Java 语言进阶与 Android 技术相关面试题

线程池原理与 Android 相关问题

50.线程的详细概念

线程是程序执行中一个单一的顺序控制流程，是程序执行流的最小单元，是处理器调度和分派的基本单位，一个进程可以有多个线程，各个线程之间共享程序的内存空间（也就是所在进程的内存空间），线程是一个进程中代码的不同执行路线，线程上下文切换比进程上下文切换要快得多。

51.Android 中的性能优化相关问题

由于手机硬件的限制，在 Android 手机中过多的使用内存，会容易导致 oom（out of memory 内存溢出），过多的使用 CPU 资源，会导致手机卡顿，甚至导致 ANR(Application Not Responding 应用程序无响应)，Android 主要从以下几部分对此进行优化：

1. 布局优化，使用 herarchyviewer(视图层次窗口)工具
2. 删除无用的空间和层级；
3. 选择性能较低的 viewgroup，比如在可以选择 RelativeLayout 也可以使用 LinearLayout 的情况下，优先使用 LinearLayout，因为相对来说 RelativeLayout 功能较为复杂，会占用更多的 CPU 资源；
4. 使用标签<include/>重用布局、<merge/>减少层级、<viewStub/>进行预加载（用的时候才加载）；

绘制优化：指 view 在 ondraw 方法中避免大量的耗时的操作，由于 onDraw 方法可能会被频繁的调用；ondraw 方法中不要创建新的局部变量，

ondraw 方法被频繁的调用，很容易引起 GC；ondraw 方法不要做耗时的操作；

线程优化：使用线程池来管理和复用线程，避免程序中出现大量的 Thread，同时可以控制线的并发数，避免相互抢占资源，而导致线程阻塞；

52. 内存泄漏的相关原因

如果一个无用的对象（不需要再使用的对象）仍然被其他对象持有引用，造成该对象无法被系统回收，以致该对象在堆中所占有的内存单元无法被释放而造成内存空间浪费，这种情况就是内存泄漏，常见的内存泄露的场景有：

单例模式：因为单例的静态特性使得他的生命周期同应用的生命周期一样长，如果一个对象已经没有用处了，但是单例还持有他的引用，那么在某个应用程序的生命周期他都不能正常回收，从而导致内存泄漏；

静态变量导致内存泄漏：静态变量存储在方法区，他的生命周期从类加载开始，到整个进程结束，一旦静态变量初始化，他所持有的引用只有等到进程结束才会释放；

非静态内部类导致内存泄漏：非静态内部类（包括匿名内部类）默认就会持有外部类的引用，当非静态内部类对象的生命周期比外部对象的生命周期长时，就会导致内存泄漏，通常在 Android 开发中，如果要使用内部类，但又要规避内存泄漏，一般会采用静态内部类+弱引用的方式；

未取消注册或回掉导致内存泄漏：比如在 Activity 中注册广播，如果在 Activity 销毁后不取消注册，那么这个广播会一直存在于系统中，同非静态内

部类一样持有 Activity 引用，导致内存泄漏，因此在注册广播后一定要取消注册；

53.通过 Handler 在线程间通信的原理

Android 中主线程是不能进行耗时操作的，子线程不能进行更新 UI，所以就有了 Handler，其作用就是实现线程之间的通信。

Handler 在整个线程通信的过程中，主要有四个对象，Handler、Message、MessageQueue、Looper 等，当应用创建时，就会在主线程中创建 Handler 对象，将要传递的信息保存到 Message 中，Handler 通过调用 sendMessage()方法将 Message 发送到 MessageQueue 中，Looper 对象不断调用 Loop( )方法从 MessageQueue 中取出 Message 交给 handler 进行处理，从而实现线程之间的通信。

54.Android 中动画的类型：

帧动画：通过指定每一帧图片和播放的时间，有序地进行播放而形成动画效果；

补间动画：通过指定的 view 的初始状态、变化时间、方式等，通过一系列的算法去进行图形变换从而形成动画效果，主要有 Alpha、Scale、Translate、Rotate 四种效果；

属性动画：在 Android3.0 开始支持，通过不断改变 view 的属性，不断的重绘而形成动画效果；

55.理解 Activity、View、Window 三者之间的关系

使用一个比喻形容他们的关系，Activity 像一个工匠（控制单元），Window 像窗户（承载模型）、View 像窗花（显示视图）、LayoutInflater 像剪刀、XML 配置像窗花图纸：

Activity 构造的时候会初始化一个 Window，准确的说应该是 PhoneWindow；

这个 PhoneWindow 有一个”ViewRoot “，这个”ViewRoot”是一个 View 或者说是 ViewGroup，是最初的根视图；

“ViewRoot”是通过 addView 方法来添

这些 view 的事件的监听，是由 WindowManagerService 来接收消息，并且回调 Activity 函数，比如 onClickListener、onKeyDown 等；

加一个个 View 的，比如 TextView、Button 等；

56.Android 中 Context 详解：

Context 是一个抽象基类，译为上下文，也可理解为环境，是用户操作操作系统的一个过程，这个对象描述的是一个应用程序环境的全局信息，通过它可以访问应用程序的资源和相关的权限，简单的说 Context 负责 Activity、Service、Intent、资源、Package 和权限等操作。Context 层次如下图：

第一层：Context 抽象类；

第二层：一个 ContextImpl 的实现类，里面拥有一个 PackageInfo 类，PackageInfo 类是关于整个包信息的类，一个 ContextWraper 是一个

Context 的包装类，里面有一个 ContextImpl 类的实例，通过整个实例去调用 ContextImpl 里面的方法；

第三层：Service 和 Application 直接继承 ContextWrapper，但是 Activity 需要引入主题，所以有了 ContextThemeImpl 类；

总结：在使用 Context 对象获取静态资源，创建单例对象或者静态方法时，多考虑 Context 的生命周期，不要使用 Activity 的 Context，要使用生命周期较长的 Application 的 Context 对象，但并不是所有的情况都使用 Application 的 Context，在创建 Dialog、view 等控件的时候，必须使用 Activity 的 Context 对象。

57. Java 中 double 和 float 类型的区别

float 是单精度类型，精度是 8 位有效数字，取值范围是 10 的-38 次方到 10 的 38 次方，float 占用 4 个字节的存储空间；

double 是双精度类型，精度是 10 的-308 次方到 10 的 308 次方，double 类型占用 8 个字节的存储空间；

默认情况下都用 double 来表示，所以如果要用 float，则应在数字对象后加字符 f。

58. Android 常用的数据存储方式（4 种）

使用 SharedPreference 存储：保存基于 xml 文件存储的 key-value 键值对数据，通常用来存储一些简单的配置信息；

文件存储方式：Context 提供了两个方法来打开数据文件的文件 IO 流；

SQLite 存储数据：SQLite 是轻量级的嵌入式数据库引擎，支持 SQL 语言；

网络存储数据：通过网络存储数据；

59.ANR 的了解及优化

ANR 全名 Appication NotResponding ，也就是“应用程序无反应”，当操作在一段时间内无法得到系统回应时就会出现 ANR 错误，造成 ANR 的主要原因是由于线程的任务在规定的时间内没有完成造成的；

60.Android 垃圾回收机制和程序优化 System.gc( )

垃圾收集算法的核心思想：对虚拟机的可用内存空间，即堆空间中的对象进行识别，如果对象正在被引用，则称其为存活对象，反之，如果对象不再被引用，则为垃圾对象，可以回收其占据的空间，用于再分配。

JVM 进行次 GC 的频率很高，但因为这种 GC 占用的时间极短，所以对系统产生的影响不大，但主 GC 对系统的影响很明显，触发主 GC 的条件主要有下：

当应用程序空闲时，即没有应用线程在运行时，GC 会被调用，因为 GC 在优先级别最低的线程中进行，所以当应用繁忙时，GC 就不会被调用；

Java 堆内存不足时，主 GC 会被调用，当应用线程正在运行，并在运行过程中创建新对象时，若这时内存空间不足，JVM 会强制调用主 GC，以便回收内存用于新的分配，若 GC 一次之后仍无法满足，则会继续进行两次，若仍无法满足，则会报 OOM 错误；

System.gc( )函数的作用只是提醒虚拟机希望进行一次垃圾回收，但并不一定保证会进行，具体什么时候进行取决于虚拟机；

61.Android 平台的优势和不足

Android 平台手机 5 大优势:

开放性：Android 平台首先就是其开放性，开发的平台允许任何移动终端厂商加入到 Android 联盟中来。显著的开放性可以使其拥有更多的开发者；

挣脱运营商的束缚：在过去很长的一段时间，手机应用往往受到运营商制约，使用什么功能接入什么网络，几乎都受到运营商的控制，而 Android 用户可以更加方便地连接网络，运营商的制约减少；

丰富的硬件选择：由于 Android 的开放性，众多的厂商会推出千奇百怪，功能特色各具的多种产品。功能上的差异和特色，却不会影响到数据同步、甚至软件的兼容；

开发商不受任何限制：Android 平台提供给第三方开发商一个十分宽泛、自由的环境，不会受到各种条条框框的阻扰；

无缝结合的 Google 应用： Android 平台手机将无缝结合这些优秀的 Google 服务如地图、邮件、搜索等；

Android 平台手机几大不足：

安全和隐私：由于手机与互联网的紧密联系，个人隐私很难得到保守。除了上网过程中经意或不经意留下的个人足迹，Google 这个巨人也时时站在你的身后，洞穿一切；

过分依赖开发商缺少标准配置：在 Android 平台中，由于其开放性，软件更多依赖第三方厂商，比如 Android 系统的 SDK 中就没有内置音乐播放器，全部依赖第三方开发，缺少了产品的统一性；

同类机型用户很少：在不少手机论坛都会有针对某一型号的子论坛，对一款手机的使用心得交流，并分享软件资源。而对于 Android 平台手机，由于厂商丰富，产品类型多样，这样使用同一款机型的用户越来越少，缺少统一机型的程序强化。