###### Java虚拟机有多少个？？

在Android系统中，java虚拟机到底有多少个？？

例如：整个系统只有一个虚拟机？还是一个进程对应一个虚拟机。

解释：名词：Java 虚拟机，jvm实例。每个java程序都是JVM实例。

Java虚拟机对程序具有解释字节码和执行程序的能力。简单来说，jvm 实例就是一个解释器。它能解释并执行指令。与Linux 中的shell 脚本解释器一样。

###### 当我们运行一个java程序时，jvm 虚拟机是怎么工作的？？

当我们运行Run命令时，系统会先复制一个jvm 实例（可以看成是一个对象）。然后这个jvm实例会找到程序入口main方法，并执行里面的代码逻辑。

（具体jvm里面是怎么实现工作的我们就不知道了，不要试图是探索它，这是几十代人的结晶）

注：因此，当我们分析一个新的程序时，重要的是找到程序的入口。

###### jvm 实例 是否有运算能力？？

例如：对变量的自加 i++ ，自加运算时。在不离开jvm内存中，它是否能对变量自加？还是说必须依靠CPU的计算能力？？

答：目测来说，java虚拟机，是具有运算能力的。原因是：据说，jvm 具有自己的精简指令。

但是，我们根据数字电路知识知道，所有的软件都必须依赖硬件。只有计数器才具有计算的能力。所有的命令的执行，都是二进制的相加。而整个软件系统的核心和 计算器 之间的核心就是 指令集。

Jvm 也有指令集，因此，按道理来说，jvm 是具有运算能力的

###### 如何理解Android 中的AMS

我们知道AMS 可以启动 或 停止 一个进程。还可以进程的保活。

AMS 是怎么管理Activity、service组件的？？这些组件的对象是在哪个进程中的？？

首先，强调一点：AMS只是一个对象，java中一切皆对象。我们知道AMS只是System\_server进程中的一个stub 类型的对象。具体这个对象是怎么被线程调用的，我们不分析。

进程与进程之间是独立，而且进程是在底层的，就连jvm 都是用C++写的。更直接点， jvm 其实也是native 程序。也就是说AMS 凭什么管理其他进程？？？

答案是：管理不了，它只是提供了一个接口，从它是stub 对象就可以知道，它是利用了Binder通讯，至于它找上了谁，我们无法得知。同理，任何进程的四大组件，都是在自己的进程中生成对象的。它们只是在AMS中有一个影子，AMS通过这个影子，控制了它们的生命周期。

AMS能管理的是四大组件影子(代理)。

###### 内核 和 驱动 的关系

###### Android 系统应该关注那些进程？？？

Jvm虚拟机（包括Zygote进程和 Zygote 分裂出来的 system\_server进程）

Init 进程 以及和它启动的native 进程。

Kernel 层

注：kthread 是kernel层的鼻祖，init 是native层的鼻祖，zygote是java层的鼻祖。

###### Android 的主线程也用了Looper.loop为什么不会阻塞？？？

* 1. 首先要知道4个类，Message 、Loop、MessageQueue、Handler
  2. 不同线程，能使用同一个handler 的原因，所有对象都是在堆内存的，线程的交互，就转化为——》拿到对象的引用。