

重磅 | 嵌入式学习路线图

原创 布道师Peter 人人极客社区 2020-11-14 09:42

收录于合集

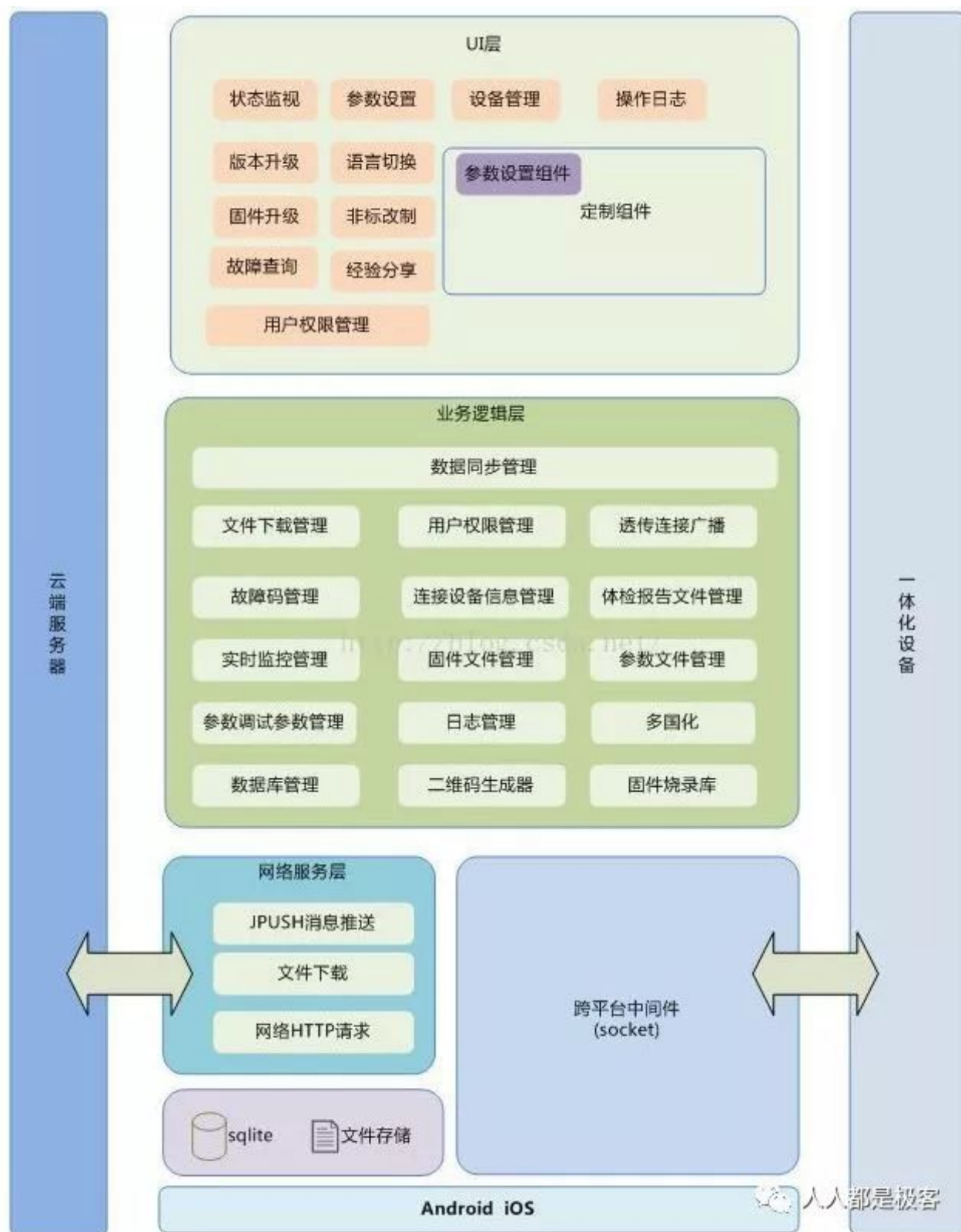
#Linux 98 #嵌入式 1

可能是年前跳槽的比较多，遇到不少同学咨询到嵌入式行业发展和职业规划的问题，这里总结一下嵌入式行业的机遇和选择，希望对读者们有所帮助。

我们暂且宏观上把程序员分为3类：业务类，专业类，系统类。

- 业务类

业务类更多的是在应用程序。随着移动互联网的快速发展出现一批 UI 设计师，这里的设计师是指 APP 的界面设计，在注重用户体验的今天对于界面的设计出现水涨船高的需求。一时间 Android, IOS 的 APP 开发者如雨后春笋般涌出，待遇也是不低。高级的应用程序员除了界面的开发外也会涉及程序内部的业务逻辑，现在的 APP 逐渐演化成很多层的架构，比如分为业务逻辑层，基本功能模块层，UI 界面层等，如下图所示：



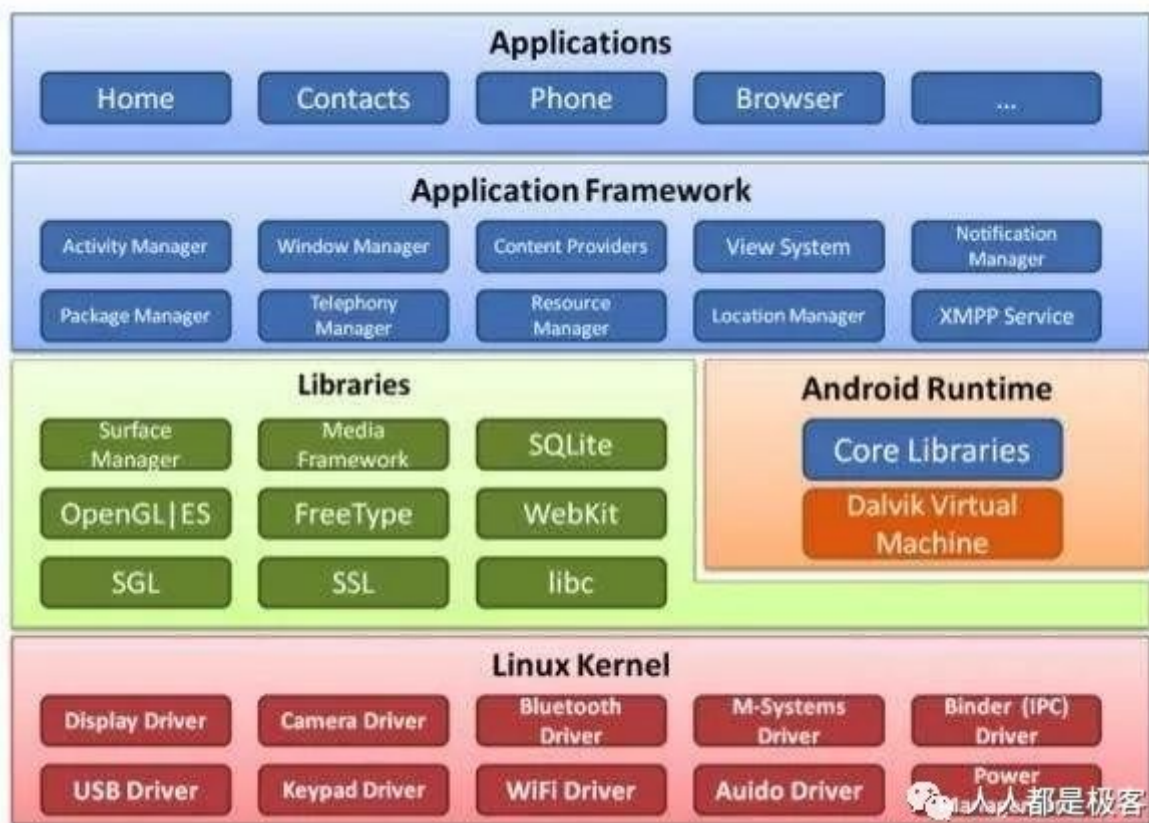
一个 APP 就包括了很多内容。如果志向写应用程序的小伙伴，我建议你先练好基本功：数据结构，算法导论，网络编程，数据库等。然后根据感兴趣的行业深耕学习。因为这一行的开发者和业务联系比较大，后续换行业就不太吃香，比如美图秀秀开发者更多的是注重在图形显示这一块，网易云音乐更多的就注重在音频这一块。当然不能以偏概全，很多能力是通用的，我这里的举例主要是相对而言。应用程序开发者会随着业务的多变性经常加班 coding，解 bug。所谓的码农更多的是出现在这一领域。

- 专业类



应用程序开发者是完成业务的直接执行者，夸张的讲应用程序开发者就是 API 调用者，但这些 API 是如何实现的？不同的专业领域有不同的 API。以上图 APP 框架为例，HTTP 网络请求就是调用了网络领域的 API 接口，SQLITE 就是调用了数据库领域的 API 接口，再比如目前比较火的人工智能，语音识别，图像处理等都属于专业类领域。这一领域的工作者拥有比较强的理论知识，算法知识，多以研究生或者博士生为主。这一领域的优势是待遇高，可替代性低，但也因为研究领域的专一性在找工作时也面临一些局限性。

- 系统类



这里系统主要指 linux 系统。系统是个太大的概念，有上层 framework 系统也有底层内核机制，也包括驱动开发，甚至硬件也要知道。这一领域的人更多的是在嵌入式行业。在操作系统领域对知识要求很多：

1. 看懂电路图
2. 看懂芯片手册
3. 有编写，移植驱动的能力
4. 懂内核的实现机制
5. 懂C语言，C++，JAVA等

这一行的优势是学好后行业通杀，大公司基本都有这方面人才的需求；相对做应用程序的人不会经常因为业务需求的变动搞得天天加班；行业稳定越老越吃香，不太会出现程序员35岁职业生涯问题。另外操作系统是很通用性的知识，夸张的讲只要是 IT 行业，学点操作系统的知识肯定是如虎添翼的作用：

1. 硬件工程师通过学习可以理解软件的运行原理
2. APP 工程师学些系统知识更有利于走向全栈
3. 大学生学习linux对找工作多有益处
4. 学好 linux 即可以做开发也可以做运维
5. 永远不会淘汰的技术，只会越老越吃香

嵌入式 linux 学习路线

本文把操作系统默认为 linux，讲讲怎么学习嵌入式 linux 系统。简单地说，嵌入式 linux 系统里含有 bootloader、内核、驱动程序、根文件系统、应用程序这5大块。而应用程序，我们又可以分为：C/C++、Android。所以，嵌入式Linux+Android系统包含以下部分内容：

- ARM
- Bootloader
- Linux内核
- 驱动程序
- 根文件系统
- Android Framework
- 使用C/C++编写的应用程序
- Android APP

根据以上内容我准备了一系列的达人课程，希望对有志于成为全栈嵌入式开发者有所帮助，这里分享下课目表安排和学习经验：

1. ARM

学习硬件知识的目的在于能看懂原理图，看懂通信协议，看懂芯片手册。这里推荐一些书：

- 《微机原理》，可以理解一个计算机的组成原理
- 《数字电路》，掌握一些逻辑运算，理解各种门电路的原理
- 《ARM体系结构与编程》，对ARM的运行原理解释的很到位

我相信看完这些书对ARM和硬件知识的掌握足够了，对于初学者只想浅尝辄止的了解，我推荐之前的一个chat《一小时教你学会 ARM 架构》。

2. Bootloader

bootloader有很多种，vivi、u-boot等等，最常用的是u-boot。u-boot功能强大、源码比较多，对于编程经验不丰富、阅读代码经验不丰富的人，一开始可能会觉得难以掌握。但是，u-boot的主要功能就是：启动内核。它涉及：读取内核到内存、设置启动参数、启动内核。按照这个主线，我们尝试自己从零编写一个bootloader，这个程序相对简单，可以让我们快速理解u-boot主要功能的实现。相关内容有：

- u-boot分析之编译体验
- u-boot分析之Makefile结构分析
- u-boot分析之源码第1阶段
- u-boot分析之源码第2阶段

- u-boot分析之u-boot命令实现
- u-boot分析_uboot启动内核

3. Linux内核

内核在工作中的作用可以说是潜移默化的，虽然看起来没有直接性的工作项目，但绝对是走向架构师的必经之路。本课程会涉及到以下知识点：

- Linux总线，设备，驱动模型的探究
- Linux设备树的深入理解
- Linux的启动流程
- Linux设备和驱动的相遇
- 动手定制一个开发板

我相信通过这些课程的学习可以基本上掌握内核的运行原理，入个门是没有问题的。另外推荐本书给大家《linux内核设计与实现》。

4. 驱动程序

驱动程序=Linux驱动程序软件框架+ARM开发板硬件操作，有了ARM的知识和阅读数据手册电路图的能力，再加上Linux内核的运行原理，基本上写驱动就是API调用的体力活了。这里推荐宋宝华老师的《linux设备驱动开发详解》。后续关于驱动的内容暂定如下：

- LCD驱动程序
- 触摸屏驱动程序
- USB驱动程序
- NAND FLASH驱动程序
- NOR FLASH驱动程序
- 网卡驱动程序
- 声卡驱动程序

5. 根文件系统


在开发应用程序时，也需要搭建文件系统，把各种库、配置文件放进去；在发布产品时，你还需要修改配置文件，使得产品可以自动运行程序；甚至你想实现插上U盘后自动启动某个程序，这也要修改配置文件；这一切，都需要你理解根文件系统的构成，理解内核启动后是根据什么配置文件来启动哪些应用程序。根文件系统相对比较简单，可以根据以下路线学习：

- Linux根文件系统目录结构
- 移植Busybox
- init进程介绍及用户程序启动过程
- 使用glibc库
- 制作/使用文件系统映象文件

成为全栈嵌入式开发者任重而道远，让我们按照上面的学习路线一步一步行动起来。

人人都是极客

针对嵌入式人工智能，边缘计算，物联网等专业技术分享和交流平台，内容涉及arm，linux，android等各方面。





收录于合集 #Linux 98

[上一篇 · Linux 调试调优技术](#)

喜欢此内容的人还喜欢

从 Hadoop 到云原生，大数据平台如何做存算分离
Juicedata



IGBT适用于ZVS 还是 ZCS?
英飞凌工业半导体



MySQL 8.0 数据字典表
一树一溪

