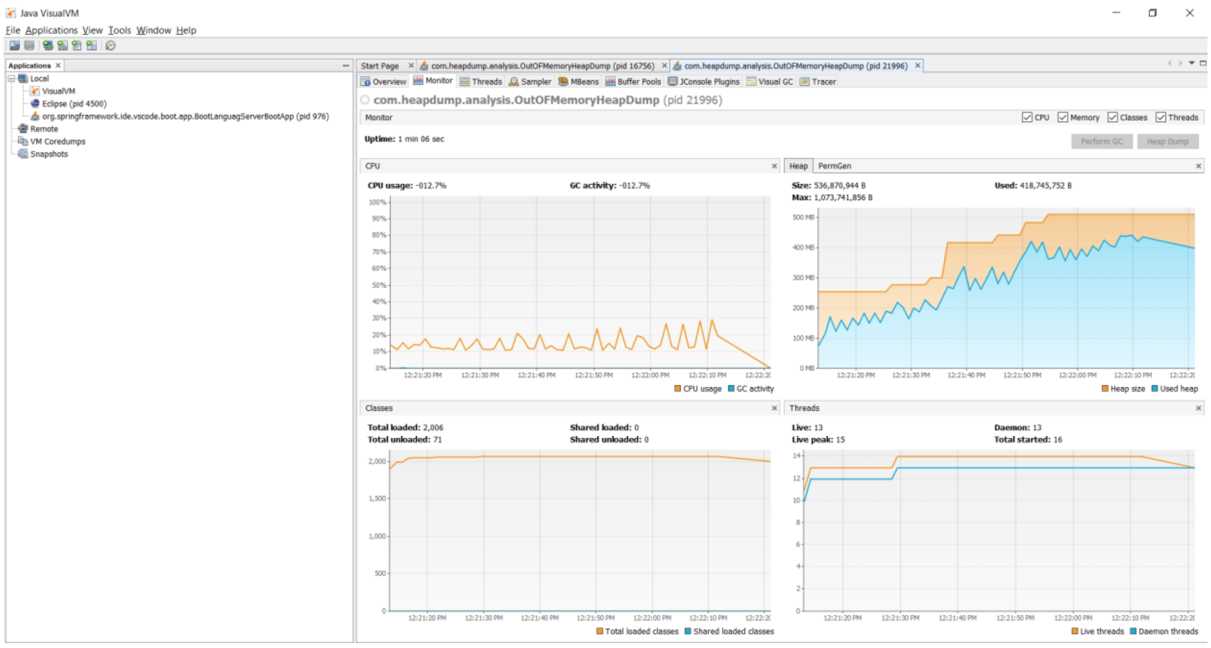
OutOFMemoryHeapDump

HEAP MONITOR:



Uptime: 1 min 06 sec

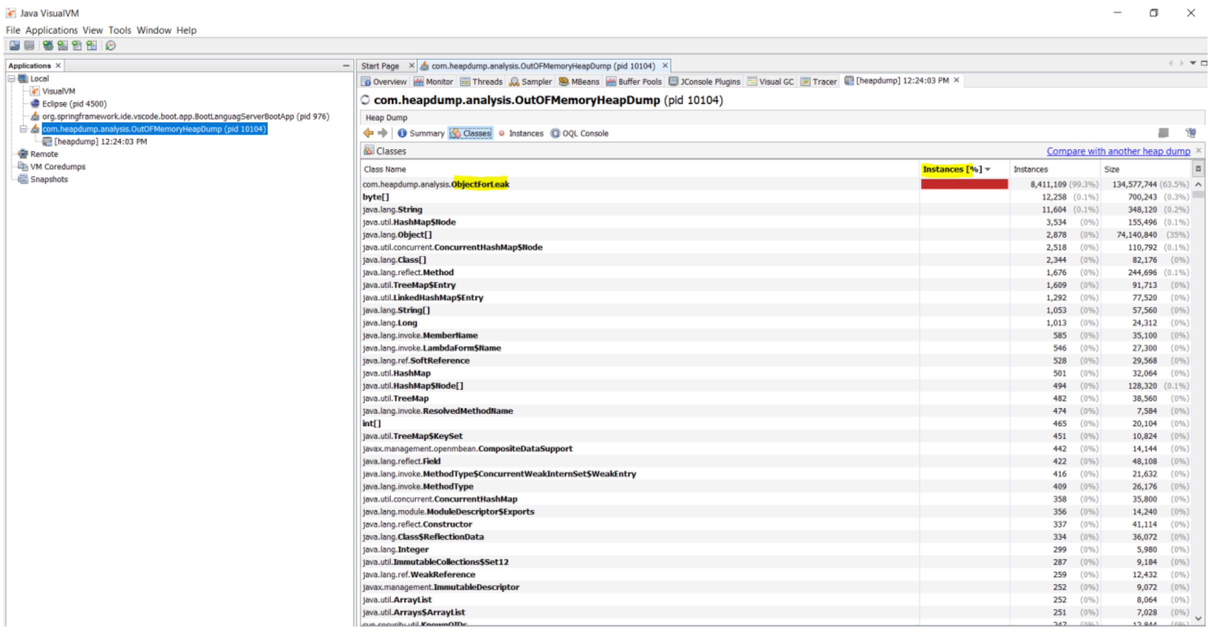
CPU usage: -012.7% GC activity: -012.7%

Size: 536,870,944 B Used: 418,745,752 B
Max: 1,073,741,856 B

Total loaded: 2,006 Shared loaded: 0
Total unloaded: 71 Shared unloaded: 0

Live: 13 Daemon: 13
Live peak: 15 Total started: 16

HEAP DUMP

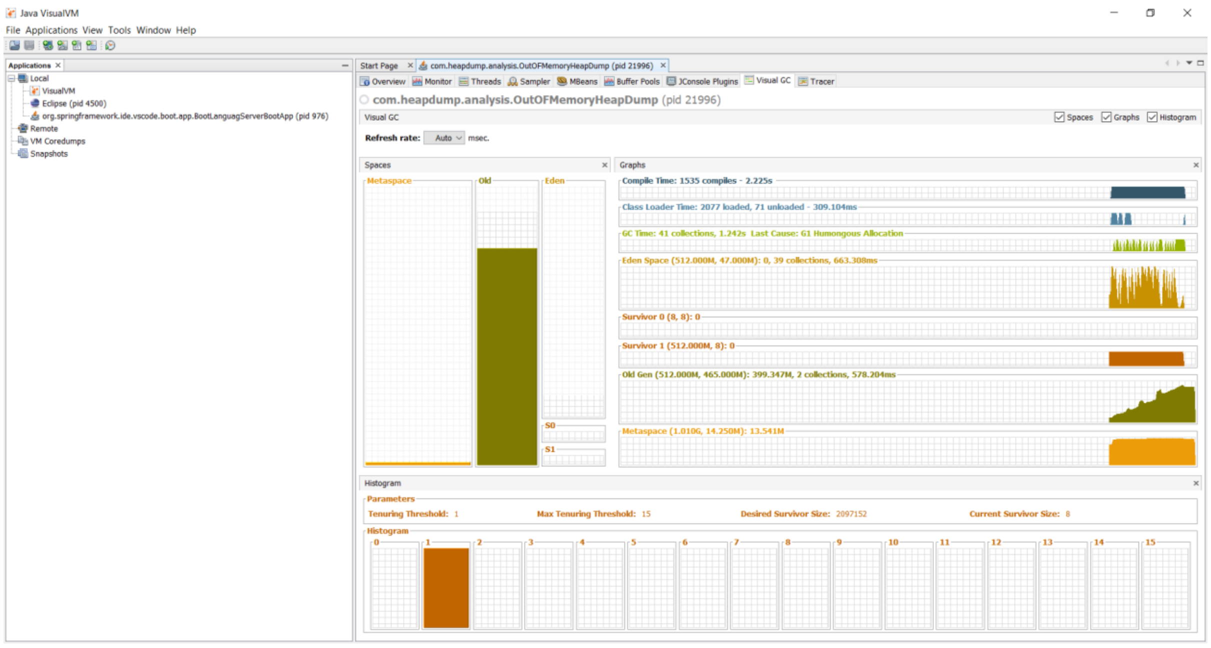


com.heapdump.analysis.OutOFMemoryHeapDump (pid 10104)

Classes

Class Name	Instances [%]	Instances	Size
com.heapdump.analysis.**ObjectForLeak**	🟥	8,411,109 (99.3%)	134,577,744 (63.5%)
byte[]		12,258 (0.1%)	700,243 (0.3%)
java.lang.**String**		11,604 (0.1%)	348,120 (0.2%)
java.util.**HashMap$Node**		3,534 (0%)	155,496 (0.1%)
java.lang.**Object[]**		2,878 (0%)	74,140,840 (35%)
java.util.concurrent.**ConcurrentHashMap$Node**		2,518 (0%)	110,792 (0.1%)
java.lang.**Class[]**		2,344 (0%)	82,176 (0%)
java.lang.reflect.**Method**		1,676 (0%)	244,696 (0.1%)
java.util.**TreeMap$Entry**		1,609 (0%)	91,713 (0%)
java.util.**LinkedHashMap$Entry**		1,292 (0%)	77,520 (0%)
java.lang.**String[]**		1,053 (0%)	57,560 (0%)
java.lang.**Long**		1,013 (0%)	24,312 (0%)
java.lang.invoke.**MemberName**		585 (0%)	35,100 (0%)
java.lang.invoke.**LambdaForm$Name**		546 (0%)	27,300 (0%)
java.lang.ref.**SoftReference**		528 (0%)	29,568 (0%)
java.util.**HashMap**		501 (0%)	32,064 (0%)
java.util.**HashMap$Node[]**		494 (0%)	128,320 (0.1%)
java.util.**TreeMap**		482 (0%)	38,560 (0%)
java.lang.invoke.**ResolvedMethodName**		474 (0%)	7,584 (0%)
int[]		465 (0%)	20,104 (0%)
java.util.**TreeMap$KeySet**		451 (0%)	10,824 (0%)
javax.management.openmbean.**CompositeDataSupport**		442 (0%)	14,144 (0%)
java.lang.reflect.**Field**		422 (0%)	48,108 (0%)
java.lang.invoke.**MethodType$ConcurrentWeakInternSet$WeakEntry**		416 (0%)	21,632 (0%)
java.lang.invoke.**MethodType**		409 (0%)	26,176 (0%)
java.util.concurrent.**ConcurrentHashMap**		358 (0%)	35,800 (0%)
java.lang.module.**ModuleDescriptor$Exports**		356 (0%)	14,240 (0%)
java.lang.reflect.**Constructor**		337 (0%)	41,114 (0%)
java.lang.**Class$ReflectionData**		334 (0%)	36,072 (0%)
java.lang.**Integer**		299 (0%)	5,980 (0%)
java.util.**ImmutableCollections$Set12**		287 (0%)	9,184 (0%)
java.lang.ref.**WeakReference**		259 (0%)	12,432 (0%)
javax.management.**ImmutableDescriptor**		252 (0%)	9,072 (0%)
java.util.**ArrayList**		252 (0%)	8,064 (0%)
java.util.**Arrays$ArrayList**		251 (0%)	7,028 (0%)
sun.security.util.**KnownOIDs**		247 (0%)	12,844 (0%)

VISUAL GC



com.heapdump.analysis.OutOFMemoryHeapDump (pid 21996)

Visual GC

Refresh rate: Auto msec.

Compile Time: 1535 compiles - 2.225s

Class Loader Time: 2077 loaded, 71 unloaded - 309.104ms

GC Time: 41 collections, 1.242s Last Cause: G1 Humongous Allocation

Eden Space (512.000M, 47.000M): 0, 39 collections, 663.308ms

Survivor 0 (8, 8): 0

Survivor 1 (512.000M, 8): 0

Old Gen (512.000M, 465.000M): 399.347M, 2 collections, 578.204ms

Metaspace (1.010G, 14.250M): 13.541M

Histogram

Parameters

Tenuring Threshold: 1 Max Tenuring Threshold: 15 Desired Survivor Size: 2097152 Current Survivor Size: 8

Reason for leak:

An infinite loop of creating new Objects causes this leak.