

JEUS 웹 관리자 안내서



Copyright © 2005 Tmax Soft Co., Ltd. All Rights Reserved.

Copyright Notice

Copyright©2005 Tmax Soft Co., Ltd. All Rights Reserved.

Tmax Soft Co., Ltd.

대한민국 서울시 강남구 대치동 946-1 글라스스타워 18 층 우)135-708

Restricted Rights Legend

This software and documents are made available only under the terms of the Tmax Soft License Agreement and may be used or copied only in accordance with the terms of this agreement. No part of this document may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, or optical, without the prior written permission of Tmax Soft Co., Ltd.

소프트웨어 및 문서는 오직 TmaxSoft Co., Ltd.와의 사용권 계약 하에서만 이용이 가능하며, 사용권 계약에 따라서 사용하거나 복사할 수 있습니다. 또한 이 매뉴얼에서 언급하지 않은 정보에 대해서는 보증 및 책임을 지지 않습니다.

이 매뉴얼에 대한 권리는 저작권에 보호되므로 발행자의 허가 없이 전체 또는 일부를 어떤 형식이나, 사진 녹화, 기록, 정보 저장 및 검색 시스템과 같은 그래픽이나 전자적, 기계적 수단으로 복제하거나 사용할 수 없습니다.

Trademarks

Tmax, WebtoB, WebT, and JEUS are registered trademarks of Tmax Soft Co., Ltd.

All other product names may be trademarks of the respective companies with which they are associated.

Tmax, WebtoB, WebT, JEUS 는 TmaxSoft Co., Ltd.의 등록 상표입니다.

기타 모든 제품들과 회사 이름은 각각 해당 소유주의 상표로서 참조용으로만 사용됩니다.

Document info

Document name: JEUS 웹 관리자 안내서

Document date: 2005-06-06

Manual release version: 3

Software Version: JEUS 5

차 례

1 개요	37
2 소개	39
2.1 개요	39
2.2 JEUS 5 웹 관리자의 새로운 특징들	39
2.2.1 JMX	39
2.2.2 JAXB	40
2.2.3 배치 API	40
2.2.4 디자인 및 UI 컴포넌트	40
2.3 웹 관리자 실행 환경	41
2.4 웹 관리자 구성	41
2.4.1 디렉토리 구조	41
2.4.2 화면 구성	42
2.5 결론	43
3 웹 관리자 시작 및 종료	45
3.1 웹 관리자 시작	45
3.1.1 웹 관리자 설정	45
3.1.2 로그 인	46
3.2 웹 관리자 종료	46
3.2.1 로그아웃	46
3.2.2 웹 관리자 종료	47
4 JEUS 매니저	49
4.1 JEUS 매니저 설정	49

4.1.1	기본 설정	49
4.1.2	리스너 기본 설정	50
4.1.3	리스너의 SSL 설정	51
4.1.4	리스너 쓰레드 풀 설정	52
4.1.5	서비스 설정	52
4.1.6	에러 로그 설정	53
4.2	JEUS 매니저 모니터링	53
4.2.1	장애 모니터링	53
4.2.2	JNDI 트리.....	53
5	클러스터링	55
5.1	클러스터링 구성	55
5.1.1	클러스터링에 노드 추가	55
5.1.2	클러스터링에서 노드 삭제	56
5.2	클러스터링 제어	57
5.2.1	노드 부팅	57
5.2.2	노드 다운	59
5.2.3	노드 종료	59
5.3	JEUSMain.xml 분배	60
5.3.1	JEUSMain.xml 분배	60
6	엔진 컨테이너	63
6.1	엔진 컨테이너 구성	63
6.1.1	엔진 컨테이너 개요	63
6.1.2	새로운 엔진 컨테이너 생성	63
6.1.3	기존의 엔진 컨테이너 복사	64
6.1.4	엔진 컨테이너 삭제	66
6.2	엔진 컨테이너 설정	67
6.2.1	기본 설정	67

6.2.2	에러 로그 설정	68
6.2.3	사용자 로그 설정	68
6.2.4	자동 배치 설정	68
6.2.5	상호 운용 설정	69
6.2.6	라이프 사이클 설정	70
6.2.7	리소스 참조 설정	71
6.2.8	기타 설정	72
6.3	엔진 컨테이너 제어.....	73
6.3.1	엔진 컨테이너 시작	73
6.3.2	엔진 컨테이너 다운	74
6.3.3	엔진 컨테이너 모두 시작	74
6.3.4	엔진 컨테이너 모두 다운	75
6.4	엔진 컨테이너 통계.....	75
6.4.1	엔진 컨테이너 통계	75
6.4.2	JNDI 트리.....	76
7	엔진 개요.....	79
7.1.1	엔진 개요	79
8	EJB 엔진	81
8.1	EJB 엔진 구성	81
8.1.1	EJB 엔진 추가.....	81
8.1.2	EJB 엔진 제거.....	82
8.2	EJB 엔진 설정	83
8.2.1	기본 설정	83
8.2.2	에러 로그 설정	84
8.2.3	활성 관리 (Active Management) 설정	84
8.2.4	HTTP 호출 설정	85
8.2.5	타이머 서비스 기본 설정	87

8.2.6 타이머 서비스 쓰래드 풀	88
8.2.7 타이머 서비스 지속성 설정	89
8.3 EJB 엔진 제어	90
8.3.1 EJB 엔진 시작	90
8.3.2 EJB 엔진 다운	91
8.4 EJB 엔진 통계	91
8.4.1 EJB 모듈 통계	91
8.5 EJBMMain.xml 분배	92
8.5.1 EJBMMain.xml 분배	92
9 서블릿 엔진.....	95
9.1 서블릿 엔진 구성	95
9.1.1 서블릿 엔진 추가	95
9.1.2 서블릿 엔진 삭제	96
9.2 서블릿 엔진 설정	97
9.2.1 기본 설정	97
9.2.2 에러 로그 설정	98
9.2.3 엑세스 로그 설정	98
9.2.4 사용자 로그 설정	100
9.2.5 모니터링 설정	101
9.2.6 세션 클러스터 설정	101
9.3 서블릿 엔진 제어	101
9.3.1 서블릿 엔진 시작	101
9.3.2 서블릿 엔진 다운	102
9.4 WEBMain.xml 분배	103
9.4.1 WEBMain.xml 분배	103
10 컨텍스트 그룹	105
10.1 컨텍스트 그룹 개요	105

10.2 컨텍스트 그룹 구성.....	105
10.2.1 컨텍스트 그룹추가.....	105
10.2.2 컨텍스트 그룹 삭제	107
10.3 컨텍스트 그룹 설정.....	107
10.3.1 기본 설정	107
10.3.2 컨텍스트 추가	109
10.3.3 컨텍스트 삭제	109
10.3.4 가상 호스트 추가	109
10.3.5 가상 호스트 수정	109
10.3.6 가상 호스트 삭제	110
10.3.7 인코딩 설정	110
10.3.8 엑세스 로그 설정	111
10.3.9 사용자 로그 설정	111
10.3.10 JSP 엔진 설정	112
10.3.11 세션 설정	113
10.3.12 응답 헤더 설정	114
10.4 컨텍스트 그룹 제어.....	115
10.4.1 정지(Suspend).....	115
10.4.2 정지 해제(Resume).....	116
10.4.3 종료(Terminate)	116
10.4.4 재시작 (Restart)	117
10.5 컨텍스트 그룹 통계.....	117
10.5.1 웹모듈 통계	117
11 웹 서버 리스너.....	119
11.1 웹 서버 리스너 구성.....	120
11.1.1 웹 서버 리스너 추가	120
11.1.2 웹 서버 리스너 삭제	123

11.1.3 WebToB 리스너 백업 추가.....	123
11.1.4 WebToB 리스너 백업 삭제.....	124
11.2 웹 서버 리스너 설정.....	124
11.2.1 기본 설정	124
11.2.2 쓰레드 풀 설정	125
11.2.3 기타 설정	126
11.3 웹 서버 리스너 제어.....	127
11.3.1 WebToB 리스너 정지(Suspend)	127
11.3.2 WebToB 리스너 계속(Resume)	128
11.4 웹 서버 리스너 통계.....	128
11.4.1 쓰레드 풀 전체 통계	128
11.4.2 쓰레드 풀 개별 통계	129
12 JMS 엔진.....	131
12.1 JMS 엔진 구성	131
12.1.1 JMS 엔진 추가.....	131
12.1.2 JMS 엔진 삭제	132
12.2 JMS 엔진 설정	133
12.2.1 기본 설정	133
12.2.2 서버 채널 설정	134
12.2.3 에러 로그 설정	136
12.2.4 엑세스 로그 설정	136
12.2.5 스토리지 설정	137
12.2.6 쓰레드 풀 설정	140
12.2.7 클러스터 설정	140
12.2.8 듀러블 서브스크라이버 설정	141
12.2.9 큐 클러스터 설정	142
12.2.10 토퍽 클러스터 설정	143

12.3 JMS 엔진	144
12.3.1 JMS 엔진 시작	144
12.3.2 JMS 엔진 다운	145
12.4 JMSMain.xml 분배	145
12.4.1 JMSMain.xml 분배	145
13 웹 서버 엔진.....	147
13.1 웹 서버 엔진 구성	147
13.1.1 웹 서버 엔진 추가	147
13.1.2 웹 서버 엔진 삭제	148
13.2 웹 서버 엔진 설정	149
13.2.1 노드 설정	149
13.2.2 서버 그룹 추가	150
13.2.3 서버 그룹 삭제	151
13.2.4 서버 추가	152
13.2.5 서버 삭제	153
13.2.6 URI 추가	154
13.2.7 URI 삭제	154
13.2.8 별칭(Alias) 추가	155
13.2.9 별칭 삭제	156
13.2.10 확장(Extension)추가	156
13.2.11 확장(Extension) 삭제	157
13.2.12 로깅 추가	157
13.2.13 로깅 삭제	158
13.3 웹 서버 엔진 제어	159
13.3.1 시작	159
13.3.2 다운	159
13.4 웹 서버 엔진 통계	160

14 배치	163
14.1 표준 배치.....	163
14.1.1 J2EE 어플리케이션 모듈 배치.....	163
14.1.2 웹 컨텍스트 배치	168
14.1.3 시작	170
14.1.4 정지	170
14.1.5 제거	172
14.2 영구(Permanent) 배치.....	173
14.2.1 배치	173
14.2.2 Application 설정	174
14.2.3 제거	174
14.3 자동 배치.....	175
15 J2EE 모듈	177
15.1 어플리케이션 모듈.....	177
15.1.1 어플리케이션 모듈 설정	177
15.2 EJB 모듈.....	178
15.2.1 설정	178
15.2.2 통계	178
15.2.3 EJB 통계.....	179
15.3 웹 모듈.....	180
15.3.1 설정	180
15.3.2 통계	180
15.3.3 서블릿 통계	181
15.3.4 서블릿 일시 정지(Suspend).....	181
15.3.5 서블릿 정지 해제	182
15.3.6 서블릿 종료	182
15.3.7 서블릿 재시작	183

15.4 리소스 아답터 모듈.....	183
15.4.1 설정	183
15.4.2 통계	183
15.5 어플리케이션 클라이언트 모듈.....	184
15.5.1 설정	184
16 엔진 컨테이너 리소스.....	185
16.1 JDBC 데이터 소스.....	185
16.1.1 통계	185
16.1.2 가능	186
16.1.3 불가능	186
16.1.4 연결 재조정(Shrink).....	186
16.1.5 XA 트랜잭션 재시작(Resync).....	187
16.1.6 재설정	187
16.2 JCA 리소스	189
16.2.1 연결 통계	189
16.2.2 연결 풀 통계	189
16.2.3 연결 풀의 가능	190
16.2.4 연결 풀의 불가능	190
16.2.5 연결 풀의 연결 재조정(Shrink).....	191
16.2.6 연결 풀의 XA 트랜잭션 재시작	191
16.3 WebT 데이터 소스.....	191
16.3.1 WebT 데이터 소스 통계	191
16.4 트랜잭션 관리자.....	192
16.4.1 기본 설정	192
16.4.2 쓰레드 풀 설정	193
16.4.3 통계	194
16.5 JMS 리소스	195

17 엔진 컨테이너 서비스	197
18 JDBC 데이터 소스	199
18.1 JDBC 데이터 소스 구성	199
18.1.1 추가	199
18.1.2 삭제	203
18.2 JDBC 데이터 소스 제어	204
18.2.1 JDBC 데이터 소스 바인드	204
18.3 클러스터 JDBC 데이터 소스 구성	204
18.3.1 추가	204
18.3.2 클러스터 JDBC 데이터 소스 삭제	206
18.4 클러스터 JDBC 데이터 소스 제어	207
18.4.1 클러스터 JDBC 데이터 소스 바인드	207
19 JMS 리소스.....	209
19.1 JMS 리소스 구성	209
19.1.1 커넥션 팩토리 추가	209
19.1.2 토픽 데스터네이션 추가	211
19.1.3 큐 데스터네이션 추가	212
19.2 JMS 리소스 설정	214
19.2.1 커넥션 팩토리기본 설정	214
19.2.2 커넥션 팩토리 쓰래드 풀 설정	214
19.2.3 데스터네이션 기본 설정	214
19.3 JMS 리소스 제어	215
19.3.1 커넥션팩토리 바인드	215
19.3.2 데스터 네이션 바인드	215
19.3.3 듀러블 서브스크라이버(Durable Subscriber)에서 메시지 삭제	216
19.4 JMS 리소스 통계	216

19.4.1 듀러블 서브스크라이버 통계	216
19.4.2 소비자(Cusumer) 통계	217
19.4.3 클라이언트 통계	217
19.4.4 클라이언트 연결 통계	217
19.4.5 클라이언트 연결 세션 통계	218
20 URL 리소스.....	219
20.1 URL 구성.....	219
20.1.1 추가	219
20.1.2 삭제	219
20.2 URL 설정.....	220
20.2.1 URL 설정	220
21 자바 메일.....	221
21.1 자바 메일 구성	221
21.1.1 자바 메일 추가	221
21.1.2 자바 메일 삭제	222
21.2 자바 메일 설정.....	222
21.2.1 자바 메일 설정	222
22 WebT 데이터 소스.....	223
22.1 WebT 데이터 소스 구성	223
22.1.1 개요	223
22.1.2 추가	223
22.1.3 삭제	226
22.2 WebT 데이터 소스 설정.....	227
22.2.1 기본 설정	227
22.2.2 연결 풀 설정	227
22.2.3 로깅 설정	228
22.2.4 보안 설정	228

22.3 WebT 데이터 소스 제어.....	228
22.3.1 WebT 데이터 소스 바인드	228
22.4 클러스터 WebT 데이터 소스 구성.....	229
22.4.1 추가	229
22.4.2 삭제	230
22.5 클러스터 WebT 데이터 소스 제어.....	231
22.5.1 클러스터 WebT 데이터 소스 바인드	231
23 MQ 리소스.....	233
23.1 MQ 리소스 구성	233
23.1.1 개요	233
23.1.2 추가	233
23.1.3 삭제	235
23.2 MQ 데이터 소스 설정	236
23.2.1 기본 설정	236
23.3 MQ 데이터 소스 제어	236
23.3.1 MQ 데이터 소스 바인드.....	236
24 JEUS 매니저 서비스	237
24.1 네이밍 서버 설정.....	237
24.1.1 네이밍 서버 기본 설정	237
24.1.2 네이밍 서버 쓰레드 풀 설정	238
24.1.3 로컬 네이밍 서버 설정	239
24.2 JMX 서버.....	239
24.2.1 HTTP 아답터 설정	239
24.2.2 JMXMP 설정	240
24.2.3 RMI 아답터 설정	241
24.2.4 SNMP 아답터 설정	242
24.2.5 MLet 등록	243

24.3 클래스 FTP.....	244
24.4 스케줄러.....	244
24.4.1 스케줄러 작업 추가	244
24.4.2 스케줄러 작업 제거	244
24.4.3 스케줄러 작업 설정	244
24.5 로그 서비스.....	245
24.5.1 기본 설정	245
24.5.2 콘솔 핸들러 추가	246
24.5.3 파일 핸들러 추가	247
24.5.4 SMTP 핸들러 추가.....	248
24.5.5 소켓 핸들러 추가	250
24.5.6 사용자 핸들러 추가	251
24.5.7 핸들러 삭제	252
24.5.8 로그 수준 제어	253
25 보안.....	255
25.1 도메인.....	255
25.1.1 도메인 추가	255
25.1.2 도메인 삭제	256
25.2 보안 서비스.....	256
25.2.1 서비스 추가	256
25.2.2 서비스 삭제	257
25.2.3 서비스 수정	257
25.3 서브젝트.....	258
25.3.1 기본 설정에서의 서브젝트 추가	258
25.3.2 기본 설정에서의 서브젝트 수정	259
25.3.3 서브젝트 생성	259
25.3.4 서브젝트 삭제	260

25.3.5 서브젝트 저장	260
25.3.6 서브젝트에 Principal 생성	260
25.3.7 서브젝트에서 Principal 삭제	261
25.3.8 서브젝트에 크리덴셜 팩토리 추가	262
25.3.9 서브젝트에서 크리덴셜 팩토리 삭제	263
25.3.10 서브젝트에 크리덴셜 팩토리 수정	264
25.4 정책.....	264
25.4.1 컨텍스트 식별자 생성	264
25.4.2 컨텍스트 식별자 삭제	265
25.4.3 정책 저장	265
25.4.4 역할 생성	265
25.4.5 역할 삭제	267
25.4.6 역할 수정	268
25.4.7 리소스 생성	268
25.4.8 리소스 삭제	269
25.4.9 리소스 수정	270
26 세션 추적.....	271
26.1 세션 클러스터링 설정	271
26.1.1 세션 클러스터링 설정	271
26.2 분산 세션 서버	272
26.2.1 분산 세션 서버 기본 설정	272
26.2.2 분산 세션 서버 파일 DB 설정	272
26.2.3 분산 세션 서버 백업 설정	272
26.2.4 분산 세션 서버 공통 기본 설정	273
26.2.5 분산 세션 서버 공통 기본 파일 DB 설정	274
26.2.6 분산 세션 서버 통계	274
26.3 중앙집중식 세션 서버.....	275

26.3.1 중앙집중식 세션 서버 기본 설정	275
26.3.2 세션 관리자 추가	275
26.3.3 세션 관리자 삭제	277
26.3.4 세션 관리자 기본 설정	277
26.3.5 세션 관리자 스토리지 설정	278
26.3.6 세션 관리자 모니터링	278
26.3.7 서블릿 엔진에 중앙집중식 세션 서버 연결 설정	279
26.3.8 중앙집중식 세션 서버 통계	280
26.4 로컬 세션 서버.....	281
26.4.1 로컬 세션 서버 통계	281
27 JNLP 서비스.....	283
27.1 JNLP 설정	283
27.1.1 JNLP 서비스 추가	283
27.1.2 JNLP 서비스 설정 변경	285
27.1.3 JNLP 서비스 삭제	286
28 로그 분석 서비스.....	287
28.1 소개.....	287
28.1.1 화면 구성	287
28.1.2 시작	291
28.1.3 종료	293
28.2 로그분석.....	293
28.2.1 온라인 에러 로그분석	293
28.2.2 온라인 액세스 로그 분석	296
28.2.3 오프라인 에러-액세스 로그 분석	299
28.2.4 에러 로그 분석 결과 보기	301
28.2.5 액세스 로그 분석 결과 보기	302
28.2.6 Log Format Pattern	304

28.3 Log Message Monitoring Service.....	307
29 가상 호스트 서비스.....	309
29.1 가상 호스트 설정.....	309
29.1.1 가상 호스트 설정	309
29.2 가상 호스트 구성.....	309
29.2.1 가상 호스트 추가	309
29.2.2 가상 호스트 삭제	310
29.3 가상 호스트 분배.....	311
29.3.1 가상 호스트 분배	311
30 장애 모니터링	313
30.1 장애 모니터링 설정.....	313
30.2 장애 모니터링.....	314
30.2.1 장애 모니터링	314
30.3 장애 모니터링 대상 목록.....	315
30.3.1 EJB 엔진 ejb-active-management-ratio	315
30.3.2 EntityBean thread-max-warning-ratio.....	316
30.3.3 EntityBean thread-max-fatal-ratio	316
30.3.4 Stateless 세션 Bean thread-max-warning-ratio	316
30.3.5 Stateless 세션 Bean thread-max-fatal-ratio	317
30.3.6 Stateful 세션 Bean thread-max-warning-ratio	317
30.3.7 Stateful 세션 Bean thread-max-fatal-ratio	317
30.3.8 MessageDrivenBean thread-max-warning-ratio.....	318
30.3.9 MessageDrivenBean thread-max-fatal-ratio.....	318
A tmonitor.xml XML 요소 참조.....	321
A.1 소개	321
A.2 XML Schema/XML 트리	322
A.3 Element Reference	322
A.4 tmonitor.xml 예제	324

B 아이콘	329
색 인.....	333

그림 목차

그림 1 웹 관리자의 디렉토리 구조.....	41
그림 2 웹 관리자 화면 구성.....	42
그림 3 로그인 폼.....	46
그림 4 로그 아웃 링크.....	46
그림 5 JEUS Manager 설정 선택	49
그림 6 JEUS 매니저의 기본 설정	50
그림 7 JEUS 매니저 리스너의 기본 설정.....	51
그림 8 JEUS 매니저 리스너의 SSL 설정.....	51
그림 9 JEUS 매니저 리스너의 쓰레드 풀 설정.....	52
그림 10 JEUS 매니저 서비스 설정	53
그림 11 JNDI 트리	54
그림 12 JEUS 노드 클러스터링 선택	56
그림 13 JEUS 클러스터에 노드 추가	56
그림 14 클러스터에서 노드 삭제.....	57
그림 15 JEUS Manager Control 선택	58
그림 16 노드 부트.....	58
그림 17 노드 다운.....	59
그림 18 노드 종료.....	60
그림 19 JEUSMain.xml 분배 선택	61
그림 20 JEUSMain.xml 분배	61
그림 21 엔진 컨테이너 생성 선택.....	63
그림 22 엔진 컨테이너 생성.....	64

그림 23 엔진 컨테이너 생성 후 노드 트리.....	64
그림 24 엔진 컨테이너 복사 선택.....	65
그림 25 엔진 컨테이너 복사.....	66
그림 26 엔진 컨테이너 설정 선택.....	67
그림 27 엔진 컨테이너 자동 배치설정.....	69
그림 28 엔진 컨테이너 상호 운용 설정.....	70
그림 29 엔진 컨테이너 라이프 사이클 설정.....	71
그림 30 엔진 컨테이너 리소스 참조 설정.....	72
그림 31 엔진 컨테이너 기타 설정.....	73
그림 32 엔진 컨테이너 시작.....	74
그림 33 엔진 컨테이너다운.....	74
그림 34 엔진 컨테이너모두 시작.....	75
그림 35 엔진 컨테이너 모두 다운.....	75
그림 36 엔진 컨테이너 모니터링.....	76
그림 37 JNDI 트리 보기 선택.....	77
그림 38 Local JNDI 트리.....	78
그림 39 엔진 폴더.....	81
그림 40 EJB 엔진 생성 링크.....	82
그림 41 EJB 엔진 생성 링크.....	82
그림 42 EJB 엔진 제거.....	83
그림 43 EJB 엔진 설정 선택.....	83
그림 44 EJB 엔진 기본 설정.....	84
그림 45 EJB 엔진 HTTP Invocation 설정.....	86
그림 46 EJB 엔진 타이머 서비스 기본 설정.....	87
그림 47 EJB 엔진 타이머 서비스의 지속성 설정.....	89
그림 48 EJB 엔진 시작.....	90
그림 49 EJB 엔진 다운.....	91

그림 50 EJB 엔진 통계	92
그림 51 EJB 엔진 통계 정보	92
그림 52 EJBMMain.xml 분배 선택	93
그림 53 EJBMMain.xml 분배	93
그림 54 새 서블릿 엔진 추가	95
그림 55 서블릿 엔진 추가	96
그림 56 서블릿 엔진 삭제	97
그림 57 서블릿 엔진 설정 선택	97
그림 58 서블릿 엔진 기본 설정	98
그림 59 서블릿 엔진 시작	102
그림 60 서블릿 엔진 다운	102
그림 61 WEBMain.xml 분배 선택	103
그림 62 WEBMain.xml 분배	104
그림 63 컨텍스트 그룹 추가 선택	106
그림 64 컨텍스트 그룹 추가	106
그림 65 컨텍스트 그룹 삭제	107
그림 66 컨텍스트 그룹 설정 선택	108
그림 67 컨텍스트 그룹 기본 설정	108
그림 68 인코딩 설정	111
그림 69 JSP 엔진 설정	113
그림 70 세션 설정	114
그림 71 응답 헤더 설정	115
그림 72 컨텍스트 그룹 제어	116
그림 73 컨텍스트 그룹 모니터링 선택	118
그림 74 컨텍스트 그룹에 배치된 웹 모듈 모니터링	118
그림 75 새 웹 서버 리스너 생성 선택	120
그림 76 웹 서버 리스너 추가 - 1 단계 리스너 종류	120

그림 77 웹 서버 리스너 추가 - 2 단계 기본 설정	121
그림 78 웹 서버 리스너 추가 - 3 단계 쓰레드 풀	121
그림 79 웹 서버 리스너 추가 - 4 단계 기타 설정	122
그림 80 웹 서버 리스너 lsnr2 가 추가된 후 노드 트리	122
그림 81 웹 서버 리스너 삭제.....	123
그림 82 웹 서버 리스너 기본 설정.....	125
그림 83 웹 서버 리스너 쓰레드 풀 설정.....	126
그림 84 WebToB 리스너 정지	128
그림 85 웹 서버 리스너 통계 선택.....	129
그림 86 웹 서버 리스너 모니터링.....	129
그림 87 웹 서버 리스너의 쓰레드 풀 모니터링.....	130
그림 88 JMS 엔진 추가 선택	131
그림 89 JMS 엔진 추가	132
그림 90 JMS 엔진 삭제	133
그림 91 JMS 엔진 설정 선택	133
그림 92 JMS 엔진 기본 설정	134
그림 93 JMS 서버 채널 설정	135
그림 94 JMS 서버 채널 고급 선택 사항	136
그림 95 DBStorage 선택	137
그림 96 DataSource 선택	138
그림 97 FileStorage 선택	139
그림 98 FileStorage 설정	139
그림 99 JMS 엔진 쓰레드 풀설정	140
그림 100 JMS 엔진 클러스터 설정	141
그림 101 JMS 엔진 드러블 서브스크라이버 설정	142
그림 102 JMS 엔진 큐 클러스터 설정	143
그림 103 JMS 엔진 토픽 클러스터 설정	144

그림 104 JMS 엔진 시작	144
그림 105 JMS 엔진 다운	145
그림 106 JMSMain.xml 분배 선택	146
그림 107 JMS 엔진 선택	146
그림 108 웹 서버 엔진 추가 링크.....	147
그림 109 웹 서버 엔진 추가.....	148
그림 110 웹 서버 엔진 삭제.....	149
그림 111 웹 서버 엔진 설정 선택.....	149
그림 112 웹 서버 엔진 노드 설정.....	150
그림 113 서버 그룹 추가.....	151
그림 114 서버 추가.....	153
그림 115 URI 추가.....	154
그림 116 별칭 추가.....	155
그림 117 새 확장 추가.....	157
그림 118 로깅 추가.....	158
그림 119 웹 서버 엔진 시작.....	159
그림 120 웹 서버 엔진 통계 선택.....	160
그림 121 웹 서버 엔진 통계.....	161
그림 122 새 어플리케이션 모듈 배치 선택.....	164
그림 123 어플리케이션 모듈 배치 - 1 단계 모듈 선택	165
그림 124 어플리케이션 모듈 배치 - 2 단계 대상 선택.....	165
그림 125 J2EE 어플리케이션 모듈 배치 - 3 단계 컴포넌트별 선택사항 입력	166
그림 126 J2EE 어플리케이션 모듈 배치 - 3 단계 일반 선택 사항 입력	167
그림 127 J2EE 어플리케이션 모듈 배치 - 4 단계 배치	167
그림 128 J2EE 어플리케이션 모듈 배치 -배치 실패	168
그림 129 웹 컨텍스트 배치 선택.....	169
그림 130 웹 컨텍스트 배치 - 1 단계 배치 디렉토리 선택	169

그림 131 모듈 정지 선택.....	171
그림 132 모듈 정지.....	171
그림 133 Module Stop 후 Module 들의 상태.....	172
그림 134 모듈 제거 선택.....	172
그림 135 모듈 제거.....	173
그림 136 영구 배치 후 모듈이 추가된 모습.....	173
그림 137 영구 모듈의 설정.....	174
그림 138 영구 모듈의 제거.....	175
그림 139 어플리케이션 배치 기술자.....	177
그림 140 EJB 모듈 통계.....	178
그림 141 EJB 통계.....	179
그림 142 웹 모듈의 배치 기술자.....	180
그림 143 웹 모듈 기본 통계.....	181
그림 144 리소스 아답터 모듈 모니터링.....	184
그림 145 JDBC 데이터 소스 모니터링.....	186
그림 146 JDBC 데이터 소스의 보안 재설정.....	188
그림 147 JDBC 데이터 소스의 연결 풀 재설정.....	188
그림 148 JCA 연결의 연결 통계	189
그림 149 JCA 리소스의 연결 풀 통계	190
그림 150 트랜잭션 관리자의 기본 설정.....	193
그림 151 트랜잭션 관리자의 쓰레드 풀 설정.....	194
그림 152 트랜잭션 관리자의 통계.....	195
그림 153 새 JDBC 데이터 소스 생성 선택.....	199
그림 154 JDBC 데이터 소스 생성 - 1 단계 데이터 소스 선택.....	200
그림 155 JDBC 데이터 소스 생성 - 2 단계 속성 설정.....	201
그림 156 JDBC 데이터 소스 생성 - 3 단계 연결 풀 설정.....	202
그림 157 JDBC 데이터 소스 생성 - 4 단계 생성.....	202

그림 158 JDBC 데이터 소스 삭제.....	203
그림 159 JDBC 데이터 소스 바인드.....	204
그림 160 클러스터 JDBC 데이터 소스 생성 선택.....	205
그림 161 클러스터 JDBC 데이터 소스 생성 -1 단계 기본 설정.....	205
그림 162 클러스터 JDBC 데이터 소스 생성 - 2 단계 데이터 소스 선택.....	205
그림 163 클러스터 JDBC 데이터 소스 생성 - 3 단계 데이터 소스 생성.....	206
그림 164 클러스터 JDBC 데이터 소스의 삭제.....	206
그림 165 새 커넥션 팩토리 생성 선택.....	210
그림 166 새 커넥션 팩토리 생성 - 1 단계 기본 설정	210
그림 167 새 커넥션 팩토리 생성 -2 단계 쓰레드 풀 설정	211
그림 168 새 커넥션 팩토리 생성 -3 단계 생성	211
그림 169 새 토픽 데스터네이션 생성 -1 단계 기본 설정	212
그림 170 토픽 테스터네이션 생성 - 2 단계 생성	212
그림 171 새 큐 데스터네이션 생성 -1 단계 기본 설정	213
그림 172 큐 데스터네이션 생성 - 2 단계 생성	213
그림 173 커넥션 팩토리 바인드.....	215
그림 174 URL 리소스 추가.....	219
그림 175 자바메일 리소스.....	221
그림 176 새 WebT 데이터 소스 생성 선택.....	224
그림 177 새 WebT 데이터 소스 생성 - 기본 설정	224
그림 178 새 WebT 데이터 소스 생성 - 연결 풀	225
그림 179 새 WebT 데이터 소스 생성 - 로깅	225
그림 180 새 WebT 데이터 소스 생성 - 기타(보안).....	226
그림 181 새 WebT 데이터 소스 생성 - 생성	226
그림 182 WebT 데이터 소스 개요 - 데이터 소스 목록	227
그림 183 새 클러스터 WebT 데이터 소스 생성 - 기본 설정	229
그림 184 새 클러스터 WebT 데이터 소스 생성 - 데이터 소스들	229

그림 185 새 클러스터 WebT 데이터 소스 생성 - 생성	230
그림 186 노드 트리에서의 WebT 데이터 소스	230
그림 187 클러스터 WebT 데이터 소스 개요	231
그림 188 새 MQ 데이터 소스 생성 - 1 단계 리소스 탑재 설정	234
그림 189 새 MQ 데이터 소스 생성 - 2 단계 속성 설정	234
그림 190 새 MQ 데이터 소스 생성 - 3 단계 생성	235
그림 191 MQ 데이터 소스 삭제	235
그림 192 MQ 데이터 소스 바인드	236
그림 193 네이밍 서버 기본 설정	238
그림 194 네이밍 서버 쓰래드 풀설정	239
그림 195 JMX HTTP 아답터 설정	240
그림 196 JMX JMXMP 아답터 설정	241
그림 197 JMX RMI 아답터 설정	242
그림 198 JMX SNMP 아답터 설정	243
그림 199 새 콘솔 핸들러 생성	246
그림 200 새 파일 핸들러 생성	248
그림 201 새 SMTP 핸들러 생성	249
그림 202 새 소켓 핸들러 생성	250
그림 203 사용자 로그 핸들러 추가	252
그림 204 로그 핸들러 삭제	253
그림 205 로거 트리	254
그림 206 로그 수준 설정	254
그림 207 새 보안 도메인 추가	256
그림 208 서비스 추가	257
그림 209 서브젝트 추가	259
그림 210 Principal 추가	261
그림 211 Credential Factory 추가	263

그림 212 컨텍스트 식별자 생성.....	265
그림 213 역할 추가.....	267
그림 214 Resource 추가.....	269
그림 215 세션 클러스터링 설정.....	271
그림 216 세션 서버 공통 기본 설정.....	273
그림 217 세션 서버 공통 기본 파일 DB 설정	274
그림 218 분산 세션 서버 통계.....	275
그림 219 세션 관리자 생성 - 1 단계 기본 설정	276
그림 220 세션 관리자 생성 - 2 단계 스토리지 설정	277
그림 221 세션 관리자 통계.....	279
그림 222 서블릿 엔진에 중앙집중식 세션 서버 설정.....	280
그림 223 중앙집중식 세션 서버 통계.....	281
그림 224 로컬 세션 서버 통계.....	281
그림 225 노드 트리에서의 JNLP 서비스	284
그림 226 JNLP 서비스 추가	285
그림 227 생성된 JNLP 서비스	285
그림 228 JNLP 서비스 설정	286
그림 229 LogAnalyzer Exploere Main Page.....	288
그림 230 Off-Line Setting View	289
그림 231 Report View	290
그림 232 StackTrace View.....	291
그림 233 LogAnalyzer 선택	292
그림 234 LogAnalyzer Main Window	292
그림 235 Error Log Filter Service - 1 단계 Logger Name 선택	294
그림 236. Error Log Fileter Service - 2 단계 Analysis Rule 정의	295
그림 237 Error Log Filter Service - 3 단계 Analysis Successfully Started	296
그림 238 Access Log Analysis Service - 1 단계 Logger Name 선택.....	297

그림 239 Access Log Analysis Service - 2 단계 Analysis Rule 정의	298
그림 240 로그 포맷 %{Content- Type}i %{J 세션 ID}c %{test1}r %{test2}r %{sess1}s %{sess2}s regular expression 정의	299
그림 241 Logger node 선택	300
그림 242 Analysis Status Task Name List 선택	301
그림 243 Report Display	302
그림 244 Result Report List 선택	303
그림 245 Report Display	304
그림 246 StackTrace Service – 1 단계 Logger Name 선택	307
그림 247 StackTrace Service - 2 단계 FileHandler 의 StackTrace 버튼 선택	308
그림 248 가상 호스트 설정	310
그림 249 가상 호스트 분배	311

표 목차

표 1 JEUS 클러스터에 나타나는 노드들의 상태 아이콘	42
표 2 Access Log Format 의 Pattern ID.....	99
표 3 웹 서버 리스너 쓰레드 풀의 쓰레드 상태 종류.....	130
표 4 webaccess.properties sample	298
표 5 Error Log Format 의 Pattern Attribute	304
표 6 Access Log Format 의 Pattern Attribute.....	305
표 7 장애 통계 목록 : EJB 엔진 ejb-active-management-ratio.....	315
표 8 장애 통계 목록 : EntityBean thread-max-warning-ratio	316
표 9 장애 통계 목록 : EntityBean thread-max-fatal-ratio	316
표 10 장애 통계 목록 : Stateless 세션 Bean thread-max-warning-ratio	316
표 11 장애 통계 목록 : Stateless 세션 Bean thread-max-fatal-ratio	317
표 12 장애 통계 목록 : Stateful 세션 Bean thread-max-warning-ratio.....	317
표 13 장애 통계 목록 : Stateful 세션 Bean thread-max-fatal-ratio.....	317
표 14 장애 통계 목록 : MessageDrivenBean thread-max-warning-ratio.....	318
표 15 장애 통계 목록 : MessageDrivenBean thread-max-fatal-ratio.....	318

매뉴얼에 대해서

매뉴얼의 대상

본 매뉴얼은 JEUS 웹 관리자를 이용해서 JEUS에 대해 상호 작용을 원하는 사용자를 위해 만들었다. 본 매뉴얼은 JEUS 관리자와 어플리케이션 배치자에게 특별히 유용하다. 또한 관리와 어플리케이션 배치를 위한 도움이 되는 기능에 대한 유용한 정보들을 찾을 수 있다.

매뉴얼의 전제 조건

본 매뉴얼을 사용하기 전에 반드시 JEUS 소개뿐 아니라 JEUS Server 안내서의 내용에 대한 지식이 있어야 한다. 본 매뉴얼을 읽기에 앞서 JEUS Server 안내서를 읽을 것을 권한다. 그리고, JEUS Server 안내서를 포함한 기타 다른 매뉴얼을 읽으면서 본 매뉴얼을 참조할 수 있다.

매뉴얼 구성

시스템을 관리하는 가장 편리한 방법을 제공하는 웹 관리자에 대하여 알아야 한다.

비록 웹 관리자를 사용하는 구체적인 방법이 각 매뉴얼에 상세하게 설명이 될지라도, 여러분들은 웹 관리자의 일반적인 사용에 대한 지식이 필요할 것이다.

본 매뉴얼은 첫째, 관리자와 배치자를 위해 웹 관리자 그 자체의 일반적인 개요를 제공한다. 또한, 웹 관리자에 대한 참조 안내자로서 매뉴얼을 사용할 수 있다.

이 매뉴얼은 각각의 서버 구성 요소에 대한 사용방법을 상세하게 실었다. 총 29개의 장과 2개의 부록으로 구성되어 있다.

관련 매뉴얼

본 매뉴얼은 GUI Tool 사용자에게 도움을 줄 것이다. 다음의 매뉴얼을 읽으면서 본 매뉴얼을 참조할 수 있다.

- JEUS Server 안내서
- JEUS Web Container 안내서
- JEUS EJB 안내서
- JEUS JMS 안내서
- JEUS Client Application 안내서

일러두기

표기 예	내용
텍스트	본문, 12 포인트, 바탕체 Times New Roman
텍스트	본문 강조
CTRL+C	Ctrl 과 C 를 동시에 누름
public class myClass { }	Java 코드
<system-config>	XML 문서
참고: / 주의:	참고 사항과 주의 할 사항
설정 메뉴를 연다	UI 의 버튼과 같은 컴포넌트
JEUS_HOME	JEUS 가 실제로 설치된 디렉토리
JEUS_BASEPORT	JEUS 에서 사용하는 BASEPORT

표기 예	내용
j eusadmi n nodename	콘솔 명령어와 문법
[파라미터]	옵션 파라미터
< xyz >	'<'와 '>' 사이의 내용이 실제 값으로 변경됨
	선택사항
...	파라미터 등이 반복 되어서 나옴
? , + , *	보통 XML 문서에 각각 “없거나, 한번”, “한 번 이상”, “없거나 여러 번”을 나타낸다.
....	XML이나 코드 등의 생략
<<FileName.ext>>	코드의 파일명
그림1.	그림 이름이나 표 이름

OS에 대해서

본 문서에서는 모든 예제와 환경 설정을 Microsoft Windows™의 스타일을 따랐다. 유닉스같이 다른 환경에서 작업하는 사람은 몇 가지 사항만 고려하면 별무리 없이 사용할 수 있다. 대표적인 것이 디렉토리의 구분자인데, Windows 스타일인 “\”를 유닉스 스타일인 “/”로 바꿔서 사용하면 무리가 없다. 이외에 환경 변수도 유닉스 스타일로 변경해서 사용하면 된다.

그러나 Java 표준을 고려해서 문서를 작성했기 때문에, 대부분의 내용은 동일하게 적용된다.

용어 설명

다음에 소개되는 용어는 본 문서 전체에 걸쳐서 사용되는 용어이다. 용어가 이해하기 어렵거나 명확하지 않을 때는 아래 정의를 참조하기 바란다.

Term	Definition
JEUS Builder	JEUS Application Builder
JEUS DD	JEUS 배치 Descriptor 의 약어.
Module DD	J2EE Specific 배치 Descriptor 의 약어
WebManager	JEUS 의 웹 기반 관리 도구

연락처

Korea

Tmax Soft Co., Ltd
18F Glass Tower, 946-1, Daechi-Dong, Kangnam-Gu, Seoul 135-708
South Korea
Tel: 82-2-6288-2114
Fax: 82-2-6288-2115
Email: info@tmax.co.kr
Web (Korean): <http://www.tmax.co.kr>

USA

Tmax Soft, Inc.
560 Sylvan Ave, Englewood Cliffs NJ 07632
USA
Tel: 1-201-567-8266
FAX: 1-201-567-7339
Email: info@tmaxsoft.com
Web (English): <http://www.tmaxsoft.com>

Japan

Tmax Soft Japan Co., Ltd.
6-7 Sanbancho, Chiyoda-ku, T 확인 yo 102-0075
Japan
Tel: 81-3-5210-9270
FAX: 81-3-5210-9277
Email: info@tmaxsoft.co.jp
Web (Japanese): <http://www.tmaxsoft.co.jp>

China

Beijing Silver Tower, RM 1507, 2# North Rd Dong San Huan,
Chaoyang District, Beijing, China, 100027
Tel: 86-10-6410-6148
Fax: 86-10-6410-6144
E-mail : info@tmaxchina.com.cn
Web (Chinese): <http://www.tmaxchina.com.cn>

1 개요

JEUS 웹 관리자는 웹을 통해서 JEUS 를 관리할 수 있는 사용자 인터페이스이다. 비록 명령 라인에서 사용할 수 있는 여러 도구들이 존재하지만 JEUS 의 모든 요소들을 관리하기에는 그 기능들이 제한적이다. 웹 관리자는 JEUS 의 모든 요소들을 포괄적으로 관리 할 수 있는 기능들을 제공한다.

우선 종합적인 시스템을 관리할 수 있는 통합된 인터페이스를 제공한다.

또한, 시스템을 운영하는 관리자가 JEUS 웹 관리자를 통해서 JEUS 의 각 구성 요소들의 상태를 실시간으로 모니터링하고 제어할 수 있도록 한다.

본 매뉴얼은 총 30 개의 장과 2 개의 부록으로 이루어져 있다. 각 장은 JEUS 의 각 구성요소에 대해서 설명한다. 또한 각 장은 각 구성 요소의 구성, 설정, 제어, 모니터링으로 나누어서 상세히 설명하였다. 그리고 마지막으로 3 개의 부록에서는 데이터 소스와 장애모니터링을 위한 XML 요소 참조와 웹 관리자에서 사용하는 아이콘에 대해서 실었다.

2 소개

2.1 개요

JEUS 4.X 이하 버전에서는 JEUS 를 관리하기 위한 도구로서 스윙 기반의 JManager 와 웹 기반의 WebManager 를 제공하였었다. 그러나 두 어플리케이션의 기능들이 중복되어 역할의 분리가 확실하지 않아 사용하기 불편한 점이 있었다. 이에 JEUS 5 에서는 모든 JEUS 의 구성요소와 어플리케이션 모듈들의 배치 및 관리를 웹 기반의 웹 관리자를 통해서 하고 EAR 이나 JAR, WAR 같은 어플리케이션 모듈들을 작성하고 패키징하는 기능은 스윙 기반의 JEUS Builder 를 통해서 할 수 있도록 역할을 확실히 분리하였다.

이번 장에서는 우선 JEUS 5 웹 관리자의 새로운 특징들에는 어떠한 것이 있는지 알아 볼 것이다. 다음, 웹 관리자를 실행시키기 위한 환경에 대해서 알아보고, 마지막으로 JEUS 웹 관리자의 디렉 토리 구성과 화면 구성에 대해서 개략적으로 알아 보고 결론을 내릴 것이다.

2.2 JEUS 5 웹 관리자의 새로운 특징들

JEUS 5 부터 JEUS 서버 내부적으로 많은 변화가 생겼다. 특히 관리상의 관점에서 봤을 때 JMX, JAXB, 배치 API 의 추가가 가장 큰 변화라고 할 수 있다. 이번 절에서는 이들이 새롭게 추가됨으로서 웹 관리자에 어떠한 새로운 특징들이 추가되었는지와 이전 버전에서의 웹 관리자와 무엇이 다른지에 대해서 알아볼 것이다.

2.2.1 JMX

JMX 는 *Java Management Extension* 의 약자로서 *Web Application Server* 를 관리하기 위한 표준적인 방법이다. JMX 서버내에는 웹 어플리케이션 서버의 각 구성요소들을 대신하여 관리를 해주는 MBean 들이 존재한다. MBean 은 각 구성요소의 속성을 접근하거나 상태를 모니터링하고 제어를 할 수 있는 기능들을 제공한다.

JEUS 5 웹 관리자는 JEUS 구성 요소들에 대한 모든 관리를 각 MBean 들의 메소드 호출을 통해서 하고 있다.

각 구성 요소들에 대한 MBean 과 제공되는 속성 및 메소들에 대한 자세한 정보는 JMX 안내서와 JMX javadoc 을 참조하기 바란다.

2.2.2 JAXB

JAXB 는 XML 문서를 Java 객체를 통해서 접근하는 방법을 제공한다. JAXB 객체들은 XML Schema 파일로부터 생성되며 어플리케이션들은 이 JAXB 객체들을 이용함으로서 XML 문서를 직접 다룰 필요가 없어졌다.

JEUS 5 의 모든 설정 파일들 - *JEUSMain.xml*, *WEBMain.xml*, *EJBMain.xml*, *WSMain.xml* 등등 –을 위한 XSD 파일들은 XML Schema 파일들로 재정의 되었으며 이를 기반으로 하여 XML Schema 파일들을 JAXB 객체들로 전환하였다. *JEUS_HOME/lib/system/jeus.jar* 파일의 *jeus.xml.binding* 이하에 존재하는 패키지들은 각 XML Schema 파일과 관련된 JAXB 객체들을 나타낸다.

JEUS 5 웹 관리자는 JEUS 내의 모든 설정을 JAXB 를 통해서 함으로서 요소의 순서가 바뀜으로서 생기는 문제라든가 각 요소의 값을 XML Schema에 정의한 타입으로 설정하지 않았을 때 생기는 유효검사 문제등에 대해서 신경쓰지 않고 XML 파일들을 관리할 수 있게 되었다.

JAXB 에 대한 내용은 JEUS Server 안내서를 참조하기 바란다.

2.2.3 배치 API

J2EE 1.4 이전 버전에서의 어플리케이션 모듈들의 배치는 각 웹 어플리케이션 서버가 제공하는 인터페이스를 통해서만 이루어졌다. 그러나 J2EE 1.4 버전부터는 모든 웹 어플리케이션 서버가 배치 API 를 반드시 구현해야만한다. 이에 따라 웹 어플리케이션 서버가 **배치 API** 를 구현하게 된다면 어떠한 어플리케이션에서도 코드의 변경없이 표준적인 방법으로 어플리케이션 모듈들을 배치할 수 있게되었다.

JEUS 5 도 J2EE 1.4 스펙을 준수하는 배치 API 를 완벽하게 구현하였으며 웹 관리자에서는 이 API 를 통해서 배치관련 기능들을 구현하였다.

2.2.4 디자인 및 UI 컴포넌트

JEUS 5 웹 관리자에서 또 다른 주요 변경사항은 전체적인 화면 디자인의 변경이다. 사용에 있어서의 편리함만을 추구한 것이 아니라 눈에서 편안하게 받아들일 수 있도록 하여 작업하는 동안 피로함을 느끼지 않게 디자인 면에서 대폭적으로 변경되었다. 또한 각 UI 컴포넌트들은 사용성을 고려하여 제작되어 관리자가 좀 더 편리하게 웹 관리자를 사용하게 하였다.

2.3 웹 관리자 실행 환경

웹 관리자를 실행하기 위해서는 최소 다음의 환경을 갖추어야 한다.

1. 운영체제
모든 OS에서 사용 가능
2. HTTP 브라우저
인터넷 익스플로러 5.0 이상
넷스케이프 6.0 이상

위의 사양은 최소 사양이며 윈도우 2000 이상의 운영체제를 갖춘 PC에서 인터넷 익스플로러 5.5 이상의 버전을 사용할 것을 권장한다.

2.4 웹 관리자 구성

이 번 절에서는 JEUS 5 웹 관리자의 디렉토리 구조와 주요 화면 구성에 대해서 간략하게 소개할 것이다. 좀 더 구체적인 설명은 이후 관련된 장에서 자세히 설명 할 것이다.

2.4.1 디렉토리 구조

다음은 JEUS 5 웹 관리자의 디렉토리 구조를 타나낸다.

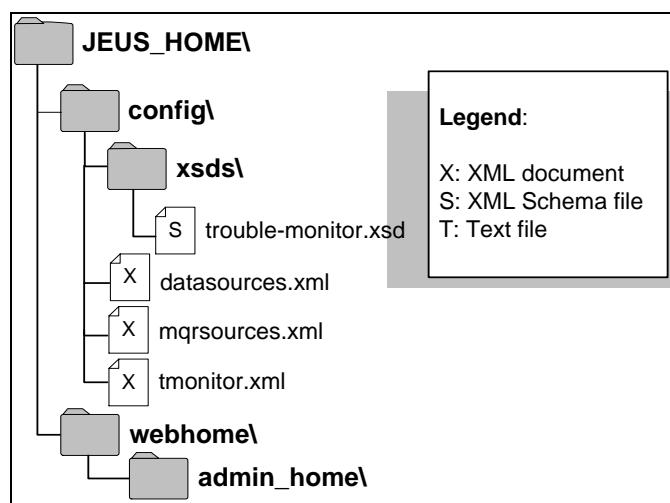


그림 1 웹 관리자의 디렉토리 구조

웹 관리자는 *JEUS_HOME/webhome/admin_home* 디렉토리에 존재하며 웹 관리자가 직접 관리 하는 XML 문서 파일이 두 개 존재한

다. “*tmonitor.xml*” 파일은 장애 모니터링에 필요한 한계값이나 제약 값들을 정의하는 파일이고, “*datasources.xml*”과 “*mqresources.xml*” 파일은 각각 JDBC 데이터 소스와 MQ 데이터 소스를 설정하기 위해서 각 데이터 소스에 대한 속성들을 정의해 놓은 파일이다. 이들 파일들의 내용을 사용자가 변경하게 되면 웹 관리자에 바로 적용된다. 이들 XML 문서에서 사용하는 요소들에 대한 구체적인 설명은 부록을 참조하기 바란다.

2.4.2 화면 구성

JEUS 웹 관리자의 화면은 크게 *Cluster View*, *Node View*, *Main View* 3 개의 부분으로 나뉘어 진다.



그림 2 웹 관리자 화면 구성

- Cluster View:** Cluster View는 클러스터링된 JEUS 노드들의 목록을 보여주는 화면이다. 각 노드들은 상태에 따라 아이콘이 변경되며 좌측 상단에 위치한 갱신버튼(刷新)을 클릭하면 노드 목록이 갱신된다.

표 1 JEUS 클러스터에 나타나는 노드들의 상태 아이콘

아이콘	상태	설명
	Running	노드가 정상적으로 동작중인 상태
	Stopped	노드가 다운된 상태이나 JEUS 매니저는 동작 중인 상태. 이 경우에는 노드 아이콘 오른쪽의 노드 이름에 링크가 있어서 링크를 클릭 할 경우

		우 설정을 변경할 수 있다.
	Failed	노드나 그 하위 구성요소들에 문제가 생겼을 경우.
	Downed	노드가 완전 다운되었을 경우. 이 경우는 JEUS 매니저도 다운된 상태라서 설정변경도 불가능하다.

- **Node View:** Node View는 Cluster View에서 선택된 혹은 기본적으로 웹 관리자가 실행되고 있는 노드의 구조가 트리 형태로 나타나는 화면이다. 이 화면 역시 Cluster View와 마찬가지로 좌측 상단에 위치한 간단 버튼을 누르면 전체 트리가 간단된다. 트리에 사용된 아이콘들에 대한 설명은 부록 B를 참조하라.
- **Main View:** Main View는 웹 관리자에서의 실제적인 작업공간이다. Node View의 트리에서 어떠한 구성요소를 클릭하거나 컨텍스트 메뉴를 클릭하였을 때 이 Main View에 내용들이 출력된다. Main View는 크게 Header, Navigation, Body 3 부분으로 나뉜다. Header 부분은 웹 관리자를 사용하고 있는 사용자에 대한 정보와 흡, 설정, 로그아웃, 도움말과 같은 각종 유ти리티 메뉴를 보여주며 Navigation은 현재 어디서 작업이 이루어지는 보여준다. 마지막으로 Body 부분은 실제 작업 내용들을 보여준다.

2.5 결론

이번 장에서는 JEUS 5 웹 관리자의 새로운 특징들과 화면 구성에 대해서 간략하게 알아보았다.

새로이 개발된 JEUS 5의 웹 관리자를 통해서 시스템 관리자는 이전 보다 더욱 편리하고 종합적으로 JEUS의 각 구성요소들을 관리 할 수 있게 되었으며 어플리케이션 배치자들은 어플리케이션 모듈들을 빠르고 쉽게 배치할 수 있게 될 것이다.

3 웹 관리자 시작 및 종료

이번 장에서는 웹 관리자를 사용하기에 앞서, 웹 관리자를 시작하고 종료하는 방법에 대해서 설명할 것이다.

3.1 웹 관리자 시작

3.1.1 웹 관리자 설정

웹 관리자를 시작하기 위해서는 *JEUSMain.xml* 파일의 “/jeus-system/node/enable-webadmin” 요소를 “true”로 설정하여야 한다. 다음은 JEUS 부팅시 웹 관리자를 시동하도록 설정한 “*JEUSMain.xml*” 파일의 일부를 보여주고 있다.

```
<<JEUSMain.xml>>
<jeus-system>
    <node>
        <name>sebong</name>
        <class-ftp>true</class-ftp>
        <sequential-start>false</sequential-start>
        <enable-webadmin>true</enable-webadmin>
        . . .
    </node>
    . . .
</jeus-system>
```

3.1.2 로그 인

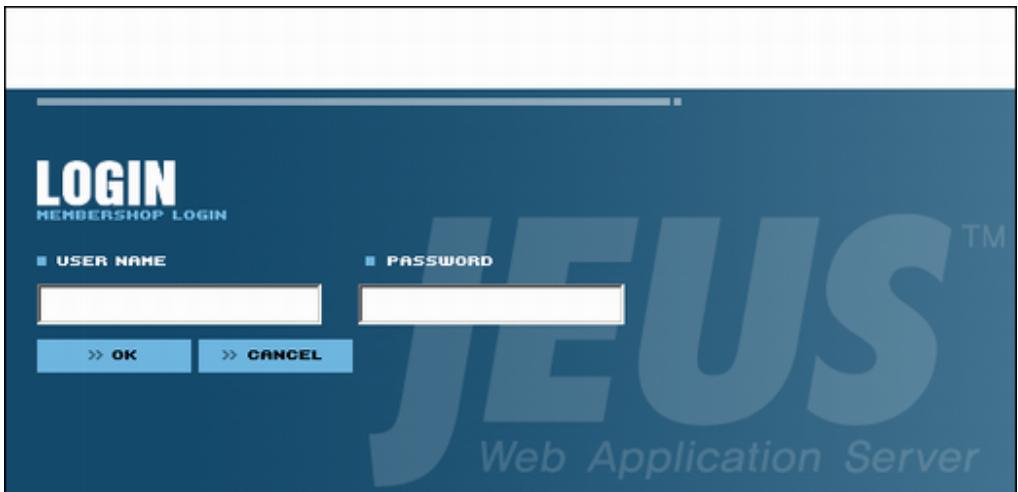


그림 3 로그인 폼

1. 웹 브라우저의 주소 창에 "*http://<IP address>:9744/webadmin*"을 입력한다. 9744는 JEUS_BASEPORT에 8을 더한 값이며 JEUS_BASEPORT의 기본값은 9736이다. 설정한 JEUS_BASEPORT에 맞게 입력하면 된다.
2. 로그인 페이지에서 관리자 아이디와 암호를 입력한 후 확인 버튼을 클릭한다.
3. 로그인 성공시 관리자 메인 페이지로 이동되며 실패시 오류 메시지가 출력된다.

3.2 웹 관리자 종료

3.2.1 로그아웃

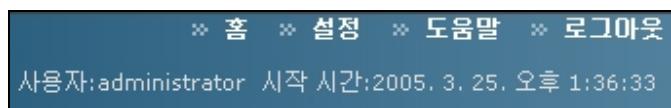


그림 4 로그 아웃 링크

1. 로그인을 한 후 주화면 상단에 위치한 로그아웃 링크를 클릭한다.
2. 로그아웃을 클릭하면 웹 관리자 사용자 세션이 종료되며 로그인 페이지로 이동된다.

3.2.2 웹 관리자 종료

웹 관리자는 JEUS 가 다운될 때 같이 종료된다. 따라서 JEUS 노드를 다운시킬 때 다운시키고자 하는 노드 목록에 웹 관리자가 실행되고 있는 노드가 포함되어 있는지 확인해야 한다.

4 JEUS 매니저

JEUS 매니저는 JEUS의 하위 시스템을 광범위하게 관리하기 위한 하나의 어플리케이션이다. 보통 노드와 JEUS 매니저는 같은 것은 아니지만 매우 밀접하게 관련되어 있다. 각 JEUS 매니저는 반드시 한 개의 노드를 관리하고, 각 노드는 반드시 한 개의 JEUS 매니저에 의해서 관리된다. 이런 이유로 인해 JEUS 웹 관리자에서의 JEUS 매니저 관리는 노드 관리까지 포함하고 있다.

4.1 JEUS 매니저 설정

4.1.1 기본 설정

JEUS 매니저 > 설정 > 기본 설정

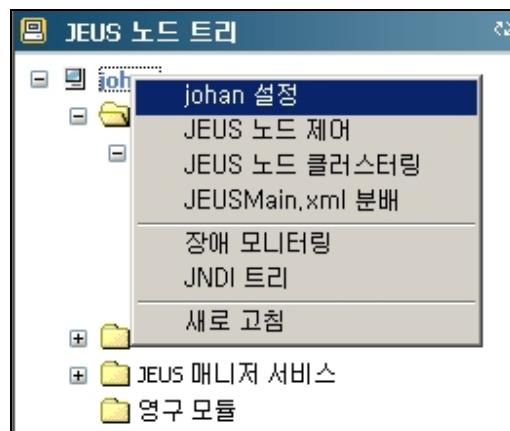


그림 5 JEUS Manager 설정 선택

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하거나 컨텍스트 메뉴에서 **johan 설정** 메뉴를 클릭한다.
2. 주 화면의 탭에서 **설정**의 **기본 설정** 탭을 선택한다.
3. 장애시 백업할 노드의 이름과 순차적인 시작을 사용할 것인지 여부에 대한 설정을 한다. 설정된 값들은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.



그림 6 JEUS 매니저의 기본 설정

4.1.2 리스너 기본 설정

JEUS 매니저 > 설정 > 리스너 > 기본 설정

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하거나 컨텍스트 메뉴에서 **johan** 설정 메뉴를 클릭한다.
2. 주 화면에 나타나는 탭에서 설정의 리스너 탭을 선택한다.
3. 화면에 나타난 안쪽 탭에서 기본 설정 탭을 선택한다.
4. **Backlog** 값을 설정한다. 설정된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭한다.

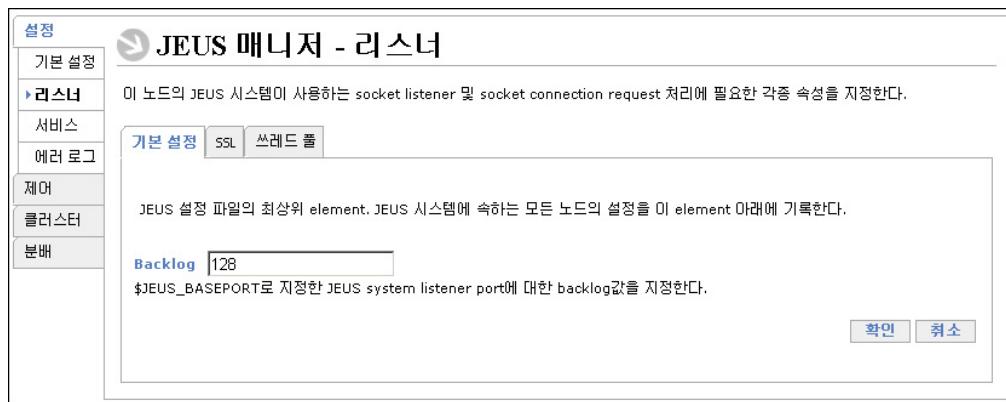


그림 7 JEUS 매니저 리스너의 기본 설정

4.1.3 리스너의 SSL 설정

JEUS 매니저 > 설정 > 리스너 > SSL

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하거나 컨텍스트 메뉴에서 **johan** 설정 메뉴를 클릭 한다.
2. 주 화면의 탭에서 설정의 리스너탭을 선택한다.
3. 화면에 나타난 안쪽 탭에서 **SSL** 탭을 선택한다.
4. SSL을 사용시 대기할 포트 번호를 입력한다. 설정된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭 한다.



그림 8 JEUS 매니저 리스너의 SSL 설정

4.1.4 리스너 쓰레드 풀 설정

JEUS 매니저 > 설정 > 리스너 > 쓰레드 풀

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하거나 컨텍스트 메뉴에서 **johan 설정** 메뉴를 클릭한다.
2. 주 화면의 탭에서 설정의 리스너탭을 선택한다.
3. 화면에 나타난 안쪽의 탭에서 쓰레드 풀 탭을 선택한다.
4. 쓰레드 풀의 쓰레드들의 최소값과 최대값 그리고 풀내에 존재하는 쓰레드들을 검사할 주기를 입력한다. 최대값과 검사 주기는 런타임에 적용되며 최소값의 설정은 런타임시 의미가 없음으로 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭한다.



그림 9 JEUS 매니저 리스너의 쓰레드 풀 설정

4.1.5 서비스 설정

JEUS 매니저 > 설정 > 서비스

일반적인 서비스들은 각각 자신을 관리하는 MBean 을 가지고 있으며 독립적인 페이지에서 관리되나 JEUS 매니저의 클래스 FTP, JNLP, 웹 관리자와 같은 단순한 서비스들은 이 화면에서 관리한다.

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하거나 컨텍스트 메뉴에서 **johan 설정** 메뉴를 클릭한다.

2. 주 화면에 나타난 탭에서 설정의 서비스탭을 선택한다.
3. 각각 서비스들을 사용할지 여부를 선택한다. 설정된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

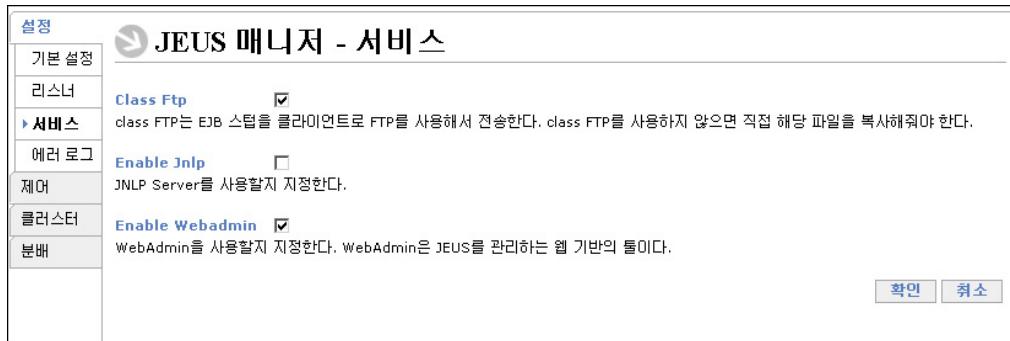


그림 10 JEUS 매니저 서비스 설정

4.1.6 에러 로그 설정

JEUS 매니저 > 설정 > 에러 로그

에러 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

4.2 JEUS 매니저 모니터링

4.2.1 장애 모니터링

JEUS 매니저 > 장애 모니터링

30 장애 모니터링에서 설명

4.2.2 JNDI 트리

JEUS 매니저 > JNDI 트리

이 메뉴는 JEUS 매니저에 원격(Remote)바인딩된 객체들을 보여준다. 일반적으로 각종 리소스나 Remote EJB의 바인딩 여부를 확인할 때 사용한다.

1. 노드 트리에서 최상위에 위치한 JEUS 매니저의 컨텍스트 메뉴에서 **JNDI 트리** 메뉴를 클릭하거나 설정 화면의 상단에 위치한 **JNDI 트리** 링크를 클릭한다.

2. JEUS 매니저에 바인딩된 객체들의 트리가 나타난다.
3. 좌측의 트리에서 노드를 선택하여 클릭하면 바인딩 이름과 클래스 이름, 해시 코드가 출력된다.

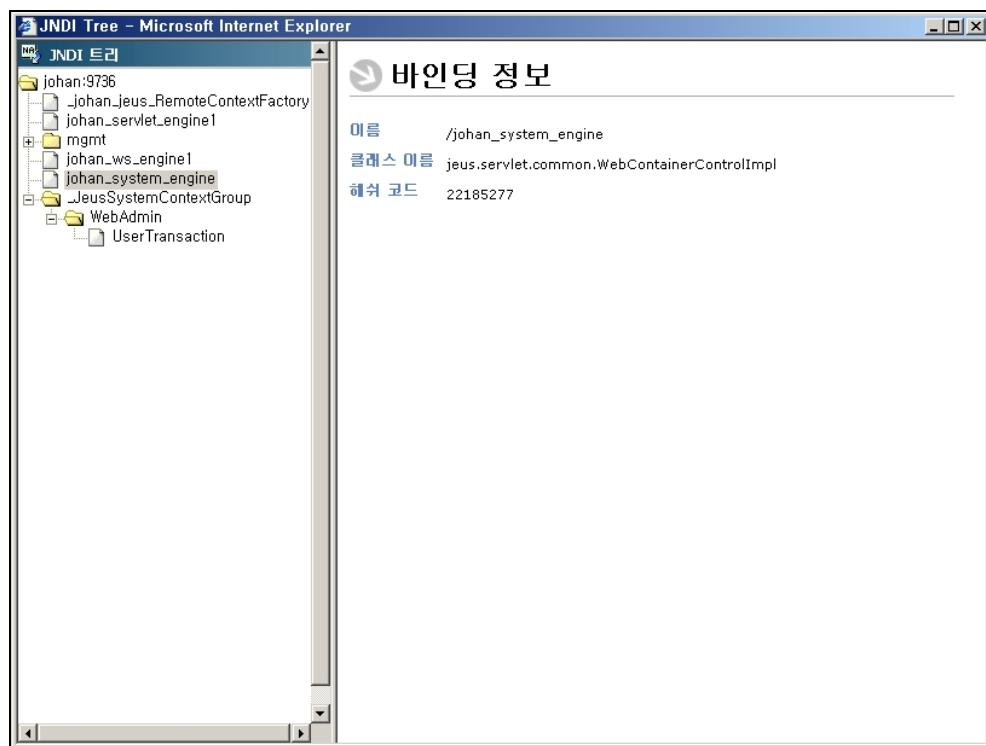


그림 11 JNDI 트리

5 클러스터링

일반적으로 시스템 과부하를 신뢰성있게 처리하기 위해 여러 JEUS를 연결해서 하나의 클러스터링을 구성한다.

이 번장에서는 웹 관리자를 통해서 클러스터링을 어떻게 구성하고 클러스터링된 노드들을 어떻게 제어하는지에 대해서 알아보도록 할 것이다.

5.1 클러스터링 구성

5.1.1 클러스터링에 노드 추가

JEUS 매니저 > 클러스터링 > 추가

클러스터링을 구성하는 방법에는 정적인 방법과 동적인 방법이 존재한다.

정적으로 클러스터링을 구성한다는 것은 JEUSMain.xml에 클러스터링에 참여하는 노드들을 미리 등록하고 부팅시에 클러스터링을 구성하는 것을 말한다. JEUS 웹 관리자를 통해서 정적 클러스터링을 구성하기 위해서는 다음의 조건을 만족해야 한다.

- 클러스터링에 참여하고자 하는 노드들의 JEUS 매니저가 동작중 - "JEUS_HOME/bin/jeus" 명령을 수행 시킴-이어야 한다.
- 각 노드들은 시스템의 hosts 파일에 등록되어 있거나 JEUS의 가상 호스트 설정 파일 - JEUS_HOME/config/vhost.xml-에 등록되어 있어야 한다.
- 가상 호스트가 아닌 실제 물리적인 노드를 사용할 경우에는 JEUS_BASEPORT 가 같아야 한다.
- 가상 호스트는 가상 호스트끼리 물리적 노드는 물리적 노드끼리만 클러스터링이 가능하다. 가상 호스트 설정은 29 가상 호스트 서비스를 참조하기 바란다.

참고 : 동적인 클러스터링 대한 설명은 JEUS Server 안내서를 참고하기 바란다.

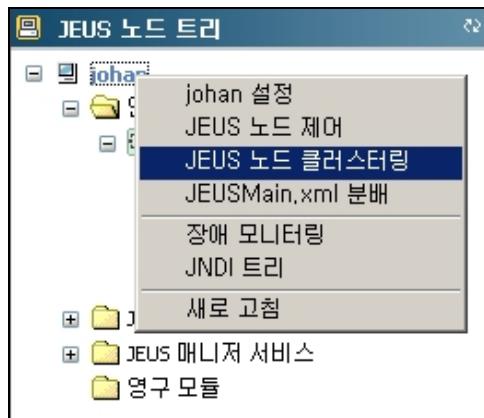


그림 12 JEUS 노드 클러스터링 선택

1. 노드 트리에서 최상위에 위치한 JEUS 매니저를 바로 클릭하여 주화면에 나타난 팝업에서 클러스터링 탭을 선택하거나 컨텍스트 메뉴에서 JEUS 노드 클러스터링 메뉴를 클릭한다. 메뉴를 클릭하면 주화면에 현재 클러스터링 되어있는 노드가 트리형식으로 나타난다.
2. 노드 필드에 추가하고자 하는 노드 이름을 입력한다.
3. 추가 버튼을 클릭한다.
4. 노드가 성공적으로 추가되면 Cluster View에 등록된 노드가 추가된다.

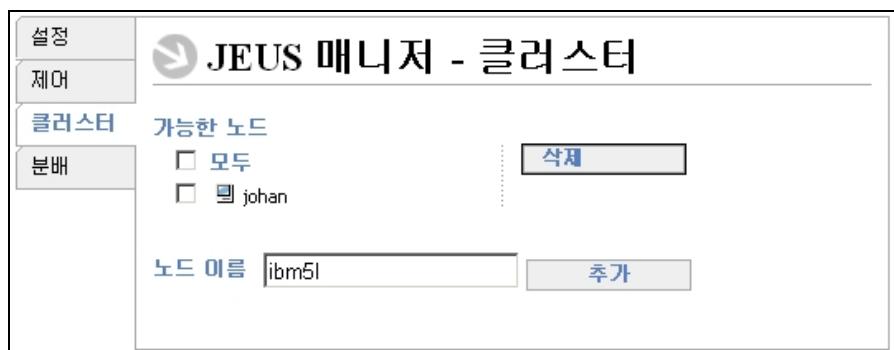


그림 13 JEUS 클러스터에 노드 추가

참고 : 클러스터링에 추가된 노드들이 비록 JEUS 매니저만 동작하는 경우라도 Cluster View의 노드를 클릭하면 Node View에서 XML 파일들에 설정된 값에 따라 트리가 나타나며 각각의 값들을 수정을 할 수 있다.

5.1.2 클러스터링에서 노드 삭제

JEUS 매니저>클러스터링> 삭제

JEUS 의 클러스터링에서 노드를 삭제하는 것은 단순히 JEUSMain.xml 의 노드 목록에서 삭제하는 것을 말한다. 곧 운영중인 클러스터링에서의 동적 삭제는 불가능 하며 노드를 삭제하고 전체 시스템을 리부팅 할 경우에만 클러스터링에서 삭제가 완전히 되므로 주의해야 한다.

1. 노드 트리에서 최상위에 위치한 **JEUS 매니저**를 바로 클릭한 후 주화면에 나타난 탭에서 **클러스터링** 탭을 선택하거나 컨텍스트 메뉴에서 **JEUS 노드 클러스터링** 메뉴를 클릭한다.
2. 화면 안쪽의 **노드 탐색기**에서 삭제하고자 하는 노드를 선택한다.
3. **삭제** 버튼을 클릭한다.
4. 노드가 성공적으로 삭제되면 **Cluster View**에 등록된 노드가 삭제된다.

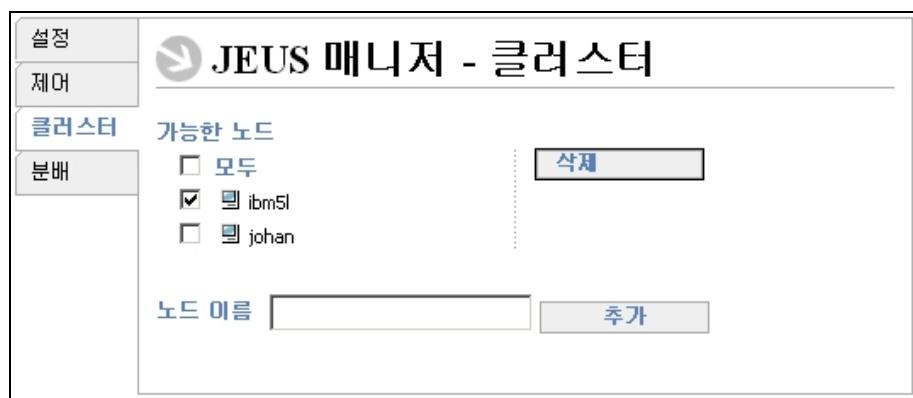


그림 14 클러스터에서 노드 삭제

참고 : 현재 동작중이던 클러스터링에서 노드들을 삭제한 후에는 클러스터링에 참여했던 노드들이 모두 리부팅 하기 전까지 다른 새로운 노드를 추가하지 않도록 해야한다. 이런 경우 클러스터링된 노드들의 순서가 엉켜버릴 수가 있다.

5.2 클러스터링 제어

5.2.1 노드 부팅

[JEUS 매니저](#) > [제어](#) > [부트](#)

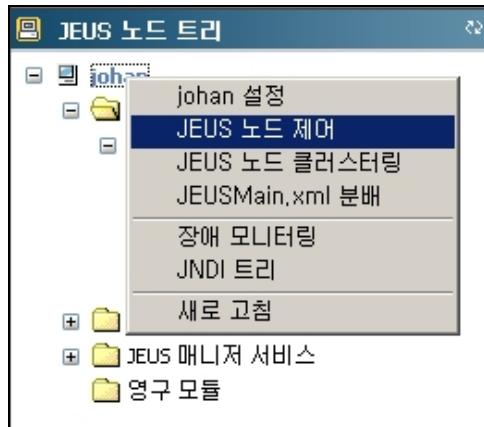


그림 15 JEUS Manager Control 선택

1. 노드 트리에서 최상위에 위치한 **JEUS 매니저**를 바로 클릭한 후 주화면에 나타난 템에서 **제어** 템을 선택하거나 컨텍스트 메뉴에서 **JEUS 노드 제어** 메뉴를 클릭한다.
2. 안쪽의 템 패널에서 **부트** 템을 선택한다.
3. 노드 탐색기에서 부팅하고자 하는 노드들을 선택한다.
4. 확인 버튼을 클릭한다.
5. 선택된 노드들이 성공적으로 부팅되면 **Cluster View**에 부팅된 노드의 아이콘 상태가 변경된다.



그림 16 노드 부트

참고 : 노드를 부팅하기 위해서는 JEUS 매니저가 실행되고 있어야 한다. 실행에 앞서 각 노드에서 "jeus" 명령을 실행하였는지 확인해 봄야 한다.

5.2.2 노드 다운

JEUS 매니저 > 제어 > 다운

1. 노드 트리에서 최상위에 위치한 **JEUS 매니저**를 바로 클릭한 후 주 화면에 나타난 탭에서 **제어** 탭을 선택하거나 컨텍스트 메뉴에서 **JEUS 노드 제어** 메뉴를 클릭한다.
2. 안쪽의 탭 패널에서 **다운** 탭을 선택한다.
3. 노드 탐색기에서 다운하고자 하는 노드들을 선택한다.
4. **확인** 버튼을 클릭한다.
5. 노드들이 성공적으로 다운되면 **Cluster View**에 다운된 노드의 아이콘이 변경된다.



그림 17 노드 다운

참고 : JEUS 웹 관리자가 실행되고 있는 노드를 다운할 경우 서버와의 연결이 종료되어 브라우저에 오류 메시지가 출력되므로 다운하기 전에 웹 관리자가 실행중인 노드가 포함되어있는지 여부를 확인해 보아야 한다.

5.2.3 노드 종료

JEUS 매니저 > 제어 > 종료

1. 노드 트리에서 최상위에 위치한 **JEUS 매니저**를 바로 클릭한 후 주 화면에 나타난 탭에서 **제어** 탭을 선택하거나 컨텍스트 메뉴에서 **JEUS 노드 제어** 메뉴를 클릭한다.

2. 안쪽의 템 패널에서 **종료**탭을 선택한다.
3. 노드 탐색기에서 다운하고자 하는 노드들을 선택한다.
4. 확인 버튼을 클릭한다.
5. 노드들이 성공적으로 종료되면 **Cluster View**에 종료된 노드의 아이콘이 변경되며 노드의 정보를 볼 수 있는 링크가 사라진다.



그림 18 노드 종료

참고 : JEUS 웹 관리자가 실행되고 있는 노드를 종료할 경우 서버와의 연결이 종료되어 브라우저에 오류 메시지가 출력되므로 종료하기 전에 웹 관리자가 실행중인 노드가 포함되어 있는지 여부를 확인해 보아야 한다.

5.3 JEUSMain.xml 분배

JEUSMain.xml 분배는 현재 선택된 노드의 JEUSMain.xml 을 전체 노드에 동일하게 복사하는 것을 말한다. 이 기능은 노드들을 부팅하기 전에 분배를 통해서 동일한 설정을 적용 할 경우에만 사용하는 것으로 노드들이 이미 부팅 되었을 경우에는 사용하지 않는 것이 좋다.

5.3.1 JEUSMain.xml 분배

JEUS 매니저 > 분배



그림 19 JEUSMain.xml 분배 선택

1. 노드 트리에서 최상위에 위치한 **JEUS 매니저**를 바로 클릭한 후 주 화면에서 **분배**탭을 선택하거나 컨텍스트 메뉴에서 **분배메뉴**를 클릭한다.
2. 노드 탐색기에서 JEUSMain.xml 분배하고자 하는 노드들을 선택한다.
3. 확인 버튼을 클릭한다.



그림 20 JEUSMain.xml 분배

6 엔진 컨테이너

엔진 컨테이너는 JEUS 노드의 핵심 구성요소이다. 이 엔진 컨테이너는 J2EE 어플리케이션들(EJB, 서블릿, JMS)을 실행할 책임을 가지는 JEUS 엔진들을 관리한다.

엔진 컨테이너는 그 내부에서 동작하고 있는 엔진들에게 중요한 서비스들을 제공한다. 이 서비스들에는 트랜잭션이나 네이밍과 보안등이 있다. 또한 엔진 컨테이너는 배치 대상의 기본 단위가 된다. 어플리케이션 모듈의 배치에 관한 내용은 14 배치를 참조하기 바란다.

6.1 엔진 컨테이너 구성

6.1.1 엔진 컨테이너 개요

JEUS 매니저 > 엔진 컨테이너 개요

현재 등록된 엔진 컨테이너의 목록과 엔진 컨테이너의 상태를 볼 수 있다. 또한 엔진컨테이너에 대한 명령을 내릴 수 있다.

6.1.2 새로운 엔진 컨테이너 생성

JEUS 매니저 > 엔진 컨테이너 개요 > 추가

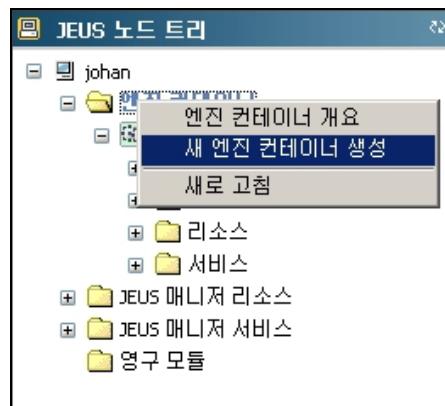


그림 21 엔진 컨테이너 생성 선택

1. 노드 트리에서 JEUS 매니저 바로 밑에 위치한 엔진 컨테이너 폴더를 클릭한 후 하단의 새 엔진 컨테이너 생성 링크를 클릭하거나 컨텍스트 메뉴에서 새 엔진 컨테이너 생성 메뉴를 클릭한다.
2. 엔진 컨테이너 이름 필드에 새로운 엔진 컨테이너 이름을 입력한다.
3. 생성 버튼을 클릭한다.

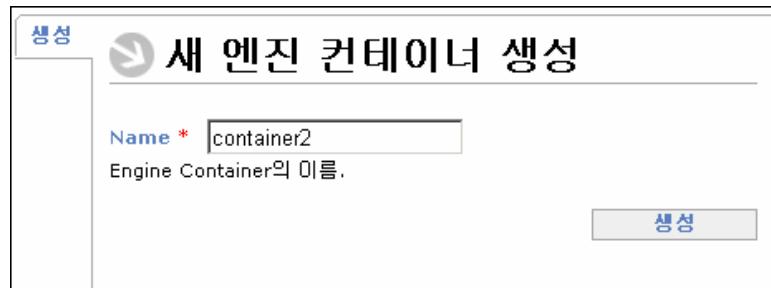


그림 22 엔진 컨테이너 생성

4. 엔진 컨테이너가 생성되면 노드 트리에 방금 새로 생성된 엔진 컨테이너가 비활성 상태로 나타나고 엔진 컨테이너 개요 페이지로 이동된다. 엔진 컨테이너 개요 페이지에서도 새로 생성된 엔진 컨테이너의 이름과 상태와 명령들이 나타날 것이다.



그림 23 엔진 컨테이너 생성 후 노드 트리

6.1.3 기존의 엔진 컨테이너 복사

JEUS 매니저 > 엔진 컨테이너 개요 > 복사

엔진 컨테이너 복사는 동일한 설정을 가지는 엔진 컨테이너를 생성하고자 할 때 사용되는 기능이다. 일반적으로 엔진 컨테이너의 모든 설정은 동일 할 수 있으나 일부 서비스들의 IP 포트들은 동일할 경우 문제가 발생할 수 있으므로 복사가 끝난 후 해당 값들을 중복되지 않게 변경하여야 한다. 예를 들어 서블릿 엔진의 웹 리스너의 포트라든가 JMS 엔진의 포트등이 대표적이라 할 수 있다.



그림 24 엔진 컨테이너 복사 선택

1. 노드 트리에서 **JEUS 매니저** 바로 밑에 위치한 **엔진 컨테이너** 폴더를 클릭한 후 복사하고자 하는 엔진 컨테이너에서 (□) 아이콘을 클릭하거나 노드 트리에서 복사하고자 하는 엔진 컨테이너를 선택한 후 컨텍스트 메뉴에서 **johan_container1 복사** 메뉴를 클릭한다.
2. 주화면에 나타난 양식에서 엔진 컨테이너 이름 필드에 새로운 **엔진 컨테이너** 이름을 입력한다.
3. 자동으로 생성된 엔진 이름이 아닌 다른 이름으로 엔진을 만들고자 한다면 엔진 이름 필드의 내용을 변경하여 입력한다.
4. 입력이 완료되면 **생성** 버튼을 클릭한다.

생성

엔진 컨테이너 복사

Name * Engine Container의 이름.

엔진

서블릿 엔진 이름 * Engine의 이름.

EJB 엔진 이름 * Engine의 이름.

웹 서비스 엔진 이름 * Engine의 이름.

JMS 엔진 이름 * Engine의 이름.

생성

그림 25 엔진 컨테이너 복사

- 새 엔진 컨테이너가 생성되면 노드 트리에 생성된 엔진 컨테이너가 비활성 상태로 나타나고 엔진 컨테이너 개요 페이지로 이동된다. 엔진 컨테이너 개요 페이지에서도 새로 생성된 엔진 컨테이너의 이름과 상태 명령들이 나타날 것이다.

6.1.4 엔진 컨테이너 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 삭제

현재 동작중에 엔진 컨테이너에 대한 삭제는 불가능하다. 엔진 컨테이너를 삭제하기 위해서는 우선 삭제하고자 하는 엔진 컨테이너를 다운 시켜야 한다. 엔진 컨테이너의 다운에 관한 내용은 6.3.2_엔진 컨테이너 다운을 참조하기 바란다. 또한 주의할 점은 하나의 노드에는 반드시 하나이상의 엔진 컨테이너를 가져야 하므로 삭제시 노드가 적어도 하나의 엔진 컨테이너를 가지는지 확인해 보아야 한다.

1. 노드 트리에서 JEUS 매니저 바로 밑에 위치한 엔진 컨테이너 폴더를 클릭한 후 주 화면에 나타난 엔진컨테이너 목록에서 삭제하고자 하는 엔진 컨테이너에서 (trash) 아이콘을 클릭한다.
2. 엔진 컨테이너가 삭제되면 노드 트리에 해당 엔진 컨테이너 아이콘(■)이 사라지고 엔진 컨테이너 개요 페이지로 이동된다. 엔진 컨테이너 개요 페이지에서도 삭제된 엔진 컨테이너의 정보가 사라지게 된다.

6.2 엔진 컨테이너 설정

6.2.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 기본 설정

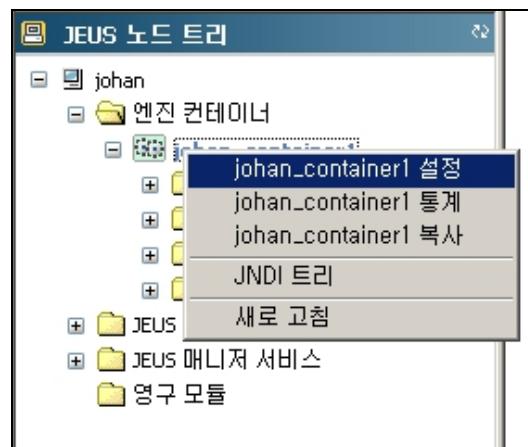


그림 26 엔진 컨테이너 설정 선택

1. 노드 트리에서 엔진 컨테이너 폴더 밑에 존재하는 엔진 컨테이너 중 하나를 선택하여 클릭한 후 주화면에 나타난 탭에서 설정탭을 선택한다.
2. 하위 탭 패널에서 기본 설정을 선택한다.
3. **Command Option**이나 추가 **Class Path**를 입력한다.

필드	설명
Command Option	설정파일에 존재하지 않는 추가적인 선택사항으로 java.lang.System의 속성에 등록되는 값이므로 -D로 시작하는 key=value 값을 입

	력해야한다.
Class Path	<i>JEUS_HOME/lib/system,</i> <i>JEUS_HOME/lib/application,</i> <i>JEUS_HOME/lib/datasource</i> 디렉토리에 존재하지 않는 다른 라이브러리 파일들을 추가하고 싶을 경우 이 곳에 해당 라이브러리의 경로를 추가한다. 형식은 일반적인 JDK Class Path 형식과 동일하다.

4. 입력이 완료되면 확인 버튼을 클릭한다.

6.2.2 에러 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 에러 로그

에러 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

6.2.3 사용자 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 사용자 로그

사용자 로그 관련 설정은 24.5 로그 서비스 장에서 설명할 것이다.

6.2.4 자동 배치 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 자동 배치

자동 배치는 지정된 디렉토리를 주기적으로 검사하면서 변경된 내용이 있을 경우 자동으로 배치를 시도하는 기능이다. 이 기능을 사용하면 어플리케이션을 수동적으로 배치가 없으므로 어플리케이션을 개발할 때나 잦은 업데이트가 필요한 어플리케이션일 경우 유용하게 사용할 수 있다.

1. 노드 트리에서 **엔진 컨테이너** 폴더 밑에 존재하는 **엔진 컨테이너** 중 설정하고자 하는 **엔진 컨테이너**를 클릭한 후 주 화면의 탭에서 **설정** 탭을 선택한다.
2. 하위 탭 패널에서 **자동 배치**를 선택한다.

3. **Path** 와 **Check Interval** 을 입력한다. 설정된 값은 다음 부팅시에 적용된다.

필드	설명
Path	자동 배치가 동작할 디렉토리의 폴 래스를 지정한다. 지정하지 않으면 JEUS_HOME/webhome/app_home 을 사용한다.
Check Interval	파일의 변경 여부를 확인할 주기를 밀리초 단위로 지정한다.

4. 입력이 완료되면 **확인** 버튼을 클릭한다.

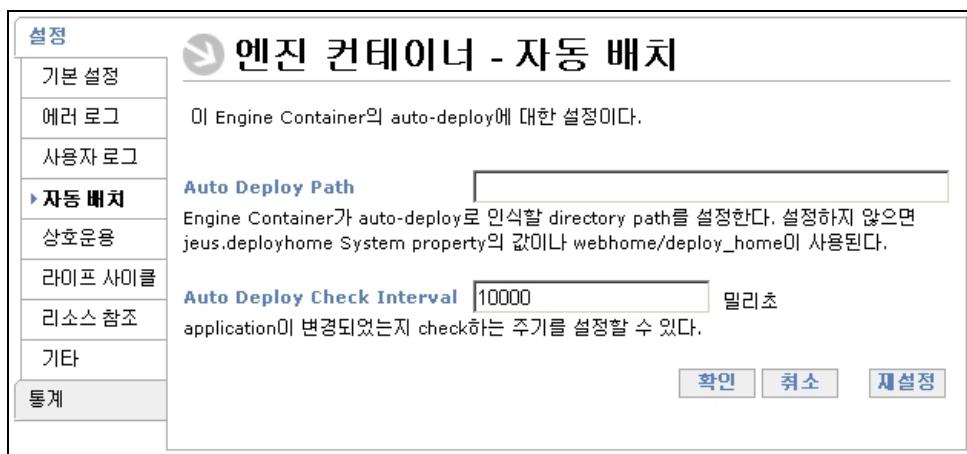


그림 27 엔진 컨테이너 자동 배치 설정

5. 설정을 초기화 하려면 **재설정** 버튼을 클릭한다.

6.2.5 상호 운용 설정

JEUS 마니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 상호운용

이 설정에서는 이 엔진 컨테이너가 IIOP 프로토콜을 사용하는 모든 ORB 와 상호동작 할 수 있도록 한다.

1. 노드 트리에서 **엔진 컨테이너** 폴더 밑에 존재하는 **엔진 컨테이너**중 하나를 클릭한 후 설정탭을 선택한다.
2. 하위 탭 패널에서 **상호운용**을 선택한다.

3. OTS 사용여부를 입력 한다.
4. 입력이 완료되면 확인 버튼을 클릭 한다.

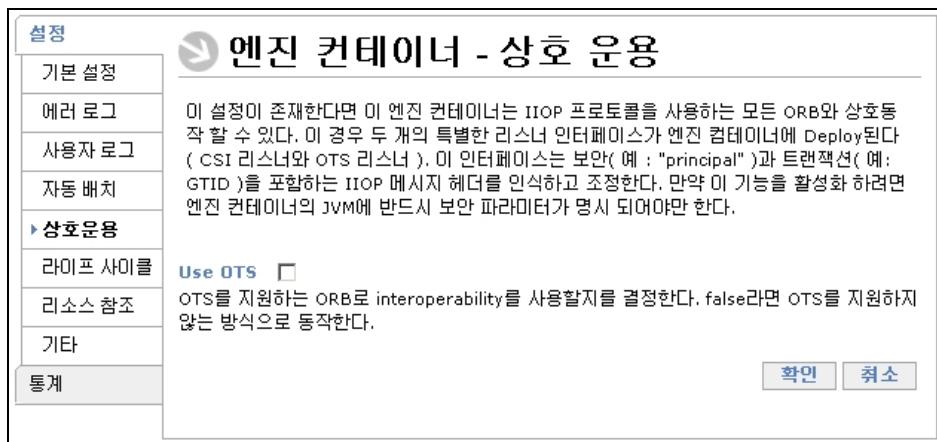


그림 28 엔진 컨테이너 상호 운용 설정

6.2.6 라이프 사이클 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 라이프 사이클

엔진 컨테이너의 각종 라이프 사이클 이벤트에 호출할 메소드를 지정 할 수 있다.

1. 노드 트리에서 엔진 컨테이너 폴더 밑에 존재하는 엔진 컨테이너 중 하나를 클릭한 후 설정탭을 선택한다.
2. 하위 탭 패널에서 **라이프 사이클**을 선택한다.
3. 클래스 이름과 메소드를 입력한다.
4. 입력이 완료되면 확인 버튼을 클릭한다.



그림 29 엔진 컨테이너 라이프 사이클 설정

6.2.7 리소스 참조 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 리소스 참조

엔진 컨테이너가 부트할 때 리소스를 새로운 이름으로 등록할 필요가 있는 경우 사용한다. 주로 서로 다른 데이터소스를 같은 이름으로 사용하고자 할 때 유용하다. 이 설정을 하면 동일한 배치기술자(Deployment Descriptor)를 이용하여 각 컨테이너마다 다른 리소스를 사용할 수 있다.

1. 노드 트리에서 **엔진 컨테이너** 폴더 밑에 존재하는 엔진 컨테이너 중 하나를 클릭한 후 설정탭을 선택한다.
2. 하위 탭 패널에서 **리소스 참조**를 선택한다.
3. 각 Jndi Info에 "reference name =export name"의 값을 입력한다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

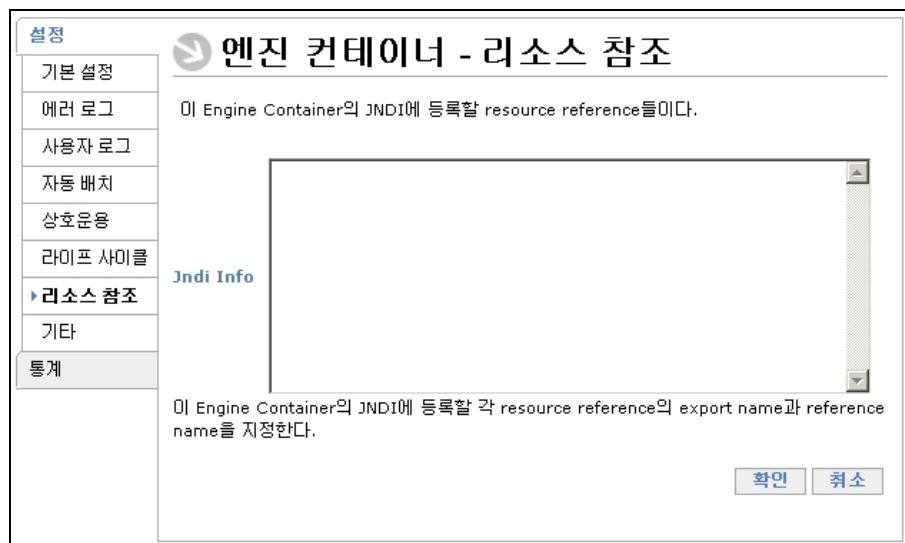


그림 30 엔진 컨테이너 리소스 참조 설정

6.2.8 기타 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 설정 > 기타

이 페이지에서는 기타 설정들을 입력한다. 각 항목에 대한 자세한 설명은 JEUS Server 안내서의 부록을 참조하라.

1. 노드 트리에서 **엔진 컨테이너** 폴더 밑에 존재하는 엔진 컨테이너 중 하나를 클릭한 후 설정탭을 선택한다.
2. 하위 탭 패널에서 **기타**를 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부팅시에 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

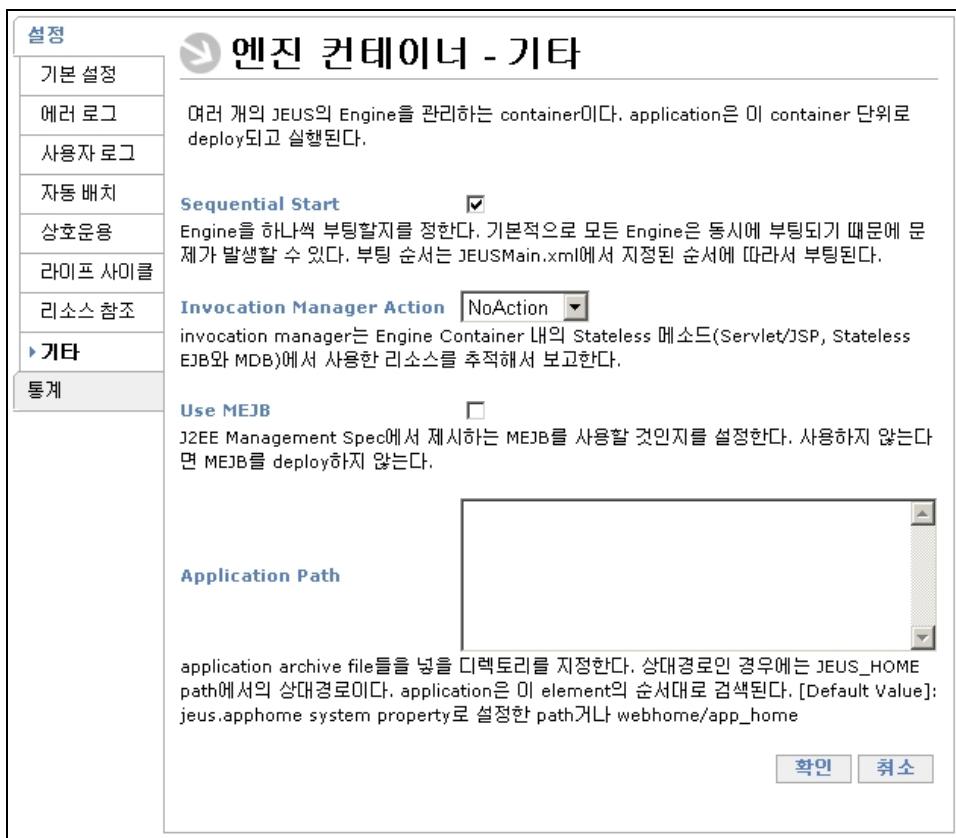


그림 31 엔진 컨테이너 기타 설정

6.3 엔진 컨테이너 제어

6.3.1 엔진 컨테이너 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 시작

1. 노드 트리에서 **JEUS 매니저** 바로 밑에 위치한 **엔진 컨테이너** 폴더를 클릭한 후 엔진 컨테이너 목록에서 시작하고자 하는 엔진 컨테이너의 시작 버튼을 클릭한다.
2. 잠시 후 엔진 컨테이너가 시작되고 시작이 완료되면 노드 트리에서 해당 엔진 컨테이너의 상태가 활성화된 아이콘(■)으로 변경된다.

이름	상태	명령	
johan_container1	Running	<input type="button" value="시작"/> <input type="button" value="다운"/>	
johan_container2	Stopped	<input type="button" value="시작"/> <input type="button" value="다운"/>	

그림 32 엔진 컨테이너 시작

6.3.2 엔진 컨테이너 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 다운

1. 노드 트리에서 **JEUS 매니저** 바로 밑에 위치한 **엔진 컨테이너** 폴더를 클릭한 후 엔진 컨테이너 목록에서 다운하고자 하는 엔진 컨테이너의 **다운** 버튼을 클릭한다.
2. 잠시 후 엔진 컨테이너가 다운되고 노드 트리에서 해당 엔진 컨테이너의 상태가 비활성아이콘()으로 변경된다.

이름	상태	명령	
johan_container1	Running	<input type="button" value="시작"/> <input type="button" value="다운"/>	
johan_container2	Running	<input type="button" value="시작"/> <input type="button" value="다운"/>	

그림 33 엔진 컨테이너다운

6.3.3 엔진 컨테이너 모두 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 모두 시작

1. 노드 트리에서 **JEUS 매니저** 바로 밑에 위치한 **엔진 컨테이너** 폴더를 클릭한 후 **모두 시작**버튼을 클릭한다.
2. 잠시 후 엔진 컨테이너가 모두 시작되고 **Cluster View** 와 **Node View** 가 생신된다.



그림 34 엔진 컨테이너 모두 시작

6.3.4 엔진 컨테이너 모두 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 모두 다운

1. 노드 트리에서 JEUS 매니저 바로 밑에 위치한 엔진 컨테이너 폴더를 클릭한 후 모두 다운버튼을 클릭한다.
2. 잠시 후 엔진 컨테이너가 모두 다운되고 Cluster View 와 Node View 가 갱신된다.



그림 35 엔진 컨테이너 모두 다운

6.4 엔진 컨테이너 통계

6.4.1 엔진 컨테이너 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 통계

default 엔진 컨테이너를 제외한 각 엔진 컨테이너는 각각이 하나의 JVM 위에서 동작한다. 따라서 엔진 컨테이너 통계정보의 주요 내용은 JVM에 관한

정보를 보는 것이다. 특히, 현재 사용되고 있는 메모리 Heap의 크기를 측정하는데 사용된다.

1. 노드 트리에서 엔진 컨테이너 폴더 밑에 존재하는 엔진 컨테이너중 통계를 보고자 하는 엔진 컨테이너를 클릭한 후 통계탭을 선택한다.



그림 36 엔진 컨테이너 모니터링

6.4.2 JNDI 트리

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JNDI 트리

이 메뉴는 엔진 컨테이너에 Local 바인딩된 객체들을 보여준다. 일반적으로 각종 리소스나 Local EJB 의 바인딩 여부를 확인할 때 사용한다



그림 37 JNDI 트리 보기 선택

1. 노드 트리에서 최상위에 위치한 엔진 컨테이너의 컨텍스트 메뉴에서 **JNDI 트리**메뉴를 클릭하거나, 주화면에서 엔진 컨테이너 설정의 오른 쪽 상위에 있는 **JNDI 트리** 링크를 클릭한다.
2. 엔진 컨테이너에 바인딩된 객체들의 트리가 나타난다.
3. 좌측의 트리에서 노드를 선택하여 클릭하면 바인딩 이름과 클래스 이름, 해시 코드가 출력된다.

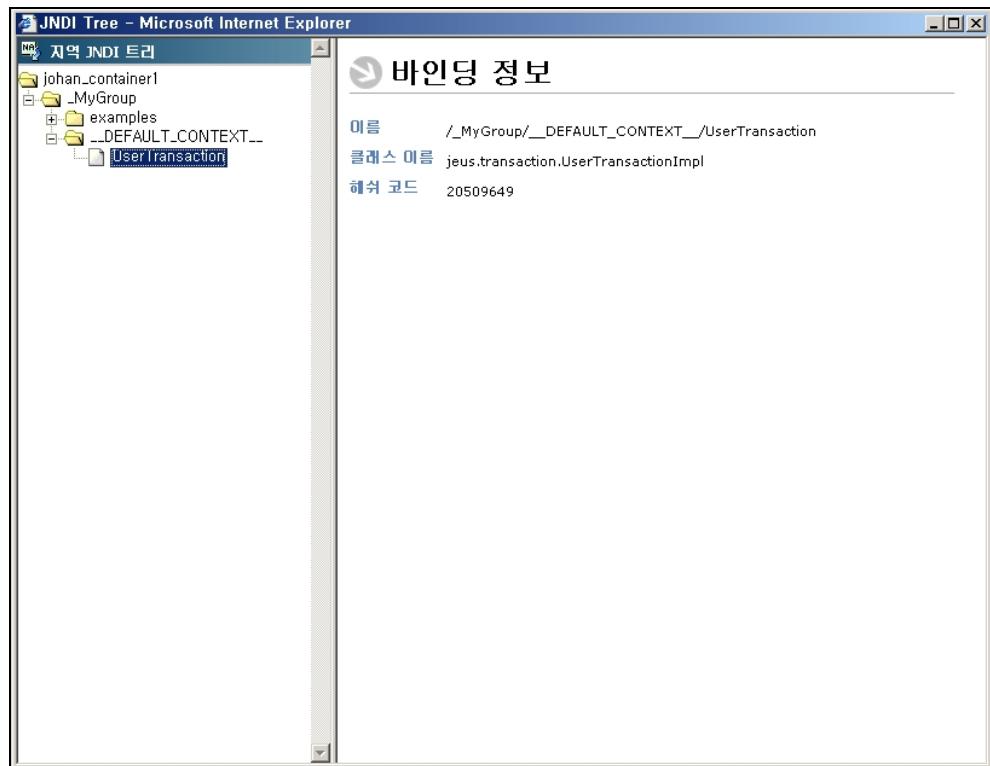


그림 38 Local JNDI 트리

7 엔진 개요

7.1.1 엔진 개요

JEUS 메뉴저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요

현재 엔진 컨테이너에 등록된 엔진들의 목록과 엔진들의 상태를 볼 수 있다.
또한 엔진들에 대한 명령을 내릴 수 있다.

8 EJB 엔진

EJB 엔진은 JEUS 의 엔진 컨테이너 내에서 설정되고 실행된다(JEUS Server 안내서 참조). 그리고, 각각의 EJB 엔진들은 각각의 JVM에서 실행된다. 즉, 한 엔진 컨테이너는 하나의 JVM을 뜻한다. 단 한 개의 EJB 엔진만이 각 엔진 컨테이너에 존재할 수 있기 때문에 각 EJB 엔진은 각 JVM 안에서 운영된다고 볼 수 있다.

이번 장에서는 EJB 엔진의 설정 및 제어와 모니터링에 대해서 알아본다.

8.1 EJB 엔진 구성

8.1.1 EJB 엔진 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 EJB 엔진 생성

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.



그림 39 엔진 폴더

2. 주 화면 하단의 새 EJB 엔진 생성을 클릭한다.



그림 40 EJB 엔진 생성 링크

- EJB 엔진 생성 페이지가 나타나면 엔진 이름 필드에 추가하고자 EJB 엔진의 이름을 입력하고 기존에 같은 이름으로 존재하는 설정파일을 그대로 사용하고 싶을 경우는 기존 설정 사용을 선택한다.



그림 41 EJB 엔진 생성 링크

- 입력이 완료되면 생성 버튼을 클릭한다.
- EJB 엔진이 생성되면 엔진 폴더 밑에 EJB 엔진 아이콘이 비활성 상태 ()로 추가되며 엔진 개요 페이지에 EJB 엔진이 추가되어 나타난다.

8.1.2 EJB 엔진 제거

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 EJB 엔진 제거

- 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.
- 주 화면에 나타난 엔진 목록에서 “ejb” 타입의 엔진에서 ()를 클릭 한다.

3. EJB 엔진이 제거되면 엔진 폴더 밑에 EJB 엔진 아이콘(エン진)이 삭제되고 엔진개요 페이지의 엔진 목록에서도 삭제된다.



그림 42 EJB 엔진 제거

8.2 EJB 엔진 설정

8.2.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 기본 설정

1. 노드 트리에서 엔진폴더 밑에 존재하는 EJB 엔진을 클릭하거나 컨텍스트 메뉴에서 EJB 엔진 설정을 선택한다.

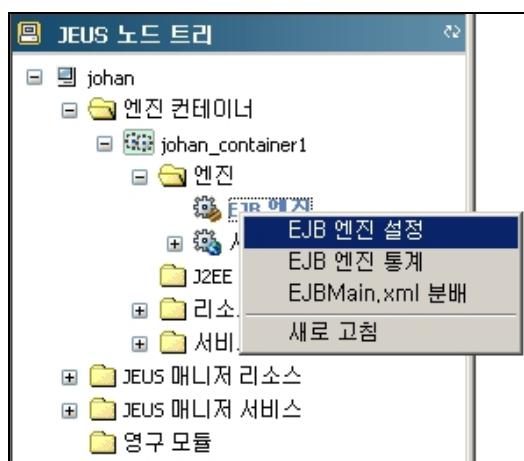


그림 43 EJB 엔진 설정 선택

2. 주 화면에 나타난 탭에서 기본설정탭을 선택한다.



그림 44 EJB 엔진 기본 설정

- 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

필드	설명
Resolution	EJB Passivation 과 Gabage Collection 이 시간을 주기로 시행된다.
Enable User Notification	EJB 엔진의 예외를 로그로 출력한다.

- 입력이 완료되면 확인 버튼을 클릭한다.

8.2.2 에러 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 에러 로그 설정

에러 로그 관련 설정은 로그 서비스 장에서 설명할 것이다.

8.2.3 활성 관리 (Active Management) 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 활성 관리

활성 관리는 EJB 모듈에 문제가 발생한 경우에 EJB 엔진이 전자우편을 통해 문제의 내용을 발송하는 기능을 일컫는다. 이런 상황이 발생할 수 있는 경우로는 EJB 클래스의 버그로 인하여 EJB 의 쓰레드 블럭을 오랫동안 지속하면서 시스템 리소스를 낭비하고 시스템 성능을 저하시키고 있는 경우를 들 수 있다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 **EJB** 엔진을 클릭하거나 컨텍스트 메뉴에서 **EJB** 엔진 설정을 선택한다.
2. 주 화면에 있는 탭에서 **활성 관리** 탭을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

필드	설명
Max Blocked Thread	EJB 엔진이 재시작 하기 전까지 허용할 수 있는 블록된 EJB Thread 의 최대 개수이다.
Max Idle Time	지정된 시간 동안 thread 가 블럭된 채 요청을 받지 않고 idle 상태에 있으면 “블록된 thread 리스트”로 추가된다. 이 설정은 엔진에서 블럭된 thread 를 판단하는 기준이 된다. 단위는 밀리초이다.

4. 입력이 완료되면 **확인** 버튼을 클릭한다.

8.2.4 HTTP 호출 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > HTTP 호출

이 설정은 EJB 모듈 중 특정 EJB 에 HTTP 호출을 설정하고 사용할 때 사용한다. 이 기능의 EJB 에서의 사용은 EJB 안내서를 참고하기 바란다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 **EJB** 엔진을 클릭하거나 컨텍스트 메뉴에서 **EJB** 엔진 설정을 선택한다.
2. 주 화면의 탭에서 **HTTP 호출** 탭을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

필드	설명
URL	반드시 HTTP-RMI 스텝으로부터 호출되는 RMI handler 서블릿의 URI (jeus.rmi.http.서블릿 Handler)를 입력한다. 이 URI에서는 프로토콜, IP, 포트를 제외한 서블릿 요청 경로만을 넣는다. 프로토콜은 “HTTP”로 IP는 RMI runtime과 같은 주소로 간주된다. 이 말은 HTTP-RMI 요청을 받은 웹 서버와 Web container가 RMI runtime과 같은 머신에 있어야 된다는 것이다. 그러면 RMI runtime의 주소는 RMI 스텝에게 알려지게 된다. 웹서버의 포트는 반드시 다음 “HTTP port”에 설정해야 한다.
HTTP port	HTTP-RMI 요청을 받고 처리할 웹 서버 또는 Web Container를 설정해야 한다. 이 웹서버/Web container는 반드시 RMI handler 서블릿이 배치되어 실행되고 있어야 한다.

4. 입력이 완료되면 확인 버튼을 클릭한다.

EJB 엔진 - HTTP 호출

이 기능을 설정하면 클라이언트 측의 EJB stub과 원격지의 RMI 실행환경은 HTTP-RMI 요청 (Request)으로 통신한다. 이것은 방화벽을 사이에 두고 EJB에 접근할 때 사용된다. 이 모드 (HTTP 호출 모드)를 사용할 때 클라이언트가 EJB stub에서 메소드를 호출하면 HTTP-RMI 요청 (Request)은 이것을 웹 컨테이너로 보낼 웹서버로 발송된다. 그리고 이것은 RMI Handler Servlet (jeus.rmi.http.ServletHandler)으로 보내지고 여기서 Handler Servlet은 요청 (Request)으로부터 HTTP 헤더를 제거한 뒤 이것을 RMI 실행환경으로 전송한다. 이 element가 설정되기 앞서 jeus.rmi.http.ServletHandler Servlet은 반드시 JEUS 웹 컨테이너에 Deploy되어 있어야만 한다(JEUS 웹 가이드를 참고한다.).

Url *	<input type="text"/>
HTTP-RMI stub에 의해 호출될 RMI Handler Servlet (jeus.rmi.http.ServletHandler)의 URI 경로가 반드시 설정되어야 한다. 이 URL은 프로토콜, 웹 서버 IP, 포트번호를 제외하고 오직 Servlet 요청 경로만을 설정해야 한다. 프로토콜은 HTTP, RMI 실행환경과 웹 서버는 같은 IP 주소를 가지고 있다고 가정한다(이것은 웹 서버와 웹 컨테이너는 반드시 HTTP-RMI 요청을 같은 머신에서 받는다는 것을 의미한다). 포트번호는 다음에 설명할 element(HTTP-port)에서 설정한다.	
Http Port * <input type="text" value="80"/> HTTP-RMI 요청을 받고 처리할 웹 서버의 포트번호를 설정한다. 이 웹 서버/웹 컨테이너에서는 반드시 RMI Handler Servlet이 Deploy되어 있고 이미 실행 중이어야만 한다.	
<input type="button" value="확인"/> <input type="button" value="취소"/> <input type="button" value="재설정"/>	

그림 45 EJB 엔진 HTTP Invocation 설정

5. 입력을 초기화 하기 위해서는 재설정 버튼을 클릭한다.

8.2.5 타이머 서비스 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 타이머 서비스 > 기본 설정

EJB 타이머 서비스는 EJB로 하여금 특정한 시간 또는 주기적으로 타이머 콜백을 받을 수 있도록 하는 서비스이다.

JEUS EJB 타이머 서비스는 기본적으로 스펙을 따르지만 지속적으로 타이머 서비스를 관리하는 기능은 성능과 사용자의 필요에 따라 선택적으로 사용할 수 있도록 구현되었다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 **EJB 엔진**을 클릭하거나 컨텍스트 메뉴에서 **EJB 엔진 설정**을 선택한다.
2. 주 화면의 탭에서 **타이머 서비스** 탭을 선택한다.
3. 왼쪽의 탭에서 **기본 설정** 탭을 선택한다.

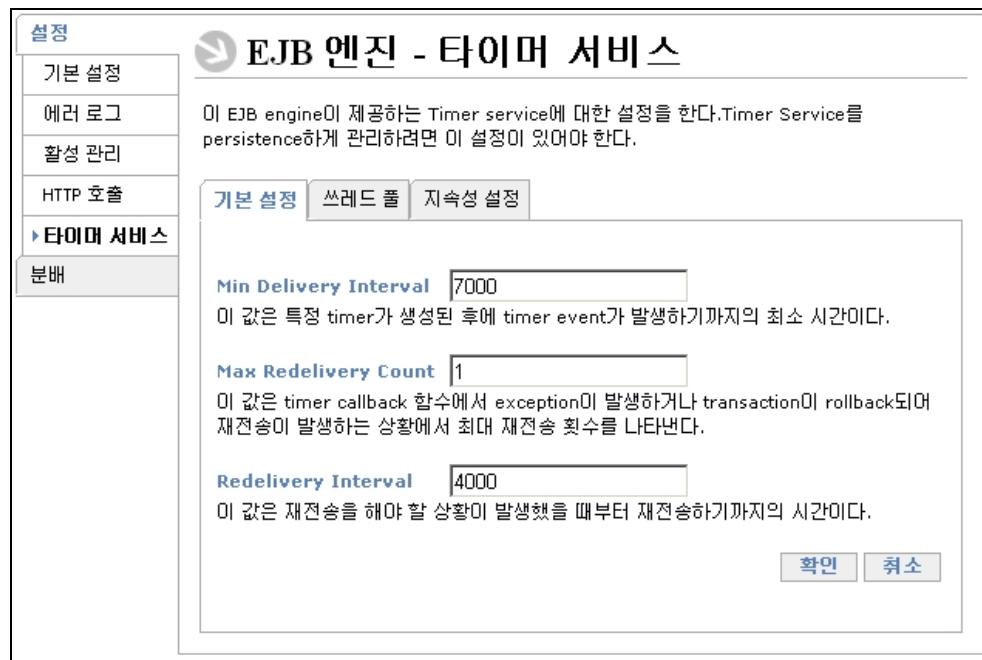


그림 46 EJB 엔진 타이머 서비스 기본 설정

4. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

필드	설명
Minimum Delivery Interval	이 값은 특정 timer 가 생성된 후에 timer event 가 발생하기까지의 최소 시간이다. 이 값이 작을수록 EJB 엔진에 부하를 줄 수 있다. 단위는 ms 이며 기본값은 7000 이다.
Max Redelivery Count	이 값은 timer callback 함수에서 exception 이 발생하거나 transaction 이 rollback 되어 재전송이 발생하는 상황에서 최대 재전송 횟수를 나타낸다. 기본값은 1 이다.
Redelivery Interval	이 값은 재전송을 해야 할 상황이 발생했을 때부터 재전송하기까지의 시간이다. 단위는 ms이고 기본값은 4000 이다.

5. 입력이 완료되면 확인 버튼을 클릭한다.

8.2.6 타이머 서비스 쓰레드 풀

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 타이머 서비스 > 쓰레드 풀

1. 노드 트리에서 엔진폴더 밑에 존재하는 **EJB 엔진**을 클릭하거나 컨텍스트 메뉴에서 **EJB 엔진 설정**을 선택한다.
2. 주 화면의 탭에서 **타이머 서비스** 탭을 선택한다.
3. 안쪽의 탭에서 **쓰레드 풀** 탭을 선택한다.
4. 쓰레드 풀의 쓰레드의 최소값과 최대값 그리고 쓰레드 풀을 검사할 주기를 입력한다. 최대값과 주기는 런타임에 적용되며 최소값의 설정은 런타임시 의미가 없음으로 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭한다.
6. 입력을 초기화 하기 위해서는 재설정 버튼을 클릭한다.

8.2.7 타이머 서비스 지속성 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 설정 > 타이머 서비스 > 지속성 설정

1. 노드 트리에서 엔진폴더 밑에 존재하는 **EJB** 엔진을 클릭하거나 컨텍스트 메뉴에서 **EJB** 엔진 설정을 선택한다.
2. 주 화면의 탭에서 **타이머 서비스** 탭을 선택한다.
3. 안쪽에 존재하는 탭에서 **지속성 설정** 탭을 선택한다.



그림 47 EJB 엔진 타이머 서비스의 지속성 설정

4. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

필드	설명
DB Vendor	이 값은 Timer CMP bean이 사용하는 DB의 vendor를 지정하는 것이다.

Data Source Name	이 값은 Timer CMP bean 이 사용하는 datasource resource 의 이름을 지정한다.
Engine Type	이 값은 Timer CMP bean 이 사용하는 엔진 type 을 지정한다. 자세한 것은 CMP 설정을 참고하기 바란다.
Table Name	이 값은 Timer CMP bean 이 사용할 DB table 이름을 지정한다. 기본값은 Jeus_Timer 이다.

5. 입력이 완료되면 확인 버튼을 클릭한다.
6. 입력을 초기화 하기 위해서는 재설정 버튼을 클릭한다.

8.3 EJB 엔진 제어

8.3.1 EJB 엔진 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 시작

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 엔진 목록에서 타입이 “ejb”인 엔진의 시작 버튼을 클릭한다.
3. 잠시 후 EJB 엔진이 시작되고 노드 트리의 EJB 엔진의 아이콘(.getNodeIcon())이 활성화 된다.



그림 48 EJB 엔진 시작

8.3.2 EJB 엔진 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 시작

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 엔진 목록에서 타입이 “ejb”인 엔진의 다운버튼을 클릭한다.
3. 잠시 후 EJB 엔진이 다운되고 노드 트리의 EJB 엔진의 아이콘(■)이 비활성화 된다.



그림 49 EJB 엔진 다운

8.4 EJB 엔진 통계

EJB 엔진은 자체의 모니터링은 없으며 배치된 모듈에 대한 전체적인 모니터링 기능만을 가지고 있다.

8.4.1 EJB 모듈 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 통계

1. 노드 트리에서 **EJB 엔진**을 클릭하거나 컨텍스트 메뉴에서 **EJB 엔진 통계**를 선택한다.

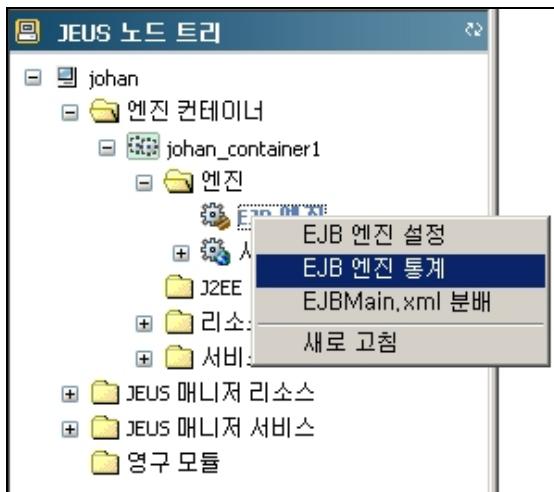


그림 50 EJB 엔진 통계

2. 현재 이 EJB 엔진에 배치된 모든 EJB 모듈의 통계 정보가 나타난다.

EJB 엔진 - 통계							
애플리케이션 이름	모듈 이름	빈 이름	생성	삭제	요청	커밋	롤백
null	simpleBean	simpleBean	0	0	0	0	0

표 편집 [새로 고침](#)

그림 51 EJB 엔진 통계 정보

8.5 EJBMain.xml 분배

EJBMain.xml 분배는 현재 선택된 노드의 *EJBMain.xml* 을 전체 EJB 엔진에 동일하게 복사하는 것을 말한다. 이 기능은 EJB 엔진을 시작하기 전에 분배를 통해서 동일한 설정을 적용 할 경우에만 사용하는것으로 EJB 엔진들이 이미 시작되었을 경우에는 사용하지 않는 것이 좋다.

8.5.1 EJBMain.xml 분배

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > EJB 엔진 > 분배

- 노드 트리에서 **EJB 엔진**을 클릭하여 **분배**탭을 선택하거나 컨텍스트 메뉴에서 **EJBMMain.xml** 분배를 선택한다.

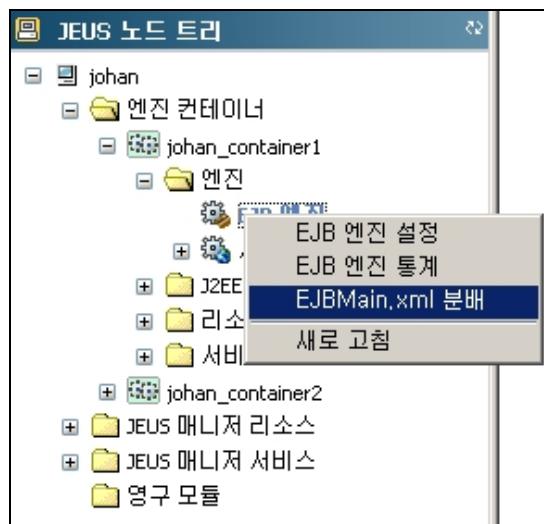


그림 52 EJBMMain.xml 분배 선택

- 엔진 탐색기에서 분배하고자 하는 EJB 엔진들을 선택한다.
- 확인버튼을 클릭한다.



그림 53 EJBMMain.xml 분배

9 서블릿 엔진

서블릿 엔진은 사용자 요청에 기반하여 동적인 웹 컨텐츠를 생성하는 엔진이다. 동적인 컨텐츠는 자바 서블릿이나 JSP로 생성된다. 엔진 컨테이너가 지원하는 트랜잭션이나 네이밍, 보안 서비스를 이용하여 이러한 자바 컴포넌트들을 실행하는 것이 바로 서블릿 엔진이다.

이 번장에서는 서블릿 엔진의 구성, 설정, 모니터링 및 분배에 대해서 알아볼 것이다.

9.1 서블릿 엔진 구성

9.1.1 서블릿 엔진 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 서블릿 엔진 추가

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.
2. 주 화면 하단의 새 서블릿 엔진 추가을 클릭한다.



그림 54 새 서블릿 엔진 추가

3. 서블릿 엔진 생성 페이지가 나타나면 엔진 이름 필드에 추가하고자 서블릿 엔진의 이름을 입력하고 기존에 같은 이름으로 존재하는 설정파일을 그대로 사용하고 싶을 경우는 기존 설정 사용을 선택한다.

4. 입력이 완료되면 생성 버튼을 클릭한다.



그림 55 서블릿 엔진 추가

5. 서블릿 엔진이 생성되면 엔진폴더 밑에 서블릿 엔진 아이콘(■)이 비활성 상태로 추가되며 엔진 개요 페이지에 서블릿 엔진이 추가되어 나타난다.

9.1.2 서블릿 엔진 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 제거

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.
2. 주 화면의 엔진 목록에서 “servlet” 타입의 엔진의 ■를 클릭한다.
3. 서블릿 엔진이 제거되면 엔진 폴더 밑에 서블릿 엔진 아이콘(■)이 삭제되고 엔진 개요페이지의 엔진 목록에서도 삭제된다.



그림 56 서블릿 엔진 삭제

9.2 서블릿 엔진 설정

9.2.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 기본 설정

- 노드 트리에서 엔진폴더 밑에 존재하는 서블릿 엔진을 클릭하거나 컨텍스트 메뉴에서 서블릿 엔진 설정을 선택한다.

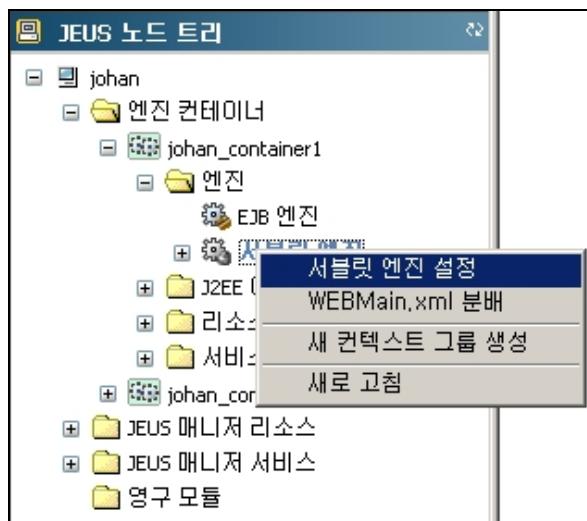


그림 57 서블릿 엔진 설정 선택

- 주 화면의 탭에서 기본 설정 탭을 선택한다.

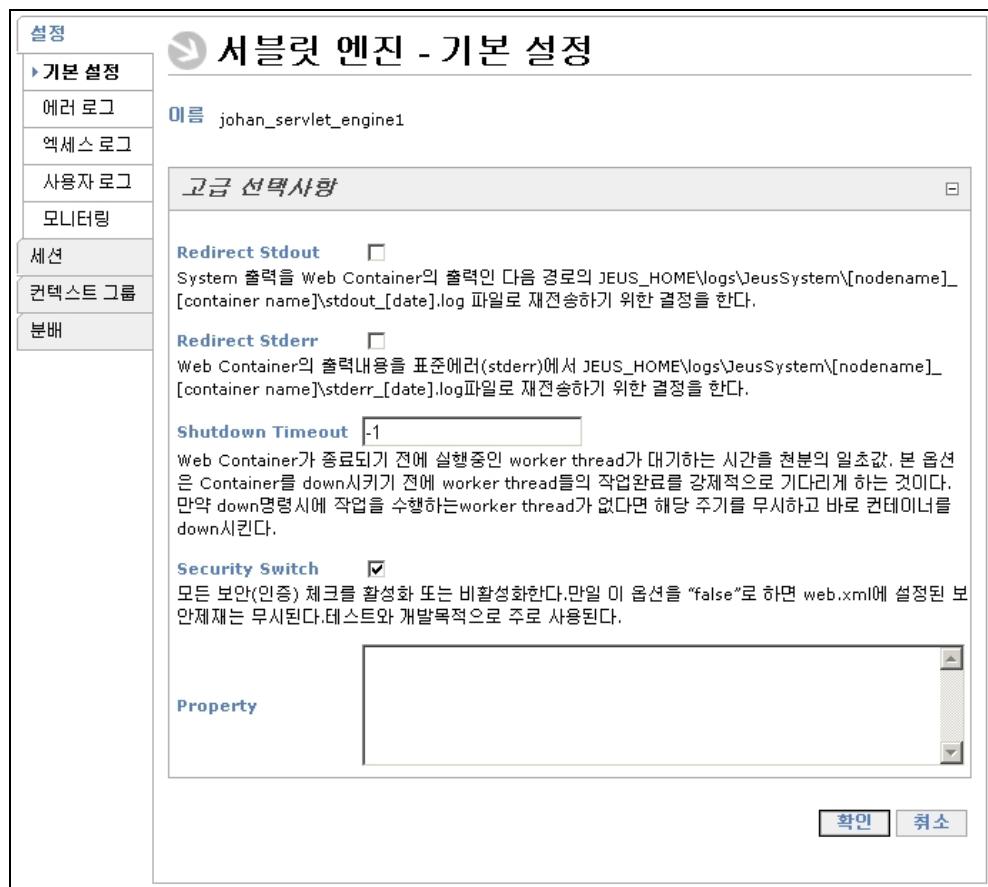


그림 58 서블릿 엔진 기본 설정

3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

9.2.2 에러 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 에러 로그

에러 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

9.2.3 액세스 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 액세스 로그

엑세스 로그의 일반적인 설정은 로그 서비스 장에서 설명할 것이다. 그러나 서블릿 엔진의 엑세스 로그의 Format 설정은 다른 로그에 없는 것이므로 이 절에서 설명할 것이다.

엑세스 로그의 Format 은 다음의 Pattern ID 들의 조합으로 이루어 진다. 각 Pattern ID 는 ‘%’로 시작되며 ‘%’로 시작되지 않는 문자열은 고정된 문자열로 로그에 바로 출력된다.

아래 표 2 는 엑세스 로그의 Pattern ID 와 설명을 보여준다.

표 2 Access Log Format 의 Pattern ID

Pattern ID	설명
%a	Remote IP address
%A	Local IP address
%b	Bytes sent, excluding HTTP headers, or '-' if zero
%B	Bytes sent, excluding HTTP headers
%h	Remote host name (or IP address if resolveHosts is false)
%H	Request protocol
%l	Remote logical username from identd (always returns '-')
%m	Request method (GET, POST, etc.)
%p	Local port on which this request was received, or '-' if none and it doesn't follow %U
%q	Query string (prepended with a '?' if it exists)
%r	First line of the request (method and request URI)

%s	HTTP status code of the response
%S	User Session ID
%t	Date and time, in Common Log Format
%u	Remote user that was authenticated (if any), else '-'
%U	Requested URL path
%v	Local server name
%D	Time taken to process the request, in millis
%T	Time taken to process the request, in seconds
%{xxx}i	for incoming headers
%{xxx}c	for a specific cookie
%{xxx}r	xxx is an attribute in the servlet request
%{xxx}s	xxx is an attribute in the http session
%{xxx}t	xxx is JDK standard DateFormat String which replaces default log date and time.

참고 : %{xxx}로 시작하는 Pattern ID 는 xxx 부분에 헤더나 세션 같은 객체의 속성을 지정할 때 사용된다. “xxx”는 속성의 키값을 나타낸다.

JEUS 의 기본적인 로그 Format 은 [%{yyyy.MM.dd HH:mm:ss}t] %a "%m %U%q" %s %D 이다.

9.2.4 사용자 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 > 엔진 > 설정 > 사용자 로그

사용자 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

9.2.5 모니터링 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 모니터링

모니터링은 서브릿 엔진의 자동 관찰 시스템이다. 여기에는 서블릿 엔진의 리소스를 체크하고 문제 발생시에 적절한 대응을 하기 위한 네 가지의 모니터링 작업이 존재한다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 서블릿 엔진을 클릭하거나 컨텍스트 메뉴에서 서블릿 엔진 설정을 선택한다.
2. 주 화면의 탭에서 모니터링 탭을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 바로 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

9.2.6 세션 클러스터 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 세션

세션 설정은 26 장 세션 추적에서 자세히 설명할 것이다.

9.3 서블릿 엔진 제어

9.3.1 서블릿 엔진 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 시작

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 주 화면의 엔진 목록에서 타입이 “servlet”인 엔진의 시작 버튼을 클릭한다.

3. 잠시 후 서블릿 엔진이 시작되고 노드 트리의 서블릿 엔진 아이콘(■)이 활성화 된다.



그림 59 서블릿 엔진 시작

9.3.2 서블릿 엔진 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 시작

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 주 화면의 엔진 목록에서 타입이 “servlet”인 엔진의 **다운** 버튼을 클릭한다.
3. 잠시 후 서블릿 엔진이 다운되고 노드 트리의 서블릿 엔진 아이콘(■)이 비활성화 된다.



그림 60 서블릿 엔진 다운

9.4 WEBMain.xml 분배

WEBMain.xml 분배는 현재 선택된 노드의 *WEBMain.xml* 을 전체 서블릿 엔진에 동일하게 복사하는 것을 말한다. 이 기능은 서블릿 엔진을 시작하기 전에 분배를 통해서 동일한 설정을 적용 할 경우에만 사용하는것으로 서블릿 엔진들이 이미 시작되었을 경우에는 사용하지 않는 것이 좋다.

서블릿 엔진의 경우 웹 서버 리스너가 포함되어 있으므로 분배후에는 동일한 노드에 존재하는 서블릿 엔진의 경우 웹 서버 리스너의 포트번호가 중복 되지 않는지 검사해야한다.

9.4.1 WEBMain.xml 분배

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 시작

1. 노드 트리에서 서블릿 엔진을 클릭하여 분배탭을 선택하거나 컨텍스트 메뉴에서 **WEBMain.xml** 분배 메뉴를 선택한다.

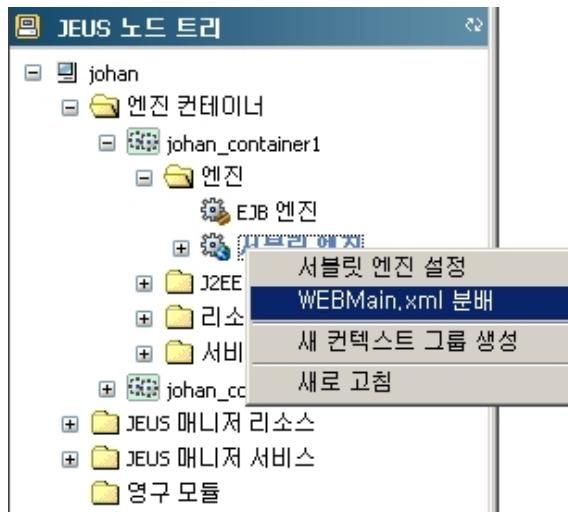


그림 61 WEBMain.xml 분배 선택

2. 엔진 탐색기에서 분배하고자 하는 서블릿 엔진을 선택한다.
3. 확인버튼을 클릭한다.



그림 62 WEBMain.xml 분배

10 컨텍스트 그룹

각 서블릿 엔진은 하나 이상의 컨텍스트 그룹을 포함할 수 있다. 컨텍스트 그룹은 여러 개의 컨텍스트와 가상 호스트를 그룹화 할 수 있는 JEUS에서만 제공하는 기능이다.

이 장에서는 컨텍스트 그룹의 구성과 설정, 제어 및 모니터링에 대해서 알아 볼 것이다.

10.1 컨텍스트 그룹 개요

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹

현재 엔진 서블릿 엔진에 등록된 컨텍스트 그룹들의 목록과 각각의 상태를 볼 수 있다. 또한 컨텍스트 그룹에 대한 명령을 내릴 수 있다.

10.2 컨텍스트 그룹 구성

10.2.1 컨텍스트 그룹추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 새 컨텍스트 그룹 추가

1. 노드 트리에서 서블릿 엔진을 클릭하여 컨텍스트 그룹을 선택한후 새 컨텍스트 그룹 생성 링크를 클릭하거나 서블릿 엔진의 컨텍스트 메뉴에서 새 컨텍스트 그룹 생성을 선택한다.

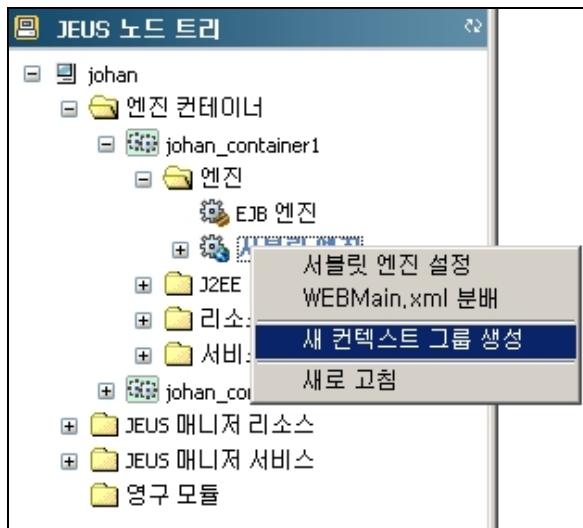


그림 63 컨텍스트 그룹 추가 선택

2. 컨텍스트 그룹 생성 페이지에서 컨텍스트 그룹 이름과 docbase를 비롯한 기타 항목을 입력한다. 컨텍스트 그룹을 생성시에 다음과 같은 사항을 주의한다.
 - A. 웹 리스너의 ID가 중복되지 않아야 한다.
 - B. 웹 리스너의 포트 번호가 중복되지 않아야 한다.
3. 입력이 완료되면 생성 버튼을 클릭한다.

This is a configuration form for creating a new context group. It has two tabs: '1. 기본 설정' (Basic Settings) and '2. 생성' (Create). The '1. 기본 설정' tab is active, showing the following fields:

- Group Name ***: A text input field with the placeholder 'Group Name *'.
- Group Docbase ***: A dropdown menu set to 'webapps'. Below it is a note: 'Context group의 기본 디렉토리를 정의한다. 이 디렉토리는 해당 Context group에 속한 Context 디렉토리의 모든 것을 포함한다.'
- 고급 선택사항**: A section containing a checkbox for 'Print Error To Browser' and a note: 'JSP나 Servlet을 수행하는 경우 에러발생시에 자세한 에러내용을 브라우저에 전송할지를 결정한다. 이 메시지는 개발 기간 동안에는 유용하지만 운영환경에서는 비활성화 해야 한다.'

At the bottom right of the form is a '다음 >' (Next >) button.

그림 64 컨텍스트 그룹 추가

4. 컨텍스트 그룹이 생성되면 서블릿 엔진 밑에 컨텍스트 그룹아이콘(■)이 비활성 상태로 추가되며 컨텍스트 그룹 페이지에 컨텍스트 그룹이 추가되어 나타난다.

10.2.2 컨텍스트 그룹 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 삭제

컨텍스트 그룹을 삭제하기 위해서는 컨텍스트 그룹이 비활성 상태이어야 한다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 주 화면의 탭에서 컨텍스트 그룹 탭을 선택한다.
2. 컨텍스트 그룹 목록에서 지우고자 하는 컨텍스트 그룹의 ■를 클릭한다.
3. 컨텍스트 그룹이 삭제되면 노드 트리에서 아이콘(■)이 삭제되며 컨텍스트 그룹 목록에서도 사라진다.



그림 65 컨텍스트 그룹 삭제

10.3 컨텍스트 그룹 설정

10.3.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 기본 설정

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹을 클릭한다.

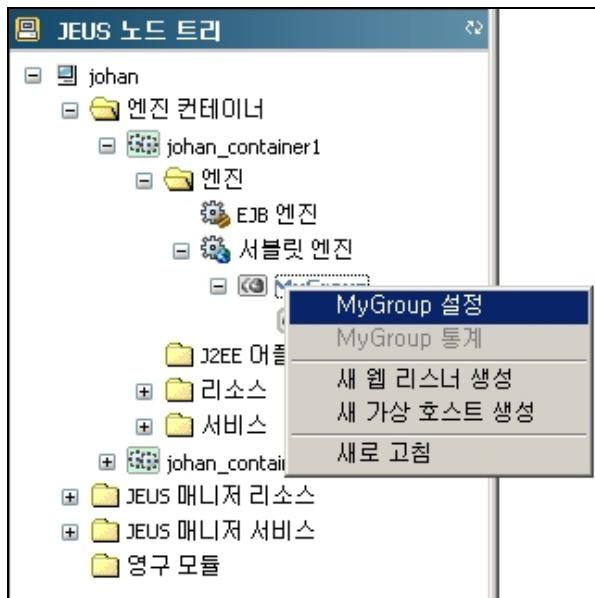


그림 66 컨텍스트 그룹 설정 선택

2. 주 화면의 탭에서 기본 설정을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

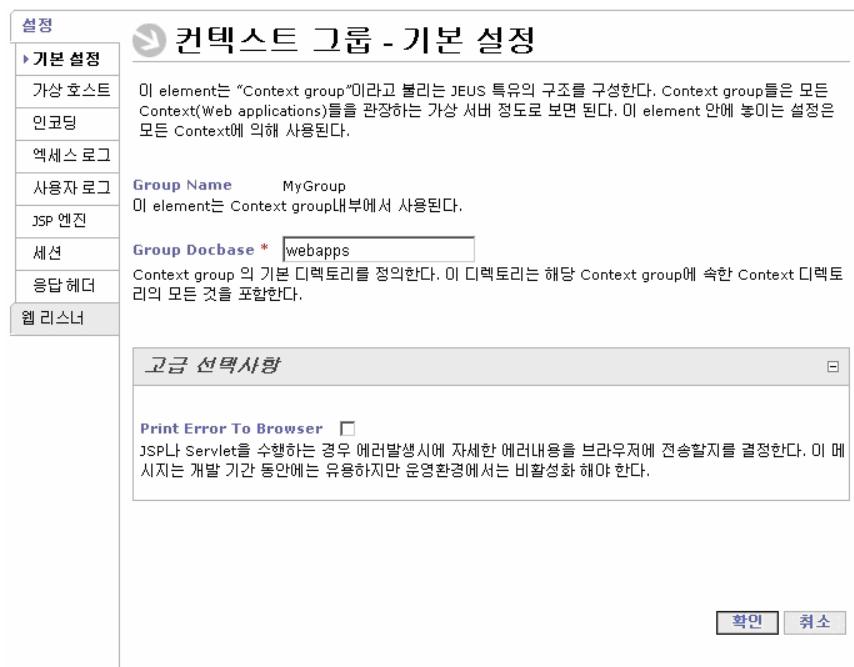


그림 67 컨텍스트 그룹 기본 설정

4. 입력이 완료되면 확인 버튼을 클릭한다.

10.3.2 가상 호스트 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 가상 호스트

가상 호스트는 웹 사이트가 다른 컨텍스트에게 주어진 컨텍스트 경로뿐만 아니라 요청 주소에 있는 도메인 이름에 의해서도 연결될 수 있도록 한다. 이렇게 함으로써 두 개 이상의 DNS 이름(예, “www1.foo.com” and “www2.foo.com”)을 하나의 서블릿 엔진에 설정하여 다른 컨텍스트를 서비스할 수 있다. 가상 호스트는 보기에 따라서 별도의 서블릿 엔진으로 보일 수 있다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.
2. 하위 탭에서 가상 호스트를 선택한다.
3. 하단의 새 가상 호스트 생성 링크를 클릭한다.
4. 각 항목들을 입력한다. 호스트 목록에는 줄별로 호스트의 이름을 등록하면 된다.
5. 입력을 완료하면 생성 버튼을 클릭한다.
6. 가상 호스트 페이지로 이동하면 목록에 추가된 가상 호스트가 나타난다.

참고 : 서블릿 엔진이 시작된 이후에 입력된 가상호스트는 다음 부트 시에 동작 한다.

10.3.3 가상 호스트 수정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 가상 호스트 수정

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.
2. 하위 탭에서 가상 호스트를 선택한다.

3. 수정하고자 하는 가상 호스트의 링크를 클릭한다.
4. 각 항목들을 수정한다. 변경된 항목은 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭한다.

10.3.4 가상 호스트 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 가상 호스트 삭제

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.
2. 하위 탭에서 가상 호스트를 선택한다.
3. 삭제하고자 하는 가상 호스트의 를 클릭한다.
4. 가상 호스트 페이지로 이동하여 목록에서 해당 가상호스트가 삭제된다.

참고 : 비록 가상 호스트가 목록에서 사라졌다고 하나 실제로는 서비스가되고 있으므로 실제적으로는 부트시에 삭제된다고 할 수 있다.

10.3.5 인코딩 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 인코딩

컨텍스트 그룹은 등록된 모든 컨텍스트에 의해 사용될 수 있는 인코딩 설정을 가지고 있다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.
2. 하위 탭에서 인코딩을 선택한다.
3. 각각의 태입에 인코딩을 입력한다. 변경된 내용은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

설정

- 기본 설정
- 가상 호스트
- 인코딩**
- 엑세스 로그
- 사용자 로그
- JSP 엔진
- 세션
- 응답 헤더
- 웹 리스너

컨텍스트 그룹 - 인코딩

요청 인코딩

Default [선택]

만일 사용자가 Servlet API를 통해 인코딩을 설정하지 않았고, <forced> 속성도 사용하지 않아서 HTTP 요청 헤더에 지정된 인코딩이 없으면 이 속성의 인코딩을 사용한다.

Forced [선택]

이 속성이 지정되면, HTTP데이터를 java.lang.String 객체로 변환할 때 항상 여기에 설정한 인코딩으로 사용된다. 따라서 가장 높은 우선순위를 가지고 있다.

응답 인코딩

Default [선택]

만일 사용자가 Servlet API를 통해 인코딩을 설정하지 않았고, <forced> 속성도 사용하지 않아서 HTTP 요청 헤더에 지정된 인코딩이 없으면 이 속성의 인코딩을 사용한다.

Forced [선택]

이 속성이 지정되면, HTTP데이터를 java.lang.String 객체로 변환할 때 항상 여기에 설정한 인코딩으로 사용된다. 따라서 가장 높은 우선순위를 가지고 있다.

POST 데이터 인코딩

Default [선택]

만일 사용자가 Servlet API를 통해 인코딩을 설정하지 않았고, <forced> 속성도 사용하지 않아서 HTTP 요청 헤더에 지정된 인코딩이 없으면 이 속성의 인코딩을 사용한다.

Forced [선택]

이 속성이 지정되면, HTTP데이터를 java.lang.String 객체로 변환할 때 항상 여기에 설정한 인코딩으로 사용된다. 따라서 가장 높은 우선순위를 가지고 있다.

확인 **취소**

그림 68 인코딩 설정

10.3.6 엑세스 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 엑세스 로그

이 설정은 컨텍스트 그룹별 엑세스 로그로서 설정 방법은 서블릿 엔진의 설정과 같다. 기본적인 설정 방법은 24.5 로그 서비스를 참조하고 포맷에 대한 정보는 9.2.3_엑세스 로그 설정을 참조하라

10.3.7 사용자 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 사용자 로그

유저 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

10.3.8 JSP 엔진 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > JSP 엔진

JSP 엔진은 JSP 컴파일러를 말한다. 이 설정에는 JSP를 어떻게 컴파일 할 것인지에 대한 선택 항목들이 있다.

1. 노드 트리에서 **컨텍스트 그룹**을 클릭하거나 서블릿 엔진의 **컨텍스트 그룹** 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 **컨텍스트 그룹** 링크를 클릭한다.
2. 하위 탭에서 **JSP 엔진**을 선택한다.
3. 각 항목들을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

설정

- 기본 설정
- 가상 호스트
- 인코딩
- 액세스 로그
- 사용자 로그
- JSP 엔진**
- 세션
- 응답 헤더
- 웹 리스너

컨텍스트 그룹 - JSP 엔진

Keep Generated

이 element는 JSP페이지로부터 생성된 자바 소스 파일을 컴파일하여 Servlet클래스 파일을 생성한 이후에 자바 소스 파일의 저장 여부를 결정한다. 디버깅 목적으로 이 파일들은 유용하다.

Java Compiler

JSP의 자바 소스를 Servlet 클래스 파일로 컴파일하기 위한 자바 컴파일러를 지정한다. 디폴트설정이 가장 효율적이기 때문에 이 element를 사용하지 않는 것을 권장한다.

Jsp Work Dir

JSP workdir은 Servlet 소스 파일과 클래스파일들이 저장되는 위치다. 개발자 또는 관리자는 디버깅목적으로 해당 디렉토리를 사용할 수 있다. 본 element는 일반적으로 사용할 필요가 없다.

고급 선택사항

Compile Output Dir

이 속성을 설정하여 JSP 파일에 의해 생성된 클래스 파일들을 <jsp-work-dir> 디렉토리가 아닌 다른 디렉토리에 둘 수 있다. 이 속성을 설정하지 않으면 클래스 파일들은 jspwork 디렉토리에 위치한다. 본 element는 보통 사용되지 않는다.

Compile Option

Servlet 컴파일러로 사용되는 옵션으로, 일반적으로는 사용되지 않는다.

Compile Encoding

본 속성은 JSP파일들이 파싱되어 생성된 Servlet 소스 파일을 compile할 때 ?encoding 옵션에 지정하는 값이다. 본 설정은 좀처럼 사용되지 않는다. Web Container는 적합한 설정을 자동으로 선택한다.

Check Included Jspfile

이 속성이 “true”로 설정되면, “include” directive 방식으로 include된 JSP파일의 변경 여부를 include한 JSP파일이 실행될 때 체크하여 변경된 경우 해당 JSP파일을 recompile 하도록 하는 기능이다. 기본적으로는 include된 JSP파일들은 변경 여부를 점검되지 않는다.

확인 **취소**

그림 69 JSP 엔진 설정

10.3.9 세션 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 세션

세션 설정은 컨텍스트 그룹과 서블릿 엔진이 HTTP 세션 객체를 내부적으로 어떻게 처리할 것인지에 대한 정책을 설정하는 것이다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.

2. 하위 탭에서 세션 탭을 선택한다.
3. 각 항목들을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭 한다.

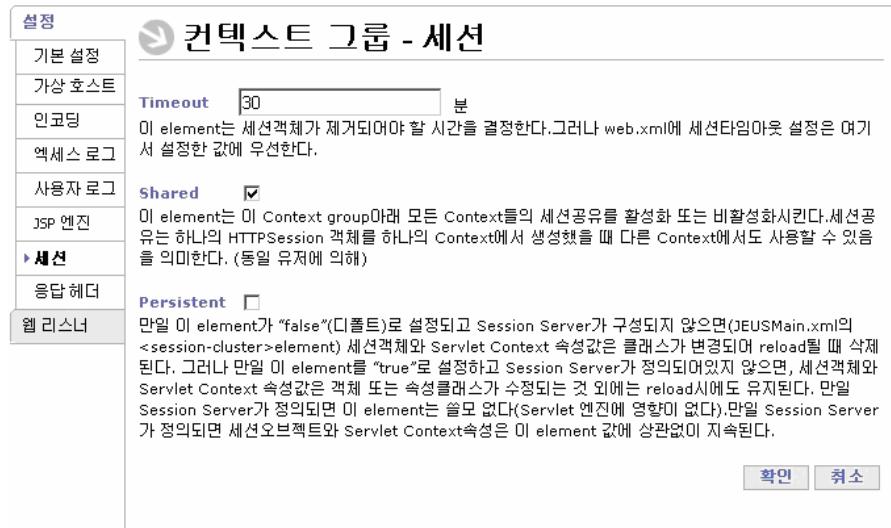


그림 70 세션 설정

10.3.10 응답 헤더 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 설정 > 응답 헤더

응답 헤더 설정은 세션 ID Cookie HTTP header에 전달될 값을 지정한다. 이 설정들은 컨텍스트 그룹의 모든 세션 ID Cookie에 적용된다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하거나 서블릿 엔진의 컨텍스트 그룹 탭의 컨텍스트 그룹 목록에서 설정하고자 하는 컨텍스트 그룹 링크를 클릭한다.
2. 하위 탭에서 응답 헤더 탭을 선택한다.
3. 각 항목들을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭 한다.

설정

- 기본 설정
- 가상 호스트
- 인코딩
- 액세스 로그
- 사용자 로그
- JSP 엔진
- 세션
- ▶ 응답 헤더**
- 웹 리스너

컨텍스트 그룹 - 응답 헤더

■ 세션 아이디 쿠키

Version 쿠키 ID 버전을 설정한다.

Domain Session id 쿠키를 보내는 서버의 도메인 이름을 설정한다. cookie는 이 도메인 요청에 대해서만 되돌아온다.

Path 하나의 session id 쿠키가 보내질 도메인 내에 URL 경로를 설정한다. 쿠키는 도메인이 적합할 때까지 해당 URL에 어떤 요청과 더불어 보내진다(<domain>element을 참조). 예를 들어 만일 "/examples"란 경로가 설정되고, 도메인은 ".foo.com"으로 설정되고 클라이언트 요청들은 "www.foo.com/examples" 이면 클라이언트의 쿠키는 요청과 더불어 보내진다.

Max Age Session id 쿠키의 "expires" 속성을 설정한다. 이 시간주기가 되면 쿠키는 클라이언트로부터 제거되고 더 이상 보내지 않는다.

Secure Session id 쿠키의 "secure" 속성을 설정한다. 만일 본 element가 "true"로 설정되면 session id 쿠키는 오직 secure HTTPS 커넥션으로 보내진다.

■ 사용자 헤더 필드

Header Field 본 Context group의 모든 HTTP 응답들에 포함하는 필드 이름과 값.

확인 **취소**

그림 71 응답 헤더 설정

10.4 컨텍스트 그룹 제어

10.4.1 정지(Suspend)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 제어 > 정지

임시적으로 컨텍스트 그룹과 그의 모든 컨텍스트들을 종료한다. 임시 종료된 컨텍스트 그룹은 클라이언트 요청을 서비스하지 못한다.

1. 노드 트리에서 서블릿 엔진을 클릭하여 컨텍스트 그룹 탭을 선택한다.
2. 컨텍스트 그룹 목록에서 정지하고자 하는 컨텍스트 그룹의 정지버튼을 클릭한다.
3. 요청이 실행되고 해제버튼이 활성화 된다.



그림 72 컨텍스트 그룹 제어

10.4.2 정지 해제(Resume)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 제어 > 정지 해제

임시적으로 종료된 컨텍스트 그룹을 다시 서비스 상태로 돌린다.

1. 노드 트리에서 서블릿 엔진을 클릭하여 컨텍스트 그룹 탭을 선택한다.
2. 컨텍스트 그룹 목록에서 서비스를 계속하고자 하는 컨텍스트 그룹의 해제버튼을 클릭한다.
3. 서비스가 다시 실행되고 정지버튼이 활성화 된다.

10.4.3 종료(Terminate)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 제어 > 종료

서블릿 엔진의 운영환경에서 컨텍스트 그룹과 그 컨텍스트를 제거한다. 모든 파일들과 설정들은 그대로 보존 된다.

1. 노드 트리에서 서블릿 엔진을 클릭하여 컨텍스트 그룹 탭을 선택한다.

2. 컨텍스트 그룹 목록에서 종료하고자 하는 컨텍스트 그룹의 **종료**버튼을 클릭한다.
3. 종료가 실행되고 재시작 버튼만 활성화 된다.

10.4.4 재시작 (Restart)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 제어 > 종료

컨텍스트 그룹을 재시작 한다. 즉, 모든 컨텍스트들을 재시작 한다.

1. 노드 트리에서 서블릿 엔진을 클릭하여 컨텍스트 그룹 탭을 선택한다.
2. 컨텍스트 그룹 목록에서 재시작하고자 하는 컨텍스트 그룹의 **재시작**버튼을 클릭한다.

참고 : 컨텍스트 그룹은 그 상태에 따라 제어 명령이 제한된다. 컨텍스트 그룹 내에 런타임에 배치된 웹 모듈이 하나라도 포함되어 있으면, 정지(suspend) 외에 다른 명령은 제한된다. 따라서 이 경우에는 해제(resume), 종료(terminate), 재시작(restart) 버튼이 비활성화되어 명령을 실행할 수 없게 된다.

10.5 컨텍스트 그룹 통계

10.5.1 웹모듈 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 통계 > 웹 모듈

이 기능에서는 현재 이 컨텍스트 그룹에 배치된 웹 모듈의 상태를 모니터링 한다.

1. 노드 트리에서 모니터링 하고자 하는 컨텍스트 그룹을 클릭하여 **통계**를 선택하거나 컨텍스트 메뉴에서 **MyGroup 통계**를 선택한다.

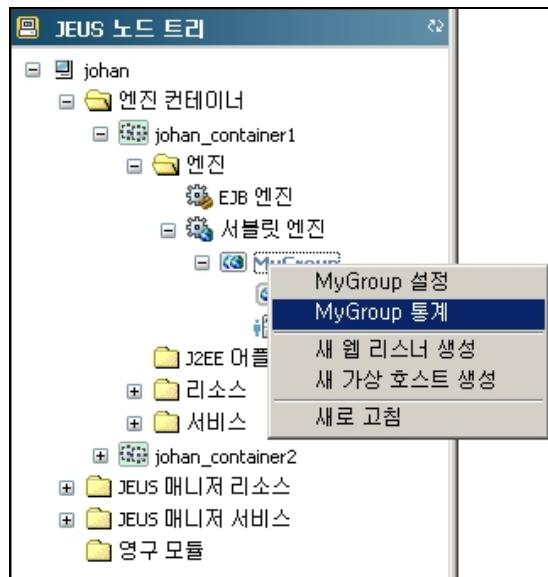


그림 73 컨텍스트 그룹 모니터링 선택

2. 주 화면의 탭에서 웹 모듈 탭을 선택한다.
3. 현재 이 컨텍스트 그룹에 배치된 모든 웹 모듈의 통계 정보가 나타난다.

The screenshot shows the '컨텍스트 그룹 - 웹 모듈 통계' (Context Group - Web Module Statistics) page. On the left, there is a sidebar with tabs: '설정' (Settings), '웹 리스너' (Web Listener), and '통계' (Statistics), with '통계' being the active tab. The main area displays a table with the following data:

웹 모듈 이름	요청	처리 시간	총 세션 수	활성중인 세션 수	비활성중인 세션 수	지역 세션 수	명령
examples	0	0	0	0	0	0	<button>재로딩</button> <button>정지</button> <button>해제</button> <button>종료</button> <button>재시작</button>

Below the table are buttons for '표 편집' (Table Edit) and '새로 고침' (Refresh).

그림 74 컨텍스트 그룹에 배치된 웹 모듈 모니터링

참고 : 웹 모듈 통계에서의 명령 또한 컨텍스트 그룹에서와 마찬 가지로 그 상태에 따라 제어 명령이 제한된다. 컨텍스트 그룹 내에 런타임에 배치된 웹 모듈이 하나라도 포함되어 있으면, 재로딩(reload)과 정지(suspend) 외에 다른 명령은 제한된다. 따라서 이 경우에는 해제(resume), 종료(terminate), 재시작(restart) 버튼이 비활성화되어 명령을 실행할 수 없게된다.

11 웹 서버 리스너

웹 서버 리스너는 일반적으로 웹 서버나 HTTP 클라이언트가 직접 접근 할 수 있는 서블릿 엔진쪽의 소켓이라고 생각할 수 있다. 이 리스너(소켓)는 웹 서버(또는 HTTP 클라이언트)로부터 요청을 받고 서블릿 엔진에서 처리한 정적 혹은 동적인 내용을 반환한다.

서블릿 엔진에서는 모두 8종류의 웹 서버 리스너를 제공한다.

1. **HTTP:** HTTP 리스너는 서블릿 엔진이 자체적으로 가지고 있는 최소 규모의 “웹서버”라고 할 수 있다. 이것은 정적인 내용과 JSP/서블릿에 대한 기본 HTTP 요청을 받고 처리할 수 있는 기능을 가진다.
2. **Secure:** Secure 리스너는 HTTP 리스너와 동일하지만 SSL(HTTPS)도 지원한다는 것이 다르다.
3. **TCP:** TCP 리스너는 일반적인 리스너에 맞춤 프로토콜(HTTP를 사용하지 않음)을 사용하는 리스너이다.
4. **UDP:** UDP 리스너는 TCP 리스너와 동일하나 UDP 프로토콜을 사용한다.
5. **Apache:** Apache 리스너는 앞단의 아파치 서버와 연동을 가능하게 한다.
6. **WebtoB:** WebtoB 는 JEUS 웹 어플리케이션 서버의 기본 웹 서버다.
7. **Tmax:** Tmax 리스너는 TP-Monitor 인 Tmax 와의 연동을 가능하게 한다. 기본적인 설정은 WebtoB 리스너와 유사하다.
8. **AJP13:** 이 리스너는 AJP 1.3 protocol 을 사용하는데, Apache 리스너가 AJP 1.2 protocol 을 사용하는 것을 제외하고는 Apache 리스너와 동일하게 동작한다.

이 장에서는 웹 서버 리스너의 구성과 설정 및 제어, 모니터링에 대해서 알아 볼 것이다.

11.1 웹 서버 리스너 구성

11.1.1 웹 서버 리스너 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 새 웹 서버 리스너 생성

- 노드 트리에서 컨텍스트 그룹을 클릭한 후 웹 서버 리스너 탭을 선택한 후 새 웹 서버 리스너 생성링크를 클릭하거나 컨텍스트 그룹의 컨텍스트 메뉴에서 새 웹 서버 리스너 생성을 택한다.



그림 75 새 웹 서버 리스너 생성 선택

- 리스너 종류: 추가하고자 하는 웹 서버 리스너의 종류를 선택한다.



그림 76 웹 서버 리스너 추가 - 1 단계 리스너 종류

3. 기본 설정: 기본적인 설정을 입력한다. 포트는 다른 리스너와 중복이 되지 않도록 입력해야 한다.

1. 리스너 종류
2. 기본 설정
3. 쓰레드 풀
4. 기타
5. 생성

웹 리스너 생성 - 기본 설정

Listener Id * lsnr1
해당 listener를 식별할 유일한 이름을 말한다. 이 이름은 구성파일의 다른 부분들로부터 해당 listener를 언급할 때 사용된다.

Port * 8089
클라이언트로부터 HTTP 요청을 받는데 사용되는 포트. IANA(www.iana.org)에 따르면 HTTP 포트는 80에 할당된다.

고급 선택사항

< 이전 다음 >

그림 77 웹 서버 리스너 추가- 2 단계 기본 설정

4. 쓰레드 풀: 이 설정은 각 리스너의 작업 쓰레드 풀의 크기와 행동 방식을 결정짓는다.

1. 리스너 종류
2. 기본 설정
3. 쓰레드 풀
4. 기타
5. 생성

웹 리스너 생성 - 쓰레드 풀

Min * 10
Pool에서 worker thread를 유지해야 하는 초기 개수이며 최소 개수를 의미한다.

Max 100
Pool에서 worker thread를 유지해야 하는 최대 개수를 의미한다.

Step 1
Pool 사이즈가 증가할 때마다 pool에 더해지는 worker thread의 개수를 나타낸다.

Max Idle Time 300000 밀리초
Pool에서 제거되기 전에 idle 상태로 머물러 있는 worker thread의 최대시간을 말한다.

Max Wait Queue 4
요청들을 담기 위해 새로운 worker thread를 생성하기 전에 이 element는 request wait queue 큐에 넣어질 요청의 최대 수를 설정한다.

쓰레드 상태 통보

Max Thread Active Time 10000
Thread가 block으로 간주되는 시간 값을 설정한다. Thread가 정해진 시간까지 실행되고 있으면 block된 것으로 간주한다.

Notify Threshold 10
경고통지가 e-mail을 이용하여 통지자에게 가기 전까지 존재하는 blocked thread의 최대값을 설정한다.

Restart Threshold 10
Web Container restart가 수행되기 전까지 존재하는 blocked thread의 최대값을 설정한다.

Notifier Id notifier_id
사용하는 email 통지자의 ID를 말한다.

Notify Subject notify_subject
Notify-threshold가 초과될 때 보내지는 e-mail의 제목을 설정한다.

Restart Subject restart_subject
Restart-threshold 가 초과될 때 보내질 e-mail의 제목을 설정한다.

< 이전 다음 >

그림 78 웹 서버 리스너 추가- 3 단계 쓰레드 풀

5. 기타: 기타 항목들에 대해서 입력을 한다. 이 설정 부분은 리스너 탑재에 따라 다소 차이가 난다. 각 탑재별 설정은 서블릿 안내서를 참고하기 바란다.

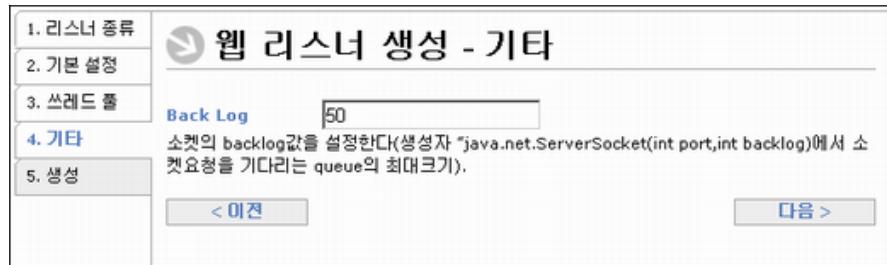


그림 79 웹 서버 리스너 추가 - 4 단계 기타 설정

6. 입력이 완료되면 생성 버튼을 클릭한다.
7. 웹 서버 리스너가 생성되면 노드 트리에서 컨텍스트 그룹 밑에 웹 서버 리스너 아이콘((listener icon))이 비활성 상태로 추가되며 웹 서버 리스너 페이지에 웹 서버 리스너가 추가되어 나타난다.



그림 80 웹 서버 리스너 lsnr2가 추가된 후 노드 트리

참고 : 비록 웹 서버 리스너가 추가되더라도 서블릿 엔진을 다시 시작하지 않으면 동작하지 않는다.

11.1.2 웹 서버 리스너 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 새 웹 서버 리스너 삭제

1. 노드 트리에서 컨텍스트 그룹을 클릭한 후 웹 서버 리스너 탭을 선택한다.
2. 웹 서버 리스너 목록에서 삭제하고자 하는 웹 서버 리스너의 를 클릭한다.
3. 웹 서버 리스너가 삭제되면 노드 트리에서 삭제되고 웹 서버 리스너 폐이지에서 사라진다.



그림 81 웹 서버 리스너 삭제

참고 : 비록 웹 서버 리스너가 삭제되어도 서블릿 엔진을 다시 시작해야 실제적으로 삭제가 된다.

11.1.3 WebToB 리스너 백업 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > WebToB 리스너 > 백업 추가

WebToB 백업은 주 서버가 심각한 장애 상태에 들어갔을 때 두 번째 WebtoB 서버에 연결하기 위한 연결 정보를 정의한다. 여기서 주의할 것은 백업 서버에 쓰레드 풀에 대한 설정이 없다면 주 WebToB 리스너의 설정에서 상속 받는다는 것이다.

WebToB 백업 추가하는 방법은 WebToB 리스너 추가하는 방법과 유사하다.

11.1.4 WebToB 리스너 백업 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > WebToB 리스너 > 삭제

1. 노드 트리에서 **WebToB 리스너**를 클릭 한 후 **백업탭**을 선택한다.
2. WebToB 백업목록에서 삭제하고자 하는 WebToB 백업의 를 클릭 한다.
3. WebToB 백업이 삭제되면서 목록에서 사라진다.

11.2 웹 서버 리스너 설정

11.2.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 설정 > 기본 설정

1. 노드 트리에서 변경하고자 하는 웹 서버 리스너를 클릭하거나 컨텍스트 그룹의 웹 서버 리스너 탭의 웹 서버 리스너 목록에서 웹 서버 리스너 링크를 클릭한다.
2. 하위 탭에서 **기본 설정**을 선택한다.

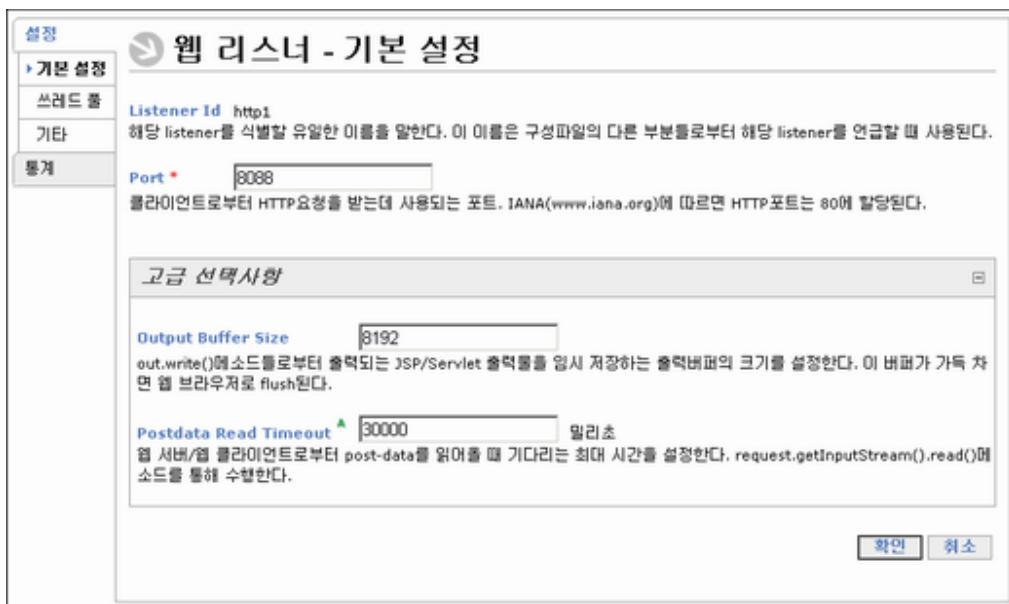


그림 82 웹 서버 리스너 기본 설정

3. 각 항목의 값을 입력한다. Postdata Read Timeout 값을 제외한 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

11.2.2 쓰레드 풀 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 설정 > 쓰레드 풀

1. 노드 트리에서 변경하고자 하는 웹 서버 리스너를 클릭하거나 컨텍스트 그룹의 웹 서버 리스너 탭의 웹 서버 리스너 목록에서 웹 서버 리스너 링크를 클릭한다.
2. 하위 탭에서 쓰레드 풀을 선택한다.
3. 각 항목의 값을 입력한다. Thread State Notification 을 제외한 모든 값은 런타임시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

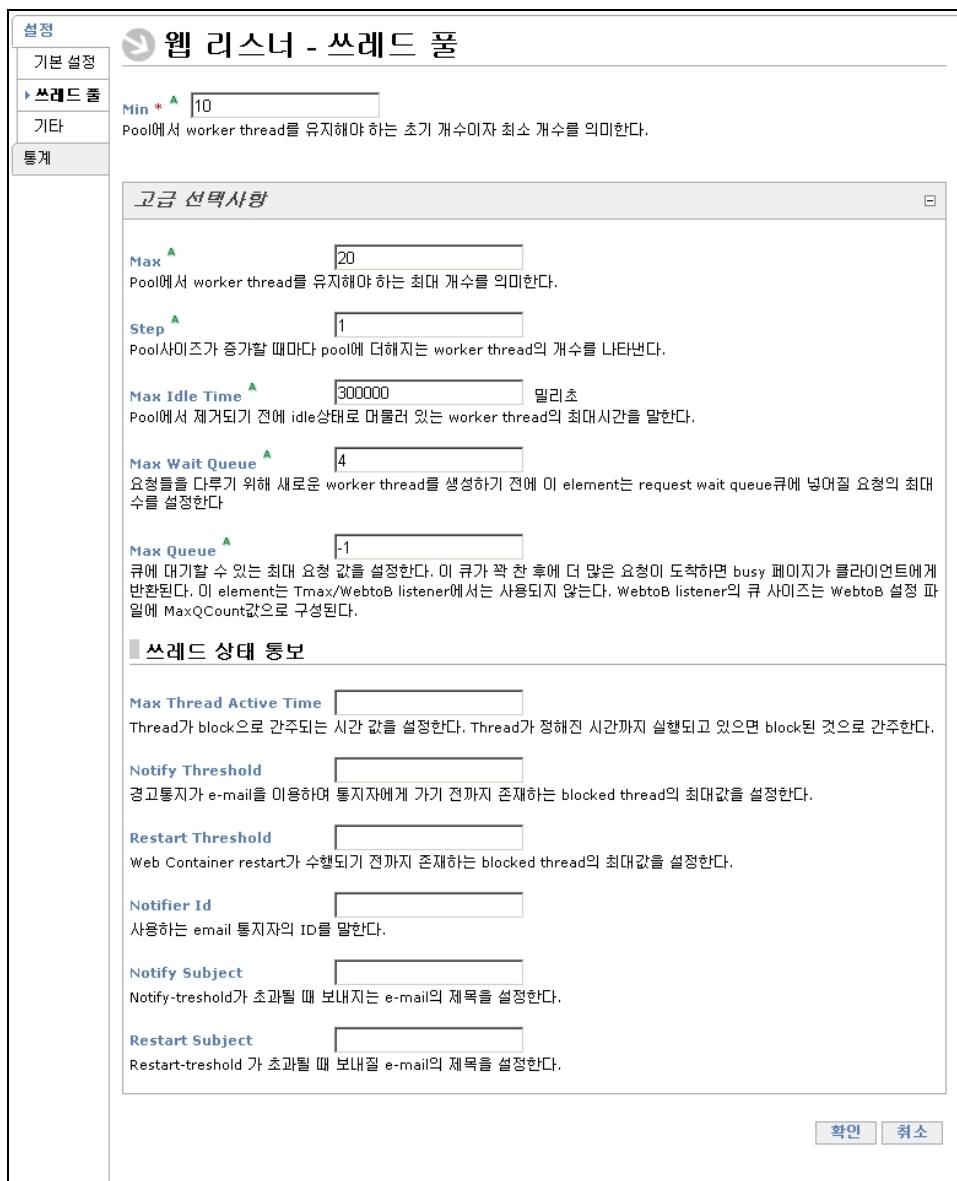


그림 83 웹 서버 리스너 쓰래드 풀 설정

11.2.3 기타 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 설정 > 기타

1. 노드 트리에서 변경하고자 하는 웹 서버 리스너를 클릭하거나 컨텍스트 그룹의 웹 서버 리스너 탭의 웹 서버 리스너 목록에서 웹 서버 리스너 링크를 클릭한다.
2. 하위 탭에서 기타 탭을 선택한다.
3. 각 항목의 값을 입력한다. 변경 내용은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

11.3 웹 서버 리스너 제어

일반적으로 웹 서버 리스너의 제어는 허용되지 않는다. 다만 WebToB 리스너의 경우에는 정지와 계속이 존재한다.

11.3.1 WebToB 리스너 정지(Suspend)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 제어 > 정지]

이 명령은 임시적으로 WebToB 리스너를 종료한다. 임시 종료된 WebToB 리스너는 클라이언트 요청을 서비스하지 못한다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하여 웹 서버 리스너 탭을 선택한다.
2. 웹 서버 리스너 목록에서 정지하고자 하는 WebToB 리스너의 정지버튼을 클릭한다.
3. 정지가 실행되고 계속버튼이 활성화 된다.



그림 84 WebToB 리스너 정지

11.3.2 WebToB 리스너 계속(Resume)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 > 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 제어 > 계속

임시적으로 종료된 WebToB 리스너를 다시 시작한다.

1. 노드 트리에서 컨텍스트 그룹을 클릭하여 웹 서버 리스너탭을 선택한다.
2. 웹 서버 리스너 목록에서 서비스를 계속하고자 하는 WebToB 리스너의 계속버튼을 클릭한다.
3. 서비스가 다시 시작되고 정지버튼이 활성화 된다.

11.4 웹 서버 리스너 통계

웹 서버 리스너에는 요청을 처리하는 작업 쓰레드(Worker Thread)들을 가지고 있다. 따라서 웹 서버 리스너에서는 이 쓰레드 풀의 현재 상태를 모니터링 한다. 일반적으로 쓰레드 풀은 각 웹 서버 리스너마다 하나씩 존재하며 WebToB 의 경우는 여러 개가 올 수 있다.

11.4.1 쓰레드 풀 전체 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 > 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 통계

- 노드 트리에서 웹 서버 리스너를 선택한 후 하위 탭에서 통계를 선택하거나 컨텍스트 메뉴에서 확인하려는 리스너 이름의 통계를 선택한다.

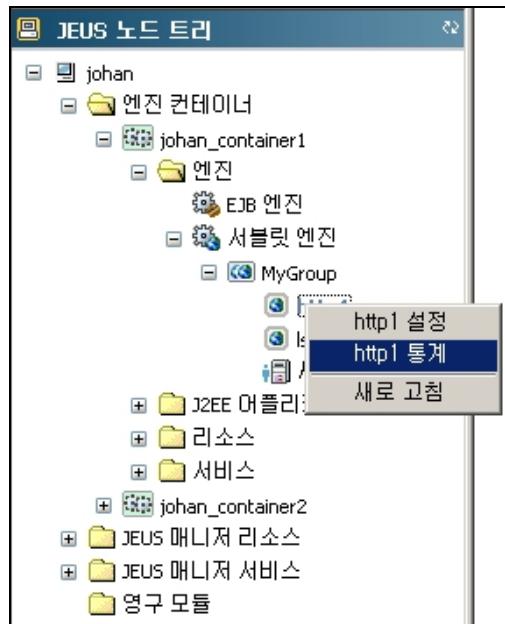


그림 85 웹 서버 리스너 통계 선택

- 모든 쓰레드 풀의 모니터링 정보가 출력된다.

The screenshot shows the 'Web Listener - Statistics' page. On the left, there are tabs for '설정' and '통계'. The '통계' tab is selected. In the center, there is a table titled '쓰레드 풀 이름' with one row for 'http1'. The table has columns: 쓰래드 풀 이름, 총 쓰래드, 활성중인 쓰래드, 블럭된 쓰래드, 최대 쓰래드, 대기중인 큐. The values are: http1, 10, 0, 0, 20, 0. Below the table is a link '표 편집' and a button '새로 고침'.

쓰래드 풀 이름	총 쓰래드	활성중인 쓰래드	블럭된 쓰래드	최대 쓰래드	대기중인 큐
http1	10	0	0	20	0

그림 86 웹 서버 리스너 모니터링

11.4.2 쓰래드 풀 개별 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 컨텍스트 그룹 개요 > 컨텍스트 그룹 > 웹 서버 리스너 > 통계 > 쓰레드 풀

- 노드 트리에서 웹 서버 리스너 탭을 선택한다.

2. 하위 탭에서 통계를 선택한다.
3. 구체적인 모니터링 정보를 볼 쓰레드 풀을 선택한다.
4. 선택된 쓰레드 풀의 모니터링 정보가 출력된다. 상태는 현재 상태를 나타내면 다음 4 가지가 있다.

표 3 웹 서버 리스너 쓰레드 풀의 쓰레드 상태 종류

상태	설명
Active	현재 요청을 처리하는 상태
Blocked	요청을 처리를 하던 중 블럭된 상태
Reconnection	웹 서버에 연결을 다시 맺는 중
Waiting	요청을 대기 하는 중

The screenshot shows the JEUS Web Manager interface with the '통계' (Statistics) tab selected. At the top, it shows summary statistics: 활성중인 쓰래드 0, 대기중인 쓰래드 10, 블럭된 쓰래드 0, and 재연결중인 쓰래드 0. Below this is a table titled '쓰래드 풀 - 통계' listing 10 threads (http1-w0 to http1-w9) with their status, elapsed time, total requests, average response time, and idle count. At the bottom right of the main content area are two buttons: '표 편집' (Edit Table) and '새로 고침' (Refresh).

쓰래드 아이디	상태	경과시간(밀리초)	총 요청 수	평균 처리 시간(밀리초)	미플라케이션
http1-w0	waiting	444532	0	0	
http1-w1	waiting	444532	0	0	
http1-w2	waiting	444532	0	0	
http1-w3	waiting	444532	0	0	
http1-w4	waiting	444532	0	0	
http1-w5	waiting	444532	0	0	
http1-w6	waiting	444532	0	0	
http1-w7	waiting	444516	0	0	
http1-w8	waiting	444516	0	0	
http1-w9	waiting	444516	0	0	

그림 87 웹 서버 리스너의 쓰래드 풀 모니터링

12 JMS 엔진

JMS 엔진은 Enterprise Messaging System에 접근하기 위해서 필요한 하위 컴포넌트들을 포함하고 있는 엔진이다. JMS 엔진은 또한 JEUS 내의 다른 엔진들과 함께 연동이 가능하다. 예를 들어 Message Driven Bean (JEUS WAS EJB 엔진에서 배치 됨)은 JEUS JMS 을 이용한다.

이번 장에서는 JMS 엔진의 구성과 설정 및 제어에 대해서 알아 볼 것이다. JMS의 커넥션 팩토리와 데스터네이션 및 클라이언트에 관한 것은 JMS 리소스 장에서 자세히 알아보고 이번장에서는 JMS 엔진에 관한 것에 대해서만 알아볼 것이다.

12.1 JMS 엔진 구성

12.1.1 JMS 엔진 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 JMS 엔진 추가

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.
2. 하단의 새 JMS 엔진 생성을 클릭한다.



그림 88 JMS 엔진 추가 선택

3. JMS 엔진 생성 페이지가 나타나면 엔진 이름 필드에 추가하고자 JMS 엔진의 이름을 입력하고 기준에 같은 이름으로 존재하는 설정파일을 그대로 사용하고 싶을 경우는 기준 설정 사용을 선택한다.
4. 입력이 완료되면 생성 버튼을 클릭한다.
5. JMS 엔진이 생성되면 엔진 폴더 밑에 JMS 엔진 아이콘(odb)이 비활성 상태로 추가되며 엔진 개요 페이지에 JMS 엔진이 추가되어 나타난다.

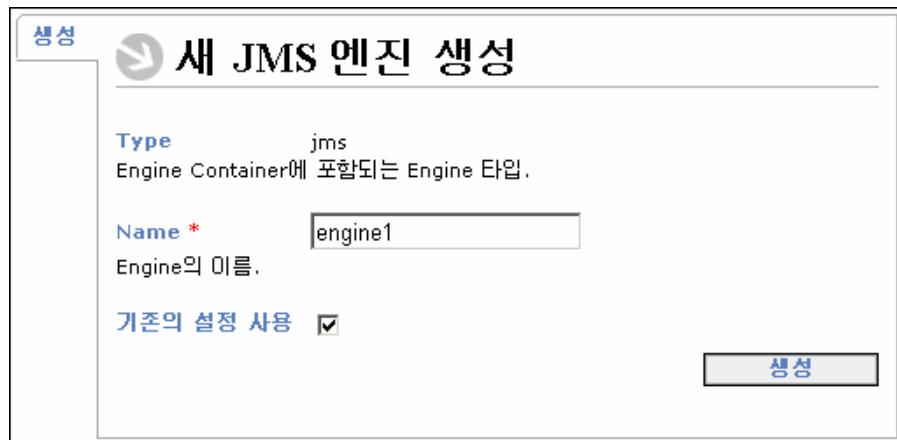


그림 89 JMS 엔진 추가

12.1.2 JMS 엔진 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 JMS 엔진 삭제

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진 폴더를 클릭한다.
2. 엔진 목록에서 “jms” 타입의 엔진의 편집 버튼을 클릭한다.
3. JMS 엔진이 삭제되면 엔진 폴더 밑에 JMS 엔진 아이콘이 삭제되고 엔진 페이지의 엔진 목록에서도 삭제된다.



그림 90 JMS 엔진 삭제

12.2 JMS 엔진 설정

12.2.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 기본 설정

- 노드 트리에서 엔진폴더 밑에 존재하는 JMS 엔진을 클릭하거나 컨텍스트 메뉴에서 JMS 엔진 설정을 선택한다.

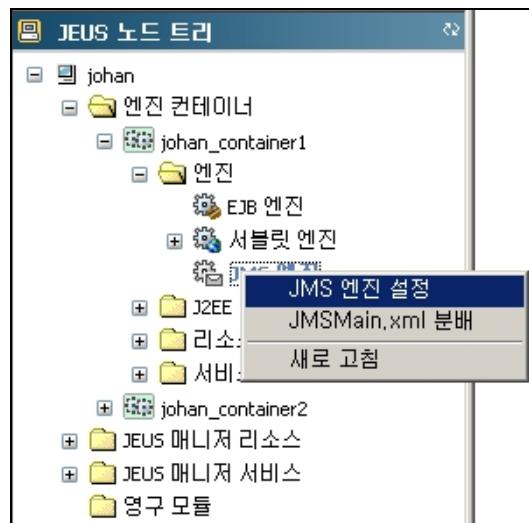


그림 91 JMS 엔진 설정 선택

- 하위 탭 패널에서 기본 설정 탭을 선택한다.
- 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.

4. 입력이 완료되면 확인 버튼을 클릭한다.



그림 92 JMS 엔진 기본 설정

12.2.2 서버 채널 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 서버 채널 설정

메시징 서비스를 제공하기 위한 서버 채널에 대한 설정으로 최소 하나 이상 설정되어야 한다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭 패널에서 **서버 채널 설정** 탭을 선택한다.
3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 **추가** 버튼을 클릭한다.

JMS 엔진 - 서버 채널

메시징 서비스를 제공하기 위한 서버 채널에 대한 설정이다. 최소 하나 이상 설정되어야 한다.

이름	포트	SSL 사용	서버 주소
jms1	9910	false	127.0.0.1

Name * 서버 채널의 이름이다. Connection Factory 에 채널 정보를 지정하기 위해 용도로 사용된다.

Port * 해당 서버 채널의 서비스 포트 번호이다.

Use Ssl * 서버 채널에 SSL 을 적용할 것인지를 설정한다.

고급 선택사항

추가

그림 93 JMS 서버 채널 설정

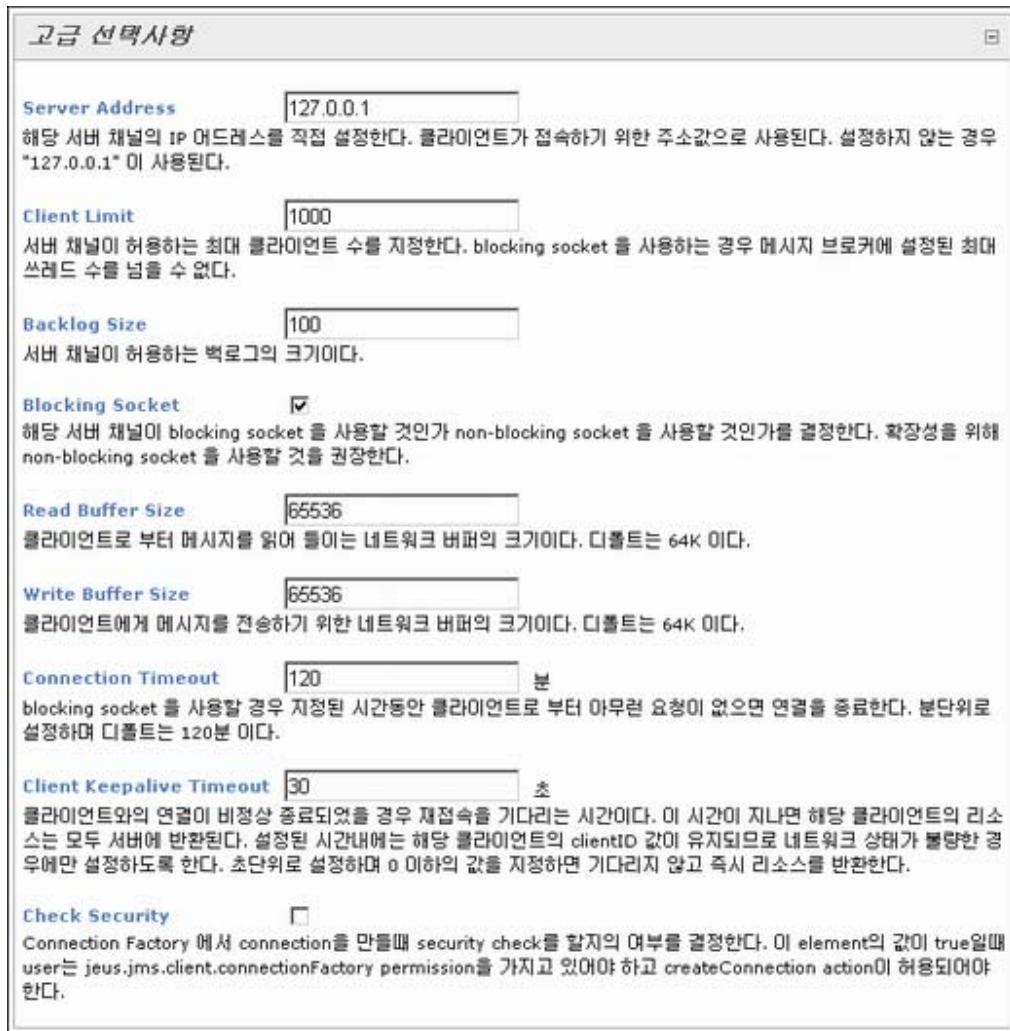


그림 94 JMS 서버 채널 고급 선택 사항

12.2.3 에러 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 에러 로그

에러 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

12.2.4 액세스 로그 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 액세스 로그

엑세스 로그 관련 설정은 24.5 로그 서비스에서 설명할 것이다.

12.2.5 스토리지 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 스토리지

JMS 서버가 다운이 될 경우에, 백업이나 복구기능을 제공하기 위한 설정이다. 파일이나 데이터베이스에서 하나를 설정해야 한다.

다음은 데이터베이스일 경우 설정 방법이다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 JMS 엔진을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭 패널에서 **스토리지 탭**을 선택한다.
3. **스토리지 종류**에서 **DBStorage**를 선택한다.



그림 95 DBStorage 선택

4. **DataSource**에서 적절한 데이터 소스 이름을 선택한다. 그리고 각 테이블 이름을 지정한다.
5. 입력이 완료되면 **확인** 버튼을 클릭한다.

JMS 엔진 - 스토리지

persistent message 를 위한 storage 정보를 설정한다.

스토리지 종류

database storage 를 사용하는 경우 database storage 에 대한 설정을 한다.

데이터 소스

Data Source Name 사용하고자 하는 JNDI 에 binding 된 datasource 의 이름을 지정한다.

Vendor 해당 DataSource가 연결된 DB 종류를 명시한다. 지정되어 있지 않으면 JEUSMain.xml에 등록된 해당 DataSource의 vendor가 사용된다.

Destination Table Name JMS에서 사용하는 destination table name을 지정한다.

Message Table Name JMS에서 사용하는 message table name을 지정한다.

Relation Table Name JMS에서 사용하는 relation table name을 지정한다.

Subscription Table Name JMS에서 사용하는 subscription table name을 지정한다.

Transaction Table Name JMS에서 사용하는 transaction table name을 지정한다.

Key Table Name JMS에서 사용하는 key table name을 지정한다.

그림 96 DataSource 선택

다음은 파일일 경우 설정 방법이다.

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭 패널에서 **스토리지탭**을 선택한다.
3. **스토리지 종류**에서 **FileStorage** 을 선택한다.



그림 97 FileStorage 선택

4. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 확인 버튼을 클릭한다.



그림 98 FileStorage 설정

참고 : 스토리지 설정을 초기화 하기 위해서는 재설정 버튼을 클릭한다.

12.2.6 쓰레드 풀 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 쓰레드 풀

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭에서 **쓰레드 풀**을 선택한다.
3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

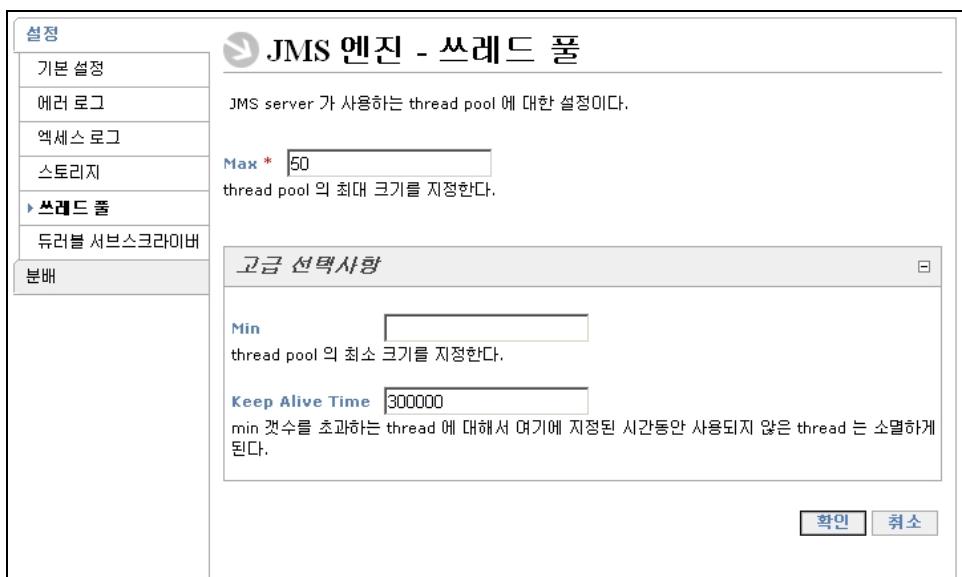


그림 99 JMS 엔진 쓰레드 풀 설정

12.2.7 클러스터 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 클러스터

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭에서 **클러스터**를 선택한다.

3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 추가 버튼을 클릭한다.
5. 클러스터가 추가 되면 설정 페이지 맨 위의 테이블에 설정된 클러스터의 내용이 표시된다.

이름	아이디	아이피 주소	SSL 사용	포트	접속 주기
cluster1	1	127.0.0.1	false	60000	방

Broker Name * []
클러스터간 통신에 사용되는 이름이다. 클러스터링에 참여하는 모든 JMS 메시지 브로커는 각각 유일한 값을 가져야 한다.

Broker Id * []
클러스터간의 통신에 사용되는 브로커 아이디이다. 클러스터에 참여하는 모든 JMS 메시지 브로커는 각각 유일한 값을 가져야 한다. 값은 지정하지 않을 경우 cluster 테이블에 설정된 순서대로 임의 지정된다.

Ip Address * []
해당 JMS 메시지 브로커의 ip-address 주소값이다.

Use Ssl * []
SSL을 사용할 것인지를 설정한다.

고급 선택사항

Jms Port []
해당 JMS 메시지 브로커의 service port 값이다.

Connection Interval [60000]
다른 클러스터 서버와의 connection 을 시도하는 주기이다.

Weight [1]
server-weighted 방식의 메시지 분산 방식을 사용하는 경우 이 값이 사용된다.

추가

그림 100 JMS 엔진 클러스터 설정

12.2.8 듀러블 서브스크라이버 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 설정 > 듀러블 서브스크라이버

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭에서 **듀러블 서브스크라이버**를 선택한다.
3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.

4. 입력이 완료되면 **추가** 버튼을 클릭한다.
5. 드리블 서브스크라이버가 추가 되면 설정 페이지 맨 위의 테이블에 설정된 드리블 서브스크라이버의 내용이 표시된다.

클라이언트 아이디	이름	대스터네이션 이름	메세지 선택자
client1	dsubscriber1	dest1	

Client Id *
이 태그는 클라이언트를 식별하는 값을 설정한다. connection-factory element 내에서 뿐만 아니라 durable-subscriber element 내에서 모든 client-id 값을 중복 되어서는 안된다.

Name *
JMS 메시지 브로커 내에서 관리 목적으로 사용되는 Durable Subscriber의 이름이다.

Destination Name *
Durable Subscriber가 메시지를 받고자 하는 대스터네이션의 이름이다.

Message Selector
Durable Subscriber의 message selector를 설정한다.

그림 101 JMS 엔진 드리블 서브스크라이버 설정

12.2.9 큐 클러스터 설정

[JEUS 매니저](#) > [엔진 컨테이너 개요](#) > [엔진 컨테이너](#) > [엔진 개요](#) > [JMS 엔진](#) > [설정](#) > [큐 클러스터](#)

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭에서 **큐 클러스터**를 선택한다.
3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.

주의 : target 항목의 값은 필수이므로 반드시 값을 입력해야 한다. 또한 이 값은 반드시 클러스터에 참여하는 broker name 이어야 한다.

4. 입력이 완료되면 **추가** 버튼을 클릭한다.
5. 추가된 큐 클러스터의 내용은 상단의 테이블에 표시된다.

JMS 엔진 - 큐 클러스터

이름	분산 방식	로열 배증	대상
qcluster1	round-robin	1	aa, bb
qcluster2	random	2	aa

Queue Cluster Name *

queue 클러스터링을 위한 클러스터링 태입을 설정한다.

Cluster Distribute queue 클러스터링에 참여하는 각각의 브로커에 메시지를 분산하기 위한 방식을 지정한다. server-weighted, consumer-weighted, round-robin, random 중에 하나를 지정할 수 있다.

Local Preference local 브로커에서 처리하는 메시지의 비중을 높인다. server-weighted 혹은 consumer-weighted 방식을 사용할 경우에만 의미가 있다.

Target * 클러스터링에 참여하는 broker name 을 지정한다. 해당 이름은 jms-server-info 에 설정된 이름이어야 한다.

그림 102 JMS 엔진 큐 클러스터 설정

12.2.10 토픽 클러스터 설정

[JEUS 매니저](#) > [엔진 컨테이너 개요](#) > [엔진 컨테이너](#) > [엔진 개요](#) > [JMS 엔진](#) > [설정](#) > 토픽 클러스터

1. 노드 트리에서 엔진폴더 밑에 존재하는 **JMS 엔진**을 클릭하거나 컨텍스트 메뉴에서 **JMS 엔진 설정**을 선택한다.
2. 하위 탭에서 **토픽 클러스터**를 선택한다.
3. 각 항목의 값을 입력한다. 변경된 항목은 다음 부트시에 적용된다.

주의 : target 항목의 값은 필수이므로 반드시 값을 입력해야 한다. 또한 이 값은 반드시 클러스터에 참여하는 broker name 이어야 한다.

4. 입력이 완료되면 **추가** 버튼을 클릭한다.
5. 추가된 토픽 클러스터의 내용은 상단의 테이블에 표시된다.

이름	전달 방식	대상
tdcluster1	non-durable	aa
tdcluster2	durable	aa, bb

Topic Cluster Name *: topic 클러스터링을 위한 클러스터링 타입을 설정한다.

Relay Type: non-durable
위에서 지정된 cluster 서버에 message 를 전달하는 방식이다. non-durable 과 durable을 지정할 수 있다.

Target *: 클러스터링에 참여하는 broker name 을 지정한다. 해당 이름은 jms-server-info 에 설정된 이름이어야 한다.

추가

그림 103 JMS 엔진 토픽 클러스터 설정

12.3 JMS 엔진

12.3.1 JMS 엔진 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 제어 > 시작

1. 노드 트리에서 엔진 폴더를 클릭 한다.
2. 엔진 목록에서 타입이 “jms”인 엔진의 시작 버튼을 클릭 한다.
3. 잠시 후 JMS 엔진이 시작되고 노드 트리의 JMS 엔진 아이콘()이 활성화 된다.

이름	엔진 종류	상태	명령
johan_ejb_engine1	ejb	Running	시작 다운
johan_jms_engine1	jms	Stopped	시작 다운
johan_servlet_engine1	web	Running	시작 다운

새 웹 서버 엔진 생성

그림 104 JMS 엔진 시작

12.3.2 JMS 엔진 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 제어 > 다운

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 엔진 목록에서 타입이 “jms”인 엔진의 다운 버튼을 클릭한다.
3. 잠시 후 JMS 엔진이 다운되고 노드 트리의 JMS 엔진 아이콘()이 비활성화 된다.



그림 105 JMS 엔진 다운

12.4 JMSMain.xml 분배

JMSMain.xml 분배는 현재 선택된 노드의 *JMSMain.xml* 을 전체 JMS 엔진에 동일하게 복사하는 것을 말한다. 이 기능은 JMS 엔진을 시작하기 전에 분배를 통해서 동일한 설정을 적용 할 경우에만 사용하는것으로 JMS 엔진들이 이미 시작되었을 경우에는 사용하지 않는 것이 좋다.

JMS 엔진의 경우 JMS 엔진 Port 가 포함되어 있으므로 분배후에는 동일한 노드에 존재하는 JMS 엔진의 경우 JMS 엔진 Port 가 중복되지 않게 설정해 주어야 한다.

12.4.1 JMSMain.xml 분배

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > JMS 엔진 > 분배

1. 노드 트리에서 JMS 엔진을 클릭한 후 주 화면에서 **분배**탭을 선택하거나 컨텍스트 메뉴에서 **JMSMain.xml 분배**를 선택한다.

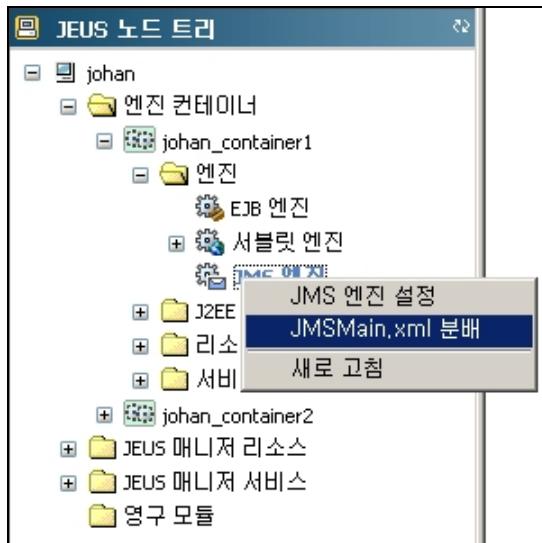


그림 106 JMSMain.xml 분배 선택

2. 화면의 엔진 탐색기에서 분배하고자 하는 JMS 엔진들을 선택한다.



그림 107 JMS 엔진 선택

3. 분배버튼을 클릭한다.

13 웹 서버 엔진

웹 서버 엔진은 JEUS의 앞단에 위치하며, HTTP 클라이언트들과 JEUS 시스템을 연동시키기 위해서 서블릿 엔진과 연결되어 있다.

웹 서버 엔진을 설정할 때 사용되는 웹 서버는 JEUS 웹 서버이다. JEUS 웹 서버는 WebtoB 웹 서버(표준 version)의 기능인 클러스터링 같은 일부기능이 제외된 웹 서버이다.

이번 장에서는 웹 서버 엔진의 구성 및 설정, 모니터링에 대해서 알아볼 것이다.

13.1 웹 서버 엔진 구성

13.1.1 웹 서버 엔진 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 새 웹 서버 엔진 추가

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진폴더를 클릭한다.
2. 하단의 새 웹 서버 엔진 추가를 클릭한다.



그림 108 웹 서버 엔진 추가 링크

3. 웹 서버 엔진생성 폼페이지가 나타나면 엔진 이름 필드에 추가하고자 웹 서버 엔진의 이름을 입력하고 기존에 같은 이름으로 존재하는 설정파일을 그대로 사용하고 싶을 경우는 기존 설정 사용을 선택한다.
4. 입력이 완료되면 생성 버튼을 클릭한다.

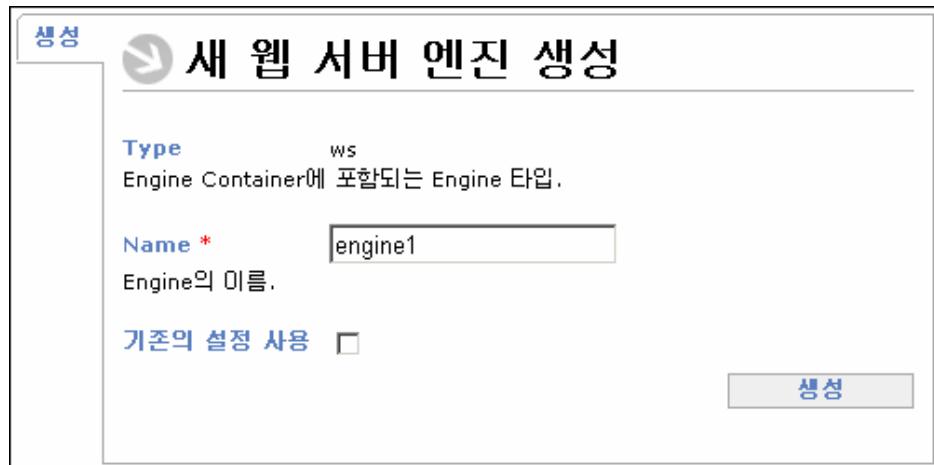


그림 109 웹 서버 엔진 추가

5. 웹 서버 엔진이 생성되면 엔진폴더 밑에 웹 서버 엔진 아이콘()이 비활성 상태로 추가되며 엔진페이지에 웹 서버 엔진이 추가되어 나타난다.

13.1.2 웹 서버 엔진 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 엔진 삭제

1. 노드 트리에서 엔진 컨테이너 밑에 존재하는 엔진폴더를 클릭한다.
2. 엔진 목록에서 “ws” 타입의 엔진에서 를 클릭한다.
3. 웹 서버 엔진이 제거되면 엔진폴더 밑에 웹 서버 엔진 아이콘이 삭제되고 엔진 개요 페이지의 엔진 목록에서도 삭제된다.

이름	엔진 종류	상태	명령
johan_ejb_engine1	ejb	Running	<button>시작</button> <button>다운</button>
johan_jms_engine1	jms	Running	<button>시작</button> <button>다운</button>
johan_servlet_engine1	web	Running	<button>시작</button> <button>다운</button>
johan_ws_engine1	ws	Stopped	<button>시작</button> <button>다운</button> 삭제

그림 110 웹 서버 엔진 삭제

13.2 웹 서버 엔진 설정

13.2.1 노드 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 노드

- 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.

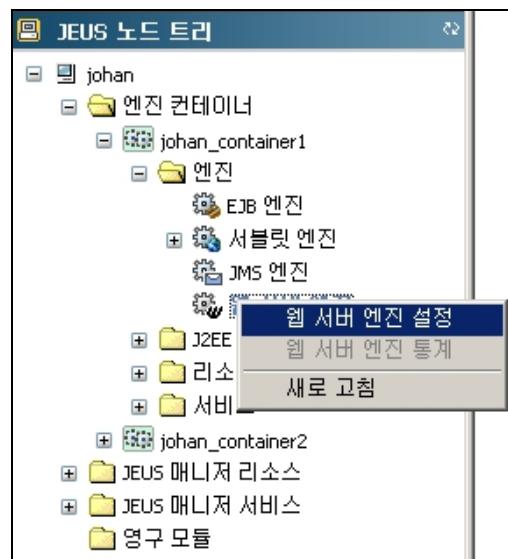


그림 111 웹 서버 엔진 설정 선택

- 하위 탭에서 노드를 선택한다.

3. 각 항목의 값을 입력한다. 변경된 항목은 다시 부트시에 적용된다.

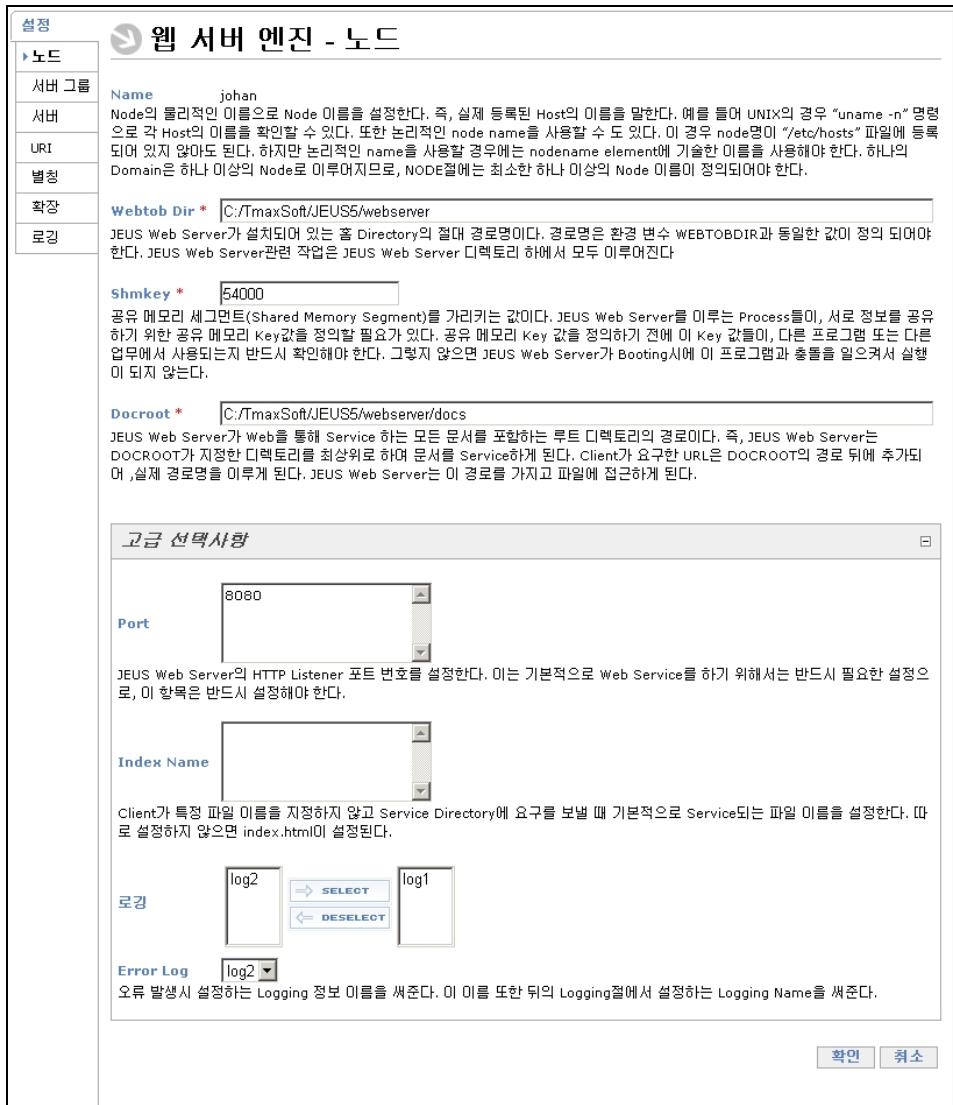


그림 112 웹 서버 엔진 노드 설정

4. 입력이 완료되면 확인 버튼을 클릭한다.

13.2.2 서버 그룹 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 설정 > 서버 그룹 > 새 서버 그룹 생성

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.

2. 하위 탭에서 서버 그룹을 선택한다.
3. 서버 그룹 이름과 서버 타입을 입력한다. Server Type에는 HTML, CGI, SSI, PHP, JSV가 있다. 서블릿 엔진과 연동하기 위해서는 **JSV**를 선택해야 한다.
4. 입력이 완료되면 추가버튼을 클릭한다.

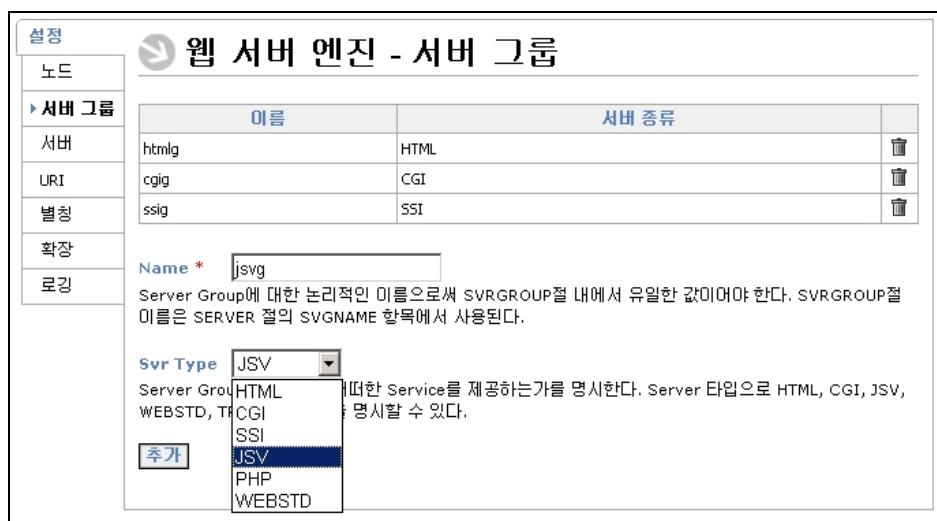


그림 113 서버 그룹 추가

13.2.3 서버 그룹 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 서버 그룹 > 서버 그룹 삭제

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 서버 그룹을 선택한다.
3. 서버 그룹 목록에서 삭제하고자 하는 서버 그룹을 선택한 후 **삭제** 버튼을 클릭한다.
4. 서버 그룹 목록에서 선택한 서버 그룹이 사라진다.

13.2.4 서버 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 설정 > 서버 > 새 서버 생성

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 설정 웹 서버 엔진을 선택한다.
2. 하위 탭에서 서버를 선택한다.
3. 이름과 서버 그룹을 입력한다.
4. 입력이 완료되면 추가버튼을 클릭한다.

설정

노드

서버 그룹

▶ 서버

URI

별칭

확장

로깅

웹 서버 엔진 - 서버

이름	서버 그룹 이름	최소 프로세스	최대 프로세스
html	htmlg	2	10
cgi	cig	2	10
ssi	ssig	2	10
MyGroup	jsgv	4	10

Name *

Server의 실행 파일 이름으로써 일반적으로 Server 이름은 유일(Unique) 해야 한다. 즉 하나의 Server 이름은 SERVER 절에 단 한번만 정의되어야 한다. 같은 이름을 중복하여 이용하면 환경 파일의 Compile 시에 Error가 발생하게 된다.

Svg Name htmlg

Server가 속해 있는 Server Group을 정의한다. 여기에 사용되는 값은 반드시 SVRGROUP 절에서 정의된 Server Group 이름이어야 한다. Server와 SVRGROUP 절의 연결을 통해서 Server가 어떤 Node에서 동작할 것인지, 어떤 리소스 매니저(데이터 베이스/데이터 베이스)를 사용하는지 알 수 있으며, 해당 리소스 매니저를 열 때 필요한 파라미터를 넘겨 줄 수 있다.

고급 선택사항

Min Proc

기본적으로 기동 될 Server Process의 개수를 결정한다. 기존의 Client/Server 모델에서는 Client 당 Server Process가 하나씩 기동 되는 형식으로 동작하였으나 JEUS Web Server는 그보다 효율적인 구조를 가지고 있다. JEUS Web Server에서는 Server Process의 수는 일정하게 유지하고 하나의 Server Process가 여러 개의 Client 요구를 Service 할 수 있다. MinProc는 이러한 Server Process의 개수를 조절하는 것으로써 운영 경험을 통해 적절한 개수를 지정할 필요가 있다. 이 MinProc는 Server Process의 최소 개수를 나타내는 것으로 처음 JEUS Web Server가 Booting 될 때 시작되는 Process의 수와 같다고 생각하면 된다.

Max Proc

MinProc와 더불어 Server Process 개수를 결정하는 항목이다. MaxProc는 MinProc를 포함하여 추가적으로 기동 시킬 수 있는 Process의 최대 개수이다. Server Process는 기본적으로 JEUS Web Server Booting 시점에 위에서 정의된 MinProc 개수 만큼만 기동 되고, 부하가 높아지는 경우 MaxProc 개수까지 Server Process가 자동적으로 기동 될 수 있다. 이 값 또한 운영 경험을 통해 적절한 개수 조정이 필요하다. 이는 특히 Web Server의 성능에 많은 영향을 줄 수 있는 것이다. 만약 관리자가 자신의 Web Server 가 주로 HTML Service를 위주로 한다면, 이에 대한 Service의 MinProc와 MaxProc를 적당히 큰 값으로 설정하고 다른 것들을 적은 값으로 설정한다면 다른 Web Server들과 유사한 Hardware Overhead를 가지면서도 좋은 성능을 낼 수 있게 된다.

추가

그림 114 서버 추가

13.2.5 서버 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 서버 > 서버 삭제

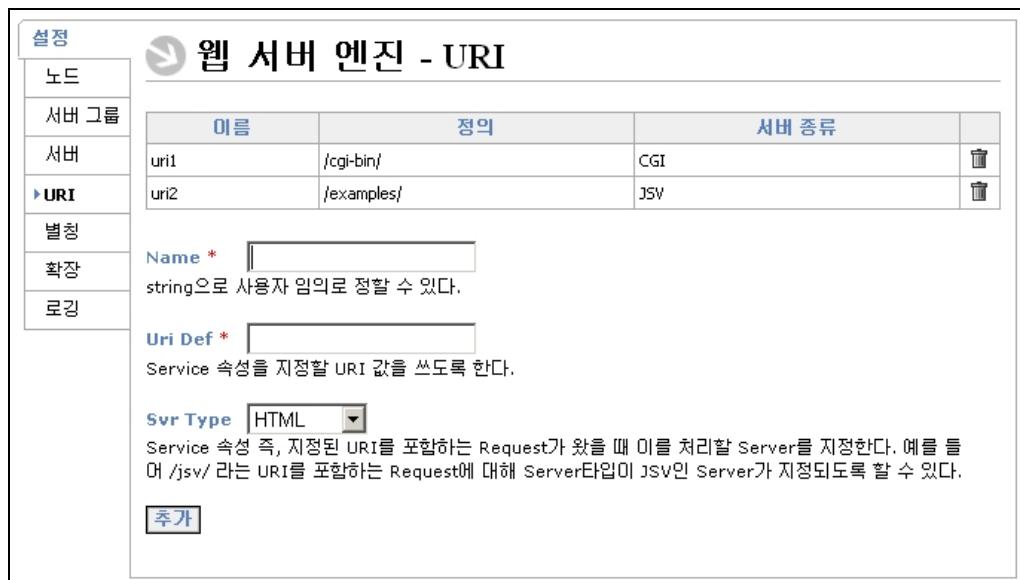
1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 서버를 선택한다.

3. 서버목록에서 삭제하고자 하는 서버를 선택한 후 을 클릭 한다.
4. 서버목록에서 선택한 서버가 사라진다.

13.2.6 URI 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > URI > 새 URI 생성

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 **URI**를 선택한다.
3. 이름과 정의, 서버 타입을 입력한다.
4. 입력이 완료되면 추가버튼을 클릭한다.



이름	정의	서버 종류
uri1	/cgi-bin/	CGI
uri2	/examples/	JSV

Name *

string으로 사용자 임의로 정할 수 있다.

Uri Def *

Service 속성을 지정할 URI 값을 쓰도록 한다.

Svr Type

Service 속성 즉, 지정된 URI를 포함하는 Request가 왔을 때 이를 처리할 Server를 지정한다. 예를 들어 /jsv/라는 URI를 포함하는 Request에 대해 Server타입이 JSV인 Server가 지정되도록 할 수 있다.

그룹 115 URI 추가

13.2.7 URI 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > URI > 새 URI 생성

1. 노드 트리에서 엔진 폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 **URI**를 선택한다.
3. URI 목록에서 삭제하고자 하는 URI를 선택한 후 을 클릭한다.
4. URI 목록에서 선택한 URI가 사라진다.

13.2.8 별칭(Alias) 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 별칭 > 새 별칭 생성

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 **별칭** 탭을 선택한다.
3. 입력 항목들을 입력한다.
4. 입력이 완료되면 **추가**버튼을 클릭한다.

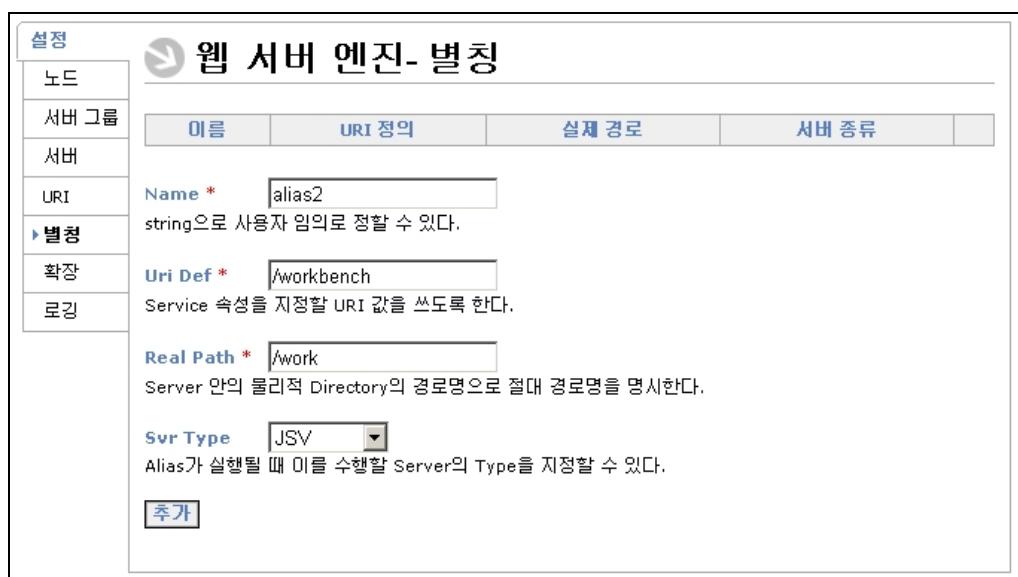


그림 116 별칭 추가

13.2.9 별칭 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 설정 > 별칭 > 별칭 삭제

1. 노드 트리에서 엔진 폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 별칭을 선택한다.
3. 별칭 목록에서 삭제하고자 하는 별칭을 선택한 후 을 클릭한다.
4. 별칭 목록에서 선택한 별칭이 사라진다.

13.2.10 확장(Extension)추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 설정 > 확장 > 새 확장 추가

1. 노드 트리에서 엔진 폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 설정 웹 서버 엔진을 선택한다.
2. 하위 탭에서 확장을 선택한다.
3. 입력 항목들을 입력한다.
4. 입력이 완료되면 추가버튼을 클릭한다.



그림 117 새 확장 추가

13.2.11 확장(Extension) 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 확장 > 확장 삭제

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 확장을 선택한다.
3. 확장목록에서 삭제하고자 하는 확장을 선택한 후 을 클릭한다.
4. 확장 목록에서 선택한 확장이 사라진다.

13.2.12 로깅 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 > 엔진 > 설정 > 로깅 > 새 로깅 추가

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 로깅을 선택한다.
3. 입력 항목들을 입력한다.

4. 입력이 완료되면 추가버튼을 클릭 한다.

이름	파일 이름	포맷	옵션
log1	C:/TmaxSoft/JEUS5/log/access.log	DEFAULT	sync
log2	C:/TmaxSoft/JEUS5/log/error.log	ERROR	sync

Name *
string으로 사용자 임의로 정할 수 있다.

File Name *
로그를 저장할 파일의 경로명과 파일 이름을 설정한다.

Format *
Log File에 기록될 내용과 기록 방식을 설정한다. Format이 따로 설정되지 않으면 JEUS Web Server가 지원하는 default 로그 파일 형식이 사용된다.

Option
Logging 방식을 Option을 줌으로써 바꿀 수 있다.

추가

그림 118 로깅 추가

13.2.13 로깅 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 엔진 > 설정 > 로깅 > 로깅 삭제

1. 노드 트리에서 엔진폴더 밑에 존재하는 웹 서버 엔진을 클릭하거나 컨텍스트 메뉴에서 웹 서버 엔진 설정을 선택한다.
2. 하위 탭에서 로깅을 선택한다.
3. 로깅 목록에서 삭제하고자 하는 로깅을 선택한 후 을 클릭한다.
4. 로깅목록에서 선택한 로깅이 사라진다.

13.3 웹 서버 엔진 제어

13.3.1 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 제어 > 시작

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 엔진 목록에서 타입이 “ws”인 엔진의 시작버튼을 클릭한다.
3. 잠시 후 웹 서버 엔진이 시작되고 노드 트리의 웹 서버 엔진의 아이콘(✳)이 활성화 된다.



The screenshot shows the 'Engine Overview' section of the JEUS Manager. It lists four engines: ejb, jms, web, and ws. The ws engine is currently stopped. There are 'Start' and 'Stop' buttons for each engine, and a status icon (red circle with a white exclamation mark) next to the ws engine's status.

이름	엔진 종류	상태	명령
johan_ejb_engine1	ejb	Running	<button>시작</button> <button>다운</button>
johan_jms_engine1	jms	Running	<button>시작</button> <button>다운</button>
johan_servlet_engine1	web	Running	<button>시작</button> <button>다운</button>
johan_ws_engine1	ws	Stopped	<button>시작</button> <button>다운</button>

그림 119 웹 서버 엔진 시작

13.3.2 다운

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버
엔진 > 제어 > 다운

1. 노드 트리에서 엔진 폴더를 클릭한다.
2. 엔진 목록에서 타입이 “ws”인 엔진의 다운버튼을 클릭한다.
3. 잠시 후 웹 서버 엔진이 다운되고 노드 트리의 웹 서버 엔진아이콘(✳)이 비활성화 된다.

13.4 웹 서버 엔진 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 웹 서버 엔진 > 통계

1. 노드 트리에서 웹 서버 엔진을 선택한 후 하위 탭에서 통계를 선택하거나 컨텍스트 메뉴에서 웹 서버 엔진 통계를 선택한다.

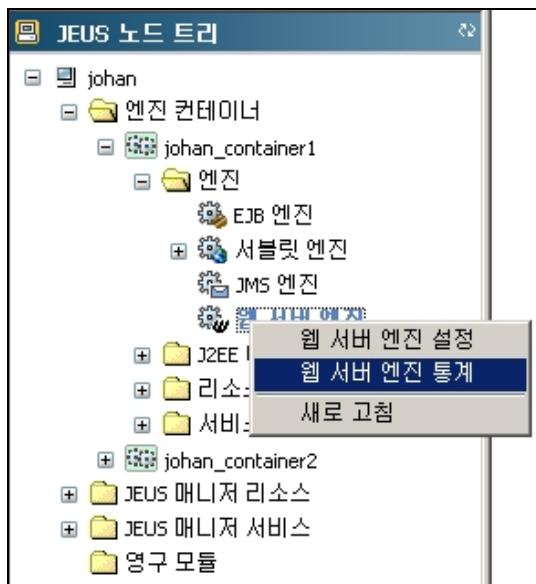


그림 120 웹 서버 엔진 통계 선택

2. 웹 서버 엔진의 모니터링 정보가 출력된다. 서버 상태에는 다음의 3 가지가 존재한다.

상태	설명
READY	서버가 요청 대기중
NOTREADY	서버가 요청을 대기하지 않음
FAIL	서버 기동이 실패됨

설정	웹 서버 엔진 - 통계					
통계	서버 이름	상태	최소 프로세스	최대 프로세스	실행 수	평균 수행 시간(밀리초)
	html	READY	2	10	0	0.0
	cgi	READY	2	10	0	0.0
	ssi	READY	2	10	0	0.0
	MyGroup	NOTREADY	4	10	0	0.0

표 편집

[새로 고침](#)

그림 121 웹 서버 엔진 통계

14 배치

배치는 어플리케이션의 서비스들을 시작하기 위해서 JEUS에 모듈 파일을 올리고 제어하는 모든 동작을 말한다.

JEUS에서 제공하는 배치에는 표준 배치, 영구적인 배치와 자동 배치 3 가지가 있다.

표준 배치는 운영중인 서버에서 배치를 하는 것으로 일반적으로 배치라 함은 표준을 말한다.

영구적인 배치는 서버가 부트되기 전에 *JEUSMain.xml*에 미리 등록해 두었다가 부트되면서 배치를 하는 것으로 서비스 내용에 변경이 없을 시에 이 배치방법을 사용하면 서버가 부트할 때마다 새로이 배치를 할 필요가 없다.

자동 배치는 JEUS가 지정된 디렉토리나 파일을 주기적으로 검사하면서 만일 새로운 어플리케이션 모듈 파일이 생기거나 갱신되었을 경우 자동으로 배치를 하는 것을 말한다. 이 배치를 사용하면 모듈 파일이 바뀔 때마다 배치를 하는 번거로움이 없어 진다.

14.1 표준 배치

일반적으로 배치라 함은 표준 배치를 말한다. JEUS 5은 배치 API를 구현하였으며 표준 배치는 이 API를 이용하여 동작한다.

배치 명령에는 분배, 시작, 정지, 제거(Distribute, Start, Stop, Un 배치)가 있으며 보통 배치라 함은 배치와 시작을 같이 실행하는 경우를 말한다. JEUS의 배치는 기본적으로 소스 자동 분배 방식으로 동작하며 2 단계 배치를 선택적으로 사용할 수 있다. 2 단계 배치 옵션을 사용하면 배치 대상중 하나라도 배치가 실패하면 모든 배치가 실패되는 선택 사항으로 모든 서버에서 동시에 서비스를 시작할 필요가 있을 때 유용하다.

14.1.1 J2EE 어플리케이션 모듈 배치

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈

배치가 가능한 모듈의 종류에는 EAR, JAR, WAR, RAR, CAR 5 가지 종류가 있다. 모든 모듈에 대해서 배치 방법은 동일하며 모듈 선택사항에 있어서 몇 가지 차이가 있으므로 이것에 대해서는 해당 단계를 설명할 때 자세히 알아 볼 것이다.

1. 노드 트리에서 **J2EE 어플리케이션 모듈폴더**를 클릭하거나 컨텍스트 메뉴에서 새 어플리케이션 모듈 배치를 선택한다.

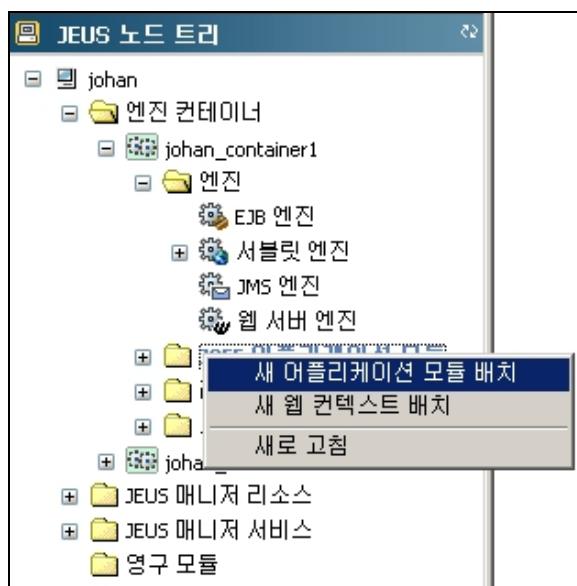


그림 122 새 어플리케이션 모듈 배치 선택

2. 배치할 모듈을 선택한다. 어플리케이션 모듈파일을 선택하는 방법에는 3 가지가 있다.

- 모듈 파일 선택: 어플리케이션 모듈 파일이 JEUS 서버의 JEUS_HOME/webhome/app_home에 존재할 경우 사용한다.
- 파일 업로드: 클라이언트에 존재하는 모듈 파일을 서버에 전송하는 경우 사용한다. 어플리케이션 홈 선택 사항에서는 파일을 업로드할 어플리케이션 홈 디렉토리를 선택한다.
- 절대 경로: 모듈 파일이 서버측에 존재하나 JEUS_HOME/webhome/app_home에 존재하지 않을 경우 사용한다.



그림 123 어플리케이션 모듈 배치 - 1 단계 모듈 선택

배치 대상 탐색기에서 모듈을 배치할 대상들을 선택한다. 최소한 하나이상의 대상이 선택되어야 한다. 만일 모듈이 파일이 아닌 디렉토리인 경우(EXPLODED_COMPONENT나 EXPLODED_EAR)라면 모듈파일을 소유하고 있는 엔진 컨테이너만이 대상이 될 수 있다. 또한 WAR를 포함하는 EAR이거나 WAR 모듈일 경우는 컨텍스트 그룹을 선택할 수 있게 되어있다. 보통 클러스터링 내의 컨텍스트 그룹은 거의 대부분 같은 이름을 사용하나 만일 이름이 다를 경우라면 이곳에 컨텍스트 그룹 이름을 선택 한다.



그림 124 어플리케이션 모듈 배치 - 2 단계 대상 선택

- 각 어플리케이션 모듈별로 선택사항을 입력한다.

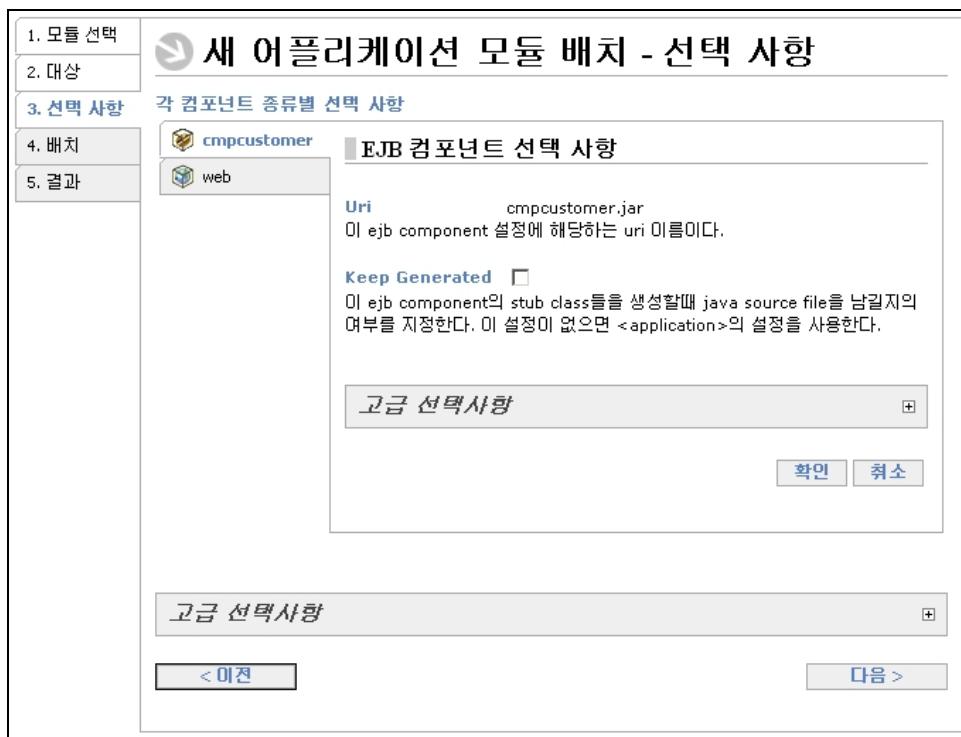


그림 125 J2EE 어플리케이션 모듈 배치 - 3 단계 컴포넌트별 선택사항 입력

- 일반적인 선택 사항을 입력한다.



그림 126 J2EE 어플리케이션 모듈 배치 - 3 단계 일반 선택 사항 입력

5. 2 단계 배치를 할 것인지 여부를 선택한다.

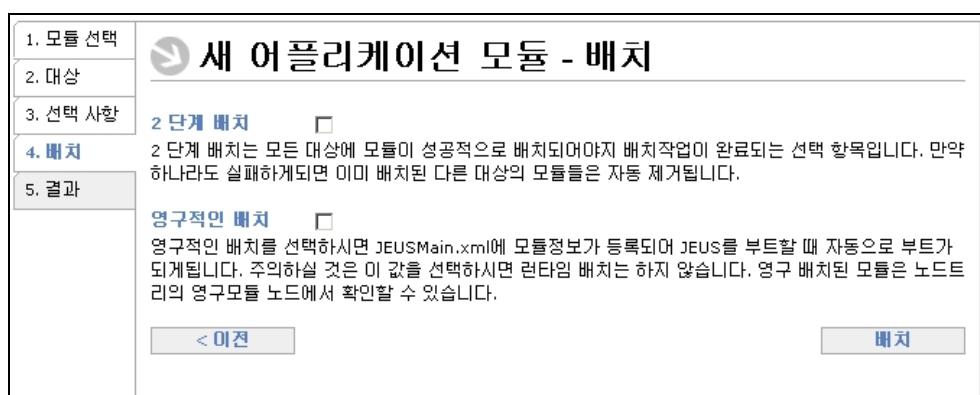


그림 127 J2EE 어플리케이션 모듈 배치 - 4 단계 배치

6. 배치 버튼을 클릭 한다.
7. 배치가 모두 완료되면 완료된 모듈이 노드 트리의 **J2EE 어플리케이션 모듈** 폴더 밑에 나타나면 초기 페이지로 이동한다.
8. 배치가 일부 실패하면 결과탭이 나타나며 실패한 대상과 실패 원인을 볼 수 있다.

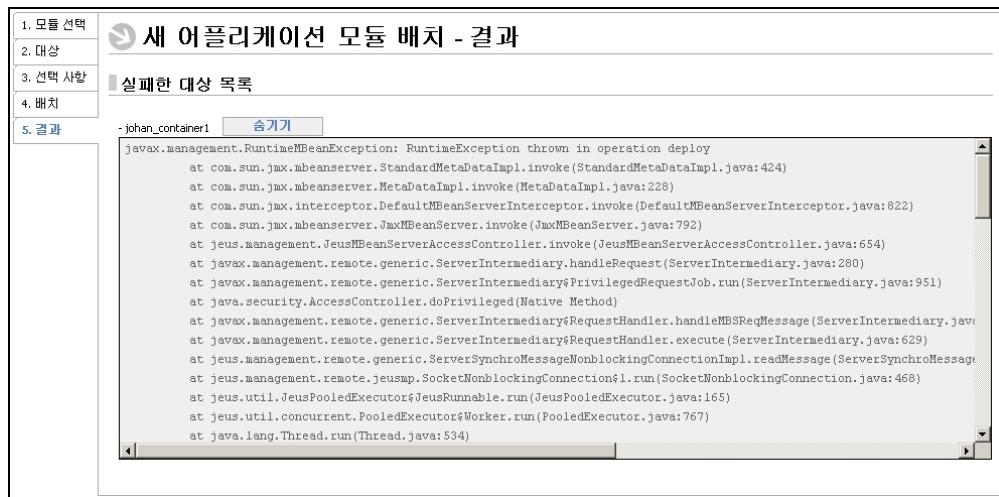


그림 128 J2EE 어플리케이션 모듈 배치-배치 실패

14.1.2 웹 컨텍스트 배치

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈

이 배치는 웹 컨텍스트 디렉토리를 배치할 경우 사용한다. 다시 말해 예전의 방식대로 서블릿 컨텍스트를 배치하고 싶을 경우 사용한다. 그러나, 가능하면 WAR 파일을 만들어서 배치하는 것을 적극 권장한다. 배치방식은 위의 일반적인 배치와 같으나 모듈 파일을 선택하는 것에 있어서 약간의 차이가 있다.

1. 웹 컨텍스트의 배치를 위해선 노드 트리의 **J2EE 어플리케이션 모듈** 폴더의 컨텍스트 메뉴에서 새 배치 웹 컨텍스트를 선택한다.

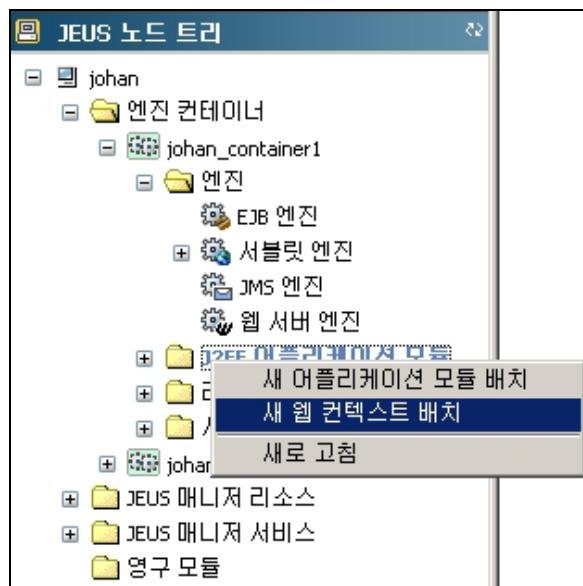


그림 129 웹 컨텍스트 배치 선택

- 파일을 브라우징 할 수 있는 화면이 나타나면 아이콘으로 표시된 파일을 선택한 후 다음버튼을 클릭한다.



그림 130 웹 컨텍스트 배치 - 1 단계 배치 디렉토리 선택

- 이후 과정은 일반적인 배치와 동일하므로 생략한다.

참고 1 : 이 방법으로 배치를 사용할 경우에 배치 대상을 여럿 선택하면 다른 대상으로는 해당 컨텍스트를 WAR 파일로 생성하여 전송하게 된다.

참고 2 : 배치된 후에는 일반 웹 모듈과 동일하게 관리된다.

참고 3 : WAR 파일내의 META-INF/jeus-web-dd.xml 파일에 context-path 설정이 없으면 WAR 모듈의 이름이 context-path의 역할을 대신한다.

14.1.3 시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 모듈 > 시작

일시적으로 멈춰져 있는 모듈을 시작할 때 사용하는 명령이다.

1. 노드 트리의 **J2EE 어플리케이션 모듈** 폴더에서 시작할 모듈을 클릭한 후 배치탭을 선택한 후 하위 탭에서 시작탭을 선택한다.
2. 현재 이 모듈이 배치되고 상태가 **stopped**인 모듈이 대상 탐색기에 나타난다.
3. 대상 탐색기에서 시작하고자 하는 대상을 선택한다.
4. 시작버튼을 클릭한다
5. 모듈들이 시작되면 노드 트리에서 해당 모듈들의 상태가 변경된다.

14.1.4 정지

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 모듈 > 정지

현재 서비스 중인 모듈의 동작을 일시적으로 정지 시킬 때 사용하는 명령이다.

1. **J2EE 어플리케이션 모듈** 폴더에서 정지할 할 모듈을 클릭한 후 배치탭을 선택한 후 하위 탭에서 정지탭을 선택하거나 선택한 모듈의 컨텍스트 메뉴에서 해당 모듈 이름의 정지 항목을 선택한다.

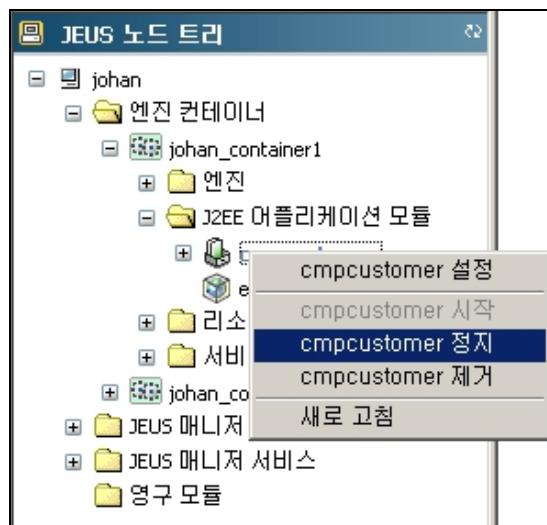


그림 131 모듈 정지 선택

2. 현재 이 모듈이 배치되고 상태가 running 이거나 failed 인 모듈들의 엔진 컨테이너들이 대상 탐색기에 나타난다.
3. 대상 탐색기에서 정지하고자 하는 대상들을 선택한다.



그림 132 모듈 정지

4. 정지버튼을 클릭 한다
5. 모듈들이 정지되면 노드 트리에서 해당 모듈들의 아이콘의 상태가 변경 된다.



그림 133 Module Stop 후 Module 들의 상태

14.1.5 제거

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 모듈 > 제거

배치된 모듈을 제거시킬 때 사용하는 명령이다. EAR에 포함된 모듈들(JAR, WAR, CAR, RAR)의 경우는 따로 제거 되지 않는다.

1. **J2EE 어플리케이션 모듈** 폴더에서 제거할 할 모듈을 클릭한 후 **배치탭**을 선택한 후 하위 탭에서 **제거탭**을 선택하거나 선택한 모듈의 컨텍스트 메뉴에서 **eartest 제거**를 선택한다.

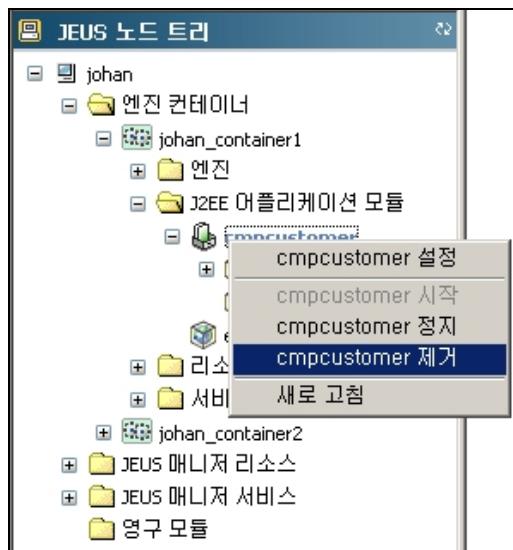


그림 134 모듈 제거 선택

2. 현재 이 모듈이 배치된 모든 대상들이 대상 탐색기에 나타난다.
3. 대상 탐색기에서 제거하고자 하는 모듈이 배치된 대상들을 선택한다.



그림 135 모듈 제거

4. 제거 버튼을 클릭한다
5. 모듈들이 제거되면 노드 트리에서 해당 모듈들이 사라진다.

14.2 영구(Permanent) 배치

영구 배치는 *JEUSMain.xml*에 어플리케이션을 등록한 후 JEUS가 부트할 때 배치하는 방법을 말한다.

14.2.1 배치

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션
모듈

영구 배치는 일반적인 배치에서 추가적인 선택 사항을 입력함으로서 수행된다. 배치 과정 중 마지막 단계에서 영구 배치를 선택하게 되면 영구 배치가 실행된다.

배치가 완료되면 영구 모듈폴더에 배치된 모듈 아이콘이 나타난다.



그림 136 영구 배치 후 모듈이 추가된 모습

14.2.2 Application 설정

JEUS 마니저 > 영구 모듈 > 모듈 > 설정

보통 한번 배치가 되면 JEUSMain.xml 등록된 설정을 변경하는 경우는 거의 없으나 변경하고 싶은 경우라면 이 기능을 사용하여 변경할 수 있다.

단순히 XML을 직접 편집하는 것이므로 Application의 요소에 대해서 잘 알고 있어야 한다.

1. 노드 트리에서 영구 모듈 폴더의 하위 디렉토리에서 설정을 변경하고자 하는 모듈을 선택하고 설정탭을 선택하거나 컨텍스트 메뉴에서 해당 모듈 이름의 설정을 선택한다.
2. 설정을 편집한 후 확인 버튼을 클릭한다.

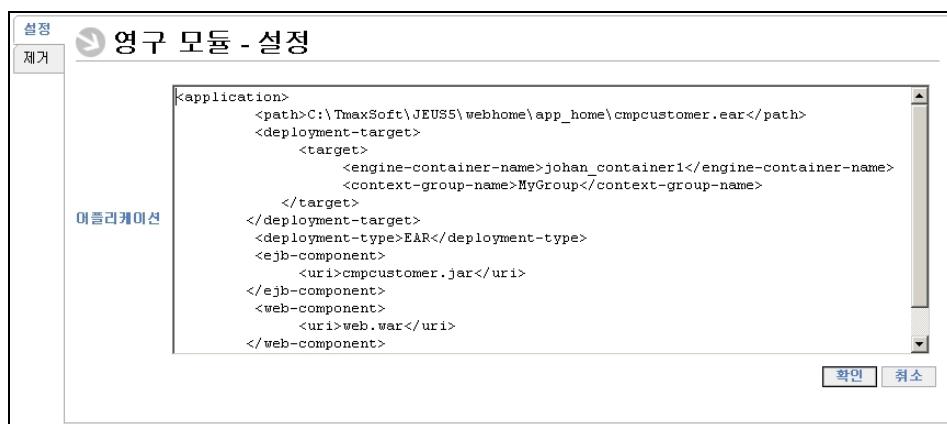


그림 137 영구 모듈의 설정

14.2.3 제거

JEUS 마니저 > 영구 모듈 > 모듈 > 제거

영구 모듈의 제거는 단순히 JEUSMain.xml에서 해당 모듈의 설정을 삭제하는 것이다. 이 명령을 실행하면 모든 Node로부터 해당 모듈 설정이 삭제된다.

1. 노드 트리의 영구 모듈 폴더의 하위 디렉토리에서 모듈을 선택하고 제거탭을 선택하거나 컨텍스트 메뉴에서 모듈의 제거를 선택한다.
2. 제거를 클릭한다.

- 노드 트리의 영구 모듈디렉토리에서 해당 모듈이 삭제된다.

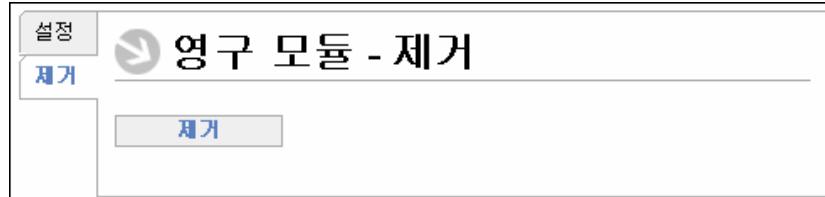


그림 138 영구 모듈의 제거

14.3 자동 배치

자동 배치는 엔진 컨테이너의 자동 배치 설정이나 모듈 배치시 Application 설정에서 주기 적인 검사 시간을 지정함으로써 수행된다.

좀더 자세한 설명은 6.2.4_자동 배치 설정을 참조하기 바란다.

15 J2EE 모듈

J2EE 표준 모듈에는 5 가지 타입이 있다. 어플리케이션 모듈, EJB 모듈, 웹 모듈, 리소스 아답터 모듈, 어플리케이션 클라이언트 모듈.

이번 장에서는 이들 각 모듈의 설정과 설정에 대해서 알아 볼 것이다.

15.1 어플리케이션 모듈

어플리케이션 모듈은 EJB JAR, WAR, CAR, RAR 모듈 파일들을 하나의 파일에 묶어 놓은 모듈이다.

15.1.1 어플리케이션 모듈 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 설정

1. **J2EE 어플리케이션 모듈** 폴더에서 설정을 볼 어플리케이션 모듈을 클릭한 후 설정 탭을 선택한다.
2. “*application.xml*” 설정 파일의 내용이 출력된다.



The screenshot shows the JEUS application configuration interface. On the left, there's a navigation tree with '설정' (Configuration) at the top, followed by 'application.xml' and '배치'. The main area is titled 'J2EE 어플리케이션 - 배치 기술자' (J2EE Application - Deployment Technology). It displays the XML code for the 'application.xml' file:

```
<!DOCTYPE application PUBLIC '-//Sun Microsystems, Inc./DTD J2EE Application 1.3//EN' 'http://java.sun.com/dtd/application_1_3.dtd'>
<application>
  <display-name>cmpcustomer</display-name>
  <module>
    <ejb>
      <ejb-jar>cmpcustomer.jar</ejb-jar>
    </ejb>
    <module>
      <web>
        <web-uri>web.war</web-uri>
        <context-root>customer</context-root>
      </web>
    </module>
  </module>
</application>
```

그림 139 어플리케이션 배치 기술자

15.2 EJB 모듈

15.2.1 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > EJB 모듈 > 설정

1. J2EE 어플리케이션 모듈 폴더에서 설정을 볼 EJB 모듈을 클릭한 후 설정 탭을 선택한다.
2. 하위 탭에서 “ejb-jar.xml”이나 “jeus-ejb-dd.xml”을 선택한다.
3. 선택된 설정 파일의 내용이 출력된다.

15.2.2 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > EJB 모듈 > 통계

1. J2EE 어플리케이션 모듈 폴더에서 통계 정보를 볼 EJB 모듈을 선택한 후 통계탭을 선택하거나 모니터링 하고자 하는 EJB 모듈의 컨텍스트 메뉴에서 EJB 모듈이름의 통계를 선택한다.
2. EJB 목록과 모니터링 정보가 출력된다.

The screenshot displays the 'EJB Engine - Statistics' interface. On the left, there are tabs for '설정' (Configuration), '통계' (Statistics, which is selected), and '배치' (Batch). The main title is 'EJB 엔진 - 통계'. Below the title is a table with the following data:

빈 이름	생성	삭제	요청	커밋	룰백
CustomerBean	0	0	0	0	0
AddressBean	0	0	0	0	0
SubscriptionBean	0	0	0	0	0

At the bottom right of the table, there is a link '표 편집' (Edit Table). A blue button at the bottom right says '새로 고침' (Refresh).

그림 140 EJB 모듈 통계

15.2.3 EJB 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > EJB 모듈 > EJB > 통계

1. 노드 트리에서 모니터링 하고자 하는 EJB를 클릭한 후 통계탭을 선택하거나 모니터링 하고자 하는 EJB의 컨텍스트 메뉴에서 **EJB 이름의 통계**를 선택한다.
2. EJB 통계 정보가 출력된다.

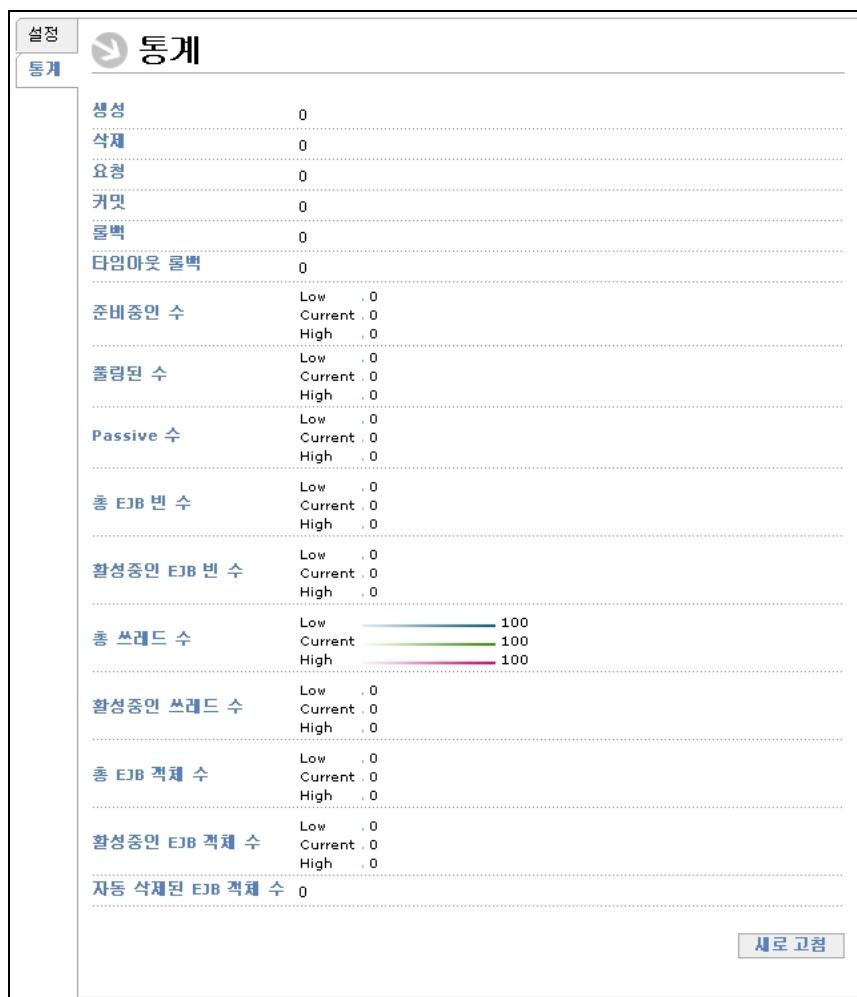


그림 141 EJB 통계

15.3 웹 모듈

15.3.1 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 웹 모듈 > 설정

1. J2EE 어플리케이션 모듈 폴더에서 설정을 볼 웹 모듈을 클릭한 후 설정 탭을 선택한다.
2. 하위 탭에서 “web.xml”이나 “jeus-web-dd.xml”을 선택한다.
3. 선택된 설정 파일의 내용이 출력된다.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<jeus-web-dd xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
    <context-path>/examples</context-path>
    <docbase>examples</docbase>
</jeus-web-dd>
```

그림 142 웹 모듈의 배치 기술자

15.3.2 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 웹 모듈 > 통계

1. J2EE 어플리케이션 모듈 폴더에서 설정을 볼 웹 모듈을 클릭한 후 통계 탭을 선택하거나 컨택스트 메뉴에서 웹 모듈 이름의 통계를 선택한다.
2. 하위 탭에서 기본 정보를 선택한다.
3. 웹 모듈의 통계 정보가 출력된다.



그림 143 웹 모듈 기본 통계

15.3.3 서블릿 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 웹 모듈 > 통계 > 서블릿

1. **J2EE 어플리케이션 모듈 폴더**에서 설정을 볼 웹 모듈을 클릭한 후 **통계** 탭을 선택하거나 컨택스트 메뉴에서 **웹 모듈 이름의 통계**를 선택한다.
2. 하위 탭에서 **서블릿**을 선택한다.
3. 서블릿의 모니터링 정보가 출력된다. 서블릿 상태에는 다음의 3 가지가 있다.

상태	설명
Ready	서블릿 클래스가 로드 되어서 서비스 할수 있게 준비된 상태
Not Loaded	서블릿 클래스가 로드 되지 않은 상태
Suspend	서블릿이 중단된 상태

15.3.4 서블릿 일시 정지(Suspend)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 웹 모듈 > 통계 > 서블릿 > 일시 정지

임시적으로 서블릿 종료한다. 임시 종료된 서블릿은 클라이언트 요청을 서비스하지 못한다.

1. 노드 트리의 **J2EE 어플리케이션 모듈** 폴더에서 제어를 할 서블릿이 포함된 웹 모듈을 클릭한 후 통계탭을 선택하거나 컨텍스트 메뉴에서 웹 모듈 이름의 통계를 선택한다.
2. 하위 탭에서 서블릿을 선택한다.
3. 서블릿 목록에서 정지하고자 하는 서블릿의 정지버튼을 클릭한다.
4. 정지가 실행되고 해제버튼이 활성화 된다.

15.3.5 서블릿 정지 해제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > **J2EE 어플리케이션 모듈** > 어플리케이션 모듈 > 웹 모듈 > 통계 > 서블릿 > 정지 해제

임시적으로 종료된 서블릿 을 다시 서비스 상태로 돌린다.

1. 노드 트리의 **J2EE 어플리케이션 모듈** 폴더에서 제어를 할 서블릿이 포함된 웹 모듈을 클릭한 후 통계탭을 선택하거나 컨텍스트 메뉴에서 웹 모듈 이름의 통계를 선택한다.
2. 하위 탭에서 서블릿을 선택한다.
3. 서블릿 목록에서 정지 해제할 서블릿 의 해제버튼을 클릭한다.
4. 정지 해제가 실행되고 정지버튼이 활성화 된다.

15.3.6 서블릿 종료

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > **J2EE 어플리케이션 모듈** > 어플리케이션 모듈 > 웹 모듈 > 통계 > 서블릿 > 종료

웹 모듈에서 서블릿 제거한다. 모든 파일들과 설정들은 그대로 보존 된다.

1. 노드 트리에서 **J2EE 어플리케이션 모듈** 폴더에서 제어를 할 서블릿이 포함된 웹 모듈을 클릭한 후 통계탭을 선택하거나 컨텍스트 메뉴에서 웹 모듈 이름의 통계를 선택한다.
2. 하위 탭에서 서블릿을 선택한다.

3. 서블릿 목록에서 종료하고자 하는 서블릿의 종료버튼을 클릭한다.

15.3.7 서블릿 재시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 웹 모듈 > 통계 > 서블릿 > 재시작

서블릿을 재시작 한다.

1. 노드 트리에서 **J2EE 어플리케이션 모듈** 폴더에서 설정을 볼 웹 모듈을 클릭한 후 통계탭을 선택하거나 컨텍스트 메뉴에서 웹 모듈 이름의 경지를 선택한다.
2. 하위 탭에서 서블릿을 선택한다.
3. 서블릿 목록에서 재시작하고자 하는 서블릿의 재시작버튼을 클릭한다.

15.4 리소스 아답터 모듈

15.4.1 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 리소스 아답터 모듈 > 설정

1. **J2EE 어플리케이션 모듈** 폴더에서 설정을 볼 리소스 아답터 모듈을 클릭한 후 설정 탭을 선택한다.
2. 하위 탭에서 “ra.xml”이나 “jeus-connector-dd.xml”을 선택한다.
3. 선택된 설정 파일의 내용이 출력된다.

15.4.2 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 리소스 아답터 모듈 > 통계

1. **J2EE 어플리케이션 모듈** 폴더에서 모니터링 할 리소스 아답터 모듈을 클릭한 후 통계탭을 선택한다.
2. 모니터링 정보가 출력된다.



그림 144 리소스 아답터 모듈 모니터링

15.5 어플리케이션 클라이언트 모듈

15.5.1 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > J2EE 어플리케이션 모듈 > 어플리케이션 모듈 > 어플리케이션 클라이언트 모듈 > 설정

1. **J2EE 어플리케이션 모듈** 폴더에서 설정을 볼 어플리케이션 클라이언트 모듈을 클릭한 후 설정 탭을 선택한다.
2. 하위 탭에서 “application-client.xml”이나 “jeus-client-dd.xml”을 선택한다.
3. 선택된 설정 파일의 내용이 출력된다.

16 엔진 컨테이너 리소스

엔진 컨테이너의 리소스에는 JDBC 데이터 소스, JCA, WebT, JTA, JMS 가 있다. JDBC 데이터 소스와 WebT 데이터 소스는 JEUS 매니저에서 추가 가능하며 엔진 컨테이너에서는 단지 모니터링과 제어만 가능하다. JTA는 엔진 컨테이너마다 항상 존재하는 리소스이며 JMS 리소스는 JMS 엔진이 작동할 때만 사용 가능하다. JCA 리소스는 리소스 아답터 모듈을 배치하였을 경우 사용 가능하다.

따라서 JDBC 데이터 소스와 WebT 데이터 소스의 구성에 관한 내용은 각각 18_JDBC 데이터 소스와 22_WebT 을 참조하고 JMS 리소스는 19_JMS 리소스, JCA 리소스는 14 배치를 참조하기 바란다.

16.1 JDBC 데이터 소스

JDBC 데이터 소스는 JEUS 매니저 리소스에서 정의되며 어떤 엔진 컨테이너에서 JNDI를 통해 특정 JDBC 데이터 소스를 처음으로 Lookup 하였을 경우 엔진 컨테이너에 JDBC 데이터 소스가 나타난다.

16.1.1 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 통계

현재 이 엔진 컨테이너에서 사용 중인 JDBC 데이터 소스를 모두 모니터링 할 수 있는 기능이다.

1. 노드 트리에서 **JDBC** 폴더를 클릭한다.
2. JDBC 데이터 소스의 목록과 통계 정보가 출력된다.
3. 필요에 따라 아래쪽의 **새로 고침** 버튼을 이용하여 통계 정보를 갱신한다.

The screenshot shows a table titled "JDBC 데이터 소스" (JDBC Data Source) with two rows of data. The columns are: 노출 이름 (Name), 생성 (Create), 닫음 (Close), 폴 크기 (Pool Size), 사용 가능한 폴 크기 (Available Pool Size), 대기중인 쓰레드 (Waiting Threads), 제연결 (Established Connections), and 명령 (Command). The first row has values: datasource2, 4, 0, 4, 1, 0, 0, 불가능 (Not Possible), and 연결 재조정 (Reconnect). The second row has values: datasource1, 4, 0, 4, 4, 0, 0, 불가능 (Not Possible), and 연결 재조정 (Reconnect). There are also buttons for 표 편집 (Table Edit) and 새로 고침 (Refresh).

노출 이름	생성	닫음	폴 크기	사용 가능한 폴 크기	대기중인 쓰레드	제연결	명령
datasource2	4	0	4	1	0	0	불가능 연결 재조정
datasource1	4	0	4	4	0	0	불가능 연결 재조정

그림 145 JDBC 데이터 소스 모니터링

16.1.2 가능

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 통계 > 가능

이 기능은 불가능 명령에 의해 작동이 정지된 데이터 소스를 작동 가능하게 하는 명령이다.

1. 노드 트리에서 **JDBC** 폴더를 클릭 한다.
2. JDBC 데이터 소스의 목록에서 가능버튼을 클릭 한다.
3. 데이터 소스가 가능되면 버튼의 레이블이 불가능으로 변경된다.

16.1.3 불가능

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 통계 > 불가능

이 기능은 현재 작동중인 데이터 소스의 서비스를 정시시킬 때 사용한다.

1. 노드 트리에서 **JDBC** 폴더를 클릭 한다.
2. JDBC 데이터 소스의 목록에서 불가능버튼을 클릭 한다.
3. 데이터 소스가 불가능 하게되면 버튼의 레이블이 가능으로 변경된다.

16.1.4 연결 재조정(Shrink)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 통계 > 연결 재조정

이 기능은 휴지 상태의 JDBC 연결들을 제거함으로서 연결 풀의 크기를 줄여 준다.

1. 노드 트리에서 **JDBC** 폴더를 클릭한다.
2. JDBC 데이터 소스의 목록에서 재조정버튼을 클릭한다.
3. 풀의 크기가 줄어들었는지 확인한다.

16.1.5 XA 트랜잭션 재시작(Resync)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 통계 > 연결 재조정

이 기능은 JDBC 데이터 소스의 XA 트랜잭션이 DB의 장애로 인해 중단되었을 때 재시도하도록 한다.

1. 노드 트리에서 **JDBC** 폴더를 클릭한다.
2. JDBC 데이터 소스의 목록에서 **XA 트랜잭션 재시작** 버튼을 클릭한다.

16.1.6 재설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JDBC 데이터 소스 > 재설정

이 기능은 데이터 소스의 일부 속성을 실행 시간에 변경하는 기능이다. 재설정을 사용한 변경은 *JEUSMain.xml*에 저장되지 않는다.

변경 가능한 속성에는 사용자 이름, 암호 그리고 연결풀의 최대값, 최소값, 증가폭이 있다.

1. 노드 트리에서 재설정하고자 하는 JDBC 데이터 소스를 클릭한 후 재설정 탭을 선택한다. 혹은 컨텍스트 메뉴에서 해당 데이터소스의 재설정을 선택한다.
2. 안쪽의 탭에서 보안 혹은 연결 풀 탭을 선택한다.
3. 변경하고자 하는 속성에 새로운 값을 입력한다.
4. 입력이 완료되면 확인 버튼을 클릭한다.

- 연결 풀을 변경하였다면 JDBC 데이터 소스 모니터링 화면을 통해서 변경된 값이 적용되었는지 확인해 볼 수 있다.

The screenshot shows the 'JDBC 데이터 소스 - 재설정' (JDBC Data Source - Configuration) screen. On the left, there's a sidebar with tabs for '통계' (Statistics), '제설정' (Configuration), and '연결풀' (Connection Pool). The main area has two tabs: '보안' (Security) and '연결풀' (Connection Pool). The '보안' tab is selected. It contains fields for '사용자 이름' (User Name) and '비스워드' (Password), both of which are currently empty. At the bottom right are '확인' (Check) and '취소' (Cancel) buttons.

그림 146 JDBC 데이터 소스의 보안 재설정

The screenshot shows the same configuration screen, but the '연결풀' (Connection Pool) tab is selected. It contains three input fields: '최대' (Max) with value '10', '최소' (Min) with value '2', and '증가 폭' (Increase Step) with value '2'. The layout is identical to the security configuration screen, with '보안' and '연결풀' tabs at the top and '확인' and '취소' buttons at the bottom right.

그림 147 JDBC 데이터 소스의 연결 풀 재설정

16.2 JCA 리소스

16.2.1 연결 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 연결 통계

1. 노드 트리에서 모니터링 하고자 하는 **JCA 리소스**를 클릭한 후 하위 탭에서 연결 탭을 선택한다.
2. JCA 리소스의 연결 통계 정보가 출력된다.

컨넥션 풀 토큰	관리 컨넥션 풀 토큰	평균 대기 시간(밀리초)	평균 사용시간(밀리초)
eis/whitebox-xa	eis/whitebox-xa	0	0

그림 148 JCA 연결의 연결 통계

16.2.2 연결 풀 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 연결 통계

1. 노드 트리에서 모니터링 하고자 하는 **JCA 리소스**를 클릭한 후 하위 탭에서 연결 탭을 선택한다.
2. JCA 리소스의 연결 풀 통계 정보가 출력된다.
3. 필요에 따라 아래쪽의 재설정 버튼을 이용하여 모니터링 정보를 갱신한다.

JCA 리소스 - 연결 풀								
컨넥션 팩토리	관리 컨넥션 팩토리	평균 대기 시간(밀리초)	평균 사용시간(밀리초)	생성	닫힘	가용률	풀크기	대기증만 쓰레드
eis/whitebox-xa	eis/whitebox-xa	0	0	0	0	0	0	0

표 편집

[새로 고침](#)

그림 149 JCA 리소스의 연결 풀 통계

16.2.3 연결 풀의 가능

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 제어 > 가능

이 기능은 불가능 명령에 의해 작동이 정지된 JCA 리소스 연결 풀을 작동 가능하게 하는 명령이다.

1. 노드 트리에서 가능하게 하고자 하는 JCA 리소스를 클릭한 후 하위 탭에서 제어탭을 선택한다.
2. 연결 풀목록에서 가능버튼을 클릭한다.
3. 연결 풀이 가능하게되면 버튼의 레이블이 불가능으로 변경된다.

16.2.4 연결 풀의 불가능

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 제어 > 불가능

이 기능은 현재 작동중인 JCA 리소스 연결 풀을 정지 시키는 기능이다.

1. 노드 트리에서 불가능하게 하고자 하는 JCA 리소스를 클릭한 후 하위 탭에서 제어탭을 선택한다.
2. 연결 풀 목록에서 불가능버튼을 클릭한다.
3. 연결 풀이 불가능하게되면 버튼의 레이블이 가능로 변경된다.

16.2.5 연결 풀의 연결 재조정(Shrink)

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 제어 > 연결 재조정

이 기능은 휴지 상태의 연결들을 제거함으로서 연결 풀의 크기를 줄여준다.

1. 노드 트리에서 연결 재조정을 할 **JCA 리소스**를 클릭한 후 하위 탭에서 **제어**탭을 선택한다.
2. 연결 풀 목록에서 **연결 재조정**버튼을 클릭한다.
3. 풀의 크기가 줄어들었는지 확인한다.

16.2.6 연결 풀의 XA 트랜잭션 재시작

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JCA 리소스 > 제어 > XA 트랜잭션 재시작

이 기능은 연결 풀의 XA 트랜잭션이 장애로 인해 중단되었을 때 재시도하도록 한다.

1. 노드 트리에서 XA 트랜잭션을 재시도할 **JCA 리소스**를 클릭한 후 하위 탭에서 **제어**탭을 선택한다.
2. 연결 풀 목록에서 **XA 트랜잭션 재시작**버튼을 클릭한다.

16.3 WebT 데이터 소스

16.3.1 WebT 데이터 소스 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > WebT > 통계

1. 노드 트리에서 모니터링 하고자 하는 WebT 를 클릭한다.
2. WebT 의 통계 정보가 출력된다.

16.4 트랜잭션 관리자

16.4.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 리소스 > 트랜잭션
> 기본 설정

트랜잭션 관리자의 기본적인 설정을 한다.

1. 노드 트리에서 설정을 변경하고자 하는 **트랜잭션 관리자**를 클릭한 후 설정탭을 선택하거나 트랜잭션 관리자의 컨텍스트 메뉴에서 **트랜잭션 관리자 설정**을 선택한다.
2. 하위 탭에서 **기본 설정**을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 런타임에 바로 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

설정

기본 설정

쓰레드 풀

통계

트랜잭션 - 기본 설정

트랜잭션 매니저(TM)는 Global Transaction을 시작하고 종료한다. 트랜잭션을 종료할 때 TM은 RM(리소스 매니저)과 통신을 하면서 commit인지 rollback인지 결정한다. 이렇게 함으로써 TM은 Global Transaction의 원자성을 보장하게 된다. 그러나 실제 상황에서는 많은 예외적인 상황이 발생하는데, 이에 대한 대응책으로 다양한 타임아웃 메커니즘을 제공한다. 하위 element에서는 TM의 타임아웃 등을 설정한다.

Use Nio TM 사이의 통신을 Nonblocking I/O를 이용할지의 여부를 결정한다.

고급 선택사항

Active Timeout 밀리초
글로벌 트랜잭션이 시작되면 이 시간 안에 commit이 실행되어야 한다. 그렇지 않으면 트랜잭션 매니저가 rollback 시켜버린다.

Prepare Timeout 밀리초
transaction이 commit될 때 Root Coordinator는 이 시간 내에 Sub Coordinator와 리소스 매니저로부터 'prepare' 신호를 받아야 한다. 만약 받지를 못하면 Root Coordinator는 글로벌 트랜잭션을 rollback 시킨다.

Prepared Timeout 밀리초
transaction이 commit되어 Root Coordinator로부터 prepare message를 받으면 Sub Coordinator는 prepare에 대한 응답을 Root Coordinator로 보내고 global decision을 기다린다. Sub Coordinator는 자신의 Root Coordinator로부터 대기해 설정된 시간 안에 global decision을 받아야 한다. 만약 이 시간 내에 받지 못하면, Root Coordinator로 다시 'prepare'에 대한 응답 메시지를 보낸다. 그래도 여전히 시간 내에 global decision이 오지 않는다면, <heuristic-rollback>의 값이 true일 때 Global Transaction을 rollback 시켜버린다. <heuristic-rollback>이 false이면, Root Coordinator로 메시지를 보내고 global decision을 기다리기를 계속 한다.

Commit Timeout 밀리초
Root Coordinator는 Sub Coordinator와 리소스 매니저에게 commit message를 보낸 후 이 시간 이내에 'commit'이나 'rollback'에 대한 결과를 받아야 한다. 만약 결과가 오지 않으면, Root Coordinator는 글로벌 트랜잭션을 'Uncompleted List'에 기록해서, 트랜잭션이 완전히 끝나지 않았음을 남겨둔다.

Recovery Timeout 밀리초
이 값은 트랜잭션 복구 시에 사용된다. 트랜잭션 매니저는 트랜잭션 복구를 위해서 복구될 트랜잭션 정보를 가져오려고 한다. 만약 다른 트랜잭션 매니저에서 이 시간 내에 복구 정보를 알려주지 않으면, <heuristic-rollback>이 true 일 때, 해당 트랜잭션을 rollback 시킨다. 값이 false이면 트랜잭션 복구를 시스템 관리자에게 남겨두고 더 이상 진행하지 않는다.

Uncompleted Timeout 밀리초
트랜잭션 매니저는 전체 트랜잭션 처리를 완료하기 위해, 실패한 글로벌 트랜잭션의 목록을 보관한다. 완료되지 못한 글로벌 트랜잭션의 정보는 복구 처리시에 사용되므로, 이 타임 아웃 시간까지 보관된다. 그러므로 이 시간이 너무 짧으면 복구 정보가 빨리 지워지게 되고, 트랜잭션 매니저가 해당 글로벌 트랜잭션의 무결성을 복구할 수 없게 된다. 그 결과 글로벌 트랜잭션 복구를 위해서, 시스템 관리자가 많은 작업을 직접 처리해야만 한다.

Capacity

이 값을 사용해서 JEUS 트랜잭션 매니저는 내부 구조를 최적화시킨다. 트랜잭션 매니저가 동시에 처리하는 글로벌 트랜잭션의 개수를 고려해서 값을 정한다.

Heuristic Rollback 어떤 이유로 Global Transaction이 완료되지 못하는 경우, 이 값이 true이면 트랜잭션 매니저가 rollback 시켜 버리고, false라면 시스템 관리자가 처리하도록 Uncompleted List에 남겨둔다.

Resolution 밀리초
타임 아웃 메커니즘의 기본 시간 간격. 이 값을 작게 설정하면 타임 아웃이 세밀하게 작동한다. 값이 너무 크면 타임 아웃이 너무 느슨하게 작동한다.

확인 **취소**

그림 150 트랜잭션 관리자의 기본 설정

16.4.2 쓰레드 풀 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 트랜잭션 관리자 > 쓰레드 풀

JEUS 트랜잭션 매니저에서는 다른 트랜잭션 매니저간의 내부 통신을 처리하거나 글로벌 트랜잭션의 커밋처리를 위해서 여러 개의 작업 쓰레드를 사용한다.

1. 노드 트리에서 설정을 변경하고자 하는 **트랜잭션 관리자**를 클릭한 후 설정탭을 선택하거나 트랜잭션 관리자의 컨텍스트 메뉴에서 **트랜잭션 관리자 설정**을 선택한다.
2. 하위 탭에서 **쓰레드 풀**을 선택한다.
3. 각 항목의 값을 입력한다. 최소값은 런타임에 의미가 없으므로 최소값을 제외한 설정된 값은 실행중에 바로 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.



그림 151 트랜잭션 관리자의 쓰레드 풀 설정

16.4.3 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 트랜잭션 관리자 > 통계

1. 노드 트리에서 설정을 변경하고자 하는 **트랜잭션 관리자**를 클릭한 후 통계탭을 선택하거나 트랜잭션 관리자의 컨텍스트 메뉴에서 **트랜잭션 관리자 통계**를 선택한다.

2. 트랜잭션 관리자의 통계정보가 출력된다.

트랜잭션 - 통계	
활성 수	0
커밋 수	4
롤백 수	0
타임마웃 수	0
타임 마웃 롤백 수	0
활성 타임 마웃 수	0
준비(Prepare) 타임 마웃 수	0
준비된(Prepared) 타임 마웃 수	0
커밋 타임 마웃 수	0
휴리스틱 롤백 수	0
평균 수행 시간	19

[새로 고침]

그림 152 트랜잭션 관리자의 통계

16.5JMS 리소스

JMS 리소스는 19_JMS 리소스를 참조하라.

17 엔진 컨테이너 서비스

엔진 컨테이너내에는 스케줄러, JMX 서버, 로그 서비스가 존재한다. JEUS 매니저에도 동일한 서비스가 존재하므로 자세한 설명은 24_JEUS 매니저 서비스을 참조하기 바란다.

18 JDBC 데이터 소스

JEUS 데이터 소스는 어플리케이션에서 JDBC를 이용해 데이터 베이스에 연결하고 할 경우 사용하는 리소스이다. JDBC 데이터 소스의 추가 및 삭제는 JEUS 관리자에서 이루어지며 JEUS 관리자에 바인딩된 객체를 각 엔진 컨테이너에서 Lookup하여 사용한다.

JEUS에서 JDBC 드라이버 구성은 시도할 때

`$JEUS_HOME\lib\datasource` 디렉토리 안에 JDBC 드라이버 클래스 라이브러리가 있는지 먼저 확인해야 한다.

18.1 JDBC 데이터 소스 구성

18.1.1 추가

JEUS 메니저 > JDBC 데이터 소스 > 새 JDBC 데이터 소스 생성

1. 노드 트리에서 **JEUS 리소스**의 **JDBC** 폴더를 클릭한 후 새 **JDBC 데이터 소스 생성** 링크를 클릭하거나 **JDBC** 폴더의 컨텍스트 메뉴에서 새 **JDBC 데이터 소스 생성**을 선택한다.

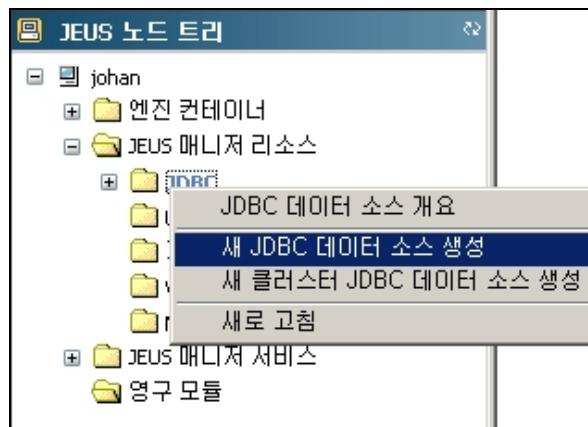


그림 153 새 JDBC 데이터 소스 생성 선택

2. DBMS를 선택한다. DBMS를 선택하면 아래 데이터 소스를 선택하는 필드가 업데이트 된다.

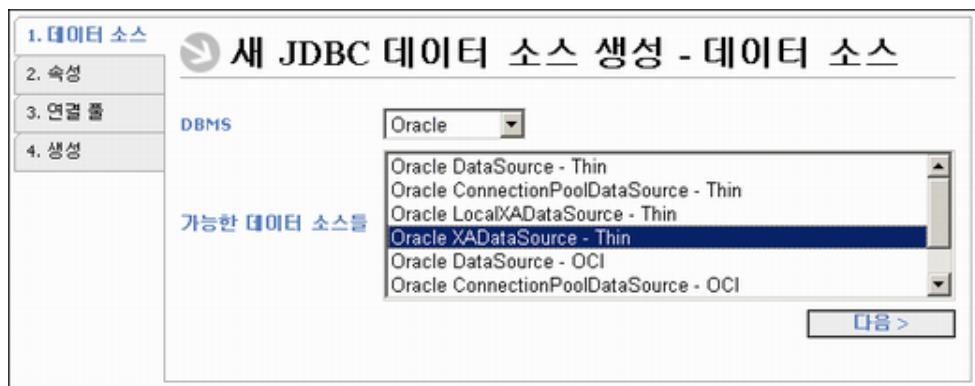


그림 154 JDBC 데이터 소스 생성 - 1 단계 데이터 소스 선택

3. 데이터 소스를 선택한 후 다음을 클릭한다.
4. 데이터 소스 속성들을 입력한다. 추가 속성에는 데이터 소스의 추가적인 속성들을 입력한다. 입력 형식은 “키[:자료형]=값”이다. 자료형이 생략될 경우는 java.lang.String 이 기본 타입이다. 예를 들어 데이터소스 클래스에 setMaxStmtCount(int count) 라는 메소드가 존재하고 10 이라는 값을 입력하고자 한다면, “maxStmtCount:int=10” 으로 입력하면 된다.

새 JDBC 데이터 소스 생성 - 속성

1. 데이터 소스	Vendor	oracle
2. 속성	JDBC 드라이버 밴더의 이름.	
3. 연결 풀	Data Source Class Name	oracle.jdbc.xa.client.OracleXADataSource
4. 생성	JDBC드라이버의 datasource클래스 이름.	
	Data Source Type	XADatasource
	DataSource의 타입.	
	Export Name *	datasource2
	DataSource의 JNDI이름. 이 값은 Naming Server에 datasource를 등록할 때 사용될 것이다.	
	Database Name *	ORCL
	Database의 이름. Oracle은 database의 SID.	
	Port Number *	1521
	Database listener의 포트번호.	
	Server Name *	localhost
	Database가 실행되는 곳의 서버이름.	
	User *	scott
	DB 사용자 ID로 transaction처리들을 위해서는 충분한 system특권을 가지고 있어야 한다.	
	Password *	tiger
	DB 사용자의 password이다. <encryption> 설정에 따라 Base64로 인코딩된 값이 들어올 수 있다.	
	추가 속성들	
	Description	
	DataSource에 대한 설명을 할 수 있는 element이다.	
< 이전		다음 >

그림 155 JDBC 데이터 소스 생성 - 2 단계 속성 설정

5. 연결 풀 설정을 한다.

새 JDBC 데이터 소스 생성 - 연결 풀

쓰레드 풀

고급 선택사항

가능한 연결 대기

그림 156 JDBC 데이터 소스 생성 - 3 단계 연결 풀 설정

- 추가할 데이터소스를 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.

새 JDBC 데이터 소스 생성 - 생성

이 JDBC 데이터 소스를 지금 바인드 하시겠습니까?

< 미전 **생성**

그림 157 JDBC 데이터 소스 생성 - 4 단계 생성

7. 데이터 소스가 추가되면 노드 트리의 **JDBC** 폴더에 JDBC 데이터 소스가 나타난다.

18.1.2 삭제

JEUS 메뉴지 > JDBC 데이터 소스 > 삭제

1. 노드 트리의 **JEUS 리소스**의 **JDBC** 폴더를 클릭한다.
2. JDBC 데이터 소스 목록에서 삭제하고자 하는 데이터 소스를 선택한 후  아이콘을 클릭한다.
3. JDBC 데이터 소스 목록에서 해당 JDBC 데이터 소스가 사라지며 바인딩 되지 않은 데이터 소스라면 노드 트리에서 사라진다.



노출 이름	벤더	클래스 이름	데이터 소스 종류	명령
datasource1	oracle	oracle.jdbc.xa.client.OracleXADataSource	XADatasource	
datasource2	oracle	oracle.jdbc.xa.client.OracleXADataSource	XADatasource	

 새 JDBC 데이터 소스 생성

 삭제

노출 이름	컨넥션 준비	데이터 소스 수	명령
-------	--------	----------	----

 새 JDBC 클러스터 데이터 소스 생성

그림 158 JDBC 데이터 소스 삭제

참고 : JDBC 데이터 소스가 이미 바인딩 되어있다면 삭제를 하더라도 바인딩 객체는 사라지지 않으므로 주의해야한다.

18.2 JDBC 데이터 소스 제어

18.2.1 JDBC 데이터 소스 바인드

JEUS 매니저 > JDBC 데이터 소스 > 바인드

JDBC 데이터 소스 바인드는 네이밍 서버에 데이터 소스 객체를 바인딩하여 어플리케이션이 Lookup 하여 사용할 수 있게하는 기능이다.

1. 노드 트리의 **JEUS 리소스**의 **JDBC** 폴더를 클릭한다.
2. JDBC 데이터 소스 목록에서 바인딩하고자 하는 데이터 소스를 선택한 후 **바인드** 버튼을 클릭한다.
3. JDBC 데이터 소스 목록에서 해당 JDBC 데이터 소스의 **바인드** 버튼이 비활성화 되며 노드 트리에는 활성화된 아이콘()이 나타난다.



그림 159 JDBC 데이터 소스 바인드

18.3 클러스터 JDBC 데이터 소스 구성

클러스터 JDBC 데이터 소스는 여러 데이터 소스를 하나의 데이터 소스로 사용하여 어떤 데이터 소스에 장애가 발생할 시 다른 데이터 소스를 사용하여 안정된 서비스를 할 수 있도록 하는 데이터 소스이다.

이 데이터 소스를 사용하기 위해서는 적어도 2개 이상의 데이터 소스가 준비해야 한다.

18.3.1 추가

JEUS 매니저 > JDBC 데이터 소스 > 새 클러스터 JDBC 데이터 소스 생성

1. 노드 트리의 **JEUS 리소스**의 **JDBC** 폴더를 클릭한 후 **새 클러스터 JDBC 데이터 소스** 링크를 클릭하거나 **JDBC** 폴더의 컨텍스트 메뉴에서 **새 클러스터 JDBC 데이터 소스**를 선택한다.

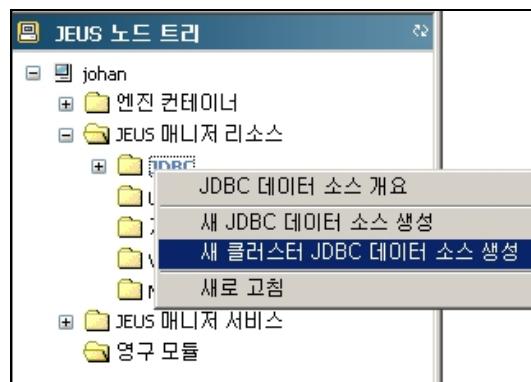


그림 160 클러스터 JDBC 데이터 소스 생성 선택

2. 기본 설정을 입력한다.



그림 161 클러스터 JDBC 데이터 소스 생성-1 단계 기본 설정

3. 클러스터링 할 데이터소스들을 선택한다. 선택된 데이터 소스가 라운드 로빈 방식으로 사용되므로 순서를 고려하여 선택해야한다.

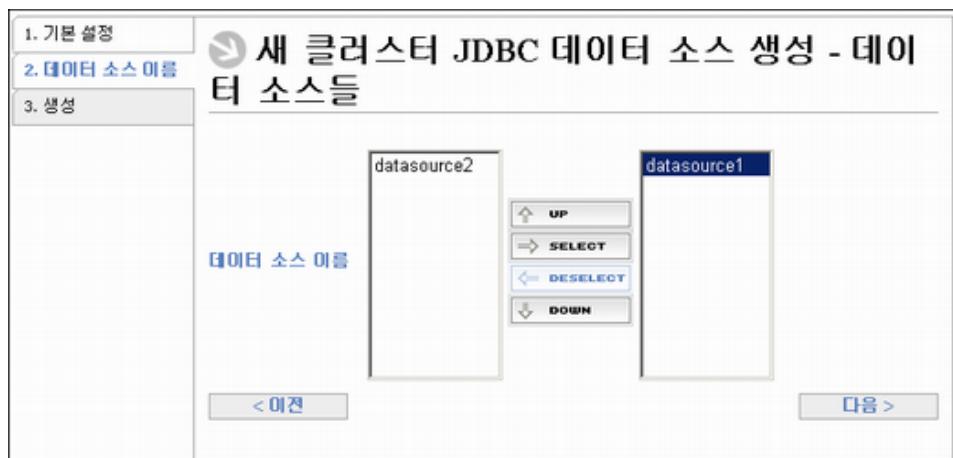


그림 162 클러스터 JDBC 데이터 소스 생성 - 2 단계 데이터 소스 선택

- 추가할 데이터소스를 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.

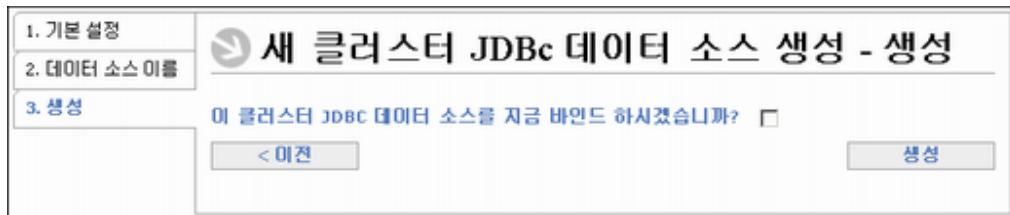


그림 163 클러스터 JDBC 데이터 소스 생성 - 3 단계 데이터 소스 생성

- 데이터 소스가 추가되면 노드 트리의 JDBC 폴더에 클러스터 JDBC 데이터 소스가 나타난다.

18.3.2 클러스터 JDBC 데이터 소스 삭제

JEUS 메뉴 > JDBC 데이터 소스 > 클러스터 JDBC 데이터 소스 삭제

- 노드 트리의 JEUS 리소스의 JDBC 폴더를 클릭한다.
- 클러스터 JDBC 데이터 소스 목록에서 삭제하고자 하는 데이터 소스를 선택한 후 아이콘을 클릭한다.
- 클러스터 JDBC 데이터 소스 목록에서 해당 클러스터 JDBC 데이터 소스가 사라지며 바인딩 되지 않은 데이터 소스라면 노드 트리에서 사라진다.

클러스터 JDBC 데이터 소스				
노출 이름	컨넥션 준비	데이터 소스 수	명령	
datasource3	false	1		

그림 164 클러스터 JDBC 데이터 소스의 삭제

참고 : 클러스터 JDBC 데이터 소스가 이미 바인딩 되어있다면 삭제를 하더라도 바인딩 객체는 사라지지 않으므로 주의해야한다.

18.4 클러스터 JDBC 데이터 소스 제어

18.4.1 클러스터 JDBC 데이터 소스 바인드

클러스터 JDBC 데이터 소스 바인드는 네이밍 서버에 데이터 소스 객체를 바인딩하여 어플리케이션이 Lookup하여 사용할 수 있게 하는 기능이다.

1. 노드 트리에서 **JEUS 리소스**의 **JDBC 폴더**를 클릭한다.
2. 클러스터 JDBC 데이터 소스 목록에서 바인딩하고자 하는 데이터 소스를 선택한 후 **바인드** 버튼을 클릭한다.
3. 클러스터 JDBC 데이터 소스 목록에서 해당 클러스터 JDBC 데이터 소스의 **바인드** 버튼이 비활성화 되며 노드트리에는 활성화된 아이콘()이 나타난다.

19 JMS 리소스

JMS 리소스에는 컨넥션 팩토리, 토픽/큐 테스터 네이션, 클라이언트가 있다.

참고 : JMS 리소스를 사용하기 위해서는 JMS 엔진이 엔진 컨테이너에 포함되어야 한다.

19.1 JMS 리소스 구성

컨넥션 팩토리는 클라이언트 어플리케이션 사용을 편리하게 하기 위해서 JEUS JMS 서버에서 연결을 만들고 저장하는데 사용이 된다.

테스터네이션은 JMS 클라이언트 어플리케이션들이 메시지들을 보내고 받는 장소이다.

19.1.1 컨넥션 팩토리 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 새 컨넥션 팩토리 생성

1. 노드 트리에서 컨넥션 팩토리 폴더를 클릭한 후 새 컨넥션 팩토리 생성 링크를 클릭하거나 컨텍스트 메뉴에서 새 컨넥션 팩토리 생성을 선택한다.

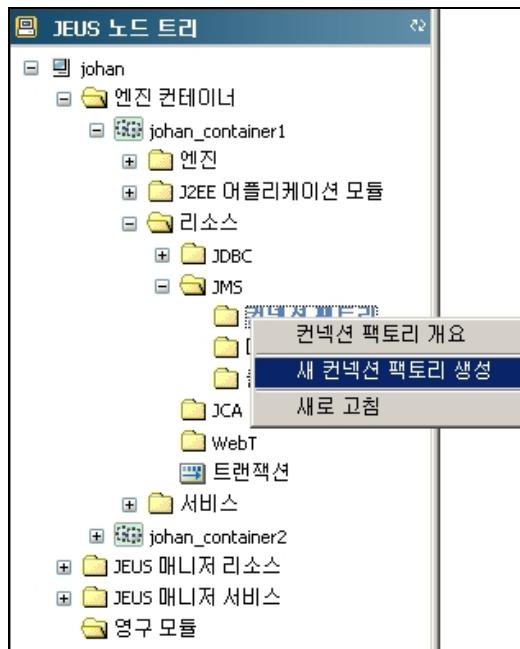


그림 165 새 커넥션 팩토리 생성 선택

- 기본 설정을 입력한다. 여기서는 커넥션 팩토리 타입을 설정한다. 타입에는 총 6 가지(nonxa, xa, queue, topic, xaqueue, xatopic)가 있다.

The dialog box is titled '새 커넥션 팩토리 생성 - 기본 설정'. It has three tabs: 1. 기본 설정 (selected), 2. 쓰래드 풀, and 3. 생성.

Type *: nonxa
해당 Connection Factory 의 종류를 설정한다.

Name *:
JMS 시스템 내에서 관리의 목적으로 사용되는 Connection Factory 의 이름이다.

고급 선택사항

Service:
해당 Connection Factory 가 연결할 service 의 이름을 설정한다. service-config 에 지정된 이름을 사용하도록 한다. 이름을 지정하지 않을 경우 처음 설정된 서비스로 임의 지정된다.

Export Name:
해당 Connection Factory 가 네이밍 서버에 binding 되는 이름. 설정하지 않으면 name 속성이 그대로 사용된다.

Fixed Client Id
connection ID 의 생성 정책이다. true 일 경우 지정된 client-id 를 그대로 사용하며 false 인 경우 JMS 메시지 브로커에 의해 자동적으로 생성된다. 생성된 connection ID 는 getClientID() API 를 이용하여 확인할 수 있다.

Client Id:
해당 Connection Factory 를 이용하여 생성되는 connection 에 기본값으로 설정되는 ClientID 값이다.

다음 >

그림 166 새 커넥션 팩토리 생성 - 1 단계 기본 설정

3. 쓰래드 풀 설정을 한다.

새 커넥션 팩토리 생성 - 쓰래드 풀

Max * 300
thread pool 의 최대 크기를 지정한다.

고급 선택사항

Min
thread pool 의 최소 크기를 지정한다.

Keep Alive Time 300000
min 갯수를 초과하는 thread 에 대해서 여기에 지정된 시간동안 사용되지 않은 thread 는 소멸하게 된다.

< 이전 무시 다음 >

그림 167 새 커넥션 팩토리 생성 -2 단계 쓰래드 풀 설정

4. 추가할 커넥션 팩토리를 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭 한다.

새 커넥션 팩토리 생성 - 생성

이 커넥션 팩토리를 지금 바인드 하시겠습니까?

< 이전 생성

그림 168 새 커넥션 팩토리 생성 -3 단계 생성

5. 커넥션 팩토리가 추가되면 노드 트리의 커넥션 팩토리 폴더에 커넥션 팩토리의 아이콘(⌚)이 나타난다..

19.1.2 토픽 데스터네이션 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 새 토픽 데스터네이션 생성

- 노드 트리에서 데스터네이션 폴더를 클릭한 후 새 토픽 데스터네이션 생성링크를 클릭하거나 컨텍스트 메뉴에서 새 토픽 데스터네이션 생성을 선택한다.
- 기본 설정을 입력한다.

새 Topic 데스터네이션 생성 - 기본 설정

Name * JMS 메시지 브로커 내에서 관리 목적으로 사용되는 데스터네이션의 이름이다.

고급 선택사항

Export Name 해당 데스터네이션이 네이밍 서버에 binding 되는 이름. 설정하지 않으면 name 속성이 그대로 사용된다.

Topic Cluster Name topic-cluster 로 지정된 클러스터링 타입을 선택한다.

Local Distribute round-robin multiple-receiver로 설정된 queue 데스터네이션의 메시지 분산 방식을 결정한다. round-robin, random 중에 하나를 지정할 수 있다.

Limit 해당 destination에서 사용할수 있는 최대 메모리 크기를 설정한다. 사용중인 메모리가 이 값을 초과하는 경우 클라이언트의 메시지 전달은 바로 예외 처리되게 된다.

High Mark 플로우 컨트롤을 사용하기 시작하는 메모리 크기를 설정한다.

Low Mark 소프트 캐싱을 시작하는 메모리 크기를 설정한다.

[다음 >](#)

그림 169 새 토픽 데스터네이션 생성 - 1 단계 기본 설정

- 추가할 토픽 데스터네이션 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.

새 Topic 데스터네이션 생성 - 생성

이 데스터네이션을 지금 바인드 하시겠습니까?

[< 이전](#)

[생성](#)

그림 170 토픽 데스터네이션 생성 - 2 단계 생성

- 토픽 데스터네이션이 추가되면 노드 트리의 데스터네이션폴더에 토픽 데스터네이션 아이콘(☞) 또는 (☞)이 나타난다.

19.1.3 큐 데스터네이션 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 새 큐 데스터네이션 생성

1. 노드 트리에서 데스터네이션 폴더를 클릭한 후 새 큐 데스터네이션 생성 링크를 클릭하거나 컨텍스트 메뉴에서 새 큐 데스터네이션 생성을 선택 한다.
2. 기본 설정을 입력 한다.

새 Queue 데스터네이션 생성 - 기본 설정

Name * JMS 메시지 브로커 내에서 관리 목적으로 사용되는 데스티네이션의 이름이다.

고급 선택사항

Export Name 해당 데스티네이션이 네이밍 서버에 binding 되는 이름. 설정하지 않으면 name 속성이 그대로 사용된다.

Multiple Receiver queue 탑의 데스티네이션에 대해 다중의 receiver 를 협용하여 분산처리를 할 것인지를 설정한다.

Queue Cluster Name queue-cluster 로 지정된 클러스터링 탑을 선택한다.

Local Distribute round-robin multiple-receiver 로 설정된 queue 데스티네이션의 메시지 분산 방식을 결정한다. round-robin, random 중에 하나를 지정할 수 있다.

Limit 해당 destination 에서 사용할수 있는 최대 메모리 크기를 설정한다. 사용중인 메모리가 이 값을 초과하는 경우 클라이언트의 메시지 전달은 바로 어려 처리되게 된다.

High Mark 풀로우 컨트롤을 사용하기 시작하는 메모리 크기를 설정한다.

Low Mark 소프트 캐싱을 시작하는 메모리 크기를 설정한다.

다음 >

그림 171 새 큐 데스터네이션 생성 -1 단계 기본 설정

3. 추가할 큐 데스터네이션의 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.

새 Queue 데스터네이션 생성 - 생성

이 데스터네이션을 지금 바인드 하시겠습니까?

< 이전 **생성**

그림 172 큐 데스터네이션 생성 - 2 단계 생성

19.2 JMS 리소스 설정

19.2.1 컨넥션 팩토리 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 설정 >
컨넥션 팩토리 기본 설정

1. 노드 트리에서 변경하고자 하는 컨넥션 팩토리를 클릭 한다.
2. 하위 템에서 기본 설정탭을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭 한다.

19.2.2 컨넥션 팩토리 쓰레드 풀 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 설정 >
컨넥션 팩토리 쓰레드 풀

1. 노드 트리에서 변경하고자 하는 컨넥션 팩토리를 클릭 한다.
2. 하위 템에서 쓰레드 풀탭을 선택한다.
3. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인 버튼을 클릭 한다.

19.2.3 데스터네이션 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 설정 >
데스터네이션 기본 설정

1. 노드 트리에서 변경하고자 하는 데스터네이션을 클릭 한다.
2. 각 항목의 값을 입력한다. 설정된 값은 다음 부트시에 적용된다.
3. 입력이 완료되면 확인 버튼을 클릭 한다.

19.3 JMS 리소스 제어

19.3.1 컨넥션팩토리 바인드

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 제어 > 컨넥션팩토리 바인드

1. 노드 트리에서 JMS 의 컨넥션 팩토리폴더를 클릭한다.
2. 컨넥션 팩토리 목록에서 바인딩하고자 하는 컨넥션 팩토리를 선택한 후 바인드 버튼을 클릭한다.
3. 컨넥션 팩토리목록에서 해당 컨넥션 팩토리의 바인드버튼이 비활성화 되며 노드트리에는 활성화된 아이콘()이 나타난다.



그림 173 컨넥션 팩토리 바인드

19.3.2 데스터 네이션 바인드

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 제어 > 데스터네이션 바인드

1. 노드 트리에서 JMS 의 데스터네이션 폴더를 클릭한다.
2. 데스터네이션 목록에서 바인딩하고자 하는 데스터네이션을 선택한 후 바인드 버튼을 클릭한다.
3. 데스터네이션 목록에서 해당 데스터네이션의 바인드 버튼이 비활성화 되며 노드트리에는 활성화된 아이콘()이 나타난다.

19.3.3 듀러블 서브스크라이버(Durable Subscriber)에서 메시지 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 토픽 테스터 네이션 > 듀러블 서브스크라이버 > 메시지 삭제

듀러블 서브스크라이버 메시지의 경우 각각의 듀러블 서브스크라이버의 메시지를 정보(생성시간, 전송여부, 태입)에 따라 관리자가 삭제할 수가 있다.

1. 노드 트리에서 메시지를 삭제하고자 하는 듀러블 서브스크라이버를 가지는 토픽 테스터 네이션을 클릭한다.
2. 하위 탭에서 듀러블 서브스크라이버를 선택한다.
3. 듀러블 서브스크라이버 목록에서 메시지를 삭제하고자 하는 듀러블 서브스크라이버를 선택한다.
4. 메시지 탭을 클릭한다.
5. 메시지 정보를 보고 삭제할 메시지를 선택한다.
6. 삭제 버튼을 클릭한다.

19.4 JMS 리소스 통계

19.4.1 듀러블 서브스크라이버 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 토픽 테스터 네이션 > 듀러블 서브스크라이버 > 통계

듀러블 서브스크립션은 클라이언트에 메시지를 발송하는 것을 보장하는 토픽 테스터 네이션에서 사용된다. 듀러블 서브스크립션은 수신자가 연결을 종결할 때, 비활성화된 것으로 간주하여 클라이언트가 비활성화된 동안 메시지를 저장해 두었다가, 활성화 상태일 때 듀러블 서브스크라이버에 메시지를 발송 한다.

듀러블 서브스크라이버는 토픽 테스터 네이션일 경우만 사용된다.

1. 노드 트리에서 모니터링 하고자 하는 토픽 테스터 네이션을 클릭한다.
2. 듀러블 서브스크라이버 탭을 선택한다.

-
3. 통계 정보가 출력된다.

19.4.2 소비자(Cusumer) 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 데스터네이션 > 소비자 > 통계

데스터네이션으로부터 메시지를 수신하는 클라이언트에 대한 정보를 보여 준다.

1. 노드 트리에서 모니터링 하고자 하는 데스터네이션을 클릭한다.
2. 소비자탭을 클릭한다.
3. 통계 정보가 출력된다.

19.4.3 클라이언트 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 클라이언트 > 통계

현재 JMS 엔진에 등록된 모든 클라이언트에 대한 정보를 볼 수 있다.

1. 노드 트리에서 JMS 밑의 **Client** 폴더를 클릭한다.
2. 통계 정보가 출력된다.

19.4.4 클라이언트 연결 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 클라이언트 > 연결 > 통계

클라이언트에서 사용하는 연결에 관한 정보를 볼 수 있다.

1. 노드 트리에서 JMS 밑의 **클라이언트** 폴더를 클릭한다.
2. 클라이언트 목록에서 클라이언트 이름을 클릭한다.
3. 연결탭을 선택한다.
4. 통계 정보가 출력된다.

19.4.5 클라이언트 연결 세션 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > JMS 리소스 > 클라이언트 > 연결 > 세션 > 통계

연결에서 사용하고 있는 세션정보를 볼수 있다.

1. 노드 트리에서 JMS 밑의 클라이언트폴더를 클릭한다.
2. 클라이언트목록에서 클라이언트 이름을 클릭한다.
3. 세션 탭을 선택한다.
4. 세션목록에서 통계를 보고자하는 세션을 클릭한다.
5. 통계 정보가 출력된다.

20 URL 리소스

URL 리소스는 URL 을 JNDI 에 등록하는데 사용된다. 이렇게 하면 클라이언트 어플리케이션에서 URL 을 간접적으로 접근할 수 있다.

20.1 URL 구성

20.1.1 추가

JEUS 매니저 리소스 > URL > 새 URL 생성

1. 노드 트리에서 **URL** 폴더를 클릭한다.
2. 노출이름과 URL에 값을 입력한다.
3. 입력이 완료되면 **추가**버튼을 클릭한다.
4. URL 리소스가 생성되면 노드 트리의 URL 폴더 밑에 아이콘()이 생성되고 URL 목록에 추가된다.

노출 이름	URL	
tmaxsoft	http://www.tmax.co.kr	

Export Name *
JNDI name은 Naming Server에 URL을 bind할 때 사용된다.

Url *
URL은 bind된 JNDI Server의 JNDI name에 매팅된다.

[추가]

그림 174 URL 리소스 추가

20.1.2 삭제

JEUS 매니저 리소스 > URL > 삭제

1. 노드 트리에서 **URL** 폴더를 클릭한다.
2. 삭제하고자 하는 URL을 선택하고  아이콘을 클릭한다.
3. URL 리소스가 삭제되면 URL 목록에서 삭제된다.

20.2 URL 설정

20.2.1 URL 설정

JEUS 매니저 리소스 > URL > 설정

1. 노드 트리에서 **URL** 폴더를 클릭한다.
2. 변경하고자 하는 URL의 링크를 클릭한다.
3. **URL**을 변경한다.
4. 입력이 완료되면 확인버튼을 클릭한다.

21 자바 메일

자바 메일은 SMTP 와 같은 메일 프로토콜을 지원하는 것을 사용하여 클라이언트 어플리케이션으로부터 이메일을 보내는데 사용한다.

21.1 자바 메일 구성

21.1.1 자바 메일 추가

JEUS 매니저 > 자바 메일 > 새 자바메일 생성

1. 노드 트리에서 자바메일폴더를 클릭 한다.
2. 노출 이름과 속성들을 입력한다. 속성은 “키=값” 형식으로 입력 한다.
3. 입력이 완료되면 추가버튼을 클릭 한다.
4. 자바 메일 리소스가 생성되면 노드 트리의 자바메일 폴더 밑에 아이콘이 생성되고 자바메일 목록에 추가된다.

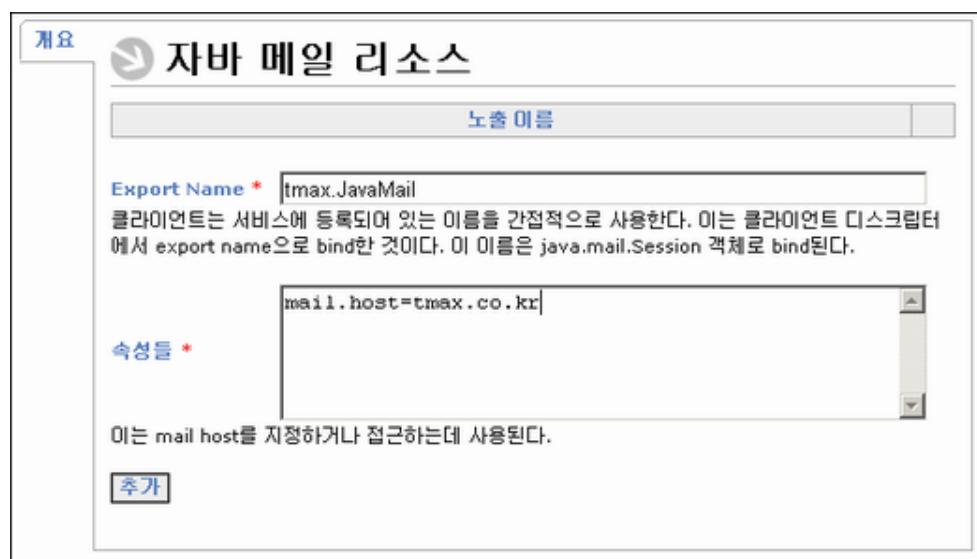


그림 175 자바메일 리소스

21.1.2 자바 메일 삭제

JEUS 관리자 > 자바 메일 > 삭제

1. 노드 트리에서 자바 메일폴더를 클릭한다.
2. 삭제하고자 하는 자바 메일을 선택하고  아이콘을 클릭한다.
3. 자바메일 리소스가 삭제되면 자바 메일목록에서 삭제된다.

21.2 자바 메일 설정

21.2.1 자바 메일 설정

JEUS 관리자 > 자바 메일 > 설정

1. 노드 트리에서 자바 메일폴더를 클릭한다.
2. 변경하고자 하는 자바 메일의 링크를 클릭한다.
3. 속성을 변경한다.
4. 입력이 완료되면 확인버튼을 클릭한다.

22 WebT 데이터 소스

WebT는 “Web Transaction”의 약자로 C/S 환경의 미들웨어 제품인 Tmax 와 웹 환경의 WAS(Web Application Server)제품인 JEUS 간의 트랜잭션 서비스를 지원하는 제품이다.

WebT 데이터 소스는 JEUS 내의 어플리케이션에서 WebT를 이용해 Tmax 서버와 연결하여 할 경우 사용하는 리소스이다. WebT 데이터 소스의 추가 및 삭제는 JEUS 관리자에서 이루어지며 JEUS 관리자에 바인딩된 객체를 각 엔진 컨테이너에서 Lookup 하여 사용한다.

JEUS에서 WebT 데이터 소스를 사용하고자 한다면
\$JEUS_HOME\$/lib/system 디렉토리안에 WebT 라이브러리 (webt.jar)가 있는지 먼저 확인해야한다.

22.1 WebT 데이터 소스 구성

22.1.1 개요

JEUS 매니저 > WebT > WebT 데이터 소스 개요

이 화면에는 현재 JEUS 매니저에 등록된 WebT 데이터 소스의 목록이 출력된다. 또한 클러스터 WebT 데이터 소스가 있다면 클러스터 WebT 데이터 소스 목록에 설정된 클러스터 목록이 나타난다. 각 목록에서 해당 WebT 데이터 소스 이름을 클릭하면 해당 설정을 볼 수 있으며 명령에서 바인드를 클릭하면 WebT 데이터 소스가 JNDI에 등록된다.

JNDI에 대한 자세한 내용은 JEUS Server 안내서를 참고하기 바란다

22.1.2 추가

JEUS 매니저 > WebT > 새 WebT 데이터 소스 생성

1. 노드 트리에서 JEUS 리소스의 WebT 폴더를 클릭한 후 새 WebT 데이터 소스 생성 링크를 클릭하거나 WebT 폴더의 컨텍스트 메뉴에서 새 WebT 데이터 소스 생성을 선택한다.

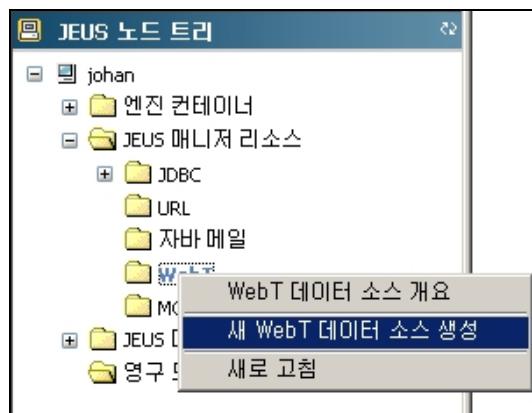


그림 176 새 WebT 데이터 소스 생성 선택

- 기본 설정을 입력한다. 이 설정에서는 Tmax 서버와의 연결 정보와 기본적인 설정을 입력한다. 고급 선택 사항의 Tmax Property에는 “키=값”의 형식으로 입력하면 된다. 키값에 해당하는 값은 WebT 안내서를 참고 하기 바란다.

새 WebT 데이터 소스 생성 - 기본 설정

1. 기본 설정
2. 연결 품
3. 로깅
4. 기타
5. 생성

Export Name * webtdatasource1
JNDI에 등록되어 서비스 되는 이름이다.

Host Name * localhost
Tmax server의 이름이나 IP 주소를 말한다.

Port * 8888
Tmax server환경 파일에 설정된 "PORT" 값과 동일한 값을 설정해 주어야 한다.

고급 선택사항

다음 >

그림 177 새 WebT 데이터 소스 생성 - 기본 설정

- 다음 단계의 풀링에서는 WebT 연결에 대한 풀링을 설정한다.

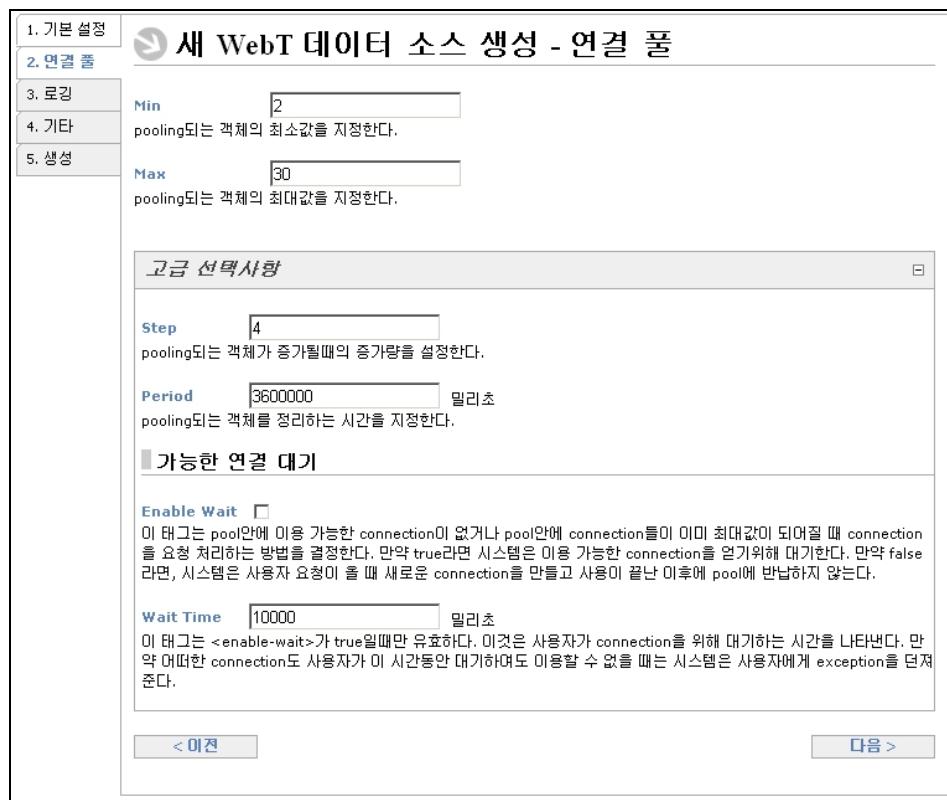


그림 178 새 WebT 데이터 소스 생성 - 연결 풀

- 다음 단계의 로깅 설정에서는 WebT 와 Tmax 간의 통신중 발생하는 로그를 저장하는 설정을 한다.

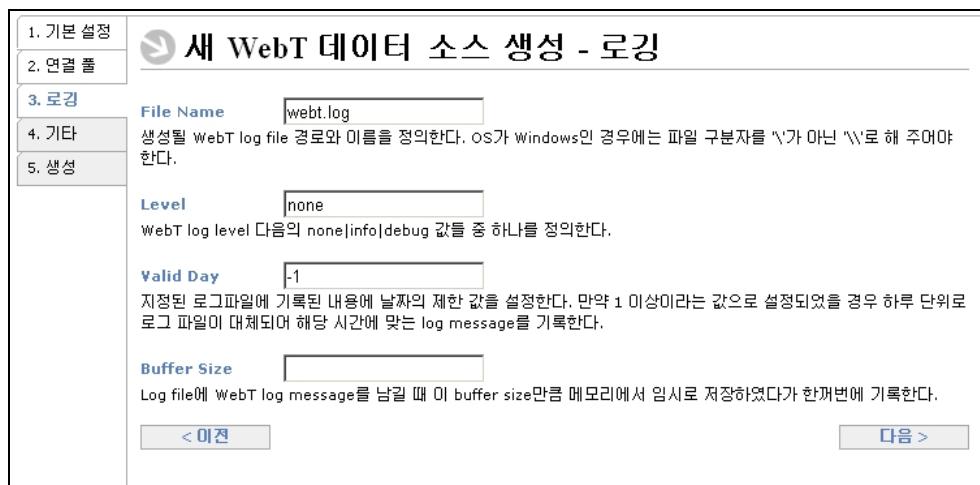


그림 179 새 WebT 데이터 소스 생성 - 로깅

- 마지막 단계인 기타 설정에서는 보안에 관한 설정을 한다.

새 WebT 데이터 소스 생성 - 기타

보안

User Name Tmax server로의 연결시 Tmax server에 등록된 user에 대한 인증을 위해 user name을 설정한다.

User Password Tmax server로의 연결시 Tmax server에 등록된 user에 대한 인증을 위해 user password를 설정한다.

Domain Name Tmax server로의 연결시 Tmax server에 등록된 domain에 대한 인증을 위해 domain name을 설정한다.

Domain Password Tmax server로의 연결시 Tmax server에 등록된 domain에 대한 인증을 위해 domain password를 설정한다.

< 이전

다음 >

그림 180 새 WebT 데이터 소스 생성 - 기타(보안)

- 모든 설정을 완료하였으면 생성을 한다. 생성 시 JNDI에 바인드 하고자 한다면 바인드 선택사항을 선택하고 생성 버튼을 눌러야 한다.

새 WebT 데이터 소스 생성 - 생성

WebT 데이터 소스를 지금 바인드 하시겠습니까?

< 이전

생성

생성

그림 181 새 WebT 데이터 소스 생성 - 생성

- WebT 데이터 소스가 생성되면 노드 트리에 WebT 데이터 소스 아이콘 (WEB)이 추가되며 WebT 데이터 소스 개요 화면의 WebT 데이터 소스 목록에 새로 생성된 WebT 데이터 소스가 추가된다(그림 186 참조).

22.1.3 삭제

JEUS 메뉴 > WebT > WebT 데이터 소스 개요 > 삭제

- 노드 트리에서 JEUS 리소스의 WebT 폴더를 클릭하면 WebT 데이터소스 개요 화면이 나타난다.
- 화면의 WebT 데이터 소스 목록에서 삭제하고자 하는 WebT 데이터 소스의 ■ 아이콘을 클릭한다.

WebT 데이터 소스				
노출 이름	호스트 이름	포트	명령	
webtdatasource1	localhost	9000	바인드	
webtdatasource2	localhost	9001	바인드	
새 WebT 데이터 소스 생성				

그림 182 WebT 데이터 소스 개요 - 데이터 소스 목록

3. WebT 데이터 소스가 목록에서 사라지며 노드트리에서도 사라진다.

참고 : WebT 데이터 소스가 목록에서 사라진다 해도 실제 JNDI에서 사라진 것은 아니다. 따라서 삭제를 완전하게 하기 위해서는 JEUS 매니저를 재시작해야 한다.

22.2 WebT 데이터 소스 설정

22.2.1 기본 설정

JEUS 매니저 > WebT > WebT 데이터 소스 개요 > 설정 > 기본 설정

1. 노드 트리의 **WebT** 폴더에서 수정하고자 하는 WebT 데이터 소스를 클릭하거나 **WebT 데이터 소스 개요** 화면에서 WebT 데이터 소스를 클릭한다.
2. 탭에서 **기본 설정** 탭을 선택한다.
3. 속성을 변경한다. 변경된 사항은 다음 부트 혹은 바인드 시에 적용된다.
4. **확인** 버튼을 클릭한다.

22.2.2 연결 풀 설정

JEUS 매니저 > WebT > WebT 데이터 소스 개요 > 설정 > 연결 풀

1. 노드 트리의 **WebT** 폴더에서 수정하고자 하는 WebT 데이터 소스를 클릭하거나 **WebT 데이터 소스 개요** 화면에서 WebT 데이터 소스를 클릭한다.
2. 탭에서 **연결 풀** 탭을 선택한다.
3. 속성을 변경한다. 변경된 사항은 다음 부트 혹은 바인드 시에 적용된다.

4. 확인 버튼을 클릭한다.

22.2.3 로깅 설정

JEUS 매니저 > WebT > WebT 데이터 소스 개요 > 설정 > 로깅

1. 노드 트리의 **WebT** 폴더에서 수정하고자 하는 WebT 데이터 소스를 클릭하거나 **WebT 데이터 소스 개요** 화면에서 WebT 데이터 소스를 클릭한다.
2. 탭에서 **로깅 탭**을 선택한다.
3. 속성을 변경한다. 변경된 사항은 다음 부트 혹은 바인드 시에 적용된다.
4. 확인 버튼을 클릭한다.

22.2.4 보안 설정

JEUS 매니저 > WebT > WebT 데이터 소스 개요 > 설정 > 기타 > 보안

1. 노드 트리의 **WebT** 폴더에서 수정하고자 하는 WebT 데이터 소스를 클릭하거나 **WebT 데이터 소스 개요** 화면에서 WebT 데이터 소스를 클릭한다.
2. 탭에서 **기타 탭**을 선택한다.
3. 보안관련된 속성을 변경한다. 변경된 사항은 다음 부트 혹은 바인드 시에 적용된다.
4. 확인 버튼을 클릭한다.

22.3 WebT 데이터 소스 제어

22.3.1 WebT 데이터 소스 바인드

JEUS 매니저 > WebT > WebT 데이터 소스 개요 > 바인드

1. WebT 데이터 소스 개요 화면에서 바인드 하고자 하는 WebT 데이터 소스의 **바인드** 버튼을 클릭한다.
2. 바인드가 완료되면 노드트리의 WebT 데이터 소스 아이콘()이 활성화되고 **바인드** 버튼이 비활성화 된다.

22.4 클러스터 WebT 데이터 소스 구성

22.4.1 추가

JEUS 화면 > WebT > 새 WebT 클러스터 데이터 소스 생성

1. 노드 트리의 **JEUS 리소스**의 **WebT** 폴더를 클릭한 후 새 클러스터 **WebT 데이터 소스 링크**를 클릭한다.
2. 기본설정을 입력한다.



그림 183 새 클러스터 WebT 데이터 소스 생성 - 기본 설정

3. 클러스터링 할 데이터소스들을 선택한다. 선택된 데이터 소스가 라운드 로빈 방식으로 사용되므로 순서를 고려하여 선택해야한다.

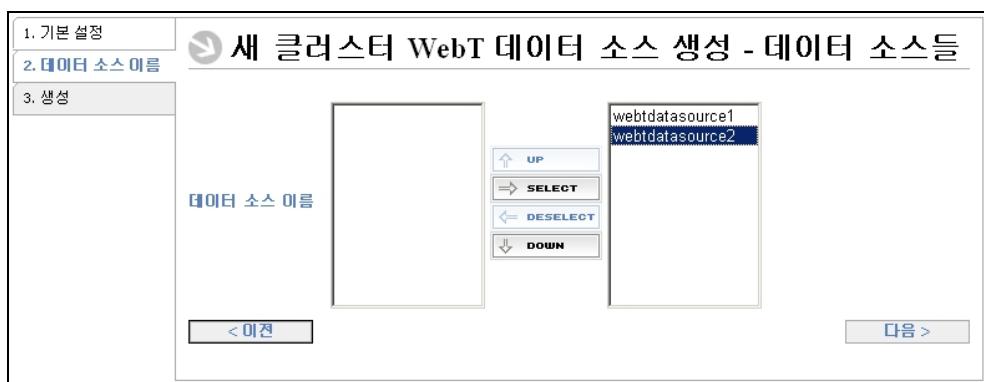


그림 184 새 클러스터 WebT 데이터 소스 생성 - 데이터 소스들

4. 추가할 데이터소스를 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.



그림 185 새 클러스터 WebT 데이터 소스 생성 - 생성

- 데이터 소스가 추가되면 노드 트리의 WebT 폴더에 클러스터 WebT 데이터 소스가 나타난다.



그림 186 노드 트리에서의 WebT 데이터 소스

참고 : 클러스터 WebT 데이터 소스는 클러스터 JDBC 데이터 소스와는 달리 하나의 클러스터 데이터 소스만을 생성할 수 있다. 따라서 클러스터 데이터 소스가 하나 생성되었다면, 새 WebT 클러스터 데이터 소스 생성 링크는 비활성화 된다.

22.4.2 삭제

JEUS 매니저 > WebT > 새 WebT 클러스터 데이터 소스 삭제

- 노드 트리의 JEUS 리소스의 WebT 폴더를 클릭한다.
- 클러스터 WebT 데이터 소스 목록에서 삭제하고자 하는 데이터 소스를 선택한 후 아이콘을 클릭한다.



그림 187 클러스터 WebT 데이터 소스 개요

- 클러스터 WebT 데이터 소스 목록에서 해당 클러스터 WebT 데이터 소스가 사라지며 바인딩 되지 않은 데이터 소스라면 노드 트리에서 사라진다.

참고 : 클러스터 WebT 데이터 소스가 이미 바인딩 되어있다면 삭제를 하더라도 바인딩 객체는 사라지지 않으므로 주의해야한다.

22.5 클러스터 WebT 데이터 소스 제어

22.5.1 클러스터 WebT 데이터 소스 바인드

- 클러스터 WebT 데이터 소스 목록에 설정된 클러스터 WebT 데이터 소스의 바인드버튼을 클릭한다.
- 바인드가 완료되면 노드트리의 WebT 데이터 소스 아이콘()이 활성화되고 바인드 버튼이 비활성화 된다.

23 MQ 리소스

앞서 설명한 몇 가지 리소스 외 JEUS 와 연결 할 수 있는 비정규화된 리소스로 MQ 리소스가 있다. JEUS 에서 설정할 수 있는 MQ 리소스로는 IBM MQ 와 SonicMQ 가 있다. IBM MQ 는 IBM 에서 Message Queue 를 구현한 제품이고, Sonic MQ 는 Sonic Software 사(www.sonicsoftware.com)에서 messaging 을 구현한 제품이다.

IBM MQ 는 JEUS 의 JMS 와 통합되어있다. 그러므로 J2EE Connector 로 IBM MQ 를 사용하는 것은 피하고, JMS API 를 사용한다. 이는 SONIC MQ 제품도 마찬가지이다.

23.1 MQ 리소스 구성

23.1.1 개요

JEUS 매니저 > MQ > MQ 데이터 소스 개요

이 화면에는 현재 JEUS 매니저에 등록된 MQ 데이터 소스의 목록이 출력된다. 목록에서 MQ 데이터 소스 이름을 클릭하면 해당 설정을 볼 수 있으며 명령에서 바인드를 클릭하면 MQ 데이터 소스가 JNDI 에 등록된다.

JNDI 에 대한 자세한 내용은 JEUS Server 안내서를 참고하기 바란다

23.1.2 추가

JEUS 매니저 > MQ > 새 MQ 데이터 소스 생성

1. 노드 트리에서 **JEUS 리소스**의 **MQ** 폴더를 클릭한 후 **새 MQ 데이터 소스 생성 링크**를 클릭하거나 **MQ** 폴더의 컨텍스트 메뉴에서 **새 MQ 데이터 소스 생성**을 선택한다.
2. Vendor 를 선택한다. Vendor 를 선택하면 아래 리소스 타입을 선택하는 필드가 업데이트 된다.



그림 188 새 MQ 데이터 소스 생성 - 1 단계 리소스 타입 설정

3. 리소스 타입을 선택한 다음 다음 버튼을 누른다.
4. 데이터 소스 속성을 입력한다. 추가 속성에는 데이터 소스의 추가적인 속성을 입력한다. 입력 형식은 “키[:자료형]=값”이다. 자료형이 생략될 경우는 java.lang.String 이 기본 타입이다. 예를 들어 데이터소스 클래스에 setMaxStmtCount(int count) 라는 메소드가 존재하고 10 이라는 값을 입력하고자 한다면, “maxStmtCount:int=10” 으로 입력하면 된다.

그림 189 새 MQ 데이터 소스 생성 - 2 단계 속성 설정

5. 추가할 데이터소스를 생성과 동시에 바인딩 하고 싶다면 바인딩 선택 항목을 선택한 후 생성버튼을 클릭한다.



그림 190 새 MQ 데이터 소스 생성 - 3 단계 생성

6. 데이터 소스가 추가되면 노드 트리의 **MQ** 폴더에 MQ 데이터 소스가 나타난다.

23.1.3 삭제

JEUS 매니저 > MQ > MQ 데이터 소스 개요 > 삭제

- 노드 트리에서 **JEUS 리소스**의 **MQ** 폴더를 클릭하면 MQ 데이터 소스 개요 화면이 나타난다.
- 화면의 WebT 데이터 소스 목록에서 삭제하고자 하는 MQ 데이터 소스의 아이콘을 클릭한다.
- MQ 데이터 소스가 목록에서 사라지며 노드트리에서도 사라진다.



그림 191 MQ 데이터 소스 삭제

참고 : MQ 데이터 소스가 목록에서 사라진다 해도 실제 JNDI에서 사라진 것은 아니다. 따라서 삭제를 완전하게 하기 위해서는 JEUS 매니저를 재시작 해야한다.

23.2 MQ 데이터 소스 설정

23.2.1 기본 설정

JEUS 메니저 > MQ > MQ 데이터 소스 개요 > 설정 > 기본 설정

1. 노드 트리의 **MQ** 폴더에서 수정하고자 하는 MQ 데이터 소스를 클릭하거나 **MQ 데이터 소스 개요** 화면에서 MQ 데이터 소스를 클릭한다.
2. 탭에서 기본 설정 탭을 선택한다.
3. 속성을 변경한다. 변경된 사항은 다음 부트 혹은 바인드 시에 적용된다.
4. 확인 버튼을 클릭한다.

23.3 MQ 데이터 소스 제어

23.3.1 MQ 데이터 소스 바인드

JEUS 메니저 > MQ > 바인드

MQ 데이터 소스 바인드는 네이밍 서버에 데이터 소스 객체를 바인딩하여 어플리케이션이 Lookup하여 사용할 수 있게하는 기능이다.

1. 노드 트리의 **JEUS 리소스**의 **MQ** 폴더를 클릭한다.
2. MQ 데이터 소스 목록에서 바인딩하고자 하는 데이터 소스를 선택한 후 **바인드** 버튼을 클릭한다.
3. MQ 데이터 소스 목록에서 해당 MQ 이터 소스의 **바인드** 버튼이 비활성화 되며 노드 트리에는 활성화된 아이콘()이 나타난다.

mqdatasource1	ibmmq	com.ibm.mq.jms.MQQueueConnectionFactory	QCF	바인드	
---------------	-------	---	-----	------------	--

그림 192 MQ 데이터 소스 바인드

24 JEUS 매니저 서비스

24.1 네이밍 서버 설정

Java Naming and Directory Interface™ (JNDI)는 자바 어플리케이션이 네트워크에서 객체를 찾고 가져올 수 있도록 하는 API이다. 어플리케이션이 알아야 하는 것은, 찾아서 사용하려는 객체의 논리적인 이름뿐이다. 사용자의 관점에서 볼 때는 이전의 엔터프라이즈 환경보다 손쉽게 다양한 객체를 사용할 수 있다.

JEUS Naming Server는 JNSServer와 JNSLocal로 구성되어 있다. 이 둘은 서로 다른 설정을 가진다. JNSServer에서는 JNSLocal의 커넥션을 받아들이는 설정과 다른 JNSServer와 접속하기 위한 설정이 필요하고, JNSLocal은 JNSServer와 접속하기 위한 것과 JNDI 트리의 반영을 위한 설정이 필요하다.

24.1.1 네이밍 서버 기본 설정

JEUS 매니저 > JEUS 매니저 서비스 > 네이밍 서버 > 기본 설정

1. 노드 트리에서 **네이밍 서버**를 클릭한다.
2. 하위 탭에서 **서버**를 클릭한다.
3. 폼내의 탭 패널에서 **기본 설정**을 클릭한다.
4. 각 설정 항목을 입력한다. 변경된 설정은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭한다.



그림 193 네이밍 서버 기본 설정

24.1.2 네이밍 서버 쓰레드 풀 설정

JEUS 매니저 > JEUS 매니저 서비스 > 네이밍 서버 > 쓰레드 풀

1. 노드 트리에서 **네이밍 서버**를 클릭한다.
2. 하위 탭에서 **서버**를 클릭한다.
3. 폼내의 탭 패널에서 **쓰레드 풀**을 클릭한다.
4. 각 설정 항목을 입력한다. 최소값은 런타임시에 의미가 없으므로 최소값을 제외한 나머지는 런타임에 적용이 된다.
5. 입력이 완료되면 **확인**버튼을 클릭한다.



그림 194 네이밍 서버 쓰래드 풀 설정

24.1.3 로컬 네이밍 서버 설정

JEUS 매니저 > JEUS 매니저 서비스 > 네이밍 서버 > 로컬 서버 설정

1. 노드 트리에서 **네이밍 서버**를 클릭한다.
2. 하위 탭에서 **로컬**을 클릭한다.
3. 각 설정 항목을 입력한다. 입력한 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인**버튼을 클릭한다.

24.2 JMX 서버

JMX 서버는 JEUS 매니저와 각 엔진 컨테이너에서 사용된다. 아래의 설정 방법은 엔진 컨테이너에서의 JMX 서버 설정과 같다.

24.2.1 HTTP 아답터 설정

JEUS 매니저 > JEUS 매니저 서비스 > JMX 서버 > HTTP 아답터 설정

1. 노드 트리에서 **JMX**를 클릭 한다.
2. 하위 템에서 **아답터**를 클릭 한다.
3. 안쪽의 템페널에서 **HTTP**를 클릭 한다.
4. **HTTP** 포트를 입력 한다. 입력된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭 한다.



그림 195 JMX HTTP 아답터 설정

24.2.2 JMXMP 설정

JEUS 매니저 > JEUS 매니저 서비스 > JMX 서버 > JMXMP

1. 노드 트리에서 **JMX**를 클릭 한다.
2. 하위 템에서 **아답터**를 클릭 한다.
3. 안쪽의 템페널에서 **JMXMP**를 클릭 한다.
4. **JMXMP** 포트를 입력 한다. 입력된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭 한다.

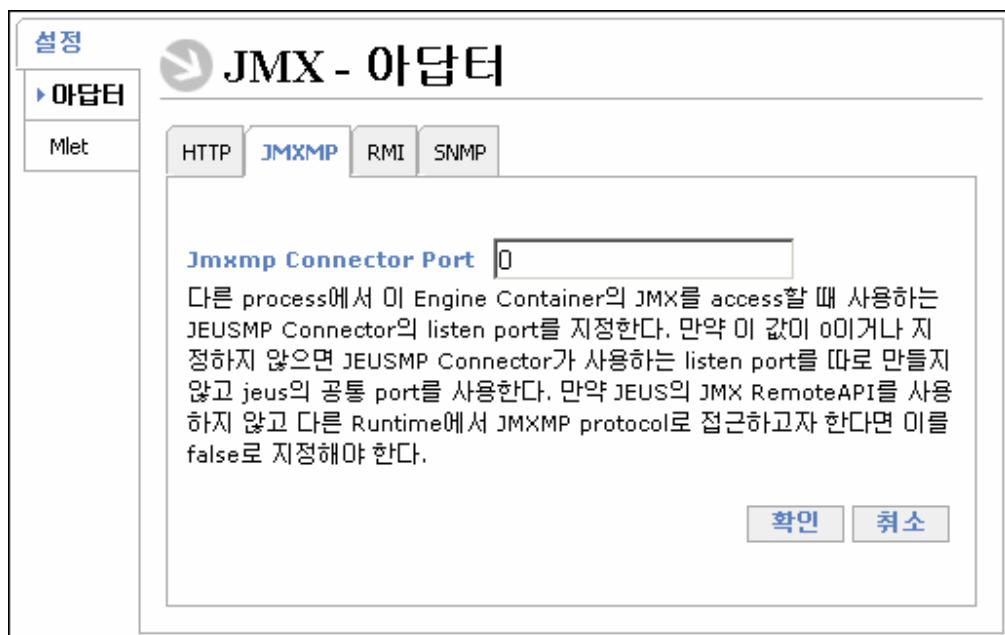


그림 196. JMX JMXMP 아답터 설정

24.2.3 RMI 아답터 설정

JEUS 매니저 > JEUS 매니저 서비스 > JMX 서버 > RMI 아답터 설정

1. 노드 트리에서 **JMX**를 클릭한다.
2. 하위 탭에서 **아답터**를 클릭한다.
3. 안쪽의 탭페널에서 **RMI**를 클릭한다.
4. RMI 포트를 입력한다. 입력된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭한다.



그림 197 JMX RMI 아답터설정

24.2.4 SNMP 아답터설정

JEUS 매니저 > JEUS 매니저 서비스 > JMX 서버 > SNMP 아답터 설정

1. 노드 트리에서 **JMX**를 클릭한다.
2. 하위 탭에서 **아답터**를 클릭한다.
3. 안쪽의 탭에서 **SNMP**를 클릭한다.
4. 각 설정 항목을 입력한다. 변경된 값은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭한다.
6. 설정을 초기화 하고 싶을 경우 **재설정**버튼을 클릭한다.

JMX - 아답터

Mlet

HTTP JMXMP RMI SNMP

Snmp Adaptor Port * [4096]
SNMP 아답터의 리스너 포트

고급 선택사항

Snmp Version [3] SNMP 버전을 지정하며 1, 2 또는 3을 지정할 수 있다.

Snmp Max Packet Size [4096] 바이트
SNMP 패킷에 대한 최대값을 설정하며 최소 256바이트부터 설정 할 수 있다.

Snmp Security [] 보안을 적용시킬 것 인지를 설정한다. 보안은 SNMP 버전 3에서만 설정이 가능 하다.

Trap Demon
트랩데몬을 줄별로 "\${ip_address}:\${port}"의 형식으로 입력한다.
192.168.1.2:9999
192.168.1.3:9999

쓰레드 풀

Max [30] pooling되는 객체의 최대값을 지정한다.

Min [2] pooling되는 객체의 최소값을 지정한다.

Period [3600000] 밀리초
pooling되는 객체를 정리하는 시간을 지정한다.

확인 **취소** **재설정**

그림 198 JMX SNMP 아답터 설정

24.2.5 MLet 등록

JEUS 매니저 > JEUS 매니저 서비스 > JMX 서버 > MLet 등록

1. 노드 트리에서 **JMX**를 클릭 한다.
2. 하위 탭에서 **MLet** 을 클릭 한다.
3. Mlet 의 경로를 줄별로 입력한다. 입력된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인**버튼을 클릭 한다.

24.3 클래스 FTP

JEUS 매니저 > JEUS 매니저 서비스 > 클래스 FTP

클래스 FTP Server 는 EJB 스텁 클래스를 FTP 를 통해서 전송하기 위해 사용한다.

클래스 FTP 의 설정은 4.1.5_서비스 설정을 참조하라.

24.4 스케줄러

스케줄러는 미리 지정한 시간에 특정 작업들이 실행되도록 하는데 사용된다.

스케줄러 서비스의 시작과 종료는 4.1.5_서비스 설정을 참조하라.

24.4.1 스케줄러 작업 추가

JEUS 매니저 > JEUS 매니저 서비스 > 스케줄러 > 새 스케줄러 작업 생성

1. 노드 트리에서 스케줄러를 클릭한다.
2. 새 스케줄러 작업 생성링크를 클릭한다.
3. 각 설정 항목을 입력한다.
4. 입력이 완료되면 생성 버튼을 클릭한다.
5. 작업 목록에 새로 만들어진 작업이 추가된다.

24.4.2 스케줄러 작업 제거

JEUS 매니저 > JEUS 매니저 서비스 > 스케줄러 > 스케줄러 작업 제거

1. 노드 트리에서 스케줄러를 클릭한다.
2. 삭제하고자 하는 작업을 선택한 후  아이콘을 클릭한다.
3. 작업 목록에서 작업이 삭제된다.

24.4.3 스케줄러 작업 설정

JEUS 매니저 > JEUS 매니저 서비스 > 스케줄러 > 스케줄러 작업 설정

1. 노드 트리에서 **스케줄러**를 클릭한다.
2. 스케줄러 작업목록에서 변경하고자 하는 작업을 선택한다.
3. 각 설정 항목을 입력한다. 변경된 값은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인**버튼을 클릭한다.

24.5 로그 서비스

JEUS 는 JEUS 에서 발생하는 여러가지 상황들을 로그를 통해 알려준다. JEUS 로그는 J2SE 1.4 이상의 버전에서 logging API 를 기반으로 동작한다. 따라서 설정하는 방식도 logging API 의 로거, 핸들러, 포맷터 구조를 그대로 반영하고 있으며 개발자가 logging API 를 이용하여 JEUS 의 로거를 원하는 대로 사용할 수 있다.

비록 로거의 생성이 자유롭기는 하나 JEUS 내부적으로 사용되는 로거중에 각각의 설정을 XML 파일에 영구적으로 저장하는 것들이 있다. 이 장에서는 영구적으로 저장되는 로그 설정과 런타임에 로그의 수준을 제어하는 것에 대해서 알아볼 것이다.

참고 : 이 절에서는 서블릿 엔진의 엑세스로그를 기준으로 설명할 것이다. 다른 로그의 설정도 엑세스 로그와 유사하다.

24.5.1 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 엑세스로그 > 기본 설정

기본 설정에서는 로그의 수준과 포맷과 같은 공통적인 항목에 대해서 설정 한다.

1. 노드 트리에서 **서블릿 엔진**을 클릭한 후 **설정** 탭을 클릭하거나 컨텍스트 메뉴에서 **설정** 서블릿 엔진을 선택한다.
2. 주 화면의 탭에서 **엑세스로그**를 선택한다.
3. 각 설정 항목을 입력한다. 변경된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 **확인** 버튼을 클릭한다.

참고 : 로그 수준의 수정은 런타임에 적용되지 않는다. 로그 수준의 제어에 대해서는 24.5.8 로그 수준 제어를 참조하기 바란다.

24.5.2 콘솔 핸들러 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 엑세스 로그 > 새 콘솔 핸들러 생성

콘솔 핸들러는 로그를 콘솔로 출력하는 핸들러이다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 설정 서블릿 엔진을 선택한다.
2. 주 화면의 탭에서 엑세스 로그를 선택한다.
3. 화면의 하단에서 새 콘솔 핸들러 생성을 클릭한다.
4. 각 설정 항목을 입력한다.
5. 입력이 완료되면 생성 버튼을 클릭한다.
6. 핸들러 목록에 콘솔 핸들러가 추가된다.

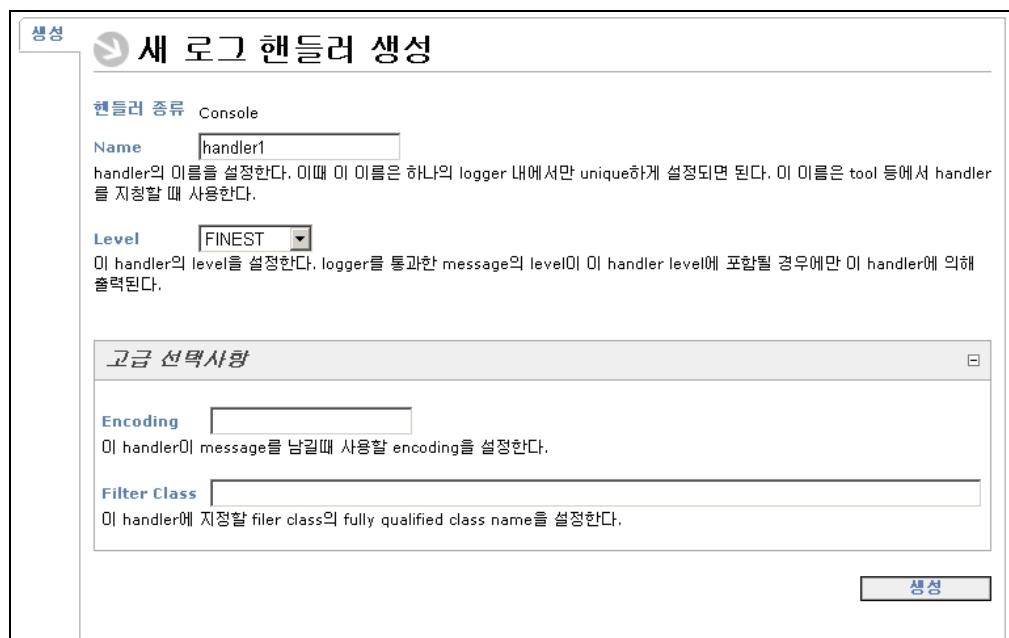


그림 199 새 콘솔 핸들러 생성

24.5.3 파일 핸들러 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 엑세스 로그 > 새 파일 핸들러 생성

파일 핸들러는 로그를 파일로 출력하는 핸들러이다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 서블릿 엔진 설정을 선택한다.
2. 하위 탭에서 엑세스 로그를 선택한다.
3. 새 파일 핸들러 생성을 클릭한다.
4. 각 설정 항목을 입력한다.
5. 입력이 완료되면 생성 버튼을 클릭한다.
6. 핸들러 목록에 파일 핸들러가 추가된다.

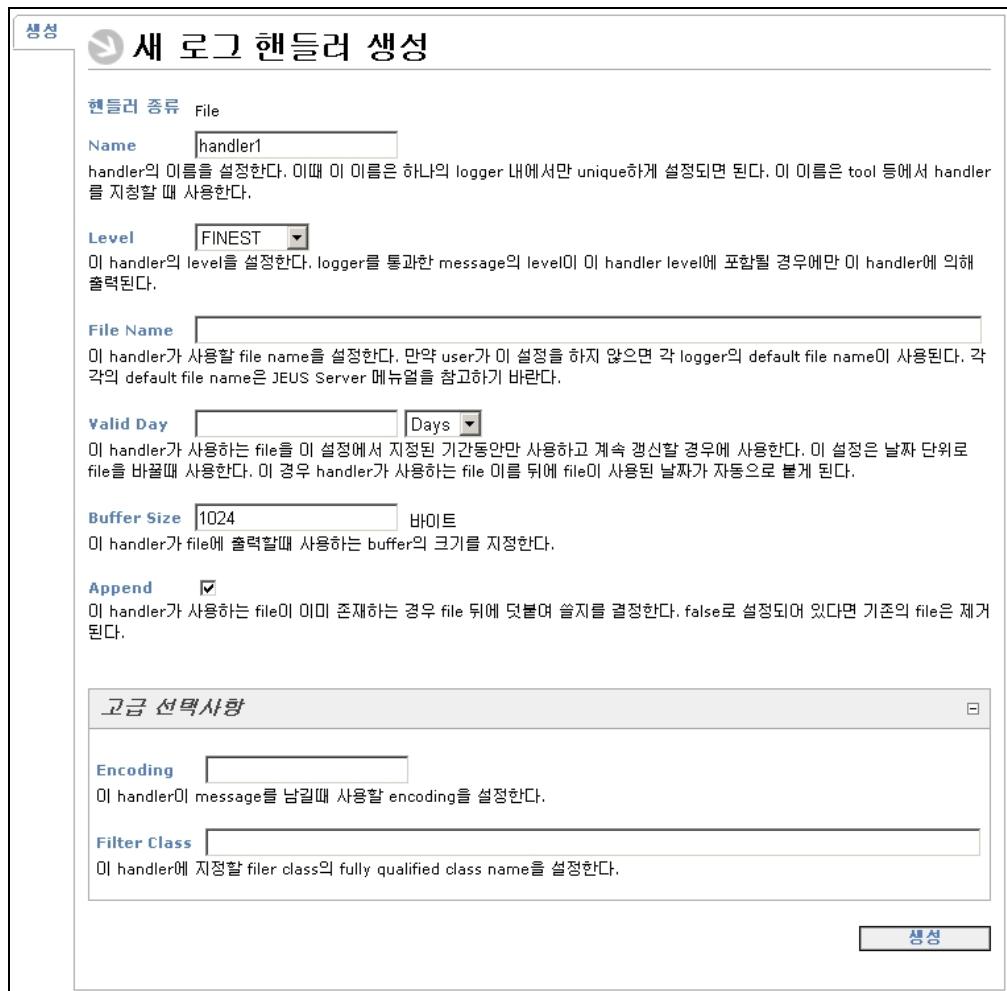


그림 200 새 파일 핸들러 생성

24.5.4 SMTP 핸들러 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 설정 > 엑세스 로그 > 새 SMTP 핸들러 생성

SMTP 핸들러는 로그를 이메일로 출력하는 핸들러이다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 설정 서블릿 엔진을 선택한다.
2. 하위 탭에서 엑세스 로그를 선택한다.
3. 새 SMTP 핸들러 생성을 클릭한다.

4. 각 설정 항목을 입력한다.
5. 입력이 완료되면 생성버튼을 클릭한다.
6. 핸들러 목록에 SMTP 핸들러가 추가된다.

생성 새 로그 핸들러 생성

핸들러 종류 Ssmtp
Name handler1
 handler의 이름을 설정한다. 이때 이 이름은 하나의 logger 내에서만 unique하게 설정되면 된다. 이 이름은 tool 등에서 handler를 지정할 때 사용한다.

Level FINEST
 이 handler의 level을 설정한다. logger를 통과한 message의 level이 이 handler level에 포함될 경우에만 이 handler에 의해 출력된다.

Smtp Host Address *
 email을 보낼 smtp server의 주소를 지정한다.

From Address *
 email을 보내는 사람의 address를 지정한다.

To Address *
 email을 받는 사람의 address를 지정한다.

Cc Address
 email을 참조로 받는 사람의 address를 지정한다.

Bcc Address
 email을 숨은 참조로 받는 사람의 address를 지정한다.

Send For All Messages
 이 handler가 등록한 logger의 log() method를 통해 들어온 message들이 이 handler로 들어왔을 때 이를 email로 보낼 대상으로 여길지를 설정한다. 만약 false로 설정되어 있으면 logger의 특별한 send() method로 호출된 message들만 email로 전송된다. 즉, 처음부터 email로 보낼 의도로 지정된 message들만 email로 전송된다.

고급 선택사항

Encoding
 이 handler의 message를 남길 때 사용할 encoding을 설정한다.

Filter Class
 이 handler에 지정할 filter class의 fully qualified class name를 설정한다.

생성

그림 201 새 SMTP 핸들러 생성

24.5.5 소켓 핸들러 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 엑세스 로그 > 새 소켓 핸들러 생성

소켓 핸들러는 로그를 지정된 소켓으로 전송하는 핸들러이다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 설정 서블릿 엔진을 선택한다.
2. 하위 탭에서 엑세스 로그를 선택한다.
3. 새 소켓 핸들러 생성을 클릭한다.
4. 각 설정 항목을 입력한다.
5. 입력이 완료되면 생성버튼을 클릭한다.
6. 핸들러 목록에 소켓 핸들러가 추가된다.

생성 새 로그 핸들러 생성

핸들러 종류 Socket

Name handler의 이름을 설정한다. 이때 이 이름은 하나의 logger 내에서만 unique하게 설정되면 된다. 이 이름은 tool 등에서 handler를 지정할 때 사용한다.

Level handler의 level을 설정한다. logger를 통과한 message의 level이 이 handler level에 포함될 경우에만 이 handler에 의해 출력된다.

Address * 이 handler가 생성될 때 message들을 보낼 곳의 IP address를 설정한다.

Port * 이 handler가 생성될 때 message들을 보낼 곳의 port를 설정한다.

고급 선택사항

Encoding 이 handler가 message를 남길 때 사용할 encoding을 설정한다.

Filter Class 이 handler에 지정할 filter class의 fully qualified class name를 설정한다.

그림 202 새 소켓 핸들러 생성

24.5.6 사용자 핸들러 추가

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 엑세스 로그 > 새 사용자 핸들러 생성

사용자가 만든 핸들러 클래스를 지정하는 항목이다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 설정 서블릿 엔진을 선택한다.
2. 하위 탭에서 엑세스 로그를 선택한다.
3. 새 사용자 핸들러 생성을 클릭한다.
4. 각 설정 항목을 입력한다.
5. 입력이 완료되면 생성버튼을 클릭한다.
6. 핸들러 목록에 사용자 핸들러가 추가된다.

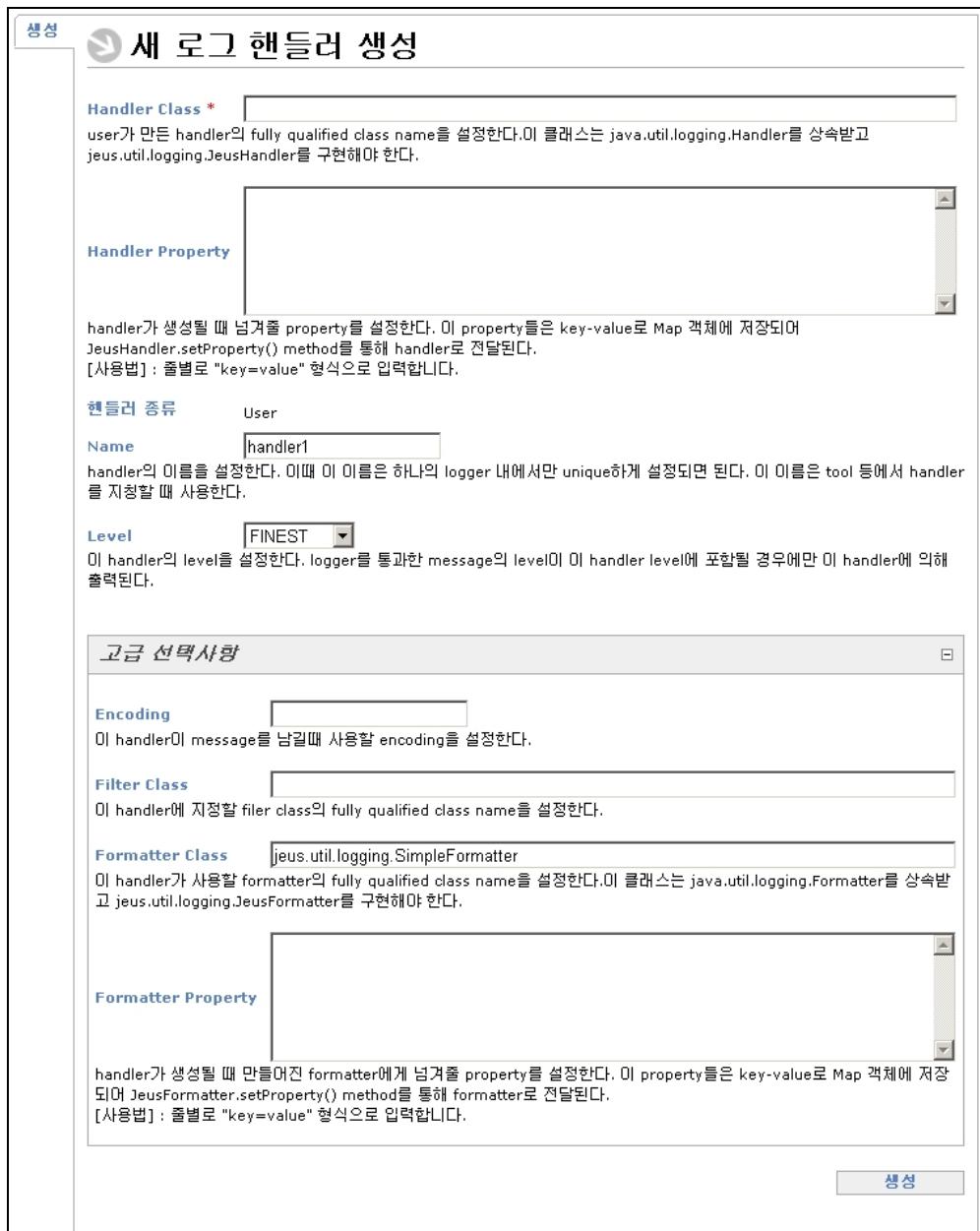


그림 203 사용자 로그 핸들러 추가

24.5.7 핸들러 삭제

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 설정 > 엑세스 로그 > 핸들러 삭제

핸들러의 삭제는 모든 핸들러 타입에서 똑 같다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 설정 탭을 클릭하거나 컨텍스트 메뉴에서 설정 서블릿 엔진을 선택한다.
2. 하위 탭에서 엑세스 로그를 선택한다.
3. 삭제하고자 하는 핸들러를 선택한 후  아이콘을 클릭한다.
4. 핸들러가 삭제되고 핸들러 목록에서 사라진다.

로그 핸들러

이름	종류	수준	인코딩	필터 클래스	
handler1	Console	FINEST			
handler2	File	FINEST			

그림 204 로그 핸들러 삭제

참고 : 로그 핸들러를 삭제하더라도 실제적으로 삭제되는 것은 아니고 다음 부트시에 적용된다.

24.5.8 로그 수준 제어

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 로그 서비스

JEUS 내에서 사용되는 모든 로그는 런타임에 레벨을 조정 할 수 있다. 엔진 컨테이너의 로그일 경우는 엔진 컨테이너의 로그 서비스를 이용하고 JEUS 매니저의 로그일 경우는 JEUS 매니저 서비스의 로그 서비스를 이용한다.

1. 노드 트리에서 로그를 클릭한다.
2. 좌측에는 로거 트리가 나타난다.
3. 로거 트리에서 수준을 조정하고자 하는 로거를 선택한다.

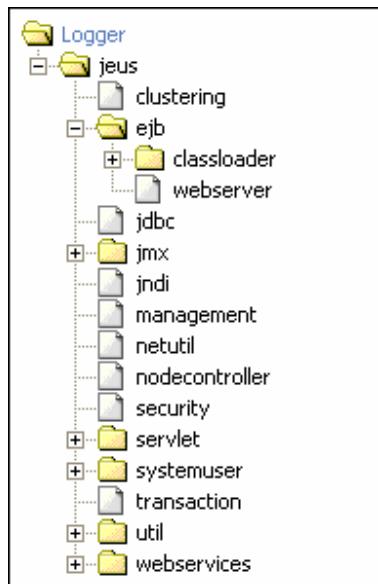


그림 205 로거 트리

- 로거를 선택하면 우측에 현재 로거의 이름과 로그 수준을 출력한다.
- 로그 수준 콤보박스에서 원하는 로그 수준을 선택한 후, 만약 하위의 로거에까지 동일한 로그 수준을 적용하고 싶을 경우 반복 적용을 선택 한다.
- 입력이 완료되면 확인버튼을 클릭한다.

로거 이름	jeus.ejb.webserver
수준	SEVERE
반복적으로 적용	<input type="checkbox"/>
확인 취소	

그림 206 로그 수준 설정

25 보안

25.1 도메인

도메인이란 보안 서비스 인스턴스의 집합이라 할 수 있다. 하나의 보안 시스템에 여러 개의 도메인이 동시에 있을 수 있다.

도메인 개념의 배경에는 서로 다른 어플리케이션이나 JEUS 하위 시스템이 각각 별도의 보안 서비스를 사용할 수 있도록 하자는데 있다. 도메인은 보안 서비스를 각 어플리케이션별로 분리하는 역할을 한다.

25.1.1 도메인 추가

JEUS 매니저>JEUS 매니저 서비스>보안>도메인>추가

1. 노드 트리에서 **보안**을 클릭한 후 도메인목록 하단에 위치한 새 보안 도메인 생성을 클릭한다.
2. 기존에 존재하는 도메인설정을 그대로 복사하고 싶을 경우 원본 도메인 이름에서 도메인을 선택한다.
3. 도메인 이름에 생성하고자 하는 도메인 이름을 입력한다. *SYSTEM_DOMAIN*은 이미 존재하므로 생성 할 수가 없다.
4. 동일한 도메인의 이름에 해당하는 디렉토리가 있을 경우, 그 디렉토리의 설정을 그대로 사용하려면 ”기존의 설정 사용” 체크 박스를 선택한다.
5. 입력이 완료되면 **생성**을 클릭한다.
6. 생성된 도메인의 아이콘()이 노드 트리에 나타난다.

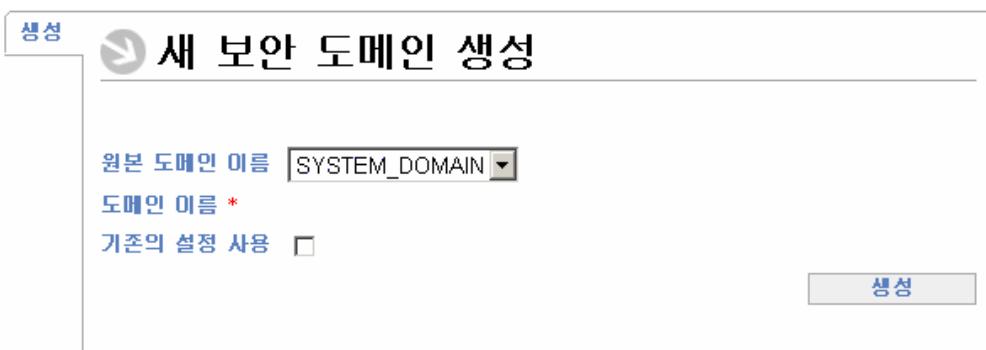


그림 207 새 보안 도메인 추가

25.1.2 도메인 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 삭제

1. 노드 트리에서 보안을 클릭 한다.
2. 삭제하고자 하는 도메인을 선택한 후 을 클릭 한다.
3. 도메인이 삭제되고 노드 트리에서 도메인 아이콘()이 사라진다.

25.2 보안 서비스

보안 시스템의 각 기능들을 구현하고 있는 클래스들을 서비스라고 한다. 다시 말해 서비스란 특정 보안 기능 - 인증, 권한 부여, 네트워킹, 기타 - 을 제공하는 SPI(Service Provider Interface)를 구현한 클래스이다.

25.2.1 서비스 추가

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서비스 > 추가

1. 노드 트리에서 서비스를 추가하고자 하는 도메인을 클릭한 후 서비스를 클릭 한다.
2. 하위 탭에서 마스터를 선택한다.
3. 생성 버튼을 클릭 한다.
4. 각 입력 항목을 입력한다. 속성에는 서비스의 속성들을 “키=값” 형식으로 입력한다.

5. 입력이 완료되면 확인 버튼을 클릭한다.
6. 서비스 목록에 서비스가 추가된다.

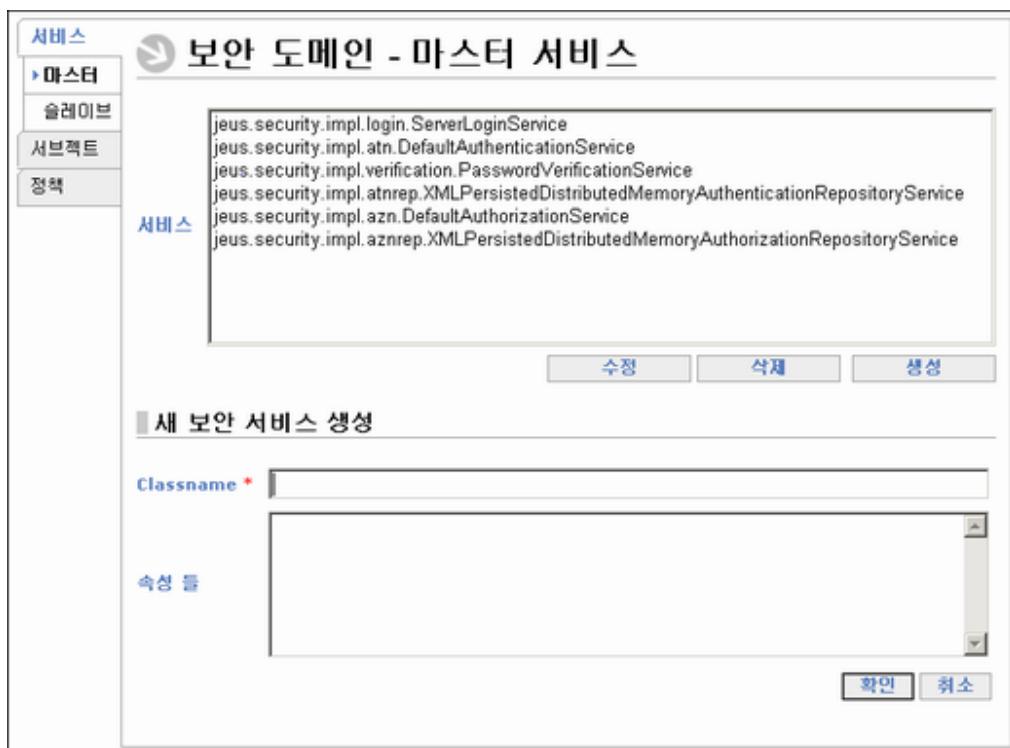


그림 208 서비스 추가

25.2.2 서비스 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서비스 > 삭제

1. 노드 트리에서 서비스를 추가하고자 하는 도메인을 클릭한 후 서비스를 클릭한다.
2. 하위 탭에서 마스터를 선택한다.
3. 삭제하고자 하는 서비스를 선택한 후 삭제 버튼을 클릭한다.
4. 서비스 목록에서 서비스가 삭제된다.

25.2.3 서비스 수정

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서비스 > 수정

1. 노드 트리에서 서비스를 추가하고자 하는 도메인을 클릭한 후 서비스를 클릭한다.
2. 하위 탭에서 마스터를 선택한다.
3. 수정 버튼을 클릭한다.
4. 각 입력 항목을 입력한다.
5. 입력이 완료되면 확인 버튼을 클릭한다.
6. 서비스 목록에 서비스가 추가된다.

25.3 서브젝트

Subject 설정에는 기본 설정과 전문가 설정으로 나뉘어져 있다. 기본 설정에서는 패스워드 설정을 `jeus.security.resource.PasswordFactory` 을 이용해서 하는 경우 즉, BASE64 인코딩을 사용하여 패스워드를 만드는 경우 사용된다. 전문가 설정은 좀더 세밀하게 입력할 경우 사용된다.

25.3.1 기본 설정에서의 서브젝트 추가

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 기본 설정 > 추가

1. 노드 트리에서 Subject 를 추가하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 기본 설정을 선택한다.
3. 생성버튼을 클릭한다.
4. 각 항목을 입력한다.
5. 입력이 완료되면 확인 버튼을 클릭한다.
6. Subject 목록에 방금 생성한 서브젝트가 추가된다.

새 보안 서브젝트 생성

이름 *	<input type="text"/>
암호 *	<input type="password"/>
암호 확인 *	<input type="password"/>
<input type="button" value="확인"/> <input type="button" value="취소"/>	

그림 209 서브젝트 추가

25.3.2 기본 설정에서의 서브젝트 수정

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 기본 설정 > 설정

1. 노드 트리에서 서브젝트를 추가하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 기본 설정을 선택한다.
3. 수정하고자 하는 서브젝트를 선택한 후 수정버튼을 클릭한다.
4. 암호를 새로 입력한다.
5. 입력이 완료되면 확인 버튼을 클릭한다.

25.3.3 서브젝트 생성

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > 생성

1. 노드 트리에서 Subject를 추가하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 전문가 설정을 선택한다.
3. 생성버튼을 클릭한다.
4. 서브젝트 이름을 입력한다.
5. 입력이 완료되면 확인 버튼을 클릭한다.

6. 서브젝트목록에 방금 생성한 서브젝트가 추가된다.

25.3.4 서브젝트 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 삭제

1. 노드 트리에서 삭제하고자 하는 서브젝트를 가진 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 서브젝트 목록에서 삭제하고자 하는 서브젝트를 클릭한다.
3. 삭제버튼을 클릭한다.
4. 서브젝트 목록에서 서브젝트가 삭제된다.

25.3.5 서브젝트 저장

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 저장

서브젝트에 대한 변경 사항은 메모리 내에서만 이루어진다. 따라서 물리적으로 영구적으로 저장하기 위해서는 이 기능을 사용해야 한다.

1. 노드 트리에서 서브젝트를 저장하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 저장버튼을 클릭한다.
3. 서브젝트들이 저장된다.

25.3.6 서브젝트에 Principal 생성

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > *Principle* 생성

1. 노드 트리에서 Principal을 추가하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 템에서 전문가를 선택한다.
3. 수정버튼을 클릭한다.
4. 하단의 템에서 **Principal**을 선택한다.

5. 생성 버튼을 클릭한다.
6. 각 항목을 입력한다.
7. 입력이 완료되면 확인버튼을 클릭한다.
8. 방금 생성한 Principal o] Principal 목록에 추가된다.

서브젝트 수정

Principal 크레덴셜 팩토리

jeus[main]

삭제 생성

새 주역 생성

클래스 이름 * jeus.security.resource.GroupPrincipalImpl

이름 *

확인 취소

그림 210 Principal 추가

25.3.7 서브젝트에서 Principal 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > **Principle 삭제**

1. 노드 트리에서 Principal 을 삭제하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 전문가를 선택한다.
3. 수정버튼을 클릭한다.
4. 하단의 탭에서 **Principal** 을 선택한다.

5. Principal 목록에서 삭제하고자 하는 Principal 을 선택한다. Main Principal –”[main]”으로 표시된 Principal –은 삭제가 불가능하다.
6. 삭제 버튼을 클릭한다.
7. Principal 목록에서 방금 삭제한 Principal 이 삭제된다.

25.3.8 서브젝트에 크리덴셜 팩토리 추가

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > 크리덴셜 팩토리 추가

1. 노드 트리에서 Credential Factory 를 추가하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 템에서 전문가를 선택한다.
3. 수정버튼을 클릭한다.
4. 하단의 템에서 크리덴셜 팩토리를 선택한다.
5. 생성버튼을 클릭한다.
6. 각 입력항목을 입력한다. 속성의 값은 “키=값” 형식으로 입력한다. 만약 값이 한 줄이상이라면 시작 줄은 “키=[”으로 끝나는 줄은 ”]]”으로 입력하고 그 중간에 값을 입력하여야 한다.
7. 입력이 완료되면 확인버튼을 클릭한다.
8. 방금 생성한 크리덴셜 팩토리가 크리덴셜 팩토리 목록에 추가된다.

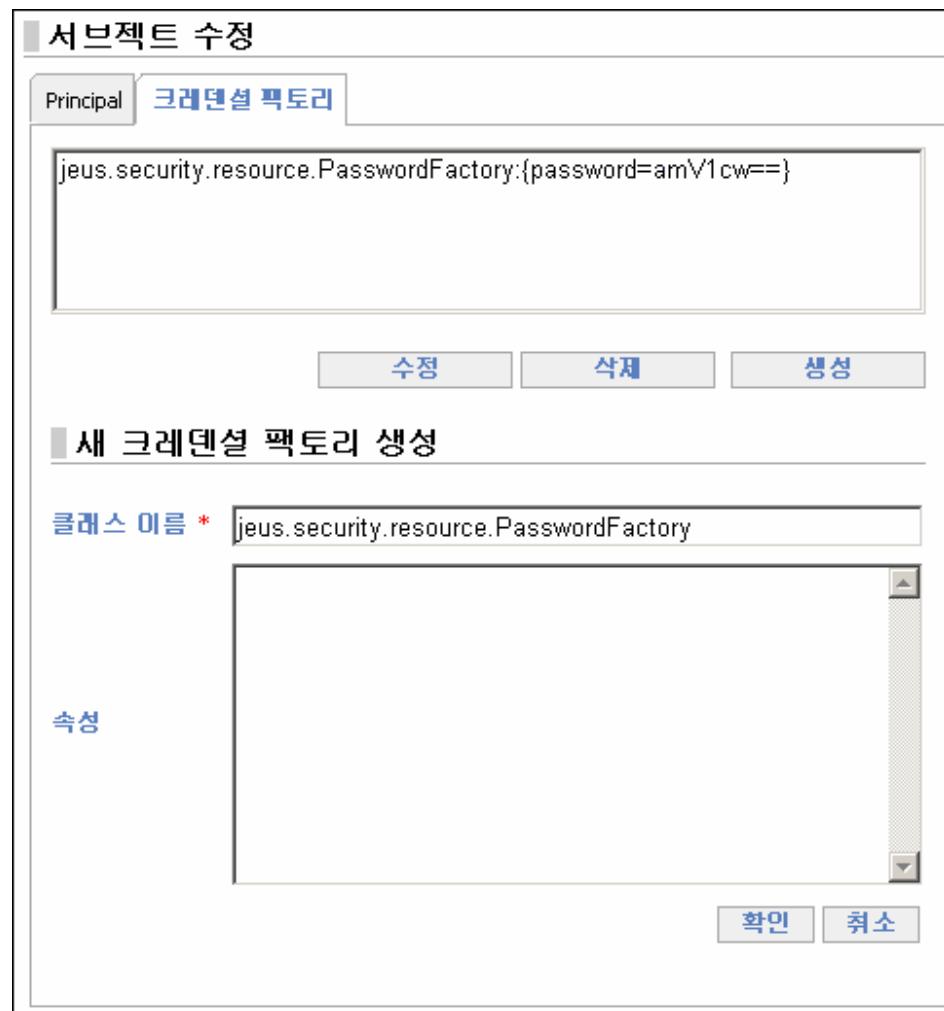


그림 211 Credential Factory 추가

25.3.9 서브젝트에서 크리덴셜 팩토리 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > 크리덴셜 팩토리삭제

1. 노드 트리에서 크리덴셜 팩토리를 삭제하고자 하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 전문가를 선택한다.
3. 수정버튼을 클릭한다.
4. 하단의 탭에서 크리덴셜 팩토리를 선택한다.

5. 크리덴셜 팩토리 목록에서 삭제하고자 하는 크리덴셜 팩토리를 선택한다.
6. 삭제 버튼을 클릭한다.
7. 크리덴셜 팩토리 목록에서 크리덴셜 팩토리가 삭제된다.

25.3.10 서브젝트에 크리덴셜 팩토리 수정

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 서브젝트 > 전문가 설정 > 크리덴셜 팩토리 수정

1. 노드 트리에서 수정하고자 크리덴셜 팩토리를 포함하는 도메인을 클릭한 후 서브젝트를 클릭한다.
2. 하위 탭에서 전문가를 선택한다.
3. 수정버튼을 클릭한다.
4. 하단의 탭에서 크리덴셜 팩토리를 선택한다.
5. 크리덴셜 팩토리 목록에서 수정하고자 하는 크리덴셜 팩토리를 선택한다.
6. 수정 버튼을 클릭한다.
7. 각 입력 항목을 입력한다.
8. 입력이 완료되면 확인버튼을 클릭한다.

25.4 정책

25.4.1 컨텍스트 식별자 생성

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 새 컨텍스트 식별자 생성

1. 노드 트리에서 컨텍스트 식별자를 추가하고자 하는 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록 아래의 생성 버튼을 클릭한다.

3. 컨텍스트 식별자에 입력을 한다.
4. 입력이 완료되면 확인 버튼을 클릭한다.
5. 컨텍스트 식별자가 컨텍스트 식별자 목록에 추가된다.

The screenshot shows a dialog box titled "새 컨텍스트 생성" (New Context Creation). Inside, there is a text input field labeled "컨텍스트 식별자 *". Below the input field are two buttons: "확인" (Confirm) on the right and "취소" (Cancel) on the far right.

그림 212 컨텍스트 식별자 생성

25.4.2 컨텍스트 식별자 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 컨텍스트 식별자 삭제

1. 노드 트리에서 컨텍스트 식별자를 삭제하고자 하는 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 삭제하고자 하는 컨텍스트 식별자를 선택한다.
3. 삭제 버튼을 클릭한다.
4. 컨텍스트 식별자가 컨텍스트 식별자 목록에서 사라진다.

25.4.3 정책 저장

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 저장

1. 노드 트리에서 저장하고자 하는 정책들이 있는 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 저장 버튼을 클릭한다.
3. 정책들이 저장된다.

25.4.4 역할 생성

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 역할 생성

1. 노드 트리에서 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 역할을 생성하고자하는 컨텍스트 식별자를 선택한다.
3. 탭에서 역할 탭을 선택한다.
4. 생성 버튼을 클릭한다.
5. 각 입력 항목을 입력한다.
6. 입력이 완료되면 확인 버튼을 입력한다.
7. 생성된 역할이 역할 권한 목록에 추가된다.

The screenshot shows the JEUS Web Manager interface with the following details:

- Header:** JEUS 웹 관리자 안내서 | JEUS
- Top Navigation:** 역할 (selected), 리소스
- Section 1: 역할 권한**
 - Table header: 역할 권한
 - Table body: administrator
 - Action buttons: 수정, 삭제, 생성
- Section 2: 새 역할 권한 생성**
 - Form fields:
 - 이름 *: jndiUser
 - 클래스 이름 *: jeus.security.resource.RolePermission
 - 행동: (empty input field)
 - 주역: (checkboxes)
 - administrator securityadmin: SELECT button is highlighted
 - jeus: DESELECT button is highlighted
 - 검사: (checkbox)
 - 제외: (checkbox)
 - Action buttons: 확인, 취소

그림 213 역할 추가

25.4.5 역할 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 역할 삭제

1. 노드 트리에서 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 삭제하고자 하는 역할이 포함된 컨텍스트 식별자를 선택한다.
3. 탭에서 역할 탭을 선택한다.

4. 역할 목록에서 역할을 선택한다.
5. 삭제 버튼을 클릭한다.
6. 역할이 역할 목록에서 사라진다.

25.4.6 역할 수정

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 역할 수정

1. 노드 트리에서 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 수정하고자 하는 역할이 포함된 컨텍스트 식별자를 선택한다.
3. 탭에서 역할 탭을 선택한다.
4. 역할 목록에서 역할을 선택한다.
5. 수정버튼을 클릭한다.
6. 각 항목을 입력한다.
7. 입력이 완료되면 확인버튼을 클릭한다.

25.4.7 리소스 생성

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 리소스 생성

1. 노드 트리에서 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 리소스를 추가하고자 하는 컨텍스트 식별자를 선택한다.
3. 탭에서 리소스탭을 선택한다.
4. 생성버튼을 클릭한다.
5. 각 항목을 입력한다.
6. 입력이 완료되면 확인버튼을 클릭한다.

리소스 권한

리소스	jeus.*{*}
-----	-----------

새 리소스 권한 생성

이름 *	jeus.jndi.*
클래스 이름 *	jeus.security.resource.ResourcePermission
행동	lookup
역할	<div style="display: flex; align-items: center;"> <div style="border: 1px solid #ccc; padding: 5px; margin-right: 10px;">administrator</div> <div style="border: 1px solid #ccc; padding: 5px; margin-right: 10px;">jndiUser</div> <div style="margin-right: 10px;"> <input type="button" value="SELECT"/> <input type="button" value="DESELECT"/> </div> </div>
검사	<input type="checkbox"/>
제외	<input type="checkbox"/>

확인 **취소**

그림 214 Resource 추가

25.4.8 리소스 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > 정책 > 리소스 삭제

1. 노드 트리에서 도메인을 클릭한 후 정책 탭을 클릭한다.
2. 컨텍스트 식별자 목록에서 삭제하고자 하는 리소스가 포함된 컨텍스트 식별자를 선택한다.
3. 탭에서 리소스 탭을 선택한다.
4. 리소스 목록에서 리소스를 선택한다.

5. 삭제버튼을 클릭한다.
6. 리소스가 리소스목록에서 사라진다.

25.4.9 리소스 수정

JEUS 매니저 > JEUS 매니저 서비스 > 보안 > 도메인 > Policy > Resource 수정

1. 노드 트리에서 도메인을 클릭한 후 **Policies** 탭을 클릭한다.
2. Context 목록에서 수정하고자 하는 Resource 가포함된 Context를 선택한다.
3. 탭에서 **Resource** 탭을 선택한다.
4. Resource 목록에서 Resource를 선택한다.
5. 수정버튼을 클릭한다.
6. 각 항목을 입력한다.
7. 입력이 완료되면 확인버튼을 클릭한다.

26 세션 추적

26.1 세션 클러스터링 설정

26.1.1 세션 클러스터링 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿
엔진 > 설정 > 세션

1. 노드 트리에서 서블릿 엔진을 클릭한 후 세션 탭을 클릭 한다.
2. 하위 탭에서 기본 설정을 선택 한다.
3. 각 입력사항을 입력 한다. 입력된 항목은 다음 부트시에 적용된다.
4. 입력이 완료되면 확인버튼을 클릭 한다.

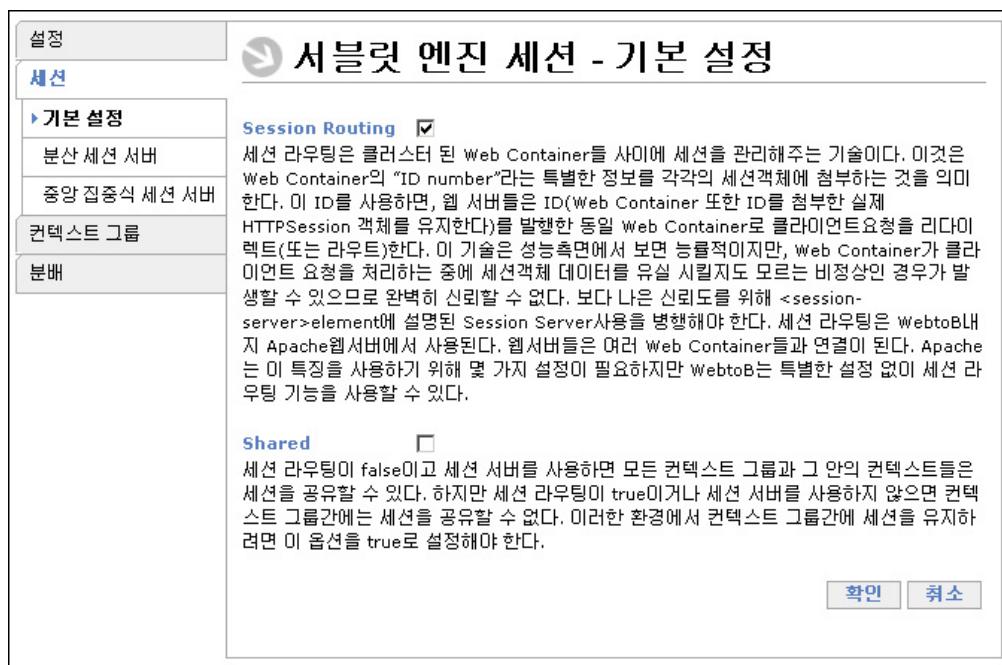


그림 215 세션 클러스터링 설정

26.2 분산 세션 서버

분산 세션 서버는 JEUS 5 부터 새롭게 소개되는 세션 서버 방식이다. 기존의 중앙 집중식 세션 서버의 문제점들을 개선하여 보다 높은 성능을 발휘 할 수 있으므로 가능하면 이 세션서버를 사용할 것을 적극 권장한다.

26.2.1 분산 세션 서버 기본 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 분산 세션 서버 > 기본 설정

1. 노드 트리에서 서블릿 엔진을 클릭한 후 세션탭을 클릭한다.
2. 하위 탭에서 분산 세션 서버를 선택한다.
3. 폼내의 탭 패널에서 기본 설정을 선택한다.
4. 설정 값들을 입력하고, 입력이 완료되면 확인버튼을 클릭한다.

26.2.2 분산 세션 서버 파일 DB 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 분산 세션 서버 > 파일DB

1. 노드 트리에서 서블릿 엔진을 클릭한 후 세션 탭을 클릭한다.
2. 하위 탭에서 분산 세션 서버를 선택한다.
3. 폼내의 탭 패널에서 파일 DB 를 선택한다.
4. 각 입력 사항을 입력한다. 입력된 내용은 다음 부트시에 적용된다.
5. 입력이 완료되면 확인버튼을 클릭한다.

26.2.3 분산 세션 서버 백업 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 분산 세션 서버 > 백업

1. 노드 트리에서 서블릿 엔진을 클릭한 후 세션탭을 클릭한다.

2. 하위 탭에서 **분산 세션 서버**를 선택한다.
3. 폼내의 탭 패널에서 **백업**을 선택한다.
4. 각 입력 사항을 입력한다. 입력된 내용은 다음 부트시에 적용된다.
5. 입력이 완료되면 **확인**버튼을 클릭한다.

26.2.4 분산 세션 서버 공통 기본 설정

JEUS 매니저 > JEUS 매니저 서비스 > 분산 세션 서버 > 기본 설정

이 설정은 모든 분산 세션 서버에 공통적으로 영향을 미치는 설정이다. 이 설정을 하기 앞서 적어도 하나 이상의 분산 세션 서버가 설정되어 있어야 하므로 각 서블릿 엔진에서 분산 세션 서버를 미리 설정하기 바란다.

1. 노드 트리에서 **분산 세션 서버**를 클릭한 후 **기본 설정**탭을 클릭한다.
2. 각 입력 사항을 입력한다. 입력된 내용은 다음 부트시에 적용된다.
3. 입력이 완료되면 **확인**버튼을 클릭한다.

설정	설정 내용	설명
▶ 일반 설정		
기본 파일 DB		
Connect Timeout 60000 밀리초	WebContainer에 존재하는 session server간 socket connection을 생성할 때 적용되는 timeout 값이다.	
Read Timeout 60000 밀리초	WebContainer에 존재하는 session server간 통신시에 적용되는 read timeout 값이다.	
Backup Trigger 1000 오브젝트	local session server에서 session 객체의 update가 어느 정도 발생하였을 때 backup session server로 update된 session 객체들을 backup할지를 결정한다. 이 설정에 지정된 횟수 만큼 local session server에 session object update가 발생하면 backup를 수행한다.	
Check To 30000 밀리초	얼마만큼의 시간 간격으로 backup과정을 수행할지를 결정한다. 이 설정에 지정된 시간 주기로 update된 session 객체가 있는지를 조사하고 update된 session 객체가 존재하면 backup를 수행한다.	
Check Level set	사용된 session을 remote web container 또는 local file db에 백업하기 전에 백업할 필요가 있는지를 체크하는 것이 필요하다. 이 설정은 백업의 필요성을 체크하는 기준을 정한다. 기본적으로 사용된 세션이 invalidate되었을 경우 설정한 기준에 관계없이 백업한다.	

그림 216 세션 서버 공통 기본 설정

26.2.5 분산 세션 서버 공통 기본 파일 DB 설정

JEUS 매니저 > JEUS 매니저 서비스 > 분산 세션 서버 > 기본 파일 DB 설정

1. 노드 트리에서 분산 세션 서버를 클릭한 후 기본 파일 DB 탭을 클릭한다.
2. 각 입력 사항을 입력한다. 입력된 내용은 다음 부트시에 적용된다.
3. 입력이 완료되면 확인버튼을 클릭한다.



그림 217 세션 서버 공통 기본 파일 DB 설정

26.2.6 분산 세션 서버 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 > 엔진 > 세션 컨테이너

세션 서버의 모니터링은 세션 컨테이너를 통해서 한다. 세션 컨테이너는 세션 객체를 관리하는 서블릿 엔진 내부의 객체로서 세션 서버 설정에 따라 나타나는 위치가 다르다. 세션 클러스터링 설정에서 shared 를 “true”로 하였을 경우에는 서블릿 엔진 밑에 나타나며 그렇지 않을 경우에는 각 컨텍스트 그룹 밑에 나타나게 된다.

1. 노드 트리에서 세션 컨테이너를 클릭한다.

2. 모니터링 정보가 출력된다.

종류	Distributed
비활성 세션 수	0
지역 파일 DB 크기	0
백업 세션 수	0
비활성 백업 세션 수	0
백업 파일 DB 크기	0
연결 수	0

그림 218 분산 세션 서버 통계

26.3 중앙집중식 세션 서버

26.3.1 중앙집중식 세션 서버 기본 설정

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 기본 설정

이 설정은 중앙집중식 세션 서버를 가동 시키기 위한 설정이다. 세션 서버를 가동하기 위해서는 세션 관리자를 적어도 하나이상 가지고 있어야 한다.

1. 노드 트리에서 중앙집중식 세션서버를 클릭한다.
2. 하위 탭에서 기본 설정을 선택한다.
3. Resolution 을 입력한다. Resolution 은 세션 데이터의 무효화여부를 검사하는 주기를 설정한다.
4. 입력이 완료되면 확인버튼을 클릭한다.

26.3.2 세션 관리자 추가

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 새 세션 서버 매니저 생성

1. 노드 트리에서 중앙집중식 세션 서버를 클릭한다.
2. 하위 탭에서 세션 관리자를 선택한다.
3. 세션 관리자 목록 아래에 있는 새 세션 관리자 생성을 클릭한다.
4. 세션 관리자의 일반 설정을 입력한다.

새 세션 관리자 생성 - 기본 설정

1. 기본 설정
2. 스토리지
3. 생성

Name session의 이름을 설정한다. jeus clustering system상에서 unique한 값을 지정해야한다.

고급 선택사항

Multi Session 이 설정은 두개 이상의 session-server가 연계하여 중앙 집중식 session clustering을 수행하게 하는 설정이다. clustering 규모가 커서 하나의 session-server로 session clustering을 수행하기에 부적절한 환경에서 이 설정을 이용한다.

Passivation To 밀리초 memory에 존재하는 session 객체를 일정시간 사용하지 않으면 삭제하고 대신 file-db에 저장된 객체를 사용하게 하는 설정이다.

Removal To 밀리초 file-db에 저장된 session 객체의 보존 기간을 지정한다.

Operation To 밀리초 session-manager와 WebContainer간 read operation에 적용될 timeout을 지정한다.(RMI를 사용할 경우에만 의미가 있다.)

Check To 밀리초 얼마만큼의 시간 간격으로 backup과정을 수행할지를 결정한다. 이 설정에 지정된 시간 주기로 update된 session 객체가 있는지를 조사하고 update된 session 객체가 존재하면 backup를 수행한다.

Backup Name Old session-manager의 backup으로 사용할 session-manager의 name을 지정한다.

Backup Trigger objects local session manager에서 session 객체의 update가 어느 정도 발생하였을 때 backup session manager로 update된 session 객체들을 backup할지를 결정한다. 이 설정에 지정된 횟수 만큼 local session manager의 session object update가 발생하면 backup를 수행한다.

[다음 >](#)

그림 219 세션 관리자 생성 - I 단계 기본 설정

5. 세션 관리자의 스토리지를 입력한다.

1. 기본 설정
2. 스토리지
3. 생성

새 세션 관리자 생성 - 스토리지

Min Hole 일정 시간 file-db를 운용하면 file의 크기가 필요이상 커지게 된다. 이 설정에 지정된 횟수 만큼 file I/O가 발생하면 file packing을 수행하여 필요이상 file 크기가 늘어나는 것을 막는다.

Packing Rate 일정 시간 file-db를 운용하면 file의 크기가 필요이상 커지게 된다. 현재 session 객체 갯수 대비 file I/O 횟수가 지정된 ratio를 넘어서면 file packing을 수행하여 필요이상 file 크기가 늘어나는 것을 막는다.

고급 선택사항

File Db Path file-db에 경로를 지정한다.

File Db Name file-db 이름을 지정한다.

< 이전 다음 >

그림 220 세션 관리자 생성 - 2 단계 스토리지 설정

6. 입력이 완료되면 생성 버튼을 클릭한다.
7. 노드 트리에 비활성화된 세션 매니저 아이콘(■)이 추가되고 세션 매니저 목록 페이지로 이동한다.

26.3.3 세션 관리자 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 새 세션 서버 매니저 삭제

1. 노드 트리에서 중앙집중식 세션 서버를 클릭한다.
2. 하위 탭에서 세션 관리자를 선택한다.
3. 세션 관리자 목록에서 삭제하고자 하는 세션 관리자를 선택한 후 ■를 클릭한다.
4. 세션 관리자가 목록과 노드트리에서 사라진다.

26.3.4 세션 관리자 기본 설정

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 세션 관리자 > 기본 설정

1. 노드 트리에서 중앙 집중식 세션 서버를 클릭한다.
2. 하위 탭에서 세션 관리자를 선택한다.
3. 세션 관리자 목록에서 수정하고자 하는 세션 관리자를 클릭한다.
4. 하위 탭에서 기본 설정을 선택한다.
5. 각 입력 항목을 입력한다. 각 입력 항목에 대한 설명은 26.3.2 세션 관리자 추가를 참조하기 바란다.
6. 입력이 완료되면 확인버튼을 클릭한다.

26.3.5 세션 관리자 스토리지 설정

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 세션 관리자 > 스토리지 설정

1. 노드 트리에서 중앙 집중식 세션 서버를 클릭한다.
2. 하위 탭에서 세션 관리자를 선택한다.
3. 세션 관리자 목록에서 수정하고자 하는 세션 관리자를 클릭한다.
4. 하위 탭에서 스토리지를 선택한다.
5. 각 입력 항목을 입력한다. 각 입력 항목에 대한 설명은 26.3.2 세션 관리자 추가를 참조하기 바란다.
6. 입력이 완료되면 확인버튼을 클릭한다.

26.3.6 세션 관리자 모니터링

JEUS 매니저 > JEUS 매니저 서비스 > 중앙집중식 세션 서버 > 세션 관리자 > 스토리지 설정

이 모니터링은 세션 관리자가 관리하는 세션의 수를 보여준다.

1. 노드 트리에서 중앙 집중식 세션 서버를 클릭한다.
2. 하위 탭에서 세션 매니저를 선택한다.

3. 세션 관리자 목록에서 모니터링하고자 하는 세션 관리자를 클릭 한다.
4. 하위 탭에서 통계를 선택한다.
5. 통계 정보가 출력된다.



그림 221 세션 관리자 통계

26.3.7 서블릿 엔진에 중앙집중식 세션 서버 연결 설정

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 설정 > 중앙집중식 세션 서버

서블릿 엔진에서 세션 서버를 이용하기 위해서는 세션서버에 대한 정보를 입력해야한다. 서블릿 엔진은 이 설정을 통해 세션 서버에 세션을 저장하게 된다.

1. 노드 트리에서 서블릿 엔진을 클릭한 후 세션탭을 클릭한다.
2. 하위 탭에서 중앙 집중식 세션 서버를 선택한다.
3. 각 입력 항목을 입력한다.
4. 입력이 완료되면 확인버튼을 클릭한다.

설정

세션

기본 설정

분산 세션 서버

▶ 중앙 집중식 세션 서버

컨텍스트 그룹

분배

서블릿 엔진 세션 - 중앙 집중식 세션 서버

Server Name *

이 element는 이 Context group에서 Session Server를 사용하기 위해 Session Server 이름(JNDI export name)을 설정한다. 그 export 이름은 JEUSMain.xml에 <session-manager>< name >에 있는 값을 설정한다.

Max *

이 element는 Session Server와의 커넥션 풀을 유지하는 최대 커넥션 개수를 구성한다.

Min

이 element는 Session Server와의 커넥션 풀을 유지하는 초기 최소 커넥션 개수를 설정한다.

Step 1

이 element는 적어도 하나의 새로운 커넥션 생성이 필요할 때 Session Server와의 커넥션 풀에서 추가해주는 커넥션 개수를 설정한다.

Get Connection Timeout 10000

새 커넥션이 사용 가능하게 되는 걸 기다리는 최대 시간.

Backup Server Name

Primary Session Server가 문제가 생겼을 시에 사용할 Backup Session Server의 이름.

Connect Timeout 60000

커넥션이 처음 생성될 때 하나의 새로운 커넥션이 Session Server와 연결이 되는 걸 기다려야 하는 최대 시간.

Read Timeout 120000

Session Server로부터 데이터를 읽어오기 위해 기다리는 최대 시간.

확인 **취소** **재설정**

그림 222 서블릿 엔진에 중앙집중식 세션 서버 설정

26.3.8 중앙집중식 세션 서버 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 세션 컨테이너

세션 서버의 통계는 세션 컨테이너를 통해서 한다. 세션 컨테이너는 세션 객체를 관리하는 서블릿 엔진 내부의 객체로서 세션 설정에 따라 나타나는 위치가 다르다. 세션 클러스터링 설정에서 shared 를 “true”로 하였을 경우에는 서블릿 엔진 밑에 나타나며 그렇지 않을 경우에는 각 컨텍스트 그룹 밑에 나타나게 된다.

1. 노드 트리에서 세션 컨테이너를 클릭한다.
2. 통계 정보가 출력된다.



그림 223 중앙집중식 세션 서버 통계

26.4 로컬 세션 서버

분산이나 중앙집중식 세션 서버 설정을 하지 않을 경우에는 서블릿 엔진 자체 내에서 세션을 관리한다. 로컬 세션 서버의 경우 별도의 설정이 없으므로 이 절에서는 통계만 설명할 것이다.

26.4.1 로컬 세션 서버 통계

JEUS 매니저 > 엔진 컨테이너 개요 > 엔진 컨테이너 > 엔진 개요 > 서블릿 엔진 > 세션 컨테이너

1. 노드 트리에서 세션 컨테이너를 클릭한다.
2. 통계 정보가 출력된다.



그림 224 로컬 세션 서버 통계

27 JNLP 서비스

JNLP는 Java Network Launching Protocol의 약자로 소프트웨어 컴포넌트의 배포 및 실행에 관한 프로토콜이다. JNLP를 통해서 어플리케이션이 J2EE 서버로 접속할 수 있는 환경이 자동으로 구성된다. JNLP에 대한 자세한 내용은 <http://java.sun.com/products/javawebstart/>를 참고하기 바란다.

웹 관리자에서는 JNLP 서비스를 하기 위한 환경 파일인 JNLPMain.xml을 설정하고 설정 내용을 변경할 수 있도록 도와준다.

주의 : 웹 관리자를 통해 JNLPMain.xml을 설정할 경우에는 반드시 `JEUS_HOME/config/node_name/JNLPServer/JNLPMain.xml`이 미리 있어야 한다. 해당 폴더(JNLPServer)나 JNLPMain.xml 파일이 없는 경우에는 오류가 발생한다.

27.1 JNLP 설정

27.1.1 JNLP 서비스 추가

JEUS 매니저 > JEUS 매니저 서비스 > JNLP > 개요

JNLP 서비스를 사용하기 위해 JNLPMain.xml을 설정하게 된다.

1. 노드 트리에서 **JNLP**를 클릭한다.

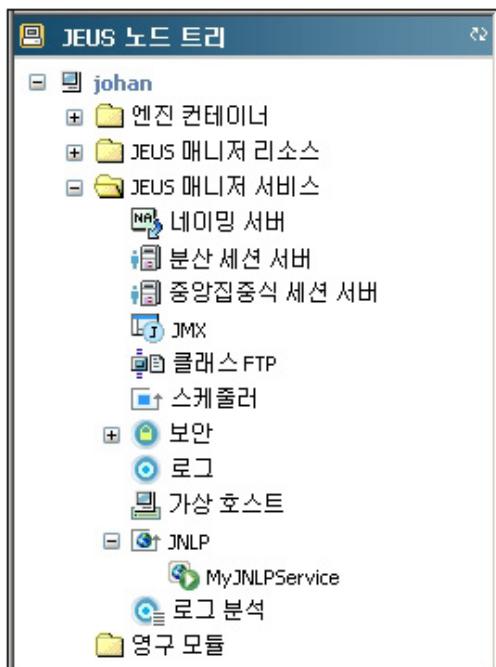


그림 225 노드 트리에서의 JNLP 서비스

2. 각 필드에 값을 넣고 추가 버튼을 누른다. 이 때 추가된 JNLP Resource 들은 테이블에 나타나게 된다.

JNLP 서비스

이름	경로	OS	구조	지역	버전	
HelloJesus/HelloJesus.jar	HelloJesus/HelloJesus.jar					
HelloJesus/HelloJesus.jnlp	HelloJesus/HelloJesus.jnlp					
Update/HelloJesus.jar	Update/HelloJesus10.jar				1.0	
Update/HelloJesus.jar	Update/HelloJesus20.jar				2.0	
Update/Update.jnlp	Update/Update.jnlp				1.0	

Rsc Name * JNLP Server에서 제공하는 resource의 이름. client에서는 해당 resource를 요청할 경우 rsc-name을 URL에 사용한다. 예) http://localhost:9744/jnlp>HelloJesus/rsc-name

Rsc Path * JNLP Server에서 제공하는 resource 파일에 대한 실제 물리적인 경로정보를 지정한다. 예) <rsc-path>c:\jeus50\webhome\client_home\HelloJesus\HelloJesus.jar</rsc-path>

Os Name resource가 고려해야 할 os를 지정한다. 이 값이 JVM의 시스템 파라미터 os.name의 prefix라면 resource를 사용할 수 있다. 만약 이 속성이 지정되어 있지 않다면 모든 OS에서 사용할 수 있다. 이 값은 Java의 시스템 프로퍼티인 os.name과 일치해야 한다. <os-name>, <arch>, <locale> 값을 여러 개 넣을 때는 공백으로 구별한다. 만약 공백이 값의 일부로 사용되면 공백 앞에 ""를 넣는다. 예를 들면 "Windows\95 Windows\2000"은 "Windows 95"와 "Windows 2000"을 지원한다는 의미이다. 여기서 "95"와 "2000"앞에 공백을 넣기 위해서 ""가 사용되었다.

Arch resource 항목이 사용되어야 하는 하드웨어 구조를 지정한다. 이 속성 값이 시스템 파라미터 os.arch의 시작부분과 같으면 resource 항목은 사용할 수 있다. 만약 이 속성이 지정되어 있지 않다면 모든 하드웨어에서 사용할 수 있다. 이 값은 <os-name>과 같은 형식으로 기술한다.

Locale 항목이 특정 지역에서 의존하는 resource임을 지정한다. 만약 이 값이 지정되어 있으면 resource항목은 지정된 locale 정보를 기본값으로 가지고 있는 JNLP 클라이언트에서만 사용할 수 있다. 이 값이 지정되어 있지 않으면 모든 클라이언트에서 사용 가능하다. 이 값은 <os-name>과 같은 형식으로 기술한다.

Version 이 자원의 버전을 지정한다. 클라이언트 시스템이 원하는 버전과 일치할 경우에만 이 자원을 사용할 수 있고 지정된 값이 없는 경우 모든 클라이언트 시스템에서 사용 가능하다.

추가

그림 226 JNLP 서비스 추가

27.1.2 JNLP 서비스 설정 변경

JEUS 매니저 > JEUS 매니저 서비스 > JNLP > 설정

JNLP 서비스 설정 내용을 변경하게 된다.

- 노드 트리에서 **JNLP – JNLP 서비스**를 클릭한다. 또는 JNLP 개요에서 테이블에 나타나는 JNLP 서비스의 이름을 클릭한다.

이름	경로	OS	구조	지역	버전	
HelloJesus/HelloJesus.jar	HelloJesus/HelloJesus.jar					
HelloJesus/HelloJesus.jnlp	HelloJesus/HelloJesus.jnlp					
Update/HelloJesus.jar	Update/HelloJesus10.jar				1.0	
Update/HelloJesus.jar	Update/HelloJesus20.jar				2.0	
Update/Update.jnlp	Update/Update.jnlp				1.0	

그림 227 생성된 JNLP 서비스

- 각 필드의 변경될 값을 넣고 확인 버튼을 누른다.

JNLP 서비스 설정

Rsc Name * JNLPServer에서 제공하는 resource의 이름. client에서는 해당 resource를 요청할 경우 rsc-name을 URL에 사용한다. 예) <http://localhost:9744/jnlp>HelloJeus/rsc-name>

Rsc Path * JNLPServer에서 제공하는 resource 파일에 대한 실제 물리적인 경로정보를 지정한다. 예) <rsc-path>c:\jeus50\webhome\client_home\HelloJeus\HelloJeus.jar</rsc-path>

Os Name resource가 고려해야 할 os를 지정한다. 이 값이 JVM의 시스템 파라미터 os.name의 prefix라면 resource를 사용할 수 있다. 만약 이 속성이 지정되어 있지 않다면 모든 OS에서 사용할 수 있다. 이 값은 Java의 시스템 프로퍼티인 os.name과 일치해야 한다. <os-name>, <arch>, <locale> 값을 여러 개 넣을 때는 공백으로 구별한다. 만약 공백이 값의 일부로 사용되면 공백 앞에 "\"를 넣는다. 예를 들면 "Windows\ 95 Windows\ 2000"은 "Windows 95"와 "Windows 2000"을 지원한다는 의미이다. 여기서 "95"와 "2000"앞에 공백을 넣기 위해서 "\"가 사용되었다.

Arch resource 항목이 사용되며 하는 하드웨어 구조를 지정한다. 이 속성 값이 시스템 파라미터 os.arch의 시작부분과 같으면 resource 항목은 사용할 수 있다. 만약 이 속성이 지정되어 있지 않다면 모든 하드웨어에서 사용할 수 있다. 이 값은 <os-name>과 같은 형식으로 기술한다.

Locale 항목이 특정 지역에서 의존하는 resource임을 지정한다. 만약 이 값이 지정되어 있으면 resource항목은 지정된 locale 정보를 기본값으로 가지고 있는 JNLP 클라이언트에서만 사용할 수 있다. 이 값이 지정되어 있지 않으면 모든 클라이언트에서 사용 가능하다. 이 값은 <os-name>과 같은 형식으로 기술한다.

Version 이 자원의 버전을 지정한다. 클라이언트 시스템이 원하는 버전과 일치할 경우에만 이 자원을 사용할 수 있고 지정된 값이 없는 경우 모든 클라이언트 시스템에서 사용 가능하다.

그림 228 JNLP 서비스 설정

27.1.3 JNLP 서비스 삭제

JEUS 메뉴 > JEUS 메뉴 서비스 > JNLP > 삭제

JNLP 서비스를 삭제한다.

1. 노드 트리에서 **JNLP** 클릭한다. 테이블에 나타나는 JNLP 서비스 중 해당 리소스의 삭제 아이콘을 클릭한다.
2. 해당 리소스가 삭제되어 테이블에서 사라진다.

28 로그 분석 서비스

28.1 소개

로그 분석 서비스는 JEUS 내에서 제공하는 Log service에 대한 분석 서비스를 제공한다.

JEUS에서 제공하는 로그 서비스에는 Error Log Filtering Service와 Access Log Analysis Service, Log Message StackTrace Service, Off-Line Log Analysis Service가 있다.

Error Log Filtering Service는 JEUS 내에 존재하는 모듈에서 남기는 ErrorLog Message를 추려서 살펴볼 수 있도록 해 준다.

Access Log Analysis Service는 AccessFormatter에 의해서 존재하는 데이터에 대해서 분석한 통계 정보를 살펴 볼 수 있다.

Log Message StackTrace Service는 Runtime 시에 현 logger에 적용되고 있는 log message를 통계할 수 있다.

Off-Line Log Analysis Service Off-Line 환경에서 JEUS가 제공하는 **Logger Type (ERROR_LOGGER, ACCESS_LOGGER)**을 준수하여 사용자가 등록한 Fomatter와 LogRecord로 남겨진 로그 메시지에 대해서 각각의 Error Log Filtering Service와 Access Log Analysis Service를 제공한다.

28.1.1 화면 구성

LogAnalysis Explorer 화면은 크게 *Logger Tree View*, *Handler View(or Off-Line View)*, *Status View*, *Rule View(or Report View or StackTracke View)* 4 부분으로 나누어 진다.

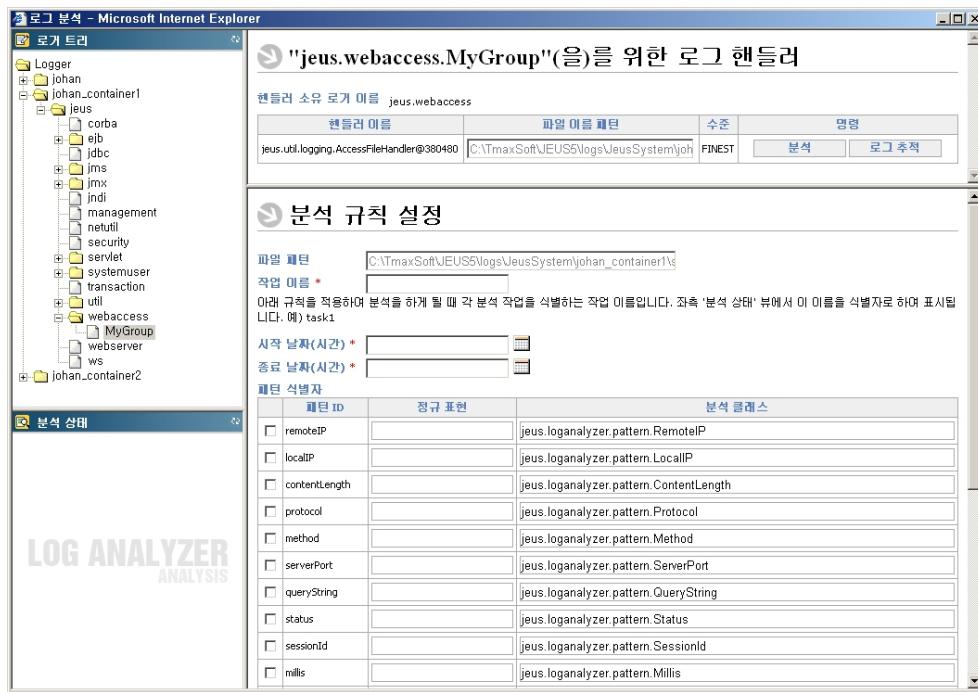


그림 229 LogAnalyzer Explorere Main Page

- **Logger Tree View:** 현재 Runtime에 JEUS 내에 적용되고 있는 Logger 목록을 보여주는 화면이다. 좌측 상단에 위치한 **생신버튼**()을 클릭하면 노드 목록이 생신된다.
- **Handler View:** Handler View는 Logger Tree View에서 특정 Logger 이름이 선택되면 해당 Logger에 적용되고 있는 File Handler에 대한 내용이 나타나는 화면이다. File Handler List 별로 Analysis 와 Stack Tracke Service 를 할 수 있어 화면 역시 Cluster View 와 마찬가지로 좌측 상단에 위치한 **생신 버튼**을 누르면 전체 트리가 생신된다. 트리에 사용된 아이콘들에 대한 설명은 [부록 B 아이콘](#)을 참조하라.
- **Rule View:** Rule View는 Handler View에서 특정 File Handler 정보의 Analysis 를 버튼을 클릭을 하면 나타나는 화면이다. 실제 Analysis 작업 을 이루어 지기 위한 분석 Rule 을 정의할 수 있다.
- **Status View:** Status View는 Analysis Task 의 실제 분석이 종료되었거나 분석 진행 중 혹은 분석 실패와 같은 상태 정보를 리스트 형태로 나타나는 화면이다. 이 화면 역시 Logger Tree View 와 마찬가지로 좌측 상단에 위치한 **생신 버튼**을 누르면 분석 Task 리스트가 생신된다.

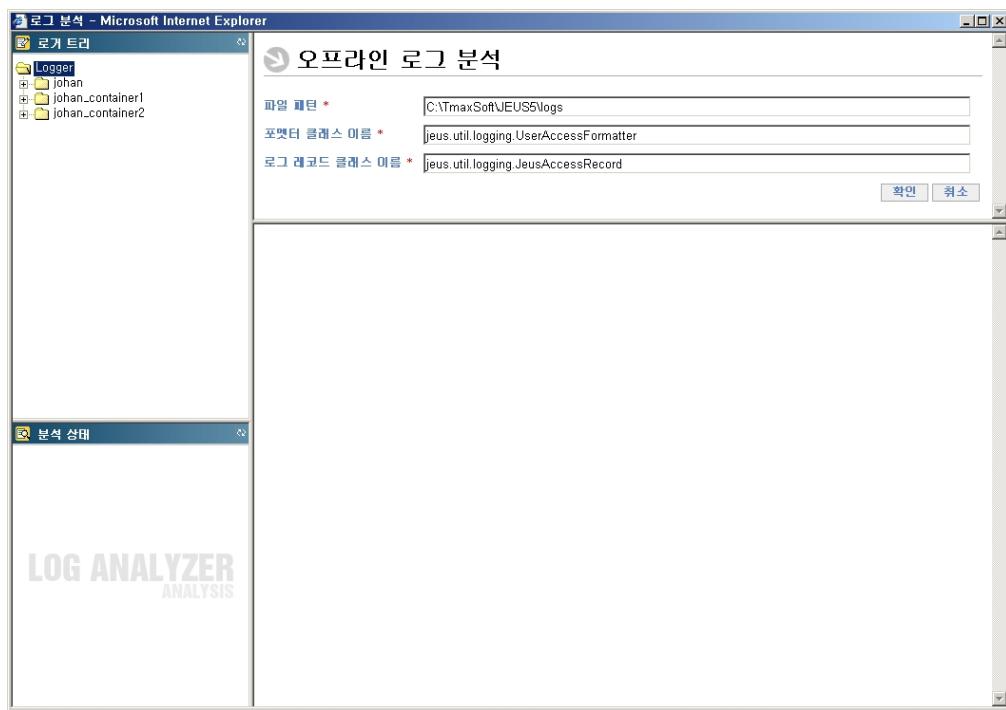


그림 230 Off-Line Setting View

- **Off-Line View:** Off-Line View는 Off-Line 환경에서 Logger Tree에서 최상의 노드(Logger)를 선택한 경우에는 Handler View에 사용자가 직접 File Handler 정보를 등록할 수 있는 화면이다.

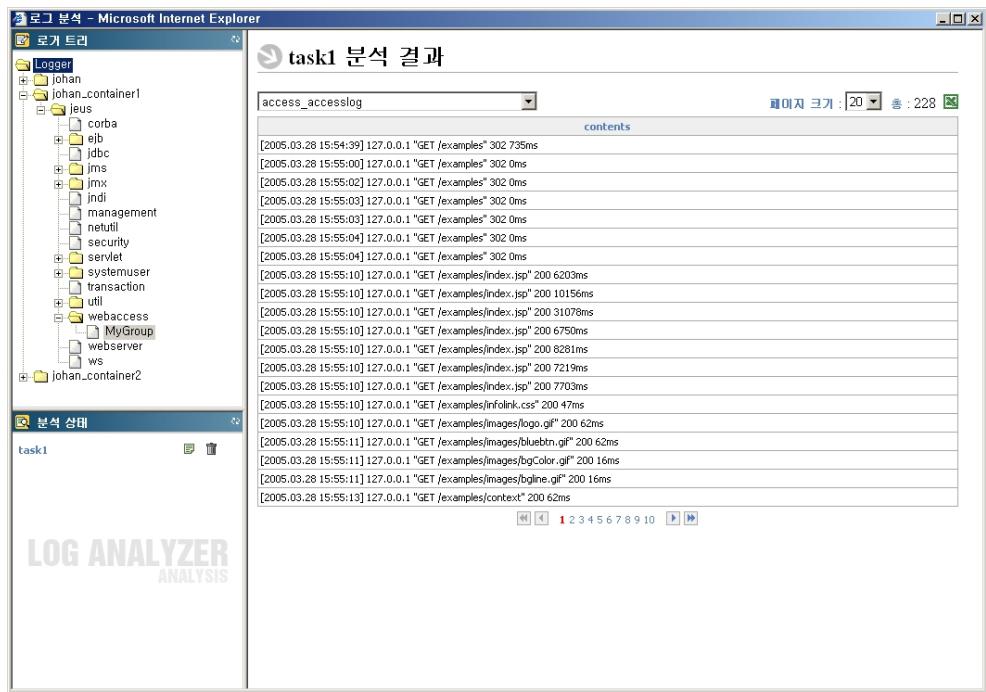


그림 231 Report View

- **Report View:** Status View에서 특정 Analysis Task를 선택한 경우 해당 분석 결과 내용을 나타내주는 화면이다.

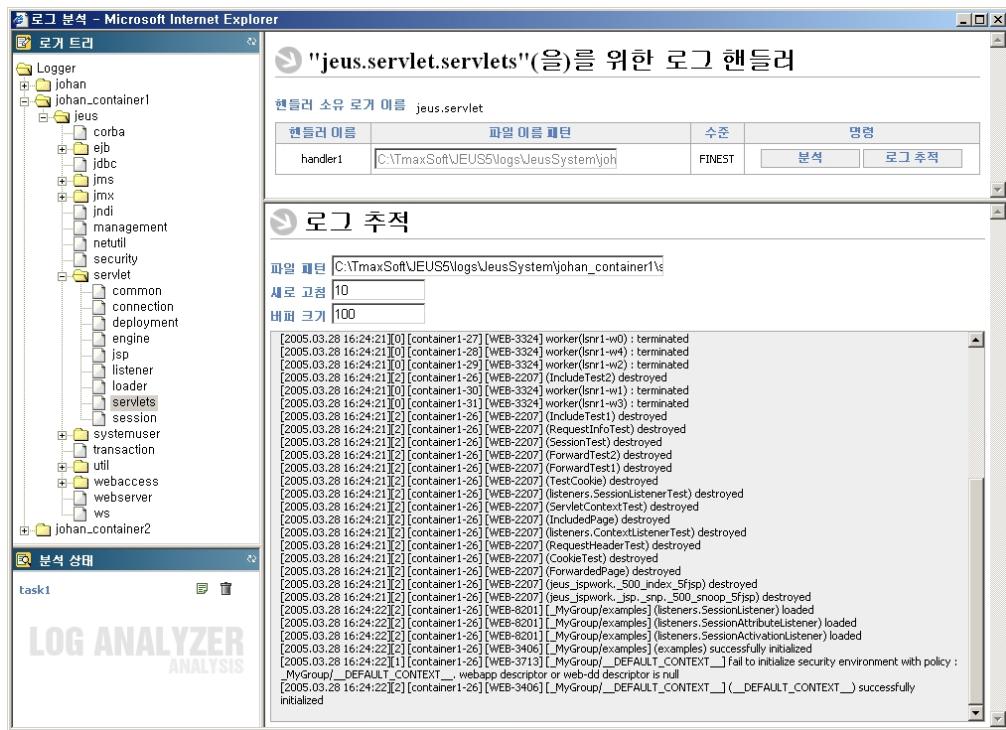


그림 232 StackTrace View

- **StackTrace View:** Handler View에서 특정 File Handler의 Stack Trace 버튼을 클릭하면 해당 패턴에 일치하는 로그 파일의 메시지 Stack Trace 나타나는 화면이다.

28.1.2 시작

JEUS 매니저 > JEUS 매니저 서비스 > 로그 분석 서비스 > 시작

JEUS 5에서는 Jeus Service type에 Log 분석을 할 수 있는 Mbean이 존재한다. JEUS Web Administration Tool에서 JEUS Service에 등록된 LogAnalyzer Service를 이용해서 JEUS Log 분석을 시작할 수 있다.

LogAnalyzer에서 Error Log Filtering Service, Access Log Analysis Service, Log Message StackTrace Service를 제공한다. 각각이 이용 방법에 차이가 있으므로 이것에 대해서는 해당 단계를 설명할 때 자세히 알아 볼 것이다.

- 노드 트리에서 **JEUS Services** 폴더를 클릭하면 JEUS Service type으로 등록된 LogAnalyzer Mbean 노드가 보인다.

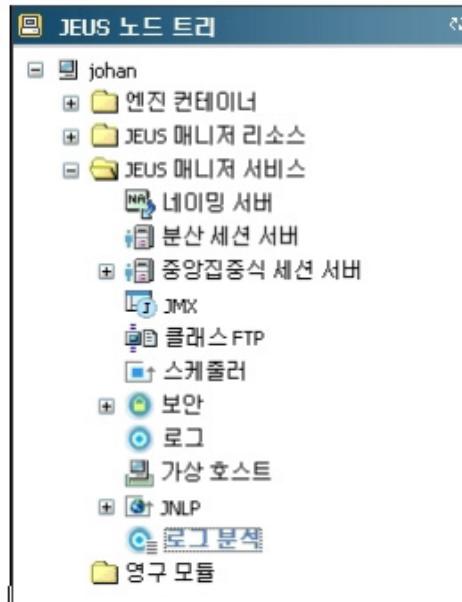


그림 233 LogAnalyzer 선택

- LogAnalyzer 노드 아이콘을 클릭하면 아래와 같이 LogAnalyzer window 팝업창이 뜬다.

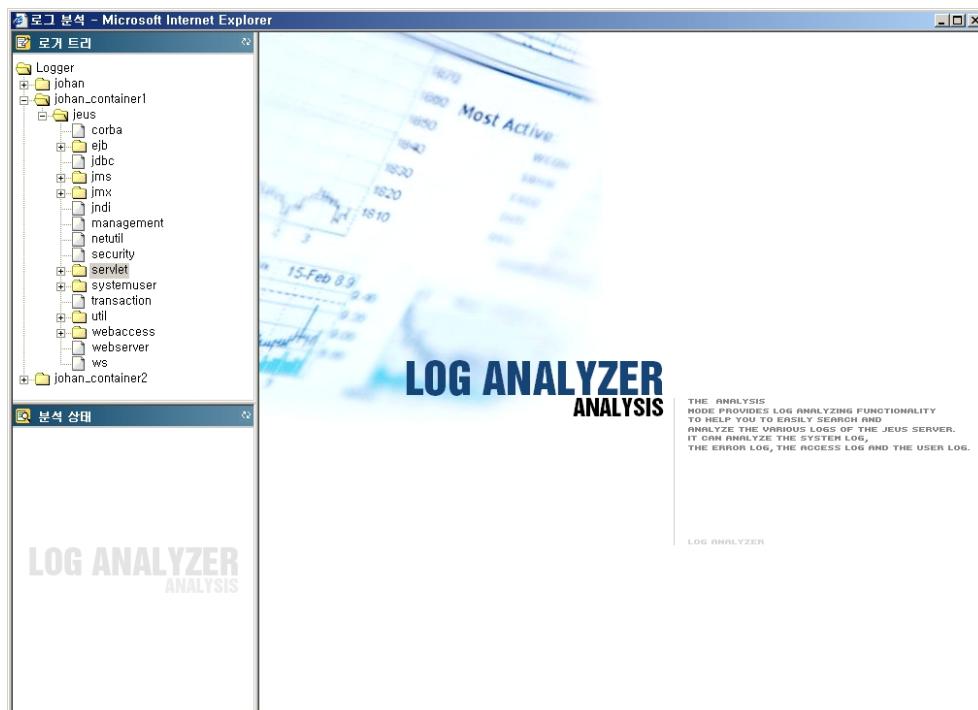


그림 234 LogAnalyzer Main Window

28.1.3 종료

LogAnalysis Explorer 를 close 한다.

28.2 로그분석

28.2.1 온라인 에러 로그분석

JEUS 매니저 > JEUS 매니저 서비스 > 로그 분석 서비스 >*Logger* >
ERROR_LOGGER TYPE 의 *Logger Name*

JEUS LogAnalyzer 에서 JEUS 에서 내에서 출력하는 Error Log Message 를 Filtering 해서 보여주는 서비스를 제공한다.

다음은 간단한 에러 로그에 대한 Filtering Service 를 단계별로 알아보도록 하자.

- Logger Tree 에서 **ERROR_LOGGER** type 의 logger name 을 선택한다. **Handler Explorer** 에서 선택된 logger 에 적용되고 있는 File Handler 리스트 정보를 보여준다. 현 Handler List 가 적용되고 있는 logger 에 대한 정보와 해당 logger 에 등록된 file handler 의 이름과 로그 파일 패턴, 로그 레벨 정보, Action 버튼들이 보여준다.

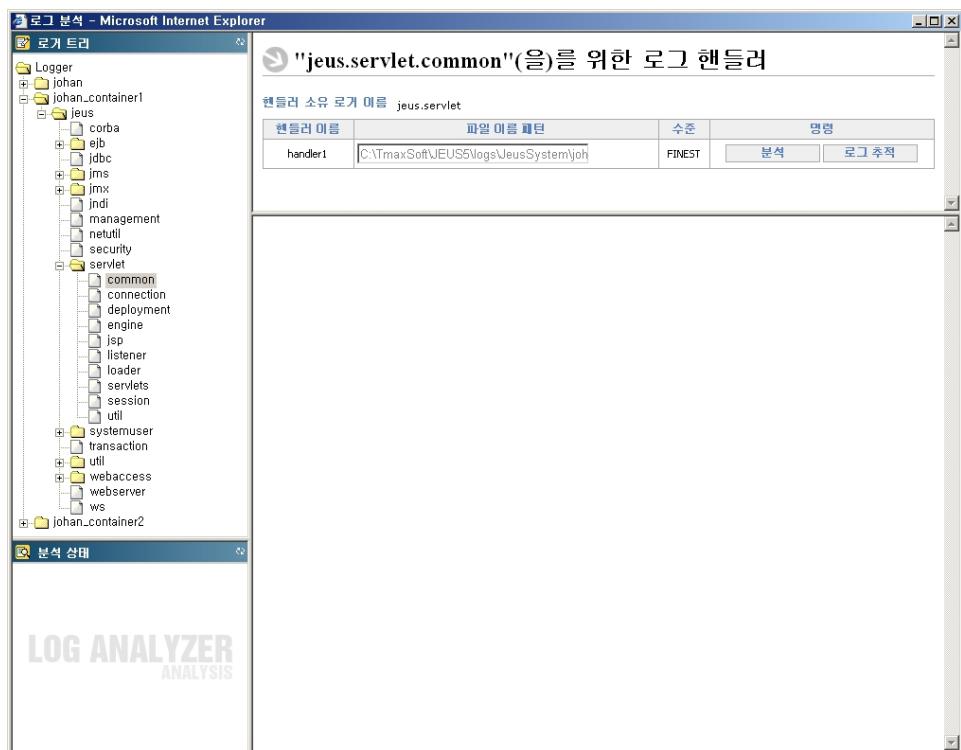


그림 235 Error Log Filter Service - 1 단계 Logger Name 선택

- 분석하고자 하는 File Handler의 **Analysis** 버튼을 선택한다. **Rule Explorer**에 선택한 File Handler에 적용되고 있는 Log File Pattern에 대한 일반적인 선택 사항을 입력한다.
 - **Task Name:** 로그 분석시 unique 한 단위 이름.
 - **Start Date:** 로그 파일의 분석 시작 날짜.
 - **End Date:** 로그 파일의 분석 종료 날짜.
 - **Pattern Identifiers:** 해당 File Handler에 적용되고 있는 메시지 Formatter에 대한 Regular Expression을 선택적으로 정의 할 수 있다. 로그 메시지 Format에서 Pattern Identifiers에서 정의된 Regular Expression에 매칭되는 로그 메시지만 Filtering 해서 보여준다. **ERROR_LOGGER** Type Pattern은 28.2.6의 **Log Format Pattern**를 참조한다.

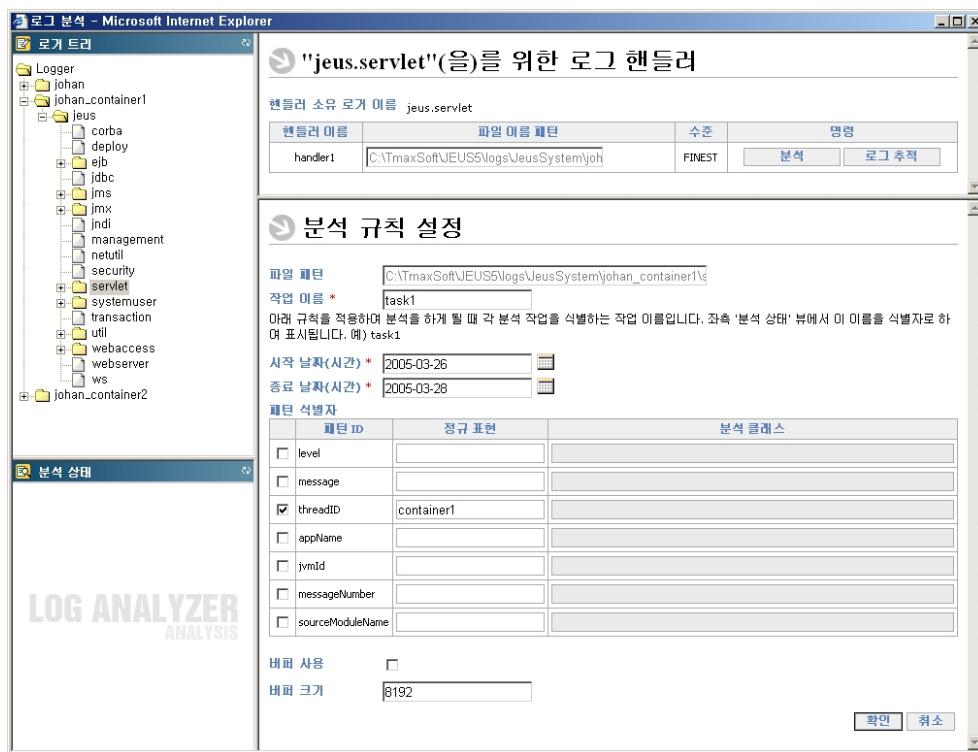


그림 236. Error Log Fileter Service - 2 단계/Analysis Rule 정의

- 하단의 버퍼 사용 유무와 버퍼 크기를 설정하여 분석시에 적용할 수 있다.
- Rule Explorer** 항목들을 적절히 설정한 후에 **확인** 버튼을 클릭한다.

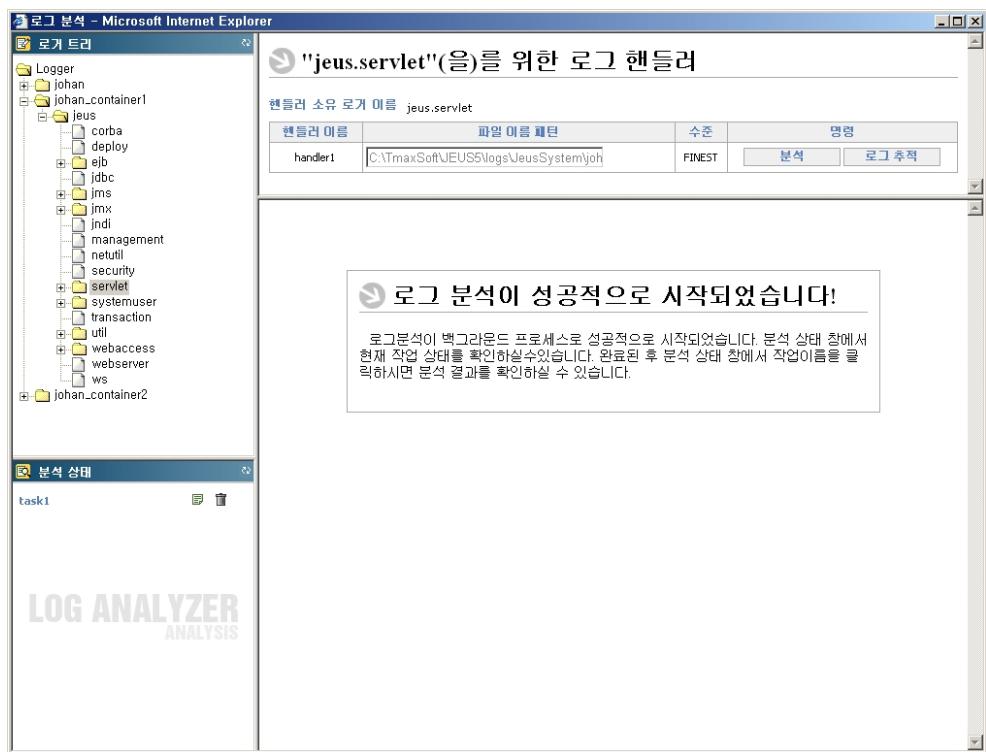


그림 237 Error Log Filter Service - 3 단계 Analysis Successfully Started

28.2.2 온라인 액세스 로그 분석

JEUS 매니저 > JEUS 매니저 서비스 > 로그 분석 서비스 >Logger > ACCESS_LOGGER TYPE 의 Logger Name

JEUS LogAnalyzer에서 WebContainer에서 내에서 출력하는 Access Log Message에 대한 Filtering 서비스와 대략적인 통계 정보 서비스를 제공한다. 다음은 간단한 Access 로그에 대한 서비스를 단계별로 알아보도록 하자.

- Logger Tree에서 **ACCESS_LOGGER** type의 logger name을 선택한다. Handler Explorer에서 **28.2.1 ERROR_LOGGER** type 분석 시와 동일한 정보를 보여준다.

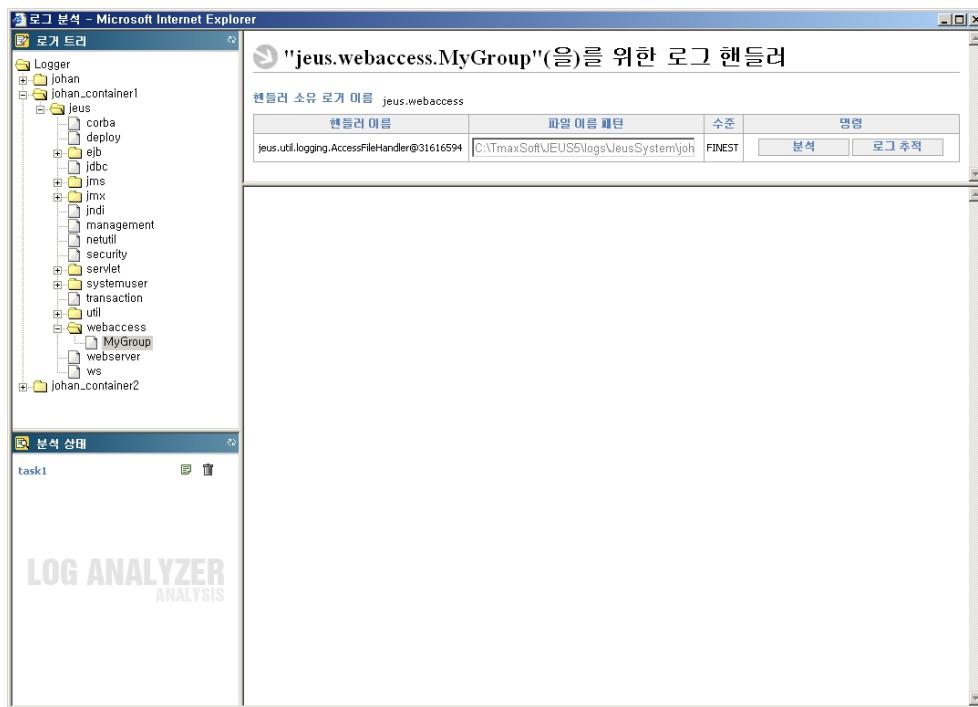


그림 238 Access Log Analysis Service - 1 단계 Logger Name 선택

- 분석하고자 하는 File Handler 의 **Analysis** 버튼을 선택한다. **Rule Explorer**에 선택한 File Handler에 적용되고 있는 Log File Pattern에 대한 일반적인 선택 사항을 입력한다.
 - **Task Name:** 로그 분석시 unique 한 단위 이름.
 - **Start Date:** 로그 파일의 분석 시작 날짜.
 - **End Date:** 로그 파일의 분석 종료 날짜.
 - **Pattern Identifiers:** 해당 File Handler에 적용되고 있는 메시지 Formatter에 대한 Regular Expression을 선택적으로 정의 할 수 있다. 로그 메시지 Format에서 Pattern Identifiers에서 정의된 Regular Expression에 매칭되는 로그 메시지만 Filtering 해서 보여준다. **ACCESS_LOGGER** Type Pattern 설명은 **28.2.6**을 참조한다.

패턴 ID	정규 표현	분석 클래스
remoteIP		jeus.loganalyzer.pattern.RemoteIP
localIP		jeus.loganalyzer.pattern.LocalIP
contentLength		jeus.loganalyzer.pattern.ContentLength
protocol		jeus.loganalyzer.pattern.Protocol
method		jeus.loganalyzer.pattern.Method
serverPort		jeus.loganalyzer.pattern.ServerPort
queryString		jeus.loganalyzer.pattern.QueryString
status		jeus.loganalyzer.pattern.Status
sessionId		jeus.loganalyzer.pattern.SessionId
millis		jeus.loganalyzer.pattern.Millis
remoteUser		jeus.loganalyzer.pattern.RemoteUser
requestURL		jeus.loganalyzer.pattern.RequestURL
serverName		jeus.loganalyzer.pattern.ServerName
processingTime		jeus.loganalyzer.pattern.ProcessingTime

그림 239 Access Log Analysis Service - 2 단계/Analysis Rule 정의

- 위의 Statistic Class 항목에 나타나는 클래스 명은 실제 해당 Pattern에 대한 hit 수와 hit Rate 값에 대한 기본 통계치를 구현한 클래스 명이다. Default 값은 -Dloganalyzer.properties=webaccess.properties 경로 설정된 webaccess.properties 파일에서 정의된 값이다.

표 4 webaccess.properties sample

```

remoteIP=jeus.loganalyzer.pattern.RemoteIP
localIP=jeus.loganalyzer.pattern.LocalIP
contentLength=jeus.loganalyzer.pattern.ContentLength
protocol=jeus.loganalyzer.pattern.Protocol
remoteUser=jeus.loganalyzer.pattern.RemoteUser
serverName=jeus.loganalyzer.pattern.ServerName
serverPort=jeus.loganalyzer.pattern.ServerPort
method=jeus.loganalyzer.pattern.Method
status=jeus.loganalyzer.pattern.Status
queryString=jeus.loganalyzer.pattern.QueryString
requestURL=jeus.loganalyzer.pattern.RequestURL
sessionId=jeus.loganalyzer.pattern.SessionId
processingTime=jeus.loganalyzer.pattern.ProcessingTime

```

- Rule Explorer 항목들을 적절히 설정한다. Access Log Pattern ID 중에 headerValues, cookieValue, requestAttributes, sessionAttributes 속성은 하

나 이상을 로그 포맷으로 정의가 가능하기 때문에 Expression 과 Statisitic Class 명을 순차적으로 아래와 같은 형태로 설정 해야 한다.

<input checked="" type="checkbox"/> headerValues	Content-Type=application/	jeus.loganalyzer.pattern.ContentType
<input checked="" type="checkbox"/> cookieValues	JSESSIONID=dkNG*	jeus.loganalyzer.pattern.JSESSIONID
<input checked="" type="checkbox"/> requestAttributes	test1=a.a.*;test2=b.b.*	jeus.loganalyzer.pattern.Test1, jeus.loganalyzer.pattern.Test2
<input checked="" type="checkbox"/> sessionAttributes	sess1=c.c.*;sess2=d.d.*	jeus.loganalyzer.pattern.SESSION1, jeus.loganalyzer.pattern.SESSION2

그림 240 로그 포맷 $\%{Content-Type}i \%\{J\}c \%\{test1\}r \%\{test2\}r \%\{sess1\}s \%\{sess2\}s$
regular expression 정의

- 확인 버튼을 클릭한다.
- 이 후 결과 내용을 확인 하는 과정은 **28.2.1**에 설명한 것과 동일하므로 생략한다.

28.2.3 오프라인 에러-엑세스 로그 분석

JEUS 매니저 > JEUS 매니저 서비스 > 로그 분석 서비스 > Logger

Off-Line 환경에서 User 가 정의한 로그 포맷에 관하여 Log Analysis Service 를 제공한다.

- **Logger Tree Explorer**에서 최상의 root 인 Logger Node 를 클릭 한다.

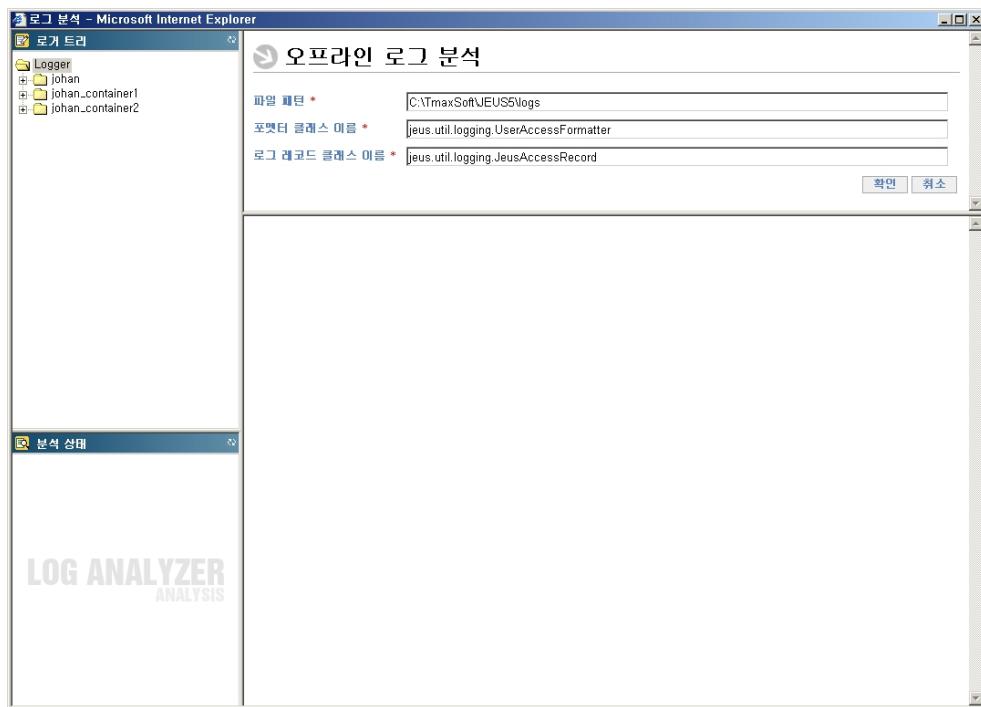


그림 241 Logger node 선택

- **Off-Line Settings Explorer**에 다음의 항목을 설정한 다음 확인 버튼을 누른다.
 - **File Pattern:** 사용자가 분석하고자 하는 File Name Pattern 을 정의한다.
 - **Formatter Class Name:** 사용자가 등록한 패키지 명을 포함한 Formatter 클래스이름을 정의한다. 정의된 Formatter 클래스는 java.util.logging.Formatter 를 상속하고, jeus.util.logging.JeusFormatter interface 를 implements 해야 한다.
 - **Log Record Class Name:** JEUS 엔진에 대한 여러 로그 분석을 하는 경우에는 jeus.util.logging.ErrorLogRecord 클래스를 상속하고 WebContainer 의 액세스 로그 분석을 하고자 하는 경우에는 jeus.util.logging.JeusAccessRecord 를 상속한 패키지 명을 포함한 클래스 이름을 정의한다.
- 이 후 과정은 Log Record 클래스가 jeus.util.logging.ErrorLogRecord 클래스를 상속하는 경우에는 **28.2.1**, jeus.util.logging.JeusAccessRecord 클래스를 상속하는 경우에는 **28.2.2** 과 동일하므로 생략한다.

28.2.4 에러 로그 분석 결과 보기

- Analysis Status Explore에서 Analysis status가 분석 종료인 task name list 중에 결과를 확인하고 싶은 에러 로그를 분석한 task name 명을 클릭 한다.

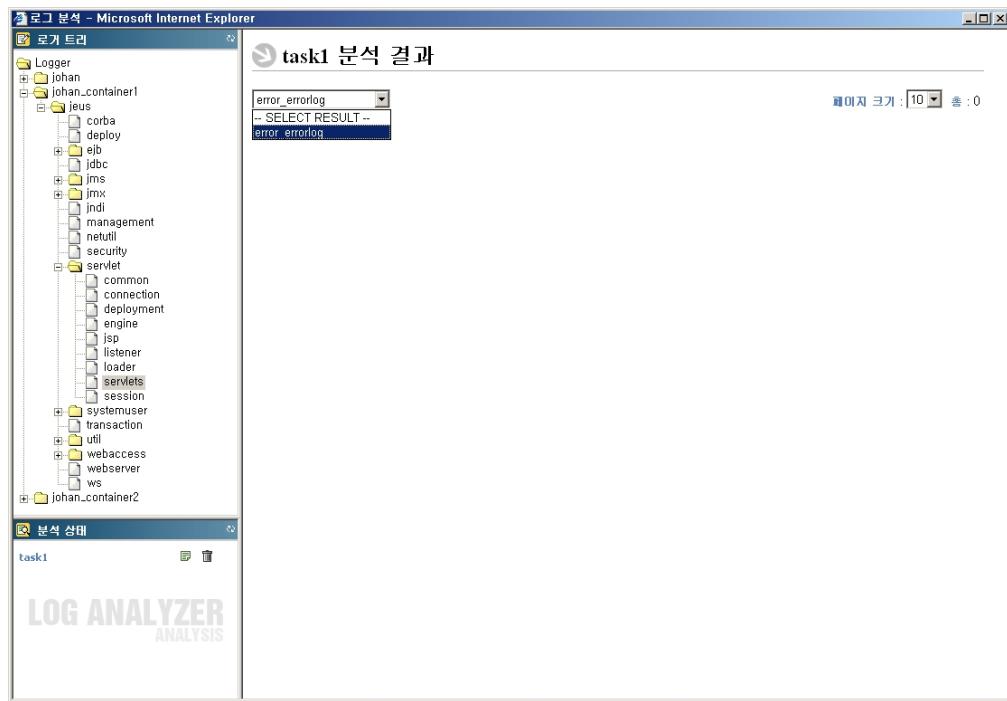


그림 242 Analysis Status Task Name List 선택

- Analysis Result Explore에서 분석 결과를 확인하고 싶은 Report 항목을 선택한다.
- Page Size:** 한 페이지에 display 할 결과 메시지 rows 를 정한다

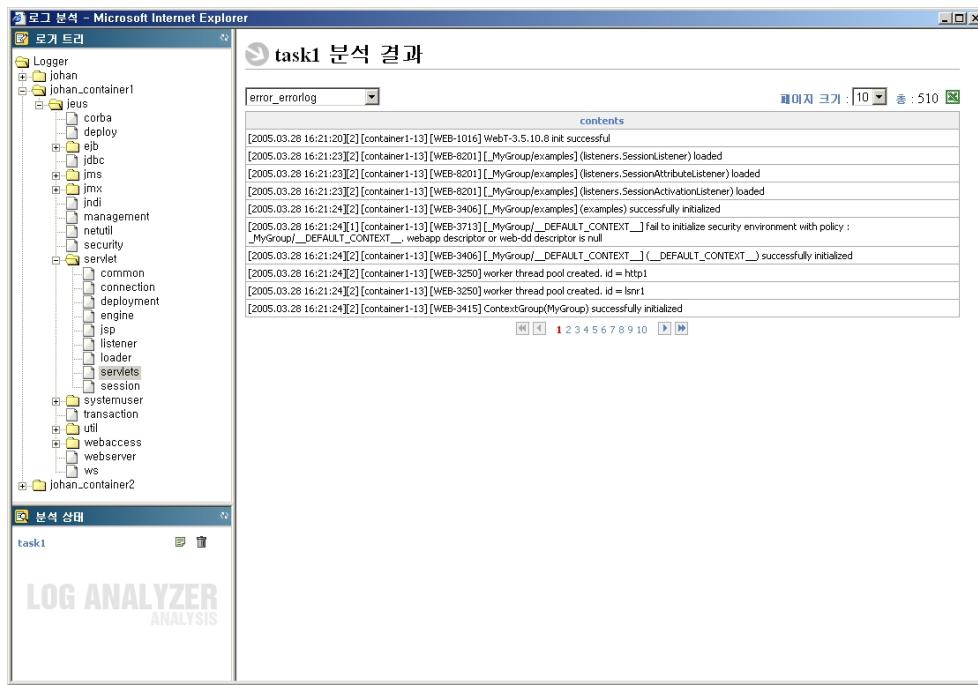


그림 243 Report Display

28.2.5 액세스 로그 분석 결과 보기

- **Status Explore**에서 Analysis status 가 분석 종료인 task name list 중에 결과를 확인하고 싶은 Access Log 파일을 분석한 task name 명을 클릭한 후 **Analysis Result Explore**에서 분석 결과를 확인하고 싶은 Report 항목을 선택한다

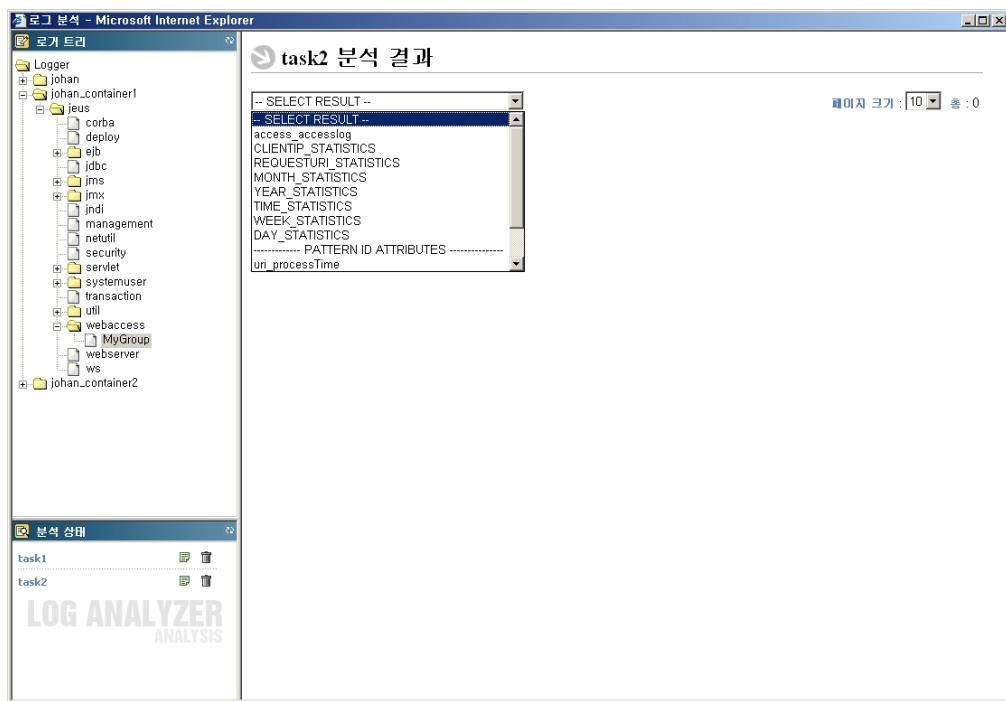


그림 244 Result Report List 선택

- **Analysis Result Explore**에서 분석 결과를 확인하고 싶은 Report 항목을 선택한다.
- Access Log 파일을 분석한 경우 Error Log 파일 분석시에 Filtering 분석 결과 외의 기본적인 통계 정보를 일부 제공해 준다.
 - **CLIENT_통계**: client ip 별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
 - **REQUEST_통계**: request url 별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
 - **YEAR_통계**: 년도별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
 - **MONTH_통계**: 월별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
 - **DAY_통계**: 날짜별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.

- **WEEK_통계**: 요일별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
- **TIME_통계**: 시간별로 분석된 로그파일의 히트수, 히트률, 요청수, 에러수, 페이지 성공률, 평균 수행시간 통계치 정보를 알려준다.
- **PATTERN_ID_ATTRIBUTES**: JEUS WebContainer에서 제공하는 혹은 사용자가 등록한 Pattern ID Attribute에 대한 hit 수와 hit rate 정보를 Pattern ID 별로 보여준다.

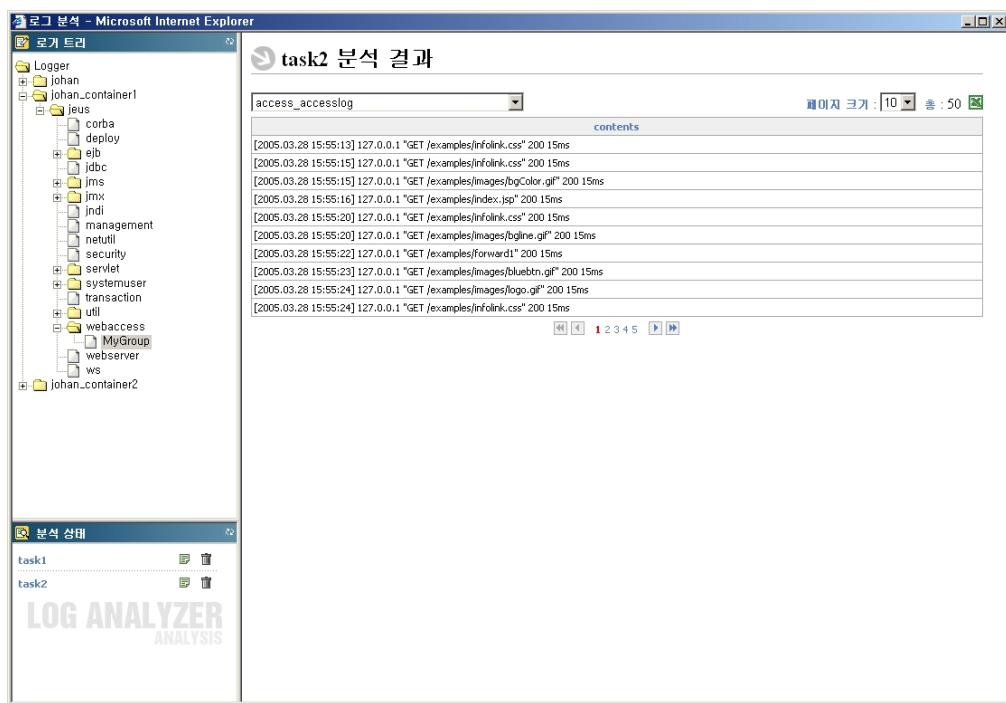


그림 245 Report Display

28.2.6 Log Format Pattern

- **LogAnalyzer**에서 로그 메시지 분석시에 해당 메시지에 대한 Filtering 기능을 하기위해서 각각의 적용되고 있는 Formatter에 어떤 attribute 요소가 있는지 파악해야 한다. **28.2.1**, **28.2.2**에서 살펴보았듯이 Rule Explorer에서 정의할 수 있는 Pattern Identifier는 다음의 크게 두 부분으로 나뉜다.
 - **Error Log Format**: JEUS 내에서 출력하는 여러 로그 메시지 Pattern.ID Attribute.

표 5 Error Log Format ↗ Pattern Attribute

Pattern Attribute	설명
level	handler 의 level. level 의 값으로는 logging API 의 level 인 SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST 가 올수 있다. 기본 설정은 INFO 이다
message	JEUS 내에서 정의된 로그 메시지
threadID	해당 메시지를 출력하는 고유 threadID
appName	WebContainer 에서 출력하는 메시지인 경우 해당 ContextGroup 과 context 별 구분을 하기 위한 이름
jvmId	로그를 출력하는 jvm 이 엔진 컨테이너 경우에는 container name, manager 인 경우에는 manager name 을 의미
messageNumber	로그 메시지의 고유 number
sourceModuleName	해당 로그 메시지를 출력한 JEUS 내의 모듈명

- **Access Log Format:** WebContainer 의 Access Log 메시지 Pattern ID Attribute..

표 6 Access Log Format of Pattern Attribute

Pattern Attribute	Pattern ID	설명
remoteIP	%a	Remote IP address

localIP	%A	로컬 IP address
contentLength	%b	Bytes sent, excluding HTTP headers, or '-' if zero
protocol	%H	Request protocol
remoteUser	%l	Remote logical username from identd (always returns '-')
method	%m	Request method (GET, POST, etc.)
serverPort	%p	local port on which this request was received, or '-' if none and it doesn't follows %U
queryString	%q	Query string (prepended with a '?' if it exists)
status	%s	HTTP status code of the response
sessionId	%S	User Session ID
millis	%t	Date and time, in Common Log Format
remoteUser	%u	Remote user that was authenticated (if any), else '-'
requestURL	%U	Requested URL path
serverName	%v	Local server name
processTime	%D	Time taken to process the request, in millis
headerValues	%{xxx}i	for incoming headers

cookieValue s	%{xxx}c	for a specific cookie
requestAttri butes	%{xxx}r	xxx is an attribute in the Servlet Request
sessionAttri butes	%{xxx}s	xxx is an attribute in the HttpSession
dateFormat	%{xxx}t	xxx is JDK standard DateFormat String which replaces default log date and time.

28.3 Log Message Monitoring Service

JEUS 매니저 > JEUS 매니저 서비스 > 로그 분석 서비스 > *Logger* > *Logger Name* 선택 > *Handler List Table*

Runtime 시에 JEUS 엔진에서 출력되고 있는 Log Message들을 브라우저를 통해서 실시간으로 통계 할 수 있는 서비스를 제공한다.

- Logger Tree Explorer에서 로그 메시지를 Runtime 시에 확인하고자 하는 Logger Name을 선택한다.



그림 246 StackTrace Service – 1 단계 Logger Name 선택

- 해당 Logger에 등록된 File Handler List 중에서 실시간으로 로그 메시지를 통계하고자 하는 File Handler의 Stack Trace 버튼을 클릭한다. 해당 File Handler에 적용되고 있는 Log File Pattern에 매칭되어 현재 로그 메시지가 출력되고 있는 로그 파일의 메시지가 현재 시점을 시작으로 주기적으로 메시지가 Stack Trace Explorer에 업데이트가 된다.

- **Refresh:** 주기적으로 Stack Trace Explorer 에 로그 메시지가 업데이트 되는 주기(초단위).
- **Buffer Size: Refresh** 주기별로 로그 메시지를 업데이트 할 때 적용되는 Buffer Size 값.

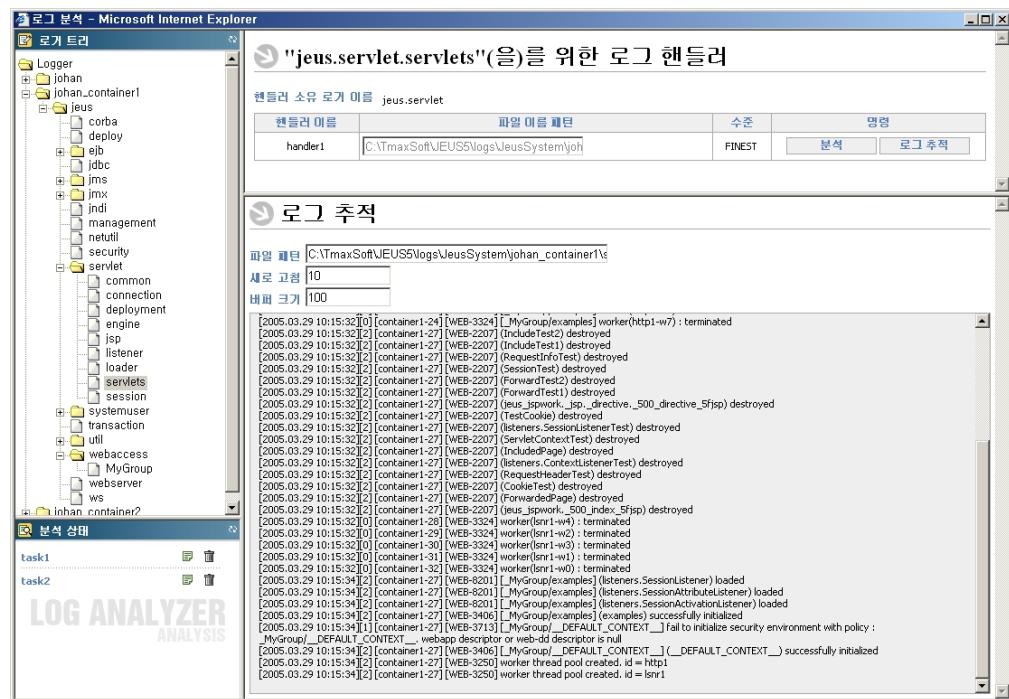


그림 247 StackTrace Service - 2 단계 FileHandler 와 StackTrace 버튼 선택

29 가상 호스트 서비스

JEUS 4.x 이전 버전에서는 하나의 IP Address에 하나의 노드만이 있을 수 있었다. 보통 하나의 머신에 하나의 JEUS 서버가 있었다고 할 수 있다. 그러나 JEUS 5에서는 하나의 머신 혹은 하나의 IP Address에서 가상 호스트를 이용하여 여러 JEUS 서버를 운영 할 수 있다.

가상 호스트 서비스는 서블릿 엔진의 가상 호스트와 다른 것이므로 주의해야 한다. 여기서 가상 호스트란 가상 노드라고 보면 된다.

가상 호스트의 보다 자세한 설명은 JEUS Server 안내서를 참고하기 바란다.

29.1 가상 호스트 설정

29.1.1 가상 호스트 설정

JEUS 매니저 > JEUS 매니저 서비스 > 가상 호스트 > 설정

1. 노드 트리에서 **가상 호스트**를 클릭한다.
2. 설정 탭을 선택한다.
3. **Enable Virtual Host**를 선택한다. 이 값을 "true"로 설정하면 가상 호스트가 동작하며 그렇지 않은 경우는 동작하지 않는다. 이 설정은 다음 부트시에 적용된다.
4. 입력을 완료하면 확인버튼을 클릭한다.

29.2 가상 호스트 구성

29.2.1 가상 호스트 추가

JEUS 매니저 > JEUS 매니저 서비스 > 가상 호스트 > 설정 > 새 가상 호스트 추가

가상호스트를 추가하기 위해서는 JEUS_BASEPORT 와 가상의 호스트 이름을 정해야 한다. JEUS_BASEPORT 는 같은 머신에 위치한 다른 노드에서 사용하는 포트 범위(JEUS_BASEPORT + 14)를 제외한 포트를 사용하여야 한다. 추가된 값은 바로 사용가능하다.

1. 노드 트리에서 **가상 호스트**를 클릭 한다.
2. 설정 탭을 선택한다.
3. **가상 호스트**에 가상 호스트의 목록을 입력한다. 입력 포맷은 “실제이름: 포트 \$WHITE_SPACE\$ 가상이름” 이다. \$WHITE_SPACE\$는 공백이나 탭이다.
4. 입력이 완료되면 확인버튼을 클릭 한다.

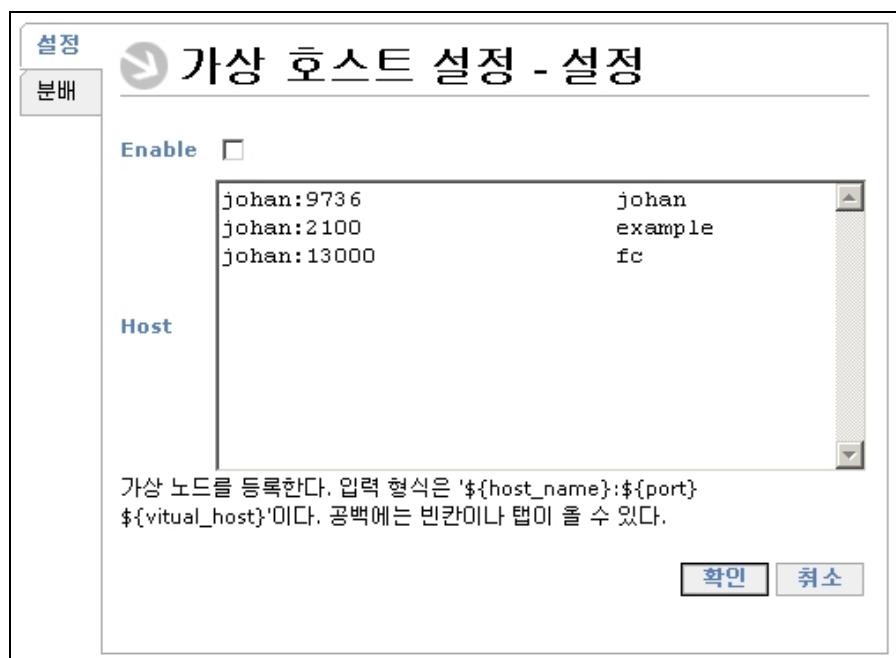


그림 248 가상 호스트 설정

29.2.2 가상 호스트 삭제

JEUS 매니저 > JEUS 매니저 서비스 > 가상 호스트 > 설정 > 가상 호스트 삭제

1. 노드 트리에서 **가상 호스트**를 클릭 한다.
2. 설정 탭을 선택한다.

3. 삭제하고자 하는 가상 호스트의 줄을 삭제한다.
4. 입력이 완료되면 확인버튼을 클릭한다.

참고 : 삭제된 값은 바로 영향을 미친다.

29.3 가상 호스트 분배

가상 호스트는 주로 클러스터링 환경에서 사용된다. 따라서 클러스터링내의 모든 노드들은 동일한 가상 호스트 목록을 가지고 있어야 한다. 가상 호스트의 분배는 가상 호스트 파일 - *JEUS_HOME/config/vhosts.xml* - 을 클러스터내의 모든 노드에 전송하여 동일한 설정을 갖게하는 기능이다.

29.3.1 가상 호스트 분배

JEUS 매니저 > JEUS 매니저 서비스 > 가상 호스트 > 설정 > 가상 호스트 분배

1. 노드 트리에서 **가상 호스트**를 클릭한다.
2. **분배**탭을 선택한다.
3. 분배를 하고자 하는 노드를 선택한다. 가능하면 모든 노드를 선택하는 것이 좋다.
4. 선택이 완료되면 **분배**버튼을 클릭한다.

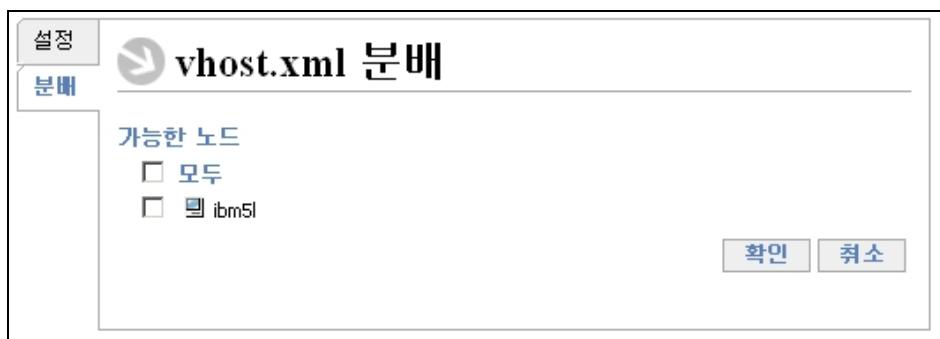


그림 249 가상 호스트 분배

30 장애 모니터링

시스템을 운영하다 보면 때때로 시스템의 구성요소중 일부에 장애(Trouble)가 발생할 수가 있다. 여기서 장애란 반드시 문제가 발생하여 시스템을 다운시키거나 서비스를 중지시키는 것만을 말하지 않는다. 여기서의 장애란 시스템의 어떤 한 구성요소가 사용중인 자원이 시스템 관리자가 관심을 가지는 수준이상으로 사용되는 경우를 말하며 일반적인 장애도 이 범주에 속한다고 할 수 있다.

예를 들어, JDBC의 데이터 소스의 최대 컨넥션 수를 30개로 설정하고, 25개 이상이 동시에 사용되면 경고(Warning) 수준, 28이 이상이면 위험(Fatal) 수준이라고 시스템 관리자가 설정하였다고 가정하자. 시스템을 운영중에 과부하로 인해서 26개의 컨넥션을 사용하였다면 경고수준으로서 컨넥션 풀의 재설정을 이용하여 컨택션의 수를 조절 한다든지 하는 조치를 취해야 할 것이다. 또한, 위험 수준에 도달한다면 시스템을 확장해서 부하를 분산하는 등의 조치를 취해야 할 것이다.

장애 모니터링 기능은 바로 이런 장애상황들을 통계해서 보다 시스템을 효율적으로 관리하기 위한 기능이다.

이번 장에서는 장애 모니터링을 하기 위한 설정 및 모니터링에 관해 알아 볼 것이다.

30.1 장애 모니터링 설정

장애 모니터링을 하기 위해서는 JEUS_HOME/config/tmonitor.xml에 장애 모니터링을 할 대상과 값을 입력해야한다.

다음의 예를 보자.

```
<<tmonitor.xml>>
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<trouble-monitor xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
    <mbean>
        <j2ee-type>JeusService</j2ee-type>
        <jeus-type>EJBEngine</jeus-type>
```

```

<constraint>
  <key>ejb_active_management_ratio</key>
  <value>0.8</value>
</constraint>
</mbean>
</trouble-monitor>

```

위의 예는 EJB 엔진의 Active Management에서 Block 된 Thread를 모니터링하기 위한 설정을 보여준다. 모니터링을 하는 MBean의 J2EE Type은 JeusService이며 JEUS Type은 EJB 엔진이고 관찰 대상은 ejb_active_management_ratio로서 0.8을 설정하였다. 이렇게 설정하게되면 EJB를 운영하던중에 Block 된 Thread의 비율이 80% 이상이 될 경우 Warning이라는 표시를 해주게 된다.

각 요소에 관한 자세한 설명은 를 참조하기 바란다.

30.2 장애 모니터링

30.2.1 장애 모니터링

JEUS 매니저 > 장애 모니터링

웹 관리자의 장애 모니터링은 SnapShot 개념을 이용한다. 즉, 노드 트리의 최상위 노드인 JEUS 매니저가 갱신되는 시점(노드를 변경하거나 컨텍스트 메뉴에서 새로고침을 클릭하거나 JEUS 노드 트리의 새로고침을 클릭할 경우)에 해당 노드의 모든 장애 상황을 검사하여 내부적으로 가지고 있다가 (SnapShot) 검색을 요청 하면 보여주는 방식이다.

1. JEUS 클러스터의 새로고침 버튼을 클릭하여 클러스터링 내의 어떤 노드에 장애가 발생하였는지 관찰한다.
2. 클러스터내의 어떤 노드에 장애가 발생하면 노드의 아이콘이 으로 변한다. 장애가 발생한 노드가 있으면 해당 노드를 클릭한다.
3. 노드 트리에서 JEUS 매니저의 컨텍스트 메뉴에서 장애 모니터를 선택한다.
4. 현재 문제가 발생한 상황에 대해서 출력이 된다.

이름	설명

J2EE Type	문제가 발생한 구성요소의 ObjectName 의 J2EE Type 이다.
JEUS Type	문제가 발생한 구성요소의 ObjectName 의 JEUS Type 이다.
이름	문제가 발생한 구성요소의 이름이다.
제약 키	모니터링 대상이 되는 설정의 키 이름이다.
제약 값	모니터링 대상이 되는 설정의 값이다.
원인	문제가 생긴 원인. 대체로 제약 값과 동일한 형식의 값으로서 현재 값을 나타낸다.
상태	현재 상태에 대한 레벨이 표시된다. SAFE, WARNING, FATAL 이 있으며 각각의 레벨은 구성요소마다 판단 기준이 다르므로 각 구성요소에 대한 안내서를 참고하기 바란다.

30.3 장애 모니터링 대상 목록

이 절에서는 현재 JEUS 에서 가능한 장애 통계 목록에 대해서 설명할 것이다.

30.3.1 EJB 엔진 ejb-active-management-ratio

표 7 장애 통계 목록: EJB 엔진 ejb-active-management-ratio

J2EE Type	JeusService	JEUS Type	EJB 엔진
Constraint Key	ejb-active-management-ratio		
Constraint Value	0.0~1.0		
Description	EJB 엔진의 active manager 에서 block 되는 thread 가 전체 thread 중에 차지하는 비율이 이 값 이상이면		

	WARNING 이 발생한다.
--	-----------------

30.3.2 EntityBean thread-max-warning-ratio

표 8 장애 통계 목록 : EntityBean thread-max-warning-ratio

J2EE Type	EntityBean	JEUS Type	
Constraint Key	thread-max-warning-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 WARNING 이 발생한다.		

30.3.3 EntityBean thread-max-fatal-ratio

표 9 장애 통계 목록 : EntityBean thread-max-fatal-ratio

J2EE Type	EntityBean	JEUS Type	
Constraint Key	thread-max-fatal-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 FATAL 이 발생한다.		

30.3.4 Stateless 세션 Bean thread-max-warning-ratio

표 10 장애 통계 목록 : Stateless 세션 Bean thread-max-warning-ratio

J2EE Type	Stateless 세션 Bean	JEUS Type	
Constraint Key	thread-max-warning-ratio		

Constraint Value	0.0~1.0
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 WARNING 이 발생한다.

30.3.5 Stateless 세션 Bean thread-max-fatal-ratio

표 11 장애 통계 목록 : Stateless 세션 Bean thread-max-fatal-ratio

J2EE Type	Stateless 세션 Bean	JEUS Type	
Constraint Key	thread-max-fatal-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 FATAL 이 발생한다.		

30.3.6 Stateful 세션 Bean thread-max-warning-ratio

표 12 장애 통계 목록 : Stateful 세션 Bean thread-max-warning-ratio

J2EE Type	Stateful 세션 Bean	JEUS Type	
Constraint Key	thread-max-warning-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 WARNING 이 발생한다.		

30.3.7 Stateful 세션 Bean thread-max-fatal-ratio

표 13 장애 통계 목록 : Stateful 세션 Bean thread-max-fatal-ratio

J2EE Type	Stateful 세션 Bean	JEUS Type	
Constraint Key	thread-max-fatal-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 FATAL 이 발생한다.		

30.3.8 MessageDrivenBean thread-max-warning-ratio

표 14 장애 통계 목록 : MessageDrivenBean thread-max-warning-ratio

J2EE Type	MessageDrivenBean	JEUS Type	
Constraint Key	thread-max-warning-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 WARNING 이 발생한다.		

30.3.9 MessageDrivenBean thread-max-fatal-ratio

표 15 장애 통계 목록 : MessageDrivenBean thread-max-fatal-ratio

J2EE Type	MessageDrivenBean	JEUS Type	
Constraint Key	thread-max-fatal-ratio		
Constraint Value	0.0~1.0		
Description	EJB 의 thread-max 설정에 대해 현재 운영되는 thread 의 갯수가 차지하는 비율이 이 값 이상이면 FATAL 이 발생한다.		

A tmonitor.xml XML 요소 참조

A.1 소개

본 부록의 참조는 장애 모니터링 설정 파일인 tmonitor.xml 의 모든 태그에 대해서 설명하고 있다. 이 파일의 Schema 파일은 “JEUS_HOME\config\xsds” 디렉토리의 “trouble-monitor.xsd” 파일이다.

본 레퍼런스는 3 부분으로 나눠져 있다.

1. **XML Schema/XML 트리:** XML 설정 파일의 모든 태그 리스트를 정리 했다. 각 노드의 형식은 다음과 같다.
 - a. 태그 레퍼런스로 빨리 찾아보기 위해서 각 태그마다 인덱스 번호(예: (11))를 붙여놓았다. 태그 레퍼런스에서는 이 번호 순서로 설명한다.
 - b. XML Schema에서 정의한 XML 태그명을 <**tag name**> 형식으로 표시한다.
 - c. XML Schema에서 정의한 Cardinality를 표시한다. “?” = 0 개나 1 개의 element, “+” = 1 개 이상의 element, “*” = 0 개 이상의 element, (기호가 없음) = 정확히 1 개의 element
 - d. 몇몇 태그에는 “P” 문자를 붙여놓았는데, 해당 태그는 성능에 관계되는 태그라는 것을 뜻한다. 이 태그는 설정을 튜닝할 때 사용된다.
2. **태그 레퍼런스:** 트리에 있는 각 XML 태그를 설명한다.
 - a. **Description:** 태그에 대한 간단한 설명.
 - b. **Value Description:** 입력하는 값과 타입.
 - c. **Value Type:** 값의 데이터 타입. 예) String
 - d. **Default Value:** 해당 XML 을 사용하지 않았을 때 기본적으로 사용되는 값

- e. **Defined values:** 이미 정해져 있는 값.
 - f. **Example:** 해당 XML 태그에 대한 예.
 - g. **Performance Recommendation:** 성능 향상을 위해서 추천하는 값.
 - h. **Child Elements:** 자신의 태그 안에 사용하는 태그.
3. **Example XML 파일:** “tmonitor.xml”에 대한 완전한 예제.

A.2 XML Schema/XML 트리

```
(1) <trouble-monitor>
(2) <mbean>*
    (3) <j2ee-type>
    (4) <jeus-type>?
    (5) <property>*
        (6) <key>
        (7) <value>
    (8) <constraint>*
        (9) <key>
    (10) <value>
```

A.3 Element Reference

```
(1) <trouble-monitor>
```

<i>Description</i>	장애 모니터링 설정 파일의 루트 요소. 장애 모니터링에 관한 모든 설정을 이 요소 아래에 설정한다.
--------------------	---

<i>Child Elements</i>	(2)mbean*
-----------------------	-----------

```
(2) <trouble-monitor> <mbean>
```

<i>Description</i>	장애 모니터링 대상이 되는 MBean 과 제약 값들을 설정하는 요소이다.
--------------------	--

<i>Child Elements</i>	(3)j2ee-type (4)jeus-type?
-----------------------	-------------------------------

(5) **property***(8) **constraint*****(3) <trouble-monitor> <mbean> <j2ee-type>**

Description MBean 의 j2eeType 키값을 입력한다. 모든 MBean 은 j2eeType 을 가지고 있다.

Value Type token

(4) <trouble-monitor> <mbean> <jeus-type>

Description MBean 의 jeusType 키값을 입력한다. 보통 j2eeType 이 JeusService 일 경우 jeusType 이 반드시 존재한다.

Value Type token

(5) <trouble-monitor> <mbean> <property>

Description MBean 의 j2eeType 과 jeusType 을 제외한 추가 속성을 지정한다. 이 값을 설정하면 특정 MBean 에 대한 설정이 가능하다.

Example 이 예제는 `tmax_servlet_engine1`이라는 WebEngine 에만 적용되는 설정이다. `<j2ee-type>JeusService</j2ee-type> <j2ee-type>WebEngine</j2ee-type> <property> <key>name</key> <value>tmax_servlet_engine1</value> </property>`

Child Elements (6) **key**
(7) **value**

(6) <trouble-monitor> <mbean> <property> <key>

Description property 의 key 값이다.

Value Type token

(7) <trouble-monitor> <mbean> <property> <value>

Description property 의 value 값이다.

Value Type token

(8) <trouble-monitor> <mbean> <constraint>

<i>Description</i>	모니터링할 제약사항을 입력한다.
<i>Example</i>	<constraint> <key> servlet-session-session-count </key> <value> 10000 </value> </constraint>
<i>Child Elements</i>	(9) key (10) value
(9) <trouble-monitor> <mbean> <constraint> < key >	
<i>Description</i>	property 의 key 값이다.
<i>Value Type</i>	token
(10) <trouble-monitor> <mbean> <constraint> < value >	
<i>Description</i>	property 의 value 값이다.
<i>Value Type</i>	token

A.4 tmonitor.xml 예제

```
<<tmonitor.xml>>
<?xml version="1.0" encoding="EUC-KR" standalone="yes"?>
<trouble-monitor xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
    <mbean>
        <j2ee-type>JeusService</j2ee-type>
        <jeus-type>EjbEngine</jeus-type>
        <constraint>
            <!-- EJB Engine 의 active manager 에서 block 되는 thread 가
                전체 thread 중에 차지하는 비율이 이 값 이상이면
                WARNING 이 발생한다. -->
            <key>ejb-active-management-ratio</key>
            <value>0.8</value>
        </constraint>
    </mbean>
    <mbean>
        <j2ee-type>EntityBean</j2ee-type>
        <constraint>
```

```

<!-- EJB 의 thread-max 설정에 대해 현재 운영되는
    thread 의 갯수가 차지하는 비율이 이 값 이상이면
    WARNING 이 발생한다. -->
<key>thread-max-warning-ratio</key>
<value>0.9</value>
</constraint>
<constraint>
    <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
        thread 의 갯수가 차지하는 비율이 이 값 이상이면
        FATAL 이 발생한다. -->
    <key>thread-max-fatal-ratio</key>
    <value>0.95</value>
</constraint>
</mbean>
<mbean>
    <j2ee-type>StatelessSessionBean</j2ee-type>
    <constraint>
        <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
            thread 의 갯수가 차지하는 비율이 이 값 이상이면
            WARNING 이 발생한다. -->
        <key>thread-max-warning-ratio</key>
        <value>0.9</value>
    </constraint>
    <constraint>
        <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
            thread 의 갯수가 차지하는 비율이 이 값 이상이면
            FATAL 이 발생한다. -->
        <key>thread-max-fatal-ratio</key>
        <value>0.95</value>
    </constraint>
</mbean>
<mbean>
    <j2ee-type>StatefulSessionBean</j2ee-type>
    <constraint>
        <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
            thread 의 갯수가 차지하는 비율이 이 값 이상이면
            WARNING 이 발생한다. -->
        <key>thread-max-warning-ratio</key>
        <value>0.9</value>
    </constraint>

```

```

</constraint>
<constraint>
    <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
        thread 의 갯수가 차지하는 비율이 이 값 이상이면
        FATAL 이 발생한다. -->
    <key>thread-max-fatal-ratio</key>
    <value>0.95</value>
</constraint>
</mbean>
<mbean>
    <j2ee-type>MessageDrivenBean</j2ee-type>
    <constraint>
        <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
            thread 의 갯수가 차지하는 비율이 이 값 이상이면
            WARNING 이 발생한다. -->
        <key>thread-max-warning-ratio</key>
        <value>0.9</value>
    </constraint>
    <constraint>
        <!-- EJB 의 thread-max 설정에 대해 현재 운영되는
            thread 의 갯수가 차지하는 비율이 이 값 이상이면
            FATAL 이 발생한다. -->
        <key>thread-max-fatal-ratio</key>
        <value>0.95</value>
    </constraint>
</mbean>
<mbean>
    <j2ee-type>JDBCDataSource</j2ee-type>
    <constraint>
        <!-- DataSource 에서 connection 을 얻을 때 waiting 하고
            있는 thread 가 connection pool 의 크기에 비해
            이 값 이상이 되면 WARNING 이 발생한다. -->
        <key>connection-waiting-warning-ratio</key>
        <value>0.8</value>
    </constraint>
    <constraint>
        <!-- DataSource 에서 connection 을 얻을 때 waiting 하고
            있는 thread 가 connection pool 의 크기에 비해
            이 값 이상이 되면 FATAL 이 발생한다. -->

```

```
<key>connection-waiting-fatal-ratio</key>
<value>0.9</value>
</constraint>
</mbean>
</trouble-monitor>
```


B 아이콘

아이콘	JEUS 객체	구분
	JEUS 노드 JEUS 매니저	
	풀더	
	엔진 컨테이너	
	EJB 엔진	
	서블릿 엔진	
	JMS 엔진	
	웹 서버 엔진	
	컨텍스트 그룹	
	웹 서버 리스너	
	세션 컨테이너 세션 서버	
	어플리케이션 모듈	모듈
	EJB 모듈	모듈
	웹 모듈	모듈

	리소스 아답터 모듈	모듈
	어플리케이션 클라이언트 모듈	모듈
	엔터티 빈	
	상태 세션 빈	
	무상태 세션 빈	
	메시지 빈	
	JCA 연결 풀	리소스
	JCA 관리 연결 풀	리소스
	JCA	리소스
	JDBC 데이터 소스	리소스
	클러스터 JDBC 데이터 소스	리소스
	자바메일	리소스
	URL 리소스	리소스
	WebT 데이터 소스	리소스
	클러스터 WebT 데이터 소스	리소스
	MQ 리소스	리소스
	JMS 클라이언트	리소스

	컨넥션 팩토리	리소스
	큐 데스터네이션	리소스
	토픽 데스터네이션	리소스
	클래스 FTP 서비스	서비스
	가상 호스트 서비스	서비스
	세션 관리자	
	보안	서비스
	스케줄러	서비스
	트랜잭션	서비스
	네이밍 서버	서비스
	JMX	서비스
	JNLP 서비스	서비스
	JNLP 리소스	서비스
	로그 서비스	서비스
	로그 분석	서비스
	리소스 아답터	
	보안 도메인	
	문서	

색 인

2

2 단계 배치 169

ㄱ

가능 186, 190
 가상 호스트 111
 가상 호스트 서비스 309
 계속 130

ㄴ

네이밍 서버 237
 노드 151
 노드 다운 59
 노드 부트 57
 노드 종료 59

ㄷ

도메인 255
 듀러블 서비스크라이버 143

ㄹ

라이프 사이클 70
 로그 서비스 245
 로그 수준 246, 253
 로깅 159
 로컬 세션 서버 281
 리소스 아답터 모듈 183

리소스 참조 71

리스너 50

ㅁ

모니터링 101

ㅂ

바인드 204, 215, 236
 배치 165
 배치 API 40
 백업 272
 별칭 157
 보안 255
 보안 서비스 256
 분산 세션 서버 272
 불가능 186, 190

ㅅ

사용자 로그 100, 114
 사용자 핸들러 251
 상호 운용 69
 서버 154
 서버 그룹 152
 서블릿 엔진 95
 세션 115
 세션 관리자 275, 278
 세션 추적 271
 세션 클러스터 101

세션 ID Cookie.....	116
소켓 핸들러	250
스케줄러	244
스케줄러 작업	244
스토리지	139, 278
시작	171

쓰

쓰래드 풀	52, 88, 127, 142, 193
-------------	-----------------------

○

애플리케이션 모듈	177
애플리케이션 클라이언트 모듈.....	184
에러 로그	53, 98, 138
엑세스 로그	98, 113, 138
엔진 컨테이너	63
엔진 컨테이너 리소스.....	185
엔진 컨테이너 서비스.....	197
연결 세션	218
영구적인 배치	174
웹 모듈	119, 180
웹 서버 리스너	121
웹 서버 엔진	149
웹 컨텍스트	170
응답 헤더	116
인코딩	112

ㅈ

자동 배치	68, 176
자동 배치 Check Interval.....	69
자동 배치 Path	69
자바 메일	221
장애 모니터링	53, 313
재설정	187

재시작.....	119, 183
절대 경로.....	166
정지.....	117, 129, 172, 181
정지 해제.....	118, 182
제거.....	176
제거.....	173
종료.....	118, 182
중앙집중식 세션 서버	275
지속성 설정	89

ㅋ

컨텍스트.....	109
컨텍스트 그룹.....	105
콘솔 핸들러.....	246
큐 데스터네이션.....	212
클라이언트.....	217
클래스 FTP.....	244
클러스터	142, 144, 145
클러스터 JDBC 데이터 소스	204
클러스터링	55

ㅌ

타이머 서비스.....	87
토파 데스터네이션	211
토파 컨넥션 팩토리	209
트랜잭션 관리자.....	192

ㅍ

파일 업로드.....	166
파일 핸들러	247
파일 DB	272, 274
표준 배치	165

○

화장	158, 159
활성 관리	84

A

Active	132
Active Management	84
AJP13 리스너	121
Alias	157
Apache 리스너	121
Application	175
application.xml	177
application-client.xml	184

B

Backlog	50
Blocked	132
Body	43

C

CGI	153
Class Path	68
Cluster View	42
Command Option	67
Consumer	217

D

Data Source Name	90
DB Vendor	89
DBMS	199, 233
DBStorage	139
Definition	156
Distribute EJBMain.xml	92
Distribute WEBMain.xml	103
Durable Subscriber	216

E

EJB 모듈	178
EJB 엔진	81
ejb-jar.xml	178
Enable User Notification	84
Engine Type	90
EXPLODED_COMPONENT	167
EXPLODED_EAR	167
Extension	158, 159

F

FAIL	162
FileStorage	140
Format	99

H

Header	43
Heap	76
HTML	153
HTTP 리스너	121
HTTP 아답터	239
HTTP 호출	85
HTTP port	86

J

J2EE 모듈	177
JAXB	40
JCA 리소스	189
JCA 리소스 연결	189
JCA 리소스 연결 풀	189
JDBC 데이터 소스	185, 199
JEUS 매니저	49
JEUS 매니저 서비스	237
jeus-client-dd.xml	184
jeus-connector-dd.xml	183

jeus-ejb-dd.xml	178
JEUSMain.xml 분배.....	60
jeus-web-dd.xml	180
JMS 리소스.....	195, 209
JMS 엔진.....	133
JMSMain.xml 분배	147
JMX	39
JMX 서버	239
JMXMP	240
JNDI 트리.....	53
JSP 엔진	114
JSV	153
JVM	75

L

Log Analysis Service.....	287
logging API.....	245

M

Main View	43
Max Blocked Thread.....	85
Max Idle Time	85
Max Redelivery Count.....	88
Minimum Delivery Interval	88
MLet	243

N

Navigation.....	43
Node View	43
Not Loaded	181
NOTREADY	162

P

Pattern ID.....	99
PHP	153
Policy	264

Principle	260
-----------------	-----

R

ra.xml	183
Ready	181
READY	162
Reconnection.....	132
Redelivery Interval.....	88
Resolution	84, 275
Restart	119
Resume.....	118, 130
Resync.....	187
RMI 아답터	241

S

Secure 리스너	121
Server Type	153, 156
Shrink	186, 191
SMTP 핸들러	248
SNMP 아답터	242
SSI.....	153
SSL.....	51
Subject.....	258
Suspend	117, 129, 181
SYSTEM_DOMAIN.....	255

T

Table Name	90
TCP 리스너	121
Terminate	118

U

URI.....	156
URL	86
URL 리소스.....	219
User Log.....	68

W

Waiting	132
web.xml	180
WebT 데이터 소스	191, 223, 233
WebT 데이터 소스 통계.....	191
WebtoB 리스너	121

WebToB 리스너 백업	125
Worker Thread	130

X

XA 트랜잭션 재시작	191
-------------------	-----