Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Management hra v Unity enginu

2023

Karolína Tobiášová

Vedoucí práce:
Mgr. Petr Osička, Ph.D.

Studijní program: Informatika,
Specializace: Programování a vývoj software

Bibliografické údaje

Autor:	Karolína Tobiášová
Název práce:	Management hra v Unity enginu
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2023
Studijní program:	Informatika, Specializace: Programování a vývoj software
Vedoucí práce:	Mgr. Petr Osička, Ph.D.
Počet stran:	36
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Karolína Tobiášová
Title:	Management game in Unity Engine
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2023
Study program:	Computer Science, Specialization: Programming and Software Development
Supervisor:	Mgr. Petr Osička, Ph.D.
Page count:	36
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Práce popisuje vývoj management hry s tematikou zahradnické firmy ve fiktivním magickém světě. Pro tvorbu hry byl použit herní engine Unity. Implementace je v programovacím jazyce C#. Grafické prvky hry byly vytvořeny v editoru Pixelart. Zvukové efekty byly vygenerovány v editoru SFXR. Hudba je nahraná a upravena v programu Audacity. V práci se zabývám návrhem a vývojem hry. Práce je zakončena uživatelskou příručkou.

Synopsis

This thesis describes the development of a management game with the theme of a gardening company in a fictional magical world. The Unity game engine was used for the development of this game. The implementation is in C#. The graphical elements of the game were created in Pixelart. The sound effects were generated in SFXR. The music was recorded and edited in Audacity. In this thesis I am involved in the design and development of the game. The thesis concludes with a user manual.

Klíčová slova: management hra; Unity; C#; tvorba hry

Keywords: management game; Unity; C#; game development

Poděkování patří Mgr. Petru Osičkovi, Ph.D za trpělivé vedení mé bakalářské práce.

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

Obsah

Seznam obrázků

1 Úvod

Management hra je typ videohry, ve které hráč řídí a spravuje určitý prvek, jako například podnik, město, zábavní park, hotel, nebo jinou organizaci. Cílem hry je například maximalizovat zisk, dosáhnout určité úrovně rozvoje nebo uspokojovat potřeby a přání obyvatel města.

Existuje mnoho populárních management her, z nichž některé jsou už mnoho let velmi úspěšné. Níže uvádím několik příkladů:

- SimCity [1]
- Two Point Hospital [2]
- Rimworld [3]
- Prison Architect [4]
- RollerCoaster Tycoon [5]

Cílem mé práce bylo vytvoření hry výše zmíněného žánru Management. Podtéma hry jsem si zvolila zahradnickou firmu. Hráčovým úkolem je se o firmu starat a pomoc jí prosperovat. Hráč vytváří různé úkoly pro pracovníky, což může být například orání půdy, sazení rostlin, či prodávání plodin zákazníkovi. Na konci dne hráč zaplatí poplatky za správu budov a dá plat svým pracovníkům. Každým dalším dnem hráč rozšiřuje a zlepšuje svoji farmu, dokud se nestane nejlepším farmářem široko daleko.

Pro tvorbu hry jsem používala multiplatformní herní engine Unity [6], který podporuje vývoj 2D i 3D her libovolného žánru. Také podporuje tvorbu skriptů v objektově orientovaném jazyce C# [7]. Při programování skriptů jsem využívala vývojové prostředí Visual Studio [8].

Jelikož jsem si grafickou stránku hry chtěla tvořit sama, zvolila jsem si tvorbu 2D hry. Grafické prvky jsem tvořila v online editoru Pixelart [9]. Ten je přehledný, jednoduchý na ovládání a je bezplatný.

Pro tvorbu zvuků ve hře jsem zvolila editor SFXR [10], který umožňuje vytvářet 8bitové zvuky. Složitější zvuky hry jsem získala ze stránky freesound.org [11]. Na tvorbu hudby v pozadí jsem využila své vlastní znalosti s hudebním nástrojem. Následnou nahrávku jsem upravila v editoru Audacity [12].

Vývoj hry jsem si vybrala, jelikož je to mým koníčkem, ve kterém bych se chtěla zlepšovat.

2 Návrh a objektový návrh

Při tvorbě hry jsem používala iterativní model softwarového procesu. Tedy práci jsem si rozdělila do menších úseků, které jsem si prvně navrhla, poté naprogramovala a nakonec otestovala. Práce byla rozdělena následovně:

- Mapa a práce s ní
- Inventář
- Úkoly pro pracovníky
- Zákazníci a nakupování
- Pracovníci
- Čas
- Ukládání

Nyní bych se zaobírala důležitými prvky návrhu pro vybrané zajímavé úseky.

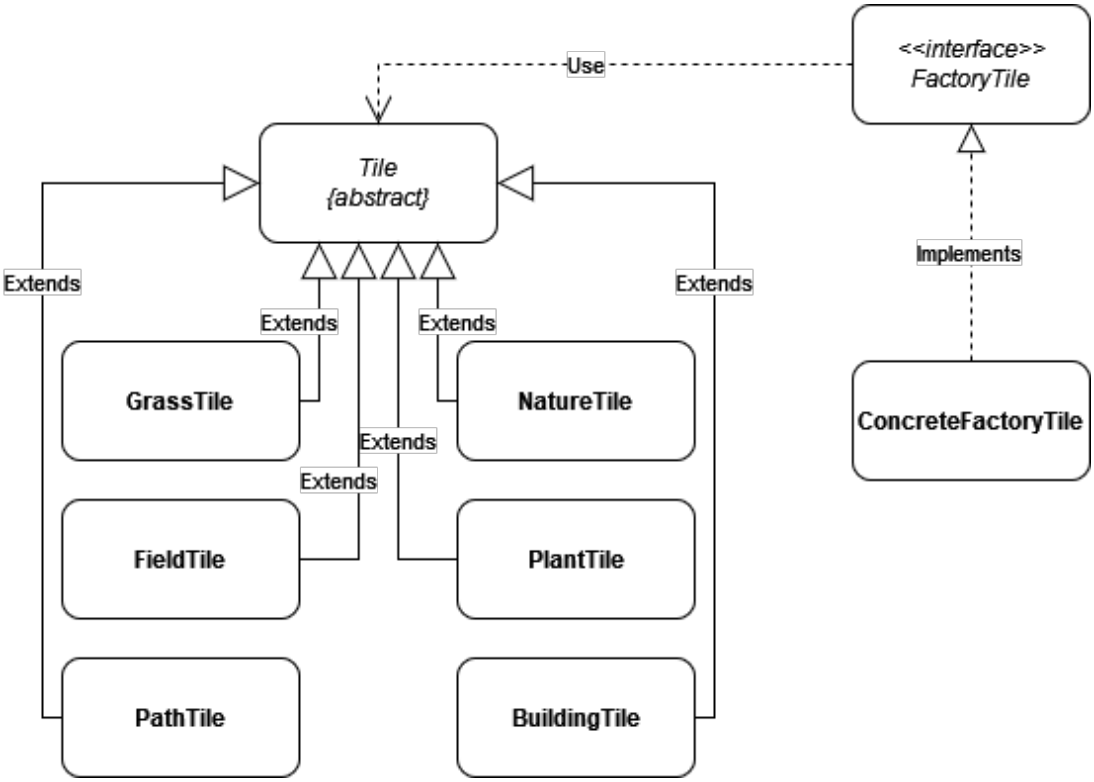
2.1 Mapa a práce s ní

Herní mapa je obdélníkového tvaru skládající se z dlaždic. Každá dlaždice na mapě je dána souřadnicemi tvaru (x, y). Ty určují, kde se dlaždice nachází. Dále každá dlaždice obsahuje určité informace důležité pro její typ. Ve hře mám několik typů dlaždic. Ty jsou:

- Tráva
- Pole
- Cesta
- Příroda
- Rostlina
- Budova

Při tvorbě mapy je důležité typy dlaždic rozlišovat. Každá nese jiné vlastnosti, jinak se chová, nebo má jiný vzhled. Třída tvořící nové dlaždice nedokáže předvídat typy nových dlaždic. Proto mi sem při návrhu krásně zapadl návrhový vzor Továrna.

Továrna je třídní vytvářecí vzor, jehož účelem je definovat rozhraní pro vytváření objektu, ale samostatné vytváření je ponecháno podtřídám. Výhoda tohoto postupu je v bezproblémovém přidání nové dlaždice.



Obrázek 1: Diagram tříd pro tvorbu nových dlaždic.

Všechny dlaždice na mapě spravuje třída správce mapy. Mezi funkcionalitu třídy správce mapy spadá:

- Uchovávání kolekce na dlaždice
- Uchovávání šířky a výšky mapy
- Uchovávání souřadnic pro důležité budovy
- Změna dlaždice
- Zjištění, jaká dlaždice se nachází na daných souřadnicích

Nejzajímavější funkcionalitou správce je metoda na změnu dlaždice. Ta se použije například při stavbě hliněné cesty. Dlaždice lze změnit z typu tráva na jakoukoliv ostatní (cesta, pole, ...). Naopak, kdybychom chtěli například místo pole postavit kamennou cestu, musíme prvně pole zničit a poté cestu postavit. Zničením pole se nám objeví opět dlaždice typu tráva. Jelikož je možné zrušit jakýkoliv úkol, musí si nové dlaždice pamatovat předchozí dlaždici, než se nová dlaždice postaví.

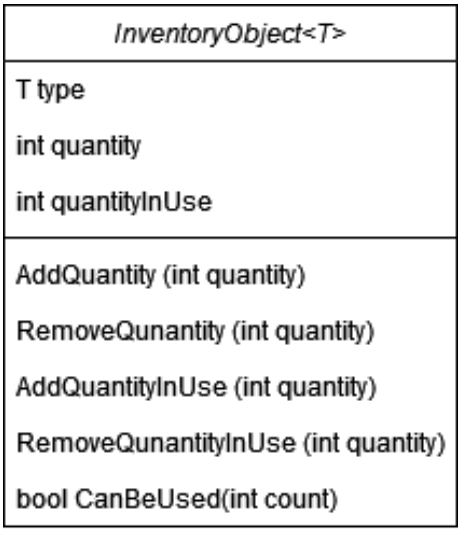
Menší komplikaci tvoří stavba dlaždic typu budova. Budovy zabírají obvykle více než jednu dlaždici. Tam, kam hráč klikne na mapě, se vloží střed budovy. Od

středu se pak vypočítají okraje podle rozměru. Aby existoval střed, je důležité, aby rozměr budovy byl vždy lichý. Ve středu jsou uloženy všechny informace o budově, okraje pouze odkazují na jejich střed. Speciálním okrajem je vstup, který může být důležitý pro pracovníky, či zákazníky pro interakci s budovou. Na vstup mají odkaz všechny dlaždice, které budova pokrývá.

2.2 Inventář

Mnou navržený inventář spravuje předměty v inventáři a mince.

Předmět inventáře může být tří typů. Semínko, nářadí a plodina z rostlin. Ke každému předmětu je potřeba znát počet kusů v inventáři a počet právě používaných kusů. Pro tyto účely jsem navrhla generickou třídu předmět inventáře. Kromě výše zmíněných vlastností poskytuje metody na správu kusů a používaných kusů. Má také predikát, zda objekt může být použit.



Obrázek 2: Návrh generické třídy objektu v inventáři.

Správu všech předmětů inventáře má na starosti třída správce. Pro každý typ předmětu má třída kolekci. Na uložení informace o mincích obsahuje třída primitivní číselný datový typ.

2.3 Úkoly pro pracovníky

Hráč během hraní hry vytváří nové úkoly pro pracovníky. Úkol může být například Postavit hliněnou cestu na souřadnicích $(2, 3)$ a hráč si může přát, aby tento úkol byl vypracován co nejrychleji.

Každý úkol má tedy vlastnosti, které ho jednoznačně identifikují. Tyto vlastnosti jsou:

- Priorita

- Typ úkolu
- Typ a subtyp dlaždice, s kterým je úkol spjat
- Seznam souřadnic, na kterých se úkol musí provést
- Seznam souřadnic, na kterých už je úkol prováděn

Díky prioritě určuji, jak rychle má být daný úkol vypracován. Čím vyšší číslo priority úkol obsahuje, tím se rychleji úkol musí splnit. Rychlost plnění úkolu je závislá i na pracovnících. Hráč může změnit prioritu úkolů, nebo smazat dané úkoly.

Ve hře existuje několik typů úkolů, které hráč může zadat. Tyto typy jsou:

- Stavění
- Orání
- Zalití
- Hnojení
- Sazení
- Sbírání
- Prodávání
- Nakupování

S typem úkolu souvisí typ a subtyp dlaždice. Touto dvojicí se určí například co se má postavit, jaká rostlinka se má zasadit, ...

Každý úkol má dva seznamy souřadnic. V prvním jsou uloženy všechny souřadnice, kterých se úkol týká. Tento seznam je prospěšný, aby se nehromadili stejné úkoly, pro každou dvojici souřadnic (x, y). Pokud přibyde nový úkol, který je podobný již existujícímu úkolu, přidá se souřadnice nového úkolu do seznamu již existujícího. Druhý seznam slouží k určení, na jaké souřadnici dlaždice již někdo pracuje.

S úkoly pracuje třída jejich správce. Ta obsahuje kolekci všech úkolů a metody na zacházení s nimi. Nejzajímavější metodou správce je hledání úkolu pro pracovníka. Tato metoda vybírá nejvhodnější úkol na základě priority úkolu a zájmu pracovníka k úkolu. Může se tedy stát, že pracovník nedostane nejdůležitější úkol, ale méně důležitý, který ho baví.

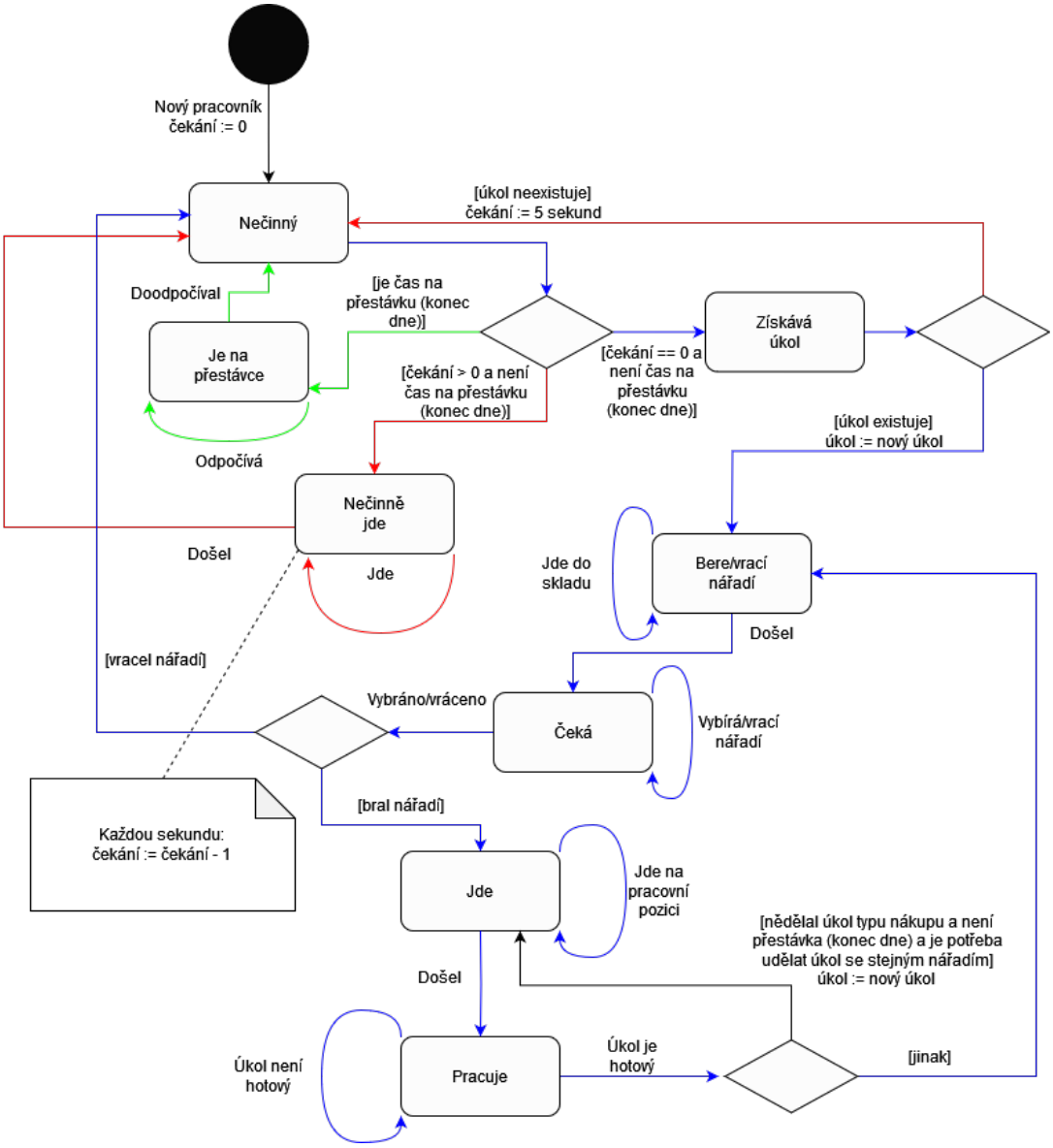
Speciální úkoly jsou typu prodávání a nakupování. Každý z nich má navíc jednu vlastnost. Musí mít určeny, s kým pracovník při dané práci spolupracuje. Tedy pro úkol typu prodávání je vlastnost, jakému zákazníku má balíček věcí prodat. Pro úkol typu nakupování je, od koho má koupit nákupní seznam.

2.4 Pracovníci

Pracovníci jsou jednotky, které nezávazně na hráči provádí úkoly. Nejdůležitější vlastností pracovníka je jeho stav. Podle stavu pracovník vykonává určitou činnost. Stav se mění v závislosti na předchozím stavu. Stav pracovníka může být jeden z následujících:

- Nečinný
- Nečinně jde
- Získává úkol
- Bere/vrací nářadí
- Jde
- Pracuje
- Čeká
- Je na přestávce

Změny stavu pracovníka by se daly popsat následujícím diagramem.



Obrázek 3: Stavový diagram třídy pracovník.

Základní průběh vykonávání činnosti pracovníka popisuje následující scénář. Scénář je rozdělen do několika průběhů, každý je na diagramu označen jinou barvou pro přehlednost. Normální průběh je vyznačen v diagramu modrou barvou. Průběh pokud úkol neexistuje je červenou, dále nastání přestávky(konce dne) je zelenou. Nakonec existence úkolu, pro který již pracovník má nářadí, je v diagramu černou barvou.

Scénář vykonávání činnosti pracovníka.

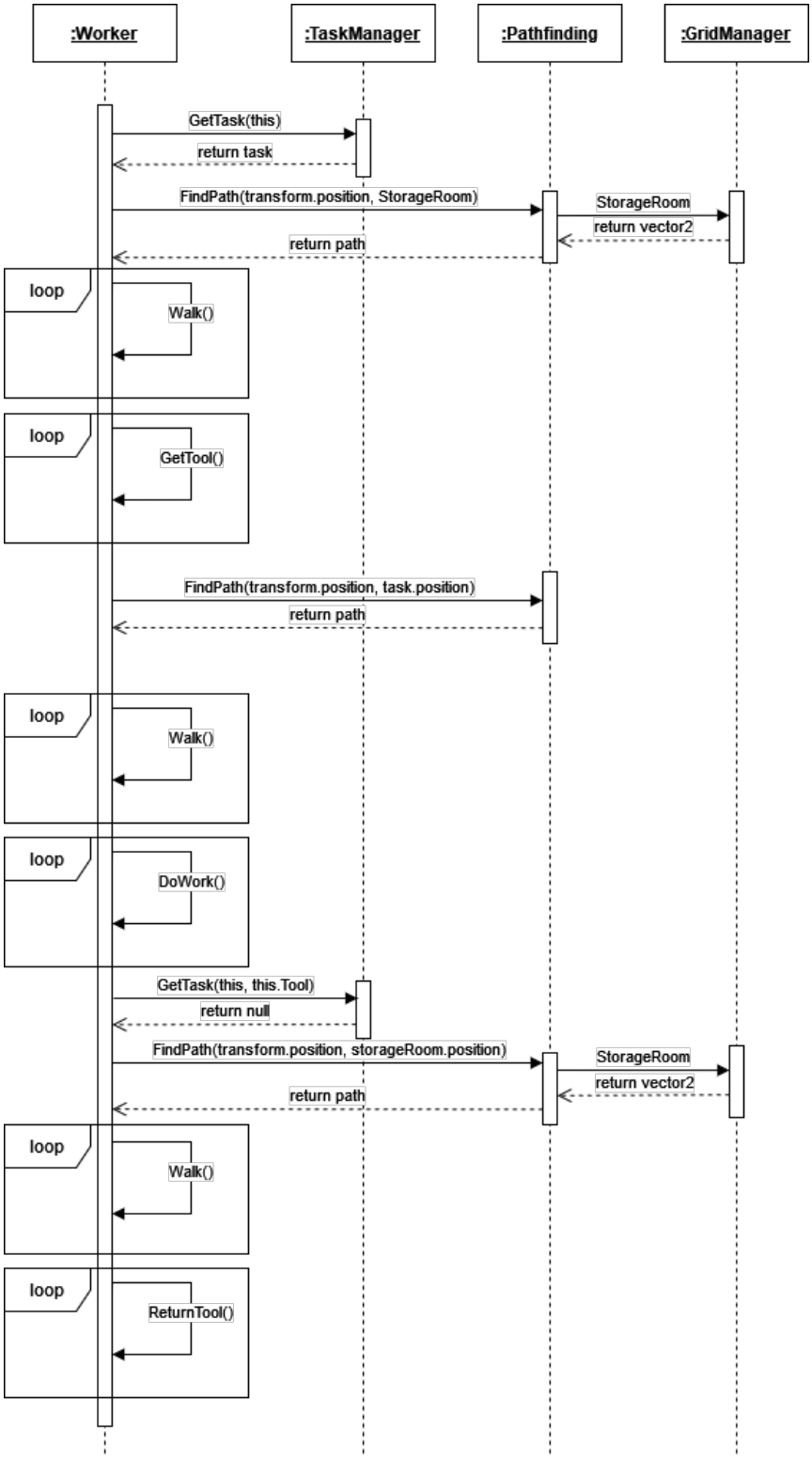
Normální průběh: Pracovník se chystá přejít ze stavu *Nečinný*. Jelikož čekání má nastaveno na 0 a čas přestávky zatím nenastal, přejde do stavu *Získává úkol*. Úkol podle správce úkolů existuje a pracovník přechází do stavu *Bere/vrací*

nářadí. V tomto momentu si uloží nový úkol, zjistí si potřebné nářadí pro daný úkol a vydá se pro něj do skladu. Když dojde ke skladu přepíná si stav na *Čeká*. Nyní si vybírá nářadí. Poté, co má vybráno, přechází do stavu *Jde* a přesouvá se na souřadnice své pracovní pozice. Když dojde na svoji pozici, začíná pracovat. Nyní je ve stavu *Pracuje*. Než dokončí svůj úkol, nemůže opustit svou pozici. Po hotovém úkolu se dostává zase do stavu *Bere/vrací nářadí*. Opět se vydává do skladu. U skladu se přepíná do stavu *Čeká* a vrací nářadí. Když je nářadí vráceno přepíná se do stavu *Nečinný*. Kde je možné získat další úkol pro pracovníka. Normální průběh pracovníka je také vyobrazen sekvenčním diagramem níže 4.

Průběh, když úkol neexistuje: Pracovník se nachází ve stavu *Nečinný* a pokouší se získat nový úkol. Přechází do stavu *Získává úkol*. Žádný však v daný moment neexistuje. Pracovník se vrací zpět do stavu *Nečinný* a do proměnné čekání si nastaví 5 sekund, aby nedošlo k opětovnému tázání na nový úkol. Nyní když se pokusí přejít ze stavu *Nečinný* dostává se do stavu *Nečinně jde*, jelikož v proměnné čekání je hodnota větší jak 0. V tomto stavu si pracovník zvolí náhodnou lokaci na mapě a jde k ní. Během chůze se mu snižuje každou sekundou hodnota proměnné čekání o 1. Když dojde na náhodnou pozici, opět přechází do stavu *Nečinný*.

Průběh, když nastane přestávka: Pracovník se může nacházet v jakémkoliv stavu, když mu přijde upozornění, že je čas přestávky. Do stavu *Je na přestávce* se může pouze dostat ze stavu *Nečinný*. Pokud je ve stavech *Získává úkol, Bere/vrací nářadí, Čeká, Jde, Pracuje* musí prvně dokončit získaný úkol, než se může vydat na přestávku. Pokud je ve stavu *Nečinně jde* musí prvně dojít na náhodně zvolenou pozici na mapě. Když se dostane do stavu *Je na přestávce*, prvně dojde k hlavní budově, vstoupí do ní a začne odpočívat. Až si odpočine, vrací se do stavu *Nečinný*. Stejný proces nastane, když pracovník dostane upozornění na konec dne.

Průběh, když existuje úkol, pro který pracovník má nářadí: Pracovník právě dokončil zadaný úkol. Chystá se změnit stav ze stavu *Pracuje*. Pokud existuje úkol, pro který již má u sebe nářadí, přepíná se do stavu *Jde* a jde na novou pracovní pozici. Na úkol s daným nářadím se neptá, pokud prováděl úkol nakupování nebo již je čas na přestávku. Pokud úkol s nářadím, které má u sebe, není nyní potřeba vykonat, přechází do stavu *Bere/vrací nářadí*.



Obrázek 4: Sekvenční diagram vyobrazující normalní průběh ze scénáře.

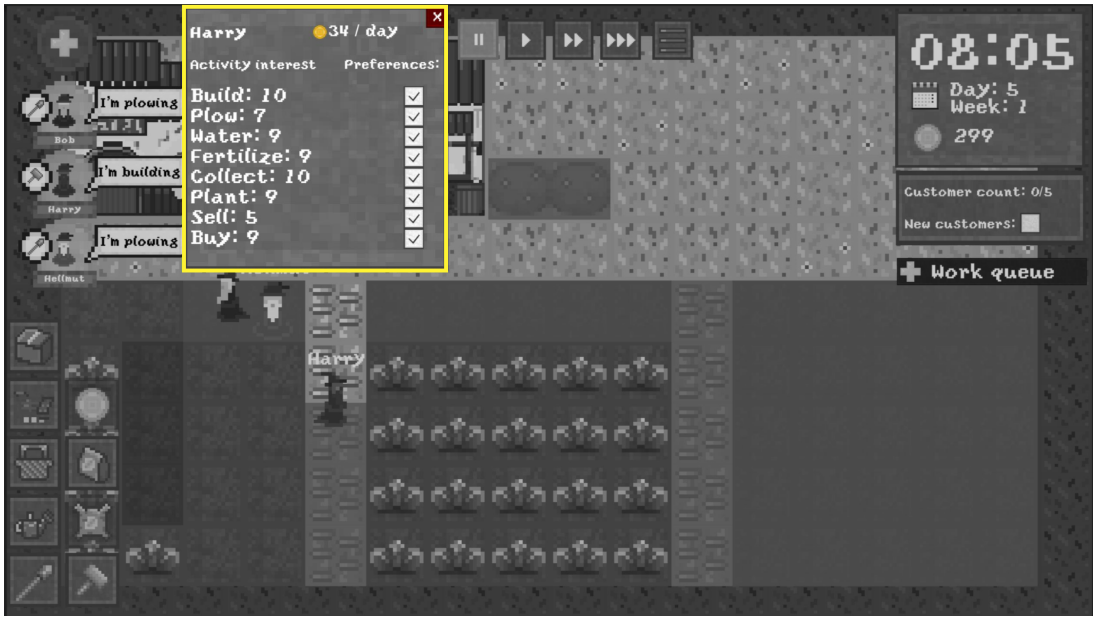
Diagram 4 je obohacen o názvy použitých metod při normálním průběhu po-

psaném v scénáři výše. Metoda a její implementace pro získání úkolu od správce úkolů je více rozepsána v kapitole Fronta úkolů a její přidělování pracovníkům 3.2. Jakým způsobem funguje hledání cest ve hře, je obsaženo v kapitole Hledání nejkratší cesty na mapě 3.1.

Malá nesrovnalost v použitých metodách je u metody `Walk()`, která v implementaci u pracovníka neexistuje. Chůze je řešena pomocí metody `FixedUpdate()`, která je součástí Unity. Tato metoda je volána automaticky jednou za pevný časový interval, který je nastaven v nastavení projektu. Hlavním účelem použití metody `FixedUpdate()` je přesné řízení pohybu objektů a simulace fyzikálních jevů. Metoda `FixedUpdate()` je vhodná pro tyto účely, protože se volá v pravidelných intervalech, což zajišťuje konzistentní chování fyzikálního systému nezávisle na rychlosti snímání obrazovky. Metoda `Walk()` je pouze pro účely srozumitelnosti diagramu.

Dalšími důležitými prvky u každého pracovníka jsou dvě kolekce. První obsahuje zájem pracovníka plnit typy úkolů. Například jak můžete vidět na obrázku 5 pracovník se jménem Harry má k úkolu typu stavění zájem 10. Což je největší možný zájem. Velikost zájmu určuje i rychlost vykonávání úkolů dané aktivity. Takže je výhodné si volit nové pracovníky s vysokými zájmy. Na druhou stranu podle zájmů se odvíjí i pracovníkům plat. Na obrázku 5 lze vidět, že Harry není zrovna levný pracovník.

Druhou kolekcí jsou hráčovi preference na plnění typů úkolů. Při vytvoření pracovníka se preference zvolí automaticky podle první kolekce. Poté v průběhu hry je hráč může měnit podle svého uvážení. Z obrázku 5 jde vidět, že pracovník Harry má zaškrtnuté všechny typy úkolů, tedy bude dělat všechny úkoly. Kdyby měl například prodávání odškrtnuté, protože nemá zrovna vysoký zájem k tomuto úkolu, nebude žádný úkol typu prodávání vykonávat.



Obrázek 5: Kolekce zájmů a hračových preferencí u pracovníka.

3 Zajímavé části implementace

V této části bych vyzdvihla několik řešení mého vývoje.

3.1 Hledání nejkratší cesty na mapě

Schéma problému:

Název: Nalezení nejkratší cesty v grafu

Vstup: Neorientovaný graf $G = (V, E)$, vrcholy $s, u \in V$ a hranové ohodnocení $d : E \to N$

Výstup: Nejkratší cesta z s do u.

Pro řešení tohoto problému jsem zvolila algoritmus A*. Nyní bych tento algoritmus více popsala. Znalosti spojené s algoritmem A* jsem studovala podle literatury [13] a [14]

Algoritmus na vstupu dostává neorientovaný graf $G = (V, E)$ a počáteční vrchol s a koncový vrchol u, pro které platí $s, u \in V$. Grafu v mojí hře odpovídá herní mapa. Množina vrcholů V je tvořena z jednotlivých dlaždic mapy. Množina hran E je definována následovně: Pro každé dva vrcholy $c, v \in V$ platí, že $(c, v) \in E$ právě když dlaždice, odpovídající vrcholům c, v, leží v herní mapě vedle sebe a obě jsou průchozí. Neprůchozí dlaždicí je například kámen nebo budova. Hranové ohodnocení d je ke každé hraně $(c, u) \in E$ vypočítáno podle rychlosti chůze na dlaždicích odpovídající vrcholům $c, u \in V$.

Algoritmus pracuje tak, že prohledává graf a snaží se najít nejkratší cestu od počátečního vrcholu k cílovému vrcholu. K tomu používá funkci obvykle označenou $f(x)$. Tato funkce se dá vyjádřit jako: $f(x) = g(x) + h(x)$.

Funkce $g(x)$ vyjadřuje cenu cesty od počátečního vrcholu k aktuálnímu vrcholu x.

Funkce $h(x)$ představuje heuristickou funkci, která odhaduje vzdálenost od aktuálního vrcholu k cílovému.

Při získávání hodnoty heuristické funkce $h(x)$ i funkce $g(x)$ využívám tzv. Manhattanskou vzdálenost. Manhattanská vzdálenost je vzdálenost mezi dvěma body v prostoru, kde se vzdálenost měří jako součet absolutních hodnot rozdílů souřadnic v jednotlivých osách. Cestování horizontálně a vertikálně má hodnotu 10, diagonálně 14. Funkce $h(x)$ je pouze odhadem, protože při jejím výpočtu nepočítám s žádnou překážkou v cestě. Funkce $g(x)$ je součtem Manhattanské vzdálenosti a hranových ohodnocení od počátečního k aktuálnímu vrcholu x.

Cílem algoritmu A* je najít cestu z počátečního vrcholu do cílového vrcholu tak, aby funkce f(x) byla minimalizována. Při každém rozšiřování vrcholu se vybírá vrchol s nejnižší hodnotou f(x), což znamená, že se preferují vrcholy, které mají nižší cenu dosažení a jsou blíže k cíli.

V mojí práci jsem naprogramovala funkci reprezentují algoritmus A*. Menším rozdílem z předešlého zadaní je, že má funkce nepřijímá jako argument dvě dlaždice, ale dvě souřadnice dlaždic. Později až v těle funkce zjišťuji dlaždice na dané souřadnici. Tento způsob mi přišel lepší, jelikož jakýkoliv objekt může lehce

```
1   private float GetDistance(Tile start, Tile end)
2   {
3     Vector2 startVector = start.transform.position;
4     Vector2 endVector = end.transform.position;
5
6     float xDistance = Mathf.Abs(Mathf.FloorToInt(startVector.x) -
          Mathf.FloorToInt(endVector.x));
7     float yDistance = Mathf.Abs(Mathf.FloorToInt(startVector.y) -
          Mathf.FloorToInt(endVector.y));
8
9     if(xDistance > yDistance)
10    return 14 * yDistance + 10 * (xDistance - yDistance);
11    return 14 * xDistance + 10 * (yDistance - xDistance);
12  }
```

Zdrojový kód 1: Funkce na získání vzdálenosti mezi dvěma dlaždicemi

přistoupit k svým souřadnicím. Dále ve funkci prohledávám mapu a snažím se najít posloupnost souřadnic od počáteční po koncovou souřadnici, která tvoří nejkratší cestu na mapě.

Na začátku metody si zjistím, jaká dlaždice reprezentuje počáteční souřadnice a jaká koncové. Jestliže je koncová dlaždice průchozí, pokračuji dál. Pokud ne, hledám nejbližší průchozí dlaždici, která se stane novou koncovou dlaždicí.

Poté potřebuji dvě kolekce pro prohledávání mapy. Otevřený seznam a uzavřený seznam. Otevřený seznam obsahuje dlaždice, které byly dosud objeveny, ale zatím nebyly zpracovány. Uzavřený seznam obsahuje dlaždice, které byly již zpracovány a byly tedy již vybrány a odebrány z otevřeného seznamu. Tyto dlaždice zůstávají v uzavřeném seznamu, aby se zabránilo opakovanému zpracování a prozkoumávání stejných dlaždic. Do otevřeného seznamu přidám počáteční dlaždici.

Následně v cyklu procházím otevřeným seznamem a vybírá dlaždici s nejnižší celkovou hodnotou $f(x)$ pro další prozkoumání. Tuto dlaždici pak odeberu z otevřeného seznamu a přidám ji do uzavřeného seznamu. Poté prozkoumám všechny sousedy právě prozkoumané dlaždice a pokud jsou průchozí a nejsou již v uzavřeném seznamu, přidám je do otevřeného seznamu. Pro každého souseda spočítám jeho celkovou hodnotu $f(x)$ a porovnám ji s jeho současnou hodnotou. Pokud je nová cena nižší, aktualizuje se cena a předek v cestě této dlaždice.

Pokud je koncová dlaždice nalezena, vrátím nejkratší cestu mezi počáteční a koncovou dlaždicí, reprezentovanou posloupností souřadnic. Pokud nebyla nalezena žádná cesta, vrátí nulovou hodnotu.

3.2 Fronta úkolů a její přidělování pracovníkům

Všechny úkoly uchovává správce úkolů, třída TaskManager. Úkoly jsou uloženy v seznamu podle jejich priorit. Úkol s nejvyšší prioritou je nejdůležitější a měl

by být co nejdříve vykonán.

Nové úkoly se do fronty přidávají nakonec, mají tedy nejhorší prioritu. Protože si myslím, že úkoly zadány dříve, chce hráč vykonat co nejrychleji. Jediný rozdíl je u úkolu typu prodávání, který se automaticky vloží na začátek s nejvyšší prioritou. Jako naznačení, že obsloužení zákazníků, by měla být hráčova priorita, jelikož na konci dne může kvůli tomu přijít o peníze.

Hráč může prioritu prací měnit v grafickém uživatelském rozhraní. Tato změna vymění priority prací a poté setřídí seznam prací, právě podle nich. Nakonec znovu načte objekty ve hře, aby změna byla viditelná i pro hráče.

```
1    public void ReloadTasks()
2   {
3     var taskCopy = new List<Task>(_tasks);
4     taskCopy.Sort();
5     DestroyAllTasks();
6     foreach (var task in taskCopy)
7     {
8       LoadTask(task.ActivityType, task.InteractionType, task.
            SubtypeTile, task.Range, task.IsBeingWorkedOn, task.Customer
            , task.BuyObject);
9     }
10   }
```

Zdrojový kód 2: Znovunačtení úkolů ve hře

Pracovník je od toho, aby pracoval. Pokud si pracovník hledá úkol, dotazuje se třídy TaskManager pomocí metody `GetTask(Worker worker)`, zda nějaký úkol pro něj existuje. Metoda vrací hodnotu pravda, pokud pracovník dostane práci, nepravda v opačném případě.

V těle metody, jak jde vidět ve zdrojovém kódu, se najde pracovníkovi nejvíce vhodný úkol. K tomu se využívá metoda `GetMostSuitableTask(Worker worker)`. Tato metoda prohledává celou frontu úkolů. Pomocí priorit a zájmů pracovníka najde pro daného pracovníka nejvíce vhodný úkol. Dále se v této metodě využívají preference pracovníka. Pokud má pracovník vypnutou preferenci u nějakého typu úkolů, tak se při vyhledávání tyto úkoly automaticky přeskakují.

Poté zkontroluji zda se nějaký úkol pro pracovníka našel. Jestliže ano, vytvořím objekt třídy TaskForWorker. Tato třída je zjednodušení třídy Task, která je normálně využívaná pro úkoly. Vytvořila jsem tuto třídu pro jednodušší zacházení z třídy pracovníka. Zjednodušený úkol se nastaví pracovníkovi. Nakonec informujeme správce inventáře o využívaných nástrojích k danému úkolu. S výjimkou úkolů typu prodávání a hnojení, jelikož ty se nastavují již při vzniku daného úkolu. A pracovník může jít dále spokojeně vykonávat úkol.

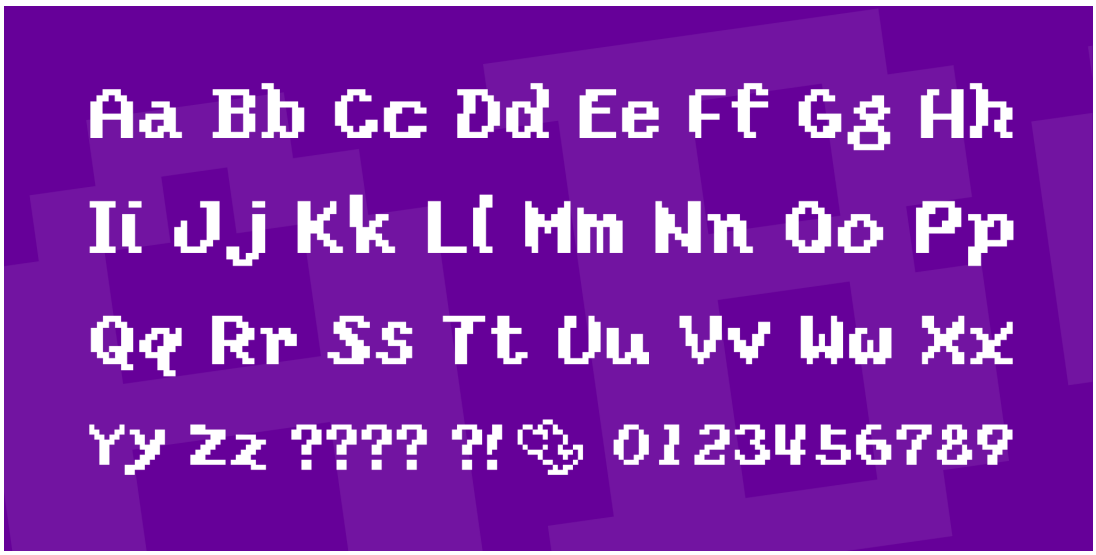
```
1    public bool GetTask(Worker worker)
2   {
3     var task = GetMostSuitableTask(worker);
4     if (task != null)
5     {
6       var taskForWorker = MakeTaskForWorker(task);
7       if (task.ActivityType == TypeOfActivity.Buy) task.BuyObject.
            Worker = worker;
8       worker.Task = taskForWorker;
9       if (taskForWorker.ActivityType != TypeOfActivity.Sell &&
10      taskForWorker.ActivityType != TypeOfActivity.Fertilize)
11      {
12        InformationAboutTypeOfActivity.Dict.TryGetValue(taskForWorker.
              ActivityType, out var info);
13        InventoryManager.Instance.UseTools(info.UsedTool, 1);
14      }
15
16      return true;
17    }
18    return false;
19  }
```

Zdrojový kód 3: Získání úkolu pro pracovníka

4 Uživatelská příručka

4.1 Font písma

I když jsem grafické prvky hry tvořila sama, na font písma jsem si netroufla. Font písma, který je používaný ve hře, je zdarma pro osobní i komerční účely. [15]



Obrázek 6: Používaný font ve hře.

4.2 Hlavní menu

Po spuštění hry se spustí hlavní menu. Kde hráč má několik možností výběru:

- "New Game". Spustí hráči novou hru.
- "Continue Game". Pokud hráč, má rozehranou hru, dovoluje v ní pokračovat. Opačně nabídka není ani možná zvolit.
- "Options". Hráč se dostane do voleb nastavení.
- "Quit". Hra se vypne.

Obrázek 7: Hlavní menu.

4.3 Příběh

Po spuštění nové hry se automaticky zapne scéna s příběhem. Jsi výborný programátor, jenom poslední dobou toho už máš dost. Žádný tvůj kód nefunguje. Když jdeš z práce domů, objeví se přímo před tebou kolo štěstí. Proč s ním nezatočit? Vyhrál jsi! Farmu? Tak jako proč ne? S touto myšlenkou se zázračně přemístíš do magického světa na farmu. A celá je jen tvoje! Ale počkat... Co je to tím malým písmem dole napsáno?

Balíček farma obsahuje 50 penízků do začátku. Také obsahuje zaměstnance, který požaduje 10 penízků každý den. Denně se také musí platit údržba budov, což je 20 penízků. Balíček také obsahuje nějaké nástroje a semínka. Pokud na konci dne máš méně než 0 penízků, farma je vrácena do kola štěstí pro "vyhořelé"programátory. Asi je čas začít pracovat...

Obrázek 8: Scéna z příběhu.

4.4 Ovládání

Kamera. Pro lepší přehled nad mapou může hráč pohybovat kamerou pomocí kláves WSAD nebo šipek. Oddalovat a přibližovat kameru lze pomocí kolečka myši. Pro stisku klávesnice R se kamera přesune do počáteční pozice.

Přidávání prací a jejich plnění. Pomocí tlačítek zobrazených na Obrázku 9 může hráč zadávat úkoly pracovníkům. Každé tlačítko vyobrazuje jiný typ úkolu. Pro rychlejší přístup má každé tlačítko klávesovou zkratku.

Obrázek 9: Tlačítka pro tvorbu prací.

Tlačítko vyobrazující kladívko zadává příkazy k stavbě. Zde si hráč může vybrat z cest a přírodních ozdob. Pokud se mu již jakákoliv věc na mapě nelíbí a postavil ji on sám, může ji kdykoliv zničit za pomoci červeného kladívka. Tlačítko pro stavbu má klávesovou zkratku B.

Upozornění pro hráče, pokud nějaká věc vypadá neprůchozí (například kámen), musí mít mezi sebou dvě neprůchozí dlaždice mezeru.

Tlačítko s lopatkou dělá nová pole. Klávesová zkratka je P. Tlačítko s hnojivem (konvicí, semínky) hnojí (zalévá, sází rostlinu). Lze pouze na dlaždici typu pole. Pro použití hnojiva nic nesmí být na poli zasazeno. Pro volbu hnojení (zalévání, sázení) stačí stisknout klávesu F (V, L). Tlačítko s košíkem umožňuje sbírat rostliny. Plodiny lze pouze získat z rostlin, které jsou plně vyrostlé. Sbírání lze zvolit klávesou C. Tlačítko s nákupním vozíkem slouží na nakupování ve městě. Rychlejší přístup je přes klávesu M. Tlačítko s penízkem zobrazuje dostupné zákazníky. Pokud nemáme žádné zákazníky, s tlačítkem nejde interagovat. Pokud máte zákazníky, lze menu otevřít i přes klávesu N.

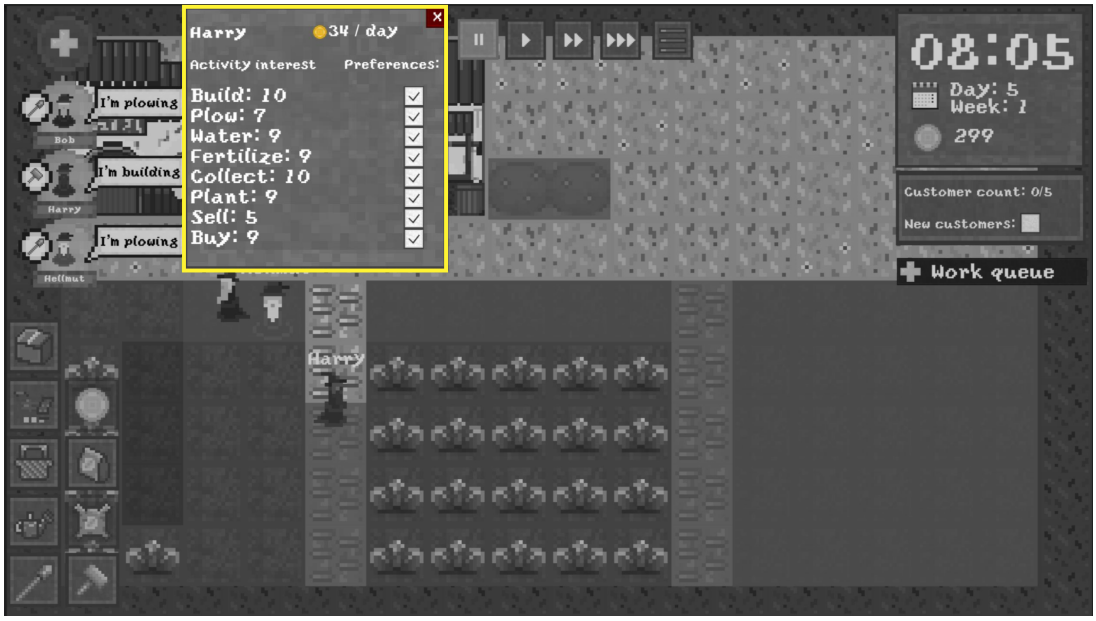
Na obrázku 10 je vyobrazena fronta úkolů, které ještě nejsou splněny. Hráč může měnit pořadí úkolů klikem na šipku v pravém horním rohu. Úkol lze zrušit klikem na křížek v levém horním rohu. Lze pouze zrušit úkoly, na kterých žádný pracovník nepracuje. Úkoly, na kterých někdo pracuje, jsou zvýrazněny žlutou barvou. Frontu úkolů lze skrýt klikem na tlačítko minus v levém horním rohu. Otevřít frontu úkolů lze přes tlačítko plus, které bude namísto tlačítka minus.

Obrázek 10: Fronta úkolů.

Pracovníci. Hra začíná s jedním pracovníkem, rychlost jeho práce je průměrná, ale má zájem k jakékoliv práci. Hráč může mít až 4 pracovníky. Nového pracovníka přidá klikem na tlačítko plus. Upozornění patří pro hráče, aby si své pracovníky vybírali důkladně. Jednou přijatý pracovník je tu pro Vás navždy. Nelze jej smazat.

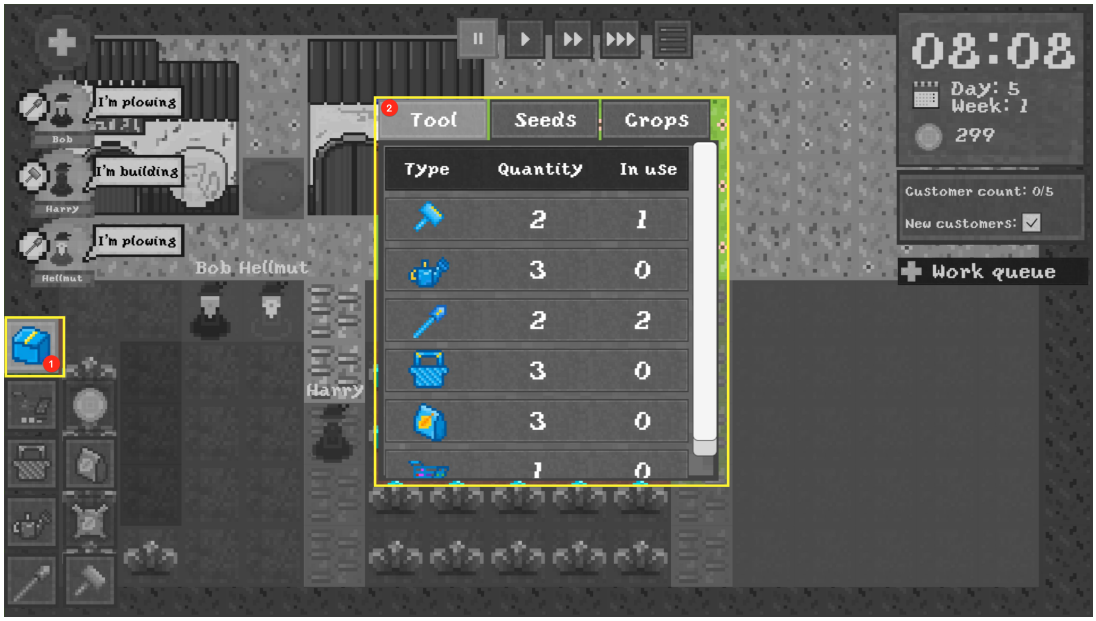
Všechny pracovníky vidí hráč v levém horním rohu, jak je vyobrazeno na Obrázku 11. U každého pracovníka hráč vidí, co právě dělají a jaký nastroj u daného úkolu používají. Po kliku na ikonu pracovníka, se hráči zobrazí okno s větším množstvím informací. Zde i může měnit preference pracovníka, Obrázek 12 .

Obrázek 11: Pracovníci.



Obrázek 12: Pracovníci – bližší informace.

Inventář. Inventář hráč zobrazí po kliku na tlačítko s obrázkem krabice, nebo může používat klávesovou zkratku I. Menu inventáře obsahuje tři záložky. Nářadí, semínka a plodiny. Po kliku na záložku se zobrazí obsah inventáře v dané kategorii. Jednotlivé položky mají obrázek dané věci s celkovým počtem kusů a počtem kusů právě využívaných.



Obrázek 13: Inventář.

Zákazníci. Maximální počet zákazníků je pět. Noví zákazníci chodí pravidelně každou hodinu, pokud je zaškrtnuté políčko "New customers". Pokud počet zákazníku je roven maximálnímu počtu, políčko se odškrtne a nejde zaškrtnout do doby, než se počet nezmenší.

Obrázek 14: Vytvoření nového zákazníka.

Nápověda. Pro lepší orientaci ve hře, jsem přidala nápovědu. Ta se zobrazí na určitých prvcích, když na ně najedeme kurzorem. Kromě názvu prvku se mohou zobrazit dodatečné informace. U tlačítek s klávesovou zkratkou se zobrazí daná zkratka. U rostlin se zobrazí délka růstu a u stavebních položek délka stavby.

Obrázek 15: Nápověda u tlačítka s klávesovou zkratkou.

Přechod zpět do hlavního menu a rychlé zavírání oken. Když chceme hru ukončit, klikneme na tlačítko vyobrazené na obrázku 16. Upozornění pro hráče, pokud hru opustíte v jiný čas, než na začátku nového dne Vaše hra nebude uložena! Hra se ukládá každý nový den. První uložení je druhý den ráno. Při ukládání se zobrazí ikona ukládání.

Poslední klávesovou zkratkou je klávesa Escape. Po stisku této klávesy se zavřou všechna otevřená okna a odeberou se všechna zvolená tlačítka. Kromě tlačítek spojených s časem.

Obrázek 16: Způsob, jak se dostat do hlavního menu.

Rychlost běhu času. Ve hře jsou k volbě 4 možné rychlosti plynutí času. První je pauza, tedy čas je zastaven. Druhý je normální rychlost. Každou naši sekundu ubyde ve hře 1 minuta. Třetí je 2-násobné zrychlení. Tedy každou naši sekundu ubydou ve hře 2 minuty. A na konec je 3-násobné zrychlení, což znamená, že každá naše sekunda jsou ve hře 3 minuty.

Obrázek 17: Tlačítka na volení rychlosti plynutí času.

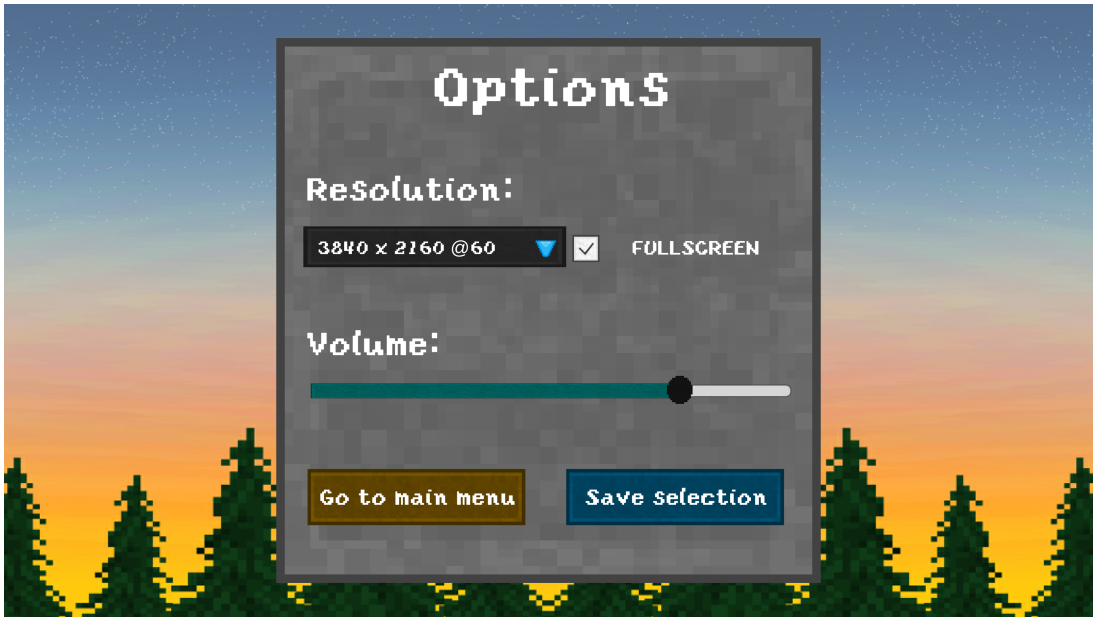
4.5 Rekapitulace dne

Den končí v 20:00. Po dosažení toho času, všichni pracovníci jdou do hlavní budovy, všichni zákazníci, kteří nečekají na vyřízení zakázky, vyrazí domů. Zákazníci, kteří museli čekat na vyřízení zásilky, dostanou slevu 50%. Když všichni pracovníci dorazí domů, objeví se okno s rekapitulací dne. V ten moment se strhnou poplatky spojené s budovami a proplatí se denní mzda pracovníkům. Pokud je hráč se svými penízky v nezáporných číslech, posouvá se do dalšího dne. Pokud penízky jsou záporné, hra skončila a může to zkusit znovu. Při pokračování do dalšího dne se hra ukládá.

Obrázek 18: Rekapitulace dne.

4.6 Nastavení

Ve volbě Options, v hlavním menu, se Vám zobrazí nabídka se základními prvky nastavení. Nachází se zde rozlišení obrazovky společně s obnovovací frekvencí, možnost zda chcete mít hru přes celou obrazovku a ovládaní hlasitosti. Při zvolení Vašich preferencí nezapomeňte na kliknutí tlačítka pro uložení, jinak se vše vrací do předchozího nastavení.



Obrázek 19: Ukázka zvoleného nastavení.

Závěr

V mojí bakalářské práci jsem vytvořila Management hru, která se odehrává ve fiktivním magickém světě. Tematicky jsem se rozhodla hru pojmenovat Magic: The Gardening, jelikož zde hráč ovládá pracovníky zahradnické firmy. Zvládla jsem vytvořit jednoduchý příběh ke hře, mapu hry, tvorbu úkolů a jejich přidělování pracovníkům. Samozřejmě jsem také naprogramovala jednotlivé pracovníky a inventář pro uchovávání různých položek. Kromě implementace jsem namalovala grafickou stránku hry a vytvořila hudbu na pozadí. Práce na projektu mi osobně přinesla cenné zkušenosti a moc jsem si ji užila.

V práci je možné nadále pokračovat. Hlavní věc, co mě napadá, je přidat druh úkolů "Vaření lektvarů". Dále mě napadá přidat magická zvířata, která by například snášela vzácné léčivé vejce. Pro lepší požitek ze hry by bylo zajímavé přidat různá vylepšení a předměty, které by se odemkly až za určité milníky ve hře. Vylepšení by mohla být rozšíření mapy nebo zvýšení atributu u pracovníka. Možností je hodně a určitě by to byla zajímavá výzva do budoucna.

Conclusions

In my bachelor thesis I created a Management game that takes place in a fictional magical world. I decided to name the game Magic: The Gardening, as the player controls the workers of a gardening company. I managed to create a simple story for the game, a game map, task creation and assigning tasks to the workers. I also programmed the individual workers and the inventory for storing various items. In addition to the implementation, I drew the graphics and created the background music. Working on the project was a valuable experience for me and I really enjoyed it.

The work could be continued. The main thing I can think of is to add a "Brewing Potion" type of task. I can also think of adding magical animals that would, for example, lay rare healing eggs. For a better enjoyment of the game, it would be interesting to add various upgrades and items that would unlock after some milestones in the game. Enhancements could be map expansions or attribute increases for a worker. There are a lot of possibilities how to continue and it would definitely be an interesting challenge for the future.

A Obsah elektronických dat

`bin/`
: Adresář obsahující hotovou spustitelnou hru.

`src/`
: Adresář obsahující projekt spustitelný v editoru Unity.

`doc/`
: Adresář obsahující vypracovaný text bakalářské práce.

`README.txt`
: Textový soubor s instrukcemi k spuštění hry a projektu.

Literatura

[1] Arts, Electronic. *SimCity*. Dostupný z: ⟨https://www.ea.com/cs-cz/games/simcity⟩.

[2] Studios, Two Point. *Two Point Hospital*. Dostupný z: ⟨https://store.steampowered.com/app/535930/Two_Point_Hospital/⟩.

[3] Studios, Ludeon. *Rimworld*. Dostupný z: ⟨https://store.steampowered.com/app/294100/RimWorld/⟩.

[4] Software, Introversion. *Prison Architect*. Dostupný z: ⟨https://store.steampowered.com/app/233450/Prison_Architect/⟩.

[5] Sawyer, Chris. *RollerCoaster Tycoon*. Dostupný z: ⟨https://www.rollercoastertycoon.com/⟩.

[6] *Herní engine Unity*. Dostupný z: ⟨https://unity.com/⟩.

[7] *Programovací jazyk C#*. Dostupný z: ⟨https://learn.microsoft.com/en-us/dotnet/csharp/⟩.

[8] *Vyvojové prostředí Visual Studio*. Dostupný z: ⟨https://visualstudio.microsoft.com/cs/⟩.

[9] *Editor Pixelart*. Dostupný z: ⟨https://www.pixilart.com/⟩.

[10] *Editor SFXR*. Dostupný z: ⟨https://sfxr.me/⟩.

[11] *Stránka freesound.org*. Dostupný z: ⟨https://freesound.org/⟩.

[12] *Program Audacity*. Dostupný z: ⟨https://www.audacityteam.org/⟩.

[13] Wikipedie. *A** [online]. [cit. 2021-7-9]. Dostupný z: ⟨https://cs.wikipedia.org/wiki/A*⟩.

[14] Computerphile. *A* (A Star) Search Algorithm - Computerphile* [online]. [cit. 2017-2-15]. Dostupný z: ⟨https://www.youtube.com/watch?v=ySN5Wnu88nE&list=PL1k1oeP9tX-um_KCaDL-8yeGaU8tjmlwD&index=8⟩.

[15] *Font používaný ve hře*. Dostupný z: ⟨https://www.1001fonts.com/yoster-island-font.html⟩.