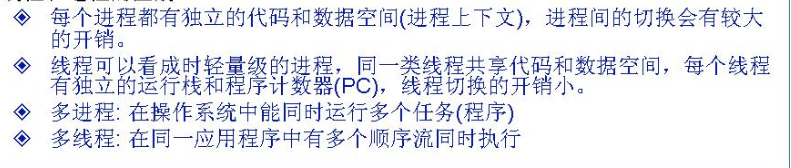
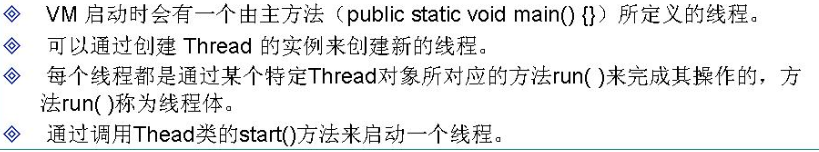
1. 线程是一个程序内部的顺序控制流。
2. 线程是**一个程序里面不同的执行路径**。
3. 线程和进程的区别：

进程是一个静态的概念，一个class文件，一个exe文件都是进程。进程开始执行指的是进程里的主线程开始执行。在机器里面实际上运行的都是线程。



1. Java的线程是通过java.lang.Thread类来实现的。



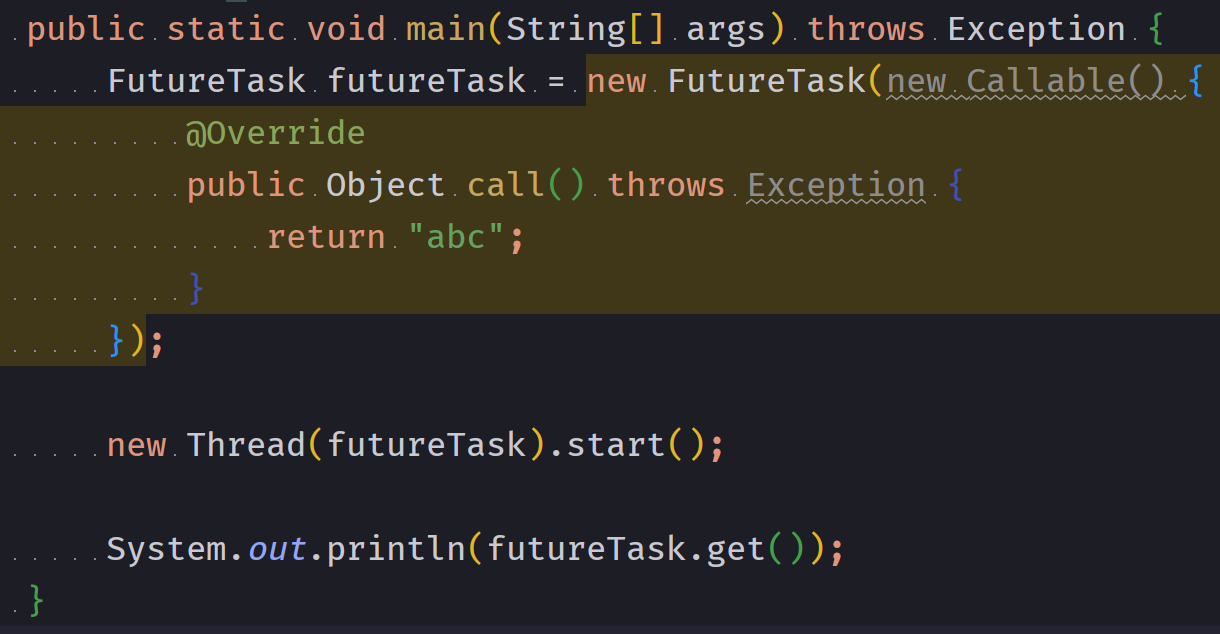
1. 线程的创建和启动：可以有两种方式创建新的线程(必须调用start方法，由start方法自动调用run方法；若自己直接调用run方法，则是方法调用，不是线程启动)
   1. **第一种(推荐，就用这个)**：实现Runnable接口



* 1. 第二种：继承Thread类

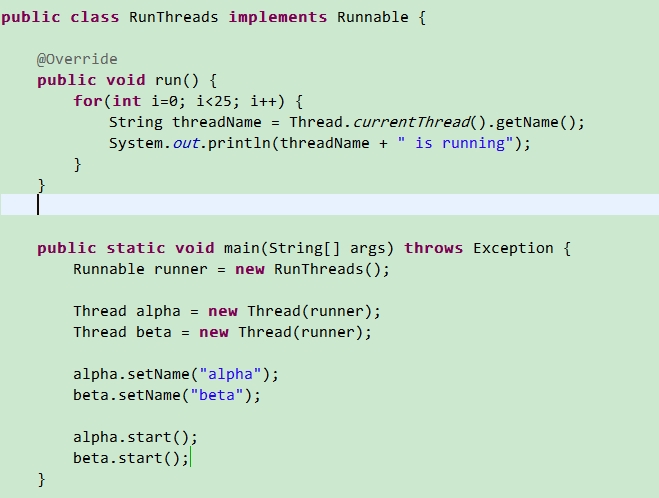


* 1. 第三种：FutureTask，Callable，可以有返回值

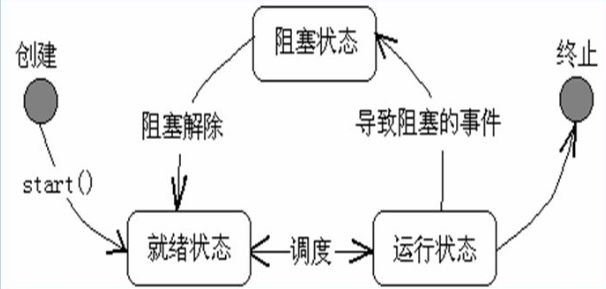


Note：**推荐使用接口**，即第一种方法，不从Thread类继承(实现接口的话还可以从别的类继承，并且还可以实现别的接口)

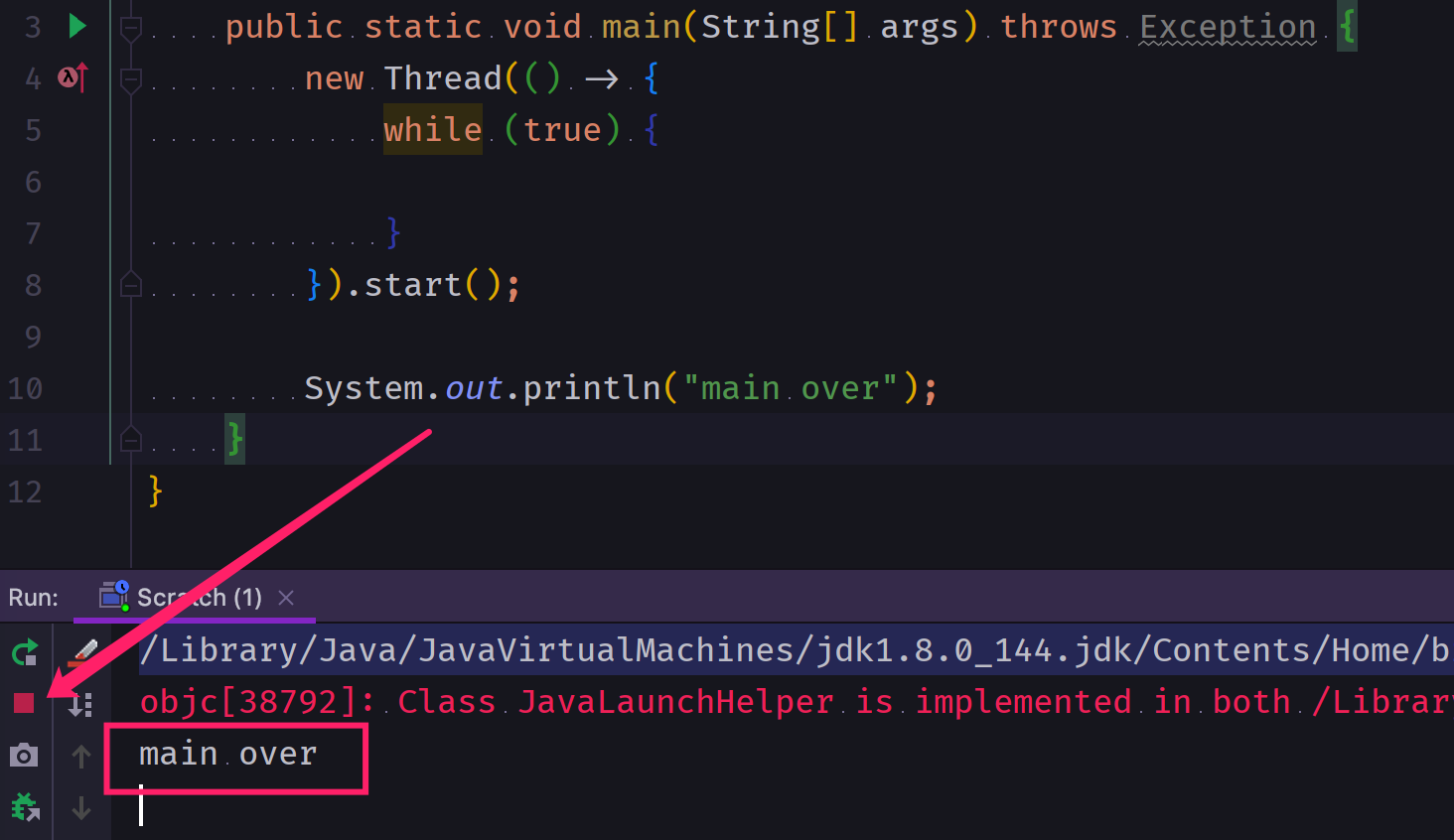
简单实例：



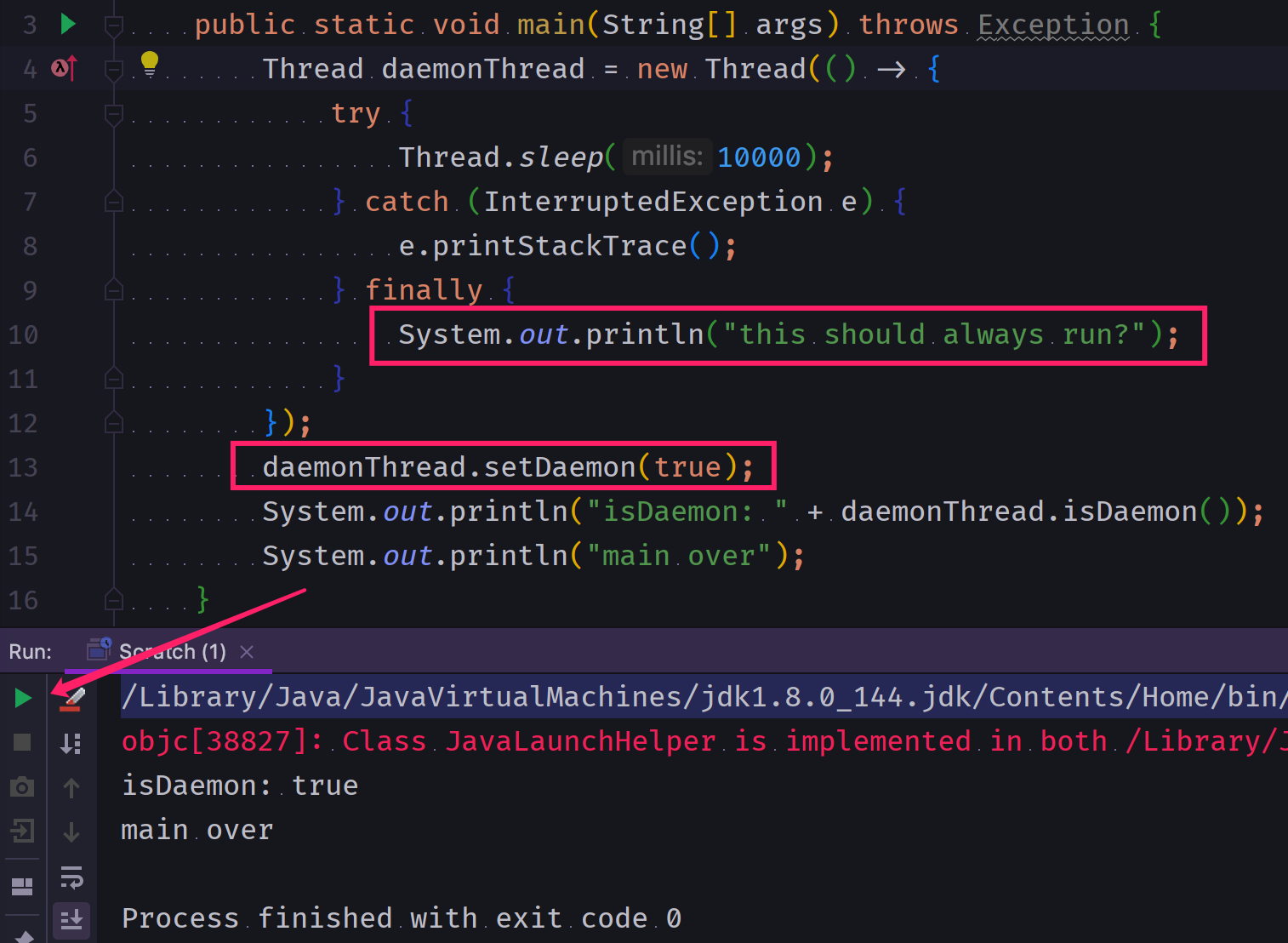
1. 线程的状态转换：



1. 守护线程与用户线程，Java中的线程分为两类，分别为daemon线程(守护线程〉和 user 线程(用户线程)，当最后一个非守护线程结束时，JVM会正常退出，即只要有一个user线程还没有结束，正常情况下JVM就不会退出
   1. 后台(Daemon)进程，在调用线程的start()方法前，调用setDaemon(true)将线程设置为后台进程。通过isDaemon()方法判断线程是否为后台进程
   2. 当所有的非后台线程结束时，会“突然”终止所有的后台线程，即使后台线程有finally也执行不到
   3. 简单示例
      1. 用户线程不结束，JVM不退出



* + 1. 守护进程『暴毙』



1. 定时器java.util.Timer类

