**Android程序员常见的问题**

**31. 请解释下Android程序运行时权限与文件系统权限的区别。**

  Android程序执行需要读取到安全敏感项必需在androidmanifest.xml中声明相关权限请求, 打电话,访问网络,获取坐标,读写sd卡,读写联系人等..安装的时候会提示用户…

文件系统的权限是linux权限. 比如说sharedpreference里面的Context.Mode.private Context.Mode.world\_read\_able  Context.Mode\_world\_writeable

777自己同组其他

root 是危险的行为

**32. 系统上安装了多种浏览器，能否指定某浏览器访问指定页面？**

   找到对应的浏览器的意图,传递数据URI , 激活这个意图

      Intent intent = new Intent();

      intent.setClassName(packageName, className);

      Intent.setdata(uri);

**33. 对android主线程的运用和理解。**

   主ui线程不能执行耗时的操作,

**34. 对android虚拟机的理解，包括内存管理机制垃圾回收机制。**

虚拟机很小,空间很小,谈谈移动设备的虚拟机的大小限制 16M ,

谈谈加载图片的时候怎么处理大图片的,

垃圾回收,没有引用的对象,在某个时刻会被系统gc掉.

System.gc(); 不是立刻去执行垃圾回收.

手机电视. 视频点播. Onstop() 通知gc();

别的界面图片白板.

遇到一个不可复现的bug, 不解决 won't fix

**35.Framework工作方式及原理，Activity是如何生成一个view的，机制是什么。**

反射 , 配置文件

 可以讲下activity的源码,比如说每个activity里面都有window.callback和keyevent.callback,一些回调的接口或者函数吧.框架把activity创建出来就会调用里面的这些回调方法,会调用activity生命周期相关的方法.

setContentView();

普通的情况:

Activity创建一个view是通过 ondraw 画出来的, 画这个view之前呢,还会调用onmeasure方法来计算显示的大小.

Surfaceview 直接操作硬件  opengl .

图像要想被显示到界面上, 需要设备显卡, 显存.

写到显存.

**36.  android本身的一些限制，比如apk包大小限制，读取大文件时的时间限。**

 这个问题问的有问题,apk包大小限制不好说,

极品飞车有100M 还是能装到手机上,

世面google market 上大程序  主程序很小 5~10M    下载sdcard

15分钟之内申请退款

apk包,精简包, 素材存放在服务器. 游戏程序.

读大文件的时间限制应该是main线程里面的时间限制吧.5秒.

**37. 如何加载的音乐信息，如何改善其效率。**

Android提供mediascanner,mediaStore等接口, 音乐文件的信息都会存放到系统的数据库表中,可以通过content provider获取,

显示出来,改善效率,是个常见问题, 可以从以下几个方面作答,

分批加载数据, 延时加载数据, 合理使用缓存等...

Java虚拟机, 目标, 加快java程序执行速度

预先加载一些都会的使用的大的class的字节码, 提前加载.

时间换时间

空间换时间

**38.ListView如何提高其效率？**

 复用convertview, 历史的view对象

异步加载数据,  分页加载数据,onscallLinster();

Static class ViewHolder

使用静态的view对象避免创建过多的view.

把下载后的数据缓存到数据库里

客户端和服务器协同作战 .

Http 请求

If-modify-since : 检查数据是否更新  1.html 2001年  302 拿缓存

**39. 启动应用后，改变系统语言，应用的语言会改变么？**

  会

**40. 启动一个程序，可以主界面点击图标进入，也可以从一个程序中跳转过去，二者有什么区别？**

 区别是根据activity在manifest里面的配置,这个activity可能会放在不同的task栈里面

intent设置的flag  flag\_new\_task

**41.Android程序与Java程序的区别？**

 Android程序用android sdk开发,java程序用javasdk开发.

Android SDK引用了大部分的Java SDK，少数部分被AndroidSDK抛弃，比如说界面部分，java.awt  swing  package除了java.awt.font被引用外，其他都被抛弃，在Android平台开发中不能使用。

android sdk 添加工具jar httpclient ,pull  openGL

将Java 游戏或者j2me程序移植到Android平台的过程中，

Android SDK 与Java SDK的区别是很需要注意的地方。

1.6的不支持 httpUrlconning 获取 last-modified信息的处理

simpleDataFormat

**42.Android中Task任务栈的分配。**

假如有三个Activity A B C，A跳到B，然后B跳到C，现在需要从C跳到A如何传递数据而且使效率最高呢？

首先我们来看下Task的定义，Google是这样定义Task的：a task is what the user experiences as an "application."It's a group of related activities, arranged in a stack. A task is a stack ofactivities, not a class or an element in the manifest file. 这意思就是说Task实际上是一个Activity栈，通常用户感受的一个Application就是一个Task。从这个定义来看，Task跟Service或者其他Components是没有任何联系的，它只是针对Activity而言的。

Activity有不同的启动模式, 可以影响到task的分配

Task，简单的说，就是一组以栈的模式聚集在一起的Activity组件集合。它们有潜在的前后驱关联，新加入的Activity组件，位于栈顶，并仅有在栈顶的Activity，才会有机会与用户进行交互。而当栈顶的Activity完成使命退出的时候，Task会将其退栈，并让下一个将跑到栈顶的Activity来于用户面对面，直至栈中再无更多Activity，Task结束。

事件         Task栈（粗体为栈顶组件）

点开Email应用，进入收件箱（Activity A）       A

选中一封邮件，点击查看详情（Activity B）      AB

点击回复，开始写新邮件（Activity C）      ABC

写了几行字，点击选择联系人，进入选择联系人界面（Activity D）     ABCD

选择好了联系人，继续写邮件     ABC

写好邮件，发送完成，回到原始邮件         AB

点击返回，回到收件箱         A

退出Email程序      null

如上表所示，是一个实例。从用户从进入邮箱开始，到回复完成，退出应用整个过程的Task栈变化。这是一个标准的栈模式，对于大部分的状况，这样的Task模型，足以应付，但是，涉及到实际的性能、开销等问题，就会变得残酷许多。

比如，启动一个浏览器，在Android中是一个比较沉重的过程，它需要做很多初始化的工作，并且会有不小的内存开销。但与此同时，用浏览器打开一些内容，又是一般应用都会有的一个需求。设想一下，如果同时有十个运行着的应用（就会对应着是多个Task），都需要启动浏览器，这将是一个多么残酷的场面，十个Task栈都堆积着很雷同的浏览器Activity，

是多么华丽的一种浪费啊。

于是你会有这样一种设想，浏览器Activity，可不可以作为一个单独的Task而存在，不管是来自那个Task的请求，浏览器的Task，都不会归并过去。这样，虽然浏览器Activity本身需要维系的状态更多了，但整体的开销将大大的减少，这种舍小家为大家的行为，还是很值得歌颂的

standard", "singleTop","singleTask", "singleInstance"。

standard模式，是默认的也是标准的Task模式，在没有其他因素的影响下，使用此模式的Activity，会构造一个Activity的实例，加入到调用者的Task栈中去，对于使用频度一般开销一般什么都一般的Activity而言，standard模式无疑是最合适的，因为它逻辑简单条理清晰，所以是默认的选择。

而singleTop模式，基本上于standard一致，仅在请求的Activity正好位于栈顶时，有所区别。此时，配置成singleTop的Activity，不再会构造新的实例加入到Task栈中，而是将新来的Intent发送到栈顶Activity中，栈顶的Activity可以通过重载onNewIntent来处理新的Intent（当然，也可以无视...）。这个模式，降低了位于栈顶时的一些重复开销，更避免了一些奇异的行为（想象一下，如果在栈顶连续几个都是同样的Activity，再一级级退出的时候，这是怎么样的用户体验...），很适合一些会有更新的列表Activity展示。一个活生生的实例是，在Android默认提供的应用中，浏览器（Browser）的书签Activity（BrowserBookmarkPage），就用的是singleTop。

singleTask，和singleInstance，则都采取的另辟Task的蹊径。

标志为singleTask的Activity，最多仅有一个实例存在，并且，位于以它为根的Task中。所有对该Activity的请求，都会跳到该Activity的Task中展开进行。singleTask，很象概念中的单件模式，所有的修改都是基于一个实例，这通常用在构造成本很大，但切换成本较小的Activity中。最典型的例子，还是浏览器应用的主Activity（名为Browser...），它是展示当前tab，当前页面内容的窗口。它的构造成本大，但页面的切换还是较快的，于singleTask相配，还是挺天作之合的。

singleInstance显得更为极端一些。在大部分时候singleInstance与singleTask完全一致，唯一的不同在于，singleInstance的Activity，是它所在栈中仅有的一个Activity，如果涉及到的其他Activity，都移交到其他Task中进行。这使得singleInstance的Activity，像一座孤岛，彻底的黑盒，它不关注请求来自何方，也不计较后续由谁执行。在Android默认的各个应用中，很少有这样的Activity，在我个人的工程实践中，曾尝试在有道词典的快速取词Activity中采用过，

是因为我觉得快速取词入口足够方便（从notification中点选进入），并且会在各个场合使用，应该做得完全独立。

大的apk 拆成很多小的apk

  ●Activity的android:taskAffinity=""属性姻缘关系

1.配置后当启动这个activity时就先去找有没有activity的亲和力属性相同有就加入这个

         activity所在的任务中没有就新开任务

2.affinity起作用需要的条件而者具备一个:

                   1.intent包含FLAG\_ACTIVITY\_NEW\_TASK标记

                   2.activity元素启用了allowTaskReparenting属性.

**43. 在Android中，怎么节省内存的使用，怎么主动回收内存？**

  回收已经使用的资源,

  合理的使用缓存

合理设置变量的作用范围…  application 对象

//未来的某一段时间执行

System.gc();

**44. 不同工程中的方法是否可以相互调用？**

  可以,列举aidl访问远程服务的例子.

**45. 在Android中是如何实现判断区分电话的状态，去电，来电、未接来电？**

Day8 showAddressService.java

**46.dvm的进程和Linux的进程, 应用程序的进程是否为同一个概念**

Dvm的进程是dalivk虚拟机进程,每个android程序都运行在自己的进程里面,

每个android程序系统都会给他分配一个单独的liunx  uid(user id),

每个dvm都是linux里面的一个进程.所以说这两个进程是一个进程.

**47.sim卡的EF 文件有何作用**

  Sim卡相关的东西,没接触过

**48. 如何判断是否有SD卡？**

  配置文件中有sd卡的权限, 通过environment的静态方法,

if (Environment.getExternalStorageState().equals(Environment.MEDIA\_MOUNTED)){

**49. 嵌入式操作系统内存管理有哪几种，各有何特性。**

页式，段式，段页，等…

**50. 什么是嵌入式实时操作系统, Android 操作系统属于实时操作系统吗?**

实时操作系统是指当外界事件或数据产生时，能够接受并以足够快的速度予以处理，其处理的结果又能在规定的时间之内来控制生产过程或对处理系统作出快速响应，并控制所有实时任务协调一致运行的嵌入式操作系统。主要用于工业控制、军事设备、航空航天等领域对系统的响应时间有苛刻的要求，这就需要使用实时系统。又可分为软实时和硬实时两种，而android是基于linux内核的，因此属于软实时。

硬实时. 火星探测车.

软实时.

**51. 一条最长的短信息约占多少byte?**

中文70(包括标点)，英文160，160个字节这个说法不准确,

要跟手机制式运营商等信息有关.

做实验,看源码

ArrayList<String> msgs =sms.divideMessage(message);

        for (String msg : msgs) {

           sms.sendTextMessage(phoneNumber, null, msg, pi, null);

}

**52.Linux中跨进程通信的几种方式。**

linux编程

# 管道( pipe )：管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。

# 有名管道 (named pipe) ：有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。

# 信号量( semophore ) ：信号量是一个计数器，可以用来控制多个进程对共享资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源时，其他进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。

# 消息队列( message queue )：消息队列是由消息的链表，存放在内核中并由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。

# 信号 ( sinal ) ：信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。

# 共享内存( shared memory )：共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号两，配合使用，来实现进程间的同步和通信。

# 套接字( socket ) ：套解口也是一种进程间通信机制，与其他通信机制不同的是，它可用于不同及其间的进程通信。

**53. 谈谈对Android NDK的理解。**

native develop kit   只是一个交叉编译的工具  .so

         1.什么时候用ndk,  实时性要求高,游戏,图形渲染,

opencv (人脸识别) , ffmpeg ,rmvb  mp5 avi 高清解码. ffmpeg,opencore.

2.为什么用ndk,ndk的优点 ,缺点 ,

我们项目中那些地方用到了ndk,

**54. 谈谈Android的优点和不足之处。**

1、开放性,开源ophone  阿里云( 完全兼容android)

2、挣脱运营商束缚

3、丰富的硬件选择 mtk android

4、不受任何限制的开发商

5、无缝结合的Google应用

缺点也有5处：

1、安全问题、隐私问题

2、卖手机的不是最大运营商

3、运营商对Android手机仍然有影响

4、山寨化严重

5、过分依赖开发商，缺乏标准配置版本过多

**55.Android系统中GC什么情况下会出现内存泄露呢？**

视频编解码/内存泄露

检测内存泄露   工具

导致内存泄漏主要的原因是，先前申请了内存空间而忘记了释放。如果程序中存在对无用对象的引用，那么这些对象就会驻留内存，消耗内存，因为无法让垃圾回收器GC验证这些对象是否不再需要。如果存在对象的引用，这个对象就被定义为"有效的活动"，同时不会被释放。要确定对象所占内存将被回收，我们就要务必确认该对象不再会被使用。典型的做法就是把对象数据成员设为null或者从集合中移除该对象。但当局部变量不需要时，不需明显的设为null，因为一个方法执行完毕时，这些引用会自动被清理。

Java带垃圾回收的机制,为什么还会内存泄露呢?

Vector v = new Vector(10);

 for(int i = 1; i < 100; i++)      {

　Object o = new Object();

v.add(o);

o = null;

}//此时，所有的Object对象都没有被释放，因为变量v引用这些对象。

Java 内存泄露的根本原因就是保存了不可能再被访问的变量类型的引用

**56.Android UI中的View如何刷新。**

在主线程中  拿到view调用Invalide()方法,查看画画板里面更新imageview的方法

在子线程里面可以通过postInvalide()方法;

**57. 简单描述下Android 数字签名。**

Android 数字签名

      在Android系统中，所有安装到系统的应用程序都必有一个数字证书，此数字证书用于标识应用程序的作者和在应用程序之间建立信任关系

Android系统要求每一个安装进系统的应用程序都是经过数字证书签名的，数字证书的私钥则保存在程序开发者的手中。Android将数字证书用来标识应用程序的作者和在应用程序之间建立信任关系，不是用来决定最终用户可以安装哪些应用程序。

这个数字证书并不需要权威的数字证书签名机构认证(CA)，它只是用来让应用程序包自我认证的。

同一个开发者的多个程序尽可能使用同一个数字证书，这可以带来以下好处。

(1)有利于程序升级，当新版程序和旧版程序的数字证书相同时，Android系统才会认为这两个程序是同一个程序的不同版本。如果新版程序和旧版程序的数字证书不相同，则Android系统认为他们是不同的程序，并产生冲突，会要求新程序更改包名。

(2)有利于程序的模块化设计和开发。Android系统允许拥有同一个数字签名的程序运行在一个进程中，Android程序会将他们视为同一个程序。所以开发者可以将自己的程序分模块开发，而用户只需要在需要的时候下载适当的模块。

在签名时，需要考虑数字证书的有效期：

(1)数字证书的有效期要包含程序的预计生命周期，一旦数字证书失效，持有改数字证书的程序将不能正常升级。

(2)如果多个程序使用同一个数字证书，则该数字证书的有效期要包含所有程序的预计生命周期。

(3)Android Market强制要求所有应用程序数字证书的有效期要持续到2033年10月22日以后。

Android数字证书包含以下几个要点：

 (1)所有的应用程序都必须有数字证书，Android系统不会安装一个没有数字证书的应用程序

 (2)Android程序包使用的数字证书可以是自签名的，不需要一个权威的数字证书机构签名认证

 (3)如果要正式发布一个Android，必须使用一个合适的私钥生成的数字证书来给程序签名，而不能使用adt插件或者ant工具生成的调试证书来发布。

 (4)数字证书都是有有效期的，Android只是在应用程序安装的时候才会检查证书的有效期。如果程序已经安装在系统中，即使证书过期也不会影响程序的正常功能。