# JAVA Alkalmazások

# előadás beadandó feladat dokumentáció.

Készítette: Szász András Gábor

Neptun kód: ZYKV5N

GITHUB Repository: <https://github.com/SzaszZYKV5N/JAVAEloadasBeadando>

A feladatot egyedül készítettem el. A munkafolyamat során az elkészült munkarészeket a GITHUB -ra két profilon keresztül töltöttem fel, SzaszAndrasGabor és SzaszZYKV5N felhasználói fiókokat használva.

A JAVA Alkalmazások elnevezésű tárgyhoz kapcsolódóan elkészítettem a JAVA FX beadandó feladatot. A feadat kiválasztásához a meghírdetett témák közül az 1-02-Vízvezeték szerelők adatbázist választottam, ennek megfelelően az alkalmazás is ehhez a témához kapcsolódóan lett kialakítva.

Az alkalmazás elkészítése során az IntelliJ segítségével létrehozott JavaFX sablont tekintettem kiindulási pontnak, melynek átalakítása (refaktorálása) során létrejöttek a MenuApplication.java, a MenuController.java és a menu.fxml állományok.

Az alkalmazás grafikus alapját képező menu.fxml állományban lett kialakítva a kezdőképernyő, amely megjelenés elemei egy vBox típusú külső virtuális doboz keretben lettek összefogva.

A vertikális elrendezésű Vbox-on belül AnchorPane elemen belül egy Menubar vezérlőelem, és egy másik, változó tartalmat megjelenítő AnchorBox került lehelyezésre.

Az elrendezés kialakításához a Scene Builder programot alkalmaztam, illetve az .fxml állományban szövegesen is szerkesztett állományt menu.fxml állományként mentettem el.

A Menubar vezérlőn belül a feladat kiírás szerinti menü és almenü pontok a Menu és Menuitem tagekkel lett létrehozva.

A változó tartalmat megjelenítő AnchorPane elem a rootPane fx azonosítót kapta, az menüpontokban kijelölt tartalmak ebben a konténerelemben jelennek majd meg.

A képen képernyőkép, szöveg, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

A MenuController fájlban az egyes menüpontokhoz az onAction tulajdonságoknál a következő függvény szerint lehet a rootPane konténerbe behívni az egyes menüpontokhoz tartozó grafikus tartalmakat:

@FXML  
private void Ujablak(ActionEvent event) throws IOException {  
 AnchorPane pane = FXMLLoader.*load*(getClass().getResource("second.fxml"));  
 rootPane.getChildren().setAll(pane);  
}

Azonban a beadandó feladat összesen 15 menüpontot tartalmaz, ezért nem célszerű minden menüponthoz új függvényt írni, ehelyett két egyszerű metódust alkalmaztam, melynek lényege, hogy mindegyik menüpont egy eseménykezelőt hív meg, amely megvizsgálja, hogy melyik menuitem hívta meg, és ennek megfelelően választja ki a fetöltendő .fxml fájlt.

@FXML  
private void handleMenuAction(ActionEvent event) {  
 MenuItem source = (MenuItem) event.getSource();  
 String fxmlFileName = "";  
  
 switch (source.getText()) {  
 case "Olvas":  
 fxmlFileName = "olvas.fxml";  
 break;  
 case "Olvas2":  
 fxmlFileName = "olvas2.fxml";  
 break;  
 // További esetek hozzáadása szükség szerint  
 default:  
 System.*out*.println("Ismeretlen menüpont: " + source.getText());  
 return;  
 }  
  
 loadFXMLToRootPane(fxmlFileName);  
}

A betöltést elvégző kódrészlet:

private void loadFXMLToRootPane(String fxmlFileName) {  
 try {  
 FXMLLoader loader = new FXMLLoader(getClass().getResource(fxmlFileName));  
 Parent newContent = loader.load();  
 rootPane.getChildren().setAll(newContent);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
}

## Adatbázis menü

Adatbázis létrehozása és tárolása

Az alkalmazásnak alkalmasnak könnyen hordozhatónak kell lennie, és nem szabad külső adatbázistól függenie. Ezt legegyszerűbben úgy lehet elérni, ha az adatokat vagy egyszerű szöveges állományban, vagy pl. SQLite állományban tároljuk, egy könnyez hozzáférhetőkönyvtárban. Az SQLite adatbázisok kezelése a JAVA rendszerekben támogatott, a tárolandó táblák és adatok pedig egyetlen fájlban megtalálhatók. A létrehozandó SQLite fájlt a c:/adatbazis könyvtárban helyeztem el, illetve hoztam létre. Az elkészült alkalmazás futtatásához a célgépen ezt a könyvtárat lére kell hozni és az adatállományt bele kell másolni.

Az adatbázis szerkezete a következő:

**Táblák:**

***szerelo*** (*az*, *nev*, *kezdev*)

*az* a szerelő azonosítója (szám), ez a kulcs

*nev* a szerelő neve (szöveg)

*kezdev* a szerelő ettől az évtől dolgozik a cégnél (szám)

***munkalap*** (*az*, *bedatum*, *javdatum*, *helyaz*, *szereloaz*, *munkaora*, *anyagar*)

*az* a munkalap azonosítója (szám vagy számláló), ez a kulcs

*bedatum* a munka felvételének dátuma (dátum)

*javdatum* a munka elvégzésének dátuma (dátum)

*helyaz* a munkavégzés helyének azonosítója (szám)

*szereloaz* a munkát elvégző szerelő azonosítója (szám)

*munkaora* a munka elvégzéséhez szükséges idő (szám)

*anyagar* a munka elvégzése során felhasznált anyag ára (szám)

***hely*** (*az*, *telepules*, *utca*)

*az* a javítás helyének azonosítója (szám), ez a kulcs

*telepules* a javítás helyének települése (szöveg)

*utca* a javítás helyének utca, házszáma (szöveg)

A képen szöveg, képernyőkép, Betűtípus, Téglalap látható

Automatikusan generált leírás

DB Browser for SQLite segítségével létrehoztam egy új adatbázist, ésbeimportáltam a txt fájlban tárolt adatokat.

A képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, szám, szoftver látható

Automatikusan generált leírás

Pom.xml-ben hozzáadtam a függőségeket:

<dependency>  
 <groupId>org.xerial</groupId>  
 <artifactId>sqlite-jdbc</artifactId>  
 <version>3.43.0.0</version>  
</dependency>  
<dependency>  
 <groupId>com.github.gwenn</groupId>  
 <artifactId>sqlite-dialect</artifactId>  
 <version>0.1.1</version>  
</dependency>

Az adatbázis tábláknak megfelelően létrehoztam az entitás osztályokat, Hely, Munkalap, Szerelo néven konstruktorokkal, getterekkel és setterekkel.

Létrehoztam egy OlvasDAO. java osztályt a táblázatban megjelenítendő adatok részére. Az ebben az osztályban meghatározott változók az összetett lekérdezéstábláknál lettek felhasználva.

Az olvas.fxml-ben egy a menu.fxml-be beágyazandó felület lett kialakítva, AnchorPane konténeren belül. A táblázat kilakítását a főbb méretek miatt itt alakítottam ki.

A képen szöveg, képernyőkép, szoftver, Betűtípus látható

Automatikusan generált leírás

A menüponthoz önálló kontrollert hoztam létre OlvasController néven. Ebben a fájlban történik meg a táblázat oszlopváltozóinak és konfigurációjának kialakítása. Az initialize() lefutása során a táblázat előre beállított értékei törlődnek, majd megtörténik a táblázat oszlopainak és mezőtípusainak beállítása.

Ezt követően automatikusan meghívja a Tablabovit() metódus, ami feltölti a táblázatot.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

A táblázat sorait a következő formátumú osztály alapján állítjuk elő:

OlvasDAO newOlvasDAO = new OlvasDAO(1,"Budapest", "Podhorszky utca 68"," Kiss Lajos",3,15000);

Az SQL adatbázisból egy mindhárom tábla adatait feldolgozó lekérdezést készítettem, egyszerü jdbc módszerével.

OlvasDAO ujsor;  
Connection conn = DriverManager.*getConnection*("jdbc:sqlite:/c:/adatbazis/javabead.db");  
Statement stmt = conn.createStatement();  
ResultSet rs;  
rs = stmt.executeQuery(  
 "SELECT m.helyaz, h.telepules, h.utca, sz.nev, m.munkaora, m.anyagar " +  
 "FROM szerelo sz, munkalap m, hely h " +  
 "WHERE sz.az = m.szereloaz AND m.helyaz = h.az"+" ORDER BY m.helyaz"  
);  
while(rs.next()){  
 ujsor = new OlvasDAO(rs.getInt("helyaz"), rs.getString("telepules"), rs.getString("utca"),rs.getString("nev"), rs.getInt("munkaora"), rs.getInt("anyagar") );  
 tablazat.getItems().add(ujsor);  
  
}

Eredménye:

A képen szöveg, képernyőkép, szoftver, szám látható

Automatikusan generált leírás

## SoapKliens menü

## Párhuzamos menü

## Forex menü