Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Vývoj videoher v herním enginu Unity

Bc. Jiří Dřímal

Diplomová práce

2022

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Jiří Dřímal**
Osobní číslo:	**I20203**
Studijní program:	**N0613A140007 Informační technologie**
Téma práce:	**Vývoj videoher v herním enginu Unity**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování

Cílem práce je popis problematiky vývoje videoher a vytvoření ukázkové počítačové hry s využitím vhodné metodiky.
V teoretické části práce budou objasněny klíčové znalosti potřebné při vývoji videoher se zaměřením především na počítačové hry. Jedná se zejména o popis jednotlivých metodik a vývojových modelů, které se uplatňují během celého vývojového procesu. Dále se teoretická část bude zabývat i popisem a možnostmi herního enginu Unity.
V praktické části bude vytvořena konkrétní ukázková počítačová hra typu RPG ve stylu 2D tilemap. V rámci celého vývojového procesu budou vhodně uplatněny principy a metodiky zmíněné v teoretické části. V praktické části bude celý proces vývoje dokumentován podrobným popsáním všech jednotlivých vývojových fází.
Hra bude vytvořena jako desktopová aplikace s využitím jazyka C# a vývojového nástroje Unity. Při vývoji hry budou kromě dostupných nástrojů, skriptů a assetů třetích stran použity i vlastní skripty a assety.

Rozsah pracovní zprávy: **cca 60 normostran**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

- NOVAK, Jeannie. *Game Development Essentials: An Introduction*. 3rd edition. New York: Cengage Learning, 2011. ISBN 978-1111307653.
- BATES. *Game Design*. 2nd ed. Boston: Premier Press. ISBN 9781592004935.

Vedoucí diplomové práce: **prof. Ing. Antonín Kavička, Ph.D.**
Katedra softwarových technologií

Datum zadání diplomové práce: **8. listopadu 2021**
Termín odevzdání diplomové práce: **20. května 2022**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

prof. Ing. Antonín Kavička, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 30. listopadu 2021

Prohlašuji:

Práci s názvem Vývoj videoher v herním enginu Unity jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 19.05.2022

Bc. Jiří Dřímal v. r.

PODĚKOVÁNÍ

Děkuji panu Ing. Petru Veselému za pomoc, vstřícnost a možnost konzultací při tvorbě mé diplomové práce. Dále děkuji panu prof. Ing. Antonínu Kavičkovi, PhD za vedení této práce. A také děkuji své rodině za velkou podporu během celého studia.

ANOTACE

Diplomová práce se zabývá popisem procesu vývoje videoher s následným vytvořením ukázkové počítačové hry s využitím vhodné metodiky v herním enginu Unity. V diplomové práci jsou objasněny důležité pojmy a specifika z oblasti vývoje videoher jako například herní design. Dále se práce zaměřuje na modely a metodiky vývojového procesu, herní vývoj v praxi a herní engine Unity. V rámci praktické časti je předveden celý vývojový proces, který je zakončen vydáním konkrétní vytvořené počítačové hry.

KLÍČOVÁ SLOVA

proces vývoje videoher, hra, herní engine Unity, metodiky vývojového procesu

TITLE

Video game development in the Unity game engine

ANNOTATION

The diploma thesis deals with the description of the video game development process followed by the creation of a sample computer game using a suitable methodology in the Unity game engine. The thesis explains important concepts and specifics in the field of video game development such as game design. Furthermore, the thesis focuses on models and methodologies of the development process, game development in practice and the Unity game engine. In the practical part, the whole development process is demonstrated, which ends with the release of a specific developed computer game.

KEYWORDS

video game development process, game, Unity game engine, development proces methodologies

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

2D	Two-Dimensional (dvoudimenzionální, dvourozměrný)
3D	Three-Dimensional (trojdimenzionální, trojrozměrný)
4K	Označení pro rozlišení obrazu s přibližně 4 000 pixely
AAA	Označení pro vysoko rozpočtové hry
AI	Artificial Intelligence (umělá inteligence)
API	Application Programming Interface (aplikační rozhraní)
AR	Augmented Reality (rozšířená realita)
CPU	Central Processing Unit (centrální procesorová jednotka)
DAW	Digital Audio Workstation (zařízení či SW pro práci se zvukem)
DLC	Downloadable Content (stahovatelný obsah)
DRM	Digital Rights Management (správa digitálních práv)
DVD	Digital Versatile Disc (digitální víceúčelový disk)
FPS	Frames per Second (snímková frekvence)
GDACZ	Game Developers Association Czech (Asociace českých herních vývojářů)
GDD	Game Design Document (dokument herního designu)
GDLC	Game Development Life Cycle (herní vývojový cyklus)
GPU	Graphics Processing Unit (grafický procesor)
GUI	Graphic User Interface (grafické uživatelské rozhraní)
HDR	High Dynamic Range (vyšší dynamický rozsah)
HUD	Heads-Up Display
HW	Hardware
ID	Identifier (identifikátor)
IDE	Integrated Development Environment (vývojové prostředí)
KISS	Keep it Simple, Stupid

LOD	Level of Detail (úroveň detailu)
LTS	Long-Term Support (verze s dlouhodobou podporou)
LVL	Level (úroveň)
MVP	Minimum Viable Product (produkt s nejmenší možnou funkcionalitou)
NFT	Non-Fungible Token (nezaměnitelný token)
NPC	Non-Playable Character (nehratelná postava)
OS	Operating System (operační systém)
P&L	Profit and Loss statement (výkaz zisku a ztráty)
P2W	Pay to Win (zaplať a vyhraj)
PC	Personal Computer (osobní počítač)
PoC	Proof of Concept (důkaz principu)
PR	Public Relations (vztahy s veřejností)
QA	Quality Assurance (zajištění jakosti)
RAM	Random-Access Memory (paměť s přímým přístupem)
RPG	Role-Playing Game (hra na hrdiny)
RUP	Rational Unified Process
SSD	Solid-State Drive (polovodičový disk)
SW	Software
TMP	TextMeshPro
UI	User Interface (uživatelské rozhraní)
UML	Unified Modeling Language
URL	Uniform Resource Locator (webová adresa)
VR	Virtual Reality (virtuální realita)
XP	Experience Point (zkušenostní body)
XP	Extreme Programming (extrémní programování)

ÚVOD

Odvětví videoherního průmyslu se každoročně rozrůstá a to, ať už z pohledu ekonomického či sociálního dopadu. Náročnost vývoje videoher se taktéž zvětšuje a bez moderních vývojových nástrojů, postupů a pokročilých herních enginů by vývoj velkých tzv. AAA her již zřejmě nebyl udržitelný. Stále se však videoherní průmysl potýká s problémy, jako je (zejména v česku) signifikantní nedostatek kvalifikovaných pracovníků, resp. nedostatek vývojového know-how uchazečů o práci. Mnohé vývojářské firmy tak musí buď přijmout lépe zkušené vývojáře ze zahraničí (to však může být administrativně náročné), anebo dovzdělávat své nové zaměstnance, což však růst celého odvětví značně zbrzďuje.

Velké moderní herní enginy, jako je například Unity, mohou tuto situaci zlepšit. Licenční politika herních enginů je často velmi přívětivá, a tak dovoluje téměř beznákladový vývoj jakémukoliv týmu. Malé vývojářské týmy, či dokonce jednotlivci si tak mohou vyzkoušet videoherní vývoj a získat tak cenné praktické zkušenosti, které jim poté zaručeně pomohou na trhu práce.

Rozmach herních enginů lze zřetelně pozorovat na narůstajícím počtu malých (středních) her od menších (nezávislých) vývojářských studií. Hry lze samozřejmě vytvářet i bez použití herních enginů, avšak potom je aktuálními zákazníky požadovaná míra kvality (resp. komplexnosti, originality) hry téměř nedosažitelná. Herní enginy zajišťují vývojářům technické základy hry, a tak své volné prostředky mohou zužitkovat spíše na samotnou herní náplň (lepší hratelnost, více obsahu, lepší použité technologie, větší úroveň detailu apod.).

Cílem diplomové práce je průzkum celého procesu videoherního vývoje v herním enginu, jenž bude nejprve popsán z teoretického hlediska, zahrnující i popis jednotlivých vývojových metodik a modelů. Následně se práce zaměří na praktické hledisko vývoje, a to konkrétně realizací celého vývojového procesu, za pomocí vhodně zvolené vývojové metodiky. Produktem tohoto procesu bude funkční počítačová hra menšího rozsahu, vyvinuta ve zvoleném herním enginu Unity. Práce je tak určena především pro začínající videoherní vývojáře s minimální praktickou zkušeností vývoje, ale také pro kohokoliv, koho dané téma zajímá a má přehled alespoň o základních pojmech jako jsou: videoherní průmysl, programovací jazyk (především C#) apod.

Vysvětlení těchto a dalších pojmů týkajících se vývoje videoher a herních enginů, lze mimo jiné nalézt v mé bakalářské práci „Vývoj videoher a testování herních enginů“. Tato práce byla vypracována a úspěšně obhájena v roce 2020. Zabývala se zevrubným

teoretickým popisem vývoje videoher a videoherního průmyslu a praktickým srovnáním vybraných herních enginů. Z tohoto srovnání herních enginů jsem využil získané poznatky při tvorbě této diplomové práce. Jedním z těchto užitečných poznatků byla volba herního enginu Unity, který se právě nejlépe hodí pro začínající vývojáře a méně rozsáhlé hry.

Na téma vývoj videoher existují i jiné odborné práce, které však popisují toto téma z jiných pohledů, než kterým se ubírá tato diplomová práce. V kontrastu s ostatními pracemi se má práce zaměřuje především na celý komplexní proces vývoje videoher. Pro úplnost ještě doplním, že pojmy hra a videohra jsou zaměnitelné a označují produkty videoherního průmyslu, tedy digitální formy her (software s primárním účelem zábavy jako jsou např. PC hry, konzolové hry, mobilní hry, …). Tato práce se zaměřuje zejména na počítačové hry.

1 VYSVĚTLENÍ POJMŮ

Pro lepší porozumění práce je nutné nejprve objasnit pojmy, které jsou pro videohry a proces jejich vývoje specifické. Některé tyto pojmy vycházející z angličtiny jsou z důvodu neexistence ekvivalentních českých slov počeštěny (skloňovány), nebo jsou používány v originální anglické formě i v českém prostředí. Stejně tomu tak je i v této práci. Pro lepší srozumitelnost, je u některých pojmů v závorce uveden nejbližší český ekvivalentní výraz.

1.1 Hry

Hry, videohry či počítačové hry jsou velmi specifický typ software, jenž je odlišný zejména svým primárním zaměřením na interaktivitu s uživatelem, čímž pro uživatele vytváří určitý zážitek. Ten je zpravidla zpracován audiovizuální formou. Hry mají většinou určený nějaký postup, jak ji hráč má hrát a díky těmto interaktivním pravidlům a funkcím, jenž se nazývají herní mechaniky, se hráč dostává ve hře dál s cílem dále prohloubit jeho prožitek. Hry však nemusí být určeny pouze k zábavě, ale dají se využít také pro interaktivní výuku, umělecké vyjádření, zpracování vážných témat a virtuálních prostředí či posilování psychiky. Hry se tak dle typu zážitku, jenž chtějí vyvolat a stylu hraní, dělí na herní žánry.

Základní herní žánry jsou akční hry, adventury, akční adventury, RPG neboli role-playing games (hra na hrdiny), simulátory, logické nebo puzzle hry a strategické hry. Tyto základní herní žánry mají mnoho podžánrů a odvozenin. A často je zařazení hry do určitého žánru, díky inovátorským myšlenkám či kombinací jednotlivých žánrů, poněkud obtížné. Při vývoji je výběr žánru velmi důležitý, jelikož musí být kompatibilní se základní ideou vyvíjené hry. Například je důležité správné provázání herního žánru a cílové platformy či skupiny zákazníků. Každý žánr potřebuje při vývoji jiné zkušenosti a technologie.

AAA hry a indie (vysoko rozpočtové a nezávislé hry) jsou označení pro rozdělení her dle formy jejich vydání. Za AAA (triple-a) tituly se označují hry pod záštitou nějakého herního vydavatele, čímž jsou vývojářům, výměnou za úplnou kreativní svobodu, poskytnuty finanční prostředky a jiné benefity, a tím je od hry také očekávaná větší kvalita, obsah a cena. Získání podpory vydavatele však není jednoduché, a tak si vydání řeší někteří nezávislí (indie) vývojáři buď sami, nebo pomocí vydavatelských platforem, které však na rozdíl od vydavatelů nepřináší finanční benefity. Tyto nezávislé hry jsou poté menší s nižší prodejní cenou, ale často kreativnější, na rozdíl od spíše konzervativnějšího přístupu AAA vydavatelů.

Remaky a remastery (předělávky) jsou tituly, které nově zpracovávají (předělávají) původně vydanou hru. Remake je kompletní znovuvytvoření hry na základě původní myšlenky pomocí nových technologií, nových assetů, popřípadě drobných změn designu mechanik a ovládání. Remaster je zpravidla pouze vylepšení audiovizuálního zpracování, a proto jsou často zpracovávány pomocnými vývojářskými týmy (studii) či externími studii. Důvodů pro vytvoření nového zpracování dříve vydané hry může být mnoho, od podpory (budování) komunity, nefunkčnosti staré hry na nových systémech (platformách), neodpovídající kvalita zpracování zastaralého audiovizuálu (popř. herních mechanik, ovládání), až po čistě pragmatické finanční důvody či marketingové kampaně.

DLC, content update, datadisky jsou formy rozšíření původně vydaného produktu. Nejde o plnohodnotné pokračování, ale pouze o malé rozšíření myšlenky, funkcionality, obsahu či audiovizuálního pojetí. Velmi rozšířené jsou DLC neboli downloadable content (stahovatelný obsah), které jsou distribuovány skrze internet, a to buďto zdarma nebo častěji za zlomek původní ceny hry. Content update (aktualizace obsahu) je většinou menší než DLC a zpravidla nabízená zdarma. Datadisky již dnes nejsou tak časté, přidávají nejvíce obsahu a distribuovány mohou být i ve fyzické formě (např. na DVD).

Mikrotransakce, Free-to-play, Pay-to-win, lootboxy jsou formy zpoplatnění zamčeného herního obsahu za reálné peníze. Mikrotransakce (tedy malé platby) původně vycházející z free-to-play (F2P) her, což jsou hry distribuované zdarma, se postupně dostávají i do plně zpoplatněných her, což vzbuzuje určitou míru kritiky. Daleko kontroverznější forma mikrotransakcí jsou praktiky Pay-to-win (P2W), jenž umožňují získání výhody platícím hráčům nad neplatícími, což vytvoří značně nefér hru více hráčů. Kontroverzní jsou i lootboxy, což jsou zpoplatněné balíčky náhodných virtuálních předmětů, které však neregulovaně podporují chování podobné při závislosti na hazardních hrách. V některých státech tak již lootboxy byly v jisté formě omezeny nebo dokonce zakázány.

NPC, AI, herní předměty, inventář jsou pojmy související se samotným hraním (zejména RPG her). NPC neboli non-player character je každá herní postava, jenž není ovládaná hráčem, ale AI. Tato Artificial Intelligence (umělá inteligence) je v herním slangu široký pojem, znamenající jakoukoliv počítačem řízenou interakci NPC k hráči. Herní předměty jsou pak virtuální předměty, se kterými může hráč nějakým způsobem manipulovat. Inventář je osobní úložiště herní postavy (či nějakého herního předmětu) pro uskladnění těchto předmětů. [1] [2]

XP, level, skilly, atributy, buffy tyto pojmy se používají zejména u RPG žánru her a znázorňují postup a možnosti hráče, jeho vývoj (vylepšování) v průběhu hry. Po získání dosaženého množství XP neboli experience points (zkušenostních bodů) se hráči přidělí nový level nebo LVL (úroveň). Obě tyto jednotky jsou měřítka vývoje hráčské postavy, resp. hráčského postupu hrou. Díky nim si poté hráč může vylepšit určité skilly (dovednosti) či atributy (vlastnosti) jeho herní postavy, a tím si buď usnadnit další postup hrou (zvětšení statistik, parametrů herní postavy) či získat přístup k nové herní mechanice. Buffy (kladný efekt), popřípadě debuffy (záporný efekt) představují dočasnou změnu statistik herní postavy.

1.2 Herní vývoj

Cheaty neboli cheat kódy jsou kódy ve formě nějakého pořadí písmen či stisknutí kláves (tlačítek), jenž způsobí aktivaci či úpravu nějaké konkrétní (skryté) funkce či parametru ve hře. Cheaty jsou do her implementovány jejími vývojáři za účelem usnadnění práce při debuggingu. Mohou tak např. dát hráči nesmrtelnost, neomezené zdroje, herní předměty či urychlení postupu hrou. Cheaty jsou v některých hrách pro svou potenciální alternativní zábavnou zkušenost ponechány úmyslně. Naopak ve hrách více hráčů je jakékoliv nabourání pravidel tvrdě trestáno.

Herní bugy, patche, glitche, FPS, optimalizace jsou pojmy týkající se technické stránky her. Hry, jakožto velmi komplexní SW systémy, jsou na herní bugy neboli chyby ve hře obzvláště náchylné. Patche (záplaty) pak tyto chyby po vydání hry opravují. Glitche jsou specifické chyby u her, které způsobují krátkodobou závadu, jenž je často audiovizuálního charakteru, popřípadě simulace herního světa (fyziky). FPS neboli frames per second (snímky za sekundu) je údaj o počtu snímků, které hra (herní engine) poskytuje hráči jako vizuální odezvu, za jednu sekundu. Pokud je hra špatně či není optimalizovaná, což je proces ladění hry pro získání lepšího výkonu, pak může být počet FPS nízký a hra tak může být díky tomu nehratelná.

Roadmapa a development hell jsou pojmy týkající se stavu vyvíjené hry. Roadmapa je velmi hrubý plán vývoje, který naznačuje plánované větší herní mechaniky, větší verze hry nebo jiné její milníky (Beta fáze, vydání, vydání DLC apod.). A to včetně hrubého časového odhadu jejich dokončení. Jednoduché a veřejně přístupné roadmapy mohou sloužit i pro marketingové účely. Development hell je neformální označení stavu vývoje, kdy se vývoj projektu neposouvá kupředu, je zavalen problémy, nestíhají se termíny, nastávají příliš časté změny v přístupu k projektu (vedení, vývojářů, technologií) a jeho dokončení

je v nedohlednu. Pokud je práce na takovémto projektu dokončena, bývá jeho kvalita často rozporuplná.

Assety, sprity, pixel art, HUD jsou pro herní vývoj důležité audiovizuální pojmy. Asset je obecný název pro jakýkoliv herní objekt využívaný při herním vývoji. Assety jsou jakákoliv média či data a může se tak označovat jakýkoliv 2D obrázek (sprite), 3D model, textura, materiál, animace, zvuk, skript a další objekty, ze kterých se hra skládá. Většinou se tato data (soubory) importují do nějakého herního enginu. Zvláštním typem assetů jsou pak nativní (předpřipravené) asssety herního enginu, jako např. světlo, kamera, tilemapa apod. Jako sprity se označují většinou malé 2D obrázky (mohou být i animované), z nichž se poté skládají větší celky. Ať už významem zasazení několika spritů do velké scény, nebo více obdobných spritů vytvářející dojem pohybu (animaci) či otáčení objektu (využití ve 3D grafice u billboardingu).

Častým grafickým stylem pro tvorbu spritů je tzv. pixel art. Pixel art je unikátní svým vizuálním stylem, kde jednotlivé pixely slouží jako stavební kameny, z nichž se skládá obraz. Má tak i v dnešní době připomínat grafiku starých her s nízkým rozlišením z období 8 a 16 bitových počítačů a herních konzolí. HUD je metoda zobrazování pro hráče důležitých informací skrze GUI neboli grafického uživatelského rozhraní. V některých hrách se použití HUD prvků snaží minimalizovat pro lepší ponoření se do hry. Ve většině her však HUD zobrazuje alespoň základní informace o hráčově postavě, průběhu hry, nebo potřebné ovládací a pomocné prvky. [1] [2]

1.3 Pojmy SW vývoje a herní enginy

IDE, verzování, API a frameworky jsou pojmy spojené zejména s programováním. IDE neboli česky vývojové prostředí je SW určený pro usnadnění práce vývojářů při vývoji. Většinou obsahuje editor zdrojového kódu, debugger, případně podporu pro verzovací systém, kompilátor (nebo interpret) a další specificky určené nástroje. Verzovací systémy se používají k verzování, tedy sledování změn, např. v souborech zdrojového kódu. API je obecně komunikační rozhraní určené k nabízení SW služeb jinému SW. Je to sbírka procedur, funkcí, tříd, či protokolů, které může programátor využívat ve svém vlastním SW. Framework je SW struktura sloužící k podpoře vývoje SW projektů. Může obsahovat podpůrné nástroje, programy, knihovní API a často také doporučené postupy, dokumentaci či návrhové vzory.

Middleware je jakýkoliv specializovaný SW poskytující ostatnímu SW služby nad rámec služeb poskytovaných operačním systémem. Někdy jsou jako middleware označovány např. i celé herní enginy, někdy pouze některé jejich částí či technologie. [2]

Skripty jsou většinou drobné programy, které jsou napsány v libovolném skriptovacím jazyce a často jsou od zbytku projektu izolovány tak, že je možná jejich změna i bez nutnosti rekompilovat zbytek projektu. Skripty, použité při tvorbě hry v herním enginu, umožňují vytvářet vlastní komponenty či systémy (např. AI), obsluhovat herní mechaniky a události, v reálném čase měnit vlastnosti komponent a reagovat na uživatelský vstup.

Herní engine je specializovaný typ SW někdy zjednodušeně označovaný pouze jako framework či middleware, jenž poskytuje herním vývojářům technologické základy pro vývoj videoher. Herní enginy (někdy překládané jako herní jádra, nebo herní motory) jsou většinou komplexní sada modulů, vývojových nástrojů a API umožňující efektivní (vizuální) vývoj a podporu znovu použitelnosti herních prvků, a to i díky zabudovanému editoru (či dokonce IDE). Různé enginy jsou různě funkcionálně rozsáhlé a obecně bohužel platí, že čím pokročilejší funkce herní engine nabízí, tím se stává méně univerzálním. Ty více univerzální nabízí také snadné portování (převedení) videoher na různé herní platformy s pouze drobnými nutnými úpravami.

Moduly (či subsystémy, middlewary) jenž herní enginy nabízí, poskytují pro herní vývoj univerzálně potřebné technologické služby jako např. 2D a 3D grafický renderer („vykreslovač“ využívající nějaké grafické API např. Vulkan). Dále simulaci fyziky, zpracování animací, správu uživatelského vstupu, zvukový systém, řízení umělé inteligence, řízení síťové komunikace, další funkce a technologie (jako správa paměti, dat, skriptů, využití dostupných zdrojů atp.) a sestavení (buildování) samotné hry. Využití těchto univerzálně použitelných modulů ve formě herního enginu tak výrazně ulehčuje, zlevňuje a unifikuje vývoj videoher i znovu použitelnost jejich částí v dalších projektech. Velké herní enginy jsou vyvíjeny zpravidla ve velkých společnostech, které je buď používají uzavřeně (proprietární herní enginy), nebo je nabízejí v komerční či nekomerční variantě ostatním vývojářům (některé i jako open source). [3] [4]

2 SPECIFIKA PROCESU HERNÍHO VÝVOJE

Proces vývoje videoher má mnoho specifických prvků, s nimiž je nutné počítat. Svou povahou může být velmi proměnlivý až chaotický, a proto je velmi důležité řízení projektu a komunikace v týmu. Videohry jsou SW s největší mírou uživatelské interakce, a proto také největší potřebou chápání uživatelské zkušenosti. Ta je nejdůležitější zejména při návrhu herního designu, aby hra dosáhla cíleného herního zážitku. Vývoj videoher může být dominován uměleckými či designovými vlivy, ale na druhou stranu také potřebou specifických technologií a pokročilého programování. Proto jsou herní vývojářské týmy velmi oborově rozmanité.

2.1 Herní vývojové požadavky

Možná nejtěžší částí procesu videoherního vývoje je stanovení základních požadavků a jejich následný design tak, aby dosáhl požadovaného herního zážitku, ale zároveň aby byl vývoj uskutečnitelný. Proto je nutné si hned na začátku vývoje přizpůsobit požadavky dostupným zdrojům. Navíc každá hra je rozsahem i náročností na vývoj jiná, a proto neexistuje univerzální návod, jak tuto zřejmě nejtěžší část vývoje udělat správně. Ale existují obecná doporučení.

Nejprve je důležité vytyčení pevných mantinelů projektu, to znamená vymezení požadavků volbou nějakého herního žánru a zaměření, nebo využití nějaké již vydané hry pro inspiraci. Z morálního a legálního hlediska by vyvíjená hra neměla pouze kopírovat cizí práci, ale nějak ji modifikovat, či obohatit. Není dobré dělat úplně odlišný design hry, který nikdo nepochopí, ale je nutná jistá míra vlastní invence pro vymezení se a vytvoření něčeho nového a zajímavého.

Je nutné zvolit si typ hry či její zaměření a také její primární části (např. jaké herní módy bude obsahovat). Bude hra obsahovat multiplayer neboli hru pro více hráčů? Jestliže ano, tak bude nutit hráče spíše ke kooperaci či kompetenci, anebo bude hra rozdělena na části PvP (hráči proti sobě) a PvE (hráči proti prostředí – proti NPC)? Dále je nutné si ujasnit cílený rozsah hry (kolik obsahu bude hráči nabízet a jakým způsobem). Bude to hra s lineárním, větveným, nebo volným postupem hrou a bude tak nabízet otevřený svět (open world)? Takovéto a další otázky je nutné si ihned na začátku vývoje zodpovědět a vyvodit z nich další postupy.

Poté je nezbytné určit cílovou skupinu hráčů a její vlastnosti: věk, pohlaví, jazyk, hráčské schopnosti (herní obtížnost), kulturní aspekty atd. Výsledné působení hry je možné formovat aplikováním některých sociologických a psychologických poznatků. Např. analýza motivace

proč hráč hru hraje, je jedním se základních kamenů, podle kterých by se vývoj měl řídit. I proto jsou u velkých vývojářských společností součástí vývojového týmu odborníci z již zmíněných sociologických a psychologických oborů.

Důležité je také určit cílové HW a SW platformy, na které bude hra vyvíjena, popř. portována (převedena). A to z důvodů designu, náročnosti na HW, zpracování ovládání, podpory HW periferií a dalších pokročilých funkcionalit jako je např. podpora modování (uživatelských úprav hry, tzv. modů[1]). Nakonec je nutné pro vývoj získat zdroje: časové, finanční a lidské. Pro časové je nutné vytvořit odpovídající časový plán. Pro finanční je možné oslovení investorů, či vydavatelů. A pro lidské zdroje je nutné buď zainvestovat do pracovních pozic nebo využít outsourcing (vykonání práce prostřednictvím externích dodavatelů) některých částí projektu. [5]

2.2 Herní design

Po celou dobu hledání základních herních požadavků je nutné myslet na herní design. Herní design (game design, herní návrh) je velmi složité a rozsáhlé téma. Herní design navrhuje, jak se hra bude hrát a o čem vlastně bude. Jinak řečeno navrhuje, jak bude vypadat její obsah, cíle a pravidla, resp. hratelnost, prostředí, příběh a postavy. Herní design je *„proces vytváření cílů, kterých se hráči cítí být motivováni dosáhnout, a pravidel, kterými se hráč při pronásledování cílů musí řídit“*. [6] Herní design bývá podrobně popsán v Game design dokumentu (GDD). Hratelnost (gameplay) je postup definovaný pravidly hry, spojením mezi hráčem a hrou, výzvami a jejich překonáváním, dějem a hráčovým vztahem k němu. Definuje, jak se hra hraje.

Jak již bylo naznačeno, pro dobrý herní design je důležité porozumět, proč by hráč měl hru hrát. Motivací může být touha po dosažení cíle, po získání nových vědomostí či schopností, po porovnání se s ostatními hráči, po emocích či po relaxaci, po sociální interakci nebo naopak po sociální izolaci. Motivací může být mnoho a dobrý herní design by toho měl využít tak, aby chtěl hráč dobrovolně se hrou strávit svůj volný čas. U primárně ziskově navrhovaných her (hry jako služba), kde platí, že čím déle hráč hru hraje, tím je větší pravděpodobnost, že v ní utratí nějaké peníze, je tlak na hráčovu motivaci maximalizován. [7]

Mezi základní poučky herního designu patří, kromě empatie k hráči a naslouchání jeho zpětné vazby, také stav ponoření se do hry. Ten nastává tehdy, kdy hráč kvůli příběhu, dosažení cíle, hernímu světu, herním nebo sociálním interakcím, zkrátka kvůli konstantnímu

[1] Slovo „mod“ je odvozeno od slova modifikace

toku podmětů (flow) k hráči, nechce hru pozastavit nebo ji dokonce vypnout. Stavu ponoření naopak škodí příliš lehké či náročné úkony, opakování úkonů (grind), zpomalení hratelnosti (hráč dokončil cíl, je znevýhodněn, prochází GUI hry, herní postava zemřela), hra není spravedlivá, viditelné limitace hry, špatně napsaný příběh či dialogy, „hloupá“ AI, špatný audiovizuální či technický stav, dlouhé načítání a ukládání stavu hry, jakákoliv náhlá změna a jakýkoliv obsah jenž nějak vybočuje.

Alespoň částečného stavu ponoření (udržování flow) by mělo být dosáhnuto samostatně pomocí dobře navrženého tzv. gameplay-loopu (hratelnostní smyčky). Nejhorší, co by se hráči mohlo ve hře stát, je to, že by se tempo hry zastavilo (např. hráč musí čekat) a hráč by se tak začal nudit. Proto je mimořádně důležité, aby měl hráč v jakýkoliv moment hry co dělat a k tomu právě slouží gameplay-loop. Je to cyklus akcí, které je ve hře možné dělat stále dokola (nebezpečí nezáživného grindu) a tím se posunout kupředu. Obvykle je tato smyčka navržena tak, aby u uživatele vyvolala neurochemickou odměnu (např. uvolnění dopaminu) tedy zadostiučinění, uspokojení či radost z pokroku. Je to základ návrhu hratelnosti (někdy označován jako core gameplay). Příkladem může být cyklický model: výzva (zabij monstrum), odměna (získání zlata), očekávání (nákup nového vybavení) a uvolnění dopaminu. [8]

Na začátku hry obvykle bývá návod (tutoriál) jak hru hrát. Měl by vysvětlit základní ovládání a hratelnost, ale neměl by být až příliš podrobný, čímž by hráče od hry odradil. Průběhem hry by se obtížnost výzev měla mírně zvyšovat, ale vždy by měla být pro hráče zvládnutelná tak, aby ho hra nezačala frustrovat. Je dobré mít ve hře milníky (např. extra silné nepřátele, tzv. bossy), po jejich překonání se ukončí jedna část hry a započne jiná, čímž hráč pocítí uvolnění napětí a radost z nové etapy. Hráč by vždy měl mít dlouhodobé, střednědobé a krátkodobé cíle. Kromě povinných cílů, by hra měla hráči nabídnout i volitelné cíle a také by měla hráče podněcovat k samostatnému vytváření cílů a objevování hry (herního světa). Také by hráč měl vždy přesně vědět, kde se nachází a proč dělá to, co dělá. Chabý, ale nejjednodušší design úkolů (cílů) či náhodných událostí (eventů) je např. „dones tohle támhle“ (tzv. fetch questy).

Mezi další základní poučky herního designu patří neposouvat hráče v postupu hrou příliš dozadu, ale také ne dopředu (např. prozrazení hádanky je až ta poslední možnost). Hráč by měl cítit (a vidět) důsledky svých činů ve hře. Měl by se také zlepšovat společně

se svou virtuální postavou. Čím více interakcí, tím lépe. Nadsazeně by se dalo říci, že každý vstup od hráče by měl ve hře mít odpovídající zpětnou vazbu (alespoň jednoduchý zvuk).

Velmi důležité je dát hráči volnost v hraní (různé možnosti, cesty), avšak stále jej udržovat na správné cestě pro postup hrou. Dále je důležité pozvolna uvolňovat nové, neznámé, zajímavé a překvapivé věci a také vizuálně znázorňovat postup a příběh hry, čímž se prohloubí hráčova zkušenost a ponoření do hry. Zároveň je důležité hráče nepřehlcovat (např. velkým množstvím ovládacích prvků, hromadou textu i úkolů) a urychlit mu nedůležité prvky hry (např. přednastavením nedůležitých hodnot) či možnost přeskočení úvodních videí a prohlášení, tutoriálu, dialogů, dlouhých animací a cutscene (do hry zakomponovaných videí). [5], [9]

2.3 Herní studio

Společnosti zabývající se vývojem videoher se nazývají herní studia. Ty se liší jak svou velikostí, tak přístupem k vývoji a firemní filosofií. V případě velkých herních studií, vyvíjejících více projektů zároveň, jsou jejich zaměstnanci po čas jejich vývoje rozděleni do vývojových týmu. Každý tým je poté dále rozdělen na části dle jejich vývojového zaměření a jednotlivým vývojářům jsou přiděleny konkrétní vývojové role. Většina firem má jasnou hierarchickou strukturu zaměstnanců, ale existují i studia s velmi plochým systémem řízení vývoje. Kromě velkých herních studií existují také malá, či dokonce jednočlenná studia, kde se rozdělení na týmy a role smazává a takovýto herní vývojáři musí být dostatečně multifunkční. Jednotlivé tvůrčí části vývojových týmů a jejich role jsou popsány v následujících odstavcích. [9]

2.3.1 Designová část

Herní designéři se při návrhu hry musí řídit již zmíněnými (viz. podkapitola 2.2) a také dalšími poučkami herního designu. Jejich kreativní forma práce vyžaduje určitou míru volnosti a často zahrnuje techniky, jako např. brainstorming, různé druhy analýz, prototypování apod. Jejich prací je návrh de-facto všech prvků hry, a přitom musí zachovat vše jednoduché (dodržování principu KISS), intuitivní a konzistentní, čímž vytvoří požadovaný a ucelený herní zážitek. Herní designeři jsou u větších projektů a studií děleni dle konkrétních zaměření.

Vedoucí (lead, creative director) designér koordinuje práci a komunikaci ostatních designérů a členů týmu. Zajišťuje konzistentnost a dokumentaci herního obsahu. Často činí

důležitá designová rozhodnutí a prezentuje je i mimo vývojový tým. Někdy také sám vytváří základní téma a zasazení hry, popř. i pozadí děje. Měl by být umělecky a technický zdatný.

Narrative designeři jsou v podstatě scénáristé, kteří vytváří herní příběh a dialogy. Kromě toho se také snaží nalézt způsob, jak je pro hráče prezentovat a spojit je s herními prvky a hratelností.

Level designeři vytváří za pomocí herních editorů či jiných nástrojů virtuální prostředí (herní svět, lokace, plán – level) tak, aby hráče směřovalo ke správnému postupu hrou, nezdržovalo jeho průchod hrou (zkratky, fast travel mechaniky), podněcovalo touhu po objevování (skrytý a bonusový obsah neboli easter eggs) a sloužilo pro navržený herní obsah.

Content designeři vytváří herní obsah, jako jsou postavy, předměty, úkoly a cíle. S nimi souvisí systém designeři, kteří mají na starost návrh herních pravidel a vlastností předmětů a systémů.

Mezi další designery patří **audio designéři**, zaměřující se na zvuky, hudbu a dabing. Dále **designeři rozhraní**, kteří určují rozložení, obsah, navigaci a použitelnost funkcí herního rozhraní (HUD a celkové GUI). **Techničtí designeři** slouží jako most mezi designery a programátory.

2.3.2 Umělecká část

Umělecká část týmu úzce spolupracuje s výše zmíněnými designery. Zahrnuje role jako **vedoucí umělců (lead artist, art director)**, který řídí umělecký tým, ale také budoucí podobu a styl vyvíjené hry a určuje tak jaké pocity a náladu má hráč při pohledu na hru cítit.

S herním designem velmi souvisí práce **konceptových umělců**. Ti vytváří náčrty prostředí, předmětů a postav. Jejich práce je využita pro vizuální představu navrhovaného designu a příběhu (storyboardy) a dále použita pro dokumentaci, vyjednávání s partnery a marketingové účely.

2D umělci jsou široká skupina umělců vytvářející finální 2D grafiku, zahrnující vytváření textur, (souborů) spritů, grafiku pro GUI a jiné grafické prvky. **3D umělci (modeláři)** modelují 3D objekty a provádí další s nimi související práce.

Animátoři jsou velmi důležitá součást uměleckého týmu. Kromě samotných animací se podílí na zpracování cutscene, motion capture a vizuálních efektů. I v uměleckém týmu je potřeba spolupráce s programátory, kterou zajišťují techničtí umělci.

2.3.3 Audio část

Audio část týmu se skládá z vývojářů zaměřených na audio složku hry jak z uměleckého, tak technického pohledu. **Vedoucí (audio director)** má obdobnou zodpovědnost jako vedoucí umělců. **Skladatelé** vytváří, popřípadě řídí tvorbu či licencuji hudbu pro hru. **Zvukový specialisté** se starají o návrh či implementaci zvuků, zvukových efektů a ambientního ozvučení. **Dabing specialisté** se pak starají o kvalitní zpracování dabingu a vedou dabingové herce.

2.3.4 Kódovací část

Kódovací část složená primárně z programátorů pak dává všechny vývojové části dohromady. **Vedoucí (technik director, lead) programátor** řídí programátorský tým a určuje nástroje, HW a standardy použité při vývoji. **Síťový programátor** zajišťuje kvalitu síťové architektury, databází a dalších online prvků potřebných primárně ve hrách pro více hráčů. **Engine programátoři** jsou potřební v případě úprav enginu či tvorby vlastního. Vývoj ostatních vývojových nástrojů mají na starost **nástrojoví programátoři**, kteří úzce spolupracují se zbytkem studia.

Programátoři umělé inteligence vytváří systémy pro řízení NPC, stavové diagramy nebo pokročilé behaviorální stromy, určování strategie, hledání cest a další systémy, jenž řídí chování herních objektů. **Podpůrní programátoři** a **skriptéři** programují herní postavy, systémy, mechaniky, události, akce a cíle. Ve velkých herních studiích mohou být reprezentovány další zaměření jako jsou **audiovizuální programátoři**, **programátoři rozhraní**, **programátoři fyziky** a **programátoři dalších middleware systémů**.

2.3.5 Testovací část

Testovací část týmu se zabývá jak testováním, tak i kvalitou hry (quality assurance – QA). Má na starost udržení kvality výsledného produktu. **Vedoucí (lead) tester** má za úkol řídit ostatní testery, rozhodovat nad důležitostí nalezení chyby a vyhodnocovat, zda je vše provedeno včas a kvalitně. Zároveň také hledá různé nekonzistence vůči dokumentaci. **Compatibility testeři** většinou nejsou přímo součástí studia. Testují, zda je hra na cíleném SW a HW (různé PC, konzole, periferie atd.) hratelná v odpovídající kvalitě.

Produkční, QA a regresní testeři provádí největší část interního testování všeho druhu. **Beta testeři** jsou často dobrovolní uživatelé, kteří nejenže můžou nalézt často ukryté chyby, ale také dávají uživatelskou zpětnou vazbu ohledně hratelnosti, zábavnosti a stabilitě hry ještě

před jejím vydáním. **Focus testeři** jsou lidé se hrou neobeznámeni a poskytují svoji zpětnou vazbu zejména pro plánování marketingu.

Kromě těchto tvůrčích týmů jsou součástí herních studií také marketingové, producentské týmy a ostatní zaměstnanci jako jsou legální, finanční a často také různí odborní poradci (psychologové, historici, architekti, futuristi atd.). [9]

3 MODELY A METODIKY VÝVOJOVÉHO PROCESU

Vývoj větších softwarových projektů je z hlediska organizace práce a návrhu velmi obtížný. Specificky právě u velkých herních projektů je řízení jejich vývoje o to složitější, zejména kvůli organizaci různorodých kreativních a uměleckých disciplín. Problémům, kterým musí producenti (resp. manažeři) při řízení velkých projektů (týmů) čelit, je mnoho. Od největších problémů týkajících se nedostatku rozpočtu, času, kvality, až po zdánlivě triviální, jako jsou komunikační problémy, špatné plánování, nízká produktivita či morálka týmu. Zvládnutí řízení vývojového procesu a včasné vyřešení vzniklých problémů je pro úspěšné vydání hry klíčové.

Proto se při vývoji využívají různé vývojové modely a metodiky, jenž předepisují organizaci práce a způsoby řízení postupu vývoje tak, aby co nejlépe odpovídaly specifickým potřebám vývoje projektu a týmu, a tím předcházeli vzniku již zmíněných problémů. Některé modely či metodiky však nemusí být pro konkrétní vývojové týmy a jejich herní projekty výhodné. Proto je důležité chápat jednotlivé rozdíly mezi nimi a poté si zvolit jeden takový, jenž bude pro vývoj opravdu přínosný.

3.1 Vodopádový vývojový model

Je jedním z nejstarších modelů a pro moderní vývoj her není příliš vhodný. Model se skládá z několika fází, které na sebe sekvenčně navazují tak, že po dokončení jedné fáze se přechází na další a takhle postupně až do konce poslední fáze, kde je produkt hotov. Jednotlivé fáze se tak nikdy neopakují. Některé verze modelu mají rozdílné definice fází, ale obecně lze říci, že se ve vodopádovém modelu nachází tyto fáze: analýza požadavků, návrh (design), implementace, verifikace a údržba.

Jelikož se definice všech požadavků a funkcí tvoří pouze na samotném začátku vývoje (kdy ještě nemusí být zcela známy), tak je tento model vhodný spíše pro projekty, u kterých je jejich vývoj předem jasně daný, neměnný a se kterým má tým již předchozí zkušenosti. V mírně chaotickém herním vývoji se proto dá použít, tak akorát na vývoj nějakého ne příliš inovativního pokračování, DLC, remasteru či remaku.

Existuje také modifikovaný vodopádový model, který umožňuje mírné prolínání fází. To vývoj zefektivňuje a umožňuje to částečnou paralelizaci jednotlivých, na sobě nezávislých částí vývoje. Takovéto řízení projektu může být náročné zejména z důvodu nevykonání práce, která by po dokončení předešlé fáze přišla vniveč. Tento vylepšený vodopádový model může najít uplatnění zejména u méně designově komplexních her, jako jsou některé adventury. [8]

3.2 Iterativní vývojový model

Tento model často zahrnující pojmy, jako prototypování nebo agilní vývoj, je na rozdíl od vodopádového modelu více vhodný pro vývoj většiny videoher. Model zahrnuje pouze tři fáze: design, prototyp a vyhodnocení. Avšak jakmile jsou tyto tři fáze provedeny, tak začíná tzv. nová iterace a jednotlivé fáze vývoje se opakují. Důležité je vyhodnocení a následná úprava designu hry na základě aktuálně dostupného prototypu. Celý tento proces se opakuje do doby, než se z prototypu stane kompletní hotový produkt s požadovanou úrovní kvality. Tento přístup také umožňuje zapojení vydavatele a zákazníků do vývoje ve fázi vyhodnocení (např. formou předběžného přístupu ke hře, resp. prototypu), a tím získání důležité zpětné vazby. Na iterativní model vzniklo velké množství různých metodik a jeho modifikací. [8] [9]

Jednou z těchto modifikací je Inkrementální vývojový model. Celý projekt je rozdělen na menší individuální části, které se poté vyvíjí odděleně (někdy i za použití lineárního vodopádového modelu) na základě jejich vysoké priority a poté jsou na ně postupně navazovány části s menší a menší prioritou. Výhodou toho přístupu je, že po každé dokončené části je možnost získání zpětné vazby, a tím možnost pozitivního ovlivnění vývoje další části. Nevýhodou může být přílišná fragmentace projektu. U vývoje her to může znamenat, že zatímco např. části assetů jsou vytvářeny na základě vodopádového přístupu, tak designové části jsou vyvíjeny spíše iterativně.

Ještě více flexibilní přístup vycházející z iterativního modelu nabízí Agilní model vývoje. Je vhodný pro vývoj většiny dnešních her, kdy požadavky nejsou na počátku jejich vývoje známy, kdy je nutné vytvářet nové technologie a kdy je velké zapojení zákazníků i vydavatele do vývojového procesu. Model agilního vývoje je podobný vodopádovému modelu, avšak s tím rozdílem, že reakce na zpětnou vazbu může nastat v jakékoliv fázi vývoje a v jakémkoliv rozsahu. Tyto změny se tak po vzoru iterativního modelu projevují v další iteraci vývoje, avšak tyto iterace jsou u některých agilních metodik uskutečňovány mnohem častěji (v rámci týdnů či dnů) a je uplatňován daleko menší důraz na byrokracii při plánování.

Agilní vývoj naopak není vhodný u projektů, u kterých jsou předem známy požadavky a jsou při jejich vývoji použity dobře známé technologie (pokračování, DLC, …). Agilní model řízení projektu nemusí být vhodný pro malé a nezkušené vývojářské týmy. A to zejména z důvodu větší potřeby komunikace a také svobody z hlediska zavedení jednotlivých fází vývoje oproti např. vodopádovému modelu, a tím pádem také velmi obtížné predikování budoucího postupu vývoje, čímž mohou nastat finanční či časové problémy. [10], [11]

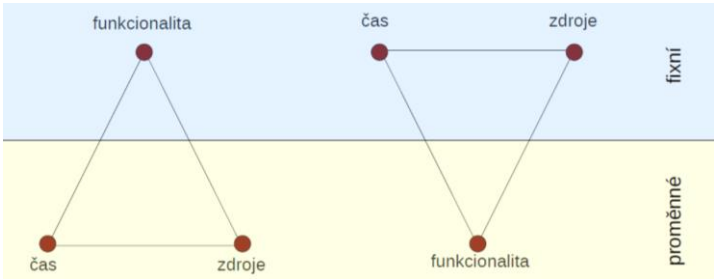
Existuje mnoho různých přístupů, jaké modely je možné při vývoji SW (her) využívat, jakými metodikami se řídit nebo jaké praktiky využívat. Mezi doposud nezmíněné modely patří například V model, Spirálový model, Cleanroom model a další. Tyto modely popisují spíše posloupnost činností vývoje a snaží se popsat obecný způsob, jak k řízení vývoje přistupovat. Neobsahují žádné konkrétní definice, jak jednotlivé činnosti provádět a také nijak nereflektují konkrétní potřeby vývoje herního projektu a týmu. Pro tento účel vnikly různé metodiky, jenž tyto modely, a tedy i celý vývojový proces konkretizují.

3.3 Vývojové metodiky

Některé metodiky jsou spíše doporučení, naopak jiné určují dokonce i role vývojářů, jejich úkoly, odpovědnosti, pravidla komunikace v týmu apod. Při výběru vhodné metodiky je tak vždy důležité, vyhodnotit náročnost (potřeby) projektu a dostupné zdroje vývoje jako jsou čas, finance, lidské zdroje, a především vývojové zkušenosti. Jestliže je tým malý (nebo dokonce jednočlenný), nezkušený a na vývoj není mnoho času či finančních zdrojů může nasazení velmi striktní metodiky spíše uškodit. Uškodit může však také nasazení příliš liberální metodiky, a to především v případě velkého týmu s dostatkem finančních a nedostatkem časových zdrojů. [12]

Jestliže je pro konkrétní projekt zvolena vyhovující metodika, tak je možné tuto metodiku pod vedením zkušeného producenta v počátečních fázích vývoje optimalizovat, např. změnou striktnosti jejího dodržování či ji mírně modifikovat. Jestliže se v průběhu vývoje zjistí, že zvolená metodika vývojářům nevyhovuje, musí se pod dohledem producenta přístup k vývoji změnit. Avšak v tomto případě nastává riziko nezvládnutí řízení a uvržení vývoje do tzv. development hell, který může v krajním případě končit restartem či zrušením projektu.

Metodiky se kromě jejich striktnosti (konkrétnosti) mohou dělit i podle přístupu k základním složkám vývoje na klasické (neagilní či rigorózní) a agilní. Zatímco klasické metodiky považují za fixní složku funkcionalitu vyvíjeného softwaru, tak čas společně se zdroji považuje za proměnné složky. U agilních metodik to bývá naopak (viz. obr. 1). [11]



Obrázek 1 - Dělení metodik: vlevo klasické a vpravo agilní [13]

3.3.1 Rigorózní metodiky vývoje

Klasické (rigorózní) metodiky kladou větší důraz na plánování vývoje a díky tomu přináší do vývoje SW mnoho pozitiv. Většinou se vyznačují potřebou definice funkčních požadavků, což u kancelářského SW není problém, avšak při občas chaotickém herním vývoji už to problém být může. Některé z těchto metodik čistě opisují pravidla řízení vodopádového, spirálového či jiného modelu a případně jej pouze mírně upravují, jako např. u vodopádovém modelu je takto navržena metodika Structured Systems Analysis and Design Method (SSADM). Jiné metodiky rozšiřují modely více, jako např. u iterativního modelu často používané metodiky Unified Process a Rational Unified Process (RUP).

Unified Process využívá modelovací jazyk UML. Organizuje vývoj do pěti základních aktivit: stanovení požadavků, analýza, návrh, implementace a testování. Každá tato aktivita pak prochází čtyřmi fázemi (přičemž v každé fázi může proběhnout jedna nebo více iterací): zahajovací fáze, fáze rozpracování, fáze konstrukce a fáze zavedení. Každá fáze je jinak pracovně náročná a všechny jsou zakončeny vyhodnocovacím milníkem pro úspěšný přechod do další fáze. Existuje mnoho nástrojů, jenž usnadňují implementaci Unified Process. Jednou z nejvíce využívaných je Rational Unified Process (RUP).

RUP je komerční implementace metodiky Unified Process, aktuálně vyvíjená dceřinou firmou společnosti IBM. RUP předepisuje procesy, role, činnosti a další prvky vývoje, avšak není příliš striktní a lze ji použít spíše jako framework, který lze upravit ke specifickým potřebám vývoje. Vychází z šesti základních osvědčených praktik pro vývoj: iterativní vývoj, aktivní správa požadavků, komponentová architektura, vizuální modelování, ověřování kvality software a řízení změn. Definuje základní stavební bloky a elementy vývoje, které popisují cíle, potřebné dovednosti a podrobné vysvětlení, jak vývojových cílů dosáhnout. RUP je robustní metodika s propracovanou dokumentací, a proto může být zvládnutí jejího správného nasazení pro menší vývojové týmy příliš složité. [12]

Za zmínku ještě stojí hybridy mezi vodopádovou a agilní metodikou vývoje např. „Agifall“ nebo „WAgile“. U Agifall není nutné čekat na dokončení předchozích fází, aby bylo možné přejít k další fázi a zároveň je nutné rozsáhlé plánování, výzkum a strategie, jako je tomu u vodopádového přístupu. Ale zároveň je zde větší míra flexibility a přijímání změn tak, jako je tomu u agilního přístupu. [14]

3.3.2 Agilní metodiky vývoje

V posledních letech zřejmě nejoblíbenějším přístupem k vývoji her jsou agilní metodiky. [11] Tyto metodiky nabízí v oblasti plánování (designu) projektu největší míru volnosti. Důležité je však dodržovat ucelenost jednotlivých iterací, tedy stále by měly probíhat všechny předem určené iterační fáze. Obecně se jedná o fáze: plánovaní, design, implementace, testování, nasazení a vyhodnocení. Zpětná vazba (včetně té od hráčů) a vzájemná komunikace je pro agilní vývoj klíčová stejně tak, jako schopnosti rychle řešit nastalé problémy, získávání potřebných zdrojů a přizpůsobení se změnám v projektu. Agilní metodiky byly zpopularizovány v roce 2001, kdy byl představen Manifest agilního vývoje.

Tento manifest definoval hodnoty a principy, jenž jsou základními kameny dnes nejrozšířenějších agilních metodik či frameworků jako jsou Scrum, Extrémní programování, Lean, Crystal a mnoho dalších. Manifest uvádí čtyři základní hodnoty (v citaci zvýrazněné) agilního vývoje:

„**Jednotlivci a interakce** před procesy a nástroji

Fungující software před vyčerpávající dokumentací

Spolupráce se zákazníkem před vyjednáváním o smlouvě

Reagování na změny před dodržováním plánu“ [15]

Velmi populární agilní metodikou je Scrum. Metodika je určena ideálně pro menší týmy, ty je tedy v případě velkých společností nutné předem vytvořit. Poté co jsou pro vyvíjený projekt navrhnuty požadavky a cíle, tak se ohodnotí a zařadí do projektového backlogu (seznamu). Z něj se poté vytváří menší úkoly, které budou jednotlivé týmy plnit, a které lze splnit v časově ohraničených iteracích tzv. sprintech. Časový rámec sprintů je většinou dlouhý 2 až 4 týdny. Na začátku každého sprintu probíhá plánování práce a vytvoření dalších úkolů. Jednotlivé týmy vyhodnocují dosažený pokrok na denně pořádaných patnáctiminutových mítincích tzv. denních scrumech.

V průběhu sprintu jednotlivé týmy řeší přidělené úkoly a komunikují o nastalých problémech. Na konci sprintu se mohou konat sprint review a sprint retrospective, na kterých se předvede odvedená práce, vyhodnotí se proběhlý sprint a získá se potřebná zpětná vazba pro další sprint (do projektového backlogu mohou být zařazeny nové důležitější požadavky a vyřazeny ty méně důležité). Poté je spuštěn nový sprint a takto pořád dokola, než je buď projektový backlog vyčerpán anebo je vývoj projektu ukončen. [12], [14]

Další agilní metodikou je Extrémní programování (XP). Tato metodika je vhodná spíše pro menší projekty a týmy (do 10 členů). Extrémní programování, jak již název napovídá, bere osvědčené vývojové principy a dotahuje je do extrému. XP je určeno pro velmi proměnlivý vývoj, který je však dobře řízený (komunikovaný) a u něhož je velký důraz na kooperaci, verzování a automatické testování. Tato metodika je velmi flexibilní a dalo by se říci, že eliminuje hranice mezi jednotlivými vývojovými fázemi. XP je charakteristické neustálou revizí kódu (včetně používaní párového programování, kdy dvojice programátorů pracuje na jednom kódu), klade důraz na testování a krátké iterace. Základní hodnoty XP jsou: komunikace, respekt, jednoduchost, zpětná vazba a odvaha dělat těžká rozhodnutí. XP je poněkud kontroverzní metodika, a kromě názorů o její prospěšností je také slyšet početná kritika jejích odpůrců. [16]

Zajímavým agilním přístupem k vývoji je Lean development. Původně vychází z principů používaných při výrobě produktů firmy Toyota. Je-li Lean development korektně aplikován, tak by měl zajistit efektivní, rychlou formu vývoje s kvalitním a nenákladným výsledným produktem. Předepisuje několik základních principů jako jsou: vytváření hodnoty pro zákazníka, eliminování zbytečností, optimalizace celku, posilování týmu a neustálé vylepšování. Doporučuje také užitečné praktiky, jakou je například Test-driven development. Z Leanu také vychází praktika Kanban (japonsky tabule), jejíž cílem je vizuálně (pomocí štítků na tabuli) demonstrovat vývojový proces. Kanban je často využíván i v ostatních agilních metodikách. [17] [18]

Mezi další agilní vývojové přístupy a praktiky řadíme: Crystal metodiky, Vývoj řízený vlastnostmi, Rapid application development, Agile Unified Process (AUP), DSDM, DevOps, Continuous integration, Continuous delivery, a také multi-týmové frameworky: LeSS, DAD, NEXUS, SAFe a další. Za zmínku ještě stojí částečně agilní metodiky: PSP, která je určena pro osobní řízení jednotlivců a TSP, jenž z něj vychází a dovoluje samo-organizaci jednotlivců v rámci týmu. Různých vývojových přístupů a metodik je opravdu mnoho, a není jednoduché zvolit si tu, jenž bude nejvíce optimální, a kterou se tým a vývoj projektu bude řídit. [12]

4 HERNÍ VÝVOJ V PRAXI

Průzkumy ukazují, že projekty vyvíjené pomocí vodopádového modelu jsou dnes na ústupu, a naopak převažují iterativní, respektive agilní vývojové modely a metodiky. I v porovnání s klasickými iterativními metodikami se v rámci produktivity herního vývoje zdají agilní metodiky jako jasný vítěz s průměrným přírůstkem produktivity o 79 %. Efektivita agilního vývoje je způsobena právě možností pravidelné kontroly kvality, stavu projektu, dostupných zdrojů a případné změny požadavků (funkcionality) či technologií. [19]

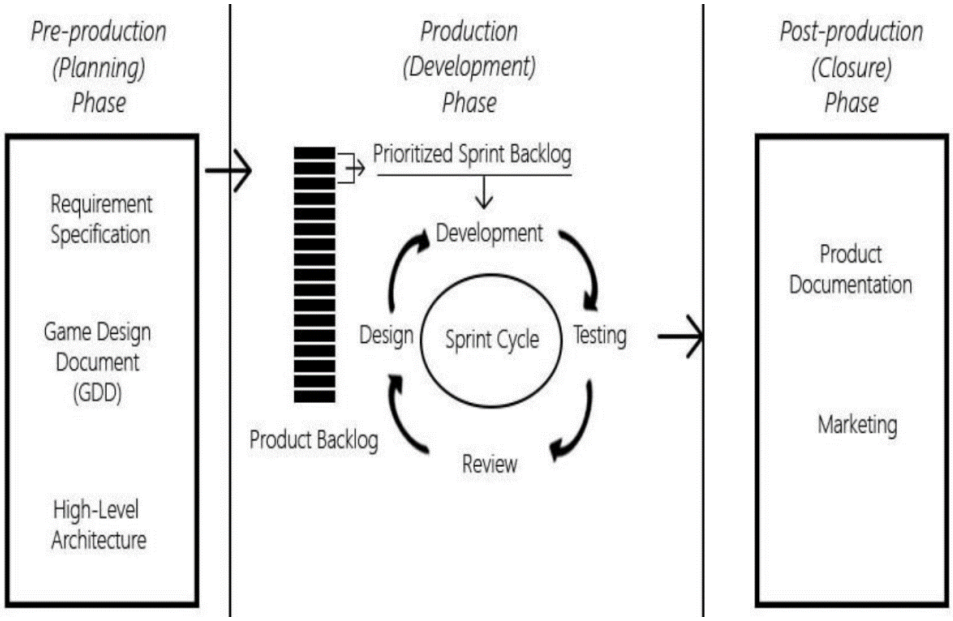
Právě v herním vývoji je poměrně běžné, že s vyčerpáním dostupných zdrojů, se zredukuje či dokonce úplně vypustí nějaká naplánovaná, ale již nedostatečně implementovaná funkcionalita a projekt se dokončí bez ní. Nebo naopak, že s dostatkem zdrojů se některé funkcionality přidělávají nad rámec původního návrhu. Takovéto změny právě nejlépe reflektují agilní metodiky.

Alternativní přístupy vývoje her, jež využívají modelovací jazyk UML jsou např. Model-driven development, který rozděluje projekt na tři části reprezentované class diagramy (Structure, Behavior, Control) a Component-based development, který rozděluje projekt na jednotlivé komponenty (grafika, GUI, zvuky, fyzikální middleware, …), což zaručuje velkou míru znovu použitelnosti. Mezi nejoblíbenější agilní metodiky patří Extrémní programování a Scrum. Extrémní programování je většinou realizováno pomocí programování v párech a zkombinováno s přístupem Test-driven development. Jedná se o velmi rychlou metodu programování se zajištěním vysoce kvalitního kódu.

Scrum (popř. modifikovaný Scrum) je dle různých průzkumů v aktuálním vývoji her nejpoužívanější agilní metodikou. Dle zkušeného project leadera a popularizátora Scrumu v procesu vývoje videoher Clintona Keitha, je Scrum pro vývoj her vhodný také proto, že podporuje vytváření unikátních a zábavných nápadů pro hru. Nasazení a plné využití potenciálu Scrumu nemusí být jednoduché (obzvlášť u menších týmů) a je dobré provést vývojáře školením, kurzem či hrou jakou je např. Playscrum. S narůstající popularitou začaly vznikat i různé jeho modifikace a nové metody. Pro herní vývoj výhodné Scrum metody jsou např.: Scrum of scrum, ABC sprint, Distributed Software development, sdPP + iGDD. [19]

V případě použití klasického Scrumu je jeho nasazení v procesu vývoje her znázorněné na následujícím obrázku číslo 2. Celý vývoj je rozdělen na tři hlavní fáze: Plánování, Vývoj, Závěrečná fáze. V plánovací fázi je důležité vytvoření požadavků a první verze GDD a následně začíná příprava architektury. Poté je ve vývojové části vytvořen projektový

backlog a následným utvořením Sprint backlogu startuje první z mnoha Sprintů. Ten je složen ze čtyř částí: design, vývoj, testování a vyhodnocení. Poté co je dosažena požadovaná kvalita, tak je v závěrečné fázi vytvořena dokumentace a zahájena hlavní marketingová kampaň. [19]



Obrázek 2 - Herní vývoj s využitím metodiky Scrum [19]

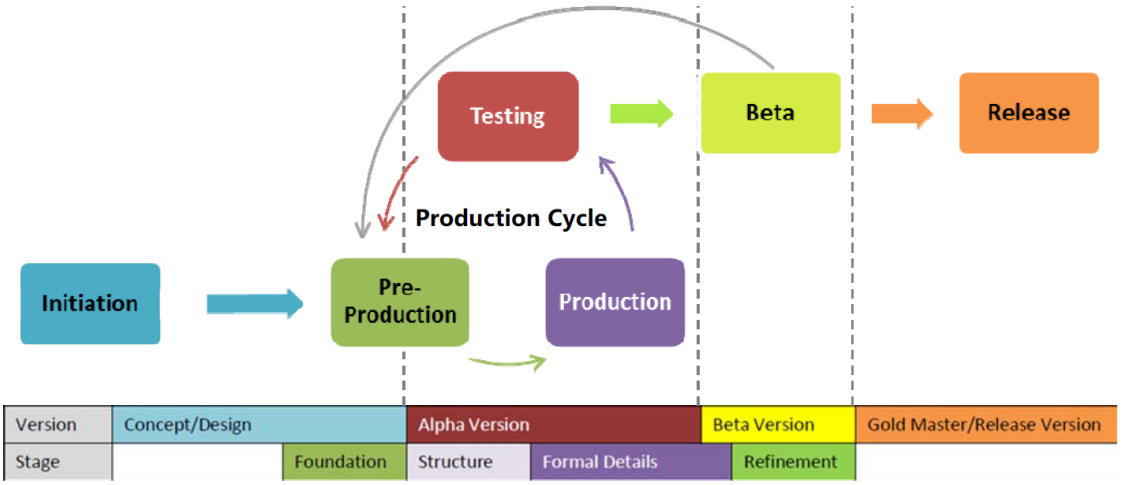
4.1 Herní vývojový cyklus (GDLC)

Ať už je na proces vývoje nasazena jakákoliv metodika, tak jeho celkový průběh, či rozdělení na fáze je obecně velmi podobné. Tento předpis průběhu vývoje nazýváme vývojový cyklus nebo životní cyklus (ang. life-cycle), pakliže obsahuje i fázi údržby. Vývojový cyklus dělí průběh vývoje do několika fází a zhruba naznačuje, které práce jednotliví vývojáři mají v dané fázi plnit či alespoň jaké výstupy by jednotlivé fáze měly mít. Jak již bylo řečeno několikrát, herní vývoj se od vývoje klasického software velmi liší, a z toho je také nutné vyvodit odlišnosti ve vývojovém cyklu.

I přesto, že je obecný průběh herního vývoje podobný, neznamená to však, že existuje pouze jeden univerzální herní vývojový cyklus (GDLC), jenž by dostatečně a přehledně pokryl vývoj všech herních projektů. Existuje mnoho různých forem GDLC. Za zmínku stojí např. tyto: Blitz Games Studios, Arnold Hendrick's, Doppler Interacttive, Heather Chandler's a Ramadan's & Widyani's. Poslední jmenovaný z nich však trochu vyniká.

Tento GDLC byl prezentován v roce 2013 na konferenci ICACSIS a vznikl na základě analýzy prvních čtyř jmenovaných GDLC. Tento cyklus znázorňuje více preferovaný iterativní přístup k vývoji a také byl v rámci výzkumu úspěšně otestován v praxi. Fáze tohoto

cyklu jsou v herním vývoji obecně považované za signifikantní a branné za jakýsi standart. Jak je vidět na obrázku č. 3, tak tento standartní GDLC dělí průběh herního vývoje na 6 významných fází: Zahájení, Předprodukce, Produkce, Testování, Beta a Vydání. [20]



Obrázek 3 - Iterativní GDLC [20]

4.1.1 Jednotlivé fáze iterativního GDLC

Vývojový proces začíná zahajovací fází, které se zpravidla účastní pouze designéři, koncept umělci, producenti a vedoucí některých oddělení. Jejich úkolem je vymyslet základní myšlenku projektu (High Concept), z ní poté vytvořit a sepsat koncept vyvíjené hry. Základní myšlenka (o rozsahu pár vět) by měla přesně vysvětlit o čem vyvíjená hra je. Popisuje unikátní koncept, díky kterému je hra pro hráče zajímavá a odlišná od konkurence. Dokáže být také užitečná při těžkých rozhodnutích v pozdějších fázích vývoje. V této fázi vývoje se také již definují primární designová rozhodnutí, vytváří se konceptuální ilustrace nebo se stanovují stěžejní prvky příběhu. [21]

Většinou se v **zahajovací fázi** také stanovují marketingové cíle, začíná jednání s vydavateli či investory, a právě pro ně jsou určeny Game proposal dokument a Koncept dokument. Game proposal je dvou stránkový „leták“ určený pro rychlé shrnutí Koncept dokumentu na business mítincích. A jeho primárním zaměřením by mělo být, proč bude vyvíjená hra úspěšná a jak vydělá peníze. Koncept dokument je profesionálně zpracovaný deseti až dvaceti stránkový dokument, který na první pohled zaujme vydavatele či investora. Obsahuje konceptuální ilustrace a měl by se skládat z těchto sekcí: základní myšlenka, žánr, hratelnost, vlastnosti hry, prostředí, příběh, cílová skupina, hardwarové platformy, odhadovaný harmonogram, rozpočet a P&L, analýza konkurence, tým, analýza rizik a shrnutí.

Po dokončení zahajovací fáze se vývojový cyklus přesouvá do **fáze předprodukční**, někdy nepřesně označované jako Proof-of-concept (PoC). V případě iterativního vývojového modelu je tato fáze součástí iterativního cyklu (tedy každé iterace). Hlavními cíli této fáze je ladění designu, ověření spolupráce s vydavatelem a ověření proveditelnosti návrhu jak z hlediska vstupu (tedy náročnosti na dostupné zdroje), tak i výstupu (jestli na základě prototypu návrh hry dává smysl) vývojového procesu (tedy PoC). Splnění těchto cílů lze dosáhnout prací na několika dalších (v jednotlivých iteracích proměnlivých) dokumentech.

Nejrozsáhlejším z nich (někdy i několik set stran) je Game design dokument. Ten má za cíl pro vývojáře popsat veškeré vstupy a výstupy ve vztahu hra-hráč. Jsou to veškeré procesy, systémy, vlastnosti, design, příběh, vizuál, zvuk atd. Prostě vše, co tu hru definuje a co je potřeba vývojáři vypracovat. Z něj dále vychází Art Production Plan, jež definuje vzhled hry a Technical Design Document, který přesně definuje SW, HW, infrastrukturu a technologie potřebné k vývoji. Další důležité dokumenty jsou zahrnuty do souboru plánů Project plan, jenž zajišťují především plánování práce, správu zdrojů a sledování postupu vývoje. Výstupem předprodukčních fází je funkční prototyp. To je kus SW, který zachycuje High Concept, a který dokazuje proveditelnost a funkčnost konceptu, tzv. „na vlastní oči“, což bývá při spolupráci s investory či vydavateli často to nejdůležitější.

Jednotlivé prototypy mohou být zaměřeny na různé vývojové oblasti. Zatímco v předprodukční fázi vyvinuté prototypy se zaměřovaly především na koncept, funkcionalitu a zábavu, tak v následující **produkční fázi** se programátoři zaměřují především na kompletní mechaniky, působivost, vzhled, výkonnost a odstraňování bugů. Umělecký tým vytváří herní modely, animace, textury a jiné grafické prvky. Tým zabývající se audiem vytváří zvuky, hudbu a nahrávky dabingu. Designeři dotváří herní prostředí, charaktery a uskutečňují se pouze menší změny v GDD. Tým mající na starost úpravy herního enginu, simulaci fyziky či jiných technických nástrojů, může po prvotní iteraci předprodukční fáze, vynechat některé následující iterační fáze. [20] [8]

Testovací fáze se zaměřuje na všechny vývojové oblasti. Testuje se funkčnost funkcí, bezpečnost herních mechanik, vyváženost obtížnosti, umělá inteligence, vykonání skriptů, testují se nahlášené chyby, ale také celkové působení a zábavnost hry. Ať už formou interních playtestů prováděné testery nebo díky zpětné vazbě od hráčů, je možné tyto chyby zdokumentovat a analyzovat. V pozdějších iteracích produkčního cyklu se může testovací fáze rozšiřovat pod pojmenováním Alfa verze. V těchto iteracích je již hra hratelná od začátku

do konce, technologicky hotová a testování se tak zaměřuje spíše na dopilování celkové podoby hry. Výstupem každé testovací fáze jsou Hlášení o chybě, Žádosti o změnu a Rozhodnutí o vývoji, jehož výsledek rozhodne, zda je čas postoupit do fáze Beta nebo opětovné opakování produkčního cyklu.

V pokročilých iteracích Alfa, z nichž pramenící prototypy, již mohou být dostupné zákazníkům (např. pomocí předběžného přístupu), jsou práce na designu hry téměř hotovy. Vývoj přechází do předposlední Beta fáze GDLC. Ve fázi Beta je již hra funkcionálně i assetově kompletní. Při vstupu do Beta fáze by již měly být veškeré práce hotovy a jediné zaměření by se mělo týkat optimalizací, testováním a opravováním bugů. I tyto bugy by však již měly být spíše jen nedodělky a již by se neměly objevovat chyby kritické, které by zabránily vydání hry. Testování probíhá podobně jako v testovací fázi, avšak kromě interního testovacího týmu a hráčů, se k testování využívají i externí společnosti, které mohou testovat kompatibilitu různých SW a HW konfigurací.

Fáze Beta se vyznačuje také tím, že vrcholí marketingové kampaně a práce PR oddělení. Výstupem Beta fáze je Hlášení o chybách a uživatelská zpětná vazba získaná po čas pořádání Open Beta či předběžných přístupů. V některých volnějších metodách řízení vývoje je možné, je-li to nezbytné, se z fáze Beta vrátit zpět do produkčního cyklu. V opačném případě často nastává v posledních týdnech před vydáním tzv. crunchování. To je období, kdy je na vývojáře vyvinut nepřiměřený tlak pro to, aby se stihlo udělat vše potřebné před naplánovaným datem vydání. Vývojáři jsou nuceni brát si neplacené přesčasy a pracovat na maximální vytížení. Pokud se stihne vše potřebné a není nutné odložit datum vydání, tak se v případě vývoje hry na konzole odesílá prototyp k finálnímu testování a schválení výrobci konzolí.

Poslední **fáze vydání (dokončovací fáze)** začíná uzavřením vývoje a následným vytvořením verze „gold master“. Tato finální verze je následně distribuována vydavatelem. Od odeslání vydavateli, až po datum vydání se dále pracuje na opravách pro tzv. „day one patch“. Tento patch je vydán v den vydání hry a má za cíl odstranit poslední velké bugy či nedodělky. Mimo jiné se v této fázi také dodělává projektová dokumentace, provádí se meetingy pro sdílení zkušeností a závěrečné zhodnocení vývoje a plánuje se další práce pro údržbu či budoucí rozšíření hry. Fáze údržby se pak skládá z testování, analýzy a opravy bugů, a také z možného vývoje rozšíření (content updaty, DLC, datadisky) či z plánování práce na nové hře, čímž započne nový vývojový cyklus. [8] [21] [22]

4.2 Problémy a chyby nastalé během vývoje a vydání hry

Kromě již zmíněného crunchování, které má na vývojáře v některých společnostech opravdu špatné dopady, se vývojáři her potýkají s dalšími problémy. Zatímco crunchování je problém, o kterém se ví již dlouhou dobu a v dnešní době se jej společnosti snaží co nejvíce minimalizovat, tak se v poslední době začaly objevovat nové problémy související s posunem některých společenských norem. Tak jako v jiných odvětvích zábavního průmyslu, postupně vychází najevo různá obvinění týkající se nerovných příležitostí, nepřiměřeného ohodnocení, nepřátelského pracovního prostředí či dokonce různého obtěžování zaměstnanců. V poslední době největší kauza zahrnující v podstatě vše zmíněné, se týká společnosti Activision Blizzard. V Severní Americe se tak již objevují snahy o založení odborů podporující videoherní vývojáře.

Problém týkající se zaměstnanců, a ve velké míře vyskytující se v České republice, je jejich nedostatek. Dle průzkumu pořádaného Asociací českých herních vývojářů (GDACZ) má více než polovina českých společností zabývající se vývojem videoher nedostatek vývojářů a neustále otevřenou alespoň jednu pracovní pozici. Zejména umělecké profese, jako animátoři či grafici, jsou velmi žádaní. A jelikož je nedostatek absolventů z českých škol a také nedostatek učitelů a škol se zaměřením na vývoj her, tak to některá česká herní studia řeší zaměstnáváním cizinců. Tím však vznikají další problémy, které se GDACZ pokouší řešit. Jedná se zejména o složitou administrativu, přílišnou byrokracii, nekompetitivní platové ohodnocení, zkrátka nedostatečnou a neefektivní státní podpora českého videoherního průmyslu. [23]

Dalším problémem, jenž se obzvláště u videoher projevuje mnohem více než u ostatního SW, je pirátství. Pirátství, tedy v kontextu videoher neoprávněné nelegální nabytí či sdílení nějaké hry, může zapříčinit vývojářskému studiu velké (či dokonce fatální) finanční ztráty. Pirátství se týká téměř výhradně platformy PC a částečně také mobilních platforem. Pro zajištění ochrany hry před internetovými piráty využívají vývojáři různé formy protipirátských ochran či techniky DRM. Tyto techniky zajišťují protipirátskou ochranu pomocí přídavného SW, ověřitelného SW klíče či jiné SW metody. Většinou tyto DRM ochrany zajišťují platformy vydavatelů (herní obchody) respektive herní spouštěče tzv. „launchery“.

Existuje i diametrálně odlišný přístup k pirátství, a to tzv. DRM free vydavatelé, jenž nabízejí přístup k hrám bez jakékoliv protipirátské ochrany. A to z důvodu početné kritiky ze strany poctivých zákazníků, jenž kvůli různým DRM ochranám musí např. mít stálé připojení

k internetu i u her pro jednoho hráče. Některé DRM ochrany také požadují velké množství systémových oprávnění (čímž vyvstává riziko zneužití soukromých dat uživatelů) a také požadují velké množství HW prostředků (zejména procesorový čas), čímž celý systém zpomalují. Není tedy neobvyklé, že při použití „cracků“ (nebo jiných forem prolomení DRM ochran) je výkon hry lepší, než při spuštění hry oficiální (legální) cestou s využitím DRM.

Obdobná negativa jako u DRM ochran mohou vyvstat i při použití tzv. „anticheatů“. Anticheat je externí SW určený především pro hry více hráčů, který odhaluje (a případně i trestá) hráče, jenž nějakým způsobem podvádí. Jedná se o hráče využívající „cheaty“ nebo „hacky“, které jim dávají herní výhodu nad ostatními hráči, čímž ničí celkový požitek ze hry. Tomuto zneužití herních systémů lze předejít průběžnou opravou nezabezpečených míst v kódu hry, popřípadě nasazením správců serveru. Přístup k řešení problematiky DRM ochran či anticheatů si každé studio musí rozhodnout samo, dle podstaty vyvíjeného herního projektu a jeho cílové skupiny. [24]

Velmi časté problémy při vývoji pramení z nezvládání řízení vývoje, špatné komunikace napříč týmem, špatného zázemí, nízké motivace týmu, nezkušenosti členů týmu, přehlcení bugy, špatné externí spolupráce a přehnaných ambicí. Různých problémů, jenž mohou během vývoje hry nastat, a které musí vývojáři řešit je zkrátka mnoho. Za zmínku stojí také řešení lokalizace, kde je nutné zvážit jak časovou, tak finanční návratnost. Dále je nutné stanovit cílovou skupinu zákazníků (kvůli věkovému omezení her tzv. ratingu) a cílený trh, jemuž se celý design hry musí přizpůsobit. Při vývoji mohou nastat také právní či finanční problémy, např. problémy s šedým trhem s klíči (obcházení oficiálních distributorů a prodávání nelegálně získaných herních klíčů), porušení licencí, různé žaloby apod. [25] [8]

Nejrozšířenější chybou, která má však pro finanční úspěch projektu často fatální následky, je podcenění marketingu. Primárním účelem marketingu je přesvědčit co nejvíce potencionálních zákazníků k pořízení vyvíjené hry, což není v dnešním saturovaném videoherním trhu vůbec jednoduché. Marketing musí být nejaktivnější především ke konci vývoje, ale je nutné s ním počítat po celou jeho dobu (např. při plánování rozpočtu, při cílení na specifický trh je nutná změna designu apod.). Důležitou součástí je i určení správné cenové strategie (jednorázové platby, předplatné, doplňkový obsah, free-to-play, mikrotransakce, reklamy, crowdfunding), nacenění a slev. Avšak hlavní odpovědností členů marketingového oddělení je inzerce a PR.

Inzerce může být provedena formou audiovizuální reklamy v TV, v časopisech, na fyzických místech, na internetu, respektive sociálních sítích (s využitím tvůrců obsahu – influencerů) či formou demoverze (předběžného přístupu). Což znamená omezenou část hry (popřípadě kompletní starší díl v herní sérii) uvolněnou zdarma (případně se slevou). PR neboli česky vztahy s veřejností, mají za cíl co nejlépe prezentovat jak právě vyvíjenou hru, tak samotné vývojářské studio nebo vydavatelství. PR má na starost rozšiřování povědomí, budování komunity a budování značky. Kromě již zmíněných forem inzerce k tomu využívá tisková prohlášení, rozhovory, ukázky, recenze, ocenění, sponzorování, prezentace a veřejné akce.

Nejsilnější formou herního marketingu je marketing šířený „slovem“. Proto je zákaznická podpora, naslouchání zpětné vazbě, podpora módů, nepraktikování kontroverzních praktik (např. P2W, loot-boxy, NFT) a celkové budování komunity (např. roadmapy, komunitní akce, merchandise, sběratelské edice) to nejdůležitější. Navíc tuto formu marketingu si mohou finančně dovolit i vydavatelsky nezávislí vývojáři. Ostatní formy jsou poměrně finančně i časově náročné, a proto jsou většinou zajištěny vydavateli, marketingovými či PR agenturami. S vydavatelem tedy odpadnou problémy se zajištěním marketingu a potřebných financí na vývoj, avšak na oplátku si vydavatelé vezmou část výdělku a právo ovlivňovat vývoj. Je tedy důležité zvolit si nejlépe vyhovující směr a s případným vydavatelem (distributorem) udržovat dobré vztahy, jinak mohou nastat další problémy. [11] [6] [9]

4.3 Nástroje, testovací postupy a technologie

Kromě herních enginů (engine Unity bude popsán v následující kapitole) je při herním vývoji zapotřebí dalších podpůrných SW nástrojů. Podobně jako u metodik, je i u nástrojů důležité, aby bylo jejich použití pro vývoj přínosné a nebylo na obtíž. Větší vývojářská studia tak často investují nemalé zdroje pro vývoj vlastních, na míru vytvořených, nástrojů. Studia, která takovéto zdroje nemají, využívají nástroje třetích stran. Důležité jsou zejména nástroje pro firemní komunikaci a plánování. Do této kategorie se řadí libovolný emailový klient, Skype, Microsoft Teams, Yammer, Google Meet, Zoom, Slack, popřípadě balíky nástrojů jako Microsoft 365 nebo Google Workspace. Dále také specializovaný plánovací SW jako např. Trello, Jira, Redmine, Backlog, Basecamp.

Designery nejčastěji používaný SW bude pravděpodobně nějaký textový procesor, Microsoft Word a Excel, poté CeltX, Final Draft, Quest, popřípadě UML modelovací nástroje. Pro prototypování je možno využít Stencil, GDevelop či GameSalad. Výtvarné oddělení používá pro tvorbu assetů Blender, Daz3D, Krita, či jiný SW od firem jako Adobe, Autodesk, Pixologic. Pro tvorbu pixel artu jsou vhodné programy Aseprite, PyxelEdit a Piskel.

Za zmínku ještě stojí Houdini, což je SW pro tvorbu animací, SW pro tvorbu vegetace SpeedTree a rozsáhlá knihovna vysoce kvalitních vizuálních assetů Quixel. Oddělení zabývající se audiem využívá SW Audacity, FL Studio, Pro Tools či jiné SW DAW řešení. Programátoři pracují s odpovídajícím IDE (např. Visual Studio, IDE od společnosti JetBrains), dále mohou využít různé externí kompilátory jako např. Incredibuild, a poté samozřejmě verzovací nástroje jako např. Git, Mercurial či Helix Core. [9] [26]

Pro testování a optimalizaci je nutné použití nějakého debuggeru či monitorovacího nástroje, které jsou většinou již součástí IDE. Mezi velmi dobré doplňkové monitorovací nástroje patří např. GameBench. Jestliže je zvolena forma testování Continuous Integration, pak je možné využit online nástroj CircleCI. Jako u jiného SW i při vývoji her je možné využití unit testů, testování funkcionalit, integračního testování, systémového testování a u konzolových her i akceptačního testování. Kromě těchto „klasických" testovacích technik se ještě provádí testování kompatibility s vybranou škálou HW, lokalizační testy, které hledají chyby v překladu hry, soak testy (detekování úniku paměti) a v případě her pro více hráčů také zátěžové testování systému. Možná nejpřínosnější ze všech je otevřené Beta testování. Jedná se v podstatě o uživatelské akceptační testování, díky kterému se odhalí široká škála chyb a také celková zábavnost hry. [5]

S testováním souvisí také odpovídající HW platforma. V případě vývoje pro konzole jsou k testování využity speciálně upravené verze daných konzolí (jenž nejsou veřejně dostupné) označované jako „DevKit". Pro vývoj na ostatních platformách je naopak důležité použití co nejvíce různorodého HW a SW. Jak při testování, tak při vývoji jsou však upřednostněny spíše aktuální platformy a technologie. Mezi takovéto technologie patří dnes již téměř standartní (avšak stále poměrně finančně náročný) systém motion capture, což je technologie pro digitalizaci pohybu. Podobná technologie je 3D skenování, díky čemuž je možné digitalizovat nějaký reálný předmět do formy virtuálního objektu a popřípadě získat i jeho fotorealistický vizuál (díky použití velmi citlivých skenerů).

Různé technologie je možné zakomponovat i přímo do hry a její hratelnosti, a tím vytvořit originální herní zážitek. Kromě VR a AR technologií, jež vytvořily v podstatě nový žánr her, je možné přizpůsobit ovládání či hratelnost nějakému hernímu příslušenství. U přenosných zařízení je možné využití různých pohybových či geolokačních senzorů, mikrofonu, fotoaparátu a samozřejmě dotykové obrazovky. Z hlediska SW lze použít např. procedurální generování obsahu, adaptivní hudbu, behaviorální stromy, cloud computing, distribuované výpočty (např. SpatialOS), nové simulační a grafické technologie atd. [8]

5 HERNÍ ENGINE UNITY

Veškeré informace nacházející se v této a také následující kapitole pocházejí z vlastní tvorby, nebo z vlastní rešerše oficiálních zdrojů Unity (zejména Unity dokumentace), pokud nebude uvedeno jinak. [3] [7] [27]

5.1 Obecně

Unity je multiplatformní herní engine, jenž je od roku 2005 vyvíjen společností Unity Technologies[2]. Není nijak žánrově omezen a slouží tak k tvorbě jakéhokoliv 2D či 3D projektu. Aktuálně se řadí mezi nejrozšířenější a nejpoužívanější herní enginy, s velkou oblibou zejména u indie vývojářů. Mimo jiné může za tento fakt také dlouhodobá licenční politika Unity, nabízející za určitých podmínek licenci zdarma. Aktuálně Unity nabízí tři neomezené licence pro týmy odstupňované dle nabídky podpory a roční ceny na zaměstnance od Plus (400 $), Pro (1800 $) až po Enterprise (2400 $), která je určena pro plnou podporu vývoje ve velkých studiích. Individuální licence jsou pak nabízeny zdarma ve variantě pro studenty a osobní užití s podmínkou, že roční výdělky či financování projektu nepřekročí 100 000 $.

Herní engine Unity aktuálně podporuje vývoj pro více než 25 rozličných platforem. Jsou to jak PC platformy, tak mobilní platformy, webové platformy, platformy herních konzolí a platformy pro VR a AR. Zhruba polovina všech her na mobilní platformy a asi 60 % všech VR a AR produktů byly vyvinuty za pomocí Unity. Unity aktuálně nenabízí vlastní IDE pro editaci podporovaných skriptů v jazyce C# (Unity API s frameworkem Mono), avšak spolupracuje v rámci integrace s IDE Microsoft Visual Studio[3]. Samotný engine je napsán v jazyce C++. Unity podporuje i nejrozšířenější API jako jsou Microsoft DirectX[4] (Windows a Xbox), OpenGL[5] (Windows, Linux a částečně macOs), OpenGL ES (Android a iOS) a OpenGL nástupce Vulkan[6] (všechny již zmíněné platformy, ale také např. Nintendo Switch).

Unity v dnešní době nabízí také velkou nabídku různých pokročilých nástrojů a technologií. Jako příklad lze uvést nedávnou prezentaci technického dema Enemies, při které Unity Technologies předvedla nové dechberoucí zpracování lidské herní postavy včetně fotorealistického zpracování očí, vlasů, pleti a dalších prvků, a to vše rendrováno (vykresleno)

[2] Unity Technologies (https://unity.com/)
[3] Microsoft Visual Studio (https://visualstudio.microsoft.com/)
[4] DirectX (https://docs.microsoft.com/cs-cz/windows/win32/directx)
[5] OpenGL (https://www.opengl.org/)
[6] Vulkan (https://www.vulkan.org/)

v reálném čase v rozlišení 4K. Veškeré nástroje použité při tvorbě tohoto dema budou v rámci balíčku Digital Human 2.0 vydány ve verzi Unity 2022. Kromě dnes již relativně zaběhlých technologií Nvidia RTX a Nvidia DLSS, Unity dále nabízí nové zpracování HDR obrazu, Screen Space Global Illumination (globální osvětlení) a Ambient Occlusion (stínění okolím).

Kromě licence zdarma spočívá přívětivost Unity pro nezkušené vývojáře zejména v rozsáhlé, ale přesto přehledné oficiální dokumentaci Unity editoru a API, která obsahuje příklady použití a vysvětlující obrázky. Dále Unity zdarma nabízí oficiální vzdělávací kurzy, ukázky, video návody a částečně i uživatelskou podporu. Další částí je již nyní velmi početná komunita herních vývojářů v Unity, která dále prohlubuje možnosti podpory řešení vzniklých problémů a také poskytuje velké množství různých návodů. Poslední částí je rozsáhlý Unity Asset Store[7]. Tento internetový obchod nabízí prodej a nákup desetitisíců uživatelských assetů, kde zhruba sedm tisíc z nich je k dispozici zcela zdarma.

Unity je vcelku strohý, co se týče nabídky předpřipravených herních objektů, z důvodu zachování co největšího rozsahu použití. I tehdy může přijít Asset Store vhod pro doplnění některých těchto prvků (charakter, player controller, typy kamer apod.). Unity se využívá také pro tvorbu filmů, vizualizací (automobilní, lékařské, architektonické) a simulací (např. nedávno odhalená a nižšími zaměstnanci Unity Technologies kritizovaná podpora pro vojenské simulátory). V herním enginu Unity pracují velké a úspěšné společnosti, které v něm daly za vznik neméně úspěšným hrám. Patří mezi ně např. Subnautica, Escape from Tarkov, Rust, Cities Skylines, Hearthstone, RimWorld, Pokémon Go či české hry Ylands a Beat Saber.

5.2 Unity editor

V Unity editoru probíhá největší tvůrčí část samotného vývoje. Praktické ovládání editoru bude popsáno v následující podkapitole 5.3. Nejdůležitějším prvkem editoru je náhled scény. Scéna je virtuální herní plocha, na které se hra odehrává (často označovaná jako herní svět, level či mapa). Její styl je závislý na žánru hry, formě zpracování obrazových dimenzí, pohledu kamery (first-person, izometrický apod.) a stylu její implementace. Specifickým stylem implementace je procedurální generování mapy nebo jejich částí, kdy je mapa generována algoritmicky, a nikoliv ručně pomocí okna scény vývojářem. Pro lepší představu reálné podoby scény ve hře se lze v editoru přepnout do pohledu „hra“ viděného z perspektivy hlavní herní kamery.

[7] Unity Asset Store (https://assetstore.unity.com/)

Ke spuštění simulace reálného běhu hry pak slouží funkce play. Výsledná hra tak běží přímo v editoru Unity, a to bez nutnosti manuálního buildování. Tuto simulaci běhu hry lze také pozastavit či pokračovat v ní tzv. po krocích, což umožňuje přesné hledání chyb. Ve všech těchto režimech je také možné upravovat scénu dle libosti, kdy se však veškeré provedené změny po ukončení funkce play vrátí zpět do původního stavu, což je výhodné při různém experimentování. Zobrazení tohoto náhledu je možné upravit pomocí rozbalovací nabídky a také je možné do něj přidat některé obrazové pomocníky (Gizmos) a statistiky.

Ve scéně se nachází herní objekty. Ty je možné roztřídit na několik druhů jako jsou 2D objekty (geometrické útvary, křivky, tilemapy atd.), 3D objekty (geometrické útvary, terén, strom atd.), objekty efektů (částicové systémy, trail objekty atd.), objekty nasvícení (různé druhy světel, reflection probes), audio a video objekty (zdroje zvuku, videopřehrávač), prvky GUI (text, tlačítko, scrollbar atd.) a další speciální typy objektů (prázdný objekt, kamera, prefaby). Pomocí prefabů je možné předpřipravení nějakého jiného objektu (či množiny objektů) i mimo aktivní herní scénu a následně jej vkládat do scén jakožto odkaz na tento prefab. To velice usnadňuje práci se skupinami objektů stejného typu s pouze drobnými odlišnostmi. Při změně nějaké společné vlastnosti ji poté stačí změnit v prefabu a automaticky se změní i ve všech odkazovaných objektech umístěných ve scénách.

Proměnné vlastnosti herních objektů, jenž jsou uskupovány do větších celků nazývaných jako komponenty, jsou pro určení chování těchto objektů klíčové. Každý herní objekt obsahuje základní komponentu Transform, pro určení vlastností polohy objektu ve scéně. Unity dále nabízí přes 100 předpřipravených komponent. Mezi ně patří např. různé Renderery (vlastnosti vykreslení objektu), komponenty Rigidbody (vlastnosti pro simulaci fyziky), Collidery (vlastnosti určování kolizí) a mnoho dalších. Tyto komponenty je možné libovolně přiřazovat jakémukoliv objektu. Pokud žádný předpřipravený komponent nevyhovuje požadovaným vlastnostem, pak je možné tyto vlastnosti objektu doplnit přiřazením libovolného C# skriptu do komponenty Script.

Speciální Unity assety jsou (fyzikální) materiály a shadery. Ty dokáží určovat vlastnosti a vzhled povrchu nějakého objektu, definují např. jak odráží světlo, jak reaguje na tření při kolizi, tvrdost objektu apod. Další speciální assety jsou Unity animace vytvořené pomocí nástroje Animation a spravované nástrojem Animator. Mezi další nástroje Unity patří Audio mixer (pro správu zvuků), Timeline (pro tvorbu cutscene), Sprite editor, Tile pallete (správa tile palet), správa navigace AI, nástroje pro nasvícení, profiler (pro sledování výkonu),

různé debuggery, správce služeb, správce balíčků (a vlastněných assetů v Asset Store) a nástroj Colaborate. To je systém pro týmové sdílení a synchronizaci Unity projektů v cloudu.

Kromě již zmíněných nových grafických technologií disponuje Unity dalšími běžnějšími post-processingovými efekty. Patří mezi ně Anti-aliasing, Bloom, Chromatic Aberration, Grain, Lens Distortion, Color grading, Motion Blur, mlha, hloubka ostrosti, mixování barev a mnoho dalších. Unity disponuje nastavením grafických úrovní kvality (včetně LOD systému), které je však poněkud ukryto společně s dalšími užitečnými nastaveními v nastavení projektu.

5.3 Praktický vývoj v Unity

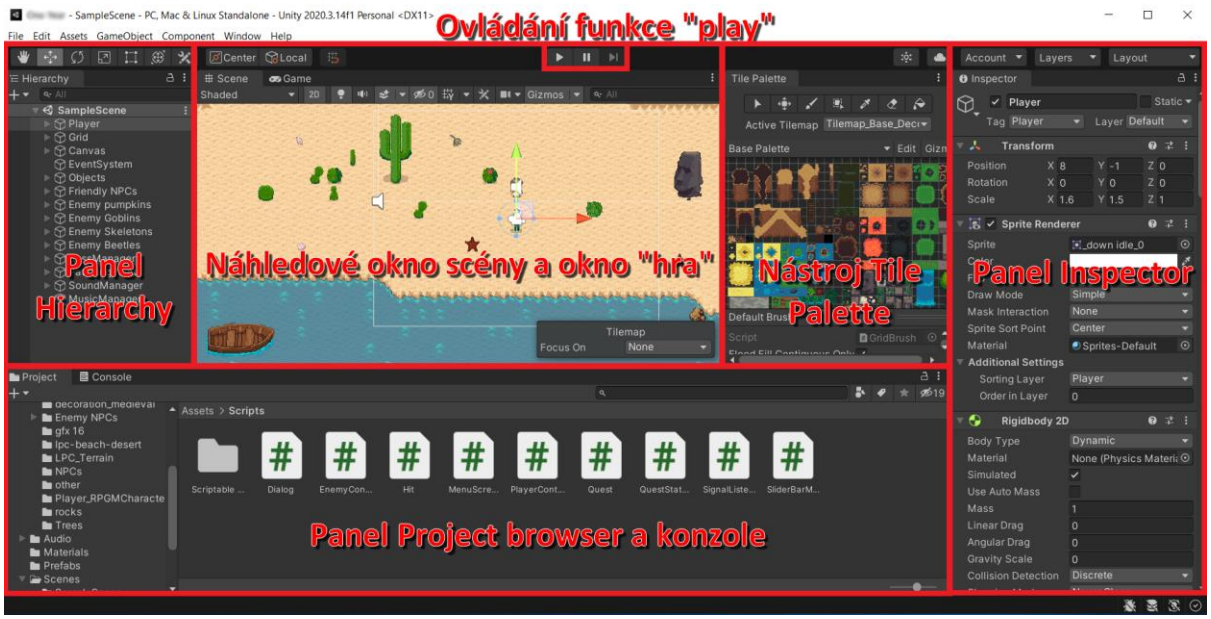
Herní engine Unity byl pro praktický vývoj ukázkové hry zvolen hned z několika důvodů. Zřejmě ten nejdůležitější je podpora licence pro použití zdarma s dostatečnými nástroji (pro vyvíjený projekt) a obří kolekcí různých assetů použitelných často taktéž zdarma. Ty je navíc možné získat přímo z Unity Asset Store. Dalším důvodem je také využití programovacího jazyka C# pro tvorbu skriptů a jednoduchého zpracování ovládání UI podobného SW Autodesk Maya[8], které byly již dříve použity autorem práce v rámci studia předmětu Skriptování v 3D grafických softwarech.

Základní stavební kameny Unity, tedy jednoduchost použití, navržení pro žánrově univerzální vývoj a široká podpora platforem, jsou pro vývoj tohoto typu projektu také výhodné vlastnosti. Editor unity je v aktuální LTS verzi po stránce optimalizace velmi dobře zpracovaný, úsporný na využití HW prostředků a proces kompilace je rychlý a jednoduchý. Zřejmě největší výhodou oproti ostatním herním enginům je však obrovská uživatelská základna a obecná podpora při vývoji, takže jej zvládne i začínající vývojář.

Pro zahájení vývoje ukázkové hry v Unity bylo nejprve zapotřebí vytvoření Unity účtu. Ten mimo jiné také spravuje aktivované licence pro použiti Unity a také zajišťuje správu knihovny assetů získaných z Unity Asset Store. Po zvolení licence pro použiti Unity zdarma a vložení některých potenciálně užitečných assetů, bylo provedeno stažení instalačního balíčku Unity Hubu. Po jednoduché instalaci Unity Hubu byla vybrána a nainstalována nejnovější LTS verze Unity 2020.3.14f1.

[8] Autodesk Maya (https://www.autodesk.cz/products/maya/)

Nebyly nainstalovány žádné přídavné moduly, jelikož jediný pro vývoj skutečně potřebný modul, resp. IDE Microsoft Visual Studio Community 2019, byl autorem práce nainstalován již dříve. To však způsobilo problém s rozpoznáním tohoto IDE ze strany Unity. Řešením bylo manuální znovu vygenerování projektových souborů a přiřazení IDE v nastavení projektu Unity. Samotný vývoj poté započal vytvořením nového projektu prostřednictvím předpřipravené projektové šablony pro tvorbu projektu ve 2D. Poté již mohlo započít hlubší seznámení se s uživatelským prostředím Unity editoru.



Obrázek 4 - Rozložení uživatelského rozhraní Unity editoru

Hlavní okno editoru (viz. obr. 4) je rozděleno na menší nástrojové panely. Samozřejmostí je možnost úpravy rozložení těchto panelů a také vytvoření uživatelských schémat pro uložení stavu rozložení a navrácení do výchozího stavu. V hlavním okně editoru se v dolní části nachází panel pro práci se soubory hry nazývaný Project browser a panel s konzolí. Nad těmito panely vlevo se nachází panel znázorňující hierarchii objektů obsažených v právě aktivní scéně. Uprostřed hlavního okna editoru se nachází náhledové okno, jenž slouží pro vizuální znázornění, úpravu a navigaci skrze aktivní scénu a herní pohled. Navigace skrze aktivní scénu a úprava jejich objektů je zde provedena intuitivním ovládáním obecně známým z populárních grafických programů. V horní liště editoru se nacházejí tlačítka pro přepínání módu úprav a další pomocné nástroje a nastavení (včetně menu lišty editoru). Všechny panely podporují systém drag&drop pro snadné ovládání. V pravé části hlavního okna editoru se nachází panel Inspektor, který detailně zobrazuje vlastnosti označených herních objektů a souborů s možností úprav jejich vlastností.

6 PRAKTICKÝ VÝVOJ KONKRÉTNÍ HRY

Ještě před začátkem vývoje je důležité stanovení omezení dostupných zdrojů a platforem k realizaci projektu. Dostupné zdroje pro vývoj herního projektu v rámci této práce byly stanoveny následovně. Lidské zdroje byly z podstaty práce omezeny na pouze jednu osobu, a to autora práce. Využití finančních zdrojů bylo omezeno s cílem co nejvíce minimalizovat potřebné náklady projektu a s vědomím přímé neúčasti investorů či vydavatelů v projektu. Taktéž časová složka dostupných zdrojů byla tvrdě omezena datem odevzdání této práce. Mezi dostupné platformy pak patří PC (s OS Windows 10) a mobilní zařízení (s OS Android 12). Rozsah herního projektu vyvíjeného v rámci této práce byl poté stanoven s ohledem k těmto omezeným zdrojům a s vývojovou platformou PC (s OS Windows).

6.1 Zahajovací fáze

Na samotném začátku vývoje je nutno zpracovat nápady a myšlenky do ucelenější formy, zohlednit omezené zdroje, vytyčit mantinely a položit tak základy, na kterých bude vyvíjená hra stavět. K tomu jako inspirace posloužily staré díly herní série The Legend of Zelda[9], a také především nové hry stylu „top-down“. Konkrétně to jsou hry: Stardow Valley[10] a Graveyard keeper[11]. Obě hry jsou prodejně velmi úspěšné, pohybující se v řádech miliónů prodaných kusů. Stardow Valley byl původně projekt jednoho vývojáře. Zpracování obou her je přiměřeně komplexní, aby byl jejich vývoj za omezených zdrojů uskutečnitelný a zároveň, aby na hráče nepůsobilo stroze (lacině). Zvládnutí této správné míry komplexnosti je při návrhu hry velmi důležité. Použití úspěšných her jako inspirací je pro vývoj velmi výhodné, jelikož se na jejich základě dají odvodit přibližné nároky a výsledky vyvíjeného projektu. Tedy např. zda byl zvolený žánr funkční, jak hra působila na hráče, jaké postupy a technologie byly při vývoji zvoleny, co by se dalo při vývoji vylepšit atd.

Pro vyvíjenou hru je tak na základě výše uvedeného, zvolena odlehčenější forma herního žánru RPG, tedy hra na hrdiny s jednodušší formou zpracování (někdy označována jako RPG-like). Hra je určena čistě pro jednoho hráče (pouze PvE mód) a neobsahuje jakýkoliv obsah pro hru více hráčů. Hra je vzhledem k omezeným zdrojům obsahově poměrně plytká s možností budoucího rozšíření. Zvolení pixel artové grafické reprezentace primárně pomocí systému top down 2D tilemaps je taktéž založeno na základě určité jednoduchosti zpracování, dostupnosti vybraných assetů a inspirací jinými hrami. V rámci autorovi vlastní invence

[9] The Legend of Zelda (1986) (https://www.zelda.com/)
[10] Stardow Valley (2016) (https://www.stardewvalley.net/)
[11] Graveyard keeper (2018) (https://www.graveyardkeeper.com/)

je pro vyvíjený projekt navrženo poutavé postupování hrou a zasazení herního světa do středověkého království, v rámci lineárně vyprávěného příběhu.

Výše zmíněné nápady a myšlenky jsou základní kameny pro tvorbu koncepčních dokumentů, pomocí nichž je poté vývoj hry řízen. Na základě nich taktéž byla zpracována následující základní myšlenka projektu neboli **High Concept:** „*Po ztroskotání lodi jste vyplaveni na pláž středověkého království. Dokážete tuto bídnou situaci přežít a postavit se opět na vlastní nohy? Plňte zábavné úkoly. Poznejte tento unikátní herní svět a překonejte jeho nástrahy ve 2D pixel art zpracování s RPG prvky.*“ [autor práce]

Jelikož je rozsah hry velmi malý (jak obsahově, tak komplexností herních systémů) a odpovídá tak omezené velikosti dostupných zdrojů, tak byl také odpovídajícím způsobem zkrácen i rozsah Koncept dokumentu, GDD a dalších dokumentů. Zjednodušeně vypracovány a uvedeny jsou tak jen některé neduplicitní významné části těchto dokumentů. Některé (pro tento projekt zbytečné) dokumenty byly z vývojového procesu vypuštěny úplně. Tím bylo zajištěno ušetření významné části dostupného časového zdroje. Shrnutí konkrétních částí zjednodušeného Koncept dokumentu pro vyvíjenou hru vypadá následovně.

Hratelnost a vlastnosti hry:

Základní motivací hráče by měla být zábavná aktivita skládající se z postupu hráče hrou, vylepšováním jeho herní postavy a vyprávěním příběhu. Hráč ovládá svou herní postavu a může interagovat s některými herními objekty a NPC. Hra bude mít klasickou hratelnostní smyčku: výzva, odměna, očekávání. Tedy hráč obdrží nějaký úkol, ten splní a dostane za něj odměnu ve formě bodů zkušeností XP (případně dalších herních prvků), a tím získá možnost vylepšení jeho herní postavy. Tím, jak svou herní postavu vylepšuje, tak má možnost dalšího postupu hrou (větší výzvy) až do doby, dokud se nedostane na konec hry. Dalším možným rozšířením herních systémů mohou být systémy výroby předmětů a správy či stavění budov.

Prostředí a příběh hry:

Herní prostředí (viz. obr. 5) se skládá ze základní herní mapy (s možností rozšíření o další herní mapy) ve stylu středověkého království na pobřeží moře. Celá mapa je ohraničena přírodními překážkami (moře, skály, řeka). Na jižní a východní straně mapy se nachází pobřeží, na západní straně se nachází les následovaný neprostupnými skalami. Na severní straně se nachází řeka. Na severo-východní straně se nachází samotný hrad a podhradí, které se směrem ke středu mapy mění na venkov s obilnými poli. Základní příběhová zápletka

je vcelku jednoduchá. Po již zmíněném ztroskotání na pláži si hráčova postava nic z předchozího života nepamatuje. Lidé v království trpí pod krutovládou zdejšího krále. Nikdo nesmí království opustit. A jedinou možností pro lepší život je vybudování postavení, a poté sesazení krále.

Obrázek 5 - Základní herní mapa

Cílová skupina uživatelů a HW platformy:

Z důvodů ušetření dostupných vývojových zdrojů nebyl proveden průzkum trhu či dokonce focus testing. Cílovou skupinu je tak možné pouze odhadnout na základě autorova zaujetí tímto herním žánrem a některými odhady u dvou inspirativních her. Cílová skupina hráčů je složena z rovnoměrného zastoupení mužů a žen ve věku od 9 let, se základními hráčskými schopnostmi, bez specifického kulturního a jazykového omezení. Tento typ her není náročný na ovládání, pochopení a soustředění, proto jej vyhledávají spíše tzv. casual (příležitostní) hráči, či lidé ne tolik kompetitivního charakteru toužící především po odpočinkovém prožitku. Proto byl zvolen relativně vysoký poměr hráček a díky absenci složitých či nevhodných témat, také přístupnost pro děti. Z HW platforem jsou u této cílové skupiny oblíbené PC a mobilní platformy. Pro vývoj byla zvolena platforma PC, s případnou vidinou budoucího portu pro mobilní platformy.

Časový harmonogram:

Vývoj hry v tomto rozsahu by neměl být moc časově náročný. Rozdělení časového zdroje na jednotlivé vývojové fáze bude poměrně nerovnoměrné. Časově nejnáročnější, pohybující se v řádu týdnů, budou fáze Zahájení, Předprodukce a zejména Produkce. Na Testovací, Beta a Dokončovací fáze bude vyhrazeno pouze pár dnů. Což může způsobit pokles kvality hry při vydání, avšak to se dá v případě profitu opravit za pomocí patchů po vydání. Ke hře není v tuto chvíli plánován žádný dodatečný obsah.

Finanční harmonogram:

Finanční harmonogram je z hlediska minimalizace míry výdajů velmi jednoduchý. Všechny použité assety by měly být dostupné k použití zdarma. Hra bude distribuována na bezplatné službě itch.io[12], která také bude zajišťovat „slovní" formu marketingu. Kvůli podstatě projektu nebudou využity žádné jiné inzerční a marketingové služby (demo) či jiné výdaje (mzdy). V této chvíli není předpokládán zisk, hra bude uvolněna zdarma. To se však může změnit s případným dalším vývojem. Poté by byl zisk zajištěn jednorázovou platbou (1 $ až 5 $).

Analýza rizik a konkurence:

Největším rizikem projektu je nedostatečná zkušenost a umělecká zručnost autora, jenž se pravděpodobně projeví při zpracování assetů. Dalším rizikem je nedostatek časového zdroje. Zejména kvůli těmto dvou rizikům hrozí nedostatečně kvalitní zpracování při vydání, a navíc s ohledem k široké nabídce konkurence, může hrozit nedostatečné oslovení zákazníků. V rámci služby itch.io, která nabízí přes půl milionu her zdarma, včetně asi stovek podobně zpracovaných her, je konkurence velká.

6.2 Předprodukční fáze

Ještě, než započne předprodukční fáze, tak je nutné si z dostupných vývojových modelů a metodik vybrat takové, které budou pro řízení vývoje vhodné. Jelikož je vyvíjený projekt pro autora práce novou zkušeností, tak je nutné předpokládat nedostatečné definování požadavků, vznik nepředvídatelných problémů a velmi omezené zdroje. Z těchto a dalších důvodů byl pro vývoj zvolen iterativní, resp. agilní model vývoje. Díky principům agilního vývoje bude řízení projektu snazší a méně chaotičtější. U opravdu malých projektů je důležité se držet těchto základních principů a nic víc není potřeba, avšak u tohoto projektu a projektů většího rozsahu je nutná volba nějaké konkrétní metodiky.

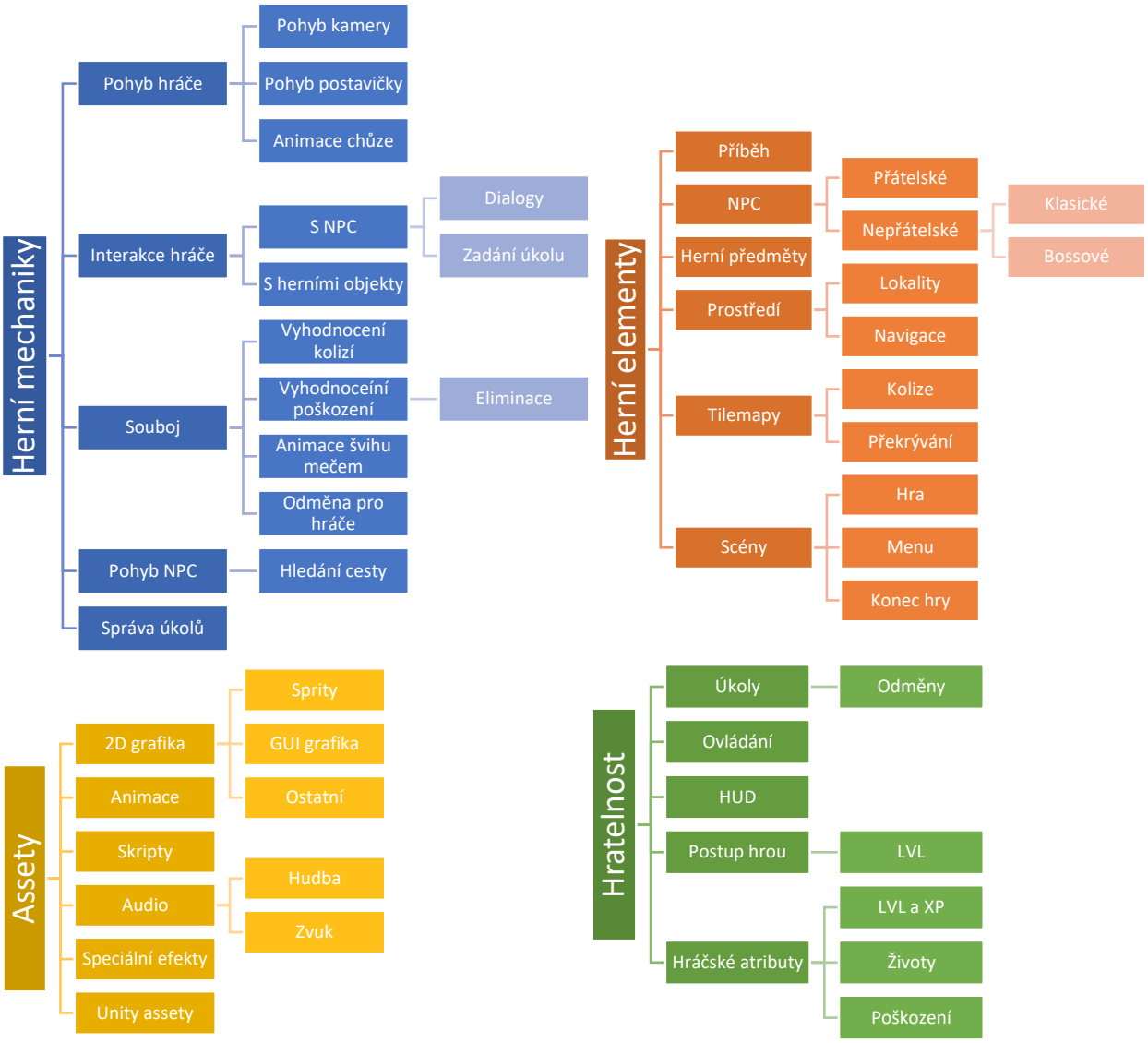
[12] itch.io (https://itch.io/)

Vzhledem k povaze projektu a jednočlenného týmu nebylo možné použít nějakou konkrétní striktní agilní metodiku beze změny, jelikož jsou určeny především pro vývoj ve více lidech. Z toho důvodu byl zvolen liberálnější přístup řízení vývoje, jemuž nejblíže odpovídají upravené praktiky metodik Kanban, potažmo Lean. Toto rozhodnutí je v souladu s průzkumem internetových fór, kde sólo vývojáři doporučovali tento přístup k řízení sólo vývoje a taktéž jej vylepšovali o další vhodné praktiky. Použitá a autorem vhodně upravená metodika Kanban, jenž nebude obsahovat některé nekompatibilní praktiky, a naopak přidá některé vhodné, bude ctít tyto principy:

- vizualizace vývojového procesu (Kanban štítková tabule),
- rozdělení úkolů dle míry detailů (abstrakce) za použití milníků (pro vytyčení iterací),
- omezení rozpracované práce (dokončit úkol před zahájením dalšího),
- minimalizace velikosti pracovních úkolů (a jasné názvy úkolů),
- rychlá reakce na změny (a na případnou zpětnou vazbu),
- zaměření se na celek a jeho výsledné působení a
- nedělat nic navíc, rozhodovat se co nejpozději a zlepšovat se v průběhu (iterace).

Před samotným zahájením designerských prací na GDD se u větších projektů zpracovává PoC, jenž má za cíl odhalit, zda je navržený koncept proveditelný, a hlavně uplatnitelný na trhu. Je v něm nutné prokázání potřeby, což je u vytvářené hry v celku jednoduché, jelikož se výrazně inspiruje na jiných velmi úspěšných projektech. Dalším krokem je brainstorming nad problémy, funkcionalitami a jejich možnými řešeními, což se hodí při následném definování pracovních úkolů. Následuje vyhodnocení zpětné vazby a zvážení technické proveditelnosti, což u vyvíjené hry již bylo dokázáno opět pomocí již zmíněných inspirativních projektů. Vzájemnou funkčnost všech těchto aspektů je poté nutné dokázat vypracováním prototypu (označovaného jako MVP).

Předprodukční fáze se obvykle týká zejména práce na GDD. Tento důležitý dokument stanovuje všechny prvky vyvíjené hry, jenž je nutné vypracovat. Avšak jak již bylo mnohokrát avizováno, forma zpracování jinak velmi důležitých dokumentů je v tomto případě, z důvodu rozsahu projektu a jednočlenného týmu, jiná. Jelikož je GDD určeno primárně pro dodržení konvencí mezi členy týmu, tak jeho vypracování v tomto případě nedává smysl. Avšak pro udržení přehledu o důležitých aspektech vyvíjené hry je nutné tyto aspekty přehledně zaznamenat. Proto byla vytvořena minimalistická forma GDD skládající se z jeho 4 hlavních částí, jenž jsou reprezentovány těmito původními přehledovými grafy.

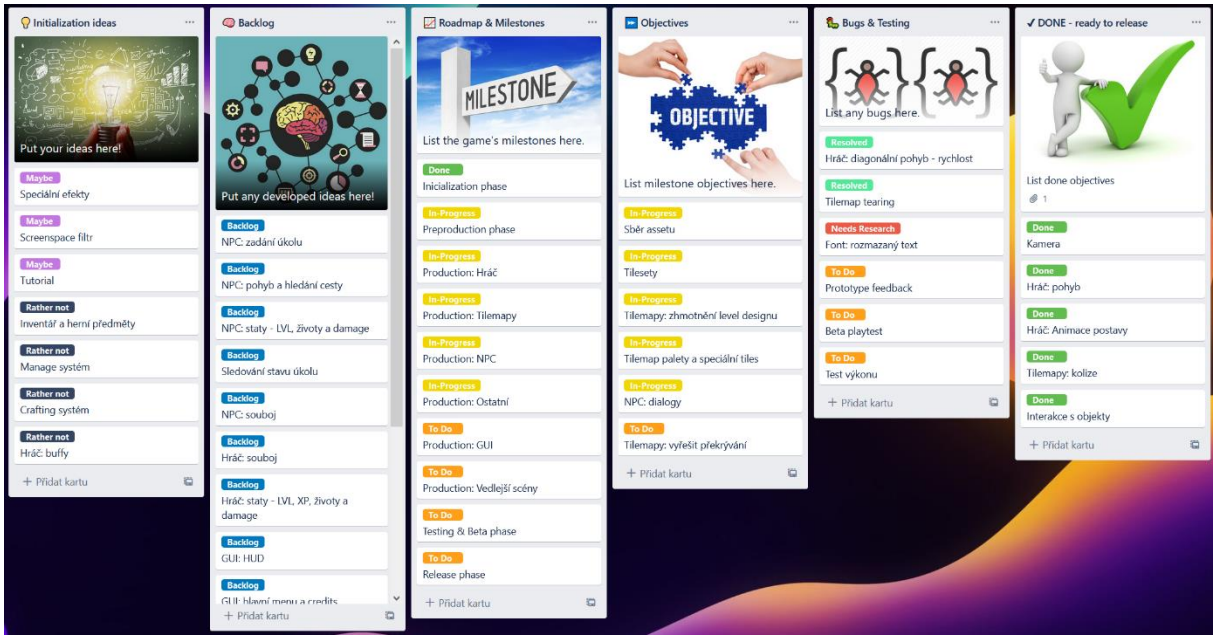


Obrázek 6 - Prvotní verze přehledových grafů čtyř hlavních částí GDD

Na základě těchto grafů byla poté zpracována vizualizace vývojového procesu s metodikou Kanban za použití služby Trello[13]. Tato webová služba nabízí bezplatnou správu projektů, právě díky znázornění ve stylu Kanban, tedy pomocí úkolů a štítků na tabuli. Tabule byla rozdělena na 6 sloupců, jak je vidět na obrázku č. 7. První zleva (Initialization ideas) je určen pro nápady, které vznikly v inicializační fázi, a které se mohou, ale také nemusejí zrealizovat. Druhý sloupec nazvaný Backlog je určen jako odkladní prostor pro veškeré úkoly, jenž bylo nutné zpracovat. Další sloupec Roadmap obsahuje jednotlivé fáze a milníky vývoje. A poté následuje sloupec Objectives znázorňující právě řešené úkoly. Předposledním sloupcem je Bugs & Testing, který znázorňuje případné chyby a testy. A následuje poslední

[13] Trello (https://trello.com)

sloupec (DONE – ready to release) pro hotové úkoly. Každý sloupec obsahuje úkoly, jenž jsou označené barevnými štítky vyjadřující stav daného úkolu. Trello tak slouží jako zjednodušený Project plan.



Obrázek 7 - Trello s vizualizací vývojového procesu

V rámci předprodukční fáze proběhlo také hledání použitelných assetů do vyvíjené hry. Až na pár výjimek, které byly vytvořeny autorem práce, jsou veškeré umělecké assety získány legální cestou z assetových úložišť či obchodů. A to z důvodu vyhodnocení uměleckých dovedností autora práce jako nedostatečných pro tvorbu všech použitých assetů. Velmi rozsáhlým a užitečným webovým úložištěm bylo OpenGameArt.org. Toto webové úložiště obsahuje především 2D i 3D grafiku, ale také fonty, zvuky a hudbu, a to vše pod licencemi nabízející tyto média zdarma. Každý nabízený asset je licencován pod jednou nebo více z těchto licencí: CC-BY-SA 3.0 (4.0), CC-BY 3.0 (4.0), CC0[14], OGA-BY 3.0[15], GPL 3.0 (2.0) a LGPL 3 (2.1)[16]. Každá má trochu jiné podmínky použití, které je nutné dodržet.

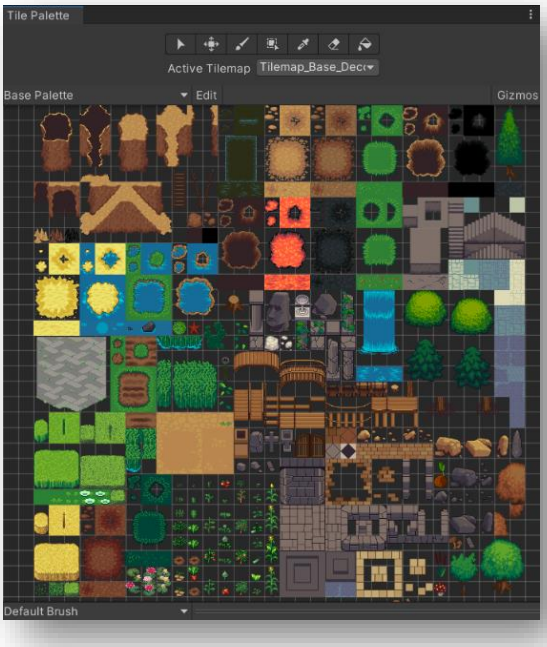
Významným zdrojem hudby a zvuků, které jsou nabízeny pod licencemi Creative Commons je ccMixter. Dalšími zdroji assetů dostupnými zdarma mohou být Itch.io, Kenney.nl, ArtStation, Reddit, Unity Asset Store a mnoho dalších. Bohužel kvalita těchto assetů je hodně kolísavá. Navíc dodržení určité formy stylu, formátu a jednotvárnosti použitých assetů je značně důležité, a z tohoto důvodu bylo hledání použitelných assetů velmi časově náročné.

[14] CC-BY-SA 3.0 (4.0), CC-BY 3.0 (4.0), CC0 (https://creativecommons.org/licenses/)
[15] OGA-BY 3.0 (https://opengameart.org/content/oga-by-30-faq)
[16] GPL 3.0 (2.0) a LGPL 3 (2.1) (http://www.gnu.org/licenses/licenses.html)

Výstupem předprodukční fáze je v pozdějších iteracích koncepční prototyp. Tento prototyp byl vytvořen na základě Koncept dokumentu a snažil se tak dokázat proveditelnost, zábavnost a funkčnost všech stanovených koncepčních náležitostí. Proveditelnost byla dokázána praktickým zpracováním základních herních mechanik. Zábavnost a funkčnost konceptu byla dokázána pomocí testovacího průchodu s dočasným testovacím obsahem (assety, dialogy, úkoly). Prototyp neobsahoval všechny herní mechaniky, neobsahoval některé důležité assety (např. zvuky), neobsahoval zakončení hry a obsahoval pouze jednu hlavní scénu. Důsledkem vypracování tohoto prototypu bylo shledáno navržení konceptu vyvíjené hry jako korektní.

6.3 Produkční fáze

Vývoj samotné hry v herním enginu Unity je hlavní náplní produkční fáze. Po úspěšném stažení a instalaci Unity Hub, LTS verze Unity editoru (2020.3.14.f1) a IDE Microsoft Visual Studio (ve verzi Community 2019) bylo zapotřebí založení nového projektu. Po zvolení verze editoru, názvu projektu a předpřipravené šablony pro 2D projekty byl projekt vytvořen včetně jedné základní scény. Následně byly do struktury projektu naimportovány, nastaveny a rozříznuty (pomocí nástroje Sprite editor) potřebné sprity a tilesety. Tilesety jsou sady spritů primárně vytvořených pro zasazení jednotlivých tilů (dlaždic) do tilemap objektů. Tyto objekty jsou jakési vrstvy podkladu herní plochy umístěné do objektu Grid (mřížky), díky čemuž jsou tilemapy konzistentně rozděleny na jednotlivé tily.

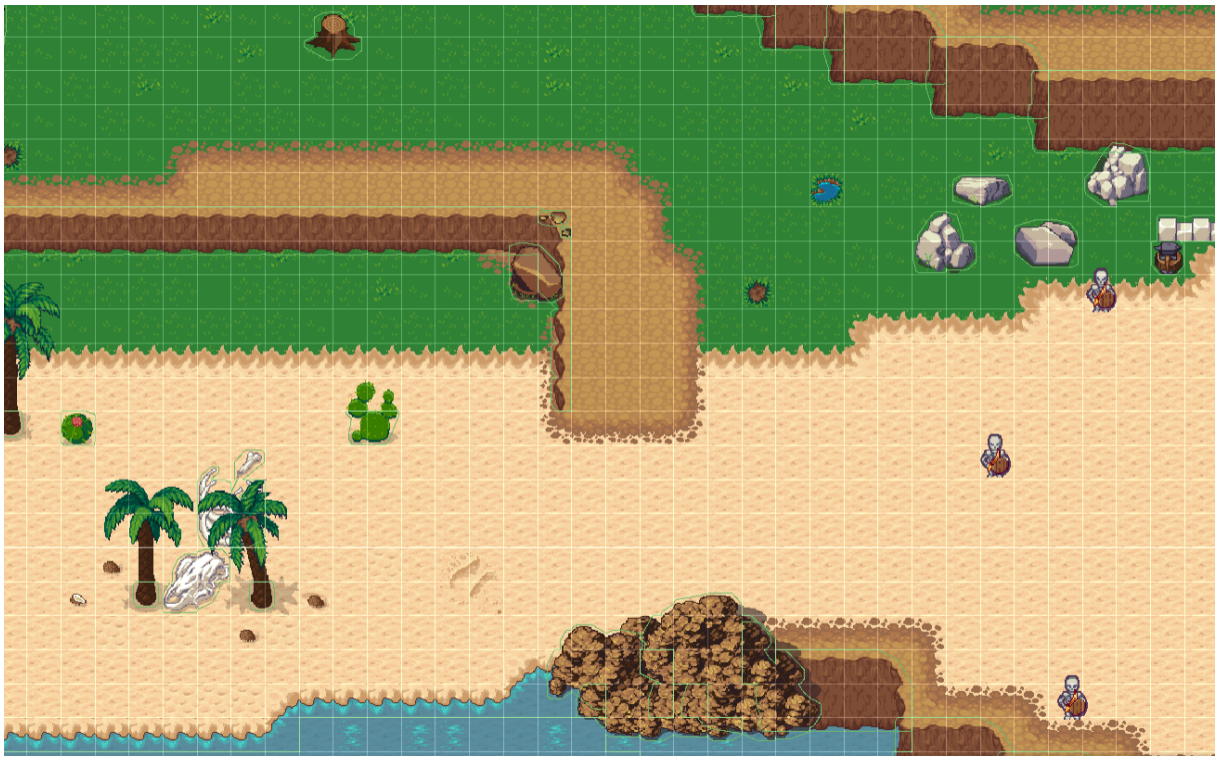


Obrázek 8 - Hlavní podkladový tileset jakožto základní paleta v nástroji Tile Pallete

Dalším krokem bylo vytvoření jednotlivých tilemap objektů pro tvorbu světa. K naplňování těchto tilemap objektů byl použit nástroj Tile Palette. V něm byly za pomocí tilesetů vytvořeny tile palety, čímž byly rozděleny na jednotlivé tily, které se již mohou nanášet na jednotlivé tilemapy. Pro lepší práci s nanášením těchto tilů byly použity speciální rule tily[17], které však v této verzi editoru nebyly normálně dostupné a bylo je nutné do projektu přidat jakožto balíček nástrojů Unity. Pro přidání tohoto balíčku bylo nutné vytvořit tuto cestu v projektovém adresáři: `Packages/com.unity.2d.tilemap.extras` a následně

[17] Rule tily (https://learn.unity.com/tutorial/using-rule-tiles)

do tohoto podadresáře stáhnout a rozbalit zdrojový kód z veřejného GitHub repositáře: `Unity-Technologies/2d-extras` (z branch odpovídající verzi editoru). Po restartování editoru již bylo možné tyto rule tily použít.

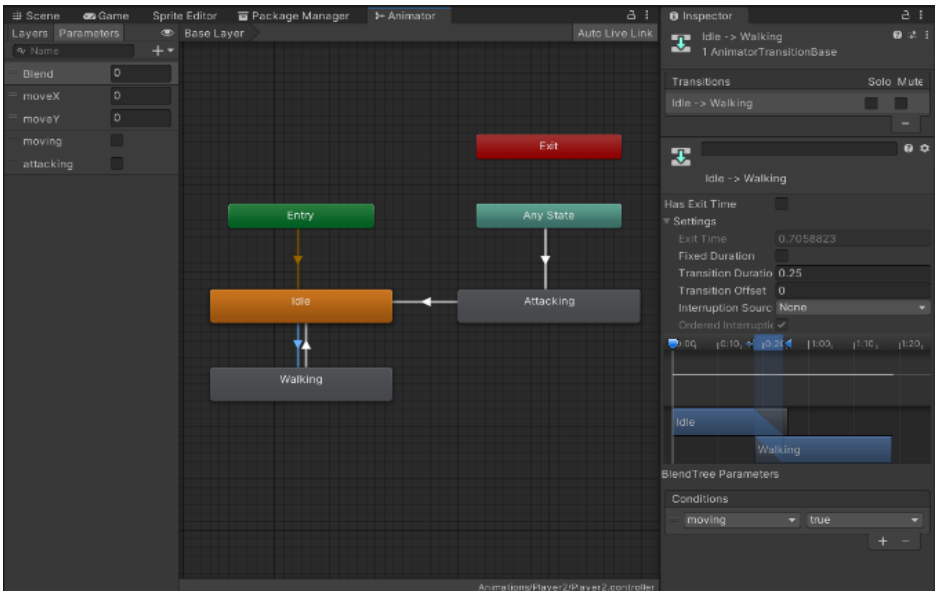


Obrázek 9 - Vrstvení tilemap se zelenými obrysy znázorňující kolizní plochy

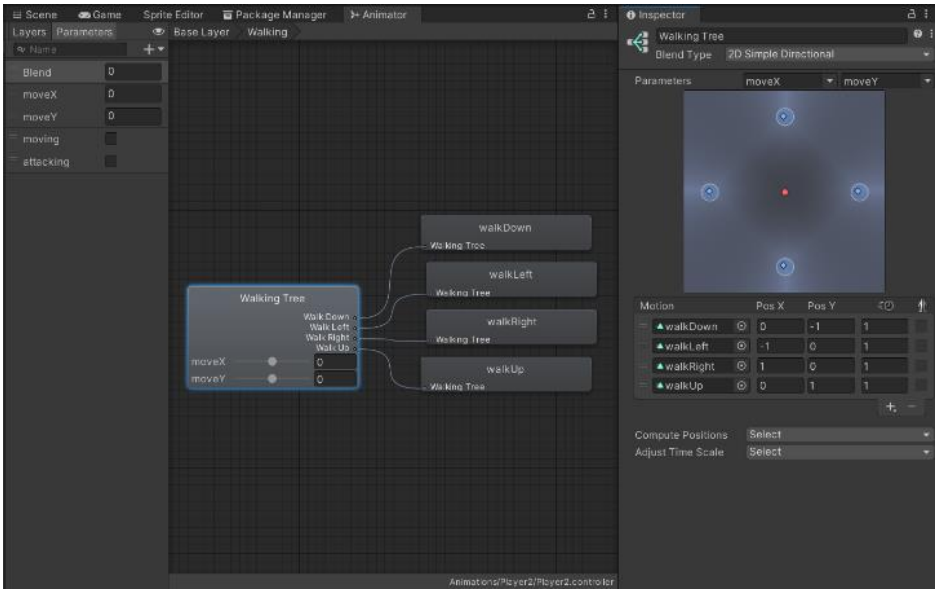
Tilemap bylo vytvořeno více pro dosažení vrstvení (viz. obr. 9), kde je potřeba umístit několik tilů na sebe, například z důvodu plynulého vizuálního přechodu mezi tily, pro podporu kolizních tilů či tilů s průhlednou částí (využívaných pro dekorace). Kromě základní podkladové tilemapy byly tedy vytvořeny další tilemapy pro dekorace a následně i tilemapy s kolizemi a tilemapy pro překrytí hráče. Tohoto překrytí je dosaženo pomocí pořadí řazení v Tilemap Rendereru. Tilemapy pro překrytí hráče, u nichž je hráč vykreslován za nimi, je také nutné nastavit na (nebo nad) řadící vrstvu určenou pro hráče. Tilemapy s kolizemi byly vytvořeny pomocí statického Rigidbody 2D, Tilemap Collider 2D, kterého obsluhoval Composit Collider 2D, díky němuž se drobné kolizní boxy shlukly do větších celků, což výrazně pomohlo s optimalizací výkonu hry.

Ze spritu postavy byl vytvořen základ pro vizuální reprezentaci hráče. Následně pro něj byly přidány a nastaveny komponenty Sprite Renderer, Rigidbody 2D a Box Collider 2D a první skript `PlayerController.cs` zajišťující pohyb hráče včetně správy animací. Ovládání pohybu je nastaveno na výchozí klávesy „`WASD`“ nebo „`šipky`“. Poté byly vytvořeny všechny

animace hráče pomocí nástroje Animation a následně byla objektu Player přiřazena komponenta Animator pro řízení těchto animací. Tu je nutné za pomocí nástroje Animator nastavit. Jeho praktické použití lze vidět na následujících obrázcích. Pro jednotlivé animace byly vytvořeny Blend tree s patřičnými parametry a nastavením jednotlivých animací. Mezi Blend tree byly následně udělány podmíněné přechody bez exit time a fixní délky pro hladké přechody animací. Tyto animační přechody a Blend tree lze vidět na obrázku č. 10, kde jsou také vidět vlastnosti podmíněného animačního přechodu Idle → Walking. Vnitřní struktura těchto Blend tree je řízena parametry naplněnými hodnotami z již zmíněného skriptu a následně jim jsou nastaveny konkrétní animace, jak lze vidět na obrázku č. 11.



Obrázek 10 - Nástroj Animator – animační graf hráče s vlastnostmi označeného přechodu



Obrázek 11 - Nástroj Animator – vnitřní struktura a vlastnosti Walking Blend tree

V další iteraci byla vytvořena mechanika interakce a dialogu s přátelským NPC, z čehož byl posléze vytvořen prefab pro vícenásobné použití. Přátelské NPC se skládá z komponent Sprite renderer, Box Collider 2D pro kolizi s hráčem a druhý o něco větší Box Collider 2D s parametrem trigger pro spuštění událostí a skriptů `Dialog.cs` a `SignalListener.cs`. Tohoto trigger Box Collideru bylo využito ve skriptu `Dialog.cs`, jenž řeší systém dialogů a úkolů. Kromě textových zpráv, úkolu, ID dalšího úkolu a signálů obsahuje tento skript také vstup pro hráče, dialogové okno a textový objekt dialogového systému. Prefab NPC je rodič objektu dialogové okno, který má potomka objekt Canvas. A ten je rodičem objektu Text (TMP).

Objekt dialogové okno slouží pro vymezení velikosti Canvasu, vykreslení pozadí a aktivaci dialogového systému. Canvas je UI objekt sloužící pro vykreslování GUI objektů, a v tomto případě je použit s nastavením pro vykreslení ve World Space (tzn. v souřadnicích scény, nikoliv obrazovky, jak tomu ve výchozím použití bývá). UI objekt Text (TMP) následně zajišťuje cílené vykreslení textových zpráv, jež jsou získány ze vstupu skriptu `Dialog.cs`. Pro aktivaci dialogu musí být hráč uvnitř triger Box Collideru u NPC, které je připraveno vést dialog a následně zmáčknout klávesu „`E`“, čímž dojde buď k aktivaci dialogu nebo splnění dialogového úkolu. Možnost této interakce je znázorněna pomocí dočasného zobrazení malého spritu nad hlavou hráče (pop up). Poté co dialog proběhne, tak je hráči přidělen úkol a tento dialog už nemůže být aktivován znovu (díky zvýšení hodnoty „`ID dalšího úkolu`“).

Chování tohoto pop up objektu a jiných objektů řízených nějakými herními událostmi bylo zpracováno za pomocí ručně vytvořeného systému signálů. Tento systém byl navržen na základě návrhového vzoru Observer[18] (pozorovatel). Pokud pozorovatelé implementování ve skriptu `SignalListener.cs` (dědící ze třídy herního objektu) přijmou signál, tak spustí zabudovaný UnityEvent systém, na němž může být nastaveno volání jakékoliv funkce z přiřazeného objektu. Pozorovaný signál, který je implementován ve skriptu `SignalSender.cs` (dědící z Unity třídy ScriptableObject), kde přidává a odebírá pozorovatele do svého seznamu. Pokud je někde v ostatních skriptech pozorovaný signál vyvolán, tak zavolá na všechny své pozorovatele, které má v seznamu.

Za pomocí Unity třídy pro lepší práci s daty ScriptableObject a rozhraní pro serializaci byly vytvořeny pomocné skriptovací objekty, které byly využity pro ukládání a sdílení dat

[18] Návrhový vzor Observer (https://www.itnetwork.cz/navrh/navrhove-vzory/gof/observer-pozorovatel-navrhovy-vzor/)

nezávislých na konkrétní scéně. Byly vytvořeny tři typy těchto skriptovacích objektů: float, int a string. Všechny mají dvě hodnoty. Běhovou (runtime) hodnotu, která je po deserializaci naplněna inicializační hodnotou. Dohromady bylo vytvořeno dvanáct skriptovacích objektů. Využity byly pro správu hodnot, např. zdraví a poškození nepřátelského NPC a hráče, postupu v aktuálním úkolu, ID dalšího úkolu apod. Tyto skriptovací objekty byly taktéž využity pro vypsání určitých hodnot na HUD.

Poté, co hráč pomocí klávesy „E“ odkliká několik textových zpráv, z nichž se dialog s přátelským NPC skládá, tak je poslední textová zpráva z tohoto dialogu převedena na text zadání úkolu. Úkoly mohou zadat pouze přátelská NPC skrze skript `Dialog.cs`, ve kterém se nastavují sdílené informace, vyvolává se úkolový signál, a také se zde nachází další potřebné informace k úkolu ve formě objektu `Quest.cs`, který je následně předán hráči. Ve skriptu `Dialog.cs` je také vyhodnocován (nastavuje hráči odměnu) dialogový typ úkolů. Cílem tohoto typu úkolu je, aby hráč došel k dalšímu dialogu u jiného přátelského NPC. Splnění klasického typu úkolu je vždy podmíněno zabitím určitého počtu konkrétního nepřátelského NPC. Tuto hodnotu představuje skriptovací objekt `currentQuestValue`, který je porovnáván s číselnou proměnnou `requiredQuestValue` v metodě `GoalIsReached`, kterou je možné vidět ve zdrojovém kódu objektu `Quest.cs` níže (deklarace using byly vynechány).

```
[System.Serializable]
public class Quest
{
    public int questId;
    public IntValue currentQuestValue;
    public StringValue currentQuestMessage;
    public SignalSender questCompleteSignal;
    public string requiredEntity;
    public int requiredQuestValue;
    public int experienceReward;

    public bool GoalIsReached()
    {
        return (currentQuestValue.runtimeValue >= requiredQuestValue);
    }

    public void UpdateQuestValue(string entity)
    {
        if(requiredEntity == entity) currentQuestValue.runtimeValue++;
    }

    public void Complete()
    {
        questCompleteSignal.Raise();
        currentQuestMessage.runtimeValue = "";
        currentQuestValue.runtimeValue = 0;
    }
}
```

Objekt `Quest.cs` dále obsahuje metodu `UpdateQuestValue` pro aktualizaci hodnoty `currentQuestValue` v případě, že je splněna podmínka na požadovanou entitu (řetězec „NPC“ v případě dialogového typu úkolu nebo u klasického typu název nepřátelského NPC). A také obsahuje metodu `Complete` pro vyvolání signálu „úkol splněn“ a vymazání sdílených hodnot. O zbytek úkolové funkcionality se stará hráčský skript `PlayerController.cs`. V něm jsou při zabití nepřátelského NPC volány výše zmíněné metody objektu `Quest.cs` a také se v něm při splnění úkolu přičítá odměna. Ta je realizována pomocí přičtení bodů zkušeností v metodě `giveXP`. Za dosažení 100 bodů XP je hráči přičten jeden level, se kterým se hráči pomocí metody `improveStats` postupně střídavě zvyšují statistiky maximálního zdraví a poškození (a také je hráčovo aktuální zdraví nastaveno na maximální zdraví).

Tyto statistiky a také aktuální hodnoty hráčského počtu levelů, XP, životů a textu aktuálního úkolu jsou vizuálně znázorněny pomocí HUD. Ten je primárně tvořen ručně vytvořeným assetem, na kterém jsou tyto hodnoty textově zobrazeny i za pomoci dvou ukazatelů. Tyto ukazatele pro aktuální hodnotu zdraví a počtu XP jsou herní objekty, které jsou potomky scénového objektu Canvas (se souřadnicemi obrazovky), a které obsahují komponentu Slider. Nastavování hodnot pro HUD zajišťují `SliderBarManager.cs` a `QuestStatHUDManager.cs`, což jsou skripty, které v reakci na zavolání signálů propisují sdílené hodnoty skriptovacích objektů do hodnot konkrétních HUD objektů. V rámci tohoto Canvasu je realizována také obrazovka „konec hry“. Ta je tvořena nápisem se zdůvodněním konce hry a také dvěma tlačítky. Tlačítka „Oživit“ (pro oživení hráčské postavy na výchozím bodě mapy) a „Zpět do menu“, která jsou ovládána skriptem `MenuScreenController.cs`.

Zřejmě nejnáročnější bylo vypracování soubojového systému. Ten u hráče představoval vytvoření a nastavení nových animací. Obsluhu přepínání hráčových animací zajišťuje skript `PlayerController.cs`. V něm byla vytvořena kooperační rutina, jenž je volána při aktivaci příslušné klávesy pro útok „mezerník“, přičemž nastavuje stavy a animace, ve kterých se hráč aktuálně nachází. Mapování (přiřazení) kláves je v Unity poměrně ukryto ve správci vstupů, který se nachází v nastavení projektu. Hráč se tak díky tomuto přepínání stavů, jenž realizuje jako reakci na vnější vstupy podobá stavovému automatu. Stavů má celkem pět: nečinný, chůze, útok, interakce a vrávorání. Do stavu vrávorání se hráč dočasně dostane, pokud je zasažen útokem nepřátelského NPC (vyhodnocení probíhá ve skriptu `Hit.cs`).

V tomto případě je hráči snížen počet životů o hodnotu poškození od nepřátelského NPC, a pokud je počet životů na nule, tak hráč zemřel a aktivuje se obrazovka „konec hry“. Poté může být hráč „oživen“ pomocí metody `respawn`. Pokud hráč nezemřel, tak se spustí kooperační rutina, jenž hráčskému Sprite Rendereru dočasně nastaví ručně vytvořený materiál `FlashMaterial`, který svou bílou barvou vizualizuje efekt problesknutí pro znázornění zranění. Stav vrávorání (způsobený zasažením) má i nepřátelské NPC a jeho nastavení má na starosti stejný skript `Hit.cs` jako u zasažení hráče. Tento skript vyhodnocuje kolize trigger kolizních boxů, a poté působí na cílené Rigidbody silou určenou hodnotou „zpětného rázu“, což způsobí odstrčení cílené zasažené entity (nepřátelského NPC či hráče).

Pro funkcionalitu hráčského útoku jsou nutné objekty trigger kolizních boxů. Tyto objekty jsou v hierarchii potomci hráčského objektu (hned vedle objektu hlavní kamery a pop up objektu) a ve scéně jsou umístěny na každou stranu hráčského spritu. Každý z těchto čtyř trigger kolizních boxů je tvořen z komponenty Polygon Collider 2D a skriptu `Hit.cs`. Samotný hráčský objekt dále obsahuje dosud nezmíněné komponenty Box Collider 2D (s trigger parametrem) pro kolizi zasažení nepřátelským NPC a skripty pozorovatelů pop up signálu a signálu zabitého nepřátelského NPC.

Obecné nepřátelské NPC je v mnohém velmi podobné hráči. Mezi jeho obecně využívané komponenty patří Sprite Renderer, Animator, Rigidbody 2D, Box Collider 2D, Box Collider 2D (s trigger parametrem) pro kolizi zasažení hráčem a skripty `EnemyController.cs` a `Hit.cs`. V něm se nachází řízení chování nepřátelského NPC, a to ať už z hlediska přepínání stavů (obdobné jako u hráče, akorát bez stavu interakce) nebo řízení svých animací, pro které se směr animace vypočítává na základě předchozího pohybu.

Chování nepřátelského NPC je implementováno třemi způsoby. Buďto se pohybuje směrem k hráči (jestliže je hráč v nastaveném dosahu) s cílem do něj narazit nebo na něho zaútočit (v obou případech způsobí hráči zranění), anebo se pohybuje směrem k následujícímu hlídkovacímu bodu. Tyto body jsou neviditelné herní objekty umístěné různě po scéně, podle kterých může nepřátelské NPC navigovat svůj pohyb. Avšak pohyb k hráči má vždy vyšší prioritu. Kooperační rutina s efektem zranění a metoda pro vyhodnocení zasažení jsou podobné s těmi u hráče.

Při smrti nepřátelského NPC (před tím, než je ze scény jeho objekt smazán) je na jeho místo do scény dočasně přidán prefab se Sprite Renderer a Animatorem s animací imitující efekt pro smrt tohoto NPC. Kromě hlavní scény byly do hry implementovány další dvě pomocné

scény. Jedna scéna pro hlavní menu hry a druhá pro titulky. Obě obsahují kromě kamery objekt Canvas s obrázkem na pozadí, objekt Text (TMP) a tlačítka řízená `MenuScreenController.cs` skriptem. Scéna pro titulky obsahuje ještě další objekty Text (TMP) v objektu Image pro výpis titulků. V hlavní scéně se také nachází další objekty pro dekorační účely.

Poté bylo ještě nutné zapracovat do hry systém starající se o audio složku hry. První část tohoto systému spočívá v ručně vytvořených objektech umístěných přímo do scény. Prvním z nich je MusicManager, který je složen z komponenty Audio Source a dvou `SignalListener.cs` skriptů. Druhý objekt SoundManager obsahuje pět těchto skriptů a ovládá pomocí nich pět svých potomků (objektů), které mají komponentu Audio Source s nastaveným zvukem. O druhou část systému se starají samotné objekty `Player` a přátelské NPC, jenž mají své vlastní Audio Source komponenty. Úplně na konec produkční fáze byly do hry rozmístěny a nastaveny výše zmíněné objekty, a také proběhlo zpracování závěrečného souboje, aby se tak naplnila představa o herní náplni a samotném průchodu hrou.

6.4 Testovací fáze

Při testovací fázi se odhalilo mnoho problémů a nedostatků, které bylo následně nutné prozkoumat a najít jejich nejlepší možné řešení. Vzhledem k povaze vyvíjeného projektu probíhalo testování v rámci všech vývojových iterací a pouze manuálně formou průběžného ověřování funkčnosti jednotlivých funkcionalit. Další využitou formou testů byly lokalizační testy, které odhalily některé gramatické chyby v dialozích u přátelských NPC.

Jedním z prvních nalezených problémů byl tzv. tilemap tearing (trhání). Systém, jakým jsou v Unity vykreslovány tilemapy, má již po několik verzí jednu poměrně nepříjemnou vizuální chybu. Ta se projevuje náhodnými grafickými glitchemi v podobě malé mezery mezi dvěma náhodnými tily. Výskyt tohoto glitche je velmi sporadický, avšak může působit rušivě. Řešením tohoto problému může být nastavení drobné negativní hodnoty pro velikost mezery mezi jednotlivými tily v Gridu (to, ale může způsobovat další problémy). Nebo nepoužitím Unity tilemap systému a vygenerování těchto tilemap v externím programu (Tiled Map Editor[19] a následné pře-generování do Unity pomocí SuperTiled2Unity[20]). Třetí a při vývoji zvolenou opravou bylo nasazení speciálního materiálu na každou použitou tilemapu. Tento materiál má nastavené hodnoty shaderu na Sprites/Default a aktivní režim pixel snap.

[19] Tiled Map Editor (https://www.mapeditor.org/)
[20] SuperTiled2Unity (https://seanba.com/supertiled2unity.html)

Mezi další chyby nalezené při testování patřilo rozmazávání naimportovaných spritů. To bylo vyřešeno globálním vypnutím techniky antialiasingu v nastavení kvality projektu, jelikož je její použití u tohoto typu pixel artové grafiky nevyhovující. To však tento problém zcela nevyřešilo, jelikož byly pozorovány drobné grafické glitche při animacích spritů. Ty byly odstraněny vypnutím automatické komprese a přenastavením módu filtrování assetů na možnost point (tedy bez filtrování) v nastavení zdrojových spritů. Tyto sprity jsou relativně malé, a proto u nich zakázání komprese nepředstavuje problém.

S tímto však souvisí další problém, a to je rozmazávání zobrazovaného textu v objektech Text. Tento problém byl nejprve potlačen, již zmíněným zákazem antialiasingu, zvětšením vzorkování a dalšími úpravami fontu. To však u malých velikostí fontu nestačilo, a tak byl využit vylepšený systém pro vykreslování textu pomocí speciálního objektu Text (TMP), který se nachází v externím přídavném Unity balíčku TextMeshPro. S tím však souvisí další problém, a tím je proměnlivá velikost fontů, která je způsobena různou délkou textů v dialogovém okně a automatickým výpočtem velikosti fontů. Tento čistě vizuální problém nemá jednoduché řešení, a vzhledem k jeho malé prioritě ve vydané verzi hry přetrval.

Dalšími drobnými chybami byly špatně nastavené přechody animací, časování pomocné rutiny ve skriptu `PlayerController.cs` při animaci hráčského útoku a chybně nastavené opakování této animace. Další chybou bylo využití nefixovaného deltaTime v tom samém skriptu, což při pohybu hráče diagonálně způsobilo značné zrychlení pohybu.

6.5 Beta fáze

Poté co hra již byla po stránce funkcionalit, tak po stránce assetů kompletní, tak se vývoj přesunul do předposlední fáze Beta. V této fázi Beta testování probíhalo mimo jiné také formou uživatelského testování a testování kompatibility (optimalizace). Na základě uživatelské zpětné vazby byly odhaleny další i poměrně zásadní chyby. Jednou takovou bylo špatné zastavování hráče a objektů po přidání síly „zpětného rázu“. To bylo u hráče vyřešeno dočasným zmražením Rigidbody 2D a u objektů (neinteraktivních NPC) jeho odstraněním. Další chybou bylo posunutí hlídkovacích bodů po ose Z do negativních hodnot, jenž vzniklo nedopatřením, a které vyústilo v nefungující hlídkování nepřátelských NPC.

Vzhledem k rozsahu projektu, zaměření této práce a dostupným zdrojům probíhalo testování odpovídajícím způsobem. Uživatelské testování tak proběhlo pouze po dobu asi jedné hodiny, na čtyřech lidech s různými herními zkušenostmi, jejichž ústní zpětná vazba poté byla zaznamenána do listinných poznámek. Obdobně je na tom i seznam testovaného HW,

jenž je složen z poměrně výkonově rozmanitých HW sestav, které měl autor k dispozici. I přes tyto omezené zdroje přinesly provedené testy užitečné výsledky.

Uživatelská zpětná vazba přinesla mimo odhalení herních chyb, také poznatky v oblasti zábavnosti a optimalizace herního zážitku. Byla přenastavena hlasitost jednotlivých zvukových efektů. Dále bylo mírně upraveno GUI hry, aby zobrazovaný text působil výrazněji a byl přidán popisek vysvětlující ovládání hry do hlavního menu. Mírně byla vylepšena samotná mapa, zejména přidáním některých dekorativních prvků, které mají navíc navigační charakter pro hráčovu lepší orientaci ve světě. A poté byl vytvořen speciální fyzikální materiál slippery přiřazený objektu `Player` pro klouzavější pohyb hráče okolo kolizních boxů ve scéně.

Nakonec této fáze proběhlo testování kompatibility a optimalizace. HW a SW sestavy a výsledky testů lze vidět v níže uvedené tabulce 1. Pro optimalizační testy byla využita různorodá výkonnostní škála HW. Rozsahově omezené testy kompatibility zahrnovaly spuštění hry a její hraní na různých verzích OS Windows 10 (ve verzích 21H1 a 21H2) a na linuxové distribuci Xubuntu ve verzi 20.04.3 LTS. Optimalizační testy probíhaly formou hraní po dobu jedné minuty za pomocí FPS měřícího SW Fraps[21] a Steam[22]. A kvůli výsledkům přesahujícím 60 Hz obnovovací frekvenci obrazovek proběhly na buildech s vypnutou funkcí vertikální synchronizace.

Všechny testy na OS Windows proběhly v pořádku, ale spuštění linuxového buildu občasně selhalo. Tuto proměnlivou stabilitu hry při jejím spuštění, zřejmě způsobil zvolený testovací SW Steam. Bez něj proběhlo spuštění hry vždy v pořádku. Výsledky testů odhalily, že hra běží bez větších technických problémů i na starších HW sestavách a průměrný počet FPS vysoce převyšující průmyslový standard 60 FPS (pro monitory s obnovovací frekvencí 60 Hz) značí, že není potřeba hru, jakkoliv dále optimalizovat.

Tabulka 1 - Kompatibilní a optimalizační testy[23]

	OS	CPU	RAM	GPU	SSD	Průměr FPS
PC	Win. 10 21H2	Intel Core i5 - 6600K	16 GB	Nvidia GeForce GTX 1060	ANO	1320
Notebook	Win. 10 21H2	AMD Ryzen 5 - 5600H	8 GB	Nvidia GeForce GTX 1650	ANO	365
Notebook	Ubuntu 20.04.3	Intel Core i5 – 2430M	6 GB	Nvidia GeForce GT 540M	NE	280
PC	Win. 10 21H1	AMD Athlon II X2 240	4 GB	Nvidia GeForce 310	NE	240

[21] Fraps (https://fraps.com/)
[22] Steam (https://store.steampowered.com/)
[23] OS s označením Win. představují OS Microsoft Windows v edici Pro

6.6 Dokončovací fáze

Jakmile testy proběhly s pozitivními výsledky (viz. předchozí podkapitola 6.5), tak byl vývoj hry uzavřen a bylo provedeno už jen poslední nastavení projektu v Unity pro proces buildování hry. Byl zvolen výsledný název a upravilo se nastavení výsledné kvality. Následně bylo nastaveno pořadí scén, byla zvolena cílová platforma a architektura (pro buildování na Linux bylo zapotřebí stáhnout přídavný buildovací modul), a poté již byl vytvořen build (tzv. gold master), který byl uložen do zvolené složky. Do ní byla následně přidána předem vytvořená složka Attribution obsahující textové soubory s popisem assetů a jejich autorských práv. Poté už byla jen cílená složka s buildem zkomprimována do archívu zip.

Posledním krokem vývojového procesu už bylo jen samotné vydání hry na službě itch.io. Tato webová služba nabízí uživatelům hosting, prodej a další služby v souvislosti s primárně nezávislými hrami (včetně rozsáhlé nabídky asssetů). Itch.io je pro běžné užití zcela zdarma, nepřebírá autorská práva, nevynucuje použití DRM, nevkládá nevyžádané reklamy a pouze pokud je hra na službě prodávána, tak si bere nastavitelný procentuální podíl ze zisku. K vydání hry nabízené zdarma, což bylo autorovým záměrem, je zapotřebí pouhá registrace na této službě. Při vytváření nového projektu, což je webová stránka hostovaná na itch.io (na které je poté možnost stažení hry), bylo zapotřebí stanovit titulek, URL, krátký popis, cover obrázek, typ projektu, status vydání a údaje o platbě (nastaveny na 0 $). Následně proběhlo samotné nahrání zmíněného zip archívu a doplněn popisek. Poté už byl jen nastaven žánr hry na adventuru a hře byly přiřazeny tagy, např. akční adventura, top-down, retro, pixel art apod. Po uložení údajů byla zobrazena projektová stránka a byla hra vydána (viz. obr. 12).

Obrázek 12 - Projektová stránka s možností stažení hry na itch.io

6.7 Zhodnocení vývoje a ukázka hry

Vývoj byl rozděl do klasických vývojových fází GDLC, jejichž průběh byl postupem vývoje dokumentován. Oproti vývoji her „normálního“ či velkého rozsahu v týmu sestávajícím se z mnoha vývojářů, byl tento konkrétní vývojový proces, jak již bylo uvedeno v podkapitole 6.1, mírně zjednodušen. To však neznamená, že byl vývojový proces jednoduchý. Právě naopak zejména kvůli velmi omezeným zdrojům a nezkušenosti autora práce s vývojem hry tohoto typu, byl vývoj velmi náročný. Největší chybou bylo autorovo podcenění komplexnosti některých herních systémů a mechanik. Problém způsobilo také vybrané grafické zpracování hry. Mnoho času bylo věnováno hledání použitelných vizuálních assetů, a to zejména u NPC, které v tomto grafickém zpracování potřebují rozdílené assety (včetně animací) pro 4 směry.

V rámci celého vývojového procesu byly uplatněny principy z upravené metodiky Kanban (viz. podkapitola 6.2). Zvolení této metodiky a provedení již zmíněných drobných úprav pro použití v rámci jednočlenného týmu bylo velmi přínosné. Principy této metodiky výrazným způsobem pomáhaly zvládat řízení práce během celého náročného vývojového procesu. Zejména principy: vizualizace pomocí štítkové tabule v rámci webové služby Trello, omezení rozpracované práce, zaměření se na celek a rychlá reakce na změny, byly velmi přínosné.

Pro konkrétní praktickou ukázku vývoje byl zvolen herní engine Unity ve verzi 2020.3.14f1 LTS. V této verzi Unity probíhal vývoj, až na pár nedostatků, pohodlně a bez větších problémů. I přes výskyt občasných chyb v této verzi, bylo z důvodu nepodložené obavy nekompatibility s již vyvíjeným projektem rozhodnuto neaktualizovat na novější verzi. Největšími nedostatky byla občas problémová integrace IDE Microsoft Visual Studio, občasná poplašná chybová hlášení a také dlouhodobě vyskytující se grafické glitche u tilemap. Pozitivní věcí na Unity bylo snadné ovládání včetně podpory drag&drop u vstupních parametrů herních objektů a skriptů. Dále také rychlé spouštění herního náhledu a jednoduchá práce s nástroji.

Výslednou vyvinutou hru bylo možné plně hrát s herním obsahem dvanácti jednoduchých úkolů, včetně obtížnějšího závěrečného souboje. Pro hru byl vytvořen originální, ručně vytvořený svět pomocí systému tilemap a s vizuálním stylem pixel art s pohledem typu top-down (ze shora dolů). Při vytváření hry bylo použito 26 audiovizuálních assetů či sad assetů (tilesety) třetích stran. Kromě nich byly veškeré další použité assety výsledkem tvůrčí

činnosti autora, za použití programů Piskel[24], Adobe Photoshop[25] a Audacity[26]. K těmto autorem práce vytvořeným assetům lze také přidat 13 skriptů napsaných v jazyce C# a další assety vytvořené přímo v herním enginu Unity.

Výsledná hra byla veřejně vydána pomocí webové služby itch.io, kde je volně k dostání ve formě stažitelného archivu zip. Po rozbalení tohoto archivu již není potřeba jakákoliv další instalace a hra je po zapnutí spustitelného souboru plně hratelná. Hru je možné nalézt na službě itch.io skrze vyhledávání pod názvem „Středověké království“, popřípadě pomocí zadání této URL: https://george222cz.itch.io/medieval-kingdom [27] do webového prohlížeče.

Pro ukázku vytvořené hry formou následujících obrázků byly zvoleny takové herní okamžiky, které zachycují významné herní mechaniky či systémy. Na obrázku č. 13 níže lze vidět scénu s hlavním menu hry, která zachycuje výřez hlavní mapy hry na pozadí a také obsahuje jednoduchý popis ovládání.

Obrázek 13 - Hlavní menu hry

Další obrázek č. 14 zachycuje scénu na samotném začátku hry, při které dostává hráč první úkol, skrze dialog s přátelským NPC. Lze si také všimnout zpracování HUD, které textově zobrazuje počet levelů, maximálních životů i poškození a formou dvou ukazatelů počet aktuálních životů a XP.

[24] Piskel (https://www.piskelapp.com/)
[25] Adobe Photoshop (https://www.adobe.com/cz/products/photoshop.html)
[26] Audacity (https://www.audacityteam.org/)
[27] Projektová stránka se hrou na itch.io (https://george222cz.itch.io/medieval-kingdom)

Obrázek 14 - Zadání úkolu skrze dialog s přátelským NPC

Na tomto obrázku č. 15 je vidět zadání a postup aktuálního úkolu, a především také souboj s právě zasaženým goblinem (nepřátelské NPC s viditelným efektem zranění). Také je zde vidět funkce ukazatelů aktuálního počtu zdraví, XP a dalších statistik herní postavy.

Obrázek 15 - Souboj s nepřátelským NPC

7 ZÁVĚR

V diplomové práci byla popsána problematika vývoje videoher v celém svém rozsahu se zaměřením na vývoj počítačových her, a díky využití vhodné konkrétní metodiky, dokázala proveditelnost vývoje relativně komplexního herního projektu s omezenými dostupnými zdroji. K tomu byly využity znalosti získané v rámci samostudia a studia na vysoké škole, zejména pak z předmětů: Pokročilé techniky programování, Počítačová grafika 2D, Teoretická informatika, Skriptování v 3D grafických softwarech a Projektování SW systémů.

Nejprve byly v diplomové práci vysvětleny důležité pojmy, týkající se odvětví herního vývoje. Dále pak byla objasněna specifika herního vývoje, jako jsou vývojové požadavky, herní design a podoba herních studií z hlediska vývojových týmů a rolí konkrétních vývojářů. Následně byly v práci vysvětleny modely a metodiky vývojového procesu s větší mírou zaměření na jednotlivé agilní metodiky. V práci bylo taktéž uvedeno, jak vypadá herní vývoj v praxi, zejména v rámci použitých metodik a vývojového cyklu, nastalých problémů během vývoje, používaných nástrojů a technologií. Poté byl představen herní engin Unity a zaměření práce bylo přesunuto z teoretické části do části praktické.

V rámci praktické části práce byly zdokumentovány všechny fáze herního vývojového procesu, který byl postaven na principech vhodně zvolené upravené metodiky Kanban, jejíž praktická ukázka byla hlavním cílem této části. Samotnému procesu tvorby v herním enginu Unity předcházelo plánování hry, které bylo shrnuto v zahajovací fázi a částečně uceleno ve zjednodušené formě Koncept dokumentu. Dalšími důležitými prvky vývojového procesu bylo vytvoření štítkové tabule ve webové službě Trello a následný sběr assetů. Tyto dokumenty vzniklé během vývoje byly zařazeny do diplomové práce jakožto přílohy.

Výsledným produktem vývojového procesu byla konkrétní ukázková počítačová hra ve vizuálním stylu zpracování 2D tilemap. Vzniklá počítačová hra byla dostatečným prostředkem pro otestování teoretických znalostí uvedených v první části práce a praktických principů zvolené metodiky. V rámci vývoje v herním enginu Unity bylo zapotřebí využití jazyka C# pro tvorbu skriptů, které řídí chování hry. A také bylo třeba využít dalšího SW jako např. Piskel, Adobe Photoshop a Audacity pro vlastní tvorbu či úpravu assetů. Hra byla otestována a v aktuální verzi je plně hratelná.

Na současný stav vývoje výsledné hry by se v budoucnu mohlo navázat, např. vylepšením soubojového systému, prohloubením herního obsahu, zvětšením míry ovlivňování vývoje herní postavy, anebo tvorbou nových systémů výroby předmětů a správy či stavění budov.

8 SEZNAM POUŽITÝCH ZDROJŮ

1. **TechnologyAdvice.** Definitions Archives - Webopedia. *Webopedia.* [Online] 2022. [Citace: 9. 2 2022.] https://www.webopedia.com/definitions/.

2. **Techopedia.** Technology Dictionary. *Techopedia.* [Online] 2022. [Citace: 9. 2 2022.] https://www.techopedia.com/dictionary.

3. **Unity Technologies.** Unity - Manual. *Unity Documentation.* [Online] 2022. [Citace: 28. 6 2021.] https://docs.unity3d.com/2020.3/Documentation/Manual/index.html.

4. **Lewis, Michael a Jacobson, Jeffrey.** Game Engines In Scientific Research. *Computer Science & Engineering.* [Online] 2002. [Citace: 22. 8 2021.] https://www.cse.unr.edu/~sushil/class/gas/papers/GameAIp27-lewis.pdf.

5. **Bethke, Erik.** *Game Development and Production.* Plano : Wordware Publishing, 2003. ISBN: 1556229518.

6. **Jirkovský, Jan.** *Game industry: vývoj počítačových her a kapitoly z herního průmyslu.* Praha : D.A.M.O., 2011. ISBN 978-809-0438-712.

7. **Unity Technologies.** 10 game design tips for new developers. *Unity.* [Online] 2021. [Citace: 15. 7 2021.] https://unity.com/how-to/beginner/10-game-design-tips-new-developers.

8. **Bates, Bob.** *Game design.* 2nd ed. Boston : Premier Press, 2004. ISBN 9781592004935.

9. **Novak, Jeannie.** *Game development essentials.* Third edition. New York : Delmar, 2012. ISBN 1111307652.

10. **Gunderloy, Mike.** *Z kodéra vývojářem.* Brno : Computer Press, 2007. ISBN: 8025115178.

11. **Norita, Ahmad B.** How to launch a successful video game: A framework. *Entertainment Computing.* [Online] 2017. [Citace: 27. 4 2020.] https://www.sciencedirect.com/science/article/abs/pii/S1875952117300861. ISSN: 1875-9521.

12. **Martinů, Jiří a Čermák, Petr.** METODIKY VÝVOJE SOFTWARE. [Online] 2018. [Citace: 26. 9 2021.] https://dl1.cuni.cz/pluginfile.php/864918/mod_resource/content/1/Metodiky-v%C3%BDvoje-software-studijn%C3%AD-text.pdf.

13. *Agilní metodiky.* **Buchalcevová, Alena.** Praha : Česká zemědělská univerzita v Praze ČZU, 2002. Konference OBJEKTY 2002. ISBN: 80-213-0947-4.

14. **Smartsheet.** Project Management Guide. *Smartsheet.* [Online] 2021. [Citace: 2. 10 2021.] https://www.smartsheet.com/content-center/best-practices/project-management/project-management-guide .

15. **Kolektiv.** *Agile Manifesto.* [Online] Utah : Agile Alliance, 2001.

16. **Fojtík, Rostislav.** Extreme Programming in development of specific software. *ScienceDirect.* [Online] 2011. [Citace: 21. 9 2021.] https://www.sciencedirect.com/science/article/pii/S1877050911000330.

17. **Ebert, Christof, Pekka, Abrahamsson a Nilay, Oza.** Lean Software Development. *Vector.* [Online] 2012. [Citace: 25. 10 2021.] https://cdn.vector.com/cms/content/consulting/publications/Ebert_LeanDevelopment_Intro_IEEESoftware2012.pdf. ISSN: 0740-7459.

18. **Šochová, Zuzana a Kunce, Eduard.** *Agilní metody řízení projektů.* 2. vydání. Brno : Computer Press, 2019. ISBN 978-80-251-4961-4.

19. **Kristiadi, Dedy Prasetya, Sudarto, Ferry a Sugiarto, Dian.** Game Development with Scrum methodology. *IEEE Xplore.* [Online] 2019. [Citace: 21. 10 2021.] https://ieeexplore.ieee.org/document/9144963. ISBN: 978-1-7281-5862-4.

20. **RAMADAN, Rido a WIDYANI, Yani.** Game development life cycle guidelines. *IEEE Xplore.* [Online] 2013. [Citace: 26. 6 2019.] http://ieeexplore.ieee.org/document/6761558/. ISBN: 978-1-4799-4692-1.

21. **Pickell, Devin.** The 7 Stages of Game Development. *G2.com.* [Online] 2019. [Citace: 27. 4 2020.] https://www.g2.com/articles/stages-of-game-development.

22. **Liming, Drew a Vilorio, Dennis.** Work for Play: Careers in Video Game Development. *ERIC.* [Online] 2011. [Citace: 21. 2 2020.] https://eric.ed.gov/?id=EJ945968. ISSN 0199-4786.

23. **GDACZ.** ČESKÉ POČÍTAČOVÉ HRY. [Online] 2019. [Citace: 27. 6 2019.] https://gda.cz/wp-content/uploads/2019/03/CeskePocitacoveHry.pdf.

24. **Piacentini, Alessandro.** The Effects of Piracy and Counterfeiting in the Video Games Industry. [Online] 2018. [Citace: 27. 4 2020.] https://tesi.luiss.it/21320/1/665731_PIACENTINI_ALESSANDRO.pdf.

25. **Tian, HaiYun, Bosser, Anner-Gwenn a Brooke, Phillip J.** Behaviour-Based Cheat Detection in Multiplayer Games with Event-B. *Formal Methods.* [Online] 2012. [Citace: 27. 4 2020.] https://link.springer.com/chapter/10.1007/978-3-642-30729-4_15. ISBN: 978-3-642-30729-4.

26. **Corrales, Enrique.** Best Video Game Development Tools. *developer.com.* [Online] TechnologyAdvice, 2021. [Citace: 5. 3 2022.] https://www.developer.com/languages/best-video-game-development-tools/.

27. **Rasheva, Silvia.** Introducing Enemies: The latest evolution in high-fidelity digital humans from Unity. *Unity.* [Online] Unity Technologies, 2022. [Citace: 23. 4 2022.] https://blog.unity.com/news/introducing-enemies-the-latest-evolution-in-high-fidelity-digital-humans-from-unity.

9 SEZNAM PŘÍLOH

Přílohy byly uloženy externě a jsou dostupné online na portále informačního systému STAG Univerzity Pardubice a na kompaktním disku, jenž je součástí vytištěné diplomové práce.

- Zdrojové soubory vypracované hry (v archivu zip)
- Finální build vypracované hry (spustitelný soubor Středověké království.exe v archivu zip)
- High Concept dokument
- Zjednodušený Koncept dokument
- Project plan dokument (zpracovaný pomocí služby Trello)