

趣看云视频直播点播服务

录播上传SDK Android开发指南

版本：1.0

2016.05



目录

一、	系统介绍及快速使用.....	2
1.1	SDK 的使用前言	2
1.2	SDK 初始化流程	2
1.3	录制视频使用流程.....	2
1.4	录制过程中的可选操作.....	2
1.5	上传文件使用流程.....	2
1.6	上传文件过程中的可选操作.....	3
二、	Andiord 客户端初始工作.....	3
2.1	项目初始配置.....	3
2.2	ClientSdk 类的接口.....	3
2.2.1	初始化函数.....	3
2.2.2	添加消息侦听器.....	4
2.2.3	用户密钥的校验.....	5
2.2.4	文件上传接口.....	5
2.2.5	删除上传文件接口.....	6
2.2.6	暂停所有上传中的文件.....	6
2.2.7	文件提交接口.....	6
2.2.8	文件合并接口.....	7
2.2.9	获取当前正在上传的文件.....	7
2.2.10	获取版本号.....	7
2.3	RecordContext 类的接口.....	7
2.3.1	初始化摄像机和麦克风.....	8
2.3.2	关闭摄像头和麦克风.....	8
2.3.3	开始文件录像.....	9
2.3.4	结束文件录像.....	9
2.3.5	暂停文件录像.....	9
2.3.6	开启或关闭闪光灯.....	10
2.3.7	切换前后摄像头.....	10
2.3.8	开启或者关闭声音.....	10
2.3.9	设置手动聚焦或者自动聚焦.....	11
2.3.10	手动聚焦.....	11
2.3.11	截图.....	11
2.3.12	设置 logo.....	12
三、	DEMO 介绍	12
四、	Demo 的数据库介绍.....	13

一、系统介绍及快速使用

1.1 SDK 的使用前言

- 使用 SDK 前要先调用 `appKeyCheck` 设置 `appkey` 进行验证。
- 由于一次录播时间有可能会很长，所以 `sdk` 对录像的录制进行了分片处理，目前设定为 5 分钟生成一个 `mp4` 的录像文件。
用户将这些 `MP4` 文件上传到阿里云服务器后，可以通知我们的服务器，对这些文件进行文件合并，最终可以在服务器上生成一个完整的视频文件。
Demo 的作用就是在本地建立一个数据库，管理这些文件及是否上传到阿里云服务器的状态。

1.2 SDK 初始化流程

- 1、调用 `void init(Context context, int logLevel)` 来设置全局属性和日志等级。
- 2、调用 `void addListener(Handler msgHandle)` 来设置消息回调，用于接收之后的录像和文件上传消息。
- 3、调用 `void appKeyCheck(String appKey)` 来设置 `appkey` 的验证，如果没有验证则之后的录像和文件上传都会失败。

1.3 录制视频使用流程

- 1、开启摄像头 `boolean startCamera`，里面包括了分辨率、码率的设置，帧率目前是定死为 20 帧每秒。
- 2、开始录制 `void startRecord`
- 3、（可选）设置 logo `int setLogoInfo`
- 4、暂停录制 `void pauseRecord`
- 5、结束录制 `void stopRecord`
- 6、关闭摄像头 `void stopCamera`

1.4 录制过程中的可选操作

切换前后摄像头、设置自动聚焦或者手动聚焦、关闭或开启声音、关闭或者开启闪光灯。

1.5 上传文件使用流程

- 1、上传文件 `void newFileSync`
- 3、处理回调通知
 - 3.1 文件上传到阿里云服务器完成后，会先通知我们的服务器，文件上传完成了。（第

一次 SDK 会自动触发，如果通知失败了需要用户手动触发)

3.2 获取到通知完成的请求后，检查本地其他文件。

3.3 都上传并通知完成后，可以触发文件合并请求。(合并请求 SDK 不会自动触发，需要用户调用接口，或者直接在后台进行文件合并的选择)

1.6 上传文件过程中的可选操作

删除本次上传进度、删除已经上传的文件。检查当前正在上传的文件

二、Andiord 客户端初始工作

2.1 项目初始配置

提供的 jar 文件: \libs 下 qkuploaddk-1.x.x.jar, aliyun-oss-sdk-android-2.2.0.jar, fastjson-1.1.27.jar, okhttp-3.2.0.jar, okio-1.6.0.jar

提供的 so 文件: \jniLibs\armeabi-v7a 下的 libqkffmpeg.so 和 libqkrecord.so

2.2 ClientSdk 类的接口

`import com.qukan.qkrecorduploadsdk.ClientSdk;` 该类全部是静态方法，可以直接调用

2.2.1 初始化函数

类型	描述
函数	<code>public static void init(Context context, int logLevel);</code>
参数	[in]ctx: android.content.Context 对象 [in]logLevel: sdk 的日志等级，即 android.util.Log 的日志等级，DEBUG、INFO、WARN、ERROR 等
返回值	

说明	该函数完成 ClientSdk 的一些全局属性的设置，比如应用程序上下文，日志等级。该函数必须首先被调用，一般在 android 应用程序主 activity 的 onCreate() 函数中调用。
----	---

2.2.2 添加消息侦听器

类型	描述
函数	public static void addListener(Handler msgHandle)
参数	[in] msgHandle: android.os.Handler 对象
返回值	
说明	当 SDK 内部产生一些消息通知的时候，可以通过注册消息接收句柄来接收这些消息并处理。addListener() 方法可以被多个 activity 同时注册，各个 activity 的消息处理函数仅处理自己关心的消息即可。一般在各个 activity 的 onResume() 中调用

消息类型：

```
public class FileSyncApi
{
    public static final int MSG_FILEUPLOAD_TRACE = 100; // 发送文件上传处理的调试信息, 不做任何判断用
    public static final int MSG_FILEUPLOAD_PROGRESS = 101; // 文件上传的进度
    public static final int MSG_FILEUPLOAD_COMPLETE = 102; // 文件上传结束, 可能是成功也可能是失败
    public static final int MSG_FILEUPLOAD_DELETE = 103; // 删除上传文件的消息, 包括已经上传的文件和正在上传的文件
    public static final int MSG_FILEUPLOAD_PAUSE = 104; // 文件上传暂停 (andiord 目前没有提供)

    public static final int MSG_APPKEY_CHECK = 200; // 发送 appkey 检查结果
    public static final int MSG_FILE_UPLOAD = 201; // 文件提交结束, 可能成功也可能失败
    public static final int MSG_FILE_MERGER = 202; // 文件合并结束, 可能成功也可能失败
}

public class RecordContext
{
    public static final int MSG_FILERECORD_END = 500; // 文件分包消息, 返回的是录制完成的文件夹名
    public static final int MSG_INIT_CAMERA_FAILED = 9999001; // 初始化编码器失败
    public static final int MSG_SDCARD_STATUS_ERROR = 9999002; // SD 卡访问失败
}
```

2.2.3 用户密钥的校验

类型	描述
函数	public static void appKeyCheck(String appKey)
参数	[in] appKey: appKey 密钥字符串
返回值	
说明	校验Appkey, 并通过回调通知用户校验结果, 在文件上传之前必须先调用这个接口, 校验成功或者失败时会返回 <i>MSG_APPKEY_CHECK</i> 消息

2.2.4 文件上传接口

类型	描述
函数	public static void newFileSync(FileInfo fileInfo)
参数	[in] fileInfo: 文件属性结构体
返回值	
说明	把传入的文件上传服务器, 文件上传中会不断的返回 <i>MSG_FILEUPLOAD_PROGRESS</i> 来提示上传进度, 文件上传成功或者失败后会返回 <i>MSG_FILEUPLOAD_COMPLETE</i> 消息,

```
public class FileInfo implements Cloneable
{
private String fileName; // 文件名, 是需要上传文件的本地文件名, 需要传入时填写好。
private String filePath; // 文件路径, 需要上传文件的本地文件路径, 需要传入时填写好。
private boolean bUpload; // 是否正在上传, 文件上传时内部使用, 传入时无需填写。
private String fileLength; // 文件的长度, 需要传入时填写好。
private String uploadName; // 文件在服务器保存时的文件名, 需要在传入时填写好。命名规则需要参考 demo。命名规则为: 唯一标识符_分片号.MP4 (如果是同一个录像文件的则分片号为不断递增)。如果是采用录播生成的文件则返回文件就可以作为 uploadName。如果是相册获取的文件则需要自己生成 uploadName。
private String uploadPath; // 文件上传到服务器后的路径, 传入时无需填写。
private String fileType; // 文件类型, 2: 直播视频; 3: 录播视频
}
```

2.2.5 删除上传文件接口

类型	描述
函数	<code>public static void deleteFileSync(FileInfo fileInfo)</code>
参数	[in] fileInfo: 文件属性结构体
返回值	
说明	删除上传的文件, 包括已经上传的和正在上传的, 如果正在上传的文件则取消这次上传并删除服务器上的文件, 删除成功后会返回MSG_FILEUPLOAD_DELETE

2.2.6 暂停所有上传中的文件

类型	描述
函数	<code>public static void pausedFileSync (FileInfo fileInfo)</code>
参数	[in] fileInfo: 文件属性结构体
返回值	
说明	暂停上传中的文件, 目前andiord不支持这个接口

2.2.7 文件提交接口

类型	描述
函数	<code>public static void fileUpload(FileInfo fileInfo)</code>
参数	[in] fileInfo: 文件属性结构体
返回值	
说明	把条件信息提交给服务器, 如果成功或者失败会返回MSG_FILE_UPLOAD消息

2.2.8 文件合并接口

类型	描述
函数	<code>public static void fileMerger(ArrayList<String> fileList)</code>
参数	[in] fileList: 需要合并的文件的 uploadName 列表
返回值	
说明	发送文件合并消息，发送的是uploadName的列表，该列表必须根据不同的分片进行排序。

2.2.9 获取当前正在上传的文件

类型	描述
函数	<code>public static String uploadingFileName()</code>
参数	
返回值	返回正在上传的文件路径和文件名。
说明	返回正在上传的文件路径和文件名。

2.2.10 获取版本号

类型	描述
函数	<code>public static String getVersion()</code>
参数	
返回值	返回 SDK 的版本号
说明	返回SDK的版本号

2.3 RecordContext 类的接口

`import com.qukan.qkrecorduploadsdk. RecordContext` 该类全部是静态方法，可以直接调用

2.3.1 初始化摄像机和麦克风

类型	描述
函数	<pre>public static boolean startCamera(SurfaceHolder sh, int cameraSizeType, int videoDataRate, int cameraId, int screenOrientation)</pre>
参数	<p>[in]sh: 需要显示摄像机预览画面的 surface</p> <p>[in]videoCameraSize: 视频尺寸</p> <pre>public static final int CAMERA_SIZE_320x240 = 1; public static final int CAMERA_SIZE_640x480 = 2; public static final int CAMERA_SIZE_320x180 = 3; public static final int CAMERA_SIZE_512x288 = 4; public static final int CAMERA_SIZE_640x360 = 5; public static final int CAMERA_SIZE_768x432 = 6; public static final int CAMERA_SIZE_1024x576 = 7;</pre> <p>[in]videoBitrate: 视频比特率(也称为视频码流), 单位 kbps. 一般来说, 低清 320*240 的码流是 200-300 之间; 标清 640*480 的码流是 500-800 之间; 高清 1280*720 的码流是 1200-1500 之间.</p> <p>[in]cameraId: 摄像机 ID, CameraInfo.CAMERA_FACING_BACK 或者 CameraInfo.CAMERA_FACING_FRONT</p> <p>[in]screenOrientation: 屏幕的方向</p> <pre>public static final int SCREEN_LANDSCAPE = ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE; public static final int SCREEN_PORTRAIT = ActivityInfo.SCREEN_ORIENTATION_PORTRAIT</pre>
返回值	<p>true: 表示初始化成功</p> <p>false: 表示初始化失败</p>
错误消息	<p>EventType.INIT_ENCODER_FAILED: 初始化编码器, 某些手机可能不支持</p> <p>EventType.SDCARD_STATUS_FAILED: SD 卡存储失败, 如果直播时开启了本地录像功能, 那么需要检查可用的 sd 卡, 如果该 sd 卡无法读取或容量不足, 那么会产生该错误消息</p>
说明	<p>在 init() 函数调用成功后, 用户在 surfaceCreate() 函数中初始化摄像头和麦克风</p>

2.3.2 关闭摄像头和麦克风

类型	描述
函数	<pre>public static void stopCamera()</pre>
参数	
返回值	

说明	用户结束录播，调用 stopRecord()方法之后，调用该方法关闭摄像头和麦克风，将资源归还给 android 系统。注意直播结束后，这个函数必须要调用，否则可能会造成摄像头和麦克风资源被抢占无法归还的情况。
----	---

2.3.3 开始文件录像

类型	描述
函数	public static boolean startRecord(boolean audioFlag ,String volume, String recordPath)
参数	[in] audioFlag: 是否开启声音，true 表示开启声音，false 表示关闭声音 [in] volume: SD 卡的路径 [in]videoCameraSize: 文件保存路径
返回值	true 表示成功， false 表示失败
说明	开始文件的录像，采用分段录像的方式，每个 15 分钟会打包一个文件，并通过 MSG_FILERECORD_END 消息把文件名回调出来，文件名的命名规则为：唯一标识符_分片号.MP4, 如果 appkey 没有登录则会返回 false。

2.3.4 结束文件录像

类型	描述
函数	public static void stopRecord()
参数	
返回值	
说明	结束文件录像，并通过MSG_FILERECORD_END消息把最后一个录像的名字回调出来。

2.3.5 暂停文件录像

类型	描述
函数	public static void pauseRecord()
参数	
返回值	
说明	暂停文件录像，需要再次开启文件录像的可以调用startRecord（开始文件录像接口）

2.3.6 开启或关闭闪光灯

类型	描述
函数	public static boolean switchFlash(boolean value)
参数	[in] flashFlag: true 表示开启闪光灯, false 表示关闭闪光灯
返回值	函数调用是否成功, true 表示调用成功, false 表示调用失败
说明	在调用 startCamera() 之后, 就可以调用该函数开启或关闭闪光灯。调用 startRecord () 录像的过程中, 也可以调用该函数开启或关闭闪光灯。

2.3.7 切换前后摄像头

类型	描述
函数	public static void switchCamera(int newCameraId)
参数	[in] newCameraId: 摄像机 ID, CameraInfo.CAMERA_FACING_BACK (后置) 或者 CameraInfo.CAMERA_FACING_FRONT (前置)
返回值	
说明	前后摄像头切换, 如果在录像时进行切换则会对录像文件产生切片。

2.3.8 开启或者关闭声音

类型	描述
函数	public static boolean switchAudio(boolean audioFlag)
参数	[in] audioFlag: 是否开启声音, true 表示开启声音, false 表示关闭声音
返回值	
说明	开启或者关闭录像时的声音。

2.3.9 设置手动聚焦或者自动聚焦

类型	描述
函数	<code>public static void setAutoFocus(boolean autoFocus , Camera.AutoFocusCallback autoFocusCallback)</code>
参数	[in] autoFocus: 是否自动聚焦, true 为自动聚焦, false 为手动聚焦 [in] autoFocusCallback: 自动聚焦的回调函数, 如果是设置为手动聚焦则传入 null。
返回值	
说明	设置手动聚焦或者自动聚焦

2.3.10 手动聚焦

类型	描述
函数	<code>public static void manulFocus(float x, float y, int nWidth, int nHeight, Camera.AutoFocusCallback autoFocusCallback)</code>
参数	[in] x: 手动聚焦的横坐标, 单位像素 [in] y: 手动聚焦的纵坐标, 单位像素 [in] surfaceWidth: 整个摄像头预览界面的像素宽度 [in] surfaceHeight: 整个摄像头预览界面的像素高度 [in] autoFocusCallback: 聚焦的回调函数, 可以设置为 null
返回值	
说明	手动聚焦的模式, 就可以调用该函数使手机的摄像头修改聚焦的焦点位置。 如果需要自动聚焦, 可以参看 demo 中提供的使用传感器触发聚焦的代码

2.3.11 截图

类型	描述
函数	<code>public static boolean takePicture(String fileName)</code>
参数	[in] fileName: 截图文件存放的路径和名字。包括保存的路径和文件名, 目前只支持 jpg 文件的保存
返回值	
说明	录播过程中的图片截图

2.3.12 设置 logo

类型	描述
函数	<pre>public static int setLogoInfo(int iLogoIndex, int iWidth, int iHeight, byte[] pcBmpData, int iOffset, int iLength)</pre>
参数	<p>[in] iLogoIndex: 设置 logo 的位置与大小</p> <pre>public static final int LEFT_LOGO_SMALL = 0; // logo 小 public static final int LEFT_LOGO_MIDDLE = 1; // logo 中 public static final int LEFT_LOGO_LARGE = 2; // logo 大 public static final int RIGHT_LOGO_SMALL = 3; // logo 小 public static final int RIGHT_LOGO_MIDDLE = 4; // logo 中 public static final int RIGHT_LOGO_LARGE = 5; // logo 大</pre> <p>[in] iWidth: 图片的宽度</p> <p>[in] iHeight: 图片的高度</p> <p>[in] pcBmpData: 图片的原始数据</p> <p>[in] iOffset: 图片数据的偏移位置</p> <p>[in] iLength: 图片数据的长度</p>
返回值	
说明	<p>在录播画面中添加上logo, 目前的位置只分左上和右上, logo的大小分小、中、大三种, 不同的分辨率会选择不同的大小。宽大于640的使用大的logo, 宽小于等于640并且大于320的使用中的logo, 宽小于等于320的使用小的logo。以上是SDK里内部选择的, 所以推荐用户设置logo时最好把大、中、小三种尺寸的logo都设置好, 以免出现切换到其他分辨率logo没有出现的情况了。</p>

三、DEMO 介绍

fileUploadActivity 演示了文件的选择、上传流程。

fileRecordActivity 演示了录像的流程。

四、Demo 的数据库介绍

表 qk_file_upload, 用于保存需要上传的文件

字段	属性	描述
filePath	String	文件路径
fileStatus	String	文件当前的状态。 <i>init</i> 未上传, <i>syncing</i> 上传中, <i>synced</i> 上传结束, <i>uploading</i> 提交中, <i>uploaded</i> 提交结束, <i>merge</i> 合并中, <i>end</i> 结束
uploadPath	String	服务器上文件存放的路径, 在文件上传时获取, 并在文件提交时需要使用
fileUploadName	String	服务器上文件存放的文件名, 录像产生的文件采用文件名就可以了, 相册选取的文件需要根据规则生成 (唯一标识符_分片.MP4), 在文件提交和文件合并时需要使用。

表 qk_file_map, 用户存放录像后生成的需要合并的文件。如果 fileList 为空则表示需要提交文件合并的消息, 合并的文件名采用 uploadList 里的文件名。

字段	属性	描述
key	String	录像生成后的分片文件名里的唯一标识符。
fileList	String	未合并的文件列表
uploadList	String	已经合并的文件列表