java并发编程之背景知识

原创 2018-04-13 小孩子 我们都是小青蛙

这两天公众号粉丝涨的太快，我有点儿飘～都有点静不下心写东西了，承蒙各位厚爱，帮我传播分享，我就可以减少宣传我公众号的时间来专心的继续原创文章了。由于关于MySQL的内容还没写完，今天就准备再开一个专题 --- java并发编程。哈哈，这个专题可是面试中最爱提的问题哈，但是网上的系列博客或者现实中的书籍对此的描述并非常绕口，并不易于小白理解，在写作本系列文章时，我参考了《java并发编程实战》、《java并发编程的艺术》、《深入理解计算机系统》、《think in java》、《java语言规范》《深入理解java虚拟机》等等书籍，还有的想不起来了，它们总有这样那样的问题让小白挠头，解释不清的地方，最最重要的他们叙述方式都极其死板，一点儿都不轻松，不好玩儿～所以我准备从零开始，写一个给小白看的java并发编程的专题，只要你具有基本的java编程知识，就可以从头开始看这一系列文章，放心，文章里的铺垫我都给你做好了，你只需要安安心心坐下来，不要跳着看！不要跳着看！不要跳着看！不要跳着看！就可以了。好了扯犊子结束，开始正文。非常重要的几条阅读建议：

1. 最好使用电脑观看。
2. 如果你非要使用手机观看，那请把字体调整到最小，这样观看效果会好一些。
3. 碎片化阅读并不会得到真正的知识提升，要想有提升还得找张书桌认认真真看一会书，或者我们公众号的文章😊😊。
4. 如果觉得不错，各位帮着转发转发，如果觉得有问题或者写的哪不清晰，务必私聊我～
5. 本公众号的文章都是需要被系统性学习的，这篇文章是java并发编程系列的第一篇，只需要你掌握基本语法的java语法就行（我不知道你的java语法是不是培训机构教的，不过之后可能会开java语法的专题，等我～）

操作系统发展回顾

裸机

老早之前的计算机只有一个处理器，而一个处理器在同一时刻只能处理一条指令，换句话说，我们的代码需要一行一行的按顺序被计算机执行，计算机只能把一个程序完整的执行完，然后再执行第二个程序。所以计算机专业的同学们要排队去机房做实验，一个人执行完然他的程序后，第二个人再执行自己的程序，这也就意味着所有计算机资源是被一个程序独占的，计算机资源包括处理器、内存、硬盘、输入 / 输出设备啥的。这样的计算机系统我们称之为 裸机 。

简单批处理系统

后来人们发现对于价格高昂的计算机设备来说，在换人的过程中就浪费了好多时间，时间就是金钱，有这些时间可以多执行好多程序了。所以有人写了一个程序，把所有同学们需要做实验的程序都放在这个程序里排个队，由这个程序来协调各个同学们的程序执行，一个执行完了立即换成另一个，这样就不用人工干预了，所以他们把这样的系统叫做 简单批处理系统 ，而那个负责协调各个童鞋们程序的程序，就是所谓的 操作系统 的雏形。

多道批处理系统

我们知道，处理器的速度是嗖嗖的，比内存访问的速度快好多个数量级，而内存又比硬盘、打印机等 I/O 设备啥的快好多个数量级，而程序执行过程中又免不了从硬盘里读个文件，往打印机输出个啥的，所以处理器浪费了好多时间等待这些I/O操作的完成。再一次，时间就是金钱，为了尽可能的剥削计算机的运算能力，在程序遇到 I/O 操作或者什么其他会阻塞程序执行的操作时，处理器会转向执行其它的程序，什么时候这个阻塞的操作完成了，再掉过头继续执行它。从宏观上看，处理器可以各个程序轮流执行，所以这样的系统就称为 多道批处理系统 。

分时处理

有的同学对排队执行程序这个事儿很有意见，大家都是学生，凭啥先执行你的后执行我的，你要是写了个 while(true) ，那大家都得玩儿完～所以人们给每个程序都分配一点处理器时间去轮流执行，每个程序分配到的执行时间就叫做 时间片 ，这个过程也叫做 分时处理 。又因为处理器速度太快了， 时间片 的大小可以做到微秒毫秒的大小，所以这个切换的过程对于人来说根本感觉不出来，所以看起来像各个程序在同时执行。不过后来人们在一台计算机上又装了多个处理器，就是我们常听说的 4核 、 8核 啥的，所以也可能真正的在同时执行。

而 时间片 具体设置为多大，处理器怎么切换各个程序的执行，这些工作就是所谓的 操作系统 来控制的。

进程

进程的概念和特点

我们自己写的程序，也就是所谓的 用户程序 是由操作系统来管理的，人们把一个执行着的程序叫做一个 进程 (英文名： Process)，每个 进程 都有这么两个特点：

1. 资源所有权

   程序在运行过程中需要一定的资源，比如内存、 I/O 啥的，这些东西不能在不同进程间共享，假如一个进程占了另一个进程的内存，那另一个进程的数据不就丢失了么；一个进程正在使用打印机输出东西，另一个进程也使用的话，不就尴尬了么。所以进程所拥有的这些资源是不能共享的，而这种资源分配的活是由 操作系统 来管理的。

2. 调度/执行

   操作系统 会为它管理的进程分配 时间片 ，来调度哪个进程应该被处理器处理，哪个应该先休息一会儿。

所以我们现在电脑里每个运行着的程序都是一个进程，可以打开你的任务管理器(windows)或者活动监视器(mac)，看到我们的电脑里其实有好多好多进程喔，什么QQ、微信、音乐播放器、视频播放器啥的。

进程的状态

在操作系统级别上，进程根据它运行的情况，可以分成下边5种状态：

1. 新建

   刚刚创建的那个时刻， 操作系统 会为这个进程在内存中创建相应的数据，比如分配这个进程的ID，优先级什么的～这个状态持续的时间比较短。

2. 就绪

   一旦该进程相关的一些数据创建好了，这个进程就会被放在一个叫 就绪队列 的队列里，之后操作系统就会从这个队列里挑一个进程放到处理器里执行。

3. 运行

   这个进程被 操作系统 分配了时间片，处理器开始执行它。

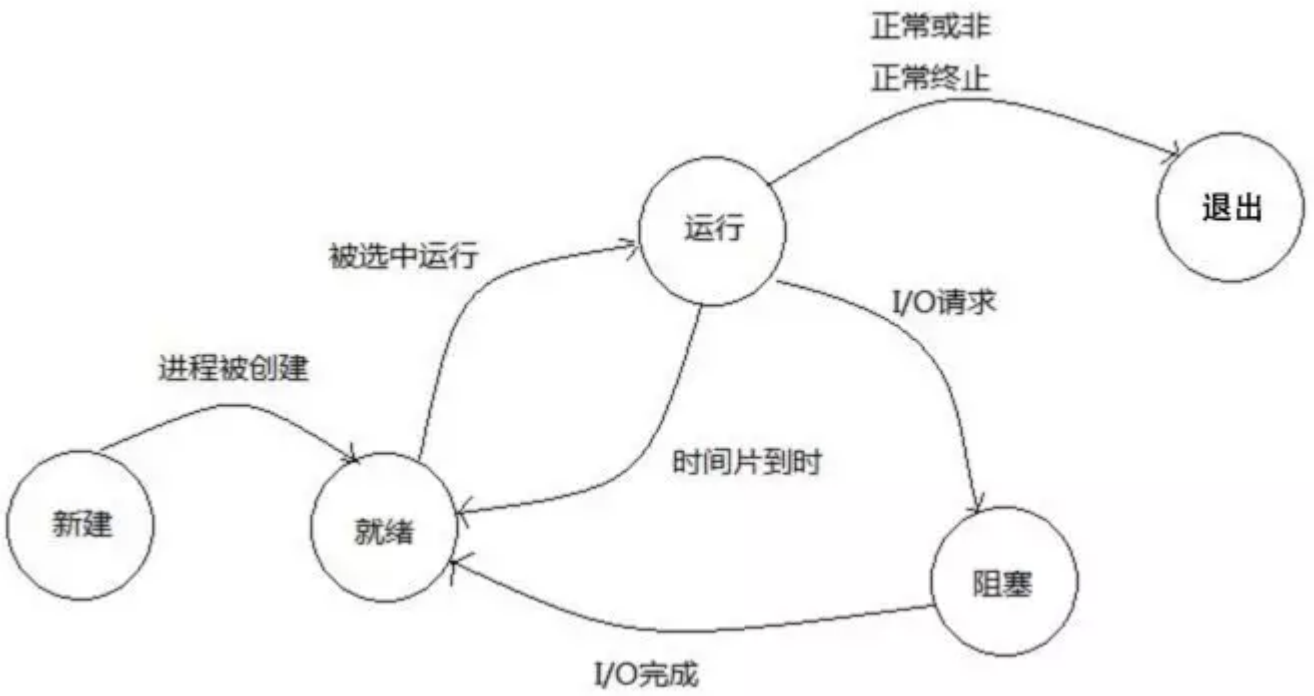
4. 阻塞

   在执行进程的过程中，可能遇到某些阻塞的动作，比如 I/O 操作，处理器如果一直等待该阻塞动作完成的话就太浪费时间了，所以会把等待阻塞动作完成的进程放到一个叫 阻塞队列 的队列里，之后并不会从这个队列里挑选即将执行的进程，而是直到该阻塞动作完成，才重新把该进程放到 就绪队列 里等待执行。

5. 退出

   该进程执行完毕，或者遇到了什么错误，或者操作系统就是想弄死它，它就被杀死了，这个状态里操作系统可能还会记录一下死因啥的，这个过程也很短。

这些状态的具体转换过程看下图：



串行编程和并行编程

到目前为止，我们的编程模式都是 串行编程 ，也就是处理器执行完一条指令再执行另一条。举个例子啊，比如你妈给你布置了两个 任务 ：一是去打瓶酱油，二是去烧一壶水，按照 串行编程 的方式就是两个任务依次进行，也就是说你先打酱油，然后回来再烧水。这么做没啥问题，但是没有效率啊，所以你也可以先把水烧上，然后去打酱油，回来正好水烧开了～这种多个任务同时进行的编程方式就叫做 并行编程 。

回到计算机中来， 串行编程 的方式就是我们把所有要完成的任务放到一个 进程 中去执行，而 并行编程 的方式就是我们去开几个 进程 同时执行(注：这个 同时 可能是假的，如果只有单个处理器，那就是看上去是 同时 的)。这种 并行编程 的优势是显而易见的：

1. 充分利用多个处理器

   现代计算机的处理器越来越多，不用白不用。如果所有的任务都塞到1个进程中，而计算机实际有100个处理器，那么将会有99%的计算能力将被浪费。

2. 防止任务的阻塞

   即使是在单个处理器中，为多个任务创建多个进程也是有好处的。先执行的任务可能会需要执行某些 I/O 操作而造成阻塞，所以就需要等待 I/O 操作完成才能继续执行。但是如果为每个任务创建一个进程之后，一个任务阻塞掉并不会影响别的任务的正常执行。

3. 简化编程的模型

   把一个大任务拆分成若干个小任务自然是会事情清晰许多(注：也不是绝对啊)，也就是所谓的大事化小，小事化了～所以把各个任务分配到不同进程里去执行在我们编程方面也会容易一些(注：不是绝对啊)。

线程

线程的概念

进程 是个好东西，可以给每个任务都分配一个进程以达到并发执行的目的。可是运行了一段时间人们发现还是有一些不好的地方的：

1. 不同进程之间的资源不能够共享。

   这个对于为了一个大目标细分成的若干小任务很不友好。比方说对于一个网站来说，针对每一个访问网站的连接都去创建一个进程，如果我们想累加一下总的访问连接数就比较麻烦了，因为各个进程不能去修改同一块内存。

2. 创建、切换、销毁进程成本太大。

   戈操作，至于操作系统底层对于创建、切换、销毁进程都要做哪些东西不是我们唠叨的范围，但是这些操作的开销真的很大，你知道这一点就好了～

再返回头来看 进程 的两个特点，一是对资源的所有权，二是可以作为操作系统调度和执行的单位，这两个特点是没有关系的，也就是说独立的，现代的好多操作系统已经把这两个特点给拆开了，可以被调度和执行的单位通常被称作 线程 或者 轻量级进程 ，而拥有资源所有权的单位通常被称为 进程 。

小贴士：
由于历史原因，原先的`进程`既是资源的分配单位，也是调度和执行的单位，所以我们打开一个程序就相当于打开一个进程。
提出`线程`概念之后，这种`打开一个程序就相当于打开一个进程`的叫法也被保留了下来。其实现在打开一个程序的意思是`打开一个进程并且打开若干

线程的特点

看一下这个 线程 的各种特点

1. 一个进程至少对应一个线程

   操作系统每开始执行一个程序，都会为其分配所需的资源以及创建若干个 程序执行流 ，也就是说会创建一个线程以及若干线程。

2. 各个 线程 可以共享同一个进程中的各种资源

   因为 线程 是从属于某个进程，所以进程中的内存、 I/O 啥的资源这个 线程 都可以访问到。接着上边那个例子，我们可以对于每一个连到网站的连接都可以分配一个线程，然后在申请一块儿内存表示连接数，每创建一个线程都把连接数加1，问题就愉快的解决了。

3. 创建、切换、销毁线程的成本远低于原先进程的成本

   因为只是一个调度和执行的单位，本来就是原先进程概念的一部分，所以创建、切换、销毁线程的成本就小多了。

4. 线程通信比进程通信的效率高

   原先进程间通信需要配合各种机制，现在线程间由于可以共享内存，所以通信就easy多了。

   至于进程间通信需要啥机制我这就不说了，不是我们讨论的范围。

所以 线程 的提出完美的解决了一开始提出的把不同任务分配给不同进程执行的缺陷，现在我们可以把不同的任务分配给不同的 线程 去执行，不仅可以方便通信交流，而且创建和使用的成本还低～

线程的状态

由于线程只是单纯的继承了线程中调度和执行的特性，所以原先进程拥有的状态，现在线程一样拥有，也就是： 创建 、 就绪 、 执行 、 阻塞 、 退出 。

总结

由于本篇比较简单，总结就免了。

题外话

写文章挺累的，有时候你觉得阅读挺流畅的，那其实是背后无数次修改的结果。如果你觉得不错请帮忙转发一下，万分感谢～

投诉