## 本文由 <u>简悦SimpRead</u> 转码,原文地址 hollischuang.gitee.io

## Description

在学习过了前面几篇文章之后,相信很多人对于 Java 中的多线程都有了一定的了解,相信很多读者已经尝试过中写一些多线程的代码了。

但是我之前面试过很多人,很多人都知道多线程怎么实现,但是却不知道如何调试多线程的代码,这篇文章我们来介绍下如何调试多线程的代码。

首先我们写一个多线程的例子,使用继承 Runnable 接口的方式定义多个线程,并启动执行。

```
/**
* @author Hollis
public class MultiThreadDebug {
    public static void main(String[] args) {
       MyThread myThread = new MyThread();
       Thread thread1 = new Thread(myThread, "thread 1");
        Thread thread2 = new Thread(myThread, "thread 2");
        Thread thread3 = new Thread(myThread, "thread 3");
       thread1.start();
       thread2.start();
       thread3.start();
   }
}
class MyThread implements Runnable {
   @Override
    public void run() {
       System.out.println(Thread.currentThread().getName() + "
running");
我们尝试在代码中设置断点,并使用debug模式启动。
```

```
public class MultiThreadDebug {
                     public static void main(String[] args) {
   MyThread myThread = new MyThread();
                           Thread thread1 = new Thread(myThread, name: "thread 1"); Thread thread2 = new Thread(myThread, name: "thread 2"); Thread thread3 = new Thread(myThread, name: "thread 3");
                           thread1.start();
                           thread2.start();
                            thread3.start();
               class MyThread implements Runnable {
                      public void run() {
                           System.out.println(Thread.currentThread().getName() + " running");
                MvThread > run()
Debug:  MultiThreadDebug
▶ 🥸 "thread 1"@486 in group "main": RUNNING
                                                                                                                               + ▼ = this = {MyThread@488}
  ▶ I "Common-Cleaner"@419 in group "InnocuousThreadGroup": WAIT
▶ I "Finalizer"@490: WAIT
▶ I "main"@1 in group "main": RUNNING
                                                                                                                                         (1) Class has no fields
▶ I "Reference Handler"@489: RUNNING

▶ I "Signal Dispatcher"@491: RUNNING

▶ I "thread 2"@492 in group "main": RUNNING
                                                                                                                              •
       ∰ "thread 3"@493 in group "main": RUNNING
```

如题,程序启动后,会进入一个线程的断点中,我们尝试看一下当前是哪个线程:

```
Expression:

Thread.currentThread().getName()

Result:

▼ oo result = "thread 1"

► 1 value = {byte[8]@500}

1 coder = 0

1 hash = 0
```

发现是 thread 1 进入了断点。接着,我们尝试让代码继续执行,代码就直接结束运行,并且控制台打印如下:

```
Connected to the target VM, address: '127.0.0.1:55768', transport:
'socket'
thread 3 running
Disconnected from the target VM, address: '127.0.0.1:55768', transport:
'socket'
thread 2 running
thread 1 running

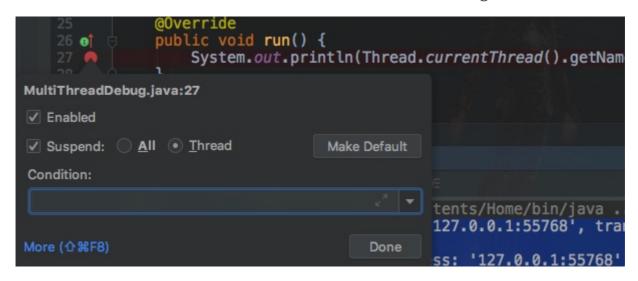
Process finished with exit code 0
```

如果我们多次执行这个代码,就会发现,每一次打印的结果都不一样,三个线程的输出顺序是随机的,并且每一次 debug 只会进入到一个线程的执行。

每次执行结果随即是因为不一定哪个线程可以先获得 CPU 时间片。

那么,我们怎么才能让每一个线程的执行都能被 debug 呢?如何在多线程中进行 debug 排查问题呢?

其实,在 IDEA 中有一个设置,那就是当我们在断点处单击鼠标右键就会弹出一个设置对话框,当我们把其中的 All 修改为 Thread 之后,尝试重新执行 debug 代码。



重新执行之后,就可以发现,每一个线程都会进入到断点当中了。