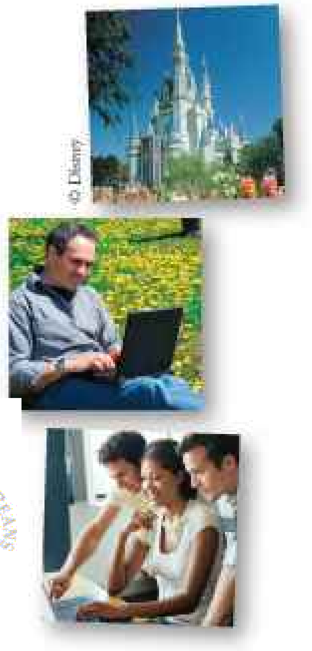
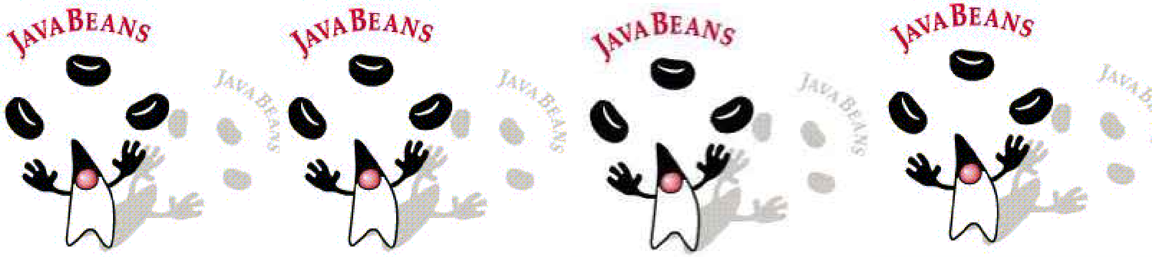
IBM JVM SDK V6 Introduction and System z Update

Theresa Tai
IBM System z New Technology Center
Poughkeepsie, New York
ttai@us.ibm.com

February 26, 2008
Session 8365





Trademarks

IBM, AIX, CICS, DB2, IMS, z/OS, OS/390, S/390, System z, VisualAge, WebSphere Application Server, WebSphere Studio, z/VM - are trademarks or registered trademarks of the IBM Corporation

Sun, Sun Microsystems, JavaSoft, Java, JavaBeans, JDK, Java 2 Micro Edition, J2ME, Java 2 Standard Edition, J2SE, Java 2 Enterprise Edition, J2EE - are trademarks or registered trademarks of Sun Microsystems Inc.

Disclaimer

Performance charts: The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

Content

- IBM SDK V6 Technology Review
  - The Best of SDK V6 Functions / Features / Performance
- SDK Currency
- 31bit and 64bit Environment Considerations
- IBM JVM – Major Building Blocks
- 2008 Focus for Java on System z
  - Delivering SDK V6
  - Supporting Key IBM Middleware and Operating System
  - IBM Specialty Engine zAAP Enhancements for Java Workloads
  - Looking Ahead: What's New and Exciting
- Supplementary Materials on SDK V5 and V6
- Summary

SHARE
Technology · Connections · Results

IBM SDK V6

IBM SDK V6 on System z

- IBM SDK for z/OS, Java Technology Edition, V6
  - Product number is 5655-R31
  - The subscription and service number is 5655-I48
  - Supporting 31-bit and 64-bit
  - eGA December 21, 2007 and SDF delivery January 18, 2008
- System Requirements
  - z/OS V1.7 or higher
  - z9 BC, z9 EC, z990, and z890
- Most Java applications executed on IBM 31-bit or 64-bit for z/OS SDK V5 are expected to run unchanged on SDK V6, with the exception of deprecated SDK V5 APIs

SDK V6 Reference Materials

- Prerequisites
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/j6prereq64.html
- Download SDK V6
  - http://www.ibm.com/java (z/OS)
  - http://www.ibm.com/developerworks/java/jdk/linux/download.html (zLinux)
- SDK V6 APIs
  - http://java.sun.com/javase/6/docs/api/
- Deprecated APIs
  - http://java.sun.com/javase/6/docs/api/deprecated-list.html
- Incompatibilities, visit Sun site:
  - http://java.sun.com/javase/6/webnotes/compatibility.html#incompatibilities
- Restrictions and Other Considerations
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/j6restrict31.html
- SDK V6 key features
  - http://java.sun.com/javase/6/features.jsp
- IBM SDK V6 Guide
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/pdf/java6/sdkguide.zos.pdf

SDK V6 Java Specification

- The IBM SDK for Java Version 6 is compliant with Sun's Java specification
  - JSR 118: Web Services Metadata for the Java Platform
  - JSR 173: Streaming API for XML
  - JSR 199: Java Compiler API
  - JSR 222: Java Architecture for XML Binding (JAXB) 2.0
  - JSR 221: JDBC 4.0 API Specification
  - JSR 223: Scripting for the Java Platform
  - JSR 224: Java API for XML-based Web services (JAX-WS) 2.0
  - JSR 250: Common Annotations for the Java Platform
  - JSR 269: Pluggable Annotation Processing API

IBM SDK V6 Highlight

- IBM Java 6 SDK focuses on platform Stability, Performance and Diagnostics
  - New serviceability APIs
  - New EVTK (Extensible Verbose Toolkit) RAS tooling
  - Enhanced Diagnostic Information
  - JVM Tool Interface (JVMTI) Extension
  - JIT Performance Improvement
- RACF Writable Keyring Support
  - Write access to RACF keyrings through the JCECCARACFKS and JCERACFKS KeyStores
- XML JAXP 1.4 support and XSLT processing
- Java Batch Component Directory Change
  - Executable DLLs in new directory

EVTK has been renamed to "GCMV"

"**IBM Monitoring and Diagnostic Tools for Java – GC and Memory Visualizer**"

SDK 6 Serviceability Tool Enhancement

- Diagnostic Tool Framework for Java (DTFJ) supports
  - Javadumps, System dump viewer based on DTFJ
- Extensible Verbose Toolkit for analyzing GC logs
  - Provides data visualization and customized reports with GC tuning recommendations
- Improved diagnostic information
  - Native stack traces in Javadumps and console dumps
  - Consolidated command-line tools under –Xcheck
  - Heapdump compression and performance improvements
  - Documentation updates
    - Diagnostics Guide, documentation launchpad, searchable documentation
- The result report(s) are formatted into either an HTML or text document

Serviceability Tools for Java Applications

- Memory Dump Diagnostic
  - OutOfMemoryError
    - Identify data structures that are likely causes of memory leaks
  - Analyze heap growth
    - Identify objects with large size differences
- Dump Analyzer for Java (based on DTFJ)
  - A tool to analyze Java dumps and system dumps
- ThreadAnalyzer
  - Process Java dump files (javacores and thread dumps)
  - Designed with WebSphere Application Server in mind
    - Provice statistics based on thread behaviors that matter to the application server
  - Identifies deadlock situations
- Extensible Verbose Toolkit
  - Enable you to investigate memory-based Java performance problems

GC Extensible Verbose Toolkit

- The Extensible Verbose Toolkit (EVTK) is a visualizer for GC data
- It analyzes the verbose GC logs to
  - Diagnosing a memory leak
  - Sizing a heap space
  - Application performance problems
  - Estimating application throughput and response times
- A little effort will get you a better understanding of your application characteristics
- Enable you to detect application memory usage problem and improve performance
- The EVTK (renamed to GCMV) is a free download within the IBM Support Assistant (ISA)

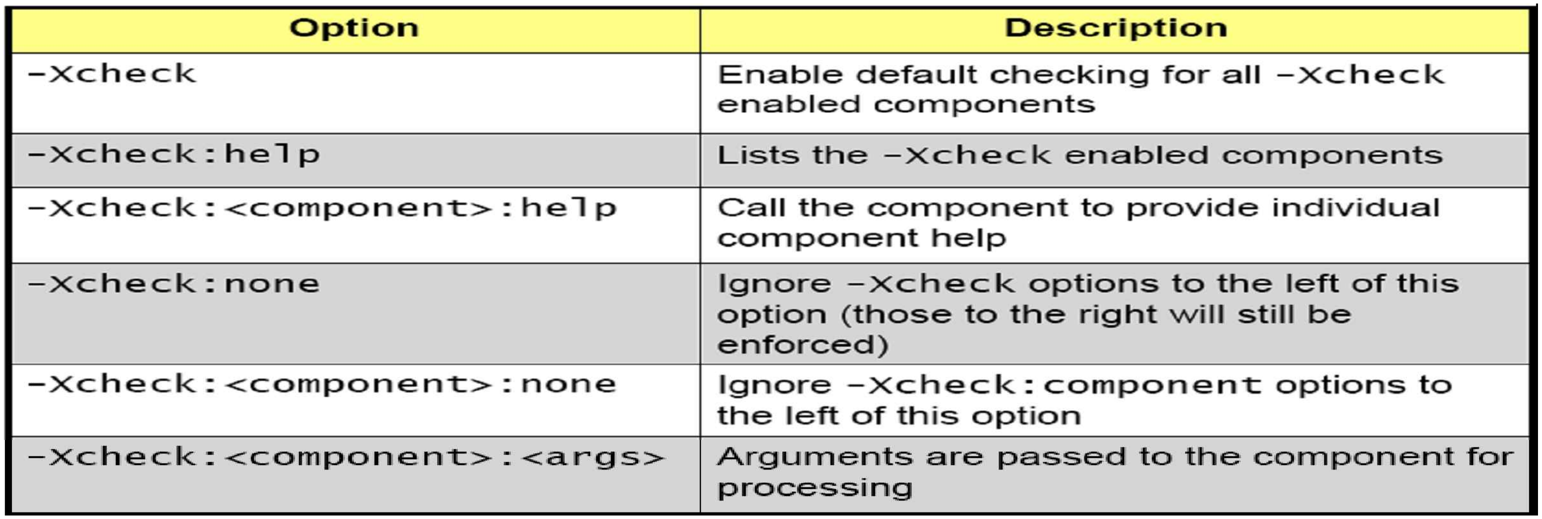
Note: EVTK has been renamed to “IBM Monitoring and Diagnostic Tools for Java”
GCMV – **GC** and **M**emory **V**isualizer

IBM Monitoring and Diagnostic Tools for Java

Usage Scenarios

- Investigate performance problems
  - Long pausing or unresponsiveness
- Evaluate Java heap size
  - Check heap occupancy and adjust accordingly
- GC policy tuning
  - Evaluate GC characteristics, compare different GC policies
- Isolate memory growth problems
  - Heap growth over time
  - Determine the general health of a Java application

The –Xcheck Parameter



Option	Description
`-Xcheck`	Enable default checking for all -Xcheck enabled components
`-Xcheck:help`	Lists the -Xcheck enabled components
`-Xcheck:<component>:help`	Call the component to provide individual component help
`-Xcheck:none`	Ignore -Xcheck options to the left of this option (those to the right will still be enforced)
`-Xcheck:<component>:none`	Ignore -Xcheck:component options to the left of this option
`-Xcheck:<component>:<args>`	Arguments are passed to the component for processing

- -Xcheck:jni replaces –Xrunjnichk
- -Xcheck:memory replaces –memorycheck
- -Xcheck:gc replaces –Xrunj9gcchk24
- -Xcheck:classpath
  - classpath checking for missing or unreadable classes
- Old command-line options are no longer supported, and will be ignored

XML JAXP 1.4 Support

- IBM implementation of JAXP 1.4 in Java SDK 6
  - Parsing / datatype / validation services provided by XML4J / XL XP-J
  - Transformation / XPath services provided by XLTXE-J
- JAXP 1.4
  - Mix of open source and proprietary components
  - Introduction of StAX (pullparsing) APIs
  - XLTXE-J compiler is the default XSLT processor
  - New XML APIs introduced in Java SDK 6 which are not under the JAXP umbrella
    - Only JAXP 1.4 implementation supported by Toronto
    - JAXB 2.0, JAX-WS, etc.. (Sun implementations) supported by the Java Technology Center (IBM Hursely)
- XSLT Processing:
  - The new XL TXE-J XSLT compiler has been designed for performance and is now the default XSLT processor in SDK V6
  - The XL TXE-J compiler replaces the XSLT4J compiler
    - The XSLT4J interpreter is still available

XL XP-J: high-performance non-validating parser; XML4J: validating parser

Java Batch Component

- JZOS
  - Same functionality as SDK 5
  - Binaries location changes
    - {java_home}/**lib/s390**/libjzos.so (31-bit)
    - {java_home}/**lib/s390x**/libjzos.so (64-bit)
    - {java_home}/lib/ext/ibmjzos.jar
- Keep an eye on New JZOS features in SDK 6 SR1
- JRIO
  - Same functionality as SDK 5
  - Binaries location changes
    - {java_home}/**lib/s390**/libJrioOS390.so (31-bit)
    - {java_home}/**lib/s390x**/libJrioOS390.so (64-bit)
    - {java_home}/lib/ext/recordio.jar
    - **{java_home}/demo/jrio/recjava.jar**

Other Notable Items in V6

- Performance Enhancements
  - Just In Time Compiler (JIT)
    - Overall performance improvements
    - New and improved JIT optimizations
  - Heap dump performance enhancement
    - Improved dump performance
    - Able to write the dump file faster with
      - Compression – there is less data to write to file
      - Caching – crossing over to do native I/O operations less frequently
    - Writing heap dumps can be over 10x faster than SDK 5

- The JVMTI (Tool Interface) specification has been expanded
  - JVMTI replaced JVMRI (RAS Interface) as of SDK 5
  - Providing access to heap contents
  - Allowing tools to connect to a running JVM on demand
  - Expanding the interfaces to support event-based retransformation operations

Diagnostic Tooling Framework for Java (DTFJ)

- DTFJ is an API used to build Java diagnostic tools
- DTFJ is the architecture behind the Dump Analyzer
- It can process system dumps and java dumps
- The DTFJ API helps diagnostic tools access the
  - Memory locations stored in the dump
  - Relationships between memory locations and Java internals
  - Java threads running within the JVM
  - Native threads held in the dump
  - Java classes and objects that were present in the heap
  - Details of the machine on which the dump was produced
  - Details of the Java version that was being used
  - The command line that launched the JVM

IBM Support Assistant (ISA)

- An integrated serviceability workbench that can be customized
  with Java tools
  - **Memory Dump Diagnostic for Java** – processes heap dumps and helps solve OutOfMemoryErrors
  - **IBM Dump Analyzer for Java** – processes system dumps and helps diagnose crashes, deadlocks, and other problems
  - **ThreadAnalyzer** – processes java dumps (thread dumps) and helps diagnose deadlocks
  - **Extensible Verbose Toolkit** – analyzes garbage collection output
- ISA Serviceability Tools
  - Available as a free download at
    - http://www.ibm.com/software/support/isa
    - Need to download and install Java-related plug-ins
- The primary IBM toolset for solving Java problems
- Work with SDK V5 or SDK V6

31-bit and 64-bit Environment Considerations

- In 31-bit mode
  - The address space has a max of 2GB virtual memory and a 31-bit JVM
- In 64-bit mode
  - The address space has a max of 16 Exabytes of virtual memory and a 64-bit JVM
- Maximum Heap Settings (Best Practices)
  - 31-bit:
    - Linux: ~800Mb SLES7 2.4.17 Kernel or RHEL 3.0
      ~1800Mb SLES8 2.4.19 Kernel or higher (by adjusting mapped_base)
    - z/OS: ~1400Mb WebSphere Application Server ~ 800MB
  - 64-bit:
    - Linux: SDK 1.4.1, SDK 5, SDK 6
    - z/OS: SDK 1.4.2, SDK 5, SDK 6
    - Both limited to how much storage (physical memory) on your system
      - Within reason, of course

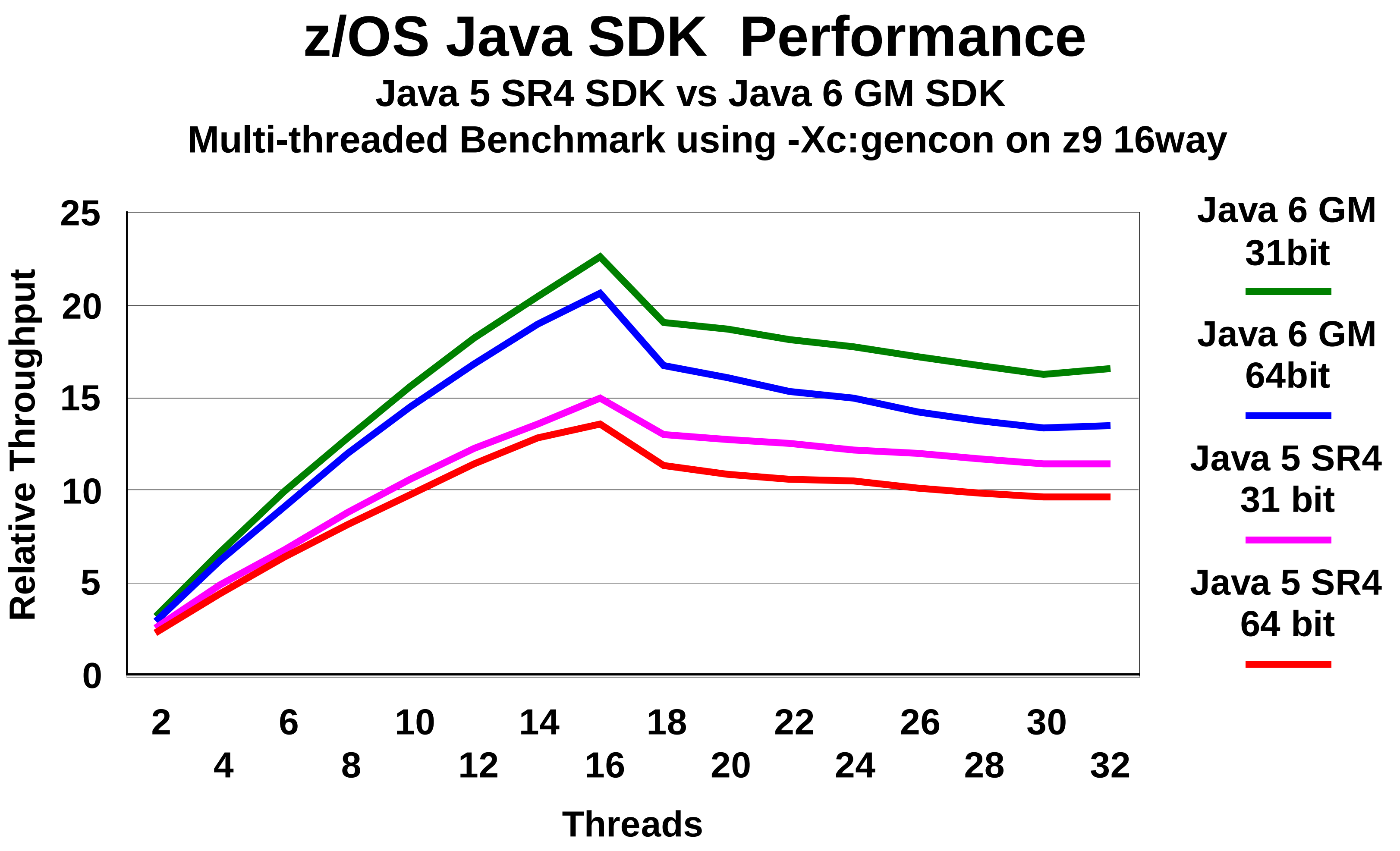
The Cost of 64-bit

- Hardware effects
  - Primarily D-cache "pressure"
  - z/Architecture supports both 31-bit and 64-bit addressability
    - Data cache is fixed size for machine
    - 64-bit pointers "twice" as large as 31-bit pointers
  - Also impacts I-cache performance
    - 64-bit instructions tend to be 6-byte instead of 2 or 4
- Software effects
  - some 31-bit instructions have no 64-bit equivalent
    - Must be implemented with series of 64-bit opcodes
    - = additional path length for same function
  - Increased cost for entry/exit linkage
    - Registers are twice as wide
  - At the current HW and SW level
    - 31-bit to 64-bit cost is ~10% for same SDK level

Running Java Applications on System z

- Moving applications to a 64-bit environment
    - IBM 64bit Java Porting Guide
        - http://download.boulder.ibm.com/ibmdl/pub/software/dw/jdk/64bitporting/64BitJavaPortingGuide.pdf
- Garbage Collection in SDK V5
    - Garbage collection policies and behavior
        - http://www-128.ibm.com/developerworks/java/library/j-ibmjava2/index.html
- Redbook
    - z/OS 64bit C/C++ and Java Programming Environment
        - http://www.redbooks.ibm.com/abstracts/redp9110.html
- IBM SDK Diagnostic Guide
    - Diagnostic documentation (including SDK V6)
        - http://www.ibm.com/developerworks/java/jdk/diagnosis/

SDK V5 and V6 Performance on z9



JZOS Batch Toolkit 2.2.1 Enhancements

- JZOS 2.2.1 contains functional enhancements such as automatic generation of record classes and complementary tools and sample code
- Support for automatic generation of record classes from Assembler DSECTs (package com.ibm.jzos.recordgen.asm)
  - See the sample JCL member DSECTGEN and the document *JZOS Assembler Record Generator Users Guide.pdf* in the doc directory
- Support for automatic generation of record classes from COBOL copybook DSECTs (com.ibm.jzos.recordgen.cobol)
  - See the sample JCL member COBGEN and the document *JZOS COBOL Record Generator Users Guide.pdf* in the doc directory
- Please visit the following site for details
  - http://www.alphaworks.ibm.com/tech/zosjavabatchtk
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/jzos/overview.html

*** * Please attend session 8368 * ***

What is Around the Corner for JZOS

- Changed PdsDirectory to read PDSE dirs in compatibility mode as well as PDS dirs
- Wrappers for CatalogSearch, LOCATE and OBTAIN
- Batch Launcher jar executables: If the first two arguments supplied to the launcher are "-jar /path/to/jarfile", the manifest of the indicated jar file will examined to determine which main() class to run
- MVS job submission and status with Java/Rexx integration
- MVS job output with Java/Rexx integration (requires z/OS 1.9 and SDSF)
- z/OS Logstream interfaces for IXGCONN (connect) and IXGWRITE (write)
- CPU time interface via the clock() C-library routine
- Process ID interfaces via the getpid() and getppid() C-library routines
- Datatype Field converters
- Process set environment via setEnv() C-library routine
- makeFifo interface via mkfifo() C-library routine

http://www.alphaworks.ibm.com/tech/zosjavabatchtk

*** * Please attend session 8368 * ***

SHARE
Technology • Connections • Results

SDK Currency

© Disney

SDK for z/OS Currency

- IBM Developer Kit for OS/390, Java 2 Technology Edition V1.3
  - Currently at SDK1.3.1 Level; Product 5655-D35
  - Build Level: February 5, 2007 UK21865 at SR 27a

EOS September 2007

- IBM SDK for z/OS, Java 2 Technology Edition, Version 1.4
  - Currently at SDK1.4.2 level; Product 5655-I56; Supported on z/OS V1.2 +
  - Build Level: 31 bit August 13, 2007 UK28320/APAR PK50801 SR9a
  - Build Level: 64 bit July 8, 2007 PTF UK27003/APAR PK42533 SR9
- IBM 31-bit and 64-bit for z/OS, Java 2 Technology Edition, Version 5
  - Product 5655-N98 and 5655-N99 for 31 & 64 bit respectively, z/OS V1.6 +
    - Build Level: 31-bit October 25, 2007, PTF UK30941/APAR PK55819 SR6b)
    - Build Level: 64-bit October 25, 2007, PTF UK30954/APAR PK55857 SR6b)
- IBM 31-bit and 64-bit for z/OS, Java 2 Technology Edition, Version 6
  - GA in January 2008
- For automatic service, Subscription and Support (5655-I48)
- No Charge Product and it is supported by the normal IBM support channels

NOTE: The EOS date for SDK 1.4.2 is September 2009
The EOS date for SDK V5 is September 2011

SDK for Linux on System z Currency

- IBM Developer Kit for Linux® Java 2 Technology Edition, Version 1.3
  - SDK1.3.1 Level
  - SuSe SLES 8, Turbo, at SR10-1

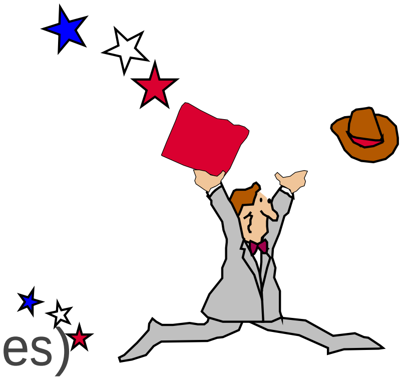
EOS September 2007

- IBM 31 and 64-bit SDK for Linux® on zSeries, Java 2 Technology Edition, Version 1.4
  - SDK 1.4.2 Level at SR 10
- IBM 31 and 64-bit SDK for Linux on zSeries, Java 2 Technology Edition, V 5
  - SDK V5 at SR 6
- IBM 31 and 64-bit SDK for Linux on zSeries, Java 2 Technology Edition. V 6
  - SDK V6 GA
- Delivery and Service
  - On DeveloperWorks at
    - https://www6.software.ibm.com/dl/lxdk/lxdk-p
  - Also available from LINUX distributors
  - Level 1, 2 service by IGS contract
  - Same EOS as SDK for z/OS: 9/07 for SDK 1.3.1 and 9/09 for SDK 1.4

See a summary of tested platforms at http://www-128.ibm.com/developerworks/java/jdk/linux/tested.html

Understanding IBM JVM Major Building Blocks

IBM Java_{TM} 2 Objective - Value Add



- IBM Owned SDK Asset Base
- Improved Quality of IBM SDKs
  - Better development processes
  - More consistent functional implementation
  - Performance enhancements across platforms
  - IBM Technology Added Value (JVM and Classes)
  - IBM Just In Time (JIT) compiler
- Leverage new technology in both IBM hardware and software
- Continually Improving Tools for Application Development and Deployment
- Improved Performance, Scalability, RAS and improved System Exploitation

GOAL: Deliver Complete, Fully Compliant, Leadership SDKs

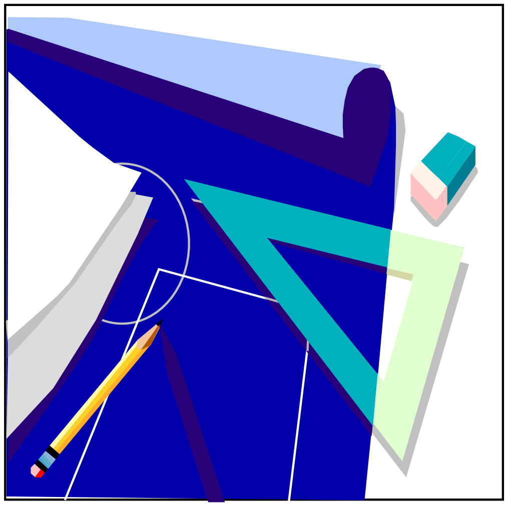
Java Strategy for System z

- Establish Java as a "de facto" programming environment on System z
  - Deliver J2EE capability in synch with Java industry standards (J2EE Certified, SUN's CTS)
- Lead with z/OS Qualities of Service
  - Market leader in delivering Java technology
  - z/OS platform for mission critical workload
- Enable all "Application Execution Environments" to support Java2 based applications:
  - WebSphere Application Server
  - Transaction Servers, ie. CICS & IMS
  - DB2 data base (Stored Procedures)
  - Enable connectivity to middleware
  - Messaging queuing
  - Java Batch processing (the inclusion of JZOS)

Continue to provide world class Development and Deployment Tooling

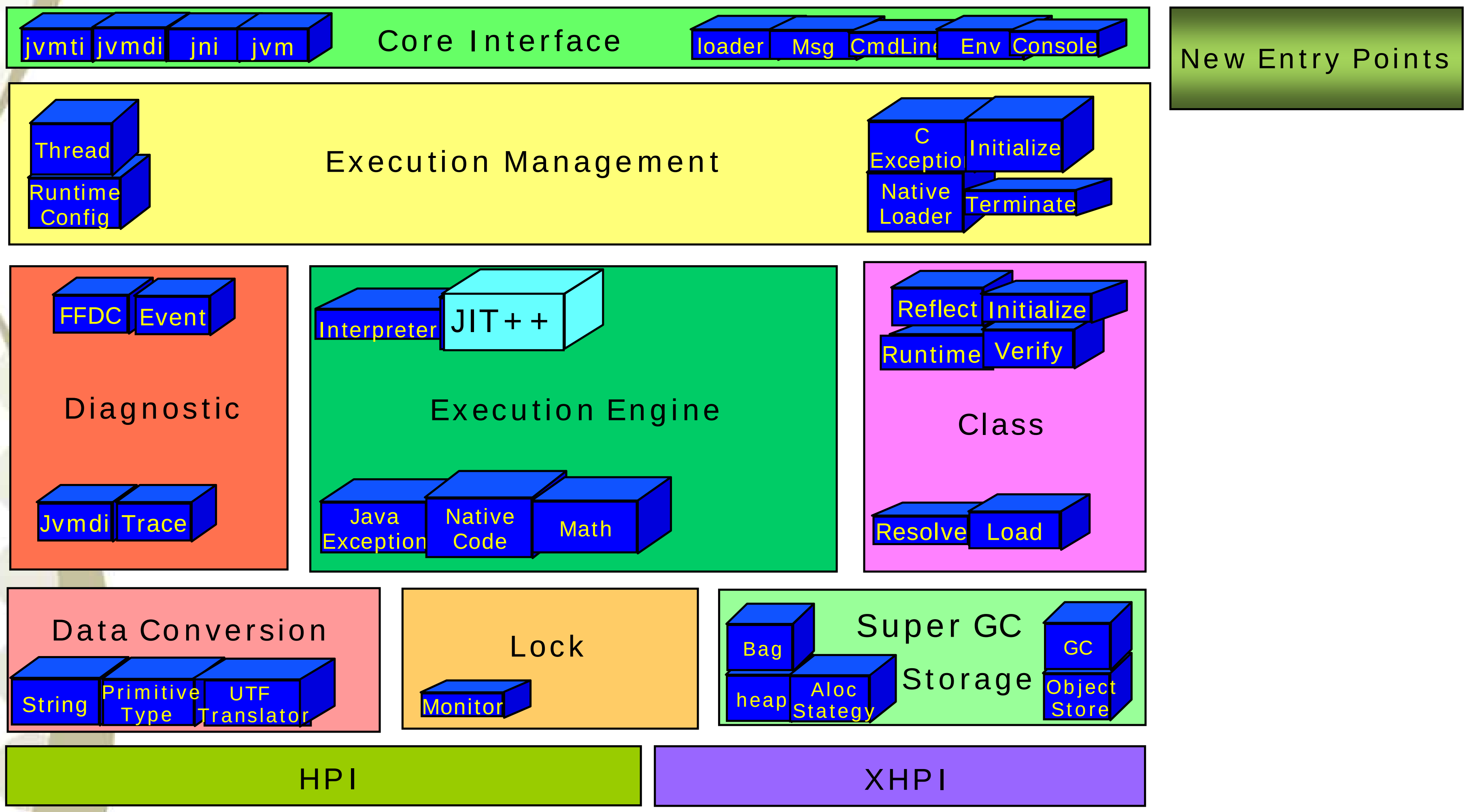
The Re-engineered IBM JVM

- The Re-engineered JVM common code base, plus Platform-specific code for
  - File handling
  - ASCII vs EBCDIC code page
  - JRIO (Java Record I/O)
  - Java Batch Support
  - RAS Characteristics
  - Profiling and Security APIs
  - RACF Integration
  - Hardware instruction set



- Integral Part of WebSphere Application Server platform
- Goal: Leading market in delivering Java technology

IBM JVM Additional Value-Add



Core Interface
jvmti jvmdi jni jvm
loader Msg CmdLin Env Console

New Entry Points

Execution Management
Thread
Runtime Config
C Exceptio
Initialize
Native Loader
Terminate

Diagnostic
FFDC Event
Jvmdi Trace

Execution Engine
Interpreter JIT++
Java Exception
Native Code
Math

Class
Reflect Initialize
Runtime Verify
Resolve Load

Data Conversion
String
Primitive Type
UTF Translator

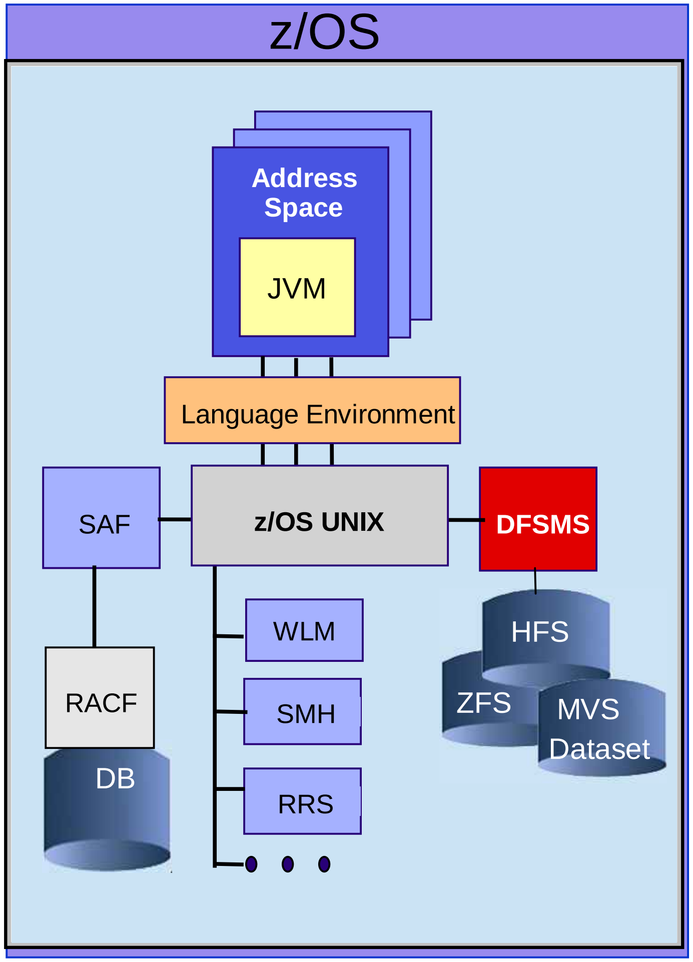
Lock
Monitor

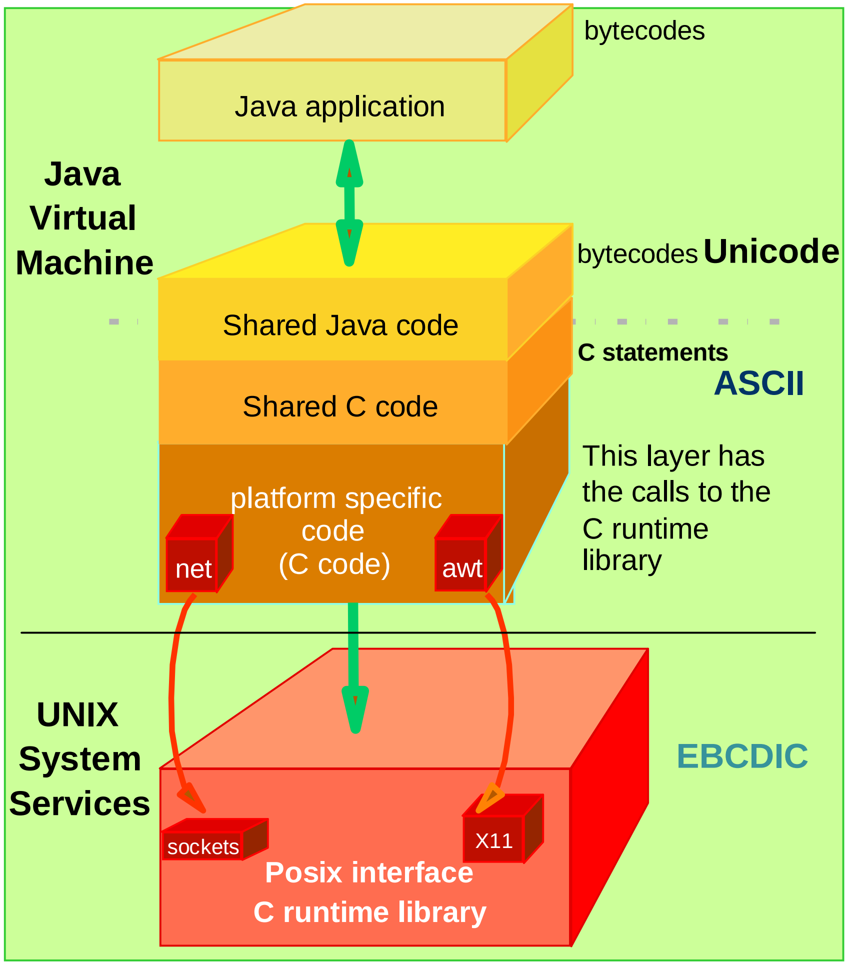
Super GC Storage
Bag
heap
Aloc Stategy
GC
Object Store

HPI

XHPI

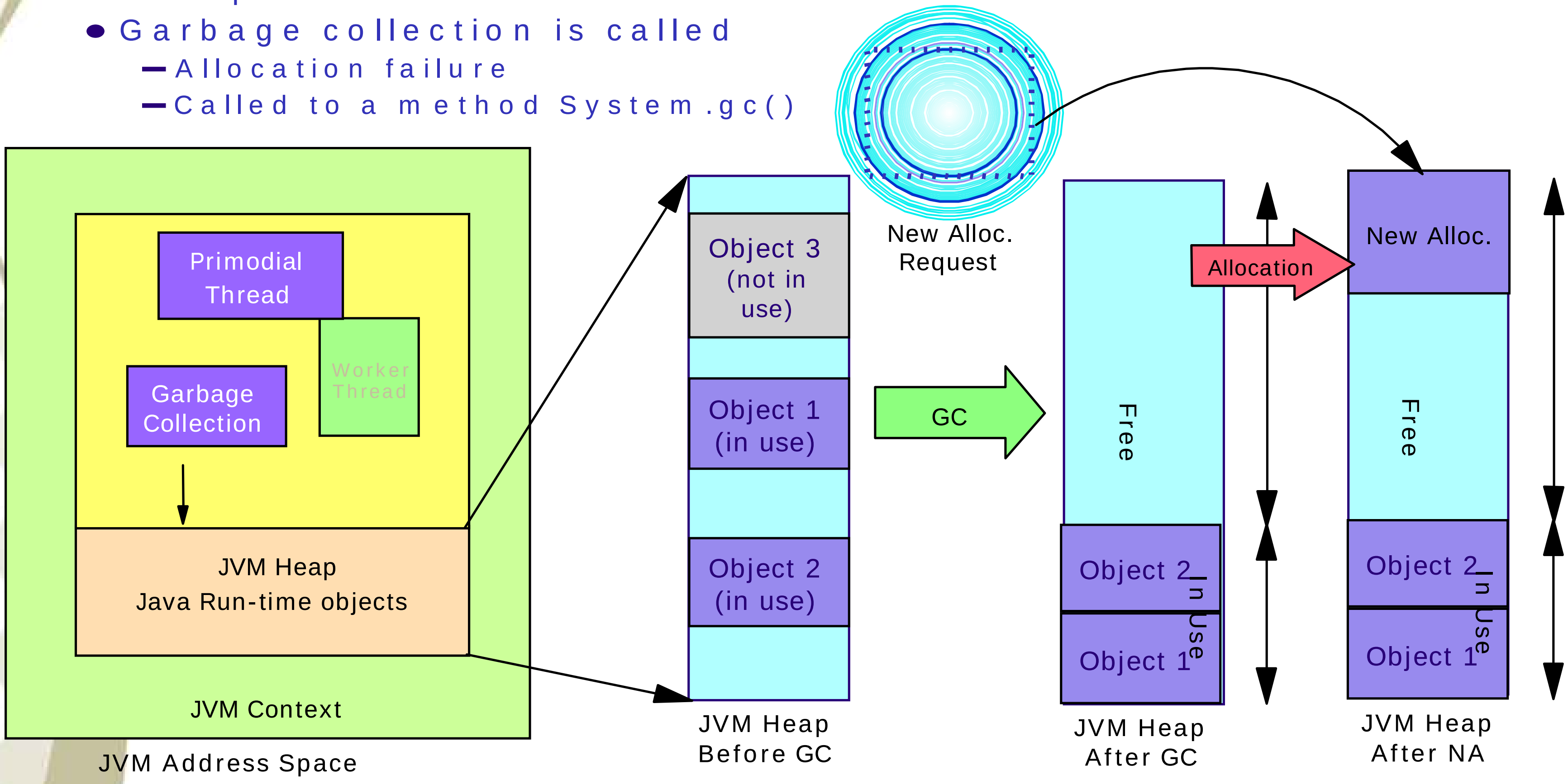
IBM JVM on z/OS





What is Garbage Collection (GC) ?

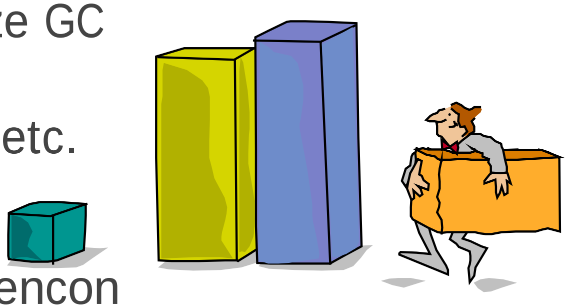
- Garbage collecton is within the JVM that automatically detects objects that are no longer being used and frees them to reclaim the space
- Garbage collection is called
  - Allocation failure
  - Called to a method System.gc()



Memory Management

- Java Heap & Garbage Collection
  - Do nothing, use default settings
    - JVM and GC will "settle in" at that 70% occupancy rate
    - Tune min/max heap size settings if you know your application

- Java heap size and GC frequency
  - Use small heap, frequent, fast GC
  - Use large heap, infrequent, slow GC
- Make sure you have enough physical memory to support the heap size
- GC performance
  - Use command line trace -verbose:gc to analyze GC performance
  - –Xtgc:dump|concurrent|compation|backtrace etc.
- GC Policy
  - Xgcpolicy:optthruput|optavgpause|subpool|gencon



Best Practices for Better Performance

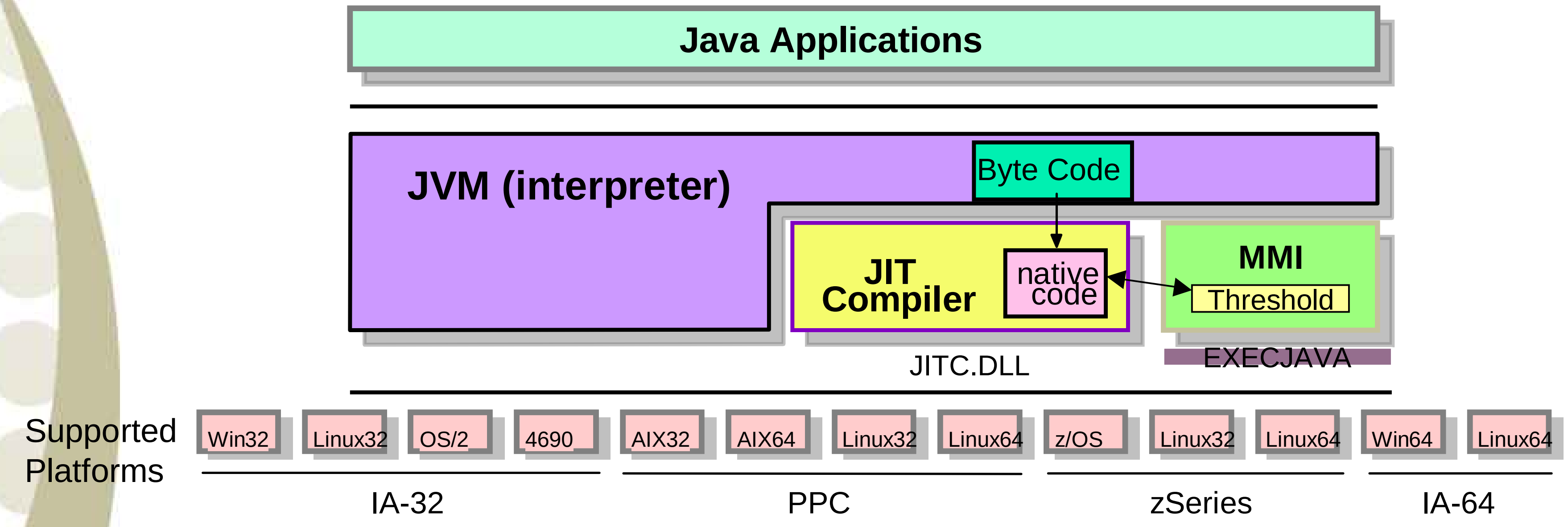
- Try to maintain 70% Heap Occupancy Rate
  - Free space 30%
- Try not to start out with a LARGE heap unless needed
- Try not to start out with min heap = max heap
  - Although some applications may benefit from it, after tuning
- Make sure the heap never pages
  - i.e. maximum heap size must fit in physical memory
    - Have enough real storage to keep paging low
- Factors that effect performance
  - The latest SDK release and z hardware
  - Just In Time Complier (JITC)
  - Java Heap size and settings
  - Garbage Collection

Best Practices for Better Performance (2)

- Avoid finalizers
  - Can never guarantee when a finalizer will run
  - If used, avoid allocating objects within the finalizer method
  - verbosegc trace will show if any finalizers are being called
- Avoid compaction (if possible)
  - verbosegc trace will show if compaction is occurring
  - Compaction usually caused by requests for large memory allocations
  - Analyze requests for large memory allocations, and avoid them if possible
    - e.g. split large arrays into pieces
- gcpolicy (default setting is –Xgcpolicy:optthruput)
  - Try using -Xgcpolicy:gencon if having issues with:
    - pause times and compaction
  - Application is transaction-based with short-lived objects
  - Especially when using large heaps

IBM JIT: Just-In-Time Compiler

- What does the JIT compiler do?
  - It dynamically generates machine code for frequently used bytecode sequences in Java applications while they are running
  - To improve performance by optimizing machine code execution
- MMI (Mixed Mode Interpreter) component
  - Designed to optimize the startup time and runtime performance of Java applications
  - Using a fast Assembler byecode interpreter (EXECJAVA)



Java Applications

JVM (interpreter)

Byte Code

JIT Compiler

native code

MMI

Threshold

JITC.DLL

EXECJAVA

Supported Platforms

IA-32: Win32, Linux32, OS/2, 4690

PPC: AIX32, AIX64, Linux32, Linux64

zSeries: z/OS, Linux32, Linux64

IA-64: Win64, Linux64

Java Record I/O

- JRIO is a class library, similar to java.io
- JRIO provides record-oriented access on z/OS
  - Virtual Sequential Access Method (VSAM) data sets (KSDS only)
  - Non-VSAM record-oriented data sets
  - The System Catalog
  - Partitioned data set (PDS) directory
  - DDName and GDGs support
    - GDG for PDS
  - SPACE and DISP parameter support
  - Navigational support for HFS directories
  - zAAP eligible
- JRIO Overview
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/jrio/overview.html
- How do I …. and sample programs
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/jrio/jrioread.html
- Java stand-alone applications on z/OS volume 1
  - http://www.redbooks.ibm.com/abstracts/sg247177.html

IBM Java Batch Launcher & Toolkit

- JZOS Java Batch Toolkit
  - For z/OS SDKs only - z/OS V1.6 and up
  - Integrated as part of z/OS SDK
    - SDK1.4.2 SR 6 and SDK5 SR 3 (09/06)
  - SDK 1.4.2 (31-bit only), SDK V5 and V6 (31-bit and 64-bit)
  - SMP/E installable
    - SMP/E processing will put start proc in SYS1.PROCLIB
    - load module in PDSE SYS1.SIEALNKE
    - sample JCL in SYS1.SAMPLIB
    - There is no post-install job
  - Non-SMP/E install
    - Will be similar to the current JZOS install documentation
    - Sample start proc, load module, and JCL will have to be moved appropriately
  - Supported by normal IBM support channels
  - Java batch toolkit and migration documentations are available on Java z/OS website (along with sample code)
    - http://www.ibm.com/servers/eserver/zseries/software/java/
- “Java Stand-alone Applications on z/OS, Volume 2”
  - http://www.redbooks.ibm.com/abstracts/sg247291.html
- To learn more about JZOS Batch Toolkit: Session 8368

zAAP: IBM Specialty Engine on z/OS

- zAAP (zSeries Application Assist Processor) for Java Workloads
- New processor type on z890, z990, z9 hardware supporting z/OS
- Order using Feature Code 6520
  - You can order up to one zAAP per configured or unassigned standard CPs on the processor
- A specialized z/OS Java execution environment for Java-based applications
  - WAS V5.1 or later
  - CICS/TS V2.3 or later
  - DB2 V7 and V8, IMS V8 or later
- No anticipated modifications to your Java applications
- Require z/OS V1R6 and SDK 1.4 or above with PTF UQ88783
- Usage Projection
  - z/OS V1.6 with RMF workload activity report to collect the “Would Have Been” zAAP usage by setting:
    - IEAOPTxx member of SYS1.PARMLIB:
      - *PROJECTCPU=YES*
- For more information on zAAP, please visit
  - http://www-03.ibm.com/systems/z/zaap/

Java Security on System z

- Allow Java application easy access to complex security capabilities within Java framework

JAAS	Java Authentication and Authorization
JCE	Java Cryptograph Extension using CCA (Common Cryptographic Architecture)
IBMJCECCA	IBM System z HW Cryptographic device
JSSE	Java Secure Socket Extension Support (SSL and TLS)
SAF Interface	z/OS Security Services in Java (JNI for SAF call)

- How much security do you really need?
  - Define / validate current security model
  - Use industry standards-based security APIs as much as possible
  - Don't spend a million to protect a dime
  - Don't make hasty decisions
  - Introduce change gradually

Development Tools

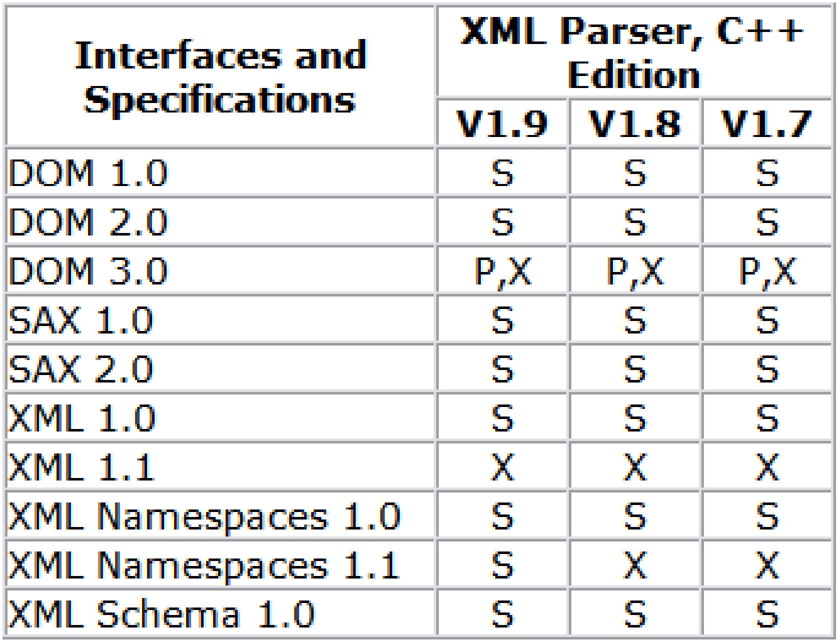
- IBM WebSphere Developer for System z (available 9/07)
  - IBM Rational® Developer for System z™ V7.1 includes capabilities that can help make traditional mainframe development, Web development, and integrated service-oriented architecture (SOA)-based composite development fast and efficient. COBOL, PL/I, C, C++, High-Level Assembler, and Java™ developer communities can also be more productive when they take advantage of these functions. IBM Rational Developer for System z integrates with and extends the IBM Rational Software Delivery Platform (SDP).
  - WebSphere Developer for System z V7.x is the follow-on replacement for WebSphere Studio Enterprise Developer V5 and WebSphere Developer for zSeries V6.
  - http://gwareview.software.ibm.com/software/awdtools/devzseries/
- IBM WebSphere Integration Developer
  - http://www.ibm.com/developerworks/websphere/zones/businessintegration/roadmaps/wid/roadmap-wid.html
- Debug Tool for z/OS
  - http://www-306.ibm.com/software/awdtools/debugtool/

Application Performance Tooling

- IBM Application Analyzer for z/OS
  - http://www-306.ibm.com/software/awdtools/apa/
- IBM Performance Optimization toolkit for Rational performance tester
  - Monitor applications running on the application server during performance test under load
  - Provide analysis of the performance trace data in Rational Performance Tester
  - http://www-306.ibm.com/software/rational/toolkit/ipo_toolkit.html
- Rational performance tester
  - A multi-user testing tool for validating web application scalability before deployment
  - http://www-306.ibm.com/software/awdtools/tester/performance/index.html
- Test, profile and monitor applications
  - http://www.ibm.com/developerworks/edu/os-dw-os-ecl-tptp.html
- Tools for Eclipse development customers
  - http://www-304.ibm.com/jct03002c/software/rational/offerings/design/matrix/
- Application Performance Analyzer Tool for z/OS
  - http://www-306.ibm.com/software/awdtools/apa/

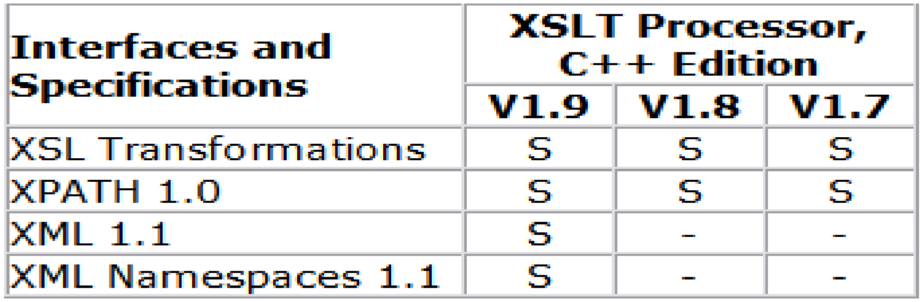
z/OS XML Toolkit Components

Interfaces and Specifications for XML Parser, C++ Edition



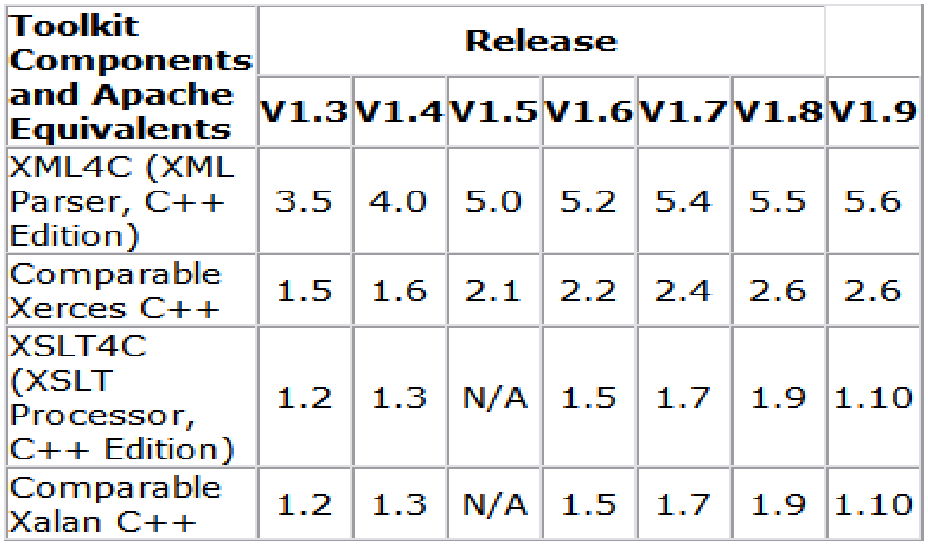
Interfaces and Specifications	XML Parser, C++ Edition		
	V1.9	**V1.8**	**V1.7**
DOM 1.0	S	S	S
DOM 2.0	S	S	S
DOM 3.0	P,X	P,X	P,X
SAX 1.0	S	S	S
SAX 2.0	S	S	S
XML 1.0	S	S	S
XML 1.1	X	X	X
XML Namespaces 1.0	S	S	S
XML Namespaces 1.1	S	X	X
XML Schema 1.0	S	S	S

Interfaces and Specifications for XSLT Processor, C++ Edition



Interfaces and Specifications	XSLT Processor, C++ Edition		
	V1.9	**V1.8**	**V1.7**
XSL Transformations	S	S	S
XPATH 1.0	S	S	S
XML 1.1	S	-	-
XML Namespaces 1.1	S	-	-

XML Toolkit Component Version Summary



Toolkit Components and Apache Equivalents	Release						
	V1.3	**V1.4**	**V1.5**	**V1.6**	**V1.7**	**V1.8**	**V1.9**
XML4C (XML Parser, C++ Edition)	3.5	4.0	5.0	5.2	5.4	5.5	5.6
Comparable Xerces C++	1.5	1.6	2.1	2.2	2.4	2.6	2.6
XSLT4C (XSLT Processor, C++ Edition)	1.2	1.3	N/A	1.5	1.7	1.9	1.10
Comparable Xalan C++	1.2	1.3	N/A	1.5	1.7	1.9	1.10

z/OS XML Toolkit

- SMP/E install or require the XML Toolkit for z/OS V1.7, V1.8, or V1.9 as a prerequisite or corequisite to another product :
  - Go to the XML toolkit for z/OS download site & the V1.9 package files
    - Toolkit.pax.Z
    - XMLSMPE.README.txt
  - The V1.9 download package contains V1.7, V1.8, and V1.9 installations
    - Refer to the V1.9 Program Directory for installation information
    - The default installation will install all releases that are not already present on your system

- XML toolkit for z/OS download (SMP/E and Non-SMP/E install)
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/download/
- XML toolkit education links
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/education/
- XML Usage Information
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/usage/
- XML Performance
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/perform/
- XML Tools
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/tools/
- XML FAQs
  - http://www-03.ibm.com/servers/eserver/zseries/software/xml/FAQS/

Looking Ahead: What's New & Exciting

- SDK V6
  - Additional functional and performance enhancements
  - Keep an eye out for z/OS specific security features and JZOS enhancements
    - SDK V6 SR 1
- PHP (Hypertext Preprocessor) version 5.1.2 for z/OS
  - A general-purpose scripting language that is well-suited for Web development
    - PHP's syntax is similar to that of C and Pearl
  - PHP for z/OS includes an extension to access DB2 for z/OS via ODBC
    that allows PHP applications to access DB2 data on z/OS
  - This is an unpriced feature for IBM ported tools for z/OS
- Web 2.0 Starter Toolkit for IBM DB2 – Web server and PHP modules
  - Code enabled for mash-ups, Ajax applications and Web feeds
- IBM continue to provide timely Service Refresh
  - For SDK 1.4.x, SDK V5 and SDK V6

Summary and Q & A

- Continue to provide System z SDK technology base for
  - WebSphere, CICS, IMS, MQ and DB2
  - Linux Middleware
  - ISVs
- Continued rollout of Java2 including new IBM architectures to allow better platform integration, function, tailoring and performance
- Recommendation: Stay Current by visiting our web site Frequently http://www.ibm.com/servers/eserver/zseries/software/java
- Reporting a problem http://www-1.ibm.com/servers/eserver/zseries/software/java/services.html
- Q & A

SDK V5 Highlight

Supplemental Materials

IBM SDK V5 on System z

- IBM "Value Add" changes
  - Large number of Java Runtime Changes
    - New GC implementation
    - New Just In Time (JIT) Compiler
    - Shared Classes on all Platforms
    - New RAS Functionalities and Tools
  - Number of RAS and Debugging Changes
  - New functionalities and JVMTI (JVM Tool Interface)
- IBM 31-bit and 64-bit SDK for z/OS, Java 2 Technology Edition, Version 5
  - Provides a full-function SDK at Java 2 technology level with Sun SDK 5 APIs
  - Available from the IBM eServer zSeries Java web site and on tape from IBM Software Delivery and Fulfillment (SDF)
- System requirements
  - z/OS V1.6 or z/OS.e V1.6 or later is required
  - z800, z890, z900, z990 and z9
- Compatibility
  - 31bit SDK V5 is compatible with SDK V1.4.x with the exception of Persistent Reusable JVM

SDK V5 GC Implementation

- Uses a “Type Accurate” Collector
  - Does not suffer from “retained garbage”
  - Does not suffer from pinned/dosed objects
- Introduces a new Generational GC Policy
  - -Xgcpolicy:gencon
    - “gencon” – controls the behavior of the Garbage Collector
    - Minimized the amount of pause time in GC cycle
      - *The combined used of concurrent and generational GC*
  - The default setting is –Xgcpolicy:optthruput
- Two generational collector
  - The new and old areas

V5 JIT Implementation & Shared Classes

- New JIT Implementation
  - Uses a separate compilation thread
    - Methods queued for compiling
  - Can recompile methods
    - 5 optimization levels
    - The busiest methods are always optimized most aggressively
  - Still carries out compilation under –Xdebug
    - Termed "full speed debug"

- Shared Classes
  - Available on all server platforms
  - Static class data caches in shared memory
    - Shared between all IBM JVMs
    - All application and bootstrap classes shared
    - Cache persisted beyond lifetime of any JVM, but lost on shutdown/reboot
  - Provides saving to memory footprint and start up time

JVMTI (JVM Tool Interface)

- JVMTI is a two-way communication interface between the JVM and a native interface
  - Replace the JVMPI (profiler interface) & JVMDI (diagnostic interface)
  - JVMTI allows 3rd parties to develop debugging, profiling and monitoring tools for the JVM
    - More than one agents can be attached to a JVM at any one time
  - JPDA tools – Java Platform Debugging Architecture
  - DTFJ – Diagnostic Tooling Framework for Java
    - A Java application API from IBM to support the building of Java diagnostics tools
- IBM does provide a simple profiling agent based on HPROF interface
  - JVMTI agents can be loaded at startup using short or long forms of the command-line option:
    - -agentlib:hprof=<options>
      - assumes that a folder containing hprof.dll is on the library path, or -agentpath:C:\sdk\jre\bin\hprof.dll=<options>
- HPROF on SDN (Sun Developer Network)
  - http://java.sun.com/developer/technicalArticles/Programming/HPROF.html

Download & Migration Considerations

- Download SDK V5
  - z/OS downloads available free from:
    - http://www.ibm.com/java
  - zLinux download from the developerWorks website:
    - http://www.ibm.com/developerworks/java/jdk/linux/download.html
- Incompatibilities
  - http://java.sun.com/j2se/1.5.0/compatibility.html
- Deprecated APIs
  - http://java.sun.com/j2se/1.5.0/docs/api/deprecated-list.html
- Prerequisites
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/j5prereq31.html
- Restrictions and Other Considerations
  - http://www-03.ibm.com/servers/eserver/zseries/software/java/j5restrict31.html
- SDK V5 APIs
  - http://java.sun.com/j2se/1.5.0/docs/apip

SDK 6 Supplemental Materials

What's new in the IBM SDK V6

- NEW EVTK RAS Tooling:
  - The Extensible Verbose Toolkit (EVTK) is a data visualizer for analyzing heap memory usage and garbage collection in the JVM
  - EVTK plots garbage collection log and trace output, and can be extended to plot other forms of input, such as SPECjbb and Trade 6 benchmark data
  - EVTK can also save data to jpeg or csv formats for further reporting and analysis
  - To use the EVTK, you will need to install IBM Support Assistant:
    - A free download -- http://www-306.ibm.com/software/support/isa/
    - The above site provides details on how to download and install IBM Support Assistant and acquire the EVTK plug-in
- XSLT Processing:
  - The new XL TXE-J XSLT compiler has been designed for performance and is now the default XSLT processor in Java SDK V6
  - The XL TXE-J compiler replaces the XSLT4J compiler
    - The XSLT4J interpreter is still available

What's new in the IBM SDK V6? (cont'd)

- Class sharing between Java Virtual Machines (JVMs):
  - SDK V6 allows compiled code for methods active during JVM startup to be cached to improve the startup time
  - Additional class sharing improvements help reduce the overall memory footprint
- Enhanced diagnostics information:
  - The SDK V6 Java Diagnostics and User Guides are improved to provide accuracy and ease of use
  - A new documentation Launchpad will direct you to diagnostics and API documentation with ease
  - Guides are also being made available on-line where they can be searched by your favorite search engines

SDK V6 Generic Features

- XML processing and Web services:
  - XML specifications have been incorporated into this Java release
    - Streaming API for XML (StAX)
    - Java API for XML Binding (JAXB) 2.0
    - Java API for XML-based Web Services (JAX-WS) 2.0
  - SDK V6 built-in support for enhanced Web services metadata and APIs for processing XML digital signatures
- Annotations-based programming:
  - In SDK V6, the annotations model has been expanded to include new built-in annotation types and annotation processing APIs
    - Annotations - a mechanism for embedding metadata into Java source code, were introduced in Java 5.0
- Application client APIs:
  - Several new APIs have been added to the SDK to support application client operations
    - GIF image writer - the ability to access helper applications registered in the native desktop
    - Native support for fast flash screen display, and support for system tray icons
    - Swing components have improved drag-and-drop capability and support for multi-threaded programming

SDK V6 Generic Features (cont'd)

- JDBC 4.0:
  - Some of the highlights of this updated specification include updated SQL and XML support
    - Automatic driver loading, improved connection management, close association with JDBC RowSet implementations
    - Built-in annotations to make it easier for applications to manipulate data
- Java compiler APIs:
  - Java 6 includes a set of compiler APIs that allow a Java program to call a Java compiler and retrieve and examine the compiler's output in a structured fashion
- Internationalization:
  - Changes to client processing allow for improved internationalization support
    - such as pluggable locales that allow existing Java Runtime Environments to be extended to support custom locales
    - APIs that transform Unicode strings into different canonical forms, and improvements to the ResourceBundle class
    - This release of the SDK will also support internationalized Internet domain names and URIs
- Other features:
  - Masked command-line password entry, a framework for connecting to external scripting engines, and bidirectional variants of several Collections classes

Thank You