Android 的事件分发机制

**一、View的事件分发总结：**

View的onTouchEvent和OnTouch区别   
还是以自定义的TestButton为例。 

我们可以通过重写onTouchEvent方法来处理诸如down move up的消息：

public class TestButton extends Button {

public TestButton(Context context) {

super(context);

// TODO Auto-generated constructor stub

}

public TestButton(Context context, AttributeSet attributeSet) {

super(context, attributeSet);

// TODO Auto-generated constructor stub

}

@Override

public boolean onTouchEvent(MotionEvent event) {

boolean value = super.onTouchEvent(event);

System.out.println("super.onTouchEvent: " + value+ " event: " + event.getAction());

return value;

}

也可以通过实现OnTouchListener的接口，然后设置TestButton的onTouchListener可以达到同样的目的 

class OnTouchListenerTest implements View.OnTouchListener{

@Override

public boolean onTouch(View v, MotionEvent event) {

return false;

}

}

TestButton b = (TestButton)findViewById(R.id.button);

OnTouchListenerTest listener = new OnTouchListenerTest();

b.setOnTouchListener(listener);

但上述两种监听有什么区别呢？

先看一下Android源码中对于View中dispatchTouchEvent的实现：

public boolean dispatchTouchEvent(MotionEvent event){

... ...

if(onFilterTouchEventForSecurity(event)){

ListenerInfo li = mListenerInfo;

if(li != null && li.mOnTouchListener != null && (mViewFlags & ENABLED\_MASK) == ENABLED

&& li.mOnTouchListener.onTouch(this, event)) {

return true;

}

if(onTouchEvent(event)){

return true;

}

}

... ...

return false;

}

可以看到onTouchListener的接口的优先级是要高于onTouchEvent的，假若onTouchListener中的onTouch方法返回true，

表示此次事件已经被消费了，那onTouchEvent是接收不到消息的。

因为Button的performClick是利用onTouchEvent实现，假若onTouchEvent没有被调用到，那么Button的Click事件也无法响应。

综合来讲：

onTouchListener的onTouch方法优先级比onTouchEvent高，会先触发。

假如onTouch方法返回false会接着触发onTouchEvent，反之onTouchEvent方法不会被调用。

内置诸如click事件的实现等等都基于onTouchEvent，假如onTouch返回true，这些事件将不会被触发。

**二、ViewGroup 的事件分发总结：**

1. Android事件分发是先传递到ViewGroup，再由ViewGroup传递到View的。

2. 在ViewGroup中可以通过onInterceptTouchEvent方法对事件传递进行拦截，onInterceptTouchEvent方法返回true代表不允许事件继续向子View传递，返回false代表不对事件进行拦截，默认返回false。

3. 子View中如果将传递的事件消费掉，ViewGroup中将无法接收到任何事件。