**1.Pattern**

<https://www.runoob.com/java/java-regular-expressions.html>

**2.Synchronized**

1. Interrupted相关

|  |
| --- |
| **public class** MyThread **extends** Thread{ } |
| **public class** isInterruptedDemo {  **public static void** main(String[] args) {  MyThread myThread = **new** MyThread();  System.***out***.println(**"main myThread.start before , Thread.interrupted() = "** + Thread.*interrupted*());  myThread.start();  *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 给 myThread 线程打上 中断标记 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **myThread.interrupt();**  **//给当前线程（main）打上 中断标记** *//Thread.currentThread().interrupt();   /\*  https://blog.csdn.net/zhuyong7/article/details/80852884  \* Thread.interrupted(): 返回当前线程是否被打上中断标记  \* 第一个红框中断的线程是我们自己创建的myThread线程，我调用的interrupted()，  \* 由上面源码可知是判断当前线程的中断状态，当前线程是main线程，我根本没有中断过main线程，所以2次调用均返回“false”  \* \*/* System.***out***.println(**"[1] main myThread.interrupt() after , Thread.interrupted() = "** + **Thread.*interrupted*()**);  System.***out***.println(**"[2] main myThread.interrupt() after , Thread.interrupted() = "** + **Thread.*interrupted*()**);  } } |
| Screenshot from 2020-12-01 19-11-26 |

|  |
| --- |
| **public class** isInterruptedDemo2 {  **public static void** main(String[] args) {  MyThread myThread = **new** MyThread();  System.***out***.println(**"main myThread.start before , Thread.interrupted() = "** + Thread.*interrupted*());  myThread.start();  *//给 myThread 线程打上 中断标记  //myThread.interrupt();  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 给当前线程（main）打上 中断标记 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **Thread.currentThread().interrupt();**   */\*  https://blog.csdn.net/zhuyong7/article/details/80852884  \* Thread.interrupted(): 返回当前线程是否被打上中断标记  \* 第一个红框中断的线程是当前线程 main，interrupt 给当前线程打上中断标记  \* 由上面源码可知是判断当前线程的中断状态，当前线程是main线程，  所以第 1 次调用均返回 “true”，*  ***由于Thread.interrupted()在执行完毕后会清除中断状态（即将中断状态置为false），所以第 2 次调用均返回 “false”*** *\* \*/* System.***out***.println(**"[1] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*interrupted*()**);  System.***out***.println(**"[2] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*interrupted*()**);  } } |
| Screenshot from 2020-12-01 20-33-53 |

|  |
| --- |
| **public class** isInterruptedDemo3 {  **public static void** main(String[] args) {  MyThread myThread = **new** MyThread();  System.***out***.println(**"main myThread.start before , Thread.interrupted() = "** + Thread.*interrupted*());  myThread.start();  *// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 给 myThread 线程打上 中断标记 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **myThread.interrupt();**  *//给当前线程（main）打上 中断标记  //Thread.currentThread().interrupt();   /\*  https://blog.csdn.net/zhuyong7/article/details/80852884  \* Thread.interrupted(): 返回当前线程是否被打上中断标记  \* 第一个红框中断的线程是我们自己创建的myThread线程，我调用的interrupted()，  \* 由上面源码可知是判断myThread线程的中断状态，  所以第 1 次调用均返回 “true”，由于Thread.interrupted()在执行完毕后不会清除中断状态  所以第 2 次调用均返回 “true”  \* \*/* System.***out***.println(**"[1] main myThread.interrupt() after , myThread.isInterrupted() = "** + **myThread.isInterrupted()**);  System.***out***.println(**"[2] main myThread.interrupt() after , myThread.isInterrupted() = "** + **myThread.isInterrupted()**);  } } |
| Screenshot from 2020-12-01 20-35-46 |

|  |
| --- |
| **public class** isInterruptedDemo4 {  **public static void** main(String[] args) {  MyThread myThread = **new** MyThread();  System.***out***.println(**"main myThread.start before , Thread.interrupted() = "** + Thread.*interrupted*());  myThread.start();  *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 给 myThread 线程打上 中断标记 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **myThread.interrupt();**  *//给当前线程（main）打上 中断标记  //Thread.currentThread().interrupt();   /\*  https://blog.csdn.net/zhuyong7/article/details/80852884  \* Thread.interrupted(): 返回当前线程是否被打上中断标记  \* 第一个红框中断的线程是当前线程 myThread，interrupt 给myThread 线程打上中断标记  \* 由上面源码可知是判断当前线程的中断状态，当前线程是main线程，  所以两次调用均返回 “false”  \* \*/* System.***out***.println(**"[1] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*currentThread*().isInterrupted()**);  System.***out***.println(**"[2] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*currentThread*().isInterrupted()**);  } } |
| Screenshot from 2020-12-01 20-38-24 |

|  |
| --- |
| **public class** isInterruptedDemo5 {  **public static void** main(String[] args) {  MyThread myThread = **new** MyThread();  System.***out***.println(**"main myThread.start before , Thread.interrupted() = "** + Thread.*interrupted*());  myThread.start();  *//给 myThread 线程打上 中断标记  //myThread.interrupt();  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 给当前线程（main）打上 中断标记 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **Thread.*currentThread*().interrupt();**   */\*  https://blog.csdn.net/zhuyong7/article/details/80852884  \* Thread.interrupted(): 返回当前线程是否被打上中断标记  \* 第一个红框中断的线程是当前线程 main，interrupt 给当前线程打上中断标记  \* 由上面源码可知是判断当前线程的中断状态，当前线程是main线程，  所以第 1 次调用均返回 “true”，由于Thread.currentThread().isInterrupted()在执行完毕后不会清除中断状态  所以第 2 次调用均返回 “true”  \* \*/* System.***out***.println(**"[1] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*currentThread*().isInterrupted()**);  System.***out***.println(**"[2] main Thread.currentThread() after , Thread.interrupted() = "** + **Thread.*currentThread*().isInterrupted()**);  } } |
| Screenshot from 2020-12-01 20-39-30 |