

HenCoder Plus 第 14 课 讲义

ViewGroup 的触摸反馈、ViewPager

ViewGroup 的触摸反馈

- 除了重写 `onTouchEvent()`，还需要重写 `onInterceptTouchEvent()`
- `onInterceptTouchEvent()` 中，`ACTION_DOWN` 事件做的事和 `onTouchEvent()` 基本一致或完全一致
 - 原因：`ACTION_DOWN` 在多数手势中起到的是起始记录的作用（例如记录手指落点），而 `onInterceptTouchEvent()` 调用后，`onTouchEvent()` 未必会被调用，因此需要把这个记录责任转交给 `onInterceptTouchEvent()`。
 - 有时 `ACTION_DOWN` 事件也会在经过 `onInterceptTouchEvent()` 之后再转交给自己的 `onTouchEvent()`（例如当没有触摸到子 View 或者触摸到的子 View 没有消费事件时）。因此需要确认在 `onInterceptTouchEvent()` 和 `onTouchEvent()` 都被调用时，程序状态不会出问题。
- `onInterceptTouchEvent()` 中，`ACTION_MOVE` 一般的作用是确认滑动，即当用户朝某一方向滑动一段距离（touch slop）后，ViewGroup 要向自己的子 View 和父 View 确认自己将消费事件。
 - 确认滑动的方式：`Math.abs(event.getX() - downX) > ViewConfiguration.getXxxSlop()`
 - 告知子 View 的方式：在 `onInterceptTouchEvent()` 中返回 `true`，子 View 会收到 `ACTION_CANCEL` 事件，并且后续事件不再发给子 View
 - 告知父 View 的方式：调用 `getParent().requestDisallowInterceptTouchEvent(true)`。这个方法会递归通知每一级父 View，让他们在后续事件中不要再尝试通过 `onInterceptTouchEvent()` 拦截事件。这个通知仅在当前事件序列有效，即在这组事件结束后（即用户抬手后），父 View 会自动对后续的新事件序列启用拦截机制

VelocityTracker

- 如果 `GestureDetector` 不能满足需求，或者觉得 `GestureDetector` 过于复杂，可以自己处理 `onTouchEvent()` 的事件。但需要使用 `VelocityTracker` 来计算手指移动速度。
- 使用方法：
 - 在每个事件序列开始是（即 `ACTION_DOWN` 事件到来时），通过 `VelocityTracker.obtain()` 创建一个实例，或者使用 `velocityTracker.clear()` 把之前的某个实例重置
 - 对于每个事件（包括 `ACTION_DOWN` 事件），使用 `velocityTracker.addMovement(event)` 把事件添加进 `VelocityTracker`
 - 在需要速度的时候（例如在 `ACTION_UP` 中计算是否达到 fling 速度），使用 `velocityTracker.computeCurrentVelocity(1000, maxVelocity)` 来计算实时速度，并通过 `getXVelocity()` / `getYVelocity()` 来获取计算出的速度

- 方法参数中的 1000 是指的计算的时间长度，单位是 ms。例如这里填入 1000，那么 getXVelocity() 返回的值就是每 1000ms（即一秒）时间内手指移动的像素数
- 第二个参数是速度上限，超过这个速度时，计算出的速度会回落到这个速度。例如这里填了 200，而实时速度是 300，那么实际的返回速度将是 200
 - maxVelocity 可以通过 viewConfiguration.getScaledMaximumFlingVelocity() 来获取

scrollTo / scrollBy 和 computeScroll()

- scrollTo() / scrollBy() 会设置绘制时的偏移，通常用于滑动控件设置偏移
- scroll 值表示绘制行为在控件内部内容的起始偏移（类似：我要从内容的第 300 个像素开始绘制），因此 scrollTo() 内的参数填正值时，绘制内容会向负向移动
- scrollTo() 是瞬时方法，不会自动使用动画。如果要用动画，需要配合 View.computeScroll() 方法
 - computeScroll() 在 View 重绘时被自动调用
 - 使用方式：

```
// onTouchEvent() 中:
overScroller.startScroll(startX, startY, dx, dy);
postInvalidateOnAnimation();

.....

// onTouchEvent() 外:
@Override
public void computeScroll() {
    if (overScroller.computeScrollOffset()) { // 计算实时位置
        scrollTo(overScroller.getCurrX(),
overScroller.getCurrY()); // 更新界面
        postInvalidateOnAnimation(); // 下一帧继续
    }
}
```

问题和建议？

课上技术相关的问题，都可以在学员群里和大家讨论，我一旦有时间也都会来解答。如果我没来就 @ 我一下吧！

具体技术之外的问题和建议，都可以找丢物线（微信：diuwuxian），丢丢会为你解答技术以外的一切。



更多内容:

- 网站: <https://hencoder.com>
- 微信公众号: HenCoder

HenCoder

给高级 Android 工程师的进阶手册

微信公众号: HenCoder

微博：扔物线

知乎专栏: HenCoder

稀土掘金：扔物线

<http://hencoder.com>

