[FAQ13232]L 预置apk

[DESCRIPTION]

1, 如何将带源码的 APK 预置进系统?
2, 如何将无源码的APK预置进系统?
3, 如何预置APK使得用户可以卸载，恢复出厂设置时不能恢复?
4, 如何预置APK使得用户可以卸载，并且恢复出厂设置时能够恢复?

[SOLUTION]

一、如何将带源码的APK预置进系统?

1）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
2）将 Test APK的Source code 拷贝到 Test 文件夹下，删除 /bin 和 /gen 目录
3）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)
```

4）打开文件 device\mediatek\common\device.mk
将 Test 添加到 PRODUCT_PACKAGES 里面。

```
PRODUCT_PACKAGES += Test
```

5）重新 build 整个工程

二、如何将无源码的 APK 预置进系统?

1）在 packages/apps 下面以需要预置的 APK 名字创建文件夹，以预置一个名为Test的APK为例
2）将 Test.apk 放到 packages/apps/Test 下面
3）在 packages/apps/Test 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_PREBUILT_JNI_LIBS:= \
@lib/armeabi/libtest.so \
@lib/armeabi/libtest2.so
LOCAL_CERTIFICATE := PRESIGNED
include $(BUILD_PREBUILT)
```

若无so，删除LOCAL_PREBUILT_JNI_LIBS
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apk同级目录下的lib文件夹中
若apk支持不同cpu类型的so，针对so的部分的处理：

```
Ifeq ($(TARGET_ARCH),arm)
LOCAL_PREBUILT_JNI_LIBS := \
@lib/armeabi-v7a/xxx.so\
@ lib/armeabi-v7a/xxxx.so
else ifeq ($(TARGET_ARCH),x86)
LOCAL_PREBUILT_JNI_LIBS := \
@lib/x86/xxx.so
else ifeq ($(TARGET_ARCH),arm64)
LOCAL_PREBUILT_JNI_LIBS := \
@lib/armeabi-v8a/xxx.so
…
```

即将和TARGET_ARCH对应的so抽离出来

4）打开文件 device\mediatek\common\device.mk
将 Test 添加到 PRODUCT_PACKAGES 里面。

```
PRODUCT_PACKAGES += Test
```

5）重新 build 整个工程
注：如果App使用System Level的permission，需要預置到/system/priv-app底下（原在/system/app)。
修改Android.mk，增加LOCAL_PRIVILEGED_MODULE := true，以声明app需要放在/system/priv-app下。

三、如何预置APK使得用户可以卸载，恢复出厂设置时不能恢复？

1）在 packages/apps 下面以需要预置的 APK 名字创建文件夹，以预置一个名为Test的APK为例
2）将 Test.apk 放到 packages/apps/Test 下面
3）在 packages/apps/Test 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_PRIVILEGED_MODULE := true
LOCAL_MODULE_PATH := $(TARGET_OUT_DATA_APPS)
LOCAL_CERTIFICATE := PRESIGNED
include $(BUILD_PREBUILT)
```

4）打开文件 device\mediatek\common\device.mk
将 Test 添加到 PRODUCT_PACKAGES 里面。

```
PRODUCT_PACKAGES += Test
```

5）重新 build 整个工程
注意：这个比不能卸载的多了一句

```
LOCAL_MODULE_PATH := $(TARGET_OUT_DATA_APPS)
```

四、如何预置APK使得用户可以卸载，并且恢复出厂设置时能够恢复？

1在 vendor\mediatek\proprietary\binary\3rd-party\free下面以需要预置的 APK 名字创建文件夹，以预置一个名为Test的APK为例
2 将Test.apk 放入vendor\mediatek\proprietary\binary\3rd-party\free\Test下面

3 在vendor\mediatek\proprietary\binary\3rd-party\free\Test 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)/vendor/operator/app
include $(BUILD_PREBUILT)
```

2 打开文件device\mediatek\common\device.mk

将 Test 添加到 PRODUCT_PACKAGES 里面。

```
PRODUCT_PACKAGES += Test
```

3 然后重新build整个工程

请注意：

若需要apk作为32bit的apk运行，则需要在Android.mk中定义

```
LOCAL_MULTILIB :=32
```