自定义控件（尚学谷）

# 1.Android手机系统

我们要知道为什么造成OOM，务必来看看Android系统的一些知识

我们使用Android手机的同学应该都知道，手机越用越卡，而且时不时就会出现内存不足的情况，为什么呢？原因有以下：

1，我们知道Android是由Java语言编写，那么熟悉Java的同学应该知道，Java是运行在虚拟机上的，而不是像苹果直接运行在系统上，虚拟机不仅非常耗性能，更是垃圾场，随着时间越来越久，垃圾越来越多，内存就会越来越不足

2，熟悉Android的同学知道，我们app是可以长时间运行在后台的，不像苹果，当用户不用这个app的时候，过了一会系统就杀死了这个app了，既然Android的app是可以长时间运行在后台，那么就会不断的占用内存和进程，特别是些大型的app，更是消耗很大

3，现在有很多app没有按照Android的规格去开发，没有在意app的性能优化，导致app越来越大，消耗的性能也越来越多，在按照上述第2点，结果可想而知

4，熟悉Java的同学应该知道，Java有非常强大的GC机制，可以回收一些没必要的对象来保证内存的分配，但是，Java的虚拟机却不能做到随用随放，也就是说，当我们内存不足的情况下，它才可能回收，但是却不会预先回收垃圾来为我们的app分配足够大的内存，在按照上述第1点，也很容易知道结果

5，手机配置不高（除非你换iphone，否则还是逃不过Android系统机制）

上述就是Android系统的一些特性，但是还不够，我们从进程机制来分析

# 2.进程机制和内存机制

我们知道了app是运行在虚拟机上，Android为每一个app都单独分配了一个虚拟机，也就是说每个app都有自己的进程，每个进程都有自己的内存空间，这样做的好处就是当我们当前的app出现问题的时候，系统仅仅杀死当前进程，不会导致其他的app受到牵连，回收之后释放出内存给其他app使用，进程从高到底分为5种：

前台进程

正在与用户交互的进程，通俗来讲就是你当前使用app的进程

可见进程

可以被用户看到，但是没有和用户交互，例如一个activity以对话框的形式覆盖在当前activity上面，当前activity可以被用户看到，但是不和用户交互

服务进程

这个相信大家都熟悉，也就是我们常说的service，能够运行在后台，常见的有音乐类的app

后台进程

注意，这个后台进程不要和服务进程搞混了，它的意思是说当前app在后台运行，例如我启动了app，然后点击home返回到桌面，那么这个app就会被切回到后台进程

空进程

空进程指的是在这些进程内部，没有任何东西在运行。保留这种进程的的唯一目的是用作缓存，以缩短该应用下次在其中运行组件所需的启动时间

当我们内存不足资源紧缺的时候，会从低到高去分别释放进程，从而使app有足够的内存运行，Android设备出厂以后，java虚拟机对单个应用的最大内存分配就确定下来了，超出这个值就会OOM，不同的设备会有不同的内存空间，所以啊，买Android手机的朋友，最好选配置好一点的，不然卡死你，并且如果多app运行在后台的话，也会增加流量消耗

其实Android的这种机制有利也有弊，弊我想大家都知道了，那利是什么呢？不知道大家发现一个问题没，上述说了Android和苹果的机制不同，苹果是会杀死进程，也就是说我重新打开这个app的时候，是需要重新去链接网络获取信息的，这样对有些用户来说也挺烦的，如果我在玩游戏，突然来消息了，我切换出去久了再切换回来，我又重新来到登录界面了，我就挺烦这个的，但是Android如果内存足够，就不需要重新登录了，另外，Android的app从后台切换到前台时，由于系统没有杀死app，界面是不需要重新读取，启动速度会加快，也挺人性化的

内存泄漏

我们了解了内存机制之后，我们来看看内存泄漏，内存泄漏通俗一点讲就是应该被释放的资源，因为某些原因而没有被释放掉，为什么会这样呢？我们需要简单了解下Java的垃圾回收机制（GC）

GC机制

我们在开发过程中，少不了对象的创建，我们所创建的对象都是存储在Java虚拟机的内存之中，我们并不用去关心对象的释放，因为这一切都是由GC帮我们去完成，但是，这是有条件的.

说道GC的垃圾回收机制，需要了解Java引用，强引用，软引用，弱引用，虚引用

1，无用对象

2，内存不足

无用对象简单概括就是，我们创建出一个对象的时候，但并没有对这个对象赋值或者说没用到这个对象，另外一种，这个对象已经使用完了，不会再被其他程序调用，举个例子，我们使用I/O流的时候，我们使用完了之后都需要释放掉，因为我们使用完之后已经没用其他程序在去调用了，那么这个对象应当属于无用对象

当内存空间出现紧缺，Java会回收一些进程来确保内存空间，我们知道我们每一个app都是有单独的进程，回收进程的时候，也就是将我们的对象资源进行回收，也就是将我们无用对象进行回收

但是，如果在一个进程当中，对象已经用完了，但是因为某些原因被使用，GC回认为这个对象可能还需要被其他程序调用，但是实际上没有程序再去调用它，那么就会导致这个对象一直在占用着内存，另外一种，如果两个无用对象相互牵引着，也会导致GC无法回收，内存一直会被占用，就会导致了**内存泄漏**，举几个例子：

1，广播在Activity销毁的时候没有释放掉：因为广播持有Activity对象，当Activity销毁时，GC去回收它的对象，但是这时候，我们没有释放广播，所以它没办法去回收，就会导致内存泄漏

2，Handler做耗时操作：如果我们在Handler中做比较耗时的操作，在这个操作还没有完成的时候，我们销毁Activity，Handler应当是个无用对象，但是它却赖着不走，GC也无法强制把它带走，也会造成内存泄漏，经常handle还会操作UI控件

3，非静态内部类的静态调用：我们在Activity中定义一个内部类，然后以静态的方式去引用它，我们知道静态的特性是唯一性，也就是说这个被定义为静态的对象的生命周期是和app生命周期一样长，但是我们的内部类是非静态的，是依附在当前Activity的生命周期上的，当Activity被回收后，因为内部类的对象持有Activity对象，而且它的生命周期是app的生命周期，所以会导致Activity没办法被正常回收

.。。。

还有很多其他的例子，就不一一列举了，现在大概了解什么是内存泄漏以及内存泄漏造成的原因了吧

内存溢出

内存泄漏会造成内存无法被释放，系统无法为其他一些app开辟更多的内存空间，当我们内存泄漏越来越大的时候，已经超出我们正常内存空间的时候，就是我们常说的**内存溢出**

相信大家对内存泄漏和内存溢出的危害都有一定的了解，它可以让我们手机越来越卡，严重的会造成系统崩溃，举个很通俗的例子，我们往袋子中注入水，当越来越多的时候，水会溢出来，并且严重的时候袋子会裂开，所以，不要小看内存溢出，小的内存溢出可能我们感觉不到，但是积少成多，终究会发生问题

这也就是为什么我们手机系统会越来越卡，也是我们为什么Android系统会崩溃的原因，我们来简单总结下：

1，app都有自己单独的进程，每台设备在出厂的时候都会设定了我们单个app所占用的内存

2，app运行在后台时，当系统内存不足的时候，会根据进程的级别高低去回收进程

3，当我们对象资源无法被回收的时候，会造成内存泄漏，如果app的内存泄漏过多，超过内存空间，就会造成内存溢出

4，内存泄漏和内存溢出会造成手机系统越来越卡，严重的会导致系统崩溃

5，在实际开发之中，我们要严格按照Android的规范来编写程序，避免内存泄漏内存溢出等问题发生

作者：ReturnYHH  
链接：https://www.jianshu.com/p/49770f581690  
來源：简书  
简书著作权归作者所有，任何形式的转载都请联系作者获得授权并注明出处。