How to deal with "java.lang.OutOfMemoryError: Java heap space" error?

Asked 16 years, 3 months ago Modified 9 months ago

Viewed 2.4m times

591

I am writing a client-side **Swing** application (graphical font designer) on **Java 5**. Recently, I am running into `java.lang.OutOfMemoryError: Java heap space` error because I am not being conservative on memory usage. The user can open unlimited number of files, and the program keeps the opened objects in the memory. After a quick research I found Ergonomics in the 5.0 Java Virtual Machine and others saying on Windows machine the JVM defaults max heap size as `64MB`.

Given this situation, how should I deal with this constraint?

I could increase the **max heap size** using **command line** option to java, but that would require figuring out available RAM and writing some launching program or script. Besides, increasing to some **finite** max does not **ultimately** get rid of the issue.

I could rewrite some of my code to persist objects to file system frequently (using database is the same thing) to free up the memory. It could work, but it's probably a lot work too.

If you could point me to details of above ideas or some alternatives like **automatic virtual memory, extending heap size dynamically**, that will be great.

java jvm out-of-memory heap-memory

Share

Improve this question

Follow

edited Oct 24, 2020 at 14:58

BalusC
1

asked Sep 1, 2008 at 1:10

Eugene Yokota
95.5k ● 45 ● 217 ● 320

The default max heap size of 64 MB is from before J2SE 5.0. For J2SE 8.0 information, see "Garbage Collector Ergonomics" at docs.oracle.com/javase/8/docs/technotes/guides/vm/… . – Andy Thomas Apr 8, 2015 at 15:36

---

6 If you landed here because every OOM question is duped to this one, make sure you also check out: stackoverflow.com/questions/299659/… It provides the solution for cleaning up memory references 'just in time' before the OOM. SoftReferences may be the tool that solves your actual problem. – Steve Steiner Apr 24, 2018 at 20:13

The Sun/Oracle JVM has always been quite rigid about specifying the amount memory to use (and having some interesting defaults if left on its own). That was one of the

nice things of the Microsoft JVM back then - it was *fast* and it could use whatever memory the machine had.
– Thorbjørn Ravn Andersen Feb 21, 2021 at 16:12

in my case I missed **gradle.properties** file to add in my project – Mohd Qasim Mar 14, 2022 at 7:30

32 Answers

Sorted by: Highest score (default)

1 2 Next

315

Ultimately you always have a finite max of heap to use no matter what platform you are running on. In Windows 32 bit this is around `2GB` (not specifically heap but total amount of memory per process). It just happens that Java chooses to make the default smaller (presumably so that the programmer can't create programs that have runaway memory allocation without running into this problem and having to examine exactly what they are doing).

So this given there are several approaches you could take to either determine what amount of memory you need or to reduce the amount of memory you are using. One common mistake with garbage collected languages such as Java or C# is to keep around references to objects that you **no longer** are using, or allocating many objects when you could **reuse** them instead. As long as objects have a reference to them they will continue to use heap space as the garbage collector will not delete them.

In this case you can use a Java memory profiler to determine what methods in your program are allocating large number of objects and then determine if there is a way to make sure they are no longer referenced, or to not allocate them in the first place. One option which I have used in the past is "JMP" http://www.khelekore.org/jmp/.

If you determine that you are allocating these objects for a reason and you need to keep around references (depending on what you are doing this might be the case), you will just need to increase the max heap size when you start the program. However, once you do the memory profiling and understand how your objects are getting allocated you should have a better idea about how much memory you need.

In general if you can't guarantee that your program will run in some finite amount of memory (perhaps depending on input size) you will always run into this problem. Only after exhausting all of this will you need to look into caching objects out to disk etc. At this point you should have a very good reason to say "I need Xgb of memory" for something and you can't work around it by improving your algorithms or memory allocation patterns. Generally this will only usually be the case for algorithms operating on large datasets (like a database or some scientific analysis program) and then techniques like caching and memory mapped IO become useful.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:44

Hearen
7,778 ● 4 ● 57 ● 73

answered Sep 1, 2008 at 1:29

Ben Childs
4,570 ● 1 ● 22 ● 9

10 OpenJDK and OracleJDK have bundled profiler - jvisualvm. If you want more conveniences I'd suggest commercial Yourkit. – Petr Gladkikh Apr 17, 2013 at 4:40

2 Use close unused object and array, say resultsSet.close(); fileOutputStream.close(); fileOutputStream.flush(); , I use resultsSet.close() and was working magically. – MrSalesi Sep 3, 2021 at 22:35

@MrSalesi or just implement closable interface – z atef Mar 25 at 4:32

162

Run Java with the command-line option `-Xmx`, which sets the *maximum* size of the heap.

See here for details.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:51
Hearen
7,778 ● 4 ● 57 ● 73

answered Sep 1, 2008 at 5:00
David Webb
193k ● 57 ● 316 ● 302

9 How to set this parameter for ever? Cause I'm using 'gradlew assemble' command. – Dr.jacky Jan 19, 2017 at 7:32

5 Run->Run Configurations->Click on arguments->inside VM arguments type -Xms1g -Xmx2g – Arayan Singh May 24, 2018 at 18:38

108

You could specify **per** project how much heap space your project wants

Following is for **Eclipse Helios/Juno/Kepler**:

Right mouse click on

```
Run As - Run Configuration - Arguments - Vm Arguments
```

then add this

```
-Xmx2048m
```

Share Improve this answer Follow

edited Mar 23, 2019 at 8:49
Hearen
7,778 ● 4 ● 57 ● 73

answered Feb 25, 2011 at 18:45
allenhwkim
27.7k ● 18 ● 92 ● 127

1 hi bighostkim and cuongHuyTo, where is "Arguments"..i can able to see upto Run Configuration. Please tel me. My need to download and store nearly 2000 contacts from gmail. It crash due to out of memory exception – AndroidRaji Nov 24, 2012 at 5:49

@AndroiRaji: you right click your mouse onto the Java class that has a runnable main (that is "public static void main(String[] args)"), then choose Run As - Run Configuration. Then "Arguments" is the tab right after the Main (you see the tabs Main, Arguments, JRE, Classpath, Source, Environment, Common). – CuongHuyTo Mar 4, 2014 at 10:06

78

Increasing the heap size is not a "fix" it is a "plaster", 100% temporary. It will crash again in somewhere else. To avoid these issues, write high performance code.

1. Use local variables wherever possible.
2. Make sure you select the correct object (EX: Selection between String, StringBuffer and

StringBuilder)

3. Use a good code system for your program(EX: Using static variables VS non static variables)
4. Other stuff which could work on your code.
5. Try to move with multy THREADING

Share Improve this answer Follow

edited Mar 24, 2019 at 6:41

answered Feb 1, 2014 at 5:10

PeakGen
22.9k ● 99 ● 284 ● 493

This is so true. I am trying to fix one issue where I'm getting OOM on AWT thread but if I use different new thread, I am not getting OOM issue. All I can find online is increase heap size for AWT thread. – Ashish Feb 11, 2019 at 5:43

2 @Ash: Yes, fix the core problem instead of looking for plasters. – PeakGen Feb 12, 2019 at 7:59

Garbage collection and the memory management approach in Java was supposed to solve all these malloc-dealloc complications of its predecessors :(Of course I completely agree with this answer it's just a shame the defaults don't make it easy to write code with lean data-structures that are cleaned up ASAP. – Davos Oct 11, 2019 at 11:51

11 I do not agree. You need to set it to some value initially. If it turns out to be insufficient it does not necessarily mean that your application is bad. Maybe you were too optimistic. It is a valid solution to set it to a higher value in this case. – Zsolt Sky Jun 10, 2021 at 12:30

Sometimes the data is to blame, rather than the code. There exist trully massive datasets that need to be crosschecked in memory. – DustWolf Oct 26, 2023 at 11:40

33

Big caveat ---- at my office, we were finding that (on some windows machines) we could not allocate more than 512m for Java heap. This turned out to be due to the Kaspersky anti-virus product installed on some of those machines. After uninstalling that AV product, we found we could allocate at least 1.6gb, i.e, `-Xmx1600m` (m is mandatory other wise it will lead to another error "Too small initial heap") works.

No idea if this happens with other AV products but presumably this is happening because the AV program is reserving a small block of memory in every address space, thereby preventing a single really large allocation.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:46

Hearen
7,778 ● 4 ● 57 ● 73

answered Apr 14, 2009 at 13:34

David
6,152 ● 6 ● 37 ● 41

26

VM arguments worked for me in eclipse. If you are using eclipse version 3.4, do the following go to `Run --> Run Configurations -->` then select the project under maven build --> then select the tab "JRE" --> then enter `-Xmx1024m`.

Alternatively you could do `Run --> Run Configurations --> select the "JRE" tab -->` then enter -`Xmx1024m`

This should increase the memory heap for all the builds/projects. The above memory size is 1 GB. You can optimize the way you want.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:50
Hearen
7,778 ● 4 ● 57 ● 73

answered Jun 9, 2011 at 20:48
loveall
261 ● 3 ● 2

25

I would like to add recommendations from oracle trouble shooting article.

Exception in thread thread_name:
java.lang.OutOfMemoryError: Java heap space

> The detail message Java heap space indicates object could not be allocated in the Java heap. This error does not necessarily imply a memory leak

Possible causes:

1. *Simple configuration issue*, where the specified heap size is insufficient for the application.
2. *Application is unintentionally holding references to objects*, and this prevents the objects from being garbage collected.
3. *Excessive use of finalizers*.

> One other potential source of this error arises with applications that make excessive use of finalizers. If a class has a finalize method, then objects of that type do not have their space reclaimed at garbage collection time

After *garbage collection*, the objects are queued for *finalization*, which occurs at a later time. *finalizers* are executed by a daemon thread that services the finalization queue. If the *finalizer* thread cannot keep up with the finalization queue, then the Java heap could fill up and this type of **OutOfMemoryError** exception would be thrown.

One scenario that can cause this situation is when an application creates *high-priority threads* that cause the *finalization* queue to increase at a rate that is faster than the rate at which the finalizer thread is servicing that queue.

Share Improve this answer
Follow

edited Mar 23, 2019 at 8:47
Hearen
7,778 ●4 ●57 ●73

answered Mar 22, 2016 at 18:35
Ravindra babu
38.9k ●11 ●256 ●219

19

Yes, with `-Xmx` you can configure more memory for your JVM. To be sure that you don't leak or waste memory. Take a heap dump and use the Eclipse Memory Analyzer to analyze your memory consumption.

Share Improve this answer
Follow

edited Mar 23, 2019 at 8:53
Hearen
7,778 ●4 ●57 ●73

answered Oct 9, 2008 at 8:22
kohlerm
2,624 ●1 ●18 ●23

4 JVMJ9VM007E Command-line option unrecognised: -Xmx Could not create the Java virtual machine. Downvote
– Philip Rego Jan 15, 2019 at 22:28

12

add the below code inside **android/gradle.properties**:

```
org.gradle.jvmargs=-Xmx4096m -XX:MaxPermSize=4096m -
XX:+HeapDumpOnOutOfMemoryError
org.gradle.daemon=true
org.gradle.parallel=true
org.gradle.configureondemand=true
```

Share Improve this answer

Follow

answered Aug 26, 2021 at 8:17

CodingEra
1,777 ● 13 ● 21

11

Follow below steps:

1. Open `catalina.sh` from tomcat/bin.
2. Change JAVA_OPTS to

   ```
   JAVA_OPTS="-Djava.awt.headless=true -Dfile.encodin
   -Xmx1536m -XX:NewSize=256m -XX:MaxNewSize=256m -XX
   -XX:MaxPermSize=256m -XX:+DisableExplicitGC"
   ```

3. Restart your tomcat

Share Improve this answer

Follow

edited Mar 23, 2019 at 8:46

Hearen
7,778 ● 4 ● 57 ● 73

answered Oct 7, 2013 at 10:56

Pradip Bhatt
658 ● 9 ● 15

11

If you came here to search this issue from REACT NATIVE.

Then i guess you should do this

```
cd android/ && ./gradlew clean && cd ..
```

Share Improve this answer Follow

answered Sep 8, 2021 at 6:44

Muhammad Raqib
334 ● 3 ● 13

9

By default for development JVM uses small size and small config for other performance related features. But for production you can tune e.g. (In addition it Application Server specific config can exist) -> (If there still isn't enough memory to satisfy the request and the heap has already reached the maximum size, an OutOfMemoryError will occur)

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

-XX:ParallelGCThreads=8
-XX:+CMSClassUnloadingEnabled
-XX:InitiatingHeapOccupancyPercent=70
-XX:+UnlockDiagnosticVMOptions
-XX:+UseConcMarkSweepGC
-Xms512m
-Xmx8192m
-XX:MaxPermSize=256m (in java 8 optional)
```

For example: On linux Platform for production mode preferable settings.

After downloading and configuring server with this way http://www.ehowstuff.com/how-to-install-and-setupapache-tomcat-8-on-centos-7-1-rhel-7/

1.create setenv.sh file on folder /opt/tomcat/bin/

```
   touch /opt/tomcat/bin/setenv.sh
```

2.Open and write this params for setting preferable mode.

```
nano  /opt/tomcat/bin/setenv.sh

export CATALINA_OPTS="$CATALINA_OPTS -XX:ParallelGCThr
export CATALINA_OPTS="$CATALINA_OPTS -XX:+CMSClassUnlo
export CATALINA_OPTS="$CATALINA_OPTS -XX:InitiatingHea
export CATALINA_OPTS="$CATALINA_OPTS -XX:+UnlockDiagno
export CATALINA_OPTS="$CATALINA_OPTS -XX:+UseConcMarkS
export CATALINA_OPTS="$CATALINA_OPTS -Xms512m"
export CATALINA_OPTS="$CATALINA_OPTS -Xmx8192m"
export CATALINA_OPTS="$CATALINA_OPTS -XX:MaxMetaspaceS
```

3. `service tomcat restart`

> Note that the JVM uses more memory than just the heap. For example Java methods, thread stacks and native handles are allocated in memory separate from the heap, as well as JVM internal data structures.

Share Improve this answer

Follow

edited Mar 23, 2019 at 8:45

Hearen
7,778 ● 4 ● 57 ● 73

answered May 26, 2016 at 12:00

Musa
2,652 ● 29 ● 25

9

Add this line to your gradle.properties file

```
org.gradle.jvmargs=-Xmx2048m -XX:MaxPermSize=512m -
XX:+HeapDumpOnOutOfMemoryError -Dfile.encoding=UTF-8
```

It should work. You can change MaxPermSize accordingly to fix your heap problem

Share Improve this answer Follow

answered Nov 24, 2021 at 13:09
Talha Akbar
819 ● 7 ● 24

8

Easy way to solve `OutOfMemoryError` in java is to increase the maximum heap size by using JVM options `-Xmx512M`, this will immediately solve your OutOfMemoryError. This is my preferred solution when I get OutOfMemoryError in Eclipse, Maven or ANT while building project because based upon size of project you can easily ran out of Memory.

Here is an example of increasing maximum heap size of JVM, Also its better to keep -Xmx to -Xms ration either 1:1 or 1:1.5 if you are setting heap size in your java application.

`export JVM_ARGS="-Xms1024m -Xmx1024m"`

Reference Link

Share Improve this answer Follow

edited Jun 20, 2012 at 10:34
mtk
13.6k ● 16 ● 74 ● 116

answered Sep 4, 2011 at 5:05
chaukssey
81 ● 1 ● 1

1 Any idea why we need to keep them in 1:1 or 1:1.5 ratio?
– ernesto Oct 25, 2012 at 10:56

7

I read somewhere else that you can try - catch java.lang.OutOfMemoryError and on the catch block, you can free all resources that you know might use a lot of memory, close connections and so forth, then do a `System.gc()` then re-try whatever you were going to do.

Another way is this although, i don't know whether this would work, but I am currently testing whether it will work on my application.

The Idea is to do Garbage collection by calling System.gc() which is known to increase free memory. You can keep checking this after a memory gobbling code executes.

```
//Mimimum acceptable free memory you think your app ne
long minRunningMemory = (1024*1024);
```

```
Runtime runtime = Runtime.getRuntime();

if(runtime.freeMemory()<minRunningMemory)
 System.gc();
```

Share Improve this answer

Follow

edited Mar 23, 2019 at 8:52

Hearen
7,778 ● 4 ● 57 ● 73

answered Aug 17, 2010 at 7:30

mwangi
1,626 ● 1 ● 20 ● 23

7 In general I think the JVM will prefer to Garbage Collect (GC) rather than throw an OutOfMemoryError. Explicitly calling System.gc() after OutOfMemoryError might possibly help on some VMs/configurations, but I would not expect it to work very well in the general case. However, dropping unnecessary object references would definitely help in almost all cases. – Mike Clark Nov 20, 2010 at 10:50

8 @mwangi Calling System.gc() directly from the code is generally a bad idea. It's just a **suggestion** to JVM that GC should be performed, but there's absolutely **no guarantee** that it will be performed. – user1014581 Oct 26, 2011 at 12:51

this is bad for 2 reasons. 1. Garbage collection is not guaranteed and 2. catching errors is a very bad practice. There is a distinction between exceptions and errors for that reason – Yorgos Lamprakis Aug 17, 2023 at 12:49

If you need to monitor your memory usage at runtime, the `java.lang.management` package offers `MBeans` that can

6

be used to monitor the memory pools in your VM (eg, eden space, tenured generation etc), and also garbage collection behaviour.

The free heap space reported by these MBeans will vary greatly depending on GC behaviour, particularly if your application generates a lot of objects which are later GC-ed. One possible approach is to monitor the free heap space after each full-GC, which you may be able to use to make a decision on freeing up memory by persisting objects.

Ultimately, your best bet is to limit your memory retention as far as possible whilst performance remains acceptable. As a previous comment noted, memory is always limited, but your app should have a strategy for dealing with memory exhaustion.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:53

Hearen
7,778 ● 4 ● 57 ● 73

answered Oct 9, 2008 at 12:04

Leigh
4,318 ● 1 ● 29 ● 17

6

In android studio add/change this line at the end of `gradle.properties (Global Properties)`:

```
...
org.gradle.jvmargs=-XX\:MaxHeapSize\=1024m -Xmx1024m
```

if it doesn't work you can retry with bigger than 1024 heap size.

Share Improve this answer Follow

answered Oct 24, 2020 at 14:35

Nima Ganji
601 ● 9 ● 18

5

I have faced same problem from java heap size.

I have two solutions if you are using java 5(1.5).

1. just install jdk1.6 and go to the preferences of eclipse and set the jre path of jav1 1.6 as you have installed.
2. Check your VM argument and let it be whatever it is. just add one line below of all the arguments present in VM arguments as -Xms512m -Xmx512m -XX:MaxPermSize=...m(192m).

I think it will work...

Share Improve this answer Follow

edited Mar 23, 2019 at 8:51

Hearen
7,778 ● 4 ● 57 ● 73

answered May 10, 2011 at 18:48

Soumya Sandeep Mohanty

59 ● 1 ● 1

5

If you are using Android Studio just add these lines with **gradle.properties** file

org.gradle.jvmargs=-Xmx2048m -XX:MaxPermSize=512m -XX:+HeapDumpOnOutOfMemoryError -Dfile.encoding=UTF-8

Share Improve this answer Follow

answered Jan 9, 2022 at 17:40

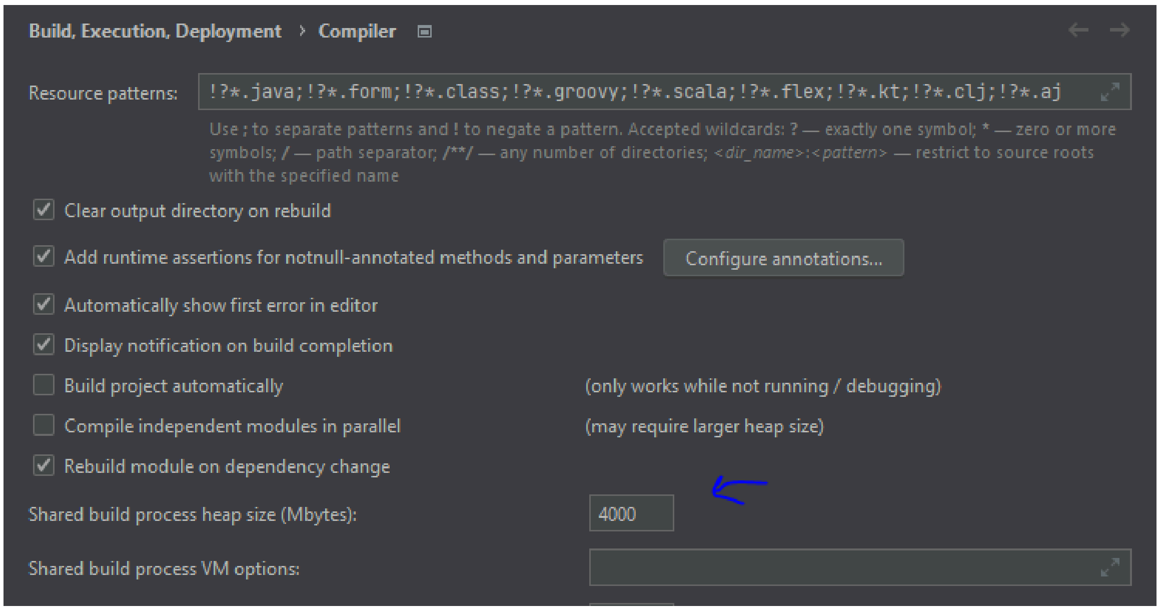
Neeraj Singh

668 ● 11 ● 15

5

In my case it solved by assigning more memory to `Shared build process heap size` in intellij settings.

Go to intellij settings > Compiler > Shared build process heap size


Build, Execution, Deployment › Compiler

Resource patterns: `!?*.java;!?*.form;!?*.class;!?*.groovy;!?*.scala;!?*.flex;!?*.kt;!?*.clj;!?*.aj`

Use ; to separate patterns and ! to negate a pattern. Accepted wildcards: ? — exactly one symbol; * — zero or more symbols; / — path separator; /**/ — any number of directories; <dir_name>:<pattern> — restrict to source roots with the specified name

- [x] Clear output directory on rebuild
- [x] Add runtime assertions for notnull-annotated methods and parameters
- [x] Automatically show first error in editor
- [x] Display notification on build completion
- [ ] Build project automatically (only works while not running / debugging)
- [ ] Compile independent modules in parallel (may require larger heap size)
- [x] Rebuild module on dependency change

Shared build process heap size (Mbytes): 4000

Shared build process VM options:

Share Improve this answer Follow

answered Feb 9, 2022 at 10:19

Morteza

662 ● 9 ● 17

4

Note that if you need this in a deployment situation, consider using Java WebStart (with an "ondisk" version, not the network one - possible in Java 6u10 and later) as it allows you to specify the various arguments to the JVM in a cross platform way.

Otherwise you will need an operating system specific launcher which sets the arguments you need.

Share Improve this answer Follow

edited Mar 23, 2019 at 8:52

Hearen

7,778 ● 4 ● 57 ● 73

answered May 19, 2009 at 4:45

Thorbjørn Ravn Andersen

75.3k ● 34 ● 199 ● 352

1 Java WebStart is being phased out. I am not yet aware of a suitable replacement. – Thorbjørn Ravn Andersen Dec 31, 2018 at 12:33

4

Android Studio

File -> Invalidate Caches and Restart solved it for me :)

Share Improve this answer Follow

answered Mar 14, 2021 at 13:35

itzo
1,249 ● 14 ● 18

4

To address the **java.lang.OutOfMemoryError: Java heap space** issue, you need to modify **gradle.properties** file.

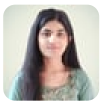
Add this line: `org.gradle.jvmargs=-Xmx2048m`

This line instructs Gradle to allocate a maximum of 2048 megabytes of memory to the JVM heap, which can help prevent out-of-memory errors during Gradle builds.

You can also use: `org.gradle.jvmargs=-Xms1024m` or `org.gradle.jvmargs=-Xmx4096m`.

Share Improve this answer Follow

answered Mar 14 at 4:47



Samiya Khan
359 ● 3 ● 5

3

If this error comes up during APK generation in react-native, cd into the android folder in your project and do:

```
./gradlew clean
```

then

```
./gradlew assembleRelease
```

If error persists, then, restart your machine.

edited Sep 10, 2021 at 0:12

Share Improve this answer

Follow

answered Aug 29, 2021 at 13:37

Chukwunazaekpere
1,012 ● 1 ● 7 ● 16

2

Regarding to netbeans, you could set max heap size to solve the problem.

Go to 'Run', then --> 'Set Project Configuration' --> 'Customise' --> 'run' of its popped up window --> 'VM Option' --> fill in '-Xms2048m -Xmx2048m'.

Share Improve this answer

Follow

edited Mar 23, 2019 at 8:48

Hearen
7,778 ● 4 ● 57 ● 73

answered Jun 12, 2017 at 22:56

htlbydgod
348 ● 2 ● 8

1

If everything else fails, in addition to increasing the max heap size try also increasing the swap size. For Linux, as of now, relevant instructions can be found in https://linuxize.com/post/create-a-linux-swap-file/.

This can help if you're e.g. compiling something big in an embedded platform.

Share Improve this answer Follow

answered Nov 11, 2019 at 22:39

nccc
1,155 ● 3 ● 11 ● 20

1

Java OOM Heap space issue can also arise when your DB connection pool got full.

I faced this issue because of my Hikari Connection pool (when upgraded to Spring boot 2.4.*) was full and not able to provide connections anymore (all active connections are still pending to fetch results from database).

Issue is some of our native queries in JPA Repositories contain **ORDER BY ?#{#pageable}** which takes a very long time to get results when upgraded.

Removed **ORDER BY ?#{#pageable}** from all the native queries in JPA repositories and OOM heap space issue along with connection pool issue got resolved.

Share Improve this answer Follow

edited May 12, 2021 at 20:44

answered Apr 22, 2021 at 23:46

Haranath Boggarapu
303 ● 4 ● 11

If this error occurs right after execution of your junit tests, then you should execute **Build** -> **Rebuild Project**.

1

Share Improve this answer Follow

edited May 28, 2021 at 7:35

answered May 28, 2021 at 7:28

samuelru
119 ● 1 ● 5

0

If this issue is happening in Wildfly 8 and JDK1.8,then we need to specify MaxMetaSpace settings instead of PermGen settings.

For example we need to add below configuration in setenv.sh file of wildfly. `JAVA_OPTS="$JAVA_OPTS -XX:MaxMetaspaceSize=256M"`

For more information, please check [Wildfly Heap Issue]

Share Improve this answer Follow

edited Aug 28, 2017 at 5:10

aristotll
9,147 ● 6 ● 34 ● 54

answered Apr 4, 2017 at 15:52

satish
1,641 ● 1 ● 12 ● 4

0

If you keep on allocating & keeping references to object, you will fill up any amount of memory you have.

One option is to do a transparent file close & open when they switch tabs (you only keep a pointer to the file, and when the user switches tab, you close & clean all the objects... it'll make the file change slower... but...), and maybe keep only 3 or 4 files on memory.

Other thing you should do is, when the user opens a file, load it, and intercept any OutOfMemoryError, then (as it is not possible to open the file) close that file, clean its objects and warn the user that he should close unused files.

Your idea of dynamically extending virtual memory doesn't solve the issue, for the machine is limited on resources, so you should be carefull & handle memory issues (or at least, be carefull with them).

A couple of hints i've seen with memory leaks is:

--> Keep on mind that if you put something into a collection and afterwards forget about it, you still have a strong reference to it, so nullify the collection, clean it or do something with it... if not you will find a memory leak difficult to find.

--> Maybe, using collections with weak references (weakhashmap...) can help with memory issues, but you **must** be carefull with it, for you might find that the object you look for has been collected.

--> Another idea i've found is to develope a persistent collection that stored on database objects least used and transparently loaded. This would probably be the best approach...

Share Improve this answer

Follow

edited Mar 23, 2019 at 8:52

Hearen

7,778 ● 4 ● 57 ● 73

answered Sep 29, 2010 at 7:47

SoulWanderer

305 ● 3 ● 11

1 2 Next

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.