Java Overview and Java SE 6 What's New

Max Li
Campus Ambassador in HKUST

Gigi Li
Campus Ambassador in CUHK

Sun Microsystems, Inc.
What's the meaning of Sun Microsystems?

- Andy Bechtolsheim
- Vinod Khosla
- Scott McNealy
- Bill Joy
What is Sun Microsystems doing?
Product of Sun

- **Computer servers and workstations**
  (based on Sun's SPARC, AMD's Opteron, Intel's Xeon)

- **Storage systems**
  (Sun StorageTek 5800 System, Sun Fire X4500 storage server, SAM-QFS filesystem)

- **Operating System**
  (Solaris OS)

- **Developer tools**
  (Netbeans, Sun Studio)

- **Platform& Standards**
  (Java, Java SE, Java EE, Java ME)

- **Database Management Systems**
  (JavaDB, PostgreSQL for Solaris, MySQL)

- **Others**
  (Sunspot, Systems Management, Mozilla Suite, Identity Manager etc.)
Campus Ambassador Program
Towards Education
500 Sun Campus Ambassadors

- 75 in U.S.
- 40 in Brazil
- 40 in Russia
- 100 in India
- 100 in China
- 15 in Canada
- 20 in Latin America
- 7 in Australia & New Zealand
- 12 in Asia South
- 3 in Japan and Korea
- 15 in Central/North Europe
- 5 in France
- 15 in Germany
- 5 in Portugal and Spain
- 5 in Italy
- 20 in South/Eastern EMEA
- 8 in the UK
Sun Academic Initiative Program

Take the online course of Sun technologies for FREE!

based on 2 websites:
http://learningconnection.sun.com
http://sunsite.scut.edu.cn

Nearly 60% discount for the Sun Certificate !!!

SCJP: Sun Certified Java Programmer
SCSA: Sun Certified System Administration for the Solaris Operating System
Sun Certified Developer for Sun Application Server
Sun Certified Engineer for Sun Directory Server
share the most to the world!
Opening Day: June 14, 2005

BUILDING A COMMUNITY
I NEED YOU for MUSTANG DEVELOPMENT

http://mustang.dev.java.net
Community

Sun Developer Community of China
http://developers.sun.com.cn

Open Technology Community
http://www.opentech.org.cn

NetBeans Community
http://www.netbeans.org

OpenSolaris Community
http://www.opensolaris.org

Java Community
http://www.java.net

OpenJDK Community
http://community.java.net/OpenJDK
Java and Netbeans Overview
What is Java?
Java is a Brand!
Compatibility is Guaranteed!
Java is a Programming Language!
Java is a Platform!
Where is Java?
Java is Everywhere

NASA's Mars Exploration Rovers

- Appliances/Toys: 11,000M
- Handhelds: 2,600M
- Entertainment: 1,300M
- Computers: 1,080M
- Industrial/Automotive: 400M
Java is FREE Software
Java is being used by everyone!

Printers
Game Consoles
Robots
TVs
PDAs
Webcams
Samsung
Philips
Epson
Panasonic
Google
GM
Ebay
Boeing
Ricoh
Canon
Agilent Technologies
LG
BMW
NASA
Jet Propulsion Laboratory
California Institute of Technology
Lottery Terminals
STBs
Telescopes
Medical Equipment
Consumer Electronics
Twelve Years and Still Accelerating

6M Developers
5.5B Devices
2.5M GlassFish
800M Desktops
1.8B Phones
11M Televisions

Participation
Java Dominates in Financial Services

“Do you strategically use J2EE and/or .NET?”

- J2EE: 40%
- .NET: 7%
- Both: 40%
- No strategic decision: 7%
- Don’t know: 6%

Base: 55 European enterprise architects at financial services firms

Source: Forrester’s Architect Survey on European Financial Industry in Jan 2008
Java Community
Java.net: The Source for Java Technology Collaboration

• What it is:
  > Web-based community for Java developers
  > Open, collaborative Java development
  > Communities with common interests (java gaming)

• Active and vibrant
  > 175,000+ members
  > 2,200+ projects
  > 19 Java communities
  > 74 hosted JUGs
  > 100 RSS feeds
  > Blogs, Wikis, Javapedia
Java.sun.com: Learning Community

Online Resources for Professional Development and Certification:

- Tutorials and Code Camps
- Online Sessions and Courses
- Java Technology Professional Certification
- Instructor-Led Courses
- Quizzes
JavaOne

- Largest developer conference in the world
- Over 14,000 attendees
- 174+ Technical Sessions, 145 BOFs, 37 Hands on Labs
- 23,000+ JavaOne Online subscribers worldwide
- Free multimedia content for SDN subscribers

Mark Your Calendar!
San Francisco: May 6-9, 2008

dev footnote: developers.sun.com/learning/javaoneonline/
NetBeans
What Is NetBeans?

There are two answers...

#1: It is an award-winning IDE

- For Java developers (but we're adding more languages...)
- Free, open-source - http://www.netbeans.org
- Large (and growing) community of users that can help.
#2: It is a Platform

- For building rich client applications that will run on *any* operating system that supports a standard JVM.
- Provides a rich framework of windows, menus, tool bars, actions, etc.
- Used for building a wide variety of applications
What Is NetBeans?

Seeing is believing

• Quick Examples
  > Minex
What makes NetBeans the best?

1. Ant Based Project System
2. Friendly Editor
3. Powerful J2EE Support
4. Visual J2ME Development
5. An integrated Profiler
6. Collaboration tools
What makes NetBeans the best? (cont.)

7. Spectacular GUI Builder (Project Matisse)
8. Refactoring support
9. CVS support
10. Java BluePrints Solution Catalog
11. Module Development Support
12. Update Center
What's New In NetBeans 6.0?

• Code editor enhancements
• Ruby, JRuby, & Ruby on Rails support
• Easier installation and upgrading
• Swing GUI data binding
• Integrated profiling, profiling “points”
• Integrated visual web features
• Mobility graphical game builder
• SOA & UML enhancements
• Netbeans platform enhancements
Netbeans 6.0 Resources

• Netbeans Product Site: http://www.netbeans.org
  > Download: previews, current & past releases of IDE, plugins
  > Learning: tutorials, technical articles, flash demos
  > Community: latest news, forums, events, mailing lists

• Netbeans Wiki: http://wiki.netbeans.org/wiki/
  > Open-source documentation site for Netbeans

• Planet Netbeans: http://www.planetnetbeans.org/
  > Aggregate for all Netbeans-related blogs

• Source Code: cvs.netbeans.org:/cvs
  > CVS source code access for Netbeans platform + IDE
Opportunity

Go Get It !
Hello World

The Scene Behind Programming with Java
public class HelloWorld {
    public static void main( String[] arg ) {
        System.out.println( "Hello World" );
    }
}
Compile “Hello World”
Compile ...

HelloWorld.java → Compiler → HelloWorld.class
javac and g++ ??

javac

• compile to binary bytecode
• resolve all reference as symbolic “handles”
• memory layout is defined upon execution

g++

• compile to binary machine code
• resolve all “pointers” into memory address
• memory layout is defined by compiler
class System

```java
System.out.println( "Hello World" );
```
Problem with Pointers

• Fragile Super-class Problem
• Security Problem
Fragile Super-class Problem

Although there is no change in the impl. of B, B is still needed to re-compile.
Security Problem

```java
class A {
    public:
        A() : x(9), y(-9) {}
        int getX() { return x; }
        int getY() { return y; }

    private:
        const int x;
        const int y;
    }

void Hack_Class_A( A *ptr );

int main() {
    printf( "byte size - %d\n", sizeof(byte) );
    printf( "int size - %d\n", sizeof(int) );
    printf( "A size - %d\n", sizeof(A) );

    A instance;
    printf( "x - %d\n", instance.getX() );
    printf( "y - %d\n", instance.getY() );
    printf( "\n" );
    Hack_class_A( &instance );
    printf( "\n" );
    printf( "Hacked =\n" );
    printf( "x - %d\n", instance.getX() );
    printf( "y - %d\n", instance.getY() );
    return 0;
}
```

invoke the Hack_Class routine
The Hack Class Output

Memory layout of class A:
byte 0 - 0
byte 1 - 0
byte 2 - 0
byte 3 - 9
byte 4 - 255
byte 5 - 255
byte 6 - 255
byte 7 - 247
x' 0, y' 255
x' 0, y' 65535
x' 0, y' 16777215
x' 9, y' -9
Hacked =]
x - 9
y - 9

re-construct private variable
Modify the private constant
Run "Hello World"
Run...

HelloWorld.class

Class Loader  Verifier  Interpreter

JVM
Class Loader

Local File System Class Namespace
HelloWorld.class

Network-loaded Class Namespace
WorldWideWeb.class
Verifier

- Verify the executing Bytecode comes from different compiler
- No forging pointers
- No violating access restrictions
- No object is accessed with invalid type
How & Why?

♩ How to manage memory without pointer in Java?

♩ Why need manage memory?
public class HelloWorld {
    public static void main(String[] arg) {
        System.out.println( "Hello World" );
    }
}
public final class String {

    private final char value[];
    private final int offset;
    private final int count;
    private int hash;

}
public class HelloWorld {
    public static void main(String[] arg) {
        System.out.println("Hello World" + "~");
    }
}
public final class String {

    public String concat( String str ) {
        int otherLen = str.length();
        if ( otherLen == 0 ) {
            return this;
        }

        char buf[] = new char[count + otherLen];
        getChars( 0, count, buf, 0 );
        str.getChars( 0, otherLen, buf, count );
        return new String( 0, count + otherLen, buf );
    }
}

return new String, where is the old one?
Where is the old String?

- Un-referenced ... becomes memory leak?
- Deallocated immediately?
- Threw into the Trash?
The Garbage Collector

- Problem of Explicit malloc() and free()
  1. Dangling reference
  2. Memory leak

- GC automatically allocate memory and deallocate memory

- Responsibility:
  1. Allocating memory
  2. Ensuring referenced object in memory
  3. Recovering memory no longer in use
Simple GC

class ReferenceCounter {
  public:
    static void NewReference( Object *obj ) {
      ++obj->reference_count;
    }
    static void ThrowAway( Object *obj ) {
      --obj->reference_count;
      if ( obj->reference_count == 0 ) {
        free( obj );
      }
    }
};
GC Design Choice

- Serial vs Parallel
- Concurrent vs Stop-the-World
- Compact vs Non-compact vs Copying
GC Design in JVM

Pre-HotSpot:

JVM
Conventional Garbage Collection

Post-HotSpot:

Exact JVM (JVM 1.2.2)
Exact Garbage Collection

JVM 1.3
Generational Garbage Collection
Generation Collection

Heap

Young Generation

Old Generation
The Philosophy

- Most object dies young
- Objects survived in several rounds of GC are important
- Large objects are mostly important
- Few references from older to younger objects exist
public int Search( Vector<String> vector, String value ) { 
    Iterator<String> iter = vector.iterator();
    for( int i = 0; iter.hasNext(); ++i ) {
        if ( iter.next() == value ) {
            return i;
        }
    }
    return -1;
}
Generation Characteristic

Young Generation (Minor Collection)
- High garbage density
- Occupy small heap space
- Carry frequent GC

Old Generation (Major Collection)
- Low garbage density
- Occupy big heap space
- Carry few GC
How it works?? (young -> old)

Figure 1. Generational garbage collection
Java Hotspot Generations

- Young generation
  - Eden
    - From
    - To
  - Survivor
- Old generation
- Permanent generation
  - (e.g. Class, Method objects)
Hotspot Collectors

- Serial Collector
- Parallel Collector
- Parallel Compacting Collector
- Concurrent Mark-sweep Collector
Serial Collector — young generation

**Figure 3. Serial young generation collection**

**Figure 4. After a young generation collection**
Serial Collector — old generation

Figure 5. Compaction of the old generation
Parallel Collector – young generation

Figure 6. Comparison between serial and parallel young generation collection
Parallel Compacting Collector — old generation

Mark phase
- live objects are marked in parallel

Summary phase
- calculate density and find the region worth to compact

Compaction phase
- compaction is not carried in the dense prefix
Concurrent Mark-Sweep Collector

old generation

Serial Mark-Sweep-Compact Collector

Stop-the-world pause

Concurrent Mark-Sweep Collector

Initial Mark

Concurrent Mark

Remark

Concurrent Sweep

Figure 7. Comparison between serial and CMS old generation collection
Concurrent Mark-Sweep Collector

Figure 8. CMS sweeping (but not compacting) of old generation
Hotspot Collectors

- Serial Collector
  -XX:+UseSerialGC

- Parallel Collector
  -XX:+UseParallelGC

- Parallel Compacting Collector
  -XX:+UseParallelOldGC

- Concurrent Mark-sweep Collector
  -XX:+UseConcMarkSweepGC
Java Past, Now, and Future

The Evolution Process
Java SE Timeline

1995
- JDK 1.0
- J2SE 1.2 “Playground”

1997
- JDK 1.1
- J2SE 1.3 “Kestrel”

1999
- J2SE 1.4 “Merlin”

2001

2003
- J2SE 5.0 “Tiger”

2005
- Java SE 6

2007

2009
- Java SE 7
History

• 1995 (1.0) – The First Public Release

• 1997 (1.1) – Nested Class Added
  Support for Function Objects

• 2001 (1.4) – Assertions Added
  Support for Verifying Codes
Java SE 5 Language Features

- Autoboxing and unboxing
- Enhanced for loop
- Static import
- Typesafe enumerations
- Variable argument list
- Generics
- Annotations (metadata)

http://java.sun.com/j2se/1.5.0/docs/relnotes/features.html#lang
```java
int sum = 0;
List<Integer> list = Arrays.asList(1, 2, 3);
for (int i: list)
    sum += i;
System.out.println("sum = " + sum);
```
Generics

• New code
  > Example 1

```java
List<String> list = new LinkedList<String>();
list.add("hello world");
String msg = list.iterator().next();
```

> Example 2

```java
public void print(Hashtable<String, Integer> list) {
    for (String key: list.keySet())
        System.out.println(key + " = "
            + list.get(key));
}
```
Autoboxing of Primitive Types

- Old code

```java
ArrayList list = new ArrayList();
list.add(0, new Integer(42));
int total = ((Integer)list.get(0)).intValue();
```

- New Code

```java
ArrayList<Integer> list = 
    new ArrayList<Integer>();
list.add(0, 42);
int total = list.get(0);
```
Enhanced for Loop

• Old code

```java
Iterator iter = hashSet.iterator();
while (iter.hasNext()) {
    Object obj = iter.next();
    ...
}
```

• New code

```java
for (Object obj: hashSet) {
    ...
}
```
Variable Argument List

- Old code

```java
public void printArgs(String[] args) {
    ...
}
```

- New code

```java
printArgs(x, y);
printArgs(x, y, z);

public void printArgs(String... args) {
    for (String a: args)
        System.out.println(a);
```
Java SE 6 Top 10 Feature

- Web Services
- Scripting
- Database
- More Desktop APIs
- Monitoring Management
- Compiler Access
- Pluggable Annotations
- Desktop Deployment
- Security
- The -lities: Quality, Compatibility, Stability

Synchronization Optimization

- All modern JVMs incorporate light-weight locking
  > Avoid associating an OS mutex / condition variable (heavy-weight lock) with each object
- Effective because most locking is uncontended – not competed by threads
Java Object Header

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark Word (32/64bits)</td>
<td></td>
</tr>
<tr>
<td>Class Metadata Address (32/64bits)</td>
<td></td>
</tr>
<tr>
<td>Array Length (32/64bits)</td>
<td></td>
</tr>
</tbody>
</table>

2 words for Object, 3 words for Arrays
Mark Word

<table>
<thead>
<tr>
<th>bitfields</th>
<th>tag bits</th>
<th>state</th>
</tr>
</thead>
<tbody>
<tr>
<td>hash age 0</td>
<td>01</td>
<td>unlocked</td>
</tr>
<tr>
<td>ptr to lock record 00</td>
<td>lightweight locked</td>
<td></td>
</tr>
<tr>
<td>ptr to heavyweight monitor 10</td>
<td>inflated</td>
<td></td>
</tr>
<tr>
<td>thread ID epoch age 1 01</td>
<td>marked for GC biasable</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 1.** Synchronization-related states of an object’s mark word.
Light Weight Locking
-unlocked

Execution Stack
Method Activation
Lock Record

hash + age | 01

Object Header
Thread stack
Light Weight Locking -locked

Execution Stack
- Method Activation
- hash + age | 01

Stack Pointer

CAS

Object Header

Displaced Mark Word

Thread stack
Light Weight Locking -contented

Mutex Pointer

Object Header

Mutex
OR
Condition Variable

Memory
Light Weight Locking -recursed

Execution Stack

Method Activation

hash + age | 01 recursion count

Thread stack

CAS

Stack Pointer

Object Header
Observation

- Most lockings are not only uncontended, but performed repeatedly by the same thread

>>> Make it cheap for a single thread to reacquire a lock can be an optimization
Biased Locking

Figure 2. State transitions of an object’s mark word under biased locking.
What happens when Revoke?

- Fallback to light weight locking
- Need to wait for Global Safepoint (No bytecode is executing)
- Thread stack is walked and lock record is enumerated
- Update Object Header when object is locked
Revoking ...

Thread stack

- Execution Stack
- Method Activation
- Displaced Mark

Stack Pointer

Object Header

Update

walk
Comparing the Lockings

Light Weight Locking
- lock (CAS) : unlock (CAS)
- lock (CAS) : unlock (CAS)

Biased Locking
- lock (CAS) : unlock (TAB)
- lock (TAB) : unlock (TAB)

Execution
JavaSE 7
Welcome to the Open-Source JDK Community!

This is the place to collaborate on the open-source JDK, an implementation of the Java Platform, Standard Edition specification defined in the Java Community Process.

How to participate

You can participate in this Community in many ways; here are some of the possibilities:

- Check out the latest blogs on Planet JDK
- Join the discussion on one or more of the mailing lists
- Join the live conversation via IRC on irc.oftc.net #openjdk
- Get to know the existing Groups that make up the Community
- Survey the growing set of Projects
- Read the tutorial on how to build and hack on the OpenJDK code with the NetBeans IDE
- Browse the code on the web, or use Mercurial to make your own local clone
- Contribute a patch to fix a bug, enhance an existing component, or define a new feature
Where Are We?

JDK 5 2004/9/30
JDK 6 2006/12/11
JDK 7 2008/H2
Future?

Figure 4: The Strategic Coexistence Of .NET And J2EE Wins In European Financial Services

“Do you strategically use J2EE and/or .NET?”

- J2EE: 40%
- .NET: 7%
- Both: 40%
- Don’t know: 6%
- No strategic decision: 7%

Base: 55 European enterprise architects at financial services firms

Source: European Financial Services Architecture Strategy Online Survey 2007

Source: Forrester Research, Inc.

Source: Forrestor’s Architect Survey on European Financial Industry in Jan 2008

Further Readings

http://openjdk.java.net/
http://java.sun.com/docs/white/langenv/
http://java.sun.com/j2se/1.5.0/docs/relnotes/features.html#lang
Q & A