## 模块化了之后关闭一段activity，用clearTop

## 视频的activity换成为单任务栈，然后moveBackStack

## 安卓序列化的问题，构造方法，简略缩打

## Messenge不能传递parcelable对象！！！只能传递系统的！

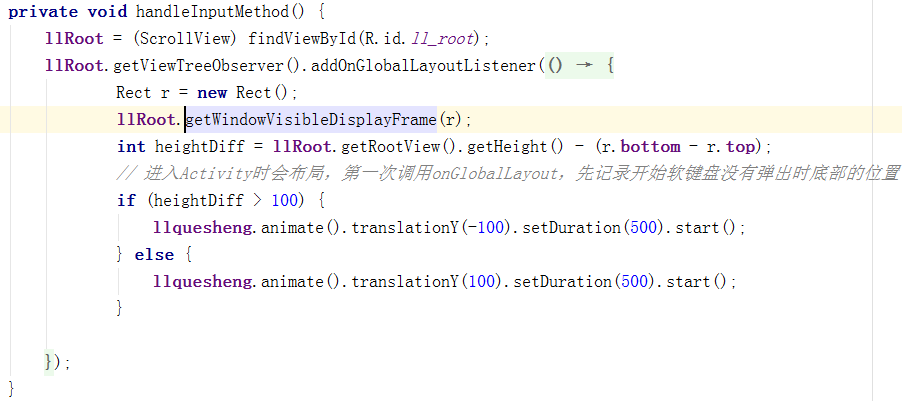
## Ipc中传递的对象不会是同一个，所以通过判断是否是同一个对象进行操作是不可取的。用RemoteCallbackList

## 判断textview是否超过两行，是的话隐藏，用了viewTreeObserveable，onGlobalLayout

## JsonIgnore在设了set、get方法的上面无效。需要在方法上面也设置。

在类上面@JsonIgnoreProperties(value = {"hibernateLazyInitializer", "handler"})（字段名）

## 判断是否调用了手机出入法



## 手写签名遮罩层事件传递

## 依赖pdfview后，运行闪退



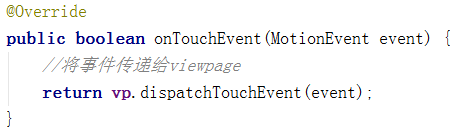
不规则图形，用**clipChildren**

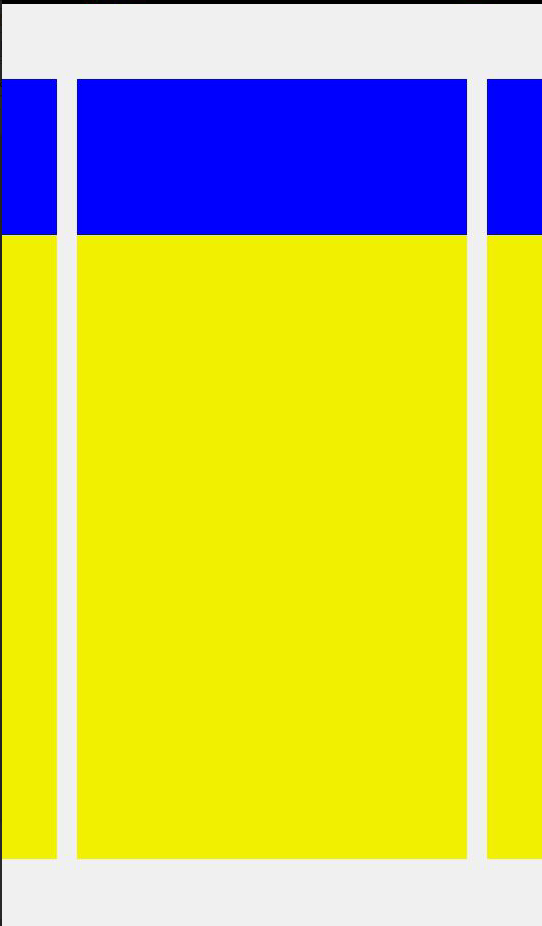


还有两边可见的viewpage，父布局用这个，viewpage两边设置margin。

还要设置page之间的margin，因为这样按住两边的activity滑动动不了，所以还要把事件传递给viewpage







## Activity反复横跳

如果已经启动了四个Activity：A，B，C和D，在D Activity里，想再启动一个Actvity B，但不变成A,B,C,D,B，而是希望是A,C,D,B，则可以像下面写代码（标准模式）

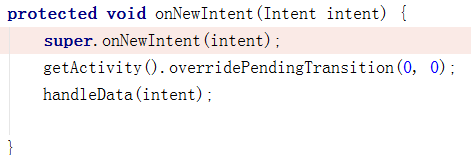
Intent intent = new Intent(this, MainActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_REORDER\_TO\_FRONT);

startActivity(intent);

但是这样跳转可能会有页面闪烁，需要关闭动画，在startActivity后面用

在newIntent的时候也调用一下



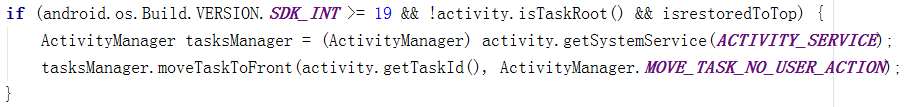
### 后面追加问题

有的手机返回会跳到桌面bug，修复如下：

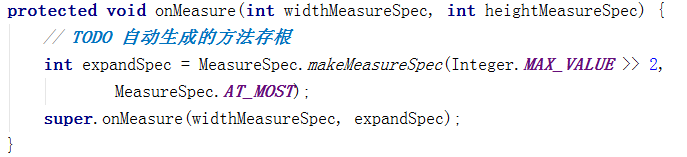
再destroy，和finish中，把这个带到前面。现在newIntent中记录是否是record



然后在destroy和finish中判断，带到前台



## GrideView在ListView和在ScrollView中不会自适应高度

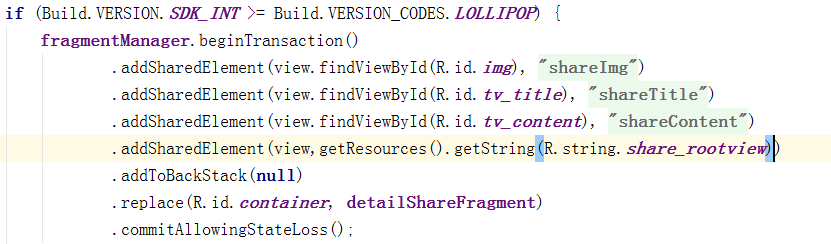


继承重写方法

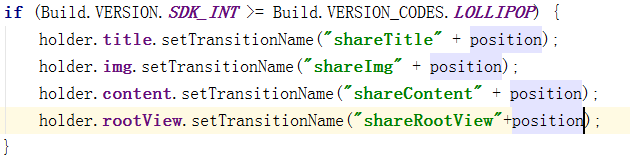
## Fragment跳转共享元素，listview，recyclerview

中的item要设置不同的transitionName，这个只是用来区分不同的元素，跟在最后操作中设置的没有什么关系，最后设置的是需要启动的界面里面共享元素的transitionName。

跳转的地方



Adapter里面



## 第三方混淆，在gradle里面配置

# Fragment 中的commit，commitNow，commitAllowStateloss，commitNowAllowStateloss

Commit并没有立即执行，commit是立即执行

因为有时候activity，pause，或stop了，已经saveInstance了，那么commit就会报错，用allowstateloss

# 选择图片、预览图片过多闪退android.os.TransactionTooLargeException: data parcel size 591440 bytes

通过intent传参数到下一个activity，由于图片地址过多，超过限制，闪退

# 多层viewpager不显示问题

Adapter的supportManager用getChildManager；

# CoordinatorLayout与listview冲突解决

让ListView 实现 NestedScrollingChild

public class MyListView extends ListView implements NestedScrollingChild{

private final NestedScrollingChildHelper mScrollingChildHelper;

public MyListView(Context context, AttributeSet attrs) {

super(context, attrs);

mScrollingChildHelper = new NestedScrollingChildHelper(this);

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.LOLLIPOP) {

setNestedScrollingEnabled(true);

}

}

@Override

public void setNestedScrollingEnabled(boolean enabled) {

mScrollingChildHelper.setNestedScrollingEnabled(enabled);

}

@Override

public boolean isNestedScrollingEnabled() {

return mScrollingChildHelper.isNestedScrollingEnabled();

}

@Override

public boolean startNestedScroll(int axes) {

return mScrollingChildHelper.startNestedScroll(axes);

}

@Override

public void stopNestedScroll() {

mScrollingChildHelper.stopNestedScroll();

}

@Override

public boolean hasNestedScrollingParent() {

return mScrollingChildHelper.hasNestedScrollingParent();

}

@Override

public boolean dispatchNestedScroll(int dxConsumed, int dyConsumed, int dxUnconsumed,

int dyUnconsumed, int[] offsetInWindow) {

return mScrollingChildHelper.dispatchNestedScroll(dxConsumed, dyConsumed,

dxUnconsumed, dyUnconsumed, offsetInWindow);

}

@Override

public boolean dispatchNestedPreScroll(int dx, int dy, int[] consumed, int[] offsetInWindow) {

return mScrollingChildHelper.dispatchNestedPreScroll(dx, dy, consumed, offsetInWindow);

}

@Override

public boolean dispatchNestedFling(float velocityX, float velocityY, boolean consumed) {

return mScrollingChildHelper.dispatchNestedFling(velocityX, velocityY, consumed);

}

@Override

public boolean dispatchNestedPreFling(float velocityX, float velocityY) {

return mScrollingChildHelper.dispatchNestedPreFling(velocityX, velocityY);

}

最后在xml布局文件中,在MyListView外层嵌套一层NestedScrollView即可

<android.support.v4.widget.NestedScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:fillViewport="true"

app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior">

<com.dipaitv.widget.MyListView

android:id="@+id/listview"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

</com.dipaitv.widget.MyListView>

</android.support.v4.widget.NestedScrollView>