Jstack(console thread dump for java)

Koi bhi developer, when stuck in production, will know the power of jstack,
Basically jstack ek tool hai jo runtime pe java process ka memory and cpu utilization show karta hai, say for example agar koi process bahut time le raha hai then hum using jstack usko analyse kar saktey hai, best tool for Advance users.

Example

```java
public class Looping {
    public static void main(String args[]){
        while(true)
            System.out.println("Hello");
    }
}
```

Ye bahut jyada cpu time lega, kuki infinite loop ho raha hai, sometime in big code it does happens, so this tool will exactly pinpoint to the location

Example in linux

```
jstack 25247 (process id, in my case its 25247)
```

```
"main" #1 prio=5 os_prio=31 tid=0x00007f8c1e009000 nid=0x2903
runnable [0x000070000d240000]
   java.lang.Thread.State: RUNNABLE
     at java.io.FileOutputStream.writeBytes(Native Method)
     at java.io.FileOutputStream.write(FileOutputStream.java:326)
     at
java.io.BufferedOutputStream.flushBuffer(BufferedOutputStream.java:8
2)
     at
java.io.BufferedOutputStream.flush(BufferedOutputStream.java:140)
     - locked <0x00000006c001ce40> (a java.io.BufferedOutputStream)
     at java.io.PrintStream.write(PrintStream.java:482)
     - locked <0x00000006c000c378> (a java.io.PrintStream)
     at sun.nio.cs.StreamEncoder.writeBytes(StreamEncoder.java:221)
     at
sun.nio.cs.StreamEncoder.implFlushBuffer(StreamEncoder.java:291)
     at sun.nio.cs.StreamEncoder.flushBuffer(StreamEncoder.java:104)
     - locked <0x00000006c000c338> (a java.io.OutputStreamWriter)
     at
java.io.OutputStreamWriter.flushBuffer(OutputStreamWriter.java:185)
     at java.io.PrintStream.write(PrintStream.java:527)
     - eliminated <0x00000006c000c378> (a java.io.PrintStream)
     at java.io.PrintStream.print(PrintStream.java:669)
     at java.io.PrintStream.println(PrintStream.java:806)
     - locked <0x00000006c000c378> (a java.io.PrintStream)
     at com.Looping.main(Looping.java:5).  // pinpoint the problem
```

Jaisey ki we can see that using this humlog problem tak directly
pahuch saktey hai and then we can resolve this.

Visually agar analyse karana hai then jvisualvm best tool hai.