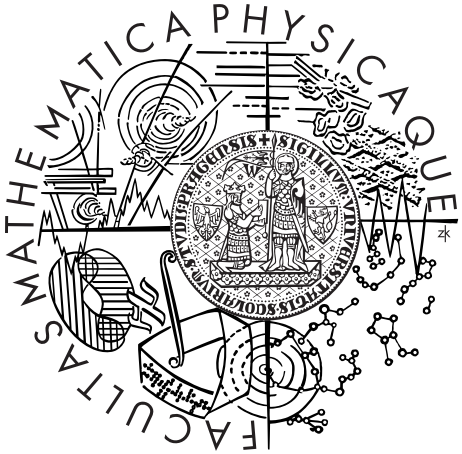
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Dominik Škoda

Monkey Bank Robbers

Katedra aplikované matematiky

Vedoucí bakalářské práce:	RNDr. Bernard Lidický PhD.
Studijní program:	Informatika, programování (IP)
Studijní obor:	Informatika

Praha 2011

Na tomto místě bych velice rád poděkoval vedoucímu své práce RNDr. Bernardu Lidickému PhD. za bystré nápady a pomoc při práci na projektu.

Také bych rád poděkoval svým rodičům a nejbližším, za vytvoření zázemí, které umožnilo této práci vzniknout.

Prohlašuji, že jsem tuto bakalářskou 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 3. srpna 2011 Podpis autora

Název práce: Monkey Bank Robbers

Autor: Dominik Škoda

Katedra: Katedra aplikované matematiky

Vedoucí bakalářské práce: RNDr. Bernard Lidický PhD., Katedra aplikované matematiky

Abstrakt: Práce se týká vytvoření počítačové hry inspirované hrou RoboRally. Program následuje schéma klient-server a umožňuje hráčům hrát společně hru po internetu. Aplikace běžící na serveru spravuje jednotlivé hry a stará se o synchronizaci jednotlivých hráčů. Program dále dovoluje implementovat umělou inteligenci, která se může připojovat do her s jinými hráči. Každý hráč si může vytvořit jakoukoliv mapu a hrát hru s kteroukoliv mapou. Projekt je navržen tak, aby se v budoucnu dal snadno rozšířit. Program je vytvořen v jazyce JAVA.

Klíčová slova: Desková hra, klient-server architektura.

Title: Monkey Bank Robbers

Author: Dominik Škoda

Department: Department of Applied Mathemetics

Supervisor: RNDr. Bernard Lidický PhD., Department of Applied Mathemetics

Abstract: The project concerns of creation of computer game inspirated by the game RoboRally. The program is based on Client-Server architecture and allowes players to play together via Internet. The application running on server manages individual games and takes care of synchronization between players. The program allowes to implement artificial inteligence which can join individual game with another players. Each player can create any map and play a game with any map. The project is designed for simple extension in future. The program is developed in JAVA language.

Keywords: Board Game, Client-Server Architecture.

Obsah

Úvod

RoboRally [1] je moderní akční logická desková hra pro více hráčů. Každý hráč ovládá svého robota a pomocí akčních karet mu plánuje pohyb po hrací ploše. Cílem každého hráče je projet praporky rozmístěné na hrací ploše v pořadí jejich číslování a stát se prvním hráčem, kterému se tak podaří učinit. Nejzábavnějším prvkem hry je však interakce mezi roboty a působení továrních zařízení na ně. Tím se z hrací plochy stává místo, kde na roboty číhá nebezpečí na každém kroku.

Monkey Bank Robbers je počítačová hra, jejíž vytvoření je inspirováno hrou RoboRally a snaží se přenést její prvky z deskové hry do hry počítačové. Při návrhu projektu na napsání této hry neexistovala žádná elektronická verze RoboRally. Nyní sice existuje, ale její hraní s jinými hráči je podmíněno registrovaným členstvím.

Hra samotná nemá žádný příběh. Skládá se pouze z hraní jednotlivých map s více hráči či s počítačem řízenými protivníky. Hra obsahuje základní mapy, které se shodují s hracími deskami RoboRally. Hráč má také možnost v editoru map upravovat či vytvářet mapy vlastní.

Na rozdíl od počítačové hry RoboRally, která má za cíl věrně kopírovat svou deskovou předlohu, si Monkey Bank Robbers klade za cíl snadnou rozšiřitelnost o nová políčka, či nové akční kartičky. A s tímto předpokladem byl samotný program navrhován. Dále se také počítá se snadným nahrazením celé grafiky programu, aby hra mohla vypadat podobně jako RoboRally či zcela odlišně. Při samotném vývoji programu ovšem nebyl na grafickou stránku kladen důraz.

Ještě bych rád pro přehlednost na úvod uvedl, že hra probíhá v kolech a každé kolo sestává z pěti tahů, v každém tahu se provádějí definované akce v definovaném pořadí. Toto je dále přesně rozvedeno v části s pravidly hry 8.1.

První kapitola práce analyzuje odlišnosti od předlohy, jíž byla hra RoboRally. Je zde rozebráno, co v Monkey Bank Robbers chybí, co bylo přidáno a jak byla změněna pravidla hry.

Druhá kapitola popisuje hospodaření aplikace s externími zdroji, jejich načítání z disku a to, v jaké podobě se k nim v programu přistupuje.

Třetí kapitola se věnuje popisu jádra hry. Je v ní rozebrán jazyk, ve kterém byl projekt vytvořen a knihovny, které byly použity. Popisují se tam také hlavní objekty jádra hry, jejich propojení a možnost jejich interpretace v textové podobě.

Klientské části aplikace se věnuje čtvrtá kapitola. Jsou v ní popsány hlavní objekty této části aplikace a jejich propojení.

Serverové části aplikace je věnována kapitola pátá. Opět jsou v ní popsány objekty této části aplikace a jejich vzájemné propojení.

Komunikace mezi klientem a serverem je popsána v kapitole šesté. Jsou zde informace od tvaru zpráv až po jejich výčet i s významem.

To, jak samotná aplikace funguje je rozebráno v kapitole sedmé. V této kapitole je popsáno, jak funguje aplikace na straně klienta, serveru i v samotném jádře.

Osmá kapitola se věnuje uživatelské dokumentaci. Jsou v ní popsána pravidla hry, uživatelské rozhraní programu a jeho ovládání i samotná instalace programu.

Na závěr celé práce je projekt porovnán s obdobnými hrami a je nastíněno, kam se bude projekt dále ubírat.

1. Odlišnosti od RoboRally

1.1 Chybějící prvky

Monkey Bank Robbers oproti RoboRally neobsahuje karty vylepšení. Je tak učiněno z důvodu značného zjednodušení programu. Díky tomuto usnadnění jsem se mohl více zaměřit na komunikaci mezi klienty a serverem a synchronizaci hráčů ve hře.

V předloze také existuje políčko, na kterém se sbíhají dva pohyblivé pásy, které ústí jedním směrem, nebo políčko, na kterém je zeď s více než jedním laserem. Tyto prvky jsem také vynechal, protože hra samotná je plnohodnotná i bez nich. Navíc vrstvy políček ve hře jsou navrženy pro snadné rozšíření a tak se dané prvky dají doplnit, pokud se k nim udělá odpovídající grafické znázornění.

1.2 Přidané prvky

Hra je navrhnuta tak, aby se dala snadno rozšířit. Lze tedy definovat nové vrstvy v políčkách hrací plochy. Více o tomto je popsáno v části o implementaci hry. Monkey Bank Robbers v daném stavu neobsahuje žádné prvky navíc od předlohy, kvůli snaze o věrnost k předloze dle zadání.

1.3 Pozměněná pravidla

Monkey Bank Robbers má oproti RoboRally mírně pozměněná pravidla. Snižování poškození se provádí po každém tahu v kole, pokud jednotka stojí na políčku s touto vlastností, oproti provádění této akce pouze na konci kola. Navštívení vlaječky se započítá pouze, pokud jednotka stojí na takovém políčku na konci kola, oproti započtení na konci jakéhokoliv tahu v kole.

2. Externí zdroje

Externí zdroje programu jsou rozděleny do třech kategorií, každé kategorii pak odpovídá jedna složka s daným zdrojem. Kategorie jsou následující: grafické, zvukové a mapové. Jim odpovídající složky jsou po řadě: `res/graphics`, `res/sounds` a `res/maps`. Vyjímku tvoří soubory, které obsahují definici základních vrstev políček a jejich preferované pořadí, které jsou také uloženy v `res/maps` i když se o mapy nejedná.

2.1 Grafické a zvukové zdroje

Grafické zdroje jsou soubory, které obsahují binární data. Těmito daty jsou obrázky uložené v nějakém standardizovaném formátu. Mnou použitý formát je .png[1], který byl zvolen pro výborný kompresní poměr, kterého dosahuje na programem používaných obrázcích a také z důvodu, že umožňuje uložit alfa kanál obrázků.

Grafické zdroje jsou rozděleny do složek: cards - pro obrázky karet, efects - pro obrázky efektů ve hře, environment - pro obrázky na jednotlivých políčkách ve hře, monkeys - pro obrázky jednotek, numbers - pro obrázky používaných číslic, screens - pro obrázky obrazovek se zprávami, tokens - pro obrázky žetonů.

Zvukové soubory již nejsou dále členěny, protože jich je pouze malé množství. Zvukové soubory jsou ve formátu .wav[2]

Grafické i zvukové soubory jsou do aplikace načítány standardními knihovnami jazyka JAVA.

2.2 Reprezentace objektů

Objekty, které jsou přenášeny po síti mezi klientem a serverem, nebo jsou ukládány do soubobů na disku je třeba také reprezentovat. V úvahu připadaly dvě možnosti. První je použití serializace objektů přímo jazykem JAVA a tedy implementování interface `Serializable` na takových objektech. Druhá možnost je textová reprezentace objektů. Zvolil jsem možnost textové reprezentace z několika důvodů. Implementování interface `Serializable` je podmíněno implementováním tohoto interface na všechny objekty v dané třídě obsažené a některé knihovní objekty, například `image` tento interface nemají a ani nemohou mít implementován. Také k reprezentaci některých objektů není potřeba ukládat všechny proměnné v něm obsažené, u některých stačí jen nejdůležitější charakteristiky a tím se zmenší i objem dat, který je po síti přenášen.

Byla tedy použita textová reprezentace objektů ve formátu XML[3] [4], který je pokryt standardem, má knihovní podporu, je dobře strojově čitelný a lze jej upravovat v prostém textovém editoru. Postupy, jak s tímto jazykem pracovat jsou uvedeny v [5] Použitím jazyka XML sice značně naroste prostor uložených dat, ale je to jediná nevýhoda, kterou jsem tím musel přijmout.

[1] Portable Network Graphics.
[2] Waveform Audio File Format
[3] Extensible Markup Language.

Objekty, které lze reprezentovat v jazyce XML mají předefinovanou funkci `toString()`, která vrací jejich textovou podobu v jazyce XML. Takovými objekty jsou: `Card`, `CardPackage`, `Croft`, `Environment`, `Layer` a `Unit`. Všechny tyto objekty jsou v textové podobě přenášeny mezi klientem a serverem.

Mapy se na disk ukládají také v textové podobě v jazyce XML. Na disku tvoří soubor s příponou .xml a nejde o nic jiného, než o uloženou textovou podobu objektu `Environment`.

Na disku ve složce `res/maps` jsou také uloženy dva zvláštní soubory. Jedná se o `___basic_layers.xml` a `___basic_layers_order.xml`. První soubor obsahuje definice základních vrstev, ze kterých se skládají políčka. Jeho ukázka je v příloze na straně 35. Samotné vrstvy jsou poté reprezentovány pouze svým jménem a jejich definice jsou načteny v programu jak na straně serveru, tak na straně klienta. Druhý soubor obsahuje jednotlivé vrstvy a to v pořadí, ve kterém je preferováno jejich zobrazení v mapovém editoru. Ukázka tohoto souboru je v příloze na straně 36.

2.3 Načtené externí zdroje

Grafické zdroje a všechny základní vrstvy jsou do programu načteny při jeho spuštění. Jedná se o nejobjemnější soubory. Nestane se tedy, že by při jejich potřebě musel uživatel čekat na jejich načtení.

Obrázky jsou v programu drženy a poskytovány objektem třídy `ImagesHolder`. Tento objekt umožňuje přístup k jednotlivým obrázkům pomocí jména souboru s obrázkem bez přípony. Na straně serveru není vytvořena instance tohoto objektu a tím nedochází k načtení obrázků do programu běžícího na serveru.

Vrstvy jsou v programu drženy a poskytovány objektem třídy `LayersHolder`. Tento objekt poskytuje přístup ke všem vrstvám pomocí jejich jména. V celém programu existuje vždy jen jedna instance každé vrstvy. To velice zjednodušuje animaci všech políček na hrací ploše. Každé políčko obsahuje reference na vrstvy, ze kterých se skládá. Objekt třídy `LayersHolder` poskytuje metodu, která posune index zobrazovaného obrázku dané vrstvy a tím dochází k animování dané vrstvy ve všech políčkách hrací plochy, kde je vrstva obsažena. Odpadá tím při animaci procházení všech políček hrací plochy, kterých je 192 a značně se tím snižuje režie hry.

3. Jádro hry

Celé jádro hry se nachází v balíčku `Shared`. Jádro je tvořeno třídami, které reprezentují jednotlivé objekty ve hře, hru samotnou a třídy, které umožňují hru hrát nebo definují veličiny v ní.

3.1 Jazyk a knihovny

Celý projekt je psán v jazyce JAVA 6 [2]. Jsou používány standardní knihovny tohoto jazyka a navíc dvě knihovny externí. Těmito knihovnami jsou `org.xml.sax`, ta obsahuje SAX[1] [6] parser XML souborů a `org.apache.commons.io.IOUtils`, ze které je používána funkce `toInputStream(String)`, která řetězec v parametru vrací v podobě `InputStreamu`. Díky překladu zdrojového kódu do bytecode, uživatel nemusí řešit instalování externích knihoven. Potřebné třídy jsou přeloženy přímo do výsledné aplikace.

3.2 Objekty jádra

Objekty jádra hry tvoří: hrací plocha, která je dále rozdělena na políčka a ta jsou rozdělena do vrstev, jednotky, které se při hře pohybují po hrací ploše, balíčky karet, které jsou tvořeny jednotlivými kartami a simulátor, odehrávající jednotlivá kola hry. Pro určení různých směrů ve hře jsou použity tři výčtové typy. `Movement` reprezentuje relativní[2] směr pohybu a je používán ve třídě reprezentující kartu. `Direction` reprezentuje absolutní[3] směr pohybu. `Rotation` reprezentuje směr rotace.

Dále je zde definován interface `Driftable`, ten definuje metody pro samotný pohyb jednotek po hrací ploše a pro pohyb prvků na hrací ploše. Na objektu, který implementuje tento interface, jsou simulátorem volány funkce pro již vypočítaný pohyb.

Hrací plocha je reprezentována třídou `Environment`. Objekt této třídy obsahuje jednotlivá políčka hrací plochy, jednotky v dané hře, název mapy a název hry, kapacitu hry a identifikátor hry. Kapacita hry je číslo, které určuje, pro kolik hráčů byla hra založena. Identifikátor hry je unikátní číslo, pod kterým je založená hra vedena na serveru.

Políčko hry je reprezentováno třídou `Croft`. Objekt této třídy obsahuje pole vrstev políčka. Každé políčko je definováno až šesti vrstvami. Úroveň vrstvy, tedy její pozice v poli, je definována jejím sériovým číslem. Je zřejmé, že v jednom políčku tedy nemohou existovat dvě vrstvy na stejné úrovni.

Vrstva políčka je reprezentována třídou `Layer`. Objekt této třídy obsahuje následující: název vrstvy, pod kterým je vrstva definována, fázi tahu, ve kterém je vrstva aktivní, směr, kterým vrstva posouvá jednotku, když je aktivní a jednotka se nachází na políčku s touto vrstvou, směr, kterým vrstva rotuje jednotkou, když je aktivní a jednotka se nachází na políčku s touto vrstvou, velikost poškození,

[1] Simple API for XML

[2] Vzhledem ke směru, ve kterém je jednotka natočena

[3] Vzhledem k hrací ploše

které tato vrstva způsobuje, směr, ve kterém se na dané vrstvě nachází zeď, směr, ve kterém se na dané vrstvě nachází laser, sériové číslo vrstvy, řetězec definující speciální vlastnosti vrstvy, název obrázku, reprezentující tuto vrstvu, obrázek této vrstvy, transformaci obrázku vrstvy před jeho zobrazením, počet animací dané vrstvy a proměnnou, která určuje rozdíl souřadnic myši a souřadnic levého horního rohu vrstvy, při jejím přetahování myší.

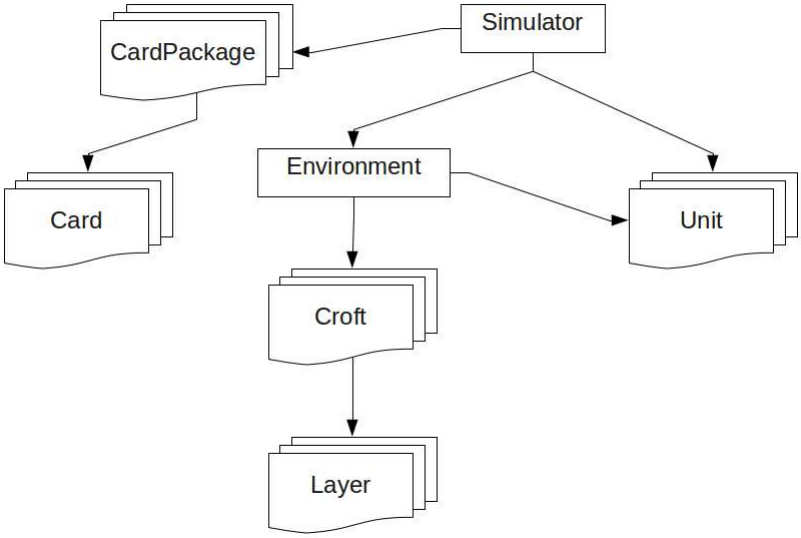
Jednotka je reprezentována třídou `Unit`. Objekt této třídy obsahuje následující: poškození jednotky, indikátor, zda je jednotka odstavena, indikátor naplánování odstavení jednotky, počet životů jednotky, pozici jednotky na hrací ploše, pozici místa, na které je jednotka umístěna při případném návratu do hry, směr, kterým je jednotka na hrací ploše otočena, počet vlaječek, které jednotka úspěšně navštívila, identifikátor jednotky, identifikátor, zda je jednotka aktivní, index přidělené barvy jednotky, číslo pozice, na které jednotka dokončila hru, posun jednotky oproti její pozici v ose x a y, a index do obrázku, který ukazuje natočení jednotky.

Balíček kater je reprezentován třídou `CardPackage`. Objekt této třídy obsahuje identifikátor balíčku a seznam karet v balíčku obsažených.

Karta je reprezentována třídou `Card`. Objekt této třídy obsahuje relativní směr pohybu na kartě, vzdálenost pohybu na kartě, směr rotace na kartě, prioritu dané karty, indikátor, zda je karta zamčená, obrázek dané karty, obrázek zámku na kartě a obrázky číslic priority. Nakonec je zde ještě rozdíl souřadnic myši a souřadnic levého horního rohu karty, při jejím přetahování myší.

Simulátor je reprezentován třídou `Simulator`. Tato třída má na starosti odehrání jednotlivých kol dané hry. Objekt této třídy obsahuje hrací plochu a jednotky v dané hře, objekt implementující interface `Driftable` a seznam balíčků karet na dané kolo hry. Simulátor dostane balíčky naplánovaných karet jednotlivých jednotek. Podle identifikátoru jednotky a balíčku karet pozná, které karty patří které jednotce. Jedno kolo simulátor odehraje v pěti tazích. Každý tah provede následující akce. Vybere dosud nepoužitou kartu daného tahu s nejvyšší prioritou. Vypočítá pohyb jadnotky, které tato karta patří. Sem spadá kontrola zdí ve směru pohybu, které mohou pohybu zabránit a kontrola dalších jednotek, které mohou být pohybem posunuty. Následně se provede pohyb samotný a to nejem jednotky, jejíž karta se vykonává, ale také jednotek, které jsou tímto pohybem ovlivněny. Pohyb samotný je uskutečněn objektem s interface `Driftable`. Na konci pohybu se zkontroluje, zda některá jednotka neskončila v díře, či nevyjela z hrací plochy. Takto se odehrají všechny karty daného tahu. Následně se provede pohyb prvků na hrací ploše jak je popsáno v návodu 8.1 společně s jednotkami, které jsou tímto ovlivněny. Pohyb samotný opět obstarává objekt s interface `Driftable`. Po této fázi dojde v vystřelení všech laserů na hrací ploše a laserů z jednotek. Vypočte se poškození jednotek, které je jim přidáno. Na závěr tahu se odebere poškození jednotkám, které stojí na políčkách, která odebírají poškození. Takto se provede každý tah daného kola. Na konci kola se přidá navštívená vlaječka jednotkám, které stojí na políčku s vlaječkou, již mají právě navštívit. Po zahrání jednoho kola simulátor čeká na seznam balíčků naplánovaných karet pro kolo následující.

Relativní směr je reprezentován výčtovým typem `Movement`. Je použit výhradně ve třídě `Card`. Obsahuje hodnoty FORWARD, pro určení směru kupředu, BACKWARD, pro určení směru dozadu, RIGHT pro určení směru vpravo a LEFT, pro určení směru vlevo.



Obrázek 3.1: Provázání objektů jádra hry.

Absolutní směr je reprezentován výčtovým typem `Direction`. Obsahuje hodnoty NORTH, pro určení směru na sever hrací plochy, SOUTH, pro určení směru na jih hrací plochy, WEST, pro určení směru na západ hrací plochy, EAST pro určení směru na východ hrací plochy a NO pro určení žádného směru.

Směr rotace je reprezentován výčtovým typem `Rotation`. Obsahuje hodnoty RIGHT, pro rotaci doprava o 90 °, LEFT, pro rotaci doleva o 90 °, U_TURN, pro rotaci o 180 ° a NO, pro rotaci o 0 °.

Matody pro pohyb jsou definovány v interface `Driftable`. Jedná se o pohyb jednotek a prvků hrací plochy. Tyto metody jsou volány objektem třídy `Simulator`, po vypočtení výsledného pohybu. Použití tohoto interface odděluje implementaci pohybu na serveru a na klientské aplikaci. Na serveru je kladen důraz na co největší jednoduchost a rychlost těchto metod. Na klientské aplikaci tyto metody zajišťují také animace pohybu a zvuky s tím spojené. Dále u klienta dochází ke zvýraznění právě hraných karet a k jejich zamykání, případně odemykání. Toto se děje také prostřednictvím objektu, který v klientské aplikaci implementuje interface `Driftable`. Takový objekt tedy zajišťuje také zpětnou vazbu, odehrávání kola hry, na zbytek aplikace.

3.3 Provázání objektů

Jedna hra je reprezentována objekty `Environment` a `Simulator`. `Environment` obsahuje všechna políčka (`Croft`) hrací plochy a všechny jednotky (`Unit`) ve hře. `Croft` obsahuje šest vrstev (`Layer`), které ho definují. `Simulator` obsahuje reference na `Environment` a opět na všechny jednotky (`Unit`) ve hře, pro snadný přístup k nim. Dále také `Simulator` obsahuje objekt implementující interface `Driftable`. Na začátku odehrání každého kola, je do objektu `Simulator` předán seznam balíčků karet (`CardPackage`). Balíček karet obsahuje seznam jednotlivých karet (`Card`). Přehledné provázání objektů jádra hry je znázorněno na obrázku 3.1.

3.4 Načítání a ukládání objektů

Některé objekty je třeba ukládat a načítat z disku. Jedná se o objekty `Environment` a `Layer`. Některé objekty je potřeba posílat po síti mezi klientskou a serverovou aplikací. Takovými objekty jsou `Environment`, `Unit`, `CardPackage`, `Card` a všechny výše popsané výčtové typy. Proto mají všechny tyto objekty předefinovanou funkci `toString()`. Tato funkce vrací popis daného objektu v textové formě v jazyce XML. V takovéto podobě se objekt snadno uloží na disk nebo pošle po síti.

Výčtové typy jsou v základu jazyka JAVA v textové formě reprezentovány názvem své hodnoty. Takováto reprezentace je pro naše účely dostačující, proto nebylo nutné funkci `toString()` u výčtových typů předefinovávat.

Načítání objektů z textové podoby v jazyce XML se v programu provádí SAX parserem. Tento parser je velice jednoduchý a rychlý. SAX parser ovšem potřebuje k načtení určitého textu v jazyce XML odpovídající handler. Jedná se o objekt, který implementuje imterface obsahující metody, které jsou volané parserem při načítání textu v jazyce XML. Takovými objekty jsou v projektu `XMLCardLoader`, pro načítání karet, `XMLLayerLoader`, pro načítání vrstev, `XMLLayerOrderLoader`, pro načítání preferovaného pořadí vrstev, `XMLMapLoader`, pro načítání hrací plochy a `XMLUnitLoader`, pro načítání jednotek. Tyto objekty slouží k načítání řečených objektů.

Objekt `Environment` je mimo jiné definován také polem obsahujícím políčka hrací plochy. Políčka jsou definována polem vrstev. Když tedy chceme objekt `Environment` transformovat do textové podoby, musíme transformovat i objekty `Croft` a `Layer`. Proto mají i tyto objekty předefinované metody `toString()`, které vracejí jejich textovou podobu v jazyce XML. Naprosto stejná situace nastává u objektu `CardPackage`, který obsahuje objekty `Card`. I objekt `Card` má tedy předefinovanou funkci `toString()`, která vrací jeho textovou podobu v jazyce XML.

Ukázka, jak vypadají jednotlivé objekty v textové podobě, je přiložena v příloze. `Environment` se nachází na straně 39, `Unit` se nachází na straně 40, `Card` se nachází na straně 37 a `CardPackage` se nachází na straně 38.

4. Klientská část aplikace

Klientská část aplikace se nachází v balíčku `Client`. Jsou zde třídy grafického uživatelského rozhraní a třídy, které zajišťují komunikaci se serverovou částí aplikace. Také se zde nachází třída, která implementuje rozhraní `Driftable` z jádra aplikace.

Spolu s jádrem aplikace, tedy balíkem `Shared` 3, vytvářejí téměř celou klientskou aplikaci. Program, který vznikne přeložením těchto dvou balíků poskytuje rozhraní, díky kterému je již možné hru hrát, pokud na serveru běží serverová část aplikace.

Bez připojení k serveru je v programu možné pouze vytvářet nové mapy a měnit ty stávající. Pokud se uživatel pokusí zahájit hru, ať už jejím vytvořením, nebo připojením se k ní, program se automaticky pokusí připojit k serveru se serverovou částí aplikace.

Balík `AI_Client` tvoří poslední článek v klientské části programu. Jedná se o třídy, které umožňují ve hře použít protihráče, jež jsou řízeni počítačem.

4.1 Objekty klientské části

Jednotlivými objekty, které tvoří klientskou část programu jsou: Samotné okno programu, panely[1], které obsahují úvodní menu, menu pro založení nové hry, menu pro připojení k založené hře, spuštěnou hru, mapový editor, nabízené karty, naplánované karty, nabízené vrstvy, přechodové obrazovky se zprávou pro uživatele, status vybrané jednotky ve hře a panel, který zobrazuje hrací plochu. Dále se zde nacházejí dva objekty, které zajišťují komunikaci se serverem. Posledním objektem v balíku `Client` je objekt, implementující rozhraní `Driftable` z balíku `Shared` 3.

Okno programu je reprezentováno třídou `GUI`. Zajišťuje přepínání mezi jednotlivými panely aplikace a zobrazuje je v okně programu. Představuje centrální prvek klientské části aplikace. Dále se stará o připojení aplikace k serveru a o zobrazování zpráv pro uživatele. Při zavření okna aplikace se postará o informování serveru o této události.

Úvodní menu je reprezentováno třídou `LoginScreen`. Umožňuje přepínat mezi editorem map, menu pro založení nové hry a menu pro připojení do hry. Při přepnutí do editoru map dojde k zobrazení samotného editoru v hlavním okně programu, zatímco při přepnutí do jednoho ze zmíněných menu dojde k zobrazení daného menu přímo v panelu s úvodní obrazovkou.

Menu pro založení nové hry je reprezentováno třídou `CreateGamePanel`. Toto menu obsahuje okénko se seznamem dostupných map, panel s hrací plochou, který zobrazuje aktuálně vybranou mapu, pole pro zadání názvu vytvářené hry a nabídku pro počet hráčů ve vytvářené hře. Nakonec je zde tlačítko pro samotné založení hry s nastavenými parametry. Stiskem tohoto tlačítka se na server odešle požadavek o založení nové hry s danou mapou a nastavenými parametry.

Menu pro připojení do hry je reprezentováno třídou `JoinGamePanel`. Toto menu obsahuje okénko se seznamem dostupných her na serveru, panel s hrací

[1] Slovo panel v textu míní třídu JPanel, která slouží k vykreslení vlastního obsahu.

plochou, který zobrazuje mapu aktuálně vybrané hry, a dvě tlačítka. Jedno slouží k aktualizaci seznamu dostupných her, druhé potom k odeslání požadavku na server o připojení do zvolené hry.

Panel se spuštěnou hrou je reprezentován třídou `GameScreen`. Tento panel je v okně programu zobrazován samostatně. Obsahuje panel s hrací plochou, panel s nabízenými kartami, panel s naplánovanými kartami, panel, který zobrazuje status vybrané jednotky ve hře a tlačítka pro ovládání hry. Mezi těmito tlačítky se nachází tlačítko pro potvrzení naplánovaných karet, tlačítko pro opuštění hry a tlačítka, jimiž se přepínají zobrazení informací o jednotkách ve hře. Dále se objekt této třídy stará o přetahování karet myší mezi panelem s naplánovanými kartami a panelem s nabízenými kartami.

Nabízené karty jsou reprezentovány třídou `OfferedCards`. Jedná se o panel, který obsahuje balíček karet přidělený na dané kolo a stará se o jejich zobrazení. Umožňuje odebírat a přidávat jednotlivé karty. Na pozadí tohoto panelu je zobrazena barva, která reprezentuje jednotku konkrétního hráče.

Naplánované karty jsou reprezentovány třídou `PlannedCards`. Jedná se o panel, který obsahuje balíček karet, které hráč plánuje na následující kolo, a stará se o jeho zobrazení. Umožňuje odebírat a přidávat jednotlivé karty. Velikost balíčku karet je zde omezena na pět karet. Pokud před odesláním naplánovaných karet na server neobsahuje balíček karet v tomto panelu pět karet, jsou chybějící karty doplněny náhodně z balíčku karet v panelu s nabízenými kartami.

Status jednotek je zobrazován objektem třídy `StatusPanel`. Tento panel zobrazuje informace o vybrané jednotce. Je zde zobrazeno poškození jednotky, její počet životů, zda je jednotka odstavena a jaké vlaječky jednotka navštívila. Také je zde zobrazena jednotka samotná, kvůli přehlednosti, o kterou jednotku se jedná. Nakonec je v panelu ještě přítomno odpočítávání, které pomocí hodin (třídy `Clock`) odměřuje čas určený k naplánování karet na příští kolo. Zbývající čas je znázorněn progress barem (třída `JProgressBar`). Pokud vyprší čas určený k odeslání naplánovaných karet a hráč dosud nepotvrdil jejich naplánování, dojde k jejich automatickému odeslání a případně k jejich doplnění z panelu s nabízenými kartami.

Panel s hrací plochou je reprezentován třídou `GameBoard`. Tento panel slouží primárně k zobrazování zvolené mapy. Při rozehrané hře dochází v tomto panelu také k zobrazování jednotek ve hře a k zobrazování veškerých animací.

Mapový editor je reprezentován třídou `MapEditor`. Jedná se o panel, který obsahuje panel s hrací plochou, panel s nabízenými vrstvami, okénko se seznamem dostupných map a tlačítka pro ovládání editoru map. Mapový editor umožňuje vytvářet nové mapy a upravovat stávající mapy přetahováním jednotlivých vrstev políček z panelu nabízených vrstev do hrací plochy. O zobrazování přetahované vrstvy se stará samotný objekt třídy `MapEditor`. Dále tento objekt umožňuje načítat existující mapy z disku a ukládat nové mapy na disk.

Nabízené vrstvy jsou obsaženy v panelu `OfferedLayers`. Tento panel obsahuje všechny základní vrstvy, ze kterých se dají poskládat jednotlivá políčka hrací plochy. Stará se také o jejich zobrazování a umožňuje jejich propagaci do hrací plochy s vytvářenou mapou.

Přechodové obrazovky jsou zobrazovány panelem `WaitingPanel`. Tento panel dovoluje nastavit zprávu[2] v podobě obrázku, která má být zobrazena uživateli

[2]Jedná se o zprávy, které jsou zobrazeny, pokud aplikace čeká na odpověď od serveru.

a stará se o její vykreslení do okna programu poté, co je do okna programu přidán. Tento panel obsahuje jediné tlačítko, které umožňuje opustit přechodovou zprávu a navrátit se do úvodního menu.

Odesílání zpráv na server probíhá v objektu třídy `Sender`. Tento objekt poskytuje metody pro odeslání jednotlivých zpráv. Při odesílání některého z objektů z jádra hry 3 se tento objekt postará o jeho transformaci do textové podoby.

Přijímání zpráv ze serveru provádí objekt třídy `Receiver`. Hlavní metoda tohoto objektu běží ve vlastním vlákně, čeká na přijetí zprávy, přijímá zprávu a když je celá zpráva přijata, informuje o tom příslušnou část klientské aplikace.

Pohyb jednotek zajišťuje objekt třídy `Drift`. Tento objekt implementuje interface `Driftable`. Metody tohoto objektu jsou volány objektem třídy `Simulator`, který se nachází v jádru apllikace 3. Tento objekt zajišťuje animování pohybu jednotek i prvků na hrací ploše. Stará se také o přehrávání zvuků prvků hrací plochy. Zajišťuje vazbu na zbytek klientské aplikace. Provádí také zamykání a odemykání naplánovaných karet, jak je uvedeno v pravidlech hry 8.1.

4.2 Provázání objektů

Vstupním bodem programu je metoda `main(String[])` nacházející se ve třídě `Main`. Tato metoda vytvoří nové okno programu reprezentované objektem třídy `GUI` a spustí ho v novém vlákně.

Okno programu si v sobě drží panel, který zobrazuje spuštěnou hru reprezentovaný objektem třídy `GameScreen`, úvodní menu programu reprezentované objektem třídy `LoginScreen`, panel pro zobrazení přechodové obrazovky reprezentovaný objektem třídy `WaitingPanel`, panel obsahující mapový editor, který je reprezentovaný objektem třídy `MapEditor` a zajišťuje přepínání mezi nimi.

Panel se spuštěnou hrou se skládá z hrací plochy (objekt třídy `GameBoard`), panelu nabízených karet (objekt třídy `OfferedCards`), panelu naplánovaných karet (objekt třídy `PlannedCards`), panelu informujícího o stavu vybrané jednotky (objekt třídy `StatusPanel`) a tlačítek hry. Dále obsahuje objekt třídy `Simulator` z jádra hry 3.

Panel s hrací plochou obsahuje objekt třídy `Environment`, který reprezentuje hrací plochu a také se nachází v jádru hry 3.

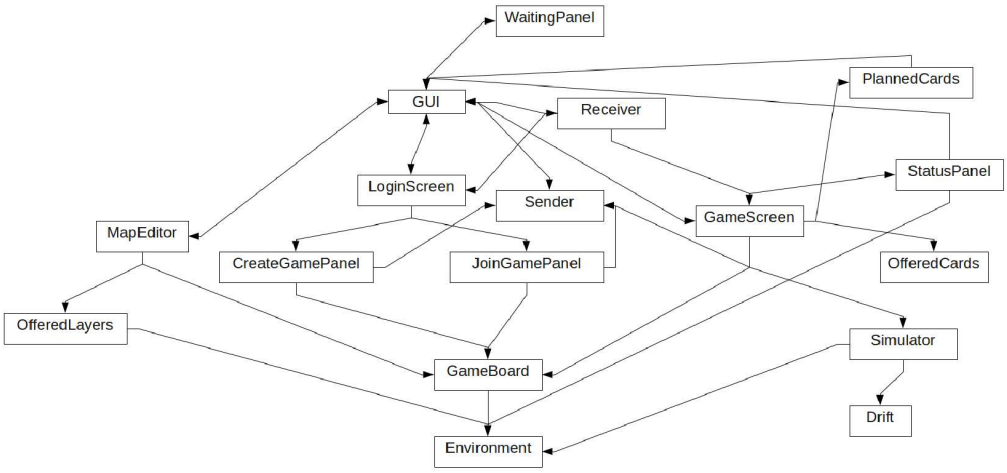
Panel informující o stavu vybrané jednotky obsahuje objekt třídy `Unit`, který reprezentuje jednotku, jejíž informace jsou zobrazovány a který se nachází v jádru hry 3.

Panel s nabízenými kartami a panel s naplánovanými kartami v sobě obsahují balíček karet, který je reprezentován polem karet (objekt třídy `Card`).

Úvodní menu programu obsahuje menu pro založení nové hry (objekt třídy `CreateGamePanel`) a menu pro připojení do hry (objekt třídy `JoinGamePanel`). Umožňuje také přepínat zobrazení mezi těmito objekty a navíc také mezi editorem map.

Mapový editor obsahuje panel s hrací plochou a hrací plochu, která reprezentuje vytvářenou mapu. Dále obsahuje panel s nabízenými vrstvami (objekt třídy `OfferedLayers`), který obsahuje jednotlivé vrstvy (objekty třídy `Layer`).

Všechny objekty balíku `Client` mohou volat metody na objektu okna programu a tím přepínat zobrazený obsah okna a provádět další akce, které třída



Obrázek 4.1: Provázání objektů klientské části.

`GUI` umožňuje. Objekty mohou dále využívat služeb objektu třídy `Sender`, díky kterému mohou posílat zprávy na server.

Podrobné schéma provázání objektů je znázorněno na obrázku 4.1.

4.3 Počítačem ovládaný klient

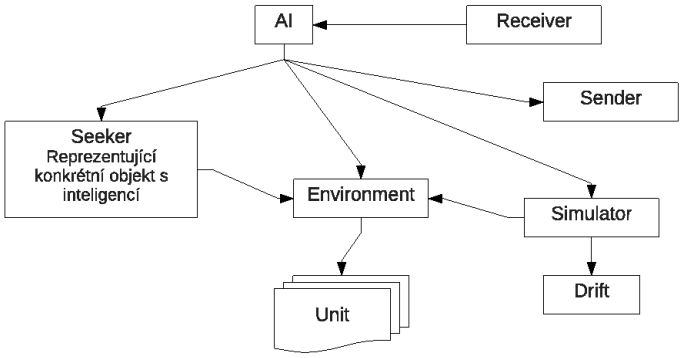
Jedná se o takového klienta, jehož karty plánuje sám program. Třídy této klientské části programu jsou v balíčku `AI_Client`. Jsou zde třídy zajišťující komunikaci se serverem, jedná se o `Sender` a `Receiver`. Dále je zde třída, která vše řídí a je centrálním prvkem tohoto klienta. Touto třídou je `AI`. Také se zde nachází interface `Seeker`, který obsahuje metody, které definují rozhraní třídy s umělou inteligencí. V základu programu je zde třída `SimpleSeeker`, která poskytuje velice jednoduchou umělou inteligenci pro plánování karet. Poslední třída v tomto balíčku je `Drifter`, která implementuje interface `Driftable` a slouží k pohybu jednotek v tomto klientovi.

Tento balíček společně s balíčkem `Shared` tvoří funkční jednotku, která se jako kterýkoli klient připojuje k serveru a komunikuje s ním totožným způsobem.

Funkce objektů tříd `Receiver` a `Sender` jsou stejné jako v klientské části programu popisované výše. Funkce objektu třídy `Drifter` zde ovšem odpovídá objektu třídy stejného názvu na serveru, neboť zde není potřeba nic vizualizovat.

Centrální bod tohoto klienta je, jak bylo zmíněno, objekt třídy `AI`. Nacházejí se zde metody, které reagují na přijaté zprávy ze serveru a starají se o odesílání zpráv na server. Dále je zde zajištěno spouštění plánování karet umělou inteligencí a také odeslání výsledku na server v časovém limitu určeném na plánování karet.

Rozhraní pro umělou inteligenci je popsáno v interface `Seeker`. Úkolem umělé inteligence je z daného balíčku karet vybrat pětici, která reprezentuje naplánované karty tohoto klienta. Jak dobře se o daný úkol postará je věcí konkrétní implementace. Instance klienta s umělou inteligencí je vytvořena v klientské části programu na žádost serveru. Takto vytvořená instance se připojí do dané hry jako kterýkoli jiný klient a server mezi nimi již nerozlišuje. Instance je vytvářena ve třídě `Receiver` v klientské části programu. Toto je místo, ve kterém se dá



Obrázek 4.2: Provázání objektů klientské části s inteligencí.

nahradit implementace inteligence.

Výpočty umělé inteligence mohou býti velice náročné a mohou tak zatěžovat hráčův počítač. Proto se server stará o rovnoměrné rozdělení všech klientů s umělou inteligencí mezi všechny počítače připojených hráčů.

Základní inteligence je zde v podobě objektu třídy `SimpleSeeker`. Obsahuje jednoduchý algoritmus na hledání následující vlaječky a plánování pohybu, aby se na vlaječku její jednotka dostala. Tato třída samozřejmě implementuje interface `Seeker`. Význam jednotlivých metod tohoto interface je následující.

`set_game(Environment)` má na starosti připojení dané mapy hry se všemi jednotkami k objektu s inteligencí.

`set_id(int)` informuje objekt s inteligencí o identifikátoru jednotky tohoto klienta.

`set_AI(AI)` umožňuje provázat inteligenci s centrálním bodem a využívat jeho funkcí. Takovou funkcí, která se dá využít je například odeslání naplánovaného balíčku před koncem doby určené k plánování a tak zkrátit čas čekání ostatních klientů ve hře.

`set_card_package(CardPackage)` je metoda, která startuje plánování karet z daného balíčku. Tato metoda je spuštěna v samostatném vlákně a tak neblokuje zbytek programu.

`add_card(Card)` slouží k doplnění chybějící karty v balíčku s naplánovanými kartami. Doplňující kartu posílá server.

`get_planned_cards()` slouží k ukončení plánování a k vrácení balíčku s naplánovanými kartami. Objekt s umělou inteligencí by po zavolání této metody měl ihned ukončit plánování karet a vrátit svůj výsledek.

Provázání objektů v klientovi s umělou inteligencí je znázorněno na obrázku 4.2.

5. Serverová část aplikace

Serverovou část aplikace tvoří třídy v balíčku `Server`. Jsou zde třídy zajišťující komunikaci s klientem, třída reprezentující běžící hru na serveru, třída, která držící seznam založených her a umožňuje připojení klientů do nich a třída, která zajišťuje pohyb jednotek v běžící hře na serveru, pokud je to potřeba. Spolu s balíčkem `Shared` z jádra aplikace 3 tvoří celou serverovou aplikaci.

Tato aplikace přijímá připojující se klienty. Umožňuje jim zakkládat nové hry a takto založené hry si drží v seznamu. Na žádost klienta poskytuje tento seznam založených her. Zajišťuje připojování klientů do založených her. Pokud se daná hra zaplní klienty, je spuštěna a odstraněna z tohoto seznamu. Spuštěná hra již sama komunikuje s připojenými klienty. Klient je zde reprezentován dvojicí objektů: `Sender` a `Receiver`, které jsou po síti připojeny k danému klientovi.

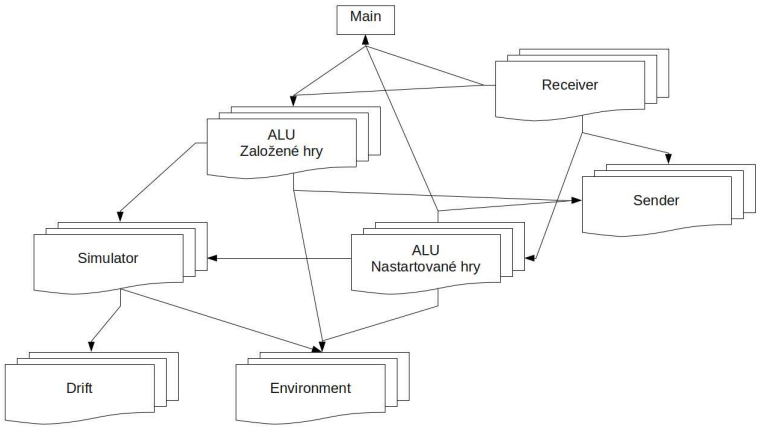
Serverová aplikace může fungovat dvěma způsoby. V jednoduchém režimu jednotlivé spuštěné hry pouze přijímají zprávy od klientů, zajišťují jejich synchronizaci a posílají jim zprávy. V kontrolním režimu navíc kontrolují data, která jsou přijata od jednotlivých klientů a zajišťují tak, že klienti nemohou podvádět. V tomto režimu jsou navíc jednotlivá kola hry simulována na serveru kvůli kontrole pozic jednotlivých jednotek ve hře.

5.1 Objekty serverové části

Centrálním objektem je v serverové aplikaci objekt třídy `Main`. Tento objekt přijímá připojující se klienty a stará se o přidělování unikátního identifikátoru jednotlivým klientům. Omezuje také maximální množství připojených klientů. Připojeným klientům dovoluje zakládat nové hry a stará se o přidělování unikátního identifikátoru jednotlivým hrám. Vede si seznam založených her, tento seznam poskytuje na vyžádání klientům a dovoluje jim, připojit se do konkrétní hry. Pokud je hra zaplněna připojenými klienty, tento objekt ji spustí a vyřadí ji ze svého seznamu založených her.

Jednotlivé hry jsou reprezentovány objektem třídy `ALU`. Takový objekt je založen s danou hrací plochou (objektem třídy `Environment`) z jádra aplikace 3, která definuje název hry, její identifikátor a počet hráčů, kteří se do hry musí připojit. Pokud hra není zaplněna v daném časovém intervalu je ze serveru automaticky smazána. Objekt reprezentující hru umožňuje přidávat jednotlivé klienty do hry. Pokud je přidáno tolik klientů, kolik má být ve hře, je hra smazána ze seznamu založených her a spustí se v samostatném vlákně programu. Odehrání jednoho kola hry na serveru vypadá následovně: Odešlou se balíčky nabízených karet jednotlivým klientům, od klietů jsou následovně přijaty balíčky naplánovaných karet, které jim jsou, po přijetí posledního, odeslány k odehrání kola. Jakmile klienti odehrají kolo s naplánovanými kartami, odešlou na server své jednotky. Na takovou zprávu hra na serveru zareaguje posláním nových balíčků s nabízenými kartami. Pokud je server spuštěný v režimu kontroly, kontrolují se přijaté balíčky od klientů, zda se skládají z nabízených karet a kontrolují se přijaté jednotky, zda odpovídají stavu po odehrání daného kola. Jednotlivá kola jsou tedy ve hře na serveru také simulována.

Pohyb jednotek na serveru zajišťuje objekt třídy `Drift`, který implementuje



Obrázek 5.1: Provázání objektů na serveru.

interface třídy `Driftable` obsažený v jádru hry 3. Na serveru je prioritou jednoduchost a rychlost každého pohybu. Proto tento objekt žádný pohyb neanimuje. Dochází zde pouze k přepsání souřadnic jednotlivých jednotek. Metody tohoto objektu jsou volány objektem třídy `Simulator` z jádra aplikace 3, pokud běží server v režimu kontroly a tudíž je potřeba odehrávat jednotlivá kola hry i na serveru.

Posílání zpráv klientovi má na starosti objekt třídy `Sender`. Tento objekt obsahuje metody pro posílání jednotlivých zpráv. Pokud je potřeba ve zprávě poslat nějaký objekt z jádra aplikace 3, tento objekt se postará o jeho transformaci do textové podoby.

Přijímání zpráv od klienta má na starosti objekt třídy `Receiver`. Tento objekt obsahuje metodu, která běží v samostatném vlákně, čeká na přijetí zprávy a po jejím celém načtení o této skutečnosti informuje tu část aplikace, která má na starosti příslušnou zprávu.

5.2 Provázání objektů

Vstupním bodem do aplikace, je metoda `main(String[])` obsažená v objektu třídy `Main`. Tato metoda se stará o připojování nových klientů k serveru.

Centrální objekt je instance třídy `Main`. Ten obsahuje seznam založených her (objekty třídy `ALU`).

Každá hra na serveru (objekt třídy `ALU`) obsahuje hrací plochu reprezentovanou objektem třídy `Environment` z jádra aplikace 3 a seznam klientů připojených do hry, kteří jsou reprezentováni svým připojením v podobě objektu třídy `Sender`, svou jednotkou v podobě objektu třídy `Unit` z jádra aplikace 3 a svým balíčkem karet v podobě objektu třídy `CardPackage`. Pokud je aplikace na serveru spuštěna v režimu kontrol, obsahuje každá hra také objekt třídy `Simulator` z jádra aplikace 3, který má hrací plochu dané hry a objekt třídy `Drift`.

Objekt třídy `Receiver` obsahuje reference na objekt `Main` a na objekt třídy `ALU`, pokud je klient do nějaké hry připojený. Podrobné schéma provázání objektů je znázorněno na obrázku 5.1.

6. Komunikace klienta a serveru

Pro komunikaci mezi serverem a jednotlivými klienty musel být vyvinut protokol, který celou komunikaci definuje. Samotnou komunikaci tvoří jednotlivé zprávy, které mají nadefinovaný tvar. Zprávy jsou textové a jejich obsah je tvořen pouze znaky ze znakové sady ASCII[1]. Konec každé zprávy je definován znakem, který má hodnotu 0, jedná se o znak '\0'.

6.1 Definice tvaru zpráv

Všechny zprávy, které se posílají mezi klientem a serverem jsou v jazyce XML. Kořenovým elementem každé zprávy je <message>. Každá zpráva obsahuje element <message_type>, který je přímým podelementem elementu <message>. Obsah tohoto elementu definuje typ přenášené zprávy a obsah zbytku zprávy je na něm závislý.

6.2 Zprávy od klienta k serveru

Zprávy od klienta k serveru jsou dvojího druhu. Jedná se o zprávy s žádostí a o zprávy informační. Zprávy s žádostí očekávají nějakou reakci ze strany serveru a jeho odpověď. Informační zprávy pouze informují server o stavu klientské aplikace či rozehrané hry.

Zprávy s žádostí jsou těchto typů: `create_game`, `join_game`, `card_request`, `available_games_request`.

Informační zprávy mají tyto typy: `package`, `error`, `leave_game`, `disconnect`, `unit`.

Zpráva typu `create_game` obsahuje textovou podobu objektu `Environment`. Význam zprávy je požadavek o vytvoření nové hry na serveru. Hra, která se má vytvořit je přesně definována obsahem elementu <environment>.

Zpráva typu `join_game` obsahuje element <game_id>, jehož obsahem je identifikátor založené hry na serveru. Význam zprávy je požadavek o připojení klienta do dané hry.

Zpráva typu `available_games_request` kromě elementu s typem zprávy neobsahuje nic jiného. Požadavkem této zprávy je zaslání výpisu založených her na serveru ke klientovi, který o to požádal.

Zpráva typu `card_request` neobsahuje žádné další informace. Jedná se o požadavek na server, o zaslání chybějící karty tomuto klientovi.

Zpráva typu `package` obsahuje balíček naplánovaných karet v textové podobě. Tato zpráva informuje hru na serveru, do které je daný klient připojený, o hráčem naplánovaných kartách na příští kolo hry.

Zpráva typu `error` obsahuje elementy <error_number>, který obsahuje číselný kód chyby a <error>, který obsahuje textový popis chyby. Tato zpráva informuje server, že došlo k nějaké chybě na straně klienta.

[1] American Standard Code for Information Interchange.

Zpráva typu `leave_game` neobsahuje žádné další informace. Tato zpráva informuje hru na serveru, do které je daný klient připojený o tom, že tento klient opustil danou hru.

Zpráva typu `disconnect` neobsahuje žádné další informace. Zpráva informuje server o tom, že daný klient se odpojuje od serveru.

Zpráva typu `unit` obsahuje textovou podobu jednotky, která patří hráči. Zpráva informuje spuštěnou hru, do které je daný klient připojený o stavu jeho jednotky.

6.3 Zprávy od serveru ke klientovi

Zprávy od serveru ke klientovi jsou všechny informačního charakteru. Buď informují o stavu serveru, nebo o stavu hry na serveru. Jedinou vyjímkou je zpráva `ai_request`, která požaduje na klientovi vytvoření nového klienta s umělou inteligencí.

Zprávy ze serveru jsou následujících typů: `package`, `packages`, `list_of_games`, `game_starts`, `error`, `game_deleted`, `joined_successfully`, `game_created`, `unit`, `card`.

Zpráva typu `ai_request` obsahuje element <game_id>, který obsahuje identifikátor hry, do které se má vytvořený klient s umělou inteligencí připojit. Klient, který tuto zprávu obdrží, je povinen vytvořit klienta s umělou inteligencí.

Zpráva typu `package` obsahuje balíček karet v textové podobě. Tento balíček karet obsahuje karty, ze kterých bude hráč plánovat pohyb své jednotky na další tah. Zároveň tato zpráva informuje klienta, že začala plánovací část nového kola.

Zpráva typu `packages` obsahuje balíčky naplánovaných karet všech hráčů v textové podobě. Tyto balíčky obsahují karty, které se odehrají v tomto kole. Zároveň zpráva informuje klienta, že začala odehrávací část kola.

Zpráva typu `list_of_games` obsahuje všechny hry v textové podobě, které jsou vytvořené na serveru. Hry jsou reprezentované objekty třídy `Environment`. Tato zpráva je reakcí na klientovu žádost `available_games_request`.

Zpráva typu `game_starts` obsahuje hrací plochu dané hry v textové podobě (objekt třídy `Environment`) a všechny jednotky v textové podobě, které jsou do hry zapojené. Zpráva informuje klienta o začátku hry, přičemž specifikuje mapu hry a všechny jednotky ve hře. Zpráva je posílána všem klientům ve hře po jejím nastartování, které nastane po jejím zaplnění jednotlivými klienty.

Zpráva typu `error` obsahuje elementy <error_number>, který obsahuje číselný kód chyby a <error>, který obsahuje textový popis chyby. Tato zpráva informuje klienta, že došlo k nějaké chybě na straně serveru.

Zpráva typu `game_deleted` obsahuje element <game_id>, který obsahuje identifikátor určité hry. Tato zpráva informuje klienta o zrušení dané hry. Tato zpráva je posílána všem klientům, kteří jsou do dané hry připojeni.

Zpráva typu `joined_successfully` obsahuje element <granted_id>, který obsahuje identifikátor, který je přidělený danému klientovi a element <color>, který obsahuje index přidělené barvy danému klientovi. Tato zpráva informuje klienta o úspěšném připojení do hry a čekání na její spuštění.

Zpráva typu `game_created` obsahuje element <game_id>, který obsahuje identifikátor dané hry. Tato zpráva informuje klienta, který poslal požadavek o vytvoření hry, o jejím vytvoření.

Zpráva typu `unit` obsahuje jednu jednotku v textové podobě. Tato zpráva informuje klienta o některé změně, která v dané jednotce nastala. Takovou změnou může být odstavení jednotky na následující kolo, nebo vyřazení jenotky ze hry. Klient tuto změnu rozezná z hodnot poslané jednotky.

Zpráva typu `card` obsahuje jednu kartu v textové podobě. Tato karta je jednotce zaslána na žádost `card_request`. Karta je určena k zaplnění prázdné pozice naplánovaných karet, která je uzamykána.

7. Fungování aplikace

7.1 Jádro hry

Jádro hry se stará o odehrávání jednotlivých kol hry. Simulátor, reprezentovaný objektem třídy `Simulator`, na začátku odehrávací fáze kola dostane balíčky naplánovaných karet všech jednotek. Hrací plochou je zde objekt třídy `Environment` a jednotky jsou drženy v seznamu. Všechny pohyby a pořadí všech akcí se řídí pravidly hry 8.1. Kolo se odehrává v pěti tazích, ve kterých se provádí následující akce.

Pro každou jednotku vypočte a provede její pohyb podle naplánované karty na daný tah. Karty daného tahu jsou brány v pořadí určeným jejich prioritami sestupně. Při výpočtu pohybu simulátor hledá zdi a jednotky v cestě. Pokud se v cestě nacházejí jiné jednotky, je s nimi také pohnuto. Pokud na konci, nebo v průběhu pohybu některá z jednotek skončí mimo hrací plochu nebo na políčku s dírou, ztrácí život a neúčastní se zbytku kola.

Pohne se s prvky na hrací ploše a posune se jednotkami, které jsou tímto ovlivněny.

Vypočítá se, přes která políčka půjdou paprsky laserů a které jednotky jimi budou zasaženy. Poté se provede vypálení laserů a zasaženým jednotkám je přidáno příslušné poškození.

Nakonec tahu se provede odebrání poškození jednotkám, které stojí na políčku, jež odebírá poškození.

Na konec tahu se jednotkám, stojícím na políčku s vlaječkou, která je pro jednotku následující, započítá navštívená vlaječka. Jednotky, které měly naplánované odstavení na další tah jsou odstaveny a jednotky, které byly v tomto tahu odstaveny jsou znovu aktivovány. Jednotky, které v tomto tahu ztratily život a ještě jim nějaký zbývá, jsou vráceny do hry.

Po sérii těchto akcí vlákno simulátoru čeká na balíčky naplánovaných karet pro další kolo.

7.2 Hra na serveru

Hra na serveru po svém nastartování probíhá v následujících cyklech.

Klientům je odeslán balíček karet určený k naplánování akcí na dané kolo. Poté se čeká na přijetí balíčku naplánovaných karet od každého klienta. Když je přijat balíček karet od klienta dochází ke kontrole, zda se jedná o katry, které byly klintovi zaslané, pokud byl server spuštěn v režimu kontroly.

Když jsou přijaty balíčky od všech klientů, jsou každému klientovi zaslány balíčky naplánovaných karet všech klientů. Pokud byl server spuštěn v režimu kontroly, dochází zde k simulaci kola s danými balíčky naplánovaných karet. Dále se čeká na přijetí jednotky každého hráče od jednotlivých klientů.

Když jsou přijaty všechny jednotky od jednotlivých klientů, dochází k jejich kontrole proti jednotkám, se kterými bylo odehráno kolo na serveru, pokud je server spuštěn v režimu kontroly. Dále hra na serveru pokračuje dalším cyklem, dokud všechny jednotky nedokončí hru.

Pokud se v průběhu kontrol stane, že balíček karet nebo jednotka neprojdou kontrolou, je klient, u něhož se problém vyskytl vyřazen ze hry a odpojen od hry běžící na serveru.

7.3 Hra u klienta

Spuštěná hra v klientské aplikaci probíhá v následujících cyklech.

Aplikace čeká na přijetí balíčku karet od serveru. Když je tento balíček přijat, začíná plánovací část kola pro hráče. Tato část kola končí potvrzením naplánovaných karet hráčem nebo uplynutím časového limitu plánovací části kola. Poté je balíček s naplánovanými kartami poslán na server a čeká se na přijetí balíčků karet všech hráčů.

Ze serveru jsou přijaty balíčky naplánovaných karet všech hráčů. Tyto balíčky jsou předány do simulátoru a je s nimi odehrána druhá část kola. Po odehrání je na server poslána jednotka daného hráče. Hra pokračuje dalším cyklem dokud všechny jednotky nedokončí hru.

U klienta s umělou inteligencí probíhá stejně, ovšem karty jsou plámované programem, nikoliv uživatelem.

8. Uživatelská dokumentace

8.1 Pravidla hry

Tento návod je převzat z knihy pravidel hry RoboRally a mírně modifikován pro potřeby Monkey Bank Robbers.

V každém kole hráč získá balíček karet, ze kterého naplánuje pět karet s instrukcemi pro svou jednotku. Jak vypadá taková karta je znázorněno na obrázku 8.1. Poté, co všichni hráči naplánují karty pro své jednotky, se odehraje celé kolo hry. Odehrání kola hry spočívá v posunu jednotek po hrací ploše. Jak vypadá jednotka ukazuje obrázek 8.2, hrací plocha je na obrázku 8.4, všechny prvky, které se vyskytují na hrací ploše shrnuje obrázek 8.3. Takto hra pokračuje, dokud všichni hráči nedokončí hru. Hra se dá dokončit dvěma způsoby: Navštívením všech vlaječek na hrací ploše v pořadí jejich čísel, nebo ztrátou všech životů jednotky. Vítězem hry se stává hráč, který se svou jednotkou jako první navštíví dané vlaječky. Navštívení vlaječek není však tak snadné, jak se na první pohled zdá, protože všechny jednotky vykonávají pohyb zároveň. Jednotky se proto vzájemně pletou do cesty, odstrkují se z naplánované trasy a střílejí po sobě lasery. Občas je největší výzvou přežití.

Každé kolo je rozděleno na část plánovací a na část odehrávací a probíhá následovně:

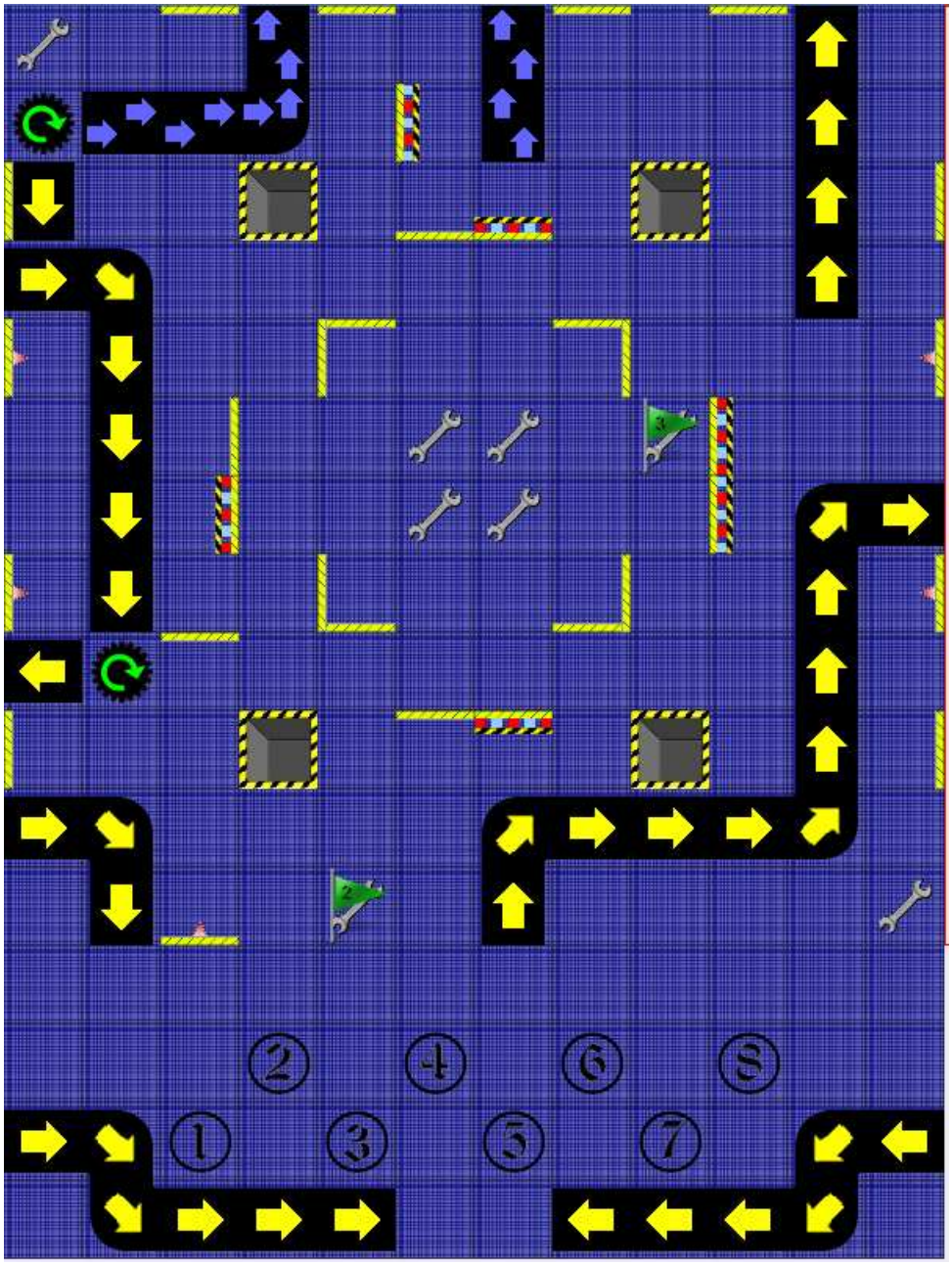
- Plánovací část
  - Hráč obdrží balíček akčních karet
  - Hráč naplánuje pět karet pro svou jednotku a potvrdí naplánování
- Odehrávací část
  - Jednotky na hrací ploše vykonají naplánované pohyby a jsou ovlivňovány prvky na hrací ploše
  - Provedou se závěrečné efekty v daném kole

Balíček karet pro dané kolo obsahuje náhodné karty. Počet karet se odvíjí od poškození jednotky, pro kterou je balíček určen. Hráč, jehož jednotka nemá žádné poškození obdrží balíček s devíti kartami. S každým poškozením jednotky

Obrázek 8.1: Karta s pohybem o dvě pole dopředu.

Obrázek 8.2: Jednotka zeleného hráče.

Obrázek 8.3: Popis prvků hrací plochy.



Obrázek 8.4: Hrací plocha.

je v balíčku o jednu kartu méně. Například hráč, jehož jednotka má 3 poškození obdrží pouze 6 karet.

Z obdrženého balíčku nabízených karet hráč vybere pět karet, kterými naplánuje pohyb pro svou jednotku v následujícím kole. Když je hráč s plánováním hotov, potvrdí svou volbu. Plánování karet je časově omezeno. Na každé naplánování karet na následující kolo má každý hráč 2 minuty. Pokud po této době není hráč s plánováním hotov, jsou karty, které nebyly naplánovány, vybrány náhodně.

Pohyb jednotek na hrací ploše je ovlivněn kartami, které mají naplánované, ale také ostatními jednotkami a prvky na hrací ploše. Každé kolo je rozděleno na pět tahů, jimž odpovídá pět naplánovaných karet. V každém tahu se provedou následující akce:

1. Jednotky vykonají pohyb naplánovaný na kartě daného tahu v pořadí daném prioritami karet
2. Prvky na hrací ploše vykonají svůj pohyb a působí na jednotky
3. Lasery na hrací ploše a lasery jednotek vypálí
4. Jednotkám stojícím na políčku, které snižuje poškození, je sníženo poškození, pokud nějaké mají

Pohyb jednotek se provede podle naplánované karty. Pokud má jednotka naplánován pohyb dopředu o 2 pole, posune se o ně ve směru, kterým je natočena. Couvání posune jednotku o 1 pole dozadu a tak dále. Priorita dané karty určuje, kdy se karta v daném tahu zahraje. Nejprve přicházejí na řadu karty s nejvyšší prioritou, jako poslední se zahrají karty s nejnižší prioritou. Když jednotka narazí do jiné jednotky pohne s ní ve směru svého pohybu, pokud se tam ovšem nenachází zeď, která pohyb nedovolí. Jednotka může být posunuta na kterékoliv políčko a dokonce může být i vytlačena z hrací plochy. Nárazy do zdi nepůsobí žádné poškození.

Pohyb prvků na hrací ploše má také své definované pořadí:

1. Expresní pásy se pohnou o 1 pole ve směru šipek na nich umístěných
2. Expresní pásy spolu s normálními pásy se pohnou o 1 pole ve směru šipek na nich umístěných
3. Odstrkovače se pohnou od zdi, pokud jsou aktivní
4. Otočná kola zarotují o 90 ° ve směru šipek na nich umístěných

Na pohyblivých pásech jsou normálně posunuty všechny jednotky, které na nich stojí. Protože tento pohyb není definován kartami, probíhá najednou. Může ovšem nastat situace, kdy více než jeden pohyblivý pás ústí na jedno políčko. Pokud na takové políčko mají být posunuty jednotky stojící na pásech zároveň, není na takové políčko posunuta žádná jednotka. Lze si to představit tak, že jednotky na daném políčku do sebe narazí a odrazí se zpět odkud byly posunuty.

Pohyblivé pásy také mohou obsahovat zatáčku o 90 °. Jednotka, která je posunuta pásem na takovéto místo, je při posunu otočena v daném směru. Pokud se ovšem na takové místo jednotka pohne buď svou naplánovanou kartou nebo jiným prvkem, než je pohyblivý pás, k otočení samozřejmě nedochází.

Střelba laserů se provede po pohybu všech jednotek a prvků na hrací ploše. Jednotky vystřelí paprsek laseru ve směru, kterým jsou momentálně natočeny. Jednotka, která je zasažena laserem získá jedno poškození za každý paprsek, který ji zasáhl. Lasery neprojdou ani roboty, ani zdmi na hrací ploše.

Odebírání poškození jednotkám se provede v případě, že jednotka stojí na konci tahu na políčku, které odebírá poškození. Jedná se o políčko se symbolem klíče. Takové jednotce je odebráno jedno poškození, pokud nějaké má.

Když je provedeno všech pět tahů a jednotka stojí na políčku s vlaječkou, která následuje v řadě navštívených vlaječek jednotkou, je tato vlaječka označena u jednotky jako navštívená. Vlaječky je potřeba navštívit v pořadí daném čísly na vlaječkách. Pokud se v dalším průběhu hry stane, že jednotka ztratí život, je tato jednotka vrácena do hry na místo poslední vlaječky, kterou jednotka navštívila.

Naplánování odstavení jednotky v následujícím kole se může provést kdykoliv před rozdáním karet na dané kolo. Pokud hráč naplánuje odstavení jednotky na další kolo, neobdrží v dalším kole žádné karty a jeho jednotka v takovém kole nebude vykonávat žádný pohyb. Když je jednotka odstavena, sníží se její poškození na nulu. Odstavená jednotka sice nevykonává žádné akční karty, ale stále na ní působí prvky hrací plochy a ostatní jednotky. Proto během tahu, kdy je jednotka odstavená, může získat nové poškození, či být dokonce zničena. V dalším kole je jednotka opět schopna pohybu a je třeba naplánovat pro ni nové karty.

Po dokončení jednoho kola se hráči odeberou naplánované kartičky, pokud nejsou uzamčeny. Pokud v průběhu kola nějaká jednotka ztratila život, je navrácena do hry na místo poslední navštívené vlaječky nebo na startovací pozici, pokud ještě žádnou vlaječku nenavštívila. je-li návratová pozice obsazena, je jednotka navrácena poblíž této pozice, na nejbližší volné políčko. Jednotka, která se takto vrátí do hry, získává automaticky dvě poškození. Jednotky, které byly odstaveny na dané kolo jsou aktivovány na kolo následující.

V průběhu hry jednotka získává poškození. S každým novým poškozením své jednotky hráč dostává o jednu akční kartu méně. Pokud jednotka získá pět poškození, každé následující poškození uzamkne jednu z naplánovaných karet.

- 0 poškození – hráč dostane 9 akčních karet.
- 1 poškození – hráč dostane 8 akčních karet.
- 2 poškození – hráč dostane 7 akčních karet.
- 3 poškození – hráč dostane 6 akčních karet.
- 4 poškození – hráč dostane 5 akčních karet.
- 5 poškození – hráč dostane 4 akční karty, plánovací pozice 5 uzamčena.
- 6 poškození – hráč dostane 3 akční karty, plán. pozice 4, 5 uzamčeny.
- 7 poškození – hráč dostane 2 akční karty, plán. pozice 3, 4, 5 uzamčeny.
- 8 poškození – hráč dostane 1 akční kartu, plán. pozice 2, 3, 4, 5 uzamčeny.
- 9 poškození – hráč nedostane akční karty, všechny plán. pozice uzamčeny.

- 10 poškození – zničení jednotky.

Pokud má hráč uzamčeny plánovací pozice, karty, které byly naplánovány zde zůstávají a nejdou nahradit. Jednotka poté tyto karty vykonává stále dokola v daném tahu každého kola, dokud nejsou pozice znovu odemčeny. Pokud dojde k uzamčení pozice s žádnou naplánovanou kartou (je-li jednotka odstavena), doplní se karta náhodně. Plánovací pozice jsou odemčeny, pokud jsou jednotce odebrána určitá poškození.

Jednotka je zničena pokud obdrží desáté poškození, pohne se na políčko s dírou, nebo vyjede mimo hrací plochu. Zničená jednotka ztrácí život a do hry se vrací, zbývý-li jí nějaký život, na úplném konci kola.

8.2 Instalace hry

Pro spuštění hry je zapotřebí mít na počítači nainstalovaný software firmy Oracle Java Virtual Machine verze 6[1]. Tento software je zdarma dostupný na adrese `www.java.com/en/download/`.

Když je Java 6 již na počítači nainstalována, stačí z CD s hrou zkopírovat složku distribuce, do vámi zvolemé složky, ve které má hra být nainstalována. Může se jednat například o složku `C:`
`Program Files` na systémech Windows nebo o složku `~/games` na systémech Unixového typu.

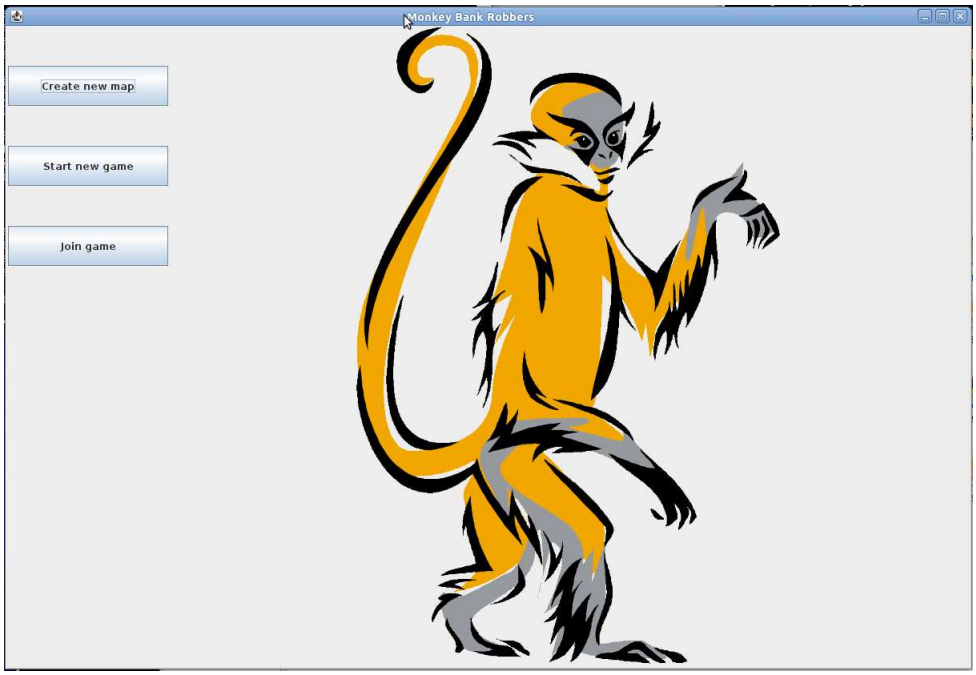
Hra se spustí z daného umístění ve složce distribuce souborem MBR.jar. Pro pohodlnější spouštění aplikace lze vytvořit odkaz na tento soubor. Vytvoření odkazu závisí na použitém systému a je popsáno v nápovědě k danému systému.

Pokud bude uživatel chtít přeložit projekt ze zdrojových kódů, je pro tento účel připraven soubor `build.xml`, který obsahuje definice základních úkolů spojených s překladem pro apache `ant`[2]. Spouští se příkazem `ant [jméno_úkolu]`. `jméno_úkolu` tvoří parametr, který specifikuje činnost programu ant. Hlavní připravené úkoly jsou: `compile` (překlad celého projektu), `compile_c` (překlad klientské části projektu), `compile_s` (překlad serverové části projektu), `java_doc` (vygenerování dokumentace programu), `jar_server` (vytvoření souboru .jar[3] serverové aplikace), `jar_client` (vytvoření .jar souboru klientské apl.), `run_server` (spuštění serverové aplikace) a `run_client` (spuštění klientské části aplikace). Celý projekt se nachází ve složce `project`.

K samostatnému překladu projektu je potřeba použít externí knihovnu `org.apache.commons.io.IOUtils`, kterou je možné zdarma získat na adrese `http://commons.apache.org/io/download_io.cgi`. Tato knihovna je také zahrnuta v projektu ve složce `ext` a `built.xml` ji při překladu zahrnuje do programu.

Obsah přiloženého CD je popsán v příloze na straně 41.

[1] Zkráceně Java 6.
[2] ant.apache.org
[3] JAR – Java ARchiv.



Obrázek 8.5: Úvodní menu.

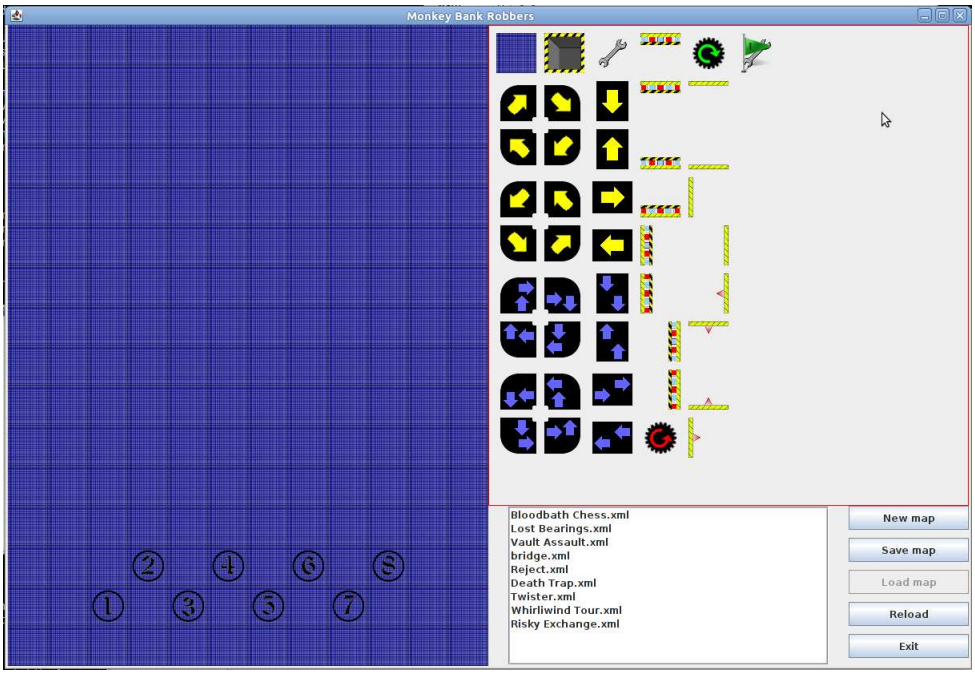
8.3 Ovládání hry

Při spuštění aplikace se uživateli zobrazí okno s úvodním menu viz obrázek 8.5. Stiskem tlačítka `Create new map` se v okně spuštěné aplikace otevře editor map. Po stisku tlačítka `Start new game` může uživatel založit na serveru novou hru. Tlačítko `Join game` po stisku umožní uživateli připojit se do založené hry.

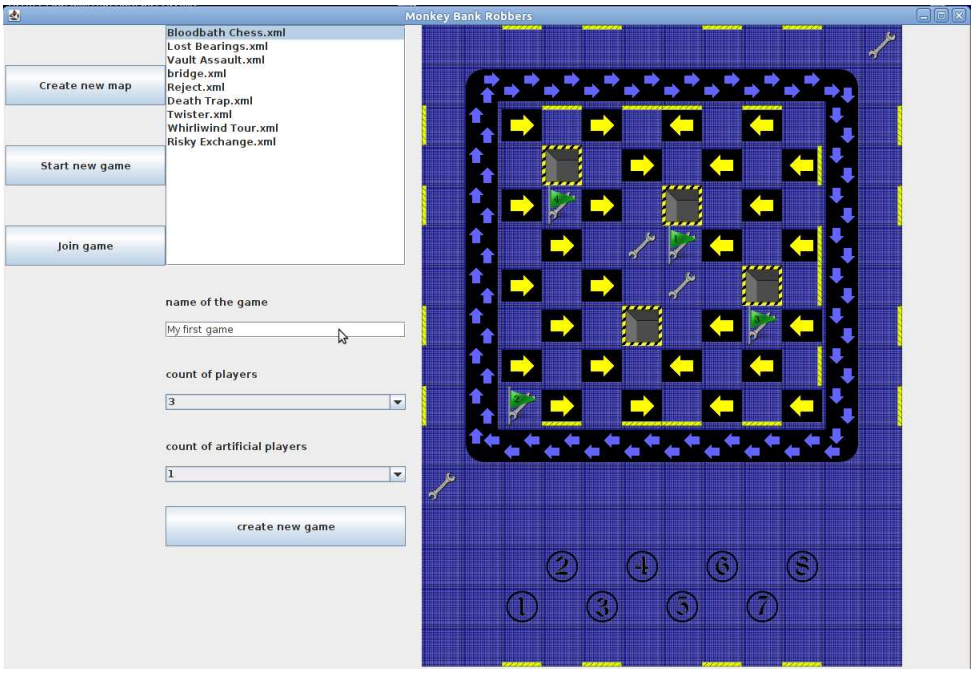
Editor map je zobrazen na obrázku 8.6. V levé části se nachází hrací plocha, která zobrazuje vytvářenou mapu. V pravé části jsou nabídnuté vrstvy, které se přetažením myší do hrací plochy vloží do políčka, na kterém jsou puštěny. Tímto způsobem je možné vytvořit novou mapu. Nová mapa se ukládá stiskem tlačítka `Save map`. Uživatel je vyzván k zadání názvu mapy a následně je mapa uložena. Tlačítko `New map` vymaže všechny prvky na hrací ploše. Nalevo od tlačítek je seznam uložených map. Vybráním jedné položky seznamu a stiskem tlačítka `Load map` dojde k načtení zvolené mapy do hrací plochy. Načtenou mapu je možné dále upravovat. Tlačítko `Reload` způsobí obnovení seznamu uložených map. Tlačítkem `Exit` opustí uživate editor map a vrátí se na úvodní obrazovku.

Zakládání nové hry znázorňuje obrázek 8.7. V seznamu dostupných map uživatel vybere požadovanou mapu. Mapa se při výběru zobrazí na hrací ploše v pravé části okna. V políčku `Name of the game` uživatel vyplní jméno vytvářené hry. Pod tímto políčkem vybere, kolik hráčů bude hra obsahovat. Ještě níže vybere, kolik z těchto hráčů bude řízeno počítačem. Stiskem tlačítka `create new game` dojde k odeslání požadavku na server o založení nové hry a program čeká na připojení do založené hry.

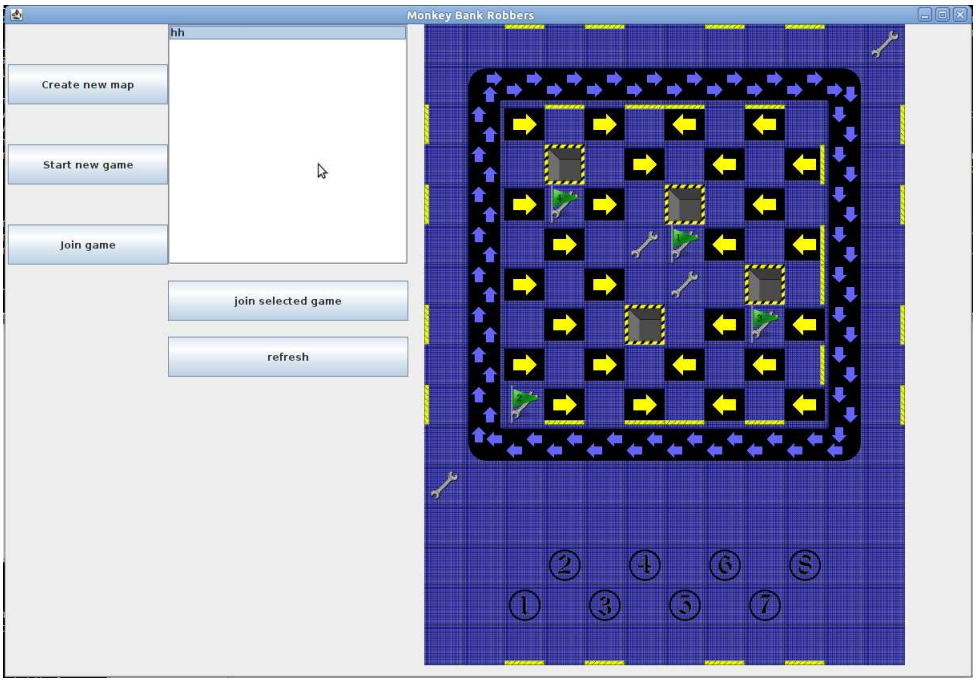
Připojit se do založené hry v případě, že hru založil někdo jiný, lze v okně pro připojování do her viz obrázek 8.8. V seznamu dostupných her uživatel vybere požadovanou hru. Mapa vybrané hry se zobrazuje na hrací ploše v pravé části okna. Tlačítkem `refresh` lze aktualizovat seznam dostupných her. Tlačítko `join selected game` způsobí odeslání požadavku o přijení do hry na server a program



Obrázek 8.6: Editor map.



Obrázek 8.7: Obrazovka pro zakládání nové hry.



Obrázek 8.8: Obrazovka pro připojování do založených her.

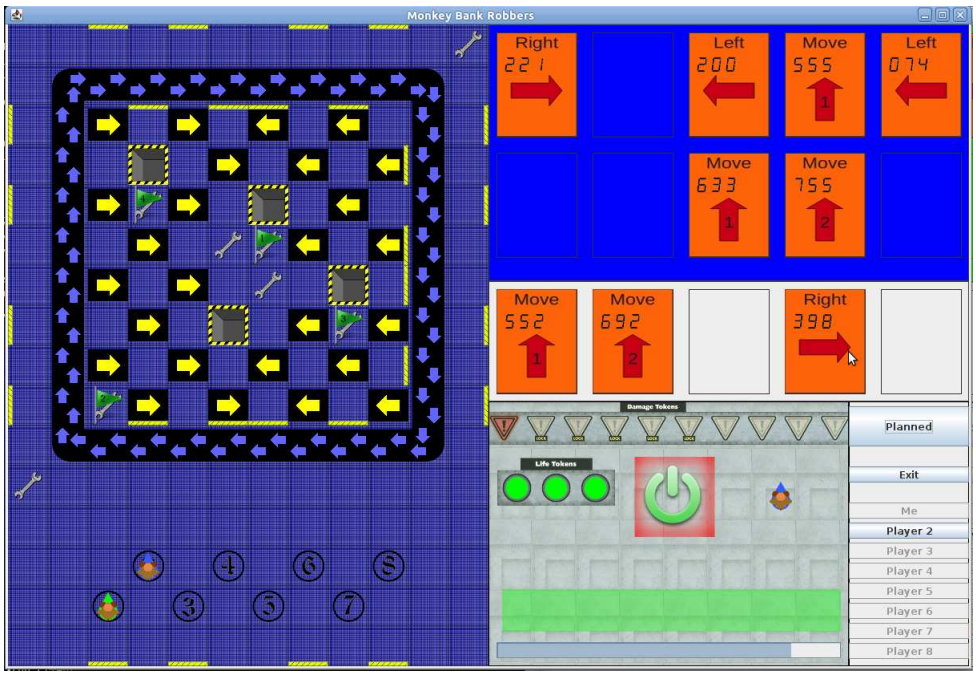
čeká na připojení.

Po připojení do hry se uživateli ukáže okno zobrazené na obrázku 8.9. V levé části okna je hrací plocha s jednotkami jednotlivých hráčů. V pravé horní části okna jsou karty, určené k plánování pohybu jednotky v daném kole. Pod těmito kartami je pět pozic pro naplánované karty. Karty z obou částí může uživatel libovolně přesouvat mezi sebou myší. Pod naplánovanými kartami je okénko s informacemi o vybrané jednotce. Okénko ukazuje poškození jednotky v podobě žlutých trojúhelníčků s vykřičníkem, počet životů jednotky v podobě zelených koleček, navštívené vlaječky se zobrazují v zeleném pásu. Jednotka jejíž informace jsou zobrazeny je v okénku také nakreslena. Modrý proužek dole v okénku odměřuje zbývající čas na naplánování karet.

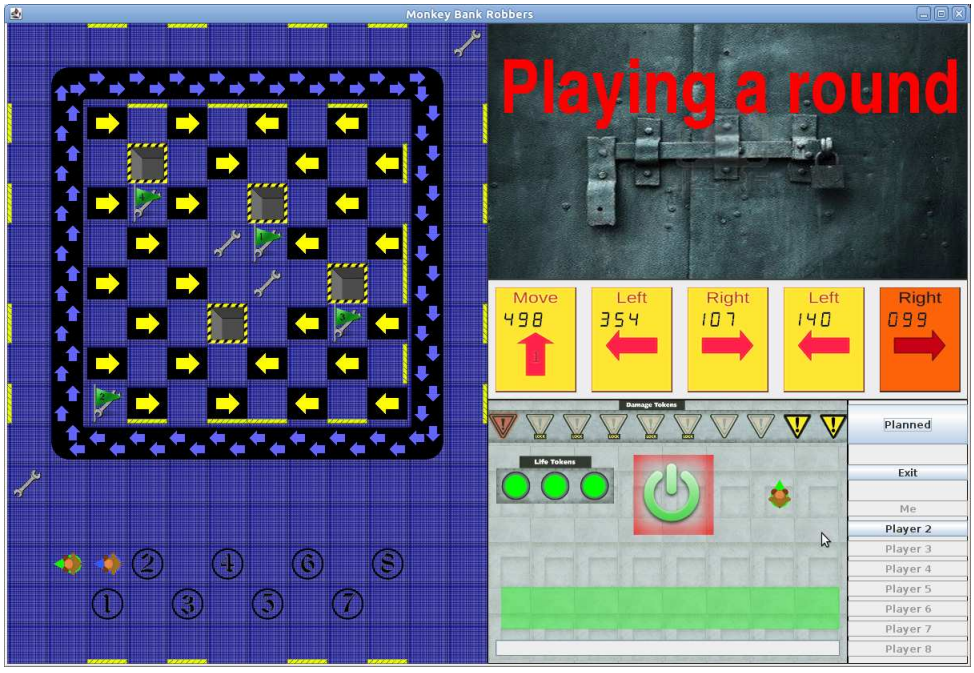
Pomocí symbolu vypínače může hráč naplánovat odstavení jednotky na následující kolo. Zelený symbol znamená, že odstavení není naplánováno. Modrý symbol značí naplánované odstavení. Červený symbol značí, že je jednotka v tomto kole odstavena.

Tlačítkem `Planned` hráč potvrdí své naplánované karty. Tlačítko `Exit` slouží k opuštění rozehrané hry a vrátí uživatele na úvodní obrazovku programu. Zbylá tlačítka slouží pro přepínání zobrazení informací mezi jednotlivými jednotkami ve hře.

Jak vypadá odehrávací část kola je ukázáno na obrázku 8.10. V této části kola hráč čeká, až se vykonají všechny naplánované pohyby. Karty, které jsou odehrány jsou zvýrazněny, aby hráč přesně věděl, ve kterém tahu se kolo nachází. Když se odehraje celé kolo, opět přichází na řadu plánovací část kola nového. Takto hra pokračuje, dokud všichni hráči hru nedokončí.



Obrázek 8.9: Plánování karet ve hře.



Obrázek 8.10: Odehrávání kola ve hře.

Závěr

Srovnání s obdobnými projekty

Výhody Monkey Bank Robbers: vytváření libovolných map, hraní i v případě, že zrovna není k dispozici fyzický protivník, hra s protivníky, kteří se nacházejí kdekoli, pokud mají počítač a připojení k Internetu.

Nevýhody Monkey Bank Robbers: velikost hrací plochy nelze rozšířit, absence speciálních karet, nutnost používání počítače s připojením k Internetu.

Jak vypadá hra RoboRally [1] ukazuje obrázek 8.11.

Výhody: zvětšování hrací plochy skládáním jednotlivých hracích desek, možnost upravovat pravidla jednotlivých her a předepsané scénáře her.

Nevýhody: jednotlivé mapy jsou pevně dány potiskem hracích desek, nutnost přítomnosti protihráčů, potřeba vlastnit tuto hru.

Jak vypadá hra RoboRally, která je dostupná on-line [3] ukazuje obrázek 8.12.

Výhody: věrná podoba s deskovou hrou RoboRally, hraní i v případě, že zrovna není k dispozici fyzický protivník, hra s protivníky, kteří se nacházejí kdekoli, pokud mají počítač a připojení k Internetu.

Nevýhody: nutnost registrace a poplatek, při hraní s fyzickými protihráči, bez registrace je dostupné pouze demo hry na přednastavených mapách proti počítačovému protihráči, chybí zde možnost vytvářet vlastní mapy a používat speciální karty.

Budoucnost projektu

Aplikace by se měla v budoucnu dočkat nové grafiky, která bude věrnější předloze, a také druhé grafiky, která bude obsahovat jako jednotky opice a hrací plocha bude stylizována do prostředí banky.

Dále by se měla zvýšit rozšiřitelnost programu tak, aby bylo možné přidávat i nově definované kartičky pohybu a aby hra obsahovala speciální kartičky.

Může být také využit prostor, který program vytváří k definování nových zajímavých políček hry a k implementaci kvalitní umělé inteligence.

Obrázek 8.11: Desková hra RoboRally.

Obrázek 8.12: On-line verze hry RoboRally.

Seznam použité literatury

[1] Originální hra RoboRally,
http://en.wikipedia.org/wiki/RoboRally

[2] Java^{TM}Platform, Standard Edition 6,
http://download.oracle.com/javase/6/docs/api/

[3] On-line verze hry RoboRally,
www.gametableonline.com/welcome.php

[4] Extensible Markup Language (XML),
www.w3.org/XML/

[5] XML Tutorial,
www.w3schools.com/xml/

[6] Simple API for XML,
en.wikipedia.org/wiki/Simple_API_for_XML

Přílohy

Definice základních vrstev

```
<BasicLayers>
    <Layer>
        <phase>0</phase>
        <damage>0</damage>
        <move>NO</move>
        <rotation>NO</rotation>
        <wall>NO</wall>
        <laser>NO</laser>
        <serial_number>0</serial_number>
        <special></special>
        <name>background</name>
        <image>
            <image_source>background</image_source>
            <image_animation>0</image_animation>
            <image_flip>false</image_flip>
            <image_rotation>0</image_rotation>
        </image>
    </Layer>
    <Layer>
        <phase>0</phase>
        <damage>0</damage>
        <move>NO</move>
        <rotation>NO</rotation>
        <wall>NO</wall>
        <laser>NO</laser>
        <serial_number>1</serial_number>
        <special>repair</special>
        <name>flag_1</name>
        <image>
            <image_source>flag_1</image_source>
            <image_animation>0</image_animation>
            <image_flip>false</image_flip>
            <image_rotation>0</image_rotation>
        </image>
    </Layer>
    ...
</BasicLayers>
```

Definice pořadí vrstev v mapovém editoru

```
<order>
    <layer>background</layer>
    <layer>belt_se</layer>
    <layer>belt_en</layer>
    <layer>belt_es</layer>
    <layer>belt_ne</layer>
    <layer>fbelt_se</layer>
    <layer>fbelt_en</layer>
    <layer>fbelt_es</layer>
    <layer>fbelt_ne</layer>

    <layer>pit</layer>
    <layer>belt_ws</layer>
    <layer>belt_nw</layer>
    <layer>belt_sw</layer>
    <layer>belt_wn</layer>
    <layer>fbelt_ws</layer>
    <layer>fbelt_nw</layer>
    <layer>fbelt_sw</layer>
    <layer>fbelt_wn</layer>

    <layer>wrench</layer>
    <layer>belt_ns</layer>
    <layer>belt_sn</layer>
    <layer>belt_we</layer>
    <layer>belt_ew</layer>
    <layer>fbelt_ns</layer>
    <layer>fbelt_sn</layer>
    <layer>fbelt_we</layer>
    <layer>fbelt_ew</layer>

    <layer>hammer</layer>
    <layer>pusher_24_n</layer>
    <layer>pusher_135_n</layer>
    <layer>pusher_24_s</layer>
    <layer>pusher_135_s</layer>
    <layer>pusher_24_w</layer>
    <layer>pusher_135_w</layer>
    <layer>pusher_24_e</layer>
    <layer>pusher_135_e</layer>

    <layer>gear_l</layer>
    <layer>gear_r</layer>
    <layer>wall_n</layer>
</order>
```

Textová podoba objektu třídy Card

```
<card>
    <movement>FORWARD</movement>
    <rotation>NO</rotation>
    <distance>2</distance>
    <priority>745</priority>
</card>
```

Textová podoba objektu třídy CardPackage

```
<card_package>
    <id>2</id>
    <card>
        <movement>FORWARD</movement>
        <rotation>U_TURN</rotation>
        <distance>0</distance>
        <priority>52</priority>
    </card>
    <card>
        <movement>FORWARD</movement>
        <rotation>NO</rotation>
        <distance>1</distance>
        <priority>520</priority>
    </card>
    <card>
        <movement>FORWARD</movement>
        <rotation>LEFT</rotation>
        <distance>0</distance>
        <priority>232</priority>
    </card>
    <card>
        <movement>FORWARD</movement>
        <rotation>RIGHT</rotation>
        <distance>0</distance>
        <priority>257</priority>
    </card>
    <card>
        <movement>FORWARD</movement>
        <rotation>RIGHT</rotation>
        <distance>0</distance>
        <priority>158</priority>
    </card>
    <card>
        <movement>BACKWARD</movement>
        <rotation>NO</rotation>
        <distance>1</distance>
        <priority>439</priority>
    </card>
    <card>
        <movement>FORWARD</movement>
        <rotation>NO</rotation>
        <distance>1</distance>
        <priority>560</priority>
    </card>
</card_package>
```

Textová podoba objektu třídy Environment

```
<environment>
    <capacity>0</capacity>
    <ai>0</ai>
    <name_of_the_game>null</name_of_the_game>
    <name_of_the_map>Whirliwind Tour</name_of_the_map>
    <id_of_the_game>0</id_of_the_game>
        <croft>
            <layer>background</layer>
            <layer>wrench</layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
        </croft>
        <croft>
            <layer>background</layer>
            <layer>belt_ns</layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
        </croft>
        <croft>
            <layer>background</layer>
            <layer></layer>
            <layer>pusher_24_n</layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
        </croft>
        ...
        <croft>
            <layer>background</layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
            <layer></layer>
        </croft>
</environment>
```

Objekt třídy unit v textové podobě

```
<unit>
    <id>1</id>
    <color>0</color>
    <position>
        <x>2</x>
        <y>14</y>
    </position>
    <direction>NORTH</direction>
    <checkpoint>0</checkpoint>
    <checkpoint_position>
        <x>2</x>
        <y>14</y>
    </checkpoint_position>
    <active>true</active>
    <damage>0</damage>
    <power_down>false</power_down>
    <life>3</life>
    <finish>0</finish>
</unit>
```

Obsah přiloženého CD

Na přiloženém CD se nachází následující obsah.

- bakalarska-prace_Monkey_Bank_Robbers_Dominik-Skoda_2011.pdf
  - elektronická verze tohoto textu
- distribution
  - zkompilovaný program klientské aplikace
- documentation
  - programová dokumentace vytvořená programem javadoc
- project
  - zdrojové kódy a zdroje celého projektu