Task Core 1 – Spike: Taking Chances

Link to GitHub: https://github.com/SoftDevMobDev-2023-Classrooms/core1-Dank143

Goals:

- Applying listener in the app.
- Applying multiple languages to the app.
- Creating layouts with both linear and constraint layouts.
- Reserving the state of the activities.
- Understanding different states of activities.
- Using logs and IDE for testing and debugging.
- Working with a single activity application.

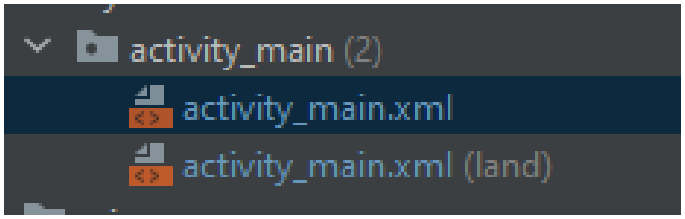
Tools and Resources Used

- Android Studio IDE
- Canvas documentation
- Git and GitHub
- Kotlin programming language
- Testing on Android Explained: https://www.youtube.com/watch?v=gJPclNjOwP8&t=439s

Knowledge Gaps and Solutions

Gap 1: Layouts for landscape and potrait

The default app (using the Empty Views Activity) only has the portrait layout. To create a landscape layout, we will use the design view and click the option for landscape layout, thus creating a new XML file with (land) after its name:



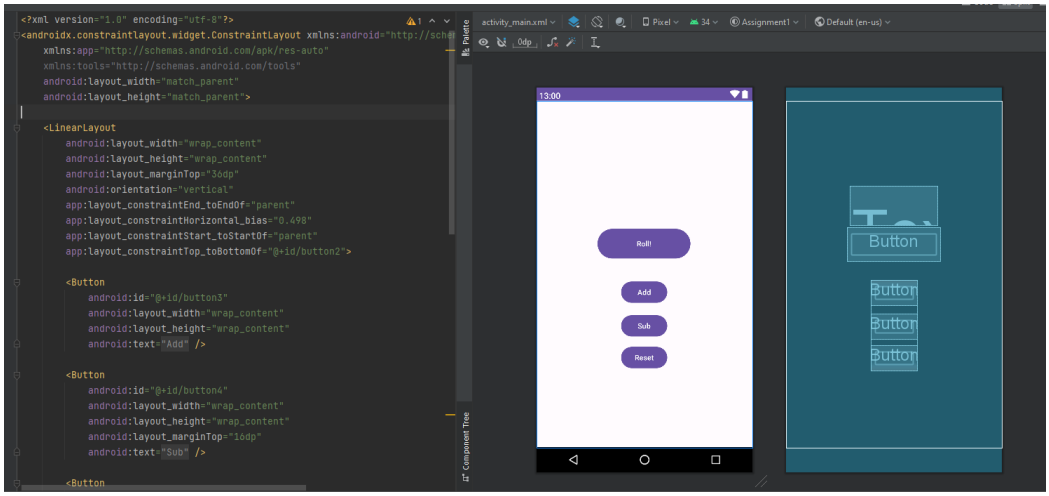
Gap 2: Applying listeners

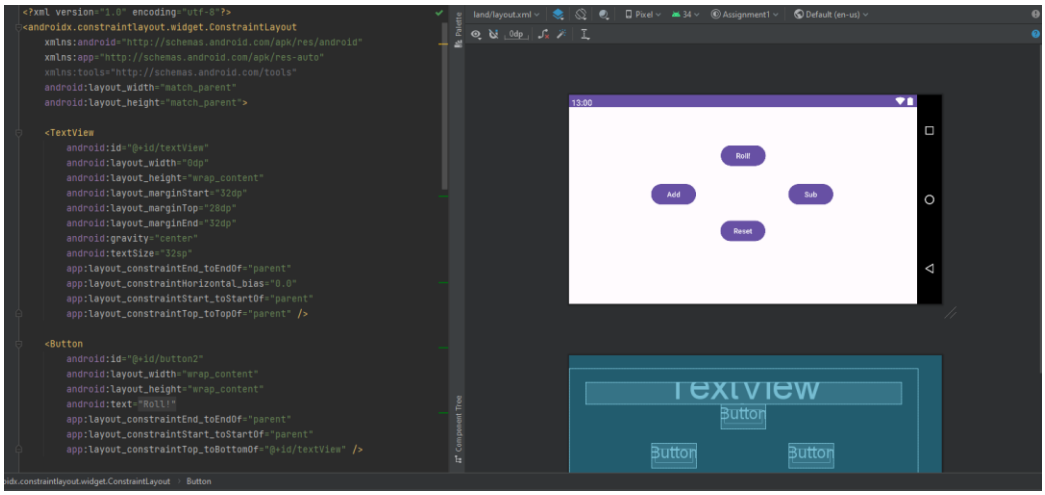
The logic and operations of the program were built around the listeners. It was implemented for the buttons of the program, which will create a sound everytime a button is pressed. The code that implemented listeners is shown below:

```
// Set onClick listeners for buttons
rollButton.setOnClickListener { it: View!
    rollDice()
    setButtonStates()
}

addButton.setOnClickListener { it: View!
    addValue()
    setButtonStates()
}

subButton.setOnClickListener { it: View!
    subtractValue()
    setButtonStates()
}

resetButton.setOnClickListener { it: View!
    resetValue()
}
```

Gap 3: Using both linear and constraint layouts

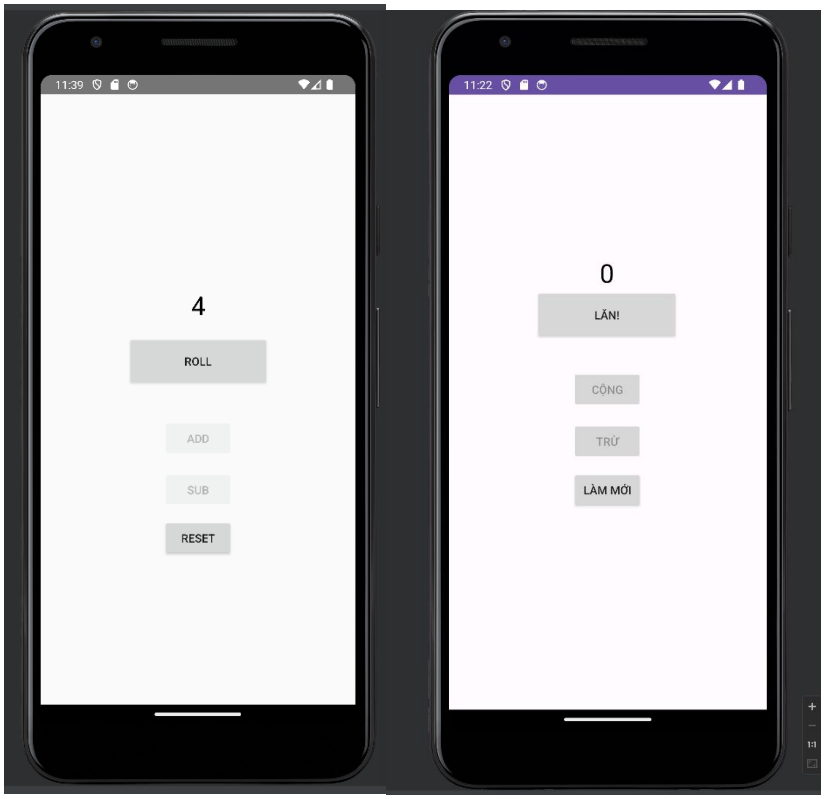
Based on the requirement and the demo video, I have implemented both layouts in the app. While the portrait has both layouts implemented, the landscape version only use the constraint layout, as show in the images below:





Gap 4: Using multiple languages

The app has 2 possible languages: English and Vietnamese. In order to accomplish this, we first create a new value resource file, choose the local qualifier, then select Vietnamese as the language (selecting the region is optional). The default language for this program is English. Only the string value can differ between the string tags, and the two files must be identical. The ids kept in the two strings.xml files, string externalisation enables this capability. The resource file will change whenever the language configuration on the device is altered, making it easier to use and build the application.




```xml
<resources>
    <string name="app_name">Assignment1</string>
    <string name="roll">Roll!</string>
    <string name="textview">TextView</string>
    <string name="add">Add</string>
    <string name="sub">Sub</string>
    <string name="reset">Reset</string>
</resources>
```

```xml
<resources>
    <string name="app_name">Assignment1</string>
    <string name="roll">Lăn!</string>
    <string name="textview">TextView</string>
    <string name="add">Cộng</string>
    <string name="sub">Trừ</string>
    <string name="reset">Làm mới</string>
</resources>
```

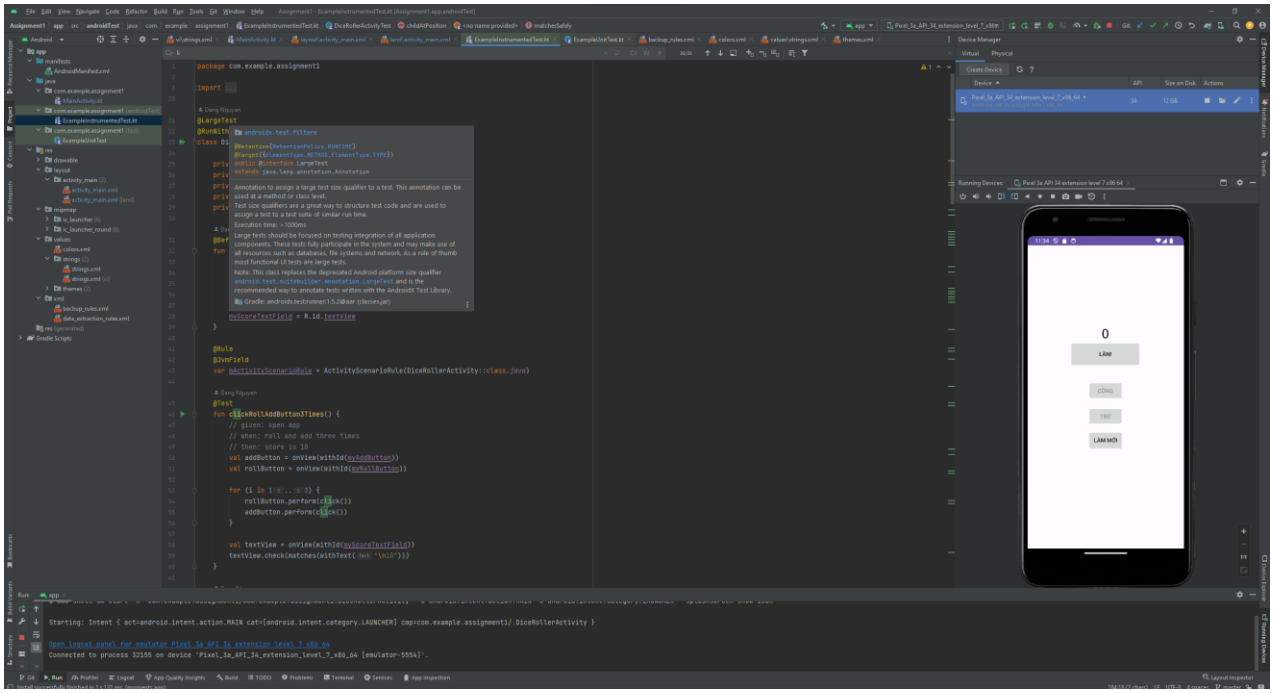
Gap 5: Saving the state of the program

To do this, I use the “saveInstanceState” to save the “currentValue” of the program, which is shown in the picture below:

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
```

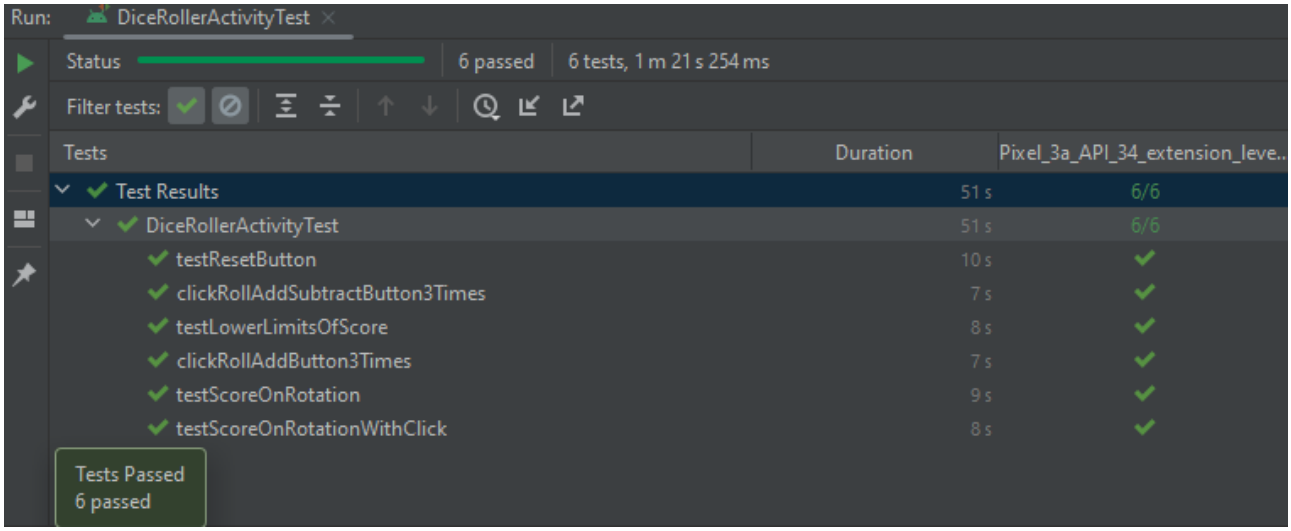
Gap 6: Using the IDE (i.e. Android Studio)

All of the coding, running, debugging and testing are done on Android Studio, which helps me to identify errors and use the given test to make sure the program is running smoothly:



Gap 7: Testing the program

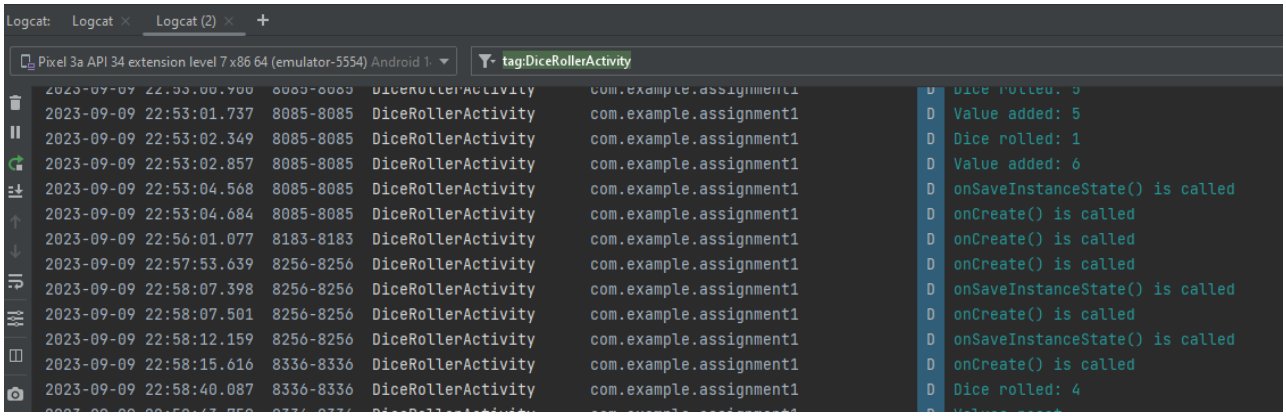
Using the given test by the convenor on GitHub, I have managed to pass all tests which ensure that my app is running as intended.



Gap 8: Using log messages

To see the logs of the program, we override the function for each state of the program's lifecycle and add a log message with a tag (I named this tag “DiceRollerActivity”). To view these changes, we open Logcat as shown in this image below:

```
// Log a message to indicate that onCreate() is called
Log.d( tag: "DiceRollerActivity",  msg: "onCreate() is called")
```



Gap 9: Handling saving of state

The state of this app upon screen rotation is saved using “saveInstanceState” as shown below:

```
override fun onSaveInstanceState(outState: Bundle) {
    super.onSaveInstanceState(outState)
    // Save the current state
    outState.putInt("currentValue", currentValue)
    outState.putInt("diceRoll", diceRoll)
    outState.putBoolean("isRollButtonEnabled", rollButton.isEnabled)
    outState.putBoolean("isAddButtonEnabled", addButton.isEnabled)
    outState.putBoolean("isSubButtonEnabled", subButton.isEnabled)
    outState.putBoolean("isFirstRoll", isFirstRoll) // Save first roll flag
```

Open Issues and Recommendations

Initially, the app was made with both the dice and the current value appeared. This is not what I desire as the demo shown that only the current value (which is 0 at start) is shown before the Roll! button is pressed. So I have changed my code to create a value called “isFirstRoll”

From this

```
class MainActivity : ComponentActivity() {
    private var currentValue = 0
    private var diceRoll = 0
```

To this

```
class DiceRollerActivity : AppCompatActivity() { // Use AppCompatActivity for compatib
    private var currentValue = 0
    private var diceRoll = 0
    private var isFirstRoll = true // Flag to track the first roll
```