Eötvös Loránd Tudományegyetem
Informatikai Kar
Információs Rendszerek Tanszék

Többasztalos és -felhasználós póker játék adatbázis modellezése

Ács Zoltán
tanársegéd

Fehér Valentin
Programtervező Informatikus BSc

Budapest, 2016

Tartalomjegyzék

1. Felhasználói dokumentáció

1.1. Telepítés

Nincs szükség telepítésre.

1.2. Futtatás

A program java programozási nyelvben lett megírva, így a kifordított állomány egy jar file, melyet parancssorból az alábbi utasítással tudunk futtatni

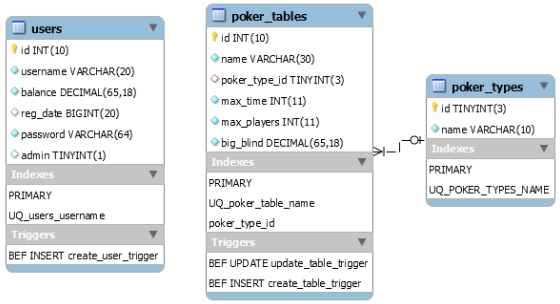
```
java -jar <filenev>
```

1.3. Felhasznált technológiák

A szakdolgozatomat eclipse fejlesztőkörnyezetben írtam, amelyet végül mavennel - build rendszer - fordítottam ki és csomagoltam be. A szakdolgozat felhasznál egy külső könyvtárat, amely a nyertes kiértékelési feladatát látja el. A programcsomagot meg kellett támogatni egy adatbázissal is - MySQL - , amely az adatok perzisztens tárolásáért felel. A programcsomag szerver-kliens architektúrában került implementálásra, amely tovább bomlik kliens oldalon MVC (Model-View-Controller) tervezési stílusra. A modulok közötti kommunikáció RMI Java API felhasználásával történik.

1.4. Adatbázis séma

1. ábra. Adatbázis séma



Az

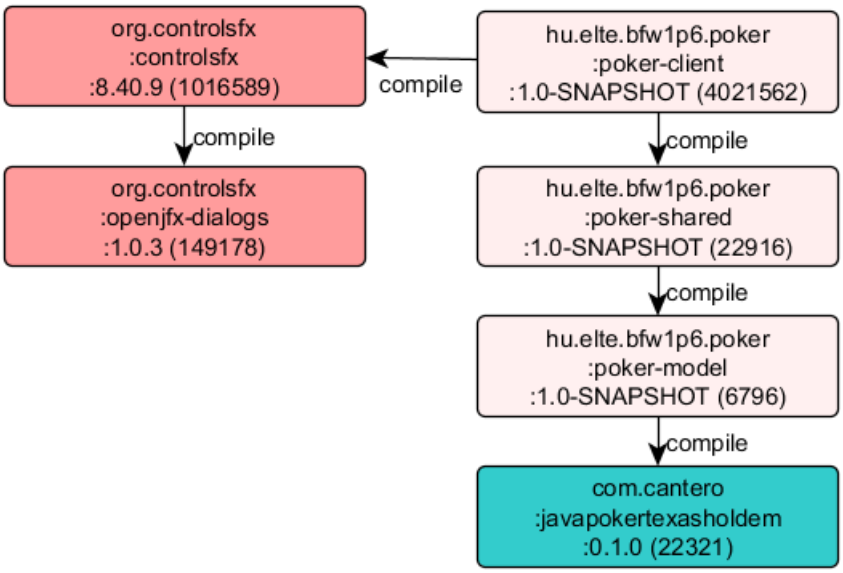
1.5. Modulok

A programcsomag 6 fő modult tartalmaz

- poker-server A póker játék szervere, amely magát a játékot szolgáltatja.

- poker-client A póker játék kliense, amely segítségével a szerverhez lehet csatlakozni.
- poker-shared A póker játék azon modulja, amelytől a szerver és a kliens egyaránt függ.
- poker-persist Az adatok letárolásáért felelős modul.
- poker-model A póker játék modellezéséért felelős csomag.
- javapokertexasholdem Külső könyvtár, amely a nyertes játékos kiértékelési feladatot végzi.

2. ábra. Kliens modulra bontása

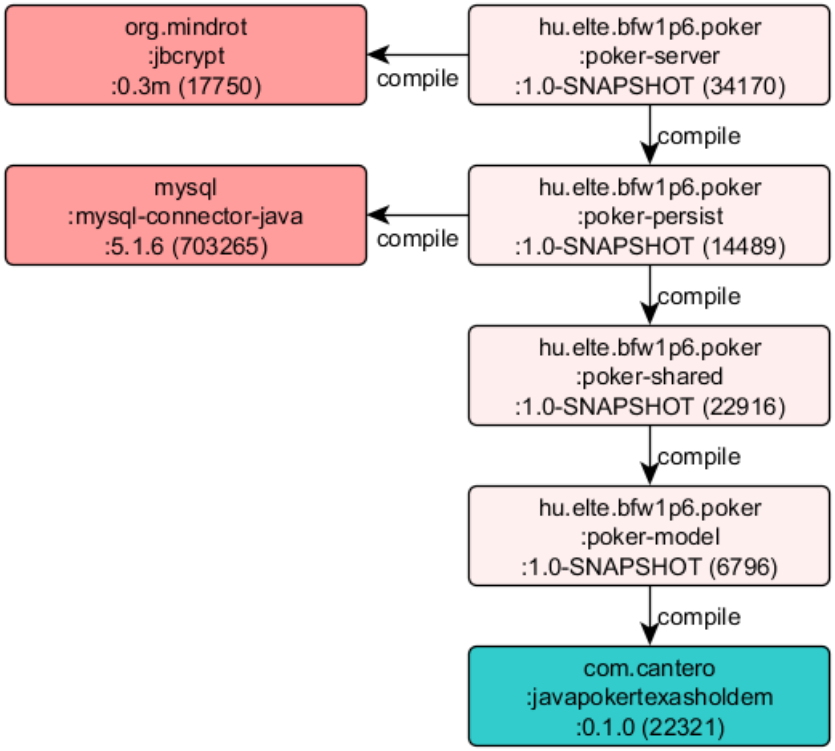


A modulok közötti függőséget a ... ábra mutatja. A programcsomag két fő modulra bontható: poker-server és poker-client. A szerver a jelszavak titkosítására bcrypt eljárást alkalmaz, amelynek a biztonságát sózással növeli. Továbbá a szerver felhasznál még egy külső csomagot - mysql-connector-java -, amely az adatbázis kapcsolatért felel. A poker-shared modul felel a szerver és a kliens jól definiált kommunikációjáért. A shared modul többek között tartalmazza a közös interfészeket, kivételeket és a póker utasítások megvalósítását. A kliens közvetlenül függ ettől a modultól, azonban a szerver és a shared modul közé beékelődött a poker-persist modul.

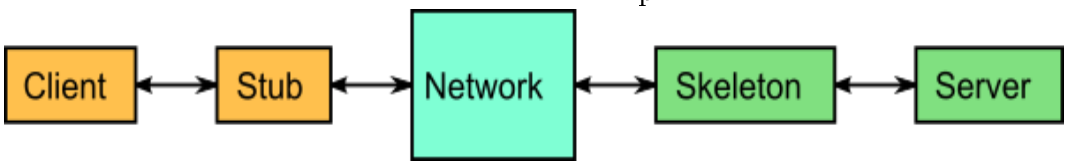
2. Funkciók

Ahogy a témabejelentőben is szerepel..... RMI kép wikiről, majd azt megmagyarázni a shared modullal, kliens hívja, jól definiált interfész etc.... A szerver vázáért a PokerRemote interface felel, amely a játékon végrehajtható műveletek összefogásáért

3. ábra. Szerver modulra bontása



4. ábra. RMI koncepció



felel. Itt található az összes funkció, amely megvalósításra került, mint például játék asztal létrehozása, új felhasználó létrehozás, admin jog kiosztása stb. A kliens ezt a vázat tudja elkérni a registryből, mint kliens-oldali szervercsonk, amelyeken a műveletek meg tudja hívni. Az összes megvalósított funkciót le kell írni? Felsorolás szintjén, vagy hogy? Rövid magyarázattal? És amelyik egyértelmű? Pl. felhasználó módosítása... login...

3. Tovább fejlesztési lehetőségek

- Az adatbázis viszonylag alacsony absztrakciós szinten került implementására, azonban mivel néhány tábláról beszélhetünk csak, ezért igyekeztem elkerülni a keretrendszerek általi overheadet. Ugyanakkor ezen a ponton sokat fejlődhet a programcsomag, ha a későbbiek során esetlegesen bonyolultabban kellene modellezni a játékot adatbázis szempontjából. Például dialektusok - akár Liquibase (hivatkozás) - használata elfedheti a tényleges adatbázis-kezelő rendszer általánosságait, így eggyel magasabb szintre helyezhető a megvalósítás.

- A felhasználói élményen sokat javíthat az animációk használata. A megjelenítés sokkal lágyabb, folyékonyabb lehetne Transition/Animation (bibliográfiába hivatkozás...) objektumok használatával.
- Akár a komplett RMI architektúrát (JDK 1.1-ben jelent meg 18 éves technológia [a http meg 16...]) le lehetne váltani, és helyette REST szoftverarchitektúrát tenni, amely modernebb megjelenést (AngularJS, reszponzív design) és modernebb eszközöket vonna maga után.

A **??** képen látható az adatbázis séma.

4. Tesztelés

4.1. Funkcionális tesztelés

Funkció	Elvárt eredmény
Regisztráció	A program jelezte a felhasználónak, hogy a regisztráció sikeresen megtörtént, és vi
Bejelentkezés	A formot helyesen kitöltve a program sikeresen autentikálta és l
Tábla módosítás	-
Tábla törlés	-