未回答起的问题 :

1. AsyncTask, 如何规避在调用Cancel()的时候取消doIngBackground() 内部的操作;

* Activity生命周期结束，AsyncTask会一直执行doInBackground()方法直到方法执行结束
* 如果cancel(boolean)调用了,doInBackground()仍然会执行到方法结束，只是不会去调用onPostExecute()方法
  + 解决这个问题的话， 我觉得应该是加个weakreference引用到AsyncTask
* 内存泄露，AsyncTask的生命周期可能比Activity的长
* 在AsyncTask全部执行完毕之后，进程中还是会常驻corePoolSize个线程
* AsyncTask串行和并行的问题
  + 默认是串行，提供了并行函数。
  + 如果需要并行， 需要处理好doInBackground()访问公共资源同步的问题

1. IntentService 生命周期, 以及IntentService结束时候, 是否会关闭线程

* Service存在的问题 :
  + Service不会专门启动一个单独的进程，它与所在的应用位于同一个进程中
  + Service不是一个线程，因此不应该在Service中直接处理耗时操作的任务
* IntentService :
  + IntentService会创建单独的worker线程来处理所有的Intent请求
  + IntentService会创建单独的worker线程来处理onHandlerIntent()方法实现的代码，开发者无需要处理多线程问题
  + 当所有请求处理完成后，IntentService会自动停止，开发者无须调用stopSelf(startId)方法来尝试停止该Service
    - 这里stopSelf(startId)和stopSelf()区别：
      * stopSelf()会立刻停止服务，这个时候还有可能有其他消息未处理；
      * stopSelf(int startId)会等待所有的消息都处理完毕后才会终止服务。一般来说，stopSelf(int startId)会在停止之前判断最近启动服务的次数是否和startId相等，相等则停止，不等就等待。
  + 生命周期 ：
    - onCreate -> onStartCommand -> onStart -> onHandleIntent

1. Activity启动流程 , (源码复述)
2. HashMap相关问题， ConcurrentHashMap并没有问
3. 设计一个方案 : 显示当前app的activity是否属于前台可见 , 不能使用ApplicationCallback回调
4. HashMap如何减少hash碰撞