Widget组件（二）

2021年6月30日 10:26

一、 Basics Widget

Basics Widget，主要有以下几种：

1. Container：设置Widget的背景、尺寸、定位等
2. Row：在水平方向上布置子窗口Widget列表
3. Column：在垂直方向上布置子窗口Widget列表
4. Image：显示图像
5. Text：文本
6. Icon：符合Material Design设计规范的图标
7. RaisedButton（现在叫ElevatedButton）：符合Material Design设计规范的凸起按钮
8. Scaffold：实现Basics的Material Design布局结构
9. APPbar：Material Design的应用栏
10. FlutterLogo：以Widget形式展示一个Flutter图标，可以调整样式
11. PlaceHolder：绘制一个框，占位用途

Material Design（又叫做“材料设计”，以下简称MD），是由谷歌创建和发表的一种设计语言，旨在把物理世界的体验带进屏幕，其中去掉现实中的杂质和随机性，仅保留其最原始纯净的形态、变化与过渡，最终还原最贴近真实的用户体验，达到一种简洁与直观的效果。

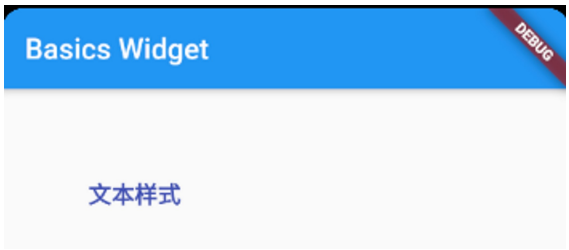
二、例子

2.1 文本

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

voidmain(){
runApp(MyApp());
}
classMyAppextendsStatelessWidget{
constMyApp({Key?key}):super(key:key);

@override
Widgetbuild(BuildContextcontext){
returnMaterialApp(
title:'WelcometoFlutter',
home:Scaffold(
appBar:AppBar(
title:Text('BasicsWidget'),
),
body:Padding(
padding:EdgeInsets.all(60.0),
child:Text(
'文本样式',
style:TextStyle(
fontSize:16.0,
color:Colors.indigo,
fontStyle:FontStyle.normal,
```

```
fontWeight:FontWeight.bold
),
),
),
),
);
}
}
```

结果为



2.2 图片

Image的构造函数有很多种：

1. new Image：从ImageProvider获取图片
2. new Image.asset：使用key从AssetBundle获取图片
3. new Image.network：加载网络图片
4. new Image.file：从文件中获取图片
5. new Image.memory：用于从Uint8List获取图片

Image主要的属性为fit，用于表示图片的填充模式，参数类型为BoxFit，可取以下值：

1. contain：按原比例全图显示
2. cover：全图填充，可能拉伸或裁减
3. fill：全图显示
4. fitHeight：高度填充
5. fitWidth：宽度填充
6. none：保持图片原始大小，裁减掉位于目标外的图片
7. scaleDown：缩小时等价于contain，否则等价于none

2.3 凸起按钮

可通过onPressed来回调按钮的点击

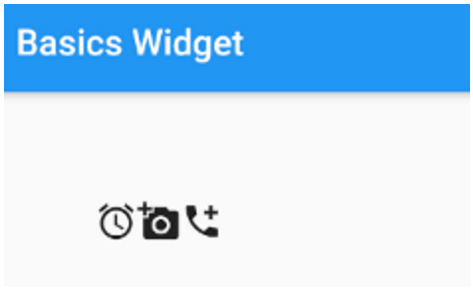
```
import'package:flutter/cupertino.dart';
import'package:flutter/material.dart';

voidmain(){
runApp(MyApp());
}

classMyAppextendsStatelessWidget{
constMyApp({Key?key}):super(key:key);

@override
```

```
Widgetbuild(BuildContextcontext){
returnMaterialApp(
title:'WelcometoFlutter',
home:Scaffold(
appBar:AppBar(
title:Text('BasicsWidget'),
),
body:Padding(
padding:EdgeInsets.all(60.0),
child:ElevatedButton(
onPressed:()=>print('onPressed'),
child:Text(
'onPress',
style:TextStyle(fontSize:10),
)),
)),
);
}
}
```

2.4 Row

Row可以用于在水平方向显示数组中的子元素Widget

```
@override
Widgetbuild(BuildContextcontext){
returnMaterialApp(
title:'WelcometoFlutter',
home:Scaffold(
appBar:AppBar(
title:Text('BasicsWidget'),
),
body:Padding(
padding:EdgeInsets.all(60.0),
child:Row(
children:<Widget>[
Icon(Icons.access_alarm),
Icon(Icons.add_a_photo),
Icon(Icons.add_call)
],
))));
}
```



2.5 Column和Expanded

Column用法和Row一样，可以使用Expanded配合Row和Column使用，用来填充剩余空间

```
@override
Widgetbuild(BuildContextcontext){
returnMaterialApp(
title:'WelcometoFlutter',
home:Scaffold(
appBar:AppBar(
title:Text('BasicsWidget'),
),
body:Padding(
padding:EdgeInsets.all(60.0),
child:Row(
children:<Widget>[
Icon(Icons.access_alarm),
Icon(Icons.add_a_photo),
Icon(Icons.add_call),
Expanded(child:FittedBox(fit:BoxFit.contain,child:constFlutterLogo(),),fl
ex:2,),
Expanded(child:Text("剩余三分之一"),flex:1,)
],
))));
}
```

其中，flex可以调整Expanded之间的占比

2.6 Container

可以设置背景、尺寸、定位

```
Widgetbuild(BuildContextcontext){
returnMaterialApp(
title:'WelcometoFlutter',
home:Scaffold(
appBar:AppBar(
title:Text('BasicsWidget'),
),
body:Padding(
padding:EdgeInsets.all(60.0),
child:Container(
decoration:BoxDecoration(color:Colors.lightGreen),
child:Text('Container'),
padding:EdgeInsets.all(36.0),
margin:EdgeInsets.all(10.0),
)
)));
}
```