图文 082、一个关键问题：如何在JVM内存溢出的时候自动dur

详情 评论

一个关键问题：

如何在JVM内存溢出的时候自动dump内存快照？

石杉老哥重磅力作：《互联网java工程师面试突击》（第3季）**【强烈推荐】**：

全程真题驱动，精研Java面试中**6大专题的高频考点**，从面试官的角度剖析面试

（点击下方蓝字试听）

《互联网Java工程师面试突击》（第3季）

1、前文回顾

上一篇文章已经给大家介绍了线上运行的系统如果发生OOM异常如何监控到

要不是主动监控，要不是被动发现，总之无论如何，你的系统在经过你的GC优化后，平时也许跑的很正常，结果突然某一天就OOM了，那你总是会知道的。

今天的文章就给大家讲一下，假设你知道了自己的系统发生OOM了，应该怎么来处理呢?

2、解决OOM问题的一个初步思路

首先第一个问题，假设发生OOM了，必然说明系统中某个区域的对象太多了，塞满了那个区域，而且一定是无法回收掉那些对象，最终才会导致内存溢出的。

既然是这个思路，要解决OOM的话，首先就得知道到底是什么对象太多了最终导致OOM的?

所以你想知道什么对象太多导致OOM的，就必须得有一份JVM发生OOM时的dump内存快照

只要有了那个dump内存快照，你就可以用之前介绍过的MAT之类的工具瞬间分析得到什么对象太多了。

那么现在一个关键问题来了，到底怎么做才可以在JVM内存溢出的时候自动dump出来一份内存快照呢?

3、在OOM的时候自动dump内存快照

看到这里，大家必须得对一个事情有个概念，大家可以思考一下，假设JVM发生OOM了，你觉得JVM是完全来不及处理然后突然进程就没了吗？也就是JVM是看起来非常突然的自己无法控制的就挂掉了吗?

其实不是的，只要之前对JVM OOM的各种情况原理有一定的了解，就会知道JVM本身在发生OOM之前都会尽可能的去进行GC腾出来一些内存空间

如果GC后还是没有空间，放不下对象， 才会触发内存溢出的。

所以JVM自己对OOM情况的发生是完全有把控权的，他知道什么时候会触发OOM，也是他自己感觉不行的时候才会去触发的

所以OOM的发生并不是大家想的那样，突然之间内存太多了，JVM自己都没反应过来就直接崩溃了，并非如此。

因此JVM如果知道要发生OOM了，此时完全可以让他做点事情

什么事情呢?

我们可以让他在OOM时dump一份内存快照，事后我们只要分析这个内存快照，一下就可以知道是哪些可恶的对象占用了所有的内存，并且还无法释放。

此时你就需要在JVM的启动参数中加入如下的一些参数:

```
-XX:+HeapDumpOnOutOfMemoryError
-XX:HeapDumpPath=/usr/local/app/oom
```

第一个参数意思是在OOM的时候自动dump内存快照出来，第二个参数是说把内存快照放到哪儿去

只要你加入了这两个参数，在JVM OOM崩溃的时候，无论你是立马主动收到一个报警，还是被动让客服通知了你，立马就可以去找OOM时候的内存快照了。

4、迄今为止我们可以得出的一份JVM参数模板

在学习完这篇文章之后，我们最常用的一些JVM参数已经全部学习完了，而且可以总结一份JVM参数模板出来供大家进行参考，大家未来对自己的系统只要根据自己的情况去调整就可以了:

```
“-Xms4096M -Xmx4096M -Xmn3072M -Xss1M  -XX:MetaspaceSize=256M -XX:MaxMetaspaceSize=256M -
XX:+UseParNewGC -XX:+UseConcMarkSweepGC -XX:CMSInitiatingOccupancyFaction=92 -
XX:+UseCMSCompactAtFullCollection -XX:CMSFullGCsBeforeCompaction=0 -XX:+CMSParallelInitialMarkEnabled -
```

XX:+CMSScavengeBeforeRemark -XX:+DisableExplicitGC -XX:+PrintGCDetails -Xloggc:gc.log -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/usr/local/app/oom”

这份JVM参数模板基本上涵盖了所有你需要的一些参数

首先是各个内存区域的大小分配，这个是需要你精心调优的

其次是两种垃圾回收器的指定，接着是一些常规性的CMS垃圾回收的参数，可以帮助优化偶尔发生的Full GC性能。

最重要的，就是平时要打印出来GC日志，GC日志可以配合你用jstat工具分析GC频率和性能的时候用，jstat可以分析出来GC的频率，但是对每次具体的GC情况，可以结合GC日志来看。

还有就是在OOM的时候需要自动dump内存快照，这样即使突然发生OOM，你只要得知了这个事，立马就可以去分析内存快照了。

接下来，**我们会用三篇文章来依次带着大家用MAT工具来分析Metaspace、栈内存、堆内存三种内存溢出下的内存快照文件**

同时结合GC日志，让大家更加清晰的理解每次发生内存溢出时候的具体过程和原理，以及解决内存溢出的排查过程。

End

狸猫技术窝精品专栏及课程推荐：
《从零开始带你成为消息中间件实战高手》
《21天互联网Java进阶面试训练营》（分布式篇）（现更名为：**互联网java工程师面试突击第2季**）
《互联网Java工程师面试突击》（第1季）

互联网Java面试突击第三季相关问题QA：

如何提问：每篇文章都有评论区，大家可以尽情在评论区留言提问，我都会逐一答疑

（ps：评论区还精选了一些小伙伴对**专栏每日思考题的作答**，有的答案真的非常好！大家可以通过看别人的思路，启发一下自己，从而加深理解**)**

如何加群：购买了狸猫技术窝专栏的小伙伴都可以加入狸猫技术交流群，具体加群方式，请参见**专栏目录菜单下的文档**：《付费用户如何加群？》 **（购买后可见）**

（群里有不少**一二线互联网大厂的助教**，大家可以一起讨论交流各种技术）