

Flutter 可滚动Widget -- CustomScrollView

[CustomScrollView](https://docs.flutter.io/flutter/widgets/CustomScrollView-class.html)

(<https://docs.flutter.io/flutter/widgets/CustomScrollView-class.html>) 是可以使用 slivers 来自定义滑动效果的可滚动Widget。

代码所在位置

flutter_widget_demo/lib/customccrollview/CustomScrollViewV

CustomScrollView 的使用

CustomScrollView 里面可以添加多个 Widget，而且可以为 Widget 提供复杂的滑动效果，需要为其 slivers 参数赋值，而且 slivers 参数只能接受特定的 Widget，例如：

```
CustomScrollView(  
  slivers: <Widget>[  
    const SliverAppBar(  
      pinned: true,  
      expandedHeight: 250.0,  
      flexibleSpace: FlexibleSpaceBar(  
        title: Text('Demo'),  
      ),  
    ),  
    SliverGrid(  
      gridDelegate:
```

```

SliverGridDelegateWithMaxCrossAxisExtent(
  maxCrossAxisExtent: 200.0,
  mainAxisSpacing: 10.0,
  crossAxisSpacing: 10.0,
  childAspectRatio: 4.0,
),
delegate: SliverChildBuilderDelegate(
  (BuildContext context, int index) {
    return Container(
      alignment: Alignment.center,
      color: Colors.teal[100 * (index %
9)],
      child: Text('grid item $index'),
    );
  },
  childCount: 20,
),
),
SliverFixedExtentList(
  itemExtent: 50.0,
  delegate: SliverChildBuilderDelegate(
    (BuildContext context, int index) {
      return Container(
        alignment: Alignment.center,
        color: Colors.lightBlue[100 * (index
% 9)],
        child: Text('list item $index'),
      );
    },
  ),
),
],
)

```

CustomScrollView 在一个页面使用的 Demo 为:

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

void main() => runApp(CustomScrollViewWidget());

class CustomScrollViewWidget extends
StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Test',
      home: Scaffold(
        appBar:
          AppBar(title: new Text('Flutter 可滚动
Widget -- CustomScrollView')),
        body: CustomScrollView(
          slivers: <Widget>[
            const SliverAppBar(
              pinned: true,
              expandedHeight: 250.0,
              flexibleSpace: FlexibleSpaceBar(
                title: Text('Demo'),
              ),
            ),
            SliverGrid(
              gridDelegate:
SliverGridDelegateWithMaxCrossAxisExtent(
                maxCrossAxisExtent: 200.0,
                mainAxisSpacing: 10.0,
                crossAxisSpacing: 10.0,
                childAspectRatio: 4.0,
              ),
```

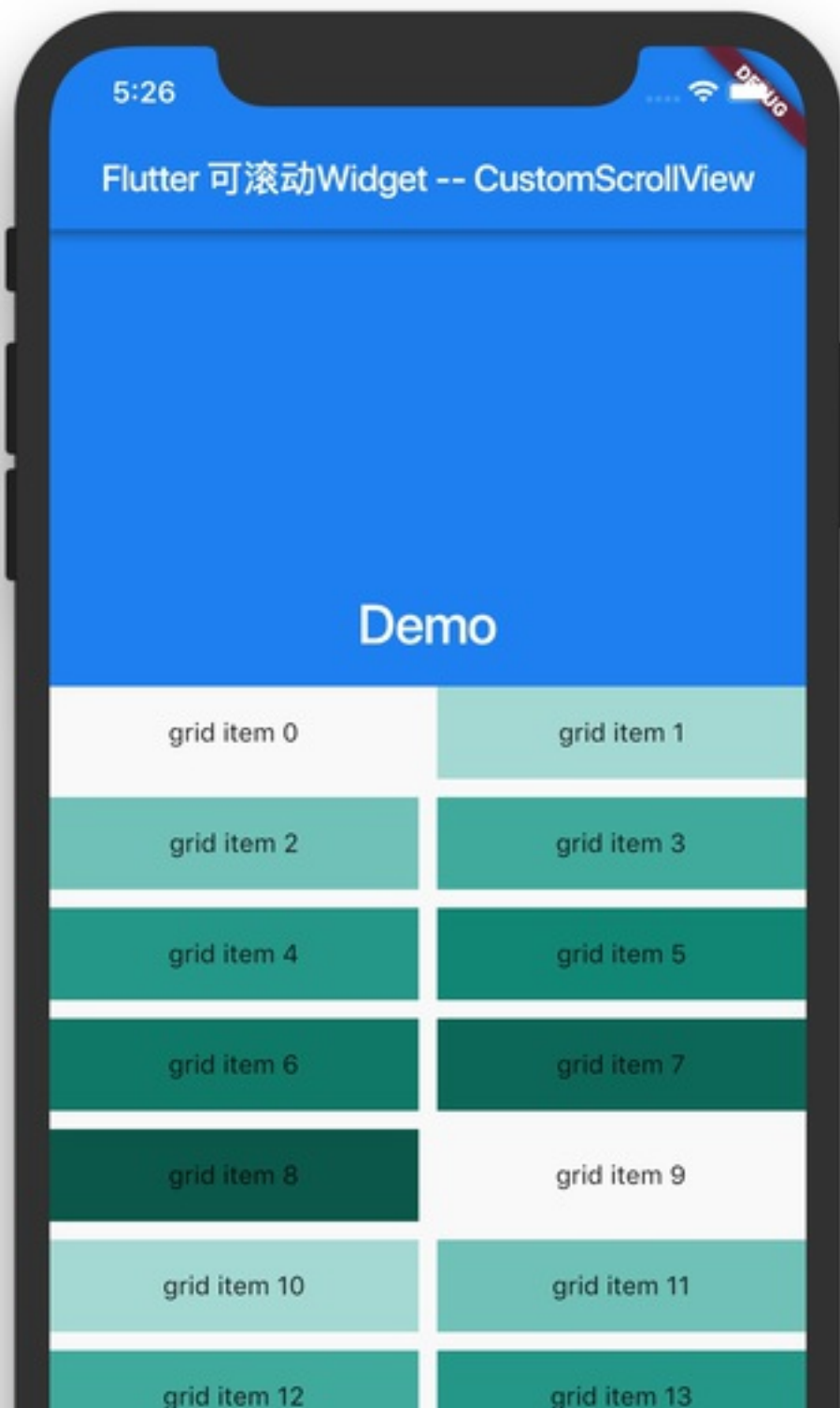
```

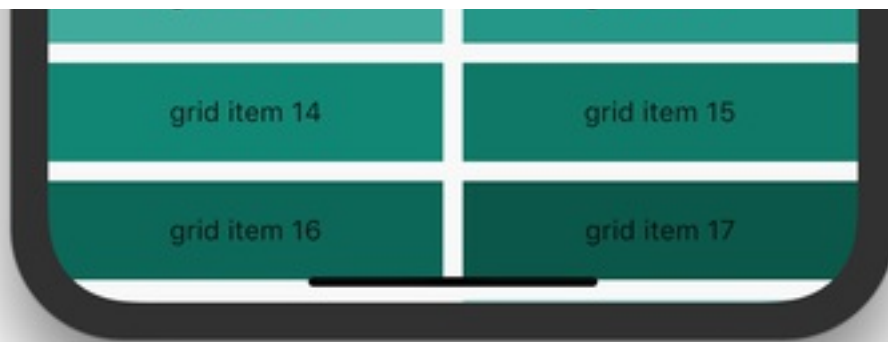
        delegate:
SliverChildBuilderDelegate(
    (BuildContext context, int index)
{
    return Container(
        alignment: Alignment.center,
        color: Colors.teal[100 *
(index % 9)],
        child: Text('grid item
$index'),
    );
},
    childCount: 20,
),
SliverFixedExtentList(
    itemExtent: 50.0,
    delegate:
SliverChildBuilderDelegate(
    (BuildContext context, int index)
{
    return Container(
        alignment: Alignment.center,
        color: Colors.lightBlue[100 *
(index % 9)],
        child: Text('list item
$index'),
    );
},
),
],
),

```

```
    ),  
  );  
}  
}
```

运行后的效果为：





CustomScrollView 的构造函数及参数说明

CustomScrollView 的构造函数为：

```
class CustomScrollView extends ScrollView {  
  const CustomScrollView({  
    Key key,  
    Axis scrollDirection = Axis.vertical,  
    bool reverse = false,  
    ScrollController controller,  
    bool primary,  
    ScrollPhysics physics,  
    bool shrinkWrap = false,  
    Key center,  
    double anchor = 0.0,  
    double cacheExtent,  
    this.slivers = const <Widget>[],  
    int semanticChildCount,  
    DragStartBehavior dragStartBehavior =  
    DragStartBehavior.down,  
  })  
  ...  
}
```

参数名字	参数类型	意义
key	Key	Widget 的标识
scrollDirection	Axis	滑动的方向 默认为 Axis.vertical，垂直滑动
reverse	bool	控制 CustomScrollView 的排列顺序，是按照插入还是按照插入顺序相反的顺序。 默认为 false，就是按照插入顺序，第一个插入的在头部，当 reverse 为 true 时，插入的会在底部 可以控制 CustomScrollView 的位置 ScrollController 提供以下功能： 1.设置 CustomScrollView 初始位置 2.可以控制 CustomScrollView 否存储和恢复滑动的位置 3.可以读取、设置当前滑动位置 可以继承 ScrollController 定义的功能 当 primary 为 true 时，controller 必须为 null 是否是和父级关联的主滚动控制器
controller	ScrollController	

primary	bool	<p>当为 true 时，即使 CustomScrollView 里没内容也能滑动</p> <p>设置 CustomScrollView 效果</p> <p>值必须为 ScrollPhysics 的，如有如下的值：</p>
physics	ScrollPhysics	<p>AlwaysScrollableScrollPhysics 可以让 CustomScrollView 足够的内容也能滑动</p> <p>ScrollPhysics():CustomScrollView 在没有足够的 content 的时候，是否根据列表项的总长度，CustomScrollView 的长度，值为 false。</p> <p>当 shrinkWrap 为 false 时，CustomScrollView 会在内容上扩展到可占用的最大空间</p>
shrinkWrap	bool	<p>当 shrinkWrap 为 true 时，CustomScrollView 在滚动时占用的空间就是其列表项的，但是这样会很耗性能，因为列表项发生变化时，CustomScrollView 的大小需要重新计算</p>
center	Key	<p>放在 CustomScrollView Widget 的 key</p> <p>CustomScrollView 开始时的偏移量</p> <p>如果 anchor 为 0.0，则 CustomScrollView 的子从头开始排列</p>

anchor	double	<p>如果 anchor 为 0.5，则 CustomScrollView 的子从中间开始排列</p> <p>如果 anchor 为 1.0，则 CustomScrollView 的子从底部开始排列</p> <p>CustomScrollView 可见区域的前面和后面的区域可以用来绘制项，这部分区域的 item 即使不会加载出来，所以当滑动到该区域的时候，缓存的区域就会可见，cacheExtent 就表示缓存可见部分的前面和后面有多少</p>
cacheExtent	double	
slivers	List<Widget>	CustomScrollView 的列表项
semanticChildCount	int	<p>提供语义信息的列表项的数量</p> <p>默认为 CustomScrollView 的数量</p> <p>确定处理拖动开始行为的方式</p> <p>如果设置为 [DragStartBehavior.start] 检测到拖动手势时将开始拖动行为</p> <p>如果设置为 [DragStartBehavior.down] 在首次检测到向下事件时开始拖动行为</p>
dragStartBehavior	DragStartBehavior	

CustomScrollView 的 slivers 属性的值，只能是以 Sliver 开头的一系列 Widget：

- SliverList

- SliverFixedExtentList
- SliverGrid
- SliverPadding
- SliverAppBar