≡ MENU

Karunsubramanian.com

Resources for transforming IT Operations

One important change in Memory Management in Java 8

by KARUN SUBRAMANIAN *on* JULY 20, 2014

Share Tweet Like 29

Oracle's latest edition for Java – Java 8 was released in March 2014. As usual, tons of new features have been added. There is one major change in the Memory management area that I want to discuss today.

"So long PermGen, Hello Metaspace !!"

Oracle has completely gotten rid of 'PermGen' and replaced it with Metaspace.

What is PermGen ?

Short form for Permanent Generation, PermGen is the memory area in Heap that is used by the JVM to **store class and method objects**. If your application loads lots of classes, PermGen utilization will be high. PermGen also **holds 'interned' Strings**

The size of the PermGen space is configured by the Java command line option **`-XX:MaxPermSize`**

TikTok Ads Self Serve Platform
is now available in India
· Quality Traffic · Pixel Tracking · Target 20+ markets worldwide
Create an Account

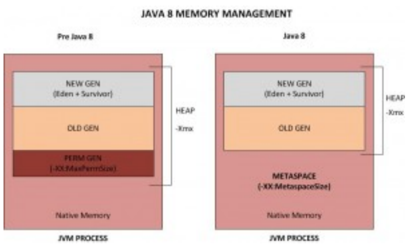
Typically 256 MB should be more than enough of PermGen space for most of the applications

However, It is not unusal to see the error "**`java.lang.OutOfMemoryError: PermGen space`**" if you are loading unusual number of classes.

Gone are the days of OutOfMemory Errors due to PermGen space. **With Java 8, there is NO PermGen**. That's right. So no more OutOfMemory Errors due to PermGen

TikTok Ads Self Serve Platform
is now available in India
· Quality Traffic · Pixel Tracking · Target 20+ markets worldwide
Create an Account

The key difference between PermGen and Metaspace is this: while PermGen is part of Java Heap (Maximum size configured by -Xmx option), **Metaspace is NOT part of Heap.** Rather Metaspace is part of **Native Memory (process memory)** which is only limited by the Host Operating System.


JAVA 8 MEMORY MANAGEMENT
Pre Java 8
Java 8
NEW GEN (Eden + Survivor)
OLD GEN
PERM GEN (-XX:MaxPermSize)
HEAP -Xmx
METASPACE (-XX:MetaspaceSize)
Native Memory
JVM PROCESS

So, what is the significance of this change?

While you will NOT run out of PermGen space anymore (since there is NO PermGen), you may consume excessive Native memory making the total process size large. The issue is, if your application loads lots of classes (and/or interned strings), **you may actually bring down the Entire Server** (not just your application). Why ? Because the native memory is only limited by the Operating System. This means you can literally take up all the memory on the Server. Not good.

It is critical that you add the new option **-XX:MaxMetaspaceSize** which sets the Maximum Metaspace size for your application.

Take better finance decisions

Attend a workshop in Delhi, Mumbai & Bengaluru Or Customise a workshop for your corporate.

KPMG Academy

Note that it is no longer sufficient to just monitor the Heap Size. You must also monitor the Metaspace which you can do by just **keeping an eye on the 'process size'** using your Operating System utilities (Example: 'top' in Unix/Linux, 'Task Manager' in Windows).

Bonus Tip:

You can use the ***jmap*** command to print out Memory statistics of your current pre Java 8 application.

jmap -permstat <PID>

There you have it. With Java 8, PermGen is gone and Metaspace is in. Metaspace is part of Native Memory and NOT part of Java Heap. While this change may not be significant during development stage of the application, it is critical to consider this when going to production as you might not only bring down your application but bring down the entire server if your application eats up excessive Metaspace.

Make sure the Application Administrators and QA are made aware of this significant change and ensure **adequate monitoring** during QA phase (load testing) and in production.

Good Luck

Share Tweet Like 29

{ 10 comments… add one }

Michael Lightfoot November 20, 2017, 10:44 pm

Corrections:
PermGen is not part of the HEAP and is not controlled by Garbage collection
The maximum memory for Metaspace is controlled by -XX:MaxMetaspaceSize

REPLY LINK

Karun Subramanian November 25, 2017, 11:46 am

Hi Michael, thanks for the corrections.
Good point on the -XX:MaxMetaspaceSize. I've updated the blog post.
On Permgen being part of Java Heap, there are contradicting documentation even in Java official website. In my experience, I've also seen changing permgen resulting in changes in Heap utilization. In any case, we don't have to worry about Permgen going forward. 🙂
Thanks.

REPLY LINK

palacsint December 1, 2017, 3:09 pm

You might want to update the figure too 🙂