一般的用户app，是没有直接操作硬件的权限的。比如关机和重启。

默认的SDK并没有提供应用开发者直接的Android系统关机或重启的API接口，一般来讲，实现Android系统的关机或重启，需要较高的权限（系 统权限甚至Root权限）。所以，在一般的APP中，如果想要实现关机或重启功能，要么是在App中声明系统权限，要么是通过某种“间接”的方式，比如广 播或反射，来间接实现系统关机或重启。再者，就是放在[源码](http://www.2cto.com/ym" \t "http://www.2cto.com/kf/201412/_blank)环境中进行编译，这样做有一个好处，就是可以直接调用Android中不公开的API，这是Eclipse＋SDK没法达到的效果。

将APP提升至系统权限

1. 在AndroidMenifest.xml中的manifest下添加:android:sharedUserId="android.uid.system"
2. 添加关机权限:<uses-permission android:name="android.permission.SHUTDOWN"/>

这时会发现eclipse报错，说“Permission is only granted to system apps”，只有系统级应用才能用这个权限。这样直接编译是通不过的，所有得把项目放在源码中进行编译，由于需要在源码中编译项目，所以需要为项目编写mk文件，在项目根目录下添加Android.mk文件，内容如下所示：

LOCAL\_PATH:= $(call my-dir)//

include $(CLEAR\_VARS)

LOCAL\_MODULE\_TAGS := optional

LOCAL\_SRC\_FILES := $(call all-java-files-under, src)

LOCAL\_PACKAGE\_NAME := PowerActionDemo

LOCAL\_CERTIFICATE := platform

include $(BUILD\_PACKAGE)

1. APK的源码放到安卓源码的packages/apps/目录下，用mm进行编译

备注：

系统中所有使用android.uid.system作为共享UID的APK，都会首先在manifest节点中增加 android:sharedUserId="android.uid.system"，然后在Android.mk中增加 LOCAL\_CERTIFICATE := platform。可以参见Settings等

系统中所有使用android.uid.shared作为共享UID的APK，都会在manifest节点中增加 android:sharedUserId="android.uid.shared"，然后在Android.mk中增加 LOCAL\_CERTIFICATE := shared。可以参见Launcher等

系统中所有使用android.media作为共享UID的APK，都会在manifest节点中增加 android:sharedUserId="android.media"，然后在Android.mk中增加LOCAL\_CERTIFICATE := media。可以参见Gallery等。

另外，应用创建的任何文件都会被赋予应用的用户标识，并且正常情况下不能被其他包访问。当通过 getSharedPreferences（String，int）、openFileOutput（String、int）或者 openOrCreate Database（String、int、SQLiteDatabase.CursorFactory）创建一个新文件时，开发者可以同时或分别使用 MODE\_WORLD\_READABLE和MODE\_WORLD\_RITEABLE标志允许其他包读/写此文件。当设置了这些标志后，这个文件仍然属于自 己的应用程序，但是它的全局读/写和读/写权限已经设置，所以其他任何应用程序可以看到它。

关于签名：

build/target/product/security目录中有四组默认签名供Android.mk在编译APK使用：

1、testkey：普通APK，默认情况下使用。

2、platform：该APK完成一些系统的核心功能。经过对系统中存在的文件夹的访问测试，这种方式编译出来的APK所在进程的UID为system。

3、shared：该APK需要和home/contacts进程共享数据。

4、media：该APK是media/download系统中的一环。

应用程序的Android.mk中有一个LOCAL\_CERTIFICATE字段，由它指定用哪个key签名，未指定的默认用testkey.

对于使用eclipse编译的apk，可以使用signapk.jar来手动进行签名，其源码在build/tools/signapk下，编译后 在out/host/linux-x86/framework/signapk.jar，也可以从网上下载。使用方法，以platform为例：java -jar ./signapk　platform.x509.pem platform.pk8 input.apk output.apk  (platform.x509.pem platform.pk8在build/target/product/security获取)

另一种给应用提权的方法：

倘若应用程序需要对/dev/xxx（设备文件，如video、usb、serial等）进行读写操作，就需要提升其权限。提权方法不唯一，需要根据具体需求情况而选择。归根结底，终究都落到chmod 777 /dev/xxx 上

要么在安卓源文件的init.rc下直接修改设备权限