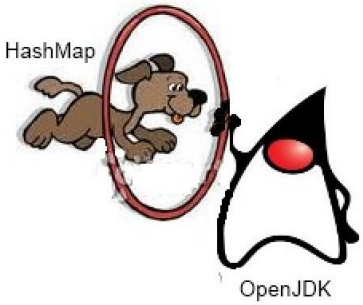

OpenJDK and HashMap …. Safely teaching an old dog new (Off-Heap!) Tricks

by

Peter Lawrey,
Ben Cotton

1. Brief History of HashMap
2. java.util.HashMap (not Thread Safe)
3. thrSafeHM = Collections.*synchronizedMap*(hm); (thread Safe, but lock stupid)
4. java.util.concurrent.ConcurrentHashMap (thread safe,lock smart, but not "perfect")
5. All thus far are 100% On-Heap object operands
6. Mention Doug Lea's sun.misc.Unsafe survey and Peter/Christoph's Off-heap JEP
7. On-Heap Advantages
    a. Familiar
    b. Not Unsafe
    c. Automatic GC services – no need to self-manage malloc()/free() operations
    d. Full in-place integration with Java Lock API and JMM
8. Off-Heap advantages
    a. Never again worry about a "Stop The World" GC event
    b. Can be used as a native IPC transport (no IP-loopback of java.net.Socket)
9. HashMap's future (for both on-Heap and off-Heap versions)
    a. Adaptable to being transformed into full-blown JCACHE operand (JSR-107 compliant)
    b. Adaptable to being transformed into an ACID-transaction capable Cache operand
    c. Adaptable to being transformed into an HTM-Transaction benefiting operand (re-supply here Peter's blog on HTM, and the consequent discussion w/ Gil, et.al. at https://groups.google.com/forum/#!topic/mechanical-sympathy/f84bwRQpyTQ)
    d. Adaptable to being the "tree" in a high-performance (CAP-Brewer coherent) distributed Java data grid "forrest".
    e. A replacement for same host IPC (way better than IP loopback!)