**Android面试之---谈谈你对Android NDK的理解.**

2010-03-10 21:11:25

标签：[Android](http://blog.51cto.com/tag-Android.html" \t "_blank) [面试](http://blog.51cto.com/tag-%E9%9D%A2%E8%AF%95.html) [NDK](http://blog.51cto.com/tag-NDK.html) [移动开发](http://blog.51cto.com/tag-%E7%A7%BB%E5%8A%A8%E5%BC%80%E5%8F%91.html) [休闲](http://blog.51cto.com/tag-%E4%BC%91%E9%97%B2.html)

1、前言   
 6月 26 日， Google Android 发布了 NDK ，引起了很多发人员的兴趣。 NDK 全称： Native Development Kit 。下载地址为： <http://developer.android.com/sdk/ndk/1.5_r1/index.html> 。

2、误解   
 新出生的事物，除了惊喜外，也会给我们带来一定的迷惑、误解。

2.1、误解一： NDK 发布之前， Android 不支持进行 C 开发   
 在Google 中搜索 “NDK” ，很多 “Android 终于可以使用 C++ 开发 ” 之类的标题，这是一种对 Android 平台编程方式的误解。其实， Android 平台从诞生起，就已经支持 C 、 C++ 开发。众所周知， Android 的 SDK 基于 Java 实现，这意味着基于 Android SDK 进行开发的第三方应用都必须使用 Java 语言。但这并不等同于 “ 第三方应用只能使用 Java” 。在 Android SDK 首次发布时， Google 就宣称其虚拟机 Dalvik 支持 JNI 编程方式，也就是第三方应用完全可以通过 JNI 调用自己的 C 动态库，即在 Android 平台上， “Java+C” 的编程方式是一直都可以实现的。

 当然这种误解的产生是有根源的：在Android SDK 文档里，找不到任何 JNI 方面的帮助。即使第三方应用开发者使用 JNI 完成了自己的 C 动态链接库（ so ）开发，但是 so 如何和应用程序一起打包成 apk 并发布？这里面也存在技术障碍。我曾经花了不少时间，安装交叉编译器创建 so ，并通过 asset （资源）方式，实现捆绑 so 发布。但这种方式只能属于取巧的方式，并非官方支持。所以，在 NDK 出来之前，我们将 “Java+C” 的开发模式称之为灰色模式，即官方既不声明 “ 支持这种方式 ” ，也不声明 “ 不支持这种方式 ” 。

2.2、误解二：有了 NDK ，我们可以使用纯 C 开发 Android 应用   
 Android SDK采用 Java 语言发布，把众多的 C 开发人员排除在第三方应用开发外（ 注意：我们所有讨论都是基于“ 第三方应用开发 ” ， Android 系统基于 Linux ，系统级别的开发肯定是支持 C 语言的。 ）。NDK 的发布，许多人会误以为，类似于 Symbian 、 WM ，在 Android 平台上终于可以使用纯 C 、 C++ 开发第三方应用了！其实不然， NDK 文档明确说明： it is not a good way 。因为 NDK 并没有提供各种系统事件处理支持，也没有提供应用程序生命周期维护。此外，在本次发布的 NDK 中，应用程序 UI 方面的 API 也没有提供。至少目前来说，使用纯 C 、 C++ 开发一个完整应用的条件还不完备。

3、NDK 是什么   
对NDK 进行了粗略的研究后，我对 “NDK 是什么 ” 的理解如下：

1、NDK 是一系列工具的集合。

NDK提供了一系列的工具，帮助开发者快速开发 C （或 C++ ）的动态库，并能自动将 so 和 java 应用一起打包成 apk 。这些工具对开发者的帮助是巨大的。   
NDK集成了交叉编译器，并提供了相应的 mk 文件隔离 CPU 、平台、 ABI 等差异，开发人员只需要简单修改 mk 文件（指出 “ 哪些文件需要编译 ” 、 “ 编译特性要求 ” 等），就可以创建出 so 。   
NDK可以自动地将 so 和 Java 应用一起打包，极大地减轻了开发人员的打包工作。   
2、NDK 提供了一份稳定、功能有限的 API 头文件声明。

 Google明确声明该 API 是稳定的，在后续所有版本中都稳定支持当前发布的 API 。从该版本的 NDK 中看出，这些 API 支持的功能非常有限，包含有： C 标准库（ libc ）、标准数学库（ libm ）、压缩库（ libz ）、 Log 库（ liblog ）。

4、NDK 带来什么   
1、NDK 的发布，使 “Java+C” 的开发方式终于转正，成为官方支持的开发方式。

使用NDK ，我们可以将要求高性能的应用逻辑使用 C 开发，从而提高应用程序的执行效率。   
使用NDK ，我们可以将需要保密的应用逻辑使用 C 开发。毕竟， Java 包都是可以反编译的。   
NDK促使专业 so 组件商的出现。（乐观猜想，要视乎 Android 用户的数量）   
2、NDK 将是 Android 平台支持 C 开发的开端。

 NDK提供了的开发工具集合，使开发人员可以便捷地开发、发布 C 组件。同时， Google承诺在 NDK 后续版本中提高 “ 可调式 ” 能力，即提供远程的 gdb 工具，使我们可以便捷地调试 C 源码。在支持 Android 平台 C 开发，我们能感觉到 Google 花费了很大精力，我们有理由憧憬 “C 组件支持 ” 只是 Google Android 平台上C 开发的开端。毕竟， C 程序员仍然是码农阵营中的绝对主力，将这部分人排除在 Android 应用开发之外，显然是不利于 Android 平台繁荣昌盛的。