

Task 2: States and Transitions

Create a Java class that simulates a thread going through different lifecycle states: NEW, RUNNABLE, WAITING, TIMED_WAITING, BLOCKED, and TERMINATED. Use methods like sleep(), wait(), notify(), and join() to demonstrate these states..

```
class StateAndTransition implements Runnable {  
    private final Object lock = new Object();  
  
    @Override  
    public void run() {  
        System.out.println(Thread.currentThread().getName() + ": State RUNNABLE");  
  
        // Demonstrate TIMED_WAITING using sleep  
        try {  
            System.out.println(Thread.currentThread().getName() + ": State TIMED_WAITING (sleeping)");  
            Thread.sleep(1000);  
        } catch (InterruptedException e) {  
            Thread.currentThread().interrupt();  
        }  
  
        // Demonstrate WAITING using wait  
        synchronized (lock) {  
            try {  
                System.out.println(Thread.currentThread().getName() + ": State WAITING (waiting for lock)");  
                lock.wait();  
            } catch (InterruptedException e) {  
                Thread.currentThread().interrupt();  
            }  
        }  
  
        // Continue running  
        System.out.println(Thread.currentThread().getName() + ": State RUNNABLE again");  
  
        // Demonstrate BLOCKED  
        synchronized (lock) {
```

```
        System.out.println(Thread.currentThread().getName() + ": Acquired lock, not BLOCKED anymore");
    }

    System.out.println(Thread.currentThread().getName() + ": State TERMINATED (exiting run)");
}

public static void main(String[] args) {
    StateAndTransition lifecycle = new StateAndTransition();

    // State NEW
    Thread thread = new Thread(lifecycle, "Thread-1");
    System.out.println(thread.getName() + ": State NEW");

    // Start the thread
    thread.start();

    // Notify the waiting thread
    try {
        Thread.sleep(500); // Ensure the thread reaches the wait state
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }

    synchronized (lifecycle.lock) {
        lifecycle.lock.notify();

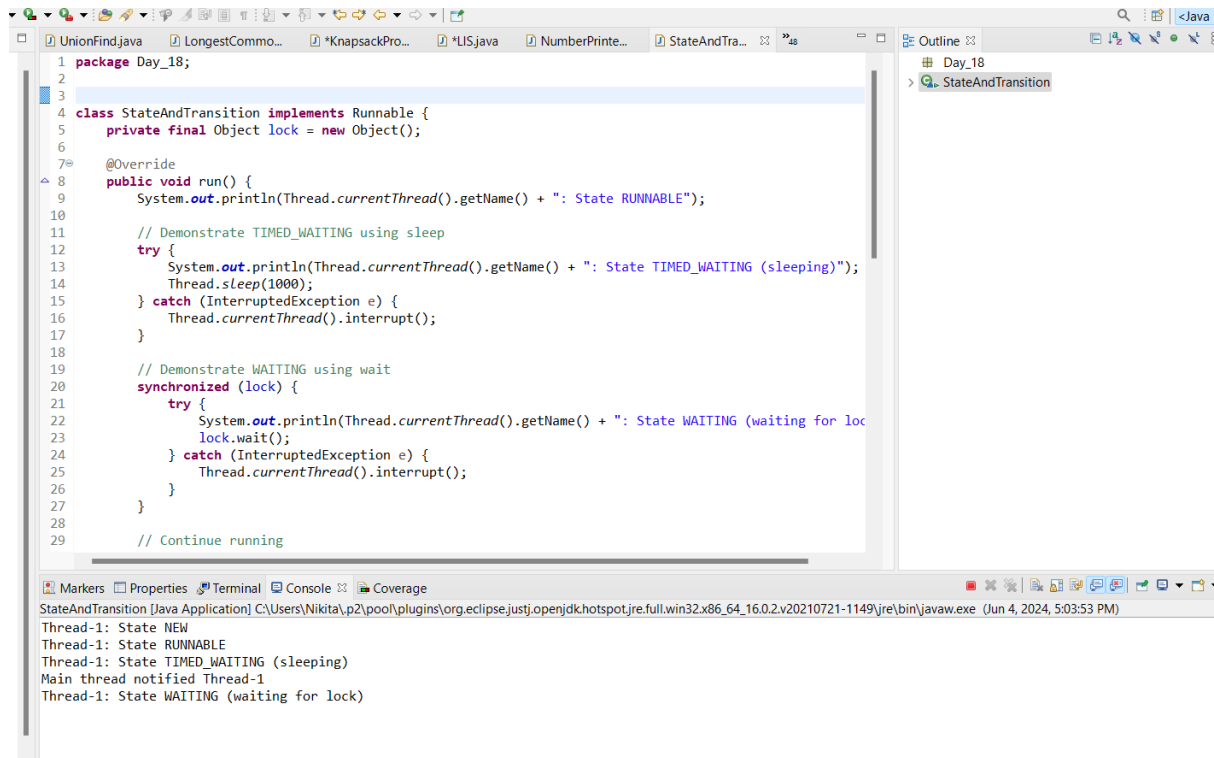
        System.out.println("Main thread notified " + thread.getName());
    }

    // Wait for the thread to terminate
    try {
        thread.join();
    } catch (InterruptedException e) {
        Thread.currentThread().interrupt();
    }
}
```

```
System.out.println(thread.getName() + ": State TERMINATED (confirmed in main)");
```

```
}
```

```
}
```



The screenshot shows the Eclipse IDE with a Java project named 'Day_18'. The main editor displays the source code for 'StateAndTransition.java'. The code defines a class that implements the 'Runnable' interface. It uses a 'lock' object to demonstrate thread synchronization. The 'run()' method prints the thread's state at various points: 'RUNNABLE', 'TIMED_WAITING (sleeping)', and 'WAITING (waiting for lock)'. It uses 'Thread.sleep(1000)' to simulate a delay and 'lock.wait()' to wait for the lock. The console at the bottom shows the execution output, confirming the thread's state transitions.

```
1 package Day_18;
2
3
4 class StateAndTransition implements Runnable {
5     private final Object lock = new Object();
6
7     @Override
8     public void run() {
9         System.out.println(Thread.currentThread().getName() + ": State RUNNABLE");
10
11         // Demonstrate TIMED_WAITING using sleep
12         try {
13             System.out.println(Thread.currentThread().getName() + ": State TIMED_WAITING (sleeping)");
14             Thread.sleep(1000);
15         } catch (InterruptedException e) {
16             Thread.currentThread().interrupt();
17         }
18
19         // Demonstrate WAITING using wait
20         synchronized (lock) {
21             try {
22                 System.out.println(Thread.currentThread().getName() + ": State WAITING (waiting for lock)");
23                 lock.wait();
24             } catch (InterruptedException e) {
25                 Thread.currentThread().interrupt();
26             }
27         }
28
29         // Continue running
30     }
31 }
```

Console Output:

```
StateAndTransition [Java Application] C:\Users\Nikita\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (Jun 4, 2024, 5:03:53 PM)
Thread-1: State NEW
Thread-1: State RUNNABLE
Thread-1: State TIMED_WAITING (sleeping)
Main thread notified Thread-1
Thread-1: State WAITING (waiting for lock)
```