Monitoring Java applications and detecting performance problems using JDK tools and NetBeans Profiler

Tomáš Hůrka, Jiří Sedláček
tomas.hurka@sun.com, jiri.sedlacek@sun.com
Agenda

● Introduction

● Problems of Java applications

● Monitoring tools in JDK
  ● JDK 5.0
  ● JDK 6.0

● Performance diagnosis, profiling
  ● Profilers, profiling techniques
  ● NetBeans Profiler
Introduction

 Goals of this session:

 - Describe typical problems of Java applications
 - Present tools and techniques to discover these problems
Agenda

● Introduction
● Problems of Java applications
● Monitoring tools in JDK
  ● JDK 5.0
  ● JDK 6.0
● Performance diagnosis, profiling
  ● Profilers, profiling techniques
  ● NetBeans Profiler
Problems of (Java) applications

- User point of view
  - Startup time
  - Response time
  - Memory footprint
Problems of (Java) applications

● Threading problems
  ● Blocking EDT, deadlocks etc.

● Performance problems
  ● Inefficient algorithms, bad scalability etc.

● Memory problems
  ● Wasting memory, memory leaks
Problems of Java applications

- Special to Java
  - Compiled vs. interpreted code
  - Classloading
  - Garbage Collection
  - Tuning JVM parameters
**Agenda**

- Introduction
- Problems of Java applications
- Monitoring tools in JDK
  - JDK 5.0
  - JDK 6.0
- Performance diagnosis, profiling
  - Profilers, profiling techniques
  - NetBeans Profiler
Monitoring JDK 5.0

- Solaris, Linux, Mac OS X
  - jstack, jmap, jinfo commandline utilities
  - Post-mortem analysis of thread stacks, heap, ...
  - Can also inspect live process
- Solaris 10:
  - Dtrace jstack action
Monitoring JDK 5.0

- Windows
  - jps
- jconsole
- 5.0u7 adds `-XX:+HeapDumpOnOutOfMemoryError`
- Troubleshooting guide
Monitoring JDK 5.0

- DEMO
Monitoring JDK 6.0

- Improvements to commandline utilities
  - live process inspection (Windows)
  - capture heap dump
    - `jmap -dump:file=<file>,live <pid>`
  - `jhat`
  - change “manageable” options/flags dynamically
- Built-in DTrace probes (Solaris 10)
Monitoring JDK 6.0

- Improvements to commandline utilities
  - expose `java.util.concurrent` lock information
  - `jstack -l`

- `jconsole` improvements
  - start management agent in local VM
  - plug-in support
  - UI improvements
Monitoring JDK 6.0

● DEMO
Agenda

- Introduction
- Problems of Java applications
- Monitoring tools in JDK
  - JDK 5.0
  - JDK 6.0
- Performance diagnosis, profiling
  - Profilers, profiling techniques
  - NetBeans Profiler
Profilers, techniques

- **Profiler**
  - Tool that tracks the performance of computer program
  - JVM metrics, graphs, call trees, execution times, memory usage insights, heuristics etc.

- **Techniques**
  - JVM hooks, sampling, instrumentation
NetBeans Profiler

- Integrated into NetBeans IDE
- Profiler 5.5.1 (stable) or 6.0 Beta 1
- Dynamic bytecode instrumentation, root methods, calibration
NetBeans Profiler Demos (6.0 Beta 1)

- DEMO: Application monitoring, threads
- DEMO: CPU profiling, optimization
- DEMO: Memory analysis, leak detection
Conclusion

- Powerful tools for application monitoring available in JDK
- Profilers help you to discover performance & memory problems in your applications
- You can use NetBeans Profiler for free:o)
Resources

● **JDK 5.0 Troubleshooting Guide**

● **JDK 6.0 Troubleshooting Guide**

● **Java Performance Tuning**
  - [http://www.javaperformancetuning.com](http://www.javaperformancetuning.com)

● **NetBeans Profiler (download, docs...)**
  - [http://profiler.netbeans.org](http://profiler.netbeans.org)
Monitoring Java applications and detecting performance problems using JDK tools and NetBeans Profiler

Tomáš Hůrka, Jiří Sedláček
tomas.hurka@sun.com, jiri.sedlacek@sun.com