Java Developer Internship

Task 6: Java GUI – ToDo App

Submitted by:

Soumen Das

Date:

November 21, 2025

Submission for Elevate Labs

1 Objective

The objective of this task was to build a functional To-Do list desktop application using the Java Swing framework. The application is designed to introduce the core concepts of GUI (Graphical User Interface) development in Java, including event handling, layout management, and component interaction.

2 Requirements & Tools

- **Language:** Java
- **Library:** Swing (javax.swing)
- **Components Used:** JFrame, JButton, JTextField, JList, DefaultListModel, JScrollPane.

3 Implementation Details

The application was built using the standard `javax.swing` library.

3.1 Key Features

1. **Task Entry:** Users can type a task into a text field.
2. **Adding Tasks:** Clicking the ”Add Task” button or pressing Enter adds the item to the list model. Input validation ensures empty tasks are not added.
3. **Viewing Tasks:** A scrollable list displays all current tasks.
4. **Deleting Tasks:** Users can select a task from the list and click ”Delete Selected Task” to remove it.

4 Source Code

Below is the complete source code for the `ToDoApp.java` file.

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

public class ToDoApp extends JFrame {

    private DefaultListModel<String> listModel;
    private JList<String> taskList;
    private JTextField taskInput;
    private JButton addButton;
    private JButton deleteButton;

    public ToDoApp() {
        // Set up the main frame
        setTitle("To-Do List - Soumen Das");
        setSize(400, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout(10, 10));

        // --- Top Panel ---
        JPanel inputPanel = new JPanel();
        inputPanel.setLayout(new BorderLayout(5, 5));
        inputPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 0, 10));

        taskInput = new JTextField();
        addButton = new JButton("Add Task");

        inputPanel.add(taskInput, BorderLayout.CENTER);
        inputPanel.add(addButton, BorderLayout.EAST);

        // --- Center List ---
        listModel = new DefaultListModel<>();
        taskList = new JList<>(listModel);
        taskList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

        JScrollPane scrollPane = new JScrollPane(taskList);
        scrollPane.setBorder(BorderFactory.createCompoundBorder(
                BorderFactory.createEmptyBorder(10, 10, 10, 10),
                BorderFactory.createLineBorder(Color.GRAY)
        ));

        // --- Bottom Panel ---
        JPanel bottomPanel = new JPanel();
        bottomPanel.setBorder(BorderFactory.createEmptyBorder(0, 10, 10, 10));
        deleteButton = new JButton("Delete Selected Task");
        bottomPanel.add(deleteButton);

        add(inputPanel, BorderLayout.NORTH);
        add(scrollPane, BorderLayout.CENTER);
        add(bottomPanel, BorderLayout.SOUTH);

        // --- Logic ---
        ActionListener addAction = e -> addTask();
        addButton.addActionListener(addAction);
        taskInput.addActionListener(addAction);

        deleteButton.addActionListener(e -> deleteTask());
    }

    private void addTask() {
        String task = taskInput.getText().trim();
        if (!task.isEmpty()) {
            listModel.addElement(task);
            taskInput.setText("");
            taskInput.requestFocus();
        } else {
            JOptionPane.showMessageDialog(this, "Please enter a task.");
        }
    }

    private void deleteTask() {
        int selectedIndex = taskList.getSelectedIndex();
        if (selectedIndex != -1) {
            listModel.remove(selectedIndex);
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            try {
                UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
            } catch (Exception ignored) {}
            new ToDoApp().setVisible(true);
        });
    }
}
```

Listing 1: ToDoApp.java

5 Conclusion

This project successfully demonstrates the use of Java Swing to create a functional desktop application. The application handles user input and dynamic data modification (adding/removing items from a list model) effectively.