


性能调优-网易最佳实践-上课笔记2

1 堆外内存（直接内存）

-Xmx64m 堆最大64M

```
Exception in thread "main" java.lang.OutOfMemoryError: Direct buffer memory
    at java.nio.Bits.reserveMemory(Bits.java:693)
    at java.nio.DirectByteBuffer.<init>(DirectByteBuffer.java:123)
    at java.nio.ByteBuffer.allocateDirect(ByteBuffer.java:311)
    at com.study.jvm.OutOfMemoryDemo2.main(OutOfMemoryDemo2.java:17)
```



1、堆外内存与堆的区别

-verbose:gc -XX:+PrintGCDetails

2 堆外内存的上限是多大？

- 未显式指定上限：默认是基本= Xmx
- 显式指定： -XX:MaxDirectMemorySize=40m

jmap -heap pid -Xmx4G

```
Window Help
omdemo/pom.xml Cpu100Demo2... FullGC Demo1... OutOfMemory... Bits.class DirectByteB... ByteBuffer.c... Cpu100Demo4...
700     }
701 }
702
703 private static boolean tryReserveMemory(long size, int cap) {
704     // -XX:MaxDirectMemorySize limits the total capacity rather than the
705     // actual memory usage, which will differ when buffers are page
706     // aligned.
707     long totalCap;
708     while (cap <= maxMemory - (totalCap = totalCapacity.get())) {
709         if (totalCapacity.compareAndSet(totalCap, totalCap + cap)) {
710             reservedMemory.addAndGet(size);
711             count.incrementAndGet();
712             return true;
713         }
714     }
715 }
716
717 return false;
718 }
719
```

Red arrows in the image point to the following lines: 703 (method signature), 708 (while loop condition), 709 (if statement), 710 (reservedMemory.addAndGet), 711 (count.incrementAndGet), and 712 (return true).

3 堆外内存如何回收?

FullGC时, 回收引用堆外内存的堆中对象时, 顺带回收对应的堆外内存。

堆内: 空间不够; system.gc();

4 堆外内存要等待堆得GC时才会回收, 合适吗?

不合适。

怎么办?

被动等, 变主动呼唤: 当堆外空间不够时, 主动调用System.gc();

4 堆外内存OOM分析思路

OOM是空间不够:

- 1、确认堆外空间大小, 是否够用 (不够, 如果内存够, 调大)
- 2、确认系统内存空间够
- 3、确认空间为什么不回收, 是否禁用了显式GC -XX:+DisableExplicitGC
- 4、如果上面3条都OK, 确认空间中都放了什么, 谁持有它

ByteBuffer CharBuffer dump heap

堆外内存分析工具: google-perftools

2 线程-CPU

系统慢、假死问题排查：

- 1、系统资源耗用 top
- 2、网络情况
- 3、应用自身

线程死锁

jstack 线程栈快照工具

线程等待

线程过多

耗系统内存、堆内存

```
Thread t = new Thread(); // 堆
```

```
t.start();
```

死亡 t.state

操作系统线程

线程有自己的线程栈空间 --》调用栈 栈帧 --》一次方法调用 系统内存 -Xss 1M

```
java -Xmx64m com.study.jvm.xxxx
```

操作系统需频繁切换

cpu 100%

cpu占用100%

nid native id

```
top h q
```

```
top -H -p pid 线程
```

jstack pid > stack.log 得到线程快照

线程ID == nid

