
JavaOne™

ORACLE

What to Expect from HotRockit

Marcus Hirt

What this talk is all about…

- Outline what some of the results from merging Hotspot and JRockit will look like (and already looks like!)
- Give a rough idea of in what order to expect the technology to appear
- ***Only*** about what the convergence will bring
  - I.e. will NOT be talking about module system, lambdas etc
  - This is a bit of a roadmap-ish talk

Disclaimer: JDK release targets are our current best estimates and can (and likely will) change!

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Assumptions

- Assume you know what a JVM is
  - Neat piece of software that translates your byte codes
  - Helps to know it is an adaptively optimizing platform access to runtime information
- There are several different ones
  - Most of you are probably using HotSpot
  - The rest of you are probably using JRockit or IBM J9
  - Big test suite to ensure compatibility (JCK)

Oracle's JVM

(2002 – BEA Systems acquires Appeal, gets JRockit)

2008 – Oracle acquires BEA, gets JRockit

2010 – Oracle acquires Sun, gets HotSpot

Oracle's JVM

(2002 – BEA Systems acquires Appeal, gets JRockit)

2008 – Oracle acquires BEA, gets JRockit

2010 – Oracle acquires Sun, gets HotSpot

So, now what?

Oracle's JVM

(2002 – BEA Systems acquires Appeal, gets JRockit)

2008 – Oracle acquires BEA, gets JRockit

2010 – Oracle acquires Sun, gets HotSpot

So, now what?

- Build best of breed JVM
  - Base it on HotSpot
  - Add the most appreciated JRockit features

Roadmap

JRCMD -> JCMD

JDK 7 Time Frame (Update)

- Command line utility to enumerate and send commands to running JVMs
- Will be renamed to JCMD
- First port will have a limited set of commands

J[R]CMD Examples

JDK 7 Updates & Forward

`jrcmd (jcmd)`

prints the running JVMs – PID plus main class

`jrcmd <PID> help (help)`

prints the available commands

`jrcmd <PID> print_threads (Thread.print)`

prints thread dumps

`jrcmd <PID> print_object_summary (GC.class_histogram)`

prints a histogram of the heap, by class

`[start|stop|dump|clone]_flightrecording (JFR.[start|stop|dump])`

controls the Java Flight Recorder (more on this later)

J[R]CMD Examples

JDK 8 & Later

`jrcmd <PID> heap_diagnostics (No yet)`

prints heap information, including semi-ref details

`jrcmd <PID> print_memusage (Not yet)`

prints native memory allocation, down to individual allocation sites

JCMD Demo

Using mostly JRCMD, but with JCMD sneak peak!

JMX Agent Update

The JDK 7 Time Frame (Update)

- Use same port for RMI Registry and RMI Server
  - Easier firewall configuration
- Life Cycle Control
  - Even after the JVM has been started (using JCMD)
- Improved ergonomics

```
-Xmanagement:ssl=false,authenticate=false
```

Instead of:

```
-Dcom.sun.management.jmxremote.port=7091
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=false
```


JDP (Java Discovery Protocol)

The JDK 7 Time Frame (Update)

- Multicasting heartbeat for JVM services
- Used to discover manageable JVMs on the network
- Also to discover JVM's no longer running
- Normally used with the JMX management agent
  - Follows the management agent life cycle

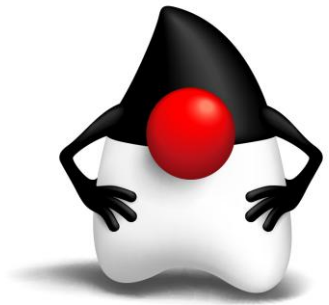
```
-Xmanagement:port=7091,autodiscovery=true
```


MBean Updates

JDK 7 Time Frame

- Many JRockit MBeans will be ported
  - Better support for the Mission Control console
- Some functionality already available:
  - `OperatingSystem#getProcessCpuLoad()`
  - `OperatingSystem#getSystemCpuLoad()`
  - `Threading#getThreadAllocatedBytes(long [] threadIDs)`
- Examples of coming functionality:

```
DiagnosticCommandMBean (MBean API for JCMD access)
ProfilingMBean         (method invocation counts and timing)
PerfCounterMBean       (MBean access to internal perf counters)
```



Console Demo

JRockit Flight Recorder -> Java Flight Recorder

JDK 7 Time Frame (update)

- In-Flight Recorder for Java (profiling and diagnostics)
  - Always on
  - Very low overhead
  - Dump data anytime
  - Go back in time to see what lead up to a problem
- Rich GUI in JMC
  - Integration with the Oracle stack (WLS, DMS etc)
  - Built-in GUI editor
  - Build and export custom plug-ins directly from Mission Control

JFR Demo

Memleak -> On-line Heap Analyzer

JDK 8/9 Time Frame

- Low overhead heap analyzer
  - Piggybacks on the GC
- On-line analysis
  - No need for large memory consuming heap dumps

Memleak Demo

No More Perm-Gen

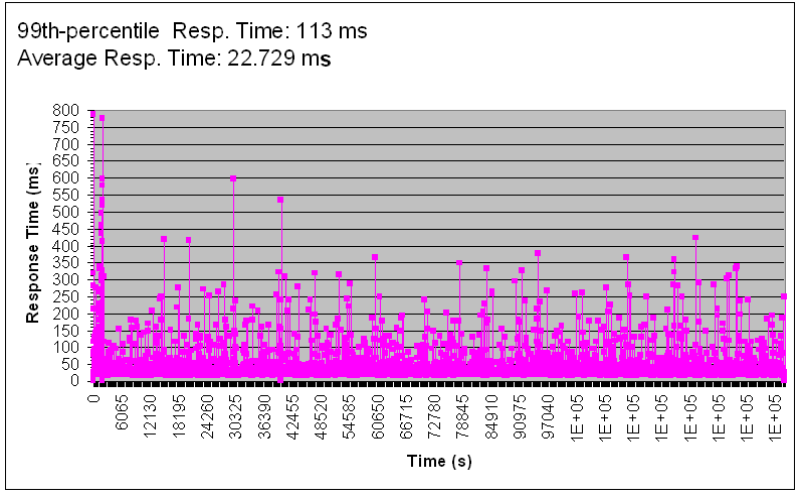
JDK 8 Time Frame

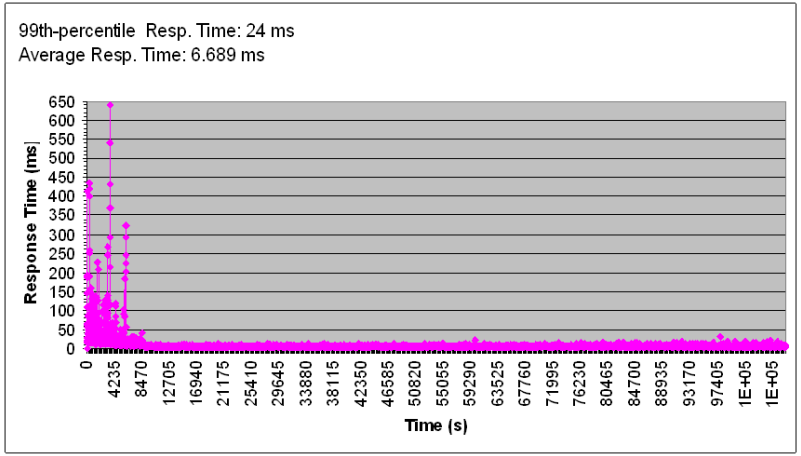
- Perm-gen will be removed
- Will use native memory and heap and allocate as needed
- No need to decide the required size up front
- No need for tuning

Other Improvements

After JDK 8 Time Frame

- Deterministic GC
  - Soft real-time GC
  - Pause time target
- Compiler optimizations





JavaOne ORACLE

JRockit Free!

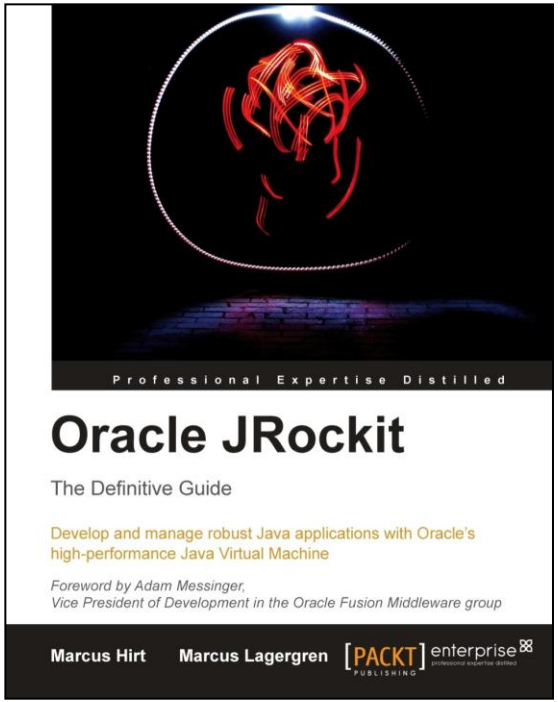
Now!

- JRockit is now under the same license as HotSpot
  - This means you can start trying out the technology right away!
- Mission Control is free for development
- Take it for a spin and send feedback!

→ marcus.hirt@oracle.com ←

Shameless Book Plug

Oracle JRockit: The Definitive Guide



Q&A

Tokyo 2012

De 4 à 6 de Abril de 2012

San Francisco 2012

De 30 de Setembro à 4 de Outubro de 2012


JavaOne™