FAKULTA
ELEKTROTECHNICKÁ
ČVUT V PRAZE

Testování herního enginu MIDGE2D

Semestrální práce v předmětu B6B36TS1 Testování softwaru

Joshua David Crofts

LS 23/24

Obsah

Úvod

Cíl dokumentu

Tento dokument popisuje testování softwaru – specificky herní engine MIDGE2D. Zde naleznete popis programu a testovacích scénářů.

Cíl práce

Cílem je navrhnout testovací scénáře a otestovat existující program MIDGE2D. Testy budou navrženy za pomocí způsobů testování přednášených v předmětu Testování softwaru.

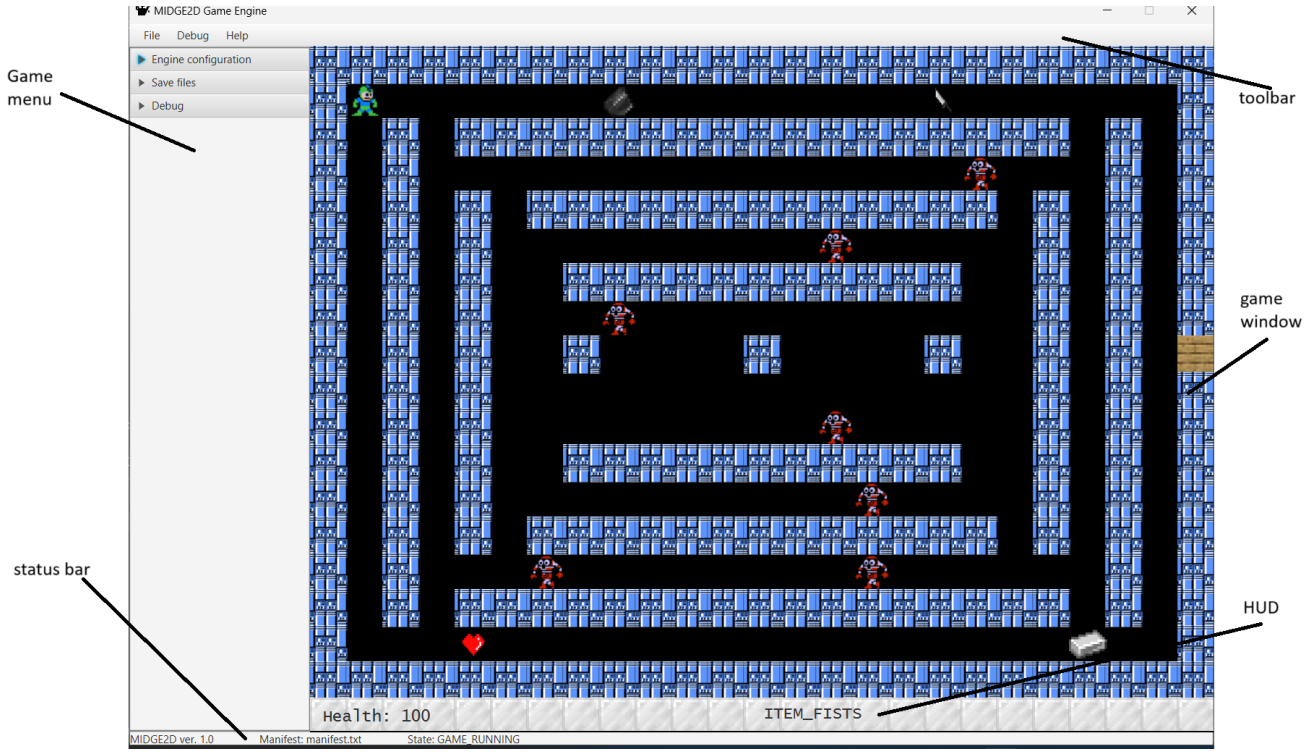
Výběr programu

Program je autorova semestrální práce v předmětu PJV – Programování v JAVA. Byl napsán podle zadání pro herní engine. Podporuje načítání mapy ze souboru v textové podobě, inventářový systém, ukládání a načítání, multithreading atd.

Popis programu

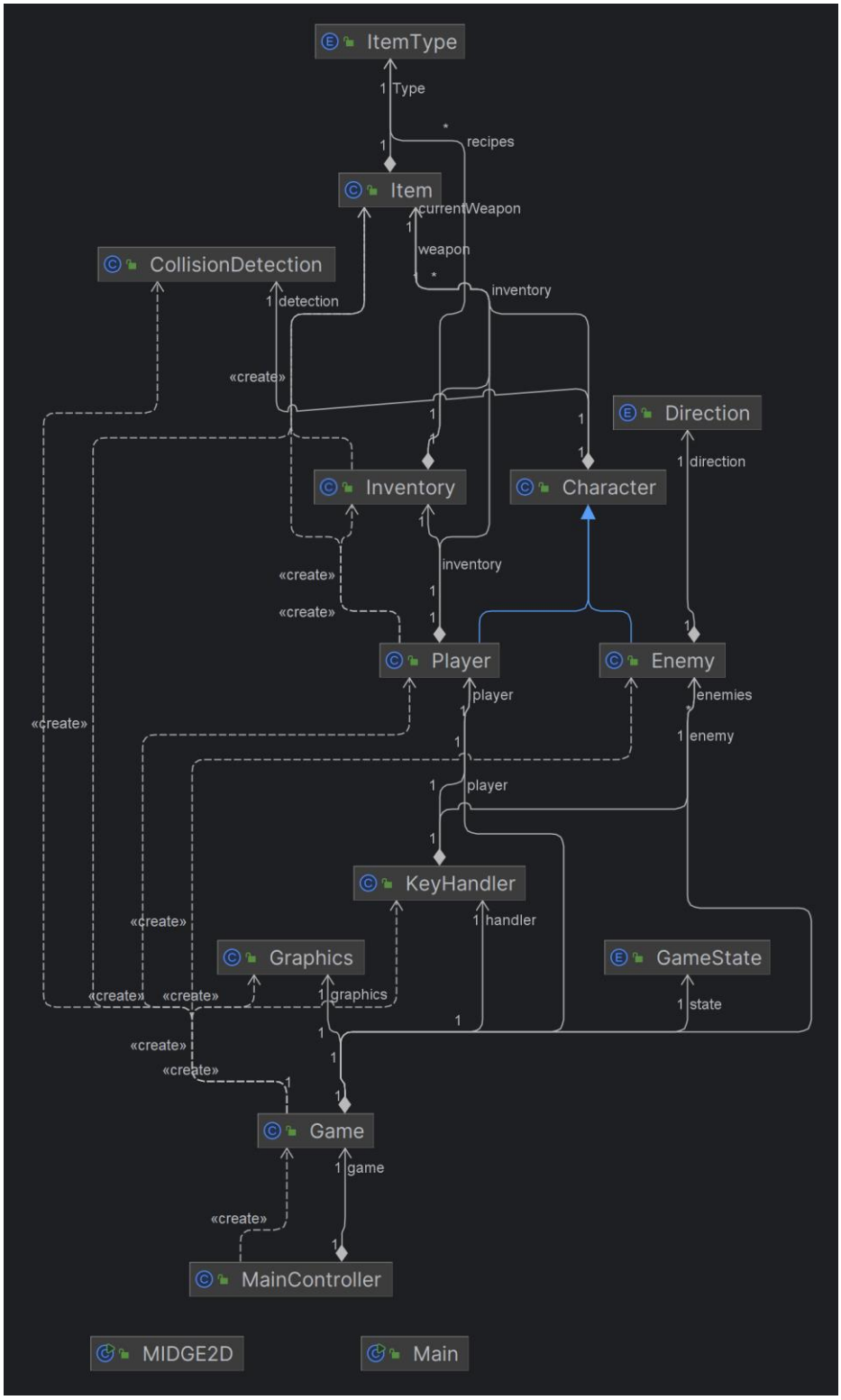
MIDGE2D je dvourozměrný herní engine podporující pohled seshora napsaný v programovacím jazyce Java, využívající knihovnu pro grafické rozhraní JavaFX. Hráč ovládá postavu pohybující se labyrintem, sbírající předměty a snažící se projít do dalšího levelu. Pomocí sesbíraných předmětů může hráč vytvářet nové předměty („craftit“). Hráč může narazit na nepřítele, kteří se pohybují zprava doleva. V případě střetnutí se pohyb hráče a nepřítele zastaví a nastává bojový režim. Pokud zdraví postavy klesne na nulu, hra končí.

Herní engine má grafické rozhraní umožňující další manipulaci s běžící hrou. Může se zde nastavovat režim logování, rychlost časovače hry či ukládání, popř. načítání herních stavů. Ukládání je umožněno pomocí serializace objektů do binárního souboru s příponou .midgesave – načítání pomocí deserializace a nastavení parametrů hry.



Obrázek 1 - Přehled programu MIDGE2D

UML Diagram tříd



Obrázek 2 - UML Diagram MIDGE2D

Popis tříd

MainController.java

Stará se o grafické rozhraní a interakci hráče s programem. Využívá JavaFX.

Game.java

Stará se o chod hry – načítá mapy ze souborů, umožňuje ukládání a načítání her, kontroluje stav hráče a nepřátel.

KeyHandler.java

EventHandler pro stisknutí kláves W, A, S, D, C, 1, 2, 3, 4, ENTER.

Graphics.java

Umožňuje vykreslování hry na canvas – využívá JavaFX.

CollisionDetection.java

Kontroluje, zdali může hráč či nepřítel pokračovat v chůzi, pokud ne, hlavní metoda vrací false.

Character.java

Logika pro chování všech postav. Z této třídy dědí další dvě třídy – Player.java a Enemy.java.

Player.java

Logika pro chování hráče.

Enemy.java

Logika pro chování nepřítele.

Item.java

Logika pro modelování předmětů ve hře.

Inventory.java

Logika pro manipulaci s předměty pomocí inventářového systému. Umožňuje vytváření nových předmětů atd.

Přehled enumů

Direction

Popisuje směry, kterými se může hráč a nepřátelé pohybovat.

```
MOVEMENT_UP,
MOVEMENT_DOWN,
MOVEMENT_LEFT,
MOVEMENT_RIGHT
```

GameState

Popisuje stav hry.

```
GAME_RUNNING,
GAME_STOPPED,
GAME_CRAFTING,
GAME_FINISHED,
MAP_COMPLETE
```

ItemType

Popisuje druh předmětu.

```
ITEM_KNIFE,
ITEM_FISTS,
ITEM_GUN,
ITEM_IRON,
ITEM_FLINT,
ITEM_STEEL
```

Návrh testovací strategie

Prioritizace částí aplikace

Proces	Podproces	Požadavek	Poškození	Vysvětlení	Část systému	Pravděpodobnost	Vysvětlení	Třída rizika
Chod hry	Tick	Kontrola stavu hry	H	Hra nebude reagovat	Game.java	M	Využívání multithreadingu, náročné kontroly každý tick	A
Grafika	Výkres grafiky	Vykreslování mapy na displej uživatele	H	Hra nebude správně zobrazována	Graphics.java	M	Využití knihovny JavaFX, málo zkušeností	B
Detekce kolize	-	Detekce kolize postav	M	Postavy budou chodit přes zdi	CollisionDetection.java	L	nenáročný kód na správu	C
Zpracování stisků kláves	-	Správné reakce na stisknutí kláves	M	Hra nebude responzivní	KeyHandler.java	M	JavaFX, málo zkušeností	B

Tabulka 1 - Prioritizace částí aplikace

Test levels

Část systému	Třída rizika	Revize	Vývojářské testy	Systémové testy	UAT	Test v produkci
Chod hry	A	ano	ano	ano	ne	ne
Grafika	B	ne	ano	ne	ne	ne
Detekce kolize	C	ne	ano	ne	ne	ne
Uživatelské rozhraní	B	ano	ne	ne	ano	ne

Tabulka 2 - Seznam test levels herního enginu

Testovací scénář

Testovací scénář signatury checkCoords(int x, int y) v třídě CollisionDetection.java

Třída CollisionDetection.java je třída s vysokou prioritou, jelikož běží při každém „ticku“ hry na každé vytvořené instanci postavy. Stará se o to, aby se postavy nemohly pohybovat přes objekty, u kterých to je zakázané (stěny, dřevo).Třídy ekvivalence signatury checkCoords(int x, int y)

- X < 0 && Y < 0 – nevalidní
- X >= 0 && X < počet řádků mapy && Y >= 0 && Y < počet sloupců mapy – validní
- X >= počet řádků mapy && Y >= počet sloupců mapy – nevalidní

Mezní hodnoty

Proměnná	Hodnota
X	-1
X	počet řádků mapy (=19)
Y	-1
Y	počet sloupců mapy (=26)

Tabulka 3 - Mezní hodnoty signatury checkCoords

Test cases

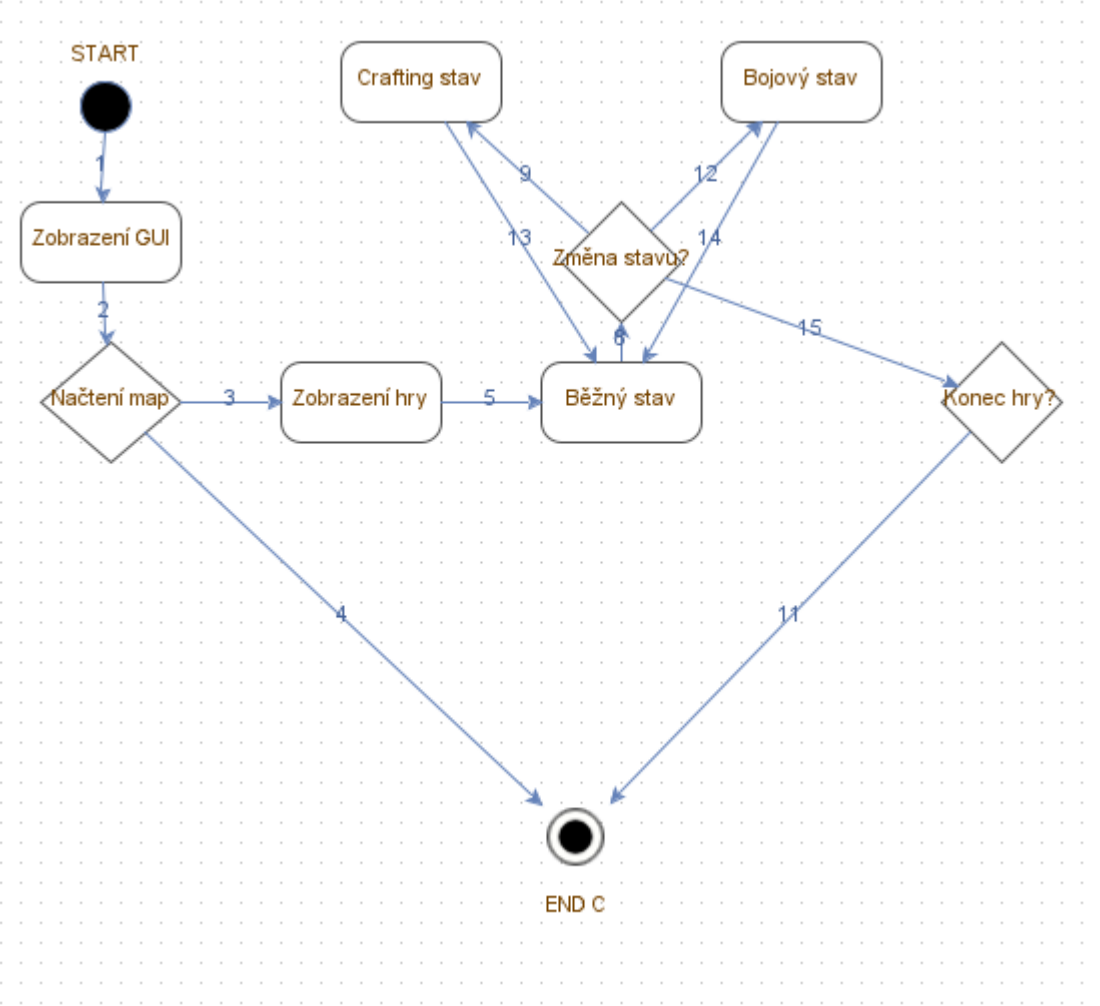
Test cases využívají výše zmíněné hodnoty. Páry byly vygenerovány pomocí programu allpairs.exe.

Case	Xcoord	Ycoord	Pairings
1	-1	-1	1
2	-1	0	1
3	-1	2	1
4	-1	26	1
5	0	-1	1
6	0	0	1
7	0	2	1
8	0	26	1
9	2	-1	1
10	2	0	1
11	2	2	1
12	2	26	1
13	19	-1	1
14	19	0	1
15	19	2	1
16	19	26	1

Tabulka 4 - Seznam vygenerovaných test cases

Test průchodů

Proces bude popisovat generování mapy a průběhu hry. Diagram a test cases byly vytvořeny pomocí programu PCTgen.



Obrázek 3 - Diagram průběhu hry

Node	Sub-combinations of edges
Běžný stav	5 - 8 14 - 8 13 - 8
Změna stavu?	8 - 9 8 - 12 8 - 15
Zobrazení hry	3 - 5
Bojový stav	12 - 14
Zobrazení GUI	1 - 2
Crafting stav	9 - 13
Načtení map	2 - 3 2 - 4
Konec hry?	15 - 11

Obrázek 4 - Tabulka podkombinací hran

No.	Test sequence
1	1 - 2 - 3 - 5 - 8 - 9 - 13 - 8 - 12 - 14 - 8 - 15 - 11
2	1 - 2 - 4

Obrázek 5 - Test cases

Detailní testovací scénář

Základní chod programu

Shrnutí: spuštění programu, načtení mapy, posunutí hráče o pár polí, uzavření aplikace

Pracnost: nízká

Popis: Uživatel spustí aplikaci. Uživatel vybere vhodný balíček map, který poté načte. Uživatel se libovolně posune o pár políček. Uživatel zavře hru.

Vstupní podmínky: program je vypnut, balíček map je připraven

Testovací data: připravený balíček map, obsahující soubor manifest.txt

Očekáváný výsledek: Nedojde k žádné komplikaci při spuštění programu a načtení mapy. Uživateli se zobrazí plně funkční a responzivní mapa, na které se budou pohybovat postavy. Uživatel sám bude moct ovládat hlavní postavu hry. Uzavření hry bez uložení proběhně bez problémů

Implementace testů

Zdrojový kód s testy je dostupný na Gitlab repozitáři:

https://gitlab.fel.cvut.cz/B232_B0B36PJV/croftjos

Závěr

Předmět TS1 byl pro mě náročnější, jelikož jsem neměl předtím zkušenost s testováním softwaru. Studijních materiálů je naštěstí dost, takže se člověk nemůže ztratit. Co mě docela mrzí na předmětu je jeho organizace. Popis domácích úkolů byl absolutně hektický, tak moc, že jsem většinou strávil víc času čtením zadání než pracováním. Jsem ale vděčný za to, že oproti minulým běhům se obsah semestrální práce zmenšil, což mi vyhovovalo, jelikož jsem měl hodně práce i v ostatních předmětech.

Touto prací jsem si osvědčil postupy při testování softwaru, od analýzy po psaní unit testů. Chtěl bych ale poukázat na to, že některé prezentace mohou být zastaralejší, protože jsem např. dostával samé chyby při používání frameworku Mockito, které jsem musel zdlouhavě řešit. Jinak práce odpovídá tomu, co se učilo přes semestr.

Seznam použitých obrázků a diagramů