

ORACLE®

Agenda

- VisualVM – a brief overview
- Bringing in extensions
- Powerusers' tips & tricks
- Wrap up

What Is VisualVM?

- Unified access to a bunch of monitoring tools already available in JDK
 - **jps**, **jstat**, **jmap**, **jstack** and **jhat**
- Built on top of NetBeans Platform (RCP)
- Open source project started in 2008
 - <http://visualvm.dev.java.net>
- Included in Sun JDK distribution since 1.6.0_7
 - the name is **jvisualvm**
 - see <https://visualvm.dev.java.net/releases.html> for mapping between the tool version and jdk version
- Latest version 1.3.1 (just released)

Built-in Capabilities

- Auto-discovery of JVMs
 - local
 - remote over **jstatd**
- Remote JVMs over **JMX**
- Application args, JVM flags, System properties
- Basic JVM telemetry (CPU, Memory, Threading)
- Generating thread/heap dump (even remotely)
- Application profiling
 - instrumented
 - sampled (*since 1.3*)

Demo



Extensibility

- Modular application built on NetBeans Platform (RCP)
- Cleanly defined APIs
 - datasources
 - views
 - applications
 - preferences
- Plugin center
 - VisualVM 1.3 - 23 tested and verified plugins for
 - any 3rd party plugin center can be added (eg. BTrace)
- Developer starting point
<https://visualvm.dev.java.net/api-quickstart.html>

Extensibility: Plugins

- MBeans
 - visual mbeans browser
- Visual GC
- Extensions
 - updates for new JVMs etc.
- Security
 - setting up keystore for SSL
- JConsole
 - JConsole plugins wrapper

Extensibility: Plugins !NEW!

- Threads Inspector
 - enhancing thread behavior analysis
- Tracer
 - displaying various metrics in co-related timeline
 - easily extensible by custom probes
 - readily available probes for
 - Swing: paints, updates, layouts etc.
 - JavaFX: pulses, events etc.
 - JVM internals: JIT, GC etc.
 - jvmstat perf counters
 - not depending on the underlying technology

Demo



Tips and Tricks

- Remote access to **JVM**
- Monitoring JVMs running as Windows services
- Unleashing the power of OQL

Remote Access to JVM

- Run **jstatd**
 - jstatd = remote proxy for jvmstat
- Enable JMX support

Running jstatd

- jstatd tool available in JDK
- Needs security policy in place
 - defined by system property `java.security.policy`

- Policy file – allow all

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

- Do not forget to customize the permissions
- RMI server host name
 - defined by system property `java.rmi.server.hostname`
 - Necessary for applications on Ubuntu

Remote JVM – JMX Setup

- Specify system properties for the application
 - `com.sun.management.jmxremote.port=<port>`
 - `com.sun.management.jmxremote.authenticate=true/false`
 - `com.sun.management.jmxremote.ssl=true/false`
 - value “false” not recommended for production
 - needs more configuration when turned on
 - `java.rmi.server.hostname=<host name>`
 - necessary for applications on Ubuntu

Remote JVM – JMX over SSL

- Generate keystore
 - `keytool -genkey -keystore mySrvKeystore -keyalg RSA`
- System properties for VisualVM
 - `javax.net.ssl.keyStore`
 - `javax.net.ssl.keyStorePassword`
- System properties for application
 - `javax.net.ssl.trustStore`
 - `javax.net.ssl.trustStorePassword`

Demo



Monitoring JVMs running as Windows services

- Run VisualVM or jstatd as Windows service
 - *instsrv.exe* and *srvany.exe* from eg. **Windows Server 2003 Resource Kit Tools**
 - modify registry to add the declared service
 - use “Local System” account
 - enable **Allow service to interact with desktop**
- Step-by-step guide available
 - http://blogs.sun.com/nbprofiler/entry/monitoring_java_proc...

Analyzing Heap with OQL

- OQL = **O**bject **Q**uery **L**anguage
- *select s from java.lang.String s where s.count > 0*
- Not a standardized language – using jHat dialect
 - JavaScript based engine
 - can be extended by custom JavaScript functions
 - provides many extensions for heap analysis
 - <https://visualvm.dev.java.net/oqlhelp.html>
- VisualVM adds
 - syntax highlighting
 - query persistence
 - integration with heap walker

Demo



Wrap Up





SOFTWARE. HARDWARE. COMPLETE.