


【Android 应用开发】多点触控（多点触控事件 | PointerId | PointerIndex | 坐标获取 | 触摸点个数）

原创 韩曙亮 于 2020-02-08 00:03:26 发布 阅读量3.1k 收藏 15 点赞数 2 版权

分类专栏: #Android 应用开发 Android UI 文章标签: 多点触控 PointerId PointerIndex

 Android 应用开发 同时被 2 个专栏收录 ▾

10 订阅 64 篇文章 订阅专栏

文章目录

- I . 多点触控事件
- II . PointerId 与 PointerIndex 简介
- III . 本次 PointerId 与 PointerIndex 获取
- IV . PointerId 与 PointerIndex 互相转化
- V . 触摸点个数获取 (getPointerCount)
- VI . 触摸点坐标获取
- VII . 多点触控 示例代码 (参考)

I . 多点触控事件

1 . 多点触控事件：

- ① ACTION_DOWN : 第一个手指按下
- ② ACTION_MOVE : 所有的手指移动
- ③ ACTION_UP : 最后一个手指抬起
- ④ ACTION_POINTER_DOWN : 中间的手指按下 (已经有手指按下)
- ⑤ ACTION_POINTER_UP : 中间手指抬起 (还有手指在触摸中)

2 . 获取多点触控事件：调用 MotionEvent 对象的 getActionMasked() 可以获取多点触控事件，即上面的 5 种触摸事件；

3 . 获取并处理多点触控事件代码示例：

① 获取多点触控事件：

```
1 //获取当前的多点触控触摸事件
2 int actionMasked = event.getActionMasked();
```

② 处理多点触控事件：

```
1 //处理 5种多点触控事件
2 switch (actionMasked){
3
4     case MotionEvent.ACTION_DOWN : //第一个手指按下
```

```

5         break;
6
7         case MotionEvent.ACTION_MOVE :           //手指移动
8             break;
9
10        case MotionEvent.ACTION_UP :             //最后一个手指抬起
11            break;
12
13        case MotionEvent.ACTION_POINTER_DOWN :    //中间的手指按下 ( 已经有手指按下 )
14            break;
15
16        case MotionEvent.ACTION_POINTER_UP :      //中间手指抬起 ( 还有手指在触摸中 )
17            break;
18
19    }

```

II . PointerId 与 PointerIndex 简介

1 . Pointer 触摸点 :

① **PointerId** : 触摸点标识, 一旦赋值不可更改;

手指按下, 该触摸点产生 自带触摸点 ID 属性, 以及分配一个触摸点索引, 当手指抬起, 该触摸点 ID 值回收; 期间触摸点 ID 不会改变;

② **PointerIndex** : 触摸点索引, 触摸点的索引值可能会变;

该值从 0 开始计数, 取值范围是 0 ~ 触摸点个数 - 1, 每当有一个中间触摸点抬起, 后面的触摸点就会依次替补该空缺;

2 . MotionEvent 与 Pointer 触摸点关系 :

MotionEvent 对象中存储多个触摸点信息, 每个触摸点都有 **PointerId** (触摸点标识) 和 **PointerIndex** (触摸点索引) 两个属性;

3 . PointerId 与 PointerIndex 运行机制 :

PointerId 触摸点 ID 标识, 只要按下, 不抬起, 值不变, 如果中间手指抬起, 那么出现中间的 ID 空置的情况; 3 个手指按下, **PointerId** 分别是 0, 1, 2; 中间的抬起, 那么 **PointerId** 的 ID 值会空着, 两个手指的 ID 分别是 0 和 2;

PointerIndex 触摸点索引值, 3 个手指按下, 索引值依次是 0, 1, 2, 如果第二个手指抬起, 后续的第三个的索引会由 2 变成 1, 顺序补位替补上去;

III . 本次 PointerId 与 PointerIndex 获取

① 获取本次的触摸事件的 触摸点 ID :

```

1 //获取本事件对应的 pointerIndex
2 int actionIndex = event.getActionIndex();

```

② 获取本次触摸事件的 触摸点 索引 :

```
1 //4 . 获取 actionIndex 对应的 PointerId
2 int pointerId = event.getPointerId(actionIndex);
```

IV . PointerId 与 PointerIndex 互相转化

PointerId 与 PointerIndex 关联 及 转化关系 :

- ① 两者特点 : PointerId 是固定不变的 , PointerIndex 有顺序补位的特征 , 是不确定的 ;
- ② PointerId -> PointerIndex : 通过调用 MotionEvent 对象的 findPointerIndex(int pointerId) 方法可以获取指定 pointerId 对应的 PointerIndex 值 ;
- ③ PointerIndex -> PointerId : 通过调用 MotionEvent 对象的 getPointerId(int pointerIndex) 方法可以获取指定 PointerIndex 对应的 pointerId 值 ;

```
1 //3 . 获取 PointerId 对应的 PointerIndex
2 // 查找 ID值 对应的索引值
3 int pointerIndex = event.findPointerIndex(PointerId);
4 ...
5 // 获取 索引值 对应的 ID 值
6 int PointerId = event.getPointerId(PointerIndex)
```

V . 触摸点个数获取 (getPointerCount)

获取当前触摸点个数 :

```
1 //获取当前触摸点个数
2 int pointCount = event.getPointerCount();
```

VI . 触摸点坐标获取

获取触摸点坐标 :

- ① 获取当前触摸事件坐标 :

```
public final float getX()
public final float getY()
```

- ② 获取指定触摸点索引 PointerIndex 坐标 : (注意不是触摸点标识)

```
public final float getX(int pointerIndex)
public final float getY(int pointerIndex)
```

```

1 //1 . 获取 坐标
2
3 //1.1 获取 当前 触摸 事件的 x y 坐标
4 float x = event.getX();
5 float y = event.getY();
6
7 //1.2 获取 指定 pointerIndex 的 x y 坐标 ( 获取第 0 个索引的坐标 )
8 int pointerIndex = 0;
9 float x_pointer = event.getX(pointerIndex);
10 float y_pointer = event.getY(pointerIndex);

```

VII . 多点触控 示例代码 (参考)

```

package kim.hsl.multitouch;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.util.Log;
import android.view.MotionEvent;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {

        //多点触控示例

        /*
        1 . 多点触控事件 :
            多点触控事件 Action 由 pointerIndex 和 动作码 进行或运算 合成
            调用 event.getActionMasked() 即可获取动作码 , 屏蔽 pointerIndex 触摸索引

        2 . PointerId 和 PointerIndex :
            Pointer 就是触摸点 ;

            两个值的特点 :
                PointerId : 唯一性的表示一个触摸点 , 该值如果触摸上 , 任意移动 , 只要不抬起 , 就不变 ;
                PointerIndex : 触摸点的索引 , 是一个从 0 开始计数的连续的数值 , 同样是第 0 个索引 , 代表的可能是不同

            运行机制 :
                PointerId : 只要按下 , 不抬起 , 值不变 , 如果中间手指抬起 , 那么出现中空的情况 ;
                3 个手指按下 , PointerId 分别是 1 , 2 , 3 ;
                中间的抬起 , 那么 PointerId 的 ID 值会空着 , 两个手指的 ID 分别是 1 和 3

                PointerIndex : 3 个手指按下 , 如果第二个手指抬起 , 后续的第三个的索引会由 2 变成 1 , 顺序补位替补上 ;

            PointerId 与 PointerIndex 转换换 :
                pointerId 是确定的 , PointerIndex 有顺序补位的特征 , 是不确定的
                通过 findPointerIndex(int pointerId) 方法可以获取指定 pointerId 的 PointerIndex 值

        3 . 获取坐标 :
            获取当前坐标 : event.getX();
            获取指定 pointerIndex 坐标 : event.getX(pointerIndex);
        */
    }
}

```

4 . 多点触控的 5 种事件 :

单点触控需要处理 ACTION_DOWN , ACTION_MOVE , ACTION_UP

多点触控需要处理 ACTION_DOWN , ACTION_MOVE , ACTION_UP , ACTION_POINTER_DOWN , ACTION_POINTER_UP

多点触控中这些动作有不同的含义 :

ACTION_DOWN : 第一个手指按下

ACTION_MOVE : 所有的手指滑动事件

ACTION_UP : 最后一个手指抬起

ACTION_POINTER_DOWN : 中间的手指按下 (已经有手指按下)

ACTION_POINTER_UP : 中间手指抬起 (还有手指在触摸中)

5 . 多点触控与单点触控对比

多了两个事件 ACTION_POINTER_DOWN 和 ACTION_POINTER_UP

多了触摸点 ID 标识与触摸点索引机制

*/

//1 . 获取 坐标

//1.1 获取 当前 触摸 事件的 x y 坐标

float x = event.getX();

float y = event.getY();

//1.2 获取 指定 pointerIndex 的 x y 坐标 (获取第 0 个索引的坐标)

int pointerIndex = 0;

float x_pointer = event.getX(pointerIndex);

float y_pointer = event.getY(pointerIndex);

//2 . 获取 触摸事件 相关属性

//获取当前的多点触控触摸事件

int actionMasked = event.getActionMasked();

//获取本事件对应的 pointerIndex

int actionIndex = event.getActionIndex();

//获取当前触摸点个数

int pointCount = event.getPointerCount();

//3 . 获取 PointerId 对应的 PointerIndex

// 查找第 1 个按下的手指的索引值

pointerIndex = event.findPointerIndex(0);

//4 . 处理对应的事件操作

switch (actionMasked){

case MotionEvent.ACTION_DOWN : //第一个手指按下

Log.i("TAG", "ACTION_DOWN : 第一个手指按下");

break;

case MotionEvent.ACTION_MOVE : //手指移动

Log.i("TAG", "ACTION_MOVE : 手指移动");

break;

case MotionEvent.ACTION_UP : //最后一个手指抬起

Log.i("TAG", "ACTION_UP : 最后一个手指抬起");

break;

case MotionEvent.ACTION_POINTER_DOWN : //中间的手指按下 (已经有手指按下)

Log.i("TAG", "ACTION_POINTER_DOWN : 中间的手指按下 (已经有手指按下)");

break;

```
        case MotionEvent.ACTION_POINTER_UP :    //中间手指抬起 ( 还有手指在触摸中 )
            Log.i("TAG", "ACTION_POINTER_UP : 中间手指抬起 ( 还有手指在触摸中 )");
            break;

    }

    return true;
}
```