목 차

WAS(Web Application Server)와의 연동 ····· 2
개요 ····· 3
WAS에 대한 일반적인 서블릿 연동 절차 ····· 4
웹 로직과의 연동 ····· 9
WebSphere와의 연동 ····· 19
SilverStream과의 연동 ····· 48
Apache Jserv와의 연동 ····· 54
Tomcat과의 연동 ····· 56
Resin과의 연동 ····· 59
Inprise Application Server와의 연동 ····· 62
BES와의 연동 ····· 67
Oracel OC4J(ORION)와의 연동 ····· 68
HPAS(Bluestone)8과의 연동 ····· 68
JRun과의 연동 ····· 69
EAServer와의 연동 ····· 75
JEUS 3.0과의 연동 ····· 82
iPlanet 6.5와의 연동 ····· 84
Sun J2EE와의 연동 ····· 94

WAS(Web Application Server) 연동

- 개요
- WAS에 대한 일반적인 서블릿 연동 절차
- 웹로직과의 연동
- WebSphere와의 연동
- SilverStream과의 연동
- Apache Jserv와의 연동
- Tomcat과의 연동
- Resin과의 연동
- Inprise Application Server와의 연동
- BES와의 연동
- Oracel OC4J(ORION)와의 연동
- HPAS(Bluestone)8과의 연동
- JRun과의 연동
- EAServer와의 연동
- JEUS 3.0과의 연동
- iPlanet 6.5와의 연동
- Sun J2EE와의 연동

개요

오즈 엔터프라이즈 서버 서블릿 타입은 웹 애플리케이션 서버(이하 WAS라 칭함)와 연동하여 설정할 수 있습니다.

독자적으로 실행되는 데몬 타입의 오즈 엔터프라이즈 서버와는 달리 서블릿 타입의 오즈 엔터프라이즈 서버는 WAS에 탑재되어 WAS 위에서 동작하는 하나의 웹 애플리케이션(서블릿)으로 사용됩니다.

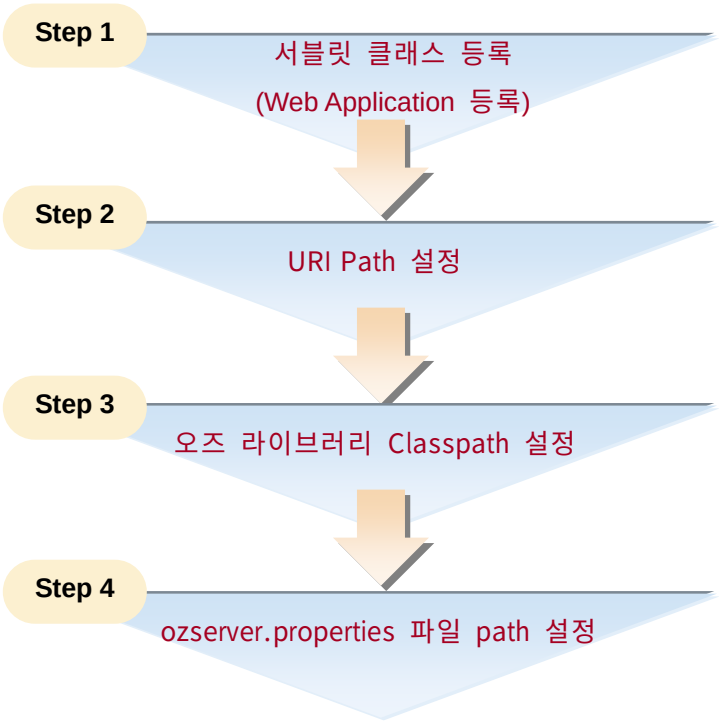
오즈 엔터프라이즈 서버 서블릿 제품은 다음과 같은 특성을 가집니다.

1. WAS가 이용하는 포트를 그대로 이용하게 됨으로 오즈 엔터프라이즈 서버를 이용하기 위한 별도의 TCP 포트 (데몬 제품의 경우 포트 번호 8003번)가 필요 없습니다. HTTP 프로토콜을 이용하기 때문에 보안상의 문제 등으로 별도의 TCP 포트를 사용할 수 없는 경우에 매우 유용합니다.
2. WAS가 이용하는 리소스를 공유해서 이용할 수 있습니다.(스레드풀, DB Pool, 데이터소스, 메모리 등) WAS가 이용하는 DB Pool을 오즈 엔터프라이즈 서버가 이용함으로써 데이터베이스 접속에 대한 시스템의 부담을 줄일 수 있고 WAS가 전체적으로 관리하기 때문에 관리자가 쉽게 관리할 수 있습니다.

오즈 엔터프라이즈 서블릿 제품은 데몬 제품과는 달리 설치 이후 반드시 서블릿과의 연동 절차(Deploy:디플로이)를 걸쳐야 합니다. 서블릿 연동 설정 이후에는 오즈 엔터프라이즈 서버는 WAS와 하나로 작동하게 됩니다. 예를 들어 WAS가 시작되면 오즈 엔터프라이즈 서버도 자동으로 실행되며, WAS가 종료되면 오즈 엔터프라이즈 서버도 함께 종료됩니다. WAS 중에서는 자신에 등록된 웹 애플리케이션을 구동시키거나 정지시키는 기능을 콘솔을 통해 제공하고 있는 것이 있습니다. 이때는 WAS의 설정에 따라 구동 및 정지됩니다.

WAS에 대한 일반적인 서블릿 연동 절차

오즈 엔터프라이즈 서버 서블릿 제품과 WAS를 연동하기 위해서는 다음의 절차를 거쳐야 합니다.



각 Step에는 여러 Case가 존재합니다. 해당 WAS에 따라 알맞은 Case를 참고하시기 바랍니다.

Step 1 서블릿 클래스 등록

■ **case1 : 오즈 서블릿은 WAS의 웹 애플리케이션으로 등록**

가장 일반적인 방법으로 오즈 엔터프라이즈 서버의 WEB-INF 디렉토리가 있는 path를 WAS에 등록합니다. WAS에 따라 관리자 콘솔이 제공되는 경우 제공되는 관리자 콘솔을 이용하여 등록 가능하며, 제공되지 않는 경우 설정 파일을 직접 편집해야 합니다.
(WAS별 설정 파일은 각 WAS별 설명을 참조하시기 바랍니다.)

■ **case2 : WAR(EAR) 파일 이용**

WEB-INF 디렉토리를 포함한 ozservlet 디렉토리를 통째로 묶은 WAR 파일 또는 EAR을 만든 후 WAS의 servlet configuration file에 만든 war 파일에 대한 정보를 설정합니다. WAS의 관리자 콘솔에서 User Interface를 제공하는 경우 이 User Interface를 이용하여 war 파일에 대한 정보를 입력하게 됩니다. WAR 및 EAR 파일을 만들때는 WAS에서 사용하고 있는 자바를 이용하여 만들어야 합니다. 자바 버전 및 벤더가 다를 경우 오작동 할 경우가 있습니다.

■ **case3 : 서블릿 실행 java class 등록**

실행할 서블릿의 package 및 class 이름 (oz.server.OZServlet)을 등록합니다. 이 경우 OZServlet 관련 jar 라이브러리들의 경로가 WAS 실행시 참조하는 classpath에 등록되어 있어야 합니다.

■ **case4 : 자동 등록(Auto Deploy)**

최신 WAS의 경우에는 정해진 Web Application 폴더에 오즈 서블릿을 복사(디렉토리 복사, WAR 복사)하면 자동으로 웹 애플리케이션으로 등록됩니다.

Step 2 **URI Path 설정**

■ **case1 : 오즈 엔터프라이즈 서버의 WEB-INF/web.xml 수정**

서블릿에 URI을 맵핑하는 표준적인 방법입니다. web.xml 파일을 열어 url-pattern 항목에 OZ 서블릿에게 할당할 URL을 입력합니다.

■ **case2 : 관리자 콘솔 이용**

WAS의 관리자 콘솔에서 URI 관련 메뉴를 찾아 설정해 줍니다.

■ **case3 : 기타**

WAS의 서블릿 설정 파일이나 URI 관련 설정을 직접 수정하여 설정해 줍니다.

Step 3 오즈 라이브러리 CLASSPATH 설정

- **case1 : WEB-INF/classes, WEB-INF/lib 자동 설정**

  WAR 파일 또는 WEB-INF 디렉토리를 등록한 경우 자동으로 필요한 라이브러리들이 classpath에 등록됩니다.

- **case2 : WAS 실행 CLASSPATH에 등록**

  Servlet Java Class를 직접 등록한 경우 또는 WEB-INF나 WAR 파일을 통해 설치할 수 없는 경우 직접 WAS의 CLASSPATH 환경변수에 OZ의 경로를 등록해 주어야 합니다. WAS 실행 Shell Script나 전역 환경 변수를 수정하여 오즈 서블릿에 필요한 라이브러리들을 CLASSPATH에 추가합니다.

- **case3 : 기타**

  WAS의 Configuration 파일에 서블릿용 라이브러리를 따로 설정하는 경우 그곳에 OZ 서블릿 라이브러리들의 PATH를 입력해 줍니다.

Step 4 ozserver.properties 파일 path 설정

- **case 1 : WEB-INF/web.xml 수정**

  init-param 태그를 OZSERVER_HOME으로 하고 value에 오즈 서블릿 홈 디렉토리의 절대 경로를 입력해 줍니다.

- **case2 : init parameter 직접 등록**

  WAS의 설정 파일이나 설정 메뉴에서 init parameter를 설정하는 곳을 찾아 태그 이름을 OZSERVER_HOME으로 하고 value에 ozserver home의 절대 경로를 입력해 줍니다.

- **case 3 : classpath에 등록**

  WAS에 정의된 classpath에 ozserver.properties 파일을 위치 시킵니다. (일반적으로 오즈 서블릿 홈 디렉토리 아래의 WEB-INF/classes 디렉토리를 사용합니다.)

■ **case 4 : servlet 실행 홈 디렉토리를 통해 등록**

등록된 오즈 서블릿의 URL에 접속을 시도 하였을 때 ozserver.properties 파일을 찾지 못할 경우 STDOUT 출력 로그 또는 웹 브라우저에 서블릿 실행 홈 디렉토리에 대한 절대 경로를 출력해 줍니다. 그 위치에 ozserver.properties 파일을 복사해 둡니다.

※ Case1부터 Case4까지의 방법 중 해당 WAS에 알맞은 방법을 선택하여 등록해 주고, 알맞은 WAS 방법을 찾지 못했을 경우 Case1부터 Case4까지 순서대로 세팅해 봅니다.

web.xml ({$OZ_HOME}/WEB-INF/web.xml)

URL상에서 사용할 오즈 서블릿과 OZA의 Alias를 생성, URL의 Alias 경로와 매핑시켜주는 역할을 하는 파일입니다. WAS에 오즈를 등록 후 오즈에서 사용하게될 URL Alias 등을 설정합니다.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">

<web-app>
        <servlet>
                <servlet-name>httpserver</servlet-name>
                <servlet-class>oz.server.OZServlet</servlet-class>

        </servlet>

        <servlet>
                <servlet-name>ozhello</servlet-name>
                <servlet-class>HelloWorld</servlet-class>
        </servlet>

    <servlet-mapping>
        <servlet-name> httpserver </servlet-name>
        <url-pattern>/server </url-pattern>
    </servlet-mapping>

    <servlet-mapping>
```

```
        <servlet-name>httpserver</servlet-name>
        <url-pattern>/server/*</url-pattern>
        <init-param>
            <param--name> OZSERVER_HOME </param-name>
        <param-value>c:\Program Files]Forcs\OZ Family\OZ Report 3.0\ozservlet
</param-value>
        <description> set ozserver home path </description>
      </init-param>
    </servlet-mapping>

    <servlet-mapping>
        <servle-name>ozhello</servlet-name>
        <url-pattern>/hello</url-pattern>
    </servlet-mapping>

</web-app>
```

위와 같이 배포되는 web.xml 파일은 별다른 설정없이 그대로 사용할 수 있습니다.

<url-pattern>/server</url-pattern> 부분이 URL에서 접근 가능한 오즈 서블릿의 Alias로 다음과 같은 형태로 접근 가능합니다.

http://ip:prot_no/oz/ozserver

```
************************************************
      OZ Server FrameWork Servlet 3.0.0101.20040303 20:22.120
                        copyright. FORCS Co., Ltd.
 ************************************************

  ---    OZ Version Info    ------------------------------------------
   OZ Server FrameWork Servlet : 3.0.0101.20040303 20:22.120

  ---    System Information    ----------------------------------------------------
          OS Name                : Windows 2000
          Version                : 5.0
          Java Vendor            : Sun Microsystems Inc.
          Version                : 1.3.1_07
          Class Version          : 1.3.1_07
          Servlet Container      : Jaguar Server Version 4.0
          Running Path           : null
          Ciper Suite            : null
          Certificates           : null
```

```
Session                 : null
system.IPs              : [0th] 218.36.12.71
                          [1th]211.116.251.102
```

상단의 OZ Version Info에서 오즈 서버 프레임웍 버전을 확인할 수 있습니다. OZ Server FrameWork Servlet 3.0.0101.20040303 20:22.120은 버전은 3.0이고 프로토콜 번호는 0101, 세부 버전 정보는 20040303 20:22.120임을 나타냅니다. 아래 부분의 System Information에는 해당 시스템에 대한 정보가 나타납니다. 해당 OS 종류(Windows 2000 / Solaris / AIX / HP-UX 등)와 해당 OS의 버전, JAVA Vendor, JAVA Version, Class version, Servlet Container(WAS 정보)가 나타나며, Running path 부분에는 현재 오즈 엔터프라이즈 서버가 설치된 곳의 경로가 나타납니다.

웹로직과의 연동

웹로직 6.1 이전

웹 애플리케이션이 등록될 파일 이름 : weblogic.properties

웹 애플리케이션이 등록될 형식 :

```
#### OZ Servlet
weblogic.httpd.webApp.oz=C:/program files/OZ/
```

웹로직의 weblogic.properties 파일을 열어 'weblogic.httpd.webApp.oz=' 라고 입력 한 후 오즈가 위치한 곳의 경로를 입력합니다.

■ **데이터베이스의 설정**

오즈 서블릿의 자체 DB Pooling을 사용할 경우 오즈 서버 데몬 버전에서와 같은 방법으로 오즈의 db.properties 파일을 설정합니다. 반면, 웹로직의 DB Pooling을 상속받아 사용하기 위해서는 weblogic.properties에서 웹로직이 정의한 Connection Pool을 찾아 해당 Pool의 이름을 오즈의 db.properties 파일에서 정의하여 사용하면 됩니다.

■ **weblogic.properties 파일의 DB Pool 정보**

※ 웹로직 6.1 이상에서는 웹로직 admin console에서 DB Pool을 설정합니다.

```
weblogic.jdbc.connectionPool.oracle_thin=\
```

```
url=jdbc:oracle:thin:@211.116.251.7:1521:forcs815,\
driver=oracle.jdbc.driver.OracleDriver,\
initialCapacity=1,\
maxCapacity=2,\
capacityIncrement=1,\
props=user=scott;password=tiger

weblogic.allow.reserve.weblogic.jdbc.connectionPool.oracle_thin=everyone
```

■ **weblogic.jdbc.connectionPool.oracle_thin=**

'oracle_thin'이 웹로직에서 사용하는 thin 드라이버의 Connection Pool 이름입니다. 그 이후의 모든 정보는 thin 드라이버를 사용하기 위해 기술해야 할 기본적인 항목들입니다. Weblogic.properties 파일에서 예제와 같이 마지막 라인이 있는지 반드시 확인해야 합니다. 만일 없거나 'everyone'으로 설정되어 있지 않으면, 웹로직 룰에 따라 오즈 서블릿이 해당 connection pool을 이용할 수 있도록 권한을 설정해 줍니다.

앞에 예시된 weblogic.properties 파일과 같이 설정된 오라클 DB Pool을 사용할 때 오즈의 db.properties 파일은 다음과 같이 설정합니다.

```
oacle1.vendor=user
oracle1.driver=weblogic.jdbc20.pool.Driver
oracle1.url=jdbc:weblogic:pool:oracle_thin
oracle1.serverAddress=211.116.251.7
oracle1.portNo=1521
oracle1.sid=forcs815
oracle1.user=scott
oracle1.password=tiger
oracle1.maxconns=5
oracle1.initconns=0
oracle1.timeout=5
oracle1.loglevel=debug
```

※ initconns는 반드시 0으로 설정합니다.

설치 후에는 확인 작업을 수행합니다. 과정은 다음과 같습니다.

오즈 엔터프라이즈 서버의 db.properties 파일을 설정할 때에는 현재 웹로직에서 사용하는 JDBC 드라이버가 표준 JDBC 2.0과 호환되는지를 반드시 확인해야 합니다.

Oracle의 경우에는 JDBC 드라이버중 classes111.zip은 표준 JDBC 2.0과 호환되지 않으며, classes12.zip이 표준 JDBC 2.0과 호환됩니다.

이와 관련된 부문은 ‘startWebLogic.sh/startWebLogic.cmd’에서 확인할 수 있습니다.
모든 환경 설정이 끝나면 웹로직을 재실행합니다.

인터넷 브라우저를 통해 다음의 URL로 확인할 수 있습니다.

```
http://Weblogic-ip:7001/oz/server
```

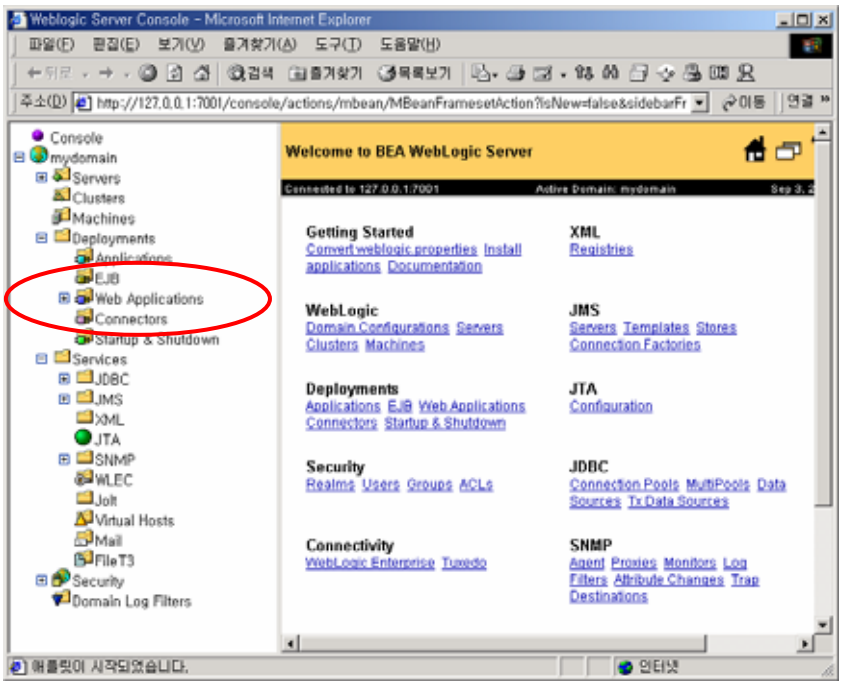
7001 : 웹로직에서 설정한 Port Number
/oz : weblogic.properties에서 정의한 오즈의 url path
/server : 오즈의 web.xml에서 정의된 오즈 서블릿의 url pattern

웹로직 6.1 이상

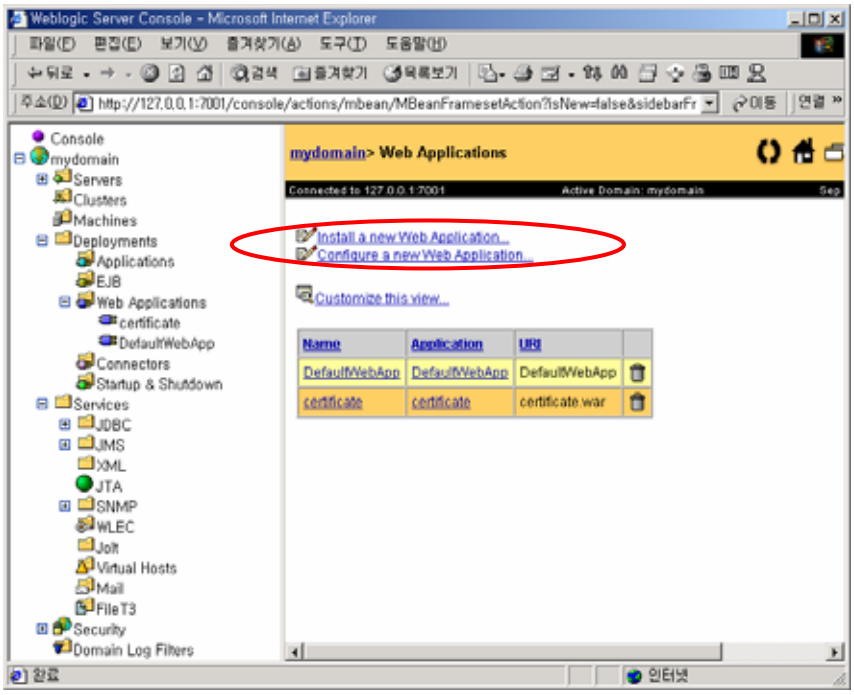
웹로직 6.1의 경우 웹로직에서 지원하는 관리자 콘솔을 이용하여 오즈 엔터프라이즈 서버를 등록합니다.

■ **환경설정 절차**

웹로직 서버를 실행 시킵니다. 웹로직 서버 콘솔을 실행 시킵니다.

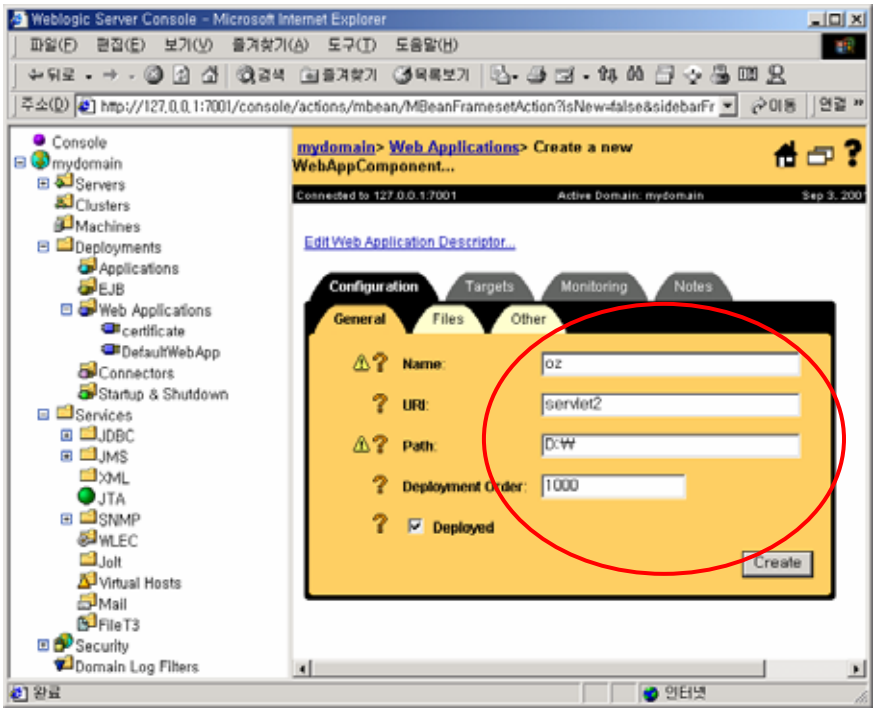


[Web Application]을 선택하여 오즈를 등록합니다.

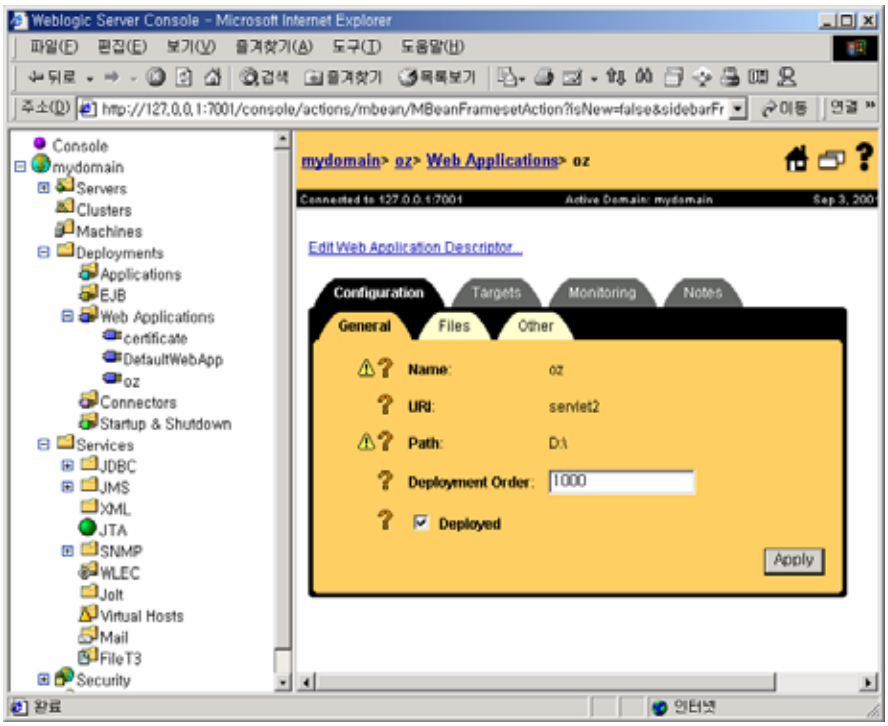


[configure a new Web Application]을 선택합니다.

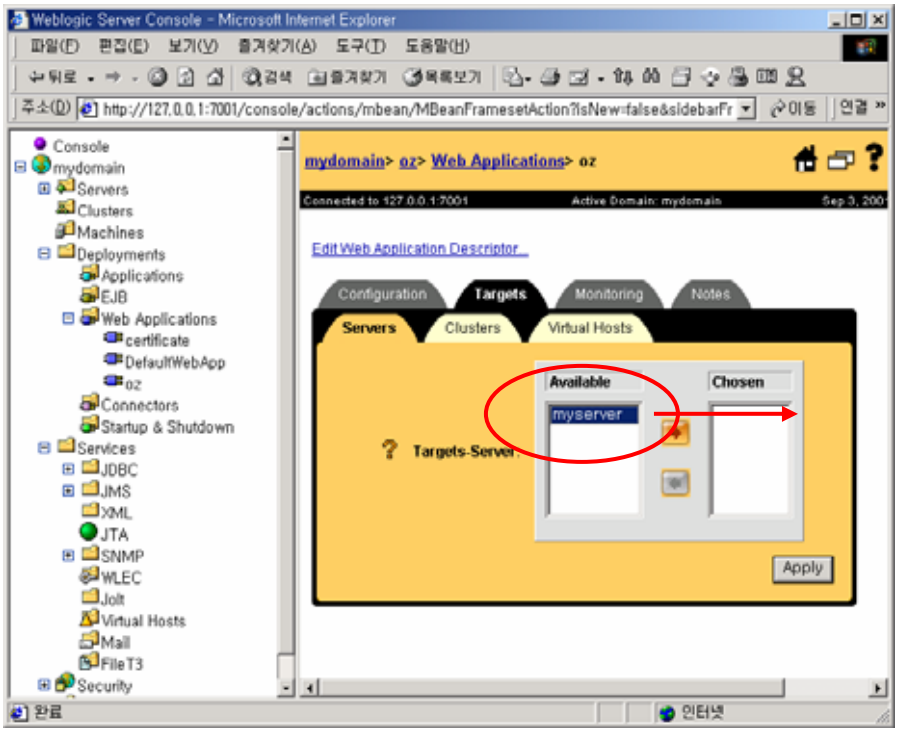
예를 들어 오즈 서블릿 메인 디렉토리를 'D:/servlet2'로, 이름은 'oz'로 등록하고자 한다면 [Configuration] – [General] 탭에서 각 란에 다음과 같이 입력합니다.

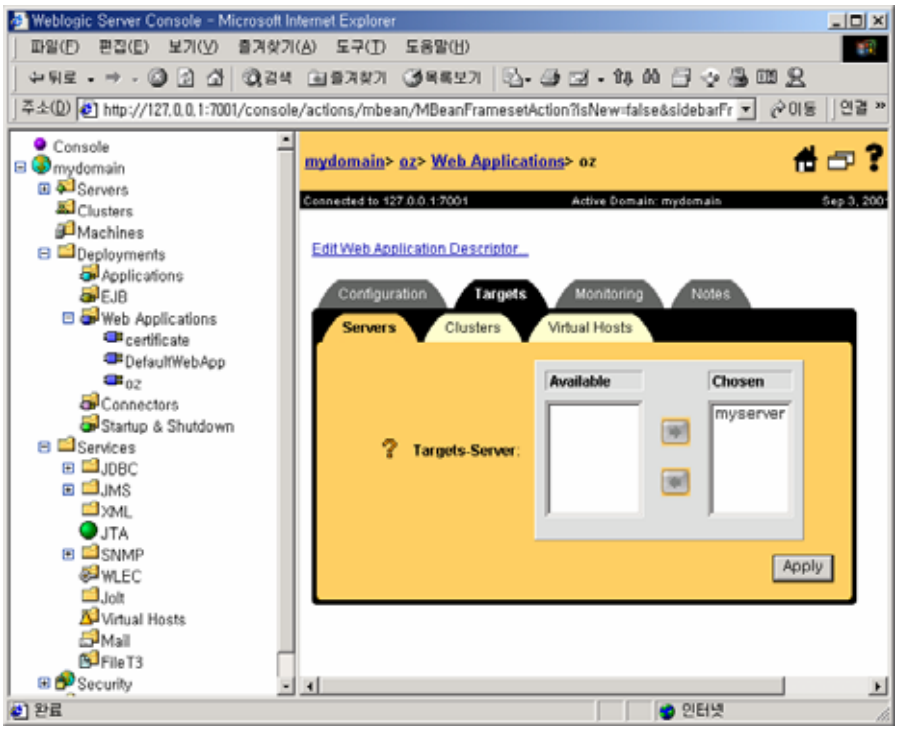


입력한 후 [Create]를 클릭하면 다음과 같이 등록됩니다.



[Targets] 탭으로 이동합니다. 오즈를 등록할 Target 서버를 선택합니다.





[Apply] 클릭하여 등록을 마칩니다.

오즈가 정상적으로 작동하는지 확인해 봅니다

http://127.0.0.1:7001/oz/server

(서버는 오즈 서블릿 홈 디렉토리의 WEB-INF/web.xml에서 설정해 준 서블릿 url pattern입니다.)

※ 참고

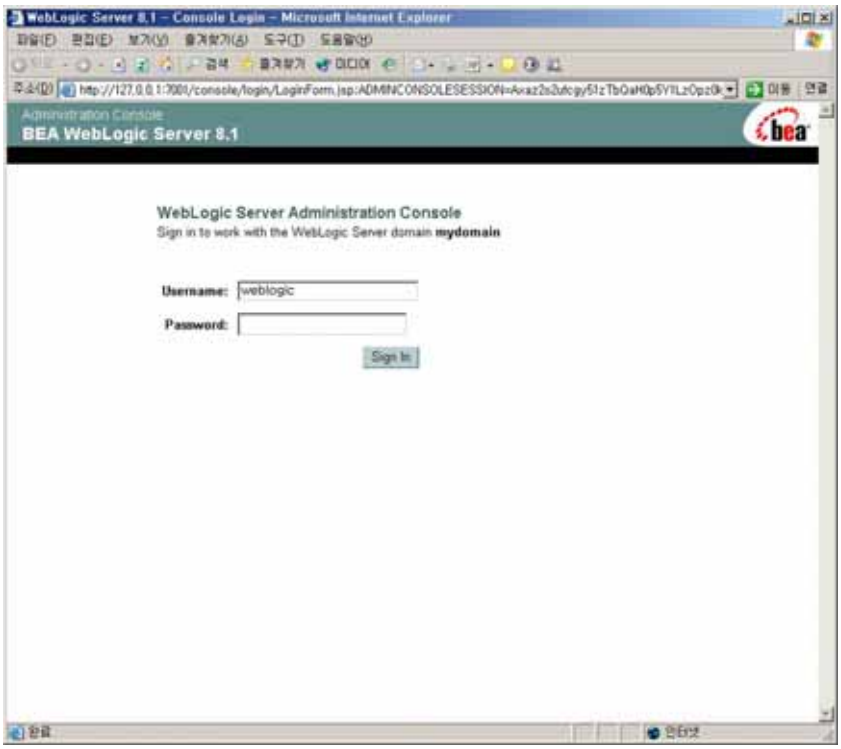
- {$WEBLOGIC6.1}₩config₩mydomain₩config.xml 파일을 열어 오즈가 등록된 정보가 포함되어 있는지 확인합니다.
- 서비스팩이 설치되어 있는 경우 앨리어스명을 반드시 5자 이상으로 지정하여야 합니다.

웹로직 8.x 이상

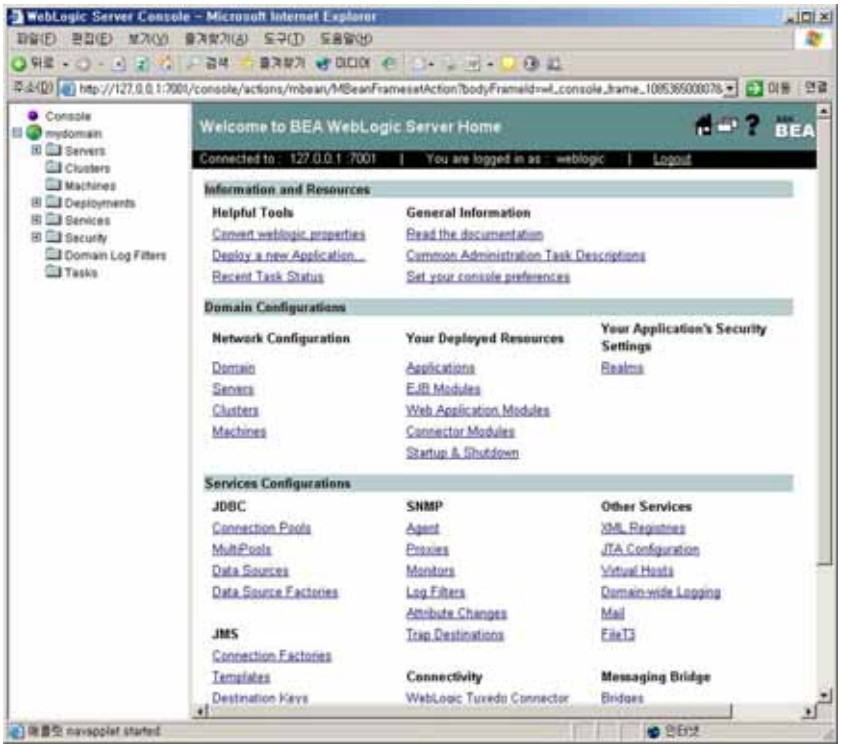
웹로직 8.x의 경우에도 웹로직에서 지원하는 관리자 콘솔을 이용하여 오즈 엔터프라이즈 서버를 등록합니다.

■ **Weblogic 8.x OZ Servlet 설치하기**

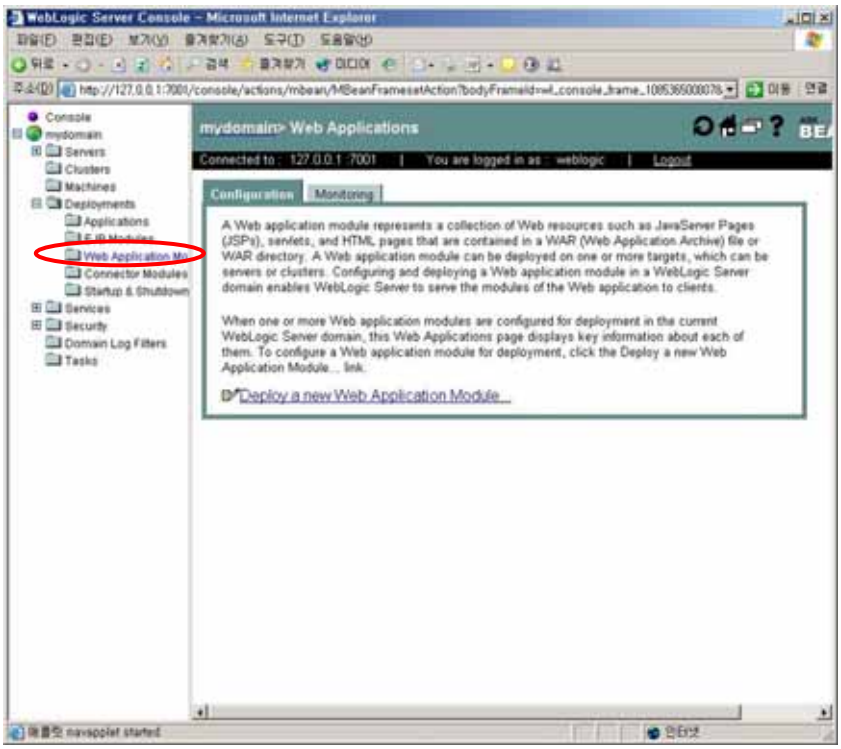
먼저 웹로직 관리자 콘솔에 관리자 계정을 이용하여 로그인 합니다.



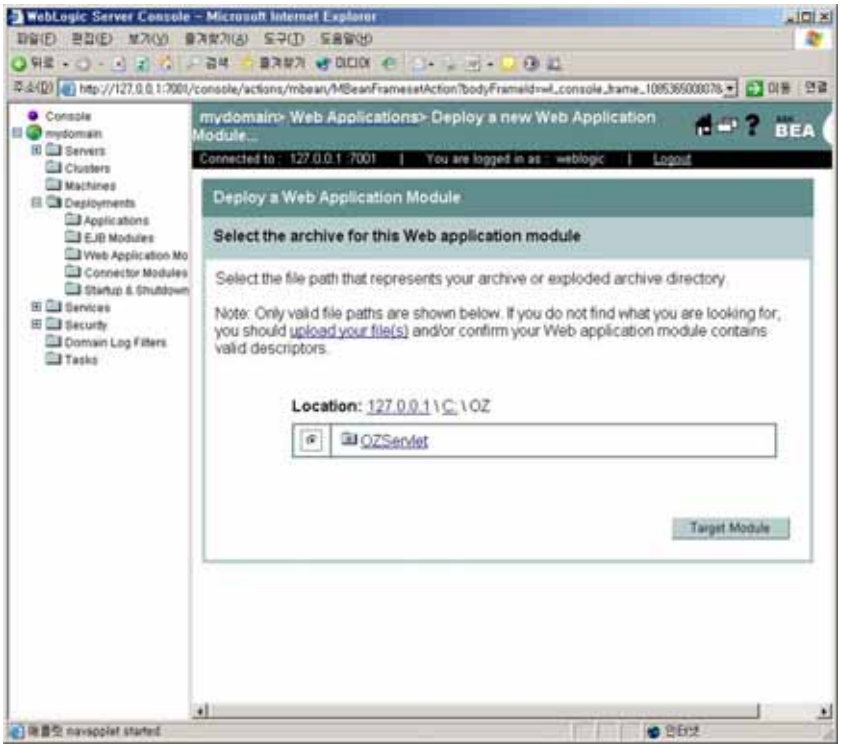
설치할 Domain을 선택합니다.



[Deployments] – [Web Application Modules]을 선택합니다.

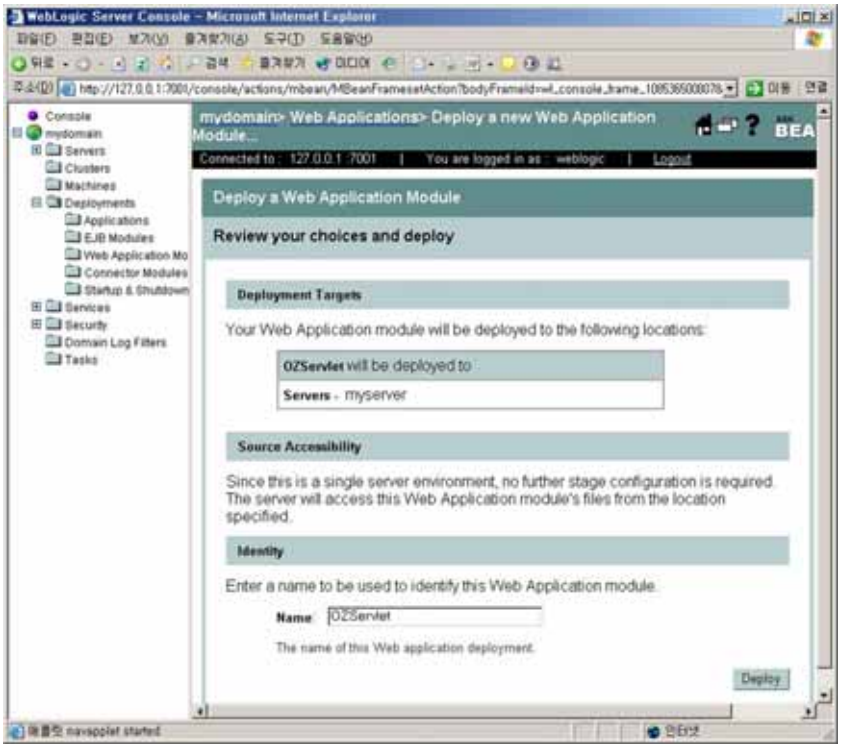


OZ Servlet Type 제품의 서버 폴더를 선택합니다.

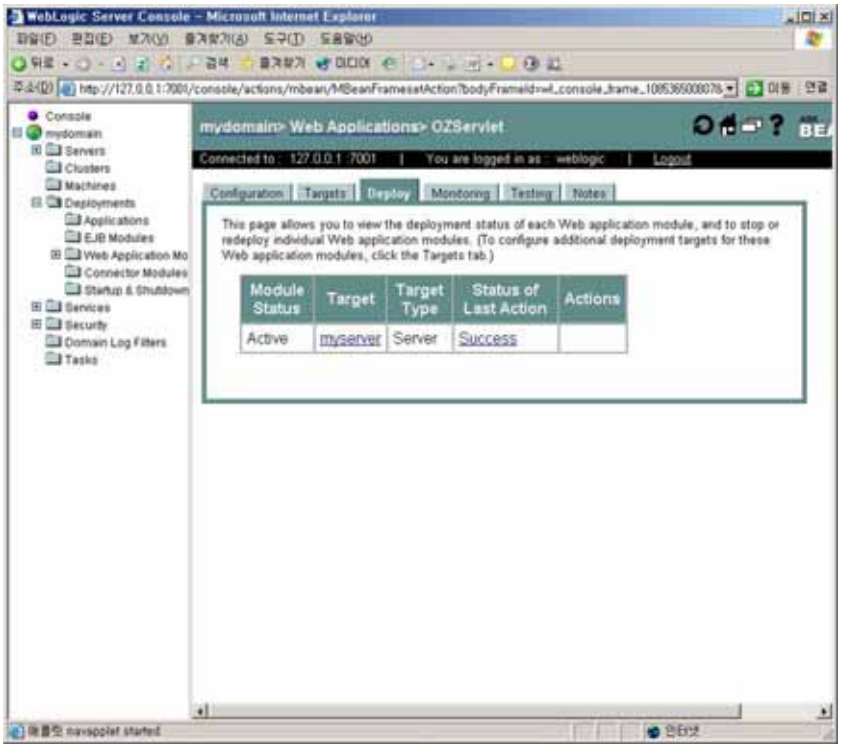


OZ Server URI 등록 후 [Deploy] 버튼을 클릭합니다.

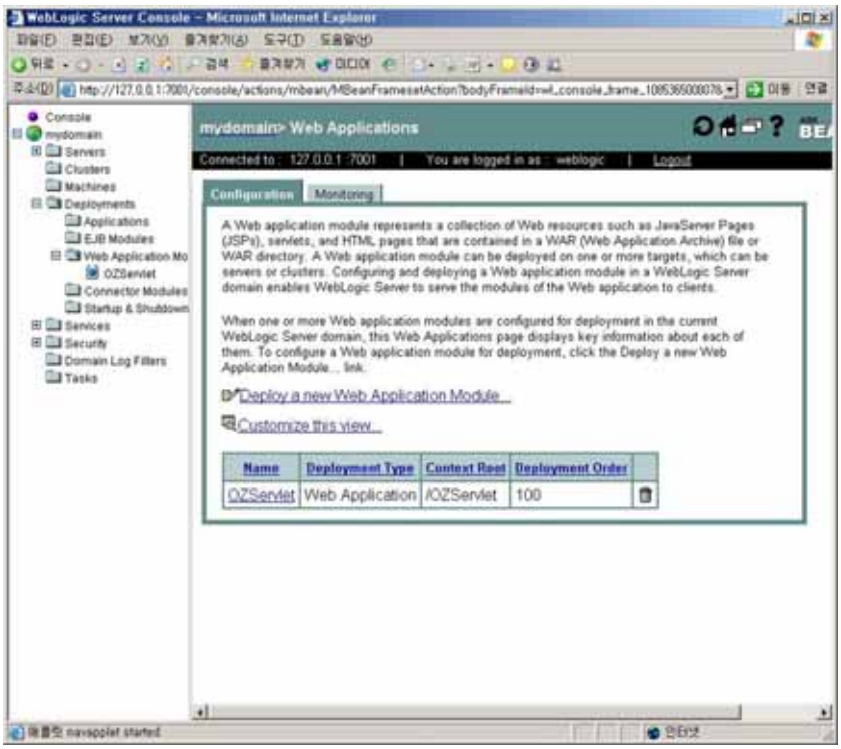
예) URI를 “OZServlet”로 입력시 → http://ip:port/OZServlet/server로 오즈서버 호출



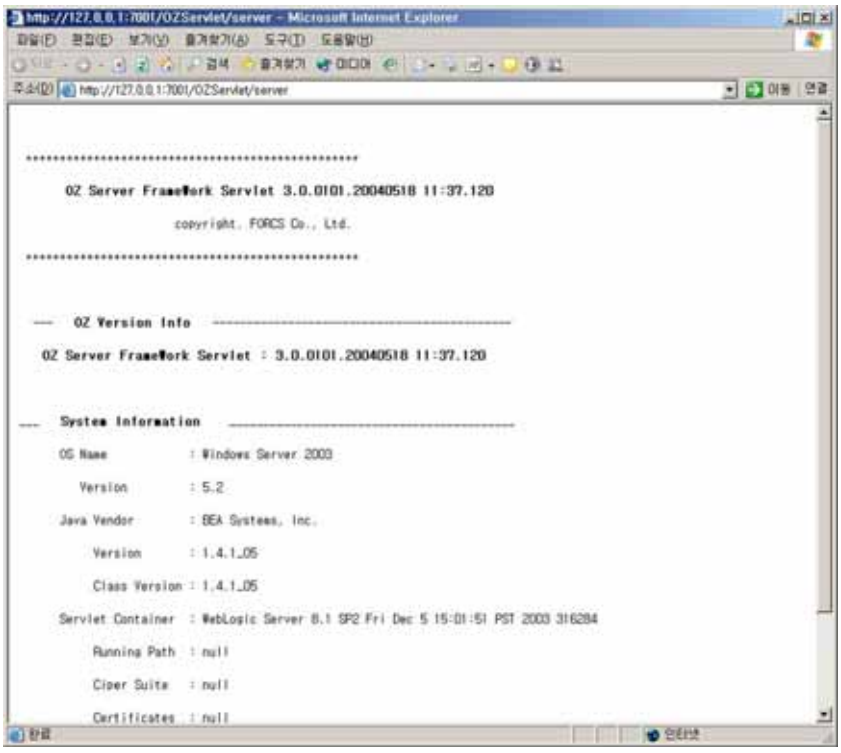
서버에서 정상적으로 Deploy 되었을 경우 아래와 같은 화면이 나타납니다.

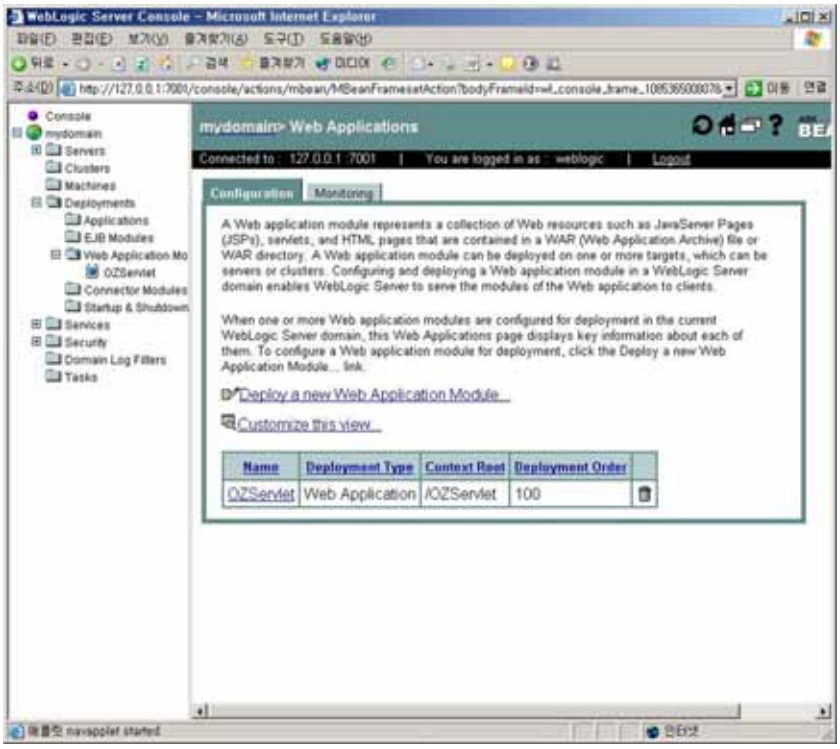


왼쪽 트리의 [Deployments] – [Web Application Modules]에서 OZServlet이 등록된 것을 확인할 수 있습니다.



웹 브라우저를 이용하여 http://ip:port/OZServlet/server에 접속하면 오즈 서버 정보 화면이 정상적으로 표시되는지 확인할 수 있습니다.





WebSphere와의 연동

WebSphere 3.5

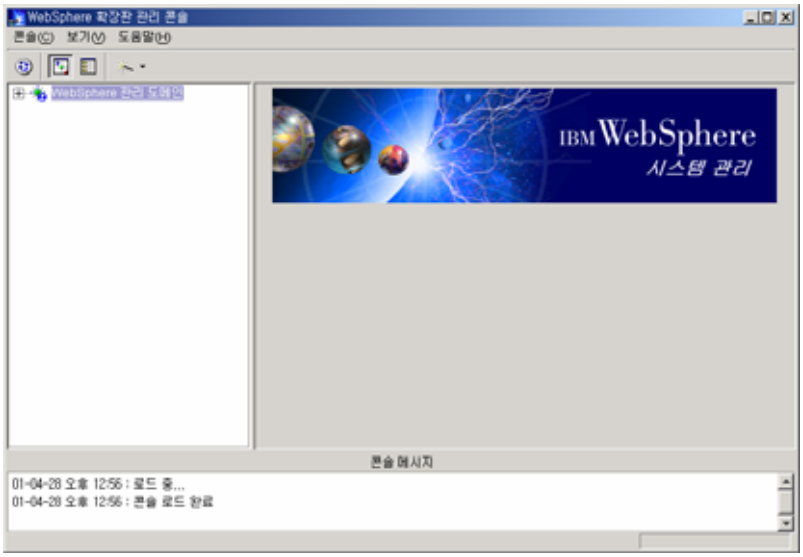
WebSphere와 오즈를 연동하기 위해서는 WebSphere의 Admin 서비스가 반드시 시작되어 있어야 합니다.

■ **디렉토리 구조**

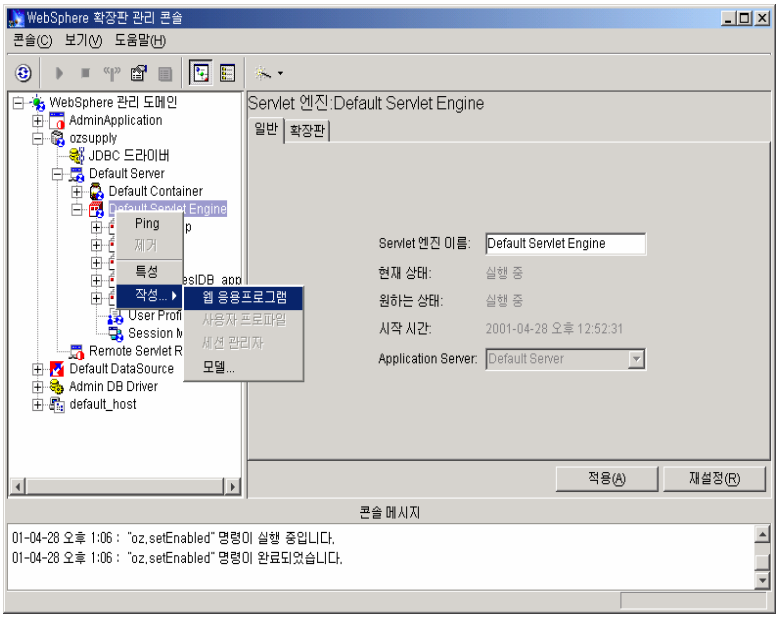
WebSphere 디렉토리는 ($WebSphere_Home)이라고 가정합니다. 오즈 서블릿 디렉토리 구조는 기본 디렉토리 구조와 같습니다.

■ **오즈 서블릿 설치**

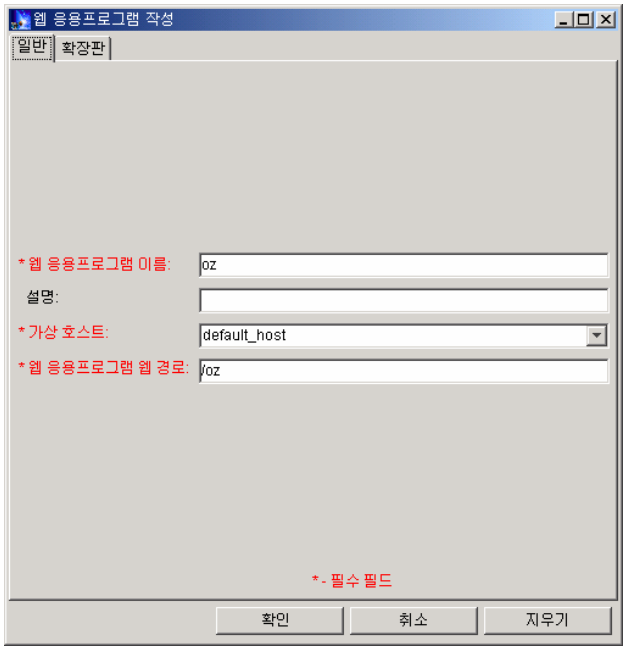
[시작] – [프로그램] – [WebSphere]에서 [관리 콘솔]을 실행합니다.



왼쪽 패널의 아이콘을 클릭하면 서버 이름(호스트명)의 서비스가 있습니다. 하위 메뉴중에서 [Default Server] – [Default Servlet Engine]을 선택합니다. 마우스 오른쪽 버튼을 클릭하여 [작성] – [웹 응용프로그램]을 선택합니다.

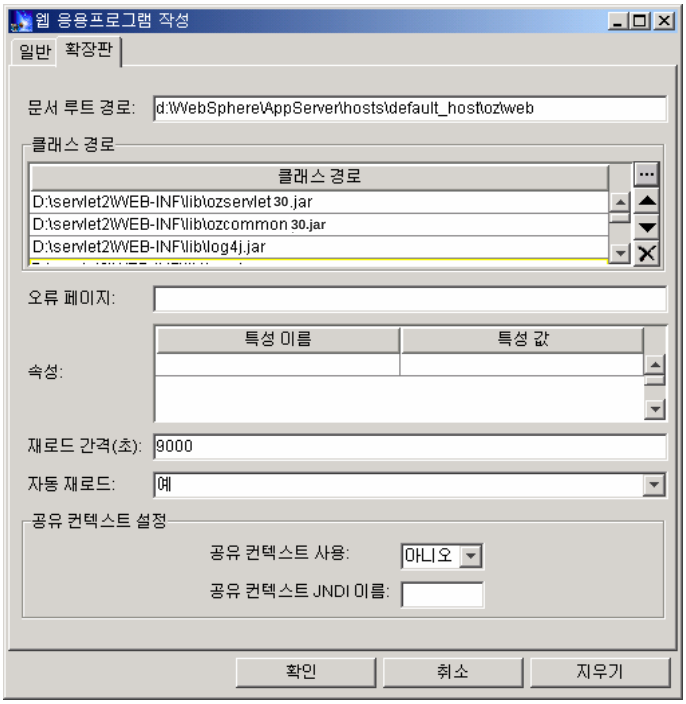


[일반] 탭에서 등록할 [웹 응용프로그램 이름]란에 'oz'를 입력하고, [설명]란에는 간략한 설명을 적어 줍니다. [가상호스트]란에 기본값을 'default_host'로 설정한 다음 [웹 응용프로그램 웹 경로]에 '/oz'를 입력합니다. 이 값이 오즈 서블릿의 URL Path가 됩니다.

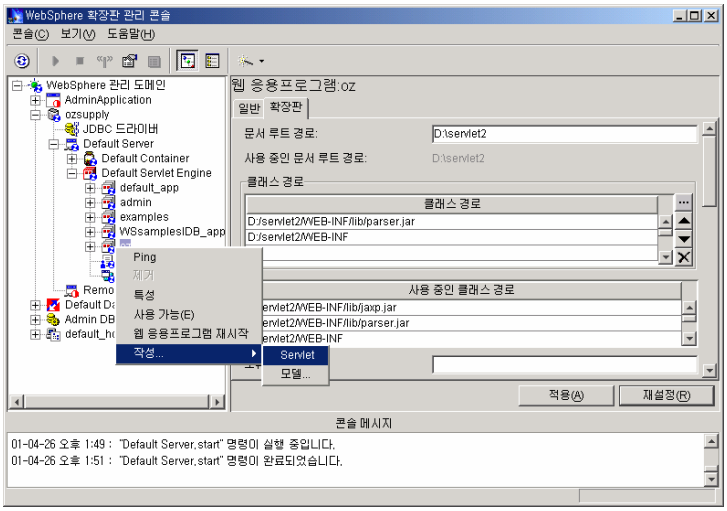


다음은 [확장판] 탭을 클릭합니다.

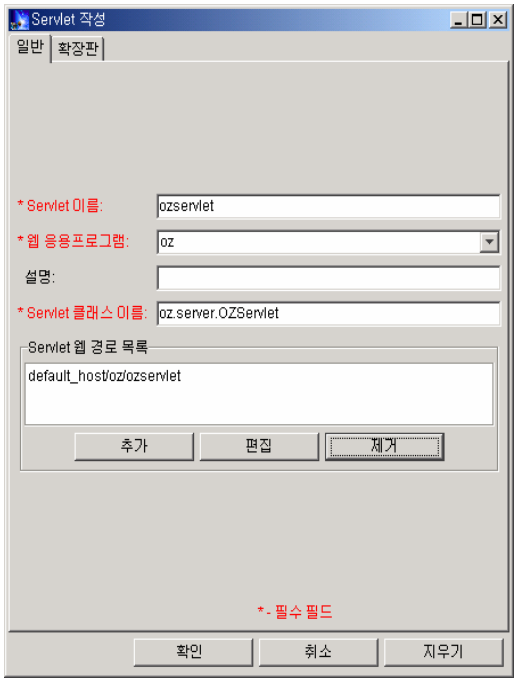
[클래스 경로]에 오즈를 위한 각각의 클래스 경로를 적어줍니다. 기본적으로 ozlib30.jar, log4j.jar, parser.jar를 경로로 잡아주고 JDBC 드라이버를 위한 디렉토리와 각각의 JDBC 파일 경로를 잡아줍니다.



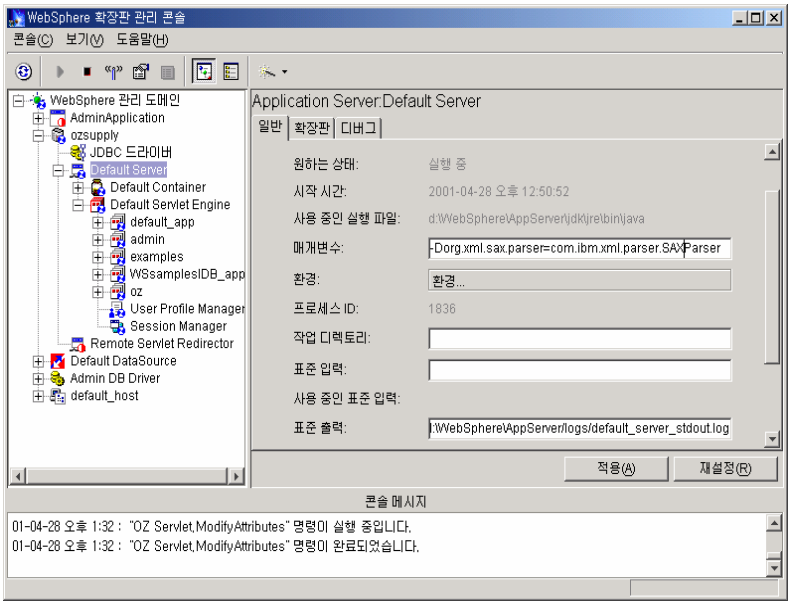
메인 패널에 웹 응용프로그램 'oz'가 생성 되었습니다. 마우스 오른쪽 버튼을 클릭한 후 [작성] – [Servlet]을 선택합니다. 이곳에서 웹 응용프로그램으로 등록된 'oz'의 환경을 등록합니다.



[Servlet 이름]란에 'ozservlet', [웹 응용프로그램]란은 기본값인 'oz' 선택합니다. [Servlet 클래스 이름]란에는 'oz.server.OZServlet'이라고 입력합니다. [Servlet 웹 경로 목록]에 [추가] 버튼을 클릭한 다음 'server'라고 입력합니다. 이것이 오즈의 Servlet 웹 경로가 됩니다.



오즈와 WebSphere를 위해서는 오즈의 xml 파서가 Sun의 파서(Parser.jar)가 아닌 IBM의 파서(Crimson.jar)임을 알려 주어야 합니다.
콘솔의 왼쪽 창에서 [Default Server]를 클릭하여 나오는 다음과 같은 화면에서 파서를 설정해 줍니다. 또한 오즈의 xml.properties 파일에서도 IBM의 해당 파서에 맞게 변경해 주어야 합니다. 자세한 내용은 “오즈 엔터프라이즈 서버 관리자 매뉴얼(Ⅰ)의 Ⅳ.오즈 엔터프라이즈 서버 설정 - xml.properties”를 참조하시기 바랍니다.



여기서 [매개변수]란에 ‘-Dorg.xml.sax.parser=com.ibm.xml.parsers.SAXParser’라고 입력합니다.

모든 설정이 완료되면 확인 작업을 수행합니다. 데이터베이스 연결 설정을 확인합니다. 오즈 서블릿의 자체적인 DB Pooling을 사용할 때에는 오즈 서버 데몬 버전에서와 같은 방법으로 db.properties 파일을 설정합니다.
웹 브라우저에서 http://localhost/oz/server라고 입력하여 서버의 기본 정보 페이지를 호출하여 정상적으로 표시되는지 확인합니다.

WebSphere 4.0

1. **오즈 서블릿 디렉토리에서 servlet30.war 파일을 만듭니다.**

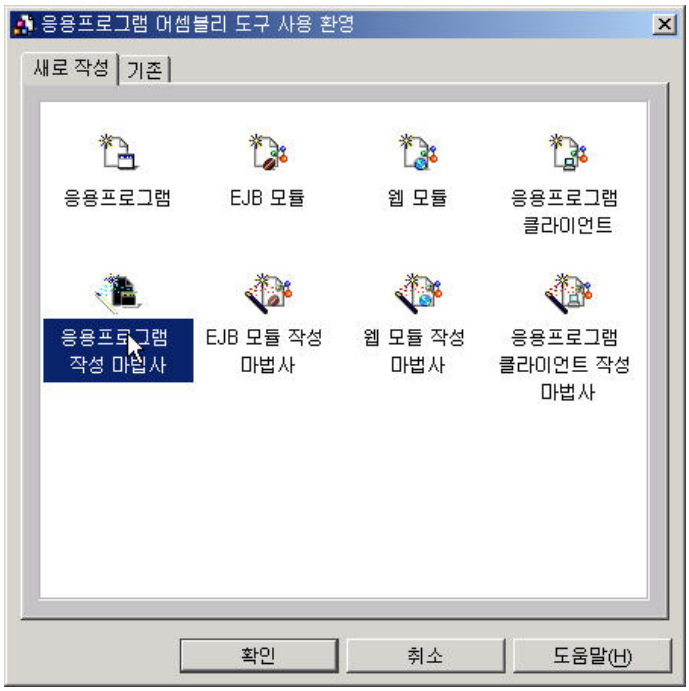
   jar cvf servlet30.war * <windows용>

2. **웹스피어의 응용프로그램 어셈블리 도구 사용 환경을 실행시켜 servlet30.ear 파일을 만듭니다.**

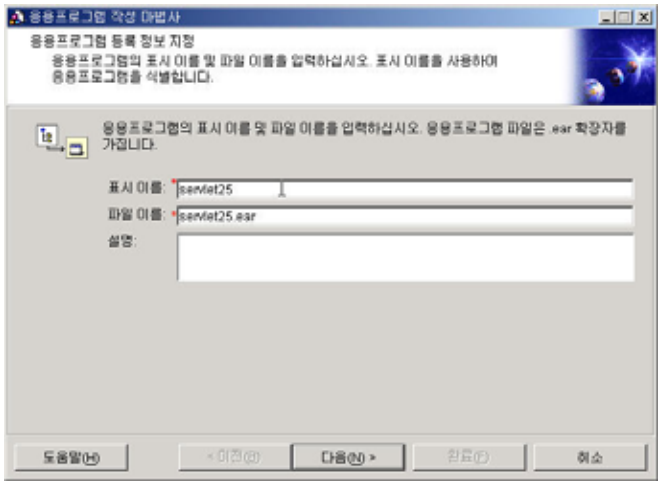
   jar cvf servlet30.war * <windows용>

■ **Servlet30.ear 파일 등록 과정**

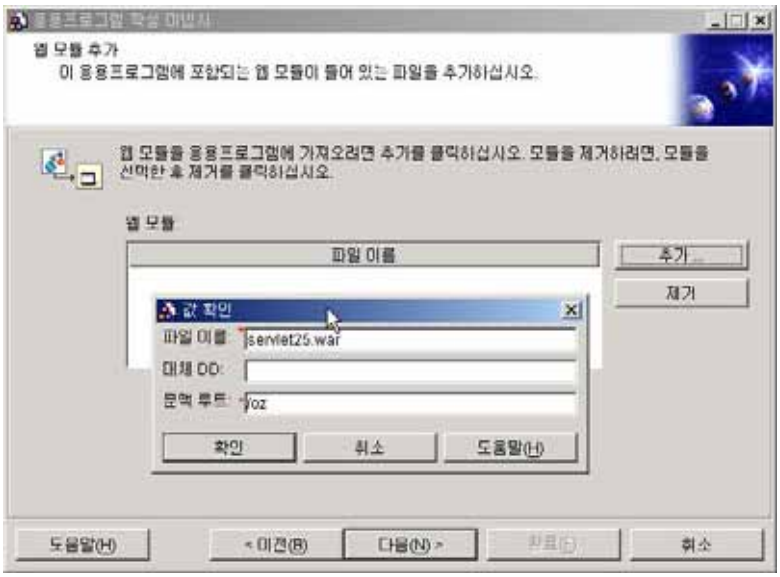
[응용프로그램 어셈블리 도구 사용 환경]에서 [응용프로그램 작성 마법사]를 선택합니다.

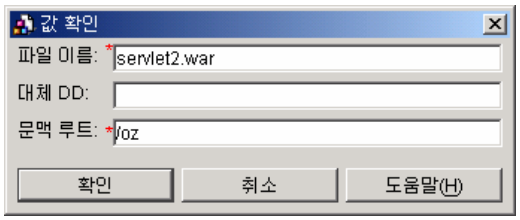


[응용프로그램 작성 마법사]에 servlet30.ear 파일에 대한 정보를 입력합니다.



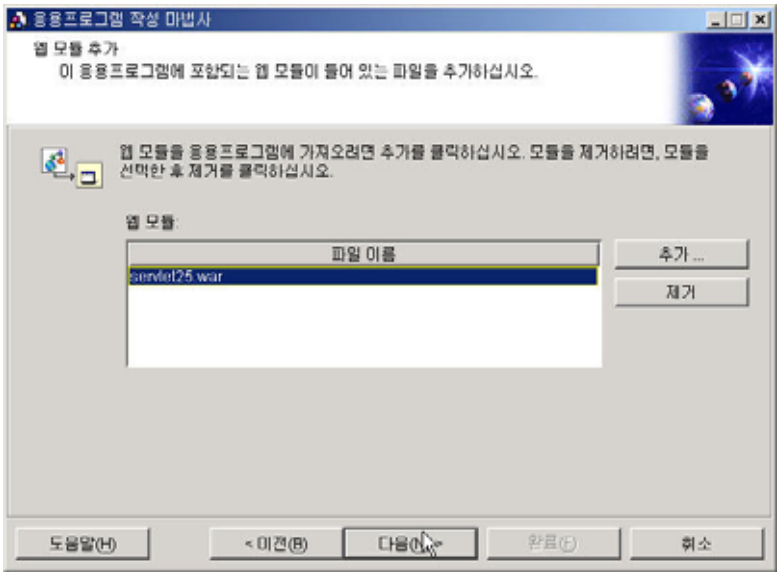
[다음]을 누르다가 웹 모듈 추가 창이 나타나면 servlet30.war를 등록합니다.



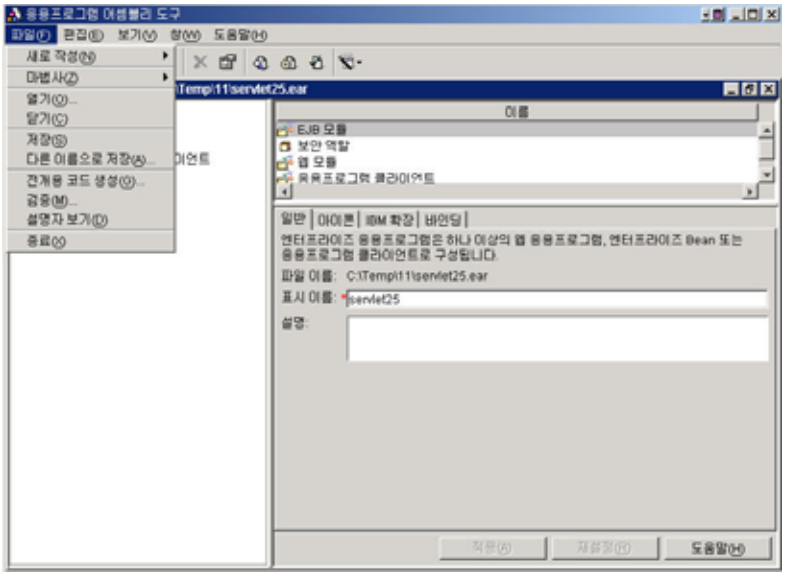


문맥 루트의 오즈는 웹상에서 호출하는 오즈 서블릿의 URL Path의 Alias가 됩니다.

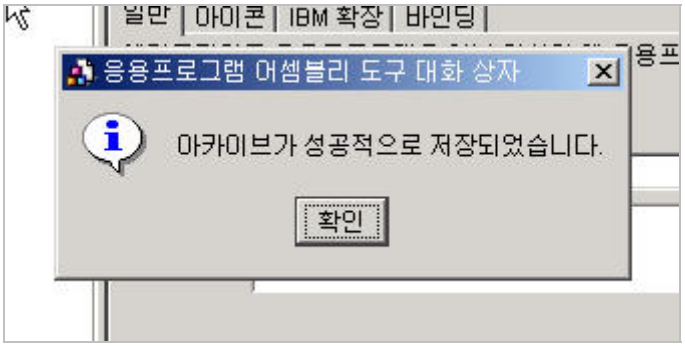
[다음]을 클릭하면서 각 설정들을 등록한 후 완료합니다.



[다른 이름을 저장]을 선택하여 servlet30.ear 파일로 저장합니다.



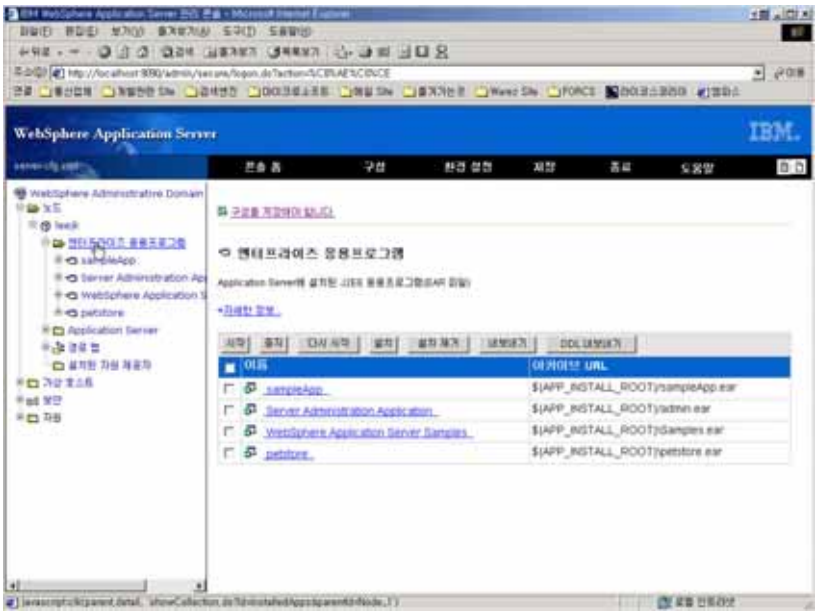
다음과 같은 화면이 나오면 [확인] 버튼을 클릭합니다. 저장이 완료되었습니다.



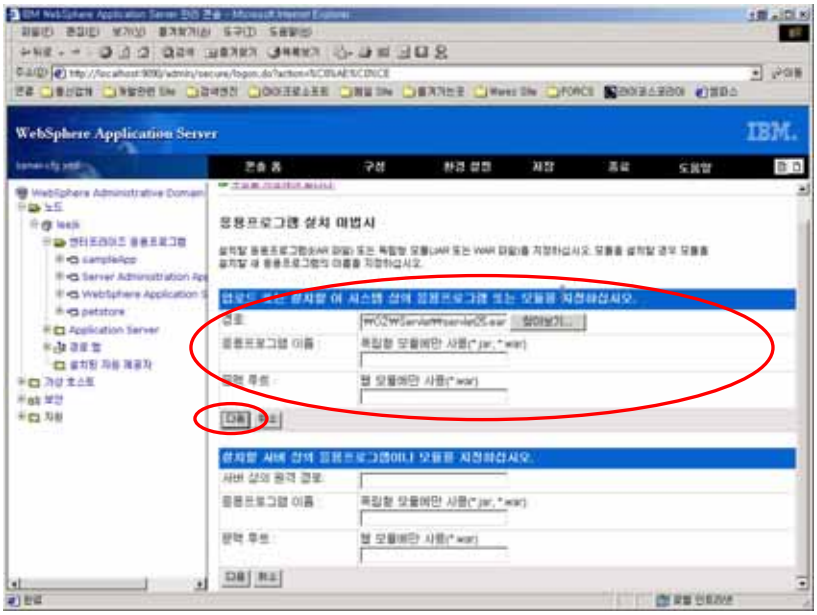
3. WebSphere 관리콘솔을 실행하여 오즈를 등록합니다.(Single과 Server로 구분됩니다.)

■ **Single에서 servlet30.ear 설치 과정**

[엔터프라이즈 응용프로그램]을 선택한 후 servlet30.ear을 설치합니다.

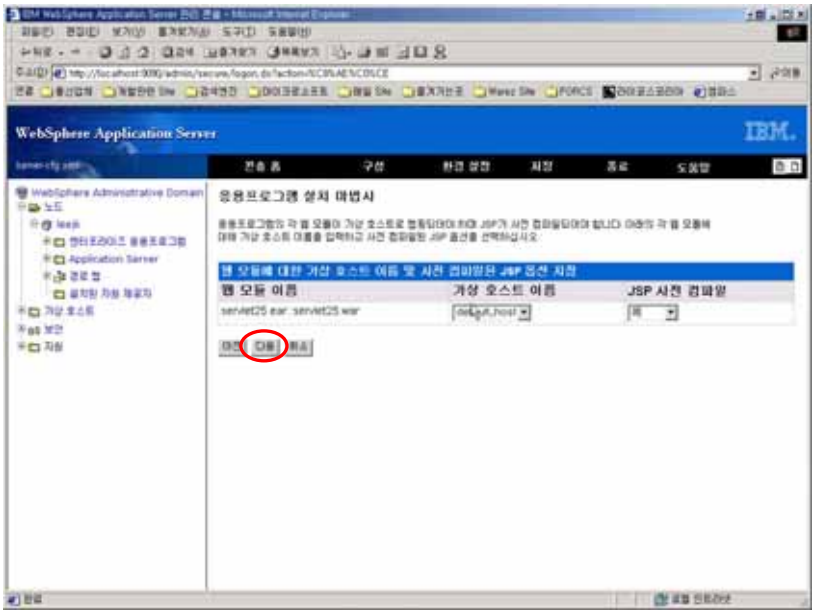


[설치]를 클릭하여 servlet30.ear 파일을 찾은 후 [다음]을 클릭합니다.

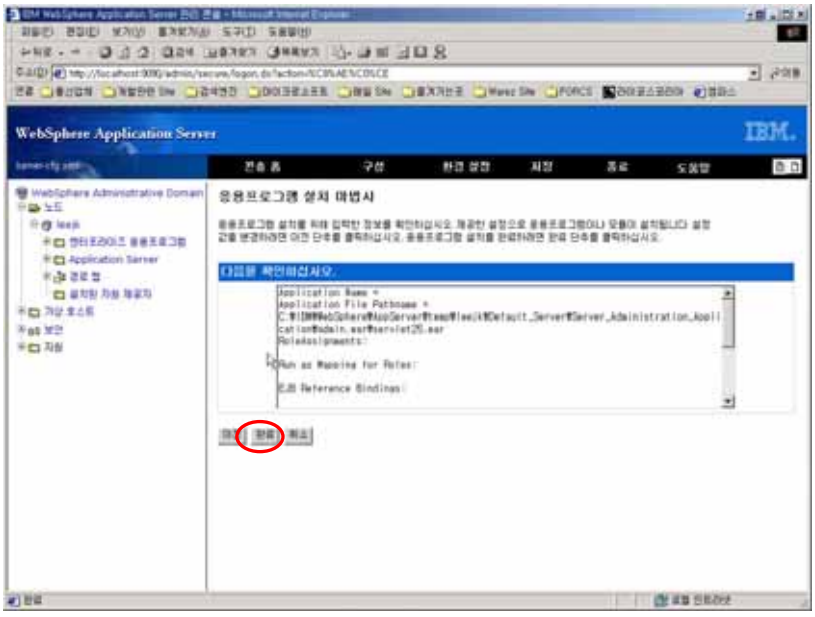


※ 로컬 PC에 파일(servlet30.ear)이 있을 때는 위쪽의 [업로드 또는 설치할 이 시스템 상의 응용프로그램 또는 모듈을 지정하십시오]에서 찾아주고, 서버에 파일이 있고 원격으로 설치할때는 아래쪽 [설치할 서버 상의 응용프로그램이나 모듈을 지정하십시오]에서 경로를 등록해 줍니다.

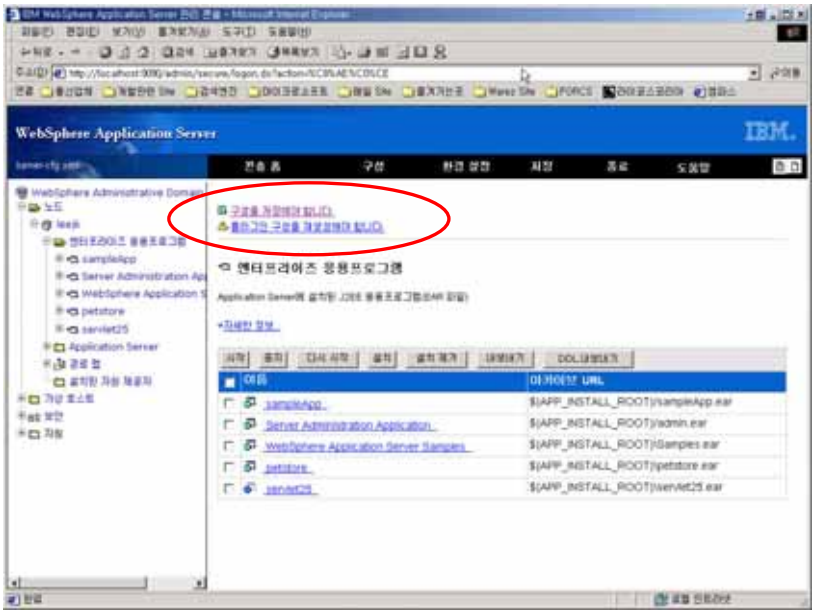
[가상 호스트 이름]을 설정합니다. [웹 모듈 이름]을 확인하고 [다음]을 클릭합니다.



응용 프로그램의 관련된 정보를 확인하고 [완료]를 누릅니다. [완료]를 누른 후 약간의 시간이 흐른 다음에 등록(Deploy)이 완료됩니다.

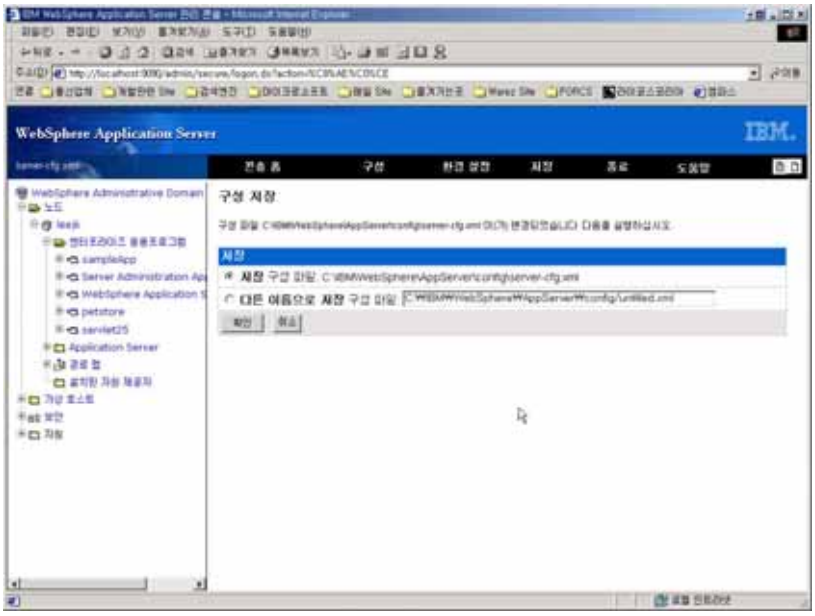


다음과 같이 'servlet30' 응용프로그램이 설치됩니다.

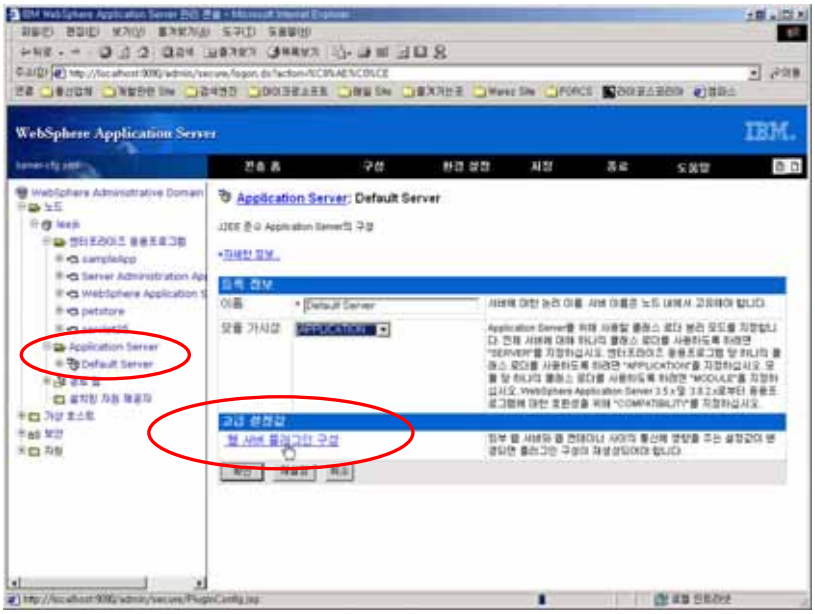


※ 처음 설치 후에는 아카이브가 정상으로 나오지 않습니다. 이 문제를 해결하기 위해서는 '구성을 저장해야 합니다' 및 '플러그인 구성을 재생성해야 합니다'를 반드시 클릭해야 합니다.

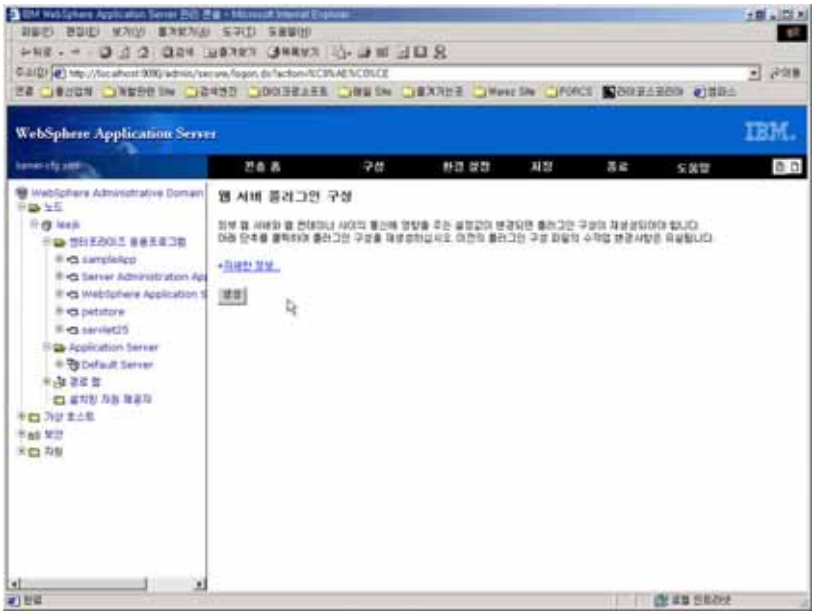
[저장]을 클릭하여 변경된 환경 파일을 저장합니다.



다음에는 [Application Server] – [Default Server]의 [고급 설정값]에서 '웹 서버 플러그인 구성' 항목을 클릭합니다.

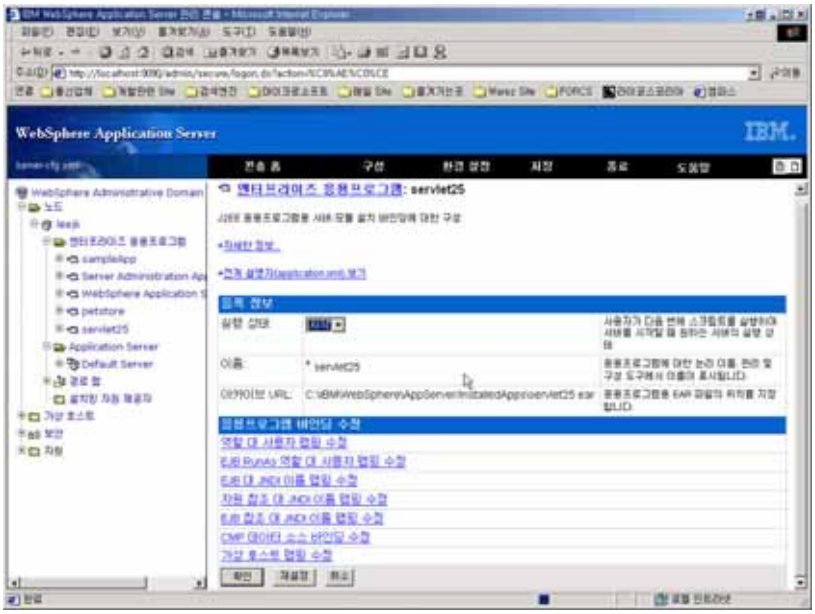


다음과 같은 화면이 표시되면 [생성]을 클릭하여 완료시킵니다. 구성을 다시 저장하라는 메시지가 나오면 다시 저장합니다.

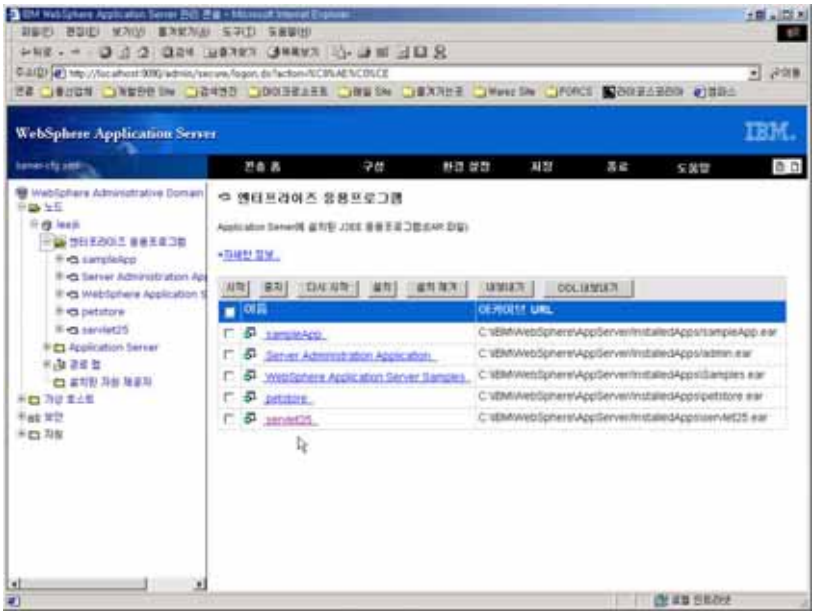


WebSphere를 다시 시작 실행시키면 오즈 서블릿도 함께 실행됩니다.

[엔터프라이즈 응용프로그램]에서 'servlet30'를 선택하고 [확인] 버튼을 클릭하면 웹스피어에 등록된 OZ에 대한 구성 상태를 확인할 수 있습니다.



다음과 같이 오즈 서블릿을 시작 혹은 중지시킬 수 있습니다.

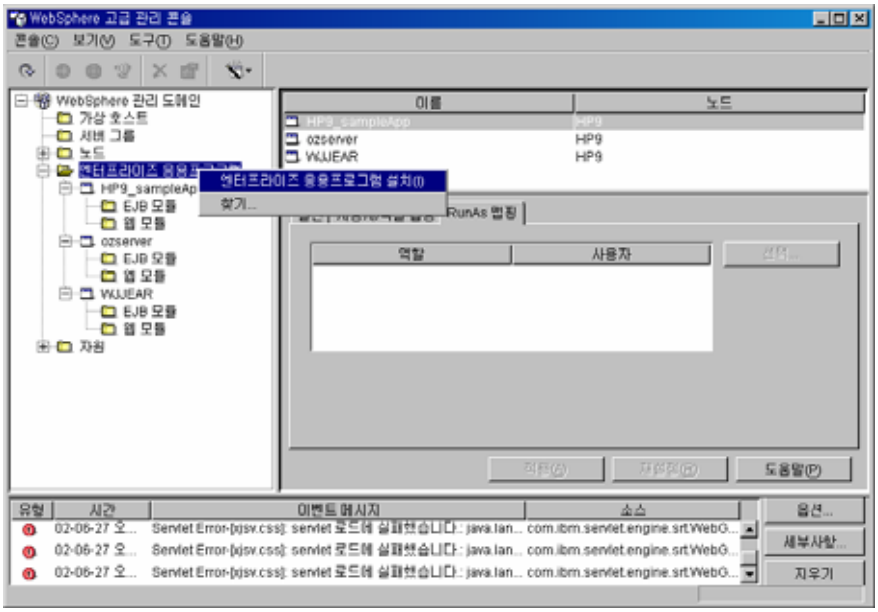


※ 참고

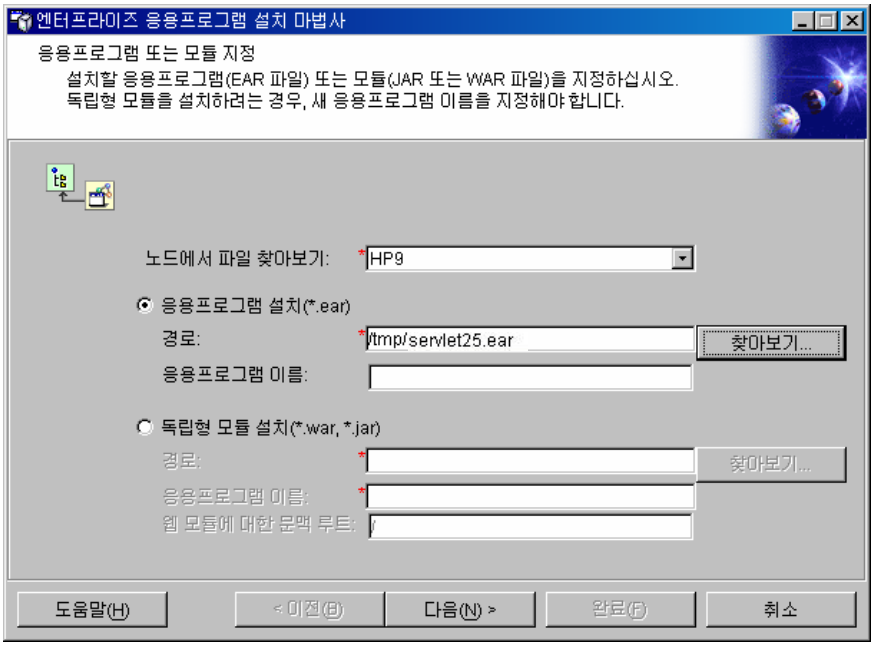
WebSphere 3.5에서는 응용 프로그램별로 재실행이 가능하였으나 4.0에서는 WebSphere를 재실행해야만 설정된 응용 프로그램이 재실행됩니다.

■ **Server에서 servlet30.ear 설치 과정**

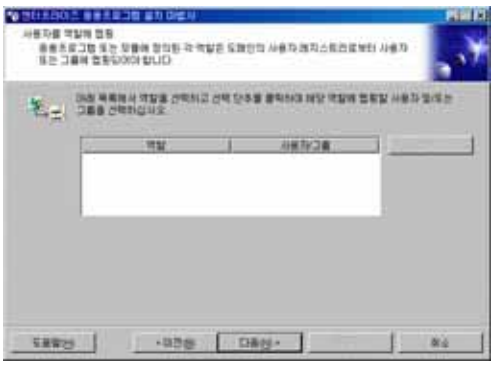
[WebSphere 고급 관리 콘솔]에서 [엔터프라이즈 응용프로그램] – [엔터프라이즈 응용프로그램 설치]를 선택합니다.



[엔터프라이즈 응용프로그램 설치 마법사]에서 servlet30.ear을 설치합니다.

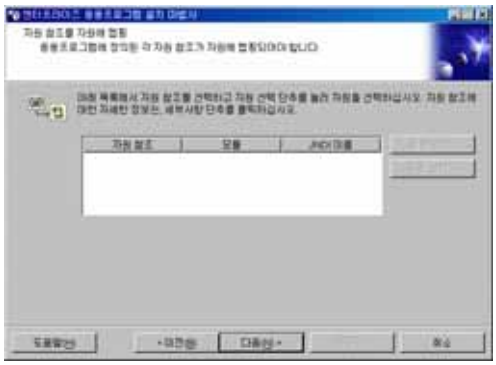


[다음]을 클릭하면 다음과 같은 화면들이 나타납니다.



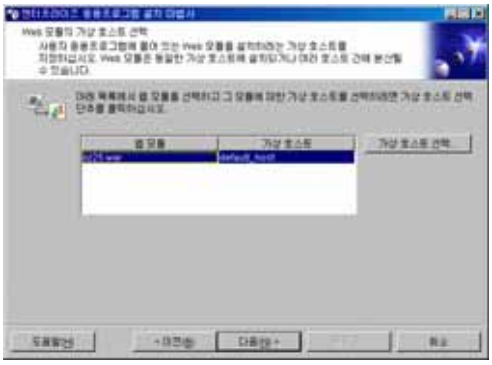


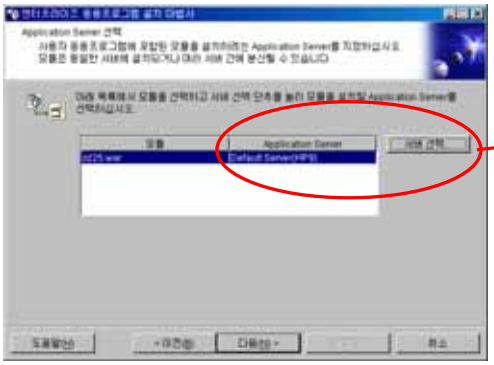


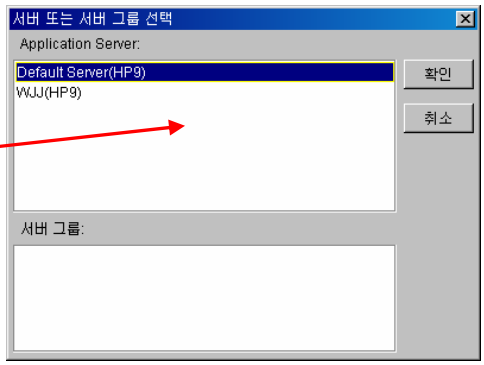




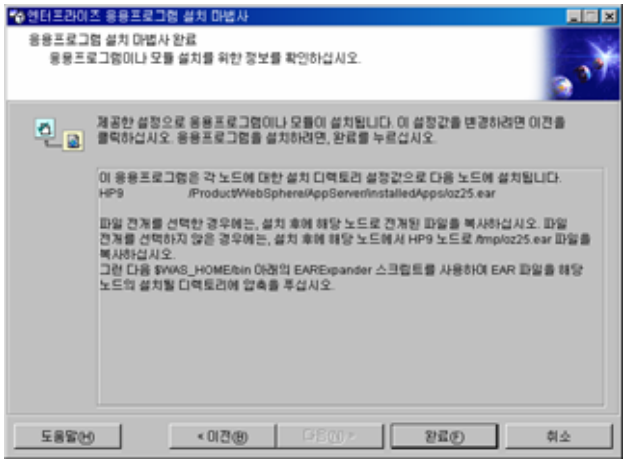




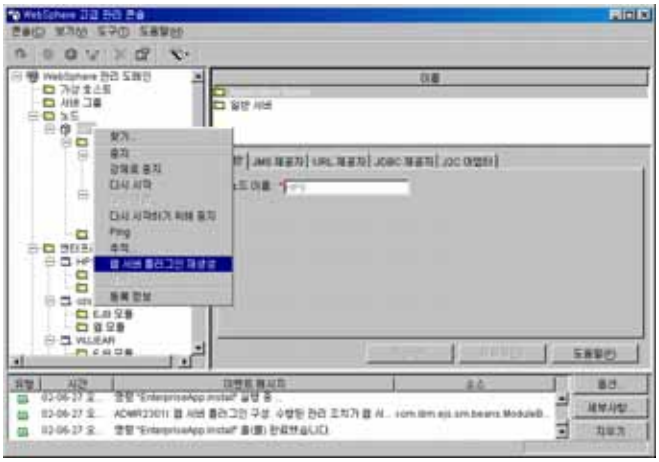




설치할 서버(노드) 그룹을 선택한 후 [확인]을 클릭하면 다음과 같이 [응용 프로그램 설치 마법사 완료] 메시지가 나타납니다. [완료]를 클릭하면 설치 완료됩니다.



설치 완료 후 노드(servlet30.ear를 설치할 서버)를 선택한 후 '웹 플러그인 재생성'을 해줍니다. 플러그인 재생성을 한 후 서버를 재실행하면 오즈도 같이 연동 실행됩니다.



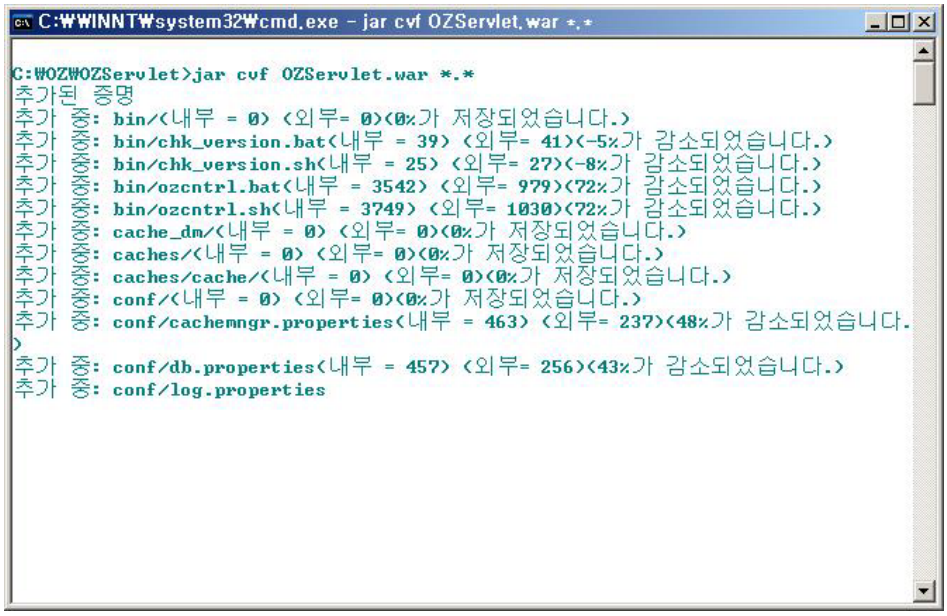
WebSphere 5.0

오즈 서버(servlet버전) 디렉토리 전체를 서버의 설치 디렉토리에 복사합니다.

jar cvf OZServlet.war * <windows용>

WEB-INF/classes/ozserver.properties와 conf/db.properties의 정보를 수정하고 오즈 서버에서 사용할 jdbc driver 파일을 WEB-INF/lib/에 복사합니다.

복사한 디렉토리에서 jar cvf OZServlet.war *.*를 실행하여 OZServlet.war를 만듭니다.



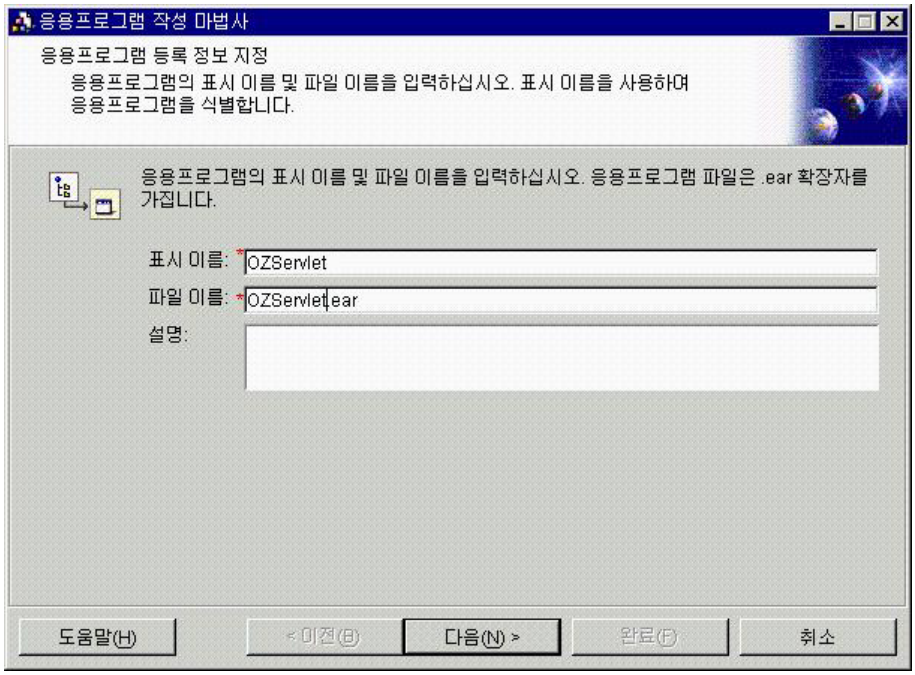
[WebSphere] – [AppServer] – [bin]에서 assembly.sh를 실행합니다.

■ **OZServlet.war 파일 등록 과정**

[응용프로그램 어셈블리 도구 사용 환경]에서 [응용프로그램 작성 마법사]를 선택한 후 [확인] 버튼을 클릭합니다.

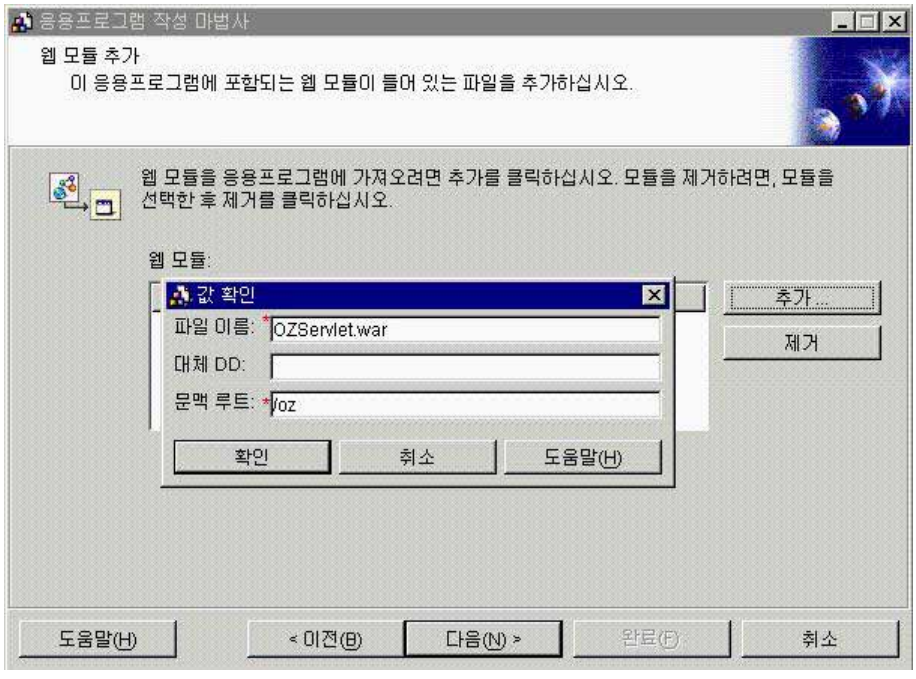


다음과 같은 화면이 나타나면 파일 이름을 OZServlet.ear로 변경합니다.

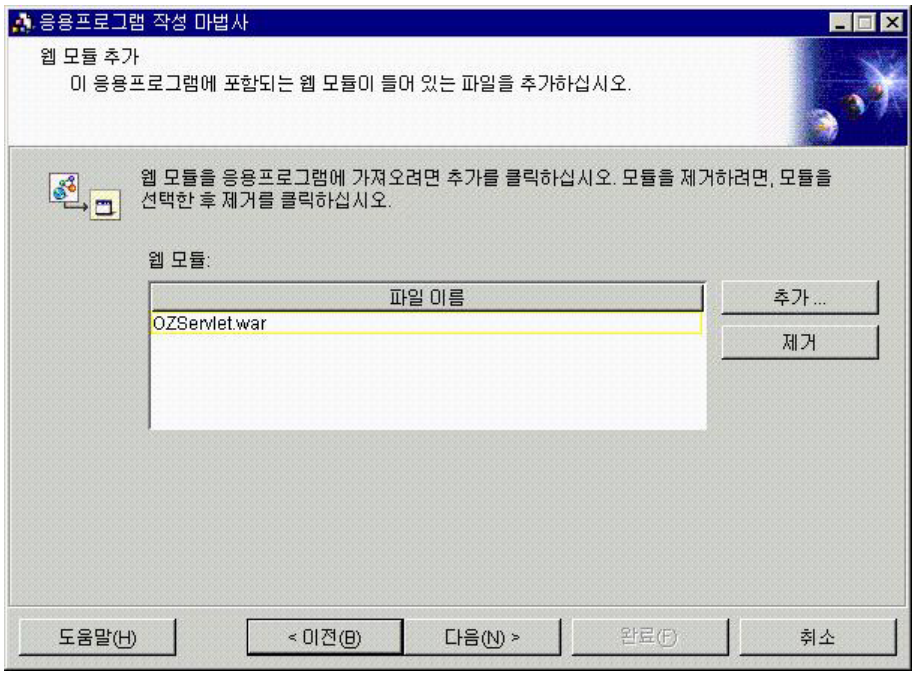


웹 모듈을 추가합니다. [추가] 버튼을 클릭하여 [값 확인]창이 나타나면 다음과 같이 입력합니다.

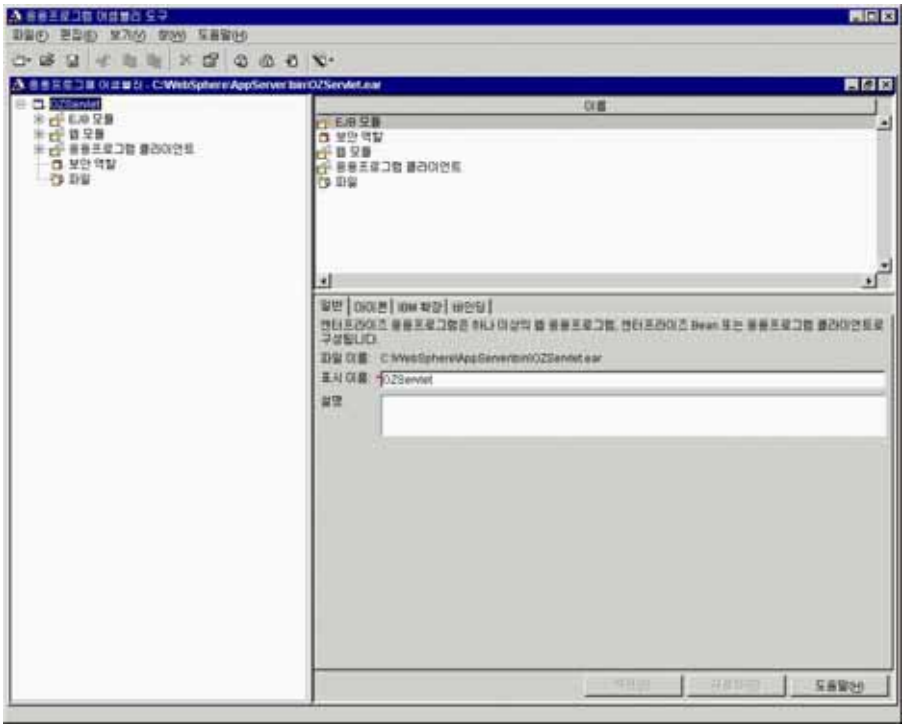
- 파일 이름 : OZServlet.war
- 문맥 루트 : /oz



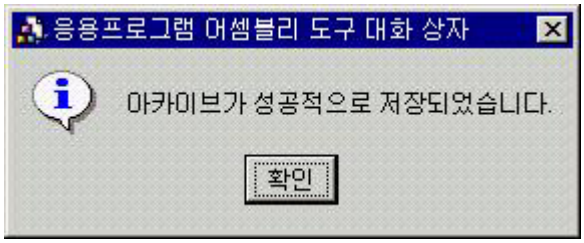
[확인] 버튼을 클릭하면 다음과 같이 웹 모듈이 추가됩니다. [다음] 버튼을 클릭합니다.



[파일] 메뉴에서 [다른 이름으로 저장]을 선택하여 OZServlet.ear 파일로 저장합니다.

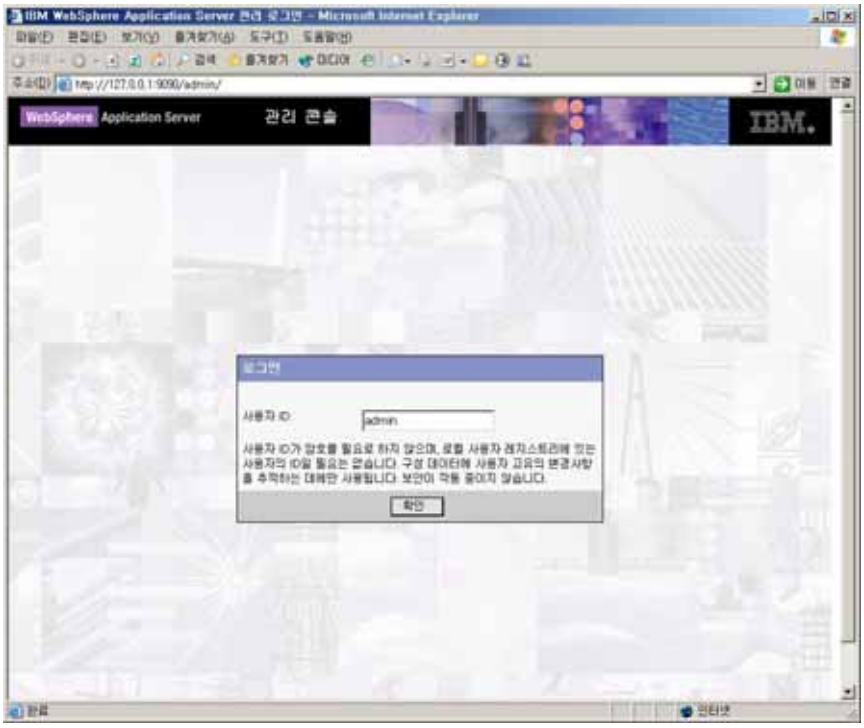


다음과 같은 화면이 나오면 [확인] 버튼을 클릭합니다. 저징이 완료되었습니다.

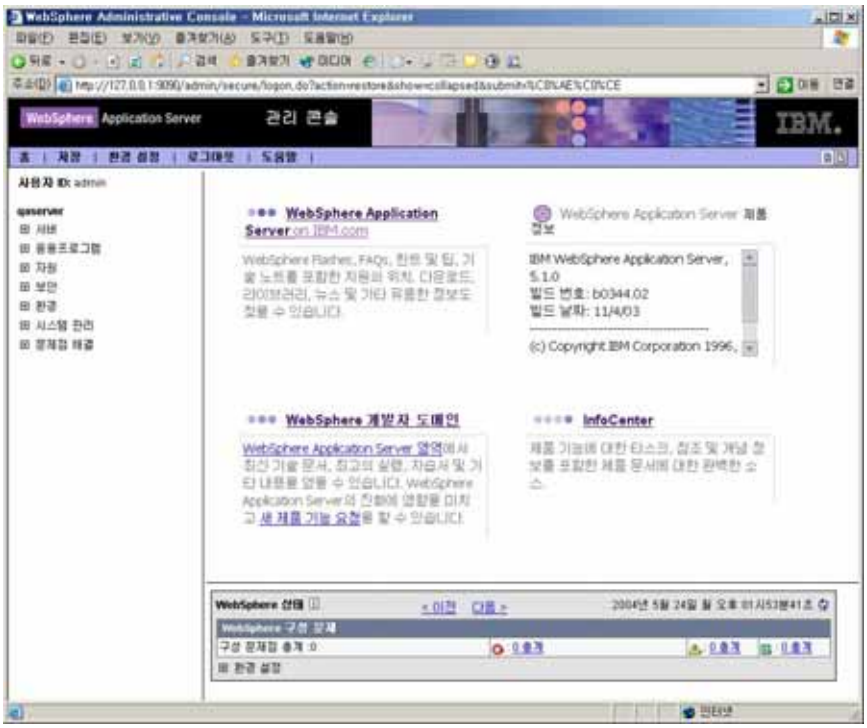


■ **WebSphere 관리 콘솔을 실행하여 오즈 등록하기**

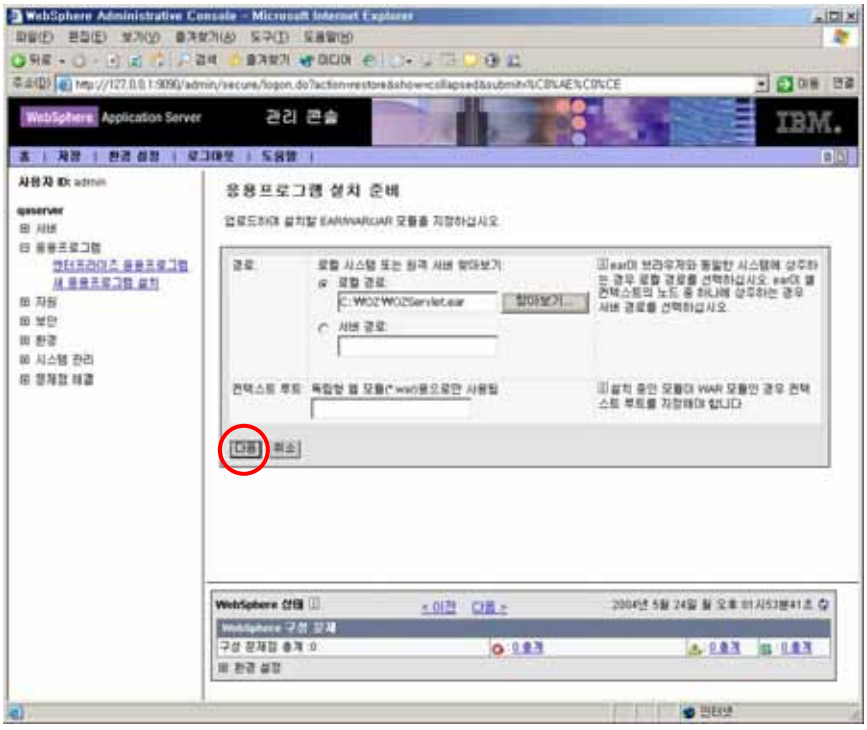
[로그인] 창에 사용자 ID를 입력합니다.



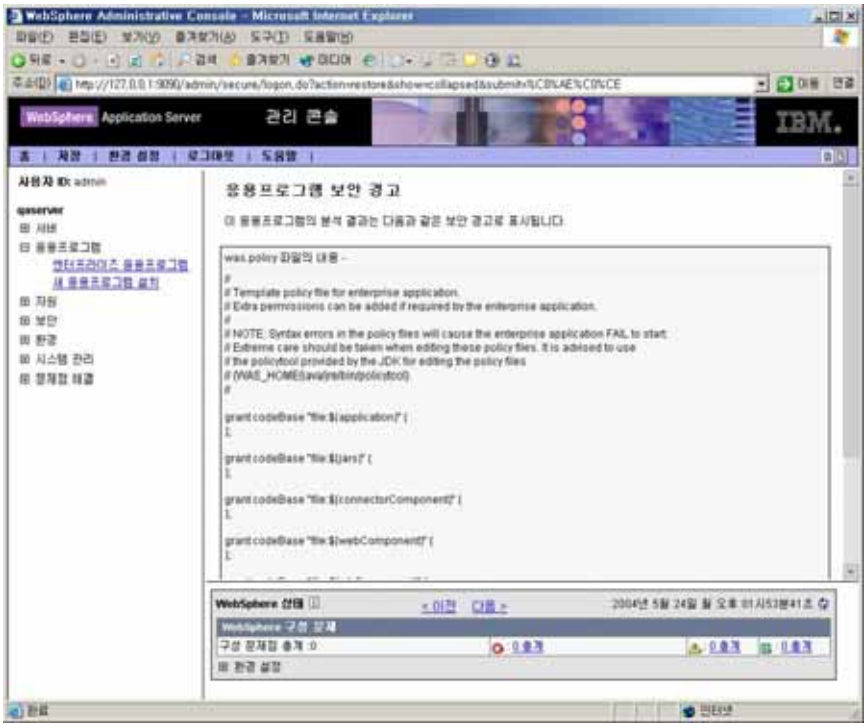
[확인] 버튼을 클릭하면 다음과 같은 창이 나타납니다.



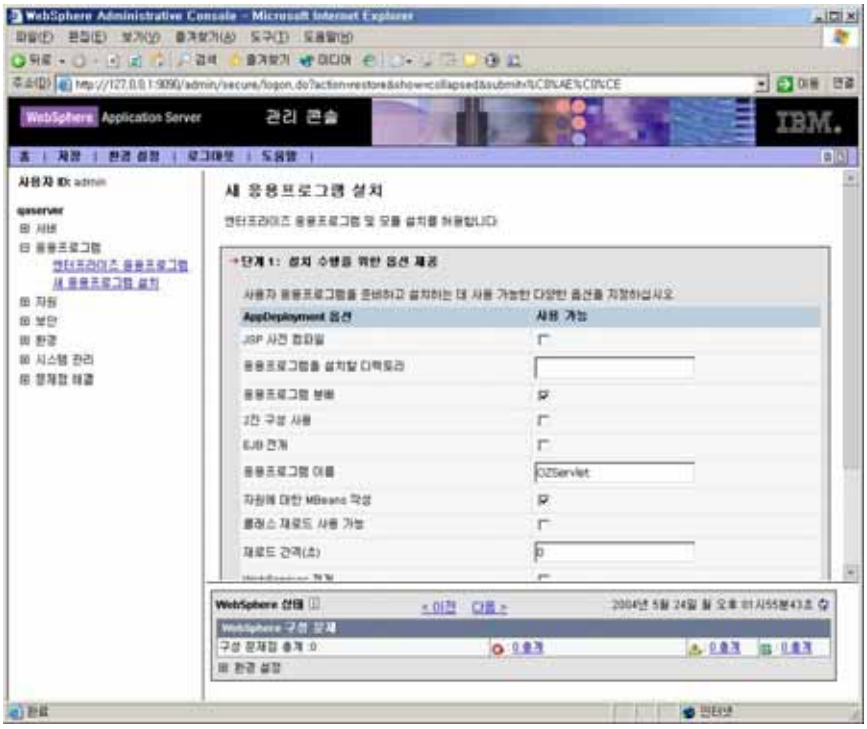
왼쪽 트리에서 [응용프로그램] – [엔터프라이즈 응용프로그램 새 응용프로그램 설치]를 선택한 후 [경로]에서 '로컬 경로'에서 OZServlet.ear 파일을 찾은 후 [다음] 버튼을 클릭합니다.



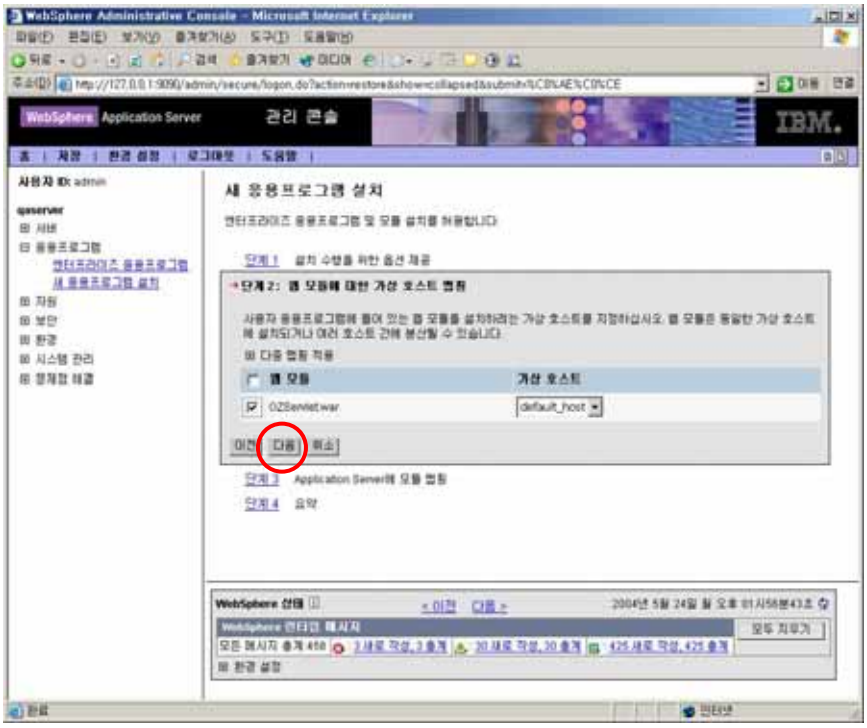
다음과 같은 응용프로그램 보안 경고가 나타나면 [다음] 버튼을 클릭합니다.



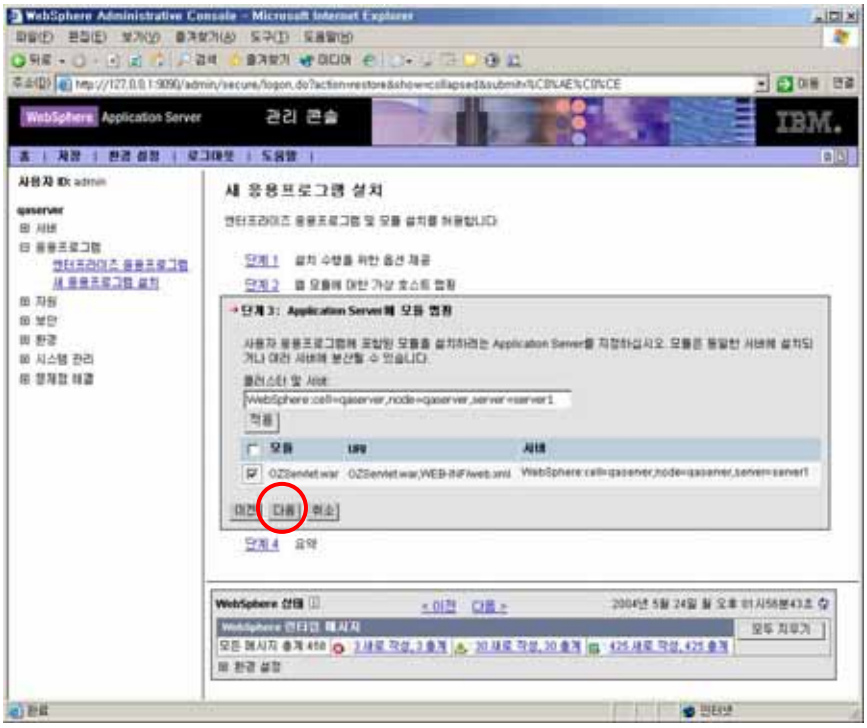
다음과 같이 설치 수행을 위한 옵션을 설정하는 창이 나타납니다. [응용프로그램 이름]란에 'OZServlet'이 설정되어 있는지 확인한 후 [다음]을 클릭합니다.



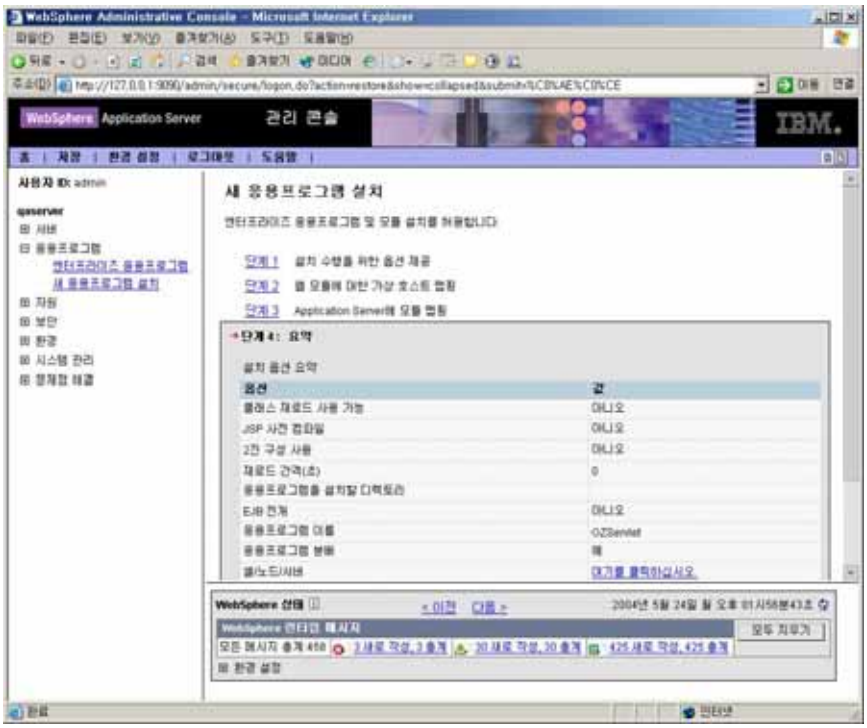
[웹 모듈]과 [가상 호스트]를 확인한 후 [다음] 버튼을 클릭합니다.



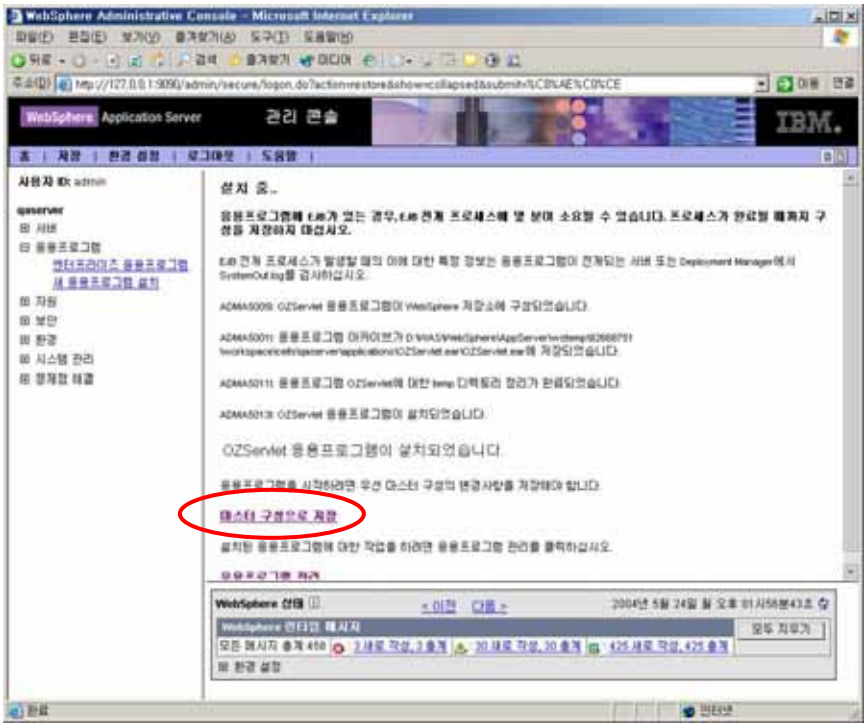
다음과 같은 화면이 나타나면 [다음] 버튼을 클릭합니다.



응용프로그램 관련 정보를 확인한 후 [완료] 버튼을 클릭합니다.



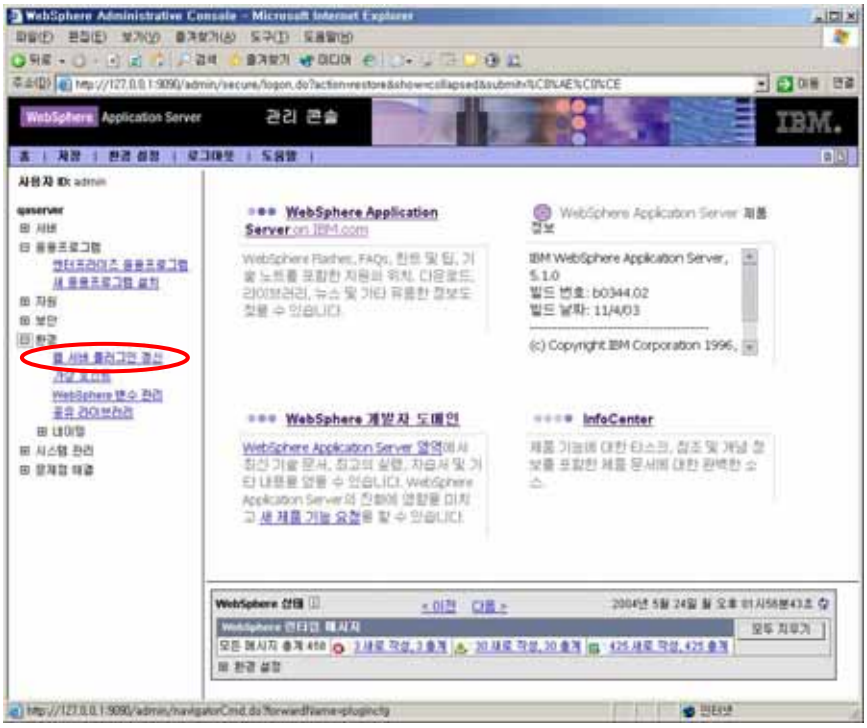
다음 화면과 같이 'OZServlet' 응용프로그램이 설치됩니다. [마스터 구성으로 저장]을 클릭합니다.



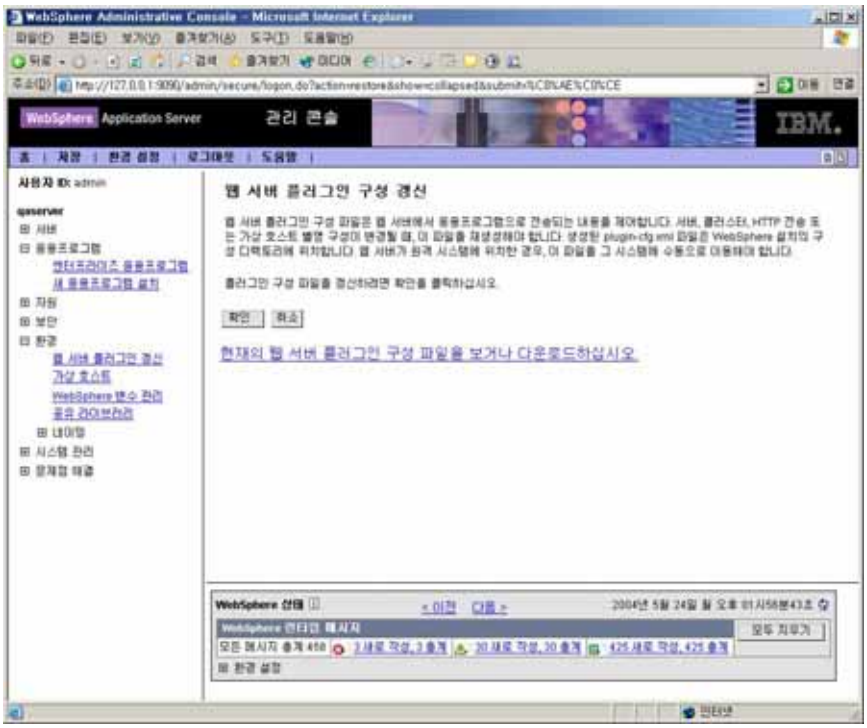
다음과 같은 화면이 나타나면 [저장] 버튼을 클릭합니다.



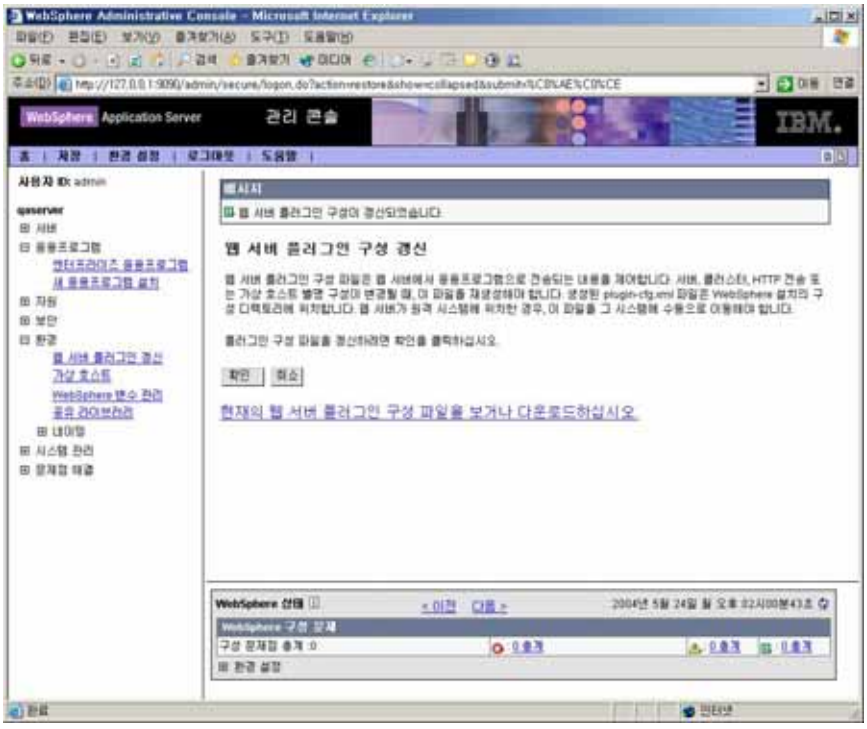
다음과 같은 화면이 나타나면 왼쪽 트리에서 [환경] – [웹 서버 플러그인 갱신]을 선택합니다.



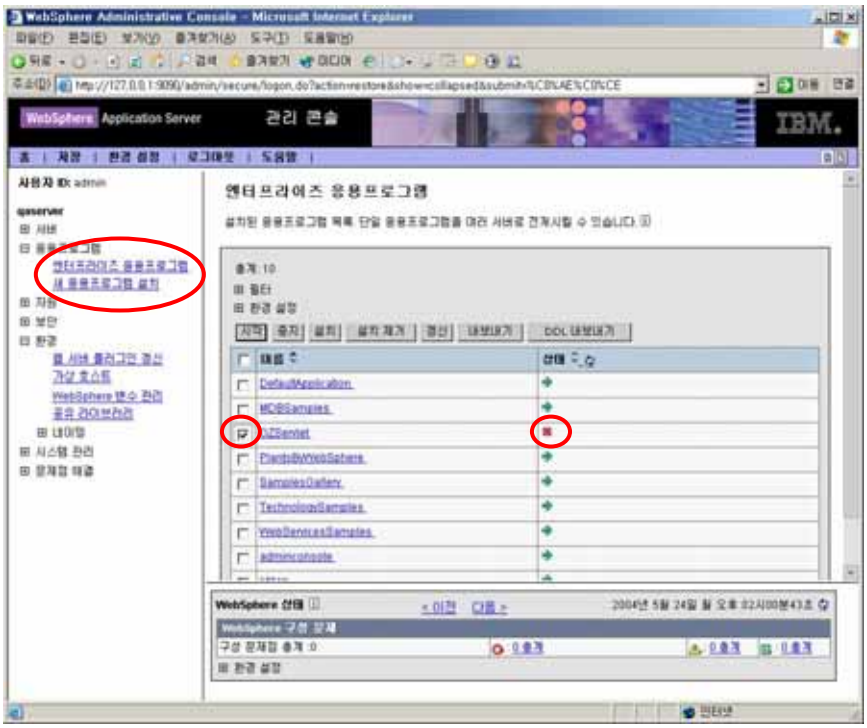
[확인] 버튼을 클릭하여 변경된 환경 파일을 저장합니다.



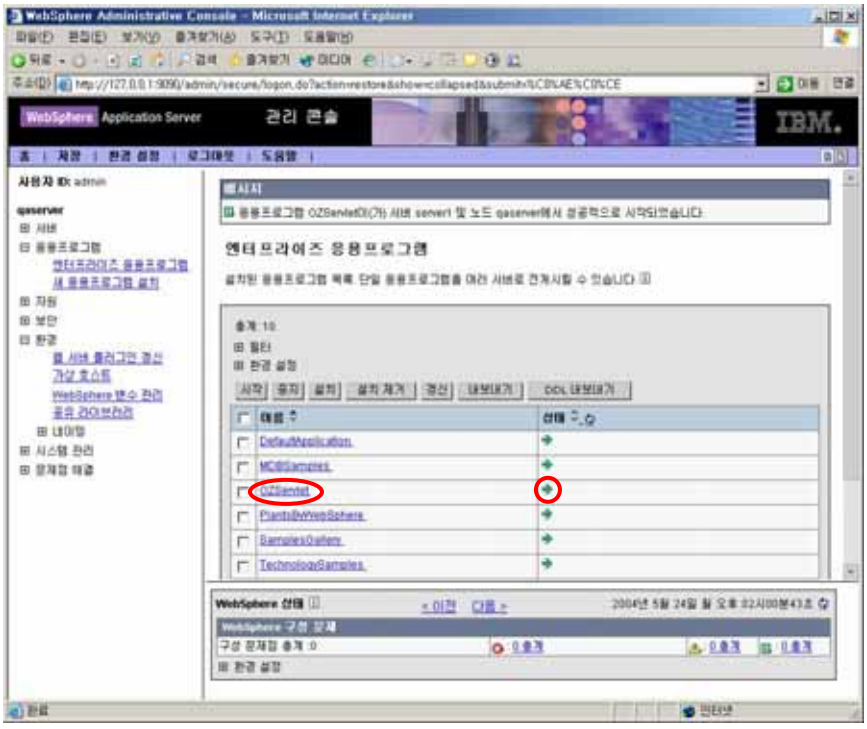
다음과 같이 웹 서버 플러그인 구성이 갱신되었음을 확인할 수 있습니다.



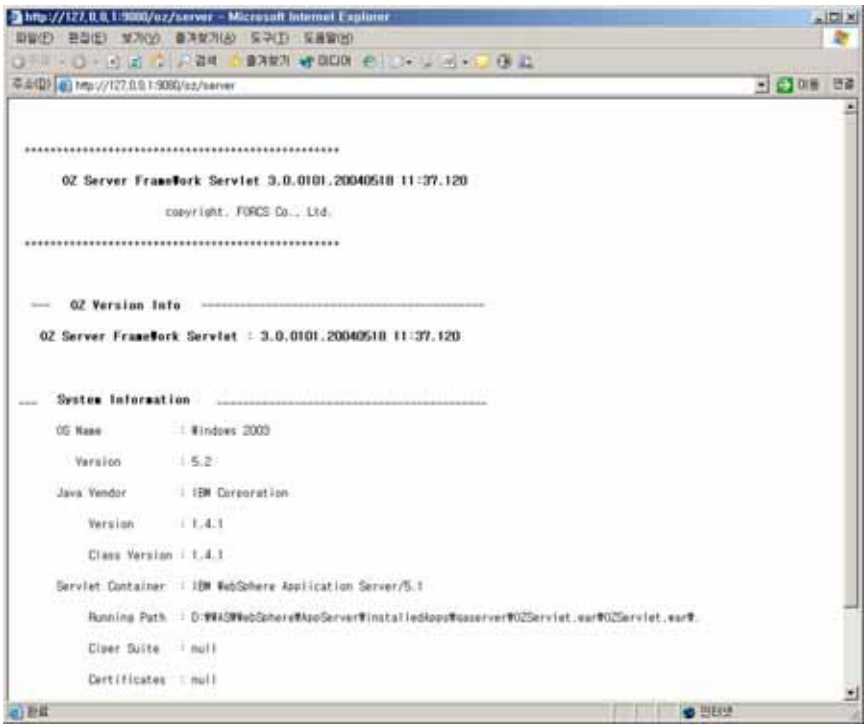
왼쪽 트리에서 [응용프로그램] – [엔터프라이즈 응용프로그램 새 응용프로그램 설치]를 선택한 후 'OZServlet'을 체크한 후 [시작] 버튼을 클릭합니다.



다음과 같이 'OZServle't의 상태가 바뀐 것을 확인할 수 있습니다.



웹 브라우저를 이용하여 http://ip:port/oz/server에 접속하면 오즈 서버 정보 화면이 정상적으로 표시되는지 확인할 수 있습니다.



SilverStream과의 연동

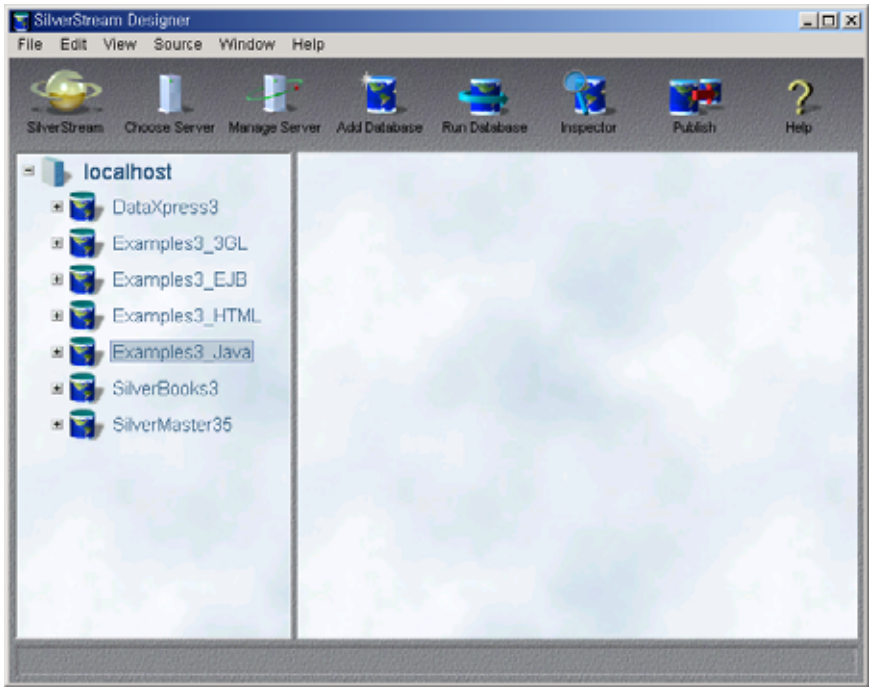
디렉토리 구조

SilverStream 디렉토리는 C:\SilverStream35로 가정합니다. 오즈 서블릿 디렉토리 구조는 데몬 기본 디렉토리 구조와 동일합니다.

설치

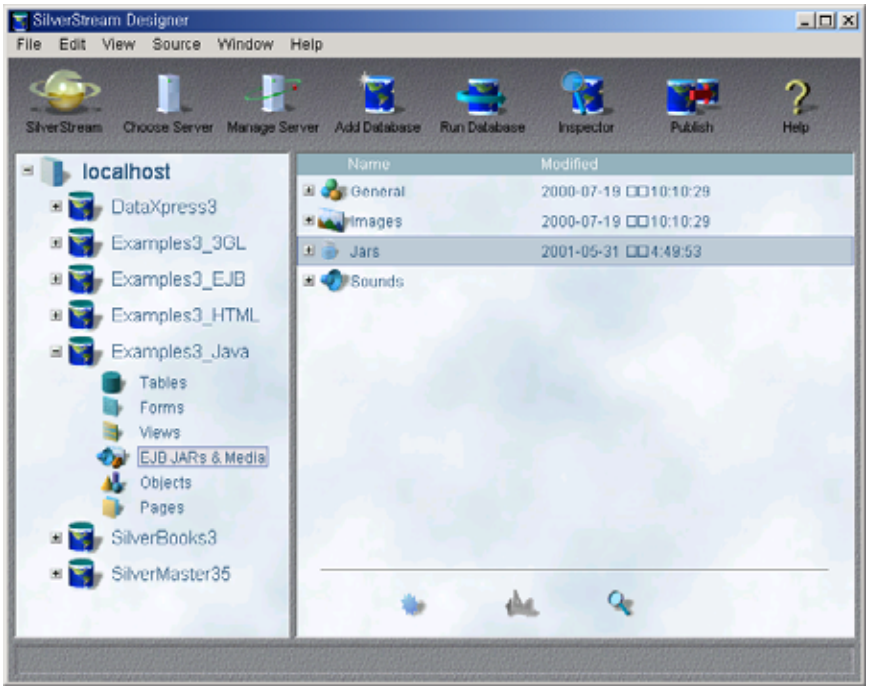
'OZServlet\WEB-INF\classes'에 있는 ozserver.properties 와 'OZServlet\conf'에 있는 db.properties 등을 적절히 수정합니다.

그 후 [SilverStream Designer]를 실행시키면 아래와 같은 화면이 나타납니다.



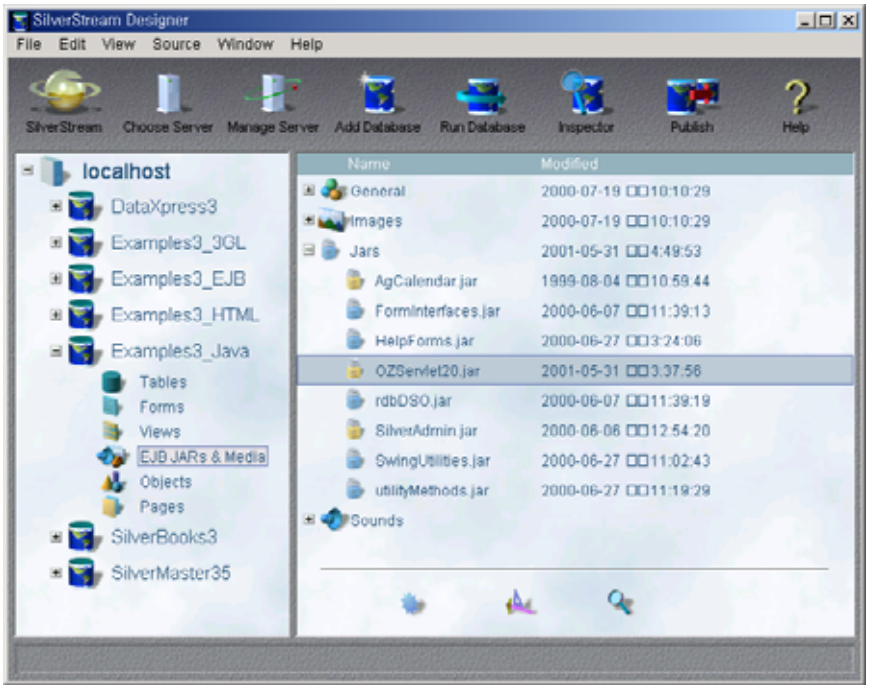
왼쪽 트리는 SilverStream에 연결된 데이터베이스로 이 중 한 곳에 OZServlet을 설치해야 합니다. 설치할 DB를 'Examples3_Java'로 정했다고 가정합니다. ('SilverMaster35'에 설치하는 것은 권장하지 않습니다.)

트리에서 [Examples3_Java] 클릭하면 다음 화면과 같이 서브 그룹들이 나옵니다. 여기에서 [EJB JARs & Media]를 선택하면 아래와 같은 화면이 나타납니다.



오른쪽 화면의 'Jars' 선택하고, 화면 하단의 3개의 아이콘 중 맨 왼쪽에 있는 아이콘( ]- 'New')을 클릭하면 4개의 메뉴가 나옵니다. 이 중 [UPLOAD JAR, ZIP, or EJB] 메뉴를 선택합니다.

다음과 같이 파일 선택창이 뜨면 ./WEB-INF/lib에 있는 ozlib30.jar를 선택합니다. 그러면 아래 화면과 같이 ozlib30.jar이 추가됩니다.

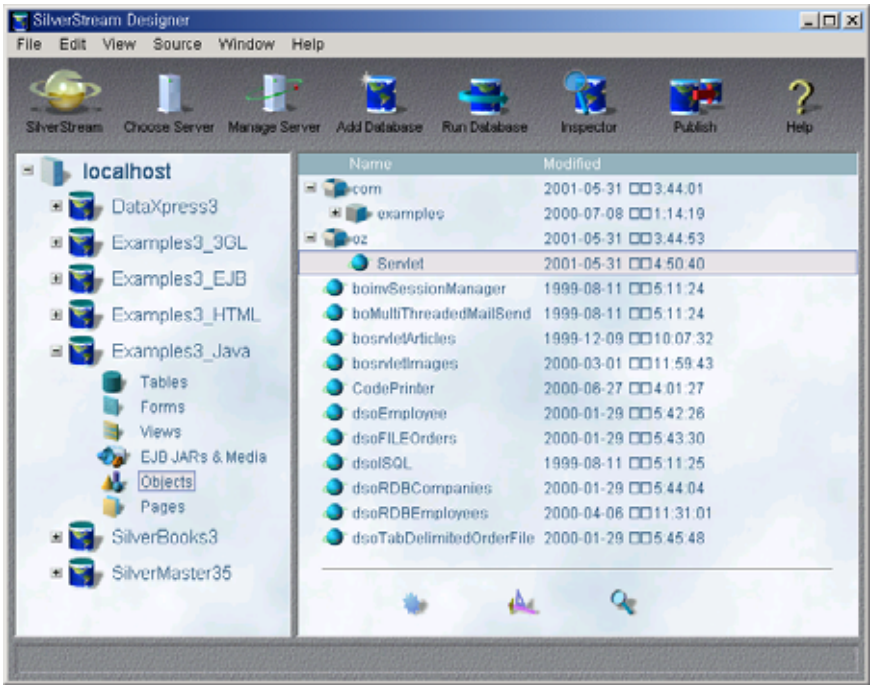


그 다음 아래와 같은 command를 실행시킵니다.

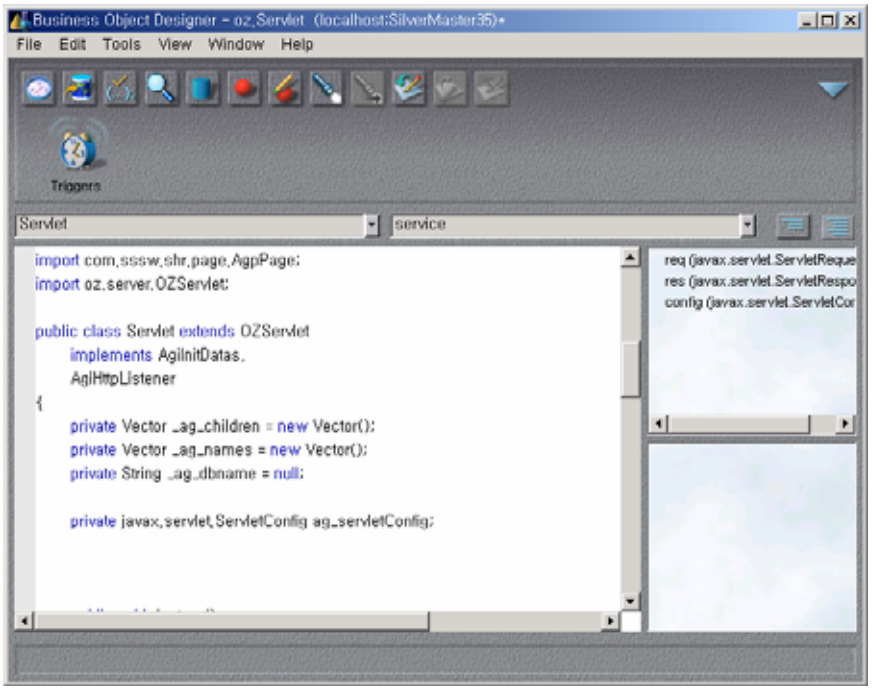
```
C:\SilverStream35\bin>silvercmd ImportSource localhost Examples3_Java
D:\servlet\SilverStream\Servlet.java -f
D:\servlet\SilverStream\Servlet.xml -p:true
```

- localhost는 SilverStream이 설치된 호스트 이름입니다. 만약 SilverStream이 기본적인 80번 port를 쓰지 않고 다른 port 번호를 사용할 경우 localhost:8080과 같이 port 번호를 명시해 주십시오.
- Examples3_Java는 ozserver30.jar를 등록시킨 DB 이름입니다.
- –p:true를 반드시 해 주어야 합니다.
- SilverStream server에 접속하기 위한 username과 password가 있다면 맨 뒤에 –U:username –P:password와 같이 명시해 주어야 합니다.
- 옵션에 대한 자세한 정보는 'silvercmd ImportSource –?'를 실행하여 참조하시기 바랍니다.

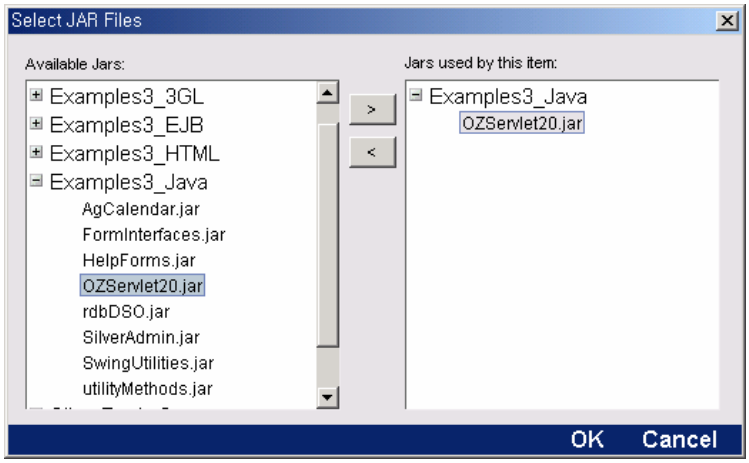
다시 [SilverStream Designer] 화면으로 돌아와서 [Examples3_Java]의 [Objects]를 클릭하면 아래 화면과 같이 'oz'라는 package가 새로 생기고 그 아래에 'Servlet'이라는 항목이 새로 나타납니다.



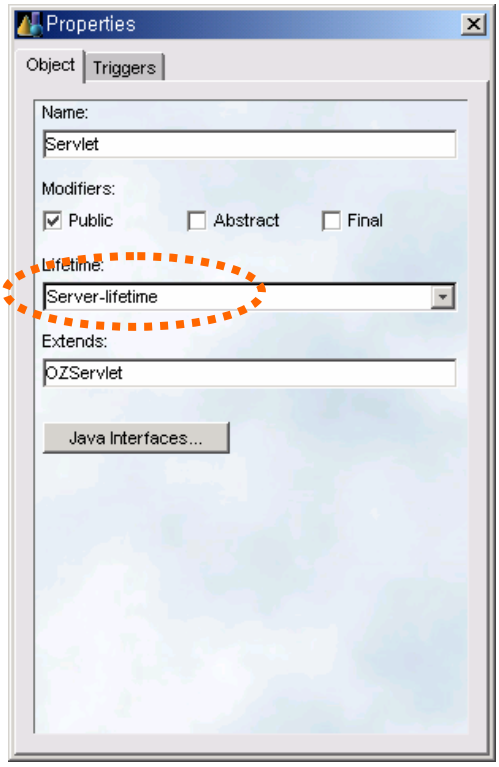
이 'Servlet' 항목을 더블 클릭하면 아래와 같이 [Business Object Designer] 창이 나타납니다.



여기서 메뉴 중 [File] – [Jar Files]를 선택하면 다음과 같이 jar 파일을 선택할 수 있는 [Select JAR Files] 창 하나가 나타납니다. [Available Jars]란에서 'Examples3_Java'의 'ozlib30.jar'를 [Jars used by this item]란으로 옮기면 다음과 같은 화면이 됩니다. [OK] 버튼을 클릭합니다.(만약 다음과 같은 화면이 뜨지 않고 아무런 반응도 없다면 [Business Object Designer] 창을 닫고 다시 띄웁니다. 창을 닫을 때 [Save] 창이 뜨면 'No'로 선택합니다.)



다음으로 화면 위의 [Triggers]라는 아이콘에서 마우스 오른쪽 버튼을 클릭하면 [Properties]라는 메뉴가 나타납니다. 이 메뉴를 선택하면 다음과 같은 화면이 나타납니다.



- [Object] 탭에서 [Lifetime]이 'Server-lifetime'인 것을 반드시 확인합니다. 만약 'Event-lifetime'이라면 반드시 'Server-lifetime'로 바꿔줍니다.
- [Triggers] 탭의 [URLs….] 버튼을 클릭하면 'OZServlet'이 나옵니다. 이 'OZServlet'은 OZServlet을 접근하기 위한 URL path입니다. (실제로 오즈 서블릿 url은 http://localhost/Examples3_Java/OZServlet과 같이 됩니다.)

모든 절차가 끝나면 <Ctrl+S>를 눌러 저장합니다. 저장이 되면서 컴파일 됩니다. 컴파일이 끝나면 창을 닫습니다.

이제 마지막으로 커맨드 상태로 와서 아래의 내용으로 SilverStream 서버를 실행시키는 배치 파일을 만들어야 합니다. 그리고 SilverStream 서버를 이 배치 파일을 통해 실행시킵니다.

SilverStram 실행 배치 파일 생성

```
Set
AGCLASSPATH=C:\OZServlet\conf;C:\OZServlet\lib\ozlib30.jar;\lib\parser.jar;C:\OZServlet\lib\log4j.jar
silverserver -nodbcheck +Dorg.xml.sax.parser=com.sun.xml.parser.Parser
```

- 가장 아랫줄을 제외한 환경 변수 설정은 Windows 2000 Server에서는 [내컴퓨터]의 환경변수 설정에 등록해도 됩니다.
- 위 AGCLASSPATH라는 환경 변수를 세팅한 부분에 오즈 서블릿 라이브러리 jar 파일과 JDBC관련 jar(zip)파일 경로를 추가해 줍니다. (예> C:\OZServlet\lib\jconn2.jar 등)
- –nodbcheck은 SilverStream 서버를 실행시키는데 필요한 옵션입니다. 만약 SilverStream이 설치된 곳에서 원래 –nodbcheck 옵션을 주지 않았다면 생략합니다.

마지막으로 웹 브라우저에서 'http://localhost/Examples3_Java/OZServlet'를 입력하여 OZServlet start 메시지가 제대로 뜨는지 확인하면 설정이 완료됩니다.

Apache Jserv와의 연동

JServ는 Apache 웹 서버 엔진을 사용하는 대부분의 웹 서버 및 애플리케이션 서버에 기본적으로 탑재되어 있는 서블릿 엔진입니다. 여기에서 설명하는 파일 및 디렉토리 구조는 JServ의 기본 구조입니다. Oracle iAS, Netscape Web Server 등은 해당 디렉토리에서 서블릿 세팅 부분을 살펴보면 비슷하게 설정할 수 있도록 이루어져 있습니다. 이외에 WebLogic, WebSphere와 같이 특이한 설정 구조를 가진 제품들을 제외한다면 대부분 설정해야 할 항목들은 유사하기 때문에 JServ를 참조하시기 바랍니다.

디렉토리 구조

{$Jserv_Home} = D:/Apache/Jserv1.1

■ **Jserv의 디렉토리 구조**

[디렉토리]	파일	설명
./conf	jserv.properties	오즈 서블릿의 Classpath 추가
./servlet	zone.properties	오즈 서블릿 Alias 추가

환경 설정 절차

ozlib30.jar 파일과 ozserver.properites 파일이 위치한 디렉토리 그리고 ./WEB-INF/lib 아래에 있는 필요한 파일의 CLASSPATH를 다음과 같이 추가합니다. 이 과정은 기본적인 데이터베이스(Oracle) 설정으로 필요에 따라서는 ./WEB-INF/lib 아래의 파일들을 추가해 줍니다.

■ **jserv.properties**

```
#
wrapper.classpath=D:/apache/JServ1.1/ApacheJServ.jar
wrapper.classpath=D:/jsdk2.0/lib/jsdk.jar

####################    OZ    ############################
wrapper.classpath=D:/servlet2/WEB-INF/classes
wrapper.classpath=D:/servlet2/WEB-INF/lib
wrapper.classpath=D:/servlet2/WEB-INF/lib/ozlib30.jar
wrapper.classpath=D:/servlet2/WEB-INF/lib/log4j.jar
wrapper.classpath=D:/servlet2/WEB-INF/lib/parser.jar
wrapper.classpath=D:/servlet2/WEB-INF/lib/classses12.zip
```

URL Pattern 설정

zone.properties 파일에 다음 항목들을 추가합니다. 오즈 서블릿의 Alias, Startup을 설정합니다(Startup 부분은 웹 서버에 따라 없을 수도 있습니다). 각 항목들은 사용하는 웹 서버의 매뉴얼을 참고 하시기 바랍니다.

■ **{$Jserv_HOME}/servlets/zone.properties**

```
############ S E R V L E T   P A R A M E T E R S ##############
…
# Startup Servlets
# OZ
servlets.startup=ozservlet
# Servlet Aliases
# OZ
servlet.ozservlet.code = oz.server.OZServlet
```

데이터베이스 연결 설정을 확인합니다. 오즈 서블릿의 자체적인 DB Pooling을 사용할 때에는 오즈 서버 데몬 버전에서와 같은 방법으로 db.properties 파일을 설정합니다.

웹 서버 및 서블릿이 실행 중인지 확인하고(http://Jserv-ip로 접속하여 웹 서버 확인), http://Jserv-ip/servlets/ozservlet로 접속하여 오즈 서블릿이 제대로 설치되어 있는지 살펴봅니다.

Tomcat과의 연동

Tomcat은 웹로직과 거의 유사한 환경을 가지고 있습니다. 일부분의 설정을 제외하고는 거의 동일한 설정을 사용합니다.

Tomcat 3.X

■ **디렉토리 구조**

{$Tomcat_Home} = D:/Apache/Tomcat3.2

- Tomcat의 디렉토리 구조

[디렉토리]	파일	설명
./bin	Tomcat.bat/Tomcat.sh	Tomcat 실행 파일
./conft	server.xml	오즈 서블릿의 URL Alias 추가

■ **환경 설정 절차**

- 오즈 등록

  먼저 Tomcat의 설치 경로를 <TOMCAT_HOME>이라고 가정할 때에는 <TOMCAT_HOME>/bin/tomcat.sh 또는 tomcat.bat 파일에 오즈의 클래스 경로를 추가한 후 server.xml 파일에 오즈 서블릿 폼이 d:/servlet이라면 오즈 서블릿의 Alias를 다음과 같이 등록합니다.

```
## oz servlet setting
<Context path="/oz"
docBase="D:/servlet"
crossContext="true"
debug="0"
reloadable="true" >
</Context>
```

- web.xml의 수정

  ./WEB-INF/web.xml 파일은 공통 web.xml과 동일합니다. 먼저 데이터베이스 연결 설정을 확인합니다. 오즈 서블릿의 자체적인 DB Pooling을 사용할 때에는 오즈 서버 데몬 버전에서와 같은 방법으로 db.properties 파일을 설정합니다. 설치 후에는 확인 작업을 수행합니다.

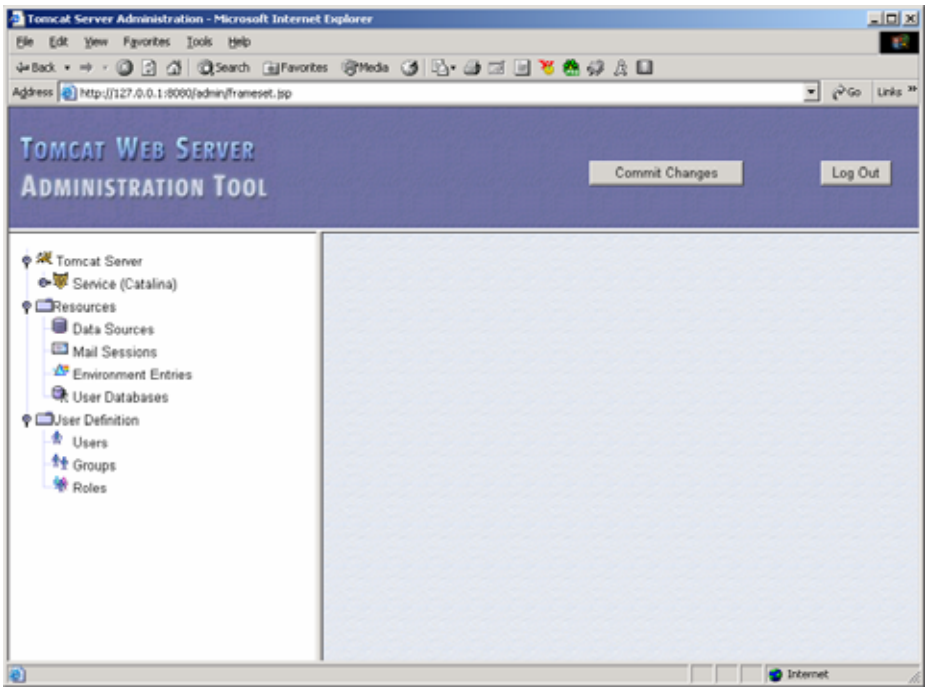
웹 서버 및 서블릿이 실행 중인지 확인하고(http://WebHost-ip/로 접속하여 웹 서버를 확인합니다.), http://WebHost-ip/oz/ozserver로 접속하여 오즈 서블릿이 제대로 설치되어 있는지 살펴봅니다.

Tomcat 4.X와 5.X

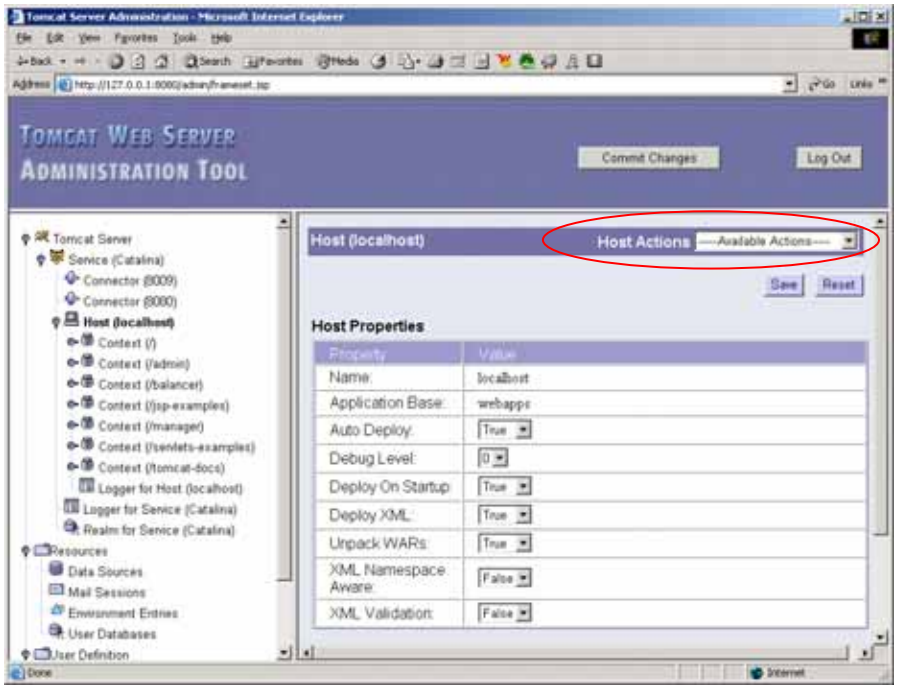
오즈 서버(servlet 타입) 디렉토리를 서버의 설치 디렉토리에 복사합니다.

WEB-INF/classes/ozserver.properties와 conf/db.properties의 정보를 수정하고 오즈 서버에서 사용할 jdbc driver 파일을 WEB-INF/lib/에 복사합니다.

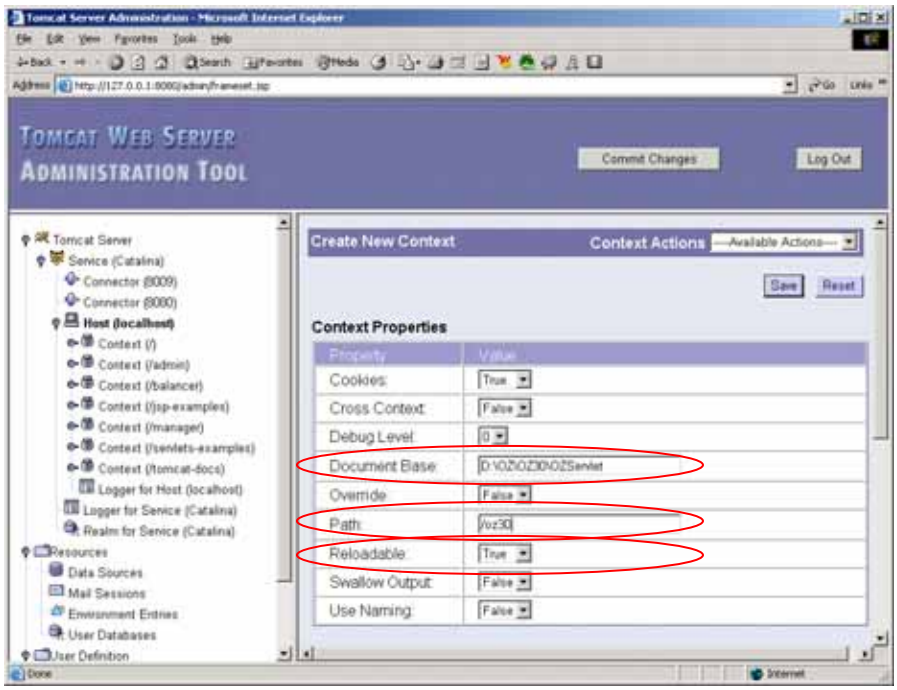
Admin Console을 실행시킵니다. (http://localhost:4848/asadmin/index.html)



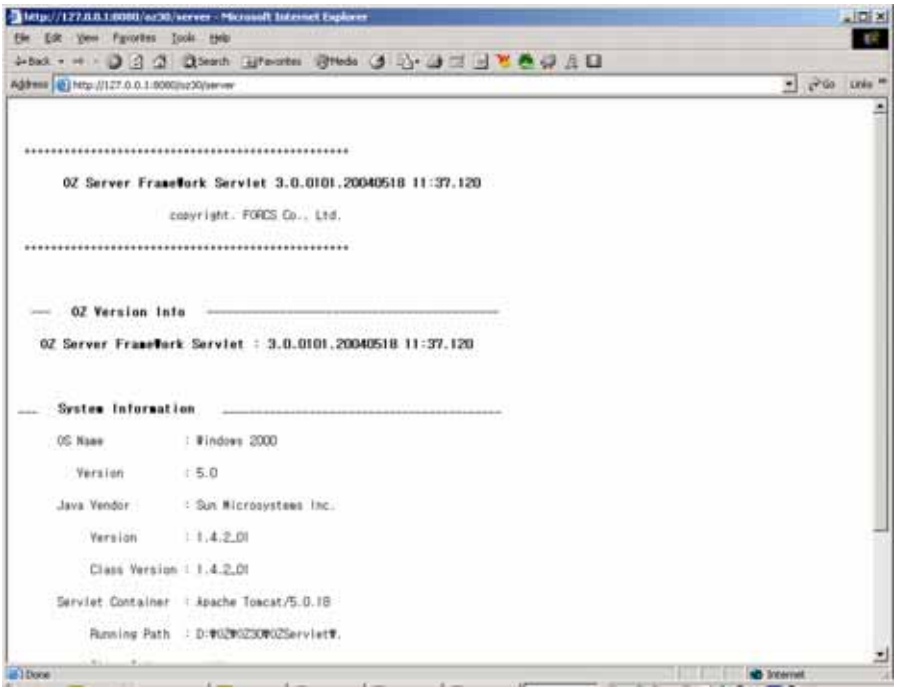
[Tomcat Server] – [Service] – [Host]를 선택합니다.



[Host Actions]에서 'Create New Contexts'를 선택하면 다음과 같은 화면이 나타납니다. [Context Properties]에 오즈 서버 정보를 입력합니다. [Document Base]에는 복사해 놓은 오즈 서버 경로를 입력하고, [Path]에는 오즈 서버에 사용한 URL을, [Reladable]는 'True'로 설정한 후 [Save] 버튼을 클릭합니다.



설치한 오즈 서버가 정상적으로 동작하는지 웹 브라우저를 통해 확인합니다. (http://localhost:8080/oz30/server)



Resin과의 연동

디렉토리 구조

Resin은 특정한 디렉토리 구조를 요구하지 않습니다. 사용자의 편의에 따라 디렉토리 구조를 설정합니다. 오즈 서블릿 디렉토리가 D:\servlet이라고 하면 오즈 서블릿 디렉토리 구조는 앞에서 살펴본 기본 디렉토리 구조와 동일합니다.

환경 설정 절차

Resin의 환경 설정 파일 resin.conf에 오즈 서블릿에 대한 설정을 추가해 줍니다.

■ **{$Resin_Home}/conf/httpd.conf**

```
….
CauchoHost localhost 6802
AddHandler caucho-request jsp
<Location /servlet/*>
      SetHandler caucho-request
</Location>

{$Resin_Home}>/conf/resin.conf
<app-dir>D:/servlet</app-dir>
<serverlet-maping url-pattern='/oz/*'    servlet-name='ozservlet'/>
<serverlet servlet-name='ozservlet'    servlet-class='oz.servlet.OZServlet'>
</servlet>
```

■ **{$Resin_Home}>/conf/resin.conf**

```
<app-dir>D:/servlet</app-dir>
<serverlet-maping url-pattern='/oz/*'    servlet-name='ozservlet'/>
<serverlet servlet-name='ozservlet'    servlet-class='oz.servlet.OZServlet'>
</servlet>
```

클래스 경로 설정은 Resin의 실행 파일을 실행할 때 넘겨 줍니다. 그러므로 오즈에 관련된 디렉토리를 클래스 경로에 추가시켜 주어야 합니다.

```
httpd –Dorg.xml.parser=com.sun.xml.parser.Parser –classpath D:/servlet;
D:/servlet/WEB-INF; D:/servlet/WEB-INF/classes; D:/servlet/WEB-INF/lib;
```

httpd는 실행 파일입니다.
위에서 '–Dorg.xml.parser=com.sun.xml.parser.Parser'는 Parser에 관련된 옵션으로 반드시 넣어주어야 합니다.

데이터베이스가 연결되었는지 확인합니다. 오즈 서블릿의 자체적인 DB Pooling을 사용할 때에는 오즈 서버 데몬 버전에서와 같은 방법으로 db.properties 파일을 설정합니다. 웹 서버 및 서블릿이 실행 중인지 확인하고(http://WebHost-ip로 접속하여 웹 서버를 확인합니다.) http://WebHost-ip/oz/ozservlet로 접속하여 오즈 서블릿이 제대로 설치되어 있는지 살펴봅니다.

Resin (1.2.9 이상)

기존 방식대로 설치해도 상관없으나 resin.conf에 <add-dir>c:\A</add-dir>처럼 이미 다른 디렉토리가 설정되어 있는 경우에는 오즈를 war를 묶어서 설치해야 합니다. 기존의 방식은 <add-dir>C:\Servlet</add-dir>처럼 <add-dir> 위치에 OZ 경로를 설정해 주면 됩니다. War로 묶는 방식을 권장합니다.

- **서블릿 Alias 잡아주기**

  war 파일로 등록할 경우 다른 설정없이 오즈 서블릿의 web.xml에서 URL Pattern으로 사용되게 됩니다.

- **resin.conf**

```
<servlet-mapping url-pattern='/oz/*' servlet-name='ozservlet'/>
<!-- configure oz-servlet with init parameters -->
<servlet servlet-name='ozservlet' servlet-class='oz.servlet.OZServlet'>
<init-param info='A servlet information string'/>
</servlet >
```

- **오즈 파일을 war로 묶기**

  예) c:\oz>jar cvf oz.war *

  위의 예제를 실행시키면 oz.war가 생깁니다. oz.war를 resin의 webapps 폴더에 이동시킨 후 resin을 실행시키면 oz.war 파일이 webapps 폴더에 자동으로 풀립니다.(resin을 실행시킨 후에 webapp 폴더에 이동해도 상관없습니다). 풀린 오즈 파일을 환경에 맞도록 ozserver.properties를 수정합니다.
  웹 브라우저에서 http://localhost:8080/oz/ozservlet로 실행시켜 봅니다.

```
-- SAX parser class was not found (check your CLASSPATH).
java.lang.ClassNotFoundException: com.sun.xml.parser.Parser
```

  오즈 보고서를 호출할 때 위와 같은 에러가 날 경우 xml.properties에서 XML 파서 부분을 수정하면 됩니다. 자세한 사항은 “오즈 엔터프라이즈 서버 관리 매뉴얼(Ⅰ)의 Ⅳ.오즈 엔터프라이즈 서버 설정-xml.properties” 부분을 참조하시기 바랍니다.

Inprise Application Server와의 연동

IAS(Inprise Application Server)는 기본적으로 Tomcat 서블릿 엔진을 사용하며 현재 IAS 4.5 버전이 지원하는 Tomcat Servlet 3.2가 기본적으로 설치되어 있습니다.
오즈 서블릿을 연동하기 위해서는 두 가지의 방법이 있습니다.
줄바꿈 첫번째 방법은 오즈 서블릿의 모든 파일을 IAS의 서블릿 폴더에 복사한 다음 오즈의 web.xml을 수정하는 방법입니다.
줄바꿈 두번째 방법은 IAS에서 Tomcat이 설치된 폴더의 server.xml 파일 설정을 수정하여 오즈 서블릿을 등록하고 오즈의 web.xml을 적절히 수정하는 방법입니다.

Inprise Application Server 4.1 이하

■ IAS 서블릿 디렉토리에 복사해서 설치

- 파일의 복사
  오즈 서블릿의 파일들을 IAS의 서블릿 디렉토리에 복사하기 위해 /Borland/AppServer/var/servers/<SERVERNAME>/wars/tomcat3/ 아래에 'oz'라는 디렉토리를 만든 후 오즈 서블릿의 파일들을 복사합니다 (<SERVERNAME>은 설치시 설정한 서버의 이름입니다). 이때 'oz' 디렉토리가 오즈 서블릿을 호출하는 Alias가 됩니다. 이렇게 생성한 oz의 하위 디렉토리는 Tomcat 설치시와 동일한 구조로 이루어지며 오즈 서블릿 기본 구조입니다. 오즈 서블릿 디렉토리 구조는 앞서 살펴본 기본 디렉토리 구조와 동일합니다. ./oz/WEB-INF/web.xml 파일을 수정합니다.

- web.xml의 수정
  오즈를 호출하기 위한 URL 맵핑에 대한 정보를 입력합니다. 기본 설정과 동일하게 설정하며, 거의 수정할 필요가 없습니다.

```
<servlet-mapping>
<servlet-name>httpserver</servlet-name>
<url-pattern>/ozservlet/*</url-pattern>
</servlet-mapping>
```

- 오즈 서블릿의 실행 확인
  설치 후에는 확인 작업을 수행합니다. 서블릿의 URL은 다음과 같습니다.

```
http://Server_name:8080/ozservlet/server
```

■ **Tomcat의 설정을 직접 수정해서 설치**

- 파일 수정
  /Borland/AppServer/var/servers/<SERVERNAME>/adm/tomcat3/conf/server.xml 파일의 내용 중에서 다음 사항을 추가합니다. 이 곳에서 path="/oz"가 오즈 서블릿의 URL Alias입니다. (앞에서 설정한 Tomcat 설정과 같습니다.) 이때 오즈의 서블릿 경로를 D:\servlet이라고 가정합니다.

- servlet.xml

```
<Context trusted="false"
docBase="D:/servlet"
reloadable="true"
debug="0"
path="/oz"
crossContext="true">
</Context>
```

- web.xml 파일 수정
  오즈의 web.xml 파일을 Tomcat의 설정을 참조해서 수정하시기 바랍니다. 거의 수정할 필요가 없습니다.

- 오즈 서블릿의 확인
  설치 후에는 확인 작업을 수행합니다. URL은 다음과 같습니다. 데이터베이스 연결을 설정 확인합니다. 오즈 서블릿의 자체적인 DB Pooling을 사용할 때에는 오즈 서버 데몬 버전에서와 같은 방법으로 db.properties 파일을 설정합니다.

```
http://servername:8080/oz/server
```

Inprise Application Server 4.1 설정

■ **ClassPath 잡기**

IAS의 클래스 패스는 java.options라는 파일에서 잡아야 합니다. 이 파일은 IAS 서버의 루트 폴더에서 properties/server/<서버이름> 폴더에 위치해 있습니다.

예) C:\Inprise\AppServer\properties\server\goldenhawk\java.options

이 파일을 에디터로 열면 아래와 같은 구문을 볼 수 있습니다.

```
# Set the default heap size to 32 Mb
-Xms32m
# Set the maximum heap size to 32 Mb
-Xmx32m
# Make the default buffer size 1024 bytes
-Dvbroker.orb.streamChunkSize=1024
```

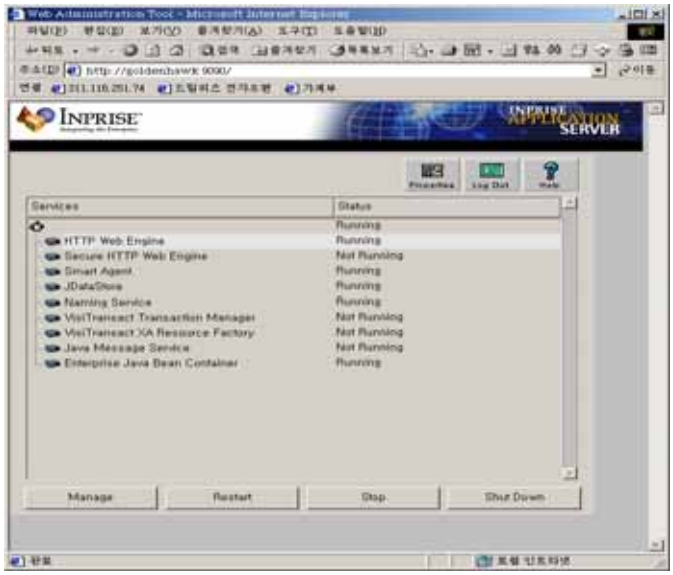
여기에 다음처럼 오즈 서블릿의 클래스들을 등록 시킵니다. 여기서 오즈 서블릿은 c:\Inprise\AppServer\oz에 위치한 것으로 가정합니다.

```
# Set the default heap size to 32 Mb
-Xms32m
# Set the maximum heap size to 32 Mb
-Xmx32m
-classpath     C:\Inprise\AppServer\oz\WEB-INF\lib\ozlib30.jar;
C:\Inprise\AppServer\oz\WEB-INF\lib\parser.jar;
                    C:\Inprise\AppServer\oz\WEB-INF\lib\log4j.jar;
C:\Inprise\AppServer\oz\WEB-INF\classes;
C:\Inprise\AppServer\oz\WEB-INF\lib\crimson.jar;
```

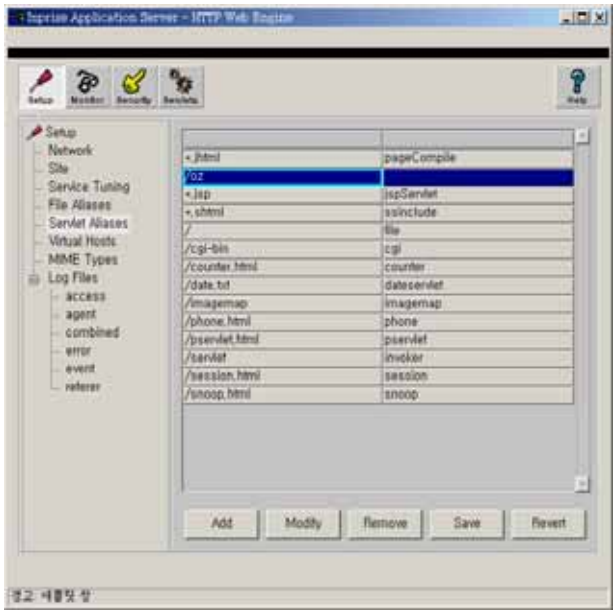
오즈 서블릿이 사용하는 라이브러리 파일이기 때문에 반드시 입력해야 합니다. 또한 사용하는 DB의 jdbc 드라이버 파일로 입력해야 합니다.
여기서는 ozlib30.jar, parser.jar, log4j.jar, classes폴더, crimson.jar 파일들이며, jkjdbc.jar는 MS-SQL용 jdbc 드라이버입니다. 만약, 사용자의 다른 DB(Oracle, Sybase, Informix, DB2 등)를 쓴다면 여기서 쓰는 jdbc 드라이버로 대체하여 입력해야 합니다.

- **Servlet의 Alias 등록**

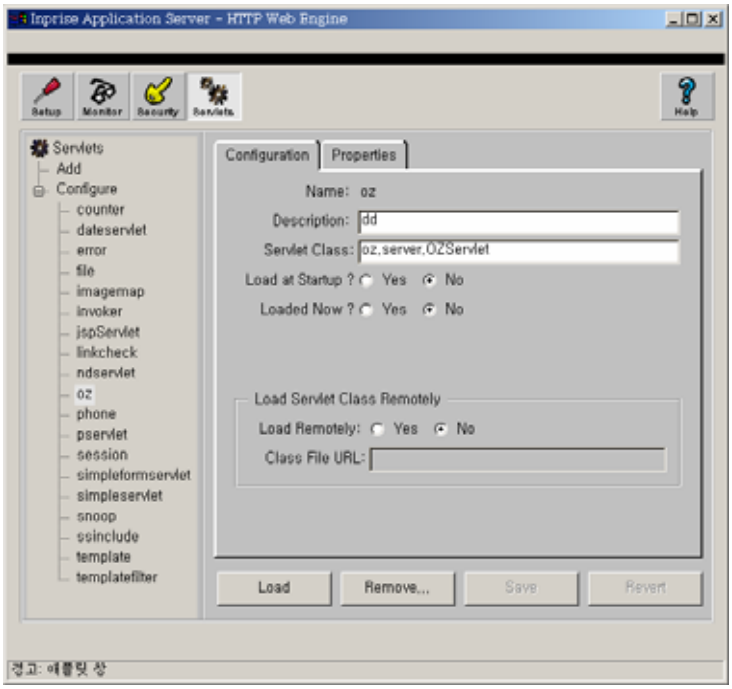
  관리 모드로 들어가서 [HTTP Web Engine]을 선택한 후 [Manage]를 클릭합니다.



[Servlet Aliases]에 '/oz'를 등록 시킵니다. ('/oz'는 오즈 서블릿의 Alias가 됩니다.)



화면 상단 메뉴중에서 [Servlets]를 클릭한 후 다음 화면과 같이 서블릿을 등록시킵니다.



모든 등록이 완료되면 [Save] 버튼을 클릭합니다.

모든 작업이 끝나면 IAS를 재실행합니다.
웹 브라우저에서 http://localhost:8080/oz/server로 실행시킵니다.

BES(5.0.2)와의 연동

BES에 OZServlet 설치

OZServlet30를 임의의 디렉토리에 놓습니다.

예) C:\OZServlet30 이라는 이름으로 올려 놓습니다.

복사한 디렉토리를 선택하여 오른쪽 마우스를 클릭하여 읽기전용 체크 옵션을 삭제합니다.

C:\OZServelet30\WEB-INF\classes\ozserver.properties 파일을 열어 다음 부분을 수정합니다.

Path.oz.server.home=c:\OZServlet30

C:\OZServlet30\conf\db.properties 파일을 열어 DB Pool을 설정합니다.

윈도우의 [시작] – [실행] – [cmd]에서 c:\OZServlet30 디렉토리로 접근합니다. war로 묶기 위해 다음과 같은 옵션을 작성합니다.

jar –cvf oz.war *.*

그러면 c:\OZServlet30\oz.war라는 파일이 생성됩니다.

BES(5.0.2)와의 연동

BES(Borland Enterprise Server)를 실행하여 console을 실행합니다.

Standard 부분에서 웹 모듈을 등록합니다. Deploy Wizard를 이용하여 oz.war 파일을 등록합니다.

등록이 완료되었으면 웹 브라우저를 띄워 다음과 같은 주소를 입력합니다.

```
http://ip:port/oz/server
```

ip : 개인 컴퓨터에 등록될 경우는 local host가 되며, 개발 서버에서는 개발 서버 ip 주소가 됩니다.

port: 사용중인 port입니다.

oz : 앨러어스명 입니다.

server : url-pattern입니다.

정상적으로 등록이 되었다면 http://localhost:8080/ozserver로 오즈를 실행합니다.

Oracle OC4J(ORION)와의 연동

기본 오즈 서블릿 설정을 맞추고 (ozserver.properties와 db.properties) war 파일로 만듭니다. 만약 /home/oz/ozservlet에 위치해 있다면 그 디렉토리에서 jar cvf servlet2.war *를 실행시킵니다. servlet2.war 파일을 orion 디렉토리 아래 applications 디렉토리에 복사하고 오즈 응용프로그램을 등록합니다.

오리온의 home/config에 있는 applicaton.xml에 다음과 같이 추가합니다.

application.xml

```
<web-module id="defaultWebApp" path="../default-web-app" />
<web-module id="servlet2" path="../applications/servlet2.war" />
```

home/config/default-web-site.xml에 다음 내용을 추가합니다.

default-web-site.xml

```
<default-web-app application="default" name="defaultWebApp" />
<web-app application="default" name="servlet2" root="/oz" />
```

오리온을 재시작하면 오즈가 등록됩니다. 다음을 실행하면 오즈 서블릿 확인 페이지가 나타납니다.

```
http://orion.server.ip:port/oz/server
```

HPAS(Bluestone) 8과의 연동

System Console을 실행하여 HP_AS를 띄웁니다. http://localhost:9090/으로 실행여부를 확인합니다. 오즈 서블릿을 적당한 디렉토리에 복사하고 다음과 같이 환경을 설정해 줍니다. 참고로 오즈 서블릿을 D:\ozservlet에 복사한 것으로 가정합니다.

{$HPAS}\config\hpas\ 아래에 있는 j2ee-partition-service-config.xml 파일을 다음과 같이 수정합니다.

j2ee-partition-service-config.xml

```
<war application="HPAS_DEFAULT"
context-path="/helloxtf"
```

```
mime-type="application/x-java-directory"
reload-on-archive-mod="false"
temp-directory="C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp"
timestamp="1008831723679"
url="file:///C:/was/hpas/trailmaps/intermediate/xtf/helloxtf/"/>
```

위의 내용을 복사한 후 바로 아래 부분에 붙여 주고, context-path와 url 주소만 변경하고 재시작 합니다.

```
<war application="HPAS_DEFAULT"
context-path="/oz"
mime-type="application/x-java-directory"
reload-on-archive-mod="false"
temp-directory="C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp"
timestamp="1008831723679"
url="file:///d:/ozservlet"/>
```

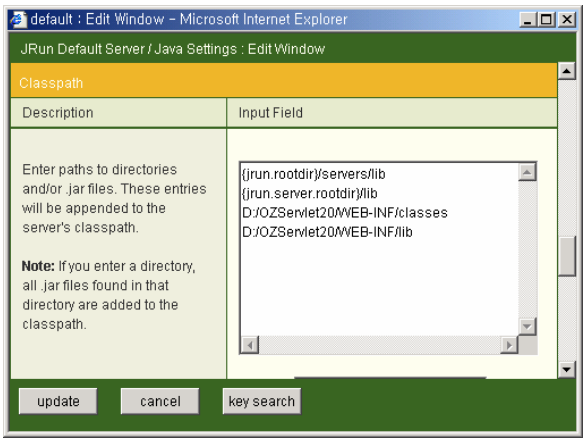
temp-directory와 timestampe는 환경에 맞게 설정합니다.
다음을 실행하면 오즈 서블릿 확인 페이지가 나타납니다.
http://localhost:9090/oz/server

JRun과의 연동

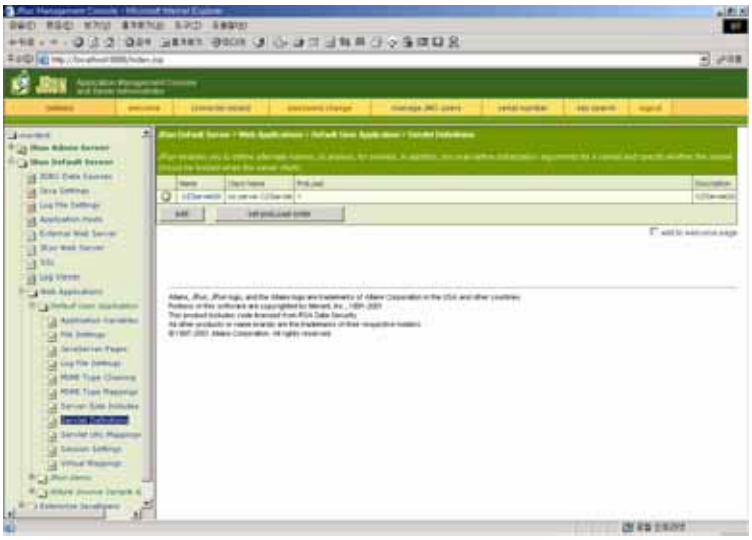
JRun 3.1과의 연동

OZServlet을 폴더(d:\ozservlet에 가정)로 복사한 다음 JRUN 어드민 서버를 실행합니다. 어드민 서버에서 좌측에 뜨는 메뉴 중 설치할 서버를 선택한 후 Java Settings의 Classpath를 누릅니다. (보통은 default server에 설치합니다)

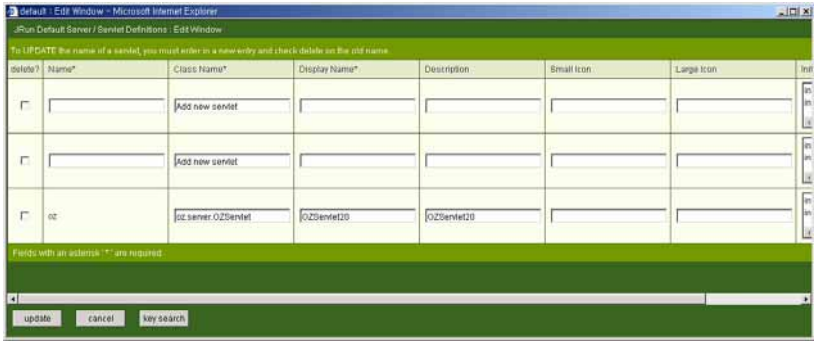
오즈 서블릿 전체 폴더 중 WEB-INF\lib와 WEB-INF\classes까지만 설정하고 [update] 버튼을 클릭합니다.



다음은 어드민 서버에서 좌측에 뜨는 메뉴 중 설치할 서버를 선택한 후 Web Applications → Default User Application → Servlet Definitions의 메뉴로 들어갑니다.



메뉴에서 [Edit]를 선택하고 [name]에는 웹에서 호출할 alias를 설정(여기서는 oz로 설정), [Class Name]에는 오즈 서블릿의 메인 클래스인 'oz.server.OZServlet'을 설정, [Display Name]에는 원하는 이름을 주고 [update] 버튼을 클릭합니다.



JRUN을 재시작하고 설치한 오즈 서버가 정상적으로 동작하는지 웹 브라우저(http://주소:8100/servlet/oz(Alias명))를 통해 확인합니다. 주의할 점은 JRUN 자체에서 servlet이란 Alias명 아래 오즈 설정 Alias명이 설정됩니다.

※ 참고

오즈 WEB-INF₩web.xml은 JRUN에서는 사용되어지지 않으므로 다른 WAS 설정의 http://주소:포트/oz/ozservlet와 다릅니다.

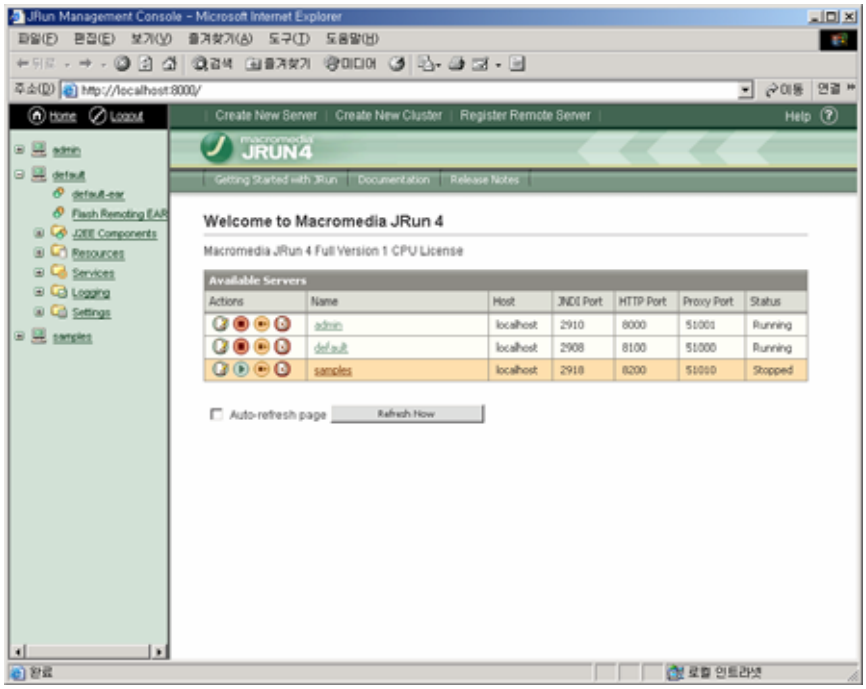
JRUN 4.0과의 연동

JRUN 4.0이 설치된 환경에 따라 설치하여 오즈 서블릿 디렉토리에서 ozserver.properties 경로를 맞춘 후 jar cvf oz.war *.*와 같이 해서 oz.war 파일을 만듭니다.

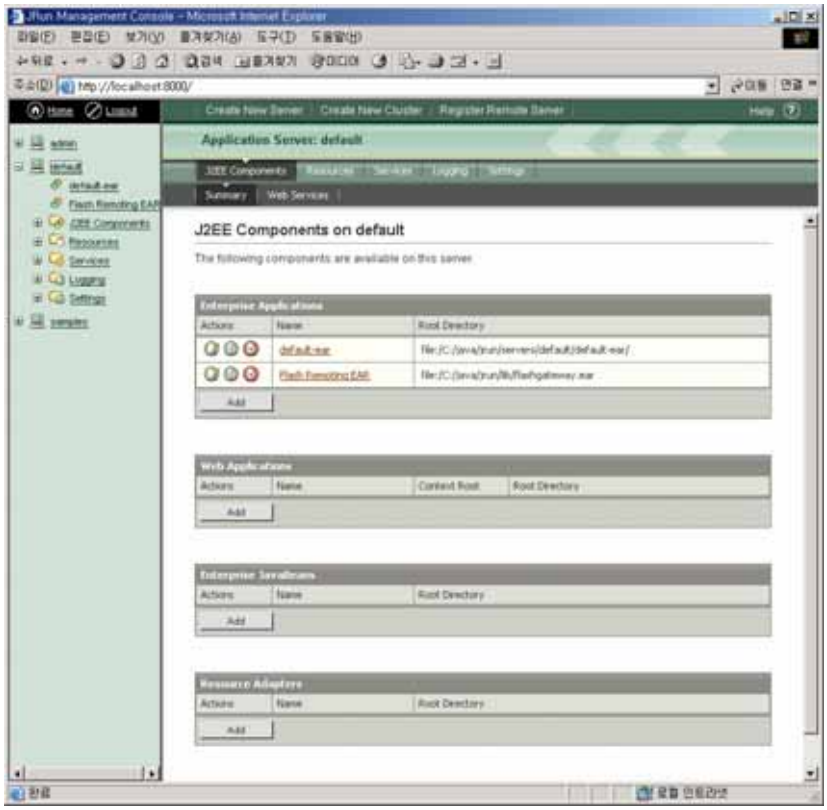
JRUN console 창을 엽니다.

http://localhost:8000/

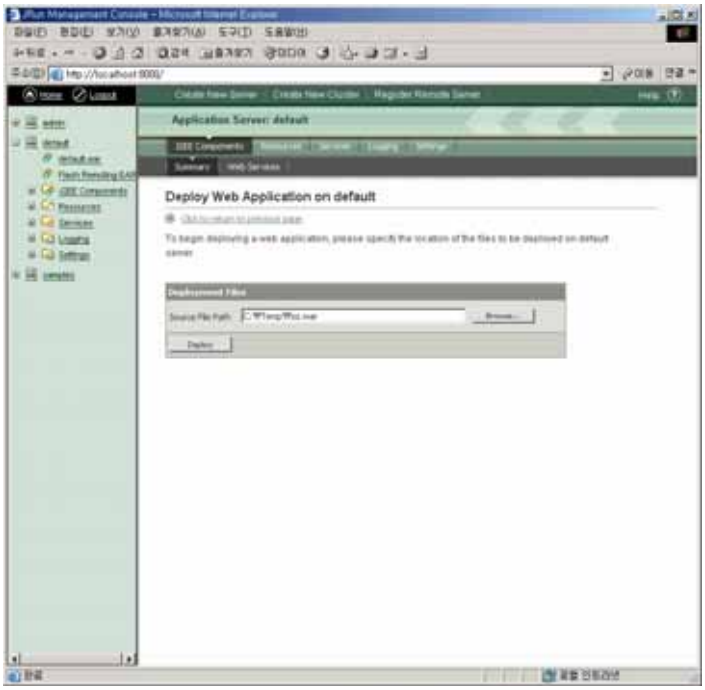
login / password를 입력하면 다음과 같은 창이 나타납니다.



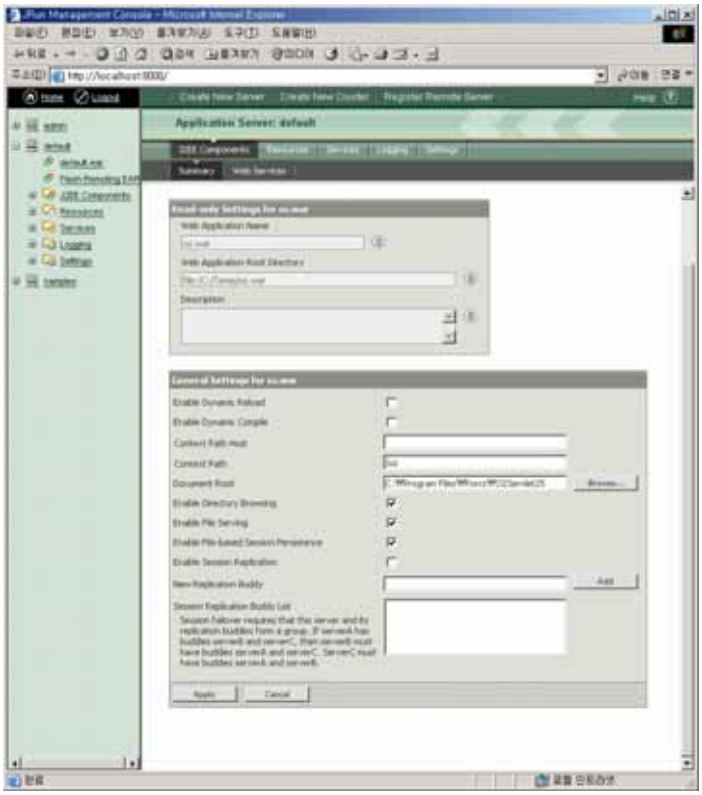
오른쪽 트리에서 [default]를 선택합니다.



[Web applications]탭에서 [add]를 선택합니다.



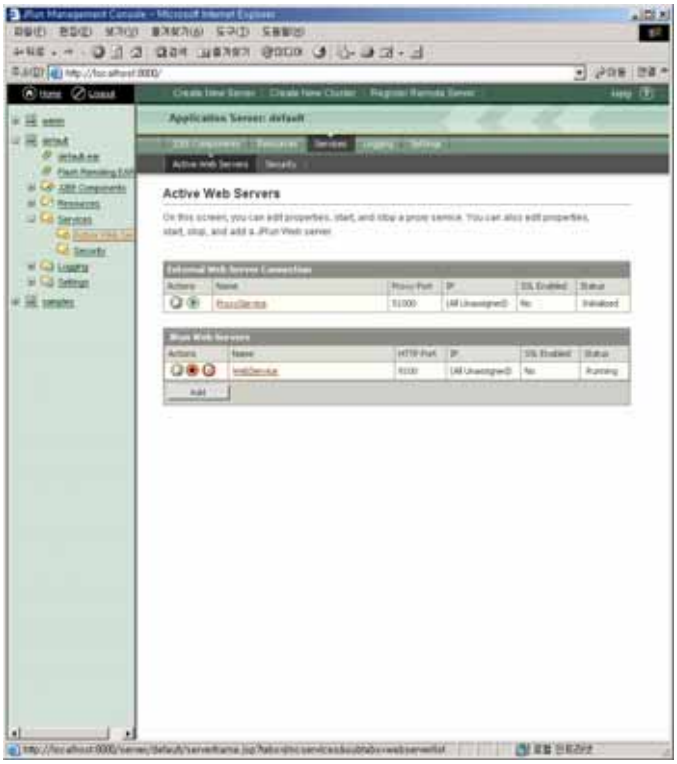
위에서 만든 war 파일을 선택을 한 후에 [Deploy] 버튼을 클릭합니다.



[general setting] 부분에서 해당 부분을 설정하고 [apply] 버튼을 클릭합니다.
http://localhost:8100/oz/server로 하여 웹 브라우저를 통해 확인할 수 있습니다.



default 서버의 포트는 [default] – [Service] – [Active Web Server]에서 확인할 수 있습니다.



EAServer와의 연동

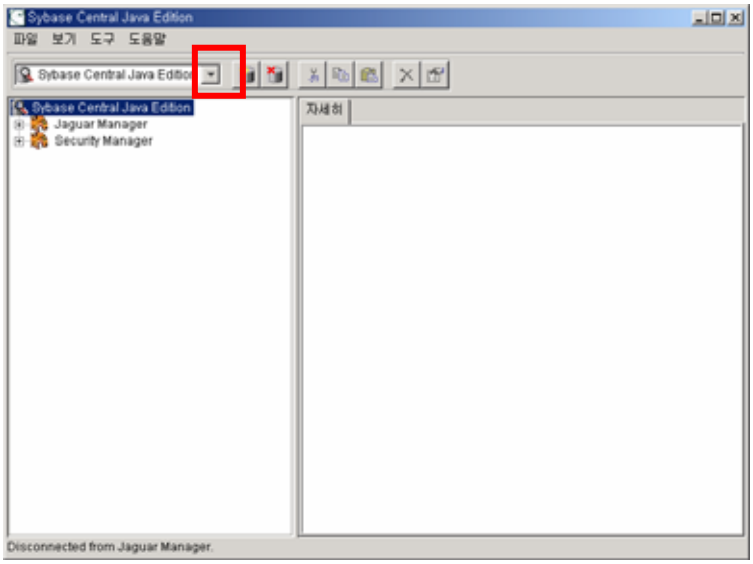
OZServlet을 오즈 서블릿 홈에서 oz.war 파일로 묶습니다. jar 명령은 jdk가 설치되어 패스가 잡혀있어야 실행이 가능합니다.

예) jar cvf oz.war *

Jaguar Server를 실행시킨 다음 Jaguar Manager를 실행 시킵니다. Jaguar Server가 먼저 실행되어야만 Jaguar Manager가 실행됩니다.

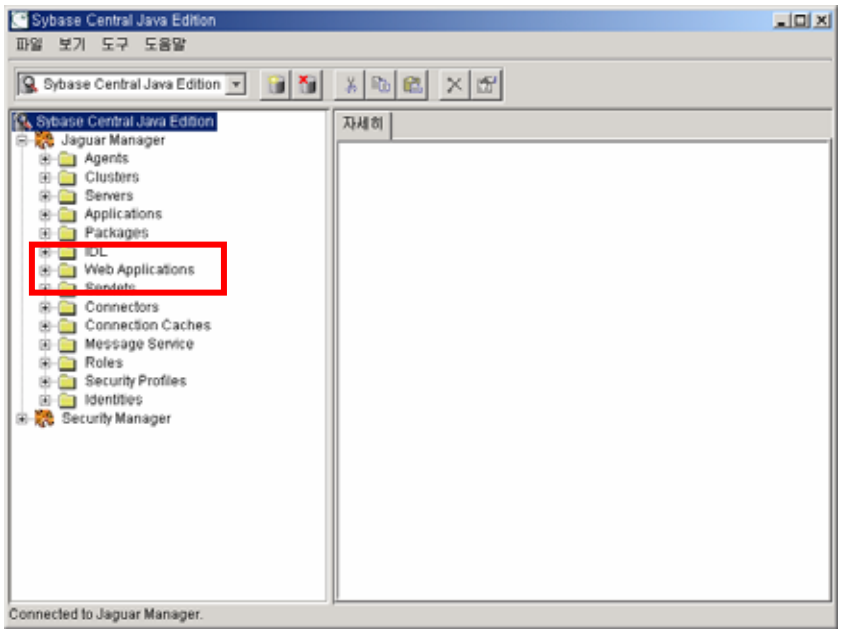


<Jaguar Server 실행 화면>

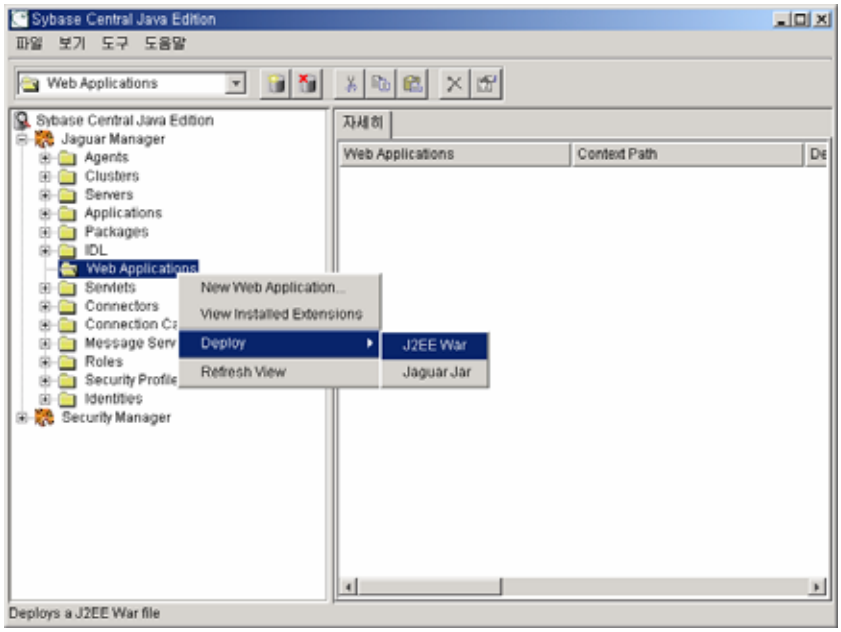


<Jaguar Manager 실행 화면>

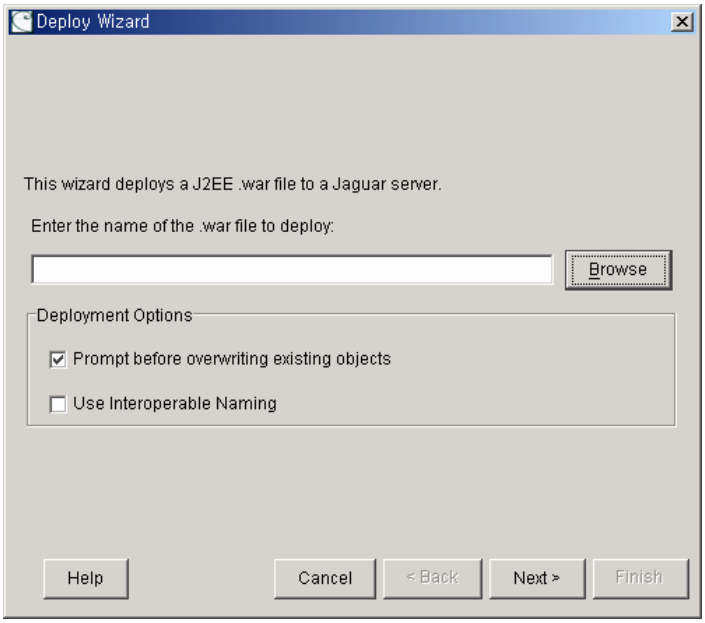
Jaguar Manger가 Jaguar Server와의 커넥션을 맺도록 Jaguar Manger의 데이터베이스 모양의 버튼을 클릭합니다.



war로 묶은 파일들을 전개하기 위해 위의 [web Applications]에 등록합니다.

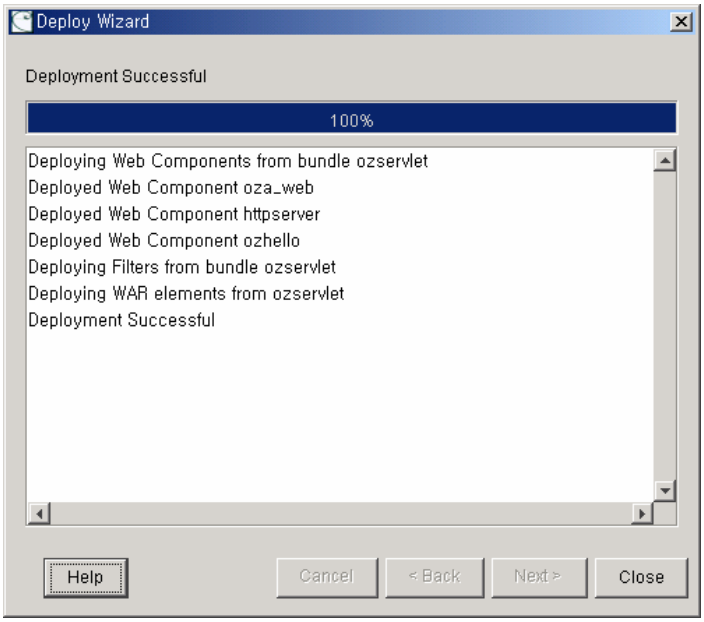


[Web Applications]에 등록하는 화면입니다.

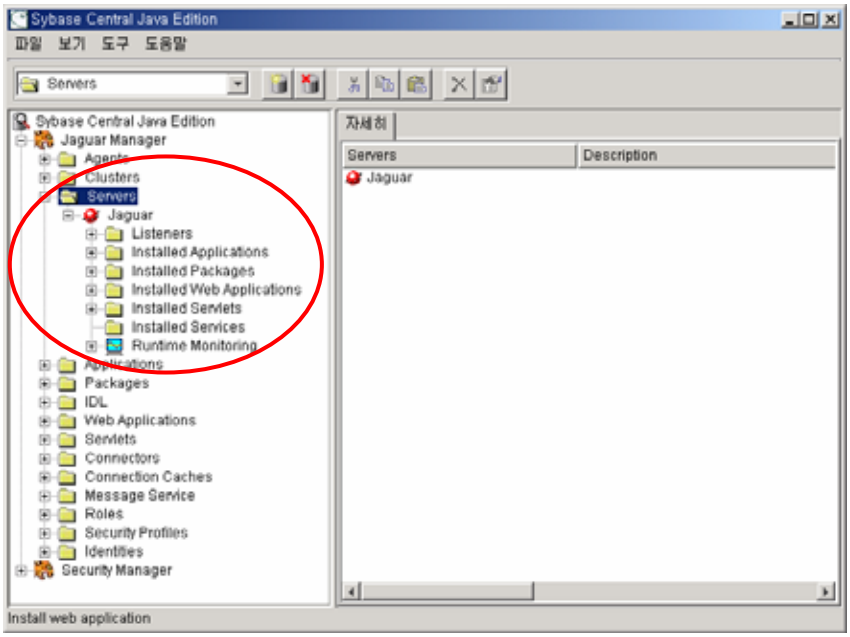


J2EE WAR를 선택하면 다음과 같은 화면이 나타납니다. [Browse] 버튼을 클릭한 뒤 war 파일로 묶은 OZServlet 파일을 선택합니다. 나머지 설정은 Default로 한 후 [Next] 버튼을 클릭하면 됩니다.

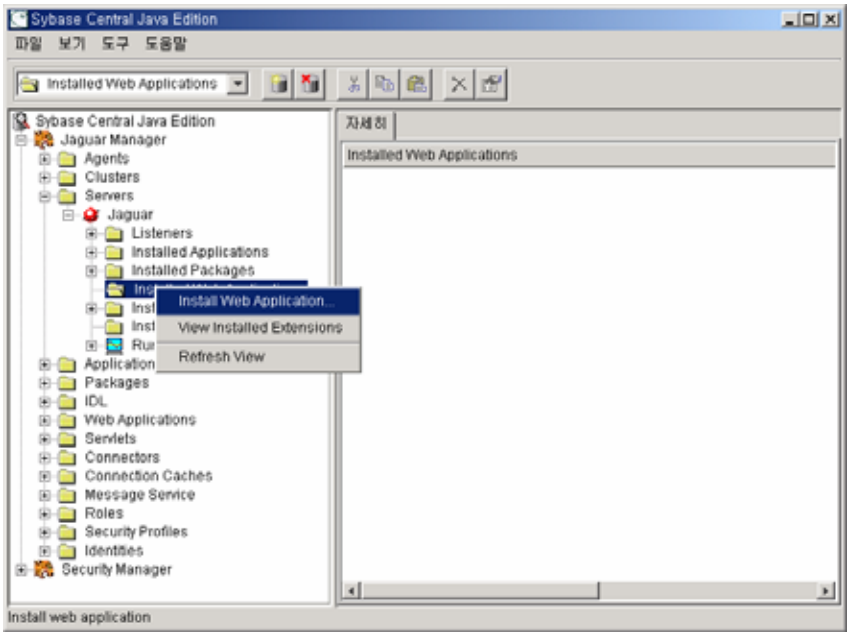
[Web Application]에 등록한 후 이제 실제 EAServer에 설정해 주어야 합니다.



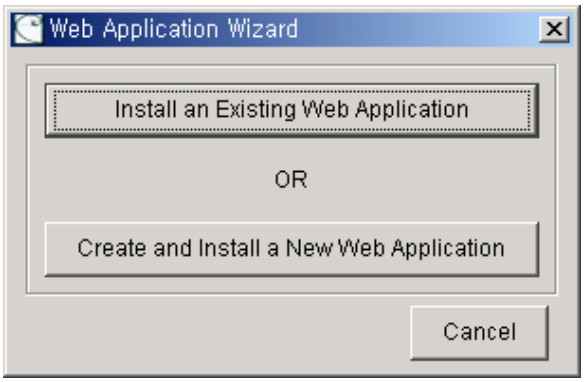
<EAServer에 OZServlet.war 파일이 성공적으로 전개된 화면>



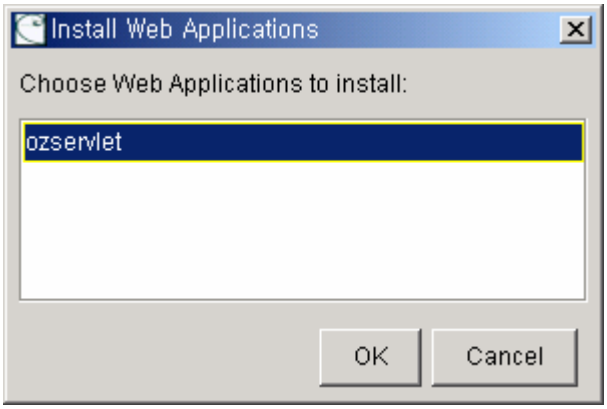
[Web Applications]에 등록한 후 이제 실제 EAServer에 등록시켜야 합니다.



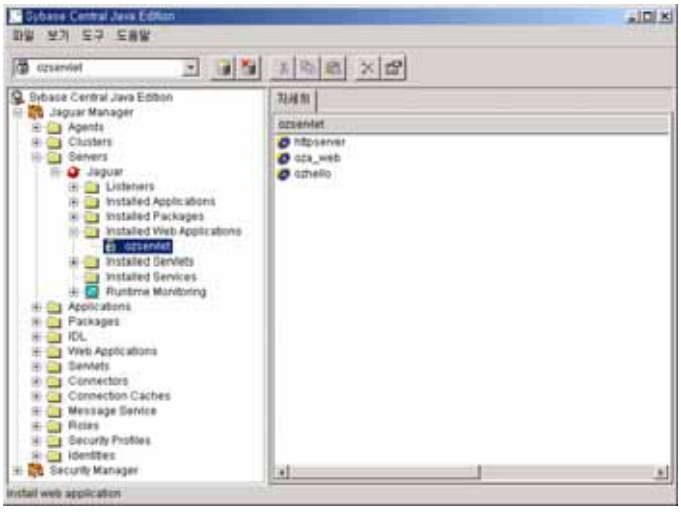
그림과 같이 [Installed Web Applicaiotns]을 선택한 후 마우스 오른쪽 버튼을 클릭하여 [Install Web Application]을 선택합니다.



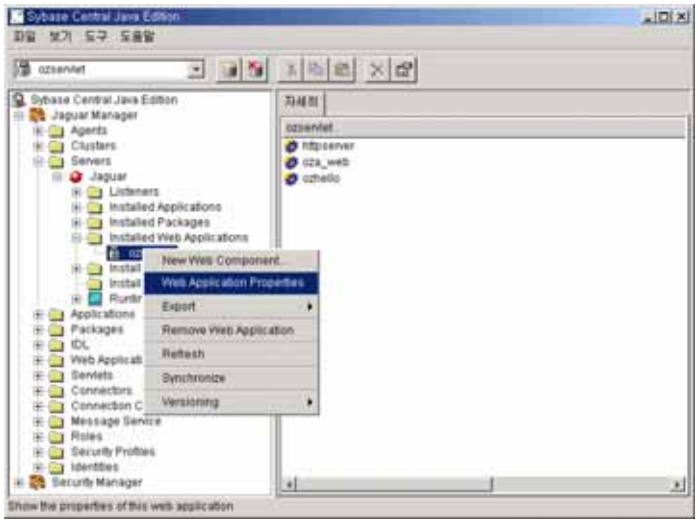
다음과 같은 화면이 나타나면 이미 [Web Application]에 등록되었으므로 위의 [Install an Existing Web Application] 버튼을 클릭합니다.



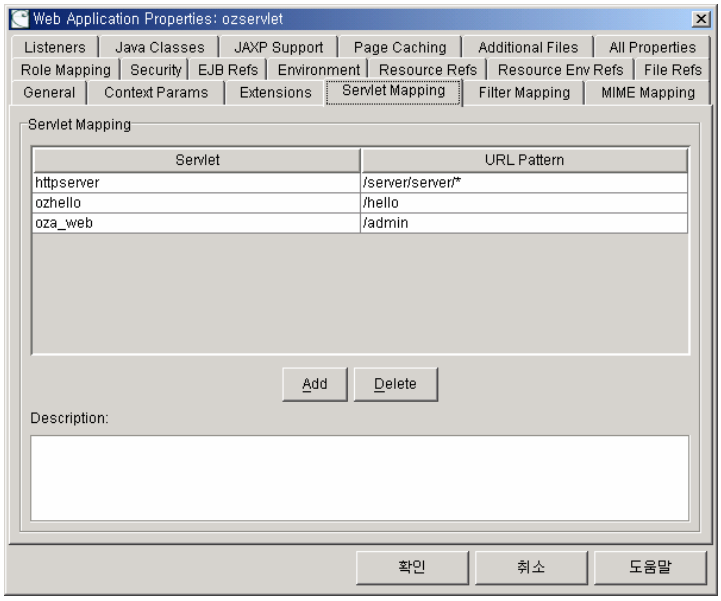
ozservlet를 선택한 후 [OK] 버튼을 클릭합니다.



위 화면과 같이 [OZServlet]이 등록된 것을 확인할 수 있습니다.



[ozservlet]에서 오른쪽 버튼을 클릭 후 그림과 같이 [Web Application Properties]를 선택합니다.



위의 화면이 나오면 [Servlet Mapping] 메뉴를 선택합니다. [URL Pattern] 부분에서 /server/server/*부분이 오즈의 앨리어스가 됩니다.

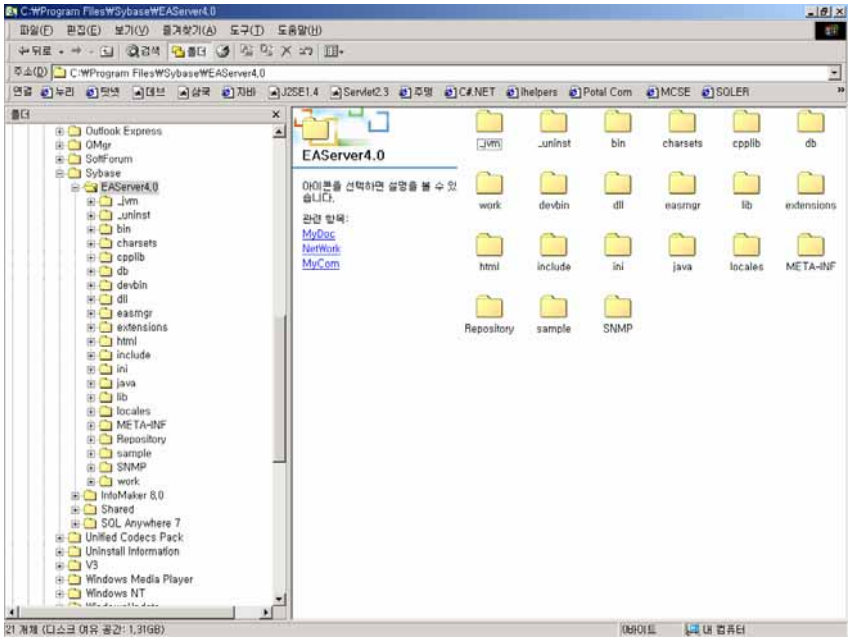
즉, http://localhost:8080/ozservlet/server/server과 같이 해주면 오즈의 서블릿 실행 화면을 볼 수 있습니다. 여기서 /server/server/* 부분은 수정 가능합니다.

오즈 서블릿을 실행하기 전에 반드시 {EAServer}\Repository\WebApplication\ozservlet \WEB-INF\classes\ozserver.properties 파일의 설정을 환경에 맞게 수정한 후 실행합니다.

※ 참고

예) /server/*

여기서 ozservlet이란 Alias 부분을 war 파일로 묶을 때의 ozservlet 파일명을 나타냅니다.



JEUS 3.0과의 연동

JEUS 3.0이 설치 폴더를 확인후 [$JEUS_HOME]\webhome\servlet_home\webapps에 오즈 폴더(oz)를 만들고 오즈 서블릿 파일을 해당 폴더에 복사합니다.

[$JEUS_HOME]\config\[$DEFAULT_SERVER]\[$DEFAULT_SERVER]_servlet_engine1 에 있는 container.xml 파일을 편집합니다.

```
<Context UserLogBufferSize="512"
ContextName="Examples"
EnableJSP="true"
AutoReload="true"
UserLog="default"
ContextPath="/examples"
UserLogValidDays="-1"
DocBase="examples" />
```

위의 내용을 복사하여 추가한 후 다음과 같이 수정합니다.

```
<Context UserLogBufferSize="512"
ContextName="oz"
EnableJSP="true"
AutoReload="true"
UserLog="default"
ContextPath="/oz"
UserLogValidDays="-1"
DocBase="oz" />
```

[$JEUS_HOME]\webhome\servlet_home\webapps\oz\WEB-INF\classes\ozserver.properties를 경로에 맞게 수정한 다음 JEUS30을 실행시킵니다.



<JEUS30 Server 실행화면>

jeusadmin을 실행시킵니다.

예) jeusadmin default_server (기본 서버)

[login]명은 administrator이고 [패스워드]는 해당관리자에게 문의합니다. 올바르게 접속된 경우 boot를 치면 서블릿 엔진이 실행됩니다.

```
D:₩Jeus30₩bin>jeusadmin chois73
[ErrorMsgManager] Message Manager is initialized
[JeusCommander] Jeus 3.0 Jeus Manager Controller
[JeusCommander] Login
>administrator
[JeusCommander] Password
>
[JeusCommander] [administrator] login successful
chois73>boot
[JeusCommander] chois73 boot done
```

<JeusAdmin 실행화면>

http://localhost:8088/oz/server로 설치 결과를 확인합니다.

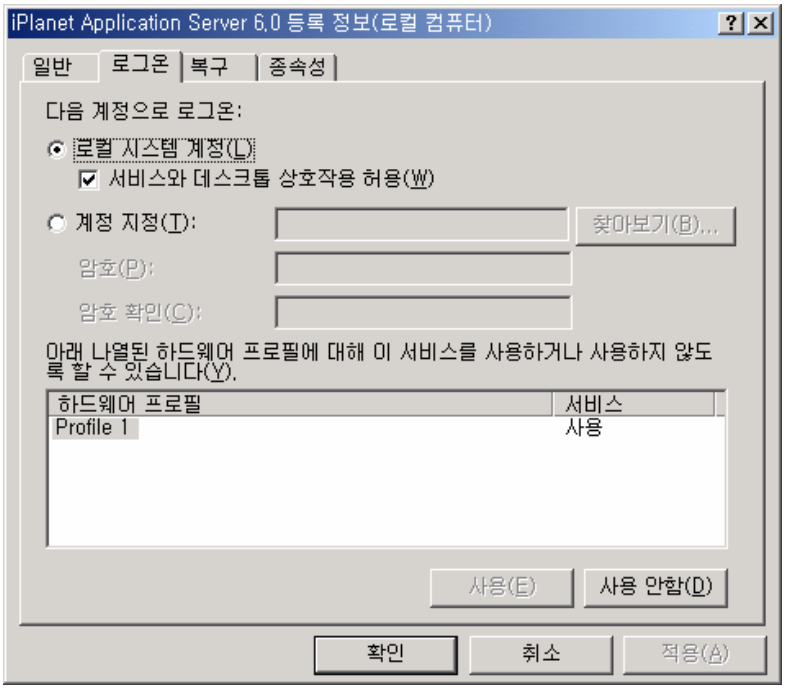
※ 참고

포트 변경은 container.xml의 설정값을 변경하면 됩니다.

iPlanet(6.5)와의 연동

iPlanet에 OZ Servlet 설치

우선 iPlanet의 로그를 보기 위해서는 win 환경에서 서비스에 등록되어 있는 iPanet Application Server 등록정보의 [서비스와 데스크톱 상호작용 허용]을 체크합니다.



ozservlet을 적당한 디렉토리에 설치합니다.

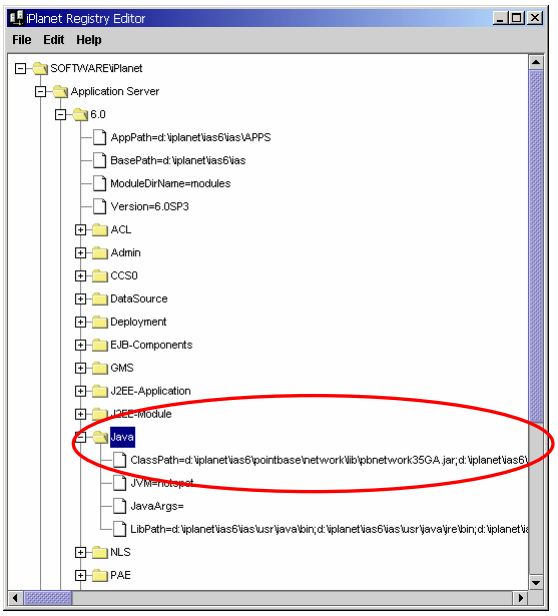
Classpath에 다음과 같은 파일들을 추가합니다.

■ **Windows**

D:\ozservlet\WEB-INF\lib\ozlib30.jar;D:\ozservlet\WEB-INF\lib\log4j.jar;+xml 파서 또는 DB용 jdbc를 오즈용으로 사용할 경우 별도로 추가합니다.

※ Windows용은 6개 실행파일(deploytool.bat, iascontrol.bat, iasdeploy.bat, j2eeappreg.bat, kregedit.bat, ksvradmin.bat, resreg.bat)에 각각 Classpath를 추가해야 합니다.

다음과 같이 kregedit.bat을 실행하여 Classpath를 추가합니다.



■ **Unix**

.profile에만 추가하고 콘솔을 다시 시작하면 Classpath가 모든 실행 파일에 적용됩니다.

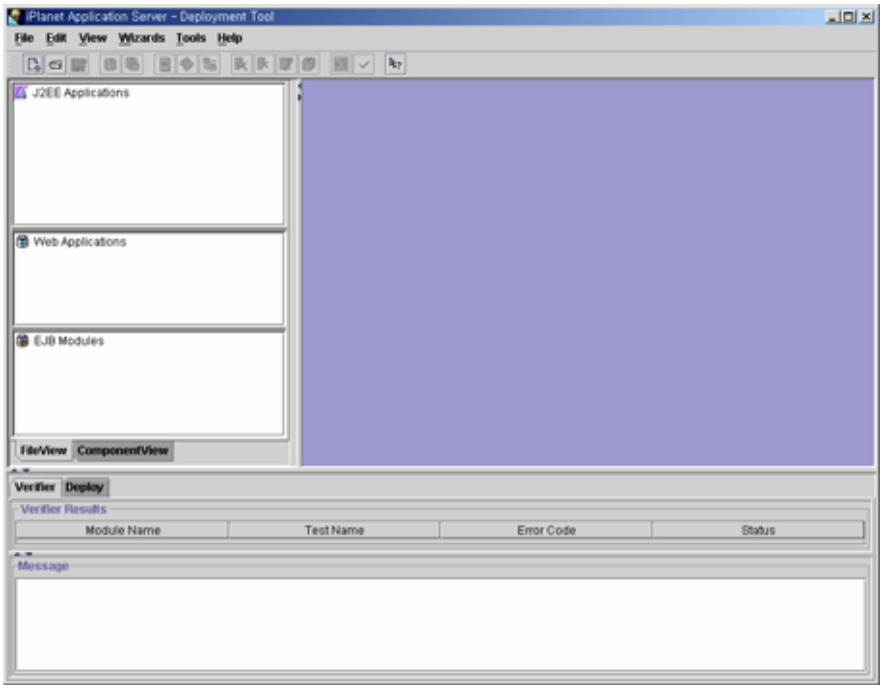
iPlanet에 oz.war를 Deploy하기 위해서는 다음과 같은 파일이 필요합니다.
(oz_Home)/WEB-INF/classes/ozserver.properties

ozlib30.jar 파일은 오즈 실행 서블릿 클래스가 있기 때문에 Deploy를 위해 압축을 풉니다.
(oz_Home)/WEB-INF/lib/ozlib30/ 아래 압축 풀린 모든 파일

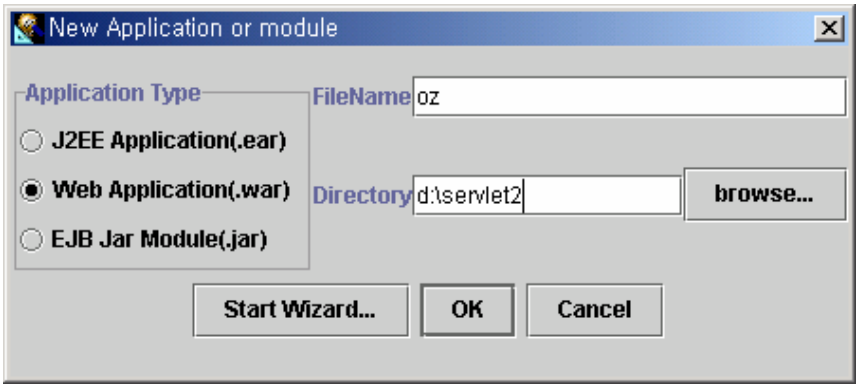
■ **iPlanet에 Deploy Tool 사용하기**

- Windows
  (ias_home)/bin/deploytool.bat
- Unix
  (ias_home)/bin/deploytool

1) iPanet Appication Server – Deployment Tool 실행

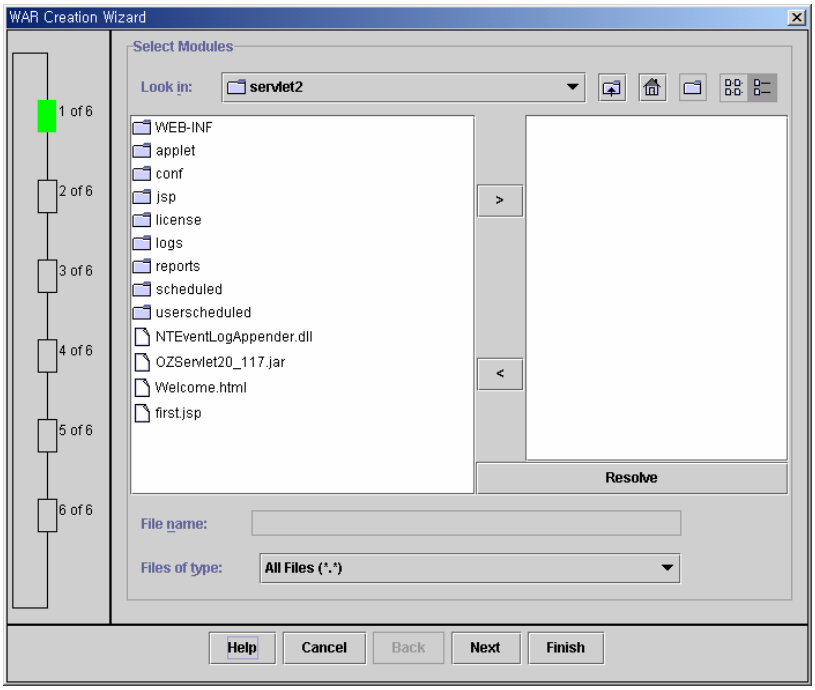


2) [File] – [New Application or module] 선택

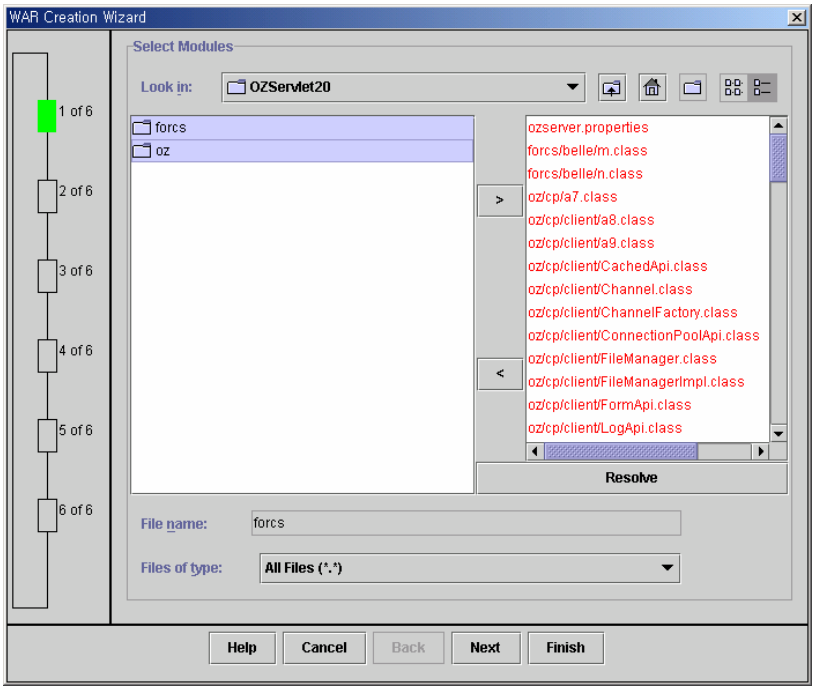


[Application Type]에서 Web Application(.war)를 선택하여 [FileName]에 'oz'를 입력하고 [Directory]에는 oz.war가 보여질 디렉토리를 설정합니다.

3) StartWizard



4) Deploy할 파일들 선택

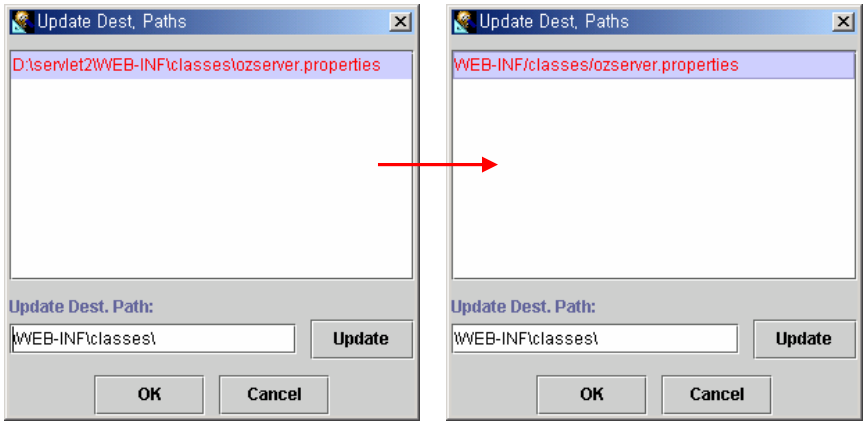


① ozserver.properties

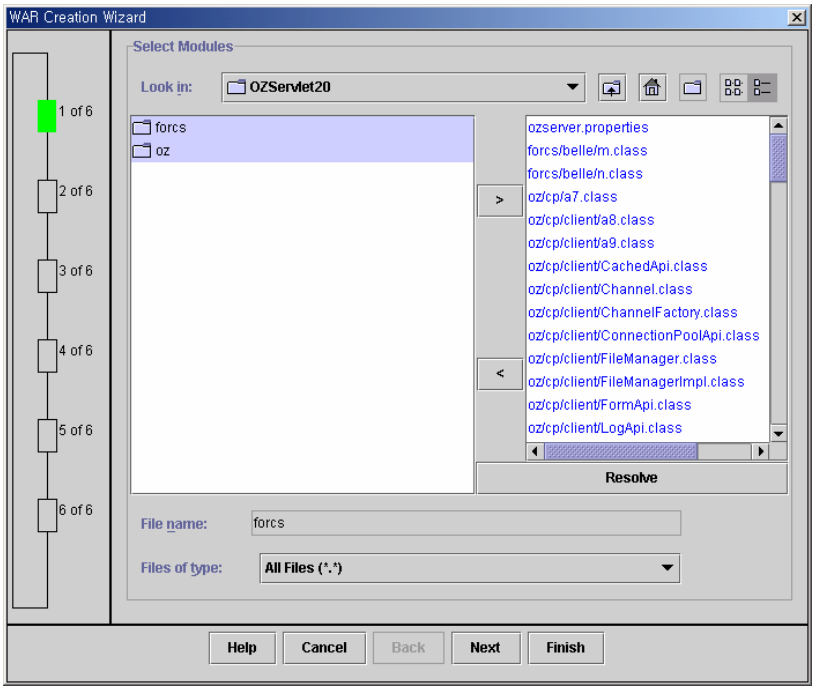
② (oz_home)/WEB-INF/lib/ozlib30/ 아래 압축을 푼 디렉토리들

Deploy할 파일을 선택하고 [Resolve]를 클릭합니다.

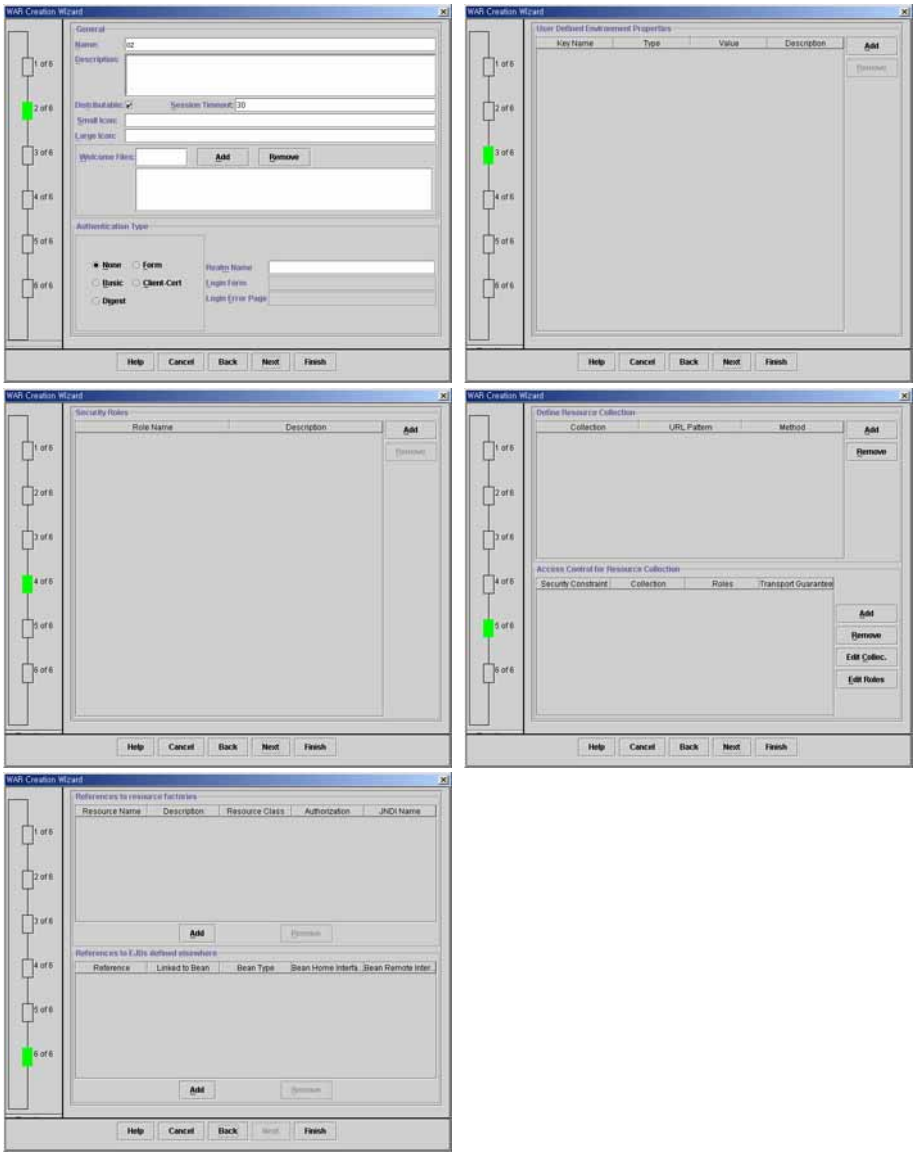
5) [Update Destination Paths]에서 [Update Dest. Path]를 WEB-INF/classes 로 바꾸고 [Update] 버튼을 클릭하면 아래 화면과 같이 ozserver.properties 경로가 바뀌게 됩니다.



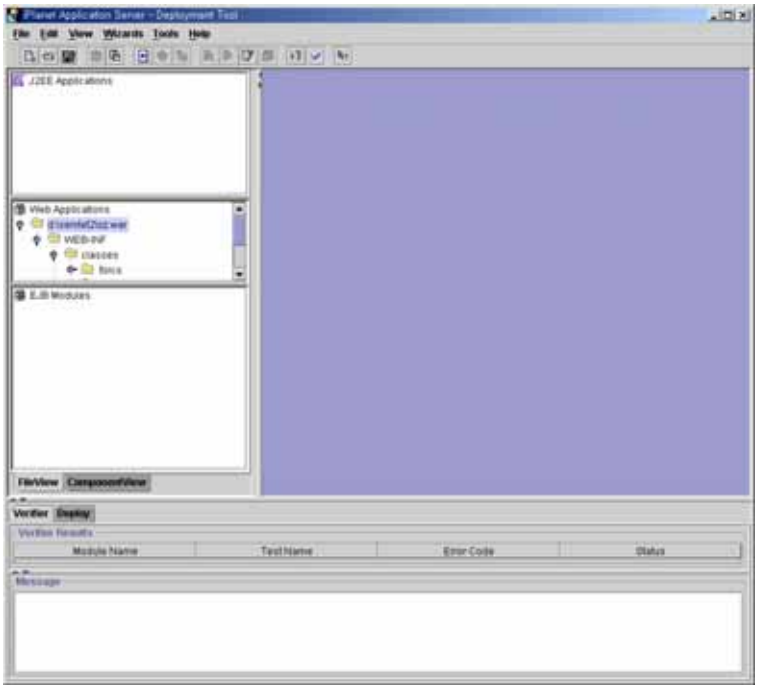
6) [OK] 버튼을 클릭하면 다음과 같이 파란색으로 글씨들이 바뀐 것을 확인할 수 있습니다.



7) 2 of 6부터는 default 값을 사용합니다.



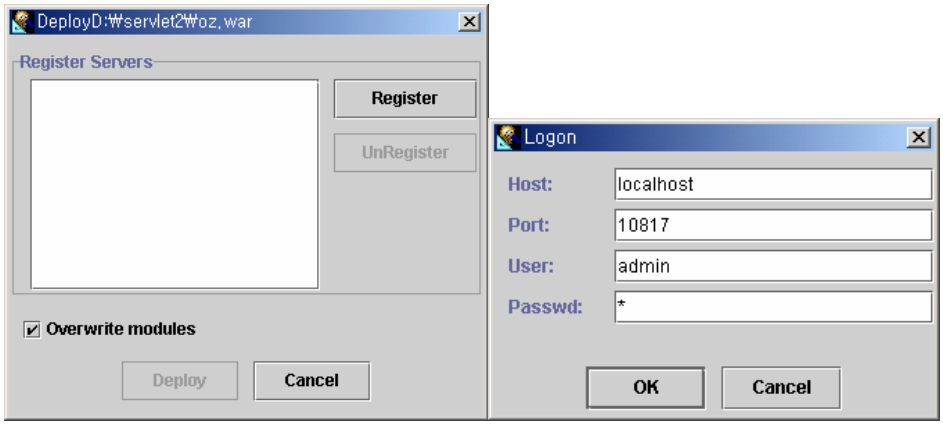
[Finish]를 클릭하면 다음과 같은 화면이 나타납니다.



8) oz.war의 [저장] 버튼을 클릭하여 저장한 후 파일이 닫히면 다시 엽니다.

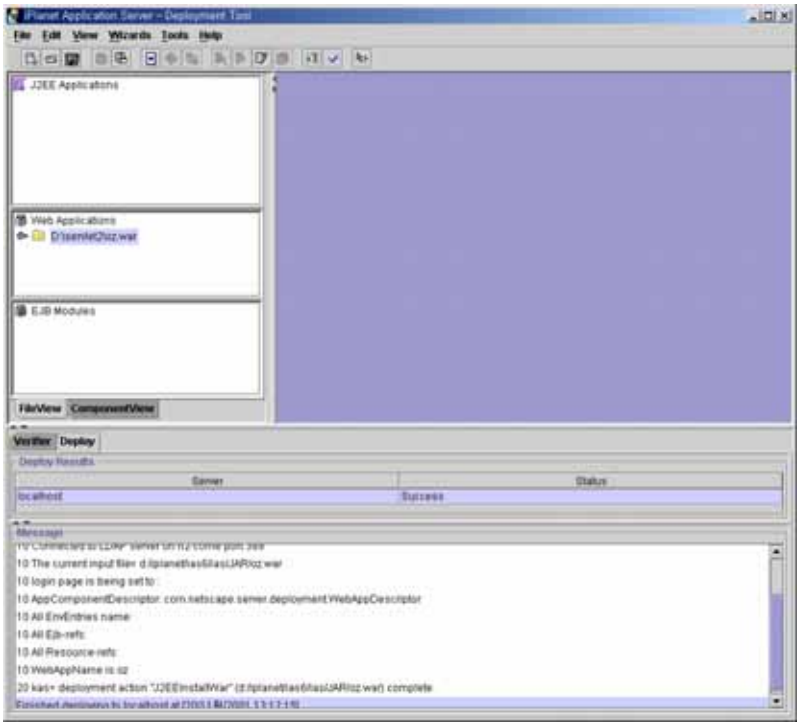
9) oz.war 파일을 열어 [Deploy]를 합니다.

[File] – [Deploy(ctrl + D)] 선택





[Overwrite modules]에 체크를 하고 [Deploy] 버튼을 클릭하면 다음과 같은 화면이 나타납니다.



Deploy Result success가 나타나면 설정이 완료됩니다.

10) 오즈 서블릿 설치 확인

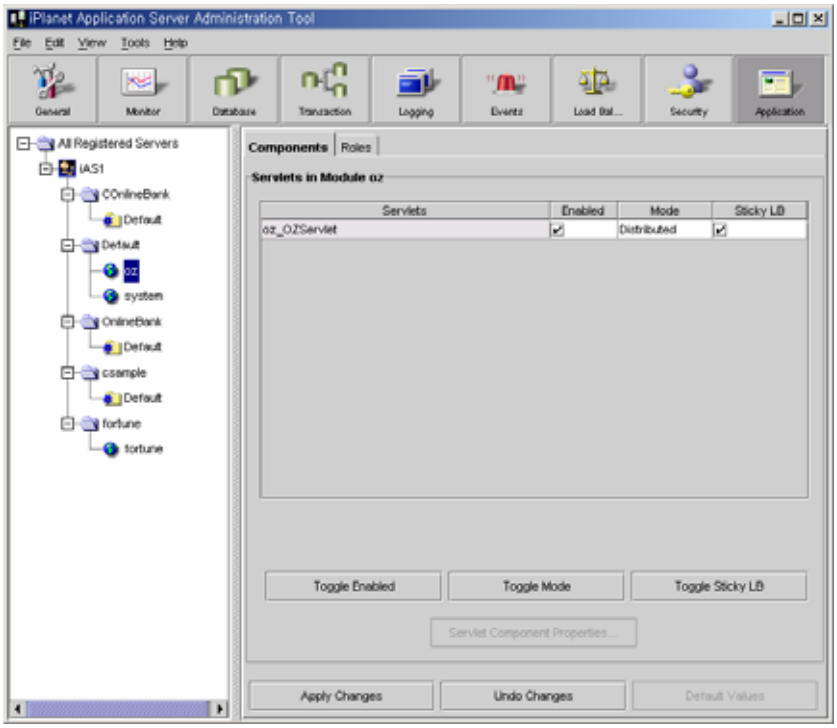
admintool을 실행합니다.

- Windows

  ksvradmin.bat

- Unix

  ksvradmin

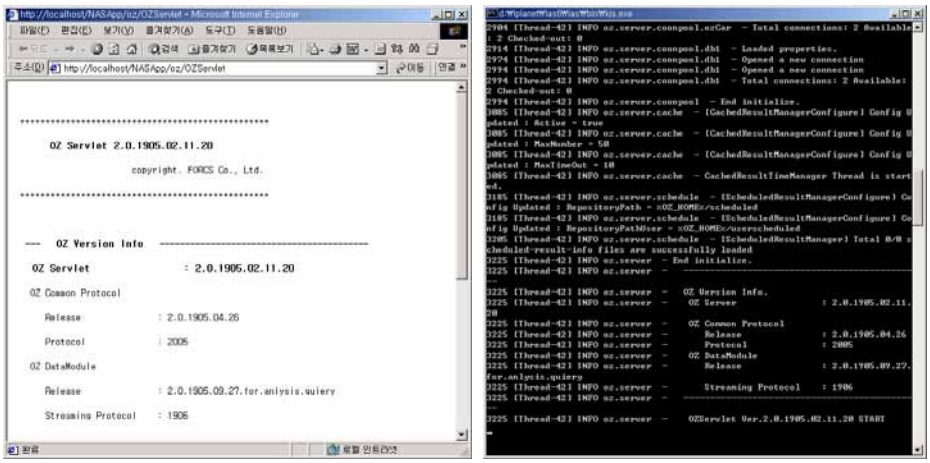


※ Application 탭 선택

다음과 같이 oz가 등록되어 있는 것을 확인할 수가 있습니다.

서블릿 엔진이 OZServlet으로 등록되었습니다.

웹상에서 확인할때는 http://127.0.0.1/NASApp/oz/server



※ iPlanet에 서블릿 설치 및 설정시 문제점 및 유의사항

1. **iPlanet은 서블릿의 이름이 대소문자 구분을 하지 않으므로 서블릿 등록시 유의해야 합니다.**
   이런 경우 (iPlanet_Home)/bin/kregedit.sh를 실행시켜 helloworld로 등록되어 있는 모든 모듈을 제거해야 정상 등록됩니다.

2. **iPlanet은 서블릿 표준스팩을 준수하고 있지 않으므로 서블릿 패키지 등록시 유의해야 합니다.**
   표준 스팩에는 웹모듈의 WEB-INF/web.xml이 웹 서비스 기본설정 파일이 되며, WEB-INF/classes가 Deploy시 Classpath로 잡혀서 해당 서블릿 클래스들이 자동로드됩니다. 또한 WEB-INF/lib 아래의 모든 jar 파일들은 자동으로 해당 웹모듈 로딩시에 함께 로딩되게 됩니다. 하지만, iPlanet은 애플리케이션 모듈이 아닌 웹 모듈임에도 불구하고 WEB-INF/ias-web.xml을 새로이 생성을 하며 WEB-INF/web.xml을 무시하게 됩니다. 그리고 WEB-INF/classes가 애플리케이션의 CLASSPATH로 잡히지 않아서 어떤 설정 파일도 로딩하지 못합니다. 또한 WEB-INF/lib가 다른 WAS와는 전혀 다르게 CLASSPATH로 잡혀 있어서 어떤 jar 파일도 인식못하고 단지 jar 파일의 압축을 풀어 놓아야 인식하게 됩니다.

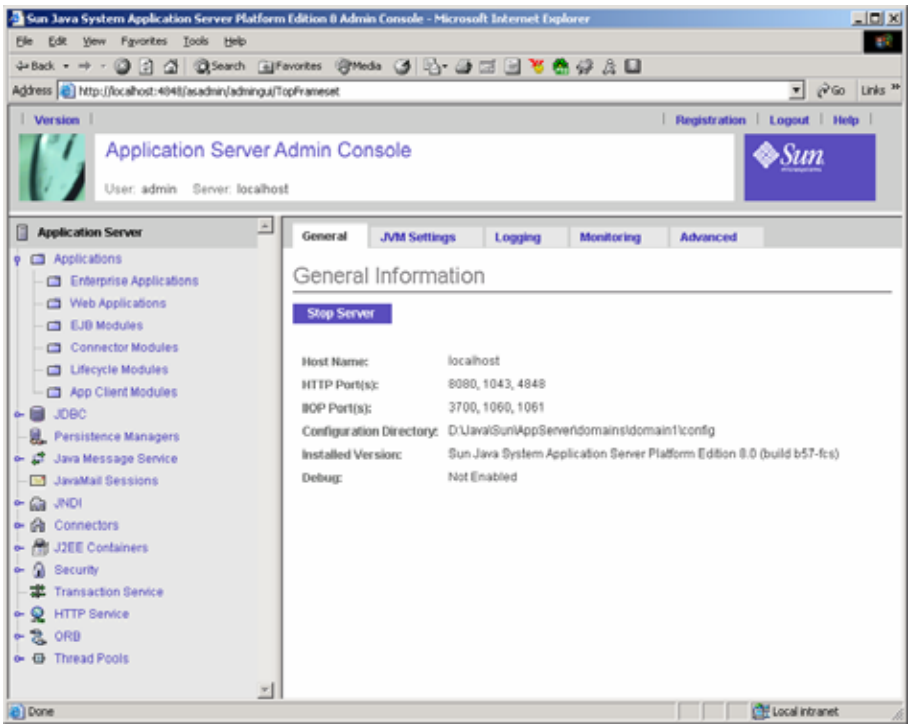
3. **위의 2번 문제로 인해 Deploy 하고자 하는 서블릿을 참조하는 패키지들은 모두 애플리케이션의 환경변수 설정을 쉘파일에 등록시켜야 합니다 .**
   /was/ias65/ias/env/iasenv.ksh이 파일에 원하는 패키지 파일과 설정(properties)파일을 CLASSPATH로 잡아서 서블릿을 로딩시켜야 합니다.

SUN J2EE와의 연동

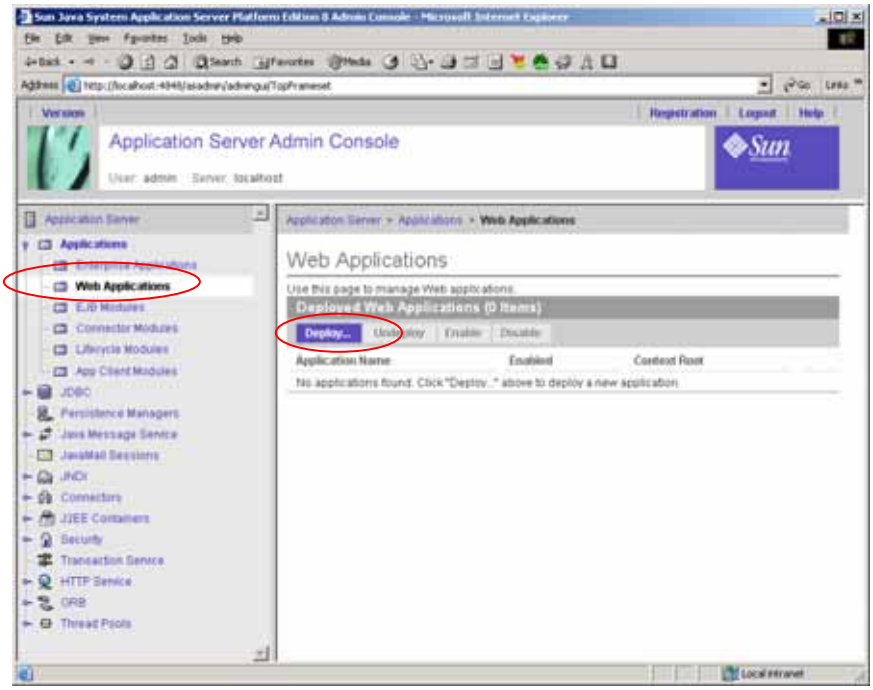
iPlanet의 오즈 서버(servlet 타입) 디렉토리를 서버의 설치 디렉토리에 복사합니다.

WEB-INF/classes/ozserver.properties와 conf/db.properties의 정보를 수정하고 오즈 서버에서 사용할 jdbc driver 파일을 WEB-INF/lib/에 복사합니다.

Admin Console을 실행시킵니다.(http://localhost:4848/asadmin/index.html)

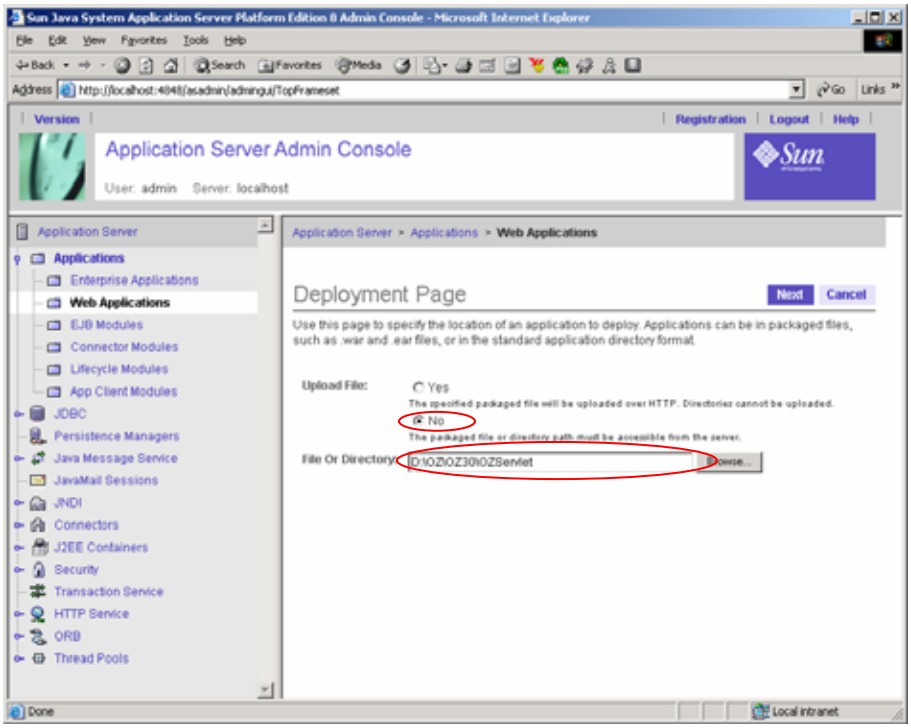


[Application Server] – [Web Applications] – [Web Applications]를 선택합니다.

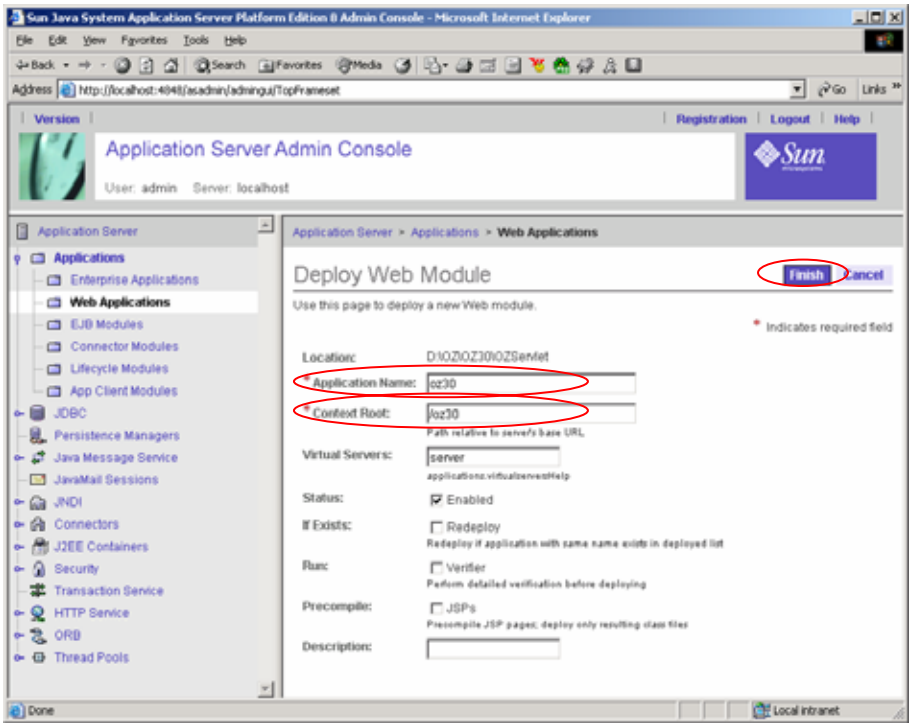


[Deploy] 버튼을 클릭하여 오즈 서버를 새로운 웹 애플리케이션으로 등록합니다.

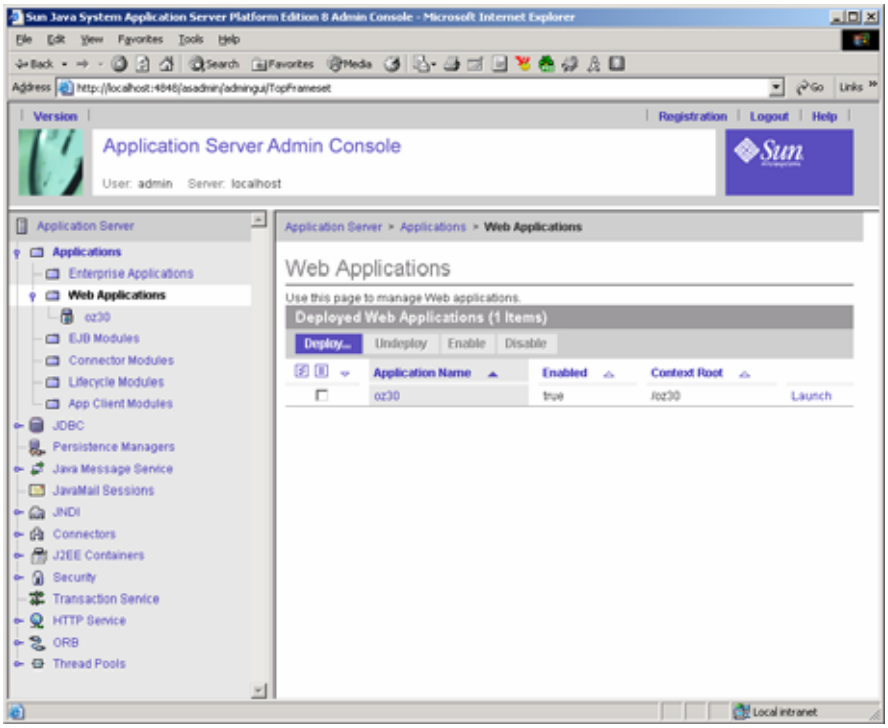
다음 화면에서 [Upload File]에 'No'를 선택하고, [File Or Directory]에 오즈 서버를 복사한 경로를 입력합니다.



[Nex] 버튼을 클릭하여 다음과 같은 화면이 나타나면 [Application Name]에 ‘oz30’을, [Context Root]에 ‘/oz30’을 입력합니다.



[Finish] 버튼을 클릭하면 다음과 같은 화면이 나타납니다.



설치한 오즈 서버가 정상적으로 동작하는지 웹 브라우저를 통해 확인합니다.
(http://localhost:8080/oz30/server)

