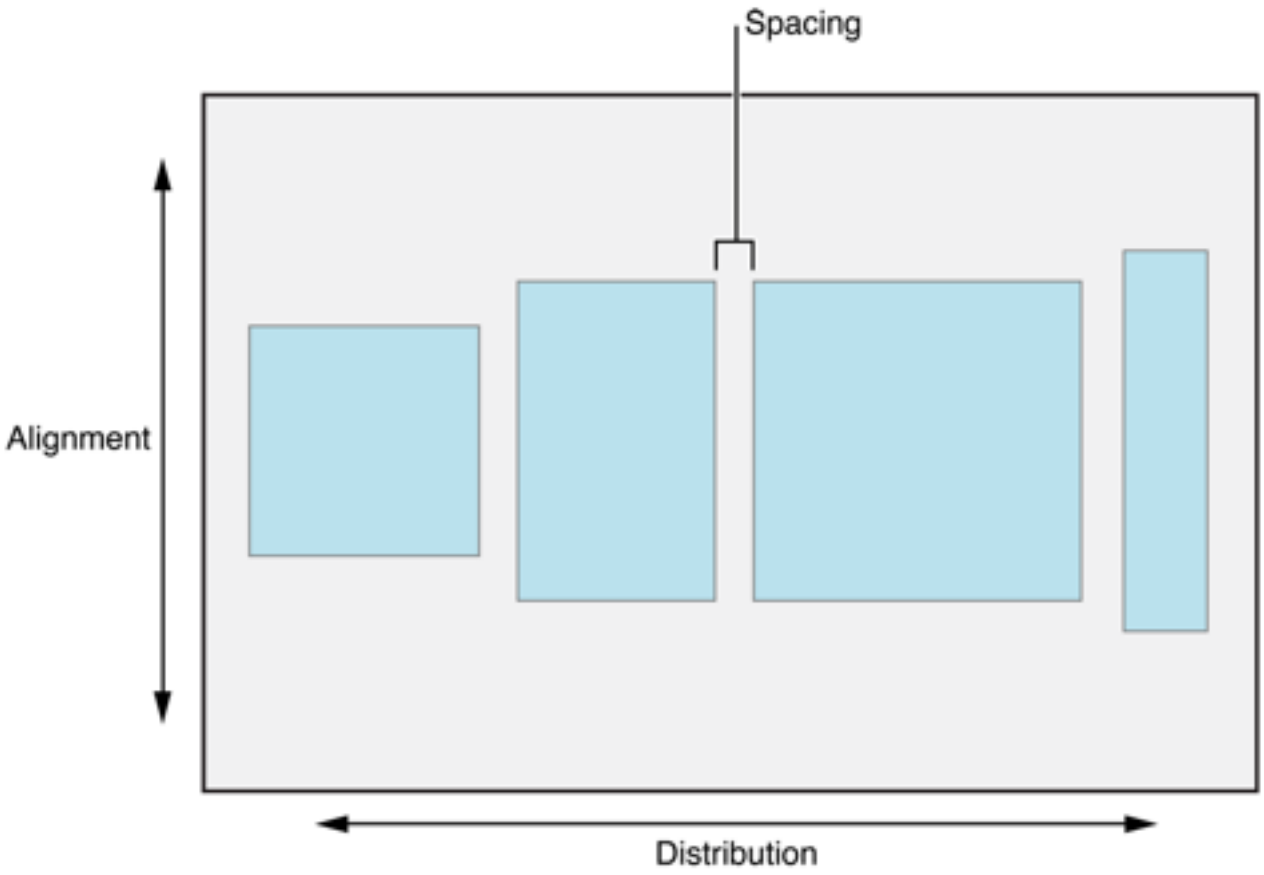
UIStackView

UIStackView为布局一个（列或行）视图的集合提供了一种流线型的用户界面，Stack views 让你利用Auto Layout的能力，来创建能够动态的适应设备的方向、屏幕的宽度、和可用范围的改变的用户界面。Stack view管理在它的arrangedSubviews属性中所有的视图的布局。这些视图根据他们在arrangedSubviews数组中的顺序，沿着stack view的axis被排列。精确的布局变量由stack view的axis, distribution, alignment, spacing,和其他属性来决定。



打开你想要编辑的Storyboard，使用stack view。从对象库中拖拽一个水平/垂直的Stack View，并且放到你希望的位置。接下来，拖拽view或者control，放置在stack中，作为它的内容。如果需要，你能继续往stack中添加views和controls。Interface Builder会根据它的内容重新调整尺寸大小。你还能够通过在Attributes inspector中修改Stack View的属性来调整stack的内容的外观。

注意

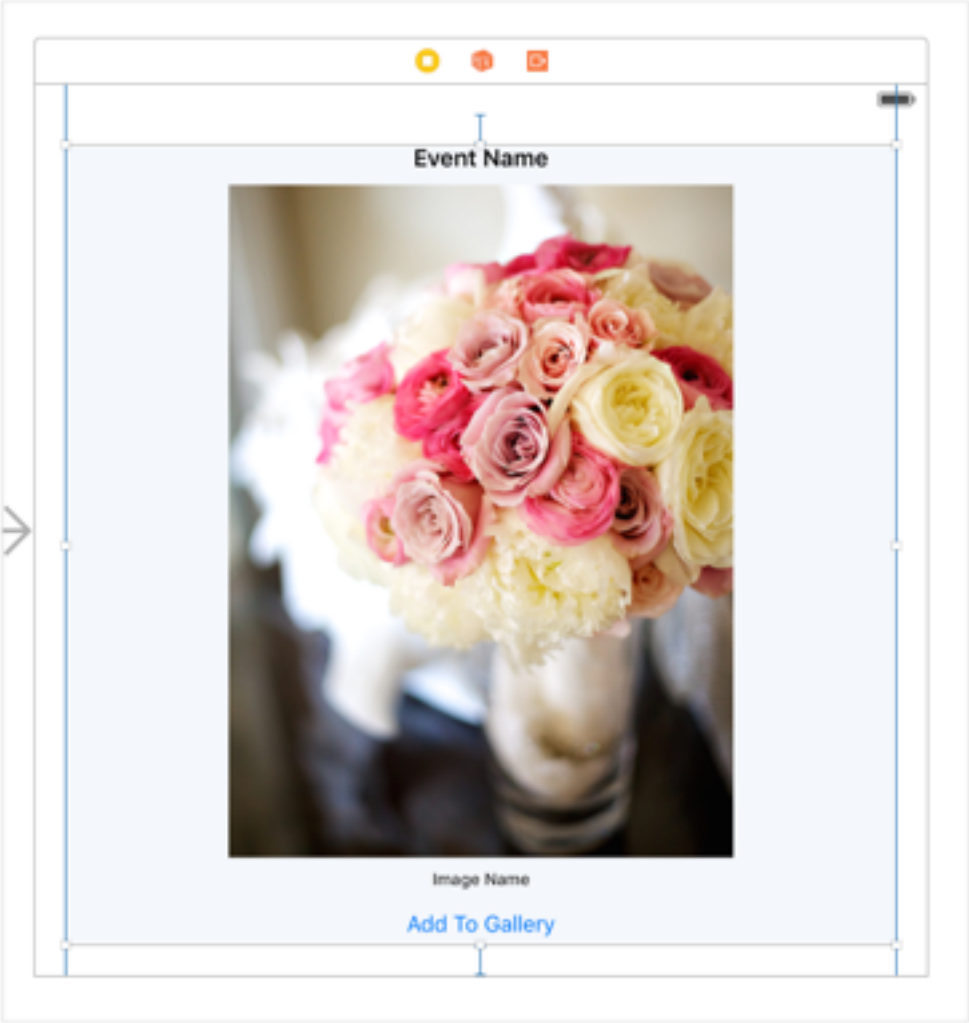
你需要负责指定 stack 视图的位置和尺寸（可选的）。然后 stack 视图将管理它的内容的布局和尺寸。

Stack View and Auto Layout

Stack 视图使用自动布局来控制它安排的views的位置和尺寸大小。stack 视图沿着它的axis用它的edges排列第一个和最后一个被管理视图。在一个水平 stack 中，这意味着第一个被管理视图的leading edge固定在 stack 的leading edge，并且最后一个被管理视图trailing固定在 stack 的trailing edge。在一个垂直 stack中，top edge 和 bottom edge 分别被固定在 stack 的 top edge 和 bottom edge。如果你设置了 stack view的 layoutMarginsRelativeArrangement 属性为 YES，stack view将使用相关的margin代替它的edge来固定其中的内容。

对于除去 UIStackViewDistributionFillEqually 分布以外的所有分布方式，当沿着stack的axis计算被安排视图的尺寸大小的时候, stack view使用被管理视图的 intrinsicContentSize 属性。UIStackViewDistributionFillEqually 分布将调节所有被管理视图拥有相同尺寸，沿着 stack 的axis填充 stack view。如果可能，stack 视图将拉伸所有被管理视图，来匹配其在 stack 的axis上最长的原有尺寸（译注：保证长宽比的情况下根据 stack 轴向长度拉伸视图）。

对于除去 UIStackViewAlignmentFill 对齐方式以外的所有的对齐方式，当垂直stack的axis计算被安排视图的尺寸大小的时候，stack view使用其管理的视图的 intrinsicContentSize 属性。UIStackViewAlignmentFill 调节了所有其管理的视图，使这些视图填充 stack 视图垂直于其轴向空间。如果可能，stack 视图将拉伸其所有管理的视图来匹配其垂直于 stack 轴向的最大固有尺寸。



Positioning and Sizing the Stack View

尽管 Stack view允许你不直接使用Auto Layout来布局其内容，你将仍然需要使用自动布局来定位stack view。通常情况下，这意味着需要固定stack view的至少两个边界相邻的来定义它的位置。没有额外约束的情况下，系统会根据它的内容计算stack view的尺寸：

- 沿着 stack view的axis，适应的尺寸等于其管理的视图尺寸加上两个视图间距的和；
- 垂直于 stack view的axis，适应尺寸等于其管理的视图中最大的视图的尺寸；
- 如果 stack view的 layoutMarginsRelativeArrangement 属性设置为 YES，stack view的适应尺寸会被增加（包括边距空间）。

你可以提供额外的约束来具体说明 stack view的高度、宽度或者两者兼有。在这些情况下，stack view调整了其管理的视图的布局和尺寸来填充指定区域。精确的布局变量根据 stack 视图的属性获得。可以通过查看 UIStackViewDistribution 和 UIStackViewAlignment 枚举，以获得一个 stack view怎么处理在其内容空间多余或空间不足的情况的完整描述。

你也可以根据 stack view的第一条或最后一条基线定位它，来替代使用顶部、底部或者中心的Y值。类似于 stack view的适应尺寸，这些基线都是基于 stack view的内容计算得到的。

- 一个水平的 stack view调用 viewForFirstBaselineLayout 方法或 viewForLastBaselineLayout 方法时返回它最高的视图。如果最高的视图也是一个 stack view，那么其返回的将是在嵌套的 stack 视图上调用 viewForFirstBaselineLayout 方法或 viewForLastBaselineLayout 方法的结果。
- 一个垂直的 stack view当调用 viewForFirstBaselineLayout 方法时返回的是其管理的第一个视图，当调用 viewForLastBaselineLayout 方法时返回的是其管理的最后一个视图。如果这两个视图之一也是 stack view，那么其返回的将是在嵌套的 stack view上对应调用 viewForFirstBaselineLayout 方法或 viewForLastBaselineLayout 方法的结果。

注意

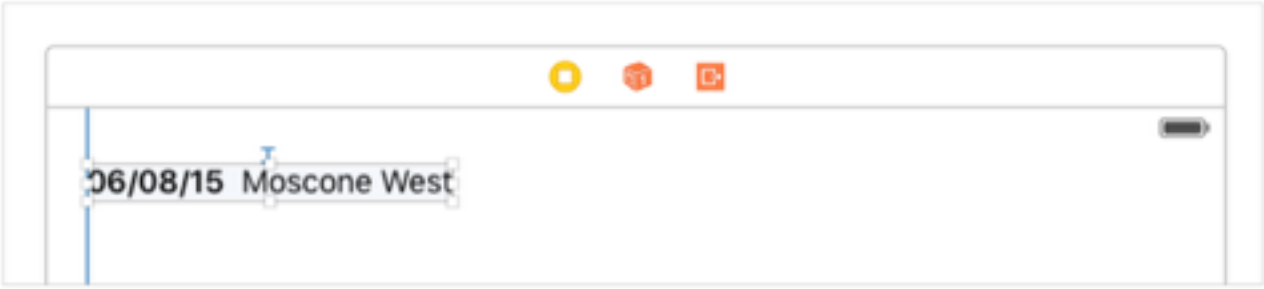
基线对齐方式只作用于那些高度匹配其原本内容高度的视图。如果视图被拉伸或压缩过，那么基线将出现在错误的位置上。

Common Stack View Layouts

这有一些用于布局stack view内容的通用方法。

- **只是定义位置**。你可以通过固定两个与其父视图相邻的边界来定义 stack view的位置。使用这种方式，stack view的尺寸大小将根据其管理的视图在两个维度上自由扩展。当你想要 stack view的内容按照其原有内容尺寸显示，和你想要管理其他与 stack view有关联的用户接口元素时候，这种方式是特别有效。

例一，stack view的 leading edge 和 top edge 都已经相对固定于其父视图。文本标签将根据带有8个点的两者之间的空间作为第一基线校准，左对齐的 stack view的内容。

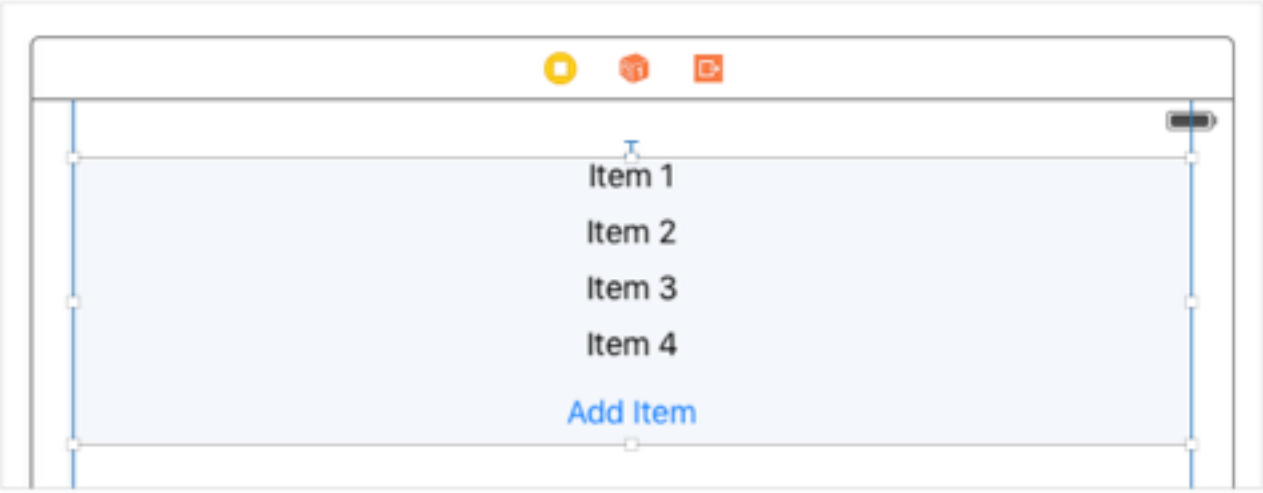


- **定义沿着 stack view的axis定义其的尺寸。**这种方式，沿着 stack view的axis固定相对于其父视图的两个边缘，定义了 stack view沿着其axis的尺寸。你还需要固定其他边界中的一个，来定义 stack view的位置。stack view将沿着它的axis改变尺寸和位置来填充定义的空间；然而，未固定的边界将根据其管理的最大视图的尺寸自由移动。

  例二，stack 视图的leading、top、trailing edges都已经相对于其父视图固定了。使用 UIStackViewDistributionFill 分布使得其内容重设尺寸来填充它的宽度，并且由于文本框有比标签更低的内容紧凑优先级，它将在必要的时候被拉伸。

- **定义垂直于 stack viewde axis的尺寸。**这种方式跟**例二**相似，但是你固定了垂直于 stack view的axis的两个边缘和沿着轴向的一个边缘。这使得 stack view在你增加或移除其管理的视图时将沿着其轴向扩展或收缩。除非你使用了 UIStackViewDistributionFillEqually 分布，被管理的视图将根据其原有尺寸调节尺寸。垂直于其axis，在定义的范围内，根据stack view的对齐方式来布局views。

  例三，展示了一个包含了四个标签和一个按钮的垂直 stack view。这个 stack view使用了8个点的间距和 UIStackViewAlignmentCenter 对齐方式。stack view的高度将根据 stack 内部的元素的增减而增大或收缩。



- **定义 stack view的位置和尺寸**。在这种方式中，你固定了 stack view的所有四个边缘。导致 stack view将在提供的范围之内布局它的内容。

  例四，展示了一个所有四个边缘都相对于其父视图固定的垂直 stack view。通过使用 UIStackViewAlignmentCenter 对齐方式和 UIStackViewDistributionFill 分布方式，stack view 确保其内容将水平居中和垂直填充屏幕。然而，获得期望的布局需要两个额外的步骤。默认情况下，stack view会垂直拉伸label而不是image view。要缩放图片控件，就要降低其内容紧凑优先级到低于标签。额外的，为了保持图片缩放时的长宽比，你必须设置图片视图的模式为 Aspect Fit。增加一个图片视图与 stack view之间宽度相等约束将有助于确保图片将被缩放来填充可用范围。

Managing the Stack View's Appearance

UIStackView 是 非渲染的UIView 的子类；它没有提供其自有的任何用户接口。相反的，它只管理被其管理的视图的位置和尺寸。因此，有些属性(如 backgroundColor)在 stack 视图上是无效的。类似的，你无法重写 layerClass，drawRect: 或 drawLayer:inContext: 方法。

这里有一系列的属性来定义 stack 视图如何布局其内容。

- axis(轴向) 属性决定了 stack 的朝向，只有垂直或水平；
- distribution(分布) 属性决定了其管理的视图在沿着其轴向上的布局；
- alignment(对齐) 属性决定了其管理的视图在垂直于其轴向上的布局；
- spacing(间距) 属性决定了其管理的视图间的最小间距；
- baselineRelativeArrangement 属性决定了其视图间的垂直间距是否根据基线测量得到；
- layoutMarginsRelativeArrangement 属性决定了 stack view布局其管理的视图时是否要参照它的布局边距

通常情况下，你会使用一个 stack view来布局小数量的视图。你可以通过在其他 stack 视图中嵌套多个 stack 视图的方式创建更加复杂的视图层次结构。

例五，展示了一个包含两个水平 stack view的垂直 stack view。每一个水平 stack view各包含一个label和一个text field。

你也可以通过对被管理的视图增加额外的约束，来完美的调节一个被管理视图的外观。举例说明，你可以使用约束类设置视图的最小或最大的高度或宽度。或者你可以为一个视图定义一个宽高比。当布局其内容时候，stack view将使用这些约束。举例来说，在**例四**中，image view有一个宽高比的约束，当图片尺寸更改时，image view也会更改。

注意

当给一个 stack view内的视图增加约束时要特别注意避免引入冲突。作为惯例，如果一个视图的尺寸在一个指定的维度上默认回到其原本内容尺寸，那么你可以安全的在这个维度上增加约束。

Maintaining Consistency Between the Arranged Views and Subviews

Stack view确保它的 arrangedSubviews 属性将一直是其 subviews 属性的子集。明确的说，stack view施行了以下规则：

- 当stack view增加了一个视图到它的 arrangedSubviews 数组的时候，其也将添加这个视图作为子视图，如果还未增加的话。
- 当一个子视图从 stack view中被移除的时候，那么 stack view也将将其从 arrangedSubviews 数组中移除。
- 从 arrangedSubviews 移除一个视图并不会将其作为子视图移除。stack view将不再管理该视图的尺寸和位置，但是该视图仍将是视图层级的一部分，并且假如这个视图可见的情况下仍会被渲染到屏幕上。

尽管 arrangedSubviews 数组一直包含着 subviews 数组的一个子集，这些数组间的顺序仍然是独立的。

- arrangedSubviews 数组的顺序定义了展现在 stack 中的视图的顺序。对于水平 stack view，这些视图将以阅读顺序布局，即较小索引的视图在较大索引视图的前方。在英语中，视图的布局是安按照从左至右。对于垂直 stack view，这些视图是从上到下平布局，及较小索引的视图在较大索引视图的上方。
- subviews 数组中的顺序定义了子视图在Z轴上是顺序。如果视图重叠，有较小索引的子视图将出现在有较大索引的子视图后方。

Dynamically Changing the Stack View's Content

当视图被加入、移除或插入 arrangedSubviews 数组时，或当一个被管理的子视图的 hidden 属性改变时，stack view都会自动更新它的布局。

```
UIView * firstView = self.stackView.arrangedSubviews[0];
firstView.hidden = YES;
```

stack view也会自动响应其任何属性的改变。举例，你可以更新 stack 视图的 axis 属性来动态改变的朝向。

```
if (self.stackView.axis == UILayoutConstraintAxisHorizontal)
{
    self.stackView.axis = UILayoutConstraintAxisVertical;
}
else {
```

```
        self.stackView.axis = UILayoutConstraintAxisHorizontal;
    }
```

你可以动态的同时更改被管理子视图的hidden属性和更改stack view的属性，通过在block动画中设置这些更改。

```
    [UIView animateWithDuration:0.25 animations:^{
        UIView * firstView = self.stackView.arrangedSubviews[0];
        firstView.hidden = YES;
    }];
```

最后，你可以直接在Interface Builder中为许多 stack view属性定义特定的 “size-class” 类型值。当stack view的size class改变的时候，系统会自动地动态修改这个改变。