Android Calculator

SEG2105 - Introduction to Software Engineering – Fall 2021

Lab 1 (2%)

Lab Goals

- Give you exposure to the Android Studio tool.
- Learn about event-driven systems, such as clicking a button and having an event occur.
- Become familiar with creating user interfaces and connecting code to various elements.
- Learn to troubleshoot (if necessary).

Lab Objectives

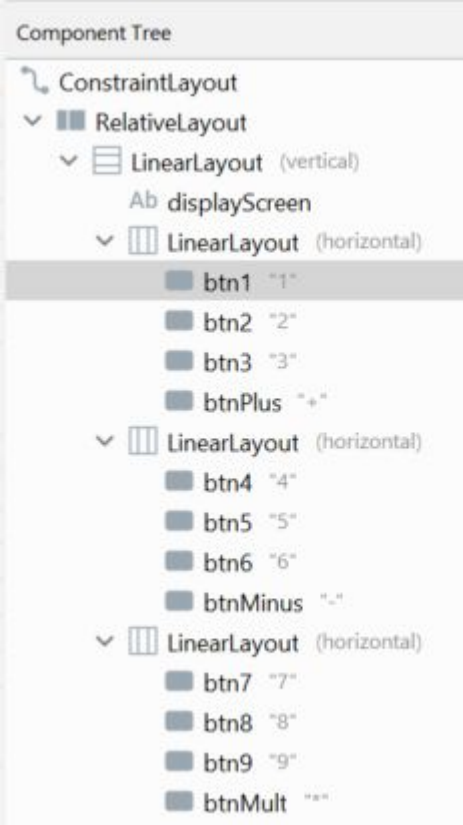
- Create a simple calculator app
- Include the following operations:
  - addition (+)
  - subtraction (-)
  - multiplication (*)
  - division (/)
- Include:
  - decimal (32.5)
  - clear (clears the screen)
  - equal (displays the result)
- Consider at least 2 edge cases:
  - some examples are given on the next slide

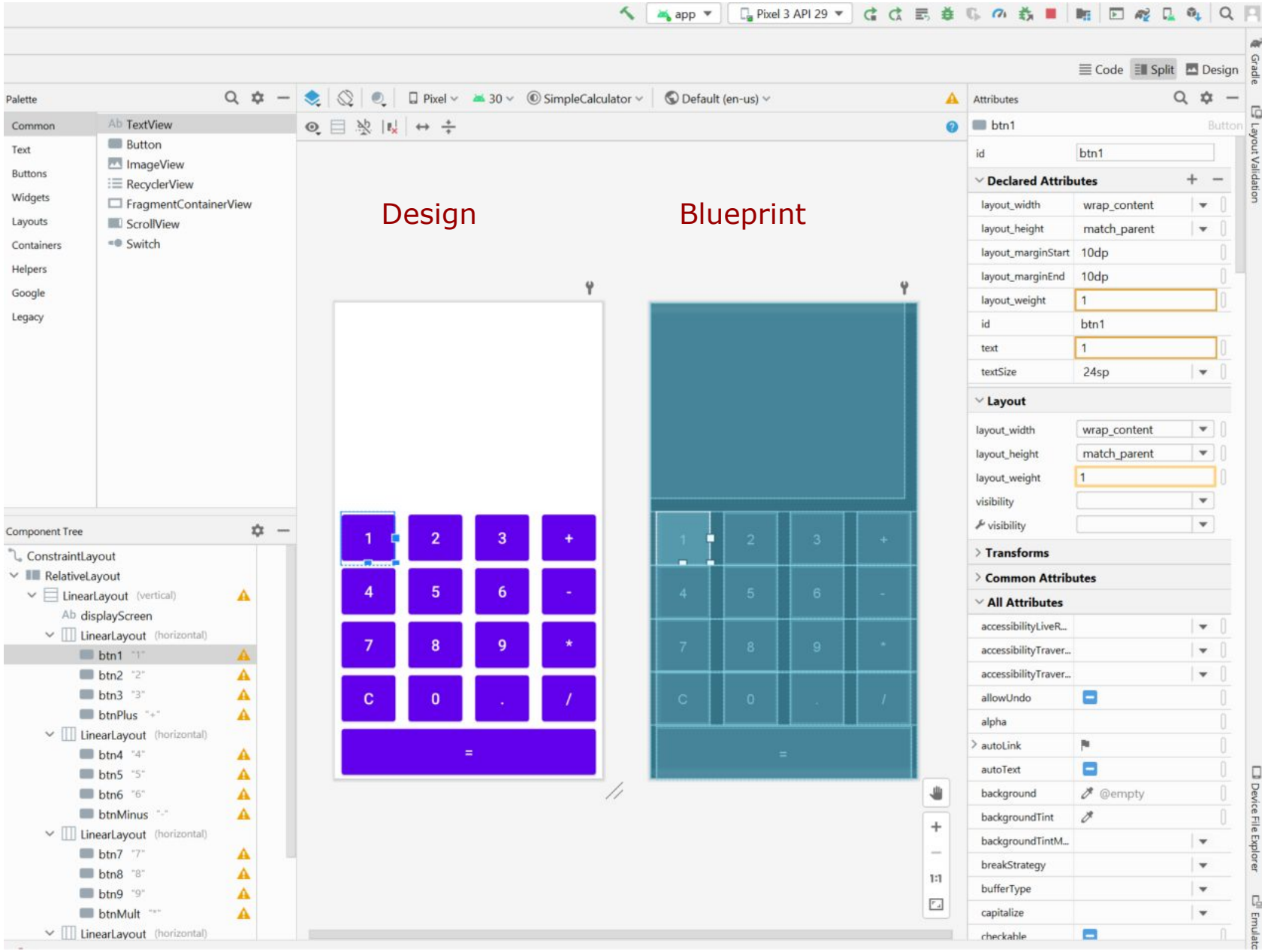
Considerations

- You might want to consider:
  - What happens when the screen is cleared and user clicks the "=" button? The app should not crash.
  - What happens when the user enters a negative number? (ex. -5+3) Does your code understand the input?
  - What if the user is performing float operations (ex. 5.3+1)? Does the result display "6.3" instead of "6"?
  - What if the user is doing integer operations (ex. 3+6)? Does the result display "9" instead of "9.0"?

Layout

- You can use a RelativeLayout with LinearLayouts, as show in the screenshot.
- You can use the ConstraintLayout to create different constraints and build a responsive UI.
  - Ex. constrain a button to the left side of another button.
  - https://developer.android.com/training/basics/firstapp/building-ui
- Sometimes it is easier to drag and drop elements from the palette to the Component Tree.
- Make sure to give all important elements an id.




app
Pixel 3 API 29
Code
Split
Design
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentContainerView
ScrollView
Switch
Pixel
30
SimpleCalculator
Default (en-us)
Design
Blueprint
1
2
3
+
4
5
6
-
7
8
9
*
C
0
.
/
=
Component Tree
ConstraintLayout
RelativeLayout
LinearLayout (vertical)
displayScreen
LinearLayout (horizontal)
btn1 "1"
btn2 "2"
btn3 "3"
btnPlus "+"
LinearLayout (horizontal)
btn4 "4"
btn5 "5"
btn6 "6"
btnMinus "-"
LinearLayout (horizontal)
btn7 "7"
btn8 "8"
btn9 "9"
btnMult "*"
LinearLayout (horizontal)
Attributes
btn1
Button
id
btn1
Declared Attributes
layout_width
wrap_content
layout_height
match_parent
layout_marginStart
10dp
layout_marginEnd
10dp
layout_weight
1
id
btn1
text
1
textSize
24sp
Layout
layout_width
wrap_content
layout_height
match_parent
layout_weight
1
visibility
visibility
Transforms
Common Attributes
All Attributes
accessibilityLiveR...
accessibilityTraver...
accessibilityTraver...
allowUndo
alpha
autoLink
autoText
background
@empty
backgroundTint
backgroundTintM...
breakStrategy
bufferType
capitalize
checkable
Gradle
Layout Validation
Device File Explorer
Emulato

- Sometimes you will want to declare attributes using the .xml code instead.
- Get comfortable with both the Code and Design view that Android Studio offers.

activity_main.xml | MainActivity.java

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="h
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical">

            <TextView
                android:id="@+id/displayScreen"
                android:layout_width="match_parent"
                android:layout_height="300dp"
                android:layout_marginEnd="20dp"
                android:layout_marginBottom="20dp"
                android:gravity="bottom|end"
                android:textAlignment="viewEnd"
                android:textSize="48sp" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:orientation="horizontal">
```

Interface

- You have full flexibility to experiment with the layout and design of your application.

Source: Apple

Source: Blackberry

Variable declaration

- This example shows a boolean approach for the calculator logic.
- The double and boolean variables will be used for the mathematical operations.

activity_main.xml | MainActivity.java

```java
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    Button btn0, btn1, btn2, btn3, btn4, btn5, btn6, btn7, btn8, btn9, btnAdd, btnSub, btnDiv,
    btnMul, btnClr, btnEql, btnDec;

    TextView display;

    double val1, val2;

    boolean add, sub, mul, div;
```

- Alternatively, you can use the enum approach in your application.
- You are also free to use any other logic that you would like.

```
public class MainActivity extends AppCompatActivity {
    Button btn0, btn1, btn2, btn3, btn4, btn5, btn6, btn7, btn8, btn9, btnAdd, btnSub, btnDiv,
    btnMul, btnClr, btnEql, btnDec;

    TextView display;

    double val1, val2;

    enum Operator{none, add, minus, multiply, divide}
    Operator optr = Operator.none;
```

Variable initialization

- Still in the MainActivity, initialize the buttons under the onCreate method.
- findViewById() uses the id's you gave to the elements in the layout. Make sure all buttons are given proper id's.

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

    btn0 = findViewById(R.id.btn0);
    btn1 = findViewById(R.id.btn1);
    btn2 = findViewById(R.id.btn2);
    btn3 = findViewById(R.id.btn3);
    btn4 = findViewById(R.id.btn4);
    btn5 = findViewById(R.id.btn5);
```

button.setOnClickListener()

- Write the code for setOnClickListener() for each type of button.
- You want the number to be shown in the TextView, therefore we use setText().
- You also want to be able to enter large values (ex. 123), therefore we use getText() and then add our string.
- An example for "1" is seen below.
- Do this for each calculator button.

```
btn1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        display.setText(display.getText() + "1");
    }
});
```

- Build the setOnClickListener() for each of the different operations (+, -, /, *) and decimal, clear, equal.
- You can use whatever logic you think is best. There is no one answer.

```
btnEql.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {

        //....
```

Expectations

- All group members must participate in completing the lab.
- Only one member needs to submit the application via Brightspace.
- Your application should not crash when performing the expected objectives.
- Mark breakdown:
  - Full marks are awarded if all lab objectives are met
  - Part marks are awarded if some lab objectives are met
  - No marks are awarded if you do not submit the lab or submit past the deadline