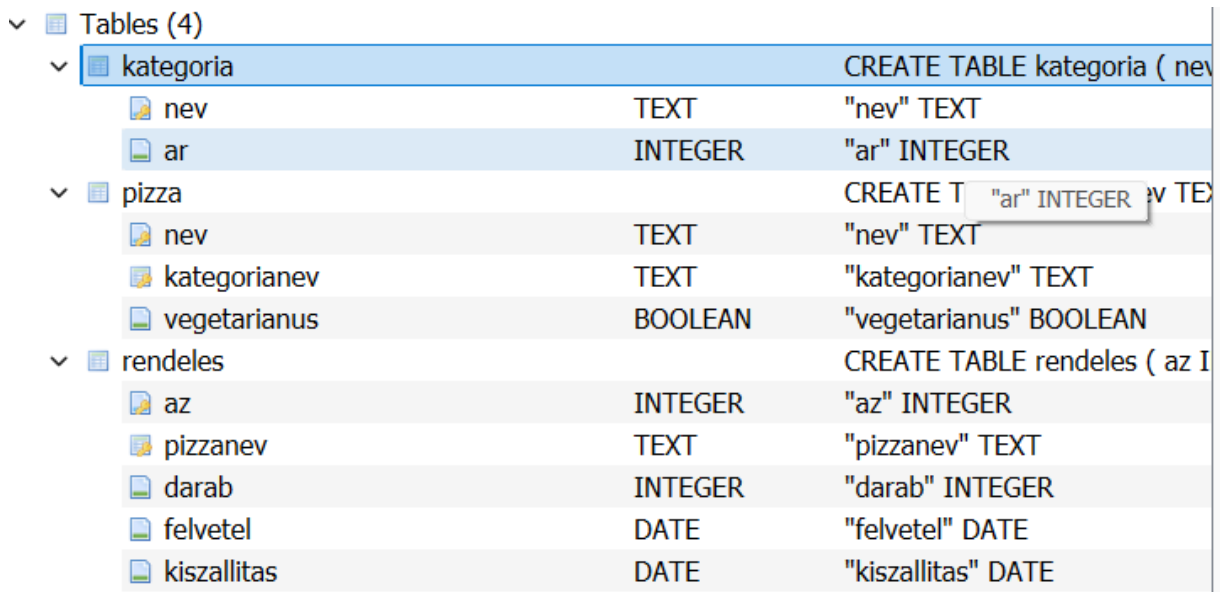


Java Előadás Beadandó

Github Link: <https://github.com/luczay/JavaElodasBeadando>

Adatbázis megvalósítása

Az adatbázisunk 3 táblából áll, pizza, rendelés és kategória. Ezeket az 1. ábrán lehet megtekinteni. Az adatbázist SQLite-ban csináltuk, így nem kell egy külön szerveren adatbázist futtatni, hanem szimplán az adatok.db fájl elég az adatbázis megvalósításához, és ezt lehet hostnak megadni az sql beállításainál. A beállítást tartalmazó fájl a 2. ábra tartalmazza.



Tables (4)		
▼ categoria	CREATE TABLE categoria (nev	
nev	TEXT	"nev" TEXT
ar	INTEGER	"ar" INTEGER
▼ pizza	CREATE T "ar" INTEGER ;v TE	
nev	TEXT	"nev" TEXT
kategorianev	TEXT	"kategorianev" TEXT
vegetarianus	BOOLEAN	"vegetarianus" BOOLEAN
▼ rendelés	CREATE TABLE rendelés (az I	
az	INTEGER	"az" INTEGER
pizzanev	TEXT	"pizzanev" TEXT
darab	INTEGER	"darab" INTEGER
felvetel	DATE	"felvetel" DATE
kiszallitas	DATE	"kiszallitas" DATE

1. ábra

```
private static final String URL = "jdbc:sqlite:C:\\Users\\plaur\\Downloads\\Adolf\\java alkalmazasok\\JavaElodasBeadando\\adatok.db";
```

2. ábra

Az adatbázist a DbManager osztályból lehet elérni, itt vannak metódusok a különböző lekérdezésekhez. Az osztály egy része lenti ábrán látható, a többi részére pedig a feladatok megoldásainál majd kitérünk.

```

public class DbManager {
    private static final String URL = "jdbc:sqlite:C:\\Users\\plaur\\Downloads\\Adolf\\java_alkalmazasok\\JavaFoadasBeadando\\adatok.db";

    private static Connection connect() throws SQLException {
        return DriverManager.getConnection(URL);
    }

    public static void createPizza(String nev, String kategorianev, boolean vegetarianus) {
        String query = "INSERT INTO pizza (nev, kategorianev, vegetarianus) VALUES (?, ?, ?)";

        try (Connection conn = connect();
            PreparedStatement pstmt = conn.prepareStatement(query)) {

            pstmt.setString(1, nev);
            pstmt.setString(2, kategorianev);
            pstmt.setBoolean(3, vegetarianus);

            pstmt.executeUpdate();
        } catch (SQLException sqlException) {
        }
    }

    public static void modifyPizza(String nev, String newName, String newKategorianev, Boolean newVegetarianus) {
        StringBuilder queryBuilder = new StringBuilder("UPDATE pizza SET ");
        boolean hasCategory = newKategorianev != null;
        boolean hasVegetarian = newVegetarianus != null;
        boolean hasNewName = newName != null;

        if (hasCategory) queryBuilder.append("Kategorianev = ?");
    }
}

```

3. ábra

Főmenü leírása

Az applikáció kezdő oldala egy főmenüt tartalmaz. Minden egyes menüpont egy adott feladat vagy részfeladat megoldását tartalmazza.



4. ábra

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.layout.VBox?>

<VBox alignment="CENTER" spacing="20.0" xmlns:fx="http://javafx.com/javafx"
      fx:controller="com.beadando.eloadasbeadandoui.MainMenuController"
      prefWidth="400.0" prefHeight="600">
  <padding>
    <Insets bottom="20.0" left="20.0" right="20.0" top="20.0"/>
  </padding>
  <Button text="1. Feladat - Olvas" onAction="#onFeladat1aClick"/>
  <Button text="1. Feladat - Olvas2" onAction="#onFeladat1bClick"/>
  <Button text="1. Feladat - Ír" onAction="#onFeladat1cClick"/>
  <Button text="1. Feladat - Módosít" onAction="#onFeladat1dClick"/>
  <Button text="1. Feladat - Töröl" onAction="#onFeladat1eClick"/>
  <Button text="2. Feladat" onAction="#onFeladat2Click"/>
  <Button text="3. Feladat" onAction="#onFeladat3Click"/>
  <Button text="4. Feladat" onAction="#onFeladat4Click"/>
</VBox>

```

5. ábra

```

7 public class MainMenuController {  luczay *
8
9     @FXML no usages  luczay
10    protected void onFeladat1cClick() throws IOException {
11        BeadandoUIApplication.switchScene( fxmlFile: "feladat1c.fxml", title: "1. Feladat - Ír");
12    }
13
14    @FXML no usages  luczay
15    protected void onFeladat1dClick() throws IOException {
16        BeadandoUIApplication.switchScene( fxmlFile: "feladat1d.fxml", title: "1. Feladat - Módosít");
17    }
18
19    @FXML no usages  luczay
20    protected void onFeladat1eClick() throws IOException {
21        BeadandoUIApplication.switchScene( fxmlFile: "feladat1e.fxml", title: "1. Feladat - Töröl");
22    }
23
24    @FXML no usages  luczay *
25    protected void onFeladat2cClick() throws IOException {
26        BeadandoUIApplication.switchScene( fxmlFile: "feladat3.fxml", title: "2. Feladat");
27    }
28
29    @FXML no usages  luczay *
30    protected void onFeladat3cClick() throws IOException {
31        BeadandoUIApplication.switchScene( fxmlFile: "feladat2.fxml", title: "3. Feladat");
32    }
33
34    @FXML no usages  luczay
35    protected void onFeladat4cClick() throws IOException {

```

6. ábra

1. Feladat

Az Olvas menüt az 1. Feladat – Olvas menüpont alatt lehet elérni. A három tábla együttes lekérdezését egy hosszú sql lekérdezéssel valósítottuk meg, amiben JOIN-okat használtunk, hogy összefűzzük a 3 táblát. Az adatokat TableView-ban jelenítettük meg, az egyes cellák értékeit a controller osztályból töltöttük fel.

1. Feladat - Olvas

Kategória Néve	Kategória Ár	Pizza Néve	Vegetáriánus	Rendelés Azonosító	Darab	Felvétel Idő	Kiszállítás Idő	
apród	850	Caribi	false	41	1	2005.11.13 7:39:00	2005.11.13 8:05:00	
apród	850	Caribi	false	77	1	2005.11.14 3:29:00	2005.11.14 4:29:00	
apród	850	Caribi	false	260	1	2005.11.17 23:05:00	2005.11.18 1:09:00	
apród	850	Caribi	false	280	1	2005.11.18 9:59:00	2005.11.18 12:18:00	
apród	850	Caribi	false	311	1	2005.11.19 1:49:00	2005.11.19 3:51:00	
apród	850	Caribi	false	355	1	2005.11.20 0:57:00	2005.11.20 3:55:00	
apród	850	Caribi	false	359	1	2005.11.20 2:25:00	2005.11.20 3:21:00	
apród	850	Caribi	false	542	1	2005.11.23 23:29:00	2005.11.24 2:12:00	
apród	850	Caribi	false	544	1	2005.11.24 0:03:00	2005.11.24 2:17:00	
apród	850	Caribi	false	572	1	2005.11.24 13:35:00	2005.11.24 15:05:00	
apród	850	Caribi	false	607	1	2005.11.25 3:23:00	2005.11.25 4:19:00	
apród	850	Caribi	false	692	1	2005.11.26 19:17:00	2005.11.26 20:44:00	

Főmenü

7. ábra

```

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.layout.VBox?>

<VBox alignment="CENTER" spacing="20.0" xmlns="http://javafx.com/javafx"
  xmlns:fx="http://javafx.com/fxml"
  fx:controller="com.beadando.eloadasbeadandoui.Feladat1aController"
  prefWidth="850.0" prefHeight="400.0">
  <padding>
    <Insets bottom="20.0" left="20.0" right="20.0" top="20.0"/>
  </padding>

  <TableView fx:id="rendelesTable">
    <columns>
      <TableColumn fx:id="kategoriaNevColumn" text="Kategória Néve"/>
      <TableColumn fx:id="kategoriaArColumn" text="Kategória Ár"/>
      <TableColumn fx:id="pizzaNevColumn" text="Pizza Néve"/>
      <TableColumn fx:id="vegetarianusColumn" text="Vegetáriánus"/>
      <TableColumn fx:id="rendelesAzColumn" text="Rendelés Azonosító"/>
      <TableColumn fx:id="darabColumn" text="Darab"/>
      <TableColumn fx:id="felvetelColumn" text="Felvétel Idő"/>
      <TableColumn fx:id="kiszallitasColumn" text="Kiszállítás Idő"/>
    </columns>
  </TableView>

  <Button text="Főmenü" onAction="#onBackToMenuClick"/>

```

8. ábra

```

@FXML no usages ▲ luczay
protected void initialize() {
    kategoriaNevColumn.setCellValueFactory(new PropertyValueFactory<>( s: "kategoriaNev"));
    kategoriaArColumn.setCellValueFactory(new PropertyValueFactory<>( s: "kategoriaAr"));
    pizzaNevColumn.setCellValueFactory(new PropertyValueFactory<>( s: "pizzaNev"));
    vegetarianusColumn.setCellValueFactory(new PropertyValueFactory<>( s: "vegetarianus"));
    rendelesAzColumn.setCellValueFactory(new PropertyValueFactory<>( s: "rendelesAz"));
    darabColumn.setCellValueFactory(new PropertyValueFactory<>( s: "darab"));
    felvetelColumn.setCellValueFactory(new PropertyValueFactory<>( s: "felvetel"));
    kiszallitasColumn.setCellValueFactory(new PropertyValueFactory<>( s: "kiszallitas"));

    loadTableData();
}

private void loadTableData() { 1 usage ▲ luczay
    List<RendelesExpanded> rendelesList = DbManager.getRendelesekExpandedAll();
    ObservableList<RendelesExpanded> observableList = FXCollections.observableArrayList(rendelesList);

    rendelesTable.setItems(observableList);
}

@FXML ▲ luczay
protected void onBackToMenuClick() throws IOException {
    try {
        BeadandoUIApplication.switchScene( fxmlFile: "main-menu.fxml", title: "Főmenü");
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

```

9. ábra

```

124 @ public static List<RendelesExpanded> aggregateQuery(String pizzaName, boolean vegetarianus, String category, String date) { 1 us
125     List<RendelesExpanded> temp_results = new ArrayList<>();
126     List<RendelesExpanded> final_results = new ArrayList<>();
127
128     String query = "SELECT * FROM rendeles r JOIN pizza p ON r.pizzanev = p.nev JOIN kategoria k ON p.kategorianev = k.nev";
129
130     try (Connection conn = connect();
131         PreparedStatement stmt = conn.prepareStatement(query)) {
132
133         ResultSet rs = stmt.executeQuery();
134         while (rs.next()) {
135             RendelesExpanded expanded = new RendelesExpanded(
136                 rs.getString( columnName: "kategorianev"),
137                 rs.getInt( columnName: "ar"),
138                 rs.getString( columnName: "pizzanev"),
139                 rs.getBoolean( columnName: "vegetarianus"),
140                 rs.getInt( columnName: "az"),
141                 rs.getInt( columnName: "darab"),
142                 rs.getString( columnName: "felvetel"),
143                 rs.getString( columnName: "kiszallitas")
144             );
145             temp_results.add(expanded);
146         }
147
148         for (RendelesExpanded result : temp_results)
149         {
150             if (
151                 result.isVegetarianus() == vegetarianus

```

10. ábra

Az Olvas2 menü az 1. Feladat – Olvas2 menüpont alatt található. Valami oknál fogva az sql kód nem működik megfelelően, mivel az összes adatot visszaadja az adatbázisból. Ezért az összes adatot lekérjük, majd a szűrésnél megadott

értékek alapján mi magunk szűrjük ki a megfelelő rekordokat a forráskódban. Az így kapott adatokat pedig egy statikus mezőben tároljuk el a controller osztályban, így mikor az az osztály meghívódik, az értékek már elérhetőek lesznek.

1. Feladat - Olvas2

Dátum (yyyy-MM-dd):

yyyy-MM-dd

Pizza Neve:

Barbecue chicken

Babos

Barbecue chicken

Betyáros

Caribi

Country

Csabesz

Csupa sajt

Erdő kapitánya

Erős János

Excellent

☐ apród

☐ főnemes

☐ király

☐ lovag

Küld

Főmenü

11. ábra


```

@FXML no usages luczay +1*
protected void onSubmitClick() {
    String selectedPizza = pizzaComboBox.getValue();
    boolean vegetarianus = vegetarianCheckBox.isSelected();

    RadioButton selectedCategoryButton = (RadioButton) categoryToggleGroup.getSelectedToggle();
    String selectedCategory = (selectedCategoryButton != null) ? selectedCategoryButton.getText() : null;

    if (selectedPizza == null || selectedCategory == null) {
        showAlert( title: "Error", message: "Kérem töltsön ki minden mezőt!");
        return;
    }

    // Query the database
    List<RendelesExpanded> result = DbManager.aggregateQuery(selectedPizza, vegetarianus, selectedCategory, dateInputField.getText());

    try {
        Feladat1bHelperController.setResults(result);
        BeadandoUIApplication.switchScene( fxmleFile: "feladat1bhelper.fxml", title: "Eredmények");
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

```

12. ábra

```

public static List<RendelesExpanded> aggregateQuery(String pizzaName, boolean vegetarianus, String category, String date) {
    List<RendelesExpanded> temp_results = new ArrayList<>();
    List<RendelesExpanded> final_results = new ArrayList<>();

    String query = "SELECT * FROM rendeles r JOIN pizza p ON r.pizzanev = p.nev JOIN kategoria k ON p.kategorianevev = k.nev";

    try (Connection conn = connect();
        PreparedStatement stmt = conn.prepareStatement(query)) {

        ResultSet rs = stmt.executeQuery();
        while (rs.next()) {
            RendelesExpanded expanded = new RendelesExpanded(
                rs.getString( columnName: "kategorianevev"),
                rs.getInt( columnName: "ar"),
                rs.getString( columnName: "pizzanev"),
                rs.getBoolean( columnName: "vegetarianus"),
                rs.getInt( columnName: "az"),
                rs.getInt( columnName: "darab"),
                rs.getString( columnName: "felvetel"),
                rs.getString( columnName: "kiszallitas")
            );
            temp_results.add(expanded);
        }

        for (RendelesExpanded result : temp_results)
        {

```

13. ábra

```

for (RendelesExpanded result : temp_results)
{
    if (
        result.isVegetarianus() == vegetarianus
        && result.getPizzaNev().equals(pizzaName)
        && result.getKategoriaNev().equals(category)
    ) {
        DateTimeFormatter formatter1 = DateTimeFormatter.ofPattern("yyyy.MM.dd");
        DateTimeFormatter formatter2 = DateTimeFormatter.ofPattern("yyyy-MM-dd");

        LocalDate date1 = LocalDate.parse(result.felvetel.split(regex: " ")[0], formatter1);

        LocalDate date2 = LocalDate.parse(date, formatter2);

        if (date1.isAfter(date2)) {
            final_results.add(result);
        }
    }
}

catch (SQLException sqlException) {

```

14. ábra

Az Ír, Töröl és Módosít menü hasonló menüpontban található, mint az előző feladatok megoldásai. Mivel ezek szimpla feladatok, így a megvalósításról melléeltünk csak képeket.

15. ábra

```

public class FeladaticController { 1 usage 1 HamNorb*

    private void loadKategoriaDropdown() { 1 usage 1 HamNorb
        List<String> categoryNames = DbManager.getAllCategoryNames();
        if (categoryNames != null && !categoryNames.isEmpty()) {
            kategoriaDropdown.setItems(FXCollections.observableArrayList(categoryNames));
        } else {
            System.out.println("Failed to load categories or no categories found.");
        }
    }
}

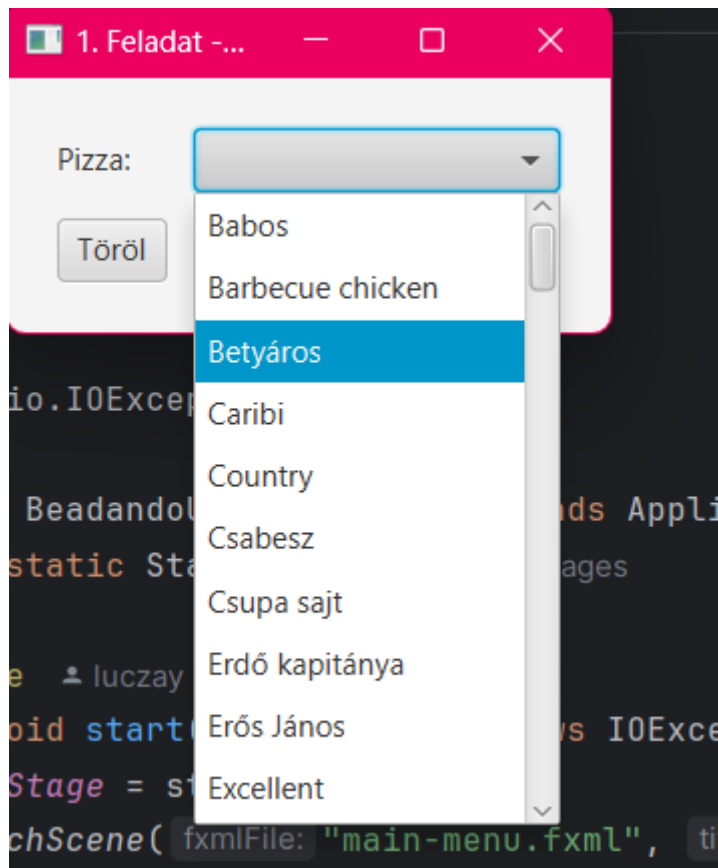
@FXML no usages 1 HamNorb*
protected void onAddPizzaClick() {
    String pizzaNev = pizzaNevField.getText();
    String kategoriaNev = kategoriaDropdown.getValue();
    boolean vegetarianus = vegetarianusCheckBox.isSelected();

    if (pizzaNev == null || kategoriaNev == null) {
        System.out.println("All fields must be filled out.");
        return;
    }

    try {
        DbManager.createPizza(pizzaNev, kategoriaNev, vegetarianus);
        System.out.println("Pizza successfully added!");
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println("Failed to add pizza.");
    }
}

```

16. ábra



17. ábra

```
public class Feladat1Controller {
    private HamNorb *
    public void initialize() {
        loadPizzaDropdown();
    }

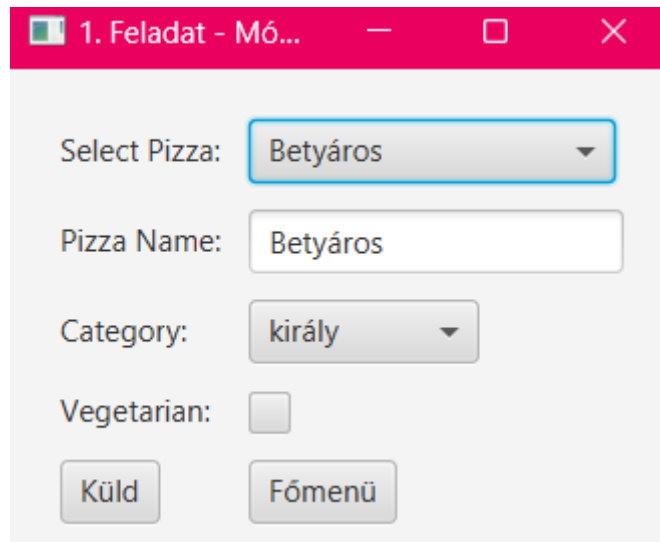
    private void loadPizzaDropdown() {
        List<String> pizzaNames = DbManager.getAllPizzaNames();
        pizzaDropdown.setItems(FXCollections.observableArrayList(pizzaNames));
    }

    @FXML
    protected void onSubmitClick() {
        String selectedPizzaName = pizzaDropdown.getValue();
        if (selectedPizzaName == null || selectedPizzaName.isEmpty()) {
            System.out.println("No pizza selected.");
            return;
        }

        try {
            DbManager.deletePizza(selectedPizzaName);
            System.out.println("Pizza successfully deleted!");
            pizzaDropdown.getItems().remove(selectedPizzaName);
            pizzaDropdown.setValue(null);
        } catch (Exception e) {
        }
    }

    @FXML
    protected void onBackToMenuClick() throws IOException {
    }
}
```

18. ábra



1. Feladat - Mó...

Select Pizza: Betyáros

Pizza Name: Betyáros

Category: király

Vegetarian: ☐

Küld Főmenü

19. ábra

```
public class FeladatIdController { HamNorb +1*

    private void onPizzaSelected() { 1 usage HamNorb +*
        String selectedPizzaName = pizzaDropdown.getValue();
        if (selectedPizzaName != null) {
            // Fetch pizza details from database
            selectedPizza = DbManager.getPizzaDetails(selectedPizzaName);
            if (selectedPizza != null) {
                pizzaNevField.setText(selectedPizza.getNev());
                kategoriaDropdown.setValue(selectedPizza.getKategorianev());
                vegetarianusCheckBox.setSelected(selectedPizza.isVegetarianus());
            }
        }
    }

    @FXML no usages HamNorb +1*
    protected void onSubmitClick() {
        if (selectedPizza == null) {
            System.out.println("Nincs kiválasztott pizza.");
            return;
        }

        String newPizzaNev = pizzaNevField.getText();
        String newKategoriaNev = kategoriaDropdown.getValue();
        boolean newVegetarianus = vegetarianusCheckBox.isSelected();

        try {
            DbManager.modifyPizza(selectedPizza.getNev(), pizzaNevField.getText(), newKategoriaNev, newVegetarianus);
        }
    }
}
```

20. ábra

3. Feladat

Mi a párhuzamos programvégrehajtást színek változtatásával mutatjuk be. Az oldalon 2 label látható, mindegyik label más időpontban vált színt, és a színek teljesen random vannak kiválasztva. A random színeket úgy valósítottuk meg, hogy a random színek kódjait a random függvénnyel választjuk ki. Párhuzamosság eléréhez pedig Threadet használtunk.



21. ábra

```

public class Feladat2Controller { /* luczay */

    @FXML /* no usages */ /* luczay */
    public void initialize() {
        executor = Executors.newFixedThreadPool(/* nThreads: 2 */);

        executor.submit(() -> changeLabelColor(label1, /* interval: 1000 */));

        executor.submit(() -> changeLabelColor(label2, /* interval: 2000 */));
    }

    private void changeLabelColor(Label label, int interval) { /* 2 usages */ /* luczay */
        while (true) {
            try {
                Color randomColor = new Color(Math.random(), Math.random(), Math.random(), /* v3: 1.0 */);

                Platform.runLater(() -> label.setStyle("-fx-font-size: 24px; -fx-background-color: " + toHexString(randomColor) + "; -fx-padding: 10px;"));

                Thread.sleep(interval);
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
                break;
            }
        }
    }

    private String toHexString(Color color) { /* 1 usage */ /* luczay */
        int r = (int) (color.getRed() * 255);
        int g = (int) (color.getGreen() * 255);
    }
}

```

22. ábra