mathema campus 2009

Eins für alles?

Monitoring und Profiling mit VisualVM

Thomas Much

thomas@muchsoft.com
www.muchsoft.com

MATHEMA

Referent

- Dipl.-Inform. Thomas Much

- IT-Architekt, Softwareentwickler & Projektcoach
  - Java seit 1996, C++ seit 1993, davor ObjectPascal, Eiffel ...
- Programmiertrainer
  - seit 1997
- Autor
  - "Java für Mac OS X"
    Galileo Computing 2005
    630 Seiten, ISBN 3-89842-447-2

Agenda

- Monitoring und Profiling
- JDK-Tools
- VisualVM
- VisualVM mit GlassFish (Sun Application Server)
- Fazit

Monitoring und Profiling

Software-Probleme (1)

- Arten
  - dauerhafte Performance-Probleme
  - über die Zeit anwachsende Performance-Probleme
  - sporadische Performance-Probleme
  - Überlast
  - gelegentliche Fehler
  - regelmäßige Fehler

Software-Probleme (2)

- Gründe
  - Anwendungsfehler
  - HTTP-Session
  - Datenbankprobleme, schlechtes SQL
  - Netzwerküberlastung
  - ...
- Abhilfe: Monitoring und Profiling

Monitoring und Profiling

- Monitoring
  - beschreibt Erfassung von Zuständen, die Beobachtung, Überwachung oder Steuerung eines Vorgangs (Prozesses) mittels technischer Hilfsmittel
- Profiling
  - Einsatz von Programmwerkzeugen (Tools), die das Laufzeitverhalten von Software analysieren

➔ Monitoring entdeckt Probleme,
Profiling geht ihnen auf den Grund.

Performance-Profiling

Wie kann Performance-Profiling durchgeführt werden?

- Eigene Zeitmessung
  - `System.currentTimeMillis(), System.nanoTime()`
- JVM-Schnittstellen (JVMPI, JVMTI)
- Bytecode-Instrumentierung

JVMPI / JVMTI

- JVMPI
  - JVM Profiler Interface
  - seit Java 1.1 dabei
  - aber immer nur "experimentell" mit zahlreichen Problemen
  - seit Java 5.0 deprecated, seit Java 6 nicht mehr dabei
- JVMTI
  - JVM Tool Interface
  - seit Java 5.0 dabei
  - löst JVMPI (und Debugger-API JVMDI) ab

→ zahlreiche Tools bauen darauf auf

JDK-Tools

JDK-Tools

- **jinfo, jps**
  - Prozess-Informationen und -Ids
- **jstack**
  - Prozess-Stacktrace
- **hprof**
  - Zeit- und Speicher-Profiling
- **jmap, jhat**
  - Speicher-Dumps, Histogramme, Auswertung als Web-Server
- **jstat, jstatd**
- **jconsole**

jstat / jstatd

- `jstat -gcutil` *pid ms count*↵

oder remote: pid@host:port

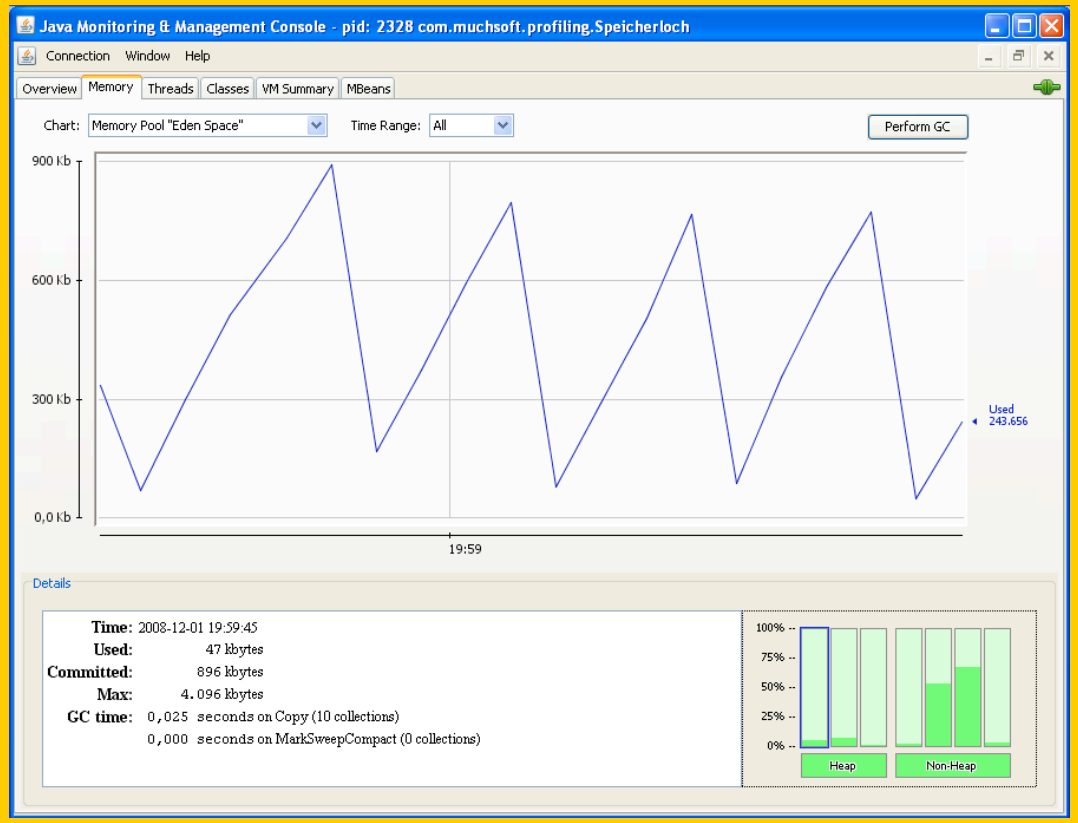
- `start rmiregistry`↵
  `jstatd`
  `-J-Djava.security.policy=jstatd.all.policy`↵

jstatd.all.policy

```
grant codebase "file:${java.home}/../lib/tools.jar" {
   permission java.security.AllPermission;
};
```

JConsole

- seit JDK 5.0 fester Bestandteil
- Aufruf mit `jconsole`



VisualVM

VisualVM (1)

- Download von <*https://visualvm.dev.java.net/*>
- seit JDK 6u7 fester Bestandteil (!)
- NetBeans-RCP-Anwendung

- benötigt Java 6 als Runtime

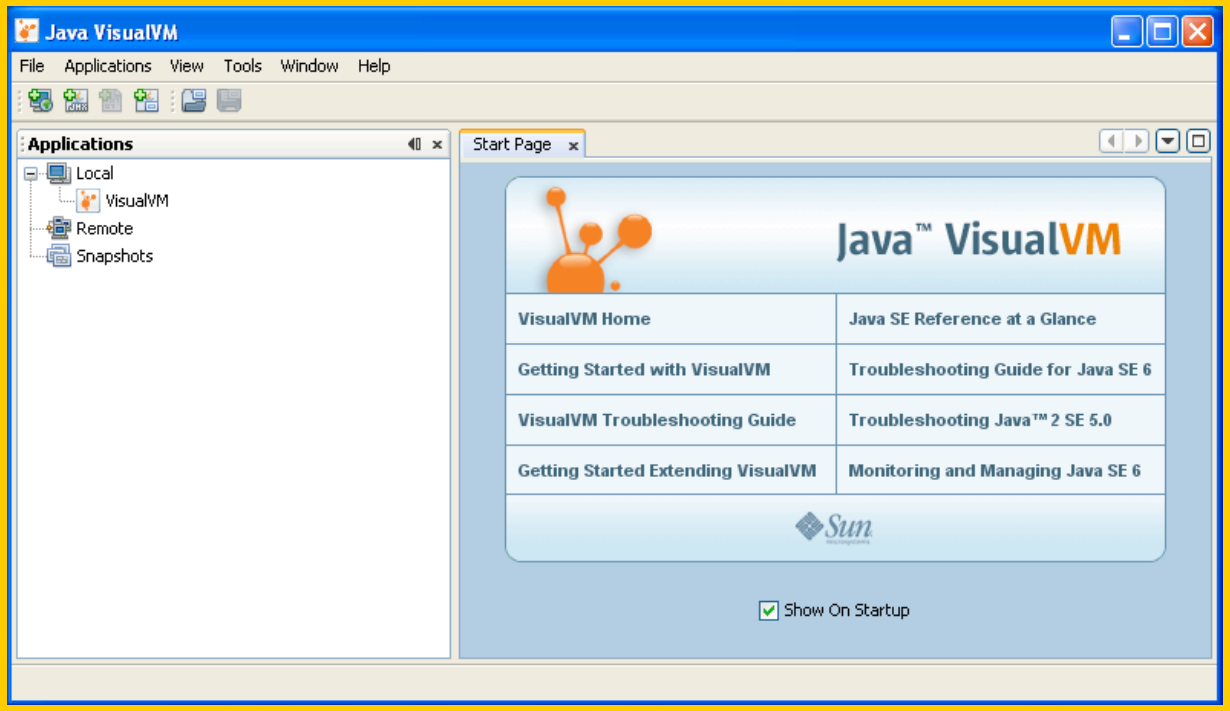
- kann Anwendungen mit Java 1.4, 5.0 und 6 untersuchen
  - je neuer die Ziel-JVM ist, desto mehr Informationen stehen zur Verfügung

VisualVM (2)

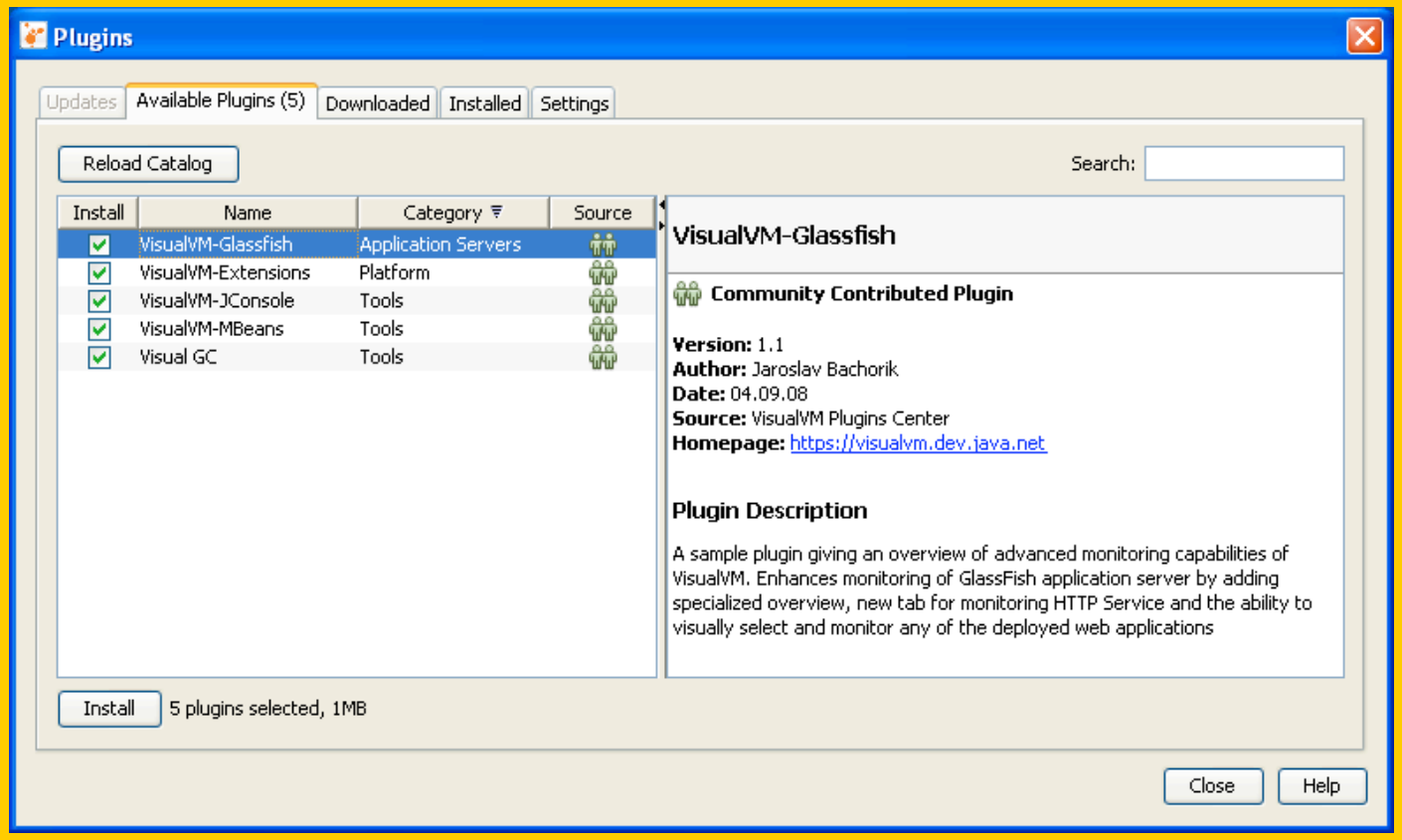
Feature	JDK 1.4.2 local/remote	JDK 5.0 local/remote	JDK 6 local	JDK 6 remote
Overview	✓	✓	✓	✓
System Properties			✓	
Monitor	✓	✓	✓	✓
Threads		✓	✓	✓
Profiler			✓	
Thread Dump			✓	
Heap Dump			✓	
Heap Dump on OOME			✓	
MBean Browser		✓	✓	✓
JConsole Wrapper		✓	✓	✓

VisualVM (3)

- Aufruf mit `jvisualvm`
- beim ersten Start erfolgt eine Kalibrierung



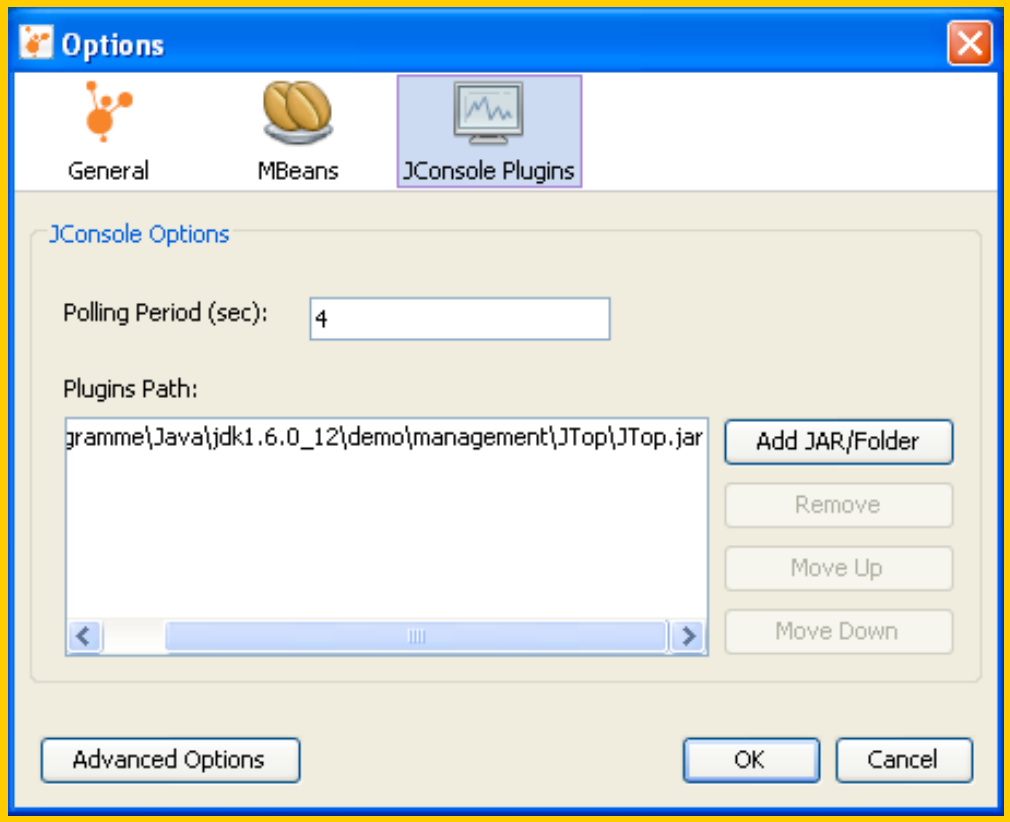
VisualVM: Plugins



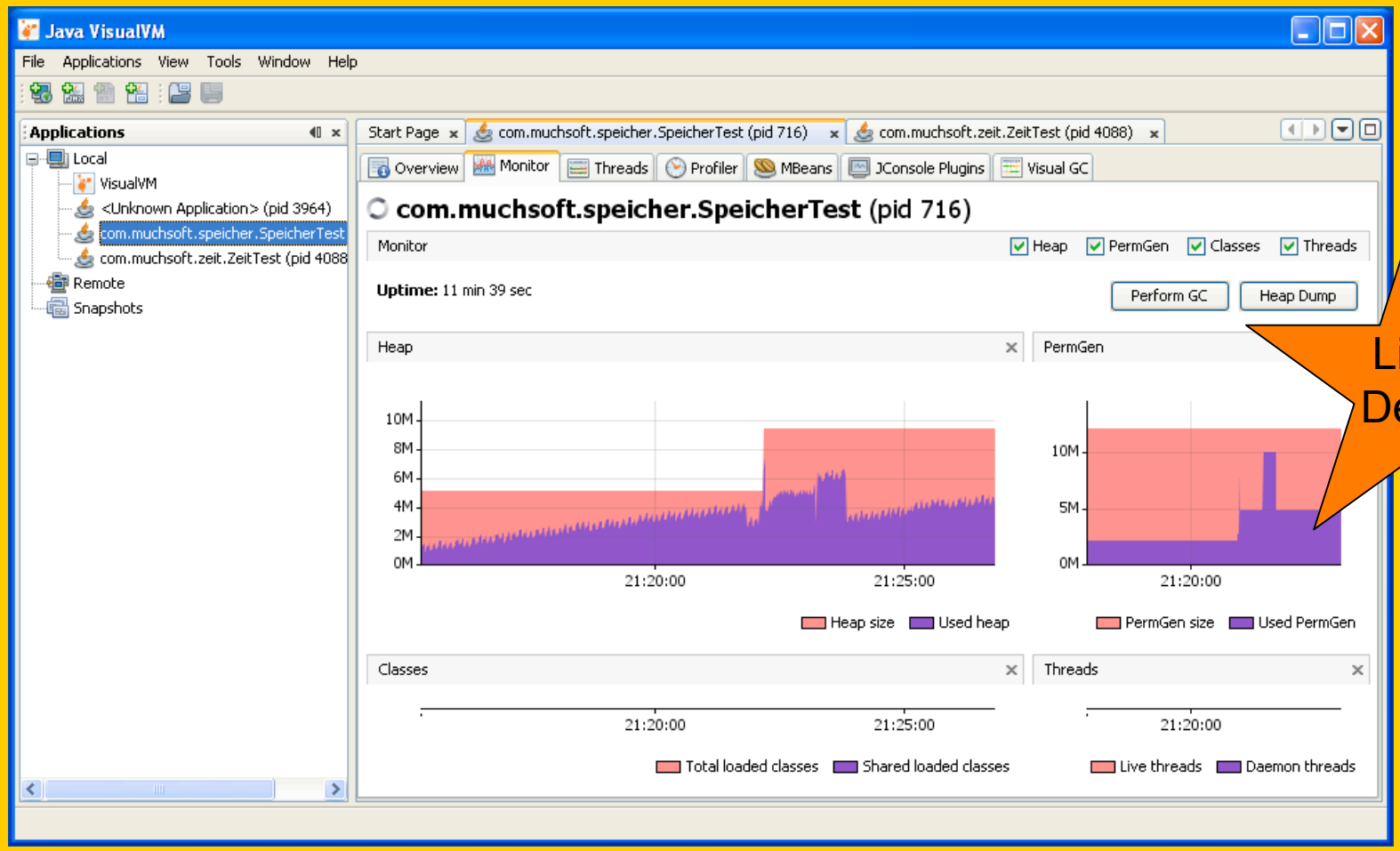
- am besten alle Plugins installieren, vor allem VisualGC

VisualVM: Optionen / JTop

- mit dem JConsole-Plugin kann z.B. JTop installiert werden:

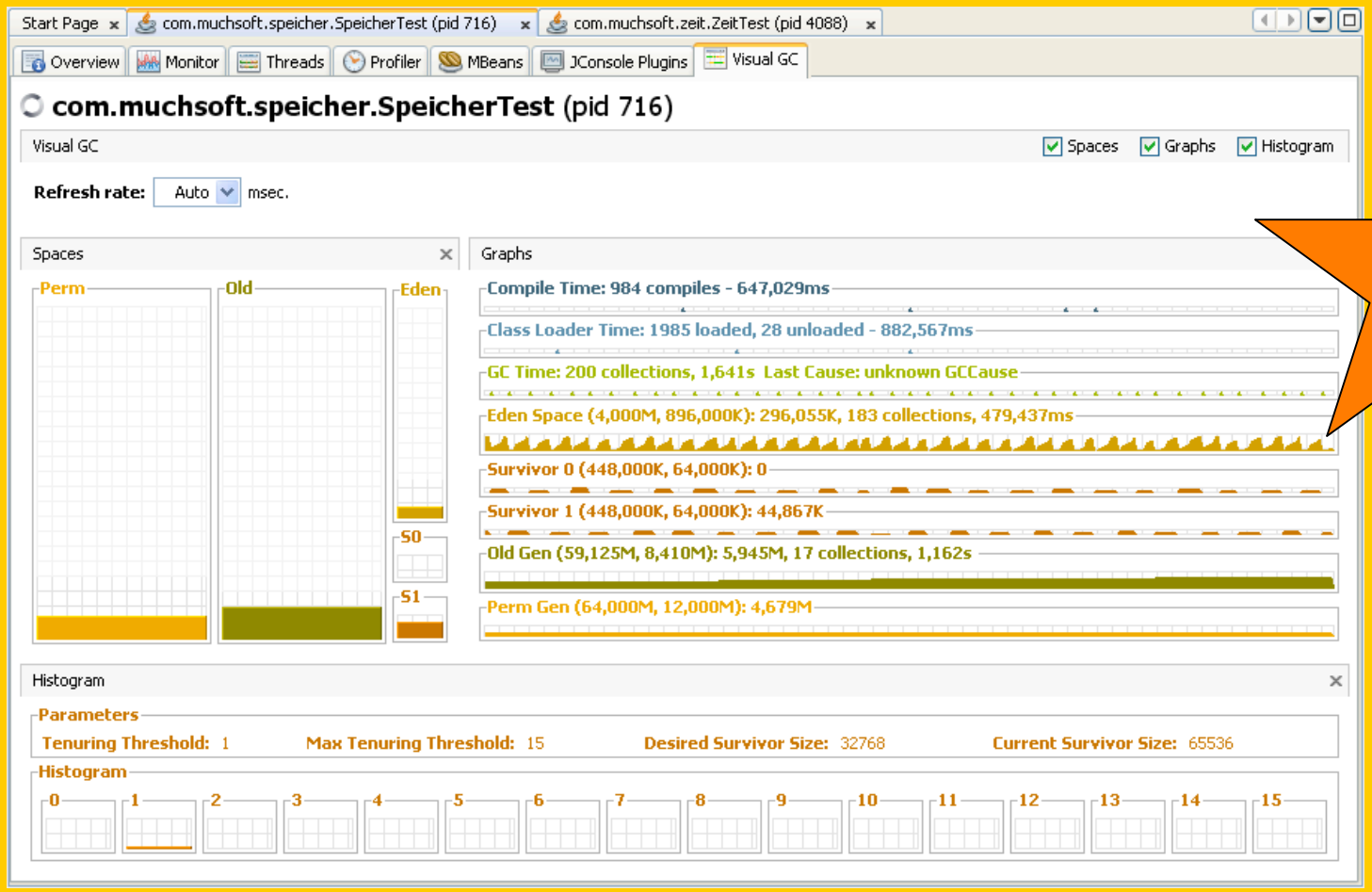


VisualVM: Monitoring



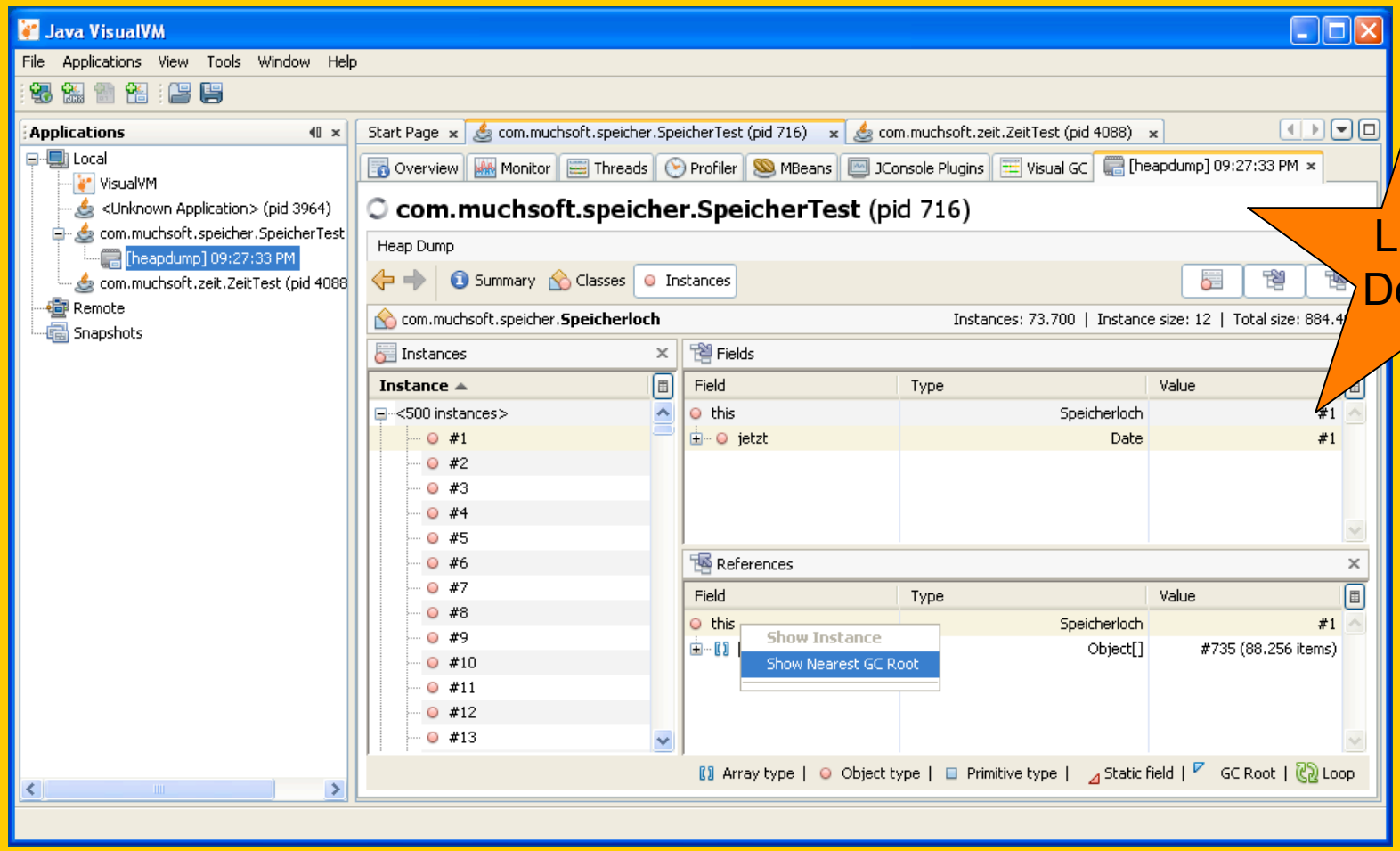
Live-Demo

VisualVM: Visual GC

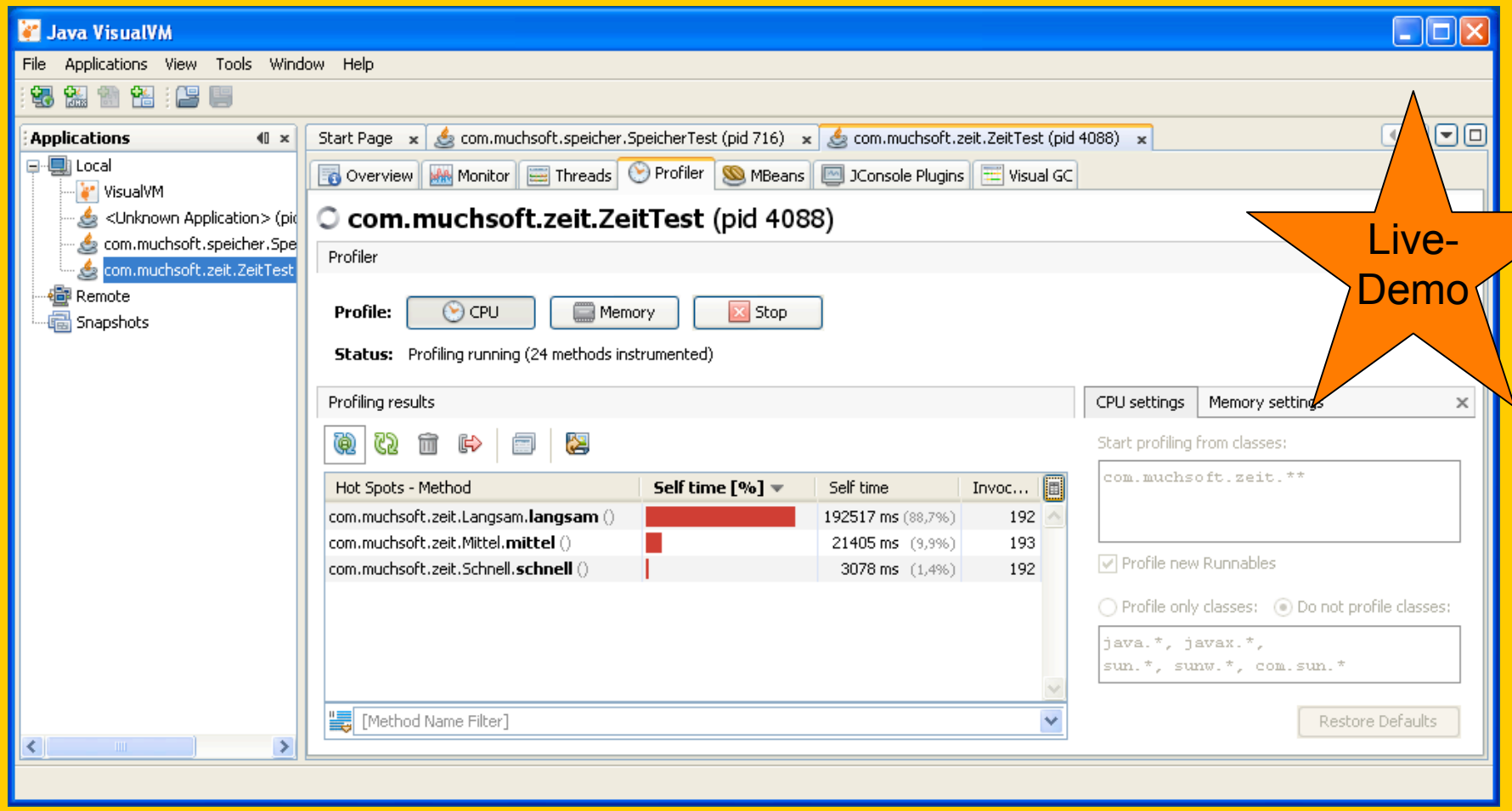


Live-Demo

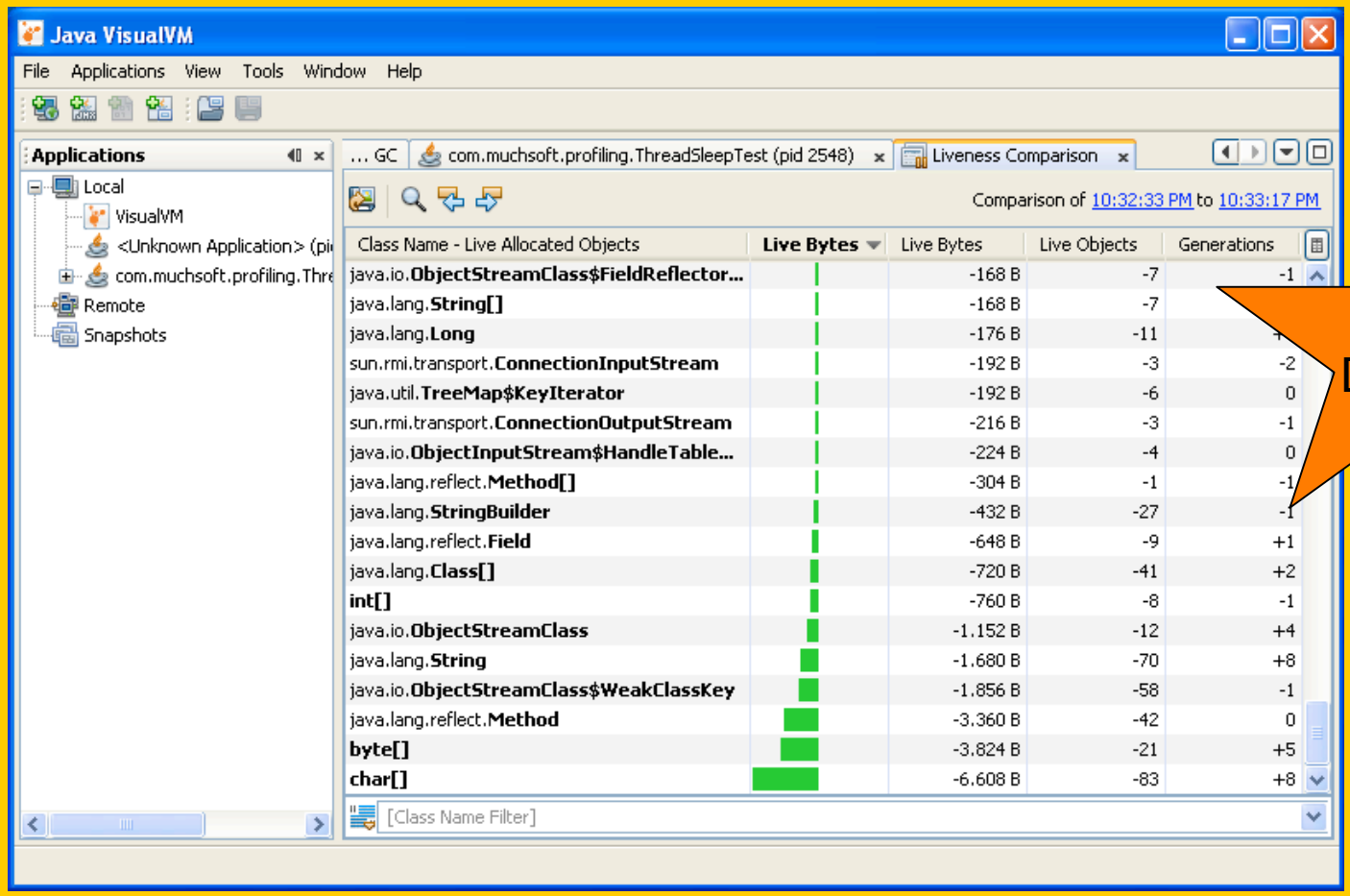
VisualVM: Speicherlecks finden



VisualVM: Profiling



VisualVM: Memory-Snapshots vergleichen



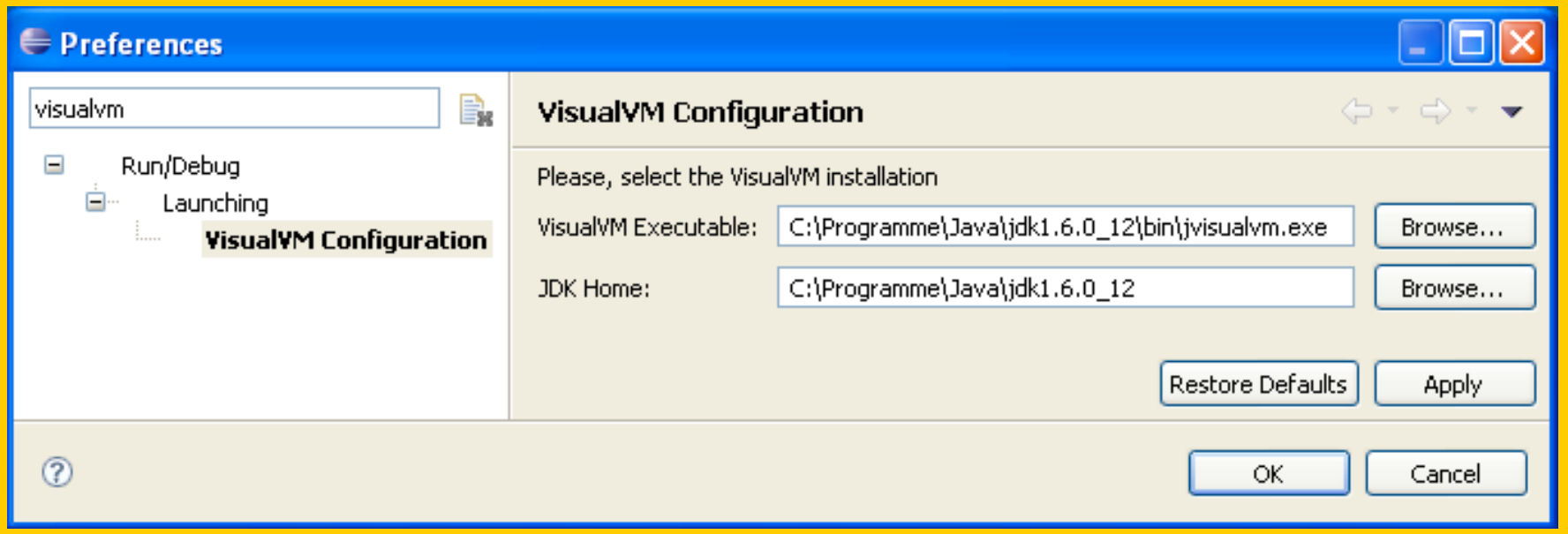
Live-Demo

VisualVM: Probleme

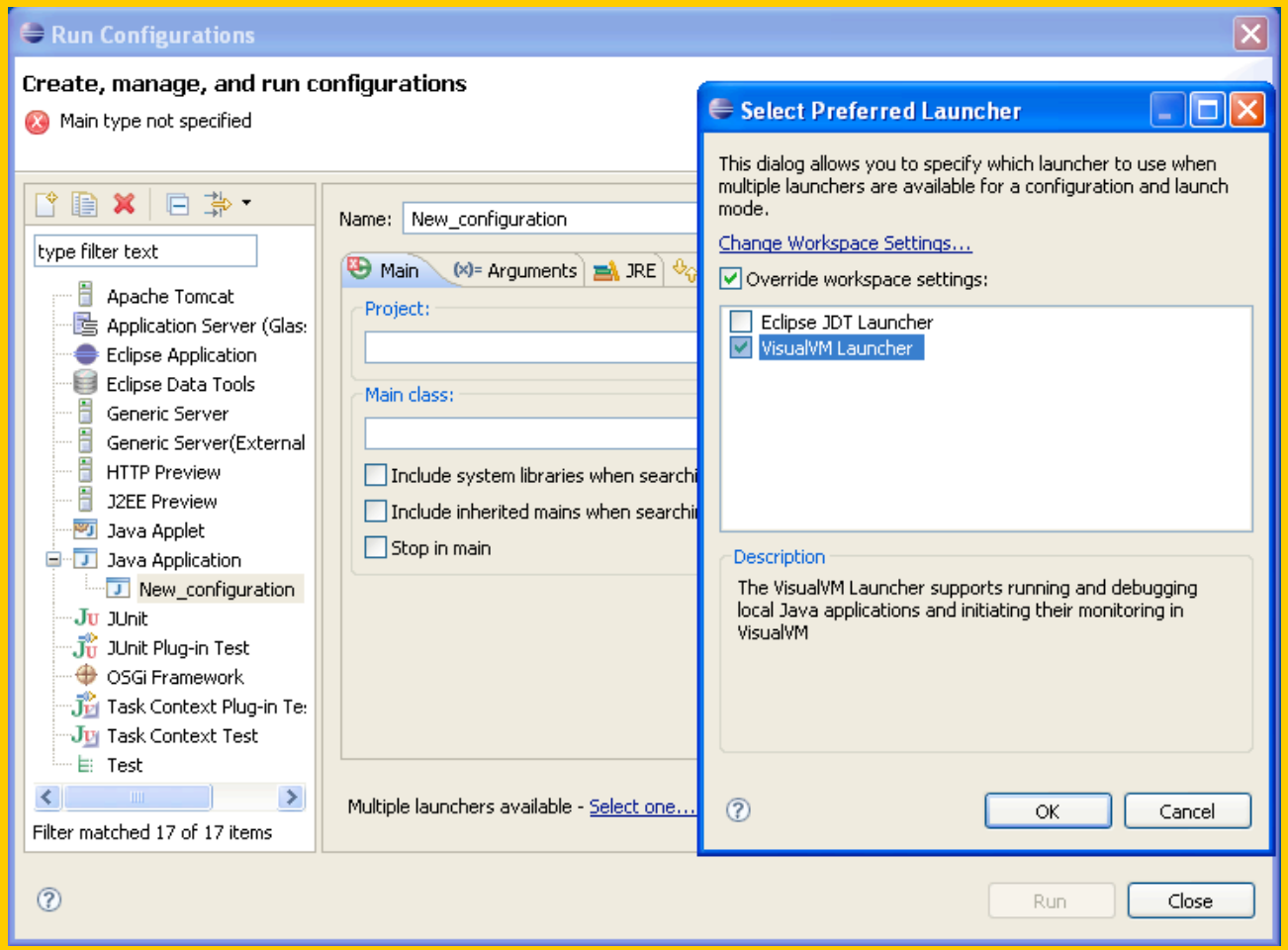
- VisualVM ist noch relativ neu, es gibt noch ein paar Probleme... siehe <*https://visualvm.dev.java.net/troubleshooting.html*>
- beispielsweise kann das Profiling lokale Ziel-JVMs zum Absturz bringen
  - Class-Sharing deaktivieren
  - Ziel-JVM mit `-Xshare:off` starten

Eclipse: VisualVM-Plugin

- Download von *<https://visualvm.dev.java.net/eclipse-launcher.html>*
- Konfiguration:



Eclipse: Anwendung+VisualVM starten

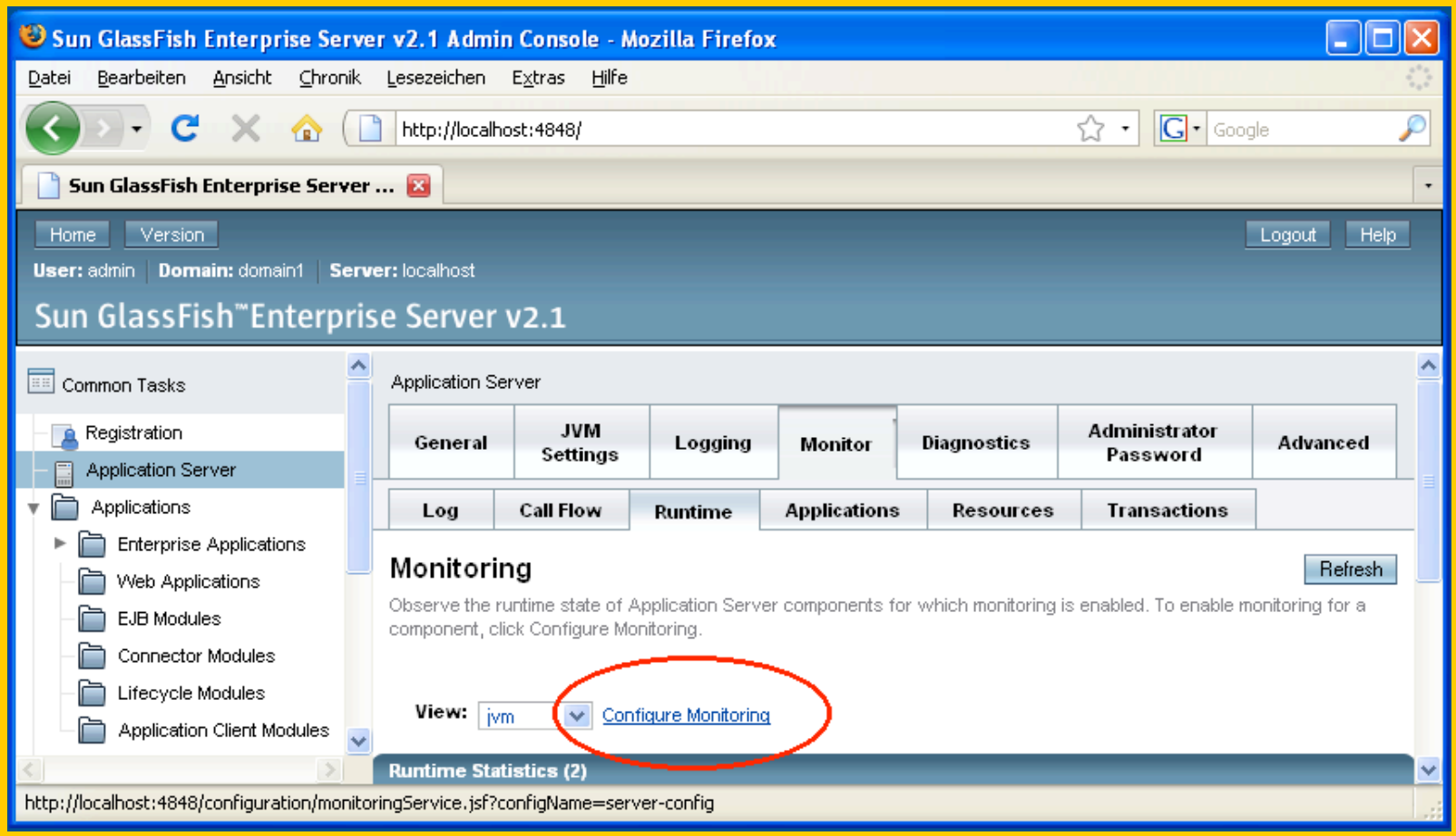


VisualVM mit GlassFish

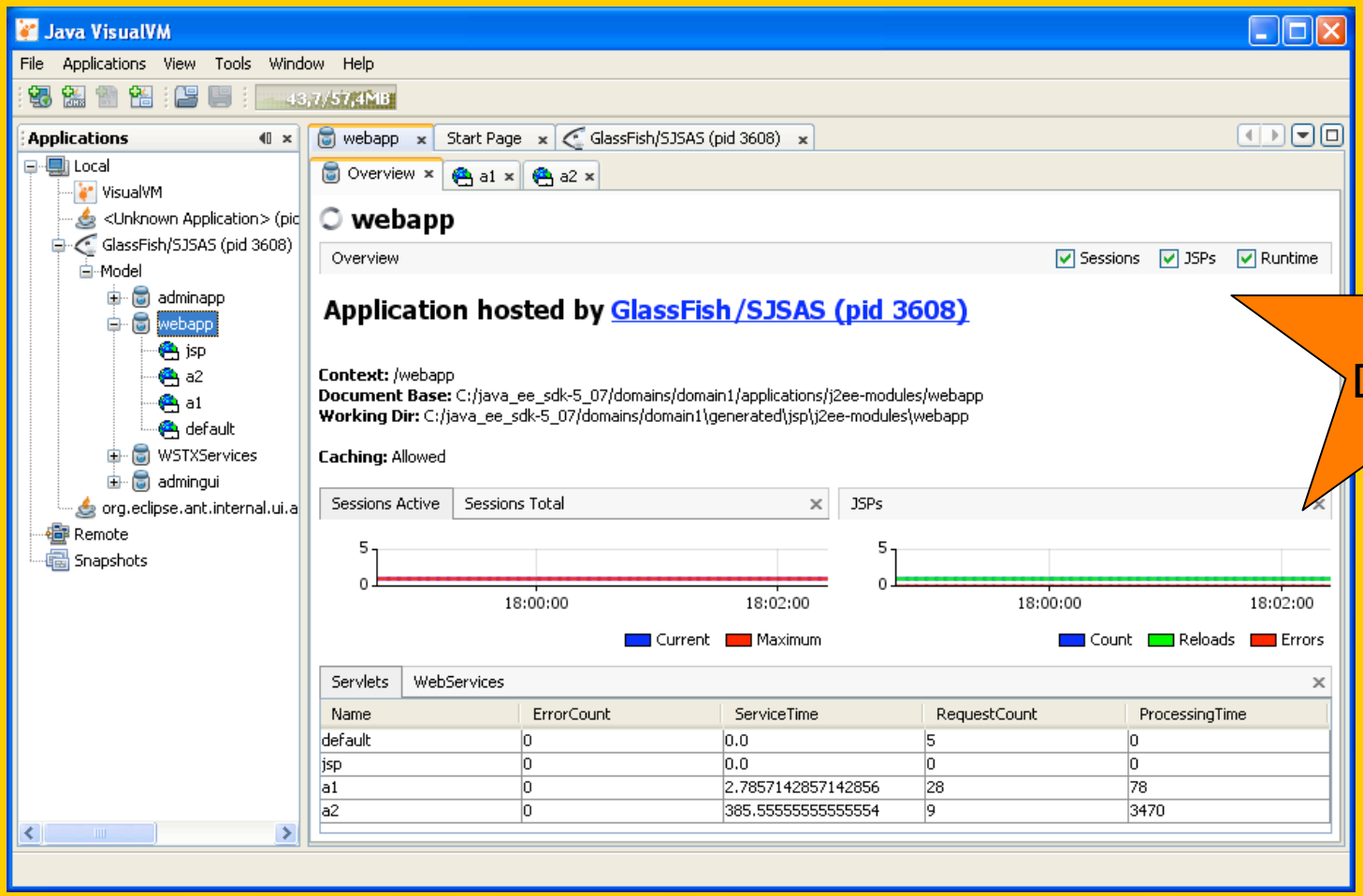
VisualVM mit GlassFish

- GlassFish starten
- für Remote-Zugriff: `jstatd` starten
  - auf dem Server muss der RMI-Port (1099) offen sein
- VisualVM starten
  - GlassFish-Plugin verwenden
  - Monitoring in GlassFish aktivieren

GlassFish: Monitoring aktivieren



GlassFish: Monitoring mit VisualVM



Live-Demo

VisualVM: Fazit

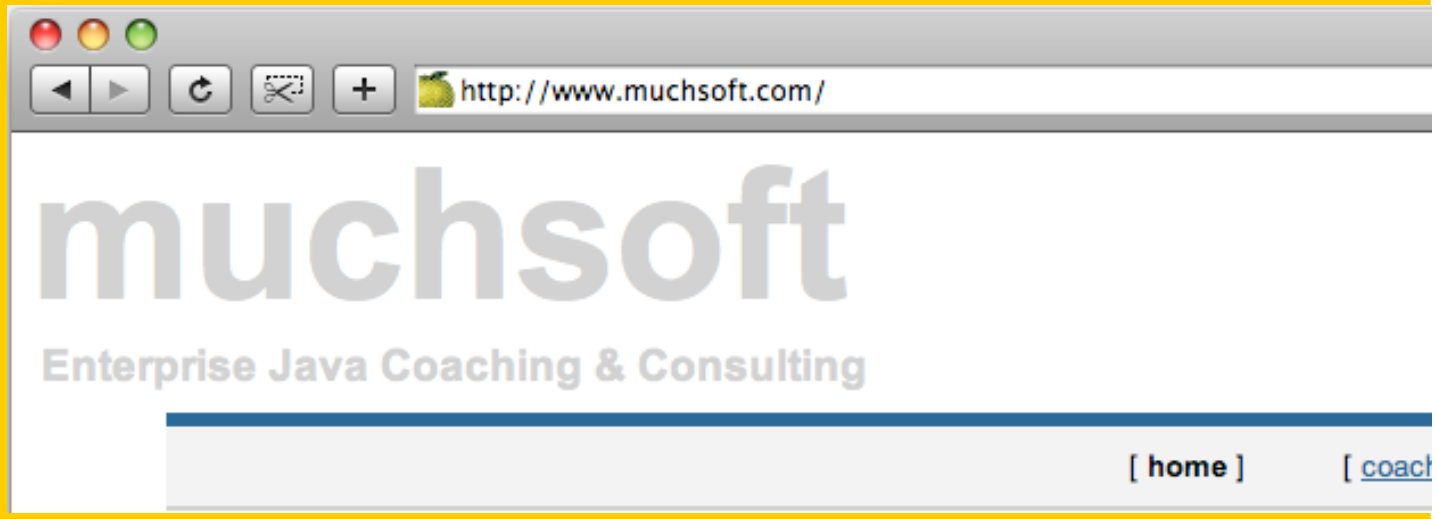
VisualVM: Fazit

- Positiv:
  - einfach zu installieren und einzusetzen
  - Open Source, kostenfrei, fester Bestandteil vom JDK
  - soll laut Sun zur Standard-Management-Anwendung von Java-Applikationen werden
  - viel Potential!
- Negativ:
  - noch nicht 99%ig stabil
  - kein Remote-Profiling
  - keine Instrumentierung (Probes o.ä.)
  - → hierfür TPTP oder kommerzielle Produkte erforderlich

mathema campus 2009

Fragen?

Vielen Dank!



MATHEMA