|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Object Oriented Programming** | **Semester:** | **III** |
| **Date of Performance:** | **16 / 10 / 2023** | **Batch No:** | **Batch A - 3** |
| **Faculty Name:** | **Pragya Gupta** | **Roll No:** | **16014022050** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** | **\_\_\_ / 25** |

|  |  |  |
| --- | --- | --- |
| **Writing Program (07)** | **Performance in lab**  **and Viva (05 + 03)** | **Post lab questions, conclusion and**  **Completion (03 + 02 + 05)** |
|  |  |  |

**Experiment No: 10**

**Title: Applet and Event Handling**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| Learn how to implement GUI in Java. |

|  |
| --- |
| **COs to be achieved:** |
| **CO5**: Design and build simple Graphical User Interfaces. |

|  |
| --- |
| **Tools used:** |
| 1. [Event Handling in Java - GeeksforGeeks](https://www.geeksforgeeks.org/event-handling-in-java/) |

|  |
| --- |
| **Theory:** |
| (About Applet and list different class, Information variables of Event Class)  **Applets** in Java are small applications that are accessed and run on the internet. They are designed to be embedded within an HTML page and are commonly used to provide interactive features to web applications. Applets were widely used in the early days of the internet for various purposes like games, animations, and simple interactive applications. However, their usage has diminished in favor of more modern web technologies like HTML5, CSS, and JavaScript.  Here are some of the key points about applets in Java:   1. Applets are Java programs that are designed to be transmitted over the internet and run in the Java-enabled web browser. 2. They run inside a Java virtual machine (JVM), which is embedded within the web browser. 3. Applets are subject to certain security restrictions, such as not being able to access the local file system of the client and being restricted in terms of network connectivity. 4. Applets have specific lifecycle methods such as init(), start(), stop(), and destroy(), which are called at different stages of their execution. 5. They can be created using the java.applet package.   **Event handling** in Java is a crucial aspect of developing interactive applications. In the context of applets, various events can be generated during their lifecycle, such as mouse events, keyboard events, focus events, etc. Java provides a comprehensive set of event classes to handle these events. Some of the important event classes in Java are:   1. ActionEvent: This class is used to handle action events, typically generated when a component like a button is clicked. 2. KeyEvent: This class is used to handle keyboard events, including key presses and key releases. 3. MouseEvent: This class is used to handle mouse events, such as mouse clicks, mouse movements, and mouse drags. 4. WindowEvent: This class is used to handle window-related events, such as window activation, window deactivation, etc. 5. ComponentEvent: This class is used to handle component-related events, such as component resizing, component movement, etc.   These event classes contain various information variables that provide details about the events. Some common information variables in event classes include:  **Source:** The object that generated the event.  **ID:** An integer that represents the type of event.  **Command:** For ActionEvent, it represents the command associated with the action.  **KeyChar:** For KeyEvent, it represents the character associated with the key event.  **X, Y:** For MouseEvent, it represents the coordinates of the mouse pointer at the time of the event.  These information variables help in understanding and handling events effectively in Java applications. |

|  |
| --- |
| **Code:** |
| 1. **Write a java program to draw Oval, Rectangle, Line and fill the color in it and display it on Applet.**   import javax.swing.\*;  import java.awt.\*;  public class exp10\_q1 extends JFrame {      public exp10\_q1() {          setTitle("Shape Drawing using Swing");          setSize(400, 400);          setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);          // Create a custom JPanel for drawing          DrawingPanel drawingPanel = new DrawingPanel();          add(drawingPanel);          // Center the JFrame          setLocationRelativeTo(null);      }      public static void main(String[] args) {          SwingUtilities.invokeLater(() -> {              exp10\_q1 frame = new exp10\_q1();              frame.setVisible(true);          });      }      private class DrawingPanel extends JPanel {          @Override          protected void paintComponent(Graphics g) {              super.paintComponent(g);              setBackground(Color.white);              // Draw an oval with a red border and a filled blue color              g.setColor(Color.red);              g.drawOval(50, 50, 100, 80);              g.setColor(Color.blue);              g.fillOval(50, 50, 100, 80);              // Draw a rectangle with a green border and a filled yellow color              g.setColor(Color.green);              g.drawRect(200, 50, 100, 80);              g.setColor(Color.yellow);              g.fillRect(200, 50, 100, 80);              // Draw a line              g.setColor(Color.black);              g.drawLine(50, 200, 300, 200);          }      }  }   1. **Write a java program that works as a simple calculator. Use a GridLayout to arrange Buttons for digits and for the + - \* / operations. Add a text field to display the result. Handle any possible exceptions like divide by zero.**   import javax.swing.\*;  import javax.swing.border.EmptyBorder;  import java.awt.\*;  import java.awt.event.ActionEvent;  import java.awt.event.ActionListener;  public class exp10\_q2  {      private JFrame frame;      private JTextField textField;      private double result;      private String operator;      private boolean start;      public exp10\_q2()      {          frame = new JFrame("Simple Calculator");          frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);          frame.setSize(400, 500); // frame size          frame.setLayout(new BorderLayout());          textField = new JTextField(20); // text field size          textField.setFont(new Font("Arial", Font.PLAIN, 24)); // font size          textField.setBorder(new EmptyBorder(20, 0, 0, 0));          frame.add(textField, BorderLayout.NORTH);          JPanel panel = new JPanel();          panel.setLayout(new GridLayout(5, 4, 10, 10)); // grid layout          String[] buttons = {"7", "8", "9", "/",                  "4", "5", "6", "\*",                  "1", "2", "3", "-",                  "0", ".", "=", "+"};          for (String button : buttons) {              JButton btn = new JButton(button);              btn.setFont(new Font("Arial", Font.PLAIN, 24));              panel.add(btn);              btn.addActionListener(new CalculatorListener());          }          frame.add(panel, BorderLayout.CENTER);          frame.setVisible(true);          result = 0;          operator = "=";          start = true;      }      private class CalculatorListener implements ActionListener      {          public void actionPerformed(ActionEvent e) {              String command = e.getActionCommand();              if (command.charAt(0) >= '0' && command.charAt(0) <= '9' || command.equals(".")) {                  if (start) {                      textField.setText("");                      start = false;                  }                  textField.setText(textField.getText() + command);              } else {                  if (start) {                      if (command.equals("-")) {                          textField.setText(command);                          start = false;                      } else                          operator = command;                  } else {                      try {                          double value = Double.parseDouble(textField.getText());                          calculate(value);                          operator = command;                          start = true;                      } catch (NumberFormatException ex) {                          textField.setText("Error");                          start = true;                      }                  }              }          }          private void calculate(double n)          {              switch (operator) {                  case "+":                      result += n;                      break;                  case "-":                      result -= n;                      break;                  case "\*":                      result \*= n;                      break;                  case "/":                      if (n == 0) {                          textField.setText("Error");                          start = true;                      } else {                          result /= n;                      }                      break;                  case "=":                      result = n;                      break;              }              textField.setText("" + result);          }      }      public static void main(String[] args)      {          new exp10\_q2();      }  }   1. **Develop a GUI which will have following field.**    1. **Name : TextField**    2. **Department : Choice/Dropdown**    3. **Mobile Number :**    4. **Gender : Check Box**    5. **About Yourself : Text Area**    6. **SUBMIT button in Center**   **After click on SUBMIT button, it should display “Welcome <Name>, to the < Department>.**  import java.awt.\*;  import java.awt.event.ActionEvent;  import java.awt.event.ActionListener;  import javax.swing.\*;  public class exp10\_q3  {      private JFrame frame;      private JTextField nameField, mobileField;      private Choice departmentChoice;      private JCheckBox maleCheckbox, femaleCheckbox;      private JTextArea aboutArea;      public exp10\_q3()      {          frame = new JFrame("User Form");          frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);          frame.setSize(400, 400);          frame.setLayout(new GridLayout(5, 1));          JPanel panel1 = new JPanel(new FlowLayout()); // First panel for name, department, and mobile number          panel1.add(new JLabel("Name:"));          nameField = new JTextField(20);          panel1.add(nameField);          panel1.add(new JLabel("Department:"));          departmentChoice = new Choice();          departmentChoice.add("EXCP");          departmentChoice.add("COMPS");          departmentChoice.add("IT");          departmentChoice.add("MECH");          panel1.add(departmentChoice);          panel1.add(new JLabel("Mobile Number:"));          mobileField = new JTextField(20);          panel1.add(mobileField);          frame.add(panel1);          JPanel panel2 = new JPanel(new FlowLayout()); // Second panel for gender checkboxes          panel2.add(new JLabel("Gender:"));          maleCheckbox = new JCheckBox("Male");          femaleCheckbox = new JCheckBox("Female");          panel2.add(maleCheckbox);          panel2.add(femaleCheckbox);          frame.add(panel2);          JPanel panel3 = new JPanel(new FlowLayout()); // Third panel for about yourself          panel3.add(new JLabel("About Yourself:"));          aboutArea = new JTextArea(5, 20);          aboutArea.setLineWrap(true);          JScrollPane scrollPane = new JScrollPane(aboutArea);          panel3.add(scrollPane);          frame.add(panel3);          JPanel panel4 = new JPanel(new FlowLayout()); // Fourth panel for the submit button          JButton submitButton = new JButton("Submit");          submitButton.addActionListener(new SubmitListener());          panel4.add(submitButton);          frame.add(panel4);          frame.setVisible(true);      }      private class SubmitListener implements ActionListener      {          public void actionPerformed(ActionEvent e) {              String name = nameField.getText();              String department = departmentChoice.getSelectedItem();              String gender = "";              if(maleCheckbox.isSelected()){                  gender = "Male";              }              if(femaleCheckbox.isSelected()){                  gender = "Female";              }              String message = "Welcome " + name + ", to the " + department + ". You are " + gender + ".";              JOptionPane.showMessageDialog(frame, message);          }      }      public static void main(String[] args)      {          new exp10\_q3();      }  } |

|  |
| --- |
| **Output:** |
| Addition - 9 + 9:    Subtraction - 100 – 76:    Division - 81 / 9:    Multiplication - 3 \* 9: |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| 1. **Write a java program that handles all mouse events and shows the event name at the center of the window when mouse event is fired (Use Adapter classes).**   import java.awt.\*;  import java.awt.event.MouseAdapter;  import java.awt.event.MouseEvent;  import javax.swing.\*;  public class MouseEventDemo extends JFrame {      private JLabel label;      public MouseEventDemo() {          setTitle("Mouse Events Demo");          setSize(300, 200);          setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);          label = new JLabel();          label.setHorizontalAlignment(SwingConstants.CENTER);          add(label, BorderLayout.CENTER);          addMouseListener(new MouseAdapter() {              @Override              public void mouseClicked(MouseEvent e) {                  label.setText("Mouse Clicked");              }              @Override              public void mousePressed(MouseEvent e) {                  label.setText("Mouse Pressed");              }              @Override              public void mouseReleased(MouseEvent e) {                  label.setText("Mouse Released");              }              @Override              public void mouseEntered(MouseEvent e) {                  label.setText("Mouse Entered");              }              @Override              public void mouseExited(MouseEvent e) {                  label.setText("Mouse Exited");              }          });          setVisible(true);      }      public static void main(String[] args) {          new MouseEventDemo();      }  }     1. **What is the difference between an Applet and a Java Application?**  * An applet is a Java program that runs in a web browser with the help of a Java Virtual Machine (JVM), whereas a Java application is a standalone program that runs on a computer. * Applets require the Applet class and use the Applet lifecycle methods, while Java applications use the main method as the entry point. * Applets have security restrictions imposed by the browser, while Java applications have fewer restrictions.  1. **What are the restrictions imposed on Java applets?**  * Applets cannot access the local file system of the client. * They cannot communicate with other servers except the one from which they are loaded. * Applets are restricted from certain system properties and environment variables. * They have limited access to the user's system resources for security reasons.  1. **Why Swing is called light weight?**  * Swing components are not dependent on native peers and are rendered using pure Java code. * They do not rely on the underlying native operating system's widgets, making them platform-independent. * Swing components are flexible and highly customizable, allowing developers to create user interfaces with various styles and themes. * They have less overhead compared to AWT (Abstract Window Toolkit) components, making them more efficient and faster in terms of rendering and performance. |

|  |
| --- |
| **Conclusion:** |
| Through the Java experiment involving the implementation of GUI, I gained practical insights into creating graphical user interfaces using various Java libraries like Swing and AWT. By utilizing components such as buttons, text fields, labels, and event handling mechanisms, I developed an understanding of building interactive and user-friendly applications. |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |