Проект по Визуелно Програмирање
Документација за Ritmiko – Ритам игра со Македонски народни песни

1. Опис на апликација

Ritmiko е ритам игра слична на игри како Guitar Hero, нотите се појавуваат на дното на екранот и играчот треба да го притисне точниот тастер кога ке се преклопат нотата и индикаторот.

Овај документ опишува упатсвото за користење, features на играта и техничката имплементација.

2. Упатство за користење

Навигација низ мениа:

На насловната страна (Figure 1) имаме три опции:

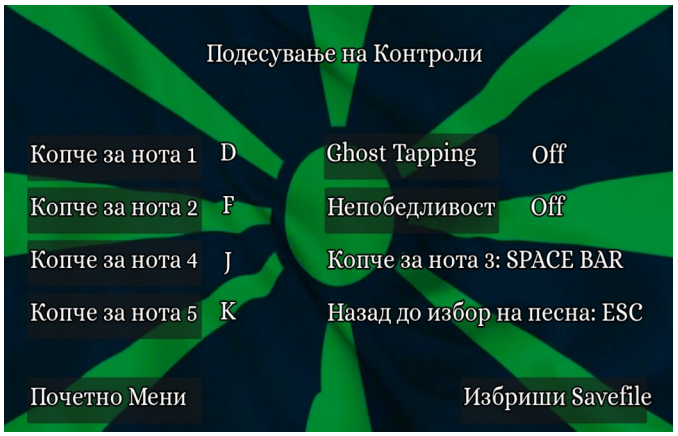
Старт: Не носи на екранот за избор на песна.

Контроли: Не носи на екранот за подесување на контролите.

Излез: Ја исклучува апликацијата.

Figure 1: Насловна Страна / Main Menu

На екранот за подесување на контроли (Figure 2), можеме да ги промениме типките кои се мапирани за секоја од петте ноти (освен нота три, која е секогаш мапирана на Space Bar) користејќи ги копчињата „Note 1 Key”, „Note 2 Key”, „Note 4 Key” и „Note 5 Key”.



Притискање на едно од овие копчиња ќе создаде prompt кој му кажува на играчот да притисне на произволен тастер (Figure 3). Кога играчот ќе кликне на тастер по свој избор, соодветната нота ќе биде мапирана според изборот на играчот.

Притискање на копчето „Ghost Tapping” ќе ја промени опцијата од „Off” во „On” или обратно. Кога „Ghost Tapping” е на „On”, притискање на кое било од копчињата за нота ако нема нота ќе резултира во „Miss!” што ќе биди објаснато подолу во делот кој опишува Gameplay. Оваа информација можеме да ја видиме и доколку направиме hover over со маусот што ќе покажи tooltip со соодветната информација.

Притискање на копчето „Непобедливост“ ќе ја промени опцијата од „Off” во „On” или обратно. Кога „Непобедливост“ е на „On”, не можеме да добиеме Game Over без разлика колку пати промашиме нота. Исто како и за опцијата за Ghost Tapping, и ова копче има tooltip што го опишува ова.

Доколку го притисниме копчето „Избриши Savefile“, текстот на копчето ќе се промени во „Потврди“. Доколку тогаш го притиснеме копчето уште еднаш, контролите ќе се ресетираат на почетните вредности и сите резултати кои сме ги добиле при играње на песните ќе се избришат.

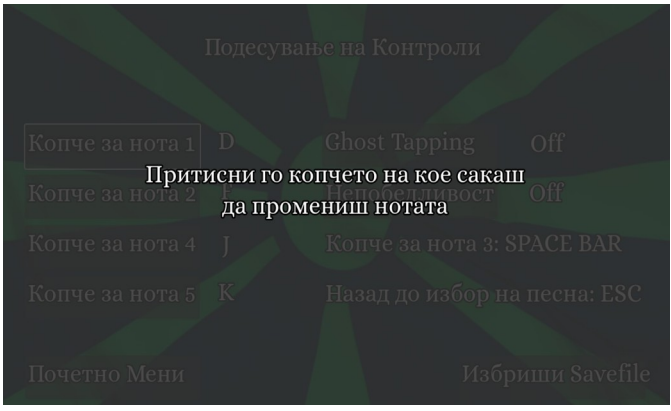


Figure 3: Промена на тастер

Конечно, доколку притиснеме на „Почетно Мени“ се враќаме на насловната страна.

На екранот за избор на песна (Figure 4) на левата страна можеме со кликање на копчињата да одбериме која песна сакаме да ја играме.

Доколку направиме hover со глувчето врз некоја од песните, излегува статистиката од највисокиот резултат кој сме го постигнале на таа песна, оцената според резултатите, како и нивото на тежина (Figure 5 и 6). Доколку не ја имаме играно песната претходно, добиваме приказ како на Figure 5 бидејќи немаме резултати, додека ако ја имаме изиграно довиваме приказ како на Figure 6. Оцената која ја добиваме зависи од точноста која сме ја постигнале:

Figure 4: Мени за избор на песна

Figure 5: Приказ за резултати - нема предходно играно

>90%	SS
>80%	S
>70%	A
>60%	B
>50%	C
<=50%	D

Figure 6: Приказ за резултати - има предходно играно

Gameplay:

Имаме приказ од екранот за време на gameplay (Figure 7).

Figure 7: Gameplay екран

Ноти се појавуваат на дното на екранот и се движат нагоре. Целта на играта е да се притисне соодветното копче кога ќе се преклопат копчето и нотата (како што е прикажано на Figure 9).

Има 2 типови на ноти: обични (Figure 10) кои треба само да се притиснат и легато ноти (Figure 8) за кои треба да се држи копчето се додека не завршат.

Има три различни рејтинзи кој што играчот може да ги добие од секоја нота зависно со која прецизност ќе ја притиснеме: „Perfect!”, “Good!” и „Miss!”. Доколку добиваме „Perfect!” или “Good!”, комбото се зголемува. Доколку добиеме „Miss!”, комбото се ресетира на 0. Дополнително, „Perfect!” ни дава 75 поени, “Good!” дава 50 и „Miss!” одзема 25.

Figure 9: Нота преклопена со индикатор

Ќе знаеме кој тајминг сме го постигнале благодарение на text popups кои излегуваат после притискање на секоја нота (Figure 11).

Figure 10: Обични ноти

Figure 8: Легато ноти

Figure 11: Рејтинг индикатор

Долу лево е прикажана статистика од моменталните резултати на играчот кои се апдејтираат после секоја нота.

„Accuracy/Точност“ прикажува со каков тајмниг се притискаат нотите која се пресметува според разликата од точниот тајминг и тајмингот на играчот изразен во 1/100 дел од секунда. „Perfect“, „Hit” и „Miss” го прикажуваат бројот на ноти на кои играчот постигнал „Perfect!”, “Good!” и „Miss!” рејтингс соодветно. „Longest Combo” го прикажува најдолгото комбо кое сме го постигнале за време на песната.

Figure 12: Game Over екран

Конечно, на десната страна на екранот имаме мерач за енегијата од играчот. Енергијата се полни со „Perfect!”, “Good!” и се намалува со „Miss!” рејтинзи. Доколку енергијата достигне 0, добиваме Game Over и песната запира (Figure 12). Копчето „Обиди се повторно“ ја рестартира песната, додека „Почетно Мени“ не враќа на насловната страна.

Има и пример за како изгледа gameplay на Github репозитериумот: gameplay.mp4.

3. Техничка имплементација

Source code од проектот се наоѓа во директориумот source_code/scenes/*.cs на Github репозитериумот.

Godot Game Engine:

Заради ограниченият број на страни на документацијата, наместо јас да опишам основите на Godot Game Engine, ќе поставам линк до поопширната документација на Godot а во документацијата на Ritmiko ќе се фокусирам само на кодот и класите кои јас ги имам имплементирано.

Godot документација: https://docs.godotengine.org/en/stable/index.html

Класи и начин на чување на податоци

Song Charts

Информациите за како и со кој тајминг играта да создава ноти за соодветната песна се складирани во посебни фајлови наречени Song Charts. Овие фајлови се .txt фајлови во кои во форма на plain text е складирана HashMap<String, List<int>>.

Клучовите е времето (изразено во стотти дел од секунда) во кое се создаваат ноти кои се опишани во вредноста, односно List.

List<int> содржи пет вредности кои опишуваат секоја колона од нотите. Доколку е 0, ништо не се случува; доколку е 1 се создава обична нота; доколку е 2, се создава легато нота; и 4 го означува крајот на легато нотата.

Извадок од ваков file (source_code/song_charts/makedonsko_devojche.txt):

{"-95":[0,0,0,0,2],"-25":[0,0,0,1,0],"-22":[0,0,0,0,4],"7":[0,2,0,0,0],"80":[0,4,0,0,2],"153":[0,0,0,1,4],"183":
[0,0,0,0,1],"217":[0,0,1,0,0],"412":[0,0,0,0,2],"474":[0,0,0,1,4],"511":[0,2,0,0,0],"572":[2,0,0,0,0],"577":
[0,4,0,0,0],"638":[4,0,1,0,0],"709":[0,0,1,0,0],"895":[0,0,0,0,2],"963":[0,0,0,1,0],"964":[0,0,0,0,4],"1004":
[0,2,0,0,0],"1064":[2,0,0,0,0],"1065":[0,4,0,0,0],"1141":[4,0,0,1,0],"1172":[0,0,0,0,1],"1210":[0,0,1,0,0],"1392":
[0,0,0,0,2],"1507":[0,0,0,2,0],"1513":[0,0,0,0,4],"1636":[0,2,0,4,0],"1748":[0,4,1,0,0],"1890":[0,0,0,1,0],"1923":
[0,0,0,0,1],"1956":[0,0,0,1,0],"1996":[2,0,0,0,0],"2062":[4,0,0,0,0],"2064":[0,0,0,1,0],"2095":[0,1,0,0,0],
...

Овие фајлови се наоѓаат во директориумот source_code/song_charts/ на Github репозитериумот.

Нота

Нотите се објекти во кои се складирани следните информации:

int	lifetime	Опишува колку за колку време нотата ќе престане да постои и ќе резултира во „Miss” доколку не биде притисната. Се намалува за 1 секој стотти дел од секунда.
float	speed	Брзината со која нотата се движи нагоре, променувајќи ја својата Y позиција за дадената вредност секој стотти дел од секунда.
String	noteType	Опишува однесувањето на нотата и се користи во методите од нотата.
int	row	Одредува во која колона на ноти ќе се создаде. Нотата го менува и изгледот доколку row==2 бидејќи тогаш се појавува во колоната каде нотата е SPACE BAR.

Самиот објект се состои од Control Node (кој го користам за да ја одредам позицијата) и три текстури (една за обична нота, друга за обична нота кога е во колоната од SPACE BAR и трета за легато ноти која се одредува со noteType и row променливите).

Освен давање на рејтинг „Miss” кога lifetime ќе достигне вредност од -13, освен естетски, нотите немаат удел во играта. Детекцијата за притискање на ноти се наоѓа во source_code/scenes/playernotes.cs додека создавањето на нотите е овозможено од source_code/scenes/notes_spawner.cs.

Menu Button/Копче за мени

Menu Button е класата која ја имаат сите копчиња во мениата. Сама по себе нема функционалности освен анимација. Функцијата на секое копче е одредена од објектот родител на копчињата така што класата Menu Button повикува функцијата "call_function" и "hover_function" на родителот кога копчето ќе се притисне/ќе се направи hover со глувчето соодветно. Овие функции потоа повикуваат други функции кои извршуваат одредени операции. Ова го имам вака дизајнирано за да може да се преискористи истиот код за секое копче, променувајќи го кодот дирекно во Parent објектот зависно од потребата. Класата Menu Button ги содржи следните податоци:

String	parentFunctionId	Функцијата која копчето ја повикува кај родителот кога ќе биде притиснато.
String	hoverFunctionId	Функцијата која копчето ја повикува кај родителот кога ќе биде hovered со глувчето.
String	labelText	Текстот кој го пишува на копчето.
int	minimumWidth	Ширината на позадината на која е поставен текстот и големината на просторот кој детектира дали е hovered од глувчето.
bool	stayHovered	Дали ќе се обои текстот жолто кога глувчето ќе излезе од копчето, означувајќи дека е последното копче врз кое што глувчето седело hovered.

Song Data/Податоци за секоја песна

Song Data е посебна класа во која се чуваат информациите за секоја песна која може да се игра. Содржи следните информации:

String	title	Насловот на песната кој се прикажува на екранот за избор на песна.
String	difficulty	Тежината која се прикажува на екранот за избор на песна.
String	noteSpeed	Ја одредува „speed“ променливата на сите ноти во песната.
List<String>	playerCharts	Името на Song Chart фајлови за песната. Причината што е листа е за да овозможи лесна идна имплементација на повеќе играчи доколку има потреба.
String	audiopath	Filepath до аудио фајлот за песната.
Int	songLength	Одредува после колку време завршува песната и се зачувуваат резултатите на играчот. Изразено во стоттини секунди.

SongResults.save

SongResults.save е фајлот во кој се снимаат највисоките резултати постигнати од играње на песните. Ги складира информациите во форма на HashMap т.е. Dictionary<String, List<float>>.
Keys се ID стринговите од песните, додека листата ги содржи освоените поени, Accuracy/точност и најдолгото комбо, во тој редослед, кои потоа се користат за прикажување на највисоко освоените резултати во менито за избор на песна.

controlsSetup.save

controlsSetup.save е, слично на SongResults.save, фајл на кој се снимаат податоците за копчињата за нотите, како и дали се овозможени сетинзите за GhostTapping и Непобедливост.
Се сериализира променливата keys (Array<String>), ghostTapping(bool) и invincibility(bool) што се наоѓаат во глобално вчитаната скрипта source_code/scenes/settings.cs и се снимаат во фајл.

Скрипти и функционалности

Заради рестрикцијата на 5 страни, ќе ги опишам само клучните функционалности на најважните скрипти:

source_code/scenes/note_menu.cs: Оваа скрипта одредува се од Gameplay освен детекцијата за притискање ноти. Во оваа скрипта контролира кога ќе заврши, како и податоците за поени, точност, комбо, итн.

_Ready(): Иницијализира песната така што ги зема податоците од соодветниот SongData фајл, одредувајќи ја должината, соодветните SongCharts на соодветните играчи (во овој случај, само еден), итн.

_Process(): Функција која секој physics frame (стотти дел од секунда) проверува дали поминало доволно време за да заврши песната во кој случај ги зачувува податоците и се враќа на менито за избор на песна.

noteHit(int lifetime = 0, String result = "Perfect"): Функцијата која се справува со како рејтизите од отсвираните ноти ќе имаат ефект врз статистиките од играчот и ја пресметува вредностите на точноста, комбото, поените и енергијата на играчот. Иста така ги пушта соодветните анимации и звучни ефекти и ги повикува функциите UpdateStatistics() и createPrompt().

UpdateStatistics(): го апдејтира нивниот приказот на статистиките од играчот.

createPrompt(String text, Color color): создава техтуален prompt кој го означува рејтингот на отсвирената нота.

HoldNote(): ги зголемува поените се додека се држи легато нота.

source_code/scenes/playernotes.cs: Оваа скрипта се справува со детекција на свирење на нотите. Оваа скрипта има 5 деца, секое репрезентира секоја колона од ноти. На овие деца се додаваат нотите како деца, со кои се извршува детекцијата на дали нотата е соодветно кликната односно „отсвирена“.

_Ready(): Иницијализира буквите кои се прикажани на индикаторите за ноти според controlsSetup.save фајлот.

_Process(): Детектира кои копчиња се притиснати. Прво, доколку некое копче е притиснато, индикаторите визуелно се променуваат за да покажат дека играчот ги има притиснато соодветните копчиња.
Потоа, функционално, оваа скрипта го зема lifetime атрибутот на првото дете на соодветната колона чие копче е притиснато. Доколку lifetime атрибутот на нотата дете со индекс 0 е во временските рамки опишани со променливите/константите „perfectWindow” и „goodWindow”, тогаш се повикува функцијата noteHit(noteChild.lifetime, постигнатRating) од note_menu скриптата и истовремено се повикува built in функцијата QueueFree() од нотата, која ја бриши од меморија. Доколку тајмингот е „Perfect”, се повикува и функцијата createIndicator(notePosition).
Користејќи ја променливата List<bool> holdDownReq исто така детектира и управува со легато нотите, повикувајќи ја функцијата holdNote() од note_menu секој physicsFrame доколку соодветно се отсвири легато нотата се додека не заврши или играчот ја пушти премногу рано, резултирајќи во „Perfect”/„Miss” рејтинг соодветно.

createIndicator(Vector2 position, bool isSpaceNote = false): Создава визуелен индикатор со цец да прикажи дека играчот имал „Perfect” рејтинг. Position одредува позицијата на индикаторот, додека isSpaceNote одредува дали текстурата треба да се промени во текстура соодветна за SPACE BAR нотата.