

Android App Development of Calculator

The Department of
INFORMATION SCIENCE AND
ENGINEERING

Submitted to

Tanzila Nargis

Preethi Salian k

Department of Information Science and
Engineering

Submitted by

Pratheeksha K N (4nm21is111)

Prathvi hegde (4nm21is113)

Abstract

This project focuses on developing a feature-rich calculator application for Android using the Java programming language and Android Studio. The aim of the application is to provide a userfriendly interface for performing basic arithmetic operations and advanced mathematical calculations on Android devices.

The development process begins with setting up the project in Android Studio, creating the necessary layout files, and designing an intuitive user interface for the calculator. The UI will include buttons for numbers, arithmetic operators, and special functions.

The core functionality of the calculator will be implemented using Java, enabling users to perform addition, subtraction, multiplication, and division operations. The application will be designed to handle user input validation to ensure accurate calculations and a smooth user experience. Real-time feedback will be provided to users while inputting and displaying results.

Throughout the development process, emphasis will be placed on optimizing the app's performance, ensuring responsiveness, and delivering an error-free user experience. Compatibility with various screen sizes and orientations will be considered to provide a consistent UI across different Android devices.

Finally, the completed calculator application will be tested extensively to identify and resolve any bugs or issues that may arise. The project's objective is to create a fully functional, efficient, and user-friendly calculator app that caters to the needs of Android users for everyday mathematical tasks.

Working

The working of a calculator app in Android Studio using Java involves several key components and functionalities. Below is a step-by-step explanation of how the calculator app operates:

1. User Interface (UI) Design:

The first step is to design the user interface (UI) for the calculator app. This is done using XML layout files in Android Studio. The layout will typically consist of buttons for numbers (0-9), arithmetic operators (+, -, *, /), and special functions (e.g., clear, equals, decimal point).

2. Button Click Listeners:

Each button in the UI will have a corresponding click listener attached to it. When a user taps on a button, the associated click listener will be triggered, allowing the app to respond to the user's input.

3. User Input Handling:

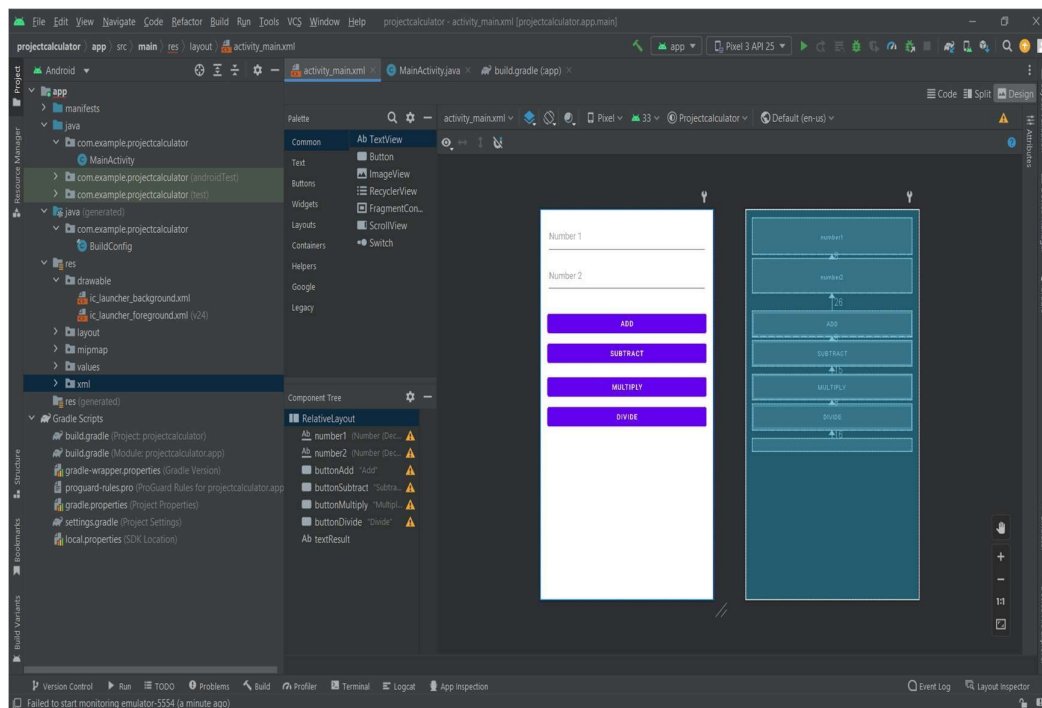
As the user interacts with the calculator, the app needs to handle the input correctly. For example, when the user presses a number button, the app should display that number on the calculator screen. If the user presses an operator, the app should store that operator until the user enters the next number or equals sign.

4. Arithmetic Operations:

When the user presses the equals (=) button, the calculator app will perform the arithmetic operation based on the numbers and operator entered by the user. This involves parsing the input, performing the calculation, and displaying the result on the calculator screen.

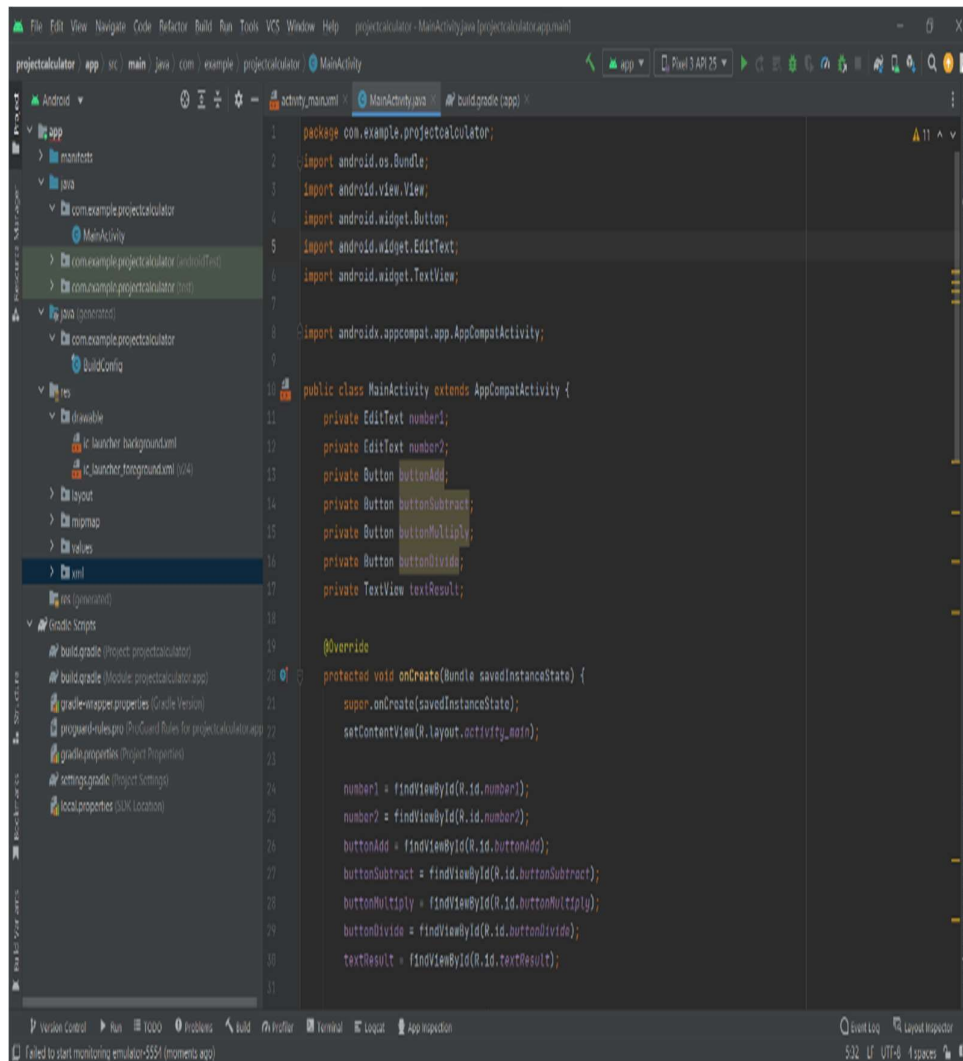
22-07-2023

Activity.xml file

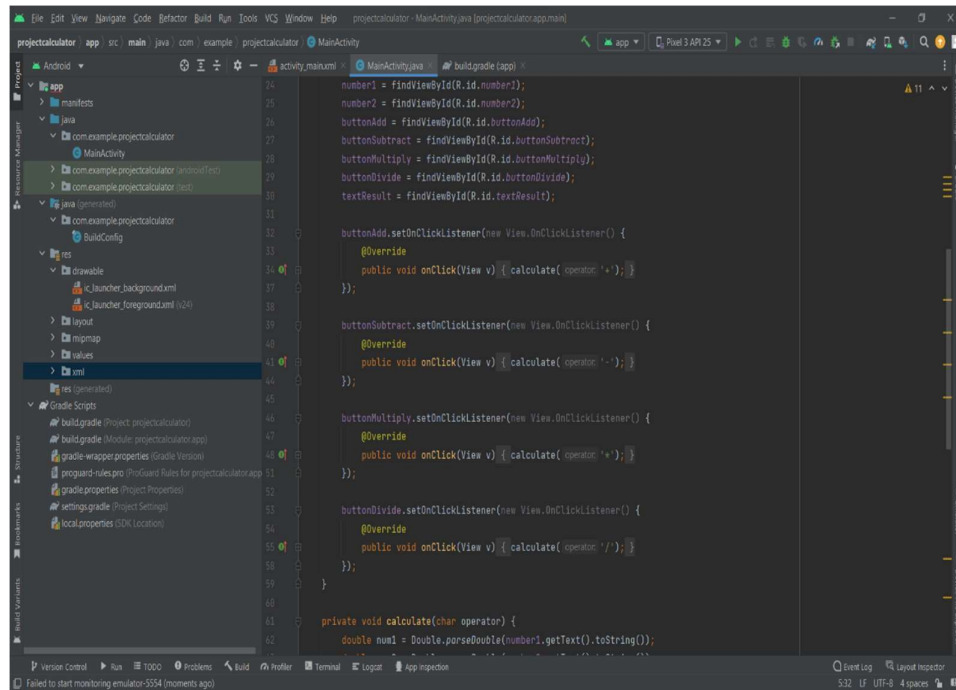


22-07-2023

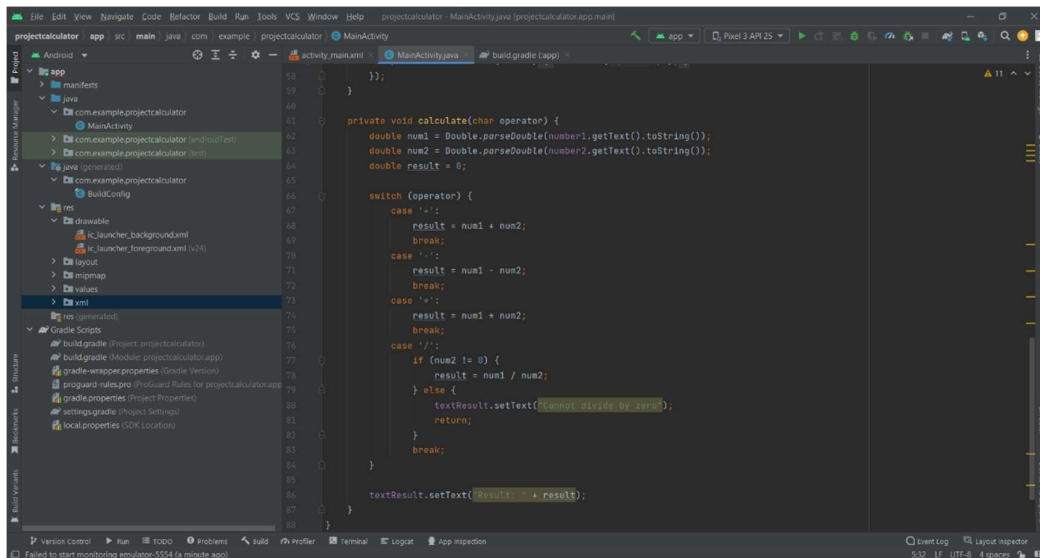
MainActivity.java



22-07-2023



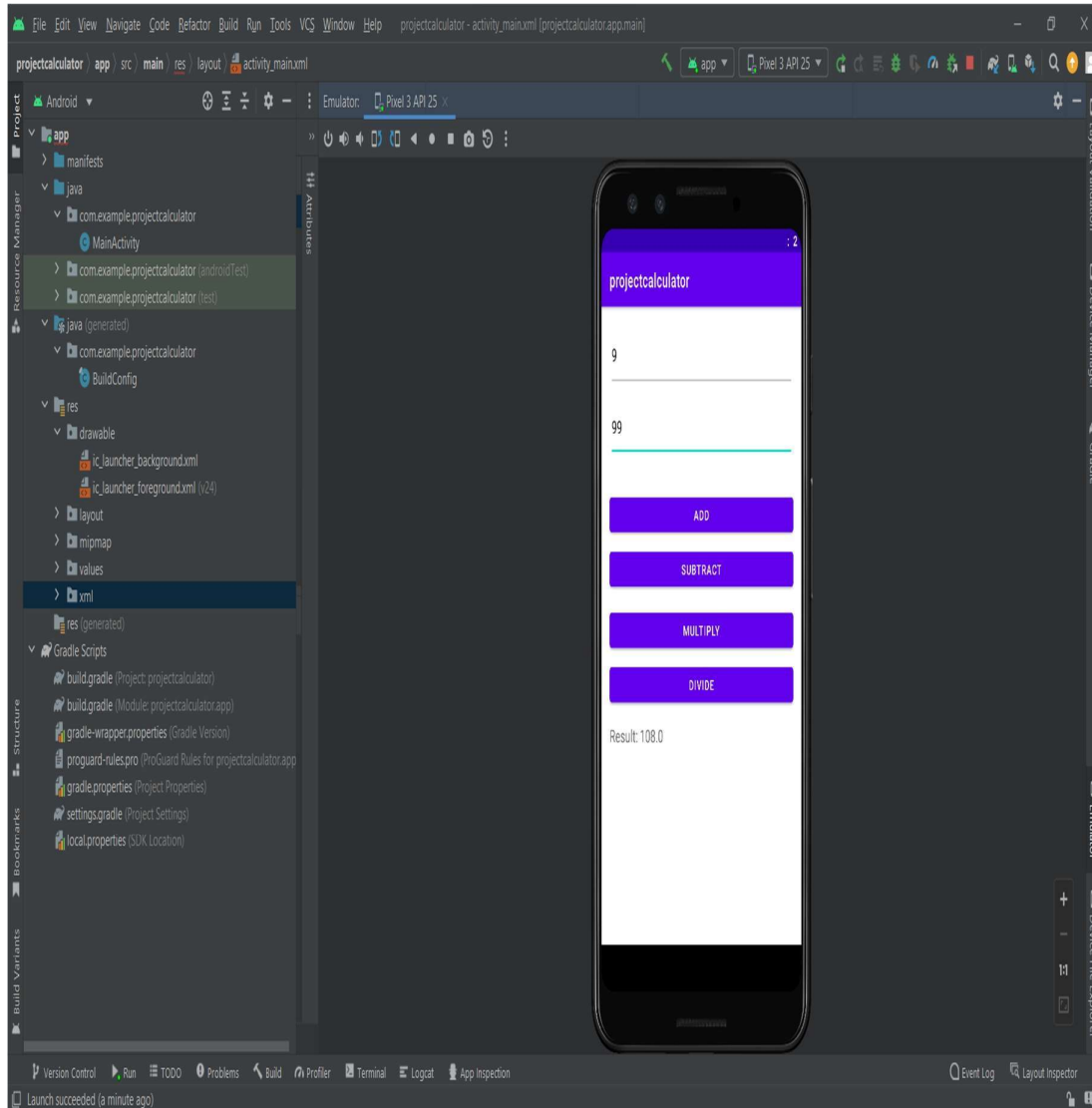
```
24: number1 = findViewById(R.id.number1);
25: number2 = findViewById(R.id.number2);
26: buttonAdd = findViewById(R.id.buttonAdd);
27: buttonSubtract = findViewById(R.id.buttonSubtract);
28: buttonMultiply = findViewById(R.id.buttonMultiply);
29: buttonDivide = findViewById(R.id.buttonDivide);
30: textResult = findViewById(R.id.textResult);
31:
32: buttonAdd.setOnClickListener(new View.OnClickListener() {
33:     @Override
34:     public void onClick(View v) { calculate(operator '+'); }
35: });
36:
37: buttonSubtract.setOnClickListener(new View.OnClickListener() {
38:     @Override
39:     public void onClick(View v) { calculate(operator '-'); }
40: });
41:
42: buttonMultiply.setOnClickListener(new View.OnClickListener() {
43:     @Override
44:     public void onClick(View v) { calculate(operator '*'); }
45: });
46:
47: buttonDivide.setOnClickListener(new View.OnClickListener() {
48:     @Override
49:     public void onClick(View v) { calculate(operator '/'); }
50: });
51:
52: private void calculate(char operator) {
53:     double num1 = Double.parseDouble(number1.getText().toString());
54:     double num2 = Double.parseDouble(number2.getText().toString());
55:     double result = 0;
56:     switch (operator) {
57:         case '+':
58:             result = num1 + num2;
59:             break;
60:         case '-':
61:             result = num1 - num2;
62:             break;
63:         case '*':
64:             result = num1 * num2;
65:             break;
66:         case '/':
67:             if (num2 != 0) {
68:                 result = num1 / num2;
69:             } else {
70:                 textResult.setText("Cannot divide by zero");
71:                 return;
72:             }
73:             break;
74:     }
75:     textResult.setText("Result: " + result);
76: }
```



```
58: });
59:
60: private void calculate(char operator) {
61:     double num1 = Double.parseDouble(number1.getText().toString());
62:     double num2 = Double.parseDouble(number2.getText().toString());
63:     double result = 0;
64:     switch (operator) {
65:         case '+':
66:             result = num1 + num2;
67:             break;
68:         case '-':
69:             result = num1 - num2;
70:             break;
71:         case '*':
72:             result = num1 * num2;
73:             break;
74:         case '/':
75:             if (num2 != 0) {
76:                 result = num1 / num2;
77:             } else {
78:                 textResult.setText("Cannot divide by zero");
79:                 return;
80:             }
81:             break;
82:     }
83:     textResult.setText("Result: " + result);
84: }
```

22-07-2023

Output



Conclusion

The calculator's core functionality was implemented in Java, allowing users to perform basic arithmetic operations, such as addition, subtraction, multiplication, and division.

the development of the calculator app was a valuable learning experience, providing insights into Android app development, Java programming, and UI/UX design. The successful completion of this project has produced a valuable tool for users, helping them with everyday mathematical calculations in a user-friendly and convenient manner.

While the current version of the calculator app meets the desired objectives, there is always room for improvement. Future enhancements could include adding scientific functions, exploring additional themes, and incorporating accessibility features to make the app more inclusive to all users.

In conclusion, the Android development of the calculator app has resulted in an efficient, and user-friendly application that showcases the power and versatility of Java and Android Studio for creating functional and practical mobile applications.