A képen szöveg látható

Automatikusan generált leírás

JavaFX beadandó

Forint

Katona Zsolt

OJZGLU

Sándor Adrián

GMGCIY

2024

Tartalomjegyzék

[Bevezetés 4](#_Toc183727865)

[1. A feladatról 5](#_Toc183727866)

[2. Adatbázis 6](#_Toc183727867)

[2.1. Adatbázis felépítése 6](#_Toc183727868)

[2.1.1. anyag tábla 6](#_Toc183727869)

[2.1.2. tervezo tábla 7](#_Toc183727870)

[2.1.3. erme tábla 7](#_Toc183727871)

[2.1.4. akod tábla 7](#_Toc183727872)

[2.1.5. tkod tábla 8](#_Toc183727873)

[2.1.6. deviza tábla 8](#_Toc183727874)

[2.1.7. params tábla 8](#_Toc183727875)

[3. CRUD menüpont 9](#_Toc183727876)

[4. MNB adatszolgáltatás menüpont 11](#_Toc183727877)

[5. Párhuzamos programvégrehajtás menüpont 14](#_Toc183727878)

[6. Forex menüpont 15](#_Toc183727879)

[Összefoglalás 20](#_Toc183727880)

A szöveg gépelése és formázása után a tartalomjegyzéket frissíteni kell. A frissítéshez kattintson az egér jobb gombjával a tartalomjegyzék tetszőleges elemére, majd válassza a *Mezőfrissítés* menüpontot! A megjelenő párbeszédablakon válassza a *Teljes jegyzék frissítése* lehetőséget!

Bevezetés

A JavaFX egy Java alapú keretrendszer, mely lehetővé teszi asztali alkalmazások fejlesztését modern, grafikus felhasználói felülettel. A JavaFX eszközöket kínál a felhasználói felület komponenseinek létrehozásához, mint például gombok, táblázatok és grafikonok. Támogatja az FXML nyelvet is, ami megkönnyíti a felhasználói felület elkülönítését az alkalmazás logikájától, az alkalmazás vizuális része pedig könnyen testreszabható CSS segítségével.

A JavaFX alkalmazásokban az egyik leggyakrabban alkalmazott architektúra az MVC Model-View-Controller. Ez a minta 3 fő komponenst különít el.

* Model: Az adatokat és üzleti logikát tartalmazza
* View: A felhasználói interfészt képviseli, ezek JavaFX-ben általában FXML fájlokként vannak definiálva.
* Controller: Az a réteg amely kapcsolatot biztosít a modellek és nézetek közt.

Az MVC struktúra előnye, hogy elkülöníti a különböző egységeket, így az alkalmazás karbantartása és bővítése könnyebbé válik. Az FXML pedig, a nézet definiálására különösen jól alkalmazható, mivel lehetővé teszi a felhasználói felület fejlesztését programozási logika nélkül.

1. A feladatról

Az elkészült program egy olyan JavaFX alkalmazás, mely egyrészt a Magyar Nemzeti Bank által, limitált darabszámban kiadott forint érmék adatait tartalmazza, jeleníti meg és engedi meg ezen adatok módosítását, bővítését, másrészt kapcsolatot épít az MNB SOAP szolgáltatásával, így a pontos és aktuális képet kaphatunk a jelenlegi, vagy épp a korábbi árfolyamadatokról. Az alkalmazás harmadik nagyobb funkcionális része az OANDA API-val való integráció, amelyet a devizapiaci (Forex) és más pénzügyi eszközök adataihoz és kereskedési szolgáltatásaihoz való hozzáférésre terveztek. Az API lehetővé teszi, hogy automatizált rendszereket építsünk devizakereskedéshez, valós idejű adatelemzéshez vagy éppen történelmi árfolyamadatok letöltéséhez.

A fejlesztés folyamán, elsődlegesen a funkcionalitás megvalósítását helyeztük előtérbe, emiatt a felhasználói felület egy egyszerű, mégis igényes, átlátható megjelenést kapott.

1. Adatbázis

A feladat megvalósításának egyik kritériuma volt, hogy az adatbázis nem függhet külső adatbázistól, így könnyen mozgatható az elkészült alkalmazás egyik gépről a másikra.

Ennek függvényében SQLite adatbázist használ a program, ami egy könnyű, önállóan működő, beágyazott adatbázis-kezelő rendszer, ellentétben a hagyományos szerver alapú adatbázisokkal, mint például a MySql. Az SQLite nem igényel külön adatbázisszervert, mivel minden adat egyetlen fájlban tárolódik.

Az alkalmazás és az SQLite adatbázis közötti kapcsolódás során úgy döntöttünk, hogy **nem használunk objektum-relációs leképezést (ORM)**, hanem közvetlenül natív SQL-lekérdezésekkel kezeljük az adatbázist. Ez a megközelítés több előnnyel is járhat, különösen kisebb adatbázis-kezelési igényű alkalmazások esetén.

Használatának előnyei közé tartozik:

* Teljes kontroll a lekérdezések felett, ami annyit tesz, hogy pontosan meghatározhatjuk a különböző műveleteket (adatbeolvasás, módosítás), így az optimalizálást is közvetlenül kézben tarthatjuk
* Csökkentett függőségi rendszer. ORM használata esetén további könyvtárakat, eszközöket kell beépíteni az alkalmazásba
* Könnyebb hibakezelés. A natív SQL lekérdezésekkel az adatbázissal kapcsolatos problémák feltárása egyszerűsödik

Az adatbázis műveletek kezelésére létrehoztunk egy databaseHelpert, mely az adatbázishoz való csatlakozást, CRUD műveletek megvalósítását biztosítja.

* 1. Adatbázis felépítése

A feladat megvalósítására 5 alap adatbázis tábla állt rendelkezésünkre, melyek a következők voltak.

* + 1. anyag tábla

A tábla a korlátozott darabszámban kiadott magyar forint érmékhez használt nyersanyagokat tartalmazza. Minden rekord egy konkrét fémet reprezentál. A femid mező, a tábla elsődleges kulcsa, mely azonosítja a fémeket.

A képen szöveg, sor, Betűtípus, képernyőkép látható

Automatikusan generált leírás

* + 1. tervezo tábla

A tervezo tábla a forint érmék tervezőinek adatait tárolja. Minden sor egy konkrét tervezőt reprezentál, beleértve a tervező egyedi azonosítóját és nevét. Ez a tábla a tervezők azonosítására szolgál, amely lehetővé teszi az egyes érmék készítőinek nyomon követését.

A képen szöveg, sor, Betűtípus, szám látható

Automatikusan generált leírás

* + 1. erme tábla

Az erme tábla a limitált darabszámban kiadott magyar forint érmék alapvető adatait tárolja. Minden sor egy érmeadatot reprezentál, annak címletével, tömegével, kiadott darabszámával, valamint a kibocsátás és bevonás dátumával. Ez a tábla az érmék részleteinek, információinak tárolására szolgál.

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

* + 1. akod tábla

Az akod tábla egy kapcsolótábla. Az alapanyagoknak használt fémek, azaz anyag tábla és az érmék adatait tartalmazó erme adatbázistábla adainak kapcsolatait tartalmazza. A kapcsolótáblára azért volt szükség, mert az anyagok és érmék kapcsolata több a többhöz, egy érméhez több alapanyagot is használhattak, valamint egy alapanyagot több érme anyagaként is felhasználtak.

A képen szöveg, Betűtípus, sor, képernyőkép látható

Automatikusan generált leírás

* + 1. tkod tábla

A tkod tábla szintén egy kapcsolótábla, ami a tervezők és érmék kapcsolatait reprezentálja. Az érmék és tervezőik közt szintén több a többhöz a kapcsolat, több tervező is közreműködhetett egy érme megalkotásában, valamint egy-egy tervező több érménél is szerepet játszott.

A képen szöveg, képernyőkép, Betűtípus, sor látható

Automatikusan generált leírás

* + 1. deviza tábla

A devizanemeket nem tartalmazta az importálandó állomány. Erre azért volt szükség, hogy a devizanemeket könnyen és egyszerűen megjeleníthessük a ComboBoxokban. Az MNB adatok letöltésekor, a program letölti az elérhető devizanemeket és beszúrja az esetlegesen új devizákat a táblába.



* + 1. params tábla

Az MNB adatszolgáltatás menüpontban egy felugró ablakkal kiválasztható az eszközön a letöltött adatok mentési helye. Ezt az elérési utat a paraméter táblába mentjük, így a későbbiekben, ha a fájlt módosítani kell, felülírni, vagy beolvasni, nem lesz szükség újra bekérni annak pontos címét, miután az le van tárolva az adatbázisban.

Ez a tábla a program továbbfejlesztésekor további paraméterekkel bővíthető és használható lenne.

A képen szöveg, Betűtípus, sor, szám látható

Automatikusan generált leírás

1. CRUD menüpont

Az alkalmazás menüjének első pontjában az adatbáziskapcsolat formázott a felhasználónak könnyen áttekinthető megjelenítése volt a fő cél.

Az első menüpontban az előző pontban részletezett adatbázis táblák (5 fő tábla) adatait táblázatos formában jeleníthetjük meg.

A képen szöveg, képernyőkép, szám látható

Automatikusan generált leírás

Az adatbázis leginkább látványos megjelenítést biztosító táblája kétségtelenül az érméket reprezentáló tábla. A keresés menüpontban részletes keresést teszünk lehetővé a lenyíló listával, radio gombokkal, jelölőnégyzettel, szöveges beviteli mezőkkel.

A képen szöveg, képernyőkép, szám, szoftver látható

Automatikusan generált leírás

Az adatrögzítés menüpontban szintén az érme tábla van főszerepben. Szöveges beviteli mezőkkel, dátumválasztókkal adhatók meg az adatok és menthetők az adatbázisba.

Érmék adatainak módosítására is lehetőséget biztosítunk az adatmódosítás menüpontban. Itt táblázatos formában újra megjelennek a jelenlegi rekordok, ezzel segítve a módosítandó adatok bevitelét. A táblázaton dupla kattintásra betöltjük a beviteli mezőkbe a jelenlegi adatokat, ezzel a felhasználó munkáját könnyítve, csak a módosítani kívánt értéket kell megadnia.

A képen szöveg, képernyőkép, szám, Párhuzamos látható

Automatikusan generált leírás

Végül a törlés menüpontban az érme azonosítóját kiválasztva törölhetünk rekordokat az adatbázisból. Itt egy biztonsági funkcióként a beviteli mezőben meg kell adni, az „Érme Törlése!” szöveget, ezzel igyekeztünk védekezni a véletlen adatvesztés ellen.

1. MNB adatszolgáltatás menüpont

Az MNB SOAP szolgáltatásához csatlakozhatunk és nyerhetünk ki különböző információkat a feladat által megkövetelt módon.

A SOAP kliens létrehozásában a jaxws-maven-plugint használtuk, amely automatikusan képes Java osztályokat generálni egy WSDL (Web Service Definition Language) fájl alapján. Ez a plugin különösen hasznos SOAP webszolgáltatásokkal való integráció során, mivel a generált osztályok leegyszerűsítik a szolgáltatás elérését. A pluginban megadható, hogy hol érhető el a wsdl, ami jelen esetben a <http://www.mnb.hu/arfolyamok.asmx?wsdl> címen található, megadható továbbá hogy a generált osztályokat milyen package-be mentse.

A generált osztályok és a kontroller közt az MNBSoapClientHelper osztály van segítségünkre. Ebben az osztályban valósítottuk meg többek közt az aktuális árfolyamok lekérdezését, az elérhető devizanemek lekérését is.

A letöltések menüpontban lehetőségünk van egy txt fájlba menteni a letöltött aznapi árfolyamokat. Ezt egyszerűen megtehetjük a Letöltés gombra kattintva és a felugró ablakban kiválasztva a mentés helyét. A fájl alapértelmezetten MNB néven jön létre, de a FileChooser segítségével más néven is menthető. Ahogy korábban az adatbázisok menüpontban említettük, az itt kiválasztott elérési utat mentjük arra az esetre ha a fájlt később módosítani, vagy beolvasni kell.

Részletes letöltés menüpont

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírásA felhasználólónak lehetősége van letölteni egyszerre több árfolyamot is egy fájlba.

A „ListView”-ben akármennyi valuta kiválasztható (ctrl+click).

A „letöltés és mentés” gombra kattintva a felhasználó kiválaszthatja, hogy hova és milyen néven mentse le a szöveges állományt.

A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, menü, tervezés látható

Automatikusan generált leírásA nézetet a Letoltes2Controller vezényli és a downloadMNBExchange2-view.fxml nézetfájl jeleníti meg.

A letöltött fájl tartalma:

Valuta: Valuta

Árfolyamadatok a –Dátum–dátumról:

1 Valuta = XY Ft

Kiválasztott valuták egymás alatt kerülnek kiíratásra.

A vezérlő az adatokat a MNBSoapClientHelper osztálytól veszi.

Az adatok a helper osztályban kerülnek lekérdezésre és formázásra.

Grafikonos megjelenítés menüpont

Ezen a felületen a felhasználó az általa kiválasztott valuta egy hónapra levetített értékeit grafikonos formában jelenítheti meg.

A felhasználó által megadható adatok:

* Valuta: A lenyíló listából
* Megtekinteni kívánt hónap. (YYYY-MM)

A képen szöveg, képernyőkép, szám látható

Automatikusan generált leírás

A nézetet a ChartViewController vezényli és a Chart-view.fxml nézetfájl jeleníti meg.

A vezérlő az adatokat a MNBSoapClientHelper osztálytól veszi.

Az adatok a helper osztályban kerülnek lekérdezésre és formázásra.

A grafikon a javaFX által kínált beépített megoldás.

1. Párhuzamos programvégrehajtás menüpont

A menüpontban szemléltetjük a párhuzamos programvégrehajtást, ahol is egy gomb megnyomására két külön task futása látható. A startMultiTaskAction() metódus indítására két Runnable feladat, egy task1 és egy task2 indul. A két feladat egymástól függetlenül működik, előbbi az indítás óta eltelt másodperceket jeleníti meg, mivel másodpercenként frissíti a felhasználói felületen megjelenő értéket, míg utóbbi értéke 2 másodpercenként módosul, ezzel szemléltetve, hogy a két task külön szálon fut. Az ütemezés az executorService segítségével valósul meg, mely a megadott időközönkánt futtatja a két független task-ot.

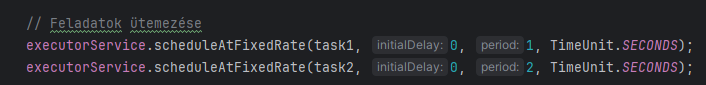
A kód párhuzamos végrehajtást demonstrál a következő módon:

* Két különálló szálon futnak a feladatok, így azok egymástól függetlenül végezhetők el.
* A két feladat eltérő gyakorisággal (1 másodperc és 2 másodperc) hajtódik végre, ami jól mutatja az ütemezés rugalmasságát.
* A GUI-frissítés szálbiztosan történik a Platform.runLater() segítségével, amely biztosítja a stabil működést.

A képen szöveg, képernyőkép, szoftver, számítógép látható

Automatikusan generált leírás

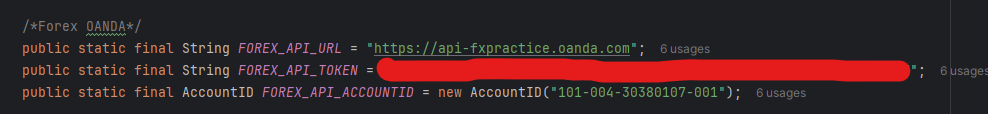
A nézetet a MultitaskController vezényli és a multitask-view.fxml nézetfájl jeleníti meg.

A két Task lényegében két időzítő.

1. Forex menüpont

Oanda API

Az alkalmazás az OANDA által kínált előre megírt modulokat (<https://github.com/oanda/v20-java/tree/master/src/com/oanda/v20> ) használja

A fő azonosítók (token, api url, accountID) a /config/AppConfig fájlban vannak definiálva.

**Számlainformációk almenü**:

A képen szöveg, képernyőkép, szoftver, szám látható

Automatikusan generált leírásA fiók / számla információk egy táblázat **szerű** gridben jelennek meg.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA lekérdezést egy előre megírt funkció kezeli, melyek megjelenítése a /forex/account.fxml fájlban történik.

**Aktuális árak**

Ez a nézet a tőzsdén található és lekérhető, összes valutapárt és az eladási, illetve vételi árat megjeleníti. Valutapárok a nézet megnyitásakor az OandaAPI-ból frissülnek.

Az adathalmaz a fenti keresőmező segítségével szűrhető. A lent elhelyezett „Frissítés” gombbal frissíthető.

**A képen szöveg, képernyőkép, szám, szoftver látható

Automatikusan generált leírás**

Az aktuális árak megjelenítését az actualPrices vezérlő kezeli. A megjelenítést az actualPrices.fxml végzi. Az árakat táblázatos nézetben jeleníti meg a program. A táblázat oszlopai a vezérlő inicializálásánál kerülnek beállításra.

Az adatok lekérését egy, már az Oanda által lefejlesztett metódus végzi.

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírásAz adatok kezelésének egyszerűsítése érdekében létrehozásra került egy modell, amely a PriceData nevet viseli. A frissítés gomb megnyomásával a **onLoadActualPricesButtonClicked** függvény kerül meghívásra.

**Historikus árak**

A menüpontban a tőzsdén található és lekérhető összes valutapárt és az elmúlt 10 óra záró árait mutatja meg táblázatos formában. Valutapárok a nézet megnyitásakor az OandaAPI-ból frissülnek. Az adathalmaz a lent elhelyezett „Frissítés” gombbal frissíthető.A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás

Az aktuális árak megjelenítést a historicPrices vezérlő és historicprices.fxml végzi. Az árakat táblázatos érhetők el, melynek oszlopai a vezérlő inicializálásánál kerülnek beállításra.

Az adatok lekérését itt is egy Oanda által lefejlesztett metódus végzi.

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Automatikusan generált leírásAz adatok kezelésének egyszerűsítése érdekében létrehozásra került egy modell, amely a CandleData nevet viseli. A frissítés gomb megnyomásával a **onLoadHistoricPricesButtonClicked** függvény kerül meghívásra.

**Pozíció nyitás**

A menüpontban a felhasználónak lehetősége van eladási(-) és vételi (+) pozíciót létrehozni.

A felhasználónak az alábbi adatokat kell megadni a pozíció nyitásához:

* Instument: A kiválasztott valutapár. Pld: EUR\_USD, EUR\_AUD. Az első érték az amit cserélni szeretne a felhasználó, a második pedig amire cserélni szeretné.
* Units: Hány cserét szeretne. Ha az összeg negatív az short pozíció (eladós), ha pozitív akkor long pozíció (vétel).

**A képen szöveg, képernyőkép, szám, Betűtípus látható

Automatikusan generált leírás**A fenti táblázatban a felhasználó láthatja, hogy jelenleg milyen nyitott pozíciói vannak.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA pozíció nyitás az OpenTradeController-rel és az openTrade.fxml-el lehetséges. Az aktuális pozíciók betöltését az ActiveTradeService szolgálja.

A pozíciókat táblázatos nézetben jeleníti meg a program. A táblázat oszlopai a vezérlő inicializálásánál kerülnek beállításra. Az „Open Trade” gomb megnyomásával a **onOpenPositionButtonClicked** függvény kerül meghívásra.

**Pozíció zárás**

Itt nyílik lehetőség eladási(-) és vételi (+) pozíciót lezárni.

A felhasználónak az alábbi adatokat kell megadni a pozíció zárásához:

* Trade ID: Egy nyitott pozíció azonosítója. (A lenti táblázatból)

**A képen szöveg, szám, sor, Betűtípus látható

Automatikusan generált leírás**

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA pozíció zárás felületének megjelenítést a CloseTradeController kezeli és a closeTrade.fxml végzi. Az aktuális pozíciók betöltését az ActiveTradeService szolgálja.

A pozíciókat táblázatos nézetben jelenítjük meg, oszlopai a vezérlő inicializálásánál kerülnek beállításra. Az adatok lekérését, illetve a pozíciók zárását, az Oanda által lefejlesztett metódus végzi. A „Close Trade” gomb megnyomásával a **onCloseTradeButtonClick** függvény kerül meghívásra.

**Nyitott pozíciók**

A felületen a felhasználó láthatja az aktuálisan nyitott pozícióit.**A képen szöveg, képernyőkép, képernyő, szám látható

Automatikusan generált leírás** A táblázat, hasonlóan az eddigiekhez, egy táblázatban jelenik meg, ennek oszlopai a vezérlő inicializálásánál kerül beállításra.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírásA pozíciók felületének megjelenítését az ActiveTradesController vezérlő kezeli és az activeTrades.fxml végzi. Az aktuális pozíciók betöltését az ActiveTradeService szolgálja ki.

Az adatok lekérését ebben az esetben is az Oanda által lefejlesztett metódus végzi. A „Frissítés” gomb megnyomásával a **onCloseTradeButtonClick** függvény kerül meghívásra.

Összefoglalás

A projekt megvalósításában igyekeztünk pontról, pontra követni a feladat által támasztott követelményeket.

A teljes fejlesztés alatt, az első lépésétől kezdve a Github verziókezelőt használtuk, ahova rendszeresen felkerültek a változások. A repository a <https://github.com/BobessA/Java_Forint> címen érhető el.

A feladatokat projektmunka módszert alkalmazva végeztük. A Githubon Katona Zsolt felhasználóneve Zsolti44, míg Sándor Adrián felhasználóneve BobessA.

A Git repository gyökér könyvtárában megtalálható a LETOLTES.zip fájl, ami tartalmazza a jar fájlt, valamint az adatok.db fájlt.

Részfeladatok megjelölése:

Katona Zsolt

* MNB letöltés 2 menüpont megvalósítása
* Grafikonon történő megjelenítés
* OANDA API integráció létrehozása
* Forex menüpont teljes megvalósítása
* Párhuzamos programvégrehajtás megvalósítása
* Megjelenítés finomhangolása

Sándor Adrián

* A projekt alapjainak létrehozása
* Adatbázis megvalósítása
* Adatbáziskapcsolat kialakítása (databaseHelper alapjai)
* SOAP kliens generálása jaxws-maven-plugin segítségével
* SOAP helper megvalósítása (alapjai)
* CRUD menüpontok megvalósítása
* MNB letöltés megvalósítása