Experiment 10

Q 1. GUI Program to display the current mouse coordinates on the window.

Answer:

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.MouseMotionAdapter;

import java.awt.event.MouseEvent;

public class MouseCoordinatesGUI {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Mouse Coordinates");

        frame.setSize(400, 300);

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

        frame.setLayout(new BorderLayout())

        JLabel label = new JLabel("Move the mouse inside the window", SwingConstants.CENTER);

        label.setFont(new Font("Arial", Font.PLAIN, 16));

        frame.add(label, BorderLayout.SOUTH);

        JPanel panel = new JPanel();

        panel.setBackground(Color.WHITE);

        frame.add(panel, BorderLayout.CENTER);

      panel.addMouseMotionListener(new MouseMotionAdapter() {

            public void mouseMoved(MouseEvent e) {

                label.setText("Mouse Coordinates: X = " + e.getX() + ", Y = " + e.getY());

            }

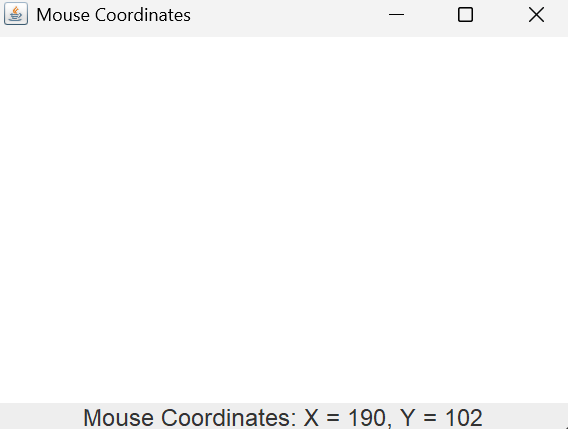
        });

        // Show the frame

        frame.setVisible(true);

    }

}



Q2. GUI Program to implement a simple Timer (using background events). Include a Start and Stop button to control the timer.

Answer:

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class SimpleTimerApp {

    private static int counter = 0;

    private static Timer timer;

    public static void main(String[] args) {

        JFrame frame = new JFrame("Simple Timer");

        frame.setSize(300, 200);

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

        frame.setLayout(new FlowLayout());

        JLabel timeLabel = new JLabel("Time: 0 seconds");

        timeLabel.setFont(new Font("Arial", Font.BOLD, 20));

        frame.add(timeLabel);

        JButton startButton = new JButton("Start");

        frame.add(startButton);

        JButton stopButton = new JButton("Stop");

        frame.add(stopButton);

        timer = new Timer(1000, new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                counter++;

                timeLabel.setText("Time: " + counter + " seconds");

            }

        });

        startButton.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                timer.start();

            }

        });

        stopButton.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                timer.stop();

            }

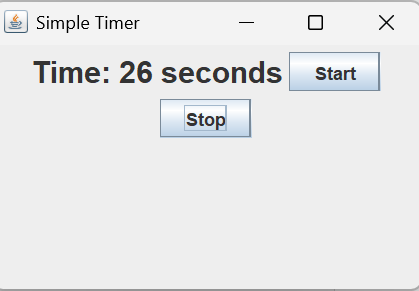
        });

        frame.setVisible(true);

    }

}

Output:



Q 3: Create a GUI with a JComboBox containing image names. On selection, display the corresponding image using a JLabel and ItemListener.

Answer:

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class ImageSelector {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Image Selector");

        frame.setSize(400, 400);

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

        frame.setLayout(new BorderLayout());

        String[] imageNames = {"apple.jpg", "banana.jpg", "grape.jpeg"};

        JComboBox<String> imageComboBox = new JComboBox<>(imageNames);

        frame.add(imageComboBox, BorderLayout.NORTH);

        JLabel imageLabel = new JLabel();

        imageLabel.setHorizontalAlignment(JLabel.CENTER);

        frame.add(imageLabel, BorderLayout.CENTER);

        imageComboBox.addItemListener(new ItemListener() {

            public void itemStateChanged(ItemEvent e) {

                if (e.getStateChange() == ItemEvent.SELECTED) {

                    String selectedImage = (String) imageComboBox.getSelectedItem();

                    ImageIcon icon = new ImageIcon(selectedImage); // Image file in the project directory

                    Image scaledImage = icon.getImage().getScaledInstance(300, 250, Image.SCALE\_SMOOTH);

                    imageLabel.setIcon(new ImageIcon(scaledImage));

                }

            }

        });

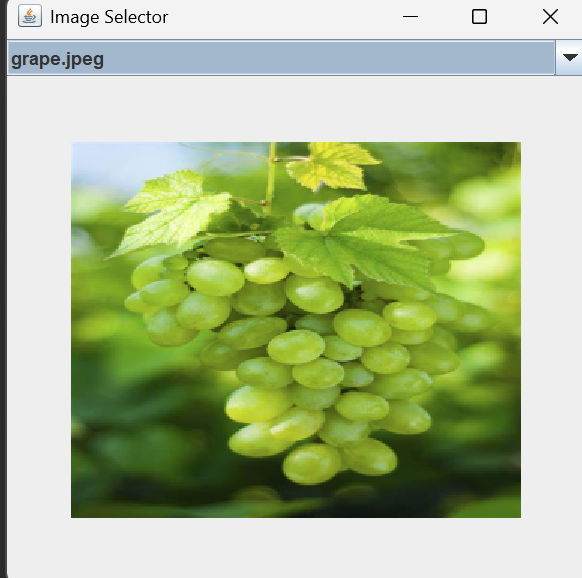
        imageComboBox.setSelectedIndex(0);

        frame.setVisible(true);

    }

}

Output:



Q 4: GUI with a JTextArea and a label. As the user types, show the character count and word count in real-time using a KeyListener.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class TextCounterGUI {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Character & Word Counter");

        frame.setSize(500, 400);

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

        frame.setLayout(new BorderLayout());

        JTextArea textArea = new JTextArea();

        textArea.setFont(new Font("Arial", Font.PLAIN, 16));

        frame.add(new JScrollPane(textArea), BorderLayout.CENTER);

        JLabel countLabel = new JLabel("Characters: 0 | Words: 0");

        countLabel.setFont(new Font("Arial", Font.BOLD, 14));

        countLabel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

        frame.add(countLabel, BorderLayout.SOUTH);

        textArea.addKeyListener(new KeyAdapter() {

            public void keyReleased(KeyEvent e) {

                String text = textArea.getText();

                int charCount = text.length();

                String[] words = text.trim().split("\\s+");

                int wordCount = (text.trim().isEmpty()) ? 0 : words.length;

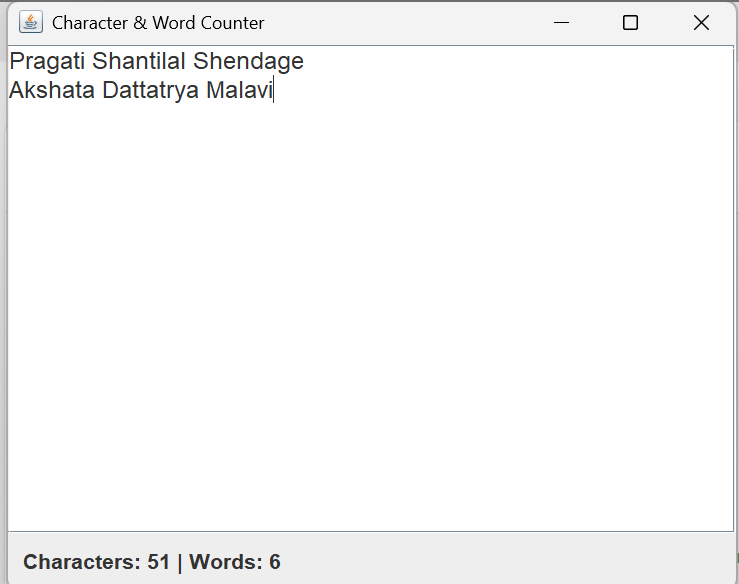
                countLabel.setText("Characters: " + charCount + " | Words: " + wordCount);

            }

        });        frame.setVisible(true);

    }

}



Q 4: Write Java GUI Program using Swing to change background on selecting color.

Answer: import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class BackgroundColorChanger {

    public static void main(String[] args){

        JFrame frame = new JFrame("Background Color Changer");

        frame.setSize(400, 300);

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

        JPanel panel = new JPanel();

        panel.setLayout(new FlowLayout());

        String[] colors = {"White", "Red", "Green", "Blue", "Yellow", "Gray"};

        JComboBox<String> colorComboBox = new JComboBox<>(colors);

        panel.add(new JLabel("Choose Background Color:"));

        panel.add(colorComboBox);

        frame.add(panel);

        colorComboBox.addItemListener(new ItemListener() {

            public void itemStateChanged(ItemEvent e) {

                if (e.getStateChange() == ItemEvent.SELECTED) {

                    String selectedColor = (String) e.getItem();

                    switch (selectedColor) {

                        case "Red":

                            panel.setBackground(Color.RED);

                            break;

                        case "Green":

                            panel.setBackground(Color.GREEN);

                            break;

                        case "Blue":

                            panel.setBackground(Color.BLUE);

                            break;

                        case "Yellow":

                            panel.setBackground(Color.YELLOW);

                            break;

                        case "Gray":

                            panel.setBackground(Color.GRAY);

                            break;

                        default:

                            panel.setBackground(Color.WHITE);

                    }

                }

            }

        });

        panel.setBackground(Color.WHITE);

        frame.setVisible(true);

    }

}

