Android: NDK中的Android.mk和Application.mk

1. 简介

Android.mk 可用来描述要编译的某个具体模块的相关信息。比如：指定编译该模块时所需要的源文件、编译该模块时要链接的库文件、该模块编译完成后生成的库的名字等等。
Application.mk 可用来描述整个应用程序编译时的相关信息。比如：指定编译支持的ABI平台(armeabi-V7a, arm64-v8a, all...)、指定NDK编译时的模块列表、指定目标Android 平台的名称等等。

2. 一个简单的Android.mk文件

```
LOCAL_PATH := $(call my-dir)

include $(CLEAR_VARS)
LOCAL_MODULE    := hello-jni
LOCAL_SRC_FILES := hello-jni.c  jnihelp.cpp Calculator.cpp

include $(BUILD_SHARED_LIBRARY)
```

`LOCAL_PATH`： 必须位于Android.mk文件的开始，用来定义源文件的位置，$(call my-dir)返回当前路径，即android.mk所在的目录。my-dir是构建系统提供的宏函数，它将会返回当前目录的路径；
`include $(CLEAR_VARS)`： 清除除LOCAL_PATH之外的所有LOCAL_XXX变量。这个清理动作是必须的，因为所有的编译控制文件都是由同一个GNU Make解析和执行，其变量是全局的，只有清理后才能避免相互影响。
`LOCAL_MODULE`： 表示Android.mk所在模块的模块名，名字必须唯一且不包含空格。构建系统在生成最终的so库文件时，会参考该模块名生成最终的so库，如：libhello-jni.so；
`LOCAL_SRC_FILES`： 编译该模块时所需的C/C++源文件，如果有多个文件需要用空格分离，如果想换行则需要在每个源文件末尾加上反斜杠"\" ,类似于C语言中的多行宏定义；
`include ${BUILD_SHARED_LIBRARY}`： 确定要构建的内容及其操作方法。BUILD_SHARED_LIBRARY 表示要编译为动态库，构建系统会生成后缀名为.so的库文件；BUILD_STATIC_LIBRARY 表示要编译为静态库，构建系统会生成后缀名为.a 的为文件。

该文件中的一些扩展的变量：

```
# 指定头文件所在的目录
LOCAL_C_INCLUDES := $(LOCAL_PATH)

# 指定在构建动态库或可执行文件时需要链接的其它库列表，如Android系统提供的日志库，OpenGL ES，EGL等
# 每个链接的库文件都必须加上-l 前缀(注意是小写英文字母l,不是数字1)
# 多个链接的库文件使用空格分隔
LOCAL_LDLIBS +=  -llog -landroid -lc
```

3. 一个简单的Application.mk文件

```
APP_ABI = armeabi-v7a
APP_MODULES = hello-jni
```

APP_ABI：指定支持的ABI平台。上面所示为armeabi-v7a, 可选的值有all (代表全平台)、arm64-v8a、x86、x86_64，多个平台用空格隔开。
APP_MODULES：如果指定，那么NDK只会编译列出的模块列表，模块名用空格隔开，如果没有指定那么NDK会编译所有的Android.mk声明的所有的LOCAL_MODULE模块。

该文件的一些扩展的变量：

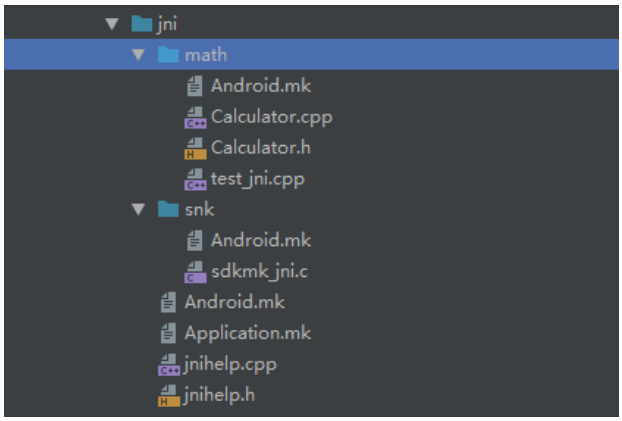
```
# 指定目标Android平台的名称
APP_PLATFORM = android-26

# 是否支持C++标准库
APP_STL := stlport_static

# 为项目中的所有C++编译传递的标记
APP_CPPFLAGS := -frtti -fexceptions -std=c++11
```

4. 多个编译模块的处理

有时候要编译的模块(UDP、FFmpeg、Openssl、libyuv)比较多，如果把所有的C/C++代码放到同一个目录下会显得比较乱，因此可以在每一个模块下都定义一个Android.mk文件，然后在jni 的根目录下放置一个Android.mk文件，内容如下：

```
include $(call all-subdir-makefiles)
```

比如：



5. 其它注意事项

1. 以LOCAL_、PRIVATE_、NDK_、APP_ 开头的名称是NDK编译系统的保留变量名称，在自定义变量时不要以以上变量名打头。如果你想在Android.mk文件中定义自己的变量，建议在名称前附加 MY_；

2. Makefile中的： = 、？ =、+=、= 的区别。= 是最基本的赋值，:= 是覆盖之前的值，?= 是如果没有被赋值过就赋予等号后面的值，+= 是添加等号后面的值。

参考链接：

1. Google Developers —— NDK Android.mk说明
2. Android.mk中以LOCAL_ 打头的变量说明
3. Android 中的android.mk 和 application.mk 文件编写(总结版)
4. Makefile 中:= ?= += =的区别
5. JNI学习——关于Android.mk / Application.mk