# SDK运行

## 1.1 设置三方库

注：此方式使用已经编译好的第三方库进行编译，通过bat脚本将三方库解压到指定路径中。

路径需要保持**纯英文路径**，有字母构成。其他符号最好也不要出现。

1.打开scripts文件夹中的build\_windows\_64\_prebuild.bat，修改解压缩工具WinRAR（set winRar="C:\Program Files\WinRAR\WinRAR.exe"）为自己电脑WinRAR路径

2.运行 build\_windows\_64\_prebuild.bat。此批处理为了解压各第三方库，并把依赖头文件移动到合适位置

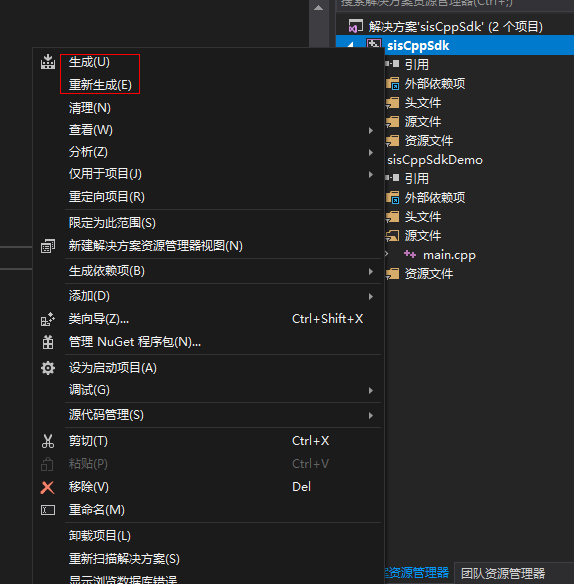
备注：如果使用x86，该脚本依然要运行。

## 1.2 编译源码

使用vs2017及以上版本打开sisCppSdk文件夹下的sisCppSdk.sln，在解决方案上sisCppSdk右键生成。以x64为例，x86生成在类似的路径下，可以在vs中通过属性查看生成路径。

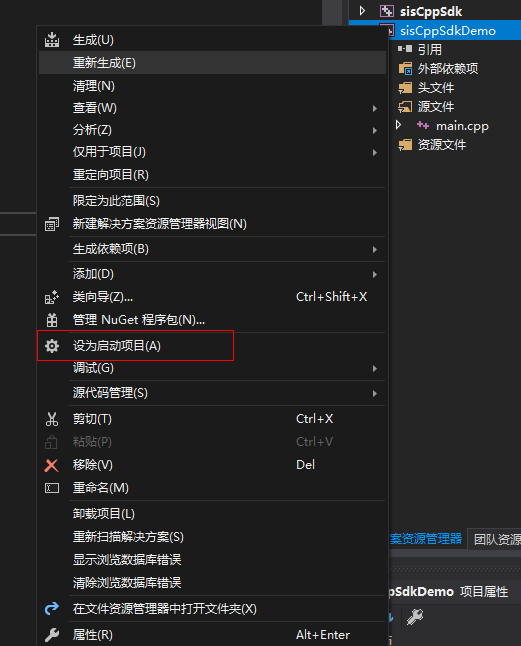
Debug生成物路径: {ProjectRoot}\build\build\_win64\SisCppSdk\x64\Debug

Release生成物路径: {ProjectRoot}\build\build\_win64\SisCppSdk\x64\Release



## 1.3 运行demo

1. 右单击sisCppSdkDemo，设为启动项目，如下图所示。



2. main cpp配置相关鉴权信息，运行即可。具体参数可参考二、使用示例。

# 二、集成SDK（可选）

备注：如果有需要**新建工程集成SDK**，则需要进行一些配置。否则在已有工程SisCppSdk的基础上修改即可。

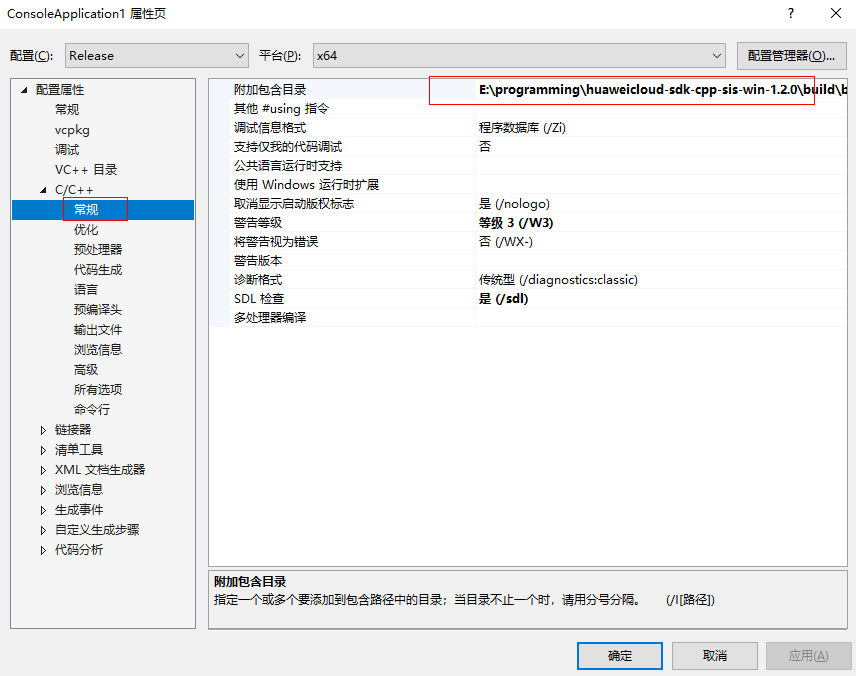
## 2.1 前置条件

1. 已完成1.1、1.2的操作。即运行bat脚本和sisCppSdk项目运行生成。

2. 1.3运行demo中能成功运行。

## 2.2 设置include路径（x64）

1. 设置入口：右单击项目 –> 属性 -> 配置属性 –> C/C++ -> 常规 –> 附加包含目录，配置include目录



2. 配置include路径：${PROJECT\_DIR}为huaweicloud-sdk-cpp-sis-win-1.2.0路径。运行bat脚本后，会自动生成build目录。

依此添加以下目录（jsoncpp、websocketpp、openssl、boost、sisCppSdk）

${PROJECT\_DIR} /build/build\_win64/thirdparty/jsoncpp-prefix/include

${PROJECT\_DIR}/build/build\_win64/thirdparty/websocketpp-prefix/websocketpp-master/websocketpp-master

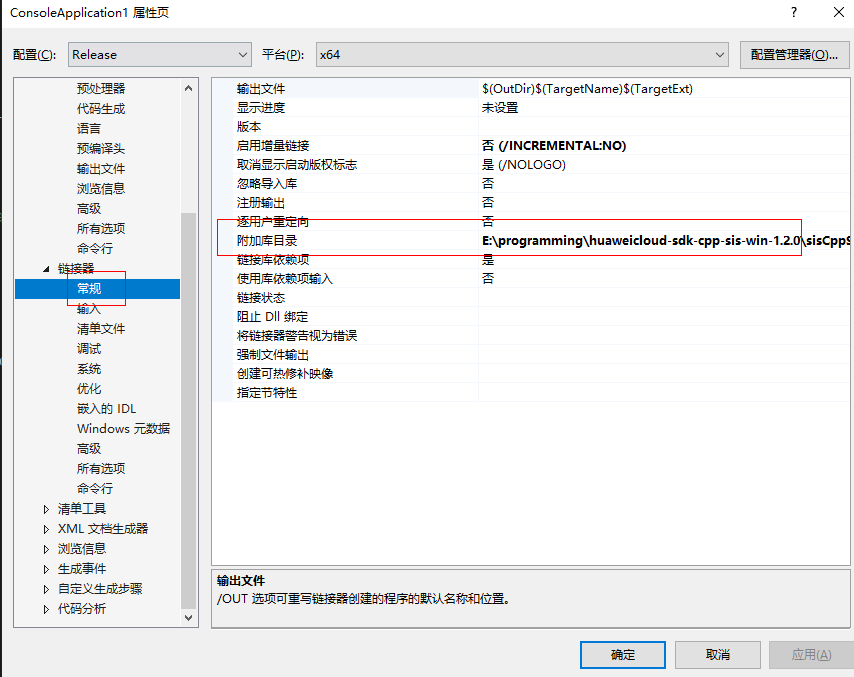
${PROJECT\_DIR}/build/build\_win64/thirdparty/openssl-prefix/include

${PROJECT\_DIR}/build/build\_win64/thirdparty/boost-prefix/boost\_1\_77\_0/boost\_1\_77\_0

${PROJECT\_DIR}/Include

## 2.3 配置library路径（x64）

1. 配置lib目录：右单击项目 –> 属性-> 配置属性 –> 链接器 -> 常规 –> 附加库目录， 配置lib目录



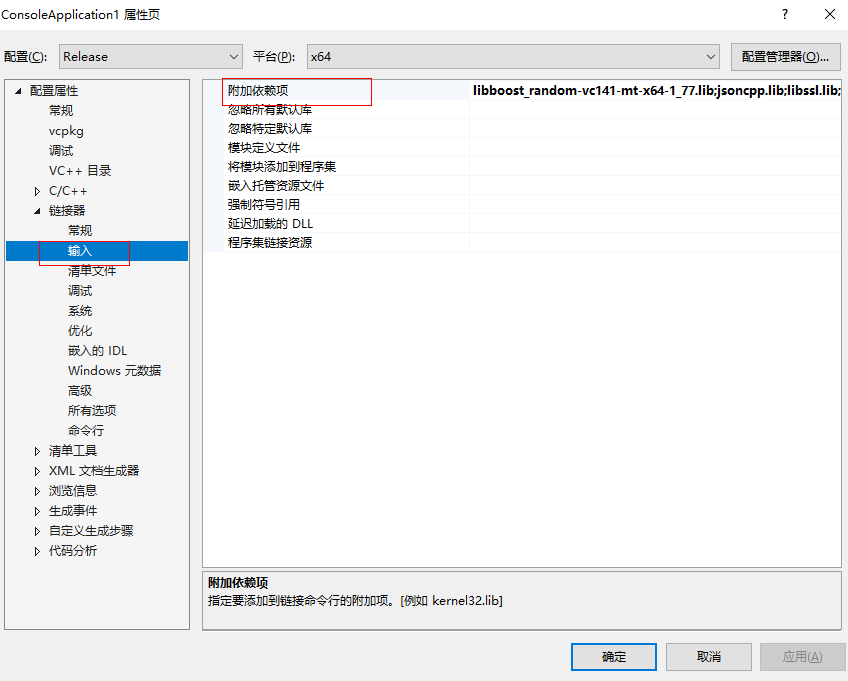
2. 依此配置如下lib目录：${PROJECT\_DIR}为huaweicloud-sdk-cpp-sis-win-1.2.0路径。（三方lib和sis sdk的lib）

${PROJECT\_DIR}/sisCppSdk/libs/x64/Release

${PROJECT\_DIR}/build/build\_win64/SisCppSdk/x64/Release

## 2.4 配置lib取值（x64）

1. 右单击项目 –> 属性-> 配置属性 –> 链接器 -> 输入 –> 附加依赖项， 配置需要用到的lib库，如下所示：



2. 依此添加如下库文件：

libboost\_random-vc141-mt-gd-x64-1\_77.lib

jsoncpp.lib

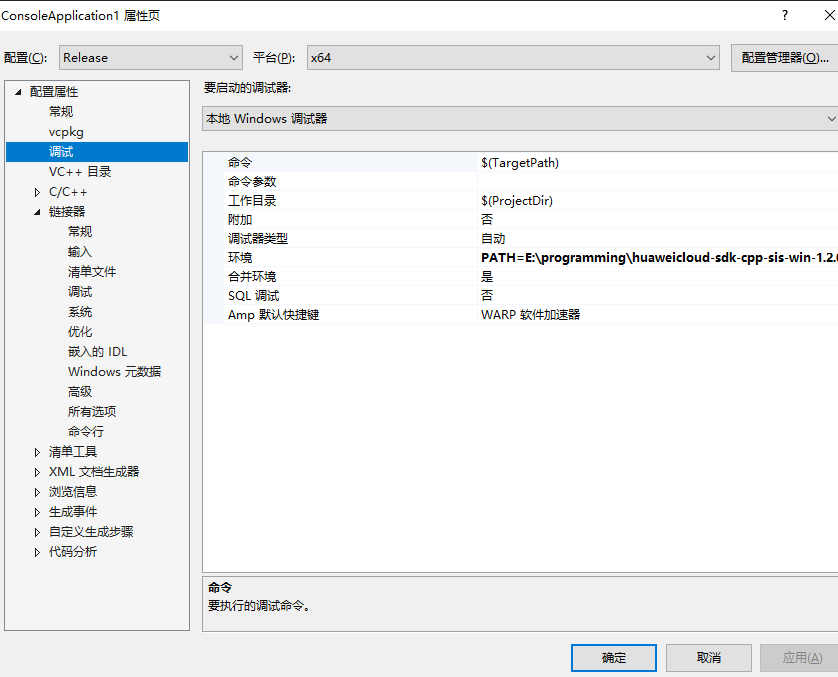
libssl.lib

libcrypto.lib

sisCppSdk.lib

## 2.5 配置dll路径（x64）

1. 右单击项目 –> 属性-> 配置属性 –> 调试 -> 环境，增加PATH=${PROJECT\_DIR}/sisCppSdk/libs/x64/Release。${PROJECT\_DIR}为huaweicloud-sdk-cpp-sis-win-1.2.0路径。或者也可用直接把目录中的dll文件移到工程中。



## 2.5 x86使用

X86使用和x64类似 ，可参考sisCppSdkDemo的x86属性配置.

备注：${PROJECT\_DIR}为huaweicloud-sdk-cpp-sis-win-1.2.0路径。

1. 配置include路径：按照2.2方法配置include路径

${PROJECT\_DIR}/Include

${PROJECT\_DIR}/ thirdparty/x86\_include

1. 配置library路径：按照2.3方法配置library路径

${PROJECT\_DIR}/build/build\_win32/SisCppSdk/Win32/Release

${PROJECT\_DIR}/sisCppSdk/libs/x86/Release

1. 配置lib取值：按照2.4方法配置lib取值

jsoncpp.lib

libcrypto.lib

libssl.lib

boost\_random-vc140-mt.lib

sisCppSdk.lib

1. 配置dll路径：按照2.5方法配置dll路径，或者也可用直接把dll复制在工程中

PATH=${PROJECT\_DIR}/sisCppSdk/libs/x86/Release

# 三、使用示例

## 3.1 使用rasr示例（实时语音识别）

1. 填写鉴权相关信息：ak、sk、region、projectId。
   1. Ak/sk可参考<https://support.huaweicloud.com/api-sis/sis_03_0051.html>
   2. region可参考<https://support.huaweicloud.com/api-sis/sis_03_0004.html>，建议cn-east-3。
   3. projectId可参考<https://support.huaweicloud.com/api-sis/sis_03_0008.html>，在“我的凭证”页面查找
2. 填写http相关超时信息
3. 设置回调函数，websocket连接响应通过回调函数处理，如连接成功，连接失败，收到事件，收到响应，收到start请求、收到end请求等。如果不设置默认使用RasrListener。
4. 设置发送音频相关参数，具体可参考api文档<https://support.huaweicloud.com/api-sis/sis_03_0030.html>
5. 依此进行如下操作：
   1. 建立连接
   2. 发送start请求
   3. 发送音频二进制文件
   4. 发送end请求
   5. 关闭连接

#include <iostream>

#include "RasrClient.h"

#include "RasrRequest.h"

#include "IoUtil.h"

void OnRasrConnect() {

std::cout << "now rasr client Connect success" << std::endl;

}

void OnRasrStart(std::string text) {

std::cout << "now rasr client receive start response: " << text << std::endl;

}

void OnRasrResp(std::string text) {

// text encoded by utf-8 contains chinese character, which will cause error code. So we should convert to ansi

// cout << "rasr receive " << text << endl;

std::cout << "now rasr client receive " << Utf8ToAnsi(text) << std::endl;

}

void OnRasrEnd(std::string text) {

std::cout << "now rasr client receive end response: " << text << std::endl;

}

void OnRasrClose() {

std::cout << "now rasr client receive Close" << std::endl;

}

void OnRasrError(std::string text) {

std::cout << "now rasr client receive error: " << text << std::endl;

}

void OnRasrEvent(std::string text) {

std::cout << "now rasr client receive event: " << text << std::endl;

}

void RasrTest() {

// 1. config parameter

// 1.1 init authInfo

// 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；

// 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD\_SDK\_AK和HUAWEICLOUD\_SDK\_SK

std::string ak = GetEnv("HUAWEICLOUD\_SDK\_AK");

std::string sk = GetEnv("HUAWEICLOUD\_SDK\_SK");

std::string region = "";

std::string projectId = "";

AuthInfo authInfo(ak, sk, region, projectId);

// 1.2 config Connect parameter

HttpConfig httpConfig;

httpConfig.SetReadTimeout(20000);

httpConfig.SetConnectTimeout(20000);

// 1.3 config callback, callback function are optional, if not set, it will use function in RasrListener

WebsocketService::ptr websocketServicePtr = websocketpp::lib::make\_shared<WebsocketService>();

websocketServicePtr->SetOnConnectFunc(OnRasrConnect); // Connect success callback

websocketServicePtr->SetOnStartFunc(OnRasrStart); // receive start response callback

websocketServicePtr->SetOnRespFunc(OnRasrResp); // receive transcribe result callback

websocketServicePtr->SetOnEndFunc(OnRasrEnd); // receive end response callback

websocketServicePtr->SetOnCloseFunc(OnRasrClose); // Close callback

websocketServicePtr->SetOnEventFunc(OnRasrEvent); // receive event callback

websocketServicePtr->SetOnErrorFunc(OnRasrError); // receive error callback

// 1.3 option, use RasrListener, You can edit RasrListener.h to finish your own business

// WebsocketService::ptr websocketServicePtr = websocketpp::lib::make\_shared<WebsocketService>();

// RasrListener rasrListener;

// websocketServicePtr->SetRasrListener(rasrListener);

// 1.4 config request parameter

RasrRequest request("pcm16k16bit", "chinese\_16k\_general");

request.SetIntermediateResult("no");

// 2. init client

RasrClient\* rasrClient = new RasrClient(authInfo, websocketServicePtr, httpConfig);

// 3. create connection :ContinueStreamConnect/ShortStreamConnect/SentenceStreamConnect

rasrClient->ContinueStreamConnect();

// 4. send start

rasrClient->SendStart(request);

// 5. send binary audio. (filePtr, fileLength, byteLen, SleepTime ). If the audio is generated by recording, then it should set sleep time 0.

int fileLength;

std::string filePath = "../sisCppSdkDemo/16k\_pcm.wav";

unsigned char\* buff = ReadBinary(filePath, &fileLength);

if (buff == nullptr) {

std::cout << filePath << " read file failed";

rasrClient->Close();

delete rasrClient;

return;

}

rasrClient->SendBinary(buff, fileLength, 3200, 50);

// 6. send end

rasrClient->SendEnd();

// 7. close

rasrClient->Close();

delete[] buff;

delete rasrClient;

}

int main() {

RasrTest();

return 0;

}

## 3.2 使用rtts示例（实时语音合成）

1. 填写鉴权相关信息：ak、sk、region、projectId。
   1. Ak/sk可参考<https://support.huaweicloud.com/api-sis/sis_03_0051.html>
   2. region可参考<https://support.huaweicloud.com/api-sis/sis_03_0004.html>，建议cn-east-3。
   3. projectId可参考<https://support.huaweicloud.com/api-sis/sis_03_0008.html>，在“我的凭证”页面查找
2. 填写http相关超时信息
3. 设置回调函数，websocket连接响应通过回调函数处理，如连接成功，连接失败，收到响应，收到start请求、收到end请求等。如果不设置默认使用RttsListener
4. 设置发送音频相关参数，具体可参考api文档 <https://support.huaweicloud.com/api-sis/sis_03_0115.html> 。可选设置发音人、采样率，音高等参数。
5. 调用Synthesis(RttsRequest request)方法进行语音合成。

#include <iostream>

#include "IoUtil.h"

#include "RttsClient.h"

#include "RttsRequest.h"

void OnRttsConnect() {

std::cout << "now rtts client Connect success" << std::endl;

}

void OnRttsStart(std::string text) {

std::cout << "now rtts client receive start response: " << text << std::endl;

}

void OnRttsEnd(std::string text) {

std::cout << "now rtts client receive end response: " << text << std::endl;

}

void OnRttsClose() {

std::cout << "now rtts client receive Close" << std::endl;

}

void OnRttsError(std::string text) {

std::cout << "now rtts client receive error: " << text << std::endl;

}

void OnRttsBinary(std::string binaryData) {

// data content can be avaliable by data() method, data length can be avaliable by size() method

// const char\* data = binaryData.data();

// int dataLength = binaryData.size();

std::cout << "now rtts client receive binary data " << binaryData.size() << std::endl;

}

void RttsTest() {

// 1. config parameter

// 1.1 init authInfo

// 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；

// 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD\_SDK\_AK和HUAWEICLOUD\_SDK\_SK

std::string ak = GetEnv("HUAWEICLOUD\_SDK\_AK");

std::string sk = GetEnv("HUAWEICLOUD\_SDK\_SK");

std::string region = "";

std::string projectId = "";

AuthInfo authInfo(ak, sk, region, projectId);

// 1.2 config Connect parameter

HttpConfig httpConfig;

httpConfig.SetReadTimeout(20000);

httpConfig.SetConnectTimeout(20000);

// 1.3 config callback, callback function are optional, if not set, it will use function in RttsListner

WebsocketService::ptr websocketServicePtr = websocketpp::lib::make\_shared<WebsocketService>();

websocketServicePtr->SetOnConnectFunc(OnRttsConnect); // Connect success callback

websocketServicePtr->SetOnStartFunc(OnRttsStart); // receive start response callback

websocketServicePtr->SetOnEndFunc(OnRttsEnd); // receive end response callback

websocketServicePtr->SetOnCloseFunc(OnRttsClose); // Close callback

websocketServicePtr->SetOnErrorFunc(OnRttsError); // receive error callback

websocketServicePtr->SetOnBinaryFunc(OnRttsBinary); // receive binary callback

// 1.3 option, use RttsListener, which can save file; You can edit RttsListener.h to finish your own business

//WebsocketService::ptr websocketServicePtr = websocketpp::lib::make\_shared<WebsocketService>();

//RttsListener rttsListener;

//rttsListener.SetSaved(true);

//rttsListener.SetFilePath("d:/test.pcm");

//websocketServicePtr->SetRttsListener(rttsListener);

// 1.4 config request parameter

std::string text = AnsiToUtf8("华为致力于把数字世界带入每个人每个家庭每个组织，构建万物互联的智能世界。");

RttsRequest request(text);

request.SetAudioFormat("pcm");

request.SetVolume(100);

request.SetSpeed(0);

request.SetPitch(0);

request.SetSampleRate("8000");

request.SetAudioProperty("chinese\_huaxiaowei\_common");

// 2. init client

RttsClient\* rttsClient = new RttsClient(authInfo, websocketServicePtr, httpConfig);

// 3. send request

rttsClient->Synthesis(request);

// wait for save file, if setSaved false in rttsListener or don't use rttsListener, it can be removed.

std::this\_thread::sleep\_for(std::chrono::milliseconds(2000));

delete rttsClient;

}

# 四、常见问题

1. 出现报错IntelliSense相关？

解决方案：右单击项目 –> 属性 –> 文本编辑器 –> c/C++ –> 高级 –> 禁用IntelliSense设置为True