YFEncoderKit 帮助文档

文档修改记录

修改日期	版本/状态	作者	修改章节	修改描述
2016/3/17	1.0	谢家勋	略	初版帮助手册
2016/4/23	1.1	谢家勋	略	【附录 1】
2016/4/26	1.2	谢家勋	略	【附录 2】
2016/5/26	1.3	谢家勋	略	【附录 3】
2016/6/17	1.4	谢家勋	略	【附录 4】
2016/7/1	2.0	谢家勋	略	【附录 5】
2016/7/6	2.1	谢家勋	略	【附录 6】
2016/7/19	3.0	谢家勋	略	【附录 7】
2016/7/26	3.1	谢家勋	略	【附录 8】
2016/8/19	3.2	谢家勋	略	【附录 9】
2016/8/25	3.3	谢家勋	略	【附录 10】
2016/9/6	3.4	谢家勋	略	【附录 11】

目录

SDK 适用范围

-该 sdk 提供 Android 移动平台的本地视频录制、视频直播等功能。
-支持以下功能：

1、视频硬编码
2、前后摄像头动态切换
3、自动对焦/手动对焦/连续自动对焦/手动调整焦距/开关闪光灯
4、录制中截图
5、静音推流

SDK 鉴权

该 SDK 在使用前需进行验证，用户需到云帆官方网站注册，申请开通流媒体引擎服务。并获取 Access Key ID 和 Token。

其中 AccessKey 会被封装入用户拿到的 sdk，Token 则由用户填写。

注册完成后，用户可以自行填写域名白名单（IP 白名单暂时无效）。该 SDK 的推流地址必须包含白名单内的域名。

用户只需要在推流之前进行一次验证。验证方法如下：

```
EncoderAuthentication.getInstance().authenticate(TOKEN, authCallBack);
```

其中 authCallBack 是回调方法，可自行实现具体内容：

```
private EncoderAuthentication.AuthCallBack authCallBack = new
EncoderAuthentication.AuthCallBack() {
        @Override
        public void onAuthenticateSuccess() {
            Log.d(TAG, "鉴权成功~！");
        }

        @Override
        public void onAuthenticateError(int errorCode) {
            Log.d(TAG, "鉴权失败啦：" + errorCode);
        }
    };
```

一般而言只需要在打开 App 的时候进行一次鉴权即可。

新版鉴权提供了查询鉴权状态的方法，

```
EncoderAuthentication.getInstance().isAuthenticateSucceed()
```

通过返回的布尔值可以知道鉴权是否成功，如果返回 false 可以尝试再次鉴权。

然后就可以愉快地直播了！

API for YfEncoderKit

YfEncoderKit 是一个用于拍摄视频或视频直播用的库。

直播使用方法

1、设置各种回调方法，即实现接口 YfEncorderKit.RecordMonitor

2、初始化摄像头，参考代码如下：

```
TextureView s = (TextureView)findViewById(R.id.surface);
YfEncorderKit r = new YfEncorderKit(this,CACHE_DIRS, false);
r.setContinuousFocus()//设置连续自动对焦
        .setRecordMonitor(this)//设置回调
        .openCamera(s);//设置预览窗口
mRecorder = r;
```

3、开始录制：

```
yfEncorderKit.changeMode(YfEncorderKit.MODE_LIVE);
//设置为直播模式
yfEncorderKit.setLiveUrl(mUrl.getText().toString());
//设置为直播服务器地址
yfEncorderKit.startRecord();
//开始录制并上传
```

4、使用结束后，释放 YfEncoderKit:

```
yfEncorderKit.release();
```

本地视频录制使用方法：

1、2 步同上。

3、开始录制：

```
yfEncorderKit.changeMode(YfEncorderKit.MODE_VOD);
```

```
//设置为本地模式
  yfEncorderKit.setVodDir(mUrl.getText().toString());
//设置视频保存地址
  yfEncorderKit.startRecord();
```

公共方法

public YfEncorderKit(Context context, String appPath, boolean filterEnable, int previewWidth, int previewHeight,int frameRate)

构造函数。

参数

context:上下文环境

appPath: 用于存储截图、视频等内容的根目录路径

filterEnable:是否允许使用滤镜模式（滤镜仅支持 4.3 或以上安卓系统，允许该模式后将无法后台推视频流，***且无法使用软编***）

previewWidth 预览宽度，即摄像头输出宽度。安卓 4.3 及以上版本并允许使用滤镜的模式下（filterEnable==true），预览宽高须大于等于编码宽高，并保持相同比例；不允许使用滤镜模式及 4.3 以下版本，预览宽高须大于等于编码宽高，当编码宽高与预览宽高不一致时，编码器会自行裁剪。

previewHeight 预览高度，即摄像头输出高度

framerate 摄像头帧率，与编码帧率一致

public void setRecordMonitor(RecordMonitor m)

设置视频录制回调

public YfEncorderKit setLandscape(boolean landscape)

设置是否使用横屏推流模式，详情参考横屏推流使用方式

public void openCamera(TextureView texture)

设置预览窗口并打开摄像头

参数

texture：预览摄像头画面的窗口。

public void resume()

*必须*在 Activity.onResume()的时候调用，继续录制/上传视频

public YfEncoderKit configLogo(Bitmap bitmap, float widthPercent, float heightPercent, float xPercent, float yPercent)

配置 logo 的源及在画面中的位置，请注意屏幕的横屏竖屏模式。

参数

bitmap: logo 源

widthPercent：logo 的宽度占屏幕宽度的比例（0~1）

heightPercent：logo 的高度占屏幕高度的比例（0~1），譬如宽度设置为 0.2f，那么在通常 16:9 竖屏的情况下这里就应该是 0.2f * 9 / 16

xPercent：logo 左边缘相对屏幕左边缘的距离比（0~1），通常情况下，该值与 widthPercent 之和不应大于 1，否则 logo 则无法完全显示在屏幕内

yPercent：logo 上边缘相对屏幕上边缘的距离比（0~1），通常情况下，该值与 heightPercent 之和不应大于 1，否则 logo 则无法完全显示在屏幕内

public static final String getRecordStateString(int state)

获取当前录制的状态描述

参数：

state：当前状态，通过 RecordMonitor 回调获得

public void pause()

Activity.onStop()时调用，关闭摄像头并停止推流，调用该方法后恢复时会自动重新推流。

public void release()

Activity.onDestroy()时调用，释放 SurfaceView 对 YfEncoderKit 的引用。

public void enalePushAudio(boolean enable)

是否推送音频流(默认推送)

public boolean isRecording()

是否正在录制状态。

public boolean changeMode(int mode, boolean hardEncoder, int bitrate, int videoWidth, int videoHeight)

切换模式。

参数：

mode：可选 YfEncorderKit.MODE_VOD 或 YfEncorderKit.MODE_LIVE，分别代表本地视频录制和直播上传视频录制

hardEncoder：是否使用硬解

bitrate：编码码率,单位 kb

videoWidth：编码宽度，安卓 4.3 或不允许开启滤镜的模式下，编码宽高必须小于等于摄像头输出宽高；安卓 4.3 及以上版本并允许开启滤镜的模式下，编码宽高和摄像头输出宽高必须保持相同比例。

videoHeight：编码高度

public int getMode()

获取当前录制模式。

public boolean isLive()

判断当前是否直播模式。

public boolean setLiveUrl(String url)

设置直播服务器地址。

public boolean setVodDir(String dir)

设置本地视频存储路径。

public YfEncorderKit setContinuousFocus()

设置连续自动对焦。

public int getCameraId()

获取摄像头 ID。

public boolean isFrontCameraUsed()

判断当前是否为前置摄像头。

public boolean switchCamera()

切换摄像头，当切换失败时返回 false，成功则返回 true。

public boolean canSwitchCamera()

判断当前设备是否支持切换摄像头。

public boolean supportFlash()

判断当前设备是否支持闪光灯。

public boolean isFlashOn()

判断当前闪光灯是否已开启。

public boolean setFlash(boolean on)

打开/关闭 闪光灯。

public boolean startRecord()

开始录制视频。

public void captureCurrentFrame(String saveName)

截取当前画面并保存。

public boolean manualFocus(Rect rect)

手动对焦，返回对焦成功与否。

public boolean manualZoom(Rect rect)

手动调焦距，返回调整成功与否。

public boolean autoFocus(Rect rect)

打开自动对焦，返回打开成功与否。

public boolean stopRecord()

停止视频录制。

public YfEncoderKit setDefaultCamera(Boolean setFrontCameraDefault)

是否设置前置摄像头为默认摄像头

public boolean setFilter(YfFilterType filterType)

启用滤镜。

目前滤镜包括有：

```
public enum YfFilterType {
    NONE,
    BEAUTY,
    LOGO
}
```

分别代表取消滤镜、美颜、添加水印。

使用水印滤镜必须先调用方法 configLogo 配置水印源及宽高、位置等。

public boolean addFilter(YfFilterType filterType)

在原有滤镜的基础上增加滤镜效果，如果添加的是 YfFilterType.NONE，则会取代所有原滤镜

public boolean removeFilter(YfFilterType filterType)

移除某滤镜效果

public void setAdjustQualityAuto(boolean adjustQualityAuto, int minBitrate)

设置是否根据网速自动调整码率/帧率及设置调整间隔，该设置需在 startRecord()之前调用才能生效

参数

adjustQualityAuto 是否打开自适应码率/帧率

minBitrate 调整的最低码率,单位 kb

public void setBufferSizeBySec(int bufferSizeBySec)

设置可播放的缓存时间，单位秒，需在开始录制之前设置，默认 4 秒，最小值为 1 秒

public void setMaxReconnectCount(int maxReconnectCount)

设置最大重连次数，0 代表关闭自动重连功能----该方法目前被屏蔽使用，暂时关闭自动重连功能。

public static boolean canUsingHardEncoder()

判断当前手机是否能使用硬编。

public static boolean canUsingFilter ()

判断当前手机是否能使用滤镜。

RecordMonitor 回调方法：

public void onError(int mode, int camId, int err, String msg)

录制错误回调。
回调参数：

mode:录制模式，包括 YfEncorderKit.MODE_VOD 与 YfEncorderKit.MODE_LIVE

camId：摄像头 ID

err：错误码

msg：附加信息

错误码包括：

```
public static final int ERR_CAMERA = 1; // 相机错误

public static final int ERR_MUX = 2;      // mux 错误

public static final int ERR_SYSTEM = 3;   // 系统错误

public static final int ERR_MUX_START = 4;   // 开始 muxer 失败

public static final int ERR_MUX_END = 5;      // 停止 muxer 失败
```

public void onStateChanged(int mode, int camId, int oldState, int newState)

录制状态改变。
回调参数：

mode:录制模式，包括 YfEncorderKit.MODE_VOD 与 YfEncorderKit.MODE_LIVE

camId: 摄像头 ID

oldState:原本状态

newState:当前状态

状态值包括：

```
public static final int STATE_UNKNOWN = -2;

public static final int STATE_ERROR = -1;

public static final int STATE_IDLE = 0;

public static final int STATE_PREPARED = 1;

public static final int STATE_RECORD_PREPARING = 2;

public static final int STATE_RECORDING = 3;
```

public void onFragment(int mode, int camId, String fragPath)

生成了新的录制片段。
回调参数：

mode:录制模式，包括 YfEncorderKit.MODE_VOD 与 YfEncorderKit.MODE_LIVE

camId: 摄像头 ID

fragPath:片段路径

public void onTimeUpdate(int time)

录制时间刷新。

public void onCapturedResult(String path)

调用 captureCurrentFrame()后返回的截图结果。

回调参数：

Path:截图成功时为图片路径，失败时为 null

public void onBufferHandleCallback(int currentBitrate, int bufferMs, int event)

当码率调整时的回调，需开启码率自适应。
回调参数：

currentBitrate：当前码率

bufferMs：当前缓存大小（单位秒）

event:恒为 0

public void onEncodeOverLoad(YfFilterType... removeFilterType);

编码帧率异常时的回调，此时如果有开启美颜或其他高损耗滤镜的话，会被强制关闭。

回掉参数:

removeFilterType:被关闭的滤镜列表

横屏推流使用方法：

1、在 AndroidManifest.xml 将当前 activity 设置为横屏模式；

2、打开摄像头(openCamera(GLSurfaceView texture))之前设置横屏模式

setLandscape(boolean landscape)

3、记得根据横竖屏调整 surface 在屏幕的宽高

API for YfMuxerManager

YfMuxerManager 是一个用于推送直播流的类。

推流使用方法

1、实现接口 YfMuxerManager.OnMuxerCallback

```
public interface OnMuxerCallback {
    public abstract void onMuxStart(String url);

   public abstract void onMuxFinished(String url);

    public abstract void onMuxError(int err, String msg);

    public abstract void onMuxSpeed(int speedKb);

    public abstract void onBufferOverflow();
}
```

2、用上述接口初始化 YfMuxerManager:

```
mMuxer = new YfMuxerManager(mOnMuxerCallback, context);
```

3、 实例化参数 YfMuxerManager.MuxerParams 并赋值

4、 启动 muxer

```
mMuxer.startMuxer(mParams)
```

5、 往 muxer 推送编码后的音视频数据

```
mMuxer.sendAudioData(flag, length, data, pts, dts);
 mMuxer.sendVideoData(flag, length, data, pts, dts);
```

6、 推流结束停止 muxer

```
    mMuxer.stopMuxer();
```

回调说明

```
public interface OnMuxerCallback {
    //muxer 启动成功
    public abstract void onMuxStart(String url);
    //muxer 关闭完成
    public abstract void onMuxFinished(String url);
    //muxer 异常回调，异常类型见下文
    public abstract void onMuxError(int err, String msg);
    //muxer 推流回调
    public abstract void onMuxSpeed(int speedKb);
    //muxer 推流时丢帧回调
    public abstract void onBufferOverflow();
}

public static final int ERR_CAMERA = 1; // 相机错误
public static final int ERR_MUX = 2;    // mux 错误
public static final int ERR_SYSTEM = 3;  // 系统错误
public static final int ERR_MUX_START = 4;  // 开始 muxer 失败
public static final int ERR_MUX_END = 5;    // 停止 muxer 失败
```

YfMuxerManager.MuxerParams 参数说明：

```
public class MuxerParams {
    /**
     * 推流地址，必须为 RTMP 协议
     */
    public String liveUrl;
    /**
     * 视频宽度
     */
    public int frameWidth;
    /**
     * 视频高度
     */
```

```
    public int frameHeight;
    /**
     * 帧率
     */
    public int frameRate;
    /**
     * 比特率（单位 kb）
     */
    public int frameBitrate;
    /**
     * 关键帧距离
     */
    public int frameIInterval;
    /**
     * 缓存的时间，当缓存的数据能播放的时间大于该值时，会进行丢帧，丢帧会有回调
     */
    public int bufferSecs;
    /**
     * 统计一次平均上传速度并回调的时间间隔
     */
    public int netReportInterval;
}
```

附录

【1】 增加设置默认摄像头的方法。

【2】 增加开启转码功能的方法。

【3】 1、删除 pauseVideo()/pauseAudio()方法，增加 enalePushAudio(boolean enable)方法;

2、captureCurrentFrame()方法增加保存文件名参数,改为 captureCurrentFrame(String saveName)

3、增加构造函数的说明

4、增加 setFilter(YfFilterType filterType)方法

5、支持后台推流

【4】 1、增加方法 setAdjustQualityAuto()开关自适应码率;

2、增加方法 setMaxReconnectCount()设置自动重连次数

3、删除配置转码的方法 setTranscodingOn()

4、录制本地视频设置的参数由视频路径改为视频名称

【5】 独立 YfMuxerManager 模块

【6】 添加 sdk 鉴权说明

【7】 1、增加方法：setLandscape(boolean landscape)、setBufferSizeBySec(int sec)
2、更改构造函数，增加预览宽高的设置；
3、添加编码宽高、帧率的设置
4、自适应码率增加最低码率设置
5、添加横屏推流模式的说明

【8】 1、暂时关闭 YfMuxerManager 模块
2、将帧率改为在构造函数处设置
3、不再开放码率自适应速度上报间隔

【9】 新增软解功能

【10】 1、删除 pause()方法
2、新增方法：configLogo()、addFilter()、removeFilter()方法
3、更新鉴权方法

【11】 1、还原 pause()方法
2、新增方法 canUsingFilter()、canUsingHardEncoder()
3、暂时关闭自适应码率功能
4、暂时关闭 SDK 内自动重连功能，屏蔽 setMaxReconnectCount()方法