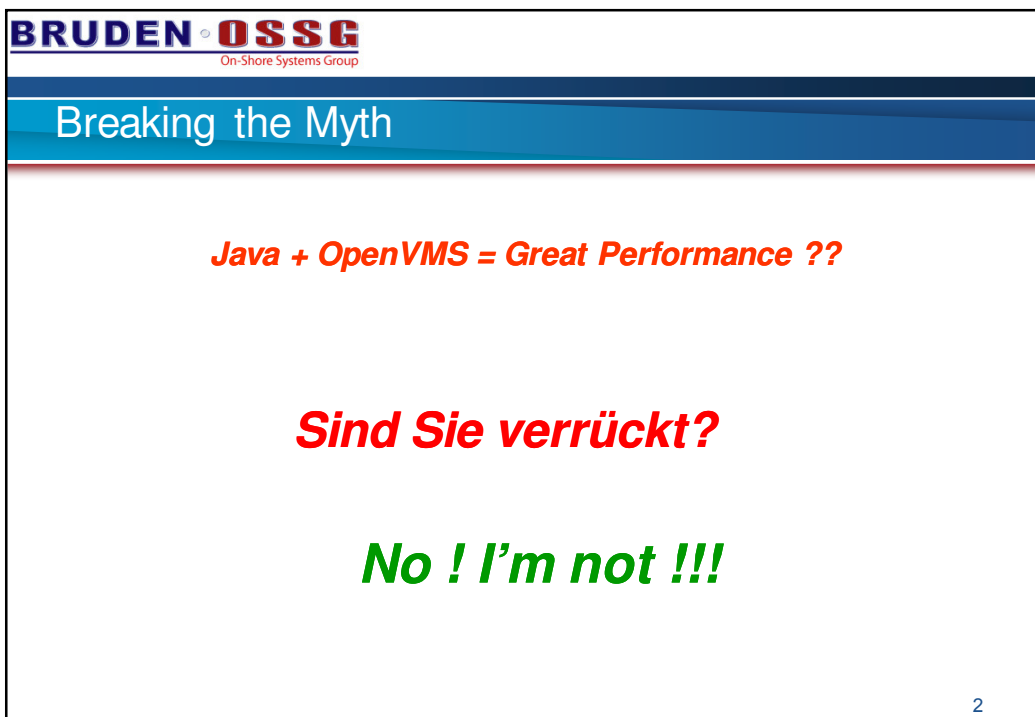




BRUDEN **OSSG**
On-Shore Systems Group

Guy Peleg
Senior Member of the Technical Staff
Director of EMEA Operations
guy.peleg@bruden.com



BRUDEN **OSSG**
On-Shore Systems Group

Breaking the Myth

Java + OpenVMS = Great Performance ??

Sind Sie verrückt?

No ! I'm not !!!

2

BRUDEN **OSSG**
On-Shore Systems Group

Breaking the Myth

- Performance Guidelines
 - **OpenVMS V8.3 running on IA64**
 - **Tune your system**
 - **Java 5**
 - **Tune the Java environment**
 - **Profile your application**

3

BRUDEN **OSSG**
On-Shore Systems Group

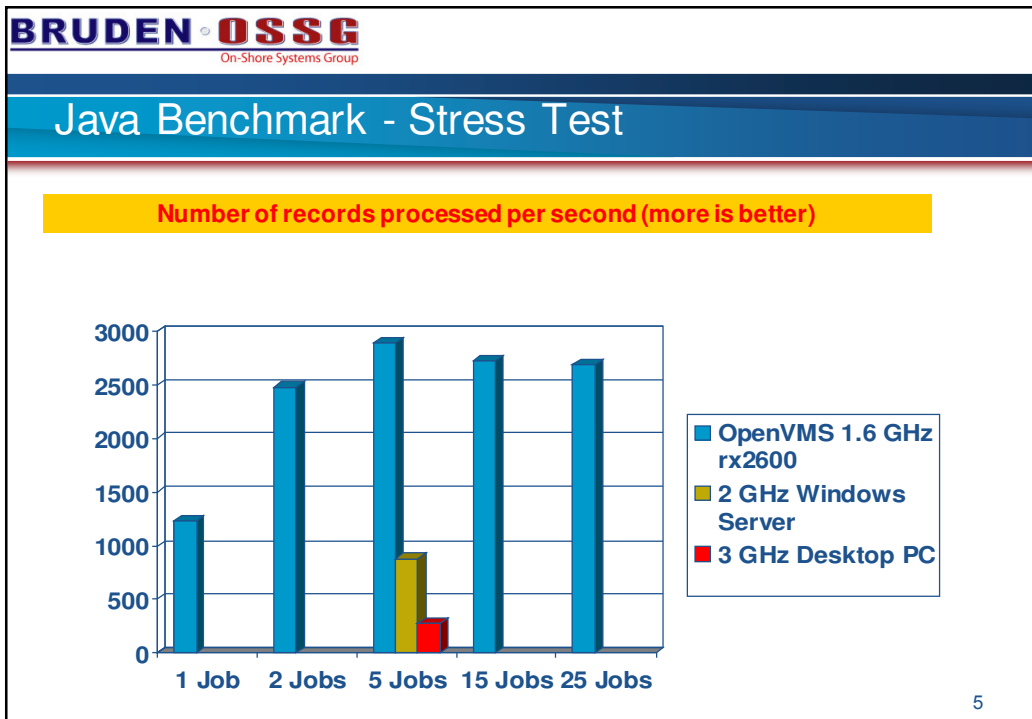
Java Benchmark

- Batch processing application
- 100% pure Java

Jobs	1GHz AlphaServer ES45 (tuned)	1.6GHZ rx2600 (not tuned)	1.6 GHZ rx2600 (tuned)	3 GHZ Desktop PC
1 Job	~70	~40	~28	~25
3 Jobs	-	-	~48	~55

Seconds to process 60,000 application records (less is better)

4



- BRUDEN** **OSSG**
On-Shore Systems Group
- ## Java on OpenVMS
- Optimal performance requires:
 - Java 5
 - HotSpot on Itanium
 - FastVM on Alpha
 - OpenVMS V8.3
 - ODS-5
 - HotSpot Major Features
 - Fast thread synchronization
 - Adaptive compilation
 - "Hotspots" get dynamically compiled
 - Code changes during lifetime of application
 - Generational garbage collector
- 6

BRUDEN **OSSG**
On-Shore Systems Group

Performance Characteristics

- HotSpot on Itanium is **MUCH** better than FastVM on Alpha
- Creating a JVM takes 3-5 seconds
 - Not suitable for short & frequent activations
- CPU bound performance 2-3 times slower than Windows
- Excellent I/O performance
 - NIO package provides same I/O performance as C
- Calling native O/S services requires JNI
 - Performance sensitive tasks

7

BRUDEN **OSSG**
On-Shore Systems Group

CPU bound Benchmarks

- rx2600 1.3 GHz
- CPU intensive / No I/Os performed

Test	C++	Java
<i>Fibonacci</i>	22.72	22.90
<i>Object Instantiation</i>	55.80	1:07.90
<i>String Concatenation</i>	1.44	2.96

Note: Tests do not take full advantage of the HotSpot technology

8

BRUDEN On-Shore Systems Group **OSSG**

Tuning Java – 30,000 feet overview

- General system tuning
- Install the main Java images resident. Use RAM disk for main classes, jars and wars.
- Process quotas & system parameters
- Tune JAVA\$ logical names
- Tune CRTL logical names
- Heap sizing, garbage collector
- Tune I/O parameters
- Optimize Java code
- Compiled code for performance critical tasks (JNI)

9

BRUDEN On-Shore Systems Group **OSSG**

System Tuning

10

BRUDEN On-Shore Systems Group **OSSG**

System Tuning

- For EVA/XP storage
 - set cluster size to be a multiple of 4.
 - 32 is the recommended value
- Install main images resident
 - pipe sh dev ddcu /file/nosys | sea sys\$pipe ".exe"
- Properly size Non-Paged Pool
- Verify pool reclamation is disabled
- Increase TCP/IP default buffer size
 - sysconfig -r inet tcp_mssdflt=1500
- SET VOLUME/NOHIGHWATER
- Global buffers on key RMS files

11

BRUDEN On-Shore Systems Group **OSSG**

System Tuning

- Avoid polluting the XFC cache
 - SET FILE/CACHE=NO_CACHE
- Set VCC_MAX_IO_SIZE to 256
- Properly size page & swap files
- SHOW FASTPATH
- LNMHASHTBL >= 8192
- Lower QUANTUM to increase throughput
- Shadowed DEGRAM devices
- LNM & FLT extensions in SDA
- LDDRIVER for testing ODS-5 on ODS-2 devices

12

BRUDEN **OSSG**
On-Shore Systems Group

Setup

13

BRUDEN **OSSG**
On-Shore Systems Group

- Install Java on an ODS-5 disk
 - Does not have to be the system disk
 - PRODUCT INSTALL /DESTINATION=xxx
- classpath
 - Unix syntax
 - `$ define classpath - "/sys$common/java/lib/jdk150_classes.zip:."`
 - Logical names are limited to 255 characters
- java\$classpath
 - OpenVMS syntax
 - `$ define java$classpath [], - disk$:[jdbc]myjdbcdrivers.jar, - wls_disk:[wlserver6_0.lib], - somedisk:[adirectory]anotherjarfile`

14

BRUDEN **OSSG**
On-Shore Systems Group

HelloWorld Example (on OpenVMS)

```
$ type myworld.java
class MyWorld {
    public static void main (String args[]) {
        System.out.println("Hello World");
    }
}
$ javac MyWorld.java

$ Java myworld.class
Can't find class myworld.class

$ Java MyWorld
Can't find class myworld

$ Java "MyWorld"
Hello World
```

15

BRUDEN **OSSG**
On-Shore Systems Group

Java™ is Case-Sensitive

- Java™ is very case-sensitive
- To Avoid quotes and really get MyWorld.class
 - \$ DEFINE DECC\$ARGV_PARSE_STYLE ENABLE
 - \$ DEFINE DECC\$EFS_CASE_PRESERVE ENABLE
 - \$ SET PROC/PARSE=EXTENDED

16

BRUDEN On-Shore Systems Group **OSSG**

Stream_LF vs Other Types

- The JDK uses DEC 'C' RTL's I/O routines
 - Non-STMLF files might work...
 - HUGE performance issue
 - 4 minutes vs 4 seconds to read a file
 - True for .java files as well
 - For best results, always verify that the files are STMLF
- ASCII file to stream_lf
 - CONVERT/FDL=[JAVA\$150.COM]STREAM_LF.FDL input output
- Converting Binary files (.class, .jar, .zip)
 - SET FILE/ATTR=(RFM:STMLF,RAT:CR,MRS:0,LRL:32767)
 - jar tf <jarfilename> to verify format

17

BRUDEN On-Shore Systems Group **OSSG**

Resident Images & RAM disks

18

BRUDEN On-Shore Systems Group **OSSG**

Resident images

- Candidates to be installed resident
 - [JAVA\$150.BIN]JAVA\$JAVA.EXE
 - [JAVA\$150.JRE.LIB.IA64.HOTSPOT]JAVA\$HOTSPOT_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$JAVA_VMS_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64.NATIVE_THREADS]JAVA\$HPI_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$VERIFY_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$JAVA_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$ZIP_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$NET_SHR.EXE
 - [JAVA\$150.JRE.LIB.IA64]JAVA\$NIO_SHR.EXE

19

BRUDEN On-Shore Systems Group **OSSG**

Resident images

- Properly size GH region to fit all images
 - GH_RES_CODE
 - GH_RES_DATA
- INSTALL ADD /RESIDENT file_spec
 - /SHARE=ADDRESS where possible
 - VMS83I_INSTALL-V0100 to force resident images back to S0/S1 space

20

BRUDEN On-Shore Systems Group **OSSG**

RAM Disks

- DEGRAM is integrated with the base O/S
- Included with EOE package or individual PAK
- \$ MC MDMANAGER
- CREATE DISK/CAPACITY=NUMBER_OF_BLOCKS
- Initialize & Mount the new device
- Copy key classes / JARs / WARs to the newly created device
- Possibly shadow with local disk
 - The shadow server is smart enough to read from memory

21

BRUDEN On-Shore Systems Group **OSSG**

Process Quotas and SYSGEN Parameters

22

BRUDEN On-Shore Systems Group **OSSG**

Minimum process quotas

- Fillm – 4096
- WSDEF – 2048 (1MB)
- WSQUO – 4096 (2MB)
- WSEXTENT – 16384 (8MB)
- DIOLM – 500
- BIOLM – 500
- TQELM - 500
- SYSGEN CHANNELCNT – 4096
- SYSGEN WSMAX - 16384

23

BRUDEN On-Shore Systems Group **OSSG**

Process Quotas

- PGFLQUOTA – 2097152
 - Matches unix default supports upto 512MB heap
- Set PGFLQUOTA to twice the heap size

$pgflquota = (2 * HEAP_IN_MB * 1024 * 1024) / 512$

- BYTLM – 500000
- QUANTUM / AWSTIME

24

BRUDEN On-Shore Systems Group **OSSG**

Process Quotas

- Inspire to fit the entire heap into physical memory
 - Increase WS*
 - Large heap and small working set will cause excessive paging activity
 - Do not oversize the working set
 - Java will use what is given to it regardless of actual requirements
 - Start with 200,000 to 400,000 pagelets (depending on number of users and available memory)
 - Size your pagefiles
- Monitoring Quotas consumption on V8.3
 - 'Q' option in SHOW PROC/CONTINUOUS
- [JAVA\$150.COM]JAVA\$CHECK_ENVIRONMENT

25

BRUDEN On-Shore Systems Group **OSSG**

Logical Names


26



Setting up Logical Names

- The run-time behavior of Java is controlled by a set of logical names
- @[JAVA\$150.COM]JAVA\$CONFIG_WIZARD
 - Generates JAVA\$CONFIG_SETUP
 - MUST be executed after JAVA\$150_SETUP to avoid overriding selection

27



Logical Names of Interest

- JAVA\$FILENAME_CONTROLS
 - Controls UNIX-Style mapping algorithms
 - To enable all options – set logical to -1
 - To disable all options – set logical to 0
 - Best option for performance
 - Filenames with multiple dots require setting DECC\$EFS_CHARSET
- JAVA\$SHOW_FILENAME_MAPPING
 - See file mapping as they occur
- Avoid setting JAVA\$FILE_OPEN_MODE
 - Results in unnecessary synchronous I/Os
 - If required set DECC\$FILE_SHARING to ENABLED

28

BRUDEN On-Shore Systems Group **OSSG**

Logical names of interest

- **JAVA\$DISABLE_MULTIDOT_DIRECTORY_STAT**
 - Avoid secondary stat() call when class can't be found in a certain path
 - Only valid if you really do not have multi-dot directories
- **JAVA\$CACHING_INTERVAL <interval in seconds>**
 - stat() information is cached
 - Some information about the files may be answered from cache
 - Cache is invalidated at the end of interval or when a write to the file occurs

29

BRUDEN On-Shore Systems Group **OSSG**

Logical names of interest

- **JAVA\$DAEMONIZE_MAIN_THREAD**
 - Avoid starving the main Java thread for CPU time when heavy use of ASTs is made.
 - New thread is created to run the main Java thread.
- **JAVA\$READDIR_CASE_DISABLE**
 - Turns off the case sensitivity filename extraction feature
 - Feature not required on ODS-5 if you ensure filenames are created/named with the correct case
 - Improves performance of scanning directories for .class files

30

BRUDEN On-Shore Systems Group **OSSG**

Logical Names of interest

- **JAVA\$OMIT_CASE_CHECK**
 - Increases performance of JAVAC and JAR commands
- **Displaying Graphics Intensive applications**
 - -Dsun.java2d.pmosffscreen=false
 - Do not store images in pixmaps by default
- **JAVA\$TIMED_READ_USE_QIO**
 - Use QIO & ASTs instead of select()
 - Better performance
 - Look at tcpip show comm/mem search for the number of SELECT structure

31

BRUDEN On-Shore Systems Group **OSSG**

Logical Names of interest

- Put files with largest number of classes first in **JAVA\$CLASSPATH**
- **JAVA\$FORK_PIPE_STYLE**
 - 1 Default mailbox driver
 - 2 Uses sockets for buffering data (requires TCP/IP) between heap and parent process
- **DECC\$ENABLE_GETENV_CACHE**
 - CRTL will cache entries returned by getenv()

32

BRUDEN On-Shore Systems Group **OSSG**

Logical Names of interest

- DECC\$LOCALE_CACHE_SIZE
 - Memory in bytes for caching locale data
 - Default is 0
- DECC\$TZ_CACHE_SIZE
 - Number of time zones that can be held in memory
 - Default is 2

33

BRUDEN On-Shore Systems Group **OSSG**

Heap Sizing & GC

34

BRUDEN **OSSG**
On-Shore Systems Group

Memory Management

- Heap
 - Large pre-allocated memory pool
- Lazy management
 - Alloc (New), no free
 - Best performing Java applications need large physical memory system
 - When we run out of space, then we cleanup (Collected unused memory)
- Speed now, pay later
 - Applications usually stall during a collection

the Heap



35

BRUDEN **OSSG**
On-Shore Systems Group

Garbage Collection

- C (malloc & free), C++ (new & delete)
- Java (new and Garbage Collection)
- (GC) reclaims the heap space previously allocated to objects no longer needed. The process of locating and removing those dead objects can stall your Java application while consuming as much as 25 percent of throughput.
- VMS Level - Poorly tuned UAF account
 - Initial startup paging (wsinc)
 - Heavy paging after initial startup (wsextent)

36

BRUDEN On-Shore Systems Group **OSSG**

Heap & GC

- Ergonomics
 - New feature in V1.5
 - Attempts to match the best selection of GC, Heap size and runtime compiler
 - Assumes that large applications run on large machines
- “The J2SE platform shields the developer from the complexity of memory allocation and garbage collection”
- No good deed goes unpunished

37

BRUDEN On-Shore Systems Group **OSSG**

Heap & GC

- The heap is arranged in three generations
 - Young
 - Tenured
 - Perm
- Three types of Garbage Collectors
 - Serial Collector
 - Throughput Collector
 - Concurrent low pause Collector

38

BRUDEN On-Shore Systems Group **OSSG**

Heap & GC

- Check for GC - run the application with `-verbose:gc`
 - `-XX:+PrintGCDetails`
- Preferably avoid GC completely, increase the heap and process quotas if needed
- Total heap size
 - Bounded below by `-Xms` and above `-Xmx`
 - For best performance `-Xms == -Xmx`
 - (today) 1500MB is hard coded limit for OpenVMS
- Young generation should be sized $3/8^{\text{th}}$ of heap size
 - `-XNewSize` and `-XMaxNewSize` bound the young generation size (or `-XNewRatio`)

39

BRUDEN On-Shore Systems Group **OSSG**

Heap & GC

- Rules of thumb for sizing the Young Generation:
 - Decide the total amount of memory you can afford to give the JVM
 - Unless you find programs with excessive major collection or pause times, grant plenty of memory to the young generation.
 - Increasing the young generation becomes counterproductive at half the total heap size
 - Be sure to increase the young generation as you increase the number of processors, since allocation can be parallelized.

40

BRUDEN On-Shore Systems Group **OSSG**

GC

- Parallel (throughput) collector
 - `-XX:+UseParallelGC`
 - Intended for applications with large number of processors
 - Uses multiple threads to execute minor collection
 - `-XX:MaxGCPauseMillis=<nnn >`

- Concurrent collector
 - `-XX:+UseConcMarkSweepGC`
 - For applications that would benefit from shorter GC and can afford to share processor resources with the GC when the application is running
 - Optimal results for applications with tenured generations of a modest size

- Serial collector
 - `-XX:+UseSerialGC`
 - Smaller applications and systems

41

BRUDEN On-Shore Systems Group **OSSG**

GC

- `-Xnoclassgc`
 - Disables class garbage collector

- `-Xss<size>` - Specifies the size of stack for each new Java thread.
 - Default size is 128KB
 - Increase if `java.lang.StackOverflowException` is seen

- `-XX:SurvivorRation=<size>`
 - Ratio of eden/survivor space size. Default is 8
 - 100MB heap – space reserved for object to survive GC is 6.25MB (100MB/8/2 survivor spaces)
 - Improves performance when new space is large and/or when the application keeps a very low percentage of objects.

42

BRUDEN On-Shore Systems Group **OSSG**

GC tips

- Reduce object life time by code inspection
 - Remove references when not required
 - Can do this explicitly with
objectref = null;
- -XX:+UseAgressiveHeap
 - Requires min of 256MB RAM
 - Overall heap will be around 3850MB
 - GC deferred as long as possible
 - Not suited to multi-app servers
 - Not supported by OpenVMS

43

BRUDEN On-Shore Systems Group **OSSG**

Tuning I/O

44

BRUDEN On-Shore Systems Group **OSSG**

Tuning I/O

- I/O's are native in Java and eventually reach RMS
- RMS tuning improves Java I/O performance.

45

BRUDEN On-Shore Systems Group **OSSG**

Optimizing your Java code

46

BRUDEN On-Shore Systems Group **OSSG**

Optimize your code

- MONITOR SYSTEM
- MONITOR PAGE
- MONITOR MODES
 - 100% user mode for compute bound applications
 - CPU speed is critical
 - Memory latency
 - Montecito HyperThreads

- JAVAC –"O"
 - Inline certain method calls
 - Only method declared private, static or final can be considered for inlining.

47

BRUDEN On-Shore Systems Group **OSSG**

Optimize your code

- Use the shift operator to multiply and divide integers by the power of 2
 - $X \gg 2$ instead of $x/4$
 - $X \ll 1$ instead of $X*2$

- Common sub expression elimination
 - Replace
 - `double x = d * (lim/max) * sz;`
 - `double y = d * (lim/max) * sy;`
 - With
 - `double depth = d * (lim/max);`
 - `double x = depth * sx;`
 - `double y = depth * sy;`

48

BRUDEN **OSSG**
On-Shore Systems Group

Optimize your code

- Comparing to zero is faster
 - Doesn't require N to be loaded
 - for (i=0; i<N; i++) should be replaced with for (i=N; --i >=0)
- Buffer, vector, map, table, heap, set....all support an initial capacity parameter in their constructors

49

BRUDEN **OSSG**
On-Shore Systems Group

Strings Vs. StringBuffer

```
public class test1
{
    public static void main (String[] args)
    {
        String s = "[";

        for (int i=1; i<=2000; i++)
        {
            if (i>1)
                s += ",";
            s += Integer.toString (i,10);
        }
        s += "]";

        System.out.println(s);
    }
}
```

50

BRUDEN **OSSG**
On-Shore Systems Group

String Vs. StringBuffer

```
public class test2
{
    public static void main (String[] args)
    {
        StringBuffer sb = new StringBuffer ();

        sb.append ("[");

        for (int i=1; i<=2000; i++)
        {
            if (i>1)
                sb.append (",");
            sb.append ( Integer.toString (i,10));
        }
        sb.append("]");
        String s = sb.toString();

        System.out.println(s);
    }
}
```

51

BRUDEN **OSSG**
On-Shore Systems Group

Profiling

- Profile, profile and profile some more...
 - java -prof myClass
 - Look at java.prof
 - java -Xprof

52




Partial output from Java.prof

```

count callee caller time
4041 java.lang.String.length()I java.lang.AbstractStringBuilder.append(Ljava/lang/String;)Ljava/lang/AbstractStringBuilder; 41
4039 java.lang.String.getChars(II[C)V
    java.lang.AbstractStringBuilder.append(Ljava/lang/String;)Ljava/lang/AbstractStringBuilder;
4018 java.lang.AbstractStringBuilder.append(Ljava/lang/String;)Ljava/lang/AbstractStringBuilder;
    java.lang.StringBuffer.append(Ljav
4001 java.lang.StringBuffer.append(Ljava/lang/String;)Ljava/lang/StringBuffer; test2.main([Ljava/lang/String;)V 306
2000 java.lang.Integer.toString(II)Ljava/lang/String; test2.main([Ljava/lang/String;)V 200
2000 java.lang.Integer.toString(I)Ljava/lang/String; java.lang.Integer.toString(II)Ljava/lang/String; 148
2000 java.lang.String.<init>(II[C)V java.lang.Integer.toString(I)Ljava/lang/String; 21
2000 java.lang.Integer.getChars(II[C)V java.lang.Integer.toString(I)Ljava/lang/String; 24
2000 java.lang.Integer.toString(I)I java.lang.Integer.toString(I)Ljava/lang/String; 26
242 java.lang.AbstractStringBuilder.append(C)Ljava/lang/AbstractStringBuilder;
    java.lang.StringBuilder.append(C)Ljava/lang/StringBu
237 java.lang.String.length()I sun.net.www.ParseUtil.decode(Ljava/lang/String;)Ljava/lang/String; 5
232 java.lang.String.charAt(I)C java.io.UnixFileSystem.normalize(Ljava/lang/String;)Ljava/lang/String; 2
231 java.lang.String.charAt(I)C sun.net.www.ParseUtil.decode(Ljava/lang/String;)Ljava/lang/String; 5
231 java.lang.StringBuilder.append(C)Ljava/lang/StringBuilder;
    sun.net.www.ParseUtil.decode(Ljava/lang/String;)Ljava/lang/String; 4
33 java.lang.String.indexOf(II)I java.lang.String.indexOf(I)I 0
32 java.lang.String.startsWith(Ljava/lang/String;I)Z java.lang.String.endsWith(Ljava/lang/String;)Z 1
28 java.lang.AbstractStringBuilder.expandCapacity(I)V
    java.lang.AbstractStringBuilder.append(Ljava/lang/String;)Ljava/lang/Abstract
25 sun.misc.VM.allowArraySyntax()Z java.lang.ClassLoader.checkName(Ljava/lang/String;)Z 0
25 java.lang.String.indexOf(I)I java.lang.ClassLoader.checkName(Ljava/lang/String;)Z 1
25 java.lang.String.length()I java.lang.ClassLoader.checkName(Ljava/lang/String;)Z 1

```


53



JRate

- *JRate is the Java™ Runtime Analysis Toolkit. Its purpose is to enable developers to better understand the runtime behavior of their Java™ programs. The term "behavior" includes, but is not limited to performance profiling.*
- JRate can:
 - accumulate timing statistics (a few ways)
 - create trace logging
 - track rate methods are called over time
 - track the response time of methods over time

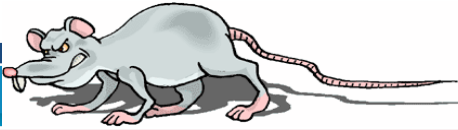
<http://jrate.sourceforge.net/>



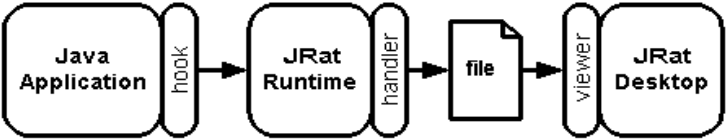
54

BRUDEN **OSSG**
On-Shore Systems Group

JRat



- JRat's design is based on three types of components:
 - Hooks enable an application to be monitored at runtime
 - Events generated by hooks are sent to the JRat Runtime where processing is delegated to one or more handlers
 - Handlers produce output files during runtime and at shutdown
 - Each output file identifies the Viewer component that can be used to navigate the data



```
graph LR; A[Java Application] -- hook --> B[JRat Runtime]; B -- handler --> C[file]; C -- viewer --> D[JRat Desktop];
```

55

BRUDEN **OSSG**
On-Shore Systems Group

Using JNI

56

BRUDEN **OSSG**
On-Shore Systems Group

Simple JNI example – Java side

```
$ ty sqrt.java
class SQRT {

public native int SQRTC (int number);

static {
    System.loadLibrary("SQRT_C");
}

public static void main (String[] args) {

    System.out.println ("This is Java...calling the C routine");

    int result = new SQRT () .SQRTC (Integer.decode(args[0]));

    System.out.println ("Result is " + result);
}
}
```

57

BRUDEN **OSSG**
On-Shore Systems Group

Simple JNI Example – C side

```
$ ty sqrt_c.c
#include "SQRT.h"
#include <math>

JNIEXPORT jint JNICALL Java_SQRT_SQRTC (JNIEnv *env, jobject this, jint arg1)
{
    printf ("This is C, input is %d\n",arg1);
    return sqrt (arg1);
}
$
```


58



Building JNI applications

- Building the SQRT example
 - javac sqrt.java
 - Javah -jni sqrt
 - Compile the C program
 - cc
 - /prefix=all/float=iieee/iieee=denorm/defin=JIT_OPTION/names=(short,as_is)/reent=multithread/includ=disk\$up:[java_ia64.java\$150.include...]
 - link/share C_IMAGE
 - Define a logical name to point to the C image
 - Run the java application

59



JNI Example – Manipulating Objects

```

$ ty main.java
class Main
{
    public static void main(String[] args)
    {
        JavaFields jf = new JavaFields();

        jf.manip();

        System.out.println();
        System.out.println("After native call:");
        System.out.println("public int i = " + jf.i);
        System.out.println("public float f = " + jf.f);
        System.out.println("static public int si = " + jf.si);
        System.out.println("private int pi = " + jf.getpi());
    }
}

```

60

BRUDEN **OSSG**
On-Shore Systems Group

JNI Example – Manipulating Objects

```
$ ty JavaFields.java
class JavaFields
{
    public int i = 1;
    public float f = 2.45f;
    static public int si = 3;
    private int pi = 5;

    public native void manip();
    public int getpi(){ return pi; }

    static
    {
        System.loadLibrary("JavaFieldsImp");
    }
}
```

61

BRUDEN **OSSG**
On-Shore Systems Group

JNI Example – Manipulating Objects

```
$ ty JavaFieldsImp.c
#include <stdio.h>
#include "JavaFields.h"

JNIEXPORT void JNICALL Java_JavaFields_manip(JNIEnv *env, jobject obj)
{
    jclass cls;
    jfieldID fid_i, fid_pi, fid_si, fid_f;
    jint i, si, pi;
    jfloat f;

    // Get the class associated with this object (class JavaFields)
    cls = (*env)->GetObjectClass(env, obj);

    // The the ID for variable 'i'
    fid_i = (*env)->GetFieldID(env, cls, "i", "I");

    // Get the value of 'i'
    i = (*env)->GetIntField(env, obj, fid_i);

    // The the ID for variable 'f'
    fid_f = (*env)->GetFieldID(env, cls, "f", "F");

    // Get the value of 'f'
    f = (*env)->GetFloatField(env, obj, fid_f);
}
```

62

BRUDEN On-Shore Systems Group **OSSG**
On-Shore Systems Group

JNI Example – Manipulating Objects

```
// The the ID for variable 'si'
fid_si = (*env)->GetStaticFieldID(env, cls, "si", "I");

// Get the value of 'si'
si = (*env)->GetStaticIntField(env, cls, fid_si);

// The the ID for variable 'pi'
fid_pi = (*env)->GetFieldID(env, cls, "pi", "I");

// Get the value of 'pi' (No protection from native code!)
pi = (*env)->GetIntField(env, obj, fid_pi);

printf("From native code:\n");
printf("public int i = %i\n", i);
printf("public float f = %f\n", f);
printf("static public int si = %i\n", si);
printf("private int pi = %i\n", pi);

// Set the value of 'i'

(*env)->SetIntField(env, obj, fid_i, 100);

// Set the value of 'f'

(*env)->SetFloatField(env, obj, fid_f, 123.45);

// Set the value of 'si'
```

63

BRUDEN On-Shore Systems Group **OSSG**
On-Shore Systems Group

BRUDEN-OSSG Services

- BRUDEN can help you improve the performance of your Java applications
- No Results = No Expense

64

BRUDEN **OSSG**
On-Shore Systems Group

Questions?

BRUDEN-OSSG thanks you for attending this session.

See us at www.BRUDENOSSG.com for:

- *Performance analysis*
 - *(Performance Results Or No Expense)*
- *Porting assistance*
- *Special OPS (OpenVMS Programming Services)*

65