PS02CMCA51 Object Oriented Programming using Java

Dr. J. V. Smart

Table of Contents

Syllabus

Syllabus with effect from the Academic Year 2021-2022

Course Code	PS02CMCA51	Title of the Course	OBJECT ORIENTED PROGRAMMING USING JAVA
Total Credits of the Course	4	Hours per Week	4

Course	1. To learn computer programming using the Java

Objectives:	programming language and the Java Platform, Standard Edition (Java SE). 2. To learn the fundamentals of object-oriented programming. 3. Learning to write object-oriented programs in Java. 4. Knowledge of important features of the Java SE platform. 5. Learning to develop graphical and database programs using Java.

Course Content

Unit	Description	Weightage* (%)
1.	**Introduction to Java** - The Java programming language: history, evolution, features - Introduction to the Java programming environment, JDK, JRE - Introduction to the IDE - Data types and wrapper classes, operators - Control structures - String handling - Basic Input-output	25
2.	**Introduction to Object-oriented Programming** - Basic concepts of object-oriented programming - Classes, instances, methods - Static and non-static members - Packages - Inheritance and polymorphism, method overriding - Final and abstract classes, abstract methods - Interfaces - Generics, enumeration - Inner classes and anonymous classes - Class loaders, class path	25
3.	**More Features of the Java Platform** - Exception handling - Input-output and file handling	25

Unit	Description	Weightage* (%)
	- The collections framework and handling classes in it - Introduction to the java.util package - Multithreading - Introduction to network programming - Introduction to lambda expressions and serialization	
4.	**Developing Graphical Programs and Database Access** - An introduction to graphics in Java - Brief introduction to AWT - The Swing library - Writing graphical programs using Swing - Using various Swing components - Managing layout using Swing - Event handling using Swing - Introduction to JDBC - Different types of JDBC drivers - Programming database applications using JDBC	25

Teaching-Learning Methodology	Blended learning approach incorporating traditional classroom teaching as well as online / ICT-based teaching practices

Evaluation Pattern

Sr. No.	Details of the Evaluation	Weightage
1.	Internal Written / Practical Examination (As per CBCS R.6.8.3)	15%
2.	Internal Continuous Assessment in the form of Practical, Viva-voce, Quizzes, Seminars, Assignments, Attendance (As per CBCS R.6.8.3)	15%
3.	University Examination	70%

Course Outcomes: Having completed this course, the learner will be able to	
1.	develop computer programs using the Java programming language and the Java SE platform.
2.	gain an understanding of fundamental object-oriented programming concepts.
3.	develop object-oriented software in Java.
4.	display knowledge of multithreading, file handling and network programming in Java.
5.	develop GUI programs in Java.
6.	have knowledge of database access in Java using JDBC.

Suggested References:

Sr. No.	References
1.	Schildt H. : Java: The Complete Reference, 9th Edition, McGraw-Hill Education, 2017.
2.	Deitel P., Deitel, H. : Java: How to Program: Early Objects, 11th Edition, Pearson Education, 2018.
3.	Rao, R. N.: Core Java: An Integrated Approach, New Edition, Dreamtech Press, 2008.
4.	Horstmann C. : Core Java Volume I – Fundamentals, 11th Edition, Prentice Hall, 2018.
5.	Horstmann C. : Core Java, Volume II – Advanced Features, 11th Edition, Prentice Hall, 2018.

On-line resources to be used if available as reference material	
1.	Java SE API Documentation.
2.	The Java™ Tutorials.

Practical Tips

Output

```
System.out.println("area=" + area);
//     System.out.println("area=" +
         Double.valueOf(area).toString());
```

```
System.out.print("ABC");
System.out.print("DEF");
System.out.print("HIJ");
System.out.println("ABC");
System.out.println("DEF");
System.out.println("HIJ");
/////// OUTPUT ///////
ABCDEFHIJ
ABC
DEF
HIJ
```

Input

```
import java.io.*;

public class Input1
{
  public static void main (String[] args)
  {
    double radius;
    BufferedReader br = new BufferedReader(new
        InputStreamReader(System.in));
    String line=null;
    System.out.print("Enter the radius: ");
    try
    {
      line = br.readLine();
    }
    catch (IOException ex)
    {
//    Logger.getLogger(Main.class.getName()).log(Level.SEVERE,
        null, ex);
      System.err.println("Exception: " + ex.getMessage());
    }
    radius = Double.parseDouble(line);
    System.out.println("The diameter is " + radius*2);
  }
}
```

```java
import java.util.Scanner;

public class Input2
{
    public static void main (String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a line of text: ");
        String line = scanner.nextLine();
        System.out.println("You entered: " + line);
        System.out.print("Enter the radius of the circle: ");
        double radius = scanner.nextDouble();
        System.out.println("The diameter is " + radius*2);
        System.out.print("Enter the length of the side of the
         square: ");
        int side = scanner.nextInt();
        System.out.println("The area of the square is " + side *
         side);
    }
}
```

The NetBeans IDE

From inception, the NetBeans IDE has been closely associated with Java, even though now it also has support for C, C++, PHP. HTML5, JavaScript, etc. It was the first Java IDE written in Java itself. While NetBeans supports writing standalone Java programs, the usual way is to create a Java project in NetBeans and work on it. There can be several programs, modules and libraries in a project. NetBeans projects use a build tool like Apache ANT, Apache Maven, etc. NetBeans has several features like automatic code formatting, code completion, automatic code generation, etc., that make software development easier.

- Designed for Java
- Written in Java
- The concept of project
- Apache ANT based build system
- NetBeans Project Directory Structure
    - nbproject
    - src
    - build
    - dist

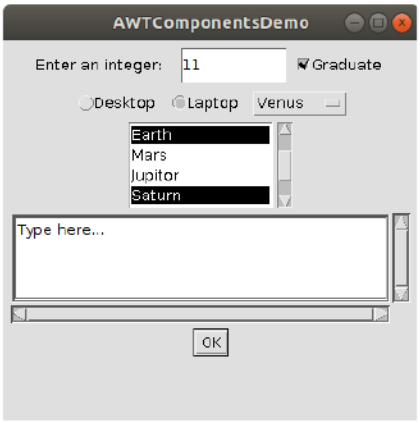
- test

Exception Handling

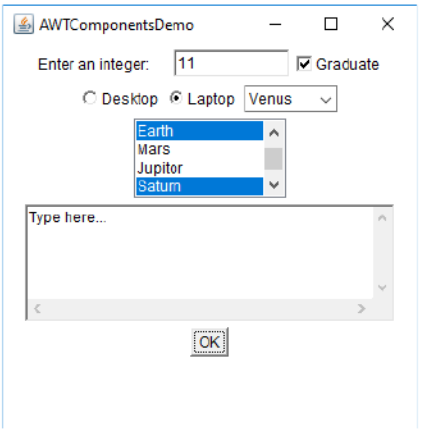
- ExceptionHandlingNew1 project
- ExceptionHandlingNew2 project
- ExceptionHandlingNew3 project

Graphical User Interface (GUI) Programming

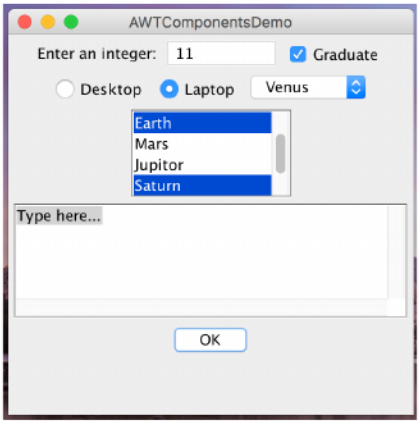
- GUI toolkits
  - AWT (Abstract Window Toolkit)
    - package: java.awt
    - The *original* Java GUI toolkit
    - A *heavyweight* toolkit
    - It uses the native system libraries for creating GUI components
    - Every AWT component has a corresponding native *peer* component. That is why it is called a heavyweight toolkit
    - Because AWT uses native components, look and feel of components differ from platform to platform
  - Swing
    - package: javax.swing
    - A *lightweight* toolkit
    - Only uses the basic graphics primitives from the operating system libraries. rest of the functionality is developed using pure Java
    - Does not have native *peer* component. that is why it is called a lightweight toolkit
    - Because Swing is 100% Java (except basic graphics primitives), look and feel are same across platforms (they may be changed using pluggable look-and-feel)



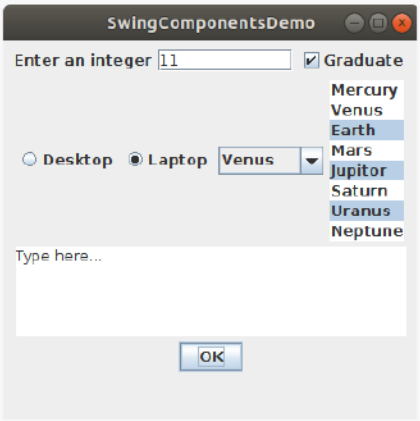
AWT Program under Linux



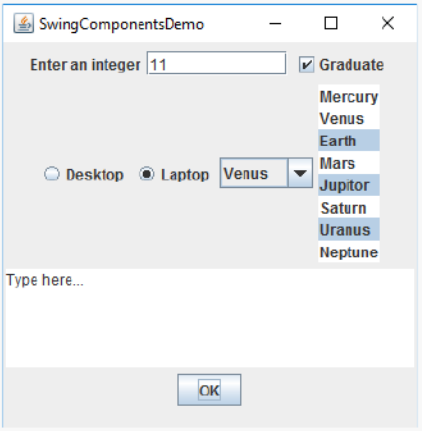
AWT Program under Windows



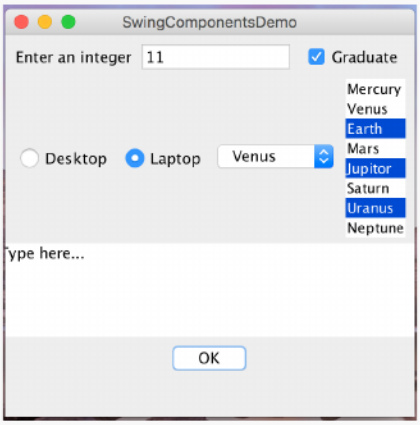
AWT Program under macOS



Swing Program under Linux



Swing Program under Windows



Swing Program under macOS

Layout Managers

In Java GUI toolkits, layout managers are used to determine the positions (and sizes) of various components. While it is possible to fix the positions of various components, it is preferred to leave the positioning to a layout manager. This allows one to write programs that adapt to and look well on displays of different sizes and resolutions.

Flow Layout

The FlowLayout layout manager arranges the components starting from the upper left corner. It places the components row-by-row, from left to right in each row.

Border Layout

The BorderLayout layout manager places the components along the four borders and the center of the panel.

AWT Components

- **Some Common AWT Components**
  - **Label** A label is used to display text that will not be edited by the user and to supply information to the user. **Useful Properties** `text`
  - **TextField** A text field used to display text that may be edited by the user. **Useful Properties** `text` , `enabled` . **Useful Methods** `getText(), setText()`
  - **Button** A button is used to let the user take some action by clicking on it. **Useful Properties** `text` . **Useful Events** `actionPerformed`

- **Checkbox** A check box can be in one of the two states - selected or unselected. If we have several check boxes, the user may select zero, one or more check boxes. However, if several check boxes are placed in a `CheckboxGroup`, then they become radio buttons and at the most one radio button can be selected from within a group. AWT does not have a separate radio button class. **Userful Properties** label. **Useful Methods** `getState()`
- **Choice** A choice component is used to let the user select one item from a dropdown list
- **List** A list component is used to display a list of items. The user may select zero or more items from the list
- **TextArea** A text area is used for input of multiline text. It can also be used to display multiline messages. **Useful Properties** `enabled`, `text`. **Useful Methods** `getText()`, `setText()`

Swing Components

- **Some Common Swing Components**
  - **JLabel** A label is used to display text that will not be edited by the user and to supply information to the user. **Useful Properties** `text`
  - **JTextField** A text field used to display text that may be edited by the user. **Useful Properties** `text`, `enabled`. **Useful Methods** `getText(), setText()`
  - **JButton** A button is used to let the user take some action by clicking on it. **Useful Properties** `text`. **Useful Events** `actionPerformed`
  - **JCheckbox** A check box can be in one of the two states - selected or unselected. If we have several check boxes, the user may select zero, one or more check boxes. **Userful Properties** `text`. **Useful Methods** `isSelected()`
  - **JRadioButton** A radio button can be in one of the two states - selected or unselected. However, if several radio buttons are placed in a `ButtonGroup`, then at the most one radio button can be selected from within a group. **Userful Properties** `text`, `buttonGroup`. **Useful Methods** `isSelected()`
  - **JComboBox** A combobox is a combination of a text field and a list. It is used to let the user select one item from a dropdown list or type a text in the text field. **Useful Properties** `model`. **Useful Methods** `getSelectedItem()`

- **JList** A list component is used to display a list of items. The user may select zero or more items from the list. How many items can be selected by the user depends on the `selectionMode` property. When selecting multiple items is allowed, the `CTRL` key can be used to select multiple items. The `SHIFT` key can be used to select a range of items between two items. **Useful Properties** `model`, `selectionMode`. **Useful Methods** `getSelectedValuesList()`
- **JTextArea** A text area is used for input of multiline text. It can also be used to display multiline messages. **Useful Properties** `enabled`, `text`. **Useful Methods** `getText()`, `setText()`

- AWTAccumulator1 project
- AWTAccumulator2 project
- AWTComponentsDemo project
- SwingDemo1 project
- SwingComponentsDemo project

```
// AWTAccumulator.java program

import java.awt.*;
import java.awt.event.*;

public class AWTAccumulator extends Frame implements
        ActionListener
{
   private Label lblInput;
   private Label lblOutput;
   private TextField tfInput;
   private TextField tfOutput;
   private int sum = 0;

   public AWTAccumulator()
   {
      setLayout(new FlowLayout());

      lblInput = new Label("Enter an Integer: ");
      add(lblInput);

      tfInput = new TextField(10);
      add(tfInput);

      tfInput.addActionListener(this);
```

```
        lblOutput = new Label("The Accumulated Sum is: ");
        add(lblOutput);

        tfOutput = new TextField(10);
        tfOutput.setEditable(false);
        add(tfOutput);

        setTitle("AWT Accumulator");
        setSize(350, 120);
        setVisible(true);

        MyWindowAdapter myWindowAdapter =
                  new MyWindowAdapter();
        this.addWindowListener(myWindowAdapter);

        /*
        this.addWindowListener(
                new WindowAdapter()
                {
                    public void windowClosing(WindowEvent we)
                    {
                          dispose();
                    }
                }
            );
        */

    }

    class MyWindowAdapter extends WindowAdapter
    {
         @Override
         public void windowClosing(WindowEvent we)
         {
             System.out.println("windowClosing()...");
             dispose();
         }

    }

    public static void main(String[] args)
    {
        AWTAccumulator frame = new AWTAccumulator();
    }

    @Override
    public void actionPerformed(ActionEvent evt)
    {
        int numberIn = Integer.parseInt(tfInput.getText());
        sum += numberIn;
//        tfOutput.setText(sum + "");
        tfOutput.setText(String.valueOf(numberIn));
```

```
        tfInput.setText("");
    }

}
```

```
// AWTComponentsDemo.java program

import java.awt.*;
import java.awt.event.*;

public class AWTComponentsDemo extends Frame
{

    private Label labelTextField;
    private TextField textFieldInteger;
    private Checkbox checkboxGraduate;
    private Label labelRadioButtonGroup1;
    private CheckboxGroup radioButtonGroup1;
    private Checkbox radioButtonDesktop;
    private Checkbox radioButtonLaptop;
    private Label labelRadioButtonGroup2;
    private CheckboxGroup radioButtonGroup2;
    private Checkbox radioButtonTubelight;
    private Checkbox radioButtonCFL;
    private Checkbox radioButtonLEDBulb;
    private Choice choice;
    private List list;
    private TextArea textArea;
    private Button buttonOK;
    private Button buttonTest;

    public AWTComponentsDemo()
    {
        setLayout(new FlowLayout());

        labelTextField = new Label("Enter an integer: ");
        add(labelTextField);

        textFieldInteger = new TextField(10);

        textFieldInteger.addKeyListener(new KeyListener()
        {
            @Override
            public void keyTyped(KeyEvent e)
            {
                System.out.println("textFieldInteger keyTyped:
        KeyChar=" + e.
                        getKeyChar());
            }
```

```
        @Override
        public void keyPressed(KeyEvent e)
        {
            System.out.println("textFieldInteger keyPressed:
    KeyCode=" +
                    e.getKeyCode() + " KeyChar=" +
    e.getKeyChar());
        }

        @Override
        public void keyReleased(KeyEvent e)
        {
            System.out.println("textFieldInteger keyReleased:
    KeyCode=" +
                    e.getKeyCode() + " KeyChar=" +
    e.getKeyChar());
        }
    });

    add(textFieldInteger);

    checkboxGraduate = new Checkbox("Graduate", true);
    add(checkboxGraduate);

    labelRadioButtonGroup1 = new Label("System Type: ");
    add(labelRadioButtonGroup1);

    radioButtonGroup1 = new CheckboxGroup();

    radioButtonDesktop = new Checkbox("Desktop",
    radioButtonGroup1, true);
    radioButtonLaptop = new Checkbox("Laptop",
    radioButtonGroup1, false);

    add(radioButtonDesktop);
    add(radioButtonLaptop);

    labelRadioButtonGroup2 = new Label("Lighting Type: ");
    add(labelRadioButtonGroup2);

    radioButtonGroup2 = new CheckboxGroup();

    radioButtonTubelight = new Checkbox("Tubelight",
    radioButtonGroup2, true);
    radioButtonCFL = new Checkbox("CFL", radioButtonGroup2,
    false);
    radioButtonLEDBulb = new Checkbox("LED Bulb",
    radioButtonGroup2, false);

    add(radioButtonTubelight);
    add(radioButtonCFL);
    add(radioButtonLEDBulb);
```

```
        choice = new Choice();
        choice.add("Mercury");
        choice.add("Venus");
        choice.add("Earth");
        choice.add("Mars");
        choice.add("Jupitor");
        choice.add("Saturn");
        choice.add("Uranus");
        choice.add("Neptune");
//        choice.select(2);
        choice.select("Earth");
        add(choice);

        list = new List();
        list.add("Mercury");
        list.add("Venus");
        list.add("Earth");
        list.add("Mars");
        list.add("Jupitor");
        list.add("Saturn");
        list.add("Uranus");
        list.add("Neptune");
        Dimension preferredSize = new Dimension(550, 500);
        list.setPreferredSize(preferredSize);
        list.setMultipleMode(true);
        list.select(2);
        list.select(5);
        add(list);

        textArea = new TextArea("Type here...", 5, 40);
        textArea.selectAll();
        add(textArea);

        MyActionListener myActionListener = new
         MyActionListener();

        buttonOK = new Button("OK");
        buttonOK.setActionCommand("ActionCommandOK");
        buttonOK.addActionListener(myActionListener);
        add(buttonOK);

        buttonTest = new Button("Test");
        buttonTest.setActionCommand("ActionCommandTest");
        buttonTest.addActionListener(myActionListener);
        add(buttonTest);



        setTitle("AWTComponentsDemo");
        setSize(350, 350);
        setVisible(true);
        textArea.requestFocusInWindow();
```

```
        this.addWindowListener(
                new WindowAdapter()
        {
            public void windowClosing(WindowEvent we)
            {
                dispose();
            }
        }
        );
    }

    void displayContents(ActionEvent e)
    {
        System.out.println("buttonOK ActionListener
        ActionCommand=" + e.getActionCommand());
        String textFieldContents = textFieldInteger.getText();
        System.out.println("textFieldContents=" +
        textFieldContents);
        System.out.println("Checkbox Graduate=" +
        checkboxGraduate.getState());
        System.out.println("RadioButton Desktop:" +
        radioButtonDesktop.getState());
        System.out.println("RadioButton Laptop:" +
        radioButtonLaptop.getState());
        System.out.println("RadioButton Tubelight:" +
        radioButtonTubelight.getState());
        System.out.println("RadioButton CFL:" +
        radioButtonCFL.getState());
        System.out.println("RadioButton LED Bulb:" +
        radioButtonLEDBulb.getState());
        System.out.println("Choice Planet=" +
        choice.getSelectedItem());
        for (String selectedItem : list.getSelectedItems())
            System.out.println("Selected Planet in the list: " +
        selectedItem);
        System.out.println("TextArea contents=" +
        textArea.getText());
    }

    class MyActionListener implements ActionListener
    {
            @Override
            public void actionPerformed(ActionEvent e)
            {
                String actionCommand = e.getActionCommand();
                System.out.println("Button pressed...
        actionCommand=" + actionCommand);
                switch (actionCommand)
                {
                    case "ActionCommandOK":
                    case "OK":
                        displayContents(e);
                        break;
                    case "ActionCommandTest":
```

```
                    case "Test":
                        System.out.println("Test button
        pressed");
                        break;
                    default:
                        System.out.println("Unknown button
        pressed");
                        break;
                }
            }
    }

    public static void main(String[] args)
    {
        AWTComponentsDemo frame = new AWTComponentsDemo();
    }

}
```

```
// SwingAccumulator.java program

import java.awt.FlowLayout;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Color;

/**
 *
 * @author class
 */
public class SwingAccumulator
{

    JFrame frame;
    JLabel jLabelInput;
    JTextField jTextFieldInput;
    JLabel jLabelAccumulator;
    JTextField jTextFieldAccumulator;
    int sum = 0;

    /**
     * Create the GUI and show it.  For thread safety,
     * this method should be invoked from the
     * event-dispatching thread.
     */
    private void createAndShowGUI()
    {
        //Create and set up the window.
```

```
        frame = new JFrame("SwingAccumulator");
        frame.setLayout(new FlowLayout());
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        jLabelInput = new JLabel("Enter an integer");
        frame.add(jLabelInput);

        jTextFieldInput = new JTextField(10);
        jTextFieldInput.addActionListener(new
        MyActionListener());
        frame.add(jTextFieldInput);

        jLabelAccumulator = new JLabel("The accumulated sum
        is:");
        frame.add(jLabelAccumulator);

        jTextFieldAccumulator = new JTextField(10);
        jTextFieldAccumulator.setEnabled(false);
//
        jTextFieldAccumulator.setDisabledTextColor(Color.DARK_GRAY),
        jTextFieldAccumulator.setDisabledTextColor(Color.RED);
        frame.add(jTextFieldAccumulator);

        //Display the window.
        frame.setSize(350, 350);
//        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        //Schedule a job for the event-dispatching thread:
        //creating and showing this application's GUI.
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                SwingAccumulator swingAccumulator = new
            SwingAccumulator();
                swingAccumulator.createAndShowGUI();
            }
        });
    }

    class MyActionListener implements ActionListener
    {
        @Override
        public void actionPerformed(ActionEvent evt)
        {
            int numberInput =
            Integer.parseInt(jTextFieldInput.getText());
            sum += numberInput;
            jTextFieldAccumulator.setText(sum + "");
            jTextFieldInput.setText("");
        }
```

```
    }

}
```

```
// SwingComponentsDemo.java program

import java.awt.Dimension;
import java.awt.FlowLayout;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.ButtonGroup;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.ListSelectionModel;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;

/**
 *
 * @author class
 */
public class SwingComponentsDemo
{

    JFrame frame;
    JLabel jLabelJTextField;
    JTextField jTextField;
    JLabel jLabelJCheckBox;
    JCheckBox jCheckBox;
    JLabel jLabelJRadioButtonDesktop;
    JRadioButton jRadioButtonDesktop;
    JLabel jLabelJRadioButtonLaptop;
    JRadioButton jRadioButtonLaptop;
    JComboBox<String> jComboBox;
    JList<String> jList;
    JTextArea jTextArea;
    JButton jButton;
    JOptionPane jOptionPane;

    /**
     * Create the GUI and show it.  For thread safety,
```

```
     * this method should be invoked from the
     * event-dispatching thread.
     */
    private void createAndShowGUI()
    {
        //Create and set up the window.

        frame = new JFrame("SwingComponentsDemo");
        frame.setLayout(new FlowLayout());
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        jLabelJTextField = new JLabel("Enter an integer");
        frame.add(jLabelJTextField);

        jTextField = new JTextField(10);
        jTextField.addKeyListener(new MyKeyAdapter());
        frame.add(jTextField);

//        jLabelJCheckBox = new JLabel("JCheckBox");
//        frame.add(jLabelJCheckBox);

        jCheckBox = new JCheckBox("Graduate", true);
        frame.add(jCheckBox);

        jLabelJRadioButtonDesktop = new JLabel("RadioButton
        Desktop");
//        frame.add(jLabelJRadioButtonDesktop);

        jRadioButtonDesktop = new JRadioButton("Desktop", true);
        frame.add(jRadioButtonDesktop);

//        jLabelJRadioButtonLaptop = new JLabel("RadioButton
        Laptop");
//        frame.add(jLabelJRadioButtonLaptop);

        jRadioButtonLaptop = new JRadioButton("Laptop", true);
        frame.add(jRadioButtonLaptop);

        ButtonGroup radioButtonGroup = new ButtonGroup();
        radioButtonGroup.add(jRadioButtonDesktop);
        radioButtonGroup.add(jRadioButtonLaptop);

        jComboBox = new JComboBox<>();
        jComboBox.addItem("Mercury");
        jComboBox.addItem("Venus");
        jComboBox.addItem("Earth");
        jComboBox.addItem("Mars");
        jComboBox.addItem("Jupitor");
        jComboBox.addItem("Saturn");
        jComboBox.addItem("Uranus");
        jComboBox.addItem("Neptune");
        jComboBox.setSelectedIndex(2);
```

```
        frame.add(jComboBox);

        String[] jListItems = {"Mercury", "Venus", "Earth",
        "Mars", "Jupitor", "Saturn", "Uranus", "Neptune",};
        jList = new JList<>(jListItems);
//      Default Value
//
        jList.setSelectionMode(ListSelectionModel.MULTIPLE_INTERVAL_
//
        jList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION),
        jList.setSelectedIndex(2);
        frame.add(jList);

        jTextArea = new JTextArea(5, 30);
        jTextArea.setText("Type here...");
        frame.add(jTextArea);
        jTextArea.selectAll();

        jButton = new JButton("OK");
        jButton.addActionListener(new ActionListener()
                {
                    @Override
                    public void actionPerformed(ActionEvent e)
                    {
//                      System.out.println("Waiting...");
//                      try
//                      {
//                          Thread.sleep(20000);
//                      }
//                      catch (InterruptedException ie)
//                      {
//                      }
                        System.out.println("----------------");
                        System.out.println("TextField:" +
        jTextField.getText());
                        System.out.println("CheckBox:" +
        jCheckBox.isSelected());
                        System.out.println("RadioButton Desktop:"
        + jRadioButtonDesktop.isSelected());
                        System.out.println("RadioButton Laptop:"
        + jRadioButtonLaptop.isSelected());
                        System.out.println("ComboBox:" +
        jComboBox.getSelectedItem());
                        System.out.print("List selected items
        (indices): ");
                        for (int selectedItem :
        jList.getSelectedIndices())
                        {
                            System.out.print(selectedItem + " ");
                        }
                        System.out.println();
                        System.out.print("List selected items
        (values): ");
```

```
                        for (String selectedItem :
        jList.getSelectedValuesList())
                        {
                            System.out.print(selectedItem + " ");
                        }
                        System.out.println();
                        System.out.println("TextArea:" +
        jTextArea.getText());
                    }
                });
        frame.add(jButton);

        //Display the window.
        frame.setSize(350, 350);
//        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        //Schedule a job for the event-dispatching thread:
        //creating and showing this application's GUI.
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                SwingComponentsDemo swingComponentsDemo = new
        SwingComponentsDemo();
                swingComponentsDemo.createAndShowGUI();
            }
        });
    }

    class MyKeyAdapter extends KeyAdapter
    {
       @Override
       public void keyTyped(KeyEvent evt)
       {
           int keyCode = evt.getKeyCode();
           char keyChar = evt.getKeyChar();
           System.out.println("Key Typed: Char=" + keyChar);
       }
    }

}
```