Leading Open Source Middleware

JOnAS 5 Commands Reference Guide

This document describes all the commands that can be used with JOnAS

JOnAS Team (François FORNACIARI)

- March 2009 -

Table of Contents

Chapter 1. jonas command

This command provides the capability to start, stop, or administrate JOnAS servers.

The following two scripts can be reviewed and possibly modified for assistance with problems or for obtaining additional information:

- **jonas** for UNIX systems
- **jonas.bat** for WINDOWS systems

There are five different sub-commands, that depend on the first mandatory argument: start, stop, admin, version, check

1.1. jonas start

Command allowing to start a JOnAS server.

1.1.1. Options

`jonas start` [-standby] [-fg] [-bg] [-win] [-tui] [-gui] [-dev] [-clean] [-n name] [-target server] [-Ddomain.name=domain]

-standby	Start a minimal JOnAS server with only mandatory services : *registry*, *jmx*. If the *discovery* service is configured in the JOnAS service list, it will bo started too. When the server startup is done, JOnAS should be in the STANDBY state. In this state, the server can either be stopped (see *jonas stop* command) or be fully started by executing the *jonas start* command to reach the RUNNING state.
-bg	Start the server in background mode
-win	Start the server in a new window
-tui	Start the Apache Felix TUI (force foreground mode)
-gui	Start the Apache Felix GUI
-dev	This option allows to start a JOnAS server by using bundles present in the default maven repository instead of bundles under *$JONAS_ROOT/lib/bundles*. A specific maven repository location can be given by setting the *m2.repository* property
-clean	Clean the Apache Felix cache before starting a JOnAS server
-n name	Set the server name. It must be unique in the domain. By default, the server name is *jonas*
-target server	This option allows to start another server or cluster (group of servers) in the domain. This action is to be executed in the environment of a running master server who's name is given by the -n option. In order to start a target server or cluster, the following conditions have to be met: • the target must belong to the domain (has to be defined in the domain's map)

- a cluster daemon has to be running on the target's host and it has to be aware of the target (see cluster daemon configuration)

-Ddomain.name=domain — Set the name of the management domain to which the server belongs

1.1.2. Description

Start a JOnAS server.

The process can be run in the foreground, in the background, or in a new window. If the background option is chosen (default option), control is given back to the caller only when the server is ready.

The gui and tui options allow to interact with the OSGi framework.

The name of the management domain to which the server belongs is given by the *domain.name* propery, or by the server name if this property is not defined.

1.2. jonas stop

Command allowing to stop a JOnAS server.

1.2.1. Options

`jonas stop` [-standby] [-n name] [-target server] [-Ddomain.name=domain]

-standby — Allows to stop all services except the mandatory ones (*registry* and *jmx*). When the server stop is done, JOnAS should be in the STANDBY state and the JVM is still running. In this state, the server can either be started (see *jonas start* command) or be fully stopped by executing the *jonas stop* command to stop all services and the JVM.

-n name — Set the name of the server to stop. By default, the server name is *jonas*

-target server — This option allows to stop another server or cluster (group of servers) in the domain.

This action is to be executed in the environment of a running master server who's name is given by the -n option. In order to start a target server or cluster, the following conditions have to be met:

- the target must belong to the domain (has to be defined in the domain's map)
- a cluster daemon has to be running on the target's host and it has to be aware of the target (see cluster daemon configuration)

-Ddomain.name=domain — Set the name of the management domain to which the server belongs

1.2.2. Description

Stop a running or a standby JOnAS server by stopping in a first time all services and then the JVM.

The name of the management domain to which the server belongs is given by the *domain.name* propery, or by the server name if this property is not defined.

1.3. jonas admin

Command allowing to administer a JOnAS server.

1.3.1. Synopsis

`jonas admin` [-win] [-n name] [-registry] [-protocol] [-username] [-password] [admin options...]

1.3.2. Description

Administer a JOnAS server.

The connection to the server is established through JMX. The default JMX service URL is build from the information contained in the local *$JONAS_BASE/conf/carol.properties* file. You can customize the JMX service URL by specifying the server name, the registry URL and the protocol name.

Set the username and the password if the connection to the remote JOnAS server requires an authentication.

Used without any administration option, this command will prompt the user for an administration command (interactive mode). Each administration command exists in a non-interactive mode, to be used in shell or bat scripts for example.

1.3.3. Options

Options can be divided in two categories, the connection options and the administration options.

1.3.3.1. Connection options

-win	Administer the server in a new window
-n name	Set the name of the server to administer. By default, the server name is *jonas*
-username	Set the username when authentication is required
-password	Set the password when authentication is required
-registry	Set the registry URL
-protocol	Set the protocol name (jrmp, iiop or irmi)

1.3.3.2. Administration options

-?	Prints the help message
-a filepath	Deploy an application from a given filepath on the current server, or on another target in the domain if the current server is a master. Note that the filepath must be an absolute path The application can be one of the following : • a standard **.jar** file. This will lead to the creation of a new EJB container. • a standard **.war** file containing a WEB module. • a standard **.ear** file containing a complete J2EE application. • a standard **.rar** file containing a RAR module.
-r filepath	Undeploy a previously deployed application from the current server or from the specified target if the current server is a master
-gc	Run the garbage collector on the current JOnAS server

-passivate	Passivate all entity bean instances. This affects only instances outside transaction
-e	List the properties of the current JOnAS server
-j	List the registered JNDI names, as seen by the current JOnAS server
-l	List the beans currently loaded by the current JOnAS server
-synch	Synchronize the entity bean instances on the current JOnAS server. Note that this affects only the instances that are not involved in a transaction
-debug topic	Set the logging level for the given topic to DEBUG
-tt timeout	Change the default timeout for transactions. Timeout is in seconds
-ping [-timeout seconds]	Wait until the JOnAS server is available

Each jonas admin option has its equivalent in the interactive mode. To enter interactive mode and access the following list of subcommands, type jonas admin with only connection options (without administration options). To exit from interactive mode, use the exit command.

interactive command	options
addbeans	**-a filepath**
env	**-e**
gc	**-gc**
help	**-?**
jndinames	**-j**
listbeans	**-l**
removebeans	**-r filepath**
sync	**-sync**
trace	**-debug topic**
ttimeout	**-tt timeout**
quit	exit interactive mode

1.4. jonas version

Command allowing to know the version of the JOnAS server.

1.4.1. Synopsis

```
jonas version
```

1.4.2. Description

Print the current version of the JOnAS server.

1.5. jonas check

Command allowing to check that the JOnAS environment is correctly set .

1.5.1. Synopsis

`jonas check` [-help]

1.5.2. Description

Check that the JOnAS server is well installed. Below the result of the jonas check command :

```
Checking jonas.properties file...
- jonas.services : registry,jmx,jtm,db,security,resource,ejb2,ejb3,jaxws,web,ear,depmonitor

Checking JORAM configuration...

Checking port availability...

The JOnAS environment seems correct.
```

1.6. jcl

Command allowing to start or stop a cluster of JOnAS instances on the local machine.

1.6.1. Synopsis

`jcl` [status|start|stop|kill]

1.6.2. Description

The command is a wrapper of the `jonas` command enabling to control a local cluster previously built with the `newjc` command.

The names and paths of the cluster instances are built from a set of variables (JCL_NUMBER_OF_NODES, JCL_BASE_PREFIX, JCL_NODE_NAME_PREFIX, ...). The list of variable in an example is given below.

```
JCL_NUMBER_OF_NODES=2

JCL_CLUSTER_DAEMON_DIR=/home/jonas/njc51/cd
JCL_BASE_PREFIX=/home/jonas/njc51/jb

JCL_NODE_NAME_PREFIX=node
JCL_DOMAIN_NAME=sampleCluster2Domain

JCL_DB_DIR=/home/jonas/njc51/db
JCL_DB_NAME=db

JCL_MASTER_DIR=/home/jonas/njc51/master
JCL_MASTER_NAME=master

export JCL_MASTER_NAME JCL_MASTER_DIR JCL_DB_NAME JCL_DB_DIR JCL_DOMAIN_NAME
 JCL_NODE_NAME_PREFIX
export JCL_NUMBER_OF_NODES JCL_CLUSTER_DAEMON_DIR JCL_BASE_PREFIX
```

The control commands (status/start/stop/...) are applied for all the instances by default. A particular instance or a subset of instances can be specified with the `-n` argument.

Example for starting node1 and node2:

```
$JONAS_ROOT/bin/jcl.sh -n 1,2 start
```

Example for getting the status for node1, node2, node3 and node4:

```
$JONAS_ROOT/bin/jcl.sh -n 1-4 status
```

Chapter 2. jclient command

Start a client container.

2.1. Options

`jclient` [-cp classpath] [-security] [-jarClient] [-traceFile] [-carolFile] [-tmpDir] [-nowsgen] java-class [args]

-jarClient	Specify the client jar to use of the ear if many.
-traceFile	Specify the configuration file to use for the traces of this client instead of the default file
-carolFile	Specify the carol.properties file to use instead of the default carol.properties file of the client.jar
-tmpDir	Specify the temp directory where unpack the ear.
-cp classpath	Add an additional classpath before running the java program.
-nowsgen	Specify if the Container shouldn't use Automated WsGen.
-security	Set a security manager using the policy file in `$JONAS_BASE/conf/java.policy`. (Used for automatic stubs downloading)

2.2. Description

The jclient command allows the user to easily start a "heavy" java client that will be able to reach beans in remote JOnAS servers and start distributed transactions.

It is not the J2EE compliant way to run a java client which is to package the java client into a J2EE container client (refer to Client Packaging).

Chapter 3. GenIC command

Generates the container classes for EJBs.

3.1. Options

`GenIC` [options...] <InputFileName>

-d directory	Specifies the root directory of the class hierarchy. This option can be used to specify a destination directory for the generated files. If the **-d** option is not used, the package hierarchy of the target class is ignored and the generated files are placed in the current directory. If the *InputFile* is an ejb-jar file, the generated classes are added to the ejb-jar file, unless the **-noaddinjar** option is set.
-invokecmd	Invokes directly the java class corresponding to the java compiler. This is useful on Windows in the event of a *CreateProcess Exception* (this occurs when the command line is too long). In this case tools.jar must be visible in the CLASSPATH
-javac *options*	Specifies the java compiler name to use (javac by default).
-javacopts *options*	Specifies the options to pass to the java compiler.
-keepgenerated	Do not immediately delete generated files.
-noaddinjar	If the *InputFile* is an ejb-jar file, do not add the generated classes to the ejb-jar file.
-nocompil	Do not compile the generated source files via the java and rmi compilers.
-novalidation	Remove xml validation during parsing.
-protocols	Comma-separated list of protocols (jrmp, iiop, irmi) for which stubs should be generated. Default is jrmp
-rmiopts *options*	Specifies the options to pass to the rmi compiler.
-verbose	Displays additional information about command execution.
-nofastrmic	Disable the use of fastrmic for stubs/ties generation.

3.2. Example

GenIC -d ../../classes sb.xml	Generates container classes of all the Enterprise JavaBeans defined in the sb.xml file. Classes are generated in the ../../ classes directory adhering to the classes hierarchy.
GenIC sb.jar	Generates container classes for all the Enterprise JavaBeans defined in the sb.jar file and adds the generated classes to this ejb-jar file.

3.3. Description

The GenIC utility generates the container classes for JOnAS from the given Enterprise Java Beans.

The *InputFileName* is either the file name of an ejb-jar file or the file name of an XML deployment descriptor of beans.

The GenIC utility does the following :

1. generates the sources of the container classes for all the beans defined in the deployment descriptor,
2. compiles these classes via the java compiler,
3. generates stubs and skeletons for those remote objects via the rmi compiler, and
4. if the InputFile is an ejb-jar file, adds the generated classes in this ejb-jar file.

3.4. Environment

If *InputFile* is an XML deployment descriptor, the classpath must include the paths of the directories in which the Enterprise Bean's classes can be found, as well as the path of the directory specified by the -d option.

If *InputFile* is an ejb-jar file, the classpath must include the path of the directory specified by the -d option.

Chapter 4. JMS commands

4.1. JmsServer

Launch the JORAM server with the existent configuration.

4.1.1. Synopsis

`JmsServer`

4.1.2. Description

Launch the JORAM server with the existent configuration.

The JORAM server configuration files are listed below:

- `$JONAS_BASE/conf/a3servers.xml`
- `$JONAS_BASE/conf/joramAdmin.xml`
- `META-INF/ra.xml` in the `joram_ra_for_jonas.rar` archive (may be in the internal repositories or in the `deploy/` directory).

4.1.3. Example

The *JmsServer* command is typically run in the background:

JmsServer & on Unix

start JmsServer on Windows

4.2. joram_raconfig

Change the host and port parameters of a given JORAM server in the configuration files.

4.2.1. Options

`joram_raconfig` [-p port] [-h host] [-s serverId]

-p port Set the listening port of the JORAM server (defaults to 16010).

-h host Set the IP address of the JORAM server (defaults to localhost).

-s serverId Set the server id of the JORAM server (defaults to 0).

4.2.2. Description

The joram_raconfig tool aims to facilitate consistent updates (accross multiple files) for the host and port parameters of a given JORAM server ID.

JORAM relies on several configuration files: `a3servers.xml`, `joramAdmin.xml` and `ra.xml`. With joram_raconfig, these configuration files are updated all together and thus the consistency is ensured.

Modified files:

- `$JONAS_BASE/conf/a3servers.xml`
- `$JONAS_BASE/conf/joramAdmin.xml`
- `META-INF/ra.xml` (in the JORAM resource adapter) is updated.

Resource adapters files are looked up in the following places:

- `$JONAS_BASE/repositories/maven2-internal/org/objectweb/joram/joram_ra_for_jonas/{joram.version}/joram_ra_for_jonas-{joram.version}.rar`
- `$JONAS_BASE/deploy/joram_ra_for_jonas.rar`

4.2.3. Example

```
>$ joram_raconfig -h localhost -p 16012 -s 0
Target JORAM Resource Adapter: /home/ ... /joram/joram_ra_for_jonas/5.2.1a/
joram_ra_for_jonas-5.2.1a.rar
```

Chapter 5. Configuration commands

5.1. newjb command

Build a new JONAS_BASE directory.

5.1.1. Synopsis

```
newjb
```

5.1.2. Description

The newjb utility builds a new JONAS_BASE directory that allows the conformance tests to be launched. At the start, the user must choose:

- the protocol among jrmp, iiop, irmi
- the database
- the web container among tomcat, jetty

The tool generates the configuration automatically.

The $JONAS_BASE variable must be set before launching the tool; it specifies the path to the new directory that will be built.

The $HOME/jb.config/lib directory must be created before launching the tool. It can contain some specific user configuration (see below).

The tool relies on JOnAS 's ant tasks ($JONAS_ROOT/templates/newjb/build-jb.xml) and thus builds a configuration compatible with the JOnAS version. First, a JONAS_BASE with default values is built, and then the configuration files are modified with the values defined in the centralized configuration file of newjb (see below).

A default configuration file is provided in $JONAS_ROOT/templates/newjb/build-jb.properties. It contains the variable parameters used by the tool, such as port number and database properties.

A user configuration can be set in the $HOME/jb.config/conf/jonas-newjb.properties file. If this file is present, the parameters it contains will override the default parameters.

By default, only the HSQL database can be configured with this tool. For other databases, the specific drivers must be stored in the $HOME/jb.config/lib directory before the run and the properties must be set in the $HOME/jb.config/conf/jonas-newjb.properties file.

The default script ($JONAS_ROOT/build-jb.xml) and its configuration ($JONAS_ROOT/templates/newjb/build-jb.properties) can be used as an example for creating a configuration tool corresponding to user's specific requirements.

5.2. newjc command

Command that builds all the configurations (ie JONAS_BASE) of the JOnAS instances for the cluster (including also a JOnAS master to manage the domain, a JOnAS DB with an embedded HSQLDB server, and a cluster daemon). It also creates the configuration files needed for mod_jk.

5.2.1. Options

`newjc` [-auto]

-auto Use the default configuration (silent mode)

5.2.2. Description

The newjc utility builds all the configurations (ie JONAS_BASE) of the JOnAS instances for the cluster (including also a JOnAS master to manage the domain, a JOnAS db with an embedded HSQLDB server, and a cluster daemon). It also creates the configuration files needed for mod_jk.

At command start, the user must choose:

1. the cluster directory where the JOnAS configuration directories (ie JONAS_BASE) will be created,
2. the prefix for the new JOnAS configuration directories (*ex*: prefix jb generates jb1, jb2, ...),
3. the protocol among jrmp, iiop, irmi,
4. the database to configure for the `db` node,
5. the cluster architecture among bothWebEjb, diffWebEjb. The first means that all the nodes are configured with both web and ejb services whereas the second one will separate the two tiers,
6. the number of web instances,
7. the number of ejb instances.

The tool generates the configuration automatically but you can customize some of the characteristics of the cluster by modifying the following files, in `JONAS_ROOT/conf/newjc`, before executing the command:

- `build-jc.properties`: global configuration for the cluster,
- `build-master.properties`: specific configuration for the master node,
- `build-db.properties`: specific configuration for the db node.

If the **-auto** option is used, the `$JONAS_BASE` variable must be set before launching the tool; it specifies the prefix of the paths to the new directories that will be built and that will contain the JONAS_BASE of the cluster members.

You can add some specific user configuration in the `$HOME/jc.config/lib`. If the directory doesn't exits, it will be created during the execution of the command.

A user configuration can be set in the `$HOME/jc.config/conf/jonas-newjc.properties` file. If this file is present, the parameters it contains will override the default parameters defined in `JONAS_ROOT/conf/newjc/build-jc.properties`.

5.2.3. Review newjc output

newjc creates `tomcat_jk.conf` and `workers.properties` files under `<cluster-directory>/conf/jk`. The #Web section of `build-jc.properties` defines configuration parameters set by *newjc* in these files. The file `tomcat_jk.conf` contains apache directives supported in `httpd.conf` or `apache2.conf`. They are isolated in a separate file for modularity, then they must be included manually in `httpd.conf` or `apache2.conf`. Refer to workers HowTo [http://tomcat.apache.org/connectors-doc/generic_howto/workers.html] documentation on the Tomcat site.

newjc generates by default four JOnAS instances (four JOnAS bases) in directories jb1, jb2, jb3 and jb4 under the cluster directory. Each JOnAS instance has its own configuration files into the `conf` directory (as any other JOnAS base). It also generates configurations for cluster daemon (`cd` directory), for master node (`master` directory), and for db node (`db` directory).

By default, jb1 and jb2 are JOnAS web container instances and jb3 and jb4 are JOnAS EJB container instances.

newjc generates a script called **jcl4sc** (**jcl4sc.bat**) for controlling the cluster examples provided with JOnAS named, sampleCluster2 and sampleCluster3. The command takes in argument the parameter status, start, stop, halt or kill.

The pertinent files for the configuration of the cluster are described below:

carol.properties - jgroups-cmi.xml

The #Carol section of `build-jc.properties` defines configuration parameters set by *newjc* in the `carol.properties` and `jgroups-cmi.xml` files . This allows JNDI replication to support load balancing at the EJB level using the CMI protocol.

Note

The multicast address and port must be identically configured for all JOnAS / Tomcat instances.

jonas.properties

The #Services section of `build-jc.properties` defines configuration parameters set by *newjc* in the `jonas.properties` file.

tomcat6-server.xml

The #Web section of `build-jc.properties` defines some configuration parameters set by *newjc* in the `tomcat6-server.xml` file . In particular, a connector XML element for AJP1.3 protocol is defined, as well as a jvmRoute to support load balancing at the web level, via this connector.

A Cluster XML element was also added. It defines parameters for achieving Session replication at the web level with JOnAS.

Refer to the AJP Connector [http://tomcat.apache.org/tomcat-6.0-doc/config/ajp.html] and the Engine Container [http://tomcat.apache.org/tomcat-6.0-doc/config/engine.html] documentation.

Note

The jvmRoute name generated by newjc is the same as the name of the associated worker defined in worker.properties. This will ensure the Session affinity.

5.2.4. Tell Apache about mod_jk

To make apache aware of this new file, edit <prefix>/conf/httpd.conf.

Copy and paste the following line after the Dynamic Shared Object (DSO) Support section.

```
Include conf/jk/tomcat_jk.conf
```

Move the jk directory created by newjc under the APACHE structure:

```
bash> mv <cluster-directory>/conf/jk $APACHE_HOME/conf
```

Restart apache so that apache can load the jk_module:

```
bash> apachectl stop && apachectl start
```

Note

Some UNIX distributions may locate the module in the folder libexec instead of the folder modules.

5.3. RAConfig command

Generate a JOnAS specific resource adapter configuration file or a resource adapter file

5.3.1. Synopsis

`RAConfig` [options...] InputFileName [OutputFileName]

5.3.2. Description

The *RAConfig* utility provides the capability to create a JOnAS specific resource adapter configuration file (jonas-ra.xml) from a ra.xml file, or update a resource adapter file.

With this command it is possible to :

- extract a JOnAS specific resource adapter configuration file (*jonas-ra.xml)* from an *ra.xml* file (see option **-path)**
- create a new JOnAS specific resource adapter configuration file (*jonas-ra.xml)* from an *ra.xml f*ile (see option **-new)**
- create a resource adapater file (*.rar file)* from a dataSource.properties file (see option **-p)**
- update a resource adapter file with a *jonas-ra.xml* (see option **-u**)

The *InputFileName* is the file name of a the resource adapter.

The *OutputFileName* is the file name of an output resource adapter. This parameter is used with the **-p** (required) or **-u** (optional) options.

5.3.3. Options

-? or -help	Give a summary of the options.
-p or -property database_properties_file_name	Specifie the name of the <database>.properties file to process. The result of this processing will be a jonas-ra.xml file that will update the /META-INF/jonas-ra.xml file in the output rar.
-dm, -ds, -cp, -xa	These options are related to the option **-p** above. They specify the rarlink value to configure respectively DriverManager, DataSource, ConnectionPoolDataSource and XAConnection. If **-dm** is used, then the conversion will be a direct reflection of the values specified in the **-p** <database>.properties file. If any of the other values are specified, then the jonas-ra.xml created will reflect options from the **-p** <database>.properties file and the user must edit

	the file based on information from the database provider for the specified type of datasource. Each database provider may have different config properties that need to be set and will be included in the database provider's documentation.
-path output_directory	Specify the directory name to place the extracted jonas-ra.xml file.
-new	Don't extract jonas-ra.xml but create a new one. The default value is the system property for `java.io.tmpdir`.
-update or -u inputname	Specify the name of the XML file to process. This file will update the */META-INF/jonas-ra.xml* file in the rar. If this argument is used, it is the only argument executed.
-rarlink rarlink	Specify the jndi name of an rar file with which to link. This option can be used when this rar file will inherit all attributes associated with the specified jndi name. If this option is specified in the jonas-ra.xml file, it is the only file needed in the rar, and the ra.xml file will be processed from the rarlink file.
-securityfile security_file_to_process	Specify the security property file to process and add security information to jonas-ra.xml. This will map the specified principal name to the user on the EIS system. The specified file must be in the following form: *principal = user::password* . When used in conjunction with the **-encrypt** option, then the resulting information will be encrypted in jonas-ra.xml.
-encrypt	Used with **-securityfile** to encrypt the specified passwords.
-jndiname jndiname	This option is deprecated with 1.5 Resource Adapter. For 1.0 Resource Adapter, this specifies the JNDI name of the connection factory. This name corresponds to the name of the <jndi-name> element of the <jonas-resource> element in the JOnAS specific deployment descriptor. This name is used by the resource service for registering in JNDI the connection factory corresponding to this resource adapter.
-novalidation	Turn off the xml dtd/schema validation.

5.3.4. Examples

RAConfig -dm -p MySQL1 $JONAS_ROOT/repositories/ maven2-internal/org/ow2/jonas/ jonas-jca-jdbc-dm/<jonas-version>/ jonas-jca-jdbc-dm-<jonas-version> MySQL_dm	Generate a MySQL_dm.rar file linked to `jonas-jca-jdbc-dm-<jonas-version>.rar`. The jonas-ra.xml file inserted is created with values coming from the ra.xml file of the `jonas-jca-jdbc-dm-<jonas-version>.rar` and values from the MySQL1.properties file This rar file can then be deployed and will replace the configured MySQL1 datasource.
RAConfig -path . XX.rar	Extract the jonas-ra.xml of XX.rar in the working directory
RAConfig -u jonas-ra.xml MyRA.rar	Update/insert the jonas-ra.xml file into the MyRA.rar file