

# Node Manager 안내서

JEUS 9.1

**TMAXSOFT**

## 저작권 공지

Copyright 2026. TmaxSoft Co., Ltd. All Rights Reserved.

## 회사 정보

(주)티맥스소프트

주소 : 경기도 성남시 분당구 정자일로 45, 티맥스소프트타워

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 9.1.1	3.4.1	2026-06-08	Fix 릴리스
JEUS 9.1	3.3.1	2025-09-30	GA 버전
JEUS 9 Fix#1	3.2.2	2025-07-10	JDK 21 지원 내용 추가
JEUS 9 Fix#1	3.2.1	2025-06-30	RC 버전
JEUS 9	3.1.2	2025-01-03	-

제품 버전	안내서 버전	발행일	비고
JEUS 9	3.1.1	2024-09-27	-

# 목차

1. 소개	1
1.1. 개요	1
1.1.1. 개념	1
1.1.2. 목적	1
1.1.3. 종류	2
1.1.4. 제약 사항	2
1.2. 타입별 노드 매니저 설정	3
1.3. 공통 기능	3
1.4. RQS 프로세스 관리 기능	4
2. Java 타입 노드 매니저	5
2.1. 개요	5
2.2. 기능별 동작 방식	5
2.2.1. 원격 머신에 있는 ManagedServer 기동	5
2.2.2. 서버 모니터링	6
2.2.3. 비정상 상태의 서버 재기동	7
2.2.4. Rolling Patch	8
2.2.5. Master와 연결	9
2.3. 환경설정	9
2.3.1. 설정 파일	9
2.3.1.1. nodes.xml	9
2.3.1.2. jeusnm.xml	10
2.3.1.3. <serverName>.properties	11
2.3.2. 필수 파일	11
2.4. 노드 설정 및 삭제	12
2.4.1. Java 타입 노드 설정	12
2.4.1.1. 콘솔 툴 사용	13
2.4.2. Java 타입 노드 삭제	14
2.4.2.1. 콘솔 툴 사용	14
2.5. 노드 매니저 시작 및 종료	14
2.5.1. Java 타입 노드 매니저 시작	15
2.5.2. Java 타입 노드 매니저 종료	16
2.6. Java 타입 노드 매니저를 통한 서버 제어	16
2.6.1. 콘솔 툴 사용	16
2.7. 로그 파일	19
3. SSH 타입 노드 매니저	20
3.1. 개요	20
3.2. 환경설정	20
3.2.1. SSH 설정	20
3.3. 노드 설정 및 삭제	22

3.3.1. SSH 타입 노드 설정 .....	22
3.3.1.1. WebAdmin 사용 .....	22
3.3.1.2. 콘솔 툴 사용 .....	23
3.3.2. SSH 타입 노드 변경 .....	26
3.3.3. SSH 타입 노드 삭제 .....	26
4. RQS 프로세스 관리 .....	27
4.1. 개요 .....	27
4.2. RQS의 동작 방식 .....	27
4.2.1. RQS 프로세스의 구동과 정지 .....	27
4.2.2. RQS 프로세스 모니터링 .....	27
4.2.3. 비정상 상태의 프로세스 재기동 .....	28
4.3. 환경설정 .....	28
4.3.1. jeusnm.xml .....	28
5. 노드 매니저 이중화 .....	30
5.1. 개요 .....	30
5.2. 이중화된 노드 매니저 동작 방식 .....	30
5.3. 이중화된 노드 매니저 사용 방법 .....	31
5.3.1. Standby 노드 매니저 .....	31
5.4. 이중화 상태에서 노드 매니저 종료 .....	31

# 1. 소개

본 장에서는 노드 매니저의 기본적인 사항과 공통 기능에 대해서 설명한다.

## 1.1. 개요

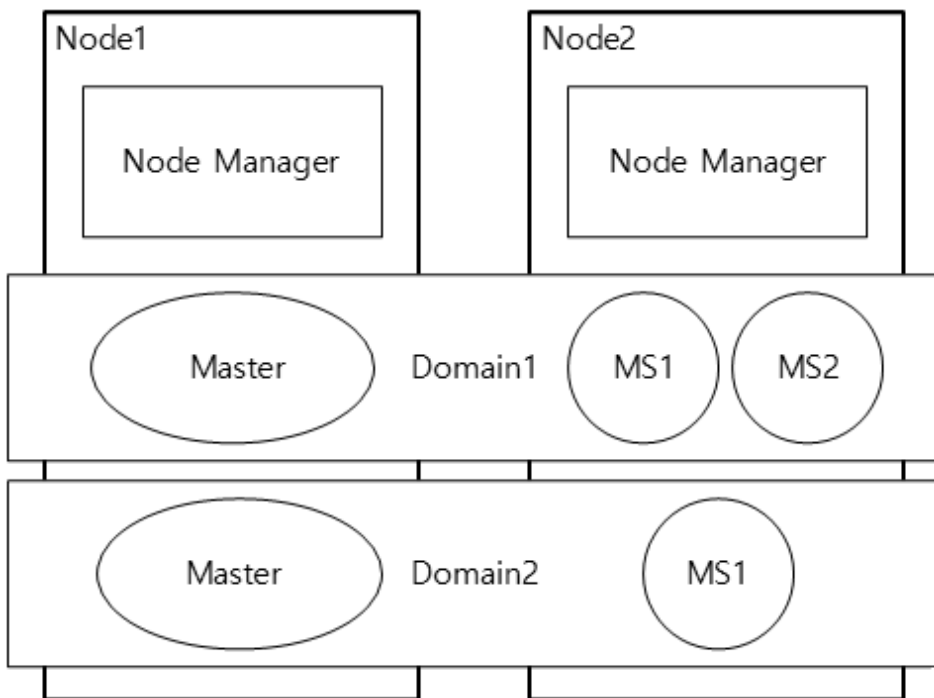
본 절에서는 노드 매니저의 기본 개념, 목적, 종류, 노드 매니저를 구성할 때의 제약사항 등 기본적인 사항에 대해 설명한다.

### 1.1.1. 개념

도메인 환경에서는 도메인을 구성하는 서버들이 여러 머신에 존재할 수 있고 하나의 머신에 여러 도메인이 존재할 수도 있다. 하나의 머신에는 하나의 JEUS가 설치되고, 설치된 JEUS에 하나의 노드 매니저가 존재할 수 있다. 이러한 환경에서 도메인에 국한되지 않고 하나의 머신에서 서버 프로세스를 관리하는 주체가 노드 매니저(Node Manager)이다. 즉, 도메인 단위가 아닌 머신 단위로 서버를 관리하고 해당 머신에 존재하는 서버들을 시작, 종료, 재기동해서 서버가 정상 상태로 서비스할 수 있게 한다. 노드 매니저는 JEUS마다 하나씩 존재한다.

서버가 실제 실행되는 머신에 설치된 JEUS를 노드(Node)라고 정의한다. 노드는 해당 머신의 주소, JEUS 설치 디렉터리 경로 등의 정보로 유일한 노드를 설정할 수 있다.

다음은 노드와 도메인, 노드 매니저의 관계를 나타낸 그림이다.



JEUS와 도메인, 노드 매니저의 관계

### 1.1.2. 목적

노드 매니저의 주요 목적은 하나의 노드에서 동작하는 서버 프로세스들을 관리하는 것이다.

노드 매니저는 Master Server(이하 Master)와 다른 원격 머신에서 동작하는 서버를 시작할 수 있다. 또한 서버가 비정상 종료되거나 사용자가 설정한 비정상 상태가 되었을 때 서버를 재기동할 수 있다. 서버가 갑자기 다운된 경우에는 Master가 SCF(System Clustering Framework)를 통해 이를 감지하여 자동으로 재시작하는 기능을 제공하고 있다.

노드당 하나의 인스턴스가 존재하고, 서버에 적용할 Patch 파일을 전달받아서 적용시켜 줄 수 있다. 또한 노드 매니저를 통해 Master에 의해 관리되는 서버를 콘솔 툴(jeusadmin)의 start-server 명령어를 사용하여 간편하게 실행할 수 있다. 또한 서버 프로세스들의 관리와 더불어 외부 프로세스를 관리할 수 있는 기능을 하는데, 현재는 RQS 프로세스를 관리할 수 있다. RQS 프로세스의 기동이나 정지 또는 이상 종료하는 경우 재시작을 할 수 있도록 도와준다. 이 기능들은 서버 프로세스 관리와는 독립된 기능으로, 설정 파일의 설정 정보만을 가지고 프로세스의 관리를 하게 된다.



노드 매니저는 부가적인 기능이므로 사용하지 않더라도 서버가 서비스하는 데 아무런 영향을 주지 않는다. 그러나 도메인의 안정적인 운영을 위해서 노드 매니저의 사용을 권장한다.

### 1.1.3. 종류

JEUS에서 제공하는 노드 매니저는 다음과 같이 2가지 종류로 구분된다.

- Java 타입 노드 매니저

Java로 구현된 노드 매니저로 OS에 상관없이 실행될 수 있다.

Master와 다른 노드에 존재하는 Managed Server(이하 MS)를 시작시킬 수 있다. 노드 매니저는 주체적으로 서버의 상태와 비정상 종료를 감지해서 문제가 발생한 서버의 재기동 동작이 SSH 타입 노드 매니저보다 빠르고 간결하게 진행된다. 또한 도메인에 Patch 파일을 적용할 수 있다. 그러나 Java 타입 노드 매니저는 JEUS가 설치되어 있어야 설정해서 사용할 수 있으므로 SSH 타입 노드 매니저와는 달리 다른 머신에 JEUS를 설치할 수 없다. Java 타입의 노드 매니저에 대한 자세한 내용은 [Java 타입 노드 매니저](#)를 참고한다.

- SSH 타입 노드 매니저

OS에서 제공하는 SSH를 사용하는 노드 매니저로 OS가 Windows인 경우에는 사용할 수 없다. Windows가 설치된 머신에서 노드 매니저 기능을 사용하려면 Java 타입 노드 매니저를 사용해야 한다.

SSH 타입 노드 매니저는 Java 타입 노드 매니저와 마찬가지로 원격 머신에 있는 서버를 기동하고, 비정상 종료를 감지하여 재기동한다. 단, SSH 타입 노드 매니저는 직접 서버 프로세스를 모니터링하지 않기 때문에 Master에서 SCF를 통해서 서버의 비정상 상태를 감지하고 SSH 타입 노드 매니저를 통해 원격 서버를 재기동한다. SSH 타입 노드 매니저가 Java 타입 노드 매니저와 다른 점은 SSH 타입 노드 매니저를 통해 다른 머신에 JEUS를 설치할 수 있다는 것이다. SSH 타입의 노드 매니저에 대한 자세한 내용은 [SSH 타입 노드 매니저](#)를 참고한다.

### 1.1.4. 제약 사항

다음은 노드 매니저를 구성할 때의 제약 사항이다.

- 도메인 내에서 사용되는 노드 매니저의 이름은 유일해야 한다.



- 설치한 하나의 JEUS 내에서는 하나의 노드 매니저만 존재한다.
- OS가 Windows라면 SSH가 제공되지 않으므로 Java 타입 노드 매니저를 사용해야 한다.
- Java 타입 노드 매니저를 사용하려면 OS에서 제공하는 서비스로 등록해서 사용할 것을 권장한다.
- Java 타입 노드 매니저를 사용하려면 jeusnm.xml 파일의 'useNodeManager' 항목을 true로 설정해야 한다.

## 1.2. 타입별 노드 매니저 설정

콘솔 툴을 사용해서 노드 매니저를 설정할 수 있다.

### • SSH 타입 노드 매니저 설정

jeusadmin에서 'add-ssh-node' 명령어를 사용한다. 옵션을 통해 추가할 노드의 이름, 호스트 정보 등 필요한 항목을 설정할 수 있다. SSH 타입 노드 매니저 설정 방법에 대한 자세한 내용은 [SSH 설정](#)을 참고한다.

### • Java 타입 노드 매니저 설정

jeusadmin에서 'add-java-node' 명령어를 사용한다. 옵션을 통해 추가할 노드의 이름, 호스트 정보 등 필요한 항목을 설정할 수 있다. Java 타입 노드 매니저 설정 방법에 대한 자세한 내용은 [Java 타입 노드 설정](#)을 참고한다.

## 1.3. 공통 기능

다음은 2가지 타입의 노드 매니저에서 공통으로 제공하는 기능이다.

### • 원격 머신에서 서버 기동

노드 매니저를 통해서 도메인에 속한 서버를 기동할 수 있다.

노드 매니저를 사용하지 않을 경우 서버를 기동하려는 원격 머신에 접속해서 스크립트를 통해 서버를 기동해야 한다. 그러나 노드 매니저를 사용한다면 원격 머신에 있는 서버도 Master에서 간단한 명령으로 기동시킬 수 있다. Master가 기동된 상태에서 도메인에 속하는 모든 MS를 기동시킬 수도 있고, 클러스터를 기동시킬 수도 있다.

노드 매니저로 서버를 기동하려면 다음의 조건이 충족되어야 한다.

- 서버를 기동하려는 노드 정보가 nodes.xml에 등록되어 있어야 한다.
- 어떤 노드에서 기동할 것인지에 대한 정보가 기동하려는 서버에 설정되어 있어야 한다.
- 노드 매니저에 접속할 수 있는 상태여야 한다.

Java 타입의 노드 매니저를 사용하는 경우에는 노드 매니저가 기동되어 있는 상태여야 하고, SSH 타입의 노드 매니저를 사용하는 경우에는 OS에 SSH Port가 열려있어야 한다.

- Java 타입의 노드 매니저를 사용하는 경우 jeusnm.xml에 노드 매니저가 설정되어 있어야 한다.

### • 비정상 상태의 서버 재기동

Java 타입의 노드 매니저에서는 모니터링하고 있는 서버가 비정상 종료할 경우 프로세스를 재기동한다. SSH

타입의 노드 매니저의 경우는 도메인의 SCF에 의해 서버가 비정상(FAILED) 상태가 되면 Master에서는 서버가 속한 노드 매니저에 서버에 대한 재기동 명령을 보내 서버가 재기동될 수 있도록 한다. 이 동작은 해당 서버에 노드가 반드시 설정되어 있어야 하고 Java 타입 노드 매니저의 경우는 노드 매니저가 기동되어 있는 상태여야 한다. 만약 비정상 상태의 MS를 재기동할 때 Master도 비정상 상태라면, 노드 매니저는 MS에 설정된 외부 저장소로부터 설정 파일을 받아와 재기동할 수 있게 해준다.



Java 타입의 노드 매니저와 SSH 타입의 노드 매니저에서 제공하는 각 기능들에 대해서는 [Java 타입 노드 매니저](#)와 [SSH 타입 노드 매니저](#)를 참고한다.

## 1.4. RQS 프로세스 관리 기능

노드 매니저에서 추가적으로 다른 프로세스를 관리할 수 있는 기능이 있다. 현재 RQS 프로세스들의 상태를 체크하고 이상 종료되었다고 판단되면 이를 다시 시작시키는 역할을 한다.

- 해당 노드에 설정된 RQS 프로세스 실행 및 관리

노드 매니저에서는 설정된 RQS의 정보를 가지고 RQS를 구동할 수 있다. 또한 설정된 port를 통하여 RQS 프로세스의 상태를 지속적으로 체크할 수 있고, 혹시 이상 종료가 되었다고 판단되었을 경우에는 이를 인지한 후 RQS 프로세스를 재시작하도록 하여 지속적인 서비스가 가능하도록 할 수 있다.

노드 매니저로 RQS 프로세스를 기동하려면 다음의 조건이 충족되어야 한다.

- RQS 프로세스에 대한 정보가 jeusnm.xml 파일에 등록되어 있어야 한다.
  - RQS 프로세스의 기동에 필요한 설정 파일이 정상적인 위치에 존재하고, 노드 매니저에 설정된 설정 값(path, port 등)들이 제대로 설정되어 있어야 한다. 설정 파일의 위치는 RQS 프로세스에서 RQSDIR 환경변수를 기반으로 추산하기 때문에 환경변수 설정도 중요하다.
  - 노드 매니저에 접속할 수 있는 상태여야 한다.
- 비정상 상태의 프로세스 재기동

노드 매니저가 RQS 프로세스를 관리하는 주요 역할은 RQS 프로세스가 이상 종료되었을 때 이를 체크하고 재시작할 수 있다는 것이다. 이를 위하여 지정된 port로 접근하여 서로 메시지를 주고받게 된다.

RQS가 이상 종료되었다고 판단하면 노드 매니저는 설정된 값들을 가지고 RQS를 재기동하게 되며, RQS에서 구동한 shared 리소스의 정리를 위해 -r 옵션을 사용하여 RQS를 구동하게 된다.



RQS 프로세스 관리에 대한 자세한 내용은 [RQS 프로세스 관리](#)를 참고한다.

## 2. Java 타입 노드 매니저

본 장에서는 Java 타입 노드 매니저를 사용하기 위한 설정 방법과 동작 방식에 대해서 설명한다.

### 2.1. 개요

Java 타입 노드 매니저는 SSH 타입 노드 매니저에 비해 빠르게 서버의 비정상 종료를 감지하여 재기동할 수 있다는 장점이 있다. 또한 OS가 Windows인 경우에 SSH 타입 노드 매니저는 사용할 수 없지만, Java 타입 노드 매니저는 OS에 상관없이 사용할 수 있다. 그러나 다른 머신에 JEUS를 설치할 수 없다는 단점이 있다.

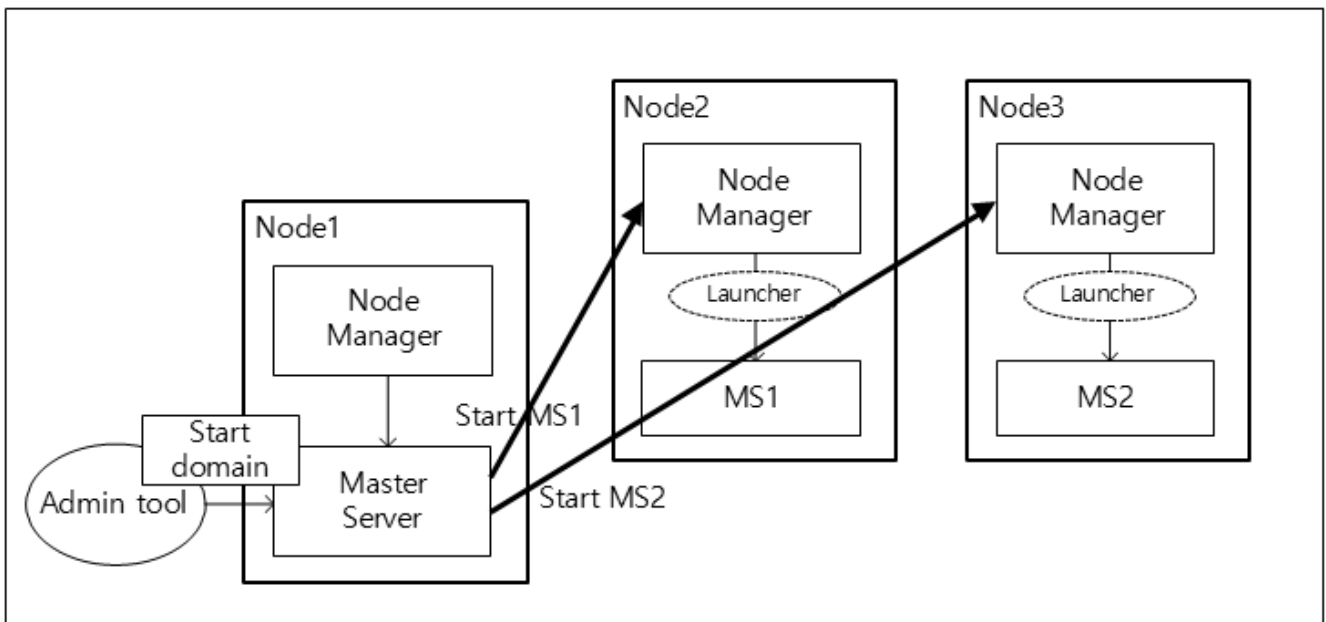
본 절에서는 Java 타입의 노드 매니저를 사용하기 위한 방법과 기본 지식에 대해서 살펴본다.

### 2.2. 기능별 동작 방식

본 절에서는 Java 타입 노드 매니저의 각 기능에 대한 동작 방식을 설명한다.

#### 2.2.1. 원격 머신에 있는 ManagedServer 기동

노드 매니저를 통해 도메인 내의 원격 머신에 있는 서버를 기동시킬 수 있다. 노드 매니저를 통해 서버를 기동하는 과정은 다음과 같다.



노드 매니저에서 원격 머신의 서버 기동

1. 노드 매니저에서 서버의 lock 파일을 생성하여 FILE LOCK을 얻는다.
2. 서버 기동을 성공한 뒤 pid 파일, state 파일을 생성하고 PID(프로세스 ID)와 서버의 상태를 기록한다.
3. `<serverName>.properties`에 username, password, masterurl 등의 정보를 업데이트한다.
4. 서버에서는 NodeManagerService를 시작하면서 address 파일을 생성하고 자신의 호스트 정보를 기록한다.



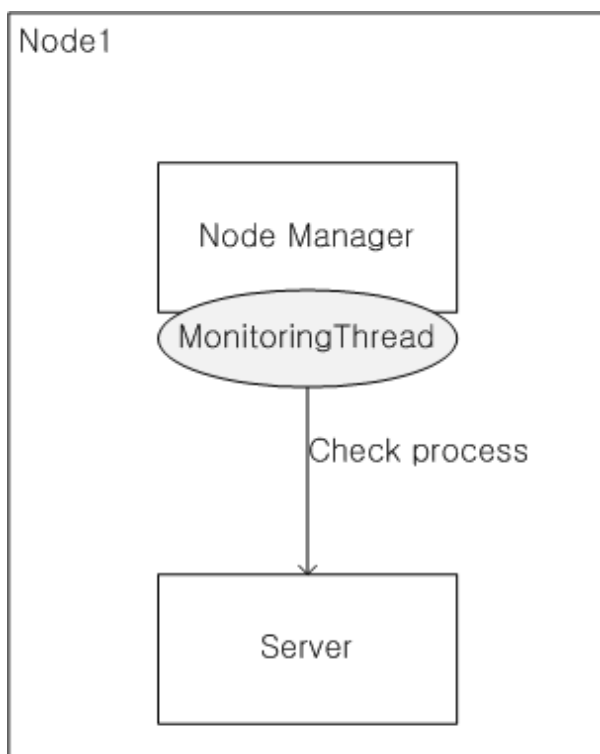
서버를 재기동할 경우에도 동일한 과정으로 진행된다.

## 2.2.2. 서버 모니터링

Java 타입 노드 매니저는 서버 프로세스의 상태를 주기적으로 모니터링해서 state 파일에 업데이트하고, 비정상 상황이 발생할 경우 서버 프로세스를 재기동한다.

노드 매니저는 노드에 존재하는 모든 서버에 대해서 모니터링하는 것이 아니고, 노드 매니저를 통해 기동한 서버의 프로세스만 모니터링한다. 따라서 서버의 비정상 종료를 감지해서 바로 재기동하고 싶다면 노드 매니저를 사용해서 기동해야 한다. 노드 매니저가 재기동된 경우에도 자신이 기동한 서버에 대한 모니터링은 계속 유지된다.

다음은 Java 타입 노드 매니저가 서버를 모니터링하는 과정을 간략하게 나타낸 그림이다.



Java 타입 노드 매니저에서 서버 모니터링

다음은 노드 매니저가 재기동된 경우에 자신이 기동한 서버를 찾아서 모니터링을 계속하는 과정에 대한 단계별 설명이다.

1. 노드 매니저가 재기동될 때 다음 경로에서 해당 노드 매니저에서 관리되고 있는 도메인 이름을 조회한다.

```
JEUS_HOME/domains
```

2. 각 도메인의 다음 경로의 서버 디렉토리를 통해 해당 노드 매니저에서 관리되고 있는 서버 이름을 조회한다.

```
JEUS_HOME/domains/<domain-name>/servers
```

3. 서버 디렉터리에서 `<serverName>.address`, `<serverName>.lck`, `<serverName>.state`, `<serverName>.pid` 파일의 존재 여부를 확인한다. 나열한 모든 파일이 존재하는 경우 서버는 운용 중인 상태이다.
4. lock 파일을 통해 FILE LOCK을 얻는다.
5. state 파일에 기록된 서버의 상태를 확인하고 서비스 가능한 상태인지를 판단한다.
  - 서비스 가능 상태 : RUNNING, STANDBY, SUSPENDED, RESUMING, SUSPENDING, STARTING
  - 서비스 불가능 상태 : SHUTDOWN, SUNTTING\_DOWN
6. pid 파일에 기록된 서버 프로세스 ID로 프로세스가 운용 중인지 여부를 확인한다. 만약 서버가 다운된 것을 확인하면 노드 매니저는 해당 서버를 재기동한다.
7. address 파일에 기록된 서버의 호스트 정보를 바탕으로 소켓 연결을 맺어보고 서버 프로세스가 정상적으로 운영되고 있는지를 확인한다. 만약 서버가 다운된 것을 확인하면 노드 매니저는 해당 서버를 재기동한다.
8. 서버가 서비스 가능한 상태라고 판단되면 주기적으로 서버의 상태를 확인하면서 서버를 모니터링한다.

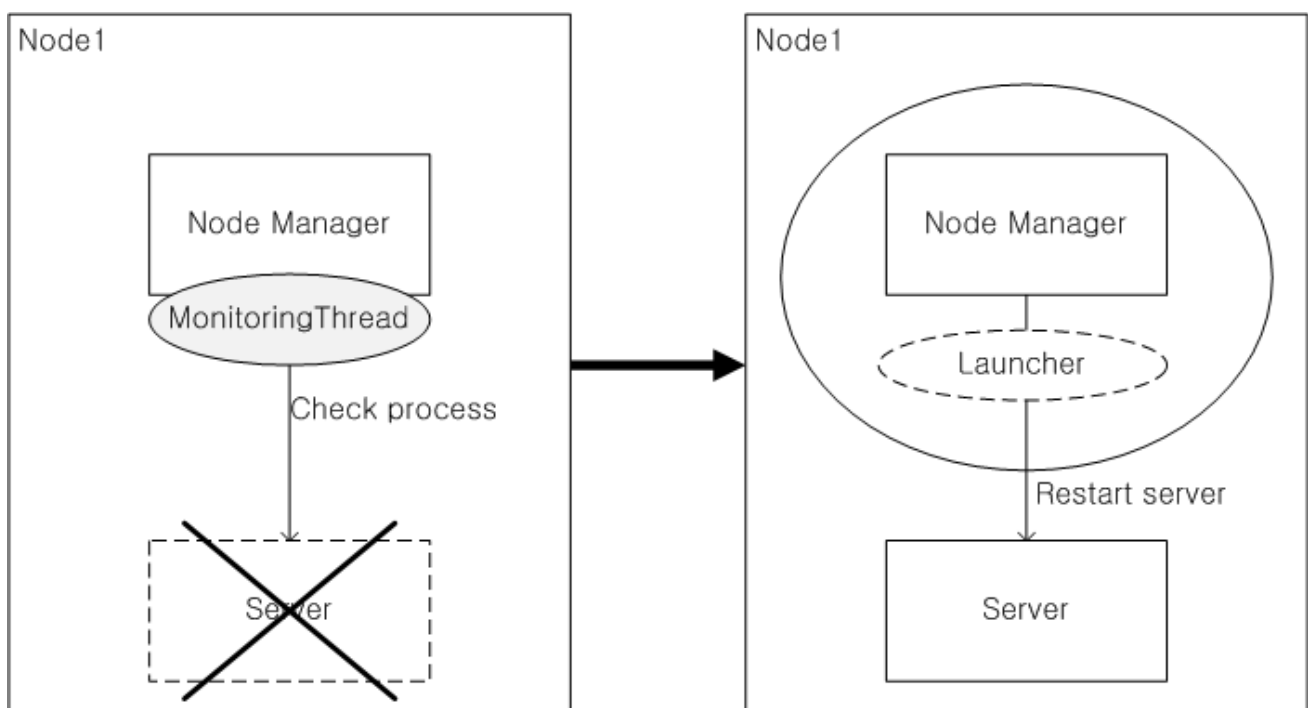


state 파일에 기록되는 서버의 상태 중 SHUTDOWN\_NEED\_TO\_RESTART 상태는 노드 매니저가 재기동될 때 이전에 노드에서 서비스 중이던 서버를 재기동하기 위해 사용되는 state이다. 주로 Windows 서비스와 같이 노드 매니저를 OS의 서비스로 등록한 경우 서버 shutdown 후 재기동될 때 기존에 서비스 중이던 서버를 재기동하여 서비스할 수 있도록 하기 위해 사용된다.

### 2.2.3. 비정상 상태의 서버 재기동

Java 타입 노드 매니저는 서버 프로세스가 비정상 종료된 상황에서 재기동하는 역할도 하지만 서버가 정의한 비정상 상태에 도달한 경우에도 서버를 재기동한다.

서버에서는 Memory Ratio를 설정해서 지정한 비율 이상으로 Heap Memory를 사용하면 자체적으로 프로세스를 종료한다. 노드 매니저는 서버가 비정상 상황에 의해 자체적으로 종료되는 상황도 감지해서 서버를 재기동한다.



Java 타입 노드 매니저의 서버 재기동

## 2.2.4. Rolling Patch

Java 타입 노드 매니저를 통해 도메인에 Patch를 적용할 수 있다. 콘솔 툴을 사용해서 원격 머신에 있는 서버에 Patch 파일을 전송 또는 삭제뿐만 아니라 Patch 적용을 위해 서버의 재기동까지 진행한다.

Master에서 도메인에 속한 서버가 존재하는 원격 머신의 노드 매니저에 접속해서 Patch 파일을 전달, 삭제, 적용할 수 있다.

- Patch 파일 전달

```
domain1.adminServer>apply-patch
```

- Patch 파일 삭제

```
domain1.adminServer>remove-patch
```

- Patch 파일 적용

-rolling 옵션을 통해 각 서버에 Patch를 적용시킬 수 있다. 이때 노드별로 하나씩 순차적으로 수행하여 서비스가 이중화되어 있다면 장애 없이 서비스할 수 있도록 한다. 단, 같은 노드 안에 여러 대의 서버를 운영하고 있는 상황이라면 노드 전체의 서버를 하나씩 순차적으로 종료하고 Patch 파일을 전송 또는 삭제한 후 다시 서버를 하나씩 순차적으로 기동해야 한다.

```
domain1.adminServer>apply-patch -rolling
```

```
domain1.adminServer>remove-patch -rolling
```

- Patch 적용에 실패한 경우 되돌리는 기능

-action 옵션을 통해서 Patch 적용에 실패한 경우의 동작에 대해서 정의할 수 있다.

```
domain1.adminServer>apply-patch -rolling -action CONTINUE|STOP|ROLLBACK
```

다음의 CONTINUE, STOP, ROLLBACK 옵션 중 하나를 설정할 수 있다.

옵션	설명
CONTINUE	Patch 적용에 실패한 노드가 있더라도 무시하고 계속 진행한다.
STOP	Patch 적용에 실패한 노드가 있으면 Patch 적용을 멈추고 리턴한다.
ROLLBACK	Patch 적용에 실패한 노드가 있으면 Patch 적용을 멈추고 이전에 성공한 노드에도 모두 적용된 Patch를 rollback한다. 이때 Patch를 삭제하는 상황에서는 Patch 파일이 이미 삭제된 노드가 존재할 수 있기 때문에 이 옵션을 설정하더라도 STOP 옵션을 설정한 것과 동일하게 더 이상 Patch 삭제를 진행하지 않고 동작을 멈춘다.

## 2.2.5. Master와 연결

노드 매니저를 통하지 않고 기동된 서버는 노드 매니저의 관리를 받을 수 없기 때문에 모니터링 기능이나 Patch 기능을 제공하지 않는다. 그러나 Master는 예외적으로 노드 매니저를 통하지 않고 기동되어도 프로세스 모니터링과 Patch 전달이 가능하다. Master는 기동이 완료되면 노드 매니저에 연결해서 자신의 정보를 전송하므로 노드 매니저에서는 이 정보를 바탕으로 Master를 모니터링할 수 있다.

Rolling Patch 기능은 Master와 노드 매니저가 연결되어 있는 상태에서만 사용이 가능하다. Master가 노드 매니저를 통해 기동되지 않았더라도 Master와 노드 매니저가 연결된 상태이므로 자신의 노드뿐만 아니라 다른 노드에도 Patch 전달 및 적용이 가능하다.



노드 매니저를 통하지 않고 기동된 서버는 모니터링하지 않는다. Java 타입 노드 매니저를 사용하면 노드 매니저를 통해 기동된 서버인 경우에 서버가 FAILED 상태가 되기 전에 노드 매니저가 서버의 비정상 종료를 감지하여 재기동할 수 있다.

## 2.3. 환경설정

Java 타입 노드 매니저를 사용하기 위해 필요한 환경을 설정해야 한다.

### 2.3.1. 설정 파일

Java 타입 노드 매니저를 사용하려면 다음의 설정 파일에 필요한 사항을 설정해야 한다. Java 타입 노드 매니저는 nodes.xml에 노드를 정의하는 것 이외에도 별도의 설정이 필요하다.

- [nodes.xml](#)
- [jeusnm.xml](#)
- [<serverName>.properties](#)

#### 2.3.1.1. nodes.xml

다음 경로에 있는 nodes.xml 파일에 어떤 타입의 노드 매니저를 사용할 것인지 설정한다.

```
JEUS_HOME/domains
```

nodes.xml은 Master에서 참조하는 노드의 설정으로 노드의 물리적인 주소를 정의하고 도메인 설정에서는 각 MS가 운영될 노드 정보를 설정해야 한다. 이 파일은 하나의 도메인에 국한된 파일이 아니라 머신에 설치된 JEUS에 존재하는 여러 도메인에서 모두 공유된다. 각 도메인의 Master에서는 node.xml에 설정된 노드 정보에 따라 각 노드의 노드 매니저에 접속해서 필요한 명령어를 수행한다.

콘솔 툴을 사용하여 노드를 추가하거나 삭제할 수 있다. 노드 추가 및 삭제에 대한 자세한 방법은 [노드 설정 및 삭제](#)를 참고한다.



Java 타입 노드 매니저는 nodes.xml에 설정된 머신 정보와 jeusnm.xml에 설정된 '**host**', '**port**' 정보가 일치하도록 주의해서 설정해야 한다.

### 2.3.1.2. jeusnm.xml

jeusnm.xml는 머신 내 노드 매니저의 동작 방식을 설정하는 설정 파일로 다음 경로에 위치한다.

```
JEUS_HOME/nodemanager
```

jeusnm.xml 파일을 열어 다음의 항목을 직접 설정한다.

항목	설명
useNodeManager	Java 타입 노드 매니저를 사용할지 여부를 설정한다. (기본값: true)
host	노드 매니저의 listen address를 설정한다. (기본값: localhost)
port	노드 매니저의 listen port를 설정한다. (기본값: 7730)
serverMonitoringPeriod	노드 매니저에서 서버 프로세스의 상태를 모니터링하는 주기를 설정한다. (기본값: 500, 단위: ms)
serverAutoRestart	서버 프로세스가 비정상 종료된 경우 자동으로 해당 서버를 재기동할지 여부를 설정한다. (기본값: true)
serverRestartTryCount	서버 재기동을 시도할 횟수를 설정한다. (기본값: 5)  ' <b>serverRestartDurationTime</b> '에 설정한 주기 안에 이 값에 설정한 횟수 이상의 서버 재기동이 시도되면 서버 프로세스에 문제가 있는 것으로 판단하고 서버 재기동을 진행하지 않는다.
serverRestartDurationTime	서버 재기동을 시도할 주기를 설정한다. (기본값: 120000, 단위: ms)  설정된 주기 안에 ' <b>serverRestartTryCount</b> '에 설정한 횟수 이상의 서버 재기동이 시도되면 서버 프로세스에 문제가 있는 것으로 판단하고 서버 재기동을 진행하지 않는다.
serverRetryRestart	서버 재기동이 실패할 경우 재시도할지 여부를 설정한다. 이 값을 true로 설정하면 서버 재기동에 성공할 때까지 재시도한다. (기본값: false)
useSSLListener	노드 매니저가 SSL을 사용할지 여부를 설정한다. (기본값: false)
keystoreFile	SSL을 사용할 때 인증에 사용될 Keystore 파일 경로를 설정한다. (기본값: NODEMANAGER_HOME/keystore)
keystorePass	SSL을 사용할 때 인증에 사용될 Keystore 파일의 패스워드를 설정한다. (기본값: jeuskeypass)
truststoreFile	SSL을 사용할 때 인증에 사용될 Truststore 파일 경로를 설정한다. (기본값: NODEMANAGER_HOME/truststore)
truststorePass	SSL을 사용할 때 인증에 사용될 Truststore 파일의 패스워드를 설정한다. (기본값: jeustruststorepass)



항목	설명
logFileName	노드 매니저의 로그 파일 위치를 설정한다. 파일의 절대 경로를 준 경우 해당 경로에 로그 파일을 만들고, 상대 경로를 준 경우 기본 로그 디렉터리에 설정한 이름으로 로그 파일을 만든다. (기본값: JEUS_HOME/nodemanager/logs/JeusNodeManager.log)
standbyPort	노드 매니저의 standby 프로세스를 돌려서 노드 매니저가 이상 종료될 때 standby 프로세스가 자동으로 대체하게 할 수 있다. 사용할 포트를 설정함으로써 기능을 켤 수 있다. 자세한 내용은 <a href="#">Standby 노드 매니저</a> 를 참고한다.
processList	RQS 프로세스들의 정보를 리스트의 형태로 설정한다. 자세한 설정 방법은 <a href="#">환경설정</a> 을 참고한다.

### 2.3.1.3. <serverName>.properties

<serverName>.properties 파일은 노드 매니저에서 서버를 기동할 때 필요한 정보들에 대한 캐시 파일이다. 노드 매니저에서 서버를 기동한 뒤 다음 경로에 해당 파일을 생성한다.

```
SERVER_HOME/nodemanager
```

노드 매니저는 서버를 기동할 때 필요한 옵션들을 기록해서 다음 재기동 시 옵션을 설정하지 않더라도 이 파일에 캐싱된 정보를 바탕으로 서버를 기동시킬 수 있다. 이 정보들은 콘솔 툴을 통해 노드 매니저에 직접 접속해서 서버를 기동시킬 때만 사용되고 Master로 MS를 기동시킬 때는 사용되지 않는다.

다음은 캐시되는 정보에 대한 설명이다.

항목	설명
masterurl	Master의 호스트 정보를 나타낸다. MS의 경우에만 masterurl을 캐시한다.
username	서버를 기동할 때 필요한 서버의 계정을 나타낸다.
password	서버를 기동할 때 필요한 서버의 패스워드를 나타낸다.
sslArguments	SSL을 사용하도록 설정된 서버의 경우 노드 매니저에서 서버에 접속할 때 SSL 인증에 필요한 SSL properties 정보를 설정한다.

### 2.3.2. 필수 파일

설정 파일 외에 노드 매니저가 서버를 기동하고 모니터링하는 데 필요한 파일을 설정해야 한다.

해당 파일은 다음 경로에 위치한다.

```
JEUS_HOME/domains/<domain_name>/servers/<server_name>/nodemanager
```

파일	설명
<code>&lt;serverName&gt;.lck</code>	<p>서버 프로세스를 기동할 때 서버가 여러 번 기동되는 것을 방지하기 위해 노드 매니저에서 생성하는 파일이다. 이 파일은 노드 매니저에서 서버 프로세스를 시작시키기 전에 생성되고 서버 프로세스가 정상 종료되면 삭제된다.</p> <p>노드 매니저가 재기동되었을 때 해당 파일의 유무에 따라 서버를 계속 모니터링할지 여부를 결정한다.</p>
<code>&lt;serverName&gt;.pid</code>	<p>서버 프로세스가 성공적으로 기동된 후에 노드 매니저에서 생성하는 서버의 프로세스 ID를 기록한 파일이다. 이 파일은 노드 매니저에서 서버 프로세스를 정상적으로 기동시킨 뒤에 생성되고 서버 프로세스가 정상 종료되면 삭제된다.</p> <p>노드 매니저가 재기동되었을 때 해당 파일의 유무에 따라 서버를 계속 모니터링할지 결정한다. 또한 파일에 기록된 PID에 해당하는 프로세스의 존재 여부에 따라 서버의 비정상 종료를 판단해서 노드 매니저가 재기동된 후에도 서버를 재기동하여 계속 모니터링할 수 있다.</p>
<code>&lt;serverName&gt;.state</code>	<p>서버 프로세스의 기동이 성공적으로 완료된 후 노드 매니저에서 생성하는 서버의 상태를 기록한 파일이다. 이 파일은 노드 매니저에서 서버 프로세스를 정상적으로 기동시킨 뒤에 생성되고 서버 프로세스가 정상 종료되면 삭제된다. 노드 매니저에서는 주기적으로 서버의 상태를 체크하면서 해당 파일에 서버의 상태를 업데이트한다.</p> <p>노드 매니저가 재기동되었을 때 해당 파일의 유무에 따라 서버를 계속 모니터링할지 결정하고, 파일에 기록된 서버의 상태를 확인하고 서버의 비정상 종료를 판단해서 노드 매니저가 재기동된 후에도 서버를 재기동하여 계속 모니터링할 수 있다.</p>
<code>&lt;serverName&gt;.address</code>	<p>서버가 기동하면서 생성하는 파일로 서버의 Listen Address와 Listen Port가 기록된다. 이 파일은 서버가 기동될 때 생성되고 정상 종료되면 삭제된다.</p> <p>노드 매니저가 재기동되었을 때 해당 파일의 유무에 따라 서버를 계속 모니터링할지 결정하고, 파일에 기록된 호스트 정보로 서버를 계속 모니터링할 수 있다. 또한 서버로의 MBean 요청을 통해 서버 상태를 주기적으로 모니터링하여 서버의 비정상 종료 여부를 확인한다.</p>

## 2.4. 노드 설정 및 삭제

Java 타입의 노드는 콘솔 툴을 사용해서 설정하고 삭제할 수 있다.

### 2.4.1. Java 타입 노드 설정

본 절에서는 콘솔 툴을 사용해서 Java 타입 노드를 설정하는 방법에 대해 설명한다.



콘솔 툴을 사용해서 서버를 기동하려면 서버에 노드의 이름(Node Name)이 설정되어 있어야 한다.

### 2.4.1.1. 콘솔 툴 사용

다음은 콘솔 툴을 사용해서 노드를 설정하는 방법이다.

1. **add-java-node** 명령어를 사용해서 'node1'이라는 노드를 추가한다.

```
[MASTER]domain1.adminServer>add-java-node node1 -host 192.168.34.65 -port 7730
The node [node1] was successfully added.
```

2. **show-node** 명령어를 사용해서 추가된 노드의 설정을 조회한다. Java 타입의 'node1'이 조회되는 것을 확인할 수 있다.

```
[MASTER]domain1.adminServer>show-node node1
=====
+-----+-----+-----+
| Property | Value |
+-----+-----+-----+
| Node Name | node |
| Host | 192.168.34.65 |
| Mapped Servers |
| Node Type | JAVA |
| NodeManager Port | 7730 |
| Use SSL | false |
+-----+-----+-----+
=====
```

또한 **list-nodes** 명령어를 사용해서 도메인의 노드들을 조회할 수 있다.

```
[MASTER]domain1.adminServer>list-nodes
=====
+-----+-----+-----+-----+
| Node name | Type | Under control | JEUS version |
+-----+-----+-----+-----+
| node1 | JAVA | N | - |
+-----+-----+-----+-----+
=====
```

3. **modify-java-node** 명령어나 **modify-node** 명령어를 사용해서 노드를 변경할 수 있다.

```
[MASTER]domain1.adminServer>modify-java-node node1 -port 7731
The node [node1] was modified successfully. Check the results using "show-node"
[MASTER]domain1.adminServer>modify-node node1 -port 7732
The node [node1] was modified successfully. Check the results using "show-node"
```

4. 변경된 노드 정보를 확인하려면 **show-node** 명령어를 사용해서 해당 노드의 설정을 조회한다.

```
[MASTER]domain1.adminServer>show-node node1
=====
+-----+-----+-----+
=====
```

Property	Value
Node Name	node
Host	192.168.34.65
Mapped Servers	
Node Type	JAVA
NodeManager Port	7732
Use SSL	false



add-java-node 명령어와 list-nodes 명령어에 대한 자세한 사용 방법은 JEUS Reference 안내서의 "노드 관리 관련 명령어"를 참고한다.

## 2.4.2. Java 타입 노드 삭제

본 절에서는 콘솔 툴을 사용하여 Java 타입 노드를 삭제하는 방법에 대해 설명한다.

### 2.4.2.1. 콘솔 툴 사용

다음은 콘솔 툴을 사용해서 노드를 삭제하는 방법이다.

1. remove-node 명령어를 사용해서 'node1'이라는 노드를 삭제한다.

```
[MASTER]domain1.adminServer>remove-node node1
The node [node1] was successfully removed.
```

2. 추가된 노드를 확인하려면 list-nodes 명령어를 사용해서 노드 목록을 조회한다. 노드 목록에 Java 타입의 'node1'이 조회되지 않는 것을 확인할 수 있다.

```
[MASTER]domain1.adminServer>list-nodes
=====
+-----+-----+-----+-----+
| Node name | Type | Under control | JEUS version |
+-----+-----+-----+-----+
(No data available)
=====
```



remove-node 명령어와 list-node 명령어에 대한 자세한 사용 방법은 JEUS Reference 안내서의 "노드 관리 관련 명령어"를 참고한다.

## 2.5. 노드 매니저 시작 및 종료

Java 타입의 노드 매니저는 SSH 타입의 노드 매니저와 달리 별도의 시작 및 종료 프로세스가 필요하다.

## 2.5.1. Java 타입 노드 매니저 시작

Java 타입의 노드 매니저는 스크립트를 통해서 실행시킨다. JEUS\_HOME/bin 하위에 존재하는 **startNodeManager**를 실행시키면 다음과 같이 노드 매니저가 Master나 콘솔 톨로부터 명령어를 받아 서버를 제어할 준비를 한다. startNodeManager -h로 사용할 수 있는 옵션을 확인할 수 있다.

```
JEUS_HOME/bin$ startNodeManager
*****
- JEUS Home      : /home/jeus/jeus9
- Added Java Option :
- Java Vendor    : Sun
*****
...
[2016.07.28 13:55:32][2] [nodemanager-1] [NodeManager-0101] The node manager is starting.
[2016.07.28 13:55:32][2] [nodemanager-1] [NodeManager-0102] Initializing the node manager
configuration.
[2016.07.28 13:55:32][2] [nodemanager-1] [NodeManager-0108] Beginning to listen:
localhost/127.0.0.1:7730.
[2016.07.28 13:55:32][2] [nodemanager-10] [NodeManager-0109] Processing the request.....
```

노드 매니저가 서버를 모니터링하는 중에 종료되어 다시 기동되었다면 이전에 모니터링하던 서버를 계속 모니터링하기 위한 작업을 한다. 이때는 다음과 같은 로그가 발생한다.

```
JEUS_HOME/bin$ startNodeManager
*****
- JEUS Home      : /home/jeus/jeus9
- Added Java Option :
- Java Vendor    : Sun
*****
...
[2016.07.28 13:59:16][2] [nodemanager-1] [NodeManager-0101] The node manager is starting.
[2016.07.28 13:59:16][2] [nodemanager-1] [NodeManager-0102] Initializing the node manager
configuration.
[2016.07.28 13:59:16][2] [nodemanager-1] [NodeManager-0108] Beginning to
listen:localhost/127.0.0.1:7730.
[2016.07.28 13:59:16][2] [nodemanager-1] [NodeManager-0115] Domain=[domain1], Server=[adminServer]
[2016.07.28 13:59:16][2] [nodemanager-10] [NodeManager-0109] Processing the request.....
[2016.07.28 13:59:16][2] [nodemanager-1] [NodeManager-0115] Domain=[domain1], Server=[server1]
[2016.07.28 13:59:16][2] [nodemanager-11] [NodeManager-0137] Beginning to monitor the
server[adminServer] in the domain[domain1].
[2016.07.28 13:59:16][2] [nodemanager-12] [NodeManager-0137] Beginning to monitor the server[server1]
in the domain[domain1].
[2016.07.28 13:59:16][2] [nodemanager-11] [NodeManager-0145] The process is alive(Server=adminServer,
Process ID=4856).
[2016.07.28 13:59:17][2] [nodemanager-12] [NodeManager-0145] The process is alive(Server=server1,
Process ID=5376).
```



노드 매니저를 기동할 때 Master를 같이 기동할 수도 있다. 도메인을 구성하고 처음으로 노드 매니저를 띄우려고 할 때 Master 기동 옵션을 준 경우 해당 노드에 Master를 기동시켜준다.

## 2.5.2. Java 타입 노드 매니저 종료

Java 타입의 노드 매니저를 종료하는 방법은 스크립트를 사용하거나 콘솔 툴을 사용할 수 있다.

### • 스크립트 사용

기동할 때와 마찬가지로 스크립트를 통해 노드 매니저를 종료할 수 있다. **stopNodeManager**를 실행하면 Java 타입 노드 매니저가 종료된다.

```
JEUS_HOME/bin$ stopNodeManager -host localhost -port 7730
*****
- Usage : stopNodeManager -host host -port port
*****
Succeed to stop the node manager.
```

노드 매니저를 종료할 때 **-properties** 옵션을 통해 jeusnm.xml 파일의 경로를 지정할 수 있다. 노드 매니저 설정 파일로부터 host와 port 정보를 읽어 노드 매니저를 종료한다.

```
JEUS_HOME/bin$ stopNodeManager -properties JEUS_HOME/nodemanager/jeusnm.xml
*****
- Usage : stopNodeManager -host host -port port
*****
Succeed to stop the node manager.
```

### • 콘솔 툴 사용

콘솔 툴을 통해 **nm-stop** 명령어를 실행하면 노드 매니저가 종료된다. nm-stop 명령어에 대한 자세한 사용법은 JEUS Reference 안내서의 "stop-nodemanager"를 참고한다.

```
JEUS_HOME/bin$ jeusadmin
JEUS 9 Administration Tool
To view help, use the 'help' command.
offline>nm-stop -host localhost -port 7730
Succeed to stop the node manager.
```

## 2.6. Java 타입 노드 매니저를 통한 서버 제어

Java 타입 노드 매니저를 통해 서버를 제어할 수 있다. 본 절에서는 콘솔 툴을 사용한 서버 제어 방법에 대해 설명한다.

### 2.6.1. 콘솔 툴 사용

노드 매니저가 서버를 기동/종료시키고 서버의 상태를 확인하는 등의 제어는 콘솔 툴을 사용할 수도 있다.

콘솔 툴을 통해 서버를 제어하려면 먼저 콘솔 툴과 노드 매니저가 연결된 상태여야 하고 필요한 작업이 완료되면 접속을 종료한다.

- 노드 매니저에 접속

**nm-connect** 명령어를 사용해서 노드 매니저에 접속한다.

```
JEUS_HOME/bin$ jeusadmin
JEUS 9 Administration Tool
To view help, use the 'help' command.
offline>nm-connect -host localhost -port 7730 -domain domain1
The connection to the node manager domain1 has been established.
[NodeManager]domain1>
```

- 노드 매니저를 통한 서버 기동

**nm-start-server** 명령어를 사용해서 서버를 기동한다.

```
[NodeManager]domain1>nm-start-server -server adminServer -u jeus -p jeus
succeed to start server[adminServer].
RUNNING
[NodeManager]domain1>nm-start-server -server server1 -u jeus -p jeus
-masterurl 192.168.0.4:9736
succeed to start server[server1].
RUNNING
```

다음과 같이 **nm-start-server** 명령어로 노드 매니저에 접속하는 것까지 한 번에 수행할 수도 있다.

```
JEUS_HOME/bin$ jeusadmin
offline>nm-start-server -host localhost -port 7730 -domain domain1 -server adminServer -u jeus -p
jeus
succeed to start server[adminServer].
RUNNING
```

- 스크립트로 노드 매니저의 제어를 받는 Master 기동

Master에 한해서는 노드 매니저를 통해 서버를 기동하지 않아도 노드 매니저와 연결하고 노드 매니저의 제어를 받게 할 수 있다.

**startMasterServerNM** 스크립트를 사용하면 노드 매니저로 연결하고 Master 기동 명령어를 내려 노드 매니저가 Master를 기동시키게 된다. 콘솔 툴에서 노드 매니저로 접속하고 서버를 기동하는 두 번의 명령어를 하나의 스크립트로 제공하는 것이다.



startMasterServer 스크립트를 통해 Master를 기동한 경우에도 노드 매니저의 관리를 받을 수 있다. Master는 기동된 후 노드 매니저에 연결을 시도해서 자신의 정보를 등록한다. 노드 매니저에 Master의 정보가 등록되면 노드 매니저에서 Master의 비정상 상태를 파악하고 재기동하는 등의 제어가 가능해진다.

```
JEUS_HOME/bin$ startMasterServerNM -host 192.168.0.26 -port 7730 -domain domain1 -server
adminServer -u jeus -p jeus
```

```
*****
- Usage : startMasterServerNM -host host -port port -domain domain -server
server1 -u username -p password
*****

succeed to start server[adminServer].
RUNNING
```

## • 노드 매니저를 통한 서버 상태 확인

**nm-state-server** 명령어를 사용해서 서버 상태를 확인한다.

```
[NodeManager]domain1>nm-state-server -server adminServer -u jeus -p jeus
server[adminServer] : RUNNING
[NodeManager]domain1>nm-state-server -server server1 -u jeus -p jeus
server[server1] : RUNNING
```

## • 노드 매니저를 통한 서버 종료

**nm-stop-server** 명령어를 사용해서 서버를 종료한다.

```
[NodeManager]domain1>nm-stop-server -server server1 -u jeus -p jeus
succeed to stop server[server1].
[NodeManager]domain1>nm-stop-server -server adminServer -u jeus -p jeus
succeed to stop server[adminServer].
```

## • 노드 매니저와 접속 종료

**nm-disconnect** 명령어를 사용해서 노드 매니저와 접속을 종료한다.

```
[NodeManager]domain1>nm-disconnect
disconnect to node manager.
offline>
```

## • Master 접속 모드에서 서버 기동

**start-server** 명령어를 통해 MS를 기동할 때 노드 매니저를 통해서 서버를 기동한다.

```
[MASTER]domain1.adminServer>start-server server1
The server [server1] was successfully started.
```



콘솔 툴에서 Java 타입 노드 매니저에 접속/종료 명령어와 Java 타입 노드 매니저를 통한 서버 제어 명령어에 대한 자세한 사용 방법은 JEUS Reference 안내서의 "노드 관리 관련 명령어"를 참고한다.



## 2.7. 로그 파일

노드 매니저의 로그는 JEUS\_HOME/nodemanager/logs 하위에 JeusNodeManager.log라는 파일에 남겨진다. 노드 매니저에서 서버를 기동하거나 모니터링하면서 발생하는 로그 메시지를 기록한다. 또한 서버가 기동되면서 서버에서 발생하는 부팅 로그 메시지도 기록된다.

노드 매니저의 로그 레벨을 설정하려면 노드 매니저를 기동하는 스크립트에 다음과 같이 설정할 수 있다.

```
-Djeus.nodemanager.log.level=FINEST
```



로그와 로그 레벨에 대한 자세한 내용은 JEUS Server 안내서의 "Logging"을 참고한다.

## 3. SSH 타입 노드 매니저

본 장에서는 SSH 타입 노드 매니저를 사용하기 위한 설정 방법에 대해서 설명한다.

### 3.1. 개요

SSH 타입 노드 매니저는 UNIX 계열의 운영체제에서 사용할 수 있는 노드 매니저 타입으로 해당 노드로 SSH(Secure Shell) 명령을 수행하여 동작한다. 노드에 JEUS 서버를 추가할 때 쉽게 설치할 수 있는 기능을 제공하고 Master 머신에서 SSH 서버를 실행해서 로컬 노드를 구성할 수도 있다. 그러나 Windows의 경우에는 SSH 타입 노드 매니저를 지원하지 않는다.



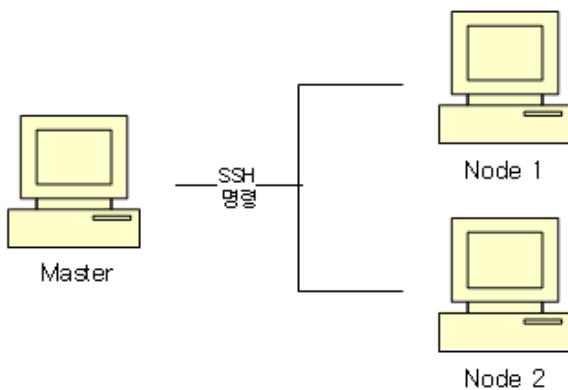
본 장에서는 원격 설정과 로컬(Master 머신) 설정에 대한 혼동을 피하기 위해서 원격 노드, 원격 서버를 기준으로 설명한다.

### 3.2. 환경설정

SSH 타입 노드 매니저를 사용하기 위해서는 몇가지 환경설정이 필요하다.

#### 3.2.1. SSH 설정

UNIX 계열의 운영체제에서 SSH 타입 노드 매니저 기능을 사용하려면 Master 머신에서 노드로 SSH 접속이 가능해야 한다. 즉, Master 머신은 SSH 클라이언트가 되고, 원격 노드는 SSH 서버가 된다. 본 절에서는 SSH 서버의 구동은 별도로 설명하지 않으며, SSH 접속을 위해 필요한 설정들만 설명한다.



원격 노드로의 SSH 접속

JEUS에서 원격 노드로의 SSH 접속은 개인 Key를 이용하는 접속만 허용된다. 개인 Key를 통한 접속을 위해 Master 머신에서 ssh-keygen을 이용하여 공개 Key와 개인 Key를 생성한다. Key를 생성할 때 개인 Key를 암호화하는 passphrase는 JEUS에서 지원하지 않으므로 설정하지 않는다.

생성한 개인 Key와 공개 Key는 USER\_HOME/.ssh 폴더에 "id\_rsa", "id\_rsa.pub"라는 이름의 파일로 저장된다. 공개 Key "id\_rsa.pub"의 내용을 접속하려는 원격 노드의 USER\_HOME/.ssh/authorized\_keys 파일에 추가하면 개인 Key를 이용한 접속이 가능해진다.

다음과 같이 SSH 명령어를 실행해서 개인 Key를 통한 SSH 접속이 가능한지 확인할 수 있다.

```
ssh <ssh-user-name>@<remote-host-address>
```

다음은 SSH 명령어의 예시이다.

```
$ ssh jeus@192.168.23.129
```

SSH 명령어를 사용한 원격 서버 실행을 위해서는 해당 노드에 JAVA\_HOME이 환경변수로 설정되어 있어야 한다. SSH는 명령어를 수행할 때 원격 머신의 USER\_HOME/.bashrc 파일을 통해 설정되는 환경변수를 이용하므로 .bashrc 파일을 다음과 같이 수정한다. JEUS에서의 SSH 명령어 수행은 기본적으로 non-interactive 모드로 동작한다. non-interactive 모드일 경우 설정한 환경변수가 동작하지 않을 수 있으므로 다음과 같이 최상위에 환경변수를 설정한다.

```
# ~/.bashrc

export JAVA_HOME='/home/java/jdk17'
export PATH='/home/java/jdk17/bin':$PATH

# If not running interactively, don't do anything
[ -z "$PS1" ] && returnssh
...
```

Korn shell인 경우에는 USER\_HOME/.kshrc에 위의 내용을 설정한다. 그리고 USER\_HOME/.ssh/environment 파일을 새로 생성해서 ENV=~/.kshrc를 추가한다.

```
$ cat ~/.ssh/environment
ENV=~/.kshrc
```

위의 설정이 제대로 동작하려면 /etc/ssh/sshd\_config 파일의 PermitUserEnvironment를 'yes'로 설정하고 sshd를 재시작한다.

원격 노드에 JAVA\_HOME 환경변수를 설정한 후 Master 머신에서 다음의 명령어를 통해 설정이 제대로 적용되었는지 확인한다.

```
$ ssh jeus@192.168.23.129 java -version
java version "17.0.6" 2023-01-17 LTS
Java(TM) SE Runtime Environment (build 17.0.6+9-LTS-190)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.6+9-LTS-190, mixed mode, sharing)
```

위의 명령어가 정상적으로 동작한다면 SSH를 사용할 준비가 완료된 것이다.

## 3.3. 노드 설정 및 삭제

SSH 타입의 노드는 WebAdmin과 콘솔 툴을 사용해서 설정하고 삭제할 수 있다.

### 3.3.1. SSH 타입 노드 설정

SSH 설정이 완료되면 JEUS에 노드의 설정을 적용해야 한다.

노드는 기본적으로 다음의 3가지 기본 정보가 필요하고, 이 정보와 함께 노드의 종류별로 필요한 설정을 한다.

- ID로 사용할 노드의 이름
- 원격 머신의 주소
- 원격 머신의 JEUS 설치 디렉터리 정보

SSH 타입 노드 매니저는 UNIX 환경에서만 지원 가능하다. SSH 접속을 위해서 포트 번호, 사용자 이름 그리고 개인 Key 파일 경로 정보를 설정해야 한다. 본 절에서는 WebAdmin과 콘솔 툴을 사용하여 JEUS에 SSH 타입 노드를 설정하는 방법을 설명한다.

#### 3.3.1.1. WebAdmin 사용

다음은 WebAdmin을 사용해서 SSH 타입 노드를 설정하는 방법이다.

1. 노드와 관련된 모든 사항은 **[Nodes]** 메뉴를 통해 편집 및 동작이 가능하다. WebAdmin 메인 화면에서 **[Nodes]** 를 선택한다.
2. **Nodes 화면**에는 추가된 노드 목록이 조회된다. 사용자가 노드를 추가하지 않아도 노드 목록에는 JEUS를 설치할 때 기본적으로 생성되는 노드가 조회된다. 노드 목록에서는 노드의 이름, 종류, 제어 가능 여부, 해당 노드에 설치된 JEUS 버전 정보 등을 확인할 수 있다. 노드 추가를 위해 **[추가]** 버튼을 클릭한다.
3. **Nodes 화면**에서 노드의 이름, 호스트 주소, JEUS 설치 경로를 입력한다. 노드의 타입으로 **'Ssh'**를 선택하고 관련 항목을 입력한 뒤 **[추가]** 버튼을 클릭한다.
4. **Nodes 화면**에 노드가 성공적으로 추가되었다는 메시지와 함께 노드 목록에 추가한 노드 정보가 조회된 것을 확인할 수 있다.

원격 노드에 JEUS 설치가 완료되면 **'JEUS version'** 컬럼에서 설치된 JEUS 버전을 확인할 수 있다. JEUS가 설치되지 않은 상태라면 **'JEUS version'** 컬럼에 JEUS 버전이 출력되지 않는다. 제어 가능한 노드인 경우 **[install]** 버튼을 클릭하여 JEUS를 설치할 수 있다.

5. JEUS가 설치된 노드에서 서버를 실행하려면 앞에서 정의한 노드를 해당 서버에 매핑시켜야 한다. 서버 설정 화면의 **'Node Name'** 항목에 노드 이름을 선택해서 서버에 매핑한다.
6. JEUS가 설치된 노드에 매핑된 서버는 화면 왼쪽의 **[Monitoring] > [Servers]** 메뉴를 선택하면 조회되는 서버 목록에서 확인할 수 있다.

조회된 서버 목록에서 원하는 서버를 선택한 후 **[start]** 버튼을 클릭하면 나오는 화면에서 각 항목을 설정한 후 **[확인]** 버튼을 클릭하면 해당 서버가 시작된다.



서버를 기동하는 방법은 **[Monitoring] > [Servers]** 메뉴 외의 **[Servers]** 메뉴를 이용할

수도 있다. 해당 메뉴를 사용한 방법은 [Java 타입 노드 매니저](#)를 참고한다.

7. 원격 노드에 서버 실행이 완료되면 화면 상단에 서버가 시작되었다는 메시지와 함께 **'Status'** 컬럼에 해당 서버의 상태가 'RUNNING'으로 표시된 것을 확인할 수 있다.

### 3.3.1.2. 콘솔 툴 사용

다음은 콘솔 툴을 사용해서 노드를 설정하는 방법이다.

1. **add-ssh-node** 명령어를 사용해서 다음과 같이 JEUS에 SSH 타입 노드를 추가한다.

```
[MASTER]domain1.adminServer>add-ssh-node node2 -host 192.168.23.129
-dir /home/jeus/jeus9 -user jeus -privatekey /home/jeus/.ssh/id_rsa
The node [node2] was successfully added.
```

2. 추가한 SSH 노드가 정상적으로 동작하는지 여부를 **check-ssh-node** 명령어를 통해 확인할 수 있다. 해당 노드에 SSH를 통해 Java 명령어를 수행해보는 명령어로 다음과 같이 Java를 수행할 수 있다고 나타나면 추가한 노드가 정상적으로 동작한다는 의미이다.

```
[MASTER]domain1.adminServer>check-ssh-node node2
The Domain Administration Server can execute the "java" process via SSH.
```

3. 추가된 노드는 nodes.xml에 해당 설정이 저장되며, **show-node** 명령어를 통해 다음과 같이 설정을 확인할 수 있다.

```
[MASTER]domain1.adminServer>show-node node2
=====
+-----+-----+
| Property | Value |
+-----+-----+
| Node name | node2 |
| Host | 192.168.23.129 |
| Mapped servers | |
| Node Type | SSH |
| Installed directory | /home/jeus/jeus9 |
| SSH user name | jeus |
| SSH private key | /home/jeus/.ssh/id_rsa |
| SSH port | 22 |
+-----+-----+
=====
```

4. 이렇게 구성된 노드들은 **list-nodes** 명령어를 통해 노드의 종류, 제어 가능 여부, 설치된 JEUS의 버전 정보를 확인할 수 있다.

```
[MASTER]domain1.adminServer>list-nodes
=====
+-----+-----+-----+-----+
| Node | Type | Control | JEUS Version |
+-----+-----+-----+-----+
```

Node name	Type	Under control	JEUS version
node1	JAVA	N	-
node2	SSH	Y	JEUS 9.1.1

위와 같이 노드가 설정되어 있는 경우 콘솔 툴의 **install-jeus** 명령어를 통해 해당 노드에 JEUS를 설치할 수 있다.

```
[MASTER]domain1.adminServer>install-jeus node2
JEUS was successfully installed on the node [node2].
```



install-jeus 명령어는 Master 머신에 설치된 파일을 복사하는 작업이기 때문에 해당 노드가 Master 머신과 동일한 OS일 경우에만 정상 동작이 가능하고, 다른 OS일 경우 native 라이브러리를 정상적으로 사용할 수 없기 때문에 정상 동작이 불가능하다. 환경에 따라 수행 시간이 오래 걸릴 수 있음에 유의한다.

- 이렇게 설정된 노드에 서버를 매핑해야 해당 서버가 해당 노드에서 동작할 수 있게 된다. **modify-server** 명령어의 -node 옵션을 사용해서 매핑한다.

다음은 특정 서버를 특정 노드에 매핑하는 예시이다.

```
[MASTER]domain1.adminServer>modify-server server1 -node node2
Successfully performed the MODIFY operation for server (server1).
Check the results using "list-servers server1 or modify-server server1"
```

- 다음과 같이 **show-node** 명령어를 사용하여 해당 노드를 확인해 보면 서버가 매핑된 것을 확인할 수 있다.

```
[MASTER]domain1.adminServer>show-node node2
=====
+-----+-----+
| Property | Value |
+-----+-----+
| Node name | node2 |
| Host | 192.168.23.129 |
| Mapped servers | server1 |
| Node Type | SSH |
| Installed directory | /home/jeus/jeus9 |
| SSH user name | jeus |
| SSH private key | /home/jeus/.ssh/id_rsa |
| SSH port | 22 |
+-----+-----+
=====
```

- 이렇게 노드에 매핑된 서버는 콘솔 툴의 **start-server** 명령어를 통해 간편하게 시작 및 종료가 가능하고, 서버를 실행하는 데 필요한 모든 정보는 자동으로 설정된다.

```
[MASTER]domain1.adminServer>start-server server1
The server [server1] was successfully started.
```

만약 서버가 동작하고 있는 노드에서 kill 명령어를 통해 해당 서버를 강제 종료하면, Master는 해당 서버가 FAILED 상태임을 인지하고 자동으로 해당 서버를 재시작한다.

다음과 같이 **server-info** 명령어를 통해 해당 서버의 상태 변화를 확인할 수 있다.

```
[MASTER]domain1.adminServer>server-info
Information about Domain (domain1)
=====
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Server | Status | Node | PID | Clu | Latest | Need | Listen | Running |
|         |        | Name |     | ster| Start Time | to | Ports | Engines |
|         |        |      |     |     | / Shutdown | Restart |      |         |
|         |        |      |     |     | Time      |      |      |         |
+-----+-----+-----+-----+-----+-----+-----+-----+
| admin | RUNNING | N/A | 104 | N/A | 2016-08-23 | false | base-0.0. | jms, |
| Server | (00:11:1 | 36 |  |  | (화) 오후 |  | 0.0:9736 | ejb, web |
| (*) | 7) |  |  |  | 12:44:00 KST |  | http-serv |  |
|         |        |  |  |  |  |  | er-0.0.0.0 |  |
|         |        |  |  |  |  |  | :8088 |  |
|         |        |  |  |  |  |  | jms-0.0.0 |  |
|         |        |  |  |  |  |  | .0:9741 |  |
+-----+-----+-----+-----+-----+-----+-----+-----+
| serve | FAILED( | nod | N/A | N/A | N/A | N/A | N/A | N/A |
| r1 | 00:02:33) | e2 |  |  |  |  |  |  |
+-----+-----+-----+-----+-----+-----+-----+-----+
| serve | SHUTDOWN | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| r2 |  |  |  |  |  |  |  |  |  |
+-----+-----+-----+-----+-----+-----+-----+-----+
=====
```

```
[MASTER]domain1.adminServer>server-info
Information about Domain (domain1)
=====
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Server | Status | Node | PID | Clu | Latest | Need | Listen | Running |
|         |        | Name |     | ster| Start Time | to | Ports | Engines |
|         |        |      |     |     | / Shutdown | Restart |      |         |
|         |        |      |     |     | Time      |      |      |         |
+-----+-----+-----+-----+-----+-----+-----+-----+
| admin | RUNNING | N/A | 104 | N/A | 2016-08-23 | false | base-0.0. | jms, |
| Server | (00:14:4 | 36 |  |  | (화) 오후 |  | 0.0:9736 | ejb, web |
| (*) | 8) |  |  |  | 12:44:00 KST |  | http-serv |  |
|         |        |  |  |  |  |  | er-0.0.0.0 |  |
|         |        |  |  |  |  |  | :8088 |  |
|         |        |  |  |  |  |  | jms-0.0.0 |  |
|         |        |  |  |  |  |  | .0:9741 |  |
+-----+-----+-----+-----+-----+-----+-----+-----+
| serve | RUNNING | nod | 6516 | N/A | 2016-08-23 | false | BASE-0.0. | jms, |
| r1 | (00:00:1 | e2 |  |  | (화) 오후 |  | 0.0:9836 | ejb, web |
|         | 2) |  |  |  | 12:58:36 KST |  |  |  |
+-----+-----+-----+-----+-----+-----+-----+-----+
=====
```

serve	SHUTDOWN	N/A	N/A	N/A	N/A	N/A	N/A	N/A
r2								



본 절에서 사용되는 각 명령어에 대한 자세한 사용 방법은 JEUS Reference 안내서의 "Server Management 관련 명령어", "Domain Configuration 관련 명령어", "노드 매니저 관련 명령어"를 참고한다.

### 3.3.2. SSH 타입 노드 변경

SSH 타입 노드는 WebAdmin이나 콘솔 툴을 사용해서 변경할 수 있다.

1. **modify-ssh-node** 명령어나 **modify-node** 명령어를 사용해서 노드를 변경할 수 있다.

```
[MASTER]domain1.adminServer>modify-ssh-node node2 -host 192.168.23.128
The node [node2] was modified successfully. Check the results using "show-node"
[MASTER]domain1.adminServer>modify-node node2 -host 192.168.23.127
The node [node2] was modified successfully. Check the results using "show-node"
```

2. 변경된 노드 정보를 확인하려면 **show-node** 명령어를 사용해서 해당 노드의 설정을 조회한다.

```
[MASTER]domain1.adminServer>show-node node2
=====
+-----+-----+
| Property | Value |
+-----+-----+
| Node Name | node2 |
| Host | 192.168.23.127 |
| Mapped Servers | server2 |
| Node Type | SSH |
| Installed directory | /home/jeus/jeus9 |
| SSH User Name | jeus |
| SSH Private Key | /home/jeus/.ssh/id_rsa |
| SSH Port | 22 |
+-----+-----+
=====
```

### 3.3.3. SSH 타입 노드 삭제

SSH 타입 노드는 WebAdmin이나 콘솔 툴을 사용해서 삭제할 수 있다. 삭제 방법은 Java 타입 노드의 방법과 동일하므로 본 절에서는 설명을 생략한다. 자세한 내용은 [Java 타입 노드 삭제](#)를 참고한다.



## 4. RQS 프로세스 관리

본 장에서는 RQS(Reliable Queue Server)를 관리할 수 있는 노드 매니저의 기능에 대해서 설명한다.

### 4.1. 개요

노드 매니저는 서버 이외의 프로세스를 관리하는 기능도 제공한다. 현재는 RQS(Reliable Queue Server) 프로세스 관리를 지원하며 추후 추가될 수 있다. RQS 관리는 RQS 프로세스를 구동하거나 정지시키는 작업을 의미한다. 또한 RQS 프로세스가 죽었는지 살았는지를 체크하여, 제대로 종료되지 않았다면 프로세스를 다시 구동시키는 것이다. 이런 기능은 서버나 노드의 설정과는 상관이 없으며, 별도로 설정된 정보를 통하여 동작하게 된다.



RQS 프로세스를 관리하는 노드 매니저는 같은 머신, 즉 같은 노드에 있는 노드 매니저가 관리한다. 다른 머신에 있는 RQS 프로세스나 노드 매니저가 실행되지 않고 있는 RQS 프로세스는 관리할 수 없다.

### 4.2. RQS의 동작 방식

본 절에서는 노드 매니저가 RQS를 관리하는 방식에 대해서 설명한다.

#### 4.2.1. RQS 프로세스의 구동과 정지

RQS 프로세스들은 설정 파일에 저장된 file path와 옵션들을 통해서 구동된다. RQS는 서버와는 독립적인 프로세스들이기 때문에 구동과 정지도 서버와는 별개로 작동한다. 노드 매니저는 구동할 때 RQS 관련 설정 정보를 읽고, 그에 따라 RQS 프로세스 모니터링을 시작한다. 이때 RQS 프로세스가 구동되어 있지 않으면 구동한다.

1. jeusnm.xml 파일에서 RQS 정보를 읽어들이어 저장한다.
2. RQS 정보를 이용하여 RQS 관리 객체를 생성하고, 모니터용 스레드를 생성한다.
3. 모니터링 스레드는 읽어들이는 정보를 이용하여 프로세스에 접근한 후에 모니터링을 시작한다.
4. 모니터링을 하여 RQS 프로세스가 비정상 종료되거나 응답하지 않을 경우 RQS 프로세스를 재시작한다.



프로세스를 재기동할 경우에도 동일한 과정으로 진행된다.

#### 4.2.2. RQS 프로세스 모니터링

RQS 프로세스의 모니터링에는 별도의 파일을 생성하지 않는다. 프로세스의 상태는 모니터링 스레드와 RQS 프로세스가 주고받는 메시지에 의존한다. 프로세스에 접근할 때 사전에 정의된 메시지를 통하여 접근하게 되고, 프로세스의 정상 동작 여부도 주기적으로 주고받는 메시지에 의존한다. 또한 RQS 프로세스가 정상 종료되는 것을 판단할 때도 종료 메시지를 주고받는 것으로 판단할 수 있다.

노드 매니저는 주기적으로 주고받는 메시지를 통하여 RQS 프로세스의 상태를 체크한다. 만약 이 상태 체크 메시지에 대한 응답이 오지 않는다면 몇 번의 메시지를 더 보내고, 그래도 응답이 없다면 비정상 종료되었다고 판단한다.

### 4.2.3. 비정상 상태의 프로세스 재기동

노드 매니저가 주기적으로 보낸 메시지에 수차례 응답이 오지 않아 RQS 프로세스가 비정상 상태로 종료되었다고 판단하였을 경우에는 프로세스의 재기동 절차를 밟게 된다. 프로세스 재기동은 먼저 기존의 RQS들을 확실히 죽이고 다시 시작하게 되는데, 다시 시작한 후에는 또 다시 모니터링 스레드를 생성하여 다시 시작한 프로세스도 모니터링할 수 있도록 한다.

## 4.3. 환경설정

노드 매니저가 RQS 프로세스를 관리하기 위해서는 여러 가지 설정이 필요하다. RQS 프로세스는 다른 서버나 매니저와 별도로 실행되는 프로세스이기 때문에 이 설정을 통하여 구동되고 관리된다.

### 4.3.1. jeusnm.xml

RQS 프로세스는 서버나 기타 시스템과는 별도로 돌아가는 시스템이기 때문에 RQS 구동은 전적으로 jeusnm.xml 파일의 설정에 의존한다. RQS 프로세스의 생성이나 삭제 역시 jeusnm.xml 파일을 직접 수정하여 추가 및 삭제할 수 있다.

jeusnm.xml 항목에서 rqsList 부분이 RQS 프로세스에 대한 정보를 다루고 있다. processList의 하위 항목으로 RQS라는 항목을 두어서 각각의 RQS 프로세스에 대한 설정 값을 저장함으로써 노드 매니저가 프로세스를 관리하도록 할 수 있다.

- processList

RQS 프로세스들에 대한 정보를 담고 있다.

processList 항목 아래에는 하나의 프로세스 당 하나의 RQS 항목으로 묶여서 설정된다. 여러 개의 프로세스를 관리하고 싶으면 RQS 항목을 여러 개 추가하는 것으로 설정 가능하다.

- RQS

다음은 각 RQS 설정 항목에 대한 설명이다. (\*: 필수 항목)

항목	설명
domainName *	RQS 프로세스들의 그룹 이름을 설정한다.  여러 프로세스들은 domainName이라는 가상의 이름을 가진 그룹에 포함된다.
processName *	RQS 프로세스의 이름을 설정한다.  RQS 프로세스의 시작이나 정지 명령을 내릴 때 사용한다.

항목	설명
path *	RQS 프로세스의 실행 파일 위치를 설정한다.  프로세스가 실행될 때 이 path에 설정된 파일을 실행한다.
port *	RQS 프로세스의 모니터링에 사용될 포트 번호를 설정한다.  RQS 프로세스의 생존 여부를 알기 위한 메시지 교환용 포트이다.
option	RQS를 실행할 때 필요한 argument를 설정한다. RQS 매뉴얼을 참조하여 필요한 기능을 설정한다.
rqmdir	여러 개의 RQS를 띄워야 할 때에는 각자 다른 설정 파일을 가지고 실행된다.  각자 다른 설정 파일을 인식시키는 방법은 RQSDIR이라는 환경변수를 주는 방법이 있는데, 여기에 값을 설정해 주면 이를 인식하여 설정 파일에 접근할 수 있다.
retryCount	접속 실패 시 재접속을 시도하는 횟수를 설정한다. (기본값: 5)
monitoringPeriod	모니터링 접속 시도 시 각 접속 시도 간 시간 간격을 설정한다. (기본값: 500)

## 5. 노드 매니저 이중화

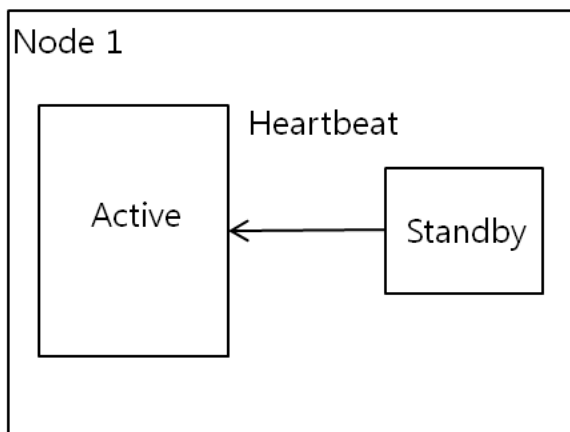
본 장에서는 노드 매니저가 이상 종료되는 경우 노드 매니저의 복구를 가능하게 하는 이중화 기능을 설명한다.

### 5.1. 개요

노드 매니저는 서버의 동작과 종료를 도와주고, 서버에 장애가 발생할 때에는 이를 체크하여 재구동시키는 역할을 한다. 하지만 정작 노드 매니저에 이상이 발생하여 종료되는 상황에는 취약할 수 있다. 그렇기에 노드 매니저 이중화 기능을 추가하여 노드 매니저가 이상 종료되었을 경우에 미리 실행하여 대기시켜 둔 노드 매니저가 대신 일을 계속 하도록 할 수 있다. 본 절에서는 이에 대한 설명과 주의점을 다룬다.

### 5.2. 이중화된 노드 매니저 동작 방식

이중화된 노드 매니저는 Active와 Standby 두 가지 형태로 동작하게 된다. 이 중 Active 노드 매니저만이 실제 Master나 서버들의 요청을 처리하고 상태를 관리하게 된다.



이중화된 노드 매니저의 동작 방식

Standby 노드 매니저는 평상시에는 Active 노드 매니저의 상태만을 체크한다. Active 노드 매니저의 상태 체크에서 이상을 감지하면 Active가 이상 종료되었다고 판단하여 Standby가 Active의 역할을 대신하게 된다. 또한 이 Standby 노드 매니저를 감시하기 위한 또 하나의 Standby 노드 매니저가 실행된다.

정리하면, 이중화된 노드 매니저의 동작 과정은 다음과 같다.

1. Active 노드 매니저가 시작되고 그와 동시에 Standby 노드 매니저를 시작한다.
2. Standby 노드 매니저는 지속적으로 Active 노드 매니저의 상태를 감시한다.
3. Standby 노드 매니저가 Active 노드 매니저의 이상을 감지한다.
4. Standby 노드 매니저는 또 다른 Standby 노드 매니저를 준비한다.
5. 기존의 Standby였던 노드 매니저가 Active가 되어 서버로부터의 요청을 처리한다.

이때 상태 체크에 사용하는 포트는 설정 파일에 설정된 포트로 통신하게 된다. 만약 이 값이 설정되지 않았다면 노드 매니저 이중화 기능을 사용하지 않는 것으로 간주한다.



Standby가 Active 노드 매니저의 이상 종료를 체크하여 Active로 될 때 Standby 노드 매니저를 추가로 실행하여 자신을 감시하게 한다. Active 노드 매니저가 이상 종료될 때마다 이는 반복적으로 일어난다.

## 5.3. 이중화된 노드 매니저 사용 방법

노드 매니저 이중화를 사용하기 위해서는 Standby 노드 매니저와 Active 노드 매니저가 상태를 주고받는 포트를 설정해 줄 필요가 있다. 이 포트는 [설정 파일](#)에서 설명한 것과 같이 standbyPort 항목을 설정함으로써 사용할 수 있다. 설정된 포트를 통하여 Active와 Standby 노드 매니저가 통신을 한다.

이중화를 사용하지 않으려면 standbyPort를 설정하지 않으면 된다. 이 값이 비어 있다면 노드 매니저는 이중화 기능을 사용하지 않는 것으로 간주하여 Standby 노드 매니저를 실행하지 않는다.

### 5.3.1. Standby 노드 매니저

Standby 노드 매니저는 현재의 Active 역할을 하는 노드 매니저의 구동 과정에서 실행된다. 이때 실행되는 노드 매니저는 Standby 모드로 실행되며, Active의 상태를 지속적으로 체크하게 된다.

```
[nodemanager-1] [NodeManager-0201] The standby node manager is starting.
[nodemanager-1] [NodeManager-0102] Initializing the node manager configuration.
```

또한 Standby 노드 매니저는 실제로 서버의 요청을 받아서 처리하지는 않기 때문에 별도의 로그 파일에 정보를 저장하게 된다. 이 별도의 로그는 노드 매니저 로그 파일과 같은 위치에 생성되며, 노드 매니저 이름 뒤에 '\_standby'라는 문자열이 추가된 로그에 저장된다. 이 로그는 Standby 노드 매니저들만이 사용하는 로그이다.

Standby 노드 매니저가 Active가 될 때에는 Standby용 로그에 Active로 전환된다는 내용을 남긴 다음 Active용 로그를 사용하도록 전환된다. 그 후에 새롭게 생성된 Standby 노드 매니저가 다시 Standby용의 로그를 사용하게 된다. 그렇기 때문에 실제 서버의 요청이나 서버를 관리하는 로그는 Active용 로그에만 남게 된다. Standby 로그에는 Standby 노드 매니저의 구동과 Active 노드와의 통신만이 남게 된다.

Standby 노드 매니저는 Active 노드 매니저와 처음 접속할 때 Active의 PID를 받아온 후 이를 로그로 남긴다. 이 PID를 보고 차후에 어떤 프로세스가 Active인지 알 수 있다.

## 5.4. 이중화 상태에서 노드 매니저 종료

이중화된 노드 매니저는 Active 노드 매니저가 이상 종료될 때 Standby 노드 매니저가 이를 체크하여 노드 매니저가 계속 떠 있을 수 있도록 한다. 그렇기 때문에 노드 매니저를 종료할 때에는 stopNodeManager 스크립트를 이용하여 종료해야 한다. stopNodeManager 스크립트는 Active 노드 매니저로 하여금 Standby 노드 매니저에 종료 메시지를 보내게 한다. 이 메시지를 받는 Standby 노드 매니저는 접속을 정리하고 로그에 기록하는 과정까지 진행한 후에 안전하게 종료된다.



문제가 발생하여 노드 매니저를 강제 종료시키게 된다면, 지속적으로 Standby 노드 매니저가

구동되기 때문에 완벽히 종료되지 않을 수 있다. 이런 경우에는 Standby 노드 매니저를 먼저 종료시킨 후에 Active를 종료시킨다. 다만 강제 종료하게 되면 로그가 제대로 남지 않거나 다른 문제가 있을 수 있으니 가능한 stopNodeManager 스크립트와 같은 방법을 사용하도록 한다.