<https://blog.csdn.net/weiyuefei/article/details/50414731>

**进程是什么？**

程序并不能单独运行，只有将程序装载到内存中，系统为它分配资源才能运行，而这种执行的程序就称之为进程。程序和进程的区别就在于：程序是指令的集合，它是进程运行的静态描述文本；进程是程序的一次执行活动，属于动态概念。

在多道编程中，我们允许多个程序同时加载到内存中，在操作系统的调度下，可以实现并发地执行。这是这样的设计，大大提高了CPU的利用率。进程的出现让每个用户感觉到自己独享CPU，因此，进程就是为了在CPU上实现多道编程而提出的。

**有了进程为什么还要线程？**

进程有很多优点，它提供了多道编程，让我们感觉我们每个人都拥有自己的CPU和其他资源，可以提高计算机的利用率。很多人就不理解了，既然进程这么优秀，为什么还要线程呢？其实，仔细观察就会发现进程还是有很多缺陷的，主要体现在两点上：

* 进程只能在一个时间干一件事，如果想同时干两件事或多件事，进程就无能为力了。
* 进程在执行的过程中如果阻塞，例如等待输入，整个进程就会挂起，即使进程中有些工作不依赖于输入的数据，也将无法执行。

如果这两个缺点理解比较困难的话，举个现实的例子也许你就清楚了：如果把我们上课的过程看成一个进程的话，那么我们要做的是耳朵听老师讲课，手上还要记笔记，脑子还要思考问题，这样才能高效的完成听课的任务。而如果只提供进程这个机制的话，上面这三件事将不能同时执行，同一时间只能做一件事，听的时候就不能记笔记，也不能用脑子思考，这是其一；如果老师在黑板上写演算过程，我们开始记笔记，而老师突然有一步推不下去了，阻塞住了，他在那边思考着，而我们呢，也不能干其他事，即使你想趁此时思考一下刚才没听懂的一个问题都不行，这是其二。

现在你应该明白了进程的缺陷了，而解决的办法很简单，我们完全可以让听、写、思三个独立的过程，并行起来，这样很明显可以提高听课的效率。而实际的操作系统中，也同样引入了这种类似的机制——线程。

**线程的优点**

因为要并发，我们发明了进程，又进一步发明了线程。只不过进程和线程的并发层次不同：进程属于在处理器这一层上提供的抽象；线程则属于在进程这个层次上再提供了一层并发的抽象。如果我们进入计算机体系结构里，就会发现，流水线提供的也是一种并发，不过是指令级的并发。这样，流水线、线程、进程就从低到高在三个层次上提供我们所迫切需要的并发！

除了提高进程的并发度，线程还有个好处，就是可以有效地利用多处理器和多核计算机。现在的处理器有个趋势就是朝着多核方向发展，在没有线程之前，多核并不能让一个进程的执行速度提高，原因还是上面所有的两点限制。但如果讲一个进程分解为若干个线程，则可以让不同的线程运行在不同的核上，从而提高了进程的执行速度。

例如：我们经常使用微软的Word进行文字排版，实际上就打开了多个线程。这些线程一个负责显示，一个接受键盘的输入，一个进行存盘等等。这些线程一起运行，让我们感觉到我们输入和屏幕显示同时发生，而不是输入一些字符，过一段时间才能看到显示出来。在我们不经意间，还进行了自动存盘操作。这就是线程给我们带来的方便之处。

**进程与线程的区别**

* 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.线程自己基本上不拥有系统资源,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的线程共享进程所拥有的全部资源。
* 一个线程可以创建和撤销另一个线程，同一个进程中的多个线程之间可以并发执行。

进程和线程的主要差别在于它们是不同的操作系统资源管理方式。进程有独立的地址空间，一个进程崩溃后，在保护模式下不会对其它进程产生影响，而线程只是一个进程中的不同执行路径。线程有自己的堆栈和局部变量，但线程之间没有单独的地址空间，一个线程死掉就等于整个进程死掉，所以多进程的程序要比多线程的程序 健壮，但在进程切换时，耗费资源较大，效率要差一些。但对于一些要求同时进行并且又要共享某些变量的并发操作，只能用线程，不能用进程。

**本质区别：**

**是否共享地址空间，及共享多少地址空间。传统意义上, 进程之间是不共享地址空间的，而线程是共享着进程的地址空间。**

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/weiyuefei/article/details/50414731