MediaMuxer API 简介

MediaExtractor 是什么?

顾名思义，MediaExtractor 可以从数据源中**提取**经过编码的媒体数据。MediaExtractor 不仅可以解析本地媒体文件，还可以解析网络媒体资源。

MediaMuxer 是什么?

同样，名字已经说明了一切。MediaMuxer 可以将多个流混合**封装**起来，支持 MP4、Webm 和 3GP 文件作为输出，而且从 Android N 开始，已经支持在 MP4 中混合 B 帧了。

这次的任务是什么?

这次的任务是从一个 MP4 文件中只提取视频数据，并封装为一个新的 MP4 文件。外在表现就是将一个**有声视频**，转换为一个**无声视频**。

MediaExtractor 的作用就是将音频和视频分离。

主要是以下几个步骤：

1、创建实例

```
MediaExtractor mediaExtractor = new MediaExtractor();
```

2、设置数据源

```
mediaExtractor.setDataSource(path);
```

3、获取数据源的轨道数，切换到想要的轨道

```
// 轨道索引
int videoIndex = -1;
// 视频轨道格式信息
MediaFormat mediaFormat = null;
// 数据源的轨道数
int trackCount = mediaExtractor.getTrackCount();
for (int i = 0; i < trackCount; i++) {
    MediaFormat format = mediaExtractor.getTrackFormat(i);
    String mimeType = format.getString(MediaFormat.KEY_MIME);
    if (mimeType.startsWith("video/")) {
        videoIndex = i;
        mediaFormat = format;
        break;
    }
}
// 切换到想要的轨道
mediaExtractor.selectTrack(videoIndex);
```

4、对所需轨道数据循环读取读取每帧，进行处理

```
while (true) {
    // 将样本数据存储到字节缓存区
    int readSampleSize = mediaExtractor.readSampleData(byteBuffer, 0);
    // 如果没有可获取的样本，退出循环
    if (readSampleSize < 0) {
        mediaExtractor.unselectTrack(videoIndex);
        break;
    }
    ...
    ...
    // 读取下一帧数据
    mediaExtractor.advance();
}
```

5、完成后释放资源

```
mediaExtractor.release();
```

MediaMuxer

MediaMuxer 的作用是生成音频或视频文件；还可以把音频与视频混合成一个音视频文件。

主要是以下几个步骤:

1、创建实例

```
MediaMuxermediaMuxer = new MediaMuxer(path, format);
```

path: 输出文件的名称；format: 输出文件的格式，当前只支持 MP4 格式。

2、将音频轨或视频轨添加到 MediaMuxer，返回新的轨道

```
int trackIndex = mediaMuxer.addTrack(videoFormat);
```

3、开始合成

```
mediaMuxer.start();
```

4、循环将音频轨或视频轨的数据写到文件

```
 while (true) {
     // 将样本数据存储到字节缓存区
     int readSampleSize = mediaExtractor.readSampleData(byteBuffer, 0);
     // 如果没有可获取的样本，退出循环
     if (readSampleSize < 0) {
         mediaExtractor.unselectTrack(videoIndex);
         break;
     }
     bufferInfo.size = readSampleSize;
     bufferInfo.flags = mediaExtractor.getSampleFlags();
     bufferInfo.offset = 0;
     bufferInfo.presentationTimeUs = mediaExtractor.getSampleTime();
```

```
        mediaMuxer.writeSampleData(trackIndex, byteBuffer, bufferInfo);
        // 读取下一帧数据
        mediaExtractor.advance();
    }
```

5、完成后释放资源

```
mediaMuxer.stop();
mediaMuxer.release();
```


实例

从 MP4 文件中分离出视频生成无声视频文件。

```
/**
  * 分离视频的视频轨，输入视频 input.mp4，输出 output_video.mp4
  */
private void extractVideo() {
    MediaExtractor mediaExtractor = new MediaExtractor();
    MediaMuxer mediaMuxer = null;
    File fileDir = FileUtil.getExternalAssetsDir(this);
    try {
        // 设置视频源
        mediaExtractor.setDataSource(new File(fileDir,
VIDEO_SOURCE).getAbsolutePath());
        // 轨道索引
        int videoIndex = -1;
        // 视频轨道格式信息
        MediaFormat mediaFormat = null;
        // 数据源的轨道数
        int trackCount = mediaExtractor.getTrackCount();
        for (int i = 0; i < trackCount; i++) {
            MediaFormat format = mediaExtractor.getTrackFormat(i);
            String mimeType = format.getString(MediaFormat.KEY_MIME);
            if (mimeType.startsWith("video/")) {
                videoIndex = i;
                mediaFormat = format;
                break;
            }
        }
        // 切换到想要的轨道
        mediaExtractor.selectTrack(videoIndex);
        File outFile = new File(FileUtil.getMuxerAndExtractorDir(this),
OUTPUT_VIDEO);
        if (outFile.exists()) {
            outFile.delete();
        }
        mediaMuxer = new MediaMuxer(outFile.getAbsolutePath(),
MediaMuxer.OutputFormat.MUXER_OUTPUT_MPEG_4);
        // 将视频轨添加到 MediaMuxer，返回新的轨道
        int trackIndex = mediaMuxer.addTrack(mediaFormat);
        ByteBuffer byteBuffer =
ByteBuffer.allocate(mediaFormat.getInteger(MediaFormat.KEY_MAX_INPUT_SIZE));
        MediaCodec.BufferInfo bufferInfo = new MediaCodec.BufferInfo();
```

```
            mediaMuxer.start();
            while (true) {
                // 将样本数据存储到字节缓存区
                int readSampleSize = mediaExtractor.readSampleData(byteBuffer, 0);
                // 如果没有可获取的样本，退出循环
                if (readSampleSize < 0) {
                    mediaExtractor.unselectTrack(videoIndex);
                    break;
                }
                bufferInfo.size = readSampleSize;
                bufferInfo.flags = mediaExtractor.getSampleFlags();
                bufferInfo.offset = 0;
                bufferInfo.presentationTimeUs = mediaExtractor.getSampleTime();
                mediaMuxer.writeSampleData(trackIndex, byteBuffer, bufferInfo);
                // 读取下一帧数据
                mediaExtractor.advance();
            }
            Toast.makeText(this, "分离视频完成", Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (mediaMuxer != null) {
                mediaMuxer.stop();
                mediaMuxer.release();
            }
            mediaExtractor.release();
        }
    }
```

分离音频、合成音视频