**进程与线程**

参考：http://blog.csdn.net/yaosiming2011/article/details/44280797

http://www.cnblogs.com/tiankong101/p/4229584.html

<http://www.cnblogs.com/lmule/archive/2010/08/18/1802774.html>

<https://www.jianshu.com/p/354066717f78>

<https://www.cnblogs.com/CarpenterLee/p/9558026.html>

1. 线程与进程
2. 进程：具有一定独立功能的程序关于某个数据集合上的一次运行活动，是系统进行资源分配和调度的最小单元。进程在执行过程中拥有独立的内存单元。
3. 线程：进程的一个实体,是CPU调度和分派的最小单元，比进程更小的能独立运行的基本单位。进程在执行过程中拥有独立的内存单元，线程只拥有一点在运行中必不可少的系统资源(如程序计数器,一组寄存器和栈)，但是它可与同属一个进程的其他的线程共享进程所拥有的全部资源。每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。
4. 进程与线程的关系： 一个进程可以有多个线程（通过CPU调度，在每个时间片中只有一个线程执行），同一进程中的所有线程共享进程的全部资源。
5. 进程与线程的区别：
6. 每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行
7. 多线程的意义在于一个应用程序中，有多个执行部分可以同时执行
8. 线程执行开销小，但不利于资源的管理和保护；而进程正相反。
9. 进程与线程的同步
10. 进程：无名管道、有名管道、信号、信号量、共享内存、消息队列；
11. 线程：互斥量、读写锁、自旋锁、线程信号、条件变量。
12. 线程

参考：<https://blog.csdn.net/yb223731/article/details/94560006>

<https://www.jianshu.com/p/f65ea68a4a7f>

1. Join()方法挂起当前线程，直到调用Join()方法的线程执行完毕。
2. 多线程的优缺点

参考：<https://www.jianshu.com/p/959cf355b574>

<https://blog.csdn.net/lingyiwin/article/details/78311458>

1. 优点
2. 充分利用系统资源
3. 程序快速响应
4. 缺点
5. 频繁切换上下文：每次切换时，需要保存当前的状态起来，以便能够进行恢复先前状态，而切换上下文非常损耗性能；
6. 临界区共享数据线程安全问题：多线程访问临界区共享数据时，需要使用同步机制保证线程安全，且只有得到锁的线程可以运行，其他线程需要等待锁释放。不当的线程同步容易造成死锁。
7. 线程的实现
8. 继承Thread类
9. 实现Runnable接口
10. AsyncTask
11. FutureTask

参考：<https://blog.csdn.net/qq_39654841/article/details/90631795>

<https://www.cnblogs.com/linghu-java/p/8991824.html>

1. HandlerThread
2. ThreadPoolExecutor线程池

参考：<https://www.cnblogs.com/rinack/p/9888717.html>

1. 若正在运行的线程小于核心线程数，则创建一个核心线程；
2. 若正在运行的线程数大于等于核心线程数，则将任务加入任务队列；
3. 若加入任务队列失败，则需要创建一个非核心线程；
4. 若线程总数大于最大线程数，则创建线程失败，进入拒绝策略（可自定义）。
5. 线程同步
6. volatile仅能使用在变量级别,  
   synchronized则可以使用在变量,方法.
7. volatile仅能实现变量的修改可见性,但不具备原子特性，可以多个线程同时访问。

synchronized则可以保证变量的修改可见性和原子性，同一时刻只能有一个线程访问。

1. volatile不会造成线程的阻塞  
   synchronized可能会造成线程的阻塞.
2. volatile标记的变量不会被编译器优化，被volatile修饰的变量的读写操作都必须在内存中进行  
   synchronized标记的变量可以被编译器优化.
3. Synchronized和Lock的区别：

参考：<https://www.cnblogs.com/handsomeye/p/5999362.html>

1. Lock可以设置锁等待的时间、可以响应中断、允许多个线程进行读操作和告诉线程是否成功获得锁，synchronized则不能；
2. Lock不是Java语言内置的，它是接口，通过实现这个接口的类来实现同步访问；而synchronized是Java语言的关键字，属于内置特性。
3. Lock需要用户手动释放锁（即便发生异常也需要手动释放锁），而synchronized由系统自动释放；