# **常见基本命令**

javac \*.java：编译class文件

java \*：运行class文件

javap –c \*.class：查看字节码

javap –verbose \*.class：查看详细的类加载器信息

# **jps—JVM Process Status Tool**

**显示指定系统所有的HotSpot虚拟机进程**

## jps ：列出Java程序进程ID和Main函数名称

## jps -q ：只输出进程ID

## jps -m ：输出传递给Java进程（主函数）的参数

## jps -l ：输出主函数的完整路径

## jps -v ：显示传递给Java虚拟的参数

# jstat—JVM Statistics Monitoring Tool

显示本地或远程虚拟机进程的类加载、内存、垃圾回收、JIT编译等运行时的数据；监视VM内存内的各种堆和非堆的大小及其内存使用量。

命令格式：jstat -<options> [-t] [-h<lines>] <vmid> [<interval> [<count>]]

options：由以下值构成

## -class：显示ClassLoader的相关信息，

Load*ed:加载class的数量*

Bytes：所占用空间大小

Unloaded：未加载数量

Bytes:未加载占用空间

Time：时间

## -compiler：显示JIT编译的相关信息

Compiled：编译数量。

Failed：失败数量

Invalid：不可用数量

Time：时间

FailedType：失败类型

FailedMethod：失败的方法

## -gc：显示与GC相关信息

S0C：第一个幸存区的大小

S1C：第二个幸存区的大小

S0U：第一个幸存区的使用大小

S1U：第二个幸存区的使用大小

EC：伊甸园区的大小

EU：伊甸园区的使用大小

OC：老年代大小

OU：老年代使用大小

MC：方法区大小

MU：方法区使用大小

CCSC:压缩类空间大小

CCSU:压缩类空间使用大小

YGC：年轻代垃圾回收次数

YGCT：年轻代垃圾回收消耗时间

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

## -gccapacity：显示各个代的容量和使用情况

NGCMN：新生代最小容量

NGCMX：新生代最大容量

NGC：当前新生代容量

S0C：第一个幸存区大小

S1C：第二个幸存区的大小

EC：伊甸园区的大小

OGCMN：老年代最小容量

OGCMX：老年代最大容量

OGC：当前老年代大小

OC:当前老年代大小

MCMN:最小元数据容量

MCMX：最大元数据容量

MC：当前元数据空间大小

CCSMN：最小压缩类空间大小

CCSMX：最大压缩类空间大小

CCSC：当前压缩类空间大小

YGC：年轻代gc次数

FGC：老年代GC次数

## -gcnew：显示新生代信息

S0C：第一个幸存区大小

S1C：第二个幸存区的大小

S0U：第一个幸存区的使用大小

S1U：第二个幸存区的使用大小

TT:对象在新生代存活的次数

MTT:对象在新生代存活的最大次数

DSS:期望的幸存区大小

EC：伊甸园区的大小

EU：伊甸园区的使用大小

YGC：年轻代垃圾回收次数

YGCT：年轻代垃圾回收消耗时间

## -gcnewcapacity：显示新生代大小和使用情况

NGCMN：新生代最小容量

NGCMX：新生代最大容量

NGC：当前新生代容量

S0CMX：最大幸存1区大小

S0C：当前幸存1区大小

S1CMX：最大幸存2区大小

S1C：当前幸存2区大小

ECMX：最大伊甸园区大小

EC：当前伊甸园区大小

YGC：年轻代垃圾回收次数

FGC：老年代回收次数

## -gcold：显示老年代信息

MC：方法区大小

MU：方法区使用大小

CCSC:压缩类空间大小

CCSU:压缩类空间使用大小

OC：老年代大小

OU：老年代使用大小

YGC：年轻代垃圾回收次数

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

## -gcoldcapacity：显示老年代大小

OGCMN：老年代最小容量

OGCMX：老年代最大容量

OGC：当前老年代大小

OC：老年代大小

YGC：年轻代垃圾回收次数

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

## -gcpermcapacity：显示永久代大小

MCMN: 最小元数据容量

MCMX：最大元数据容量

MC：当前元数据空间大小

CCSMN：最小压缩类空间大小

CCSMX：最大压缩类空间大小

CCSC：当前压缩类空间大小

YGC：年轻代垃圾回收次数

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

## -gccause：显示垃圾收集相关信息（同-gcutil），同时显示最后一次或当前正在发生的垃圾收集的诱发原因；-gcutil：显示垃圾收集信息

S0：幸存1区当前使用比例

S1：幸存2区当前使用比例

E：伊甸园区使用比例

O：老年代使用比例

M：元数据区使用比例

CCS：压缩使用比例

YGC：年轻代垃圾回收次数

FGC：老年代垃圾回收次数

FGCT：老年代垃圾回收消耗时间

GCT：垃圾回收消耗总时间

## -printcompilation：输出JIT编译的方法信息

Compiled：最近编译方法的数量

Size：最近编译方法的字节码数量

Type：最近编译方法的编译类型。

Method：方法名标识。

## -t：在输出信息前加上一个Timestamp列，显示程序的运行时间

## -h：可以在周期性数据输出后，输出多少行数据后，跟着一个表头信息

## interval：用于指定输出统计数据的周期，单位为毫秒

## count：用于指定一个输出多少次数据

jstat -class -t 6864 1000 2  输出ClassLoader信息

jstat -compiler -t 6864 查看JIT编译信息

Compiled：编译任务执行次数

Failed：编译失败次数

Invalid：编译不可用次数

Time：编译总耗时

FailedType：最后一次编译失败类型

FailedMethod：最后一次编译失败的类名和方法名

jstat -gc 6864  GC相关堆信息

S0C,S1C：S0和S1（from区）大小 KB

S0U,S1U：S0和S1（from区）已使用大小 KB

EC,OC,PC：eden，old，perm大小 KB

EU,OU,PU：eden，old，perm已使用大小 KB

YGC,FGC：新生代GC，FullGC次数

YGCT,FGCT：新生代GC，FullGC耗时

GCT：GC总耗时

jstat -gccapacity 6864 ：查看各个代大小

NGCMN,NGC,OGCMN,PGCMN:新，当前新生代，老，持的最小值 KB

NGCMX,OGCMX,PGCMX:新，老，持的最大值KB

jstat -gccause 6864 显示GC原因

LGCC：上次GC原因

GCC：这次GC原因

jstat -gcnew 6864 ：查看新生代详细参数

TT：新生代晋升到老年代年龄

MTT：新生代晋升到老年代年龄最大值

DSS：所需survivor大小

jstat -gcnewcapacity 6864：输出新生代各区大小

S0CMX：S0最大值KB

S1CMX：S1最大值KB

ECMX：eden区最大值 KB

jstat -gcutil 6864： 查看相关百分比

S0,S1,E,O,P：s0，s1，eden，old，perm使用的百分比

# **jinfo—JVM Configuration Info Java**

显示并修改虚拟机各项参数。

命令格式：jinfo [option] <pid>

-flag <name> to print the value of the named VM flag

-flag [+|-]<name> to enable or disable the named VM flag

-flag <name>=<value> to set the named VM flag to the given value

-flags to print VM flags

-sysprops to print Java system properties

<no option> to print both of the above

-h | -help to print this help message

# jmap—JVM Memory Map for Java

用于生成堆转储快照文件、查询finallize执行队列、Java堆和永久代的详细信息。打印出某个java进程（使用pid）内存内的所有'对象'的情况（如：产生那些对象，及其数量）。

jmap -histo 11512 >c:/MyFile/s.txt：生成PID为11512的Java程序的对象统计信息，并输出到s.txt中

jmap -dump:format=b,file=c:\MyFile\heap.hprof 11652 ：等到Java程序的当前堆快照。然后可以用各种工具分析该文件。

jmap -finalizerinfo 11652 : 观察系统finalizer队列中的对象，一个不恰当的finalize()方法可能导致对象堆积在finalizer队列中，使用上面的参数可以查看堆积在finalizer队列中的对象。

# jhat—JVM Heap Dump Browser

用于分析生成的堆转储快照文件，并且建立web服务，提供查看分析结果。（非常耗机器资源）

jhat c:\MyFile\heap.hprof ：利用jhat分析刚刚jmap输出的堆文件。

# jstack—JVM Stack Trace for Java

用于显示当前虚拟机内每一条线程正在执行的方法堆栈集合，可用于分析线程长时间卡顿的原因。

jstack -l 23612 >c:\MyFile\deadlock.txt : -l 会打印锁的附加信息。

# JCMD

可以用它来导出堆，查看Java进程，导出线程信息，执行GC等

jcmd -l ：列出当前系统中的所有Java虚拟机

jcmd 12720 help ：针对每一个虚拟机，jcmd可以使用help列出它们所支持的命令。