Android坐标与动画

# 需求与背景

## 需求1

1. 描述：在一个空间中，从随机布点的位置（互不重复）向中心点聚集，动画效果是由大到小，由清晰到模糊。

涉及知识点：getX(), getLeft(), getLayoutParam(), getY(), getRight(), getTop(), getTranslateX(), getTranslateY(), getMeasureWidth(), getMeasureHeight(), setScaleX(), setScaleY(), setTranslateX(), setTranslateY(), getWidth(), getHeight()。

解决步骤：

* + 1. setTranslateX
    2. Asdfaf

待解决问题：对所涉及知识点的了解。

## 需求2

# 基本知识

## 获取View的宽高

1. getWidth()是实际高度，getMeasureWidth()是XML文件中配置的宽度。通常getWidth()为0，不知道原因。GetWidth()的值等于getRight() - getLeft().

## 获取View相对父View的位置

* + 1. GetRight()，右边界到父View的左边界的距离
    2. GetLeft()，左边界到父View的左边界距离
    3. GetButtom()，底部到父View的顶部距离
    4. getX()和getY()是获取点击事件距离件左边或上边的距离，即视图坐标。
    5. getRaw()和getRawY()是获取到的是点击事件距离整个屏幕左边的距离，即绝对坐标。

## 获取View相对屏幕位置

## 平移View

* + 1. x方向平移
    2. y方向平移

# 通用解决方法

## 问题类型1

* + 1. 解决步骤
    2. 涉及知识

## 问题类型2

# 自定义View

## View的绘制流程

1. View在Activity中显示出来，要经历测量、布局和绘制三个步骤，分别对应三个动作：measure、layout和draw。测量：onMeasure()决定View的大小；

布局：onLayout()决定View在ViewGroup中的位置； 绘制：onDraw()决定绘制这个View。

1. 什么时候调用onMeasure方法？ 当控件的父元素正要放置该控件时调用View的onMeasure()。ViewGroup会问子控件View一个问题：“你想要用多大地方啊？”，然后传入两个参数— —widthMeasureSpec和heightMeasureSpec；这两个参数指明控件可获得的空间以及关于这个空间描述的元数据。更好的方法是传递子控件View的高度和 宽度到setMeasuredDimension()里，直接告诉父控件需要多大地方放置子控件。在onMeasure()的最后都会调用setMeasuredDimension()；如果不调用，将 会由measure()抛出一个IllegalStateException()。自定义View的onMeasure()： --> 测量View的大小。系统帮我们测量的高度和宽度都是MATCH\_PARENT；当我们设置明确的宽度和高度时，系统测量的结果就是我们设置的结果。当设置为WRAP\_CONTENT，或者是MATCH\_PARENT时，系统测量的结果就是MATCH\_PARENT的长度。
2. 参考网址：<https://www.cnblogs.com/CVstyle/p/6399188.html>

# 动画

Android的动画分为三种：View动画（平移TranslateAnimation、旋转RotateAnimation、透明度AlphaAnimation，缩放ScaleAnimation）、帧动画和属性动画。属性动画是Android11后才有的特征。

|  |  |  |
| --- | --- | --- |
| 类别 | 子类 | 用法 |
| View动画 |  |  |
| 属性动画 | ValueAnimator | 不作用于任何对象，对数值做动画。 |
| ObjectAnimator | 必须指定作用对象和属性。 |

## 属性动画

1. 属性动画有两个监听器：AnimatorListener， AnimatorUpdateListener（动画是由许多帧组成的，没播放一帧，就会调用一次。）。相比View动画，属性动画支持对任意属性做动画，比如宽度。属性动画要求作用的对象提供该属性的get和set方法，即对对象的”abc”做动画，此对象必须由setAbc和getAbc方法。
2. 插值器和估值器：Interpolator、 TypeEvaluator，属性动画中，此二者非常重要。它们是实现非匀速动画的重要手段。[参考链接1](https://blog.csdn.net/carson_ho/article/details/72863901)。目前系统已经提供的插值器包括加速、先加速再减速、最后阶段弹球效果、周期运动、减速、匀速、缓冲前进冲出终点后再回到终点，可以满足大部分动画要求。 比如，线性插值器的返回值和输入值一样（看getInterpolation方法），因此t=20ms时插值器返回的值是0.5，这意味着x属性的改变是0.5。这个时候插值器的工作已经完成了，插值器的工作就是根据时间流逝的百分比计算出当前属性值改变的百分比。我们得到了当前属性改变的百分比是0.5，即50%。下一步就是要算出x具体变为了什么值，这个时候估值算法就起作用了，它的作用就是根据当前属性改变的百分比来计算改变后的属性值。
3. 自定义插值器只需要实现Interpolator / TimeInterpolator接口并复写getInterpolation。
4. 估值器协助插值器用于实现非线性运动的动画效果。自定义估值器需要实现 TypeEvaluator接口并复写evaluate()。
5. 使用PropertyValuesHolder实现各个属性同步动画。
6. 使用Keyframe实现将动画按时间拆分。例如：



## 注意事项

1. 做动画过程中，建议开启硬件加速，以提高动画的流畅性。动画中尽量使用dp，因为px会导致在不同手机上表现不一致。
2. Viw动画并未改变View的状态即属性（x，y）的值，因此有时出现动画完成后，view.setVisibility(Gone)失效的情况，这时需要调用view.clearAnimation()。
3. 属性动画中的无限循环的动画，需要在activity推出时停止，否则此activity无法释放，view动画不存在此问题。属性动画会永久改变View在内存中的状态（位置、alpha等），直到重现加载此view。
4. animation和animator最好不要混用。尽量用属性动画（animator），不要用View动画（animation）。
5. Animation 是针对视图外观的动画实现，动画被应用时外观改变但视图的触发点不会发生变化，还是在原来定义的位置。 Animator  是针对视图属性的动画实现，动画被应用时对象属性产生变化，最终导致视图外观变化。
6. 属性动画（ObjectAnimator）无限循环，发现会出现停止不了动画的情况下。

## 比较好的动画效果

1. [浮动按钮动画效果](https://blog.csdn.net/qq_22770457/article/details/71249181?utm_source=gold_browser_extension)。
2. [联动动画](https://blog.csdn.net/u012045061/article/details/69568807)。
3. [NestedScrolling滑动机制](https://www.cnblogs.com/wjtaigwh/p/6398562.html)。

# Drawable

## Layer-list

## Selector

## shape