Android 应用层到底层代码编写

作者:程姚根　　联系方式:chengyaogen123@163.com

一、在硬件抽象层添加模块

1. 添加模块相关的头文件

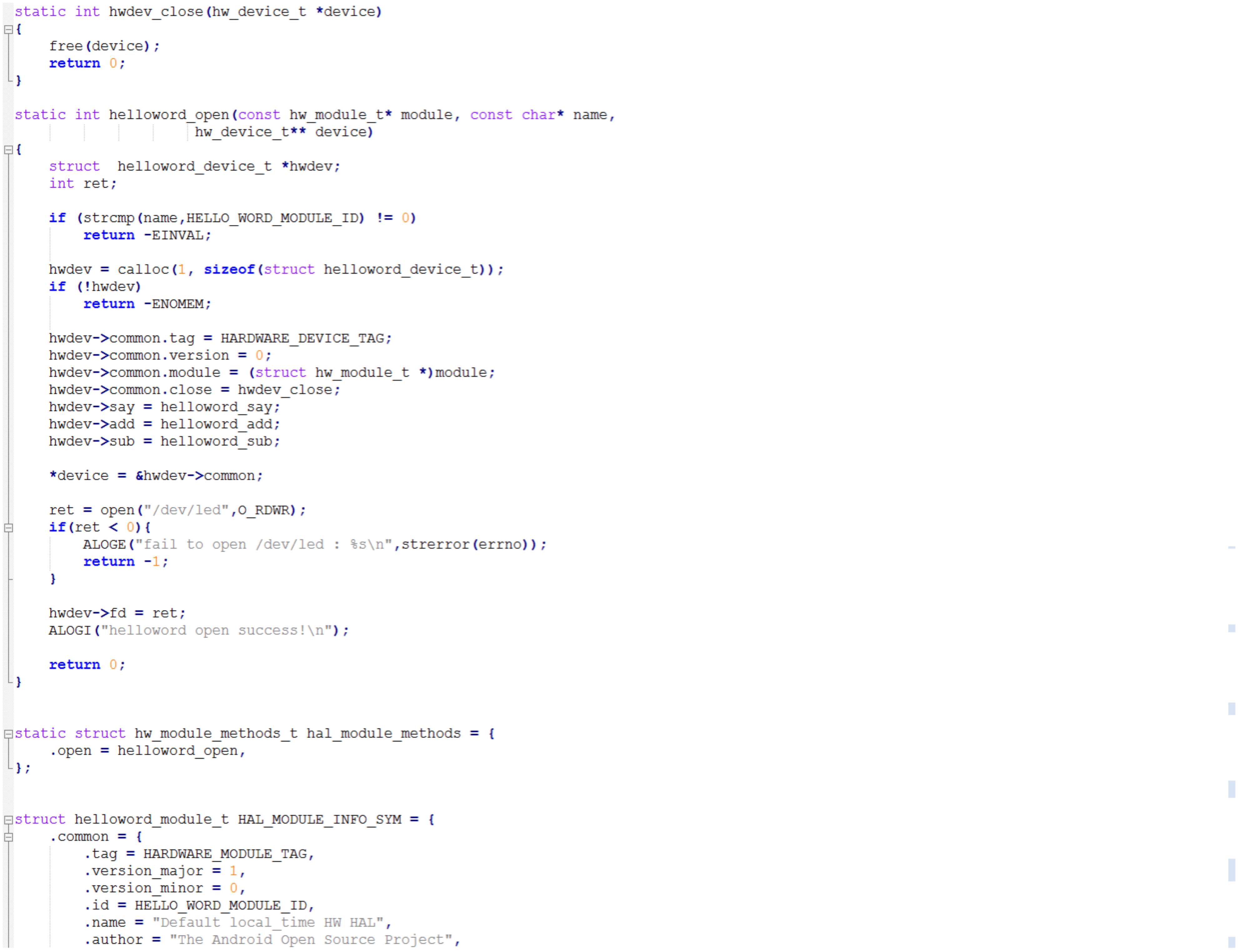
hardware/libhardware/include/hardware/helloword.h

```
#ifndef _HEAD_WORD_H
#define _HEAD_WORD_H
#include <hardware/hardware.h>

__BEGIN_DECLS


//定义模块ID
#define HELLO_WORD_MODULE_ID  "helloword"

//定义模块
struct helloword_module_t{
    struct hw_module_t common;
    char *des;
};

//定义设备描述结构体
struct helloword_device_t{
    struct hw_device_t common;
    void (*say)(struct helloword_device_t *hdev);
    int  (*add)(struct helloword_device_t *hdev,int a,int b);
    int  (*sub)(struct helloword_device_t *hdev,int a,int b);
    int fd;
};

__END_DECLS

#endif
```

2.添加自己的模块代码

在hardware/libhardware/modules目录下新建子目录helloword
(1)helloword.c

```c
#include <hardware/vibrator.h>
#include <hardware/hardware.h>
#include <hardware/helloword.h>

#define LOG_TAG "HelloWord"

#include <cutils/log.h>
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <math.h>
#include <sys/ioctl.h>
#include "fspad723_led.h"

static void helloword_say(struct helloword_device_t *hdev)
{
    ALOGI("helloword day!\n");
    return;
}

static int  helloword_add(struct helloword_device_t *hdev,int a,int b)
{
    int ret;
    int fd = hdev->fd;

    ALOGI("helloword add!\n");
    ret = ioctl(fd,LED_ON,0);
    if(ret < 0){
        ALOGE("fail to ioctl LED_ON : %s\n",strerror(errno));
        return -1;
    }

    return (a + b);
}

static int  helloword_sub(struct helloword_device_t *hdev,int a,int b)
{
    int ret;
    int fd = hdev->fd;

    ALOGI("helloword sub!\n");
    ret = ioctl(fd,LED_OFF,1);
    if(ret < 0){
        ALOGE("fail to ioctl LED_OF : %s\n",strerror(errno));
        return -1;
    }

    return (a - b);
}
```

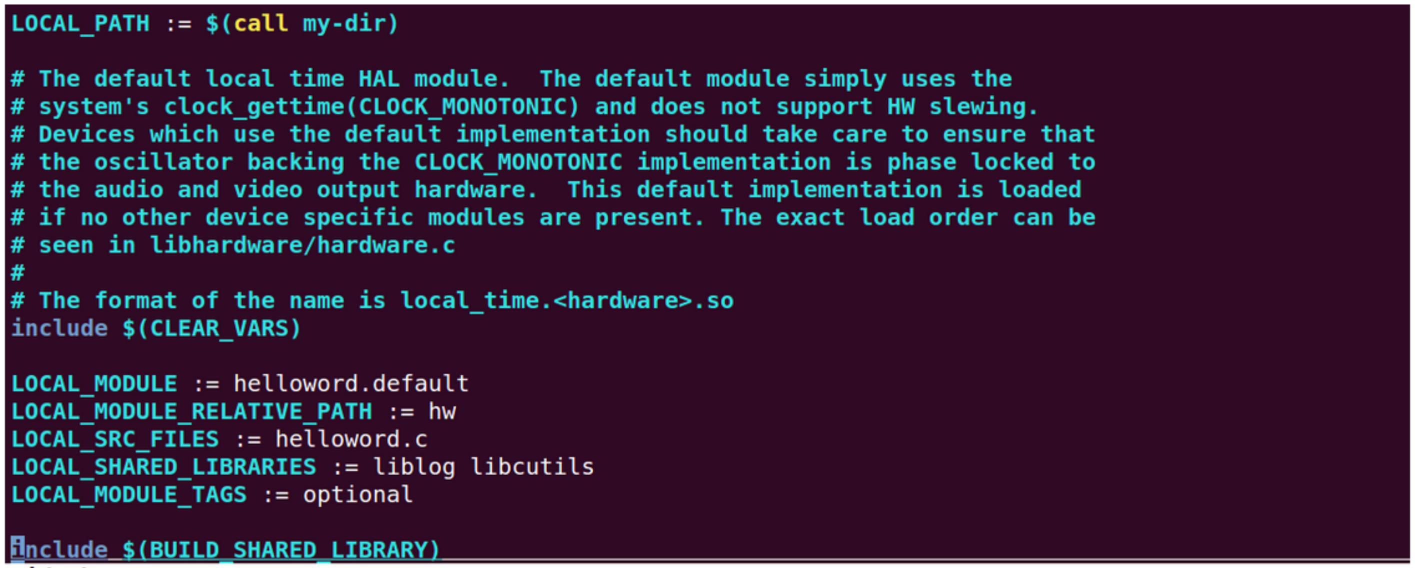


```c
static int hwdev_close(hw_device_t *device)
{
    free(device);
    return 0;
}

static int helloword_open(const hw_module_t* module, const char* name,
                          hw_device_t** device)
{
    struct  helloword_device_t *hwdev;
    int ret;

    if (strcmp(name,HELLO_WORD_MODULE_ID) != 0)
        return -EINVAL;

    hwdev = calloc(1, sizeof(struct helloword_device_t));
    if (!hwdev)
        return -ENOMEM;

    hwdev->common.tag = HARDWARE_DEVICE_TAG;
    hwdev->common.version = 0;
    hwdev->common.module = (struct hw_module_t *)module;
    hwdev->common.close = hwdev_close;
    hwdev->say = helloword_say;
    hwdev->add = helloword_add;
    hwdev->sub = helloword_sub;

    *device = &hwdev->common;

    ret = open("/dev/led",O_RDWR);
    if(ret < 0){
        ALOGE("fail to open /dev/led : %s\n",strerror(errno));
        return -1;
    }

    hwdev->fd = ret;
    ALOGI("helloword open success!\n");

    return 0;
}


static struct hw_module_methods_t hal_module_methods = {
    .open = helloword_open,
};


struct helloword_module_t HAL_MODULE_INFO_SYM = {
    .common = {
        .tag = HARDWARE_MODULE_TAG,
        .version_major = 1,
        .version_minor = 0,
        .id = HELLO_WORD_MODULE_ID,
        .name = "Default local_time HW HAL",
        .author = "The Android Open Source Project",
        .methods = &hal_module_methods,
    },

    .des = "helloword hal test!!!!",
};
```

(2)fspad723_led.h

```
#ifndef __S5PC100_LED_H
#define __S5PC100_LED_H

#define LED_ON  _IO('k', 0)
#define LED_OFF _IO('k', 1)

#endif
```

(3)Android.mk



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

# The default local time HAL module.  The default module simply uses the
# system's clock_gettime(CLOCK_MONOTONIC) and does not support HW slewing.
# Devices which use the default implementation should take care to ensure that
# the oscillator backing the CLOCK_MONOTONIC implementation is phase locked to
# the audio and video output hardware.  This default implementation is loaded
# if no other device specific modules are present. The exact load order can be
# seen in libhardware/hardware.c
#
# The format of the name is local_time.<hardware>.so
include $(CLEAR_VARS)

LOCAL_MODULE := helloword.default
LOCAL_MODULE_RELATIVE_PATH := hw
LOCAL_SRC_FILES := helloword.c
LOCAL_SHARED_LIBRARIES := liblog libcutils
LOCAL_MODULE_TAGS := optional

include $(BUILD_SHARED_LIBRARY)
```

3.编译模块

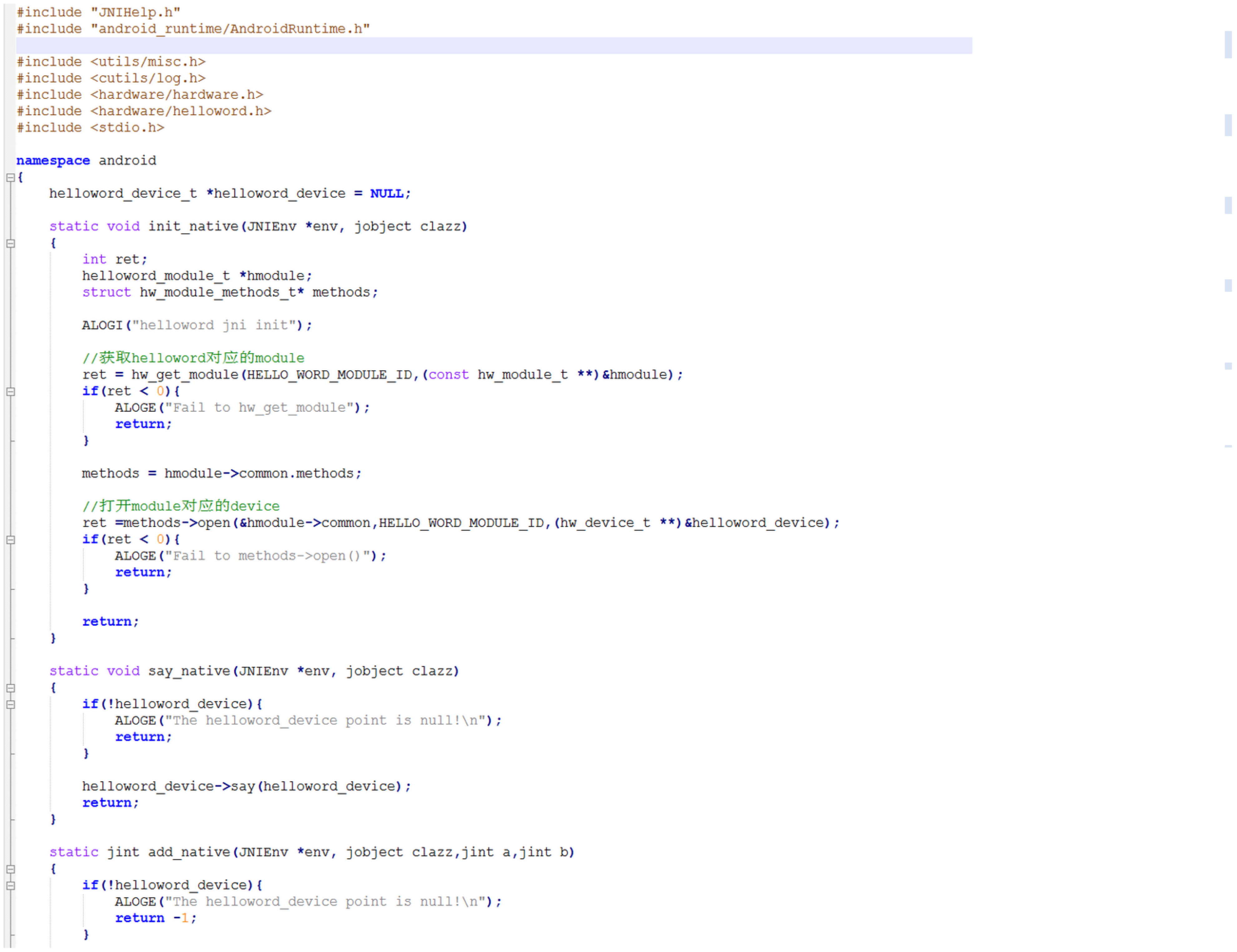
mmm hardware/libhardware/modules/helloword

二、Android runtime 中添加JNI C/C++代码

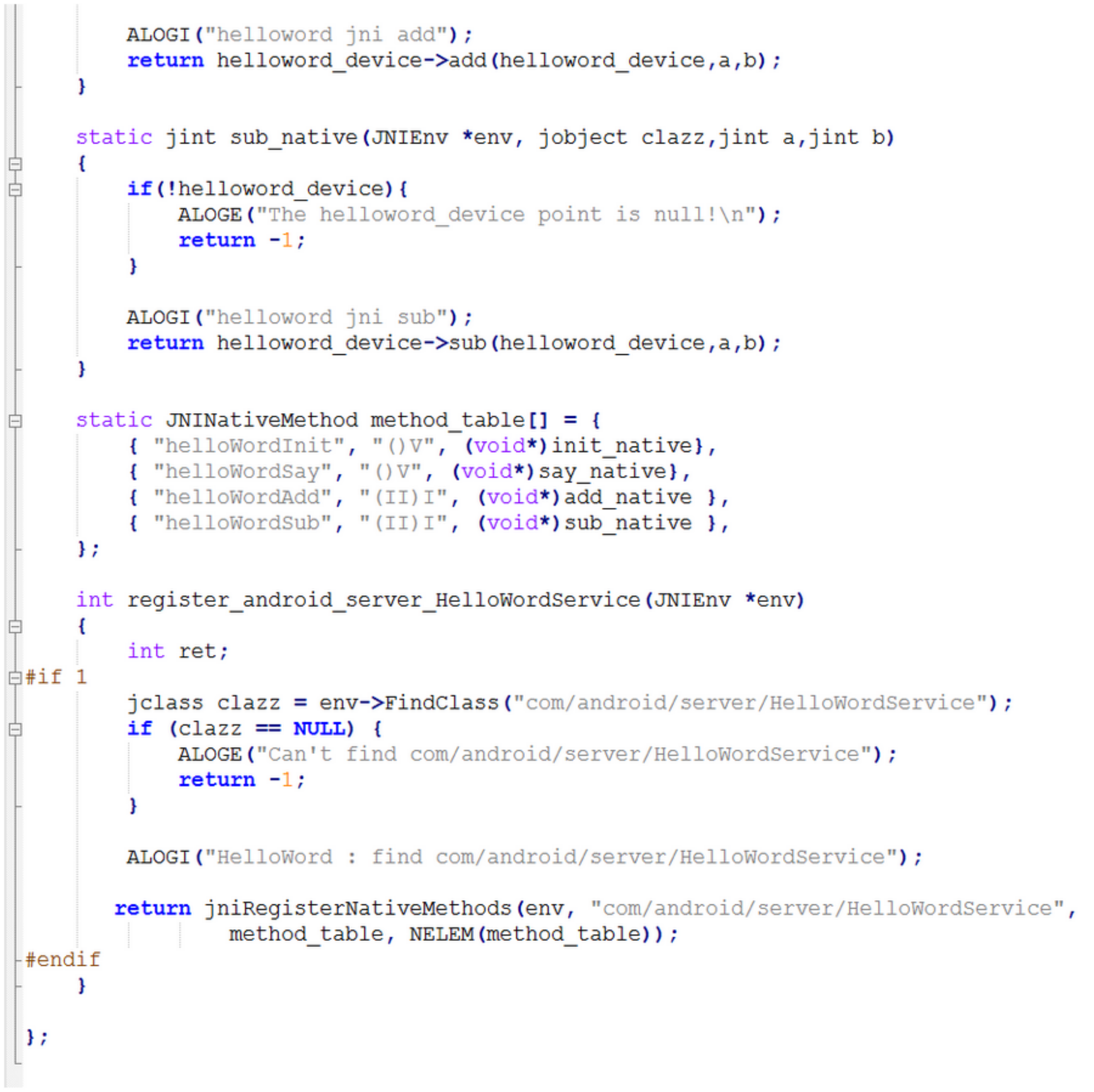
1.添加JNI C/C++代码

frameworks/base/services/core/jni/com_android_server_HelloWordService.cpp

```
#define LOG_TAG "HelloWordService"
#include "jni.h"
```

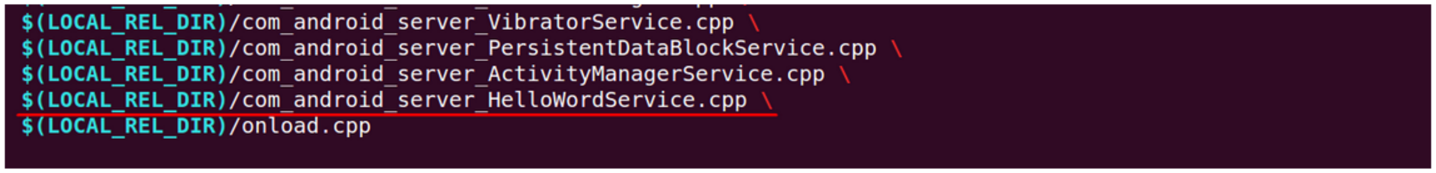


```
#include "JNIHelp.h"
#include "android_runtime/AndroidRuntime.h"

#include <utils/misc.h>
#include <cutils/log.h>
#include <hardware/hardware.h>
#include <hardware/helloword.h>
#include <stdio.h>

namespace android
{
    helloword_device_t *helloword_device = NULL;

    static void init_native(JNIEnv *env, jobject clazz)
    {
        int ret;
        helloword_module_t *hmodule;
        struct hw_module_methods_t* methods;

        ALOGI("helloword jni init");

        //获取helloword对应的module
        ret = hw_get_module(HELLO_WORD_MODULE_ID,(const hw_module_t **)&hmodule);
        if(ret < 0){
            ALOGE("Fail to hw_get_module");
            return;
        }

        methods = hmodule->common.methods;

        //打开module对应的device
        ret =methods->open(&hmodule->common,HELLO_WORD_MODULE_ID,(hw_device_t **)&helloword_device);
        if(ret < 0){
            ALOGE("Fail to methods->open()");
            return;
        }

        return;
    }

    static void say_native(JNIEnv *env, jobject clazz)
    {
        if(!helloword_device){
            ALOGE("The helloword_device point is null!\n");
            return;
        }

        helloword_device->say(helloword_device);
        return;
    }

    static jint add_native(JNIEnv *env, jobject clazz,jint a,jint b)
    {
        if(!helloword_device){
            ALOGE("The helloword_device point is null!\n");
            return -1;
        }
```



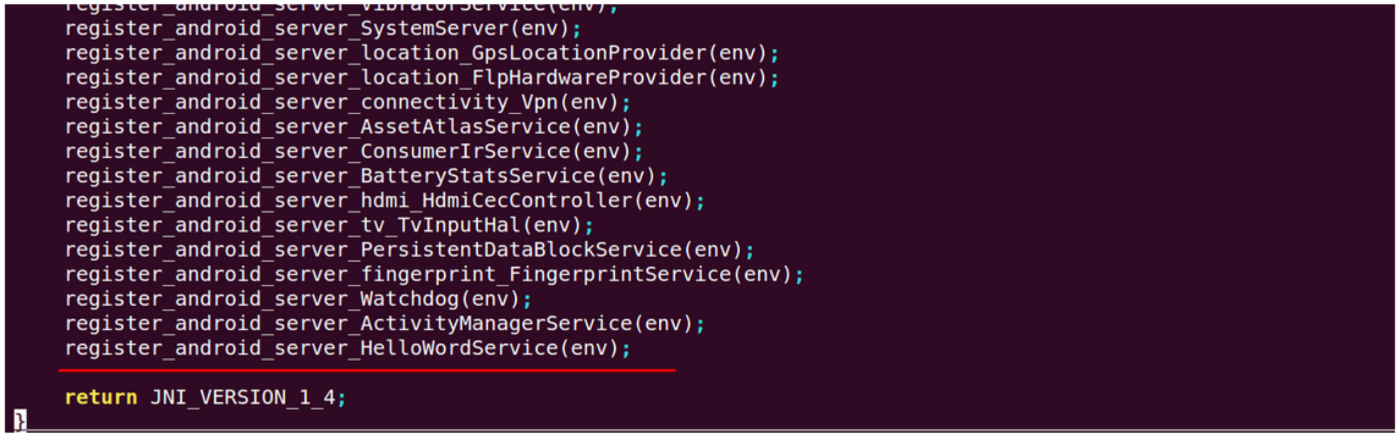
```
        ALOGI("helloword jni add");
        return helloword_device->add(helloword_device,a,b);
    }

    static jint sub_native(JNIEnv *env, jobject clazz,jint a,jint b)
    {
        if(!helloword_device){
            ALOGE("The helloword_device point is null!\n");
            return -1;
        }

        ALOGI("helloword jni sub");
        return helloword_device->sub(helloword_device,a,b);
    }

    static JNINativeMethod method_table[] = {
        { "helloWordInit", "()V", (void*)init_native},
        { "helloWordSay", "()V", (void*)say_native},
        { "helloWordAdd", "(II)I", (void*)add_native },
        { "helloWordSub", "(II)I", (void*)sub_native },
    };

    int register_android_server_HelloWordService(JNIEnv *env)
    {
        int ret;
#if 1
        jclass clazz = env->FindClass("com/android/server/HelloWordService");
        if (clazz == NULL) {
            ALOGE("Can't find com/android/server/HelloWordService");
            return -1;
        }

        ALOGI("HelloWord : find com/android/server/HelloWordService");

       return jniRegisterNativeMethods(env, "com/android/server/HelloWordService",
                method_table, NELEM(method_table));
#endif
    }

};
```

2.添加自己的jni文件到frameworks/base/services/core/jni/下的Android.mk文件中



```
 $(LOCAL_REL_DIR)/com_android_server_VibratorService.cpp \
 $(LOCAL_REL_DIR)/com_android_server_PersistentDataBlockService.cpp \
 $(LOCAL_REL_DIR)/com_android_server_ActivityManagerService.cpp \
 $(LOCAL_REL_DIR)/com_android_server_HelloWordService.cpp \
 $(LOCAL_REL_DIR)/onload.cpp
```

3.在frameworks/base/services/core/jni/onload.cpp中注册自己的JNI函数



```
    register_android_server_VibratorService(env);
    register_android_server_SystemServer(env);
    register_android_server_location_GpsLocationProvider(env);
    register_android_server_location_FlpHardwareProvider(env);
    register_android_server_connectivity_Vpn(env);
    register_android_server_AssetAtlasService(env);
    register_android_server_ConsumerIrService(env);
    register_android_server_BatteryStatsService(env);
    register_android_server_hdmi_HdmiCecController(env);
    register_android_server_tv_TvInputHal(env);
    register_android_server_PersistentDataBlockService(env);
    register_android_server_fingerprint_FingerprintService(env);
    register_android_server_Watchdog(env);
    register_android_server_ActivityManagerService(env);
    register_android_server_HelloWordService(env);

    return JNI_VERSION_1_4;
}
```

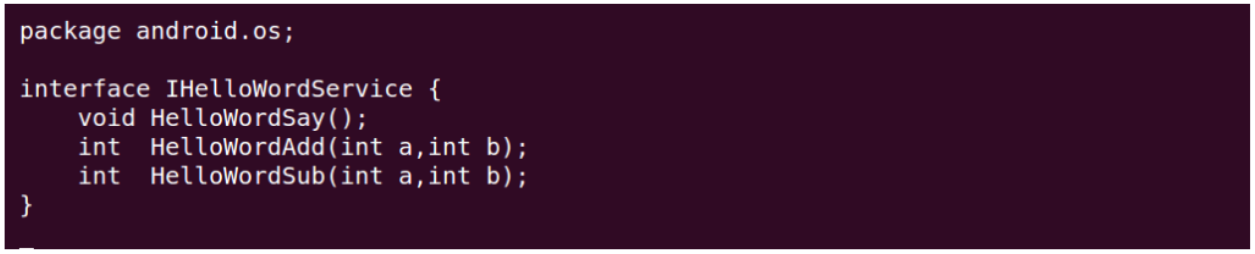
4.编译JNI代码

mmm frameworks/base/services

三、AIDL

1.添加APP与service之间的通信接口

在Android系统中，硬件服务一般是运行在一个独立的进程中为各种应用程序提供服务。因此，调用这些硬件服务的应用程序与这些硬件服务之间的通信需要通过代理来进行。为此，我们要先定义好通信接口。进入到**frameworks/base/core/java/android/os**目录，新增IHelloWordService.aidl接口定义文件：



```
package android.os;

interface IHelloWordService {
    void HelloWordSay();
    int  HelloWordAdd(int a,int b);
    int  HelloWordSub(int a,int b);
}
```

2.将自己的AIDL文件添加到frameworks/base/android.mk文件中

```
 LOCAL_SRC_FILES += /
  ..................................................................
  core/java/android/os/IVibratorService.aidl /
  core/java/android/os/IHelloWordService.aidl /
  core/java/android/service/urlrenderer/IUrlRendererService.aidl /
  ..................................................................
```

3.编译IHelloWordService.aidl文件

mmm frameworks/base/

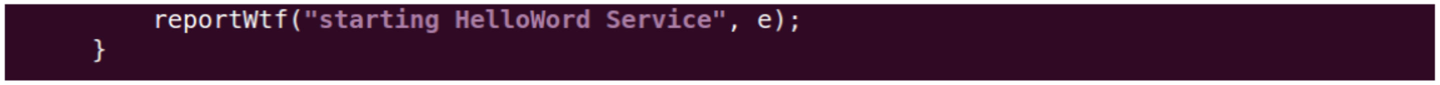
四、添加自己的服务

1.进入 frameworks/base/services/core/java/com/android/server目录下，添加HelloWordService.java代码

```
package com.android.server;
import  android.os.IHelloWordService;
import  android.util.Slog;

public class HelloWordService extends IHelloWordService.Stub{
    HelloWordService(){
        Slog.i("HelloWordService","HelloWordService init ...");
        helloWordInit();
    }


    public void HelloWordSay(){
        Slog.i("HelloWordService","HelloWordService say ...");
        helloWordSay();
        return;
    }

    public int HelloWordAdd(int a,int b){
        Slog.i("HelloWordService","HelloWordService add");
        return helloWordAdd(a,b);
    }

    public int HelloWordSub(int a,int b){
        Slog.i("HelloWordService","HelloWordService sub");
        return helloWordSub(a,b);
    }

    public native  void  helloWordInit();
    public native  void  helloWordSay();
    public native  int   helloWordAdd(int a,int b);
    public native  int   helloWordSub(int a,int b);
}
```

2.修改framework/base/services/java/com/android/server目录下的SystemServer.java文件
在startOtherServices()中添加自己的service

```
      try {
          Slog.i(TAG, "HelloWord Service");
          ServiceManager.addService("helloword", new HelloWordService());
      } catch (Throwable e){
```

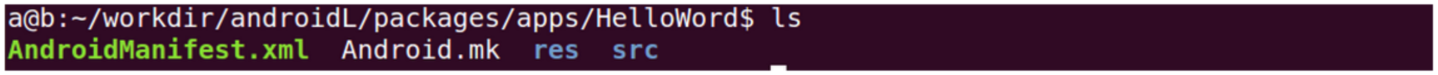


```
        reportWtf("starting HelloWord Service", e);
    }
```

3.编译相关代码

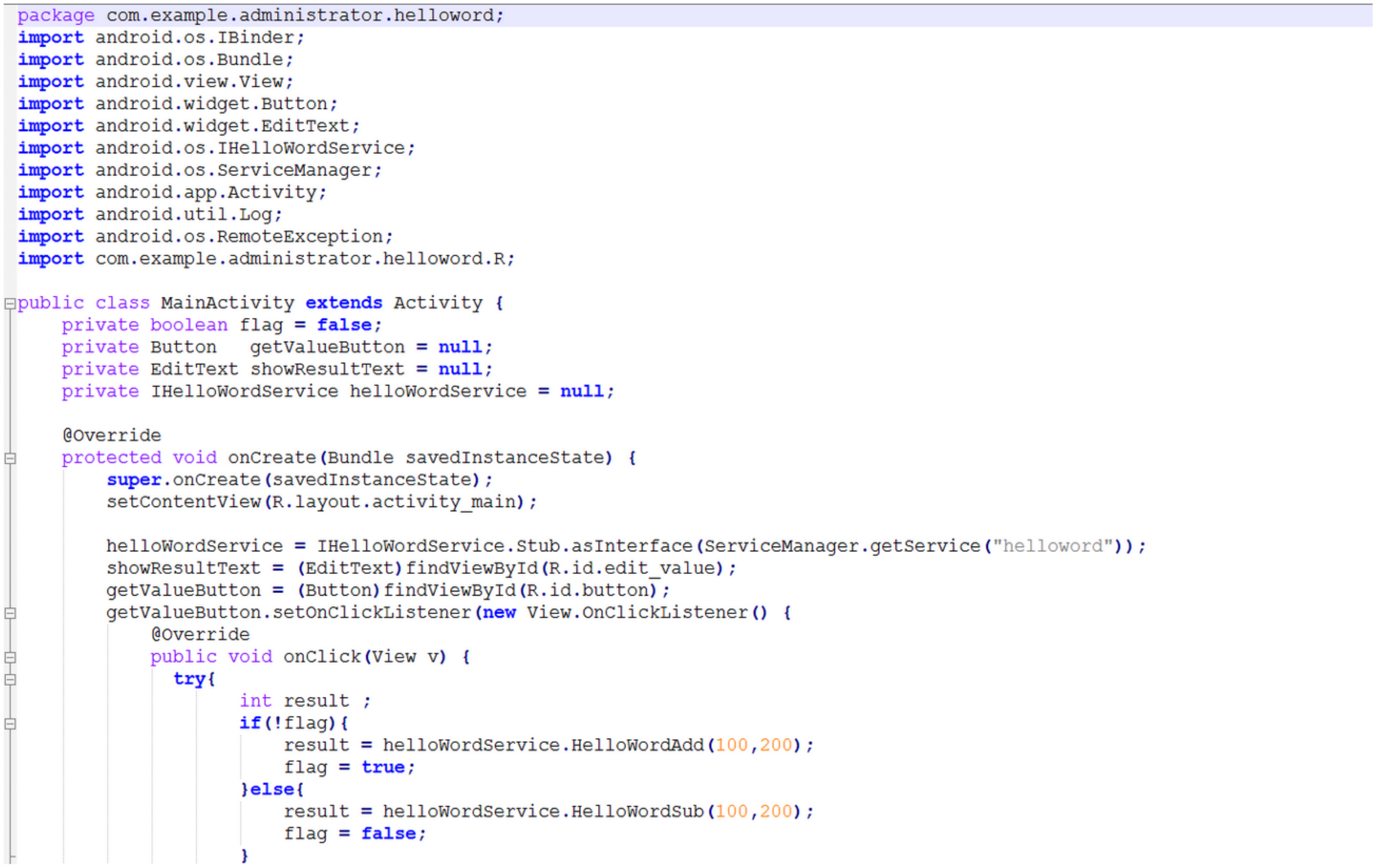
mmm framework/base/services

五、编写Android App程序

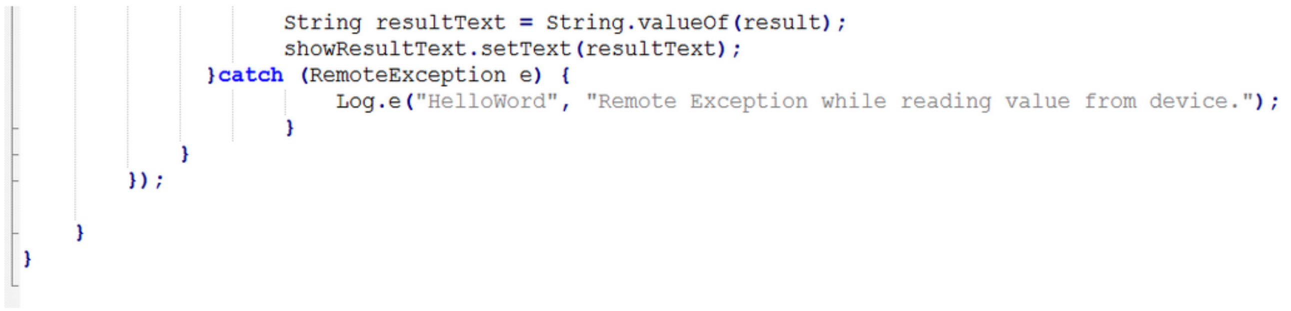


```
a@b:~/workdir/androidL/packages/apps/HelloWord$ ls
AndroidManifest.xml  Android.mk  res  src
```

1.src\com\example\administrator\helloword\MainActivity.java

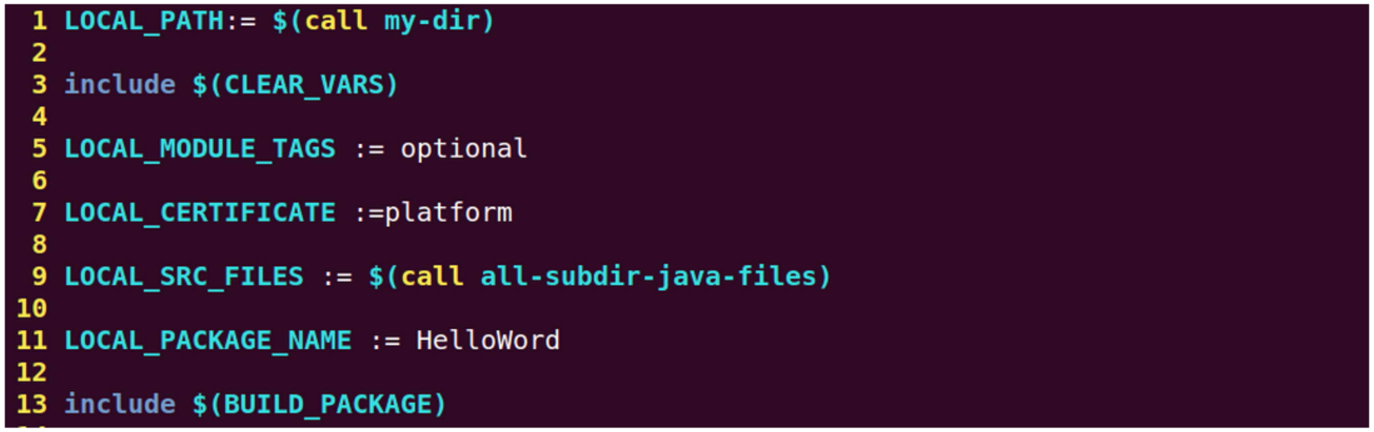


```java
package com.example.administrator.helloword;
import android.os.IBinder;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.os.IHelloWordService;
import android.os.ServiceManager;
import android.app.Activity;
import android.util.Log;
import android.os.RemoteException;
import com.example.administrator.helloword.R;

public class MainActivity extends Activity {
    private boolean flag = false;
    private Button   getValueButton = null;
    private EditText showResultText = null;
    private IHelloWordService helloWordService = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        helloWordService = IHelloWordService.Stub.asInterface(ServiceManager.getService("helloword"));
        showResultText = (EditText)findViewById(R.id.edit_value);
        getValueButton = (Button)findViewById(R.id.button);
        getValueButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
              try{
                    int result ;
                    if(!flag){
                        result = helloWordService.HelloWordAdd(100,200);
                        flag = true;
                    }else{
                        result = helloWordService.HelloWordSub(100,200);
                        flag = false;
                    }
```



```java
                        String resultText = String.valueOf(result);
                        showResultText.setText(resultText);
                    }catch (RemoteException e) {
                        Log.e("HelloWord", "Remote Exception while reading value from device.");
                    }
                }
            });

        }
}
```

2.Android.mk



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

include $(CLEAR_VARS)

LOCAL_MODULE_TAGS := optional

LOCAL_CERTIFICATE :=platform

LOCAL_SRC_FILES := $(call all-subdir-java-files)

LOCAL_PACKAGE_NAME := HelloWord

include $(BUILD_PACKAGE)
```