使用动画移动视图

屏幕上的对象通常需要重新定位。这种情况可能是由于用户互动或幕后进行某些处理而导致的。您应使用动画将对象从起始位置移动到结束位置，而不应立即更新对象位置，立即更新对象位置会使对象从一个区域闪跳到另一个区域。

Android 提供了一些可让您在屏幕上重新定位视图对象的方法，例如 ObjectAnimator (/training/animation/UseObjectAnimator)。您可以提供希望对象停留的结束位置，以及动画的持续时间。您可以将此方法与时间插值器结合使用，以控制动画的加速或减速。

使用 ObjectAnimator 更改视图位置

`ObjectAnimator` (/reference/android/animation/ObjectAnimator) API 可用于轻松更改视图在指定时间段内的属性。它包含用于创建 **`ObjectAnimator`** (/reference/android/animation/ObjectAnimator) 实例的静态方法，具体取决于您要添加动画的属性类型。在屏幕上重新定位视图时，您将使用 **`translationX`** 和 **`translationY`** 属性。

以下示例 **`ObjectAnimator`** (/reference/android/animation/ObjectAnimator) 在 2 秒内将视图移到距离屏幕左侧 100 像素的位置：

KOTLIN (#KOTLIN)JAVA

```
ObjectAnimator animation = ObjectAnimator.ofFloat(view, "translationX", 100f);
animation.setDuration(2000);
animation.start();
```

此示例使用 **`ObjectAnimator.ofFloat()`** (/reference/android/animation/ObjectAnimator#ofFloat(T,%20android.util.Property%3CT,%20java.lang.Float%3E,%20android.util.Property%3CT,%20java.lang.Float%3E,%20android.graphics.Path)) 方法，因为平移值必须是浮点值。第一个参数是您想要添加动画的视图。第二个参数是您要添加动画的属性。由于需要水平移动视图，因此使用了 **`translationX`** 属性。最后一个参数是动画的结束值。该值为 100，表示距离屏幕左侧的像素数为 100。

下一个方法以毫秒为单位指定动画应持续的时间。在此示例中，动画将运行 2 秒（2000 毫秒）。

最后一个方法用于让动画开始运行，这会更新视图在屏幕上的位置。

如需详细了解如何使用 **`ObjectAnimator`** (/reference/android/animation/ObjectAnimator)，请参阅使用 ObjectAnimator 添加动画 (/guide/topics/graphics/prop-animation#object-animator)。

添加曲线动作

虽然使用 **`ObjectAnimator`** (/reference/android/animation/ObjectAnimator) 很方便，但默认情况下它会使用起点和终点之间的直线重新定位视图。Material Design 在确定曲线时不仅要考虑动画时长，还要考虑对象在屏幕上的空间运动。使用曲线动作有助于提升应用的真实感，让动画更有趣。

使用 PathInterpolator

`PathInterpolator` (/reference/android/view/animation/PathInterpolator) 类是 Android 5.0 (API 21) 中引入的新插值器。它基于贝塞尔曲线或 **`Path`** (/reference/android/graphics/Path) 对象。此插值器在一个 1x1 的正方形内指定一个动作曲线，定位点位于 (0,0) 和 (1,1)，而控制点则使用构造函数参数指定。如需创建 **`PathInterpolator`** (/reference/android/view/animation/PathInterpolator) 对象，可以创建 **`Path`** (/reference/android/graphics/Path) 对象并将其提供给 **`PathInterpolator`** (/reference/android/view/animation/PathInterpolator)：

KOTLIN (#KOTLIN)JAVA

```
// arcTo() and PathInterpolator only available on API 21+
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
  Path path = new Path();
```

```
    path.arcTo(0f, 0f, 1000f, 1000f, 270f, -180f, true);
    PathInterpolator pathInterpolator = new PathInterpolator(path);
  }
```

您也可以将路径插值器定义为 XML 资源：

```
<pathInterpolator xmlns:android="http://schemas.android.com/apk/res/android"
        android:controlX1="0.4"
        android:controlY1="0"
        android:controlX2="1"
        android:controlY2="1"/>
```

创建 **PathInterpolator** (/reference/android/view/animation/PathInterpolator) 对象后，您可以将其传递给 **Animator.setInterpolator()** (/reference/android/animation/Animator#setInterpolator(android.animation.TimeInterpolator)) 方法。然后，Animator 会使用插值器在其启动时确定时长或路径曲线。

KOTLIN (#KOTLIN)JAVA

```
ObjectAnimator animation = ObjectAnimator.ofFloat(view, "translationX", 100f);
animation.setInterpolator(pathInterpolator);
animation.start();
```

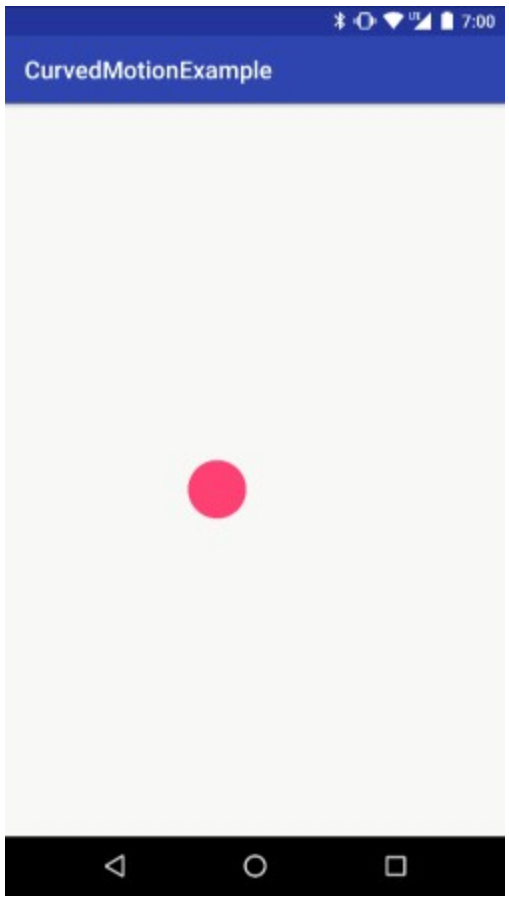
定义您自己的路径

ObjectAnimator (/reference/android/animation/ObjectAnimator) 类具有新的构造函数，可让您同时使用两个或多个属性以及路径，对路径上的多个坐标添加动画。例如，以下 Animator 使用 **Path** (/reference/android/graphics/Path) 对象为视图的 X 和 Y 属性添加动画：

KOTLIN (#KOTLIN)JAVA

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
  Path path = new Path();
  path.arcTo(0f, 0f, 1000f, 1000f, 270f, -180f, true);
  ObjectAnimator animator = ObjectAnimator.ofFloat(view, View.X, View.Y, path);
  animator.setDuration(2000);
  animator.start();
} else {
  // Create animator without using curved path
}
```

以下是弧形动画的效果：



如果您不想创建自己的时长或路径曲线，系统为 Material Design 规范中的三条基本曲线提供了 XML 资源：

- `@interpolator/fast_out_linear_in.xml`
- `@interpolator/fast_out_slow_in.xml`
- `@interpolator/linear_out_slow_in.xml`