# View

**View 的位置参数**

left：View 父容器左上角横坐标

top：View 父容器左上角纵坐标

right：View 父容器右下角横坐标

bottom：View 父容器右下角纵坐标

x 和 y 代表的是 View 左上角的坐标。x 轴 和 y 轴的正方向分别为右和下。

translationX 和 translationY 是 View 左上角相对于父容器的偏移量。

x=left+translationX

**MotionEvent**

在手指接触屏幕后所产生的一系列事件中，典型的时间类型有如下三种：

ACTION\_DOWN：手指刚接触屏幕的一瞬间。

ACTION\_MOVE：手指在屏幕上移动。

ACTION\_UP：手指从屏幕上松开的一瞬间。

|  |  |
| --- | --- |
| getX() / getY() | 返回相对于当前 View 的左上角 x 和 y 坐标。 |
| getRawX() / getRawY() | 返回相对于手机屏幕左上角的 x 和 y 坐标。 |

**TouchSlop**

TouchSlop 是系统所能识别出的被认为是滑动的最小距离。

ViewConfiguration.get(context).getScaledTouchSlop()

**VelocityTracker**

VelocityTracker 用于做手指在滑动过程中的速度追踪，包括水平和竖直方向的速度。

使用：

首先，在 View 的 onTouchEvent 方法中追踪当前单击事件的速度

val velocityTracker = VelocityTracker.obtain()

velocityTracker.addMovement(event)

接着，当我们想知道当前的滑动速度时，这个时候可以采用如下方法来获得当前速度：

// 计算滑动速度，计算的时间间隔为 1000ms

// 也即，计算 1000ms 内手指所滑动的像素数

velocityTracker.computeCurrentVelocity(1000)

val xVelocity = velocityTracker.getXVelocity()

val yVelocity = velocityTracker.getYVelocity()

当手指从右往左滑动时，水平速度即为负值

在不需要使用它时，需要调用clear()方法来重置并回收内存

velocityTracker.clear()

velocityTracker.recycle()

**GestureDetector**

用于辅助检测用户的单击、滑动、长按、双击等行为。一般情况下我们需要处理用户手指的单击、长按和滑动操作，可以直接在 onClick()、onLongClick() 和 onTouchEvent() 中处理即可，如果还需要 View 处理用户的双击操作，则需要借助 GestureDetector。

**scrollTo和scrollBy**

为了实现 View 的滑动，View 提供了专门的方法来实现这个功能，那就是 scrollTo() 和 scrollBy()，

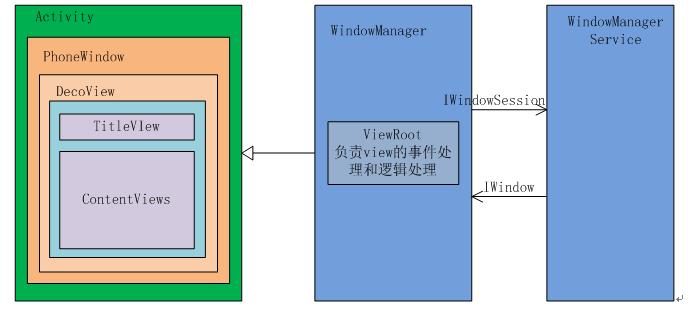
mScrollX：View 左边缘-内容左边缘的在水平方向的距离

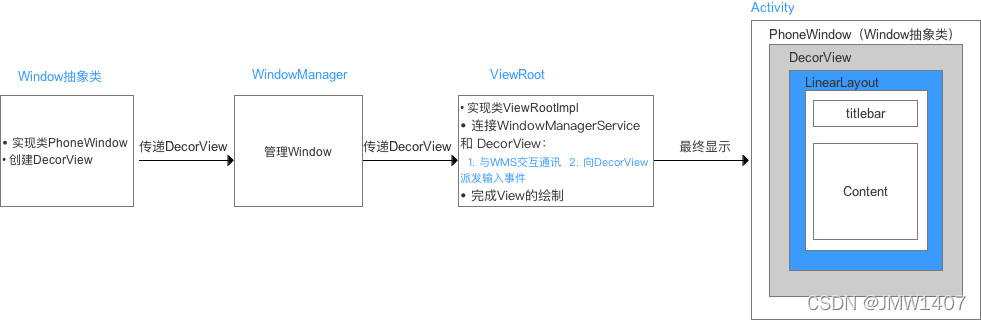
mScrollY：View 上边缘和内容上边缘的在水平方向的距离

**Scroller**

Scroller 是一个对象，用于实现 View 的弹性滑动（一定的时间间隔内，不是瞬时的）。

# UI管理系统的层级关系





PhoneWindow：Android系统中最基本的窗口系统，继承自Windows类，负责管理界面显示以及事件响应。它是Activity与View系统交互的接口

DecorView：PhoneWindow中的起始节点View，继承于View类，作为整个视图容器来使用。用于设置窗口属性。它本质上是一个FrameLayout

ViewRoot：Activtiy启动时创建，负责管理、布局、渲染窗口UI等等

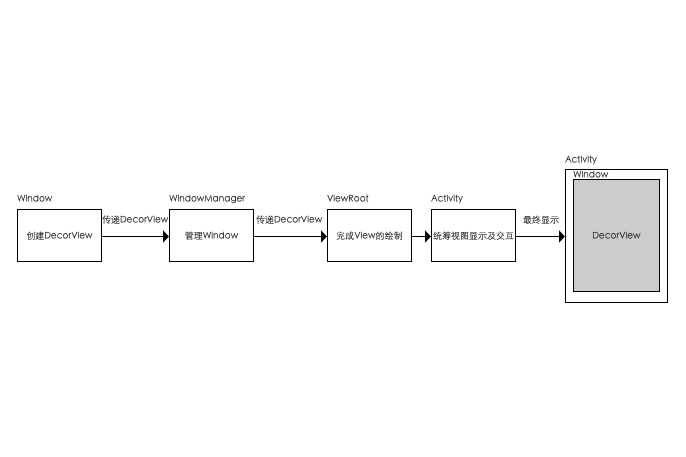
**ViewRoot与DecorView**

ViewRoot是连接WIndowManager和DecorVIew的纽带。View的三大流程(measure layout draw)都是通过ViewRoot完成的。

在ActivityThread中，Activity创建后，DecorView会被添加到Window中，同时创建ViewRootImpl，然后将DecorView与ViewRootImpl建立关联(通过ViewRoot的setView方法)。

**ViewRoot、DecorView、Window和Activity的关系**





****“activity，window，View 三者之间的关系是什么？”****

1、window 是 activity 里的一个实例变量，本质是一个接口，唯一的实现类是 PhoneWindow。

2、activity 的 setContentView 方法实际上是就是交给 phonewindow 去做的。window 和 View的关系可以类比为显示器和显示的内容。

3、每个 activity 都有一个“显示器” window，“显示的内容”就是DecorView。这个“显示器”定义了一些方法来决定如何显示内容。比如 setTitleColor setTitle是设置导航栏的颜色和 title , setAllowReturnTransitionOverlap 设置进/出场动画等等。

****所以 window 是 activity 的一个成员变量，window 和 View 是“显示器”和“显示内容”的关系。****