Here's a simple Flutter app that achieves what you described:

Dart

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatefulWidget {

@override

State<MyApp> createState() => \_MyAppState();

}

class \_MyAppState extends State<MyApp> {

// Text editing controllers for the textboxes

final firstNumberController = TextEditingController();

final secondNumberController = TextEditingController();

// Variable to store the sum

double sum = 0.0;

// Function to calculate the sum and update the textview

void calculateSum() {

double firstNumber = double.tryParse(firstNumberController.text) ?? 0.0;

double secondNumber = double.tryParse(secondNumberController.text) ?? 0.0;

setState(() {

sum = firstNumber + secondNumber;

});

}

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: const Text('Sum Calculator'),

),

body: Padding(

padding: const EdgeInsets.all(20.0),

child: Column(

children: [

TextField(

controller: firstNumberController,

keyboardType: TextInputType.number,

decoration: const InputDecoration(

labelText: 'First Number',

),

),

const SizedBox(height: 10.0),

TextField(

controller: secondNumberController,

keyboardType: TextInputType.number,

decoration: const InputDecoration(

labelText: 'Second Number',

),

),

const SizedBox(height: 20.0),

ElevatedButton(

onPressed: calculateSum,

child: const Text('Calculate'),

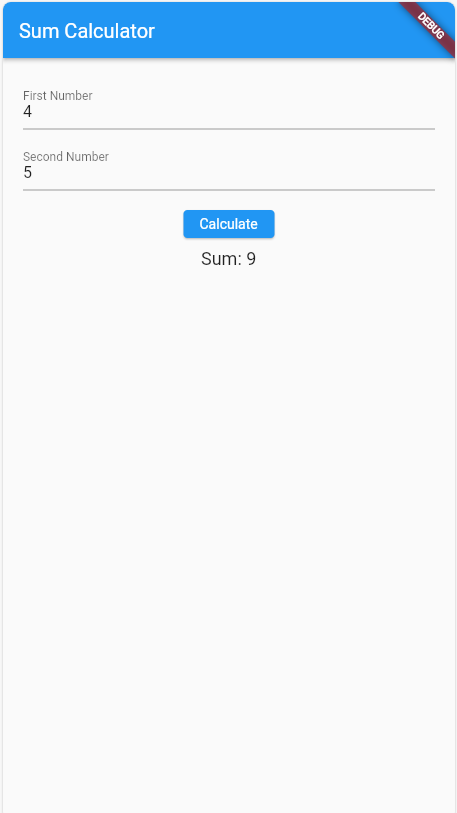
),

const SizedBox(height: 10.0),

Text(

'Sum: $sum',

style: const TextStyle(fontSize: 18.0),

 ),

],

),

),

),

);

}

}

This code defines a MyApp class with a stateful widget \_MyAppState. It uses two TextEditingController objects to manage the text input from the user in the textboxes.

The calculateSum function retrieves the text from the controllers, converts them to doubles (handling potential parsing errors), calculates the sum, and updates the sum variable using setState. This triggers a rebuild of the widget tree, displaying the updated sum in the Text widget.

Remember to save this code in a .dart file (e.g., main.dart) and run flutter run to launch the app on your device or emulator.

Okay, let's break down the Flutter code step-by-step for beginners:

**1. Importing Flutter:**

Dart

import 'package:flutter/material.dart';

This line imports the Flutter library, which provides all the building blocks to create a Flutter app. You'll need this line in almost every Flutter project.

**2. The main Function:**

Dart

void main() {

runApp(MyApp());

}

This is the entry point of your application. The main function is the first function that gets called when you run the app. Here, it calls the runApp function with an instance of MyApp class, essentially telling Flutter to launch this app.

**3. The MyApp Class:**

* MyApp extends StatefulWidget: This line defines a class named MyApp that inherits from StatefulWidget. This means the app's state can change over time, which is needed to update the sum when the user clicks "Calculate".

**4. State Management:**

* \_MyAppState extends State<MyApp>: This defines a private class named \_MyAppState that inherits from State<MyApp>. This connects the MyApp widget with its state (\_MyAppState).

**5. Text Editing Controllers:**

Dart

final firstNumberController = TextEditingController();

final secondNumberController = TextEditingController();

These lines create two TextEditingController objects. They manage the text input from the user in the two textboxes. Whenever the user types something, these controllers store the text.

**6. Variable for Sum:**

Dart

double sum = 0.0;

This line declares a variable named sum of type double and initializes it to 0.0. This will store the calculated sum of the two numbers.

**7. calculateSum Function:**

Dart

void calculateSum() {

double firstNumber = double.tryParse(firstNumberController.text) ?? 0.0;

double secondNumber = double.tryParse(secondNumberController.text) ?? 0.0;

setState(() {

sum = firstNumber + secondNumber;

});

}

This function is responsible for calculating the sum and updating the UI.

* double.tryParse(firstNumberController.text): This line tries to convert the text from the firstNumberController to a double value. If it fails (e.g., non-numeric characters), it returns null. The nullish coalescing operator (??) ensures we have a valid number by assigning 0.0 in case of parsing errors. Similarly for secondNumber.
* setState(() { ... }): This line wraps the code that modifies the state of the app (in this case, updating the sum variable). Whenever you change the state within a stateful widget, you need to use setState to notify Flutter that the UI needs to be rebuilt with the updated information.

**8. Building the UI (build method):**

Dart

@override

Widget build(BuildContext context) {

// ... rest of the code defining the UI elements ...

}

The build method is responsible for building the UI of the app. It returns a Widget object that represents the entire user interface.

**9. Scaffold - The Main Layout:**

Dart

return MaterialApp(

home: Scaffold(

// ... app elements go here ...

),

);

This part defines the main layout of the app using a Scaffold widget. It provides a basic app structure with an optional appBar (top bar) and a body which contains the main content of the screen.

**10. UI Elements:**

* AppBar: This creates the top bar of the app with the title "Sum Calculator".
* Padding: This adds some padding around the content within the body.
* Column: This arranges the UI elements vertically within the body.
* TextField: These widgets represent the two textboxes for user input. They use the firstNumberController and secondNumberController for managing the text.
  + keyboardType: TextInputType.number: This specifies that the keyboard should be optimized for numerical input when the user focuses on these textboxes.
* ElevatedButton: This creates the "Calculate" button that triggers the calculateSum function when clicked.
* Text: This displays the calculated sum with a label "Sum:".

**11. Running the App:**

After saving this code in a .dart file (e.g., main.dart), you can run the app using the command flutter run in your terminal. This