**移动终应用系统开发**

**实**

**验**

**指**

**导**

**书**

**章崇文**

**202202296**

**计算机222**

**计算机技术实验中心**

**2024年12月28日**

**实验四 图形图像和多媒体技术**

一、实验目的

1.掌握如何设置Paint画笔的相关属性

2.掌握如何利用Canvas绘制各种图形、图像，以及图像变换方法

3.掌握制作帧动画、补间动画的方法

4.掌握利用SoundPool播放音频的方法

5.掌握利用MediaPlayer播放音频的方法

6.掌握利用VideoView播放视频的方法

7.掌握利用MediaPlayer和SurfaceView播放视频的方法

**二、实验要求**

1.学会使用Painth画笔和Canvas画布绘制各种图形

2.掌握图像的四种变换方法

3.掌握制作帧动画、补间动画的方法

4.熟练掌握利用SoundPool播放音频的方法

5.熟练掌握利用MediaPlayer播放音频的方法

6.熟练掌握利用VideoView播放视频的方法

7.熟练掌握利用MediaPlayer和SurfaceView播放视频的方法

三、实验内容及步骤

程序1

要求：

**1.编写一个Android应用程序，使用Paint和Canvas绘制简单的2维图形。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）布局文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<FrameLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:id=*"@+id/frameLayout1"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"* >

<!—或将创建的MyView放这里，或在后台Java代码中添加-->

</FrameLayout>

**2） 在包下面，添加一个View的子类（也可作为Activtiy子类），参考代码如下：**

图形用户界面, 文本

中度可信度描述已自动生成

文本

描述已自动生成

文本, 信件

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

**3）后台Java代码，参考如下：**

文本, 信件

描述已自动生成

电子屏幕

描述已自动生成

程序2

要求：

**1.编写一个Android应用程序，利用Matrix生成对应的变换矩阵，然后使用Canvas调用对应的变换矩阵对图像进行变换。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）将素材文件夹中的“rabbit.png”拷贝到res目录下的drawable文件夹中。**

**2）后台Java代码，参考如下：**

文本, 信件

描述已自动生成

文本, 信件

描述已自动生成

文本

描述已自动生成

文本, 信件

描述已自动生成

文本, 信件

描述已自动生成

文本, 信件

描述已自动生成

猫站在电视萤幕画面

描述已自动生成

程序3

要求：

**1.编写一个Android应用程序，制作一个类似播放相处中相片的帧动画。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）将素材文件夹中的“p01.png，p02.png，p03.png... p10.png”拷贝到res目录下的drawable文件夹中。**

**2）在res目录中，添加一个“anim”子文件夹，并且在其中添加一个xml文件，文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<animation-list xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:oneshot=*"false"*>

<item android:drawable=*"@drawable/p01"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p02"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p03"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p04"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p05"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p06"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p07"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p08"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p09"*

android:duration=*"1000"*

android:visible=*"true"*/>

<item android:drawable=*"@drawable/p10"*

android:duration=*"1000"*

android:visible=*"true"*/>

</animation-list>

**3）在res下的布局文件代码文件如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:orientation=*"vertical"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*>

<Button

android:id=*"@+id/btn"*

android:text=*"播放帧动画"*

android:layout\_width=*"fill\_parent"*

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

android:layout\_gravity=*"center\_horizontal"*/>

<ImageView

android:id=*"@+id/iv"*

android:layout\_width=*"fill\_parent"*

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

android:layout\_gravity=*"center\_horizontal"*/>

</LinearLayout>

**4）后台Java代码如下：**

文本

描述已自动生成

手机屏幕的截图

描述已自动生成

程序4

要求：

**1.编写一个Android应用程序，通过对同一幅图像进行不同变换制作一个补间动画。**

**2.在空白处画出运行效果**

指导：

具体步骤如下：

**1）将素材文件夹中的“bigandroid.png”拷贝到res目录下的drawable文件夹中。**

**2）在res目录中，添加一个“anim”子文件夹，并且在其中添加一个xml文件，文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<set xmlns:android=*"http://schemas.android.com/apk/res/android"*>

<!-- 透明度的变换 -->

<alpha

android:fromAlpha=*"0.0"*

android:toAlpha=*"1.0"*

android:duration=*"5000"* />

<!-- 尺寸的变换，interpolator表示变换的速度变化模式-->

<scale

android:interpolator= *"@android:anim/accelerate\_decelerate\_interpolator"*

android:fromXScale=*"0.0"*

android:toXScale=*"1.0"*

android:fromYScale=*"0.0"*

android:toYScale=*"1.0"*

android:pivotX=*"50%"*

android:pivotY=*"50%"*

android:fillAfter=*"false"*

android:duration=*"9000"*/>

<!-- 位置的变换 -->

<translate

android:fromXDelta=*"30"*

android:toXDelta=*"0"*

android:fromYDelta=*"30"*

android:toYDelta=*"0"*

android:duration=*"10000"* />

<!-- 旋转变换 -->

<rotate

android:interpolator=*"@android:anim/accelerate\_decelerate\_interpolator"*

android:fromDegrees=*"0"*

android:toDegrees=*"+360"*

android:pivotX=*"50%"*

android:pivotY=*"50%"*

android:duration=*"10000"*/>

</set>

**3）在res目录下的布局文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:orientation=*"vertical"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*><!-- 声明一个垂直分布的线性布局 -->

<ImageView

android:id=*"@+id/iv"*

android:src=*"@drawable/bigandroid"*

android:layout\_width=*"wrap\_content"*

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

android:layout\_gravity=*"center\_horizontal"*/><!-- 声明一个ImageView控件 -->

<Button

android:id=*"@+id/btn1"*

android:text=*"播放补间动画"*

android:layout\_width=*"fill\_parent"*

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

android:layout\_marginRight=*"10dip"*/><!-- 声明一个Button控件 -->

</LinearLayout>

**4）后台Java代码，参考如下：**

文本

描述已自动生成

手机屏幕的截图

描述已自动生成

程序5

要求：

**1.编写一个Android应用程序，使用SoundPool播放音频。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）在res文件夹中新建一个raw文件夹，将素材文件夹中的cuckoo.wav、doorbell.wav、phonebell.wav、ring.wav 、windbell.wav复制到 raw文件夹中。**

**2)布局文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"horizontal"* >

<Button

android:id=*"@+id/button1"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"风铃声"* />

<Button

android:id=*"@+id/button2"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"布谷鸟叫声"* />

<Button

android:id=*"@+id/button3"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"门铃声"* />

<Button

android:id=*"@+id/button4"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"电话声"* />

</LinearLayout>

**3）后台Java代码，参考如下：**

图形用户界面, 文本

描述已自动生成

表格

低可信度描述已自动生成

文本, 信件

描述已自动生成

形状

描述已自动生成

程序6

要求：

**1.编写一个Android应用程序，使用MediaPlayer播放音频。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）在res文件夹中新建一个raw文件夹，将素材文件夹中的redsong.mp3复制到 raw文件夹中。**

**2）布局文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"* >

<TextView

android:id=*"@+id/hint"*

android:layout\_width=*"wrap\_content"*

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

android:padding=*"10px"*

android:text=*"单击“开始”按钮播放音频"* />

<LinearLayout

android:id=*"@+id/linearLayout1"*

android:layout\_width=*"match\_parent"*

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

<Button

android:id=*"@+id/button1"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"播放"* />

<Button

android:id=*"@+id/button2"*

android:layout\_width=*"wrap\_content"*

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

android:enabled=*"false"*

android:text=*"暂停"* />

<Button

android:id=*"@+id/button3"*

android:layout\_width=*"wrap\_content"*

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

android:enabled=*"false"*

android:text=*"停止"* />

</LinearLayout>

</LinearLayout>

**3）后台Java代码，参考如下：**

文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

文本, 信件

描述已自动生成

文本

描述已自动生成

电子设备的屏幕

描述已自动生成形状

描述已自动生成形状

描述已自动生成

程序7

要求：

**1.编写一个Android应用程序，编写一个Android应用程序，使用MediaPlayer+SurfaceView播放视频。**

**2.在空白处画出运行效果图。**

指导：

具体步骤如下：

**1）需要将某个视频文件导入到你手机SD卡中，例如把素材中的“beargolf.mp4”导入到手机的SD卡中**

**2）布局文件代码如下：**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:orientation=*"vertical"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

> <!--添加一个垂直的线性布局 -->

<SurfaceView

android:id=*"@+id/surfaceView"*

android:layout\_width=*"320px"*

android:layout\_height=*"240px"*

/> <!--添加一个SurfaceView用于播放视频 -->

<LinearLayout

android:layout\_width=*"fill\_parent"*

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

> <!--添加一个线性布局 -->

<Button

android:id=*"@+id/play2\_Button"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"播放"*

/> <!--添加一个按钮 -->

<Button

android:id=*"@+id/pause2\_Button"*

android:layout\_width=*"wrap\_content"*

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

android:text=*"暂停"*

/> <!--添加一个按钮 -->

</LinearLayout>

</LinearLayout>

3）后台Java代码，参考如下：

文本

描述已自动生成

文本, 信件

描述已自动生成

文本

描述已自动生成

图形用户界面, 应用程序

描述已自动生成电脑萤幕的截图

描述已自动生成

四、习题

1.如何设置画笔的颜色、笔触的宽度？如何绘制一个实心矩形？如何绘制字体大小为25的一行文本？如何绘制路径？

**（1）设置画笔颜色和笔触宽度**

在 Android 开发中，通常使用 Paint 类来操作画笔。以下是设置画笔颜色和笔触宽度的方法：

Paint paint = new Paint();

// 设置颜色，例如设置为红色，有多种方式，以下是常见的几种

// 方式一：使用 Color 类中预定义的颜色常量

paint.setColor(Color.RED);

// 方式二：通过 RGB 值来设置颜色，参数分别为红、绿、蓝分量（取值范围0-255）

paint.setColor(Color.rgb(255, 0, 0));

// 设置笔触宽度，单位为像素，例如设置宽度为 5 像素

paint.setStrokeWidth(5);

**（2）绘制实心矩形**

可以使用 RectF 类定义矩形的区域，然后结合 Paint 类和 Canvas 类来绘制实心矩形，示例代码如下：

Paint paint = new Paint();

// 设置为填充样式，绘制实心图形

paint.setStyle(Paint.Style.FILL);

RectF rect = new RectF(100, 100, 200, 200); // 定义矩形左上角坐标（100,100）和右下角坐标（200,200）

Canvas canvas; // 获取到有效的 Canvas 对象，例如在自定义 View 的 onDraw 方法中传入的 canvas

canvas.drawRect(rect, paint);

**（3）绘制字体大小为 25 的一行文本**

同样借助 Paint 类来设置文本相关属性，再通过 Canvas 类绘制，代码示例如下：

Paint paintText = new Paint();

paintText.setTextSize(25); // 设置字体大小为 25 像素

String text = "要绘制的文本内容";

canvas.drawText(text, x, y, paintText); // x 和 y 是文本绘制的起始坐标位置，例如（50, 50）表示从（50,50）这个点开始绘制文本

**（4）绘制路径**

使用 Path 类来定义路径，然后通过 Canvas 类的 drawPath 方法绘制，示例如下：

Path path = new Path();

path.moveTo(100, 100); // 移动到起始点（100,100）

path.lineTo(200, 200); // 绘制直线到（200,200）这个点

path.lineTo(300, 150); // 继续绘制直线到下一个点

// 可以继续添加更多的路径操作，如曲线等

canvas.drawPath(path, paint); // 使用定义好的画笔 paint 绘制路径

2. 请描述Matrix类的旋转变换方法setRotate(60)、postRotate(60)、preRotate(60)以及setRotate(60,100,150)的区别是什么？以及如何对一幅图像进行平移变换或缩放变换或扭曲变换并显示之？

**（1）旋转变换方法区别**

* **setRotate(60)**：  
  它会重置 Matrix 对象的变换矩阵，然后设置一个旋转角度为 60 度（默认绕坐标原点 (0, 0) 旋转）的旋转变换。后续的变换操作会基于这个新设置的旋转矩阵进行叠加，之前已有的变换效果会被覆盖掉。
* **postRotate(60)**：  
  它是在当前 Matrix 对象已有的变换矩阵基础上，再添加一个旋转 60 度（默认绕坐标原点 (0, 0) 旋转）的旋转变换，是一种后乘的方式，会与之前的变换效果进行组合，例如之前已经有平移等变换，执行此操作后旋转会叠加在已有变换之上。
* **preRotate(60)**：  
  它同样是添加一个旋转 60 度（默认绕坐标原点 (0, 0) 旋转）的旋转变换，但它是在前乘的方式下进行，意味着这个旋转操作会先作用于后续的变换，例如如果后续还有平移等变换，平移的坐标会先经过这次旋转的影响再进行实际的平移计算。
* **setRotate(60, 100, 150)**：  
  这个方法不仅设置了旋转角度为 60 度，还指定了旋转的中心点坐标为 (100, 150)，即会围绕该指定点进行旋转操作，同样会重置 Matrix 对象的变换矩阵后设置此旋转效果，覆盖之前的变换内容。

**（2）图像变换并显示（以平移、缩放、扭曲为例）**

* **平移变换**：

Matrix matrix = new Matrix();

matrix.postTranslate(dx, dy); // dx 和 dy 分别是水平和垂直方向的平移距离，例如（10, 20）表示向右平移 10 像素，向下平移 20 像素

Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.your\_image); // 获取要变换的图像资源

Canvas canvas; // 获取有效的 Canvas 对象用于绘制

canvas.drawBitmap(bitmap, matrix, paint); // 通过设置了平移变换的 matrix 绘制图像到 canvas 上

* **缩放变换**：

Matrix matrix = new Matrix();

matrix.postScale(sx, sy, px, py); // sx 和 sy 分别是水平和垂直方向的缩放比例因子，如（2, 2）表示宽高都放大 2 倍，px 和 py 是缩放的中心点坐标，若为 (0, 0) 则以左上角为中心缩放

Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.your\_image);

Canvas canvas;

canvas.drawBitmap(bitmap, matrix, paint);

* **扭曲变换（通过 setSkew 方法实现）**：

Matrix matrix = new Matrix();

matrix.setSkew(kx, ky); // kx 和 ky 分别是水平和垂直方向的扭曲系数，例如 (0.5, 0) 表示水平方向按一定比例扭曲

Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.your\_image);

Canvas canvas;

canvas.drawBitmap(bitmap, matrix, paint);

3. 如何制作一个帧动画？

步骤如下：

1. **准备帧图片资源**：将每一帧对应的图片放置在项目的 drawable 等资源目录下，例如 frame1.png、frame2.png 等。
2. **在 XML 中定义动画资源（可选，但方便管理）**：  
   在 res/anim 目录下创建一个 XML 文件（如果没有 anim 目录则需创建），示例如下：

<?xml version="1.0" encoding="utf-8"?>

<animation-list xmlns:android="http://schemas.android.com/apk/res/android"

android:oneshot="false"> <!-- oneshot 为 false 表示循环播放，true 表示只播放一次 -->

<item

android:drawable="@drawable/frame1"

android:duration="100" /> <!-- 每帧显示的时长，单位毫秒，这里设置为 100 毫秒 -->

<item

android:drawable="@drawable/frame2"

android:duration="100" />

<!-- 可以继续添加更多帧的配置 -->

</animation-list>

1. **在代码中使用动画资源**：

ImageView imageView = findViewById(R.id.your\_image\_view); // 获取用于显示动画的 ImageView

AnimationDrawable animationDrawable = (AnimationDrawable) getResources().getDrawable(R.drawable.your\_animation\_list); // 获取定义好的动画资源

imageView.setImageDrawable(animationDrawable);

animationDrawable.start(); // 启动动画

4.如何制作一个补间动画？

补间动画主要有四种类型：透明度动画（Alpha）、缩放动画（Scale）、平移动画（Translate）、旋转动画（Rotate），以下以平移动画为例说明基本步骤：

1. **在 XML 中定义动画（也可完全在代码中实现，这里以 XML 方式为例）**：  
   在 res/anim 目录下创建 XML 文件，如 translate\_animation.xml，示例如下：

<?xml version="1.0" encoding="utf-8"?>

<translate

xmlns:android="http://schemas.android.com/apk/res/android"

android:fromXDelta="0"

android:toXDelta="100"

android:fromYDelta="0"

android:toYDelta="0"

android:duration="1000" /> <!-- fromXDelta 和 fromYDelta 是起始坐标偏移量，toXDelta 和 toYDelta 是结束坐标偏移量，duration 是动画时长，单位毫秒 -->

1. **在代码中加载并启动动画**：

Animation animation = AnimationUtils.loadAnimation(this, R.anim.translate\_animation); // 加载动画资源

View view = findViewById(R.id.your\_view); // 获取要应用动画的视图

view.startAnimation(animation); // 启动动画

其他类型动画（Alpha、Scale、Rotate）在 XML 中定义时只需修改对应的属性即可，如 alpha 动画设置 fromAlpha 和 toAlpha 来控制透明度变化等。

5.使用SoundPool播放音频的主要步骤是什么？

1. **创建 SoundPool 对象**：

SoundPool soundPool = new SoundPool.Builder()

.setMaxStreams(maxStreams) // 设置最大同时播放的音频流数量，根据需求设定

.build();

1. **加载音频资源**：

int soundId = soundPool.load(context, R.raw.your\_audio\_file, 1); // 加载音频文件，第二个参数是音频资源在 res/raw 目录下的文件名，第三个参数是优先级，一般设为 1

1. **播放音频**：

soundPool.play(soundId, leftVolume, rightVolume, priority, loop, rate);

// soundId 是加载音频时返回的音频资源 ID，leftVolume 和 rightVolume 是左右声道音量（取值范围 0 - 1），pri

6. 如何使用MediaPlayer播放音频、暂停播放、停止播放音频？

**（1）播放音频**

MediaPlayer mediaPlayer = new MediaPlayer();

try {

mediaPlayer.setDataSource(context, Uri.parse("android.resource://" + getPackageName() + "/" + R.raw.your\_audio\_file)); // 设置音频数据源，这里以资源文件为例

mediaPlayer.prepare(); // 准备播放，可能会抛出异常，需处理

mediaPlayer.start(); // 开始播放

} catch (IOException e) {

e.printStackTrace();

}

**（2）暂停播放**

if (mediaPlayer!= null && mediaPlayer.isPlaying()) {

mediaPlayer.pause(); // 如果正在播放则暂停

}

**（3）停止播放**

if (mediaPlayer!= null) {

mediaPlayer.stop(); // 停止播放

mediaPlayer.reset(); // 重置 MediaPlayer 状态，以便后续重新使用

mediaPlayer.release(); // 释放资源，释放后需重新创建才能使用

}

7. 如何使用VideoView播放视频、暂停播放、停止播放？

**（1）播放视频**

VideoView videoView = findViewById(R.id.your\_video\_view);

videoView.setVideoPath("android.resource://" + getPackageName() + "/" + R.raw.your\_video\_file); // 设置视频文件路径，这里以资源文件为例

videoView.start(); // 启动视频播放

**（2）暂停播放**

if (videoView!= null && videoView.isPlaying()) {

videoView.pause(); // 如果正在播放则暂停

}

**（3）停止播放**

if (videoView!= null) {

videoView.stopPlayback(); // 停止视频播放

}

8. 如何使用MediaPlayer+SurfaceView播放视频、暂停播放、停止播放？

**（1）播放视频**

SurfaceView surfaceView = findViewById(R.id.your\_surface\_view);

SurfaceHolder surfaceHolder = surfaceView.getHolder();

MediaPlayer mediaPlayer = new MediaPlayer();

try {

mediaPlayer.setDataSource(context, Uri.parse("android.resource://" + getPackageName() + "/" + R.raw.your\_video\_file));

mediaPlayer.setDisplay(surfaceHolder); // 将 MediaPlayer 与 SurfaceView 的 SurfaceHolder 关联，用于显示视频画面

mediaPlayer.prepare();

mediaPlayer.start();

} catch (IOException e) {

e.printStackTrace();

}

**（2）暂停播放**

if (mediaPlayer!= null && mediaPlayer.isPlaying()) {

mediaPlayer.pause();

}

**（3）停止播放**

if (mediaPlayer!= null) {

mediaPlayer.stop();

mediaPlayer.reset();

mediaPlayer.release();

}

五、实验小结

**1 编程遇到的问题及解决方案**

**问题 1：**

* **问题描述**：使用 SoundPool 播放音频时，未正确处理音频加载的异步过程，导致播放失败或出现异常。
* **解决方案**：
  + 为 SoundPool 的 load 方法添加 OnLoadCompleteListener 监听器，确保音频加载完成后再播放，例如：

收起

java

soundPool = new SoundPool.Builder().setMaxStreams(10).build();

soundPool.setOnLoadCompleteListener(new SoundPool.OnLoadCompleteListener() {

@Override

public void onLoadComplete(SoundPool soundPool, int sampleId, int status) {

if (status == 0) {

soundPool.play(sampleId, 1, 1, 1, 0, 1); // 加载成功后播放音频

}

}

});

soundPool.load(context, R.raw.your\_audio, 1);

**问题 2：**

* **问题描述**：使用 MediaPlayer 时，忘记调用 prepare() 或 prepareAsync() 方法就直接调用 start()，会导致异常抛出。
* **解决方案**：
  + 对于本地文件，使用 setDataSource() 后，根据文件大小和性能需求，调用 prepare() （同步）或 prepareAsync() （异步）方法，再开始播放。例如：

收起

java

MediaPlayer mediaPlayer = new MediaPlayer();

mediaPlayer.setDataSource(context, Uri.parse("your\_audio\_url"));

mediaPlayer.prepareAsync(); // 或 prepare() 对于较小文件

mediaPlayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {

@Override

public void onPrepared(MediaPlayer mp) {

mp.start(); // 准备好后开始播放

}

});

**问题 3：**

* **问题描述**：在 MediaPlayer 状态转换过程中，如从暂停状态恢复播放时操作不当，可能会导致异常。
* **解决方案**：
  + 仔细处理 MediaPlayer 的状态机，在不同状态下执行相应操作。例如，从暂停恢复播放时，检查 isPlaying() 状态，若暂停则调用 start()：

收起

java

if (mediaPlayer!= null &&!mediaPlayer.isPlaying()) {

mediaPlayer.start();

}

**四、视频播放方面**

**问题 1：**

* **问题描述**：使用 VideoView 或 MediaPlayer + SurfaceView 播放视频时，未正确设置视频文件的路径或数据源，导致无法播放。
* **解决方案**：
  + 确保使用正确的文件路径或 URI 格式，对于资源文件使用 android.resource:// 协议，对于外部存储文件使用 file:// 协议，对于网络文件使用 http:// 或 https:// 协议。例如：

VideoView videoView = findViewById(R.id.videoView);

videoView.setVideoPath("android.resource://" + getPackageName() + "/raw/your\_video");

* 对于 MediaPlayer + SurfaceView：

MediaPlayer mediaPlayer = new MediaPlayer();

SurfaceView surfaceView = findViewById(R.id.surfaceView);

SurfaceHolder surfaceHolder = surfaceView.getHolder();

mediaPlayer.setDisplay(surfaceHolder);

mediaPlayer.setDataSource("android.resource://" + getPackageName() + "/raw/your\_video");

mediaPlayer.prepare();

mediaPlayer.start();

**问题 2：**

* **问题描述**：在使用 MediaPlayer + SurfaceView 时，没有正确处理 SurfaceHolder 的 Callback，导致视频无法显示或显示异常。
* **解决方案**：
  + 为 SurfaceHolder 添加 Callback，在 surfaceCreated 回调中进行 MediaPlayer 的初始化和播放操作，确保 Surface 准备好后再进行视频播放，例如：

SurfaceHolder surfaceHolder = surfaceView.getHolder();

surfaceHolder.addCallback(new SurfaceHolder.Callback() {

@Override

public void surfaceCreated(SurfaceHolder holder) {

try {

mediaPlayer.setDisplay(holder);

mediaPlayer.setDataSource("your\_video\_url");

mediaPlayer.prepare();

mediaPlayer.start();

} catch (IOException e) {

e.printStackTrace();

}

}

// 其他回调方法的实现

});

**问题 3：**

* **问题描述**：在视频播放过程中，未处理 MediaPlayer 的异常状态，如网络中断、文件格式不支持等。
* **解决方案**：
  + 为 MediaPlayer 添加 OnErrorListener，在出现错误时进行相应处理，例如：

mediaPlayer.setOnErrorListener(new MediaPlayer.OnErrorListener() {

@Override

public boolean onError(MediaPlayer mp, int what, int extra) {

// 处理错误，如重新加载、提示用户等

return false;

}

});

在程序调试过程中，使用 Android Studio 的 Logcat 工具查看日志输出，对于异常信息，可以根据错误堆栈信息定位问题所在。对于界面显示问题，可以使用布局检查工具检查布局和视图的属性设置是否正确。同时，多利用断点调试，观察变量值和程序执行流程，帮助理解代码运行过程中的问题。对于复杂问题，可以将大段代码拆分成小模块进行测试，逐步排查问题所在。

提供一些具体的代码示例，展示如何正确使用Path类的各种操作方法

绘制图形时，对Color类的ARGB值理解有误怎么办？

如何在Android中调试和解决程序运行时出现的错误？

**2 实验收获及体会**

**一、图形绘制方面**

**收获 1：掌握画笔（Paint）的使用**

* 学会使用 Paint 类来设置画笔的各种属性，包括颜色（通过 setColor() 方法，可以使用 Color 类的预定义常量，也可以使用 Color.rgb() 或 Color.argb() 自定义颜色）、笔触宽度（通过 setStrokeWidth() 方法）以及绘制风格（通过 setStyle() 方法，如 Paint.Style.STROKE 用于绘制空心图形，Paint.Style.FILL 用于绘制实心图形）。
* 理解了不同颜色设置方法的灵活性，能够根据具体的设计需求精确控制颜色的呈现。例如，在创建自定义 UI 元素时，可以为不同的元素选择合适的颜色组合，以实现丰富的视觉效果。

**收获 2：掌握图形绘制操作**

* 能够熟练使用 Canvas 类进行基本图形的绘制，如 drawCircle() 绘制圆形、drawRect() 绘制矩形、drawPath() 绘制路径。
* 对于 Path 类的使用有了深入理解，掌握了 moveTo() 用于指定路径起始点，lineTo() 用于添加直线段，以及如何组合多个 lineTo() 操作绘制复杂的折线路径。还学会了如何使用 addRect() 等方法为路径添加矩形等几何形状，实现更加多样化的图形绘制。
* 明白了如何根据不同的设计需求，使用这些图形绘制方法来创建自定义的图形元素，为 Android 应用的 UI 设计增添独特的视觉元素，如绘制自定义的图标、装饰线条等。

**收获 3：掌握文本绘制**

* 掌握了使用 Paint 类设置文本属性，如 setTextSize() 来调整字体大小，使用 drawText() 方法将文本绘制到 Canvas 上，同时理解了绘制文本时 x 和 y 坐标的含义（y 坐标表示基线位置）。
* 学会了使用 drawTextOnPath() 方法沿着指定路径绘制文本，为文本的呈现方式提供了更多创意和灵活性，可用于创建艺术字效果或让文本沿着特定的路径排列，如弯曲的文本标签。

**二、动画方面**

**收获 1：理解帧动画（AnimationDrawable）**

* 掌握了在 Android 中通过 XML 资源文件定义帧动画的方法，包括在 animation-list 元素中设置每个帧的 drawable 资源和 duration 时长，以及 oneshot 属性来控制动画是否循环播放。
* 学会了在代码中加载和启动帧动画，将 AnimationDrawable 应用到 ImageView 上并调用 start() 方法开始播放，实现了动画的循环或单次播放效果。
* 了解了如何准备多帧的图片资源，将静态的图片组合成动态的帧动画，为应用的 UI 元素添加生动的动画效果，例如制作加载动画或简单的动态图标。

**收获 2：掌握补间动画**

* 理解了不同类型的补间动画（如平移、旋转、缩放、透明度变化），可以通过 XML 或代码来定义这些动画。在 XML 中可以设置动画的起始值（如 fromXDelta、toXDelta 等）和持续时间 duration。
* 学会使用 AnimationUtils.loadAnimation() 方法从 XML 中加载动画资源，并应用到 View 上，通过 startAnimation() 方法启动动画。
* 理解了不同的插值器（Interpolator）对动画速度的影响，能够根据需要选择合适的插值器（如 AccelerateInterpolator 使动画加速、DecelerateInterpolator 使动画减速），为应用添加流畅自然的动画过渡效果，提升用户体验。

**收获 3：掌握 Matrix 类的变换操作**

* 深入理解了 Matrix 类的旋转方法，包括 setRotate() 用于重置矩阵并设置旋转，postRotate() 用于后乘矩阵添加旋转，preRotate() 用于前乘矩阵添加旋转，以及 setRotate(angle, x, y) 用于指定旋转中心。
* 学会使用 Matrix 类进行平移、缩放、扭曲等变换操作，如 postTranslate() 进行平移、postScale() 进行缩放，以及 setSkew() 进行扭曲。
* 可以将 Matrix 应用到 Bitmap 绘制中，通过 Canvas.drawBitmap() 结合 Matrix 对图像进行变换，实现复杂的图像变换效果，为应用开发中的图像特效、自定义视图的变换提供了强大的工具，比如实现图片的旋转、缩放、倾斜等效果。

**三、音频播放方面**

**收获 1：使用 SoundPool 播放音频**

* 掌握了 SoundPool 的创建，通过 SoundPool.Builder() 可以设置最大同时播放的音频流数量。
* 学会使用 load() 方法加载音频资源，并使用 play() 方法播放音频，同时理解了 play() 方法中参数的含义，如左右声道音量、播放优先级、循环次数和播放速率。
* 了解到 SoundPool 适合播放短音频效果，如按钮点击音效等，并且学会处理音频加载的异步过程，通过 OnLoadCompleteListener 确保音频加载完成后再播放，提高了应用的音频播放性能和可靠性。

**收获 2：使用 MediaPlayer 播放音频**

* 学会使用 MediaPlayer 播放较长的音频文件，掌握了 setDataSource() 方法设置音频源（可以是资源文件、文件路径或网络 URL），以及 prepare() 或 prepareAsync() 方法的区别和使用场景。
* 掌握了音频播放的基本控制操作，如 start() 开始播放、pause() 暂停播放、stop() 停止播放和 reset() 重置状态。
* 掌握了处理 MediaPlayer 的状态转换，在不同状态下正确操作 MediaPlayer，如在 OnPreparedListener 中开始播放音频，避免了因状态错误导致的异常，同时添加 OnErrorListener 来处理播放过程中的错误，保证了音频播放的稳定性和容错性。

**四、视频播放方面**

**收获 1：使用 VideoView 播放视频**

* 掌握了 VideoView 的基本使用，通过 setVideoPath() 方法设置视频文件的路径（可以是资源文件或其他文件路径），使用 start() 方法启动视频播放。
* 学会使用 pause() 和 stopPlayback() 方法控制视频的暂停和停止，为简单的视频播放需求提供了便捷的实现方式，适合快速实现视频播放功能，如在一个简单的视频播放应用中。

**收获 2：使用 MediaPlayer + SurfaceView 播放视频**

* 理解了 SurfaceView 和 SurfaceHolder 的关系，以及如何将 MediaPlayer 与 SurfaceView 结合使用。通过 SurfaceHolder 的 Callback 机制，在 surfaceCreated() 回调中进行 MediaPlayer 的设置和播放操作，确保 Surface 准备好后才开始播放视频。
* 掌握了在 MediaPlayer + SurfaceView 组合中，进行视频播放、暂停、停止的操作，同时处理 MediaPlayer 的状态转换和错误处理，为开发更复杂的视频播放功能提供了更灵活和强大的实现方式，如开发视频播放器应用，可更好地控制视频播放的细节，提高播放性能和用户体验。

**体会**

* 通过本次实验，深刻体会到了 Android 开发中丰富的 UI 设计和多媒体处理功能，能够为应用程序添加丰富多彩的图形、动画、音频和视频元素，极大地增强了应用的交互性和吸引力。
* 在开发过程中，理解文档和 API 参考的重要性，尤其是对于 Paint、Canvas、Animation 等类的使用，很多属性和方法的细节需要通过官方文档的仔细阅读才能深入理解，避免因对属性的误解而导致开发错误。
* 认识到调试的重要性，在出现问题时，使用 Logcat 查看日志，使用断点调试观察变量和程序流程，以及使用布局检查工具，这些调试手段有助于快速定位和解决问题。
* 对不同多媒体元素的开发流程有了系统的认识，从资源的准备（如图形资源、音频文件、视频文件），到代码的实现和逻辑控制，再到错误处理和优化，使自己在开发多媒体应用时能够更加得心应手，能够从用户体验的角度出发，为应用添加更具创意和实用的功能。

通过这次实验，不仅积累了技术知识，还培养了自己解决实际问题的能力和对 Android 开发的整体认知，为进一步开发更复杂的 Android 应用打下了坚实的基础。

**3 不足之处及下一步需改进的地方**

**一、图形绘制方面**

**不足 1：图形绘制的细节把控不够精确**

* 在使用 Paint 和 Canvas 进行图形绘制时，对于一些复杂图形的绘制，如带有曲线的图形（贝塞尔曲线等），还不够熟练，绘制的曲线形状可能无法精确达到预期效果。
* 对于绘制图形时坐标的计算和控制不够灵活，有时难以将图形精确地定位到想要的位置，特别是涉及到多个图形的组合和相对位置关系时，容易出现偏差。

**改进措施 1：**

* 深入学习曲线绘制相关的 Path 类方法，如 quadTo() 和 cubicTo()，通过更多的练习和实际案例，掌握如何精确控制曲线的形状和走向。可以参考一些开源项目或在线教程中的曲线绘制代码，理解其实现思路。
* 加强对图形坐标系统的理解，在绘制多个图形时，先规划好图形之间的相对位置关系，使用辅助线或注释在代码中明确标注坐标的计算逻辑，提高坐标计算的准确性。可以考虑封装一些辅助函数，将常用的坐标计算过程抽象出来，方便调用。

**不足 2：图形绘制性能考虑较少**

* 在绘制大量图形或复杂图形时，没有考虑性能优化，可能会导致 UI 响应速度变慢，尤其是在一些性能较低的设备上，可能出现卡顿现象。

**改进措施 2：**

* 学习 Android 图形绘制的性能优化技巧，例如使用 Canvas 的 clipRect() 方法对绘制区域进行裁剪，只绘制可见区域，避免不必要的绘制操作。
* 对于重复绘制的元素，考虑使用 Bitmap 缓存，将其存储在内存中，避免重复计算和绘制，提高绘制性能。在绘制复杂图形时，可以尝试使用 Canvas 的硬件加速功能，提高绘制速度，但同时要注意硬件加速可能带来的兼容性问题，需要进行充分的测试。

**二、动画方面**

**不足 1：动画的组合和衔接不够流畅**

* 在使用帧动画和补间动画时，对于多个动画的组合使用不够熟练，难以实现流畅的动画过渡和复杂的动画序列，例如从一个动画结束后如何平滑地过渡到另一个动画，动画的衔接不够自然。

**改进措施 1：**

* 深入研究动画的监听器，如 Animation.AnimationListener 中的 onAnimationEnd() 方法，在一个动画结束后触发另一个动画，实现多个动画的顺序执行和无缝衔接。
* 学习使用 AnimatorSet 来组合多个动画，通过设置动画的播放顺序、同时播放、延迟播放等，更好地控制动画的组合效果，实现更加复杂和流畅的动画序列，以满足复杂的 UI 交互需求。

**不足 2：对动画的性能优化和自定义效果的实现有限**

* 对于动画的性能优化缺乏了解，在动画运行时可能会出现卡顿现象，尤其是复杂的动画场景。
* 对于一些特殊的自定义动画效果，如物理动画（例如模拟物体的弹性运动），不知道如何实现。

**改进措施 2：**

* 了解 Android 动画的性能优化技巧，如减少不必要的属性动画更新频率，使用 ViewPropertyAnimator 代替 ObjectAnimator 进行简单的动画，它会自动优化性能。
* 学习 Android 中的物理动画库，如 DynamicAnimation，或者使用自定义的 ValueAnimator 结合物理公式，实现更具创意和自定义的动画效果，提升动画的流畅性和逼真度。

**三、音频播放方面**

**不足 1：对音频播放的高级功能掌握不足**

* 仅掌握了基本的音频播放、暂停、停止操作，对于音频的音效处理（如均衡、混响等）、音频流的动态控制（如实时调整音量、音高）等高级功能不熟悉。

**改进措施 1：**

* 深入学习 Android 的音频处理 API，如 AudioTrack 类，探索如何实现音频的音效处理和高级音频控制功能。
* 研究开源的音频处理库，如 TarsosDSP，学习如何使用这些库实现音频的各种高级功能，为音频播放添加更多的特效，提升用户的听觉体验。

**不足 2：音频播放的错误处理不够完善**

* 在使用 SoundPool 和 MediaPlayer 时，对于一些异常情况的处理还比较简单，仅能处理一些基本的错误，对于网络音频播放时可能出现的复杂网络错误（如网络抖动、缓冲不足等）缺乏应对能力。

**改进措施 2：**

* 完善音频播放的错误处理逻辑，针对不同的错误情况，如网络异常、音频文件格式不兼容、播放权限问题等，编写更加详细和健壮的错误处理代码。
* 对于网络音频播放，添加缓冲机制，通过监听 MediaPlayer 的缓冲状态，显示缓冲进度条，提高用户体验，确保音频播放的稳定性。

**四、视频播放方面**

**不足 1：对视频播放的高级功能不熟悉**

* 仅实现了基本的视频播放、暂停、停止操作，对于视频的高级功能，如视频的快进、快退、进度条拖动、字幕添加、视频滤镜等功能还不会实现。

**改进措施 1：**

* 学习 MediaPlayer 的更多方法和属性，例如 seekTo() 实现视频的快进和快退操作，掌握如何与 SeekBar 结合实现视频进度的拖动控制。
* 研究如何为视频添加字幕，例如使用 MediaPlayer 与 SubtitleView 结合，以及如何使用 OpenGL 等技术为视频添加滤镜，提升视频播放的功能性和用户体验。

**不足 2：视频播放的兼容性和性能问题**

* 在不同设备和屏幕分辨率下，视频播放可能会出现画面拉伸、模糊或性能不佳的情况，对于视频的编解码格式的兼容性问题考虑较少。

**改进措施 2：**

* 学习如何根据不同的设备屏幕比例调整视频的显示比例，避免画面拉伸或变形，使用 SurfaceView 的 SurfaceHolder 回调方法，根据 Surface 的大小调整视频的显示方式。
* 研究视频编解码的知识，了解不同的视频格式和编解码器，确保在不同设备上能够兼容更多的视频文件，同时考虑使用 ExoPlayer 等第三方库，提高视频播放的性能和兼容性。

**五、综合方面**

**不足 1：代码结构和可维护性不够好**

* 在实现图形、动画、音频和视频功能时，代码的结构可能比较混乱，不同功能的代码交织在一起，可维护性和扩展性较差，不利于后期的维护和功能扩展。

**改进措施 1：**

* 学习设计模式，如 MVC（Model-View-Controller）、MVP（Model-View-Presenter）或 MVVM（Model-View-ViewModel），将不同功能的代码分离到不同的模块中，提高代码的结构清晰度和可维护性。
* 对代码进行合理的封装和抽象，将重复的代码逻辑封装成函数或类，使代码更简洁，易于理解和修改。

**不足 2：缺乏对用户体验的综合考虑**

* 在实现功能时，主要关注技术实现，对于用户体验的整体考虑不够，例如在动画播放时没有添加动画的取消功能，音频和视频播放时的用户提示较少。

**改进措施 2：**

* 在开发过程中，更多地从用户的角度出发，添加用户友好的交互元素，如动画的取消按钮、音频视频播放的加载提示、播放进度显示等，提高用户体验。
* 进行用户测试，收集用户反馈，根据反馈优化功能和用户体验，让开发的应用更加贴合用户的实际需求。

通过总结这些不足之处和改进措施，在今后的学习和开发中可以更有针对性地进行学习和实践，不断提升自己在 Android 多媒体开发领域的能力，开发出更加优质和用户友好的应用程序。