LAB12

19-10-2024

Q1)

Aim:

Write a program that will display check boxes and option buttons they are numbered from1 to. Use a textbox to display the number those corresponding boxes or button checked.

Algorithm:

Initialize the Frame and Components:

- Create a JFrame and set up the layout to FlowLayout.
- Initialize three checkboxes (CheckBox 1, CheckBox 2, CheckBox 3).
- Initialize three radio buttons (Option 1, Option 2, Option 3).
- Group the radio buttons so only one can be selected at a time.
- Initialize a non-editable text field to display the selected options.

Add Components to Frame:

- Add the checkboxes, radio buttons, and the text field to the JFrame.

ActionListener Setup:

- Create an ActionListener that reacts whenever a checkbox or radio button is selected or deselected.
- Inside the ActionListener, call the updateSelectedItems() method to update the text field.

UpdateSelectedItems Method:

- Create a method updateSelectedItems() to:
    - Clear the text field.
    - Check which checkboxes and radio buttons are selected.

- Append the corresponding numbers (1, 2, 3) for the selected checkboxes and radio buttons to a string.
- Update the text field to display the selected numbers.

Display the Frame:

- Set the frame visibility to true to display the GUI.

Main Method:

- Use SwingUtilities.invokeLater() to start the GUI in a thread-safe manner and instantiate the Q1 class.

Code:

```
/*

Write a program that will display check boxes and option buttons they are numbered from1 to. Use a textbox to display the number those corresponding boxes or button checked.

 */

package LAB12;


import javax.swing.*;

import java.awt.*;

import java.awt.event.*;


public class Q1 {


   private JFrame frame;

   private JCheckBox checkBox1, checkBox2, checkBox3;

   private JRadioButton radioButton1, radioButton2, radioButton3;

   private JTextField textField;
```

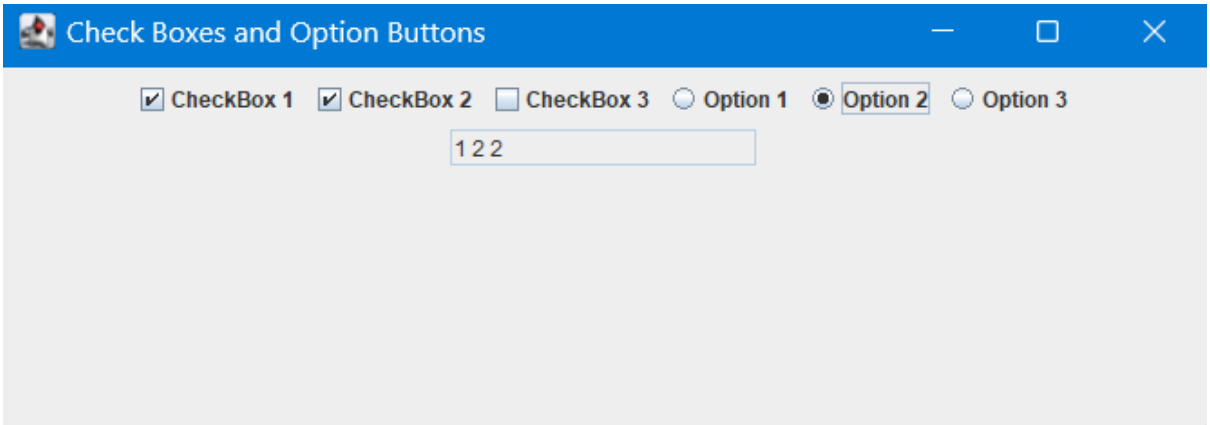
```
private ButtonGroup radioGroup;

public Q1() {
   frame = new JFrame("Check Boxes and Option Buttons");
   frame.setLayout(new FlowLayout());
   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   frame.setSize(300, 200);

   checkBox1 = new JCheckBox("CheckBox 1");
   checkBox2 = new JCheckBox("CheckBox 2");
   checkBox3 = new JCheckBox("CheckBox 3");

   radioButton1 = new JRadioButton("Option 1");
   radioButton2 = new JRadioButton("Option 2");
   radioButton3 = new JRadioButton("Option 3");

   radioGroup = new ButtonGroup();
   radioGroup.add(radioButton1);
   radioGroup.add(radioButton2);
   radioGroup.add(radioButton3);

   textField = new JTextField(15);
   textField.setEditable(false);

   frame.add(checkBox1);
   frame.add(checkBox2);
```

```
    frame.add(checkBox3);
    frame.add(radioButton1);
    frame.add(radioButton2);
    frame.add(radioButton3);
    frame.add(textField);

    ActionListener updateTextField = new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            updateSelectedItems();
        }
    };

    checkBox1.addActionListener(updateTextField);
    checkBox2.addActionListener(updateTextField);
    checkBox3.addActionListener(updateTextField);
    radioButton1.addActionListener(updateTextField);
    radioButton2.addActionListener(updateTextField);
    radioButton3.addActionListener(updateTextField);

    frame.setVisible(true);
}

private void updateSelectedItems() {
    StringBuilder selectedItems = new StringBuilder();
```

```
    if (checkBox1.isSelected()) {
        selectedItems.append("1 ");
    }
    if (checkBox2.isSelected()) {
        selectedItems.append("2 ");
    }
    if (checkBox3.isSelected()) {
        selectedItems.append("3 ");
    }

    if (radioButton1.isSelected()) {
        selectedItems.append("1 ");
    }
    if (radioButton2.isSelected()) {
        selectedItems.append("2 ");
    }
    if (radioButton3.isSelected()) {
        selectedItems.append("3 ");
    }

    textField.setText(selectedItems.toString());
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> new Q1());
}
```

}

Output:



Q2)

Aim:

Write a program to create a Applet life cycle.

Algorithm:

Applet Initialization (init()):

- When the applet is loaded for the first time, the init() method is called.
- Print "Applet is initialized" to the console.

Applet Start (start()):

- After initialization, or when the applet comes into view, the start() method is called.
- Print "Applet is started" to the console.

Applet Stop (stop()):

- When the applet goes out of view or is no longer active, the stop() method is called.
- Print "Applet is stopped" to the console.

Applet Destruction (destroy()):

- Just before the applet is unloaded or destroyed, the destroy() method is called.
- Print "Applet is destroyed" to the console.

Painting the Applet (paint()):

- The paint(Graphics g) method is responsible for rendering any graphical output.
- Draw the string "Welcome to Applet Life Cycle Demo" at coordinates (20, 50) in the applet window.

Code:

Q2COMMENTS.java

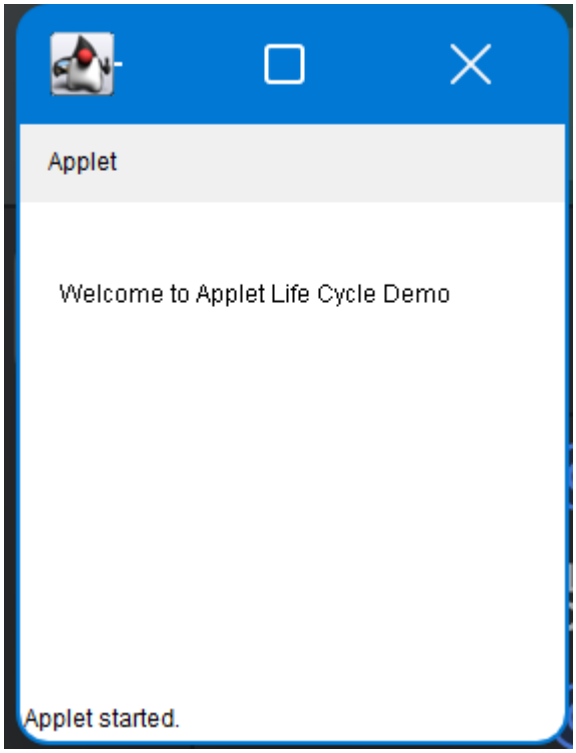
```
/*
Write a program to create a Applet life cycle.
 */
package LAB12;

import java.applet.Applet;
import java.awt.Graphics;

public class Q2COMMENTS extends Applet {
   public void init() {
      System.out.println("Applet is initialized.");
   }

   public void start() {
      System.out.println("Applet is started.");
   }

   public void stop() {
      System.out.println("Applet is stopped.");
   }
```

```
    public void destroy() {
        System.out.println("Applet is destroyed.");
    }

    public void paint(Graphics g) {
        g.drawString("Welcome to Applet Life Cycle Demo", 20, 50);
    }
}
```

applet.html

```
<html>
  <body>
    <applet code="LAB12.Q2COMMENTS" width="300" height="200">
    </applet>
  </body>
</html>
```

Output:




```
Applet is initialized.
Applet is started.
Applet is stopped.
Applet is destroyed.
```