移动互联网技术

Android 程序开发大作业

技术报告

班　　级：2017211311

姓　　名：王大卫

学　　号：2017211510

小组成员：罗浩然、王大卫

一、内容及要求

➢ 整体要求

1. 任务：开发一个短视频 App
2. API：https://beiyou.bytedance.com/api/invoke/video/invoke/video
3. 使用 RecyclerView 显示视频列表 (一页显示多个 item)
4. 使用 Glide 加载封面图

➢ 必要功能：

1. 从视频信息流点击某个视频封面进入播放页面
2. 根据视频信息的 url 播放视频
3. 单击视频窗口暂停/继续

二、小组分工

➢ 罗浩然：

设计 Recyclerview 监听和视频播放代码

➢ 王大卫：

设计 xml 布局及视频、封面图片接口

三、主要思路

1. 在和 RecyclerView 的同级布局文件中，在右下角放置一个隐藏 FrameLayout，当正在播放的列表滑出界面的时候，将右下角的 FragmeLayout 设置为显示，并将播放的 SurfaceView 添加到右下角的 FragmeLayout 播放。

2. 在列表的 ViewHolder 布局文件中放置一个 FragmeLayout，当点击播放按钮时，将 SurfaceView 添加到 FragmentLayout 中播放。

3. 为 RecyclerView 设置 addOnChildAttachStateChangeListener 事件，这个监听有两个重要的方法。

```
public void onChildViewAttachedToWindow(View view);
public void onChildViewDetachedFromWindow(View view);
```

四、布局：

1. 首先看下主界面布局代码。

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <android.support.design.widget.CoordinatorLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <android.support.design.widget.AppBarLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <android.support.v7.widget.Toolbar
                android:id="@+id/toolbar"
                android:layout_width="match_parent"
                android:layout_height="?attr/actionBarSize"
                android:background="@color/colorPrimary"
                app:layout_scrollFlags="scroll|enterAlways|snap" />
        </android.support.design.widget.AppBarLayout>

        <android.support.v7.widget.RecyclerView
            android:id="@+id/recyclerView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:layout_behavior="@string/appbar_scrolling_view_behavior" />
    </android.support.design.widget.CoordinatorLayout>
    <!-- 右下角的 FragmeLayout -->
```

```
    <FrameLayout
        android:id="@+id/video_root_fl"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:background="#000"
        android:visibility="gone" />
    <!--全屏播放的 FragmeLayout-->
    <FrameLayout
        android:id="@+id/video_full_screen"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:visibility="gone" />
</RelativeLayout>
```

可以在布局文件中看到，在右下角中放置了一个隐藏的 FrameLayout，当正在播放的列表滑出界面时我们会使用这个 FrameLayout 来放置播放视频的 SurfaceView。

2. 在 RecyclerView 中的 item 布局文件中也有一个 FrameLayout。

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/item_cardview"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:layout_margin="10dp"
    android:background="#fff"
    android:elevation="8dp"
    android:padding="5dp"
    app:cardBackgroundColor="#fff"
    app:cardCornerRadius="5dp">

    <!--列表播放使用的 FrameLayout-->
    <FrameLayout
        android:id="@+id/item_video_root_fl"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:visibility="gone" />

    <ImageView
```

```
        android:id="@+id/item_imageview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop" />

    <ImageView
        android:id="@+id/item_image_play"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:src="@drawable/ic_play_circle_outline_white_48dp" />
</android.support.v7.widget.CardView>
```

item 中这个 FrameLayout 用于点击列表中播放按钮将要播放的 SurfaceView 添加到这个 FrameLayout 中。

五、设置监听：

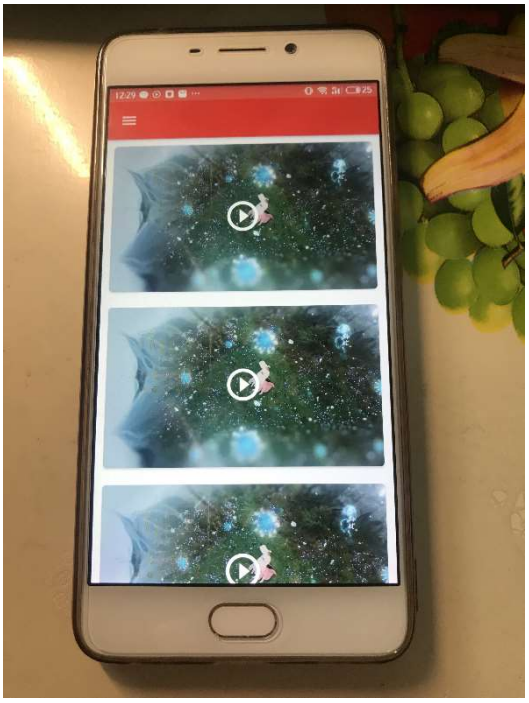
为 RecyclerView 添加 addOnChildAttachStateChangeListener 监听，当正在播放的 item 滑出界面时会回调 onChildViewDetachedFromWindow 这个方法，我们在这个方法中判断如果 FragmeLayout 中有视频播放，将右下角的 FrameLayout 设置为显示，移除 item 布局中的 SurfaceView 并将其添加到右下角的 FrameLayout,将记录这个 item 的位置，当再次将这个 item 滑动到界面中时，会回调 onChildViewAttachedToWindow 这个方法。同理再将这个右下角中的 FrameLayout 中的 SurfaceView 移除并设置为隐藏，再将 SurfaceView 添加到 item 中的 FrameLayout 播放。

```
//为 RecyclerView 添加 addOnChildAttachStateChangeListener 监听
recyclerView.addOnChildAttachStateChangeListener(new
RecyclerView.OnChildAttachStateChangeListener() {
    @Override
    public void onChildViewAttachedToWindow(View view) {
        if (videoPosition == -1 || videoRootViewFl.getVisibility() != View.VISIBLE) {
            return;
        }
        if (videoPosition == recyclerView.getChildAdapterPosition(view)) {
            videoPosition = -1;
            showVideo(view, VIDEO_PATH);
        }
    }
```

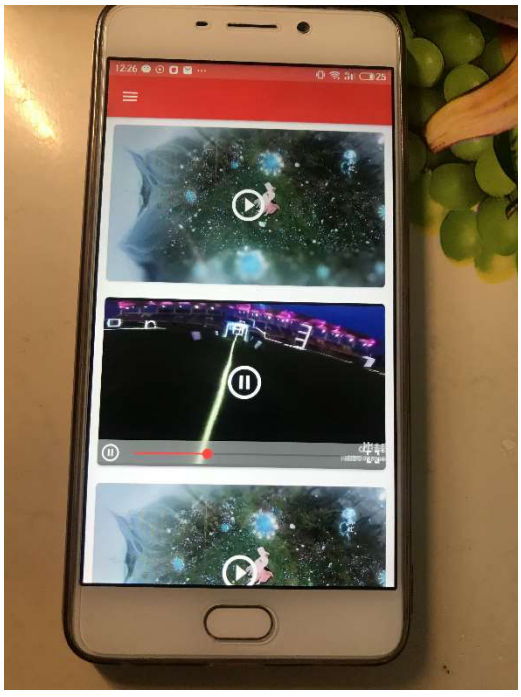
```
 @Override
  public void onChildViewDetachedFromWindow(View view) {
  if (videoView == null || videoRootViewFl.getVisibility() == View.VISIBLE) {
        return;
  }
 View v = view.findViewById(R.id.item_video_root_fl);
 if (v != null) {
       FrameLayout fl = (FrameLayout) v;
        videoPosition = recyclerView.getChildAdapterPosition(view);
        if (fl.getChildCount() > 0) {
                fl.removeAllViews();
                int position = 0;
                if (videoView.isPlaying()) {
                      position = videoView.getPosition();
                      videoView.stop();
                }
                videoRootViewFl.setVisibility(View.VISIBLE);
                videoRootViewFl.removeAllViews();
                lastView = videoRootViewFl;
                videoRootViewFl.addView(videoView, new ViewGroup.LayoutParams(-1, -1));
                videoView.setVideoPath(VIDEO_PATH);
                videoView.start();
                videoView.seekTo(position);
        }
        fl.setVisibility(View.GONE);
 }
 v = view.findViewById(R.id.item_imageview);
 if (v != null) {
        if (v.getVisibility() != View.VISIBLE) {
              v.setVisibility(View.VISIBLE);
        }
 }
 v = view.findViewById(R.id.item_image_play);
 if (v != null) {
        if (v.getVisibility() != View.VISIBLE) {
              v.setVisibility(View.VISIBLE);
        }
   }
   }
});
```

六、实现效果：

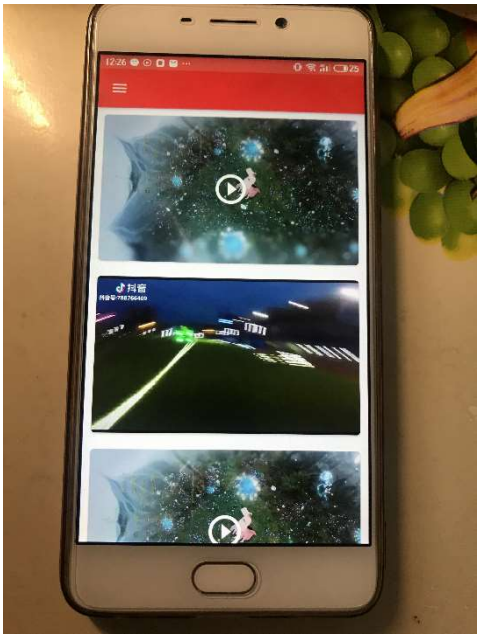
1. 主体界面设计为 RecyclerView 列表模式，每个列表有一个含有 Glide 显示的封面图片，视频和图片均来自要求的 API 网址。



2. 点击三角形播放按钮，播放其中一个视频，封面图片隐去，开始播放视频，并出现进度条，可以随时调整观看视频进度。



3. 若对视频没有操作，视频会保持播放状态，并隐藏播放进度条和暂停按钮，以达到最好的观看效果，点击视频任意位置可以唤出进度条和暂停按钮，随时对视频进行操作。



4. 下拉列表，会看到很多网络视频，当正在播放的视频划出屏幕，会在右下角出现小窗口（画中画）观看视频，当点击播放另一个视频时，正在播放的视频会停止播放。