Síťová hra - Lodě

KIV/UPS - Semestrální práce

student:	Petr Vondrovic
studijní číslo:	A20B0273P
email:	vondrp@students.zcu.cz
datum:	06. 11. 2022

1 Pravidla hry

Lodě, popřípadě Námořní bitva je hra pro dva hráče, ve které je cílem potom všechny lodě soupeře. Oba hráči mají vlastní herní pole o velikosti 10x10 na němž jsou rozmístěny jejich lodě různých velikostí.

Lodě mohou být rozmístění pouze vodorovně nebo horizontálně. Je tedy vyloučeno umístění úhlopříčné nebo jinak šikmé. Lodě se navzájem nesmí překrývat a vzájemně se dotýkat a to ani rohem. Během hry se pozice lodí nemění.

Při hře se účastnící střídají ve „střelbě“ na jednotlivá pole protivníka, který oznámí zda došlo nebo nedošlo k zásahu, popřípadě zda byla loď zásahem potopena. Byla-li loď potopena je její okolí označeno jako minutí.

Hra končí, když jeden z hráčů přijde o celou flotilu.

2 Implementace programu

Kromě implementace hry podle specifikovaných pravidel obsahuje hra systém místností. Místnost vytvoří jeden z hráčů a jakmile se k němu připojí protivník hra započne. Každá místnosti má své identifikační číslo a hráči mohou získat jejich seznam, který obsahuje jména hráčů v místnosti.

Dojde-li ke ztrátě spojení nebo odpojení se, je stav hry a klienta stále k dispozici a uživatel se může připojit zpět.

Server si mapuje čísla deskriptoru ke klientovi, aby dokázal identifikovat kontext uživatele.

3 Komunikační protokol

Komunikační protokol je nešifrovaný a textový. Je tak zajištěna jeho srozumitelnost pro vývoj, testování a kontrolu. Zprávy jsou dělené na 4 druhy a to žádosti, potvrzovací, datové a chybové. Žádosti odesílá klient a přijíma server, zatímco u ostatních typů zpráv je to naopak.

Zprávy protokolu využívají k dělení jednotlivých parametrů zpráv středník (;). Každá zpráva je na samostatném řádku, jsou tedy zakončené znakem (\n);

3.1 Stav uživatele

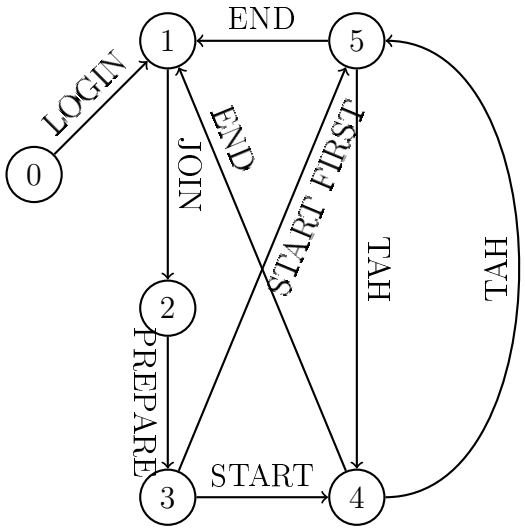
Klient se může nacházet v jednom z několika stavů. Na základě stavu se odvozeno, které akce mohou být prováděny.

Seznam stavů:

- 0 - Nepřihlášený
- 1 - V lobby
- 2 - V místnosti
- 3 - Ve hře - připravuje hrací plochu
- 4 - Ve hře - čeká
- 5 - Ve hře - na rahu

3.2 Stavový diagram

Všechny stavy se mohou dostat do nepřipojeného/nepřihlášeného stavu.



Obrázek 1: Stavový diagram

3.3 Chybové kódy

1. Interní chyba
2. Špatný formát zprávy
3. Špatný formát uživatelského jména
4. Uživatelské jméno je zabrané
5. Neplatný stav uživatele
6. Místnost je plná
7. Mimo rozsah herního plánu
8. Už zasažené pole
9. Neplatné umístění lodě
10. Neplatný identifikátor lodě

3.4 Skupiny zpráv

3.4.1 Navázání spojení

„connected“ zpráva serveru zaslaná při úspěšném navázání spojení klienta se serverem.
Zpráva serveru:
`connected`

3.4.2 Připojení

Po úspěšném navázání spojení se klient může přihlásit. Přihlašování probíhá příkazem „login_req;(name)“. Server může odpovědět buď s „login_ok;(name);(state)“ nebo s „login_err;(error_code)“. Na základě stavu se hráč připojí buď do lobi nebo hry/místnosti z níž odešel (při spadnutí klienta).
Požadavek klienta:
`login_req;(name)`
Možné odpovědi serveru:
`login_ok;(name);(state)`
`login_err;(error_code)`

`(name)` - jméno, kterým se klient přihlašuje
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.3 Vytvoření místnosti

Žádost o vytvoření místnosti se provádí pomocí zprávy „room_create_req“. Server může vytvoření místnosti buď potvrdit nebo ohlásit chybu.
Požadavek klienta:
`room_create_req`
Možné odpovědi serveru:
`room_create_ok`
`room_create_err;(error_code)`

`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.4 Seznam místností

Pro připojení do místnosti je třeba získat její identifikační číslo na nějž se klient ptá příkazem „room_list“ a je mu odpovězeno zprávou „room_list_data;(id1);(name1);...“ za níž se nachází dvojice identifikátoru místnosti a hráče, který v ní čeká.
Požadavek klienta:
`room_list`
Možné odpovědi serveru:
`room_list_data;(id1);(name1);(id2);(name2)...`
`room_list_err|(error_code)`

`(idX)` - identifikátor místnosti - nezáporné celé číslo
`(nameX)` - jméno uživatele, který je aktuálně v místnosti
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.5 Připojení do místnosti

S identifikátorem místnosti se klient může přihlásit příkazem „room_join_req;(room_id)“.
Povedlo-li se připojení dostanu klient zprávu s jménem protivníka „room_join_ok;(opp_name)“.
Hráč čekající v místnosti dostane zprávu s jménem připojeného uživatele „room_join_opp;(name)“.
Požadavek klienta:
`room_join_req;(id)`

Možné odpovědi serveru:
`room_join_ok;(opp_name)`
`room_join_err;(error_code)`
Zprávy protihráči:
`room_join_opp;(name)`

`(id)` - identifikátor místnosti - nezáporné celé číslo
`(opp_name)` - jméno protihráče, který je v místnosti
`(name)` - jméno připojujícího protihráče
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.6 Odejít z místnosti

Z místnosti nebo ze hry se dá odejít příkazem „room_leave_req“. Server odpoví potvrzením nebo s chybou „room_leave_err;(error_code)“. Je-li ve hře nebo v místnosti oponent, dostane zprávu o odpojení protihráče „room_leave_opp“. Odešel-li hráč během hry, hra skončí. **Požadavek klienta:**
`room_leave_req`
Možné odpovědi serveru:
`room_leave_ok`
`room_leave_err;(error_code)`
Zprávy protihráči:
`room_leave_opp;(name)`

`(name)` - jméno odpojeného protihráče
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.7 Vstup do hry

Po té co se do místnosti připojí oba hráči dostanou od serveru zprávu „game_conn“. Tato zpráva značí, že oba hráči nyní přejdou do fáze přípravy hry. **Zpráva serveru:**
`game_conn`

3.4.8 Herní plocha připravena

Po vstupu do hry oba hráči v rámci klienta rozmístí své lodě. Po dokončení rozmisťování lodí pošlou serveru zpráv, že jsou připraveni spolu s se svou hrací plochou. Plocha je ve formě řetězce tvořen třemi typy znaků – E (prázdno), číslice v rozsahu 0 až 6 jakožto symboly jednotlivých lodí a znak čárky dělící řádky (,). Příkaz je „game_prepared;(EEE00EEEEE,...)“. Je-li vše v pořádku odpoví server se zprávou „game_prepared_ok“, při chybě pošle „game_prepared_err;(e

Jsou-li obě hrací plochy připraveny pošle navíc jednomu z hráčů zprávu „game_play“ značící začátek hry.
Požadavek klienta:
`game_prepared;(EEE00EEEEE,...)`
Možné odpovědi serveru:
`game_prepared_ok`
`game_prepared_err;(error_code)`
`game_play`

`(name)` - jméno odpojeného protihráče
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

Poznámka: Informace o lodí jsou zaslány v podobě jejich identifikátorů, protože ten pomáhá serveru a klientovi označit okolí lodě při jejím zničení.

3.4.9 Střelba

Příkaz vystřelit na danou pozici „game_fire_req;(x);(y)“. Bylo-li vystřeleno na pozici v platném rozsahu, která ještě zasažena nebyla dostane klient od serveru zprávu spolu s informací v jakém stavu je zasažená pozice a jestli při zásahu byla zničena loď „game_fire_ok;(x),(y),(status)
V opačném případě dostanou zprávu chybovou „game_fire_err;(error_code)“. Při úspěchu dostane protihráč informaci o zasaženém poli „game_opp_fire;(x);(y);(status);(ship_destroyed)“.
Požadavek klienta:
`game_fire_req;(x);(y)`
Možné odpovědi serveru:
`game_fire_ok;(x),(y),(status);(ship_destroyed)`
`game_fire_err;(error_code)`
Zprávy protihráči:
`game_opp_fire;(x);(y);(status);(ship_destroyed)`

`(x)` - x-ová souřadnice
`(y)` - y-ová souřadnice
`(status)` - stav zasaženého pole - H (loď zasažena), M (minul)
`(ship_destroyed) - 0 - loď žije, 1 - loď zničena`
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

3.4.10 Informace o hře

Pro potřeby znovu připojení nebo synchronizace dat při chybě může klient vyslat žádost „game_info_req“. Server vrátí chybovou zprávu „game_info_err;(error_code)“ nebo herní data obsahující stav klientova a protivníkova herního pole „game_info_data;(client_status);(client_board);(opp_name);(opp_board)“.
Požadavek klienta:
`game_info_req`
Možné odpovědi serveru:
`game_info_data;(client_status);(client_board);(opp_name);(opp_board)`
`game_info_err|(error_code)`

`(client_status)` - status klienta (viz. Stavový diagram)
`(opp_name)` - jméno protihráče
`(client_board)` - řetězcová reprezentace klientova hracího plánu. Obsahuje symboly `E` - prázdné pole, `H` - zasažená loď, `M` - zasažené prázdné pole a číslice od `0` do `6` symbolující jednotlivé lodě.
`(opp_board)` - řetězcová reprezentace protivního hracího plánu. Obsahuje symboly `E` - prázdné pole, `H` - zasažená loď a `M` - zasažené prázdné pole.
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

Poznámka: U `(opp_board)` a `(client_board)` nejsou jednotlivé řádky odděleny čárkou (,) jako v případě přípravy hry (viz. Herní plocha připravena). Je to tak z důvodu lehčího zpracování herního plánu do řetězcové podoby na straně serveru.

3.4.11 Ukončení hry

Zprávu zasílá server klientům s parametrem jména vítěze.
Zpráva serveru:
`game_end;(winner_name)`

`winner_name` - jméno vítěze

3.4.12 Odpojení

Příkaz s žádostí o odpojení od serveru je „logout_req“. Server odpoví buď s „logout_ok“, případně s chybou „logout_err;(error_code)“.
Požadavek klienta:
`logout_req`
Možné odpovědi serveru:
`logout_ok`
`logout_err;(error_code)`
Zprávy protihráči:
`room_leave_opp;(name)`

`(name)` - jméno odpojeného spoluhráče
`(error_code)` - identifikační číslo chyby (viz. Chybové kódy)

4 Implementace serveru

Server je implementovaný v programovacím jazyce C. Pro realizace paralelního serveru je použita funkce `select`, zbytek serveru je realizován v jednom vlákně. Server je tvořen strukturama pro jednotlivá připojení a data ukládá do datových struktur. Například jednolitvé místnosti/hry mají přiřazené identifikátor, kterým jsou rychle získatelné z array listu. Uživatelé jsou ukládány v hashmapě, kde je lze rychle vyhledat pomocí jejich jména.

Informace o činnostech serveru jsou zasílána na výstup a do souboru `trace.log` se ukládají trasovací informace. Po ukončení serveru za pomocí signalu (např. CTRL + C) je vytvořen soubor `stats.txt`, kde jsou uloženy statistické údaje o činnosti serveru.

4.1 Popis složek a jednotlivých souborů

4.1.1 main.c/h

Soubor funkcí se stará o vstup do programu inicializaci základních struktur, a procesování jednotlivých zpráv.

4.1.2 Složka structures

Ve složce se nachází výše zmíněné datové struktury array listu a hashmapy.

4.1.3 Složka game

Ve složce game se nachází soubory `ship.c/h` a `shipsGame.c/h`. Ty slouží jakožto modely lodí a hry/místnosti a funkce sloužící k jejich nastavení.

4.1.4 Složka communication

Složka obsahuje soubory starající se o obsluhu připojení.

`server.c/h` se stará o udržování informací napříč dalšími funkcemi, obsluhuje nová připojení a odpojuje ty ukončující.

Soubor `client.c/h` obstarává nastavení klienta.

`commands.c/h` obsahuje seznam všech očekávajících příchozích zpráv a jejich obsluhu.

V souboru `errors.h` se nachází přehled definovaných chybových stavů.

5 Implementace klienta

Klient je implementovaný v jazyce Java na grafické platformě JavaFX.

5.1 Rozdělení

Aplikace je implementovaná pomocí MVC modelu (model, view, controller). View vrstva se nachází ve složce resources a skládá se z fxml souborů jednotlivých oken aplikace. Ve vztahu 1 ku 1 se ve složce controller se nachází kontrolery oken, které se starají o propojení pohledů s modely.

Seznam oken:

- Connection - okno sloužící k navázání spojení se serverem a připojení se
- Lobby - lobby, ve kterém mohou uživatelé podat žádost o vytvoření nové místnosti nebo se připojit do místnosti jiného hráče
- Room - místnost, v níž hráč po jejím vytvoření čeká než se připojí protihráč.
- Game - okno v němž se odehrává hra. V horní části se nachází jméno hráče a protihráče společně s tlačítkem Ready, sloužící k odeslání připraveného hracího pole. V pravé části okna se nachází textová oblast v níž se nachází zprávy o průběhu hry (+ připojení/odpojení protihráče a další informace). Ve středu obrazovky leží vlevo hráčovo herní plocha a vpravo od ní plocha protihráče.
- Result - okno zobrazující se po skončení hry obsahující jméno vítěze.

Modely jsou rozdělené do modelů připojení a hry.

5.2 Modely připojení

5.2.1 Reciever

Třída reprezentuje samostatné vlákno, uzavřené v nekonečné smyčce. Vlákno přijímá zprávy ze serveru. A posílá je na zpracování třídě `MessageHandler`.

5.2.2 MessageHandler

Instance třídy obsluhuje jednotlivé možné zprávy ze serveru a vykonává jejich obslužné akce. Pokud počet po sobě jdoucích zpráv, které neodpovídají žádné zprávě, překročí nastavenou mez, klient se od serveru odpojí.

5.2.3 Ostatní

- ConnectionModel - obsahuje Socket použitý k připojení a ke komunikaci se serverem
- Room - instance místnosti
- Stats - přepravka statistik o připojení (počet poslaných/přijatých zpráv a bajtů)

5.3 Modely hry

5.3.1 GameModel

Obsahuje data herních ploch, informaci o vítězi a současný stav hry. Nachází se zde metody sloužící k inicializaci herních plánů, rozmístění lodí a obstarání zasažení pole herního plánu.

5.3.2 GameBoard

Grafická reprezentace herního plánu zobrazující data z `GameModel`.

5.3.3 Ostatní

- Position - přepravka 2D pozice
- GameStatus - enum možných stavů hry, každý stav má přiřazené identifikační číslo
- SquareStatus - enum stavů jednotho čtverce herního pole
- ShipType - enum typu lodí obsahující jejich možné různé délky
- Player - reprezentuje hráče
- Ship - reprezentace lodě uchovávající pozici prvního políčka lodi, informaci je-li loď položena vertikálně, zda je s lodí manipulováno a její identifikační číslo (id).
- Square - datové reprezentace jednotlivých čtverců, ze kterých se skládá herní plocha. Obsahuje své souřadnice, stav `SquareStatus` a popřípadě odkaz na loď `Ship` nacházející se v poli.

5.4 Hlavní a pomocné třídy

Zbývající třídy jsou buď pomocné nebo hlavní/řídící třídy aplikace.

5.4.1 App

Třída reprezentující grafickou aplikaci, zároveň se jedná o třídu s návrhovým vzorem jedináček, má tedy přístupnou jednu svou instanci. Instance shlukuje odkazy na jednotlivé moduly aplikace. Server je doplněn o sbírání stručných statistik přenosů. Statistika je přístupná z pohledu lobby.

5.4.2 Ostatní

- Main - spouštěcí třída aplikace, která volá `App`
- SceneEnum - enum jednotlivých možných scén (oken) aplikace
- AlertFactory - třída užívaná k posílání chybových zpráv

6 Uživatelská dokumentace

6.1 Překlad serveru

Pro překlad serveru je potřeba nástroj `cmake`, `make` a překladač `gcc`. Nejprve vygenerujeme soubor makefile zadáním příkazu `cmake -B ./build/`. Přejdeme do adresáře `build` příkazem `cd build` a v něm spustíme překlad programu příkazem `make`. Po dokončení operace by měl být vytvořen v adresáři `build` soubor `server`.

6.2 Spuštění serveru

Pro spuštění serveru potřebujeme nejprve znát číslo portu služby. Řekněme, že zvolíme číslo `9123`. Spuštění serveru provedeme tak, že v adresáři `build` zadáme příkaz `./server 9123`. Server by se měl spustit a potvrdit to první vypsanou zprávou.

6.3 Překlad a spuštění klienta

Klient lze spustit skrz službu `gradle`, kdy se program automaticky přeloží, sestaví a spustí příkazem `gradle run`.

Případně můžeme nechat vygenerovat spustitelný soubor JAR příkazem `gradle jar`. Výsledný JAR soubor se uloží do adresáře `build/libs/` pod názvem `UPS-SP-Client-1.0.jar`. Tento soubor můžeme spustit příkazem `java -jar UPS-SP-Client-1.0.jar` (pokud máme nainstalované potřebné knihovny a správně nastavené cesty).

7 Závěr

Server s klientem spolu síťově komunikují na popsaném protokolu a zároveň je přes něj realizována síťová hra. Programy reagují na výpadky a jsou stabilní. Avšak v práci se vyskytují drobné ústupky v optimalizaci či původní koncepci, které byly způsobeny zejména nevhodným časovým rozvržením práce v kontextu s ostatními činnostmi studenta. Každopádně požadavky na práci jsou splněny.