VIVEKANANDA GLOBAL UNIVERSITY

Master of Computer Application

Object Oriented Programming Using Java (PGCSA111)

Calculator Application Project Report

PROJECT GUIDE:
Mr. Narayana Vyas

SUBMITTED BY:
AKSHITA SHARMA (24CSA3BC001)
ABHINAV KUMAR (24CSA3BC110)
AASTHA KANWAR (24CSA3BC067)
AJEET KUMAR YADAV (24CSA3BC111)
ABHAY JANGIR (24CSA3BC053)

ACKNOWLEDGEMENT

I have taken this opportunity to express my gratitude and humble regards to the Vivekananda Global University to provide an opportunity to present a project on the "Calculator application"

I would also be thankful to my project guide Mr. Narayana Vyas to help me in the completion of my project and the documentation. I have taken efforts in this project, but the success of this project would not be possible without their support and encouragement.

I would like to thanks our Dean sir "Dr. R C Tripathi" to help us in providing all the necessary books and other stuffs as and when required .I show my gratitude to the authors whose books has been proved as the guide in the completion of my project I am also thankful to my classmates and friends who have encouraged me in the course of completion of the project.

Thanks

AJEET KUMAR YADAV (24CSA3BC111)
AASTHA KANWAR (24CSA3C067)
ABHAY JANGIR (24CSA3BC053)
AKSHITA SHARMA (24CSA3BC001)
ABHINAV KUMAR (24CSA3BC110)

Place : Jaipur
Date : 23-03-2025

DECLARATION

We hereby declare that this Project Report titled “Calculator Application” submitted by us and approved by our project guide, to the Vivekananda Global University, Jaipur is a bonafide work undertaken by us and it is not submitted to any other University or Institution for the award of any degree diploma / certificate or published any time before.

Student Name: AJEET KUMAR YADAV (24CSA3BC111)
AASTHA KANWAR (24CSA3BC067)
ABHAY JANGIR (24CSA3BC053)

Project Guide: Mr. Narayana Vyas

Table of Contents

1. Introduction

Overview of the Calculator Application

The Calculator application is a simple yet functional tool designed to help users perform fundamental arithmetic operations efficiently. With a clean and interactive graphical user interface (GUI) built using Java Swing, the calculator provides an intuitive experience for users of all levels. It enables users to execute operations such as addition, subtraction, multiplication, and division with ease, making it a useful utility for students, professionals, and anyone needing quick calculations.

This project aims to demonstrate the implementation of basic mathematical functions within a GUI-based Java application. It is an excellent example of integrating Java Swing components and handling user input through event-driven programming. The application is lightweight and runs smoothly on any system with Java installed, making it highly accessible.

Features of the Application

The key features of this calculator application include:

- **Basic Arithmetic Operations**: Supports addition (+), subtraction (-), multiplication (*), and division (/).
- **Graphical User Interface (GUI)**: Built using Java Swing to provide a visually appealing and user-friendly interface.
- **Responsive Input System**: Allows users to interact through buttons, ensuring smooth and accurate input processing.
- **Clear and Intuitive Design**: Simple layout with well-spaced buttons and a large text display for easy readability.
- **Real-Time Calculations**: Instantly computes results when the equal (=) button is pressed.
- **Error Handling**: Prevents invalid inputs and handles division by zero appropriately.
- **Cross-Platform Compatibility**: Runs on any system with Java installed, ensuring broad usability.

2. Technologies Used

Java (JDK 8 or higher)

Java is the core programming language used to develop this calculator application. It provides a robust and platform-independent environment for building software applications. The program is written in Java, ensuring it can run on any operating system that supports the Java Virtual Machine (JVM). Java's object-oriented nature also allows for efficient code organization and reuse.

Swing (GUI Framework)

Swing is a part of Java's Standard Library used to create graphical user interfaces (GUIs). It provides a wide range of components such as buttons, text fields, panels, and layouts that make it easier to design user-friendly applications. Swing enables the development of interactive applications with event-driven programming, ensuring a smooth user experience.

Event-Driven Programming

This calculator application relies on event-driven programming to handle user interactions. Java's ActionListener interface is used to detect and respond to button clicks, ensuring real-time updates and calculations. The integration of event-driven mechanisms allows the application to be responsive and interactive.

3. Project Structure

The project consists of a single main file that handles the entire functionality of the calculator.

Calculator.java (Main Application File)

This file contains the complete implementation of the calculator, including:

- **GUI Setup**: Uses Java Swing components such as JFrame, JTextField, and JButton to create an interactive interface.
- **Event Handling**: Implements ActionListener to process button clicks and perform operations accordingly.
- **Arithmetic Operations**: Handles basic calculations like addition, subtraction, multiplication, and division.
- **User Input Management**: Displays input and results in a text field while ensuring smooth data flow between operations.
- **Error Handling**: Manages potential input errors like division by zero and invalid operations.
- **Application Execution**: Initializes the calculator UI and makes it responsive to user actions.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Calculator extends JFrame implements ActionListener {
    private JTextField textField;
    private double num1, num2, result;
    private char operator;

    public Calculator() {
        setTitle("Simple Calculator");
        setSize(400, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        textField = new JTextField();
        textField.setEditable(false);
        textField.setFont(new Font("Arial", Font.BOLD, 24));
        add(textField, BorderLayout.NORTH);

        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new GridLayout(4, 4, 10, 10));

        String[] buttons = {
                "7", "8", "9", "/",
                "4", "5", "6", "*",
                "1", "2", "3", "-",
                "0", "C", "=", "+"
        };

        for (String text : buttons) {
            JButton button = new JButton(text);
            button.setFont(new Font("Arial", Font.BOLD, 20));
            button.addActionListener(this);
            buttonPanel.add(button);
        }

        add(buttonPanel, BorderLayout.CENTER);
        setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        String command = e.getActionCommand();

        if (command.charAt(0) >= '0' && command.charAt(0) <= '9') {
            textField.setText(textField.getText() + command);
        } else if (command.equals("C")) {
            textField.setText("");
            num1 = num2 = result = 0;
        } else if (command.equals("=")) {
            num2 = Double.parseDouble(textField.getText());
            switch (operator) {
                case '+': result = num1 + num2; break;
                case '-': result = num1 - num2; break;
                case '*': result = num1 * num2; break;
                case '/': result = num1 / num2; break;
            }
        textField.setText(String.valueOf(result));
        } else {
            num1 = Double.parseDouble(textField.getText());
            operator = command.charAt(0);
            textField.setText("");
        }
    }

    public static void main(String[] args) {
        new Calculator();
    }
}
```

4. Code Explanation

4.1 Importing Required Libraries

To build this application, we need to import essential Java Swing libraries such as:

- javax.swing.*: Provides GUI components like buttons, text fields, and frames.
- java.awt.*: Used for layout management and event handling.
- java.awt.event.*: Required for capturing user interactions (e.g., button clicks).

4.2 Class Definition (Calculator Class)

The Calculator class serves as the main driver for the application. It extends JFrame to create the main window and implements ActionListener to handle user interactions.

4.3 Constructor (Calculator())

The constructor initializes the calculator UI and sets up the core components:

- **Setting up JFrame**: The main window where all components are placed.
- **Creating the Display (JTextField)**: Shows user input and results.
- **Adding Buttons (JButton)**: Numeric keys (0-9), operators (+, -, *, /), clear (C), and equals (=) are created.
- **Layout Management (BorderLayout, GridLayout)**: Arranges components for an intuitive interface.

4.4 Action Handling (actionPerformed Method)

Handles button clicks and executes corresponding operations:

- **Handling Number Inputs**: Captures user input and updates the display.
- **Handling Operators (+, -, *, /)**: Stores the first number and operator for later computation.
- **Performing Calculations**: Executes the operation when = is pressed and displays the result.
- **Clearing the Display**: Resets input fields when the C button is pressed.

5. How to Run the Project

To run the calculator application, follow these steps:

Step 1: Install Java Development Kit (JDK)

Ensure that Java is installed on your system. You can check by running the following command in the terminal or command prompt:

```
java -version
```

If Java is not installed, download and install the latest JDK from the official Oracle website or install OpenJDK.

Step 2: Download or Create the Calculator.java File

Create a new file named Calculator.java and copy the source code into it.

Step 3: Compile the Code

Open the terminal or command prompt, navigate to the directory containing Calculator.java, and compile it using:

```
javac Calculator.java
```

This will generate a Calculator.class file, which is the compiled version of your Java program.

Step 4: Run the Program

Execute the compiled Java program using:

```
java Calculator
```

This will launch the calculator application with the graphical user interface.

Step 5: Use the Calculator

Once the application starts, you can interact with it by pressing the numeric keys and operators. Click the = button to get results and C to clear the display.

6. Future Enhancements

- **Adding Decimal Support**: Implement functionality to handle decimal numbers for more precise calculations.
- **Implementing Backspace Functionality**: Introduce a backspace button to allow users to delete the last digit entered.
- **Memory Functions**: Add memory storage features (M+, M-, MR) to store and recall values.
- **Improved UI with Better Styling**: Enhance the visual aesthetics using modern UI frameworks or styling techniques.
- **Keyboard Input Support**: Allow users to enter numbers and operations directly from the keyboard.
- **Scientific Functions**: Expand functionality by including trigonometric, logarithmic, and exponential calculations.
- **History Feature**: Implement a feature that stores previous calculations for reference.
- **Dark Mode**: Add a dark mode option for better visibility and user experience

7. Conclusion

The Calculator application is a simple yet effective tool for performing basic arithmetic operations. Built using Java Swing, it provides a user-friendly interface, real-time calculations, and cross-platform compatibility. With features like error handling and an intuitive design, it offers a seamless user experience.

Looking ahead, the application can be further enhanced by incorporating more advanced functions, improving UI aesthetics, and adding additional features such as history tracking and scientific calculations. This project serves as an excellent example of Java-based GUI development and can be expanded for more complex use cases in the future.

8. References

- Oracle Java Documentation: https://docs.oracle.com/en/java/
- Java Swing Tutorial: https://www.javatpoint.com/java-swing
- Event Handling in Java: https://www.geeksforgeeks.org/event-handling-in-java/

Thank You!