

# 使用主题共享颜色和字体样式

为了在整个应用中共享颜色和字体样式，我们可以使用主题。定义主题有两种方式：全局主题或使用Theme来定义应用程序局部的颜色和字体样式。事实上，全局主题只是由应用程序根MaterialApp创建的Theme！

定义一个主题后，我们可以在我们自己的Widgets中使用它。另外，Flutter提供的Material Widgets将使用我们的主题为AppBar、Buttons、Checkboxes等设置背景颜色和字体样式。

## 创建应用主题

为了在整个应用程序中共享包含颜色和字体样式的主题，我们可以提供ThemeData给MaterialApp的构造函数。

如果没有提供theme，Flutter将创建一个默认主题。

```
new MaterialApp(  
  title: title,  
  theme: new ThemeData(  
    brightness: Brightness.dark,  
    primaryColor: Colors.lightBlue[800],  
    accentColor: Colors.cyan[600],  
  ),  
);
```

请参阅ThemeData文档以查看您可以定义的所有颜色和字体。

## 局部主题

如果我们想在应用程序的一部分中覆盖应用程序的全局的主题，我们可以将要覆盖得部分封装在一个Theme Widget中。

有两种方法可以解决这个问题：创建特有的ThemeData或扩展父主题。

创建特有的 ThemeData

如果我们不想继承任何应用程序的颜色或字体样式，我们可以通过new ThemeData()创建一个实例并将其传递给ThemeWidget。

```
new Theme(  
  // Create a unique theme with "new ThemeData"  
  data: new ThemeData(  
    accentColor: Colors.yellow,  
  ),  
  child: new FloatingActionButton(  
    onPressed: () {},  
    child: new Icon(Icons.add),  
  ),  
);
```

```
),
```

```
);
```

## 扩展父主题

扩展父主题时无需覆盖所有的主题属性，我们可以通过使用[copyWith](#)方法来实现。

```
new Theme(
```

```
  // Find and Extend the parent theme using "copyWith". Please see the next
```

```
  // section for more info on `Theme.of`.
```

```
  data: Theme.of(context).copyWith(accentColor: Colors.yellow),
```

```
  child: new FloatingActionButton(
```

```
    onPressed: null,
```

```
    child: new Icon(Icons.add),
```

```
  ),
```

```
);
```

## 使用主题

现在我们已经定义了一个主题，我们可以在Widget的build方法中通过Theme.of(context)函数使用它！

Theme.of(context)将查找Widget树并返回树中最近的Theme。如果我们的Widget之上有一个单独的Theme定义，则返回该值。如果不是，则返回App主题。事实上，

FloatingActionButton真是通过这种方式找到accentColor的！

下面看一个简单的示例：

```
new Container(
```

```
  color: Theme.of(context).accentColor,
```

```
  child: new Text(
```

```
    'Text with a background color',
```

```
    style: Theme.of(context).textTheme.title,
```

```
  ),
```

```
);
```

## 完整的例子

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

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

```
void main() {
```

```
  runApp(new MyApp());
```

```
}
```

```
class MyApp extends StatelessWidget {
```

```
  @override
```

```
  Widget build(BuildContext context) {
```

```
    final appName = 'Custom Themes';
```

```
    return new MaterialApp(
```

```

        title: appName,
        theme: new ThemeData(
          brightness: Brightness.dark,
          primaryColor: Colors.lightBlue[800],
          accentColor: Colors.cyan[600],
        ),
        home: new MyHomePage(
          title: appName,
        ),
      );
    }
  }
}

```

```

class MyHomePage extends StatelessWidget {
  final String title;

  MyHomePage({Key key, @required this.title}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return new Scaffold(
      appBar: new AppBar(
        title: new Text(title),
      ),
      body: new Center(
        child: new Container(
          color: Theme.of(context).accentColor,
          child: new Text(
            'Text with a background color',
            style: Theme.of(context).textTheme.title,
          ),
        ),
      ),
      floatingActionButton: new Theme(
        data: Theme.of(context).copyWith(accentColor: Colors.yellow),
        child: new FloatingActionButton(
          onPressed: null,
          child: new Icon(Icons.add),
        ),
      ),
    );
  }
}

```