# 

# **KIV/UPS – Semestrální práce**

# **Othello**

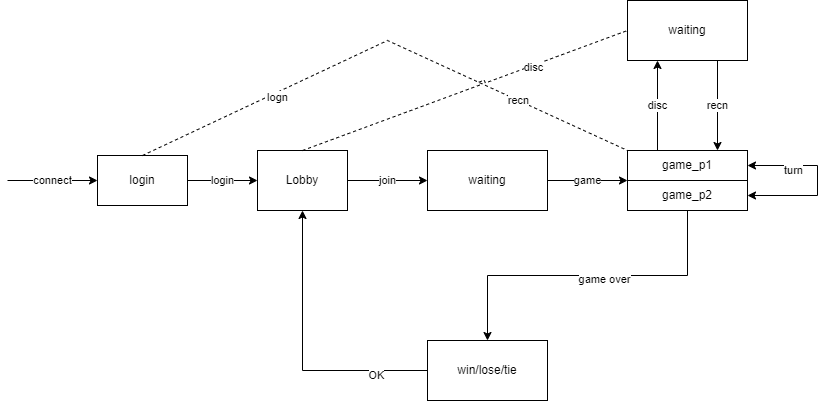
## David Šavel 14. 1. 2023

### Othello

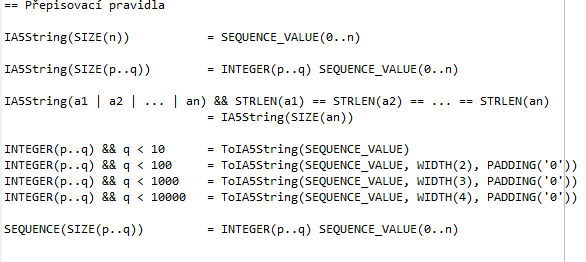
V této hře proti sobě hrají vždy dva hráči. Oba hráči mají kameny své barvy a postupně je pokládají na herní plochu (8 x 8). Cílem hry je mít na herní ploše po celém jejím zaplnění více kamenů, než soupeř. Na začátku hry jsou uprostřed herní plochy již umístěny křížen dva kameny od každé barvy. Hráči se střídají a každý musí při svém tahu umístit kámen a to tak, aby obklíčil alespoň jeden soupeřův kámen. Obklíčené kameny se přebarví na barvu hráče, který je obklíčil. Obklíčit kámen lze v horizontálním, vertikálním i diagonálním směru (i ve více směrech naráz). Ve hře (typicky ke konci) může dojít k situaci kdy hráč nemůže obklíčit žádné soupeřovy kameny. V tom případě musí přeskočit tah, avšak pokud hrát může, tak hrát musí. V deskové podobě hry si hráči musí sami otáčet obklíčené kameny, avšak v této digitální verzi by to bylo poměrně nekomfortní. Jednak by to zněkolikanásobilo herní čas, navíc by uživatel stále poněkud nezáživně klikal.

### Popis protokolu

Všechny zprávy přenášené mezi klientem a serverem mají stejný magic a to “KIVUPS“. Následují čtyři znaky, které určují typ zprávy, a to jsou "logn", "join", "turn", "game", "disc", "recn" a "ping". Konkrétní formát zpráv je popsán v obrázku níže. Zpráva typu “turn“ přenáší číslo, které značí index herní plochy kam daný hráč zahrál. Význam všech zpráv je patrný z názvu, za zmínku stojí, že zpráva “disc“ se posílá dvakrát. Nejprve po detekci odpojení hráče a podruhé po nějakém časovém intervalu, pokud se hráč znovu nepřipojí, aby soupeř nečekal na jeho opětovné připojení nekonečně dlouho. Přezdívka hráče smí být maximálně devět znaků dlouhá.



Obrázek 1 stavový diagram -> posloupnost zpráv



Obrázek 2 -> formát zpráv

Klient i server hlásí, když jeho protějšek přestane komunikovat -> neodpovídá na zprávy. Pokud nezačne komunikovat do určitého časového intervalu, aplikace to zahlásí a ukončí s ním komunikaci (v případě klienta se ukončí úplně). Další chybová hláška se vypíše, pokud při navazování spojení dojde k nějaké chybě a spojení se navázat nepodaří. Chybová hláška je vypsána i v případě, že se klient nebo server ukončí, znamená to okamžité ukončení spojení a v případě klienta opět i ukončení programu. Poslední chybová hláška je vypsána po přijetí nevalidní zprávy. V tom případě je postu stejný jako při ukončení protějšku. Za nevalidní zprávy jsou považovány všechny zprávy, které neobsahují magic (výjimkou jsou zprávy potvrzení přijetí OK).

### Popis Implementace

**Server**

Server je psaný v jazyce C a má pouze jednu třídu. Hlavní metoda třídy čeká v nekonečném cyklu na připojení klientů. Pro každého klienta jsou spuštěny po připojení dvě vlákna. Jedno přijímá a obsluhuje zprávy od klienta a druhé neustále kontroluje, jak dlouhou dobu nepřišla zpráva od klienta a po překročení limitu posílá ping zprávu.

Všechna data, která si server uchovává jsou uloženy v globálních proměnných, které jsou při spuštění alokovány. Aplikační vrstva je zpracována ve vlákně *thread\_fnc*, která přijímá zprávy ukládá si data uložená ve zprávách. Podle zpráv si i ukládá informace o připojených klientech jako zda jsou ve hře, čekají na hru, nebo zda nejsou dočasně nedostupní. Do konzole server vypisuje pouze jaké přichází a odchází zprávy a k některým zprávám vypisuje krátké hlášení o tom co se na základě zprávy stalo.

**Klient**

Klient je psaný v jazyce Java a má celkem 7 tříd. Hlavní třída pouze podle počtu parametrů vytvoří instanci třídy *SocketManager*, která řídí celou aplikaci. Tato třída tedy naváže spojení se serverem a poté spustí jedno vlákno, které přijímá zprávy od serveru a podle nich řídí celou aplikaci -> vytváří/zavírá okna, posílá zprávy jiným třídám atd. Třída *SocketManager* spouští ještě vlákno, které posílá ping zprávu, pokud nepřišla zpráva od serveru v nějakém časovém intervalu. Tato třída také odesílá veškeré zprávy, které se na server posílají. Zbylých pět tříd klienta jsou tříd, které dědí od třídy *JFrame* a každá z nich zobrazuje jedno z oken se kterými aplikace pracuje. Jsou to tedy třídy pro zadaní loginu, další pro připojení do hry, pak pro samotnou hru, další pro zobrazení výsledků po konci hry a poslední je okno, které se zobrazí, pokud server přestane komunikovat a čeká se na obnovu komunikace.