Tervezési minták elemzése: Amőba szoftverprojekt

Model-View-Controller (MVC) minta

A projekt gerincét az **MVC architektúra** alkotja, amely a JavaFX keretrendszer természetes felépítését követi.

- **Model (Amoba.java, User.java, Bot.java):** Ez a réteg felelős az üzleti logikáért és az adatok tárolásáért. A modell nem tud a felhasználói felület létezéséről, így önmagában is tesztelhető (Unit tesztekkel).
- **View (Main.fxml, Amoba.fxml, stb.):** Az XML alapú FXML fájlok definiálják a deklaratív felületet. Ez lehetővé teszi a design és a kód szétválasztását.
- **Controller (AmobaController.java, MainController.java):** A híd a Modell és a Nézet között. Kezeli a felhasználói interakciókat, frissíti a játék állapotát, és vezérli a UI elemek (feliratok, gombok) módosítását.

Facade (Homlokzat) minta

A **DatabaseManager.java** osztály a Facade tervezési mintát valósítja meg az adatbázis-műveletek során.

- **Probléma:** Az SQL lekérdezések, a JDBC kapcsolatok kezelése és a hibakezelés összetett folyamat.
- **Megoldás:** A DatabaseManager egy egyszerűsített interfészt ("homlokzatot") nyújt a többi osztály számára. A vezérlőnek nem kell ismernie az SQL szintaktikát vagy az SQLite belső működését; csupán olyan magas szintű metódusokat hív meg, mint a saveMatchup() vagy a getBotLeaderboard().

Strategy (Stratégia) minta

A **Bot.java** felépítése a Stratégia minta alapelveit követi a döntéshozatali logika során.

- **Alkalmazás:** A bot lépéskalkulációs logikája (calculateMove) egy különálló osztályba van ágyazva.
- **Előny:** Ez lehetővé teszi, hogy a játékos elleni algoritmus (stratégia) cserélhető legyen. A kód felépítése támogatja, hogy később különböző nehézségi szinteket (pl. RandomStrategy, MinimaxStrategy) vezessünk be anélkül, hogy a játék fő vezérlőjét módosítani kellene.

Observer (Megfigyelő) minta

A JavaFX eseménykezelő rendszere az **Observer mintára** épül, amely a UI interakciók alapja.

- **Működés:** A gombok (Button) mint megfigyelt alanyok eseményeket generálnak (pl. kattintás).
- **Alkalmazás:** A @FXML annotációval ellátott metódusok (pl. handleButtonClick) "feliratkoznak" ezekre az eseményekre. Amikor az esemény bekövetkezik, a gomb értesíti a megfigyelőt (a Controllert), amely végrehajtja a szükséges műveletet.

További minták és kódminőségi megoldások

Static Utility & Singleton jelleg

A DatabaseManager statikus metódusokat használ, ami biztosítja, hogy az adatbázis-kapcsolat egyetlen központi ponton keresztül legyen elérhető az alkalmazás teljes élettartama alatt, elkerülve a felesleges erőforrás-pazarlást.

Data Transfer Object (DTO)

A **ScoreEntry** belső osztály egy egyszerű DTO-ként funkcionál. Célja az adatok (név és pontszám) egybecsomagolt szállítása az adatbázis-réteg és a ranglista nézet (TableView) között, megőrizve az adatok integritását.

Összegzés

A fenti tervezési minták alkalmazása jelentősen növelte a szoftver minőségét. Az MVC biztosítja a tiszta struktúrát, a Facade elrejti az alacsony szintű technikai részleteket, a Strategy pedig rugalmasságot ad a jövőbeli fejlesztésekhez. Ez a felépítés tette lehetővé a kód 80% feletti tesztlefedettségét is.