WebSphere User Group UK

21st March 2013

Build your own runtime monitoring with the Health Center API

Speaker : Toby Corbin

Agenda

- Health Center overview
  - What is it ?
  - How to enable it
- Health Center demo
- Health Center API
- Deadlock monitoring example
- Method Profiling monitoring example

Health Center overview

- What problem am I solving?
  - What is my JVM doing? Is everything ok?
  - Why is my application running slowly? Why is it not scaling?
  - Am I using the right options?
- Live monitoring tool with very low overhead
- Understand how your application is behaving
  - Monitor Class loading, File I/O, Environment settings, Garbage Collection, Method Profiling, Locking, Native memory use, Threads
- Diagnose potential problems, with recommendations
- Works at the JVM level
- Suitable for all Java applications
- Powerful API allowing embedding of Health Center into other applications

Health Center overview cont.

- The tool is provided in two parts:
  - An agent that collects data from a running application.
  - An Eclipse-based client that connects to the agent.
- The Agent ship’s with the following vm’s:
  - Java 5sr9 and upwards
  - Java 6sr3 and upwards
- The latest version of the agent is always available from within the Health Center Client
  - Recommended to always update to the latest version of the agent
- Agent package unzips over the jre directory of the JVM you are using to run the application to monitor

How to enable an application for monitoring

- Full instructions are provided within the help shipped with the Health Center Client but in most cases as simple as :

  For 5 SR9 and earlier, or Java 6 SR4 and earlier

  ```
  java -agentlib:healthcenter -Xtrace:output=healthcenter.out HelloWorld
  ```

  For Java 5 SR10 and later, or Java 6 SR5 and later (including Java 7)

  ```
  java -Xhealthcenter HelloWorld
  ```

Health Center Demo

Health Center API

- The 2.1 release of Health Center contains a powerful API. The API allows Java™ developers to embed Health Center in their applications

- With a few lines of code, you can embed the monitoring power of Heatlh Center in your own Eclipse based application and harness its monitoring power to troubleshoot problems

```
// Create the connection object:
ConnectionProperties conn1 = new ConnectionProperties("localhost", 1973);
// Connect to the Health Center agent, using the previous connection settings:
HealthCenter hcObject = HealthCenterFactory.connect(conn1, true);
// Get garbage collection data and print:
GCData gcData = hcObject.getGCData();
System.out.println("GC Mode is " + gcData.getGCMode().toString());
```

Health Center API cont.

- Each area of data is accessed from methods on the HealthCenter object

```
getClassesData()
getEnvironmentData()
getGCData()
getIOData()
getLockingData()
getMethodTraceData()
getNativeMemoryData()
getProfilingData()
getThreadsData()
```

Health Center API cont.

- Each area has its own methods
- For example, the **`ThreadsData`** object returned from **`HealthCenter.getThreadsData()`** has methods

```
deadlockDetected()
getThreadCounts()
getThreads()
```

Deadlock monitoring example

Method Profiling monitoring example

Where to find more information

- IBM Monitoring and Diagnostic Tools for Java™ on developerWorks
  http://www.ibm.com/developerworks/java/jdk/tools/
- Health Center API articles
  - Monitor a Java application with the Health Center API parts 1 and 2
    - http://www.ibm.com/developerworks/library/j-healthcareapi1/index.html
    - http://www.ibm.com/developerworks/library/j-healthcareapi2/index.html
- IBM Support Assistant (ISA)

  http://www.ibm.com/software/support/isa
- Email javatool@uk.ibm.com

Any Questions?