VoIP C++ SDK User's Guide

1.VoIP C++ SDK Introduction

Easiio VoIP C++ SDK is a PC SDK for Audio and Video Communication using Easiio Cloud Video PBX backend, with enhanced audio and video encoding and transport technology developed by Easiio. Including the multiparty video conference capability, support mobile clients with HD audio and video, better network transport, reduce the effect of network packet loss on the audio and video quality, such that the audio and video is acceptable with 10 – 20 % packet loss rate.

2. How to include Easiio VoIP C++ SDK into your application

2.1 Easiio VoIP C++ SDK files：

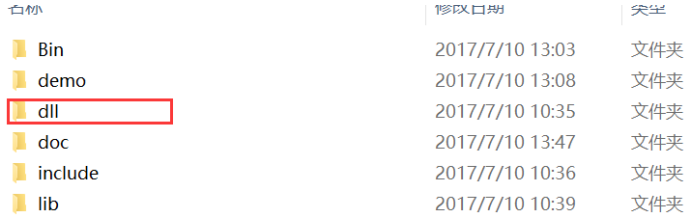
* EasiioLibrary.h is the header file for Easiio SDK

* EasiioLibrary.dll is SDK dynamic library

EasiioLibrary.lib is the static library

* the other dll files should be dynamic library，put all of those dll file under the same folder of your exe file

dll files is under the "dll" folder：



2.2 Easiio VoIP C++ SDK integration guide

2.2.1 Initialization

（1）Pass the call back function EASIIO_CALLBACK_INTERFACE

（2）Initialized with Developer Key and Token

```
EASIIO_CALLBACK_INTERFACE *cbInterface;

cbInterface = new EASIIO_CALLBACK_INTERFACE;
memset(cbInterface, 0, sizeof(EASIIO_CALLBACK_INTERFACE));
cbInterface->onLoginResult = &onLoginResult;
cbInterface->onPJSIPServiceStateChanged = &onPJSIPServiceStateChanged;
cbInterface->onCallStatusChanged = &onCallStatusChanged;
cbInterface->onTwoWayCallStatusChanged = &onTwoWayCallStatusChanged;
cbInterface->onIncomingCall = &onIncomingCall;
```

```
cbInterface->onMakeTwoWayCallResult = &onMakeTwoWayCallResult;
cbInterface->onCallRecordingReaponse = &onCallRecordingReaponse;
cbInterface->onGetAllRecordResponse = &onGetAllRecordResponse;
cbInterface->onGetRecordResponse = &onGetRecordResponse;
cbInterface->onMakeCallResult = &onMakeCallResult;
cbInterface->onMeetingResult = &onMeetingResult;

// Initialize call back function
EasiioInit(cbInterface);

// Initialize Developer Key and Token
EasiioInitDeveloperKeyAndToken("[Developer Key]", "[Token]");
```

2.2.2 Login and logout

（1）Login

Call EasiioLogin(const char* account, int paramCount, EasiioLoginParams* params);

account: Third party account，use the application's own account number.

paramCount: Additional parameters count

params: Pointer to the first parameters, see sample for details

Login result will be in call back function: void(*onLoginResult)(int resultCode, const char* resultMsg);

（2）Logout

Call EasiioLogout(); SDK will disconnect the account

2.2.3 SIP status call back

void(*onPJSIPServiceStateChanged)(int stateCode); // SIP status notification

stateCode please see header file EasiioLibrary.h for details and explanations of SIP State

2.2.4 VoIP call

call EasiioMakeCall(const char* number, int callType, const char* postValue);

Number: Number to call

callType: Type of call

postValue: additional parameter to call，shorter than 256 characters base64 encoded string.Space and other special character is not allow. Otherwise it could results in call failed.

Status call back：

void(*onMakeCallResult)(int callId, const char* toNumber，const char* meetingkey);

result please see EasiioLibrary.h

void(*onCallStatusChanged)(int callId, int callState, const char* resultCode, const char* callUUID，);

Details result of the call back please see header file EasiioLibrary.h，the CallUUID can be used for retrieving the audio recording file.

2.2.5 Two way call

call EasiioMakeTwoWayCall(const char* toNumber, const char* fromNumber, int openRecord);

toNumber: called number

fromNumber: caller number

openRecord: turn on recording or not

Call back for the call results：

void(*onMakeTwoWayCallResult)(EasiioResponseReason reason, const char* caller, const char* callee, const char* callUUID, const char* retryId);

Two way call call back function：

void(*onTwoWayCallStatusChanged)(int callState, const char* callUUID, const char* caller, const char* callee, int retrying);

2.2.6 Call control functions

（1） To Answer

call EasiioAnswerCall(int callId, int callType);

callId is from the Call status notification, for example:

void(*onIncomingCall)(int callId, int callType, const char* fromNumber, const char* postValue); // incoming call notification

callType call type support Voice and video

（2） To Reject

call EasiioRejectCall(int callId);

callId is from the Call status notification, same as To Answer

（3） To Hangup

Call EasiioHangupCall(int callId);

callId，same as incoming call，if it is an outbound call, it is from the return value of the make call function.

（4） Hold/Unhold

Hold call EasiioHoldCall(int callId);
Unhold call EasiioUnHoldCall(int callId);

callId：same as above

（5） Mute/Unmute

Mute call EasiioMuteCall(int callId);

Unmute call EasiioUnMuteCall(int callId);

callId：same as above

（6） Send DTMF

Call EasiioSendDTMF(int callId, const char* dtmf);

callId：same as above

Dtmf： digit 0-9，*，# single character string

（7） Adjust mic volumn

Call EasiioAdjustMicLevel(int callId, int level);

callId：same as above

Level: 0-40

（8） Turn on and off recording

Call EasiioSwitchRecordInCall(int recordSwitch, const char* callUUID);

recordSwitch: Recording Switch

CallUUID: Received in call status call back function

void(*onCallStatusChanged)(int callId, int callState, const char* resultCode, const char* callUUID);

Result of the call return by call back function：

void(*onCallRecordingReaponse)(EasiioResponseReason reason, int switchRecord, const char* callUUID);

2.2.7 Retrieve PBX internal account number

Call：EasiioGetCurrentPBXAccount(char* pbxAccount);

2.2.8 Retrieve audio recording table

（1）Retrieve all recording list

Call： EasiioGetAllRecord(int recordType);

recordType: There are two type one is in call recording, one is complete recording

Call back：

void(*onGetAllRecordResponse)(EasiioResponseReason reason, int count, EasiioRecord *record);

（2）Retrieve single call recording table

Call： EasiioGetRecord(int recordType, const char* callUUID);

recordType: same as above

CallUUID: obtained by call status call back function.

Result call back：

void(*onGetRecordResponse)(EasiioResponseReason reason, const char* callUUID, int count, EasiioRecordItem *recordItem);

2.2.9 Surveillance camera count

(1) Retrieve surveillance camera count

Call： EasiioGetDeviceCount(int *DevCount);

(2) Retrieve single surveillance camera info

Call：EasiioGetDeviceInfoById(int index, char *deviceInfo);

index：Camera index，If number of camera is n，index is from 0 to (n-1);

(3) Start surveillance camera preview

Call：EasiioStartPreview(int size, char **urlArray);

size: of the url array

urlArray：is the rtsp camera url array，retrieve by (2) above

(4) Close surveillance camera

Call：EasiioStopPreview();

(5) Check surveillance status

Call：EasiioPreviewStatus(int *status);

2.3.0 Video conference

Before start video conference make sure the account support video conference. Video conference support single layer or multi-layer conference. Please see the configuration of the admin page. (Default is single layer meeting)

(1) Create meeting

Call：EasiioCreateMeeting(const char* desc, int priority);

Call back result：

void(*onMeetingResult)(EasiioResponseReason reason,int count, MeetingInfo *info);

(2) Get all meeting tables

Call：EasiioGetMeetingList();

Call back result：

void(*onMeetingResult)(EasiioResponseReason reason,int count, MeetingInfo *info);

(3) Close meeting

Call：EasiioDeleteMeeting(const char* meeting_key);

(4) Start meeting

Call：EasiioStartMeeting(const char* meeting_key,const char* resolution,const char* platform,char* start_time);

resolution: Video resolution，may be different resolution from application default setup

(5) End meeting

Call：EasiioStopMeeting(const char* meeting_key, const char* resolution, const char* platform);

resolution: Normal resolution，maybe different from in meeting resolution

(6) Retrieve meeting participants

To get participant information

Call：EasiioGetParticipantInfo(const char* meeting_key, ParticipantInfo *participantinfo);

(7) Meeting status

Before start meeting or join meeting use this function to check meeting status.

Call：EasiioGetMeetingStatus(const char* meeting_key);

(8) Join meeting

Call：EasiioJoinMeeting(const char* meeting_key);

Call back：void(*onMakeCallResult)(int callId, const char* toNumber，const char* meetingkey);

Meetingkey: if meetingkey is return, it means join meeting successfully.

(9) Set video resolution

When leave the meeting, application may want to reset the video resolution to default value.

Call：EasiioSetVideoResolution(const char* resolution);

(10) Update meeting information

When leave meeting, app may call this function to update meeting info.

Call：EasiioUpdateJoinMeetingInfo(const char* meeting_key,const char* operate,const char* easiio_id);

(11) Meeting attendee leave meeting notification to lower layer(For multi-layer meeting)

Notify lower layer attendee to re-join meeting

Call：EasiioSendSwitchUpstreamRequest(int callId, int type);

2.3.1 Destroy CPP SDK

To complete destroy SDK call EasiioDestroy(); difference from logout，to Logout the account, call EasiioLogout.

2.3.2 CPP SDK Log

SDK will keep log file under the installed folder，For example EasiioLibraryLog-2016-03-14.log

Or call void EasiioSetLogDir(const char* logDir); to set log file path. The path has to be available or created by application.