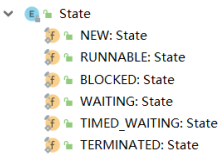
线程的状态及转换

2019年8月3日 15:36

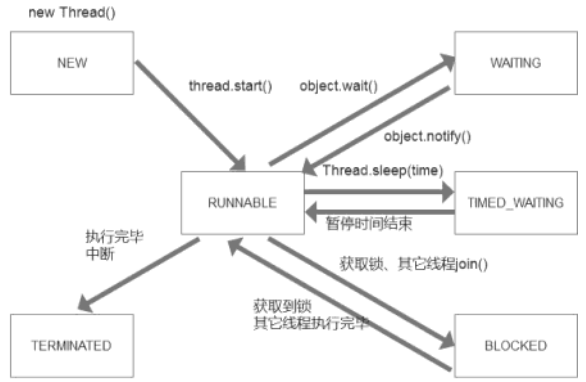
Thread类中的状态为




```
package thread;

public class ThreadStateDemo implements Runnable{
    private static final Object o = new Object();

    @Override
    public void run() {
        synchronized (o) {
            try {
                // 线程0到达此处，线程1等待线程0释放锁，线程2状态为BLOCKED
                Thread.sleep(300L); // 语句1
                // 线程0到达此处，状态为WAITING并释放锁，线程1获取到锁执行语句1
                o.wait(100);
                // 线程1还未到达wait()时，主线程唤醒线程0，线程0继续执行
                Thread.sleep(100L); // 语句3
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }



    public static void main(String[] args) {
        Thread thread = new Thread(new ThreadStateDemo(), "线程0");
        Thread thread1 = new Thread(new ThreadStateDemo(), "线程1");
        System.out.println(thread.getName() + thread.getState()); // NEW
        thread.start();
        thread1.start();
        System.out.println(thread.getName() + thread.getState()); // RUNNABLE
        try {
            Thread.sleep(400L); // 延迟400ms，此时线程0进入wait并释放锁，线程1拿到锁执行语句1
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println(thread.getName() + thread.getState()); // WAITING
        System.out.println(thread1.getName() + thread1.getState()); // TIMED_WAITING
        // 主线程等待锁直到线程1进入wait释放锁
        synchronized (o) {
            o.notifyAll(); // 唤醒两个线程
        }
        System.out.println(thread.getName() + thread.getState()); // 拿到锁的线程进入语句3，另一个线程则BLOCKED
        try {
            thread.join(); // 等待执行完毕，主线程状态为WAITING
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println(thread.getName() + thread.getState()); // TERMINATED

    }
}
```

输出的结果为：

线程0NEW
线程0RUNNABLE
线程0WAITING
线程1TIMED_WAITING
线程0BLOCKED
线程0TERMINATED



线程执行的方法

主线程
1. 启动两个线程
2. 延迟400ms
3. 唤醒某个等待的线程，被唤醒的线程状态为RUNNABLE
4. 等待线程0执行完毕，Thread.join()，此时主线程的状态是WAITING，线程0执行完毕时状态为TERMINATED

线程0开始执行Thread.start()状态为RUNNABLE
线程1等待获取锁，状态为BLOCKED

锁住的内容
1. 延迟300ms，Thread.sleep()状态为TIMED_WAITING
2. wait()，状态为WAITING
3. 延迟100ms

注：

thread.join()方法源码为：

```
while (isAlive()) {
    wait(0);
}
```

即主线程中的thread.join()方法可以替换为：

```
synchronized(thread) {
    while (thread.isAlive()) {
        // 不停的进行判断，防止被假唤醒
        try {
            thread.wait(0);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

所以此时主线程的状态为WAITING

wait()方法就是wait(0)

```
public final void wait() throws InterruptedException {
    wait(0);
}
```