

一、Flutter写UI方式

1、Android的UI布局

(1) XML声明式

(2) 代码命令式

2、IOS

(1) Storyboard

(2) 代码命令式

3、Flutter的UI写法

一、Flutter写UI方式

Flutter采用了一种新的方式来写UI，就是通过代码声明式。

1、Android的UI布局

(1) XML声明式

通过XML来布局，但是在代码了无法直接修改UI，只能通过findViewById之后再修改UI

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   android:layout_width="match_parent"
5   android:layout_height="wrap_content"
6   android:gravity="center_horizontal"
7   android:orientation="vertical">
8
9   <ImageView
10     android:layout_width="@dimen/dp_72"
11     android:layout_height="@dimen/dp_72"
12     android:layout_marginTop="@dimen/dp_40"
13     android:src="@drawable/global_nearby_user_location" />
14
15   <TextView
16     android:id="@+id/tv_nearby_user_location"
```

```

17 style="@style/BaseWrapParent"
18 android:layout_marginTop="@dimen/dp_12"
19 android:textSize="@dimen/textsize_15"
20 tools:text="开启 定位功能 后你就可以查看附近的书友啦~" />
21
22 <TextView
23 android:id="@+id/tv_open_location"
24 android:layout_width="@dimen/dp_255"
25 android:layout_height="@dimen/dp_43"
26 android:layout_marginTop="@dimen/dp_41"
27 android:background="@drawable/selector_yellow_ff6a33_22dp_yellow_ff6a33"
28 android:gravity="center"
29 android:text="@string/tv_nearby_bookshelf_fragment_open_location"
30 android:textColor="@color/white_FFFFFFFF"
31 android:textSize="@dimen/textsize_15" />
32 </LinearLayout>

```

(2) 代码命令式

代码命令式可以直接来修改UI，但是代码中无法看到布局的UI元素的关系和嵌套层次

```

1 // 分别实现各个view
2 ViewA a = new ViewA(...)
3 ViewB b = new ViewB(...)
4 ViewC c1 = new ViewC(...)
5 ViewC c2 = New ViewC(...)
6 // 然后将子view 添加到容器中
7 a.add(b)
8 b.add(c1)
9 b.add(c2)

```

如果这里有很多UI，我们无法直接看UI元素的关系和布局层次

2、IOS

(1) Storyboard

通过Storyboard来组织视图和设置约束

(2) 代码命令式

可以直接操作组件，但是无法直接看出UI元素的关系和布局层次

3、Flutter的UI写法

Flutter使用的是代码声明式，只能通过代码来书写布局，主要优点是两方面

- 可以在代码中直接操作UI
- 可以在代码中看出UI组件之间的关系，同时也可以看出布局层次
- 减少了XML解析的步骤，构建效率更高

```
1 class TestStatelessWidget extends StatelessWidget {  
2   final String content;  
3  
4   TestStatelessWidget(this.content);  
5  
6   @override  
7   Widget build(BuildContext context) {  
8     // TODO: implement build  
9     return MaterialApp(  
10      title: 'StatelessfulWidget Demo',  
11      color: Colors.black,  
12      home: Scaffold(  
13        appBar: AppBar(  
14          title: Text('Flutter Demo'),  
15        ),  
16        body: Center(  
17          child: Text(content),  
18        ),  
19      ),  
20    );  
21  }
```