Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE

Adam Huječek

Triplanetary

Katedra aplikované matematiky

Vedoucí bakalářské práce: RNDr. Josef Cibulka

Studijní program: Informatika, Programování (IP)

Studijní obor: Informatika

Praha 2012

Děkuji RNDr. Josefu Cibulkovi za trpělivé vedení této práce a veškerou jeho pomoc, bez které by tento projekt nemohl vzniknout. Také bych chtěl poděkovat své rodině za jejich neochvějnou podporu.

Prohlašuji, že jsem tuto práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 31. 7. 2012

Název práce: Triplanetary

Autor: Adam Huječek

Katedra: Katedra aplikované matematiky

Vedoucí bakalářské práce: RNDr. Josef Cibulka, Katedra aplikované matematiky

Abstrakt: Hlavním cílem této práce je převedení polozapomenuté tahové deskové stolní hry Triplanetary ze 70. a počátku 80. let minulého století na monitory dnešních počítačů. Program umožňuje více hráčům hrát na jednom počítači v takzvaném režimu hotseat dva z mnoha scénářů dostupných v původní hře - závodní Grand Tour s případnou účastí počítačem ovládaných protivníků a bojový Nova pro tři hráče. Nicméně díky vhodnému návrhu lze snadno implementovat i zbývající scénáře či samozřejmě přidat i úplně nové za předpokladu znalosti jazyka JAVA, ve kterém je hra naprogramována. Další předností je možnost hru kdykoliv uložit a přerušit a později se k ní vrátit.

Klíčová slova: tahová stolní hra, šestiúhelníková hrací plocha

Title: Triplanetary

Author: Adam Huječek

Department: Department of Applied Mathematics

Supervisor: RNDr. Josef Cibulka, Department of Applied Mathematics

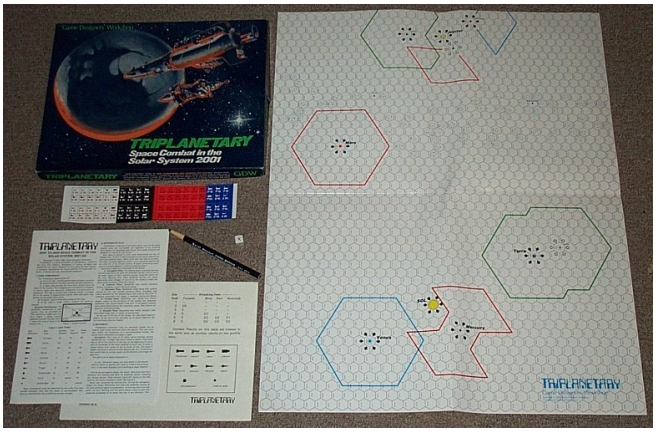
Abstract: The main purpose of this work is to bring the almost forgotten turn-based board game Triplanetary from the 70s and the beginning of the 80s of the last century to the screens of today´s computers. The program allows multiple players to play on one computer in the so called hotseat mode two of the several scenarios available in the original game - racing Grand Tour with the option to play with computer controlled opponents and battle Nova for three players. However thanks to the suitable design it is easy to implement the rest of scenarios or of course add completely new ones provided the user has the knowledge of JAVA which the game is programmed in. Another advantage is the option to save and exit the game at any time and return to it later.

Keywords: turn-based board game, hexagonal game board

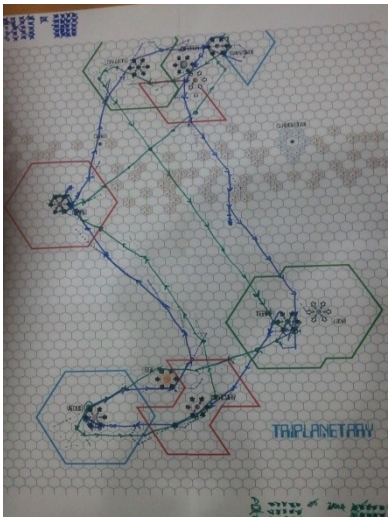
Obsah

Úvod

Původní Triplanetary [1][2] je vědecko-fantastická válečná stolní tahová hra vytvořená Marcem W. Millerem a Johnem Harshmanem ze společnosti Game Designers´ Workshop v roce 1973 s podtitulem „Ship to Ship Space Combat in the Solar System, 2001 AD“ (Vesmírný boj lodě proti lodi ve Sluneční soustavě roku 2001). Druhé vydání pravidel od Johna M. Astella z roku 1981 přineslo nové scénáře a drobné úpravy pravidel a jejich dovysvětlení. V současné době vlastní autorská práva Steve Jackson Games[3].



Obrázek 0-1: Původní balení



Obrázek 0-2: Průběh hry

Základní pravidla hry (více vizte první kapitolu) společná pro všechny scénáře (který každý používá ještě svoje specifická pravidla) se mohou zpočátku zdát složitá, ale v podstatě na nich není nic komplikovaného. Hra se dělí na tahy odpovídající jednomu dni a tah se skládá ze segmentů (jednoho za každého hráče) a ty zase z fází (Navigace, Výzbroj, Pohyb, Boj a Zásobování). Ve svém segmentu hráč ovládá svou flotilu vesmírných lodí – případně i své planetární a orbitální základny a tak dále – na hrací ploše tvořené šestiúhelníky reprezentující naši hvězdnou soustavu. Samotnou herní desku pokrývala fólie, na kterou se speciálními mazatelnými fixy zakreslovaly pohyby lodí jednotlivých hráčů, jejichž cíle, rozložení sil, typy lodí a jejich vybavení a podobně závisí na konkrétním scénáři, pro který se rozhodli, vždy ale musí respektovat první Newtonův pohybový zákon[4]:

Jestliže na těleso nepůsobí žádné vnější síly, nebo výslednice sil je nulová, pak těleso setrvává v klidu nebo v rovnoměrném přímočarém pohybu.

To znamená, že pokud loď nezažehne své motory a nepřipraví se tak o cenné palivo, nebo se na ní neprojeví gravitace nějakého kosmického tělesa, pohybuje se v segmentu svého majitele úplně stejně jako kolo minulé. Podobný princip můžeme nalézt i v klasické podlavicové hře Formulky hrané na čtverečkovaném papíře. Stejná pravidla platí i pro torpéda, miny a atomové bomby, které spolu s palubními zbraněmi a taranováním slouží k vesmírným soubojům.

Tato práce se zabývá počítačovou hrou Triplanetary, která si stolní verzi bere za vzor a snaží se věrně převést klasickou hratelnost na počítačové obrazovky. V první

kapitole naleznete stručný výtah z původních pravidel, který samozřejmě není vyčerpávající, klade si za úkol pouze uvést čtenáře do problematiky.

Z technických důvodů však bylo nutné pozměnit některá herní pravidla, o čemž pojednává druhá kapitola, stejně jako o chybějících či naopak přidaných prvcích.

Třetí kapitola se věnuje instalaci a spuštění programu, jeho grafickému uživatelskému rozhraní a ovládání hry.

Ve čtvrté kapitole naleznete popis samotné implementace hry, popis externích souborů, použitých knihoven a samozřejmě nejdůležitějších balíčků a tříd programu. Také se zde nachází seznámení s nejzajímavějšími algoritmy použitými ve hře, jako je například umělá inteligence ve scénáři Grand Tour.

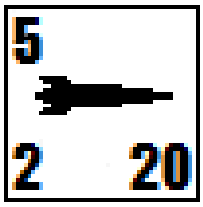
V závěru práce jsem zhodnotil celý projekt a zamyslel se nad jeho budoucností.

1. Stručný výtah z původních pravidel Triplanetary

Tato kapitola si neklade za cíl nahradit pravidla[2], nabízí pouze stručný přehled toho nejdůležitějšího, co člověk musí mít na paměti, když se snaží proniknout do programu přenášejícího tuto hru na obrazovky počítačů.

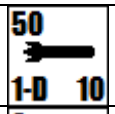
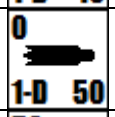
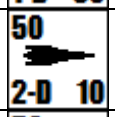
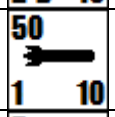
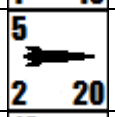
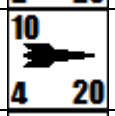
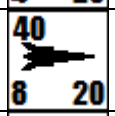
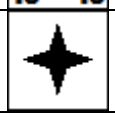
1.1. Žetony lodí

Každý typ lodí má svůj vlastní žeton, který reprezentuje všechny instance plavidel tohoto typu.



Obrázek 1-1: Žeton korvety (5 tun nákladu, 2 bojová síla, 20 jednotek paliva)

V levém horním rohu se nachází údaj o velikosti nákladového prostoru, číslo pod ním v levém dolním rohu udává bojovou sílu. Poslední informací na žetonu je kapacita palivových nádrží v pravém dolním rohu.

Žeton	Typ lodi	Síla	Palivo	Nosnost	Cena (Cr)	Cena (B)
	Transport	1D	10	50	10	5
	Tanker	1D	50	0	10	5
	Liner	2D	10	50	50	10
	Packet	1	10	50	20	10
	Korveta	2	20	5	40	20
	Křižník	4	20	10	80	40
	Fregata	8	20	40	150	80
	Dreadnaught	15	15	50	600	150
	Orb. základna	16	0	-	1000	160

Tabulka 1: Třídy lodí

Civilní lodě mají u síly příponu D a nemohou útočit, protiútočit ani přetěžovat motory. Cena (Cr) určuje cenu v kreditech, cena (B) v bodech podle bojové síly (používáno např. ve scénáři Nova).

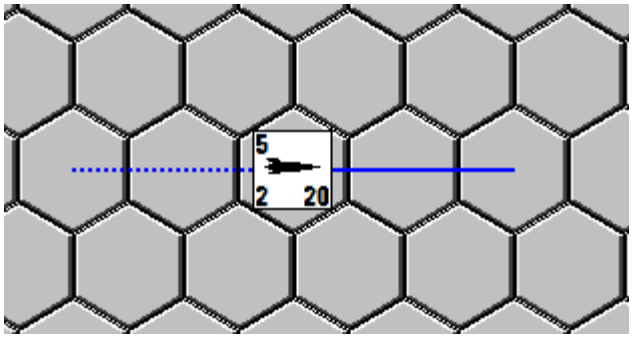
1.2. Tah

Hra se dělí na tahy reprezentující jednotlivé dny. Tahy se skládají ze segmentů za každého hráče. Pořadí hráčských segmentů v tahu se určí hodem kostkou nebo vzájemnou dohodou. Každý segment obsahuje tyto fáze:

- Navigační (více v 1.3) – hráč na tahu prozkoumá pozice svých lodí, orbitálních základen, torpéd, min a atomových bomb a podle předchozích kurzů určí jejich další pohyb a případně se rozhodne zažehnout motory nebo (ne)použít slabou gravitaci
- Výzbroje (více v 1.4.2) – vesmírné lodě mohou vypustit torpéda, miny a atomové bomby
- Pohybová – lodě, miny, torpéda a atomové bomby hráče na tahu se pohnou podle vypočítaných kurzů; pokud zbraně zasáhnou cíl, vybuchnou ještě tuto fázi
- Bojová (více v 1.4.1) – lodě mohou bojovat respektujíce pravidla pro boj, napadené lodě mohou provádět protiútoky; jestliže objekt v pohybové fázi prolétal asteroidy, na poškození se hází tuto fázi
- Zásobovací (více v 1.6.4) – lodě mohou načerpat palivo, nabrat zásoby, vykládat a nakládat, rabovat a zachraňovat

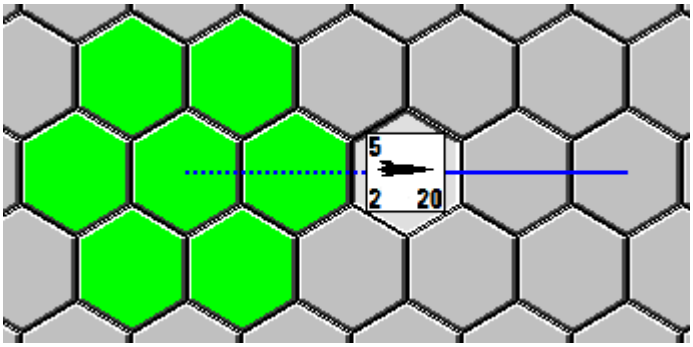
1.3. Pohyb

Každá loď má pohybový vektor vedoucí z její předchozí polohy do té současné. Základní pravidlo zní: v pohybové fázi svého majitele všechny lodě neurychlené zášlehem motoru nebo gravitací vesmírného tělesa urazí stejnou vzdálenost stejným směrem, jako se pohybovaly v minulém kole.



Obrázek 1-2: Zachování hybnosti

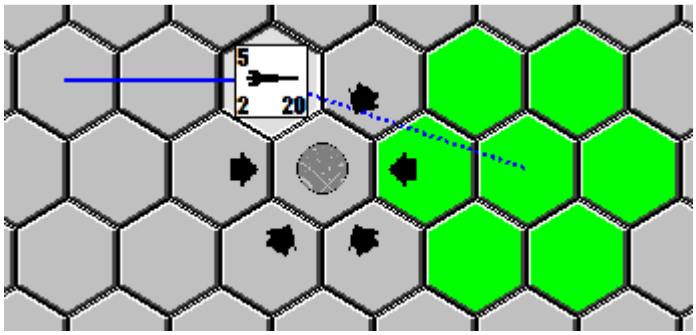
Lodě mění kurz spalováním paliva během navigační fáze, kdy se spočítaný kurz změní o jeden hex v libovolném směru za každou použitou jednotku paliva. Civilní lodě mohou spálit jen jednu jednotku za kolo, bojové lodě až dvě jednotky použitím přetížení motorů, vizte 1.3.4.



Obrázek 1-3: Zážeh motorů

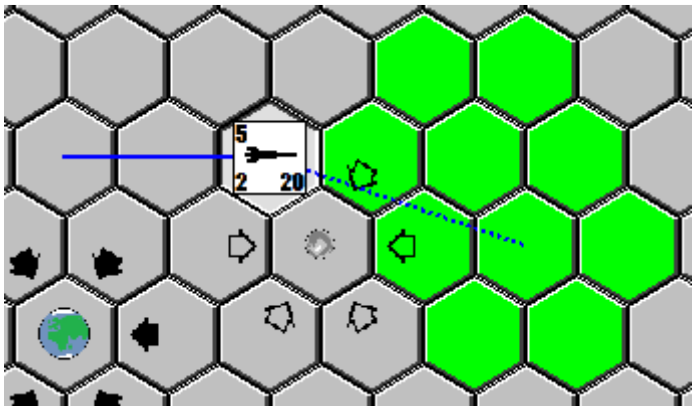
1.3.1. Gravitace

Planety, přírodní družice a Slunce působí na blízko prolétající objekty gravitační silou, reprezentovanou na hrací ploše šipkami na hexech kolem těchto těles. Průlet tímto hexem se projeví v následujícím kole stejně jako spálení jedné jednotky paliva ve směru šipky. Efekt políček s gravitací je kumulativní a dosahuje i na sousední část hexu s vesmírným tělesem vyvolávajícím gravitaci.



Obrázek 1-4: Vliv gravitace

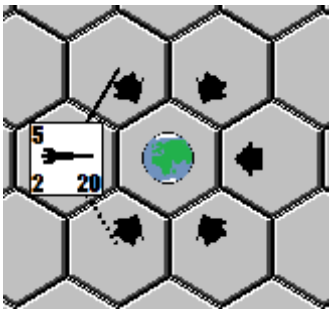
Některé přírodní satelity (Měsíc a Io) mají slabou gravitaci reprezentovanou neplnou šipkou. Loď prolétávající jedním takovým hexem může efekt gravitace ignorovat. Pokud ale objekt přejde přes dva hexy se slabou gravitací za sebou, druhá gravitace se chová jako silná gravitace bez ohledu na to, jestli ta první byla brána v potaz nebo ne.



Obrázek 1-5: Slabá gravitace a možnost jejího ignorování

1.3.2. Orbita

Loď pohybující se v gravitačním poli vesmírného tělesa rychlostí jednoho hexu za kolo bez spalování paliva se nachází na orbitě tohoto tělesa.



Obrázek 1-6: Loď na orbitě

1.3.3. Vzlet a přistání

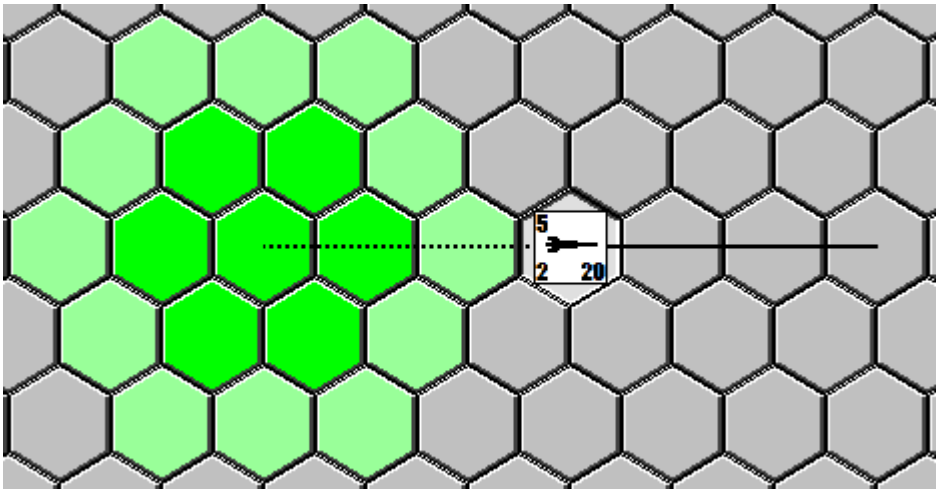
Lodě mohou vzlétat pouze z přátelských planetárních základen. Při vzletu loď získá rychlost jeden hex za kolo a dostane se na hex sousedící s planetou. Její rychlost se ale vyruší s gravitací planety, a pokud příští kolo nezažehne motory do správného směru, zřítí se zpět na povrch planety.

Lodě, jejichž kurz protne nějaké vesmírné těleso, jsou zničeny, nicméně jestliže spálí jednu jednotku paliva, mohou přistát na libovolné straně planety, kterou právě obíhají. V dalších kolech mohou zase vzlétnout, ale jen skrz hex, z něhož přistály.

Na Ceres a Clandestine se přistává prostým zastavením na jejich hexech.

1.3.4. Přetěžování motorů

Válečné lodě mohou přetížit své motory a tím spálit dvě jednotky paliva v jednom kole. Další takové přetížení ale lze provést až po opravě motorů spojené s doplněním paliva u přátelské stanice.



Obrázek 1-7: Přetížení motorů

1.3.5. Taranování

Na jednom hexu může být libovolné množství lodí a nic jim ani nebrání sebou navzájem prolétat. Nicméně hráč na tahu se může pokusit svou lodí taranovat ty soupeřovy, jež se nacházejí na hexech, jejichž středy útočníkova loď prolétá, ale jen jednou za kolo. Poškození způsobené taranem oběma lodím se vyhodnotí hodem kostkou podle následující tabulky (význam vizte 1.4.3):

Hod	Poškození
1	-
2	-
3	-
4	-
5	D2
6	D2

Tabulka 2: Taranování

Miny, torpéda a atomové bomby samy nemohou taranovat a explodují, jsou-li úspěšně taranovány.

1.4. Boj

Boj je pokus znehybnit nebo zničit nepřátelskou loď pomocí taranu, min, torpéd, atomových bomb a palubních zbraní. Miny, torpéda a atomové zbraně se vypouštějí během fáze Výzbroje, taranování probíhá v pohybové fázi a z palubních zbraní se střílí v bojové fázi.

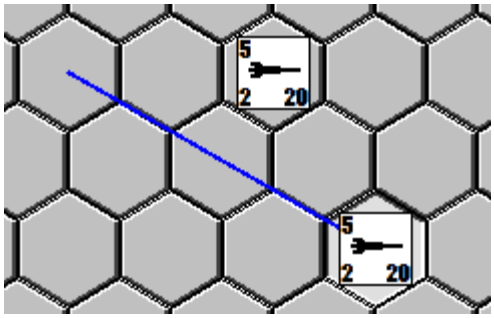
1.4.1. Palubní zbraně

Bojová síla lodě zahrnuje jak zbraně a munici při útoku, tak i strukturální integritu při obraně. Při boji se vyhodnotí poměr součtu sil útočících lodí k součtu sil bránících se lodí a zaokrouhlí se ve prospěch obránce na jednu z hodnot v tabulce poškození. Poté se hodí kostkou a k výsledku se přičtou opravy za vzdálenost a relativní rychlost a podle tabulky se všem bránícím se lodím přidělí příslušné poškození. Vizte 1.4.3.

Jestliže je na hexu více lodí, útok může směrovat na libovolný počet z nich. Pokud útočí více lodí, pak se bere největší modifikátor za vzdálenost i relativní rychlost. Každá loď může být napadena jen jednou za bojovou fázi a stejně tak může jen jednou za bojovou fázi útočit. Útok mohou zahájit pouze lodě hráče na tahu, nicméně do protiútoků se mohou zapojit lodě všech hráčů.

1.4.1.1. Dosah a bojová vzdálenost

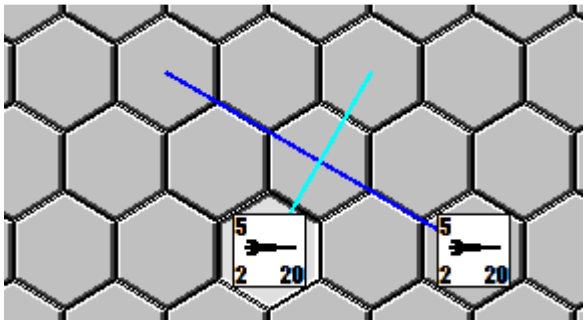
Vzdálenost mezi obráncem a útočníkem je modifikátor bojového hodu. Počítá se mezi nejbližší pozicí útočníka k obránci v tomto kole, a pokud je útočících lodí více, počítá se největší vzdálenost.



Obrázek 1-8: Vzdálenost je dva, ale bojová vzdálenost je jen jedna

1.4.1.2. Relativní rychlost

Rozdíl relativních rychlostí útočníka a obránce je modifikátor bojového hodu a spočítá se jako vzdálenost konců jejich pohybových vektorů vycházejících ze společného počátku. Pokud tato vzdálenost překročí dva, od bojového hodu se odečte její hodnota zmenšená o dvojku. Útočí-li více lodí, bojový hod se sníží o největší možný postih.



Obrázek 1-9: Relativní rychlost je čtyři, postih k hodu je tedy dva

1.4.1.3. Protiútoky

Napadené lodě mohou provést protiútok proti libovolnému počtu útočníků ještě před samotným udělením poškození. Protiútok, do kterého se také mohou připojit všechny lodě sdílející s napadenou lodí polohu a kurz, pak probíhá stejně jako obyčejný útok. Jestliže protiútočí více lodí, bere se vždy největší postih za vzdálenost a relativní rychlost. Civilní lodě nemohou provádět útoky ani protiútoky.

1.4.2. Výzbroj

Pokud loď převážející miny, torpéda nebo atomové bomby právě netankuje, nepřistává nebo nevzlétá, pak z ní může hráč vypustit ve fázi výzbroje jeden takový objekt, který se pak pohybuje v jeho pohybové fázi, je ovlivňován gravitací a který vybuchne po pěti kolech, nebo vletí-li na hex obsazený vesmírným tělesem, lodí, minou, torpédem nebo atomovou bombou.

Miny – výbušniny bez vlastního pohonu, které po vypuštění převezmou vektor své mateřské lodě, která tedy musí změnit kurz, aby se své vlastní mině vyhnula. Mina vybuchne, jestliže jakoukoli částí jejího hexu proletí nějaká loď nebo pokud naopak ona proletí hexem jiného plavidla nebo plavidel, a každá postižená loď si poté zvlášť hodí kostkou a obdrží poškození podle následující tabulky (význam vizte 1.4.3):

Hod	Poškození
1	-
2	-
3	-
4	D1
5	D2
6	D3

Tabulka 3: Poškození po kontaktu s minou

Palubní zbraně a planetární obrana nemají na miny žádný vliv, ostatní miny, torpéda a atomové bomby ano. Miny váží deset tun a každá loď schopná je převážet je může i vypustit.

Torpéda – mobilní výbušné nálože s jednoduchým naváděcím systémem, které sice po vypuštění převezmou vektor své mateřské lodě, ale mohou ve fázi svého vypuštění zrychlit o dva hexy v libovolném směru. Torpédo vybuchne, jestliže vletí na hex obsazený nějakou lodí, která si poté hodí kostkou a obdrží poškození podle následující tabulky (význam vizte 1.4.3):

Hod	Poškození
1	-
2	D5
3	E
4	E
5	E
6	E

Tabulka 4: Poškození po kontaktu s torpédem

Jestliže se na hexu nachází více lodí, hází postupně, dokud není jedna poškozena nebo zničena, nebo dokud všechny neuniknou bez jediného škrábance. Palubní zbraně a planetární obrana nemají na torpéda žádný vliv, ostatní torpéda, miny a atomové bomby ano. Torpéda váží dvacet tun a vypouštět je mohou pouze válečné lodě.

Atomové bomby – jaderné zbraně určené k devastaci povrchu planet, po vypuštění převezmou pohybový vektor mateřské lodě, která tedy musí změnit kurz, aby se své vlastní bombě vyhnula. Atomová bomba vybuchne, jestliže vletí do hexu obsazeného lodí, základnou nebo asteroidy, ale ne torpédem, minou nebo jinou bombou, a zničí vše na tomto hexu. Pokud zasáhne povrch planety, zničí vše na straně, kam dopadla. Torpéda a miny se k atomovým bombám chovají jako k lodím. Palubní zbraně a planetární obrana na ně útočí vždy poměrem 2:1, ale počítají se běžné opravy za vzdálenost a relativní rychlost. Poškozená atomová bomba nemůže fungovat. Jaderné zbraně váží dvacet tun a každá loď schopná je převážet je může i vypustit, ale civilní lodě mohou mít na palubě pouze jednu.

1.4.3. Poškození

V tabulce poškození jsou tři druhy výsledků:

- Pomlčka značí, že útok nebyl úspěšný a cíl neutrpěl žádné znatelné poškození
- DN – loď je na N (1-5) tahů znehybněna, to znamená, že nemůže zažehnout své motory, útočit ani vypouštět miny, torpéda ani atomové bomby; výjimkou jsou lodě třídy Dreadnaught, které mohou pálit z palubních zbraní; poškození je opravováno ve stejné fázi, jako bylo uděleno
- E – loď je zničena

Hod	Poměr sil útočníků k obráncům				
	1:2	1:1	2:1	3:1	4:1
1	-	-	-	-	D2
2	-	-	-	D2	D3
3	-	-	D2	D3	D4
4	-	D2	D3	D4	D5
5	D2	D3	D4	D5	E
6	D3	D4	D5	E	E

Tabulka 5: Boj a poškození

Poškození je kumulativní, pokud by loď měla být znehybněna šest a více kol, je zničena.

1.5. Navigace

Většina políček hrací plochy reprezentuje prázdný prostor, a nepředstavuje pro navigátory žádné problémy. Některé hexy ale obsahují jasné nebezpečí, se kterým se musí počítat.

1.5.1. Vesmírná tělesa

Na mapě existují tři typy vesmírných těles: Slunce, planety a přirozené satelity. Každé má určitou velikost a nezabírá celý hex, na kterém se nachází, a pouze loď, která se střetne se samotným povrchem, je zničena. Zbytek hexu se považuje za součást okolních políček s gravitací.

Na Slunci se nedá přistát a loď, která se o to pokusí, je zničena. Na planetách lze přistát za předpokladu, že se na povrchu nachází základna.

1.5.2. Asteroidy

Asteroidy představují pro rychle letící vesmírné lodě vážné riziko. Plavidla prolétající polem asteroidů rychlostí větší než jeden hex za kolo hodí kostkou a obdrží poškození podle následující tabulky:

Hod	Poškození
1	-
2	-
3	-
4	-
5	D1
6	D2

Tabulka 6: Poškození po střetu s asteroidy

1.6. Základny

Základny se nacházejí na planetách, měsících a asteroidech a slouží jako zdroj paliva a planetární obrany. U každého scénáře se uvádí, kde se které základny nacházejí a komu patří.

1.6.1. Planetární základny

Základnám na planetách a měsících se říká planetární a slouží jako zdroj planetární palby a paliva. Lodě, které přistály na planetární základně, jsou imunní vůči palubním zbraním, minám, torpédům a taranování, ale ne atomovým bombám. Na druhou stranu takové lodě nemohou pálit ze svých palubních zbraní ani vypouštět miny, torpéda a atomové bomby.

1.6.2. Základny v asteroidech

V pásu asteroidů jsou dvě základny na Ceres a Clandestine, které slouží stejně jako planetární základny, jen místo planetární obrany jednou za kolo mohou vypustit torpédo.

1.6.3. Orbitální základny

Orbitální základny se příliš neliší od těch v asteroidech, jen jsou semi-mobilní. Každá váží padesát tun. Pouze lodě třídy Transport je mohou převážet a vypouštět a to ještě jen na orbitě planety nebo satelitu nebo na povrchu planety, který ještě neobsahuje základnu. Jednou vypuštěnou základnu již nelze naložit zpátky.

Funkcí se orbitální základny neliší od planetárních (pokud stojí na planetě) nebo asteroidových (pokud obíhají na orbitě).

1.6.4. Zásobování

Během zásobovací fáze mohou základny doplňovat vesmírným lodím palivo, miny, torpéda a atomové bomby, kterých má každá základna neomezené zásoby. Pro přenos zásob z asteroidové nebo orbitální základny s ní musí zásobovaná loď vyrovnat kurz. Planetární základny zásobují lodě, které na nich přistály, nebo které nad nimi prolétají na orbitě.

Kdykoli loď získá zásoby ze základny, prochází také údržbou, což bojovým lodím umožní opět použít přetížení motorů. Zásobované lodě nemohou ve stejném hráčském segmentu pálit ze svých palubních zbraní a vypouštět miny, torpéda a atomové bomby, stejně tak zásobující základna nemůže ve stejném hráčském segmentu pálit z planetárních zbraní ani vypouštět torpéda.

1.6.5. Planetární obrana

Planetární základny mohou během bojové fáze svého majitele pálit na nepřátelské lodě nacházející se přímo nad nimi používajíce vždy poměr 2:1 a ignorujíce modifikátory za vzdálenost a relativní rychlost.

1.7. Scénáře

Původní stolní hra Triplanetary obsahuje množství scénářů, z nichž byly v tomto projektu implementovány zatím dva.

1.7.1. Grand Tour 2047

Tento scénář je v pravidlech [2] uveden následujícím příběhem:

Gesichtkreis Sternschiffbau A.G. nabízí velkou odměnu vítězi výroční velké ceny. Tato událost je závodním vrcholem každé dekády pro vrcholové závodníky a návrháře vesmírných lodí.

Lodě: každý závodník dostane jednu korvetu na Zemi.

Zvláštní pravidla: palivo lze čerpat jen na základnách na Zemi, Venuši, Marsu a Callisto, boj je povolen.

Vítězství: každá loď musí proletět alespoň jedním hexem gravitačního pole každého vesmírného tělesa s plnou gravitací a přistát na Zemi; první, komu se to podaří, vyhrává – pokud se více závodníkům povede přistát ve stejném kole, rozhodne menší spotřeba paliva.

Varianty: po několika hrách se hráči naučí optimální cesty, proto se místo toho může závodit po předem zvolené trase, například se kontrolní body musí navštívit v abecedním pořadí, podle velikosti, vzdálenosti od Slunce a podobně.

1.7.2. Nova

Tento scénář je v pravidlech [2] uveden následujícím příběhem:

Dokonce i přežití celé rasy může být podřízeno malicherné politice, jak by to možná mělo být. Jestliže je přežití pro nejsilnější, mají pak slabí právo žít?

Tento scénář je určen pro tři hráče: proamerický Západní blok, komunistický Východní blok a Mimozemské útočníky.

Lodě: Jak Západní, tak i Východní blok sestaví své flotily za 500 bodů. Mimozemšťané dostanou flotilu čtyř křižníků.

Speciální pravidla: Východní a Západní blok hodem jedné kostky určí, které kolonie jim budou patřit: 1=Venuše, 2=Mars, 3=Ceres, 4=Callisto, 5=Clandestine, 6=Merkur. Pokud padne stejné číslo, hází se znovu. Kolonie mají na povrchu jen jednu základnu. Východní blok si vybere tři souvislé strany zemského povrchu, Západní blok získá zbylé tři. Západní blok si vybere jednu stranu měsíce, Východní blok poté také jednu. Po umístění všech pozemských lodí na základny svých majitelů, se objeví mimozemské lodě na straně hrací desky nejblíže Jupiteru. Cílem mimozemské flotily je zničit Slunce bombami Nova, které převážejí její sebevražedné lodě a které automaticky vybuchnou, pokud se dostanou na orbitu Slunce. Kromě Nova bomb mají mimozemské lodě plný náklad min. Zásobování je

dostupné na všech přátelských základnách, pozemští hráči mohou nabídnout své základny druhému bloku.

Vítězství: Východní nebo Západní blok vyhrává zničením poslední mimozemské lodě. Pro tyto účely je zničení definováno i jako znehybnění lodi, která se poté už nemůže vyhnout srážce nebo opuštění hrací plochy. Mimozemšťané vyhrávají odpálením Nova bomby na orbitě Slunce.

Varianty: Změňte velikost mimozemské nebo pozemské flotily podle průběhu a výsledku předchozích her.

2. Odlišnosti od stolní verze Triplanetary

Tato kapitola pojednává o změnách v herních pravidlech, které bylo nutné provést z technických důvodů.

2.1. Chybějící prvky

Počítačová verze Triplanetary na rozdíl od té stolní neobsahuje možnost rabovat znehybněné lodě a předávat mezi loďmi palivo, což značně zjednodušuje celou zásobovací fázi a zpřehledňuje grafické uživatelské rozhraní.

Hráči se také nemohou vzdávat a toto rozhodnutí úzce souvisí s vyškrtnutím rabování a obtížným objektivním určováním, kolik paliva je potřeba pro bezpečný návrat na základnu, jelikož pravidla požadují, aby právě takové množství pohonných hmot na vzdávající se lodi zůstalo.

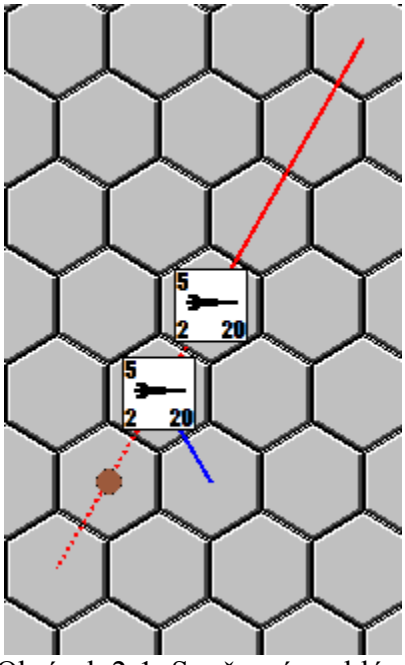
2.2. Přidané prvky

Z důvodu zachování věrnosti předloze jsem nepřidával žádné nové prvky, ale se znalostmi programovacího jazyka JAVA lze přidat nové druhy statických i pohybujících se objektů a do jisté míry upravovat i chování hry.

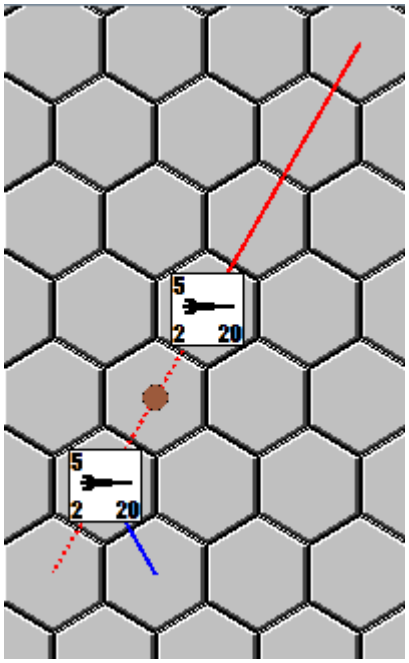
2.3. Pozměněná pravidla

Protože ovládání počítačové hry klade úplně jiné nároky než hraní deskové hry, bylo potřeba přepracovat jednotlivé fáze hráčských segmentů:

První dvě fáze (Navigace a Výzbroj) zůstávají beze změny, jen navigační fázi jsem přejmenoval na pohybovou, protože pohybová fáze z původních pravidel nyní probíhá automaticky po fázi Výzbroje, a není ji proto třeba speciálně označovat. Poté nastává úplně nová fáze nazvaná Taran, kde přeživší lodě mohou taranovat lodě, které na své cestě potkaly. Bohužel to znamená, že například loď, která ve svém tahu opustila hrací plochu a tím se zničila, nemůže předtím taranovat nepřítele na svém kurzu. Alternativou by bylo provádět samotný pohyb až po taranovací fázi, ale to by zase hrozilo, že loď bude na své cestě zničena dříve, než se k cíli svého útoku dostane.

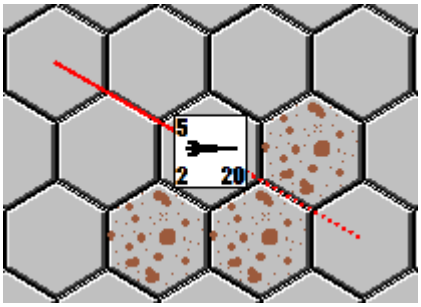


Obrázek 2-1: Současný problém taranování



Obrázek 2-2: Problém alternativního taranování

Druhou změnou původních pravidel je chování při průletu asteroidy. Lodě, které prolétávají přesně po společné straně dvou hexů obsahujících meteory, si házejí na poškození dvakrát, zatímco ve stolní hře stačí hod jeden. Toto zjednodušení souvisí s algoritmem použitým pro získávání políček, přes která loď proletěla, jenž se musí používat i pro ostatní objekty jako miny a podobně. Hráče to alespoň donutí chovat k pásu asteroidů patřičný respekt.



Obrázek 2-3: Průlet dvěma asteroidovými poli

3. Uživatelské rozhraní a ovládání

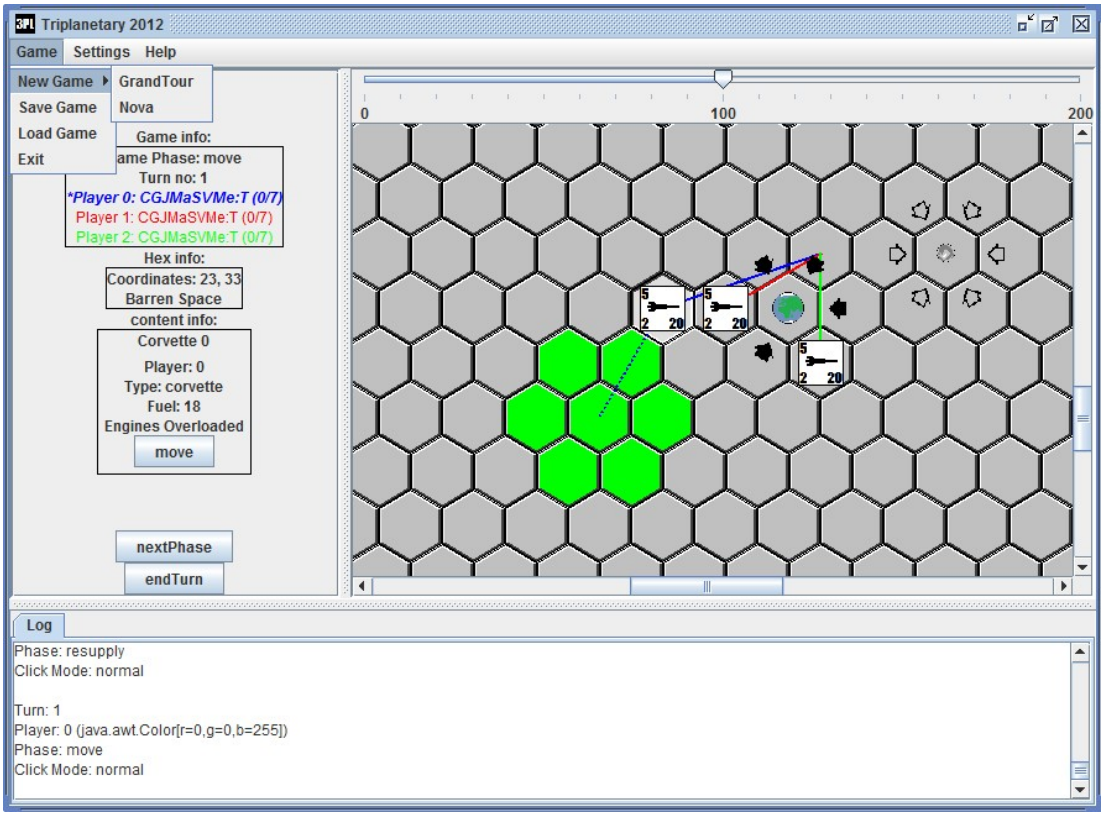
Pro spuštění hry stačí rozbalit distribuční archiv a zadat tento příkaz v adresáři hry:

java -cp classes;lib/HexBtn.jar triplanetary.Triplanetary

za předpokladu, že uživatel má nainstalovanou Javu verze alespoň 6 (lze ji zdarma stáhnout z webu firmy Oracle[7]) a její adresář bin v cestě. Aby si tento nepříliš intuitivní příkaz nemusel hráč pamatovat, nachází se v adresáři s hrou dva skripty: triplanetary.bat pro Windows a triplanetary.sh pro Unix.

3.1. Popis uživatelského rozhraní

Grafické uživatelské rozhraní se objeví několik vteřin po samotném spuštění hry po načtení zdrojů nezbytných k jeho zobrazení. Jeho cílem je co nejvíce zjednodušit nastavení hry a umožnit uživatelům se co nejrychleji ponořit do samotné hry.



Obrázek 3-1: Celkový pohled na aplikaci

Hlavní menu má několik záložek: Hra, Nastavení a Nápověda. V záložce Hra se nachází tlačítko Nová hra pro spuštění nového scénáře, který si uživatel z výsuvného menu vybere. Další dvě položky slouží pro uložení současné hry, respektive načtení hry uložené dříve. Poslední tlačítko hru ukončí.

V Nastavení si hráč vybere, kolik posledních tahů lodí se má vykreslovat (0 značí všechny). Položka Nápověda zobrazí informace o aplikaci.

Ve spodní části obrazovky se nachází Log, který slouží k textovému informování uživatele, vypisuje se zde například poškození lodí ze soubojů a srážek s asteroidy a podobně.

V horní části levého panelu se zobrazují informace o samotné hře, jako kolikáté je kolo, který hráč právě táhne nebo která fáze segmentu probíhá. Níže se nacházejí ovládací prvky týkající se vybraného políčka a případně objektů na něm. Ve spodní části levého panelu leží tlačítka pro ukončení segmentu, fáze a případné provedení útoku.

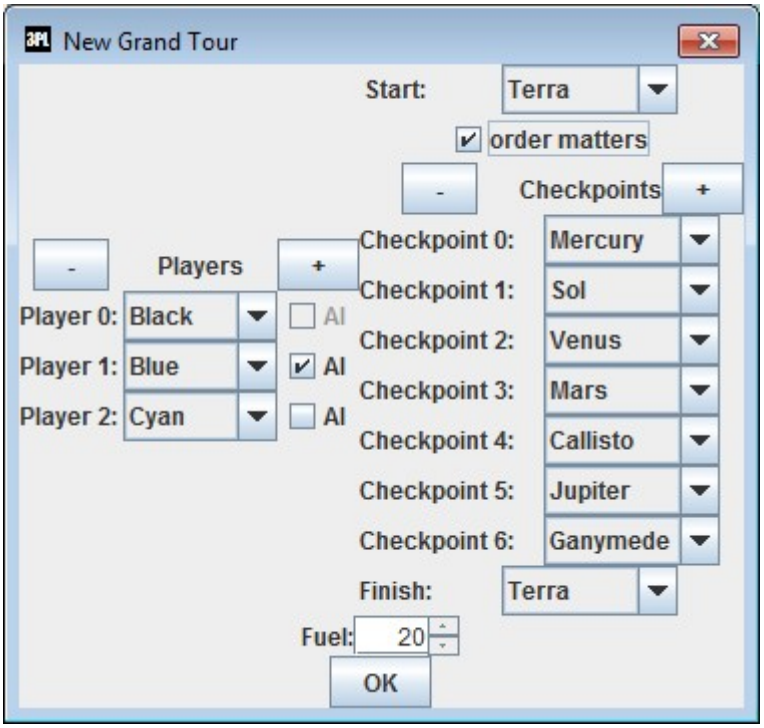
V horní části klientské části se nachází posuvník pro zvětšování a zmenšování políček hrací plochy.

Největší prostor na obrazovce zabírá samotná hrací deska složená ze šestiúhelníkových tlačítek.

Hráč si může snadno přizpůsobit rozdělení obrazovky ke svému obrazu kliknutím na příslušný oddělovač a potažením do požadovaného místa. Při ukončení bude toto uloženo a při příštím spuštění pozice oddělovačů opět obnovena.

3.1.1. Specifika uživatelského rozhraní scénáře Grand Tour

Na obrazovce nové hry scénáře Grand Tour tlačítko +, respektive –, v levé části dialogu přidává, respektive ubírá, hráče. V ComboBoxu si závodník může vybrat barvu, která bude reprezentovat dráhu jeho lodi, a pokud bude zaškrtnut CheckBox AI, pak za tohoto hráče bude hrát umělá inteligence. Poznámka: za prvního závodníka musí hrát člověk, stejně tak závod končí, jestliže budou všichni lidští hráči ze hry vyřazeni. Dále je možné určit množství paliva na závodních lodích.



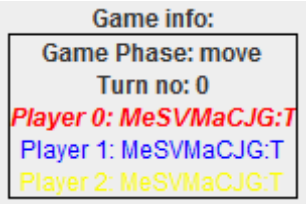
Obrázek 3-2: Obrazovka nové hry Grand Tour

V pravé části dialogu nové hry se nastavuje startovní planetoid, kontrolní body, kolem kterých musí závodníci proletět a cíl, kde po přistání prvního závodníka závod končí. Checkbox pod startem určuje, jestli se checkpointy musí navštívit v daném pořadí.

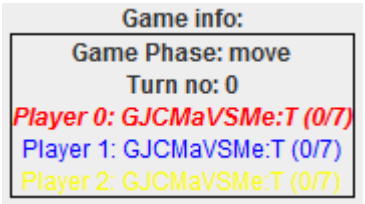
Během závodu se u každého hráče zobrazuje, které kontrolní body (planetární tělesa s plnou gravitací) musí ještě navštívit, než bude moci přistát v cíli (údaj za dvojtečkou). Při závodu, kde nezáleží na pořadí, se vypisuje i poměr dosažených/všech checkpointů. Každý kontrolní bod má svou zkratku:

Slunce	S
Merkur	Me
Venuše	V
Země (Terra)	T
Mars	Ma
Callisto	C
Jupiter	J
Ganymed	G

Tabulka 7: Zkratky kontrolních bodů



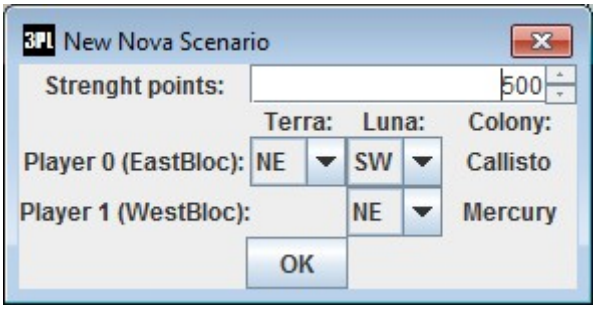
Obrázek 3-3: Herní informace když záleží na pořadí



Obrázek 3-4: Herní informace když nezáleží na pořadí

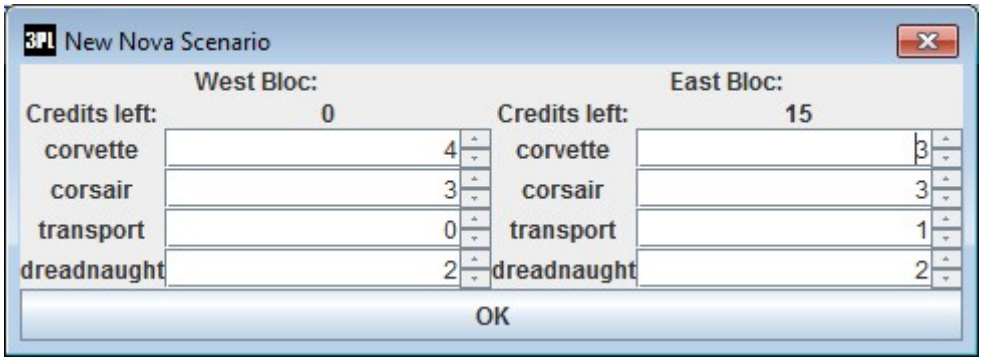
3.1.2. Specifika uživatelského rozhraní scénáře Nova

Na první obrazovce nové hry bojového scénáře Nova pro tři hráče se nachází SpinEdit s počtem bodů, za které si budou moci hráči za Západní a Východní blok nakoupit své lodě. Pod ním se nalézají ComboBoxy s nabídkou základen na Zemi (zde vybírá Východní blok, dostane zvolenou stranu a dvě další po směru hodinových ručiček, zbytek získá Západní blok) a Měsíci (kde má naopak přednost Západní blok). Vzdálené kolonie se náhodně rozdělí automaticky.



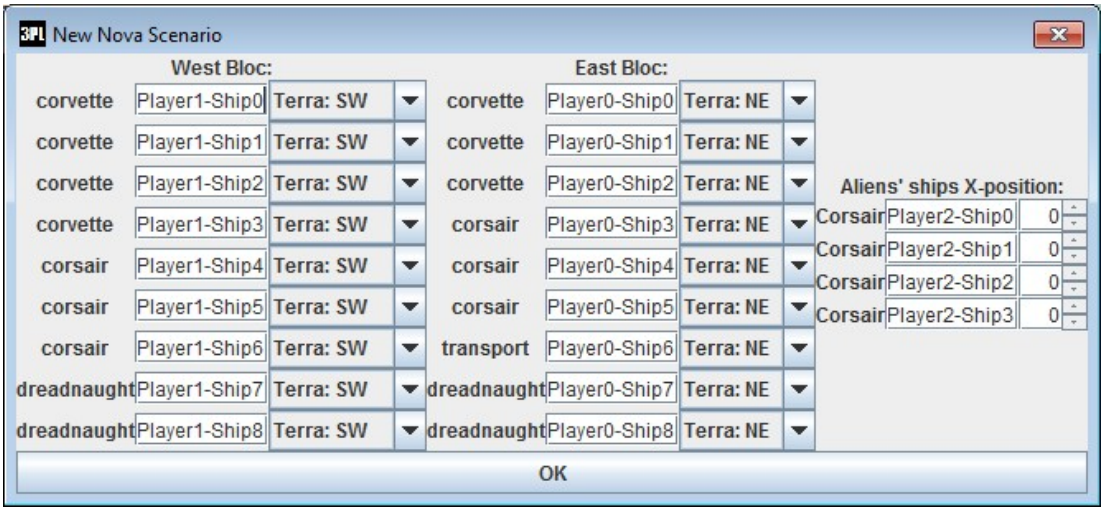
Obrázek 3-5: První obrazovka nové hry scénáře Nova

Na druhé obrazovce scénáře Nova si pozemští hráči za body, jejichž počet určil předchozí dialog, nakoupí jednotlivé lodě, respektive zadají, kolik lodí kterého typu si přejí nasadit proti svým protivníkům.



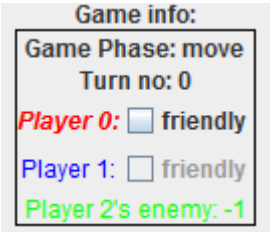
Obrázek 3-6: Druhá obrazovka nové hry scénáře Nova

Na třetí obrazovce se lodím nakoupeným v předchozím dialogu přiřadí jména a základny, kde začnou hru. Mimozemský hráč si zde také pojmenuje své čtyři útočné lodě a hlavně jim vybere startovní pozici na horním okraji mapy.



Obrázek 3-7: Třetí obrazovka nové hry scénáře Nova

Během samotného boje o osud Sluneční soustavy se v pravém horním rohu u každého pozemského hráče zobrazuje CheckBox vyjadřující jeho postoj k tomu druhému. U mimozemského hráče se zobrazuje, který jeho protivník vyhraje hru, pokud budou jeho lodě zničeny (-1 značí remízu).



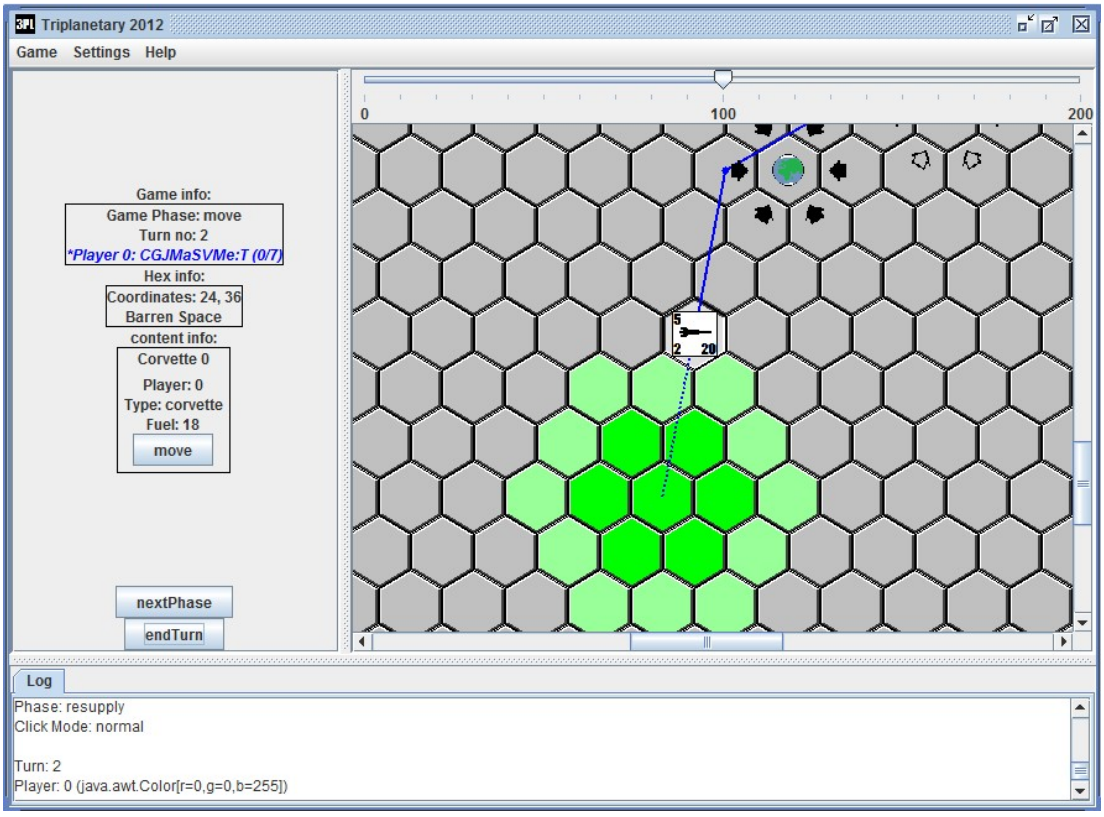
Obrázek 3-8: Informace o probíhající hře ve scénáři Nova

3.2. Ovládání

Ovládání je jednoduché a intuitivní a uživatel si vystačí s myší s jedním tlačítkem, i když druhé tlačítko může být v některých chvílích (vizte dále) použito pro urychlení a usnadnění ovládání snížením počtu potřebných kliknutí.

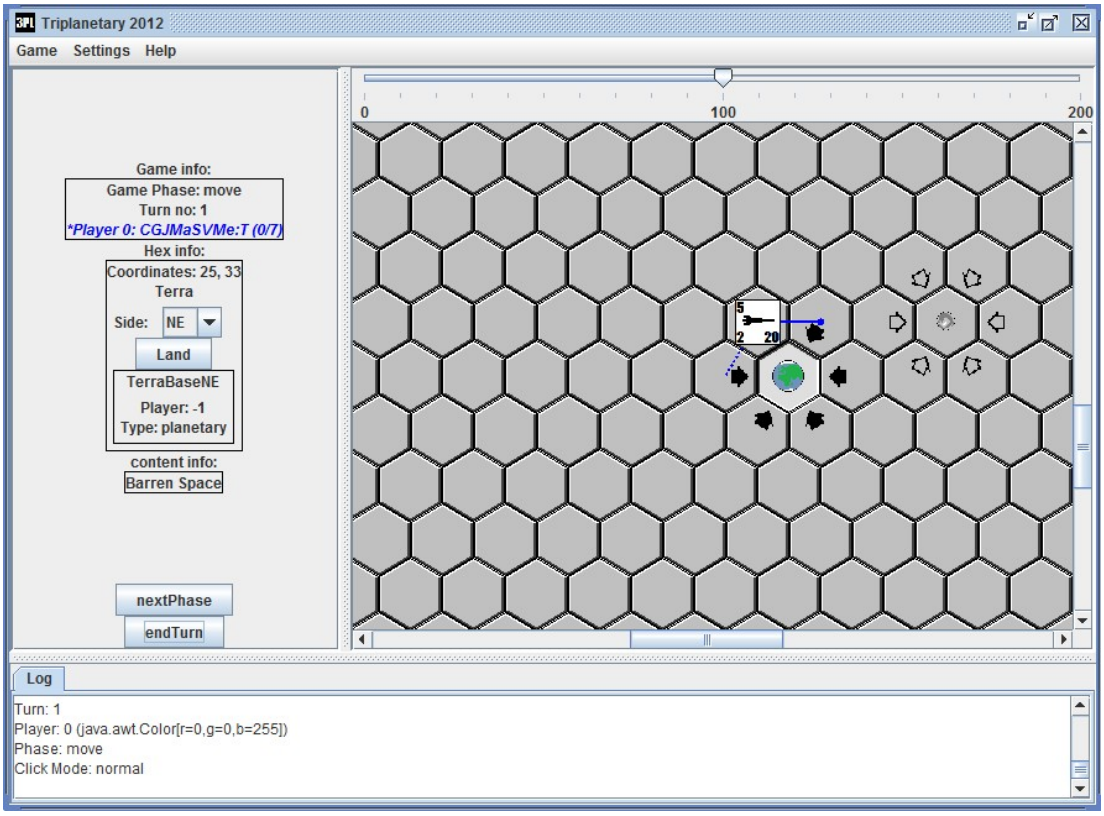
3.2.1. Pohybová fáze

Pro pohyb po kliknutí na políčko s požadovanou lodí (případně po jejím zvolení pomocí tlačítek se šipkami, pokud pole obsahuje více pohyblivých objektů) stačí stisknout tlačítko Pohyb a poté kliknout na zelené políčko reprezentující validní tahy (světle zvýrazněné znamenají použití přetížení motorů). Případně lze po vybrání lodi přímo kliknout na cílové políčko pravým tlačítkem myši.



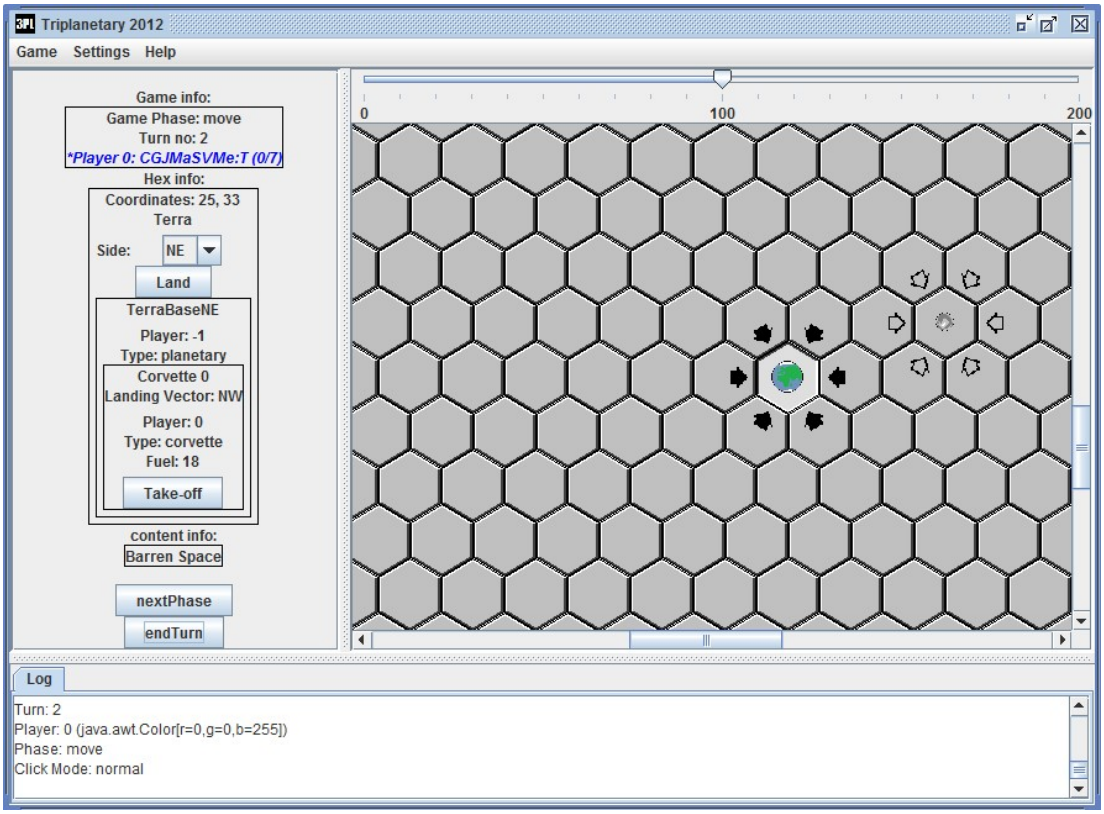
Obrázek 3-9: Nastavení kurzu

Pro přistání po kliknutí na políčko s cílovou planetou stačí vybrat z ComboBoxu, na kterou stranu planety má loď přistát, kliknout na tlačítko Přistát, potom na políčko s lodí na orbitě (případně zvolit správnou loď pomocí tlačítek se šipkami, pokud pole obsahuje více pohyblivých objektů) a nakonec opět tlačítko Přistát.



Obrázek 3-10: Přistávání

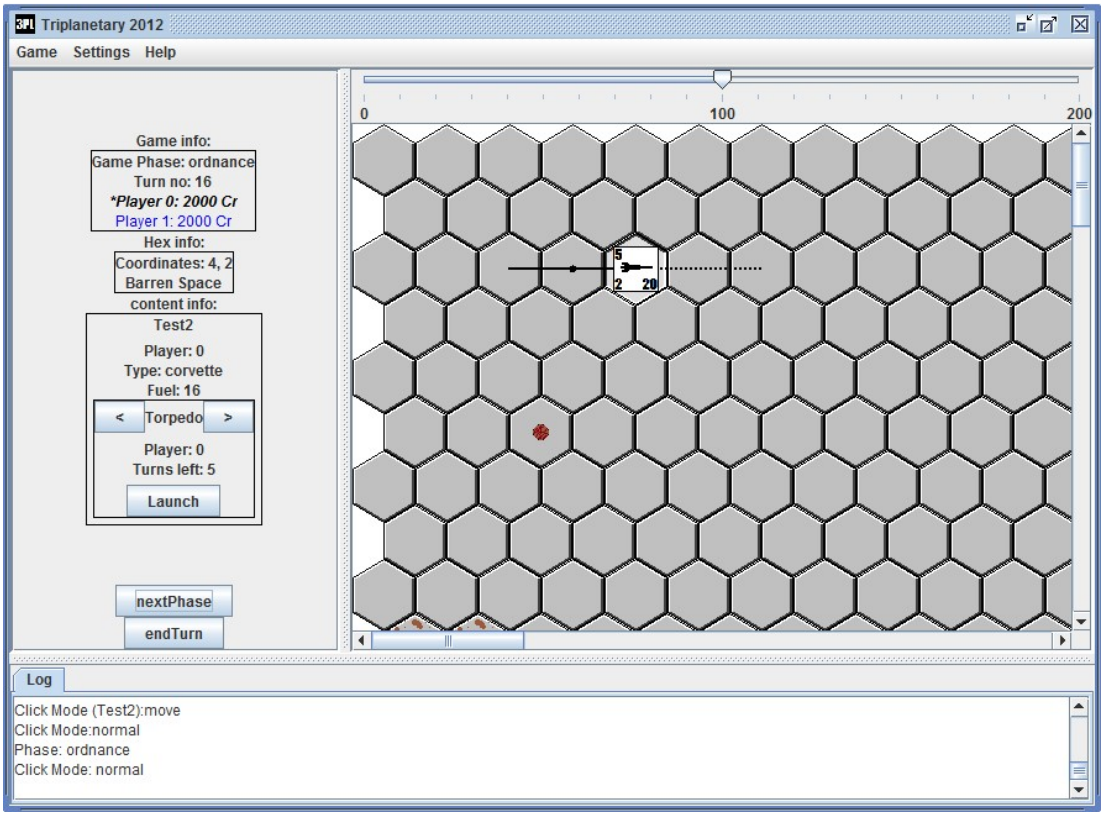
Pro vzlet po kliknutí na planetu a zvolení strany obsahující loď ji stačí pouze vybrat pomocí tlačítek se šipkami a stisknout tlačítko Take-off.



Obrázek 3-11: Vzlet

3.2.2. Fáze výzbroje

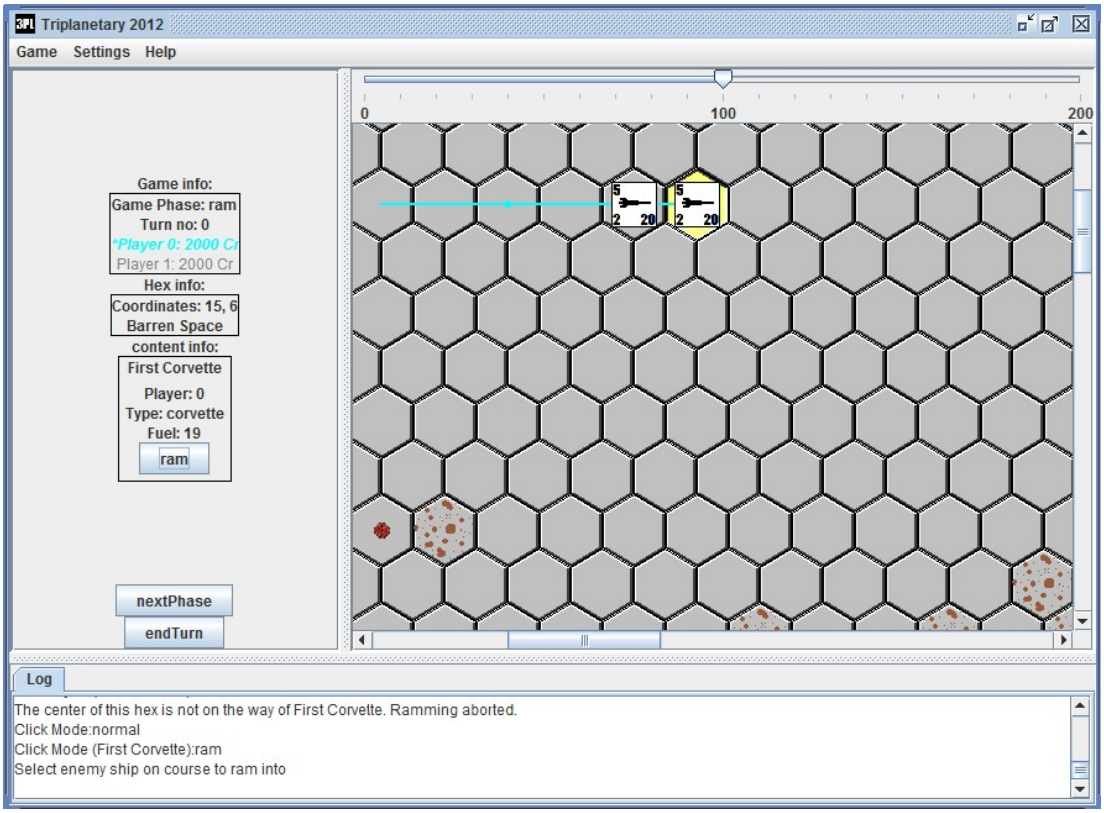
Po kliknutí na políčko s požadovanou lodí (případně po jejím zvolení pomocí tlačítek se šipkami, pokud pole obsahuje více pohyblivých objektů) stačí pomocí tlačítek se šipkami zvolit náklad (torpédo, minu nebo atomovou bombu) a vypustit ho pomocí akčního tlačítka. Po vypuštění torpéda lze ještě v této fázi navíc upravit jeho kurz jednorázovým zášlehem jeho motorů, což se provede stejně jako změna kurzu kosmických lodí.



Obrázek 3-12: Vypuštění torpéda

3.2.3. Taranovací fáze

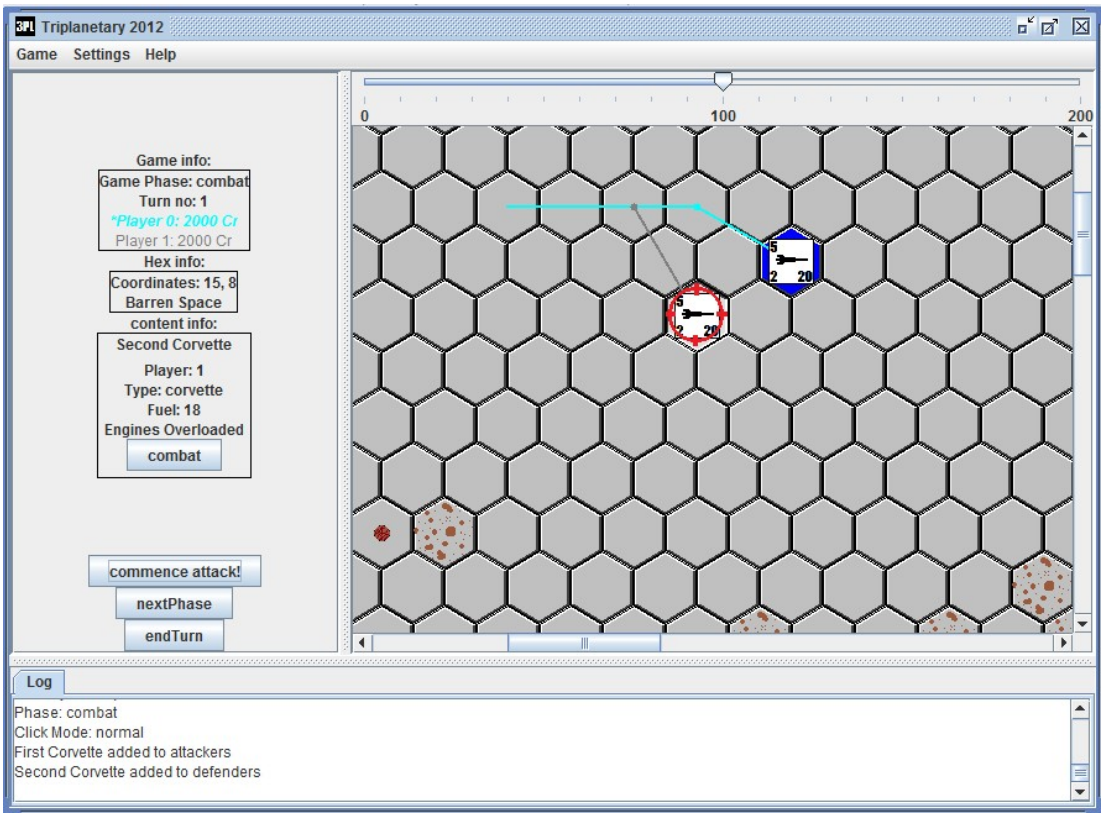
Po kliknutí na políčko s požadovanou lodí (případně po jejím zvolení pomocí tlačítek se šipkami, pokud pole obsahuje více objektů) stačí stisknout tlačítko Ram a poté kliknout na políčko, jehož střed loď proletěla a kde se nachází potencionální cíle. Nachází-li se na hexu více lodí, pomocí tlačítek se šipkami lze vybrat zamýšlený cíl a nakonec stisknout tlačítko Ram into pro dokončení taranu.



Obrázek 3-13: Taranování

3.2.4. Bojová fáze

Po kliknutí na políčko s požadovanou lodí (a jejím případným vybráním pomocí tlačítek se šipkami) stačí kliknout na tlačítko Boj. Pokud loď patří táhnoucímu hráči, pak se přidá mezi útočníky, případně se z nich odebere, v opačném případě se přidá mezi obránce (pokud je první nebo se nachází na políčku s nějakým obráncem), případně se z nich odebere. Políčko s obránci označuje zaměřovací kříž, který má červenou barvu, jestliže všechny pohyblivé objekty na políčku patří mezi cíle, jinak má barvu oranžovou. Políčko s útočníky má tmavě modré pozadí, pokud jeho všechny pohyblivé objekty patří mezi útočníky, jinak má pozadí světle modré.

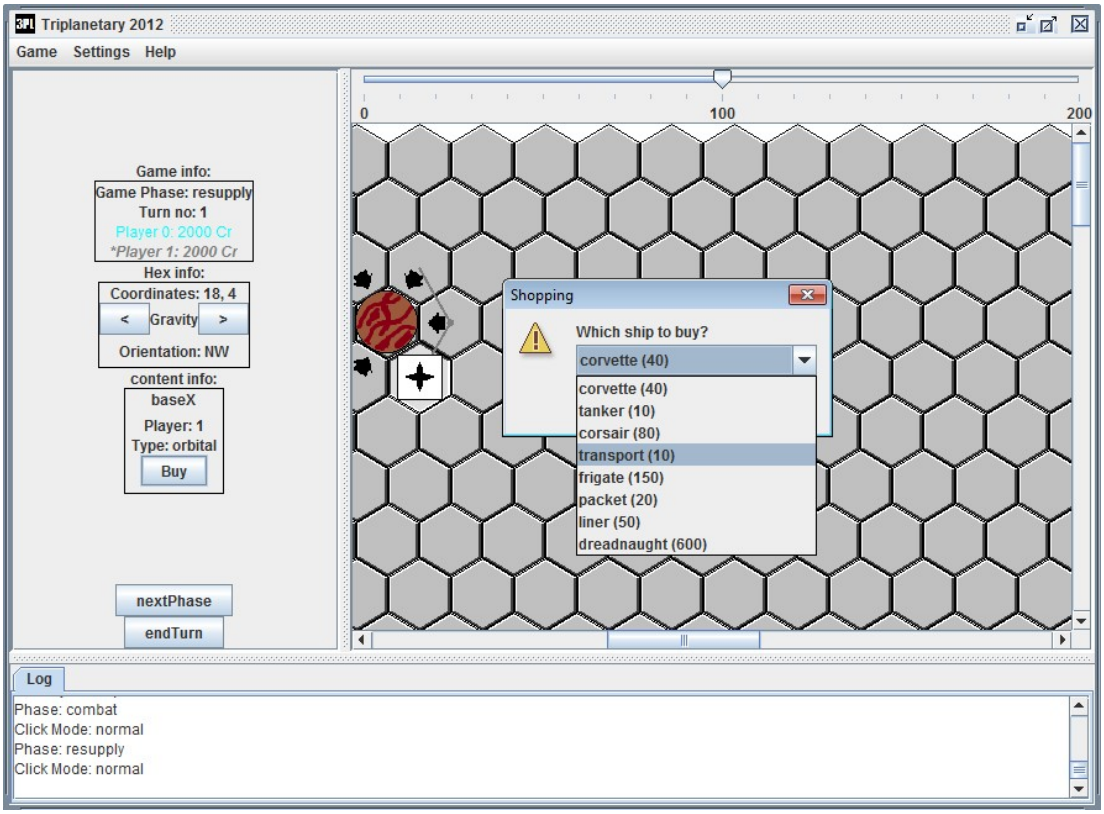


Obrázek 3-14: Boj

Nakonec kliknutí na tlačítko Provést útok zobrazí dialog s otázkou, jestli se obránci chtějí bránit protiútokem, jehož navolení probíhá stejně jako navolení původního útoku, jen novými obránci se mohou stát pouze původní útočníci a novými útočníky se mohou stát všechny lodě, které sdílí kurz s libovolným z původních obránců.

3.2.5. Zásobovací fáze

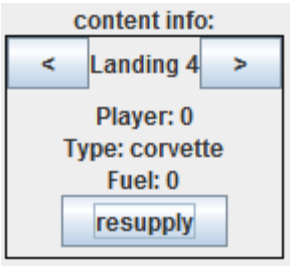
Pro nákup nových lodí stačí zvolit přátelskou základnu a kliknout na tlačítko Nákup. Objeví se dialog s ComboBoxem, kde lze navolit typ požadované lodi, a dialog s jednoduchým Editem pro zadání jména nové lodi. Poznámka: nákup lodí je dostupný pouze v některých scénářích!



Obrázek 3-15: Nákup lodi

Obrázek 3-16: Pojmenování nové lodi

Pro zásobování lodí stačí po kliknutí na její políčko (a případně jejím vybráním pomocí tlačítek se šipkami) stisknout tlačítko Zásobování a nakonec kliknout buď na to samé políčko, pokud obsahuje přátelskou orbitální základnu sdílející kurz se zásobovanou lodí, nebo na planetu s přátelskou základnou, nad kterou zásobovaná loď prolétá na orbitě. V některých scénářích se zobrazí také dialog, ve kterém si lze navolit, jakou novou výzbroj si hráč přeje na loď naložit.



Obrázek 3-17: Tankování lodi Landing 4

4. Implementace

Celá hra Triplanetary je psána v programovacím jazyku JAVA 6 a kromě standardních knihoven (jako jsou například AWT a Swing pro grafické uživatelské rozhraní) využívá externí knihovny org.w3c.dom pro práci s xml soubory a uk.co.keang.hex.gui[5], která je ve formě .jar souboru distribuována přímo s hrou a která nabízí šestiúhelníkový layout a tlačítka.

4.1. Nastavení a externí zdroje

Nastavení hry se nachází v souboru ./configuration.properties a je uloženo ve standardním javovském properties formátu: jmeno.property=hodnota_property na jednotlivých řádcích, # značí komentář do konce řádku. Právě do tohoto souboru se uloží nastavení při ukončování hry a načítá se z něj při spuštění hry.

```
#Sat Jun 23 21:05:34 CEST 2012
frame.height=654#výška okna se hrou
horizontalDivider.location=266#pozice horizontálního oddělovače
turnsBack=0#kolik tahů zpět se vykreslují pozice lodí
frame.Y=64#y-ová pozice okna se hrou
frame.width=889#šířka okna se hrou
frame.X=308#x-ová pozice okna se hrou
verticalDivider.location=433#pozice vertikálního oddělovače
locale.language=en#lokalizace
hex.height=86#defaultní výška hexu
```

Tabulka 8: Příklad konfiguračního souboru

V adresáři ./data se nacházejí externí zdroje. Patří mezi ně grafické soubory obrázků jednotlivých vesmírných objektů ve formátu png, xml soubory map a soubor ShipTypes.csv s definicemi typů vesmírných lodí. Grafické zdroje a soubor ShipTypes.csv se většinou načítají až ve statických konstruktorech tříd, které je využívají, takže se nestane, že by zabíraly místo dříve, než je hra potřebuje. Soubory map se čtou až při vytváření nové hry, která si je vyžádala, a zpracovávají se DOM parserem, který sice není tak efektivní jako SAX parser, ale nabízí pohodlnější práci a pro nepříliš rozsáhlé a komplikované mapy Triplanetary je plně dostačující. Všechny objekty na mapě od lodí přes miny po planetoidy v konstruktoru požadují objekt třídy org.w3c.dom.Element, který byl vytvořen parserem podle reprezentace v XML souboru mapy. Příklad mapy naleznete v příloze (devátá kapitola).

```
name;combat;warship;fuel;cargo;image;damagefire;carrybases;cost
corvette;2;true;20;5;data/corvette.png;false;false;40
```

Tabulka 9: Příklad ShipTypes.csv

V souboru ./classes/localization_en.properties se nachází popisky některých ovládacích prvků.

Uložené hry jsou defaultně uchovávány v souborech v adresáři ./saves s příponou .sav, ale hráč si samozřejmě může zvolit, kam se rozehraná hra uloží. Ukládání samotné funguje na principu standardní serializace jazyka JAVA, kdy se do souboru binárně serializuje objekt reprezentující probíhající hru, který je instancí třídy dědící od třídy triplanetary.Game (vizte 4.2.1 Balíček triplanetary). Binární serializace jsem zvolil, protože editace uložených her není tak snadná a brání se tak podvádění (i když samozřejmě by šlo ukládat například hash souboru a upravené hry odmítnout načíst).

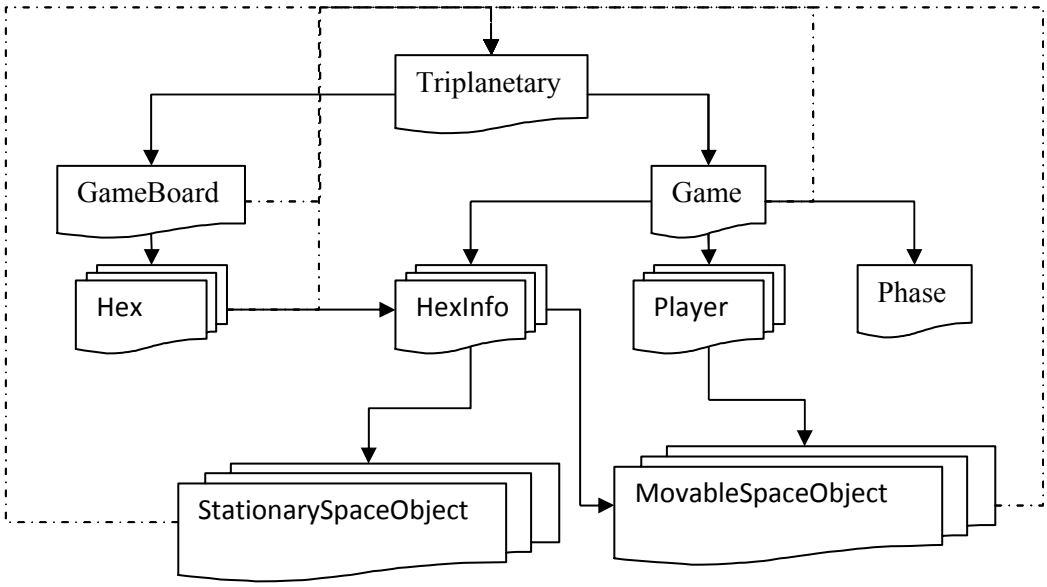
Neméně důležitý je i fakt, že při správném návrhu aplikace její implementace nestojí skoro žádnou námahu navíc. Posledním důvodem je menší velikost souborů než u textového ukládání, která se by se dala ještě dále snížit vynecháním nepotřebných informací, mezi něž patří například prázdná políčka.

4.2. Balíčky a třídy

Celý projekt se skládá z pěti hlavních balíčků: triplanetary, triplanetary.auxiliary, triplanetary.objects, triplanetary.scenarios.nova a triplanetary.scenarios.grandtour. Tato sekce postupně popíše všechny balíčky a třídy v nich obsažené a nakonec osvětlí i jejich vzájemné vztahy.

4.2.1. Balíček triplanetary

Balíček triplanetary obsahuje základní třídy, které se využívají v celém programu.



Obrázek 4-1: Základní vztahy mezi třídami (není vyčerpávající)

Třída Triplanetary je nejdůležitější třídou celé hry, protože právě ona ji zastřešuje a stará se o vykreslování grafického uživatelského rozhraní a o většinu herní logiky spojené s ovládacími prvky. Na objekt této třídy, který řídí celou hru, si většina herních objektů udržuje referenci, jelikož právě přes ni mohou ovlivňovat všechny aspekty hry. Z tohoto důvodu jsou téměř všechny položky veřejné, aby se případným rozšířením a novým scénářům vyšlo vstříc a umožnilo jim prakticky libovolně ovlivňovat a přizpůsobovat si pravidla a herní logiku. Pro zobrazení uživatelského rozhraní si vyžádá od jednotlivých momentálně označených objektů (například scénáře, zvoleného políčka hrací plochy a tak podobně) swingové komponenty, které co nejlépe odrážejí současný stav objektu.

Třída Triplanetary také obsahuje několik vnitřních tříd sloužících jako listenery na události spojené s ovládacími prvky a výčtový typ ClickMode, který určuje režim kliknutí na šestiúhelníková políčka herní plochy.

Třída Game reprezentuje rozehranou hru a je předkem jednotlivých scénářů. Obsahuje herní logiku, která v některých scénářích podléhá změnám pravidel. Právě ona udržuje stav hry (od počtu hráčů přes současnou fázi až po číslo hráče na tahu) a případně se serializuje při ukládání a deserializuje při načítání. Je také zodpovědná za parsování mapy při tvorbě nové hry.

Třída Game obsahuje také vnitřní třídy, z nichž jsou nejdůležitější Phase a Player. Výčtový typ Phase reprezentuje jednotlivé fáze hráčského segmentu. Třída Player reprezentuje jednotlivé hráče a udržuje vitální informace, jako například hráčovu barvu, seznam jeho lodí ve volném vesmíru a příznak, jestli je ovládán umělou inteligencí.

Třída HexInfo reprezentuje jedno logické šestiúhelníkové políčko herní plochy. Udržuje si seznam MovableSpaceObjectů jako svůj obsah a seznam StationarySpaceObjectů jako svoje pozadí. Každý objekt této třídy má několik druhů souřadnic. Kromě ArrayCoordinates a HexCoordinates, o kterých bude řeč níže, nabízí i kartézské souřadnice svých vrcholů a středu v imaginární herní ploše, kde každý hex je pravidelný šestiúhelník se stranou délky 1, což se velmi hodí pro výpočty, například týkající se průletu lodí skrz hexy.

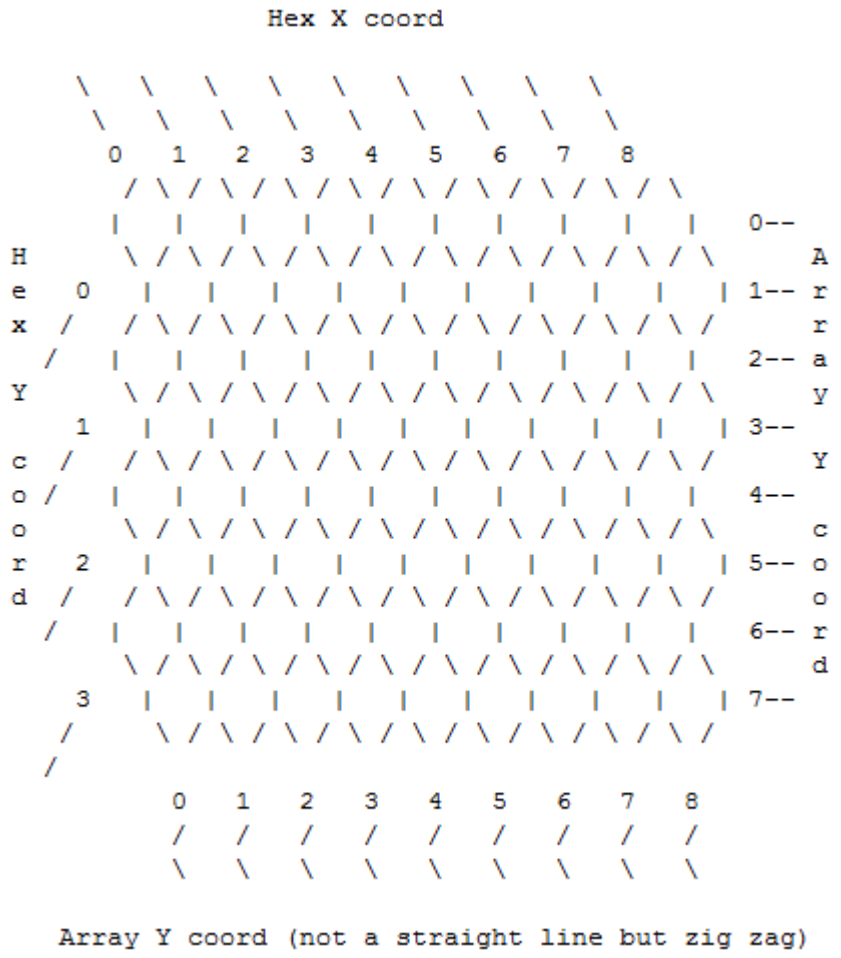
Třída Hex dědí od uk.co.keang.hex.gui.HexToggleButton[5], je to tedy vizuální komponenta reprezentující políčko herní plochy chovající se jako tlačítko. Udržuje si referenci na odpovídající objekt třídy HexInfo a vykresluje na sebe jeho pozadí a obsah. Stará se také o zobrazování informací o právě zvoleném hexu.

Třída GameBoard dědí od JPanelu, je to tedy vizuální komponenta reprezentující hrací plochu a obsahující pole objektů třídy Hex. Počítá s tím, že bude používat šestiúhelníkový layout uk.co.keang.hex.gui.HexLayout[5] a že se do ní budou ukládat pouze objekty Hex. Stará se také o vykreslování čar reprezentujících trajektorii pohybujících se objektů.

Třída Utils je pouze úložiště statických metod používaných v různých částech programu. Například funkce pro výpočet vzdálenosti hexů, seznamu hexů nalézajících se na spojnici středů dvou hexů, funkce pro zjišťování, zda se úsečka protíná s kružnicí a podobně.

Práce se souřadnicovým systémem pro šestiúhelníčkové hrací plochy je komplikovaná. Proto jsem se inspiroval velice sofistikovaných článkem [6], který navrhuje dvě soustavy souřadnic (jednu pro umístění šestiúhelníků do pravoúhlého pole a druhou pro přirozenější použití v hexovém světě) a nabízí efektivní algoritmy pro práci v nich. Toto se projevilo v celém programu, nejočividnější to však je na třídách ArrayCoordinates a HexCoordinates, které reprezentují pozici šestiúhelníku

v pravoúhlé soustavě souřadnic pro pole, respektive pozici v šestiúhelníkové soustavě souřadnic. Obě třídy jsou velmi jednoduché a velmi podobné – jejich hlavním obsahem jsou vlastně jen dvě položky pro uchovávání x-ové a y-ové souřadnice, na což by stačila třída java.awt.Point. Bylo však výhodnější je od sebe oddělit, aby se snížil počet programátorských chyb v programu.



Obrázek 4-2: Hexové a pravoúhlé souřadnice[6]

Další jednoduchá třída DoublePoint slouží pro udržování x-ových a y-ových souřadnic typu Double. Využívá se například ve třídě HexInfo pro souřadnice středu a vrcholů na jednotkové hrací ploše.

Generická třída Pair je inspirovaná třídou std::pair jazyka C++ a slouží ke sdružení dvou objektů do dvojice. Využívá se například ve třídě Planetoid pro zpárování informace o směru přistání na planetě (výčtový typ Orientation, viz dále) a reference na přistanuvší objekt.

Výčtový typ Orientation reprezentuje šest směrů hexové hrací plochy. Využívá se například pro určování směru gravitace ve třídě Gravity a pro uchovávání informace o směru přistání pohyblivých objektů na planetách ve třídě Planetoid.

Třída ScenarioLoader je jednoduchý potomek třídy java.io.ObjectInputStream, který má za úkol deserializovat objekty. Od obyčejného ObjectInputStreamu se liší

tím, že pokud nebude třída objektu nalezena na classpath, podívá se ještě do adresáře ./scenarios, kde se uchovávají třídy potřebné pro jednotlivé scénáře.

4.2.2. Balíček triplanetary.auxilliary

V balíčku triplanetary.auxilliary se nachází abstraktní třídy a rozhraní, které reprezentují vesmírné objekty vyskytující se na hrací ploše. Konkrétní třídy se potom nacházejí v balíčku triplanetary.objects (vizte 4.2.3).

Třída SpaceObject je vrcholem dědičnosti vesmírných objektů. Nabízí základní metody společné pro všechny potomky, jako je například nastavování a zjišťování polohy na hrací ploše. Za pozornost stojí určitě konstruktor požadující objekt třídy Triplanetary (z důvodu snazšího ovlivňování grafického uživatelského rozhraní) a objekt třídy org.w3c.dom.Element získaný DOM parserem z xml souboru s mapou a také metoda restore, která nastavuje referenci na objekt třídy Triplanetary, který řídí hru a která se musí zavolat po deserializaci.

Abstraktní třída StationarySpaceObject reprezentuje objekty, které se po hrací ploše nepohybují a tvoří tak pozadí hexů. Mezi potomky této třídy patří například třídy Planetoid, Asteroids a Gravity.

Abstraktní třída MovableSpaceObject reprezentuje objekty, které se po hrací ploše pohybují a tvoří tak obsah hexů. Mezi povinnosti této třídy patří hlavně uchovávání předchozích pohybů, aby bylo možné je vykreslovat na herní ploše. Od této třídy dědí například třídy Ship a Base.

Rozhraní IRound zajišťuje, že se třídy, které ho implementují, dají vykreslit jako kruhové objekty. Rozhraní je určeno pouze pro potomky třídy SpaceObject, ale počítá se s ním hlavně pro potomky třídy StationarySpaceObject. Mezi třídy, které ho implementují, patří například třídy Planetoid a Asteroids. Pole asteroidů sice nespadají z definice mezi kruhové objekty, ale na rozdíl od žetonů mohou při poloměru 1 zabrat celý obsah hexu, což je pro asteroidy vizuálně vhodnější.

Rozhraní IToken označuje třídy, které se dají a mají vykreslit jako žetony. Rozhraní je určeno pouze pro potomky třídy SpaceObject, ale počítá se s ním hlavně pro potomky třídy abstraktní MovableSpaceObject. Mezi třídy, které ho implementují, patří například třídy Ship, Mine, Base, ale i Gravity.

Rozhraní IMovableSpaceObject slouží jako společný předek pro rozhraní určená pro potomky třídy MovableSpaceObject. Obsahuje všechny veřejné metody MovableSpaceObjectu a SpaceObjectu tak, aby je už jednotlivá rozhraní nemusela sama vyjmenovávat.

Abstraktní třída Collider je společným předkem tříd Mine, Torpedo a Nuke s velmi podobným chováním.

Další rozhraní tohoto balíčku popisuje Tabulka 10.

Jméno	Popis	Implementováno v
ICombatant	Pro zapojení do vesmírného boje pomocí palubních zbraní jako útočník nebo obránce	Ship a Nuke
ILandTO	Pro přistávání na planetách a vzlet z nich	Ship
IRammable	Pro taranování ať už jako útočník nebo obránce	Ship, přes Collider Mine, Torpedo a Nuke
ITransportable	Pro třídy přepravované v jiných objektech (většinou lodích)	Base, přes Collider Mine, Torpedo a Nuke
ICollider	Pro třídy nebezpečné pro objekty, které přelétávají přes jejich hex nebo do kterých narazí	přes Collider Mine, Torpedo, Nuke

Tabulka 10: Rozhraní pro MovableSpaceObjecty, potomci IMovableSpaceObjectu

4.2.3. Balíček triplanetary.objects

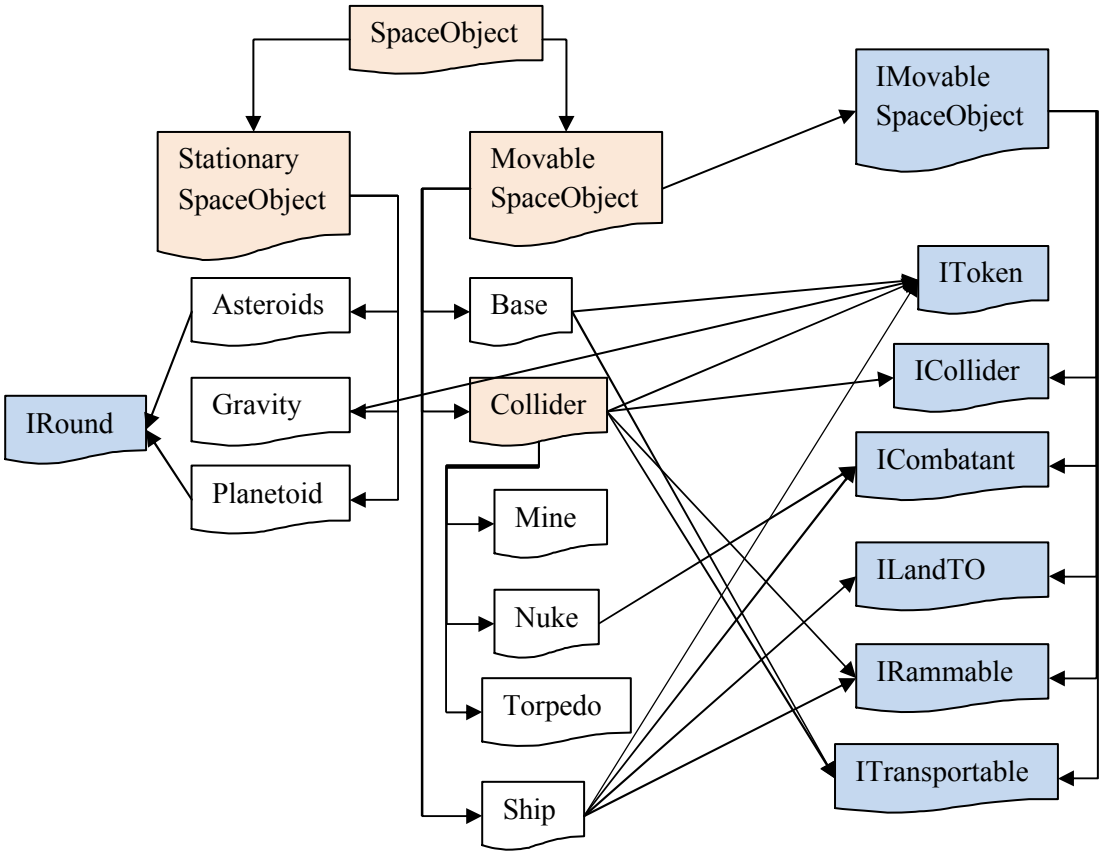
Balíček triplanetary.objects obsahuje konkrétní třídy, které reprezentují vesmírné objekty vyskytující se na hrací ploše a které dědí od tříd a implementují rozhraní z balíčku triplanetary.auxilliary. Jejich přehled se nachází v Tabulkách 11 a 12.

Jméno	Popis	Impl. rozhraní
Asteroids	Pole asteroidů, které může poškodit lodě, jež jím prolétávají příliš rychle	IRound
Gravity	Gravitační (ať už plný nebo jen slabý) efekt na hexu	IToken
Planetoid	Planety, měsíce a hvězdy	IRound

Tabulka 11: Potomci třídy StationarySpaceObject

Jméno	Popis	Impl. rozhraní
Base	Orbitální a planetární základny, může být přepravována loďmi typu Transport	IToken, ITransportable
Mine	Miny vybuchující při kontaktu s vesmírnými loděmi.	IToken, IRammable, ICollider, ITransportable (přes abstraktního předka Collider)
Nuke	Atomové bomby ničící vesmírné lodě, čistící pole asteroidů a ničící povrchy planet včetně základen	IToken, IRammable, ICollider, ITransportable (přes abstraktního předka Collider), ICombatant
Ship	Všechny vesmírné lodě, přiřazen v konstruktoru jeden z typů načítaných ve statickém konstruktoru ze souboru ./data/ShipTypes.csv (vizte 4.1 Nastavení a externí zdroje)	IToken, ICombatant, IRammable, ILandTO
Torpedo	Torpéda vybuchující při kontaktu s vesmírnými loděmi	IToken, IRammable, ICollider, ITransportable (přes abstraktního předka Collider)

Tabulka 12: Potomci třídy MovableSpaceObject



Obrázek 4-3: Přehled tříd, abstraktních tříd a rozhraní pro objekty na hrací ploše

4.2.4. Balíček triplanetary.scenarios.grandtour

Balíček triplanetary.scenarios.grandtour obsahuje třídy nezbytné pro správný průběh scénáře Grand Tour.

Třída GrandTour jako potomek třídy Game reprezentuje hru scénáře Grand Tour. Od svého předka se liší především implementací umělé inteligence (vizte 4.3.1) a dialogem při startu hry, kdy si hráči mohou nastavit kontrolní body, svoje barvy a podobně (vizte 3.1.1). A právě s těmito rozdíly souvisí i její vnitřní třídy ShipStatus (popisující stav lodi ovládané umělou inteligencí) a NewGrandTourDialog (sloužící hráčům k nastavení parametrů scénáře). Další vnitřní třída GTPlayer dědící od Game.Player má za úkol uchovávat i další důležité informace o hráčích, jako jsou například kontrolní body, kterými již hráčova korveta proletěla, množství použitého paliva a příznak, zda již hráč dokončil závod.

Třída GTTarget dědící od StationarySpaceObject slouží pouze k označení kontrolních bodů při načítání mapy scénáře Grand Tour.

Třída GTShip jen mírně rozšiřuje svého předka Ship o možnost vyvést jinak protected statické položky se seznamy typů lodí a lodí přistanuvších na planetách a o počítání spáleného paliva.

Třída GTBase naopak spíše omezuje možnosti svého předka Base, protože ve scénáři Grand Tour nelze dokupovat zbraně a nové lodě, ale na druhou stranu je palivo zdarma. Také zpřístupňuje jinak protected statickou položku se seznamem planetárních základen.

4.2.5. Balíček triplanetary.nova

Balíček triplanetary.nova obsahuje třídy nezbytné pro správný průběh scénáře Nova.

Třída NShip dědící od Ship musela být rozšířena hlavně z důvodu odlišného chování mimozemských lodí: nemohou tankovat a musí si pamatovat, kdo je poslední poškodil, aby se při kompletním zničení mimozemské flotily dalo určit, která z lidských frakcí vyhrála.

Třída NBase se od svého předka Base liší tím, že nenabízí možnost kupovat nové lodě, ale na druhou stranu je zase doplňování paliva a výzbroje zdarma. Také se musí umět vyrovnat se situací, kdy se její vlastník chová ke druhé pozemské frakci přátelsky, a tedy planetární obrana po jejích lodích nestřílí, ale dokonce se jim umožňuje zde tankovat.

Třída NPlanetoid jen mírně rozšiřuje svého předka Planetoid, a to pouze o snazší přístup k seznamu všech planetoidů, aby na nich mohly být snáze rozmístěny lidské základny.

Třída NMine jen lehce upravuje svého předka Mine, a to o detekci poškození mimozemských lodí minami lidských frakcí.

4.3. Zajímavé algoritmy

V této sekci si popíšeme nejzajímavější algoritmy, které se v projektu používají.

4.3.1. Umělá inteligence ve scénáři Grand Tour

Veškerá umělá inteligence se soustřeďuje do následujících metod třídy GrandTour:

- void AIland(GTPlayer player, Ship ship)
- ShipStatus[] AImove(GTPlayer player, Ship ship)
- void AIram(GTPlayer player, Ship ship, ShipStatus[] path)
- void AIresupply(GTPlayer player, Ship ship)
- void calculateShipStatusValue(int forwardView, ShipStatus status, GTPlayer player)
- void counterAttackAI()
- void counterAttackOnlyAI()
- void endPhase()
- JButton getSelectButton(Container c)
- boolean isSafe(ShipStatus status)
- void newGame()

V této sekci si jednotlivé metody popíšeme a přiblížíme si, jak se vlastně umělá inteligence chová a jak je implementována. Základním principem je, že AI by mělo používat stejné ovládací prvky jako hráč, aby se zaručilo, že nepodvádí. Z tohoto důvodu není možné nechat hrát pouze AI, protože by hrozilo přetečení zásobníku volání, jelikož metody řídící AI volají procedury spojené s ovládacími prvky a ty zase funkce umělé inteligence. Tento cyklus se přeruší až ve chvíli, kdy na řadu přijde člověk. Závodu se tedy musí vždy zúčastnit alespoň jeden lidský hráč a po vyřazení posledního nepočítačového závodníka hra končí.

Umělá inteligence hraje poměrně defenzivně, to znamená, že se vždy zapojuje do protiútoků, dále že sama od sebe nepálí z palubních zbraní, protože se bojí případného protiútoku, a že taranuje, pouze pokud by stejně další dvě kola nechtěla měnit kurz, takže případné poškození jí nevadí.

V metodě newGame se kromě jiného hrubou silou řeší problém obchodního cestujícího, jestliže pravidla závodu netrvají na přesném pořadí dosažení kontrolních bodů. Umělá inteligence se v tomto případě chová úplně stejně jako při závodě, kde záleží na pořadí prolétání kolem jednotlivých checkpointů, proto se musí před samotnou hrou kontrolní body optimálně seřadit. Za kritérium se bere pouze celková uražená vzdálenost, neuvažuje se tedy například náročnost na spotřebu paliva.

Metody counterAttackAI() a counterAttackOnlyAI() slouží k provádění protiútoků za účasti umělé inteligence. Z důvodu jejich případného zbytečného zkomplikování tyto poměrně jednoduché metody porušují základní princip AI ve hře Triplanetary tím, že přímo volají metody třídy Triplanetary provádějící vyhodnocení útoku a nepoužívají tedy stejné prostředky jako hráči.

Metoda counterAttackAI() přidává do protiútoku, kterého se účastní i lodě lidských hráčů (to znamená, že už jsou zvoleny cíle protiútoku), lodě ovládané počítačovými hráči, a to všechny, které se ho mohou zúčastnit.

Metoda counterAttackOnlyAI() obstarává protiútok, kterého se účastní pouze lodě ovládané umělou inteligencí. Za cíl protiútoku se určí všechny útočící lodě (z důvodu omezení daných scénářem to může být pouze jedna loď) a poté se volá counterAttackAI().

V metodě endPhase() se mimo jiné volají i zbylé metody spojené s chováním umělé inteligence, nastal-li segment počítačem ovládaného hráče:

Funkce ShipStatus[] AImove(GTPlayer player, Ship ship) si nejprve vygeneruje všechny dovolené kurzy v následujících pěti tazích (takzvané okénko) ignorujíc případy, kdy by byla loď zničena (na to používá funkci boolean isSafe(ShipStatus status)). Potom tyto možné kurzy ohodnotí pomocí metody calculateShipStatusValue(ShipStatus status, GTPlayer player) (vizte dále) a z těch nejlepších jednu náhodně vybere. Nakonec změní kurz lodi a vrátí trajektorii, kterou AI v tomto kole sleduje.

Metoda calculateShipStatusValue(int forwardView, ShipStatus status, GTPlayer player) ohodnocuje status lodí podle následujících kritérií:

- zbývající palivo (navíc postih, pokud je aktuálního paliva méně jak čtyřicet pět procent maximální kapacity)
- přetížené motory
- dosažení kontrolních bodů (+ bonus pokud se to nestalo v posledním tahu okénka)
- prolétávání poli asteroidů
- vzdálenost k dalšímu kontrolnímu bodu, případně už cíli
- orbita kolem cíle, pokud jsou všechny ostatní kontrolní body dosaženy
- orbita kolem planetoidu se základnou, pokud je aktuálního paliva méně jak čtyřicet procent maximální kapacity
- rychlost (větší jak čtyři už totiž koliduje s velikostí okénka, takže se AI snaží „nepřepálit“)

Metoda AIland(GTPlayer player, Ship ship) zajišťuje přistání, jestliže loď již obletěla všechny kontrolní body a nalézá se na orbitě cíle.

Metoda AIram(GTPlayer player, Ship ship, ShipStatus[] path) se stará o provedení taranu, pokud se na trajektorii lodi ovládané umělou inteligencí nachází vhodný cíl a AI nemá v úmyslu příští dvě kola měnit kurz, čili případné poškození nijak neohrozí šanci počítačového hráče na vítězství.

Metoda AIresupply(GTPlayer player, Ship ship) načerpá do lodi palivo, pokud se ta nachází na orbitě planetoidu a přelétává nad vhodnou základnou.

4.3.2. Určování hexů proťatých úsečkou určenou středy dvou hexů

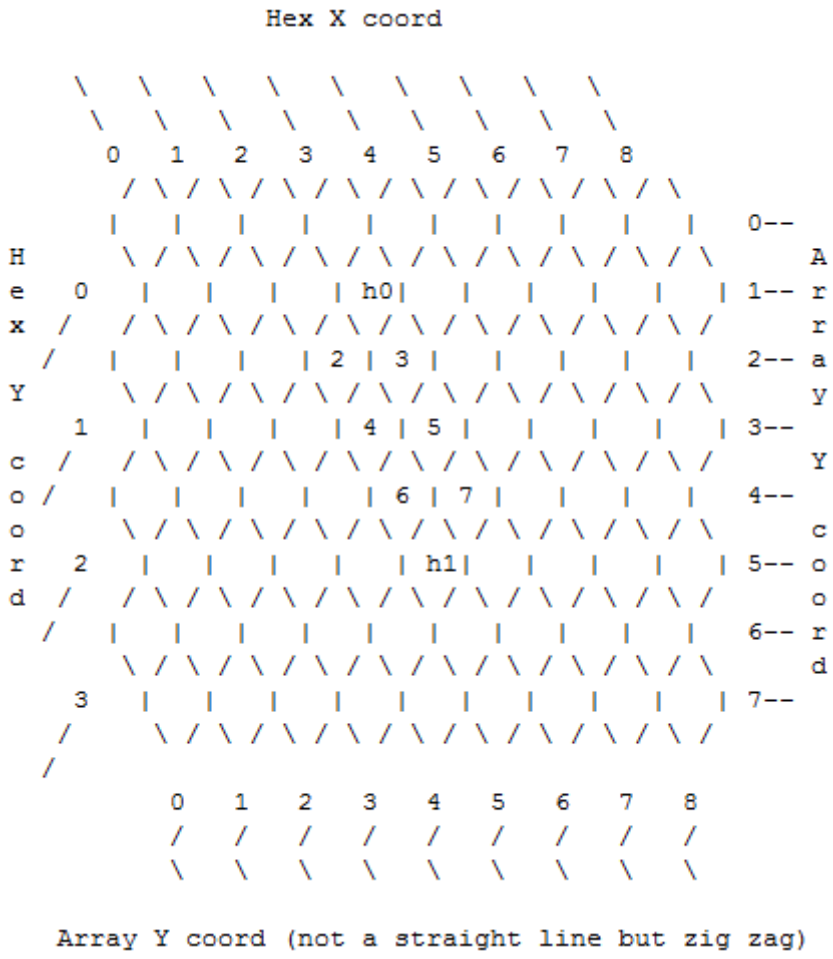
static public void hexesBetween(HexInfo[][] hexes, HexInfo h0, HexInfo h1, List<HexInfo> outputList)

Algoritmus[6] implementovaný touto statickou metodou třídy Utils je jedním z těch nejzákladnějších a nejpoužívanějších v celém projektu, od určování dalších možných pozic pohyblivých objektů ve statické funkci calculateNextPosition třídy Utils, přes výpočty modifikátorů pro boj v metodě attack a určování možných cílů taranování v metodě actionRam ve třídě Triplanetary, až po kontrolu průletu checkpointů a určování bezpečnosti kurzu lodí řízených umělou inteligencí ve třídě GrandTour.

Parametr hexes představuje dvourozměrné pole herní plochy, parametr h0, resp. h1, je výchozí, resp. cílový, šestiúhelník zkoumané úsečky. Do jediného výstupního parametru outputList procedura uloží hexy ležící na orientované spojnici středů šestiúhelníků h0 a h1. Parametr outputList nesmí být null!

Algoritmus nejprve omezí prohledávanou oblast hexovými souřadnicemi výchozího a cílového šestiúhelníku. Poté se postupně pro ty hexy tohoto „kosodélníku“, které

ještě leží na hrací ploše, pomocí statické metody hexIntersectsLine (vizte dále) třídy Utils určí, zda je zadaná úsečka protíná, a pokud ano, tak se přidají do výstupního seznamu outputList.



Obrázek 4-4: Postup procházení hexů: h0, 2, ..., 7, h1

static public boolean hexIntersectsLine(HexInfo h, double x0, double y0, double x1, double y1)

Algoritmus implementovaný touto funkcí tvoří základ velmi důležité a používané metody hexesBetween. Funkce vrací, zda přímka protíná daný hex. Jako parametry potřebuje souřadnice dvou bodů ležících na dané přímce a samotný hex. Šestiúhelník leží na přímce právě tehdy, když přímka prochází alespoň jedním jeho vrcholem, nebo když alespoň jedna dvojice jeho vrcholů leží na opačné straně dané přímky. A právě tohoto tvrzení využívá metoda hexIntersectsLine, která postupně bere jeden vrchol šestiúhelníku za druhým, a jestliže jím přímka prochází, nebo se již předtím našel vrchol ležící na opačné straně, vrátí hodnotu true. Jestliže se žádné takové vrcholy nenašly, obrazce se neprotínají a metoda vrátí hodnotu false.

5. Závěr

5.1. Zhodnocení projektu

Cílem projektu bylo převést klasickou stolní hru Triplanetary do počítačové podoby, která by byla důstojnou nástupkyní jejích elegantně jednoduchých ale přesto propracovaných principů, což se podařilo, i když současná implementace nenabízí celou škálu původně dostupných herních scénářů.

I když umělá inteligence počítačem ovládaných protivníků ve scénáři Grand Tour nepředstavuje pro zkušeného závodníka příliš velkou konkurenci, alespoň začátečníky by mohla potrápit, protože létá víceméně bezpečně a nezapomíná včas doplňovat palivo. Také v případě, že si hráči navolí neobvyklé checkpointy a že nezáleží na pořadí průletu kontrolních bodů, má AI šanci, protože poletí po nejkratší cestě, zatímco hráči budou ideální trasu většinou pouze odhadovat.

Z tohoto projektu jsem si odnesl mnoho zkušeností a spoustu věcí bych dnes již navrhl a implementoval jinak. Například herní logika a grafické uživatelské rozhraní by mohly být od sebe jasně oddělené, čímž by se umožnila snadná výměna GUI bez zásahu do herního jádra. S tím také souvisí výměna dnes nepříliš perspektivní knihovny pro uživatelské rozhraní Swing, jehož vývoj je momentálně již jen v udržovacím režimu.

Nezanedbatelnou obtíží je postoj současného majitele autorských práv Steve Jackson Games[3], pro nějž je i vydání freewareového open source projektu v konfliktu s jeho licenčním programem. Což také vysvětluje, proč se na trhu nevyskytují žádné další implementace, se kterými by se dal tento projekt srovnávat.

5.2. Budoucnost projektu

Hra by měla v budoucnu získat hezčí grafiku, podporu zvukových efektů a případně animací. Dále lze implementovat i zbylé scénáře původní stolní hry a přidat vynechané funkce jako například rabování (více vizte 2.1).

Na škodu by rozhodně nebyla ani síťová hra pomocí internetu, která by jistě pomohla přilákat více hráčů.

6. Seznam použité literatury

[1] Původní hra Triplanetary
http://en.wikipedia.org/wiki/Triplanetary

[2] Původní hra Triplanetary s možností stáhnutí pravidel
http://boardgamegeek.com/boardgame/3637/triplanetary

[3] Současný majitel práv na hru Triplanetary Steve Jackson Games
http://www.sjgames.com/

[4] Newtonovy pohybové zákony
http://cs.wikipedia.org/wiki/Newtonovy_pohybov%C3%A9_z%C3%A1kony

[5] Knihovna s šestiúhelníkovými tlačítky
http://www.keang.co.uk/hex.html

[6] Článek Clarka Verbruggeho na stránce Amita Patela
http://www-cs-students.stanford.edu/~amitp/Articles/HexLOS.html

[7] Webové stránky firmy Oracle s odkazy na stažení Javy
http://www.oracle.com/technetwork/java/javase/downloads/index.html

[8] Tutoriály pro načítání tříd mimo class path
http://www.exampledepot.com/egs/java.lang/loadclass.html
http://dzone.com/snippets/get-all-classes-within-package

7. Seznam tabulek

8. Seznam použitých zkratek

AI – Artificial Intelligence, umělá inteligence

API - Application Programming Interface, rozhraní pro programování aplikací

AWT – Abstact Window Toolkit, knihovna pro grafické uživatelské rozhraní

CD – Compact Disk, kompaktní disk

CSV – Comma Separated Value, textový souborový formát pro uchovávání tabulek

DOM – Document Object Model

IDE – Integrated Development Enviroment, vývojové prostředí

SAX – Simple API for XML, sekvenční parser XML souborů

XML - Extensible Markup Language, značkovací jazyk pro snadnou výměnu dat v textové podobě

9. Přílohy

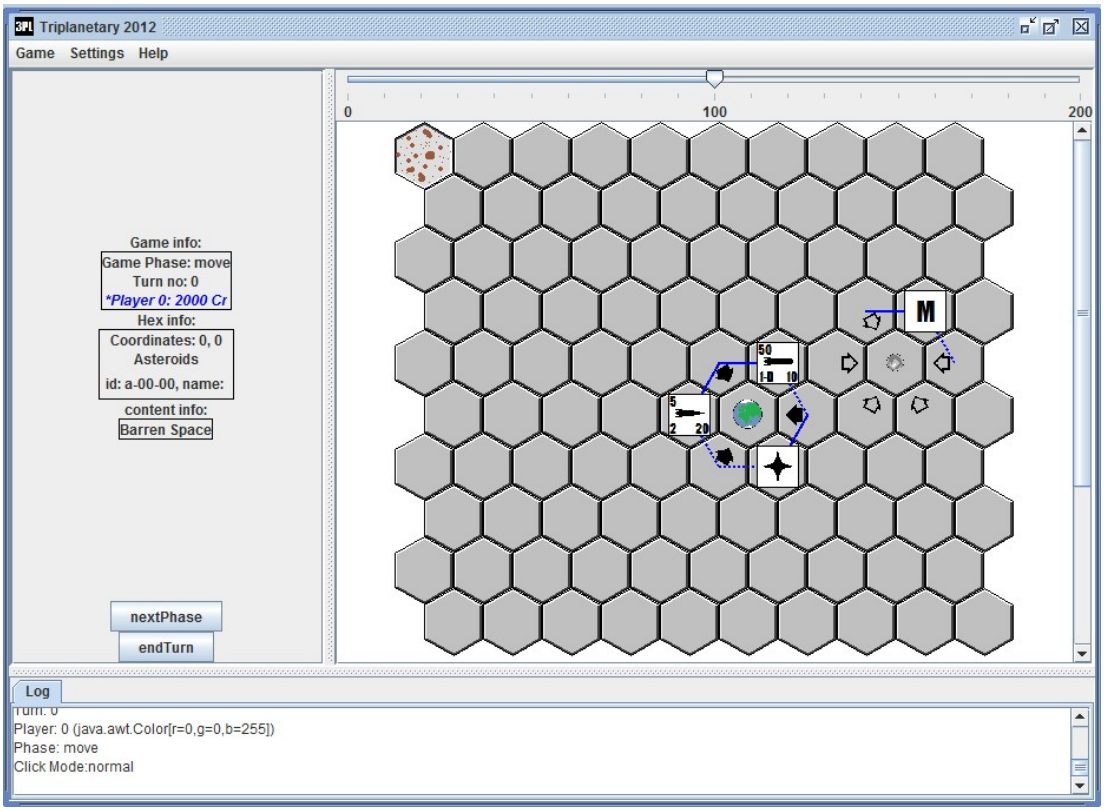
```
<?xml version="1.0" encoding="windows-1250"?>
<gameBoard name="EarthlSystem" width="10" height="10" players="1">
  <!-- LUNA -->
  <gravity id="g-07-03" x="07" y="03" orientation="SE" weak="true"/>
  <gravity id="g-08-03" x="8" y="3" orientation="SW" weak="true"/>
  <gravity id="g-07-04" x="7" y="4" orientation="E" weak="true"/>
  <planetoid id="Luna" name="Luna" image="luna" x="08" y="04" radius="0.3" type="planet"/>
  <gravity id="g-09-04" x="9" y="4" orientation="W" weak="true"/>
  <gravity id="g-07-05" x="7" y="5" orientation="NE" weak="true"/>
  <gravity id="g-08-05" x="8" y="5" orientation="NW" weak="true"/>
  <!-- LUNA -->

  <asteroids id="a-00-00" x="0" y="0"/>

  <!-- TERRA -->
  <gravity id="g-05-04" x="5" y="4" orientation="SE"/>
  <gravity id="g-06-04" x="6" y="4" orientation="SW"/>
  <gravity id="g-04-05" x="4" y="5" orientation="E"/>
  <planetoid id="Terra" name="Terra" image="terra" x="5" y="5" radius="0.5" type="planet">
    <NE>
      <base id="TerranBaseNE" name="TerranBaseNE" type="planetary" player="-1">
        <ship id="LandedCorvette" name="LandedCorvette" type="corvette" player="0" landingVector="NE"/>
      </base>
    </NE>
    <SW>
      <ship id="TerranTransport" name="TerranTransport" type="transport" player="0" landingVector="E"/>
    </SW>
  </planetoid>
  <gravity id="g-06-05" x="6" y="5" orientation="W"/>
  <gravity id="g-05-06" x="5" y="6" orientation="NE"/>
  <gravity id="g-06-06" x="6" y="6" orientation="NW"/>
  <!-- TERRA -->

  <ship id="OrbitalTransport" name="OrbitTransport" x="6" y="4" lastX="5" lastY="4" type="transport" player="0"
fuel="10">
    <base id="CarriedBase" name="CarriedBase" type="orbital" player="0"/>
  </ship>
  <ship id="OrbitalCorvette" name="OrbitalCorvette" x="4" y="5" lastX="5" lastY="4" type="corvette" player="0">
    <torpedo id="Torpedo"/>
  </ship>
  <base id="OrbitalBase" name="OrbitalBase" x="6" y="6" lastX="6" lastY="5" type="orbital" player="0"/>
  <mine id="Mine" x="8" y="3" lastX="7" lastY="3" turnsLeft="4" player="0"/>
</gameBoard>
```

Tabulka 13: Příklad jednoduché mapy



Obrázek 9 1: Zobrazení jednoduché mapy

9.1. Obsah přiloženého CD

Na CD, které je součástí této práce, se nachází tento obsah:

- triplanetary.pdf
  - elektronická verze tohoto textu
- distribution
  - zkompilovaná verze hry připravená pro distribuci
- documentation
  - vývojářská dokumentace k projektu vytvořená utilitou javadoc
- project
  - zdrojové kódy hry ve formě projektu pro IDE Netbeans 7.0.1
- rules
  - pravidla původní stolní hry Triplanetary[2]