> 实现一个跟随手指移动的view其实是特别容易实现的，不过有的时候还是挺有用的，最近做的视频互动软件就有这样的需求，大概几十行代码就可以搞定，然后记录一下吧。

实现的主要思想，就是利用onTouchListener，然后判断出手指按下的点，同时监听移动的事件，然后稍微计算一下就可以求出来view最终应该呈现的位置了，然后通过改变LayoutParams的值就可以是实现view的跟随手指拖拽的效果了，当然还可以优化，例如通过计算如果移到屏幕边缘就停下来之类的，或者哪里是不能移到地方。

```
public class TestActivity extends AppCompatActivity implements View.OnTouchListener {

    private ImageView imageView;
    private RelativeLayout relativeLayout;

    private int lastX, lastY;    //保存手指点下的点的坐标
    final static int IMAGE_SIZE = 150;

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

        imageView = (ImageView) findViewById(R.id.image);
        relativeLayout = (RelativeLayout) findViewById(R.id.layout);
        //初始设置一个layoutParams
        RelativeLayout.LayoutParams layoutParams = new RelativeLayout.LayoutParams(IMAGE_SIZE,IMAGE_S
        imageView.setLayoutParams(layoutParams);
        //设置屏幕触摸事件
        imageView.setOnTouchListener(this);
    }


    public boolean onTouch(View view, MotionEvent event) {
        switch (event.getAction() & MotionEvent.ACTION_MASK) {
            case MotionEvent.ACTION_DOWN:
                //将点下的点的坐标保存
                lastX = (int) event.getRawX();
                lastY = (int) event.getRawY();
                break;

            case MotionEvent.ACTION_MOVE:
                //计算出需要移动的距离
                int dx = (int) event.getRawX() - lastX;
                int dy = (int) event.getRawY() - lastY;
                //将移动距离加上，现在本身距离边框的位置
                int left = view.getLeft() + dx;
                int top = view.getTop() + dy;
                //获取到layoutParams然后改变属性，在设置回去
                RelativeLayout.LayoutParams layoutParams = (RelativeLayout.LayoutParams) view
                        .getLayoutParams();
                layoutParams.height = IMAGE_SIZE;
                layoutParams.width = IMAGE_SIZE;
                layoutParams.leftMargin = left;
                layoutParams.topMargin = top;
                view.setLayoutParams(layoutParams);
                //记录最后一次移动的位置
                lastX = (int) event.getRawX();
                lastY = (int) event.getRawY();
                break;
        }
        //刷新界面
        relativeLayout.invalidate();
        return true;
    }
}
```

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:tools="http://schemas.android.com/tools"
                android:id="@+id/layout"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                tools:context=".MainActivity">

    <ImageView
        android:id="@+id/image"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:background="@mipmap/ic_launcher"
        />


</RelativeLayout>
```

以上便是这个简单的view啦，思路还是很清晰的，当然能够改造的地方有很多，例如加一个惯性的效果啊，或者弄一个加速度的效果啊，都是可以的