关于Android学习的三个终极问题

缘起

读研的时候，有一次和同学一起上自习。我在学习，他也在学习。我想，大家每天都一起上课，记笔记，上自习，完成作业，智商也差不多，为何考试的时候有时候差距会很大呢？这个问题我直到今天也没有答案，这几天和朋友闲聊说到这个事情。他们有得说是智商有差距，有得说是学习的时候心不在焉——看着在学习，其实已神游大千世界。.....，不过，我自己从来没有下过类似的结论。我武断的猜测，这个问题应该是没有让人心服口服的答案。

从事软件开发行业后，发现要学的东西越来越多。尤其是近几年，发现未知的知识实在是太多。恐怕IT人活到老学到老的压力是绝无逃避的可能性了。那么，关于学习，我们需要注意什么呢？2015年google开发者社区一个集会上，我曾经分享过一些思考。4年过去了，又积累了不少经验教训（其实就是老了不少），现在想总结这些年的观察，做一个总结。我感觉可能需要至少两篇文章才能说完。一个是本篇的关于Android学习的三个终极问题，另外一个是对一专多能的思考。

去年我在周爱明老师的《大道至简》一书里看到对“反思”一词的定义，深有感触。书里说，**反思，既不是设问，也不是反问，而是对一个问题的持续关注**。这么来看，十余年来我对考试成绩好坏原因的思考就是一种反思。

在继续本文之前，我想说，我这个人讲不出什么大道理，能讲的东西都是自己想过、尝试过、甚至在周围朋友，同事身上一起试验过的。即使不能从中学到什么有价值的东西，但看到像我这样的一个圆脸中老秃胖其实一年也没多少天是无忧无虑得，你也可以开心一下了。

关于学习的三个终极问题

关于学习，我认为有三个终极问题：

这三个终极问题是：

1. 学什么？2015年时还没有flutter，RN我记得刚起。不知道在那个时候意气风发的Android红人们现在是不是会有点感触了。过了4年了，现在该学什么？Flutter？AI、区块链？是为了兴趣爱好而学，还是为了保家卫国光宗耀祖扬眉吐气世界更美好而学？
2. 怎么学？这是在你刚千思万考解决第一个问题的时候接下来的另一个大难题。是ALL IN新知识？还是新旧知识两手都要抓？不论哪种选择，你都很难和长江后浪比精力，比速度。对待新知识、新技术，长江后浪是光脚不怕穿鞋的，靠试错来成长。

3. **学到什么程度为止？**最后，你开始学了，也知道怎么学了。然后呢？钻研技术是无止境的，你要学到什么程度为止？够用就行吗？够用是在哪个范围够用？我自学完写个demo叫够用吗？还是说学完后要给公司创造多少多少价值才行？我公众号的首篇文章“深入理解的目标是什么？”和这个问题有一点关联。

很多时候，很多人并不会意识到学个知识还有什么三个终极问题。比如，周围人都学某某技术，所以我要学。或者，学某某技术有前途和钱途，那我也学。坦白来讲，我很羡慕这些人。在和平、稳定的年代，随大流是稳妥的选择。我很后悔读研的最后不该听了中科院某老师的授课。2007年，他“深度思考”了中国经济，得出了一个结论，说咱们的经济列车高速开了这么久了，肯定会下降的，然后房价会跌。所以，工作后的前3年我看着房价飞涨的时候依然坚信这个结论。幸好父母亲果断，举债买房。当时我加班夜里回来后听说自己背负了100多万的房贷还很不开心。没想再过几年我发现自己居然少背了几百万的房贷。 所以，要是你没有意识到有这三个问题，你是幸福和幸运的——如果环境没有发生变化，你也无须改变。

上面三个问题是我自己在学习问题上的反思——反思的潜意思是我也没有答案。我觉得也很少有人能给出明确的、普适的答案。我有一种强烈的预感，**在关于学习的这三个终极问题上，每个人一定都是在自己的无人区里探索**。

承认无人区的存在是有百利而无一害的。我以前每次换工作都要找机工的杨总编等朋友请教，但发现他们的意见总是不理想。终于有一天我明白了，原来我在自己的无人区中，里边只有我一个人。我的家人，朋友都不能真正理解我，他

们解决不了我的困惑。不过，当在无人区里挣扎的我知道还有那么多亲人、朋友是支持我的，勇气就会更足。坚信总会有一天，我会短暂得从无人区出来，然后再走入下一个更高级的，更有挑战的无人区。

对了，去年我才发现温伯格原来早在《**技术领导之路**》一书里就曾提到过类似的情况，那一章的中文名叫“**领导的成长**”。

接下来，我要把这几年从事Android开发相关工作中看到的东西结合自己的认识讲一讲。我觉得它们一定能帮助到在Android技术领域里谋生的一些兄弟姐妹们。

关于Android学习方面的一些探讨

根据我的经历和认识，Android技术领域划分按从下到上可分为三个大的层次：

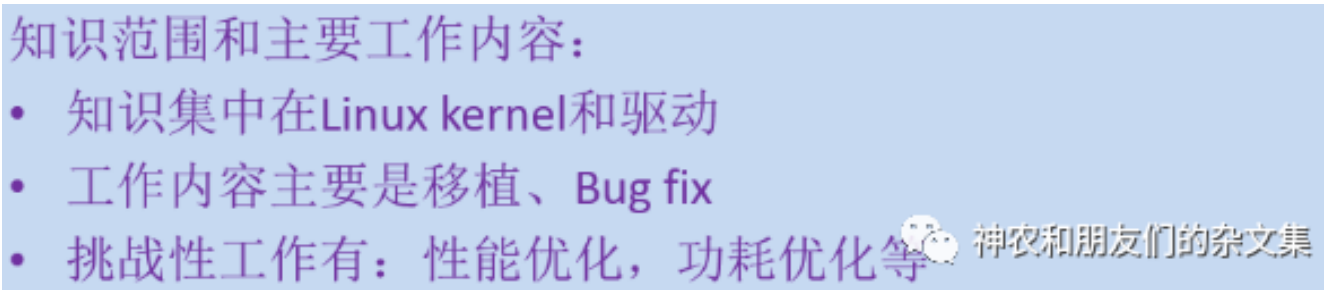
1. **Android底层开发**。这个领域的开发工作主要集中在设备厂商（比如华米OV）、芯片厂商（华为高通）等。最近这两年，伴随IoT技术的发展，一些有能力的互联网公司也需要这方面的人才。
2. **Android系统层开发**。这个领域的开发也是集中在设备厂商。但我最近发现头条这样的互联网公司也大量招聘懂Android Framework的工程师。另外，新兴的车联网、车企也对这方面人才有大量需求。
3. **Android应用层开发**。这个就不用说了，竞争非常激烈，技术迭代的速度飞快。

接下来，我会从知识范围和主要工作内容、学习路线以及需要注意培养的专业素养这三个方面来考察下上述三个

Android技术领域。

Android底层开发

Android底层开发涉及到的知识范围和主要工作内容如下：

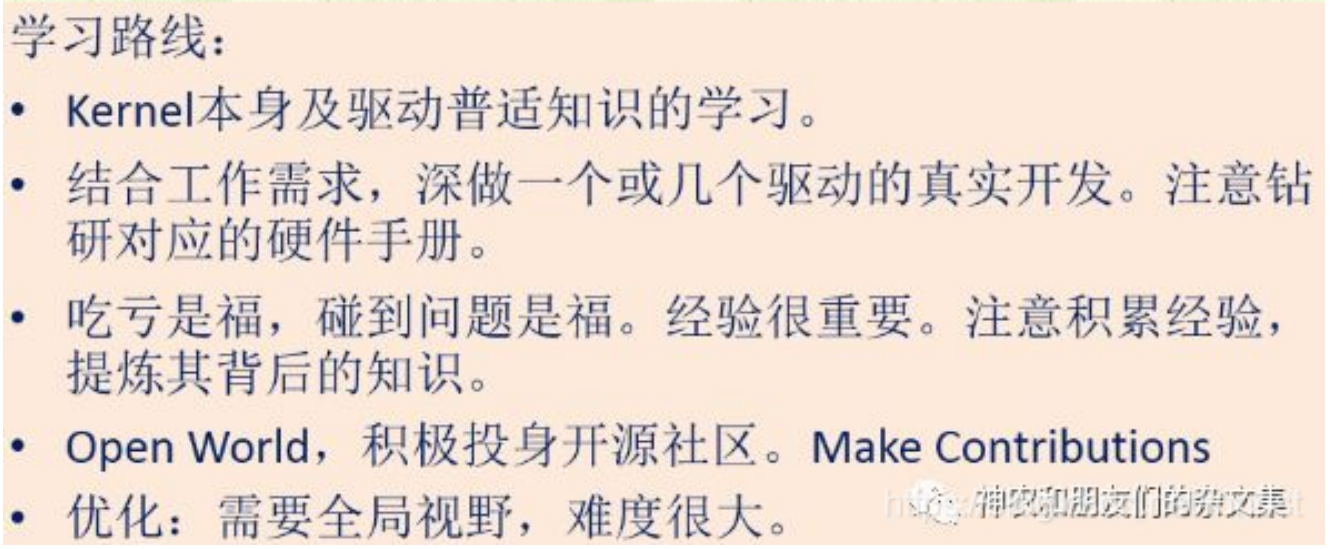


从事底层开发的码农们叫BSP或驱动开发工程师。涉及的专业知识集中在Linux Kernel或驱动方面。

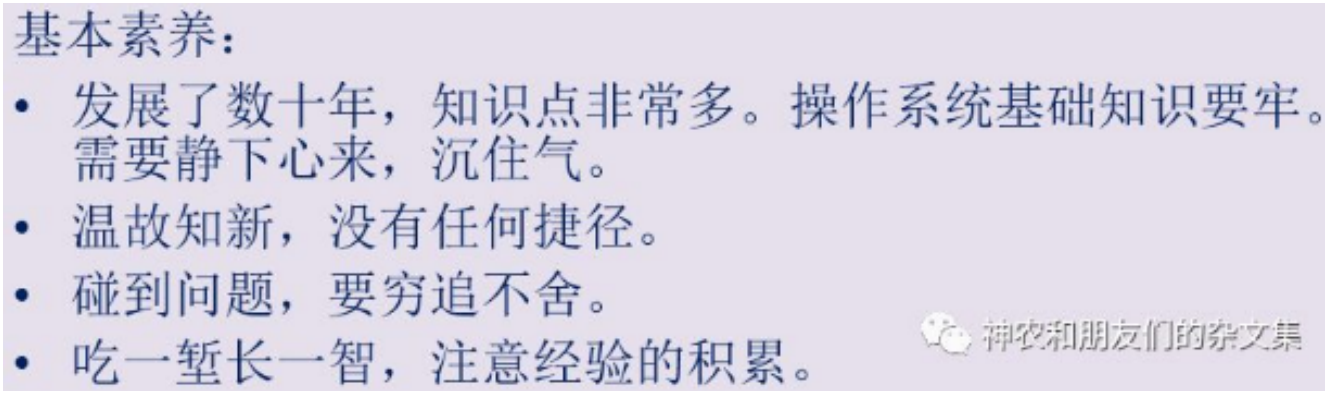
Linux Kernel是无数人为之倾倒的知识金矿。当年我也认认真真读过毛德操老师的《Linux内核情景分析》这样的源码分析书籍。这本书对我的影响非常深，以至于我策划的深入理解Android系列的几本书可看做是毛老师的手法在Android系统上的复制。

底层开发的工作内容主要是移植和Bug Fix。在某些公司的日常工作中，只要找到解决bug的patch，打上去就好了。为什么会这样？因为Kernel是一个比较稳定的软件。在这里，稳定压倒一切。当然，其中也有非常具有挑战性的工作，比如优化相关的工作等。

如果你想从事Android底层开发，下面是这个领域的学习路线：



在这个领域里工作，请牢记“吃亏是福”这句话。吃亏意味着积攒经验。这个领域里的经验是至关重要的。所以，我在下面的基本素养里还会提到它：



Android系统层开发

接着来看Android系统层开发。它涉及的知识范围和主要工作内容如下：

Android系统层开发的主要工作有因Android系统升级带来

的移植工作、或者是诸如华米OV各家系统的产品特性（因为原生Android系统没有，所以需要自己开发）。

最后，随着除手机外的智能设备兴起（如智能POS、智能汽车），这些特定设备上也有许多在系统层需要开发的工作。

相比底层开发，系统层开发有更多的个性发挥空间。MIUI、EMUI、color OS等等就是系统层个性发挥的结果。简单点说，如果可以把从事底层开发的公司看做是硬件公司，那么从事系统层开发的就可以看做是软件公司了。

我这个说法肯定是不对的，但这里想说的是这两个领域的思维方式不太一样，因为要解决的问题也不一样。

Android系统层的学习路线如下：

从编程语言上来说，Android系统层有很大一部分是用Java开发的。另外还有一大部分是用Native（C/C++）语言开发的，

所以，要想在这个领域做到游刃有余，对Java和C++语言要了解。题外话，还没了解C++的同学可以直接上C++11。它和C++98/03不同（C++之父Bjarne Stroustrup甚至说过

“C++11看起来像一门新的语言”）。

关于C++11的知识，我在CSDN博客上"C++11学习"一文（也是《深入理解Android JVM ART》一书的第五章，因为ART JVM几乎都是C++11写的）做过一个较为全面的介绍（地址在文后列出）。

Android系统层是相当吸引广大开发者的一个领域。比如很多知名Android技术公众号都会时不时讲一讲Handler的实现、UI绘制的流程、Binder的工作原理、消息传递机制等这样的知识。 这些内容早在《深入理解Android》系列的卷1、卷2和卷3都覆盖了。只不过时过境迁，年轻的人现在更适应新的传播方式。

除了这些通用知识，Android系统层还有相当一部分属于**专业领域知识**。比如，蓝牙、Wi-Fi、NFC、Telephoney（其中的双卡双待功能算是中国特色的功能）。如果你从事类似这样的专业领域的工作，**请注意把精力放在专业知识本身**，而不是Android里的代码。比如，Wi-Fi、NFC、GPS、蓝牙技术出现得远比Android早。它们各自有非常深厚的专业、理论知识。在Android中，你看到的相关代码无非是它们在特定系统上的实现罢了。**你务必要先了解这些专业、理论知识**，才能看懂Android上的代码实现。更未雨绸缪的说法是，如果将来没有Android系统的话，你一样可以靠着对专业理论知识的了解另起炉灶。

这也是我说“**基于Android、高于Android**”的意思。一个最近发生的例子是，有个帮我审稿JVM ART一书的审稿专家现在研究Flutter里的虚拟机。我相信他会比别人走得更快，更轻松一点。

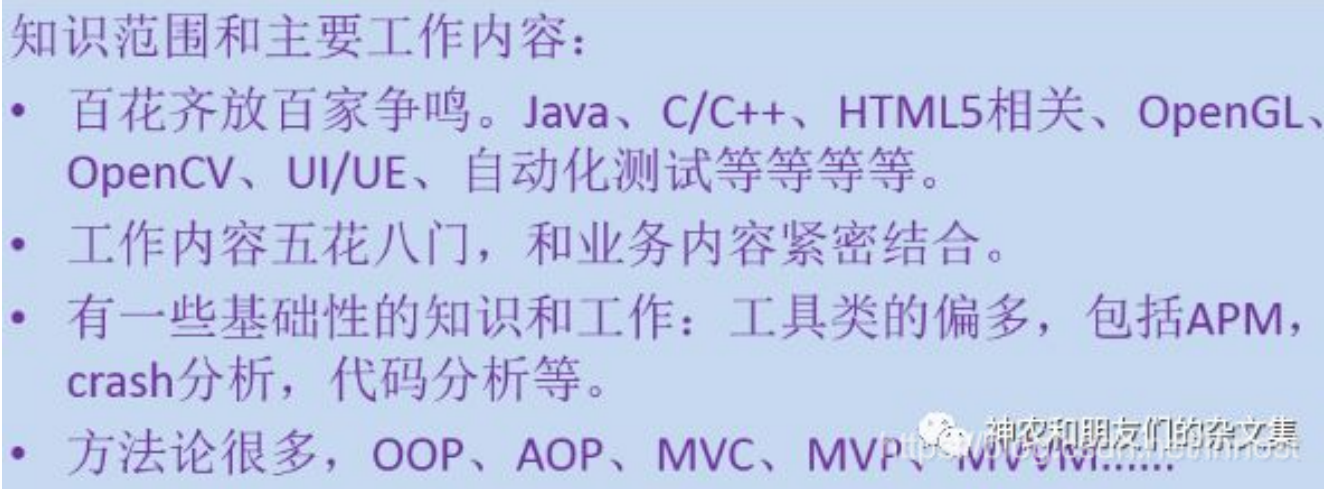
Andriod系统层开发的基本素养：

系统层这几年发展也很快（想想Android的发版速度），靠个人力量来追踪这么一个复杂系统已经很困难了，需要大家一起来知识共享。

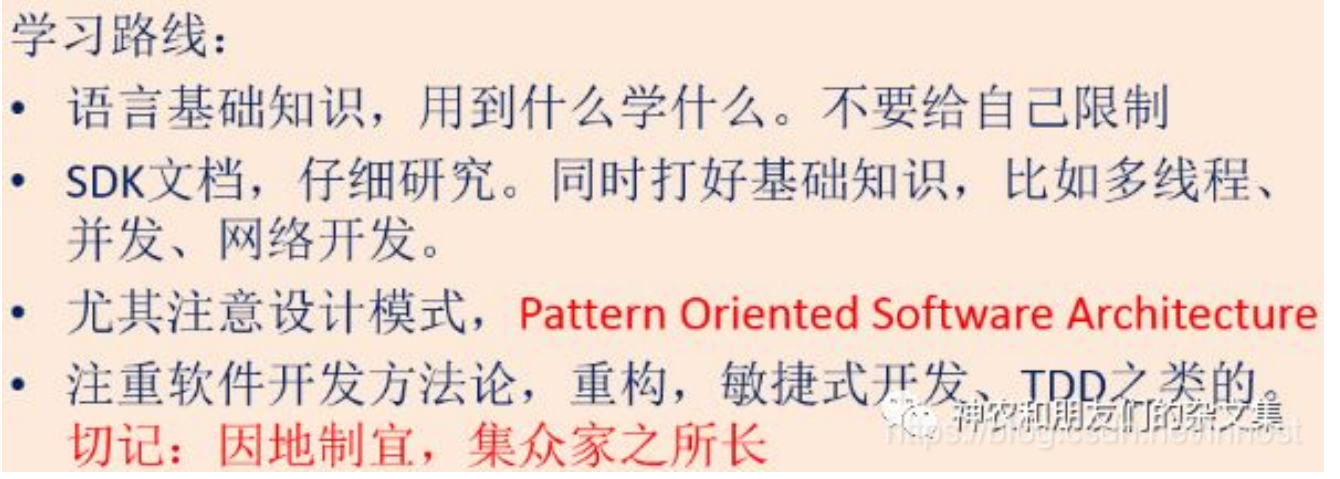
虽然很多公众号会写一些文章来介绍Android某个新系统的特点，但其实在设备厂商里是有不同的团队来跟踪新系统里不同模块的变化得。比如，除了Google I/O外，谷歌给设备厂商还组织了一个叫Android Bootcamp这样的类似培训一样的集会，里边会比较详细的、按功能、模块来介绍新系统的特点。不过，参加Bootcamp所获得的资料往往要保密。

Android应用层开发

接着来看最后一层——Android应用层开发的知识范围和主要工作内容：



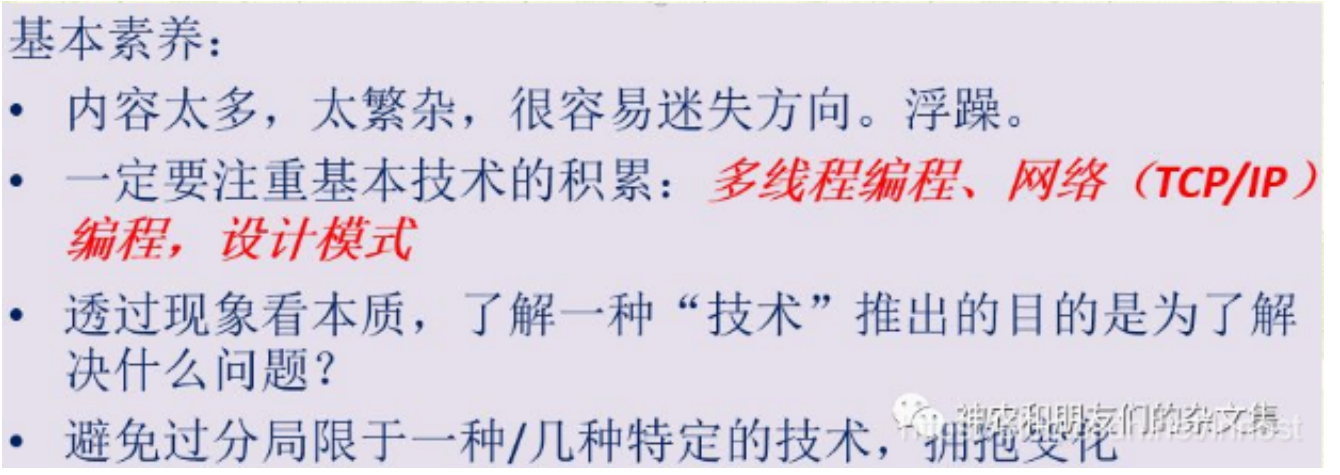
应用层开发的特点就是知识点多、更新速度极快。相信大家都有感觉。针对这个层面的开发者，我想，就不要拘泥于什么门派了，**需要什么就学什么**（注意，**不是有什么就学什么**）。相比其它两个领域，**应用层开发是为业务需求服务的**。所以，为了更好得服务业务需求，该学什么就学什么。比如，我最近的方向转到应用层开发了，但我所在行业的应用主要用H5+webview来承载业务逻辑，那么我就重点花时间学习了这块的内容。



我个人认为从事应用层开发同学的比例最大，也是最能体会“跟不上知识更新步伐”感受的人。这一领域中有没有什么**基础关键技术**？我多年观察下来觉得在应用层开发领域，多线程编程、网络编程、设计模式等几个是基础关键知识，无论如何要想办法抓在手里。

还有，软件开发的方法论、甚至产品开发、项目管理都是这个领域里需要了解的。

另外，在某股技术“风”起的时候，要判断下它是为了解决什么问题而产生的，你的业务需不需要这种技术？我对曾经一起干活的小伙伴们说过，宁愿你闲着没事干，也不要乱学，瞎学。在这个领域，功利、目的性强一点好。不要为了学习而学习，也不要为了KPI去github上重复造轮子。这也是该领域的基本素养。



顿悟及其它

前几年我一直在系统层和靠近底层的领域里搞事情，有时候也会做一些应用层的开发。近来我在转入应用层开发的时候，花了一些时间研究应用层开发和系统层开发的一些区别：

1. 不得不说，底层和系统层是存在和需要个人英雄主义的。最常见的情况就是谁改bug最厉害谁就是英雄。所以，“深入理解的目标是什么”一文里提到的手撕鬼子一幕在这两个领域里是有可能存在的。
2. 应用层技术的难度在于工程化。一个好的想法在demo上实验成功了，要推广并大规模使用并创造真正的价

值一定是要工程化该想法的。而工程化的过程中，就需要业务、产品、前后端、测试等多个团队配合。“深入理解的目标是什么”提到的虚竹如果没有王语嫣、或者没有很好的团队支撑的话，是很难在这个地方发展起来的。

3. 另外，应用层开发如果找到现成的已经有工程化实践经历的开源库，我建议能使用的话还是使用它们吧。不过，要用好人家的开源库也不是一件容易的事情。因为人家所面临的问题肯定和你的问题不一样。

不知道你有没有这样的感觉。学习过程中，如果碰到一个问题，如果反反复复、长时间去思考它的话，总有一天会明白。这其中，有极少数的几次会让你有一种顿悟的感觉。到目前为止，我职业生涯里谈得上顿悟的时候应该不超过3次。

- 最早一次是反复回顾用MFC开发一个自定义树形控件总是有内存崩溃的bug从而顿悟到内存操作到底是在干什么。
- 另外一次是做DLNA的时候根据文档协议编写对应软件的顿悟——很多时候编程只不过是把想法翻译成计算机语言而已。而交流想法的形式可以是口述、规范文档。所以，想法是本。
- 最后一次谈得上顿悟的事情是在看完《信息简史》后。此后我在开发软件的时候仿佛能看到信息在流动。用户从界面上输入信息，流经网络，后台接收并处理信息，直到存储信息。甚至当我看到领导在耳提面命下属某个问题的具体处理细节时，我也深刻感受到他在“编程”——我们都是在处理信息罢了，领导对我们编程，我们对计算机编程。

最后

1. 深入理解Android系列有好几本书已经不再出版了，我在CSDN上提供全文电子版下载 https://blog.csdn.net/Innost/article/details/43342087
2. C++11的学习可参考CSDN博文 https://blog.csdn.net/Innost/article/details/52583732
3. “深入理解的目标是什么”见 https://mp.weixin.qq.com/s/hZrTlo3sSENtOQVbJ_h2Dg
4. 看到太多同学在追求技术上所付出的努力以及所碰到的困惑。我自己总结过很多有价值的东西，但我觉得**对具体知识的掌握并不是关键，更重要的是思考问题的方式和方法**——这也是我在本篇开头会提到我对考试成绩好坏的思考。另外，我还想学习一些新的东西。这些，我都想通过某种方式分享给感兴趣的朋友们。

在《深入理解Android JVM ART》这本厚达1000页书籍写完后，我给自己说，绝不要再写这样需要“操刀自宫"才能练成的书了。我应该把复杂的知识用简单明白的方式表达出来，然后大家再结合自己的认知和经历去体会、去利用这些知识就行。

最后的最后，我期望的结果不是朋友们从我的书、文章、博客后学会了什么知识，干成了什么，而应该是说，**神农，我可是踩在你的肩膀上得喔**。

神农和朋友们

的杂文集 长按识别二维码关注我们