

JMX 안내서

JEUS 9

TMAXSOFT

저작권 공지

Copyright 2025. TmaxSoft Co., Ltd. All Rights Reserved.

회사 정보

(주)티맥스소프트

주소 : 경기도 성남시 분당구 황새울로258번길 29, 티맥스수내타워 8-9층

기술 서비스 센터: 1544-8629

홈페이지: <https://www.tmaxsoft.com>

제한된 권리

이 소프트웨어(JEUS®) 사용설명서와 프로그램은 저작권법과 국제 조약에 의해 보호됩니다. 사용설명서와 프로그램은 TmaxSoft Co., Ltd.와의 사용권 계약 하에서만 사용할 수 있으며, 사용설명서는 사용권 계약의 범위 내에서만 배포 또는 복제할 수 있습니다. 이 사용설명서의 전부 또는 일부를 TmaxSoft의 사전 서면 동의 없이 전자, 기계, 녹음 등의 수단으로 전송, 복제, 배포하거나 2차적 저작물을 작성할 수 없습니다.

이 소프트웨어 사용설명서와 프로그램의 사용권 계약은 어떠한 경우에도 사용설명서 및 프로그램과 관련된 지적 재산권(등록 여부를 불문)을 양도하는 것으로 해석되지 않으며, 브랜드나 로고, 상표 등을 사용할 권한을 부여하지 않습니다. 사용설명서는 오로지 정보 제공만을 목적으로 하며, 이로 인한 계약상의 직접적 또는 간접적 책임을 지지 않습니다. 또한 사용설명서 상의 내용이 법적 또는 상업적인 특정 조건을 만족시킬 것을 보장하지 않습니다. 사용설명서는 제품의 업그레이드나 수정에 따라 예고 없이 변경될 수 있으며, 내용상의 오류가 없음을 보장하지 않습니다.

상표 공지

JEUS®는 TmaxSoft Co., Ltd.의 등록 상표입니다.

Java, Solaris는 Oracle Corporation 및 그 자회사, 관계회사의 등록 상표입니다.

Microsoft, Windows, Windows NT는 Microsoft Corporation의 등록 상표 또는 상표입니다.

HP-UX는 Hewlett Packard Enterprise Company의 등록 상표입니다.

AIX는 International Business Machines Corporation의 등록 상표입니다.

UNIX는 X/Open Company, Ltd.의 등록 상표입니다.

Linux는 Linus Torvalds의 등록 상표입니다.

Noto는 Google Inc.의 상표입니다. Noto 글꼴은 오픈 소스입니다. 모든 Noto 글꼴은 SIL Open Font License, 버전 1.1에 따라 게시됩니다. (<https://www.google.com/get/noto/>)

기타 모든 제품들과 회사 이름은 각각 해당 소유주의 상호, 상표, 또는 등록 상표입니다.

본 사용설명서에 기재된 회사, 시스템, 제품 이름 등에 반드시 상표 표시(™, ®)를 하지는 않습니다.

오픈 소스 소프트웨어 공지

본 제품의 일부 파일 또는 모듈은 다음의 라이선스를 준수합니다.: APACHE2.0, CDDL1.0, EDL1.0, OPEN SYMPHONY SOFTWARE1.1, TRILEAD-SSH2, Bouncy Castle, BSD, MIT, SIL OPEN FONT1.1

관련 상세 정보는 제품의 다음 디렉터리에 기재된 사항을 참고하시기 바랍니다. :

`${INSTALL_PATH}/license/oss_licenses`

유지 보수

구분	지원항목	서비스 내용
제품지원	패치 & 업그레이드	무상 패치 서비스 제공
		메이저 버전 업그레이드 시 할인 혜택
		웹 지원을 통한 패치 내역 제공
기술 지원 - 기본 서비스	장애 지원	장애 발생 시 원인 분석 및 조치 Service Desk팀 → 기술팀 → R&D의 3단계 장애 분석 및 조치
	일상 지원(온라인 지원)	E-mail, 전화, 원격, 웹 사이트 등 온라인 자원을 통한 질의 응답 서비스
	고객 맞춤 지원(방문 지원)	고객의 요청으로 수행하는 방문 지원 서비스
기술 지원 - 옵션 서비스	예방 지원	정기 점검을 통한 시스템 운영현황 보고 및 장애 예방 ◦ 관리자 또는 운영자의 요구사항 수렴 ◦ 운영 현황(시스템, 엔진 운영) 보고서 제공 ◦ 필요 시 시스템 개선 권장 사항 보고
유지 보수 비용 및 기간	계약 시 별도 협의	계약 시 EOL/EOS 문서 제공

안내서 이력

제품 버전	안내서 버전	발행일	비고
JEUS 9	3.1.2	2025-01-03	-
JEUS 9	3.1.1	2024-09-27	-

목차

1. MBean 정보 조회 및 JMX Manager 환경설정	1
1.1. MBean 정보 조회	1
1.1.1. 콘솔 툴 사용	1
1.2. JMX Manager 환경설정	1
2. JMX 애플리케이션 개발	5
2.1. JMX 클라이언트 애플리케이션	5
2.2. MBean 서버 연결	5
2.2.1. JNDI 사용	5
2.2.2. JMX Remote API 사용	7
2.3. Security 설정	9
2.4. MBean Object Names	10
2.5. Spring JMX Support	12

1. MBean 정보 조회 및 JMX Manager 환경설정

본 장에서는 JEUS에서 사용하고 있는 MBean의 정보를 조회하는 방법 및 JMX Manager의 환경을 설정하는 방법을 설명한다.

1.1. MBean 정보 조회

콘솔 툴(jeusadmin)을 사용하여 JEUS에서 사용하고 있는 MBean의 정보를 조회할 수 있다.

1.1.1. 콘솔 툴 사용

콘솔 툴(jeusadmin)을 사용해서 MBean 정보를 조회할 수 있다.

jeusadmin을 통해 JEUS 서버에 접속한 후 **mbean-info** 명령어를 이용하면 다음과 같이 등록된 MBean 정보를 조회할 수 있다.

```
[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=JEUSMPCConnector,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. JMX Manager 환경설정

JMX Remote API 스펙 1.0을 따르는 클라이언트 애플리케이션에게 JEUS JMX는 JEUS의 구성과 실시간 정보를 제공한다. 본 절에서는 JMX Manager를 설정하는 방법에 대해서 설명한다.

JMX Manager 관련 설정은 설정 파일(domain.xml)의 서버 설정 하위에 존재한다. 설정을 위해서는 XML 파일을 직접 편집해야 한다.

다음은 설정 항목에 대한 설명이다.

• 기본 설정

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

RMI Connector를 사용할지 여부, HTML Adaptor 사용 여부, HTML Adaptor Port 및 MLet URL 정보를 설정한다.

항목	설명
Use Html Adaptor Port	HTML Adaptor는 HTML을 지원하는 JMX의 Protocol Adapter이다. HTML Adaptor에 대한 추가적인 사항은 JMX 시작하기 문서 의 설명을 참고한다.
Use Rmi Connector	RMI Connector를 사용할지 여부를 설정한다. 이 항목을 체크했을 경우 서버가 기동할 때 RMI Connector server 인스턴스를 생성한다. RMI Connector server에 접속하기 위한 URL은 "service:jmx:rmi://SERVER_ADDRESS:SERVER_BASE_PORT/jndi/SERVER_NAME"과 같은 형태를 갖는다.
Html Adaptor Port	HTML Adapter의 Listener Port로 웹 브라우저로 접속할 HTML Adapter의 Port를 지정한다. Port를 -1로 설정을 하면 JMX Manager가 HTML Protocol을 사용하지 않음을 의미한다. Port를 설정할 때 다른 서비스가 사용하는 Port를 사용하지 않도록 주의해서 설정한다. HTML Adaptor 설정이 정상적으로 이루어졌는지 확인하려면, 웹 브라우저를 실행한 후 서버 IP와 설정한 Port 값을 사용해 서버에 접속한다(HTML Adaptor 접속 화면 참고).
MLet URL	서버의 MBean 서버에 등록할 MLet URL을 설정한다. 설정한 MLet URL을 적용하기 위해서는 서버를 재시작해야 한다. MLet에 대한 자세한 설명은 MLet API 문서 의 설명을 참고한다.

• Snmp Adaptor

SNMP Adaptor는 JMX가 제공하는 SNMP Protocol Adapter이다. SNMP Adaptor 설정은 JMX Manager 설정의 하위 항목으로 존재한다.

```
<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>

```

항목	설명
Snmp Adaptor Port	SNMP 어댑터가 사용할 포트를 설정한다.
Snmp Version	사용할 SNMP 버전을 지정한다. 1,2 또는 3을 값으로 지정할 수 있다.
Snmp Max Packet Size	SNMP 패킷에 대한 최댓값을 설정한다. 최소 256 Byte부터 설정할 수 있다.
Snmp Security	보안 적용 여부를 설정한다. 보안은 SNMP 버전 3에서만 지정이 가능하다.
Trap Demon	장애 상황이 발생할 경우에 Trap 메시지를 보낼 서버를 설정한다. 여러 개를 설정할 수 있으며 설정한 모든 주소를 전송한다. 주소는 111.111.111.1:8888과 같은 형태로 지정한다.
Pooling	<p>SNMP 서버에서 요청을 처리하는데 사용하는 스레드 풀에 대한 설정이다.</p> <ul style="list-style-type: none"> • Min : 스레드 풀이 사용할 스레드의 최소 개수를 지정한다. • Max : 스레드 풀이 사용할 스레드의 최댓값을 지정한다. • Period : 풀 크기를 조정하는 주기를 설정한다. SNMP와 SNMP Adaptor 설정에 대한 자세한 설명은 JEUS SNMP 안내서의 "SNMP Agent 설정"을 참고한다.

다음은 HTML 어댑터가 정상적으로 동작하고 있는 경우의 접속 화면이다.

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

- [JeeType=Console, JMOManager=adminServer, JEEDomain=domain1, name=adminServer](#)
- [JeeType=JEEDomain, JMOManager=adminServer, name=domain1](#)
- [JeeType=JEEServer, JMOManager=adminServer, isTargetable=true, JEEDomain=domain1, name=adminServer](#)
- [JeeType=JDBCResource, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JMSResource, JMOManager=adminServer, JEEServer=adminServer, name=adminServer, jms](#)
- [JeeType=JARResource, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JVM, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=ConfigurationManager, JMOManager=adminServer, JEEDomain=domain1, name=adminServer](#)
- [JeeType=JeusService, jeusType=ConfigurationManagerAgentService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=CustomResourceService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=DistributedSessionServerService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=DomainApplicationManagementService, JMOManager=adminServer, JEEDomain=domain1, name=adminServer](#)
- [JeeType=JeusService, jeusType=DomainJDBCResourceServiceMBean, JMOManager=adminServer, JEEDomain=domain1, name=adminServer](#)
- [JeeType=JeusService, jeusType=JREngine, JMOManager=adminServer, JEEServer=adminServer, name=adminServer, eb](#)
- [JeeType=JeusService, jeusType=ExternalResourceService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=HttpAdaptor, JMOManager=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JAXRResourceService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JDBCResourceService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JMSMPConnector, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JMSConnectionFactoryResource, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=QueueConnectionFactory](#)
- [JeeType=JeusService, jeusType=JMSConnectionFactoryResource, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=TopicConnectionFactory](#)
- [JeeType=JeusService, jeusType=JMSDestinationResource, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=ExamplesQueue](#)
- [JeeType=JeusService, jeusType=JMSDestinationResource, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=ExamplesTopic](#)
- [JeeType=JeusService, jeusType=JMSDestinationResource, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=EUSMO, DQ](#)
- [JeeType=JeusService, jeusType=JMSEngine, JMOManager=adminServer, JEEServer=adminServer, name=adminServer, jms](#)
- [JeeType=JeusService, jeusType=JMSServiceChannel, JMOManager=adminServer, JEEServer=adminServer, JMSResource=adminServer, jms, name=JMSServiceChannel-internal](#)
- [JeeType=JeusService, jeusType=MXF-poolService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JavaMailResourceService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=JeusLogService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=RMIOConnector, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=SecurityDomain, JMOManager=adminServer, JEEDomain=domain1, SecurityService=SecurityService, name=SYSTEM_DOMAIN](#)
- [JeeType=JeusService, jeusType=SecurityPolicy, JMOManager=adminServer, JEEDomain=domain1, SecurityDomain=SYSTEM_DOMAIN, name=Policy](#)
- [JeeType=JeusService, jeusType=SecurityService, JMOManager=adminServer, JEEDomain=domain1, JEEServer=adminServer, name=SecurityService](#)
- [JeeType=JeusService, jeusType=SecuritySubject, JMOManager=adminServer, JEEDomain=domain1, SecurityDomain=SYSTEM_DOMAIN, name=Subject](#)
- [JeeType=JeusService, jeusType=ServerDeploymentService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)
- [JeeType=JeusService, jeusType=SessionContainer, JMOManager=adminServer, WebEngine=adminServer, servlet, JEEServer=adminServer, name=_webadmin](#)
- [JeeType=JeusService, jeusType=SnmpAgentService, JMOManager=adminServer, JEEServer=adminServer, name=adminServer](#)

HTML Adaptor 접속 화면

2. JMX 애플리케이션 개발

본 장에서는 JEUS에서 제공하는 MBean에 접근하기 위한 클라이언트 애플리케이션 개발에 대한 내용을 다룬다.

2.1. JMX 클라이언트 애플리케이션

본 절에서는 일반적인 JMX 클라이언트 애플리케이션의 동작에 대해 간략하게 설명한다.

1. JEUS MBean 서버에 연결하기 위해서는 아래 2가지 방법 중 하나를 사용할 수 있다.
 - JNDI에 등록되어 있는 정보를 사용(lookup)하여 연결([JNDI 사용 참고](#))
 - JMX Remote API를 사용하여 연결([JMX Remote API 사용 참고](#))
2. 사용하고자 하는 기능을 제공하는 MBean에 접근하여 속성값을 얻어오거나, 작업을 수행시킨다.
 - MBean에 접근하기 위해서는 해당 MBean을 가리키는 고유 이름이라고 할 수 있는 ObjectName을 알고 있어야 한다. JEUS에서 제공하는 MBean들이 갖는 ObjectName 형식에 대해서는 [MBean Object Names](#)를 참고한다.
 - 보안을 위해 각 MBean들이 제공하는 속성값이나 연산들은 수행 시 권한 체크를 수행하는 경우가 있다. 이러한 MBean을 사용하기 위해서는 적절한 권한을 갖는 사용자로 인증을 한 후 접근을 해야 한다. JMX 연결할 때 보안 설정에 관한 내용은 [Security 설정](#)을 참고한다.
3. 얻어온 속성값이나 수행시킨 작업의 결과를 받아와 처리한다.
4. 여러 작업을 수행하는 경우 2, 3의 과정을 반복한다.



본 장에서 설명하는 내용을 이해하기 위해서 JMX Remote API 1.0과 Jakarta EE Management 스펙에 대한 기본 지식이 필요하다. JMX Remote API에 대한 자세한 정보는 Oracle에서 제공하는 JMX Remote API 1.0 스펙과 JMX Remote API를 참고한다.

2.2. MBean 서버 연결

본 절에서는 MBean 서버로 접속하기 위한 방법에 대해서 설명한다.

2.2.1. JNDI 사용

본 절에서는 JNDI를 사용해서 JEUS를 모니터링하는 JMX 애플리케이션에 대해서 설명한다.

각 서버는 기동할 때 JMX 연결을 맺기 위해 필요한 정보를 담고 있는 javax.naming.Reference 객체를 JNDI에 등록한다. 등록할 때 사용하는 이름은 "mgmt/rmbs/adminServer"와 같은 형식을 가진다. 사용자는 JNDI에 등록되어 있는 Reference를 통해 MBean 서버와 연결을 맺을 수 있다.

다음은 JNDI를 사용하는 클라이언트 예제이다.

```
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());
        }
    }
}

```

위 예제를 작성한 후 컴파일하여 실행하면 JEUS 서버에 접속한 후 "JEUS:*"에 해당하는 MBean들의 목록을 출력한다. 예제 프로그램은 인자를 4개 받는데, 첫 번째는 서버의 hostname, 두 번째는 JEUS 사용자 이름, 세 번째는 비밀번호, 마지막은 서버 이름이다.

```
$ 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. 예제 프로그램은 jclient.jar가 있어야 실행할 수 있다. 기본적으로 jclient.jar는 JEUS_HOME/lib/client 아래에 위치한다.
2. JNDI의 자세한 정보에 대해서는 JEUS Server 안내서의 "JNDI Naming Server"를 참고한다. 만약에 JMX 애플리케이션이 서블릿 또는 EJB에서 실행된다면 JNDI 파라미터에 대한 설정은 필요하지 않다.

2.2.2. JMX Remote API 사용

본 절에서는 JMX Remote API를 사용해서 JEUS를 모니터링하는 JMX 애플리케이션에 대해서 가장 일반적으로 사용되는 방식으로 설명한다.

연결을 맺을 대상은 JMX Service URL을 통해 지정한다. JEUS MBean 서버에 연결하기 위한 URL은 아래와 같은 형태를 갖는다.

- service:jmx:jmxmp://0.0.0.0:9736/JeusMBeanServer
- service:jmx:jmxmp://0.0.0.0:9736/JEUSMP_<adminServer>(adminServer는 접근할 서버 이름에 따라 달라진다.)

다음은 JMX Remote API를 사용하는 클라이언트 예제이다.

```

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());
        }
    }
}

```

위 예제를 작성한 후 컴파일하여 실행하면 JEUS 서버에 접속한 후 "JEUS:*"에 해당하는 MBean들의 목록을 출력한다. 인자는 순서대로 서버 주소, 포트, 사용자 이름, 비밀번호이다.

```
$ 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
...
```



예제 프로그램은 jclient.jar가 있어야 실행할 수 있다. 기본적으로 jclient.jar는 JEUS_HOME/lib/client 아래에 위치한다.

2.3. Security 설정

JMX를 사용하여 JEUS 서버에 등록되어 있는 여러 MBean들이 제공하는 속성을 읽거나 작업할 경우 JEUS는 해당 연결을 맺은 사용자가 속성을 읽을 권한이 있는지 작업을 수행할 권한이 있는지를 검사한다. 각각 MBean을 사용할 때 필요한 권한에 대한 정보는 JEUS API 문서를 참고한다.

API 문서는 다음 위치에서 찾을 수 있다.

JEUS_HOME/docs/api



API 문서는 MBean을 사용할 때 필요한 권한(Permission Name)뿐만 아니라 ObjectNamePattern, 속성(Attribute), 작업(Operation) 등에 대한 정보도 제공하고 있다.

JMX 애플리케이션에서 사용자 이름이나 비밀번호와 같은 정보는 서버에 접속해서 MBeanServerConnection을 생성할 때 제공한다. 일반적으로 MBeanServerConnection을 생성할 때는 다음과 같은 코드를 사용하는데, 코드를

살펴보면 Hashtable로 전달하는 환경설정 정보에 사용자 이름과 비밀번호가 들어가는 것을 확인할 수 있다.

```
...

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();

...
```



이때 사용하는 사용자 이름과 비밀번호, 권한 설정은 JEUS Security를 이용해서 설정한다. JEUS Security 설정 방법에 대한 자세한 내용은 JEUS Security 안내서의 "보안 시스템 사용자 정보 설정"과 "보안 시스템 정책 설정"을 참고한다.

2.4. MBean Object Names

ObjectName은 MBean 객체의 기본 JMX 객체 이름이다. MBean 서버에 접근하여 MBean과 상호작용하기 위해서는 대상 MBean의 이름을 알고 있어야 한다. 만약 이름을 정확하게 알지 못하는 경우에는 알고 있는 부분을 이용하여 MBean 서버에 질의를 보낸 후 결과값을 받아 맞는 이름을 찾아볼 수 있다.

JEUS에서 제공하는 MBean에 대한 ObjectName은 아래와 같은 형태를 갖는다.

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

또는

```
<domain_name>: *
```

ObjectName은 <domain_name>으로 시작해야 하고, 각 이름과 값의 짝이 순차적으로 규정되지 않는다.

예를 들면 다음은 둘 다 JEUS 도메인 MBean의 objectname을 얻어온다.

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

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

다음은 각 항목에 대한 설명이다.

- <domain_name>
 - ObjectName의 도메인 이름으로 값은 'JEUS'이다.
- j2eeType
 - MBean은 J2EE 타입이며, J2EE Management 스펙에 의해 기술된다.
 - 다음 값들 중 하나를 설정한다.

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
 - MBean의 이름으로 각각의 MBean Object에는 유일한 값이 있다. 예를 들면 "adminServer"라는 서버가 실행하는 JVM의 이름은 'adminServer'이다.
- parent-j2eeType
 - MBean의 상위 J2EE 타입으로 각 MBean들에 계층이 규정되어 있다. 예를 들면 "JDBCDriver"의 상위 J2EE 타입은 'JDBCDataSource'이다.
- jeusType
 - JEUS JMX에서 정의된 MBean들의 타입이다. "JeusService" J2EE 타입만 몇 가지 jeusType을 가질 수 있다.
 - 다음 값들 중 하나를 설정한다.

EJBEngine	JMSClientResource	JMSConnectionFactoryResource
JMSDestinationResource	JMSDurableSubscriberResource	JMSEngine
JMSPersistenceStoreManager	JMSServiceChannel	SecurityDomain
SecurityPolicy	SecurityService	SecuritySubject

SessionContainer	SessionContainerCentral	SessionContainerP2P
ThreadPool	ThreadPool_WEBC	WebEngine
WebListener	WebServices	

- isTargetable
 - 설정값은 boolean 타입으로 사용자 AP(EJB, 서블릿, JSP)가 deploy되어 isTargetable 동작하는 MBean에서는 반드시 true로 설정되어야 한다.
- JMXManager
 - MBean 서비스를 제공하는 JMXManager의 이름이다. 일반적으로 JMXManager 항목의 값은 해당 JMXManager가 속한 서버 이름과 동일하다.

2.5. Spring JMX Support

JEUS 8 버전부터 Spring JMX를 사용할 때 JEUS 서버가 제공하는 MBean 서버를 사용할 수 있도록 라이브러리를 제공한다. 아래 경로를 기준으로, 지원하고 있는 스프링 버전 별로 라이브러리를 각각 제공하고 있다.

JEUS_HOME/lib/shared/spring-support/



JEUS에서 제공하는 라이브러리를 사용하지 않더라도 Spring JMX는 JEUS 상에서 동작한다. JEUS라이브러리를 사용하려면 애플리케이션에 라이브러리를 패키징하고, 추가적인 설정이 필요하므로 JEUS 서버가 사용하는 MBean 서버를 사용해야 할 경우에만 사용하는 것을 권장한다.

JEUS Spring JMX 지원 라이브러리를 사용하기 위해서는 먼저 라이브러리를 WEB-INF/lib 아래에 넣은 후 Spring 설정 파일에 아래와 같이 설정하여 Spring JMX가 JEUS MBean 서버를 찾아서 사용할 수 있도록 한다.

```
<?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>



Spring 프레임워크 및 Spring JMX 에 대한 자세한 사항은 [Spring Framework 공식 문서](#)를
참고한다.