GUI

```
import javax.swing.*;

public class JTableExample extends JFrame {
    // Declare components as instance variables
    private JTable table;
    private JScrollPane scrollPane;

    public JTableExample() {
        // Set up the JFrame
        setTitle("JTable Example");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Data for the JTable
        String[][] data = {
            {"1", "Bristi Maharjan", "Kathmandu"},
            {"2", "John Doe", "Lalitpur"},
        };

        // Column names
        String[] columnNames = {"ID", "Name", "Address"};

        // Initialize the table
        table = new JTable(data, columnNames);

        // Add the table to a scroll pane
        scrollPane = new JScrollPane(table);

        // Add the scroll pane to the frame
        add(scrollPane);
    }
```

```
    public static void main(String[] args) {
        // Create and display the frame
        JTableExample frame = new JTableExample();
        frame.setVisible(true);
    }
}
```

```
import javax.swing.*;
import javax.swing.tree.DefaultMutableTreeNode;

public class JTreeExample extends JFrame {
    // Declare components as instance variables
    private JTree tree;
    private JScrollPane scrollPane;

    public JTreeExample() {
        // Set up the JFrame
        setTitle("JTree Example");
        setSize(300, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Initialize and build the tree
        DefaultMutableTreeNode root = new DefaultMutableTreeNode("Person Deta
        DefaultMutableTreeNode nameNode = new DefaultMutableTreeNode("Name
        nameNode.add(new DefaultMutableTreeNode("Bristi Maharjan"));
        DefaultMutableTreeNode addressNode = new DefaultMutableTreeNode("Ad
        addressNode.add(new DefaultMutableTreeNode("Kathmandu"));
        addressNode.add(new DefaultMutableTreeNode("Lalitpur"));
        root.add(nameNode);
        root.add(addressNode);

        // Initialize tree and scroll pane
        tree = new JTree(root);
        scrollPane = new JScrollPane(tree);
```

```
        // Add the scroll pane to the frame
        add(scrollPane);
    }

    public static void main(String[] args) {
        // Create and display the frame
        JTreeExample frame = new JTreeExample();
        frame.setVisible(true);
    }
}
```

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import javax.swing.tree.DefaultMutableTreeNode;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class JTableJTreeExample {

    public static void main(String[] args) {
        // Create the frame
        JFrame frame = new JFrame("JTable and JTree Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 400);
        frame.setLayout(new BorderLayout());

        // Create a button
        JButton displayButton = new JButton("Display Data");

        // Create a panel to hold the button
        JPanel panel = new JPanel();
        panel.add(displayButton);

        // Add panel to the frame
```

```
        frame.add(panel, BorderLayout.NORTH);

        // Create a table model and table
        DefaultTableModel tableModel = new DefaultTableModel();
        JTable table = new JTable(tableModel);

        // Create a tree
        DefaultMutableTreeNode root = new DefaultMutableTreeNode("Root");
        JTree tree = new JTree(root);

        // Add a split pane to display both JTable and JTree side by side
        JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT, new
        frame.add(splitPane, BorderLayout.CENTER);

        // Action listener for the button
        displayButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Populate JTable with sample data
                String[] columnNames = {"ID", "Name", "Age"};
                Object[][] data = {
                        {1, "John", 22},
                        {2, "Anna", 20},
                        {3, "Mike", 25}
                };
                tableModel.setDataVector(data, columnNames);

                // Populate JTree with sample data
                root.removeAllChildren(); // Clear previous data
                DefaultMutableTreeNode node1 = new DefaultMutableTreeNode("Perso
                node1.add(new DefaultMutableTreeNode("Age: 22"));
                DefaultMutableTreeNode node2 = new DefaultMutableTreeNode("Perso
                node2.add(new DefaultMutableTreeNode("Age: 20"));
                DefaultMutableTreeNode node3 = new DefaultMutableTreeNode("Perso
                node3.add(new DefaultMutableTreeNode("Age: 25"));
                root.add(node1);
```

```
            root.add(node2);
            root.add(node3);
            ((DefaultTreeModel) tree.getModel()).reload();  // Refresh the tree
        }
    });

    // Show the frame
    frame.setVisible(true);
  }
}
```

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

public class NumberOperationGUI {
  public static void main(String[] args) {
    // Create a frame
    JFrame frame = new JFrame("Number Operations");
    frame.setSize(400, 200);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLayout(null);

    // Create text fields for input
    JTextField textField1 = new JTextField();
    JTextField textField2 = new JTextField();
    textField1.setBounds(50, 30, 100, 30);
    textField2.setBounds(200, 30, 100, 30);

    // Create a button
    JButton button = new JButton("Calculate");
    button.setBounds(150, 80, 100, 30);

    // Create a label to display results
    JLabel resultLabel = new JLabel("Result: ");
    resultLabel.setBounds(50, 120, 300, 30);
```

```
// Add MouseListener to handle both mouse press and release
button.addMouseListener(new MouseAdapter() {
    @Override
    public void mousePressed(MouseEvent e) {
        try {
            // Get numbers from text fields
            double num1 = Double.parseDouble(textField1.getText());
            double num2 = Double.parseDouble(textField2.getText());

            // Display the sum when the button is pressed
            double sum = num1 + num2;
            resultLabel.setText("Result: Sum = " + sum);
        } catch (NumberFormatException ex) {
            resultLabel.setText("Error: Please enter valid numbers");
        }
    }

    @Override
    public void mouseReleased(MouseEvent e) {
        try {
            // Get numbers again from text fields
            double num1 = Double.parseDouble(textField1.getText());
            double num2 = Double.parseDouble(textField2.getText());

            // Display the difference when the button is released
            double difference = num1 - num2;
            resultLabel.setText("Result: Difference = " + difference);
        } catch (NumberFormatException ex) {
            resultLabel.setText("Error: Please enter valid numbers");
        }
    }
});

// Add components to frame
frame.add(textField1);
```

```
        frame.add(textField2);
        frame.add(button);
        frame.add(resultLabel);

        // Set frame visibility
        frame.setVisible(true);
    }
}
```

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

public class EventListenersExample extends JFrame implements WindowListene

    public EventListenersExample() {
        // Set up the JFrame
        setTitle("Event Listeners Example");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        addWindowListener(this);  // Add WindowListener
        addMouseListener(this);  // Add MouseListener
        addKeyListener(this);    // Add KeyListener
        setFocusable(true);

        // Add a label to display messages
        JLabel label = new JLabel("Interact with the window, mouse, or keyboard!")
        label.setHorizontalAlignment(JLabel.CENTER);
        add(label);
    }

    // WindowListener methods
    public void windowOpened(WindowEvent e) {
        System.out.println("Window opened");
    }
```

```
    public void windowClosing(WindowEvent e) {
        System.out.println("Window closing");
    }

    public void windowClosed(WindowEvent e) {
        System.out.println("Window closed");
    }

    public void windowIconified(WindowEvent e) {
        System.out.println("Window minimized");
    }

    public void windowDeiconified(WindowEvent e) {
        System.out.println("Window restored");
    }

    public void windowActivated(WindowEvent e) {
        System.out.println("Window activated");
    }

    public void windowDeactivated(WindowEvent e) {
        System.out.println("Window deactivated");
    }

    // MouseListener methods
    public void mouseClicked(MouseEvent e) {
        System.out.println("Mouse clicked at (" + e.getX() + ", " + e.getY() + ")");
    }

    public void mousePressed(MouseEvent e) {
        System.out.println("Mouse pressed");
    }

    public void mouseReleased(MouseEvent e) {
        System.out.println("Mouse released");
```

```
    }

    public void mouseEntered(MouseEvent e) {
        System.out.println("Mouse entered the window");
    }

    public void mouseExited(MouseEvent e) {
        System.out.println("Mouse exited the window");
    }

    // KeyListener methods
    public void keyTyped(KeyEvent e) {
        System.out.println("Key typed: " + e.getKeyChar());
    }

    public void keyPressed(KeyEvent e) {
        System.out.println("Key pressed: " + e.getKeyChar());
    }

    public void keyReleased(KeyEvent e) {
        System.out.println("Key released: " + e.getKeyChar());
    }

    public static void main(String[] args) {
        // Create and display the JFrame
        EventListenersExample example = new EventListenersExample();
        example.setVisible(true);
    }
}
```

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

public class ColorSelectorGUI extends JFrame {
```

```
    // Declare components
    private JComboBox<String> colorDropdown;
    private JButton selectButton;
    private JLabel resultLabel;

    // Constructor to set up the GUI
    public ColorSelectorGUI() {
        // Set frame properties
        setTitle("Color Selector");
        setSize(400, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Set GridLayout (3 rows, 1 column, with 10px vertical gap)
        setLayout(new GridLayout(3, 1, 0, 10));

        // Initialize components
        String[] colors = { "Red", "Green", "Blue" };
        colorDropdown = new JComboBox<>(colors);
        selectButton = new JButton("Select Color");
        resultLabel = new JLabel("Selected Color: ", JLabel.CENTER);

        // Add ActionListener to button
        selectButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String selectedColor = (String) colorDropdown.getSelectedItem();

                // Set the label text and color based on the selected item
                resultLabel.setText("Selected Color: " + selectedColor);

                switch (selectedColor) {
                    case "Red":
                        resultLabel.setForeground(Color.RED);
                        break;
                    case "Green":
                        resultLabel.setForeground(Color.GREEN);
```

```
                    break;
                case "Blue":
                    resultLabel.setForeground(Color.BLUE);
                    break;
            }
        }
    });

    // Add components to the frame in GridLayout order
    add(colorDropdown);
    add(selectButton);
    add(resultLabel);

    // Make the frame visible
    setVisible(true);
  }

  // Main method to run the program
  public static void main(String[] args) {
    new ColorSelectorGUI();
  }
}
```

Here's the comparison in a concise tabular form:

Aspect	**Swing**	**AWT**
Component Set	Richer set with advanced components	Basic components like buttons, text fields
Look and Feel	Customizable, pluggable look and feel	Tied to the native OS look and feel
Platform Dependency	Platform-independent (Java-based)	Platform-dependent (uses native OS components)
Lightweight/Heavyweight	Lightweight (JVM drawn)	Heavyweight (uses native OS components)

Performance	Slower due to Java implementation	Faster for simple applications
Event Handling	Advanced, flexible event handling	Simpler, less flexible
Advanced Features	Supports advanced UI features like tooltips, icons	Lacks advanced features

Swing and **AWT** are both Java libraries used for building graphical user interfaces (GUIs), but they differ in their design, functionality, and usage.

Swing:

- **Definition**: Swing is a part of Java's Standard Library for building GUIs, introduced as a more advanced replacement for AWT. It is built entirely in Java, making it platform-independent.
- **Key Features**:
  - Provides a rich set of GUI components (e.g., tables, trees, sliders).
  - Lightweight components that are drawn by Java itself, not the native OS.
  - Customizable look and feel (pluggable look and feel), allowing developers to choose different themes for the UI.
  - Supports advanced features such as tooltips, icons, and complex event handling.

AWT (Abstract Window Toolkit):

- **Definition**: AWT is an older Java library for creating GUIs, introduced with Java 1.0. It uses native OS components to build the interface, meaning its appearance depends on the operating system.
- **Key Features**:
  - Provides basic GUI components like buttons, labels, and text fields.
  - Heavyweight components, meaning the components are rendered by the native OS (leading to platform dependency).
  - Limited customization options compared to Swing.

- Simpler and faster for basic applications but less flexible and feature-rich than Swing.

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

public class MultiplyNumbers extends JFrame implements ActionListener {

    // Declare components
    private JTextField number1Field, number2Field, resultField;
    private JButton okButton, exitButton;

    public MultiplyNumbers() {
        // Set up the frame
        setTitle("Multiply Two Numbers");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(400, 200);
        setLayout(new GridLayout(4, 2, 10, 10)); // 4 rows, 2 columns, 10px gap

        // Initialize components
        JLabel number1Label = new JLabel("Enter first number:");
        number1Field = new JTextField();

        JLabel number2Label = new JLabel("Enter second number:");
        number2Field = new JTextField();

        JLabel resultLabel = new JLabel("Result:");
        resultField = new JTextField();
        resultField.setEditable(false); // Result field is not editable

        okButton = new JButton("OK");
        exitButton = new JButton("Exit");
```

```
        // Add action listeners
        okButton.addActionListener(this);
        exitButton.addActionListener(this);

        // Add components to the frame
        add(number1Label);
        add(number1Field);
        add(number2Label);
        add(number2Field);
        add(resultLabel);
        add(resultField);
        add(okButton);
        add(exitButton);

        // Make the frame visible
        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == okButton) {
            try {
                // Parse input numbers
                double number1 = Double.parseDouble(number1Field.getText());
                double number2 = Double.parseDouble(number2Field.getText());

                // Perform multiplication
                double result = number1 * number2;

                // Display result
                resultField.setText(String.valueOf(result));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(this, "Please enter valid numbers!", "
            }
        } else if (e.getSource() == exitButton) {
            // Terminate the program
```

```
            System.exit(0);
        }
    }

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

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

public class ColorChangeButtons {
    public static void main(String[] args) {
        // Create a frame
        JFrame frame = new JFrame("Color Change Buttons");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 200);

        // Create buttons
        JButton redButton = new JButton("RED");
        JButton blueButton = new JButton("BLUE");
        JButton clearButton = new JButton("CLEAR");

        // Set the layout
        frame.setLayout(new FlowLayout());

        // Add buttons to the frame
        frame.add(redButton);
        frame.add(blueButton);
        frame.add(clearButton);

        // Action listener for RED button
        redButton.addActionListener(new ActionListener() {
```

```
            public void actionPerformed(ActionEvent e) {
                redButton.setBackground(Color.RED);
            }
        });

        // Action listener for BLUE button
        blueButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                blueButton.setBackground(Color.BLUE);
            }
        });

        // Action listener for CLEAR button
        clearButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                redButton.setBackground(null);
                blueButton.setBackground(null);
            }
        });

        // Display the frame
        frame.setVisible(true);
    }
}
```

```
package GUI;

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

public class Calculator extends JFrame{
    Calculator(){
        setLayout(new GridLayout(6,2,10,10));
```

```
        JLabel l1 = new JLabel("Calculator");
        add(l1);

        JLabel l2 = new JLabel();
        add(l2);

        JLabel l3 = new JLabel("First Number:");
        add(l3);

        JTextField t1 = new JTextField();
        add(t1);

        JLabel l4 = new JLabel("Second Number:");
        add(l4);

        JTextField t2 = new JTextField();
        add(t2);

        JLabel l5 = new JLabel("Result");
        add(l5);

        JTextField t3 = new JTextField();
        add(t3);

        JButton b1 = new JButton("Add");
        add(b1);
        b1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    int num1 = Integer.parseInt(t1.getText());
                    int num2 = Integer.parseInt(t2.getText());
                    int sum = num1 + num2;
                    t3.setText(String.valueOf(sum));
                }catch (NumberFormatException exception){
                    System.out.println("Error");
```

```
            }
        }
    });

    JButton b2 = new JButton("Sub");
    add(b2);
    b2.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            try {
                int num1 = Integer.parseInt(t1.getText());
                int num2 = Integer.parseInt(t2.getText());
                int sub = num1 - num2;
                t3.setText(String.valueOf(sub));
            }catch (NumberFormatException exception){
                System.out.println("error");
            }
        }
    });

    setSize(400,400);
    setVisible(true);
  }

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

}
```

1. **JFrame**: A top-level container used to create a window where other components are added.
2. **JButton**: A button that can trigger an action when clicked.
3. **JLabel**: A non-editable text or image label, typically used for displaying information.
4. **JTextField**: A one-line text box used for user input.
5. **JTextArea**: A multi-line text area for displaying or inputting longer texts.
6. **JCheckBox**: A box that can either be checked or unchecked, used for binary choices.
7. **JRadioButton**: A button in a group of radio buttons where only one can be selected at a time.
8. **JComboBox**: A dropdown list that allows the user to select an item from a predefined list.
9. **JList**: A component that displays a list of items from which the user can select.
10. **JSpinner**: A component for selecting numeric or object values, allowing for incremental changes.
11. **JProgressBar**: A bar that shows the progress of a task visually.
12. **JMenuBar, JMenu, JMenuItem**: Used for ↓ ting menus and menu items within the application.

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

public class SwingComponentsExample extends JFrame {

```
public SwingComponentsExample() {
    setTitle("Swing Components Example");
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLayout(new FlowLayout());

    // JButton example
    JButton button = new JButton("Click Me");
    button.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(null, "Button Clicked!");
        }
    });
```

```
    // JLabel example
    JLabel label = new JLabel("This is a JLabel");

    // JTextField example
    JTextField textField = new JTextField(15);

    // JTextArea example
    JTextArea textArea = new JTextArea(5, 15);
    textArea.setText("This is a JTextArea\\nYou can type here.");

    // JCheckBox example
    JCheckBox checkBox = new JCheckBox("Accept Terms and Conditions");

    // JRadioButton example
    JRadioButton radioButton1 = new JRadioButton("Option 1");
    JRadioButton radioButton2 = new JRadioButton("Option 2");
    ButtonGroup radioGroup = new ButtonGroup();
    radioGroup.add(radioButton1);
    radioGroup.add(radioButton2);

    // JComboBox example
    JComboBox<String> comboBox = new JComboBox<>(new String[] {"Item
1", "Item 2", "Item 3"});

    // JList example
    JList<String> list = new JList<>(new String[] {"List Item 1", "List Item 2", "Li
st Item 3"});

    // JSpinner example
    JSpinner spinner = new JSpinner(new SpinnerNumberModel(0, 0, 10, 1));

    // JProgressBar example
    JProgressBar progressBar = new JProgressBar();
    progressBar.setValue(50);
    progressBar.setStringPainted(true);
```

```
        // JMenuBar, JMenu, JMenuItem example
        JMenuBar menuBar = new JMenuBar();
        JMenu menu = new JMenu("File");
        JMenuItem menuItem = new JMenuItem("Open");
        menu.add(menuItem);
        menuBar.add(menu);
        setJMenuBar(menuBar);

        // Add components to the JFrame
        add(button);
        add(label);
        add(textField);
        add(textArea);
        add(checkBox);
        add(radioButton1);
        add(radioButton2);
        add(comboBox);
        add(new JScrollPane(list));
        add(spinner);
        add(progressBar);

        setSize(400, 600);
        setVisible(true);
    }

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

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

```
public class RadioButtonAndTableExample extends JFrame implements ActionLis

    JRadioButton rb1, rb2, rb3;
    JButton submitButton;
    JTable table;

    public RadioButtonAndTableExample() {
        // Set JFrame title and layout
        setTitle("JRadioButton and JTable Example");
        setLayout(new BorderLayout());

        // Create radio buttons and group them
        rb1 = new JRadioButton("Option 1");
        rb2 = new JRadioButton("Option 2");
        rb3 = new JRadioButton("Option 3");

        ButtonGroup group = new ButtonGroup();
        group.add(rb1);
        group.add(rb2);
        group.add(rb3);

        // Add radio buttons to a panel
        JPanel radioPanel = new JPanel();
        radioPanel.add(rb1);
        radioPanel.add(rb2);
        radioPanel.add(rb3);

        // Create submit button
        submitButton = new JButton("Submit");
        submitButton.addActionListener(this);
        radioPanel.add(submitButton);

        // Add radio panel to the top of the JFrame
        add(radioPanel, BorderLayout.NORTH);

        // Create data for the JTable
```

```
        String[][] data = {
            {"1", "John", "22"},
            {"2", "Anna", "24"},
            {"3", "Mike", "20"}
        };

        // Column names for the JTable
        String[] columnNames = {"ID", "Name", "Age"};

        // Create the JTable
        table = new JTable(data, columnNames);
        JScrollPane tablePane = new JScrollPane(table);  // Add table to a scroll par

        // Add the JTable to the center of the JFrame
        add(tablePane, BorderLayout.CENTER);

        // Set JFrame properties
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle submit button action
        if (rb1.isSelected()) {
            JOptionPane.showMessageDialog(this, "You selected: Option 1");
        } else if (rb2.isSelected()) {
            JOptionPane.showMessageDialog(this, "You selected: Option 2");
        } else if (rb3.isSelected()) {
            JOptionPane.showMessageDialog(this, "You selected: Option 3");
        } else {
            JOptionPane.showMessageDialog(this, "Please select an option!");
        }
    }
```

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