Multimediální výuka pomocí Imagine

Bakalářská práce

Petr Pechoušek

Vedoucí bakalářské práce:
PaedDr. Jiří Vaníček, Ph.D.

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky
Rok 2009

Prohlášení

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne 20.4.2009

Anotace

Cílem této bakalářské práce je vytvoření metodické příručky pro vývojové prostředí Imagine. Tato příručka se zabývá využitím jeho nástrojů pro práci s multimédii a snaží se objasnit základní pojmy a technické informace, které s problematikou multimedií souvisejí. K tomuto účelu je pro lepší pochopení a názorné předvedení vytvořena sada projektů, realizovaná v prostředí Imagine.

Klíčová slova

Logo, Imagine, multimedia, zvuk, video, projekt

Abstract

The goal of this bachelor work is to create a methodical manual for development environment Imagine. This manual deals with the use of its multimedia objects and tries to explain basic conceptions and technical informations, that relate to multimedia. For this porpuse is for better understanding and illustation made a set of projects implemented in Imagine.

Keywords

Logo, Imagine, multimedia, audio, video, project

Poděkování

Děkuji PaedDr. Jiřímu Vaníčkovi, Ph.D. za informace, rady a materiály, které mi poskytl během vypracování této bakalářské práce.

Obsah

1 Úvod

V dnešní době se setkáváme s multimédii doslova na každém kroku, ať jsou to reklamy či bannery nebo multimediální prezentace. Proto není překvapením, že i vývojové prostředí Imagine, které se dá využít i na tvorbu prezentací, implementuje sadu objektů a příkazů pro práci s multimédii.

Toto vývojové prostředí se díky své propracované metodice s úspěchem využívá na základních a středních školách k výuce programování. Jedná se o moderní didaktický nástroj, který umožňuje rychlí počáteční pokrok, tvorbu zábavných úkolů a možnost v krátké době tvořit vlastní projekty a to hlavně díky metodice výuky algoritmizace, představivosti a řešení problémů pomocí kreslení v tzv. želví geometrii.

Metodická příručka [A], kterou jsem vytvořil a je přiložena k této práci, je určena pro učitele pracující se studenty, kteří již pochopili základy programování, osvojili si základní příkazy prostředí Imagine, rádi by ve svých projektech využili některý z multimediálních objektů či příkazů, se kterými prostředí Imagine pracuje, a nestačí jim k tomu informace obsažené v nápovědě aplikace.

Mezi tyto objekty patří objekt *Zvuk*, který na pracovní ploše vytvoří tlačítko, které umožňuje přehrát nejen soubory typu MP3 či WAV. Dále je to objekt *Hudba*, který na pracovní ploše opět vytvoří tlačítko, umožňující přehrávání zvukových souborů typu MID a RMI. Dalším objektem je objekt *Melodie*, který uživateli nabízí možnost skládat a upravovat vlastní MIDI melodie. Posledním multimediálním objektem je objekt *Videoklip*, který umožňuje na pracovní ploše vytvořit přehrávací okno pro video.

Mezi nejdůležitější příkazy prostředí Imagine pracující s multimédii patří příkaz *přehraj*, který zahraje melodii danou vstupním seznamem a příkaz *hrajNotu*, který zahraje notu danou vstupním seznamem. Dalšími důležitými

příkazy jsou příkaz *přehrajZvuk* a *přehrajSoubor*, které přehrají vstupní zvukový soubor.

2 Cíle práce

Cílem práce bylo vytvoření interaktivní metodické příručky a sady úloh pro vývojové prostředí Imagine. Na těchto příkladech poté předvést a vysvětlit základní možnosti použití objektů a příkazů pracujících s digitálním videem a zvukem v prostředí Imagine.

Dalším cílem bylo objasnit teoretické základy související s multimédii a s předzpracováním digitálních audio a video souborů tak, aby se daly ve vývojovém prostředí Imagine použít.

3 Současný stav problematiky

3.1 Literatura

Jelikož se při výuce programování v prostředí Imagine využívá zejména želví geometrie, je většina česky a slovensky psané literatury soustředěna okolo tohoto tématu [1] [2] [3].

Za zmínku stojí kniha autorů A. Blaha a I. Kalaše [1], kde autoři popisují příkaz *přehraj* a vysvětlují na něm práci se seznamy neznámé délky.

Druhou publikací, která zjednodušeně popisuje objekt *Melodie*, je učebnice autorů J. Vaníčka a R. Mikeše [3].

3.2 Nápověda prostředí Imagine

Hlavním zdrojem informací ohledně využití multimédií v prostředí Imagine je nápověda aplikace. V záložce *Multimédia, hlasový vstup a výstup* se v podzáložce *Multimédia* dozvíme základní příkazy pro práci s multimédii, jimiž jsou příkazy *přehraj, hrajNotu, přehrajZvuk, přehrajSoubor, hudebníNástroj, startVideo, stopVideo.* Můžeme se zde dočíst základní parametry výše zmíněných příkazů a jejich vlastnosti.

V další částí nápovědy aplikace, nazvané *Třída PřehrávačMédií,* se dozvíme veškeré informace o tomto multimediálním objektu. Jeho celkový popis, základní nastavení, společné proměnné, události a procedury s ním spojené.

Přiložená příručka rozšiřuje tyto informace o vlastní zkušenosti a rady jak urychlit vývoj a vyhnout se nepříjemným problémům, ukázky kódu, příklady a popisuje odhalené chyby v nápovědě aplikace i v samotném prostředí Imagine.

4 Imagine a multimédia

Prostředí Imagine využívá k práci s multimédii třech přístupů.

4.1 Příkaz *přehrajZvuk*

Prvním z nich je implementované přehrávání audio souborů typu WAV pomocí příkazu *přehrajZvuk.*

4.2 Příkazy pro práci s MIDI

Druhým způsobem je implementovaná podpora tvorby zvuků pomocí softwarového syntezátoru a standardu MIDI v příkazech *hrajNotu* a *přehraj*.

Zatímco příkaz *hrajNotu* přehrává tóny na svém vstupu najednou, příkaz *přehraj* využívá k zápisu svého vstupu jakýsi pseudokód pro zapisování MIDI informací. Jeho vstup se tedy vyhodnocuje postupně.

Ke správnému chodu těchto příkazů je třeba mít nastavenu hlasitost softwarového syntezátoru v nastavení hlasitosti zvukové karty.

4.3 Podpora multimédií pomocí MCI rozhraní

Posledním způsobem, kterým prostředí Imagine přehrává multimédia, je prostřednictvím takzvaného MCI rozhraní. Příkazy *přehrajSoubor* a *startVideo* a objekt *PřehrávačMédií* pracují s tímto rozhraním a díky němu je prostředí Imagine schopno přehrávat jakýkoliv multimediální soubor, jehož podpora je nainstalována na daném počítači.

5 Metoda práce

5.1 Studium

Prvním krokem k napsání této bakalářské práce bylo studium literatury, nápovědy aplikace a dostupných internetových zdrojů. Tímto studiem jsem se seznámil se základními příkazy a objekty prostředí Imagine a poté jsem byl schopen zaměřit se na objekty a příkazy pracující s multimédii.

5.2 Tvorba multimediálních projektů a psaní příručky

Poté co jsem se naučil pracovat s prostředím Imagine, jsem se rozhodl vytvořit komplexní multimediální projekt *Beatmaker*, který je popsán v poslední kapitole přiložené příručky a ve kterém využívám většinu příkazů a objektů prostředí Imagine pracujících s multimédii.

Prací na tomto projektu jsem si vyzkoušel reálnou situaci člověka, který chce v tomto prostředí používat multimédia a na základě těchto zkušeností a zkušeností získaných při experimentování a tvorbě příkladových projektů jsem byl schopen začít psát příručku.

6 Výsledky práce

6.1 Metodická příručka

Přiložená metodická příručka je rozdělena do třech kapitol.

6.1.1 Technické informace

První kapitolou této příručky jsou technické informace. V této teoretické části pojednávám o hardwarových a softwarových nárocích na využívání multimédií a dále v ní zjednodušeně popisuji základní pojmy týkající se digitálního zvuku a videa, které by měl uživatel, který chce pracovat s multimédii, znát.

Těmito pojmy jsou vzorkování, syntéza zvuku, norma General MIDI, kodek a MCI rozhraní. Také se zde zmiňuji o nejpoužívanějších multimediálních formátech a jejich předzpracování pro využití v prostředí Imagine.

6.1.2 Nástroje pro práci s audiem a videem

Druhá část příručky popisuje jednotlivé objekty a příkazy prostředí Imagine pracující s multimédii a je rozdělena na dvě kapitoly. První pro práci s audiem a druhá pro práci s videem. Kapitola pro práci s audiem je navíc dál rozdělena na kapitolu pro práci s MIDI a kapitolu pro práci s audio soubory.

U každého příkazu či objektu je nejméně jeden příklad jeho využití. Každý příklad je popsán v příručce a jako hotový projekt zpracován na přiloženém CD [B]. Tyto příklady využívají nejen příkazy a objekty pracující s multimédii, ale samozřejmě i ostatní základní příkazy a objekty prostředí Imagine, jako jsou například příkazy *slovo* a *věta* pro práci se slovy a větami, příkazy *prvek* a *nahraď* pro práci se seznamy, objekty *Vodorovný posuvník, Tlačítko* a další. Učitel si tak může vybrat příklad, vhodný k procvičení právě probíraného tématu.

6.1.3 Využití multimédií v projektu

Ve třetí části jsou popsány komplexnější multimediální projekty. Tato část je rozdělena na dvě kapitoly. V první jsou projekty, *Beatmaker, Hra s hraním* a *Test anglických slovíček* pracující s audio prvky prostředí Imagine a druhá část popisuje projekt *Videopřehrávač* pracující s objektem *PřehrávačMédií*. Tyto projekty jsou součástí přiloženého CD [B] a slouží hlavně k demonstraci toho, co lze v prostředí Imagine za pomoci multimediálních prvků vytvořit.

6.2 Odhalené chyby v nápovědě aplikace Imagine

Při mém experimentování s objekty a příkazy prostředí Imagine jsem odhalil několik nepřesností v nápovědě aplikace.

1. Příkaz *odstartuj*, který má spustit paralelní proces, nefunguje. Jako alternativu můžeme použít příkaz *odtartujPo 0.*
2. V nápovědě je uveden jako příkaz pro zapnutí přehrávání otevřeného multimediálního souboru příkaz *přehraj*. Tento příkaz je však použitelný pouze pro přehrávání a skládání MIDI melodií. Správným příkazem pro zapnutí přehrávání otevřeného multimediálního souboru je příkaz *přehraj_mm.*
3. Při tvorbě *Příkladu 2 Kapitoly 4.1.3* jsem přišel na to, že kódy, uvedené v nápovědě aplikace u příkazu *hrajNotu*, neodpovídají správným kódům not. Experimentováním jsem přišel na to, že je nejspíše celý seznam posunut o jeden prvek. Notu C, kterou by podle tabulky uvedené v nápovědě měla reprezentovat hodnota 60, tak ve skutečnosti reprezentuje hodnota 61.
4. Při použití příkladu u příkazu *startVideo*:

```
? ignoruj startVideo "|Media\auto závod.mpg| "mojeAuto
[child [] [100 100]]
```

prostředí Imagine padne. Tato chyba bylo testována na systémech Windows XP Pro SP2 a Windows Vista a je pravděpodobně zapříčiněna špatným překladem z originální verze.

6.3 Odhalené nedostatky aplikace Imagine

Při mém experimentování s objekty a příkazy prostředí Imagine jsem odhalil několik nedostatků a chyb tohoto prostředí, které mi dokázali zkomplikovat práci.

1. Přehrávání souboru pomocí příkazu *přehrajZvuk* nelze zastavit ani klávesou *F12* ani příkazem *zastavVše.* Pokud tedy spustíme nekonečný proces pomocí příkazu *pořád [přehrajZvuk "soubor]*, způsobíme si tím nemalý problém.
2. Jelikož prostředí Imagine využívá k přehrávání multimediálních souborů MCI rozhraní, je tedy schopné přehrávat pomocí příkazu *přehrajSoubor* a objektů *Zvuk* a *PřehrávačMultimédií* jakýkoliv audio a video formát, pro který jsou nainstalovány v systému Windows kodeky. Je tedy škoda, že dialogové okno pro výběr požadovaného souboru, které se otevře při vytvoření výše zmíněných objektů, nepodporuje vybrání všech multimediálních formátů. Tento problém se dá vyřešit pomocí vlastního příkazu, který pracuje s operací *pomocníkNačtiSoubor*.
3. Pokud jako vstup příkazu *přehrajSoubor* dáme soubor s příponou CDA, zapne se přehrávání Audio CD, pokud je tedy v jednotce CD – ROM, a toto přehrávání lze přerušit pouze restartem počítače. Toto bylo testováno na systémech Windows XP Pro SP2 a Windows Vista a je to nejspíše zapříčiněno tím, že se MCI příkaz zkonstruovaný příkazem *přehrajSoubor* odevzdá zařízení

zodpovědnému za přehrávání Audio CD, tedy CDAudio, a prostředí Imagine neumí toto zařízení opětovně vypnout.

4. Pokud nastavíme audio soubor k přehrávání objektu *PřehrávačMultimédií* pomocí příkazu *soubor!*, nebo jiného ekvivalentního způsobu, zmizí přehrávací okno pro video a zůstane pouze panel ovládání. Tento panel nereaguje na žádné události myši a pokud chceme měnit nastavení v rodném listu objektu *video*, musíme si tento objekt najít v okně paměti.
5. Jako nedostatek prostředí Imagine se mi jeví i fakt, že po exportu do EXE nebo WEB projektu nelze číst z disku, ani spouštět jiné aplikace pomocí příkazu *spusťProgram.*
6. Nakonec bych jistě také uvítal jakýsi ekvivalent combo boxu, který by se hodil zejména při výběru hudebního nástroje, pro jehož realizaci jsem byl nucen využít objektu *Vodorovný posuvník.*

7 Závěr

Cílem mé práce bylo vytvořit interaktivní metodickou příručku, která by rozšiřovala nápovědu aplikace a poskytovala by studentům a učitelům jakýsi přehled možností využití multimediálních objektů v prostředí Imagine, a to se myslím podařilo.

Příručka „Multimédia v prostředí Imagine“ je v textové formě přiložena k této o bakalářské práci [A] a její interaktivní webová forma se nachází na přiloženém CD [B].

Čtenář se po přečtení mnou vytvořené příručky dozví základní teoretické a praktické informace týkající se multimédií, které jsou potřebné k jejich správnému využití nejen v prostředí Imagine.

Dále se formou tutoriálů seznámí se základními objekty a příkazy prostředí Imagine pracujícími nejen s multimédii. Z částí příručky označených jako poznámka se dozví informace, které jsem vyzkoumal experimentováním s danými objekty a příkazy a které mu poslouží nejen ke zrychlení vývoje aplikací za podpory multimédií.

8 Reference

[1] BLAHO, Andrej, KALAŠ, Ivan. *Comenius Logo: tvorivá informatika*. 1. vyd. Bratislava : CL Group, 1998. 2 sv. (157, 189 s.). ISBN 80-967999-0-8.

[2] TOMCSÁNYIOVÁ, Monika. *Poď s námi programovať v Comenius Logu : Metodický materiál pre učitelov informatiky 7. ročníka ZŠ a 3. ročníka osemročných gymnázií*. 1. vyd. Bratislava : Metodické centrum, 1998. 68 s. ISBN 80-8052-027-5.

[3] VANÍČEK, Jiří, MIKEŠ, Radovan. *Informatika pro základní školy a víceletá gymnázia, 3. díl*. Učebnice. Brno: Computer Press, 2006. 96 s. ISBN 80-251-1082-6

Příloha A – Příručka

Multimédia v prostředí Imagine

Metodická příručka

Petr Pechoušek

2009

1 Úvod

Tato příručka je určena pro ty, kteří již mají s programováním ve vývojovém prostředí Imagine nějaké zkušenosti, rádi by do svých projektů zapojili multimediální prvky, a nestačí jim k tomu základní informace popsané v nápovědě aplikace.

2 Terminologie

V celé příručce je použit jednotný styl zápisu kódu.

```
Toto je příklad kódu
```

```
Toto je další příklad kódu
```

Objekty, proměnné, vlastnosti a názvy příkazů jsou psány kurzívou.

Pro zjednodušení popisu je ve výkladu použito zkrácených jmen objektů, tak jak je pojmenovává samo prostředí Imagine, viz Tab. 1.

Název objektu	**Použitá zkratka**
Posuvník	*p1*, *p2*, *p3*, *p4*, atd…
Tlačítko	*tl1*, *tl2*, *tl3*, *tl4*, atd…
Text	*text1*, *text2*, *text3*, *text4*, atd…
Želva	*ž1*, *ž2*, *ž3*, *ž4*, atd…

Tab. 1 – Použité zkratky pro objekty

3 Technické informace

V této kapitole se seznámíme s hardwarovými a softwarovými nároky na využívání multimédií a také něco málo teorie týkající se zpracování zvuku. Dále si povíme něco o podporovaných formátech a jejich předzpracování pro využití v prostředí Imagine.

3.1 Hardwarové a softwarové nároky pro práci s multimédii

Pokud chceme na svém počítači pracovat s digitálním zvukem a videem, potřebujeme, aby byl náš počítač takzvaně multimediální. Takovýto počítač je kromě kvalitní grafické karty vybaven i kartou zvukovou a nějakým reprodukčním zařízením, jako jsou sluchátka či reproduktory. Tak jako je výsledný obraz na monitoru počítače závislí na grafické kartě, je i výsledný zvuk závislý na kartě zvukové. Dále musíme mít v počítači nainstalovány příslušné kodeky pro audio a video formáty, které chceme přehrávat.

3.1.1 Vzorkování

Abychom mohli se zvukem pracovat přímo na našem počítači, je třeba tento zvuk převést z analogové podoby do formátu, který se dá na PC dále zpracovávat. Pro převod z analogového signálu na digitální se používá metoda zvaná vzorkování.

Tato metoda funguje tak, že se mnohokrát za vteřinu odebere vzorek analogového signálu a zaznamená se výška jeho amplitudy [2]. Jelikož není možné změřit tuto výšku v každém okamžiku, zaznamenává se pouze omezené množství vzorků. Vyšší množství vzorků znamená vyšší kvalitu zaznamenaného zvuku, ale také větší velikost výsledného souboru. Tato vzorkovací frekvence se udává v jednotkách Hz a pokud chceme, aby byl zvuk věrně převeden, musí být vždy 2krát větší, než je maximální frekvence zpracovávaného zvuku. Jelikož se frekvenční rozsah, který je člověk schopen vnímat, pohybuje v rozmezí od 20Hz do 20kHz, považuje se jako optimální vzorkovací frekvence 44.1kHz, tj. zvuková kvalita Audio CD. Kvalitnější zvukové karty se vyznačují mimojiné tím, že podporují vyšší vzorkovací frekvence (až 192Khz).

3.1.2 Syntéza zvuku

Vzorkování se používá při záznamu zvuku. Zvuková karta musí ale také umět zvuk vytvářet. Když například chceme, aby zahrála notu C hudebním nástrojem piano, musí vědět jak na to. Pro tyto účely se podle kvality zvukových karet využívá dvou přístupů:

1. **Syntéza FM** - Podstatou je myšlenka modelu ADSR, tj. že hudební zvuk má podobu cyklu, který se skládá ze čtyř částí: Nástup, Pokles, Trvání, Doznívání (anglicky Attack, Decay, Sustain, Release proto ADSR). Základen této syntézy je určení vlnového průběhu daného nástroje, měněním hodnot zmíněných čtyř částí cyklu. Zjednodušená povaha modelu ADSR je na úkor hudební věrnosti, proto se tento způsob vytváření zvuků využívá pouze pro amatérské účely [2].

2. **Wavetable (tabulková) syntéza –** Profesionální zvukové karty uchovávají vlnový průběh hudebních nástrojů v paměti ROM přímo na kartě. Tímto je dosažena vysoká kvalita vytvořených zvuků. Nevýhodou jsou ale vysoké pořizovací náklady.

General MIDI

Oba tyto způsoby vytváření zvuků spojuje norma General MIDI (Musical Instrument Digital Interface), vydaná roku 1991. Nástroj (v našem případě zvuková karta), který chce vyhovovat této normě, musí mít minimálně 24 hlasů s dynamickým přiřazováním na všech 16 kanálech MIDI [1]. Dále musí mít 128 zvukových barev (nástrojů), uspořádaných podle normované tabulky zvuků, viz. Tab. 2 [1]. Protože v normě není předepsána syntéza zvuku, mohou se zvuky různých nástrojů značně lišit na různých zvukových kartách, podle toho zda karta využívá FM, či wavetable syntézu.

Číslo hudebního nástroje	**Nástrojová skupina**
01 – 08	klavíry
09 – 16	laděné bicí
17 – 24	varhany
25 – 32	kytary
33 – 40	basové zvuky
41 – 48	smyčce
49 – 56	ansábové zvuky
57 – 64	žestě
65 – 72	plátkové
73 – 80	píšťaly
81 – 88	elektronické sólové zvuky
89 – 96	elektronické doprovodné zvuky
97 – 104	elektronické zvukové efekty
105 – 112	etnické
113 – 120	perkusivní
121 - 128	zvukové efekty

Tab. 2 – Skupiny nástrojů standardu General MIDI

3.1.3 MCI rozhraní

MCI je zkratka pro Multimedia Control Interface. Jedná se o programovací rozhraní pro ovládání multimediálních zařízení, nainstalovaných na našem počítači. Tyto zařízení se dělí do čtyř skupin [6]:

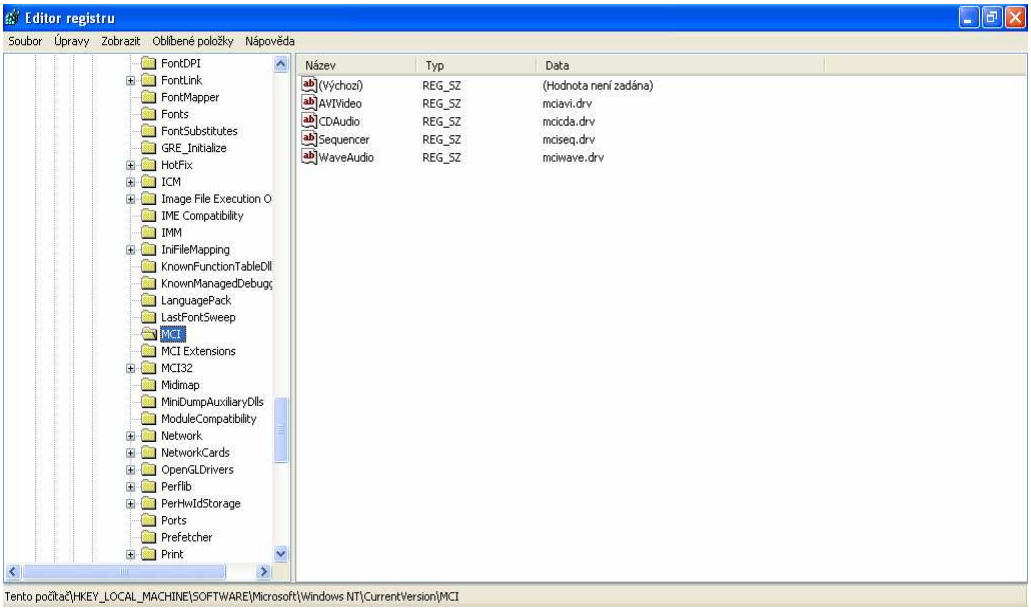
1. *CDAudio* – přehrává audio CD
2. *WaveAudio* – přehrává zvukové soubory jako WAV, MP3 atd.
3. *Digitalvideo (AVIvideo, MPEGVideo)* – přehrává digitální video
4. *Sequencer* – přehrává MIDI

Rozhraní MCI poskytuje softwarovým vývojářům jednoduše implementovat podporu multimedií v jejich programech pro prostředí Windows tím, že mu pomocí jazyka MCI mohou odesílat řetězce MCI příkazů a tím pracovat s libovolným multimediálním zařízením.

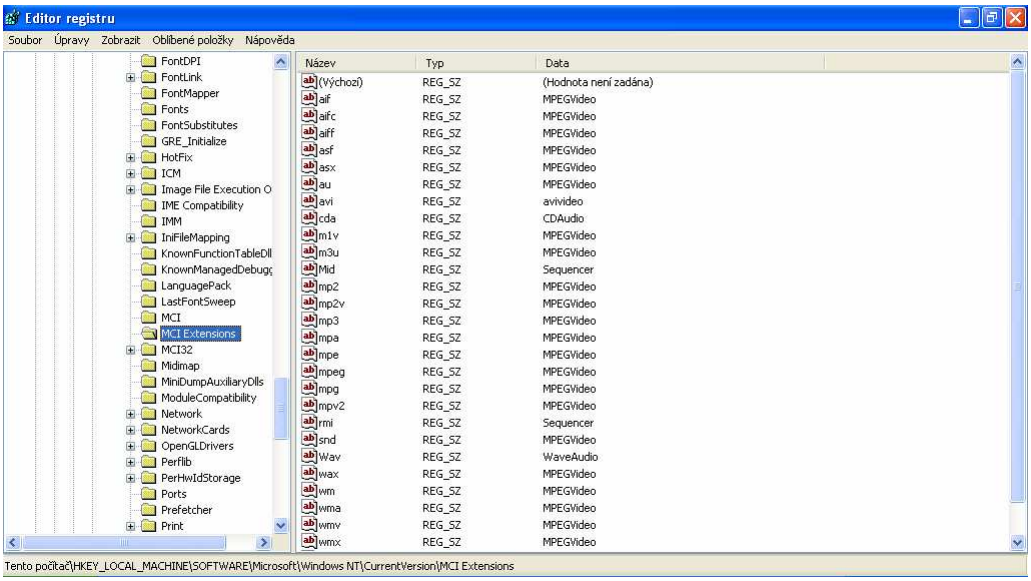
Seznam nainstalovaných zařízení můžeme najít buď v souboru *system.ini*, který se nachází v adresáři Windows, nebo v registrech Windows na adresách [6]: *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\MCI (16-bit),* viz Obr. 1.

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\ MCI32 (32-bit)

Seznam podporovaných formátů, respektive jejich přípon, lze nalézt v registrech pod adresou:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\ MCI Extensions, viz Obr. 2.



Obr. 1 - MCI zařízení - Registry



Obr. 2 – MCI – podporované přípony - Registry

3.1.4 Kodek

Kodek je zkratka ze dvou slov – **kod**ér a **dek**odér. Jedná se o zařízení či program, který multimediálnímu přehrávači nebo programu pro editaci multimédií říká, jak má

s daty tvořícími daný soubor pracovat, tj. jak je má správně přečíst, nebo naopak zapsat. Různé formáty potřebují pro své přehání různé kodeky. Pokud nemáme daný kodek nainstalovaný, přehrávač nedokáže tento soubor přehrát.

3.2 Podporované multimediální formáty a jejich předzpracování

Prostředí Imagine může díky tomu, že umí posílat MCI řetězce a tím pracovat s MCI rozhraním, přehrávat celou řadu multimediálních souborů. Obecně můžeme říci, že lze v prostředí Imagine přehrát jakýkoliv soubor, spustitelný ve Windows Media Playeru. Pokud zvolený soubor nelze přehrát, je třeba nainstalovat potřebné kodeky.

Zde jsou informace o nejpoužívanějších multimediálních formátech.

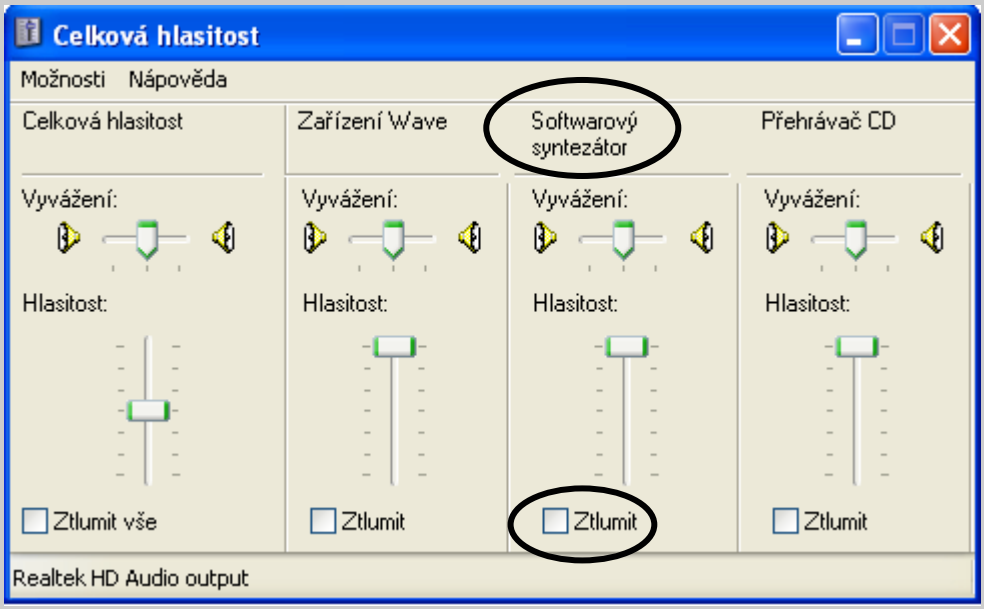
3.2.1 Audio formáty

WAV – je zkratka pro *Waveform audio formát*. Tento formát přišel společně se systémem Windows. Vzhledem k rozšíření tohoto systému je tento formát velmi oblíbený. Jelikož se zaznamenaný zvuk uchovává v nekomprimované formě, je nevýhodou tohoto formátu velikost výsledného souboru. Výhodou je naopak vysoká kvalita zvuku. Pro záznam zvuku do formátu WAV můžeme použít freeware aplikaci Záznam zvuku, která je součástí systému Windows. Tato aplikace umožňuje znamenat pouze 60 vteřin. Pro více informací o této aplikaci doporučuji přečíst [5].

MP3 – je zkratka MPEG1 AUDIO Layer 3, což je označení kompresního algoritmu odvozeného z algoritmu MPEG, používaného pro kompresi videa. Tento algoritmus po blocích zkoumá vstupní data a vynechává pro lidské ucho nerozlišitelné frekvence. Tento výsledný signál poté převede na parametrickou křivku a tím zredukuje množství dat, které je potřeba uložit [2]. Výhodou je možnost volby stupně komprese a výrazně menší velikost oproti formátu WAV.

MID, RMI – je speciální formát pro uchování a přenos MIDI informací mezi hudebními nástroji, syntezátory a počítači. Na rozdíl od formátů jako jsou WAV a MP3, které v sobě uchovávají nahraný zvuk, v sobě formát MID obsahuje pouze jakési řídící informace pro hudební události [1]. Mezi tyto události patří začátek a konec trvání noty, její výška, délka, hlasitost atd. Výhodou tohoto formátu je jeho velikost. Jelikož se do souboru zapisuje pouze informace o tom co, kdy a jakým nástrojem přehrát, je datová náročnost oproti formátu MP3 několikanásobně nižší. Nevýhodou je, že výsledný zvuk při přehrávání formátu tohoto typu závisí na zvukové kartě, zejména se kterou syntézu zvuku pracuje.

Pozn.: Pro přehrávání souborů formátu MIDI, tedy typu MID a RMI, je třeba mít v nastavení zvukové karty nastavenu hlasitost SW Syntezátor, viz Obr. 3. To proto, že při přehrávání těchto formátů vytváří zvuková karta zvuk za pomoci výše zmíněné syntézy, na rozdíl od formátů WAV a MP3, kdy zvuková karta přehrává předem zaznamenaný zvuk.



Obr. 3 Hlasitost SW Syntezátor

Mezi další podporované audio formáty patří soubory typu RM a WMA.

Pokud chceme v prostředí Imagine přehrát jiný než výše zmíněný formát, musíme ho nejprve pomocí příslušného softwaru převést na formát podporovaný. Takovémuto programu se říká konvertor a existuje jich celá řada. Mezi volně šířitelné konvertory patří například *Free MP3 WMA Converter*.

3.2.2 Video formáty

AVI – je zkratka Audio Video Interleave. Je to multimediální kontejnerový formát uvedený firmou Microsoft. Soubory formátu AVI mohou obsahovat zvukovou i video stopu, což umožňuje synchronní přehrávání. Mezi nejčastěji používané kodeky pro AVI soubory jsou DivX a Indeo [3].

MPG, MPEG – toto jsou formáty, které patří pod skupinu standardů MPEG-1, což je zkratka pro Moving Picture Experts Group, volně přeloženo jako Skupina expertů pro pohyblivý obraz. Používají se ke kódování/dekódování videa či zvuku pomocí kompresních algoritmů [4]. Jelikož se tyto přípony používají i pro formát MPEG-2, je třeba pro jejich bezproblémové přehrávání doinstalovat doplňkový dekodér video MPEG-2.

MOV – je formát souborů QuickTime vytvořený firmou Apple. Soubory v tomto formátu mohou obsahovat video, animace, grafiku a prvky virtuální reality [3].

Mezi další podporované audio formáty patří soubory typu RM a WMV, MP4.

Stejně tak jako u zvukových formátů i u videa budeme potřebovat konvertor, pokud budeme chtít přehrávat soubory jiného formátu, než Imagine podporuje. Mezi volně šiřitelé patří *Free Video Converter,* který mezi sebou umí konvertovat nejpoužívanější formáty, jako jsou například AVI, MPG, MP4, WMV, ASF atd.

4 Nástroje prostředí Imagine pro práci s audiem

V této části příručky se již dostaneme k samotnému programování v prostředí Imagine. V první části si probereme příkazy pracující s audiem a v druhé části příkazy pracující s videem.

4.1 Nástroje pro práci s MIDI

> **Pozn.:** Všechny následující příkazy pracují s MIDI, proto je třeba mít pro jejich přehrávání nastavenu hlasitost SW Syntezátor v nastavení hlasitosti, viz *Kapitola 3.2 Obr. 3.*

4.1.1 Příkaz *HudebníNástroj!*

Ještě než si řekneme něco o příkazech pro přehrávání zvuků, je třeba se seznámit s příkazem *hudebníNástroj!*.

Jeho vstupem je celé číslo z intervalu od 0 do 127 (128 MIDI nástrojů) nebo jméno hudebního nástroje (klavír, kalimba), který určuje hudební nástroj pro následující příkaz *přehraj* nebo *hrajNotu.*

```
? hudebníNástroj! 121
```

Tento příkaz nastaví pro všechny následující příkazy *přehraj* a *hrajNotu* hudební nástroj *dech.* Toto nastavení zůstane, dokud nepoužijeme jiné nastavení v příkazu *přehraj*, nebo znovu nepoužijeme příkaz *hudebníNástroj!*.

4.1.2 Příkaz *Přehraj seznam*

Příkaz *přehraj* je základním nástrojem prostředí Imagine pro práci s MIDI. Pokud tento příkaz napíšeme do příkazového řádku bez vstupního seznamu, otevře se nám pomocník pro vytvoření melodie, viz *Kapitola 4.1.5*. Pokud poté vytvoříme v pomocníku melodii a potvrdíme tlačítkem *Udělej!*, pomocník za nás do příkazového řádku za příkaz *přehraj* doplní vstupní seznam, podle toho jakou melodii jsme napsali,

a tuto melodii přehraje. Vstupní seznam může obsahovat různé znaky. Jejich seznam a význam naleznete v nápovědě aplikace.

Jako vstup pro příkaz *přehraj* je možné použít i již předem definovanou melodii, kterou máme uloženu v proměnné.

Seznam příkladů:

Příklad 1: Přehrání melodie z proměnné

V tomto příkladě přehrajeme pomocí příkazu *přehraj* část skladby „Severní vítr“ od skladatelské dvojice Svěrák-Uhlíř, kterou si nejprve uložíme pomocí příkazu *dosaď* do proměnné :*SeverníVítr*.

Vstupní seznam této melodie přepíšeme z notového zápisu buď ručně, nebo pomocí pomocníka pro vytváření melodie, viz *Kapitola 4.1.5*.

```
? dosaď “SeverníVítr [S0 I73 T120 L4 O3 C C C O2 E F G A 2A P 2P G G G
                  G F E 2D. P 2P O3 C C C O2 E F G A A P A G F E C P F
                  2D 2C.]

? přehraj :SeverníVítr
```

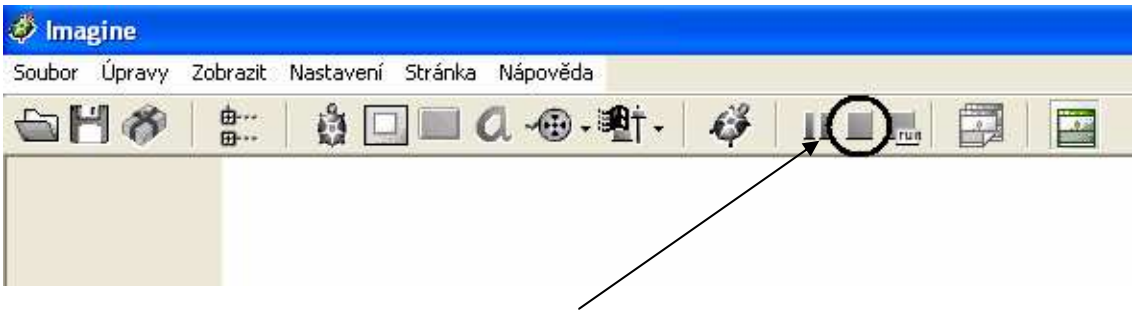
Prvním příkazem vytvoříme proměnou *SeverníVítr* a druhým příkazem tuto proměnou přehrajeme.

Příklad 2: Nekonečné přehrávání pomocí příkazu *pořád*

V tomto příkladě spustíme nekonečný proces přehrávání pomocí příkazu *pořád*. Melodie se bude neustále dokola přehrávat, dokud tento proces nezastavíme klávesou F12, příkazem *ZastavVše* a nebo tomuto příkazu odpovídajícím tlačítkem, které se nachází v horním panelu prostředí Imagine, viz Obr. 4.

```
? pořád [přehraj :SeverníVítr]
```

Tento příkaz spustí nekonečné přehrávání melodie z proměnné :*SeverníVítr*.



Obr. 5 Tlačítko Stop

Příklad 3: Nekonečné přehrávání pomocí rekurze

Ke stejnému cíli jako v předchozím příkladě můžeme dojít pomocí takzvané rekurze. Rekurze znamená, že z těla příkazu zavoláme ten samý příkaz a tím vlastně spustíme nekonečný proces. Za tímto účelem si buď z příkazové řádky nebo v paměti vytvoříme nový příkaz *Hraj*, po jehož zavolání se rozběhne nekonečný rekurzivní proces přehrávání melodie *SeverníVítr*.

```
? příkaz Hraj
> přehraj :SeverníVítr
> Hraj
> konec
? Hraj
```

Pozn.: Rozdíl mezi rekurzivním voláním a příkazem *pořád* je v tom, že když zavoláme rekurzivní příkaz pomocí objektu *Tlačítko* a jeho události

přiZapnutí, můžeme tento proces zastavit opětovným stisknutím tlačítka. To ovšem nelze při použití příkazu *pořád*, jelikož se při každém dalším stisknutí tlačítka spustí další paralelní proces (těchto souběžně spuštěných procesů může být maximálně 16, což je počet MIDI kanálů, viz *Kapitola 3.1.2*). Výhodou rekurze je také přehlednější zápis kódu při složitějších operacích.

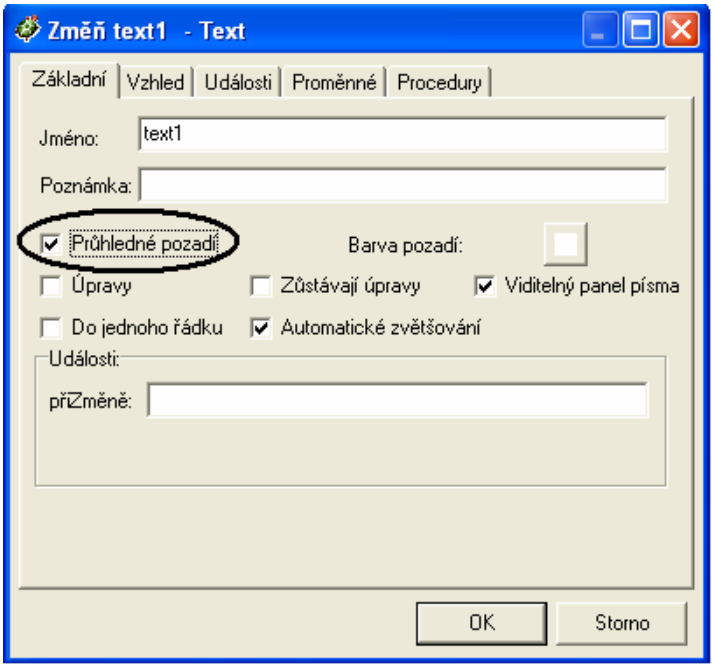
Příklad 4: Změna hudebního nástroje během přehrávání melodie

Poté co jsme se z nápovědy aplikace Imagine dozvěděli něco o struktuře vstupního seznamu pro příkaz *přehraj* a umíme spustit nekonečné přehrávání, můžeme za předpokladu, že umíme v prostředí Imagine pracovat se seznamy a slovy, měnit základní parametry melodie, jako je tempo, oktáva, délka základní noty a hudební nástroj, přímo v reálném čase při jejím přehrávání.

V tomto příkladu si tedy vytvoříme objekt *Posuvník (p1).* Pomocí tohoto posuvníku a příkazu pro úpravu seznamů *nahraď* budeme měnit hudební nástroj, kterým se melodie *SeverníVítr* přehrává. Pro zobrazení názvu nástroje použijeme objekt *Text (text1).* Pro jednodušší ovládání si vytvoříme objekt *Tlačítko – Přepínač (tl1),* které při zapnutí spustí nekonečný proces přehrávání melodie a při vypnutí tento proces ukončí. Vlastnost *Přepínač* objektu *Tlačítko* je třeba nastavit pouze při použití příkazu *pořád* pro spuštění nekonečného procesu, viz níže.

Popis objektů:

1. Objekt *Text* (*text1*) – tento objekt je zde jen pro zobrazování názvu nástroje. Pro hezčí zobrazení mu můžeme v rodném listě nastavit *Průhledné pozadí*, viz Obr. 5, a popřípadě změnit styl a velikost písma.

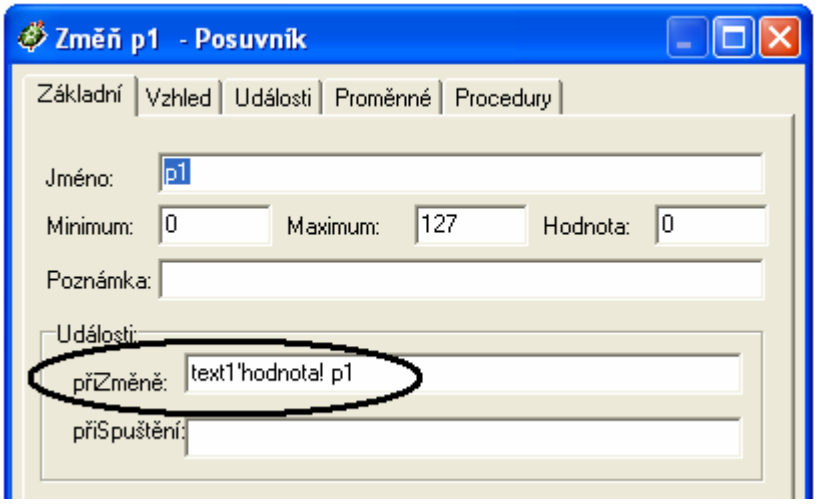


Obr. 5 Text – Průhledné pozadí

2. Objekt *Posuvník* (*p1*) – jelikož máme na výběr ze 128 nástrojů (MIDI standard), nastavíme jeho maximum na číslo 127. Abychom docílili změny hodnoty objektu *Text*, musíme v rodném listě *p1* do události *přiZměně* napsat příkaz `text1`hodnota! p1`hodnota`, viz Obr. 6. Tento kód zařídí, že se při změně *p1* nastaví jeho hodnota jako hodnota *text1*. Jelikož ale chceme aby se na místo čísla zobrazoval název hudebního nástroje, musíme tento kód poupravit.

Pozn.: Pro zrychlení práce můžeme místo zdlouhavého psaní *p1`hodnota*, pomocí tzv. *zástupné hodnoty*, viz nápověda aplikace, napsat jen *p1*. Tento způsob zápisu bude použit i dále v této příručce.

Dalším zrychlením práce může pro někoho být použití klávesové zkratky *CTRL + TAB*, která umožňuje přepínání mezi jednotlivými záložkami ať už v *Rodném listě* objektu, nebo v jakémkoliv jiném dialogovém okně, které má více záložek.



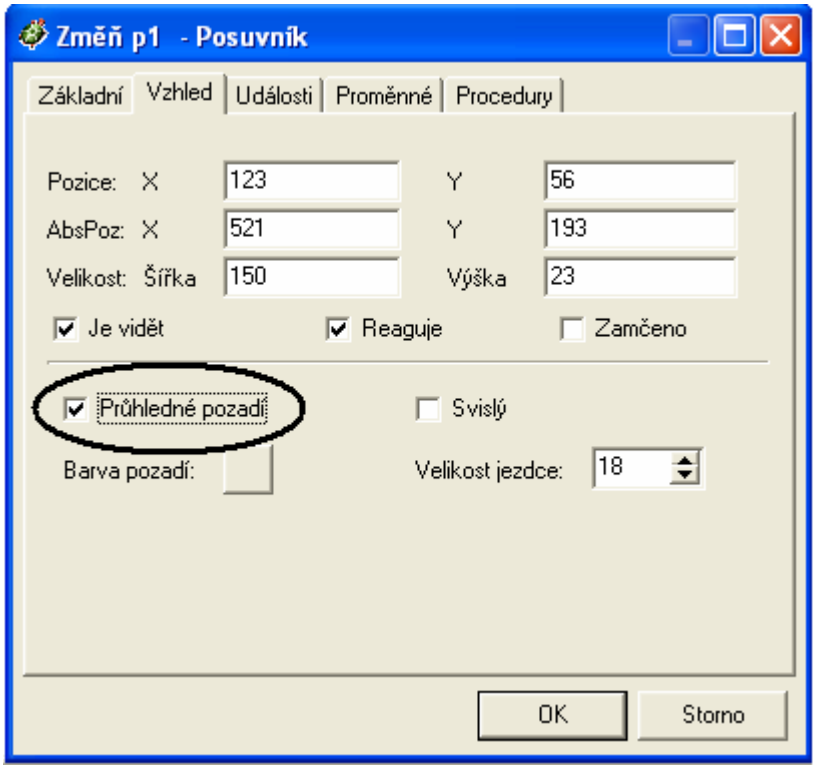
Obr. 6 Posuvník – Událost *přiZměně*

Nejprve si vytvoříme pomocí příkazu *dosaď* proměnou *nástroj*, která je seznamem názvů všech hudebních nástrojů.

```
? dosaď "nástroj [|0 klavír - křídlo| |1 otevřený klavír| |2 elektrický klavír| atd …]
```

Poté přejdeme k úpravě příkazu pro událost *přiZměně*. Tento příkaz upravíme na `text1'hodnota! prvek p1 + 1 :nástroj`, v němž příkaz *prvek* vybere prvek *p1 + 1* (jedničku přičítáme, protože hodnota *p1* se pohybuje v intervalu od 0 do 127, ale nultý prvek seznamu neexistuje) z proměnné :*nástroj* a dosadí jej jako hodnotu objektu *text1*. Pro hezčí zobrazení můžeme *p1* nastavit vlastnost *Průhledné pozadí*, viz Obr. 7.

Pozn.: Pokud se vám text nezobrazí, nebo se zobrazí pouze jeho část, je třeba zvětšit velikost jeho zobrazovacího pole. To lze provést buď v rodném listě, nebo tím že dvojklikem na objekt *text1* otevřeme Panel písma a roztáhnete jej na potřebnou velikost.
Celý seznam nástrojů naleznete na přiloženém CD v souboru *Prehraj-Pr4*, uložený v proměnné :*Nástroj*.



Obr. 7 Posuvník – Průhledné pozadí

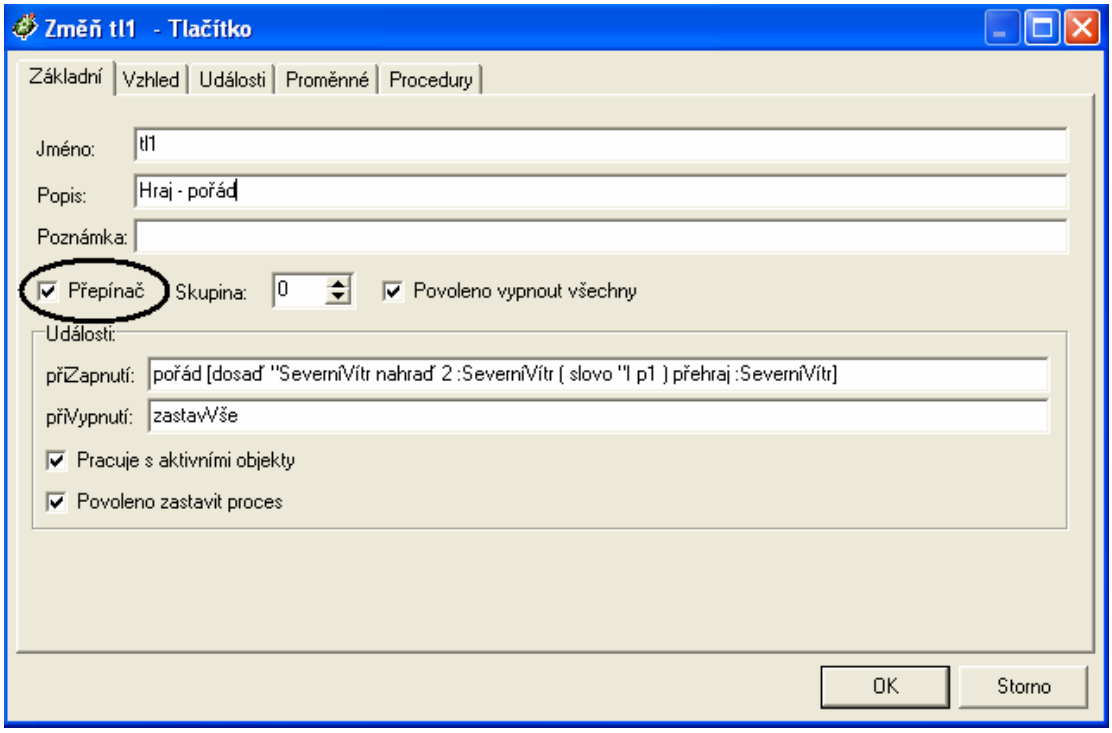
3. Objekt *Tlačítko* (*tl1*) – v rodném listě tohoto objektu si nejprve nastavíme nějaký popis například „Hraj“. Do události *přiZapnutí* zadáme příkaz, který se bude starat o změnu hudebního nástroje. Tuto změnu provedeme použitím příkazů *nahraď* a *slovo* tak, že v proměnné :*SeverníVítr* nahradíme prvek seznamu, který udává hudební nástroj, prvkem složeným ze slova I a hodnoty posuvníku *p1*. V našem případě je to druhý prvek seznamu. Abychom docílili toho, že se melodie bude přehrávat pokaždé jiným nástrojem, pokud tedy měníme hodnotu *p1*, musíme tuto změnu provádět při každém zavolání příkazu *přehraj*. Proto musí být tato změna součástí nekonečného procesu.

   Při použití příkazu *pořád* to znamená nastavit událost *přiZapnutí* na, viz Obr. 8a:

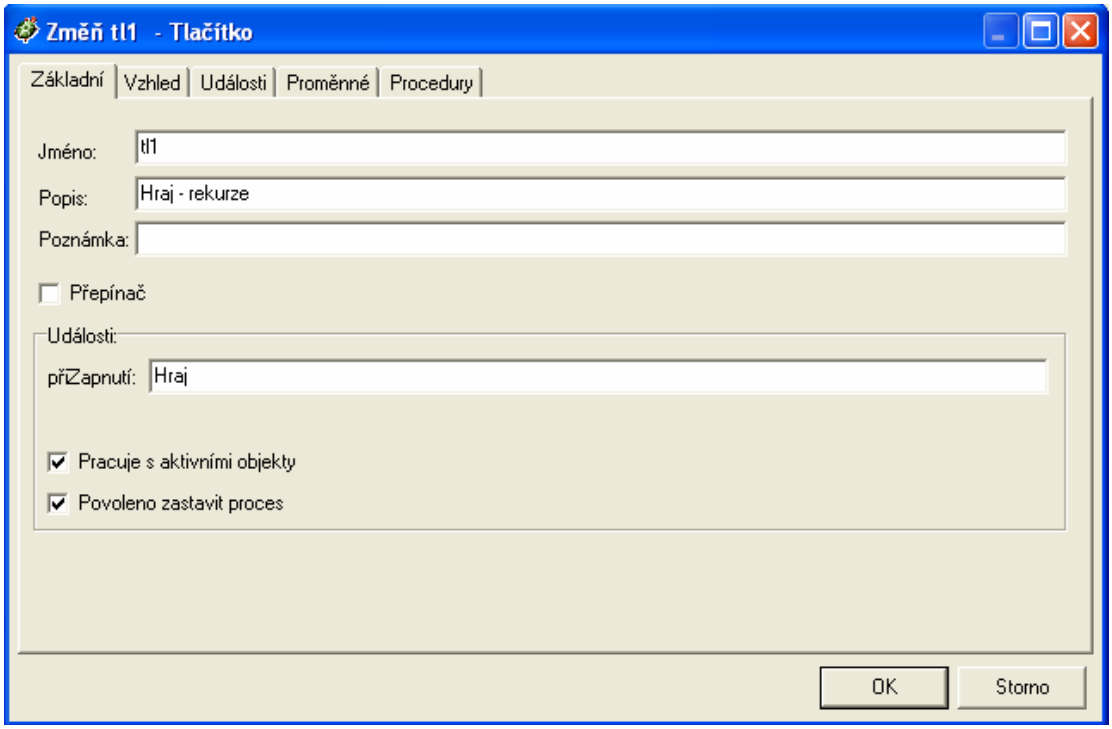
```
pořád [dosaď "SeverníVítr nahraď 2 :SeverníVítr (slovo "I p1)
přehraj :SeverníVítr]
```

Při použití rekurze to znamená nastavit událost *přiZapnutí* na *Hraj*, viz Obr. 8b, a upravit tento příkazu do podoby:

```
? příkaz Hraj
? dosaď "SeverníVítr nahraď 2 :SeverníVítr (slovo "I  p1)
? přehraj :SeverníVítr
? hraj
? konec
```

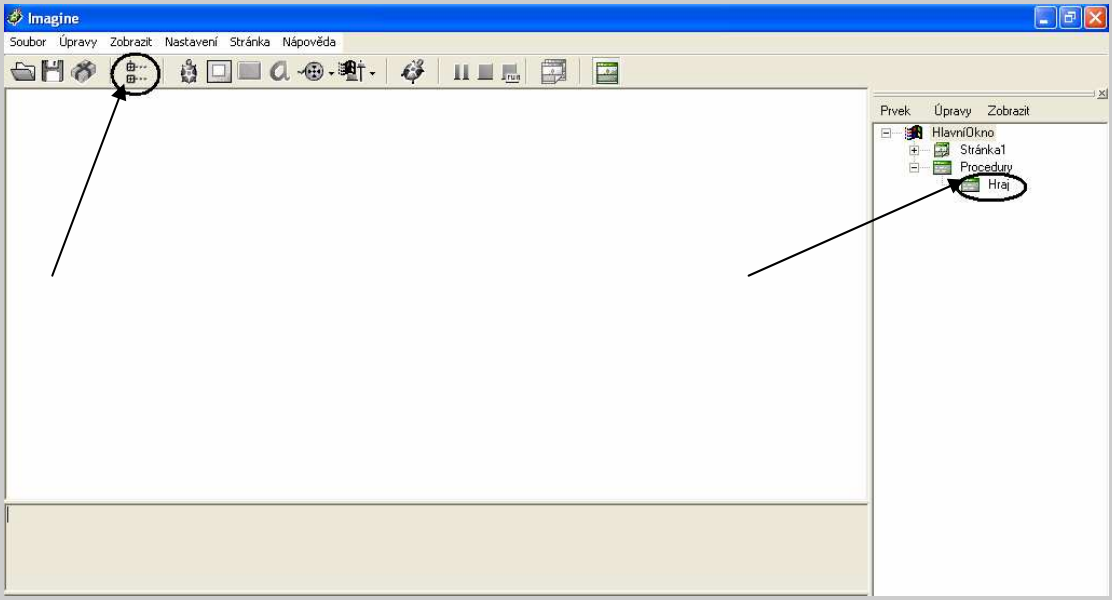


Obr. 8a Změna nástroje – objekt *Přepínač* – příkaz *pořád*



Obr. 8b Změna nástroje – objekt *Tlačítko* - rekurze

Pozn.: Již z takto jednoduchého příkazu je vidět větší přehlednost kódu rekurzivního volání oproti příkazu *pořád*.
Pro editaci příkazu Hraj můžeme použít dva způsoby. Buď napíšeme do příkazové řádky příkaz ? *uprav "hraj* a tím otevřeme *Okno Procedury*, ve kterém můžeme příkaz měnit a nebo jednoduše otevřeme *Okno Paměti*, buď pomocí tlačítka v horní liště, nebo pomocí klávesové zkratky F4, a mezi *Procedurami* najdeme požadovaný příkaz, viz Obr. 9.



Obr. 9 Okno Paměti

Pokud používáme pro nekonečný proces příkaz *pořád*, musíme nastavit objektu *tl1* vlastnost *Přepínač*, viz Obr. 8a. Tento krok nám přinese dvě výhody. První výhodou je, že pokud bychom nechali *tl1* pouze jako tlačítko, při každém jeho stisknutí by se spustilo další přehrávání té samé melodie. Druhou výhodou je, že můžeme *tl1* nastavit událost *přiVypnutí* na *zastavVše* a tím můžeme nekonečný proces přehrávání melodie zastavit.

Podobným způsobem jako v předcházejícím příkladě můžeme nejen měnit ostatní parametry melodie jako jsou tempo, oktáva, délka základní noty, ale také psát melodii v reálném čase, viz *Kapitola 6.1.2.*

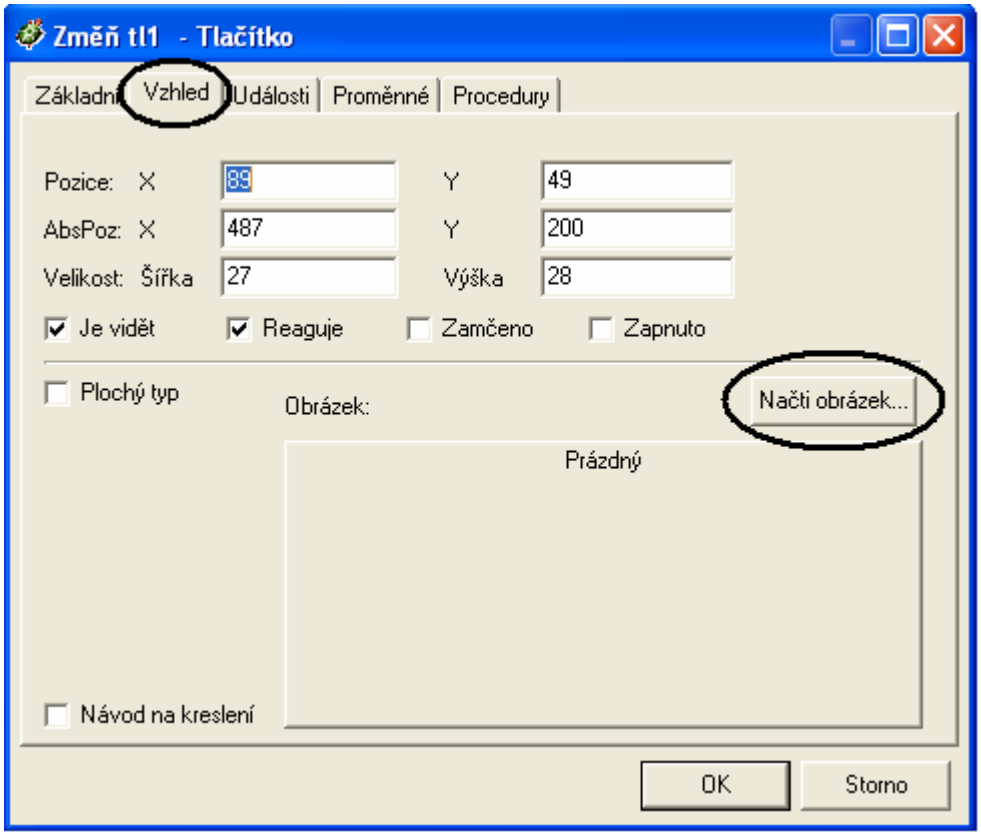
Příklad 5: Kánon

V tomto příkladu si ukážeme jak pomocí příkazu *přehraj* zahrát kánon, což je druh hudební skladby, v němž hrají dva nástroje stejnou melodii s různým zpožděním.

Za tímto účelem musíme kromě proměnné :*SeverníVitr* vytvořit objekt *Posuvník (p1)*, který nám bude udávat dobu zpoždění, objekt *Text (text1)* pro zobrazení tohoto čísla a tři objekty *Tlačítka (tl1, tl2, tl3).* První dvě tlačítka pro přehrávání melodií a třetí tlačítko pro spuštění kánonu.

Popis objektů:

1. Objekt *Text* (*text1*) – tento objekt má stejné nastavení jako objekt *text1*v minulém příkladě
2. Objekt *Posuvník* (*p1*) – tomuto objektu nastavíme v rodném listu maximum 2000. To znamená, že maximální možné zpoždění mezi první a druhou melodií budou dvě vteřiny. Dále mu jako událost *přiZměně* nastavíme zjednodušený příkaz z minulého příkladu `text1`hodnota! p1`. Jak už víme, tento příkaz se postará o zobrazení hodnoty *p1* objektem *text1*.
3. Objekt *Tlačítko* (*tl1, tl2*) – tlačítka pro spuštění melodie. V rodném listě můžeme těmto objektům nastavit buď nějaký popis, např. „Melody1“ a „Melody2“, nebo můžeme nastavit nějaký obrázek jako jejich tvar, viz Obr. 10. Do události *přiZapnutí* napíšeme příkaz pro přehrání melodie `přehraj :SeverníVítr`.



Obr. 10 Nastavení tvaru *tlačítka*

4. Objekt *Tlačítko* (*tl3*) – přepínač pro zapnutí a vypnutí kánonu. Jako popis nastavíme tomuto objektu Kánon, nastavíme ho jako *Přepínač* a do události *přiZapnutí* vložíme příkaz `tl1 čekej p1 tl2`. Tento příkaz nejprve spustí tlačítkem *tl1* první melodii, počká pomocí příkazu *čekej* tak dlouho, na jakou hodnotu je právě nastaven posuvník *p1* a poté spustí druhou melodii tlačítkem *tl2*. Nesmíme zapomenout zadat do události *přiVypnutí* příkaz *zastavVše*, který vypne všechny běžící procesy a tím i přehrávání.

Dalším možným rozšířením tohoto projektu je přidání objektu *Tlačítko* pro třetí melodii a objektů typu *Posuvník*, kterými by se měnili hudební nástroje jednotlivých melodií.

Výše zmíněná možnost není ale jedinou cestou, jak kánon zahrát. Jak už víme, pokud nastavíme tlačítku do události *přiZapnutí* nekonečný proces pomocí příkazu

pořád, můžeme tento proces spouštět několikrát za sebou stisknutím příslušného tlačítka. Pokud bychom tedy vzali minulý přiklad, tlačítku *tl1* bychom nastavili do události *přiZapnutí* nekonečný proces příkazem `pořád [přehraj :SeverniVitr]` a tlačítku *tl3* bychom nastavili do události *přiZapnutí* příkaz `tl1 čekej p1 tl1` docílili bychom stejného cíle za použití pouze dvou tlačítek.

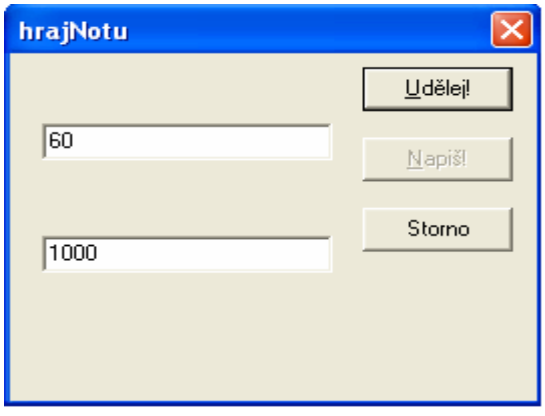
Pokud bychom chtěli, aby se melodie hrála třikrát, stačilo by rozšířit tlačítku *tl3* událost *přiZapnutí* na `tl1 čekej p1 tl1 čekej p1 tl1`. Pokud bychom přidali do projektu další objekt typu *Posuvník*, bylo by možné nastavovat délku jednotlivých zpoždění příkazem `tl1 čekej p1 tl1 čekej p2 tl1`. Problém by nastal, pokud bychom chtěli přehrávat každou melodii jiným hudebním nástrojem.

4.1.3 Příkaz *HrajNotu kód čas*

Hraj notu je dalším příkazem, který pracuje s MIDI. Pokud tento příkaz napíšeme do příkazového řádku bez parametrů, otevře se nám pomocník pro *hrajNotu*, viz Obr. 11. Poté co doplníme parametry, tedy kód noty a délku jejího hraní, pomocník je za nás dopíše za příkaz. Příkaz se poté spustí a přehraje zvolený tón hudebním nástrojem, který jsme nastavili předcházejícím příkazem *přehraj*, nebo *hudebníNástroj!*. Seznam kódů tónů lze nalézt v nápovědě aplikace.

```
? hrajNotu 60 1000
```

Tento příkaz zahraje notu C po dobu jedné vteřiny.



Obr. 11 Pomocník *hrajNotu*

Seznam příkladů:

Příklad 1: Jak zahrát náhodnou melodii

V tomto příkladu si vytvoříme jediný objekt typu *Tlačítko (tl1)*, které nám pomocí příkazu *hrajNotu* přehraje náhodnou melodii tak, že počet jednotlivých tónů, jejich hodnoty a délky budeme generovat pomocí příkazu *náhodně*. Příkaz *náhodně* generuje náhodné číslo v intervalu od 0 do číslo – 1, viz nápověda aplikace. Pro zjednodušení zápisu kódu použijeme příkaz *opakuj*.

Popis objektu:

1. Objekt *Tlačítko (tl1)* – jako událost *přiZapnutí* vložíme příkaz `opakuj 1 + náhodně 20 [hrajNotu náhodně 129 náhodně 2000]`. Tím, že jsme dali příkazu *opakuj* jako vstup *1 + náhodně 20*, jsme docílili toho, že se nám zahraje melodie dlouhá minimálně jeden a maximálně dvacet tónů. Tento atribut můžeme samozřejmě měnit dle libosti stejně tak jako vstup pro *hrajNotu*. Když například změníme tento příkaz na `hrajNotu 40 + náhodně 89`, docílíme toho, že nám budou hrát pouze noty z intervalu 40 – 128. Musíme si ale dát pozor, aby generované číslo nebylo nikdy větší než 128.

Příklad 2: Jak zahrát akord

V tomto příkladu si ukážeme, jak pomocí příkazů *hrajNotu* a *přehraj* zahrát souzvuk tří tónů, tedy akord.

Postup:

1. Z nápovědy aplikace Imagine se dozvíme, že pokud je prvním vstupem příkazu *hrajNotu* seznam kódů, přehraje se akord. Pokud tedy chceme zahrát akord pomocí tohoto příkazu, stačí napsat do příkazového řádku `? hrajNotu [60 62 64] 1000` a příkaz by měl zahrát akord složený z not C, D, E po dobu 1 vteřiny.
2. Složitější situace nastává, pokud chceme tentýž akord zahrát pomocí příkazu *přehraj*. Ten totiž nepřehrává vstupní seznam najednou ale postupně. Jelikož ale máme k dispozici až 16 MIDI kanálů, můžeme problém vyřešit tím, že spustíme tento příkaz třikrát najednou. Za tímto účelem musíme vytvořit tři objekty *Tlačítko (tl1, tl2 a tl3).* Každému z nich nastavíme událost *přiZapnutí* na `přehraj [nota]`, kde parametr *nota* bude u každého tlačítka jiný. V našem případě pro *tl1* nota C, pro *tl2* nota D a pro *tl3* nota E. Abychom hráli ve stejné oktávě jako při použití příkazu `hrajNotu [60 62 64]`, musíme alespoň jednomu tlačítku nastavit událost *přiZapnutí* na `přehraj [O2 nota].` Tímto nastavíme hlavní oktávu, ve které budou hrát všechna tlačítka. Poté vytvoříme další objekt *Tlačítko (tl4)*, které bude mít jako událost *přiZapnutí* nastaven příkaz `tl1 tl2 tl3`. Tento příkaz najednou zapne všechny tyto tlačítka a tím se přehraje akord. Pro zobrazení názvů nástrojů použijeme proměnnou :*nástroj* a událost *přiZměně* posuvníku jako v *Příkladu 4 Kapitoly 4.1.2.*

> **Pozn.:** Pro zrychlení práce můžeme při vytváření tlačítka pro přehrání tónu použít objekt *Melodie,* viz *Kapitola 4.1.5.*

Příklad 3: Jak naladit akord + změna nástroje

Když jsme v předchozím příkladu zahráli akord C, D, E pomocí příkazu `hrajNotu [60 62 64 ]` a poté pomocí třech příkazů `přehraj [nota],` byl pokaždé výsledný

zvuk jiný. V dalším příkladě si tedy zkusíme „naladit“ správný akord a přidáme si možnost hraní každého tónu jiným hudebním nástrojem po jinou délku noty.

K tomu abychom mohli měnit tóny, které příkaz *hrajNotu* přehrává, a tím naladit akord, použijeme stejného postupu jako v minulém příkladě při hraní akordu pomocí příkazu *přehraj*.

Postup:

Nejprve tedy vytvoříme tři objekty typu *Tlačítko (tl1, tl2, tl3),* které budou hrát jednotlivé tóny akordu a poté vytvoříme další objekt *Tlačítko (tl4),* který je bude spouštět. Dále budeme potřebovat tři objekty typu *Posuvník (p1, p2, p3),* kterými budeme nastavovat kódy jednotlivých not. Tyto kódy budeme zobrazovat pomocí třech objektů typu *Text (text1, text2, text3).* Stejným postupem vytvoříme posuvníky a texty pro změnu délky noty (*p4, p5, p6, text4, text5, text6*) a pro změnu hudebního nástroje (*p7, p8, p9, text7, text8, text9*). Dohromady tak budeme mít devět objektů typu *Posuvník* a devět objektů typu *Text.* Pro zjednodušení práce si vytvoříme navíc další tři objekty typu *Tlačítko (tl5, tl6, tl7).* Každý z nich bude sloužit k vyrovnání posuvníků, kterými se nastavují jednotlivé parametry, na stejnou hodnotu.

Popis objektů :

1. Objekty *Text (text1* až *text9)* – tyto objekty mají stejné nastavení jako objekty stejného typu v *Příkladě 4 Kapitoly 4.1.2*
2. Objekty *Posuvník (p1, p2, p3 - p4, p5, p6 – p7, p8, p9)* – všechny tyto objekty mají obdobné nastavení. Pro změnu hodnot příslušných textů použijeme příkaz z *Příkladu 5* předchozí kapitoly. Posuvník *p1*, *p2* a *p3* udává kód jednotlivého tónu. Nastavíme tedy jeho minimum na 0 a maximum na 128. Posuvníky *p4*, *p5 a p6* se starají o změnu délky noty. Minimum nastavíme tedy na 0 a maximum na 2000. Posuvníky *p7, p8 a p9* udávají číslo jednotlivého hudebního nástroje. Jeho minimum bude tedy 0 a maximum 127. Pro zobrazení názvu hudebního nástroje namísto čísla, použijeme stejný postup jako v *Příkladě 4 Kapitoly 4.1.2.*

3. Objekty *Tlačítko (tl1, tl2, tl3)* – všechny tyto objekty budou mít nastavenu stejnou událost *přiZapnutí*, která se bude lišit jen jmény posuvníků, které nastavují jednotlivé hodnoty. Abychom tlačítkem *tl1* přehráli tón *p1*, po dobu *p4*, hudebním nástrojem *p7*, musíme do události *přiZapnutí* tlačítka *tl1* vložit příkaz `hudebníNástroj! p7 hrajNotu p1 p4`. Modifikovaný příkaz poté vložíme i do události *přiZapnutí* tlačítek *tl2 a tl3.*
4. Objekty *Tlačítko (tl5, tl6, tl7)* – objekt *tl5*, bude mít nastavenu událost *přiZapnutí* na `pro [p8 p9] [hodnota! p7].` To znamená, že po jeho stisknutí se hodnoty posuvníků *p8* a *p9* nastaví na hodnotu posuvníku *p7*. Obdobný příkaz mají ve své události *přiZapnutí* i objekty *tl5* a *tl6.*
5. Objekt *Tlačítko (tl4)* – tento objekt je stejný jako tlačítko *tl4* v předchozím příkladu. Do události *přiZapnutí* vložíme příkaz `tl1 tl2 tl3`, který spustí všechny tlačítka najednou a tak přehraje souzvuk tří tónů. Dokážete naladit akord?

> **Pozn.:** Někoho by mohlo napadnout, že jednodušší způsob „lazení“ akordu by byl, kdybychom tlačítku *tl4* daly jako událost *přiZapnutí* příkaz `hrajNotu [p1 p2 p3] 1000`. Logicky by se mohlo zdát, že by se poté měl zahrát akord složený z hodnot posuvníků *p1*, *p2* a *p3*. Není tomu ale tak, protože v případě že zadáme jako vstup příkazu *hrajNotu* seznam kódů, příkaz *hrajNotu* uznává pouze čísla jako vstup.

4.1.4 Příkaz *Pauza*

Příkaz *pauza* je jakýmsi doplňkovým příkazem k příkazu *hrajNotu*. Tento příkaz pozastaví posloupnost not přehrávaných příkazem *hrajNotu*. Jeho vstupem je číslo, které určuje čas trvání pauzy v milisekundách.

Příklad 1: Jak zahrát tři tóny se zpožděním

V tomto příkladě modifikujeme předchozí příklad tak, aby se nám tóny hrané tlačítky *tl1, tl2, tl3* hráli pomocí příkazu *pauza* po sobě a ne najednou.

K tomu budeme potřebovat dva objekty *Posuvník (p10 a p11),* kterými budeme nastavovat délku pauzy, dva objekty typu *Text (text10* a *text11)* , které nám budou sloužit pro zobrazení hodnot *p10* a *p11* a konečně objekt *Tlačítko (tl9),* kterým budeme spouštět přehrávání.

Popis objektů:

1. Objekty *Text (text10* a *text11)* – tyto objekty mají stejné nastavení jako objekty stejného typu v příkladu, který modifikujeme. Nejrychlejší je tedy vytvořit jejich kopii.
2. Objekty *Posuvník (p10 a p11)* – tyto objekty se starají o nastavení délky pauzy. Jejich nastavení tak můžeme zdědit od objektu *p4*, *p5* nebo *p6*. Nesmíme však zapomenout upravit odpovídající jména *textů*, které se mají měnit.
3. Objekt *Tlačítko (tl9)* – v rodném listu tohoto tlačítka si nastavíme popis například na „Pauza“ a vložíme jako událost *přiZapnutí* příkaz `tl1 pauza p10 tl2 pauza p11 tl3`. Z něj je zřejmé co se bude při stisknutí tlačítka dít. Nejprve se spustí tón tlačítka *tl1*, po kterém následuje pauza dlouhá podle hodnoty *p10,* poté se spustí tón tlačítka *tl2*, pauza dlouhá podle hodnoty *p11* a nakonec se přehraje tón tlačítka *tl3.*

Pozn.: Ke stejnému výsledku dojdeme, pokud místo příkazu *pauza* použijeme příkaz *čekej*.

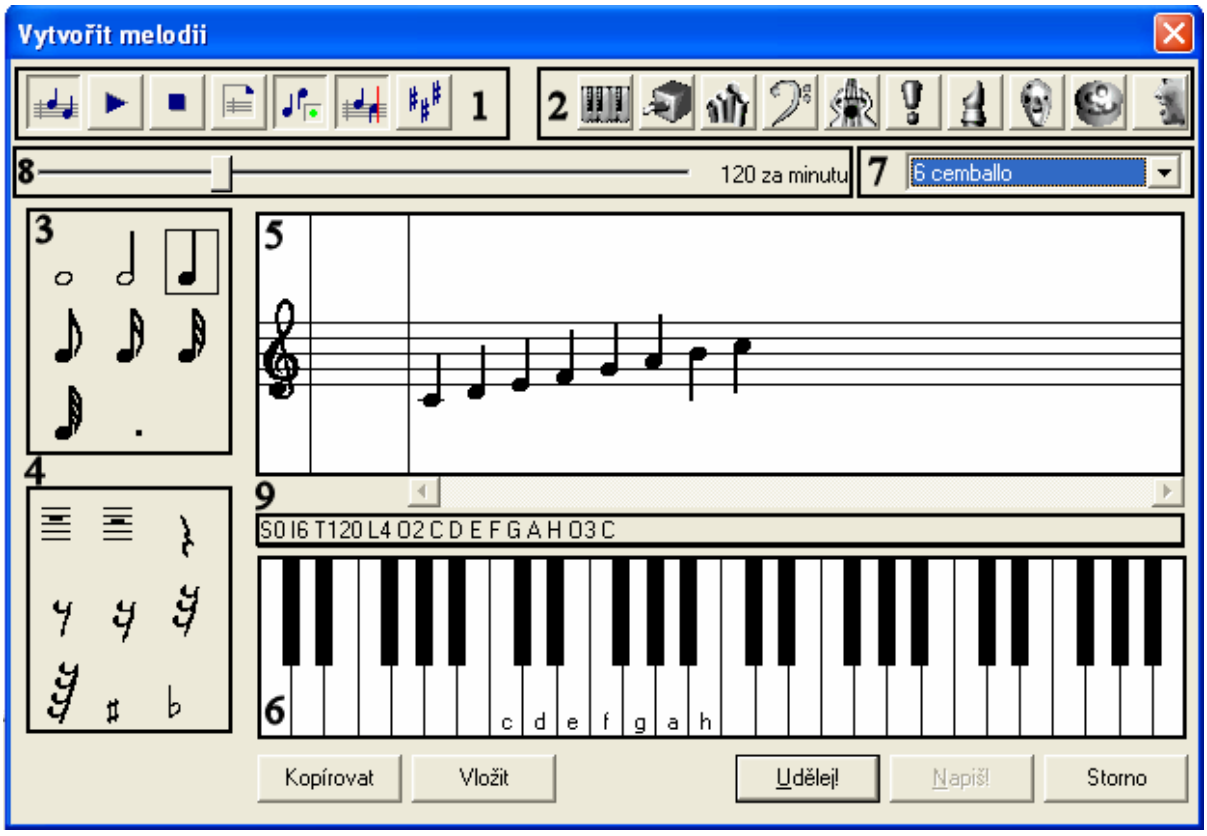
4.1.5 Objekt *Melodie*

Výše zmíněné nástroje pro práci s multimédii patří do kategorie příkazů. Nyní si povíme něco o multimediálním objektu prostředí Imagine pracujícím s MIDI.

Pokud vybereme v horní liště prostředí Imagine v záložce *Nový objekt media* objekt *Melodie*, viz Obr. 12, změní se kurzor myši na tvar notového zápisu a po kliknutí na plochu se otevře dialogové okno *Vytvořit melodii,* viz Obr.13. Jedná se vlastně o grafický zápis vstupního seznamu melodie pro příkaz *přehraj,* viz *Kapitola 4.1.2.*

Okno *Vytvořit melodii* nám usnadňuje tvorbu melodie a umožňuje snadnou úpravu jejích parametrů, ne však v reálném čase jako v *Příkladě 4 Kapitoly 4.1.2*.

Obr. 12 Objekt Melodie



Obr. 13 Okno *Vytvořit melodii*

Popis okna *Vytvořit melodii*:

1. Panel ovládání. První tlačítko zleva zapíná/vypíná zápis not z klávesnice. Druhé tlačítko přehraje zapsanou melodii a třetí tlačítko toto přehrávání zastaví. Čtvrté tlačítko smaže melodii. Páté tlačítko zapíná/vypíná zvuk během vytváření melodie. Šesté tlačítko zapíná/vypíná ukazatel průběhu

přehrávání melodie a poslední tlačítko otevře dialogové okno, pro výběr stupnice.

2. Panel pro výběr několika přednastavených hudebních nástrojů.
3. Panel pro výběr noty.
4. Panel pro výběr pomlky
5. Panel notové osnovy ve kterém se tvoří melodie
6. Panel pro tvorbu melodie pomocí klaviatury
7. Lišta pro výběr hudebního nástroje
8. Posuvník pro nastavení tempa melodie
9. Panel, který zobrazuje seznam, tak jak se předává příkazu *přehraj*

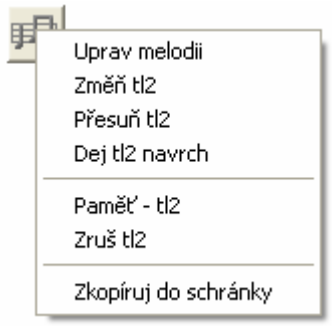
Přidávání not do notové osnovy:

1. Nejprve v levé horní části okna, číslo 3 na Obr. 13, zvolíme hodnotu noty a poté klikneme do notové osnovy, číslo 5 na Obr. 13 tam, kam chceme tuto notu umístit.
2. Přetáhneme požadovanou notu přímo na její pozici na notové osnově.
3. Vkládáme noty pomocí klaviatury.

- Pomlky lze do skladby přidávat pouze pomocí způsobu ad. 2.
- Noty i pomlky můžeme libovolně tahat po notové osnově a tím měnit jejich výšku.
- Pro smazání odtáhneme notu či pomlku ven mimo notovou osnovu.

Pokud stiskneme tlačítko OK, melodie se uloží jako vlastní proměnná :*melodie* tlačítka, které se objeví na ploše přesně tam, kde jsme klikli myší při vytváření objektu *Melodie*. Toto tlačítko má tvar notového zápisu.. Po stisknutí tlačítka se melodie začne

přehrávat. Pokud chceme melodii změnit, stačí na tlačítko kliknout levým tlačítkem myši a zvolit volbu *Uprav melodii*, viz Obr. 14.



Obr. 14 Úprava melodie

S tímto objektem se dá pracovat stejně jako s objekty typu *Tlačítko* v *Kapitole 4.1.2.* Pokud ovšem chceme měnit nějaké parametry melodie pomocí objektu *Posuvník,* v našem případě hudební nástroj, musíme objektu *p1* nastavit událost *přiZměně* nebo *přiSpuštění* na:

```
tl1'aťJeVlastní "melodie nahraď 2 tl1'melodie slovo "I p1
```

Tento příkaz při změně či spuštění posuvníku *p1* nahradí druhý prvek objektové proměnné :*melodie,* uložené v tlačítku *tl1,* slovem *I + p1* hodnota a tím změní hudební nástroj, kterým se bude melodie přehrávat. Stejným způsobem se dají měnit všechny ostatní parametry melodie.

4.2 Nástroje pro práci s audio soubory

Pozn.: Protože všechny další příkazy a objekty, které budeme používat, pracují s multimediálními soubory, které budeme načítat z disku, chtěl bych upozornit ty, kteří používají jinou distribuci operačního systému Windows než českou, že se při načítání souborů, které obsahují háčky, může vyskytnout problém. Takovýto soubor je třeba nejprve přejmenovat.

4.2.1 Příkaz *PřehrajZvuk*

Na rozdíl od všech předchozích příkazů a objektů, které pomocí MIDI vytvářely zvuky, je příkaz *přehrajZvuk* prvním ze dvou příkazů prostředí Imagine, které dokáží přehrát již uloženou zvukovou stopu. Příkaz *přehrajZvuk* pracuje pouze se soubory typu WAV.

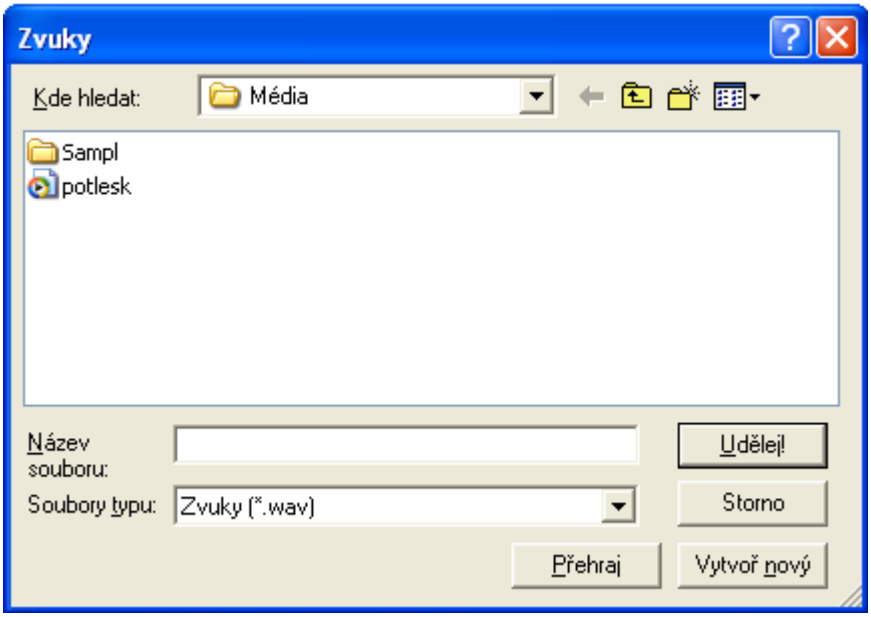
Pokud napíšeme tento příkaz do příkazového řádku bez parametrů, otevře se nám dialogové okno *Zvuky*, ve kterém můžeme vyhledat požadovaný WAV soubor, viz Obr. 15. Po potvrzení souboru dvojklikem nebo tlačítkem *Udělej!* se jeho adresa doplní do příkazového řádku za příkaz *přehrajZvuk* a přehraje se. Pokud ovšem příkaz *přehrajZvuk* napíšeme bez vstupu do události *přiZapnutí* objektu *Tlačítko*, dialogové okno pro výběr souboru se neotevře a prostředí Imagine ohlásí chybu.

```
? přehrajZvuk "potlesk.wav
```

Tento příkaz přehraje zvukovou stopu uloženou v souboru potlesk.wav.

> **Pozn.:** Přehrávání spuštěného souboru nelze zastavit ani klávesou *F12* ani příkazem *zastavVše*. Pokud tedy spustíme nekonečný proces pomocí příkazu *pořád [přehrajZvuk "soubor]*, způsobíme si nemalé nepříjemnosti. Zkuste to!

Pro použití tohoto příkazu v projektu viz *Kapitola 6.1.3*.



Obr. 15 Dialogové okno příkazu *přehrajZvuk*

4.2.2 Příkaz *PřehrajSoubor*

Příkaz *přehrajSoubor* je druhý z příkazů jazyka Imagine pracující s již uloženým audiem. Pokud jej bez parametrů napíšeme do příkazové řádky, otevře se stejně jako u příkazu *přehrajZvuk* dialogové okno *Zvuky*, které nám umožňuje najít soubory typu WAV a MP3, viz Obr. 16. Pokud jej napíšeme bez parametrů jako událost *přiZapnutí* objektu T*lačítko*, chová se stejně jako příkaz *přehrajZvuk.*

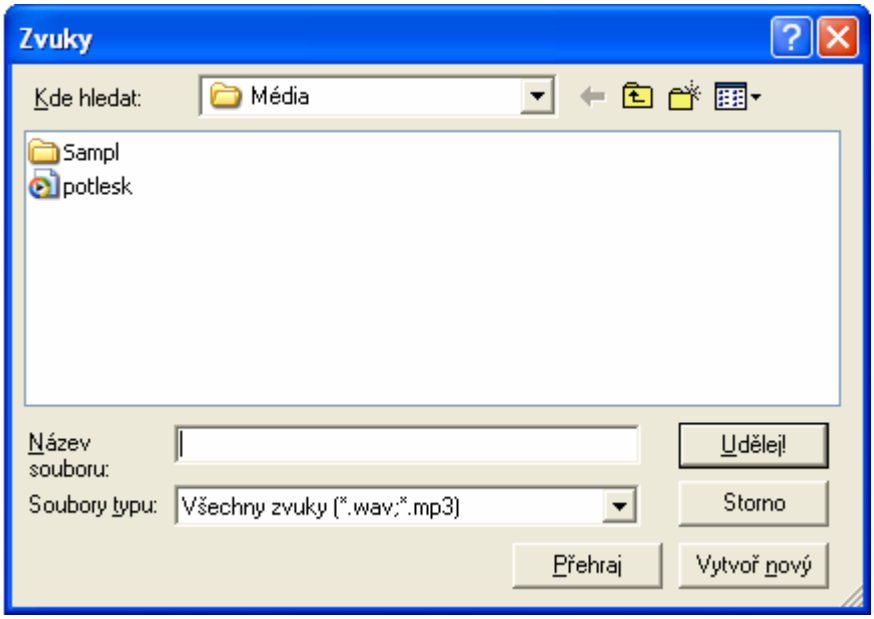
```
? přehrajSoubor “potlesk.wav
```

Tento příkaz přehraje zvukovou stopu uloženou v souboru potlesk.wav.

Pozn.: Jelikož tento příkaz odevzdává svůj vstup MCI rozhraní, můžeme jím kromě souborů WAV a MP3 přehrávat i jakýkoliv podporovaný audio formát (WMA, MID, RMI atd.). Upozorňuji však, že přehrání souboru typu CDA, což je hudební CD, může způsobit nemalé potíže tím, že toto přehrávání lze zastavit pouze restartem počítače. (Testováno na systému Windows XP Pro SP2)

Díky spolupráci s MCI rozhraním lze příkaz *přehrajSoubor* využít i

k přehrávání video souborů. Jelikož ale nelze nastavovat parametry přehrávacího okna, jako například při použití příkazu *startVideo* nebo objektu *Videoklip*, viz nápověda aplikace, je tento způsob přehrávání videa nepraktický.



Obr. 16 Dialogové okno příkazu *přehrajSoubor*

Kromě toho, že příkaz *přehrajSoubor* umí přehrávat celou škálu multimediálních formátů, má jeho použití oproti příkazu *přehrajZvuk* výhodu i v tom, že lze zastavit přehrávání souboru a to buď příkazem *zastavVše*, nebo klávesou F12.

Nevýhodou je však fakt, že příkaz *přehrajSoubor* zastaví vykonávání dalších instrukcí, dokud se všechny vstupní soubory nepřehrají. Tento problém lze vyřešit pomocí příkazu `odstartujPo 0 [přehrajSoubor “soubor]`, který spustí paralelní proces přehrávání souboru.

Pozn.: Užití příkazu *odstartujPo 0* není chybou. V nápovědě aplikace se sice doporučuje využití příkazu *odstartuj*, ale ten jaksi prostředí Imagine nezná.

Seznam příkladů:

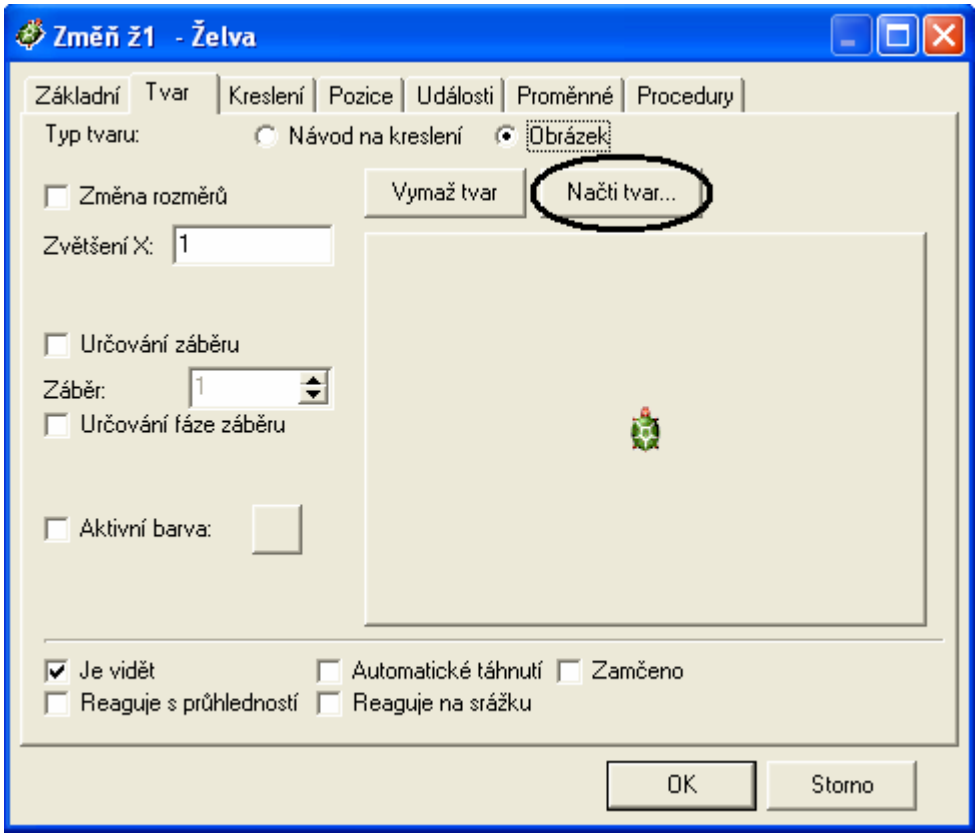
Příklad 1: Demonstrace rozdílů mezi příkazy *přehrajZvuk a přehrajSoubor*

V tomto příkladu si názorně předvedeme rozdíly mezi příkazy *přehrajZvuk* a *přehrajSoubor*.

Pro přehrání souborů použijeme dva objekty typu *Tlačítko (tl1, tl2)*. Jeden bude přehrávat soubor pomocí příkazu *přehrajZvuk* a druhý pomocí příkazu *přehrajSoubor*. Pro znázornění využijeme objekt *Želva* (*ž1*), který po tisknutí tlačítka začne v nekonečném procesu obíhat kružnici.

Popis objektů:

1. Objekt *Želva (ž1)* – tento objekt nemá žádné události a jeho nastavení je čistě na naší fantazii. Můžeme želvě změnit barvu pera pomocí příkazu *bp! "barva*, tloušťku pera pomocí příkazu *tp! číslo*, nebo jí v rodném listě nastavit zcela jiný tvar, viz Obr. 17.



Obr. 17 Nastavení tvaru *želvy*

2. Objekty *Tlačítko (tl1, tl2)* – do události při zapnutí dáme tlačítku *tl1* příkaz `přehrajZvuk "soubor.wav  ž1`pořád [do 2 vl 2 čekej 4]` a tlačítku *tl2* příkaz `přehrajSoubor "soubor.wav ž1`pořád [do 2 vl 2 čekej 4]`.

Po stisknutí *tl1* vidíme, respektive slyšíme, že se spustí zároveň přehrávání souboru *soubor.wav* a nekonečný proces pro ž*1*. Když stiskneme klávesu F12 pro zastavení všech procesů, *ž1* se zastaví, ale přehrávání pokračuje. Pokud tlačítko stiskneme znovu, želva zdvojnásobí svůj pohyb a soubor se začne přehrávat od začátku.

Naopak při stisknutí *tl2* se nejprve přehraje *soubor.wav* a teprve poté se spustí nekonečný proces pro ž*1*. Pokud stiskneme klávesu F12 během přehrávání souboru, přehrávání se ukončí.

Příklad 2: Jednoduchý hudební přehrávač

V tomto příkladu si vytvoříme jednoduchý hudební přehrávač. K načítání audio souborů využijeme příkazu *pomocníkNačtiSoubor,* viz nápověda aplikace. Tyto soubory poté sestavíme do playlistu a ten pomocí několikanásobného volání příkazu *přehrajSoubor* postupně přehrajeme. Pro jednoduchost vytvoříme přehrávač pouze pro tři písně.

Popis projektu:

Budeme potřebovat celkem osm objektů typu *Tlačítko.* Tři z nich *(tl1, tl2, tl3)* budou sloužit k vyhledání písně a další tři *(tl4, tl5, tl6)* budou vybranou píseň přidávat do playlistu. Tlačítko *tl7* tento playlist přehraje a tlačítko *tl8* jej naopak smaže. Dále budeme potřebovat čtyři objekty typu *Text*. Tři z nich *(text1, text2, text3)* budou zobrazovat adresy vybraných písni a poslední *(text4),* bude zobrazovat playlist.

Popis proměnných:

Dalším krokem je vytvoření proměnných, které budeme v tomto projektu využívat. První tři proměnné :*soubor1*, :*soubor2* a :*soubor3* v sobě budou uchovávat úplnou adresu ke zvoleným písním. Další proměnnou je pomocná proměnná :*pom*, kterou budeme využívat při načítání písní. Proměnná :*playlist* bude sloužit ke zobrazování playlistu a poslední proměnou, kterou budeme používat, je proměnná :*událost*, kterou použijeme pro nastavení události *přiZapnutí* tlačítka *tl7.*

Nyní už jen zbývá vytvořit příkazy pro načítání souborů, vytváření a rušení playlistu.

Popis příkazů:

1. *vyber :číslo* – kde parametr *číslo* udává číslo proměnné :*soubor*, do které se bude adresa souboru zapisovat a číslo objektu *Text*, kterým se tato adresa bude zobrazovat. Tento způsob jsme použili pro zjednodušení zápisu. Takto napíšeme

událost *přiZapnutí* tlačítku *tl1* pouze příkaz *vyber 1*, tlačítku *tl2* pouze příkaz *vyber 2* atd.

```
dosaď "pom pomocníkNačtiSoubor [] [Vyber audio soubor] [|Audio
Soubory@*.wav;*.mp3;*.wma|]
```

O příkazu *pomocníkNačtiSoubor* a jeho vstupech se můžeme dočíst v nápovědě aplikace. Při jeho zavolání se otevře dialogové okno *Otevřít*, které umožňuje vybrat libovolný soubor podle zvoleného vstupu. Zde jsme nastavili vstup tak, aby nám umožnil vybrat jeden z formátů typu WAV, MP3 a WMA. Pokud chceme v přehrávači přehrávat soubory jiného typu, stačí jejich přípony dopsat ve správném formátu do příkazu. Výsledek operace *pomocníkNačtiSoubor*, tedy úplná adresa vybraného audio souboru, se uloží do pomocné proměnné :*pom*.

```
když :pom <> [] [dosaď (slovo "soubor :číslo) :pom
pro (slovo "text :číslo) [hodnota! :pom]]
```

Tuto proměnnou poté otestujeme, zda není prázdná. Pokud není, uložíme její hodnotu do příslušné proměnné :*soubor*, definované parametrem :*číslo*. Tento test provádíme proto, abychom si při nechtěném otevření dialogového okna pro výběr souboru a jeho opětovném zrušení, nesmazali stávající hodnotu příslušné proměnné :*soubor*. Nakonec ještě nastavíme hodnotu příslušného *Textu* na hodnotu proměnné :*pom*, pokud tedy touto hodnotou není prázdný seznam.

2. *play :číslo* – kde parametr číslo udává číslo proměnné :*soubor*, která se přidá do playlistu. Parametr jsme použili ze stejného důvodu jako u příkazu *vyber :číslo*.

```
dosaď "událost (věta :událost "přehrajSoubor (slovo ":soubor
:číslo))
tl4'událost! "přiZapnutí :událost
```

Do proměnné :*událost* přidáme *větu* složenou z příkazu *přehrajSoubor* a příslušného názvu proměnné :*soubor*, definované parametrem :*číslo*. Tuto proměnnou :*událost* poté přiřadíme tlačítku *tl4* jako událost *přiZapnutí*.

```
dosaď "playlist (věta :playlist (slovo "soubor :číslo))
text4'hodnota! :playlist
```

Nakonec aktualizujeme proměnnou :*playlist* tím, že do ní přidáme jméno patřičné proměnné :*soubor* a tento nový *plylist* necháme zobrazit textem *text4.*

3. Příkaz *Zruš* – tento příkaz pouze vyprázdní proměnné :*událost* a :*playlist* příkazem `dosaď "událost [] dosaď "playlist []`, nastaví tlačítku *tl7* událost *přiZapnutí* na prázdný řetězec příkazem `tl7'událost! "přiZapnutí []` a nakonec vyprázdní hodnotu objektu *text4* tím, že jeho hodnotu nastaví na hodnotu proměnné :*playlist,* kterou jsme předtím vyprázdnily.

Popis objektů:

1. Objekty *Text (text1, text2, text3, text4)* - tyto objekty mají stejné nastavení jako objekty stejného typu v předcházejících kapitolách
2. Objekty *Tlačítko (tl1, tl2, tl3)* – tyto objekty slouží k otevření dialogového okna pro výběr požadovaného souboru. Jako událost *přiZapnutí* jim nastavíme příkaz *vyber 1* pro *tl1*, *vyber 2* pro *tl2* a *vyber 3* pro *tl3.*
3. Objekty *Tlačítko (tl4, tl5, tl6)* – tyto objekty slouží k přidání dané písně do playlistu. Jako událost *přiZapnutí* jim nastavíme příkaz *play 1* pro *tl1*, *play 2* pro *tl2* a *play 3* pro *tl3.*

Pozn.: Pokud uložíme náš projekt jako EXE soubor nebo jako WEBovskou aplikaci, nebude toto tlačítko reagovat, jelikož prostředí Imagine neumí po exportu do takovéhoto formátu číst z disku.

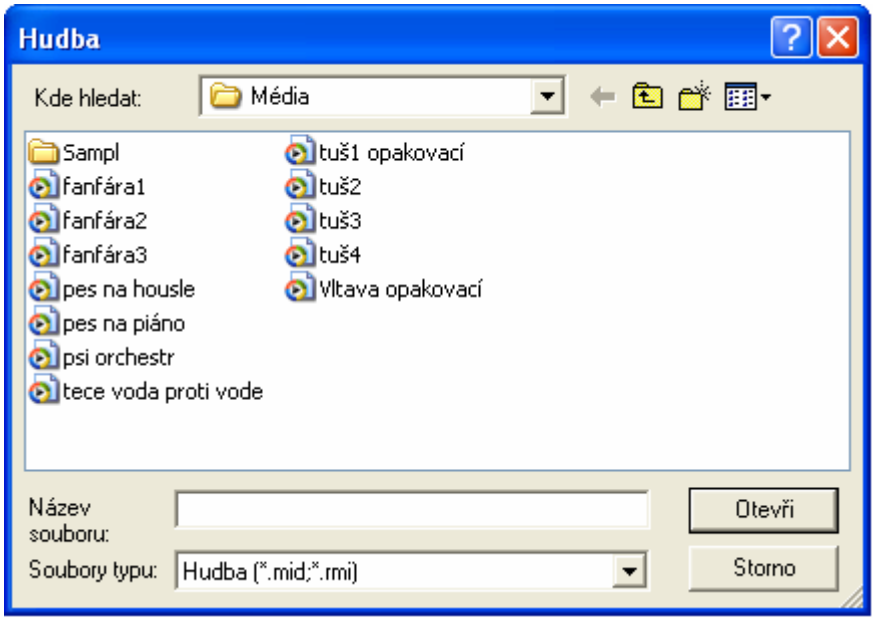
4. Objekt *Tlačítko (tl7)* – tento objekt slouží k přehrání playlistu. Jeho událost *přiZapnutí* nastavujeme v příkazu *play.*
5. Objekt *Tlačítko (tl8)* – tento objekt slouží ke zrušení playlistu. Jeho událost *přiZapnutí* nastavíme na příkaz *Zruš.*

4.2.3 Objekt *Hudba*

Výše zmíněné nástroje pro práci s multimédii patří do kategorie příkazů. Nyní si povíme něco o multimediálních objektech prostředí Imagine pracujících s audio soubory.

Pokud v záložce *Nový objekt media* vybereme objekt *Hudba*, viz Obr. 18, a poté klikneme někam do plochy, otevře se dialogové okno *Hudba*, které nám umožňuje vybrat hudební soubor formátu MID či RMI, viz Obr. 19. Po potvrzení souboru tlačítkem *Otevři* se na ploše objeví nové tlačítko se specielním tvarem symbolizujícím objekt *Hudba*. Vybraný hudební soubor se uloží jako objektová proměnná *:soubor* tohoto tlačítka. Po jeho stisknutí se soubor přehraje.

Obr. 18 Objekt *Hudba*



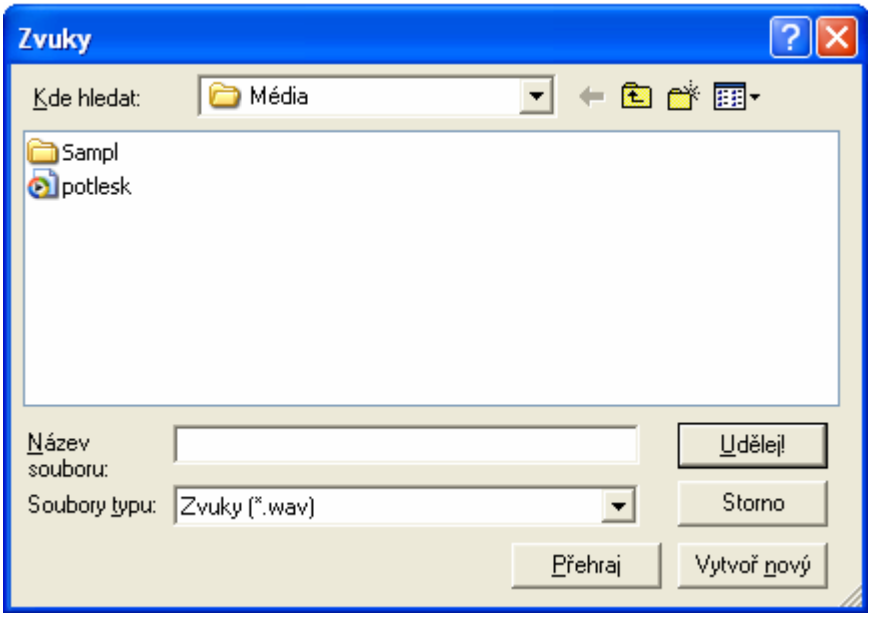
Obr. 19 Dialogové okno objektu *Hudba*

4.2.4 Objekt *Zvuk*

Posledním objektem v záložce *Nový objekt media* pracujícím výlučně s audiem, je objekt *Zvuk*, viz Obr. 20. Pokud zvolíme tento objekt, kurzor myši se změní na tvar reproduktoru a po kliknutí do plochy se stejně jako v případě objektu *Hudba* otevře dialogové okno, viz Obr.21, umožňující vybrat požadovaný audio soubor formátu WAV či MP3. Výhodou tohoto dialogového okna je možnost přehrání souboru pomocí tlačítka *Přehraj*, ještě před jeho vybráním tlačítkem *Udělej*, po jehož stisknutí se na ploše objeví tlačítko ve tvaru reproduktoru, které má ve vlastní proměnné :*soubor* uloženu adresu vybraného souboru a tento soubor po stisknutí přehraje.

Pozn.: Všimněte si, že přestože prostředí Imagine umí přehrávat celou řadu audio formátů, dialogové okno, které se při vytvoření objektu *Zvuk* otevře, umožňuje vybrat pouze soubory typu WAV a MP3. Pokud bychom chtěli pomocí tohoto objektu přehrát jiný formát, museli bychom ho nejdřív vytvořit vybráním souboru odpovídajícího formátu a poté ručně přepsat adresu souboru v proměnné :*soubor*, nebo použít příkaz `tl1'atJeVlastní "soubor :cesta`, kde *tl1* je objekt *Zvuk* a :*cesta* je pomocná proměnná, do které jsme předem uložili adresu požadovaného souboru.

Obr. 20 Objekt *Zvuk*



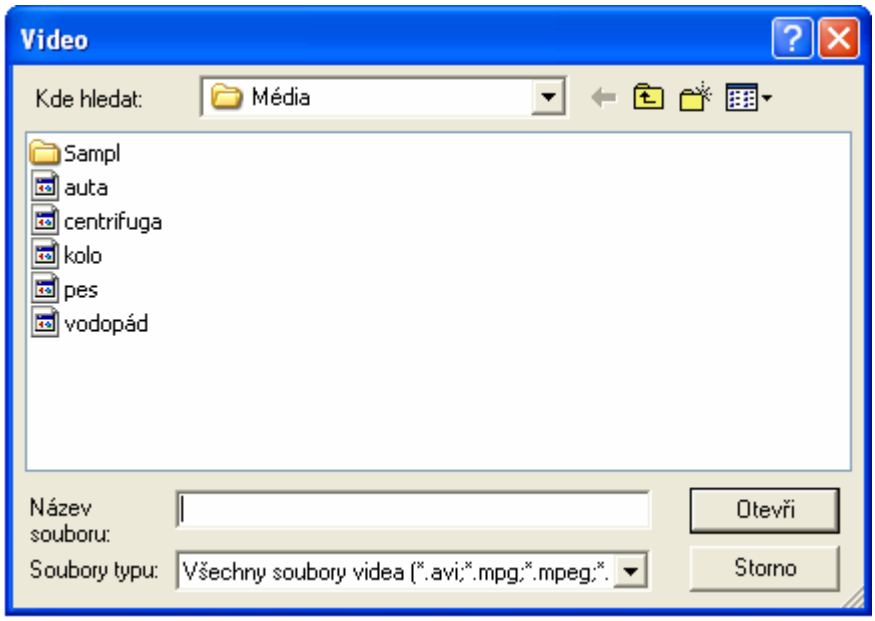
Obr. 21 Dialogové okno objektu *Zvuk*

4.2.5 Objekt *Videoklip*

Posledním z objektů, které můžeme vybrat ze záložky *Nový objekt média* je objekt *Videoklip*, viz Obr. 22. Jedná se o specielní objekt třídy objektů *Přehrávač médií*, který umožňuje přehrávat audio a video soubory. Poté co jej zvolíme a klikneme do plochy prostředí Imagine, otevře se dialogové okno pro výběr souboru který chceme přehrát, viz Obr. 23. Přestože povoleným formátem souborů pro výběr v dialogovém okně je pouze AVI, MPG, MPEG a MOV, objekt *Videoklip* přehrává i všechny podporované audio soubory typu WAV, MP3, WMA atd.

Pokud chceme pomocí objektu *Videoklip* přehrát jiný typ souboru než je možné vybrat v dialogovém okně, musíme přehrávači nastavit požadovaný soubor buď použitím příkazu *soubor! soubor*, nebo změnit adresu požadovaného souboru přímo v rodném rodného listu objektu *Videoklip* pomocí tlačítka *Procházet*.

Obr. 22 Objekt *Videoklip*



Obr. 23 Dialogové okno objektu *Videoklip*

Pozn.: Může se stát, že pokud objektu *Videoklip* nastavíme k přehrání audio soubor, zmizí přehrávací okno pro video a zůstane pouze panel ovládání, viz Obr. 24. Tento panel nereaguje na žádnou událost, přestože je hodnota operace *reaguje* ano a hodnota operace *zamknuto* ne. Objekt stále reaguje na příkazy jazyka Logo, ale pokud chceme měnit nějaká nastavení v jeho rodném listu, musíme si tento objekt najít v okně paměti. Pokud chceme opět zobrazit přehrávací okno, musíme buď objektu *Videoklip* přiřadit příkazem *soubor!* nějaký video soubor, nebo příkazem *velikost!* změnit jeho velikost, čímž se opět přehrávací okno objeví.

Obr. 24 Panel ovládání objektu *Videoklip*

Příklad 1: Loop – přehrávání nekonečené smyčky

V tomto příkladu si ukážeme jak pomocí objektu *Videoklip,* který je součástí třídy *Přehrávač médií,* přehrávat nekonečnou smyčku.

Využijeme k tomu příkazy pro práci s třídou *Přehrávač médií.* Prvním z nich je příkaz *pozicePřehrávání,* který lze použít jak pro zjištění aktuální pozice přehrávání, tak pro přesun přehrávání na danou pozici. Druhým příkazem je *dobaPřehrávání,* který vrací délku skladby v sekundách. K zastavení přehrávání využijeme příkazu *přeruš.*

Postup:

K nastavení samotné smyčky požijeme dva objekty typu *Posuvník (p1, p2).* Jeden pro začátek a druhý pro konec smyčky. Ke zobrazení času přehrávání, začátku a konce smyčky použijeme tři objekty typu *Text (text1,text2,text3).* Ke spuštění samotného přehrávání smyčky použijeme objekt *Přepínač (tl1).* Pro načtení souboru do objektu *Přehrávač médií* použijeme objektu *Tlačítko (tl2).*

Popis příkazů:

Pro přehrání smyčky si vytvoříme nový příkaz, který nazveme *loop.* Ten pomocí rekurzivního volání spustí nekonečný proces, ve kterém nejprve zapne přehrávání souboru a poté jednoduchou podmínkou ověřuje pozici přehrávání vůči posuvníku *p2.* Pokud se tato pozice dostane do stejné pozice, jako je nastaven posuvník *p2,* přesune se přehrávání pomocí příkazu *pozicePřehrávání* na pozici, kterou udává *p1* a v přehrávání se pokračuje. Pro zobrazení uplynulého času se neustále přiřazuje objektu *text3* hodnota *pozicePřehrávání.* Tento příkaz bude vypadat takto:

```
příkaz loop
video1'přehraj_mm
```

```
když video1'pozicePřehrávání >= p2'hodnota
[video1'pozicePřehrávání! p1'hodnota]
text3'hodnota! video1'pozicePřehrávání
loop
konec
```

> **Pozn.:** I když se v nápovědě aplikace Imagine píše, že pro spuštění přehrávání multimediálního souboru je příkaz *přehraj*, pravdou je, že správným příkazem je příkaz *přehraj_mm.* Pokud použijeme příkaz *přehraj*, otevře se pomocník *Vytvořit melodii*. Když s jeho pomocí poté složíme nějakou melodii a potvrdíme tlačítkem *Udělej*, melodie se přehraje, objektu *Videoklip* však nadále zůstane stejný multimediální soubor jako předtím.

Dalším příkazem je příkaz *vyber*, který použijeme k načtení požadovaného audio souboru z disku. Tento příkaz je obdobou stejnojmenného příkazu v *Příkladě 2 Kapitoly 4.2.2* a vypadá takto:

```
příkaz vyber

dosaď "pom pomocníkNačtiSoubor [] [Vyber audio soubor] [|Audio
Soubory@*.wav;*.mp3;*.wma|]

kdyz :pom <> [] [video1'soubor! :pom]

pro [p1 p2] [max! video1'dobaPřehrávání]

konec
```

Nejprve tedy pomocí operace *pomocníkNačtiSoubor* načteme soubor požadovaného formátu z disku do pomocné proměnné :*pom*, kterou poté otestujeme, zda není prázdná. Pokud není, dosadíme ji pomocí příkazu *soubor!* jako objektovou proměnnou objektu *video1*. Nakonec nastavíme maximální hodnoty posuvníků *p1* a *p2* na hodnotu, kterou vrátí operace *dobaPřehrávání* To proto aby jejich maximální hodnota byla rovna délce přehrávaného audio souboru.

Popis objektů:

1. Nejprve si vytvoříme objekt *Videoklip (video1)* a pomocí příkazu *soubor!*, nebo jiného ekvivalentního způsobu, mu nastavíme k přehrání nějaký audio soubor.
2. Objekty *Text (text1,text2,text3)* – tyto objekty mají stejné nastavení jako objekty stejného typu v předcházejících kapitolách. Pokud bychom chtěli zobrazovat jak vteřiny tak minuty, museli bychom zdvojnásobit počet objektů *Text.* Pomocí příkazu `cPodíl video1`dobaPřehrávání 60`, bychom zjistili počet minut skladby a pomocí příkazu *`zbytek video1'dobaPřehrávání 60`* počet vteřin. Tyto hodnoty bychom poté přiřazovali jednotlivým objektům *Text*.
3. Objekty *Posuvník (p1, p2)* – posuvník *p1* nám bude sloužit k nastavení začátku smyčky a posuvník *p2* k nastavení jejího konce. Do události *přiZměně* vložíme posuvíku *p1* příkaz `text1'hodnota! p1` pro nastavení hodnoty objektu *text1*.Posuvník *p2* bude mít ve své události *přiZměně* obdobný příkaz `text2'hodnota! p2`.
4. Objekt *Přepínač (tl1)* – tento objekt bude mít v události přiZapnutí příkaz *loop* pro zapnutí přehrávání nekonečné smyčky a v události *přiVypnutí* příkaz *video1'přeruš* pro jeho vypnutí.

5 Nástroje prostředí Imagine pro práci s videem

5.1 Příkazy *startVideo* a *stopVideo*

Prostředí Imagine poskytuje pro práci s videem dvojici příkazů *startVideo* a *stopVideo*. Příkazem *startVideo* se zkonstruuje MCI příkaz, který se vykoná. Podle parametrů, které zadáme jako vstup tomuto příkazu, se vytvoří nové okno a spustí přehrávání video souboru. Příkaz *stopVideo* přeruší přehrávání a toto okno zavře.

Jelikož máme k dispozici mnohem modernější přístup pro práci s videem ve formě třídy objektů *Přehrávač médií*, je použití těchto příkazů zbytečné. Seznam parametrů a bližší informace nalezneme v nápovědě aplikace.

Pozn.: Při použití příkazu:

```
? ignoruj startVideo "|Media\auto závod.mpg| "mojeAuto [child [] [100 100]]
```

který je uveden v nápovědě aplikace, prostředí Imagine reagovalo tím, že spadlo. (Testováno na systému Windows XP Pro SP2)

5.2 Objekt *Videoklip*

Jak už jsme se dozvěděli v *Kapitole 4.2.5*, objekt *Videoklip* je specielním objektem třídy objektů *Přehrávač médií*, která nám nabízí celou řadu příkazů a nastavení, viz nápověda aplikace.

Pozn.: Pokud máme v počítači nainstalovaný nějaký software pro zobrazování titulků k filmům, jako je např. *DirectVobSub*, a máme titulky ve stejném adresáři jako film, budou se nám pomocí tohoto softwaru zobrazovat při přehrávání videa objektem *Videoklip*.

Seznam příkladů:

Příklad 1: Slow - Motion – zpomalené přehrávání

V tomto příkladu si ukážeme, jak pomocí kombinace příkazů *krokVpřed* ze třídy *Přehrávač Médií* a příkazu *čekej,* zpomalit přehrávání souboru a tím docílit efektu slow-motion.

Postup:

Nejprve si samozřejmě vytvoříme objekt *Videoklip (video1)* a otevřeme požadovaný video soubor. Dále budeme potřebovat objekty *Posuvník (p1) a Text (text1)* pro nastavení a zobrazení míry zpomalení, a *Tlačítko (tl1)* pro spuštění samotného příkazu *SlowMo.* Pro načítání souboru z disku použijeme příkaz *vyber* jako v *Příkladě 2 Kapitoly 4.2.2,* který pouze modifikujeme pro výběr video formátu.

Popis příkazu:

Příkaz *SlowMo* v každém cyklu rekurzivního volání posune *video1* pomocí příkazu *krokVpřed* o jeden snímek dopředu a počká pomocí příkazu *čekej* tak dlouho, jakou hodnotu udává posuvník *p1,* respektive *text1.* Tento příkaz bude tedy vypadat takto:

```
? příkaz SlowMo
? video1'krokVpřed
? čekej p1'hodnota
? SlowMo
? konec
```

Popis objektů:

1. Objekt *Text (text1)* - tento objekt má stejné nastavení jako objekty stejného typu v předcházejících kapitolách.

2. Objekt *Posuvník (p1)* – tomuto objektu nastavíme v rodném listě požadované minimální a maximální hodnoty zpomalení. Do události *přiZměně* vložíme příkaz pro změnu hodnoty objektu *text1* `text1`hodnota! p1`.
3. Objekt *Tlačítko (tl1)* – do události *přiZapnutí* vložíme příkaz *SlowMo*, který zapne zpomalené přehrávání.

Pozn.: Příkaz *krokVpřed* bohužel nepracuje se zvukovou stopou záznamu. Výsledný slow-motion efekt je tedy němý a nelze použít na audio soubory.

Příklad 2: Loop – přehrávání nekonečené smyčky

Pro hraní nekonečné video smyčky můžeme použít stejný způsob jako v *Příkladu 1 Kapitoly 4.2.5*. Viz také *Kapitola 6.2.1*. Dejte ovšem pozor na fakt, že příkaz *dobaPřehrávání* nevrací u video souboru počet vteřin, ale počet jeho snímků.

Pozn.: Pro případ, kdy si chceme brát inspiraci z nějakého již předem vytvořeného Imagine projektu, by se nám hodila možnost mít otevřeno více projektů zároveň. Toho docílíme malým trikem. Otevřeme si první projekt a v něm otevřeme rodný list libovolného objektu. Když nyní otevřeme další projekt, spustí se nám druhá instance aplikace Imagine.

6 Využití multimédií v projektu

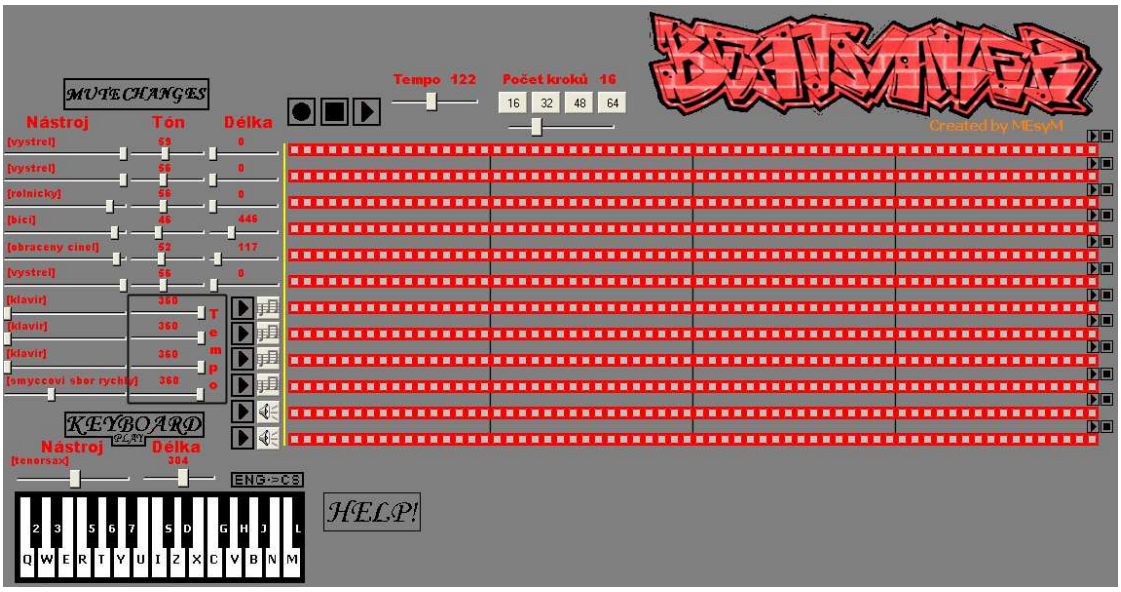
V této části příručky se podíváme na větší projekty vytvořené v prostředí Imagine, které pracují s multimédii.

6.1 Audioprojekty

Nejprve si představíme projekty pracující výhradně s audiem.

6.1.1 Beatmaker

Tento projekt, viz Obr. 25, umožňuje „skládání“ hudby pomocí takzvaných patternů, neboli šablon, jako nabízí například bicí automat Redrum, ze kterého jsem si bral inspiraci.



Obr. 25. Projekt Beatmaker

Popis projektů:

K dispozici je 12 zvukových stop po 64 krocích, které představují čtyři takty. Každý krok je tvořen objektem typu *Želva*. Tento objekt má dva záběry. Pokud na něj klikneme, změní se jeho záběr pomocí příkazu *změna* a zároveň pod sebou nakreslí

pomocí příkazu *body* buď černý nebo modrý puntík, podle toho zda má být zrovna aktivní či neaktivní.

Každá stopa má svou „hrací želvu“, která se pohybuje krok po kroku a zjišťuje pomocí příkazu *barvaBodu*, zda je krok aktivní, či naopak. Pokud je krok neaktivní a hrací želva na něj vstoupí, nic se nestane. Pokud je však aktivní, přehraje se zvuk odpovídající dané stopě.

Jelikož má většinou každá stopa nastavenu jinou délku tónu, která se poté přičítá k časové prodlevě určující tempo, není jednoduché pohyb hracích želv synchronizovat. Proto je zde navíc „nosná želva“ (*ž69*), která se stará o pohyb všech ostatních hracích želv. Příkaz *hraje* pohybuje nosnou želvou o určitý počet kroků v určitém tempu a poté ji vrátí domů. Synchronizace hracích želv je tedy řešena tak, že je jejich X-ová souřadnice nastavena pomocí příkazu *xSour*! na X-ovou souřadnici nosné želvy. Všechny želvy se tak pohybují stejně, nezávisle na délce tónu, který přehrávají. Tempo je řešeno příkazem *čekej*, který zajistí časovou prodlevu mezi jednotlivými kroky nosné želvy a počet kroků je řešen nastavením proměnné pro příkaz *opakuj* v příkazu *hraje*. Nastavení hodnoty tempa je řešeno pomocí objektu *Posuvník*. Stejně tak i nastavení hodnoty počtu kroků, kde je navíc možnost nastavení pomocí čtyř objektů typu *Tlačítko* na předem definovaný počet (16, 32, 48, 64).

Prvních šest stop pracuje s příkazem *hrajNotu* a pomocí objektů typu *Posuvník* je možné měnit výšku, délku a hudební nástroj, kterým se tón pro danou stopu bude přehrávat, tak jak je popsáno v *Příkladu 3 Kapitoly 4.1.3.*

Další čtyři stopy pracují s objektem *Melodie*, respektive s příkazem *přehraj*. Díky pomocníku *Vytvořit melodii* lze na každé této stopě složit jinou melodii, která se poté přehraje, kdykoliv odpovídající hrací želva vstoupí na aktivní krok dané stopy. Pomocí objektů typu *Posuvník* můžeme v reálném čase měnit tempo a hudební nástroj, kterým se daná melodie přehrává, bez použití pomocníka pro vytváření melodie, tak jak je popsáno v *Příkladu 4 Kapitoly 4.1.2* a *Kapitole 4.1.5*.

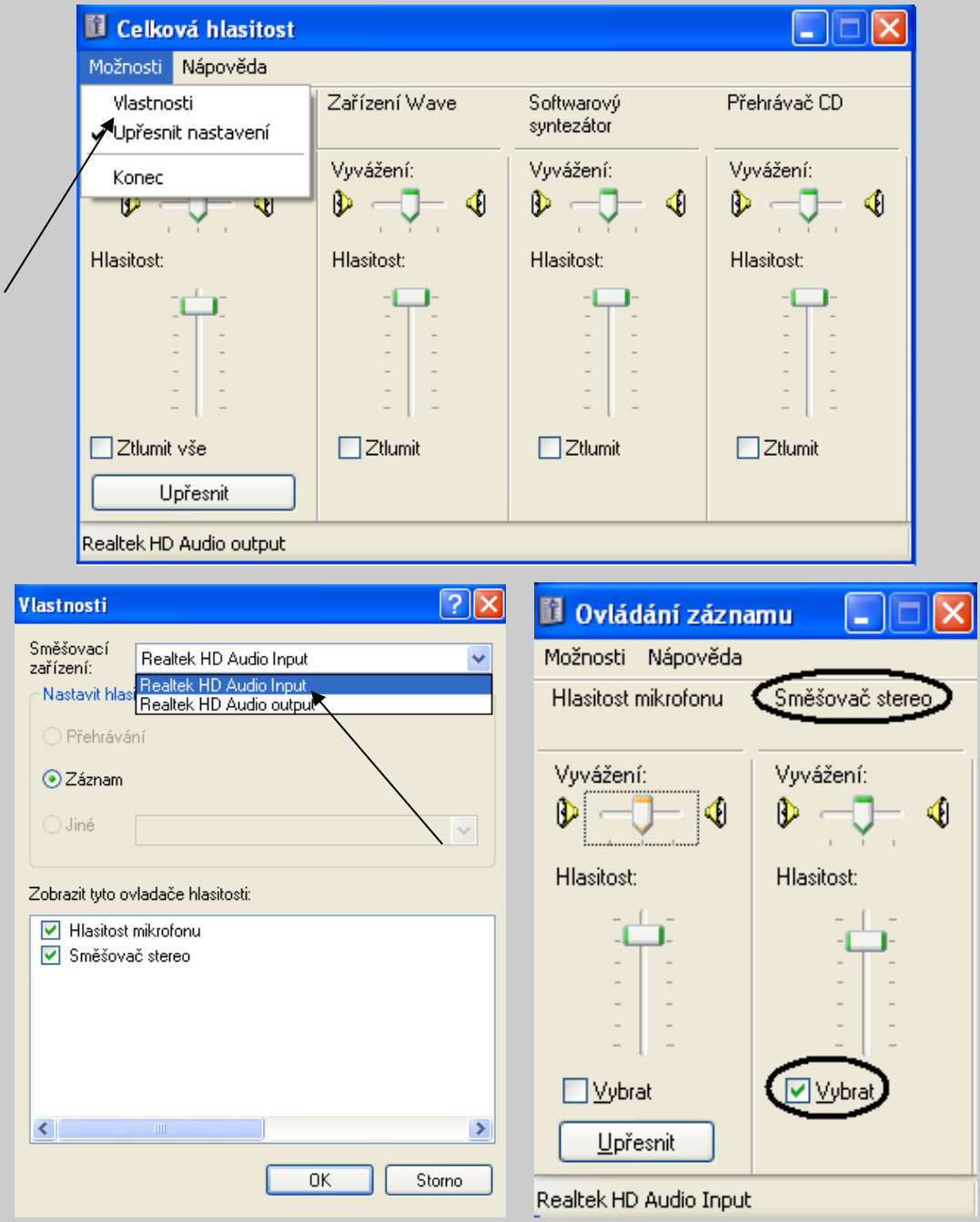
Poslední dvě stopy slouží k přehrání tzv. samplů, tj. zvukových souborů pomocí objektu *Zvuk*, respektive příkazu *přehrajSoubor*.

Dále je k dispozici objekt *Želva (ž70)* nazvaný Keyboard, který má simulovat pomocí příkazu *hrajNotu* hru na piáno. Pomocí *Posuvníku* můžeme měnit hudební nástroj, kterým budeme hrát. Jelikož prostředí Imagine nedokáže rozpoznat délku stisknutí klávesy, je toto hraní na piano omezeno na vybranou délku noty, kterou nastavíme pomocí dalšího *Posuvníku*. Další objekt typu *Želva (ž803)* slouží k přepínání anglické a české klávesnice, viz příkaz *piano*.

K zapnutí přehrávání všech 12 stop slouží objekt typu *Želva*. K vypnutí přehrávání je využit objekt typu *Tlačítko*. Další dva objekty typu *Želva* jeden jako PlaySolo a druhý jako StopSolo jsou zvlášť u každé stopy pro možnost sólového hraní jedné či více stop.

Další objekt typu *Tlačítko* slouží k zapnutí Windows aplikace *Záznam zvuku*, který můžeme využít k nahrání složené skladby do WAV formátu.

Pozn.: Pokud chceme pomocí programu *Záznam zvuku*, nebo kteréhokoliv jiného audio softwaru, nahrávat zvuky, které právě přehrává naše zvuková karta, musíme mít v nastavení zvukové karty pro audio input zapnut Směšovač stereo, viz Obr. 26.



Obr. 26 Nastavení vstupu zvukové karty

Jelikož prostředí Imagine po uložení projektu jako WEB nebo EXE souboru neumožňuje spouštění externích programů, nebude fungovat ani toto tlačítko.

Objekt *Tlačítko (tl5)* s popisem *Mute Changes*, slouží k vypnutí či zapnutí hraní tónů při jejich nastavování pomocí posuvníků.

Popis nejdůležitějších příkazů:

body – příkaz se stará o nakreslení puntíku příslušné barvy pod stisknutou želvou reprezentující jeden krok. Všechny tyto želvy mají příkaz *body* nastaven jako událost *přiKlepnutí*.

hraje – stará se o pohyb nosné želvy (*ž69*) v tempu daném posuvníkem *p3* o počet kroků daný posuvníkem *p5*. Poté ji vrátí na domovskou pozici.

hrajeme – pro nosnou želvu spustí nekonečný proces *hraje*. Pro všechny hrací želvy spustí nekonečný proces, který přísluší dané stopě, viz níže. Tento příkaz je nastaven jako událost *přiKlepnutí* objektu *Želva*, reprezentující tlačítko Play.

hrajou :nástroj :nota :délka – příkaz pro hrací želvy pracující s prvními šesti stopami, tedy s příkazem *hrajNotu*. Příkaz nejprve přemístí danou želvu na X-ovou souřadnici nosné želvy a poté otestuje, zda je barva bodu rovna *černé*. Pokud ano nastaví příkazem *hudebníNástroj!* hudební nástroj na hodnotu danou vstupním parametrem :*nástroj*, což je název příslušného posuvníku příslušné stopy, a přehraje příkazem *hrajNotu* notu danou vstupním parametrem :*nota* o délce dané vstupním parametrem :*délka*, což jsou opět jména příslušných posuvníků příslušné stopy. Nakonec pomocí příkazu *čekej* počká po dobu posuvníku *p3*, který reprezentuje tempo skladby. Tím se zajistíme, že se daný tón přehraje pouze jednou.

melo, melo1, melo2, melo3 - příkazy pro hrací želvy pracující s melodiemy, tedy s příkazem *přehraj*. Nejprve přemístí danou hrací želvu na X-ovou souřadnici nosné želvy a poté otestuje barvu bodu, na které se želva nachází. Pokud je barva *černá*, testuje se pomocí příkazu *skončil? [@tl22přiZapnutí]*, zda již neběží proces, spuštěný událostí *přiZapnutí* tlačítka *tl22*. Pokud běží, přehraje se melodie pomocí tlačítka *tl27*, pokud neběží, spustí se přehrávání melodie pomocí tlačítka *tl22*. Tento přístup nám umožňuje hrát dvojitý kánon.

nástrojM, nástrojM2 – první příkaz se stará o změnu nástroje melodie uložené v objektu *Melodie* pomocí příslušného posuvníku. Druhý příkaz se stará o změnu hodnoty příslušného posuvníku, pokud jsme nástroj změnili pomocí pomocníka pro tvorbu melodie, viz *Kapitola 4.1.5*.

piano – tento příkaz se stará o simulaci hry na piáno. V rekurzivním volání nejprve pomocí příkazu `aktivníPříkazovýŘádek! "ne` vypne možnost pracovat s příkazovým řádkem. To proto abychom mohli využít klávesnici k simulovanému hraní. Poté nastaví pomocí příkazu *hudebníNástroj!* hudební nástroj, kterým se bude hrát a nakonec podle toho, v jakém záběru je zrovna želva *ž803,* testuje pomocí operace *klávesa?* jakou klávesu jsme stiskly. Poté pomocí procedury *vPřípadě* změní záběr želvy *ž70,* podle toho jakou jsme stiskly klávesu, a příkazem *hrajNotu* zahraje danou notu s délkou, jejíž hodnotu jsme nastavili posuvníkem *p13.* Nakonec změní záběr želvy *ž70* zpět na její původní tvar.

play – příkaz, který má objekt *Želva,* reprezentující tlačítko Play, nastaven jako událost *přiKlepnutí.* Je shodný s příkazem *stop,* viz níže, s tím rozdílem, že neobsahuje příkaz *zastavVše*.

playOn, playOff – tyto příkazy zapínají, respektive vypínají, přehrávání tónu při jeho nastavování a jsou nastaveny jako událost *přiZapnutí*, respektive *přiVypnutí*, tlačítka *tl5* s popisem Mute Changes.

sampl1, sampl2 - příkazy pro hrací želvy pracující se samply. Nejprve přemístí danou hrací želvu na X-ovou souřadnici nosné želvy a poté otestuje barvu bodu, na které se želva nachází. Pokud je barva *černá,* přehrají pomocí příslušného tlačítka daný sampl.

skryjTečky - pomocný příkaz, který skryje všechny hrací želvy. Jeho obdobou je příkaz *ukažTečky.*

solo1, solo2, solo3, solo4, solo5, solo6 – tyto příkazy se starají o sólové přehrání stopy pracující s příkazem *hrajNotu* a jsou nastaveny jako událost *přiKlepnutí* odpovídajících tlačítek PlaySolo. Nejprve změní tvar nosné želvy a pomocí příkazu

ukaž zobrazí danou hrací želvu. Poté se otestuje, zda již neběží hlavní proces *hraje.* Pokud neběží, spustí se zároveň s nekonečným procesem přehrávání pro danou hrací želvu. Pokud tento proces běží, otestuje se, zda již neběží proces přehrávání pro danou stopu. Pokud ne, spustí se tento proces příkazem *hrajou :nástroj :nota :délka.* Toto testování je zde, protože pokud bychom spustili proces *hraje,* nebo *hrajou :nástroj :nota :délka* dvakrát či vícekrát, zvyšovala by se rychlost přehrávání. Důvod proč jsme místo šesti příkazů nepoužili jeden příkaz s parametrem je ten, že pokud bychom vytvořili příkaz *solo :kdo :nástroj :tón :délka* pro hraní sólové stopy, nemohly bychom tento proces přerušit příkazem *zastav [solo :kdo :nástroj :tón :délka],* aniž bychom přerušili všechny ostatní procesy se stejnými parametry.

soloMelo, soloMelo1, soloMelo2, soloMelo3 – příkazy, které fungují na stejném principu jako předchozí příkaz. Jediným rozdílem je, že se starají o sólové hraní stop pracujících s melodiemi a tedy pracují s příkazy *melo, melo1, melo2* a *melo3.*

soloSampl1, soloSampl2 – obdobné příkazy jako předchozí dva. Starají se o sólové hraní stop pracujících se samply tedy příkazy *sampl1* a *sampl2*.

soloStop :kdo :pl – příkaz, který je použit jako událost *přiKlepnutí* u tlačítek StopSolo, které se starají o zastavení hraní dané stopy. Nejprve změní záběry nosné želvy a tlačítka PlaySolo definovaného vstupním parametrem *:pl* a poté otestuje, zda skončil příkaz *Piano*. Pokud skončil, znamená to, že jsme nezapnuli přehrávání všech hracích želv pomocí tlačítka Play, ale hrajeme pouze sólovou stopu. Proto se zobrazí pouze hrací želva daná vstupním parametrem :*kdo*, pokud však příkaz *Piano* běží, znamená to, že se přehrávají všechny stopy a proto se tedy zobrazí všechny hrací želvy.

stop – tento příkaz nejprve skryje všechny hrací želvy, poté přemístí nosnou želvu na pozici [-109 112], což je pozice, ve které se nosná želva nachází při nečinosti programu, a nastaví její tvar na druhý záběr. Poté zavolá příkaz *Dm,* který přemístí všechny hrací želvy do domovských pozic. Nakonec nastaví všem tlačítkům

PlaySolo tvar na první záběr a příkazem *zastavVše* zastaví všechny běžící procesy. Příkaz *stop* je nastaven jako událost *přiZapnutí* tlačítka Stop.

tempoM, tempoM2 – jsou obdobou příkazů *nástrojM, nástrojM2*. Starají se o změnu tempa melodie.

ukažTežky – pomocný příkaz, který zobrazí všechny hrací želvy. Jeho obdobou je příkaz *skryjTečky*.

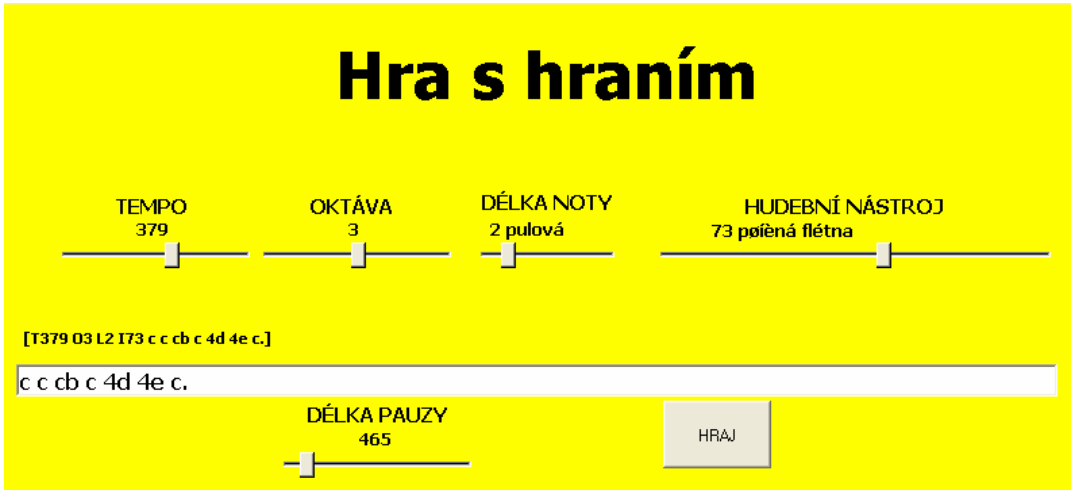
vyber – příkaz pro načítání audio souboru z disku. Viz *Příklad 2 Kapitoly 4.2.2* nebo *Kapitola 6.2.1*.

změna – příkaz, který mají jako událost *přiKlepnutí* nastaveny všechny objekty typu *Želva*, které reprezentují jeden krok skladby. Stará se o změnu jejich záběru. Pomocí podmínky zjišťuje, zda je želva právě v záběru 1, či nikoliv. Pokud je, změní se na záběr 2, pokud není, změní se na záběr 1.

6.1.2 Hra s hraním

Tento projekt umožňuje „skládání“ melodie v reálném čase, viz Obr. 27.

Hraní melodie je realizováno příkazem *přehraj*. Jeho vstupní seznam not včetně všech možných symbolů, viz nápověda aplikace, píšeme do textového řádku a tím skládáme melodii, jejíž přehrávání se stále opakuje. Jednotlivé tóny či symboly musíme oddělit mezerou.



Obr. 27 Projekt Hra s hraním

Popis objektů:

V projektu je celkem pět objektů typu *Posuvník.* Posuvníky *p1* až *p4* , které slouží k nastavení atributů melodie, jako je tempo, oktáva, základní délka noty a hudební nástroj. Posuvník *p5* slouží k nastavení délky pauzy mezi opakováním melodie. Všechny tyto objekty mají nastavenu událost *přiZměně* tak, aby se měnila hodnota příslušného objektu *Text.*

Těchto objektů typu *Text* je v projektu celkem 12. Objekty *text1* až t*ext10* slouží k popisu jednotlivých posuvníků, tak jak je popsáno v mnoha příkladech v *Kapitole 3.* Objekt *text11,* který je nazvaný *mel*, slouží jako textový řádek, ve kterém se tvoří melodie a objekt *text12* zobrazuje celý vstupní seznam, tj. parametry i tóny pro příkaz *přehraj,* tj. hodnotu proměnné :*melodie.*

Posledním objektem je objekt typu *Tlačítko (tl1),* které slouží k zapnutí samotného cyklu přehrávání.

Popis proměnných:

:melodie – uchovává melodii, která se přehrává

:melh – pomocí příkazu *hraZřádku,* viz níže, uchovává seznam not napsaných v textovém řádku *mel*

:délka – uchovává hodnotu, kterou vrací příkaz *počet mel`hodnota*, tj. délku seznamu napsaného v textovém řádku *mel*

:první, :druhý, :třetí, :čtvrtý – uchovávají daný prvek seznamu, který vrací příkaz *mel`hodnota*

:krok – uchovává pozici, ze které začneme v příštím volání příkazu *hraZřádku* dosazovat prvky ze seznamu, který vrací příkaz *mel`hodnota*, do proměnných *první, druhý, třetí, čtvrtý*

:dNoty – obsahuje seznam možných délek noty, tj. celá, půlová, čtvrtinová atd., který se využívá pro zobrazení zvolené hodnoty pomocí příslušného objektu *text.*

Popis příkazů:

2na :kolik – jelikož délka základní noty v atributu melodie může být pouze 1, 2, 4, 8, 16, 32 nebo 64 a objektu *Posuvník* nelze nastavit takovéto číslování, pomáhá nám tento příkaz převést čísla od 0 do 6 na mocninu dvou. Parametr :*kolik* tedy udává, kterou mocninu čísla 2 chceme.

hraj – je základní příkaz, který se volá po stisknutí tlačítka *tl1*. Nejpve zinicializuje všechny proměnné a poté zavolá příkaz *hraZřádku*, který načte melodii z textového řádku *mel*, viz níže. Poté příkaz *hraj* dosadí hodnoty atributů tempa, oktávy, délky základní noty a hudebního nástroje do proměnné :*melodie*, obdobně jako v *Příkladu 4 Kapitoly 4.1.2*, a hodnotu této proměnné zobrazí objektem *text12*. Nakonec tuto melodii pomocí příkazu *přehraj* přehraje a počká pomocí příkazu *čekej* tak dlouho, jakou hodnotu udává posuvník *p5*.

hraZřádku :délka – tento příkaz se stará o správné načtení melodie z textového řádku. Nejprve otestuje, zda je vstupní proměnná :*délka* větší než nula. Pokud je, dosadí do proměnných :*první, :druhý, :třetí, :čtvrtý* dané prvky seznamu, který vrací příkaz *mel`hodnota*. Čtyři prvky jsou použity proto, že to je nejdelší možné slovo, které příkaz *přehraj* bere jako svůj vstup (např. 64c#). Jaký prvek bude přiřazen jaké proměnné, určuje proměnná :*krok*. Poté se pomocí procedury *vPřípadě*,viz nápověda aplikace, testuje, zda slovo složené z proměnných :*první, :druhý, :třetí, :čtvrtý* může být použito jako vstup v příkazu *přehraj*. Pokud je slovo použitelné, přidáme ho do proměnné :*melh* a podle toho jak bylo dlouhé, zvětšíme proměnnou :*krok* a rekurzivně zavoláme příkaz *hraZřádku*, tentokrát ale se vstupní proměnnou :*délka* zmenšenou o délku použitého slova. K proměnné :*krok* přičítáme číslo o jedno větší než délka slova, protože počítáme s mezerou mezi jednotlivými prvky melodie, a to samé děláme u proměnné :*délka*. Pokud je proměnná :*délka* menší než

0, znamená to, že jsme již načetli celý řádek, proměnná :*melh* se zapíše na konec proměnné :*melodie* a příkaz se ukončí.

6.1.3 Test anglických slovíček

Projekt který prověří vaši schopnost odposlechnout anglická slovíčka a poté je správně přeložit, viz Obr. 28.

Obr. 28 Projekt Test anglických slovíček

Po startu programu se pomocí příkazu *dialog*, viz nápověda aplikace, objeví uvítací dialogové okno. Po stisknutí tlačítka OK, se spustí samotný test. Ten funguje tak, že nejprve program přehraje pomocí příkazu *přehrajZvuk* nějaké anglické slovíčko a vaším úkolem je toto slovo přeložit. Aby nebyl překlad tak těžký, máte na výběr ze tří odpovědí, realizovaných objektem *Tlačítko*. Program poté vaši odpověď vyhodnotí, sdělí vám zda byla správná či naopak a vypíše slovo v anglickém i českém jazyce na obrazovku. V neposlední řadě je program zodpovědný za počítání správných a nesprávných odpovědí a navrhnutí známky za předvedené znalosti.

Popis objektů:

V projektu je celkem pět objektů typu *Tlačítko.* Objekty *tl1, tl2* a *tl3* slouží jako nabídka správných odpovědí, objektem *tl4* si můžete slovíčko znovu přehrát, pokud jste mu neporozuměli hned napoprvé a objektem *tl5* spustíte test znovu.

Pro zobrazení správné či špatné odpovědi je použit objekt *Želva (ž1),* díky své schopnosti měnit záběry.

Dále bylo vytvořeno sedm objektů typu *Text. Text5* pro zobrazení slova v českém i anglickém jazyce, *Text3* a *Text4* mají svou hodnotu nastavenu pevně na „*Right*“, respektive „*Wrong*“, jelikož slouží pouze jako ukazatel počtu odpovědí. Počet správných, respektive nesprávných odpovědí zobrazují objekty *Text1* a *Text2*. *Text6* má pevně nastavenu hodnotu „*Tvá známka:*“ a *text7* tuto známku zobrazuje.

Popis proměnných (řazení podle abecedy):

:ans – uchovává hodnotu, kterou vrací operace *dialog* volaná příkazem *vyhodnoť*

:audio – uchovává aktuální slovíčko

:odpo – pomocná proměnná využitá v příkazu *rozhoď*

:opak - uchovává hodnotu opakování, tj. kolikrát jsme již zavolali příkaz *rozhoď*

:rA, :wA – uchovává hodnotu správných, respektive špatných odpovědí

:sAudio – uchovává seznam slovíček, která ještě nebyla použita v aktuálním testu.

:slova – uchovává seznam všech slovíček. Jména těchto slovíček se shodují s názvem souboru, který dané slovíčko reprezentuje a který poté přehrává příkaz *přehrajZvuk*.

slovník – uchovává seznam všech slovíček v českém i anglickém jazyce ve formátu *české slovo anglické slovo*.

word – uchovává aktuální slovo v českém a anglickém jazyce

známka – uchovává hodnotu právě navržené známky

Popis příkazů (řazení dle chodu programu):

start – příkaz se spustí vždy při zapnutí programu. Nejprve pomocí příkazu *dialog* zobrazí dialogové okno s jedním tlačítkem OK a po jeho stisknutí spustí příkaz *ini*.

ini – příkaz nejprve skryje objekty *ž1* a *text5*, naplní proměnnou :s*Audio* proměnnou :*slova* a dosadí do proměnných :*opak, :rA, :rW* a *známka* základní hodnoty. Tyto hodnoty poté zobrazí pomocí příslušných objektů typu *Text* a nakonec spustí příkaz *rozhoď*.

rozhoď – příkaz nejprve skryje objekty *ž1* a *text5* a poté podmínkou zjistí, zda není proměnná :*sAudio* prázdná. Pokud je, znamená to, že jsme vyčerpali všechna slovíčka ze slovníku a test končí příkazem *vyhodnoť*. Pokud však proměnná :*sAudio* obsahuje ještě nějaká slovíčka, dosadí se do proměnné :*audio* náhodný prvek ze zbývajícího seznamu. Příkazu *náhodně* je zde vstupem číslo *11 – :opak*, čímž si zajistíme, že nebudeme vybírat prvek mimo tento seznam. Tento vybraný prvek se poté převede pomocí operace *(slovo "|$Media\| :audio)* na cestu k požadovanému souboru a ten se pomocí příkazu *přehrajZvuk* přehraje. Ze seznamu :s*Audio* se poté odebere prvek, který jsme právě použili. K tomu abychom přiřadili náhodnému tlačítku správnou odpověď, použijeme pomocnou proměnnou :*odpo*, do které dosadíme náhodné číslo z intervalu 1 – 3, které bude určovat jméno tlačítka, kterému pomocí příkazu *popis!* přiřadíme hodnotu správné proměnné. Poté pomocí procedury *vPřípadě :odpo* dosadíme zbylým dvou tlačítkům náhodné hodnoty z proměnné :*slova*. Nakonec zvětšíme hodnotu proměnné :*opak* o 1.

kontrola – tento příkaz je nastaven jako událost *přiZapnutí* všech třech tlačítek s odpověďmi, tj. *tl1*, *tl2*, *tl3*. Po stisknutí daného tlačítka se v příkazu *kontrola* podmínkou zjistí, zda se popis daného tlačítka rovná proměnné :*audio*. Pokud ano, znamená to, že jsme zodpověděli správně, zavolá se příkaz *porovnej*, viz níže, ukáže se objekt *ž1* se záběrem 1, ukáže se objekt *text5* s hodnotou, kterou vrátil příkaz *porovnej* a přičte se číslo jedna k proměnné :*rA*. Pokud ne, stane se to samé s tím rozdílem že objekt *ž1* se zobrazí se záběrem 2 a jednička se

tentokrát přičte k proměnné :*wA*. Poté se aktualizují hodnoty objektů *text1* a *text2* a podle počtu správných odpovědí i *text7*. Nakonec se počká 4 vteřiny, abychom měli čas si přečíst správnou odpověď a poté se zavolá příkaz rozhoď.

porovnej :co – vstupem tohoto příkazu je proměnná :*slovník*. V této proměnné se rekurzivním voláním hledá prvek shodný s právě aktuálním slovem a to tak, že pokud je první prvek seznamu shodný s hledaným slovem, dosadí se i s následujícím prvkem do proměnné :*word*. Pokud není, volá se příkaz *porovnej* se vstupem menším o první dva prvky.

vyhodnoť – tento příkaz se volá po zodpovězení poslední otázky. Nejprve otevře dialogové okno s dvěma tlačítky. Pokud odpovíme *Ano*, spustí se příkaz *ini* a tím i nový test, pokud odpovíme *Ne*, nebo okno zavřeme jiným způsobem, otevře se další dialogové okno se zprávou BYE BYE.

6.2 Videoprojekt

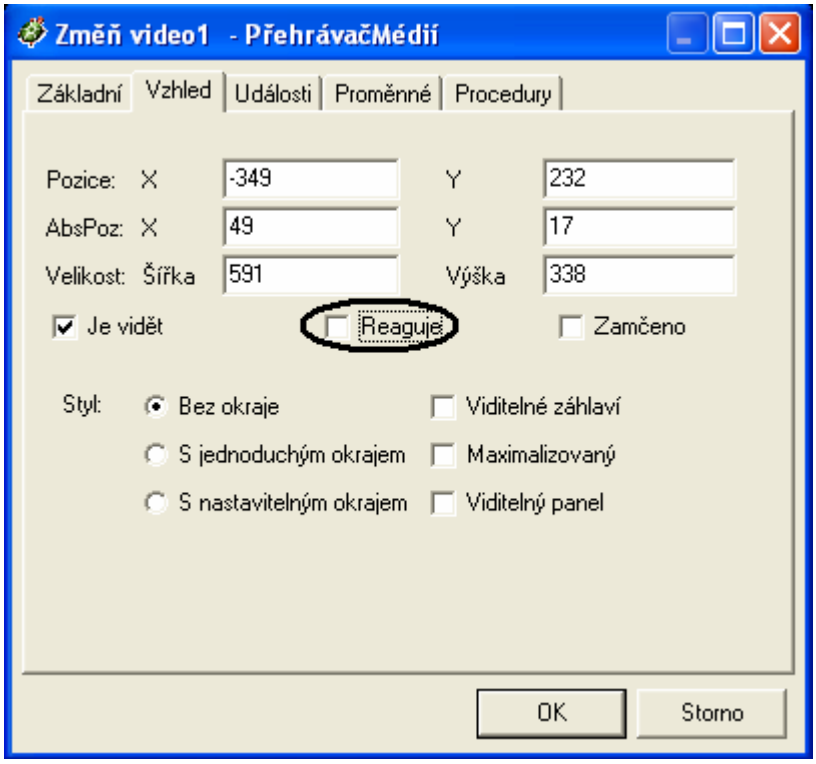
Nyní si představíme projekt pracující s videem.

6.2.1 Videopřehrávač

Tento projekt je jakýsi jednoduchý videopřehrávač, který v sobě zahrnuje některé funkce, které jsme si předvedli v *Kapitolách 4* a *5*. Těmito funkcemi jsou, načtení zvoleného typu souboru z disku, viz *Příklad 2 Kapitoly 4.2.2*, přehrávání nekonečné smyčky, viz *Příklad 1 Kapitoly 4.2.5*, a zpomalené přehrávání, viz *Příklad 1 Kapitoly 5.2*.

Popis objektů

V projektu je použit jeden objekt typu *Přehrávač médií (video1)*, který se stará o přehrávání zvoleného video souboru. Tento objekt má nastavenu hodnotu vlastnosti *Reaguje* na *ne* a vypnuté vlastnosti *Viditelné záhlaví* a *Viditelný panel* viz Obr. 29. To proto že jej chceme ovládat pouze vytvořenými tlačítky a posuvníky.



Obr. 29 Nastavení objektu *video1*

Dalšími objekty jsou čtyři objekty typu *Posuvník (p1, p2, p3, p4)*. Posuvník *p1* slouží k nastavení míry zpomalení efektu slow-motion. Posuvník *p3* pomocí příkazu *pozicePřehrávání* umožňuje nastavit pozici přehrávání daného videa. Posuvníky *p2* a *p4* slouží k nastavení začátku, respektive konce smyčky.

Dále je v projektu použito pět objektů typu *Tlačítko (tl1, tl2, tl3, tl4, tl5)*, respektive *Přepínač*. *Tl1* slouží k zapnutí zpomaleného přehrávání pomocí příkazu *SlowMo*, *tl4* pomocí příkazu *převiňNaZačítek* přesune pozici přehrávání do pozice 0. Tlačítka *tl3* a *tl5* jsou nastavena jako *přepínač*. *Tl3* při zapnutí spustí pomocí příkazu *přehraj_mm* přehrávání videa a zároveň zapne příkaz *spusť*, viz níže. Při vypnutí přehrávané video pomocí příkazu *přeruš* vypne a zároveň přeruší všechny běžící procesy příkazem *zastavVše*. *Tl5* slouží k zapnutí a vypnutí přehrávání nekonečné smyčky. Při zapnutí tedy spustí příkazy *loop* a *spusť*, viz níže. Při vypnutí ukončí přehrávání videa pomocí příkazu *přeruš* a zároveň přeruší všechny běžící procesy příkazem *zastavVše*. Poslední tlačítko *tl2*, slouží ke spuštění příkazu *vyber*, viz níže.

V projektu je využito celkem 17 objektů typu *text*, které jsou využity k popisům jednotlivých posuvníků a ke zobrazení času.

Popis proměnných:

:čas – uchovává přibližnou délku video souboru v sekundách, viz příkaz *time*

:hodiny, :minuty, :vteřiny – uchovává přibližný počet hodin, minut a vteřin spuštěného video souboru

:loopČas – uchovává hodnotu jednoho z posuvníků *p4* nebo *p2* převedenu na vteřiny, viz příkaz *loopČas*

:loopHodiny, :loopMinuty, :loopVteřiny – uchovává hodnotu jednoho z posuvníků *p4* nebo *p2* převedenu na počet hodin, minut a vteřin, viz příkaz *loopČas*

:uplČas - uchovává přibližný uplynulý čas přehrávaného video souboru v sekundách, viz příkaz *uplČas*

:uplHodiny, :uplMinuty, :uplVteřiny – uchovává přibližný počet uplynulých hodin, minut a vteřin spuštěného video souboru, viz příkaz *uplČas*

:pom – pomocná proměnná využitá v příkazu *vyber*

Popis příkazů:

loop – tento příkaz se stará o přehrávání nekonečné smyčky. V rekurzivním volání nejprve pomocí příkazu *přehraj_mm* spustí přehrávání video souboru a poté pomocí podmínky zjišťuje zda pozice přehrávání není rovna nebo větší hodnotě posuvníku *p2*. Pokud podmínka projde, přesune pozici přehrávání do pozice, kterou udává hodnota posuvníku *p4*.

slowMo – tento příkaz je shodný s příkazem *slowMo* z *Příkladu 1 Kapitoly 5.2.*

vyber – tento příkaz je modifikace příkazu použitého v *Příkladě 2 Kapitoly 4.2.2.* Nejprve pomocí příkazu *pomocníkNačtiSoubor* otevře dialogové okno pro výběr souboru zvoleného formátu. Pokud není zvolený soubor prázdným řetězcem, uloží se pomocí příkazu *soubor!* jako nový přehrávaný soubor objektu *video1*. Poté

pomocí příkazů *max!* a *dobaPřehrávání* nastaví maximální hodnotu posuvníků *p2, p3* a *p4* na délku přehrávaného video souboru.. To proto aby se po každém načtení nového video souboru přizpůsobila maximální velikost posuvníků délce videa. Nakonec spustí příkaz *time,* viz níže.

time – operace *dobaPřehrávání* vrací u video souboru počet jeho snímků (na rozdíl od audia, kde vrací délku v sekundách). Pokud tedy chceme zobrazovat délku daného video souboru ve formátu minuty – hodiny - vteřiny, musíme si nejprve tento počet snímků přepočítat. Jelikož každý video formát používá jiné snímkování, tj. jiný počet snímků za vteřinu, je vypočtený čas pouze přibližný.

Příkaz *time* tedy nejprve dosadí do proměnné :*čas* počet vteřin přehrávaného video souboru. Toto číslo získá vydělením počtu snímků, které vrací operace *dobaPřehrávání*, číslem 24, což je námi zvolený počet snímků za vteřinu. Dále pomocí matematických operací převede toto číslo na hodiny, minuty a zbylé vteřiny a dosadí je do příslušných proměnných. Nakonec pomocí těchto proměnných a objektů typu *text* zobrazí odhadovanou délku video souboru.

uplČas – tento příkaz se stará o výpočet uplynulého času. K tomuto výpočtu používá stejný přístup jako příkaz *time* s tím rozdílem, že pracuje s hodnotou, kterou vrací operace *pozicePřehrávání.* Tuto hodnotu nejprve převede stejně jako v předchozím příkazu na vteřiny a ty poté na hodiny a minuty. Daná čísla přiřadí odpovídajícím proměnným a zobrazí pomocí odpovídajících objektů typu *text.*

spusť – tento příkaz obsahuje pouze jednu řádku kódu a to `pořád [uplčas]`. Tento je využit pouze pro zpřehlednění kódů. Spustí nekonečný proces příkazu *uplČas* a tím se stará o stálou změnu daných proměnných a hodnot objektů *text*, které nám ukazují uplynulí čas video souboru.

loopČas :posuvník – příkaz s parametrem, kde parametr :*posuvník* udává u jakého objektu se má daný příkaz provést. Tento příkaz je použit jako událost *přiZměně* posuvníků *p2* a *p4* a stará se o změnu hodnot příslušných objektů typu *text*, podle toho pro jaký posuvník jsme jej použili. Těmto objektům poté pomocí příslušných

proměnných, do kterých se nejprve uložily vypočítané hodnoty stejně jako v příkazech *time* a *uplČas*, přiřadí správnou hodnotu hodin, minut a vteřin ve které se právě daný posuvník nachází.

7 Reference

[1] HOIDEKER, Jan. *Obecné MIDI (General MIDI)* [online]. 16.09.2000. Plzeň : Pavel Herout, 1999 [cit. 2009-04-20]. Dostupný z WWW: <http://www-kiv.zcu.cz/~herout/html_sbo/midi/5.html>.

[2] KAISEROVÁ, Zuzana. *Web, multimédia a streaming*. České Budějovice, 2002. 67 s. Jihočeská univerzita. Vedoucí diplomové práce PaedDr. Petr Pexa. Dostupný z WWW: <http://home.pf.jcu.cz/~pepe/Diplomky/kaiserova.pdf>.

[3] MICROSOFT, Corp. *Formáty multimediálních souborů programu Windows Media Player* [online]. c2009 [cit. 2009-04-22]. Dostupný z WWW: <http://support.microsoft.com/kb/316992/cs>.

[4] SVĚT HARDWARE, *MPEG*. [online]. 2007 [cit. 2009-04-22]. Dostupný z WWW: <http://www.svethardware.cz/glos.jsp?doc=F833B49F1A424215C1257346007ED399>. ISSN 1213-0818.

[5] VANÍČEK, Jiří, MIKEŠ, Radovan. *Informatika pro základní školy a víceletá gymnázia, 3. díl*. Učebnice. Brno: Computer Press, 2006. 96 s. ISBN 80-251-1082-6

[6] YOYO GAMES, Ltd. . *MCI* [online]. 2006-2008 [cit. 2009-04-22]. Dostupný z WWW: <http://wiki.yoyogames.com/index.php/MCI>.

Příloha B – CD s materiály

Obsahem tohoto CD jsou hotové příklady a projekty popsané v příručce. V adresáři Web se nachází webová forma interaktivní příručky „Multimédia v prostředí Imagine“ spustitelná souborem *index.html*. V adresáři Projekty\Média, jsou zvukové soubory použité v projektu *Test anglických slovíček.*

Popis souborů:

Název souboru	**Jméno příkladu**	**Číslo kapitoly**
Priklady\prehraj-Pr4.imp	Příklad 4: Změna hudebního nástroje během přehrávání melodie	4.1.2
Priklady\prehraj-Pr5.imp	Příklad 5: Kánon	4.1.2
Priklady\hrajNotu-Pr1.imp	Příklad 1: Jak zahrát náhodnou melodii	4.1.3
Priklady\hrajNotu-Pr2.imp	Příklad 2: Jak zahrát akord	4.1.3
Priklady\hrajNotu-Pr3.imp	Příklad 3: Jak naladit akord + změna nástroje Příklad 1: Jak zahrát tři tóny se zpožděním	4.1.3 4.1.4
Priklady\prehrajSoubor-Pr1.imp	Příklad 1: Demonstrace rozdílů mezi příkazy *přehrajZvuk a přehrajSoubor*	4.2.2
Priklady\prehrajSoubor-Pr2.imp	Příklad 2: Jednoduchý hudební přehrávač	4.2.2

Priklady\objektVideoklip-Pr1A.imp	Příklad 1: Loop – přehrávání nekonečené smyčky	4.2.5
Priklady\objektVideoklip-Pr1V.imp	Příklad 1: Slow - Motion – zpomalené přehrávání	5.2
Projekty\Beatmaker.imp	Beatmaker	6.1.1
Projekty\HraShranim.imp	Hra s hraním	6.1.2
Projekty\TestAnglickychSlovicek.imp	Test anglických slovíček	6.1.3
Projekty\Videoprehravac.imp	Videopřehrávač	6.2.1