浅析Java多线程程序设计机制 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/144/2021_2022__E6_B5_85_E6_9E_90Java_c104_144363.htm 多线程是Java语言的一大特性，多线程就是同时存在N个执行体，按几条不同的执行线索共同工作的情况。程序，进程，线程，可以从不同的角度去理解。程序就是一段静态的代码，可以理解成一组计算机命令的集合.进行就是这个程序一次动态的执行过程，从代码的加载到执行完毕的一个过程。线程是一个比进程小的单位，一个进程再执行的过程中可以产生多个线程，每个线程也是由生产到销毁，可以理解成是进行的子集。我个人用一个觉得还算恰当的比喻来比喻三者。QQ客户端就是一个程序，登陆一个QQ就是开始了这个程序的一个进程，再QQ上发送消息给好友就貌似这个进程中的一个线程。不知道这样比喻恰当否? 线程也是有状态和声明周期的，每个Java程序都会有一个缺省的主线程，对于应用程序applcation来说main方法就是一个主线程.Java语言使用的是Thread类及其子类的对象来表示线程的.创建一个新的线程的生命周期如下状态: 1) 新建:当一个Thread类或者其子类的对象被声明并创建时，新的线程对象处于新建状态，此时它已经有了相应的内存空间和其他资源. 2) 就绪:处于新建状态的线程被启动后，将进入线程队列排队等待CUP服务，这个时候具备了运行的条件，一旦轮到CPU的时候，就可以脱离创建它的主线程独立开始自己的生命周期. 3) 运行:就绪的线程被调度并获得CUP的处理边进入了运行状态，每一个Thread类及其子类的对象都有一个重要的run()方法，当线程对象被调度执行的时候，它将自动调

用本对象的run()方法，从第一句代码开始执行。所以说对线程的操作应该写到run()方法中. 4) 阻塞:一个正在执行的线程如果再某种情况下不能执行了.进入阻塞状态，这个时候它不能进入排队状态，只有引起了阻塞的原因消失的时候，线程才可以继续进入排队状态等待CUP处理。 5) 死亡:处于死亡状态的线程不具有继续执行的能力，线程死亡主要的原因是正常运行的线程完成了全部工作，即执行完了run()方法，另外就是被提前强制的终止了。 线程的调度也是有优先级别的，就是说同样的排列优先级高可以提前被CPU处理，主要分三个级别，高中低.分别代表的数字是10.5.1分别有三个常量代表不可以被改变。最小优先级的常量是MIN_PRIORITY，普通的优先级的常量是NORM_PRIORITY，最高的优先级的常量是MAX_PRIORITY一般主线程的优先级是普通，另外可以通过Thread类的setPriority(int a)方法来修改系统自动设置的线程优先级。 Java中编程实现多线程应有两种途径，一种是创建自己的线程子类，另外是实现一个接口Runnable。无论是那种途径最终读需要使用Thread类及其方法。Thread类有两种构造方法，public Thread()用来创建一个线程对象。public Thread(Runnable r)创建线程对象，参数r成为被创建的目标对象。这个目标必须实现Runnbale接口，给出该接口的run()方法的方法体，再方法体中操作.用两个构造方法创建完的线程就是一个新建的状态，等待处理.然后启动线程的start()方法，启动线程对象，线程进入排队状态也就是就绪状态.然后线程操作run()方法这个方法里的内容是被系统处理的内容.如果想使线程进入休眠状态可以调用sleep()方法，可以给一个参数，代表休眠的毫秒.如果给两个参数代表那秒。终止线程

用yield()方法来完成.判断线程是否销毁可以用idAlive()方法判断线程是否活着。Runnable接口只有一个方法run()方法，我们实现这个接口把要操作的代码写到这个方法中，然后再把实现了整个接口的类的实例传给Therad类的构造方法即可操作线程。 线程同步是一个再处理线程的时候需要注意的问题，同步方法要用synchronized关键字类修饰，被这个关键字修饰后，当一个线程A使用这个方法后，其它线程想使用这个方法就必须等待，知道线程A使用完该方法后方可使用.下面我写一个例子来说明线程同步，这个例子有两个线程会计和出纳，他们共同有一个账本.他们俩都可以存取方法对账本进行访问，会计使用存取方法的时候写入存钱的记录，出纳则写入取钱的记录。因此会计使用账本的时候出纳被禁止使用，反之也是如此。就是一个人使用另外一个人必须等待。下面我通过一个小程序Applet来实现这个事: import Java.applet.*. import Java.awt.*. import Java.awt.event.*. public class MyThread extends Applet implements Runnable { int money = 100. TextArea text1 = null. TextArea text2 = null. Thread Kuaiji = null. Thread Chuna = null. public void init() { Kuaiji = new Thread(this). Chuna = new Thread(this). text1 = new TextArea(20，8). text2 = new TextArea(20，8). add(text1). add(text2). } public void start() { Kuaiji.start(). Chuna.start(). } public synchronized void Cunqu(int number) { if(Thread.currentThread() == Kuaiji) { for(int i=1.i { money = money number. try {Thread.sleep(1000).} catch(Exception e){} text1.append("\n" money). } } else if(Thread.currentThread() == Chuna) { for(int i=1.i { money = money - number/2. try {Thread.sleep(1000).} catch(Exception e){} text2.append("\n"

money). } } } public void run() {
if(Thread.currentThread()==Kuaiji ||
Thread.currentThread()==Chuna) { for(int i=1.i { Cunqu(30). } } }
} 当一个线程使用同步方法中的某个变量，而此变量又需要其他线程修改后才能符合本线程的需要，那么可以再同步方法中使用wait()方法，使本线程等待，并允许其他线程使用这个同步方法.用notfyAll()方法同志所有的由于使用这个同步方法的处于等待的线程结束等待进入同步方法中运行，如果使使用notfy()就是单独同志一个线程进行同步方法进行活动.简单的理解就是wait()方法让线程等待，notfy()当一个线程运行，notfyAll()让全部线程运行。虽然Java支持多线程.一般线程不需要我们自己处理.但是也是需要了解和掌握的。再日后的项目中获取会根据不同情况，有不同的需求。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com