DSC DTU

Flutter

Cross Platform App Development with Flutter

Developer Student Clubs

Topics Covered

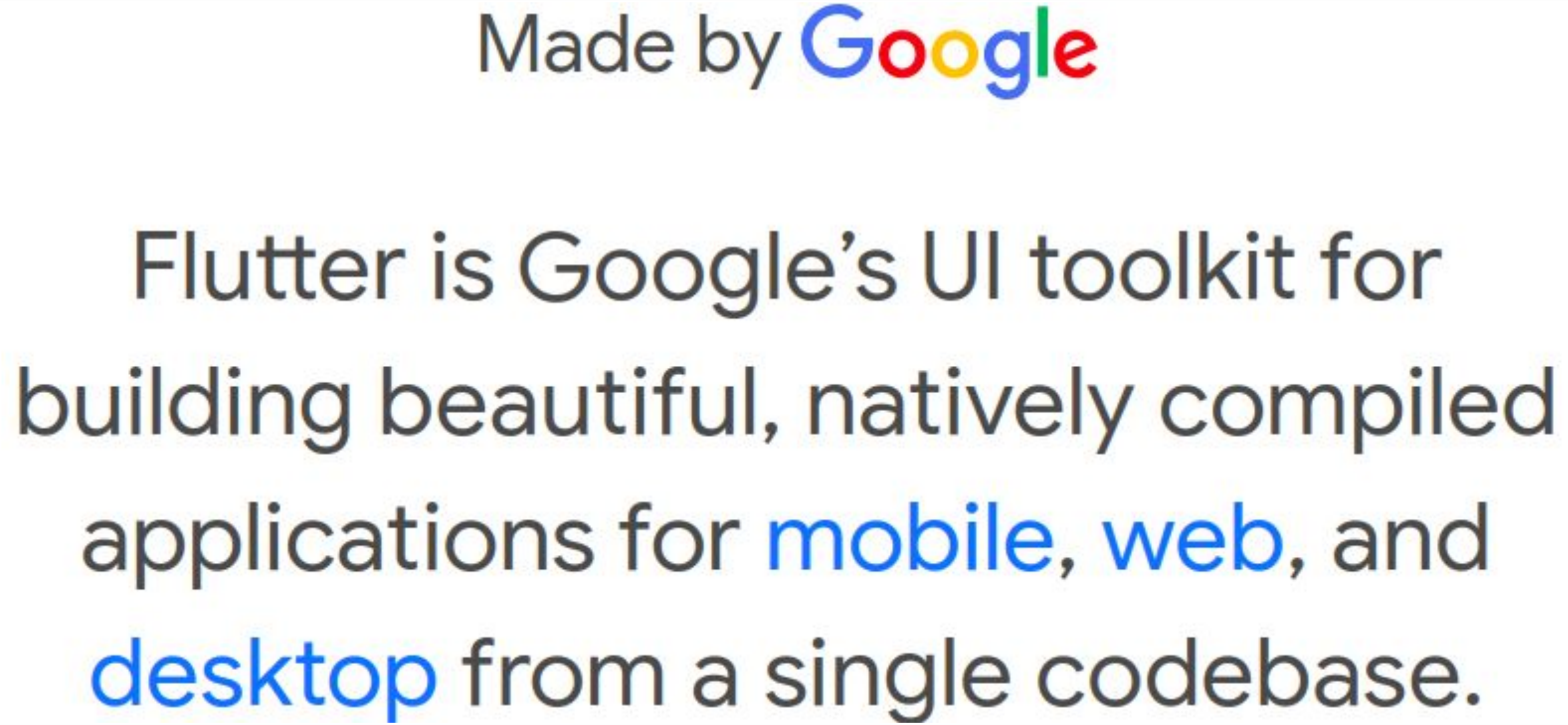
=== What is Flutter ===

======== A Quick Introduction to Dart ========

============ Flutter Overview ============

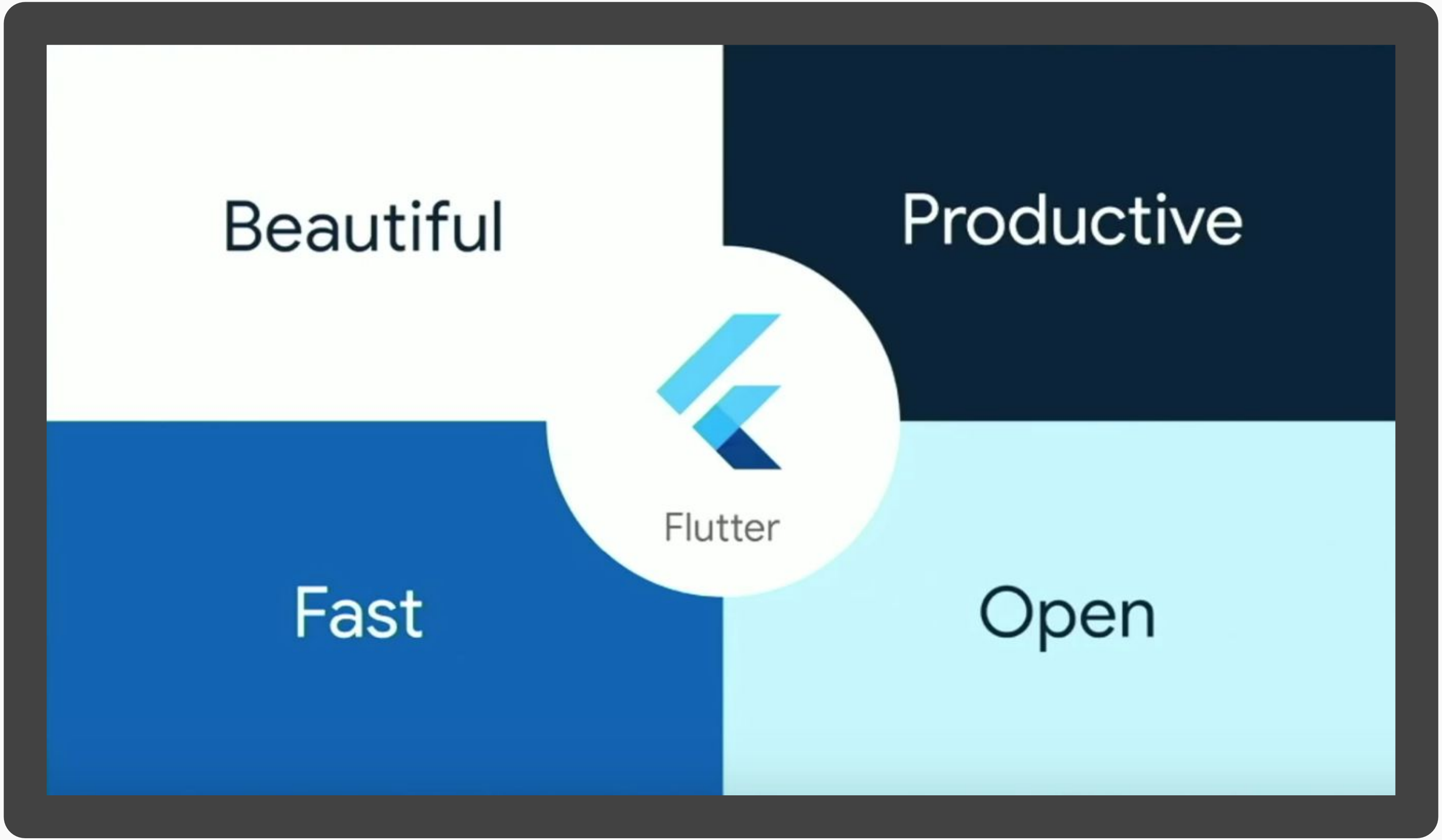
=== Your first app - A Reddit Clone ===

What is Flutter



Made by Google

Flutter is Google's UI toolkit for building beautiful, natively compiled applications for mobile, web, and desktop from a single codebase.



Beautiful

Productive

Flutter

Fast

Open

Creating Custom UI’s and Animations is easy in Flutter

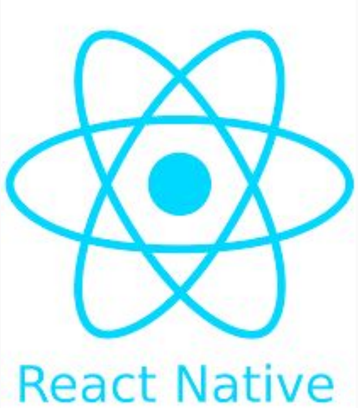
Popular Cross Platform App Development Frameworks

Flutter

Xamarin

ionic

NativeScript



React Native

Getting Started with Flutter

If you don't have Flutter installed,

You can use an online Flutter Editor for UI

Head over to this website

dartpad.dartlang.org/embed-flutter.html?sample_id=material.Scaffold.1

OR

bit.ly/2WFgTD4

Creating an App

Using Cmd line

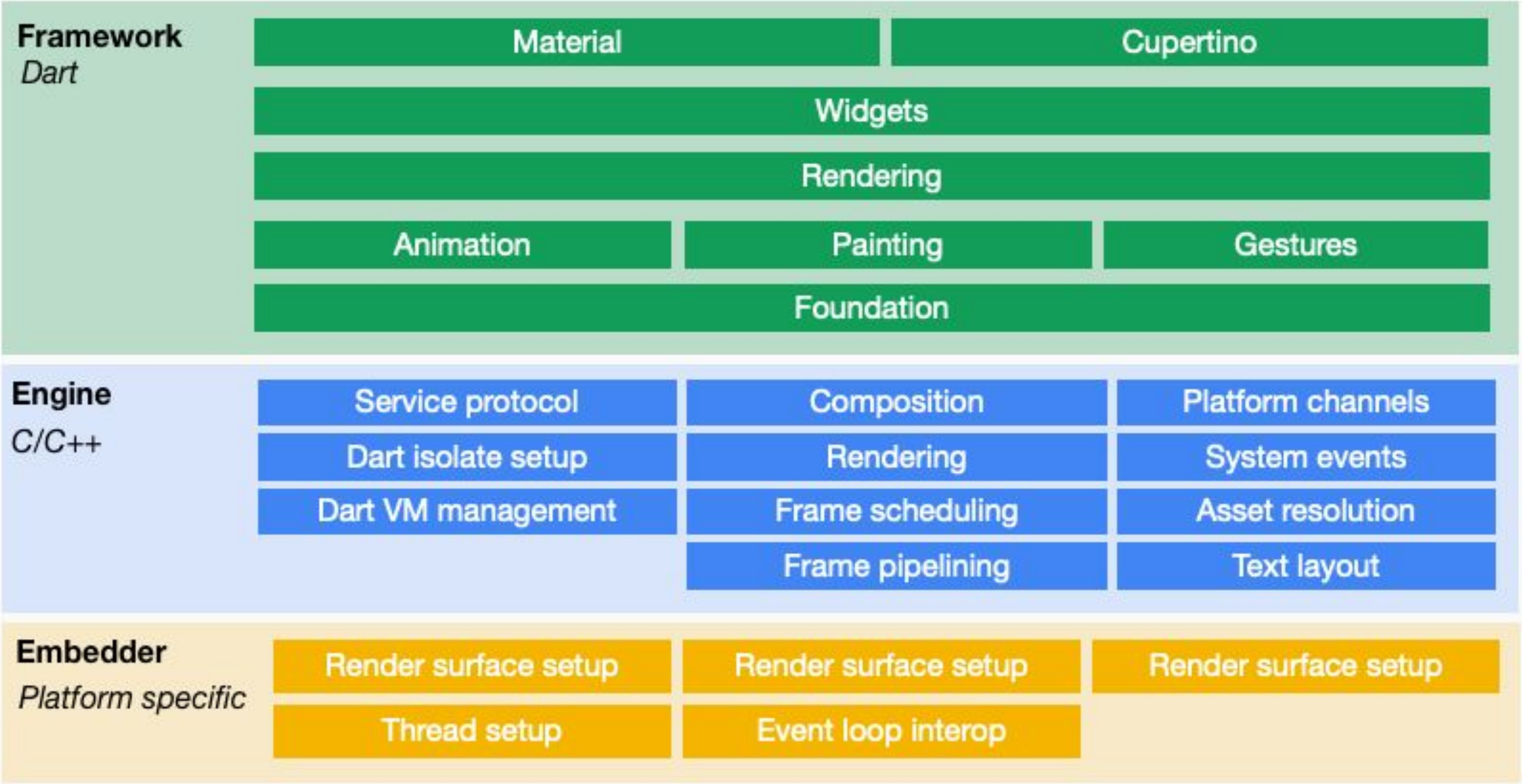
- Goto parent directory (‘say ~/projects)
- Run the following

```
flutter create --org "com.dscdtu"
-i objc -a java "example_app"
```

Using VSCode

- Ensure Flutter extension is installed
- Press `CTRL + SHIFT + P`
- Select `Flutter: New Project`
- Enter Name and Project Location

Flutter system overview



Framework
Dart

Material	Cupertino

Widgets

Rendering

Animation	Painting	Gestures

Foundation

Engine
C/C++

Service protocol	Composition	Platform channels
Dart isolate setup	Rendering	System events
Dart VM management	Frame scheduling	Asset resolution
	Frame pipelining	Text layout

Embedder
Platform specific

Render surface setup	Render surface setup	Render surface setup
Thread setup	Event loop interop	

Running an App

- Connect your phone and Turn on USB debugging in settings (if not already)
  - On the device, go to `Settings > About <device>`.
  - Tap the Build number => 7 times to make `Settings > Developer options` available.
  - Then enable the USB Debugging option
- Run the App
  - CMD line => `flutter run`
  - VSCode => `invoke Debug > Start Debugging or press F5.`

A Quick Intro to Dart

Language Key Points

- Everything is an Object, There are no primitive types. Everything inherits from the 'Object' Class.
- Dart is strongly typed, but it can infer types statically. For variables with *dynamic type*, use the keyword dynamic.
- Dart supports generic types, like List<int> or Map<String, dynamic>
- Similar to C++, Dart supports top level variables and functions

Language Key Points

- Code execution begins with the main() function.
- Unlike Java, Dart doesn’t have the keywords public, protected, and private. If an identifier starts with an underscore (_), it’s private.
- Identifiers can start with a letter or underscore (_), followed by any combination of those characters plus digits.
- Uninitialized variables have an initial value of null. (As of Dart 2.X)

Variables

```
int ca = 30;
String d = "This is a String";
// automatic static inference
var e = "This is a String Object";
const f = "This is a constant String Object"; // compile time constant

// equivalent to final String g = “This is a…”;
final g = "This is a final String Object";
// dynamic objects
dynamic h = 10; // it was an Int
h = "a String"; // but now it's a String
h = 10.02; // and now it's a Double
```

Functions

```
// function with optional positional parameter
int example1(int a, [int b = 10]) {
  return a + b;
}
// function with optional named parameter
String example2(int a, {String str1 = "default", String str2}) {
  return str1 + str2;
}
// lambda or arrow function (just a shorthand, returns a+b)
int add(int a, int b) => a + b;
//===========================================================
//===============  calling  functions  ======================
  example2(10, str2: "hello", str1: "world");
```

String Interpolation, Async Functions and Null aware operators

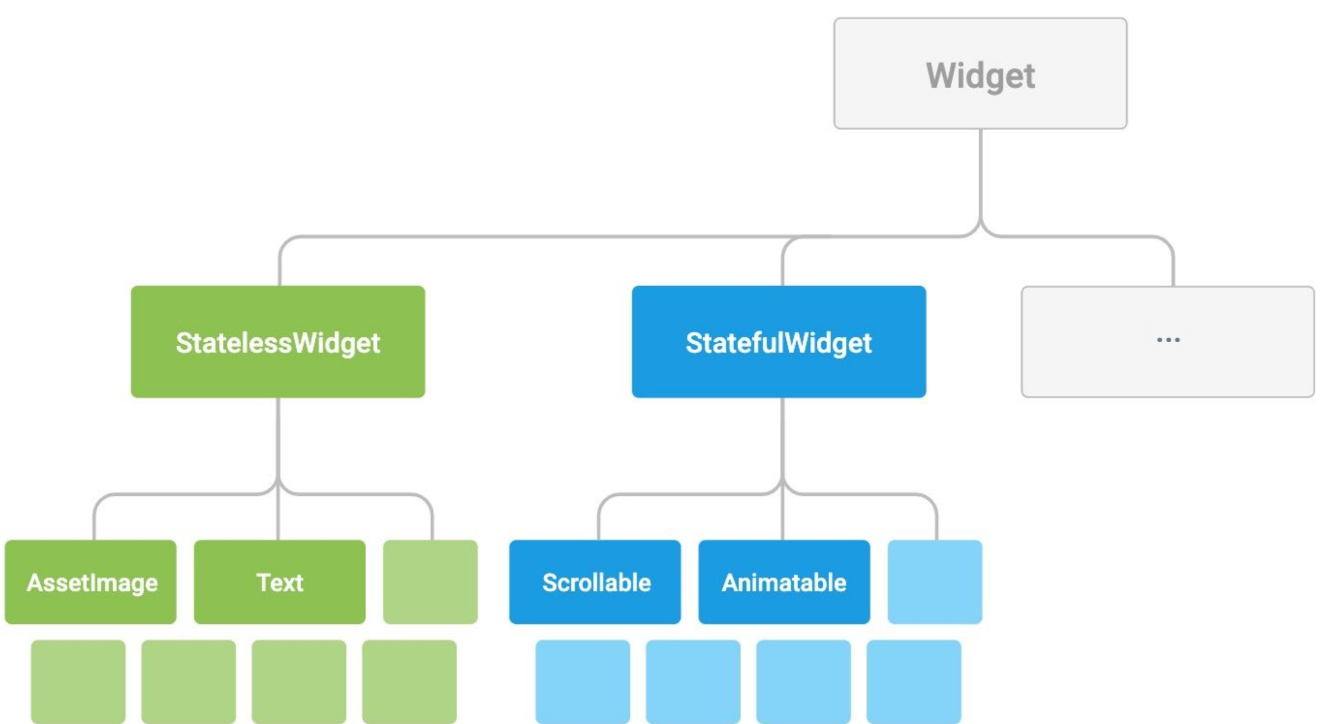
```
// string interpolation
var a = 10, b = "number";
String s = "${a.toString()} is a $b"; // prints: 10 is a number

// async function, called just like a normal function,
void delayPrint() async {
  await Future.delayed(Duration(seconds: 1));
  print("Sorry to keep u waiting");
}

// Null aware operators
a?.toString(); // only calls toString if a is not null
int x = a ?? 10; // sets x to a, or 10 if a is null
```

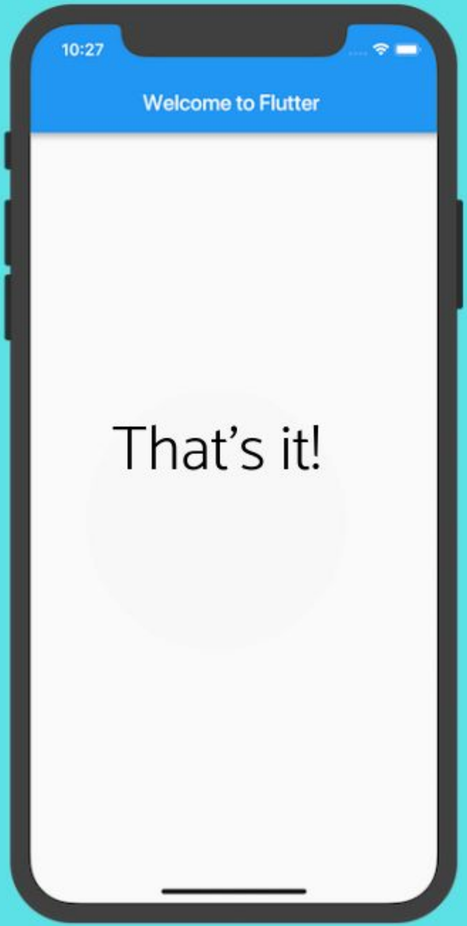
A Quick Overview of

Flutter

- Mobile
- Web
- Desktop
- Embedded

Widgets in Flutter





That’s it!

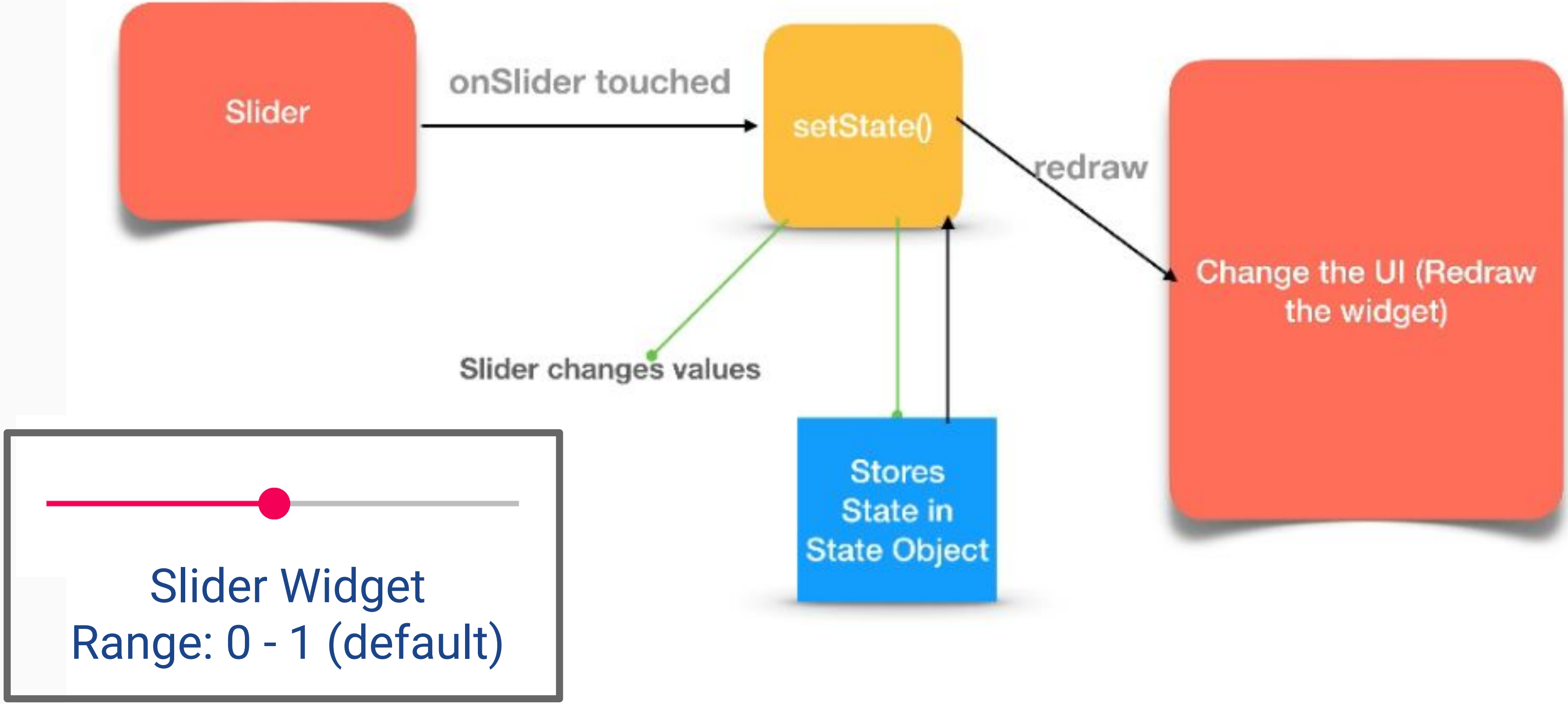
```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());


class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Welcome to Flutter',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Welcome to Flutter'),
        ),
        body: Center(
          child: Text('That's it!'),
        ),
      ),
    );
  }
}
```

Stateless and Stateful Widgets

```
class Car extends StatelessWidget {

  Car({ Key key }) : super(key:
key);

  @override
  Widget build(BuildContext context)
   {
    return Container(
    color: Colors.blue,
    );
  }
}
```

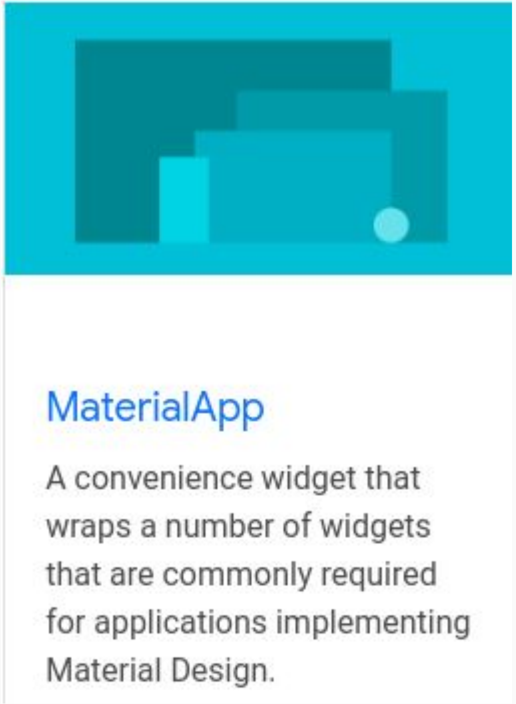
```
class Car extends StatefulWidget {
  Car({ Key key, @required child }) :
super(key: key);

  @override
  _CarState createState() => _CarState();
}

class CarState extends State<Car> {
  @override
  Widget build(BuildContext context) {
    return Container(
      color: Colors.blue,
      child: Text("Hello"),
    );
  }
}
```

Working of a Stateful Widget


Slider
onSlider touched
setState()
redraw
Change the UI (Redraw the widget)
Slider changes values
Stores State in State Object

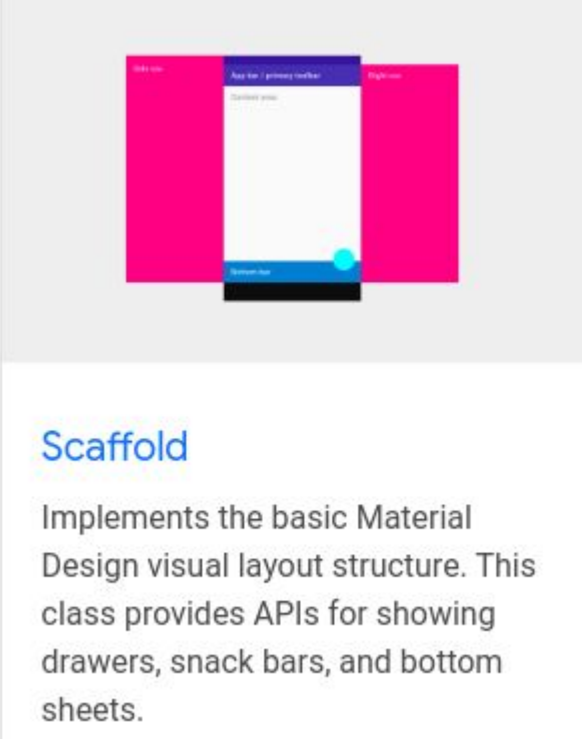
Slider Widget
Range: 0 - 1 (default)

Basic Widgets



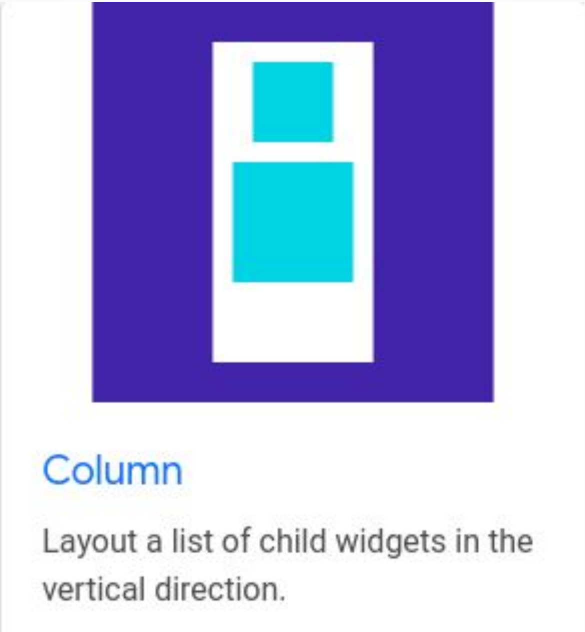
MaterialApp

A convenience widget that wraps a number of widgets that are commonly required for applications implementing Material Design.



Scaffold

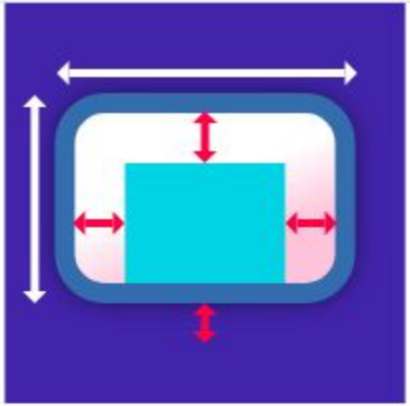
Implements the basic Material Design visual layout structure. This class provides APIs for showing drawers, snack bars, and bottom sheets.



Column

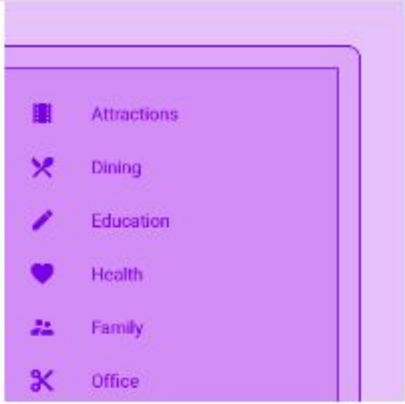
Layout a list of child widgets in the vertical direction.

Basic Widgets



Container

A convenience widget that combines common painting, positioning, and sizing widgets.



ListView

A scrollable, linear list of widgets. ListView is the most commonly used scrolling widget. It displays its children one after another in the scroll direction. In the cross axis, the children are required to fill the ListView.

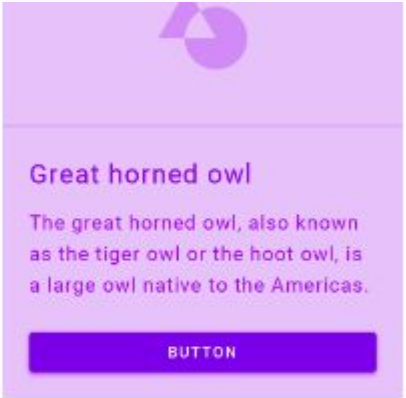
Text

A run of text with a single style.

Basic Widgets

Image

A widget that displays an image.



Great horned owl

The great horned owl, also known as the tiger owl or the hoot owl, is a large owl native to the Americas.

RaisedButton

A Material Design raised button. A raised button consists of a rectangular piece of material that hovers over the interface.

GestureDetector

A widget that detects gestures. Attempts to recognize gestures that correspond to its non-null callbacks. If this widget has a child, it defers to that child for its sizing behavior. If it does not have a child, it grows to fit the parent instead.

Lets make a Flutter APP

Reddit Clone

Link to Live Flutter-Web App and Full code at this repository

github.com/mannprerak2/quick_clone_reddit

Shortened url => bit.ly/33iyiUO