动画类型：

1 Tween Animation(补间动画也是常用的view的动画)。

2 Frame Animation(帧动画，从某种意义上来讲也属于补间动画，只是动画的表示的形式不同).

3 Property Animation(属性动画，Android 3.0 API 11以后引进的).

1 Tween Animation ：补间动画很简单明了，我们只需要去关注开始和结束的“关键帧”，其中间的过程西东直接帮我们去操作了，不需要我们“一帧帧”去定义了。

Android使用Animation代表抽象动画，包括四种子类：AlphaAnimation(透明度动画)、ScaleAnimation(缩放动画)、TranslateAnimation(位移动画)、RotateAnimation(透明度动画)。Android里面允许在java中创建Animation类对象，但是一般都会采用动画资源文件来定义动画，把界面与逻辑分离，同时也建议大家通过xml的方式来实现补间动画。

2 Frame Animation : 这个很好理解，一帧帧的播放图片，利用人眼视觉残留原理，给我们带来动画的感觉。它的原理的GIF图片、电影播放原理一样，其实早期的电影也正是用一张张图片的快速滚动来实现的，当然可以在xml中定义也可以在java代码中定义。

3 Property Animation :属性动画，这个是在Android 3.0中才引进的，它可以直接更改我们对象的属性。在上面提到的Tween Animation中，只是更改View的绘画效果而View的真实属性是不改变的。假设你用Tween动画将一个Button从左边移到右边，无论你怎么点击移动后的Button，他都没有反应。而当你点击移动前Button的位置时才有反应，因为Button的位置属性木有改变。而Property Animation则可以直接改变View对象的属性值，这样可以让我们少做一些处理工作，提高效率与代码的可读性。

（1）ValueAnimator：包含Property Animation动画的所有核心功能，如动画时间，开始、结束属性值，相应时间属性值计算方法等。应用ValueAnimator有两个步骤

1计算属性值。

2根据属性值执行相应的动作，如改变对象的某一属性。

我们的主是第二步，需要实现ValueAnimator.onUpdateListener接口，这个接口只有一个函数onAnimationUpdate()，将要改变View对象属性的事情在该接口中do。

animation.addUpdateListener(new AnimatorUpdateListener() {

@Override

public void onAnimationUpdate(ValueAnimator animation) {

//do your work

}

});

（2）ObjectAnimator：继承自ValueAnimator，要指定一个对象及该对象的一个属性，当属性值计算完成时自动设置为该对象的相应属性，即完成了Property Animation的全部两步操作。实际应用中一般都会用ObjectAnimator来改变某一对象的某一属性，但用ObjectAnimator有一定的限制，要想使用ObjectAnimator，应该满足以下条件：

1.对象应该有一个setter函数：set（驼峰命名法，大家可以自行百度）

2如下面的例子，像ofFloat之类的工场方法，第一个参数为对象名，第二个为属性名，后面的参数为可变参数，如果values…参数只设置了一个值的话，那么会假定为目的值，属性值的变化范围为当前值到目的值，为了获得当前值，该对象要有相应属性的getter方法：get

3如果有getter方法，其应返回值类型应与相应的setter方法的参数类型一致。

ObjectAnimator oa=ObjectAnimator.ofFloat(tv, alpha, 0f, 1f);

oa.setDuration(3000);

oa.start();

（3）Animator.AnimatorListener：可以为Animator设置动画监听，需要重写下面四个方法。

onAnimationStart()

onAnimationEnd()

onAnimationRepeat()

onAnimationCancel()

这里我们也可以实现AnimatorListenerAdapter，他的好处是可以只用定义想监听的事件而不用实现每个函数却只定义一空函数体。如下：

anim.addListener(new AnimatorListenerAdapter() {

public void on AnimationEnd(Animator animation){

//do your work

}

});

（4）AnimationSet：可以组合多个动画共同工作

AnimatorSet bouncer = new AnimatorSet();

bouncer.play(anim1).before(anim2);

bouncer.play(anim2).with(anim3);

bouncer.play(anim2).with(anim4)

bouncer.play(anim5).after(amin2);

animatorSet.start();

上面的代码意思是: 首先播放anim1；同时播放anim2,anim3,anim4；最后播放anim5。

（5）TimeInterplator：与Tween中的interpolator类似。有以下几种先简单介绍下：

AccelerateInterpolator　　　　　 加速，开始时慢中间加速

DecelerateInterpolator　　　 　　 减速，开始时快然后减速

AccelerateDecelerateInterolator　 先加速后减速，开始结束时慢，中间加速

AnticipateInterpolator　　　　　　 反向 ，先向相反方向改变一段再加速播放

AnticipateOvershootInterpolator　 反向加回弹，先向相反方向改变，再加速播放，会超出目的值然后缓慢移动至目的值

BounceInterpolator　　　　　　　 跳跃，快到目的值时值会跳跃，如目的值100，后面的值可能依次为85，77，70，80，90，100

CycleIinterpolator　　　　　　　　 循环，动画循环一定次数，值的改变为一正弦函数：Math.sin(2 \* mCycles \* Math.PI \* input)

LinearInterpolator　　　　　　　　 线性，线性均匀改变

OvershottInterpolator　　　　　　 回弹，最后超出目的值然后缓慢改变到目的值

TimeInterpolator　　　　　　　　 一个接口，允许你自定义interpolator，以上几个都是实现了这个接口

（6）Keyframes：可以让我们定义除了开始和结束以外的关键帧。KeyFrame是抽象类，要通过ofInt(),ofFloat(),ofObject()获得适当的KeyFrame，然后通过PropertyValuesHolder.ofKeyframe获得PropertyValuesHolder对象，如下：

Keyframe kf0 = Keyframe.ofInt(0, 400);

Keyframe kf1 = Keyframe.ofInt(0.25f, 200);

Keyframe kf2 = Keyframe.ofInt(0.5f, 400);

Keyframe kf4 = Keyframe.ofInt(0.75f, 100);

Keyframe kf3 = Keyframe.ofInt(1f, 500);

PropertyValuesHolder pvhRotation = PropertyValuesHolder.ofKeyframe(width, kf0, kf1, kf2, kf4, kf3);

ObjectAnimator rotationAnim = ObjectAnimator.ofPropertyValuesHolder(btn, pvhRotation);

上述代码的意思是：设置btn对象的width属性值使其：开始时 Width=400，动画开始1/4时 Width=200，动画开始1/2时 Width=400，动画开始3/4时 Width=100，动画结束时 Width=500。

（7）ViewPropertyAnimator：对一个View同时改变多种属性，非常推荐用这种。该类对多属性动画进行了优化，会合并一些invalidate()来减少刷新视图。而且使用起来非常简便，但是要求API LEVEL 12，即Android 3.1以上。仅需要一行代码即可完成水平、竖直移动

myView.animate().translationX(50f). translationY(100f);

（8）常需要改变的一些属性：

translationX,translationY: View相对于原始位置的偏移量

rotation,rotationX,rotationY: 旋转，rotation用于2D旋转角度，3D中用到后两个

scaleX,scaleY: 缩放比

x,y: View的最终坐标，是View的left，top位置加上translationX，translationY

alpha: 透明度

三 各个动画的优缺点：

(1) Tween Animation(补间动画)仅需定义开始与结束的关键帧，而变化的中间帧由系统补上，优点是不用准备每一帧，缺点是只改变了对象绘制，而没有改变View本身属性。因此如果改变了按钮的位置，还是需要点击原来按钮所在位置才有效。

(2 )Frame Animation( 帧动画 ) 主要用于播放一帧帧准备好的图片，类似 GIF 图片，优点是使用简单方便、缺点是需要事先准备好每一帧图片，而且如果图片过大容易导致OOM，所以用的时候要谨慎；

(3) Property Animation(属性动画)是3.0后推出的动画，优点是使用简单、降低实现的复杂度、直接更改对象的属性、几乎可适用于任何对象而仅非View类，缺点是需要3.0以上的API支持，限制较大！但是目前国外有个开源库nineoldandroids，可以提供低版本支持大家可以研究下！