

Android 录音实现 (AudioRecord)



DreamFish (/u/4ae79ad13c99) [+ 关注](#)

2017.07.14 23:27* 字数 829 阅读 679 评论 3 喜欢 8

(/u/4ae79ad13c99)



上一篇文章介绍了使用 MediaRecorder 实现录音功能 Android录音实现(MediaRecorder) (<http://www.jianshu.com/p/de779d509e6c>)，下面我们继续看看使用 AudioRecord 实现录音功能。

AudioRecord

首先看看Android帮助文档中对该类的简单概述: AndioRecord 类的主要功能是让各种 Java 应用能够管理音频资源，以便它们通过此类能够录制平台的声音输入硬件所收集的声音。此功能的实现就是通过 "pulling 同步" (reading读取) AudioRecord 对象的声音数据来完成的。在录音过程中，应用所需要做的就是通过后面三个类方法中的一个去及时地获取 AudioRecord 对象的录音数据。 AudioRecord 类提供的三个获取声音数据的方法分别是 read(byte[], int, int), read(short[], int, int), read(ByteBuffer, int)。无论选择使用那一个方法都必须事先设定方便用户的声音数据的存储格式。



开始录音的时候，一个 AudioRecord 需要初始化一个相关联的声音buffer，这个 buffer

主要是用来保存新的声音数据。这个 buffer 的大小，我们可以在对象构造期间去指定。它表明一个 AudioRecord 对象还没有被读取（同步）声音数据前能录多长的音(即一次可以录制的声音容量)。声音数据从音频硬件中被读出，数据大小不超过整个录音数据的大小（可以分多次读出），即每次读取初始化 buffer 容量的数据。

采集工作很简单，我们只需要构造一个AudioRecord对象，然后传入各种不同配置的参数即可。一般情况下录音实现的简单流程如下：

1. 音频源:我们可以使用麦克风作为采集音频的数据源。
2. 采样率:一秒钟对声音数据的采样次数，采样率越高，音质越好。
3. 音频通道：单声道，双声道等，
4. 音频格式:一般选用PCM格式，即原始的音频样本。
5. 缓冲区大小:音频数据写入缓冲区的总数，可以通过AudioRecord.getMinBufferSize获取最小的缓冲区。（将音频采集到缓冲区中然后再从缓冲区中读取）。

代码实现如下：



```
public class AudioRecorder {
    private static AudioRecorder audioRecorder;
    // 音频源：音频输入-麦克风
    private final static int AUDIO_INPUT = MediaRecorder.AudioSource.MIC;
    // 采样率
    // 44100是目前的标准，但是某些设备仍然支持22050，16000，11025
    // 采样频率一般共分为22.05KHz、44.1KHz、48KHz三个等级
    private final static int AUDIO_SAMPLE_RATE = 16000;
    // 音频通道 单声道
    private final static int AUDIO_CHANNEL = AudioFormat.CHANNEL_IN_MONO;
    // 音频格式：PCM编码
    private final static int AUDIO_ENCODING = AudioFormat.ENCODING_PCM_16BIT;
    // 缓冲区大小：缓冲区字节大小
    private int bufferSizeInBytes = 0;
    // 录音对象
    private AudioRecord audioRecord;
    // 录音状态
    private Status status = Status.STATUS_NO_READY;
    // 文件名
    private String fileName;
    // 录音文件集合
    private List<String> fileNameList = new ArrayList<>();

    private AudioRecorder() {
    }

    //单例模式
    public static AudioRecorder getInstance() {
        if (audioRecorder == null) {
            audioRecorder = new AudioRecorder();
        }
        return audioRecorder;
    }

    /**
     * 创建录音对象
     */
    public void createAudio(String fileName, int audioSource, int sampleRateInHz, int
        // 获得缓冲区字节大小
        bufferSizeInBytes = AudioRecord.getMinBufferSize(sampleRateInHz,
            channelConfig, audioFormat);
        audioRecord = new AudioRecord(audioSource, sampleRateInHz, channelConfig, au
            this.fileName = fileName;
    }

    /**
     * 创建默认的录音对象
     * @param fileName 文件名
     */
    public void createDefaultAudio(String fileName) {
        mContext = ctx;
        mHandler = handler;
        // 获得缓冲区字节大小
        bufferSizeInBytes = AudioRecord.getMinBufferSize(AUDIO_SAMPLE_RATE,
            AUDIO_CHANNEL, AUDIO_ENCODING);
        audioRecord = new AudioRecord(AUDIO_INPUT, AUDIO_SAMPLE_RATE, AUDIO_CHANNEL,
            this.fileName = fileName;
        status = Status.STATUS_READY;
    }
}
```



```
}

/**
 * 开始录音
 * @param listener 音频流的监听
 */
public void startRecord(final RecordStreamListener listener) {

    if (status == Status.STATUS_NO_READY || TextUtils.isEmpty(fileName)) {
        throw new IllegalStateException("录音尚未初始化, 请检查是否禁止了录音权限-");
    }
    if (status == Status.STATUS_START) {
        throw new IllegalStateException("正在录音");
    }
    Log.d("AudioRecorder", "===startRecord=== "+audioRecord.getState());
    audioRecord.startRecording();

    new Thread(new Runnable() {
        @Override
        public void run() {
            writeDataToFile(listener);
        }
    }).start();
}

/**
 * 停止录音
 */
public void stopRecord() {
    Log.d("AudioRecorder", "===stopRecord===");
    if (status == Status.STATUS_NO_READY || status == Status.STATUS_READY) {
        throw new IllegalStateException("录音尚未开始");
    } else {
        audioRecord.stop();
        status = Status.STATUS_STOP;
        release();
    }
}

/**
 * 取消录音
 */
public void cancel() {
    fileName.clear();
    fileName = null;
    if (audioRecord != null) {
        audioRecord.release();
        audioRecord = null;
    }
    status = Status.STATUS_NO_READY;
}

/**
 * 释放资源
 */
public void release() {
    Log.d("AudioRecorder", "===release===");
    //假如有暂停录音
    try {
```



```
        if (fileName.size() > 0) {
            List<String> filePaths = new ArrayList<>();
            for (String fileName : fileName) {
                filePaths.add(FileUtils.getPcmFileAbsolutePath(fileName));
            }
            //清除
            fileName.clear();
            //将多个pcm文件转化为wav文件
            mergePCMFilesToWAVFile(filePaths);

        } else {
            //这里由于只要录音过fileName.size都会大于0, 没录音时fileName为null
            //会报空指针 NullPointerException
            // 将单个pcm文件转化为wav文件
            //Log.d("AudioRecorder", "=====makePCMFileToWAVFile=====");
            //makePCMFileToWAVFile();
        }
    } catch (IllegalStateException e) {
        throw new IllegalStateException(e.getMessage());
    }

    if (audioRecord != null) {
        audioRecord.release();
        audioRecord = null;
    }
    status = Status.STATUS_NO_READY;
}

/**
 * 将音频信息写入文件
 * @param listener 音频流的监听
 */
private void writeDataToFile(RecordStreamListener listener) {
    // new一个byte数组用来存一些字节数据, 大小为缓冲区大小
    byte[] audiodata = new byte[bufferSizeInBytes];

    FileOutputStream fos = null;
    int readSize = 0;
    try {
        String currentFileName = fileName;
        if (status == Status.STATUS_PAUSE) {
            //假如是暂停录音 将文件名后面加个数字, 防止重名文件内容被覆盖
            currentFileName += fileName.size();
        }
        fileName.add(currentFileName);
        File file = new File(FileUtils.getPcmFileAbsolutePath(currentFileName));
        if (file.exists()) {
            file.delete();
        }
        fos = new FileOutputStream(file); // 建立一个可存取字节的文件
    } catch (IllegalStateException e) {
        Log.e("AudioRecorder", e.getMessage());
        throw new IllegalStateException(e.getMessage());
    } catch (FileNotFoundException e) {
        Log.e("AudioRecorder", e.getMessage());
    }
    //将录音状态设置成正在录音状态
    status = Status.STATUS_START;
    while (status == Status.STATUS_START) {
```



```
        readsize = audioRecord.read(audiodata, 0, bufferSizeInBytes);
        if (AudioRecord.ERROR_INVALID_OPERATION != readsize && fos != null) {
            try {
                fos.write(audiodata);
                if (listener != null) {
                    //用于拓展业务
                    listener.recordOfByte(audiodata, 0, audiodata.length);
                }
            } catch (IOException e) {
                Log.e("AudioRecorder", e.getMessage());
            }
        }
    }
    try {
        if (fos != null) {
            fos.close();// 关闭写入流
        }
    } catch (IOException e) {
        Log.e("AudioRecorder", e.getMessage());
    }
}
```

AudioRecorder 录音声音数据从音频硬件中被读出，编码格式为 PCM格式，但 PCM语音数据，如果保存成音频文件，是不能够被播放器播放的，所以必须先写代码实现数据编码以及压缩。下面实现 PCM 语音数据转为 WAV 文件。



```
/**
 * 将一个pcm文件转化为wav文件
 * @param pcmPath      pcm文件路径
 * @param destinationPath 目标文件路径(wav)
 * @param deletePcmFile 是否删除源文件
 * @return
 */
public static boolean makePCMFileToWAVFile(String pcmPath, String destinationPath, boolean deletePcmFile) {
    byte buffer[] = null;
    int TOTAL_SIZE = 0;
    File file = new File(pcmPath);
    if (!file.exists()) {
        return false;
    }
    TOTAL_SIZE = (int) file.length();
    // 填入参数, 比特率等等. 这里用的是16位单声道 8000 hz
    WaveHeader header = new WaveHeader();
    // 长度字段 = 内容的大小 (TOTAL_SIZE) +
    // 头部字段的大小(不包括前面4字节的标识符RIFF以及fileLength本身的4字节)
    header.fileLength = TOTAL_SIZE + (44 - 8);
    header.FmtHdrLeth = 16;
    header.BitsPerSample = 16;
    header.Channels = 2;
    header.FormatTag = 0x0001;
    header.SamplesPerSec = 8000;
    header.BlockAlign = (short) (header.Channels * header.BitsPerSample / 8);
    header.AvgBytesPerSec = header.BlockAlign * header.SamplesPerSec;
    header.DataHdrLeth = TOTAL_SIZE;

    byte[] h = null;
    try {
        h = header.getHeader();
    } catch (IOException e1) {
        Log.e("PcmToWav", e1.getMessage());
        return false;
    }

    if (h.length != 44) // WAV标准, 头部应该是44字节, 如果不是44个字节则不进行转换文件
        return false;

    // 先删除目标文件
    File destfile = new File(destinationPath);
    if (destfile.exists())
        destfile.delete();


    // 合成的pcm文件的数据, 写到目标文件
    try {
        buffer = new byte[1024 * 4]; // Length of All Files, Total Size
        InputStream inStream = null;
        OutputStream ouStream = null;

        ouStream = new BufferedOutputStream(new FileOutputStream(
            destinationPath));
        ouStream.write(h, 0, h.length);
        inStream = new BufferedInputStream(new FileInputStream(file));
        int size = inStream.read(buffer);
        while (size != -1) {
            ouStream.write(buffer);
        }
    } catch (IOException e) {
        Log.e("PcmToWav", e.getMessage());
        return false;
    }
    return true;
}
```



```
        size = inStream.read(buffer);
    }
    inStream.close();
    ouStream.close();
} catch (FileNotFoundException e) {
    Log.e("PcmToWav", e.getMessage());
    return false;
} catch (IOException ioe) {
    Log.e("PcmToWav", ioe.getMessage());
    return false;
}
}
if (deletePcmFile) {
    file.delete();
}
Log.i("PcmToWav", "makePCMFileToWAVFile success!" + new SimpleDateFormat("yyyy-
return true;
}
```

总结：AudioRecorder 录音相比较 MediaRecorder 使用起来会麻烦一些，但优点也是显而易见的，AudioRecorder 录音时直接操纵硬件获取音频流数据，该过程是实时处理，可以用代码实现各种音频的封装，同时也可实现暂停功能，关于实现暂停录音功能今天在这里就不赘述了，推荐大家阅读 imhxl 博主的分享 <http://blog.csdn.net/imhxl/article/details/52190451> (<http://blog.csdn.net/imhxl/article/details/52190451>)。

 Android (/nb/14327376)

[举报文章](#) © 著作权归作者所有



DreamFish (/u/4ae79ad13c99) ♂

写了 32817 字，被 5 人关注，获得了 60 个喜欢

(/u/4ae79ad13c99)

+ 关注

Most of you are familiar with the virtues of a programmer. There are three, of course: laziness, impatience,...

小礼物走一走，来简书关注我

赞赏支持

♥ 喜欢 (/sign_in?utm_source=desktop&utm_medium=not-signed-in-like-button) | :



更多分享


(<http://cwb.assets.jianshu.io/notes/images/14589114>)



(/apps/download?utm_source=nbc)


iOS开发系列--音频播放、录音、视频播放、拍照、视频录制 (/p/bea68f74f3...

原文链接<http://www.cnblogs.com/kenshincui/p/4186022.html> 音频在iOS中音频播放从形式上可以分为音效播放和音乐播放。前者主要指的是一些短音频播放，通常作为点缀音频，对于这类音频不需要进行进度、循环等控制。后者指的是一些较长的音...

 Hyman0819 (/u/ba8eafb0d9dd?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)


调研linux库alsa (/p/dcd8ee1a2bc9?utm_campaign=maleskine&utm_co...

一．声音参数基本概念：声音是连续模拟量，计算机将它离散化之后用数字表示，就有了以下几个名词术语。样本长度(sample)：样本是记录音频数据最基本的单位，计算机对每个通道采样量化时数字比特位数，常见的有8位和16位。通道数(channel)：该参数为1表示单声道，2则...

 cs1001 (/u/ab26c99e513e?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

Android 录音MediaRecorder到AudioRecord (/p/0e76bbb0c07c?utm_ca...

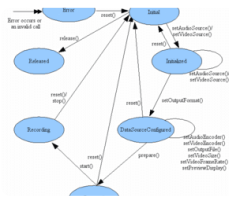
研究录音是源于即时通讯的项目。写出一个即时通讯很简单，但是写好一个即时通讯就不是一件容易的事，比如聊天中语音的加入。接下来就来描述一下自己对语音的见解和处理方式。首先写到语音，当然首当其冲的是运用到网上百分之八九十的处理方案MediaRecorder,这个也是我首先用到的...


 易瑞 (/u/03d216b3ee97?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/d7c40d796166?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

Android音视频录制与播放功能简述 (/p/d7c40d79616...

安卓平台和声音录制与播放相关的主要是4个类：MediaRecorder，MediaPlayer，SoundPool,AudioRecord和AudioTrack。MediaRecorder可以录制视频和音频到文件 MediaPlayer可以播放视频和音频文件 SoundP...




 liujc (/u/0633b9f8256b?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/af7787b409a2?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

Android AudioRecord和AudioTrack介绍 (/p/af7787...

Android音频收集和播放(一) 一、文章说明 这是自己第一次通过写文章的方式来记录在开发中的一些心得,在这里也希望这是一个好的开始并一直坚持下去,同时更是希望能帮助到有需要的开发者。 这篇文章主要讲述的是Android中使用AudioRecord类和AudioTrack...




 熊熊熊孩子 (/u/daba5093f75e?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/a000869a7d45?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

Sunny就像小太阳，浑身充满正能量 (/p/a000869a7d4...


荣格曾说：“你连想改变别人的念头都不要有。作为老师，要学习像太阳一样，只是发出光和热，每个人对阳光的反应有不同。有人觉得刺眼，有人觉得温暖，有人甚至会躲开。种子破土而出前，没有任何迹象，那是因为还没到那个点。永远相信每个人都是自己的拯救者。” 每每看到这句话，心里都会感慨万...



 陈晓依 (/u/daa03de01efd?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

HTML元素 (/p/bbf3edf52f13?utm_campaign=maleskine&utm_content=...

HTML 文档是由 HTML 元素定义的。HTML 元素 HTML 元素指的是从开始标签 (start tag) 到结束标签 (end tag) 的所有代码。注释：开始标签常被称为开放标签 (opening tag)，结束标签常称为闭合标签 (closing tag)。HTML ...

 soitif (/u/8bb147e23edc?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)


(/p/de5b29d57415?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

喝茶、送礼、投资首选智慧普洱--道济禅茶 (/p/de5b29d57415?utm_camp...



道济禅茶是由道济禅修院监制、道济(北京)文化发展有限公司推出的一款珍贵普洱茶。其原料选自云南普洱茶区的千年古茶园，经人工采摘制作而成，目前有古树生茶、白云号、净雨红茶、净雨黄茶、生态茶等多个经典系列，是当下最具收藏和投资价值的一款普洱茶。道济禅茶还具有“禅、身、心、友、礼”...




 山白传媒 (/u/6a9b0850d631?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/1f7356b130e2?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)




嫌疑人X的献身 (/p/1f7356b130e2?utm_campaign=m...

东野圭吾的推理小说，仿佛为我打开了一个新领域的大门。直白，朴素，精炼，语言平实，情节却给出巨大的张力，布局合理，环环相扣，神奇的人物构造，意外又细思极恐的结局，略有生硬，看完却让人觉得不失为一部好的小说。
“一个人只要好好活着 就足以拯救某人” 表面看上去冷酷、严谨、自律...

 Monica爱笑 (/u/1dcb5409d689?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

悲剧人生 (/p/de363ad7664f?utm_campaign=maleskine&utm_content=n...

生活就该如此折腾 不为什么 不为他人 不为眼前实物所动摇 人生 就要过活的有感觉！地球只有一个 你扮演一个人 你要演什么在于你的思想 本身如同一个机器人，机器什么时候有自己的思想 活出自己 人生就缺这一点 有人做到了 有人还在为他人扮演角色 就像你为他办事 他给你钱 人生到...

 曾少恭 (/u/3c3d835e9bc3?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

