Calculator App Project

Introduction

- Creating an onClickListener, so the same method can be called by multiple buttons.
- Basic Calculator operations.
- Builds up calculation on screen, only calculates on equals key.

Designs

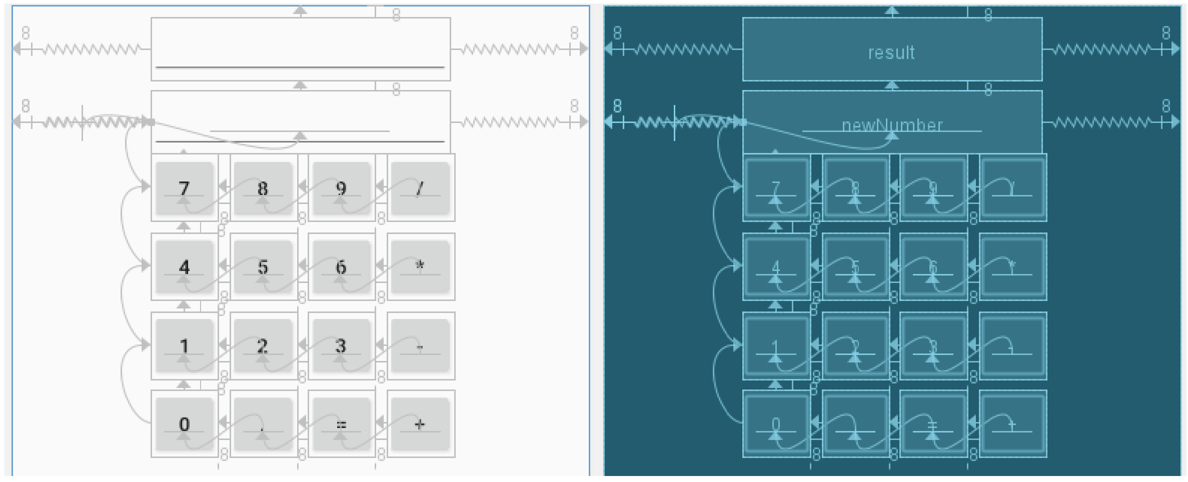


Figure 1 – Design

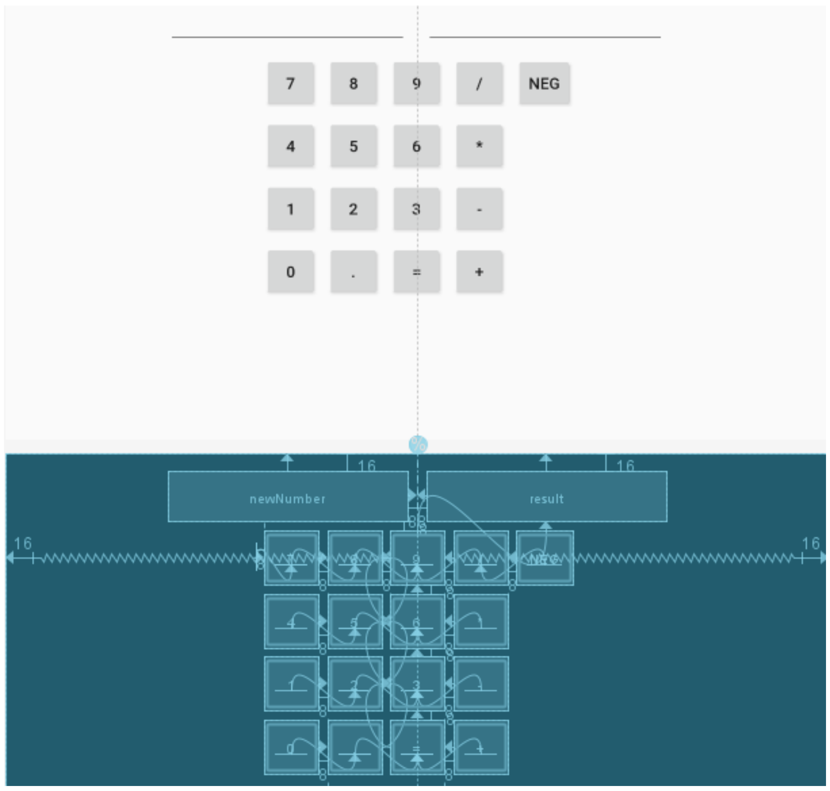


Figure 2 - Landscape Design

Code Flowchart

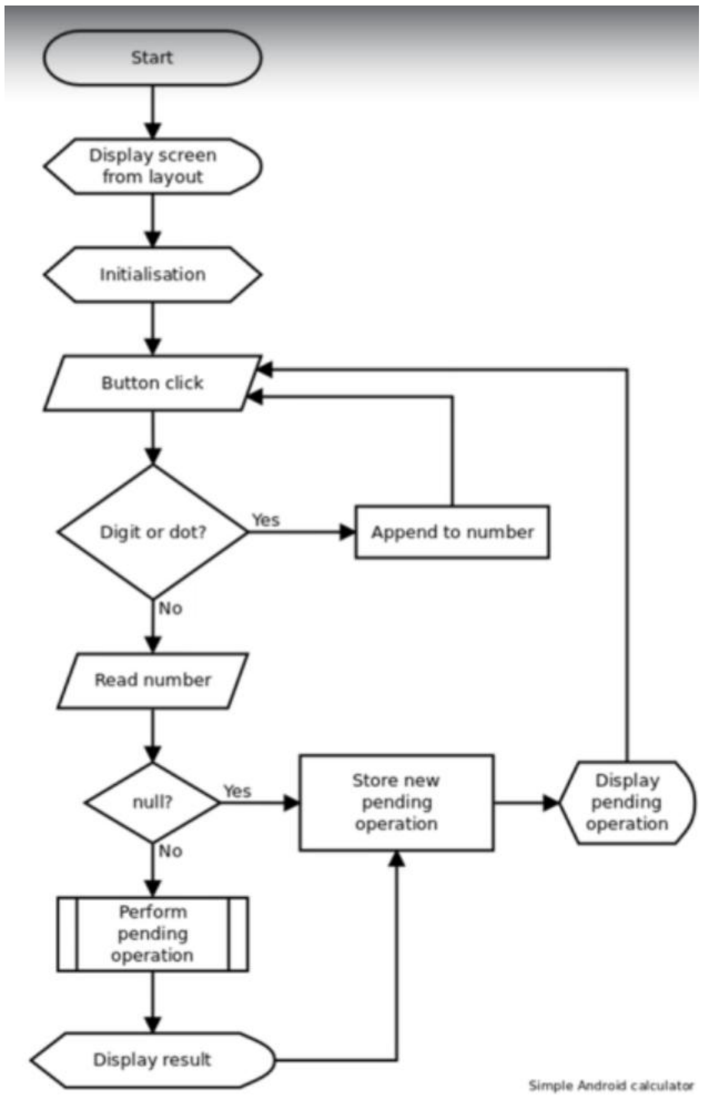
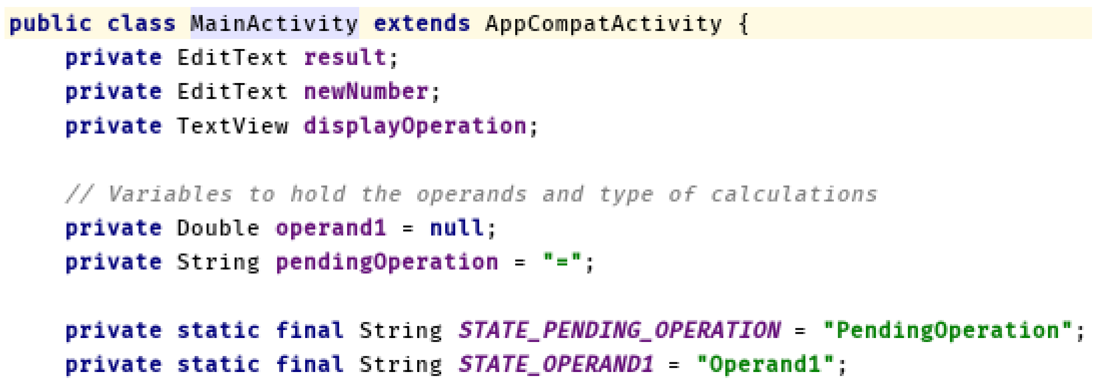

Start
Display screen from layout
Initialisation
Button click
Digit or dot?
Yes
Append to number
No
Read number
null?
Yes
Store new pending operation
Display pending operation
No
Perform pending operation
Display result
Simple Android calculator

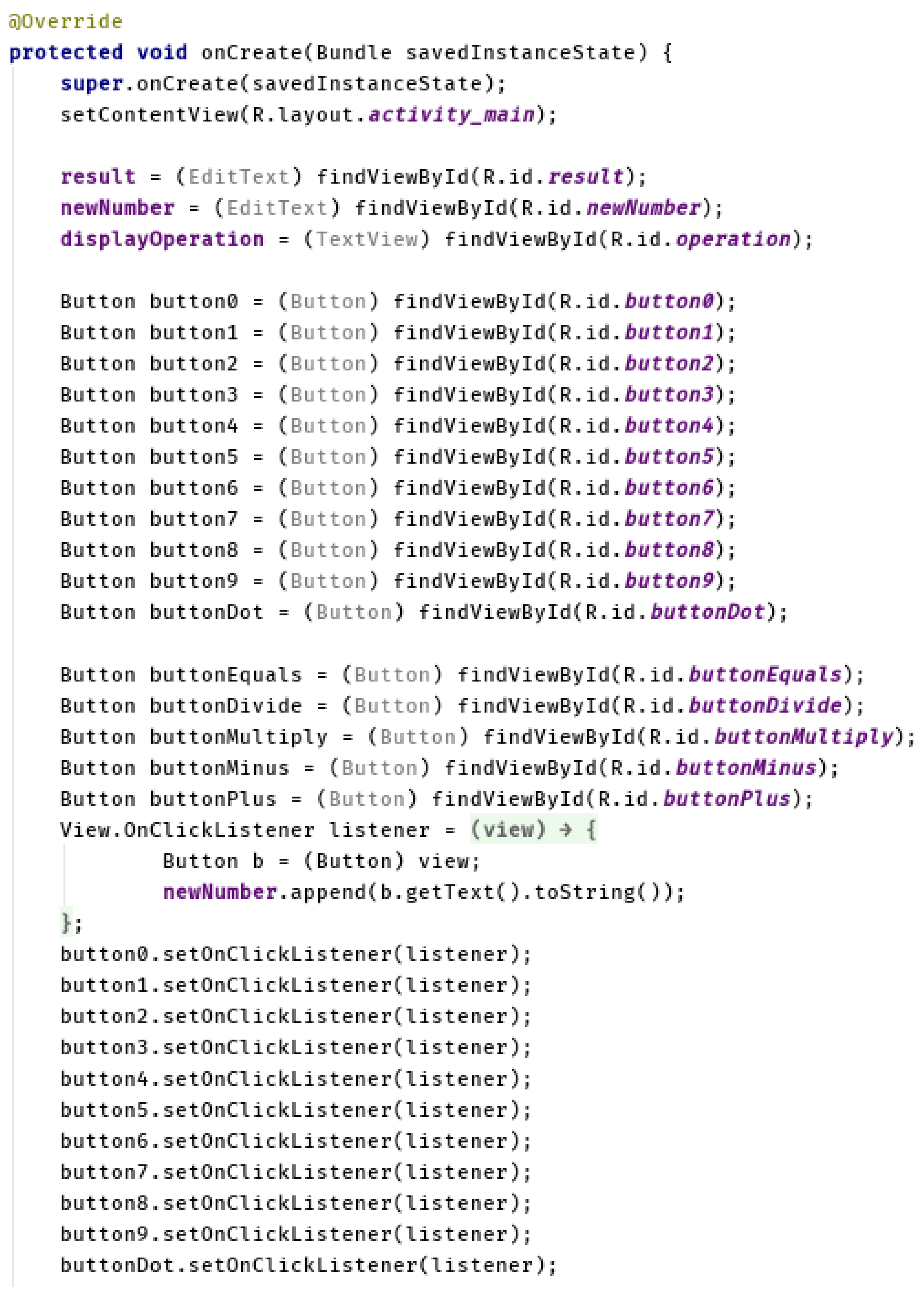
Figure 3 – Flowchart

Code



```
public class MainActivity extends AppCompatActivity {
    private EditText result;
    private EditText newNumber;
    private TextView displayOperation;

    // Variables to hold the operands and type of calculations
    private Double operand1 = null;
    private String pendingOperation = "=";

    private static final String STATE_PENDING_OPERATION = "PendingOperation";
    private static final String STATE_OPERAND1 = "Operand1";
```

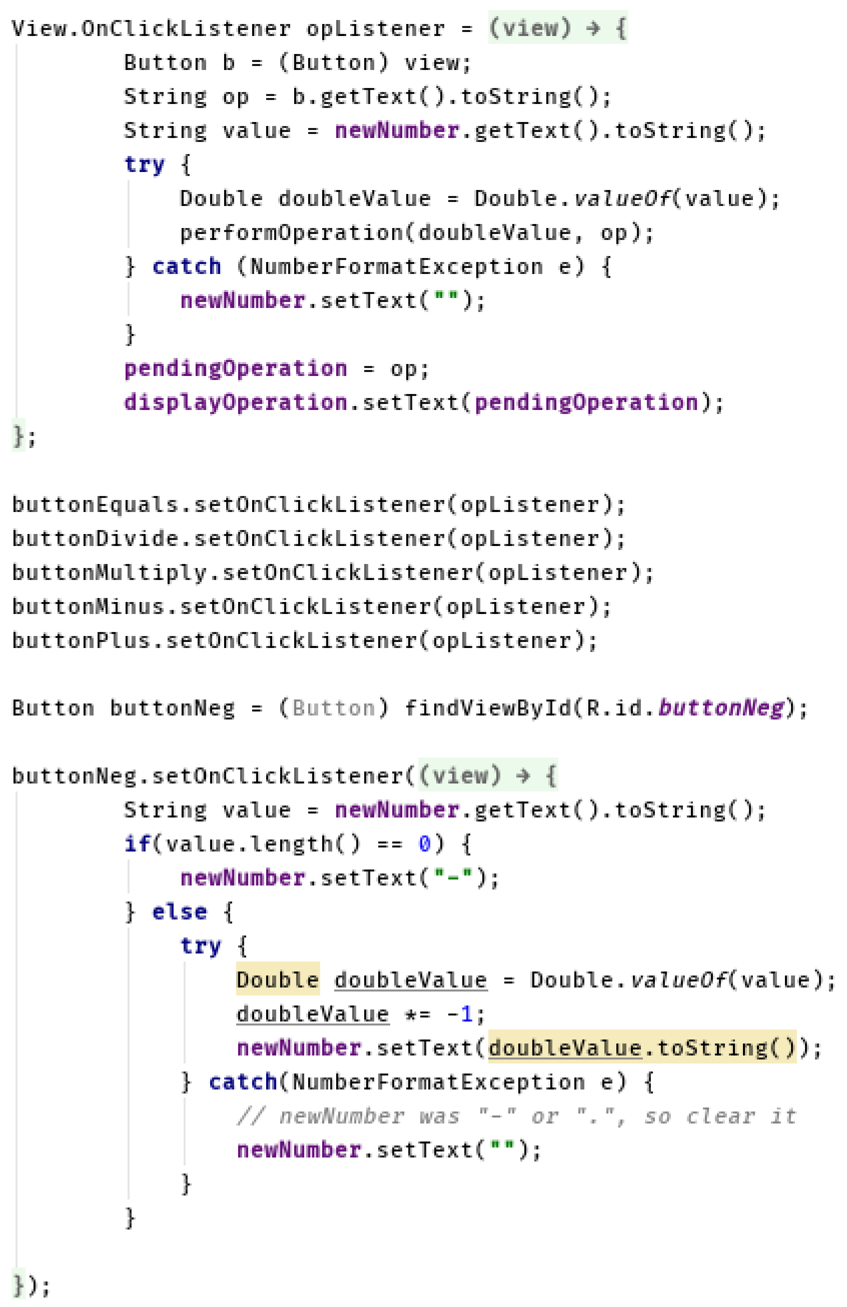
Figure 4 - Declare Variables



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    result = (EditText) findViewById(R.id.result);
    newNumber = (EditText) findViewById(R.id.newNumber);
    displayOperation = (TextView) findViewById(R.id.operation);

    Button button0 = (Button) findViewById(R.id.button0);
    Button button1 = (Button) findViewById(R.id.button1);
    Button button2 = (Button) findViewById(R.id.button2);
    Button button3 = (Button) findViewById(R.id.button3);
    Button button4 = (Button) findViewById(R.id.button4);
    Button button5 = (Button) findViewById(R.id.button5);
    Button button6 = (Button) findViewById(R.id.button6);
    Button button7 = (Button) findViewById(R.id.button7);
    Button button8 = (Button) findViewById(R.id.button8);
    Button button9 = (Button) findViewById(R.id.button9);
    Button buttonDot = (Button) findViewById(R.id.buttonDot);

    Button buttonEquals = (Button) findViewById(R.id.buttonEquals);
    Button buttonDivide = (Button) findViewById(R.id.buttonDivide);
    Button buttonMultiply = (Button) findViewById(R.id.buttonMultiply);
    Button buttonMinus = (Button) findViewById(R.id.buttonMinus);
    Button buttonPlus = (Button) findViewById(R.id.buttonPlus);
    View.OnClickListener listener = (view) → {
            Button b = (Button) view;
            newNumber.append(b.getText().toString());
    };
    button0.setOnClickListener(listener);
    button1.setOnClickListener(listener);
    button2.setOnClickListener(listener);
    button3.setOnClickListener(listener);
    button4.setOnClickListener(listener);
    button5.setOnClickListener(listener);
    button6.setOnClickListener(listener);
    button7.setOnClickListener(listener);
    button8.setOnClickListener(listener);
    button9.setOnClickListener(listener);
    buttonDot.setOnClickListener(listener);
```

Figure 5 - Buttons & Listeners

//When a button is pressed and android calls the onClick method, it passes in a reference of the button that was pressed. Not all views have text, so pass to widget (b) and can get text and append to new number.



```
View.OnClickListener opListener = (view) → {
        Button b = (Button) view;
        String op = b.getText().toString();
        String value = newNumber.getText().toString();
        try {
            Double doubleValue = Double.valueOf(value);
            performOperation(doubleValue, op);
        } catch (NumberFormatException e) {
            newNumber.setText("");
        }
        pendingOperation = op;
        displayOperation.setText(pendingOperation);
};

buttonEquals.setOnClickListener(opListener);
buttonDivide.setOnClickListener(opListener);
buttonMultiply.setOnClickListener(opListener);
buttonMinus.setOnClickListener(opListener);
buttonPlus.setOnClickListener(opListener);

Button buttonNeg = (Button) findViewById(R.id.buttonNeg);

buttonNeg.setOnClickListener((view) → {
        String value = newNumber.getText().toString();
        if(value.length() == 0) {
            newNumber.setText("-");
        } else {
            try {
                Double doubleValue = Double.valueOf(value);
                doubleValue *= -1;
                newNumber.setText(doubleValue.toString());
            } catch(NumberFormatException e) {
                // newNumber was "-" or ".", so clear it
                newNumber.setText("");
            }
        }

});
```

Figure 6 - Operator Listener

//Create a new instance of onClickListener, assigned to variable opListener. Casts the view to a button (b), then reads text and assigns it to a string (op). Also reads number (newNumber), from the editText widget. Set displayOperation to the op. Then assigns listener to the buttons.

Implement Operations

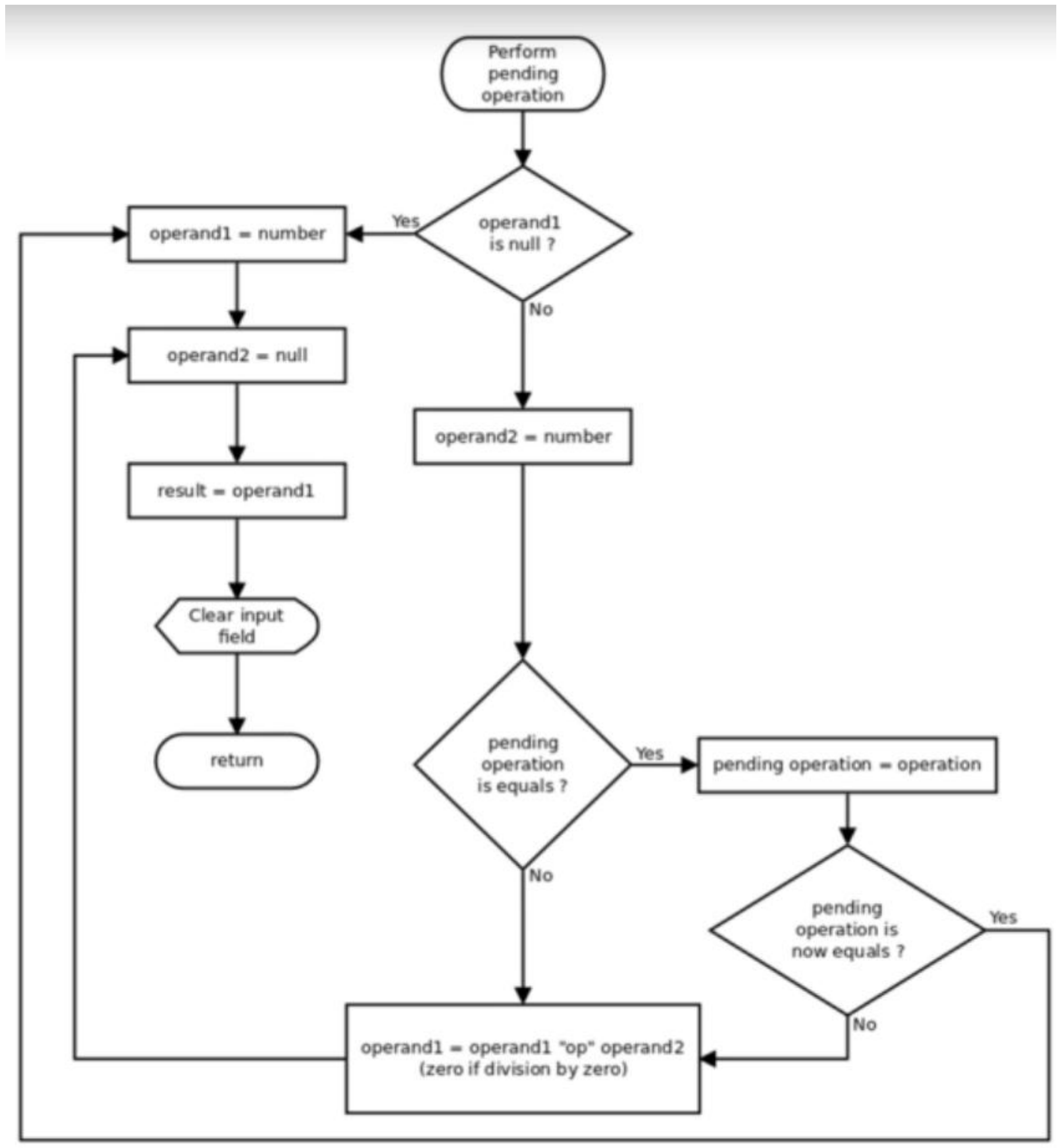
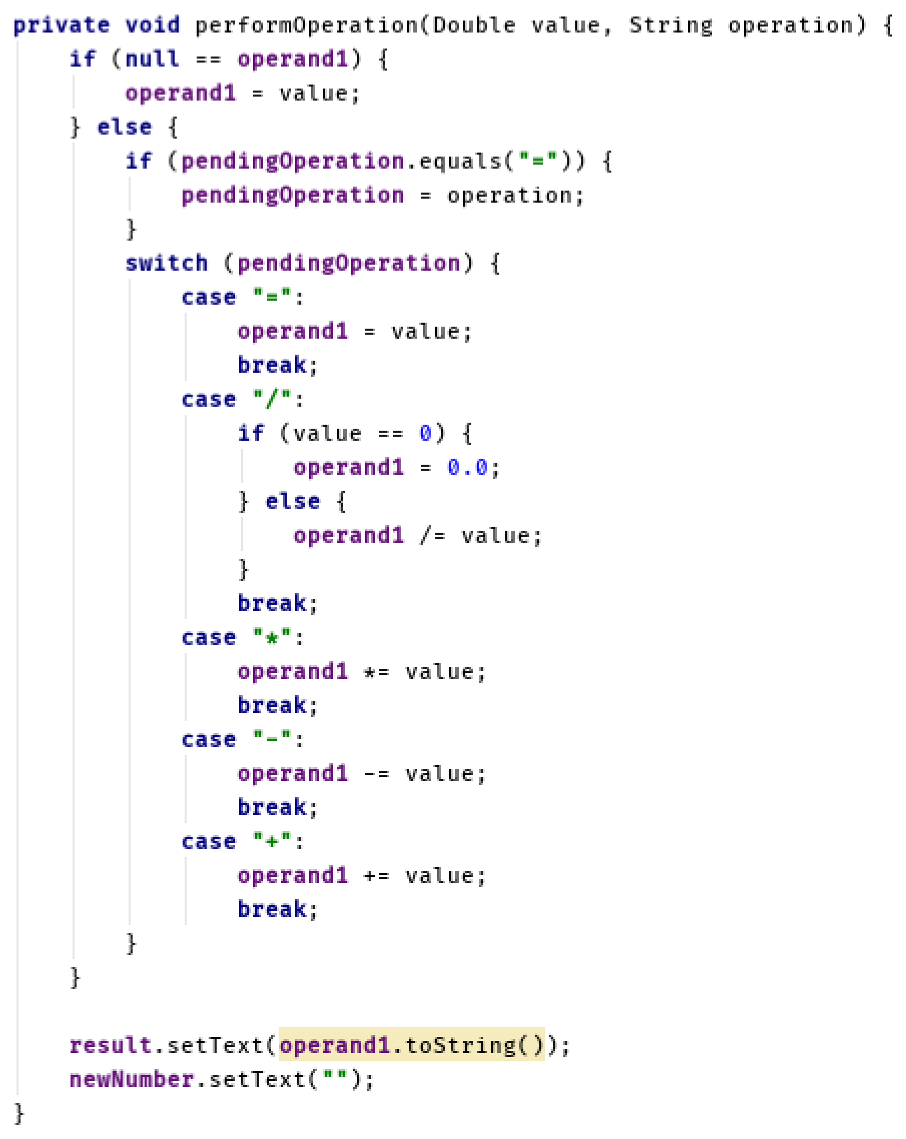

Perform pending operation
operand1 is null ?
Yes
No
operand1 = number
operand2 = null
result = operand1
Clear input field
return
operand2 = number
pending operation is equals ?
Yes
No
pending operation = operation
pending operation is now equals ?
Yes
No
operand1 = operand1 "op" operand2 (zero if division by zero)

Figure 7 - Operations Flowchart

//performOperation method is called with 2 parameters – String holding value of the EditText, we've already check if it's empty. Second – String holding the symbol that represents the operation. Need to check op1, if null there is no calculation to perform, but need to store value in the operand and update result on screen. Input widget is then cleared.

If op1 is not null, then value passed must be op2. There will always be am op pending, as initialised pendingOperation to be "=".

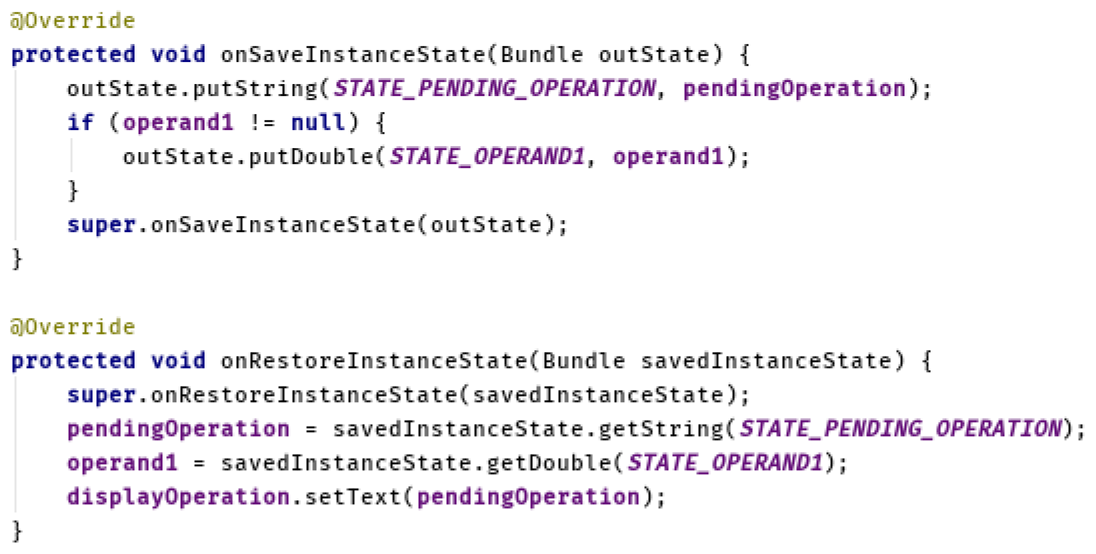
If pending operation is equals, set it to the operation that caused this method to be called – the one passed in the second parameter. If was "=" no arithmetic operation to be performed, so transfer value to op1 and display result (just update display with new input). No need to clear op2 as we will get a new op2 anyway.



```java
private void performOperation(Double value, String operation) {
    if (null == operand1) {
        operand1 = value;
    } else {
        if (pendingOperation.equals("=")) {
            pendingOperation = operation;
        }
        switch (pendingOperation) {
            case "=":
                operand1 = value;
                break;
            case "/":
                if (value == 0) {
                    operand1 = 0.0;
                } else {
                    operand1 /= value;
                }
                break;
            case "*":
                operand1 *= value;
                break;
            case "-":
                operand1 -= value;
                break;
            case "+":
                operand1 += value;
                break;
        }
    }

    result.setText(operand1.toString());
    newNumber.setText("");
}
```

Figure 8 - performOperation Method

Saved States



```java
@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putString(STATE_PENDING_OPERATION, pendingOperation);
    if (operand1 != null) {
        outState.putDouble(STATE_OPERAND1, operand1);
    }
    super.onSaveInstanceState(outState);
}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    pendingOperation = savedInstanceState.getString(STATE_PENDING_OPERATION);
    operand1 = savedInstanceState.getDouble(STATE_OPERAND1);
    displayOperation.setText(pendingOperation);
}
```

Figure 9 - Saved States Override

//Storing and retrieving the apps state when the device is rotated

//OnSavedInstanceState is called if needed, when android is about to destroy the Activity – as it does when the orientation changes.

//State can be restored in either OnCreate or OnRestorInstanceState. Using OnRestorInstanceState, as good idea to separate code, to make it clearer. Also if you try to use the Bundle in OnCreate you have to check if it's not null, but in OnRestorInstanceState it will only ever be called if there is a valid Bundle.