Experiment No : 02

Aim :

Theory :

Container class in flutter is a convenience widget that combines common painting, positioning, and sizing of widgets. A Container class can be used to store one or more widgets and position them on the screen according to our convenience. Basically, a container is like a box to store contents. A basic container element that stores a widget has a margin, which separates the present container from other contents. The total container can be given a border of different shapes, for example, rounded rectangles, etc. A container surrounds its child with padding and then applies additional constraints to the padded extent (incorporating the width and height as constraints, if either is non-null).

Properties of Container Class:

1. child: Container widget has a property 'child:' which stores its children. The child class can be any widget. Let us take an example, taking a text widget as a child.

2. color: The color property sets the background color of the entire container. Now we can visualize the position of the container using a background color.

3. height and width: By default, a container class takes the space that is required by the child. We can also specify the height and width of the container based on our requirements.

4. margin: The margin is used to create an empty space around the container. Observe the white space around the container. Here EdgeInsets.geometry is used to set the margin .all() indicates that the margin is present in all four directions equally.

5. padding: The padding is used to give space form the border of the container form its children. Observe the space between the border and the text.

6. alignment: The alignment is used to position the child within the container. We can align in different ways: bottom, bottom center, left, right, etc. here the child is aligned to the bottom center.

7. Decoration: The decoration property is used to decorate the box(e.g. give a border). This paints behind the child. Whereas foreground Decoration paints in front of a child. Let us give a border to the container. But, both color and border color cannot be given.

8. Transform: This property of the container helps us to rotate the container. We can rotate the container in any axis, here we are rotating in the z-axis.

9. Constraints: When we want to give additional constraints to the child, we can use this property.

10. ClipBehaviour: This property takes in Clip Enum as the object. This decides whether the content inside the container will be clipped or not.

11. Foreground Decoration: This parameter holds Decoration class as the object. It controls the decoration in front of the Container widget.

Code :

```
import 'dart:ui';

import 'package:flutter/material.dart';

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

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Exp2',
      theme: ThemeData(

        primarySwatch: Colors.deepOrange,
      ),
      home: const MyHomePage(title: 'Exp 2 Flutter Container'),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key, required this.title});



  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {




  @override
  Widget build(BuildContext context) {

    return Scaffold(
```

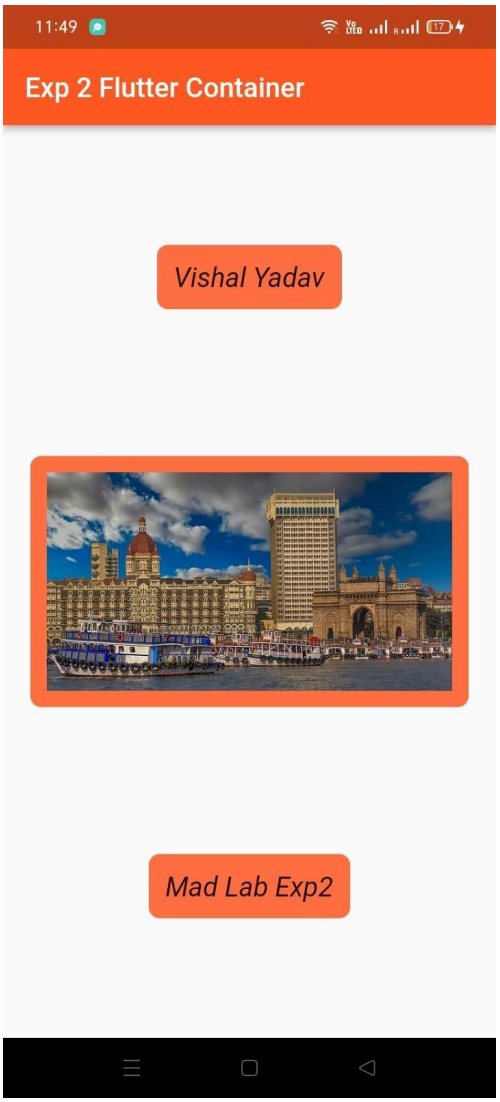
```
      appBar: AppBar(

        title: Text(widget.title),
      ),
      body: Column(
       mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: [
          Container(
            margin: EdgeInsets.all(20),
            padding: EdgeInsets.all(12),
            decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(8),
              color: Colors.deepOrangeAccent
             ),
            child: Text('Vishal Yadav',style: TextStyle(fontSize: 20,fontStyle:
FontStyle.italic),),
          ),
          Container(
            margin: EdgeInsets.all(20),
            padding: EdgeInsets.all(12),
            decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(8),
                color: Colors.deepOrangeAccent
             ),
            child: Image.asset('assets/img/img.png')
          ),
          Container(
            margin: EdgeInsets.all(20),
            padding: EdgeInsets.all(12),
            decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(8),
                color: Colors.deepOrangeAccent
             ),
            child: Text('Mad Lab Exp2',style: TextStyle(fontSize: 20,fontStyle:
FontStyle.italic),),
          )
        ],

      ),
      // This trailing comma makes auto-formatting nicer for build methods.
    );
  }
}
```

Pubspec.yaml file code :

```
name: exp2
description: A new Flutter project.

publish_to: 'none'
version: 1.0.0+1

environment:
  sdk: '>=2.18.6 <3.0.0'


dependencies:
  flutter:
    sdk: flutter

  cupertino_icons: ^1.0.2

dev_dependencies:
```

```
  flutter_test:
    sdk: flutter

  flutter_lints: ^2.0.0

flutter:

  uses-material-design: true

  assets:
     - assets/img/img.png
```

Output :



Conclusion : Therefore we have successfully created app with the help of containers