



the
POWER
of
JAVA™



JavaOne
Part of the Network and Business Solutions

Migrating Your Applications to the Java™ Platform, Release 1.5

Arjun Jayaram

Director of Engineering
Become.com
www.become.com

TS-1356

Goal of Your Talk

What Your Audience Will Gain

Benefits of migrating existing applications to the Java™ 1.5 platform

Migration issues to watch out for

How to take advantage of the superior VM performance

Agenda

Benefiting from VM improvements

Security enhancements

Java class library enhancements

XML Support

JMX™, JDBC™ APIs enhancements

Deployment improvements

Recommendations for migration

Agenda

Benefiting from VM improvements

Security enhancements

Java class library enhancements

XML Support

JMX, JDBC APIs enhancements

Deployment improvements

Recommendations for migration

VM Improvements in 1.5

Increased Speed in Invoking Smaller Applications

- Installer loads few class files from system jar file and creates a file on the OS
 - Subsequent invocations memory maps this file thereby reducing start-up time
- If JRE has not been installed using the Java platform installer, the system jar may need to be regenerated (`java -Xshare:dump`)
 - Limitation: No support in Windows 95/98/ME
- Consider if you have several small applications invoked repeatedly

VM Improvements in 1.5

Enhanced Garbage Collection

- GC scheme has changed from serial collector to parallel collector for server class VM invocations
- The default heap sizes have changed
 - Initial: Larger of 1/64th of the machine's physical memory on the machine or some reasonable minimum
 - Maximum: Smaller of ¼ of the physical memory or 1 GB
- On Windows, the default VM is client size. So change the start-up scripts to use the **-server** option for server class applications running on windows
- Reconsider the Heap size options in start-up (you may not need them or may need to reconfigure)
- Retest applications for out-of-memory error. May need to use the **-XX:GCTimeRatio** or **-XX:GCTimeLimit** options for the new GC strategy

VM Improvements in 1.5

Thread Priority Implementation on Solaris™ OS

- Java technology Threads mapped to the corresponding native threads (native T1 and T2 modes)
- This had a side effect for threads with normal priority; they ran slower than applications in 'C' with normal priority
- In 1.5, priorities in the range of 10–15 are mapped to highest possible native priority and 1–4 are mapped to lower priority
- If your multi-threaded applications are running on the Solaris OS and you change the priority, review the code and simplify if possible

Agenda

Benefiting from VM improvements

Security enhancements

Java class library enhancements

XML Support

JMX, JDBC APIs enhancements

Deployment improvements

Recommendations for migration

Security Enhancements in 1.5

Crypto and Transport Security Enhancements

- Major changes to the security stack from 1.4
- The Socket APIs have been redesigned to be non-blocking and transport independent
- If you have used `javax.security.cert` in your applications, you may benefit from the `java.security.cert` API
- The Sun Java Secure Socket Extension (JSSE) implementation now uses the Java Cryptography Extension (JCE) for all its cryptographic algorithms; This can be changed from the `java.security` properties file
- Prior to 1.5, JSSE supported only stream-based sockets; The new `javax.net.ssl.SSLEngine` abstraction works with any transport defined by the application

<http://java.sun.com/j2se/1.5.0/docs/guide/security/jsse/JSSERefGuide.html>

Security Enhancements in 1.5

Kerberos Enhancements

The Sun JSSE provider has support for Kerberos cipher suites. The following cipher suites are supported (but not enabled by default)

```
TLS_KRB5_WITH_RC4_128_SHA          TLS_KRB5_WITH_RC4_128_MD5
TLS_KRB5_WITH_3DES_EDE_CBC_SHA    TLS_KRB5_WITH_3DES_EDE_CBC_MD5
TLS_KRB5_WITH_DES_CBC_SHA        TLS_KRB5_WITH_DES_CBC_MD5
TLS_KRB5_EXPORT_WITH_RC4_40_SHA   TLS_KRB5_EXPORT_WITH_RC4_40_MD5
TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5
```

```
{
// Create socket
SSLSocketFactory sslsf = (SSLSocketFactory) SSLSocketFactory.getDefault();
SSLSocket sslSocket = (SSLSocket) sslsf.createSocket(tlsServer, serverPort);

// Enable only one cipher suite
String enabledSuites[] = { "TLS_KRB5_WITH_DES_CBC_SHA" };
sslSocket.setEnabledCipherSuites(enabledSuites);
}
```

If you have used the Java Generic Security Service (Java GSS) interface, you have more options with these enhancements.

Security Enhancements in 1.5

Smart Card and KeyStorage support

- Support for Hardware cryptographic accelerators; Use the Sun PKCS#11 provider
- The `java.security.KeyStoreBuilder` class abstracts the construction (including smartcard) and initialization of a `KeyStore` object
- It supports the use of `CallbackHandlers` for password prompting and can be subclassed to support additional features as desired by an application
- Support for multiple certificates with the new `X509KeyManager` implementation; If multiple certificates are available, it also makes the effort to pick a certificate with the appropriate key usage and prefers valid to expired certificates

Agenda

Benefiting From VM Improvements

Security Enhancements

Java Class Library Enhancements

XML Support

JMX, JDBC APIs Enhancements

Deployment Improvements

Recommendations for Migration

Java Class Library Enhancements

New Classes in `java.lang.*` and `java.util.*`

- `ProcessBuilder` class provides great support for launching another process; Can set the command string, environment, working directory, redirect error stream etc (If you have used `Runtime.exe` and passed the command line parameters in the same string, this may be a better option)
- A good match for the `printf()` `scanf()` functions in 'C'; Consider using the `Formatter` and `Scanner` classes
- Concurrency Controls—Finally!!! (following new packages JSR 166)
 - `java.util.concurrent`, `java.util.concurrent.atomic`, `java.util.concurrent.locks`
 - Provide scalable and maintainable code for thread safety and synchronization
- New useful classes in the collection framework
 - `Queue`—`PriorityQueue`, `ConcurrentLinkedQueue`, `LinkedList`, `AbstractQueue`
 - `BlockingQueue`—`Linked`, `Array`, `Priority`, `Delay` and `Synchronous Blocking`
 - `Collections.checkedInterface` ensure that your data collectors are thread safe
- New Utility classes are compelling and are likely to reduce code complexity in existing multithreaded applications

Java Class Library Enhancements

Internationalization

- Character handling is now based on version 4.0 of the Unicode standard with support for supplementary characters
- Backward compatibility for the String class; So you will be able to deserilize objects from previous VMs
- Support for Vietnamese as a locale
- Remember to retest code that does text analysis for eg:

From:

```
for (int i = 0; i < string.length(); i++) handle(string.charAt(i));
```

To:

```
int c; for (int i = 0; i < string.length(); i += string.charCount(c))  
    { c = string.codePointAt(i);  
      handle(c); }
```

Java Class Library Enhancements

Java Language Features

- Consider these tips if you are developing new code in Java technology
- Generics (Finally!!)—Provides a way for you to communicate the type of a collection to the compiler, so that it can be checked
- Use the Auto boxing and un-boxing while working with collection classes
- Use type safe enums instead of static final in your new code; Allows for type checking in collection classes
- If you have documented coding standards in your organization, you may wish to add static imports

(Allows unqualified access to static members without inheriting from the type containing the static members)

- `import static java.lang.Math.*` while used in multiple places

Agenda

Benefiting From VM Improvements

Security Enhancements

Java Class Library Enhancements

XML Support

JMX, JDBC APIs Enhancements

Deployment Improvements

Recommendations for Migration

JAXP Enhancements

New in 1.5

- Java API for XML Processing (JAXP) version 1.3 with reference implementation 1.3 which includes the following
 - Xerces version 2.6.2
 - XSLTC version 2.6.0
- Built in validation with an XML Schema with option to create custom validator
- XSLTC support will speed up XSLT transforms; Applications can consider the new JAXP for this feature
- NameSpaces 1.1 support enables handling unicode characters in tag names
- Support for XInclude (Recommendation: Do not use this until all applications have been upgraded to 1.5)

JAXP Enhancements

Limitations and Migration Recommendations

- Size limitation for the XML files—2GB should be able to handle most validation requests
 - Built in validation with an XML Schema with option to create custom validator
- Some packages have changed from JAXP 1.1
- If your application accepts external XML files, you may benefit from the fix to a potential DOS by using the `-DentityExpansionLimit` option

Agenda

Benefiting From VM Improvements

Security Enhancements

Java Class Library Enhancements

XML Support

JMX, JDBC APIs Enhancements

Deployment Improvements

Recommendations for Migration

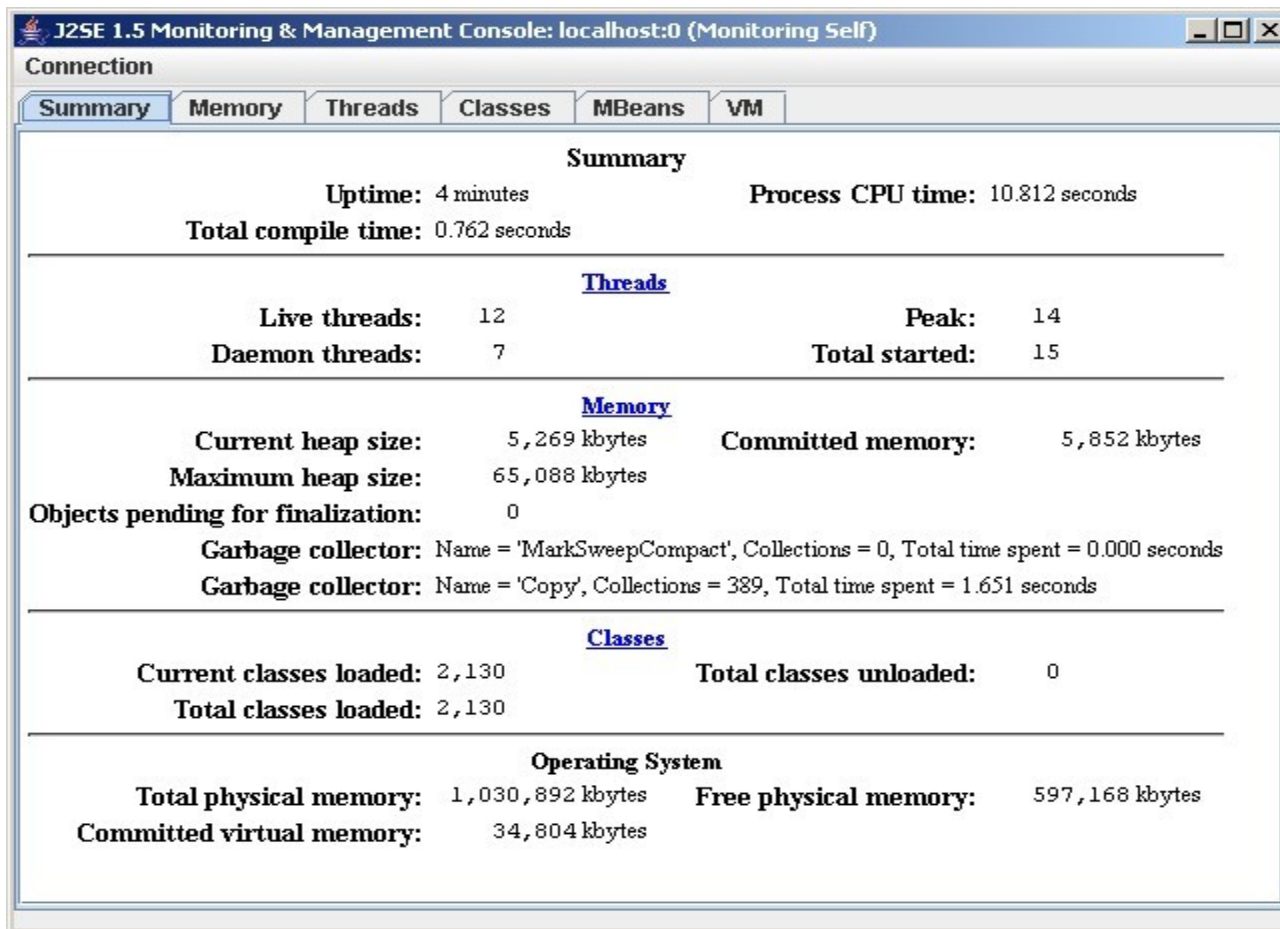
Monitoring Enhancements

New in 1.5

- The VM has been instrumented for providing monitoring and management functions with support for Java Management Extensions (JMX) and SNMP
- Developers may turn this feature on during the testing phase of applications to detect deadlocks or low memory conditions
- In built platform MBean server provides access to several system values at run time
- New jconsole provides useful information from the platform mbean server
- Enhanced logging management provides the following useful features
 - Get the name of the log level associated with the specified logger
 - Get the list of currently registered loggers
 - Sets the specified logger to the specified new level at run time

Monitoring Enhancements

JConsole



J2SE 1.5 Monitoring & Management Console: localhost:0 (Monitoring Self)

Connection

Summary | Memory | Threads | Classes | MBeans | VM

Summary

Uptime: 4 minutes Process CPU time: 10.812 seconds
 Total compile time: 0.762 seconds

Threads

Live threads: 12 Peak: 14
 Daemon threads: 7 Total started: 15

Memory

Current heap size: 5,269 kbytes Committed memory: 5,852 kbytes
 Maximum heap size: 65,088 kbytes
 Objects pending for finalization: 0
 Garbage collector: Name = 'MarkSweepCompact', Collections = 0, Total time spent = 0.000 seconds
 Garbage collector: Name = 'Copy', Collections = 389, Total time spent = 1.651 seconds

Classes

Current classes loaded: 2,130 Total classes unloaded: 0
 Total classes loaded: 2,130

Operating System

Total physical memory: 1,030,892 kbytes Free physical memory: 597,168 kbytes
 Committed virtual memory: 34,804 kbytes

JDBC API Enhancements

Different Implementations of RowSet

- JSR 114 recommendation implemented with 5 new implementations of RowSet
 - JdbcRowSet—Standard
 - CachedRowSet—Disconnects from DS
 - FilteredRowSet—Extends CachedRowSet used to get a subset of data
 - JoinRowSet—Extends CachedRowSet used to get an SQL JOIN of data from multiple RowSet objects
 - WebRowSet—Extends CachedRowSet used for XML data
- Recommendation: Check your vendor supplied JDBC API drivers for the proper implementation

Agenda

Benefiting From VM Improvements

Security Enhancements

Java Class Library Enhancements

XML Support

JMX, JDBC APIs Enhancements

Deployment Improvements

Recommendations for Migration

Deployment Enhancements

Jar Files and Java Web Start Software

- Pack200 algorithm used which reduces the size of the jar files for download
- Support for timestamped and signed jar files which solves problems of old code with valid certificates and new code without a corresponding certificate
- Import facility in Java Web Start software ensures that prerequisites for your application can be installed prior to your application installation
- Single Instance invocation is supported; So multiple install requests can be handled gracefully
- An Enterprise Configuration file is accessible as a URL so that all applications can use the same configuration properties instead of working on a copy

Agenda

Benefiting from VM Improvements

Security Enhancements

Java Class Library Enhancements

XML Support

JMX, JDBC APIs Enhancements

Deployment Improvements

Recommendations for Migration

Recommendations for Migration

- Evaluate the serialized objects from previous versions. XML-based serialization is easier than serializing object instances
- Start with the low hanging fruits—Improvements in VM, GC, etc.; Start with your start-up scripts
- If you are using Java technology security or third party security, the enhancements in 1.5 are likely to impact you
- Evaluate your entire stack of applications, migrate each application separately
- If you use third party tools or libraries, make sure that they are certified on 1.5

Recommendations for Migration

- Look at changes in Unicode 4.0 while handling code related to I18N or text analysis
- Use this opportunity to cleanup Thread synchronization code
- Java Web Start software is more robust and may be a viable option now
- Re-evaluate your Java language coding standards doc—Enhancements to the `java.lang` and `java.util` packages are likely to change coding standards in enterprise
- If you do intense text analysis, migration should look at the unicode changes; Also update your unit tests

Summary

- Significant enhancements to performance and security
- Useful improvements in lang and utils package
- Support for Unicode 4.0
- Compelling reasons to start migration efforts
- Use caution while migrating objects or schema

Q&A



the
POWER
of
JAVA™



JavaOne
Part of the Network and Business Solutions

Migrating Your Applications to the Java™ Platform, Release 1.5

Arjun Jayaram

Director of Engineering
Become.com
www.become.com

TS-1356