

Android中的多媒体



智能设备教研室



现在手机已经不是单纯的通讯工具,已经集成照相机,音乐播放器,视频播放器,游戏机等的种种功能。

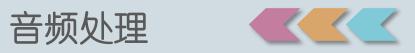




- > Android中提供了简单的API实现这些功能。
 - -声音的录制和播放
 - 视频的播放
 - 照相机的使用







- 视频处理
- 摄像头应用



使用MediaPlayer播放音频

- Android中提供了MediaPlayer类进行音频的播放,其常用方法:
 - start():开始或者恢复播放
 - stop():停止播放
 - pause(): 暂停播放
 - prepare():准备就绪
 - reset(): 重置MediaPlay
 - static MediaPlayer creat(Context,Uri):从URI来装载音频文件,返回 MediaPlayer对象
 - static MediaPlayer creat(Context,resid):从resid资源中装载音频文件,返回MediaPlayer对象
 - void setDataSource(...): 动态的获取资源

使用MediaPlayer播放音频

- setOnCompletionListener():播放完成的事件监听
- setOnErrorListener():播放错误的事件监听
- setOnPreparedListener(): prepare方法的事件监听
- setOnSeekCompleteListener(): seek方法的事件监听
- ➤ MediaPlayer缺点
 - -资源占用量高,延迟时间长
 - 不支持多个音频同时播放



➤ MediaPlayer播放音频文件

```
//播放raw文件中的音频文件
MediaPlayer mediaPlayer = MediaPlayer.create(this, R.raw.m1);
mediaPlayer.start();
```



- ➤ MediaPlayer播放音频文件的步骤
 - 创建MediaPlayer对象

```
//播放网络文件
MediaPlayer mediaPlayer = new MediaPlayer();
mediaPlayer.setDataSource("MP3网络地址");
mediaPlayer.prepare();
mediaPlayer.start();
```



- ▶ 如果程序中经常播放密集短促的音效,使用SoundPool (音效池)来管理多个短促的音效。
- ▶ 特点:
 - CPU占用量低
 - 反应延迟小
 - 可设置声音的品质、音量、播放比率等参数

使用SoundPool播放音效

▶ 常用方法:

- build():通过Builder对象构造SoundPool对象
- int load (...) : 加载声音,返回当前声音的ID
 - load(Context context, int resld, int priority)
 - load(FileDescriptor fd, long offset, long length, int priority)
 - load(AssetFileDescriptor afd, int priority)
 - load(String path, int priority)
 - 注意: priority一般设为1
- int play(int soundID, float leftVolume, float rightVolume, int priority, int loop, float rate)
 - 声音id;左、右声道音量;优先级;是否循环(0:不循环,-1:循环);播放比率(0.5~2,正常比率为1)



➤ 使用SoundPool的步骤:

- 通过new SoundPool.Builder()构造SoundPool的Builder对象
- 给Builder对象设置参数(AudioAttributes)[可选项]
- 通过Builder对象的build()方法构造SoundPool对象
- 调用SoundPool对象的load方法加载资源,最好使用HashMap来管理所加载的声音
- 给SoundPool注册加载声音资源完成监听器,并在监听器实现中调用SoundPool对象的play方法播放声音



▶ 注意:

- 可实现多个声音资源同时播放,只需在监听器实现中同时设置 play多个资源即可,不添加此监听器,可能出现听不到声音的情况
- Android5.0 (API21)以后使用Builder构造器创建SoundPool 对象, Android5.0之前使用SoundPool的构造方法创建其对象

```
//API21 (android5. 0) 以上版本推荐创建SoundPool对象的方法
SoundPool.Builder builder = new SoundPool.Builder();//创建Builder对象
//设置最多允许容纳的音频流
builder.setMaxStreams(10);
final SoundPool sp = builder.build();

//API21 (android5. 0) 以下版本创建SoundPool对象的方法
// 初始化soundPool,设置可容纳12个音频流,音频流的质量为5
SoundPool sp = new SoundPool(10, 0, 5);
```

使用SoundPool播放音效

```
//定义一个HashMap用于存放音频流的ID
final Map<Integer, Integer> musicId = new HashMap();
//通过load方法加载指定音频流,并将返回的音频ID放入musicId中
musicId.put(1, sp.load(getApplicationContext(), R.raw.x1, 1));
sp.setOnLoadCompleteListener(new SoundPool.OnLoadCompleteListener() {
   public void onLoadComplete(SoundPool soundPool, int sampleId, int status) {
       //指定播放多个音频流,可以同时播放
       soundPool.play(musicId.get(1), 1, 1, 0, 0, 1);
```



使用MediaRecorder录制音频

- ➤ 手机一般都提供了麦克风硬件, Android提供 MediaRecorder进行声音的录制。
- ▶ 操作步骤:
 - 创建MediaRecorder对象
 - 调用MediaRecroder对象的setAudioSource方法设置声音来源,一般是MediaRecorder.AudioSource.MIC
 - 调用MediaRecroder对象的setOutputFormat设置录制音频的文件格式
 - 调用MediaRecroder对象的对应方法设置音频编码,编码位率,采样率



▶ 操作步骤:

- 调用MediaRecorder对象的setOutputFile方法设置文件保存的 位置
- 调用MediaRecorder对象的prepare方法准备录制
- 调用MediaRecorder对象的start方法开始录制
- 录制完成,调用MediaRecorder对象的stop方法停止录制,调用release方法释放资源
- 需设置允许录制音频权限

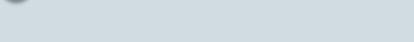
使用MediaRecorder录制音频

```
//定义一个HashMap用于存放音频流的ID
mr = new MediaRecorder(); //设置音源, 这里是来自麦克风
mr.setAudioSource(MediaRecorder.AudioSource.MIC);
//输出格式为3gp格式
mr.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
mr.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB); //编码
mr.setOutputFile(getFilesDir()+"/sound.3gp"); //输出文件路径
mr.prepare(); //准备就绪
mr.start(); //开始录制
```

mr.stop(); //停止录制 mr.release(); //释放资源



- **音频处理**
 - 视频处理



人 摄像头应用



使用VideoView播放视频

- ➤ Android通过VideoView组件进行视频的播放。
- ▶ 操作步骤:
 - 在界面布局中定义VideoView组件,或通过Java代码定义
 - 调用VideoView的方法加载视频:
 - setVideoPath(String path): 加载视频文件
 - setVideoURI(Uri uri):加载uri所对应的视频
 - 调用VideoView的start, stop, pause控制视频的播放

```
VideoView vv = findViewById(R.id.vv_video);
//创建MediaController对象
MediaController controller = new MediaController(this);
File video = new File(getFilesDir() + "/life3.mp4");
if (video.exists()) {
    //加载视频资源
    vv.setVideoPath(video.getAbsolutePath());
    //关联视频控件与MediaController
    vv.setMediaController(controller);
```



- 音频处理
 - 视频处理
- 摄像头应用







- ➤ Android提供了Camera类来实现摄像头相关的功能。步骤如下:
 - 调用CameraManager的openCamera方法打开摄像头
 - 获取摄像头设备
 - 创建CameraCaptureSession对象
 - 通过createCaptureRequest方法设置拍照或预览效果参数
 - 调用CameraCaptureSession对象的capture方法进行拍照



> 可以调用系统摄像头拍照应用并显示拍照后的图片

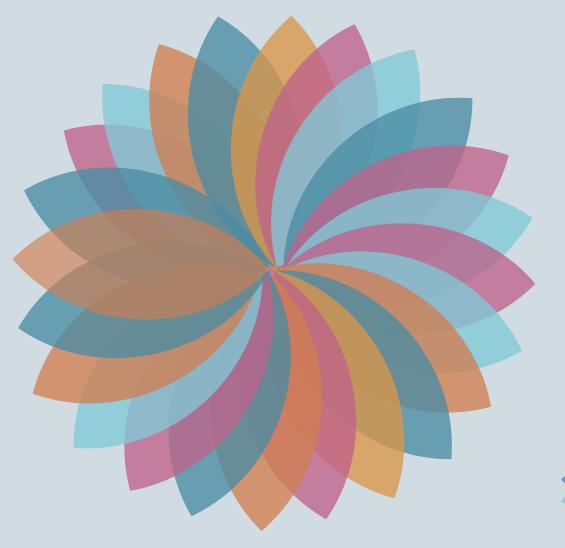
```
//调用系统摄像头拍照,注意需要添加允许操作摄像头权限
Intent cameraIntent = new
Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
startActivityForResult(cameraIntent, CAMERA_REQUEST);
```

```
//在Activity的onActivityResult回调方法中获取拍照结果并显示
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == CAMERA_REQUEST && resultCode == RESULT_OK) {
        Bitmap photo = (Bitmap) data.getExtras().get("data");
        ivPhoto.setImageBitmap(photo);
    }
}
```



- **音频处理**
 - **心** 视频处理
- **人** 摄像头应用







Thank You!

