



Row

在水平方向上排列子widget的列表。

一个小部件显示其孩子在水平数组。

导致一个孩子来填补可用的水平空间扩张, 包装 孩子在一个[扩大](#)小部件。

的[行](#)部件不滚动(通常被认为是一个错误 在一个有更多孩子[行](#)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](#)。

垂直变异, 看到[列](#)。

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

这个例子将可用空间划分为三个(水平) 文本集中在前两个地方集中在细胞和颤振的标志 第三:

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

故障排除

为什么我的行有黄色和黑色条纹警告吗?

如果non-flexible行内容(那些不是裹着[扩大](#)或[灵活的](#)小部件)在一起更广泛的比行本身, 然后行据说溢出。 当行溢出时, 行 没有任何剩余的空间之间的分享[扩大](#)和[灵活的](#)的孩子。 行报告通过画一个黄色和黑色的条纹 警告框边缘溢出。 如果有房间在外面行, 溢出的数量印在红色字体。

故事时间

例如, 假设你有这段代码:

```
new Row(  
  children: <Widget>[  
    const FlutterLogo(),  
    const Text('Flutter\'s hot reload helps you quickly and easily experiment, build UIs, add features, and fix bug  
faster. Experience sub-second reload times, without losing state, on emulators, simulators, and hardware for iOS and
```

```
Android.'),
    const Icon(Icons.sentiment_very_satisfied),
  ],
)
```

行首先问自己的第一个孩子, [FlutterLogo](#)、制定 无论大小的标志。 标志是友好和愉快地决定 是24像素。 这使得大量的房间为下一个孩子。 的行然后问下一个孩子, 文本, 来布置, 无论大小 认为是最好的。

在这一点上, 文本, 不知道有多宽太宽, 说: “好吧, 我会的 是thiiiiiiiiiiiiiiiiiiiiis宽。 ”, 远远超出的空间 行, 没有包装。

行回应, “这是不公平的, 现在我 没有更多的空间供我的其他孩子! ”, 和生气 豆芽一个黄色和黑色地带。

解决办法是包装的第二个孩子 [扩大](#) 讲述了小部件 行这个孩子应该给剩下的房间:

```
new Row(
  children: <Widget>[
    const FlutterLogo(),
    const Expanded(
      child: const Text('Flutter\'s hot reload helps you quickly and easily experiment, build UIs, add features, and fix
bug faster. Experience sub-second reload times, without losing state, on emulators, simulators, and hardware for iOS and
Android.'),
    ),
    const Icon(Icons.sentiment_very_satisfied),
  ],
)
```

现在, 行首先要求商标布局, 然后问 [图/标](#) 奠定 出去了。 的 [图标](#) 商标一样, 乐于承担一个合理的大小(也 24像素, 并非巧合, 因为 [FlutterLogo](#) 和 [图标](#) 尊敬 环境 [IconTheme](#))。 这使得一些房间留下, 现在行告诉 文本到底有多宽: 准确的剩余空间的宽度。 的 文本, 现在乐意服从一个合理的要求, 包装内的文本 宽度, 最终你会得到一个段落分割几行。

布局算法

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

布局的 [行](#) 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](#))与无限水平约束和传入的 垂直约束。 如果 [crossAxisAlignment](#) 是 [CrossAxisAlignment.stretch](#), 而不是使用严格的垂直约束 匹配传入的最大高度。
2. 把剩下的水平空间非零的孩子 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 的水平空间作为一个孩子与一个flex 1.0倍。
3. 每个剩余的孩子相同的垂直布局约束 在步骤1中,而是使用无界的水平约束,使用 水平约束的基础上,在步骤2中分配的空间量。 的孩子 [Flexible.fit](#) 属性 [FlexFit.tight](#) 是 (即给予严格约束。 ,不得不填补空间分配), 的孩子 [Flexible.fit](#) 属性 [FlexFit.loose](#) 是 (即给予宽松的约束。 ,而不是被迫填补空间分配)。
4. 的高度 [行](#) 是孩子们的最大高度(总是满足传入的垂直约束)。
5. 的宽度 [行](#) 是由的 [mainAxisSize](#) 财产。 如果 的 [mainAxisSize](#) 属性是 [MainAxisSize.max](#), 然后的宽度 [行](#) 的最大宽度的限制。 如果 [mainAxisSize](#) 属性是 [MainAxisSize.min](#), 然后的宽度 [行](#) 是和 宽度的孩子(受到的约束)。
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](#), 任何水平 空间没有被分配到和孩子是平均分配的 放置在儿童。

参见:

- [列](#), 对于一个垂直的等效。
- [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 可能通过大小较小(留下一些剩余空间未使用)。
- 的 [目录布局小部件](#)。

继承

- [对象](#)

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

构造函数

[行](#) ({[关键](#) [关](#)
 键, [MainAxisAlignment](#) mainAxisAlignment:MainAxisAlignment.start, [MainAxisSize](#) mainAxisAlignment:MainAxisSize.max, [Cr](#)
 表<[小部件](#)> 孩子们:常量[]})
 创建一个水平的孩子。[...]

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