

69% completed



Search Course

Java Concurrency
Reference

- Setting-up Threads
- Basic Thread Handling
- Executor Framework
- Executor Implementations
- Thread Pools
- Types of Thread Pools
- An Example: Timer vs ScheduledThreadPool
- ThreadPoolExecutor
- Callable Interface
- Future Interface
- CompletionService Interface
- ThreadLocal
- ThreadLocalRandom
- CountDownLatch
- CyclicBarrier
- Concurrent Collections
- ConcurrentHashMap
- ConcurrentModificationException

Practice Mock Interview →

ConcurrentModificationException

Practice Mock Interview →

Java Multithreading for Senior Engineering Interviews / ... / IllegalMonitorStateException

IllegalMonitorStateException

Learn the reasons that cause IllegalMonitorStateException to be thrown.

If you are interviewing, consider buying our [number#1 course for Java Multithreading Interviews](#).

Explanation

The `IllegalMonitorStateException` is a common programming error that can show up in concurrent programs. Depending on the structure of the program, the exception may occur consistently or only occasionally. The `IllegalMonitorStateException` exception class extends the `RuntimeException` class and according to the official documentation is thrown to indicate that a thread has attempted to wait on an object's monitor or to notify other threads waiting on an object's monitor without owning the specified monitor. In other words if you invoke `wait()` or `notify()/notifyAll()` without synchronizing on the object i.e. outside of a `synchronized` method or block (with the object as the synchronization target) then `IllegalMonitorStateException` will be thrown. Similarly, the exception is thrown when you invoke these methods on an instance of `Condition` class without acquiring the associated lock with the condition. The class is part of the `java.lang` package and not `java.util.concurrent.*`.

Repro using Lock

The program below demonstrates generating `IllegalMonitorStateException` using a `ReentrantLock`.

The program below demonstrates generating `IllegalMonitorStateException` using a `Condition` object that was instantiated from a `Lock` object.

```
1- import java.util.concurrent.locks.Condition;
2- import java.util.concurrent.locks.Lock;
3- import java.util.concurrent.locks.ReentrantLock;
4-
5- class Demonstration {
6-     public static void main( String args[]) throws Exception {
7-         Lock lock = new ReentrantLock();
8-         Condition condition = lock.newCondition();
9-
10        // throws exception because we didn't lock the associated
11        // Lock object with the condition variable before invoking
12        // await() on the condition object.
```

The fix for the above program appears below:

```
// acquire the associated Lock
lock.lock();
try {
    while (/* some condition */) {
        // always invoke await or wait in a loop to cater for spurious wake-ups
        // and after synchronizing on the associated Lock
        condition.await();
    }
} finally {
    // remember to unlock in a finally block
    lock.unlock();
}
```

Repro using object

The program in the widget below causes `IllegalMonitorStateException` by invoking `notifyAll()` on an object without synchronizing on it.

```
1- class Demonstration {
2-     public static void main( String args[] ) throws Exception {
3-         Object myObject = new Object();
4-
5-         // throws exception because we didn't synchronize
6-         // on myObject before invoking the wait() method
7-         myObject.notifyAll();
8-     }
9- }
```

The fix for the above program appears below:

```
|   Object myObject = new Object();  
  
|   synchronized (myObject) {  
|       // invoking notifyAll() in a block synchronized on myObject  
|       myObject.notifyAll();  
|   }
```

Technical Quiz

Consider the program below and state the outcome of executing the `main()` method.

```
1. public class IllegalMonitorStateException {  
    synchronized void someFunction() throws InterruptedException {  
        |   this.wait();  
        |   }  
  
    public static void main(String[] args) throws Exception {  
        |   (new IllegalMonitorStateException()).someFunction();  
        |   }  
}
```

- ☐ A. `IllegalMonitorStateException` is thrown.
- ☐ B. `InterruptedException` is thrown.
- ☐ C. Program is either blocked on `wait()` forever or exits because of a spurious wakeup.
- ☐ D. The target of `synchronized` and the object invoking `wait()` are different and the program doesn't compile.

Reset Quiz

< Q1 / Q2 >

Submit Answer

← Back lesson

✓ Mark As Completed

Next →

Phaser

TimeoutException