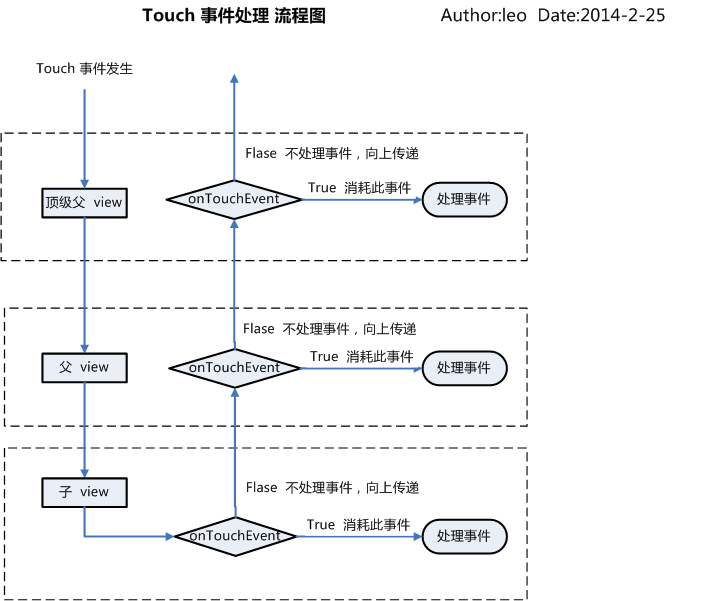
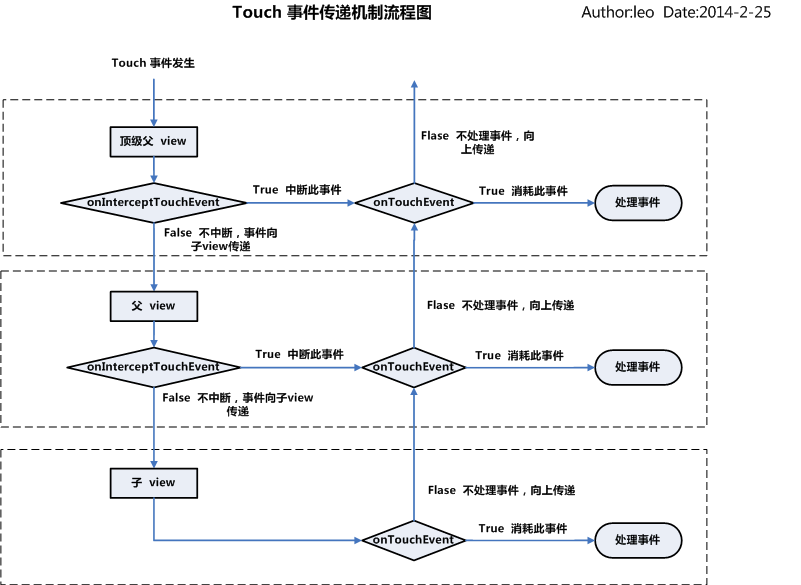
触摸事件在父view与子view之间传递的机制：





/\*\*

\* 记录第一次点击屏幕时的点的坐标

\*/

**private** **static** **int** *firstX* = 0 ;

**private** **static** **int** *firstY* = 0;

@Override

/\*\*

\* 是否中断触摸事件的传递

\* 返回true则中断父view向子view传递该事件，触摸事件由父view的onTouchEvent方法进行处理，

\* false则父view向子view传递该事件，最后由子view的onTouchEvent方法对该事件进行处理

\*/

**public** **boolean** onInterceptTouchEvent(MotionEvent ev) {

**boolean** result = **false**;

System.*out*.println("onInterceptTouchEvent :: "+ev.getAction());

**switch** (ev.getAction()) {

**case** MotionEvent.*ACTION\_DOWN*:

/\*\*在左右滑动时，滑动事件的传递被中断，由父view对该事件进行处理，

手势监听器GestureDetector没有收到滑动事件的down事件，

firstX没有记录，默认为为0，出现了图片跳动的bug

\*\*/

gd.onTouchEvent(ev);

//记录下第一个触摸屏幕的点的坐标

*firstX* = (**int**) ev.getX();

*firstY* = (**int**) ev.getY();

**break**;

**case** MotionEvent.*ACTION\_MOVE*:

//手指移动时计算移动的x轴、y轴距离

**int** disX = (**int**) Math.*abs*(ev.getX() - *firstX*);

**int** disY = (**int**) Math.*abs*(ev.getY() - *firstY*);

/\*\*如果手指在X轴移动的距离要大于Y轴移动的距离，则父view自己处理掉滑动的触摸事件，不传递给子view

\* 实现父view即viewpager的左右滑动，相反，如果Y轴滑动的距离大于X轴滑动的距离，则父view把滑动

\* 的触摸事件传递给子view，子view即ListView处理该滑动事件，实现了ListView的上下滑动

\*/

**if**(disX > disY && disX > 10){

result = **true**;

}

**break**;

**case** MotionEvent.*ACTION\_UP*:

**break**;

}

**return** result;

}

横着滑动的时候，先执行onInterceptTouchEvent方法，该方法监测到手指按下和滑动的事件，当系统发现X轴移动的距离大于Y轴移动的距离时，该方法中断滑动事件传递给子view，而是通过自己的onTouchEvent方法消费掉该滑动事件，但只在onInterceptTouchEvent方法中识别了按下的down事件，在onTouchEvent方法中未能识别按下的down事件，所以父view的onTouchEvent方法中没有记录下firstX的坐标，所以firstX的默认值为0，导致横着滑动时发生了图片跳动的bug，

因为在onInterceptTouchEvent方法中才识别了按下的down事件，所以在该方法中识别down事件的时候立刻记录下firstX的值，最终解决了水平滑动时图片跳动的bug

touch事件传递机制:

1.dispatchTouchEvent ,view执行该方法，开始分发事件

2.执行onInterceptTouchEvent方法，判断父view在向子view传递touch事件过程中是否中断传递touch事件，返回true则中断，返回false则不中断

3.执行onTouchEvent方法去消费掉该touch事件