2016/03/26

**1 关于android AsyncTask 限制以及ThreadPoolExecutor需注意**

AsyncTask 只能用来执行一些短暂的任务

/\*\*  
 \* NetAsyncTask 只能执行一些短暂的任务，  
 \*  
 \* 注意：对于长时间执行的任务，请不要放入！！！！！！  
 \*  
 \* 否则，一旦长时间执行任务占据了coresize个线程，那么后面进来的任务就会放入  
 \* sPoolWorkQueue（大小128）等待coresize线程执行完再执行（在这个过程中任务一直被 \* 阻塞），  
 \* 或者直到队列被放满，这个时候会新建maxSize-coresize个线程来缓解迟滞现象，  
 \* 但是一旦这maxSize线程也被占据没有恢复空闲，那么在sPoolWorkQueue满了，  
 \* 并且已经有maxSize个在执行，接下来放入的任务将被拒绝，  
 \* 拒绝策略使用的是设置的RejectedExecutionHandler  
 \*/

**2 关于易销毁对象(可以说非单例，静态对象都是易销毁) 内置线程需注意**

易销毁对象，例如android里面的View, Activity等里面如果使用了内部线程类作为成员变量，或者匿名内部对象。

请注意：因为内部类，匿名内部类对象会默认放入外部类的强引用，所以如果外部类的生命周期短，而内部类的生命周期更长（例如这个内部类是一个一直执行的线程类），那么内部类持有的强引用将导致外部类无法释放。

解决方式是：将这个生命周期比较长的内部类，写成静态的内部类，并且持有外部类的弱引用。这个时候请注意，访问外部类的时候请判断外部类是否存在。

同时请注意，如果你的线程结束信号是以外部类是否被释放作为信号，那么这个线程的生命长短与GC释放这个对象的时间相关，如果GC不释放，线程一直会在执行。。所以，尽量不要这么做。

对于View可以使用重载函数onDetachedFromWindow来取得退出标志