SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE
dokumentace

3D Hra (Get out) v Godot enginu

Jeroným Baron

Obor:	18-20-M/01 INFORMAČNÍ TECHNOLOGIE se zaměřením na počítačové sítě a programování
Třída:	IT4
Školní rok:	2023/2024

Poděkování

- *Rád bych vyjádřil své upřímné poděkování panu Mgr. Markovi Lučnému za cennou podporu a konzultace, které mi poskytoval před i během tvorby této závěrečné práce.*

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2023

podpis autora práce

ABSTRAKT

Tato dokumentace se věnuje procesu tvorby 3D hry žánru dungeon crawler v Godot Enginu. Projekt spojuje moderní trendy herního vývoje s cílem vytvořit příjemně hratelný titul, který hráče zabaví a nabídne pohlcující herní zážitek. Dokumentace zahrnuje podrobný popis práce s Godot Enginem, seznámení s programovacím jazykem GDScript, a postupy při tvorbě 3D modelů v softwaru Blender.

Hlavním cílem bylo vytvoření hratelné hry s propracovaným AI a pathfinding systémem pro nepřátele, kteří se pohybují v ručně navržených úrovních. Dokumentace se zaměřuje na postup vývoje, řešení technických problémů a získané poznatky během práce. Text také reflektuje, jak moderní technologie a nástroje umožňují jednotlivcům vytvářet komplexní herní projekty a posouvat své dovednosti v oblasti herního designu, programování a 3D modelování.

Tento dokument může posloužit jako inspirace a zdroj informací pro začínající vývojáře her, kteří se chtějí naučit pracovat s Godot Enginem, vytvářet vlastní 3D projekty nebo pochopit základní principy vývoje herních mechanik.

ABSTRACT

This documentation focuses on the process of creating a 3D dungeon crawler game using the Godot Engine. The project combines modern trends in game development with the goal of delivering an enjoyable and engaging game experience. It includes a detailed description of working with the Godot Engine, an introduction to the GDScript programming language, and the process of creating 3D models using Blender.

The main objective was to develop a playable game featuring advanced AI and pathfinding systems for enemies navigating custom-designed levels. The documentation highlights the development process, solutions to technical challenges, and lessons learned during the project. It also reflects on how modern tools and technologies enable individuals to create complex game projects while improving their skills in game design, programming, and 3D modeling.

This document can serve as inspiration and a valuable resource for aspiring game developers who want to learn how to work with the Godot Engine, create their own 3D projects, or understand the fundamental principles of game mechanics development.

OBSAH

ÚVOD

Hry v dnešní době představují jednu z nejpopulárnějších forem zábavy, která spojuje technologické inovace s kreativním vyprávěním příběhů. Moderní hry sahají od jednoduchých mobilních aplikací až po komplexní tituly s otevřeným světem, které hráčům nabízejí stovky hodin zábavy. Díky vývoji technologií, jako je virtuální realita, ray tracing a cloudové hraní, se zážitek z her stává stále více pohlcujícím a přístupným. Herní průmysl také přináší silný sociální prvek, ať už prostřednictvím online multiplayerových her, nebo komunit na streamovacích platformách, kde hráči sdílejí své zážitky.

Cílem mého projektu bylo naučit se pracovat a používat Godot engine, seznámit se a porozumět jazyku GDScript. Zdokonalit se v práci a vytváření 3D modelů v Blendru a během toho vytvořit hru která bude příjemně hratelná a dokáže zabavit. Mým stanoveným cílem bylo vytvořit hratelnou hru, ve které budou nepřáteli s funkčním AI a pathfindingem systémem. Ve vlastních úrovních.

V této dokumentace se pokusím ukázat svůj postup práce. Společně s postřehy, které jsem během vývoje nasbíral.

1 TEORETICKÁ A METODICKÁ VÝCHODISKA

1.1 Cíle projektu

Hlavním cílem projektu bylo vytvoření plně funkční 3D hry žánru dungeon crawler, která by byla příjemně hratelná a dokázala zaujmout hráče. Při práci na projektu jsem se zaměřil na několik klíčových oblastí: seznámení a práce s herním enginem Godot, osvojení programovacího jazyka GDScript, a zdokonalení dovedností při tvorbě 3D modelů v softwaru Blender. Dalším důležitým cílem bylo implementovat funkční AI systém pro nepřátele, který zahrnuje pathfinding pro pohyb nepřátel v prostředí.

1.2 Herní prvky

Hra byla tvořena pro žánr dungeon crawler znamenaje, že cílem hry je procházet temnými katakombami a nasbírat co nejvíce zlata a esencí ideálně bez toho, aby hráč upozornil jakéhokoli nepřítele. Hlavní mechanikou je míra nebezpečný, která varuje hráče, protože čím déle bude hráč uvnitř tím více věcí může najít ale i tím nebezpečnější, rychlejší a pozornější se stávají nepřátelé.

Druhou mechanikou je využití balíčku karet který nabízí různé bonusy, které pomáhají hráči při postupu hlouběji do katakomb.

1.3 Get out v budoucnu

Hru hodlám dále vyvíjet i po odevzdání. V budoucích verzí by mělo být podstatně více úrovní, silnější nepřátelé a rozvinutá mechanika balíčku karet, který by měl zvyšovat základní atributy hráče a jeho schopnost procházet danými úrovněmi.

2 VYUŽITÉ TECHNOLOGIE

2.1 Seznam technologii

Během vývoje hry byly použity následující technologie a nástroje:

- **Godot Engine**: Hlavní herní engine pro tvorbu hry. Godot nabízí robustní systém scén, integrované nástroje pro tvorbu 2D a 3D her a podporu pro GDScript, VisualScript i programování v C#.
- **GDScript**: Programovací jazyk specifický pro Godot Engine. Jedná se o vysokoúrovňový orientovaný jazyk inspirovaný Pythonem, který usnadňuje rychlé prototypování a tvorbu herních mechanik.
- **Blender**: Software pro tvorbu 3D modelů, animací a texturování. Blender byl využit k vytváření herních objektů, postav a dalšího obsahu.
- **OpenGL**: Renderovací framework, který Godot využívá pro vykreslování 2D a 3D grafiky.

2.2 Výhody práce v Godot Enginu

1. **Otevřený zdrojový kód**: Godot je open-source, což znamená, že je zdarma k dispozici a lze ho modifikovat podle potřeb projektu.
2. **Jednoduché rozhraní**: Intuitivní a uživatelsky přívětivé rozhraní umožňuje rychlý začátek i pro začátečníky.
3. **Podpora pro 2D i 3D**: Godot poskytuje vysoce optimalizované nástroje pro vývoj 2D i 3D her, což umožňuje flexibilitu při tvorbě různorodých projektů.
4. **Integrovaný skriptovací jazyk (GDScript)**: GDScript je snadný na naučení a velmi efektivní pro rychlý vývoj prototypů.
5. **Lehká instalace a nízké nároky**: Godot je nenáročný na hardware, což umožňuje vývoj i na starších zařízeních.
6. **Silná komunita a dokumentace**: Komunita kolem Godotu je aktivní a poskytuje řadu tutoriálů, řešení problémů a doplňkových pluginů.

2.3 Nevýhody práce v Godot Enginu

1. **Méně rozšířené oproti jiným engineům**: Ve srovnání s Unity nebo Unreal Engine nemá Godot tak rozsáhlou podporu třetích stran a integrací.
2. **Nižší výkon v komplexních 3D projektech**: I když se Godot zlepšuje v oblasti 3D, není tak optimalizovaný jako konkurence při tvorbě velmi rozsáhlých nebo graficky náročných projektů.
3. **Menší míra předpřipraveného obsahu**: Godot obsahuje méně vestavěných šablon a assetů, což může vyžadovat více práce při vývoji od nuly.
4. **Specifický jazyk GDScript**: I když je GDScript snadný na použití, nemusí vyhovovat všem vývojářům, kteří jsou zvyklí na univerzálnější jazyky, jako je C++ nebo C#.
5. **Menší zázemí pro komerční projekty**: Pro velké studio může být Godot omezenější v oblasti marketingové podpory a profesionálních služeb.

Godot Engine je ideální volbou pro jednotlivce a menší týmy, kteří chtějí vytvořit kreativní projekty s minimálními náklady a učit se moderním postupům herního vývoje.

3 PROGRAMOVÁNÍ A POUŽITÉ POSTUPY

3.1 GDScript

GDScript je skriptovací jazyk specificky navržený pro herní engine Godot. Svou syntaxí se podobá Pythonu, což z něj činí snadno srozumitelný a čitelný jazyk, zejména pro začátečníky. GDScript je přímo integrovaný do Godotu, což znamená, že umožňuje efektivní práci s herními objekty a scénami bez potřeby externích knihoven. Mezi hlavní výhody patří jednoduchost, rychlý vývojový cyklus díky vestavěné podpoře v editoru a optimalizace pro výkon při práci v herním prostředí. Díky tomu je GDScript ideální pro rychlé prototypování i plnohodnotný vývoj her.

3.1.1 Příklad jednoduchého GDScriptu

Toto je jednoduchý příklad GDSriptu pro levitaci mince

```gdscript
extends Area3D

const ROT_SPEED = 100
var move_speed = 0.005
var move_y = 0
var add = false

# Called when the node enters the scene tree for the first time.
func _ready() -> void:
	pass # Replace with function body.

# Called every frame. 'delta' is the elapsed time since the previous frame.
func _process(delta: float) -> void:
	for body in get_overlapping_bodies():
		if body.is_in_group("player"):
			queue_free()
			Global_Vars.coins = Global_Vars.coins + 1
	rotate_y(deg_to_rad(ROT_SPEED * delta))

	if move_y <= -0.35:
		add = true
	if move_y >= 0.35:
		add = false
	if add == true:
		move_y += move_speed
	if add == false:
		move_y -= move_speed

	position.y += move_y * delta * 0.5
```

3.1.2 Příklad deklarace objektů v kódu

V Godotu je klíčové slovo `@onready` použito k deklaraci proměnných, které mají být inicializovány až ve chvíli, kdy je uzel, ke kterému odkazují, připraven. To je užitečné, protože některé uzly nemusí být dostupné při samotném startu skriptu, ale jsou připravené až po inicializaci celé scény.

```
@onready var neck := $Neck
@onready var camera := $Neck/Camera3D
@onready var pause_menu := $Neck/Pause_Menu
@onready var death_menu := $Neck/Death_Menu
@onready var standing_collision := $standing_collision
@onready var crouching_collision := $crouch_collision
@onready var standing_ray_cast := $RayCast3D

@onready var Spell_slot: int = Global_Vars.Spell_slot
```

Tímto způsobem se proměnná nastaví na uzel až ve chvíli, kdy je celý uzel ke stromu připojen a připraven, což eliminuje potenciální chyby způsobené předčasným přístupem k uzlům.

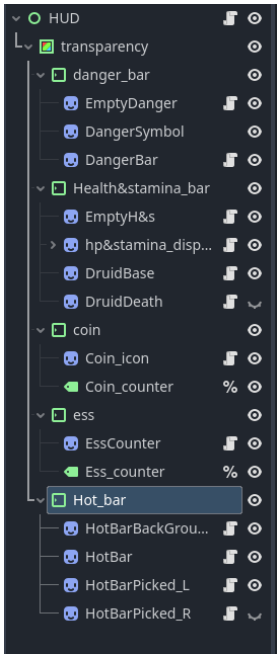
3.2 Hierarchie scén

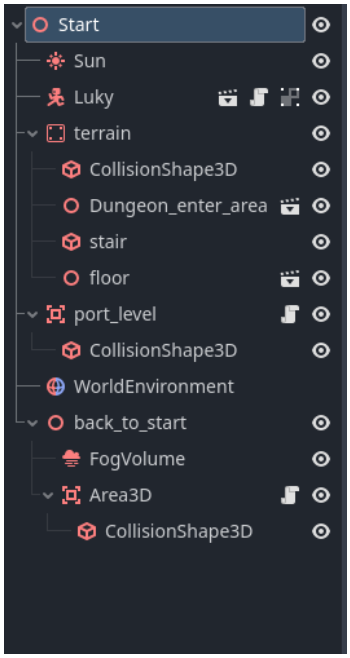
Hierarchické stavění scén v Godot enginu je klíčovým konceptem, který usnadňuje organizaci a správu herních prvků. Každá scéna je tvořena stromovou strukturou uzlů, kde každý uzel má svůj specifický typ a funkci, například zobrazení, fyziku nebo logiku. Díky hierarchii mohou uzly dědit vlastnosti od svých rodičů, což zjednodušuje opakované používání komponent. Tato organizace umožňuje, aby změny provedené na rodičovském uzlu automaticky ovlivnily všechny jeho potomky, například transformace, viditelnost nebo aktivace.

Scény lze snadno vnořovat, což umožňuje vytvoření modulárních a znovupoužitelných částí hry, jako jsou postavy, prostředí nebo UI prvky. Tento přístup výrazně zvyšuje efektivitu vývoje a přehlednost projektu.

3.2.1 Příklady Hierarchií

Zde jdou vidět příklady hierarchií dvou odlišných scén jedna pro HUD (Heads up display) a druhá pro začáteční oblast ve které celá hra začíná.

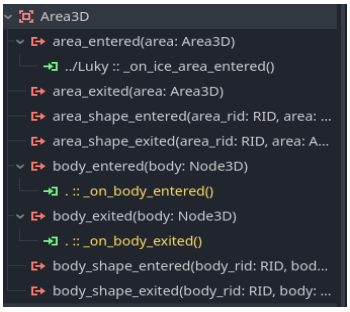




3.3 Signály

Signály v Godotu jsou výkonný nástroj pro komunikaci mezi uzly, který usnadňuje tvorbu flexibilního a modulárního kódu. Fungují na principu událostí – uzel může vysílat signál, když dojde k určité události (např. kliknutí, změna stavu), a další uzly na tento signál mohou reagovat připojenými metodami. Tento přístup eliminuje potřebu pevného propojení mezi objekty, což zlepšuje čitelnost a udržovatelnost kódu. Například pomocí signálu můžete jednoduše upozornit rodičovský uzel na dokončení akce potomkem, aniž by oba uzly o sobě věděly konkrétní detaily. Signály jsou klíčem k tvorbě čistého, událostmi řízeného systému v Godot enginu.

3.3.1 Příklad signálů



4 PRŮBĚH A VÝSLEDKY

4.1 Časová osa vývoje

4.1.1 Pohyb A Ovladatelnost

Prvním krokem pro vývoj 3D hry byl Pohyb a plynulá ovladatelnost Hráčské postavy.

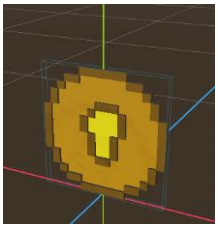
```
# Get the input direction and handle the movement/deceleration.
var input_dir := Input.get_vector("left", "right", "forward", "back")
direction = lerp(direction,(transform.basis * Vector3(input_dir.x, 0, input_dir.y)).normalized(),delta*friction)
if direction:
    velocity.x = direction.x * curent_speed
    velocity.z = direction.z * curent_speed
else:
    velocity.x = move_toward(velocity.x, 0, curent_speed)
    velocity.z = move_toward(velocity.z, 0, curent_speed)

move_and_slide()
```

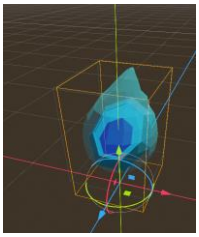
4.1.2 HUD (Heads Up Display)

Druhým splněným krokem byl funkční a responsivní HUD, který hráči dává vědět všechny potřebné informace bez toho aby byl zbytečně přehlcený.

4.1.3 Sběratelné předměty



Prvně bylo potřeba vytvořit herní měnu tak aby hráči měly co získávat a za co kupovat nová vylepšení



Poté jsem ještě vytvořil druhou měnu za kterou se dají kupovat hrací karty které poskytují silné krátkodobé bonusy

4.1.4 Menu pozastavení

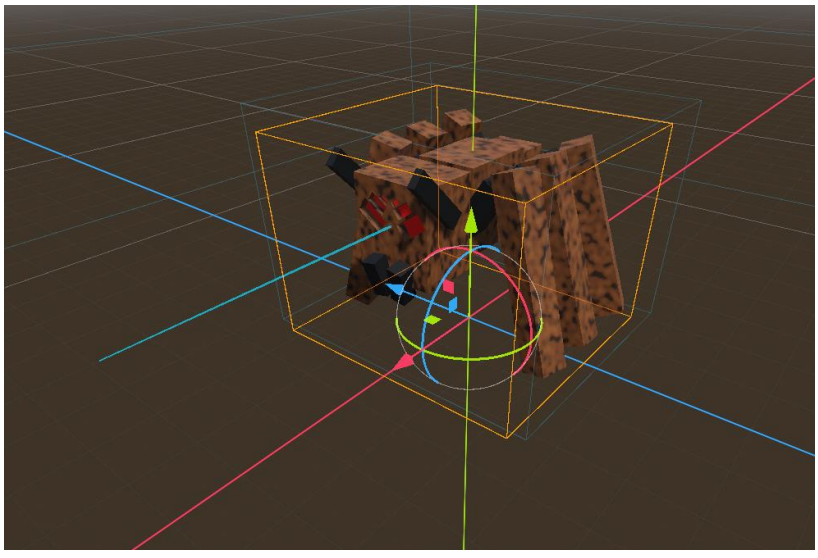
Následovně bylo potřeba vytvořit lehké Pause Menu aby šlo hru kdykoli pozastavit

```
func pauseMenu():
    if paused:
        pause_menu.hide()
        Engine.time_scale = 1
    else:
        pause_menu.show()
        Engine.time_scale = 0

    paused = !paused
```

4.1.5 Nepřátelé

Další podstatnou částí hry jsou nepřátelé, kteří nebudou dělat zrovna jednoduché posbírat všechny mince a esence.



ZÁVĚR

Cílem této práce bylo navrhnout a vytvořit 3D dungeon crawler hru pomocí Godot enginu, která demonstruje klíčové principy vývoje her, jako je hierarchická správa scén, interakce mezi herními objekty a využití signálů. Výsledkem je plně funkční prototyp, který obsahuje základní herní mechaniky, jako je pohyb hráče, sbírání předmětů a interakce s nepřáteli.

Toto řešení má praktické uplatnění jako výukový materiál pro začínající vývojáře nebo základ pro další rozvoj herního projektu. Díky modularitě a přehledné architektuře lze hru snadno rozšiřovat o nové prvky, jako jsou složitější nepřátelské AI, propracované bojové mechaniky nebo systémy generování náhodných dungeonů.

Do budoucna by bylo možné zaměřit se na optimalizaci výkonu pro větší a komplexnější dungeony, integraci pokročilých grafických efektů nebo přidání multiplayerových funkcí. Tímto způsobem může projekt sloužit nejen jako základ pro další vývoj, ale také jako inspirace pro tvorbu podobných her v Godot enginu.

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] *Godot fórum*

Dostupné z: https://godotforums.org/

[2] *Oficiální dokumentace godot enginu*

Dostupné z: https://docs.godotengine.org/en/stable/

[3] *Lukky, Godot 4.X : Ultimate First Person Controller Tutorial (2023)*

Dostupné z: https://www.youtube.com/watch?v=xIKErMgJ1Yk

[4] *Lukky, Godot 4.X : Ultimate First Person Controller Tutorial Part 2(2023)*

Dostupné z: https://www.youtube.com/watch?v=WF7d21zOD0M&t