CS 426 – Mobile Application Development

Instructor: Slim NAMOUCHI

Week 4 Lab: Designing a Multi-Activity Calculator App

Objectives

- Design and implement a simple Android calculator app with three activities.
- Practice UI design using TextView, EditText, Buttons, and layouts.
- Use Intent-based communication to pass data between activities.
- Handle input validation and user feedback gracefully.
- Utilize string resources for English and French localization.
- Implement View Binding for safer UI code interactions.

App Overview

Create an app with three screens:

1. Welcome Screen (Activity 1)

- Displays a welcome message from string resources.
- Has a button to navigate to the Calculator screen.

2. Calculator Screen (Activity 2)

- Contains two EditText fields to enter numbers.
- Four buttons for arithmetic operations: +, -, ×, ÷.
- Performs the selected operation and sends the result and operation name to the Result screen.

3. Result Screen (Activity 3)

- Displays the operation performed and the calculation result.
- Includes a button to return to the Calculator screen.

Screen Details and Code Hints

Activity 1: Welcome Screen

- Use a TextView for the welcome message, referring to `@string/welcome_message`.
- Add a Button with text from `@string/start_button`.
- On button click, start CalculatorActivity explicitly.

Example:

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

Activity 2: Calculator Screen

- Two EditText fields with hints from `@string/first_number_hint` and `@string/second_number_hint`.
- Four buttons labeled from string resources for +, -, ×, ÷.
- On clicking any operation button:
    - Validate both inputs are numeric.
    - Calculate using `when` for the selected operation.
    - Pass data to ResultActivity via intents.

Example operation handler snippet:

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

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

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


val intent = Intent(this, ResultActivity::class.java)
< pass also the numbers>
intent.putExtra("OPERATION", operationName)
startActivity(intent)
```

Activity 3: Result Screen

- Show operation and result in two separate TextViews.
- Button to finish this activity and go back to Calculator.

Example snippet:

```
val operation = intent.getStringExtra("OPERATION") ?: ""

tvOperation.text = operation
tvResult.text = result.toString()

btnBack.setOnClickListener {
    finish()
}


Exemple to help you :
val (result, operationName) = when (operator) {
    "+" -> Pair(num1 + num2, getString(R.string.addition))
    "-" -> Pair(num1 - num2, getString(R.string.subtraction))
    "*" -> Pair(num1 * num2, getString(R.string.multiplication))
    "/" -> {
        if (num2 == 0.0) {
            Toast.makeText(this, getString(R.string.error_divide_zero),
Toast.LENGTH_SHORT).show()
            return
```

```
        }
        Pair(num1 / num2, getString(R.string.division))
    }
    else -> {
        Toast.makeText(this, "Unknown operator", Toast.LENGTH_SHORT).show()
        return
    }
}
```

Localization: String Resources

English (res/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>
</resources>
```

French (res/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>
</resources>
```

Best Practices

- Enable View Binding and use it for all view accesses.
- Always validate user inputs and provide clear feedback.
- Localize all user-facing strings.
- Structure your code cleanly and clearly.

Extensions to Explore

- Display input error inline with EditText using `setError`.
- Clear inputs with an extra button.

Submission Instructions

- Zip and upload your project before the deadline.
- Include screenshots of all three activities functioning correctly.
- Ensure all coding standards and app requirements are met.