

Flutter 容器类 Widget

容器类Widget 不同于 布局类Widget:

1. 首先，布局类Widget 的子Widget 一般都是数组，而容器类Widget 的子Widget 一般只有一个。
2. 布局类Widget 是按照一定的排列方式对其 子Widget 进行排列，而 容器类Widget 用于嵌套其他 Widget，对 Widget 添加一些修饰（补白或背景色等）、变换(旋转或剪裁等)、或限制(大小等)。

在前面使用 UI Widget 的时候，想必你已经发现，大部分 UI Widget 都不能指定宽高、设置内边距和外边距，这时候就需要使用 容器类Widget 了。

Padding

Padding 通过给自己指定内边距来添加 子Widget。

代码所在位置

flutter_widget_demo/lib/containers/PaddingWidget.dart

Padding 的快速上手

Padding 的必填参数是 padding，然后其 child 参数就是你想要添加 padding 的 Widget，例如：

```
Padding(  
  padding: EdgeInsets.all(100),  
  child: Text('Hello Flutter'),  
)
```

Padding 在一个页面使用的完整 Demo 如下:

```
import 'package:flutter/material.dart';  
  
main() => runApp(new PaddingWidget());  
  
class PaddingWidget extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return new MaterialApp(  
      title: 'Test',  
      home: new Scaffold(  
        appBar: new AppBar(title: new  
Text('Flutter 容器Widget -- Padding')),  
        body: Padding(  
          padding: EdgeInsets.all(100),  
          child: Text('Hello Flutter'),  
        )),  
    );  
  }  
}
```

运行效果如下:

Padding 的构造函数及参数说明

Padding 的构造函数为:

```
class Padding extends
SingleChildRenderObjectWidget {
  const Padding({
    Key key,
    @required this.padding,
    Widget child,
  }) : assert(padding != null),
        super(key: key, child: child);
  ...
}
```

参数名字	参数类型	意义	必选 or 可选
key	Key	Widget 的标识	可选
padding	EdgeInsetsGeometry	容器内边距	必选
child	Widget	容器里显示的 Widget	可选

padding：容器内边距

padding 的类型是 EdgeInsetsGeometry，EdgeInsetsGeometry 是抽象类，我们一般使用 EdgeInsets：

EdgeInsets 的值	含义
EdgeInsets.all(double value)	上、下、左、右 边距都一样
EdgeInsets.only({this.left = 0.0,this.top = 0.0,this.right = 0.0,this.bottom = 0.0})	可以单独指定一个的边距
EdgeInsets.symmetric({double vertical = 0.0,double horizontal = 0.0,})	vertical 的值是上、下边距， horizontal 是左右边距的值

Container

Container 是一个拥有绘制、定位、调整大小的 Widget。

代码所在位置

flutter_widget_demo/lib/containers/ContainerWidget.dart

Container 的快速上手

Container 可以为 Widget 添加大小、背景等各种参数，其 child 参数就是你想要添加 Container 的 Widget，例如：

```
Container(  
  child:  
    ...  
)
```

Container 在一个页面使用的完整 Demo 如下：

```
import 'package:flutter/material.dart';  
  
main() => runApp(new ContainerWidget());  
  
class ContainerWidget extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return new MaterialApp(  
      title: 'Test',  
      home: new Scaffold(  
        appBar:  
          new AppBar(title: new
```

```
Text('Flutter 容器Widget -- Container')),
      body: Container(
        margin: EdgeInsets.only(top: 50.0,
left: 120.0), //容器外补白
        constraints:
          BoxConstraints.tightFor(width:
200.0, height: 150.0), //卡片大小
        decoration: BoxDecoration(
          //背景装饰
          gradient: RadialGradient(
            //背景径向渐变
            colors: [Colors.green,
Colors.blue],
            center: Alignment.topLeft,
            radius: .98),
          boxShadow: [
            //卡片阴影
            BoxShadow(
              color: Colors.black54,
              offset: Offset(2.0, 2.0),
              blurRadius: 4.0)
          ],
        transform: Matrix4.rotationZ(.2),
//卡片倾斜变换
        alignment: Alignment.center, //卡片
内文字居中
        child: Text(
          //卡片文字
          "Hello Flutter",
          style: TextStyle(color:
Colors.white, fontSize: 40.0),
        ),
      )),);
```

```
}  
}
```

运行效果如下：



Container 的构造函数及参数说明

Container 的构造函数为：

```

class Container extends StatelessWidget {
  Container({
    Key key,
    this.alignment,
    this.padding,
    Color color,
    Decoration decoration,
    this.foregroundDecoration,
    double width,
    double height,
    BoxConstraints constraints,
    this.margin,
    this.transform,
    this.child,
  }) :
    ...
}

```

参数名字	参数类型	意义	必选 or 可选
key	Key	Widget 的标识	可选
alignment	AlignmentGeometry	容器内 child 的对齐方式	可选
padding	EdgeInsetsGeometry	容器内边距	可选
color	Color	容器的背景色	可选
decoration	Decoration	容器的背景装饰	可选
foregroundDecoration	Decoration	容器的前景装饰	可选

foregroundColor	Color	装饰	可选
width	double	容器的宽度	可选
height	double	容器的高度	可选
constraints	BoxConstraints	容器的大小限制	可选
margin	EdgeInsetsGeometry	容器外边距	可选
transform	Matrix4	容器的变化	可选
child	Widget	容器里显示的 Widget	可选

Align

Align 可以控制其子Widget的对齐方式，并可以根据子Widget的大小自动调整自己的大小。

代码所在位置

flutter_widget_demo/lib/containers/AlignWidget.dart

Align 的快速上手

Align 可以为 Widget 添加对齐方式，其 child 参数就是你想要添加 Align 的 Widget，例如：


```
Align(  
  alignment: Alignment.topRight,  
  child: Text(  
    'Hello Flutter',  
    style: TextStyle(color: Colors.red, fontSize:  
50),  
  ),  
)
```

Align 在一个页面使用的完整 Demo 如下:

```
import 'package:flutter/material.dart';  
  
main() => runApp(new AlignWidget());  
  
class AlignWidget extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return new MaterialApp(  
      title: 'Test',  
      home: new Scaffold(  
        appBar: new AppBar(title: new  
Text('Flutter 容器Widget -- Align')),  
        body: Align(  
          alignment: Alignment.topRight,  
          child: Text(  
            'Hello Flutter',  
            style: TextStyle(color:  
Colors.red, fontSize: 50),  
          ),  
        )),  
    );  
  }  
}
```

运行效果为：

Align 的构造函数及参数说明

Align 的构造函数为：

```
class Align extends SingleChildRenderObjectWidget
{
  const Align({
    Key key,
    this.alignment = Alignment.center,
    this.widthFactor,
    this.heightFactor,
    Widget child
  }) : assert(alignment != null),
        assert(widthFactor == null || widthFactor
>= 0.0),
        assert(heightFactor == null ||
heightFactor >= 0.0),
        super(key: key, child: child);
    ...
}
```

参数名字	参数类型	意义	必选 or 可选 可选 可选 可选
key	Key	Widget 的标识	可选
alignment	Alignment	容器内 child 的对齐方式	可选

widthFactor	double	宽度因子。如果没有设置，则 Align 的宽度就是match_parent；如果为非null，则将容器的宽度设置为 子Widget的宽度 乘以此宽度因子 值必须 ≥ 0	可选
heightFactor	double	高度因子。如果没有设置，则 Align 的高度就是match_parent；如果为非null，则将容器的高度设置为 子Widget的高度 乘以此高度因子 值必须 ≥ 0	可选
child	Widget	容器里显示的 Widget	可选

Center

Center 可以将其 子Widget 居中显示在自身内部。Center 继承自 Align，其实就是 alignment 为 Alignment.center 的 Align。

代码所在位置

flutter_widget_demo/lib/containers/CenterWidget.dart

Center 的快速上手

Center 可以让 Widget 居中，其 child 参数就是你想要居中的 Widget，例如：

```
Center(  
  child: Text(  
    'Hello Flutter',  
    style: TextStyle(color: Colors.red, fontSize:  
50),  
  ),  
)
```

Center 在一个页面使用的完整 Demo 如下：

```
import 'package:flutter/material.dart';  
  
main() => runApp(new CenterWidget());  
  
class CenterWidget extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return new MaterialApp(  
      title: 'Test',  
      home: new Scaffold(  
        appBar: new AppBar(title: new  
Text('Flutter 容器Widget -- Center')),  
        body: Center(  
          child: Text(  
            'Hello Flutter',  
            style: TextStyle(color:  
Colors.red, fontSize: 50),  
          ),  
        )),  
    );  
  }  
}
```

运行效果为：

12:41



Flutter 容器Widget -- Center

Hello Flutter

Center 的构造函数及参数说明

Center 的构造函数为：

```
class Center extends Align {  
    /// Creates a widget that centers its child.  
    const Center({ Key key, double widthFactor,  
double heightFactor, Widget child })  
        : super(key: key, widthFactor: widthFactor,  
heightFactor: heightFactor, child: child);  
}
```

参数名字	参数类型	意义	必选 or 可选
key	Key	Widget 的标识	可选
widthFactor	double	宽度因子。如果没有设置，则 Align 的宽度就是match_parent；如果为 非null，则将容器的宽度设置为 子Widget的宽度乘以此宽度因子 值必须>=0	可选
heightFactor	double	高度因子。如果没有设置，则 Align 的高度就是match_parent；如果为 非null，则将容器的高度设置为 子Widget的高度乘以此高度因子 值必须>=0	可选

child

Widget 容器里显示的 Widget

可选