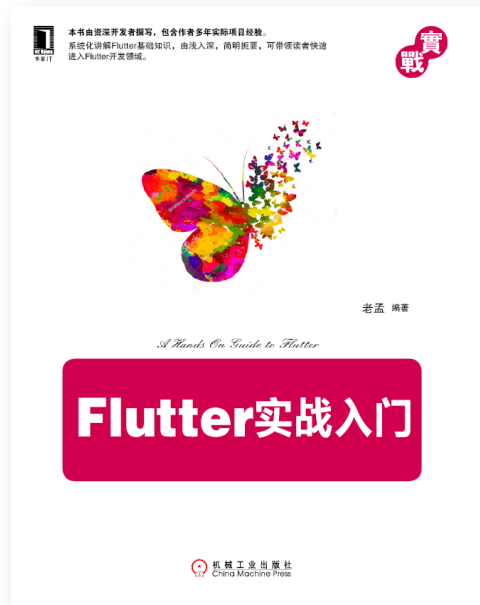


微信交流群



点击购买

GridView

GridView是一个可滚动的，2D数组控件。

基本用法如下：

```
1  GridView(  
2    gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(  
3      crossAxisCount: 3,  
4    ),  
5    children: [  
6      _createGridViewItem(Colors.primaryes[0]),  
7      _createGridViewItem(Colors.primaryes[1]),  
8      _createGridViewItem(Colors.primaryes[2]),  
9      _createGridViewItem(Colors.primaryes[3]),  
10     _createGridViewItem(Colors.primaryes[4]),  
11     _createGridViewItem(Colors.primaryes[5]),  
12     _createGridViewItem(Colors.primaryes[6]),
```


```
13     _createGridViewItem(Colors.primaryes[7]),
14
15   ],
16 )
17 _createGridViewItem(Color color){
18   return Container(
19     height: 80,
20     color: color,
21   );
22 }
```

效果如下：



gridDelegate 参数控制子控件的排列，有2个选择：

- `SliverGridDelegateWithFixedCrossAxisCount`：交叉轴方向上固定数量，对于垂直方向的GridView来说交叉轴方向指的是水平方向。

 `SliverGridDelegateWithMaxCrossAxisExtent`：交叉轴方向上尽量大，比如水平方上有500空间，指定此值为150，那么可以放3个，剩余一些空间，此时GridView将会缩小每一个Item，放置4个。

`SliverGridDelegateWithFixedCrossAxisCount`有属性介绍如下：

- `crossAxisCount`：交叉轴方向上个数。
- `mainAxisSpacing`：主轴方向上2行之间的间隔。
- `crossAxisSpacing`：交叉轴方向上之间的间隔。
- `childAspectRatio`：子控件宽高比。

设置间隔如下：

```
1  GridView(  
2      gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(  
3          crossAxisCount: 3,  
4          crossAxisSpacing: 2,  
5          mainAxisSpacing: 4  
6      )  
7      ...  
8  )
```

效果如下：



`scrollDirection` 表示滚动方向，默认是垂直方向，可以设置为水平方向。

`reverse` 表示是否反转滚动方向，比如当前滚动方向是垂直方向，`reverse` 设置为true，滚动方向为从上倒下，设置为false，滚动方向为从下倒上。

用法如下：

```
1  GridView(  
2    scrollDirection: Axis.horizontal,  
3    reverse: true,  
4    ...  
5  )
```

controller 表示滚动相关，可以通过`ScrollController`获取到当前滚动位置，或者指定滚动到某一位置，用法如下：

```
1 ScrollController _gridViewController;
2
3 @override
4 void initState() {
5   _gridViewController = ScrollController()..addListener(() {
6     print('${_gridViewController.position}');
7   });
8 }
9
10 GridView(
11   controller: _gridViewController,
12   ...
13 )
```

physics 参数控制滚动到物理特性，比如设置为不可滚动：

```
1 GridView(
2   physics: NeverScrollableScrollPhysics(),
3   ...
4 )
```

系统提供的`ScrollPhysics`有：

- `AlwaysScrollableScrollPhysics`：总是可以滑动
- `NeverScrollableScrollPhysics`：禁止滚动
- `BouncingScrollPhysics`：内容超过一屏 上拉有回弹效果
- `ClampingScrollPhysics`：包裹内容 不会有回弹

直接使用最开始的方法创建`GridView`是不推荐的，此方法不适合加载大量数据的情况，`GridView`提供了一些快速构建的方法，比如`builder`，用法如下：

```
1 GridView.builder(
2   gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
3     crossAxisCount: 3,
4   ),
```

```
5     itemBuilder: (context, index) {  
6         return Container(  
7             height: 80,  
8             color: Colors.primaryes[index % Colors.primaryes.length],  
9         );  
10    },  
11    itemCount: 50,  
12 )
```

`itemBuilder` 是构建子控件， `itemCount` 指定数据个数。

使用 `GridView.custom` 构建：

```
1  GridView.custom(  
2      gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(  
3          crossAxisCount: 3,  
4      ),  
5      childrenDelegate: SliverChildBuilderDelegate((context, index) {  
6          return Container(  
7              height: 80,  
8              color: Colors.primaryes[index % Colors.primaryes.length]);  
9      }, childCount: 50),  
10 )
```

使用 `GridView.count` 构建：

```
1  GridView.count(  
2      crossAxisCount: 3,  
3      children: List.generate(50, (i) {  
4          return Container(  
5              height: 80,  
6              color: Colors.primaryes[i % Colors.primaryes.length],  
7          );  
8      }),  
9  )
```

使用 `GridView.extent` 构建：

```
1  GridView.extent(  
2    maxCrossAxisExtent: 100,  
3    children: List.generate(50, (i) {  
4      return Container(  
5        height: 80,  
6        color: Colors.primaryes[i % Colors.primaryes.length],  
7      );  
8    }),  
9  )
```

喜欢作者

版权所有，禁止私自转发、克隆网站。

[< GridTileBar](#)

[Hero >](#)