**进程**：是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，进程是系统进行资源分配和调度的一个独立单位。

**线程**：是进程的一个实体，是CPU调度和分派的基本单位，它是比进程更小的能独立运行的基本单位。线程自己基本上不用于系统资源，只拥有一点在运行中必不可少的资源（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线程共享进程所拥有的全部资源。

#### 进程与线程的区别

1. 进程是资源分配最小单位，线程是程序执行的最小单位；
2. 进程有自己独立的地址空间，每启动一个进程，系统都会为其分配地址空间，建立数据表来维护代码段、堆栈段和数据段，线程没有独立的地址空间，它使用相同的地址空间共享数据；
3. CPU切换一个线程比切换进程花费小；
4. 创建一个线程比进程开销小；
5. 线程占用的资源要比进程少很多。
6. 线程之间通信更方便，同一个进程下，线程共享全局变量，静态变量等数据，进程之间的通信需要以通信的方式（IPC）进行；（但多线程程序处理好同步互斥是个难点）
7. 多进程程序更安全，生命力更强，一个进程死掉不会对另外一个进程造成影响（源于有独立的地址空间），多线程程序更不易维护，一个线程死掉，整个进程就死掉了（因为共享地址空间）；
8. 进程对资源保护要求高，开销大，效率相对较低，线程资源保护要求不高，但开销小，效率高，可频繁切换；

#### 加强理解，做个简单的比喻：进程=火车，线程=车厢

* 线程在进程下进行（单纯的车厢无法运行）
* 一个进程可以包含多个线程（一辆火车可以有多个车厢）
* 不同进程间数据很难共享（一辆火车上的乘客很难换到另外一辆火车，比如站点换乘）
* 同一进程下不同线程间数据很易共享（A车厢换到B车厢很容易）
* 进程要比线程消耗更多的计算机资源（采用多列火车相比多个车厢更耗资源）
* 进程间不会相互影响，一个线程挂掉将导致整个进程挂掉（一列火车不会影响到另外一列火车，但是如果一列火车上中间的一节车厢着火了，将影响到所有车厢）
* 进程可以拓展到多机，进程最多适合多核（不同火车可以开在多个轨道上，同一火车的车厢不能在行进的不同的轨道上）
* 进程使用的内存地址可以上锁，即一个线程使用某些共享内存时，其他线程必须等它结束，才能使用这一块内存。（比如火车上的洗手间）-“互斥锁”
* 进程使用的内存地址可以限定使用量（比如火车上的餐厅，最多只允许多少人进入，如果满了需要在门口等，等于人出来才能进去）-“信号量”