

TweenAnimation(补间动画)

有时也称为 (view 动画)

1、基本介绍

tweenAnimation 是2.x推出的动画，现在用的比较少，因为4.x推出了PropertyAnimation, 可以认为是改良版的tweenAnimation。

注意，这种动画不会对view的真实参数造成改变

2、动画类型

四种类型

scale 尺寸变化
rotate 旋转变化
alpha 透明度变化
translate 平移变化

同样：

实现的方式有两种，一种是xml文件，一种是java代码

xml文件对应动画的标签

(xml文件时定义在res/anim目录下的，需要自己创建)

```
<scale>  
<rotate>  
<alpha>  
<translate>
```

java代码对应动画的类

```
ScaleAnimation  
RotateAnimation  
AlphaAnimation  
TranslateAnimation
```

3、Animation通用方法介绍

Animation 属性:

xml属性	jav代码	作用
<code>android:detachWallpaper</code>	<code>setDetachWallpaper(boolean)</code>	是否在壁纸上运行
<code>android:duration</code>	<code>setDuration(long)</code>	动画的持续时间
<code>android:fillAfter</code>	<code>setFillAfter(boolean)</code>	动画结束后是否停留在结束位置
<code>android:fillBefore</code>	<code>setFillBefore(boolean)</code>	动画结束时是否还原开始位置
<code>android:fillEnabled</code>	<code>setFillEnabled(boolean)</code>	同上, 与fillBefore相同
<code>android:interpolator</code>	<code>setInterpolator(Interpolator)</code>	设置插值器
<code>android:repeatCount</code>	<code>setRepeatCount(int)</code>	重复次数
<code>android:repeatMode</code>	<code>setRepeatMode(int)</code>	有两种重复类型, <code>reverse</code> 倒序回放, <code>restart</code> 从头播放
<code>android:startOffset</code>	<code>setStartOffset(long)</code>	开启动画 <code>startAnimation(animation)</code> 之后等待执行运行动画的时间
<code>android:zAdjustment</code>	<code>setZAdjustment(int)</code>	表示被设置动画的内容运行时在Z轴上的位置 (<code>top/bottom/normal</code>), 默认为 <code>normal</code>

注意:

`startOffset` 的含义就是delay, 延迟多久才开始执行动画

`repeatMode` 需要配合`repeatCount`使用, 如果`repeatCount=1`, `repeatMode`是没有效果的

因为这是二维的动画类型, 因此Z轴上是不会有变化的

差值器是用来改变动画执行过程的速度变化的

`fillBefore` 和 `fillAfter`的区别

第一个表示动画结束后回到第一帧, 后一个表示动画结束后停留在最后一帧
都是属于补间动画的, 在set结点使用, 单独使用在 `scale`, `rotate`, `translate`, `alpha`是没有效果的。

4、xml方式实现

(1) 缩放

```
<scale
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXScale="1"
    android:fromYScale="1"
    android:toXScale="2"
    android:toYScale="2"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="300"/>
```

fromXScale, fromYScale

表示动画初始状态，view 的x轴和y轴的缩放都是view自身大小的1.0倍

toXScale, toYScale

表示动画结束状态，view 的x轴和y轴的缩放都是view自身大小的2.0倍

privotX, privotY

表示动画开始变化的位置，50%表示x轴的中心，y轴的中心。整体就是原本view的中心点（这里可以直接使用数字，10, 300, 等，表示距离左上角多少个像素开始变化）

上面的最终效果

从View的中心开始变化，在300毫秒时间内，将View从原本大小的1倍开始放大，直到放大为原本View的2倍

（2）透明度

```
<alpha
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromAlpha="0"
    android:toAlpha="1"
    android:duration="300"/>
```

fromAlpha （有效值是0到1）

表示动画初始时View的透明度值，0是完全看不见

toAlpha

表示动画结束时View的透明度值，1是完全看见

上面的最终效果

在300毫秒内，View的透明度从完全不可见到完全可见

（3）旋转

```
<rotate
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromDegrees="0"
    android:toDegrees="360"
    android:pivotX="50"
    android:pivotY="100"
    android:duration="300"/>
```

fromDegrees

动画初始时，View的偏移度数（0表示不偏移，>0表示顺时针偏移，<0表示逆时针偏移）

toDegrees

动画结束时，View的偏移度数

pivotX , pivotY

前面介绍过了

上面的最终效果

在300毫秒内，从距离View x轴50个像素点，距离y轴100个像素点的位置开始变化，将一个view从不偏移角度向顺时针方向偏移360度

（4）平移

```
<translate
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXDelta="0"
    android:toXDelta="100%"
    android:fromYDelta="0"
    android:toYDelta="100%"
    android:duration="300"/>
```

fromXDelta

表示动画初始时，View在X轴的平移距离（0表示不偏移）

fromYDelta

表示动画初始时，View在Y轴的平移距离（如果是10，则表示在Y轴向下偏移10个像素点）

toXDelta

表示动画结束时，View在X轴的平移距离（100%表示向X轴的正方向偏移原本View的宽度的1倍距离）

toYDelta

表示动画结束时，View在Y轴的平移距离（如果是-100，表示向Y轴的负方向偏移100个像素点）

上面的最终效果

在300毫秒内，将一个View从X轴正方向偏移他自身宽度的一倍距离，从Y轴正方向上偏移他自身高度的一倍距离（也就是将一个View向他的右下角偏移，直到他的左上角位置和原本的右下角位置重叠）

（5）加载

最终需要利用AnimationUtils类来加载xml文件

// 加载xml文件为一个Animation 类

Animation anim = AnimationUtils.loadAnimation(context, R.anim.rotate);

// 开始动画

targetView.startAnimation(anim);

注意

因为我们没有调用anim的其他方法设置相关配置，因此当动画结束后，view就会变回原本的状态，如果有需要，根据上面介绍的Animation的方法来设置。同时还要注意就算设置了fillAfter(true)，view的任何实际状态还是没有改变，这就是所谓的tweenAnimation不改变View的任何属性值的原因

(6) 动画集合

每次只能将一种动画设置到一个View中，显得有点太平淡了，可以通过<set>标签，设置多种动画

```
<set android:duration="1000"
  xmlns:android="http://schemas.android.com/apk/res/android">
  <scale
    android:fromXScale="0"
    android:fromYScale="0"
    android:toXScale="1"
    android:toYScale="1"
    android:pivotX="50%"
    android:pivotY="50%" />
  <rotate
    android:fromDegrees="-360"
    android:toDegrees="360"
    android:pivotX="50%"
    android:pivotY="50%" />
  <alpha
    android:fromAlpha="0"
    android:toAlpha="1" />
</set>
```

三种动画在1000毫秒内同时完成

加载方式和上面介绍的一样

5、java代码实现

(1) ScaleAnimation

Candidates for new **ScaleAnimation()** are:
[ScaleAnimation\(Context context, AttributeSet attrs\)](#)
[ScaleAnimation\(float fromX, float toX, float fromY, float toY\)](#)
[ScaleAnimation\(float fromX, float toX, float fromY, float toY, float pivotX, float pivotY\)](#)
[ScaleAnimation\(float fromX, float toX, float fromY, float toY, int pivotXType, float pivotXValue, int pivotYType, ...\)](#)

(2) RotateAnimation

Documentation for ()
animationdemo
Candidates for new **RotateAnimation()** are:
[RotateAnimation\(Context context, AttributeSet attrs\)](#)
[RotateAnimation\(float fromDegrees, float toDegrees\)](#)
[RotateAnimation\(float fromDegrees, float toDegrees, float pivotX, float pivotY\)](#)
[RotateAnimation\(float fromDegrees, float toDegrees, int pivotXType, float pivotXValue, int pivotYType, float pivotYValue\)](#)

(3) AlphaAnimation

Candidates for new **AlphaAnimation()** are:

[AlphaAnimation\(Context context, AttributeSet attrs\)](#)
[AlphaAnimation\(float fromAlpha, float toAlpha\)](#)

(4) TranslateAnimation

Candidates for new **TranslateAnimation()** are:

[TranslateAnimation\(Context context, AttributeSet attrs\)](#)
[TranslateAnimation\(float fromXDelta, float toXDelta, float fromYDelta, float toYDelta\)](#)
[TranslateAnimation\(int fromXType, float fromXValue, int toXType, float toXValue, int fromYType, float fromYValue, int toYType, ...\)](#)

使用例子

```
ScaleAnimation animation = new ScaleAnimation(0, 1, 0, 1);  
animation.setDuration(1000);  
tweenView.startAnimation(animation);
```

(5) 集合 (AnimationSet)

Candidates for new **AnimationSet()** are:

[AnimationSet\(Context context, AttributeSet attrs\)](#)
[AnimationSet\(boolean shareInterpolator\)](#)

使用

```
AnimationSet set = new AnimationSet(false);  
set.addAnimation(new AlphaAnimation(0,1));  
set.addAnimation(new RotateAnimation(0,360));  
set.setDuration(1000);  
tweenView.startAnimation(set);
```

6、动画监听器

注意这是Animation 类的动画监听器
和后面介绍的Animator 类 是不一样的

```

anim5.setAnimationListener(new Animation.AnimationListener()
{
    @Override
    public void onAnimationStart(Animation animation)
    {
        Log.w("tween", "start-----");
    }
    @Override
    public void onAnimationEnd(Animation animation)
    {
        Log.w("tween", "end-----");
    }
    @Override
    public void onAnimationRepeat(Animation animation)
    {
        Log.w("tween", "repeat-----");
    }
});

```

动画开始

动画结束

动画重复

动画监听器，
三个回调

targetView.startAniamtion(anim5);

英