IDO-EVB3568-V1 - Android 应用开发说明

IDO-EVB3568-V1

Android 应用开发说明

深圳触觉智能科技有限公司

www.industio.cn

文档修订历史

版本	修订内容	修订	审核	日期
V1.0	创建文档			2022/03/17
V1.1	增加			2022/03/18

一、ADB 调试

ADB（Android Debug Bridge）是 Android SDK 里的一个工具，用这个工具可以操作管理Android模拟器或真实的 Android 设备。主要功能有：

- 运行设备的 shell（命令行） 管理模拟器或设备的端口映射
- 计算机和设备之间上传/下载文件
- 将本地 apk 软件安装至模拟器或 Android 设备
- 网络 ADB：主机通过有线/无线网络（同一局域网）连接到 STB 设备
- USB ADB：主机通过 USB 线连接到 STB 设备

1.1 USB ADB 使用说明

连接步骤如下：

- PC安装adb工具
- 设备已经运行 Android 系统，设置->开发者选项->已连接到计算机打开，usb 调试开关打开。（默认打开）
- PC 主机只通过 USB 线连接到机器 USB OTG 口，然后电脑通过如下命令与设备相连。
- adb shell
- 测试是否连接成功，运行 adb devices 命令，如果显示机器的序列号，表示连接成功。

1.2 网络 ADB 使用说明

如果你需要使用网络 ADB 来调试设备，必须要满足如下条件：

- 设备上面首先要有网口，或者通过 WiFi 连接网络。
- 设备和研发机（PC 机）已经接入局域网，并且设备设有局域网的 IP 地址。
- 要确保研发机和设备能够相互 ping 得通。
- 研发机已经安装了 ADB。
- 确保 Android 设备中 adbd 进程（ADB 的后台进程）已经运行。adbd 进程将会监听端口 5555 来进行 ADB 连接调试。

本节假设设备的 IP 为 192.168.1.5，下文将会用这个 IP 建立 ADB 连接，并调试设备。

- 首先 Android 设备需要先启动，如果可以话，可以确保一下 adbd 启动(ps 命令查看)。
- 在 PC 机的 cmd 中，输入：`adb connect 192.168.1.5:5555`
- 如果连接成功会进行相关的提示，如果失败的话，可以先执行 `adb kill-server` 命令，然后重试连接。
- 如果连接已经建立，在研发机中，可以输入 ADB 相关的命令进行调试了。比如 `adb shell`，将会通过 TCP/IP 连接设备上面。和 USB 调试是一样的。
- 调试完成之后，在研发机上面输入如下的命令断开连接：`adb disconnect 192.168.1.5:5555`

1.3 ADB常用命令详解

1.3.1 查看设备情况

adb devices

返回的结果为连接至开发机的 Android 设备的序列号或是 IP 和端口号（Port）、状态。

1.3.2 安装 APK

将指定的 APK 文件安装到设备上：

adb install <apk 文件路径> 示例如下： adb install F:\WishTV\WishTV.apk

重新安装应用：

adb install −r <apk 文件路径> 示例如下： adb install −r F:\WishTV\WishTV.apk||

1.3.3 卸载 APK

完全卸载：

adb uninstall <package> 示例如下： adb uninstall com.wishtv

1.3.4 使用 rm 移除 APK 文件：

adb shell rm <filepath> 示例如下： adb shell rm system/app/WishTV.apk

示例说明：移除/system/app目录下的WishTV.apk文件。

1.3.5 进入设备的 shell

adb shell

1.3.6 从电脑上传文件到设备

adb push <本地路径><远程路径> 示例如下： adb push F:\WishTV\WishTV.apk /system/app

示例说明：将本地WishTV.apk文件上传到 Android 系统的system/app目录下。

1.3.7 从设备下载文件到电脑

adb pull <远程路径><本地路径> 示例如下： adb pull /system/app/Contacts.apk F:\

示例说明：将 Android 系统system/app目录下的文件或文件夹下载到本地 F:\ 目录下。

1.3.8 查看设备的系统信息

adb shell getprop

在 adb shell 下查看设备系统信息的具体命令。

二、设备接口使用方法

见此链接：https://industio.yuque.com/docs/share/04b72690-54c9-41ce-8215-60914dcb0e65?# 《IDO-EVB3568 Android接口说明_v1.2》

三、自定义API使用方法

四、apk系统签名使用方法

apk系统签名是为了调用系统的一些功能，签名有2种方法：

第一种：直接使用build/target/product/security里面的platform.x509.pem、platform.pk8

📎platform_key.zip

java -jar signapk.jar -w platform.x509.pem platform.pk8 input.apk out_signed.apk

📎platform_sign_method.rar

第二种：使用jks文件在Android Studio给app签名

生成jks文件：`./keytool-importkeypair -k demo.jks -p 123456 -pk8 platform.pk8 -cert platform.x509.pem -alias demo`

📎keytool-importkeypair-master.zip

参考：https://blog.csdn.net/tcbhaiqiang/article/details/90763809

五、root 固件使用方法

app中参考方法：

```java
public static boolean RootCommand(String command)
{
    Process process = null;
    DataOutputStream os = null;
    try
    {
        process = Runtime.getRuntime().exec("su");
        os = new DataOutputStream(process.getOutputStream());
        os.writeBytes(command + "\n");
        os.writeBytes("exit\n");
        os.flush();
        process.waitFor();
    } catch (Exception e)
    {
        Log.d("RootCommand", "Exception:" + e.getMessage());
        return false;
    } finally
    {
        try
        {
            if (os != null)
            {
                os.close();
            }
            process.destroy();
        } catch (Exception e)
        {
        }
    }
    return true;
}
```

六、framework.jar 使用方法

降out\target\common\obj\JAVA_LIBRARIES\framework_intermediates\\classes-header.jar

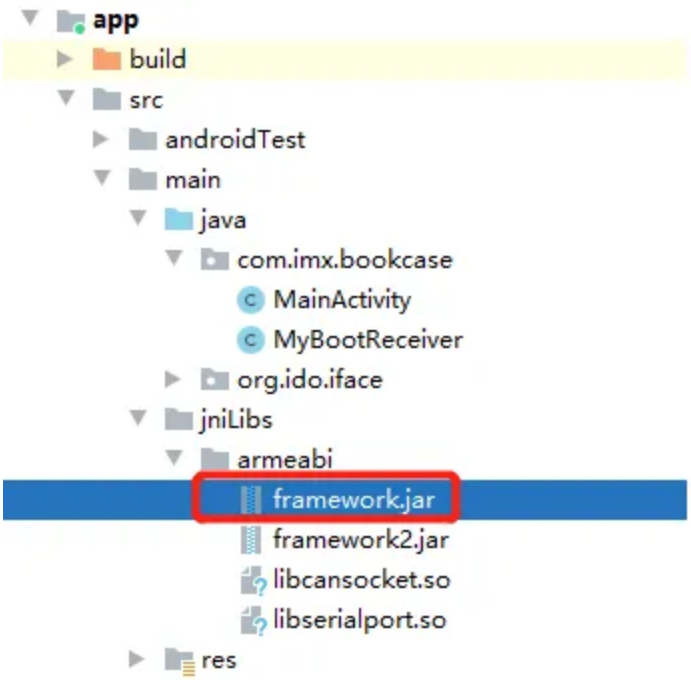
改名为：framework.jar

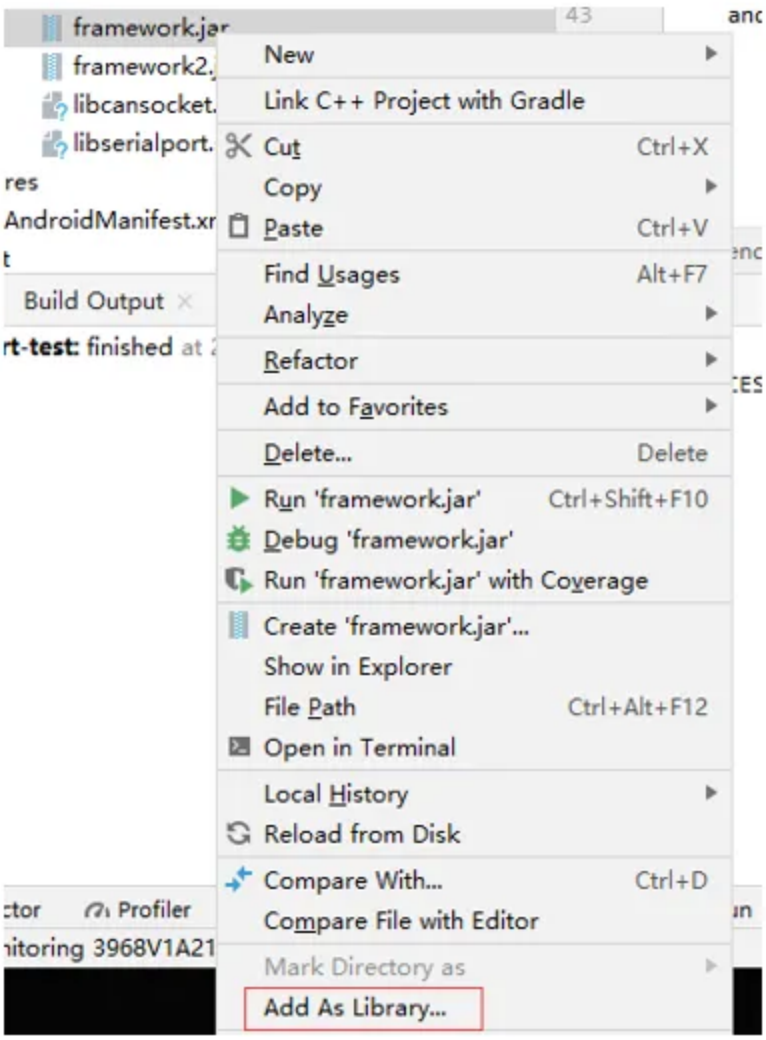
📎framework.zip解压后为framework.jar

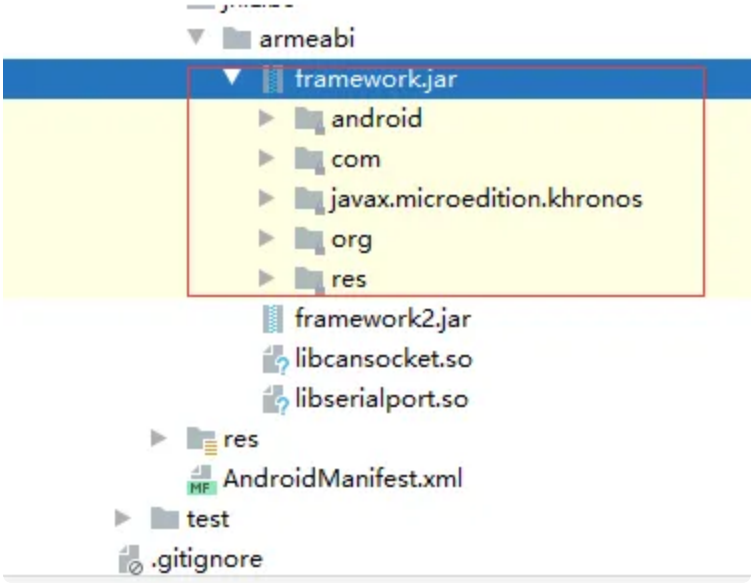
在Android Studio中使用方法参考：

导入AS:

将入jar放到下面：


app
build
src
androidTest
main
java
com.imx.bookcase
MainActivity
MyBootReceiver
org.ido.iface
jniLibs
armeabi
framework.jar
framework2.jar
libcansocket.so
libserialport.so
res


framework.jar
framework2.
libcansocket.
libserialport.
res
AndroidManifest.xr
Build Output
rt-test: finished at
New
Link C++ Project with Gradle
Cut Ctrl+X
Copy
Paste Ctrl+V
Find Usages Alt+F7
Analyze
Refactor
Add to Favorites
Delete... Delete
Run 'framework.jar' Ctrl+Shift+F10
Debug 'framework.jar'
Run 'framework.jar' with Coverage
Create 'framework.jar'...
Show in Explorer
File Path Ctrl+Alt+F12
Open in Terminal
Local History
Reload from Disk
Compare With... Ctrl+D
Compare File with Editor
Mark Directory as
Add As Library...
Profiler



调用编译时的优先级：

https://blog.csdn.net/miracast/article/details/78540657

七、双屏异显开发说明

📎Rockchip Android11 异显开发指南.pdf

📎DualScreenDemo.zip

八、扩展IO使用说明

J14为多功能IO扩展接口，默认配置为GPIO功能，使用方法如下：

序号	GPIO编号	控制接口
1	5V	
2	GND	
3	146	设置方向：/sys/class/gpio/gpio146/direction 读写电平值：/sys/class/gpio/gpio146/value
4	147	设置方向：/sys/class/gpio/gpio147/direction 读写电平值：/sys/class/gpio/gpio147/value
5	149	设置方向：/sys/class/gpio/gpio149/direction 读写电平值：/sys/class/gpio/gpio149/value
6	150	设置方向：/sys/class/gpio/gpio150/direction 读写电平值：/sys/class/gpio/gpio150/value
7	22	设置方向：/sys/class/gpio/gpio22/direction 读写电平值：/sys/class/gpio/gpio22/value
8	GND	

控制方法示例

作为输出

```shell
#设置为输出
echo out > /sys/class/gpio/gpio22/direction
#设置高电平
echo 1 > /sys/class/gpio/gpio22/value
#设置低电平
echo 0 > /sys/class/gpio/gpio22/value
```

作为输入

```shell
#设置为输入
echo in > /sys/class/gpio/gpio22/direction
#读取电平值
cat /sys/class/gpio/gpio22/value
```

以上为命令行控制示例，如果是通过代码控制，则将echo方式修改为使用write接口对设备节点进行写数据，通过将cat修改为read接口来读取设备节点获得电平值。