IBM Cúram Social Program Management

Cúram JMX Developer Guide

Version 6.0.4

Note

Before using this information and the product it supports, read the information in Notices at the back of this guide.

This edition applies to version 6.0.4 of IBM Cúram Social Program Management and all subsequent releases and modifications unless otherwise indicated in new editions.

Table of Contents

Chapter 1

Introduction

1.1 Purpose

The purpose of this guide is to describe how Cúram JMX can be extended with custom MBeans.

1.2 Audience

This guide is intended for application developers interested in providing custom operational data via Cúram JMX.

Chapter 2

Cúram JMX

2.1 What is Cúram JMX

Cúram Java Management Extensions (JMX) is an infrastructure that simplifies the instrumentation of code and the collection of application operational data using the JMX standard. Cúram JMX facilitates the creation of custom MBeans and their registration in the correct MBean server corresponding to the runtime environment.

2.2 Using Cúram JMX to Expose Application Statistics

In order to collect and expose custom application statistics an MBean needs to be created, the application code instrumented to provide the statistics and the JMX infrastructure configuration modified to initialize the newly created MBean.

2.2.1 Developing the Custom MBean

Cúram JMX supports only Open MBeans. An Open MBean is an MBean where the types of attributes and of operation parameters and return values are built using a small set of predefined Java ® classes. A multidimensional array of any one of these classes or their corresponding primitive types is also allowed. These acceptable Java data types are listed below.

- java.lang.Void
- java.lang.Boolean
- java.lang.Character
- java.lang.Byte
- java.lang.Short

- java.lang.Integer
- java.lang.Long
- java.lang.Float
- java.lang.Double
- java.lang.String
- java.math.BigDecimal
- java.math.BigInteger
- java.util.Date
- javax.management.ObjectName
- javax.management.openmbean.CompositeData
- javax.management.openmbean.TabularData

The Interface

The example below shows the definition of an interface for an MBean that returns some statistics in a tabular format and supports the reset of its statistics. It is not compulsory to declare the `reset` method. Declare it only if the MBean can or is allowed to reset its statistics.When an administrative request is made to reset all JMX statistics the JMX infrastructure inspects the MBean definition and if it finds the `reset` operation it invokes it.

```
import javax.management.openmbean.OpenDataException;
import javax.management.openmbean.TabularData;

public interface MyStatsMBean {
  /**
   * MBean attribute holding the statistics.
   */
  TabularData getStats() throws OpenDataException;
  /**
   * This method is invoked by the JMX infrastructure when
   * a request is made to reset the JMX statistics.
   */
  void reset();
}
```

Example 2.1 A custom MBean interface

End the name of the interface in StatsMBean

It is important for all MBeans that export statistics to have an interface class name that ends in `StatsMBean` .

The Implementation

Create an implementation of this interface that inherits from `curam.util.jmx.CuramMBeanAbstract` . To make it easier further on, derive the name of this class from the name of the implemented MBean

by removing the `MBean` suffix. The super class provides the MBean with access to the application configuration via the execution context and it facilitates the handling of changes in application configuration data that might be of interest to the MBean.

```
package com.mytest;

import java.util.logging.Level;
import java.util.logging.Logger;

import javax.management.openmbean.
                        CompositeDataSupport;
import javax.management.openmbean.
                        CompositeType;
import javax.management.openmbean.
                        OpenDataException;
import javax.management.openmbean.OpenType;
import javax.management.openmbean.SimpleType;
import javax.management.openmbean.TabularData;
import javax.management.openmbean.
                        TabularDataSupport;
import javax.management.openmbean.TabularType;

import curam.util.jmx.CuramMBeanAbstract;

public class MyStats extends CuramMBeanAbstract
      implements MyStatsMBean {
private static final Logger log = Logger
     .getLogger(MyStats.class.getName());

private static final OpenType[] kItemTypes
                         = new OpenType[] {
  SimpleType.STRING,
  SimpleType.LONG,
};

private static final String[] kItemNames
                 = new String[] {
   "Item",
   "Execution time(ms)"};

private static final String[] kItemDescriptions
                = new String[] {
   "The name of the item",
   "The execution time in milliseconds"};

private static TabularType stTabularType;

private static CompositeType stRowType;

private static MyStats instance;

static {
  try {
    stRowType = new CompositeType(
        "MyStatsType", "My statistics",
         kItemNames, kItemDescriptions, kItemTypes);
    stTabularType = new TabularType(
        "MyStats", "My statistics",
        stRowType, new String[] { kItemNames[0]});
  } catch (Exception e) {
    log.log(Level.SEVERE,
         "Failed to create the open types.", e);
  }
}

public MyStats() {
  super();
  instance = this;
}
```

```
/* (non-Javadoc)
 * @see com.mytest.MyStatsMBean#getStats()
 */
public TabularData getStats()
      throws OpenDataException {
  if (stRowType == null
          || stTabularType == null) {
    return null;
  }
  TabularDataSupport sup = new TabularDataSupport(
          stTabularType);

  // sample stats
  Object[] values = new Object[2];
  // ...
  // get the values
  // ...
  CompositeDataSupport cd =
                        new CompositeDataSupport(
    stRowType, kItemNames, values);
  sup.put(cd);
  return sup;
}

/**
 * This method is invoked from instrumented code
 * to update the statistics for
 * <code>item</code>.
 * @param item the item to update statistics for
 * @param executionTime the execution time
 */
public static void updateStats(
          String item, long executionTime) {
  if(instance != null) {
    instance._updateStats(item, executionTime);
  }
}

/**
 * This method is invoked by the JMX infrastructure when
 * a request is made to reset the JMX statistics.
 * @see com.mytest.MyStatsMBean#reset()
 */
public void reset() {
 // ...
 // reset the statistics
 //...
}

private void _updateStats(String item, long executionTime) {
  // ...
  // update the items average execution time
  // ...
}
}
```

Example 2.2 A custom MBean implementation

More complex MBeans that require dynamic configuration parameters or support per user data collection can override or utilize the provided protected methods in `curam.util.jmx.CuramMBeanAbstract`.

Using curam.util.jmx.NumericalCounterStatisticsAggregator

The following example shows how to use `curam.util.jmx.NumericalCounterStatisticsAggregator` and `curam.util.jmx.NumericalCounterStatistics` to calculate

and make available various arithmetic values for a numerical counter (average, minimum, maximum and standard deviation).

```
import curam.util.jmx.NumericCounterStatisticsAggregator;
...
    /** Elapsed time statistics. */
    private NumericCounterStatisticsAggregator
                                elapsedTimeStats;

    /** Error counter. */
    private AtomicLong errors;

    /** Constructor. */
    MyClass() {
      super();
      errors = new AtomicLong(0);
      elapsedTimeStats =
           new NumericCounterStatisticsAggregator();
    }

    /**
     * Get the number of invocations.
     *
     * @return the number of invocations.
     */
    long getInvocations() {
      return this.elapsedTimeStats
                    .getNumberOfSamples();
    }

    /**
     * Get elapsed time statistics.
     *
     * @return elapsed time statistics.
     */
    NumericCounterStatistics getElapsedTimeStats() {
      return elapsedTimeStats.getAll();
    }

    /**
     * Get error counter.
     *
     * @return error counter.
     */
    long getErrors() {
      return errors.get();
    }

 /**
  * Add a statistics sample.
  *
  * @param elapsedTime the elapsed time.
  * @param error true if invocation ended in error.
  */
 void addStats(long elapsedTime, boolean error) {
  boolean reset = this.elapsedTimeStats
                              .add(elapsedTime);
  if(reset) {
   // Long.MAX_VALUE overflow
   errors.set(0);
  } else if(error){
   if(errors.incrementAndGet() < 0) {
    // Long.MAX_VALUE overflow
    this.elapsedTimeStats.reset();
   }
  }
 }
...
```

Example 2.3 Using NumericalCounterStatisticsAggregator and NumericCounterStatistics

2.2.2 Updating the Configuration of Cúram JMX

The next step is to add the new MBean to the list of MBeans to be instantiated by the JMX infrastructure. Depending on where the MBean is located (Web or Enterprise Java Beans (EJB) container) modify the corresponding application property:

- `curam.jmx.configured_mbeans_ejb` – for MBeans residing in the EJB container
- `curam.jmx.configured_mbeans_web` – for MBeans residing in the Web container

See Cúram JMX Configuration Guide for more details.

2.2.3 Instrumenting Application Code

The application code needs to be instrumented to push data to the custom MBean. In order to minimize overhead check that JMX monitoring is turned on before pushing statistics to the MBean.

```
public void instrumentedMethod() {
 long startTime = System.currentTimeMillis();
 ...
 // do processing
 ...
 // check that JMX monitoring is enabled before
 // updating the MBean
 if(CuramJMXUtil.isJmxMonitoringEnabled()) {
  MyStats.updateStats("item",
       System.currentTimeMillis() - startTime);
 }
}
```

Example 2.4 Pushing elapsed time statistics to the custom MBean

Another possible instrumentation is to add execution statistics to the existing JMX services such as transaction tracing and in-flight transaction data.

```
public Result instrumentedMethod(String param) {
 try {
  return CuramJMXUtil.runAndRecord(new Callable<Result>(){
        public Result call() throws Exception {
               return myMethod(param);
        }}, "myMethod",
        TransactionInfo.getProgramUser());
 } catch (CuramJMXUtil.CallableException e) {
         throw new AppRuntimeException(e.getCause());
 }
}
```

Example 2.5 Pushing execution statistics to existing JMX services

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

1623-14, Shimotsuruma, Yamato-shi

Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typograph-

ical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources.

IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products

should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This publication documents intended programming interfaces that allow the customer to write programs to obtain the services of IBM Cúram Social Pogram Management.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/us/en/copytrade.shtml .

Java and all Java-based trademarks and logos are registered trademarks of Oracle and/or its affiliates.

Other names may be trademarks of their respective owners. Other company, product, and service names may be trademarks or service marks of others.