HenCoder Plus 第 15 课 讲义

ScalableImageView

放缩和移动

由于有两次移动，一次是初始偏移，另一次是随手指拖动，所以要分开两次写 `translate()`

```
canvas.translate(offsetX * scalingFraction, offsetY * scalingFraction); // 二次手动偏移
float imageScale = smallImageScale + (bigImageScale - smallImageScale) * scalingFraction;
canvas.scale(imageScale, imageScale, getWidth() / 2, getHeight() / 2); // 放缩
canvas.translate(originalOffsetX, originalOffsetY); // 初始偏移
canvas.drawBitmap(bitmap, 0, 0, paint);
```

GestrueDetector

用于在点击和长按之外，增加其他手势的监听，例如双击、滑动。通过在 `View.onTouchEvent()` 里调用 `GestureDetector.onTouchEvent()`，以代理的形式来实现：

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    return gestureDetector.onTouchEvent(event);
}
```

GeasureDetector 的默认监听器：OnGestureListener

通过构造方法 `GeasureDetector(Context, OnGestureListener)` 来配置：

```
gestureDetector = new GestureDetector(context, gestureListener);
```

OnGestureListener 的几个回调方法：

```
@Override
public boolean onDown(MotionEvent e) {
    // 每次 ACTION_DOWN 事件出现的时候都会被调用，在这里返回 true 可以保证必然消费掉事件
  return true;
}
```

```
@Override
public void onShowPress(MotionEvent e) {
    // 用户按下 100ms 不松手后会被调用，用于标记「可以显示按下状态了」
}

@Override
public boolean onSingleTapUp(MotionEvent e) {
    // 用户单击时被调用(长按后松手不会调用、双击的第二下时不会被调用)
  return false;
}

@Override
public boolean onScroll(MotionEvent downEvent, MotionEvent event, float
distanceX, float distanceY) {
    // 用户滑动时被调用
    // 第一个事件是用户按下时的 ACTION_DOWN 事件，第二个事件是当前事件
    // 偏移是按下时的位置 - 当前事件的位置
  return false;
}

@Override
public void onLongPress(MotionEvent e) {
  // 用户长按（按下 500ms 不松手）后会被调用
    // 这个 500ms 在 GestureDetectorCompat 中变成了 600ms (？？？)
}

@Override
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
float velocityY) {
    // 用于滑动时迅速抬起时被调用，用于用户希望控件进行惯性滑动的场景
  return false;
}
```

双击监听器：OnDoubleTapListener

通过 `GestureDetector.setOnDoubleTapListener(OnDoubleTapListener)` 来配置：

```
gestureDetector.setOnDoubleTapListener(doubleTapListener);
```

OnDoubleTapListener 的几个回调方法：

```
@Override
public boolean onSingleTapConfirmed(MotionEvent e) {
    // 用户单击时被调用
    // 和 onSingltTapUp() 的区别在于，用户的一次点击不会立即调用这个方法，而是在一定时
间后（300ms），确认用户没有进行双击，这个方法才会被调用
    return false;
```

```
}

@Override
public boolean onDoubleTap(MotionEvent e) {
    // 用户双击时被调用
    // 注意：第二次触摸到屏幕时就调用，而不是抬起时
    return false;
}

@Override
public boolean onDoubleTapEvent(MotionEvent e) {
    // 用户双击第二次按下时、第二次按下后移动时、第二次按下后抬起时都会被调用
    // 常用于「双击拖拽」的场景
    return false;
}
```

OverScroller

用于自动计算滑动的偏移。

```
scroller = new OverScroller(context);
```

常用于 `onFling()` 方法中，调用 `OverScroller.fling()` 方法来启动惯性滑动的计算：

```
@Override
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
float velocityY) {
    // 初始化滑动
    scroller.fling(startX, startY, velocityX, velocityY, minX, maxX, minY,
maxY);
    // 下一帧刷新
    postOnAnimation(flingRunner);
    return false;
}

...

@Override
public void run() {
    // 计算此时的位置，并且如果滑动已经结束，就停止
    if (scroller.computeScrollOffset()) {
        // 把此时的位置应用于界面
        offsetX = scroller.getCurrX();
        offsetY = scroller.getCurrY();
        invalidate();
        // 下一帧刷新
        postOnAnimation(this);
    }
```

```
}
```

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