

Column



Column

在垂直方向上排列子widget的列表。

列类

一个小部件显示它的孩子在一个垂直数组中。

导致孩子扩大填充可用的垂直空间, 包装 孩子在一个[扩大](#)小部件。

的[列](#)部件不滚动(通常被认为是一个错误 在一个有更多孩子[列](#)比可用的房间)。 如果 你有一行小部件和希望他们能够滚动是否存在 不足的房间, 考虑使用[列表视图](#)。

水平不同, 看到的[行](#)。

如果你只有一个孩子, 那么考虑使用[对齐](#)或[中心](#)来 这个孩子。

示例代码

这个示例使用[列](#)垂直安排三个小部件, 最后一次 填满所有的剩余空间。

```
new Column(  
  children: <Widget>[  
    new Text('Deliver features faster'),  
    new Text('Craft beautiful UIs'),  
    new Expanded(  
      child: new FittedBox(  
        fit: BoxFit.contain, // otherwise the logo will be tiny  
        child: const FlutterLogo(),  
      ),  
    ),  
  ],  
)
```

在上面的示例中, 文字和标志都集中在每一行。 在 下面的例子中, [crossAxisAlignment](#)被设置为[CrossAxisAlignment.start](#), 所以孩子们左对齐。 的[mainAxisSize](#)被设置为[MainAxisSize.min](#), 所以列收缩 适合孩子们。

```
new Column(  
  crossAxisAlignment: CrossAxisAlignment.start,  
  mainAxisSize: MainAxisSize.min,  
  children: <Widget>[  
    new Text('We move under cover and we move as one'),  
    new Text('Through the night, we have one shot to live another day'),  
    new Text('We cannot let a stray gunshot give us away'),  
    new Text('We will fight up close, seize the moment and stay in it'),  
    new Text('It' s either that or meet the business end of a bayonet'),  
    new Text('The code word is 'Rochambeau,' dig me?'),  
    new Text('Rochambeau!', style: DefaultTextStyle.of(context).style.apply(fontSizeFactor: 2.0)),  
  ],  
)
```

故障排除

当传入的垂直约束是无界的

当一个[列](#)有一个或多个[扩大](#)或[灵活的](#)孩子,是 放置在另一个[列](#),或者在一个[列表视图](#),或者在其他上下文 这并不提供一个最大高度约束的[列](#),你会 在运行时得到一个例外说有非零的孩子 flex但垂直约束是无界的。

中描述的问题,细节伴随异常, ,使用[灵活的](#)或[扩大](#)意味着后的剩余空间 列出了所有其他的孩子必须共享同样的,但是,如果 传入的纵向约束是无界的,有无限的剩余 空间。

通常解决这个问题的关键是确定的原因[列](#)是 获得无限的垂直约束。

一个常见的原因是发生[列](#)被放置在 另一个[列](#)(不使用[扩大](#)或[灵活的](#)在内部 嵌套的[列](#))。 当一个[列](#)列出了其non-flex(那些孩子 既没有[扩大](#)或[灵活的](#)周围),这给了他们无限 约束,这样他们就可以确定自己的维度(传递 无限约束通常信号的孩子 压缩其内容)。

解决方案在这种情况下通常而已 在一个包装内列[扩大](#)表明它应该 外专栏的剩余空间,而不是被允许采取任何 的房间的欲望。

显示此消息的另一个原因是嵌套[列](#)内部 一个[列表视图](#)或其他垂直滚动。 在这种情况下,真的有 无限的垂直空间(垂直滚动列表的全部意义 允许无限空间垂直)。 在这样的情况下,它通常是值得的 检查为什么内部[列](#)应该有一个[扩大](#)或[灵活的](#)孩子:内心的孩子真的应该什么尺寸的? 的解决方案 通常删除[扩大](#)或[灵活的](#)各地的小部件 内的孩子。

讨论约束,明白了[BoxConstraints](#)。

黄色和黑色条纹旗帜

当的内容[列](#)超过数量的可用空间,[列](#)溢出,内容是剪。 在调试模式下,一个黄色的 和黑色条纹栏呈现在满溢的边缘来表示 以下问题,将输出一条消息[列](#)说多少 溢出检测。

通常的解决方案是使用一个[列表视图](#)而不是一个[列](#),使 内容滚动时垂直空间是有限的。

布局算法

本节描述如何[列](#)呈现的框架。 看到[BoxConstraints](#)介绍盒布局模型。

布局的[列](#)收入在六个步骤:

1. 布局每个孩子一个零或零flex的因素(例如,这样的企业是没有希望的[扩大](#))与无限的垂直约束和传入的 水平约束。 如果[crossAxisAlignment](#)是[CrossAxisAlignment.stretch](#),而不是使用紧水平约束 匹配传入的最大宽度。
2. 把剩余的非零的孩子之间的垂直空间 flex因素(例如,那些[扩大](#)根据他们的flex) 的因素。 例如,一个孩子与一个flex 2.0倍将收到两次 垂直空间的数量作为一个孩子与一个flex 1.0倍。
3. 布局的每个剩余的孩子相同的水平 限制在步骤1中,而是使用无界的垂直 约束,使用垂直约束基础上的空间 在步骤2中分配。 的孩子[Flexible.fit](#)属性[FlexFit.tight](#)(即给出严格的约束。 ,不得不填补 分配空间),和孩子[Flexible.fit](#)属性[FlexFit.loose](#)(即有宽松的约束。 ,而不是被迫填补 分配空间)。
4. 的宽度[列](#)是孩子们的最大宽度(永远满足的水平约束)。
5. 的高度[列](#)是由的[mainAxisSize](#)财产。 如果[mainAxisSize](#)属性是[MainAxisSize.max](#),然后的高度 的[列](#)的最大高度是传入的约束。 如果[mainAxisSize](#)属性是[MainAxisSize.min](#),然后的高度[列](#)是高度的总和的孩子(传入的主题 约束)。
6. 为每一个孩子根据确定的位置[mainAxisAlignment](#)和[crossAxisAlignment](#)。 例如,如果[mainAxisAlignment](#)是[MainAxisAlignment.spaceBetween](#),任何垂直 空间没有被分配到和孩子是平均分配的 放置在儿童。

参见:

- [行](#)水平距离。
- [Flex](#),如果你事先不知道如果你想要一个水平或垂直 安排。
- [扩大](#),表示孩子应该把所有剩下的房间。
- [灵活的](#),这表明儿童应该分享剩余的房间但 离开一些剩余空间可能大小较小(未使用)。
- 的[目录布局小部件](#)。

继承

- [对象](#)
- [Diagnosticable](#)

- [DiagnosticableTree](#)
- [小部件](#)
- [RenderObjectWidget](#)
- [MultiChildRenderObjectWidget](#)
- [Flex](#)
- [列](#)

构造函数

[列](#)([{关键 关](#)
[键, MainAxisAlignment](#) mainAxisAlignment:MainAxisAlignment.start, [MainAxisSize](#) mainAxisSize:MainAxisSize.max, [Cr](#)
[表](#)<[小部件](#)> 孩子们:常量[]))
 创建一个垂直的孩子。 [\[...\]](#)

属性

[孩子们](#) → [列表](#)<[小部件](#)>
 下面的部件在树上这个小部件。 [\[...\]](#)
 最后, 继承了
[crossAxisAlignment](#) → [CrossAxisAlignment](#)
 孩子们应该如何放置沿着十字轴。 [\[...\]](#)
 最后, 继承了
[方向](#) → [轴](#)
 使用为主要轴线方向。 [\[...\]](#)
 最后, 继承了
[hashCode](#) → [int](#)
 这个对象的哈希码。 [\[...\]](#)
 只读的, 遗传的
[关键](#) → [关键](#)
 控制一个小部件替换另一个小部件在树上。 [\[...\]](#)
 最后, 继承了
[mainAxisAlignment](#) → [MainAxisAlignment](#)
 孩子们应该如何放置沿着主要的轴。 [\[...\]](#)
 最后, 继承了
[mainAxisSize](#) → [MainAxisSize](#)
 多少空间应该占据主要的轴。 [\[...\]](#)
 最后, 继承了
[runtimeType](#) → [类型](#)

一个对象的运行时类型的代表。

只读的, 遗传的

[textBaseline](#) → [TextBaseline](#)

如果调整项目根据其基线, 基线使用。

最后, 继承了

[textDirection](#) → [TextDirection](#)

决定了要把孩子水平和如何解释start和end在水平方向上。[...]

最后, 继承了

[verticalDirection](#) → [VerticalDirection](#)

决定了要把孩子垂直和如何解释start和end在垂直方向。[...]

最后, 继承了

方法

[createElement\(\)](#) → [MultiChildRenderObjectElement](#)

RenderObjectWidgets总是夸大[RenderObjectElement](#)子类。

继承了

[createRenderObject](#)(BuildContext 上下文) → [RenderFlex](#)

创建的一个实例[RenderObject](#)类, 这[RenderObjectWidget](#)表示, 使用描述的配置[RenderObjectWidget](#)。[...]

继承了

[debugDescribeChildren\(\)](#) → 列表<[DiagnosticsNode](#)>

返回一个列表[DiagnosticsNode](#)描述该节点的对象 的孩子。[...]

@protected, 继承了

[debugFillProperties](#)(DiagnosticPropertiesBuilder 属性) → 无效

继承了

[didUnmountRenderObject](#)(RenderObject renderObject) → 无效

渲染对象之前与此小部件被移除 从树上。 给定的[RenderObject](#)将相同类型的吗 返回这个对象的[createRenderObject](#)。

@protected, 继承了

[getEffectiveTextDirection](#)(BuildContext 上下文) → [TextDirection](#)

值传递[RenderFlex.textDirection](#)。[...]

@protected, 继承了

[noSuchMethod](#)(调用 调用) → 动态

当用户访问一个不存在的方法或属性调用。[...]

继承了

[toDiagnosticsNode](#)({字符串 的名字, [DiagnosticsTreeStyle](#) 风格}) → [DiagnosticsNode](#)

返回一个对象被调试的调试表示 工具和[toStringDeep](#)。[...]

继承了

[toString](#)({[DiagnosticLevel](#) minLevel:DiagnosticLevel.debug}) → 字符串

返回该对象的字符串表示。

继承了

[toStringDeep](#)({字符串 prefixLineOne:", 字符串

字符串 prefixOtherLines, [DiagnosticLevel](#) minLevel:DiagnosticLevel.debug}) → 字符串

返回一个字符串表示该节点及其后代。[...]

继承了

[toStringShallow](#)({字符串 乔伊纳:", " , [DiagnosticLevel](#) minLevel:DiagnosticLevel.debug}) → 字符串

返回一行详细描述的对象。[...]

继承了

[toStringShort](#)() → 字符串

短, 这个小部件的文本描述。

继承了

[`updateRenderObject`](#)([`BuildContext`](#) 上下文, [`RenderFlex`](#) `renderObject`)→无效

描述复制配置[`RenderObjectWidget`](#)到 鉴于[`RenderObject`](#)将相同类型的返回 对象的[`createRenderObject`](#)。[\[...\]](#)

继承了

操作

[运算符==](#)(动态 其他)→[`bool`](#)

相等操作符。[\[...\]](#)

继承了