我是如何从Android开发转framework开发的?

小迪vs同学 鸿洋 2023-01-03 08:35 发表于北京

本文作者

作者：**小迪vs同学**

链接：
https://blog.csdn.net/u013936727/article/details/127876172
本文由作者授权发布。

转framework开发快一年了，一直都想写一篇文章，分享一下自己的工作心得，也让做应用开发的小伙伴对framework开发有一定的了解，但因为种种原因耽搁了，今天就趁着工作闲暇之余，聊聊我从应用开发转framework开发的心路历程，自己也是刚开始学着写文章，文笔不太好，请见谅。

1 本人履历

一个混迹Android圈7年的小菜鸟。

做过应用开发，996那种。

做过Android讲师，在小白面前吹牛逼那种。

做过技术支持，全国到处跑那种。

2 越来越卷的应用开发

曾经跟所有Androider一样，在应用开发领域为所欲为，徒手掰Handler，脚踩ViewPager，硬刚ListView，但随着时间的推移，技术的更新迭代，ViewPager2，RecyclerView，插件化，组件化，Kotlin，jetpack，mvvm，flutter，compose，简直没完没了了，随着年龄越大，越来越感到无力，或许未来某一天学不动了，真的会被后浪给淘汰，于是，第一次真正思考起自己的职业规划，要不要转行呢，可是转行又能做什么呢，别的圈子也卷啊。

3 应用开发向framework开发过渡

某天，我在某聘上看到现在公司招聘Android开发岗，以为是应用开发，就电话聊了聊，聊了一个多小时才知道是做framework开发，起初我理解的framework开发就是定制系统，类似MIUI之类的，但是聊完之后我才知道，其实framework也有很多方向，毕竟Android源码那么庞大，根据模块划分有：

- wifi/bt-wifi和bluetooth
- multimedia-多媒体
- telephony-电话
- ...

根据业务划分有：

- TV-电视
- Iot-物联网
- Auto-车载
- ...

比起应用层层出不穷的新技术，framework层就显得十分的成熟稳重，没有太多的变化，核心技术一直都是那些东西，能够把其中一个方向给研究透彻就已经很牛掰了，想想之前卷到天际的应用开发，顿时来了兴趣，于是毅然决然辞职来到现在的公司做framework开发。

来到这里之后，真的是神仙打架，全是一群各个领域的大佬，只想说，被带飞的感觉真好。刚来就从framework最基础的东西开始学起，搭建framework开发环境，主要包括：

- Android Studio：查阅源码。
- git：拉取/上传代码。
- 编译源码所需要的环境，如openjdk、python、 libssl-dev、audit2allow、m4等。
- 远程工具向日葵/NoMachine：因为编译源码需要在Linux环境下，所以每个人都有两台主机，Windows和Ubuntu，一般都是windows远程控制Ubuntu。

- 其他一些辅助工具。

搭建好开发环境后，就可以开始从公司服务器下载Android源码，编译源码，刷机，其中最耗时的就是编译源码，第一次编译可能得花三四个小时，不同版本的Android源码编译时间不一样，一般版本越高，编译时间越长，正所谓，一杯茶，一包烟，一个bug改一天，毫不夸张，就这整个流程下来，我花了整整一周，第一次开周会的时候，同事的周报我都看不懂，都是一些framework的问题，但转过头想想以后我也能解决这些问题，顿时感觉越来越兴奋。

学完刷完机之后，开始尝试在源码里面加入自己的代码，看它会不会生效，最简单的就是加个Log日志，然后在adb logcat中看是否打印出来了，这里说明一下，查看日志不是在Android Studio中查看，AS只用于查看源码，不作为调试工具，调试都是使用adb相关工具。

自己的代码能够生效之后，就开始着手处理一些简单的问题了，起初都是处理java层的一些bug，并不涉及到native层代码，比如隐藏设置里面的一些不常用的设置选项，修复关于设备里面IMEI号显示不正确的问题等等，虽然很简单，但是对于刚入门的我来说也是很懵逼的，UI问题还好，可以根据文案，全局搜索，定位到代码处，非UI问题，那就得搞清楚Android系统的执行流程了，这就需要先了解Android源码的目录结构，每个目录都是干啥的，就跟app项目结构一样，然后要去梳理Android的启动流程，核心的一些服务和进程，PMS，AMS，WMS等等之类，进程之间的通信方式，这又需要对Linux有一定的了解，了解这些之后你才知道这个bug可能跟什么模块有关系，这就缩小问题定位范围了，然后到对应的模块下面去进一步查找，好在公司有很丰富的学习资源，需要了解什么知识点都可以在资源库里查得到，这节省了很多时间，不得不说，大公司确实不一样。

经过一两个月时间的熟悉之后，渐渐的对framework开发流程越发娴熟，很多简单的问题都能很快解决掉，但是碰到那种很怪异的问题，就很费时间，比如插入大容量SD卡黑屏，往SD卡拷贝大文件会卡顿，甚至出现anr，framework最害怕碰到的就是anr，因为anr日志给出的报错信息一般都不是根本原因，多半是底层出现了什么问题才导致上层出错，这种问题，就算自己处理十天半个月都不一样搞得定，最后都是请其他岗位的同事协助解决。

4 framework开发心得

时至今日，可以说已经对framework开发有了比较全面的认识，但仍然掩盖不了自己是个菜鸟的事实，很多东西都需要去学习，比如linux，jni，c/c++，IPC，aidl，hidl，adb shell等等，目前正在恶补这些东西，正应了那句话，学得越多，越觉得自己是个菜鸡。但是从另外一个角度来看，技术的不足恰好激发了自己的学习欲望，不断鞭策自己向上进步，慢慢的对未来的焦虑也逐渐减少，想想之前的自己，再看看现在的自己，照镜子都觉得自己那么帅，每天进步一点，量变引起质变，这是我目前抱有的信念。

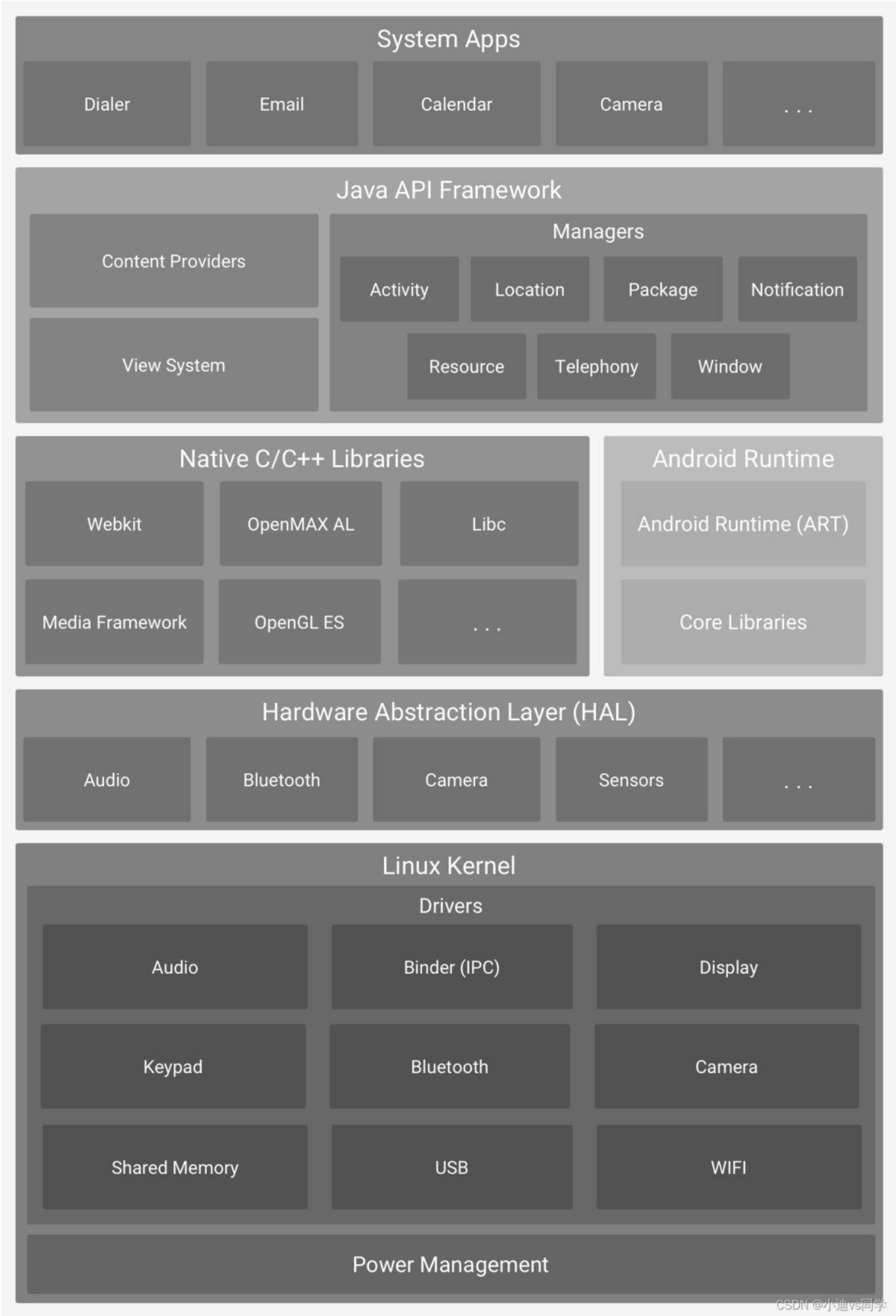
5 To App Developer

静下思考，认真规划，戒骄戒躁，步步落地。如果你不想卷应用开发了，framework开发可能是你可以选择的一条不归路。我整理了一份framework学习路线和方法，供大家参考学习。

6 framework学习路线和方法

Android架构

做framework开发，首先必须清楚Android系统架构：


System Apps
Dialer
Email
Calendar
Camera
. . .
Java API Framework
Content Providers
View System
Managers
Activity
Location
Package
Notification
Resource
Telephony
Window
Native C/C++ Libraries
Webkit
OpenMAX AL
Libc
Media Framework
OpenGL ES
. . .
Android Runtime
Android Runtime (ART)
Core Libraries
Hardware Abstraction Layer (HAL)
Audio
Bluetooth
Camera
Sensors
. . .
Linux Kernel
Drivers
Audio
Binder (IPC)
Display
Keypad
Bluetooth
Camera
Shared Memory
USB
WIFI
Power Management

这里概括一下，Android架构共分五层，分别是：

• **应用层(Application)**

各种上层运行的的app，包括系统内置app和第三方app，例如Launcher、Settings等

• **框架层(Java Framework)**

提供给应用层使用的Java库，例如WMS，AMS，PMS，各种View等

• **Native层/Android运行环境**

提供给Java Framework层使用的C/C++库，例如OpenGL

• **HAL层(硬件抽象层)**

为Java Framework层提供硬件功能，例如相机、蓝牙、各种传感器等

• **内核层**

支撑Android系统运行的Linux内核

具体说明可到Google官网查看官方对评价平台架构的说明。
https://developer.android.google.cn/guide/platform?hl=zh-cn

而我们常说的framework开发其实大多数指的是中间三层的开发，即框架层、Native层、HAL层，可想而知，framework开发的内容是非常多的，那么什么场景下需要framework开发呢，或者说framework开发能做些什么呢?

开发内容

简单来说，如果需要修改Android源码，都需要framework开发支持，具体场景包括：

• 系统定制，比如MIUI、Flyme、ColorOS、OriginOS Ocean。

• 硬件接入，常见的如车载系统的温度、车速传感器等，不同厂商的硬件标准也可能不同。

• 系统裁剪，其实也可以说是系统定制的一种，把Android中不需要的一些模块裁掉以提升系统的流畅性。

学习路线

语言基本功

语言基本功对于阅读Android源码是最基础的，framework开发写代码可能不是很多，但阅读源码是家常便饭的工作，所以需要有扎实的语言功底，其中包括：

- **Java**

Java Framework作为framework开发的入口，核心功能都是用Java写的，如果你是从app开发转framework开发，这一点肯定不是问题。

- **C/C++**

因为native层和hal层都是c/c++写的。如果没有c/c++基础，也可以先从java framework做起，慢慢过渡到native层，期间需要大量补习c/c++基础。

重要技能

- **常用的Linux命令和adb命令**

首先，Android本身是基于Linux的，其次framework开发工作都是在Linux系统中完成的，包括编译和调试，如果深入学习Linux系统，那绝对是有很大帮助的。

- **Android源码目录结构**

搞清楚每个目录下都有些什么模块，整体感受Android架构。

- **Android源码下载、编译、烧录**

framework开发无非就是下载源码、阅读源码、修改源码、编译源码、烧写lib/镜像。

- **熟练掌握Android中的重要模块**

Android是一个很庞大的系统，包括很多的模块，彻底理解所有模块是十分困难的，但一些常用的模块必须需要掌握的，比如：

1. Android启动流程：核心进程(init、logd、adbd、servicemanager、zygote)以及进程之间的父子关系。

2. 重要服务：servicemanager、PackageManagerService等。

3. App启动流程。

4. SystemProperties，系统属性。

其他的根据实际工作中接触到的模块来深入学习，比如：

1. WiFi

2. Bluetooth，蓝牙

3. Telephony，通话

4. Settings，系统设置

5. Launcher，桌面

6. Media，包括Camera，Video，Audio

7. Display，图像显示Surface相关

8. Storage，存储

9. Sensor，传感器

• JNI

java与C/C++相互调用。

• IPC(Inter-Process Communication)

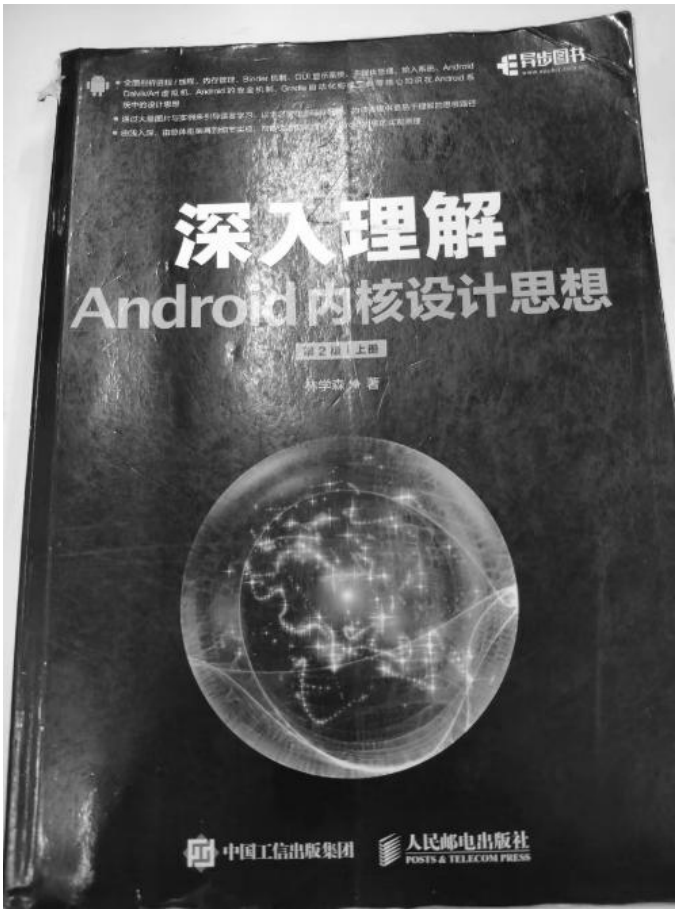
进程间通信，主要了解binder(AIDL、HIDL)。

• logcat日志抓取和分析

通常处理一个系统bug时，首先会根据系统logcat日志来定位原因。

学习方法

网上有很多framework相关的资料，但我看了一下，大多都不系统，并且恰饭居多，根据我个人的经验总结出以下一些学习方法以及学习小技巧。



首先有条件的话最好买一本framework相关的书籍，一个是因为书籍会比较系统性，不会存在有头无尾的情况，二是书籍稍微权威一点，错误率会相对较低，如果一开始你就接受一个错误的观点，后面会很难更正过来。个人推荐《深入理解Android内核设计思想》这本书，我个人也看过这本书，感觉写得非常好，需要注意的是，这本书是以Android N的源码来展开的，所以下载源码的时候建议也下载Android N的源码。

然后根据作者的思路和实际你download下来的源码进行学习，阅读源码的时候不用纠结每一行代码的意思，只需要找到主线即可，因为一个类或者一个方法可能涉及到多个模块的功能，你只需要找到你研究的那个模块的代码即可。如果你是从app开发转到framework，那么建议先从app的启动入口开始看起，一路追踪到Activity的创建以及onCreate()方法的执行，阅读过程中可能会比较晦涩难懂，但不用纠结，主要是感受一些framework层如何过渡到app层，这样让framework和Application衔接到一起，更容易感受到framework的轮廓和边界。

在系统性学习的过程中遇到一些边边角角知识点不懂的，可以网上查阅相关资料，扩展自己对framework的知识面。

很重要的一点，每学习完一个模块都需要对当前学习内容进行总结，最好是能绘出模块的架构图和流程图，总结能加深你对源码的理解。这里推荐使用Android Studio的plantUML插件来完成，插

件官网有使用教程。

https://plantuml.com/zh/

7 总结

学习framework是一件非常枯燥的事情，原因在于相比于app开发的coding，framework更侧重于对源码的reading，但学习它能让我们更深入的了解Android系统，增加自身的知识储备，降低自己在行业中的可替代性，延长自己的职业生涯，提高自身身价。如果你有想法学习framework，或许我们可以一起成长。

最后推荐一下我做的网站，玩Android: *wanandroid.com* ，包含详尽的知识体系、好用的工具，还有本公众号文章合集，欢迎体验和收藏！

推荐阅读：

一文搞懂 Gradle 配置，7.0之后有哪些变化?

App防抓包的 8 个实践

升级 Android 目标版本到 31(S) 这么多坑

扫一扫 关注我的公众号

如果你想要跟大家分享你的文章，欢迎投稿~

┏(^ 0 ^)┛明天见！

阅读原文

python中yield怎么用? 和return有什么区别

Python之禅

上线 6 天，下载破 10 万，ChatGPT 中文版 VSCode 插件来了！

GitHubDaily