# SPOS Practical\Mutex Semaphore\Mutex.java

1. **import** java.util.concurrent.Semaphore;

# public class Mutex {

1. **static** Semaphore semaphore = **new** Semaphore(1);

# static class MyLockerThread extends Thread {

1. String name = "";
2. MyLockerThread(String name) {
3. **this**.name = name; 8 }

9 **public void run**() {

# try {

1. System.out.println(name + " : acquiring lock...");
2. System.out.println(name + " : available Mutex permits now: " + semaphore.availablePermits());
3. semaphore.acquire();
4. System.out.println(name + " : got the permit!");

# try {

16 **for** (**int** i = 1; i <= 5; i++) { 17

1. System.out.println(name + " : is performing operation " + i + ", available Mutex permits : "+ semaphore.availablePermits());
2. // sleep 1 second
3. Thread.sleep(1000);

21

22 }

1. } **finally** {
2. System.out.println(name + " : releasing lock...");
3. semaphore.release();
4. System.out.println(name + " : available Mutex permits now: " + semaphore.availablePermits());

|  |  |  |
| --- | --- | --- |
| 27 |  | } |
| 28 |  | } **catch** (InterruptedException e) { |
| 29 |  | e.printStackTrace(); |
| 30 |  | } |
| 31 | } |  |

32 }

33

1. **public static void main**(String[] args) {
2. System.out.println("Total available Mutex permits : " + semaphore.availablePermits());
3. MyLockerThread t1 = **new** MyLockerThread("A");
4. t1.start();
5. MyLockerThread t2 = **new** MyLockerThread("B");
6. t2.start();
7. MyLockerThread t3 = **new** MyLockerThread("C");
8. t3.start(); 42
9. MyLockerThread t4 = **new** MyLockerThread("D");
10. t4.start(); 45
11. MyLockerThread t5 = **new** MyLockerThread("E");
12. t5.start(); 48
13. MyLockerThread t6 = **new** MyLockerThread("F");
14. t6.start(); 51

52 }

53 }