互动直播

PC 端集成

产品文档

腾讯云

文档目录

PC 端集成
下载代码

最近更新时间：2018-06-15 18:33:15

互动直播 SDK for Windows，下文称为 ILiveSDK(Windows)，是互动直播在 Windows 平台上的 SDK。通过几个简单的接口调用即可实现互动直播、上麦和基础 IM 等功能。

下载 SDK 和 Demo

在 GitHub 上将项目 clone 下来。代其中包含了 ILiveSDK(Windows)，接口文档，头文件和接口调用示例等，具体如下。

文件	说明
/suixinbo	Demo 随心播工程源码，可以和其他平台随心播互通
/iLiveSDK	SDK 的头文件和库文件
/doc	接口文档和其他说明文档
suixinbo_run	可以直接运行的 Demo 程序

运行和体验 Demo

随心播是一款基于腾讯云互动直播 SDK 的秀场直播类应用。它演示了主播直播，观众连麦互动，消息互通，点赞打赏等业务场景的实现。

运行 Demo 程序

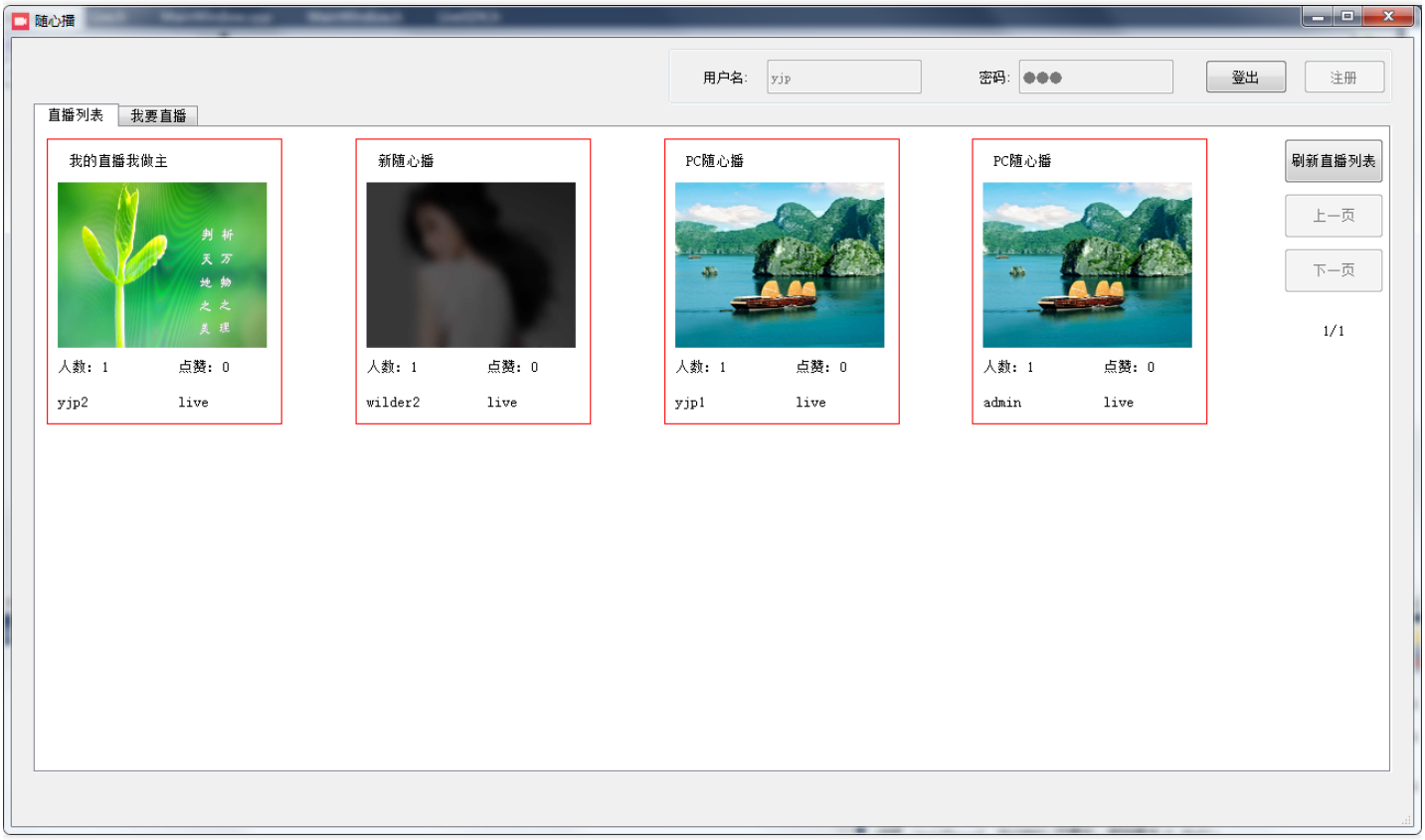
`suixinbo_run.zip` 为已经编译好的可执行包，解压后，直接双击 `suixinbo_Qt.exe` 即可运行。

注册和登录

使用其他功能前必须登录，单击右上角按钮可以进行登录和注册。

观看直播

选择直播列表 tab，然后刷新直播列表可以看到当前正在进行的直播。单击其中一个直播即可开始观看。

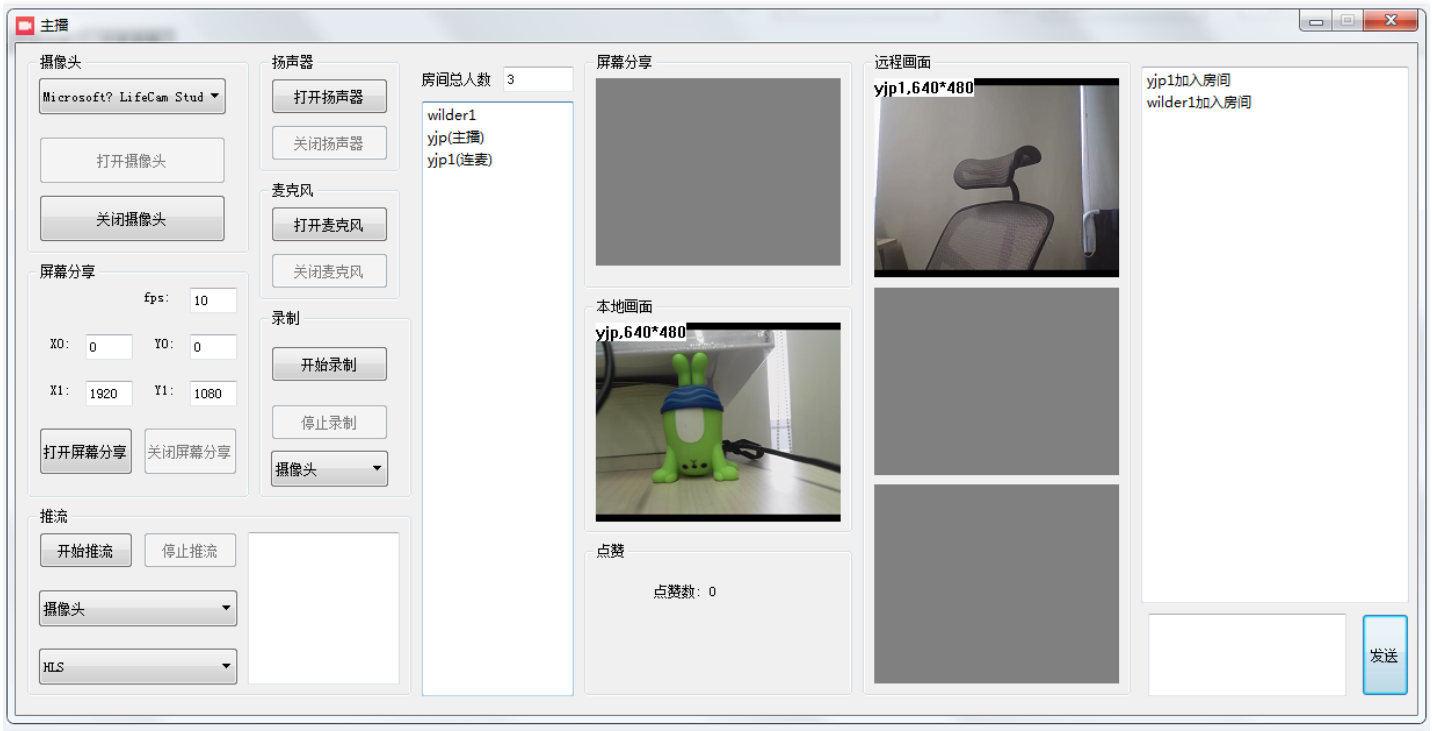


创建直播

选择我要直播 tab，单击按钮开始直播。

直播间操作

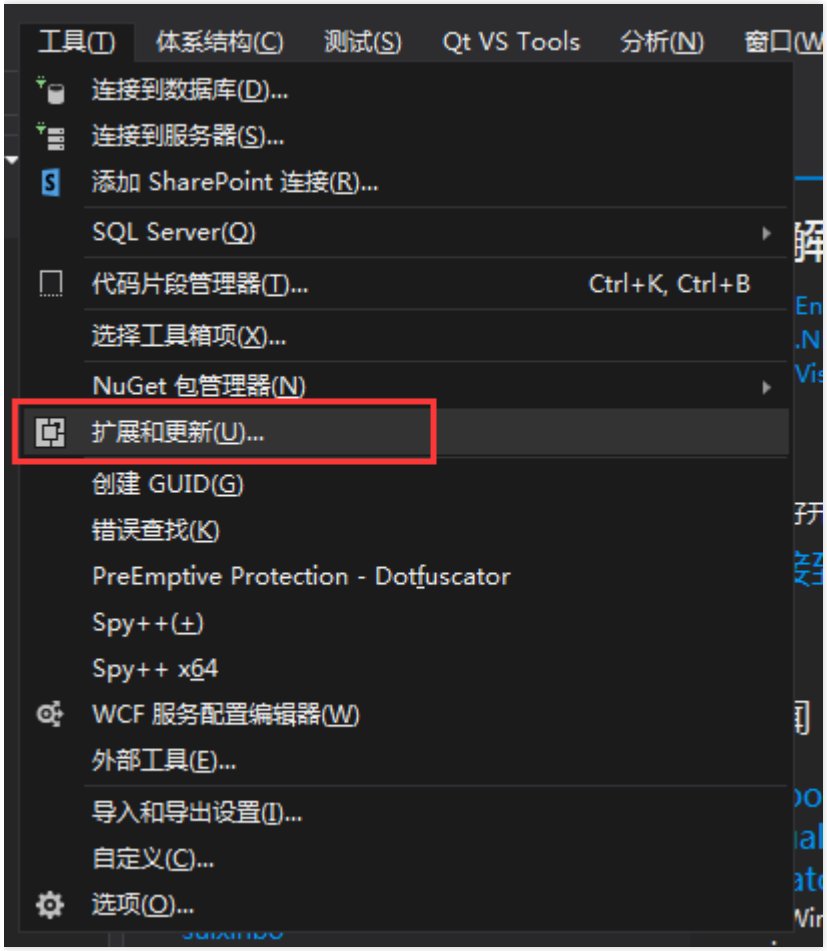
主播和观众可以在直播间内进行设备管理，推流录制，收发消息，连麦互动。



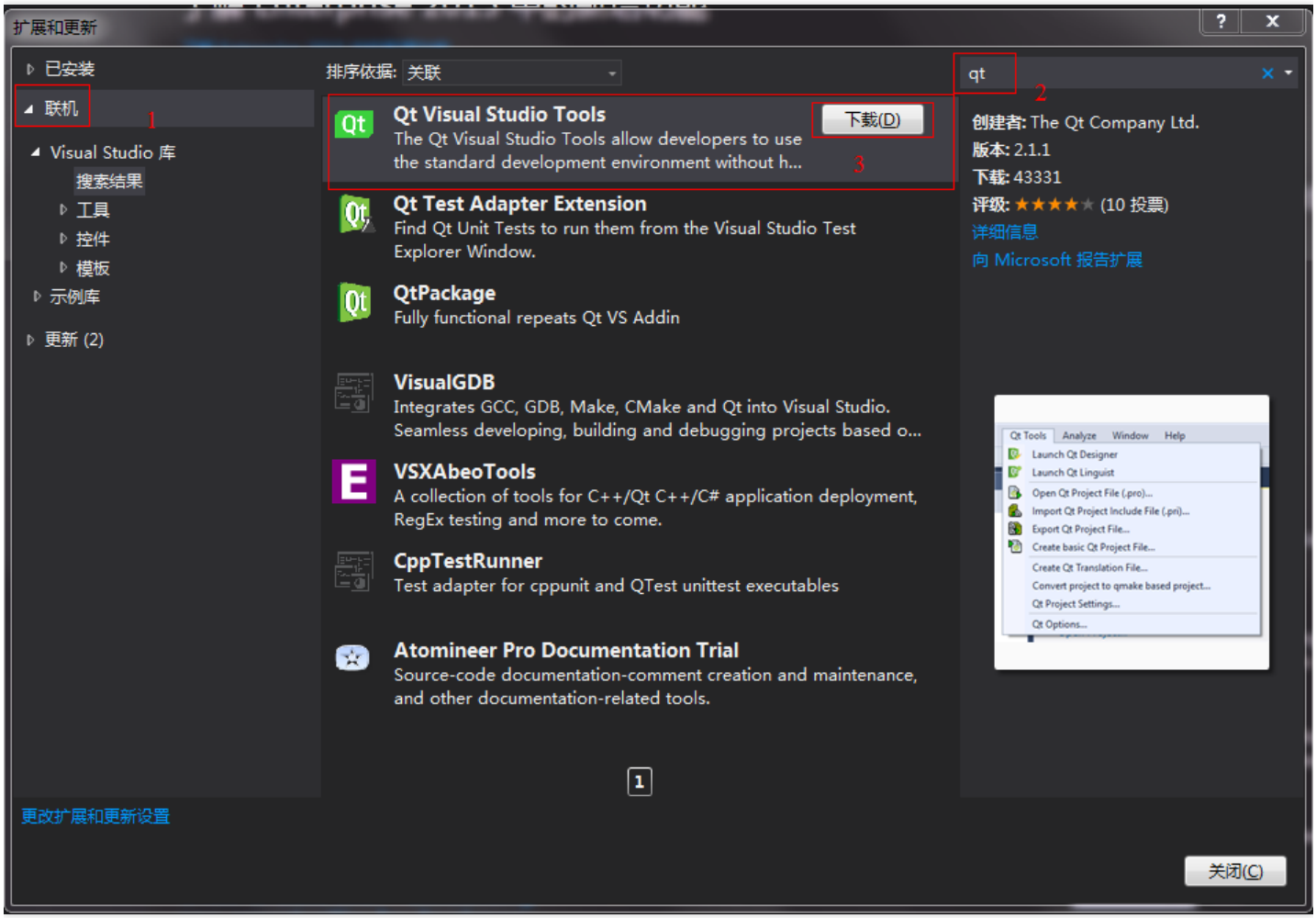
运行 Demo 工程

安装 Qt 环境

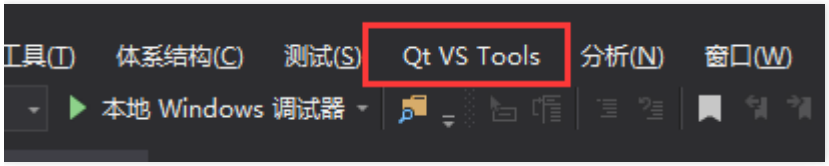
vs2010 之后的版本，可以直接在 VS 中进行安装，步骤：【工具】>【扩展和更新】。



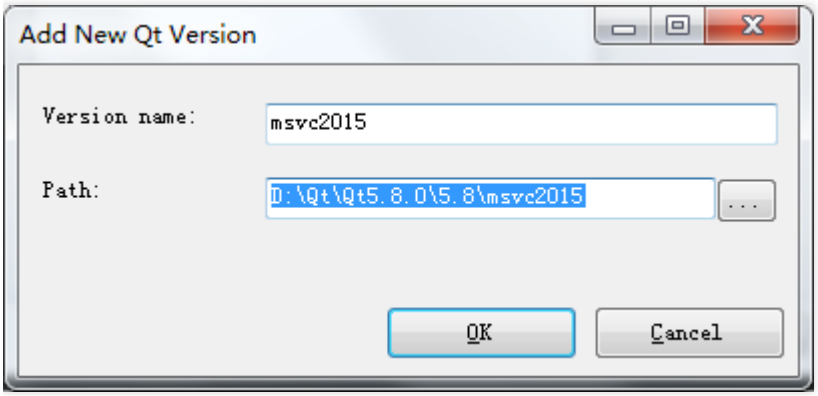
【联机】>【搜索 Qt】>【单击下载】。



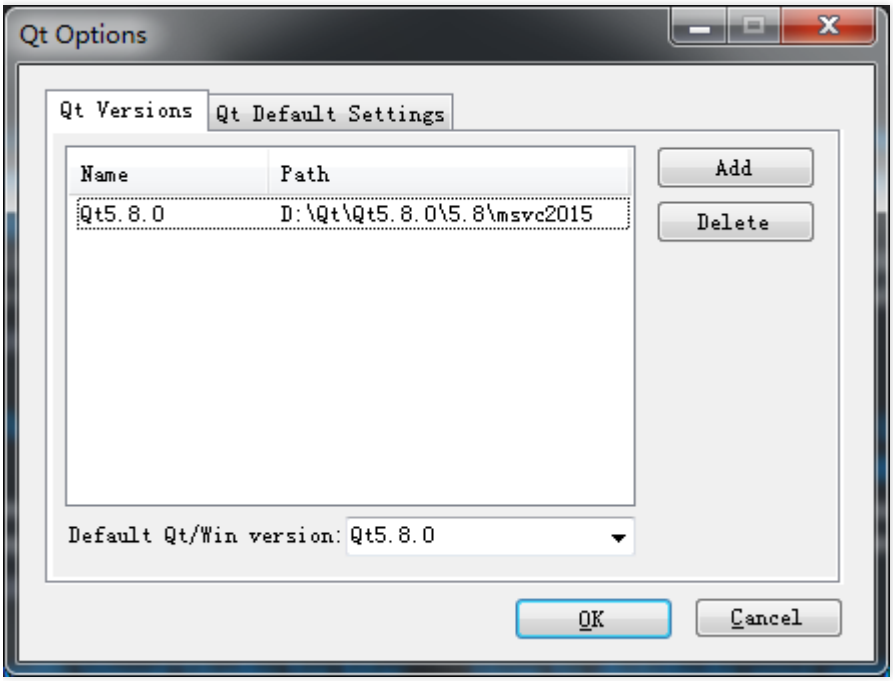
然后重启 vs，菜单栏中将会出现如下 Qt 菜单项。



此时，需要在 vs 中配置 Qt 目录，配置方法为【Qt VS Tools 菜单项】>【Qt Options】>【add】>【选择 Qt 安装目录】，如下图。



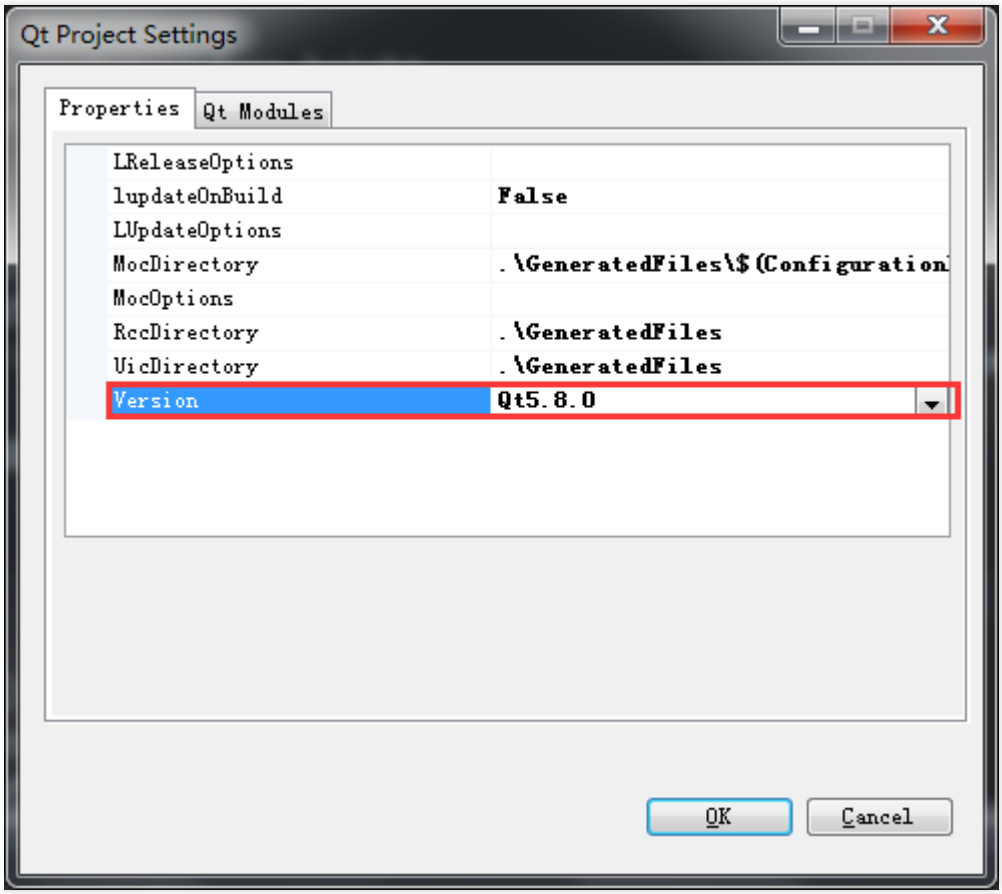
增加好后，单击确定，如下图。



注意：
VS2010 下，直接在 VS 中搜索找不到 Qt 插件，需要在 Qt 官网下载 Qt 插件，或 直接下载 Qt5.0.0 及 vs2010 插件 ，然后进行手动安装（先装 Qt，再装 Qt 插件）。

随心播项目编译

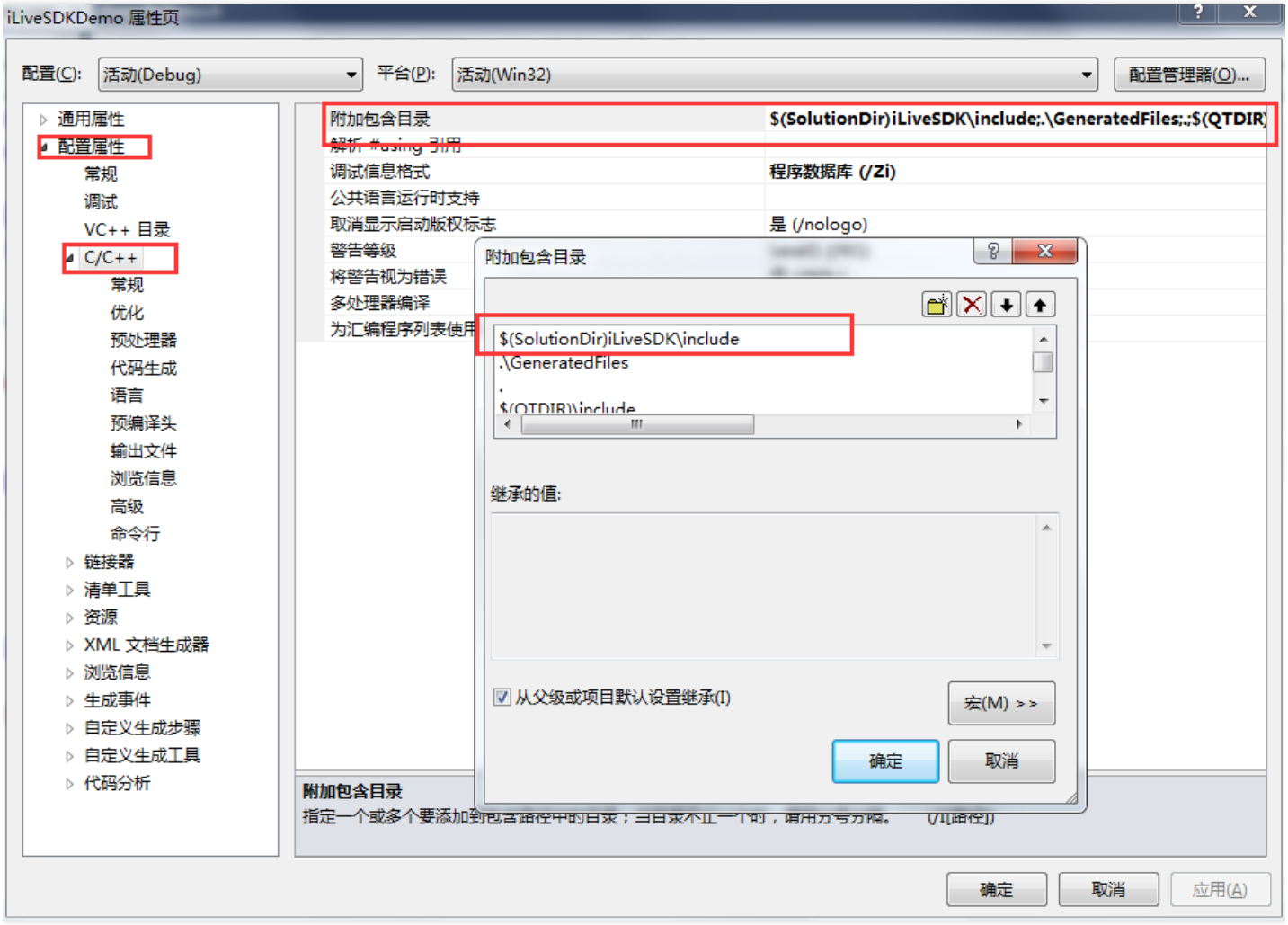
使用 vs 打开 suixinbo\sample 目录下的 suixinbo.sln，如果是 vs2010 之后的 vs 版本，会提示升级项目配置，单击"确定"。然后在项目 suixinbo_Qt 上【右键】>【Qt Project Settings】>【配置项目使用的 Qt 版本】，配置好后，即可编译运行随心播。如下图。



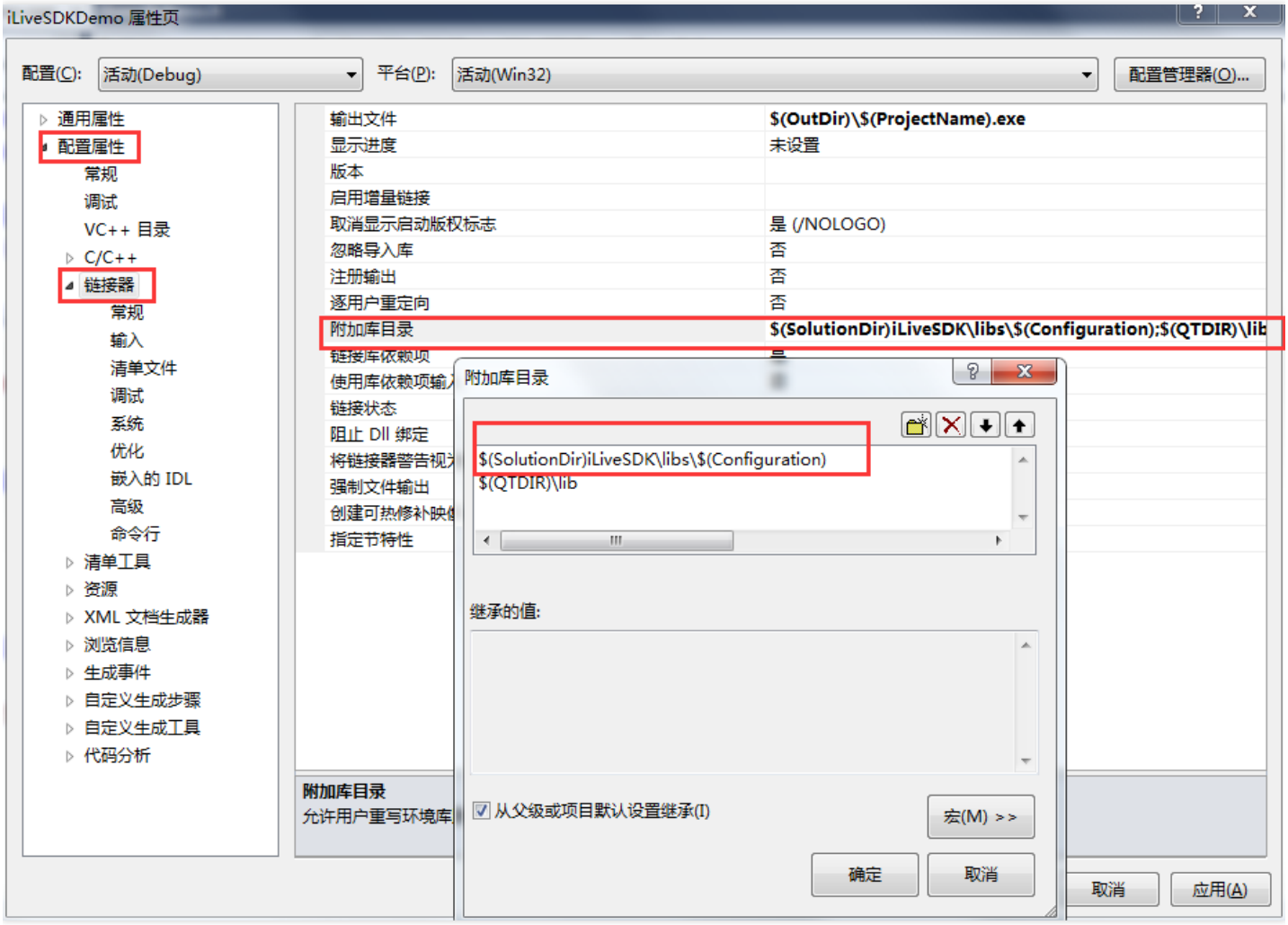
集成 iLive SDK 到自己的工程里

iLive SDK 在步骤一下载的 iLiveSDK 文件夹中，将 iLiveSDK 整个文件夹复制到解决方案文件(.sln 文件)所在的目录。

添加 include 目录：在项目的附加包含目录中添加 include 目录， $(SolutionDir)iLiveSDK\include，如下图。



添加库目录：在项目的附加库目录中添加 lib 文件所在目录，$(SolutionDir)iLiveSDK\libs\$(Configuration)，如下图。



包含头文件：在项目中包含头文件(通常是预编译头中),并使用相关命名空间，加载动态库的 lib 文件，代码如下。

```
#include  “ilive.h"
#pragma comment(lib, "ilivesdk.lib")
using namespace ilive;
```

拷贝 dll 文件到 exe 所在目录：将 libs\Debug 目录下的所有 dll 文件复制到项目的 Debug 版本运行目录下，libs\Release 目录下的所有 dll 文件复制到项目的 Release 版本运行目录下。

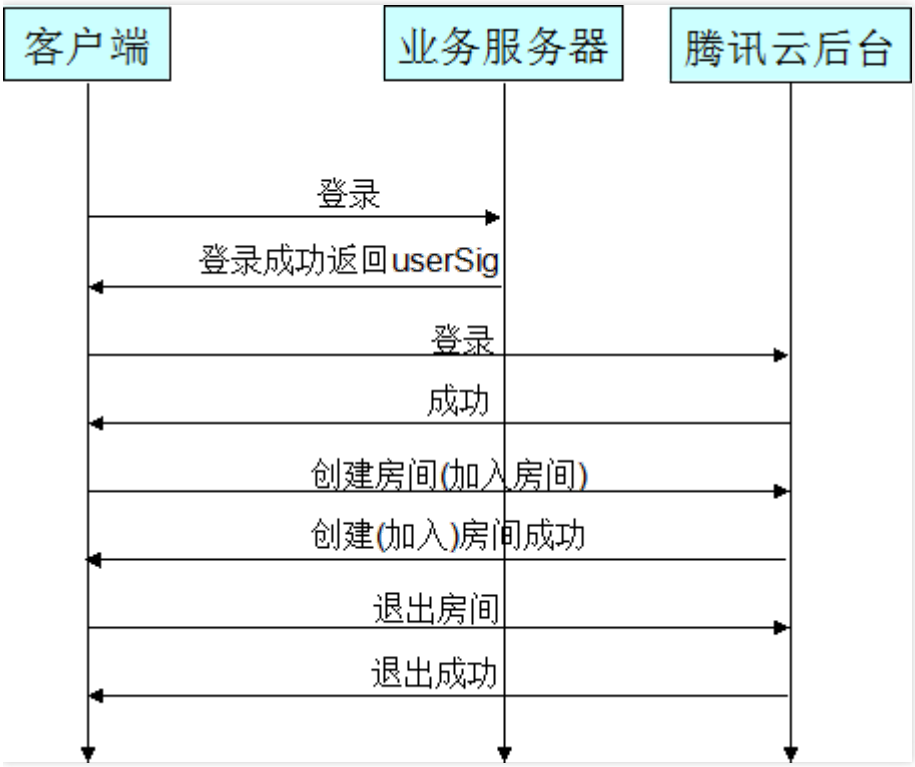
验证是否配置成功：调用 GetILive()->getVersion()，返回当前 iLiveSDK 的版本号。

直播接口

最近更新时间：2018-09-14 16:10:23

直播流程

iLiveSDK 中的音视频通讯能力被抽象为房间这个概念。在同一个房间内的成员可以看到房间内其他成员的音视频，每个房间同时最多可以有四路视频。iLiveSDK(Windows)提供了房间管理，音视频设备管理等功能，方便用户建立互动视频。具体的直播业务流程如下。



设置回调函数

iLiveSDK 通过回调将一些状态和通知告知上层，用户可以按照业务需要设置回调函数。

```
void onForceOffline()
{
//账号已经在其他地方登录
}
void OnMessage( const Message& msg )
{
```

```
}
void OnLocalVideo( LiveVideoFrame* video_frame, void* custom_data )
{
//video_frame 是本地画面每一帧的数据,用户需要显示本地画面时，在此回调函数中做渲染，渲染代码可参考随心播;
}
void OnRemoteVideo( LiveVideoFrame* video_frame, void* custom_data )
{
//video_frame 是远程画面每一帧的数据,用户需要显示远程画面时，在此回调函数中做渲染，渲染代码可参考随心播;
}
GetILive()->setForceOfflineCallback(onForceOffline); //设置被挤下线的通知函数;
GetILive()->setMessageCallBack(OnMessage); //设置消息的处理函数;
GetILive()->setLocalVideoCallBack(OnLocalVideo, NULL); //设置本地视频的回调函数;
GetILive()->setRemoteVideoCallBack(OnRemoteVideo, NULL); //设置远程视频的回调函数;
```

初始化

使用其他各项功能前必须将 iLiveSDK 初始化。

接口名	接口描述
init	iLiveSDK 内部初始化，告知 appId

参数类型	参数名	说明
int	appId	传入业务方 appid
int	accountType	传入业务方 accountType
bool	IMSupport	是否需要 IM 功能

示例：

```
int nRet = GetILive()->init(appid, AccountType);
if (nRet != NO_ERR)
{
//初始化失败
}
```

登录

用户在登录后才能使用消息通讯，互动视频等功能。登录需要填写用户 ID 和签名。其中签名是由腾讯登录服务提供的。

接口名	接口描述
login	独立模式登录到腾讯云后台

参数类型	参数名	说明
const char *	userId	用户在独立模式下注册的帐号
const char *	userSig	用户在业务方后台获取到的签名
iLiveSuccCallback	suc	登录成功回调
iLiveErrCallback	err	登录失败回调
void *	data	用户自定义的数据的指针，在成功和失败的回调函数中原封不动地返回

示例：

```
void OniLiveLoginSuccess( void* data )
{
//登录成功
}
void OniLiveLoginError( int code, const char *desc, void* data )
{
//登录失败
}
GetILive()->login(userId, userSig, OniLiveLoginSuccess, OniLiveLoginError, NULL);
```

音视频权限管理

业务层需重点关注房间成员的音视频权限。只有主播或者需要上麦的观众才能拥有音视频上行的权限。在进入或者创建房间时需要填写正确的权限。另外成员允许在房间内改变自己的权限。您可以在 腾讯云控制台配置 自身业务需要的角色及其权限，腾讯云服务器会根据房间成员不同的权限分配不同的 接入机。错误配置权限可能导致不必要的带宽支出和观众异常上行数据等问题。

创建房间

接口名	接口描述
createRoom	主播创建直播间

参数类型	参数名	说明
const iLiveRoomOption &	roomOption	主播创建房间时的配置项
iLiveSuccCallback	suc	创建房间成功回调
iLiveErrCallback	err	创建房间失败回调
void *	data	用户自定义的数据的指针，在成功和失败的回调函数中原封不动地返回

示例：

```
void OniLiveCreateRoomSuc( void* data )
{
//创建房间成功
}
void OniLiveCreateRoomErr( int code, const char *desc, void* data )
{
//创建房间失败
}
iLiveRoomOption roomOption;
roomOption.audioCategory = AUDIO_CATEGORY_MEDIA_PLAY_AND_RECORD;//互动直播场景
roomOption.roomId = 123456;
roomOption.controlRole = "LiveMaster";
roomOption.authBits = AUTH_BITS_DEFAULT;//注意权限管理
roomOption.roomDisconnectListener = OnRoomDisconnect;//自定义的房间断连的回调
roomOption.memberStatusListener = OnMemStatusChange;//自定义的成员状态变化回调
roomOption.data = this;
GetILive()->createRoom( roomOption, OniLiveCreateRoomSuc, OniLiveCreateRoomErr, this );
```

加入房间(观众)

接口名	接口描述
joinRoom	观众进入直播间

参数类型	参数名	说明
iLiveRoomOption	roomOption	观众进入房间时的配置项
SuccessCalllback	suc	加入房间成功回调
ErrorCallback	err	加入房间失败回调
void *	data	用户自定义的数据的指针，在成功和失败的回调函数中原封不动地返回

特别注意: 普通观众加入房间时，应该配置authBits无上行权限，仅观看权限。否则会和主播一样走核心机房DC，产生高额费用。

示例：

```
void OniLiveJoinRoomSuc( void* data )
{
//加入房间成功
}
void OniLiveJoinRoomErr( int code, const std::string& desc, void* data )
{
//加入房间失败
}
iLiveRoomOption roomOption;
roomOption.audioCategory = AUDIO_CATEGORY_MEDIA_PLAY_AND_RECORD;//互动直播场景
roomOption.roomId = 123456;
roomOption.controlRole = "Guest";
roomOption.authBits = AUTH_BITS_JOIN_ROOM|AUTH_BITS_RECV_AUDIO|AUTH_BITS_RECV_CAMERA
_VIDEO|AUTH_BITS_RECV_SCREEN_VIDEO;//注意权限管理
roomOption.memberStatusListener = Live::OnMemStatusChange;//自定义的成员状态变化回调
roomOption.roomDisconnectListener = Live::OnRoomDisconnect;//自定义的房间断连的回调
roomOption.data = g_pMainWindow->getLiveView();
GetILive()->joinRoom( roomOption, OniLiveJoinRoomSuc, OniLiveJoinRoomErr, this );
```

互动消息和上麦接口

最近更新时间：2018-06-15 18:33:24

互动消息

iLiveSDK(Windows)提供了消息通讯的功能。基于消息通讯可以实现房间内成员的群消息，两个用户之间的 C2C 消息（不用加好友）。当前 SDK 只支持发文本消息和自定义消息。

发送 C2C 消息

C2C 消息指的是两个用户之间的点对点聊天消息。邀请房间成员上麦，点赞等消息可以在业务层通过 C2C 自定义消息实现。具体的实现可以参考 Demo。

接口名	接口描述
sendC2CMessage	发送 C2C 消息

参数类型	参数名	说明
const char *	dstUser	接收方 ID
const Message &	message	IM 消息内容
iLiveSuccCallback	suc	发送消息成功回调
iLiveErrCallback	err	发送消息失败回调
void*	data	用户自定义数据的指针，在成功和失败的回调函数中原封不动地返回

示例：

```
void OnSendC2CMsgSuc( void* data )
{
//发送 C2C 消息成功
}
void OnSendC2CMsgErr( int code, const char *desc, void* data )
{
//发送 C2C 消息失败
}
Message message;
MessageTextElem *elem = new MessageTextElem("hello");
```

```
message.elems.push_back(elem);
GetILive()->sendC2CMessage( message, OnSendC2CMsgSuc, OnSendC2CMsgErr, NULL );
```

发送群消息

当前仅支持给当前所在房间发群消息，房间内其他成员都会收到该消息。

接口名	接口描述
sendGroupMessage	发送群消息

参数类型	参数名	说明
const Message &	message	消息内容
iLiveSuccCallback	suc	发送消息成功回调
iLiveErrCallback	err	发送消息失败回调
void*	data	用户自定义数据的指针，在成功和失败的回调函数中原封不动地返回

示例：

```
void OnSendGroupMsgSuc( void* data )
{
//发送群消息成功
}
void OnSendGroupMsgErr( int code, const char *desc, void* data )
{
//发送群消息失败
}
Message message;
MessageTextElem *elem = new MessageTextElem("hello");
message.elems.push_back(elem);
GetILive()->sendGroupMessage( message, OnSendGroupMsgSuc, OnSendGroupMsgErr, NULL );
```

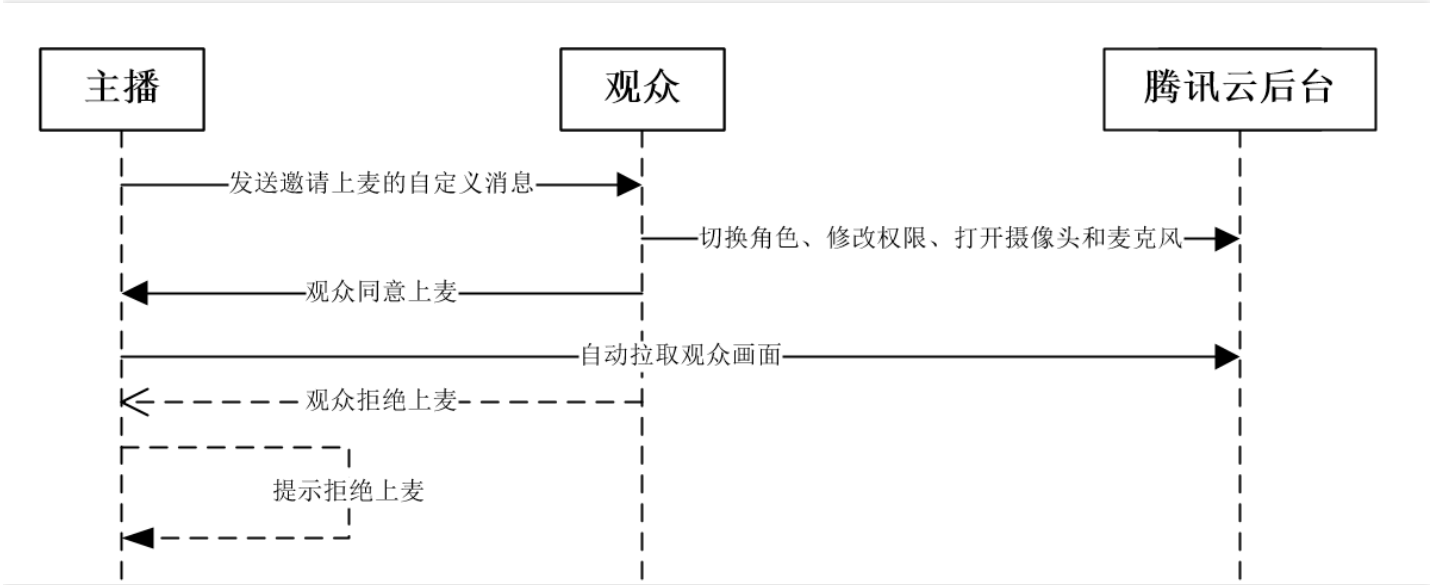
接收消息

设置消息监听回调后，SDK 每次收到消息都会通知上层。

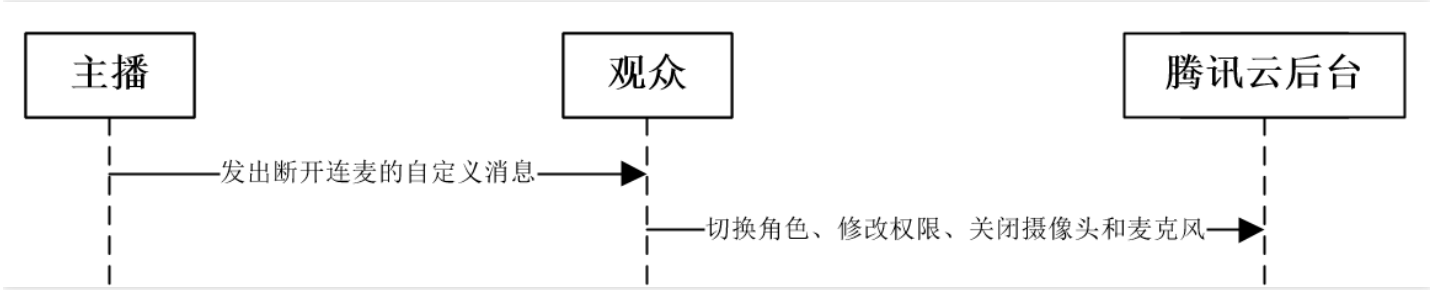
```
void OnMessage(const Message& message)
{
std::string szSender = message.sender.c_str();
for (size_t i = 0; i < message.elems.size(); ++i)
{
MessageElem *pElem = message.elems[i];
switch( pElem->type )
{
case TEXT:
{
MessageTextElem *elem = static_cast<MessageTextElem*>(pElem);
std::cout << elem->content.c_str() << std::endl;
break;
}
case CUSTOM:
{
MessageCustomElem *elem = static_cast<MessageCustomElem*>(pElem);
std::string szDate = elem->data.c_str();
break;
}
default:
break;
}
}
}
GetILive()->setMessageCallBack(OnMessage, NULL);
```

连麦互动

主播邀请上麦流程



主播断开连麦观众流程



接口

iLiveSDK 未对上/下麦进行封装，用户可以参考随心播的 sendC2CCustomCmd() 和 sendGroupCustomCmd() 函数，发送自定义消息作为邀请上麦和接受邀请的信令；观众上麦和下麦，需要切换用户角色和修改用户权限。

切换角色：

接口名	接口描述
changeRole	更改角色

参数类型	参数名	说明
const char *	szControlRole	角色字符串(由用户 App 的控制台配置)
iLiveSuccCallback	suc	成功的回调函数

参数类型	参数名	说明
iLiveErrCallback	err	失败的回调函数
void*	data	用户自定义数据的指针，在成功和失败的回调函数中原封不动地返回

示例：

```
void OnChangeRoleSuc( void* data )
{
//切换角色成功
}
void OnChangeRoleErr( int code, const char *desc, void* data )
{
//切换角色失败
}
GetILive()->changeRole(Role, OnChangeRoleSuc, OnChangeRoleErr, NULL);
```