Container

Parameter of Container:

Height & width:

Height and width define the dimension of 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 preference.

```
child: Container(
  height: 200,
  width: 200,
  color: Colors.red,
), // Container
```

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 EdgeInsects.symmetric().

```
child: Container(
  padding: const EdgeInsets.symmetric(
    horizontal: 10,
    vertical: 10,
  ), // EdgeInsets.symmetric
```

To add padding to a specific direction use EdgeInsects.only().similarly to add padding to all the ditrection use EdgeInsects.all().

```
body: Center(
  child: Container(
    padding: const EdgeInsets.only(
      left: 10,
      right: 10,
      top: 10,
      bottom: 10,
    ), // EdgeInsets.only
    width: 100,
    height: 100,
    color: Colors.blue,
  ), // Container
), // Center
```

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()

```
body: Center(
  child: Container(
    margin: const EdgeInsets.only(
      left: 10,
      bottom: 10,
      right: 10,
      top: 10,
    ), // EdgeInsets.only
    width: 100,
    height: 100,
    color: Colors.red,
  ), // Container
), // Center
```

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

```
body: Center(
  child: Container(
    width: 300,
    height: 300,
    decoration: BoxDecoration(
      border: Border.all(
        color: Colors.yellow,
        width: 5,
      ), // Border.all
    ), // BoxDecoration
  ), // Container
```

OUTPUT:

BorderRadius:

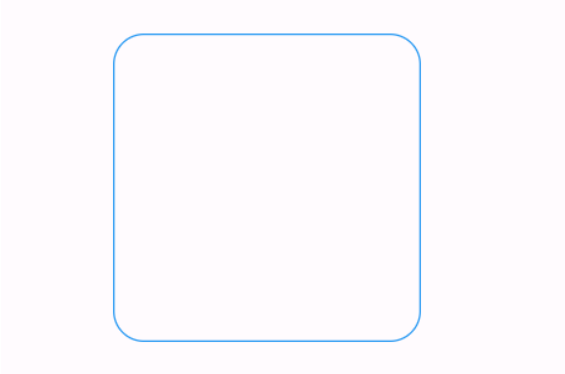
To set the border to the Container we can use Border.all() , Border.symmetric() , Border().

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().

```
body: Center(
  child: Container(
    width: 200,
    height: 200,
    decoration: BoxDecoration(
      borderRadius: const BorderRadius.all(
        Radius.circular(20),
      ), // BorderRadius.all
      border: Border.all(
        color: Colors.blue,
      )), // Border.all // BoxDecoration
  ), // Container
), // Center
```

OUTPUT:

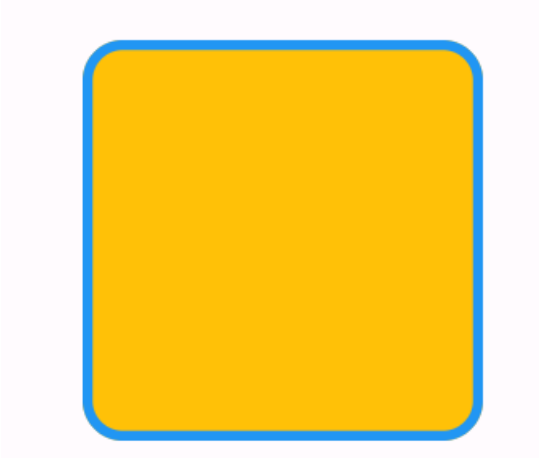


Color:

This parameter is used to color Container insise decoration.

```
body: Center(
  child: Container(
    width: 200,
    height: 200,
    decoration: BoxDecoration(
      color: Colors.amber,
      borderRadius: const BorderRadius.all(
        Radius.circular(20),
      ), // BorderRadius.all
      border: Border.all(
        color: Colors.blue,
        width: 5,
      ), // Border.all
    ), // BoxDecoration
  ), // Container
), // Center
```

OUTPUT:



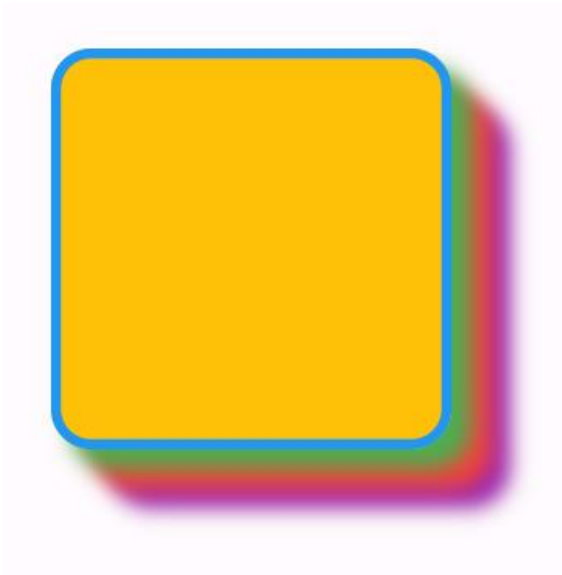
BoxShadow:

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.

```
body: Center(
  child: Container(
    width: 200,
    height: 200,
    decoration: BoxDecoration(
      color: Colors.amber,
      borderRadius: const BorderRadius.all(Radius.circular(20)),
      border: Border.all(color: Colors.blue, width: 5),
      boxShadow: const [
        BoxShadow(
          color: Colors.purple, offset: Offset(30, 30), blurRadius: 8), // BoxShadow
        BoxShadow(
          color: Colors.red, offset: Offset(20, 20), blurRadius: 8), // BoxShadow
        BoxShadow(
          color: Colors.green, offset: Offset(10, 10), blurRadius: 8), // BoxShadow
      ],
    ), // BoxDecoration
  ), // Container
), // Center
```

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 colorsto 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.

```
body: Center(
  child: Container(
    width: 300,
    height: 300,
    decoration: BoxDecoration(
      color: Colors.amber,
      border: Border.all(color: Colors.blue, width: 5),
      borderRadius: const BorderRadius.all(Radius.circular(20)),
      gradient: const LinearGradient(
        stops: [0.3, 0.5],
        colors: [Colors.red, Colors.green],
      )), // LinearGradient // BoxDecoration
  ), // Container
), // Center
```

OUTPUT: