

Dokumentace semestrální práce

Předmět: KIV/UPS • **Autor:** Danyil Hrechishkin • **Datum vytvoření:** 29.01.2026

Blackjack — TCP klient/server (JavaFX + C)

Projekt implementuje síťovou dvouhráčovou variantu hry Blackjack. Server (C, POSIX) poskytuje lobby a běh hry, klient (Java 17 + JavaFX) poskytuje UI, komunikaci a automatický reconnect při výpadku sítě.

1 Základní popis hry

Blackjack je karetní hra, ve které se hráč snaží dosáhnout hodnoty ruky co nejbližší k 21, bez jejího překročení. Tato implementace je **dvouhráčová** (bez samostatného dealera). Oba hráči dostanou dvě karty a následně se střídají v tazích.

1.1 Použitá varianta pravidel

- Používá se standardní balíček 52 karet.
- Hodnoty: 2–10 dle čísla, J/Q/K = 10, eso (A) = 11 nebo 1 (dle toho, aby součet nepřesáhl 21).
- Každý hráč začíná se dvěma kartami.
- Ve svém tahu hráč volí:
 - **HIT** — dobere jednu kartu.
 - **STAND** — ukončí dobírání.
- Pokud hráč po HIT překročí 21, nastává **bust** a pro hráče je jeho skóre nastaveno na -1.
- Hra končí, když oba hráči stojí nebo bustnou.
- Výsledkem je vítěz (vyšší skóre; -1 je nejhorší) nebo **PUSH** (remíza).

1.2 Časování a odolnost

Hra je navržena tak, aby byla použitelná i na nestabilní síti:

- Každý tah má časový limit (na serveru `TURN_TIMEOUT_SEC`).
- Klient i server používají keep-alive zprávy (PING/PONG), aby se výpadek odhalil v rozumném čase.
- Při odpojení jednoho hráče během hry server čeká omezenou dobu na jeho návrat (`RECONNECT_TIMEOUT_SEC`) druhý hráč je o tom informován.
- Klient při ztrátě spojení provádí automatický reconnect v definovaném okně.

2 Přehled řešení a rozvrstvení aplikace

Projekt je rozdělen na dvě hlavní části:

- **Server:** správa připojení, handshake, lobby, běh herních vláken, reconnect.
- **Klient:** UI scény, síťová vrstva, parsování protokolu, automatické obnovení spojení.

2.1 Architektonický přehled

Klient je rozdělen na:

- **UI vrstvu (JavaFX)** — scény, ovládání hry, zobrazování stavu spojení,
- **síťovou vrstvu** — TCP komunikace, parsování protokolu, keep-alive a reconnect.

Server je rozdělen na:

- **síťovou vrstvu** — accept-loop a per-client vlákna,
- **logiku hry** — lobby, herní vlákno a běh zápasu,
- **I/O pomocné funkce** — bezpečné čtení/psaní řádků a serializace snapshotů.

2.2 Adresářová struktura

- **server/** — serverová část v jazyce C
 - `src/main.c` — CLI entry point, načtení konfigurace, start serveru
 - `src/server.c` — síťový stavový automat klienta, registry jmen, accept-loop
 - `src/game.c` — lobby + herní logika, herní vlákna, reconnect ve hře
 - `src/protocol.c` — nízkourovňové čtení/zápis řádků a serializace snapshotů lobby
 - `include/*.h` — hlavičky (API a datové struktury)
 - `config.txt` — runtime konfigurace (IP, port, počet lobby)
- **clientBlackJack/client/** — klient v Javě
 - `com.blackjack.MainApp` — JavaFX aplikace, scény, navigace a binding na síť
 - `com.blackjack.net.NetClient` — TCP klient, reader thread, parser protokolu, keep-alive a reconnect
 - `com.blackjack.net.ProtocolListener` — callback rozhraní pro UI
 - `com.blackjack.ui.GameView` — herní okno (log, tlačítka, reconnect label)
 - `pom.xml` — Maven konfigurace (Java 17, JavaFX)

2.3 Použité knihovny a prostředí

- **Server:**
 - Jazyk: C11
 - Překladač: `gcc` (nebo kompatibilní)
 - POSIX API: `socket`, `poll`, `pthread`, `recv/send`, `shutdown`, `getifaddrs`
- **Klient:**
 - Java: **17** (Maven `maven.compiler.release=17`)
 - JavaFX: **17.0.15** (artefakty `javafx-controls`, `javafx-fxml`)
 - Build: Maven + `javafx-maven-plugin`

3 Protokol (specifikace pro alternativní implementaci)

3.1 Základní pravidla formátu zpráv

- Komunikace probíhá přes **TCP**.
- Všechny zprávy jsou **textové řádky** ukončené znakem `\n`.
- Oddělovačem parametrů je **whitespace** (mezera / tabulátor).
- Tokeny vždy začínají prefixem **C45**.
- TCP je stream: zpráva může přijít po částech nebo „slepeně“; hranice zpráv určuje pouze `\n`.

3.2 Přenášené datové typy a validace

- `<name>` — jméno hráče, neprázdné, bez whitespace, délka ≤ 63 znaků (server má `MAX_NAME_LEN=64`).
- `<lobby>` — číslo lobby (1..N); pro reconnect je povoleno 0 jako „neznámé“ (pokud není hráč ve hře, klient se vrátí do lobby listu).
- `<n>` — počet lobby ve snapshotu (kladné celé číslo).
- `<pairs>` — řetězec délky $2*n$ tvořený číslicemi; dvojice **players** (0..2) a **status** (0/1).
- `<card>` — karta jako dvouznakový kód [A23456789TJQK] [CDHS] (např. AS, 7H).
- `<sec>` — sekundy (timeout tahu, timeout reconnectu).
- `<score>` — celé číslo; -1 znamená bust, jinak typicky 0..21.

3.3 Omezení délky zpráv a důsledky pro návrh

Server čte řádky do bufferu `READ_BUF=256`; klient proto nesmí posílat extrémně dlouhé řádky (jinak hrozí `C45WRONG` / disconnect).

3.4 Přehled zpráv

Všechny zprávy jsou textové řádky zakončené `\n`.

3.4.1 Zprávy klient → server

- `C45<name>` — handshake (registrace jména).
- `C45REC <name> <lobby>` — reconnect/resume session.
- `C45J <lobby>` — join lobby.
- `C45B` — „Back“ (návrat do lobby listu / refresh snapshotu; ve hře forfeit).
- `C45H` — HIT (jen ve svém tahu).
- `C45S` — STAND (jen ve svém tahu).
- `C45P0` — odpověď na PING (`C45PI`).

3.4.2 Zprávy server → klient

- C450K — obecné potvrzení.
- C45L <n> <pairs> — snapshot lobby listu (kompaktní 1 řádek).
- C45REC_OK — reconnect akceptován.
- C45WRONG [detail] — chyba protokolu / nevalidní požadavek.
- C45DOWN [reason] — server se vypíná; klient má ukončit spojení.
- C45D <c1> <c2> — počáteční dvě karty.
- C45C <card> — hráč si dobral kartu.
- C45T <name> <sec> — kdo je na tahu a timeout v sekundách.
- C45B <name> <value> — bust (hráč překročil 21).
- C45T0 — timeout lokálního hráče (auto-stand).
- C450D <name> <sec> — oponent se odpojil, server čeká sec.
- C450B <name> — oponent se vrátil, hra pokračuje.
- C45R <p1> <s1> <p2> <s2> <winner> — výsledek hry (winner může být PUSH).

3.4.3 Keep-alive

- C45PI — PING (obousměrně).
- C45PO — PONG (obousměrně).

3.5 Poznámky

Token C45B má dva významy: klient → server C45B = „Back“, server → klient C45B <name> <value> = bust (rozliší se směrem a parametry).

3.6 Sekvenční scénáře

- **Connect + handshake:** TCP connect, klient pošle C45<name>, server odpoví C450K a pošle C45L.
- **Lobby:** klient pošle C45J <lobby>; po úspěchu přijde C450K. Klient může poslat C45B pro refresh C45L.
- **Game:** server pošle C45D, pak opakovaně C45T. Klient v tahu posílá C45H nebo C45S. Server posílá C45C, případně C45B <name> <value> a nakonec C45R.
- **Disconnect + reconnect:** při výpadku server pošle druhému hráči C450D. Odpojený hráč se vrací přes C45REC <name> <lobby> a server odpoví C45REC_OK a obnoví hru (snapshot ruky).
- **Back po hře:** po C45R klient odešle C45B a server pošle nový C45L.

3.7 Relační diagram komunikace

Ukazuje hlavní zprávy mezi klientem a serverem a typické stavy obou stran.

Legenda:

C->S klient posílá na server
S->C server posílá na klienta
[C:...] a [S:...] jsou zjednodušené stavy

1) CONNECT + HANDSHAKE + LOBBY LIST

[C:DISCONNECTED] TCP connect
[S:LISTEN] accept -> [S:HANDSHAKE]
C->S: C45<name>
S->C: C45OK
S->C: C45L <n> <pairs>
[C:LOBBY_CHOICE] [S:LOBBY_SELECT]

2) JOIN LOBBY

C->S: C45J <lobby>
S->C: C45OK
[C:WAIT_START] [S:WAIT_FOR_GAME]

3) GAME (zjednodušené)

S->C: C45D <c1> <c2>
opakovane:
S->C: C45T <name> <sec>
pokud jsem na tahu:
C->S: C45H | C45S
S->C: C45C <card> (po HIT)
S->C: C45T0 (když vyprší čas)
S->C: C45B <name> <val> (bust)
konec hry:
S->C: C45R <p1> <s1> <p2> <s2> <winner>

4) DISCONNECT + RECONNECT (během hry)

pokud klient vypadne:
S->oponent: C45OD <name> <sec>
[C:RECONNECTING] nové TCP spojení
C->S: C45REC <name> <lobby>
S->C: C45REC_OK
S->C: snapshot ruky (C45D + 0..N * C45C) nebo C45L ...

5) KEEP-ALIVE (v libovolném stavu)

C<->S: C45PI / C45PO

4 Chybové stavy a jejich hlášení

4.1 Protokolové chyby (C45WRONG...)

Server posílá C45WRONG typicky při:

- Neplatném formátu zprávy (neznámý token, chybějící parametry).
- Neplatném lobby čísle nebo akci v nesprávné fázi.
- Neplatném reconnect requestu.

Chování po chybě závisí na fázi:

- **Handshake / reconnect:** server ukončí spojení (nepřijatelný klient).
- **Lobby select:** server může poslat C45WRONG a zůstat připojený (uživatel může zadat jiné lobby).
- **Po skončení hry:** neznámé zprávy jsou považovány za chybu a spojení je ukončeno.

4.2 Kolize jmen (C45WRONG NAME_TAKEN)

Jméno hráče je unikátní napříč aktivními spojeními. Pokud hráč zvolí jméno, které je již rezervované, server odpoví:

```
C45WRONG NAME_TAKEN
```

Klient tuto situaci zobrazí uživateli a ukončí session (nový connect je nutný s jiným jménem).

4.3 Server shutdown (C45DOWN)

Pokud se server vypíná (např. SIGINT nebo ztráta bind IP), pošle všem připojeným klientům:

```
C45DOWN [reason]
```

Klient má ukončit spojení a přejít na úvodní obrazovku.

5 Implementace serveru (programátorská dokumentace)

5.1 Moduly a odpovědnosti

- `src/main.c` — načte `config.txt`, zpracuje CLI volby, inicializuje lobby a spustí server.
- `src/server.c` — accept-loop, jedno vlákno na klienta, handshake, lobby select, koordinace s herním vláknem, registry aktivních jmen.
- `src/game.c` — logika hry, správa lobby poolu, herní vlákno pro každé lobby, reconnect během hry.
- `src/protocol.c` — robustní `read_line/write_all` a serializace snapshotu lobby (C45L).

5.2 Klíčové funkce a jejich role

- `run_server(ip,port)` — listen socket, `poll()` na accept, kontrola dostupnosti bind IP, vytvoření per-client vláken; při konci pošle C45DOWN.
- `client_thread(...)` — stavový automat klienta (handshake/reconnect, lobby, čekání, návrat po hře); toleruje C45PI/C45PO.
- `send_lobbies_snapshot(fd)` — vytvoří a odešle snapshot C45L.
- `start_game_if_ready(i)` — spustí herní vlákno lobby po naplnění 2 hráči.
- `lobby_game_thread(...)` — běh hry (tahy, timeouty, reconnect, výsledky).
- `read_line()/write_all()` — bezpečné řádkové I/O nad TCP.

5.3 Síť a odolnost

Server používá `poll()` (bez aktivního čekání) a keep-alive `C45PI/C45PO`. Při shutdownu (nebo ztrátě `bind IP`) informuje klienty přes `C45DOWN` a ukončí spojení.

5.4 Paralelizace

Server používá **vlákna** (`pthread`):

- **Accept-loop** běží v hlavním vlákne a pro každého klienta spouští `client_thread()`.
- **Per-client vlákno** obsluhuje stavový automat jednoho hráče (`handshake`, `lobby`, `čekání`, `návrat`).
- **Herní vlákno lobby** je spuštěno, když lobby dosáhne kapacity 2 hráčů (`start_game_if_ready()`).

Synchronizace probíhá pomocí mutexů:

- `Lobby.mtx` — chrání stav lobby (`hráči`, `hand`, `fd`, `is_running`).
- `g_names_mtx` — chrání globální registry aktivních jmen a jejich tokenů.
- `g_clients_mtx` — seznam připojených socketů (pro rychlé odpojení všech klientů).

5.5 Obsluha reconnectu na serveru

- Klient se vrací přes `C45REC <name> <lobby>` na novém TCP spojení.
- Server odpoví `C45REC_OK` a pokračuje buď obnovou hry (`snapshot ruky`), nebo návratem do lobby listu (`C45L`).

5.6 Konfigurace serveru

Server čte `server/config.txt` (key-value, whitespace oddělené). Příklad:

```
LOBBY_COUNT 3
IP 0.0.0.0
PORT 10000
```

6 Implementace klienta

6.1 Rozvrstvení a třídy

- **MainApp:**
 - JavaFX entry point, vytváří scény (`connect`, `name`, `lobby`, `waiting`, `game`).
 - Vytváří instanci `NetClient` a registruje `ProtocolListener`.
 - Přepíná UI podle příchozích zpráv.
- **NetClient:**
 - Udržuje socket, `BufferedReader` a `OutputStream`.
 - Spouští daemon reader thread, který provádí blokující `readLine()`.
 - Implementuje parser protokolu a callbacky do `ProtocolListener`.
 - Implementuje keep-alive a automatický reconnect.

- **ProtocolListener:**

- Rozhraní callbacků pro UI (OK, snapshot lobby, herní eventy, chyby, reconnect progress).

- **GameView:**

- Herní scéna: log, skóre, tlačítka HIT/STAND, tlačítko „Back to Lobby“.
- Zobrazuje reconnect status label přímo v herním okně.
- Při obnově hry (snapshot ruky) nezdvouje log karet.

6.2 UI

Klient má obrazovky pro připojení (IP/port), zadání jména, výběr lobby a hry. Ruční **Reconnect** je dostupný až po předchozím úspěšném připojení (klient zná poslední **name** a **lobby**).

6.3 Paralelizace klienta

Klient používá **dvě hlavní vlákna**:

- JavaFX Application Thread — rendering UI a zpracování akcí uživatele.
- Reader thread (**net-reader**) — čte TCP stream, parsuje protokol a vyvolává callbacky.

Veškeré změny UI jsou prováděny pomocí `Platform.runLater(...)`.

6.4 Keep-alive a reconnect

Klient pravidelně ověřuje, že server odpovídá (PING/PONG). Při výpadku se pokusí automaticky obnovit spojení; při úspěchu pošle `C45REC <name> <lobby>` a pokračuje ve hře nebo se vrátí do lobby listu. Pokud auto-reconnect selže, uživatel použije ruční **Reconnect**.

7 Požadavky na překlad, spuštění a běh aplikace

7.1 Server (C)

Požadavky:

- Linux/Unix-like prostředí s POSIX API
- gcc nebo kompatibilní C11 překladač
- make

Postup překladu:

```
cd server
make -j
```

Spuštění:

- výchozí: server čte `config.txt` v aktuální složce,
- volitelně lze zadat IP a port přes CLI: `-i <IP> -p <PORT>`.

```
./blackjack_server -i 0.0.0.0 -p 10000
```


7.1.1 Konfigurace serveru (config.txt)

Konfigurační soubor je textový, každá položka je ve formátu **KEY VALUE** (oddělené whitespace). Podporované klíče:

- **LOBBY_COUNT** — počet lobby (1–99),
- **IP** — bind adresa (např. 0.0.0.0 nebo 127.0.0.1),
- **PORT** — port (1..65535).

7.1.2 Logování

Server vypisuje stavové informace na standardní výstup (**stdout**): připojení klientů, protokolové chyby, start/stop hry a informace o odpojení/reconnectu.

7.2 Klient (JavaFX)

Požadavky:

- Java JDK 17
- Maven

Postup překladač/spuštění:

```
cd clientBlackJack/client
mvn clean javafx:run
```

7.2.1 Poznámky k běhu JavaFX

Spuštění přes `mvn javafx:run` automaticky nastaví potřebné JavaFX moduly a knihovny. V případě ručního spouštění mimo Maven je potřeba správně nastavit `-module-path` a `-add-modules` pro JavaFX (nebo vytvořit bundlovanou distribuci).

Běh aplikace:

- Uživatel zadá IP a port serveru.
- Zadá jméno (bez mezer).
- Zobrazí se lobby list, uživatel zvolí lobby číslo.
- Po startu hry se zobrazí herní okno.

7.2.2 Ruční reconnect (uživatelský postup)

Pokud automatický reconnect selže a aplikace zobrazí hlášku „Please reconnect manually“, uživatel může:

- přejít na úvodní obrazovku (Connect),
- zadat IP/port,
- použít tlačítko **Reconnect** (je dostupné pouze pokud klient zná poslední **name** a **lobby**).

Tlačítko odešle `C45REC <name> <lobby>` a klient pokračuje v čekání na obnovení hry nebo na lobby snapshot.

8 Ukázky komunikace

8.1 Handshake a lobby snapshot

```
# Client -> Server
C45Alice

# Server -> Client
C450K
C45L 3 000000
```

8.2 Join lobby a start hry

```
# Client -> Server
C45J 1

# Server -> Client
C450K

# pozdeji, kdyz jsou v lobby 2 hraci:
C45D AS 5S
C45T Alice 60
```

8.3 Reconnect během hry

```
# Client -> Server (nove TCP spojeni)
C45REC Alice 1

# Server -> Client
C45REC_OK

# snapshot ruky (minimalne prvni 2 karty)
C45D AS 5S
C45C 3H

# pokračovani hry
C45T Bob 60
```

8.4 Poznámky

- Typické chyby serveru: C45WRONG (nevalidní požadavek), C45WRONG NAME_TAKEN (kolize jmen), C45DOWN (server se vypíná).
- Důležité timeouty: tah hráče je standardně 60 s (TURN_TIMEOUT_SEC), server čeká na reconnect ve hře 60 s (RECONNECT_TIMEOUT_SEC).

9 Závěr a zhodnocení dosažených výsledků

Byla úspěšně implementována síťová hra Blackjack ve variantě 1 vs. 1, která je rozdělena na serverovou část v jazyce C a klientskou část v Javě (JavaFX). Cílem bylo vytvořit funkční klient/server řešení s vlastním protokolem a současně zajistit použitelnost i při běžných síťových problémech (zpoždění, fragmentace, výpadek spojení).

9.1 Zhodnocení protokolu

Zvolený protokol je záměrně jednoduchý: textové řádky přes TCP, tokeny s prefixem `C45` a parametry oddělené whitespace. Tento přístup má výhody v laditelnosti (lze testovat přes `nc`) a v možnosti snadno vytvořit alternativního klienta/server. Součástí protokolu jsou:

- handshake a lobby snapshot (`C45OK`, `C45L`),
- průběh hry (`deal/turn/card/bust/timeout/result`),
- keep-alive (`C45PI/C45PO`) pro rychlejší detekci „tichých“ výpadků,
- reconnect (`C45REC`, `C45REC_OK`) pro navázání na běžící session.

9.2 Zhodnocení implementace

Server používá POSIX API (`sockets`, `poll()`, `pthread`) a běží jako vícevláknová aplikace: jedno vlákno na klienta a herní vlákno pro lobby, jakmile jsou připojeni dva hráči. Klient používá JavaFX pro UI a samostatné „reader“ vlákno pro blokující čtení ze socketu; změny UI jsou prováděny přes `Platform.runLater`.

9.3 Omezení a možnosti zlepšení

V aktuální verzi jsou nejdůležitější omezení a možné další kroky:

- **Škálování lobby listu:** lobby snapshot `C45L` je jednorázová zpráva; pro tisíce lobby je vhodnější stránkování (více zpráv) nebo jiný formát.
- **Bezpečnost:** protokol je bez autentizace a šifrování (TLS); je vhodný pro LAN/testování, ne pro veřejné nasazení.
- **Herní rozšíření:** přidání dealera, více hráčů, botů nebo statistik by vyžadovalo rozšíření protokolu i UI.
- **Testování:** přidání automatizovaných integračních testů by zlepšilo jistotu při úpravách (např. testy reconnectu a chybových stavů).