**Tarea 2 – No evaluable: Ampliación con Componentes Avanzados y Enlace de Datos**



**DESARROLLO DE INTERFACES**

**Alejandro González Venegas**

**DAM – 2**

Duke chalkboard

**ÍNDICE:**

1. MODIFICACIÓN PROYECTO ANTERIOR
2. FUNCIONAMIENTO

Entorno:

* Sistema operativo: Windows 10
* IDE: Eclipse 2023-09 (v4.29.0)
* Versión Java: Open JDK 21
* Versión Java FX: Open Java FX 21

## MODIFICACIÓN PROYECTO ANTERIOR

Para este ejercicio voy a reutilizar parte del código de la práctica anterior, pero haré bastantes cambios.

**mainview.FXML**

* Sustituyo la clase contenedora *BorderPane* por *VBox*; esto me permite organizar mejor los componentes en la escena.
* Introduzco dos etiquetas *Label* con sus respectivos *TextField* para permitir al usuario introducir su nombre y su apellido.
* Coloco el *Button* para el envío de datos justo a continuación de los dos campos de texto.
* Introduzco una *TableView* que contendrá los datos que se vayan recuperando.
* Coloco un segundo *Button* que permita al usuario borrar elementos.

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

<?**import** javafx.scene.layout.VBox?>

<?**import** javafx.scene.control.TextField?>

<?**import** javafx.scene.control.Button?>

<?**import** javafx.scene.control.TableView?>

<?**import** javafx.scene.control.TableColumn?>

<?**import** javafx.scene.control.TableColumn?>

<?**import** javafx.scene.control.Label?>

<**VBox** alignment=*"CENTER"* spacing=*"20.0"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"Controller"*>

<**Label** text=*"Enter your name"* />

<**TextField** fx:id=*"name"* maxWidth=*"149"*/>

<**Label** text=*"Enter your surname"* />

<**TextField** fx:id=*"surname"* maxWidth=*"149"* />

<**Button** text=*"New user"* onAction=*"#newUser"*/>

<**TableView** fx:id=*"tw"* maxWidth=*"300"*>

<**columns**>

<**TableColumn** text=*"Name"* prefWidth=*"149"* fx:id=*"colName"*/>

<**TableColumn** text=*"Surname"* prefWidth=*"149"* fx:id=*"colSurname"*/>

</**columns**>

</**TableView**>

<**Button** text=*"Delete user"* onAction=*"#delUser"*/>

</**VBox**>

* Recuperaré la información de los *TextField* en el controlador gracias a los identificadores *name* y *surname*;
* Con el método *newUser* crearé al usuario nuevo en el controlador.
* Identificaré los datos de cada columna de la *TableView* con *colName* y *colSurname*.
* Con el método *delUser* borraré al usuario en el controlador.

**MainApplication.java**

* Amplío el tamaño de la *Scene* a 350 x 600 para adaptarlo a la nueva funcionalidad.

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.stage.Stage;

public class MainApplication extends Application {

*@Override*

public void start(Stage primaryStage) throws Exception {

Parent root = FXMLLoader.*load*(getClass().getResource("mainview.fxml"));

primaryStage.setTitle("Desarrollo de Interfaces - DAM2");

primaryStage.setScene(new Scene(root, 350, 600));

primaryStage.show();

}

public static void main(String[] args) {

*launch*(args);

}

}

**NUEVO**: **User.java**

* Defino una clase *User*, que tiene como atributos el nombre y el apellido, con un constructor que recibe ambos parámetros.

public class User {

private String name, surname;

public User(String name, String surname) {

this.name = name;

this.surname = surname;

}

public String getName() {

return name;

}

public String getSurname() {

return surname;

}

public void setSurname(String surname) {

this.surname = surname;

}

public void setName(String name) {

this.name = name;

}

}

**Controller.java**

* Sustituyo el anterior campo de texto por atributos nuevos: dos *TextField* (uno para el nombre, otro para el apellido), una *TableView* con dos *TableColumn*, y una *ObservableList* de objetos de tipo *User*.
* Implemento el método *initialize()* para precargar siempre en la pantalla la lista de datos dentro de la *TableView*.
* Elimino el método que modelaba la acción del único *Button* e implemento dos métodos: uno para crear nuevo usuario (*newUser*) y otro para borrar al usuario seleccionado (*delUser*).

import java.util.NoSuchElementException;

import java.util.Optional;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.control.Alert;

import javafx.scene.control.TextField;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.control.Alert.AlertType;

import javafx.scene.control.ButtonType;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

public class Controller {

*@FXML*

public TextField name;

*@FXML*

public TextField surname;

*@FXML*

public TableView<User> tw;

*@FXML*

private TableColumn<User, String> colName;

*@FXML*

private TableColumn<User, String> colSurname;

*@FXML*

private ObservableList<User> users;

*@FXML*

public void initialize() {

users = FXCollections.*observableArrayList*();

this.colName.setCellValueFactory(new PropertyValueFactory<>("name"));

this.colSurname.setCellValueFactory(new PropertyValueFactory<>("surname"));

}

public void newUser() {

Alert alerta = new Alert(*AlertType*.***CONFIRMATION***, "New user request");

alerta.setHeaderText("Please confirm the new user request");

alerta.setContentText("NAME: " + name.getText() + "\nSURNAME: " + surname.getText());

Optional<ButtonType> result = alerta.showAndWait();

if (result.isPresent() && result.get() == ButtonType.***OK***)

{

User u1 = new User(name.getText(), surname.getText());

this.users.add(u1);

this.tw.setItems(users);

Alert infoNew = new Alert(*AlertType*.***INFORMATION***);

infoNew.setHeaderText("Data loaded!");

infoNew.showAndWait();

}

else {

Alert infoNew = new Alert(*AlertType*.***INFORMATION***);

infoNew.setHeaderText("No changes!");

infoNew.showAndWait();

}

}

public void delUser(ActionEvent event) {

this.tw.getItems();

ObservableList<User> singleUser;

try {

singleUser = this.tw.getSelectionModel().getSelectedItems();

if (singleUser.isEmpty())

{

Alert errAlert = new Alert(*AlertType*.***ERROR***);

errAlert.setHeaderText("Nothing to delete!");

errAlert.showAndWait();

}

else {

singleUser.forEach(users::remove);

Alert infoNew = new Alert(*AlertType*.***INFORMATION***);

infoNew.setHeaderText("User deleted from system!");

infoNew.showAndWait();

}

} catch (NoSuchElementException e) {

Alert errAlert = new Alert(*AlertType*.***WARNING***);

errAlert.setHeaderText("Empty database");

errAlert.showAndWait();

}

}

}

* Véase que, tanto en el alta como en la eliminación de datos, se lanzan cuadros de diálogo modales para que el usuario pueda confirmar o cancelar el alta, para informarle de que se ha realizado con éxito la operación solicitada, o para realizar una advertencia.

## FUNCIONAMIENTO

La pantalla principal presenta el siguiente aspecto:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se espera que el usuario introduzca datos en los dos *TextField* de la parte superior. La aplicación muestra un cuadro de diálogo para pedir confirmación al usuario antes de introducir la información en el sistema.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

El usuario acepta, otro cuadro de diálogo le confirma la operación, y se regresa a la pantalla principal.

Tabla

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Tabla

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica

Descripción generada automáticamenteEl usuario también puede cancelar el proceso si lo desea. Un ejemplo:

La eliminación de usuarios se hace seleccionado con el ratón la fila sobre la que se desea actuar. He insertado algunos registros a modo de ejemplo:

Tabla

Descripción generada automáticamenteLas sucesivas pulsaciones del botón de borrado van eliminando registros; la eliminación se confirma al usuario con un cuadro de diálogo como el siguiente:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteEn la lógica del controlador, ha sido necesario manejar una *NoSuchElementException*, que se lanza al tratar de recuperar los datos de la *ObservableList* y no poder cargar nada. Si ya no quedan registros, al cargar la *TableView* se captura la excepción y se emite una alerta de advertencia:

Y si el usuario tratase de borrar sin seleccionar nada (o ya no hubiera nada que seleccionar), se lanzaría otro cuadro de diálogo:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente