Mobile Development :

6 : Flutter for Mobile Development : Part 1

Dart, Flutter, Widget, Basic Interactivity

Professor Imed Bouchrika
National School of Artificial Intelligence
imed.bouchrika@ensia.edu.dz

Outline :

- Section 1:
  - Introduction to Flutter
  - Dart Language
  - Hello World ! in Flutter
  - Flutter Project Structure
  - Widgets for Building UIs
  - Stateless and Stateful Widgets
- Section 2 :
  - Incrementer App
  - To-Do App (No DB)

Section 1

Introduction to Flutter

Introduction to Flutter

- **What's Flutter**
  - Flutter is an open-source software development framework which uses the Dart Programming language
  - It is designed to build natively compiled applications for mobile (Both android and iOS), web, and desktop from a single codebase.
  - Flutter was first released in 2017 by Google. It has gained significant popularity in the developer community due to its versatility and ease of use.

Introduction to Flutter

- **Why use Flutter instead of Kotlin/Java**
  - Single Codebase for Multiplatforms
  - High Performance due to compiling to native code
  - Fast Development with hot reload where changes can be applied “instantly” to running apps without losing data or states.
  - Support for Material design (Android theme) and Cupertino (iOS)
  - Beautiful Interfaces and Rich Library of Widgets
  - Cross-Platform Development where you can develop on various platforms for various platforms including desktop or even web and potentially backends.
  - Backed and Supported by Google with Strong Community Support

Introduction to Flutter

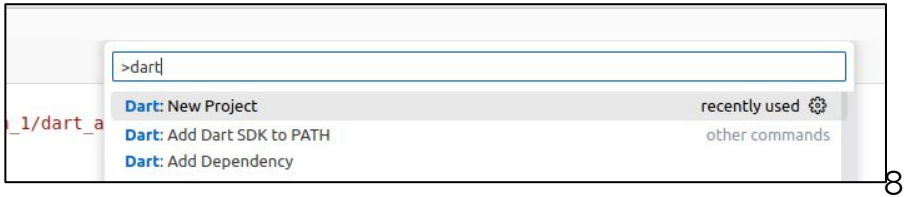
- **Popular Apps using made using Flutter**
  - Google Classroom
  - Google Pay
  - eBay
  - PUBG Mobile
  - Toyota App
  - Crédit Agricole
  - Including apps by ByteDance, Tencent…

Dart Language

- **Dart as a Programming Language**
  - Client-optimized, object-oriented programming for creating apps quickly for many platforms including Android, iOS, Linux, Windows and the web.
  - Developed by Google in 2011 to replace JavaScript, but without success.
  - Later, It gained huge popularity because it is the main language for Flutter
  - Statically typed language where data types of variables must be specified either explicitly or implicitly through inference.
  - Supports modern programming features as null safety checks and asynchronous prog.
  - Lastly, it is free and open-source + Rich standard library.

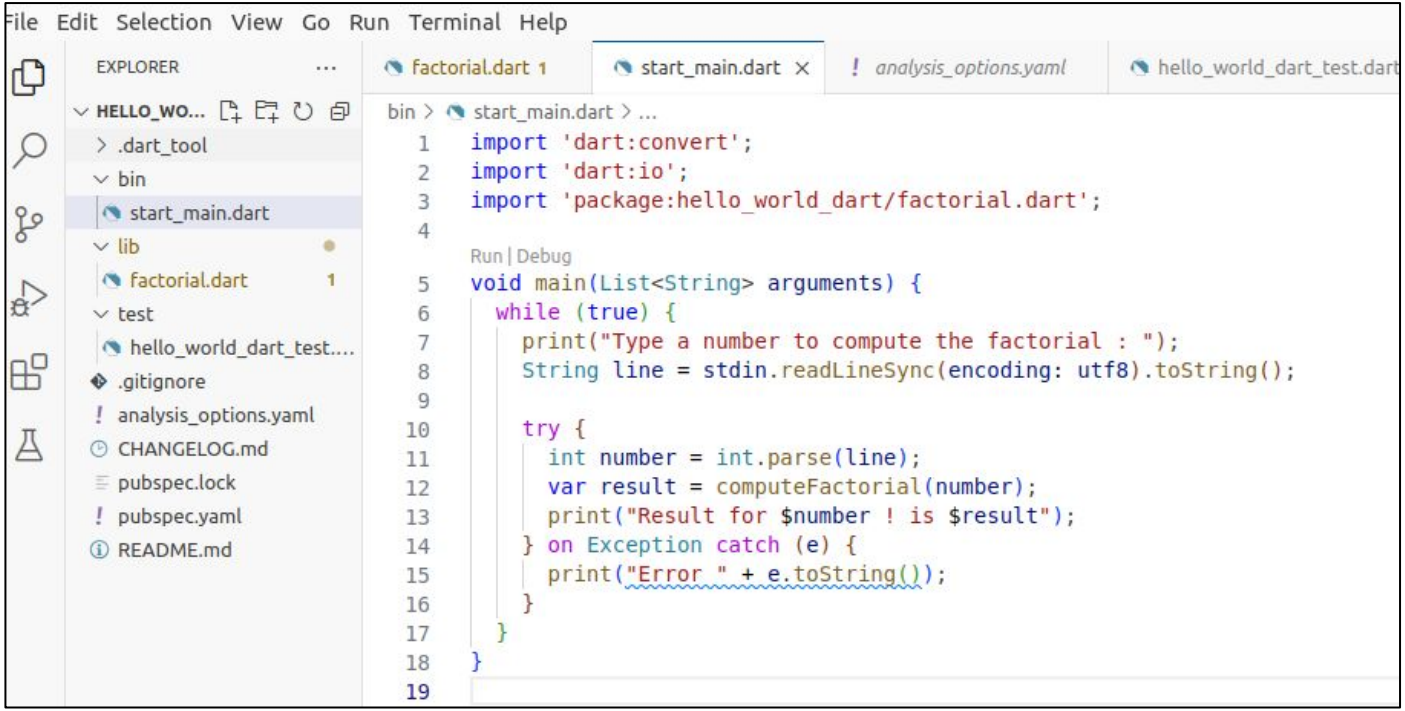
Dart Language

- **Getting Started with Dart**
  - You can start to learn and test dart online at :
    - https://dartpad.dev/?
  - Install VS + Android SDK + Dart + Flutter on your Machine
  - To create new Project, Either :
    - Ctrl + Shift + P and type “dart” → Choose Dart: **New Project**
      - Choose Console App
    - From Console:
      - **dart create project_name**



Dart Language

- **Hello World in Dart**



Dart Language

- **Hello World in Dart**

Edit Selection View Go Run Terminal Help

EXPLORER

HELLO_WORLD_DART
- .dart_tool
- bin
  - start_main.dart
- lib
  - factorial.dart 1
- test
  - hello_world_dart_test....
- .gitignore
- analysis_options.yaml
- CHANGELOG.md
- pubspec.lock
- pubspec.yaml
- README.md

factorial.dart 1 | start_main.dart | analysis_options.yaml

lib > factorial.dart > ...

```dart
import 'dart:io';

int computeFactorial(int a) {
  if (a == 1) {
    return 1;
  }
  return a * computeFactorial(a - 1);
}
```

Dart Language

- **Hello World in Dart**
  - Usually, you can hit the Run button to execute, but **prefer** to use the console with the command :
    - **dart run OR dart bin/start_main.dart**

```
TERMINAL
TERMINAL
imed@imed-Inspiron:~/Dropbox/Workspace/Teaching/ENSIA/MobileDev/MyLectures/W6/hello_world_dart$ dart bin/start_main.dart
Type a number to compute the factorial :
5
Result for 5 ! is 120
Type a number to compute the factorial :
w
Error FormatException: Invalid radix-10 number (at character 1)
w
^

Type a number to compute the factorial :
```

Dart Language

```
var myName = "Imed";
String sonName;
String DaughterName = "Amel";
int age, size = 0, volume = 1; // Not
recommended
bool isHappy = true;
const double pi = 3.14;
num quantity = 2.1;
```

- **Syntax of Dart : Variables**
  - Declaring a variable can be set as
    - **Type name = value ;** :
  - **num** is to refer to both int and double
  - **var** can used to say **any** type
    - Can we assign a **var** to **int** and later **string** ?
  - Const to create **immutable variables** (unchangeable)
  - Naming convention : **lowerCamelCase**

Dart Language

```
var myName = "Imed";
String sonName;
```

```
1 void main() {
2   var a=1;
3    a="sss";
4    print(a);
5  }
6
```

Run

Console

```
Error compiling to JavaScript:
lib/main.dart:3:5:
Error: A value of type 'String' can't be assigned to a variable of type
  a="sss";
    ^
Error: Compilation failed.
```

- Can we assign a **var** to **int** and later **string** ?

- Const to create **immutable variables** (unchangeable)
- Naming convention : **lowerCamelCase**

Dart Language

```
var myName = "Imed";
String sonName;
String DaughterName = "Amel";
int age, size = 0, volume = 1; // Not
recommended
```

- **Syntax of Dart : Variables**
  - Declaring a variable can be set as
    - **Type name = value ;** :
  - **num** is to refer to both int and double
  - **var** can used to say **any** type
    - Can we assign a **var** to **int** and late
  - IF you are forced to use dynamically typed variables (which is not recommended), you can use :
    - **dynamic a=1 ;**

```
1 void main() {
2   dynamic  a=1;
3    a="sss";
4    print(a);
5  }
6
```

Run

Console

sss

Dart Language

- **Syntax of Dart : Data Types and Converting**
  - String
    - You can use the $ sign inside a string for injecting variables into a string.
      - print("Hello $name, my age is $age")
    - You can use """ to create a multi-line string"""
  - String to int : int myAge=**int.parse**(strValue)
  - int to String : String ageStr = myAge.toString()
  - String to doube : double myWeight=**double.parse**(strValue)

Dart Language

- **Syntax of Dart : Collections : List**
  - List are like Arrays in Java defined as :
    - Growable List : List<int> ages= [10, 21 ,31]
    - Fixed Length List : var ages=List<int>.filled(5,1) // Fills default val of 1
  - Accessing :
    - ages[INDEX]
    - Or even get inversely the index by using : ages.**indexOf**(31)
  - Methods
    - **Ages.length** : to get the number of elements in the list

Dart Language

- **Syntax of Dart : Collections : List**
  - Methods
    - **ages.length** : to get the number of elements in the list
    - **add (item) :** to add an element
    - **insert(index, item)** : to add a specified index and shifting other elements
    - **remove(item)** : remove an item by value (only one item removed at a time)
    - **removeAt(index)** : remove element at a given index.
  - Properties of the List:
    - **first , last, isEmpty , isNotEmpty, reversed, single**
  - Combining two list using **spread syntax**
    - var bigList =[**...listOne, ...listTwo**] ;

Dart Language

- **Syntax of Dart : Collections : Set**
  - Set is a unique collection of items. No duplicates (in contrast to a list)

```
Set<String> fruits = {"Apple", "Orange", "Mango"};
fruits.add("Grapes");
fruits.remove("Orange");
String firstElement = fruits.elementAt(0);
bool hasMango = fruits.contains("Mango");
int fruitsCount = fruits.length ;
```

Dart Language

{key:val }

- **Syntax of Dart : Collections : Map**

```
Map<String, String> countryCapital = {
      'USA': 'Nothing',
      'India': 'New Delhi',
      'China': 'Beijing'
};
countryCapital['Algeria'] = 'Algiers';
bool checkOne = countryCapital.containsKey("India");
bool checkTwo = countryCapital.containsValue("India");
print("All keys of Map: ${countryCapital.keys}");
print("All values of Map: ${countryCapital.values}");
print("Is Map empty: ${countryCapital.isEmpty}");
print("Is Map not empty: ${countryCapital.isNotEmpty}");
print("Length of map is: ${countryCapital.length}");
```

Dart Language

- **Syntax of Dart : Operations**
  - Arithmetic Operators:
    - % Modulus
    - ~/ Integer division (Divide two numbers and give output as int)
  - Logical Operators
    - && and
    - || or
    - ! not
  - Type Test Operators
    - is (myValue is **int)**
    - is! (myValue **is!** String)

Dart Language

- **Syntax of Dart : Flow Control : IF-ELSE + SWITCH**

```
var noOfMonth=1;
if (noOfMonth == 1) {
     print("The month is jan");
} else if (noOfMonth == 2) {
     print("The month is feb");
} else {
     print("Invalid option given.");
}
```

```
var dayOfWeek = 5;
switch (dayOfWeek) {
      case 1:{
            print("Day is Sunday.");
            break;
      }
      case 2:{
            print("Day is Monday.");
            break;
      }
      Default:{
            print("Invalid Weekday.");
            break;
      }
}
```

Dart Language

- **Syntax of Dart : Flow Control : For + While**

```
for (int i = 1; i <= 10; i++) {
     print(i);
}
```

```
List<int> numbers = [1,2,3,4,5];
int total = 0;
numbers.forEach( (num)=>total= total+ num);
```

```
List<int> numbers = [1,2,3,4,5];
for (int a in numbers){
print(a);
}
```

```
int i = 1 ;
while (i <= 10) {
     print(i);
     i=i+1 ;
}
```

Dart Language

- **Syntax of Dart : Functions**

```
//function to add
int add(int a, int b) {
    var total;
    total = a + b;
    return total;
}
```

Dart Language

- **Syntax of Dart : Asynchronous Programming**
  - Asynchronous Programming is a way of writing code that allows a program to do multiple tasks at the same time.
  - **Future** represents a value that is not yet available and may be available later.
  - **async** : is employed to define a function that can perform asynchronous operations .
  - **await** : is used within async's body to pause execution while waiting for the operations to be completed.
  -

Dart Language

- S

```
void main() {
      print("Starting");
      getData();
      print("End.");
}

void getData() async{
      String data = await processLongRequest();
      print(data);
}

Future<String> processLongRequest(){
      return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

rogram to do

able later.

Dart Language

```
void main() {
```

```dart
void main() {
  print("Starting");
  getData();
  print("End.");
}

void getData() async{
  String data = await processLongRequest();
  print(data);
}

Future<String> processLongRequest(){
  return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

Run

Console

```
Starting
End.
Getting Data
```

```
    return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

Dart Language

```
void main() {
      print("Starting");
      Future<String> data=getData();
      print("Got data "+data.toString());
      print("End.");
}

Future<String> getData() async{
      String data = await processLongTask();
      print(data);
      return data;
}

Future<String> processLongTask(){
      return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

program to do

lable later.

Dart Language

```dart
void main() {
  print("Starting");
  Future<String> data=getData();
  print("Got data "+data.toString());
  print("End.");

}

Future<String> getData() async{
  String data = await processLongTask();
  print(data);
  return data;
}

Future<String> processLongTask(){
  return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

Run

Console

```
Starting
Got data Instance of '_Future<String>'
End.
Getting Data
```

```dart
Future<String> processLongTask(){
      return Future.delayed( Duration(seconds:5) , ()=> "Getting Data");
}
```

Dart Lan

- **Syntax of Dart :**

  Object Oriented Programming

```
class Student extends Person {
      String? name;
      int? age;
      int? rollNumber;
      static int? totalNumber;
      // Constructor
      Student(String name, int age, int rollNumber) {
            this.name = name;
            this.age = age;
            this.rollNumber = rollNumber;
      }
      void show() {
            super.show(); // Calling the show method of the parent
            class
            print("local method is called");
      }
}
void main() {
      Student student = Student("John", 20, 1);
      print("Name: ${student.name}");
      print("Age: ${student.age}");
      print("Roll Number: ${student.rollNumber}");
}
```

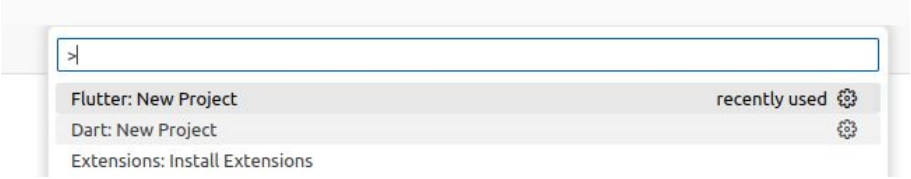
Dart Language

- **Syntax of Dart : Exception Handling**

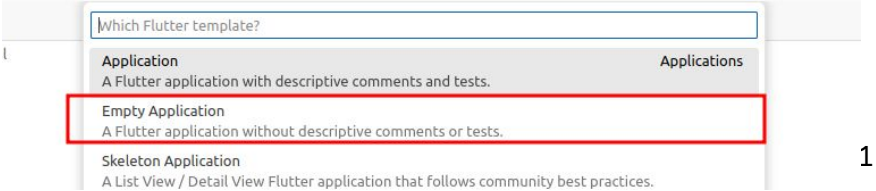
```
try {
      int number = int.parse(line);
      var result = computeFactorial(number);
      print("Result for $number ! is $result");
} catch (e, stacktrace) {
      print("Error ${e.toString()} stackTrace $stacktrace ");
}
```

Hello World ! in Flutter

- **Creating a Simple Hello World !**
  - Create a new Project
    - Ctrl + Shift + P : Type **flutter and choose : New Flutter Project**



    - Choose Empty application



Hello World

- **main.dart**

```
import 'package:flutter/material.dart';

void main() {
	runApp(const MainApp());
}

class MainApp extends StatelessWidget {
	const MainApp({super.key});

	@override
	Widget build(BuildContext context) {
		return MaterialApp(
			home: Scaffold(
				appBar: AppBar(title: Text('Hello World App')),
				body: Center(
					child: Text('Hello World!'),
				),
			),
		);
	}
}
```

Hello World

- **main.dart**

```
import 'package:flutter/material.dart';

void main() {
	runApp(const MainApp());
}

class MainApp extends StatelessWidget {
	const MainApp({super.key});

	@override
	Widget build(BuildContext context) {
		return MaterialApp(
			home: Scaffold(
				appBar: AppBar(title: Text('Hello World App')),
				body: Center(
					child: Text('Hello World!'),
				),
			),
		);
```

Flutter is declarative as opposed to imperative (Kotlin/Java)

object.title='sss' ;
object.color='#FF0000' ;

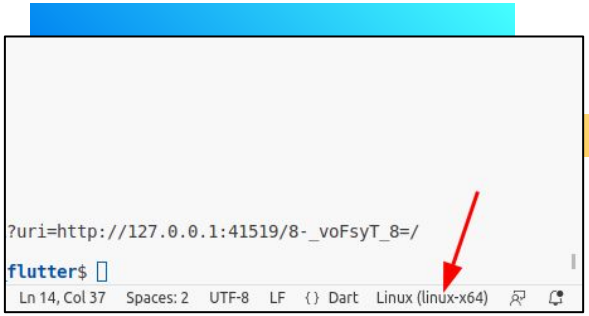
Hello World ! in Flutter

```
Widget(
    propertyOne:someData,
    propertyOne:someData,
    child: AnotherWidget...
)
```

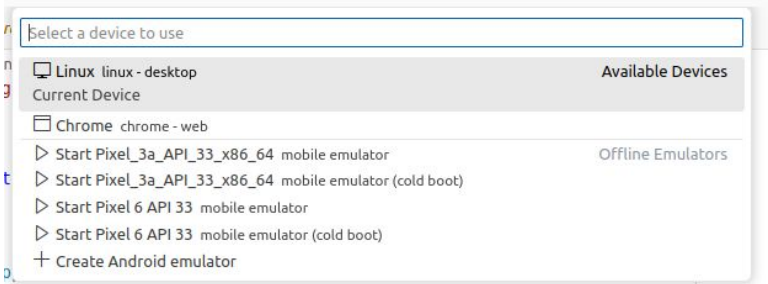
Hello World ! in Flutter

As opposed to Kotlin/Native Android where you have access to the references of all widgets (R.id.bt_add ..), With Declarative ?

Hello World ! in Flutter



- **To run**
  - Click on the device at the bottom right corner
  - Choose an emulator or device
  - Type in the terminal :

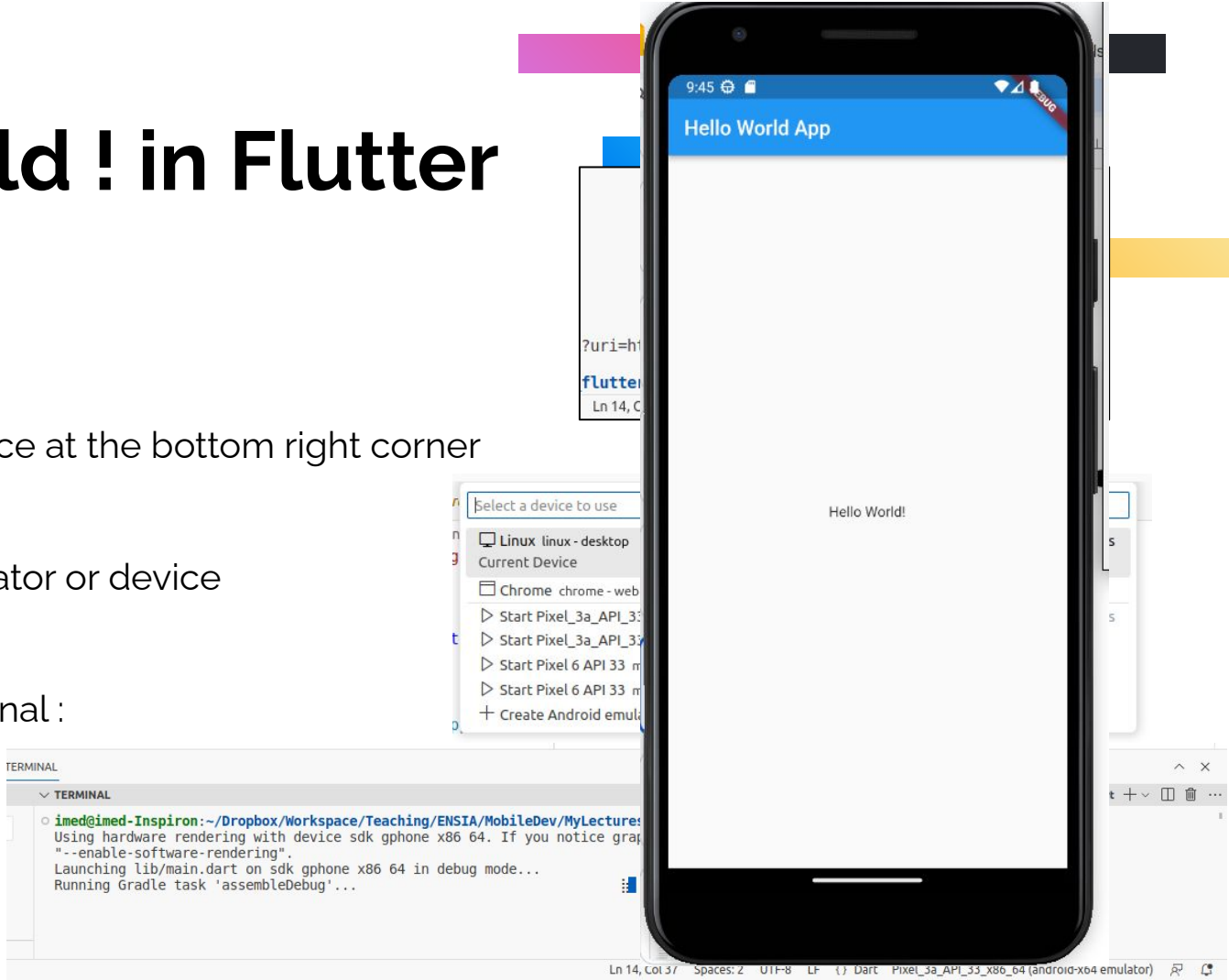


flutter run

Hello World ! in Flutter

- **To run**
  - Click on the device at the bottom right corner
  - Choose an emulator or device
  - Type in the terminal :

    flutter run

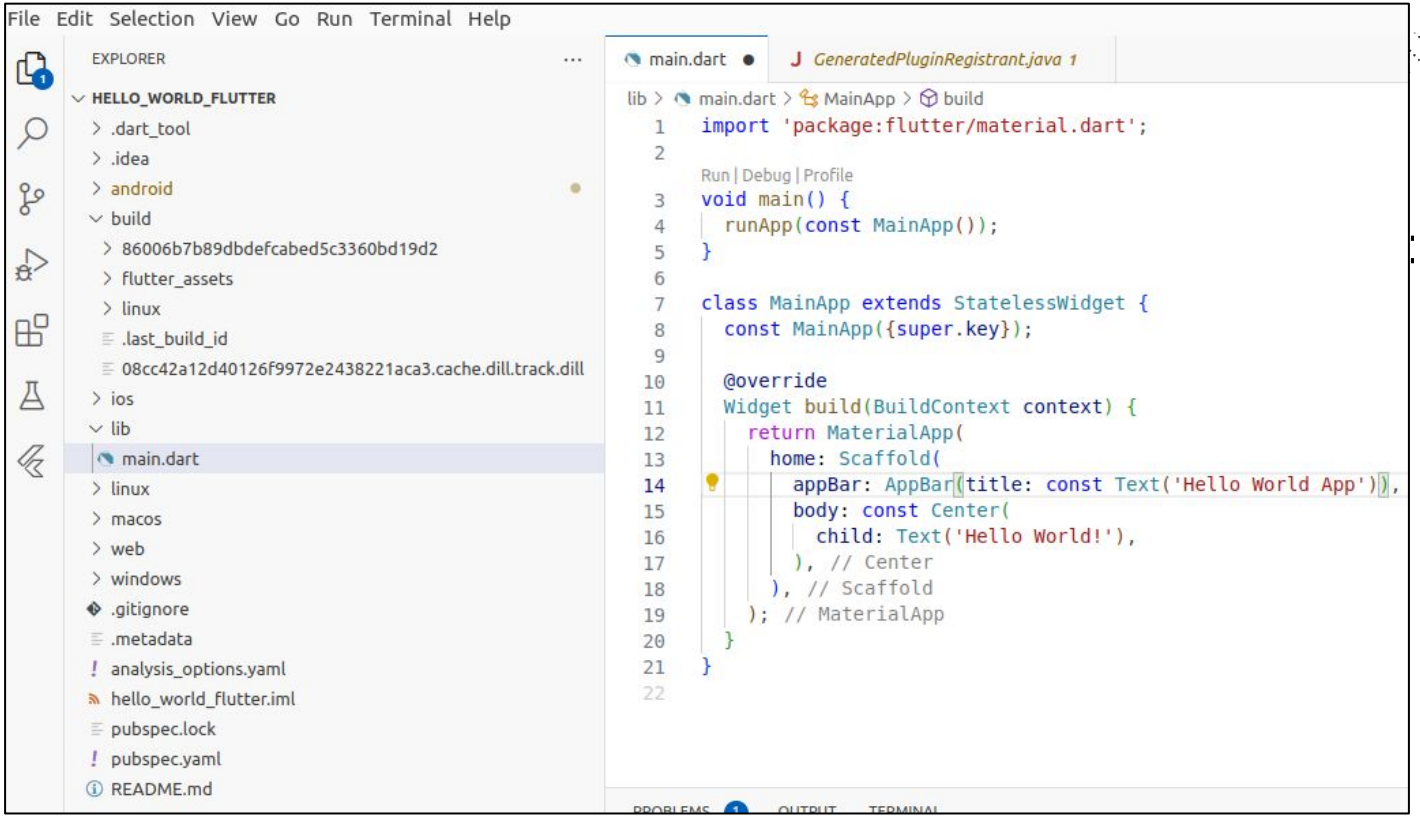


Hello World ! in Flutter

- **Hot reload**
  - When you modify the code :
  -
  -

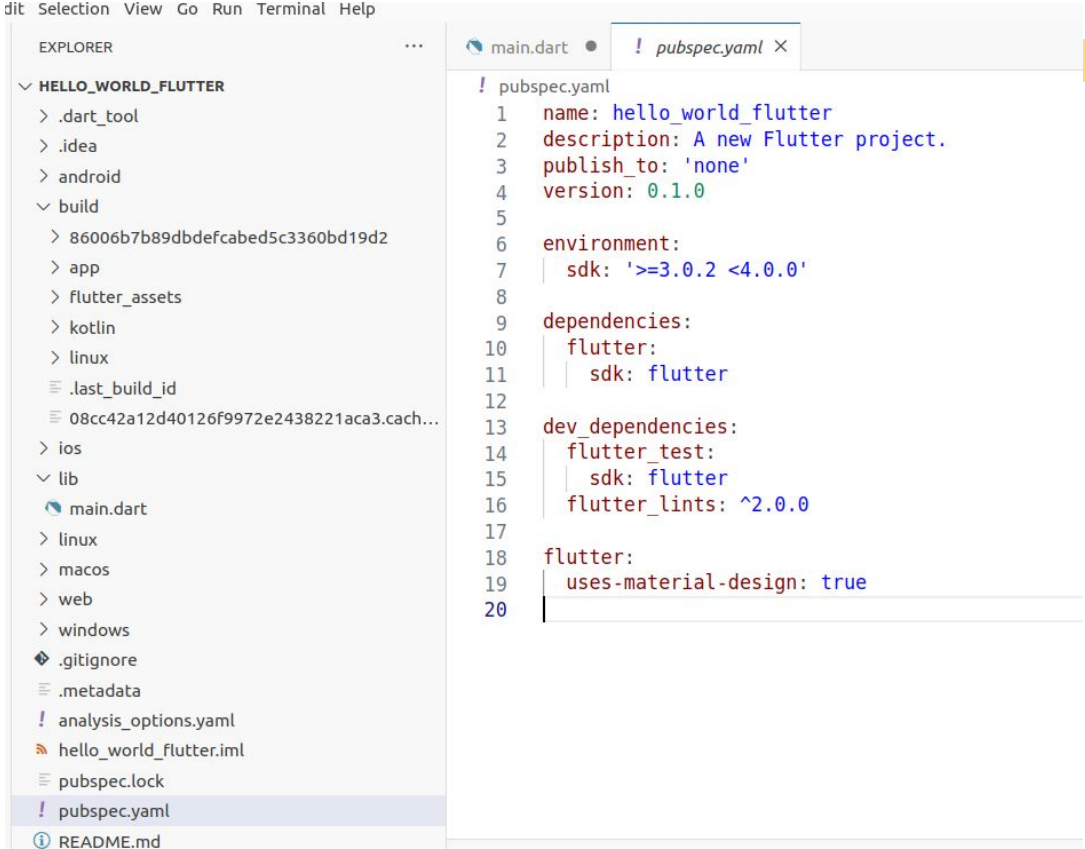
Inside the terminal, click the letter "**r**" to reload the App with the latest changes.

Flutter Project Structure



Flutter Project Structure

- **pubspec.yaml**
  - is a configuration file that is used to define and manage:
    - Dependencies
    - Metadata
    - Settings
    - Assets (Images, Fonts…)



YAML : Yet another markup language→later changed to YAML Ain't Markup Language

Flutter Project Structure

- **Where to find flutter libraries**
  - Visit https://pub.dev/ and search for whatever feature you want to add :

Flutter Project Structure

- **Where to find flutter libraries**

pub.dev

Sign in Help

datepicker

Platforms

- Android
- iOS
- Linux
- macOS
- Web
- Windows

SDKs

License

Advanced

RESULTS 657 packages

SORT BY SEARCH RELEVANCE

calendar_date_picker2

256 LIKES | 140 PUB POINTS | 98% POPULARITY

A lightweight and customizable calendar picker based on Flutter CalendarDatePicker, with support for single date picker, range picker and multi picker.

v 0.5.3 (2 months ago) Apache-2.0 Dart 3 compatible

SDK FLUTTER PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS

scroll_date_picker

74 LIKES | 140 PUB POINTS | 95% POPULARITY

A customizable and easy-to-use date picker library for Flutter. Compatible with Android & iOS & Web.

v 3.7.3 (4 months ago) BSD-3-Clause Dart 3 compatible

SDK FLUTTER PLATFORM ANDROID IOS

syncfusion_flutter_datepicker

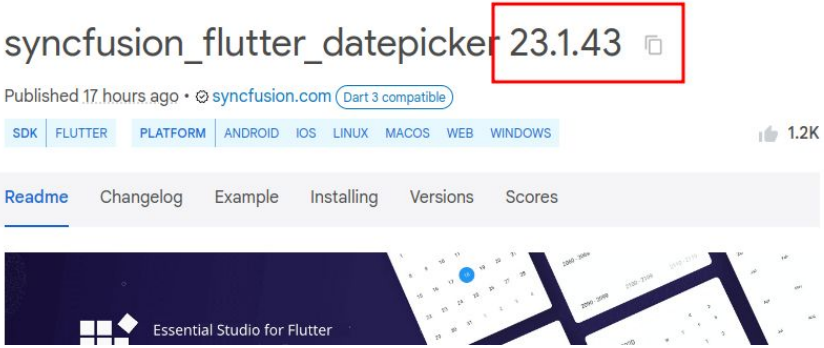
1266 LIKES | 130 PUB POINTS | 99% POPULARITY

The Flutter Date Range Picker widget allows users to easily select dates or a range of dates. It has four built-in views that allow quick navigation to the desired date.

Flutter Project Structure

- **Version of the libraries:**

```
dependencies:
  flutter:
    sdk: flutter

  cupertino_icons: ^1.0.5
  sqflite: ^2.2.8+2
  path:
  odoo_rpc: ^0.5.1
  convert: ^3.1.1
  provider: ^6.0.5
  flutter_launcher_icons: ^0.13.1
  sqflite_common_ffi: any
  sqflite_common_ffi_web: any
  flutter_barcode_scanner: ^2.0.0
  sqlite3_flutter_libs: any
  image_picker: ^0.8.7+5
  fluttertoast: ^8.2.2
```

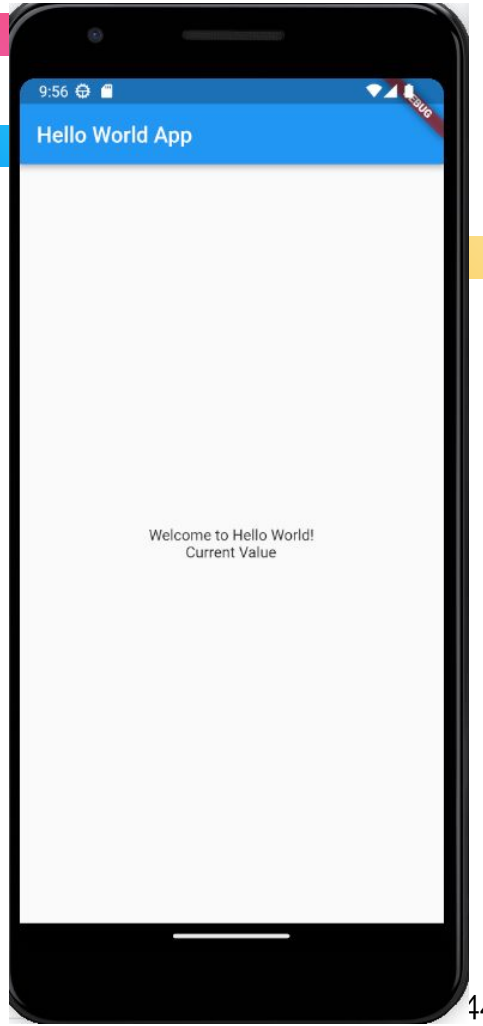


The ^ sign (Caret Syntax) = Any version guaranteed to be backwards compatible with the specified version.

Widgets for Building UIs

- **Texts vertically aligned , one line at a row:**

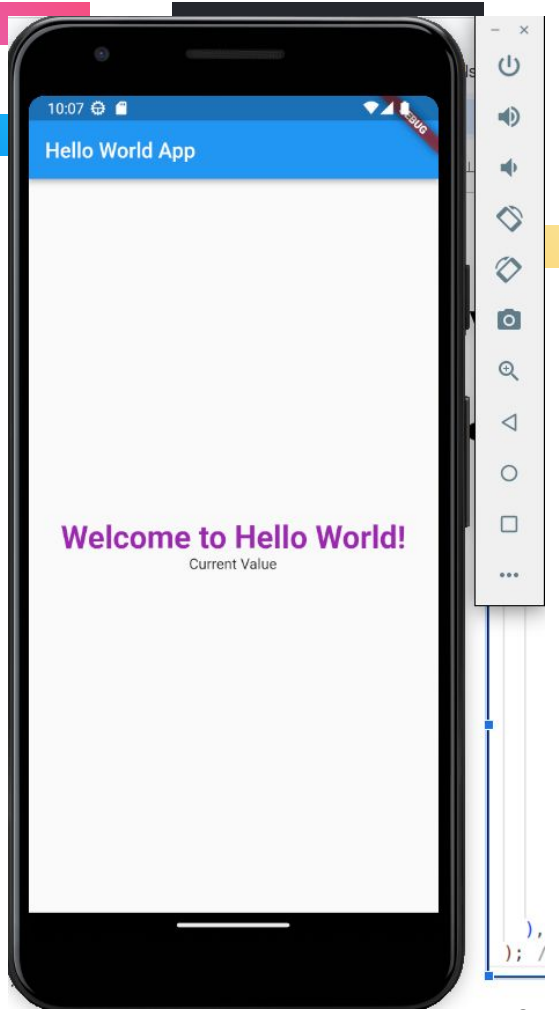
```
@override
Widget build(BuildContext context) {
    return MaterialApp(
       home: Scaffold(
       appBar: AppBar(title: const Text('Hello World App')),
       body: const Center(
       child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                crossAxisAlignment: CrossAxisAlignment.center,
                children: [
                     Text('Welcome to Hello World!'),
                     Text('Current Value'),
               ]),
              ),
          ),
     );
}
```



Widgets for Building UIs

- **Styling Text**

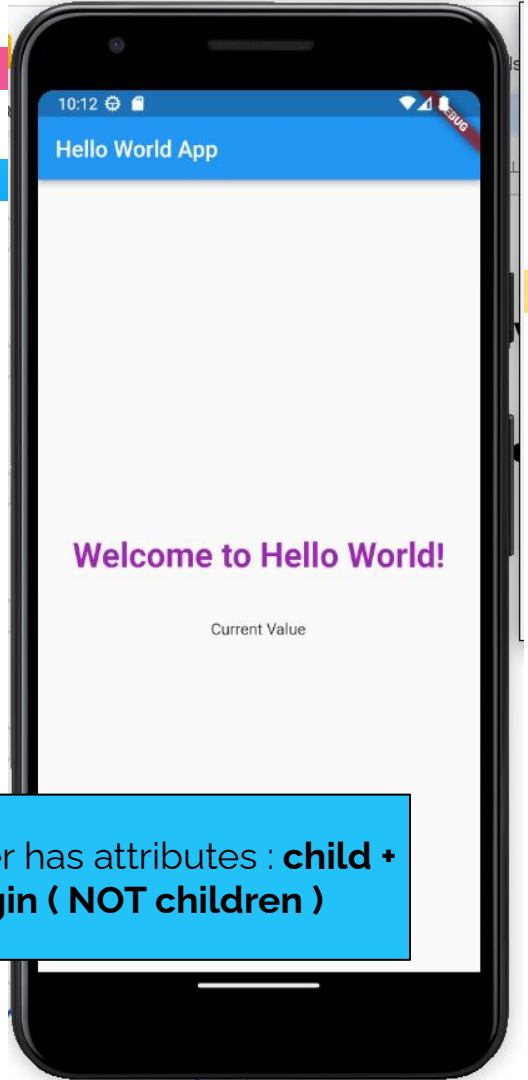
```
return MaterialApp(
  home: Scaffold(
    appBar: AppBar(title: const Text('Hello World App')),
    body: const Center(
      child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              'Welcome to Hello World!',
              textAlign: TextAlign.center,
              style: TextStyle(
                  color: Colors.purple,
                  fontWeight: FontWeight.bold,
                  fontSize: 30), // TextStyle
            ), // Text
            Text('Current Value'),
          ]), // Column
    ), // Center
  ), // Scaffold
); // MaterialApp
```



Widgets for Building UIs

- **Putting a Widget inside a Container + Margin**

```
body: Center(
  child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      crossAxisAlignment: CrossAxisAlignment.center,
      children: [
        Container(
          margin: const EdgeInsets.fromLTRB(10, 20, 10, 40),
          child: const Text(
            'Welcome to Hello World!',
            textAlign: TextAlign.center,
            style: TextStyle(
                color: Colors.purple,
                fontWeight: FontWeight.bold,
                fontSize: 30), // TextStyle
          ), // Text
        ), // Container
        Text('Current Value'),
      ]), // Column
), // Center
```

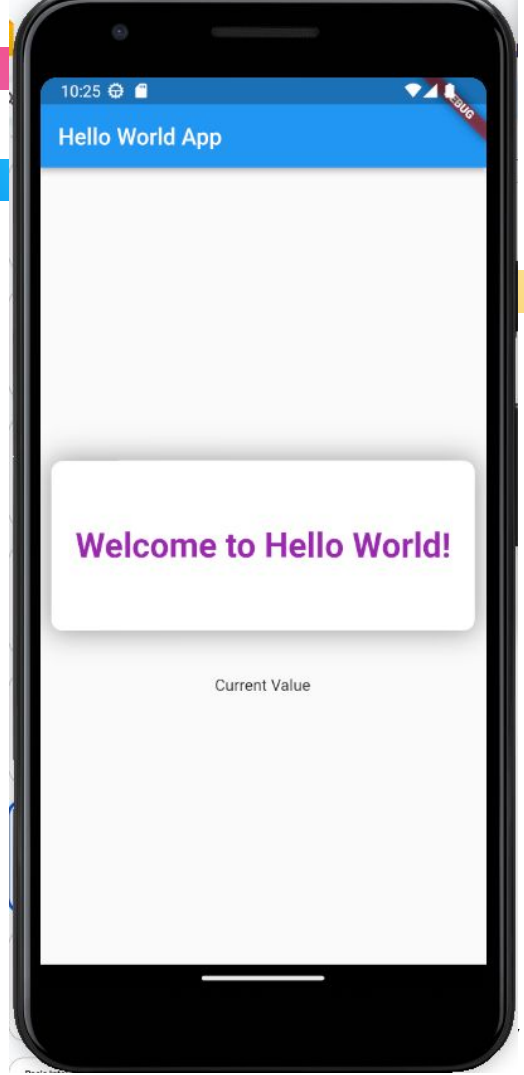


Container has attributes : **child + margin (NOT children)**

Widgets for Building UIs

- **Creating a nice rounded box with shadow**

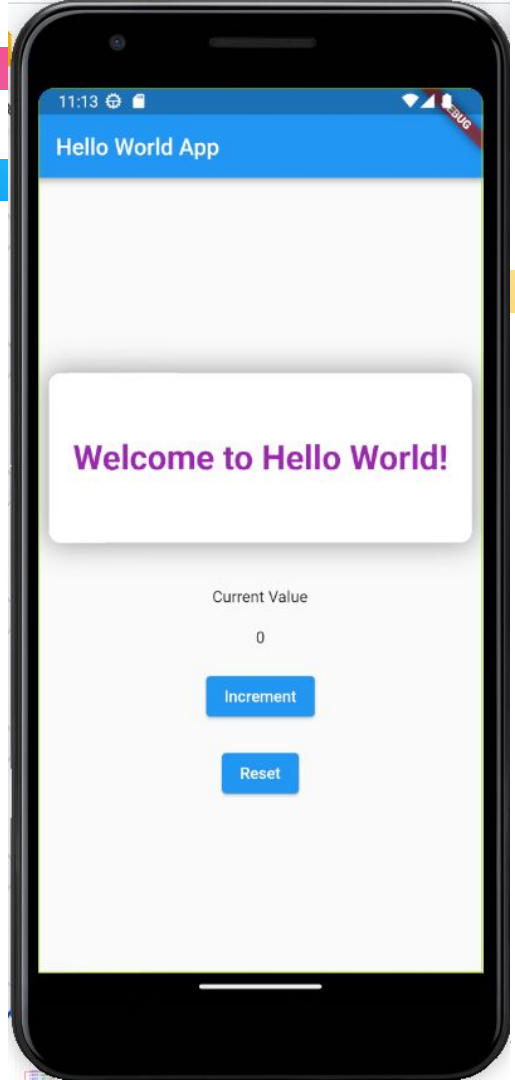
```
Container(
  height: 150.0,
  width: double.infinity,
  decoration: const BoxDecoration(
    color: Colors.white,
    borderRadius: BorderRadius.all(Radius.circular(10)),
    boxShadow: [
      BoxShadow(
        color: Colors.grey,
        blurRadius: 20.0, // Soften the shaodw
        spreadRadius: 2.0,
        offset: Offset(0.0, 0.0),
      ) // BoxShadow
    ],
  ), // BoxDecoration
  margin: const EdgeInsets.fromLTRB(10, 20, 10, 40),
  child: const Center(
    child: Text(
      'Welcome to Hello World!',
      textAlign: TextAlign.center,
      style: TextStyle(
        color: Colors.purple,
        fontWeight: FontWeight.bold,
        fontSize: 30), // TextStyle
    ), // Text
  )), // Center // Container
```



Widgets for Building UIs

- **Adding some Widgets : Text and Button**

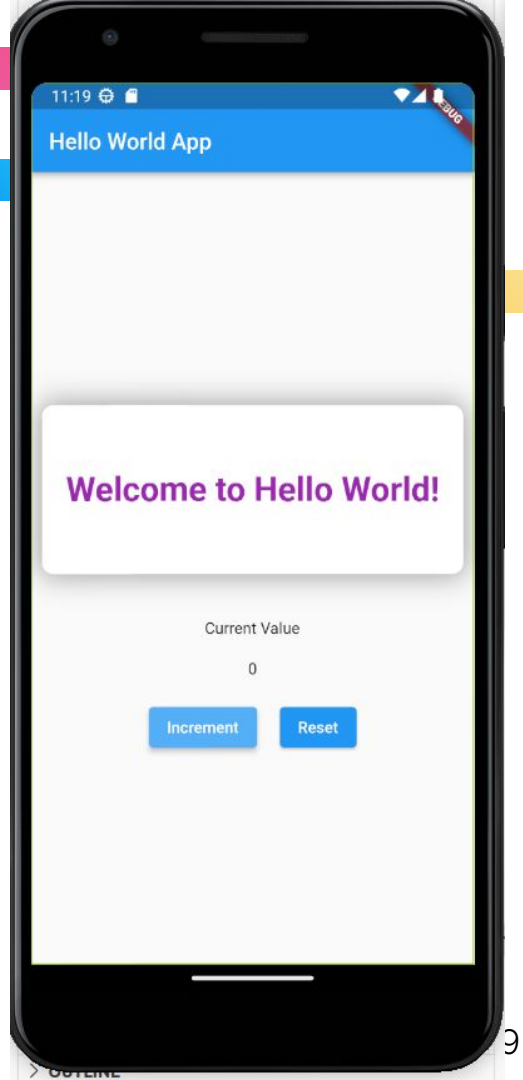
```
                    color: Colors.purple,
                    fontWeight: FontWeight.bold,
                    fontSize: 30), // TextStyle
                ), // Text
            )), // Center // Container
        const Text('Current Value'),
        const SizedBox(height: 20),
        const Text('0'),
        const SizedBox(height: 20),
        ElevatedButton(
            onPressed: () {}, child: const Text('Increment')), // Elevate
        const SizedBox(height: 20),
        ElevatedButton(onPressed: () {}, child: const Text('Reset')),
    ]), // Column
), // Center
```



Widgets for Building UIs

- **Adding Buttons horizontally aligned**

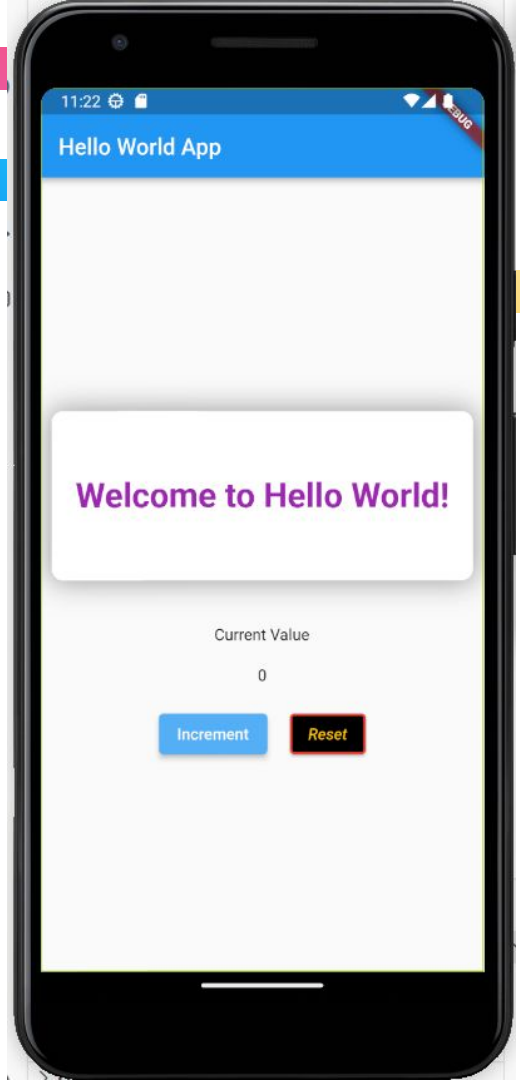
```
        const Text('0'),
        const SizedBox(height: 20),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            ElevatedButton(
                onPressed: () {}, child: const Text('Increment')), // ElevatedButto
            const SizedBox(width: 20),
            ElevatedButton(
                onPressed: () {}, child: const Text('Reset')), // ElevatedButton
          ],
        ) // Row
      ]), // Column
), // Center
```



Widgets for Building UIs

- **Styling for a Button**

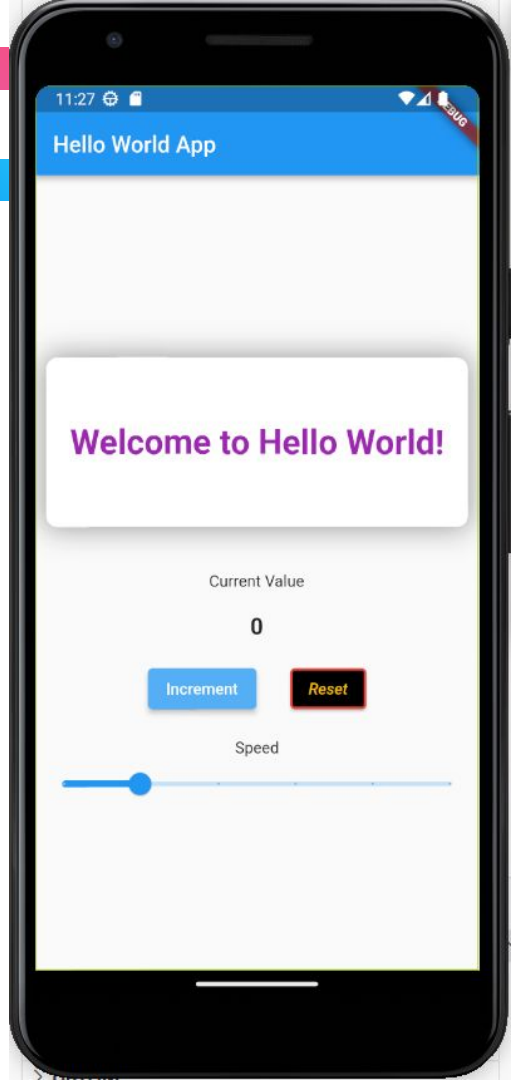
```
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  crossAxisAlignment: CrossAxisAlignment.center,
  children: [
    ElevatedButton(
        onPressed: () {}, child: const Text('Increment')), // ElevatedButton
    const SizedBox(width: 20),
    ElevatedButton(
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.black,
          foregroundColor: Colors.amber,
          side: BorderSide(color: Colors.red, width: 2),
        ),
        onPressed: () {},
        child: const Text('Reset',
            style: TextStyle(fontStyle: FontStyle.italic))), // Text // ElevatedButton
  ],
) // Row
```



Widgets for Building UIs

- **Adding a Slider**

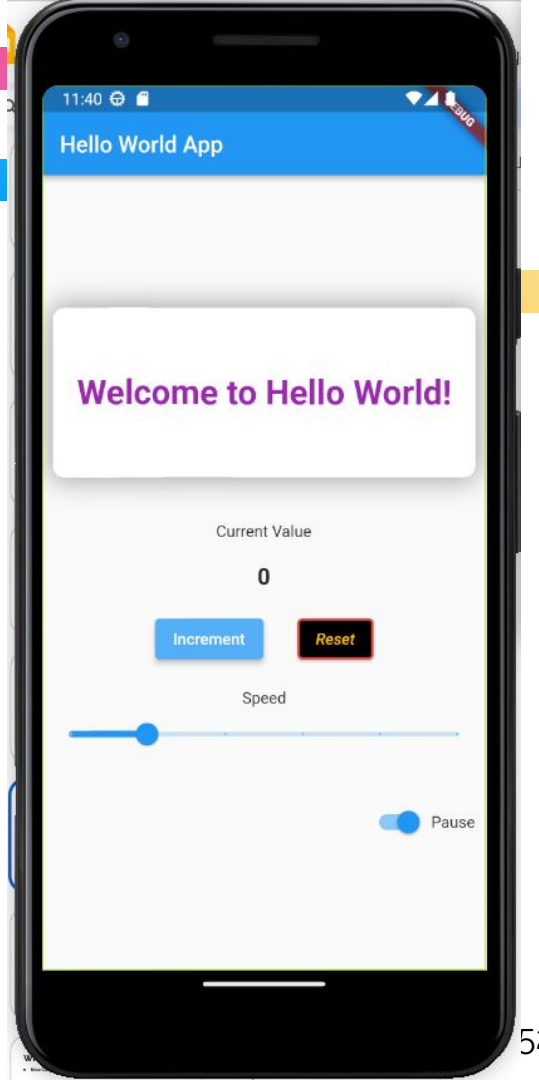
```
const SizedBox(height: 20),
const Text('Speed'),
Slider(
  divisions: 5,
  max: 100,
  value: 20,
  onChanged: (value) {},
), // Slider
```



Widgets for Building UIs

- **Adding a Switch : Toggle Button**

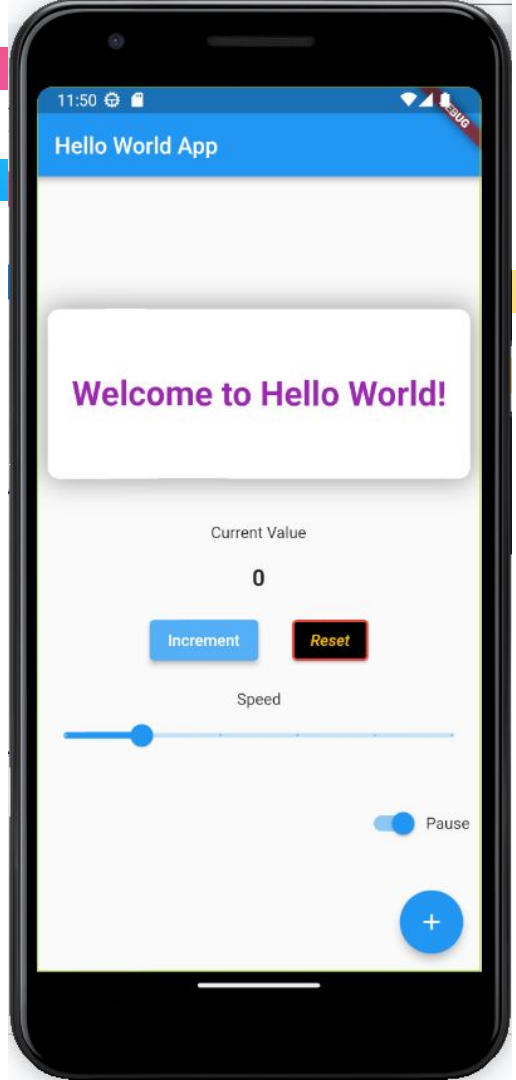
```
  ), // Slider
  const SizedBox(height: 20),
  Padding(
      padding: EdgeInsets.all(10),
      child: Row(children: [
        const Spacer(),
        Switch(
          value: true,
          onChanged: (value) {},
        ), // Switch
        const Text('Pause')
      ])), // Row // Padding
], // Column
/ Center
```



Widgets for Building UIs

- **Adding a floating Button**

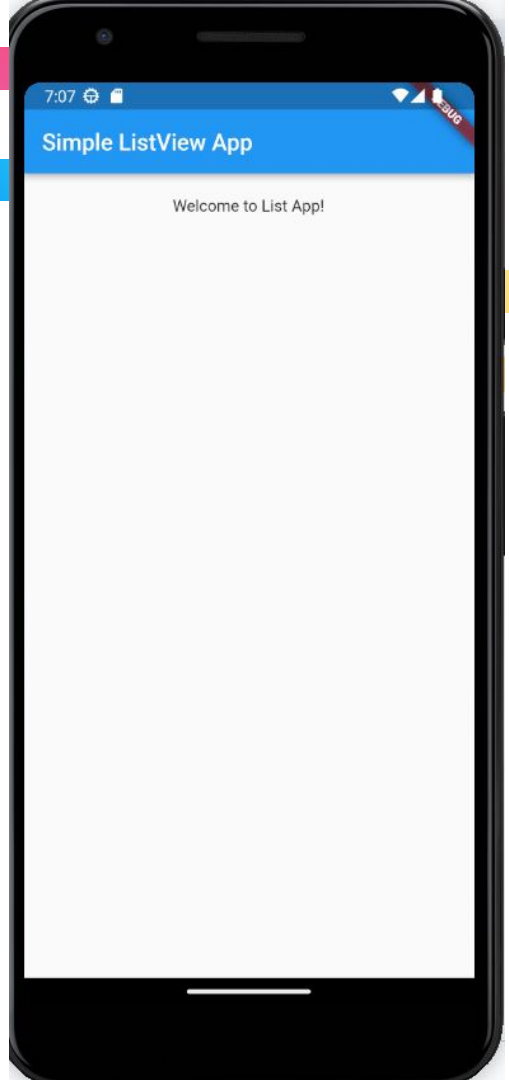
```
home: Scaffold(
  appBar: AppBar(title: const Text('Hello World App')),
  floatingActionButtonLocation: FloatingActionButtonLocation.endFloat,
  floatingActionButton: FloatingActionButton(
    onPressed: () {},
    child: Icon(Icons.add),
  ), // FloatingActionButton
  body: Center(
    child: Column(
```



Widgets for Building UIs

- **List View using Flutter**

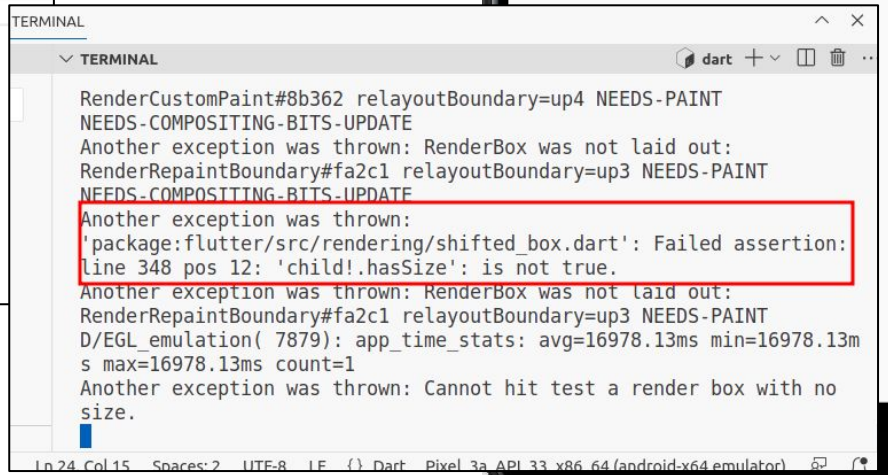
```
class MainApp extends StatelessWidget {
  MainApp({super.key});

  List<String> data = ['item 0', 'item 1', 'item 2', 'item 3', 'item 4'];

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: const Text('Simple ListView App')),
        body: const Center(
            child: Column(
                crossAxisAlignment: CrossAxisAlignment.center,
                children: [
              SizedBox(height: 20),
              Text('Welcome to List App!'),
            ])), // Column // Center
      ), // Scaffold
    ); // MaterialApp
  }
}
```

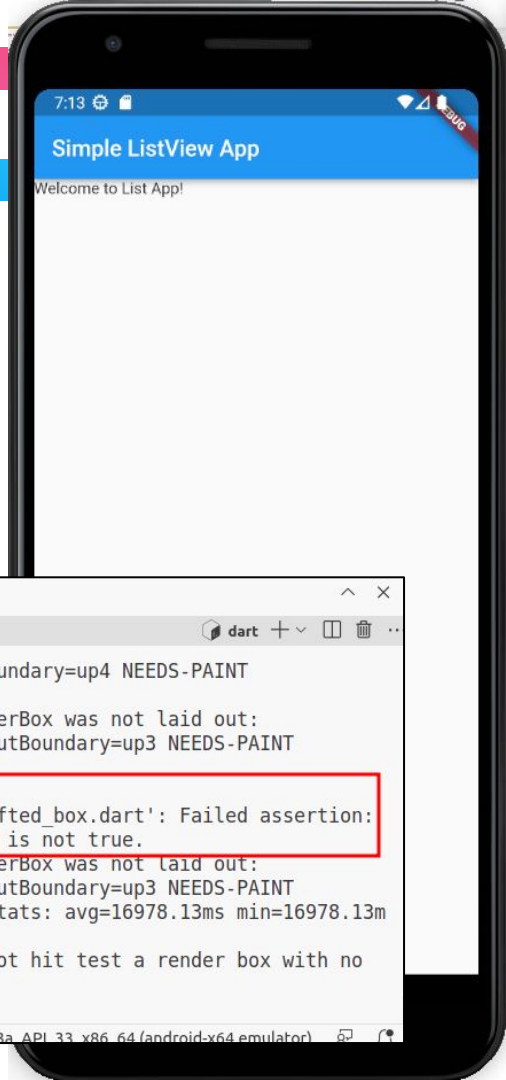


Widgets for Building UIs

- **List View using Flutter**

```
7        body: Center(
8            child: Column(
9                crossAxisAlignment: CrossAxisAlignment.center,
0                children: [
1              const SizedBox(height: 20),
2              const Text('Welcome to List App!'),
3              const SizedBox(height: 20),
4              ListView.builder(
5                itemCount: data.length,
6                itemBuilder: (context, index) {
7                  return ListTile(
8                    title: Text(data[index]),
9                  ); // ListTile
0                },
1              ), // ListView.builder
2            ])), // Column // Center
3      ), // Scaffold
```

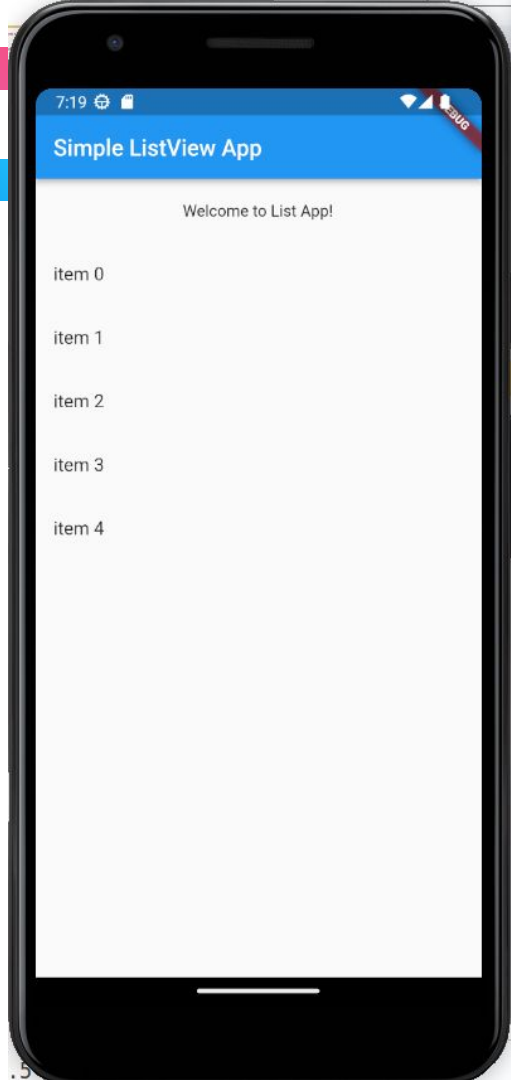




Widgets for Building UIs

- **List View using Flutter**

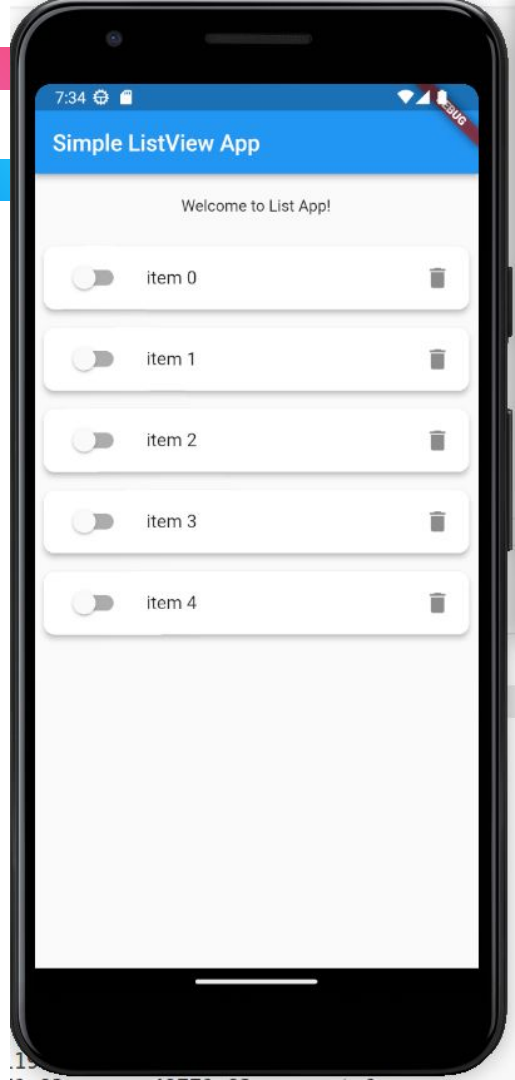
```
17          body: Center(
18              child: Column(
19                  crossAxisAlignment: CrossAxisAlignment.center,
20                  children: [
21              const SizedBox(height: 20),
22              const Text('Welcome to List App!'),
23              const SizedBox(height: 20),
24              Expanded(
25                  child: ListView.builder(
26                      itemCount: data.length,
27                      itemBuilder: (context, index) {
28                          return ListTile(
29                              title: Text(data[index]),
30                          ); // ListTile
31                      },
32                  ), // ListView.builder
33              ), // Expanded
34          ])), // Column // Center
```



Widgets for Building UIs

- **Customizing the List View using Flutter**

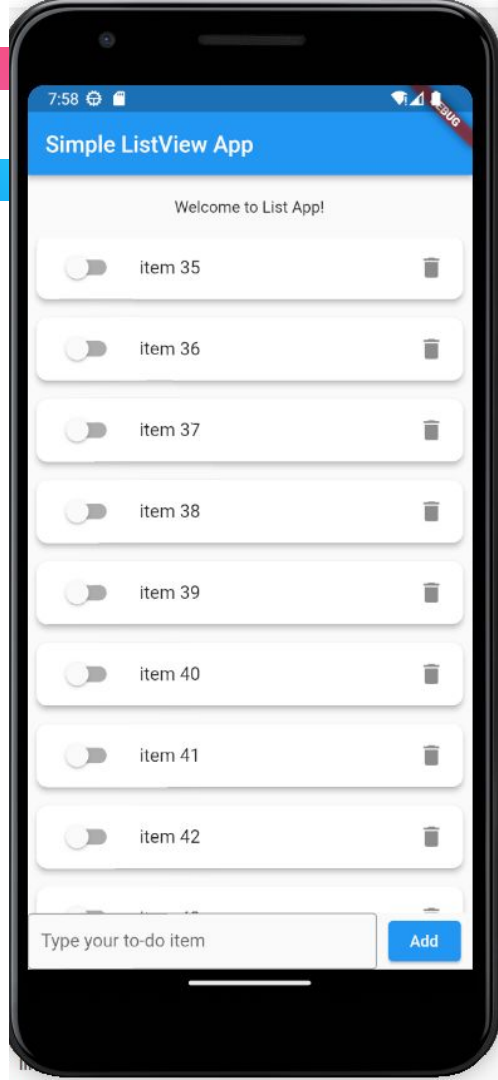
```
24          Expanded(
25            child: ListView.builder(
26              itemCount: data.length,
27              itemBuilder: (context, index) {
28                return Card(
29                  elevation: 4,
30                  margin: const EdgeInsets.all(8),
31                  shape: RoundedRectangleBorder(
32                    borderRadius: BorderRadius.circular(10.0),
33                  ), // RoundedRectangleBorder
34                  child: ListTile(
35                    title: Text(data[index]),
36                    leading: (Switch(
37                      value: false,
38                      onChanged: (value) {},
39                    )), // Switch
40                    trailing: const Icon(Icons.delete),
41                  ), // ListTile
42                ); // Card
43              },
44            ), // ListView.builder
45          ), // Expanded
```



Widgets for Building UIs

- **Fixed Bar at the bottom**

```
44                ), // ListView.builder
45              ), // Expanded
46              SizedBox(
47                height: 50,
48                width: MediaQuery.of(context).size.width,
49                child: Row(children: <Widget>[
50                  Expanded(
51                    child: TextFormField(
52                      decoration: const InputDecoration(
53                        border: OutlineInputBorder(
54                            borderSide: BorderSide(color: Colors.teal)), // OutlineInputBorder
55                        labelText: 'Type your to-do item',
56                      ), // InputDecoration
57                      keyboardType: TextInputType.text,
58                      onChanged: (newValue) {},
59                    ), // TextFormField
60                  ), // Expanded
61                  const SizedBox(width: 10),
62                  ElevatedButton(
63                    onPressed: () {},
64                    child: const Text(
65                      "Add",
66                    ), // Text
67                  ), // ElevatedButton
68                  const SizedBox(width: 10),
69                ]), // <Widget>[] // Row
70              ), // SizedBox
71            ])), // Column // Center
```



Widgets for Building UIs

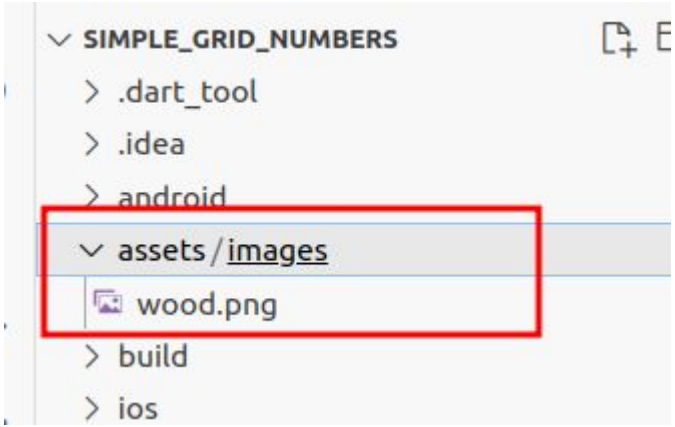
- **Creating via Loops and IFs**

```
var gridSize = 4;
@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(title: const Text('Grid of Numbers')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            for (int i = 0; i < gridSize; i++) Text('Hello World!')
          ]), // Column
      ), // Center
    ), // Scaffold
  ); // MaterialApp
}
```

Only single line statement are allowed for for or if = no { }

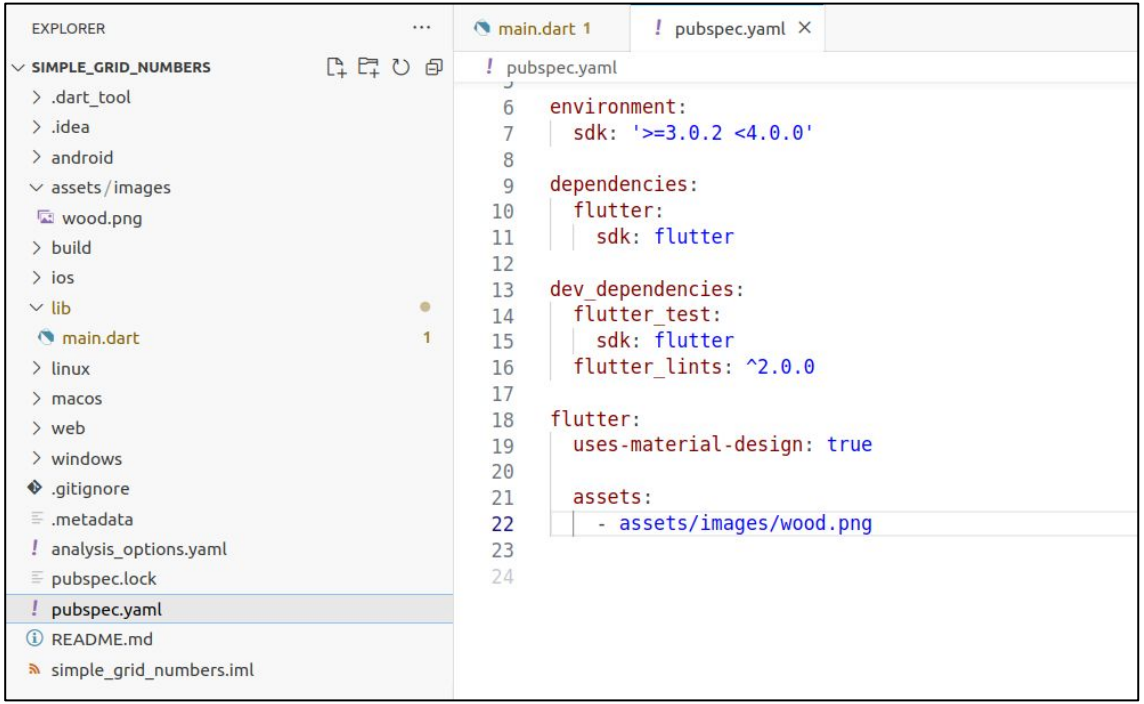
Widgets for Building UIs

- How to add an Image as a background
  - Create assets/images
    - Upload your image there



Widgets for Building UIs

- How to add an Image as a background
  - Upload pubspec.yaml file



Widgets for Building UIs

- **How to add an Image as a background**

```
return MaterialApp(
  home: Scaffold(
    appBar: AppBar(title: const Text('Grid of Numbers')),
    body: Center(
      child: Container(
      decoration: BoxDecoration(
        border: Border.all(color: Colors.blueAccent, width: 3),
      ), // BoxDecoration
      width: 80,
      height: 80,
      child: Stack(
        children: [
          Positioned.fill(
            child: Opacity(
              opacity: 0.8,
              child:
                Image.asset('assets/images/wood.png', fit: BoxFit.cover),
            ), // Opacity
          ), // Positioned.fill
          const Center(
              child: Text('ABC',
                style: TextStyle(
                  fontSize: 30, fontWeight: FontWeight.bold))), // TextStyle //
        ],
      ), // Stack
    )), // Container // Center
  ), // Scaffold
); // MaterialApp
```

Widgets for Building UIs

- **Encapsulating Widgets as Functions**

```
Widget getCell(int number) {
  return Container(
    decoration: BoxDecoration(
      border: Border.all(color: Colors.blueAccent, width: 3),
    ), // BoxDecoration
    width: 80,
    height: 80,
    child: Stack(
      children: [
        Positioned.fill(
          child: Opacity(
            opacity: 0.8,
            child: Image.asset('assets/images/wood.png', fit: BoxFit.cover),
          ), // Opacity
        ), // Positioned.fill
        Center(
            child: Text('$number',
                style: TextStyle(fontSize: 30, fontWeight: FontWeight.bold)), // Text // Center
      ],
    ), // Stack
  ); // Container
}
```

Widgets for Building UIs

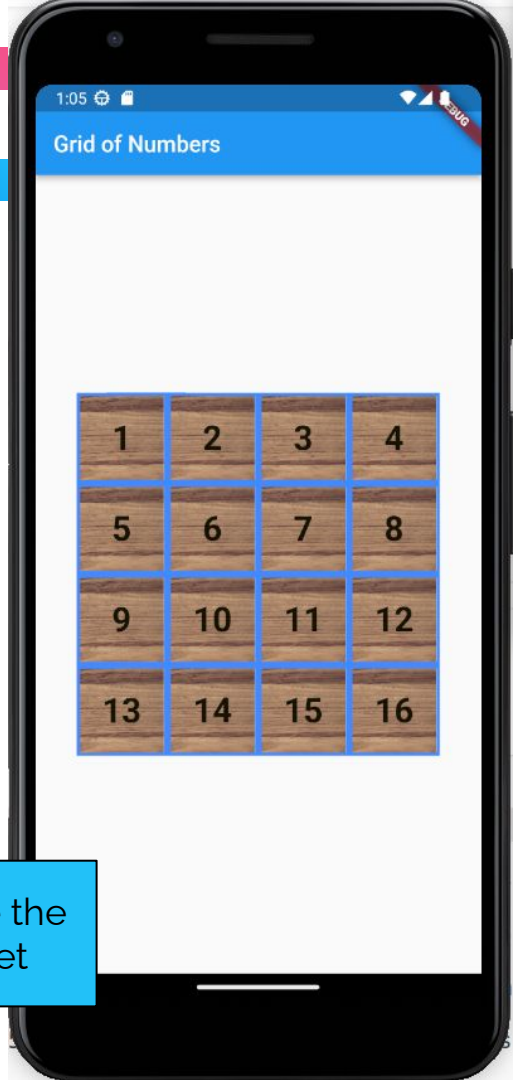
- **Encapsulating Widgets as Functions**

```
Widget build(BuildContext context) {

  return MaterialApp(
    home: Scaffold(
        appBar: AppBar(title: const Text('Grid of Numbers')),
        body: Center(
          child: getCell(234),
        )), // Center // Scaffold
  ); // MaterialApp
}

Widget getCell(int number) {
  return Container(
    decoration: BoxDecoration(
      border: Border.all(color: Colors.blueAccent, width: 3),
    ), // BoxDecoration
    width: 80,
    height: 80,
```

Widgets for Building UIs

- **How to make a numbered Grid :**

```
var gridSize = 4;
var increment = 0;
@override
Widget build(BuildContext context) {
  increment = 1;
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(title: const Text('Grid of Numbers')),
      body: Center(
        child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.center,
            children: [
              for (int row = 0; row < gridSize; row++)
                Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  crossAxisAlignment: CrossAxisAlignment.center,
                  children: [
                    for (int col = 0; col < gridSize; col++, increment++)
                      getCell(increment)
                  ],
                ), // Row
            ]), // Column
      ), // Center
    ), // Scaffold
  ); // MaterialApp
}
```

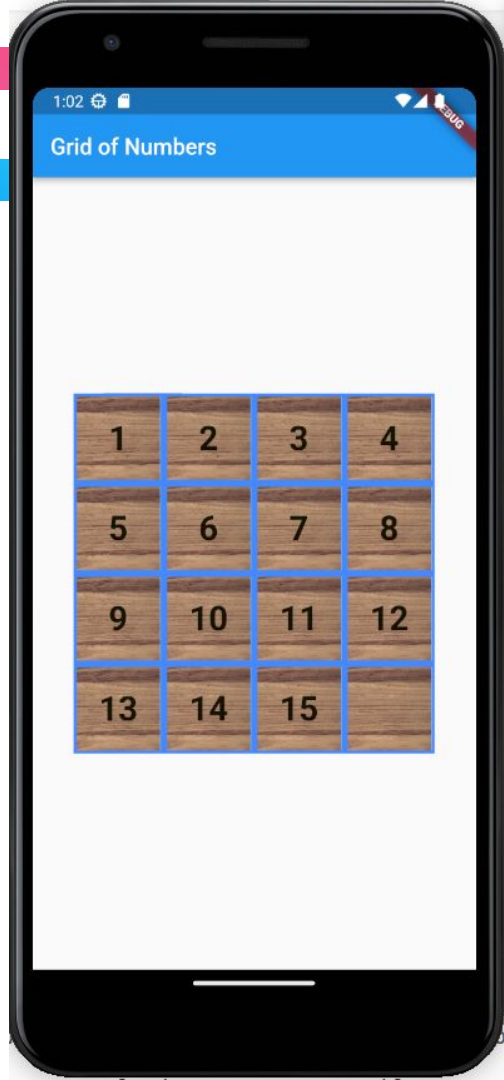


You can even use the GridView Widget

Widgets for Building UIs

- **How to make a numbered Grid : : Last Cell Empty**

```
Widget getCell(int number) {
  if (number == gridSize * gridSize) {
    return Container(
      decoration: BoxDecoration(
        border: Border.all(color: Colors.blueAccent, width: 3),
      ), // BoxDecoration
      width: 80,
      height: 80,
      child: Stack(
        children: [
          Positioned.fill(
            child: Opacity(
              opacity: 0.8,
              child: Image.asset('assets/images/wood.png', fit: BoxFit.cover),
            ), // Opacity
          ), // Positioned.fill
          Center(
            child: Text(' ',
                style:
                    TextStyle(fontSize: 30, fontWeight: FontWeight.bold))), // Text // C
        ],
```



Widgets for Building UIs

- **Learn more on other widgets:**
  - https://gallery.flutter.dev/

Stateless & Stateful Widgets for Flutter

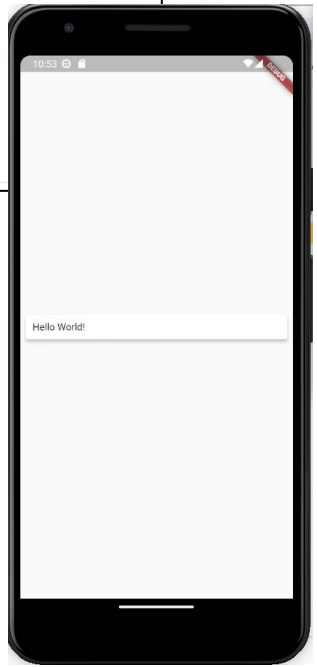
- **Stateless Widget :**
  - Once it is created and drawn on the screen, they will never change
  - Better suited for static user interfaces.

```dart
import 'package:flutter/material.dart' ;

class MyExampleStatelessWidget  extends StatelessWidget
      final String text;
      const MyExampleStatelessWidget (this.text);

      @override
      Widget build(BuildContext context) {
            return Container(
                  width: double.infinity,
                  child: Card(
                  elevation: 4,
                  margin: const EdgeInsets.all(8),
                  child: Padding(
                        padding: EdgeInsets.all(10),
                        child: Text(text)
                        )
                  )
            );
      }
}
```

main.dart > ...

```dart
import 'package:flutter/material.dart';
import 'package:stateless_widget_example/widgets/mywidget.dart';

Run | Debug | Profile
void main() {
  runApp(const MainApp());
}

class MainApp extends StatelessWidget {
  const MainApp({super.key});

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      home: Scaffold(
        body: Center(
          child: MyExampleStatelessWidget('Hello World!'),
        ), // Center
      ), // Scaffold
    ); // MaterialApp
  }
}
```

aces.



Stateless & Stateful Widgets for Flutter

- **Stateful Widget :**
  - They can change based on the interaction with the user. (i,e they are mutable)

```dart
import 'package:flutter/material.dart';
class MyStatefulExample extends StatefulWidget {
	@override
	_MyStatefulExampleState createState() => _MyStatefulExampleState();
}
class _MyStatefulExampleState extends State<MyStatefulExample> {
	int _counter = 0;
	@override
	Widget build(BuildContext context) {
		return Column(
			children: <Widget>[
				SizedBox(height: 100),
				Text('Counter: $_counter'),
				ElevatedButton(
					onPressed: () {
						setState(() {
							_counter = _counter + 1;
						});
					},
				child: Text('Increment'),
				),
			],
		);
	}
```

```dart
Run | Debug | Profile
void main() {
  runApp(MainApp());
}

class MainApp extends StatelessWidget {
  const MainApp({super.key});

  @overload
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        body: Center(
          child: MyStatefulExample(),
        ), // Center
      ), // Scaffold
    ); // MaterialApp
  }
}
```

Hello World ! in Flutter

As opposed to Kotlin/Native Android where you have access to the references of all widgets (R.id.bt_add ..), With Declarative ?

Hello World ! in Flutter

You create variables, data collections, objects and you attach them to widgets,
Later, You change the data or variable values and you setState to refresh the screen

Section 2

Demo Apps in Flutter

Incrementing App in Flutter

INTERACTIVE_INCREMEN...
- .dart_tool
- .idea
- .vscode
- android
- build
- ios
- lib
  - screens
    - homescreen.dart
  - main.dart
- linux
- macos
- web
- windows
- .gitignore
- .metadata
- analysis_options.yaml
- interactive_increme...
- pubspec.lock
- pubspec.yaml
- README.md

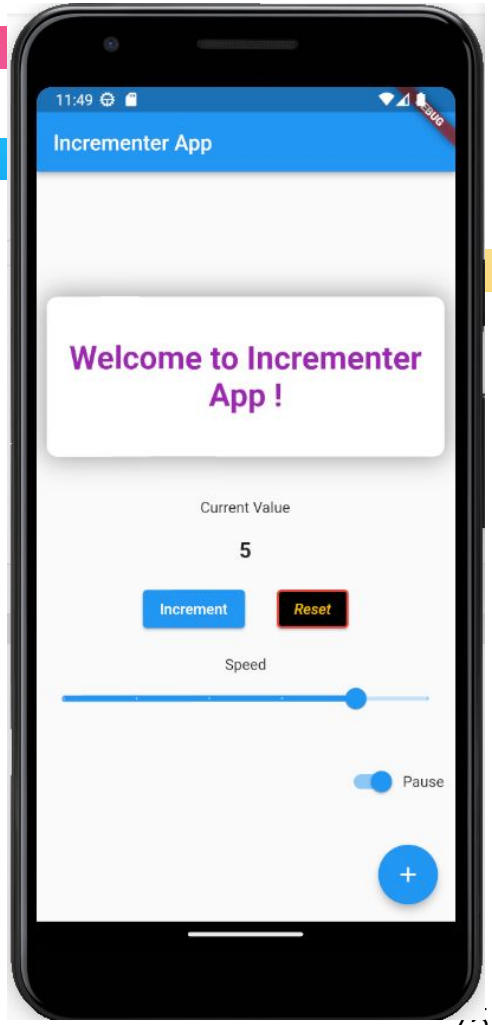
lib > main.dart > MainApp > build

```
1  import 'package:flutter/material.dart';
2  import 'package:interactive_increment_app/screens/homescreen.dart
3
   Run | Debug | Profile
4  void main() {
5    runApp(const MainApp());
6  }
7
8  class MainApp extends StatelessWidget {
9    const MainApp({super.key});
10
11   @override
12   Widget build(BuildContext context) {
13     return MaterialApp(
14       home: HomeScreen(),
15     ); // MaterialApp
16   }
17 }
18
```

PROBLEMS 6 OUTPUT TERMINAL

TERMINAL

```
D/EGL_emulation( 4043): app_time_stats: avg=601.79ms min=6.72ms max=5
```



To-Do App (No DB)

EXPLORER

INTERACTIVE_TODO_APP
- .dart_tool
- .idea
- android
- build
- ios
- lib
  - screens
    - homescreen.dart
  - main.dart
- linux
- macos
- web
- windows
- .gitignore
- .metadata
- analysis_options.yaml
- interactive_todo_ap...
- pubspec.lock
- pubspec.yaml
- README.md

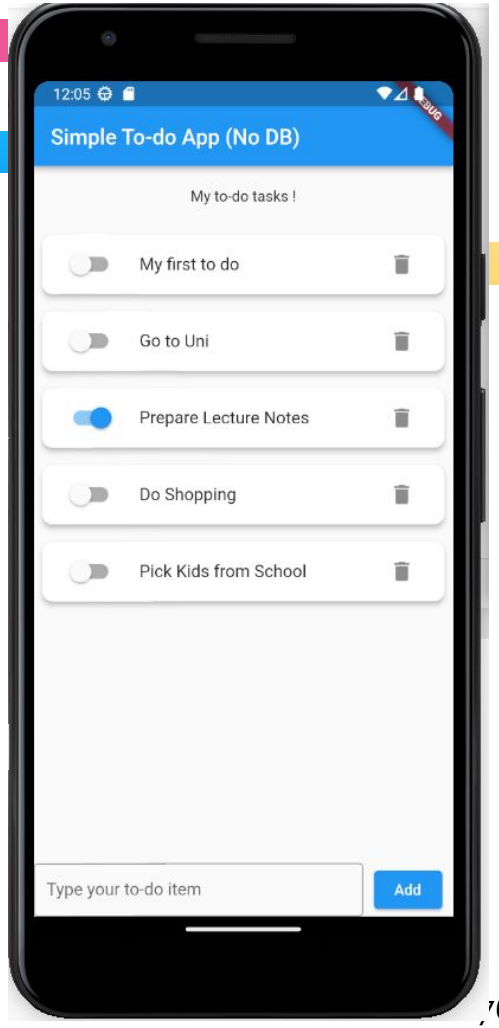
main.dart × homescreen.dart

lib > main.dart > MainApp > build

```
1  import 'package:flutter/material.dart';
2  import 'package:interactive_todo_app/screens/homescreen
3
   Run | Debug | Profile
4  void main() {
5    runApp(const MainApp());
6  }
7
8  class MainApp extends StatelessWidget {
9    const MainApp({super.key});
10
11   @override
12   Widget build(BuildContext context) {
13     return MaterialApp(
14       home: HomeScreen(),
15     ); // MaterialApp
16   }
17 }
18
```

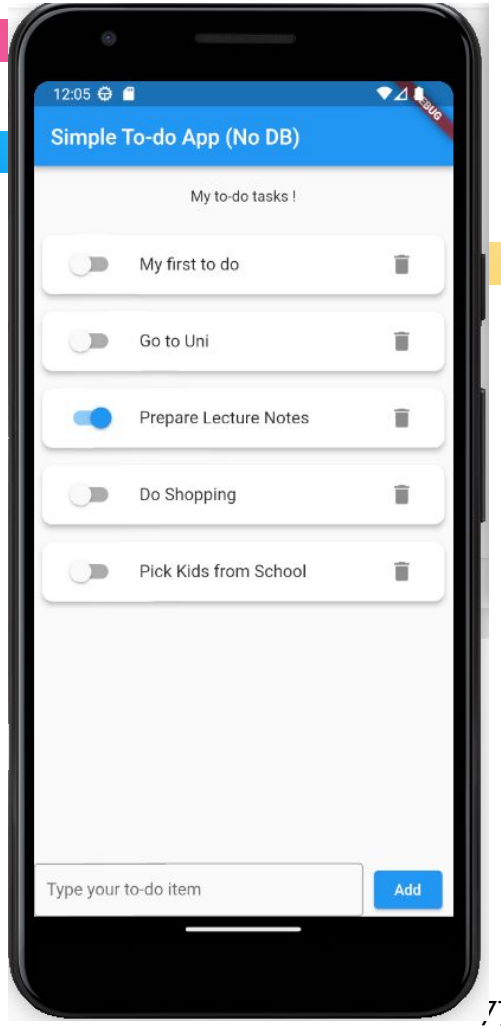
PROBLEMS 5 OUTPUT TERMINAL

TERMINAL



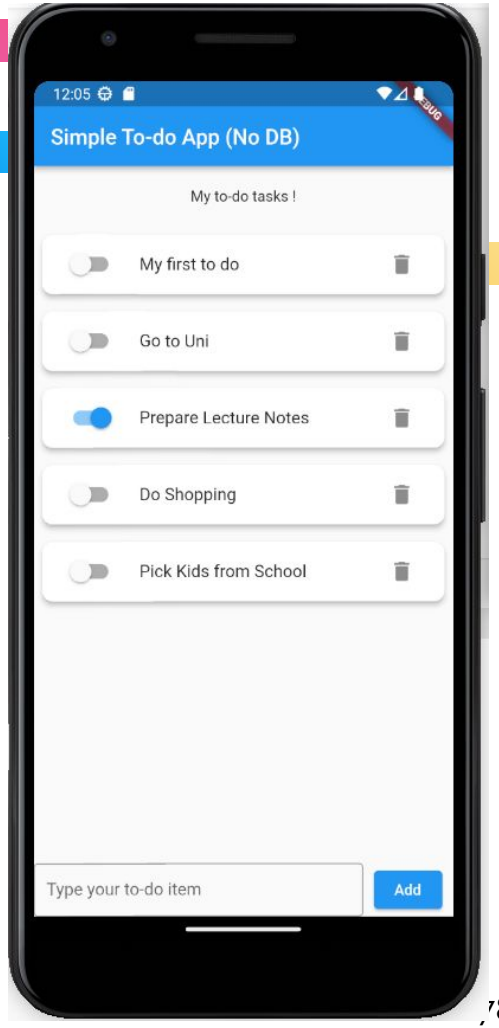
To-Do App (No DB)

```
class HomeScreen extends StatefulWidget {
  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  List<Map> data = [
    {'title': 'My first to do', 'done': false}
  ];
  String _tx_title_value = '';
  final _tx_title_controller = TextEditingController();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
```



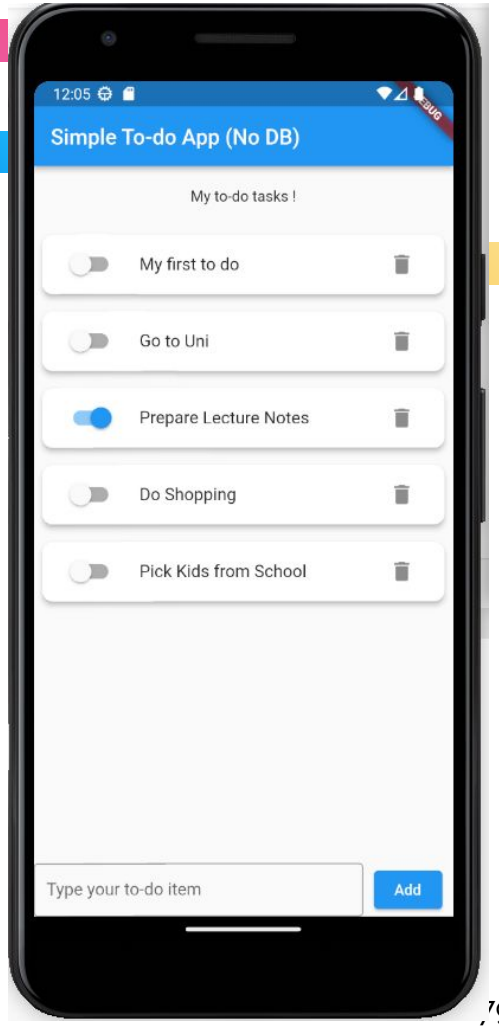
To-Do App (No DB)

```
74          ), // Expanded
75          const SizedBox(width: 10),
76          ElevatedButton(
77            onPressed: () {
78              _tx_title_controller.text = '';
79              data.add({
80                'title': _tx_title_value,
81                'done': false,
82              });
83              setState(() {});
84            },
85            child: const Text(
86              "Add",
87            ), // Text
88          ), // ElevatedButton
89          const SizedBox(width: 10),
90        ]), // <Widget>[] // Row
```



To-Do App (No DB)

```
27          itemBuilder: (context, index) {
28            return Card(
29              elevation: 4,
30              margin: const EdgeInsets.all(8),
31              shape: RoundedRectangleBorder(
32                borderRadius: BorderRadius.circular(10.0),
33              ), // RoundedRectangleBorder
34              child: ListTile(
35                title: Text(data[index]['title']),
36                leading: (Switch(
37                  value: data[index]['done'],
38                  onChanged: (value) {
39                    data[index]['done'] = value;
40                    setState(() {});
41                  },
42                )), // Switch
43                trailing: IconButton(
44                  icon: Icon(
45                    Icons.delete,
46                  ), // Icon
47                  onPressed: () {
48                    data.removeAt(index);
49                    setState(() {});
50                  },
51                ), // IconButton
52              ), // ListTile
53            ), // Card
```



Constructing UIs for Flutter Apps

- **Scaffold :** appBar, body, floatingActionButton, drawer, bottomNavigationBar
- **AppBar : title ,** leading**,** elevation
- **Container : child**, padding, margin, height, width,decoration alignment
- **Column :** children, mainAxisAlignment, crossAxisAlignment
- **Row :** children, mainAxisAlignment, crossAxisAlignment
- **Stack :** alignment, children,
- **SizedBox :** child ,height, width ,
- **Card :** child, color, elevation, margin, clipBehavior, shape
- **Expanded :** child, flex
- **Spacer :** flex
- **Padding :** padding --> EdgeInsets.all|only|fromLTRB|symmetric
- **Center :** child
- **Align** : child,alignment
- **ListView.builder :**scrollDirection, physics, itemCount, itemBuilder(context,index)
- **GridView.builder :** padding, physics, itemCount, itemBuilder(context,index)
- **Text :** style, textAlign
- **TextFormField :**onChanged, controller, decoration, validator
- **ElevatedButton :**style, onPressed, child
- **IconButton :** icon, onPressed
- **Icon :** color, size,
- **Slider :**value, max, divisions, onChanged, label
- **Switch :**value, onChanged, thumbIcon
- **ListTile :** leading, title, subtitle, trailing, tileColor, onTap
- **Image.asset :** imagepath,
- **TextStyle :** fontSize, color, fontWeigth
- **BoxDecoration :** image, shape, border, borderRaduis, color
- **Opacity :** opacity, child
- **RoundedRectangleBorder :**borderRadius, side
- **Position :** left, bottom, right, top
- **BoxShadow :** color, offset, blurStyle

Lecture Demo Apps

- **Compute Factorial in Dart**
  - https://www.dropbox.com/scl/fo/4scdnwyub1bx8on2k2zn9/h?rlkey=2mt4k04lulngz1bqop6mj3tmo&dl=0
- **Hello World App : Using Widgets.**
  - https://www.dropbox.com/scl/fo/3ug0k10vz906xfnwse5bl/h?rlkey=q7a2umvmy6vpkjwqqniu55cxz&dl=0
- **To-Do UI with a ListView**
  - https://www.dropbox.com/scl/fo/tiwsim8y10wl71i1ip4lo/h?rlkey=y7irpo6f61vbtuubybi7ghsh2&dl=0
- **Stateless Widget Example**
  - https://www.dropbox.com/scl/fo/k4q0g74b8dwpb16amctnj/h?rlkey=olxwsx7dihxe3cfcn4m85gwnj&dl=0
- **Stateful Widget Example**
  - https://www.dropbox.com/scl/fo/o07enhjv74v5890ivg2n8/h?rlkey=qoiwer5htvgh575ozpvok8cj6&dl=0
- **Incrementing App**
  - https://www.dropbox.com/scl/fo/9jtef5wr9yiskze215yc8/h?rlkey=176pv6kep1w2wgc8cvldvg9zl&dl=0
- **Interactive To-Do App**
  - https://www.dropbox.com/scl/fo/o84pyfe0m98gnay5mlynz/h?rlkey=3g3x64pnq61rt4u5e31x3drpl&dl=0

Resources

- https://m3.material.io/
- https://codelabs.developers.google.com/codelabs/flutter-codelab-first#0
- https://codewithandrea.com/tutorials/
- https://www.youtube.com/watch?v=cCMULezpNRQ
- https://www.youtube.com/watch?v=L9cP9OTUstA&list=PLjxrf2q8roU23XGwz3Km7sQZFTdB996iG
- https://www.youtube.com/watch?v=OBIuSrg_Quo&list=PLjxrf2q8roU1fRV40Ec8200rX6OuQkmnl
- https://www.youtube.com/watch?v=fQldtPigGBM&list=PLjxrf2q8roU3N8-rlgg8_jueoNT-85A6_
- https://gallery.flutter.dev/#/
- https://dart-tutorial.com/introduction-and-basics/
- https://yaz.in/assets/flutter/Flutter%20Cheat%20Sheet.pdf