FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

KIV/UPS

ONLINE HRA - PONG!

Jakub Vítek (viteja) - A19B0222P

Leden 2020

Obsah

1 Hra

Původní Pong byla jedna z nejstarších her. Jednalo se o tenisovou 2D hru, která byla původně vydána společností Atari v roce 1972.

Hratelnost této hry je velice jednoduchá. Dva hráči na opačných stranách ovládají plošiny, kterými odráží míček. Ve chvíli kdy jednomu z hráčů propadne míček, druhý hráč získává bod. Míček se odraží od stěn a od plošin hráčů, v případě odražení od hráčské plošiny je míček navíc mírně zrychlen.

2 Implementace

2.1 Server

Serverová část byla implementována v jazyku go od společnosti Google. Důvodem pro výběr tohoto jazyka bylo dynamické typování, automatická správa paměti, pohodlnější práce s řetězcem než v jazyce C a jednoduchost paralelizace.

2.1.1 Komunikace

Komunikace serveru je řešena samostatným vláknem, které řeší základní tvorbu socketů a čtení dat z nich. Server využívá hlavního socketu pro připojení nových klientů. Při připojení nového klienta se vždy vytvoří nový socket, přes který je s klientem komunikováno.

Komunikace s vícero sockety v jednom vlákně je řešena systémovým voláním operačního systému GNU/Linux *select*, ktere naplní předpřipravenou sadu socketů informací o tom, zda některý ze socketů komunikuje. Následně jsou všechny komunikující sockety postupně obslouženy. Při vytvoření nového spojení (socketu) se na serveru vytvoří a uloží do mapy struktura *Client*, tato struktura také obsahuje buffer přijatých dat. Obsluhou klientského socketu je pak myšleno přečtení příchozích dat a jejich uložení do toho bufferu.

Pro každý vytvořený klient je spuštěn dekodér (jehož funkce je popsána v sekci protokol) jehož prací je čtení dat uložených v bufferu a jejich překlad na zprávy a jejich uložení do bufferu pro zprávy.

2.1.2 Zprávy a akce

Při startu server se vytváří *Manager*, jehož běh probíhá v samostatném vlákně. Při inicializaci této struktury se registrují akce do mapy akcí, kde klíčem je typ zprávy a hodnotou odkaz na funkci, která má být spuštěna při přijetí tohoto typu zprávy. Typ zprávy je možné přečíst z hlavičky zprávy. Ve chvíli, kdy zpráva na serveru existuje, je již ověřeno, že má validní hlavičku a formát, neověřuje se však její obsah.

Na serveru jsou zprávy dvojího typu, zprávy herní a kontrolním. Kontrolní zprávy obvykle řeší akce jako například registraci či jinou správu uživatele či hry (třeba její založení či opuštění). U herních zpráv je ověřeno, zda je uživatel registrován a splňuje všechny požadavky na přesun zprávy do bufferu

herního serveru. V případě, že zpráva požadavky nesplňuje, je vrácena uživateli chybová správa se stejným typem.

2.1.3 Herní server

Herní server je běžící smyčkou ve samostatném vlákně. Běžící server může být ve stavu *PAUSED* či *UNPAUSED*. Uživatel není schopný pauzu sám o sobě vyvolat. Server je ve stavu pauzy v případě, že je ve hře připojen jeden hráč či jeden z hráčů ztratil spojení.

Smyčka herního serveru proběhne třicetkrát za sekundu. Nejprve se ověřuje, zda jsou všichni uživatelé připojení a nemají ztracené spojení - ztráta spojení je detekována jako prodleva větší jak 2 vteřiny od poslední zprávy. Následně jsou přijaty všechny uživatelské zprávy změny hráčské pozice. U těchto zpráv je provedena korekce a ověřování před zapsáním nové pozice.

V dalším kroce je proveden tick míče, aktualizace jeho pohybu, detekce kolizí. Následně je vytvořena zpráva s aktuálním stavem hry, která je odeslána připojeným hráčům. V poslední řade je ověřován konec hry. V případě, že nějaký z hráčů dosáhne 10 bodů, je herní smyčka ukončena a hráči jsou informování o výhře. Po ukončení hry je hra smazána a vlákno ukončeno.

2.2 Klient

Implementace herního klienta je provedena v jazyku Python (verze minimálně 3.7). Klient je realizován za pomocí multiplatformní sady *pygame*. Všechna herní menu byla vytvořena za použití knihovny PyGameMenu. Alert boxy byly vytvořeny za využití modulu tkinter.

Při spuštění herního klienta je uživatel přivítán menu pro připojení. Zde je možné připojit se k běžícímu serveru po vyplnění IPv4 adresy, portu, herní přezdívky. Připojení vytváří nového uživatele. Znovupřipojení požádá server o přihlášení k již existujícímu uživatelskému účtu. Při registraci či připojení k serveru klient aktivně čeká maximálně dvě sekundy na odpověd od serveru, pokud mu v této době odpověd nepříjde nebo přijde záporná, odpojí se od serveru (pokud bylo spojení navázáno)

Po přihlášení se uživatel dostane do menu s lobby, kde může tlačítky vytvořit novou hru, obnovit seznam existujících her či se připojit k nějaké hře k seznamu. Po připojení uživatel začne posílat KeepAlive zprávy, které server informují od tom, že je spojení v pořádku. Odpověd na tyto zprávy také ujišťuje uživatele, že je stále připojen k serveru. Poslední možností uživatele v tomto menu je informování serveru o tom, že se klient bude odpojovat (validní ukončení spojení). Při ukončení spojení nebo jeho ztrátě je uživatel vrácen zpět do menu sloužícího k připojení.

V případě že si hráč vytvoří do hry nebo se připojí do hry již existující, je hráči přidělena herní pozice a spuštěna herní smyčka. Hra je zobrazována tvz. "na výšku". Hráč který se připojí jako první je hráč horní, druhá hráč je umístěn do spodní části herní plochy. Oba hráči se mohou pohybovat pouze doleva a doprava tak, aby nemohli opustit herní pole. Hratelný hráč je zobrazen jako čára zeleně, protivní je zobrazen v barvě červené. Míč je zobrazen jako fialový vyplněný kruh. Hra také vykresluje bíle středovou čáru, okolo které jsou na pravé straně vykresleny body jednotlivých hráčů. V případě, že je hra pozastavena, je ve středu obrazovky vypsán text PAUSED.

2.2.1 Herní smyčka a komunikace

Jakýkoliv požadevek v menu aktivně čeká na odpověd od serveru nebo vypršení času, teprve poté je možné odeslat požadavek nový. Herní klient rozeznává v zásadě tři typy zpráv. Jedním typem je odpověd serveru na požadavek KeepAlive, který resetuje čas poslední komunikace. Druhým typem jsou zprávy řízení klienta, v rámci těchto zpráv se řeší připojení k serveru,

registrace, odpověd na založení hry či připojení ke hře jiné. Poslední zprávy jsou zprávy herní, které mají svůj vlastní buffer.

V herní smyčce se nejprve ověřuje zda je stále platné připojení k serveru. Následně se procházejí přijaté herní zprávy a odpovědi od serveru na základě, kterých jsou spouštěny akce. Následně ověřujeme, zda se uživatel nechce pohnout, v případě že ano, pohyb ověříme a odešleme na server. V poslední řade aktualizujeme data a vykreslíme je (pozice hráčů, míčku, počet bodů, atd.).

Stejně jako na serveru je klient omezen na maximálně 30 průchodů smyčky za sekundu, díky čemuž klientu může dosahovat maximálně 30FPS.

3 Protokol

Protokol vytvořený pro komunikace klientů se serverem je protokolem textovým (bytovým). Prokol vyhrazuje některé znaky (byte) jako znaky kontrolní.

Znak	Popis
<	Začátek zprávy
>	Ukončení zprávy
|	Oddělovač hlavičky a obsahu
:	Oddělovat klíče a hodnoty páru
;	Oddělovač párů
\	Escape znak

Tabulka 1: Seznam řídících znaků

Zpráva se skládá z hlavičky a obsahu. Hlavička zprávy je vždy pevně daná a musí, v tomto pořadí, obsahovat *identifikátor zprávy*, *návratový identifikátor*, *typ zprávy* a dekodér na serveru si ke zprávě sám přidává identifikátor odesílatele. Hlavičku nelze escapovat. Hodnota *"id"* určuje pořadí zprávy. Hodnota *"rid"* je nenulová pouze v případě, kdy je předpoklad, že se na zprávu bude odpovídat. Pokud klient pošle zprávu s RID 10, server odpoví se zprávou ID 10.

Obsah zprávy je tvořen dvojicí klíč a hodnota, které jsou odděleny znakem *":"*. Dvojice je ukončena znakem *";"*. U obsahu nezáleží na pořadí klíčů, důležitá je pouze jejich existence. Obsah lze escapovat znakem *"\"*, pro napsání escapovacího znaku je třeba napsat escapovací znak dvakrát za sebou (tj. *"\\"*).

Zpráva vždy začíná počátečním znakem *«"* následovaným dvojicí ve formátu *"id:%d"*. Tu následující další dvě dvojice *"rid:%d"* a *"type:%d"*. Jednotlivé dvojice jsou oddělené znakem *";"*. Následně je zde umístěn znak konce hlavičky *"|"*. Dvojice v obsahu zprávy jsou pak načítány při splnění formátu *"key:value;"* do nalezení koncového znaku *»"* nebo přetečení maximální velikosti zprávy.

3.1 Seznam zpráv - klient

3.1.1 KeepAlive

Typ zprávy: 1100
Formát:
<id:INT;rid:INT;type:1100;|status:ok;>
Klient periodicky odesílá zprávu na server. Server posílá stejnou zprávu zpět.

Klíč	Datový typ	Hodnota
status	string	ok

Tabulka 2: Obsah KeepAlive zprávy - klient i server

3.1.2 GameJoin

Typ zprávy: 2100
Formát:
<id:INT;rid:INT;type:2100;|gameID:INT;playerID:INT;>
Klient odesílá požadavek o připojení do hry.

Klíč	Datový typ	Hodnota
gameID	int	%d
playerID	int	%d

Tabulka 3: Obsah GameJoin zprávy - klient

Klíč	Datový typ	Hodnota
status	string	ok || error
msg	string	%s
player	string	"1"|| "2"

Tabulka 4: Obsah GameJoin zprávy - server

3.1.3 GameCreate

Typ zprávy: 2000
Formát:
<id:INT;rid:INT;type:2000;|playerID:INT;>
Klient odesílá požadavek o vytvoření hry.

Klíč	Datový typ	Hodnota
playerID	int	%d

Tabulka 5: Obsah GameCreate zprávy - klient

Klíč	Datový typ	Hodnota
status	string	ok || error
msg	string	%s
gameID	int	%d

Tabulka 6: Obsah GameCreate zprávy - server

3.1.4 GameList

Typ zprávy: 2300
Formát:
<id:INT;rid:INT;type:2300;|playerID:INT;>
Klient odesílá požadavek o seznam existujících her.

Klíč	Datový typ	Hodnota
playerID	int	%d

Tabulka 7: Obsah GameList zprávy - klient

Klíč	Datový typ	Hodnota
status	string	ok || error
msg	string	%s
gameCount	int	%d
gameID%d	int	%d (více klíčů)

Tabulka 8: Obsah GameList zprávy - server

3.1.5 Reconnect

Typ zprávy: 2200
Formát:
<id:INT;rid:INT;type:2200;|username:STRING;>
Klient odesílá požadavek o znovupřipojení na existující účet.

Klíč	Datový typ	Hodnota
username	string	%s

Tabulka 9: Obsah Reconnect zprávy - klient

Klíč	Datový typ	Hodnota
status	string	ok || error
msg	string	%s
gameID	int	%d
playerID	int	%d
playas	string	"1"|| "2

Tabulka 10: Obsah Reconnect zprávy - server

3.1.6 Register

Typ zprávy: 1000
Formát:
<id:INT;rid:INT;type:1000;|name:STRING;>
Klient odesílá požadavek o registraci.

Klíč	Datový typ	Hodnota
name	string	%s

Tabulka 11: Obsah Register zprávy - klient

Klíč	Datový typ	Hodnota
status	string	ok || error
msg	string	%s
playerID	int	%d

Tabulka 12: Obsah Register zprávy - server

3.1.7 GameAbandod

Typ zprávy: 2500
Formát:
<id:INT;rid:INT;type:2500;|playerID:INT;>
Klient odesílá informaci o definitivním opuštění hry. Klient ignoruje odpověd.

Klíč	Datový typ	Hodnota
playerID	int	%d

Tabulka 13: Obsah GameAbandon zprávy - klient

3.1.8 PlayerUpdateState

Typ zprávy: 3000
Formát:
<id:INT;rid:INT;type:3000;|playerID:INT;x:INT;y:INT;paused:STR-BOOL;>
Klient odesílá informace pohybu na server. Server odpovídá zprávou 2400. Deprecated hodnoty jsou ignorovány serverem ale stále vyžadovány protokolem.

Klíč	Datový typ	Hodnota
playerID	int	%d
x	int	%d
y	int	%d (deprecated)
paused	string	"true"|| "false"(deprecated)

Tabulka 14: Obsah PlayerUpdateState zprávy - klient

3.2 Seznam zpráv - server

3.2.1 GameUpdateStateClient

Typ zprávy: 2400
Server odesílá informace o aktuálním stavu hry (samovolně ale hlavně jako reakci na zprávu 3000).

Klíč	Datový typ	Hodnota
player1x	int	%d
player1y	int	%d
player2x	int	%d
player2y	int	%d
score1	int	%d
score2	int	%d
ballx	int	%d
bally	int	%d
ballspeed	int	%d
ballrotation	int	%d
paused	string	"true"|| "false"

Tabulka 15: Obsah GameUpdateStateClient zprávy - odesílatel server

3.2.2 GameEnd

Typ zprávy: 2400
Server odesílá informaci o konci hry. Ignoruje protesty klienta.

Klíč	Datový typ	Hodnota
status	string	"ok"
msg	string	"Player %d won!"

Tabulka 16: Obsah GameEnd zprávy - odesílatel server

3.3 Kontext klienta

3.3.1 Kontext klienta na serveru

Klient po připojení k serveru může v zásadě existovat v několika kontextech. Kontextem je myšleno to, jak server vidí připojeného klienta. Na základě tohoto kontextu zpracuje různé typy zpráv, které v opačném případě odmítá.

Po navázání spojení je klient ve stavu UNREGISTERED. V tomto kontextu server od klienta přijímá pouze zprávy 1000 (REGISTER) a 2200 (RECONNECT). Na všechny ostatní zprávy odpovídá odpovědí obsahující indikaci chybového stavu

V případě úspěšného zpracování zprávy ve stavu UNREGISERED se klient pro server stává REGISTERED. Ve stavu registered server zpracovává zprávy 1100 (KeepAlive), 2000 (GameCreate), 2100 (GameJoin), 2300 (GameList),

Po úspěšném založení hry, po připojení k existující hře či znovupřipojení po ztraceném spojení je klient pro server ve stavu PLAYING. V tomto stavu server zpracovává zprávy 2500 (GameAbandon) a 3000 (UpdatePlayerState).

3.3.2 Kontext v herním klientu

Klient implementuje podobný kontext jako na serveru. Klient může být ve stavu NOT-CONNECTED, kdy je uživateli v GUI zobrazeno menu pro zadání adresy, portu serveru a jména. Po úspěšné registraci je uživatel přesunut do stavu CONNECTED, kdy je mu zobrazeno menu po připojení do existující hry či založení nové (případně odpojení od serveru). Posledním stavem je stav PLAYING, kdy se uživateli vykresluje hra synchronizovaná se stavem hry na serveru a uživatel může odesílat změnu pozice své platformy na server.

4 Spuštění serveru a klienta

Sestavení serveru vyžaduje nainstalované prostředí jazyka Golang v minimální verzi 1.13. Server lze sestavit příkazem *go build .*, případně rovnou spustit příkazem *go run .* (za předpokladu, že je terminál přepnut do složky se zdrojovým kódem serveru).

Sestavení klienta vyžaduje nainstalovaný interpret jazyka Python v minimální verzi 3.7. Dále je potřeba nainstalovat moduly pygame (*pip3 install pygame*), PyGameMenu (*pip3 install pygame-menu* a modul tkinter. Klient pak lze spustit z příkazové řádky příkazem *python3 ./main.py* (za předpokladu, že je terminál přepnut do složky se zdrojovým kódem klienta).

5 Závěr

Před vytvářením této semestrální práce jsem měl nulové zkušenosti s vyvíjením her i s vyvíjením síťových aplikací. Také jsem neměl prakticky žádné zkušenosti s jazykem go. Jedinou výhodou byla moje základní znalost jazyka Python.

Vrámci tvorby aplikace jsem několikrát změnil technologii serveru i klienta, což mě stálo spoustu času, který mi na závěrečné odevzdání práce chyběl. Snažil jsem se proto vytvořit konečné řešení co nejrychleji jsem mohl. Musím však říci, že s prací nejsem spokojen, po této zkušenosti bych server i client navrhl jiným způsobem.

Přesto si však myslím, že jsem práci splnil, tedy aspoň v to doufám...

Seznam tabulek