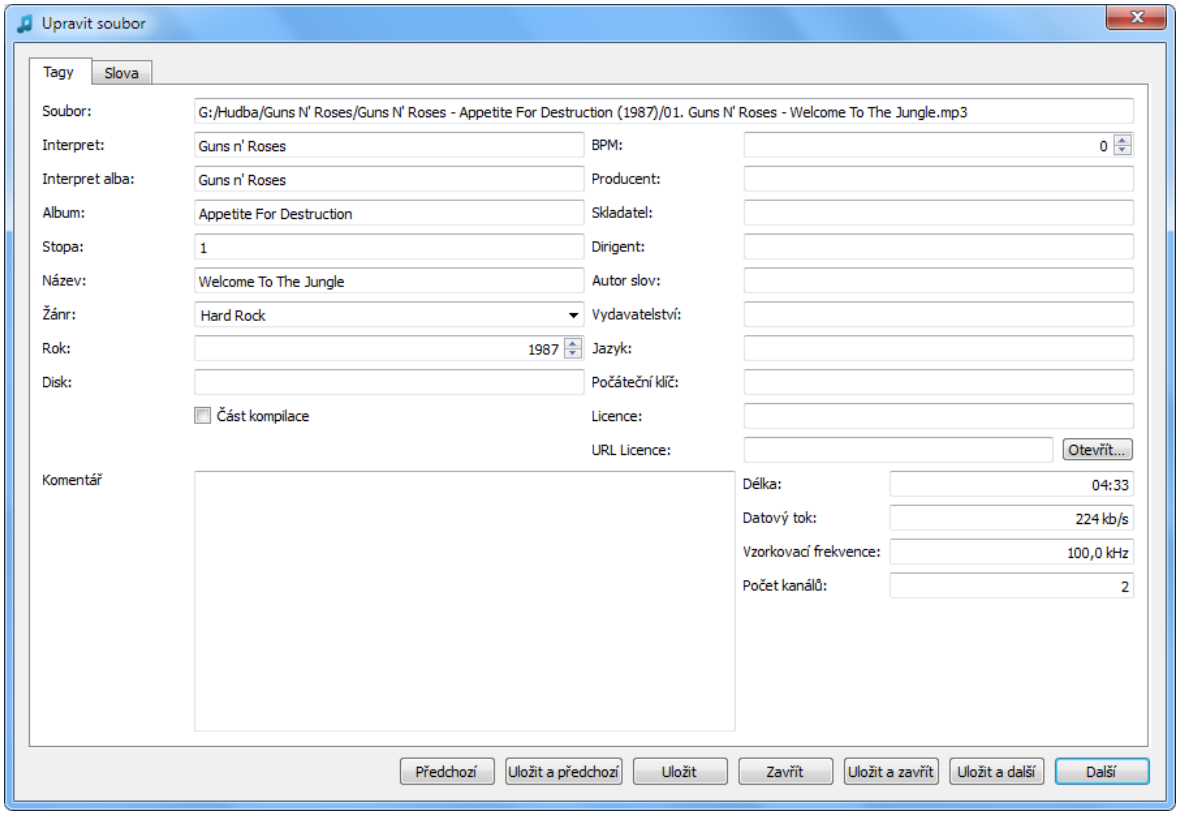
Vzhledem k tomu, že přehrávání je závislé na systémových komponentách, a to může selhat s nepříliš jasnými systémovými upozorněními (viz obrázek 4) byla do aplikace přidána možnost otevřít soubor v externím přehrávači. Použije se přehrávač, který má uživatel asociovaný s daným souborem.



Příloha A obrázek 4 – Chybová hláška přehrávače získaná ze systému

Úprava tagu souboru

Tato položka kontextové menu umožňuje přistupovat k dialogovému oknu zobrazenému na obrázku 5, ve kterém lze editovat všechny informaci po jednotlivých souborech. Položka soubor je určena pouze ke čtení (přejmenování souboru je provedeno jinak), stejně tak položky délka, datový tok, vzorkovací frekvence a počet kanálů. Ostatní položky lze editovat dle libosti.



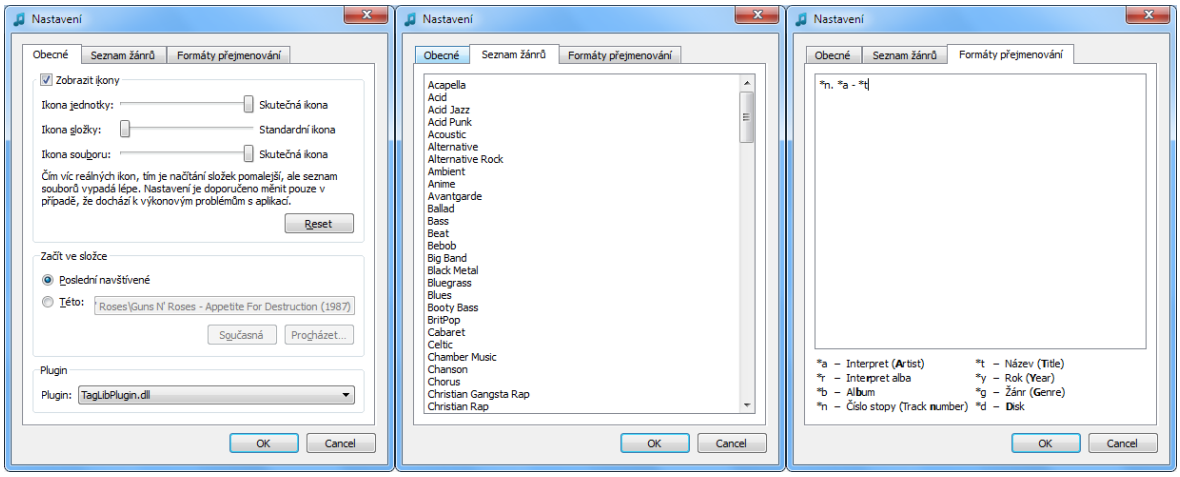
Příloha A obrázek 5 – Rozšířená úprava informací v souboru

Ve spodní části se nachází celkem sedm tlačítek, které umožňuj toho okno zavřít, přejít na další nebo předchozí soubor, uložit informace. Některé, často používané kombinace byly spojeny pod jedno tlačítko. V případě použití tlačítka „Předchozí“ nebo „Další“ a jejich modifikací při zobrazení prvního nebo posledního souboru dojde k zavření tohoto okna.

Aplikace také obsahuje podobné okno, ve kterém lze editovat informace ve více souborech najednou. Toto dialogové okno je dostupné pod tlačítkem „Více…“ v hlavním okně aplikace.

Nastavení

Posledním dialogovým oknem aplikace je nastavení. To se skládá celkem ze tří panelů, které jsou vyobrazeny na obrázku 6.



Příloha A obrázek 6 – Nastavení

Na panelu obecné lze nastavit v podstatě pouze tři položky. První určuje, zda a jaké ikony se budou zobrazovat v seznamu souborů a adresním řádku. Tato položka je zde z výkonnostních důvodů, při výpisu složky s velkým množstvím souborů (řádově tisíce a více) může dojít k až několika sekundové prodlevě, která je způsobena právě načítáním ikon.

Dále je zde nastavení složky, ve které se má aplikace nacházet po spuštění. Je zde možnost nastavit konkrétní složku nebo vždy začínat v poslední navštívené.

Jako poslední na této záložce je volba pluginu. Jak již bylo zmíněno, aplikace využívá systému pluginů, které poskytují nízko úrovňové operace se soubory. Volba pluginu tedy může velmi změnit počet a typ podporovaných souborů.

Na druhé záložce se nachází editor žánrů. Na každý řádek připadá jeden žánr. Seznam bude před uložením abecedně seřazen.

Třetí záložka obsahuje editor seznamu formátů názvů souborů. Položky tohoto seznamu jsou využity v hlavním okně aplikace při dávkovém přejmenování a podobných operacích. I zde platí pravidlo jedné položky na řádek.

Po uložení změn se nové nastavení (včetně pluginu) projeví okamžitě, není nutné aplikaci restartovat.

Příloha B – Programátorská příručka aplikace

Na přiloženém CD jsou k dispozici kompletní zdrojové kódy aplikace. Ty lze rozdělit do čtyř skupin, z toho většina jich je znovu použitelná v jiných aplikacích.

První skupina jsou soubory specifické k dané aplikaci, nejčastěji obsahují upravené komponenty, a z větší části nejsou využitelné v jiných aplikacích, proto zde nebudou podrobněji popsány.

Komponenta AddressBar

Do druhé skupiny patří komplexnější komponenta, která dává uživateli k dispozici přívětivý adresní řádek. Nachází se ve složce „addressbar“. Komponenta je podobná (ale ne úplně stejná) adresnímu řádku v Průzkumníku v operačních systémech Windows Vista a Windows 7.

Komponenta dědí od standardní Qt třídy QWidget. K jejím vlastnostem přidává metody dir() a setDir(), které umožňují nastavit, resp. získat, aktuální složku. Dále obsahuje sloty goToRoot() a goUp(), které umožňují jednoduchý přechod do nadřazené, resp. kořenové složky.

Při jakékoli změně aktuální složky je emitován signál dirChanged(), který informuje zbytek aplikace o změně aktuálního adresáře. Signál lze zachytit a aktualizovat například seznam souborů v jiné komponentě. Komponentu bohužel není možné použít pro zachycení jiné hierarchické struktury, protože je interně postavena na Qt třídě QDir reprezentující složku

Komponenta StringEditor

V třetí skupině nacházející se ve složce stringeditor se nachází znovu použitelná komponenta, která je vhodná pro komplexnější úpravy řetězců. Je založena na návrhovém vzoru Dekorátor. Díky tomu je komponenta velmi snadno rozšiřitelná.

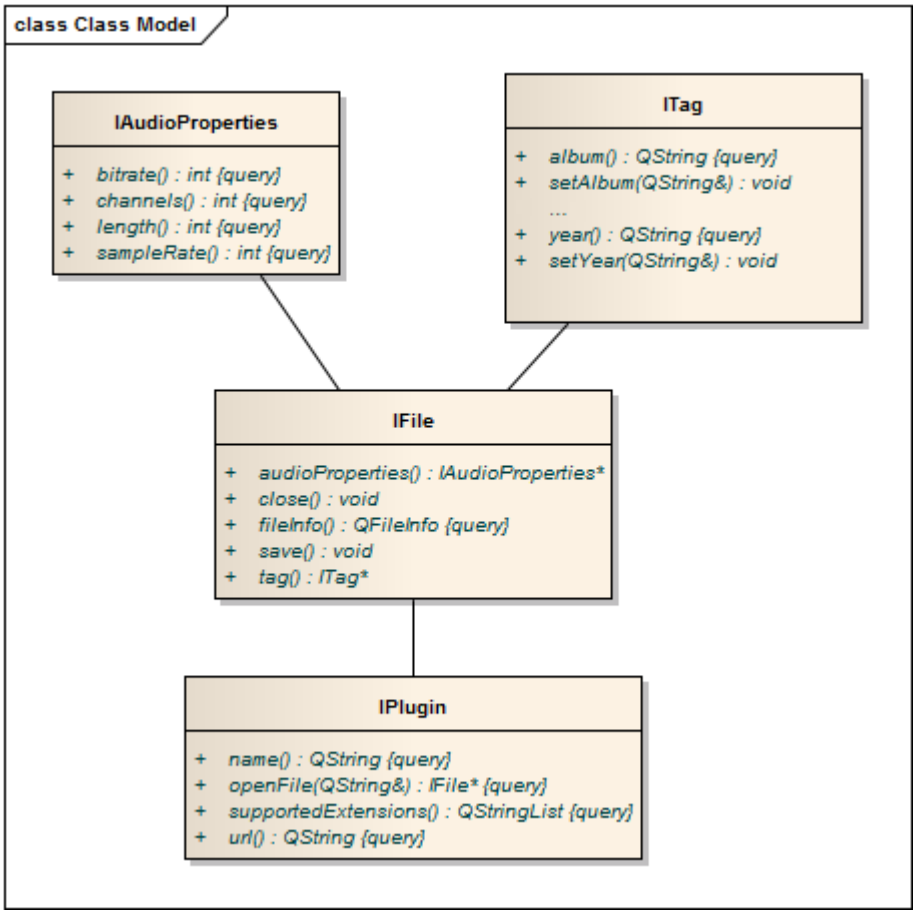
Jakákoli třída, která má s touto komponentou spolupracovat musí dědit od třídy AStringEditor. Konstruktor musí zavolat metodu next() a předat ji ukazatel na instanci potomka AStringEditoru nebo NULL. Mimo to musí třída implementovat metodu privateTransform(). Ta má následující rozhraní:

```
virtual QString privateTransform(const QString& string) const;
```

Vstupní řetězec do metody je původní, neupravený, řetězec a na výstupu má metoda upravený řetězec.

Komponenta Plugin

Čtvrtá skupina, která je umístěna ve složce plugin je z hlediska cílů práce nejdůležitější, a proto bude popsána mnohem podrobněji. Jedná se o rozhraní pro komunikaci mezi vlastní aplikací a pluginem, kterým může být jak knihovna pro práci s tagy (viz projekt ID3LibPlugin), tak plugin, která využívá ke své funkci knihovnu třetí strany pro práci na nižších úrovních (viz projekt TagLibPlugin). Diagram tříd je zobrazený na obrázku 1 této přílohy (třída ITag je zkrácena).



Příloha B obrázek 1 – Diagram tříd pluginu

Při návrhu rozhraní byla snaha o jednoduchost a zároveň komplexnost. Proto se rozhraní skládá z pouze čtyř čistě abstraktních tříd, jmenovitě:

- IPlugin,
- IFile,
- ITag,
- IAudioProperties.

Z technických důvodů je nutné, aby všechny třídy rozšiřující tyto abstraktní dědily od QObject.

Třída IPlugin je hlavní třídou pluginu, a poskytuje pouze základní metody, mezi něž patří name() a url(), které musí poskytovat základní informace o dané knihovně. Dále poskytuje seznam koncovek (například "mp3", "ogg", "wma"), které je knihovna schopna zpracovat. Poslední metodou je openFile(), která nejčastěji volá konstruktor třídy implementující abstraktní třídu IFile.

V dolní části souboru se pak nachází řádek:

```
Q_DECLARE_INTERFACE(IPlugin, "tagger.plugin.interface/1.0")
```

Tento řádek je makro, které je při kompilaci rozvinuto a při načítání pluginu do aplikace ji informuje o svém názvu a verzi. Pokud by v pluginu byla přejmenována třída IPlugin (změny názvů tříd jsou jediné dovolené úpravy v rozhraních), tak na tomto řádku také nutně musí dojít k příslušné úpravě. V případě že by došlo ke změně řetězce v druhém parametru makra, pak by aplikace nedokázala tento plugin načíst.

Třída IFile se stará o základní operace se souborem. V konstruktoru musí být otevřen soubor, jehož jméno získává IPlugin, následně přečteny všechny relevantní informace a pak musí být soubor uzavřen. Pokud toto nebude splněno, tak nebude možno přejmenovat soubory.

Mimo konstruktoru ještě třída obsahuje metody tag() a audioProperties(), které předávají ukazatele na instance tříd, jež budou popsány dále. Dále jsou zde metody save() a close(). Metoda save() otevře soubor, uloží do něj změněné informace a pak jej musí zavřít. K tomu se používá metoda close(). Ta může být zavolána i z aplikace, všechny metody této třídy musí toto brát v potaz.

Poslední dvě abstraktní třídy – ITag a IAudioProperties obsahují pouze množství getterů a setterů (settery pouze v třídě ITag). V případě, že nějaká položka není k dispozici (např. chybějící disk u ID3v1 tagu, pak plugin musí volání setteru ignorovat. Při volání getteru pak může vrátit libovolnou hodnotu, doporučuje se QString::null pro řetězcové hodnoty a −1 pro čísla.

V případě jakýchkoli problémů se doporučuje jako první zkontrolovat následující body:

- Soubory by měly být otevřeny pouze při vstupně-výstupních operacích, jinak by měly být uzavřeny, jinak může docházet k problémům s přejmenováním souborů.
- Je doporučeno, aby načtení dat probíhalo v konstruktoru (k reálnému zpomalení nedojde, v aplikaci jsou data zobrazována okamžitě).
- Pokud plugin nepodporuje nějakou koncovku souboru, pak tato nesmí být uvedena v seznamu, který je vracen metodou supportedExtensions().

- Třídy rozhraní nesmí dědit od žádné jiné třídy, a to ani QObject, kdežto implementující třídy musí mít minimálně dvojnásobnou dědičnost – od QObject a od příslušného rozhraní.
- V souboru IPlugin.h nesmí být změněn argument makra udávající název a verzi rozhraní, jinak by jej aplikace nedokázala načíst.
- Z optimalizačních důvodů je nutné, aby název výsledného pluginu končil na Plugin.dll nebo Plugin.so (např. TagLibPlugin.dll / TagLibPlugin.so, systémy s jinými koncovkami sdílených knihoven nejsou v aktuální verzi podporovány), jinak bude aplikací ignorován. Pluginy se umisťují do stejné složky, ve které se nachází spustitelný soubor. Případné další knihovny, které jsou využívány pluginy musí být umístěny také v této složce.
- Abstraktní třídy nesmí osahovat jiné metody než zde definované, a to ani privátní. Naopak implementační třídy mohou mít deklarovány i další metody, a to i veřejné.

Další informace čtenář nalezne v souborech příslušného rozhraní, které jsou dostupné ve složce plugin.

Příloha C – Seznam žánrů podporovaných v ID3v1

0. Blues
1. Classic Rock
2. Country
3. Dance
4. Disco
5. Funk
6. Grunge
7. Hip-Hop
8. Jazz
9. Metal
10. New Age
11. Oldies
12. Other
13. Pop
14. R&B
15. Rap
16. Reggae
17. Rock
18. Techno
19. Industrial
20. Alternative
21. Ska
22. Death Metal
23. Pranks
24. Soundtrack
25. Euro-Techno
26. Ambient
27. Trip-Hop
28. Vocal
29. Jazz+Funk
30. Fusion
31. Trance
32. Classical
33. Instrumental
34. Acid
35. House
36. Game
37. Sound Clip
38. Gospel
39. Noise
40. AlternRock
41. Bass
42. Soul
43. Punk
44. Space
45. Meditative
46. Instrumental Pop
47. Instrumental Rock
48. Ethnic
49. Gothic
50. Darkwave
51. Techno-Industrial
52. Electronic
53. Pop-Folk
54. Eurodance
55. Dream
56. Southern Rock
57. Comedy
58. Cult
59. Gangsta
60. Top 40
61. Christian Rap
62. Pop/Funk
63. Jungle
64. Native American
65. Cabaret
66. New Wave
67. Psychadelic
68. Rave
69. Showtunes
70. Trailer
71. Lo-Fi
72. Tribal
73. Acid Punk
74. Acid Jazz
75. Polka
76. Retro
77. Musical
78. Rock & Roll
79. Hard Rock

Příloha D – Obsah přiloženého CD

Přiložené CD obsahuje:

- tuto práci ve formátu pdf,
- zdrojový kód aplikace,
- zdrojový kód pluginů,
- soubory s rozhraním pluginů ve složce plugin,
- kompletní spustitelnou aplikaci včetně všech potřebných knihoven.

Přiložené CD neobsahuje zdrojové kódy využitých otevřených knihoven, ale obsahuje jejich hlavičkové a další soubory tak, aby šlo pluginy později zkompilovat pomocí GCC 4.4 a kompatibilních kompilátorů.