# **Experiment 9**

**1. Implementing ActionListener in Java Swing to Change JTextArea Colors**

# **Program:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Exp9\_1 {

    public Exp9\_1() {

        // Create the frame

        JFrame jframe = new JFrame("Action Listener");

        jframe.setLayout(null);

        jframe.setSize(400, 400);

        jframe.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create and add the text area

        JTextArea jtextarea = new JTextArea();

        jtextarea.setBounds(50, 50, 300, 200); // Set appropriate bounds

        jframe.add(jtextarea);

        // Create and add the button

        JButton b1 = new JButton("Change Colour");

        b1.setBounds(150, 270, 120, 30); // Set appropriate bounds

        jframe.add(b1);

        // Create and add the label

        JLabel jlabel = new JLabel("");

        jlabel.setBounds(50, 320, 300, 20); // Set appropriate bounds

        jframe.add(jlabel);

        // Add action listener to the button

        b1.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                jtextarea.setBackground(Color.YELLOW);

                jtextarea.setForeground(Color.GREEN);

                jlabel.setText("Text area color changed!"); // Update the label text

            }

        });

        // Make the frame visible

        jframe.setVisible(true);

    }

    public static void main(String[] args) {

        new Exp9\_1();

    }

}

# **Output:**

**2.CardLayout Demo**

# **Program:**

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Exp9\_2 extends JFrame implements ActionListener {

    // Global variables

    private CardLayout card;

    private JButton btn1, btn2, btn3;

    private Container cPane;

    public Exp9\_2() {

        // Create container and set CardLayout

        cPane = getContentPane();

        card = new CardLayout(); // Removed spacing arguments

        cPane.setLayout(card);

        // Create buttons and add action listeners

        btn1 = new JButton("Button 1");

        btn2 = new JButton("Button 2");

        btn3 = new JButton("Button 3");

        btn1.addActionListener(this);

        btn2.addActionListener(this);

        btn3.addActionListener(this);

        // Add buttons to the container with card names

        cPane.add(btn1, "Card 1"); // Corrected the order of arguments

        cPane.add(btn2, "Card 2");

        cPane.add(btn3, "Card 3");

    }

    public void actionPerformed(ActionEvent e) {

        // Switch cards on button click

        card.next(cPane);

    }

    public static void main(String[] args) {

        Exp9\_2 frame = new Exp9\_2();

        frame.setTitle("Card Layout Demo");

        frame.setSize(400, 300);

        frame.setVisible(true);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE); // Added to close the application properly

    }

}

# **Output:**

**3. Checkbox State Listener Demo**

# **Program:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class Exp9\_3 {

    public Exp9\_3() {

        // Create the frame

        JFrame jframe = new JFrame("Action Listener");

        jframe.setLayout(null);

        jframe.setSize(400, 300); // Adjusted size for a more compact layout

        jframe.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create and set bounds for the label

        JLabel jlabel = new JLabel("Checkbox Example");

        jlabel.setBounds(150, 20, 200, 30);

        jframe.add(jlabel);

        // Create the checkboxes with labels and set their bounds

        JCheckBox jcheckbox1 = new JCheckBox("C++");

        jcheckbox1.setBounds(50, 80, 100, 30);

        jframe.add(jcheckbox1);

        JCheckBox jcheckbox2 = new JCheckBox("Java");

        jcheckbox2.setBounds(50, 120, 100, 30);

        jframe.add(jcheckbox2);

        // Create a label to display the status of checkboxes

        JLabel statusLabel = new JLabel("");

        statusLabel.setBounds(50, 160, 300, 30);

        jframe.add(statusLabel);

        // Add action listeners to the checkboxes

        jcheckbox1.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                String status = "C++ Checkbox " + (jcheckbox1.isSelected() ? "Checked" : "Unchecked");

                statusLabel.setText(status);

            }

        });

        jcheckbox2.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                String status = "Java Checkbox " + (jcheckbox2.isSelected() ? "Checked" : "Unchecked");

                statusLabel.setText(status);

            }

        });

        // Make the frame visible

        jframe.setVisible(true);

    }

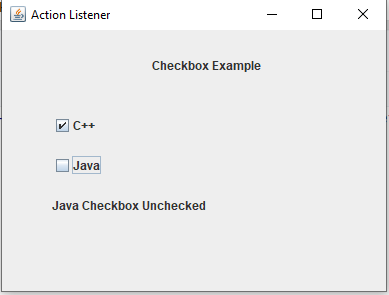
    public static void main(String[] args) {

        new Exp9\_3();

    }

}

# **Output:**



**4. Mouselistener demo**

# **Program:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Exp9\_4 {

    // Constructor to set up the GUI

    public Exp9\_4() {

        // Create the frame

        JFrame jframe = new JFrame("Mouse Listener Demo");

        jframe.setLayout(null);

        jframe.setSize(400, 300);

        jframe.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create a label to display messages

        JLabel jlabel = new JLabel("Interact with me using the mouse!");

        jlabel.setBounds(50, 100, 300, 30);

        jlabel.setHorizontalAlignment(SwingConstants.CENTER); // Center the text

        jlabel.setBorder(BorderFactory.createLineBorder(Color.BLACK)); // Add a border for visibility

        jframe.add(jlabel);

        // Create a custom mouse listener and add it to the label

        jlabel.addMouseListener(new MouseListener() {

            @Override

            public void mouseClicked(MouseEvent e) {

                jlabel.setText("Mouse Clicked at (" + e.getX() + ", " + e.getY() + ")");

            }

            @Override

            public void mousePressed(MouseEvent e) {

                jlabel.setText("Mouse Pressed");

            }

            @Override

            public void mouseReleased(MouseEvent e) {

                jlabel.setText("Mouse Released");

            }

            @Override

            public void mouseEntered(MouseEvent e) {

                jlabel.setText("Mouse Entered");

                jlabel.setBackground(Color.LIGHT\_GRAY);

                jlabel.setOpaque(true); // Ensure background color is visible

            }

            @Override

            public void mouseExited(MouseEvent e) {

                jlabel.setText("Mouse Exited");

                jlabel.setBackground(null);

                jlabel.setOpaque(false);

            }

        });

        // Make the frame visible

        jframe.setVisible(true);

    }

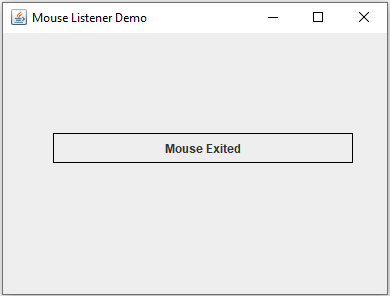
    public static void main(String[] args) {

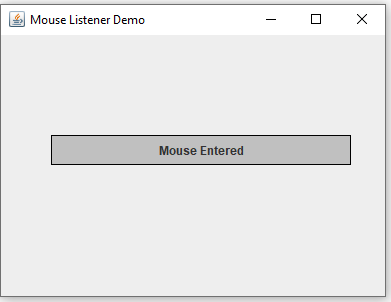
        new Exp9\_4();

    }

}

# **Output:**





**5. WindowListener Demo**

# **Program:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Exp9\_5 {

    public Exp9\_5() {

        // Create the frame

        JFrame jframe = new JFrame("Window Listener Demo");

        jframe.setLayout(null);

        jframe.setSize(400, 300);

        jframe.setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE); // Handle close operation manually

        // Create a label to display window event messages

        JLabel jlabel = new JLabel("Perform window operations and see the messages here.");

        jlabel.setBounds(20, 100, 360, 30);

        jlabel.setHorizontalAlignment(SwingConstants.CENTER);

        jlabel.setBorder(BorderFactory.createLineBorder(Color.BLACK));

        jframe.add(jlabel);

        // Add a custom window listener to the frame

        jframe.addWindowListener(new WindowListener() {

            @Override

            public void windowOpened(WindowEvent e) {

                jlabel.setText("Window Opened");

            }

            @Override

            public void windowClosing(WindowEvent e) {

                int option = JOptionPane.showConfirmDialog(jframe, "Are you sure you want to close the window?", "Close Confirmation", JOptionPane.YES\_NO\_OPTION);

                if (option == JOptionPane.YES\_OPTION) {

                    jframe.dispose();

                }

            }

            @Override

            public void windowClosed(WindowEvent e) {

                // This will be called after the window is closed (dispose() method)

                System.out.println("Window Closed");

            }

            @Override

            public void windowIconified(WindowEvent e) {

                jlabel.setText("Window Minimized");

            }

            @Override

            public void windowDeiconified(WindowEvent e) {

                jlabel.setText("Window Restored");

            }

            @Override

            public void windowActivated(WindowEvent e) {

                jlabel.setText("Window Activated");

            }

            @Override

            public void windowDeactivated(WindowEvent e) {

                jlabel.setText("Window Deactivated");

            }

        });

        // Make the frame visible

        jframe.setVisible(true);

    }

    public static void main(String[] args) {

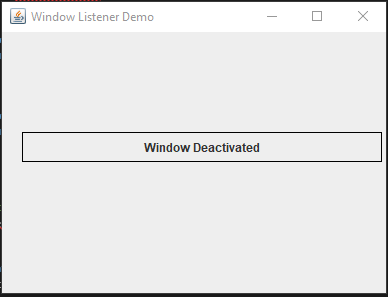
        new Exp9\_5();

    }

}

# **Output:**

# 

****