MOBI3002: A2 – UI – Calculator

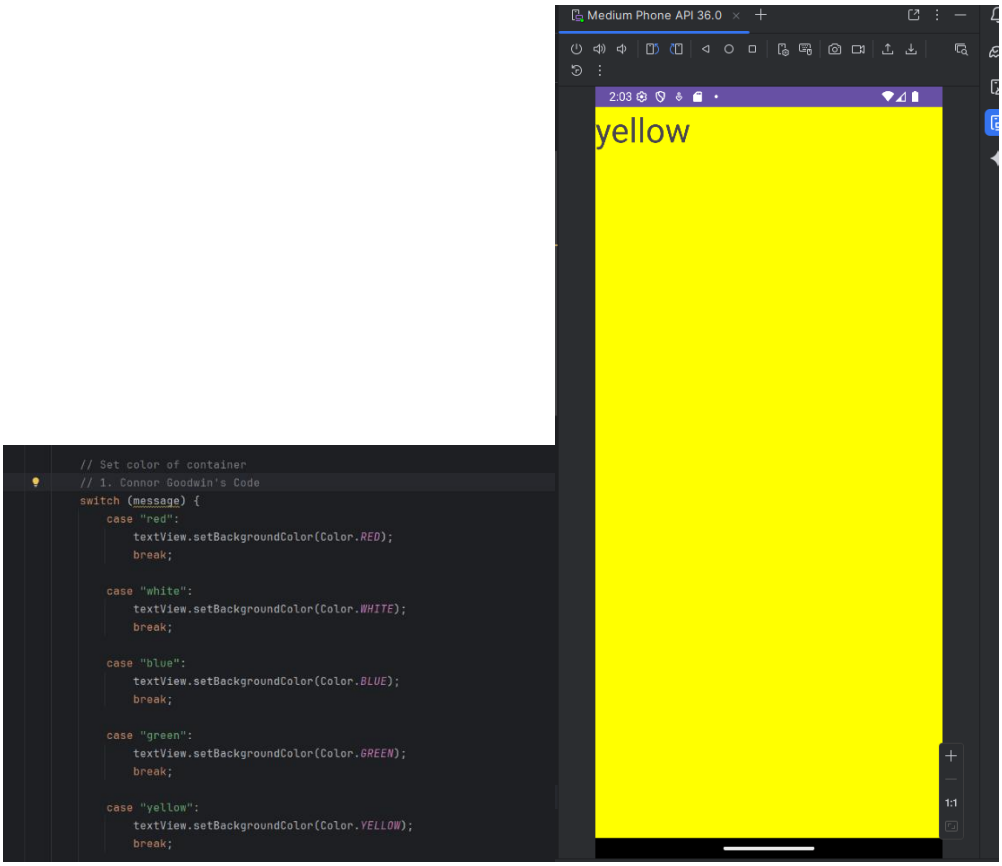
Assignment Submission

By: Connor Goodwin

W#: W0488245

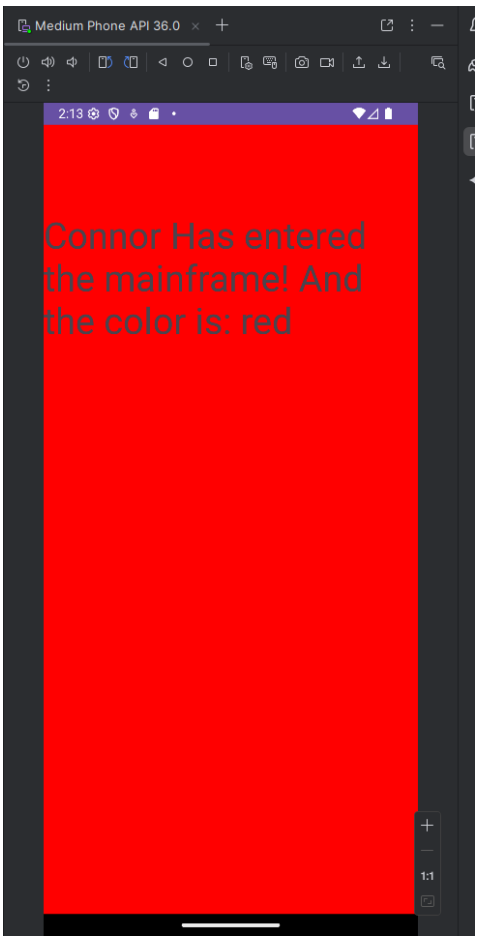
Date: 2025-09-15

1. **Make the DisplayMessageActivity background change to the colour entered in the text message using 5 different colours.**



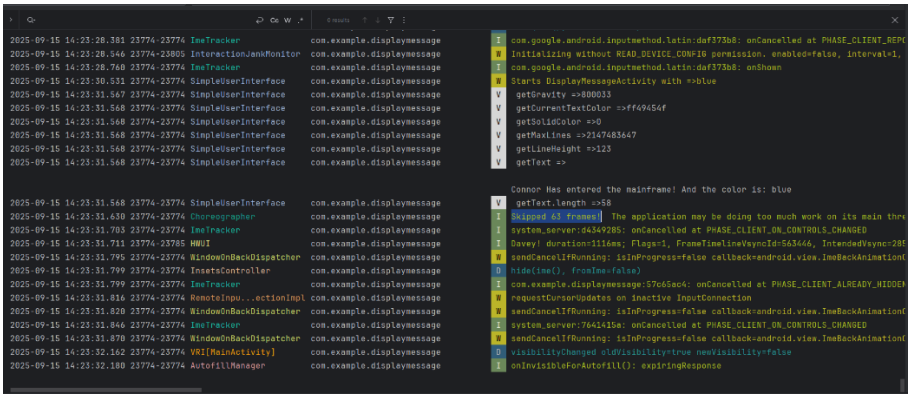
2. **Add your own message to the message sent to the activity, and display the whole message.**

```
// Add to the text
// 2. Connor Goodwin's Code
textView.setText("\n\nConnor Has entered the mainframe! And the color is: " + message);
```



3. **Add a delay in the DisplayMessageActivity onCreate() method...see if you can get "Skipped 61 frames! The application may be doing too much work on its main thread.".**

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // Connor Goodwin's Code 3.
    try {
        Thread.sleep( millis: 999);
    } catch (InterruptedException e) {
        throw new RuntimeException(e);
    }
```



3A. Why do you get the error?

I get the error because when you yield a thread using Thread.Sleep, you are stopping all code after it from executing within the thread context. In the context of the main thread, it is responsible for drawing frames, that is why the error says “Skipped x frames” because it wasn’t able to draw those frames while the thread was sleeping or yielded.

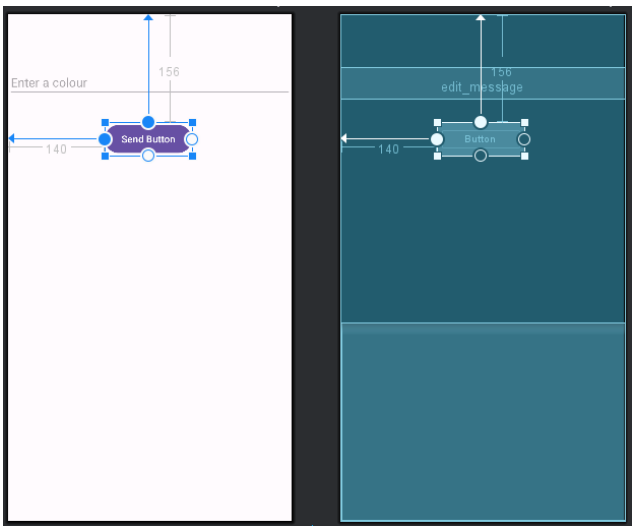
3B. How could you avoid it?

Avoid yielding the main thread, the thread that draws frames. If you need certain code to sleep, you could spawn a new thread, and have the code within it’s context sleep, rather than halting the main thread.

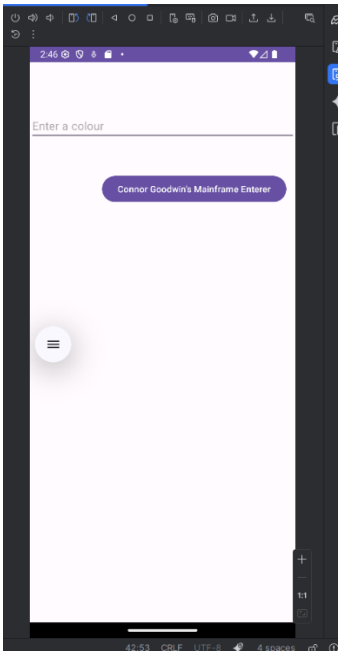
3C. What is the effect on usability of an Android App.?

It will decrease the framerate of the app itself, which means that the app will appear choppy and “laggy” or slow. This ruins user experience and can lead the user to blaming the device they are using rather than the app itself.

4. **Change the position of the "Send" button (use the design view and drag and drop it somewhere else on the screen)**



5. **Change the text in the "Send" button to something with your name in it.**

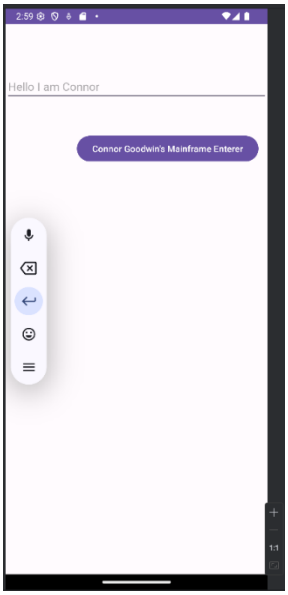


6. **Change the hint text from "Enter a colour" to something with your name in it ("Enter message to Russ").**

```
<EditText
    android:id="@+id/edit_message"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="76dp"
    android:hint="@string/enter_message_to_russ"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

7. **Change the name of the 2nd activity from "DisplayMessageActivity" to something with your name in it. (use strings.xml file... From the code line 13 of DisplayMessageActivity class... public class DisplayMessageActivity extends AppCompatActivity {}**

```
<resources>
    <string name="app_name">Simple User Interface</string>

    <string name="action_settings">Settings</string>

    <string name="edit_message">Enter a colour</string>
    <string name="button_send">Send</string>
    <string name="title_activity_display_message">Russ Message</string>

    <string name="hello_world">Hello world!</string>
    <string name="enter_message_to_russ">Hello I am Connor</string>

</resources>
```



[80%] Part 2: Calculator

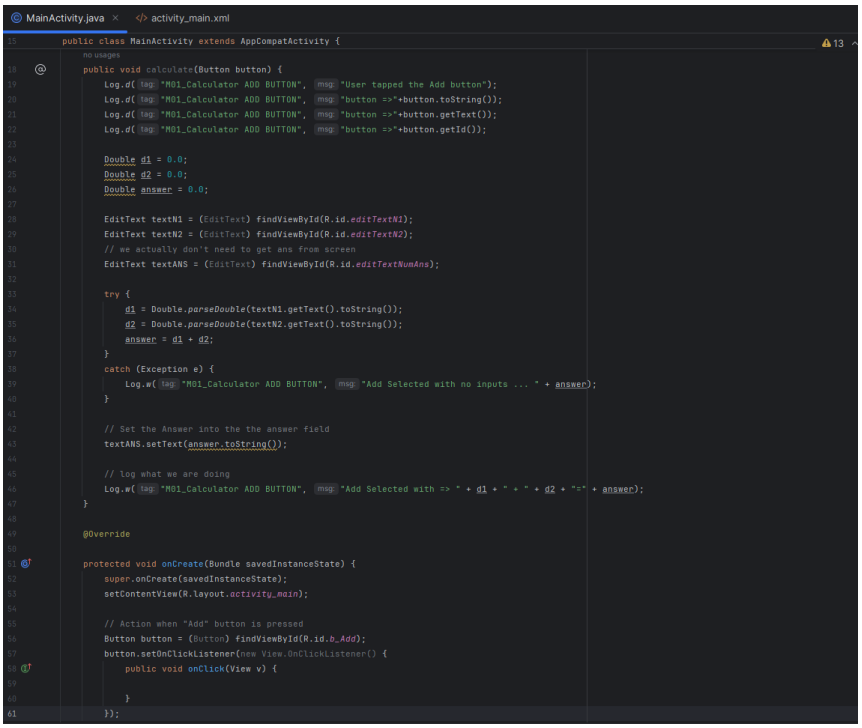
1. **Create a working Calculator App.**

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginTop="288dp"
    android:text="Subtract"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="288dp"
    android:text="Divide"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.85"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/button3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="272dp"
    android:layout_marginTop="460dp"
    android:text="Multiply"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

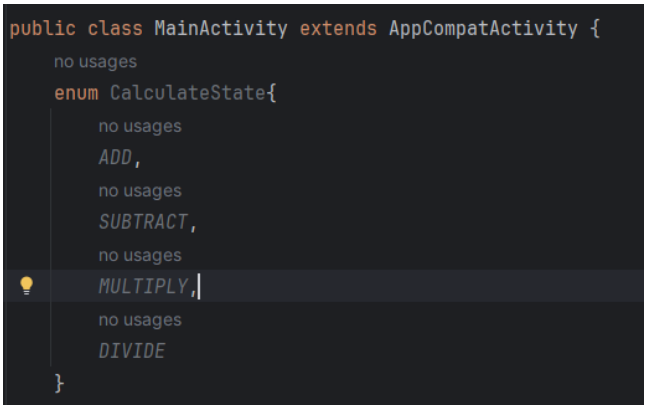
First change: Changed button text SUBTRACT to Subtrack, and then the two Button text to divide and multiply.

```
<Button
    android:id="@+id/b_Subtract"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginTop="288dp"
    android:text="Subtract"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/b_Divide"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="288dp"
    android:text="Divide"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.85"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/b_Multiply"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="272dp"
    android:layout_marginTop="460dp"
    android:text="Multiply"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Second change: Changed the button ids to be more readable.



Third change: Added the calculate function, with the plan on adding 3 more event listeners for the rest of the buttons.



Fourth change: Added enums to act as the only 4 things the calculate function can do. Every time the calculate function is called, one of these 4 enums will be passed to the function, to be used in an if statement to determine correct calculation.

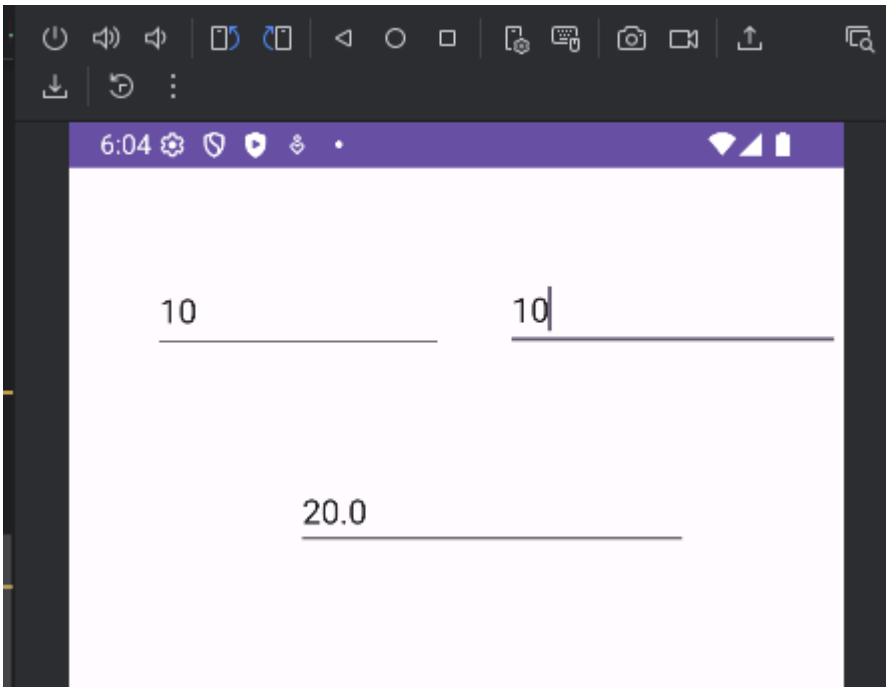
```java
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // Action when "Add" button is pressed
    Button addButton = (Button) findViewById(R.id.b_Add);
    addButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            calculate(addButton, CalculateState.ADD);
        }
    });

    Button subtractButton = (Button) findViewById(R.id.b_Subtract);
    subtractButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            calculate(subtractButton, CalculateState.SUBTRACT);
        }
    });

    Button multiplyButton = (Button) findViewById(R.id.b_Multiply);
    multiplyButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            calculate(multiplyButton, CalculateState.MULTIPLY);
        }
    });

    Button divideButton = (Button) findViewById(R.id.b_Divide);
    divideButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            calculate(divideButton, CalculateState.DIVIDE);
        }
    });
}
```

Fifth Change: Added event listeners to the last 3 buttons: subtract, multiply and divide. Each passing their button and the enum for the type of calculation that they will perform.

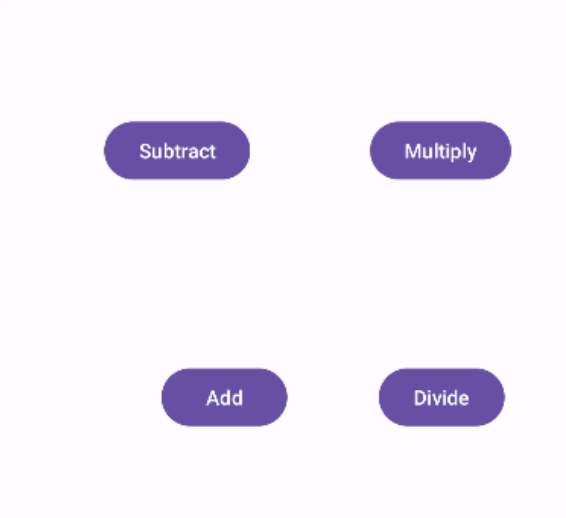
```java
if (state == CalculateState.ADD) {
    answer = d1 + d2;
} else if (state == CalculateState.SUBTRACT) {
    answer = d1 - d2;
} else if (state == CalculateState.MULTIPLY) {
    answer = d1 * d2;
} else if (state == CalculateState.DIVIDE) {
    answer = d1 / d2;
}
```

Sixth Change: Added the if statement to check which calculation should be performed based on the enum passed to the calculate function. So clicking the add button passes the add enum, etc. Done!

2. **input 2 numbers**



3. **set 4 buttons (for add, subtract, multiply and divide)**



4. **one field for the answer (set as read-only)**

```
<EditText
    android:id="@+id/editTextNumAns"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="120dp"
    android:layout_marginTop="160dp"
    android:ems="10"
    android:focusable="false"
    android:clickable="false"
    android:inputType="none"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Set as read only by setting focusable and clickable to false.

5. **id attributes are set properly (don't leave it as editText....use a pattern that others can understand, such as editText_Num1, editText_Num2, etc...**

```
EditText textN1 = (EditText) findViewById(R.id.editText_Num1);
EditText textN2 = (EditText) findViewById(R.id.editText_Num2);
```

```
<EditText
    android:id="@+id/editText_Num1"
    android:layout_width="155dp"
    android:layout_height="48dp"
    android:layout_marginStart="44dp"
    android:layout_marginTop="52dp"
    android:ems="10"
    android:inputType="numberDecimal"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<EditText
    android:id="@+id/editText_Num2"
    android:layout_width="179dp"
    android:layout_height="47dp"
    android:layout_marginStart="32dp"
    android:layout_marginTop="52dp"
    android:ems="10"
    android:inputType="numberDecimal"
    app:layout_constraintStart_toEndOf="@+id/editText_Num1"
    app:layout_constraintTop_toTopOf="parent" />
```