* **背景**
* 只使用单个线程完成多个任务（调用多个方法），肯定比用多个线程来完成用的时间更短，为何仍需多线程呢？
* **优点**
* 提高应用程序的响应。对图形化界面更有意义，可增强用户体验。
* 提高计算机系统CPU的利用率。
* 改善程序结构。将既长又复杂的进程分为多个线程，独立运行，利于理解和修改。
* **线程的分类**

Java中的线程分为两类：一种是**守护线程**，一种是**用户线程**。

* 它们在几乎每个方面都是相同的，唯一的区别是判断JVM何时离开。
* 守护线程是用来服务用户线程的，通过**在start()方法前调用thread.setDaemon(true)可以把一个用户线程变成一个守护线程**。
* Java垃圾回收就是一个典型的守护线程。
* **若JVM中都是守护线程，当前JVM将退出**。
* **线程的声明周期**

要想实现多线程，必须在主线程中创建新的线程对象。Java语言使用Thread类及其子类的对象来表示线程，在它的一个完整的生命周期中通常要经历如下的五种状态：

* **新建**：当一个Thread类或其子类的对象被声明并创建时，新生的线程对象处于新建状态。
* **就绪**：处于新建状态的线程被start()后，将进入线程队列等待CPU时间片，此时它已具备了运行的条件。
* **运行**：当就绪的线程被调度并获得处理器资源时,便进入运行状态，run()方法定义了线程的操作和功能。
* **阻塞**：在某种特殊情况下，被人为挂起或执行输入输出操作时，让出CPU并临时中止自己的执行，进入阻塞状态。
* **死亡**：线程完成了它的全部工作或线程被提前强制性地中止。
* 线程状态转换图：

