10 Examples of HotSpot JVM Options in Java

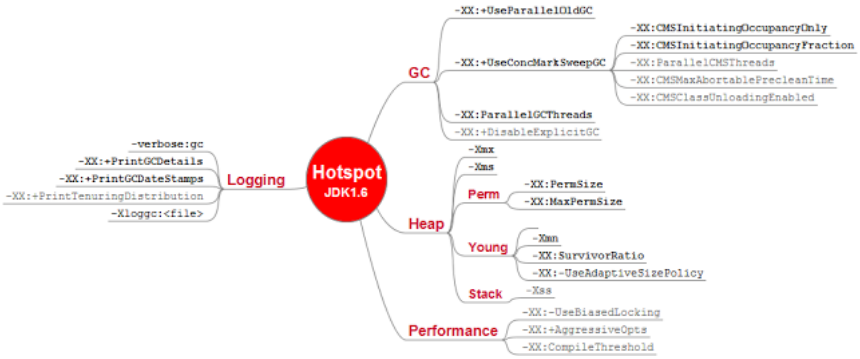
There are hundreds of **JVM parameters** or **JVM Options** exists inside sun JDK and its virtually impossible to keep track of every single JVM option and based on my experience we don't even use most of JVM flags except couple of important JVM option related to java heap size, java options for printing garbage collection details and most likely JVM switches for setting up remote debugging in Java. but there are many other useful category of JVM parameters which you at least like to be familiar even if not intending to use it more frequently. In this article we will see examples of 10 different categories of **JVM parameter** which I found useful and use more frequently than other. I would recommend to get a full knowledge of what does a particular JVM options does by referring official list of JVM options.

JVM parameters in Java

On the basis of how we specify **JVM option it can be divided into two parts**, JVM Options which starts with –X and those which starts with -XX:

1) *JVM Options that begin with -X* are non-standard (thy are not guaranteed to be supported on all JVM implementations), and are subject to change without notice in subsequent releases of the JDK.

2) *JVM Options or parameters which are specified with -XX* are not stable and are not recommended for casual use. These options are subject to change without notice also.



I was thinking about writing post on JVM options when I completed my post on **Java Heap Size** and **Java Garbage Collection** because these are two main area where we see usages of various JVM flags. But it didn't happened even after I covered OutOfMemoryError post which has some **JVM option to solve OutOfMemoryError in Java**. Now I am happy that I have completed this piece of information and its ready to be published. As always I look for your feedback, suggestions and any other JVM flags which I have missed and you guys find useful to share.

Good knowledge of JVM options specially related to GC tuning is important for time critical application e.g. **high volume low latency electronic trading platform** where every micro seconds matter. though getting right combination requires lot of profiling and trial and error and depends heavily on nature of trading application.

Important Points about JVM Options:

1) Boolean JVM options can be turned on with -XX:+ and can be turned off with -XX:-.

2) Numeric JVM Options can be set with -XX:=. Numbers can include 'm' or 'M' for megabytes, 'k' or 'K' for kilobytes, and 'g' or 'G' for gigabytes (for example, 32k is the same as 32768).

3) String JVM options can be set by using -XX:=, and usually used to specify a file, a path, or a list of commands.

The command **java -help** lists the standard options (standard across different JVM implementations) for the Java application launcher. The **command java -X can be used to see the Java application launcher's non-**

Interview Questions
core java interview question (160)
data structure and algorithm (44)
Coding Interview Question (32)
SQL Interview Questions (24)
thread interview questions (20)
database interview questions (18)
servlet interview questions (17)
collections interview questions (15)
spring interview questions (9)
Programming interview question (4)
hibernate interview questions (4)

Translate this blog
Select Language
Powered by Google Translate
Java Tutorials
date and time tutorial (17)
FIX protocol tutorial (16)
java collection tutorial (52)
java IO tutorial (25)
Java JSON tutorial (6)
Java multithreading Tutorials (32)
Java Programming Tutorials (27)
Java xml tutorial (9)

Search This Blog
Search

Follow by Email
Email address...
Submit

Followers
Followers (3988) Next
Follow

Blog Archive
► 2017 (3)
► 2016 (166)
► 2015 (126)
► 2014 (101)
► 2013 (127)
► 2012 (214)
▼ 2011 (135)
► December (27)
▼ November (14)
Database Transaction Tutorial in SQL with Example ...
Great Example of Open Closed Design Principle in J...
Top 10 Struts Interview Question And Answer - J2EE...
XPath Tutorials Examples for Beginners and Java De...
Top 25 Java Collection Framework Interview Questio...

standard (X for extension specific to that JVM) arguments.The -X options are non-standard and subject to change without notice. If you wish to detect which JVM arguments your currently running Java application is using, you can use the ManagementFactory.getRuntimeMXBean().getInputArguments()

Now here is my list of important JVM flags, switches, options or parameters which is most commonly used while running Java applications:

1) JVM memory options related to java heap size

Following three JVM options are used to specify initial and max heap size and thread stack size while running Java programs.

-Xms set initial Java heap size
-Xmx set maximum Java heap size
-Xss> set java thread stack size

2) JVM option to print gc details

-verbose:gc logs garbage collector runs and how long they're taking. I generally use this as my first tool to investigate if GC is a bottleneck for a given application.

-XX:+PrintGCDetails includes the data from -verbose:gc but also adds information about the size of the new generation and more accurate timings.

-XX:-PrintGCTimeStamps Print timestamps at garbage collection.

3) JVM parameters to specify Java Garbage collector

-XX:+UseParallelGC Use parallel garbage collection for scavenges
-XX:-UseConcMarkSweepGC Use concurrent mark-sweep collection for the old generation. (Introduced in 1.4.1)
-XX:-UseSerialGC Use serial garbage collection. (Introduced in 5.0.)

beware when you use GC Parameters if you are working on time critical application e.g. high frequency trading application. As GC is time consuming operation and its desired to create a balance.

4) JVM debug options JVM options for remote debugging

-Xdebug -Xnoagent -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8000
to read more about remote debugging check How to Setup Java remote debugging in Eclipse and 10 Java debugging tips in Eclipse

5) JVM options related to profiling

-Xprof
-Xrunhprof

6) JVM options related to java classpath

Xbootclasspath specifies classpath entries you want loaded without verification. The JVM verifies all classes it loads to ensure they don't try to dereference an object with an int, pop extra entries off the stack or push too many, and so on. This verification is part of the reason why the JVM is very stable, but it's also rather costly, and responsible for a large part of start up delay. Putting classes on the bootclasspath skips this cost, but should only be used when you know the classes have been verified many times before. In JRuby, this reduced startup time by half or more for a simple script. The -**Xbootclasspath** option can be used to either prepend (/p) or append (/a) resources to the bootstrap classpath. You Can read more about Java Classpath in my articles How Classpath Works in Java and How to Solve ClassNotFoundException in Java

7) JVM options to change Perm Gen Size

These JVM optiosn are quite useful to solve java.lang.OutOfMemoryError:Perm Gen Space.

-XX:PermSize and MaxPermSize
-XX:NewRatio=2 Ratio of new/old generation sizes.
-XX:MaxPermSize=64m Size of the Permanent Generation.

8) JVM parameters to trace classloading and unloading

Decorator design Pattern in Java with Example Java...
File permissions in UNIX Linux with Example >> Un...
LDAP Active Directory Authentication in Java Sprin...
Helloworld Example Java : How to run Java Program ...
Result of Recent Google PageRank Update in Novembe...
10 Examples of HotSpot JVM Options in Java
What is Static Variable Class method and keyword i...
10 Examples of tar command in UNIX and Linux
How to override compareTo method in Java - Example...
► October (14)
► September (19)
► August (11)
► July (7)
► June (8)
► May (5)
► April (7)
► March (3)
► February (10)
► January (10)
► 2010 (30)

Pages
Privacy Policy

-XX:+TraceClassLoading and **-XX:+TraceClassUnloading** are two JVM options which we use to print logging information whenever classes loads into JVM or unloads from JVM. These JVM flags are extremely useful if you have any memory leak related to classloader and or suspecting that classes are not unloading or garbage collected.

9) JVM switches related to logging

-XX:+TraceClassLoading and -XX:+TraceClassUnloading print information class loads and unloads. Useful for investigating if you have a class leak or if old classes (like JITed Ruby methods in JRuby) are getting collected or not. You can read more about logging in Java on my post 10 Tips while logging in Java

-XX:+PrintCompilation prints out the name of each Java method Hotspot decides to JIT compile. The list will usually show a bunch of core Java class methods initially, and then turn to methods in your application. In JRuby, it eventually starts to show Ruby methods as well

10) JVM Switches for debugging purpose

-XX:HeapDumpPath=./java_pid.hprof Path to directory or file name for heap dump.
-XX:-PrintConcurrentLocks Print java.util.concurrent locks in Ctrl-Break thread dump.
-XX:-PrintCommandLineFlags Print flags that appeared on the command line.

That's all on JVM Options, I understand its not possible to remember all JVM flags but at-least having an idea of what kind of JVM flags are available is good asset. Image for JVM parameters is from Java tuning and Nutshell. For full list of JVM options you can refer these link from Oracle Java site: Java Hotspot VM Options

Other **Java programming tutorials** you may like

Why character array is better than String for storing password
How to convert String to Date in Java with Example
How to split String in java with Example
Difference between ConcurrentHashMap and Hashtable in Java
Why wait and notify needs to called from Synchronized Context?
Difference between invokeAndWait and InvokeLater in java Swing.
How to fix Race conditions in Java with Example

Posted by Javin Paul

Labels: core java , JVM Internals
Location: United States

21 comments :

Anonymous said...

where are the logs located?? any specific folder?

November 8, 2011 at 6:08 AM

Anonymous said...

Here is another very useful option I think every java programmer needs to know of:
-XX:+HeapDumpOnOutOfMemoryError

This will dump a hprof file on the moment
an outofmemory occurs. This hprof file
can than be debugged with eclipse memory
analyzer.

November 15, 2011 at 7:03 AM

Sandeep said...

Very good points. Also all memory heap options are applicable to eclipse which can be configured in eclipse.ini file.
Extreme Java

November 17, 2011 at 3:48 AM

Javin @ Generics in Java said...

@Anonymous, Thanks for "-XX:+HeapDumpOnOutOfMemoryError" , this is indeed an useful JVM option. Heap dump can help while diagnosing OutOfMemoryError in Java.

November 17, 2011 at 5:22 PM

Javin @ Enum in Java said...

Thanks for your comments Sandeep. Indeed Memory adjustment in Eclipse is quite common because it ran out of Memory while working with large projects.

November 17, 2011 at 5:23 PM

Anonymous said...

Nice JVM Parameters list, I have copied this list of JVM options and kept in my desk for quick reference. Thanks

November 19, 2011 at 7:55 PM

Anonymous said...

There are some JVM options with -D also .What exactly is the differneces between -D vs -X vs -XX

January 4, 2012 at 3:06 AM

Javin @ JDK vs JRE said...

Hi Anonymous, JVM options with -D are system property and you can access them by using System.getProperty("user.timezone"). you can pass any property value in format -Dproperty=value to JVM. -X and -XX are actual JVM options difference is that -XX are non statndard option and may not be supported on all JVM e.g. may supported in HotSpot JVM but may not be in IBM's JVM.

January 4, 2012 at 4:49 AM

John said...

JVM options for memory can also include Setting up PermGen space. like -XX:PermSize for specifying size of PermGen and -XX:MaxPermSize for specifying maximum size of PermGen space. Also JVM options for tuning Young and Old Generatiosn like -Xmn , -XX:SurvivorRation and -XX:UseAdaptiveSizePolicy can be very useful. Anyway turning JVM for performance or memory is continuous task in Java development. let us know how these JVM options performs on your project.

June 7, 2012 at 11:54 PM

Ryan said...

most important JVM option for memory and performance in 64 bit JVM is -XX:+UseCompressedOops, which reduces size of pointers used inside JVM to 32 bit in a 64 bit machine, enabling CPU to cache more data and improve performance. It also helps to reduce GC pauses significantly.

June 12, 2012 at 9:23 PM

Anyul Rivas said...

is it possible for a web app to generate java.lang.OutOfMemoryError:Perm Gen Space by having to many system.out.println() outputs? I checked a heap dump generated on this kind of errors and found too many char[] arrays on my memory instanced.

January 7, 2013 at 4:44 AM

Javin @ CyclicBarrier Example Java said...

Hi Anyul, that's is very unlikely of running out of Perm Gen, I would rather check for classloader leaks. You may wan to check this post on ClassLoader leak on Tomcat and causing PermgGen error

January 7, 2013 at 7:08 AM

Roberto Guerra said...

So what is the stable standard jvm option?

January 29, 2013 at 7:00 AM

Anonymous said...

Very good article....thanks!

March 20, 2013 at 7:31 AM

Zim said...

Thanks Javin! I really had no idea about JVM parameters ever and I recently got a project requirement on Performance Tuning. Seriously, I wouldn't have ever known it was this easy if you wouldn't have written this article!! Thanks a ton! :D

June 11, 2013 at 12:51 PM

Anonymous said...

Hi guys, What is the JVM command to print loading and unloading of class files into memory?

July 10, 2013 at 9:35 PM

aditya said...

HeapDumpPath , JVM dumps the heap only in case of outofmemoryerror for debugging purpose am I correct ? This dump is used for the debugging purpose to find out the memory leak which has caused the out of memory exception.

November 12, 2013 at 7:42 PM

Anonymous said...

I was asked -server , -client VM option in a recent interview. I have never used it but it seems it's used a lot in finance domain. It would be nice if you can include it.

December 14, 2013 at 4:11 PM

DanteVick said...

I am New to this subject and willing to know that where to Implement these JVM commands and how ? will be glad if u can provide the Sample for this

February 4, 2014 at 10:50 PM

techcooker said...

The option "-XX:NewRatio=2" is for New/Old gen in Heap size not for "Perm Gen Size"

March 3, 2014 at 1:54 PM

Ikem said...

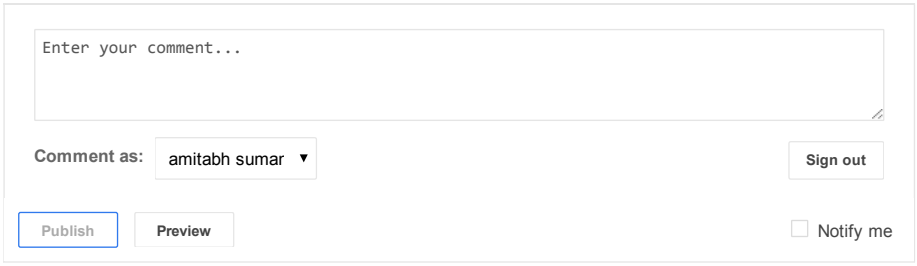
> _they_ are not guaranteed to be supported on all JVM implementations

> -_Xss_ set java thread stack size

Typos.

November 6, 2014 at 5:21 PM

Post a Comment



Newer Post Home Older Post

Subscribe to: Post Comments (Atom)