

Welcome to the session...

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.

Basically, don't trust anything I say...

© 2013 Oracle Corporation



#### ORACLE

# Java Flight Recorder - Next generation diagnostics and profiling

Rickard Bäckman HotSpot JVM Compiler Team, Java Platform

```
public class Example {
  public void saveTransaction(Transaction tn) {
    synchronized (someLock) {
        // save to some shared data structure
     }
  }
}
```

ORACLE

© 2013 Oracle Corporation

public class Example {
 public void saveTransaction(Transaction tn) {
 System.err.println(threadId + "Waiting for lock");
 synchronized (someLock) {
 System.err.println(threadId + "Got lock");
 // save to some shared data structure
 }
 System.err.println(threadId + "Lock released");
}

ORACLE

© 2013 Oracle Corporation

... few million lines of output [T1] Waiting for lock [T3] Waiting for lock [T2] Waiting for lock [T2] Got lock [T1] Got lock [T1] Lock released [T3] Got lock [T3] Lock released [T2] Lock released ... another million lines of output

© 2013 Oracle Corporation

# **Released in 7u40**

ORACLE

© 2013 Oracle Corporation



ORACLE

© 2013 Oracle Corporation

#### **Event Recorder & Profiler**



ORACLE

© 2013 Oracle Corporation

#### Implemented in the JVM



ORACLE

© 2013 Oracle Corporation

#### No restart required



ORACLE

© 2013 Oracle Corporation

#### After-the-facts analysis

#### What went wrong?

FLIGHT

RECORDER

DO INOT

PFN

© 2013 Oracle Corporation

### **Event Recorder**

#### JVM tracks lots of information

© 2013 Oracle Corporation

13

### **Event Recorder**

#### JVMTI?

© 2013 Oracle Corporation

14



### Instrumented JVM & JDK

### Make that information visible

ORACLE

© 2013 Oracle Corporation

### Profiler

#### What is my CPU doing?!?

1 J WHAN 1 E & B	3 467 L SLXCYABAD	L 4 - 047 Z 7 Z Z 291 9849
		1 W
E- U A U DODD	F 4 48 18878818 811	1 <u>6 98</u> <u>992 8 8 9 6 982 999</u>
20 C P 8 0/60212 TIC /		9 ED 734 2 819 0 4401210
A G G G B B G G G G G G G G G G G G G G		CA SIL BUY V HAS A BLASDIP
	A T OT TOLDATIA NO.	TA STA JAN J AND 1 CATING
18 5 <u>5 5 60 7672</u> 8 810		
TH 2 7 7 15 EVE N #41	E 4 4 9 EBAG#T#41 41	THE THE & A & B WAR A AATRAKS
0815 1 0 J. 987 3 001	F 4 . H . F HIS PERIO	148 849 10 X 613592X 3883962
	4 19 YE AVE I AVE II	THE REPART OF A SHELLE AND AND AND A
107 1 10 20 1 000		64 blo 86 w 0 / 788 8 6/8880/
	한 승규는 이 관심 (한국동) 이 운동성동물이	
		184 SGS IN # 4 31 48 LL9#279
	• • • • · · · · · · · · · · · · · · · ·	46 848 82 8 1 10 107 462828
1.4 Z 11.114 4.45 Z 61		107 KIN 24 W 0 65 140 TANKED
		The same the T is the same state of the same
	A NUMBER OF	HE NE
201 U 19286 878 U 10	E SPI 694 EDI65	125 FRE 48 1 1 1 149 64 5499
622 U 12107 USH E EGI	I & III I WEALS STAAAAA	188 W 8 48 - 9 E HK2 89 6234
P 20 2 2 8002 974 4 924		2 8 8 16 16 5 9 9 867 87 8 39
	T A AND A ANALYS . THIS AND	
<b>84 5 31957 93 0 04</b>		· 병원 관련되는 일반 것 같은 것 것 같은 것 한 방문 것 수 있는 것 것 같이 있는 것 같이 없는 것 같이 있는 것 같이 없는 것 같이 않는 것 같이 않는 것 같이 없는 것 같이 않는 것 같이 없는 것 같이 없는 것 같이 않는 것 같이 없는 것 같이 않는 것 같이 않는 것 않이 않는 것 같이 않는 것 않는 것 않는 것 같이 않는 것 같이 없는 것 같이 않는 것 같이 않는 것 같이 않는 것 않는 것 같이 않는 것 않는
10 8 869081 42 8 284	1	** ******* * 5 /5 // 8***
101 0 88891E F 7 857	A - 1888 2 E VISBER 4 VI	
PAN P 2 7 22 11 7 954	1 124 1 2 39248 144	a feetta 1 / misae raiet /
4 5 F 5 5 5 5 5 1 1 1 1 1 1 1 1 1 1 1 1 1		
	TOTAL DISCUSSION DE	
	IT THAT STITUTES I	
121 H TAX205 TOA 8 59	· 18 도면 TEA & L R EFA EF / 1	INE #####!#!#!# #49%8%\$70729
2)1 E GN#164 870 8 891		AA PTABABNEN TEERAAAAAAA
THA . INCLEY THE & END	108 13219 <b>8196</b> 3062 1	※● うえばえるをきる くるのママスうどをあう ぶ
264	1 12010 FARME FARMEL 1	Ed alos 1902 Hiddensels r
777 10 00 100 00 00 0	ALL ROOT ALL READER F. A.	AN ANTA NALE CONTINUESION V
	ice Thies Classes B 4	
ANT OLE BRO BEL 1 .		THE DESIG SELECTION OF T
AA6 5:X8 988 892 8 9		NOT EASTS STATETAL
349 38725-962 854 8	<b>EVY DIMOZ BIZGE B 19</b>	24:44443 61444 6248: 0
848 808071/02 241 0	24 12042 3 923 2 05 4	
ALS VESUEALED CON Y	1 40.0 4 STA S YA	10472941 ENDI TZA 245779 4
		1749897 4942 964 24 418 2
THE PROPERTY OF A		
1 * * × * * * * * * * * * * * * * * * *	CCR # CGG     #E1 & EY 1	14++*XUE 8++1 E EF 9+1 E
The Automotical to	181 4041 UT 820 I mm 4	#12607 9041 6 E5 046 #
RE TYPESAMOR BUT		BIEZY0 844 1 05 234 8
	100 1000 mt 201 8 me	
27 898978888 22	30 B 964 22 740 6 66	

© 2013 Oracle Corporation

16

### Inside the JVM

#### Core in the JVM High-level things in Java

ORACLE

© 2013 Oracle Corporation



#### Access to JVM internals & subsystems

ORACLE

© 2013 Oracle Corporation

### Inside the JVM

### Class Unloading & Safepoints

ORACLE

© 2013 Oracle Corporation

### **Non-intrusive**

#### Designed to run all the time. In your production environment

ORACLE

© 2013 Oracle Corporation

### **Non-intrusive**

#### Low overhead Captures information that is already there

ORACLE

© 2013 Oracle Corporation

### **Non-intrusive**

#### Enable & disable at runtime

© 2013 Oracle Corporation





#### Problem in the environment

© 2013 Oracle Corporation

23



#### What happened before?

© 2013 Oracle Corporation

24



#### Keeps history of information

© 2013 Oracle Corporation

25





#### Java Mission Control

#### Dump the data on an SLA breach

ORACLE

© 2013 Oracle Corporation

# **Things JFR captures**

Compilation statistics Socket read/write VM Operations **Context switches Object Count** Exceptions Class Unload Garbage Collection CPU Load Method Compilation **Object allocation Threads** Class Load **Metaspace Statistics** Locks CodeCache Profiling Thread.sleep File read/write Garbage Collection statistics

ORACLE

© 2013 Oracle Corporation

#### How to implement a profiler?

© 2013 Oracle Corporation

28

### JVMTI

© 2013 Oracle Corporation

29

#### Byte code instrumentation

© 2013 Oracle Corporation

30

#### Stop the thread, walk the stack, collect the methods.

© 2013 Oracle Corporation

#### Demo

ORACLE

© 2013 Oracle Corporation

# **Enable Java Flight Recorder**

#### -XX:+UnlockCommercialFeatures -XX:+FlightRecorder

ORACLE

© 2013 Oracle Corporation

# **Start recording**

-XX:StartFlightRecording=filename=<...>,duration=5m

ORACLE

© 2013 Oracle Corporation





### Java Mission Control or jcmd <pid> JFR.start <options>



© 2013 Oracle Corporation

# Some interesting options

#### stackdepth=<nr>

#### dumponexit=<true/false>

ORACLE

© 2013 Oracle Corporation


# Abstraction of captured information

© 2013 Oracle Corporation

37



# Captures something interesting Latency? Performance?



© 2013 Oracle Corporation

# **Instant Events**

# Happens "instantly"

# Thread start, Thread stop, Class unload

ORACLE

© 2013 Oracle Corporation

# **Duration Events**

# Happens over a period of time

# Compilation, Garbage Collection, Wait for a lock

© 2013 Oracle Corporation



#### Something that should be done repeatedly

# CPU Load, Profiling Events Context switches

ORACLE

© 2013 Oracle Corporation

#### Enable & disable

© 2013 Oracle Corporation

42

# Threshold

© 2013 Oracle Corporation

43

# Frequency / Period

© 2013 Oracle Corporation

44

#### Stack traces

© 2013 Oracle Corporation

45

# Control signal / noise



ORACLE

© 2013 Oracle Corporation

# Filter early!

© 2013 Oracle Corporation

47



# "Start a recording"

© 2013 Oracle Corporation

48



#### Generates a stream of events

© 2013 Oracle Corporation

49



# Applies a filter and captures the remaining

ORACLE

© 2013 Oracle Corporation



#### One stream of Events

© 2013 Oracle Corporation

51



# Multiple recordings

© 2013 Oracle Corporation

52



# The Union of Events

© 2013 Oracle Corporation

53

Monday, September 30, 13



# A preconfigured settings for a recording

ORACLE

© 2013 Oracle Corporation



# default & profile

© 2013 Oracle Corporation

55

Monday, September 30, 13





#### Create your own

© 2013 Oracle Corporation

56

# **Java Flight Recorder**

# Implementation

© 2013 Oracle Corporation

57

# **Java Flight Recorder**

#### Challenges Assembly!

# Implementation

File format

#### Events

C++

#### Low level Memory buffers

© 2013 Oracle Corporation

58



#### Overhead must be low

© 2013 Oracle Corporation

59



# Memory footprint

© 2013 Oracle Corporation

60



# The JVM environment is changing

© 2013 Oracle Corporation

61

Monday, September 30, 13

# Challenges

# Scalability

© 2013 Oracle Corporation

62

# **Events in detail**



ORACLE

© 2013 Oracle Corporation

# **Events in detail**

#### Self-describing Events

© 2013 Oracle Corporation

64

# **Event Metadata**

#### Name, Path, Description

© 2013 Oracle Corporation

65

# **Event Payload Metadata**

# Name, Type, Description & & Content Type

ORACLE

© 2013 Oracle Corporation



# Semantics of a Value

Content Type	Displayed as
Bytes	42 MB
Percentage	42%
Address	0xDEADBEEF
Millis	242 ms

© 2013 Oracle Corporation

67



# Can be multiple Values

Content Type	Displayed
Class	Class Loader
	Name
	Modifiers (public,final)

© 2013 Oracle Corporation

68

# **Event definition in HotSpot**

```
<event id="ThreadSleep"
path="java/thread_sleep"
label="Java Thread Sleep"
has_thread="true"
has_stacktrace="true"
is_instant="false">
    <value type="MILLIS" field="time" label="Sleep Time"/>
    </event>
```

What does it actually mean? path is a identifier for the UI, label is a description of the Event has\_thread means each event is tied to a Thread has\_stacktrace means get a stack trace for the Event is\_instant decides whether it is instant event or not Finally the payload with a type name and a description.

# **Event in C++**

```
EventThreadSleep event;
```

```
...
if (event.should_commit()) {
    event.set_time(millis);
    event.commit();
}
```

An example of an event in the HotSpot source code. This event repots the time a thread slept. Thread.sleep()

So the event is created and when the thread returns from sleep we check if the event was enabled and should be committed. If so we save the time and commit.

ORACLE

© 2013 Oracle Corporation

. . .

# Or in assembler

# 281: cmp BYTE [rbp-0x60],0x0 # should commit 285: je 300 # jump if zero 287: mov QWORD rbp-0x58],rbx # save time 291: lea rdi,[rbp-0x70] # get pointer 295: call 0x101b17dc4 # commit

This is basically the assembler code generated for the same code (for amd64). Reserve some memory on the stack, initialize it. Check if the event was enabled, otherwise jump away. Save the time, get the pointer and call commit.

The overhead of an event that is disabled is very small. Compare a value to zero and jump away.

ORACLE

© 2013 Oracle Corporation



#### **Thread-local buffers**

© 2013 Oracle Corporation

72

Monday, September 30, 13


#### Promoted to Global buffers

## Global buffers are circular



ORACLE

© 2013 Oracle Corporation

### **File Format**

# Binary proprietary

© 2013 Oracle Corporation

74

## **File Format**

#### Fast writing

© 2013 Oracle Corporation

75

Monday, September 30, 13

## **File Format**

#### Self Contained

Header Event Records Event Meta Data

© 2013 Oracle Corporation

76



## Enough High-Level Problems & Solutions



© 2013 Oracle Corporation

#### Classes are referenced by Events

© 2013 Oracle Corporation

78

## java.lang.String

© 2013 Oracle Corporation



## org.springframework.security.ui.preauth.PreAut henticatedGrantedAuthoritiesWebAuthenticatio nDetails

© 2013 Oracle Corporation

#### 99 characters!

© 2013 Oracle Corporation

81



#### Introduce a unique ID



© 2013 Oracle Corporation

## **New problem**

## Mapping

© 2013 Oracle Corporation

83



#### Must be synchronized

© 2013 Oracle Corporation

84

Monday, September 30, 13



#### Class Load?

© 2013 Oracle Corporation

85

Monday, September 30, 13



#### Class Load?

### Startup performance!

ORACLE

© 2013 Oracle Corporation



### End of a Recording?

© 2013 Oracle Corporation

87



#### End of a Recording?

#### Classes can unload!

ORACLE

© 2013 Oracle Corporation



### **Piggyback Class Unloading!**

© 2013 Oracle Corporation

89



### The Class List can grow Big

© 2013 Oracle Corporation

90



## Lots of classes + Long class names = Lots of wasted memory

ORACLE

© 2013 Oracle Corporation



#### Reserve a bit in the ID

© 2013 Oracle Corporation

92



### Tag referenced classes!

```
#define CLASS_USED 1
void use_class_id(Klass* const klass) {
   klass->_trace_id |= CLASS_USED;
}
```

ORACLE

© 2013 Oracle Corporation



### Classes come & go. Don't waste memory

ORACLE

© 2013 Oracle Corporation

# Checkpoints

#### **Special Event**

© 2013 Oracle Corporation

95

# Checkpoints

### Writes the Class List, Resets the tags & clears the Class List

class\_pool.lookup(242)
 → java.lang.Integer

```
method_pool.lookup(314)
```

ORACLE

```
java.lang.Math:pow()
```

© 2013 Oracle Corporation



#### Classes, Methods, Stack Traces, Threads, Thread Groups, Strings

ORACLE

© 2013 Oracle Corporation

# **Checkpoints - revisit**

### Events + Constant Pools + Event Metadata = Checkpoint

ORACLE

© 2013 Oracle Corporation

# **Checkpoints - revisit**

checkpoint

events constant pools

meta-data

ORACLE

© 2013 Oracle Corporation

# **Checkpoints - revisit**

#### Contain everything required to parse the events prior to the checkpoint

© 2013 Oracle Corporation

# **More information**

Whitepaper

http://www.oracle.com/missioncontrol

User Guide

http://docs.oracle.com/javase/7/docs/technotes/guides/jfr/index.html

Forum

http://forums.oracle.com/community/developer/english/java/ java\_hotspot\_virtual\_machine/java\_mission\_control

© 2013 Oracle Corporation



ORACLE

© 2013 Oracle Corporation

## Thank you

© 2013 Oracle Corporation

103

Monday, September 30, 13