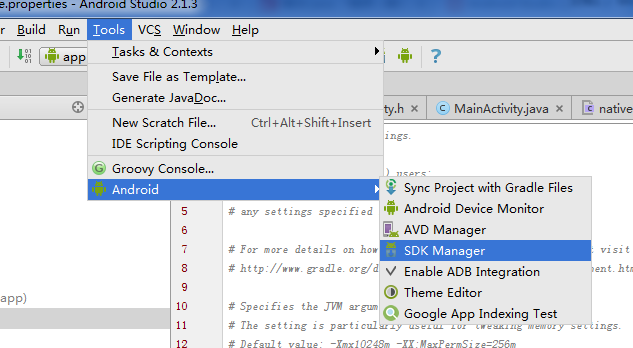
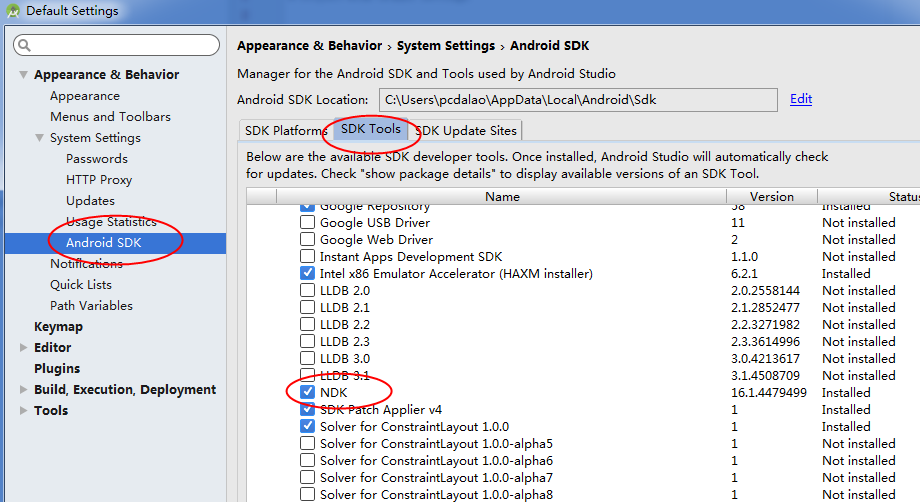
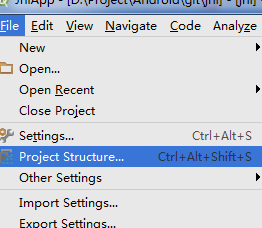
Android Studio 之 JNI 开发详解

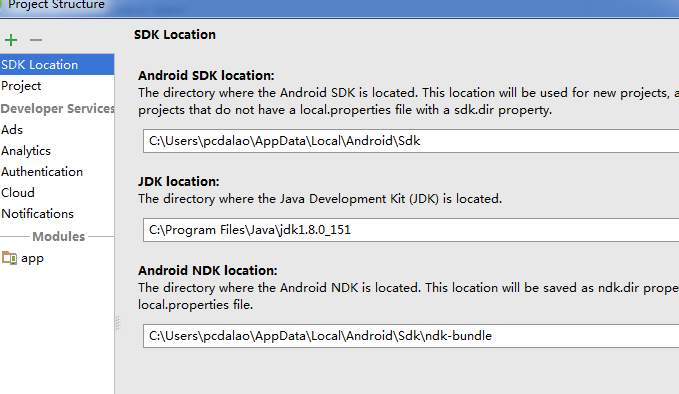
1. NDK全称是Native Development Kit, 自动将so和java应用一起打包成apk
2. JNI全称为：Java Native Interface, JNI 是本地编程接口，它使得在 Java 虚拟机内部运行的 Java 代码能够与用其它语言(如 C、C++)编写的代码进行交互
3. **本例开发环境如下：**  
   操作系统：Windows  
   开发环境：Android Studio 2.1.3 + NDK16.1.44 + Gradle 2.14.1
4. **安装NDK**



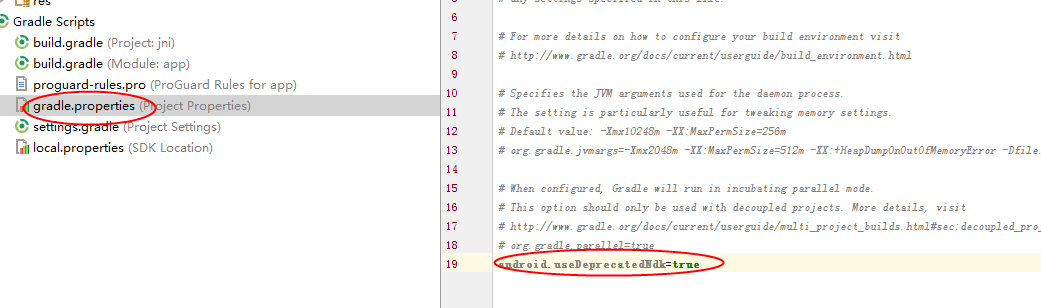


查看ndk路径：

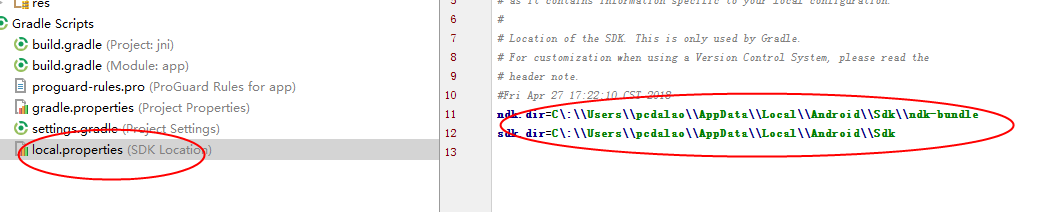




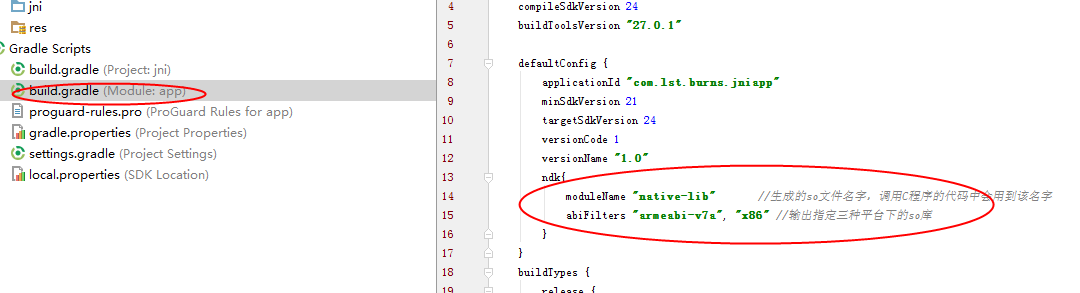
1. 新建一个项目，标准android app
2. 使用NDK进行开发



1. 在项目local.properties中加入ndk和sdk的路径 , 此处由studio自动添加，确认一下。

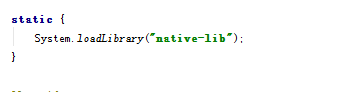


1. 在app文件夹下的build.gradle中的defaultConfig里加入如下代码



注意，如果增加 “armeabi”将导致编译错误

1. 打开MainActivity.java，添加如下代码



@Override protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

TextView textView = (TextView) findViewById(R.id.textView); textView.setText(getStringFromNative());

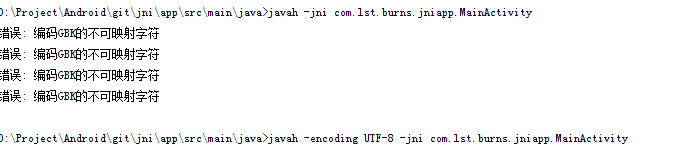
}

//声明一个本地方法，用native关键字修饰

public native String getStringFromNative();

1. 生成.h头文件  
   打开Android Studio底部的Terminal，默认命令行窗口路径已经在当前项目，输入以下命令：

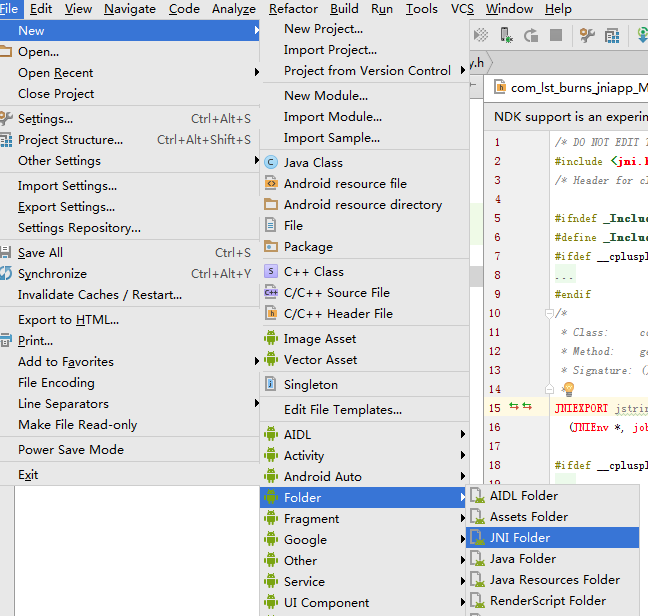


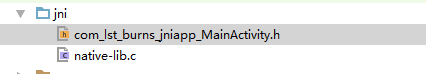


将自动生成头文件：com\_lst\_burns\_jniapp\_MainActivity.h

可以看出函数名定义规则，以后是否可以不用定义头文件？

1. 新建jni 文件夹，并将上面的头文件放到此文件夹下：





1. 在jni文件夹下新建.c 文件，如上图所示。

#include **"com\_lst\_burns\_jniapp\_MainActivity.h"**JNIEXPORT jstring JNICALL Java\_com\_lst\_burns\_jniapp\_MainActivity\_getStringFromNative  
 (JNIEnv \*env, jobject obj){  
 **char** \*str=**"String from native C"**;  
 **return** (\*env)->NewStringUTF(env, str);

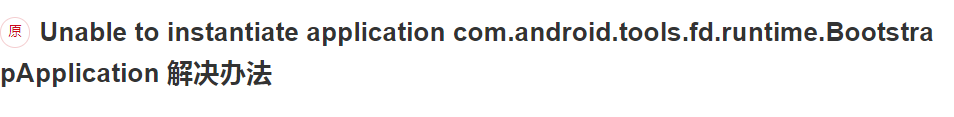
1. Build

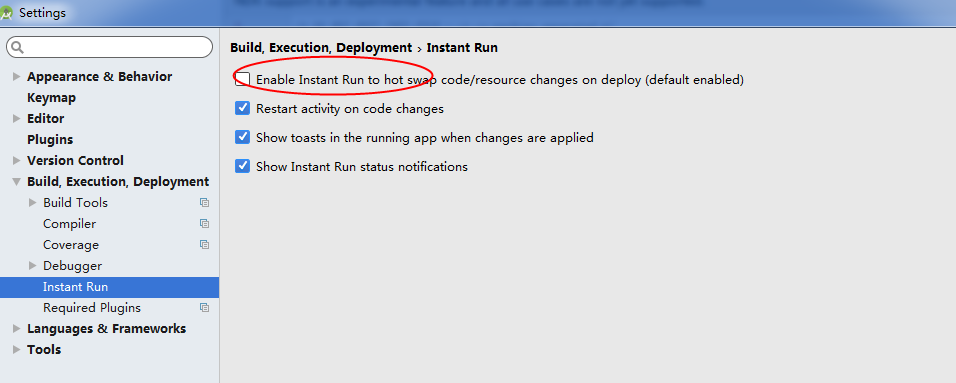
选择 Build->Make Project，看app/build/intermediates/ndk/debug/lib目录下是否生成.so文件，如果没有生成，选择 Build->Clean Project，等clean完成后，再Build->Rebuild Project，一般经过上面两步以后都能够解决问

1. 运行

模拟器上运行成功，

真机上运行不成功，显示：



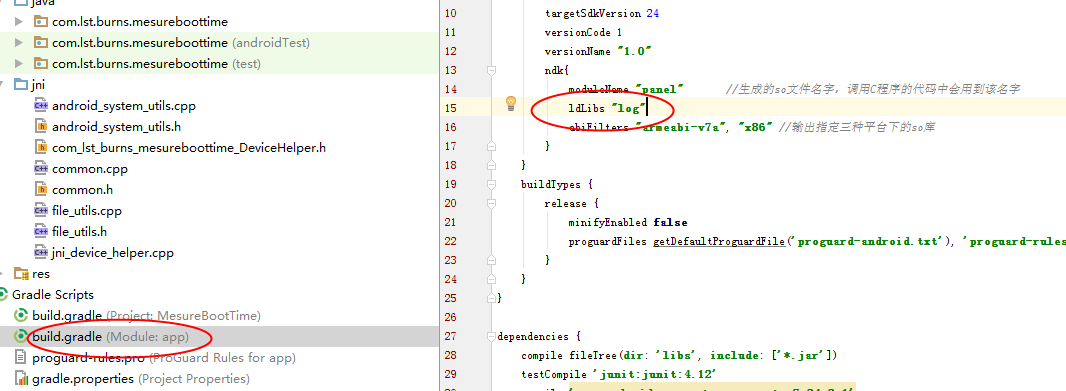


重新编译

真机上运行成功。

1. 如果jni引用了别的库，比如log

#include**<android/log.h>**



11)见例子MesureBootTime