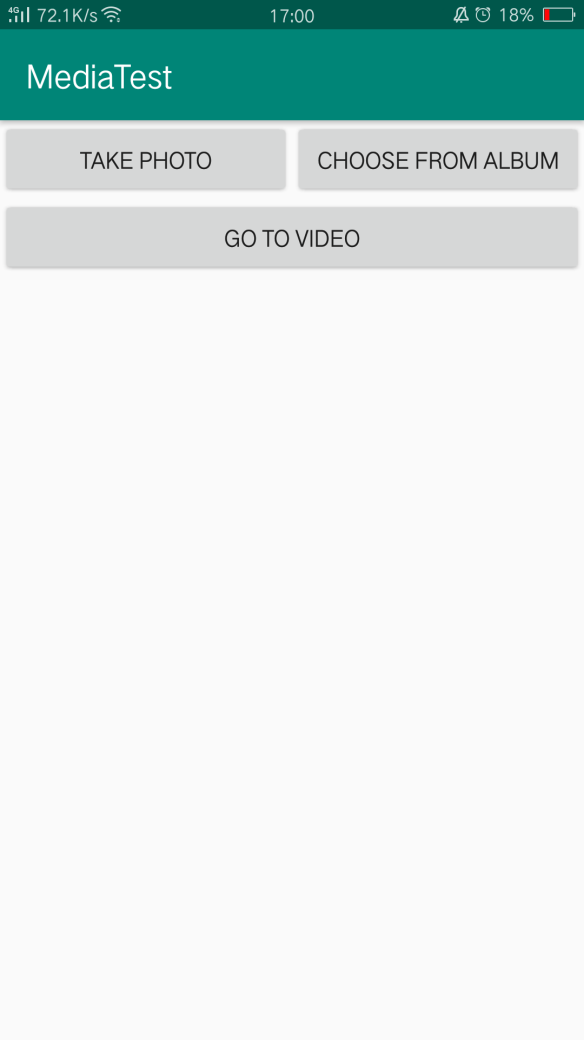
**第11次安卓作业**

201730145033 宗玥

1. 作业要求
2. 在你手机上安装一个你开发的App项目，并测试运行效果
3. 参考8.3节在App上显示来源于摄像头拍摄或相册中的照片
4. 参考8.4节在app中播放一段本人拍摄的小视频（拓展）增加选取文件播放功能，调用系统自带的文件浏览器选取视频文件路径
5. 作业展示
6. 在你手机上安装一个你开发的App项目，并测试运行效果



AndroidManifest.xml

声明权限、provider

由于软件版本问题，provider应写为

android:name="androidx.core.content.FileProvider"

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.mediatest">  
  
 <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
 <activity android:name=".VideoPlay"></activity>  
 <activity android:name=".MainActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
  
 <provider  
 android:name="androidx.core.content.FileProvider"  
 android:authorities="com.example.mediaalbumtest.fileprovider"  
 android:exported="false"  
 android:grantUriPermissions="true">  
 <meta-data  
 android:name="android.support.FILE\_PROVIDER\_PATHS"  
 android:resource="@xml/file\_paths" />  
 </provider>  
 </application>  
  
</manifest>

file\_paths.xml,将所有可能的情况都写上

<?xml version="1.0" encoding="utf-8"?>  
<paths>  
 <external-path  
 name="external"  
 path="." />  
 <external-files-path  
 name="external\_files"  
 path="." />  
 <cache-path  
 name="cache"  
 path="." />  
 <external-cache-path  
 name="external\_cache"  
 path="." />  
 <files-path  
 name="files"  
 path="." />  
</paths>

1. 显示来源于摄像头拍摄照片

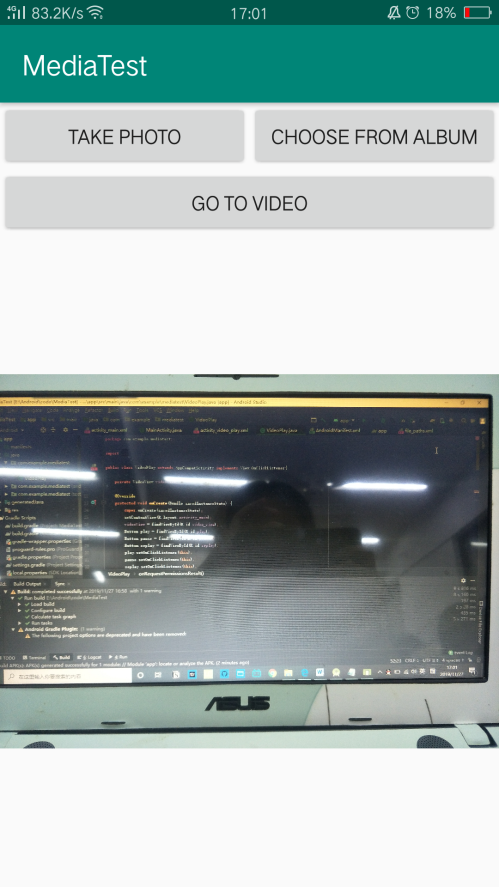
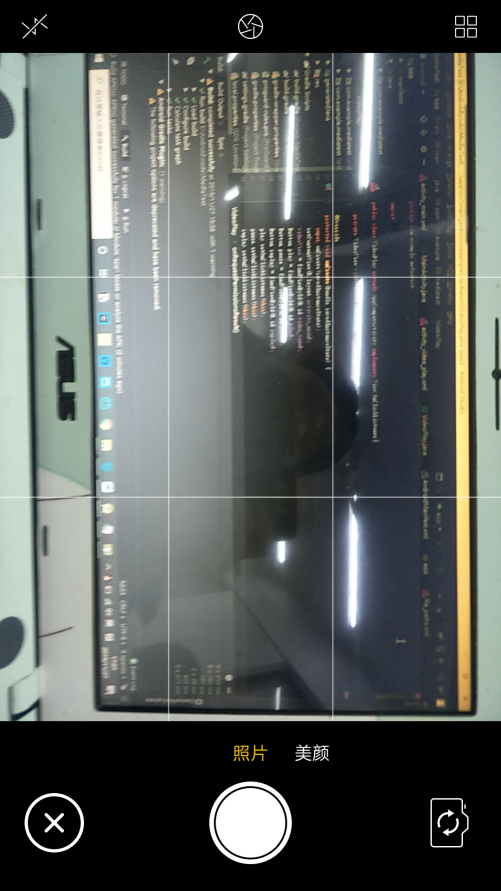
takePhoto.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View v) {  
 File outputImage=new File(getExternalCacheDir(),"output\_image.jpg");  
 try{  
 if (outputImage.exists()){  
 outputImage.delete();  
 }  
 outputImage.createNewFile();  
 }catch (IOException e) {  
 e.printStackTrace();  
 }  
 if (Build.VERSION.*SDK\_INT*>=24){  
 imageUri= FileProvider.*getUriForFile*(MainActivity.this,"com.example.albumtest.fileprovider",outputImage);  
 }else {  
  
 imageUri=Uri.*fromFile*(outputImage);  
 }  
 Intent intent=new Intent("android.media.action.IMAGE\_CAPTURE");  
 intent.putExtra(MediaStore.*EXTRA\_OUTPUT*,imageUri);  
 startActivityForResult(intent,*TAKE\_PHOTO*);  
 }  
});

调用摄像头拍照的逻辑实现

case *TAKE\_PHOTO*:  
 if (resultCode == *RESULT\_OK*) {  
 try {  
 Bitmap bitmap = BitmapFactory.*decodeStream*(getContentResolver().openInputStream(imageUri));  
 picture.setImageBitmap(bitmap);  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 }  
 break;

显示图片

private void displayImage(String imagePath) {  
 if (imagePath != null) {  
 Bitmap bitmap = BitmapFactory.*decodeFile*(imagePath);  
 picture.setImageBitmap(bitmap);  
 } else {  
 Toast.*makeText*(this, "failed to get image", Toast.*LENGTH\_SHORT*).show();  
 }  
}



1. 显示相册中的照片

给button创建监听器同时动态申请权限

chooseFromAblum.setOnClickListener(new View.OnClickListener(){  
  
 @Override  
 public void onClick(View v) {  
 if (ContextCompat.*checkSelfPermission*(MainActivity.this, Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*)!= PackageManager.*PERMISSION\_GRANTED*){  
 ActivityCompat.*requestPermissions*(MainActivity.this,new String[]{Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*},1);  
 }else{  
 openAlbum();  
 }  
 }  
});

判断是否有权限

public void onRequestPermissionsResult(int requestCode,String[] permissions,int[] grantResults){  
 switch(requestCode){  
 case 1:  
 if (grantResults.length > 0 && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {  
 openAlbum();  
 } else {  
 Toast.*makeText*(this, "You denied the permission", Toast.*LENGTH\_SHORT*).show();  
 }  
 break;  
 default:  
 }  
}

打开相册setType设置类型为图片，扩展名任意

private void openAlbum() {  
 Intent intent=new Intent("android.intent.action.GET\_CONTENT");  
 intent.setType("image/\*");  
 startActivityForResult(intent,*CHOOSE\_PHOTO*);  
}

选择照片逻辑实现

case *CHOOSE\_PHOTO*:  
 if (resultCode == *RESULT\_OK*) {  
 // 判断手机系统版本号  
 if (Build.VERSION.*SDK\_INT* >= 19) {  
 // 4.4及以上系统使用这个方法处理图片  
 handleImageOnKitKat(data);  
 } else {  
 // 4.4以下系统使用这个方法处理图片  
 handleImageBeforeKitKat(data);  
 }  
 }  
 break;

安卓版本号为4.4及以上用此方法处理图片

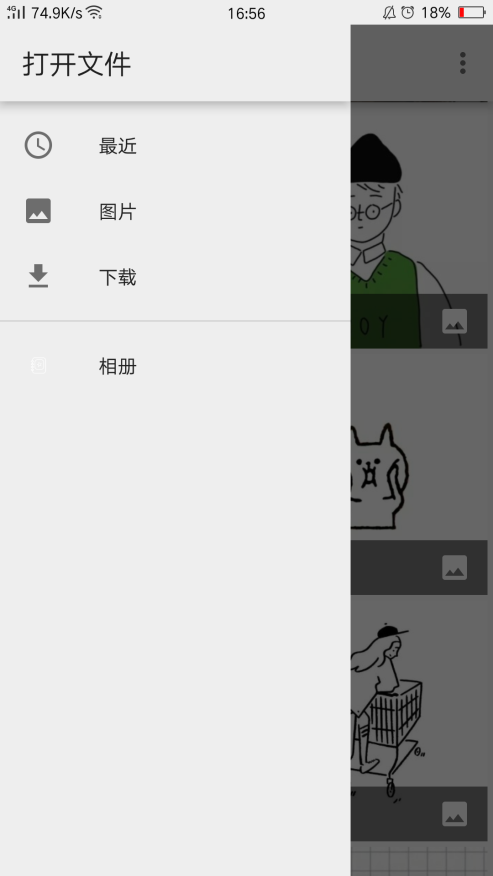
private void handleImageOnKitKat(Intent data) {  
 String imagePath = null;  
 Uri uri = data.getData();  
 Log.*d*("TAG", "handleImageOnKitKat: uri is " + uri);  
 if (DocumentsContract.*isDocumentUri*(this, uri)) {  
 // 如果是document类型的Uri，则通过document id处理  
 String docId = DocumentsContract.*getDocumentId*(uri);  
 if("com.android.providers.media.documents".equals(uri.getAuthority())) {  
 String id = docId.split(":")[1]; // 解析出数字格式的id  
 String selection = MediaStore.Images.Media.*\_ID* + "=" + id;  
 imagePath = getImagePath(MediaStore.Images.Media.*EXTERNAL\_CONTENT\_URI*, selection);  
 } else if ("com.android.providers.downloads.documents".equals(uri.getAuthority())) {  
 Uri contentUri = ContentUris.*withAppendedId*(Uri.*parse*("content://downloads/public\_downloads"), Long.*valueOf*(docId));  
 imagePath = getImagePath(contentUri, null);  
 }  
 } else if ("content".equalsIgnoreCase(uri.getScheme())) {  
 // 如果是content类型的Uri，则使用普通方式处理  
 imagePath = getImagePath(uri, null);  
 } else if ("file".equalsIgnoreCase(uri.getScheme())) {  
 // 如果是file类型的Uri，直接获取图片路径即可  
 imagePath = uri.getPath();  
 }  
 displayImage(imagePath); // 根据图片路径显示图片  
}

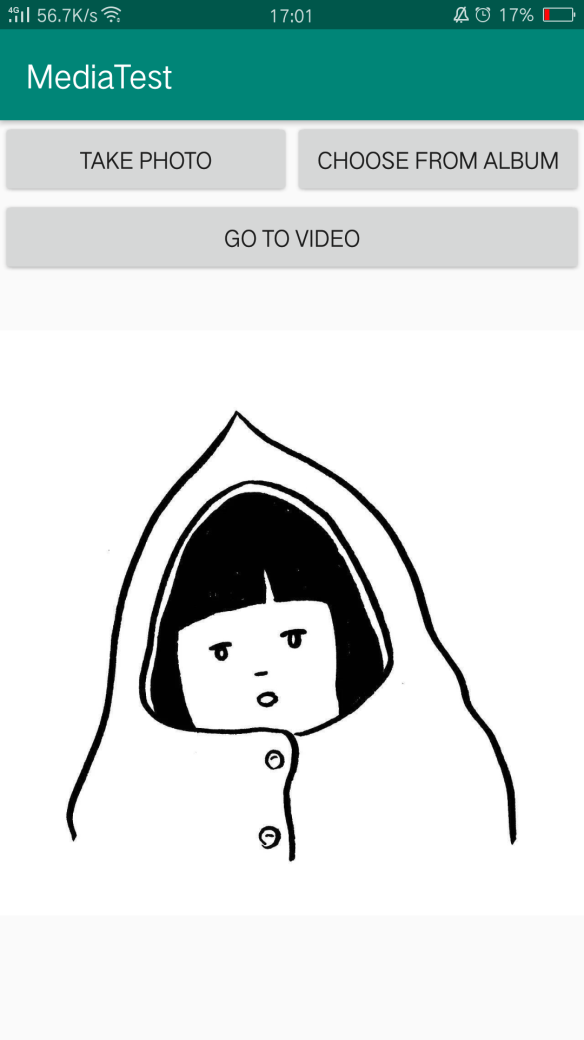
安卓版本号为4.4以下用此方法处理图片

private void handleImageBeforeKitKat(Intent data) {  
 Uri uri = data.getData();  
 String imagePath = getImagePath(uri, null);  
 displayImage(imagePath);  
}

获取土坯那路径、显示图片

private String getImagePath(Uri uri, String selection) {  
 String path = null;  
 // 通过Uri和selection来获取真实的图片路径  
 Cursor cursor = getContentResolver().query(uri, null, selection, null, null);  
 if (cursor != null) {  
 if (cursor.moveToFirst()) {  
 path = cursor.getString(cursor.getColumnIndex(MediaStore.Images.Media.*DATA*));  
 }  
 cursor.close();  
 }  
 return path;  
}  
  
private void displayImage(String imagePath) {  
 if (imagePath != null) {  
 Bitmap bitmap = BitmapFactory.*decodeFile*(imagePath);  
 picture.setImageBitmap(bitmap);  
 } else {  
 Toast.*makeText*(this, "failed to get image", Toast.*LENGTH\_SHORT*).show();  
 }  
}





视频播放判断权限

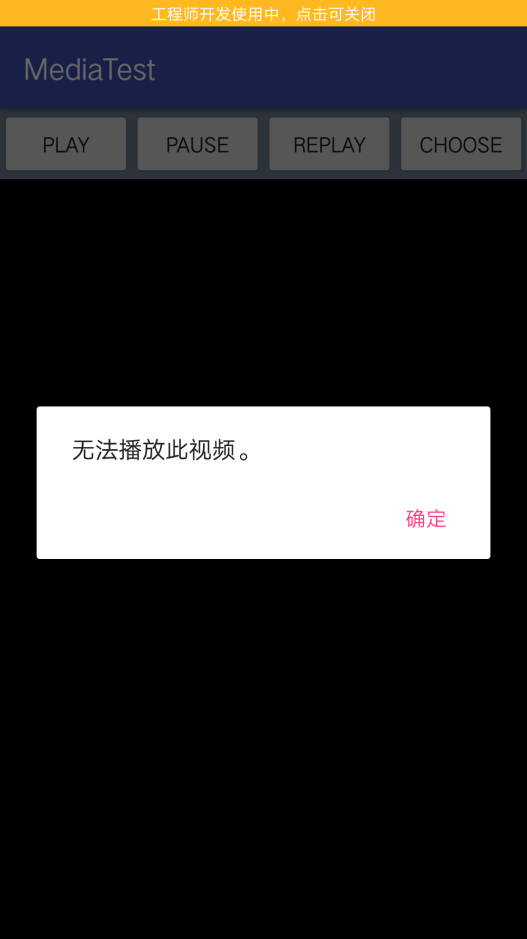
if(ContextCompat.*checkSelfPermission*(VideoPlay.this, Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*)!= PackageManager.*PERMISSION\_GRANTED*){  
 ActivityCompat.*requestPermissions*(VideoPlay.this,new String[]{Manifest.permission.*WRITE\_EXTERNAL\_STORAGE*},1);//判断你是否授权  
}  
else {  
 initVideoPath();  
}

请求权限结果

public void onRequestPermissionsResult(int requestCode,String[] permissions,int[] grantResults){  
 switch (requestCode){  
 case 1:  
 if(grantResults.length>0&&grantResults[0]==PackageManager.*PERMISSION\_GRANTED*){  
 initVideoPath();  
 }  
 else {  
 Toast.*makeText*(this,"拒绝权限将无法访问程序",Toast.*LENGTH\_SHORT*).show();  
 finish();  
 }  
 break;  
 default:  
 }  
}

1. 初始化路径“1.mp4”,由于根目录下没有该视频文件，所以弹出对话框“无法播放此视频”

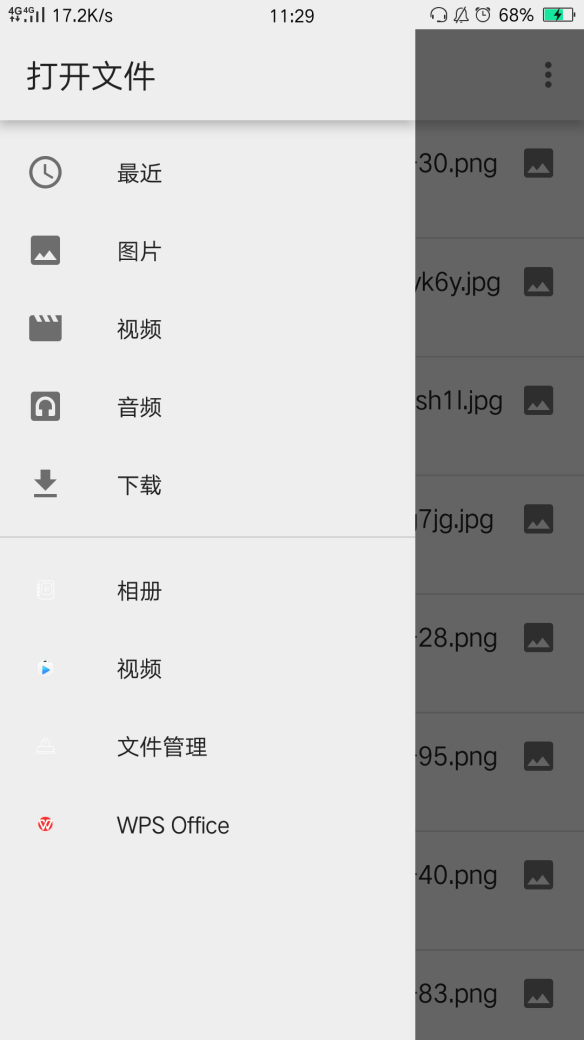
private void initVideoPath(){  
 File file=new File(Environment.*getExternalStorageDirectory*(),"movie.mp4");//打开软件直接播放的视频名字是movie.mp4  
 videoView.setVideoPath(file.getPath());//指定视频文件的路径  
}



1. 播放一段本人拍摄的小视频（拓展）增加选取文件播放功能，调用系统自带的文件浏览器选取视频文件路径

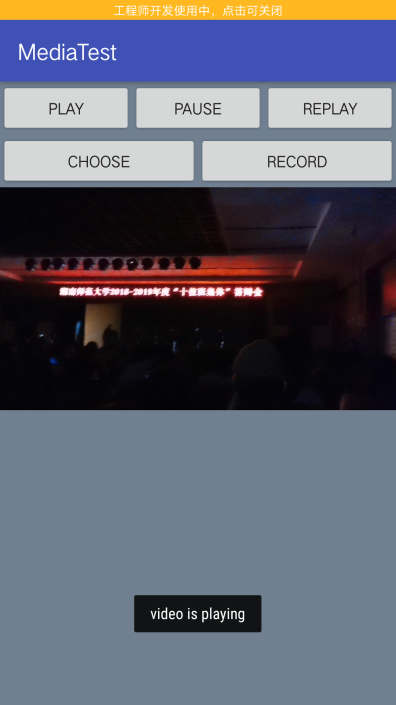
Intent intent=new Intent(Intent.*ACTION\_GET\_CONTENT*);  
intent.setType("\*/\*");//设置类型，这是任意类型  
intent.addCategory(Intent.*CATEGORY\_OPENABLE*);  
startActivityForResult(intent,1);

public void onActivityResult(int requestCode,int resultCode,Intent data){  
 if(resultCode== Activity.*RESULT\_OK*){  
 Uri uri=data.getData();  
 videoView.setVideoURI(uri);//将选择的文件路径给播放器  
 super.onActivityResult(requestCode, resultCode, data);  
 return;  
 }  
 if (requestCode == *FILE\_SELECT\_CODE*) {  
 Uri uri = data.getData();  
 Log.*i*(*TAG*, "------->" + uri.getPath());  
 }  
 super.onActivityResult(requestCode, resultCode, data);



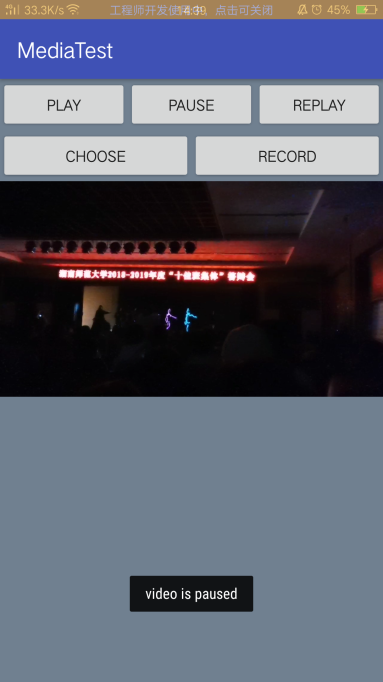
视频播放

if(!videoView.isPlaying()){//播放  
 videoView.start();  
}  
Toast.*makeText*(this,"video is playing",Toast.*LENGTH\_SHORT*).show();



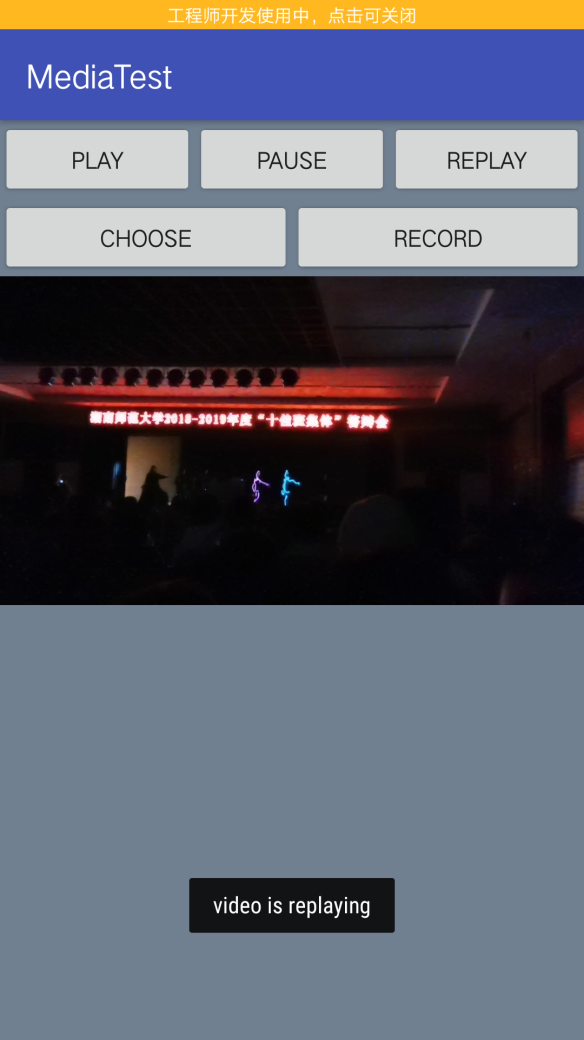
视频暂停

if(videoView.isPlaying()){//暂停  
 videoView.pause();  
}  
Toast.*makeText*(this,"video is paused",Toast.*LENGTH\_SHORT*).show();



视频重播

if(videoView.isPlaying()){  
 videoView.resume();//重新播放  
}  
Toast.*makeText*(this,"video is replaying",Toast.*LENGTH\_SHORT*).show();  
break;



释放资源

public void onDestroy(){//释放资源  
 super.onDestroy();  
 if(videoView!=null){  
 videoView.suspend();  
 }  
}