Assignment: Java Swing GUI Development (C7-JAV-11)

Topic: Foundational Concepts of Java Swing

Type: Group Presentation, Live Demo, and Lab/Challenge Exercise
Group Size: 3-5 Students (Groups 3, 4, and 5 as Presenters)

1 Overview

Dive into Java Swing for building graphical user interfaces (GUIs) in an immersive, interactive 2-hour session per group, totaling 3 sessions across multiple classes! Groups 3, 4, and 5 will deliver elaborate presentations, live demos, and a small lab/challenge exercise to engage the class, while the audience prepares questions to spark lively discussions. Use creative visuals, analogies, or gamified elements to make it memorable. Each group has a full 2-hour lecture to present, demonstrate, and conduct their challenge.

2 Learning Objectives

- Understand Java Swing architecture and its relation to AWT.
- Explore core Swing components (e.g., JFrame, JPanel, JButton, JLabel, JTextField).
- Master layout managers (FlowLayout, BorderLayout, GridLayout, BoxLayout, GridBagLayout).
- Implement event handling (ActionListener, MouseListener, KeyListener, FocusListener).
- Work with graphics and painting in Swing (e.g., drawing shapes, images, custom painting).
- Create menus, toolbars, and dialogs (JMenuBar, JMenu, JToolBar, JOptionPane, JDialog, JFileChooser).
- Handle advanced topics like threading in Swing (SwingWorker, Event Dispatch Thread).
- Integrate Swing with data-driven components (e.g., JTable, JList, JTree with models).
- Build complete Swing applications demonstrating foundational and intermediate concepts.
- Develop skills in presentation, demo creation, and designing interactive challenges.

3 Roles and Sub-Topic Division

3.1 Presenters

- **Group 3: Swing Architecture, Components, and Layouts**

  - *Sub-Topics*: Java Swing Architecture (vs. AWT, MVC pattern, pluggable look and feel), Core Components (JFrame, JPanel, JButton, JLabel, JTextField, JCheckBox, JRadioButton, JComboBox, etc.), Layout Managers (FlowLayout, BorderLayout, GridLayout, BoxLayout, GridBagLayout, custom layouts).
  - *Tasks*: Create detailed slides (20-30 slides) with explanations, diagrams (e.g., component hierarchy, layout visuals), and real-world examples. Deliver a live demo in class, building a complex GUI with various components and layouts, allowing real-time audience interaction (e.g., adjusting layouts live). Design a 20-30 minute lab/challenge exercise for classmates (e.g., modify a provided GUI to add a new component or change layouts). Include recommended setup steps for Java Swing development (e.g., IDE configuration, importing javax.swing) in the presentation.

- **Group 4: Event Handling and User Interactions**
  - *Sub-Topics*: Event Handling Mechanisms (Event Dispatch Thread, ActionListener, MouseListener, KeyListener, FocusListener, WindowListener), Handling User Inputs (button clicks, mouse events, keyboard shortcuts), Best Practices for Responsive GUIs (avoiding EDT blocking, using adapters).
  - *Tasks*: Create in-depth slides (20-30 slides) with code snippets, flowcharts (e.g., event propagation), and case studies. Deliver a live demo in class, showcasing an interactive GUI (e.g., a mini-game or form with multiple event listeners), responding to audience inputs. Design a 20-30 minute lab/challenge exercise (e.g., add an event listener to a provided GUI to handle a new user action). Include tips on debugging events in the presentation.

- **Group 5: Advanced Features and Application Building**
  - *Sub-Topics*: Graphics and Painting (Graphics2D, drawing shapes/images/text, custom components with paintComponent), Menus, Toolbars, and Dialogs (JMenuBar, JMenu, JToolBar, JOptionPane, JDialog, JFileChooser, JColorChooser), Threading in Swing (SwingWorker for background tasks), Data-Driven Components (JTable, JList, JTree with models), Building Complete Applications (integrating all concepts, error handling, MVC implementation).
  - *Tasks*: Create comprehensive slides (20-30 slides) with advanced examples, case studies, and optimization tips. Deliver a live demo in class, showcasing a full-featured Swing application (e.g., a simple editor, dashboard, or data viewer) with audience-driven modifications. Design a 20-30 minute lab/challenge exercise (e.g., enhance a provided app with a menu, dialog, or custom graphics). Include best practices for threading and data integration in the presentation.

- **Deliverables (All Groups)**:
  - Meet with the lecturer one day before the presentation to discuss content, slides, demo, and lab/challenge exercise.
  - Submit the lab/challenge exercise description and materials (e.g., starter code, instructions) to the lecturer before class via Email.

  - Submit slides (PDF/PPT) and demo code (zipped) after the presentation via Email.
  - Present in class (2 hours per group: 60 min slides, 30 min live demo, 30 min lab/challenge exercise, including Q&A).

3.2 Audience

- Pre-read materials suggested by presenters (shared via lecturer) and prepare 3-5 thoughtful questions related to the sub-topics.
- Engage in class: Ask questions, participate in live demos, complete the lab/challenge exercise.

4 Evaluation (100 Points)

- Content Clarity and Depth (30%): Accurate and thorough coverage of assigned sub-topics.
- Creativity and Engagement (25%): Engaging slides, demo, and lab/challenge (e.g., fun visuals, interactive elements, or gamified tasks).
- Demo Functionality and Complexity (20%): Working, interactive live demo showcasing sub-topics with elaborate examples.
- Lab/Challenge Quality (15%): Relevant, engaging exercise tied to sub-topics, submitted on time.
- Presentation/Teamwork (10%): Clear delivery, collaboration, and time management within 2 hours.
- **Extra Credit (5 Points)**: Most creative element (e.g., best challenge design, innovative demo, or unique analogy).

5 Tips

- Schedule the lecturer meeting early to refine your elaborate content.
- Use tools like Google Slides or GitHub for collaboration and version control.
- Test demos and lab exercises extensively to handle complex interactions smoothly.
- Make it inclusive and funsurprise the class with creativity and depth!