

1. Abstract:

This assignment involves the creation of a Java Swing application that captures and displays personal information (Name, Email, Phone Number, Date of Birth, and Gender) using a form-based interface. The application consists of a form with input fields for each of the aforementioned data points and a "Submit" button that displays the captured information in a new window upon clicking. The purpose of this project is to demonstrate fundamental Java Swing concepts and the ability to handle user input and output in a graphical user interface (GUI).

2. Introduction:

The goal of this assignment is to develop a basic Java Swing application that takes user input through a graphical form and displays the submitted data. This involves using Swing components such as `JTextField`, `JPasswordField`, `JComboBox`, `JButton`, `JLabel`, and `JPanel` to build the user interface. The main objective is to showcase your understanding of building interactive UIs and handling events in a Swing-based Java application.

The form will capture personal information like Name, Email, Phone Number, Date of Birth, and Gender. Upon submission, the application should collect the entered information and display it in a new frame. This assignment highlights essential programming techniques including event handling, form validation, and creating reusable components.

Steps:

Here are the steps involved in building the Java Swing form:

1. Setup the Project:

- Create a new Java project in your preferred IDE (e.g., IntelliJ IDEA, Eclipse).
- Ensure that the Java Development Kit (JDK) is installed and correctly configured.

2. Design the Swing Form:

- Use `JFrame` to create the main window.
- Use `JLabel` components to label each form field (Name, Email, Phone Number, Date of Birth, Gender).
- Use `JTextField` for text input (for Name, Email, and Phone Number).
- Use `JComboBox` for selecting Gender (Male, Female, Other).
- Use `JSpinner` or `JFormattedTextField` for Date of Birth to ensure proper date format.
- Add a `JButton` for the user to submit the form.

3. Layout:

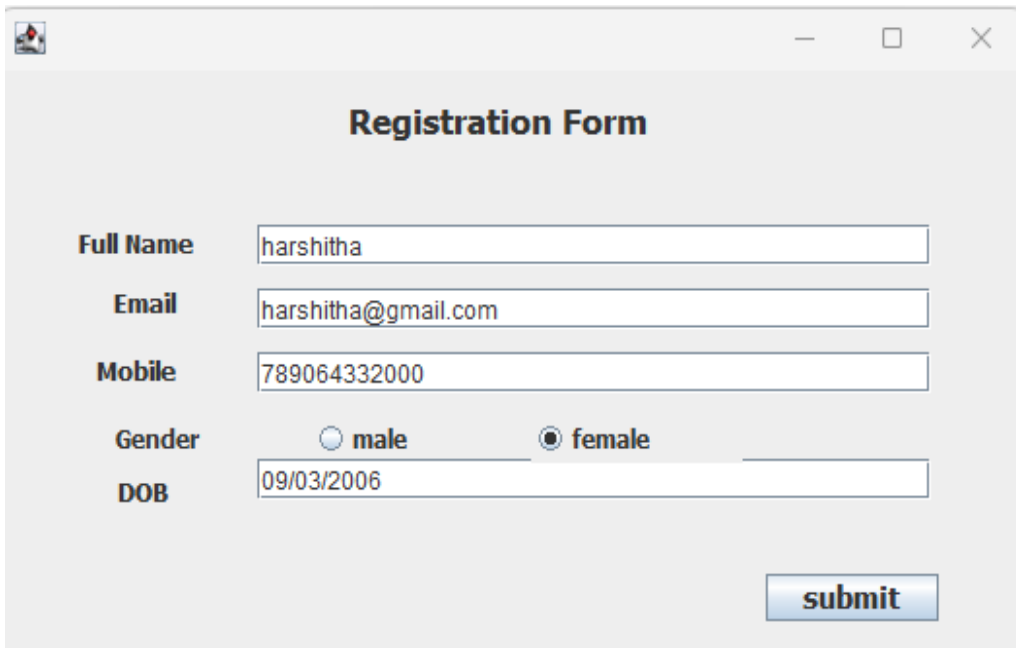
- Use `FlowLayout` or `GridLayout` for organizing the form components.
- Ensure the layout is user-friendly and aligns the labels and text fields properly.

4. Handling User Input:

- Attach action listeners to the "Submit" button to handle the form submission.

- Validate the user input (e.g., check if the email is valid, phone number is in the correct format).
 - Ensure that all fields are filled before submission.
 - 5. **Display Submitted Information:**
 - When the submit button is clicked, create a new frame (`JFrame`), and display the entered data.
 - Use `JLabel` in the new frame to display the user's input.
 - 6. **Code Implementation:**
 - Implement the event-handling mechanism for the "Submit" button.
 - Add logic to display a new frame showing the entered details.
 - 7. **Testing:**
 - Run the program and test the form by entering different sets of data.
 - Ensure that all inputs are captured and displayed correctly.
 - 8. **Final Touches:**
 - Make any adjustments to the GUI components for better aesthetics (optional).
 - Ensure that the program works correctly on different screen sizes.
-

4. Screenshots:



The screenshot shows a Java Swing window titled "Registration Form". It contains the following fields and controls:

- Full Name:** A text field containing "harshitha".
- Email:** A text field containing "harshitha@gmail.com".
- Mobile:** A text field containing "789064332000".
- Gender:** Two radio buttons labeled "male" and "female". The "female" radio button is selected.
- DOB:** A text field containing "09/03/2006".
- submit:** A button located at the bottom right of the form.

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

```
public class PersonalInfoForm {

    // Main frame for the form
    private JFrame mainFrame;
    private JTextField nameField, emailField, phoneField;
    private JSpinner dobSpinner;
    private JComboBox<String> genderComboBox;

    public PersonalInfoForm() {
        // Set up the main frame
        mainFrame = new JFrame("Personal Information Form");
        mainFrame.setSize(400, 300);
        mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        mainFrame.setLayout(new GridLayout(6, 2));

        // Initialize components
        JLabel nameLabel = new JLabel("Name:");
        nameField = new JTextField();

        JLabel emailLabel = new JLabel("Email:");
        emailField = new JTextField();

        JLabel phoneLabel = new JLabel("Phone Number:");
        phoneField = new JTextField();

        JLabel dobLabel = new JLabel("Date of Birth:");
        dobSpinner = new JSpinner(new SpinnerDateModel());

        JLabel genderLabel = new JLabel("Gender:");
        genderComboBox = new JComboBox<>(new String[] { "Male", "Female", "Other" });
    }
}
```

```
JButton submitButton = new JButton("Submit");

// Add components to frame
mainFrame.add(nameLabel);
mainFrame.add(nameField);
mainFrame.add(emailLabel);
mainFrame.add(emailField);
mainFrame.add(phoneLabel);
mainFrame.add(phoneField);
mainFrame.add(dobLabel);
mainFrame.add(dobSpinner);
mainFrame.add(genderLabel);
mainFrame.add(genderComboBox);
mainFrame.add(new JLabel());
mainFrame.add(submitButton);

// Action listener for Submit button
submitButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String name = nameField.getText();
        String email = emailField.getText();
        String phone = phoneField.getText();
        String dob = dobSpinner.getValue().toString();
        String gender = genderComboBox.getSelectedItem().toString();

        // Create a new frame to display the entered information
        JFrame resultFrame = new JFrame("Submitted Information");
        resultFrame.setSize(300, 200);
        resultFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        JPanel resultPanel = new JPanel(new GridLayout(5, 1));
```

```

        resultPanel.add(new JLabel("Name: " + name));
        resultPanel.add(new JLabel("Email: " + email));
        resultPanel.add(new JLabel("Phone: " + phone));
        resultPanel.add(new JLabel("DOB: " + dob));
        resultPanel.add(new JLabel("Gender: " + gender));

        resultFrame.add(resultPanel);
        resultFrame.setVisible(true);
    }
});

// Display the main form
mainFrame.setVisible(true);
}

public static void main(String[] args) {
    new PersonalInfoForm();
}
}

```

5. Conclusion:

In conclusion, this assignment provided an opportunity to practice working with Java Swing to build a user interface that collects, processes, and displays data. By completing this project, I gained valuable experience in GUI development, event handling, and form validation. I learned how to create responsive and user-friendly interfaces and how to organize the layout of components for optimal usability. This experience will serve as a foundation for building more complex Java applications in the future.