Android framework analysis (partI zt)

Android Framework 分析 （一）

1.目录树

/framework/base/api

/framework/base/awt

/framework/base/build

/framework/base/camera

关于 camera 的 HAL 接口库。最终生成 native 共享库 libcamera.so ,编译时根据是否定义 USE_CAMERA_STUB 来决定系统是否有 Camera 硬件支持。若没有实际的 Camera 硬件，则编译时会和虚拟 camera 静态库（libcamerastub.a,由 camerahardwarestub.cpp,fakecamera 生成）链接生成 libcamera.so。

/framework/base/cmds 关于 android 系统启动时用到的 command 等

/framework/base/cmds/am

/framework/base/cmds/app_process

可执行文件 app_process，该文件可以根据输入参数决定是 Zygote 启动（参考 init.rc 中的语句 service zygote /system/bin/app_process -Xzygote /system/bin --zygote --start-system-server）.

该执行程式会链接 libandroid_runtime.so 去链接 android runtime。后面我会在详细分析此部分。

/framework/base/cmds/backup

可执行程式 btool

/framework/base/cmds/bmgr

java 可执行程式， backup manager, java 库形式分发到目标系统/system/framework/bmgr.jar

/framework/base/cmds/bootanimation

android 启动动画效果程式，该程式必须在 android runtime 启动后运行。

/framework/base/cmds/dumpstate

android 系统调试辅助工具,生成可执行程式 dumpstate,同时建立两个程式 dumpcrash bugreport 指向该程式。

/framework/base/cmds/dumpsys

生成可执行程式 dumpsys

/framework/base/cmds/ime

java 可执行程式 ，IME 输入法 input method manager, java 库形式分发到目标系统 /system/framework/ime.jar

/framework/base/cmds/input

java 可执行程式，管理 input 事件例如 key event,text event 等，java 库形式分发到目标系统 /system/framework/input.jar

/framework/base/cmds/installd

可执行程式 installd,install manager,仅在非 simulator 系统中运行，安装到目标系统 /system/bin/installd

/framework/base/cmds/keystore

可执行程式 keystore,用途？？？仅在非 simulator 系统中运行，安装到目标系统/system/bin/keystore

/framework/base/cmds/pm

java 可执行程式，package manager，java 库形式分发到目标系统/system/framework/pm.jar

/framework/base/cmds/runtime

runtime 可执行程式，仅在 simulator 中使用

/framework/base/cmds/service

service 可执行程式，用来查找，检查，呼叫 service，安装到目标系统/system/bin/service

/framework/base/cmds/servicemanager

android 系统的 service manager，可执行文件，安装到目标系统/system/bin/servicemanager

servicemanager 会和 kernel 的 binder driver 协作共同完成 service 的添加、查询、获取、检查等。

/framework/base/cmds/surfaceflinger

surfaceflinger 可执行程式，安装到目标系统/system/bin/surfaceflinger,

该程式会初始化 surfaceflinger,surfaceflinger::inistantiate(),该程式会链接到 libsurfaceflinger.so

/framework/base/cmds/svc

/framework/base/cmds/system_server

system server 库 libsystem_server.so->system/lib/libsystem_server.so 和 system_server 可执行程式->system/bin/system_server.

该可执行程式不清楚什么用途????

/framework/core/

/framework/core/config

几个简单 java 常量，（debug 标志等）

/framework/core/java/*

framewor

k 的核心，此处主要指 application framework,java 库形式分发到/system/framework/

包括 framework.jar,framework-tests.jar sure？？？

/framework/core/jni

framework 所需的 JNI 接口实现库,分发到/system/lib/lib/libandroid_runtime.so

/framework/core/res

framework 所需的资源文件打包，/system/framework/framework-res.apk,

/framework/libs

/framework/libs/audioflinger,

生成 libaudioflinger.so,

若无实际硬件和静态库 libaudiointerface.a（audio interface 虚拟设备）链接。

若有实际硬件和 libaudio.so 链接，若支持 bluetooth，则和 liba2dp.so 链接

/framework/libs/surfaceflinger

生成 libsurfaceflinger.so

/framework/libs/ui

生成 libui.so

/framework/libs/utils

生成 libutils.so

/framework/services/java/*

system server java 可执行程式 service.jar,分发到/system/framework/service.jar

/framework/services/jni/*

system server JNI 接口实现库,libanroid_servers.so，分发到/system/lib/libanroid_servers.so

android framework 分析（二）

启动 Zygote

-Xzygote /system/bin --zygote --start-system-server

AndroidRuntime->AppRuntime

int main(int argc,const char* const argv[])

{

AppRuntime runtime;生成 AndroidRuntime 实例

...

AndroidRuntime.Start("com.android.internal.os.ZygoteInit",startSystemServer);

}

其中 AndroidRuntime.Start("com.android.internal.os.ZygoteInit",startSystemServer);

呼叫 Android::Start(const char* className,const bool startSystemServer)

/framework/base/core/jni/AndroidRuntime.cpp

该函数的处理内容：

1.处理 Jave Virtual Machine 的一些参数选项；

2.创建 Dalvik Java 虚拟机，JNI_CreateJavaVM(&mJavaVM,&env,&initArgs)；

3.注册 Android Runtime 中的 JNI 接口给虚拟机；

4.呼叫 Java 类 com.android.internal.os.ZygoteInit 的 main 函数

在 类 com.android.internal.os.ZygoteInit 的 main 函数中，

1.注册 Zygote socket 用来接收请求;

2.加载 preloaded class、resources 用来加快启动速度，文件清单在 framework.jar 中的 preloaded-classes,framework-res.apk 中的 res 中;

3.启动 System Server;

fork 出独立的进程名称为 system-server，呼叫 com.android.server.SystemServer 类的 main 函数；

在 HandleSystemServerProcess 函数中，RuntimeInit.ZygoteInit 调用会呼叫 AppRuntime 的 OnZygoteInit 函数

4.RuntimeInit.ZygoteInit 函数会呼叫 com.android.server.SystemServer 类的 main 函数。

在此 main 函数中，系统首先加载 android_server 共享库 libandroid_server.so 源代码位于 /framework/base/service/jni

在该库中有定义 JNI_OnLoad 函数，所以 Dalvik 在加载 libandroid_server.so 的时候会首先呼叫该 JNI_OnLoad 函数，该函数将 android server 注册到 Java 虚拟机中，包括 KeyInputQueue,HardwareService,AlarmManager,BatteryService,SensorService,SystemServer 等；

呼叫在 libanroid_server.so 中注册的 native 函数 init1，该函数位于 /frameworks/base/services/jni/com_android_server_SystemServer.cpp 中；

init1 函数呼叫 libsystem_server 中的 system_init 函数，该函数位于/frameworks/base/cmds /system_server/library/system_init.cpp 中，该函数将 SurfaceFlinger/AudioFlinger /MediaPlayer/CameraService 等组件注册到 ServiceManager 中

system_init 函数反过来呼叫 java 类 com.android.server.SystemServer 的 init2 函数；

5.在 init2 函数中，android 创建了 serverthread，在该 thread 中 android 开始注册各种 service 到 service manager 中

包括 EntropyService,PowerManager,ActivityManager,Telephony,PackageManager,ContentManager,ContentProvider,

BatteryService,HardwareService,AlarmManager 等等。

注意该线程使用 Looper 来执行 thread

至此 android system server 启动完成。

framework 主要是一些核心的文件，从后缀名为 jar 可以看出是是系统平台框架。

\system\framework\am.jar
\system\framework\am.odex
\system\framework\android.awt.jar AWT 库
\system\framework\android.awt.odex
\system\framework\android.policy.jar
\system\framework\android.policy.odex
\system\framework\android.test.runner.jar
\system\framework\android.test.runner.odex
\system\framework\com.google.android.gtalkservice.jar GTalk 服务
\system\framework\com.google.android.gtalkservice.odex
\system\framework\com.google.android.maps.jar 电子地图库
\system\framework\com.google.android.maps.odex
\system\framework\core.jar 核心库，启动桌面时首先加载这个
\system\framework\core.odex
\system\framework\ext.jar
\system\framework\ext.odex
\system\framework\framework-res.apk
\system\framework\framework-tests.jar
\system\framework\framework-tests.odex
\system\framework\framework.jar
\system\framework\framework.odex
\system\framework\input.jar 输入库
\system\framework\input.odex
\system\framework\itr.jar
\system\framework\itr.odex
\system\framework\monkey.jar
\system\framework\monkey.odex

\system\framework\pm.jar 包管理库
\system\framework\pm.odex
\system\framework\services.jar
\system\framework\services.odex
\system\framework\ssltest.jar
\system\framework\ssltest.odex
\system\framework\svc.jar 系统服务
\system\framework\svc.odex