

金山云Android播放内核接口文档

文档版本 v1.0.2.151217

[金山云Android播放内核接口文档](#)

[金山云概述](#)

[金山云Android SDK](#)

[使用方法](#)

[配置项目](#)

[系统权限](#)

[KSYMediaPlayer](#)

[构造函数](#)

[功能接口](#)

[设置回调接口](#)

[预定义的常量](#)

[Info类型](#)

[Error类型](#)

[SDK使用示例](#)

[反馈与建议](#)

金山云概述

金山云致力于提供专家级的云服务解决方案，金山云播放器SDK为金山云视频转码、直播等云端处理方案提供全平台的播放支持。播放器SDK支持的平台包括：

- Desktop APP
 - Windows (x86/ARM)
 - Mac
 - Linux
- Mobile
 - Android (x86/ARM)
 - iOS (ARM32/64)
- Cross Platform
 - Adobe Flash Player
 - Web Browser Plugin (ActiveX/NPAPI)

金山云Android SDK

金山云Android SDK提供了在Android平台上音视频播放的解决方案，支持主流的网络协议，文件容器格式，音视频编码格式等，并且集成有H.265/HEVC解码器，在主流Android手机能非常流畅的播放1080P视频，并有如下特性：

- 完美支持点播、直播。直播带有追功能，让用户享受最实时的直播视频。
- 支持**多实例**功能。用户可同时创建多个播放器对象，同时播放多个视频。

SDK支持的网络协议、文件格式、音视频格式如下所示：

- 网络协议：http, rtmp
- 容器格式：flv, hls, ts, mp4, mov
- 视频标准：H.264/AVC, H.265/HEVC
- 音频格式：aac, mp3

本SDK包括一个Jar包和一个so库。其中，Jar包主要包含了一个基于Android系统播放器MediaPlayer实现的播放器KSYMediaPlayer，供外界调用。而so库则包含了底层网络协议，文件格式解析及相应的解码库的实现。

下面将具体介绍如何使用仟壹Android SDK。

使用方法

配置项目

使用金山云Android SDK需引入相应资源，其中so库只支持armv7a汇编指令集

- libs/armeabi-v7a/libksyplayer.so
- libs/libksyplayer.jar
- libs/libksystat.jar

其中，jar包包名是：

- com.ksyun.media.player

如果开发者需要混淆代码，则必须在混淆的配置文件中添加如下配置，防止jar包被混淆，导致播放出错

- -libraryjars libs/libksyplayer.jar
- -libraryjars libs/libksystat.jar

系统权限

在您开始开发前，需要在您AndroidManifest.xml里添加如下权限，如若没有添加相应的权限，则会出现播放错误。

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_ST
ATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STAT
E" />
<uses-permission android:name="android.permission.READ_PHONE_STAT
E"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_ST
ORAGE" />
```

KSYMediaPlayer

以Android的系统播放器MediaPlayer为蓝本，实现了在接口命名、功能和播放状态都与MediaPlayer保持一致的播放类KSYMediaPlayer，只需创建KSYMediaPlayer的对象，设置播放地址即可开始播放视频。下面将简要介绍类KSYMediaPlayer。

构造函数

KSYMediaPlayer对象的创建采用了Builder模式，需要设置更多的参数用于SDK认证，具体如下所示：

```
KSYMediaPlayer ksyMediaPlayer;
ksyMediaPlayer = new KSYMediaPlayer.Builder(mContext).setAppId("KSY
AppId").setAccessKey("KSYAccessKey").setTimeSec("123456789").setSec
retKeySign("KSYSecretKeySign").build();
```

函数/类的的含义如下：

函数/类	功能
Builder	KSYMediaPlayer内部静态类
setAppId	设置开发者标识，标识由开放平台分发给开发者
setAccessKey	设置AccessKey，与SecretKey对应，由开放平台针对appid分 配
setTimeSec	设置时间戳，单位为秒(s)
setSecretKeySign	设置加密后的SecretKey，加密方式如后所示
enableKSYStatModule	开启/关闭数据收集模块
build	创建KSYMediaPlayer对象并返回

SecretKey的加密方式：

- SecretKeySign=md5(SecretKey + TimeSec)

功能接口

static String getVersion()

参数：无

功能：获取SDK版本信息

返回值：SDK版本

void setDataSource(String path)

参数：path 视频播放地址

功能：设置视频播放地址给播放器，播放器进入Initialized状态

返回值：无

void setDataSource(Context context, Uri uri)

参数：context Application的上下文内容

uri 视频播放地址

功能：设置视频播放地址，播放器进入Initialized状态

返回值：无

void setDataSource(Context context, Uri uri, Map<String, String> headers)

参数：context Application的上下文内容

uri 视频播放地址

headers 与获取数据的请求一同发送的头

功能：设置视频播放地址和请求的头信息，播放器进入Initialized状态

返回值：无

void setDataSource(FileDescriptor fd)

参数：fd 视频文件的FileDescriptor

功能：设置需要播放的视频源，播放器进入Initialized状态

返回值：无

void prepareAsync()

参数：无

功能：与prepare接口功能相同，不过此接口是异步的，播放器进入Preparing状态

返回值：无

void start()

参数：无

功能：在播放器发出prepared回调之后调用，开始播放视频，播放器进入Started状态

返回值：无

void pause()

参数：无

功能: 暂停视频播放, 播放器进入Paused状态

返回值: 无

`void stop()`

参数: 无

功能: 停止视频播放, 播放器进入Stopped状态

返回值: 无

`long getCurrentPosition()`

参数: 无

功能: 获取音视频的播放进度

返回值: 处于Playing状态时, 返回真实播放进度, 否则为0。单位为微妙 (us)

`void seekTo(long msec)`

参数: seek的目标位置, 单位为微妙 (us)

功能: 快进、快退

返回值: 无

`long getDuration()`

参数: 无

功能: 获取音视频的时长

返回值: Prepared之后调用会返回音视频的时长, 否则为0。单位微妙 (us)

`int getVideoWidth()`

参数: 无

功能: 获取视频宽度

返回值: Prepared之后调用返回视频宽度, 否则为0.

`int getVideoHeight()`

参数: 无

功能: 获取视频高度

返回值: Prepared之后调用返回视频高度, 否则为0.

`boolean isPlaying()`

参数: 无

功能: 判断播放器是否在Playing状态

返回值: 播放器处于Playing时返回true, 否则为false

`void release()`

参数: 无

功能: 释放播放器资源

返回值: 无

`void reset()`

参数: 无

功能: 重置播放器

返回值: 无

`void setSurface(Surface surface)`

参数: Surface

功能: 设置Surface, 用于视频渲染

返回值: 无

`void setDisplay(SurfaceHolder sh)`

参数: SurfaceHolder

功能: 设置SurfaceHolder

返回值: 无

`void setVolume(float leftVolume, float rightVolume)`

参数: 目标音频音量

功能: 设置音频音量

返回值: 无

`void setScreenOnWhilePlaying(boolean screenOn)`

参数: screenOn 值为true时, 播放时屏幕保持常亮, 反之则否

功能: 使用SurfaceHolder控制播放期间屏幕是否保持常亮。须调用接口setDisplay设置SurfaceHolder, 此接口才有效

返回值: 无

`void setWakeMode(Context context, int mode)`

参数: context Application的上下文内容

mode 设想的Wake模式

功能: 设置此播放器存活期间的电源行为模式

返回值: 无

`ITrackInfo[] getTrackInfo()`

参数: 无

功能: 获取视频中音视频的轨道信息

返回值: 含有轨道信息的数组

`void setLooping(boolean looping)`

参数: looping 是否循环播放

功能: 设置播放器循环/非循环播放

返回值: 无

`void isLooping()`

参数: 无

功能: 判断播放器是否处于循环播放状态

返回值: true 播放器处于循环播放状态

false 播放器不处于循环播放状态

`long getDownloadContentSize()`

参数: 无

功能: 获取播放至今已下载的数据大小

返回值: 播放至今已下载的数据大小, 单位: KB

`int bufferEmptyCount()`

参数: 无

功能：获取开播后播放器缓存的次数

返回值：播放器缓存次数

`float bufferEmptyDuration()`

参数：无

功能：获取开播后播放器处于缓存状态的总时长

返回值：播放器处于缓存状态的总时长，单位为秒(s)

`String getServerAddress()`

参数：无

功能：获取播放rtmp/http视频服务器IP地址。播放器进入PREPARED状态之后才可调用

返回值：播放rtmp/http视频服务器IP地址，否则为NULL

`void setTimeout(int timeout)`

参数：超时阈值，单位：毫秒(ms)

功能：设置超时阈值

返回值：无

`void getCurrentFrame(Bitmap bitmap)`

参数：位图

功能：获取视频的一帧截图，存储与传入的bitmap

返回值：无

`void setBufferSize(int size)`

参数：缓存阈值

功能：设置缓存阈值

返回值：无

`boolean setCachedDir(String cachedPath)`

参数：缓存路径，必须为文件夹路径

功能：设置视频缓存路径，必须在prepare之前设置

返回值：true 设置成功

false 设置失败

`boolean clearCachedFiles(String cachedPath)`

参数：缓存视频的路径，必须为文件夹路径

功能：循环清除文件夹下的缓存文件

返回值：true 成功清除缓存文件

false 清除缓存文件失败

设置回调接口

在接口类KSYMediaPlayer中已定义若干回调interface，开发者只需实现相应的interface并设置到KSYMediaPlayer即可收到相应的回调信息。

设置回调接口的函数有：

```
void setOnInfoListener(OnInfoListener listener)
```

参数: OnInfoListener

功能: 设置Info监听器, 播放器可通过此回调接口将消息通知开发者

返回值: 无

```
void setOnPreparedListener(OnPreparedListener listener)
```

参数: OnPreparedListener

功能: 设置Prepared状态的监听器, 在调用prepare()/prepareAsync()之后, 正常完成解析后会通过此监听器通知外界。

返回值: 无

```
void setOnCompletionListener(OnCompletionListener listener)
```

参数: OnCompletionListener

功能: 设置Completion的监听器, 在视频播放完成后会发出此回调

返回值: 无

```
void setOnBufferingUpdateListener(OnBufferingUpdateListener listener)
```

参数: OnBufferingUpdateListener

功能: 设置Buffering的监听器, 当播放器在Buffering时会发出此回调, 通知外界Buffering的进度

返回值: 无

```
void setOnSeekCompleteListener(OnSeekCompleteListener listener)
```

参数: OnSeekCompleteListener

功能: 设置Seek Complete的监听器, Seek操作完成后会有此回调

返回值: 无

```
void setOnErrorListener(OnErrorListener listener)
```

参数: OnErrorListener

功能: 设置Error监听器, 当播放器遇到error时, 会发出此回调并送出error code

返回值: 无

```
void setOnVideoSizeChangedListener(OnVideoSizeChangedListener listener)
```

参数: OnVideoSizeChangedListener

功能: 设置VideoSizeChanged的监听器, 当视频的宽度或高度发生变化时会发出次回调, 通知外界视频的最新宽度和高度

预定义的常量

在KSYMediaPlayer中定义了Info和Error类型常量

Info类型

OnInfoListener回调时的消息类型定义如下:

变量名	数值	含义
MEDIA_INFO_UNKNOWN	1	未指定的播放器信息
MEDIA_INFO_VIDEO_RENDERING_START	3	视频开始渲染
MEDIA_INFO_VIDEO_TRACK_LAGGING	700	视频复杂，解码器效率不足
MEDIA_INFO_BUFFERING_START	701	播放器开始缓存数据
MEDIA_INFO_BUFFERING_END	702	播放器缓存完毕
MEDIA_INFO_BAD_INTERLEAVING	800	视频封装有误
MEDIA_INFO_NOT_SEEKABLE	801	此视频不能seek
MEDIA_INFO_METADATA_UPDATE	802	已获得新的元数据
MEDIA_INFO_UNSUPPORTED_SUBTITLE	901	不支持此字幕
MEDIA_INFO_SUBTITLE_TIMED_OUT	902	读取字幕超时
MEDIA_INFO_VIDEO_ROTATION_CHANGED	10001	视频方向改变
MEDIA_INFO_AUDIO_RENDERING_START	10002	音频开始播放

Error类型

OnErrorListener回调时错误类型定义：

变量名	数值	含义
MEDIA_ERROR_UNKNOWN	1	未指定的播放器错误
MEDIA_ERROR_SERVER_DIED	100	多媒体服务器出错
MEDIA_ERROR_NOT_VALID_FOR_PROGRESSIVE_PLAYBACK	200	流媒体封装格式并不支持渐进播放
MEDIA_ERROR_IO	-1004	文件或网络相关操作错误
MEDIA_ERROR_MALFORMED	-1007	码流实际编码标准与文件描述不一致

MEDIA_ERROR_UNSUPPORTED	-1010	播放器不支持相应编码格式
MEDIA_ERROR_TIMED_OUT	-110	操作超时
MEDIA_ERROR_AUTH_FAILED	-1040	金山云SDK鉴权失败

其中，如果SDK鉴权失败，则会通过OnErrorListener告知开发者鉴权失败返回的errno，其具体含义如下所示：

errno	含义
6	参数不全
15	appid无效
16	accessKey无效
18	验证失败

SDK使用示例

1. 布局文件

在布局文件中定义SurfaceView，用于视频渲染

```
<SurfaceView
    android:id="@+id/player_surface"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_centerInParent="true"/>
```

2. 创建 SurfaceHolder.Callback

Surface的创建、变更、销毁都会调用此回调接口的对应方法，Surface变更时需将Surface设置到播放器中，用于视频渲染。其中ksyMediaPlayer的定义请见后面代码。

```
SurfaceHolder.Callback mSurfaceCallback = new Callback()
{
    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
        if(qyMediaPlayer != null) {
            final Surface newSurface = holder.getSurface();
            if (mSurface != newSurface) {
                mSurface = newSurface;
                ksyMediaPlayer.setSurface(mSurface);
            }
        }
    }

    @Override
    public void surfaceCreated(SurfaceHolder holder) {}

    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        if(ksyMediaPlayer != null) {
            mSurface = null;
        }
    }
}
```

3. 初始化SurfaceView和SurfaceHolder

```
SurfaceView mVideoSurfaceView = (SurfaceView) findViewById(R.id.player_surface);
SurfaceHolder mSurfaceHolder = mVideoSurfaceView.getHolder();
mSurfaceHolder.addCallback(mSurfaceCallback);
```

4. 创建播放器对象：

```
KSYMediaPlayer ksyMediaPlayer
String timeSec = String.valueOf(System.currentTimeMillis() / 1000);
// 此参数的单位为秒
String secretKeySign = md5(SecretKey + timeSec)
ksyMediaPlayer = new KSYMediaPlayer.Builder(mContext).setAppId("KSY
appId").setAccessKey("KSYAccessKey").setSecretKeySign(secretKeySig
n).setTimeSec(timeSec).build();
```

5. 设置监听器

```
ksyMediaPlayer.setOnInfoListener(mOnInfoListener)
ksyMediaPlayer.setOnBufferingUpdateListener(mOnBufferingUpdateListener);
ksyMediaPlayer.setOnCompletionListener(mOnCompletionListener);
ksyMediaPlayer.setOnPreparedListener(mOnPreparedListener);
ksyMediaPlayer.setOnErrorListener(mOnErrorListener);
ksyMediaPlayer.setOnSeekCompleteListener(mOnSeekCompletedListener);
```

4. 设置播放地址并调用prepare接口

```
try {
    ksyMediaPlayer.setDataSource(playUrl);
    ksyMediaPlayer.prepareAsync();
} catch (IOException e) {
    e.printStackTrace();
}
```

5. 在收到onPrepared回调之后，调用start接口开始播放

```
private IMediaPlayer.OnPreparedListener mOnPreparedListener = new IMediaPlayer.OnPreparedListener() {
    @Override
    public void onPrepared(IMediaPlayer mp) {
        if (ksyMediaPlayer != null)
            ksyMediaPlayer.start();
    }
};
```

反馈与建议

- 邮箱: linsong2@kingsoft.com