学Flutter要懂什么是声明式UI

# 1.两种编程范式

* 命令式编程：命令“机器”如何去做事情(how)，这样不管你想要的是什么(what)，它都会按照你的命令实现。
* 声明式编程：告诉“机器”你想要的是什么(what)，让机器想出如何去做(how)。

这样的解释想必会让人云里雾里，其实你也可以这样理解：

命令式编程：告诉你一步步细节，你去照做展示。

声明式编程：告诉你有一个东西，你把那个东西给放到指定容器。

Flutter的UI框架吸取了react的理念，即 UI是关于状态的函数。

iOS：很纯粹的命令式，new view，addsubview，new view，addsubview，这样搞。

安卓：算是半命令式吧，xml声明了UI，这是声明式的部分；但程序运行时如果要修改某个view，仍是取到这个view，再去命令式地修改。

Flutter：每个组件，会有个build函数，这里会返回一个能够完整描述UI的对象结构。每当数据改变时，就重新调用build函数，返回新的结构。如何高效渲染，就是框架去做的事情了。通过这种方式，不管是UI的初始布局结构，还是后面的修改，都是build函数返回的对象结构去声明的，完整的声明式UI由此而来。所以Flutter是构建新的widget实例，而不是改变旧的实例。

# 1.命令式UI怎么用的

1.dom中的命令式编程

const container = document.getElementById(‘container’);

const btn = document.createElement(‘button’);

btn.className = ‘btn red’;

btn.onclick = function(event) {

if (this.classList.contains(‘red’)) {

this.classList.remove(‘red’);

this.classList.add(‘blue’);

} else {

this.classList.remove(‘blue’);

this.classList.add(‘red’);

}

};

container.appendChild(btn);

2.命令式实现一个容器嵌套

// 分别实现各个view

ViewA a = new ViewA(...)

ViewB b = new ViewB(...)

ViewC c1 = new ViewC(...)

ViewC c2 = New ViewC(...)

// 然后将子view 添加到容器中

a.add(b)

b.add(c1)

b.add(c2)

声明式UI怎么用的

1.react中的声明式编程

class Button extends React.Component{

this.state = { color: 'red' }

handleChange = () => {

const color = this.state.color === 'red' ? 'blue' : 'red';

this.setState({ color });

}

render() {

return (<div>

<button

className=`btn ${this.state.color}`

onClick={this.handleChange}>

</button>

</div>);

}

}

这里我们看到根据不同的state，就完整的描绘了页面的样子。

2.使用声明式进行嵌套

Widget build(BuildContext context) {

return ViewA{

child: ViewB{

color: yellow,

children: [

ViewC(...),

ViewC(...)

]

}

}

}

部分案例参考自如下： [www.cnblogs.com/johnzhu/p/9…](https://www.cnblogs.com/johnzhu/p/9016277.html) [zhuanlan.zhihu.com/p/97385623](https://zhuanlan.zhihu.com/p/97385623)