**Input 事件在view系统中处理的流程**

**1.流程**

**InputManager -->InputChannel ---->viewrootImp**

**Viewrootimp 在初始化的时候创建一个InputStage链表**

**ViewPostImeInputStage 处理 input events to the view hierarchy**

**onTouchEvent 返回false继续分发input给子view，返回true停止分发input事件**

**Activity：**

**dispatchTouchEvent**

**onTouchEvent**

**ViewGrop：**

**dispatchTouchEvent**

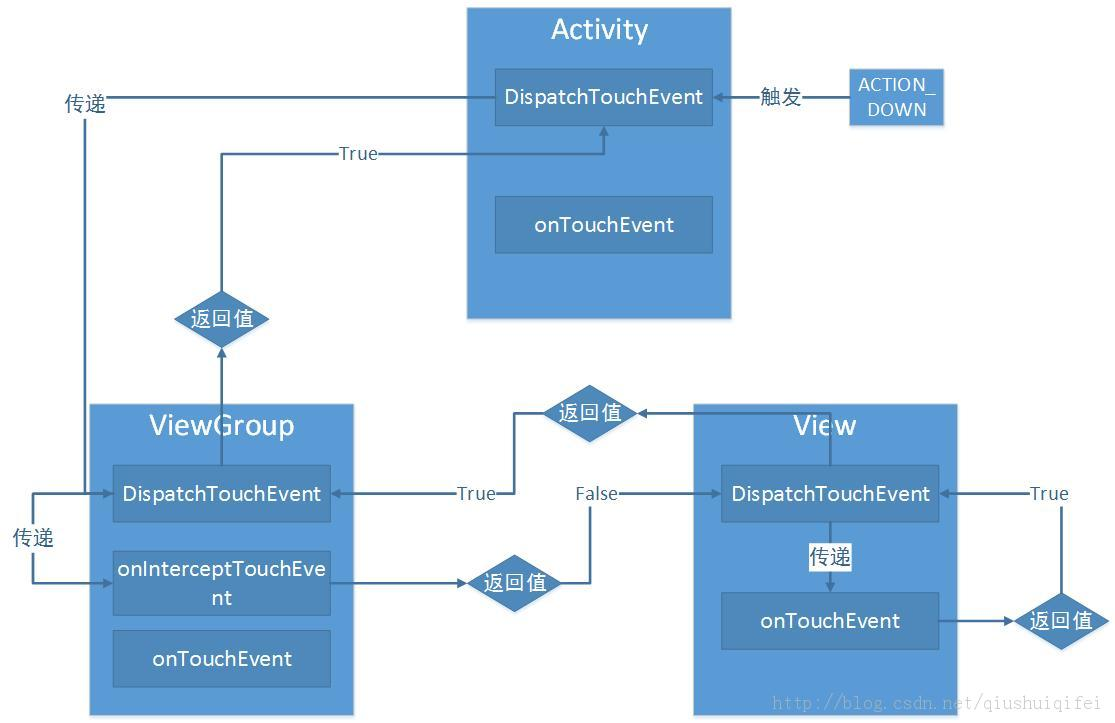
**onInterceptTouchEvent**

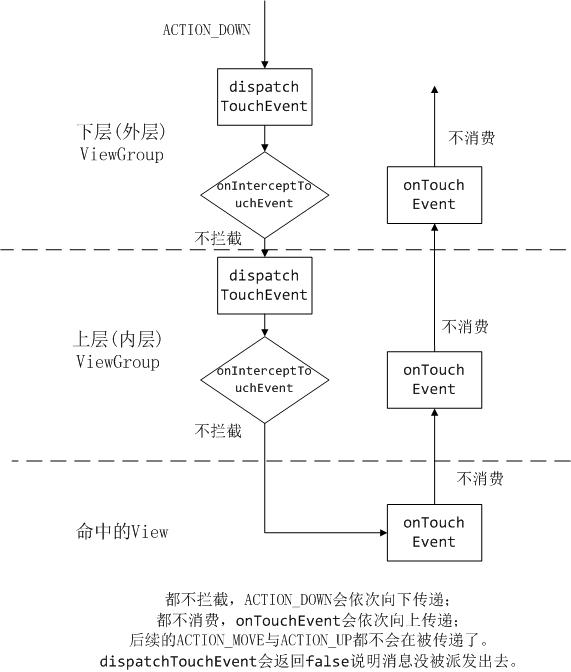
**onTouchEvent**

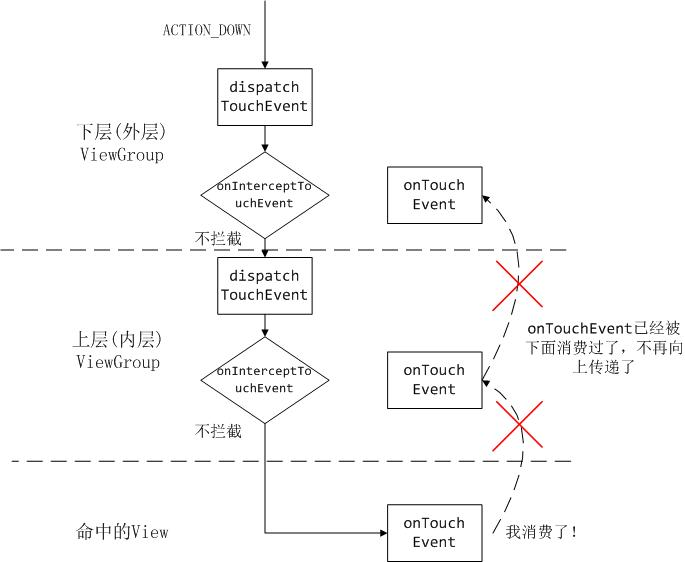
**View：**

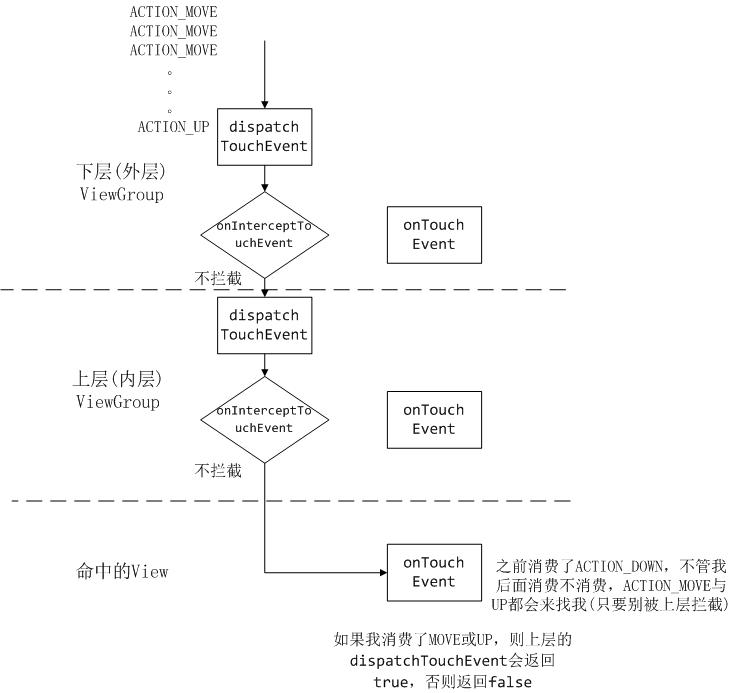
**dispatchTouchEvent**

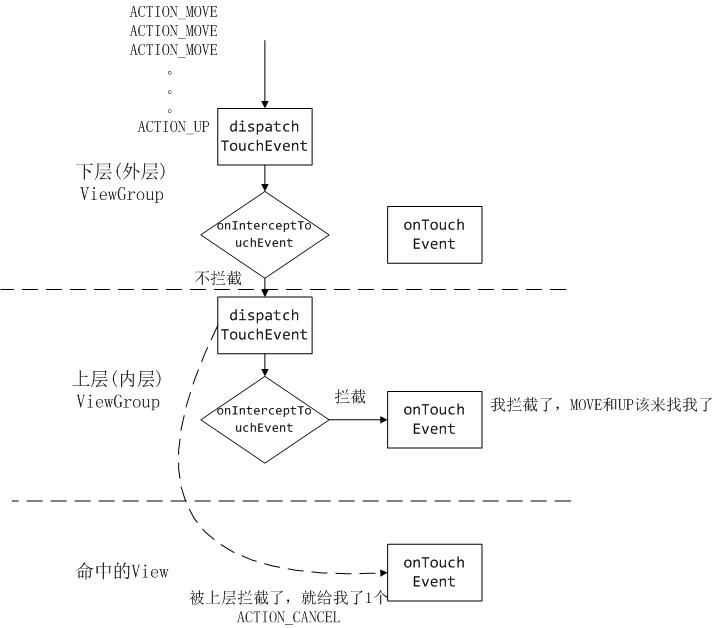
**onTouchEvent**

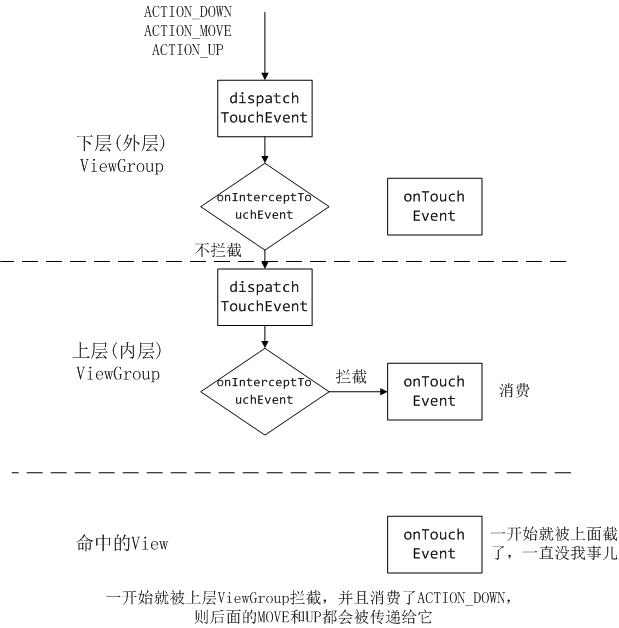












**SystemUI view的关系：**

**StatusBarWindowView**

**PhoneStatusBarView**

**NotificationPanelView**

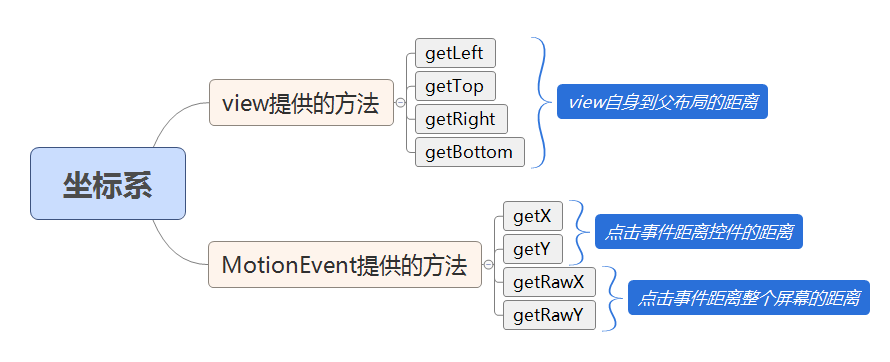
**NotificationsQuickSettingsContainer**

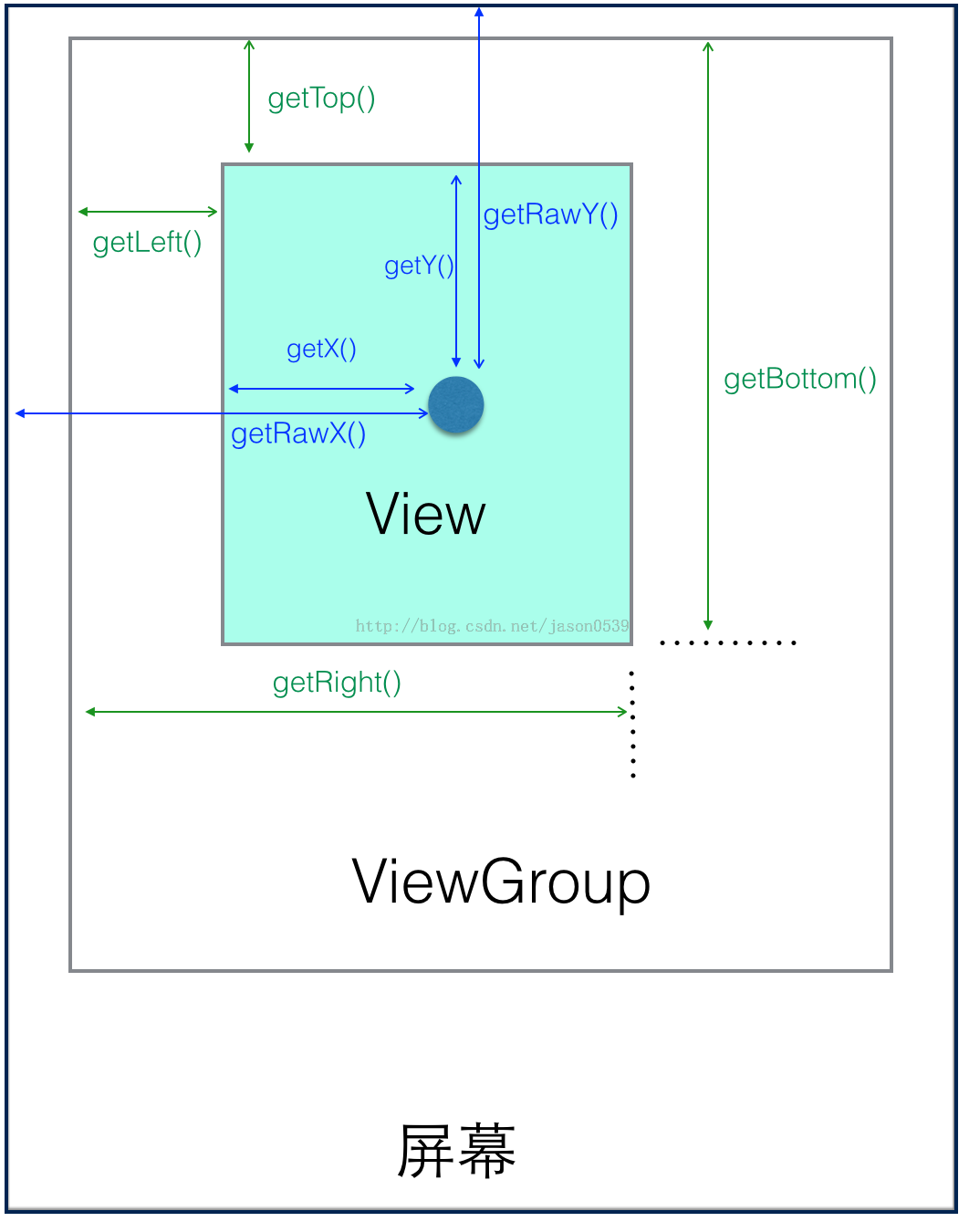
**ObservableScrollView**

**QSContainer**

**QSPanel**

**NotificationStackScrollLayout**







1. **当onInterceptTouchEvent函数拦截了一个事件，而且onTouchEvent函数返回false**

**说明没有消费相应的触摸事件，后续的move up事件也不会再传递。**

1. **当onInterceptTouchEvent函数拦截了一个事件，而且onTouchEvent函数返回true**

**Down事件会被消费，后续的move和up事件也会被派送。**

1. **onInterceptTouchEvent决定谁的onTouchEvent函数来处理event。（由上到下进行截断）**

**onTouchEvent 决定事件是否继续由下向上传递。（由下到上进行截断）**

**"main@6758" prio=5 tid=0x1 nid=NA runnable**

**java.lang.Thread.State: RUNNABLE**

**at android.view.View.dispatchTouchEvent(View.java:9963)**

**at android.view.ViewGroup.dispatchTransformedTouchEvent(ViewGroup.java:2669)**

**at android.view.ViewGroup.dispatchTouchEvent(ViewGroup.java:2344)**

**at android.view.View.dispatchPointerEvent(View.java:10254)**

**at view.ViewRootImpl$ViewPostImeInputStage.processPointerEvent(ViewRootImpl.java:4672)**

**at android.view.ViewRootImpl$ViewPostImeInputStage.onProcess(ViewRootImpl.java:4530)**

**at android.view.ViewRootImpl$InputStage.deliver(ViewRootImpl.java:4041)**

**at android.view.ViewRootImpl$InputStage.onDeliverToNext(ViewRootImpl.java:4100)**

**at android.view.ViewRootImpl$InputStage.forward(ViewRootImpl.java:4066)**

**at android.view.ViewRootImpl$AsyncInputStage.forward(ViewRootImpl.java:4215)**

**at android.view.ViewRootImpl$InputStage.apply(ViewRootImpl.java:4074)**

**at android.view.ViewRootImpl$AsyncInputStage.apply(ViewRootImpl.java:4272)**

**at android.view.ViewRootImpl$InputStage.deliver(ViewRootImpl.java:4046)**

**at android.view.ViewRootImpl$InputStage.onDeliverToNext(ViewRootImpl.java:4100)**

**at android.view.ViewRootImpl$InputStage.forward(ViewRootImpl.java:4066)**

**at android.view.ViewRootImpl$InputStage.apply(ViewRootImpl.java:4074)**

**at android.view.ViewRootImpl$InputStage.deliver(ViewRootImpl.java:4046)**

**at android.view.ViewRootImpl.deliverInputEvent(ViewRootImpl.java:6472)**

**at android.view.ViewRootImpl.doProcessInputEvents(ViewRootImpl.java:6440)**

**at android.view.ViewRootImpl.enqueueInputEvent(ViewRootImpl.java:6394)**

**At ViewRootImpl$WindowInputEventReceiver.onInputEvent(ViewRootImpl.java:6575)**

**at android.view.InputEventReceiver.dispatchInputEvent(InputEventReceiver.java:216)**

**at android.os.MessageQueue.nativePollOnce(MessageQueue.java:-1)**

**at android.os.MessageQueue.next(MessageQueue.java:323)**

**at android.os.Looper.loop(Looper.java:145)**

**at android.app.ActivityThread.main(ActivityThread.java:6485)**

**at java.lang.reflect.Method.invoke(Method.java:-1)**

**at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:938)**

**at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:828)**

**分发：**

**当input事件到达ViewRootImpl之后，将该input调用根节点view 之DecorView的dispatchPointerEvent函数进行处理，由于该类没有实现dispatchPointerEvent方法，所以使用父类View的dispatchPointerEvent方法进行处理。**

**//View.java**

**//调用ViewGroup的dispatchTouchEvent函数进行input的分发**

public final boolean dispatchPointerEvent(MotionEvent event) {

if (event.isTouchEvent()) {

if (event.getAction() == MotionEvent.ACTION\_DOWN && mHasPostUnsetPressedState) {

if (getHandler() != null && getHandler().hasCallbacks(mUnsetPressedState)) {

mUnsetPressedState.run();

mHasRunUnsetPressedState = true;

}

}

return dispatchTouchEvent(event);

} else {

return dispatchGenericMotionEvent(event);

}

}