Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Umělá inteligence herního souboje pro Unreal Engine

2023

Šimon Malíček

Vedoucí práce:
Mgr. Tomáš Mikula

Studijní program: Informační technologie,
prezenční forma

Bibliografické údaje

Autor:	Šimon Malíček
Název práce:	Umělá inteligence herního souboje pro Unreal Engine
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2023
Studijní program:	Informační technologie, prezenční forma
Vedoucí práce:	Mgr. Tomáš Mikula
Počet stran:	62
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Šimon Malíček
Title:	Combat AI for video games in Unreal Engine
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2023
Study program:	Information Technologies, full-time form
Supervisor:	Mgr. Tomáš Mikula
Page count:	62
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Tato práce se zabývá návrhem a implementací souboj s umělou inteligencí v reálném čase prostřednictvím technologie Unreal Engine. Demonstrace souboje umožňuje souboj dvojice agentů, první agent je ovládán hráčem, druhý umělou inteligencí.

V práci jsem popsal a použil techniky z oblasti herního návrhu pro vytvoření co nejlepšího pocitu ze hry. Vytvořil jsem dva typy umělé inteligence základní a pokročilou, každou pomocí odlišných systémů, které Unreal Engine nabízí. Základní pomocí systému Blueprints a pokročilou pomocí systému Behavior Tree.

Výsledkem práce je soubojový systém a jeho hratelná demonstrace. Systém mohou herní vývojáři implementovat do své hry nebo použít jeho části. Soubojový systém byl otestován a kladně hodnocen hráči, kteří převážně hrají podobný styl her.

Synopsis

The aim of this bachelor thesis is to design and implement real-time combat with an AI using Unreal Engine. The playable demonstration showcases a fight between a pair of agents, the first agent is controlled by the player, the second by the artificial intelligence.

In the thesis, I described and implemented game design techniques to create a good game feel. I created two types of AI, basic and advanced, each using a different system that the Unreal engine offers. The Basic AI is using the Blueprints system and the advanced AI is using the Behavior Tree system.

The outcome of this thesis is a combat system and its playable demonstration. Game Developers can implement the system into their games or use some parts of it. The system has been tested and rated positively by players who play primarily video games in the same genre.

Klíčová slova: Unreal Engine; Umělá inteligence; 3D hra; Herní návrh; Herní design; Vývoj počítačových her; 3D grafika; Videoherní průmysl; Umělá inteligence počítačové hry; Unreal Engine umělá inteligence; Herní logika; animace; skeletální model

Keywords: Unreal Engine, Unreal Engine AI, Unreal Engine combat system, Game Development, 3D game, 3D graphics, Game Development industry, Artificial Intelligence, Game Design, Game Feel, skeletal mesh

Děkuji všem, kteří mě při vytváření práce podporovali, a to jmenovitě vedoucímu práce Mgr. Tomášovi Mikulovi za odborné vedení a cenné rady na konzultacích.

Odevzdáním tohoto textu jeho autor místopřísežně prohlašuje, že celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1 Úvod

Vývoj počítačových her je komplexní proces, který kombinuje mnoho různých dovedností a disciplín. To je také důvodem, proč se tato práce zabývá návrhem a vývojem umělou inteligencí počítačové hry a ne vývojem kompletní hry.

Hlavní motivací této bakalářské práce je získání znalostí v oblasti umělé inteligence v počítačových hrách a na základně nich vytvořit demonstraci, kterou bude zábavné hrát. Demonstrace by měla být vytvořena pomocí nástrojů, které se standardně v herním průmyslu používají.

V dnešní době je umělá inteligence strojového učení často zmiňována v médiích. Tato práce se ovšem zabývá jiným druhem umělé inteligence. Místo strojového učení používá model Behavior Tree, který se používá pro řízení rozhodování agentů v počítačových hrách.

1.1 Cíl práce

Cílem práce je navrhnout a implementovat hratelnou demonstraci umělé inteligence souboje v reálném čase prostřednictvím technologie Unreal Engine. Demonstrace bude umožňovat souboj dvojice agentů ve vzorovém prostředí, první agent bude ovládán hráčem, druhý umělou inteligencí.

1.2 Členění práce

Práce je rozdělena do šesti části. V Teoretické části zmiňuji, jak vypadá proces vývoje hry, jaké technologie technologie se používají a jak jsem umělou inteligenci navrhnul. Třetí část je programátorská dokumentace, která popisuje fungování frameworku Unreal Engine, jakým způsobem je umělá inteligence implementována a jak je možné práci zkompilovat. Ve čtvrté část vysvětluji jak spustit hratelnou demonstraci, jak funguje ovládání a co znamenají jednotlivé UI elementy. Pátá část se zabývá výsledky uživatelského testování. Závěr je poslední část a shrnuje výsledek práce.

2 Teoretická část

2.1 Vývoj počítačových her

Vývoj počítačových her je komplexní proces, který se obecně skládá z několika fází. V této sekci každou krátce popíši. Fáze se mohou lišit zejména podle velikosti vyvíjené hry, ale obecně vývoj zahrnuje následující fáze.

2.1.1 Fáze návrhu

V této první fázi se vymýšlí, co bude hlavní náplní hry. To může zahrnovat věci jako jsou: příběh, postavy, herní mechanismy a umělecký styl. Dále se také plánuje, jaké softwarové nástroje bude potřeba vytvořit nebo licencovat. Všechna rozhodnutí se zapíší do dokumentu zvaného *Game Design Document*. Ten poté slouží jako základ, na kterém se hra staví. Game Design Document také často slouží ke komunikaci s investory a ostatními účastníky herního průmyslu - například vlastníky platformy, na kterých se hra bude vydávat, třeba Sony nebo Microsoft.

2.1.2 Fáze předprodukce

Fáze předprodukce slouží k vytvoření podrobného plánu projektu, který obsahuje více detailů. Dále se naprogramuje *prototyp hry*, které slouží k testování proveditelnosti projektu. Prototyp hry nemá finální audiovizuální prvky a všechny systémy se testují ve vzorových prostředích. Programátoři a herní návrháři testují, zda je prototyp zábavné hrát. Často se zjistí, že prototyp není proveditelný, nebo není dostatečně zábavný. Pokud se tak stane, může se změnit návrh, nebo ukončit vývoj.

2.1.3 Fáze produkce

Jedná se o nejdelší fázi vývoje. Vývojáři vědí jakou hru vytváří a postupně naplňují hru finálním obsahem. Prototyp je nahrazen hrou, která má lepší podrobně navrhnutou architekturu. Do ní poté návrháři levelů vkládají obsah - například 3D modely vytvořené umělci. Programátoři pracují na herních systémech a optimalizují hru tak, aby plynule běžela na všech podporovaných zařízeních. Herní návrháři systémy ladí a balancují. Na konci produkce se vytvoří uživatelské rozhraní UI a marketingové oddělení začne hru propagovat

2.1.4 Fáze testování

V této fázi jsou hlavní systémy funkční a hra prochází rozsáhlým testováním, které zjistí, zda je hra stabilní a neobsahuje chyby. Ty programátoři podle závažnosti opravují. Na konci testování je hra poskytnuta skutečným hráčům, kteří poskytují zpětnou vazbu.

2.1.5 Fáze vydání a podpora

Vydání hry zahrnuje vytvoření marketingových materiálů a spolupráci s vlastníky platforem, na kterých se hra bude prodávat. Podpora hry po vydání může být například oprava chyb, zlepšení výkonu nebo přidání nového obsahu.

2.2 Herní engine

Herní Engine je hlavním nástrojem vývoje. Je to framework, který pomáhá vývojářům efektivně vytvářet hry. Jeho použití může ale přesahovat do dalších odvětví jako je například filmový průmysl, kde simuluje virtuální prostředí, ve kterém herci hrají. Moderní herní frameworky zpravidla obsahují několik základních funkcionalit [1]:

- Vykreslování
- Ovládání
- Editor levelů
- Umělou inteligenci
- Animace postav
- Zvukové efekty
- Simulace fyziky
- Uživatelská rozhraní
- Hra pro více hráčů

Uvedená funkcionalita umožňuje vývojářům zaměřit se na hlavní obsah hry.

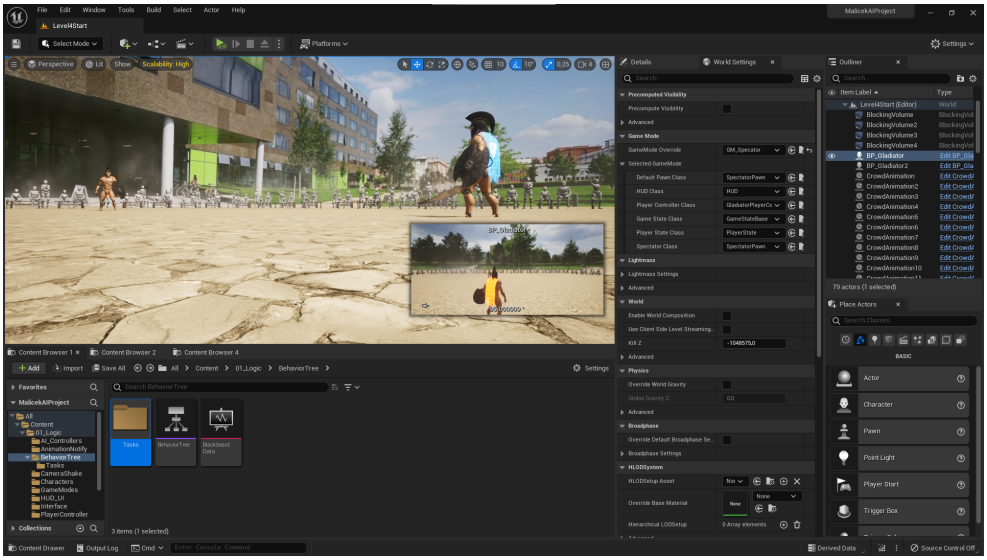
2.3 Použité technologie

Tato část se věnuje softwarovým nástrojům využitým pro vytvoření této práce.

2.3.1 Unreal Engine

V současné době se na trhu aktivně používají herní enginy od různých společností. Mezi nejznámější patří *Unreal Engine* vyvíjený Epic games [2]. Jedná se o platformu zaměřenou na 3D vykreslování v reálném čase. Unreal Engine umožňuje vývojářům vyvíjet obsah pro různé platformy, včetně PC, konzolí a mobilních zařízení.

Unreal Engine poskytuje širokou škálu nástrojů a funkcí specificky pro počítačové hry. Mezi hlavní nástroje patří sofistikovaný vizuální skriptovací systém Blueprints, který slouží jako abstraktní vrstva nad C++. Umožňuje vývojářům vytvářet nástroje a funkcionalitu přímo v editoru (obrázek č. 1) Unreal Engine bez nutnosti použití IDE nebo jiných externích nástrojů.



Obrázek 1: Unreal Engine 5 editor.

Historie

První verze Unreal Engine byla vytvořena v roce 1998 pod jménem Unreal Engine 1 [3]. V roce 1998 byla vydaná hra Unreal (viditelná na obrázku č. 2), která Unreal Engine 1 používala a měla velký kritický i komerční úspěch.

roce 2004 uvedla společnost EPIC hru Unreal Tournament 2004, který používal nový Unreal Engine 2. Ten disponoval novými funkcemi například skeletální animace, částicové vizuální efekty nebo pokročilá fyzika [3].

EPIC v roce 2006 představila Unreal Engine 3 [3], který přinesl funkce jako oblečení ovlivněné fyzikou, vylepšené realistické vykreslování materiálů, lepší osvětlení, vizuální efekty nebo podporu pro mobilní zařízení.

Rok 2015 přinesl velkou změnu byznys modelu společnosti EPIC, protože umožnila vývojářům používat Unreal Engine 4 bez nutnosti platit licenční poplatky do obratu 1 milion dolarů ročně [4].

Současnost

V roce 2022 byl vydán Unreal Engine 5, který představuje nejnovější generaci herního enginu. Poskytuje vývojářům velkou sadu nástrojů, kterou lze využít nejen pro vývoj her, ale také pro vizualizaci architektury, tvorbu filmů nebo živého vysílání televizního studia. Unreal Engine 5 přinesl novou funkcionalitu v oblasti vykreslování, dynamického osvětlení (Lze vidět na obrázku č.3) a přidal nástroje pro tvorbu 3D. V této práci používám Unreal Engine 5 ve verzi 5.0.3.

Obrázek 2: Screenshot ze hry Unreal, která využívá Unreal Engine 1 [5].

Obrázek 3: Screenshot ze hry, která využívá Unreal Engine 5 [6].

2.3.2 C++ a Blueprints Visual Scripting

Celý engine je postaven na principu kombinace výkonného nízkoúrovňového jazyka C++ a flexibilního vizuálního skriptovacího jazyka *Blueprints*. Blueprints umožňuje abstraktně pracovat s C++ kódem bez nutnosti hluboké znalosti C++. Díky této abstrakci lze zpřístupnit složité C++ systémy a funkcionality herním návrhářům a 3D umělcům.

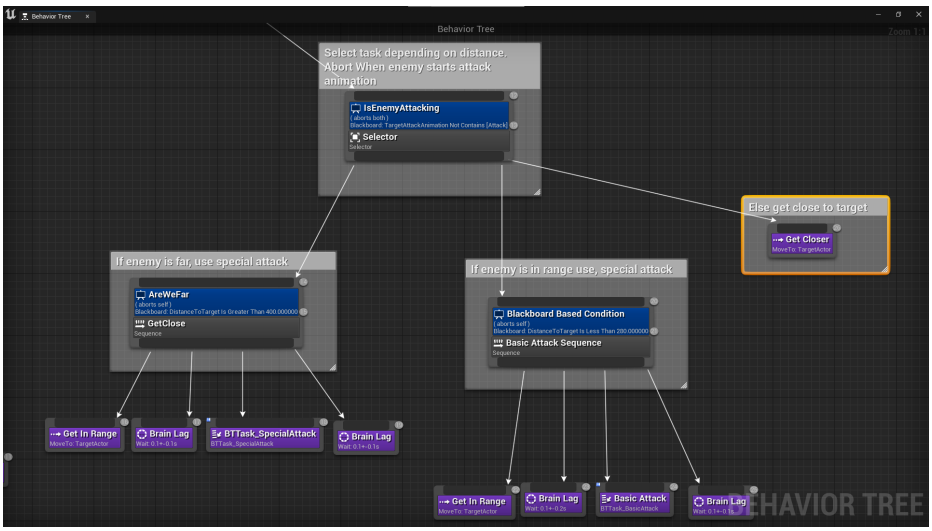
Blueprints umožňuje vývojářům vytvářet složité herní systémy a nástroje bez nutnosti tradičního programování. Používá rozhraní založené na uzlech, které znázorňují tok logiky. Uzly tvoří struktury, které se podobají vývojovým diagramům. Blueprints lze také použít pro vytváření různých editorů a nástrojů přímo v Unreal Engine. Například editor questů nebo editor dialogů postav ve hře.

2.3.3 Unreal Engine - Behavior Trees

Systém Behavior Trees (obrázek č. 4) umožňuje vytvářet logiku rozhodování umělé inteligence. Rozhoduje o tom, jakou větev stromu bude umělá inteligence provádět a za jakých podmínek.

Rozhodování se opírá o real-time data, které umožňují umělé inteligenci dynamicky reagovat, na to co se právě ve světě hry děje. Logika toho, co má umělá inteligence dělat, je mimo strom v podobě Blueprint skriptů. Tento systém je detailněji popsán v sekci 3.4.4 “Popis systému Behavior Tree” .

V této práci je systém behavior tree použitý pro řízení pokročilé umělé inteligence, která ovládá stejnou třídu `BP_Gladiator` jako ovládá hráč.

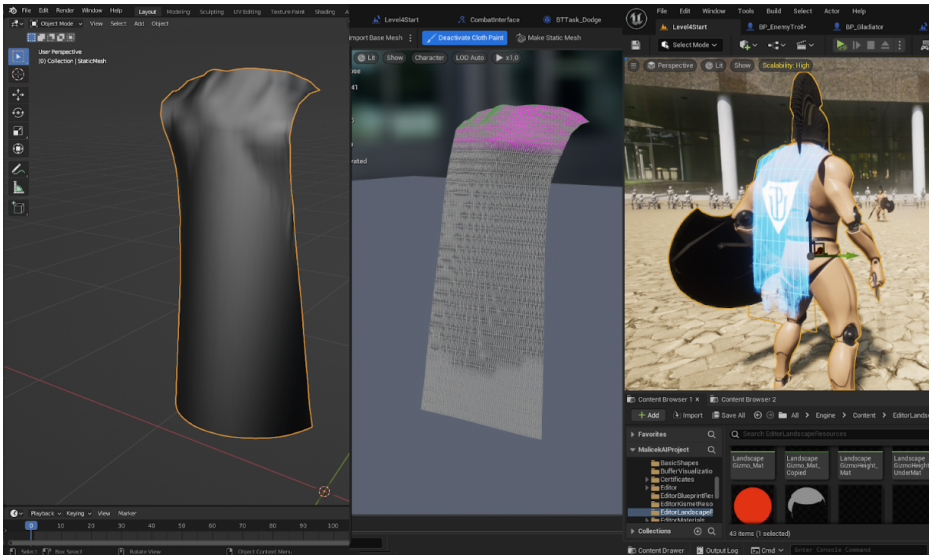


Obrázek 4: Větev systému Behavior Tree, který řídí chování třídy **BP_Troll**.

2.3.4 Blender

Blender je bezplatný a svobodný software pro 3D tvorbu. Je používán k vytváření vizuálního obsahu jako jsou 3D modely, animace nebo vizuální efekty. Obsahuje mnoho nástrojů jako například 3D modelování, texturování, simulace fyziky, animace nebo vykreslování. Podporuje různé formáty souborů včetně FBX, který je standardem pro herní průmysl.

V této práci jsem Blender použil pro vytvoření 3D modelu pláště (obrázek č. 5). Plášti jsem poté v Unreal Engine nastavil simulaci fyziky a tak může mít umělá inteligence dynamické oblečení, které se přizpůsobuje animacím postavy.



Obrázek 5: 3D model pláště, který se dynamicky přizpůsobuje animacím.

2.3.5 Gimp

Gimp (GNU Image Manipulation Program) je bezplatný grafický editor pro tvorbu a úpravu 2D bitmap. Poskytuje mnoho nástrojů a funkcí pro 2D bitmapy. V herním průmyslu se používá pro práci s 2D texturami. Podporuje velké množství formátů, jako jsou například JPEG, PNG, GIF, dokonce i nativní Adobe Photoshop formát PSD. V této práci jsem využil Gimp pro vytvoření textury pláště s logem Univerzity Palackého.

2.3.6 Audacity

Audacity je bezplatný software pro úpravu zvuku s otevřeným zdrojovým kódem. Je široce používán hudebníky a audio profesionály pro nahrávání a editaci zvukových stop. Poskytuje mnoho funkcí pro nahrávání, úpravy a editaci zvuku včetně vícestopého nahrávání, redukce šumu a tvoření zvukových efektů. V této práci jsem Audacity použil na editaci zvukových efektů souboje.

2.4 Návrh hry

Účelem návrhu hry je zprostředkování poutavého a zábavného zážitku pro hráče. *Návrh hry* zahrnuje vytvoření pravidel hry, celkovou strukturu hry a všech systémů v ní. Herní návrhář během vývoje testuje a upravuje herní systémy tak, aby byl herní zážitek co nejpoutavější. Programátoři během vývoje zpřístupňují proměnné ovlivňující chování hry a ty herní návrhář ladí a testuje. Dobře navržená hra by měla hráči poskytnout pocit uspokojení a úspěchu.

2.4.1 Pravidla a prvky Souboje

Systém životů

Každý agent v souboji má daný počet životů, pokud je agent zasažen, sníží se mu počet životů o 1. Vítězem se stává agent, který připraví protivníka o všechny životy. Počet životů je zobrazen pomocí UI elementu (obrázek č. 6).

Obrázek 6: UI element, který zobrazuje zbývající počet životů.

Základní útok (Basic Attack)

Základní útok je elementární prvek mnoha akčních her. Hráč bude tuto akci často opakovat, proto jsem vynaložil velké úsilí na to, aby provádění této akce bylo intuitivní a plynulé. Pro pocit responzivity se animace seknutí provede ihned po stisknutí tlačítka myši. To je rozdíl oproti tradiční principům animace, které chtějí zobrazit sílu útoku tím, že na začátek animace vloží snímky, kdy se postava natahuje ruku. To ovšem přispívá k nereponzivitě hry a proto by postavy ve hrách měli přejít do animace seknutí co nejrychleji po stisknutí tlačítka.

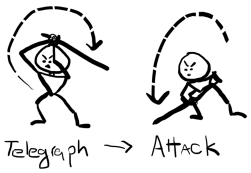
Pro lepší pocit ze hry byl implementován tzv. *combo systém*, který spouští animace útoku (obrázek č. 7) podle toho, jaké animace už byly přehrány. Díky tomuto systému se předejde trhání animací a neustálého přehrávání jedné a té samé animace. Agent si pamatuje jakou animaci přehrával naposled a podle toho vybere další.

Obrázek 7: **BP_Gladiator** v animaci základního útoku.

Speciální útok (Special Attack)

Speciální útok je navržený jako nástroj pro překonání větší vzdálenosti mezi agentem a jeho cílem. První část útoku využívá princip animace zvaný "očekávání", který signalizuje, že se agent chystá zaútočit speciálním útokem. Díky signalizování má protivník čas zareagovat a například uskočit. Signalizování se používá v akčních hrách, kde je kladen důraz na reflexy hráče - například ve hře Dark Souls [7].

Silné útoky by měly být silně signalizovány (znázorněno na obrázku č. 8 a 9), aby byl zachován pocit fair play. Pokud by byl útok špatně signalizován, hráči by se mohlo zdát, že je hra nespravedlivá [8]. Proto jsem přidal vizuální efekt blesku, který vizuálně signalizuje, že se agent chystá ke speciálnímu útoku.



Obrázek 8: Znázornění fází animace útoku [8].

Obrázek 9: Signalizování speciálního útoku.

Úhyb a nezranitelnost (Dodge)

Úhyb agentovi umožňuje stát se nezranitelný a nulovat tak přímý zásah od nepřítele (znázorněno na obrázku č. 10). Tento herní mechanizmus je velmi silný, proto jsem limitoval, jak často ho může agent používat. Existuje několik způsobů, jak používání limitovat. Mezi dva nejčastější patří limitování počtu použití a limitování pomocí časovače. V této práci jsem zvolil limitaci pomocí časovače, který se zobrazuje jako UI element.

Obrázek 10: Animace úhybu.

Časovače (Cooldowns)

Tato herní mechanika zabraňuje opakovanému používaní jedné akce. Akce má časovač, který se spustí v moment, když se daná akce začne provádět. Po dobu kdy je časovač aktivní, nemůže agent danou akci provést. Tím jak je dlouhý, se reguluje, jak často se akce mohou spouštět. Silnými akcemi se časovač zvýší, aby se hra vybalancovala a přinutila hráče používat i ostatní akce. Časovač musí být hráči vykomunikován (obrázek č. 11), jinak by se mohla hra zdát neresponzivní.

Obrázek 11: Zobrazení časovačů schopností.

2.4.2 Pocitová kvalita hry (Game Feel)

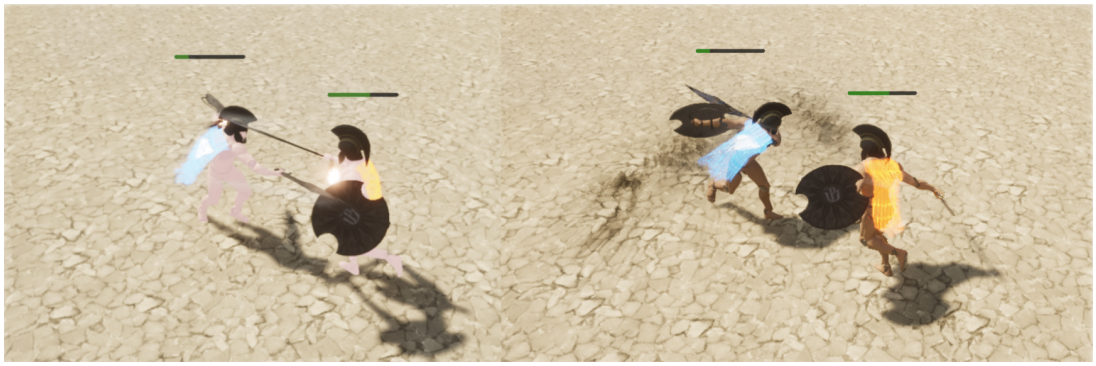
Tento pojem pokrývá velké množství herních prvků, díky kterým je hraní hry příjemnější a poutavější. Jedná se o kombinaci vizuálních, zvukových a UI efektů, které zlepšují zpětnou vazbu mezi hrou a hráčem. Pokud by prvky chyběly, hráč by cítil, že některé části hry nefungují, jak by čekal.

Chvění kamery (Screen Shake)

Když chceme ukázat sílu účinku události ve hře, můžeme trochu zachvět s kamerou a hráč dostane taktilní pocit toho, že se něco stalo. V této práci používám screen shake na mnoha místech. Například zásah nepřítele, vyvolá slabé chvění kamery a animace chůze trolla vyvolá silnější chvění.

Vizuální efekt zásahu

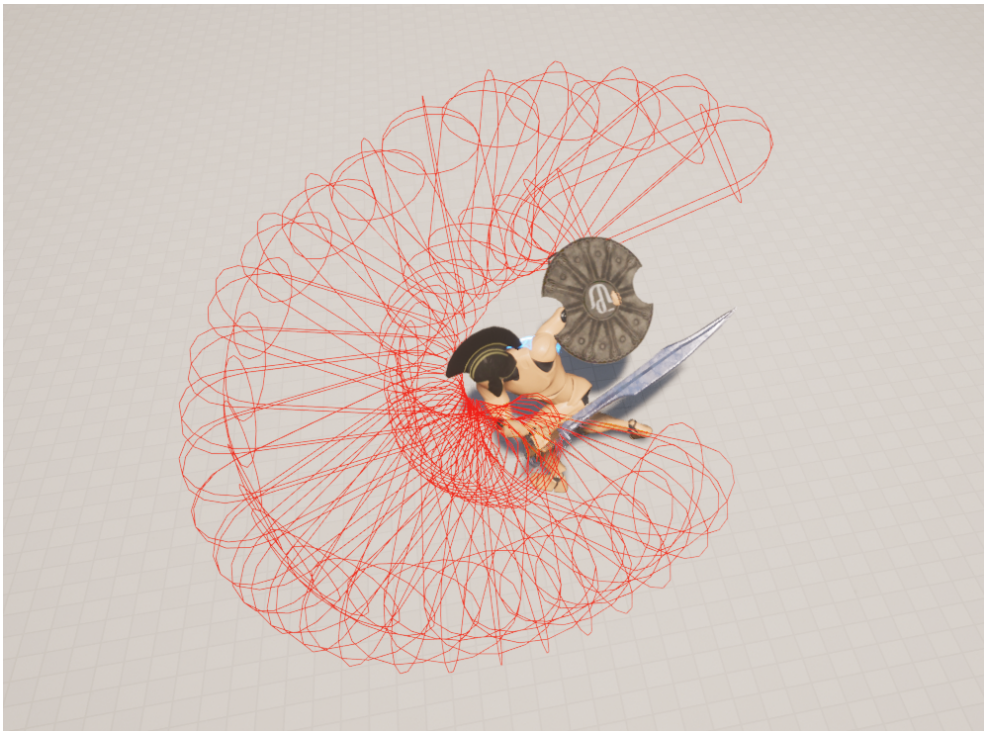
Když je agent zasažen, jeho tělo "problikne" (obrázek č. 12) a spustí se vizuální efekt zásahu (v této práci černé krve, která má reprezentovat olej robota). Je důležité dát hráči zpětnou vazbu, aby věděl, že hra správně registruje jeho akce.



Obrázek 12: Vizuální efekty zasáhnutí.

Dynamický hitbox útoku

Hitbox slouží v počítačových hrách jako nástroj, který vypočítává kolize objektů. Poskytuje informace o tom, jaké objekty s Hitbox kolidují. Dynamický hitbox je spárován s modelem meče a následuje jeho pozici v herním světě (obrázek č. 22). To umožňuje hře simulovat přesné kolize zásahu. Informace o kolizích lze využít pro spouštění herní logiky a audiovizuálních efektů zásahu. Pro lepší pocit ze hry je hitbox trochu větší než reálná velikost meče. Dynamické kolize poskytují hráči pocit fair play a předchází mylným zásahům.



Obrázek 13: Vizuálně znázorněná kolize meče.

Vizuální efekt základního útoku

Pro lepší pocitovou kvalitu útoku se přidá vizuální efekt, který zvýrazní trajektorii meče (obrázek č. 14). Využívá vizuální částicový systém, který Unreal Engine poskytuje. Efekt se zapne na začátku animace seknutí a vypne na konci.

Obrázek 14: Vizuální efekt při animaci seknutí.

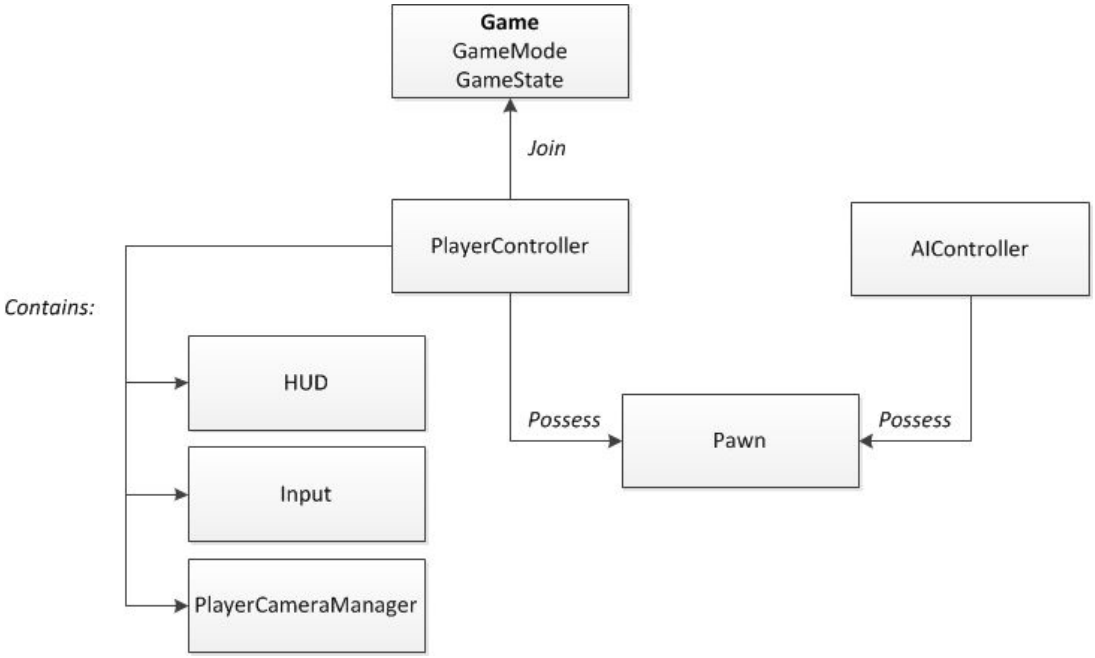
Simulace mozkového zpoždění (Brain Lag)

K mozkovému zpoždění dochází mezi okamžikem, kdy mozek přijímá signál a okamžikem, kdy je signál zpracován a interpretován. Pokud například hráč vidí, jak umělá inteligence začíná animaci speciálního útoku, pár desítek milisekund potrvá než si to uvědomí a rozhodne, jak bude reagovat. Umělá inteligence žádné takové zpoždění nemá, proto pokud uvidí, že hráč začal animaci útoku, okamžitě může reagovat. Z tohoto důvodu jsem implementoval simulaci mozkového zpoždění. Z tohoto důvodu jsem implementoval simulaci mozkového zpoždění, která umělou inteligenci "zdržuje" a znatelně tak zlepšuje pocit ze souboje.

3 Programátorská dokumentace

3.1 Unreal Engine Gameplay Framework

Unreal Engine Gameplay Framework (obrázek č. 15) je soubor tříd a komponent, které spolu komunikují a vytváří základ pro hru. Třída `PlayerController` reprezentuje hráče a je připojená k instanci hry. Třída `PlayerController` umožňuji hráči ovládat třídu `Pawn`, která reprezentuje fyzickou postavu ve světě hry. Třída `PlayerController` dále zprostředkovává input od hráče, ovládá kameru a zobrazuje HUD (Heads-up display, který spravuje UI elementy). Třída `AIController` umožňuje umělé inteligenci ovládat třídu `Pawn`.



Obrázek 15: Unreal Engine Gameplay Framework diagram [9].

3.1.1 Třída Game Mode

Koncept "hry" Unreal Engine rozděluje do dvou tříd `GameMode` a `GameState`.

Třída `GameMode` má na starosti pravidla hry, například za jakou třídu hráč bude hrát, jaká třída bude řešit hráčům input, maximální počet hráčů, jak hráč vstupuje do hry, výběr místa začátku hry nebo zda může hráč hru zastavit či nikoli [10].

Třída `GameState` sleduje události a ukládá o nich informace, které synchronizuje se všemi účastníky hry - například: jak dlouho hra běží, aktuální režim hry, kdy se kdo připojil, jestli hra začala nebo jestli se na někoho čeká [10].

V této práci má `GameMode` na starosti dvě věci, a to za jakou třídu hraje hráč a jaká třída zpracovává jeho input.

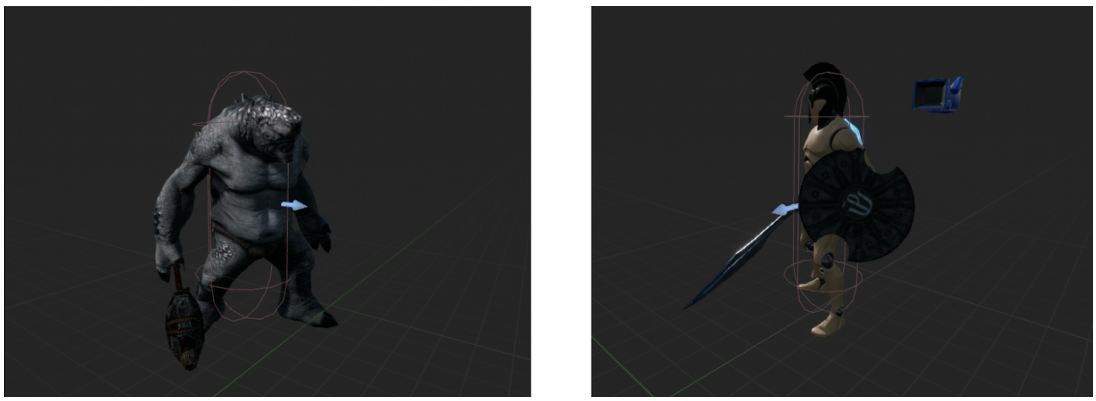
Implementovány jsou dvě třídy `GameMode`, jedna pro aktivní hraní hráč vs AI, druhá pro sledování dvojce AI agentů. Třída `GameState` není implementována, ale například by v ní mohlo být uložené kolikrát vyhrál hráč a kolikrát umělá inteligence.

3.1.2 Třída Pawn

Třída `Pawn` je agentem ve světě hry. Třída `Pawn` je základní třída pro všechny třídy, které mohou být pod kontrolou třídy `Controller`. Je schopná přijímat a vykonávat pokyny třídy `Controller`. V této práci je příkladem třídy `Pawn` třída `SpectatorPawn`, skrze kterou hráč sleduje souboj dvou agentů v levelu AI vs AI.

3.1.3 Třída Character

Třída `Character` je speciálně rozšířená třída `Pawn`. Třída `Character` by měl reprezentovat modely humanoidního stylu (vertikálně orientované). Ve výchozím nastavení má `Character` následující komponenty: `CapsuleComponent` pro kolize, komponent `CharacterMovementComponent` pro základní pohyb na mapě a `SkeletalMeshComponent` pro skeletální modely a animace. V této práci jsou dvě třídy `Character` a to `BP_Gladiator` a `BP_Troll` (zobrazeno na obrázku 16).



Obrázek 16: Třídy Character **BP_Troll** a **BP_Gladiator** v editoru.

3.1.4 Třída Controller

V kontextu hráče a umělé inteligence je třída `Controller` braná jako mozek, který ovládá fyzické tělo ve hře. Třída `Controller` je třída, která může vlastnit a řídit třídu `Pawn` a `Character`. Třída `PlayerController` k ovládání používá input od hráče, zatímco `AIController` využívá umělou inteligenci [11]. `Controller` dostává oznámení o událostech, ke kterým během hry dochází. To umožňuje třídě `Controller` na události reagovat a dynamicky se přizpůsobit průběhu hry.

3.1.5 Třída Player Controller

Třída `PlayerController` je rozhraním mezi hráčem a třídou `Pawn`, kterou ve hře ovládá. Implementuje funkcionalitu pro získávání inputu od hráče [11]. V této práci má třída `PlayerController` na starosti řízení třídy `BP_Gladiator`, zobrazování a aktualizování UI elementů, zobrazování menu nabídky a pohyb kamerou. Seznam akcí, které může hráč provádět:

- Základní útok
- Speciální útok
- Vyhnutí
- Main Menu
- Pohyb postavy
- Pohyb kamery

3.1.6 AI Controller

Cílem třídy `AIController` je pozorovat svět kolem sebe, rozhodovat se a dávat rozkazy třídě `Pawn`, kterou ovládá. Zatímco třída `PlayerController` se při rozhodování spoléhá na hráče, `AIController` reaguje na vstupy a eventy ze světa hry.

V této práci jsou 2 hlavní implementace třídy `AIController` . Obě mají různé přístupy k tomu jakým způsobem řídí třídu `Pawn`. Třída `BP_Troll` je řízena pomocí while smyčky. Třída `BP_Gladiator` je řízena pomocí Systému Behavior Tree.

3.2 Základní umělá inteligence (Trol)

Hlavní myšlenkou umělé inteligence Trol je propojení systémů, které Unreal Engine nabízí a vytvořit zábavný souboj hráče proti jednoduchému nepříteli. Model trola jsem vybral kvůli tomu, že se ve fantasy literatuře považuje za ne moc inteligentní stvoření.

Snažil jsem se zachytit, jak by mohl vypadat souboj mezi člověkem a trollem pomocí systému jako: chvění kamery, audio efekty, vizuální částicové efekty, skeletální animace a dynamické materiály.

3.2.1 Profesionální 3D model a animce

Model trola (obrázek č. 17) a jeho animace byly použity ve hře Infinity Blades. Mohl jsem tedy pracovat s modely a animacemi z reálné produkce. 3D model a animace byly společností EPIC games zdarma zpřístupněn všem Unreal Engine vývojářům [12].

Obrázek 17: **BP_Troll** ve hře (lze vidět vizuální efekt chůze).

3.2.2 Hlavní komponenty třídy BP_Troll

Třída `BP_Troll` je instancí třídy `Character` , kterou ovládá umělá inteligence. Obsahuje herní logiku jako jsou životy, pohyb po mapě, schopnost útočit, nebo zemřít. Je řízená pomocí třídy `AIController` nazvanou `Troll_AI_Controller` , která rozhoduje, jakou činnost bude třída `BP_Troll` provádět.

Kolize (CollisionCylinder)

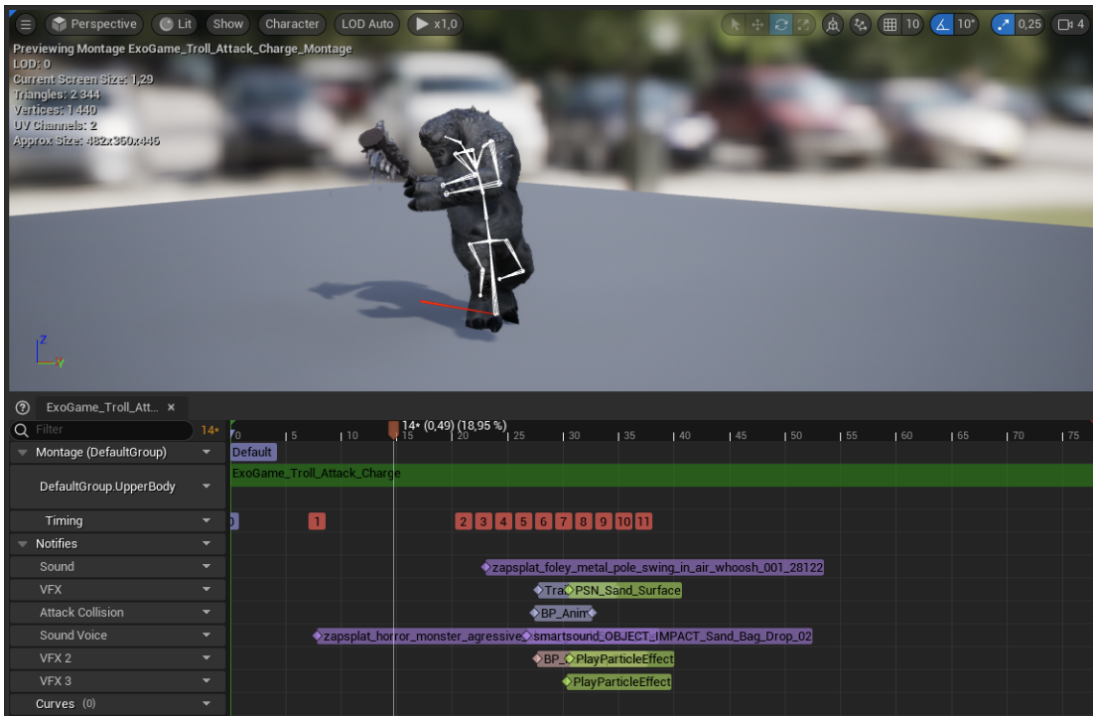
Komponent, který se stará o kolizi pohybu. Aby bylo možné rychle vypočítávat komplikované výpočty pro pohyb, zjednodušíme prostor, který třída v herním světě fyzicky zabírá na geometrický tvar válce.

UI element (Health Bar)

Komponent, který zobrazuje počet zbývajících životů. O zobrazení a aktualizaci komponentu se stará třída `BP_Troll`.

Skeletální model a animace

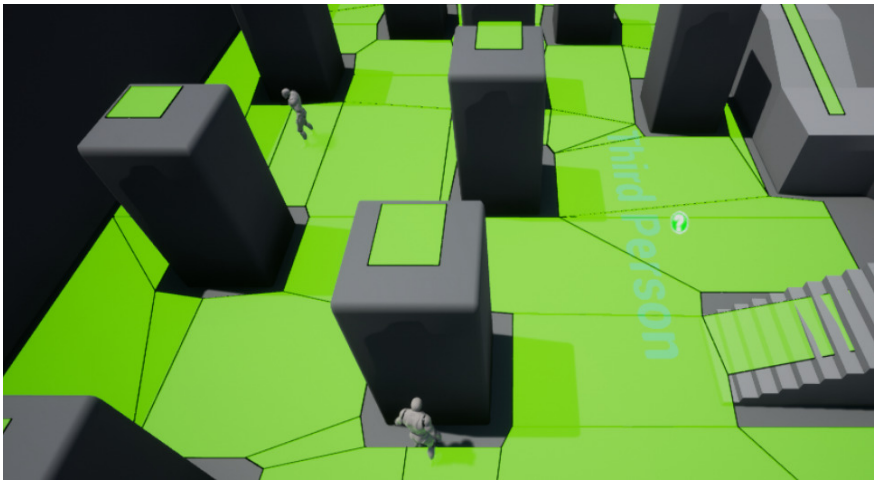
Hlavním komponentem třídy `BP_Troll` je skeletální model a jeho animace. Unreal Engine poskytuje systém animačních notifikací, který umožňuje v určitých bodech animace spouštět eventy (obrázek č. 1). Na eventy můžeme navázat herní logiku nebo audiovizuální prvky. Například každé šlápnutí v animaci chůze vyvolá chvění kamery, spustí zvuk došlápnutí a na místě došlápnutí se objeví vizuální efekt prachu. Kombinování animací a audiovizuálních efektů výrazně zlepšuje pocit ze hry.



Obrázek 18: Editor animačních eventů a notifikací.

Pohyb a NavMesh (CharacterMoveComp)

Komponenta pohybu poskytuje zapouzdření pohybového systému pro humanoidní postavy. Systém umožňuje hledat nejkratší cestu mezi bodem A a B. Aby ji bylo možné najít, musíme do levelu vložit NavMesh (znázorněno na obrázku 19). NavMesh je automaticky vytvořená mapa plochy, po které se může AI pohybovat [13].



Obrázek 19: Vizualizace NavMesh, po které se AI pohybuje [14].

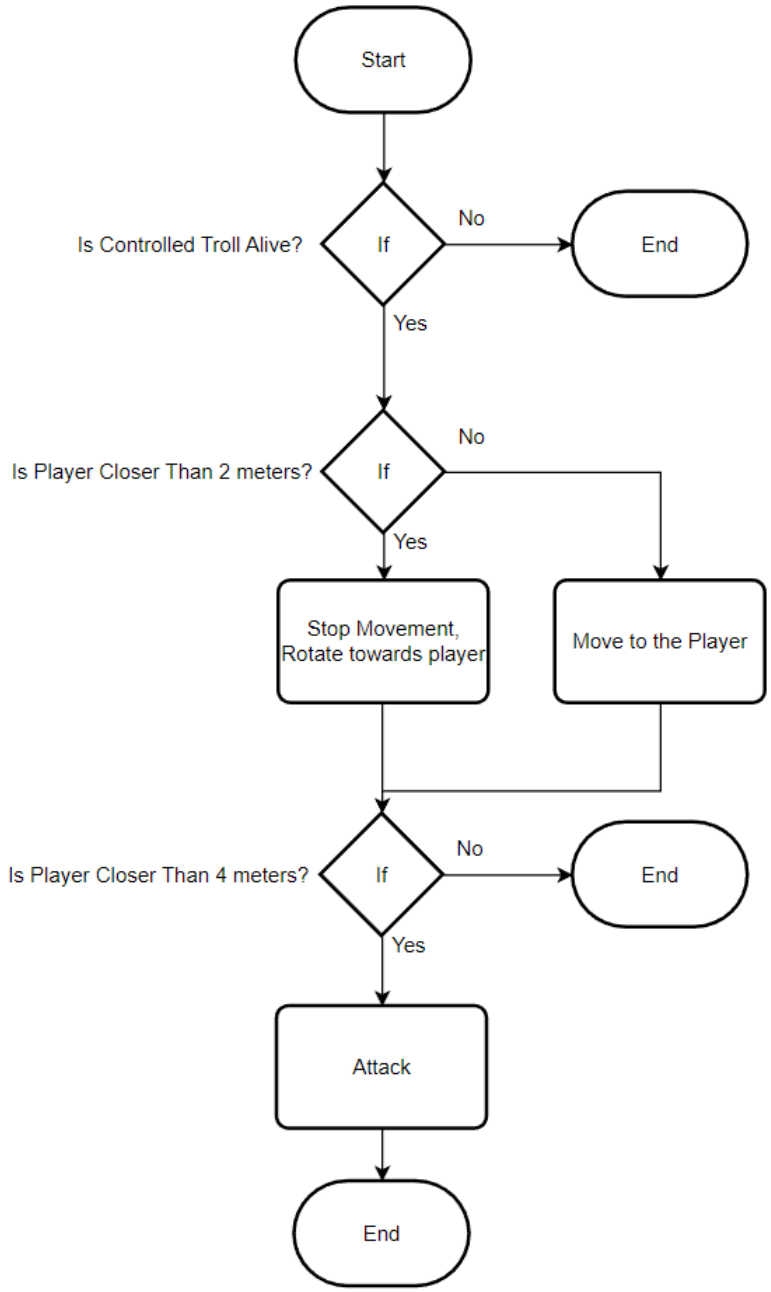
3.2.3 Hlavní logika trola

V této části jsou popsány hlavní funkce a eventy třídy `BP_Troll`. Třída `AIController` funkce následně volá.

- Combat Interface - Interface pro komunikaci mezi agenty souboje.
  - **GetHit** - Obsluha zásahu agenta, funkce nic nevrací.
  - **GetHealth** - Funkce vrací aktuální počet životů.
  - **GetMaxHealth** - Funkce vrací maximální počet životů.
  - **Death** - Obsluha smrti agenta, funkce nic nevrací.
- **Event Begin Play** - Event, který nastaví životy a inicializuje UI komponenty.
- **Event Attack** - Event, který obsluhuje logiku útoku trolla. Vybírá a přehrává animace.
- **Event get Hit** - Event, který se spustí, když je třída zasažena. Hlídá, jestli má více než 1 život, pokud ne, spustí event Death. Spouští audiovizuální efekty související se zásahem.
- **Event Death** - Event, který obsluhuje logiku úmrtí. Zpomalí čas a začne přehrávat animaci úmrtí. Po dokončení animace restartuje level.

3.2.4 Popis rozhodování základní umělé inteligence

Rozhodování základní umělé inteligence je popsáno na diagramu č.20 a děje se uvnitř třídy `Troll_AI_Controller`. Logika rozhodování je implementovaná v Event Tick, který se vykoná každý snímek hry. Myšlenka rozhodování je vcelku jednoduchá a to každý snímek hry proveď toto: *Dokud jsi naživu, pronásleduj hráče. Pokud jsi v dosahu útoku, zaútoč na něj.*



Obrázek 20: Vývojový diagram rozhodování třídy **BP_Troll**.

3.3 Pokročilá umělá inteligence (Gladiátor)

Cílem této umělé inteligence je schopnost ovládat stejnou třídu (obrázek č. 21) jako ovládá člověk. Souboj s ní by měl být zábavný, dynamický a měl by reagovat na to, co hráč dělá. Z tohoto důvodu používám technologii Behavior Tree, která umožňuje umělé inteligenci reagovat na vstupy z herního světa a rychle změnit své chování. Umělá inteligence neustále sleduje, co hráč dělá a kde se nachází. Podle těchto údajů se rozhoduje a volí chování. Například umělá inteligence chce provést útok, ale hráč začne přehrávat animaci útoku, a tak umělá inteligence změní své chování a místo útoku provede úhyb.

Obrázek 21: **BP_Gladiator** ovládaný pokročilou umělou inteligencí.

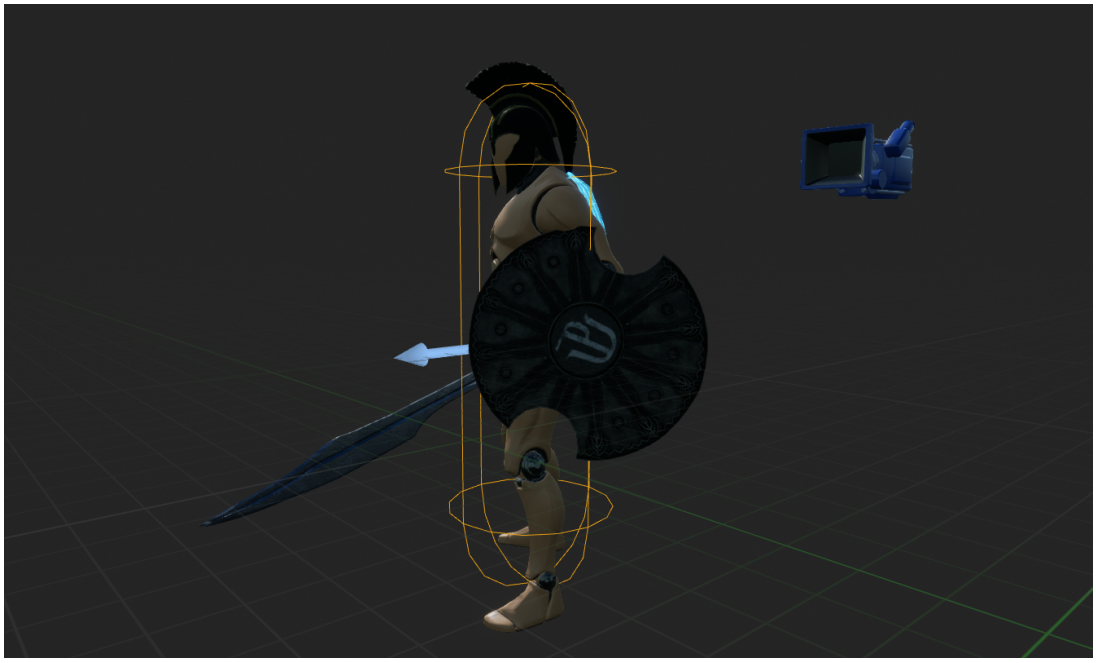
3.3.1 Třída BP_Gladiator a její hlavní komponenty

Třída reprezentující fyzickou podobu gladiátora v herním světě. Může být ovládána jak umělou inteligencí, tak hráčem. Je instancí třídy `Character` a získává tak její komponenty a vlastnosti. Obsahuje herní logiku a audiovizuální prvky. Herní logika určuje, jaké akce může gladiátor provádět, jakým způsobem se bude gladiátor pohybovat a jak bude komunikovat s ostatními agenty souboje. Audiovizuální prvky jsou například 3D modely a jejich animace, oblečení simulující fyziku (kápě), vizuální efekty (krev při zásahu), zvukové efekty kroků a sekání.

Protože může být ovládaná hráčem, je kladen důraz na to, aby její ovládání bylo pro hráče příjemné a intuitivní. Musí tak být ošetřené jevy jako jsou netrhaný pohyb kamery, přesné kolize meče nebo věrohodné animace pohybů.

Kolize (Capsule Component)

Komponent (obrázek č. 22), který se stará o kolize pohybu. Usnadňuje navigační výpočty.



Obrázek 22: Vizualizace kolize.

UI element (HealthBar)

Komponent (obrázek č. 23), který signalizuje, kolik agentovi zbývá životů.

Obrázek 23: Komponent, který zobrazuje životy třídy.

Systém kamery (Spring Arm, Camera)

Kamerový systém se v Unreal Engine skládá ze dvou spolupracujících komponent Spring Arm a Kamera (obrázek č. 24). Komponenta Kamery reprezentuje pohled hráče, používá se k výpočtu pozice, ze které hráč pohlíží na svět. Kameře můžeme nastavit vlastnosti, jako je například velikost zorného pole, typ promítání (Perspektivní, Orografické), nebo Post Processing. Spring Arm je komponent, který hráči pomáhá s ovládáním kamery. Snaží se odstranit trhavé pohyby kamerou pomocí lineární interpolace a přechází kolizím kamery a herního světa.

Obrázek 24: Komponent kamery, kterou může hráč ovládat.

Skeletální 3D model

Skeletální 3D model je speciální typ 3D modelu, který se používá v počítačových hrách nebo animacích. Oproti klasickému 3D modelu má model kostru, která umožňuje s 3D modelem pohybovat.

Kostra je hierarchická stromová struktura tvořena kostmi. Každá kost ovládá svým pohybem jinou část modelu. Animátor může kostmi pohybovat a tím animovat 3D model. Sekvence pohybů jsou zaznamenány v souboru, který se nazývá Animation Sequence.

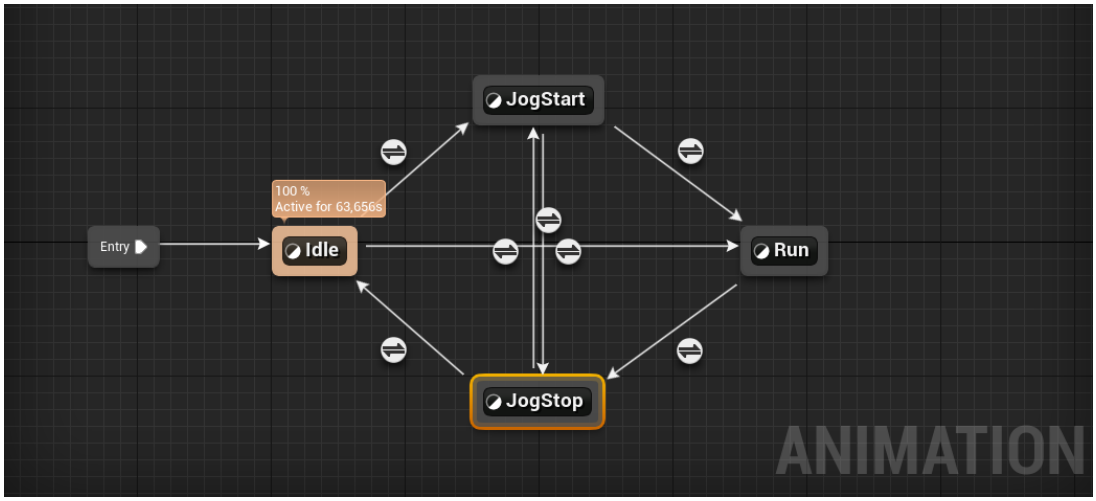
Pro lepší pocit ze hry nám Unreal Engine umožňuje animace dynamicky kombinovat na základě toho, co postava právě dělá. Unreal Engine poskytuje robustní animační systém Animation Blueprint.

Třída Animation Blueprint

Specializovaná třída skriptovacího jazyka Blueprint, který řídí animaci skeletálního modelu během hraní hry. To jak vypadá finální kombinace animací řídí graf, který provádí každý snímek hry výpočty a podle výsledků prolíná animace. Skeletální model tak může dynamicky reagovat na to, co se právě ve hře odehrává.

Animační stavový diagram (State Machine)

Stavový diagram je modulární systém, který nastavuje pravidla, podle kterých se přehrávají animace. Primárně se tento typ systému používá k řízení animací pohybu postav jako je chůze, běh a skákání. Pomocí stavového diagramu můžeme vytvářet stavy a definovat animace [15]. Poté zavedeme pravidla přechodů (obrázek č. 25), která řídí, kdy se má přejít do jiného stavu. Tento systém nám pomáhá prolínat animace a zlepšovat tak vizuální stránku hry. V této práci je stavový diagram použitý pro prolínání animací mezi stáním a během.



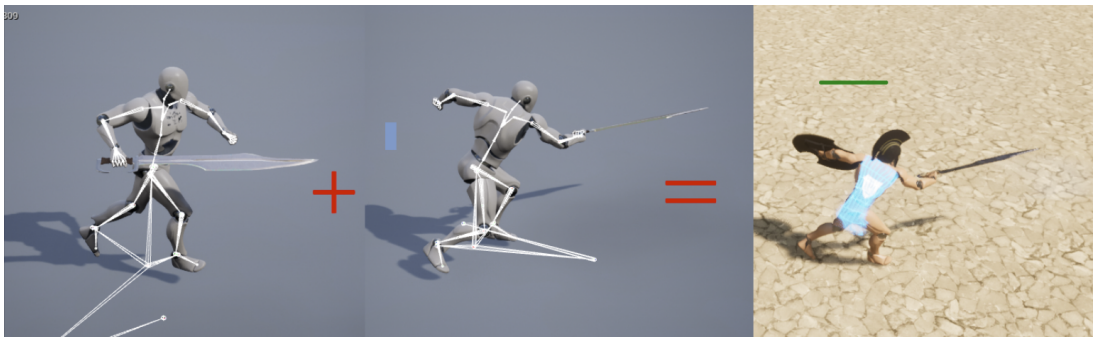
Obrázek 25: Stavový diagram, který řídí animace nohou.

Animační montaž (Animation Montage)

Animační montáž je flexibilní nástroj, který umožňuje kombinovat a selektivně přehrávat animace, a to i z tříd, které nejsou Animation Blueprint [16]. Slouží proto k řízení animačního systému "z venku". Například třída `BP_Gladiator` může poslat zprávu animačnímu systému, aby začal přehrávat animaci útoky. Dále umožňuje navázat herní eventy na konkrétní část animace. V této práci jsou animační montáže používány pro spouštění časovače útoku.

Animační sloty (Animation Blueprint)

Animační slot je nástroj, který umožňuje rozdělit skeletální model na vrstvy a každé vrstvě přidělit jinou animaci. Tím lze kombinovat různé animace pro různé části těla a vytvořit tak zcela novou animaci (obrázek č. 26). V této práci animační sloty animují horní a spodní část těla nezávisle na sobě, to umožňuje agentovi chodit a útočit najednou.



Obrázek 26: Kombinace dvou animací do jedné výsledné.

Aditivní animace (Additive Animation)

Unreal Engine umožňuj sčítat pohyby animací a vytvořit tak dynamické animace. K hlavní animaci se přidá vedlejší, která mění to, jak hlavní animace vypadá. Například animace pohybu a zásahu se sečtou a vytvoří animaci zásahu v pohybu. V této práci jsou aditivní animace využity k animování zásahu agenta (obrázek č. 27).

Obrázek 27: Dynamická animace zásahu.

3.3.2 Hlavní funkcionalita gladiátora

Tato část popisuje stěžejní eventy a funkce, které zajišťují správné fungování třídy `BP_Gladiator`. Hráč i umělá inteligence může eventy spouštět. Pohyb je zajištěn pomocí komponentu `CharacterMovement` . Třída `BP_Gladiator` komunikuje s ostatními agenty souboje pomocí Combat Interface, kterým si posílají zprávy.

- Combat Interface - Interface pro komunikaci mezi agenty souboje.
  - **GetHit** - Obsluha zásahu agenta, funkce nic nevrací.
  - **GetHealth** - Funkce vrací aktuální počet životů.
  - **GetMaxHealth** - Funkce vrací maximální počet životů.
  - **Death** - Obsluha smrti agenta, funkce nic nevrací.
- **Event BeginPlay** - Inicializuje animační systém a UI element, který zobrazuje životy agenta.
- **Event Basic Attack** - Řídí logiku základního útoku. Vybírá a spouští animace základního útoku tak, aby na sebe navazovali.
- **Event Special Attack** - Řídí logiku speciálního útoku, spravuje časovač, který umožňuje speciální útok aktivovat. Dále spouští animace a audiovizuální efekty speciálního útoku.
- **Event Sword Hit** - Aktivuje se při zásahu nepřítele. Při zásahu začne přehrávat chvění kamery a na pár snímků zastaví animaci. To simuluje sílu nárazu meče a zlepšuje tím pocit ze hry.
- **Event Dodge** - Řídí logiku úhybu. Vybírá spouštění animací, pohybuje agentem, zapínání a vypínání nezranitelnosti, startuje časovače vyhnutí.
- **Event Get Hit** - Obsluhuje událost, kdy byl agent zasažen nepřítelem. Má na starosti prevenci dvojitého zasáhnutí (kdy jedna animace zasáhne dvakrát), počítání zbývajících životů a přehrání audiovizuálních prvků spojených se zásahem (audio, vizuální částicové efekty, probliknutí skeletálního modelu).
- **Event Death** - Obsluhuje událost, kdy agentovi klesne počet životů na 0. Zpomalí čas, přehraje zvukový efekt, zastaví pohyb těla a spustí na něm fyzickou simulaci. Po sekvenci audiovizuálních prvků restartuje level a hra začne znovu.
- **Event Dispatchers** - Agent může pomocí těchto eventů komunikovat s ostatními Třídami a instancemi ve hře. Třída `BP_Gladiator` používá event dispatchers pro komunikaci s `PlayerController` . Komunikuje, v jakém stavu jsou časovače útoku, speciálního útoku a úhybu.

3.4 Rozhodování pokročilé umělé inteligence

Cílem pokročilé umělé inteligence v této práci je poskytnout hráči dobrý zážitek ze hry tím, že hráč bude bojovat proti stejné postavě jaka je ta, kterou ovládá. K tomu je potřeba, aby umělá inteligence byla schopna agenta ovládat podobným způsobem jako hráč. Měla by dělat dělat vlastní rozhodnutí, které dynamicky reagují na to co se ve hře děje. Unreal Engine poskytuje systém Behavior Tree, který dané požadavky splňuje (obrázek č. 28). Systém Behavior Tree poskytuje vizuální reprezentaci rozhodovacího procesu. Tento proces lze ladit a zprostředkovat tak co nejlepší zážitek ze hry.

Cílem není vytvořit neporazitelnou umělou inteligenci proti, které by nikdo nemohl vyhrát, ale takovou proti které je zábavné hrát.

Obrázek 28: Z jakých častí se skládá pokročilá umělá inteligence.

3.4.1 AI Controller

Behavior Tree musí být umístěn ve třídě `Controller` , skrze kterou může řídit třídu `BP_Gladiator`. Hlavní logika rozhodování je Behavior Tree a třída `AIController` je podpůrnou třídou, která poskytuje Behavior Tree informace o světě hry. Hlavní logika:

- **Event Begin Play** - Inicializuje Behavior Tree, nastavuje hodnoty v Blackboard a získává reference agentů v souboji.
- **Event Tick** - Vykonává se každý snímek hry. Aktualizovaná data v Blackboard. Provádí funkci strafe, které pohybuje umožňuje agentovi kroužit kolem jeho cíle

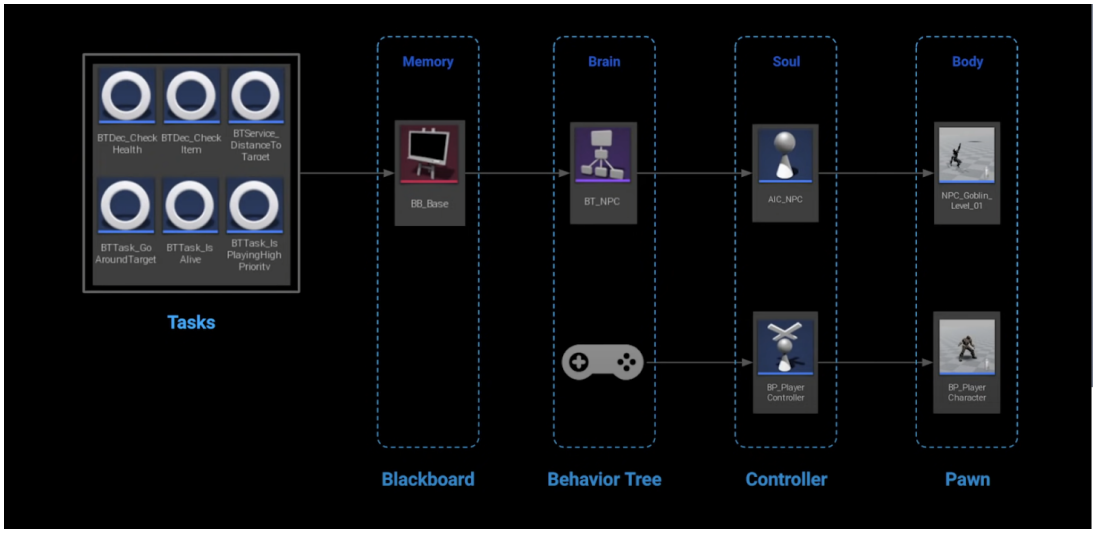
3.4.2 Popis systému Behavior Tree

Systém, který skrze třídu `AIController` řídí chování agenta v herním světě (obrázek č. 29). Třída `AIController` poskytuje informace o tom jaké eventy se dějí v herním světě a Behavior Tree podle nich může změnit své chování.

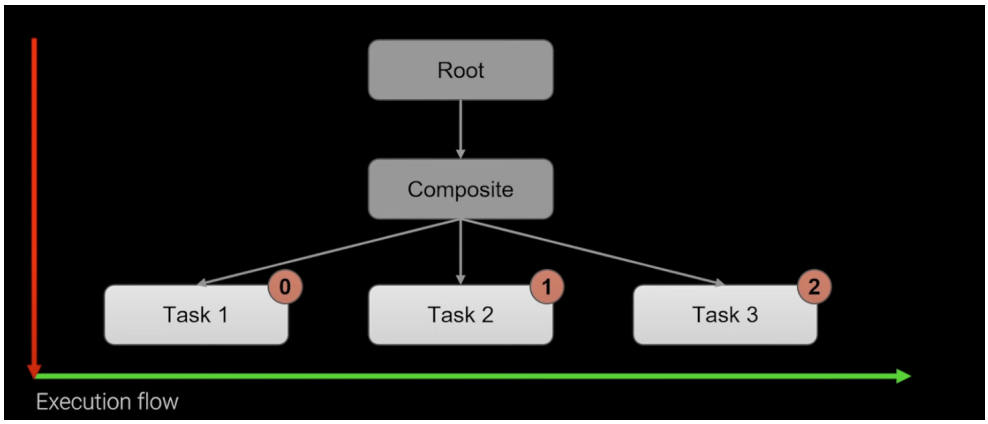
Behavior Tree se skládá z řady uzlů, z nichž každý představuje specifické chování nebo úkoly, které může umělá inteligence provést. Behavior Tree vybírá,

jaké chování zvolit. Tento výběr se řídí pravidly, které mají sadu parametrů, které herní návrhář ladí a upravuje.

Uzly jsou spolu spojeny v hierarchické struktuře, přičemž kořenový uzel je nahoře a listové uzly dole. Procházení této struktury začíná v kořenovém uzlu a prochází se shora dolů. Vybírání chování a vykonávání úkolů se děje zleva doprava (obrázek č. 30).



Obrázek 29: Systém Behavior Tree a jeho prvky [17].



Obrázek 30: Behavior Tree execution flow diagram [17].

3.4.3 Paradigma a historie

Systém se stal populární kvůli svému paradigmatu: schopnost vytvořit komplexní chování pouze pomocí akcí (která obsahují kód) a stromu, který dané akce řídí. Listové uzly stromu jsou akce a vnitřní uzly určují rozhodování. Tvorba a editace

stromu by měla být prováděna ve vizuálním editoru, který nevyžaduje znalost programování.

Systém je vizuální intuitivní a jde v něm snadno navrhnout, testovat a debugovat. Poskytuje velkou modularity, škálovatelnost a opětovnou použitelnost oproti ostatním metodám řízení chování, jako jsou konečné automaty.

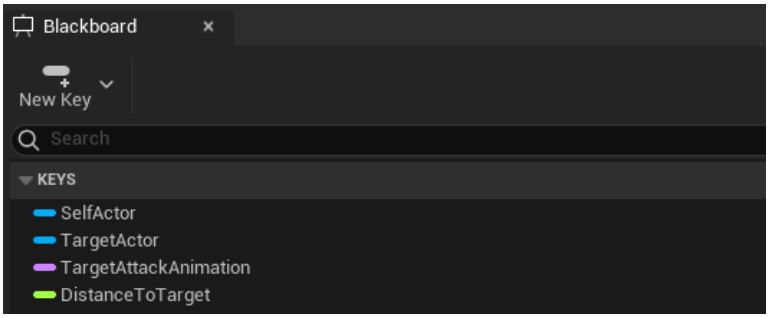
Behavior Tree systém se stal populární před více než deseti lety kvůli komerčně úspěšné hře Halo 2 (Isla, 2005; Microsoft Game Studios, 2004), kterou následovaly další jako Halo 3 (Isla, 2008; Microsoft Game Studios, 2007), Bioshock (2K Games, 2007), Spore (Champandard & Dunstan, 2012; Electronic Arts, 2008).

Několik novějších titulů, které potvrdily použití systému Behavior Tree ve svém vývojí jsou Far Cry: Primal (Ubisoft, 2016a), Tom Clancy's The Division (Ubisoft, 2016b) a Just Cause 4 (Square Enix, 2018) [18].

3.4.4 Popis prvků systém Behavior Tree

Blackboard (Data pro Behavior Tree)

Základním prvkem Behavior Tree je Blackboard, objekt, který slouží jako paměť systému (obrázek č. 31). Vše o čem chceme, aby umělá inteligence věděla, bude zaznamenáno v Blackboard jako klíč-hodnota. S těmito daty může poté umělá inteligence pracovat a měnit podle nich svoje chování. Data jsou aktualizovány třídou `AIController` , nebo prvkem zvaným Service.



Obrázek 31: Proměnné, které ovlivňují rozhodování systému Behavior Tree.

Task (Úkoly)

Task je uzel reprezentující úkol, který chceme, aby agent vykonal. Task může mít parametry, podle kterých se vykonává. Task se provede v momentě, kdy je vybrán. Po skončení Task můžeme informovat Behavior Tree o tom, jakým způsobem byl ukončen (k tomu slouží Finish with Result Task). Popis základních Task:

- **Wait (Brain Lag)** - Task, který proces rozhodování a vykonávání na určitou dobu zastaví.
- **Move To** - Task, který začne s agentem pohybovat na určené místo.
- **Play Animation** - Task, který spustí agentovi specifikovanou animaci.
- **Play Sound** - Task, který na daném místě přehraje specifikovaný zvuk.
- **Run Behavior** - Task, který umožňuje spustit Behavior Tree.
- **Set Tag Cooldown** - Task, který nastaví časovač pro určité chování.

Specifické pro třídu `BP_Gladiator`:

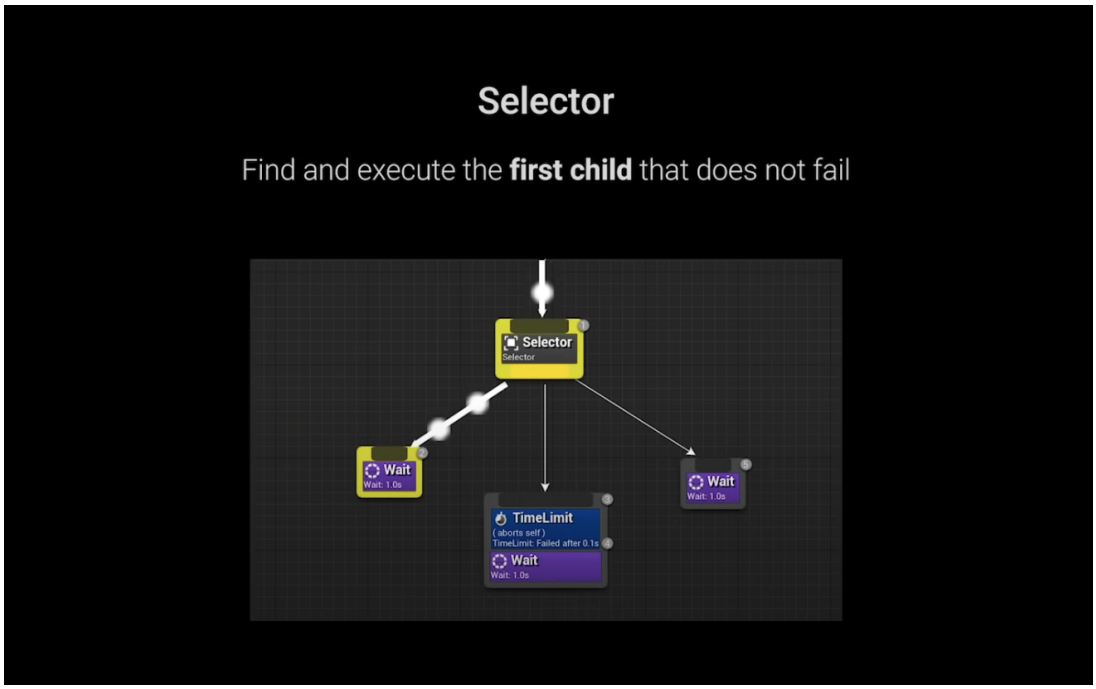
- **Update Target Distance** - Task, který každý snímek hry aktualizuje vzdálenost mezi agentem a jeho cílem.
- **Basic Attack** - Task, který v ovládané třídě `BP_Gladiator` vyvolá Event Basic Attack.
- **Dodge** - Task, který v ovládané třídě `BP_Gladiator` vyvolá Event Dodge.
- **Special Attack** - Task, který v ovládané třídě `BP_Gladiator` vyvolá Event Special Attack.

4 základní stavy úkolu:

- **Succeeded** - Task skončil úspěšně.
- **Failed** - Task skončil neúspěšně.
- **Aborted** - Task skončil, ale byl přerušen.
- **In Progress** - Dokončuje se jako rozpracovaný.

Selector (Výběr podstromu)

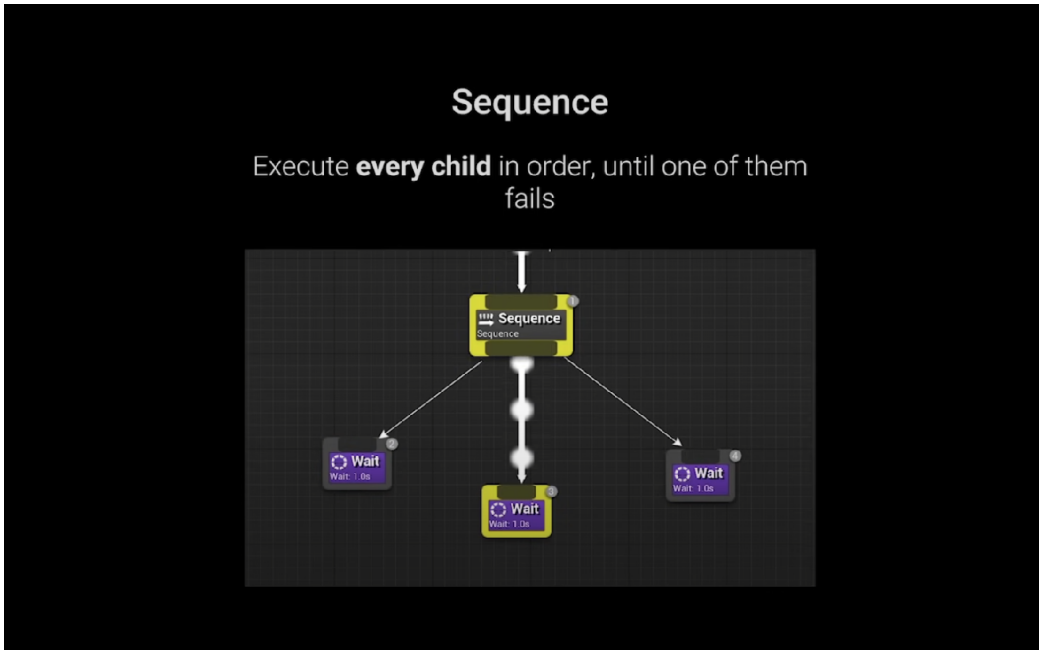
Selector vybírá první podstrom, který se může spustit a čeká na jeho dokončení (obrázek č. 32) Obvykle se používá k výběru podstromu, který se má provést. Po skončení vykonávání podstromu vrátí kontrolu svému rodiči, který pokračuje v rozhodovacím procesu [19]. V této práci je Selector použit pro rozhodování o tom, zda má umělá inteligence provést úhyb. Dokud hráč nezačne provádět útok, nemůže Selector zvolit chování úhybu.



Obrázek 32: Způsob jakým funguje Selector [17].

Sequence

Sequence se používá k vykonání sekvence úkolů v řadě za sebou (obrázek č. 33). Na rozdíl od Selector pokračuje Sequence ve vykonávání, dokud nedosáhne uzlu, který selže. Poté vrátí kontrolu svému rodiči, který pokračuje v rozhodovacím procesu [19] . V této práci je pomocí Sequence vytvořen většina chování.



Obrázek 33: Způsob jakým funguje Sequence [17].

Decorator (Podmínky vstupu)

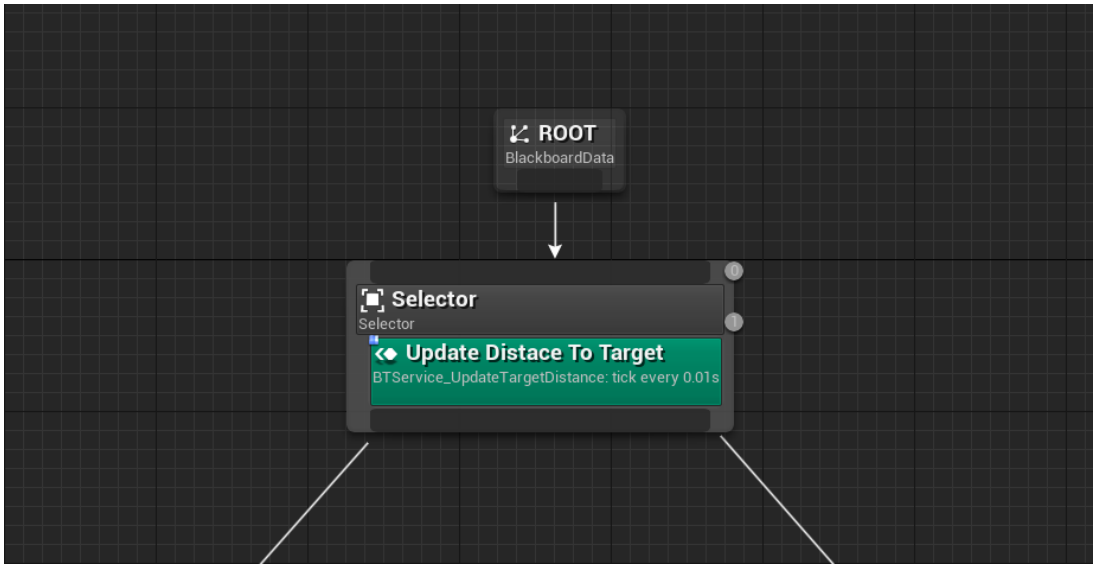
Decorator určuje podmínky, které musí být před vstupem do větve splněny (obrázek č. 34).

- **Is Enemy Attacking** - Můžeš vstoupit pouze pokud nepřátelský agent přehrává animaci útoku.
- **Blackboard condition** - Proměnná musí splňovat podmínku.
- **Cooldown** - Časovač podmínky nesmí být aktivní.
- **Is at Location** - Ovládaný agent musí být na daném místě.

Obrázek 34: Podmínka vstupu: nepřítel musí být v animaci útoku.

Service (opakovaně vykonávaný kód větve)

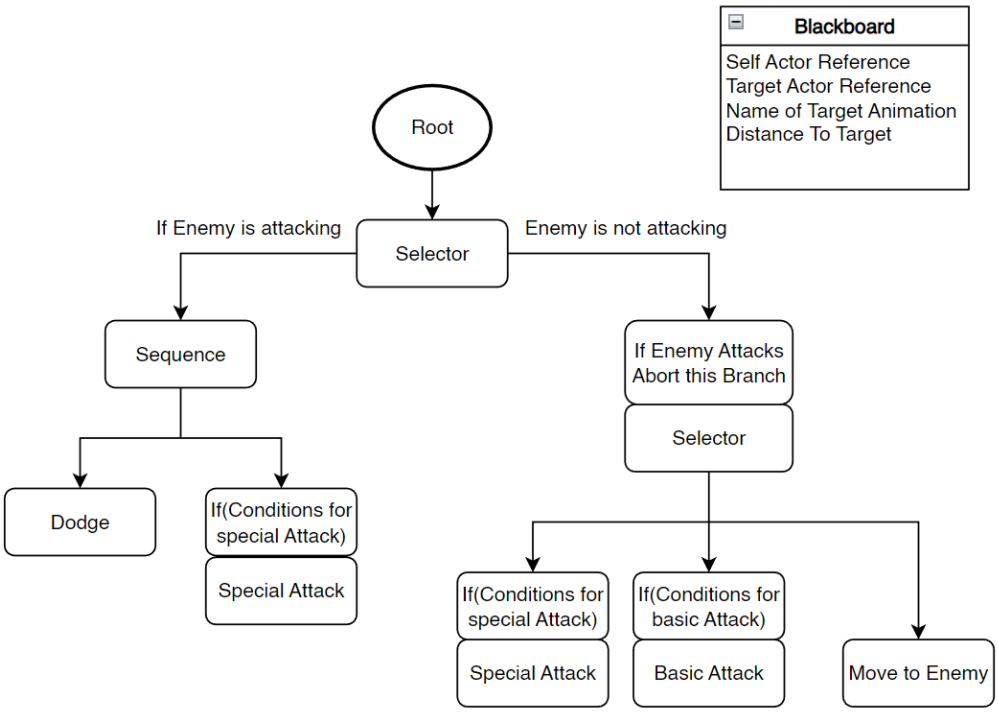
Service je kód, který se připojí k uzlu a opakovaně se podle definované frekvence vykonává. Service se opakovaně vykonává, dokud se nepřestane vykonávat větev, ke které je připojen (obrázek č. 35). Používá se ke kontrole a aktualizaci informací v Blackboard [19] . V této práci je Service používám pro aktualizaci proměnných.



Obrázek 35: Service, který každý snímek hry aktualizuje vzdálenost mezi agentem a jeho cílem.

Vývojový Diagram Rozhodování

Vývojový diagram na obrázku č.36 ukazuje hlavní logiku, podle které se Behavior Tree řídí. Všimněte si, že pravá větev přeruší své vykonávání v momentě, kdy zaznamená, že nepřátelský agent začíná provádět útok. Díky tomu může umělá inteligence dynamicky zrušit prováděný Task a přepnout na levou větev, která vykoná logiku úhybu. Behavior Tree vybírá chování útoku podle toho, v jaké vzdálenosti se oba agenti od sebe nacházejí.



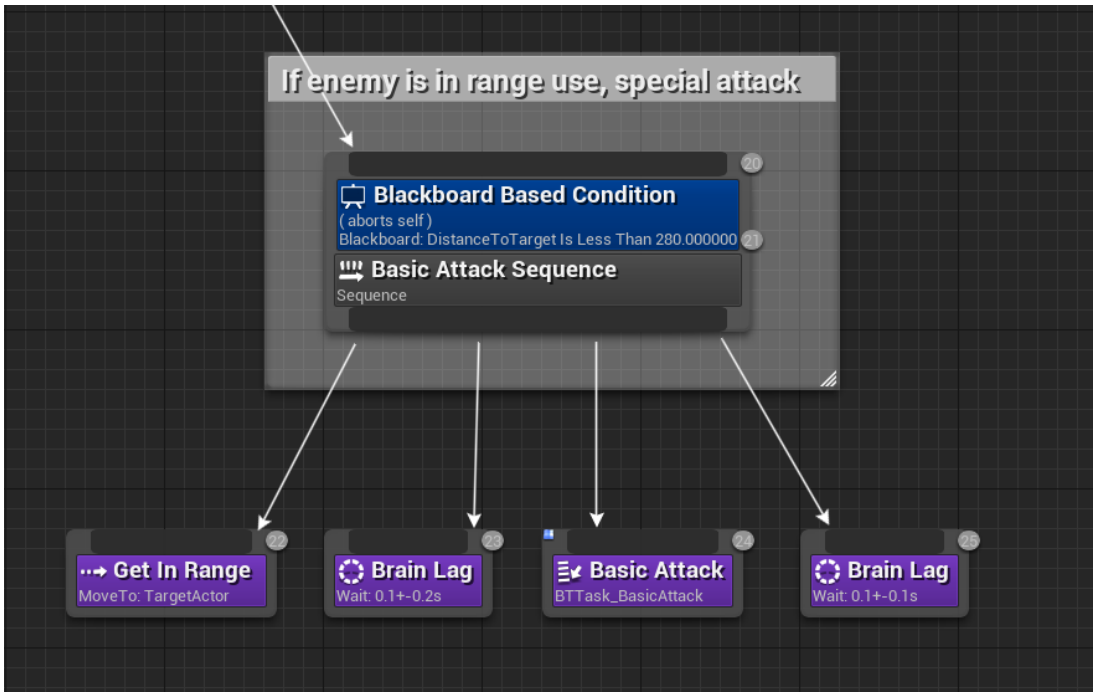
Obrázek 36: Diagram rozhodování pokročilé umělé inteligence.

Popis větve

Tato větev (obrázek č. 37) se skládá z uzlu Sequence s Dekorátorem a má čtyři úkoly. Aby bylo do větve možné vstoupit, musí být splněná podmínka dekorátoru. Ta vyžaduje, aby vzdálenost mezi agentem a jeho cílem byla větší než čtyři metry. Pokud tato podmínka není splněna, Selector nemůže tuto větev vybrat a vybere jinou. Pokud podmínka splněná je a Selector tuto větev vybere, začnou se provádět úkoly do doby, než se všechny nesplní nebo jeden selže.

Úkoly se budou vykonávat v tomto pořadí:

1. Přibliž se k agentovi na dosah speciálního útoku
2. Simuluj mozkové zpoždění (Brain lag)
3. Aktivuj Event speciální útok ve ovládané třídě `BP_Gladiator`
4. Simuluj mozkové zpoždění (Brain lag)
5. Vrať kontrolu rozhodování



Obrázek 37: Větev speciálního útoku.

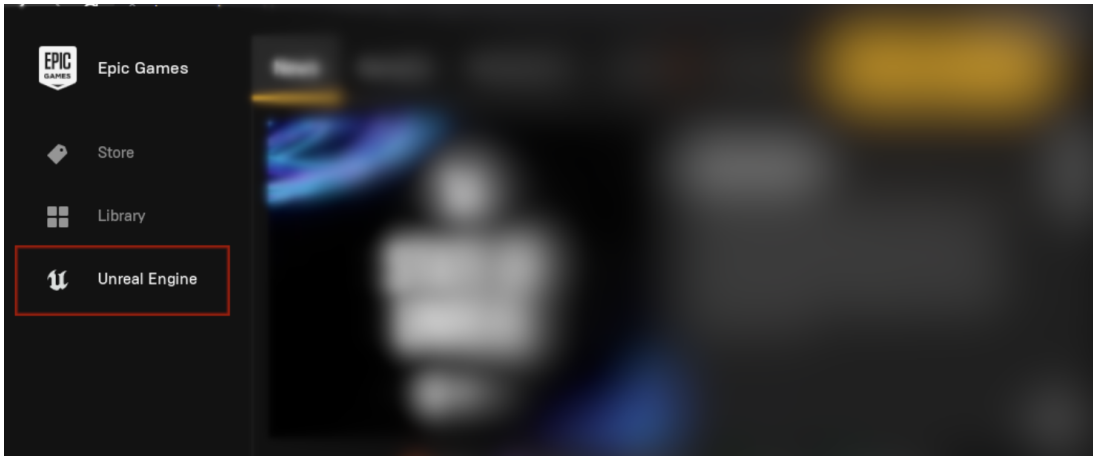
3.5 Kompilace Projektu

Aby bylo možné projekt otevřít a zkompilovat, musí být nainstalovaný Unreal Engine 5. Nejsnazší možností instalace je stáhnutí *Epic Games launcher* ve kterém je možné Unreal Engine stáhnout a nainstalovat. Epic Games launcher je možné stáhnout na adrese: [20]. Pro nejlepší kompatibilitu doporučuji nainstalovat verzi 5.0.3, ve které byl projekt vytvořený. Unreal Engine není zpětně kompatibilní a novější verze, mohou provést změny, které by vedli k chybám v projektu. Pokud nastane případ, kde Unreal Engine potřebuje vytvořit soubory pro jinou verzi, tak ukáže hlášku *Missing MalicekAIProject Modules* zvolte *Yes* a Unreal Engine sám vytvoří vše potřebné.

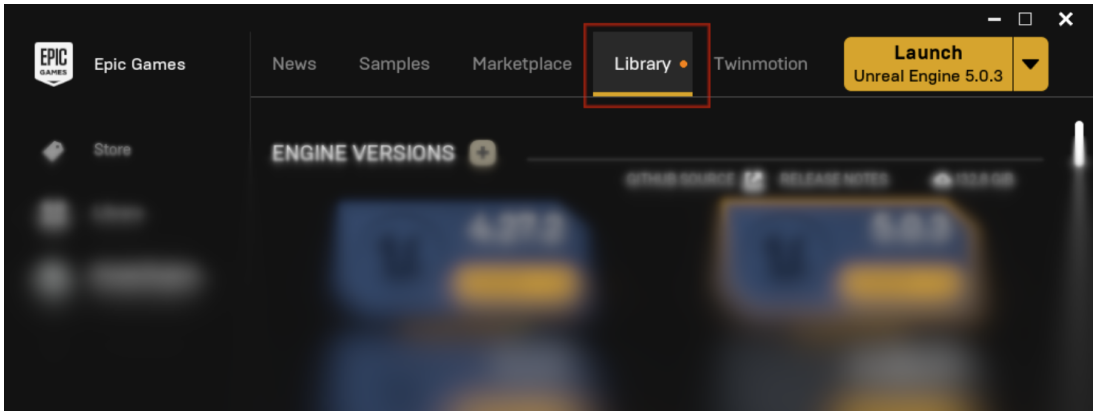
Pro spuštění projektu a demonstrace je potřeba mít v počítači podporovanou grafickou kartu (oficiální specifikace [21]). Integrované grafické karty také fungují, ovšem jejich nízký výkon, může způsobit trhanost hry. Demo je spustitelné na školních počítačích v učebně 5.003 katedry informatiky.

3.5.1 Instalace Unreal Engine pomocí Epic Games Launcher

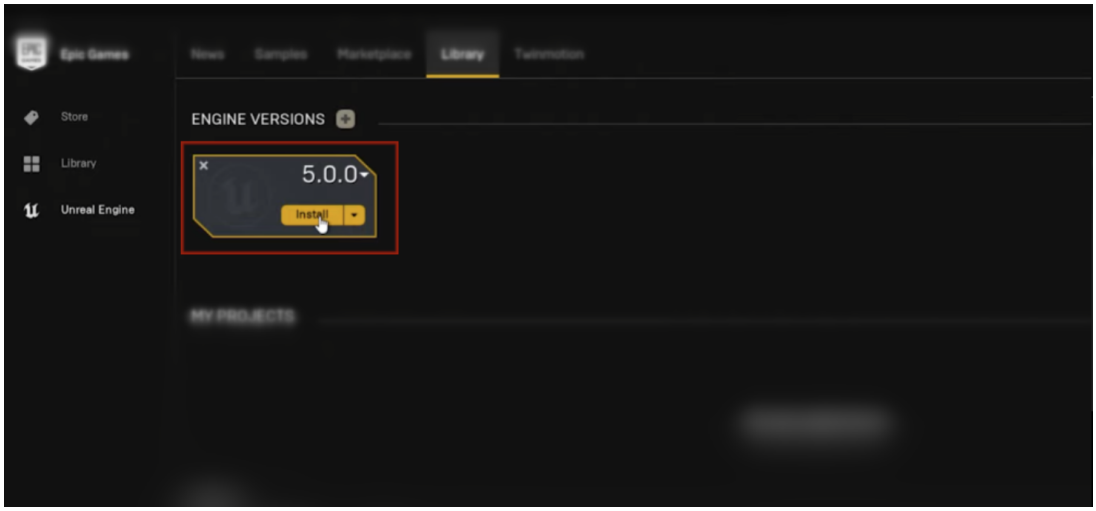
Popis postupu instalace Unreal Engine 5.



Obrázek 38: Kliknutím na “Unreal Engine” se dostaneme do sekce Unreal Engine.



Obrázek 39: Vybereme sekci “Library” v ní můžeme Unreal Engine nainstalovat.



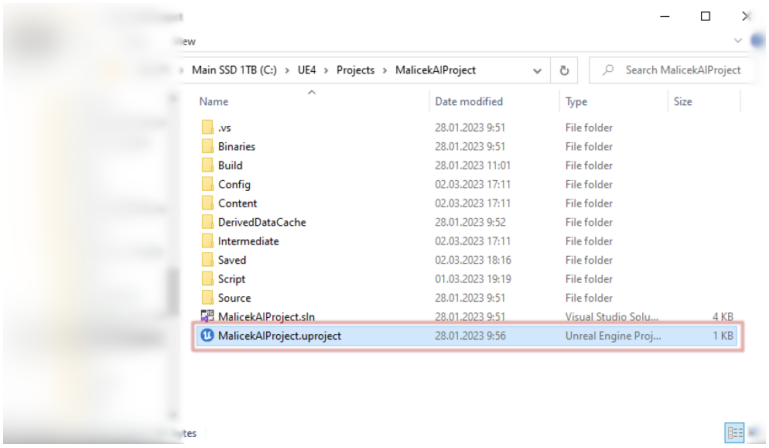
Obrázek 40: Zvolíme verzi a nainstalujeme Unreal Engine, doporučuji verzi 5.0.3.

3.5.2 Postup kompilace projektu:

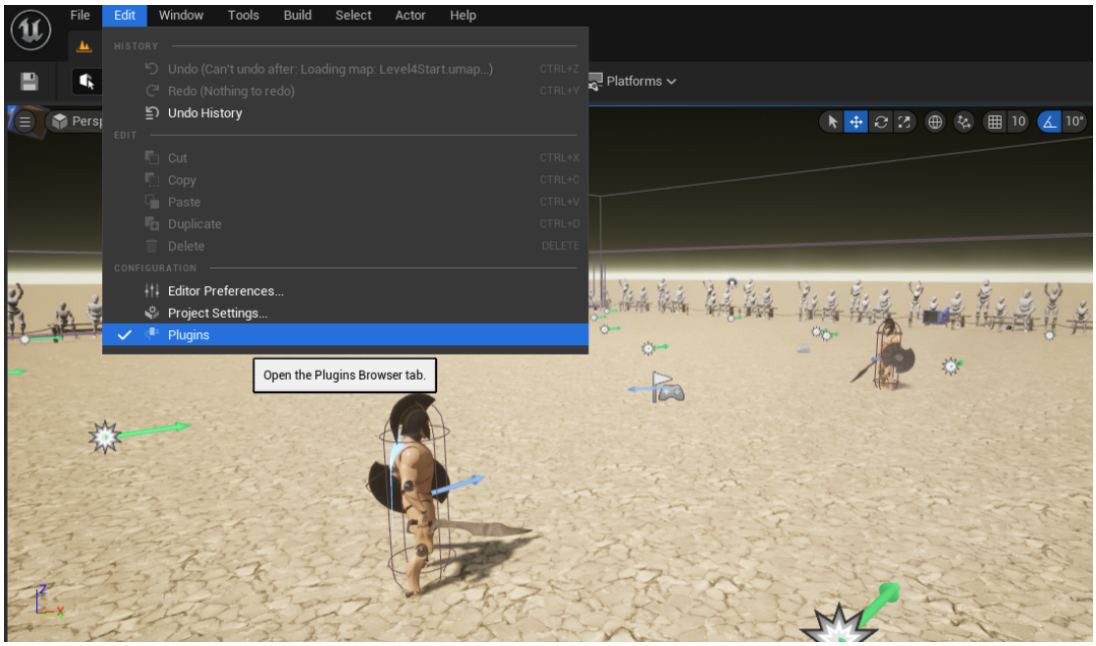
Projekt se kompiluje standardním způsobem, zvaným *"packaging"* (obrázek č. 44). Pokud je okolí arény tmavé, musíme aktivovat plugin *"HDRI Backdrop"* (obrázek č. 42), který zobrazuje okolí školy a je standardně vypnutý.

Hláška: Missing MalicekAIProject Modules

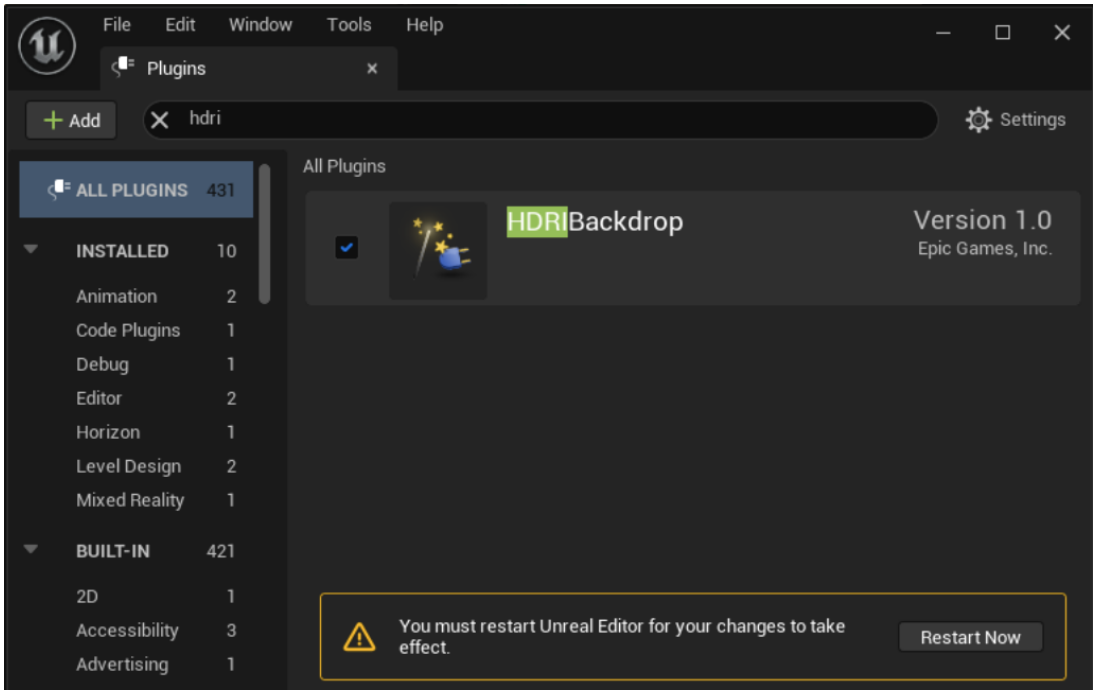
Pokud se objeví hláška "Missing MalicekAIProject Modules", zvolte Yes.



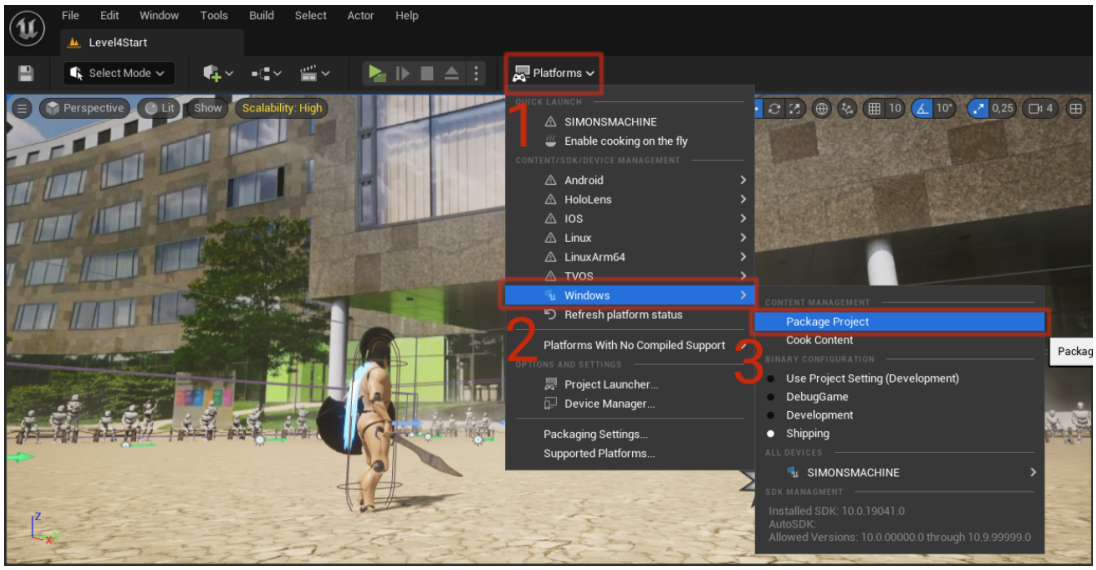
Obrázek 41: Spustíme soubor MalicekAIProject.uproject.



Obrázek 42: 1. Rozklikneme “Edit” (levý horní roh) a zvolíme “Plugins”.

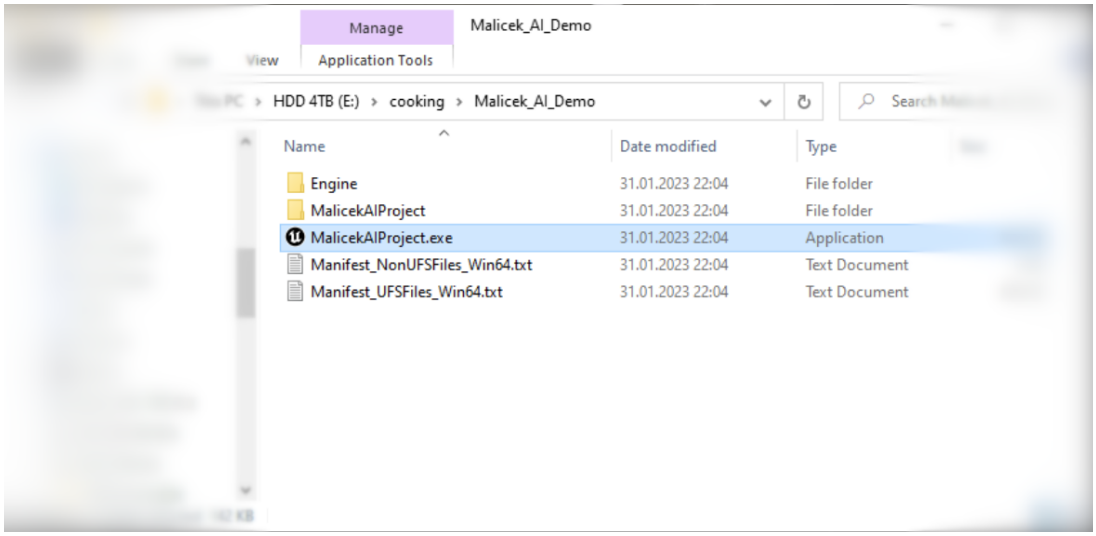


Obrázek 43: 2. V menu najdeme a aktivujeme plugin s názvem “HDRI Backdrop”, poté se musí projekt restartovat.



Obrázek 44: Rozklikneme tlačítko “Platforms”, poté “Windows” a zvolíme “Package Project”. Vybereme složku, kam se uloží zkompilovaný program a Unreal Engine začne proces kompilace.

Spustitelný soubor



Obrázek 45: Po dokončení kompilace vzniknou ve složce potřebné soubory a spustitelný soubor MalicekAIProject.exe, kterým lze hru zapnout.

4 Uživatelská Příručka

Tato kapitola popisuje hratelnou demonstraci umělé inteligence, jak ji nainstalovat, jaké existují herní módy, jak se ovládá a co znamenají její UI elementy.

4.1 Instalace hry

Hru není potřeba instalovat, stačí spustit soubor MalicekAIProject.exe. Je možné, že bude potřeba aktualizovat některé knihovny nebo ovladače. Pokud by se tak stalo, hra vše potřebné automaticky aktualizuje.

4.2 Popis hlavní nabídky

Do hlavní nabídka (obrázek č. 46) se ukáže po styknutí klávesy "Esc". Herní módy:

- **Basic AI (Troll)** - Herní mód, ve kterém hráč hraje proti základní umělé inteligenci, která ovládá třídu `BP_Troll`.
- **Advanced AI (Gladiator)** - Herní mód, ve kterém hráč hraje proti pokročilé umělé inteligenci, která ovládá třídu `BP_Gladiator`.
- **AI vs AI** - Herní mód, ve kterém hráč pozoruje zápas dvou instancí pokročilé umělé inteligence. Hráč je pouze pozorovatelem a nemůže zápas ovlivnit.
- **Exit** - Ukočení hry.

4.3 Popis ovládání

Hra má standardní ovládání (obrázek č. 47) a pohyb kamery.

- **Pohyb postavy** -Tlačítka WASD.
- **Pohyb kamerou** - Pohybem myši.
- **Základní útok** - Levé tlačítko myši.
- **Speciální útok** - Pravé tlačítko myši.
- **Úhyb** - Mezerník.

4.4 Grafické nastavení

Unreal Engine 5 je graficky velmi realistický, ale zároveň velmi náročný. Pokud hra není plynulá, doporučuji snížit grafické nastavení snížit (obrázek č. 48).

Obrázek 46: Hlavní nabídka hry, kde lze vybrat herní mód a grafické nastavení.

Obrázek 47: Graficky znázorněné ovládání.

Obrázek 48: Grafické nastavení hry.

4.5 UI Hry

Tato sekce popisuje UI elementy (obrázek č. 49), které jsou vidět při hraní.

Obrázek 49: Screenshot UI ze hry.

4.5.1 Zobrazení životů

Ke zobrazení životů je využitý UI element nad modelem agenta (obrázek č. 50).

Obrázek 50: UI element, který zobrazuje, kolik zbývá agentovi životů.

4.5.2 Zobrazení časovačů (Cooldowns)

Zobrazuje, zda může být schopnost aktivována. Pokud schopnost aktivujeme, zbarví se do šeda (obrázek č. 51).

Obrázek 51: UI element, který zobrazuje hráčovi akce a jejich časovače.

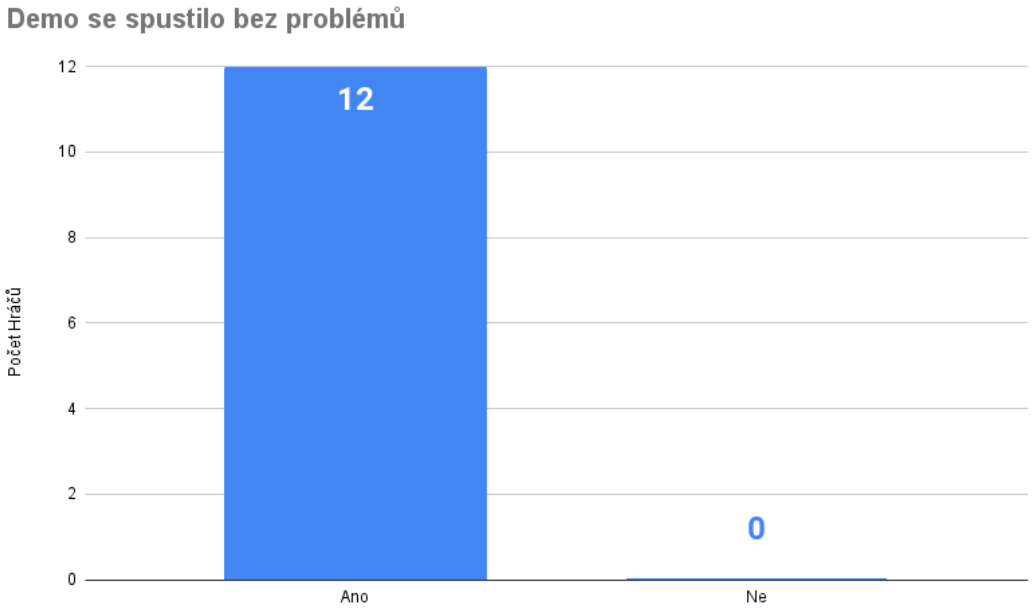
5 Výsledky uživatelského testování

Cílem testování je zjistit, zda hra funguje a zda se hráčům dobře hraje a ovládá. Demo hry bylo volně stažitelné na webovém portále *itch.io*, který se zaměřuje na hry od nezávislých vývojářů.

Hráči mohli vyplnit krátký anonymní dotazník, kde hodnotili různé aspekty hry a mohli sdělit, co se jim líbilo a nelíbilo. Použil jsem Likertovu škálu [22] a dobrovolný komentář. Celkem dotazník vyplnilo 12 hráčů.

5.1 Spustitelnost

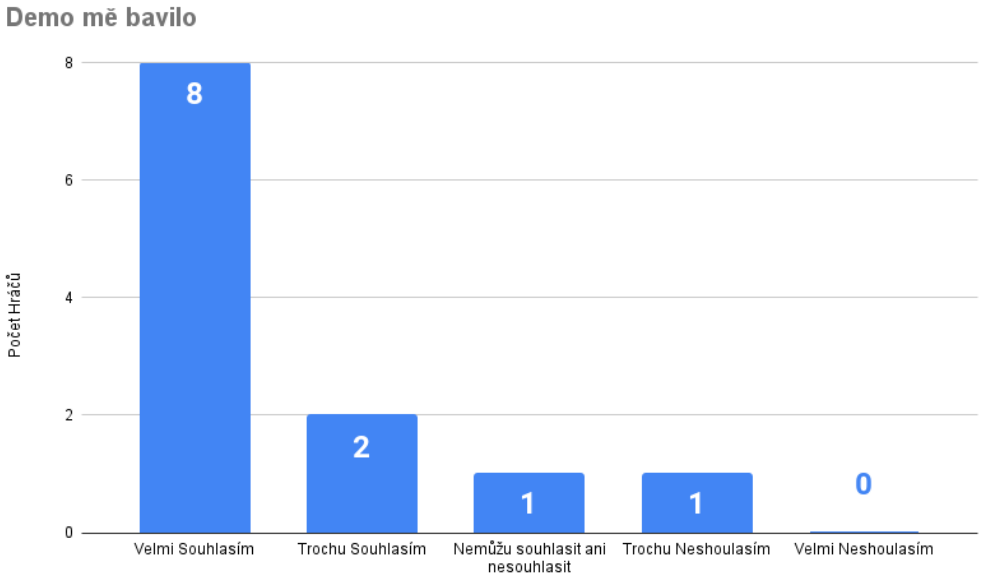
12 z 12 hráčů bylo schopno bez problému spustit demo (obrázek č. 52), tím se potvrzuje, že Unreal Engine byla dobrá volba herního enginu.



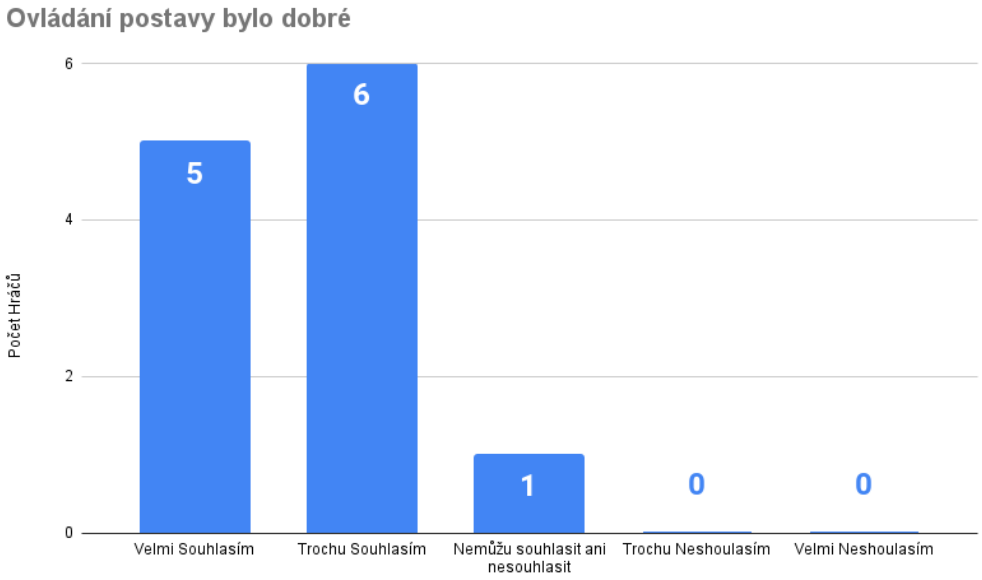
Obrázek 52: Výsledky testování spustitelnosti.

5.2 Herní návrh

Bylo důležité vytvořit demo tak, aby bylo zábavné a jeho ovládání intuitivní. Testování ukázalo, že se snaha vyplatila a demo bavilo hrát 10 z 12 hráčů (obrázek č. 53). Žádný hráč nehodnotil ovládání jako špatné (obrázek č. 54).



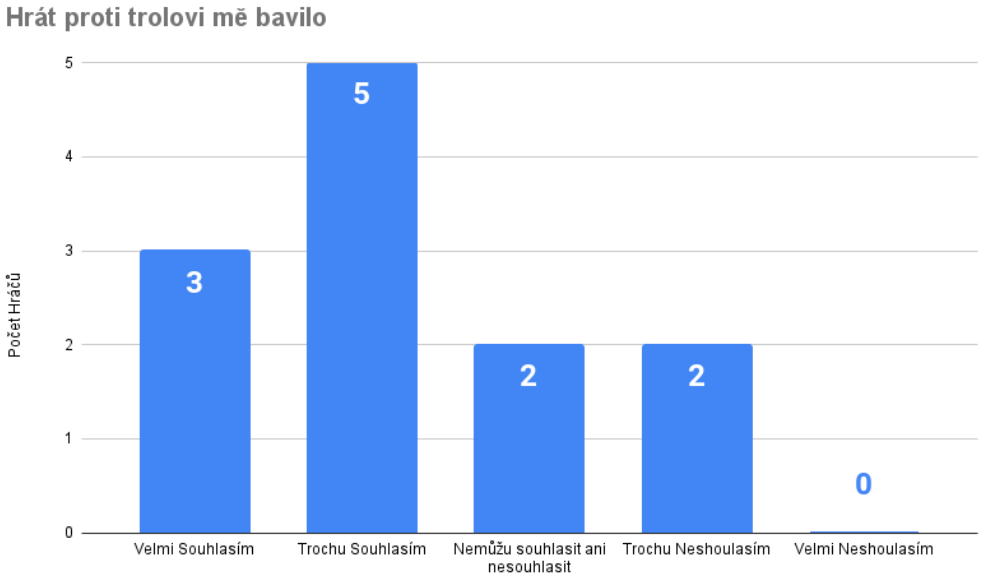
Obrázek 53: Výsledky testování zábavnosti hry.



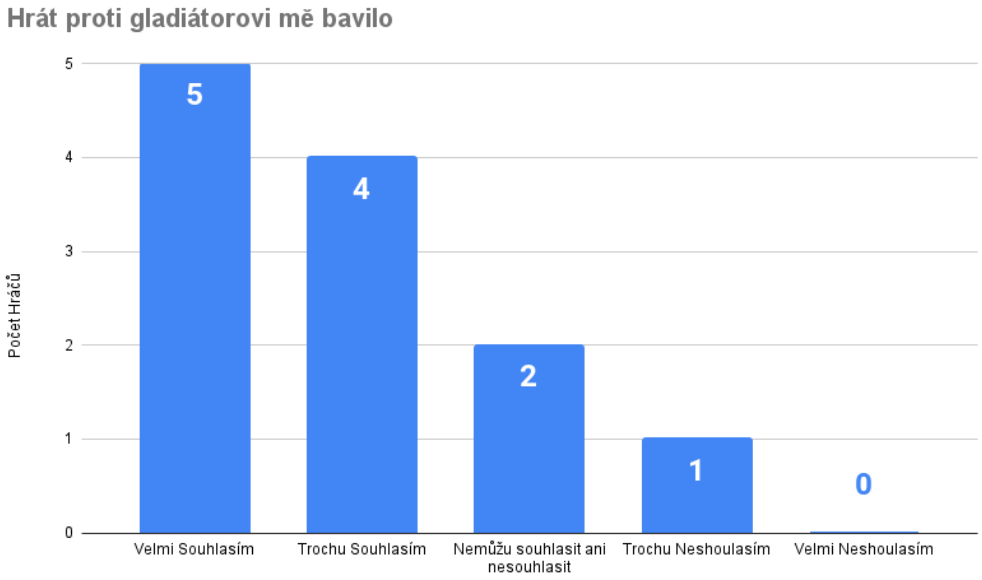
Obrázek 54: Výsledky testování pocitu z ovládání.

5.3 Umělá Inteligence

Obě třídy dopadly v hodnocení podobně, většina hráčů si souboj užila. Gladiátor (obrázek č. 56) ovládán pokročilou inteligencí měl u hráčů o trochu lepší výsledky, než trol (obrázek č. 55).



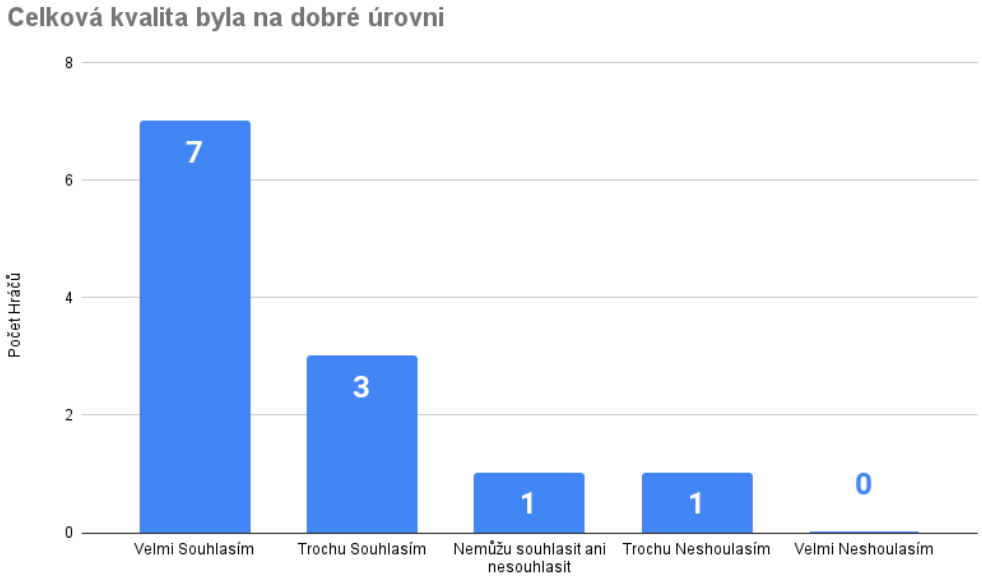
Obrázek 55: Výsledky testování základní umělé inteligenci.



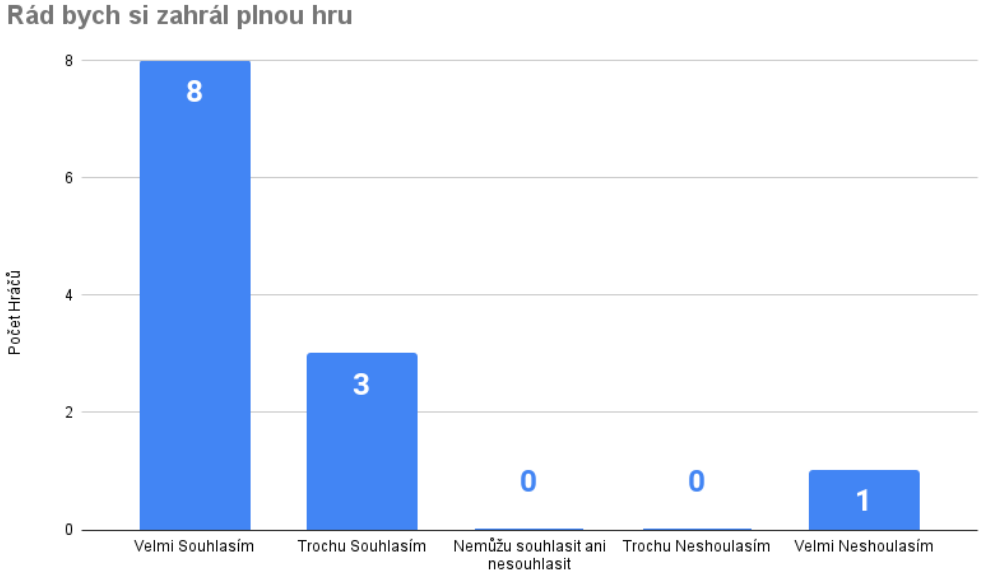
Obrázek 56: Výsledky testování pokročilé umělé inteligence.

5.4 Celkový pocit ze dema

10 z 12 hráčů vnímalo celkovou kvalitu dema jako na dobré úrovni (obrázek č. 57). 11 z 12 hráčů by si rádi zahráli plnou hru s tímto soubojovým systémem (obrázek č. 58).



Obrázek 57: Výsledky testování celkového pocitu z demonstrace.



Obrázek 58: Zájem testujících hráčů o plnohodnotnou verzi hry.

5.5 Návrhy na zlepšení

Testování odkrylo několik věcí, které je potřeba implementovat nebo zlepšit. Hlavní návrhy pro zlepšení hry:

- **Nastavení obtížnosti** - Možnost, která by umožňovala snížit obtížnost hry.
- **Tutoriál level** - Level, ve kterém by bylo vysvětleno, jak hra funguje.
- **Vizuální signalizace útoku** - Více signalizovat útok.
- **Problém "spamování" útoku** - Pomocí neustáleho "spamování" speciálního útoku lze zahnat nepřítele do kouta, kde nemá kam úteci.
- **Blokování** - Akce, která by mohla odrazit útok (tím by se mohl vyřešit problém spamování). Pokud by hráč moc "spamoval" útok, nepřítel by ho mohl blokovat a zastavit jeho útoky.

Závěr

Výstupem bakalářské práce je souboj s umělou inteligencí v reálném čase a jeho hratelná demonstrace. K vytvoření soubojového systému jsem využil Unreal Engine 5. Tím byly splněny oba cíle bakalářské práce, a to navrhnutí a implementace souboje hráče proti umělé inteligenci v reálném čase pro Unreal Engine.

Hlavním přínosem práce je soubojový systém, který mohou herní vývojáři implementovat do své hry. Dále tato práce popisuje teoretické základy herního návrhu a jejich praktickou implementaci. Práce popisuje systém Behavior Tree a jeho praktickou implementaci v Unreal Engine 5 .

Osobní motivací bylo získat znalosti z oblasti umělé inteligence v počítačových hrách a vytvoření soubojového systému, který je zábavný hrát. Díky zpětné vazbě od hráčů vím, že jsem toho docílil, protože většina hodnotila umělou inteligenci a soubojový systém pozitivně. Byl jsem velmi povzbuzen, když jedenáct z dvanácti testujících hráčů projevilo zájem zahrát si plnou hru.

Přestože se jedná pouze o demonstraci jednoho systému z celé hry, zpětná vazba od hráčů byla velmi pozitivní.

Další vývoj bude směřovat k vytvoření plné hry, která bude stavět na základech této práce a poznatků z ní získaných. Díky zpětné vazbě od hráčů vím, kde začít a co musím zlepšit.

Conclusions

The outcome of this bachelor's thesis is a combat system and its playable demonstration. That fulfilled both goals of the thesis, namely the design and implementation of real-time combat with an AI opponent in Unreal Engine and its playable demonstration.

The main contribution of the thesis is a combat system that game developers can implement in their games. Furthermore, this work describes the theoretical foundations of game design and showcases their practical implementation. The thesis also describes the Behavior Tree system and its implementation in Unreal Engine 5.

My motivation was to gain knowledge of AI in video games and create a combat system that is fun to play. Thanks to the feedback from the players, I know that I have achieved this, as the majority rated the combat system positively. I was very encouraged when eleven of the twelve testers expressed interest in playing the full game.

Although this is only a demo of one system from the entire game, the feedback from players has been very positive.

Further development will be focused on creating a full game that will build upon this thesis and the insights gained. Thanks to the feedback from the players, I know where to start, what to fix, and improve.

A Seznam Licencovaného obsahu

Licencované audiovizuální prvky a asety jsou označené a umístěny ve složce `04_NOT_FOR_DESTRIBUTION-AudioVisualLicensedAssets`. Nesmí se používat bez svolení jejich autora.

Zvukové efekty:

- https://assetstore.unity.com/publishers/62212
- https://assetstore.unity.com/publishers/27115
- https://assetstore.unity.com/publishers/52638
- https://assetstore.unity.com/publishers/62212
- https://assetstore.unity.com/publishers/68694
- https://www.youtube.com/watch?v=pUypls1WgCg
- https://assetstore.unity.com/publishers/27115
- https://www.zapsplat.com/

3D Modely a animace

- https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-enemies
- https://skfb.ly/68NFQ
- https://skfb.ly/68NFQ
- https://skfb.ly/Mu7n
- https://skfb.ly/FV6K
- https://skfb.ly/OYox
- https://www.mixamo.com/

Vizuální efekty

- https://www.unrealengine.com/marketplace/en-US/product/realistic-starter-vfx-pack-vol
- https://www.unrealengine.com/marketplace/en-US/product/niagara-footstep-vfx

Font a ikony

- https://fonts.google.com/specimen/Roboto

- https://assetstore.unity.com/publishers/59942
- https://icons8.com/
- https://icon-icons.com/
- https://www.behance.net/manshagraphics

360 foto univerzity

- https://goo.gl/maps/EZV2rVySRZLEjH4UA

B Obsah elektronických dat

U veškerých odjinud převzatých materiálů obsažených na jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Materiály, u kterých toto není splněno, jsou označeny ve složce `04_NOT_FOR_DESTRIBUTION-AudioVisualLicensedAssets` a nesmí se používat bez svolení jejich autora.

`Text/`
: Adresář s textem práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech (textových) příloh, a všechny soubory potřebné pro bezproblémové vytvoření PDF dokumentu textu.

`README.pdf`
: Soubor s informacemi o opakovatelném způsobu kompilace a spuštění práce.

`Unreal Engine 5 Project/`
: Adresáře s projektme pro Unreal Engine 5.

`Executable Demo/`
: Adresář se spustitlenou demonstrací souboje a příručkou pro uživatele.

C Seznam zkratek a cizích slov

- **AI** - umělá inteligence
- **Fair Play** - pocit, že hra je spravedlivá
- **Behavior Tree** - Systém pravidel, proměnných a úkolů, který řídí umělou inteligenci
- **State Machine** - Popisuje stavy a pravidla pro jejich změnu
- **Interface** - rozhraní pro správnou komunikaci a přenos dat mezi třídami
- **Framework** - softwarová struktura a pravidla pro organizaci a vývoj projektů
- **Herní Engine** - framework specializovaný na vývoj počítačových her
- **Unreal Engine** - herní engine specializovaný na 3D vykreslování v reálném čase
- **IDE** - vývojové prostředí usnadňující programování a debugování
- **2D Bitmapa** - obrázek tvořený pixely
- **Textura** - 2D bitmapa, se kterou umí herní engine pracovat
- **UI** - uživatelské rozhraní, fonty, ikony
- **HUD** - "Heads Up Display"zobrazuje uživatelské rozhraní
- **Navmesh** - "Navigation Mesh"abstraktní struktura používaný pro výpočet pohybu na mapě
- **Skeletální Model** - 3D model s kostrou, díky které se může pohybovat

Literatura

[1] Zelinka, Zbyněk. *Framework Unreal Editor – jeho vlastnosti a použití pro tvorbu 2D hry [online]*. 2015 [cit. 2023-03-20]. SUPERVISOR: Rudolf Pecinovský. Dostupný také z: ⟨https://theses.cz/id/i7dhy3/⟩.

[2] Moravec, Lukáš. *Optimalizační techniky v Unreal Enginu 4*. 2020 [cit. 2023-03-20]. Dostupný také z: ⟨http://hdl.handle.net/10084/140592⟩.

[3] Uzayr, Sufyan bin. *Mastering Unreal Engine: A Beginner's Guide (Mastering Computer Science)*. First. 2022. 224 s. ISBN 1032103132.

[4] Sweeney, Tim. *If You Love Something, Set It Free - Unreal Engine 4 Available for free*. 2015. Dostupný z: ⟨https://www.unrealengine.com/en-US/blog/ue4-is-free⟩.

[5] Unreal Wiki, Ureanl Fandom page. *Unreal 1 Screenshot*. Dostupný také z: ⟨https://unreal.fandom.com/wiki/Unreal_(video_game)/Gallery?file=%2521U1-Beta-SkyCaves.jpeg⟩.

[6] Epic Games, Inc. *Unreal Engine 5 Lumen Global Illumination*. Dostupný také z: ⟨https://docs.unrealengine.com/5.0/en-US/lumen-global-illumination-and-reflections-in-unreal-engine/⟩.

[7] Andiloro, Andrea. 'This must be the place': Understanding video game placeness through atmosphere and the refrain in Dark Souls. 2022.

[8] Stout, Mike. *Enemy attacks and telegraphing*. Dostupný také z: ⟨http://www.chaoticstupid.com/enemy-attacks-and-telegraphing/⟩.

[9] Epic Games, Inc. *Gameplay Framework Quick Reference*. Dostupný také z: ⟨https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/Framework/QuickReference/⟩.

[10] Epic Games, Inc. *Game Mode and Game State*. Dostupný také z: ⟨https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/Framework/GameMode/⟩.

[11] Epic Games, Inc. *Controller - documentation*. Dostupný také z: ⟨https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/Framework/Controller/⟩.

[12] Marketplace, Unreal Engine. *Infinity Blade: Adversaries*. Dostupný také z: ⟨https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-enemies⟩.

[13] Epic Games, Inc. *NavMesh Vizualization*. Dostupný také z: ⟨https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/ArtificialIntelligence/AIDebugging/⟩.

[14] Epic Games, Inc. *Character Movement Component*. Dostupný také z: ⟨https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/Networking/CharacterMovementComponent⟩.

[15] Epic Games, Inc. *Unreal Engine 5 Animation State Machines*. Dostupný také z: ⟨https://docs.unrealengine.com/5.1/en-US/state-machines-in-unreal-engine/⟩.

[16] Epic Games, Inc. *Unreal Engine - Animation Montage Overview*. Dostupný také z: ⟨https://docs.unrealengine.com/4.26/en-US/AnimatingObjects/SkeletalMeshAnimation/AnimMontage/Overview/⟩.

[17] YouTube, Unreal Engine. *Unreal Engine AI with Behavior Trees | Unreal Engine*. Dostupný také z: ⟨https://youtu.be/iY1jnFvHgbE⟩.

[18] Ramiro A. Agis Sebastian Gottifredi, Alejandro J. García. An event-driven behavior trees extension to facilitate non-player multi-agent coordination in video games. 2020. Dostupný také z: ⟨https://cs.uns.edu.ar/~ragis/Agis%20et%20al.%20(2020)%20-%20An%20event-driven%20behavior%20trees%20extension%20to%20facilitate%20non-player%20multi-agent%20coordination%20in%20video%20games.pdf⟩.

[19] Epic Games, Inc. *Behavior Tree Documentation*. Dostupný také z: ⟨https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/ArtificialIntelligence/BehaviorTrees/BehaviorTreeQuickStart/⟩.

[20] Epic Games, Inc. *Unreal Engine 5 Download*. Dostupný také z: ⟨https://www.unrealengine.com/en-US/download⟩.

[21] Epic Games, Inc. *Unreal Engine 5 Hardware Specifikace*. Dostupný také z: ⟨https://docs.unrealengine.com/5.0/en-US/hardware-and-software-specifications-for-unreal-engine⟩.

[22] Nuts, Survey. *Co to je Likertova škála, jak a kdy ji používat*. Dostupný také z: ⟨https://surveynuts.com/cs/blogs/103-co-to-je-likertova-skala-jak-a-kdy-ji-pouzivat#sthash.zLfHilig.eyFfsgQI.dpbs⟩.