

Oracle WebLogic Server

The #1 Application Server for
Enterprise Java and SOA Developers



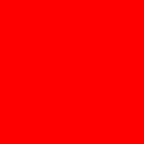
ORACLE®

JRockit Mission Control and JRockit Flight Recorder

Jeffrey West

Application Grid Product Management





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.

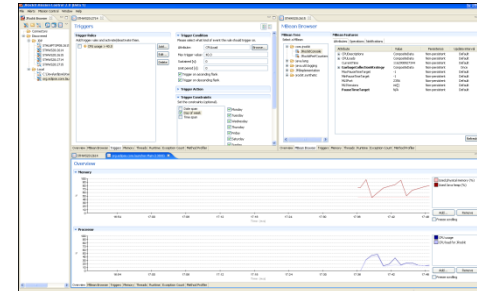
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Agenda

- • What is JRockit Mission Control?
- What is JRockit Flight Recorder?
- JRMC & JFR Demos
- Conclusion & WebLogic Resources

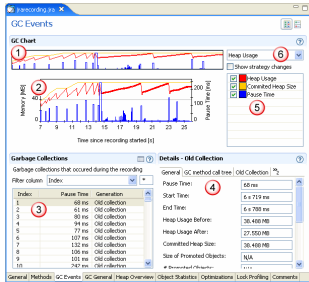
Oracle JRockit Mission Control

Monitoring



Profiling & performance tuning

Deploy



Development

Operations

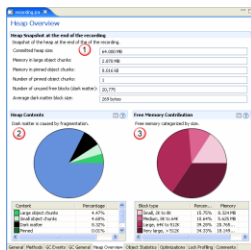
Alerts & triggers

Regression testing

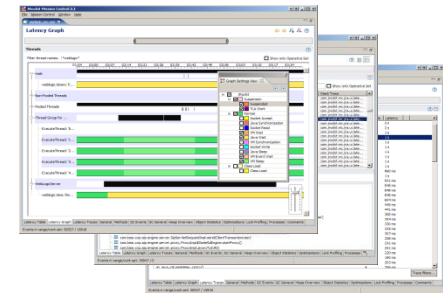
Analysis

Production diagnostics

Escalate

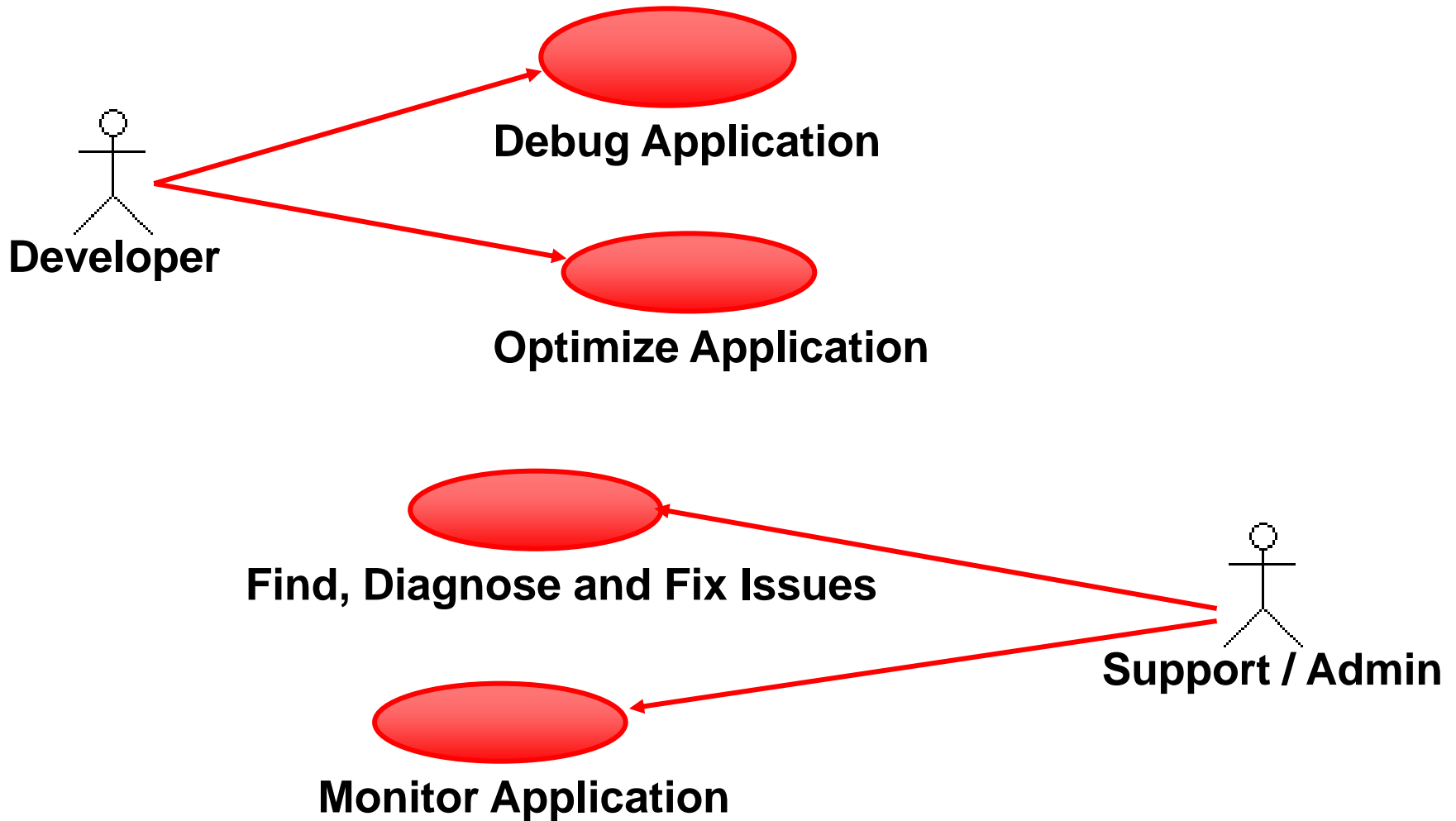


Troubleshooting



Use Cases

JRokit Mission Control



Runtime Monitoring & Profiling

JRockit Mission Control

What can you Monitor?

- CPU Usage
- Memory & Heap Usage
- Garbage Collection Activity
- Thread Usage and stack traces
- Mbeans with Mbean Browser

What can you Profile?

- User-selected Java Methods
- User-selected Exceptions

Oracle JRockit Mission Control

Monitoring Dashboard



Oracle JRockit Mission Control

Monitoring Threads

Oracle JRockit Mission Control

File Window Help

*10.0.0.51:7191 X

Threads

Live Thread Graph

Live Threads 9:53:35 AM

Filter Column Thread Name [ACTIVE] CPU Profiling Deadlock Detection Allocation

Thread Name	Thread State	Blocked Count	Total CPU Usage	Deadlocked	Allocated Bytes
[ACTIVE] ExecuteThread: '9' for queue: '...	BLOCKED	39,409	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '4' for queue: '...	BLOCKED	42,549	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '3' for queue: '...	BLOCKED	32,296	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '25' for queue: '...	WAITING	907	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '2' for queue: '...	BLOCKED	37,380	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '10' for queue: '...	BLOCKED	20,814	Not Enabled	Not Enabled	Not Enabled
[ACTIVE] ExecuteThread: '0' for queue: '...	BLOCKED	37,581	Not Enabled	Not Enabled	Not Enabled

Stack traces for selected threads

Stack traces for selected threads 9:53:35 AM

```
[ACTIVE] ExecuteThread: '9' for queue: 'weblogic.kernel.Default (self-tuning)' [62] (RUNNABLE)
  jrockit.net.SocketNativeIO.readBytesPinned line: not available [native method]
  jrockit.net.SocketNativeIO.socketRead line: 32
  java.net.SocketInputStream.socketRead0 line: not available
  java.net.SocketInputStream.read line: 129
  oracle.net.ns.Packet.receive line: 293
  oracle.net.ns.DataPacket.receive line: 104
  oracle.net.ns.NetInputStream.getNextPacket line: 315
  oracle.net.ns.NetInputStream.read line: 260
  oracle.net.ns.NetInputStream.read line: 185
  oracle.net.ns.NetInputStream.read line: 102
  oracle.jdbc.driver.T4CSocketInputStreamWrapper.readNextPacket line: 124
  oracle.jdbc.driver.T4CSocketInputStreamWrapper.read line: 80
  oracle.jdbc.driver.T4CMAREngine.unmarshalUB1 line: 1136
  oracle.jdbc.driver.T4CMAREngine.unmarshalSB1 line: 1113
```

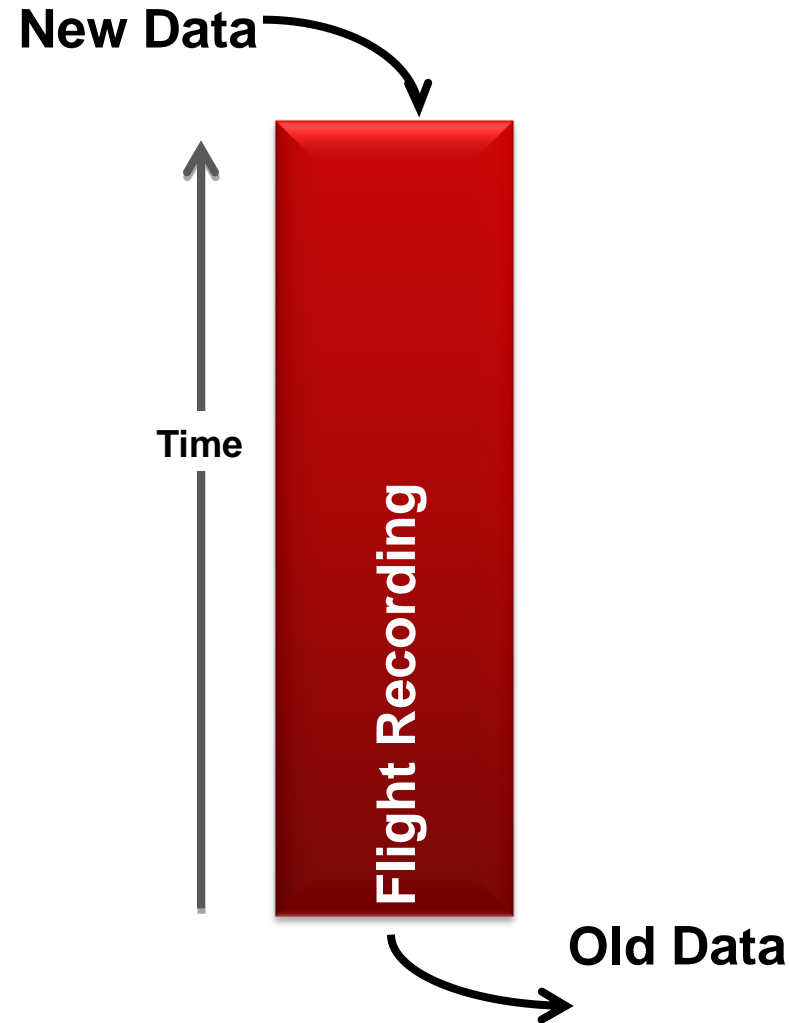
System Memory Threads

Agenda

- What is JRockit Mission Control?
- • What is JRockit Flight Recorder?
- JRMC & JFR Demos
- Conclusion & WebLogic Resources

What is the JRockit Flight Recorder?

- New in JRockit R28
- “Circular buffer” in JRockit JVM that stores diagnostic data
 - Always on
 - New data comes in and is stored, old data dropped off
- Built-in integration with JRMC
 - Replaces JRMC Runtime Analyzer and Latency Analyzer
- Very low/near zero overhead
 - Uses data already used by JVM
- Data can include events from the JVM and from any other event producer
 - WebLogic Server (WLDF)
 - Fusion Middleware (DMS)



Use Cases

JRokit Flight Recorder

- What it is designed for?
 - Provide diagnostic information in running production systems
 - Look back in time to see what happened after a crash
 - Capture most recent activity to enable analysis leading up to an issue
 - Capture data from all levels JVM, WLS, DMS, etc...
 - Offline/offsite analysis can be done using the JRMC GUI
 - JRokit dumps capture information to assist in crash-analysis
- What it is not designed for?
 - Large event payloads or very high volumes of events
 - Long history
 - Not a replacement for Debug logging or the server logging

Evolution - JRockit Flight Recorder

JRockit Mission Control 4.0 Integration

- Always On
- The Holy Grail of JVM Forensics
 - You usually want information from before the time you were alerted to the problem
- The main diagnostics and profiling tool
- Very low overhead
 - Piggybacking on JRockit internal systems
 - Very low overhead engine, events recorded thread locally
- Producer API for third party event providers
 - Working with other teams within Oracle
 - Will show demo for WLS database calls

Oracle JRockit Mission Control

The screenshot displays the Oracle JRockit Mission Control 3.0.3 interface. On the left, the 'JVM Browser' pane shows a tree view of JVM components, including 'JRockit', 'Class Loading', 'Code Generation', 'JVM', 'Java', and 'Garbage Collector'. The 'Properties' pane below it shows a list of active threads and their associated Java methods, such as 'java.util.Hashtable.get(Object)' and 'javax.naming.NameImpl.getBoolean(Properties, String)'. The main area features a 'Latency Graph' at the top, which is a red bar chart showing latency over time. Below the graph is the 'Threads' section, which displays a detailed execution timeline for the 'main' thread and several 'Pooled Threads'. The timeline is color-coded to represent different JVM activities, and includes a 'Garbage Collector' section. The bottom of the interface shows a navigation bar with tabs for 'General', 'Methods', 'GC General', 'GCs', 'Heap', 'Objects', 'Latency Log', 'Latency Graph', 'Latency Traces', 'Optimizations', 'Locks', 'Notes', and 'Processes'. The status bar at the bottom indicates 'Operative Set 0 / 5954'.

Replace with 4.0 Screenshot

Oracle JRockit Mission Control

The screenshot displays the Oracle JRockit Mission Control 3.0.3 interface. The main window shows a 'Trend Analysis - Which types are leaking?' table. The table lists various Java types with columns for Type, Growth (bytes/sec), % of Heap, Size (KB), and # Instances. The top rows are highlighted in red, indicating the most significant leaks. A large red watermark 'Replace with 4.0 Screenshot' is overlaid on the image.

Type	Growth (bytes/sec)	% of Heap	Size (KB)	# Instances
DemoLeak\$DemoObject	790	2,85%	39	1 276
java.util.Hashtable\$Entry	767	2,8%	39	1 673
java.util.Hashtable\$Entry[]	296	1,29%	18	74
java.lang.reflect.Method	204	7,44%	104	1 335
java.util.HashMap\$Entry	32	1,42%	19	850
java.lang.String	1	0,8%	146	6 265
java.lang.String[]	1	1,1%	16	527
java.lang.reflect.Field	4	3,33%	46	664
java.util.TreeMap\$Entry	2	0,72%	10	322
com.sun.jmx.mbeanserver.ConvertingMethod	1	0,17%	2	103
java.lang.reflect.Constructor[]	1	0,16%	2	103
java.lang.Class	0	13,16%	184	1 686
java.util.LinkedHashMap\$Entry	0	1,57%	22	706
java.nio.DirectByteBuffer	0	0,73%	10	219
java.lang.ref.WeakReference	0	0,67%	9	242
javax.management.MBeanAttributeInfo	0	0,62%	8	279
java.nio.DirectLongBufferU	0	0,58%	8	172
int[]	0	0,57%	7	66
java.util.WeakHashMap\$Entry	0	0,47%	6	141
java.util.concurrent.ConcurrentHashMap\$Segment	0	0,43%	6	192
sun.management.counter.perf.PerfLongCounter	0	0,38%	5	172



DEMONSTRATIONS

Find us Online!



www.oracle.com/technetwork/middleware/weblogic



www.YouTube.com/OracleWebLogic



Give us feedback! [@OracleWebLogic](https://twitter.com/OracleWebLogic)

www.twitter.com/OracleWebLogic



www.facebook.com/OracleWebLogic