CS426 - Week 4 Lab Solution

Lab Title: Designing a Multi-Activity Calculator App

Instructor: Slim Namouchi

Course: Mobile Application Development

Objectives

- Design and implement a multi-activity Android calculator app
- Use intents to navigate between screens and transfer data
- Practice GUI design with XML components (`TextView`, `EditText`, `Button`)
- Handle input validation, error messages, and localization (EN/FR)
- Apply ViewBinding for safer view access

App Overview

Activity	Layout	Purpose
WelcomeActivity	`activity_welcome.xml`	Shows welcome message and start button
CalculatorActivity	`activity_calculator.xml`	Arithmetic operations and input validation
ResultActivity	`activity_result.xml`	Displays calculated result and operation name

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest package="com.must.week_4"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher_background"
        android:label="@string/app_name"
        android:theme="@style/Theme.AppCompat.Light.NoActionBar">

        <activity android:name=".ResultActivity" />
```

```
        <activity android:name=".CalculatorActivity" />
        <activity android:name=".WelcomeActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Layout Files

activity_welcome.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".WelcomeActivity">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <TextView
            android:id="@+id/tvWelcome"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
            android:text="@string/welcome_message"
            android:textSize="22sp"
            android:textStyle="bold"
            android:gravity="center"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"/>

        <Button
            android:id="@+id/btnStart"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="@string/start_button"
            app:layout_constraintTop_toBottomOf="@id/tvWelcome"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="24dp"/>
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_calculator.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    tools:context=".CalculatorActivity">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
```

```
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <EditText
            android:id="@+id/etNum1"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:hint="@string/first_number_hint"
            android:inputType="numberDecimal"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"/>

        <EditText
            android:id="@+id/etNum2"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:hint="@string/second_number_hint"
            android:inputType="numberDecimal"
            app:layout_constraintTop_toBottomOf="@id/etNum1"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="12dp"/>

        <GridLayout
            android:id="@+id/gridOps"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            app:layout_constraintTop_toBottomOf="@id/etNum2"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="20dp"
            android:columnCount="2">

            <Button
                android:id="@+id/btnAdd"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_columnWeight="1"
                android:text="@string/add_button"/>
```

```
            <Button
                android:id="@+id/btnSub"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_columnWeight="1"
                android:text="@string/subtract_button"/>

            <Button
                android:id="@+id/btnMul"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_columnWeight="1"
                android:text="@string/multiply_button"/>

            <Button
                android:id="@+id/btnDiv"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_columnWeight="1"
                android:text="@string/divide_button"/>
        </GridLayout>

        <Button
            android:id="@+id/btnClear"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/clear_button"
            app:layout_constraintTop_toBottomOf="@id/gridOps"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="16dp"/>

    </androidx.constraintlayout.widget.ConstraintLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_result.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    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"
    android:padding="16dp"
    tools:context=".ResultActivity">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">

        <TextView
            android:id="@+id/tvOperation"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/result_label"
            android:textStyle="bold"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"/>

        <TextView
            android:id="@+id/tvExpression"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            app:layout_constraintTop_toBottomOf="@id/tvOperation"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="8dp"/>

        <TextView
            android:id="@+id/tvResult"
            android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:text="0"
            android:textSize="28sp"
            android:textStyle="bold"
            app:layout_constraintTop_toBottomOf="@id/tvExpression"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="12dp"/>

        <Button
            android:id="@+id/btnBack"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/back_button"
            app:layout_constraintTop_toBottomOf="@id/tvResult"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="20dp"/>

    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

Kotlin Source Files

WelcomeActivity.kt

```
class WelcomeActivity : AppCompatActivity() {
    private lateinit var binding: ActivityWelcomeBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityWelcomeBinding.inflate(layoutInflater)
        setContentView(binding.root)

        binding.btnStart.setOnClickListener {
            startActivity(Intent(this, CalculatorActivity::class.java))
```

```
        }
    }
}
```

CalculatorActivity.kt

```
import android.content.Intent
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity


class CalculatorActivity : AppCompatActivity() {
    private lateinit var binding: ActivityCalculatorBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityCalculatorBinding.inflate(layoutInflater)
        setContentView(binding.root)

        binding.btnAdd.setOnClickListener { handleOperation("+") }
        binding.btnSub.setOnClickListener { handleOperation("-") }
        binding.btnMul.setOnClickListener { handleOperation("*") }
        binding.btnDiv.setOnClickListener { handleOperation("/") }

        binding.btnClear.setOnClickListener {
            binding.etNum1.text?.clear()
            binding.etNum2.text?.clear()
        }
    }

    private fun handleOperation(op: String) {
        val num1 = binding.etNum1.text.toString().toDoubleOrNull()
        val num2 = binding.etNum2.text.toString().toDoubleOrNull()

        if (num1 == null || num2 == null) {
            Toast.makeText(this, getString(R.string.error_enter_numbers),
```

```
Toast.LENGTH_SHORT).show()
            return
        }

        if (op == "/" && num2 == 0.0) {
            Toast.makeText(this, getString(R.string.error_divide_zero),
Toast.LENGTH_SHORT).show()
            return
        }

        val result = when (op) {
            "+" -> num1 + num2
            "-" -> num1 - num2
            "*" -> num1 * num2
            "/" -> num1 / num2
            else -> 0.0
        }

        val intent = Intent(this, ResultActivity::class.java).apply {
            putExtra("RESULT", result)
            putExtra("OPERATION", op)
            putExtra("NUM1", num1)
            putExtra("NUM2", num2)
        }
        startActivity(intent)
    }
}
```

ResultActivity.kt

```
package com.must.week_4

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

class ResultActivity : AppCompatActivity() {
    private lateinit var binding: ActivityResultBinding
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityResultBinding.inflate(layoutInflater)
        setContentView(binding.root)

        val result = intent.getDoubleExtra("RESULT", Double.NaN)
        val op = intent.getStringExtra("OPERATION") ?: ""
        val num1 = intent.getDoubleExtra("NUM1", Double.NaN)
        val num2 = intent.getDoubleExtra("NUM2", Double.NaN)

        val opName = when (op) {
            "+" -> getString(R.string.addition)
            "-" -> getString(R.string.subtraction)
            "*" -> getString(R.string.multiplication)
            "/" -> getString(R.string.division)
            else -> ""
        }

        binding.tvOperation.text = opName
        binding.tvResult.text = result.toString()
        binding.tvExpression.text = "$num1 $op $num2"

        binding.btnBack.setOnClickListener { finish() }
    }
}
```

Localization Files

English - `values/strings.xml`

```
<resources>
    <string name="app_name">Calculator App</string>
    <string name="welcome_message">Welcome to Calculator App</string>
    <string name="start_button">Start</string>
    <string name="first_number_hint">First number</string>
    <string name="second_number_hint">Second number</string>
```

```
    <string name="add_button">+</string>
    <string name="subtract_button">-</string>
    <string name="multiply_button">×</string>
    <string name="divide_button">÷</string>
    <string name="result_label">Result</string>
    <string name="back_button">Back</string>
    <string name="error_enter_numbers">Please enter valid numbers</string>
    <string name="error_divide_zero">Cannot divide by zero</string>
    <string name="addition">Addition</string>
    <string name="subtraction">Subtraction</string>
    <string name="multiplication">Multiplication</string>
    <string name="division">Division</string>
    <string name="clear_button">Clear</string>
</resources>
```

French - values-fr/strings.xml

```
<resources>
    <string name="app_name">Application Calculatrice</string>
    <string name="welcome_message">Bienvenue dans l'application Calculatrice</string>
    <string name="start_button">Démarrer</string>
    <string name="first_number_hint">Premier nombre</string>
    <string name="second_number_hint">Deuxième nombre</string>
    <string name="add_button">+</string>
    <string name="subtract_button">-</string>
    <string name="multiply_button">×</string>
    <string name="divide_button">÷</string>
    <string name="result_label">Résultat</string>
    <string name="back_button">Retour</string>
    <string name="error_enter_numbers">Veuillez entrer des nombres valides</string>
    <string name="error_divide_zero">Impossible de diviser par zéro</string>
    <string name="addition">Addition</string>
    <string name="subtraction">Soustraction</string>
    <string name="multiplication">Multiplication</string>
    <string name="division">Division</string>
    <string name="clear_button">Effacer</string>
</resources>
```

Expected Output

1. **Welcome Screen** - displays greeting and “Start” button
2. **Calculator Screen** - input numbers and operations
3. **Result Screen** - expression with result and “Back” button

Full Code Explanation and Tips for Multi-Activity Calculator App

This document explains the complete code of each file from the Week 4 lab. It includes **design insights**, **code purpose**, and **useful tips** for students to deepen their understanding.

1. WelcomeActivity and activity_welcome.xml

Goal:

Create a simple welcome screen with a message and a button to start the calculator.

Layout (activity_welcome.xml):

- Uses `ConstraintLayout` for flexible positioning.
- Contains a centered bold `TextView` displaying a welcome message.
- A wide `Button` below the message labeled "Start".

Kotlin Code (WelcomeActivity.kt):

```
binding.btnStart.setOnClickListener {
    startActivity(Intent(this, CalculatorActivity::class.java))
}
```

- When the Start button is clicked, it launches the CalculatorActivity using an explicit `Intent`.

Tips:

- Using `ConstraintLayout` is efficient for most layouts, enabling responsive design for various screen sizes.
- ViewBinding (`ActivityWelcomeBinding`) is a safe and clean way to access views.
- Declaring WelcomeActivity as the launcher activity in the manifest makes it the entry point.

2. CalculatorActivity and activity_calculator.xml

Goal:

Build a screen to input two decimal numbers and perform basic arithmetic operations.

Layout (activity_calculator.xml):

- Two `EditText` inputs (`etNum1` and `etNum2`) with decimal input type and hint text.
- A `GridLayout` with 4 buttons for Add (+), Subtract (-), Multiply (×), and Divide (÷).
- A Clear button that resets both input fields.

Kotlin Code (CalculatorActivity.kt):

Core functionalities:

- **Input retrieval and validation:**

```
val num1 = binding.etNum1.text.toString().toDoubleOrNull()
val num2 = binding.etNum2.text.toString().toDoubleOrNull()

if (num1 == null || num2 == null) {
    Toast.makeText(this, getString(R.string.error_enter_numbers),
Toast.LENGTH_SHORT).show()
    return
}
```

Tip: Using `toDoubleOrNull()` safely parses the input without crashing if the format is invalid.

- **Handling division by zero:**

```
if (op == "/" && num2 == 0.0) {
    Toast.makeText(this, getString(R.string.error_divide_zero),
Toast.LENGTH_SHORT).show()
    return
}
```

- **Performing the calculation:**

```
val result = when (op) {
    "+" -> num1 + num2
    "-" -> num1 - num2
    "*" -> num1 * num2
    "/" -> num1 / num2
    else -> 0.0
}
```

- **Passing data via Intent extras:**

```
val intent = Intent(this, ResultActivity::class.java).apply {
    putExtra("RESULT", result)
    putExtra("OPERATION", op)
    putExtra("NUM1", num1)
    putExtra("NUM2", num2)
}
startActivity(intent)
```

- **Clear button:** clears both `EditText` fields.

Tips:

- Use `Toast` for lightweight user feedback.
- Validate inputs before calculations to prevent runtime errors.
- Clear button improves usability and testing.
- Using a `GridLayout` for operation buttons aligns them neatly and adapts to screen sizes.

3. ResultActivity and activity_result.xml

Goal:

Display the chosen operation, operands, and result with a back button.

Layout (activity_result.xml):

- A bold label `tvOperation` showing the name of the operation (Addition, etc.).

- `tvExpression` showing the math expression like "3.0 + 4.0".
- Large, bold `tvResult` displaying the numerical result.
- `btnBack` to return to CalculatorActivity.

Kotlin Code (ResultActivity.kt):

- Extract data from intent extras using

```
val result = intent.getDoubleExtra("RESULT", Double.NaN)
val op = intent.getStringExtra("OPERATION") ?: ""
val num1 = intent.getDoubleExtra("NUM1", Double.NaN)
val num2 = intent.getDoubleExtra("NUM2", Double.NaN)
```

- Map operator symbols to localized operation names:

```
val opName = when (op) {
    "+" -> getString(R.string.addition)
    "-" -> getString(R.string.subtraction)
    "*" -> getString(R.string.multiplication)
    "/" -> getString(R.string.division)
    else -> ""
}
```

- Populate the UI elements:

```
binding.tvOperation.text = opName
binding.tvResult.text = result.toString()
binding.tvExpression.text = "$num1 $op $num2"
```

- Set back button:

```
binding.btnBack.setOnClickListener { finish() }
```

Tip: Calling `finish()` closes the ResultActivity, returning to CalculatorActivity without a new instance.

Tips:

- Always provide a default value or fallback when reading extras.
- Use localized strings for user-visible text to support multiple languages.

- Display clear expressions to enhance user understanding.

4. Localization (`strings.xml` and `strings-fr.xml`)

- All user-visible strings are stored in string resource files.
- English (`values`) and French (`values-fr`) versions allow automatic switching based on device language.

Tips:

- Never hard-code strings in your layouts or code to support internationalization.
- Always use string keys consistently across languages.

5. General Architecture and Best Practices

- **ViewBinding:** It reduces boilerplate and safer than `findViewById()`.
- **Intents:** Used for explicit activity navigation and data passage.
- **Error Handling:** Always validate user input and provide helpful feedback with `Toast` or `setError()`.
- **Layout:** Using `ConstraintLayout` and `GridLayout` helps in responsive UI design.
- **Back Navigation:** Use `finish()` to return to the previous screen without creating multiple activity instances.
- **Testing:** Test inputs thoroughly, including edge cases like empty, invalid numbers, or divide by zero.

Summary

This lab guides you through fundamental Android topics:

- Multi-activity apps
- Intent-based communication
- UI design and layout management
- Input validation and error messages
- Supporting multiple languages