JMX Guide

JEUS 9

Open Source Software Notice

Document History

Product Version	Guide Version	Date	Remarks
JEUS 9	3.1.2	2025-03-24	-
JEUS 9	3.1.1	2024-12-24	-

Contents

1. Searching for MBean Information and Configuring the JMX Manager

This chapter describes how to search for information about JEUS MBeans and configure a JMX Manager environment.

1.1. Searching for MBean Information

MBean information can be searched for using a console tool (jeusadmin).

1.1.1. Using the Console Tool

MBean information can be retrieved by using a console tool named 'jeusadmin'.

Log on to the JEUS server using the jeusadmin tool and then run the **mbean-info** command. The information for the registered MBeans is displayed as follows:

```
[MASTER]domain1.adminServer>mbean-info -server adminServer
The object names of MBeans on the server [adminServer].
================================================================================
+------------------------------------------------------------------------------+
| JEUS:j2eeType=JeusService,jeusType=ThreadPool,JMXManager=adminServer,J2EEServ|
|er=adminServer,name=threadpool.System                                         |
| JEUS:j2eeType=JeusService,jeusType=JMSDestinationResource,JMXManager=adminSer|
|ver,J2EEServer=adminServer,JMSResource=adminServer_jms,name=ExamplesQueue     |
| JEUS:j2eeType=JeusService,jeusType=JEUSMPConnector,JMXManager=adminServer,J2E|
|EServer=adminServer,name=adminServer                                          |

    . . .

| JEUS:j2eeType=JeusService,jeusType=JNDIResourceService,JMXManager=adminServer|
|,J2EEServer=adminServer,name=adminServer                                      |
| JEUS:j2eeType=JTAResource,JMXManager=adminServer,J2EEServer=adminServer,name=|
|adminServer                                                                   |
| JEUS:j2eeType=JMSResource,JMXManager=adminServer,J2EEServer=adminServer,name=|
|adminServer_jms                                                               |
+------------------------------------------------------------------------------+
================================================================================
```

1.2. Configuring the JMX Manager

JEUS JMX provides JEUS components and real-time information to client applications in accordance with JMX Remote API specification 1.0. This section describes how to configure JMX Manager environment.

JMX Manager related settings are found under the server settings in the configuration file (domain.xml). To configure them, you must directly edit the XML file.

The configuration items are as follows.

- **Basic Configuration**

  ```
  <server>
  ...
      <jmx-manager>
          <use-rmi-connector>true</use-rmi-connector>
          <use-html-adaptor>true</use-html-adaptor>
          <html-adaptor-port>8098</html-adaptor-port>
          <snmp-adaptor>
              ...
          </snmp-adaptor>
          <mlet-url>file:///home/user/mlet/example.mlet</mlet-url>
      </jmx-manager>
  ...
  </server>
  ```

  Option to use an RMI Connector, HTML Adapter, HTML Adapter Port, and MLet URL information.

Item	Description
Use Html Adaptor Port	HTML Adaptor is JMX's protocol adapter that supports HTML. For more information about the HTML Adaptor, refer to Getting Started with JMX.
Use Rmi Connector	Option to use an RMI Connector. If enabled, the RMI Connector server instance is created when the server is started.The URL for connecting to the RMI Connector server is in the format "service:jmx:rmi://SERVER_ADDRESS:SERVER_BASE_PORT/jndi/SERVER_NAME".
Html Adaptor Port	The HTML adapter port is used to access the adapter through a web browser. If set to -1, the JMX Manager does not use the HTML protocol. Make sure not to use a port that is used by other services. To verify that the HTML adapter has been configured successfully, run the web browser and then access the server using the IP address of the server and the specified port number.(Refer to HTML Adaptor Access Page).
MLet URL	Sets the MLet URL to register with the MBean server. For the specified MLet URL to be applied, the server needs to be restarted. For more information about MLet, refer to MLet API Documentation.

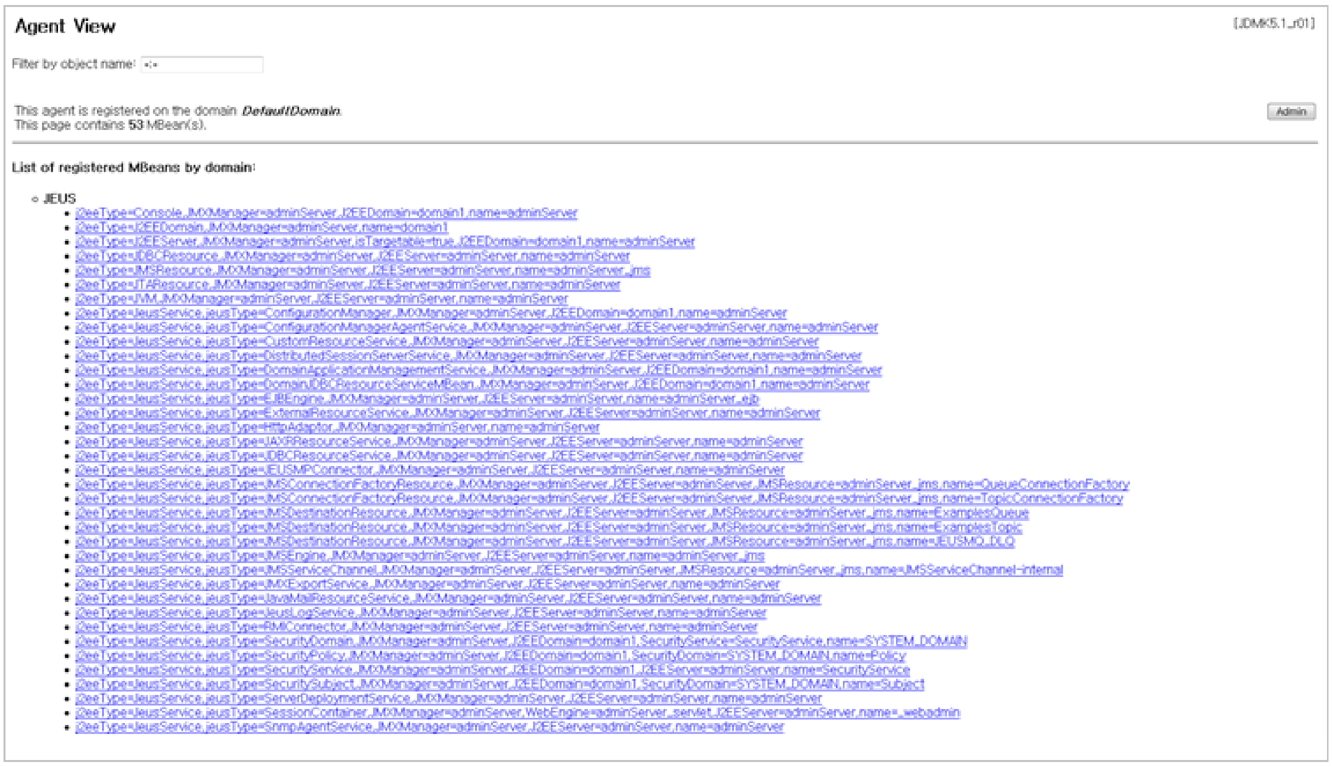
- **Snmp Adaptor**

  SNMP adapter is an SNMP protocol adapter provided by JMX. It can be set as a child item of the JMX manager setting.

```
<server>
...
    <jmx-manager>
        ...
        <snmp-adaptor>
            <snmp-adaptor-port>8099</snmp-adaptor-port>
            <snmp-version>3</snmp-version>
            <snmp-max-packet-size>256</snmp-max-packet-size>
            <trap-demon>
                <ip-address>127.0.0.1</ip-address>
                <port>9099</port>
            </trap-demon>
            <pooling>
                <min>1</min>
                <max>5</max>
                <period>30000</period>
            </pooling>
        </snmp-adaptor>
        ...
    </jmx-manager>
...
</server>
```

Item	Description
Snmp Adaptor Port	Listener port of the SNMP adapter.
Snmp Version	SNMP version. 1,2 or 3 can be used.
Snmp Max Packet Size	Maximum SNMP packet size. The minimum size is 256 bytes.
Snmp Security	Indicates whether to enable SNMP security. This option can only be specified in SNMP 3.
Trap Demon	Server to send trap messages to after an error occurs. Multiple addresses can use used. The address can be specified in the 111.111.111.1:8888 format.
Pooling	Configures properties of the thread pool that handles requests sent to the SNMP server. • Min: Minimum number of threads to be used by the tread pool. • Max: Maximum number of threads to be used by the tread pool. • Period: Sets the period for adjusting the pool size. For detailed information on SNMP and SNMP Adaptor settings, refer to "SNMP Agent Configuration" in JEUS SNMP Guide.

The following is the screen that shows the HTML adapter is running successfully.


Agent View

[JDMK5.1_r01]

Filter by object name: *:*

This agent is registered on the domain *DefaultDomain*.
This page contains **53** MBean(s).

Admin

List of registered MBeans by domain:

- JEUS
  - j2eeType=Console,JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
  - j2eeType=J2EEDomain,JMXManager=adminServer,name=domain1
  - j2eeType=J2EEServer,JMXManager=adminServer,isTargetable=true,J2EEDomain=domain1,name=adminServer
  - j2eeType=JDBCResource,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JMSResource,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer_jms
  - j2eeType=JTAResource,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JVM,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=ConfigurationManager,JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
  - j2eeType=JeusService,jeusType=ConfigurationManagerAgentService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=CustomResourceService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=DistributedSessionServerService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=DomainApplicationManagementService,JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
  - j2eeType=JeusService,jeusType=DomainJDBCResourceServiceMBean,JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
  - j2eeType=JeusService,jeusType=EJBEngine,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer_ejb
  - j2eeType=JeusService,jeusType=ExternalResourceService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=HttpAdaptor,JMXManager=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JAXRResourceService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JDBCResourceService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JEUSMPConnector,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JMSConnectionFactoryResource,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=QueueConnectionFactory
  - j2eeType=JeusService,jeusType=JMSConnectionFactoryResource,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=TopicConnectionFactory
  - j2eeType=JeusService,jeusType=JMSDestinationResource,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=ExamplesQueue
  - j2eeType=JeusService,jeusType=JMSDestinationResource,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=ExamplesTopic
  - j2eeType=JeusService,jeusType=JMSDestinationResource,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=JEUSMQ_DLQ
  - j2eeType=JeusService,jeusType=JMSEngine,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer_jms
  - j2eeType=JeusService,jeusType=JMSServiceChannel,JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,name=JMSServiceChannel-internal
  - j2eeType=JeusService,jeusType=JMXExportService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JavaMailResourceService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=JeusLogService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=RMIConnector,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=SecurityDomain,JMXManager=adminServer,J2EEDomain=domain1,SecurityService=SecurityService,name=SYSTEM_DOMAIN
  - j2eeType=JeusService,jeusType=SecurityPolicy,JMXManager=adminServer,J2EEDomain=domain1,SecurityDomain=SYSTEM_DOMAIN,name=Policy
  - j2eeType=JeusService,jeusType=SecurityService,JMXManager=adminServer,J2EEDomain=domain1,J2EEServer=adminServer,name=SecurityService
  - j2eeType=JeusService,jeusType=SecuritySubject,JMXManager=adminServer,J2EEDomain=domain1,SecurityDomain=SYSTEM_DOMAIN,name=Subject
  - j2eeType=JeusService,jeusType=ServerDeploymentService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
  - j2eeType=JeusService,jeusType=SessionContainer,JMXManager=adminServer,WebEngine=adminServer_servlet,J2EEServer=adminServer,name=_webadmin
  - j2eeType=JeusService,jeusType=SnmpAgentService,JMXManager=adminServer,J2EEServer=adminServer,name=adminServer

HTML Adaptor Access Page

2. JMX Application Development

This chapter describes how to develop client applications for accessing MBean provided in JEUS.

2.1. JMX Client Application

This section describes how to run a JMX client application.

1. The following two methods are used to connect to the JEUS MBean server.
    - Connect by looking up the information registered in JNDI (refer to Using JNDI).
    - Connect by using JMX Remote API (refer to Using JMX Remote API).
2. Access the MBean that provides the function that the user wants to use, and retrieve the property value, or execute a task.
    - To access an MBean, the user must know the ObjectName, which is a unique name that indicates the MBean. For the ObjectName format of the MBeans provided in JEUS, refer to MBean Object Names.
    - For security reasons, there are cases where a privilege check is performed when the property value or operations provided by MBeans are executed. To use such MBeans, a user must be authenticated as a user with appropriate privilege, and then access. For information about security configurations when connecting to JMX, refer to Security Setting.
3. Receive and process the result of the executed task or the retrieved property value.
4. When multiple tasks are executed, repeat steps two and three.

In order to fully understand what is explained in this chapter, basic knowledge of JMX Remote API 1.0 and Jakarta EE management specification is required. For details about JMX remote API, refer to J2EE JMX remote API 1.0 specification provided by Oracle, and the JMX remote API.

2.2. Connecting MBean Servers

This section explains how to connect to MBean servers.

2.2.1. Using JNDI

This section describes the JMX application which monitors JEUS by using JNDI.

Each server registers the javax.naming.Reference object, which contains information needed for connecting to JMX, in the JNDI. The name used for registration is in the format, "mgmt/rmbs/adminServer". Users can connect to the MBean server by using the reference registered in the JNDI.

The following is an example of a client using JNDI.

```java
package jmxclient;

import java.util.Set;
import java.util.Iterator;
import java.util.Hashtable;
import javax.management.MBeanServerConnection;
import javax.management.ObjectName;
import javax.management.remote.JMXConnector;
import javax.naming.Context;
import javax.naming.InitialContext;

/**
 * JMX Client which uses JNDI lookup.
 */
public class JMXClientUsingJndi {

    public static void main(String args[]) throws Exception {
        if(args.length < 4) {
            System.out.println("Required arguments: "
                + "hostname username password target-name");
            return;
        }

        // Step 1. Setting Environments
        String hostname = args[0];
        String username = args[1];
        String password = args[2];

        // targetName could be server name,
        // for example, "adminServer", "server1"
        String targetName = args[3];

        Hashtable env = new Hashtable();
        env.put(Context.INITIAL_CONTEXT_FACTORY, "jeus.jndi.JEUSContextFactory");
        env.put(Context.PROVIDER_URL, hostname);
        env.put(Context.SECURITY_PRINCIPAL, username);
        env.put(Context.SECURITY_CREDENTIALS, password);

        // Step 2. Getting MBeanServerConnection
        InitialContext ctx = new InitialContext(env);
        JMXConnector connector = (JMXConnector)ctx.lookup("mgmt/rmbs/" + targetName);
        MBeanServerConnection mbeanServer = connector.getMBeanServerConnection();

        // Step 3. Query
        ObjectName jeusScope = new ObjectName("JEUS:*");
        Set objectNames = mbeanServer.queryNames(jeusScope, null);

        // Step 4. Handling the Query Result
        for(Iterator i = objectNames.iterator(); i.hasNext();) {
            System.out.println("[MBean] " + i.next());
        }
    }
}
```

Write the above example and compile it. After connecting to the JEUS server, the list of MBeans applicable to "JEUS:*" will be displayed. In the example, there are four arguments. The first is the server hostname, the second is the JEUS user name, the third is the password, and the last is the server name.

```
$ java -classpath .:${JEUS_HOME}/lib/client/jclient.jar jmxclient.JMXClientUsingJndi 127.0.0.1:9736
jeus jeus adminServer

[2016.05.28 15:20:24][2] [t-1] [NET-0002] Beginning to listen to NonBlockingChannelAcceptor:
/127.0.0.1:9756.
[MBean] JEUS:j2eeType=JeusService,jeusType=ThreadPool,JMXManager=adminServer,
J2EEServer=adminServer,name=threadpool.System
[MBean] JEUS:j2eeType=JeusService,jeusType=JEUSMPConnector,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSDestinationResource,
JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,
name=ExamplesQueue
[MBean] JEUS:j2eeType=JeusService,jeusType=JeusLogService,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=ThreadPool_WEBC,JMXManager=adminServer,
WebEngine=adminServer_servlet,J2EEServer=adminServer,WebListener=http1,name=http1
[MBean] JEUS:j2eeType=JeusService,jeusType=SecurityDomain,JMXManager=adminServer,
J2EEDomain=domain1,SecurityService=SecurityService,name=SYSTEM_DOMAIN
[MBean] JEUS:j2eeType=JeusService,jeusType=DeploymentPlanManagementService,
JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSConnectionFactoryResource,
JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,
name=ConnectionFactory
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSEngine,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer_jms
[MBean] JEUS:j2eeType=JeusService,jeusType=ServerDeploymentService,
JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
...
```

1. jclient.jar is required to run the example program. In general, it is located under the JEUS_HOME/lib/client directory.
2. For detailed information on JNDI, refer to "JNDI Naming Server" in JEUS Server Guide. Note that if your JMX application runs in a servlet or EJB, you do not need to configure JNDI parameters.

2.2.2. Using JMX Remote API

This section describes how to use the JMX application, which monitors JEUS by using the JMX Remote API.

Specify the target to be connected through the JMX Service URL. The URL for connecting to the JEUS MBean server is in the following formats.

- service:jmx:jmxmp://0.0.0.0:9736/JeusMBeanServer
- service:jmx:jmxmp://0.0.0.0:9736/JEUSMP_<*adminServer*>

The second URL will be different depending on the name of the server that will access the adminServer.

The following is an example of the client that uses the JMX Remote API.

```
package jmxclient;

import java.util.Hashtable;
import java.util.Iterator;
import java.util.Set;

import javax.management.MBeanServerConnection;
import javax.management.ObjectName;
import javax.management.remote.JMXConnector;
import javax.management.remote.JMXConnectorFactory;
import javax.management.remote.JMXServiceURL;
import javax.naming.Context;

/**
 * JMX Client which uses JMX Remote API
 */
public class JMXClientUsingJmxUrl {
    private static final String URL_PATH = "/JeusMBeanServer";

    public static void main(String args[]) throws Exception {
        if(args.length < 4) {
            System.out.println("Required arguments: "
                + "hostname port username password");
            return;
        }

        // Step 1. Setting Environments
        String address = args[0];
        int port = Integer.parseInt(args[1]);
        String username = args[2];
        String password = args[3];

        Hashtable env = new Hashtable();
        env.put(JMXConnectorFactory.PROTOCOL_PROVIDER_PACKAGES, "jeus.management.remote.protocol");
        env.put(Context.SECURITY_PRINCIPAL, username);
        env.put(Context.SECURITY_CREDENTIALS, password);

        // Step 2. Getting MBeanServerConnection
        JMXServiceURL serviceURL = new JMXServiceURL("jmxmp", address, port, URL_PATH);
        JMXConnector connector = JMXConnectorFactory.connect(serviceURL, env);
        MBeanServerConnection mbeanServer = connector.getMBeanServerConnection();

        // Step 3. Query
        ObjectName jeusScope = new ObjectName("JEUS:*");
        Set objectNames = mbeanServer.queryNames(jeusScope, null);

        // Step 4. Handling the Query Result
        for(Iterator i = objectNames.iterator(); i.hasNext();) {
            System.out.println("[MBean] " + i.next());
        }
    }
}
```

Write the above example and compile it. After connecting to the JEUS server, the list of MBeans applicable to "JEUS:*" will be displayed. The arguments are the server address, port, user name, and password.

```
$ java -classpath .:${JEUS_HOME}/lib/client/jclient.jar jmxclient.JMXClientUsingJmxUrl 127.0.0.1 9736
jeus jeus

[2016.05.28 15:21:29][2] [t-1] [NET-0002] Beginning to listen to NonBlockingChannelAcceptor:
/127.0.0.1:9756.
[MBean] JEUS:j2eeType=JeusService,jeusType=ThreadPool,JMXManager=adminServer,
J2EEServer=adminServer,name=threadpool.System
[MBean] JEUS:j2eeType=JeusService,jeusType=JEUSMPConnector,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSDestinationResource,
JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,
name=ExamplesQueue
[MBean] JEUS:j2eeType=JeusService,jeusType=JeusLogService,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=ThreadPool_WEBC,JMXManager=adminServer,
WebEngine=adminServer_servlet,J2EEServer=adminServer,WebListener=http1,name=http1
[MBean] JEUS:j2eeType=JeusService,jeusType=SecurityDomain,JMXManager=adminServer,
J2EEDomain=domain1,SecurityService=SecurityService,name=SYSTEM_DOMAIN
[MBean] JEUS:j2eeType=JeusService,jeusType=DeploymentPlanManagementService,
JMXManager=adminServer,J2EEDomain=domain1,name=adminServer
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSConnectionFactoryResource,
JMXManager=adminServer,J2EEServer=adminServer,JMSResource=adminServer_jms,
name=ConnectionFactory
[MBean] JEUS:j2eeType=JeusService,jeusType=JMSEngine,JMXManager=adminServer,
J2EEServer=adminServer,name=adminServer_jms
[MBean] JEUS:j2eeType=JeusService,jeusType=ServerDeploymentService,
JMXManager=adminServer,J2EEServer=adminServer,name=adminServer
...
```

The jclient.jar file is required to run the example program. In general, the jclient.jar file is located under the JEUS_HOME/lib/client directory.

2.3. Security Setting

This section describes the security settings for the JEUS monitoring service. JEUS server performs security checks for a user who reads and performs operations on the properties of various MBeans registered with JEUS server using the JMX API. For more information about the permissions required to access each MBean, refer to JEUS API documents.

API documents can be found in the following location.

```
JEUS_HOME/docs/api
```

API documents provide permission names as well as ObjectNamePattern, attributes, operation information, and others required for using MBeans.

JMX applications provide username and password information when connecting to the server to create the MBeanServerConnection object. In general, the following code is used to create MBeanServerConnection object. The code shows that the username and password is included in the environment configuration that is sent to the hash table.

```
...

JMXServiceURL serviceURL =
        new JMXServiceURL("service:jmx:jmxmp://127.0.0.1:9736/adminServer");

Map<String, Object> env = new HashMap<String, Object>();
env.put(Context.SECURITY_PRINCIPAL, id);
env.put(Context.SECURITY_CREDENTIALS, password);
env.put(Context.INITIAL_CONTEXT_FACTORY, "jeus.jndi.JEUSContextFactory");
env.put("jmx.remote.x.request.timeout", "10");

JMXConnector connector = JMXConnectorFactory.connect(serviceURL, env);
MBeanServerConnection connection = connector.getMBeanServerConnection();

...
```

The user name, password, and permission settings used here are set using JEUS Security. For detailed information on how to set up JEUS Security, refer to "Configuring the Security System User Information" and "Configuring Security System Policies" in JEUS Security Guide.

2.4. MBean Object Names

An ObjectName is the default object name of an MBean object. In order to access the MBean server and interoperate with MBean, the name of the target MBean must be known. If the name is not fully known, then use the known parts to send a query to the MBean server, and then receive the result value to find the name.

An ObjectName for the MBean provided in JEUS has the following format.

```
<domain_name>: j2eeType=<j2eeType_value>, name=<name_value>,
    [<parent-j2eeType_value>], [jeusType = <jeusType_value>],
    [isTargetable = <isTargetable_value>],
    JMXManager = <JMXManager_value> [,*]
```

Alternative format is:

```
<domain_name>: *
```

An ObjectName must start with the <domain_name> and there is no defined sequence for each name-value pairs.

For example, both of the following ways will retrieve the object names of JEUSManager MBean.

```
JEUS:j2eeType=J2EEDomain,JMXManager=adminServer, *
```

```
JEUS:JMXManager=adminServer, j2eeType=J2EEDomain, *
```

The following describes each item:

- <domain_name>
  - JEUS domain name. Set to JEUS.
- j2eeType
  - The J2EE type of MBean, which is described in the J2EE management specifications.
  - Set with one of the following values:

AppClientModule	EJBModule	EntityBean
J2EEApplication	J2EEDomain	J2EEServer
JAXRResource	JCAConnectionFactory	JCAManagedConnectionFactory
JCAResource	JDBCDataSource	JDBCDriver
JDBCResource	JMSResource	JNDIResource
JTAResource	JVM	JavaMailResource
JeusService	MessageDrivenBean	ResourceAdaptor
ResourceAdapterModule	Servlet	StatefulSessionBean
StatelessSessionBean	URLResource	WebModule

- name
  - The name of MBean. There is a unique name for each MBean object. For example, the name of the JVM on which the "adminServer" server runs is "adminServer."
- parent-j2eeType
  - The J2EE type of the MBean’s parent. There is a defined hierarchy for MBeans. For example, the parent-j2ee type of "JDBCDriver" is "JDBCDataSource."
- jeusType
  - Mbeans' type defined in JEUS JMX. Only "JeusService" j2eeType can have several jeusTypes.
  - Set with one of the following values:

EJBEngine	JMSClientResource	JMSConnectionFactoryResource
JMSDestinationResource	JMSDurableSubscriberResource	JMSEngine
JMSPersistenceStoreManager	JMSServiceChannel	SecurityDomain
SecurityPolicy	SecurityService	SecuritySubject
SessionContainer	SessionContainerCentral	SessionContainerP2P
ThreadPool	ThreadPool_WEBC	WebEngine
WebListener	WebServices	

- isTargetable
  - A boolean type that should be set to true for the MBean on which user AP (e.g. EJB, servlet, and JSP) is deployed and is running as isTargetable.
- JMXManager
  - The name of the JMXManager that provides MBean service. In general, it is the name of the server that the JMXManager belongs to.

2.5. Spring JMX Support

JEUS 8 and later versions support a library that enables the JEUS MBean server to integrate Spring JMX. Based on the following path, a library is provided for each supported version of Spring.

```
JEUS_HOME/lib/shared/spring-support/
```

Spring JMX operates in JEUS even without the JEUS library. Because using the library requires packaging it for deployment to the application and making additional configurations, it is recommended for use only to integrate the JEUS MBean server with Spring JMX.

Before using the JEUS Spring JMX support library, place the library in the WEB-INF/lib directory, and set the configuration file for Spring as shown below so that the JEUS MBean server is available for use.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

  ...
  <bean id="mBeanServer" class="jeus.spring.jmx.JeusMBeanServerFactoryBean"/>

  <!-- this bean must not be lazily initialized if the exporting is to happen -->
<bean id="exporter" class="org.springframework.jmx.export.MBeanExporter" lazy-init="false">
    <property name="server" ref="mBeanServer"/>
    <property name="beans">
      <map>
        <entry key="bean:name=testBean1" value-ref="testBean"/>
      </map>
    </property>
  </bean>

  ...

</beans>
```

For details about the Spring framework and Spring JMX, see Spring Framework Official Documentation.