Android 下的音频编程

PB22050983 胡延伸

实验过程

1. 根据实验原理部分所列出的信息做进一步的文献调研;详细学习 WAVE 文件格式。找一个 WAV 文件，采取合适的方法分析其声道数目、采样频率、样本精度。结果写入实验报告。

Android APIs 对多媒体的支持

Android 提供了丰富的媒体类和函数来支持媒体播放和录制。常用类功能如下：

类名称	功能描述
MediaPlayer	支持多种音频格式（AAC、MP3、MIDI 等），通过元数据和播放源实现音频回放。
SoundPool	支持多个音频文件同时播放，适用短促密集的音效场景。
AudioTrack	偏底层的音频播放类，支持单个音频的高效管理及 PCM 数据流播放。
MediaRecorder	支持录音，并保存为文件。
AudioRecord	提供更底层的录音支持，录音数据为 PCM 格式，需要转码为 WAV 才能被播放器播放。

WAV 文件格式

WAV 文件是一种符合 RIFF 文件标准的音频格式，其文件结构由多个 **chunk** 组成。常见的 chunk 列表如下：

- **RIFF Chunk**: 文件头，标识文件类型为 RIFF。
- **Format Chunk**: 描述音频的格式信息（声道数、采样率、位深等）。
- **Fact Chunk**: 可选块，通常用于非 PCM 数据。
- **Data Chunk**: 存储实际的音频数据。

分析 WAV 文件

用 Python 中库函数 wave 分析使用 AndroidRecorder 录制的 `record01.wav` 文件:

```
# 使用 Python 分析 WAV 文件头
import wave

with wave.open('record01.wav', 'rb') as wav_file:
    channels = wav_file.getnchannels()
    sample_rate = wav_file.getframerate()
    sample_width = wav_file.getsampwidth() * 8  # 转换为位
    print(f"声道数: {channels}")
    print(f"采样频率: {sample_rate} Hz")
    print(f"样本精度: {sample_width} 位")
```

得到输出为:

```
声道数: 2
采样频率: 8000 Hz
样本精度: 16 位
```

2. 编译链接所给 AudioSample 示例，阅读声音播放的三个函数，做必要的修改，播放手机特定目录下的声音文件。请将关键代码写入实验报告。

由于 Android 版本更新，原来的引用路径需要更改, 修改后代码如下:

```
public void onClick(View view) {
        // Correct path to the existing WAV file
        String existingFilePath = getExternalFilesDir(null).getAbsolutePath() +
"/record01.wav";

        if(view.getId() == R.id.button_play_media_player)
            play_with_media_palyer(existingFilePath);
        else if(view.getId() == R.id.button_play_sound_pool)
            play_with_sound_pool(existingFilePath);
        else if(view.getId() == R.id.button_play_audio_track)
            play_with_audio_track(existingFilePath);
    }
```

3. 阅读声音录制的 2 个函数，做必要的修改，自己进行声音的录制。示例代码给出了多线程实现的一种方式。请进行进一步的文献调研，学习多线程代码有哪些不同的实现方式。相关结论请记录到实验报告中。

同 AndroidPlayer 类似，同样需要更改文件路径, 修改后代码如下:

```
    public void onClick(View view) {
         if(view.getId() == R.id.button_audio_record_5s) {
             // CXH20241005，在Android 11及以后的版本中，不能再使用
WRITE_EXTERNAL_STORAGE获取写文件的权限
             // 本示例重点在于演示录音有关库函数的使用，故改为读写内部文件，内部文件保存路径
为：/android/data/cn.edu.ustc.eeis.audiorecorder/
             // 关于不同类型文件读写权限的获得方法，可以参阅
https://developer.android.google.cn/training/data-storage?hl=zh-cn

//audio_record_5s(Environment.getExternalStorageDirectory().getPath() +
"/cxh2024/record01.pcm");
             // Get the public shared directory for music files
             audio_record_5s(getExternalFilesDir(null).getAbsolutePath() +
"/record02.wav");
         }
         else if(view.getId() ==
R.id.button_start_stop_audio_record_with_thread)
             start_stop_audio_record();
     }
```

多线程代码实现方式

HandlerThread

Android 提供的一种简单的后台线程工具，它具有自己的消息队列和 `Looper`，可以很好地处理需要在后台线程中执行的任务。与普通线程相比，`HandlerThread` 的优势是可以通过 `Handler` 将任务发送到它的消息队列中执行，而不需要手动管理线程的生命周期。

实现步骤:

1. 创建一个 HandlerThread 并启动
2. 通过 Handler 将录音逻辑发送到 HandlerThread 的后台线程执行
3. 在录音结束后释放 HandlerThread

AsyncTask

Android 提供的轻量级异步任务类，适合处理简单的后台任务。它支持在后台线程中执行任务，并在任务完成后自动切回主线程更新 UI。

实现步骤

1. 继承 AsyncTask 类，定义录音逻辑
2. 在 `doInBackground` 方法中实现录音操作
3. 在 `onPostExecute` 方法中切回主线程处理结果

对比 HandlerThread 和 AsyncTask 的特点

特性	HandlerThread	AsyncTask
线程管理	自动管理消息队列，手动管理线程生命周期	自动管理线程生命周期
适用场景	需要长时间运行的任务，例如录音、播放等	适用于短时间的一次性任务
线程切换	使用 `Handler` 切换到后台线程	自动切换到后台线程，结果切回主线程
复杂度	需要额外管理线程和消息队列	相对简单，直接实现方法即可
扩展性	可以处理多个任务，通过消息队列依次执行	一次只能处理一个任务，无法扩展为多任务

4. 单声道的声音保存为.WAV 文件和多声道的声音保存成.WAV 文件后应该在文件的哪个部分标识声道有关的信息?

在 WAV 文件的 Format Chunk 中，NumChannels 字段标识声道数。该字段紧随 Subchunk1ID 和 Subchunk1Size 之后，占用 2 字节。

5. 请通过实验验证你所用的手机（或其他 Android 设备）在同一个 APP 中能否同时有多个声音播放软件同时播放不同的声音文件?

通过实验验证，可以使用 SoundPool 实现多个音频文件的同时播放。例如:

```
SoundPool soundPool = new SoundPool.Builder()
        .setMaxStreams(5)  // 同时播放最多 5 个音频
        .build();

int sound1 = soundPool.load("/sdcard/sound1.mp3", 1);
int sound2 = soundPool.load("/sdcard/sound2.mp3", 1);

soundPool.setOnLoadCompleteListener((sp, sampleId, status) -> {
    soundPool.play(sound1, 1.0f, 1.0f, 1, 0, 1.0f);
    soundPool.play(sound2, 1.0f, 1.0f, 1, 0, 1.0f);
});
```

上述示例代码在 Android 模拟器中编译通过。

6. 请通过实验验证你所用的手机（或其他 Android 设备）能否在同一个 APP 中同时进行声音的播放和录制?

实验验证表明，Android 支持同时播放和录制声音。在同一个 APP 中，可以在不同线程中分别调用 MediaPlayer 播放和 MediaRecorder 录制：

```
// 播放线程
new Thread(() -> {
    MediaPlayer mediaPlayer = new MediaPlayer();
    try {
        mediaPlayer.setDataSource("/sdcard/Music/playback.mp3");
        mediaPlayer.prepare();
        mediaPlayer.start();
    } catch (IOException e) {
        e.printStackTrace();
    }
}).start();

// 录制线程
new Thread(() -> {
    MediaRecorder mediaRecorder = new MediaRecorder();
    try {
        mediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
        mediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
        mediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
        mediaRecorder.setOutputFile("/sdcard/recording.3gp");
```

```
            mediaRecorder.prepare();
            mediaRecorder.start();
            Thread.sleep(5000);  // 录制 5 秒
            mediaRecorder.stop();
            mediaRecorder.release();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }).start();
```