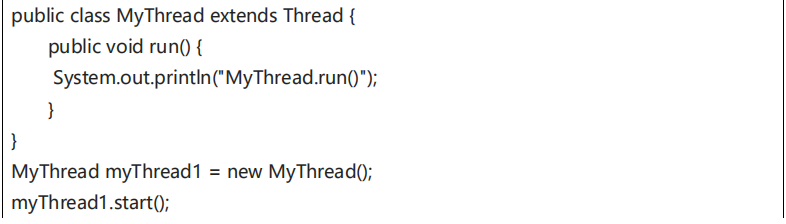
### JAVA多线程并发

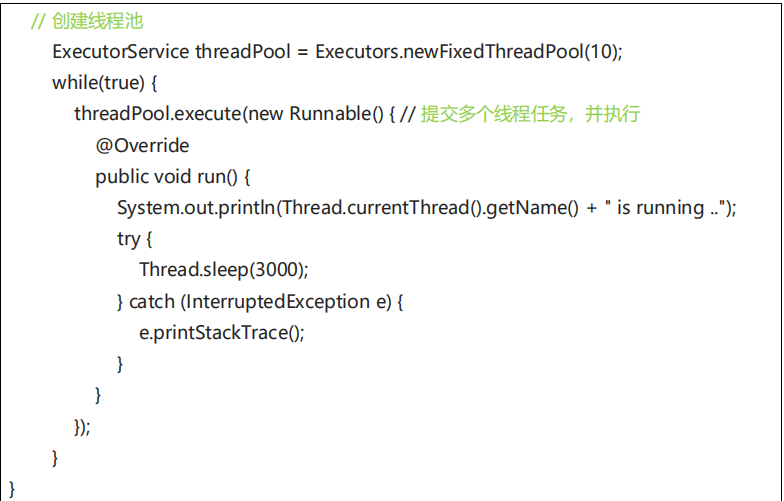
1. **Java线程实现/创建方式**
2. .继承Thread类



(2).实现Runnable接口，注意在启动此线程的时候需要先实例一个新线程并传入自己的

线程实例.

(3).基于线程池的方式



1. **4种线程池**

(1).CachedThreadPool创建一个可根据需要创建新线程的线程池，但是在以前构造的线程

可用时将重用他们，对于执行很多短期异步任务的程序而言，这种线程池可提高程序性能。

调用excute将重用以前构造的线程(如果线程可用),若无可用线程，则创建一个新线程并

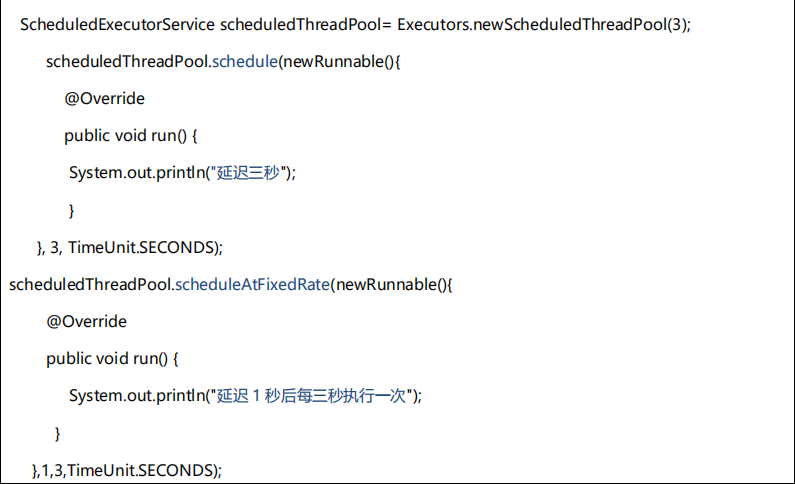
添加到池中，终止并从缓存中移除那些已经60秒未被使用的线程，因此长时间保持空闲

的线程池不会使用任何资源。

(2).FixedThreadPool创建一个可重用固定线程数的线程池，以共享的无界队列来运行这些线

程，在某个线程被显式关闭之前，池中的线程会一直存在。

(3).ScheduledThreadPool创建一个线程池，它可安排再给定延迟后运行命令或者定期执行。



(4).SingleThreadExecutor

Executors.newSingleThreadExecutor()返回一个线程池（这个线程池只有一个线程）,这个线程

池可以在线程死后（或发生异常时）重新启动一个线程来替代原来的线程继续执行下去！

3.**线程生命周期(状态)**

当线程被创建并启动以后，它既不是一启动就进入执行状态，也不是一直处于执行状态。在线程的生命周期中，它要经过新建(New)、就绪(Runnable)、运行(Running)、阻塞(Blocked)、

和死亡(Dead)5种状态，尤其当线程启动后CPU需要在多线程间切换，于是线程也会多次在

运行和阻塞之间切换。

**新建状态(New):**

当程序使用New关键字创建一个线程之后，该线程就处于新建状态，此时仅由JVM为

其分配内存，并初始化其成员变量的值；

**就绪状态(Runnable):**

当线程对象调用了start()方法之后，该线程就处于就绪状态，Java虚拟机会为其创建方

法调用栈和程序计数器，等待调度运行；

**运行状态(Running):**

如果处于就绪状态的线程获得了CPU，开始执行run()方法的线程执行体，则该线程处

于运行状态；

**阻塞状态(Blocked):**

阻塞状态是指线程因为某种原因放弃了CPU的使用权，暂时停止运行，直到再次进入

运行状态，阻塞情况分三种：

等待阻塞、同步阻塞、其他阻塞；

**死亡状态(Dead):**

线程会以下面三种方式结束，结束后就是死亡状态：

正常结束：run()或call()方法执行完成，线程正常结束。

异常结束：线程抛出一个未捕获的Exception或Error。

调用stop：直接调用该线程的stop()方法来结束，容易导致死锁，不推荐。

4.**sleep与wait区别**

(1).sleep()方法属于Thread类，wait()方法属于Object类；

(2).sleep()方法导致了程序暂停执行指定的时间，让出CPU给其他线程，当指定时间到了

又会自动恢复运行状态；

(3).在调用sleep()方法的过程中，线程不会释放对象锁；

(4).当调用wait()方法的时候，线程会释放对象锁，进入等待此对象的等待锁定池，只有

针对此对象调用notify()方法后本线程才进入对象锁定池准备获取对象锁进入运行状

态。

5.**start与run的区别**

(1).start()方法来启动线程，真正实现了多线程运行，这是无需等待run方法体代码执行

完毕，可以直接继续执行下面的代码；

(2).通过调用Thread的start方法来启动线程，这时线程处于就绪状态，并没有运行；

(3).run()方法称为线程体，它包含了要执行的这个线程的内容，线程就进入了运行状态

开始运行run函数中的代码，run方法结束运行，此线程终止，然后CPU再调度其

他线程。