# How to use C++library develop Android App

# Purpose:

How to develop Android application using official released lib.so as a 3rd part library.

# How to do it:

1. Acquire your library libXXX.so

Make sure you have gdb info file gdb.setup and gdbserver which can be used for internal debugging.

1. Set-up your Android program

Write your own Android program using JAVA as normal. Since libXXX.so is implemented using C++, you need eclipse + cdt + ndk + jni installed and configured.

For how to set-up ndk+jni environment please see details in <https://internal.autodesk360beta.com/blog/permalink/BL46601QT150b7b805d1b607ef7389bbad28>

1. Call libXXX.so as 3rd part library

3.1 Android calling 3rd part library architecture

Android App Java Code

System.loadLibrary("Wrapper");

C++ libWrapper.so

Call API directly using JNI

Released libXXX.so

3.2 Write C++ wrapper class

3.2.1  Generate.h header file:

workspace/android/NDK/hello-jni$ javah -classpath bin -d jni com.example.wrapper.wrapper

-classpath bin：indicate class path  
 -d jni: indicate the folder that generated .h files was

3.2.2 Generate .c file manually

Jstring Java\_com\_example\_hellojni\_wrapper\_Func( JNIEnv\* env, jobject thiz )

{

return (\*env)-**>**NewStringUTF(env, "Wrapper !");

}

3.2.3 Implement your own logic codes to call these functions.

3.3 Build libWrapper.so

Android system use Android.mk and Application.mk as makefiles.

Android.mk is used to generate .apk package:

LOCAL\_PATH := $(call my-dir)

# Clear variables here.

include $(CLEAR\_VARS)

LOCAL\_MODULE := XXX

LOCAL\_SRC\_FILES := libXXX.so

LOCAL\_EXPORT\_C\_INCLUDES := $(LOCAL\_PATH)

include $(PREBUILT\_SHARED\_LIBRARY)

include $(CLEAR\_VARS)

LOCAL\_MODULE := square

LOCAL\_SRC\_FILES := xxxwrapper.c \

wrapperImplCpp.cpp

LOCAL\_SHARED\_LIBRARY := XXX

LOCAL\_CFLAGS := -DANDROID\_NDK

LOCAL\_LDFLAGS += $(LOCAL\_PATH)/libxxx.so

include $(BUILD\_SHARED\_LIBRARY)

Application.mk is used to build C/C++ code to libWrapper.so:

APP\_MODULES := wrapper

APP\_STL := stlport\_static

APP\_OPTIM := release

include $(BUILD\_SHARED\_LIBRARY)

3.4 Write code by Android App side

Simply call libWrapper.so in JAVA code using

System.loadLibrary("Wrapper");

# Concerns:

1. Android version compatible problem

Only tested in Android 4.3 + ndk r19

2. Remaining warnings

Libxxx.so is built using Android develop package (cdt+gcc+lm…) so there are still some gcc-only warnings.