性能优化-网易最佳实践-上课笔记

1 JVM参数调优

目的：事前预防，事前准备

网易云课堂：

云课堂前后台应用JVM调优相关参数（建议值）：

前台（4核8G内存） JDK1.8

> -server -Xms4g -Xmx4g -Xmn2g -Xss768k -XX:MetaspaceSize=256m -XX:MaxMetaspaceSize=256m -XX:+UseConcMarkSweepGC -XX:+UseParNewGC -XX:+CMSClassUnloadingEnabled -XX:+DisableExplicitGC -XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitiatingOccupancyFraction=80 -verbose:gc -XX:+PrintGCDetails -Xloggc:${CATALINA_BASE}/logs/gc.log -XX:+PrintGCDateStamps -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=${CATALINA_BASE}/logs

-server 服务器模式

-Xms4g -Xmx4g 最小堆大小，最大堆大小

一样大小目的：避免反复向操作系统申请内存。

启动，申请到4G?

一启动就要分配最小堆大小的空间 -XX:+AlwaysPreTouch

-Xmn2g

-Xss768k 1M 可以创建更多的线程

-XX:MetaspaceSize=256m -XX:MaxMetaspaceSize=256m jdk 1.8元数据区 jdk1.7 永久代

-XX:+UseConcMarkSweepGC 老年代gc算法： cms

-XX:+UseParNewGC 年轻代 PraNewGc

-XX:+CMSClassUnloadingEnabled

-XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitiatingOccupancyFraction=80 默认 65

-XX:+DisableExplicitGC 禁用显式gc System.gc();

---- 事情准备

-verbose:gc -XX:+PrintGCDetails -Xloggc:${CATALINA_BASE}/logs/gc.log -XX:+PrintGCDateStamps

开启gc日志，文件记录

-XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=${CATALINA_BASE}/logs

OOM 开启堆的oom时的快照存储。

2 GC

FullGC stop the world

两个方面：

- 过多/频繁的FullGC 卡顿
- oom异常排查

过多/频繁的FullGC

1 什么情况下会发生 GC?

2 GC日志查看

-verbose:gc -XX:+PrintGCDetails -Xloggc:gclog.log -XX:+PrintGCDateStamps

3 FullGC的危害

4 如何来看

跟日志

jstat

5 如何查看堆的使用情况

jmap

6 FullGC 线上问题

创建对象分配不到足够空间；

System.gc();

强制禁用显式gc -XX:+DisableExplicitGC

jxl 为什么要 System.gc()?

大对象变量 。

{

……

var1 = null; // Help GC //大对象

……

}

oom异常排查

事前准备，得到快照

-XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=${CATALINA_BASE}/logs

快照分析工具：

eclipse mat： MemoryAnalyzer-1.8.1.20180910-win32.win32.x86_64

3 监控工具

Java visualVM

插件获取地址：https://visualvm.github.io/index.html

选用监控工具 zabbix cat