SAP NetWeaver Scheduler for Java

Test Catalogue

SAP
JAVA-JXBP 7.1
Version 1.1
Java External Interface for Background Processing

History		
Version	Date	Status (Comments)
1.0.0	2009-10-23	First release
1.0.1	2010-01-20	Change in section 7.2, test “Has child jobs”, step 2 – the operation JXBP.getChildJobs(JobID) changed to JXBP.hasChildJobs(JobID)
1.0.2	2010-04-12	Added JXBP Web Service descriptor location. Changed test „*5.1 Getting Job Definitions*“ to require verification of the result concerning only the demo jobs.

Copyright

Contents

Symbols

Symbol	Meaning
⚠	Warning
💬	Example
💡	Tip
🧭	Recommendation
SYN	Syntax

1 Overview

This document describes the technical certification of an external Job Schedulers for the JAVA-JXBP 7.1 interface.
You should read the interface description (JAVAJXBP71DOC.pdf), which is part of the certification package, before reading this document.

1.1Certification Requirements

1.1.1 Environment

The requirements for the certification environment of the NW JXBP external interface are as follows:

- At least one valid SAP NetWeaver Developer User license. You can obtain a license at http://www.sap.com/community/survey/index.epx?SurveyID=1089
- SAP NetWeaver 7.1 Application Server Java installation. The **supported version is 7.1 Enhancement Package 1 SP 4 or higher**.

💡 The latest state of the requirements can be found in SAP Note 1396620.

1.1.2 Scheduling System

The following requirements should be met by external schedulers connected to the SAP system using JXBP operations:

- The external job scheduling system must be able to fulfill the steps in the following test catalogue in a manner, which means the results can be reviewed.
- The external job scheduling system must also display the results of the tests in an appropriate way in its user interface.
- The external scheduler must not use other function calls to the SAP system than JXBP functions.

2 Prerequisites

2.1 Deploy certification package

As a part of the certification package there is an enterprise application archive (SampleJobsApplication.sda), which has to be deployed on your server.
You can find the archive and the deployment process description in SAP Note 1396620.
The deployed archive contains the following test jobs:

- SleepJobDemo:
  The job sleeps some time as specified by the *sleepTime* parameter (in milliseconds) and exits afterwards;
- BigLogJobDemo:
  The job writes a specified amount of log records as specified by the *LogRecordCount* parameter and exits afterwards;
- CancelableJobDemo:
  The job sleeps 5 times for 1 minute. After each wake up, before to sleep again for a minute, the job checks whether it has cancelled. If no – the job sleeps again, otherwise - finishes with status Error.
- CheckChildJobsJobDemo:
  The job starts *childJobsNum* child jobs and waits them to finish.

2.2 Register External Scheduler

Configure the external scheduler using the Scheduler NetWeaver Administrator (NWA) plug-in. The steps for registering an external scheduler are described into JXBP interface description document (JAVAJXBP71DOC.pdf), section 5.1;

2.3 Create Web Service client

In order to consume the JXBP interface, the external schedulers are supposed to create their own web service client, which will be the actual integration between the external scheduler and the Java Scheduler JXBP. The web service descriptor is located at the following address:
http://<host>:<port>/scheduer~runtime~gate~web/JXBPWebService?wsdl, where the <host> and <port> should be replaced with their actual values.

3 Structure of the test plan

For the certification of an external job scheduling system it is required, that all the JXBP operations are used in the correct way.

The tests are logically grouped in the following chapters:

Chapter 4	Information methods
Chapter 5	Operation with job definitions – operations for browsing job definitions
Chapter 6	Job operations- operations for browsing, executing and monitoring of jobs
Chapter 7	Child job operations
Chapter 8	Job log operations – here are the operations for extracting a logs from jobs executions
Chapter 9	Vendor data operations – used for associating custom data to a job
Chapter 10	Events operations – used for subscribing, querying and removing scheduler events
Chapter 11	Error handling – a few with negative tests

Each test describes a particular test scenario. The tests consist of the following sections:

- Description – describes what is under testing
- Prerequisites – the initial steps, which have to be performed before the actual test execution
- Test execution – a list of steps, which have to be executed
- Test result – the successful passing test criterions

The external scheduler is suitable for certification if all the tests are passed successfully.

4 Information methods

4.1 Getting JXBP version

Description

Get the version of JXBP.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

Call the web service operation *JXBP.getVersion()*

Test Result

The version must be ***'1.1'***

5 Job Definition operations

5.1 Getting job definitions

Description

Retrieve a list of all job definitions deployed on AS Java

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Call the web service operation *JXBP.getJobDefinitions()*
2. Open NetWeaver Administrator and browse *Operation Management - > Jobs - > Java Scheduler* or just type in the web browser *http://<host>:<port>/nwa/scheduler*, replacing the <host> and <port> with their actual values. For short let's call that procedure "Open NetWeaver Java Scheduler plug-in". Click on *Job Definition* section and check the job definition records. You need to check the following properties: *name*, *application name*, *retention period*, *description* and *parameters.*

Test Result

The mentioned properties for all four test job definitions (came with the certification package), which are returned from the web service call must be equal to these found in NetWeaver Administrator Java Scheduler plug-in.

5.2 Search job definition by name

Description

Query Java Scheduler for particular job definition by its name.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Call the web service operation *JXBP.getJobDefinitionByName(String name)* specifying the parameter "*name"* equal to "*SleepJobDemo*"

2. “Open NetWeaver Java Scheduler plug-in”. Click on *Job Definition* section and search for the job definition of “SleepJobDemo”. You need to check the following properties: *name*, *application name*, *retention period*, *description* and *parameters.*

Test Result

1. The job definition must be found in the result from web service call and in NetWeaver Administrator Java Scheduler plug-in.
2. The mentioned properties of job definition returned from the web service call must be equal to these found in NetWeaver Administrator Java Scheduler plug-in.

5.3 Search job definition by ID

Description

Query Java Scheduler for particular job definition by its identifier.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Call the web service operation *JXBP.getJobDefinitionByName(String name)* specifying the parameter *name* equal to “*SleepJobDemo*”
2. Call the web service operation JXBP.*getJobDefinitionById(JobDefinitionID id)* providing the parameter *id* equal to the job definition id returned from step 1.
3. “Open NetWeaver Java Scheduler plug-in”. Click on *Job Definition* section and search for the job definition of “SleepJobDemo”. You need to check the following properties: *name*, *application name*, *retention period*, *description* and *parameters.*

Test Result

1. The job definition must be found.
2. The mentioned properties of job definition returned from the web service call must be equal to these found in NetWeaver Administrator Java Scheduler plug-in.

6 Job operations

6.1 Execute Job

Description

Submit job for immediately execution.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Call the web service operation *JXBP.getJobDefinitionByName(String name)* providing the parameter "*name"* equal to "SleepJobDemo" as described in section 5.2.
2. Call the web service operation *JXBP.executeJob(JobDefitionID, List of JobParameterWS, int retentionPeriod)* , specifying the parameters as follows:
    a. Job Definition ID - take it from the result of execution in step 1;
    b. Job Parameters - a list of JobParameterWS. Construct such a list consisting only from one parameter called "*sleepTime*". To construct that parameter use the result from step 1. Set the parameter's value to 500 as an integer value.
    c. Retention period (int) – pass value -1 in order to keep this execution for indefinite period of time.

💡 JobParametersWS is a client class that should be generated automatically when you create your Web Service Client.

3. Get the returned ID of the job and search for that job in NetWeaver Administrator Java Scheduler plug-in.

Test Result

The job must be found in NetWeaver Administrator Java Scheduler plug-in. Its parameter "*sleepTime*" must be equal to 500. The retention period must be infinite (-1).

6.2 Cancel job

Description

Cancel a job.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*CancelableJobDemo*" in which must be a similar to section 6.1. The job will sleep for 2 minutes. Keep the ID.
2. Check that job in NetWeaver Administrator Java Scheduler plug-in and make sure that its status is *Running*.
3. Call the web service operation *JXBP.cancelJob(JobID)* providing the ID from step 1.
4. Search for that job in NetWeaver Administrator Java Scheduler plug-in after a minute.

Test Result

The job must be found in NetWeaver Administrator Java Scheduler plug-in. Its status should be *Error*.

6.3 Remove job

Description

Removes the information about job execution from the SAP NetWeaver AS Java Scheduler repository.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*SleepJobDemo*" following the steps in section 6.1. Keep the ID.
2. Call the web service operation *JXBP.removeJob(JobID)* providing the ID from step 1.
3. Search for that job in NetWeaver Administrator Java Scheduler plug-in.

Test Result

The job must not be found in NetWeaver Administrator Java Scheduler plug-in.

6.4 Remove jobs

Description

Removes the information about jobs execution from the SAP NetWeaver AS Java Scheduler repository.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*SleepJobDemo*" following the steps in section 6.1 at least twice. Keep the IDs.
2. Call the web service operation *JXBP.removeJobs(JobI[]D)* providing the IDs from step 1 as array.
3. Search for this job in NetWeaver Administrator Java Scheduler plug-in.
4. Search for these jobs in NetWeaver Administrator Java Scheduler plug-in.

Test Result

The job must not be found in NetWeaver Administrator Java Scheduler plug-in.

6.5 Get job

Description

Querying SAP NetWeaver AS Java Scheduler for particular job by ID.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*SleepJobDemo*" following the steps in section 6.1 Keep the ID.
2. Call the web service operation *JXBP.getJob(JobID)* providing the ID from step 1.
3. Find this job in NetWeaver Administrator Java Scheduler plug-in and compare a few job properties between found in plug-in job and returned from step 2 one.

Test Result

The job properties from NetWeaver Administrator Java Scheduler plug-in must have the same values as these from calling of the web service.

6.6 Get job parameters

Descriptions

Getting actual parameters which a job is using during its execution.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*SleepJobDemo*" following the steps in section 6.1 Keep the ID.
2. Call the web service operation *JXBP.getJobParameters(JobID)* providing the ID from step 1.

Test Result

1. The result should contain 4 parameters with the following values:
   a. sleepTime – 500;
   b. optionalTestParam – direction:*out*, type:*long*, with no value;
   c. optionalProperties – direction:*out*, type:*long*, with no value;
   d. actualSleepTime - direction:*out*, type:*long*, value depends on the parameter *sleepTime* and must be greater than it.

6.7 Get jobs

Description

Querying SAP NetWeaver AS Java Scheduler for particular jobs by ID.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job “*SleepJobDemo*” following the steps in section 6.1 at least twice. Keep the IDs.
2. Call the web service operation *JXBP.getJobs(JobID[])* providing the IDs from step 1 as an array.
3. Find these jobs via NetWeaver Administrator Java Scheduler plug-in and compare them with jobs returned from step 2.

Test Result

The jobs from calling of the web service should be the same as these found in NetWeaver Administrator Java Scheduler plug-in

6.8 Get job status

Description

Querying SAP NetWeaver AS Java Scheduler for the status of particular job.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job “*SleepJobDemo*” following the steps in section 6.1. Keep the ID.
2. Call the web service operation JXBP.*getJobStatus(JobID)* providing the ID from step 1.
3. Find this job in NetWeaver Administrator Java Scheduler plug-in. Check the status.

Test Result

The status must be the same.

6.9 Get jobs by status

Description

Get a list of jobs with certain status.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Call the web service operation JXBP.getJobsByStatus(JobStatus, JobIteratorWS, fetch size) providing *UNKNOWN* as status , *null* as JobIteratorWS for the first time invocation and *100* for fetch size
2. Go to NetWeaver Administrator Java Scheduler plug-in and filter all the jobs with UNKOWN status.

Test Result

The result should be the same – the *UNKNOWN* jobs count should be the same.

7 Child job operations

7.1 Get child jobs

Description

Return child jobs for certain parent job.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute job "*CheckChildJobsJobDemo*" which is similar to the steps in section 6.1 and providing 3 as parameter "*childJobsNum*". Keep the ID.
2. Call the web service operation *JXBP.getChildJobs(JobID)* providing the ID from step 1.
3. Check the parent job log with NetWeaver Administrator Java Scheduler plug-in. There must be a message about starting of 3 child jobs.
4. Compare the returned from step 2 child jobs with these in NetWeaver Administrator Java Scheduler plug-in.

Test Result

1. In the parent job log, there must be a message about starting of 3 child jobs.
2. For each child job, check the ID of the parent job. It must be equal to the ID in execution step 1.

7.2 Has child jobs

Description

Check whether a certain job has a child job(s).

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

1. Execute "*CheckChildJobsJobBeanDemo"* which is similar to the steps in section 6.1 and providing 3 as parameter "*childJobsNum*". Keep the ID.
2. Call the web service operation *JXBP.hasChildJobs(JobID)* providing the ID from step 1.
3. Check the parent job log with NetWeaver Administrator Java Scheduler plug-in.

Test Result

1. There must be a message about starting of 3 child jobs. The result of the operation must be ***"true"***.
2. Repeat that test but for job name "*SleepJobDemo*", the result must be ***"false"****.*

7.3 Have child jobs

Description

Checking are there child jobs for certain parent jobs.

Prerequisites

1. A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.
2. Execute job "*CheckChildJobsJobDemo*", specifying 3 child jobs to be run (as a parameter). Keep the ID of the parent job.
3. Execute a *"SleepJobDemo"* job according to the steps in section 6.1 and keep the ID.

Test Execution

1. Call the web service operation *JXBP.haveChildJobs(JobID[]).*
2. Specify the parameters:
   a. Array of Job ID–the IDs of the jobs, which you have kept in prerequisite 2-3 above.

Test Result

The result must be an array of 2 elements– *[true], [false]*

8 Job Log Operations

8.1 Get job log

Description

Obtain a log messages for particular job.

Prerequisites

1. A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface
2. Execute job *"BigLogJobDemo"* with a parameter *LogRecordCount* = 30. Keep the ID.

Test Execution

1. Call the web service operation *JXBP.getJobLog(JobID, LogIteratorWS, fetch size)*
Specify the parameters:
    a. Job ID– the ID of the job, which you have kept in prerequisite 2 above;
    b. LogIteratorWS – provide *null* for the first time invocation;
    c. Fetch size – 10.
2. Open NetWeaver Administrator Java Scheduler plug-in and check the logs for the same job.

Test Result

1. The result from the web service call should contain 10 log messages and should return an iterator for obtaining more logs.
2. Call the web service operation again – providing the same ID, returned log iterator and the value 20 as fetch size. The result should contain 20 messages. The returned messages should be the same as the message for that job in NetWeaver Administrator Java Scheduler plug-in.

8.2 Remove job log

Description

Removing log messages for particular job

Prerequisites

1. A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface
2. Execute "*BigLogJobDemo"* job with a parameter *LogRecordCount* = 20. Keep the ID.

Test Execution

1. Call the web service operation JXBP.removeJobLog(JobID) providing the job ID from prerequisite step 2.
2. Open NetWeaver Administrator Java Scheduler plug-in and try to find the logs for this job.

Test Result

1. There must not be any log messages for this job.
2. An additional test by invoking JXBP.getJobLog() for the created job, must return empty result.

9 Vendor data operations

9.1 Set Vendor Data

Description

Association between particular job and some custom data.

Prerequisites

1. A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface
2. Execute "*SleepJobDemo"* job following the steps in section 6.1. Keep the ID.

Test Execution

1. Call the web service operation setVendorData(JobID id, String vendorData) specifying the parameters:
    a. Job ID– the ID of the job, which you kept in prerequisite 2
    b. Vendor data – pass the string "*some vendor data*"
2. Call web service operation getVendorData(JobID id) for the same job.

Test Result

The result should be a string equal to "*some vendor data*".

9.2 Get Vendor Data

Description

Querying the vendor data for particular job.

Prerequisites

1. A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface
2. Execute "*SleepJobDemo"* job following the steps in section 6.1. Keep the ID.

Test Execution

Call the web service operation *JXBP.getVendorData(JobID)* passing the ID of the job, which has kept in test case 9.1 Set Vendor Data.

Test Result

The result should be the same - "*some vendor data*".

10 Event operations

10.1 Get JXBP Runtime Events

Description

Getting all the runtime events, used by the Scheduler.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface

Test Execution

Call the web service operation *JXBP.getJXBPRuntimeEventTypes*.

Test Result

The result must be array of strings:

- `com.sap.scheduler.runtime.JobStarting,`
- `com.sap.scheduler.runtime.JobStarted,`
- `com.sap.scheduler.runtime.JobFinished,`
- `com.sap.scheduler.runtime.JobCancelled,`
- `com.sap.scheduler.runtime.JobDefinitionDeployed,`
- `com.sap.scheduler.runtime.JobDefinitionUndeployed,`
- `com.sap.scheduler.runtime.JobDefinitionDeployed1,`
- `com.sap.scheduler.runtime.JobDefinitionUndeployed1`

10.2 Get Unhandled Events & Set Filter

Description

Getting all events, which are not handled yet by the connected External Scheduler. Set filter for events that the external scheduler is interested in.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface.

Test Execution

Call the web service operation *JXBP.getUnhandledEvents*.

Test Result

1. The result must be empty, because the external scheduler is not subscribed for any event.
2. If you apply an additional test like:
   a. Subscribe for an event job finished by invoking *JXBP,setFilter()* with parameter `com.sap.scheduler.runtime.JobFinished.`
   b. Execute a job according to section 6.1
   c. Call the web service operation *JXBP.getUnhandledEvents*.

   The result must contain:

   - Type - `com.sap.scheduler.runtime.JobFinished;`
   - Raised date – near to the execution time of job;

    - Parameter – the ID of the executed Job (in hex format). Check it with the ID for that job in NWA.

Note: Since you have subscribed for events and if there are other jobs in your system that might be run meantime, the result may contains information about more than one finished job. In that case first take the ID from NWA and then search for it in the result.

11 Error Handling

Most JXBP methods throw an exception which indicates an infrastructure problem executing the request. From client side of view, there are particular exception classes created for each method of the interface. For example method `getJobDefinitionByName` has a `GetJobDefinitionByNameFault` class, which wraps any exception that can be thrown by the server side and has details like the root exception.
The root exception (fault info) can be one of the following:

- JobIllegalStateException – when an operation for a job is performing and that job is not in required state (status)
- NoSuchJobDefinitionException – when there is no corresponding job definition
- NoSuchJobException – when there is no corresponding job
- ParameterValidationException – when a parameter is incorrect
- JXBPException – when general infrastructure problems occur

The external scheduler must be able to correctly deal with all these error situations.
Most of these exceptions can easily be provoked by passing non-existing ids, supplying wrong parameters or invoking a cancel operation on a job which is already completed.

11.1 Searching for job which does not exist.

Description

Simulate negative scenario, querying the NW Scheduler for job that does not exist.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface

Test Execution

Call the web service operation *JXBP.getJob*(JobID id) providing random JobID.

Test Result

The result should be *null.*

11.2 Remove non-existing job

Description

Simulate negative scenario, trying to remove a job which does not exist in the NW Scheduler.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface

Test Execution

Call the web service operation *JXBP.removeJob*(JobID id) providing random JobID.

Test Result

The result should be an exception with details that points to NoSuchJobException.

11.3 Execute job for non-existing job definition

Description

Simulate negative scenario, trying to execute job for job definition which does not exist in the NW Scheduler.

Prerequisites

A connection to SAP NetWeaver AS Java Scheduler has to be established through JXBP interface

Test Execution

Call the web service operation *JXBP.executeJob(JobDefitionID, List of JobParameterWS, int retentionPeriod)*, (JobID id) providing random JobDefinitionID. The rest of the parameters do not make sense here.

Test Result

The result should be an exception with details that points to NoSuchJobDefinitionException.