Gymnázium, Praha 6, Arabská 14
předmět Programování, vyučující Mgr. Jan Lána

Syntezátor

maturitní projekt

Ema Heřmánková, 4.E

březen 2025

Maturitní projekt

Předmět: Programování

Téma: Syntezátor

Autor: Ema Heřmánková

Třída: IV. E

Školní rok: 2024/2025

Vedoucí práce: Mgr. Jan Lána

Třídní učitel: Mgr. Blanka Hniličková

Čestné prohlášení:

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 1. 4. 2025 .....

Anotace:

Tato dokumentace popisuje projekt syntezátoru, který je navržen tak, aby generoval různé zvuky a umožňoval uživatelům manipulovat s různými zvukovými parametry. Projekt využívá technologii JavaFX pro vytvoření grafického uživatelského rozhraní a zvukové API pro generování zvukových signálů. Dokumentace obsahuje přehled funkcí, architektury systému, implementace, uživatelskou příručku a návrhy na budoucí rozvoj. Dále zahrnuje přiznání zdrojů a citace použitých knihoven a frameworků.

Abstract:

This documentation describes a synthesizer project designed to generate various sounds and allow users to manipulate different audio parameters. The project utilizes JavaFX technology to create a graphical user interface and audio API for sound signal generation. The documentation includes an overview of features, system architecture, implementation details, a user manual, and suggestions for future development. Additionally, it includes acknowledgments and citations for the libraries and frameworks used in the project.

Zadání maturitního projektu

Cílem projektu je vývoj digitálního syntezátoru, který bude zahrnovat základní oscilátor, ADSR obálku (Attack, Decay, Sustain, Release – čtyři fáze tvarování zvukového signálu) a základní zvukové filtry. Tento syntezátor bude vybaven jednoduchým a intuitivním grafickým uživatelským rozhraním vytvořeným pomocí JavaFX, aby uživatelé mohli snadno ovládat všechny jeho funkce. Aplikace bude navržena jako samostatný desktopový program, což znamená, že bude fungovat bez nutnosti připojení k internetu. Celkové řešení se zaměří na minimalismus v designu i funkčnosti, aniž by to omezilo klíčové možnosti syntezátoru.

Platforma:

- Java
- JavaFX

Obsah

Úvod

Tento projekt se zaměřuje na vývoj softwarového syntezátoru, který poskytuje uživatelům možnost generovat a manipulovat se zvukovými signály v reálném čase. Syntezátor je navržen tak, aby byl intuitivní a uživatelsky přívětivý, což umožňuje široké veřejnosti i hudebníkům experimentovat s různými zvukovými parametry, jako jsou hlasitost, ladění, šířka, barva a hloubka zvuku.

Projekt je postaven na technologii Java a využívá JavaFX pro vytvoření moderního a esteticky příjemného uživatelského rozhraní. Uživatelé mohou pomocí ovládacích prvků, jako jsou rotující knoby a tlačítka, snadno měnit nastavení syntezátoru a přepínat mezi různými typy vln (sine, square, saw). Součástí aplikace je také osciloskop, který vizualizuje generovaný zvuk a poskytuje tak uživatelům okamžitou zpětnou vazbu o jejich nastavení.

V této dokumentaci budou podrobně popsány jednotlivé komponenty projektu, jejich funkce, architektura systému a pokyny k použití. Kromě toho se zaměříme na možnosti dalšího vývoje a vylepšení syntezátoru, aby bylo možné reagovat na potřeby uživatelů a přizpůsobit se trendům v oblasti zvukové produkce a hudební technologie.

1 Přehled

Tento projekt představuje software pro syntezátor, který je určen pro generování a manipulaci se zvukovými signály. Hlavními komponentami projektu jsou:

1. **Hlavní aplikace (MainApp)**: Tato třída slouží jako vstupní bod pro aplikaci a zajišťuje načtení uživatelského rozhraní (UI) z FXML souboru. Taktéž se stará o přiřazení CSS stylů pro vizuální vzhled aplikace.

2. **Ovladač syntetizátoru (SynthController)**: Tato třída se zabývá logikou aplikace a spravuje interakce uživatelů s ovládacími prvky. Obsahuje metody pro nastavení knobů, ovládání vln a startování či zastavování generace zvuku.

3. **Rotující ovladač (RotatorControl)**: Tato třída představuje jednotlivé ovládací prvky (knoby), které umožňují uživatelům měnit hodnoty různých parametrů zvuku, jako jsou hlasitost, ladění, šířka, barva a hloubka. Každý knob je graficky reprezentován a reaguje na uživatelské interakce.

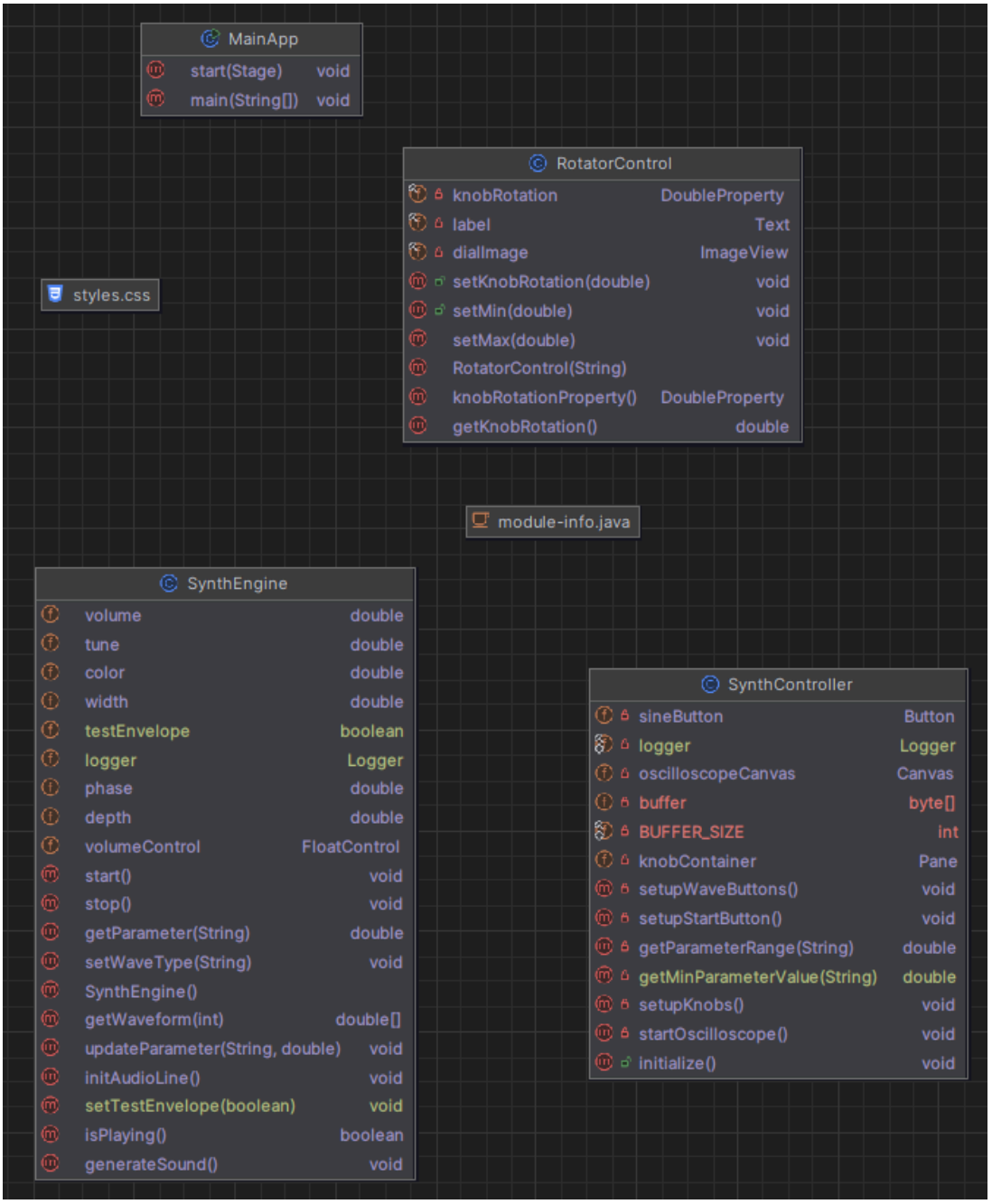
4. **Zvukový motor (SynthEngine)**: Tato třída zajišťuje generaci zvuku a správu parametrů syntetizátoru. Obsahuje metody pro nastavení typu vlny, generaci zvukových signálů a implementaci obálky ADSR (Attack, Decay, Sustain, Release) pro modulaci zvuku.

5. **Grafický osciloskop**: Aplikace obsahuje vizualizaci generovaného zvuku pomocí osciloskopu, který zobrazuje aktuální vlnovou formu. Tato funkce pomáhá uživatelům vizualizovat změny v parametrech zvuku a zajišťuje interaktivní zážitek.

Projekt je zaměřen na uživatelskou přívětivost a estetický vzhled, čímž umožňuje snadnou manipulaci s parametry zvuku a podporuje kreativní proces uživatelů. Dále se projekt snaží reagovat na potřeby uživatelů a moderní trendy v oblasti hudební technologie, což otevírá možnosti pro další rozvoj a vylepšení aplikace.

2 Architektura programu

Architektura tohoto syntezátoru je založena na modulárním návrhu, kde jednotlivé komponenty mají jasně definované odpovědnosti a komunikují mezi sebou prostřednictvím událostí a datových propojení. Projekt využívá MVC (Model-View-Controller) architekturu, která zajišťuje oddělení uživatelského rozhraní, logiky aplikace a samotného zvukového enginu.



Obr. 1 UML diagram projektu

2.1 Hlavní komponenty a jejich role

Aplikace je navržena podle architektury MVC (Model-View-Controller), která odděluje generování zvuku, uživatelské rozhraní a řízení interakcí.

Model, reprezentovaný třídou SynthEngine, zajišťuje samotnou syntézu zvuku. Obsahuje metody pro výběr typu vlnové formy, jako je sínusová, obdélníková nebo pilovitá vlna. Dále implementuje klíčové zvukové parametry, včetně frekvence, hlasitosti a ADSR obálky, která umožňuje definovat dynamiku zvuku. Jelikož SynthEngine neobsahuje žádné závislosti na uživatelském rozhraní, lze jej snadno rozšířit o další funkcionality nebo integrovat do jiných aplikací.

Uživatelské rozhraní, tedy View, je definováno pomocí JavaFX FXML, což umožňuje oddělit vzhled od logiky aplikace. Styly a vizuální úpravy jsou spravovány v souboru styles.css, který určuje celkový estetický dojem, včetně tmavého vizuálního stylu aplikace. Klíčovým prvkem UI je RotatorControl.java, který implementuje otočné knoby umožňující plynulé nastavení parametrů zvuku. Součástí rozhraní je také vizualizace osciloskopu, která poskytuje zpětnou vazbu o aktuálně generovaném zvukovém signálu.

Logiku propojení mezi modelem a uživatelským rozhraním zajišťuje SynthController. Tato komponenta zpracovává uživatelské vstupy, jako je otáčení knobů, výběr vlnové formy nebo spouštění a zastavování zvuku. Hodnoty získané z UI předává do SynthEngine, kde se promítají do změn zvukového signálu. Současně SynthController aktualizuje zobrazení rozhraní tak, aby odráželo aktuální nastavení parametrů.

Celou aplikaci spouští třída MainApp, která inicializuje JavaFX scénu a načítá soubory FXML, propojuje kontrolery s vizuálními prvky a nastavuje základní parametry aplikace. Dále se stará o aplikaci stylování, což zajišťuje jednotný vzhled a přehlednost uživatelského rozhraní.

2.2 Tok dat a komunikace mezi komponentami

Interakce uživatele s uživatelským rozhraním začíná manipulací s ovládacími prvky. Pokud uživatel otočí knobem v komponentě RotatorControl, změní tím odpovídající hodnotu parametru. Kliknutím na tlačítko pro změnu vlnové formy odešle aplikace odpovídající příkaz pro přepnutí generovaného zvuku.

Třída SynthController.java je zodpovědná za zpracování uživatelských vstupů. Přeposílá nové hodnoty parametrů do zvukového enginu SynthEngine, čímž ovlivňuje charakter generovaného zvuku. Současně řídí aktualizaci uživatelského rozhraní, například tak, že zobrazí změněné hodnoty parametrů v odpovídajících ovládacích prvcích.

Zvukový engine SynthEngine na základě aktuálně nastavených parametrů generuje odpovídající vlnovou formu. Výstupní signál je následně směrován do zvukového výstupu, kde je možné jej slyšet.

Součástí aplikace je také vizualizace zvukového signálu pomocí osciloskopu. Tento modul zpracovává výstupní signál generovaný SynthEngine a v reálném čase zobrazuje průběh zvukové vlny. Díky tomu uživatel vidí vizuální reprezentaci zvuku a může lépe pochopit, jak změna parametrů ovlivňuje výsledný signál.

2.3 Použité knihovny

JavaFX je hlavní knihovna, která byla použita pro tvorbu grafického uživatelského rozhraní aplikace syntezátoru. Tato knihovna umožňuje snadnou interakci s uživatelskými prvky, jako jsou knoby, tlačítka a osciloskop, a poskytuje nástroje pro jejich stylizaci. Díky JavaFX může být uživatelské rozhraní vizuálně atraktivní a intuitivní, což usnadňuje uživatelům manipulaci s různými zvukovými parametry.

Java Sound API je další klíčová knihovna, která byla implementována v rámci tohoto projektu. Používá se pro generování a manipulaci se zvukovými signály, což je nezbytné pro zajištění správného fungování syntetizátoru. Tato API umožňuje práci se zvukovými kanály a poskytuje nástroje pro přehrávání a zpracování zvuku. Díky Java Sound API je zajištěn kvalitní výstup zvuku, což je klíčové pro uživatelský zážitek a pro přesnost generovaných zvukových signálů.

3 Implementace

V této kapitole podrobně popíšeme implementaci jednotlivých částí projektu syntezátoru. Zaměříme se na klíčové třídy a mechanismy, které zajišťují jeho funkčnost.

3.1 Hlavní třída aplikace: MainApp.java

Třída MainApp slouží jako vstupní bod pro aplikaci syntezátoru a zajišťuje inicializaci JavaFX prostředí. V metodě start se načítá FXML soubor, který definuje strukturu uživatelského rozhraní aplikace, a vytváří se scéna s danými rozměry. Dále se přidává CSS soubor pro stylování aplikace, čímž se zajišťuje její vizuální atraktivita. Po nastavení všech komponentů se okno aplikace zobrazuje uživateli. Třída také obsahuje metodu main, která spouští aplikaci.

3.2 Ovládací třída: SynthController.java

Třída SynthController.java slouží jako spojovací prvek mezi uživatelským rozhraním a zvukovým enginem v aplikaci syntezátoru. Odpovídá za inicializaci komponentů, správu uživatelských vstupů a řízení generování zvuku. Tato třída je klíčová pro zajištění interakce uživatele s aplikací, a to prostřednictvím otočných ovladačů a tlačítek.

Při spuštění aplikace se v metodě initialize inicializuje instance třídy SynthEngine, která je zodpovědná za generaci zvuku. Následně se nastavují knoby, tlačítka pro výběr vln a tlačítko pro spuštění nebo zastavení syntezátoru. Metoda setupKnobs vytváří ovládací prvky pro každý parametr zvuku a přidává je do rozhraní. Každý knob je propojen s příslušným parametrem v SynthEngine a při změně jeho hodnoty se automaticky aktualizuje.

Dále metoda setupWaveButtons zajišťuje, že uživatel může vybrat typ vlny (sinusovou, obdélníkovou nebo pilovitou), což má přímý vliv na generovaný zvuk. Tlačítko startButton ovládá spuštění a zastavení zvuku, čímž uživateli umožňuje experimentovat s různými nastaveními v reálném čase.

Nakonec metoda startOscilloscope pravidelně aktualizuje grafiku osciloskopu na základě generovaných zvukových vln, což poskytuje vizuální zpětnou vazbu o aktuálním výstupu syntezátoru. Tato funkce zvyšuje interaktivitu aplikace a usnadňuje uživatelům sledování změn v parametrech zvuku. Celkově třída

SynthController.java hraje zásadní roli při zajištění intuitivního a efektivního uživatelského rozhraní.

3.3 Zvukový engine: SynthEngine.java

Třída SynthEngine je klíčovým prvkem softwarového syntezátoru, který se stará o generování zvuku na základě uživatelských parametrů. Tato třída definuje hlavní funkce syntezátoru, včetně výběru typu vlny, řízení frekvence, hlasitosti a implementace ADSR obálky (Attack, Decay, Sustain, Release) pro modelaci zvukových dynamik.

Jednou z hlavních funkcí této třídy je metoda initAudioLine, která inicializuje zvukovou linku pro výstup zvuku. Tato metoda nastavuje parametry audio formátu a spouští linku pro přehrávání. Třída také zahrnuje metody pro start a stop generování zvuku, přičemž při spuštění se vytváří nové vlákno, které zajišťuje kontinuální generaci zvukových signálů.

Dále třída obsahuje metodu getWaveform, která na základě aktuálně nastavených parametrů generuje zvukovou vlnu. Tato metoda využívá různé vzorce pro výpočet hodnot pro různé typy vln (sínusová, obdélníková, pilovitá) a aplikuje na ně dynamické úpravy pomocí obálky ADSR. Celkově třída SynthEngine efektivně propojuje zvukovou syntézu s uživatelským rozhraním, což uživatelům umožňuje interakci s různými zvukovými parametry a vizualizaci generovaného zvuku

3.4 Uživatelský ovládací prvek: RotatorControl.java

Třída RotatorControl.java reprezentuje grafický ovládací prvek pro nastavení parametrů syntezátoru pomocí otočného knobu. Je odvozena od třídy VBox, což umožňuje uspořádání komponent v vertikálním směru.

V konstruktoru třídy je inicializován otočný knob, jehož grafická reprezentace je vytvořena pomocí obrázku načteného ze souboru. Dále je přidán textový štítek, který slouží k identifikaci funkce knobu. Třída obsahuje metody pro nastavení a získání hodnoty rotace knobu, stejně jako pro definici minimálních a maximálních hodnot, což zajišťuje, že uživatel nemůže nastavit hodnoty mimo definované rozsahy.

Interakce s uživatelským rozhraním je zajištěna pomocí událostí myši. Při stisknutí a táhnutí myši dochází k rotaci knobu, čímž se aktualizuje jeho hodnota. Třída také poskytuje metody pro přístup k minimálním a maximálním hodnotám a k textovému štítku, což přispívá k její flexibilitě a použitelnosti v rámci aplikace syntezátoru.

3.5 Uživatelské rozhraní: hello-view.fxml

Soubor hello-view.fxml představuje klíčový komponent pro uživatelské rozhraní aplikace syntezátoru, implementovaný pomocí jazyka FXML, který je součástí technologie JavaFX. Tento soubor definuje strukturu a rozložení grafických prvků, které uživatelům umožňují interakci s funkcemi syntezátoru.

Hlavní struktura uživatelského rozhraní je organizována pomocí kontejneru VBox, který zajišťuje vertikální uspořádání prvků. V rámci tohoto kontejneru jsou umístěny jednotlivé komponenty, jako je osciloskop, kontejner pro knoby a tlačítka pro výběr vlnové formy. Osciloskop je zobrazen pomocí prvku Canvas, který poskytuje prostor pro vizualizaci generovaného zvuku, což je užitečné pro uživatele, kteří chtějí sledovat aktuální vlnovou formu.

Kontejner FlowPane pro knoby je navržen tak, aby umožnil flexibilní uspořádání otočných ovladačů, přičemž se přizpůsobuje dostupnému prostoru. Tímto způsobem mohou uživatelé snadno manipulovat s různými parametry zvuku, jako jsou frekvence, hlasitost a další.

Tlačítka pro výběr vlnové formy (sine, square, saw) jsou umístěna v horizontálním kontejneru HBox, což umožňuje přehlednou a snadno ovladatelnou volbu typu vlny, kterou chce uživatel generovat. Kromě toho je zde také tlačítko pro spuštění a zastavení syntetizátoru, které má větší rozměry a přizpůsobený styl pro snadnou identifikaci.

Celkově je hello-view.fxml navržen tak, aby byl esteticky příjemný a uživatelsky přívětivý, s tmavým pozadím a barevnými prvky, které usnadňují orientaci a manipulaci. Tento soubor je klíčovým prvkem, který odděluje vzhled aplikace od její logiky, což je v souladu s principy moderního návrhu softwaru.

3.6 Ovládací knoby a jejich implementace

3.6.1 Hlasitost (Volume)

- **Popis:** Tento knob umožňuje uživatelům nastavit hlasitost výstupu zvuku. Hodnota se pohybuje v rozmezí od 0.0 (úplné ztlumení) do 1.0 (maximální hlasitost).

- **Implementace:** Hlasitost je uložena jako proměnná v třídě SynthEngine.java a její hodnota se aplikuje při generování zvuku, čímž ovlivňuje amplitudu vlny.

3.6.2 Ladění (Tune)

- **Popis:** Knob pro ladění nastavuje frekvenci generovaného zvuku, v rozsahu od -1000 Hz do 1000 Hz.

- **Implementace:** Tato hodnota se převádí na frekvenci, kterou syntetizátor používá k výpočtu vlny. Uživatel může pomocí tohoto ovládacího prvku experimentovat s tónovými variacemi.

3.6.3 Šířka (Width)

- **Popis:** Šířka určuje, jakým způsobem bude zvukový signál modulován, zejména u čtvercové vlny.

- **Implementace:** Tato hodnota ovlivňuje, jak dlouho trvá každá fáze signálu, což umožňuje změnu charakteru zvuku.

3.6.4 Barva (Color)

- **Popis:** Knob pro barvu ovlivňuje timbre zvuku tím, že mění poměr mezi sinusovými a kosinusovými složkami v generovaných vlnách.

- **Implementace:** Tato hodnota se používá k míchání dvou různých vlnových forem, což výsledně mění zvuk.

3.6.5 Hloubka (Depth)

- **Popis:** Hloubka určuje, jak intenzivně se zvuk moduluje přidáním harmonických složek, zejména u pilovité vlny.

- **Implementace:** Uplatňuje se při generování pilovité vlny, přidává se harmonická složka, čímž se mění charakter zvuku.

3.6.6 ADSR obálka (ADSR Envelope)

Popis: ADSR obálka, což je zkratka pro Attack, Decay, Sustain a Release, je klíčovým prvkem pro modelaci dynamiky zvuku. Tato obálka určuje, jak se zvuk postupně zintenzivňuje, ustupuje a zaniká.

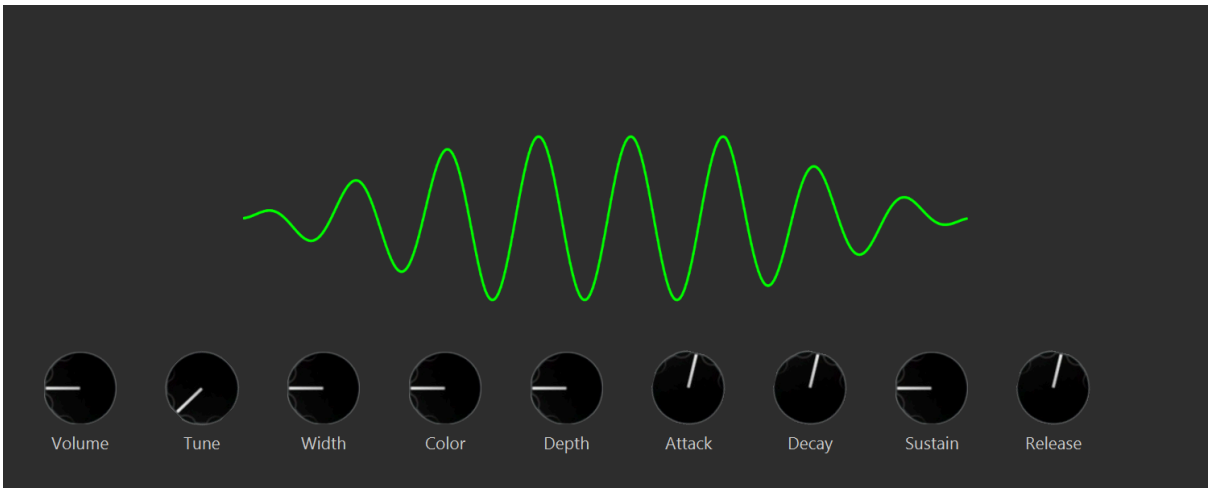
- **Attack (nástup):** Čas, během kterého zvuk dosahuje maximální intenzity od nuly.

- **Decay (pokles):** Čas, který zabere přechod od maximální intenzity k hodnotě sustain.
- **Sustain (udržení):** Úroveň intenzity, kterou zvuk udržuje po dobu trvání zvuku, dokud není ukončen.
- **Release (uvolnění):** Čas, který trvá, než zvuk zcela vymizí po uvolnění klávesy nebo ovládacího prvku.

Implementace: V syntetizátoru je ADSR obálka implementována jako součást generování zvuku v třídě SynthEngine.java. Při každém vzorkování se vypočítávají hodnoty pro jednotlivé fáze obálky na základě aktuálního času a nastavených parametrů. Když je zvuk spuštěn, časovač sleduje dobu, po kterou trvá každá fáze, a postupně modifikuje intenzitu výstupu podle definovaných hodnot pro Attack, Decay, Sustain a Release. Tímto způsobem obálka ADSR ovlivňuje dynamiku zvuku a vytváří přirozenější a expresivnější zvukové výstupy.

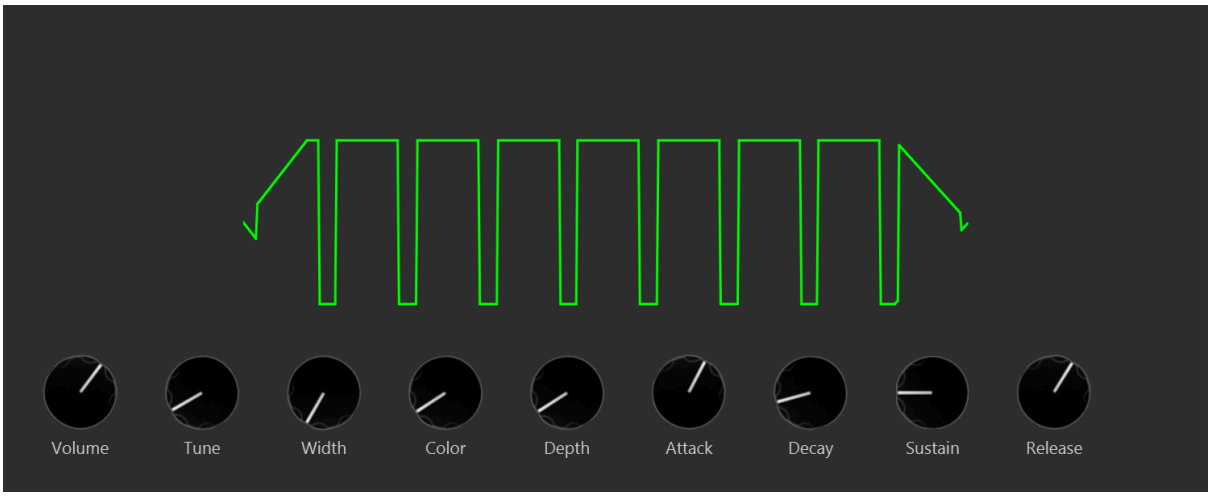
3.7 Generování vlnových forem

Sinusová vlna se vytváří pomocí metody Math.sin(), která vrací hodnoty mezi -1 a 1 na základě aktuální fáze signálu. Tato metoda generuje hladký a čistý tón, který je charakteristický pro přírodní zvuky. Sinusova vlna je základem mnoha dalších typů zvuků, a to díky své jednoduchosti a jedinečné kvalitě. Její hladký průběh zajišťuje, že nedochází k žádným ostrým přechodům, což činí tento typ vlny ideálním pro produkci tónů v hudbě.



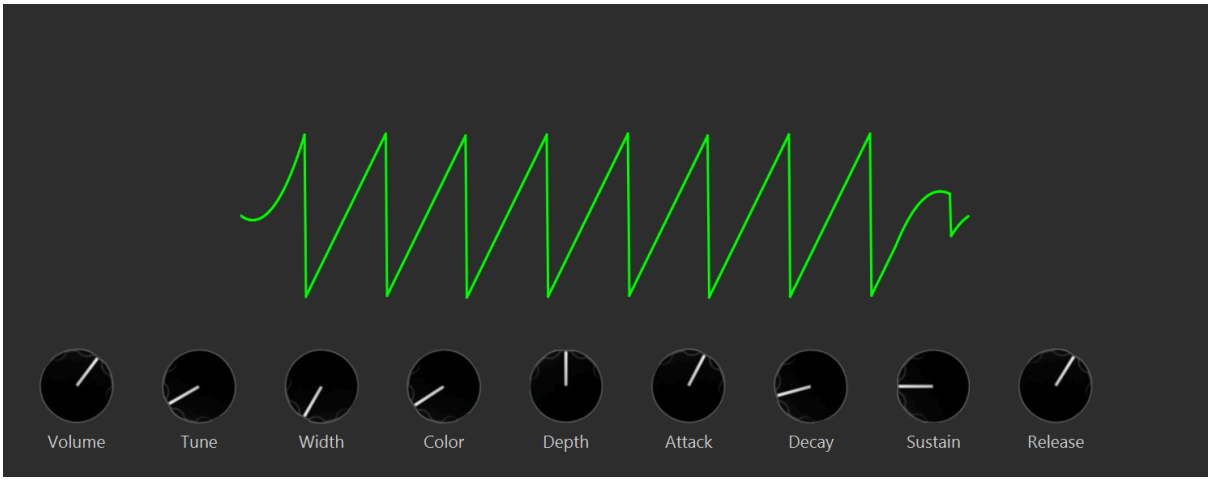
Obr. 2 Sinusová vlna na osciloskopu

Obdélníková vlna se generuje na základě fáze, která je porovnávána s nastaveným parametrem šířky. Tento parametr určuje, jak dlouho bude signál na vysoké úrovni a jak dlouho na nízké úrovni. Tímto způsobem vznikají ostré přechody mezi těmito dvěma hodnotami, což dodává obdélníkové vlně její charakteristický zvuk. Tento typ vlny se často používá v elektronické hudbě a syntéze, protože vytváří silné a agresivní zvuky, které se snadno modulují.



Obr. 3 Obdélníková vlna na osciloskopu

Pilovitá vlna se generuje pomocí lineárního vzorce, který mapuje fázi na hodnoty mezi -1 a 1. Při generaci pilovité vlny se signál postupně zvyšuje a poté rychle klesá, což vytváří charakteristický tvar. Kromě toho se k základní pilovité vlně přidává harmonická složka, která je určena parametrem hloubky. Tento typ vlny je široce používán v syntéze a zvukovém designu, zejména při vytváření basových tónů a padů.



Obr. 4 Pilovitá vlna na osciloskopu

3.8 Problémy během vývoje

Během vývoje syntezátoru se objevily určité problémy, které ovlivnily interakci s uživatelským rozhraním a funkčnost osciloskopu.

Jedním z hlavních problémů byla interakce uživatelského rozhraní. Uživatelé někdy neví, jaké hodnoty jsou přípustné pro jednotlivé ovládací prvky. To může vést k zmatku a frustraci při používání aplikace. Je proto důležité mít jasné vizuální indikátory a nápovědy, které pomohou uživatelům lépe pochopit, jaké hodnoty mohou nastavit a jakým způsobem ovlivňují zvukové výstupy.

Dalším problémem bylo dynamické škálování osciloskopu. Někdy se stalo, že vizualizace generovaných zvukových vln neodpovídala skutečnému zvuku. Tento nesoulad mohl být způsoben špatným výpočtem amplitudy, což vedlo k nepřesnostem v zobrazení osciloskopu. Pro zajištění správné vizualizace je nezbytné pečlivě zkontrolovat algoritmy, které měří a zpracovávají amplitudu zvuku.

4 Uživatelská příručka

Tato příručka slouží jako návod k použití softwarového syntezátoru. Obsahuje informace o instalaci, ovládání a funkcích aplikace.

4.1 Instalace a spuštění

Požadavky na systém:

- **Operační systém:** Windows, macOS, Linux
- **Java:** JDK 17 nebo novější
- **JavaFX:** Musí být součástí projektu
- **Zvuková karta** s podporou 44,1 kHz samplovací frekvence

Instalace:

1. Naklonujte repozitář

   git clone https://github.com/gyarab/2024-4e-hermankova-syntezator.git

2. Otevřete projekt v IDE
   - Doporučeno: IntelliJ IDEA nebo Eclipse s podporou Maven.

3. Zkontrolujte závislosti
   - JDK 17+

4. Spusťte aplikaci
   - Hlavní třída: MainApp.java
   - Příkaz: mvn javafx:run

4.2 Možné problémy a jejich řešení

Problém	**Možné řešení**
Aplikace nelze spustit	Ujistěte se, že máte správně nainstalovanou Javu (JDK 17+).
Zvuk nehraje	Zkontrolujte, zda není aplikace ztlumená a zda máte zapnutý zvuk v systému.
Knoby nereagují	Restartujte aplikaci a ujistěte se, že myš není zaseknutá.
Chyba při spuštění	Spusťte aplikaci z příkazového řádku a podívejte se na chybovou zprávu.

4.3 Ukončení aplikace

Aplikaci lze ukončit třemi způsoby:

1. Zavřením okna křížkem.
2. Stisknutím **Alt + F4** (Windows) nebo **Cmd + Q** (Mac).
3. Ukončením procesu v příkazovém řádku (Ctrl + C).

Závěr

Tento projekt softwarového syntezátoru představuje funkční aplikaci, která umožňuje generování a manipulaci se zvukovými signály prostřednictvím intuitivního uživatelského rozhraní. Implementace využívá moderní přístupy k vývoji v jazyce Java s využitím knihovny JavaFX pro vizuální zobrazení a interakci s uživatelem.

Během vývoje byla zvláštní pozornost věnována modularitě kódu, což umožňuje snadnou rozšiřitelnost a úpravy. Díky objektově orientovanému návrhu lze do budoucna přidávat další funkce, například podporu více oscilátorů, různé typy filtrů či možnost ukládání a načítání předvoleb uživatele.

Uživatelské rozhraní bylo navrženo s důrazem na jednoduchost a přehlednost. Interaktivní prvky, jako jsou otočné knoby a tlačítka, umožňují intuitivní ovládání syntezátoru bez nutnosti složitého nastavování.

Celkově projekt splnil očekávání stanovená na začátku vývoje. V budoucnu by bylo možné rozšířit jeho funkcionalitu o pokročilejší zvukové efekty, MIDI podporu či integraci s externím hardwarem. Tento projekt tak může sloužit jako základ pro další vývoj a zdokonalování v oblasti digitálního zvuku a syntézy.

Seznam použitých obrázků

Seznam zdrojů a použité literatury

- **JavaFX API Documentation**
  Oficiální dokumentace k JavaFX poskytuje detailní informace o třídách a metodách používaných v rámci platformy JavaFX. Dostupné z: https://docs.oracle.com/javase/8/javafx/api/

- **OpenJFX – Domovská stránka projektu**
  Oficiální stránka projektu OpenJFX, která nabízí zdroje, dokumentaci a komunitní podporu pro vývojáře pracující s JavaFX. Dostupné z: https://openjfx.io/

- **Internetové aplikace v JavaFX**
  Diplomová práce poskytující hlubší pohled na tvorbu internetových aplikací s využitím technologie JavaFX. Autor: Jan Novák. Dostupné z: https://is.muni.cz/th/e4n4w/diplomka.pdf

- **The Synthesis ToolKit in C++ (STK)**
  Open-source framework pro zvukovou syntézu a zpracování audia, který sloužil jako inspirace pro některé části zvukového enginu. Obsahuje implementace oscilátorů, filtrů a obálek, které byly analyzovány při návrhu struktury SynthEngine. Dostupné z: https://ccrma.stanford.edu/software/stk/

- **Java Sound API Guide**
  Dokumentace k Java Sound API, které bylo použito pro základní zvukové operace a manipulaci s audiem v rámci projektu. Dostupné z: https://docs.oracle.com/javase/tutorial/sound/