Parameter of Container :

Height , Width , Color

height and **width** define the dimensions of a widget, specifying its size along the vertical and horizontal axes, respectively,

The Container widget allows you to set properties such as height and width to control the size of the container.

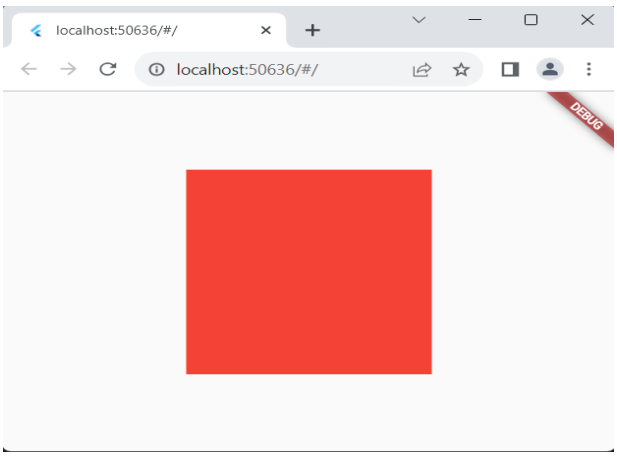
color:

Color defines the background color of the container.

Adjust values and color according to your layout and design preferences

```dart
@override
Widget build(BuildContext context) {
  return  MaterialApp(
    home: Scaffold(
      body: Center(
        child: Container(
          height:200,
          width: 200,
          color: Colors.red
        ) // Container
      ), // Center
    ), // Scaffold
  ); // MaterialApp
}
```

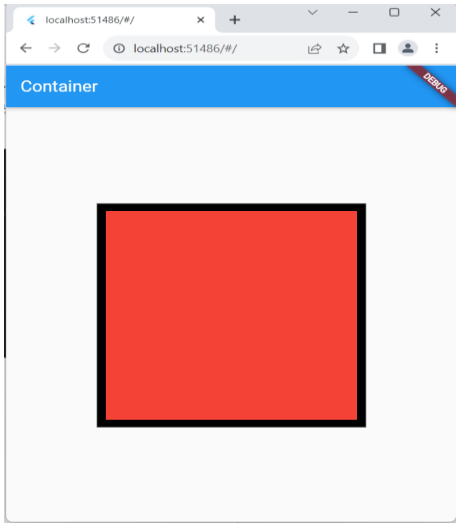
Output :



Padding –
The padding adds space around the child. It is used to create space or margins around other widgets.
We can add padding in horizontal and vertical directions using **EdgeInsets.symmetric().**

```
Container(
                height: 300,
                width: 300,
                color: Colors.black,
                padding: const EdgeInsets.symmetric(
                    vertical: 10,
                    horizontal: 10
                  ),
                child:Container(
                  height: 200,
                  width: 200,
                  color: Colors.red,
                )
              )
```

Output

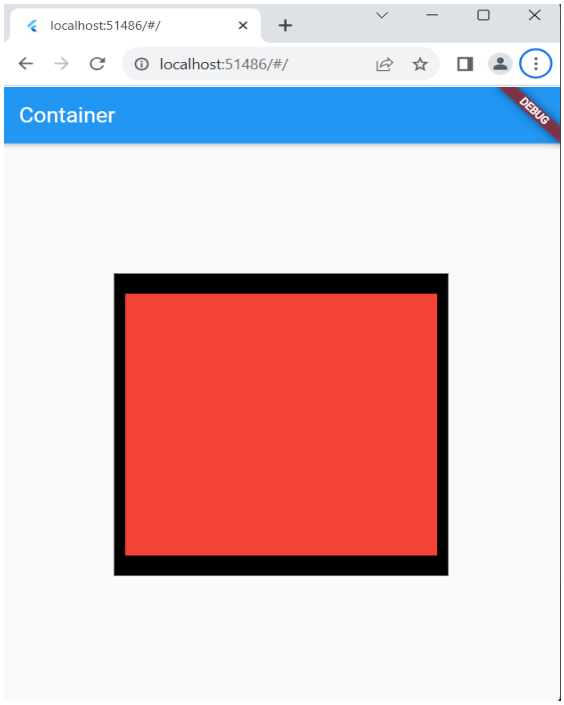


To add padding to a specific direction use EdgeInsets.only().

EdgeInsets.only().

```
Container(
                height: 300,
                width: 300,
                color: Colors.black,
                padding: const EdgeInsets.only(
                  left: 10,
                  right: 10,
                  top:20,
                  bottom: 20
                ),
                child:Container(
                  height: 200,
                  width: 200,
                  color: Colors.red,
                )
              )
```

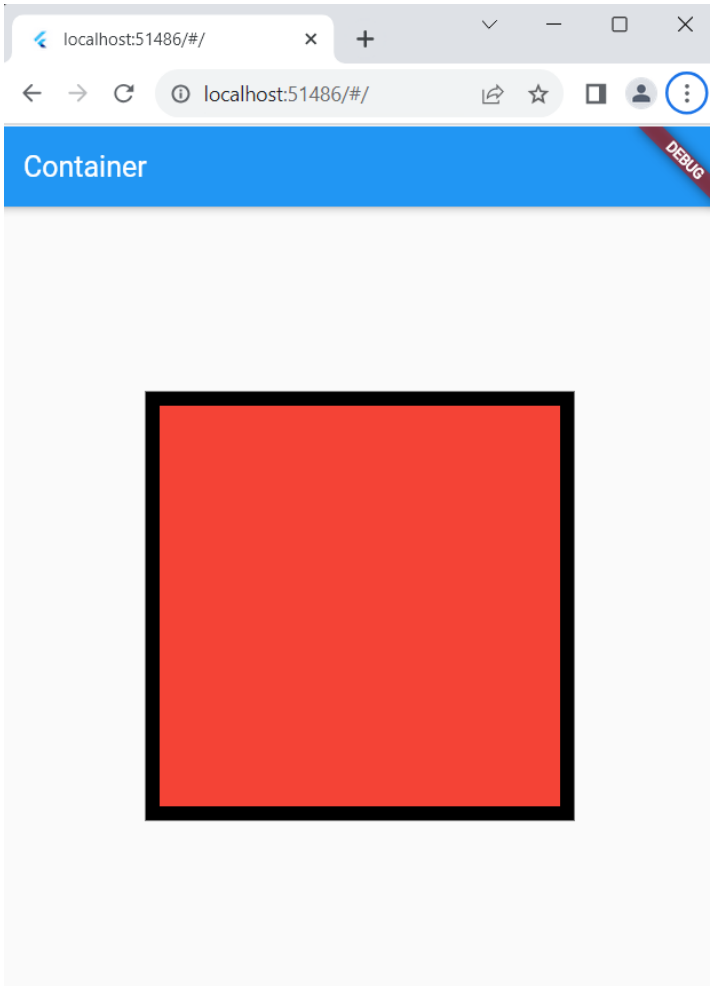
Output -



EdgeInsets.all().

```
Container(
                height: 300,
                width: 300,
                color: Colors.black,
                padding:const EdgeInsets.all(10),
                child:Container(
                  height: 200,
                  width: 200,
                  color: Colors.red,
                )
              )
```

Output -

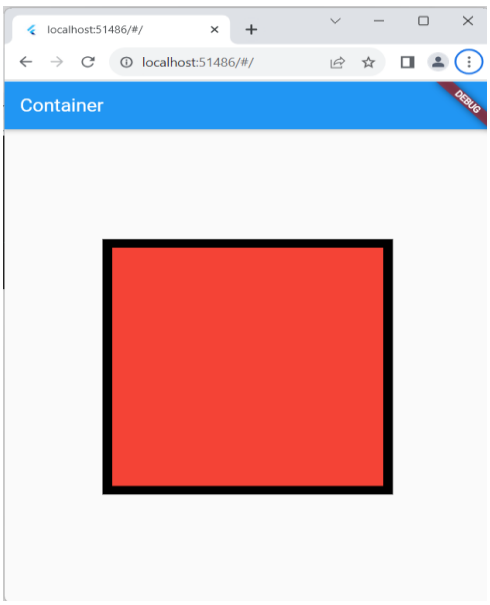


margin:-
margin is a property used to add space outside a widget's border. It represents the space
between the widget's border and the surrounding widgets or the edge of the screen.
Similar to padding we have multiple options to give margin to Container i.e

EdgeInsets.only(), EdgeInsets.all(), EdgeInsets.symmetric()

```
Container(
                height: 300,
                width: 300,
                color: Colors.black,

                child:Container(
                  height: 200,
                  width: 200,
                  color: Colors.red,
                  margin:const EdgeInsets.all(10),
                )
              )
```

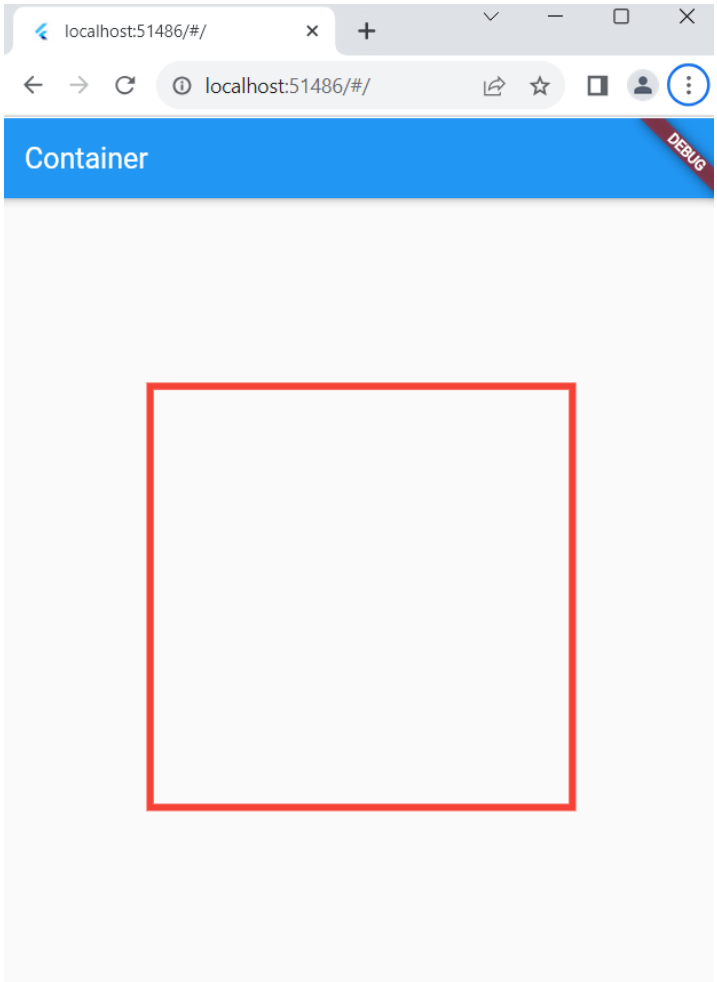
Output -



decoration:-
decoration property is used to apply visual styling to a widget, typically a Container. It
takes a BoxDecoration object that defines various visual elements such as **color, border**

```
Container(
                height: 300,
                width: 300,
                decoration:BoxDecoration(
                  border:Border.all(
                    color: Colors.red,
                    width: 5
                  )
                )
              )
```

Output -



borderRadius:

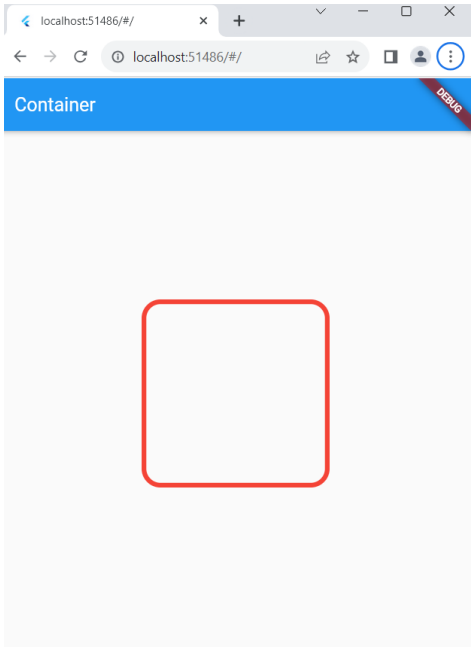
To set the border to the Container we can use **Border.all() , Border.symmetric() , Border().**

We can set the border color, and border width using the Border class.

To give the border to a particular side we can use the **Border().**

```
Container(
                height: 200,
                width: 200,
                decoration:BoxDecoration(

                  borderRadius: const BorderRadius.all(
                    Radius.circular(20)
                  ),

                  border:Border.all(
                    color: Colors.red,
                    width: 5
                  )
                )
              )
```

Output -

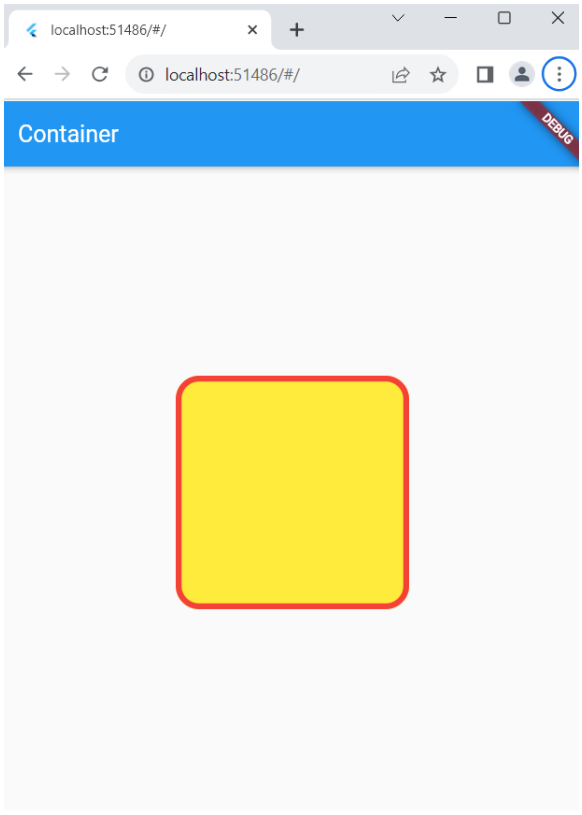


color:

This parameter is used to color the Container inside the decoration .

```
Container(
                height: 200,
                width: 200,
                decoration:BoxDecoration(
                  color:Colors.yellow,

                  borderRadius: const BorderRadius.all(
                    Radius.circular(20)
                  ),

                  border:Border.all(
                    color: Colors.red,
                    width: 5
                  )
                )
              )
```

Output -



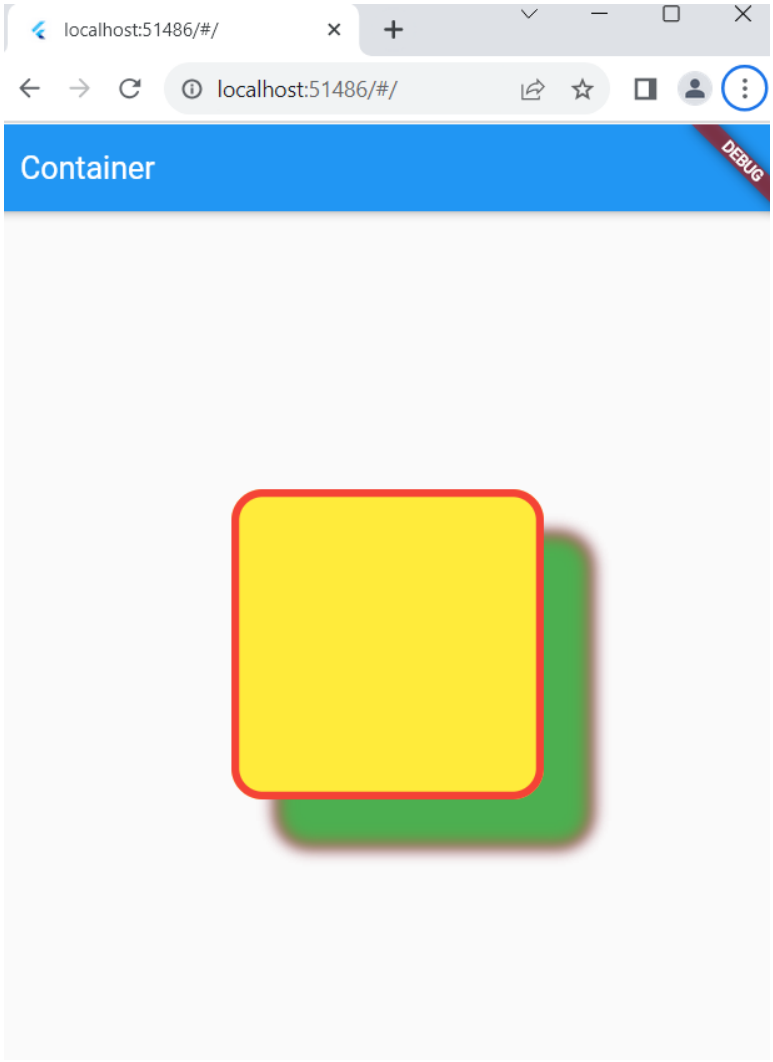
boxShadow:
It takes a List of BoxShadow. Using this we can display the shadow behind the
container. We can specify the color, offset(Position) and blurRadius in the BoxShadow.
Positive x/y offsets will shift the shadow to the right and down, while

negative offsets shift the shadow to the left and up.
The first item in the list will represent the last layer of shadow.

```
Container(
                height: 200,
                width: 200,
                decoration:BoxDecoration(
                  color:Colors.yellow,

                  borderRadius: const BorderRadius.all(
                    Radius.circular(20)
                  ),

                  border:Border.all(
                    color: Colors.red,
                    width: 5
                  ),

                  boxShadow: const[
                    BoxShadow(
                      color:Colors.purple,offset:Offset(30,
30),blurRadius: 8
                    ),
                     BoxShadow(
                      color:Colors.red,offset:Offset(30,
30),blurRadius: 8
                    ),
                     BoxShadow(
                      color:Colors.green,offset:Offset(30,
30),blurRadius: 8
```

```
                ),

            ]
        )
    )
```

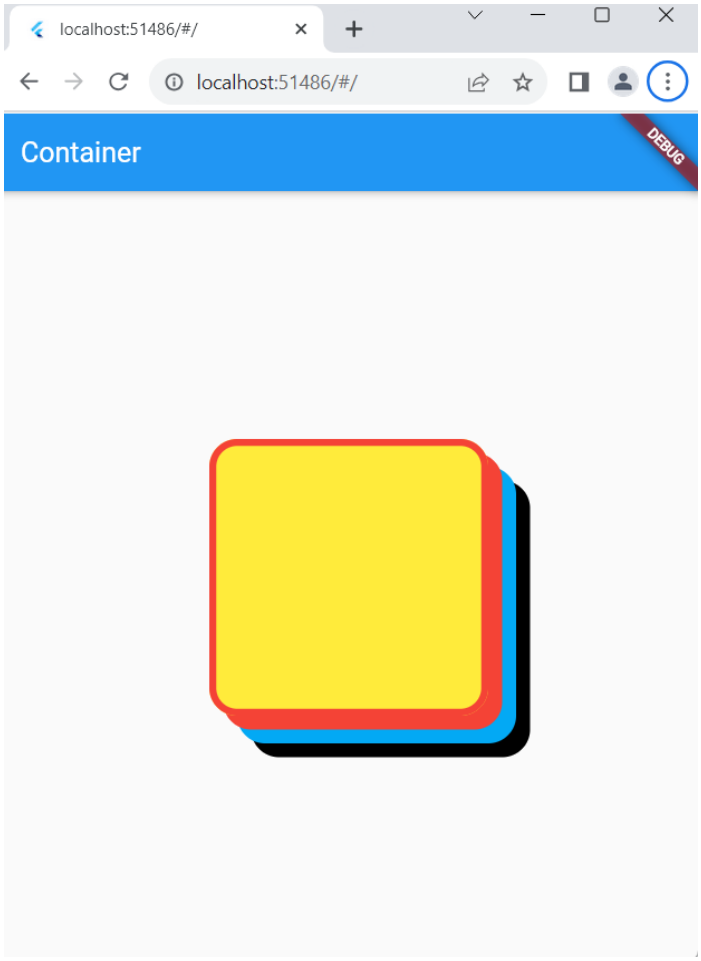
Output -



Boxshadow without blur -

```
boxShadow: const[
 BoxShadow(
    color:Colors.black,offset:Offset(30, 30),blurRadius: 0
 ),
 BoxShadow(
   color:Colors.lightBlue ,offset:Offset(20, 20),blurRadius: 0
 ),
 BoxShadow(
   color:Colors.red,offset:Offset(10, 10),blurRadius: 0
 ),
],
```

Output-



gradient:
Gradient refers to the transition between two or more colors.
We can give a gradient to the Container using
LinearGradient(), SweepGradient(), RadialGradient
In the Linear Gradient, we need to specify the List of colors to be used to display the gradient.
We can also specify the position to place the color using the stops parameter.
The values in the stop parameter range between 0.0 to 1.0.
The values in the stops list must be in ascending order.
If a value in the [stops] list is less than an earlier value in the list, then its value is assumed to equal the previous value.

```
Container(
                height: 200,
                width: 200,
                decoration:BoxDecoration(

                  color:Colors.yellow,

                  borderRadius: const BorderRadius.all(
                    Radius.circular(20)
                  ),

                  border:Border.all(
                    color: Colors.red,
                    width: 5
                  ),

                  gradient: const LinearGradient(
                    stops: [0.3,0.5],
                    colors: [Colors.red,Colors.green]
                  )
                )
              )
```

Output -