

# ***JEUS Builder 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 글라스타워 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 Builder 안내서

Document date: 2005-06-06

Manual release version: 3

Software Version: JEUS 5

# 차 례

|   |           |
|---|-----------|
| <b>1 소개.....</b>                        | <b>14</b> |
| 1.1 개요.....                             | 14        |
| 1.2 지원 모듈.....                          | 14        |
| 1.3 작업 영역.....                          | 15        |
| 1.4 제약 사항.....                          | 16        |
| 1.5 배치서술자 (Deployment Descriptor) ..... | 16        |
| 1.6 결론.....                             | 17        |
| <b>2 뷰(View) .....</b>                  | <b>18</b> |
| 2.1 모듈 트리 뷰(ModuleTree View) .....      | 19        |
| 2.1.1 트리노드 아이콘 이미지 .....                | 20        |
| 2.2 워크스페이스 뷰(Workspace View) .....      | 21        |
| 2.2.1 DD 설정 화면 .....                    | 21        |
| 2.2.2 Enterprise Bean 설정 화면.....        | 23        |
| 2.2.3 Additional Options 화면 .....       | 23        |
| 2.2.4 Basic / Expert 화면.....            | 24        |
| 2.2.5 입력 필수 사항 .....                    | 25        |
| 2.2.6 정보 의존성 .....                      | 26        |
| 2.3 메시지 뷰(Message View) .....           | 26        |
| 2.4 결론.....                             | 27        |
| <b>3 JEUS Builder 시작하기 .....</b>        | <b>28</b> |
| 3.1 실행하기.....                           | 28        |
| 3.2 모듈 열기.....                          | 29        |
| 3.2.1 배치 서술자 생성 .....                   | 29        |
| 3.3 모듈 저장하기.....                        | 30        |
| 3.4 모듈 파일 콘텐츠 다루기.....                  | 31        |

|          |                                  |           |
|----------|----------------------------------|-----------|
| 3.4.1    | Module Info .....                | 32        |
| 3.4.2    | File Contents.....               | 33        |
| 3.5      | 종료하기.....                        | 37        |
| 3.6      | 결론.....                          | 37        |
| <b>4</b> | <b>모듈 생성 .....</b>               | <b>38</b> |
| 4.1      | CMP 2.0 의 EJB 모듈 생성 .....        | 38        |
| 4.1.1    | 생성된 모듈과 관련 된 파일 .....            | 44        |
| 4.2      | Webservice 를 사용하는 WEB 모듈 생성..... | 52        |
| 4.2.1    | 생성된 모듈과 관련 된 파일 .....            | 67        |
| 4.3      | 결론.....                          | 90        |
|          | <b>색 인.....</b>                  | <b>93</b> |

## 그림 목차

|  |    |
|--|----|
| 그림 1 JEUS 시스템내의 J2EE Module.....                       | 14 |
| 그림 2 JEUS Builder 전체화면 .....                           | 18 |
| 그림 3 Application 타입의 모듈 트리 뷰.....                      | 19 |
| 그림 4 EJB 모듈 타입의 모듈 트리 뷰.....                           | 20 |
| 그림 5 WEB 모듈의 워크스페이스 뷰.....                             | 22 |
| 그림 6 EJB 모듈의 워크스페이스 뷰.....                             | 22 |
| 그림 7 Entity Bean 의 워크스페이스 뷰.....                       | 23 |
| 그림 8 Additional Options.....                           | 24 |
| 그림 9 EJB 모듈의 Basic 탭화면 .....                           | 25 |
| 그림 10 EJB 모듈의 Expert 탭화면 .....                         | 25 |
| 그림 11 필수입력항목 미입력시 메시지 대화상자.....                        | 26 |
| 그림 12 Message View 화면.....                             | 26 |
| 그림 13 JEUS Builder 초기 화면 .....                         | 28 |
| 그림 14 모듈 열기.....                                       | 29 |
| 그림 15 DD 파일 유효성검사 실패시 덮어쓰기 여부 결정 대화상자.....             | 30 |
| 그림 16 File Contents 대화상자 (Application 타입).....         | 32 |
| 그림 17 모듈파일입력 오류시 발생하는 오류메시지.....                       | 33 |
| 그림 18 모듈 클래스패스 대화상자(application type) .....            | 34 |
| 그림 19 내장모듈에 파일추가시 오류메시지.....                           | 36 |
| 그림 20 내장모듈 파일추가시 오류메시지.....                            | 36 |
| 그림 21 Application 모듈의 Extraction 모드에서의 삭제시 경고메시지 ..... | 36 |
| 그림 22 JEUS Builder 종료시 대화상자 .....                      | 37 |

|   |    |
|---|----|
| 그림 23 모듈 생성시 File Contents 대화상자를 포함한 JEUS Builder ..... | 39 |
| 그림 24 Archive Path 지정을 위한 대화상자.....                     | 40 |
| 그림 25 book20 EJB 모듈의 모듈정보.....                          | 40 |
| 그림 26 작업디렉토리 선택을 위한 대화상자.....                           | 41 |
| 그림 27 추가 대상 디렉토리.....                                   | 41 |
| 그림 28 EJB 모듈 생성시 추가할 클래스파일 선택.....                      | 42 |
| 그림 29 파일 추가후 Selected File Contents 화면.....             | 43 |
| 그림 30 파일추가후 전체화면.....                                   | 44 |
| 그림 31 JEUS Builder 초기화면 .....                           | 52 |
| 그림 32 File Contents 대화상자 초기화면 .....                     | 53 |
| 그림 33 모듈 저장위치 지정을 위한 대화상자.....                          | 54 |
| 그림 34 Module Info 설정 .....                              | 54 |
| 그림 35 추가 대상 파일 트리.....                                  | 55 |
| 그림 36 추가된 파일 트리.....                                    | 56 |
| 그림 37 파일 추가후 초기 화면.....                                 | 57 |
| 그림 38 servlet 추가 화면.....                                | 58 |
| 그림 39 servlet 추가 대화상자.....                              | 58 |
| 그림 40 init-param 입력 화면.....                             | 59 |
| 그림 41 servlet 추가후 모듈트리뷰와 워크스페이스뷰.....                   | 59 |
| 그림 42 web-app 탭에서 servlet-mapping 화면.....               | 60 |
| 그림 43 servlet-mapping 대화상자.....                         | 60 |
| 그림 44 setvlet-mapping table .....                       | 61 |
| 그림 45 jeus-web-dd 의 general .....                       | 61 |
| 그림 46 웹모듈의 webservice 설정 버튼.....                        | 62 |
| 그림 47 웹서비스 설정 초기화면.....                                 | 63 |
| 그림 48 webservcie-description 대화상자 .....                 | 64 |
| 그림 49 port-component 대화상자.....                          | 65 |

---

|   |    |
|---|----|
| 그림 50 jeus-webservice-dd 탭 화면 .....           | 66 |
| 그림 51 jeus-webservices-dd 의 service 대화상자..... | 66 |
| 그림 52 jeus-webservices-dd 의 port 대화상자 .....   | 67 |





## 표 목차

|     |                                   |    |
|-----|-----------------------------------|----|
| 표 1 | Deployment Descriptor Files ..... | 17 |
| 표 2 | 모듈트리뷰 아이콘 이미지.....                | 21 |

# 매뉴얼에 대해서

## 매뉴얼의 대상

본 매뉴얼은 JEUS Builder 툴을 이용해서 디플로이할 모듈을 조립/패키징을 원하는 사용자를 위한 것이다. 특히 배치서술자(DD:Deployment Descriptor)의 생성, 열람, 그리고 변경에 있어서 사용자에게 높은 생산성을 제공해 줄것이다.

## 매뉴얼의 전제 조건

디플로이할 각각의 모듈의 내부구조, 특성 등에 대한 충분한 이해가 있으면 본 매뉴얼을 이해하는데 별다른 어려움이 없을 것이다. 그리고 각각의 DD 스키마와 친숙한 것이 본 매뉴얼을 보는데 많은 도움이 될 것이다.

## 매뉴얼 구성

1. 소개: JEUS 시스템내에서 JEUS Builder 의 위치, 모듈의 조립/패키징, 그리고 DD 파일에 대한 개략적 설명
2. 뷰(View)
3. **JEUS Builder** 시작하기: 뷰에 따른 각 화면에 대한 상세한 설명
4. **모듈 생성**: 사용자로 하여금 JEUS Builder 에 대한 익숙함을 유도하기 위하여 유용한 모듈 생성

## 관련 매뉴얼

본 매뉴얼에서는 디플로이할 모듈의 생성과 변경 등에 대해서 설명하는데, 각각의 모듈을 사용하는 대부분의 매뉴얼이 관련되어 있다.

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

## 일러두기

| 표기 예                                      | 내용                              |
|---|---------------------------------|
| 텍스트                                       | 본문, 12 포인트, 바탕체 Times New Roman |
| <i>텍스트</i>                                | 본문 강조                           |
| CTRL+C                                    | Ctrl 과 C 를 동시에 누름               |
| <code>public class myClass { }</code>     | Java 코드                         |
| <code>&lt;system-config&gt;</code>        | XML 문서                          |
| 참고: / 주의:                                 | 참고 사항과 주의할 사항                   |
| JEUS_HOME                                 | JEUS 가 실제로 설치된 디렉토리             |
| <code>j eusadmin nodename</code>          | 콘솔 명령어와 문법                      |
| <code>&lt;&lt;FileName.ext&gt;&gt;</code> | 코드의 파일명                         |
| <b>그림 1.</b>                              | 그림 이름이나 표 이름                    |

## OS 에 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.

| 용어 Term        | Definition 정의                    |
|----------------|----------------------------------|
| <b>J2EE DD</b> | J2EE Deployment Descriptor 의 약어. |
| <b>JEUS DD</b> | JEUS Deployment Descriptor 의 약어  |

---

## 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](mailto: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](mailto:info@tmaxsoft.com)  
Web (English): <http://www.tmaxsoft.com>

### Japan

Tmax Soft Japan Co., Ltd.  
6-7 Sanbancho, Chiyoda-ku, Tokyo 102-0075  
Japan  
Tel: 81-3-5210-9270  
FAX: 81-3-5210-9277  
Email: [info@tmaxsoft.co.jp](mailto: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](mailto:info@tmaxchina.com.cn)  
Web (Chinese): <http://www.tmaxchina.com.cn>

# 1 소개

본 장에서는 JEUS Builder 가 JEUS 시스템에서 차지하고 있는 역할, 지원가능한 모듈, 그리고 작업가능한 영역 등에 대해서 간략하게 살펴볼 것이다.

## 1.1 개요

JEUS 시스템에서 사용되는 J2EE 애플리케이션은 표준 배치 서술자(J2EE DD), JEUS 배치서술자(JEUS DD), 그리고 하나 이상의 J2EE 모듈(컴포넌트)들로 구성되어 있다.

*JEUS Builder 는 J2EE 모듈을 조립/패키징하고, 배치서술자(deployment descriptor) XML 파일들을 작성할 수 있게 해주는 GUI 툴이다.*

JEUS 시스템 내에서 사용되는 J2EE 모듈의 흐름을 보면, JEUS Builder 에서 만들어서 JEUS 웹 관리자가 JEUS 엔진으로 디플로이한다[그림 1].



그림 1 JEUS 시스템내의 J2EE Module

## 1.2 지원 모듈

JEUS Builder 에서 사용되는 모든 모듈들은 Java™ Archive(JAR) 파일 형식으로 이루어져 있으며, 그 종류는 J2EE 스펙에 의거하여 Appliciaton, Applicaiton Client, Resource Adapter, Enterprise Bean, 그리고 Web Component 이다.

- *Application* 모듈은 엔진내에서 유용한 기능을 수행하는 하나 이상의 모듈로 구성된다.

- *Application Client* 모듈은 일반적으로 데스크탑에서 GUI 형태로 실행되는 자바 프로그램으로 사용자에게 middle tier 에 존재하는 각종 서비스를 손쉽게 이용할 수 있는 방법을 제공해 준다.
- *Resource Adapter* 모듈은 외부의 자원관리자와의 네트워크 연결을 구현하는 시스템 레벨의 컴포넌트로서 JDBC™ 과 같은 표준 API 를 사용하거나 리소스 어댑터(adapter)를 구현하여 엔진의 기능성을 확대시킬수 있다.
- *Enterprise Bean* 모듈(EJB 모듈)은 트랜잭션을 지원하는 환경에서 실행되며 Application 모듈의 로직을 담당하며, 웹서비스를 지원하기도 한다.
- 마지막으로 *Web Component* 모듈(WEB 모듈)은 servlet, JSP page, 필터(filter), 그리고 이벤트 리스너(event listener)로 구성되며, 클라이언트에 HTTP request 에 대한 응답을 보내기도 한다. 이 모듈 역시 웹서비스를 지원하는데 본 매뉴얼에 자세하게 설명할 것이다.

여기서 언급한 각 모듈의 명칭은 GUI 상에서 사용되므로 잘 기억해 두기 바람과 자세한 모듈에 대한 사항은 J2EE 스펙을 참고하기 바란다.

## 1.3 작업 영역

상기 언급한 모듈을 목적에 맞게 만들기 위해서 JEUS Builder 에서는 다음과 같은 작업이 가능하다.

- 모듈 이름을 지정하고 파일 또는 모듈을 추가하여 새로운 모듈을 생성할 수 있다.
- 작성된 모듈에 파일 또는 모듈을 추가할 수 있다.
- 작성된 모듈내에 있는 파일 또는 모듈을 삭제할 수 있다.
- 작성된 Application 모듈에 내장된 모듈에 파일 또는 모듈을 추가할 수 있다.
- Optional package 지원을 위해서 MANIFEST.MF 에 class-path 를 편집할 수 있다.

## 1.4 제약 사항

모듈내의 파일 콘텐츠의 변동(추가, 삭제)에 대해서는 DD 파일들에 자동으로 반영되지 않는다. 그래서 사용자가 그 변동사항에 대해서 추가적인 작업을 해야 된다.

JEUS Builder에서는 DD 파일이 있으면 그 파일을 사용하고, DD 파일이 없으면 생성한다. 그러면 모듈 생성시와 같이 DD 파일이 없는 상태에서 모듈을 생성하면 현재 패키징된 파일 콘텐츠를 반영하는 DD 파일이 생성될 것이다. 그러나 모듈 열기와 같이 DD 파일이 모듈내에 존재할 경우에, 모듈내의 콘텐츠를 변경해도 DD 파일 있기 때문에 변경된 파일 콘텐츠는 반영되지 않는다.

## 1.5 배치서술자 (Deployment Descriptor)

잘 조립되고 패키징된 모듈을 J2EE 서버에 배치하여 구동시키기 위해서는 이 모듈을 잘 설명하고 있는 배치 서술자(DD) 파일이 필요하다. JEUS 시스템에서는 JEUS 엔진에서 이 모듈을 사용하기 위해서 J2EE DD 파일과 JEUS DD 파일이 존재해야 된다.

JEUS Builder에서는 DD 파일 생성에 관련된 사용자 편의를 위해서 다음과 같은 작업을 한다.

- Enterprise Bean 모듈(Entity Bean, Session Bean, 그리고 Message-Driven Bean) 생성시 해당 DD 파일이 없을 경우에 자동으로 생성시켜준다. 그리고 생성된 DD 파일은 GUI에 반영된다.
- Web Component 모듈 생성시 해당 DD 파일 없을 경우 자동으로 생성시켜준다. 그리고 생성된 DD 파일은 GUI에 자동 반영된다.

각각의 모듈에 대해서 J2EE DD 파일과 JEUS DD 파일이 있으며 자세한 내용은 다음과 같다[표 1]. 그리고 모듈에 해당하는 확장자 또한 주의깊게 살펴보기 바란다.

그리고 표의 마지막줄에 모듈타입은 아니지만 웹서비스 지원을 위한 DD 파일에 대해서도 참고하기 바란다.



| Module Type        | J2EE DD                | JEUS DD                 | Extention |
|--------------------|------------------------|-------------------------|-----------|
| Application        | application.xml        | jeus-application-dd.xml | ear       |
| Enterprise Bean    | ejb-jar.xml            | jeus-ejb-dd.xml         | jar       |
| Application Client | application-client.xml | jeus-client-dd.xml      | jar       |
| Resource Adapter   | ra.xml                 | jeus-connector-dd.xml   | rar       |
| Web Component      | web.xml                | jeus-web-dd.xml         | war       |
| (webservice)       | webservices.xml        | jeus-webservices-dd.xml | .         |

표 1 Deployment Descriptor Files

## 1.6 결론

JEUS Builder 은 JEUS 엔진으로 디플로이할 모듈을 생성하는 툴이다. 모듈 생성시에 디플로이에 관련된 여러가지 설정을 담고 있는 DD 파일 역시 생성되어 패키징된다.

JEUS Builder 는 목적에 맞는 모듈의 생성과 모듈내에 파일 컨텐츠에 대한 조작이 가능하다. 그러나 모듈내의 파일 컨텐츠 변동사항에 대해서는 자동으로 반영되지않고 사용자가 직접 화면에서 수정해야 된다.

다음 장에서는 JEUS Builder 의 뷰(View)에 대해서 살펴보도록 하자.

## 2 뷰(View)

JEUS Builder 의 GUI 는 모듈트리 뷰(module tree view), 워크스페이스 뷰(workspace view), 그리고 메시지 뷰(message view)로 나누어 질 수 있다. 특히 워크스페이스 뷰는 각각의 DD 파일의 스키마(schema)파일을 기반으로 생성되었기 때문에 화면상의 각 항목에 대한 자세한 사항은 해당 스키마 파일을 참조하기 바란다.

앞으로 우리가 사용할 예제는 JEUS 시스템 설치 후 예제 파일로 제공하는 opc.ear 이다. 이 예제는 Sun Microsystems 에서 제공하는 Adventure Builder(<http://java.sun.com/developer/releases/adventure>)로, JEUS 시스템 내에서 사용할 수 있도록 JEUS DD 파일을 추가한 것이다. 이 파일은 JEUS\_HOME\samples\quickstart\applications 에 놓여있다.

Adventure Builder 예제는 모듈 타입이 Application 이어서 archive 내부에 여러 개의 모듈이 포함되어 있다.

opc.ear 모듈을 열었을 때, JEUS Builder 전체 화면은 다음과 같다[그림 2].

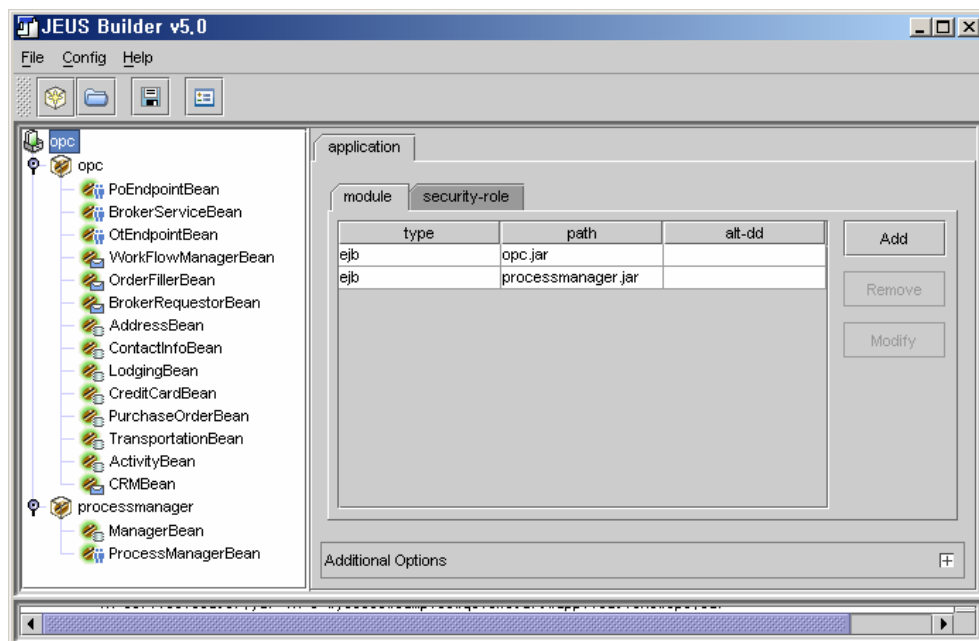


그림 2 JEUS Builder 전체화면

## 2.1 모듈 트리 뷰(ModuleTree View)

[그림 2]에서 모듈트리 뷰는 전체화면의 왼쪽 트리형태로 표현된 부분을 말한다. [그림 3]은 [그림 2]의 모듈트리 뷰를 펼쳤을 때의 캡처화면이다.

[그림 3]을 보면 루트 노드는 applicaton 모듈의 모듈이름(opc)이고, 자식 노드에는 2 개의 EJB 모듈(opc, processmanager)이 있다. 이 자식 노드의 이름은 각각의 모듈 파일명(opc.jar, processmanager.jar)에 해당된다.

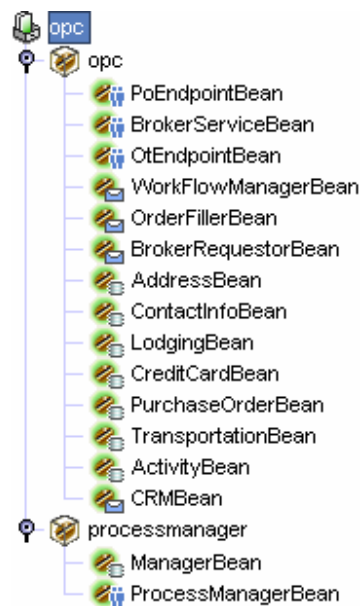


그림 3 Application 타입의 모듈 트리 뷰

각 EJB 모듈은 자신의 모듈 이름을 부모 노드로 하고 EJB 이름을 자식 노드로 하는 서브 트리 형태로 구성되어 있다.

WEB 모듈의 경우에서도 또한 자신의 모듈 이름을 부모 노드로 하고, 서블릿 이름을 자식 노드로 갖는 서브 트리를 구성하고 있다.

Application 타입 외의 독립적인(stand-alone) 모듈일 경우의 모듈 트리 뷰의 형태는 application 타입의 모듈 트리 뷰에서 보았던 각각의 서브트리와 같은 형태이다.

예를 들어 Application 타입의 모듈에 내장된 EJB 모듈을 독립적으로 열어서 본 모듈트리 뷰는 다음과 같다[그림 4].

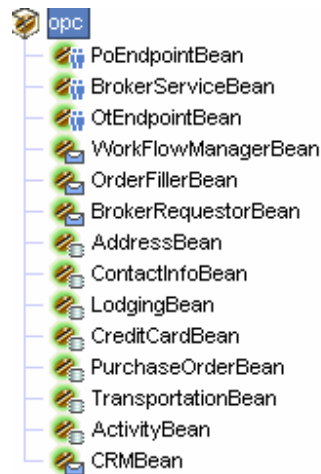






그림 4 EJB 모듈 타입의 모듈 트리 뷰

이렇게 구성된 모듈트리 뷰의 각각의 노드들을 선택함으로써 나타나는 workspace 뷰를 통해서 디플로이와 관련된 설정을 열람하거나 수정할 수 있다.

이렇듯 JEUS Builder 에서는 해당 DD 파일과 관련된 정보에 대한 접근 방법 제충적인 모듈트리 뷰를 제공함으로써, 사용자들에게 정보 접근의 용이성을 제공한다.

### 2.1.1 트리 노드 아이콘 이미지

모듈 트리 뷰에서는 각 트리 노드별로 다른 아이콘을 사용하고 있다[표 2].

| 아이콘 이미지   | 항 목                       |
|---|---------------------------|
|  | Application Module        |
|  | EJB Module                |
|  | Application Client Module |
|  | Resource Adapter Module   |







|   |                            |
|---|----------------------------|
|  | WEB Module                 |
|  | EJB Entity Bean            |
|  | EJB Stateful Session Bean  |
|  | EJB Stateless Session Bean |
|  | EJB Message-Driven Bean    |
|  | Servlet                    |

표 2 모듈트리뷰 아이콘 이미지

## 2.2 워크스페이스 뷰(Workspace View)

워크스페이스 뷰는 DD 파일과 관련된 항목들을 설정하기 위해 제공되는 화면이다. JEUS Builder에서는 J2EE DD 파일과 JEUS DD 파일의 설정과 관련된 화면이 탭 형태로 제공된다.

EJB 모듈인 경우에는 enterprise bean 종류마다 각기 다른 탭 화면을 제공한다. 그리고 사용자에게서 입력으로부터의 중압감을 해소하기 위해서 **Additional Options** 화면과 **Basic / Expert** 탭 화면을 제공한다.

이제 그 자세한 내용을 살펴보기로 하자.

### 2.2.1 DD 설정 화면

워크스페이스 뷰는 모듈트리뷰에서 선택된 노드에 따라서 다른 화면을 보여준다. Application 모듈의 루트노드를 선택하면 워크스페이스 뷰는 [그림 2]와 같이 나온다. 다른 모듈과 달리 application 모듈일 경우에는 JEUS DD 파일(jeus-application-dd.xml)이 거의 필요하지 않기 때문에, 설정화면을 보여주지 않는다.

반면 application 모듈 외의 다른 모든 모듈의 경우에는 항상 J2EE DD 파일과 JEUS DD 파일을 설정할 수 있도록 2 개의 탭화면이 제공된다. 예를 들어, [그림 5]는 consumerwebsite.ear 의 WEB 모듈을 열었을 때의 워크스페이스 뷰이다. 여기서는 J2EE DD 파일인 web.xml 파일을 설정하기

위한 **web-app** 탭, 그리고 JEUS DD 파일인 jeus-web-dd.xml 파일을 설정하기 위한 **jeus-web-dd** 탭이 있다.

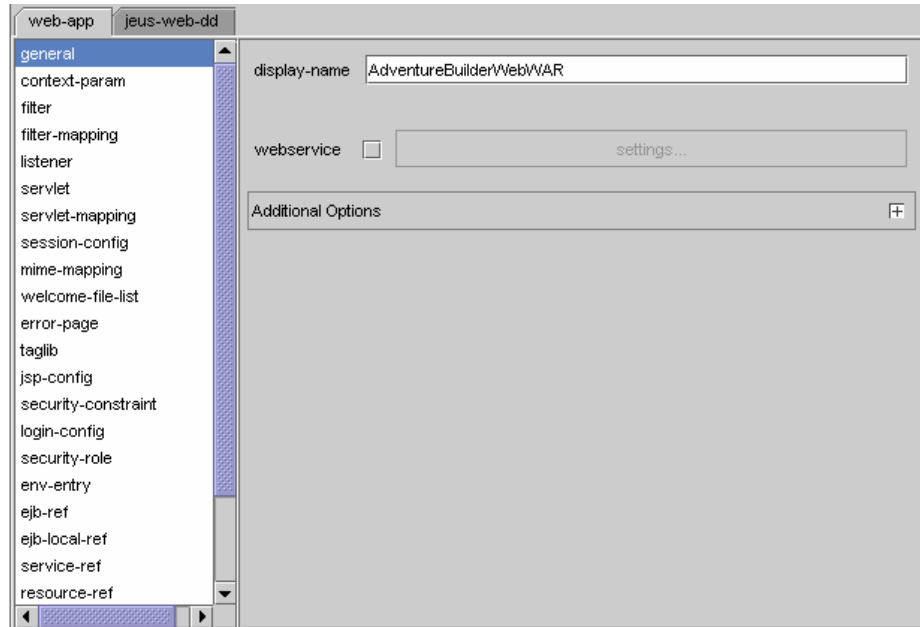


그림 5 WEB 모듈의 워크스페이스 뷰

opc.ear 의 EJB 모듈이 모듈트리 뷰에서 선택되면 J2EE DD 파일인 ejb-jar.xml 을 설정하기 위해서 **ejb-jar** 탭 화면, JEUS DD 파일인 jeus-ejb-dd.xml 을 위해서 **jeus-ejb-dd** 탭화면이 나타난다. EJB 모듈의 워크스페이스 뷰는 다음과 같다[그림 6].

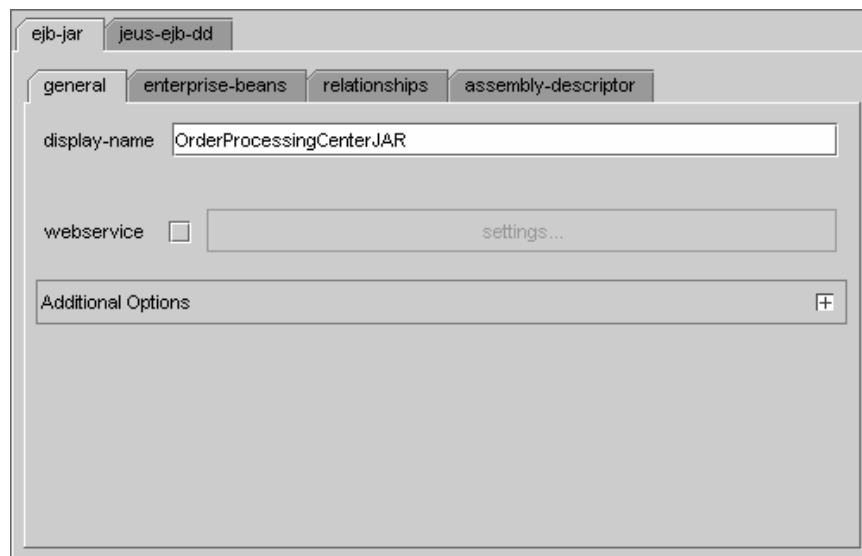


그림 6 EJB 모듈의 워크스페이스 뷰

나머지 모듈의 경우에는 [표 1]에 표기된 DD 파일을 설정하기 위한 각각의 탭화면이 보여지게 될 것이다.

### 2.2.2 Enterprise Bean 설정 화면

EJB 모듈일 경우에는 모듈 내에 존재하는 각 Enterprise Bean 에 대한 설정을 위한 탭화면을 제공한다. 이것 역시 두개의 탭 화면이 있는데 하나는 J2EE DD 파일을 위한 것이고, 나머지 하나는 JEUS DD 파일을 위한 것이다.

J2EE DD 파일을 위한 탭은 EJB 종류마다 다른데, entity bean 일 경우 **entity** 탭, session bean 일 경우 **session** 탭, 그리고 message-driven bean 일 경우에는 **message-driven** 탭이 각각 제공된다. JEUS DD 파일을 위한 탭은 EJB 종류와 상관없이 **jeus-bean** 탭 화면을 제공한다.

모듈트리 뷰에서 EJB 모듈의 entity bean 이 선택되었을 때는 다음과 같은 워크스페이스 뷰가 나타난다[그림 7].

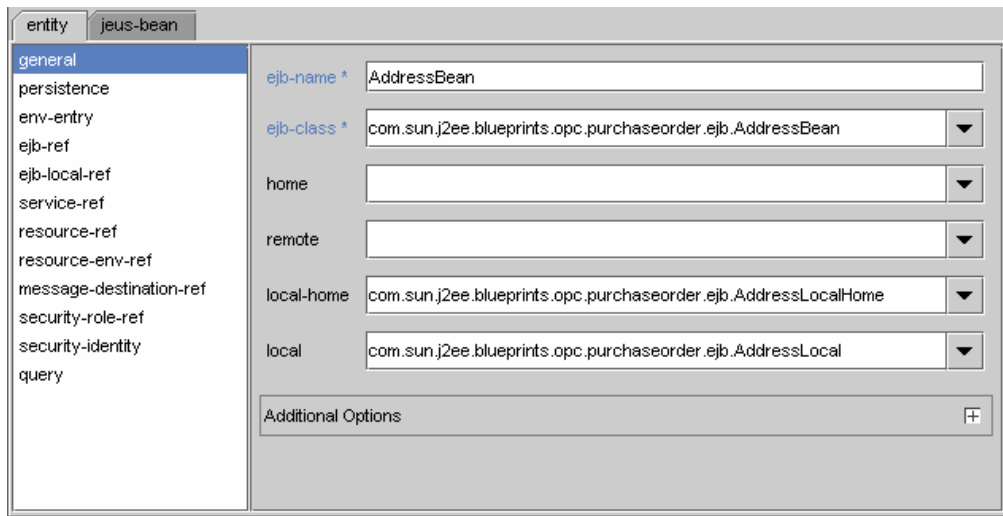


그림 7 Entity Bean 의 워크스페이스 뷰

### 2.2.3 Additional Options 화면

**Additional Options** 는 필수 입력 사항이 아니거나, 디폴트 설정 값만으로도 모듈 디플로이가 가능할 때 이들 설정화면에 대해서 감춤기능을 제공한다. 다시 말하면, **Additional Options** 화면에 있는 정보를 설정하지 않아도 모듈 디플로이에는 크게 문제가 되지 않는 것을 의미한다. 이 기능은 JEUS Builder 전반에 걸쳐 존재한다.

[그림 7]에서 감춤기능을 해제하고 난 다음의 화면은 아래와 같다[그림 8].

그림 8 Additional Options

#### 2.2.4 Basic / Expert 화면

JEUS DD 파일 설정시 필수 입력사항으로 요구되는 항목들은 **Basic** 탭 화면에, 디폴트 설정을 이용하거나 미입력해도 상관없는 항목들은 **Expert** 탭 화면으로 구분하여 J2EE 입문자나 JEUS Builder 초보자에게 입력의 편의성을 제공해 준다. **Expert** 탭 화면의 리스트 항목은 **Basic** 탭 화면의 리스트를 포함한다.

EJB 모듈과 WEB 모듈에서 항목들을 이렇게 화면 상으로 구분해서 보여주고 있다.

[그림 9]은 EJB 모듈의 기본적인 입력항목을 나타내는 **Basic** 탭 화면이고, [그림 10]은 **Basic** 탭의 기본적인 항목을 포함하고 디플로이 설정을 자세하게 위한 항목을 가지고 있는 **Expert** 탭 화면이다.



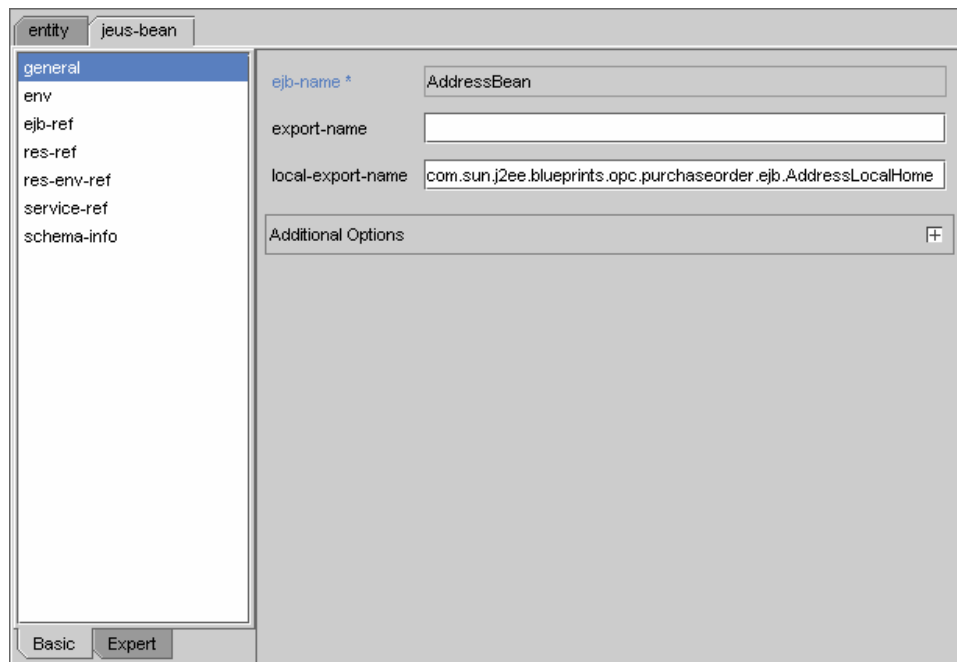


그림 9 EJB 모듈의 Basic 탭화면

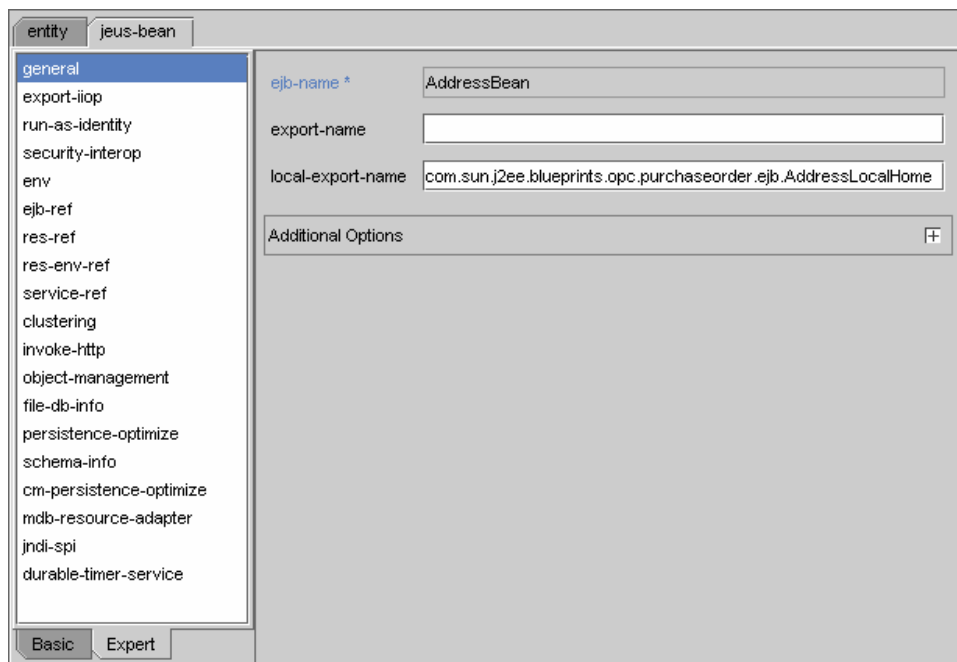


그림 10 EJB 모듈의 Expert 탭화면

### 2.2.5 입력 필수 사항

워크스페이스 뷰에서 보여지는 화면은 스키마(schema) 파일을 기반으로 해서 생성된다. 이 스키마 파일은 J2EE DD 파일과 JEUS DD 파일을 생

성하는데 사용된다. 스키마 파일에 필수사항으로 지정된 항목들은 워크스페이스 뷰에서 글자색이 짙은 청색이고 맨우측에 아스테리크(\*)가 첨자로 붙는다.

다음 화면으로 넘어갈 경우 필수 입력으로 지정된 항목이 미입력시에는 [그림 11]과 같은 메시지 대화상자가 뜬다. **Ignore** 버튼을 누르고 다음 화면으로 넘어갈 수 있으나, 데이터 의존성이 깨지거나 향후 모듈의 배포 실패 등의 여러문제를 야기할 수 있기 때문에 반드시 **Ok** 버튼을 누른 후 재입력하기 바란다.

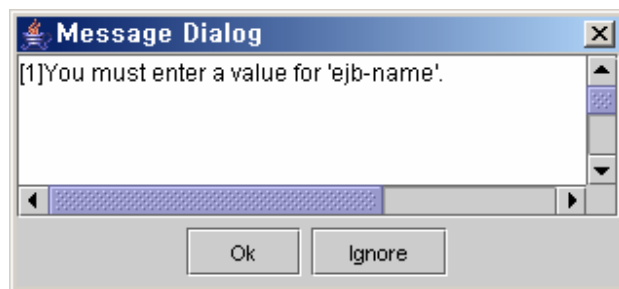


그림 11 필수입력항목 미입력시 메시지 대화상자

### 2.2.6 정보 의존성

J2EE DD와 JEUS DD 설정화면 사이에 의존성을 가지고 있는 항목들이 있다. 이렇게 의존성을 가지는 항목들은 화면상 편집 가능하지 않으며, 정보 변경시 자동으로 변경된다.

## 2.3 메시지 뷰(Message View)

JEUS Builder 전체 화면의 하단에 있는 메시지 뷰는 JEUS Builder 동작시 생성되는 각종 정보를 보여주는 화면이다[그림 12]. 이 뷰를 통해서 사용자는 자신이 실행한 JEUS Builder의 동작 결과를 확인할 수 있으며, 문제가 발생했을 경우에는 오류 메시지를 보여줌으로써 사용자가 적합한 대처방안을 모색할 수 있게끔 도와 준다.

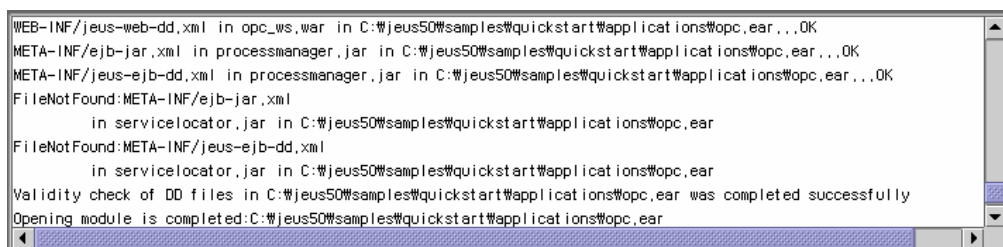


그림 12 Message View 화면

## 2.4 결론

JEUS Builder 의 화면은 모듈트리 뷰, 워크스페이스 뷰, 그리고 메시지 뷰로 구성되어 있다.

모듈트리 뷰는 사용자에게 정보 접근의 용이성을 제공하고, 워크스페이스 뷰는 스키마기반으로 생성되고 DD 파일 설정의 중심되는 화면이다. 그리고 마지막으로 메시지뷰는 JEUS Builder 의 현재 동작에 대한 메시지가 출력되는 화면이다.

다음 장부터는 실제로 JEUS Builder 를 다루어 보기로 하자.

## 3 JEUS Builder 시작하기

이번 장에서는 예제를 통해서 JEUS Builder 의 기능적인 측면을 설명하도록 하겠다.

### 3.1 실행하기

JEUS Builder 를 실행하기 위해서는 환경변수 JEUS\_HOME 만 설정되어 있으면 된다. 여기서는 JEUS 홈디렉토리가 성공적으로 설정되어 있다고 가정하자. JEUS Builder 를 윈도우 시스템에서 실행시키기 위해서는 다음과 같은 방법이 있다.

1. 윈도우즈시작>프로그램>TmaxSoft>JEUS5>JEUSBUILDER 메뉴 선택
2. 콘솔화면에서 아래와 같이 입력(배치파일 실행)  
  
C: \j eus50\bi n>j eusbui l der
3. 마지막으로 윈도우 탐색기에서 JEUS\_HOME\bin\jeusbui l der .cmd 아이콘을 더블클릭

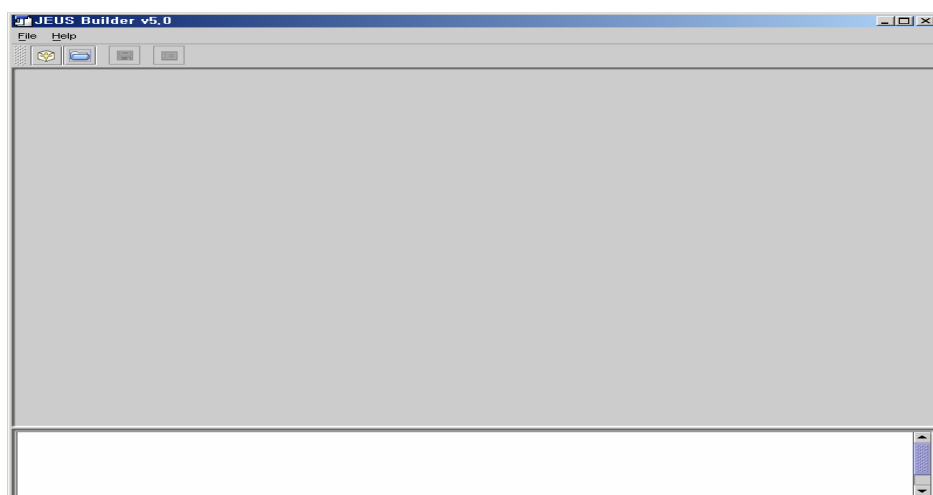



그림 13 JEUS Builder 초기 화면

환경변수 설정과 JEUS Builder 를 성공적으로 수행했다면 [그림 13]과 같은 초기 화면을 볼 수 있을 것이다. 초기 화면에서는 메시지 뷰(message view)만 볼 수 있고, 모듈트리 뷰(moduletree view)와 워크스페이스 뷰(workspace view)는 내용이 없어서 비어 있는 상태이다.

## 3.2 모듈 열기

모듈을 열기 위해서는 File>Open 메뉴를 선택 하거나 툴바에서 모듈열기 아이콘( )을 선택하면 된다. opc.ear 파일 열기를 성공했다면 [그림 14]와 같은 화면을 볼 수 있을 것이다.

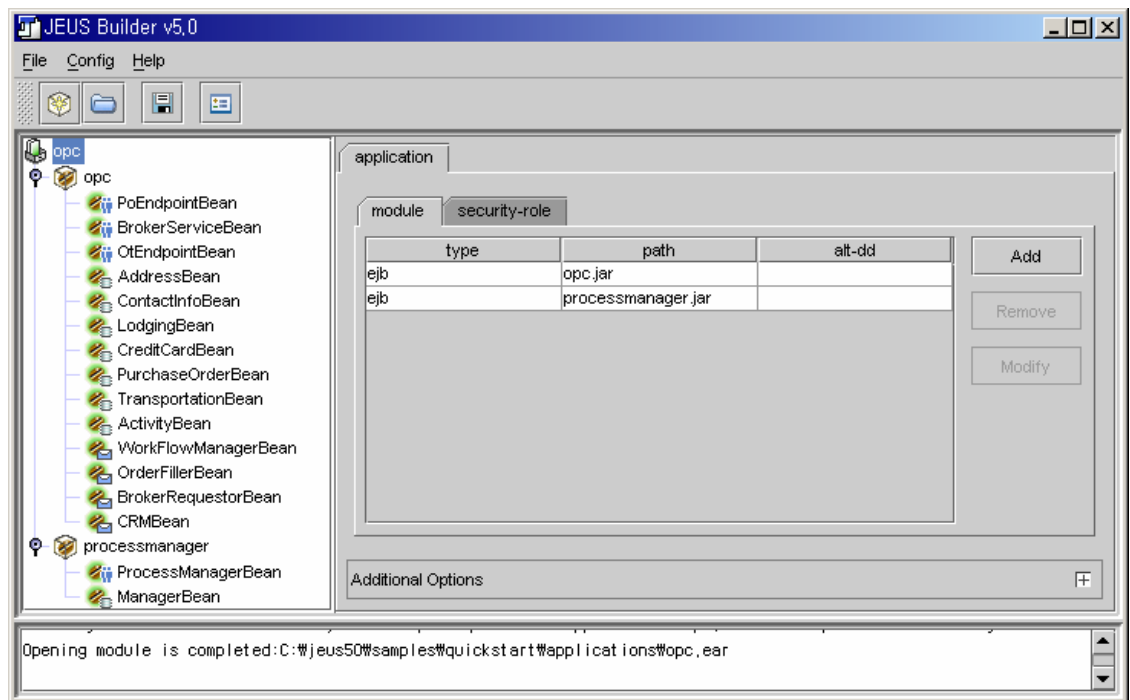


그림 14 모듈 열기

### 3.2.1 배치 서술자 생성

열고자 하는 모듈 안에 배치 서술자가 없을 경우 JEUS Builder 에서는 해당 배치 서술자 파일(J2EE DD, JEUS DD)을 생성한다.

EJB 모듈과 WEB 모듈의 경우에 배치서술자가 없을 경우에는 모듈내의 관련 파일을 분석하여 모듈 디플로이에 필요한 충분한 정보를 가지는 배치 서술자를 자동으로 생성해 준다.

더 나아가 Application 모듈의 내장된 EJB 모듈과 WEB 모듈이 배치서술자를 가지고 있지 않을 때도 자동으로 배치 서술자를 생성해준다.

그리고 JEUS Builder에서는 배치 서술자를 읽을 때 유효성 검사를 실시한다. 이 유효성검사를 실시하여 이전 버전이거나 원하는 포맷이 아닐 경우, 이 배치서술자를 무시하고 새로운 배치서술자를 생성하여 덮어쓰기를 할 것인지 결정해야 된다[그림 15].

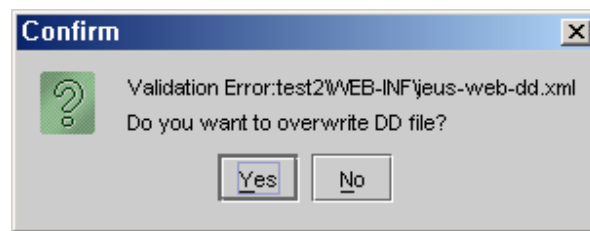


그림 15 DD 파일 유효성검사 실패시 덮어쓰기 여부 결정 대화상자


만일 위의 경우 덮어쓰기를 선택한다면 JEUS DD 파일인, jeus-web-dd.xml은 JEUS Builder 에서 자동으로 생성되며, 워크스페이스뷰에서 관련화면인 jeus-web-dd 탭 화면 또한 생성된 JEUS DD 파일의 내용을 기반으로 하여 데이터가 채워질 것이다.

그러나 덮어쓰기를 원하지 않는다면 JEUS DD 파일은 생성되지 않으며, jeus-web-dd 탭 화면 또한 보여지지 않을 것이다.

### 3.3 모듈 저장하기

모듈 저장은 현재 작업중인 J2EE DD 파일과 JEUS DD 파일에 화면에 변경된 데이터를 반영하는 과정을 말한다. 그리고 새로운 모듈을 생성할 때는 추가된 파일들을 실제로 패키징하여 archive 파일을 생성하는 것까지 포함한다.

그리고 Application 모듈의 경우에는 내장된 모듈의 J2EE DD 파일과 JEUS DD 파일 또한 저장된다.

모듈을 저장하기 위해서는 **File->Save** 메뉴를 선택하거나 툴바에서 모듈저장하기 아이콘()을 선택하면 된다. 그러면 메시지 뷰에 저장 관련된 메시지가 보여질 것이다. 저장되는 파일의 이름은 다음 절에서 설명할 파일 콘텐츠 대화상자 화면에서 **Module Info**의 **Location** 텍스트필드를 보면 알 수 있을 것이다.


그리고 현재 작업중인 모듈을 다른 이름으로 저장할 수도 있는데 **File->SaveAs** 메뉴를 선택하면 된다. 그러면 파일 콘텐츠 대화상자에 있는 저장 대상 파일의 이름 또한 수정될 것이다.

### 3.4 모듈 파일 콘텐츠 다루기

모듈을 구성하고 있는 파일 콘텐츠를 다루는 경우는 다음과 같다.

- 모듈의 최초 생성시에 파일들을 추가시키거나 삭제한다.
- 생성된 모듈을 열어서 새로운 파일들을 추가하고, 모듈 내에 있는 파일들을 삭제한다.

이렇게 모듈 내의 콘텐츠를 다룰수 있는 공간이 **File Contents** 대화상자이다.

이 대화상자는 모듈을 새로 만들 때 최초로 만나는 화면이다. 모듈 생성 또는 열기 이후에 활성화 되어 있는 파일콘텐츠 툴바 아이콘()을 눌러도 만날 수 있다.

파일콘텐츠 대화상자에서는 다음과 같은 일들을 할 수 있다.

- 모듈 타입을 지정할 수 있다.
- 모듈의 파일이름을 포함한 모듈 경로를 지정할 수 있다.
- 추가시킬 파일들의 위치를 지정할 수 있다.
- 모듈에 파일을 추가할 수 있고, 모듈내의 파일을 삭제할 수도 있다.
- 모듈의 MENIFEST.MF 에 있는 클래스패스 속성을 설정할 수 있다.

그러나, 현재 작업중인 모듈의 DD 파일이 존재한다면 이러한 콘텐츠의 변경은 반영되지 않는다. 다시 말하면 **File Contents** 대화상자에서 제거한 파일에 대한 정보는 JEUS Builder 의 워크스페이스 뷰화면에서 사용자가 수동으로 반드시 수정해 주어야 된다.

이제부터 File Contents 대화상자가 하는 일을 구체적으로 설명할 것이다. opc.ear 모듈을 연 후 파일컨텐츠 톨바 아이콘을 클릭했을 때의 화면은 다음과 같다[그림 16].

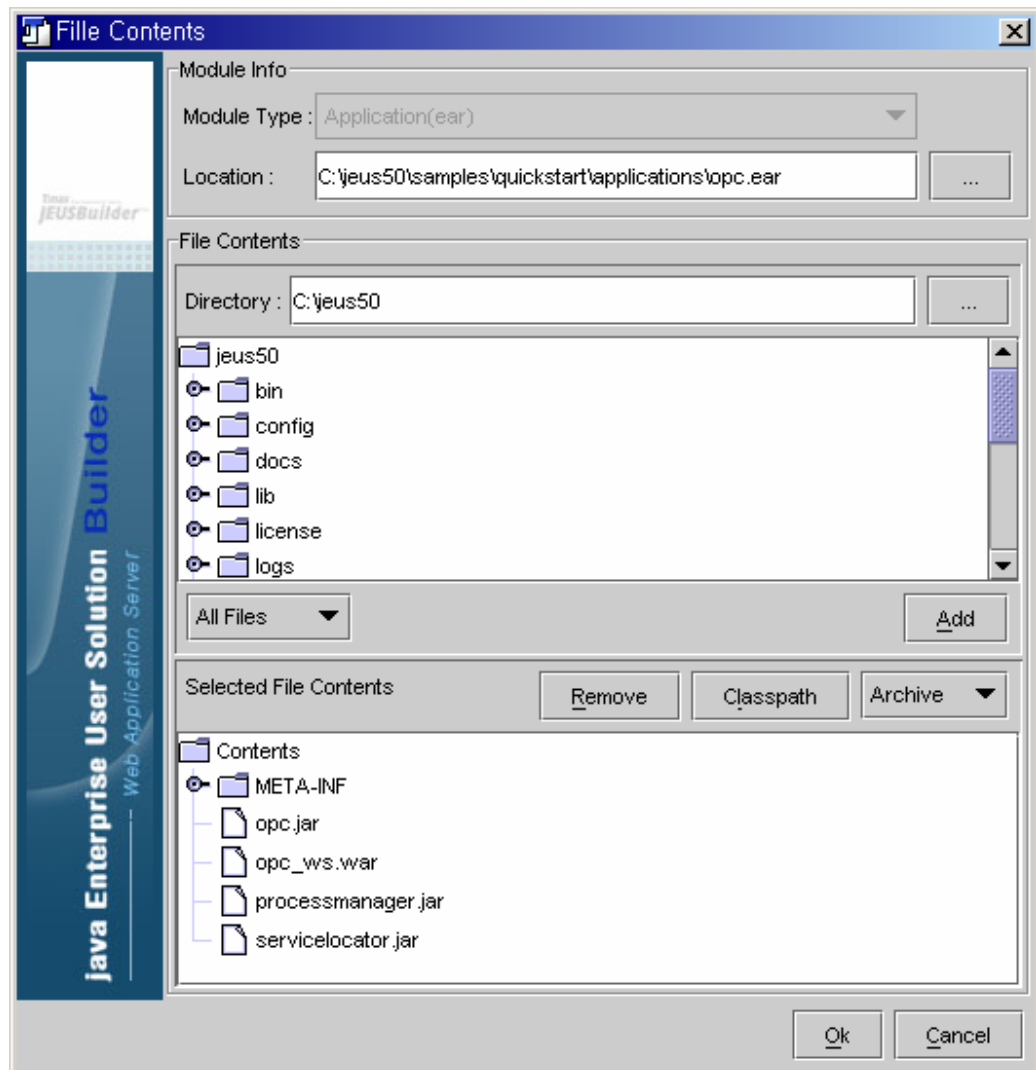


그림 16 File Contents 대화상자 (Application 타입)

### 3.4.1 Module Info

모듈의 기본적인 정보를 설정하는 곳으로 **Module Type** 콤보박스와 **Location** 텍스트 필드가 있다.

#### 3.4.1.1 Module Type

모듈타입을 선택하는 곳이다. 모듈타입의 종류는 1 장에서 자세히 살펴 보았다. 콤보박스 아이템은 다음과 같으며, 괄호안은 확장자를 나타낸다.



- Application(ear)
- Enterprise Bean(jar)
- Application Client(jar)
- Resouce Adapter(rar)
- Web Component(war)

모듈의 최초 생성시에는 콤보박스가 활성화 되어 있어서 모듈타입의 선택할 수 있다. 그러나 생성된 모듈을 열 때는 그 모듈타입이 이미 정해져 있기 때문에 콤보박스가 비활성되어 모듈타입 변경이 불가능하다.

#### 3.4.1.2 Location

여기는 모듈의 아카이브 파일의 절대경로를 표시하는 곳이다. 모듈 열기시에는 자동으로 설정된다. 모듈 생성시에는 파일선택 대화상자를 이용하여 대상파일을 지정하는 것이 편리할 것이다.

그리고 입력한 모듈의 확장자가 위에 있는 모듈타입과 일치하지 않을 때는 다음과 같은 오류 대화상자가 뜬다[그림 17]

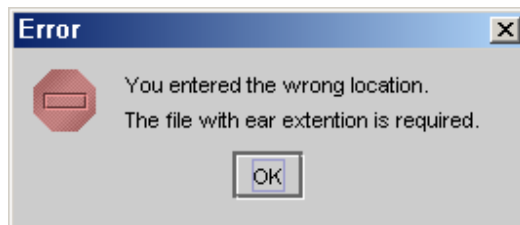


그림 17 모듈파일입력 오류시 발생하는 오류메시지

#### 3.4.2 File Contents

여기서는 실제적으로 모듈과 관련된 파일들이 다루어지는 곳이다. 선택한 파일들은 **Add** 버튼을 눌러서 아래 화면(**Selected File Contents**)으로 추가시킨다. 그리고 아래 화면에서 삭제하고픈 파일들을 선택하고 **Remove** 버튼을 누르면 해당 파일들이 삭제된다. 추가/삭제 동작은 **Module Type** 마다 조금씩 틀리므로 유심히 살펴보기 바란다.

##### 3.4.2.1 Directory 텍스트 필드

모듈에 추가할 파일에 대한 위치를 설정하는 텍스트필드이며, 브라우저 버튼을 사용하여 디렉토리를 선택하면 편리하다. 일반적인 사용법은 모

들에 추가할 파일이 있는 파일/디렉토리 바로 직전의 디렉토리를 설정하면 된다.

#### 3.4.2.2 File Type 콤보박스

작업 디렉토리 설정을 해서 보여지는 로컬 파일 시스템상의 파일들을 해당 형식에 맞게 필터링해서 보여준다. 예를 들어서 클래스 파일만 추가하고 싶을 때는 파일형식을 **Class File** 을 선택하면 작업 디렉토리의 트리에는 클래스파일만 보여지게 된다.

#### 3.4.2.3 Classpath 버튼

JAR 파일 스펙에 의해서, J2EE JAR 파일은 동일한 J2EE application 패키지에 있거나 이전에 J2EE 컨테이너에 인스톨된 .jar 파일에 있는 유틸리티(utility) 클래스, 공유(shared) 클래스나 리소스들을 참조할 수 있다.

그래서 JAR 형태의 파일은 JAR 파일의 Manifest 파일에 있는 Class-Path 엔트리에 설정 되어있는 .jar 파일을 참조할 수 있다. Manifest 파일은 JAR 파일의 META-INF/MANIFEST.MF 이다.

파일 컨텐츠 대화상자에서는 이 Classpath 버튼을 통해서 Manifest 파일의 Class-Path 엔트리를 설정할 수 있다. opc.ear 모듈에 설정된 클래스 패스는 [그림 18]와 같다.

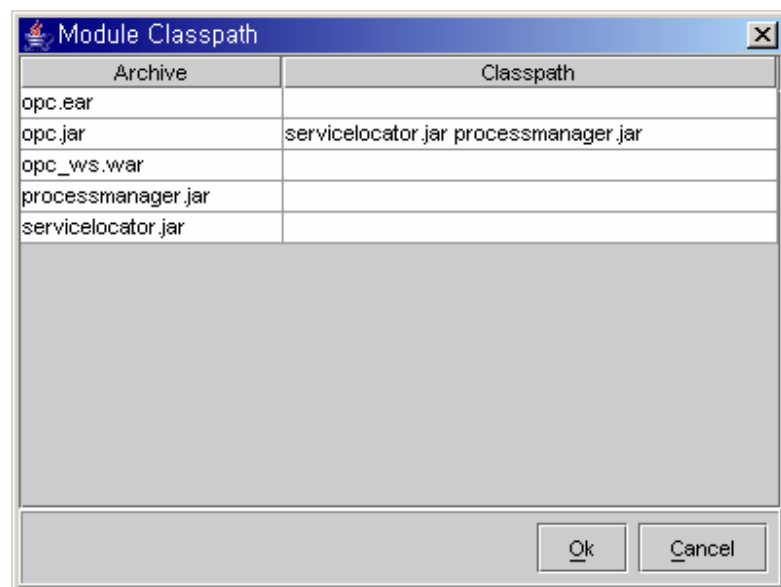


그림 18 모듈 클래스패스 대화상자(application type)

대화상자에서 **Archive** 컬럼의 제일 첫 줄은 현재 작업중인 모듈을 나타낸다. 만일 모듈 생성중에 모듈 이름이 정해지지 않았다면 (EMPTY)가 될 것이다. 이후 컬럼 내용은 내장된 모듈을 보여준다. 그러나 **Application** 타입일 경우 APP-INF/lib, **WEB** 모듈일 경우 WEB-INF/lib 에 있는 .jar 파일을 이 archive 컬럼에서 제외된다.

**Classpath** 컬럼의 내용은 해당 Manifest 파일의 Class-Path 엔트리를 읽어서 보여준다. [그림 18]에서 **Classpath** 컬럼이 비어 있다는 것은 해당 Manifest 파일의 Class-Path 엔트리에 설정된 값이 없다는 것을 나타낸다.

물론 추가하거나 수정하여 Ok 버튼을 누르면 해당 Manifest 파일에 적용된다.

*한 가지 주의할 것은 **Classpath** 컬럼에 하나 이상의 클래스패스를 입력하고자 할 때는 관련 스펙에 의거하여 반드시 스페이스 한 칸을 추가한 후 클래스 패스를 입력해야된다.*

#### 3.4.2.4 Add 버튼

이 버튼은 최초 모듈 생성시 파일을 넣을 때 또는 생성된 모듈을 열어서 추가적인 파일을 넣을 때 사용된다.

일반적으로 JEUS Builder 에서 파일을 추가할 때는 추가할 파일의 경로를 작업 디렉토리에 만든 후 최상위 디렉토리를 선택하여 추가한다. 그리고 이미 존재하는 파일에 대해서는 덮어쓰기 된다.

그런데, EJB 모듈과 Application 모듈에서의 파일추가는 다른 모듈과 차이점에 있으니 살펴보도록 하자.

EJB 모듈일 경우 클래스 파일 추가시, 그 패키지에 해당하는 디렉토리 노드가 **Selected Files Contents** 화면에 트리구조로 생성된 다음 말단노드에 클래스 파일이 추가된다. 이는 클래스의 패키지중에 최상위를 선택한 다음에 추가하는 것과 동일하게 동작한다.

Application 모듈일 경우에는 두 가지 모드 모두 선택된 파일들을 보여준다. 하나는 **Archive** 모드로써 이는 내장모듈을 파일 형태로 보여주는 준다. 다른 하나인 **Extraction** 모드는 내장모듈에 포함되는 파일들을 보여주기 위해서 디렉토리 형태로 보여준다.

그래서, Application 타입일 경우에는 내장된 모듈에 파일을 추가할 수 있는데, 이렇게 하기 위해서는 우선 **Extraction** 모드로 설정해 두고 해당 모듈을 선택해야만 된다. 내장 모듈에 파일을 추가할 경우에는 반드시

시 하나의 archive 모듈을 선택해야만 한다. [그림 19]은 대상 내장 모듈을 선택하지 않았을 경우에는 발생하는 오류 메시지이다.

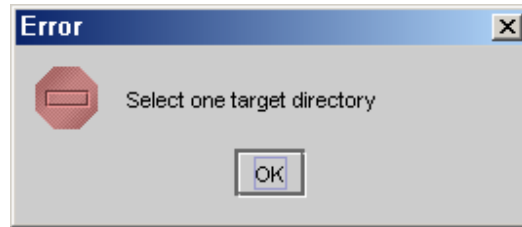


그림 19 내장모듈에 파일추가시 오류메시지

[그림 20]은 허용하지 않는 내장모듈에 파일을 추가하고자 할 경우 발생하는 오류 대화상자 이다.



그림 20 내장모듈 파일추가시 오류메시지

Application 모듈에서 내장모듈외의 다른 파일들을 추가시키하고자 할 경우는 **Archive** 모드로 파일을 추가하면 된다.

#### 3.4.2.5 Remove 버튼

사용자는 삭제할 노드들을 선택하여 **Remove** 버튼을 누르면 선택된 하나 이상의 노드가 삭제된다.

Application 모듈의 **Extraction** 모드에서는 파일을 삭제할 수 없다[그림 21]. 노드를 삭제하고 싶을 때는 **Archive** 모드로 설정한 다음에 가능하다.

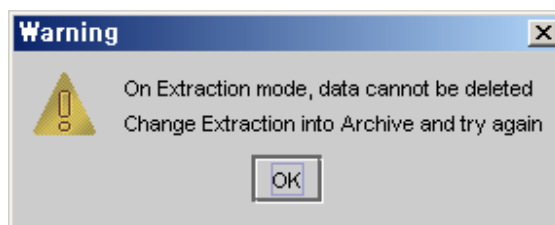


그림 21 Application 모듈의 Extraction 모드에서의 삭제시 경고메시지

### 3.5 종료하기

모든 작업이 완료되었으면 JEUS Builder 를 종료해야된다. JEUS Builder 의 종료는 **File->Exit** 메뉴나 프레임의 닫기 아이콘을 사용하면 된다. 현재 작업중인 데이터가 있으면 저장 여부를 묻는 대화상자가 뜰것이다 [그림 22]. 현재 작업중인 모듈을 저장하고 싶으면 **Yes** 버튼을, 저장하지 않을려면 **No** 버튼을, 그리고 JEUS Builder 를 종료하지 않고 메인화면으로 돌아가고 싶으면 **Cancel** 버튼을 클릭하라.

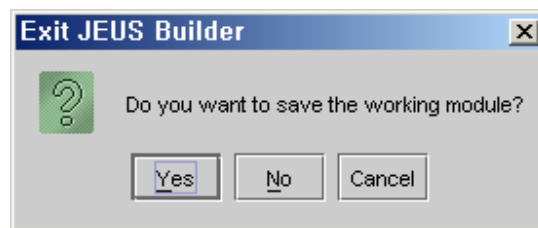


그림 22 JEUS Builder 종료시 대화상자

### 3.6 결론

이번 장에서는 JEUS Builder 실행하기, 모듈 열기, 그리고 모듈 저장하기 등 기본적인 기능에 대해서 알아보았다.

JEUS Builder 의 주요 기능은 모듈 관련파일 조립과 DD 파일 생성이다. 파일의 조립은 File Contents 대화상자를 통해서 추가 또는 삭제가능하다. 그리고 모듈을 열때에 관련 DD 파일이 존재하지 않을 때 JEUS Builder 는 최소한의 정보를 가지는 DD 파일을 생성시켜주며(EJB 모듈과 WEB 모듈의 경우에는 디플로이에 필요한 충분한 정보 포함), 모듈 저장시에 워크스페이스뷰의 각종 화면에 변경된 데이터를 DD 파일에 정확하게 반영한다.

다음 장에서는 모듈 생성에 대해서 자세히 다룰 것이다.

## 4 모듈 생성


본 장에서는 JEUS Builder 를 이용하여 모듈을 생성시키는 방법을 배울 것이다. 생성해볼 모듈로는 CMP(Container Managed Persistence)2.0 Entity Bean 을 이용한 EJB 모듈과 웹서비스를 사용하는 WEB 모듈이다.

사용되는 예제는 JEUS 를 설치후 생성되는 예제 디렉토리에서 찾을 수 있을 것이다.

### 4.1 CMP 2.0 의 EJB 모듈 생성

본 섹션에서는 CMP(Container Managed Persistence)2.0 Entity Bean 을 이용한 EJB 모듈을 만드는 것을 살펴보도록 하겠다.

간략하게 살펴보면, **File Contents** 대화상자의 **Ok** 버튼을 클릭하면 추가된 파일에 대해서 JEUS Builder 에서 기본적인 정보를 담고 있는 DD 파일들을 자동으로 생성한다. 나머지 정보에 대해서는 사용자가 워크스페이스 뷰 화면을 이용하여 설정하면 된다.

1. JEUS Builder 를 실행시키고 모듈 새로 만들기를 선택한다. 모듈 새로 만들기는 **File->New** 메뉴를 선택하거나 툴바 아이콘()을 선택하면 된다. 그러면 [그림 23]과 같이 파일을 추가할 수 있는 대화상자를 보게 될 것이다.

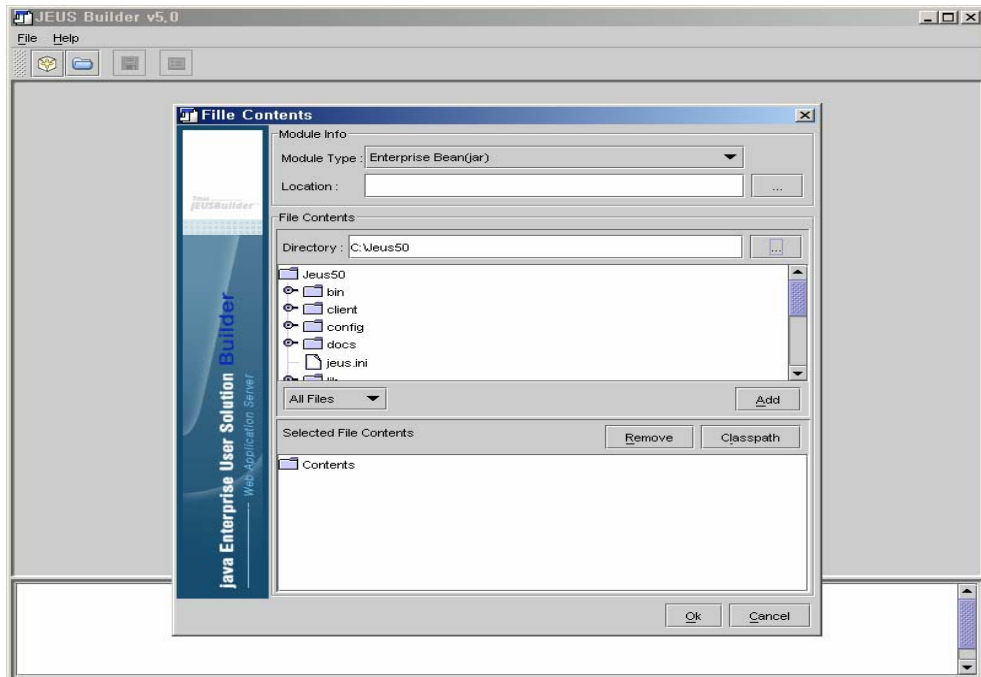


그림 23 모듈 생성시 File Contents 대화상자를 포함한 JEUS Builder

2. CMP2.0의 Entity Bean 모듈을 생성할 것이기 때문에 **Module Type**은 **Enterprise Bean(jar)**를 선택한다.
3. **Location** 입력을 위해서 오른쪽에 있는 브라우저 버튼을 클릭한다.
4. 모듈을 저장할 디렉토리 및 파일명을 지정하는 것을 도와주는 대화상자가 나타난다[그림 24]. 대화상자에 보여지는 디렉토리 디폴트 위치는 **JEUS\_HOME**이며, 작업 중인 경우에는 모듈이 있는 위치 바로 위 디렉토리가 되겠다.
5. 모듈을 저장할 디렉토리를 선택한 후, **File Name** 텍스트 필드에 생성할 모듈의 이름, **book20.jar**를 입력한다. 그리고 열기 버튼을 클릭한다.



그림 24 Archive Path 지정을 위한 대화상자

6. **Module Type** 과 **Location** 을 입력하고 난 후에 **File Contents** 대화상자의 **Module Info** 내용은 다음과 같다[그림 25]. **Location** 을 브라우저 버튼을 사용하지 않고 직접 텍스트 필드에 입력해도 상관 없다. 어쨌든 파일이름의 확장자와 **Module Type** 에서 지원하는 확장자는 반드시 일치하여야 한다. 그렇지 않으면 대화상자 종료시 오류가 발생할 것이다(3.4.1 참조).

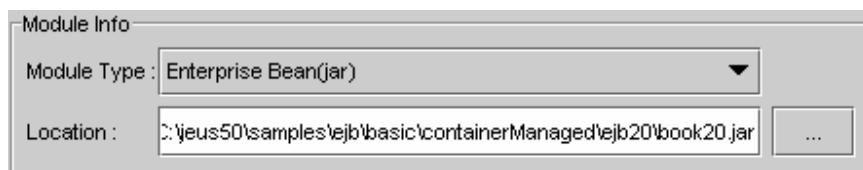


그림 25 book20 EJB 모듈의 모듈정보

7. 이제 모듈에 들어갈 클래스 파일을 추가해 보자. 우선 **Directory** 텍스트 필드 오른쪽의 브라우저 버튼을 클릭한다.
8. 그러면 **JEUS\_HOME** 을 현재 위치로 해서 디렉토리를 선택할 수 있는 파일 선택 대화상자가 나타난다[그림 26].



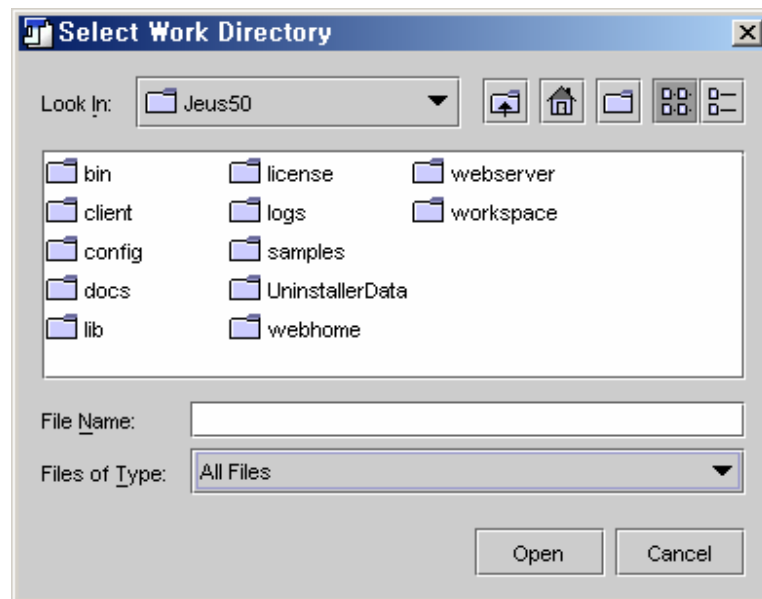


그림 26 작업디렉토리 선택을 위한 대화상자

9. 추가할 파일이 있는 바로 직전의 디렉토리를 선택하고 **Open** 버튼을 클릭한다.
10. 선택한 디렉토리 경로는 **Directory** 텍스트 필드에 나타나고, 모듈에 추가할 파일들은 트리형태로 표시된다[그림 27].

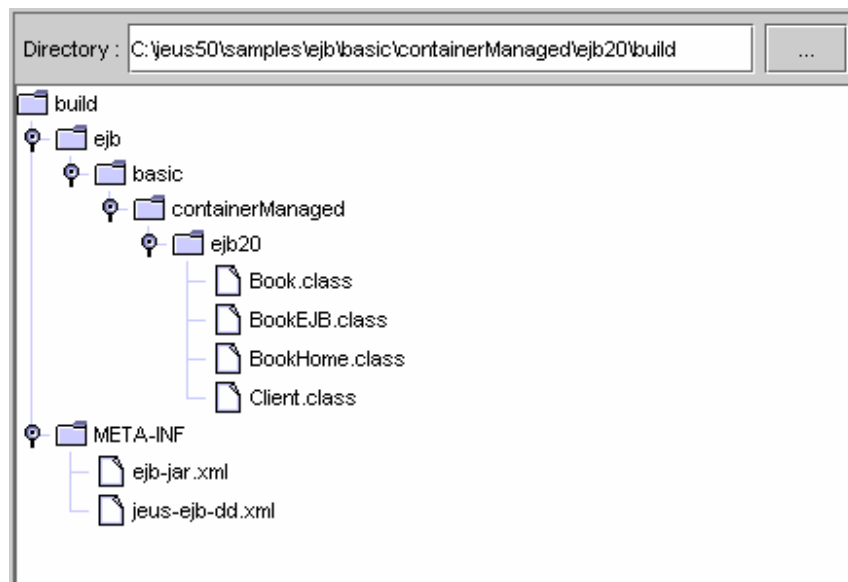


그림 27 추가 대상 디렉토리

- 트리노드에서 추가할 대상의 최상위 디렉토리를 선택한다. 다중 선택 역시 가능하다. 여기서는 [그림 27]에서와 같이 **ejb** 노드를 선택한다. 예제와 같이 DD 파일을 가지고 있다면 META-INF 노드를 선택하여 추가할 수 있다. 여기서는 DD 파일이 없다고 가정하고 DD 파일을 JEUS Builder 에서 생성시키기 때문에 추가시키지 않았다.

JEUS Builder 에서는 EJB 모듈에서 클래스를 추가할 때 편리한 방법을 제공한다. 추가하고자 하는 여러 개의 클래스 파일만 선택하여 추가버튼을 눌러도 최상위 디렉토리를 선택하는 방식과 같은 동작을 보여준다. 이는 내부적으로 클래스에서 패키지 정보를 추출하여 트리노드를 생성하기 때문에 가능하다. 아래는 클래스 파일들 만을 선택했을 때 화면을 보여주고 있다[그림 28].

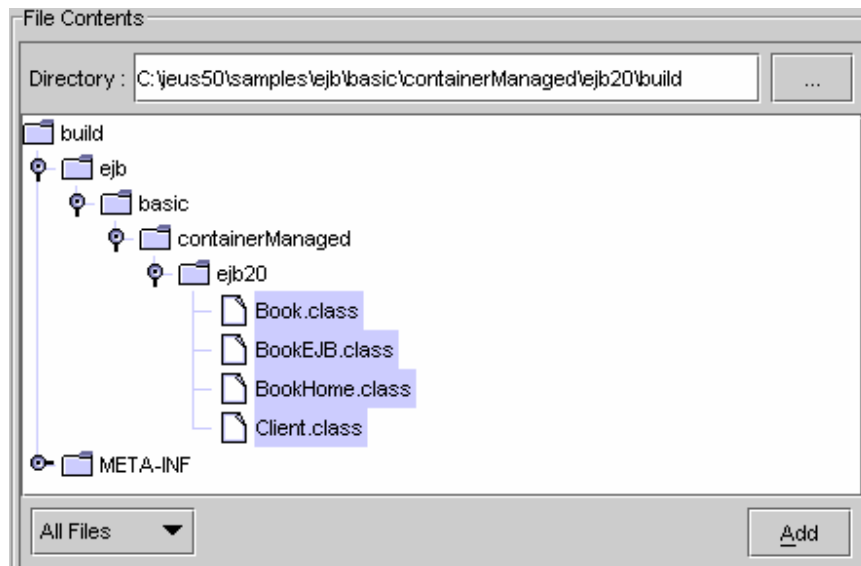


그림 28 EJB 모듈 생성시 추가할 클래스파일 선택

- 대상 파일 또는 디렉토리를 선택했다면 EJB 모듈에 파일을 추가하기 위해 **Add** 버튼을 클릭한다.
- 그러면 **Selected File Contents** 화면에 **Contents** 를 루트노드로 하고 추가 대상파일과 동일한 계층구조를 가지는 트리가 생성됨을 볼 수 있다[그림 29].

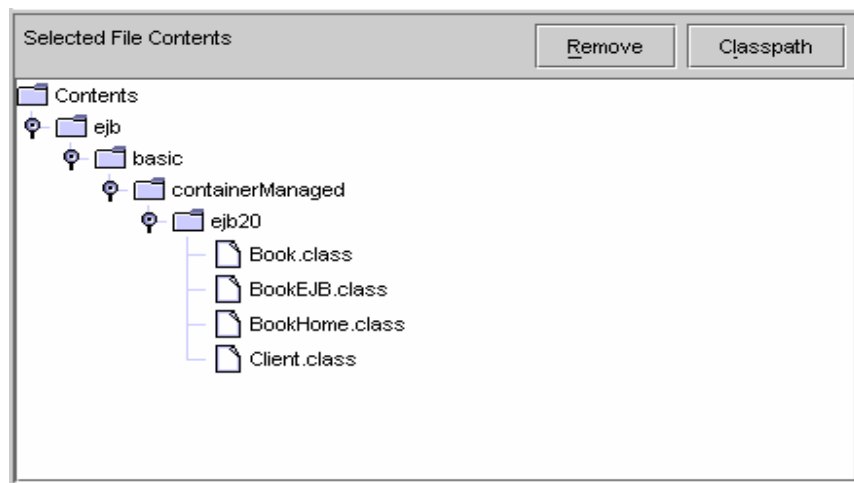


그림 29 파일 추가-후 Selected File Contents 화면

14. 추가한 파일을 삭제하고 싶을 경우에는 **Selected File Contents** 의 트리화면에서 대상 파일을 선택한 다음 **Remove** 버튼을 누르면 된다. 그리고 외부 클래스에 대한 접근이 필요할 때는 **Classpath** 버튼을 클릭하여 관련 데이터를 입력하면 된다(3.4.2.3 참조).
15. **File Contents** 대화상자에서 파일 추가와 관련된 작업을 마쳤다면 **Ok** 버튼을 클릭한다.
16. JEUS DD 파일인 jeus-ejb-dd.xml 을 자동으로 생성한다. DD 파일이 생성되는 위치는 JEUS Builder 의 작업 디렉토리인 JEUS\_HOME \workspace\assembly 이다. 이렇게 생성된 DD 파일을 기반으로 하여 모듈트리 뷰와 워크스페이스 뷰 화면이 구성된다[그림 30].

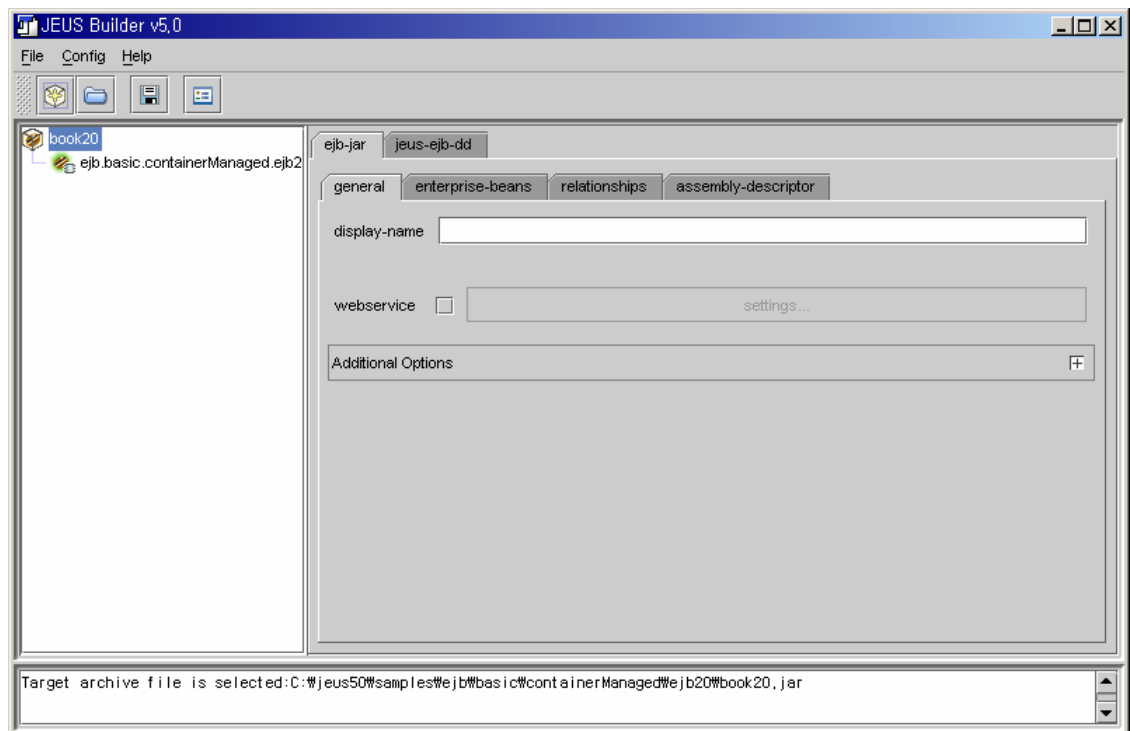


그림 30 파일추가후 전체화면

17. JEUS Builder 에 의해서 자동 생성된 두개의 DD 파일에는 모듈 디플로이에 필요한 충분한 정보가 포함되어 있다. 그러나 워크스페이스 뷰의 내용을 검토함으로써 의도되지 않거나 누락된 데이터가 있는지 반드시 확인하여 수정한다. 특히 **ejb-jar** 탭화면에서 **primkey-field** 를 매번 확인해주기 바란다.
18. 수정한 후에는 반드시 모듈을 저장해야 된다.
19. 그리고 현재 모듈에 대한 작업이 완료되었다면, 모듈의 열기 또는 모듈 새로 만들기를 통하여 다른 모듈로의 이동은 언제든지 자유롭다.

#### 4.1.1 생성된 모듈과 관련 된 파일

<<BookHome.java>>

```
package ejb.basic.containerManaged.ejb20;

import java.rmi.RemoteException;

import java.util.*;
```

```
import javax.ejb.CreateException;
import javax.ejb.EJBHome;
import javax.ejb.FinderException;

public interface BookHome extends EJBHome
{
    public Book create(String code, String title, String author,
double price,
        String publisher) throws CreateException,
RemoteException;

    public Book findByPrimaryKey(String code)
        throws FinderException, RemoteException;

    public Collection findByTitle(String title)
        throws FinderException, RemoteException;

    public Collection findAll() throws FinderException,
RemoteException;
}
```

## &lt;&lt;Book.java&gt;&gt;

```
package ejb.basic.containerManaged.ejb20;

import java.rmi.RemoteException;

import javax.ejb.EJBObject;

public interface Book extends EJBObject
{
    public String getTitle() throws RemoteException;

    public void setTitle(String title) throws RemoteException;
}
```

```
public String getAuthor() throws RemoteException;

public void setAuthor(String author) throws RemoteException;

public double getPrice() throws RemoteException;

public void setPrice(double price) throws RemoteException;

public String getPublisher() throws RemoteException;

public void setPublisher(String publisher) throws
RemoteException;

public String toBookString() throws RemoteException;
}
```

<<BookEJB.java>>

```
package ejb.basic.containerManaged.ejb20;

import javax.ejb.EntityBean;
import javax.ejb.EntityContext;

public abstract class BookEJB implements EntityBean
{
    public BookEJB() {}

    public String ejbCreate(String code, String title, String
author,
        double price, String publisher)
    {
        setCode(code);
        setTitle(title);
        setAuthor(author);
        setPrice(price);
        setPublisher(publisher);
        return null;
    }
}
```

```
}

    public void ejbPostCreate(String code, String title, String
author,
        double price, String publisher){}

    public abstract String getCode();
    public abstract void setCode(String code);
    public abstract String getTitle();
    public abstract void setTitle(String title);
    public abstract String getAuthor();
    public abstract void setAuthor(String author);
    public abstract double getPrice();
    public abstract void setPrice(double price);
    public abstract String getPublisher();
    public abstract void setPublisher(String publisher);

    public String toBookString()
    {
        return getCode() + "- [" + getTitle() + "] by " +
getAuthor() + " | " +
        getPrice() + " | " + getPublisher();
    }

    public void setEntityContext(EntityContext ctx){}
    public void unsetEntityContext(){}
    public void ejbLoad(){}
    public void ejbStore(){}
    public void ejbActivate()
    {
        System.out.println("");
        System.out.println("==== EJB Activation =====");
        System.out.println("");
    }

    public void ejbPassivate()
    {
        System.out.println("");
        System.out.println("==== EJB Passivation =====");
```

```

        System.out.println("");
    }

    public void ejbRemove()
    {
        System.out.println("");
        System.out.println("==== EJB Remove =====");
        System.out.println("");
    }
}

```

### <<ejb-jar.xml>>

```

<?xml version="1.0"?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD
Enterprise JavaBeans 2.0//EN" 'http://java.sun.com/dtd/ejb-
jar_2_0.dtd'>
<ejb-jar>
    <enterprise-beans>
        <entity>
            <ejb-name>BookBean</ejb-name>
            <home>ejb.basic.containerManaged.ejb20.BookHome</home>
            <remote>ejb.basic.containerManaged.ejb20.Book</remote>
            <ejb-class>
                ejb.basic.containerManaged.ejb20.BookEJB
            </ejb-class>
            <persistence-type>Container</persistence-type>
            <prim-key-class>java.lang.String</prim-key-class>
            <primkey-field>code</primkey-field>
            <reentrant>False</reentrant>
            <cmp-version>2.x</cmp-version>
            <abstract-schema-name>Book</abstract-schema-name>
            <cmp-field><field-name>code</field-name></cmp-field>
            <cmp-field><field-name>title</field-name></cmp-field>
            <cmp-field><field-name>author</field-name></cmp-
field>
            <cmp-field><field-name>price</field-name></cmp-field>

```



```

        <cmp-field><field-name>publisher</field-name></cmp-
field>
        <query>
            <query-method>
                <method-name>findByTitle</method-name>
                <method-params>
                    <method-param>java.lang.String</method-
param>
                </method-params>
            </query-method>
            <ejb-ql>
                SELECT OBJECT(b) FROM Book b WHERE
b.title = ?1
            </ejb-ql>
        </query>
        <query>
            <query-method>
                <method-name>findAll</method-name>
                <method-params/>
            </query-method>
            <ejb-ql>
                SELECT OBJECT(b) FROM Book b
            </ejb-ql>
        </query>
    </entity>
</enterprise-beans>
<assembly-descriptor>
    <container-transaction>
        <method>
            <ejb-name>BookBean</ejb-name>
            <method-name>*</method-name>
            <method-params/>
        </method>
        <trans-attribute>Required</trans-attribute>
    </container-transaction>
</assembly-descriptor>
</ejb-jar>

```

### <<jeus-ejb-dd.xml>>

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<jeus-ejb-dd xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
  <beanlist>
    <jeus-bean>
      <ejb-name>BookBean</ejb-name>
      <export-name>bookcmp20</export-name>
      <export-port>0</export-port>
      <single-vm-only>false</single-vm-only>
      <local-invoke-optimize>false</local-invoke-optimize>
      <thread-max>200</thread-max>
      <pooling-bean>false</pooling-bean>
      <object-management>
        <bean-pool>
          <pool-min>4</pool-min>
          <pool-max>5</pool-max>
          <resizing-period>1800000</resizing-period>
        </bean-pool>
        <connect-pool>
          <pool-min>2</pool-min>
          <pool-max>3</pool-max>
          <resizing-period>1800000</resizing-period>
        </connect-pool>
        <capacity>2000</capacity>
        <passivation-timeout>20000</passivation-timeout>
        <disconnect-timeout>60000</disconnect-timeout>
      </object-management>
      <persistence-optimize>
        <engine-type>SINGLE_OBJECT</engine-type>
        <non-modifying-method>
          <method-name>getTitle</method-name>
        </non-modifying-method>
        <non-modifying-method>
          <method-name>getAuthor</method-name>
        </non-modifying-method>
        <non-modifying-method>
          <method-name>getPrice</method-name>
        </non-modifying-method>
      </persistence-optimize>
    </jeus-bean>
  </beanlist>
</jeus-ejb-dd>
  
```

```

        <non-modifying-method>
            <method-name>getPublisher</method-name>
        </non-modifying-method>
        <non-modifying-method>
            <method-name>toBookString</method-name>
        </non-modifying-method>
        <entity-cache-size>0</entity-cache-size>
        <update-delay-till-tx>true</update-delay-till-tx>
    </persistence-optimize>
    <schema-info>
        <table-name>Bookbascmp2</table-name>
        <cm-field>
            <field>code</field>
        </cm-field>
        <cm-field>
            <field>title</field>
        </cm-field>
        <cm-field>
            <field>author</field>
        </cm-field>
        <cm-field>
            <field>price</field>
        </cm-field>
        <cm-field>
            <field>publisher</field>
        </cm-field>
        <deleting-table>true</deleting-table>
        <db-vendor>oracle</db-vendor>
        <data-source-name>SampleDS</data-source-
name>

        </schema-info>
        <database-insert-delay>ejbPostCreate</database-
insert-delay>

        <enable-instant-ql>false</enable-instant-
ql>

        <max-message>10</max-message>
    </jeus-bean>
</beanlist>
</jeus-ejb-dd>

```

## 4.2 Webservice 를 사용하는 WEB 모듈 생성

WEB 모듈 생성 역시 이전 섹션에서 설명한 EJB 모듈만큼이나 간단하다. 그래서 여기서는 좀 더 난해하게 웹서비스를 사용하는 WEB 모듈 생성에 대해서 살펴보도록 하겠다.

생성과정을 간략히 살펴보면, 우선 관련 파일을 추가한 다음 WEB 모듈의 DD 파일에 대한 정보를 설정한 다음, 마지막으로 웹서비스 DD 파일을 설정한다.

앞으로 설명할 예제 역시 JEUS 설치 디렉토리의 예제 디렉토리에서 찾을 수 있다.

1. JEUS Builder 를 실행한다[그림 31].

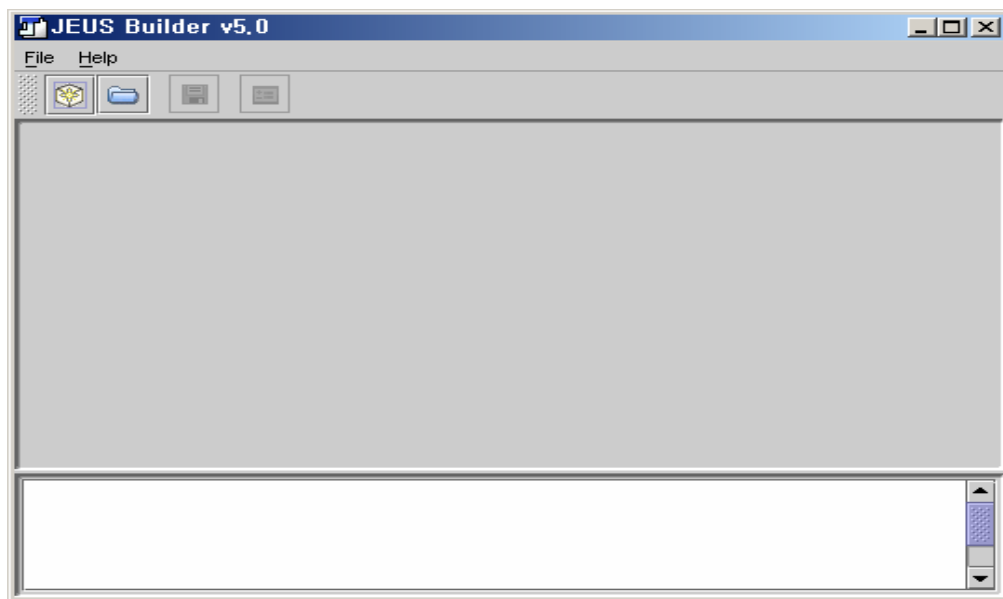


그림 31 JEUS Builder 초기화면

2. 모듈 새로 만들기를 선택한다. 모듈 새로 만들기는 **File->New** 메뉴를 선택하거나 툴바 아이콘(📁)을 선택하면 된다. 그러면 [그림 32]과 같이 파일을 추가할 수 있는 대화상자를 보게 될 것이다.

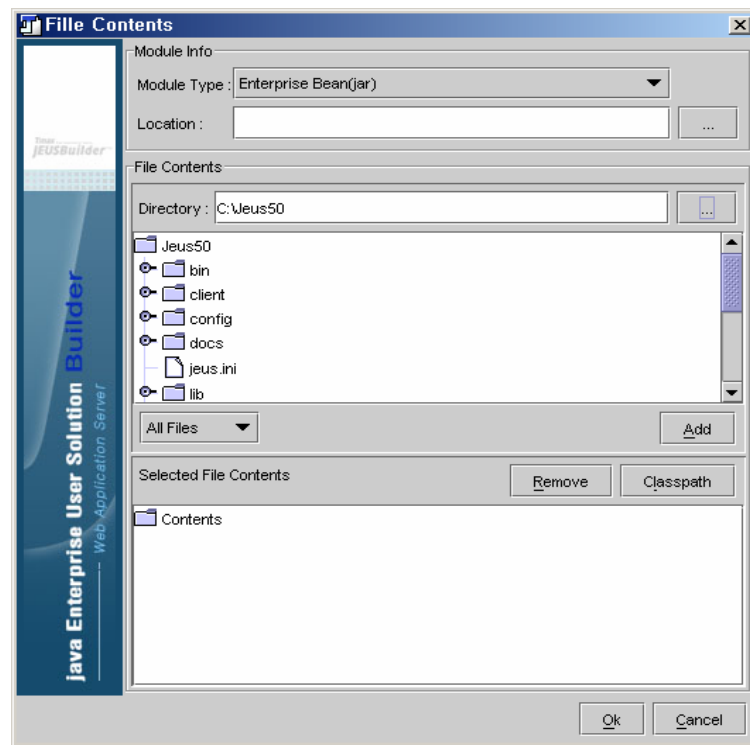


그림 32 File Contents 대화상자 초기화면

3. 우리가 WEB 모듈을 생성할 것이기 때문에 **Module Type** 콤보박스에서 맨 마지막에 있는 **Web Component(war)**를 선택한다.
4. **Location** 텍스트 필드에 저장할 모듈 경로를 지정하기 위해서 우선 오른쪽에 있는 브라우저 버튼을 클릭한다.
5. **Select Archive File Path** 대화상자에서 모듈을 저장하고 싶은 최종 디렉토리로 이동한 후 **File Name** 텍스트 필드에 employeeWebservice.war 를 입력하고 **Open** 버튼을 클릭한다[그림 33].

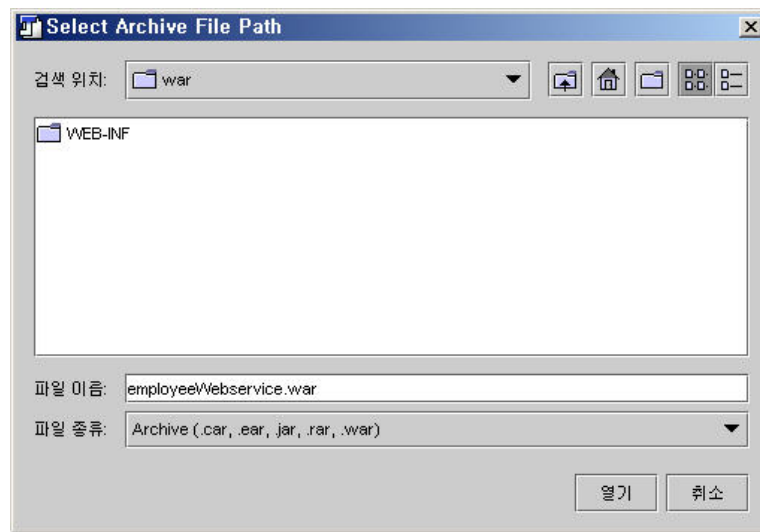


그림 33 모듈 저장위치 지정을 위한 대화상자

6. 3-5 과정을 성공적으로 수행했다면 **File Contents** 대화상자의 **Module Info** 의 내용은 다음과 같아야 된다[그림 34]. 한가지 주의해야 할 것은 **Location** 에 입력하는 모듈이름의 확장자는 반드시 선택한 **Module Type** 이 지원하는 것을 사용해야 된다. **WEB** 모듈에서 지원하는 모듈의 확장자는 .war 이다.

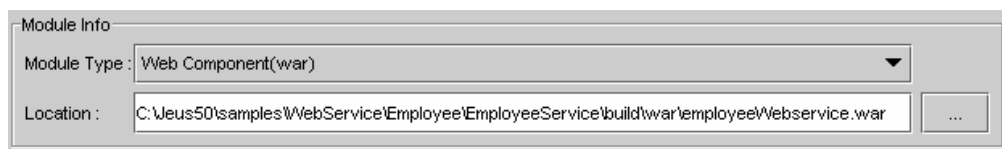


그림 34 Module Info 설정

7. **Directory** 텍스트필드에 추가 대상의 파일들이 있는 디렉토리를 지정하기 위해서 오른쪽에 있는 브라우저 버튼을 클릭한다.
8. **Select Work Directory** 에서 추가할 파일들이 존재하는 바로 직전의 디렉토리를 선택하고 대화상자의 **Open** 버튼을 클릭한다.
9. 7, 8 번과정을 잘 수행했다면 추가할 파일에 대한 목록을 트리형태로 볼 수 있을 것이다[그림 35]. 아래 그림은 트리를 말단 노드의 파일까지 볼수 있도록 펼쳐 놓은 형태를 보여주고 있다.

WEB 모듈을 생성하는 단계이기 때문에 DD 파일(web.xml, jeus-web-dd.xml)은 존재하지 않는다. 그러나, 현재 가지고 있는 DD 파일을 사용할려고 하면 작업디렉토리에 WEB-INF\web.xml, WEB-INF\jeus-web-dd.xml 을 생성하여 파일 추가시 같이 추가하면 된다.

그러면 JEUS Builder에서는 DD 파일을 새로 생성하지 않고 추가된 DD 파일을 그대로 이용한다.

서두에서 얘기했지만 우리가 지금 만들려고 하는 모듈은 웹서비스를 사용 하는 WEB 모듈을 만들려고 한다.

웹서비스를 지원하기 위해서 필요한 파일이 EmployeeService-mapping.xml 과 wsdl\EmployeeService.wsdl 이다. 언급한 두 파일에 대한 자세한 내용은 JEUS Webservice 안내서를 참조하기 바란다.

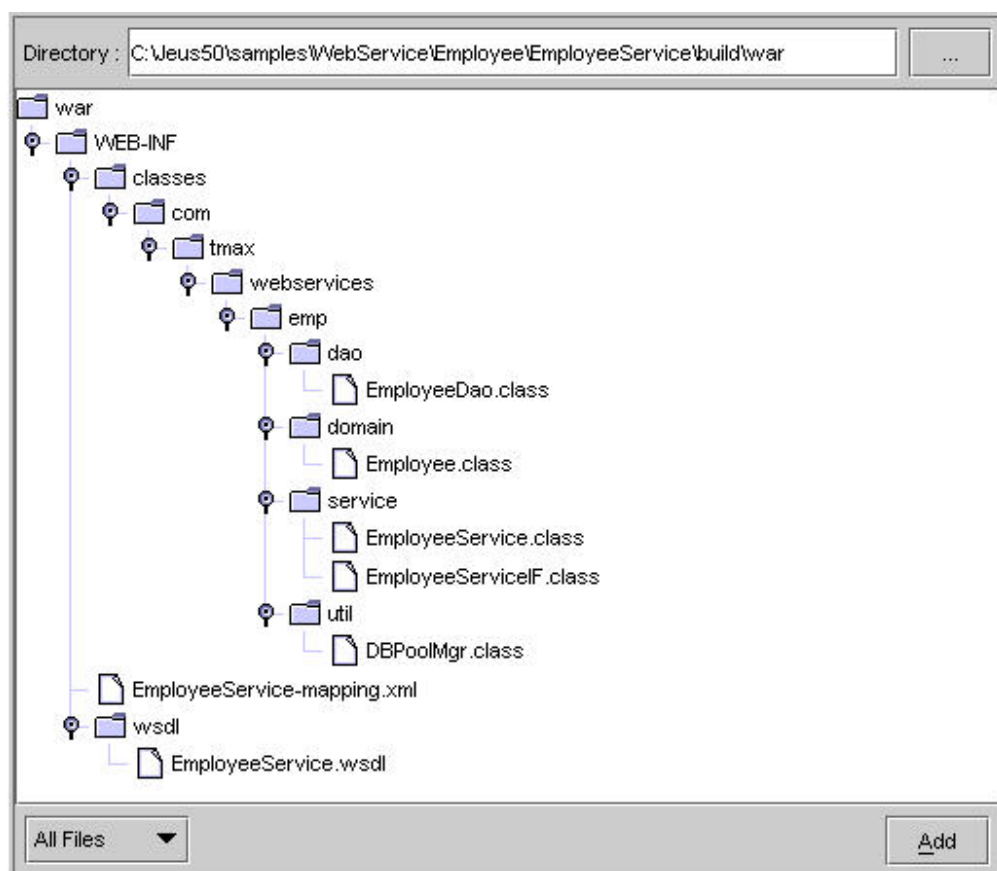


그림 35 추가 대상 파일 트리

10. **WEB-INF** 노드를 선택한 다음 **Add** 버튼을 클릭하여 파일을 추가하자. 한 개 이상의 디렉토리를 선택할 수 있다.
11. 추가된 파일트리의 구조는 추가 대상 파일의 트리 구조와 동일함을 확인할 수 있다[그림 36].

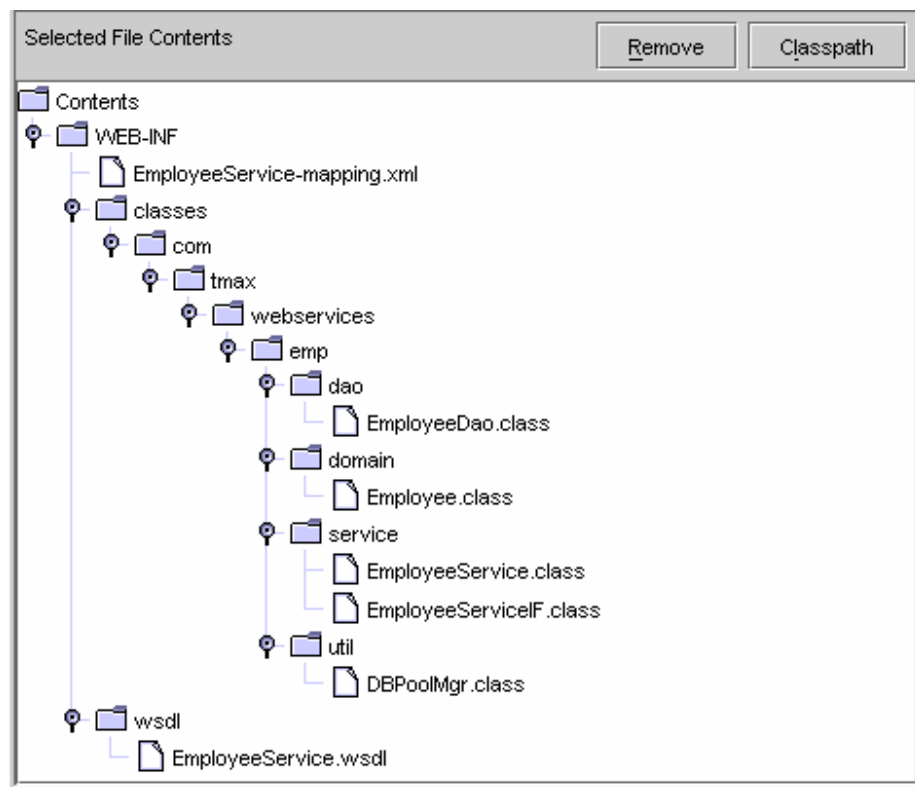


그림 36 추가된 파일 트리

12. 파일 추가 작업이 끝났으면 **File Contents** 대화상자의 **Ok** 버튼을 클릭한다.
13. WEB 모듈에서는 모듈트리뷰와 워크스페이스뷰가 JEUS Builder 에서 자동으로 생성한 DD 파일 기반으로 생성된 정보에 의해서 채워진다[그림 37]. WEB 모듈 생성시 JEUS Builder 에서 생성되는 DD 파일은 EJB 모듈 만들 때 생성되는 DD 파일에 비해서 정보가 많지 않다. 그래서 사용자 입력해야 되는 정보가 다수 있는데, 특히 servlet 관련부분을 주의 깊게 살펴 보아야 할 것이다.



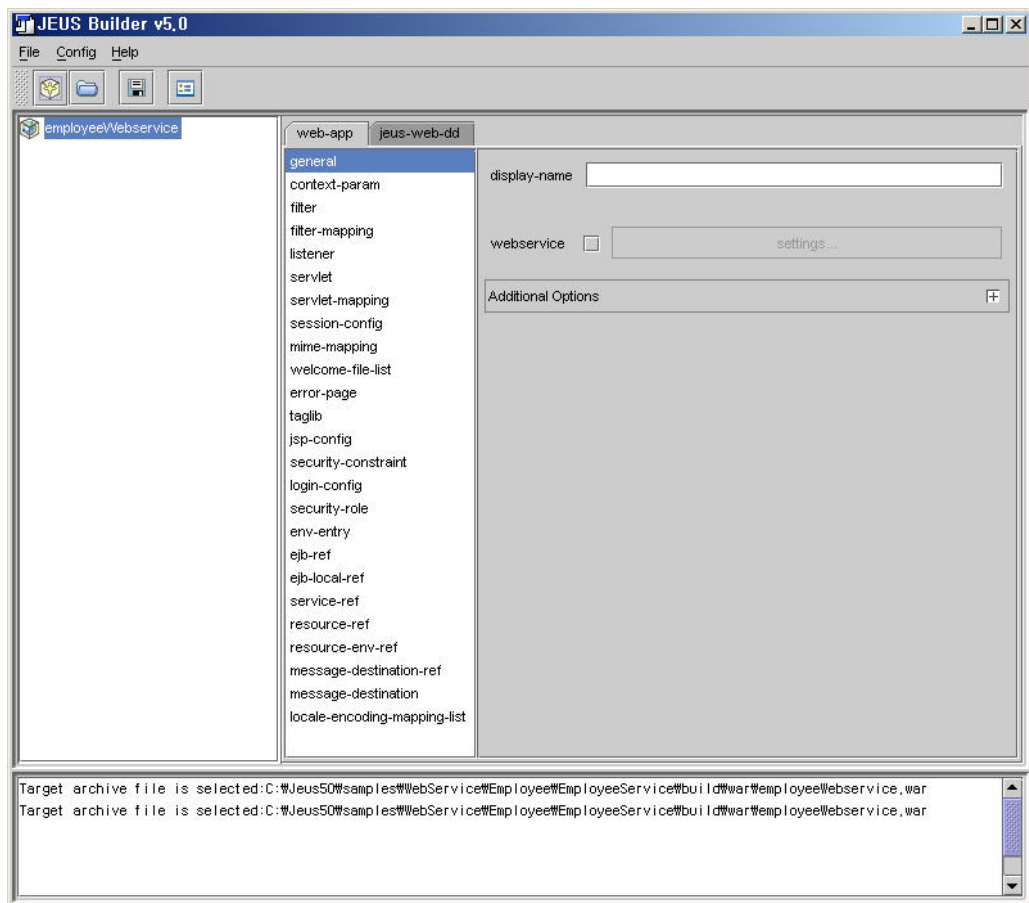


그림 37 파일 추가후 초기 화면

14. **web-app** 탭에서 **servlet** 을 클릭한다[그림 38].

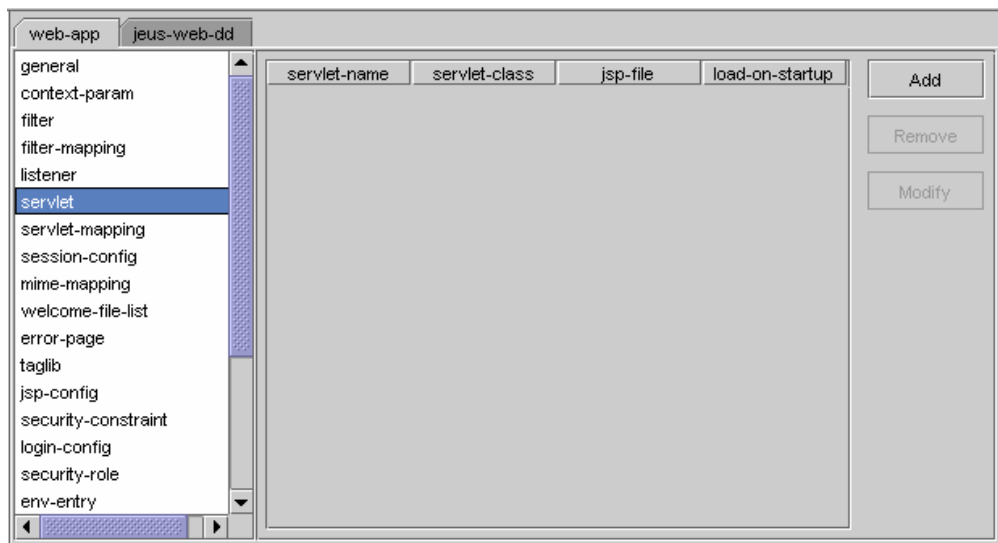


그림 38 servlet 추가 화면

15. servlet 을 추가시키기 위해서 **Add** 버튼을 클릭한다.
16. **servlet** 대화상자의 **general** 탭을 다음과 같이 입력한다.

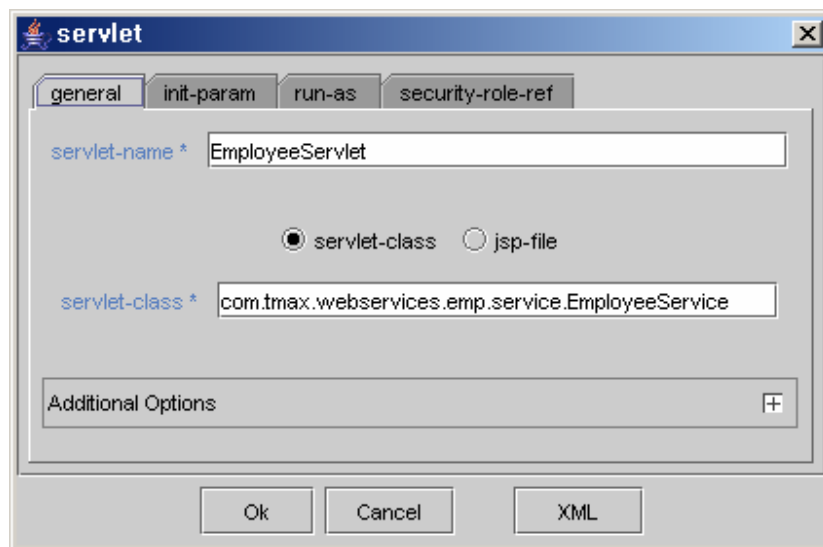


그림 39 servlet 추가 대화상자

17. **init-param** 탭에서 **Add** 버튼을 클릭하여 servlet class 의 로딩 제한시간인 **load-on-startup** 을 입력하고 **Ok** 버튼을 클릭한다[그림 40].

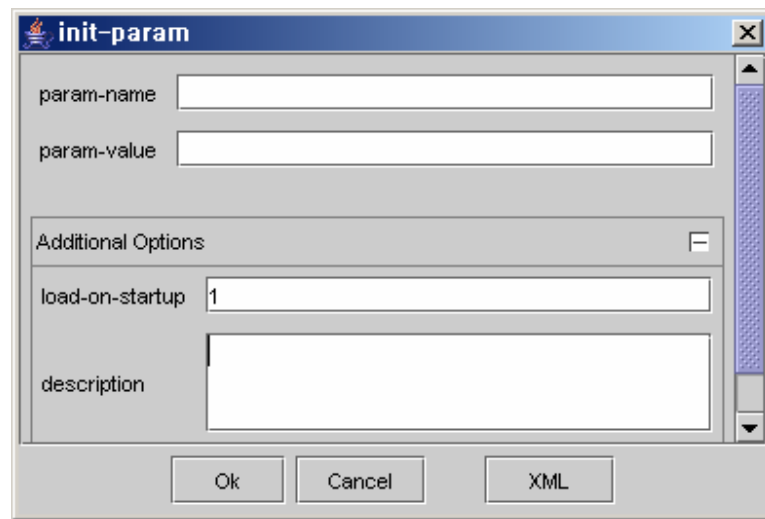


그림 40 init-param 입력 화면

18. **servlet** 대화상자에서 **Ok** 버튼을 클릭하면 모듈트리뷰에는 **servlet-name**(EmployeeServlet)을 노드이름으로 하는 트리노드가 루트노드 아래에 만들어지며, 워크스페이스뷰의 **servlet** 테이블에는 입력한 정보가 보여진다[그림 41]. 만일 모듈트리뷰에서 **servlet** 클래스의 노드(EmployeeServlet)가 보이지 않으면 루트노드를 더블클릭하면 보일 것이다.

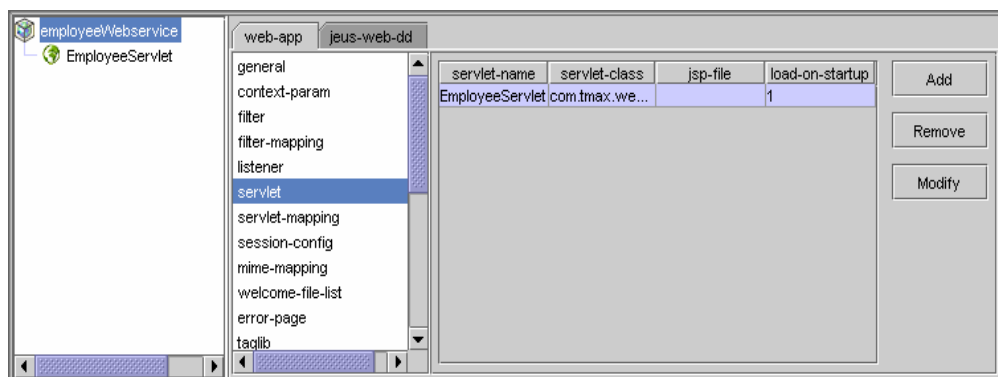


그림 41 servlet 추가후 모듈트리뷰와 워크스페이스뷰

19. 등록된 Servlet 과 URL 패턴을 매핑시키기 위해서 web-app 탭에 있는 **servlet-mapping** 을 선택한다[그림 42].

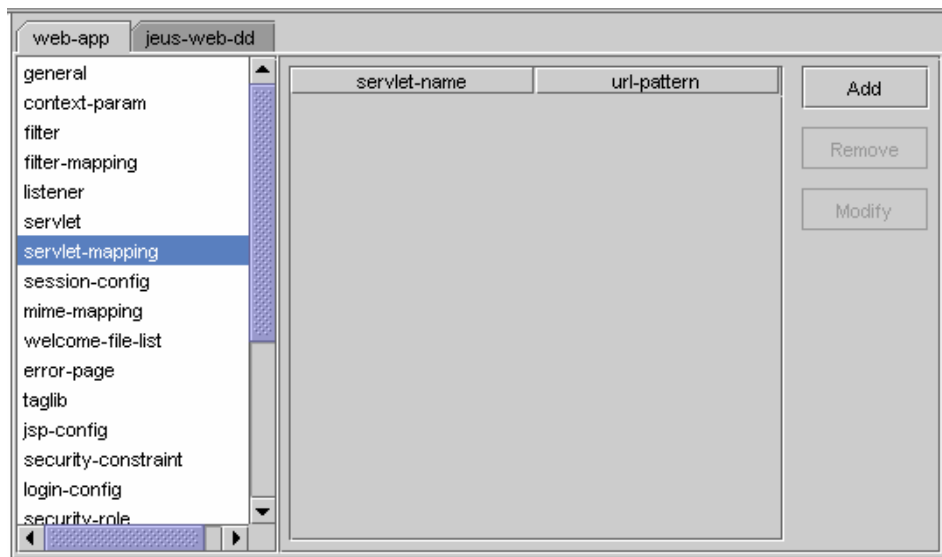


그림 42 web-app 탭에서 servlet-mapping 화면

20. **servlet-mapping** 에 정보를 추가시키기 위해 **Add** 버튼을 클릭한다.
21. 다음과 같이 **servlet-name** 과 **url-pattern** 을 입력한 다음 **Ok** 버튼을 클릭한다[그림 43].

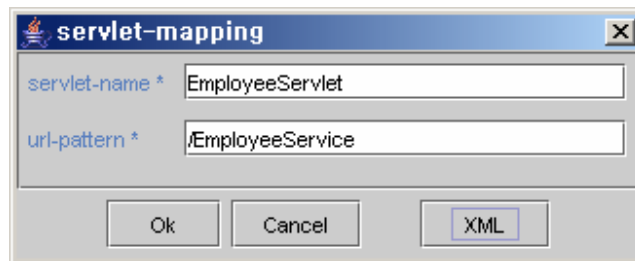


그림 43 servlet-mapping 대화상자

22. 입력한 정보는 다음과 같이 **servlet-mapping** 테이블에 표시된다[그림 44].

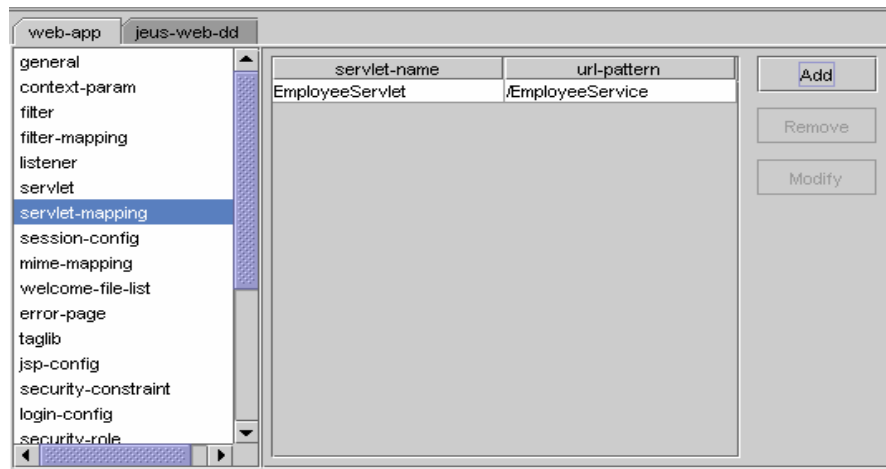


그림 44 setvlet-mapping table

23. 이제 jeus-web-dd 탭으로 이동하자. 아래 [그림 45]는 JEUS Builder 에서 자동으로 생성한 정보를 보여주고 있다. 사용자의 의도에 맞게 정보를 바꾸도록 한다.

여기까지 하면 기본적인 WEB 모듈을 생성하는 것을 본 것이다. 이제부터 웹서비스를 사용하기 위한 작업을 시작해 보자.

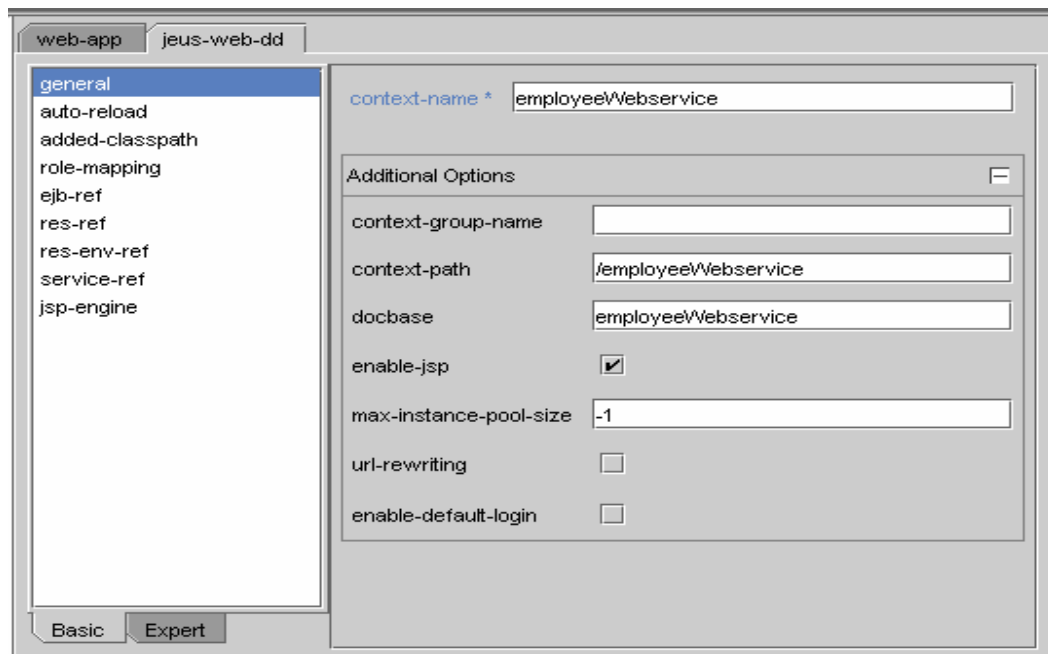


그림 45 jeus-web-dd 의 general

24. 웹서비스에 관련된 DD 파일을 만들기 위해서 **web-app** 탭의 **general** 항목을 클릭한다[그림 46].

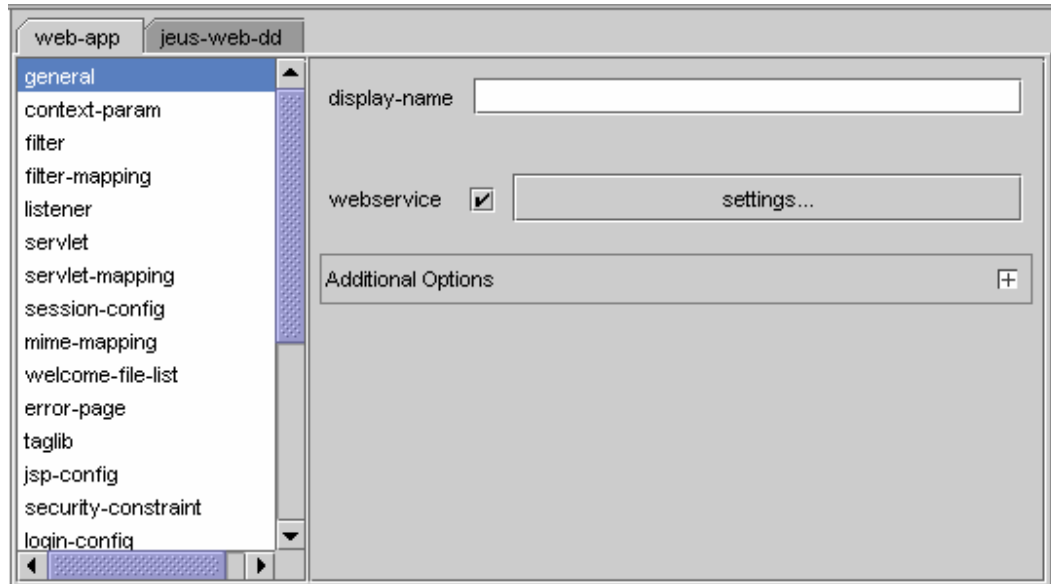


그림 46 웹모듈의 webservice 설정 버튼

25. 오른쪽에 있는 **webservice** 의 체크박스를 체크하고 **settings...**버튼을 클릭한다[그림 46].
26. **webservicex.xml** 과 **jeus-webservicex-dd.xml** 을 설정할 수 **Webservicex** 대 화상자가 팝업된다[그림 47].

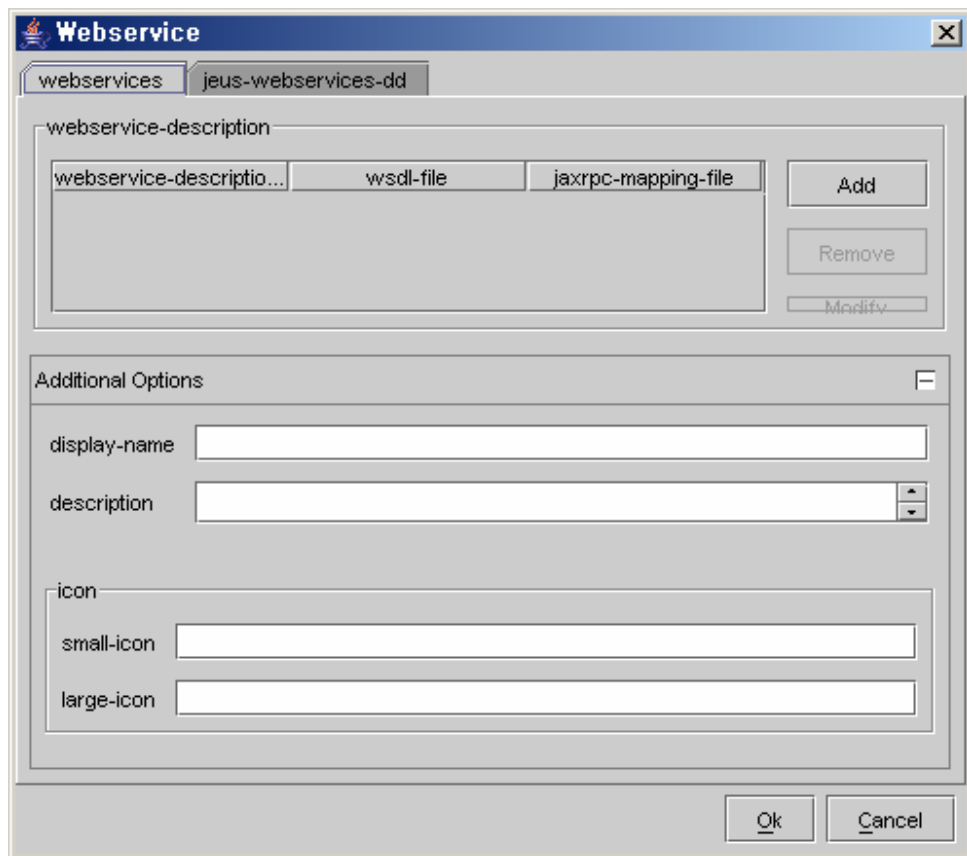


그림 47 웹서비스 설정 초기화면

27. **webservice-description** 대화상자의 위에 있는 **webservice-description-name**, **wsdl-file**, 그리고 **jaxrpc-mapping-file** 은 아래와 같이 입력한다[그림 48]. 입력하는 파일은 **File Contents** 대화상자에서 추가한 파일명이다.

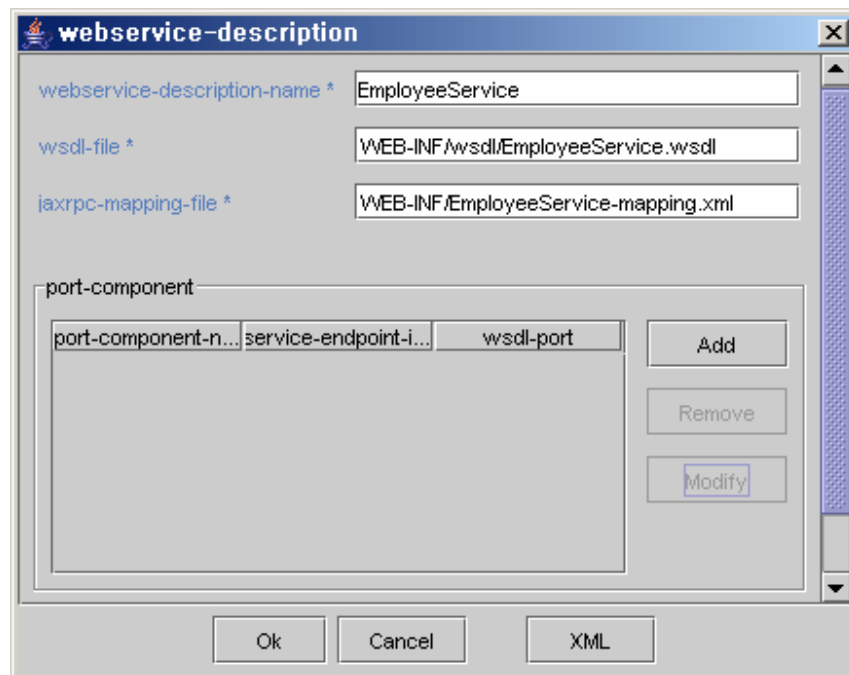


그림 48 webservicie-description 대화상자

28. **port-component** 정보를 입력하기 위해서 **Add** 버튼을 클릭한다.
29. **port-component** 대화상자에 있는 항목들에 대해서 다음과 같이 입력한다[그림 49].



그림 49 port-component 대화상자

30. 이제 웹서비스의 JEUS DD 파일의 설정을 위해서 **jeus-webservices-dd** 탭으로 이동한다.
31. **service** 정보를 등록하는 화면에 있는 **webservice-description-name** 테이블은 바로 직전에 작업한 **webservices** 탭화면의 **webservice-description** 테이블의 **webservice-description-name** 과 의존성을 가지고 있어서 동일한 값을 나타내고 있다[그림 50]. 그래서 여기서는 **Add** 버튼과 **Remove** 버튼이 없고 변경할 수 있는 **Modify** 버튼만 있다.

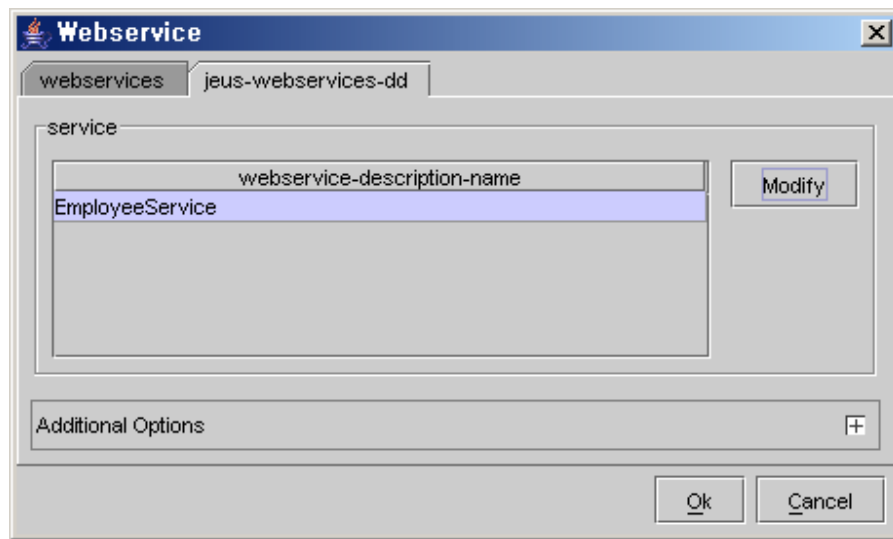


그림 50 jeus-webservice-dd 탭 화면

32. **service** 정보를 수정하기 위해서 **Modify** 버튼을 클릭한다.
33. **service** 대화상자에서 **webservises** 탭 화면과 의존성을 가지는 필드는 편집이 불가능하게 되어 있다. 그리고 **port** 정보 역시 수정만 가능하며 일부정보는 **webservises** 탭 화면과 의존성을 가지고 있다.

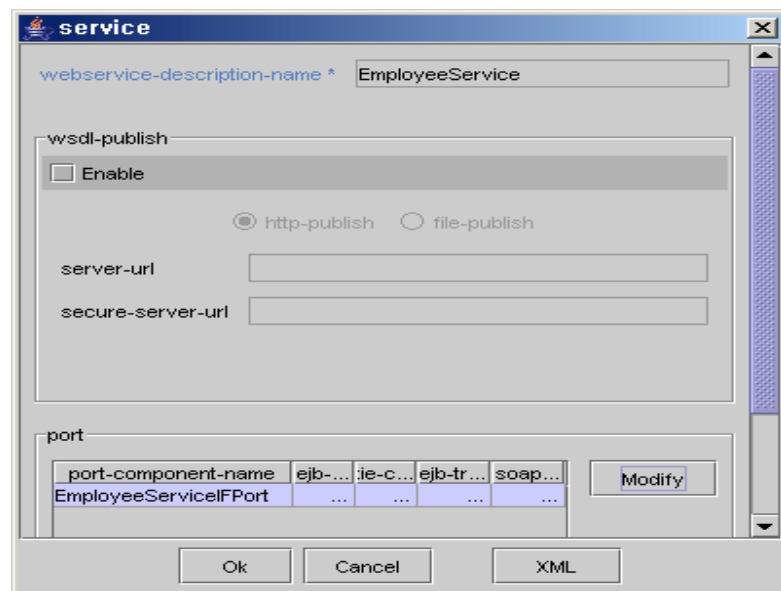


그림 51 jeus-webservises-dd 의 service 대화상자

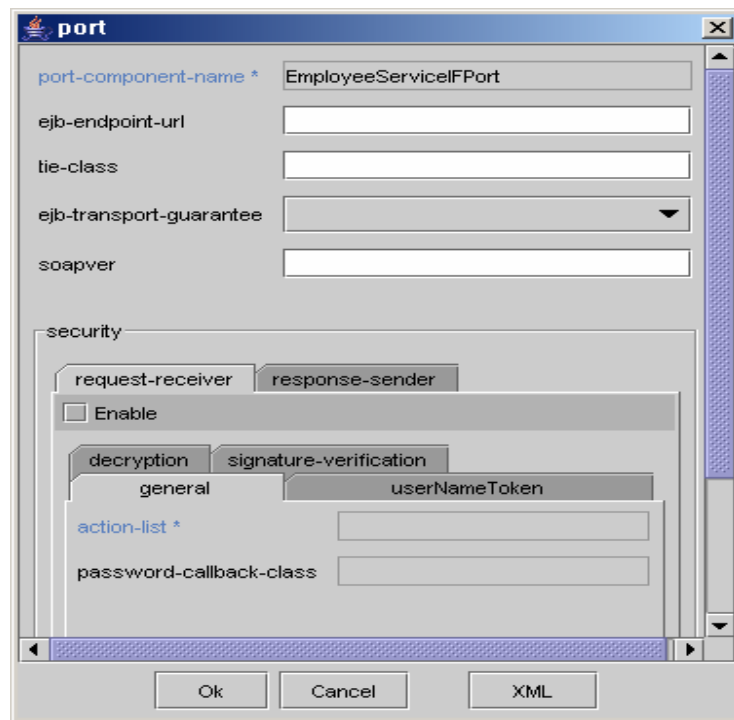


그림 52 jeus-webservices-dd 의 port 대화상자

34. **port** 정보 설정이 끝났다면 **Ok** 버튼을 클릭한다. **Service** 대화상자도 **Ok** 버튼을 클릭한다. 그리고 최종적으로 **Webservices** 대화상자의 **Ok** 버튼을 클릭한다. 그러면 웹서비스 관련 DD 파일이 생성될 것이다.
35. 웹서비스 관련 작업을 마치고 초기에 설정한 **WEB** 모듈의 **DD** 파일을 저장하자. 그러면 **File Contents** 의 **Module Info** 의 **location** 에 해당하는 경로에 모듈이 저장될 것이다.
36. JEUS Builder 를 종료함으로써 모듈 생성과정을 마치도록 한다.

#### 4.2.1 생성된 모듈과 관련 된 파일

<< EmployeeDao.java >>

```
package com.tmax.webservices.emp.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
```

```
import java.util.ArrayList;
import java.util.List;

import com.tmax.webservices.emp.domain.Employee;
import com.tmax.webservices.emp.util.DBPoolMgr;

public class EmployeeDao {

    DBPoolMgr dbMgr = DBPoolMgr.getDBPoolMgr("SampleDS");

    public EmployeeDao() {
        dbMgr = DBPoolMgr.getDBPoolMgr("SampleDS");
        if (dbMgr == null) {
            System.err.println("DBPoolMgr getInstance error...");
            return;
        }
    }

    public boolean insertEmployee(Employee emp) throws
SQLException {
        boolean flag = true;
        Connection conn = null;
        PreparedStatement pstmt = null;
        StringBuffer sb = null;

        try {
            conn = dbMgr.getConnection();
            sb = new StringBuffer();
            sb.append("INSERT INTO EMP (EMPNO, ENAME, JOB, MGR,
HIREDATE, SAL, COMM, DEPTNO) ");
            sb.append("VALUES ");
            sb.append("(?, ?, ?, ?, ?, ?, ?, ?)");

            pstmt = conn.prepareStatement(sb.toString());

            pstmt.clearParameters();

            pstmt.setString(1, emp.getEmpNo());
```

```
pstmt.setString(2, emp.getName());
pstmt.setString(3, emp.getJob());
pstmt.setString(4, emp.getManager());
pstmt.setString(5, emp.getHireDate());
pstmt.setString(6, emp.getSalary());
pstmt.setString(7, emp.getCommission());
pstmt.setString(8, emp.getDeptNo());

pstmt.executeUpdate();

} catch (SQLException e) {
    flag = false;
    e.printStackTrace();
} finally {
    resourceClose(null, pstmt, conn);
}

return flag;
}

public boolean updateEmployee(Employee emp) throws
SQLException{
    boolean flag = true;
    Connection conn = null;
    PreparedStatement pstmt = null;
    StringBuffer sb = null;

    try {
        conn = dbMgr.getConnection();
        sb = new StringBuffer();
        sb.append("UPDATE EMP SET ");
        sb.append("ENAME=?, JOB=?, MGR=?, HIREDATE=?, SAL=?,
COMM=?, DEPTNO=? ");
        sb.append("WHERE ");
        sb.append("EMPNO = ?");

        pstmt = conn.prepareStatement(sb.toString());

        pstmt.clearParameters();
```

```

        pstmt.setString(1, emp.getName());
        pstmt.setString(2, emp.getJob());
        pstmt.setString(3, emp.getManager());
        pstmt.setString(4, emp.getHireDate());
        pstmt.setString(5, emp.getSalary());
        pstmt.setString(6, emp.getCommission());
        pstmt.setString(7, emp.getDeptNo());
        pstmt.setString(8, emp.getEmpNo());

        pstmt.executeUpdate();

    } catch (SQLException e) {
        flag = false;
        e.printStackTrace();
    } finally {
        resourceClose(null, pstmt, conn);
    }

    return flag;
}

public boolean deleteEmployee(String empNo) throws
SQLException{
    boolean flag = true;
    Connection conn = null;
    PreparedStatement pstmt = null;
    StringBuffer sb = null;

    try {
        conn = dbMgr.getConnection();
        sb = new StringBuffer();
        sb.append("DELETE FROM EMP ");
        sb.append("WHERE ");
        sb.append("EMPNO = ? ");

        pstmt = conn.prepareStatement(sb.toString());

        pstmt.clearParameters();

        pstmt.setString(1, empNo);

```

```
        pstmt.executeUpdate();

    } catch (SQLException e) {
        flag = false;
        e.printStackTrace();
    } finally {
        resourceClose(null, pstmt, conn);
    }

    return flag;
}

public Employee selectEmployee(String empNo) throws
SQLException{
    Employee emp = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet rset = null;
    StringBuffer sb = null;

    try {
        conn = dbMgr.getConnection();
        sb = new StringBuffer();
        sb.append("SELECT EMPNO, ENAME, JOB, MGR, HIREDATE,
SAL, COMM, DEPTNO ");
        sb.append(" FROM EMP");
        sb.append(" WHERE EMPNO = ").append(empNo);

        stmt = conn.createStatement();

        rset = stmt.executeQuery(sb.toString());

        if (rset.next()) {
            emp = new Employee();
            emp.setEmpNo(rset.getString("EMPNO"));
            emp.setName(rset.getString("ENAME"));
            emp.setJob(rset.getString("JOB"));
            emp.setManager(rset.getString("MGR"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        resourceClose(null, stmt, conn);
    }

    return emp;
}
```

```

        emp.setHireDate(rset.getString("HIREDATE"));
        emp.setSalary(rset.getString("SAL"));
        emp.setCommission(rset.getString("COMM"));
        emp.setDeptNo(rset.getString("DEPTNO"));
    }

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        resourceClose(rset, stmt, conn);
    }

    return emp;
}

public Employee [] selectEmployees() throws SQLException{
    List employees = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet rset = null;
    StringBuffer sb = null;
    int cnt = 0;

    try {
        conn = dbMgr.getConnection();
        sb = new StringBuffer();
        sb.append("SELECT EMPNO, ENAME, JOB, MGR, HIREDATE,
SAL, COMM, DEPTNO ");
        sb.append(" FROM EMP");

        stmt = conn.createStatement();

        rset = stmt.executeQuery(sb.toString());

        Employee emp = null;
        employees = new ArrayList();

        while (rset.next()) {

```



```
        emp = new Employee();
        emp.setEmpNo(rset.getString("EMPNO"));
        emp.setName(rset.getString("ENAME"));
        emp.setJob(rset.getString("JOB"));
        emp.setManager(rset.getString("MGR"));
        emp.setHireDate(rset.getString("HIREDATE"));
        emp.setSalary(rset.getString("SAL"));
        emp.setCommission(rset.getString("COMM"));
        emp.setDeptNo(rset.getString("DEPTNO"));

        employees.add(emp);
        cnt++;
    }

} catch (SQLException e) {
    e.printStackTrace();
} finally {
    resourceClose(rset, stmt, conn);
}

Employee [] el = new Employee[cnt];
Employee e=null;
for(int k=0; k < cnt; k++)
{
    e = new Employee();
    e = (Employee)employees.get(k);
    el[k] = e;
}
return el;
}

private void resourceClose(ResultSet rset, Statement stmt,
Connection conn) {
    if (rset != null) {
        try {
            rset.close();
        } catch (SQLException e1) {
            // ignore ;
        }
    }
}
```

```

    }

    }

    if (stmt != null) {
        try {
            stmt.close();
        } catch (SQLException e2) {
            // ignore ;
        }
    }

    if (conn != null) {
        try {
            conn.close();
        } catch (SQLException e3) {
            // ignore ;
        }
    }
}
}

```

### << Employee.java >>

```

package com.tmax.webservices.emp.domain;

public class Employee {

    private String empNo;
    private String name;
    private String job;
    private String manager;
    private String hireDate;
    private String salary;
    private String commission;
    private String deptNo;

    public Employee() {

```

```
        super();
    }

    public String getCommission() {
        return commission;
    }

    public void setCommission(String commission) {
        this.commission = commission;
    }

    public String getDeptNo() {
        return deptNo;
    }

    public void setDeptNo(String deptNo) {
        this.deptNo = deptNo;
    }

    public String getEmpNo() {
        return empNo;
    }

    public void setEmpNo(String empNo) {
        this.empNo = empNo;
    }

    public String getHireDate() {
        return hireDate;
    }

    public void setHireDate(String hireDate) {
        this.hireDate = hireDate;
    }

    public String getJob() {
        return job;
    }
}
```

```
public void setJob(String job) {
    this.job = job;
}

public String getManager() {
    return manager;
}

public void setManager(String manager) {
    this.manager = manager;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getSalary() {
    return salary;
}

public void setSalary(String salary) {
    this.salary = salary;
}
}
```

<<EmployeeService.java>>

```
package com.tmax.webservices.emp.service;

import com.tmax.webservices.emp.dao.EmployeeDao;
import com.tmax.webservices.emp.domain.Employee;
import java.sql.SQLException;
```

```
public class EmployeeService {

    EmployeeDao dao = null;

    public EmployeeService() {
        dao = new EmployeeDao();
    }

    public void addEmployee(Employee emp) throws Exception {
        if (!dao.insertEmployee(emp)) {
            throw new Exception("Employee add fail");
        }
    }

    public void modifyEmployee(Employee emp) throws Exception {
        if (!dao.updateEmployee(emp)) {
            throw new Exception("Employee modify fail");
        }
    }

    public void removeEmployee(String empNo) throws Exception {
        if (!dao.deleteEmployee(empNo)) {
            throw new Exception("Employee remove fail");
        }
    }

    public Employee findEmployee(String empNo) throws Exception {
        Employee emp = dao.selectEmployee(empNo);
        if (emp == null) throw new Exception("Not exist
Employee");
        return emp;
    }

    public Employee [] findEmployees() throws SQLException {
        try {
            return dao.selectEmployees();
        } catch (SQLException e){
            throw new SQLException("Can not find employee");
        }
    }
}
```

```
    }  
}  
}
```

### <<EmployeeServiceIF.java>>

```
package com.tmax.webservices.emp.service;  
  
import java.rmi.Remote;  
import java.rmi.RemoteException;  
  
import com.tmax.webservices.emp.domain.Employee;  
  
public interface EmployeeServiceIF extends Remote {  
    public void addEmployee(Employee emp) throws RemoteException;  
    public void modifyEmployee(Employee emp) throws RemoteException;  
    public void removeEmployee(String empNo) throws RemoteException;  
    public Employee findEmployee(String empNo) throws  
RemoteException;  
    public Employee[] findEmployees() throws RemoteException;  
}
```

### <<DBPoolMgr.java>>

```
package com.tmax.webservices.emp.util;  
  
import java.sql.Connection;  
import java.sql.SQLException;  
import javax.naming.InitialContext;  
import javax.naming.NamingException;  
import javax.sql.DataSource;  
  
public class DBPoolMgr {  
    private static DBPoolMgr pool = null;  
    private DataSource ds = null;  
    private Connection con = null;  
    private String poolName = null;
```

```
private DBPoolMgr(String poolName) {
    setPoolName(poolName);
    initDataSource();
}

public static DBPoolMgr getDBPoolMgr(String poolName) {
    if(pool == null)
        pool = new DBPoolMgr(poolName);
    return pool;
}

public Connection getConnection() throws SQLException{
    if(ds == null) {
        initDataSource();
    }
    con = ds.getConnection();

    System.out.println("Connection rendering.....!!");

    return con;
}

private void initDataSource() {
    try{
        InitialContext ic = new InitialContext();
        ds = (DataSource)ic.lookup(this.getPoolName());
    }catch(NamingException ne) {
        System.out.println("NamingException error.." +
ne.toString());
    }
}

public String getPoolName() {
    return poolName;
}

public void setPoolName(String string) {
    poolName = string;
}
```

```
}  
  
}
```

### <<web.xml>>

```
<?xml version="1.0"?>  
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee">  
  <servlet>  
    <servlet-name>EmployeeServlet</servlet-name>  
    <servlet-class>  
      com.tmax.webservices.emp.service.EmployeeService  
    </servlet-class>  
    <load-on-startup>1</load-on-startup>  
  </servlet>  
  <servlet-mapping>  
    <servlet-name>EmployeeServlet</servlet-name>  
    <url-pattern>/EmployeeService</url-pattern>  
  </servlet-mapping>  
</web-app>
```

### <<jeus-web-dd.xml>>

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>  
<ns1:jeus-web-dd xmlns:ns1="http://www.tmaxsoft.com/xml/ns/jeus">  
  <ns1:context-path>/Employee</ns1:context-path>  
  <ns1:docbase>Employee</ns1:docbase>  
</ns1:jeus-web-dd>
```

### <<webservices.xml>>

```
<?xml version="1.0"?>  
<webservices version="1.1"  
  xmlns="http://java.sun.com/xml/ns/j2ee">  
  <web-service-description>
```



```

        <web-service-description-name>EmployeeService</web-service-
description-name>
        <wsdl-file>WEB-INF/wsdl/EmployeeService.wsdl</wsdl-file>
        <jaxrpc-mapping-file>WEB-INF/EmployeeService-
mapping.xml</jaxrpc-mapping-file>
        <port-component>
            <port-component-name>EmployeeServiceIFPort</port-
component-name>
            <wsdl-port
xmlns:ns2="urn:EmployeeService">ns2:EmployeeServiceIFPort</wsdl-
port>
                <service-endpoint-
interface>com.tmax.webservices.emp.service.EmployeeServiceIF</serv
ice-endpoint-interface>
                <service-impl-bean>
                    <servlet-link>EmployeeServlet</servlet-link>
                </service-impl-bean>
            </port-component>
        </web-service-description>
    </web-services>

```

## &lt;&lt;jeus-webservices-dd.xml&gt;&gt;

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<jeus-webservices-dd xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
    <service>
        <web-service-description-name>EmployeeService</web-service-
description-name>
        <port>
            <port-component-name>EmployeeServiceIFPort</port-
component-name>
        </port>
    </service>
</jeus-webservices-dd>

```

## &lt;&lt;EmployeeService.wsdl&gt;&gt;

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:EmployeeService"
xmlns:impl="urn:EmployeeService" xmlns:intf="urn:EmployeeService"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <schema targetNamespace="urn:EmployeeService"
xmlns="http://www.w3.org/2001/XMLSchema">
      <import
namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      <complexType name="Employee">
        <sequence>
          <element name="commission" type="xsd:string"/>
          <element name="deptNo" type="xsd:string"/>
          <element name="empNo" type="xsd:string"/>
          <element name="hireDate" type="xsd:string"/>
          <element name="job" type="xsd:string"/>
          <element name="manager" type="xsd:string"/>
          <element name="name" type="xsd:string"/>
          <element name="salary" type="xsd:string"/>
        </sequence>
      </complexType>
      <complexType name="ArrayOfEmployee">
        <sequence>
          <element maxOccurs="unbounded" minOccurs="0" name="item"
type="impl:Employee"/>
        </sequence>
      </complexType>
    </schema>
  </wsdl:types>
  <wsdl:message name="addEmployeeRequest">
    <wsdl:part name="in0" type="impl:Employee"/>
  </wsdl:message>
  <wsdl:message name="removeEmployeeRequest">
    <wsdl:part name="in0" type="xsd:string"/>
  </wsdl:message>

```

```

<wsdl:message name="findEmployeeResponse">
  <wsdl:part name="findEmployeeReturn" type="impl:Employee"/>
</wsdl:message>
<wsdl:message name="removeEmployeeResponse">
</wsdl:message>
<wsdl:message name="modifyEmployeeRequest">
  <wsdl:part name="in0" type="impl:Employee"/>
</wsdl:message>
<wsdl:message name="findEmployeesRequest">
</wsdl:message>
<wsdl:message name="findEmployeesResponse">
  <wsdl:part name="findEmployeesReturn"
type="impl:ArrayOfEmployee"/>
</wsdl:message>
<wsdl:message name="addEmployeeResponse">
</wsdl:message>
<wsdl:message name="modifyEmployeeResponse">
</wsdl:message>
<wsdl:message name="findEmployeeRequest">
  <wsdl:part name="in0" type="xsd:string"/>
</wsdl:message>
<wsdl:portType name="EmployeeServiceIF">
  <wsdl:operation name="addEmployee" parameterOrder="in0">
    <wsdl:input name="addEmployeeRequest "
message="impl:addEmployeeRequest"/>
    <wsdl:output name="addEmployeeResponse"
message="impl:addEmployeeResponse"/>
  </wsdl:operation>
  <wsdl:operation name="modifyEmployee" parameterOrder="in0">
    <wsdl:input name="modifyEmployeeRequest "
message="impl:modifyEmployeeRequest"/>
    <wsdl:output name="modifyEmployeeResponse"
message="impl:modifyEmployeeResponse"/>
  </wsdl:operation>
  <wsdl:operation name="removeEmployee" parameterOrder="in0">
    <wsdl:input name="removeEmployeeRequest "
message="impl:removeEmployeeRequest"/>
    <wsdl:output name="removeEmployeeResponse"
message="impl:removeEmployeeResponse"/>
  </wsdl:operation>
</wsdl:portType>

```

```

    </wsdl:operation>
    <wsdl:operation name="findEmployee" parameterOrder="in0">
      <wsdl:input name="findEmployeeRequest"
message="impl:findEmployeeRequest"/>
      <wsdl:output name="findEmployeeResponse"
message="impl:findEmployeeResponse"/>
    </wsdl:operation>
    <wsdl:operation name="findEmployees">
      <wsdl:input name="findEmployeesRequest"
message="impl:findEmployeesRequest"/>
      <wsdl:output name="findEmployeesResponse"
message="impl:findEmployeesResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="EmployeeServiceIFSoapBinding"
type="impl:EmployeeServiceIF">
    <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="addEmployee">
      <soap:operation soapAction="empAction"/>
      <wsdl:input name="addEmployeeRequest">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
      </wsdl:input>
      <wsdl:output name="addEmployeeResponse">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="modifyEmployee">
      <soap:operation soapAction="empAction"/>
      <wsdl:input name="modifyEmployeeRequest">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
      </wsdl:input>
      <wsdl:output name="modifyEmployeeResponse">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="removeEmployee">
      <soap:operation soapAction="empAction"/>
      <wsdl:input name="removeEmployeeRequest">

```

```

        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:input>
    <wsdl:output name="removeEmployeeResponse">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="findEmployee">
    <soap:operation soapAction="empAction"/>
    <wsdl:input name="findEmployeeRequest">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:input>
    <wsdl:output name="findEmployeeResponse">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:output>
</wsdl:operation>
<wsdl:operation name="findEmployees">
    <soap:operation soapAction="empAction"/>
    <wsdl:input name="findEmployeesRequest">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:input>
    <wsdl:output name="findEmployeesResponse">
        <soap:body use="literal" namespace="urn:EmployeeService"/>
    </wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="EmployeeService">
    <wsdl:port name="EmployeeServiceIFPort"
binding="impl:EmployeeServiceIFSoapBinding">
        <soap:address location="http://REPLACE_WITH_ACTUAL_URL"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

## &lt;&lt;EmployeeService-mapping.xml&gt;&gt;

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<java-wsdl-mapping version="1.1"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee

```

```

http://www.ibm.com/webservices/xsd/j2ee_jaxrpc_mapping_1_1.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/j2ee">
    <package-mapping>
        <package-type>com.tmax.webservices.emp.service</package-
type>
        <namespaceURI>urn:EmployeeService</namespaceURI>
    </package-mapping>
    <package-mapping>
        <package-type>com.tmax.webservices.emp.domain</package-
type>
        <namespaceURI>urn:EmployeeService</namespaceURI>
    </package-mapping>
    <java-xml-type-mapping>
        <java-type>com.tmax.webservices.emp.domain.Employee</java-
type>
        <root-type-qname
xmlns:typeNS="urn:EmployeeService">typeNS:Employee</root-type-
qname>
        <qname-scope>complexType</qname-scope>
        <variable-mapping>
            <java-variable-name>commission</java-variable-name>
            <xml-element-name>commission</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>deptNo</java-variable-name>
            <xml-element-name>deptNo</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>empNo</java-variable-name>
            <xml-element-name>empNo</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>hireDate</java-variable-name>
            <xml-element-name>hireDate</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>job</java-variable-name>
            <xml-element-name>job</xml-element-name>

```

```

        </variable-mapping>
        <variable-mapping>
            <java-variable-name>manager</java-variable-name>
            <xml-element-name>manager</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>name</java-variable-name>
            <xml-element-name>name</xml-element-name>
        </variable-mapping>
        <variable-mapping>
            <java-variable-name>salary</java-variable-name>
            <xml-element-name>salary</xml-element-name>
        </variable-mapping>
    </java-xml-type-mapping>
    <service-interface-mapping>
        <service-
interface>com.tmax.webservices.emp.service.EmployeeService</servic
e-interface>
        <wsdl-service-name
xmlns:serviceNS="urn:EmployeeService">serviceNS:EmployeeService</w
sdl-service-name>
        <port-mapping>
            <port-name>EmployeeServiceIFPort</port-name>
            <java-port-name>EmployeeServiceIFPort</java-port-name>
        </port-mapping>
    </service-interface-mapping>
    <service-endpoint-interface-mapping>
        <service-endpoint-
interface>com.tmax.webservices.emp.service.EmployeeServiceIF</serv
ice-endpoint-interface>
        <wsdl-port-type
xmlns:portTypeNS="urn:EmployeeService">portTypeNS:EmployeeServiceI
F</wsdl-port-type>
        <wsdl-binding
xmlns:bindingNS="urn:EmployeeService">bindingNS:EmployeeServiceIFS
oapBinding</wsdl-binding>
        <service-endpoint-method-mapping>
            <java-method-name>addEmployee</java-method-name>
            <wsdl-operation>addEmployee</wsdl-operation>

```

```

        <method-param-parts-mapping>
            <param-position>0</param-position>
            <param-
type>com.tmax.webservices.emp.domain.Employee</param-type>
            <wsdl-message-mapping>
                <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:addEmployeeRequest
</wsdl-message>
                <wsdl-message-part-name>in0</wsdl-message-
part-name>
                <parameter-mode>IN</parameter-mode>
            </wsdl-message-mapping>
        </method-param-parts-mapping>
        <wsdl-return-value-mapping>
            <method-return-value>void</method-return-value>
            <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:addEmployeeRespons
e</wsdl-message>
            </wsdl-return-value-mapping>
        </service-endpoint-method-mapping>
        <service-endpoint-method-mapping>
            <java-method-name>modifyEmployee</java-method-name>
            <wsdl-operation>modifyEmployee</wsdl-operation>
            <method-param-parts-mapping>
                <param-position>0</param-position>
                <param-
type>com.tmax.webservices.emp.domain.Employee</param-type>
                <wsdl-message-mapping>
                    <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:modifyEmployeeRequ
est</wsdl-message>
                    <wsdl-message-part-name>in0</wsdl-message-
part-name>
                    <parameter-mode>IN</parameter-mode>
                </wsdl-message-mapping>
            </method-param-parts-mapping>
            <wsdl-return-value-mapping>
                <method-return-value>void</method-return-value>

```



```

        <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:modifyEmployeeResp
onse</wsdl-message>
        </wsdl-return-value-mapping>
    </service-endpoint-method-mapping>
    <service-endpoint-method-mapping>
        <java-method-name>removeEmployee</java-method-name>
        <wsdl-operation>removeEmployee</wsdl-operation>
        <method-param-parts-mapping>
            <param-position>0</param-position>
            <param-type>java.lang.String</param-type>
            <wsdl-message-mapping>
                <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:removeEmployeeRequ
est</wsdl-message>
                <wsdl-message-part-name>in0</wsdl-message-
part-name>
                <parameter-mode>IN</parameter-mode>
            </wsdl-message-mapping>
        </method-param-parts-mapping>
        <wsdl-return-value-mapping>
            <method-return-value>void</method-return-value>
        <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:removeEmployeeResp
onse</wsdl-message>
        </wsdl-return-value-mapping>
    </service-endpoint-method-mapping>
    <service-endpoint-method-mapping>
        <java-method-name>findEmployee</java-method-name>
        <wsdl-operation>findEmployee</wsdl-operation>
        <method-param-parts-mapping>
            <param-position>0</param-position>
            <param-type>java.lang.String</param-type>
            <wsdl-message-mapping>
                <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:findEmployeeReques
t</wsdl-message>
                <wsdl-message-part-name>in0</wsdl-message-
part-name>

```

```

        <parameter-mode>IN</parameter-mode>
      </wsdl-message-mapping>
    </method-param-parts-mapping>
    <wsdl-return-value-mapping>
      <method-return-
value>com.tmax.webservices.emp.domain.Employee</method-return-
value>
      <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:findEmployeeRespon
se</wsdl-message>
      <wsdl-message-part-name>findEmployeeReturn</wsdl-
message-part-name>
    </wsdl-return-value-mapping>
  </service-endpoint-method-mapping>
  <service-endpoint-method-mapping>
    <java-method-name>findEmployees</java-method-name>
    <wsdl-operation>findEmployees</wsdl-operation>
    <wsdl-return-value-mapping>
      <method-return-
value>com.tmax.webservices.emp.domain.Employee []</method-return-
value>
      <wsdl-message
xmlns:wsdlMsgNS="urn:EmployeeService">wsdlMsgNS:findEmployeesRespo
nse</wsdl-message>
      <wsdl-message-part-name>findEmployeesReturn</wsdl-
message-part-name>
    </wsdl-return-value-mapping>
  </service-endpoint-method-mapping>
</service-endpoint-interface-mapping>
</java-wsdl-mapping>

```

## 4.3 결론

이번 장에서는 기본적인 EJB 모듈과 웹서비스를 사용하는 WEB 모듈을 생성하여 JEUS Builder의 쓰임새를 익혀보았다.

예를 들지 않은 다른 모듈의 경우에도 이와 같은 방식으로 접근하면 쉽게 원하는 작업을 할 수 있을 것이다.

JEUS Builder 의 사용법을 더 익히고 싶은 사용자들은 배포된 JEUS 에 있는 다양한 예제를 이용하기 바란다.



## 색 인

## ㄱ

모듈 열기, 18, 26, 27, 29, 30

## ㄴ

배치서술자, 10, 14, 16, 29, 30

뷰, 10, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 29, 30, 31,  
37, 38, 43, 44, 56, 59

## ㄷ

스키마, 38

## ㅇ

워크스페이스 뷰, 18, 19, 20, 21, 22, 23, 27, 29, 43

## ㅋ

클래스패스, 30, 32, 33, 39, 40, 53, 54

## A

Add, 34, 35, 43

Additional Options, 31, 32, 33, 37, 38, 39, 40, 42, 43,  
53, 54, 56, 63, 67

Application Client, 14, 15, 17, 19, 20, 21, 30, 32, 33,  
35, 36

Archive, 30, 37, 54

Archive 모드, 30, 37, 54

## E

EJB, 10, 14, 29, 30

Enterprise Bean, 14, 15, 17, 20

Expert, 21, 24, 25

## F

File Type, 33, 40, 41, 54

## J

J2EE DD, 12, 14, 16, 17, 18, 21, 22, 23, 24, 25, 26, 29,  
30, 43, 65

JAR, 18, 25, 48, 51, 82

jeusadmin, 11

## M

Manifest, 14, 15, 16, 17, 33, 53

META-INF, 35

**Module DD**, 12

Module Type, 35, 36

## R

Remove, 35, 54, 55, 81

## S

servlet, 23

## W

**WEB**, 10, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 29, 30,  
31, 37, 38, 43, 44, 56, 59

web-app, 59, 85

Web-app, 59, 85

webservice, 33, 36, 43, 48, 65