

Here is the output **before implementing lock**:

```

EXPLORER          OPERATING SYSTEM
  Bank.class
  Bank.java
  Main.class
  Main.java

Bank.java > ...
1  public class Bank {
2      private int balance = 0;
3
4      public void deposit() {
5          balance += 100;
6      }
7
8      public void withdraw() {
9          balance -= 100;
10     }
11
12     public int getValue() {
13         return balance;
14     }
15
16     public void runThread(String threadName) {
17         deposit();
18         System.out.println("Value for Thread after deposit " + threadName + ": " + getValue());
19         withdraw();
20         System.out.println("Value for Thread after withdraw " + threadName + ": " + getValue());
21     }
22 }
23

OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  PLAYWRIGHT
> < TERMINAL
ponlork@Ponlorks-MacBook-Air Operating System % cd "/Users/ponlork/Documents/Operating System/" && javac Bank.java && java Bank
Or a Java application class must extend javax.application.Application
ponlork@Ponlorks-MacBook-Air Operating System % cd "/Users/ponlork/Documents/Operating System/" && javac Main.java && java Main
Value for Thread after deposit Thread2: 200
Value for Thread after deposit Thread1: 100
Value for Thread after withdraw Thread1: 100
Value for Thread after deposit Thread3: 300
Value for Thread after withdraw Thread3: 0
Value for Thread after withdraw Thread2: 200
ponlork@Ponlorks-MacBook-Air Operating System %

```

Ln 23, Col 1 Spaces: 4 UTF-8 LF { } Java Go Live Prettier

Here is the output **before implementing lock**:

```

EXPLORER          OPERATING SYSTEM
  Bank.class
  Bank.java
  Main.class
  Main.java

Main.java > ...
1  public class Main {
2      public static void main(String[] args) {
3          Bank bank = new Bank();
4
5          // Create threads using Runnable
6          Thread t1 = new Thread(() -> bank.runThread(threadName, "Thread1"));
7          Thread t2 = new Thread(() -> bank.runThread(threadName, "Thread2"));
8          Thread t3 = new Thread(() -> bank.runThread(threadName, "Thread3"));
9
10         // Start threads
11         t1.start();
12         t2.start();
13         t3.start();
14
15         // Wait for threads to finish
16         try {
17             t1.join();
18             t2.join();
19             t3.join();
20         } catch (InterruptedException e) {
21             e.printStackTrace();
22         }
23     }
24 }

OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  PLAYWRIGHT
> < TERMINAL
Value for Thread after withdraw Thread3: 200
Value for Thread after withdraw Thread2: 0
ponlork@Ponlorks-MacBook-Air Operating System % cd "/Users/ponlork/Documents/Operating System/" && javac Main.java && java Main
Value for Thread after deposit Thread3: 300
Value for Thread after deposit Thread2: 200
Value for Thread after withdraw Thread3: 200
Value for Thread after deposit Thread1: 100
Value for Thread after withdraw Thread2: 100
Value for Thread after withdraw Thread1: 0
ponlork@Ponlorks-MacBook-Air Operating System %

```

Ln 25, Col 1 Spaces: 4 UTF-8 LF { } Java Go Live Prettier