I came across this article. Thought of sharing the findings with you.

[Avoid Container getting killed](https://blog.csanchez.org/2017/05/31/running-a-jvm-in-a-container-without-getting-killed/)

I created a standard Spring boot application and deployed on Docker container. For the base image, I used **openjdk:8u131-jre-alpine.**

I created four docker container instances of this Spring boot application with different JVM and container settings to observe how docker stats showed up for each:

1. Container - Tomcat (Container id - e61c6c1c1ba3)

JVM Settings: -Xms256m -Xmx512m -XX:MaxMetaspaceSize=128m -XX:+UseG1GC

1. Container – Tomcat (Container id - 1bbf957d5031)

JVM Settings: - Xms256m -Xmx512m -XX:MaxMetaspaceSize=128m

1. Container – Tomcat(Container id - dd7192d220d7)

JVM Settings: - XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap -XX:MaxRAMFraction=2 -XX:MaxMetaspaceSize=128m

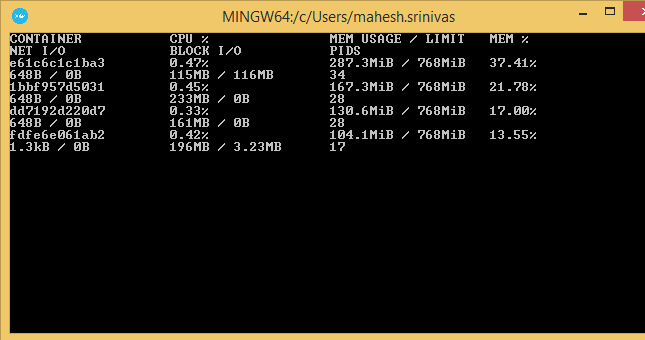
1. Container – Undertow(Container id - fdfe6e061ab2)

JVM Settings: -XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap -XX:MaxRAMFraction=2 -XX:MaxMetaspaceSize=128m

Below were observations after keeping them idle after 10 min without performing any transaction .



After 20 mins, below were the observations :



When using Undertow container with following JVM settings supported in JDK 8u131, I was easily able to minimize docker memory consumption by more than half.

-XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap -XX:MaxRAMFraction=2 -XX:MaxMetaspaceSize=128m (docker memory – 13.55 to 13.56%)

Closely behind it was the tomcat container with above JVM settings and docker memory consumed was in the range of 16.41 to 17%.

Highest docker memory consumption was with tomcat container with following JVM settings:

-Xms256m -Xmx512m -XX:MaxMetaspaceSize=128m -XX:+UseG1GC (memory – 37 to 38%)