**快传技术**

SDK接口使用文档

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 草稿  [ ] 正式修改  [ ] 正在发布 | 文件标识： | 快传SDK接口使用文档 |
| 当前版本： | mac版1.0.0 |
| 作 者： | 温清林 |
| 完成日期： | 2017/6/12 |

修订历史

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 版本 | 编写/修订说明 | 修订人 | 修订日期 | 备注 |
| 1 | 1.0.0 | 新建文档 | 温清林 | 2017/6/12 |  |
| 2 |  |  |  |  |  |

目 录

[一、 项目概述 4](#_Toc471373531)

[1、概述 4](#_Toc471373532)

[2、阅读对象 4](#_Toc471373533)

[3、术语与缩写解释 4](#_Toc471373534)

[二、产品角色 4](#_Toc471373535)

[三、SDK接口使用说明 5](#_Toc471373536)

[1、初始化及配置 5](#_Toc471373537)

[1.1 初始化SDK 5](#_Toc471373538)

[1.2 添加监听器接口 5](#_Toc471373539)

[1.3 连接云平台 5](#_Toc471373540)

[1.4 获取连接云平台状态 6](#_Toc471373541)

[1.5 ConnectionListener 监听器说明 6](#_Toc471373542)

[1.6 开启日志接口 7](#_Toc471373543)

[1.7 断开云平台连接 7](#_Toc471373544)

[1.8 获取SDK版本号 7](#_Toc471373545)

[2、VoIP通话控制接口 8](#_Toc471373546)

[2.1 添加VOIP通话状态回调 8](#_Toc471373547)

[2.2 删除VOIP通话状态回调 8](#_Toc471373548)

[2.3 发起呼叫 8](#_Toc471373549)

[2.4 释放通话(挂断) 9](#_Toc471373550)

[2.5 被叫接听 9](#_Toc471373551)

[2.6 发送DTMF 10](#_Toc471373552)

[2.7 设置扬声器 10](#_Toc471373553)

[2.8 获取扬声器状态 10](#_Toc471373554)

[2.9 设置静音 11](#_Toc471373555)

[2.10 获取静音状态 11](#_Toc471373556)

[3、VoIP通话回调接口(类：CallStateListener) 11](#_Toc471373557)

[3.1 呼叫失败 11](#_Toc471373558)

[3.2 呼入界面 12](#_Toc471373559)

[3.3 呼叫被释放(挂断) 13](#_Toc471373560)

[3.4 呼叫振铃中 13](#_Toc471373561)

[3.5 被叫接听 13](#_Toc471373562)

[4、VOIP状态码对应表 14](#_Toc471373563)

# 项目概述

## 1、概述

SDK（SoftwareDevelopmentKit,即软件开发工具包）是快传技术提供给手机端和PC端的Native Apps开发者快速集成云通讯能力的一种便捷开发工具集，SDK提供了VoIP旨在为第三方开发者在应用内快速、高效、低成本集成语音业务提供了一站式的服务。

云通讯平台Mac SDK 以静态库的方式提供给Mac平台开发人员。可以与固定电话或者其他安装有云通讯平台客户端的设备进行语音通信，包括Web浏览器和其他Android移动设备等。

## 2、阅读对象

需求评审人员、开发人员、测试人员等相关人员。

## 3、术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
| SDK | SoftwareDevelopmentKit即软件开发工具包 |
| DTMF | Dual Tone Multi Frequency即双音多频信号DTMF |
|  |  |

# 二、产品角色

开发者、合作伙伴等。

# 三、SDK接口使用说明

## 1、SDK架构

Mac 的SDK 包含2个静态库：libTcp.a和libVideo.a

### 1.1 libTcp.a

libTcp.a提供了tcp协议通信的功能，包括arms认证，连接cs服务器，打电话过程的信令交互，以及链接状态返回。该库的所有API接口都 定义在KCTTcpClient.h文件中。链接的所有状态码定义在KCTTcpDefine.h文件中；链接的状态码通过KCTTcpClient的delegate返回。

### 1.2 libVideo.a

libVideo.a提供了打电话相关的业务操作，包括拨打电话、挂断电话、接听电话等操作。

这些操作的API 都定义在KCTService.h文件中。KCTEvent.h文件提供了SDK往上层APP的回调，所有的SDK上报事件都 通过KCTService的delegate返回到上层APP。

## 2、信令交互接口说明

### 2.1 KCTTcpClient类初始化

/\*!

\* @brief 实例化KCTTcpClient的一个单例对象

\* @discussion 这个方法是生成KCTTcpClient的唯一方法，请不要使用init()等其他初始化方法，否知会出现未知的错误

\* @return KCTTcpClient类型的单例对象

\*/

+ (instancetype)sharedTcpClientManager;

/\*!

\* @brief 设置代理.

\*

\* @discussion 如果需要通过回调监控tcp连接状态之类的,就必须设置这个代理

\*

\* @param delegate tcp事件回调代理

\*/

- (void)setTcpDelegate:(id <KCTTCPDelegateBase>) delegate;

### 2.2 连接平台

/\*!

\* @brief 连接云平台.

\* @discussion (注意：这个登陆是明文登录方式：accountSid、accountToken、clientPwd都必须要MD5加密）

\* @param accountSid 开发者账户

\* @param accountToken 开发者账户密码

\* @param clientNumber client账户(用户账户)

\* @param clientPwd client密码(用户)

\* @param success 成功回调

\* @param failure 失败回调

\*/

//明文登录方式：

- (void)connect:(NSString \*)accountSid accountToken:(NSString \*)accountToken clientNumber:(NSString \*)clientNumber clientPwd:(NSString \*)clientPwd success:(void (^)(NSString \*userId))success failure:(void (^)(KCTError \*error))failure;

### 2.3 获取连接云平台状态

/\*!

\* @brief 查询客户端与云平台的连接状态,

\* @discussion

\*

\* @return yes，已连接；no，断开连接

\*/

- (BOOL)login\_isConnected;

/\*!

\* @brief 查询当前网络状态

\* @discussion

\*

\* @return 当前网络状态

\*/

- (KCTNetworkStatus)getCurrentNetWorkStatus;

### 2.4 断开云平台连接

/\*!

\* @brief 断开客户端和云平台的连接,并且根据flag标识的bool值选择是否关闭离线推送。

\*

\* @param flag 是否关闭推送的bool值。YES，关闭。NO，不关闭。

\*

\* @discussion 断开连接。

\*/

- (void)login\_uninitWithFlag:(BOOL)flag;

## 3、VoIP通话控制接口（KCTService）

### 3.1 初始化服务

#pragma mark - -------------------初始化函数-----------------------

//初始化实例

- (KCTService \*)initWithDelegate:(id<KCTEventDelegate>)delegate;

//设置代理方法

-(void)setDelegate:(id< KCTEventDelegate>)delegate;

### 3.2 呼叫控制相关函数

/\*\*

\* 拨打电话

\* @param callType 电话类型 0 语音电话 2 视频电话

\* @param called userid号码

\*/

-(void)dial:(NSInteger)callType andCalled:(NSString\*)calledNumber andUserdata:(NSString \*)callerData;

/\*\*

\* 挂断电话

\* @param callid 电话id

\*/

- (void) hangUp: (NSString\*)callid;

/\*\*

\* 接听电话

\* @param callid 电话id

\*/

- (void) answer: (NSString\*)callid;

/\*\*

\* 拒绝呼叫(挂断一样,当被呼叫的时候被呼叫方的挂断状态)

\* @param callid 电话id

\*/

- (void)reject: (NSString\*)callid;

### 3.3 DTMF函数

/\*\*

\* 发送DTMF

\* @param callid 电话id

\* @param dtmf 键值

\*/

- (BOOL)sendDTMF: (char)dtmf;

### 3.4 本地功能

/\*\*

\* 免提设置

\* @param enable NO:关闭 YES:打开

\*/

- (void) setSpeakerphone:(BOOL)enable;

/\*\*

\* 获取当前免提状态

\* @return NO:关闭 YES:打开

\*/

- (BOOL) isSpeakerphoneOn;

/\*\*

\* 静音设置

\* @param on NO:正常 YES:静音

\*/

- (void)setMicMute:(BOOL)on;

/\*\*

\* 获取当前静音状态

\* @return NO 正常 YES 静音

\*/

- (BOOL)isMicMute;

### 3.5 辅助能力

/\*\*

\* 获取SDK版本信息

\*/

- (NSString\*) getKCTSDKVersion;

/\*\*

\* 获取流量分类统计信息

\*/

- (NSDictionary\*) getNetWorkFlow;

/\*\*

\* 设置当前网络是否是IPV6，默认IPV4，建议在初始化完成后设置，也可在拨打电话之前设置。

\* @param isIpv6 NO:不是IPv6 YES:是IPv6

\*/

- (void)setIpv6:(BOOL)isIpv6;

### 3.6 视频相关

/\*\*

\* 初始化视频显示控件（本地视频显示控件和对方视频显示控件）

\* 参数 frame 窗口大小

\* @return NSOpenGLView视频显示控件:

\*/

- (NSOpenGLView \*)allocCameraViewWithFrame:(CGRect)frame;

/\*\*

\* 设置视频显示参数

\* 参数 localVideoView 设置本地视频显示控件

\* 参数 remoteView 设置对方视频显示控件

\* @return NO: YES:

\*/

-(BOOL)initCameraConfig:(UIView\*)localVideoView withRemoteVideoView:(UIView\*)remoteView;

/\*\*

\* 自定义视频编码和解码参数

\* @param KCTVideoEncAttr 编码参数

\* @param KCTVideoDecAttr 解码参数

\*/

- (void)setVideoAttr:(KCTVideoEncAttr\*)kctVideoEncAttr andVideoDecAttr:(KCTVideoDecAttr\*)kctVideoDecAttr;

/\*\*

\* 旋转显示图像角度

\* 参数 sendRotation 本地显示图像角度 数值为0 90 180 270

\* 参数 reciviedRotation 对端显示图像角度 数值为0 90 180 270

\* @return NO: YES:

\*/

- (BOOL)setRotationVideo:(unsigned int)sendRotation withReciviedRotation:(unsigned int)reciviedRotation;

/\*\*

\* 获取摄像头个数

\*/

- (int)getCameraNum;

/\*\*

\* 摄像头切换 后置摄像头：0 前置摄像头：1

\* @return YES 成功 NO 失败

\*/

- (BOOL)switchCameraDevice:(int)CameraIndex;

/\*\*

\* 切换视频模式：发送、接收、正常模式

\*

\* @param type CAMERA\_RECEIVE : 只接收视频数据（只能接收到对方的视频）

CAMERA\_SEND : 只发送视频数据（只让对方看到我的视频）

CAMERA\_NORMAL : send receive preview

\* @return YES 成功 NO 失败

\*/

- (BOOL)switchVideoMode:(KCTCameraType)type;

/\*\*

\* 视频截图

\* @param islocal: 0 是远端截图 1 是本地截图。

\* @param filename: 截图名称。

\* @param savePath: 存放路径。

\*/

- (void)cameraCapture:(int)islocal withFileName:(NSString\*)filename withSavePath:(NSString\*)savePath;

/\*\*

\* 视频来电时是否支持预览

\* isPreView: YES 支持预览 NO 不支持预览。

\* @return YES 成功 NO 失败

\*/

- (BOOL)setCameraPreViewStatu:(BOOL)isPreView;

/\*\*

\* 获取视频来电时是否支持预览

\* @return YES 支持预览 NO 不支持预览

\*/

- (BOOL)isCameraPreviewStatu;

/\*\*

\* 离线来电推送

\*/

-(void)callIncomingPushRsp:(NSString\*)callid withVps:(NSInteger)vpsid withReason:(KCTReason\*)reason;

## 4、VoIP通话回调接口(KCTEvent)

### 4.1 引擎初始化成功回调

#pragma mark 初始化函数代理

//引擎初始化成功

- (void)onInitEngineSuccessful:(NSInteger)result;

这个方法必须实现。

### 4.2 音频通话相关回调

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*互联网音视频通话的代理\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//获取当电通话Callid，前提是收到APNS推送消息后初始化callid

//收到来电回调

- (void)onIncomingCall:(NSString\*)callid withcalltype:(KCTCallTypeEnum) callType withcallerNumber:(NSString\*)callerNumber;

//收到来电回调(支持昵称)

- (void)onIncomingCall:(NSString\*)callid withcalltype:(KCTCallTypeEnum) callType withcallerinfo:(NSDictionary \*)callinfo;

//呼叫振铃回调

-(void)onAlerting:(NSString\*)called withVideoflag:(int)videoflag;

-(void)onAlerting:(NSString\*)called;

//接听回调

-(void) onAnswer:(NSString\*)callid;

//呼叫失败回调

- (void) onDialFailed:(NSString\*)callid withReason:(KCTReason\*)reason;

//释放通话回调

- (void) onHangUp:(NSString\*)callid withReason:(KCTReason\*)reason;

// 语音质量上报回调

- (void)onNetWorkState:(KCTNetworkState)networkState;

### 4.3 视频截图回调

- (void)onCameraCapture:(NSString\*)cameraCapFilePath;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*对方视频模式回调\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

- (void)onRemoteCameraMode:(KCTCameraType)type;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DTMF回调\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

- (void)onDTMF:(NSString \*)value;

### 4.4 rtp包加密解密回调

// rtp解密回调

- (void)decryptCallBack:(char \*)inMsg withOutMsg:(char \*)outMsg withInLen:(int)inLen withOutLen:(int \*)outLen;

// rtp加密回调

-(void)encryptCallBack:(char \*)inMsg withOutMsg:(char \*)outMsg withInLen:(int)inLen withOutLen:(int \*)outLen;