

오픈나루(주) Project name

JBoss EAP 설치 보고서



openmaru
APM

2019-12-27

오픈나루(주)

Table of Contents

Table of Contents.....	ii
Revision History.....	v
1. 개요.....	1
1.1 수행자 정보.....	1
1.2 고객 정보.....	1
2. 설치 인스턴스 정보 요약.....	2
3. 시스템 환경.....	2
3.1 운영체제 정보.....	2
3.1.1 서버 정보 요약.....	2
3.1.2 서버 정보 : «\$property.host.trim()»(«\$property.ip.trim()»).....	5
3.2 인스턴스 구성정보.....	7
3.3 인스턴스 접속 정보.....	8
4. JBoss EAP 설정 정보.....	10
4.1 인스턴스 구성.....	10
4.1.1 스탠드얼론 모드 디렉터리 구성.....	10
4.1.2 인스턴스 이름 규칙.....	10
4.2 JBoss 운영 스크립트.....	12
4.2.1 스탠드얼론 모드 스크립트.....	12
4.3 환경 설정 파일.....	12
5. 운영체제 환경 설정.....	20
5.1 커널 파라미터.....	20
5.2 적용한 커널 파라미터 값.....	21
5.3 사용자 limit 값 설정.....	22
6. JBoss EAP 주요 설정값.....	23
6.1 JDK 설치.....	23
6.2 Native 모듈 설치.....	24
6.2.1 추가 설치 패키지.....	24
6.2.2 web subsystem native 사용 설정.....	24
6.3 Thread Pool 설정.....	25

6.3.1	스레드 풀 설정.....	25
6.3.2	배포 설정.....	25
6.3.3	인코딩 설정.....	26
6.4	데이터소스 자동 설정내역.....	26
6.5	데이터소스 연결 테스트 방법.....	27
6.5.1	데이터소스 연결 테스트 방법.....	27
6.6	데이터소스 설정 방법.....	28
6.6.1	JDBC 드라이버를 모듈로 설정하는 방법.....	29
6.6.2	JDBC 드라이버를 애플리케이션으로 설정하는 방법.....	30
6.7	모듈 등록 및 사용방법.....	35
6.7.1	추가 모듈 디렉토리 설정.....	35
6.8	애플리케이션에서 필요한 모듈 로딩 방법.....	35
6.9	애플리케이션에서 필요하지 않은 모듈 제외방법.....	36
6.10	JDK 클래스 파일들에 접근하는 방법.....	37
6.11	Jersey 사용시 설정.....	38
6.12	JBoss EAP 주요 설정 항목.....	38
6.12.1	애플리케이션 log4j 사용시 설정 방법.....	38
6.12.2	인코딩 설정.....	38
6.12.3	TCP 클러스터링 방법.....	39
7.	JBoss 운영 방법.....	40
7.1	JBoss Admin Console.....	41
7.2	JBoss CLI 실행.....	42
8.	애플리케이션 배포 방법.....	43
8.1	웹 관리 콘솔을 이용하는 방법.....	43
8.2	JBoss CLI 를 이용하는 방법.....	44
8.3	Deployment fs-exploded(수동 디렉터리)를 이용하는 방법.....	44
8.4	Deployment Scanner 를 이용하는 방법.....	44
8.4.1	example.war 압축파일 배포 방법.....	44
8.4.2	example.war 디렉토리 배포 방법.....	44
8.5	Maven Plugin 을 이용하는 방법.....	45
8.5.1	maven deploy 방법.....	45
8.5.2	maven undeploy 방법.....	45
8.6	JSP 변경 체크 설정.....	45

8.6.1	JSP development mode 변경.....	45
8.6.2	JSP 변경 체크시간을 5 초로 변경	46
8.7	웹 애플리케이션 세션복제 설정	46
8.7.1	web.xml.....	46
8.7.2	세션 복제를 테스트하기 위한 JSP 파일.....	46
8.8	웹 세션 복제 테스트 애플리케이션	47
9.	모니터링 방법.....	48
9.1	MBean 그래프 모니터링.....	48
9.1.1	hawtio Chrome Extension 설치.....	48
9.1.2	hawtio 실행하기.....	49
9.1.3	MBean 모니터링	50
9.2	JMX 모니터링 스크립트.....	52
9.3	KHAN [apm] 모니터링	53
10.	도움이 필요하십니까?	55
11.	References	56

Revision History

Name	Date	Reason For Changes	Version
오픈나루 (service@openmaru.com)	2014/1/22	Initial Version	1.0

1. 개요

1.1 수행자 정보

본 문서는 오픈나루(opennaru.com)의 자동 설치 제품인 OPENMARU Installer 이용하여 생성된 문서입니다. 웹 / WAS 미들웨어 자동 설치 제품에 대한 문의는 support@opennaru.com 으로 하시면 됩니다.

설치한 제품 및 설치 지원 회사의 정보는 다음과 같습니다.

항목	내용
설치 제품	JBoss EAP
제품 버전	7.2.0
수행 일시	2019-12-27_12-26-40
설치지원 회사명	오픈나루
수행자	
이메일	support@opennaru.com
전화번호	02-469-5426

1.2 고객 정보

구분	내용
고객사	오픈나루(주)
Subscription #	
프로젝트명	Project name
담당자	한상진

고객 연락처	hansj@openmaru.com(010-4507-2165)
수행시간	2019-12-27_12-26-40

2. 설치 인스턴스 정보 요약

호스트 IP	인스턴스명	포트 오프셋	프로파일
192.168.182.138	admin11	100	standalone-ha.xml
192.168.182.138	front11	200	standalone-ha.xml
192.168.182.139	admin21	100	standalone-ha.xml
192.168.182.139	front21	200	standalone-ha.xml

이후 시스템의 운영 중 발생하는 문제에 대해서는 “한국 레드햇 고객지원 서비스”의 전화나 고객지원 포털을 통해서 기술지원을 받으실 수 있습니다.

- 고객지원 포털 : <http://access.redhat.com>
- 기술지원 전화 : **080-081-0880**

3. 시스템 환경

설치한 시스템 기본환경에 대한 정보입니다.

3.1 운영체제 정보

3.1.1 서버 정보 요약

서버	정보
----	----

sm2 (192.168.182.139)	운영체제	x86_64
	메모리	7990140 KB
	CPU	4 개
	Core	4 개
sm1 (192.168.182.138)	운영체제	x86_64
	메모리	7990140 KB
	CPU	4 개
	Core	4 개

3.1.2 서버 정보 : sm2(192.168.182.139)

구분	정보																		
호스트 이름	sm2																		
IP 주소	192.168.182.139																		
OS 버전																			
Kernel 버전	3.10.0-1062.el7.x86_64																		
아키텍처(bit 수)	x86_64																		
CPU 정보	Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz																		
CPU 개수	4																		
Core 개수	4																		
CPU 당 Core 수	1																		
메모리(KB)	7990140 KB																		
Disk 사용량	<table border="1"> <thead> <tr> <th>Filesystem</th> <th>Size</th> <th>Used</th> <th>Avail</th> <th>Use%</th> <th>Mounted on</th> </tr> </thead> <tbody> <tr> <td>devtmpfs</td> <td>3.8G</td> <td>0</td> <td>3.8G</td> <td>0%</td> <td>/dev</td> </tr> <tr> <td>tmpfs</td> <td>3.9G</td> <td>0</td> <td>3.9G</td> <td>0%</td> <td>/dev/shm</td> </tr> </tbody> </table>	Filesystem	Size	Used	Avail	Use%	Mounted on	devtmpfs	3.8G	0	3.8G	0%	/dev	tmpfs	3.9G	0	3.9G	0%	/dev/shm
Filesystem	Size	Used	Avail	Use%	Mounted on														
devtmpfs	3.8G	0	3.8G	0%	/dev														
tmpfs	3.9G	0	3.9G	0%	/dev/shm														

	<pre>tmpfs 3.9G 12M 3.8G 1% /run tmpfs 3.9G 0 3.9G 0% /sys/fs/cgroup /dev/mapper/rhel-root 17G 1.8G 16G 11% / /dev/sda1 1014M 150M 865M 15% /boot tmpfs 781M 0 781M 0% /run/user/1104 tmpfs 781M 0 781M 0% /run/user/0</pre>
Disk 정보	<p>Disk /dev/sda: 21.5 GB, 21474836480 bytes, 41943040 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes Disk label type: dos Disk identifier: 0x00008747</p> <pre>Device Boot Start End Blocks Id System /dev/sda1 * 2048 2099199 1048576 83 Linux /dev/sda2 2099200 41943039 19921920 8e Linux LVM</pre> <p>Disk /dev/mapper/rhel-root: 18.2 GB, 18249416704 bytes, 35643392 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes</p> <p>Disk /dev/mapper/rhel-swap: 2147 MB, 2147483648 bytes, 4194304 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes</p>
네트워크 설정	<pre>ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.182.139 netmask 255.255.255.0 broadcast 192.168.182.255 ether 00:0c:29:69:42:c8 txqueuelen 1000 (Ethernet) RX packets 1307 bytes 274082 (267.6 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 651 bytes 67172 (65.5 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0</pre>

	<pre>lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 loop txqueuelen 1000 (Local Loopback) RX packets 2803 bytes 327315 (319.6 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 2803 bytes 327315 (319.6 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0</pre>
라우팅 정보	<pre>Kernel IP routing table Destination Gateway Genmask Flags MSS Window irtt Iface 0.0.0.0 192.168.182.2 0.0.0.0 UG 0 0 0 ens33 192.168.182.0 0.0.0.0 255.255.255.0 U 0 0 0 ens33</pre>

3.1.3 서버 정보 : sm1(192.168.182.138)

구분	정보
호스트 이름	sm1
IP 주소	192.168.182.138
OS 버전	
Kernel 버전	3.10.0-1062.el7.x86_64
아키텍처(bit 수)	x86_64
CPU 정보	Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz
CPU 개수	4
Core 개수	4
CPU 당 Core 수	1
메모리(KB)	7990140 KB
Disk 사용량	<pre>Filesystem Size Used Avail Use% Mounted on devtmpfs 3.8G 0 3.8G 0% /dev tmpfs 3.9G 148K 3.9G 1% /dev/shm</pre>

	<pre>tmpfs 3.9G 12M 3.8G 1% /run tmpfs 3.9G 0 3.9G 0% /sys/fs/cgroup /dev/mapper/rhel-root 17G 3.6G 14G 21% / /dev/sda1 1014M 150M 865M 15% /boot tmpfs 781M 0 781M 0% /run/user/0</pre>																					
Disk 정보	<p>Disk /dev/sda: 21.5 GB, 21474836480 bytes, 41943040 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes Disk label type: dos Disk identifier: 0x0007f37f</p> <table border="1"> <thead> <tr> <th>Device</th> <th>Boot</th> <th>Start</th> <th>End</th> <th>Blocks</th> <th>Id</th> <th>System</th> </tr> </thead> <tbody> <tr> <td>/dev/sda1</td> <td>*</td> <td>2048</td> <td>2099199</td> <td>1048576</td> <td>83</td> <td>Linux</td> </tr> <tr> <td>/dev/sda2</td> <td></td> <td>2099200</td> <td>41943039</td> <td>19921920</td> <td>8e</td> <td>Linux LVM</td> </tr> </tbody> </table> <p>Disk /dev/mapper/rhel-root: 18.2 GB, 18249416704 bytes, 35643392 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes</p> <p>Disk /dev/mapper/rhel-swap: 2147 MB, 2147483648 bytes, 4194304 sectors Units = sectors of 1 * 512 = 512 bytes Sector size (logical/physical): 512 bytes / 512 bytes I/O size (minimum/optimal): 512 bytes / 512 bytes</p>	Device	Boot	Start	End	Blocks	Id	System	/dev/sda1	*	2048	2099199	1048576	83	Linux	/dev/sda2		2099200	41943039	19921920	8e	Linux LVM
Device	Boot	Start	End	Blocks	Id	System																
/dev/sda1	*	2048	2099199	1048576	83	Linux																
/dev/sda2		2099200	41943039	19921920	8e	Linux LVM																
네트워크 설정	<pre>ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.182.138 netmask 255.255.255.0 broadcast 192.168.182.255 ether 00:0c:29:9e:29:16 txqueuelen 1000 (Ethernet) RX packets 464628 bytes 641658183 (611.9 MiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 50626 bytes 10714945 (10.2 MiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0</pre>																					

	<pre> lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 loop txqueuelen 1000 (Local Loopback) RX packets 390 bytes 211862 (206.8 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 390 bytes 211862 (206.8 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 </pre>
라우팅 정보	<pre> Kernel IP routing table Destination Gateway Genmask Flags MSS Window irtt Iface 0.0.0.0 192.168.182.2 0.0.0.0 UG 0 0 0 ens33 192.168.182.0 0.0.0.0 255.255.255.0 U 0 0 0 ens33 </pre>

3.2 인스턴스 구성정보

호스트 IP	인스턴스명	포트 정보	
192.168.182.138	admin11	프로파일	standalone-ha.xml
		HTTP 포트	8180
		AJP 포트	8109
		멀티캐스트 IP	230.10.1.1
		JMS 멀티캐스트 IP	231.70.1.1
		mod cluster 멀티캐스트 IP	224.1.1.105
		VirtualHost 도메인명	admin.openmaru.co.kr
192.168.182.138	front11	프로파일	standalone-ha.xml
		HTTP 포트	8280
		AJP 포트	8209

		멀티캐스트 IP	230.10.2.1
		JMS 멀티캐스트 IP	231.70.2.1
		mod cluster 멀티캐스트 IP	224.1.1.105
		VirtualHost 도메인명	front.opennaru.co.kr
192.168.182.139	admin21	프로파일	standalone-ha.xml
		HTTP 포트	8180
		AJP 포트	8109
		멀티캐스트 IP	230.10.1.1
		JMS 멀티캐스트 IP	231.70.1.1
		mod cluster 멀티캐스트 IP	224.1.1.105
		VirtualHost 도메인명	admin.opennaru.co.kr
192.168.182.139	front21	프로파일	standalone-ha.xml
		HTTP 포트	8280
		AJP 포트	8209
		멀티캐스트 IP	230.10.2.1
		JMS 멀티캐스트 IP	231.70.2.1
		mod cluster 멀티캐스트 IP	224.1.1.105
		VirtualHost 도메인명	front.opennaru.co.kr

3.3 인스턴스 접속 정보

호스트 IP	인스턴스명	포트 정보	
192.168.182.13 8	admin11	관리 콘솔	http://192.168.182.138:10090/
		세션 테스트	http://192.168.182.138:8180/session
		데이터소스 테스트	http://192.168.182.138:8180/session/dstest.jsp
		테스트 애플리케이션	http://192.168.182.138:8180/
192.168.182.13 8	front11	관리 콘솔	http://192.168.182.138:10190/
		세션 테스트	http://192.168.182.138:8280/session
		데이터소스 테스트	http://192.168.182.138:8280/session/dstest.jsp
		테스트 애플리케이션	http://192.168.182.138:8280/
192.168.182.13 9	admin21	관리 콘솔	http://192.168.182.139:10090/
		세션 테스트	http://192.168.182.139:8180/session
		데이터소스 테스트	http://192.168.182.139:8180/session/dstest.jsp
		테스트 애플리케이션	http://192.168.182.139:8180/
192.168.182.13 9	front21	관리 콘솔	http://192.168.182.139:10190/
		세션 테스트	http://192.168.182.139:8280/session
		데이터소스 테스트	http://192.168.182.139:8280/session/dstest.jsp
		테스트 애플리케이션	http://192.168.182.139:8280/

4. JBoss EAP 설정 정보

4.1 인스턴스 구성

4.1.1 스탠드얼론 모드 디렉터리 구성

스탠드얼론 모드 JBoss 인스턴스 구성	
설치 디렉토리	/app/was
도메인 디렉토리	/app/was/domains
서버 인스턴스	/app/was/domains/\$SERVER_NAME
애플리케이션	/app/was/domains/applications/\$GROUP_NAME
Log 디렉토리	/app/logs/was

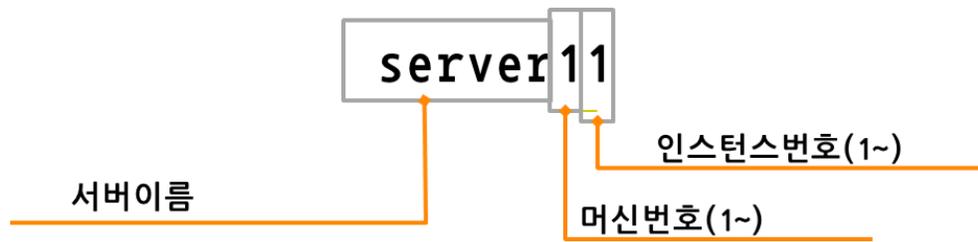
4.1.2 인스턴스 이름 규칙

\$SERVER_NAME 명명 규칙은 아래와 같은 Rule 을 사용하여 구성한다.

/app/was/domains 이라는 디렉터리 하위에 인스턴스 들을 구성한다. domains 하위에 서버 인스턴스 이름으로 디렉터리를 만든다. 인스턴스의 이름은 다음 그림과 같이 서버의 이름 + 머신번호 + 인스턴스 번호와 같은 규칙을 사용하는 것이 좋다.

실제 서버의 이름은 서비스를 나타내는 고유명칭으로 변경하여 사용하면 된다. 머신 번호는 JBoss 에 사용할 머신에 1 번부터 차례대로 붙인 번호를 사용하고, 인스턴스 번호는 같은 머신 내의 인스턴스 개수를 1 번부터 번호를 붙인다.

이렇게 인스턴스 이름을 지정하면 모든 인스턴스가 고유한 이름을 갖게 된다. 이 고유한 인스턴스 이름은 클러스터링에서 세션 ID, 로그 파일의 이름, nohup 로그 파일의 이름 등에 사용하게 된다. 장애 상황이나 로그 파일을 확인하는 등 어떤 상황에서도 서버 이름만 보면 어떤 머신의 어떤 인스턴스의 것인지 금방 구분할 수 있게 된다.



또, 포트 오프셋 번호는 '인스턴스의 번호 * 100'을 사용하면, 포트를 구분하기 쉽다. 기본 포트셋 + 포트오프셋으로 사용하는 포트가 지정되는데, 사용하는 대부분의 포트 번호가 100 번대가 0 이기 때문에 해당 인스턴스가 사용하는 번호를 기억하기 쉽다.

예를 들어 인스턴스 이름이 'frontSvr24'이라면, 2 번째 JBoss 머신의 4 번째 인스턴스를 의미하며, 이 인스턴스가 사용하는 HTTP 포트는 8480(8080 + 400) 포트를 사용하고, AJP 포트는 8409(8009 + 400) 포트를 사용한다.

4.2 JBoss 운영 스크립트

4.2.1 스탠드얼론 모드 스크립트

JBoss 인스턴스 구성	
운영 스크립트 위치	/app/was/domains/\$SERVER_NAME/*.sh

4.3 환경 설정 파일

4.3.1.1 주요 설정 파일

스크립트 파일	
env.sh	JBoss 운영환경 주요 환경 설정 스크립트
start.sh	JBoss Standalone 인스턴스 실행 스크립트
shutdown.sh	JBoss Standalone 인스턴스 정상 종료 스크립트
kill.sh	JBoss 강제 종료 스크립트
nohup.sh	nohup 로그의 tail 보기 스크립트
tail.sh	JBoss server.log 파일 tail 보기 스크립트
status.sh	JBoss 인스턴스가 실행중인지 체크하는 스크립트
jboss-cli.sh	JBoss CLI 모드로 접속하는 스크립트
jconsole.sh	JConsole 로 접속하여 JMX Bean 정보를 확인하기 위한 스크립트(GUI 환경 필요)
add-user.sh	사용자를 추가하기 위한 스크립트
dump.sh	인스턴스 장애시 Java Process 의 Thread Dump 를 받기 위한 스크립트
encrypt.sh	데이터베이스 접속 패스워드를 암호화하기 위한 스크립트

- **env.sh 파일**

env.sh 는 JBoss EAP 6 환경설정을 위한 파일이다. 각 인스턴스의 모든 설정은 이 파일에서 하면 된다. 4 개의 JBoss 인스턴스마다 각각의 인스턴스의 이름, 디렉터리의 이름, 포트 오프셋 등을 설정하면 된다. 또 이 파일에는 JVM 의 메모리 옵션, GC 옵션, OutOfMemory 오류가 발생할 때 Heapdump 를 출력하는 옵션 등이 설정되어 있다.

스탠드얼론 인스턴스가 사용할 프로파일을 변경하려면, CONFIG_FILE=standalone-full-ha.xml 과같이 프로파일 파일명을 지정하면 된다.

```
#!/bin/sh

DATE=`date +%Y%m%d%H%M%S`

export JBOSS_HOME=/app/was/jboss-eap-7.2
export DOMAIN_BASE=/app/was/domains
export SERVER_NAME=admin11

export JBOSS_LOG_DIR=/app/was/domains/$SERVER_NAME

if [ e$JBOSS_LOG_DIR = "e" ]
then
    export JBOSS_LOG_DIR="$JBOSS_HOME/log"
fi

if [ e$JBOSS_LOG_DIR != "e" ]
then
    export JBOSS_LOG_DIR="$JBOSS_LOG_DIR"
fi

# Configuration File
#export CONFIG_FILE=standalone-full-ha.xml
export CONFIG_FILE=standalone-ha.xml

export EXTERNAL_DEPLOYMENT=/app/was/domains/applications
export HOST_NAME=master
export NODE_NAME=$SERVER_NAME
export PORT_OFFSET=0
```

```
export JBOSS_USER=jboss

# Bind Address
export BIND_ADDR=<IP_ADDRESS>

export MULTICAST_ADDR=230.1.0.1
export JMS_MULTICAST_ADDR=231.7.0.1
export MODCLUSTER_MULTICAST_ADDR=224.0.1.105

export MGMT_ADDR=127.0.0.1

export CONTROLLER_IP=$MGMT_ADDR
let CONTROLLER_PORT=9999+$PORT_OFFSET
export CONTROLLER_PORT

export LAUNCH_JBOSS_IN_BACKGROUND=true

# JBoss System module and User module directory
export JBOSS_MODULEPATH=$JBOSS_HOME/modules

# JVM Options : Server
export JAVA_OPTS="-server $AGENT_OPTS $JAVA_OPTS"

# JVM Options : Memory
export JAVA_OPTS=" $JAVA_OPTS -Xms2048m -Xmx2048m -XX:MaxPermSize=256m"

export JAVA_OPTS=" $JAVA_OPTS -XX:+PrintGCTimeStamps "
export JAVA_OPTS=" $JAVA_OPTS -XX:+PrintGCDetails "
export JAVA_OPTS=" $JAVA_OPTS -Xloggc:$JBOSS_LOG_DIR/gclog/gc_$(date +%Y%m%d).log "
export JAVA_OPTS=" $JAVA_OPTS -XX:+UseParallelGC "
#export JAVA_OPTS=" $JAVA_OPTS -XX:+UseConcMarkSweepGC "
export JAVA_OPTS=" $JAVA_OPTS -XX:+ExplicitGCInvokesConcurrent "
export JAVA_OPTS=" $JAVA_OPTS -XX:-HeapDumpOnOutOfMemoryError "
export JAVA_OPTS=" $JAVA_OPTS -XX:HeapDumpPath=$JBOSS_LOG_DIR/heapdump "
# Linux Large Page Setting
#export JAVA_OPTS=" $JAVA_OPTS -XX:+UseLargePages "

#export JAVA_OPTS=" $JAVA_OPTS -verbose:gc"
export JAVA_OPTS=" $JAVA_OPTS -Djava.net.preferIPv4Stack=true"
export JAVA_OPTS=" $JAVA_OPTS -Dorg.jboss.resolver.warning=true"
```

```
export JAVA_OPTS=" $JAVA_OPTS -Dsun.rmi.dgc.client.gcInterval=3600000 "  
export JAVA_OPTS=" $JAVA_OPTS -Dsun.rmi.dgc.server.gcInterval=3600000"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.modules.system.pkgs=org.jboss.byteman,com.khan"  
export JAVA_OPTS=" $JAVA_OPTS -Djava.awt.headless=true"  
  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.server.base.dir=$DOMAIN_BASE/$SERVER_NAME"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.server.log.dir=$JBOSS_LOG_DIR"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.external.deployments=$EXTERNAL_DEPLOYMENT"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.socket.binding.port-offset=$PORT_OFFSET"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.node.name=$NODE_NAME"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.bind.address.management=$MGMT_ADDR"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.bind.address=$BIND_ADDR"  
#export JAVA_OPTS=" $JAVA_OPTS -Djboss.bind_addr=$MULTICAST_ADDR"  
#export JAVA_OPTS=" $JAVA_OPTS -Djboss.default.jgroups.stack=tcp"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.default.multicast.address=$MULTICAST_ADDR"  
export JAVA_OPTS=" $JAVA_OPTS -Djboss.messaging.group.address=$JMS_MULTICAST_ADDR"  
export JAVA_OPTS=" $JAVA_OPTS -  
Djboss.modcluster.multicast.address=$MODCLUSTER_MULTICAST_ADDR"  
  
export JAVA_OPTS=" $JAVA_OPTS -Dorg.jboss.as.logging.per-deployment=false "  
  
export JAVA_OPTS=" $JAVA_OPTS -Dserver.mode=local"  
  
echo "=====  
echo " KHAN [provisioning]      http://www.openmaru.com/ "  
echo " JBoss EAP 7.2.0          service@openmaru.com"  
echo "-----"  
echo "JBOSS_HOME=$JBOSS_HOME"  
echo "DOMAIN_BASE=$DOMAIN_BASE"  
echo "SERVER_NAME=$SERVER_NAME"  
echo "CONFIG_FILE=$CONFIG_FILE"  
echo "BIND_ADDR=$BIND_ADDR"  
echo "PORT_OFFSET=$PORT_OFFSET"  
echo "MULTICAST_ADDR=$MULTICAST_ADDR"  
echo "CONTROLLER=$CONTROLLER_IP:$CONTROLLER_PORT"  
echo "CONSOLE=http://$CONTROLLER_IP:$CONSOLE_PORT"  
echo "=====
```

- **start.sh 파일**

JBoss 인스턴스를 시작하는 스크립트이다. 이 파일에서는 JBoss 인스턴스가 이미 실행되어 있는지 조사하여 다시 실행되지 않도록 하며, env.sh 파일에서 JBOSS_USER 로 JBoss 프로세스를 관리하는 사용자로 지정한 사용자만 JBoss 인스턴스를 시작할 수 있도록 한다.

GC 로그를 남길 디렉터리나 로그 디렉터리, Heap dump 를 남길 디렉터리가 없으면 생성한다.

JBoss 인스턴스는 nohup 로그를 남기며 백그라운드로 실행된다. 기존의 nohup 로그는 현재 실행 시각을 확장자로 하여 백업하고 새로운 nohup 로그를 남긴다.

```
#!/bin/sh

DATE=`date +%Y%m%d%H%M%S`

./env.sh

PID=`ps -ef | grep java | grep "=$SERVER_NAME " | awk '{print $2}`
echo $PID

if [ e$PID != "e" ]
then
    echo "JBoss SERVER - $SERVER_NAME is already RUNNING..."
    exit;
fi

UNAME=`id -u -n`
if [ e$UNAME != "e$JBOSS_USER" ]
then
    echo "Use $JBOSS_USER account to start JBoss SERVER - $SERVER_NAME..."
    exit;
fi

echo $JAVA_OPTS

if [ ! -d "$JBOSS_LOG_DIR/nohup" ];
then
    mkdir -p $JBOSS_LOG_DIR/nohup
fi

if [ ! -d "$JBOSS_LOG_DIR/gclog" ];
then
    mkdir -p $JBOSS_LOG_DIR/gclog
fi
```

```
if [ ! -d "$JBOSS_LOG_DIR/heapdump" ];
then
  mkdir -p $JBOSS_LOG_DIR/heapdump
fi

mv $JBOSS_LOG_DIR/nohup/$SERVER_NAME.out $JBOSS_LOG_DIR/nohup/$SERVER_NAME.out.$DATE

#nohup $JBOSS_HOME/bin/standalone.sh -DSERVER=$SERVER_NAME -b $BIND_ADDR -u
$MULTICAST_ADDR -P=$DOMAIN_BASE/$SERVER_NAME/env.properties -c $CONFIG_FILE >>
$SERVER_NAME.out &

nohup $JBOSS_HOME/bin/standalone.sh -DSERVER=$SERVER_NAME -
P=$DOMAIN_BASE/$SERVER_NAME/env.properties -c $CONFIG_FILE >>
$JBOSS_LOG_DIR/nohup/$SERVER_NAME.out &

if [ e$1 = "enotail" ]
then
  echo "Starting... $SERVER_NAME"
  exit;
fi

#tail -f log/server.log
tail -f $JBOSS_LOG_DIR/nohup/$SERVER_NAME.out
```

● shutdown.sh

shutdown CLI 명령을 사용하여 JBoss 인스턴스를 종료하는 스크립트이다. shutdown 명령을 사용하여 종료하기 때문에 작업 중이던 처리를 모두 완료한 후 종료한다.

```
#!/bin/sh

./env.sh

$JBOSS_HOME/bin/jboss-cli.sh --connect --controller=$CONTROLLER_IP:$CONTROLLER_PORT --
command=:shutdown
```

● status.sh

JBoss 인스턴스가 실행 중인지 프로세스를 체크하는 스크립트이다.

```
#!/bin/sh

./env.sh
```

```
ps -ef | grep java | grep "SERVER=$SERVER_NAME "
```

- **kill.sh**

'kill -9 <pid>'로 JBoss 인스턴스를 강제 종료하는 스크립트이다. 서버를 실행하는 start.sh 스크립트에서 \$SERVER_NAME 환경변수를 지정하였기 때문에 해당 자바 프로세스만 kill 한다.

```
#!/bin/sh

./env.sh

ps -ef | grep java | grep "SERVER=$SERVER_NAME " | awk {'print "kill -9 " $2'} | sh -x
```

- **nohup.sh**

JBoss 인스턴스의 nohup 로그를 tail 로 보는 스크립트이다.

```
#!/bin/sh

./env.sh

tail -f $JBASS_LOG_DIR/nohup/$SERVER_NAME.out
```

- **jboss-cli.sh**

해당 JBoss 인스턴스에 CLI 로 접속하는 스크립트이다. 스크립트를 실행하면 env.sh 파일에 정의된 컨트롤러로 자동으로 접속한다.

```
#!/bin/sh

./env.sh

export JAVA_OPTS=" -Djava.awt.headless=false $JAVA_OPTS"

$JBASS_HOME/bin/jboss-cli.sh --controller=$CONTROLLER_IP:$CONTROLLER_PORT --connect $@
```

- **jconsole.sh**

JConsole 을 실행하는 스크립트이다. 이 스크립트를 실행하면 해당 JBoss 인스턴스에 접속하는 remoting-jmx URI 를 화면에 출력하고 JConsole 을 실행한다. 원격에서 접속하려면 화면에 출력되는 JMX URL 을 사용하면 된다.

```
#!/bin/sh

./env.sh

echo "===== "
echo " JMX URL : service:jmx:remoting-jmx://$MGMT_ADDR:$CONTROLLER_PORT"
echo "===== "
$JBASS_HOME/bin/jconsole.sh
```

- **dump.sh**

이 스크립트를 실행하면 자동으로 3 초 간격으로 5 회 스레드 덤프를 nohup 에 출력한다.

```
#!/bin/sh

./env.sh

for count in 1 2 3 4 5; do
  echo "Thread Dump : 2"
  for i in `ps -ef | grep java | grep "SERVER=$SERVER_NAME " | awk '{print $2}'`;do
    echo "+kill -3 $i"
    kill -3 $i
    echo "sleep 1 sec"
    sleep 1
  done
  echo "done"
  sleep 3
done
```

5. 운영체제 환경 설정

5.1 커널 파라미터

웹 서버와 웹 기반 미들웨어 서버는 모두 네트워크를 통해 서비스를 제공하는 시스템이다. 네트워크를 통해 데이터를 전달하기 때문에, 운영체제의 TCP/IP 에 대한 튜닝은 필수적이다. 아래 표에서 설명한 핵심적인 파라미터를 적용하는 것이 좋다. 특히 TCP 의 수신, 송신 버퍼의 크기는 운영체제가 기본적으로 제공하는 것보다 크게 설정해야 서버의 성능을 향상할 수 있다. 다음 설정을 웹 서버와 JBoss 운영 서버에 대해 모두 적용한다.

파라미터	권장값	설명
net.ipv4.tcp_keepalive_time	30	keep-alive 시간을 줄인다.
net.ipv4.tcp_fin_timeout	10	FIN 타임아웃 시간을 줄여 FD 를 빨리 확보할 수 있도록 한다.
net.core.netdev_max_backlog	2500	백로그에 들어오는 소켓 개수를 늘린다.
net.ipv4.tcp_retries1	3	TCP 연결에 문제가 있을 때 연결을 재시도하는 횟수(최솟값은 3 이다)
net.ipv4.tcp_retries2	3	TCP 연결을 끊기 전에 재시도하는 횟수를 줄인다.
net.ipv4.ip_local_port_range	1024 65000	사용할 수 있는 로컬 포트 범위를 늘린다.
net.core.rmem_max	56777216	TCP 수신 버퍼크기 최댓값을 늘린다.
net.core.rmem_default	16777216	TCP 수신 버퍼크기 기본값을 늘린다.
net.core.wmem_max	56777216	TCP 전송 버퍼크기 최댓값을 늘린다.
net.core.wmem_default	16777216	TCP 수신 버퍼크기 기본값을 늘린다.
net.ipv4.tcp_window_scaling	1	65kb 이상의 큰 TCP 윈도우 스케일링을

		사용한다.
net.ipv4.tcp_orphan_retries	0	서버 측에서 닫은 TCP 연결을 끊기 전에 확인하는 횟수를 줄인다. 기본값은 7 로 50 초~16 분 정도 걸린다.
net.ipv4.tcp_sack	0	SYNC 패킷을 전송한 후 일부 ACK 를 받지 못했을 경우 선택적으로 받지 못한 ACK 패킷을 받도록 설정할 수 있다. 0 은 받지 않는 설정이다. 패킷 유실이 많은 네트워크에서는 1 로 설정한다.

5.2 적용한 커널 파라미터 값

/etc/sysctl.conf

```
# Updates

net.ipv4.neigh.default.unres_qlen=100
net.ipv4.tcp_keepalive_time = 30
net.ipv4.tcp_fin_timeout = 10
net.core.netdev_max_backlog = 2500
net.ipv4.tcp_retries1 = 2
net.ipv4.tcp_retries2 = 3
net.ipv4.ip_local_port_range = 1024 65000
net.core.rmem_max = 56777216
net.core.rmem_default = 16777216
net.core.wmem_max = 56777216
net.core.wmem_default = 16777216
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_timestamps = 0
net.ipv4.tcp_sack = 0
net.ipv4.tcp_orphan_retries = 0
```

5.3 사용자 limit 값 설정

```
# /etc/security/limits.conf
#
#This file sets the resource limits for the users logged in via PAM.
#It does not affect resource limits of the system services.
#
#Each line describes a limit for a user in the form:
#
#<domain>    <type> <item> <value>
#
#Where:
#<domain> can be:
#   - an user name
#   - a group name, with @group syntax
#   - the wildcard *, for default entry
#   - the wildcard %, can be also used with %group syntax,
#       for maxlogin limit
#
#<type> can have the two values:
#   - "soft" for enforcing the soft limits
#   - "hard" for enforcing hard limits
#
#<item> can be one of the following:
#   - core - limits the core file size (KB)
#   - data - max data size (KB)
#   - fsize - maximum filesize (KB)
#   - memlock - max locked-in-memory address space (KB)
#   - nofile - max number of open files
#   - rss - max resident set size (KB)
#   - stack - max stack size (KB)
#   - cpu - max CPU time (MIN)
#   - nproc - max number of processes
#   - as - address space limit (KB)
#   - maxlogins - max number of logins for this user
#   - maxsyslogins - max number of logins on the system
#   - priority - the priority to run user process with
#   - locks - max number of file locks the user can hold
#   - sigpending - max number of pending signals
#   - msgqueue - max memory used by POSIX message queues (bytes)
```

```
# - nice - max nice priority allowed to raise to values: [-20, 19]
# - rtprio - max realtime priority
#
#<domain> <type> <item> <value>
#
#*      soft  core    0
#*      hard  rss     10000
#@student  hard  nproc    20
#@faculty  soft  nproc    20
#@faculty  hard  nproc    50
#ftp      hard  nproc    0
#@student  -    maxlogins 4

jboss    hard  nofile   65536
jboss    soft  nofile   65536

jboss    soft  nproc    2047
jboss    hard  nproc    16384

# End of file
```

6. JBoss EAP 주요 설정값

6.1 JDK 설치

Oracle JDK openjdk.1.8 rpm 버전을 설치하였다.

JDK 는 alternatives 명령을 이용하여 리눅스 system 의 모든 사용자가 기본으로 설치된 java openjdk.1.8 을 사용할 수 있도록 설정하였다.

JDK 는 현재 가장 최신 버전을 사용하였다.

6.2 Native 모듈 설치

JBoss EAP 의 Native 컴포넌트는 Red Hat Enterprise Linux 를 위한 OpenSSL, JBoss Native, mod_jk, mod_cluster, NSAPI for Solaris, ISAPI for Windows, HornetQ LibAIO Native 모듈이 포함되어 있다.

특히 Tomcat Native (TC-Native)는 JNI(Java Native Interface)를 이용하여 JBoss 내의 Tomcat 모듈의 코어기능을 Java 가 아닌 Native 코드를 사용할 수 있도록 하여, 전체적으로 서버 성능을 향상시킬 수 있다.

또, JBoss EAP 설치시 JBoss Native Component 들이 설치되었다. 설치 확인은 자동 설치 과정에서 libaio, apr, apr-util, openssl RPM 모듈이 설치되었다.

또, JBoss EAP 설치시 JBoss Native Component 들이 설치되었다. 설치 확인은 \$JBASS_HOME/modules/system/layers/base/org/jboss/as/web/main/lib/linux-x86_64 에 파일들이 설치되었는지 확인하면 된다.

점검 항목	현재값	설명
JBoss Native Component 적용여부	적용	jboss-eap-7.2/modules/system/layers/base/org/jboss/as/web/main/lib/linux-x86_64 디렉토리에 파일들이 설치되어 있다.

- APR(Apache Portable Runtime)및 JMS(HornetQ)에서 사용하는 AIO 가 설치되면 빠른 성능을 낼 수 있다.

6.2.1 추가 설치 패키지

```
yum install apr apr-util openssl
```

6.2.2 web subsystem native 사용 설정

```
<subsystem xmlns="urn:jboss:domain:web:1.1" default-virtual-server="default-host" instance-id="${jboss.node.name}" native="true" >
```

6.3 Thread Pool 설정

HTTP/AJP 연결에 대한 Thread Pool 의 max 개수를 충분한 값으로 설정해 주어야 한다.

```
<connector name="ajp" protocol="AJP/1.3" scheme="http" socket-binding="ajp" executor="http-thread-pool" max-connections="3260"/>
<connector name="http" protocol="HTTP/1.1" scheme="http" socketbinding="http" enable-lookups="false" executor="http-thread-pool" max-connections="3260"/>
```

6.3.1 스레드 풀 설정

HTTP/AJP 연결에 대한 Thread Pool 을 아래와 같이 Thread subsystem 에서 250 개 스레드를 사용할 수 있도록 설정하였다. AJP 연결과 HTTP 연결에 대해 별도의 스레드를 설정하여 지정하는 것도 가능하다.

```
<subsystem xmlns="urn:jboss:domain:threads:1.1" >
...
    <unbounded-queue-thread-pool name="http-thread-pool" >
        <max-threads count="250"/>
        <keepalive-time time="60" unit="minutes"/>
    </unbounded-queue-thread-pool >
...
</subsystem >
```

6.3.2 배포 설정

6.3.2.1 수동 배포 디렉터리 설정

별도의 디렉토리를 애플리케이션 배포 디렉터리로 사용할 수 있도록 설정한다.
\$GROUP_NAME.war 파일이나 디렉터리 모두 가능하다.

```
<deployments>
  <deployment name="$GROUP_NAME.war " runtime-name="$GROUP_NAME.war ">
    <fs-exploded
path="$${jboss.external.deployment:/svc/test/was/domains/applications}/$GROUP_NAME.war"/>
  </deployment>
</deployments>
```

6.3.2.2 배포스캐너 설정

별도의 디렉토리를 애플리케이션 배포 디렉토리로 사용할 수 있도록 배포 스캐너를 설정한다.

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:1.1">
  <deployment-scanner path="deployments" relative-to="jboss.server.base.dir" scan-
interval="5000"/>
  <deployment-scanner name="external-deployment-scanner"
path="$${jboss.external.deployment:/svc/test/was/domains/applications}" scan-interval="5000" auto-
deploy-exploded="false"/>
</subsystem>
```

6.3.3 인코딩 설정

웹 서브시스템의 URI 인코딩이나 Java 의 파일 인코딩 등 애플리케이션에서 사용하는 인코딩을 설정하여야 한다. env.properties 파일에는 Java 의 시스템 프로퍼티를 설정할 수 있다. 기본적으로 애플리케이션에서 사용할 UTF-8 인코딩을 위한 설정들이 되어 있다. env.properties 파일은 env.sh 파일에서 로딩하도록 설정되어 있다.

이외에도 Java 시스템 프로퍼티로 사용할 값들을 추가하여도 된다.

```
org.apache.catalina.connector.URI_ENCODING=UTF-8
org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING=true
file.encoding=UTF-8
```

6.4 데이터소스 자동 설정내역

KHAN [provisioning]으로 자동 설치과정에서 아래와 같이 데이터소스가 자동 설정되었다. 데이터베이스 연결 driver 는 module 로 설치되었다. oracle, mysql 드라이버가 자동 설치되어 있다.

구분	내용
데이터베이스 종류	none
JDBC 풀 이름	testDS
JNDI 이름	java:jboss/datasources/testDS
연결 URL	jdbc:mysql://192.168.0.22:3306/test
연결 풀 최소값	10
연결 풀 최대값	250
데이터베이스 사용자	root
데이터베이스 비밀번호	*****
preparedStatement	0
캐시 크기	

6.5 데이터소스 연결 테스트 방법

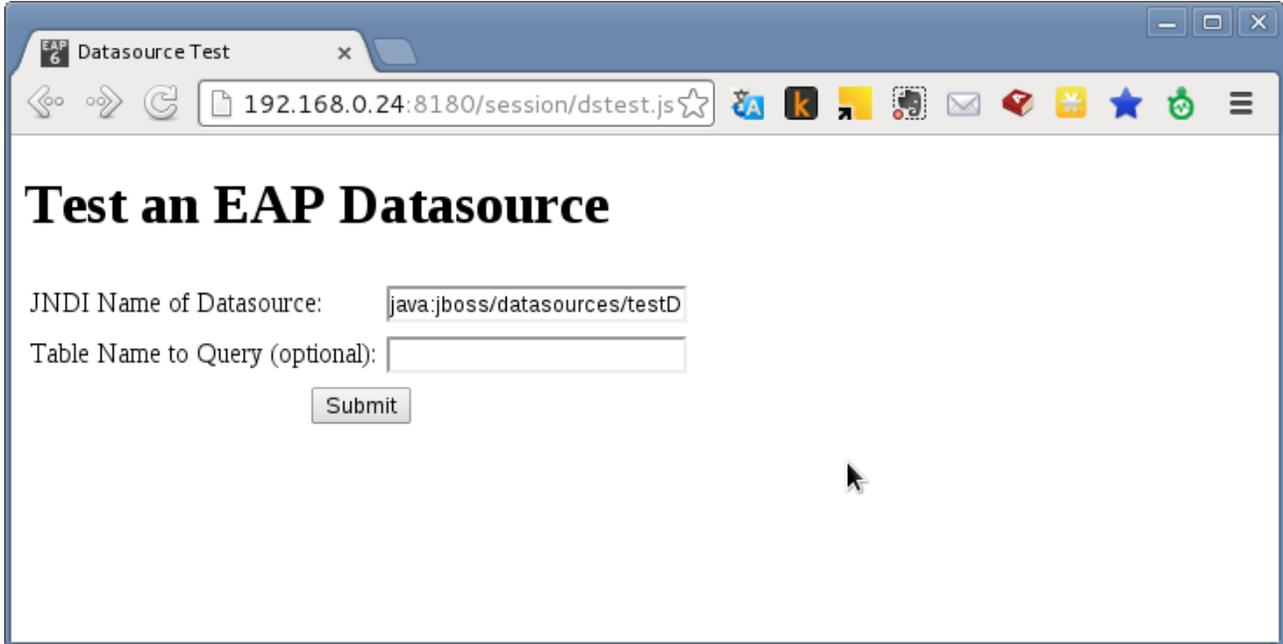
자동 설치한 session.war 애플리케이션내에 데이터소스 연결 테스트를 위한 dstest.jsp 파일이 함께 설치되었다.

http://host_ip:port/session/dstest.jsp 로 접속하여 테스트할 수 있다.

JBoss EWS 가 함께 설치되었다면, JBoss EWS 의 홈페이지로 접속하면 테스트 URL 목록이 생성되어 있으니, 이를 활용하면 된다.

6.5.1 데이터소스 연결 테스트 방법

설정된 데이터소스 JNDI 이름을 입력합니다. 테이블이 있다면, 테이블 이름을 입력하면 데이터를 출력합니다.



Test an EAP Datasource

JNDI Name of Datasource:

Table Name to Query (optional):



Results of Test

Successfully looked up DataSource named java:jboss/datasources/testDS

Successfully connected to database.

Attempting query "SELECT * FROM "

An exception was thrown: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near " at line 1

6.6 데이터소스 설정 방법

애플리케이션에서 사용할 데이터 베이스 연결 정보를 설정합니다. JBoss 에서 데이터 소스를 설정하려면 먼저, 사용하려는 데이터베이스에 대한 JDBC 드라이버를 설치하여야 하며, 설치된 드라이버에 대한 데이터 소스를 설정하여야 합니다.

먼저 데이터베이스에 대한 드라이버를 설정하여야 합니다. 설치하는 방법은 두가지 방법이 있습니다. 첫번째는, JBoss 의 코어 모듈로 설치하는 방법이며, 두번째 방법은 JDBC 드라이버 JAR 파일을 애플리케이션 처럼 배포하는 방법입니다.

참고로, KHAN [provisioning] 자동 설치시 module 로 mysql, oracle JDBC driver 가 이미 설치되어 있다.

6.6.1 JDBC 드라이버를 모듈로 설정하는 방법

1) JDBC 드라이버 복사

`jboss-eap-7.2/modules/system/layers/ext/com/mysql/main/mysql-connector-java-5.1.22-bin.jar`

2) module.xml 파일 작성

`jboss-eap-7.2/modules/system/layers/ext/com/mysql/main/module.xml` 파일을 작성한다.

```
<?xml version="1.0" encoding="UTF-8"?>

<module xmlns="urn:jboss:module:1.0" name="com.mysql">
  <resources>
    <resource-root path="mysql-connector-java-5.1.22-bin.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
  </dependencies>
</module>
```

3) JBoss CLI 를 이용한 드라이버 추가

```
/subsystem=datasources/jdbc-driver=mysql:add(driver-name=mysql,driver-module-
name=com.mysql,driver-xa-datasource-class-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource)
```

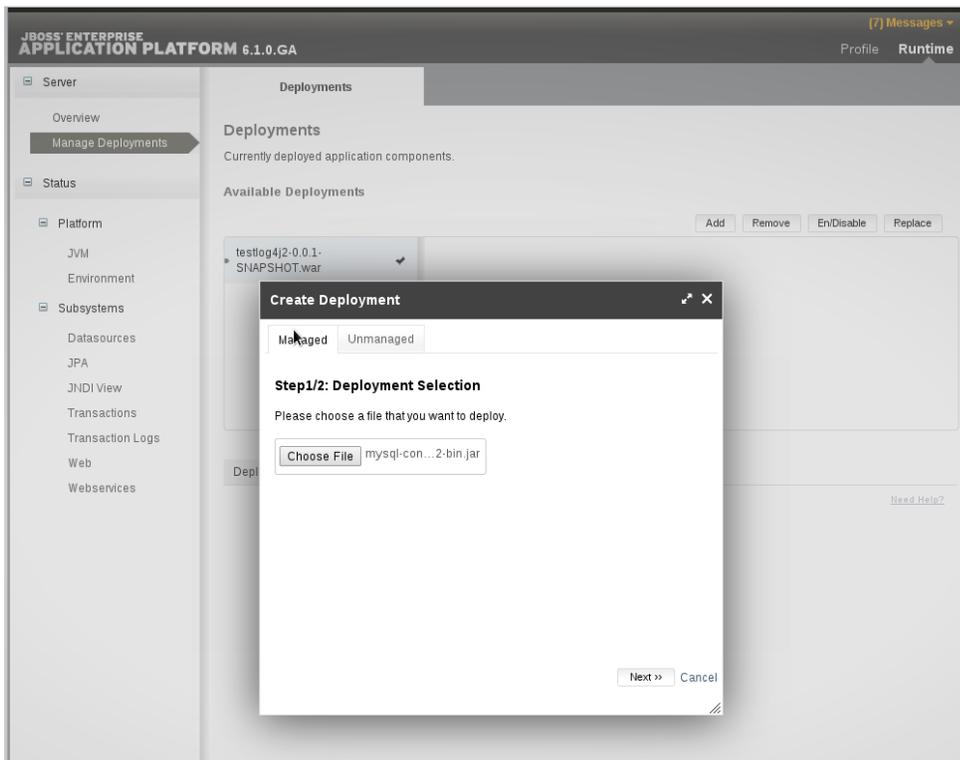
4) 데이터 소스 설정

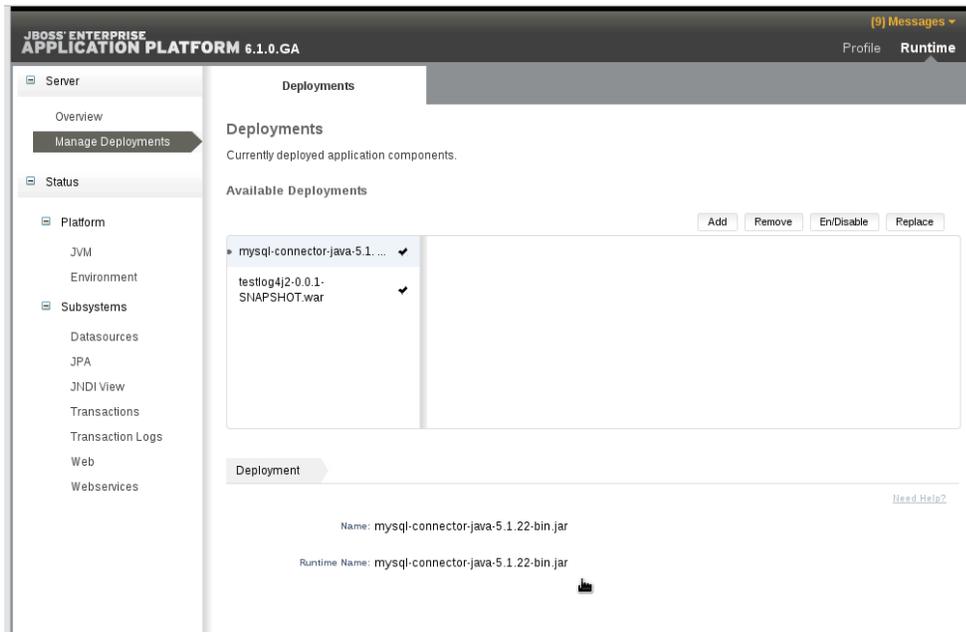
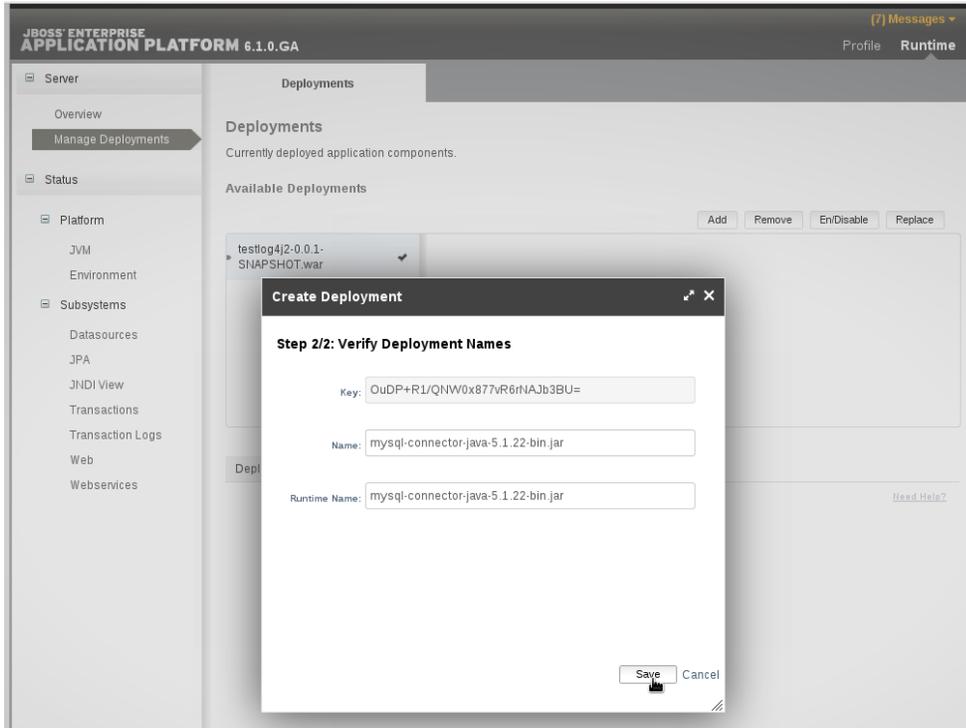
아래 URL 을 참고한다.

https://access.redhat.com/site/documentation/en-US/JBoss_Enterprise_Application_Platform/6.2/html-single/Administration_and_Configuration_Guide/index.html#chap-Datasource_Management

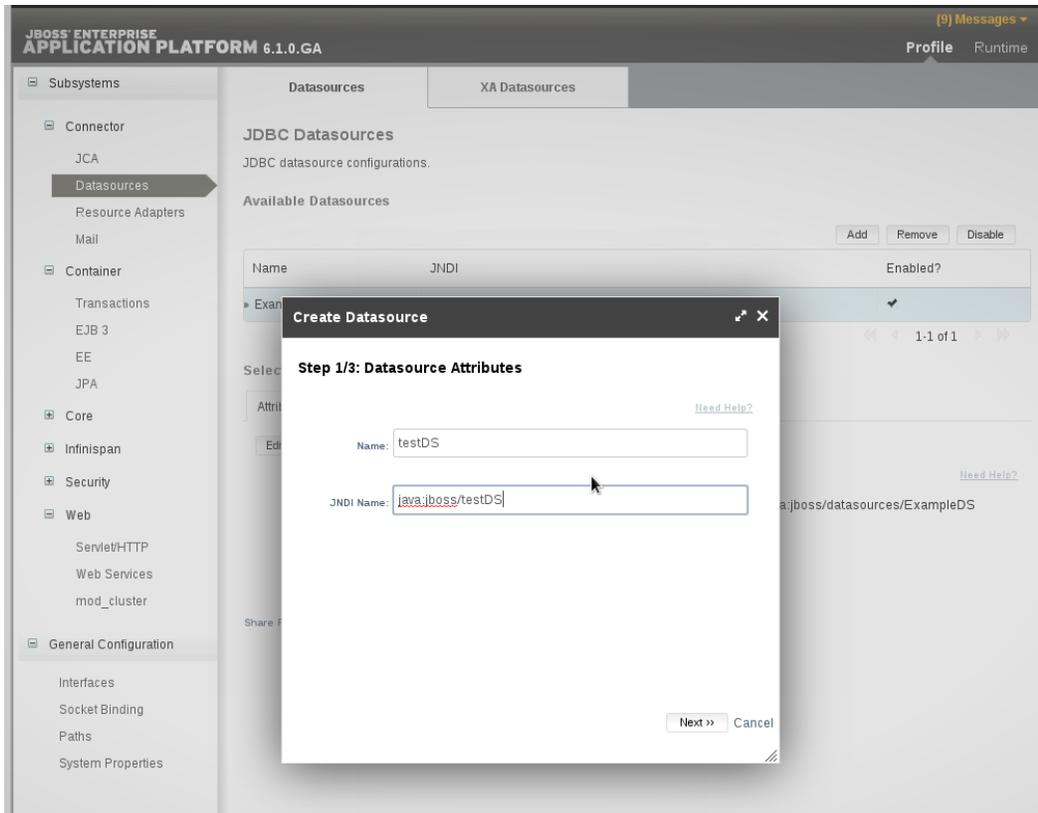
6.6.2 JDBC 드라이버를 애플리케이션으로 설정하는 방법

1) JDBC 드라이버를 배포한다.

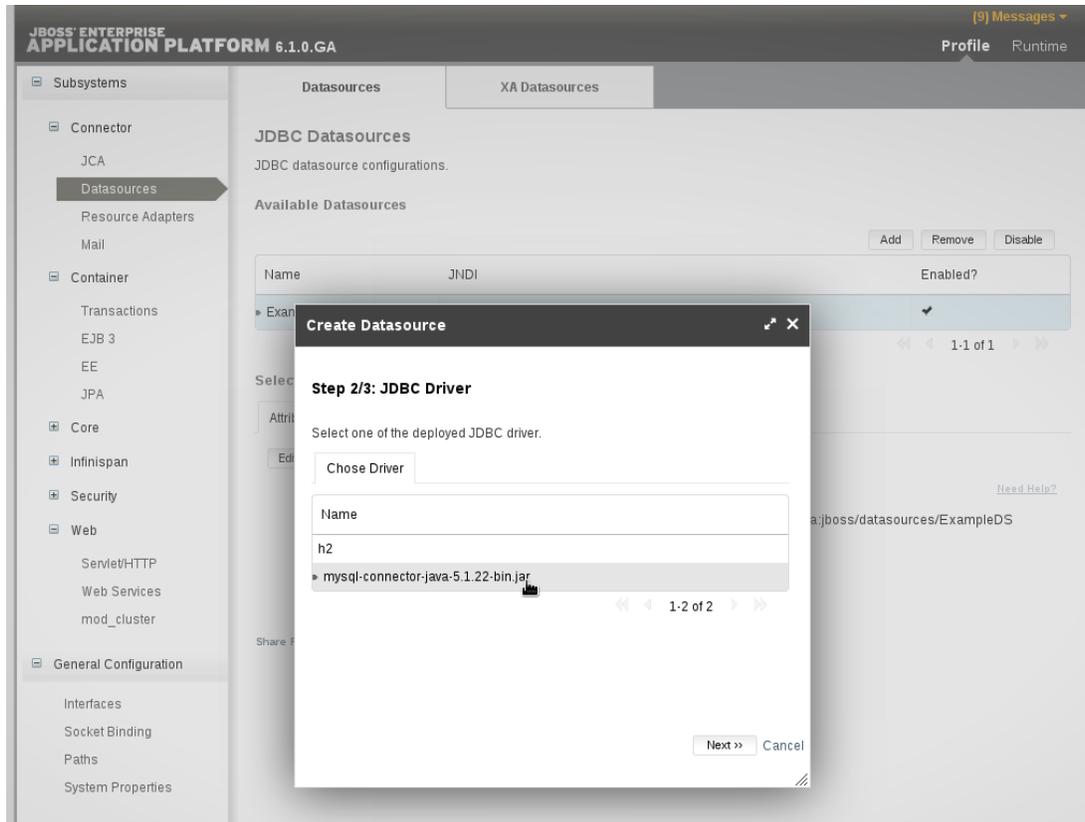




2) 데이터소스 설정



데이터 소스의 이름을 입력하고 "Next>>" 버튼을 클릭하면, 업로드된 JDBC 드라이버 JAR 파일이 표시된다.



CLI 에서 데이터소스 추가하려면 다음 명령을 입력합니다.

```
data-source add --name=[데이터 소스명] W
--profile=[설정 대상 프로파일] W
--jndi-name=[JNDI 명] W
--connection-url=[접속 URL] W
--driver-name=[드라이버명] W
--user-name=[접속 사용자명] W
--password=[접속 사용자 패스워드]
```

CLI 에서 데이터소스 관리하기 위한 명령은 다음과 같은 것이 있습니다.

```
data-source enable --name=[데이터 소스명]
data-source disable --name=[데이터 소스명]
data-source remove --name=[데이터 소스명]
```

KHAN [provisioning]에서는 데이터 소스 등록을 위한 스크립트가 자동으로 설치되었다.

/app/was/domains/admin11/cli-scripts/create-datasource.sh 파일에 새로운 데이터소스 설정을 위한 CLI 스크립트가 생성되어 있다.

아래와 같은 파일내용 중에서 사용자가 생성할 JNDI 이름, Connection URL 등을 수정하여 실행하면 데이터소스가 생성된다.

```
#!/bin/sh
# -----
# KHAN [provisioning]          http://www.openmaru.com/
# JBoss EAP 7.2.0
#
# contact : service@openmaru.com
# Copyright(C) 2013, openmaru.com, All Rights Reserved.
# -----

../env.sh

# -----
# CONNECTION POOL NAME
# -----
read -p "> Connection Pool 이름을 입력하십시오 : " DATASOURCE_NAME

# -----
# MYSQL Data Source Example
#
# jndi-name, connection-url, username, password 은 접속 환경에 맞게
# 수정하여 사용하십시오.
# -----
export MYSQL_DATASOURCE="
data-source add   WW
--name=$DATASOURCE_NAME           WW
--jndi-name=java:jboss/datasources/abcDS  WW
--driver-name=mysql                WW
--driver-class=com.mysql.jdbc.Driver  WW
--connection-url=jdbc:mysql://192.168.0.11:3306/test  WW
--min-pool-size=100                WW
--max-pool-size=250                WW
--pool-use-strict-min=true         WW
--pool-prefill=true                WW
--user-name=root                   WW
```

```
--password=passswd           WW
--valid-connection-checker-class-
name=org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLValidConnectionChecker WW
--validate-on-match=false     WW
--background-validation=true   WW
--background-validation-millis=10000 WW
--exception-sorter-class-name=org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLExceptionSorter WW
--blocking-timeout-wait-millis=60000 WW
--idle-timeout-minutes=15     WW
--track-statements=true       WW
--prepared-statements-cache-size=300 WW
--share-prepared-statements=true"
```

6.7 모듈 등록 및 사용방법

6.7.1 추가 모듈 디렉토리 설정

layers.conf 파일 작성하여, 사용자가 추가하는 모듈에 대해서 modules/system/layers/ext 디렉토리를 사용할 수 있도록 설정하였다.

jboss-eap-7.2/modules/layers.conf

```
layers=ext
```

ext 디렉토리에는 Oracle, MySQL JDBC 드라이버가 모듈로 배포되어 있다.

6.8 애플리케이션에서 필요한 모듈 로딩 방법

애플리케이션에서 필요한 JAR 파일들을 모듈로 등록하여 사용할 수 있다. 이런 경우 jboss-deployment-structure.xml 파일에 아래와 같이 dependency 를 설정하여 사용한다.

WEB-INF/jboss-deployment-structure.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss-deployment-structure>
  <deployment>
    <dependencies>
      <module name="org.springframework.spring" slot="3.1.2">
        <imports>
          <include path="META-INF**"/>
          <include path="org**"/>
        </imports>
      </module>
      <module name="org.springframework.security" slot="main"/>
      <module name="org.mybatis" slot="main"/>

      <module name="org.slf4j"/>
      <module name="org.apache.log4j"/>

      <module name="javax.validation.api"/>

    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

6.9 애플리케이션에서 필요하지 않은 모듈 제외방법

배포되는 애플리케이션에 따라 Java EE 스펙에서 필요한 클래스들을 JBoss 의 모듈로 묵시적으로 로딩한다.

애플리케이션 유형별로 로딩되는 모듈의 이름은 아래 URL 의 표를 확인한다.

<https://docs.jboss.org/author/display/AS71/Implicit+module+dependencies+for+deployments>

애플리케이션에서 JBoss 가 묵시적으로 로딩하는 모듈을 강제로 제외하려면 jboss-deployment-structure.xml 파일에 exclude 설정을 한다.

아래 예제는 JSF 2.0 클래스를 제거하고, JSF 1.2 버전을 로딩하는 예제이다.

WEB-INF/jboss-deployment-structure.xml

```
<jboss-deployment-structure xmlns="urn:jboss:deployment-structure:1.0">
  <deployment>
    <exclusions>
      <module name="javax.faces.api" slot="main"/>
      <module name="com.sun.jsf-impl" slot="main"/>
    </exclusions>
    <dependencies>
      <module name="javax.faces.api" slot="1.2" export="true"/>
      <module name="com.sun.jsf-impl" slot="1.2" export="true"/>
    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

6.10 JDK 클래스 파일들에 접근하는 방법

sun.jdk 모듈에는 JDK 에서 제공하는 클래스들에 접근할 수 있도록 제공하는 모듈이다. 이 모듈에 포함되어 있지 않은 클래스를 로딩하려면 jboss-deployment-structure.xml 파일을 작성하여 아래와 같이 로딩하면 된다.

WEB-INF/jboss-deployment-structure.xml

```
<jboss-deployment-structure xmlns="urn:jboss:deployment-structure:1.1">
  <deployment>
    <dependencies>
      <system>
        <paths>
          <path name="sun/security/action"/>
          <path name="javax/crypto "/>
        </paths>
      </system>
    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

참고 : <https://access.redhat.com/site/solutions/343033>

6.11 Jersey 사용시 설정

WEB-INF/jboss-deployment-structure.xml

```
<jboss-deployment-structure xmlns="urn:jboss:deployment-structure:1.1">
  <deployment>
    <dependencies>
      <system>
        <paths>
          <path name="com/sun/xml/bind/v2/runtime/JAXBCContextImpl"/>
        </paths>
      </system>
    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

<https://access.redhat.com/site/solutions/330173>

6.12 JBoss EAP 주요 설정 항목

6.12.1 애플리케이션 log4j 사용시 설정 방법

애플리케이션에서 log4j 설정시 아래 옵션을 추가하면 된다. 이미 env.sh 파일에 설정되어 있음.

```
export JAVA_OPTS=" $JAVA_OPTS -Dorg.jboss.as.logging.per-deployment=false "
```

6.12.2 인코딩 설정

env.sh 파일에서 env.properties 파일을 로딩하도록 설정되어 있으며, env.properties 파일에 아래 설정이 있음.

```
org.apache.catalina.connector.URI_ENCODING=UTF-8  
org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STRING=true  
file.encoding=UTF-8
```

6.12.3 TCP 클러스터링 방법

JGroups Subsystem 에 아래와 같이 tcpping stack 을 추가하고 default-stack 을 tcpping 으로 변경한다.

실제 변경방법은 env.sh 파일에서 jboss.default.jgroups.stack 을 tcp 로 변경하기만 하면 된다. 머신들의 IP 설정은 이미 KHAN [provisioning]이 자동으로 설정하였다.

```
export JAVA_OPTS=" $JAVA_OPTS -Djboss.default.jgroups.stack=tcp "
```

7. JBoss 운영 방법

JBoss 관리자인 'jboss' 계정에 alias 를 설정하였기 때문에, s1, s2 만 입력하면 인스턴스 디렉터리로 이동한다.

다음과 같이 스크립트파일들을 실행하여 JBoss 인스턴스를 관리할 수 있다.

스크립트	설명
\$./start.sh	인스턴스 시작
\$./stop.sh	인스턴스 정상 종료
\$./kill.sh	인스턴스 강제 종료
\$./status.sh	인스턴스 수행 중인지 체크
\$./nohup.sh	인스턴스 nohup 로그를 tail 로 출력
\$./tail.sh	인스턴스의 서버 로그를 tail 로 출력
\$./dump.sh	인스턴스의 스레드 덤프 출력

7.1 JBoss Admin Console

스탠드얼론 모드의 관리 콘솔 접속 URL 정보

호스트 IP	인스턴스명	관리콘솔
192.168.182.138	admin11	http://192.168.182.138:10090/
192.168.182.138	front11	http://192.168.182.138:10190/
192.168.182.139	admin21	http://192.168.182.139:10090/
192.168.182.139	front21	http://192.168.182.139:10190/

서버 IP 주소와 Port 는 서버 Port 는 9990 + \$PORT_OFFSET 로 접속하면 다음과 같이 JBoss Console 에 접근할 수 있다.

설정된 사용자 ID, 패스워드는 **wasadm/ openmaru!2** 이다.

JBoss® ENTERPRISE APPLICATION PLATFORM 6.0 Profile Runtime

Server

- Configuration
 - Manage Deployments**
- Status
 - JVM
 - Datasources
 - JPA
 - JNDI View
 - Transactions
 - Web
 - Webservices
- Runtime Operations
 - OSGi

Deployments

Currently deployed application components. Deployments that have been added through the filesystem will not be manageable through the web interface.

Available Deployments

Filter:

Name	Runtime Name	Enabled
enterprise.war <small>File System Deployment</small>	enterprise.war	✓
mobile.war <small>File System Deployment</small>	mobile.war	✓
portal.war <small>File System Deployment</small>	portal.war	✓
simple.war <small>File System Deployment</small>	simple.war	✓

1-4 of 4

7.2 JBoss CLI 실행

```

$ ./jboss-cli.sh
=====
JBOSS_HOME=/home/jjeon/jboss/eap/jboss-eap-7.2
DOMAIN_BASE=/home/jjeon/jboss/eap/domains.600
SERVER_NAME=standalone1
CONFIG_FILE=standalone-ha.xml
BIND_ADDR=0.0.0.0
PORT_OFFSET=0
MULTICAST_ADDR=230.1.0.1
CONTROLLER=127.0.0.1:9999
=====
[domain@127.0.0.1:9999 /] ls
deployment          extension
host                interface
management-client-content  path
profile            server-group
socket-binding-group      system-property
launch-type=DOMAIN      local-host-name=master
management-major-version=1      management-micro-version=0
  
```

```
management-minor-version=3      name=Unnamed Domain
process-type=Domain Controller   product-name=EAP
[domain@127.0.0.1:9999 /]
```

각 인스턴스의 9999 + \$PORT_OFFSET 포트에 해당 인스턴스에 접속할 수 있습니다. 이미 설치되어 있는 각 서버 인스턴스의 jboss-cli.sh 파일은 해당 인스턴스에 접속하도록 CLI 접속 설정이 되어 있습니다.

CLI 에 대한 자세한 설명은 아래 URL 을 참고하십시오.

https://access.redhat.com/site/documentation/en-US/JBoss_Enterprise_Application_Platform/6.2/html-single/Administration_and_Configuration_Guide/index.html#sect-The_Management_CLI

8. 애플리케이션 배포 방법

JBoss EAP 6 에서는 아래와 같이 4 가지 배포방법을 사용할 수 있다.

- JBoss CLI 를 이용하는 방법
- 웹 관리 콘솔을 이용하는 방법
- Deployment Scanner 를 이용하는 방법
- Maven Plugin 을 이용하는 방법

8.1 웹 관리 콘솔을 이용하는 방법

브라우저로 콘솔에 로그인한다.

- Runtime -- Manage Deployment 를 선택한다.
- Add Content 를 선택하여, jar, war, ear 들의 Archive 파일을 선택하여 업로드한다.
- 브라우저로 콘솔에 로그인한다.

8.2 JBoss CLI 를 이용하는 방법

- jboss-cli.sh 를 실행하여 커맨드 라인 인터페이스를 시작한다.
- deploy 로 배포할 애플리케이션 파일을 지정한다(도메인 모드인 경우엔 서버 그룹을 지정한다.)
- undeploy 명령으로 배포된 애플리케이션을 지정하여 undeploy 한다.

8.3 Deployment fs-exploded(수동 디렉터리)를 이용하는 방법

fs-exploded 디렉터리에 어플리케이션 복사 후 JBoss 를 재기동한다
example.war 압축파일이나 example.war 디렉토리 모두 가능하다.

8.4 Deployment Scanner 를 이용하는 방법

Deployment 디렉터리에 어플리케이션 복사 후 Deploy / Undeploy 하는 방법이다.
Standalone 구성에서만 동작하며 Domain 모드에서 지원하지 않는다.

어플리케이션 파일을 특정 디렉터리에 복사하여 배포 하는 방법으로 기본값은
\$JBOSS_HOME/standalone/deployments 디렉토리이다.

이 방식은 Standalone Server 에서만 동작하며 도메인 모드에서는 지원하지 않는다.

EAP5 버전의 Deployment Scanner 와 동일하며, 배포 디렉터리나 파일 스캔 간격은
standalone.xml 등에서 변경 가능하다. 배포 대상은 archive 파일과 디렉터리 형식 모두 지원한다.

\$JBOSS_HOME/standalone/deployments 디렉토리에 애플리케이션 파일(압축파일이나
디렉토리)를 복사한 후, .dodeploy 파일을 생성하면, JBoss 가 주기적으로 체크하여(기본값 5 초)
배포를 시작한다.

8.4.1 example.war 압축파일 배포 방법

```
cp target/example.war $JBOSS_HOME/standalone/deployments  
touch $JBOSS_HOME/standalone/deployments/example.war.dodeploy
```

8.4.2 example.war 디렉토리 배포 방법

```
cp -R target/example.war $JBASS_HOME/standalone/deployments  
touch $JBASS_HOME/standalone/deployments/example.war.dodeploy
```

8.5 Maven Plugin 을 이용하는 방법

Maven 기반의 개발환경에서 작성된 pom.xml 파일을 참조하여 어플리케이션을 배포 하는 방법으로, JBoss AS7 Deployment Plugin 을 사용한다. 자세한 사용법은 <http://docs.jboss.org/jbossas/7/plugins/maven/latest/> URL 을 참고한다.

8.5.1 maven deploy 방법

```
mvn package jboss-as:deploy
```

8.5.2 maven undeploy 방법

```
mvn package jboss-as:undeploy
```

8.6 JSP 변경 체크 설정

JBoss EAP 6 는 기본적으로 JSP 변경사항을 체크하지 않도록 설정되어있다. 개발환경에서는 JSP 변경사항을 체크하여 변경사항을 확인해 볼 수 있도록 설정하여야 한다. 변경하는 방법은 먼저 JSP 의 development mode 를 설정하고, 변경사항 체크 주기(check-interval)을 설정하면 된다.

KHAN [provisioning]에서 자동설치 시 /app/was/domains/admin11/cli-scripts/jsp-reload-check.sh 파일과 disable-jsp-reload-check.sh 에 JSP 변경 주기를 설정하기 위한 CLI 스크립트가 생성되어 있다.

8.6.1 JSP development mode 변경

```
/profile=full-ha/subsystem=web/configuration=jsp-configuration/:write-attribute(name=development,value=true)
```

8.6.2 JSP 변경 체크시간을 5 초로 변경

```
/profile=full-ha/subsystem=web/configuration=jsp-configuration/:write-attribute(name=check-interval,value=5)
```

8.7 웹 애플리케이션 세션복제 설정

JBoss EAP 에서 웹 애플리케이션의 세션 복제를 설정하려면 web.xml 파일에 distributable 설정을 한다.

8.7.1 web.xml

```
<?xml version="1.0"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
version="2.4">
<!-- ... -->
<distributable/>
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
</web-app>
```

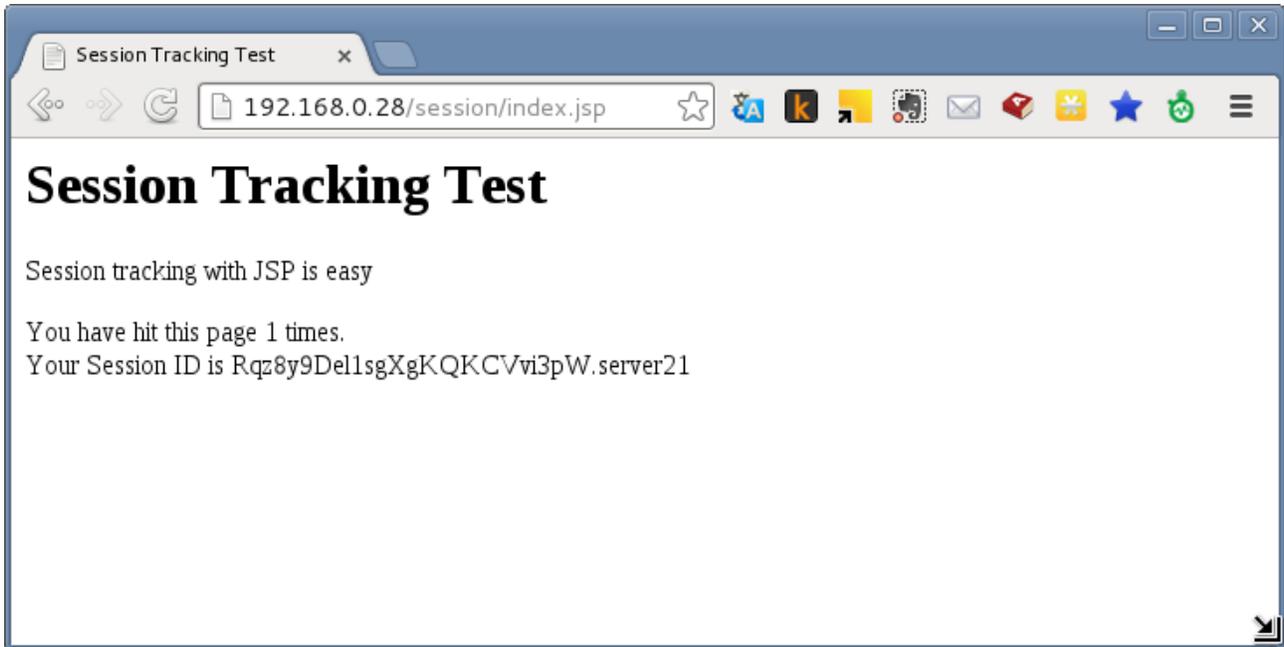
8.7.2 세션 복제를 테스트하기 위한 JSP 파일

HTTP 세션에 int 값을 저장하여 URL 호출시마다 증가시키는 간단한 JSP 페이지로 세션 복제를 테스트해 볼 수 있다. KHAN [provisioning] 설치시 모든 JBoss Instance 에 session.war 가 배포되어 있는데, 이 JSP 파일이 사용된다.

```
<head> <title>Session Tracking Test</title>
</head>
<body>
<h1>Session Tracking Test 2</h1>
<P>
Session tracking with JSP is easy
<%
<%@ page session="true" %>
if (ival == null) ival = new Integer (1);
Integer ival = (Integer) session.getValue ("counter");
// Get the session data value
openmaru.com
else ival = new Integer (ival.intValue() + 1);
session.putValue ("counter", ival);
%>
ed
You have hit this page <%= ival %> times.<br>
<%
out.println("Your Session ID is " + session.getId() + "<br>");
System.out.println("session=" + session.getId());
%>
</body></html>
```

8.8 웹 세션 복제 테스트 애플리케이션

자동 설치한 session.war 애플리케이션내에 웹 세션 복제 테스트를 위한 jsp 파일이 함께 설치되었다. http://host_ip:port/session/index.jsp 로 접속하여 테스트할 수 있다. JBoss EWS 가 함께 설치되었다면, JBoss EWS 의 홈페이지로 접속하면 테스트 URL 목록이 생성되어 있으니, 이를 활용하면 된다.



9. 모니터링 방법

9.1 MBean 그래프 모니터링

KHAN [provisioning]으로 설치한 인스턴스들은 hawtio Chrome Extension 을 사용하여 MBean 정보를 그래프로 모니터링 할 수 있다.

Chrome 브라우저의 Extension 을 설치하여 KHAN [provisioning]으로 설치한 인스턴스에 접속하여 모니터링한다.

9.1.1 hawtio Chrome Extension 설치

<https://chrome.google.com/webstore/>에 접속하여 hawtio 를 검색하여 설치한다.



구글 Web Store 에서 설치되지 않는다면, <http://hawt.io/getstarted/index.html> 에서 Chrome Extension 을 다운로드 하여 압축을 해제한 뒤, 브라우저에서 주소창에 `chrome://extensions/` 입력한다.

우측 상단의 '개발자 모드'를 선택한 후, '압축해제된 확장 프로그램 로드..' 버튼을 클릭하여 압축을 해제한 디렉토리를 입력하면 설치된다.

9.1.2 hawtio 실행하기

브라우저에 `chrome://apps` 을 입력하면 hawtio 애플리케이션을 클릭하면 실행된다.

서버 접속 정보에 KHAN [provisioning]으로 설치한 인스턴스의 IP 주소와 포트 정보, Path 는 jolokia 를 입력하고, 설치한 인스턴스의 관리자 계정 정보를 입력하면 새로운 탭이 열리며 서버에 접속된다.

서버 접속 정보는 저장하여 관리할 수 있다.

다음과 같이 서버의 접속정보를 입력하여 접속한다.

Saved Connections

Connections:   

Connection Settings

Connection name:

Scheme:

Host:

Port:

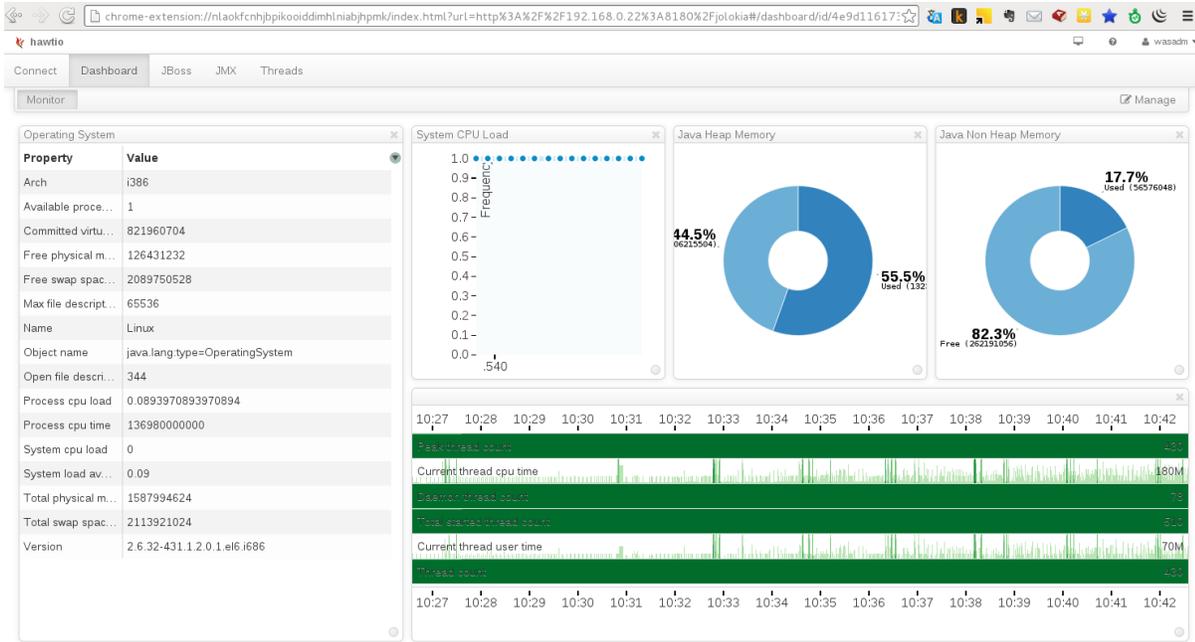
Path:

User name:

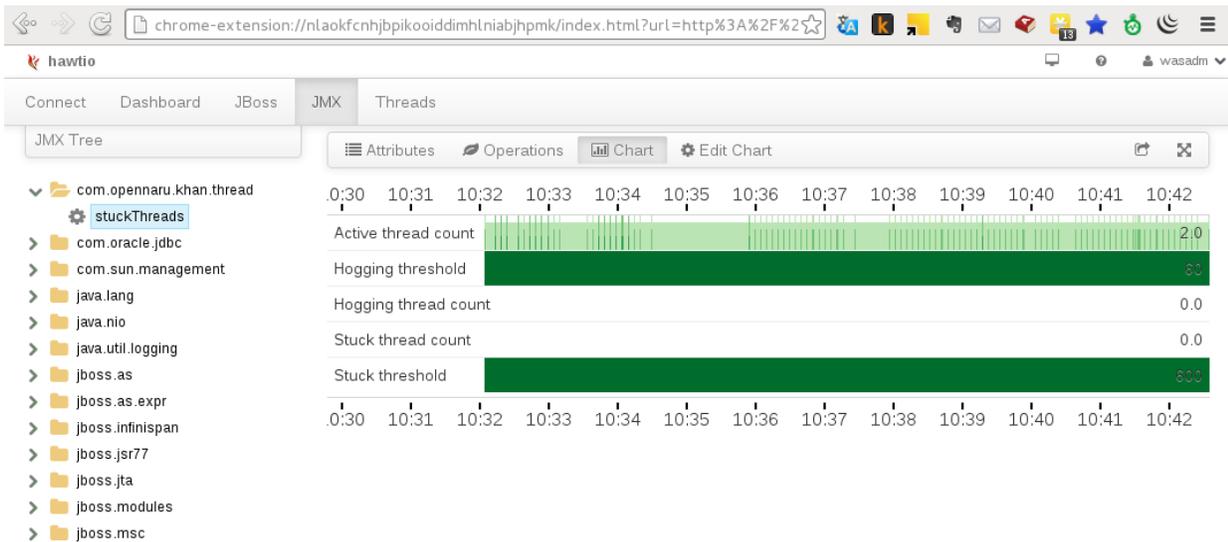
Password:

9.1.3 MBean 모니터링

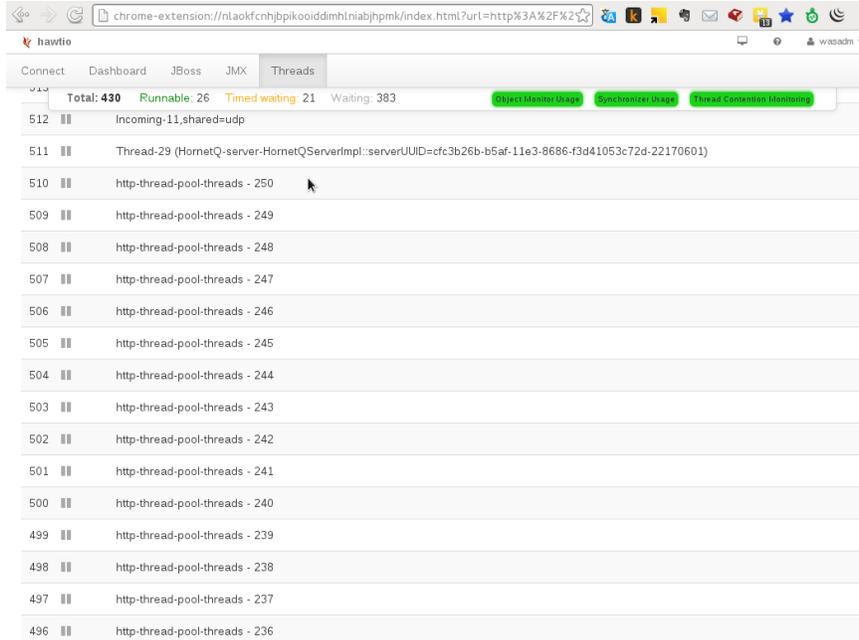
이제 접속된 인스턴스의 정보를 모니터링 할 수 있다. JBoss 에서 제공하는 모든 MBean 에 대한 정보를 모니터링하고 MBean 메소드를 실행하고 관리할 수 있다. 또, 다음과 같이 서버 인스턴스에 대한 대시보드를 확인할 수 있다.



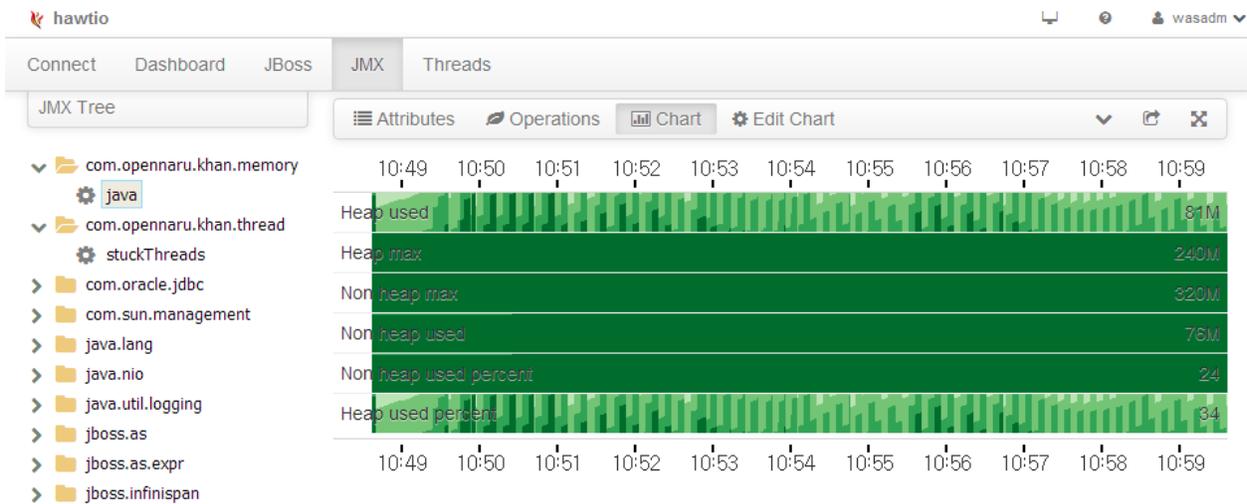
KHAN [provisioning]으로 설치하면 함께 설치되는 Stuck Thread 정보도 모니터링 할 수 있다.



현재 실행 중인 스레드에 대한 Stack Trace 정보도 모니터링 할 수 있다.



KHAN [provisioning]으로 설치할 때 자동으로 설치된 JVM Memory MBean 을 사용하면 다음과 같이 Java 메모리의 사용량 %로 모니터링 할 수 있다.



9.2 JMX 모니터링 스크립트

/app/was/domains/\$SERVER_NAME/monitoring 디렉터리에 JMX 를 이용하여 모니터링 데이터를 CSV 파일로 저장하는 모듈이 저장되어 있다.

./start_standalone_mon.sh 파일을 실행하면 log 디렉터리에 CSV 형식의 monitoring.log 파일이 생성된다.

이 파일을 엑셀에서 불러와 그래프를 그리면 주요한 모니터링 데이터 값의 추세를 확인할 수 있다.

9.3 KHAN [apm] 모니터링

KHAN [monitoring]은 Java 기반 웹 애플리케이션에 대한 실시간 모니터링을 제공함으로써 장애를 사전에 예방하고, 지속적인 성능 개선에 활용할 수 있는 APM(Application Performance Monitoring) 도구이다.

KHAN [monitoring]은 Java 기반 웹 애플리케이션에 대한 실시간 동작 모니터링 뿐만 아니라, 실시간 통계 분석 기법을 적용하여 문제를 사전에 판단할 수 있는 기능들을 제공한다.

KHAN [monitoring]은 웹 애플리케이션 서버 모니터링을 위한 제품으로 다음과 같은 기능을 제공한다.

- **서비스 만족도 지수(APDEX) 제공**

사용자 입장에서 서비스의 만족도를 하나의 숫자로 표현합니다. 서비스의 상황을 0~100까지의 지수를 통해 한눈에 파악할 수 있다.

- **'T-Map' Transaction 분포도**

서비스 전체 응답시간에 대해 직관적인 분석을 위해 트랜잭션들을 Heat-Map으로 표현하며, 선택한 Cell에 대한 SQL 쿼리를 포함한 상세 프로파일링을 제공한다.

- **실시간 예측(Forecast)**

실시간 통계 분석을 통해 앞으로 몇 분 후에 관리자가 설정한 임계 값에 도달할 것이라는 예측 이벤트를 알려준다.

- **WAS 장애 분석 도구**

WAS의 장애 분석에서 가장 많이 사용되는 JVM Thread Dump 분석도구를 제공한다. WAS Trouble Shooting에 필요한 다양한 데이터를 제공한다.

- **이상 징후 모니터링**

통계 분석을 통해 평소와는 다른 이상 값이 감지되면 이벤트를 통해 알려준다.

- **OS 자원 및 웹 서버 모니터링**

OS 의 CPU, Memory, Disk 사용량, Load Average, Network 사용량, 소켓상태, 웹 서버의 트래픽, RPS 등을 다양한 값을 모니터링 한다.

- **정교한 이벤트 처리**

모니터링 지표들에 대한 수집 값들의 통계 값으로 임계 값을 계산하기 때문에 꼭 필요한 경우에만 모니터링 이벤트가 발생한다.

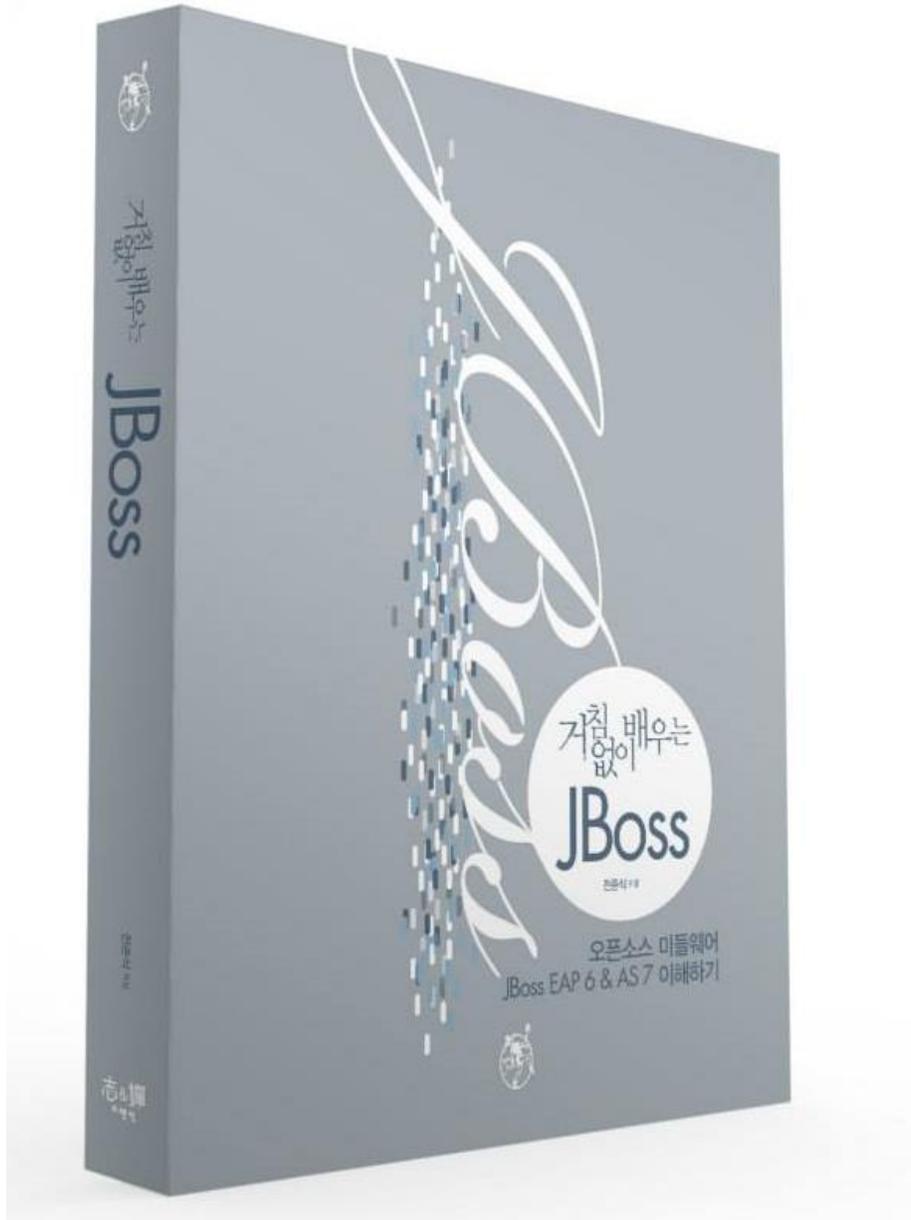
- **HTML 5 기반 User Interface**

모바일을 포함한 다양한 디바이스에서 사용할 수 있도록, HTML 5 기반의 직관적인 사용자 인터페이스를 제공합니다.

/app/was/domains/khan-agent 디렉터리에 서버에서 실행되는 JSP 나 서블릿의 실행 시간 및 주요 메소드 및 SQL 쿼리에 대한 수행시간을 측정하기 위한 모니터링 에이전트가 설치되어 있다.

10. 도움이 필요하십니까?

만약 이 문서에 설명된 절차를 수행할 때 문제를 겪는다면, 오픈나루 고객 포털(<http://support.openmaru.com>)을 방문하십시오.



11. References

- **Red Hat Documentation**
 - <http://docs.redhat.com/>

- **Red Hat 고객지원 포탈**
 - <http://access.redhat.com>

- **오픈나루 고객지원 포탈**
 - <http://support.opennaru.com>

- **오픈나루 Facebook Page**
 - <https://www.facebook.com/opennaru>



t : +82-2-469-5426 **f** : +82-2-469-7247

e : service@opennaru.com, sales@opennaru.com

h : <http://www.opennaru.com>

본 문서는 오픈나루(opennaru.com)의 자동 설치 제품인 OPENMARU Installer 이용하여 생성된 문서입니다. 본 문서에 대한 저작권은 오픈나루 주식회사에 있습니다.