事件分发机制

了解android的事件分发机制主要是了解android各种事件的执行流程顺序，事件的传递和消耗。

1. View的事件分发

View自身的onTouchEvent和dispatchTouchEvent,以及view设定onTouchListener中的onTouch执行顺序:

dispatchTouchEvent -> onTouch -> onTouchEvent

dispatchTouchEvent接受来自上层容器的事件，然后调用父类的dispatchTouchEvent进行分发，如果直接返回true这不会进行分发，但事件依旧传递。如果返回false则不会分发，事件终止。

onTouchListener不为空并且view可点击，onTouch 返回true则消耗掉事件，三个条件满足则不会传递给onTouchEvent

1. ViewGroup的事件分发

ViewGroup自身有onTouchEvent，dispatchTouchEvent,onInterceptTouchEvent.

执行顺序是：

dispatchTouchEvent -> onInterceptTouchEvent -> 子view的dispatchTouchEvent（当子view能够接受，并且onInterceptTouchEvent不拦截时，不满足则执行ViewGroup自身的onTouchEvent）

dispatchTouchEvent和view的一样，调用父类的dispatchTouchEvent进行分发，如果直接返回true这不会进行分发，但事件依旧传递。如果返回false则不会分发，事件终止。

onInterceptTouchEvent进行事件拦截，设置为true则拦截，不传给子view，传给自身的onTouchEvent处理。

应用场景：

比如你需要写一个类似slidingmenu的左侧隐藏menu，主Activity上有个Button、ListView或者任何可以响应点击的View，你在当前View上死命的滑动，菜单栏也出不来；因为MOVE事件被子View处理了~ 你需要这么做：在ViewGroup的dispatchTouchEvent中判断用户是不是想显示菜单，如果是，则在onInterceptTouchEvent(ev)拦截子View的事件；自己进行处理，这样自己的onTouchEvent就可以顺利展现出菜单栏了。