# 项目简述

该播放器的控制流程参考Android MediaPlayer的方式，其目的是：

1. 借鉴Android MediaPlayer现有的架构，站在巨人的肩膀上进行开发；
2. 当学习完该Player的开发，也方便以后转到Android Player的开发。

## 实现的功能

该播放器实现以下的功能：

1. 音视频播放；
2. 直播推流；
3. 录制，包含桌面录制和摄像头录制；
4. 视频剪辑，转码、缩放、添加水印功能等。

## 主要的接口类

1. MediaPlayer 播放功能
2. MediaPusher 推流
3. MediaRecoder 录制
4. MediaEdit 剪辑

## 涉及的技术

1. FFmpeg开源库；
2. QT；
3. C++11新特性；
4. RTMP流媒体协议；
5. RTSP流媒体协议；
6. H264视频格式的解析；
7. AAC/MP3的音频解析；
8. FLV/MP4/TS/MKV封装容器的解析。

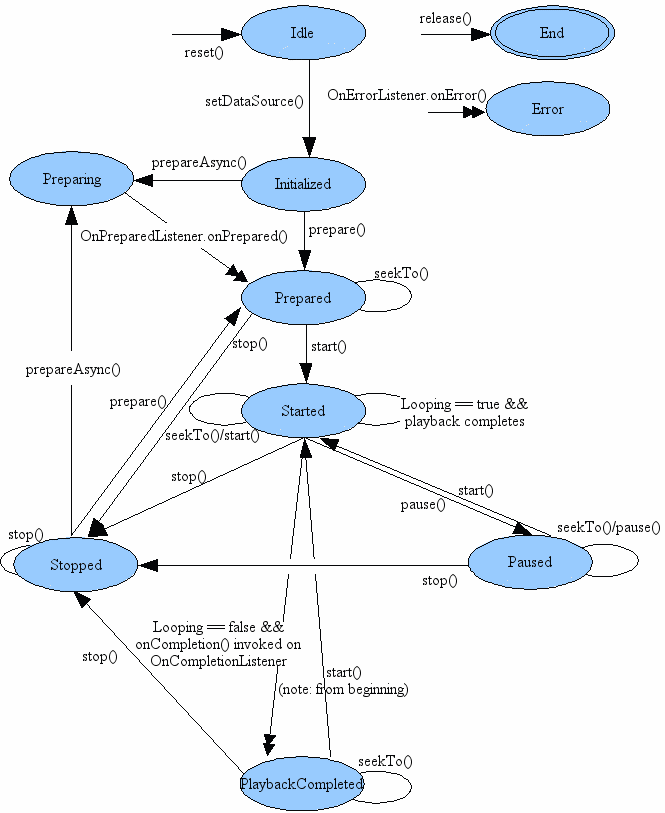
# 播放器开发

## MediaPlayer的状态图

MediaPlayer类是用于控制playback的视频/音频文件及流，其状态有：

1. Idle 状态
2. End 状态
3. Error 状态
4. Initialized状态
5. Prepared状态
6. Preparing状态
7. Started状态
8. Paused状态
9. Stopped状态
10. PlaybackCompleted状态

其状态机如图所示：



椭圆代表MediaPlayer驻留的状态

弧代表playback控制且驱动MediaPlayer状态进行过渡。

有两种类型的弧，一种是用单箭头表示的是同步方法调用，另一种有两个箭头代表异步方法调用。

从状态图中，我们能看到MediaPlayer有如下一些状态：

**Idle 状态及End 状态**

当MediaPlayer用new或者reset方法调用后，就被创建了，这个时候它是处于Idle（就绪状态），delete或者release后就属于End（结束状态），在这两种状态之间，就是MediaPlayer的生命周期。

**Error 状态**

1. 可能引起Error的原因：
2. 调用设置出现流程错误，比如还没有设置播放资源setDataResource就调用seekTo；
3. 播放的时候出现格式不支持；
4. 播放网络流时出现网络端口等。
5. 一旦MediaPlayer是End（结束）状态时，它将不再被使用

**Initialized状态**

当调用setDataSource()时，将MediaPlayer的Idle状态变成Initialized（初始化）状态，如果setDataSource在非Idle状态时调用，则将会报告异常。

**Prepared状态**

在playback开始后，接下来MediaPlayer开始首次进入Prepared（准备好）状态，有两种方式，一种是同步，一种是异步。一旦调用prepare()（同步方法）将传递MediaPlayer的Initialized状态变成Preparing，当准备工作完成后再返回。如果使用prepareAsync()（异步方法），调用直接返回，但实际的结果通过异步回调返回，调用者需要在收到异步回调后再进行下步的MediaPlayer操作。

**Started状态**

一旦Prepared状态后，属性如音频的音量已经设置。在playback开始之前，必须调用start方法成功返回，MediaPlayer的状态开始由Prepared状态转为Started状态。如果MediaPlayer已经Started了，再调用start方法，是没有任何作用的。

**Paused状态**

Playback可以是paused（暂停）和stopped（停止），且当前的playback位置可以被调整，当pause()方法时，playback就变成停止，MediaPlayer的状态由Started变成Paused。

当start()方法从paused状态恢复回来时，playback恢复之前暂停时的位置，接着开始播放，这时状态又变成started状态。

如果MediaPlayer已经paused了，再调用pause方法，是没有任何作用的，状态还是paused状态。

**Stopped状态**

调用调用stop()方法时，playback无论之前在started、paused、prepared或playbackCompleted状态都将进入stopped状态，一旦stopped状态，playback将不能stared，直到重新prepare()或prepareAsync()后，处于Prepared状态时才可start。

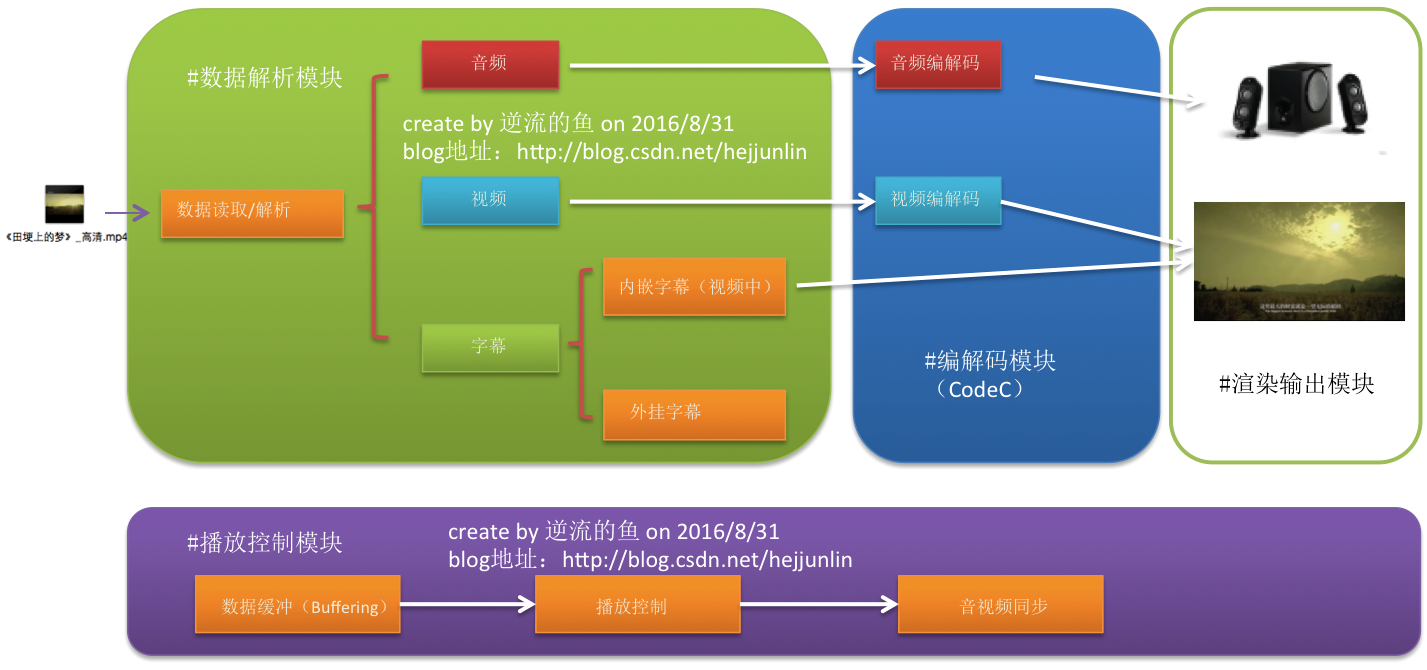
Playback位置能通过seekTo(int)方法调整，这个是一个异步方法，调用者可以注册OnSeekComplete方法捕获seekTo的结果。seekTo可能要花一小会时间才能完成，尤其是音视频被缓冲的流。

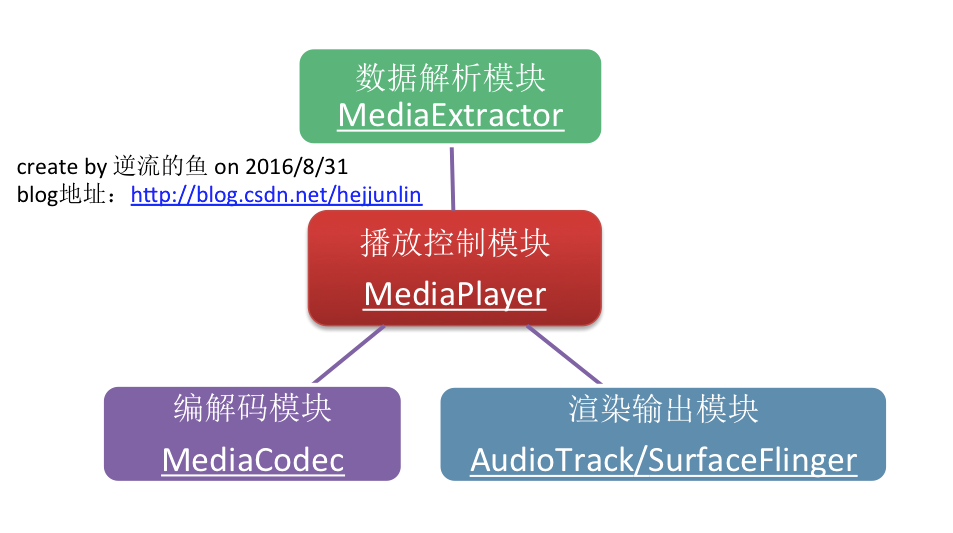
**PlaybackCompleted状态**

当前playback的位置可以通过getCurrentPosition()方法检索到，可以通过getCurrentPosition更新进度条。

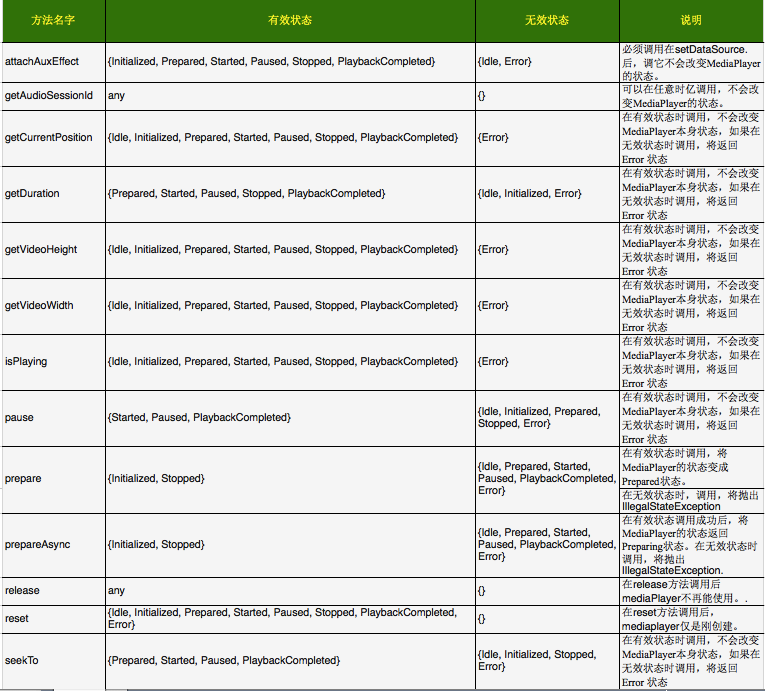
当playback到达流的末尾时，playback完成。如果循环模式在MediaPlayer中事先调用setLooping为true的话，MediaPlayer应当保持Started状态，如果setLooping被设置为false，MediaPlayer的调用者如果注册过setOnComplettionListenr，player内部将回调OnCompletetion，MediaPlayer开始进入PlaybackCompleted（播放完成）状态。在PlaybackCompleted状态时，调用start启动playback从头开始播放音视频源。

## MediaPlayer的框架





MediaPlayer方法对应有效状态及无效状态



| **方法** | **说明** |
| --- | --- |
| setDataSource | 设置多媒体数据来源(位置) |
| setVideoSurfaceTexture | 设置用SurfaceHolder来显示多媒体 |
| prepare | 准备（同步） |
| prepareAsync | 准备（异步） |
| start | 开始播放 |
| stop | 停止播放 |
| reset | 重置MediaPlayer对象为刚刚创建的状态 |
| getCurrentPosition | 得到当前播放位置 |
| getDuration | 得到文件的时间 |
| isPlaying | 是否正在播放 |
| pause | 暂停 |
| seekTo | 指定播放的位置（以毫秒为单位的时间） |
| setLooping | 设置是否循环播放 |

## MediaPlayerService



# 参考文档

1. Android Multimedia框架总结 <http://blog.csdn.net/hejjunlin/article/details/52349221>