

原创

HIDL最全编译流程

置顶

2018-11-27 10:49:07

Gunder

阅读数 3899

更多

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：<https://blog.csdn.net/u013357557/article/details/84561652>

想了解HIDL介绍的可以参考《[HIDL概述](#)》，本篇文章主要介绍HIDL的详细编译流程及简单的客户端应用（C++Android客户端的应用）。

一、准备工作

1. 整套源码，Android O或者Android P的
2. 保证全套代码已经全编译，原生代码全编译命令

- source build/envset.sh
- lunch
- make

3、hidl-gen工具已经安装，安装命令

- make hidl-gen

二、hidl-gen工具介绍

系统定义的所有的.hal接口，都是通过hidl-gen工具转换成对应的代码。hidl-gen源码路径：system/tools/hidl，是在ubuntu上可执行的二进制文件。

使用方法：hidl-gen -o output-path -L language (-r interface-root) fqname

例子：

```
hidl-gen -o hardware/interfaces/gunder/1.0/default/ -Lc++-impl -randroid.hardware:hardware/interfaces -randroid.hidl:system/libhid
android.hardware.gunder@1.0
```

参数说明：

- -L：语言类型，包括c++，c++-headers，c++-sources，export-header，c++-impl，java，java-constants，vts，makefile，androidbp，androidb等。hidl-gen可根据传入的语言类型产生不同的文件。
- fqname：完全限定名称的输入文件。比如本例中android.hardware.gunder@1.0，要求在源码目录下必须有hardware/interfaces/ gunder /1.单个文件来说，格式如下：package@version::fileName，比如android.hardware. gunder @1.0::types.Feature。对于目录来说，格式如下package@version，比如android.hardware. gunder @1.0。
- -r：格式package:path，可选，对fqname对应的文件来说，用来指定包名和文件所在的目录到Android系统源码根目录的路径。如果没有制定，是：android.hardware，目录是Android源码的根目录。
- -o：存放hidl-gen产生的中间文件的路径。

可以使用hidl-gen 查看帮助，如图1：

```

gunder@ubuntu:~/IMX8_b0beta2$ hidl-gen
usage: hidl-gen [-p <root path>] -o <output path> -L <language> (-r <interface root>)+ [-t] fqname+
-h: Prints this menu.
-L <language>: The following options are available:
check          : Parses the interface to see if valid but doesn't write files.
c++            : (internal) (deprecated) Generates C++ interface files for talking to HIDL interfaces.
c++-headers    : (internal) Generates C++ headers for interface files for talking to HIDL interfaces.
c++-sources    : (internal) Generates C++ sources for interface files for talking to HIDL interfaces.
export-header   : Generates a header file from @export enumerations to maintain legacy code.
c++-impl       : Generates boilerplate implementation of a hidl interface in C++ (for convenience).
c++-impl-headers: c++-impl but headers only
c++-impl-sources: c++-impl but sources only
java           : (internal) Generates Java library for talking to HIDL interfaces in Java.
java-constants : (internal) Like export-header but for Java (always created by -Lmakefile if @export ex
vts            : (internal) Generates vts proto files for use in vtsd.
makefile       : (internal) Generates makefiles for -Ljava and -Ljava-constants.
androidbp      : (internal) Generates Soong bp files for -Lc++-headers and -Lc++-sources.
androidbp-impl : Generates boilerplate bp files for implementation created with -Lc++-impl.
hash           : Prints hashes of interface in `current.txt` format to standard out.
-o <output path>: Location to output files.
-p <root path>: Android build root, defaults to $ANDROID_BUILD_TOP or pwd.
-r <package:path root>: E.g., android.hardware:hardware/interfaces.
-t: generate build scripts (Android.bp) for tests.
gunder@ubuntu:~/IMX8_b0beta2$

```

图1 hidl-gen的帮助信息

三、项目实例

1、在hardware/interfaces/目录下新建gunder/1.0目录，并在1.0目录中创建接口IGunder.hal。目录结构如下：

```
gunder@ubuntu-MX8:~ /IMX8_b0beta2/hardware/interfaces/gunder$ tree
```

```

.
├── 1.0
│   └── IGunder.hal

```

IGunder.hal文件里面只有一个接口IGunder和一个方法helloWorld(string name)，具体实现如下：

```

1 package android.hardware.gunder@1.0;
2
3 interface IGunder{
4     helloWorld(string name) generates (string result);
5 };
6

```

2、执行下面三条命令会自动生成对应的c++文件；

- PACKAGE=android.hardware.gunder@1.0
- LOC=hardware/interfaces/gunder/1.0/default/
- hidl-gen -o \$LOC -Lc++-impl -randroid.hardware:hardware/interfaces -randroid.hidl:system/libhidl/transport \$PACKAGE

执行命令后的目录结构如下：

```
gunder@ubuntu-MX8:~ /IMX8_b0beta2$ PACKAGE=android.hardware.gunder@1.0
```

```
gunder@ubuntu-MX8:~ /IMX8_b0beta2$ LOC=hardware/interfaces/gunder/1.0/default/
```

```
gunder@ubuntu-MX8:~ /IMX8_b0beta2$ hidl-gen -o $LOC -Lc++-impl -randroid.hardware:hardware/interfaces -randroid.hidl:system/libhidl/transport $PACKAGE
```

gunder@ubuntu-MX8:~ /IMX8_b0beta2/hardware/interfaces/gunder\$ tree

```
.
├── 1.0
│   ├── default
│   │   ├── Gunder.cpp
│   │   └── Gunder.h
│   └── IGunder.hal
```

default 是新生成的目录，Gunder.cpp和Gunder.h是新生成的两个文件，打开Gunder.h文件，去掉// extern IGunder* HIDL_FETCH_IGunder(name);前面的注释，使用直通式HAL (Passthrough 模式) 来通信。Gunder.h文件修改后如图2：

```
#ifndef ANDROID_HARDWARE_GUNDER_V1_0_GUNDER_H
#define ANDROID_HARDWARE_GUNDER_V1_0_GUNDER_H

#include <android/hardware/gunder/1.0/IGunder.h>
#include <hidl/MQDescriptor.h>
#include <hidl/Status.h>

namespace android {
namespace hardware {
namespace gunder {
namespace V1_0 {
namespace implementation {

using ::android::hardware::hidl_array;
using ::android::hardware::hidl_memory;
using ::android::hardware::hidl_string;
using ::android::hardware::hidl_vec;
using ::android::hardware::Return;
using ::android::hardware::Void;
using ::android::sp;

struct Gunder : public IGunder {
    // Methods from IGunder follow.
    Return<void> helloWorld(const hidl_string& name, helloWorld_cb _hidl_cb) override;

    // Methods from ::android::hidl::base::V1_0::IBase follow.
};

// FIXME: most likely delete, this is only for passthrough implementations
extern "C" IGunder* HIDL_FETCH_IGunder(const char* name);

} // namespace implementation
} // namespace V1_0
} // namespace gunder
} // namespace hardware
} // namespace android

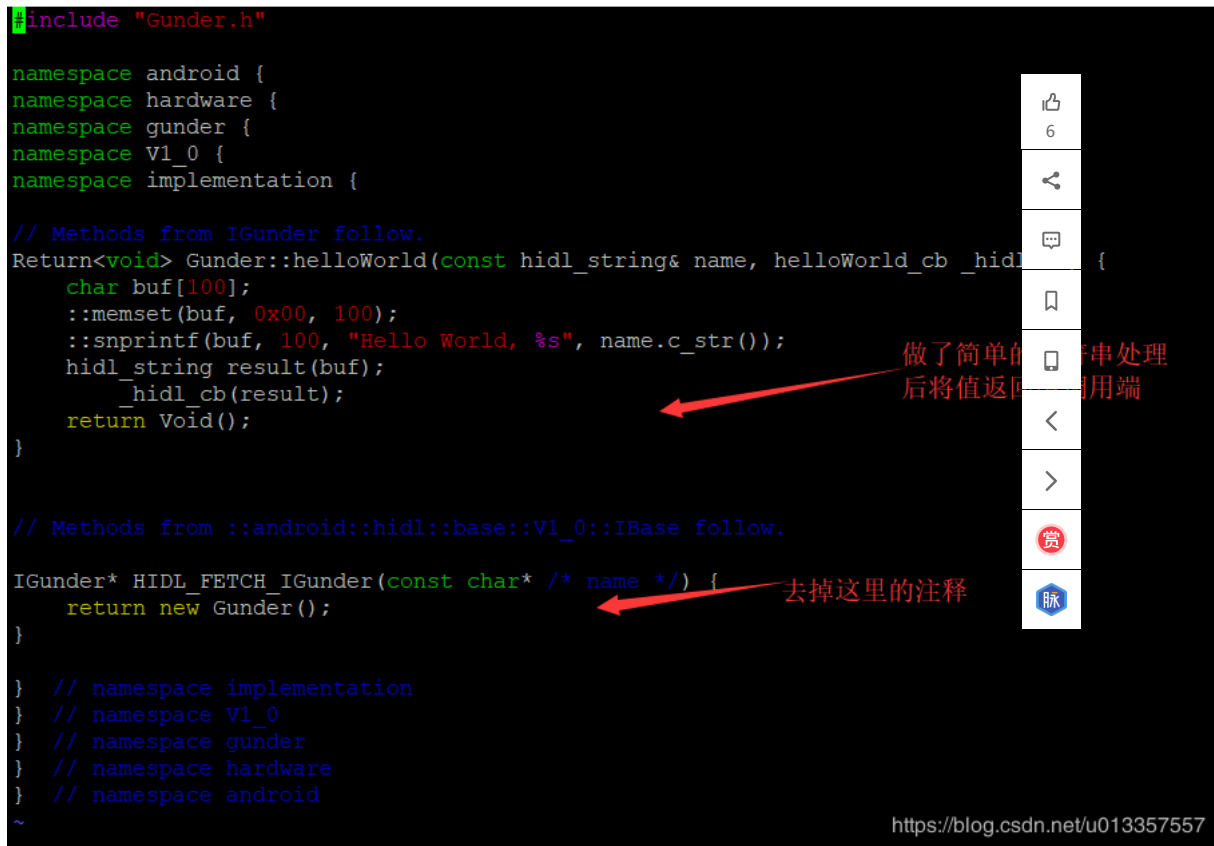
#endif // ANDROID_HARDWARE_GUNDER_V1_0_GUNDER_H
~
```

去掉了注释

<https://blog.csdn.net/u013357557>

图2 Gunder.h修改后的内容

Gunder.cpp文件也要进行对应的修改，修改后如如图3：



```

#include "Gunder.h"

namespace android {
namespace hardware {
namespace gunder {
namespace V1_0 {
namespace implementation {

// Methods from IGunder follow.
Return<void> Gunder::helloWorld(const hidl_string& name, helloWorld_cb _hidl_cb) {
    char buf[100];
    ::memset(buf, 0x00, 100);
    ::snprintf(buf, 100, "Hello World, %s", name.c_str());
    hidl_string result(buf);
    _hidl_cb(result);
    return Void();
}

// Methods from ::android::hidl::base::V1_0::IBase follow.
IGunder* HIDL_FETCH_IGunder(const char* /* name */) {
    return new Gunder();
}

} // namespace implementation
} // namespace V1_0
} // namespace gunder
} // namespace hardware
} // namespace android
}

```

做了简单的字符串处理 后将值返回给调用端

去掉这里的注释

<https://blog.csdn.net/u013357557>

图3 Gunder.cpp修改后的内容

3、执行下面命令

- `hidl-gen -o $LOC -Landroidbp-impl -randroid.hardware:hardware/interfaces -randroid.hidl:system/libhidl/transport $PACKAGE`

会在hardware/interfaces/gunder/1.0/default目录生成Android.bp文件。

gunder@ubuntu-MX8:~/IMX8_b0beta2/hardware/interfaces/gunder\$ tree

```

.
├── 1.0
│   ├── default
│   │   ├── Android.bp
│   │   ├── Gunder.cpp
│   │   └── Gunder.h
└── IGunder.hal

```

4、然后使用脚本update-makefiles.sh来更新Makefile，自动在hardware/interfaces/gunder/1.0目录生成Android.mk和 Android.bp，hardware/interfaces/gunder目录生成Android.bp。命令如下

- `./hardware/interfaces/update-makefiles.sh`

执行完命令后目录结构如下：

gunder@ubuntu-MX8:~/IMX8_b0beta2\$./hardware/interfaces/update-makefiles.sh

gunder@ubuntu-MX8:~/IMX8_b0beta2/hardware/interfaces/gunder\$ tree

```

.
├── 1.0
│   ├── Android.bp
│   └── Android.mk

```

```
| |— default
| | |— Android.bp
| | |— Gunder.cpp
| | |— Gunder.h
| |— IGunder.hal
|— Android.bp
```



5、在hardware/interfaces/gunder/1.0/default目录下新建service.cpp跟android.hardware.gunder@1.0-service.rc，其中android.hardware.gunder@1.0-service.rc是程序的入口函数。新的目录结构如下：

```
.
|— 1.0
| |— Android.bp
| |— Android.mk
| |— default
| | |— Android.bp
| | |— android.hardware.gunder@1.0-service.rc
| | |— Gunder.cpp
| | |— Gunder.h
| | |— service.cpp
| |— IGunder.hal
|— Android.bp
```

android.hardware.gunder@1.0-service.rc的实现如图：

```
service gunder_hal_service /vendor/bin/hw/android.hardware.gunder@1.0-service
class hal
user system
group system
```

service.cpp的实现如图：

```
#define LOG_TAG "android.hardware.gunder@1.0-service"
#include <android/hardware/gunder/1.0/IGunder.h>
#include <hidl/LegacySupport.h>
using android::hardware::gunder::V1_0::IGunder;
using android::hardware::defaultPassthroughServiceImplementation;

int main() {
    return defaultPassthroughServiceImplementation<IGunder>();
}
```

直通式HAL

<https://blog.csdn.net/u013357557>

打开hardware/interfaces/gunder/1.0/default目录下的Android.bp，添加编译service.cpp成为可执行文件的代码。具体添加内容如下：



```
cc_library_shared {
    name: "android.hardware.gunder@1.0-impl",
    relative_install_path: "hw",
    proprietary: true,
    srcs: [
        "Gunder.cpp",
    ],
    shared_libs: [
        "libhidlbase",
        "libhidltransport",
        "libutils",
        "android.hardware.gunder@1.0",
    ],
}

cc_binary {
    name: "android.hardware.gunder@1.0-service",
    relative_install_path: "hw",
    proprietary: true,
    init_rc: ["android.hardware.gunder@1.0-service.rc"],
    srcs: ["service.cpp"],
    shared_libs: [
        "liblog",
        "libcutils",
        "libdl",
        "libbase",
        "libutils",
        "libhardware",
        "libhidlbase",
        "libhidltransport",
        "android.hardware.gunder@1.0",
    ],
}
```

添加编辑条件来
生成android.hardware.gunder@1.0-service
可执行文件

<https://blog.csdn.net/u013357557>

到此跟HAL相关的代码就实现完了。

6、编译生成服务端跟客户端要用各种库文件。首先执行下面两条命令

- ./hardware/interfaces/update-makefiles.sh
- mmm hardware/interfaces/gunder/1.0/

执行后会生成下面的文件：

out/target/product/mek_8q/system/lib64/android.hardware.gunder@1.0.so

out/target/product/mek_8q/vendor/lib64/hw/android.hardware.gunder@1.0-impl.so

out/target/product/mek_8q/vendor/etc/init/android.hardware.gunder@1.0-service.rc

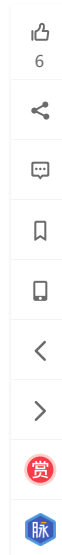
7、在manifest文件里添加vendor接口的定义编译device/fsl/mek_8q/manifest.xml文件（不同厂家路径可能不一样），添加以下内容，不然在client service的。如下：

```

<hal format="hidl">
  <name>android.hardware.boot</name>
  <transport>hwbinder</transport>
  <version>1.0</version>
  <interface>
    <name>IBootControl</name>
    <instance>default</instance>
  </interface>
</hal>
<hal format="hidl">
  <name>android.hardware.gunder</name>
  <transport>hwbinder</transport>
  <version>1.0</version>
  <interface>
    <name>IGunder</name>
    <instance>default</instance>
  </interface>
</hal>
</manifest>

```

<https://blog.csdn.net/u013357557>



8、使用C++实现客户端调用

在hardware/interfaces/gunder/1.0目录下新建test目录，并且在test目录下新建Android.bp跟GunderTest.cpp文件，这两个文件的内容如下：

GunderTest.cpp：

```

#include <android/hardware/gunder/1.0/IGunder.h>
#include <hidl/Status.h>
#include <hidl/LegacySupport.h>
#include <utils/misc.h>
#include <hidl/HidlSupport.h>
#include <stdio.h>

using ::android::hardware::hidl_string;
using ::android::sp;
using android::hardware::gunder::V1_0::IGunder;

int main() {
    android::sp<IGunder> service = IGunder::getService();
    if (service == nullptr) {
        printf("Failed to get service\n");
        return -1;
    }

    service->helloWorld("Gunder", [&](hidl_string result) {
        printf("%s\n", result.c_str());
    });
    return 0;
}

```

获取服务

调用HAL层定义的接口

<https://blog.csdn.net/u013357557>

Android.bp：

```

cc_binary {
    relative_install_path: "hw",
    defaults: ["hidl_defaults"],
    name: "gunder_client",
    proprietary: true,
    srcs: ["GunderTest.cpp"],

    shared_libs: [
        "liblog",
        "libhardware",
        "libhidlbase",
        "libhidltransport",
        "libutils",
        "android.hardware.gunder@1.0",
    ],
}

```

<https://blog.csdn.net/u013357557>



执行以下命令：

- ./hardware/interfaces/update-makefiles.sh
- mmm hardware/interfaces/gunder/1.0/

执行第一条命令是为了更新hardware/interfaces/gunder/目录下的Android.bp文件，如下：

```
// This is an autogenerated file, do not edit.
subdirs = [
    "1.0",
    "1.0/default",
    "1.0/test",
]
```

执行更新脚本后产生，这样就可以编译到test目录

执行第二条命令会生成可执行文件：

out/target/product/mek_8q/vendor/bin/hw/gunder_client

9、使用JAVA实现客户端调用

为了方便eclipse或者Android Studio调用接口函数，需要编译出classes.jar包。但是jack编译出来的文件是classes.jack。如图：

IMX8_b0beta2 > out > target > common > obj > JAVA_LIBRARIES > android.hardware.gunder-V1.0-java_intermediates

名称	修改日期	类型	大小
jack-rsc	2018/11/23 16:54	文件夹	
with-local	2018/11/23 16:54	文件夹	
classes.dex	2018/11/23 16:54	DEX 文件	9 KB
classes.jack	2018/11/23 16:54	JACK 文件	19 KB
jack_res_jar_flags	2018/11/23 16:54	文件	0 KB
jack-rsc.java-source-list	2018/11/23 16:54	JAVA-SOURCE-L...	1 KB
javali.jar	2018/11/23 16:54	JAR 文件	5 KB
link_type	2018/11/23 16:54	文件	1 KB

https://blog.csdn.net/u013357557

为了编译出classes.jar，需要打开hardware/interfaces/gunder/1.0目录的Android.mk，在include \$(CLEAR_VARS)下面添加LOCAL_JACK_ENABLE disabled。这样编译的时候就不走jack编译了。如下：


```

include $(CLEAR_VARS)
LOCAL_JACK_ENABLED := disabled ← 加上这一句后，就不会走jack编译
LOCAL_MODULE := android.hardware.gunder-V1.0-java-static
LOCAL_MODULE_CLASS := JAVA_LIBRARIES

intermediates := $(call local-generated-sources-dir, COMMON)

HIDL := $(HOST_OUT_EXECUTABLES)/hidl-gen$(HOST_EXECUTABLE_SUFFIX)

LOCAL_STATIC_JAVA_LIBRARIES := \
    android.hidl.base-V1.0-java-static \

#
# Build IGunder.hal
#
GEN := $(intermediates)/android/hardware/gunder/V1_0/IGunder.java
$(GEN): $(HIDL)
$(GEN): PRIVATE_HIDL := $(HIDL)
$(GEN): PRIVATE_DEPS := $(LOCAL_PATH)/IGunder.hal
$(GEN): PRIVATE_OUTPUT_DIR := $(intermediates)
$(GEN): PRIVATE_CUSTOM_TOOL = \
    $(PRIVATE_HIDL) -o $(PRIVATE_OUTPUT_DIR) \
    -Ljava \
    -randroid.hardware:hardware/interfaces \
    -randroid.hidl:system/libhidl/transport \
    android.hardware.gunder@1.0::IGunder

$(GEN): $(LOCAL_PATH)/IGunder.hal
    $(transform-generated-source)
LOCAL_GENERATED_SOURCES += $(GEN)
include $(BUILD_STATIC_JAVA_LIBRARY)

include $(call all-makefiles-under,$(LOCAL_PATH))

```

<https://blog.csdn.net/u013357557>

执行下面命令

- mmm hardware/interfaces/gunder/1.0/

执行完后生成了classes.jar。如图：

out > target > common > obj > JAVA_LIBRARIES > android.hardware.gunder-V1.0-java-static_intermediates >				
名称	修改日期	类型	大小	
anno	2018/11/23 18:50	文件夹		
classes	2018/11/23 18:50	文件夹		
jack-rsc	2018/11/23 16:54	文件夹		
classes.jack	2018/11/23 16:54	JACK 文件	40 KB	
classes.jar	2018/11/23 18:50	JAR 文件	18 KB	
classes-full-debug.jar	2018/11/23 18:50	JAR 文件	18 KB	
jack-rsc.java-source-list	2018/11/23 16:54	JAVA-SOURCE-L...	1 KB	
link_type	2018/11/23 16:54	文件	1 KB	

<https://blog.csdn.net/u013357557>

新建Android项目HIDLdemo，将classes.jar导入项目，MainActivity代码实现如下：

```

3 import android.hardware.gunder.V1_0.IGunder;
4 import android.os.Bundle;
5 import android.os.RemoteException;
6 import android.support.v7.app.AppCompatActivity;
7 import android.util.Log;
8 import android.view.View;
9 import android.widget.Toast;
10
11 public class MainActivity extends AppCompatActivity {
12     IGunder iGunderService;
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_hidl);
17         try {
18             iGunderService = IGunder.getService(); //获取服务
19         } catch (RemoteException e) {
20             e.printStackTrace();
21         }
22     }
23
24     public void hidlTest(View view) {
25         if (iGunderService != null) {
26             Log.d(tag: "gunder", msg: "service is connect.");
27             String s = null;
28             try {
29                 s = iGunderService.helloWorld(s: "Gunder Lin"); //调用HAL层接口
30             } catch (RemoteException e) {
31                 e.printStackTrace();
32             }
33             Log.d(tag: "gunder", s);
34             Toast.makeText(context: this, s, Toast.LENGTH_LONG).show();
35         }
36     }
37 }

```

https://blog.csdn.net/u013357557

添加Android.mk文件，然后将项目放到packages/apps/进行编译。Android.mk文件内容如下：

```

LOCAL_PATH := $(call my-dir)
include $(CLEAR_VARS)
LOCAL_PACKAGE_NAME := HIDLdemo
LOCAL_SRC_FILES := $(call all-java-files-under, src)
LOCAL_MODULE_TAGS := optional
LOCAL_STATIC_JAVA_LIBRARIES := android-support-v4
LOCAL_STATIC_JAVA_LIBRARIES += android-support-v7-appcompat
LOCAL_STATIC_JAVA_LIBRARIES += android-support-v7-gridlayout
#LOCAL_STATIC_JAVA_LIBRARIES += android-support-v13
LOCAL_STATIC_JAVA_LIBRARIES += android.hardware.gunder-V1.0-java-static
LOCAL_RESOURCE_DIR := $(LOCAL_PATH)/res
LOCAL_RESOURCE_DIR += prebuilts/sdk/current/support/v7/appcompat/res
LOCAL_RESOURCE_DIR += prebuilts/sdk/current/support/v7/gridlayout/res
LOCAL_CERTIFICATE := platform
#LOCAL_PRIVILEGED_MODULE := true
LOCAL_AAPT_FLAGS := --auto-add-overlay
LOCAL_AAPT_FLAGS += --extra-packages android.support.v7.appcompat:android.support.v7.gridlayout
#LOCAL_STATIC_JAVA_LIBRARIES += android.hardware.fingerprint-V1.0-java-static
#LOCAL_SRC_FILES := $(call all-subdir-java-files)
include $(BUILD_PACKAGE)

```

静态库

https://blog.csdn.net/u013357557

执行命令

- mmm packages/apps/HIDLdemo

会生成out/target/product/mek_8q/system/app/HIDLdemo/HIDLdemo.apk

10、测试客户端程序

执行下列命令：

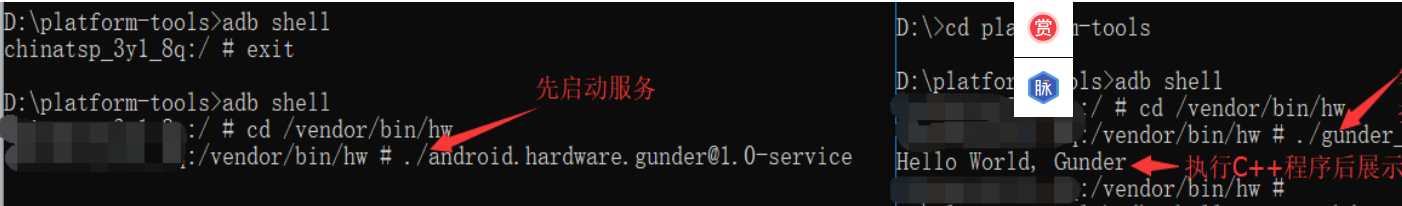
- adb push X:\IMX8_b0beta2\out\target\product\mek_8q\vendor\lib64\hw\android.hardware.gunder@1.0-impl.so /vendor/lib64/hw
- adb push X:\IMX8_b0beta2\out\target\product\mek_8q\system\lib64\android.hardware.gunder@1.0-impl.so /system/lib64
- adb push X:\IMX8_b0beta2\out\target\product\mek_8q\vendor\bin\hw\gunder_client /vendor/bin/hw
- adb push X:\IMX8_b0beta2\out\target\product\mek_8q\vendor\bin\hw\android.hardware.gunder@1.0-service /vendor/bin/hw
- adb push X:\IMX8_b0beta2\device\fs\evk_8mq\manifest.xml /vendor
- adb install -r X:\IMX8_b0beta2\out\target\product\mek_8q\system\app\HIDLdemo\HIDLdemo.apk



上面的命令是将需要的文件push到系统，方便调试。

C++客户端调试：

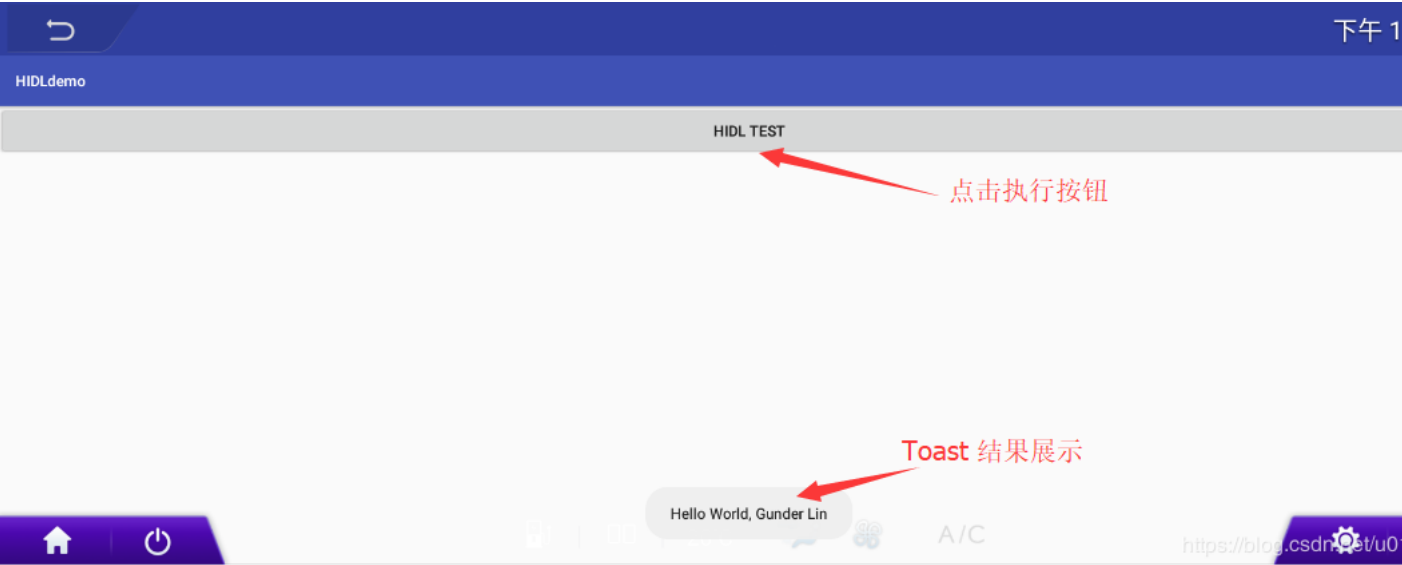
先启动./android.hardware.gunder@1.0-service服务，然后启动gunder_client程序，执行结果如下：



Android客户端执行：

先启动./android.hardware.gunder@1.0-service服务，然后通过命令拉起MainActivity界面

adb shell am start -n com.example.ytkj.hidldemo/.MainActivity，执行结果如下：



代码参考：<https://github.com/gunder1129/android-tool/tree/master/HIDL>

文章最后发布于: 201

有 0 个人打赏

Android HIDL 官方文档（一）——概述（Overview）

最近因为业务上的需求，老大让我先看着HIDL的官方文档学习学习。然而直接看英文文档还是很不习惯，就打算一边... 博文



阅读数 1万+

自：小石不识月，呼作...



想对作者说点什么

6

















Android9.0添加HIDL


1.编写hal文件并编译在hardware/interfaces/目录下创建test文件夹和基版本1.0，这个版本号分为两部分，major.m... 博文 来自： unbroken 阅读数 1017

Android HIDL 实例

前言：从AndroidHIDL详解 一文得知HIDL使用passthrough模式，为了与原来的HAL版本兼容。除了passthrough... 博文 来自： 私房菜之 --学--无-- 阅读数 2658

HIDL学习笔记之HIDL C++（第一天）

最近在学习HIDL，有很多的疑惑，在这里记录一下，加深自己的理解，以下部分大多来自官网。官网：https://sour... 博文 来自： 青青子衿 阅读数 1208



什么是工作流 怎么用

工作流

7211阅读

在Android 8.0之后版本上添加Hidl Service

目录1.编写hal文件并编译2.实现HidlInterface3.编写hidlservice4.配置manifest.xml5.hidlclient端调用5.1实现java... 博文 来自： anlory的专栏 阅读数 5015

AndroidO Treble架构下HIDL服务查询过程

通过前面的分析我们知道，Hal进程启动时，会向hw servicemanager进程注册hidl服务，那么当FrameworkServer... 博文 来自： 深入剖析Android... 阅读数 4089

IDL开发示例

转自《I D L开发》，仅供学习参考。 下载 02-28

学会了这些技术，你离BAT大厂不远了

每一个程序员都有一个梦想，梦想着能够进入阿里、腾讯、字节跳动、百度等一线互联网公司，由于身边的环境等原... 博文 来自： 平头哥的技术博文 阅读数 12万+

从入门到精通，Java学习路线导航

引言最近也有很多人来向我“请教”，他们大都是一些刚入门的新手，还不了解这个行业，也不知道从何学起，开始的... 博文 来自： wangweijun 阅读数 5万+

HIDL最全编译流程 - u013357557的专栏 - CSDN博客

HIDL最全编译流程 - 小菜琳 - CSDN博客

反转！BAT编程吸金榜来了，AI程序员刷爆了.....

2019年BAT等大厂积极布局AI领域，程序员转行学AI的门槛是什么？怎么转？

Android O HIDL的实现对接

Android8.0HIDL的实现对接1.HIDL的定义1.1.关于Android更新2.HIDL处于系统哪个部位及怎么通信的2.1.Android... 博文 来自： gh201030460222... 阅读数 9974

HIDL最全编译流程 - unbroken - CSDN博客

(六十七)HIDL 代码样式指南 - JT的专栏 - CSDN博客

分享靠写代码赚钱的一些门路

作者mezod，译者josephchang10如今，通过自己的代码去赚钱变得越来越简单，不过对很多人来说依然还是很难... 博文 来自： Python之禅的专栏 阅读数 4万+



StoneDemo

57篇文章

排名:千里之外

关注



慢慢的燃烧

1517篇文章

排名:412

关注



私房菜

308篇文章

排名:5000+

关注



By漫步

47篇文章

排名:千里之外

关注





HIDL概述 - u013357557的专栏 - CSDN博客

https://blog.csdn.net/u013357557/article/details/84561652

12/18

安卓8.0之 **HIDL** - Zach-Zona的博客 - CSDN博客

深入Android系统

From

IDL入门教程附实例代码

IDL初学者教程代码 IDL (Interactive Data Language) 交互式数据语言是进行二维及多维数据可视化分析及应用开发的理

HIDL实战笔记 - weixin_34295316的博客 - CSDN博客

HIDL实战笔记 - weixin_33720452的博客 - CSDN博客

程序员实用工具网站

目录1、搜索引擎2、PPT3、图片操作4、文件共享5、应届生招聘6、程序员面试题库7、办公、开发软件8、高清图...

羡慕**AI**高薪岗！为什么这类程序员不建议大家转型？
被众多开发工程师羡慕的AI程序员为啥这么高薪！30w只是白菜价有啥要求？

HIDL 简单介绍 - lei7143的专栏 - CSDN博客

hidl 编译 - weixin_34355881的博客 - CSDN博客

docker学习笔记

docker学习笔记常用的镜像:dockerpullanibali/pytorch:cuda-10.0Docker是什么？Docker是一个虚拟环境容器，...

我花了一夜用数据结构给女朋友写个H5走迷宫游戏

起因又到深夜了，我按照以往在csdn和公众号写着数据结构！这占用了我大量的时间！我的超越妹妹严重缺乏陪伴而...

(六十七) HIDL 代码样式指南

转载：代码样式指南HIDL代码样式类似于Android框架中的C++代码，缩进4个空格，并且采用混用大小写的文件名...

2019年10月中国编程语言排行榜

2019年10月2日，我统计了某招聘网站，获得有效程序员招聘数据9万条。针对招聘信息，提取编程语言关键字，并...

HIDL概述

HIDL背景Treble是GoogleAndroid团队的一项重大项目，意在Android操作系统框架在架构方面的一项重大改变，...

一本可陪伴一辈子的笔记本
可擦可写，可循环利用，支持OCR识别，让你的笔记本变得智能

HIDL 简单介绍

AndroidHAL类型 在此之前的ANDROID版本当中AndroidHAL没有什么特殊的特殊的，也么有什么分类，但是从an...

100 个网络基础知识普及，看完成半个网络高手

1) 什么是链接？链接是指两个设备之间的连接。它包括用于一个设备能够与另一个设备通信的电缆类型和协议。2) ...

别再翻了，面试二叉树看这 11 个就够了~

写在前边数据结构与算法：不知道你有没有这种困惑，虽然刷了很多算法题，当我去面试的时候，面试官让你手写一...

世界上最好的学习法：费曼学习法

你是否曾幻想读一遍书就记住所有内容？是否想学习完一项技能就马上达到巅峰水平？除非你是天才，不然这是不...

(六十三) HIDL C++ && HIDL Java

转载：1.https://source.android.com/devices/architecture/hidl-cpp/2.https://source.android.com/devices/a...

6

unbroken

05-31

下载

16万+

不脱发的程序猿

脉脉

从零开始深度学习102讲，送配套图书！

从原理到实战，美女科学家带你入门深度学习

C语言这么厉害，它自身又是用什么语言写的？

这是来自我的星球的一个提问：“C语言本身用什么语言写的？”换个角度来问，其实是：C语言在运行之前，得编译... [博文目录](#)：码农翻身

终于鸿蒙微内核弄懂了-程序员和鼓励师的合作

当鸿蒙OS宣布开源的时候，各种空洞的炒作，几乎把国产操作系统的技术本质掩盖了，虽然笔者没亲眼见过鸿蒙的代... [Python爱好者的专栏](#)

Android HIDL 简介

AndroidHIDL简介Qidi 2017.08.01(Markdown&Haroopad)注意：本文基于Android8.0进行分析。1、HIDL的概... 博 < 目：一程山水一程歌

JAVA-快速了解线程池的基本原理

前言说起线程池大家肯定不会陌生，在面试中属于必问的问题之一，特别是对于高并发有较高要求的企业，基本是核... 博... 目：我在风花雪月里等你

漫画 | 外行对程序员误会有多深！

作者：阿波、纯洁的微笑漫画：宁州枪手程序员如今已经发展成社会的主流职业，以至于街头的王大妈李大爷都能说... 博文 不日：纯洁的微笑

推动全社会公益氛围形成，使公益与空气和阳光一样触手可及。

公益缺你不可，众多公益项目等你PICK——百度公益 让公益像「空气和阳光」一样触手可及！
gongyi.baidu.com

学Linux到底学什么

来源：公众号【编程珠玑】作者：守望先生网站：<https://www.yanbinghu.com/2019/09/25/14472.html>前言我们... 博文 来自：守望的博客-编程珠玑

Java 网络爬虫，就是这么的简单

这是Java网络爬虫系列文章的第一篇，如果你还不知道Java网络爬虫系列文章，请参看学Java网络爬虫，需要哪些基... 博文 来自：平头哥的技术博文

对计算机专业来说学历真的重要吗？

我本科学校是渣渣二本，研究生学校是985，现在毕业五年，校招笔试、面试，社招面试参加了两年了，就我个人的... 博文 来自： 启舰

为什么程序员在学习编程的时候什么都记不住？

在程序员的职业生涯中，记住所有你接触过的代码是一件不可能的事情！那么我们该如何解决这一问题？作者|Dylan... 博文 来自： [CSDN资讯](#)

30秒内便能学会的30个超实用Python代码片段

许多人在数据科学、机器学习、web开发、脚本编写和自动化等领域中都会使用Python，它是一种十分流行的语言。... [博文](#) 来自: [读芯术的博客](#)



四种方法快速减肚子赘肉

如何去掉肚子上的赘肉

面试官：兄弟，说说基本类型和包装类型的区别吧

Java的每个基本类型都对应了一个包装类型，比如说int的包装类型为Integer，double的包装类型为Double。基本... [博文](#) 来自： [沉默王二](#)

HIDL (—)

原文见<https://source.android.com/devices/architecture/hidl/OverView>HAL接口定义语言(HIDL)是一种接口描述... 博文 来自: 少年的此间

Android HIDL 中 hidl-gen使用

前言在 [AndroidHIDL详解](#) 一文提到HIDL使用的整个过程都是跟其工具hidl-gen分不开，这一篇来详细分析hidl-gen... [博文](#) 来自: [私房菜之--学--无--悔](#)

Linux 给我的七个宝贵教训

在日常使用过程中，作为时下主流操作系统之一的Linux，还存在哪些坑？以及从它的应用过程中，我们还可以挖掘... [博文](#) 来自：[CSDN资讯](#)

失败程序员的十年总结

十年到底有多长？当我回顾过去的十年，发现好短，可以讲的事情没有几件，而且都是坏事；当我畅想未来的十年，...

程序员那些必须掌握的排序算法(下)

接着上一篇的排序算法，我们废话不多，直接进入主题。1.快速排序 快速排序（Quicksort）是对冒泡排序的一种改... [博文](#)

程序员真是太太太太有趣了！！

网络上虽然已经有了很多关于程序员的话题，但大部分人对这个群体还是很陌生。我们在谈论程序员的时候，究竟该...

史上最详细的IDEA优雅整合Maven+SSM框架（详细思路+附带源码）

网上很多整合SSM博客文章并不能让初探ssm的同学思路完全的清晰，可以试着关掉整合教程，摇两下头骨，哈一大...

知乎上 40 个有趣回复，很精辟很提神

点击蓝色“五分钟学算法”关注我哟加个“星标”，天天中午 12:15，一起学算法作者 |佚名来源 |网络整理，版权归...

实现 Java 本地缓存，该从这几点开始

缓存，我相信大家对它一定不陌生，在项目中，缓存肯定是必不可少的。市面上有非常多的缓存工具，比如 Redis、...



大型在线商城系统源码
b2b2c商城源码
9259阅读

揭开 Python 内存分配时的小秘密！

作者 | 豌豆花下猫 责编 | 胡巍巍 Python 中的sys模块极为基础而重要，它主要提供了一些给解释器使用（或由它维...

让程序员崩溃的瞬间（非程序员勿入）

今天给大家带来点快乐，程序员才能看懂。 来源：https://zhuanlan.zhihu.com/p/47066521 1. 公司实习生找 Bug...

用Python分析2000款避孕套，得出这些有趣的结论

到现在为止，我们的淘宝教程已经写到了第四篇，前三篇分别是： 第一篇：Python模拟登录淘宝，详细讲解如何使...

做好以下四点，拒做“空心”程序员

01、注重原理性知识 现在的互联网环境下，注重原理性知识学习的程序员越来越少，特别是在这种培训机构大爆炸...

技术人员要拿百万年薪，必须要经历这9个段位

很多人都问，技术人员如何成长，每个阶段又是怎样的，如何才能走出当前的迷茫，实现自我的突破。所以我结合我...



什么是工作流 怎么用
工作流
7211阅读

进程和线程的区别(超详细)

进程和线程 进程 一个在内存中运行的应用程序。每个进程都有自己独立的一块内存空间，一个进程可以有多个线程...

第二弹！python爬虫批量下载高清图

文章目录前言下载免费高清图下载带水印的精选图代码与总结 前言 在上一篇写文章没高质量配图？python爬虫绕...

面试官，不要再向我三次握手和四次挥手

三次握手和四次挥手是各个公司常见的考点，也具有一定的水平区分度，也被一些面试官作为热身题。很多小伙伴说...

为什么说 Web 开发永远不会退出历史舞台？

早在 PC 崛起之际，Web 从蹒跚学步一路走到了主导市场的地位，但是随着移动互联网时代的来临，业界曾有不少人...

Java 爬虫遇到需要登录的网站，该怎么办？

这是 Java 网络爬虫系列博文的第二篇，在上一篇 Java 网络爬虫，就是这么的简单 中，我们简单的学习了一下如何...

nginx学习，看这一篇就够了：下载、安装。使用：正向代理、反向代理、负载均衡。常用命令和配置文件


文章目录前言一、nginx简介1. 什么是 nginx 和可以做什么事情2.Nginx 作为 web 服务器3. 正向代理4. 反向代理5. ...

动画：用动画给面试官解释 TCP 三次握手过程


作者 | 小鹿 来源 | 公众号：小鹿动画学编程 写在前边 TCP 三次握手过程对于面试是必考的一个，所以不但要掌握 TC...


500行代码，教你用python写个微信飞机大战


这几天在重温微信小游戏的飞机大战，玩着玩着就在思考人生了，这飞机大战怎么就可以做的那么好，操作简单，简...




6



















阅读数 3万+

博文

阅读数 1万+

博文

阅读数 4万+

博文

阅读数 2918

博文

阅读数 1444

博文

阅读数 18万+

博文

阅读数 3万+

博文

阅读数 1万+

博文

阅读数 7377

博文

阅读数 801

博文

阅读数 2万+

博文

阅读数 12万+

博文

阅读数 8383

博文

阅读数 8636

博文

阅读数 1111

博文

阅读数 2万+

博文

阅读数 4万+

博文

从月薪3K的中专生，到身家千万的CTO！人生最大的对手，就是自己

关注“技术领导力”博客，独家大厂干货推送 文/Daniel.W David坐在我对面，窗外是梦境般的外滩夜景，繁星点点...

这应该是把计算机网络五层模型讲的最好是文章了，看不懂你打我

聊地：用心写好每一篇文章！前言 天各一方的两台计算机是如何通信的呢？在成千上万的计算机中，为什么一台计...

c#比较图片是否相同 c#.net 中文乱码 c# 时间排序 c# 必备书籍 c#里调用窗口 c#异步网络通信 c#泛型约束 c#与结构 c#检测键盘输入

堆排序c# c# 标注对象字段后

没有更多推荐了，[返回首页](#)

阅读量 1万+

博文


阅读量 1万+

博文

打赏

脉脉

©2019 CSDN 皮肤主题: 大白 设计师: CSDN官方博客



Gunder

TA的个人主页 >

私信

关注

原创19

粉丝13

获赞12

评论0

访问2万+

等级: 博客 3

周排名: 9万+

积分: 406

总排名: 16万+

最新文章

数据结构--单链表C/C++


Android JNI 访问Java中的成员变量及非native方法


Android中的JNI使用指南一


HIDL概述


常用的两种handler调用方法


分类专栏

 android 18篇

 C/C++ 7篇

 JNI 3篇

 AIDL 4篇

 cmd 1篇

展开

归档

2019年3月 2篇

2019年2月 1篇

2018年11月 2篇

2018年10月 6篇

2018年9月 2篇

2018年8月 6篇

展开

热门文章

https://blog.csdn.net/u013357557/article/details/84561652

17/18

HIDL最全编译流程

阅读数 3869

常用的两种handler调用方法

阅读数 3290

Android中C/C++的日志打印

阅读数 3154

Android 删除sdcard目录中的某些目录文件

阅读数 2166

在Android系统中实现AIDL接口回调

阅读数 1778


6

















云服务器1核1G_低至0.16j

百度智能云双11感恩钜惠,云数
RDS秒杀价35元/3个月,BOS存
CDN流量包11元秒杀



程序人生



CSDN资讯

 QQ客服


 kefu@csdn.net

 客服论坛

 400-660-0108

工作时间 8:30-22:00

关于我们 招聘 广告服务 网站地图

 百度提供站内搜索 京ICP备19004658号

©1999-2019 北京创新乐知网络技术有限公司

网络110报警服务 经营性网站备案信息

北京互联网违法和不良信息举报中心

中国互联网举报中心 家长监护 版权申诉



