ConstraintLayout

布局

居中

> xml 代码不要去记，看一遍就够了

居中于父容器

```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintBottom_toBottomOf="parent"
```

居中于控件中心

- 水平方向居中

```
app:layout_constraintStart_toStartOf="@id/view"
app:layout_constraintEnd_toEndOf="@id/view"
```

- 垂直方向居中

```
app:layout_constraintTop_toTopOf="@id/view"
app:layout_constraintBottom_toBottomOf="@id/view"
```

居中于控件的边

控件垂直居中于 `view` 的「下边」

```
app:layout_constraintTop_toBottomOf="@id/view"
app:layout_constraintBottom_toBottomOf="@id/view"
```

填充

水平方向填充父容器（通过`match_constraint`）

```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintEnd_toEndOf="parent"
android:layout_width="0dp"
```

备注：在早期版本中 `match_parent` 没有效果，必须使用 `match_constraint` 来完成。

权重

为水平方向的控件设置权重，大小为 `2:1:1`。

```
<!-- (view-1) -->
android:layout_width="0dp"
app:layout_constraintHorizontal_weight="2"

<!-- (view-2) -->
android:layout_width="0dp"
app:layout_constraintHorizontal_weight="1"

<!-- (view-3) -->
android:layout_width="0dp"
app:layout_constraintHorizontal_weight="1"
```

文字基准线对齐

```
app:layout_constraintBaseline_toBaselineOf
```

圆形定位

通过「圆心」「角度」「半径」设置圆形定位

```
app:layout_constraintCircle="@id/view"
app:layout_constraintCircleAngle="90"
app:layout_constraintCircleRadius="180dp"
```

特殊属性

约束限制

限制控件大小不会超过约束范围。

```
app:layout_constrainedWidth="true"
app:layout_constrainedHeight="true"
```

偏向

控制控件在垂直方向的 30%的位置

```
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintVertical_bias="0.3"
```

除了配合百分比定位，还有用于配合有时在「约束限制」的条件下不需要居中效果的情况

- 垂直方向居顶部

  ```
  app:layout_constraintTop_toTopOf="parent"
  app:layout_constraintBottom_toBottomOf="parent"
  app:layout_constrainedHeight="true"
  app:layout_constraintVertical_bias="0.0"
  ```

约束链

在约束链上的第一个控件上加上 `chainStyle`，用来改变一组控件的布局方式

- packed（打包）
- spread （扩散）
- spread_inside（内部扩散）

垂直方向 `packed`

```
app:layout_constraintVertical_chainStyle="packed"
```

宽高比

- 至少需要一个方向的值为 `match_constraint`
- 默认的都是宽高比，然后根据另外一条边和比例算出 `match_constraint` 的值

x:y 默认表示的都是 width:height，

有值的边，ratio 值（默认是宽高比），被约束的边（也有可能是有值的边）。

- 宽是 0dp，高是 100dp，ratio 是 2:1

默认情况下是宽是 200dp，但是我们可以指定被约束的边是 height，那么宽度就是50 dp

- 高是 0dp，宽是 100 dp，ratio 是 2:1

  默认情况下是高是 50 dp，但是我们指定被约束的边是 width，那么高度为200dp

只有一个方向`match_constraint`不要去用，如果是两个方向都是`match_constraint`那么可能会用到。

百分比布局

- 需要对应方向上的值为 `match_constraint`
- 百分比是`parent`的百分比，而不是约束区域的百分比

宽度是父容器的 30%

```
android:layout_width="0dp"
app:layout_constraintWidth_percent="0.3"
```

辅助控件

GuideLine

- 设置辅助线的方向 `android:orientation="vertical"`
- 设置辅助线的位置，根据方向不同
  - 距离左侧或上侧的距离`layout_constraintGuide_begin`
  - 距离右侧或下侧的距离`layout_constraintGuide_end`
  - 百分比 `layout_constraintGuide_percent`

Group

通过 `constraint_referenced_ids` 使用引用的方式来避免布局嵌套。

可以为一组控件统一设置 `setVisibility`

Layer

和 Group 类似，同样通过引用的方式来避免布局嵌套，可以为一组控件统一设置旋转/缩放/位移。

Barrier

通过设置一组控件的某个方向的屏障，来避免布局嵌套。

Placeholder

通过 setContentId 来将指定控件放到占位符的位置。

Flow

通过引用的方式来避免布局嵌套。

wrapMode

- chain
- aligned
- none(默认)

注意这个控件是可以被测量的，所以对应方向上的值可能需要被确定（即不能只约束同一方向的单个约束）

ConstraintSet

使用 ConstraintSet 对象来动态修改布局。

> 防止布局中有无 id 控件，可以设置`isForceId = false`

通过 `ConstraintSet#clone` 来从 `xml` 布局中获取约束集。

布局扁平化更加容易做过渡动画

在**布局修改之前** 通过加上 TransitionManager 来自动完成过渡动画。