1. Create a simple login form using JavaFX.

This program demonstrates how to create a simple login form using JavaFX. The form includes text fields for entering the username and password, and a button for submitting the login information. If the credentials are correct, a message will be displayed; otherwise, an error message will appear.

import javafx.application.Application;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.scene.layout.GridPane;

import javafx.stage.Stage;

public class LoginForm extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Login Form");

GridPane grid = new GridPane();

grid.setHgap(10);

grid.setVgap(10);

Label userLabel = new Label("Username:");

TextField userTextField = new TextField();

Label passLabel = new Label("Password:");

PasswordField passField = new PasswordField();

Button loginButton = new Button("Login");

Label messageLabel = new Label();

loginButton.setOnAction(e -> {

String username = userTextField.getText();

String password = passField.getText();

if ("admin".equals(username) && "password".equals(password)) {

messageLabel.setText("Login Successful!");

} else {

messageLabel.setText("Login Failed. Try again.");

}

});

grid.add(userLabel, 0, 0);

grid.add(userTextField, 1, 0);

grid.add(passLabel, 0, 1);

grid.add(passField, 1, 1);

grid.add(loginButton, 1, 2);

grid.add(messageLabel, 1, 3);

Scene scene = new Scene(grid, 300, 200);

primaryStage.setScene(scene);

primaryStage.show();

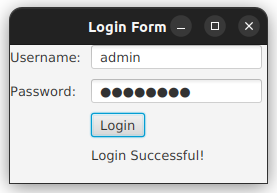
}

public static void main(String[] args) {

launch(args);

}

}



1. Design a GUI with a FlowPane to arrange buttons and a BorderPane to place a menu bar at the top and buttons at the center.

This program demonstrates a JavaFX GUI using a FlowPane to arrange buttons and a BorderPane to organize a menu bar at the top and buttons in the center, creating a structured layout.

import javafx.application.Application;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Menu;

import javafx.scene.control.MenuBar;

import javafx.scene.control.MenuItem;

import javafx.scene.layout.BorderPane;

import javafx.scene.layout.FlowPane;

import javafx.stage.Stage;

public class FlowPaneBorderPaneExample extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("FlowPane and BorderPane Example");

// MenuBar at the top

MenuBar menuBar = new MenuBar();

Menu fileMenu = new Menu("File");

MenuItem openItem = new MenuItem("Open");

MenuItem saveItem = new MenuItem("Save");

MenuItem exitItem = new MenuItem("Exit");

fileMenu.getItems().addAll(openItem, saveItem, exitItem);

menuBar.getMenus().add(fileMenu);

// FlowPane with buttons in the center

FlowPane flowPane = new FlowPane();

flowPane.setHgap(10);

flowPane.setVgap(10);

Button button1 = new Button("Button 1");

Button button2 = new Button("Button 2");

Button button3 = new Button("Button 3");

flowPane.getChildren().addAll(button1, button2, button3);

// BorderPane layout

BorderPane borderPane = new BorderPane();

borderPane.setTop(menuBar);

borderPane.setCenter(flowPane);

Scene scene = new Scene(borderPane, 400, 300);

primaryStage.setScene(scene);

primaryStage.show();

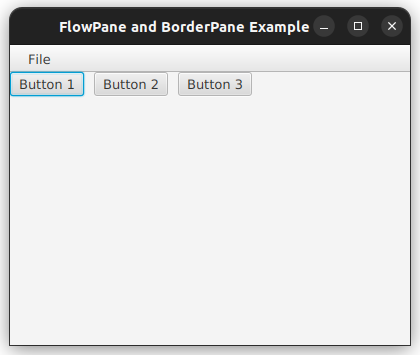
}

public static void main(String[] args) {

launch(args);

}

}



1. Create a registration form with labels and text fields arranged vertically using VBox.

This program creates a JavaFX registration form using a VBox layout to arrange labels and text fields vertically, with a simple structure for user input.

import javafx.application.Application;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

public class RegistrationFormVBox extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Registration Form");

VBox vbox = new VBox(10); // Vertical spacing of 10

vbox.setPadding(new javafx.geometry.Insets(20, 20, 20, 20));

Label nameLabel = new Label("Name:");

TextField nameField = new TextField();

Label emailLabel = new Label("Email:");

TextField emailField = new TextField();

Label passwordLabel = new Label("Password:");

PasswordField passwordField = new PasswordField();

Button registerButton = new Button("Register");

vbox.getChildren().addAll(nameLabel, nameField, emailLabel, emailField, passwordLabel, passwordField, registerButton);

Scene scene = new Scene(vbox, 300, 250);

primaryStage.setScene(scene);

primaryStage.show();

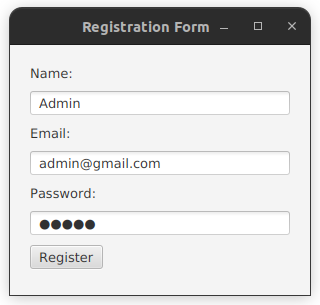
}

public static void main(String[] args) {

launch(args);

}

}



1. Design a calculator interface using a GridPane layout

This program demonstrates a simple calculator interface using JavaFX. It arranges the calculator buttons in a GridPane layout with basic arithmetic operations and a display field.

import javafx.application.Application;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.TextField;

import javafx.scene.layout.GridPane;

import javafx.stage.Stage;

public class CalculatorGridPane extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Calculator");

// Create the TextField to display calculations

TextField display = new TextField();

display.setEditable(false);

display.setAlignment(Pos.CENTER\_RIGHT);

// Create a GridPane layout

GridPane grid = new GridPane();

grid.setVgap(10); // Vertical gap

grid.setHgap(10); // Horizontal gap

grid.setAlignment(Pos.CENTER);

// Define the buttons

String[] buttonLabels = {

"7", "8", "9", "/",

"4", "5", "6", "\*",

"1", "2", "3", "-",

"C", "0", "=", "+"

};

// Add buttons to the grid

int row = 1, col = 0;

for (String label : buttonLabels) {

Button button = new Button(label);

button.setPrefSize(60, 60);

grid.add(button, col, row);

col++;

if (col > 3) {

col = 0;

row++;

}

}

// Add the display TextField at the top

grid.add(display, 0, 0, 4, 1);

Scene scene = new Scene(grid, 300, 400);

primaryStage.setScene(scene);

primaryStage.show();

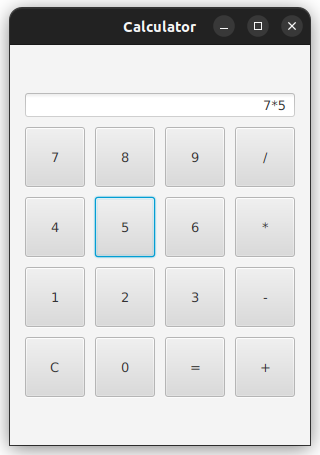
}

public static void main(String[] args) {

launch(args);

}

}



1. Create a form with Label, TextField, Button, and CheckBox. Display the entered data on a button click.

This program creates a simple JavaFX form with a TextField to enter the name, a CheckBox to subscribe to a newsletter, and a Button to submit the form. When the button is clicked, the entered data is displayed in a Label.

import javafx.application.Application;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.CheckBox;

import javafx.scene.control.Label;

import javafx.scene.control.TextField;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

public class FormWithCheckbox extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Form with Checkbox");

// Create the Label, TextField, CheckBox, and Button

Label instructionLabel = new Label("Enter your name:");

TextField nameField = new TextField();

CheckBox checkBox = new CheckBox("Subscribe to newsletter");

Button submitButton = new Button("Submit");

// Label to display the entered data

Label resultLabel = new Label();

// Button click action

submitButton.setOnAction(e -> {

String name = nameField.getText();

boolean isSubscribed = checkBox.isSelected();

resultLabel.setText("Name: " + name + ", Subscribed: " + isSubscribed);

});

// Layout with VBox

VBox vbox = new VBox(10); // Vertical spacing of 10

vbox.getChildren().addAll(instructionLabel, nameField, checkBox, submitButton, resultLabel);

// Set the scene and show the stage

Scene scene = new Scene(vbox, 300, 250);

primaryStage.setScene(scene);

primaryStage.show();

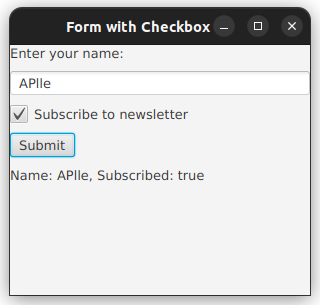
}

public static void main(String[] args) {

launch(args);

}

}



1. Implement a program that allows the user to choose a file and display its absolute path in a TextField.

This program uses JavaFX to allow the user to select a file using a FileChooser. The absolute path of the selected file is then displayed in a TextField. This provides a basic demonstration of how to interact with the file system in JavaFX.

import javafx.application.Application;

import javafx.scene.Scene;

import javafx.scene.control.\*;

import javafx.scene.layout.VBox;

import javafx.stage.FileChooser;

import javafx.stage.Stage;

import java.io.File;

public class FileChooserExample extends Application {

@Override  
public void start(Stage primaryStage) {  
 primaryStage.setTitle("Choose a File");  
  
 // Create the TextField and Button  
 TextField filePathField = new TextField();  
 filePathField.setEditable(false); // Make the text field non-editable  
 Button chooseFileButton = new Button("Choose File");  
  
 chooseFileButton.setOnAction(e -> {  
 // Create a FileChooser  
 FileChooser fileChooser = new FileChooser();  
 fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("All Files", "\*.\*"));  
  
 File selectedFile = fileChooser.showOpenDialog(primaryStage);  
 if (selectedFile != null) {  
 // Set the absolute path in the TextField  
 filePathField.setText(selectedFile.getAbsolutePath());  
 }  
 });  
  
 // Layout with VBox  
 VBox vbox = new VBox(10); // Vertical spacing of 10  
 vbox.getChildren().addAll(chooseFileButton, filePathField);  
  
  
 Scene scene = new Scene(vbox, 400, 150);  
 primaryStage.setScene(scene);  
 primaryStage.show();  
}  
  
public static void main(String[] args) {  
 launch(args);  
}  
}

