# JAVA PROGRAMMING

# DIGITAL ASSIGNMENT – 01

Aditya Firoda-16BCE2184
Prashant Mehlawat -16BCE0910

## DEVELOP A JAVA FX APPLICATION

**Registration Form Application with Database Connectivity**

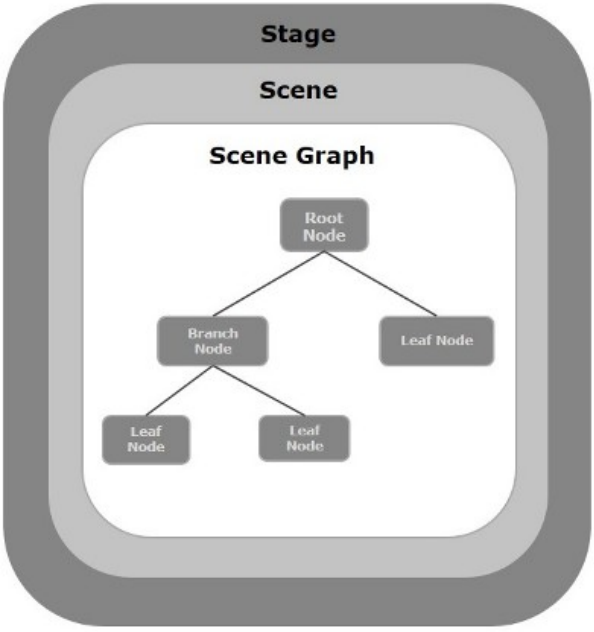
**JavaFX -**
JavaFX is a Java library used to develop Desktop applications as well as Rich Internet Applications (RIA). The applications built in JavaFX, can run on multiple platforms including Web, Mobile and Desktops.

JavaFX is intended to replace swing in Java applications as a GUI framework. However, It provides more functionalities than swing. Like Swing, JavaFX also provides its own components and doesn't depend upon the operating system. It is lightweight and hardware accelerated. It supports various operating systems including Windows, Linux and Mac OS.

**Explanation of the JavaFX Used --**

- **GridPane -** GridPane Layout pane allows us to add the multiple nodes in multiple rows and columns. It is seen as a flexible grid of rows and columns where nodes can be placed in any cell of the grid. It is represented by javafx.scence.layout.GridPane class. We just need to instantiate this class to implement GridPane.

- **Application -** In general, a JavaFX application will have three major components namely Stage, Scene and Nodes as shown in the following diagram.




- **Scene -** The JavaFX *Scene* object is the root of the JavaFX Scene graph. In other words, the JavaFX Scene contains all the visual JavaFX GUI components inside it. A JavaFX Scene is represented by the class javafx.scene.Scene. A Scene object has to be set on a JavaFX Stage to be visible. In this JavaFX Scene tutorial I will show you how to create a Scene object and add GUI components to it.

- **TextField -** TextField class is a part of JavaFX package. It is a component that allows the user to enter a line of unformatted text, it does not allow multi-line input it only allows the user to enter a single line of text. The text can then be used as per requirement.

- **Label -** Label is a part of JavaFX package . Label is used to display a short text or an image, it is a non-editable text control. It is useful for displaying text that is required to fit within a specific space, and thus may need to use an ellipsis or truncation to size the string to fit.

- **Button -** Button class is a part of JavaFX package and it can have a text or graphic or both. Button in JavaFX can be of three different types:
  Normal Button: A normal push button
  Default Button: A default button that receives a keyboard VK_ENTER press
  Cancel Button: A cancel button that receives a keyboard VK_ENTER press
  When the button is pressed an Action Event is sent. This Action Event can be managed by an EventHandler. Buttons can also respond to mouse events by implementing an EventHandler to process the MouseEvent.

- **SetOnAction -** The button's action, which is invoked whenever the button is fired. This may be due to the user clicking on the button with the mouse, or by a touch event, or by a key press, or if the developer programmatically invokes the fire() method.

- **SetTitle -** used to provide title to the stage.
- **ColumnConstraints -** Creates a column constraint object with a fixed size range, horizontal grow priority, horizonal alignment, and horizontal fill behavior.
- **Alert -** Alert is a part of JavaFX and it is a subclass of Dialog class. Alerts are some predefined dialogs that are used to show some information to the user.
- **GetScene -** Gets the value of the property scene.
- **GetWindow -** Gets the value of the property window.

**Explanation of Database Connection --**

- **Class.forName -** The java.lang.Class.forName(String name, boolean initialize, ClassLoader loader) method returns the Class object associated with the class or interface with the given string name, using the given class loader.

  The specified class loader is used to load the class or interface. If the parameter loader is null, the class is loaded through the bootstrap class loader. The class is initialized only if the initialize parameter is true and if it has not been initialized earlier.

- **PrepareStatement -** The PreparedStatement interface is a subinterface of Statement. It is used to execute parameterized query.
- **java.Sql.* -** Provides the API for accessing and processing data stored in a data source (usually a relational database) using the Java programming language.

**Java Swing -**
Java Swing is a lightweight Java graphical user interface (GUI) widget toolkit that includes a rich set of widgets. It is part of the Java Foundation Classes (JFC) and includes several packages for developing rich desktop applications in Java. Swing includes built-in controls such as trees, image buttons, tabbed panes, sliders, toolbars, color choosers, tables, and text areas to display HTTP or rich text format (RTF). Swing components are written entirely in Java and thus are platform-independent.

**CODE(With Commented Explanation) -**

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.geometry.HPos;
import javafx.geometry.Insets;
```

```
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.scene.text.Font;
import javafx.scene.text.FontWeight;
import javafx.stage.Stage;
import javafx.stage.Window;
import java.io.*;
import java.sql.*;


public class RegistrationFormApplication extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("Registration Form JavaFX Application");

        // Create the registration form grid pane
        GridPane gridPane = createRegistrationFormPane();
        // Add UI controls to the registration form grid pane
        addUIControls(gridPane);
        // Create a scene with registration form grid pane as the root node
        Scene scene = new Scene(gridPane, 800, 500);
        // Set the scene in primary stage
        primaryStage.setScene(scene);

        primaryStage.show();
    }


    private GridPane createRegistrationFormPane() {
        // Instantiate a new Grid Pane
        GridPane gridPane = new GridPane();

        // Position the pane at the center of the screen, both vertically and horizontally
        gridPane.setAlignment(Pos.CENTER);

        // Set a padding of 20px on each side
        gridPane.setPadding(new Insets(40, 40, 40, 40));

        // Set the horizontal gap between columns
        gridPane.setHgap(10);

        // Set the vertical gap between rows
        gridPane.setVgap(10);

        // Add Column Constraints

        // columnOneConstraints will be applied to all the nodes placed in column one.
        ColumnConstraints columnOneConstraints = new ColumnConstraints(100, 100,
Double.MAX_VALUE);
```

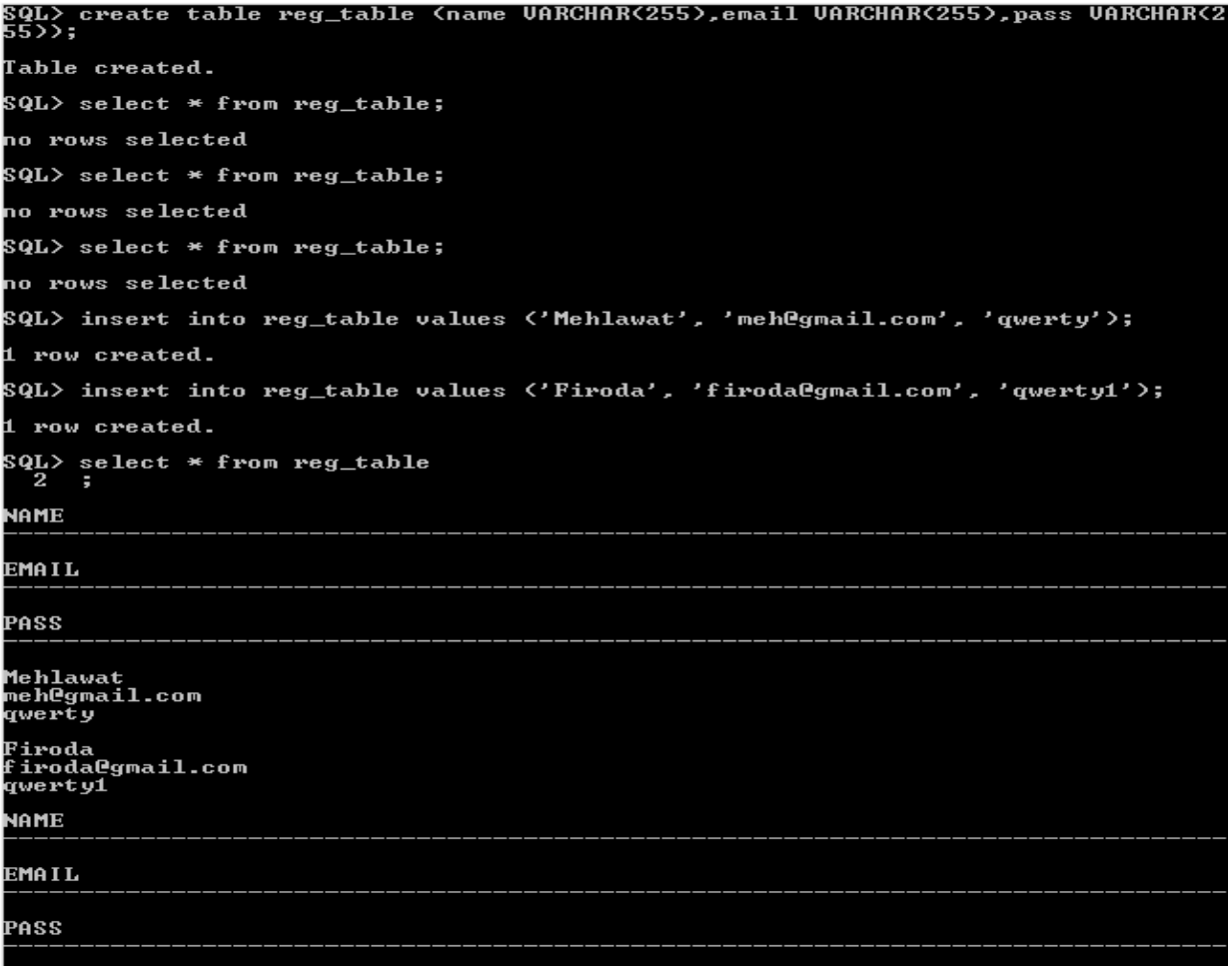
```
        columnOneConstraints.setHalignment(HPos.RIGHT);

        // columnTwoConstraints will be applied to all the nodes placed in column two.
        ColumnConstraints columnTwoConstrains = new ColumnConstraints(200,200,
Double.MAX_VALUE);
        columnTwoConstrains.setHgrow(Priority.ALWAYS);

        gridPane.getColumnConstraints().addAll(columnOneConstraints, columnTwoConstrains);

        return gridPane;
    }

    private void addUIControls(GridPane gridPane) {
        // Add Header
        Label headerLabel = new Label("Registration Form");
        headerLabel.setFont(Font.font("Arial", FontWeight.BOLD, 24));
        gridPane.add(headerLabel, 0,0,2,1);
        GridPane.setHalignment(headerLabel, HPos.CENTER);
        GridPane.setMargin(headerLabel, new Insets(20, 0,20,0));

        // Add Name Label
        Label nameLabel = new Label("Full Name : ");
        gridPane.add(nameLabel, 0,1);

        // Add Name Text Field
        TextField nameField = new TextField();
        nameField.setPrefHeight(40);
        gridPane.add(nameField, 1,1);


        // Add Email Label
        Label emailLabel = new Label("Email ID : ");
        gridPane.add(emailLabel, 0, 2);

        // Add Email Text Field
        TextField emailField = new TextField();
        emailField.setPrefHeight(40);
        gridPane.add(emailField, 1, 2);

        // Add Password Label
        Label passwordLabel = new Label("Password : ");
        gridPane.add(passwordLabel, 0, 3);

        // Add Password Field
        PasswordField passwordField = new PasswordField();
        passwordField.setPrefHeight(40);
        gridPane.add(passwordField, 1, 3);

        // Add Submit Button
        Button submitButton = new Button("Submit");
        submitButton.setPrefHeight(40);
        submitButton.setDefaultButton(true);
```

```
        submitButton.setPrefWidth(100);
        gridPane.add(submitButton, 0, 4, 2, 1);
        GridPane.setHalignment(submitButton, HPos.CENTER);
        GridPane.setMargin(submitButton, new Insets(20, 0,20,0));

        submitButton.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                if(nameField.getText().isEmpty()) {
                    showAlert(Alert.AlertType.ERROR, gridPane.getScene().getWindow(), "Form Error!",
"Please enter your name");
                    return;
                }
                if(emailField.getText().isEmpty()) {
                    showAlert(Alert.AlertType.ERROR, gridPane.getScene().getWindow(), "Form Error!",
"Please enter your email id");
                    return;
                }
                if(passwordField.getText().isEmpty()) {
                    showAlert(Alert.AlertType.ERROR, gridPane.getScene().getWindow(), "Form Error!",
"Please enter a password");
                    return;
                }
                String s=nameField.getText();
                        String em=emailField.getText();
                                String pass=passwordField.getText();
                showAlert(Alert.AlertType.CONFIRMATION, gridPane.getScene().getWindow(),
"Registration Successful!", "Welcome " + nameField.getText());
try{
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection
con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","root");
PreparedStatement st=con.prepareStatement("insert into reg_table values('?','?','?')");
st.setString(1,s);
st.setString(2,em);
st.setString(3,pass);

}
catch(Exception e){
}        }
        });
    }

    private void showAlert(Alert.AlertType alertType, Window owner, String title, String message) {
        Alert alert = new Alert(alertType);
        alert.setTitle(title);
        alert.setHeaderText(null);
        alert.setContentText(message);
        alert.initOwner(owner);
        alert.show();
    }
```

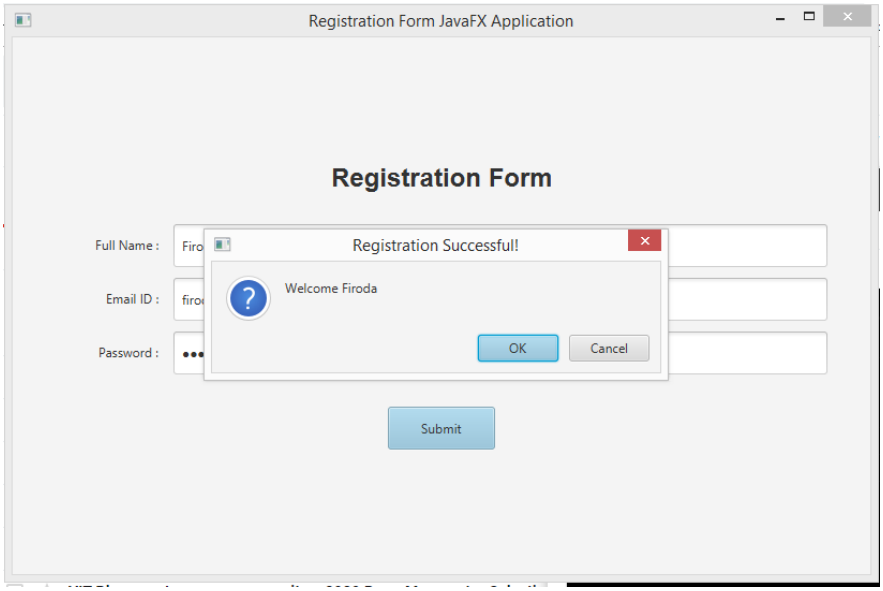
```
    public static void main(String[] args) {
        launch(args);
    }
}
```

**Screenshot OUTPUT -**

1. Database Creation and Insertion of Values



2. Output Photos




3. Output Photos