DELTA – Střední škola informatiky a ekonomie s.r.o.,
Ke Kamenci 151, Pardubice



Harvest Grove

Příjmení, jméno: Milanů Alex
Třída: 4.A
Studijní obor: Informační technologie 18-20-M/01
Školní rok: 2023/2024
Vedoucí projektu: Mgr. Jan Mottl

Zadání maturitního projektu

Jméno a příjmení:	Alex Milanů
Pro školní rok:	2022/2023
Třída:	3.A
Obor:	Informační technologie 18-20-M/01
Téma práce:	2D Top-down hra v Unity
Vedoucí práce:	Mgr. Jan Mottl

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Hra ponese název Harvest Grove – hráč se objeví na procedurálně generovaném ostrově, kde jeho úkolem je nasbírat materiál, který potřebuje na postavení farmy. Na začátku hry je daný úkol, ve kterém hráč musí najít přístav, který se náhodně objeví někde na ostrově. Po opravení přístavu se otevřou nové možnosti na cestování – další 3 ostrovy, z nichž jeden ostrov je menší město, ve kterém hráč může nakupovat / prodávat zboží nebo nakoupit nová, lepší semínka na zeleninu / ovocné stromy. Jedná se již o započatý projekt, který má za sebou vlastní větší tým – 2 programátory, 3 grafiky a 1 skladatele hudby. V rámci maturitního projektu beta verzi hry dokončím. V dokumentaci bude jasně vytyčeno, na čem jsem pracoval čistě já a na čem pracovali ostatní členové týmu. Hra je tvořena v enginu Unity a skripty budou psané v C# .

Časový harmonogram:

Září – Implementace budov a jejich funkcionalit; Předělání systému na růst zeleniny / stromů / keřů na Job System od Unity
Říjen – Zvířata: Cyklus života, jsou schopni dávat věci, jsou schopni být rozmnoženi
Listopad - Health system pro hráče; Útočící zelenina; Vaření; Saving system – možnost načíst a vytvořit novou hru
Prosinec – Quest system, Dialogue system, Implementace grafiky / audia
Leden – Beta verze hotová – testování hry
Únor – Ladění nalezených chyb a finalizace hry samotné; Začátek práce na dokumentaci
Březen – Dokumentace projektu

Prohlášení

Prohlašuji, že jsem maturitní projekt vypracoval samostatně, výhradně s použitím uvedené literatury.

V Pardubicích 2.4. 2024

Poděkování

Děkuji Mgr. Janu Mottlovi za odborné vedení při zpracovávání maturitního projektu. Dále bych rád poděkoval celému vývojářskému týmu a beta testerům. Jmenovitě bych pak zejména poděkoval Alexovi, který se mnou na hře pracuje od úplného začátku.

Anotace

Dokumentace se zaobírá 2D farmářskou hrou vytvořenou malým mezinárodním týmem pomocí herního enginu Unity. Podrobněji zkoumá technickou inspiraci jinými hrami a obsahuje game design dokument.

Klíčová slova

Hra, 2D, top-down, Unity, C#, farmaření, pixelart

Annotation

The documentation covers a 2D farming game created by a small, international team using the Unity engine. It explores in depth the technical inspiration from other games and includes a game design document.

Key words

Game, 2D, top-down, Unity, C#, farming, pixelart

Obsah

1 Úvod **1**

2 Týmová práce **2**

2.1 Má práce 2

3 Využité technologie **3**

3.1 C# 3

3.1.1 Mono vs ILL2CPP 3

3.2 Unity 3

4 Využité nástroje **4**

4.1 Visual Studio Code 4

4.2 Git 4

4.3 FMOD studio 4

5 Inspirace dalšími hrami **5**

5.1 Stardew Valley 5

5.1.1 Microsoft XNA 5

5.1.2 Monogame vs Unity 6

5.1.3 Důvod našeho rozhodnutí 6

5.2 Minecraft 7

5.2.1 Generace světa 7

5.2.1.1 Biomy 8

5.2.1.2 Šum 8

5.2.1.3 Povrch 9

6 UML **10**

6.1 Usecase diagramy 10

6.2 Class diagramy 12

7 Game design dokument **13**

7.1 Přehled hry 13

7.1.1 Koncept hry 13

7.1.2 Základní herní mechanismy 13

7.1.3 Žánr 13

7.1.3.1 Definice žánru . . . 13
7.1.4 Cílová skupina . . . 14
7.1.5 Atmosféra hry . . . 14
7.2 Gameplay a mechaniky . . . 15
7.2.1 Herní progres . . . 15
7.2.2 Cíle hry . . . 15
7.2.3 Struktura úkolů . . . 16
7.2.4 Ukládání . . . 17
7.2.5 Časové úseky . . . 18
7.2.6 Úmrtí . . . 18
7.3 Inputy . . . 19
7.3.1 Pohyb a myš . . . 19
7.3.2 Ostatní inputy . . . 19
7.4 Oblasti . . . 20
7.4.1 Domovský ostrov . . . 20
7.4.1.1 Suroviny a vegetace . . . 21
7.4.2 Lesní ostrov . . . 22
7.4.2.1 Suroviny a vegetace . . . 22
7.5 Postavy . . . 23
7.5.1 Hlavní postava . . . 23
7.5.1.1 Mrtvý Jord . . . 24
7.5.2 Nehratelné postavy . . . 25
7.5.3 Zvířata . . . 25
7.5.4 Nepřátelé . . . 26
7.5.4.1 Útoky nepřátel . . . 26
7.6 Budovy . . . 27
7.6.1 Konstrukce . . . 27
7.6.2 Zvířecí budovy . . . 27
7.6.3 Farmářské budovy . . . 27
7.6.4 Ostatní . . . 28
7.6.4.1 Ohniště . . . 28
7.6.4.2 Truhly . . . 28
7.6.4.3 Koryta . . . 28

8 Testování hry **29**

9 Závěr **30**

1. Úvod

Vývoj her je složitý a zdlouhavý proces - zahrnuje totiž vymyšlení a zpracování nápadu, design, programování a v poslední fázi i marketing. Větší herní studia mají obvykle specialisty na každou část procesu, zatímco nezávislí vývojáři, známí také jako "Indie developeři"[1], často pracují v menších týmech, kde každý člen může zastávat několik rolí současně. To může znamenat více práce pro každého jednotlivce, ale zároveň poskytuje větší svobodu ve vymýšlení nových nápadů.

Osobně se již věnuji vývoji her tři roky a mám za sebou několik menších projektů, které vznikly v rámci *game jamů* [2]. Tyto zkušenosti mi poskytly pevné základy v používání herního enginu Unity a programovacího jazyka C#. Nicméně nastal čas posunout se dál a vytvořit něco ambicióznějšího.

Proto jsem se spojil svým dlouholetým kamarádem Alexem, se kterým mě spojuje herní vývoj, s nápadem na společný projekt. Na začátku jsme měli jednoduchou myšlenku - vytvořit hru simulující život na farmě. Nicméně, jako programátoři jsme si velmi rychle uvědomili, že k úspěšnému zpracování této myšlenky budeme potřebovat pomoci nejen s grafickou částí hry, ale i se zvukovou částí. A jelikož vývoj her není pro jednotlivce, vytvořili jsme menší tým, se kterým na hře pracujeme.

Cílem maturitního projektu bylo vytvoření hratelné verze počítačové hry simulující farmaření. Tímto cílem jsme se inspirovali, ale snažili jsme se vyhnout dalším kopiím existujících her a přinést do naší hry **vlastní originální** prvky. Teoretická část dokumentace se zaměřuje na rozdíly mezi naší hrou a hrami, ze kterých jsme si brali největší inspiraci. Praktická část dokumentace se zaměřuje na game design dokument, který obsahuje podrobnější grafický popis hry, popis jednotlivých herních prvků či klávesové ovládací prvky.

[1] Indie developer - neboli nezávislý vývojář, je jednotlivec nebo malý tým, který vytváří a vydává vlastní videohry bez finanční a tvůrčí podpory větší herní společnosti. [1]

[2] Game Jam je událost, kde týmy - nebo jednotlivci - vytvářejí hry během krátkého časového období s určeným tématem. [2]

2. Týmová práce

Na projektu pracuji od února 2023 společně s mým přítelem Alexem, protože jak se říká: *"Ve dvou se to lépe táhne"*.

Herní vývoj **není** úkol pro jednoho člověka. Pokud bychom se pokusili vytvořit hru sami - od grafiky přes programování až po hudbu - pravděpodobně bychom měli velmi malý pokrok za dlouhou dobu. Tudíž se k nám během vývoje připojili další členové týmu - ať už grafici, či programátoři nebo skladatel hudby - což nám výrazně pomohlo se posunout vpřed.

2.1 Má práce

Jelikož se jedná o týmový projekt, je důležité definovat, co přesně je **mou prací**. Ale místo toho, abych se zaměřil na to, co je mou prací, raději zmíním, co **není mou prací**, jelikož seznam těchto věcí bude kratší:

- Systém na inventář - upravoval jsem
- Systém na dialogy
- Systém na vaření - předělával jsem
- Základní kámen pro procedurální generování
- Optimalizace kódu

Na grafické stránce, kromě pár drobných úprav, jsem pracoval minimálně. Veškeré materiály pro postavy, budovy, zvířata či plodiny jsou prací grafické části týmu.

3. Využité technologie

3.1 C#

C# je moderní, objektově orientovaný a typově bezpečný programovací jazyk s kořeny v rodině jazyků C, do které také patří jazyky C, C++, Java či JavaScript. C# na rozdíl od C++ poskytuje automatické uvolňování paměti či disponuje jednodušší syntaxí. Dále poskytuje ochranu před null hodnotami, LINQ a asynchronní operace. [3]

3.1.1 Mono vs ILL2CPP

Unity Engine využívá dva skriptovací backendy – Mono a IL2CPP.

- Mono využívá kompilaci JIT (just-in-time) a kompiluje kód na vyžádání za běhu.
- IL2CPP využívá kompilaci před spuštěním (ahead-of-time, AOT) a celou aplikaci zkompiluje před jejím spuštěním.

Výhodou použití skriptovacího backendu založeného na JIT je, že doba kompilace je obvykle mnohem kratší než u AOT. [4]

3.2 Unity

Unity je herní engine, který stojí za úspěšnými herními tituly jako Cities: Skylines, Rust, Genshin Impact či Subnautica. Je využíván zejména Indie developery, ale v menší míře i profesionálními vývojáři. [5]
Jakožto herní engine usnadňuje vývojářům mnoho práce, ale - z mé vlastní zkušenosti - i mnoho práce ztěžuje. Nemusíme se například starat o simulaci fyziky, vykreslování grafiky či herní smyčku.

Co pak je tedy úkolem herního vývojáře?

Herní vývojáři mají za úkol přivést nápady či koncepty k životu. Tím je na mysli implementace herních mechanismů, reakcí na hráčovy akce a celková interakce s herním světem.
Našim úkolem je tedy nejen ”rozpohybovat”grafiku, ale skutečně přivést nápady k životu.

4. Využité nástroje

4.1 Visual Studio Code

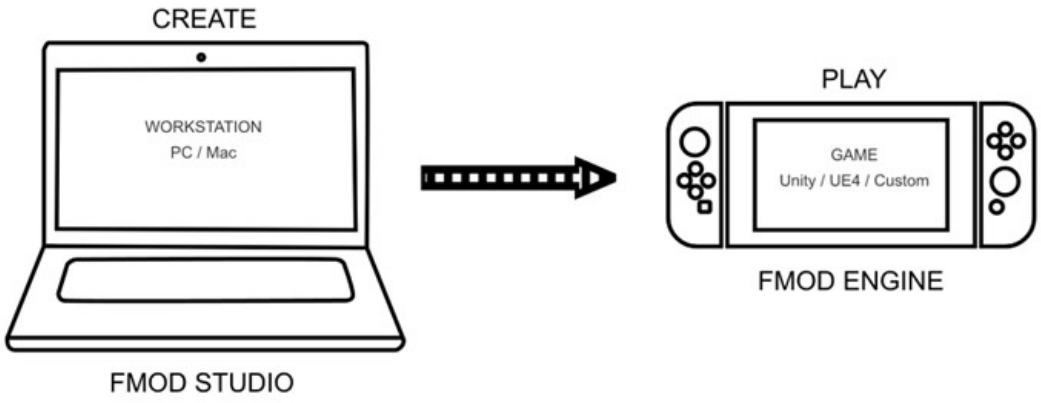
Visual Studio Code je open-source textový editor, vyvinutý společností Microsoft. Podporuje mnoho programovacích jazyků a poskytuje užitečné funkce jako syntax highlighting, inteligentní doplňování kódu, debugging, integraci s verzovacími systémy (Git) a další nástroje, které usnadňují vývoj softwaru. [6]

4.2 Git

Git je distribuovaný systém pro správu verzí, který je používáný pro sledování změn v kódu. [7] Umožňuje vývojářům spolupracovat na projektech a udržovat historii změn v kódu. Tento systém umožňuje ukládání dat v repozitářích a nabízí možnosti vytváření větví, sloučení změn a práci na různých částech kódu souběžně. [8]

4.3 FMOD studio

FMOD Studio je komplexní nástroj pro vytváření zvukového designu v herním prostředí. Poskytuje organizaci projektů, možnost sdílení předvoleb a umožňuje poslech změn v reálném čase, ať už přímo v editoru nebo ve hře. Pro sound designéry poskytuje nástroje pro tvorbu zvuků a hudby. Pro programátory nabízí usnadněnou integraci zvuku do her pomocí hotových integrací pro Unity a Unreal Engine, a dává možnost snadné integrace s vlastními herními enginy prostřednictvím API. Viz obrázek 4.3.1 [9]



Obrázek 4.3.1: Ukázka FMODu

5. Inspirace dalšími hrami

Naše hra čerpá inspiraci ze dvou herních titulů: Stardew Valley a Minecraft.
Stardew Valley nás inspirovala nejen obsahově, ale i graficky, jelikož se jedná o hru stejného žánru. Nicméně jsme nechtěli vytvořit jen další kopii Stardew Valleya, tudíž jsme přidali procedurální generování světa. Tímto se odlišujeme, neboť Stardew Valley nabízí statický svět, zatímco naši hráči zažívají nové dobrodružství s každým novým světem. Dalším rozdílem mezi naší hrou a Stardew Valley je v útočící zeleniny. Což je prvek, který Stardew Valley nenabízí.

Pokud jde o inspiraci z Minecraftu, zaměřili jsme se především na jeho procedurálně generované světy. Tuto funkci využíváme i v naší hře, akorát s menšími rozdíly. Naše světy nejsou tak velké jako v Minecraftu (naše mapa je velká *pouze* 200x200 bloků) a vytváříme **pouze jeden** ostrov.

5.1 Stardew Valley

Za celou hrou stojí pouze jeden člověk - Eric Barone. Hra původně vznikla jakožto klon hry Harvest Moon - Eric totiž nebyl spokojen s pozdějším vývojem hry a tak se rozhodl si vytvořit vlastní hru. [10]
Vývoj hry Stardew Valley započal v roce 2012 - v tu dobu bylo XNA ještě aktuální platformou pro vývoj her. Eric tudíž použil C# spolu s frameworkem XNA. Pro další aktualizace a porty (například pro iOS, Android či Switch) Eric použil MonoGame, což umožňuje vývojářům portovat hry na různé platformy a přitom zustává velmi blízké XNA frameworku. [11]

5.1.1 Microsoft XNA

XNA - ”Xbox New Architecture”- byla sada nástrojů vyvinutá společností Microsoft Gaming, založená na .NET frameworku. XNA bylo navrženo tak, aby podporovalo nezávislé vývojáře při tvorbě her pro platformy Windows a Xbox 360. Microsoft oficiálně ukončil vývoj XNA v roce 2013.
Vývojáři se přesunuli k open-source alternativě nazývané MonoGame, což je implementace frameworku Microsoft XNA 4. Díky MonoGame mohli vývojáři pokračovat ve vývoji svých existujících her a vytvářet nové hry. [12]

5.1.2 Monogame vs Unity

Monogame nabízí následující:

1. 2D a 3D vykreslování
2. Přehrávání zvukových efektů a hudby
3. Inputy z klávesnice, myši, dotyku a ovladače
4. Matematickou knihovnu optimalizovanou pro hry

Tohle vše ale nabízí i Unity, takže kde je tedy rozdíl?

Monogame **je framework**, který poskytuje základní stavební kameny pro stavbu vlastního herního enginu. Vývojář by neměl očekávat editor scény, který nabízí Unity či Unreal. Monogame sám o sobě nenabízí ani uživatelské rozhraní či engine na fyziku. Vývojáři si vše musí vytvořit sami, či se spolehnout na open-source řešení od jiných vývojářů. [13]

Unity tohle řeší za vývojáře - vývojář už dostane hotové řešení, se kterým poté pracuje. Vývojář ale nemá možnost měnit kód v pozadí, může pouze volat hotové věci.

Obojí má své výhody a nevýhody. U Monogame si vývojář staví vše sám, tudíž má větší kontrolu nad enginem, což na druhou stranu stojí čas. Ten čas, který by vývojář strávil nad vytvářením vlastního enginu, může investovat již do hotové věci - jako je například Unity. To ale za cenu toho, že nebude mít tak velkou kontrolu nad kódem a v případě výdělku se bude muset rozdělit s firmou, která stojí za hotovým řeším.

5.1.3 Důvod našeho rozhodnutí

Důvodem **našeho rozhodnutí** pro Unity je především hotové řešení. V naší hře pracujeme s tilemapami, skriptovacími objekty či animátorem, který se nám stará o animace. Využitím Unity nám odpadla nutnost vytvářet si vlastní systémy, jak by tomu bylo v případě použití Monogame. Tím jsme ušetřili čas, který jsme investovali do nových nápadů či optimalizace stávajícího kódu.

Dalším důležitým faktorem je **rozsáhlá komunita a dostupnost materiálů**. Unity je populárnější engine s velkou komunitou uživatelů, což znamená, že máme vyšší šance najít relevantní tutoriály, rychlejší řešení pro vzniklé problémy a snazší možnost navázání kontaktu s dalšími vývojáři.

5.2 Minecraft

Minecraft je sandboxová[1] hra vyvinutá společností Mojang Studios pro mobilní, počítačová či konzolová zařízení. Hráči zde prozkoumávají blokový, procedurálně generovaný svět, sbírají materiály, vyrábějí předměty a staví stavby, ať už sami či s přáteli.[15]

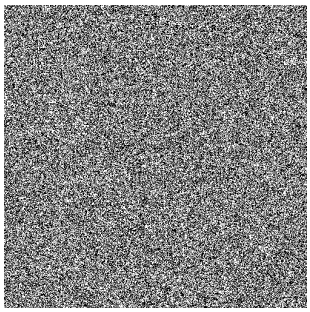
5.2.1 Generace světa

Minecraft využívá procedurálního generování, což je proces, který vytváří obsah pomocí algoritmů nežli ručně. Kromě her se využívá i k vytváření textur či 3D modelů. Procedurální generování totiž zajišťuje, že každý výtvor je **jedinečný a nepředvídatelný**. [16]

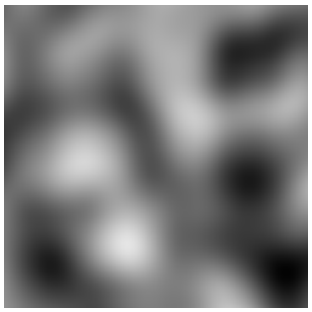
Aby se pokaždé vygeneroval jiný svět, hra používá náhodná vygenerovaná čísla z tzv. "seedu". Nicméně čistá náhodnost často způsobuje chaotické a nesouměrné terény, což může narušit herní zážitek. K vyřešení tohoto problému Minecraft využívá techniky zvané Perlin Noise[2], který zajišťuje kontinuitu. Ukázky generace šumu jsou vidět na obrázcích 5.2.1 - 5.2.3

I přesto, že šum vypadá náhodně a spojitě, vygenerovaný terén stále postrádá variace, jako jsou kopce či údolí. Z tohoto důvodu se generuje více funkcí šumu s různými frekvencemi a amplitudami, které se následně sečnou a dají přirozenější výsledek. Tyto šumové funkce se nazývají **oktávy**. [18]

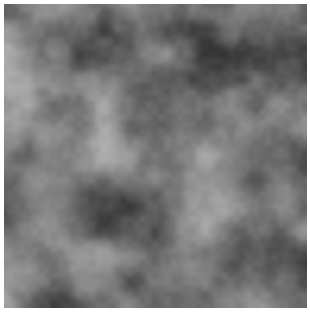
Celá generace světa probíhá ve 12 krocích, ale podrobněji se budu věnovat 3 krokům - biomu, šum a povrch - jelikož mají k naší hře nejblíže.



Obrázek 5.2.1: Bílý šum - čistá náhodnost



Obrázek 5.2.2: Přidání Perlin noisu



Obrázek 5.2.3: Perlin noise s více oktávami

[1] V sandboxové hře je hráč vybídnut k prozkoumávání prostředí a interakci s ním. Cílem sandboxové hry je, aby hráč poznal funkce, mechaniky a omezení hry prostřednictvím pokusů a omylů. U sandboxových her neexistuje konec či vítězství. [14]

[2] Perlin noise - vyvinutý Kenem Perlinem - je gradientní šum, který se využívá k vytvoření "hladké"náhodnosti.[17]

5.2.1.1 Biomy

I přesto, že v naší hře biomy nemáme, pro pozdější pochopení je dobré vědět, jakým způsobem se generují biomy v Minecraftu.

Generování biomů závisí na **šesti** parametrech: teplota, vlhkost, kontinentálnost, eroze, podivnost (weirdness) a hloubce.

Teplota a vlhkost, rozdělená každá do pěti úrovní, určují vlastnosti biomu, jako je vegetace a typ terénu.
Kontinentalita rozhoduje mezi oceánskými, plážovými a suchozemskými biomy na základě blízkosti vnitrozemí.
Eroze ovlivňuje rovinatost/hornatost terénu.
Podivnost ovlivňuje variace biomů a krajinné útvary, jako jsou vrcholy / údolí.
Hloubka odpovídá výšce terénu a ovlivňuje umístění povrchových / jeskynních biomů. [18]

5.2.1.2 Šum

Pro tvarování terénu se používá 3D Perlin noise, který určuje, zda mají být bloky plné nebo prázdné. Tato funkce šumu spolu s parametry, jako je výškové zkreslení a výška základny, umožňuje vytvářet přirozené terénní prvky, jako jsou například kopce či údolí.

Pro vytvoření dramatičtějšího terénu, jako jsou například útesy či náhorní plošiny, se využívá několik 2D šumových map mapovaných pomocí flexibilních křivek. *(Zde bych se rád odkázal zpět na Biomy, jelikož stejné šumy se využívají právě i při generaci biomů. Díky tomu vznikají měkké vazby mezi terénem a biomem)*. Tyto mapy ovlivňují jak výšku terénu, tak generování biomů a výsledkem je souvislá krajina, která odráží vybrané charakteristiky biomů. [18]

Naše hra využívá stejného, leč zjednodušeného procesu. Zjednodušeného proto, jelikož nemáme biomy. Vzhledem k tomu, že se jedná o 2D hru, Perlin noise nám vygeneruje terén, jehož hodnoty poté s hodnotami mezi 0 a 1, které představují základní vrstvu terénu, s níž poté pracujeme. Nicméně Perlin noise nám sám o sobě nevytvoří hlavní ostrov či řeky - pro dosažení těchto prvků jsou potřeba další postupy.

5.2.1.3 Povrch

Po vygenerování základního terénu se některé bloky nahradí jiným blokem v závislosti na biomu a dalších podmínkách. [18]

Jak již bylo zmíněno, Perlin noise nám vygeneroval terén s hodnotami mezi 0 a 1. Teď je pouze na nás, abychom k hodnotám přiřadili správný ”blok”, viz. tabulka 5.1. Každý ostrov má jiné bloky, nicméně hodnoty pro oceán a mělkou vodu jsou na všech ostrovech stejné.

Blok	Výška
Ocean	0
Shallow water	0.1 - 0.2
Beach	0.21 - 0.3
Grass	0.31 - 0.7
Flower grass	0.71 - 1

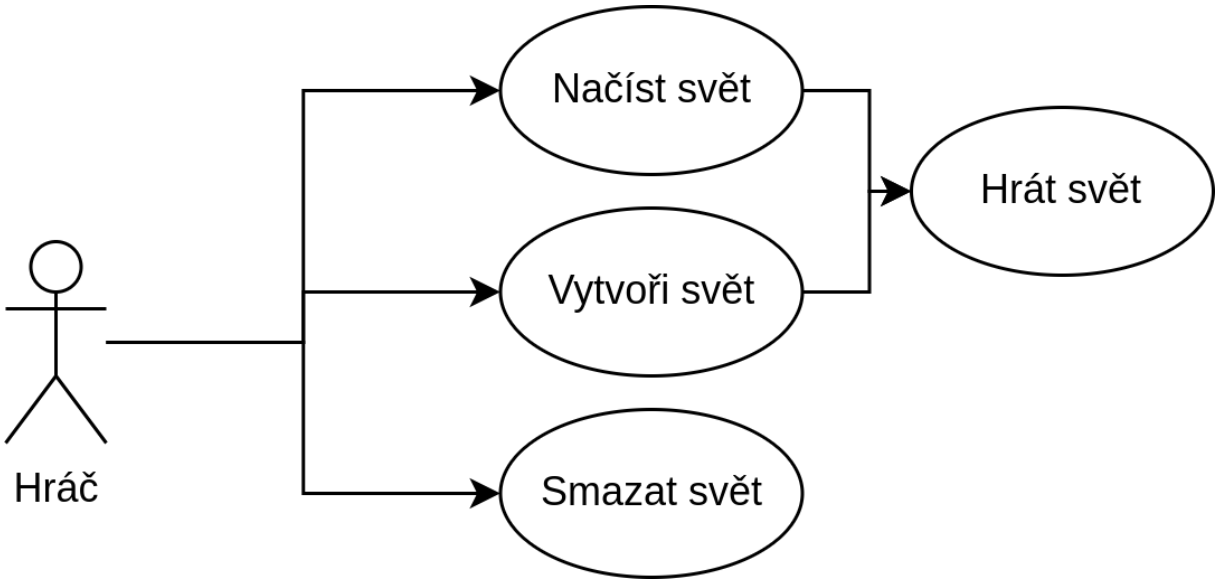
Tabulka 5.1: Tabulka ukazující blok a jeho přiřazenou výšku

Po vygenerování terénu se postaví přístav.
Jelikož víme, kde je střed mapy, náhodně si zvolíme nějakou stranu a snažím se na té straně přiblížit k moři, jelikož přístav se postaví pouze tehdy, když najdeme vodu.
V případě, že se přístav nepostaví, hráč je upozorněn na chybu a svět se nevytvoří.

6. UML

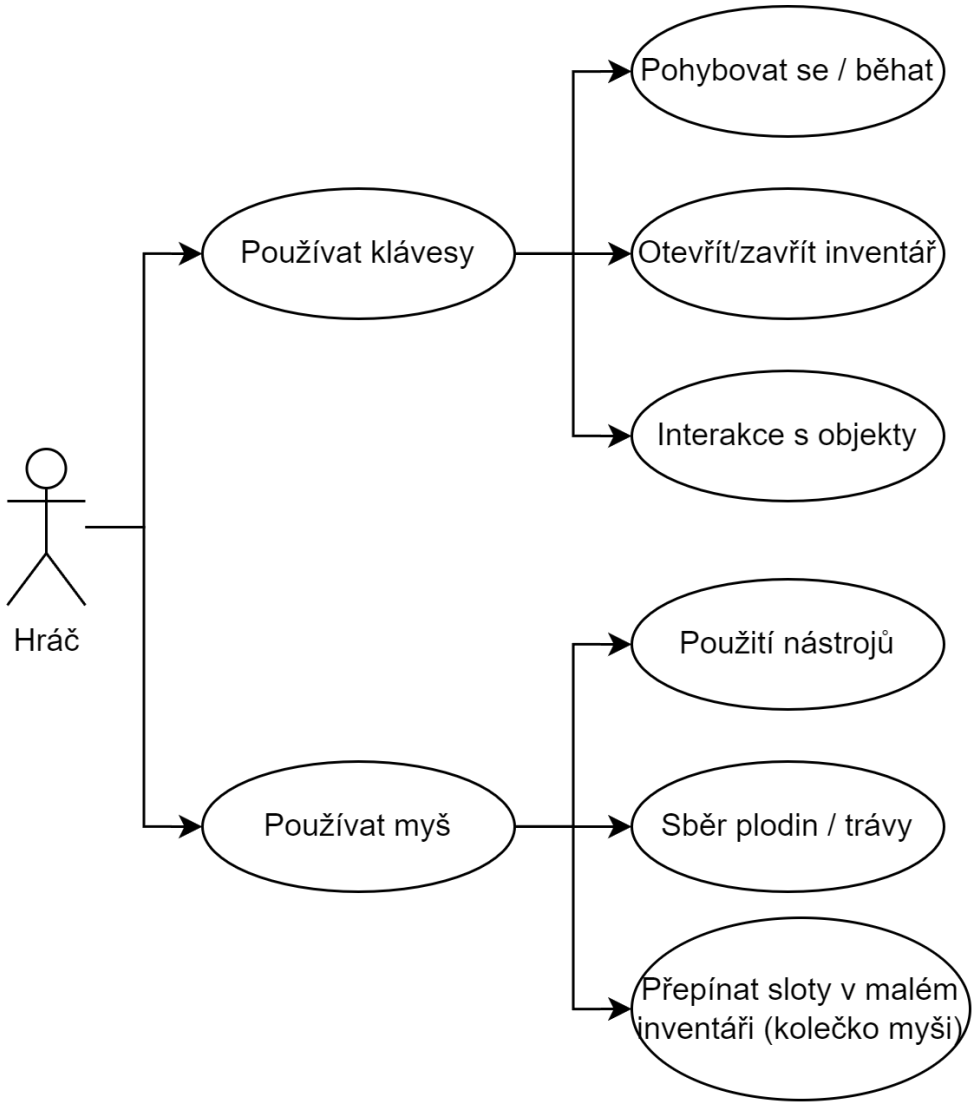
6.1 Usecase diagramy

Diagram zobrazuje hlavní menu hry se třemi možnostmi akcí: ”Načíst svět”, ”Vytvořit svět”a ”Smazat svět”. Ať už hráč zvolí načíst či vytvořit svět - v obou případech daný svět může hrát.



Obrázek 6.1.1: UML - Hlavní menu

Diagram zobrazuje možnosti interakce hráče se hrou prostřednictvím klávesnice a myši. Použití klávesnice umožňuje hráči pohyb a interakce s herním prostředím. Použití myši umožňuje hráči používat nástroje či sbírat plodiny. Kolečko myši umožňuje přepínat sloty v malém inventáři.



Obrázek 6.1.2: UML - Hráčovy akce ve světě

6.2 Class diagramy

Celý class diagram projektu lze vidět v příloze I. Třídy jsou znázorněny žlutými obdélníky, interfaci zelenými obdélníky a enumy modrými obdélníky. Některé názvy metod jsou psané kurzívou - tím je totiž značeno, že se jedná o abstraktní metodu.

Určité metody nebo vlastnosti mohou u sebe mít symbol hvězdičky. To naznačuje modifikátor přístupu "protected". Jestliže žádný symbol neexistuje, znamená to, že metoda či vlastnost jsou veřejně přístupné.

Na zbývajících class diagramech (přílohy II, III, IV) je patrný další pilíř objektově orientovaného programování (OOP) - dědičnost. V každém z těchto případů potomci tříd dědí vlastnosti a chování ze svých rodičovských tříd.

Zejména na diagramech pro nástroje (příloha II) a budovy (příloha IV) lze pozorovat další pilíře OOP - abstrakci a polymorfismus.

U itemů (příloha II) je vidět, že rodičovská třída dětí ze *ScriptableObject*.

Co je *ScriptableObject*?

ScriptableObject je datový kontejner, který lze použít k ukládání velkého množství dat nezávisle na instancích tříd. Jedním z hlavních důvodů použití *ScriptableObjects* je snížení spotřeby paměti projektu tím, že se vyhneme kopírování hodnot z objektu, který má **neměnná data** a jsou součástí tříd dědících z MonoBehaviour. [19]

Je tedy vhodné využívat *ScriptableObjects* pro položky, u kterých předpokládáme, že se jejich hodnota v průběhu hry nezmění. Zároveň poskytují i lepší organizaci projektu - můžeme si vytvořit libovolný počet položek a přesunout je do libovolné složky, přičemž se vždy soustředíme jen na data uvnitř - ukazování těchto dat poté řeší jiné skripty.

7. Game design dokument

7.1 Přehled hry

7.1.1 Koncept hry

Hráč vstoupí do 2D procedurálně generovaného světa, který se skládá ze dvou ostrovů. Na začátku hry hráč vstoupí na svůj domovský ostrov, kde je jeho hlavním úkolem najít a opravit přístav, který slouží jako brána do vnějšího světa. Po opravě přístavu již hráč může cestovat na další ostrov, kde může získat nový materiál. Zároveň ho bude každý den navštěvovat obchodník, se kterým hráč obchoduje či dostává úkoly (questy), za které pak dostává peněžní odměny. Hlavním úkolem ale nadále zůstává vybudování a rozšíření farmy na domovském ostrově.

7.1.2 Základní herní mechanismy

1. Simulace zemědělství: Sázení a sklízení plodin, chov zvířat
2. Stavění: Stavění a rozrůstání farmy a jejího okolí
3. Vaření: Připravování jídel a nápojů
4. Řemeslná výroba: Výroba nových nástrojů
5. Průzkum okolí: Objevování nových lokací a materiálů

7.1.3 Žánr

Hra spadá do *farm life simulation* žánru, s lehkým přesahem i do RPG žánru - a to zejména kvůli pruzkumné části hry.

7.1.3.1 Definice žánru

Farm life simulation je podžánr ”Life simulation”žánru. Žánr spojuje prvku sociální interakce a simulace života na farmě. V tomto žánru se hlavní hrdina obvykle vydává na venkov a zakládá farmu, často kvůli dědictví nebo z nudy ve městě. Hráč pěstuje plodiny a chová zvěř, aby vydělal peníze, a může komunikovat s širokou skupinou postav, které jsou obyvateli města, a pracovat na dosažení hlavního cíle zápletky, pokud nějaký existuje.[20]

7.1.4 Cílová skupina

Cílovou skupinou jsou hráči, kteří si oblíbili hry jako Stardew Valley či Minecraft. Tito hráči ocení pomalejší postup hry a vyhledávají atmosféru, kde nejsou do ničeho tlačeni a vše si mohou uspořádat dle sebe.

7.1.5 Atmosféra hry

Pro hru jsme zvolili klidnější a mírumilovnější atmosféru, která se točí především okolo přírody. Přeci jen největší množství času hráč stráví právě v přírodě, kde - **momentálně** - nemá žádné nepřátele, tudíž může stavět beze strachu o svůj život.

7.2 Gameplay a mechaniky

7.2.1 Herní progres

Hra je rozdělená na dvě fáze - ranou a prostřední. *(O pozdější fázi hry se budeme zabývat až v dalších letech vývoje.)*

V rané fázi se hráč seznamuje se světěm, snaží se najít nejlepší místo na postavení farmy, opatří si první plodinu a snaží se najít přístav, který se nachází někde na ostrově, a opravit ho.
Jeho úplně první plodinou (pokud pomineme bobule, které slouží hráči jakožto rychlý přísun jídla) je pšenice. Tu hráč může získat sběrem trávy - momentální šance na získání pšenice je 30%.

Hráč také musí dbát na to, aby byl schopen sesbírat potřebný materiál na dokončení přístavu, jelikož pouze tak může přejít do další fáze hry. Po dokončení přístavu hra přechází do prostřední fáze. Tím, že hráč dokončí přístav, získá nejen možnost vycestovat na další ostrov, ale také ho bude navštěvovat obchodník.

V prostřední fázi je hlavním úkolem hráče rozšíření farmy, ať už prostřednictvím nákupu nových plodin, zvěře či postavením nových budov. Vzhledem k tomu, že hra nemá konec a o další fáze hry se budeme zabývat až déle, hráč pokračuje ve hraní hry do doby, dokud ho hra neomrzí.

7.2.2 Cíle hry

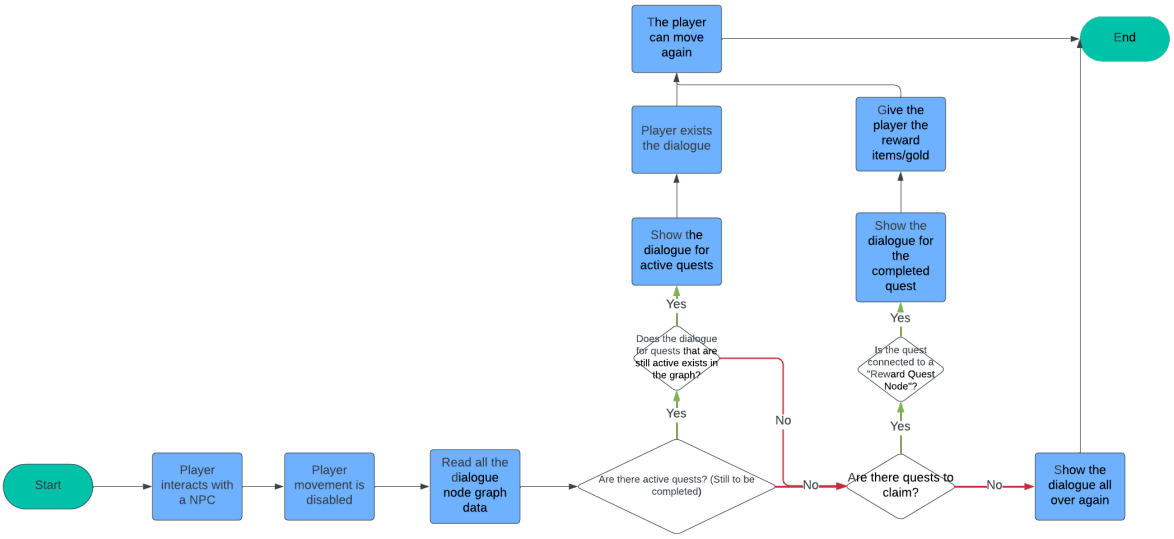
Prvním a hlavním cílem hry je postavit farmu. K tomu ale hráč potřebuje přístav, jelikož bez něj není schopen se posunout dál - bez přístavu se totiž neukáže obchodník, od které hráč může nakoupit nové plodiny či dokoupit suroviny. Tudíž druhořadým úkolem je najít a opravit přístav.

7.2.3 Struktura úkolů

Na úvod je důležité podotknout, že úkoly nebyly vypracovány mnou - celý systém byl navržen a implementován Alexem. Z tohoto důvodu je diagram v anglickém jazyce.

Pokud je přístav opraven, obchodník při každé návštěvě ostrova nabídne hráči nový úkol. Úkoly nejsou povinné a hráče nečeká postih za jejich nesplnění.

Následující diagram detailně popisuje fungování úkolů:



Obrázek 7.2.1: UML diagram ukazující jak fungují úkoly

7.2.4 Ukládání

Vzhledem k dynamické povaze hry má hráč možnost si uložit a načíst svůj herní pokrok. Data jsou ukládány **lokálně** a šifrovány, aby se k němu **běžný a programátorsky** nezkušený hráč nemohl tak lehce dostat.

Každý vytvořený herní svět má své vlastní složky (viz obrázek 7.2.2), které obsahují potřebná data pro ukládání a načítání hry.

```
Worlds/
└── -{World name}/
    ├── Islands/
    │   ├── HomeIsland/
    │   │   ├── AnimalData/
    │   │   │   ├── Animals
    │   │   │   └── FeedingInfo
    │   │   ├── TilesData
    │   │   ├── HarborData
    │   │   ├── Buildable
    │   │   └── Chests
    │   └── ForestIsland/
    │       ├── AnimalData {questionable}
    │       ├── TilesData
    │       └── HarborData
    ├── Player/
    │   ├── PlayerData
    │   ├── Inventories
    │   ├── Quests
    │   ├── UnlockedItems
    │   └── RainingData
    └── WorldSettings
```

Obrázek 7.2.2: Struktura ukládání dat

Veškerá data týkající se ostrovů se ukládají do složky *Islands* pod daný ostrov. Naopak veškerá data týkající se hráče, například jaké položky má hráč v inventáři, poslední poloha, čas či odemčené budovy / recepty, jsou uložena pod složkou *Player*. Pod složkou *WorldSettings* lze najít seed světa či pojmenování kočky.

7.2.5 Časové úseky

V průběhu hry dochází neustále k nějakým událostem. Veškeré hodinové akce jsou popsány v tabulce 7.1

Čas	Akce
00:00	Půlnoc - programově začátek nového dne
06:00	Hráč se probouzí
07:00	Zvířata jdou jíst
08:00	Zvířata dávají produkt
09:00	Hráč může cestovat na další ostrovy
13:00	Obchodník přichází
15:00	Obchodník odchází
17:00	Zvířata jdou jíst
18:00	Pokud hráč není na domovském ostrově, ukáže se varování, že má poslední hodinu
19:00	Hráč může jít spát, zvířata jdou spát

Tabulka 7.1: Tabulka ukazující jednotlivé časy a akce, které se v danou hodinu dějí

7.2.6 Úmrtí

Hráč může zemřít ze dvou důvodů. Buď mu dojde jídlo či voda, a tím postupně umírá, nebo je napaden nepřítelem. Jiná úmrtí nejsou momentálně možná.

V případě postupného úbytku životů způsobeného nedostatkem jídla či vody, stačí hráči doplnit potřebné živiny, životy se poté obnoví. Podobně je tomu při útoku nepřátel. Pokud hráč unikne z dosahu útočícího nepřítele a má dostatečné množství živin, jeho životy se postupně obnovují. Ať už z jakýkoliv příčin hráč umírá, nemůže cestovat a ani otevřít menu, které zastaví hru.

Po smrti hráč ztrácí všechny položky v inventáři a obdrží pouze klíč. Tento klíč může hráč použít k interakci (odemknutí) stromu, který se objevil na místě úmrtí. Po odemknutí stromu hráč dostane zpět své položky.

Nicméně, **ve světě může existovat pouze jeden strom**, a proto, pokud hráč zemře a strom je již někde jinde, strom se přesune na místo posledního úmrtí hráče a hráč může dostat pouze položky, které jeho inventář obsahoval při posledním úmrtí. Starší umístění stromu a jeho obsah jsou smazány.

7.3 Inputy

Inputy jsou základními akcemi, které hráč provádí v rámci hry pomocí klávesnice, myši nebo jiných ovládacích zařízení.

Jelikož využíváme balíčku ”New Input System”, hráči si mohou převázat [1]jednotlivé vstupy podle svých potřeb. Tato možnost je výhodná nejen pro leváky, ale pro všechny hráče, kteří chtějí přizpůsobit ovládání svým vlastním potřebám. Klávesy, které využíváme, jsou názorně vidět na obrázku 7.3.1



Obrázek 7.3.1: Využité klávesy - rozvržení pro praváky

7.3.1 Pohyb a myš

V naší hře jsme zvolili standardní klávesový input pro pohyb postavy a to pomocí kláves WASD. Tento způsob ovládání je běžně používán ve všech hrách s top-down pohledem nebo ve 3D prostředí. Co se myši týče, využíváme všech tří tlačítek. Pravé a levé tlačítko myši slouží k použití nástrojů či sběru trávy. Kolečko myši pak slouží k pohybu mezi jednotlivými sloty v malém inventáři.

7.3.2 Ostatní inputy

V tabulce 7.2 jsou vidět zbývající inputy a jejich příslušné akce.

Klávesa	Akce
ESC	Zavírání UI, otevírání ”pause menu”
TAB	Otevírá velký inventář
1-0	Specifický slot v malém inventáři
SHIFT	Běh
E	Interakce s objekty
Q	Otevírá questy
R	Otáčení konstrukcí
F11	Debug mód

Tabulka 7.2: Tabulka s ostatními inputy

[1]V anglickém jazyce je myšleno binding, nebo-li key-binding

7.4 Oblasti

Hra nabízí 2 oblasti. První z nich slouží k budování a rozšiřování vlastní farmy, zatímco druhá oblast slouží k průzkumu a získávání nových materiálů, které hráč může využít k vylepšení své farmy nebo k výrobě různých předmětů.

V budoucnu plánujeme rozšíření hry o další ostrovy, přičemž každý z těchto ostrovů bude nabízet unikátní prostředí, lokace, nehratelné postavy a materiály.

7.4.1 Domovský ostrov

Domovský ostrov představuje základní ostrov, na kterém si hráč staví svoji farmu.

Jednou z možností, kterou hráči na domovském ostrově mají, je stavba mostů. Tato funkce však přinesla menší problém, jelikož hráč by díky tomu mohl uniknout z mapy. Řešení tohoto problému spočívalo v tom, že ostrov je obklopen hlubokou vodou (viz obrázek 7.4.1 ukazující generaci ostrova, na kterém tmavě modrá barva značí hlubokou vodu), přes kterou hráč nemá možnost postavit most. Tímto způsobem jsme zamezili hráčům v úniku z domovského ostrova, přičemž jsme zachovali jejich volnost a možnost stavby na ostatních částech ostrova.



Obrázek 7.4.1: Ukázka generace domácího ostrova

7.4.1.1 Suroviny a vegetace

Přirozeně se ve hře vyskytují tři typy stromů - bříza, dub a vrba (viz obrázek 7.4.2). Po jejich pokácení hráč obdrží dřevo, klacky a případně i sazenici daného stromu, pokud má štěstí.

Dřevo představuje základní surovinu ve hře, která je nezbytná nejen pro opravu přístavu, ale také pro konstrukci dalších budov. Mimo to má dřevo využití i při výrobě různých předmětů a při zapalování ohniště.

Hráč má také možnost zakoupit další typy stromů, mezi které patří například třešeň či hrušeň (obrázek 7.4.3).



Obrázek 7.4.2: Design přirozeně generovaných stromů



Obrázek 7.4.3: Design ovocných stromů

Další surovinou je kámen, který se především využívá k výrobě nástrojů a v menší míře jako stavební materiál pro dokončení budov.

Poslední přirozeně se generující vegetací jsou ovocné keře - jahody a borůvky. Ty slouží zejména jako raný zdroj jídla. Pokud hráč sklidí pouze bobule, po nějakém čase zase narostou. Samozřejmě existuje i možnost keř vykopat a přesadit na jiné místo, ale je nutné počítat s tím, že keř potřebuje nějaký čas, než znovu vyroste.

7.4.2 Lesní ostrov

Na lesním ostrově není hráči dovoleno stavět. Jeho jedinými aktivitami jsou kácení stromů a sběr malin, které se každý týden obnovují. Ukázka generace ostrova je vidět na obrázku 7.4.4



Obrázek 7.4.4: Ukázka generace lesního ostrova

7.4.2.1 Suroviny a vegetace

Na lesním ostrově se vyskytují dva typy jehličnatých stromů. Po jejich pokácení stromů hráč obdrží dřevo, klacky a případně i sazenici daného stromu, pokud má štěstí. Kromě stromů se na ostrově vyskytují houby - které lze využít k výrobě jídla - nebo keře s malinami. Malinový keř hráč může přenést zpět na domovský ostrov a pěstovat je i tam.

Ukázka vegetace je vidět na obrázku 7.4.5 Oproti domovskému ostrovu je paleta pro les výrazně tmavší. Důvodem je navodění atmosféry.



Obrázek 7.4.5: Ukázka lesní vegetace

7.5 Postavy

7.5.1 Hlavní postava

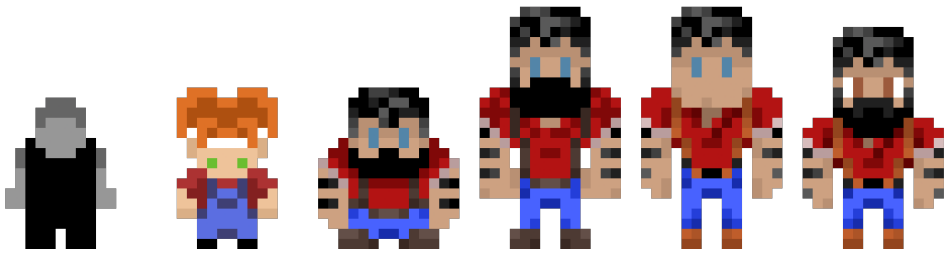
Design hlavní postavy hraje velmi důležitou roli. Postava **není pouze "estetickým prvkem"**, slouží i jako prostředník mezi hráčem a hrou, skrze kterého hráč prochází příběhem a interaguje s prostředím. Proto je důležité, aby hlavní postava **vzbuzovala u hráče příjemné pocity** a aby si s ní hráč dokázal vybudovat hlubší citový vztah.

Jord, celým svým jménem Jord Brukare, je původem ze Švédska. I přesto, že je pro švédskou populaci typická blond barva a modré oči, šli jsme opačným směrem - do tmavších barev (viz obrázek 7.5.1 ukazující různé varianty barvy vlasů a očí). Z estetického hlediska tmavší barvy působí méně rušivě a vzhledem ke světlému pozadí ostrova Jord **- díky tmavším barvám - lépe vynikne**.

Celý vývoj hráče lze vidět na obrázku 7.5.2



Obrázek 7.5.1: Design tváře



Obrázek 7.5.2: Design hráče v průběhu vývoje - vlevo úplně první prototyp, vpravo finální verze

7.5.1.1 Mrtvý Jord

Po hráčově úmrtí je třeba ukázat místo, na kterém hráč zemřel. Tudíž jsme potřebovali návrh na ukázku hráčova úmrtí. První návrh byl vcelku prostý - vytvoříme pouze jeho lebku (obrázek 7.5.3) a na místě úmrtí necháme hrob. Tento nápad nám přišel vcelku nudný a nezajímavý. Přišli jsme tudíž s lepším plánem.



Obrázek 7.5.3: Ukázka lebek

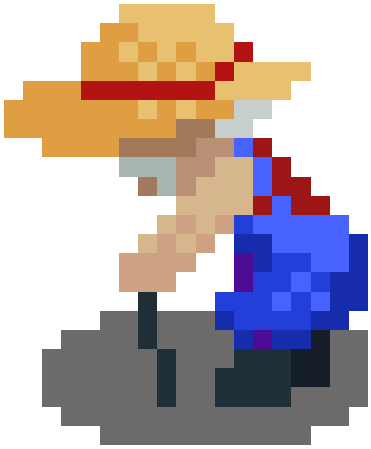
Pro Jorda jsme vytvořili strom, který vypadá jako on (obrázek 7.5.4), který se objeví na místě úmrtí. Přišlo nám to jakožto lepší a hlavně **zajímavější** alternativa.



Obrázek 7.5.4: Ukázka mrtvého Jorda

7.5.2 Nehratelné postavy

Jedinou nehratelnou postavou je obchodník, známý pod jménem Roger L. (obrázek 7.5.5). Jedná se o postaršího muže, který vlastní loď One Piece (obrázek 7.5.6), s níž každý den navštěvuje hráče. Nejenže obchodník nabízí nové zboží či zvířata k prodeji, ale také nabízí možnost odkoupit zboží od hráčů.



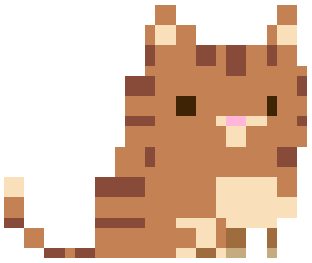
Obrázek 7.5.5: Obchodník



Obrázek 7.5.6: Obchodníkova loď

7.5.3 Zvířata

Jelikož se jedná o farmářskou hru, nesmí chybět zvířata. Prvním zvířetem, tedy spíš mazlíčkem, je kočka (obrázek 7.5.7). Hráč má možnost si ji pojmenovat dle vlastního uvážení. Kočka sice nemá žádný velký význam ve hře, slouží spíše jako společník pro hráče, aby se v herním světě necítili osaměle.



Obrázek 7.5.7: Kočka

Další zvířata jsou čistě hospodářská. Každé zvíře má dvě stádia - buď se jedná o mládě nebo o dospělého jedince. Rozdíl je v tom, že mláďata jsou sice levnější na koupi, ale nedávají produkty. Dospělý jedinec již dává produkty - například vajíčka, mléko - které pak hráč může dále zpracovávat.

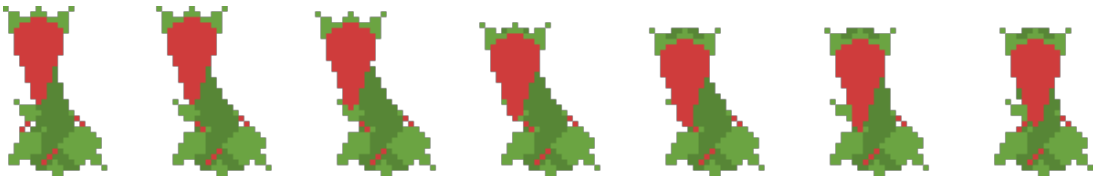
Jak mládě, tak i dospělý jedinec potřebují každý den jíst. V případě, že nedostanou najíst 3 dny, zemřou. Hráč má možnost si vychovat i svá vlastní mláďata, stačí k tomu mít jejich oblíbenou plodinu zvířete a dostatek místa na ubytování.

7.5.4 Nepřátelé

Plodiny jsou jedinými nepřáteli hráče, přičemž momentálně je dostupná pouze brambora (obrázek 7.5.8). Grafická část týmu pracuje na implementaci papriky (obrázek 7.5.9) a obilí (obrázek 7.5.10). Pro ostatní plodiny se design a útoky teprve vyvíjejí. Cílem je zajistit, aby každá plodina měla unikátní design a útoky.



Obrázek 7.5.8: Nepřítel brambora



Obrázek 7.5.9: Nepřítel paprika



Obrázek 7.5.10: Nepřítel pšenice

7.5.4.1 Útoky nepřátel

Brambora má pouze jeden útok, který se aktivuje pouze v případě, když je hráč v jejím dosahu. Jejím Útokem je švihnutí stonku. V případě, že je hráč zasažen, ztrácí životy.
Paprika má ”kousnutí”. To se opět aktivuje pouze v případě, že je hráč v dosahu útoku.

Hráč není schopen zaútočit zpět. Jediné, co může udělat je, otočit se a odejít, jelikož rostliny po čase zmizí.

7.6 Budovy

Nedílnou součástí hry jsou i budovy. Budovy mají dvě části - buď to jsou nedokončené konstrukce nebo postavené budovy, se kterými lze interagovat.

7.6.1 Konstrukce

Po odemčení *blueprintu* [2] budovy, hráč nejdříve musí položit její konstrukci a dodat potřebný materiál. Až po dodání potřebného materiálu konstrukce může být dokončena a dále využita.

7.6.2 Zvířecí budovy

Dalším typem budov jsou zvířecí budovy - ty slouží pouze jako domov pro zvířata. Každý budova je schopna pojmout jiné množství zvířat, některé pojmou více, některé méně. V případě, že hráč nemá dostatek místa v budovách, nemůže si koupit další zvířata.
Dále budovy slouží k tomu, aby si hráč mohl vyzvednout produkty, které zvířata každý den dávají.

7.6.3 Farmářské budovy

I přesto, že máme design pro 5 budov (další budovu lze vidět na obrázku 7.6.2), je funkční pouze jedna budova - tou je **farmářský domek.** (obrázek 7.6.1) V případě, že je noc a hráč se nachází v blízkosti domku, po stisknutí klávesy hráč bude spát až do rána.



Obrázek 7.6.1: Farmářský domek



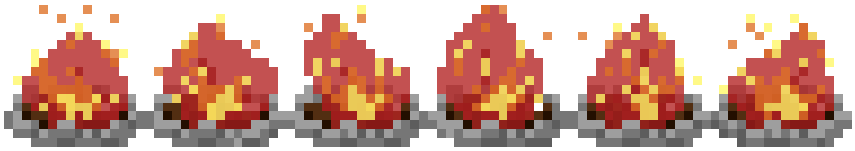
Obrázek 7.6.2: Ukázka další farmářské budovy - stodoly

[2] Díky blueprintu se hráč dozví, jaké materiály bude na stavbu potřebovat.

7.6.4 Ostatní

7.6.4.1 Ohniště

Hráč si může vytvořit a položit ohniště (obrázek 7.6.3). V případě, že je noc a hráč vloží dřevo, po stisknutí klávesy ”E”bude spát tak dlouho, kolik dřeva do ohniště vložil.



Obrázek 7.6.3: Animované ohniště

7.6.4.2 Truhly

Hráč má základně 40 slotů - 10 slotů v menším inventáři a 30 slotů ve větším inventáři. Což ale nemusí být dostačující - proto jsme do hry přidali truhly.
Nabízíme 3 typy truhel - liší se počtem dostupných slotů *(úložným prostorem)* a materiálem potřebným k výrobě.



Obrázek 7.6.4: Truhly

Nejdostupnější truhlou je **truhla z pšenice**, zároveň ale nabízí nejmenším úložný prostor. Dalším typem jsou **dřevěněné truhly** (obrázek 7.6.4). Ty jsou taková střední cesta. Posledním typem je **kovová truhla**. Nejtěžší na výrobu, ale nabízí nejvíce úložného prostoru.

7.6.4.3 Koryta

Nedílnou součástí hry jsou koryta (obrázek 7.6.5), která jsou potřeba ke krmení zvířat. Máme dva typy krmení - pro slepice a pro ostatní zvířata. Hráč si tedy musí dávat pozor, které krmení vyrábí.



Obrázek 7.6.5: Koryta

8. Testování hry

V lednu 2024 jsme spustili beta verzi naší hry, kterou jsme nabídli naším přátelům na vyzkoušení. I přes pečlivou přípravu se již během několika dnů ukázalo, že hra hra obsahuje chyby, které jsme přehlédli.
Nicméně nám naši přátelé (a později i veřejnost) **poskytli cennou zpětnou vazbu** ohledně toho, jak se jim hra dobře hraje, jaké chyby v ní našly a především postup, jakým způsobem tu danou chybu našli.

Jedním z hlavních problémů bylo opakované duplikování předmětů v inventáři. Tento problém se objevil, když hráč přesunul nástroj - například sekyru - do jiného slotu v inventáři. Dalším problémem bylo nesprávné zacházení s přidáváním předmětů do inventáře, v případě plného hotbaru. V takovém případě se předměty nevkládaly do většího inventáře a zmizely.

Ačkoliv jsme během testování narazili na více problémů/chyb, všechny jsme je v co nejkratší době opravili. Kromě nalezených chyb nás také zajímalo, jak hráči vnímají celkovou hratelnost hry.

Odezva hráčů ukázala, že mnozí z nich nevěděli, že mohou sbírat předměty ze země a byli nespokojeni, že nevidí dobu růstu plodin. Proto jsme implementovali vizuální zobrazení, které hráčům ukazuje, zda mohou předmět sebrat a také zobrazení zbývajícího času do úplného růstu plodiny.

Z tohoto důvodu je testování hry nejdůležitějším procesem ve hrách. Díky testování jsme identifikovali nové chyby či nedostatky a získali jsme užitečnou zpětnou vazbu od našich hráčů. Díky tomu jsme mohli zlepšit uživatelský zážitek pro stávající i budoucí hráče.

9. Závěr

Cílem projektu bylo vytvořit hratelnou verzi naší hry, což se nám povedlo. Nicméně, jedním z nedostatků o kterých víme, je nefunkční systém úkolů. Ten sice umožňuje zadávání úkolů, ale hráč nemá možnost si odměnu vyzvednout.

Díky projektu jsem se naučil lepší spolupráci v týmu, zejména v oblastech komunikace a správa či rozvržení úkolů. Dále jsem získal hlubší porozumění grafických a hudebních prvků, které jsou stejně klíčové jako samotné programování.

Jsme spokojeni s tím, čeho jsme dosáhli, ale zároveň si uvědomujeme, že stále máme prostor ke zlepšení a zdokonalení naší hry. Náš cíl zde ale nekončí - rádi bychom hru nadále rozvíjeli a uvedli na platformu Steam. Ale k tomu potřebujeme ještě pár let vývoje.

Seznam bibliografických citací

[1] Aron Marriott-Smith. *What is an indie game developer?* 13. břez. 2024. URL: https://gamedevelopermarketing.com/what-is-an-indie-game-developer/.

[2] Aron Marriott-Smith. *What is game jam?* 13. břez. 2024. URL: https://gamedevelopermarketi com/what-is-a-game-jam/.

[3] Microsoft. *Tour of csharp*. 14. led. 2024. URL: https://learn.microsoft.com/cs-cz/dotnet/csharp/tour-of-csharp/.

[4] Unity Technologies. *Unity Scripting Backends*. 14. led. 2024. URL: https://docs.unity3d.com/2022.1/Documentation/Manual/scripting-backends.html.

[5] Unity Technologies. *Unity Games*. 14. led. 2024. URL: https://unity.com/games.

[6] Microsoft. *Why vscode*. 14. led. 2024. URL: https://code.visualstudio.com/docs/editor/whyvscode.

[7] Git Development Community. *Git*. 14. led. 2024. URL: https://git-scm.com/.

[8] Git Development Community. *About Git*. 14. led. 2024. URL: https://git-scm.com/about.

[9] FMOD. *FMOD Studio*. 14. led. 2024. URL: https://fmod.com/studio.

[10] That Guy Glen. *How Stardew Valley Was Made by Only One Person*. 19. ún. 2024. URL: https://thatguyglen.fun/read/how-stardew-valley-was-made-by-only-one-person.

[11] Dragonfly Data Organization. *What game engine does Stardew Valley use?* 1. ún. 2024. URL: https://www.dragonflydb.io/faq/what-game-engine-stardew-valley-use.

[12] Wikipedia. *Microsoft XNA*. 21. ún. 2024. URL: https://en.wikipedia.org/wiki/Microsoft_XNA.

[13] Monogame. *What is monogame?* 21. ún. 2024. URL: https://monogame.net/articles/index.html.

[14] Margaret Rouse. *What is a Sandbox Game?* 13. břez. 2024. URL: https://www.techopedia.com/definition/3952/sandbox-gaming.

[15] Per Ladin. *What is Minecraft?* 4. břez. 2024. URL: https://www.minecraft.net/en-us/article/what-minecraft.

[16] Zenva. *What is Procedural Generation*. 7. břez. 2024. URL: https://gamedevacademy.org/what-is-procedural-generation/.

[17] Sebastian Hirnschall. *Perlin noise: What is it, how to use it, and why it's better than value noise*. 7. břez. 2024. URL: https://blog.hirnschall.net/perlin-noise/.

[18] Minecraft Wiki. *World generation*. 4. břez. 2024. URL: https://minecraft.wiki/w/World_generation.

[19] Unity Technologies. *Unity ScriptableObject Documentation*. 13. břez. 2024. URL: https://docs.unity3d.com/Manual/class-ScriptableObject.html.

[20] Wikipedia. *Farm life simulation*. 15. led. 2024. URL: https://en.wikipedia.org/wiki/Farm_life_sim.

Seznam obrázků

4.3.1 Ukázka FMODu 4
5.2.1 Bílý šum - čistá náhodnost 7
5.2.2 Přidání Perlin noisu 7
5.2.3 Perlin noise s více oktávami 7
7.2.2 Struktura ukládání dat 17
7.3.1 Využité klávesy - rozvržení pro praváky 19
7.4.1 Ukázka generace domácího ostrova 20
7.4.2 Design přirozeně generovaných stromů 21
7.4.3 Design ovocných stromů 21
7.4.4 Ukázka generace lesního ostrova 22
7.4.5 Ukázka lesní vegetace 22
7.5.1 Design tváře 23
7.5.2 Design hráče v průběhu vývoje - vlevo úplně první prototyp, vpravo finální verze 23
7.5.3 Ukázka lebek 24
7.5.4 Ukázka mrtvého Jorda 24
7.5.5 Obchodník 25
7.5.6 Obchodníkova loď 25
7.5.7 Kočka 25
7.5.8 Nepřítel brambora 26
7.5.9 Nepřítel paprika 26
7.5.10 Nepřítel pšenice 26
7.6.1 Farmářský domek 27
7.6.2 Ukázka další farmářské budovy - stodoly 27
7.6.3 Animované ohniště 28
7.6.4 Truhly 28
7.6.5 Koryta 28

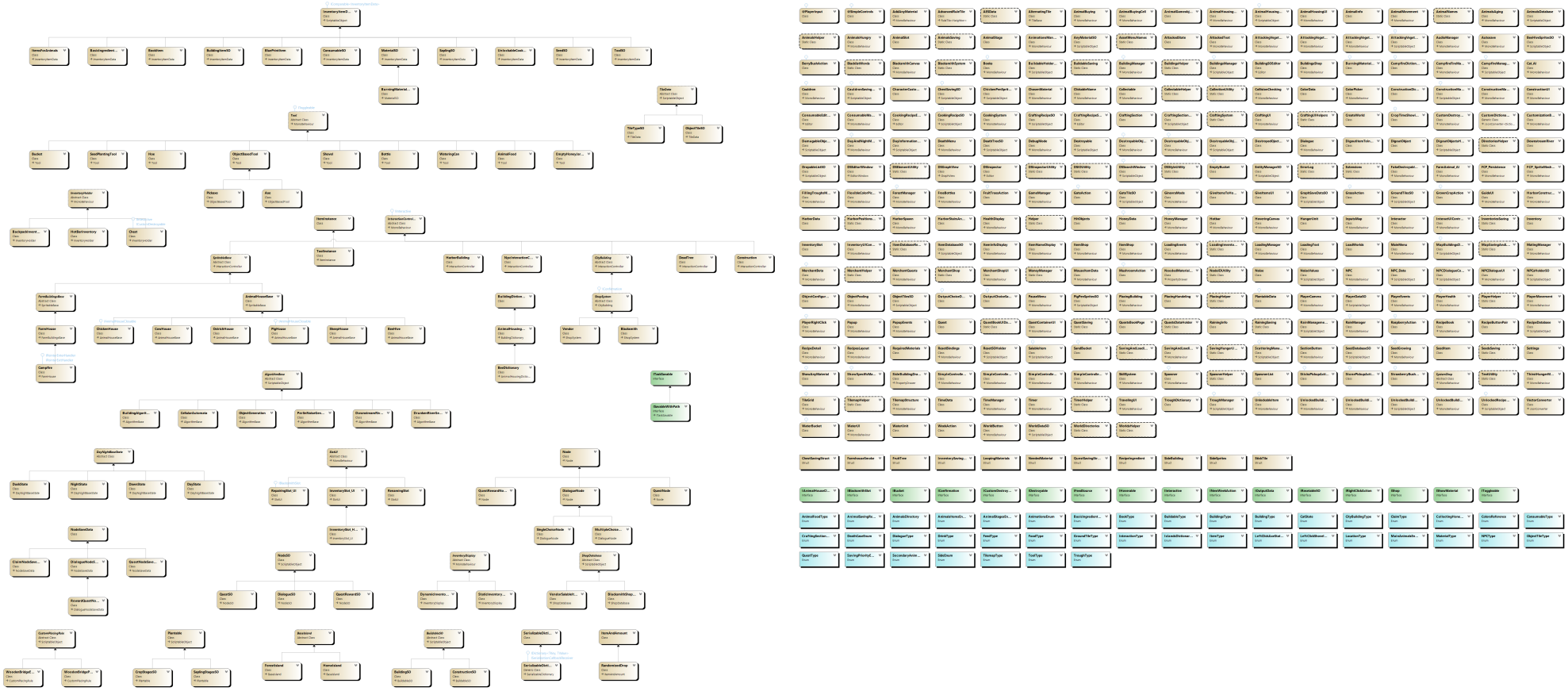
Seznam příloh

I Celkový class diagramu projektu . . . . . .
II Class diagram itemů . . . . . .
III Class diagram nástrojů . . . . . .
IV Class diagram budov . . . . . .

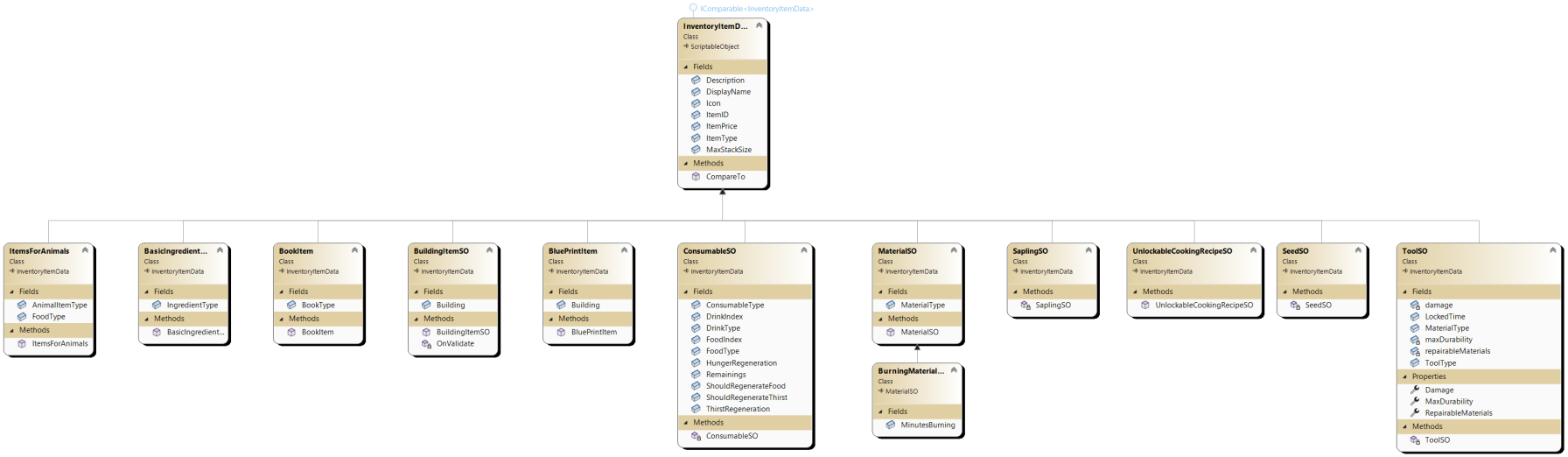
Seznam tabulek

5.1 Tabulka ukazující blok a jeho přiřazenou výšku 9

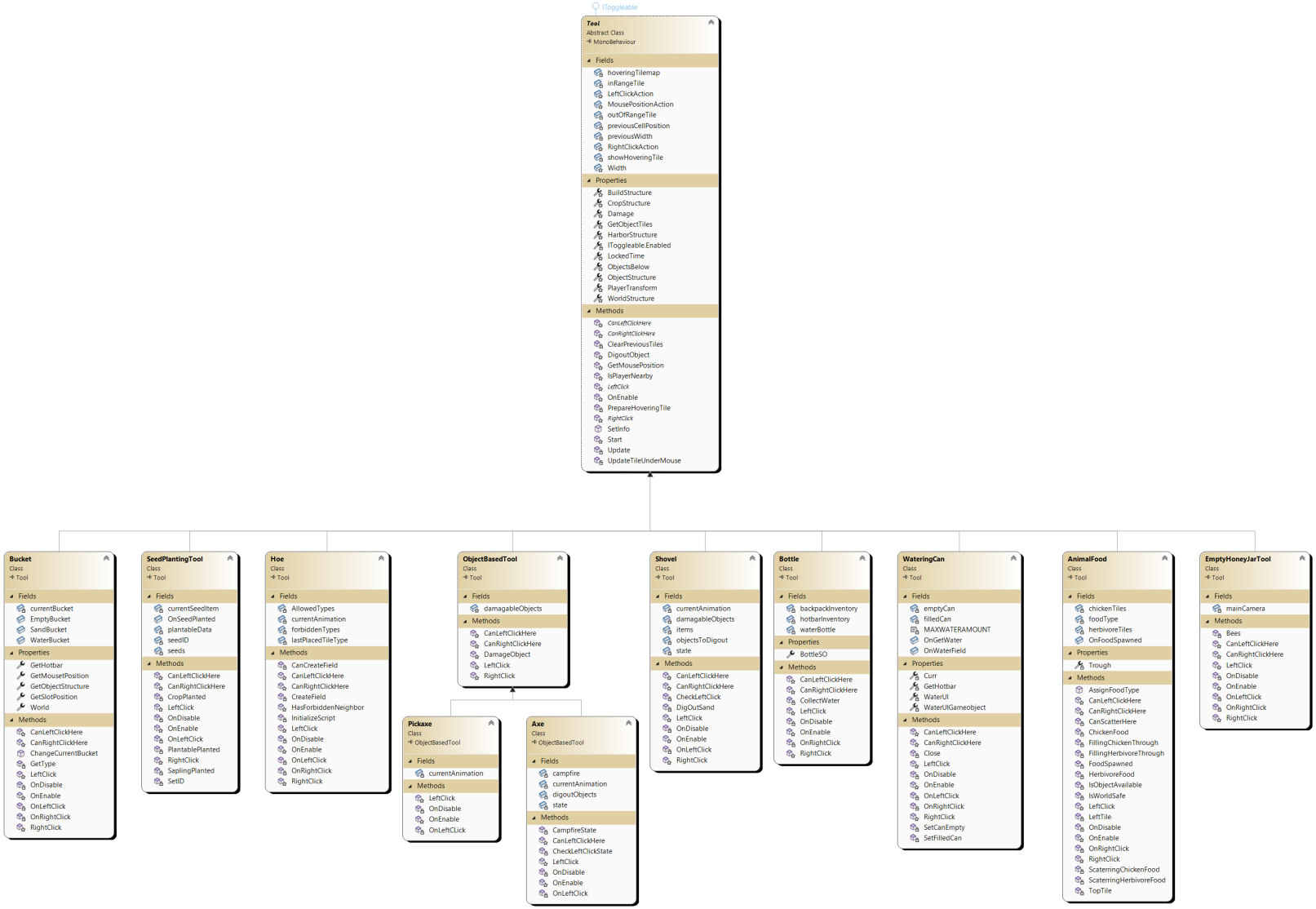
7.1 Tabulka ukazující jednotlivé časy a akce, které se v danou hodinu dějí 18
7.2 Tabulka s ostatními inputy 19



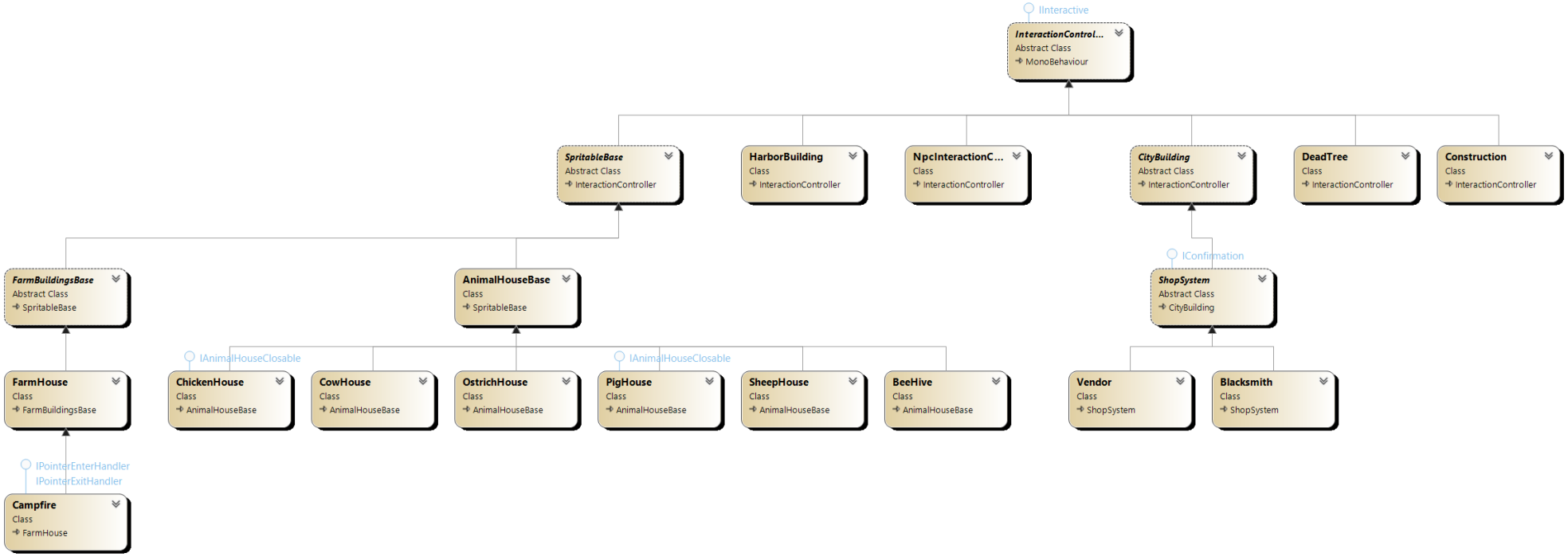
Příloha I: Celkový class diagramu projektu



Příloha II: Class diagram itemů



Příloha III: Class diagram nástrojů



Příloha IV: Class diagram budov