|  |
| --- |
| 上海与德通讯技术有限公司 |
| E183L客制化总结 |
| 需求制作&验证 |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Author | Comments |
| 1.0 | 2016/9/01 | 黄亚玲  刘扬窑 | Add E183L 客制化需求 |
| 1.1 | 2016/09/08 | 李经纬 | 更新SPN、如何区别和取消漫游、STK名称 |

E183L常用客制化

[1、 项目配置 3](#_Toc457983450)

[(1) 在wind/config/下面添加CONFIG\_E183L\_XXX.mk文件 3](#_Toc457983451)

[(2) 在wind/custom\_files/device/ginreen/E183L下面添加versionXXX文件,内容如下： 3](#_Toc457983452)

[2、客制化资源 4](#_Toc457983453)

[(1) Logo 4](#_Toc457983454)

[(2)集成开机动画、关机动画、开机铃声、关机铃声 4](#_Toc457983455)

[(3) 开关机动画的制作 4](#_Toc457983456)

[(4) 默认铃声设置 5](#_Toc457983457)

[(5)添加项目宏等 5](#_Toc457983458)

[(6) overlay资源控制 6](#_Toc457983459)

[(7) 情景模式配置（默认音量调节，默认震动音等开启） 6](#_Toc457983460)

[3、APN/SPN/ Voicemail 7](#_Toc457983461)

[(1) APN 7](#_Toc457983462)

[(2) SPN 7](#_Toc457983463)

[(3) Voicemail 7](#_Toc457983464)

[(4) 关于虚拟运营商: 7](#_Toc457983465)

[4、默认语言和输入法 8](#_Toc457983466)

[(1) 项目添加语言和默认语言 8](#_Toc457983467)

[(2) 根据SIM卡的MCC自动识别语言 8](#_Toc457983468)

[(3) 输入法 8](#_Toc457983469)

[5、桌面布局 10](#_Toc457983470)

[6、Browser/Email 11](#_Toc457983471)

[(1) Browser 11](#_Toc457983472)

[(2) Email 11](#_Toc457983473)

[7、时区 12](#_Toc457983474)

[(1) 默认时区配置 12](#_Toc457983475)

[(2)NITZ默认打开设置 12](#_Toc457983476)

[(3)默认日期格式 12](#_Toc457983477)

[(4) 24小时配置 12](#_Toc457983478)

[8、GMS 12](#_Toc457983479)

[9、内置APK的添加 13](#_Toc457983480)

[(1) 预装无源码APK 13](#_Toc457983481)

[(2) 预装有源码APK 15](#_Toc457983482)

[10、锁网 16](#_Toc457983483)

[(1) Modem内置锁网值 18](#_Toc457983484)

[(2) 工具修改Modem配置锁网 18](#_Toc457983485)

[11、紧急拨号 18](#_Toc457983486)

[12、网络漫游配置 20](#_Toc457983487)

[(1) 何为漫游 20](#_Toc457983488)

[(2) 如何区别和取消漫游 21](#_Toc457983489)

[13、内置联系人 22](#_Toc457983490)

[(1)如何内置联系人 22](#_Toc457983491)

[(2)号码匹配 22](#_Toc457983492)

[14、短信相关 22](#_Toc457983493)

[(1) 短信中心号码 22](#_Toc457983494)

[(2) 内置小区广播号码 23](#_Toc457983495)

[15、STK名称 23](#_Toc457983496)

[16、FOTA 26](#_Toc457983497)

[17、文件管理和视频播放 28](#_Toc457983498)

# 项目配置

## 在wind/config/下面添加CONFIG\_E183L\_XXX.mk文件

WIND\_PORDUCT\_NAME=P635T36

WIND\_DEVICE\_NAME=ZTE\_BLADE\_A610

WIND\_PRODUCT\_BRAND=ZTE

WIND\_PRODUCT\_MODEL=ZTE BLADE A610

WIND\_PRODUCT\_MANUFACTURER=ZTE

WIND\_PORDUCT\_BOARD=MT6735

WIND\_PORDUCT\_TIMEZONE=Europe/London

WIND\_PORDUCT\_NOTIFICATION\_SOUND=Glow.ogg

WIND\_PORDUCT\_ALARM\_ALERT=

WIND\_PORDUCT\_RINGTONE=The\_party.ogg

#WIND\_CUS\_EMERGENCY\_NUM=112,911

#WIND\_CUS\_EMERGENCY\_NUM\_EXT=112,911

WIND\_PORDUCT\_DATEFORMAT=dd-MM-yyyy

WIND\_CUS\_MIN\_MATCH=7

WIND\_CUS\_EMAIL\_SIGNATURE=

WIND\_APN\_CONF=apns-conf-E183L.xml

WIND\_SPN\_CONF=spn-conf-E183L.xml

WIND\_VSPN\_CONF\_BY\_SPN=virtual-spn-conf-by-efspn-E183L.xml

WIND\_VSPN\_CONF\_BY\_IMSI=

WIND\_VSPN\_CONF\_BY\_PNN=

WIND\_VSPN\_CONF\_BY\_GIDL=virtual-spn-conf-by-efgid1-E183L.xml

WIND\_VOICEMAIL\_CONF=voicemail-conf-E183L.xml

WIND\_MTP\_NAME=ZTE BLADE A610

WIND\_HANDHELD\_CORE\_HW=handheld\_core\_hardware\_E183L.xml

## 在wind/custom\_files/device/ginreen/E183L下面添加versionXXX文件,内容如下：

INVER=ZTE\_E183L\_K000\_V1.0B01

OUTVER=ZTE\_E183L\_K000\_V1.0

#PROVINCE=

#OPERATOR=

#INCREMENTALVER=

# 2、客制化资源

## (1) Logo

开机logo分为uboot logo和kernel logo，也就是开机时显示的第一屏：

1. 开机时通过wind/config/config\_E183L\_xxx.mk中的BOOT\_LOGO= e183l\_hd720来定义开机第一屏图片。\_hd720这个后缀名需要与device/ginreen/E183L/ProjectConfig.mk:BOOT\_LOGO =cu\_lte\_hd720 保持一致；
2. 并且vendor/mediate/proprietary/bootable/bootloader/lk/project/E183L.mk文件中的BOOT\_LOGO的值设为e183l\_hd720；
3. 在vendor/mediate/proprietary/bootable/bootloader/lk/dev/logo下新增e183l\_hd720目录，添加图片。(图片的分辨率要和ProjectConfig.mk中的宽高值一致，不然开机第一屏不会显示)
4. 在vendor/mediate/proprietary/external/libshowlogo中cust\_display.h文件分辨率相应位置添加|| defined(E183L\_HD720)

## (2)集成开机动画、关机动画、开机铃声、关机铃声

我们客制化开机动画的路径：frameworks/base/data/sounds目录

资源文件路径：frameworks/base/data/sounds/E183L

编译脚本：修改frameworks/base/data/sounds/E183LAudio.mk文件

## 开关机动画的制作

1. 定制开机动画资源需求:动画帧不超过50张，png格式，位深24位，分辨率参考产品需求；
2. 开关机动画资源文件路径：frameworks/base/data/sounds/E183L/generic创建bootanimation和shutanimation文件夹，存入开关机动画资源；
3. 在frameworks/base/data/sounds创建项目编译文件E183LAudio.mk文件

LOCAL\_PATH:= frameworks/base/data/sounds/E183L/generic

#bootanimation

PRODUCT\_COPY\_FILES += \

$(LOCAL\_PATH)/bootanimation/bootanimation.zip:system/media/bootanimation.zip

1. 修改frameworks/base/data/sounds目录下的AllAudio.mk文件，修改资源文件路径：

ifeq ($(TARGET\_PRODUCT),full\_E183L)

ifeq ($(strip $(WIND\_PROJECT\_NAME\_CUSTOM)), E183L\_MAL)

$(call inherit-product, frameworks/base/data/sounds/E183LMALAudio.mk)

elseifeq ($(strip $(WIND\_PROJECT\_NAME\_CUSTOM)), E183L\_IN)

$(call inherit-product, frameworks/base/data/sounds/E183LINAudio.mk)

else

$(call inherit-product, frameworks/base/data/sounds/E183LAudio.mk)

Endif

else

LOCAL\_PATH := frameworks/base/data/sounds

开关机动画的制作：

分别创建名为“part0”和“part1”的文件夹以及一个名为“desc.txt”文件。“part0”中存储动画的第一阶段的资源图片，“part1”存储第二阶段的资源图片，注意图片为png格式。播放控制由“desc.txt”指定，内容如下：

   48085412

   p 1 0 part0

   p 0 0 part1

各参数功能如下： ( 注意：desc.txt文本内容必须用单个空格隔开，且不能有多余空行。)

|  |  |  |  |
| --- | --- | --- | --- |
| 480 | 854 | 12 |  |
| 宽 | 高 | 每秒播放帧数 |  |
| p | 1 | 0 | part0 |
| 标志符 | 循环次数 | 阶段切换间隔时间 | 对应目录名 |
| p | 0 | 0 | part1 |
| 标志符 | 循环次数 | 阶段切换间隔时间 | 对应目录名 |

最后，将这三个组件通过存储压缩的方式压缩为bootanimation.zip文件即制作完成。

关机动画的制作也是如此，将这三个组件通过存储压缩的方式压缩为shutanimation.zip文件

## 默认铃声设置

1. 定制开关机铃声资源需求：ogg格式或者mp3格式，铃声长度不超过5s
2. 在frameworks/base/data/sounds/E183L/generic/bootaudio添加资源文件bootaudio.mp3
3. 编译脚本frameworks/base/data/sounds/E183LAudio.mk

#bootaudio

PRODUCT\_COPY\_FILES += \

$(LOCAL\_PATH)/bootaudio/bootaudio.mp3:system/media/bootaudio.mp3

来电铃声copy到system/media/audio/ringtones

通知铃声copy到system/media/audio/notifications

闹钟铃声copy到system/media/audio/alarms

默认铃声的配置：wind/config/CONFIG\_E183L\_XXX.mk

WIND\_PORDUCT\_NOTIFICATION\_SOUND=Glow.ogg

WIND\_PORDUCT\_ALARM\_ALERT=

WIND\_PORDUCT\_RINGTONE=The\_party.ogg

## (5)添加项目宏等

以添加添加开关WIND\_DEF\_OPTR\_E183L\_XXX

在wind/config/CONFIG\_E183L\_XXX.mk添加WIND\_DEF\_OPTR\_E183L\_XXX=no/yes

在wind/custom\_files/device/ginreen/E183L/device.mk添加

ifeq ($(strip $(WIND\_DEF\_OPTR\_E183L\_XXX)),yes)

PRODUCT\_PROPERTY\_OVERRIDES += ro.wind\_def\_optr\_e183l\_xxx=1

在上层java代码中通过FeatureOption. WIND\_DEF\_OPTR\_E183L\_XXX进行客制化，注意导包：



注：FeatureOption中能够添加的开关只能是boolean类型

## (6) overlay资源控制

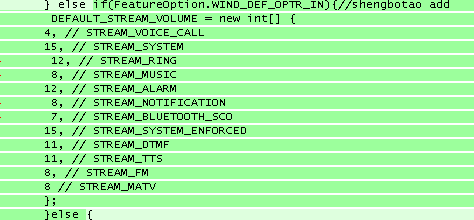
在overlay对应项目目录下修改需要资源文件，路径device/ginreen/E183LXXX/overlay/，保持修改的文件的路径与源码路径一致。



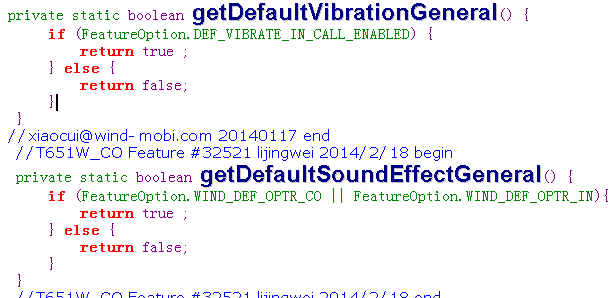
## (7) 情景模式配置（默认音量调节，默认震动音等开启）

1. 默认音量调节





（2）默认震动音等声音的开启



以上是震动和触摸屏的默认声音修改。

# 3、APN/SPN/ Voicemail

## (1) APN

1.配置路径

vendor/mediatek/proprietary/frameworks/base/telephony/etc/apns-conf-E183L\_xxx.xml

2.编译路径：wind/config/ CONFIG\_E183L\_XXX.mk: WIND\_APN\_CONF=apns-conf-E183L\_xxx.xml

3.验证方法：编译完成后，将版本down进手机，使用DDMS将data/data/com.android.providers.telephony里面的数据库telephony.db Push出来，查看里面有个carrier的表，这里面会存放当前手机中保存的所有APN的信息。手机第一次开机的时候会将anps-conf.xml里所有的apn都读到这个表里。若是添加成功即可以看到该表里面有新添加的项。

## (2) SPN

1.配置路径：

vendor/mediatek/proprietary/frameworks/base/telephony/etc/spn-conf-E183L\_xxx.xml

2.编译路径：wind/config/ CONFIG\_E183L\_XXX.mk:WIND\_SPN\_CONF=spn-conf-E183L.xml

3.虚拟运营商APN配置文件(MTK默认支持下面四种配置，配置前通过需求确认哪种)

(1)mediatek/frameworks/base/telephony/etc/ virtual-spn-conf-by-efspn-E183L.xml通过spn识别的虚拟运营商

(2)mediatek/frameworks/base/telephony/etc/virtual-spn-conf-by-efimsi-XXX.xml通过ims识别的虚拟运营商

(3)mediatek/frameworks/base/telephony/etc/virtual-spn-conf-by-efpnn-XXX.xml通过pnn识别的虚拟运营商

(4)mediatek/frameworks/base/telephony/etc/virtual-spn-conf-by-efgid1-XXX.xml通过gid1识别的虚拟运营商

## (3) Voicemail

1.配置路径：mediatek/frameworks/base/telephony/etc/voicemail-conf-E183L.xml

2.编译路径：wind/config/ CONFIG\_E183L\_XXX.mk：WIND\_VOICEMAIL\_CONF=voicemail-conf-E183L.xml

系统生成的时候会读取voicemail-conf.xml生成到对应的contact数据库中，索引是MCCMNC。

## (4) 关于虚拟运营商:

MVNO(Mobile Virtaul Network Operator)虚拟网络运营商，没有自己的实体网络，通过租用MNO(Mobile Network Operator)的网络来提供网络服务。

我们知道Spec规定operator之间是通过MCC/MNC （Mobile Country Code/Mobile Network Code）来区分的；而MVNO和对应MNO的MCC/MNC是相同的，那就需要MVNO定义额外的栏位（通常都是SIM卡中某支文件）来和对应MNO做区分；具体这个额外的栏位是什么是每个MVNO自己定义的，需要向MVNO确认。

目前MTK支持区分MVNO的方式有四种，每种区分方式对应一个xml的配置表：

1. EF\_SPN方式，对应MVNO配置到Virtual-spn-conf-by-efspn.xml中

2. EF\_IMSI方式，对应MVNO配置到Virtual-spn-conf-by-imsi.xml中

3. EF\_PNN方式，对应MVNO配置到Virtual-spn-conf-by-efpnn.xml中

4. EF\_GID1方式，对应MVNO配置到Virtual-spn-conf-by-efgid1.xml中

详情请参见：[FAQ09811] 如何区分MNO和MVNO

# 4、默认语言和输入法

## (1) 项目添加语言和默认语言

修改device/ginreen/E183L\_35M/full\_E183L\_xxx.mk文件中的PRODUCT\_LOCALES配置，添加所需

要的语言，将默认语言配置成第一个。

ifeq ($(strip $(WIND\_PROJECT\_NAME\_CUSTOM)),E183L\_35M\_TEL)

PRODUCT\_LOCALES := es\_USpt\_BRen\_US

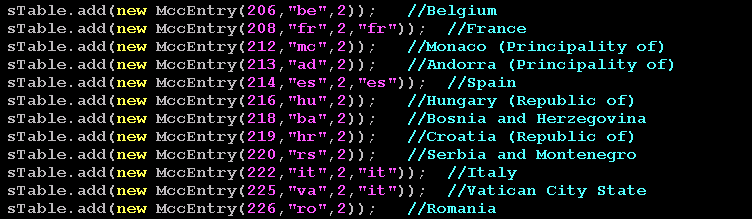
## (2) 根据SIM卡的MCC自动识别语言

Android 默认支持通过SIM识别默认国家（通过MCC）来设置默认语言。

SRC：framework/opt/telephony/src/java/com/android/internal/telephony/MccTable.java

MccTable.java中sTable数组的添加，MccEntry(MCC, ISO, smallestDigitsMCC [, Language] ).

如果遇到不能通过SIM卡来切换默认语言的问题，请确认对应的MCC列表中，是否有Language的设置。



## (3) 输入法

若要添加默认输入法宏控则按下列方法：

默认输入法(触宝为例)：

1. 在wind/config/CONFIG\_E183L\_xxx.mk添加默认输入法宏WIND\_IM\_TOUCHPAL = yes
2. 在wind/custom\_files/device/ginreen/E183L\_35M/device.mk添加：

ifeq ($(strip $(WIND\_IM\_TOUCHPAL)),yes)

PRODUCT\_PROPERTY\_OVERRIDES += ro.wind\_im\_touchpal=1

else

PRODUCT\_PROPERTY\_OVERRIDES += ro.wind\_im\_touchpal=0

endif

1. 在frameworks/base/packages/SettingsProvider/src/com/android/providers/settings/DatabaseHelper.java的中添加：

Public static final boolean WIND\_IM\_TOUCHPAL=SystemProperties.get(“ro.wind\_im\_touchpal”).equals(“1”);

loadSecureSettings()方法中添加：

if (WIND\_IM\_TOUCHPAL) {

loadSetting(stmt, Settings.Secure.DEFAULT\_INPUT\_METHOD,

"com.emoji.keyboard.touchpal/com.cootek.smartinput5.GateActivity");

}

# 5、桌面布局

主桌面布局，集成壁纸均需要在overlay中修改。

SourceFile: packages/apps/Launcher2/res/xml/default\_workspace.xml 【4.2，注意overlay】

SourceFile: packages/apps/Launcher3/res/xml/default\_workspace.xml 【4.4，注意overlay】

SourceFile: packages/apps/Launcher3/res/x ml/default\_workspace\_4X4.xml 【5.1，注意overlay】

壁纸资源放置在xx/resource\_overlay/xx/packages/apps/Launcher3/res/drawable-hdpi

修改xx/resource\_overlay/xx/packages/apps/Launcher3/res/values-nodpi/wallpapers.xml

默认壁纸放置在xx/resource\_overlay/xx/frameworks/base/core/res/res/drawable-nodpi目录，

命名为default\_wallpaper.jpg。【Android4.4默认壁纸不需要放到壁纸资源目录，也不需要在wallpapers.xml

中集成，但是Android4.2需要】

添加壁纸：

在Launcher3/WallpaperPicker/res/drawable-xxx的文件夹下增加wallpaer的图片，每个wallpaper需要

两种图片一张原图一张缩略图，如下形式

wallpaper\_01.jpg

wallpaper\_01\_small.jpg

wallpaper\_02.jpg

wallpaper\_02\_small.jpg

缩略图的文件名必须原图"文件名"+"\_small"

在Launcher3/WallpaperPicker/res/values-nodpi的wallpapers.xml 中修改如下：

<resources>

<string-array name="wallpapers"

translatable="false">

<item>wallpaper\_01</item>

<item>wallpaper\_02</item>

</string-array>

< /resources>

注意：以上的修改都在overlay 中进行！

# 6、Browser/Email

## (1) Browser

1.修改主页和书签

Sourcefile:device/ginreen/E183L/overlay/generic/vendor/mediatek/proprietary/packages/apps/Browser/res/values/strings.xml(主页)

Sourcefile:device/ginreen/E183L/overlay/generic/vendor/mediatek/proprietary/packages/apps/Browser/res/values/mtk\_strings.xml(书签，此文件需要观察homepage\_base\_site\_navigation是否符合要求以作修改)

2.搜索引擎

Sourcefile：device/ginreen/E183L\_35M/overlay/TEL/vendor/mediatek/proprietary/frameworks/base/res/res/values/donottranslate-new-search\_engines.xml(可根据不同的语言进入不同的values目录下配置不同的搜索引擎)

譬如在中文中的搜素引擎默认配置成谷歌：

<resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">

<string-array name="new\_search\_engines" translatable="false">

<!--modify search engines-->

<item>--</item><item>google--Google--google.com--search\_engine\_google--http://www.google.com/m?hl={language}&amp;ie={inputEncoding}&amp;source=android-browser&amp;q={searchTerms}--UTF-8--http://www.google.com/complete/search?hl={language}&amp;json=true&amp;q={searchTerms}</item>

</string-array>

</resources>

在<item>--</item>后添加的原因是，searchEngineManager会拿第一个作为默认搜索引擎。

frameworks/base/services/core/java/com/mediatek/search/SearchEngineManagerService.java中：

if (mDefaultSearchEngine == null) {

mDefaultSearchEngine = mSearchEngineInfos.get(0);

}

## (2) Email

(1) 默认自动数据添加

修改邮箱的APN（即设定自动登陆邮箱的服务器信息）

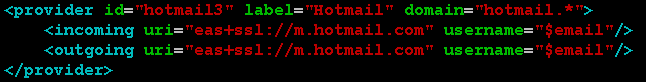
Sourcefile：device/ginreen/E183L/overlay/LIFE/packages/apps/Email/res/xml/providers.xml注意，此处修改后，也无法显示在手动配置界面，google默认没有此显示功能。

配置信息时，注意端口号和协议的编写，如果是不包含在providers.xml中的域，则需要再拷贝增加的信息到product\_provider.xml中，否则信息无法追加。

(2) 手动选择添加账户

一般来说，手动配置账户是需要用户更改的，系统默认会根据选择的账户类型填充

Sourcefile：device/ginreen/E183L/overlay/LIFE/packages/apps/Email/res/xml/manual\_setup\_helper.xml



# 7、时区

## (1) 默认时区配置

若客户定制，需明确不插卡时默认时区，配置在CONFIG\_E183L\_XXX.mk

WIND\_PORDUCT\_TIMEZONE=Europe/Moscow

## (2)NITZ默认打开设置

Sourcefile:Frameworks/base/packages/SettingsProvider/res/values修改defaults.xml

<bool name="def\_auto\_time">true</bool>

<bool name="def\_auto\_time\_zone">true</bool>

以上两项设置为true即可。

## (3)默认日期格式

日期格式若无定制，则不需修改，保持默认即可。若有定制，一般修改

wind/config/CONFIG\_E183L\_XXX.mk

WIND\_PORDUCT\_DATEFORMAT=yyyy-MM-dd

## (4) 24小时配置

24 小时制：WIND\_DATEFORMAT\_IS\_24= yes

12 小时制：WIND\_DATEFORMAT\_IS\_12= yes

# 8、GMS

1. GMS包里的信息添加
2. 项目MakeFile中配置编译宏
3. 如何检查

一般集成哪些GMS应用不会出错，主要问题出现在GMS需求表中一些属性的配置。

Client ID一般会有两种类型

|  |  |
| --- | --- |
| Client ID | Value |
| ro.com.google.clientidbase | android-zte |
| ro.com.google.clientidbase.ms | android-americamovil-{country} |
| ro.com.google.clientidbase.am | android-americamovil-{country} |
| ro.com.google.clientidbase.gmm | android-zte |
| ro.com.google.clientidbase.yt | android-zte |

|  |  |
| --- | --- |
| Client ID | Value |
| ro.com.google.clientidbase | android-zte |
| ro.com.google.clientidbase.ms | andorid-zte |
| ro.com.google.clientidbase.am | andorid-zte |
| ro.com.google.clientidbase.gmm | andorid-zte |
| ro.com.google.clientidbase.yt | andorid-zte |

代码中宏控已经加好，Client\_ID默认配置为下面表格中的值，如果为上面一种，则需要在对应项目的ProjectConfigXX中添加GMS\_CLIENTID\_ZTE\_DEFAULT=no

需要注意ro.product.xxx(model、brand、board)值，编译之后查看一下out目录下system/build.prop，检查属性值是否正确，有差异，一般修改build/tools/buildinfo.sh文件。

# 9、内置APK的添加

## 预装无源码APK

1.可卸载且恢复出厂设置可恢复(以预置一个名为Test的APK为例)

（1）将Test.apk放在vendor\mediatek\proprietary\packages\3rd-party\E183L\_XXX\XXX\removable下

（2）在vendor\mediatek\proprietary\packages\3rd-party\ E183L\_XXX\XXX\removable下面创建文件Android.mk文件内容如下（红色字体为备注）:

LOCAL\_PATH := $(call my-dir) 设置当前模块的编译路径为当前文件夹路径。即当前Android.mk 所在的目录

include $(CLEAR\_VARS) 清除变量

# Module name should match apk name to be installed

LOCAL\_MODULE := Test 要和apk的名字一样

LOCAL\_MODULE\_TAGS := optional 指该模块在所有版本下都编译

LOCAL\_SRC\_FILES := $(LOCAL\_MODULE).apk指定源apk

LOCAL\_MODULE\_CLASS := APPS 指定文件类型是apk文件，并会检查是否是apk文件

LOCAL\_MODULE\_SUFFIX := $(COMMON\_ANDROID\_PACKAGE\_SUFFIX) module的后缀.=.apk

LOCAL\_CERTIFICATE := PRESIGNED 表示这个apk已经签过名了，不需要再签名

LOCAL\_MODULE\_PATH := $(TARGET\_OUT)/vendor/operator/app指定最后的目标安装路径

include $(BUILD\_PREBUILT)

(3)在device/ginreen/E183L\_XXX/device.mk中添加:

ifeq ($(strip $(WIND\_DEF\_OPTR\_E183L\_TEL)),yes)

PRODUCT\_PACKAGES += \

ClaroApp \

ClaroJuegos \

MiTelcel \

Test\

endif

2.可卸载且恢复出厂设置不可恢复

（1）将Test.apk放在vendor\mediatek\proprietary\packages\3rd-party\ E183L\_XXX\XXX \removable下

（2）在vendor\mediatek\proprietary\packages\3rd-party\ E183L\_XXX\XXX \removable下面创建文件Android.mk文件内容如下:

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

$(CLEAR\_VARS)

# Module name should match apk name to be installed

LOCAL\_MODULE := Test

LOCAL\_MODULE\_TAGS := optional LOCAL\_SRC\_FILES :=

$(LOCAL\_MODULE).apk

LOCAL\_MODULE\_CLASS := APPS

LOCAL\_MODULE\_SUFFIX := $(COMMON\_ANDROID\_PACKAGE\_SUFFIX)

LOCAL\_CERTIFICATE := PRESIGNED

LOCAL\_MODULE\_PATH := $(TARGET\_OUT\_DATA\_APPS) include

$(BUILD\_PREBUILT)

(3)在device/ginreen/E183L\_XXX/device.mk中添加:

ifeq ($(strip $(WIND\_DEF\_OPTR\_E183L\_TEL)),yes)

PRODUCT\_PACKAGES += \

ClaroApp \

ClaroJuegos \

MiTelcel \

Test\

endif

3．不可卸载

（1）将Test.apk放在vendor\mediatek\proprietary\packages\3rd-party\ E183L\_XXX\XXX \unremovable下

（2）在vendor\mediatek\proprietary\packages\3rd-party\ E183L\_XXX\XXX \unremovable下面创建文件Android.mk文件内容如下:

LOCAL\_PATH := $(call my-dir) include $(CLEAR\_VARS)

# Module name should match apk name to be installed LOCAL\_MODULE := Test

LOCAL\_MODULE\_TAGS := optional

LOCAL\_SRC\_FILES := $(LOCAL\_MODULE).apk

LOCAL\_MODULE\_CLASS := APPS

LOCAL\_MODULE\_SUFFIX := $(COMMON\_ANDROID\_PACKAGE\_SUFFIX)

LOCAL\_CERTIFICATE := PRESIGNED

LOCAL\_PRIVILEGED\_MODULE := true有这一句表示是安装在system/priv-app下面；如果没有，表示是安装在system/app下面

#LOCAL\_PREBUILT\_JNI\_LIBS:= \

#@lib/armeabi/libtest.so \

#@lib/armeabi/libtest2.so

include $(BUILD\_PREBUILT)

(3)在device/ginreen/E183L\_XXX/device.mk中添加:

ifeq ($(strip $(WIND\_DEF\_OPTR\_E183L\_TEL)),yes)

PRODUCT\_PACKAGES += \

ClaroApp \

ClaroJuegos \

MiTelcel \

Test\

endif

注：

解压Test.apk文件，并且查看。

若无so，注释或删除LOCAL\_PREBUILT\_JNI\_LIBS及其相关部分

若有so，使用LOCAL\_PREBUILT\_JNI\_LIBS列出所有so的路径，不要忘记使用@。@标识符会将apk中的so抽离出来build进system/lib或者system/lib64中；

若apk支持不同cpu类型的so，针对so的部分的处理:

ifeq ($(TARGET\_ARCH),arm)

LOCAL\_PREBUILT\_JNI\_LIBS := \

@lib/armeabi-v7a/xxx.so\

@ lib/armeabi-v7a/xxxx.so

elseifeq ($(TARGET\_ARCH),x86)

LOCAL\_PREBUILT\_JNI\_LIBS := \

@lib/x86/xxx.so

elseifeq ($(TARGET\_ARCH),arm64)

LOCAL\_PREBUILT\_JNI\_LIBS := \

@lib/armeabi-v8a/xxx.so

endif

即将和TARGET\_ARCH对应的so抽离出来在device/ginreen/E183L\_XXX/device.mk中添加：PRODUCT\_PACKAGES += Test

## 预装有源码APK

在packages/apps下面以需要预置的APK的名字创建一个新文件夹，以预置一个名为Test

的APK为例

（1）将Test APK的Source code拷贝到Test文件夹下，删除/bin和/gen目录

（2）在Test目录下创建一个名为Android.mk的文件，内容如下：

LOCAL\_PATH:= $(call my-dir) include $(CLEAR\_VARS)

LOCAL\_MODULE\_TAGS := optional

LOCAL\_SRC\_FILES := $(call all-subdir-java-files)

LOCAL\_PACKAGE\_NAME := Test include $(BUILD\_PACKAGE)

在device/ginreen/E169F/device.mk 中添加：PRODUCT\_PACKAGES += Test

预装注意说明

若需要apk作为32bit的apk运行，则需要在Android.mk中定义

LOCAL\_MULTILIB :=32

因为L版本默认的LOCAL\_MULTILIB :=64

但是为了保险起见，也可以设置LOCAL\_MULTILIB :=both

另外，关于LOCAL\_CERTIFICATE：

1.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。

2. 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。

3.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

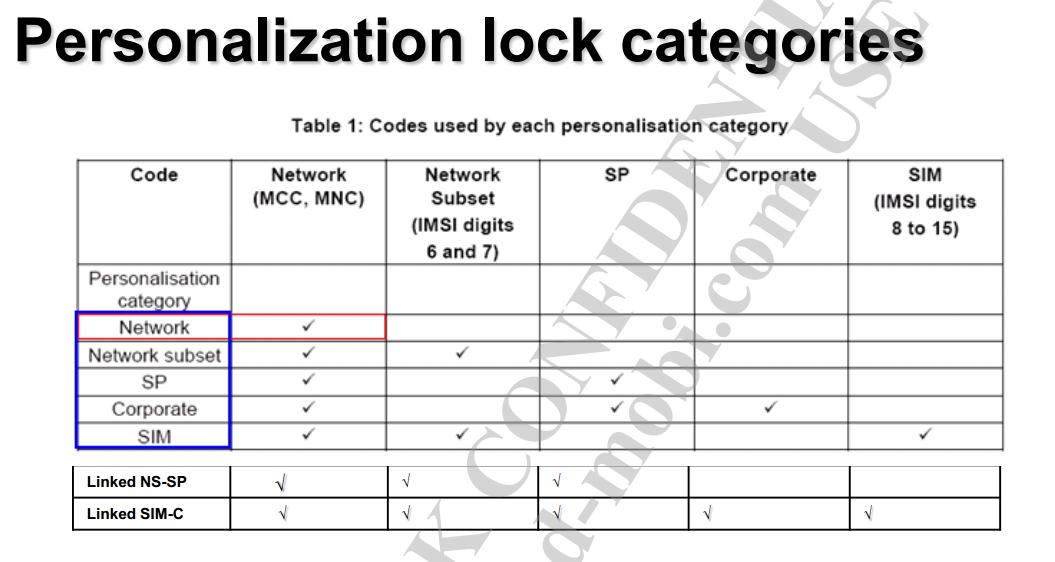
android:sharedUserId="android.media"，然后在Android.mk中增加LOCAL\_CERTIFICATE:= media

2、FOTA

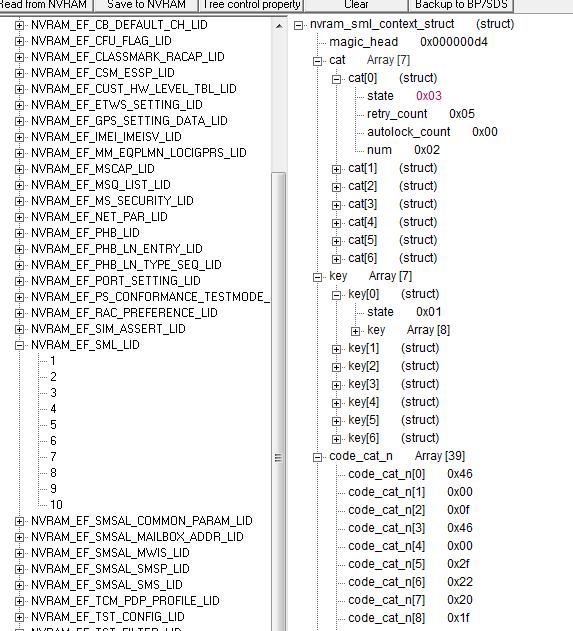
合入FOTA升级APK，同集成第三方APK方法

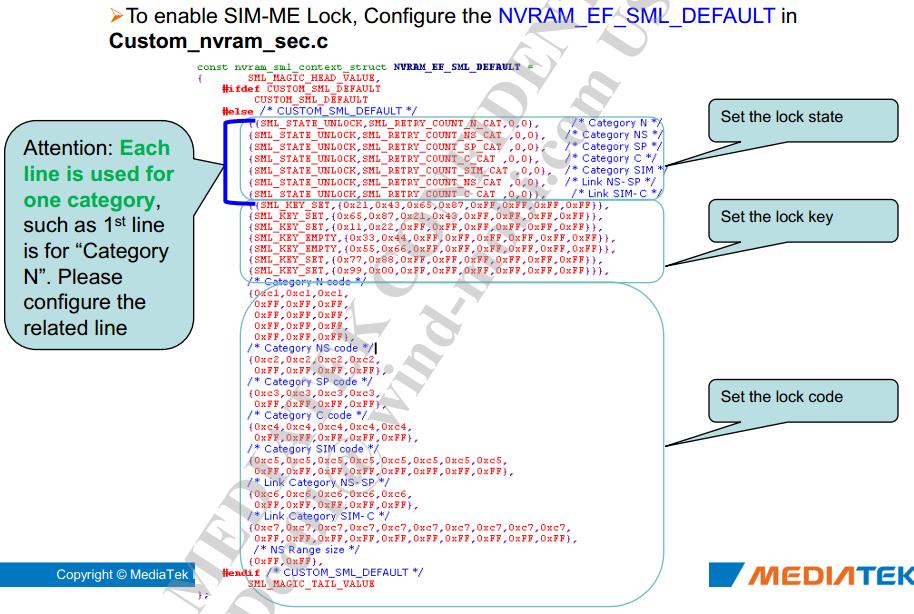
# 10、锁网

某些运营商出于利益保护，会禁止其他运营商的SIM卡通过其发布的终端入网，获取电信服务。MTK平台上常用的锁网类型如下图。前五项是3GPP协议（22.022）定义了上张表格中的5中锁网种类，最后两项是MTK自己扩展的。

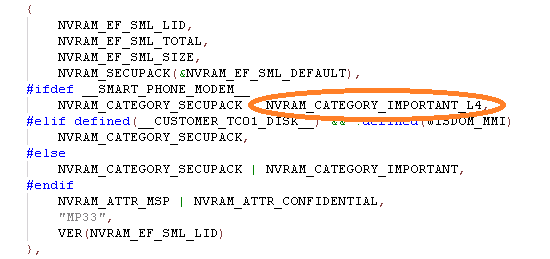


MTK平台对锁网的配置主要是Modem部分的NV的配置。如下图





SML锁网的NV参数配置如下：



主要要注意参数NVRAM\_CATEGORY\_IMPORTANT\_L4；若NVRAM具有NVRAM\_CATEGORY\_IMPORTANT属性或者有NVRAM\_CATEGORY\_IMPORTANT\_L4属性，会被自动添加NVRAM\_ATTR\_AUTO\_RECOVER属性。这样这些NVRAM在modem端读取失败之后，就会触发自BIN Region中的recovery。即，此类参数的NVRAM在下载的时候不会被清除掉，除非格式化。如果要去掉此属性，这样的修改之后，每次factory rese(恢复出厂设置)都会让锁卡数据恢复成默认值。

目前仅仅有IMEI, CALIBRATION, SIM-ME Lock相关的NVRAM会被back up到BIN Region中。

## (1) Modem内置锁网值

之前T651W所有的Android 4.2版本上的锁网参数都是直接配置在NVRAM\_EF\_SML\_DEFAULT数组里，直接编译Modem，配置对应的客制化项目。

这样配置Modem直接锁网有利也有弊。

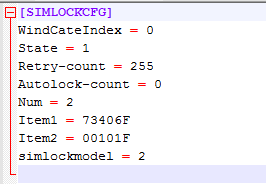
好处：版本单一，前方、工厂对版本状态比较明确，测试简单。

不便：不同版本之前升级必须格式化，如果后续调板升级必须格式化。对于兼容很多国家的渠道版本不能做到有的支持锁网，有的不锁网。

## (2) 工具修改Modem配置锁网

Android 4.4版本之后更新了锁网配置机制。改成了通过工具读取写入NVRAM\_EF\_SML\_DEFAULT字段。这样通过一个配置的锁网文件SIMLOCK.ini来锁不同的运营商。

如下是一个委内瑞拉的锁网文件



这样处理也有利也有弊

好处：渠道版本可以兼容锁网或者不锁网。AP端可以使用一套Modem兼容很多客制化。

弊端：锁网文件独立于版本，如果产线漏了锁网则无法补救。要使用此方法则需要放开META读取NVRAM\_EF\_SML\_DEFAULT的条件（默认不能查看读取），则有一定风险被破解。

详细资料参考

# 11、紧急拨号

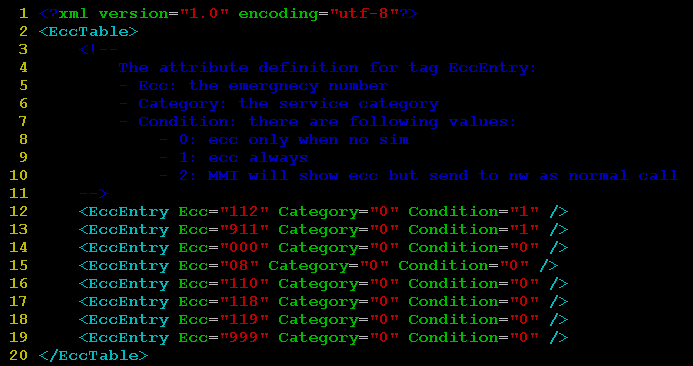
L及之后的版本紧急号码Customer的部分改成了在XML文件中来配置，文件的路径：

**E183/vendor/mediatek/proprietary/external/Ecclist**

EccList文件夹中会包含ecc\_list.xml，以及与运营商有关的ecc\_list\_OP01.xml、ecc\_list\_OPXX.xml等对应文件，此外还包括一个EccList.mk的Makefile。实际运行中会根据Makefile文件中的定义匹配对应的XML文件作为判断是否是紧急号码的来源。

在定制紧急拨号时，需在wind中对应的目录下新增一份ecc\_list\_xxx.xml文件；此外还需要修改EccList.mk，使代码编译时能够编入该ecc\_list\_xxx.xml。

下面是ecc\_list.xml文件中的内容：



**说明：**

一、添加号码请注意Condition的配置，根据需求来选择对应的值。

0:表示在无卡的时候当紧急号码；

1:表示始终当紧急号码；

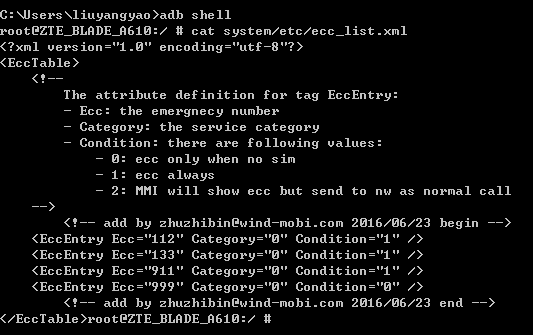
2:表示界面上显示成紧急拨号，但实际以普通方式拨出。

二、Category属性的设置于语音台选择有关，只有在实际拨打紧急号码的时候会将此号码配置的Category属性发送到Modem。国内默认都是‘0’，国外根据实际情况选择。

**验证：**

紧急拨号文件会放到手机中，其路径为：system/etc/ecc\_list.xml

因此可查看该文件是否成功编进手机，如下图：



# 12、网络漫游配置

## (1) 何为漫游

**1. 关于漫游**

A．Roaming：

Spec定义的roaming是指Operator A的sim卡注册到了Operator B的网络plmn上-------如果这部分需要特殊处理成非roaming，请参考：

ID: FAQ06637 [NW]网络漫游客制化 (手机显示漫游图标)

B. International Roaming：

Operator A的sim卡注册到了和Operator A不同国家的Operator的plmn上；

operator用MCC/MNC来区分；MCC: mobile country code；MNC：mobile network code

operator的plmn的构成也是MCC/MNC的结构

所以International Roaming<=>SIM卡所属operator的MCC ！=此卡当前所注册plmn的MCC

**2.如何知手机发生了international roaming**

1. AP NW FW的主体逻辑在GsmServiceStateTracker.java中；其中的key method是pollState，当网络发生变化时就会call此方法，对应处理方法是handlePollStateResult

2. 在handlePollStateResult中有 roaming这个变数如果漫游就会被置为true；

3. 在handlePollStateResult中的case EVENT\_POLL\_STATE\_OPERATOR:下opNames[2]记录的就是当前网络的plmn，其中前三位就是此plmn对应的MCC

4. 在SIMRecords.java中有方法getSIMOperatorNumeric可以获取卡的MCCMNC，然后其前三位就是MCC；在GsmServiceStateTracker.java中的updateSpnDisplay中有使用此方法的demo code，您可以参考

至此，您需要的info都已经有了，您只需要在handlePollStateResult方法中的roaming变数后面判断在roaming为true时当前网络的plmn对应MCC和卡的MCC如果不一样您就发出一个broadcast来通知或者做弹框之类的提示

## 如何区别和取消漫游

**1. 漫游判定的flow：**

在GsmServiceStateTracker.java中如果网络发生变化就会call pollState方法去获取最新的网络状况，response用handlePollStateResult处理，其中roaming的判定modem通过+CREG上报：



**2. 漫游客制化方法：**

对于JB\*, KK\*及以后的版本:

在regCodeIsRoaming中有留了客制化接口，只需在customEhplmn二位数组的尾部加入一组item即可；

比如韩国operator KCT 45011的卡注册了operator SKT 45005的网络显示漫游，要求作为home plmn，不显示漫游:

Private String[][] customEhplmn = {{“46000”,“46002”, “46007”},

{“45400”,“45402”,“45418”},

…,

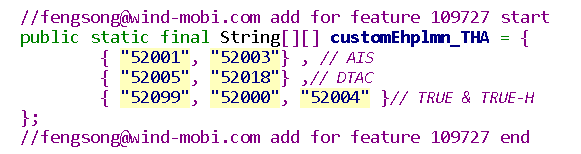
{“310150”,“310170”,“310380”,“310410”},

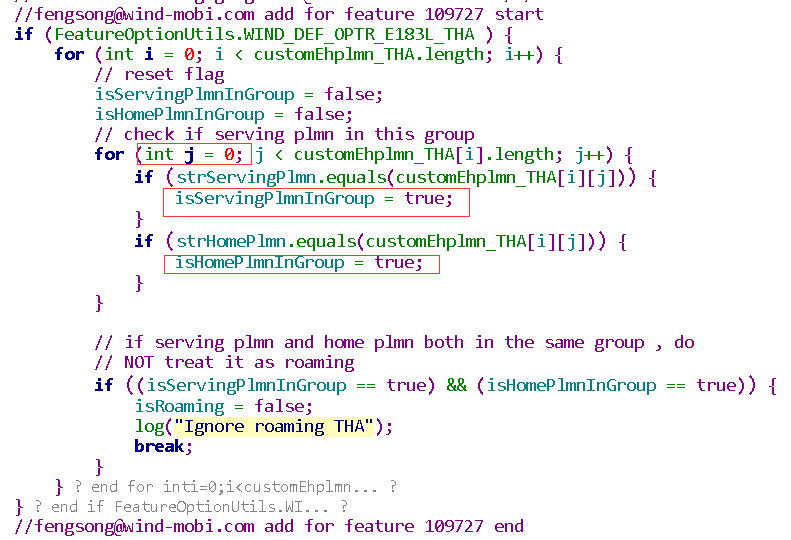
{“45011”,“45005”}}

例如E183L



**customEhplmn中定义的是所有项目都支持的漫游，如果是某个项目的定制漫游，要定义一个新的数组，并且要修改代码。**例如：





**还有一点需要注意，定义的漫游消除的数组中，如果是互相消除漫游，则二维数组　j　从０开始循环，**

**例如上面定义的：**

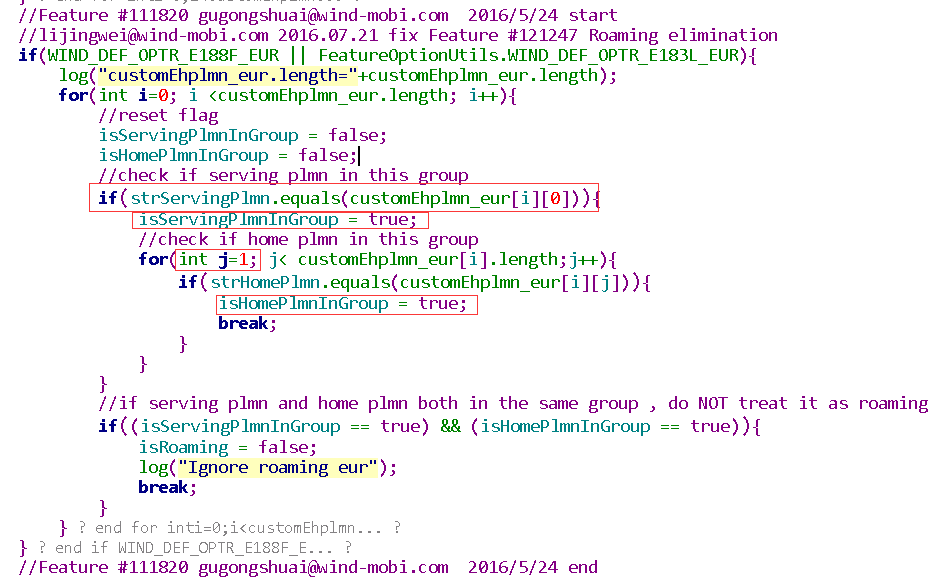
**即 52005注册到52018上消除漫游，52018 注册到52005也消除漫游。**

**还有种情况，单向的漫游消除，后面注册到前面的消除漫游，则二维数组　j　从１开始循环**

**例如：**







# 内置联系人

## (1)如何内置联系人

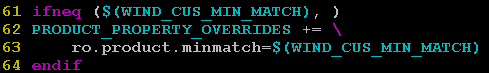
参考**[FAQ07470] 在手机中预置联系人/Service Number**

## (2)号码匹配

1. 在/wind/config/CONFIG\_E183L\_xxx.mk文件中修改WIND\_CUS\_MIN\_MATCH宏的值

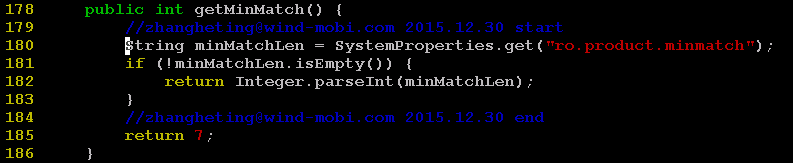


2. 查看/wind/custom\_files/build/target/product/core\_base.mk中该宏的ro值开关是否存在，不存在则添加



3. 修改PhoneNumberExt.java (/vendor/mediate/proprietary/frameworks/base/packages/

FwkPlugin/src/com/mediatek/op/telephony/目录下)



# 14、短信相关

## (1) 短信中心号码

文件 SmsPreferenceActivity.java

OnPreferenceTreeClick函数，变量gotScNumber进行赋值，

mNumberText.setInputType(EditorInfo.TYPE\_NULL);可以设置短信中心号码的只读

常见问题：短信可以接受，但是无法发送。基本就是短信中心号码错误。

## (2) 内置小区广播号码

具体要参考FAQ06820、FAQ17750，增加database操作函数，增加chanle函数，以及UI函数

设置不同频道不同声音：

CBMessageNotification.java

getCBNewMessageNotificationInfoBase()方法获取相关频道

updateNotification()方法进行不同频道不同铃声

CMAS接受不到可以参考[FAQ17750]进行相关设置

定制紧急小区广播铃声：

1. CMAS默认铃声用的是如下文件，

/vendor/mediatek/proprietary/packages/apps/CMASReceiver/res/raw/attention\_signal.ogg

1. 如果要修改，可以在raw把自己的铃声文件加进去，然后在

/vendor/mediatek/proprietary/packages/apps/CMASReceiver/src/com/mediatek/cellbroadcastreceiver/AlertTonePreviewDialog.java

/vendor/mediatek/proprietary/packages/apps/CMASReceiver/src/com/mediatek/cellbroadcastreceiver/CellBroadcastAlertAudio.java

把使用到R.raw.attention\_signal的地方改成指定为自己的铃声文件

# 15、STK名称

有的项目会要求STK名称从SIM卡中读取，即：STK名称从SET\_UP\_MENU的proactive command里面解析出来。以 Feature #103571 为例，进行讲解。

描述：

STK名称从SET\_UP\_MENU的proactive command里面解析出来，Alpha identifier就是从SIM卡读取的STK名称；不插卡时STK名称为“SIM Telcel”

定义了两个功能宏控：

WIND\_DEFAULT\_STK\_NAME=SIM Claro # 如果没从SIM卡中获取到名称，则默认显示此名称

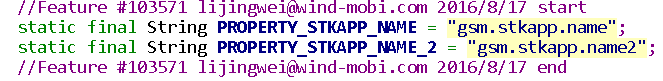
WIND\_SHOW\_STK\_NAME\_FROM\_SIM=yes # STK名称从SIM卡中读取

**目前最新代码已经实现双卡的控制，即卡1和卡2的STK名称都可以从SIM卡中获取**

实现 Feature #103571 的大致思路流程如下：

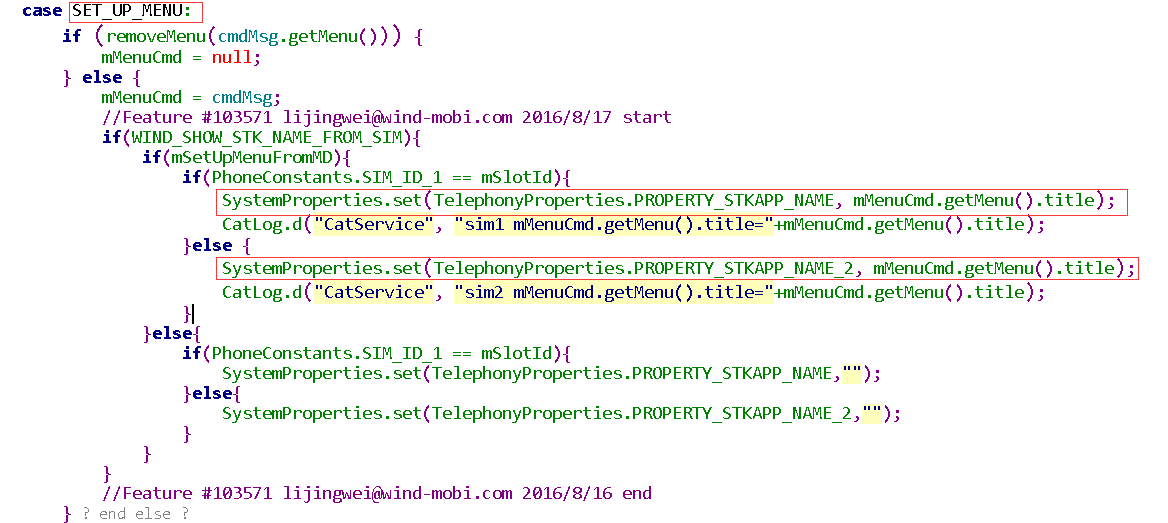
1. 在TelephonyProperties.java中为两个卡定义存放title的string





1. 当SETUP MENU上报时保存title：CatService.java中做如下修改：





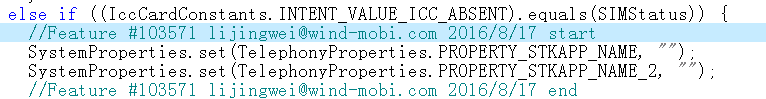
1. 动态修改Launch界面STK的名称





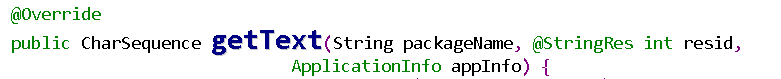
1. 当拔掉SIM 卡时，记得清除从SIM中获取的title名称

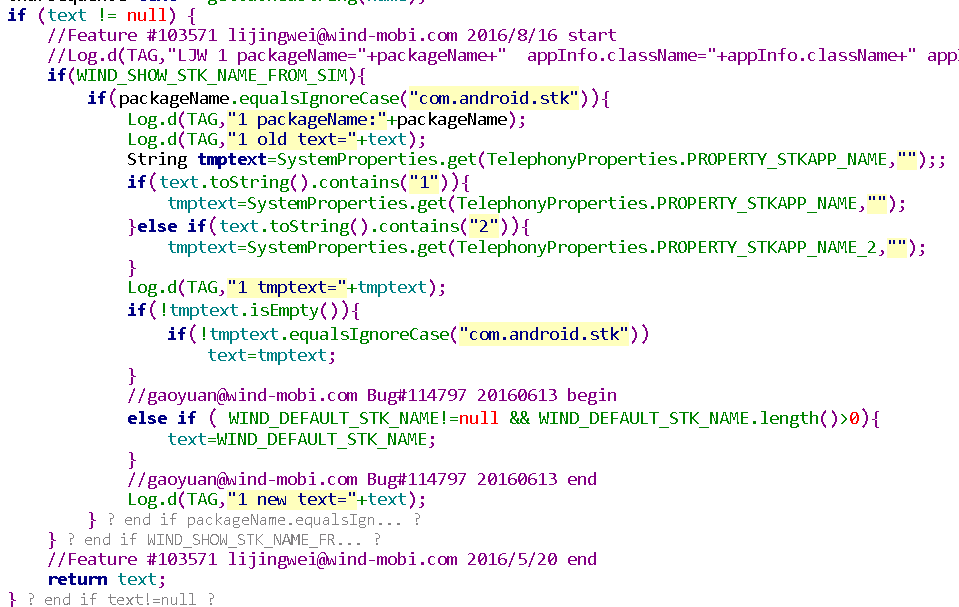


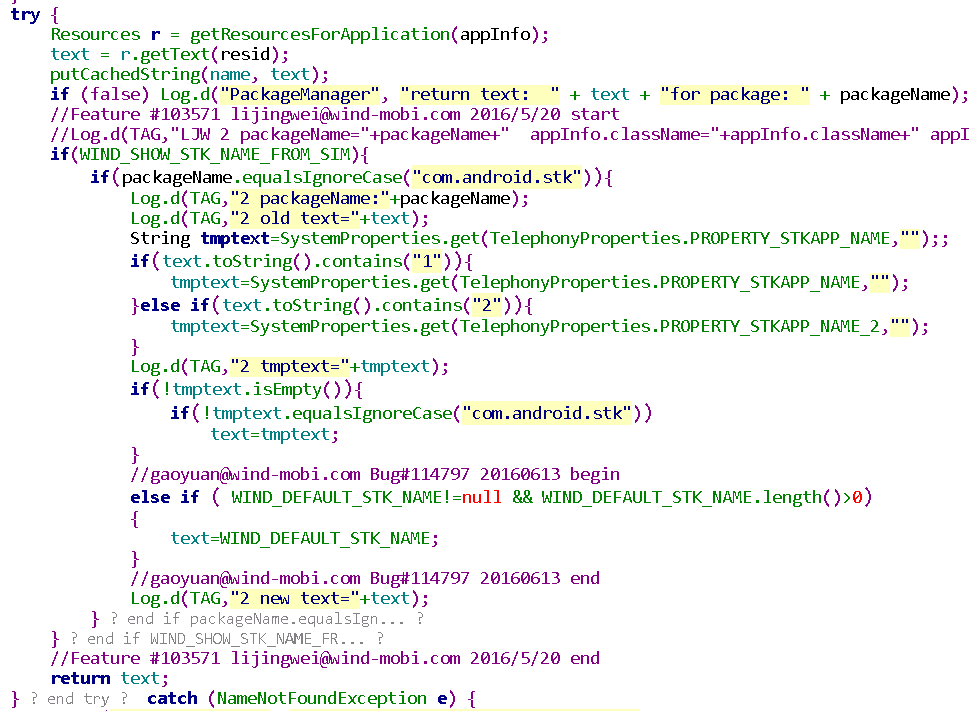


1. 把STK应用拖到桌面上，重启手机STK名称没有从SIM卡中读取









备注：以上步骤是实现STK名称从SIM卡中读取的基本流程，如果客户还要其他特殊的需求，请在此基础上自行客制化。

# 16、FOTA

**(1) FOTA 检查版本**

ZTE使用的FOTA apk是有中兴业软提供的，如果需要修改请让需求协助。

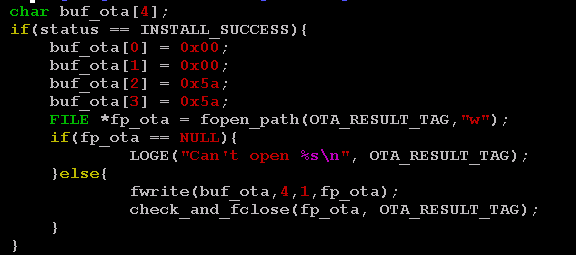
注意事项：

FOTA 最新的APK增加了Model校验功能，即：每个FOTA apk都内置限定了Model。如果检测到当前手机的Model和APK内置的Model不一致话，会提示无法升级。请联系需求提供对应的apk.

**(2) Recovry修改**

***A)、*FOTA升级成功后，提示升级失败。**

FOTA升级状态成功与否，是从QQ截图20140801164055.png文件中的标志位读取出来的，所以在recovery.cpp文件中要加入标志位的写入：



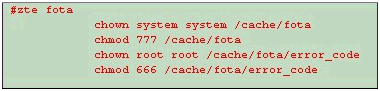
***B)、*T卡升级后，再进行FOTA升级，会提示升级失败的问题**

问题的大概原因：

如第一次先进行SD卡升级，在进入recovery模式后，SD卡升级完成后如果不存在error\_code，则会创建该文件，创建的文件是有root账户创建的，且采用的默认方式创建，所以导致该文件只能由root权限的用户才能访问。则会导致system/shell用户无法读取。之后进行fota手机，由于fota应用的用户权限是system，无法访问error\_code，就会在升级成功后提示升级失败。

修改方法：

由recovery在创建error\_code的时候强制修改该文件权限为666或者777，或者在系统初始化的时候修改：



**(3) 服务器配置**

ZTE海外项目，都放在dms.ztems.com服务器上。   
[http://dms.ztems.com:81/portal/uiloader/index.html](http://dmcn.ztems.com:81/portal/uiloader/index.html)

和FOTA相对应，服务器上的项目名称也是靠Model名称来识别的。如果修改了Model名称需要项目经理重新在服务器上配置对应的项目。

制作UPC时，源版本号和目标版本号都由外部版本号\_内部版本号组成，比如CLA\_CL\_BLADE\_A610V1.0.0\_CLA\_CL\_QB25S\_P635T36V1.0.0B02，前面的CLA\_CL\_BLADE\_A610V1.0.0是外部版本号，后面的CLA\_CL\_QB25S\_P635T36V1.0.0B02是内部版本号。服务器上的源版本号和目标版本号需要与制作UPC时填写的内容一致。

具体如何制作以及配置UPC，参考wiki:

[GD05 SW01 B 014 中兴Fota包制作流程](http://10.0.10.6/windwiki/index.php/GD05_SW01_B_014_%E4%B8%AD%E5%85%B4Fota%E5%8C%85%E5%88%B6%E4%BD%9C%E6%B5%81%E7%A8%8B)

[GD05 SW01 B 015 中兴Fota包上传流程及测试IMEI添加](http://10.0.10.6/windwiki/index.php/GD05_SW01_B_015_%E4%B8%AD%E5%85%B4Fota%E5%8C%85%E4%B8%8A%E4%BC%A0%E6%B5%81%E7%A8%8B%E5%8F%8A%E6%B5%8B%E8%AF%95IMEI%E6%B7%BB%E5%8A%A0)

# 17、文件管理和视频播放

如何屏蔽指定的文件格式

修改文件：



只要能保证不运行addFileType就可以屏蔽掉次文件格式。

例如把MTK\_WMV\_PLAYBACK\_SUPPORT宏设置为false，那么WMA格式就会被屏蔽。

该宏在

