

하나씩 쉽게 따라 해보는 IBM WebSphere Application Server(WAS) v7 – 4

이정운 (juwlee@kr.ibm.com)

하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈 그 네 번째 이야기는 개발 그 두번째 이야기 입니다. 이전 강좌에서 RAD v7.5 를 이용해서 단순한 JSP, Servlet 을 개발하고 WAR 패키지로 묶은 후에 IBM WAS 에 배포하여 구동하는 것을 테스트로 확인해봤습니다. 이번 장에서는 좀 더 복잡하고 기업에서 많이 사용되는 EJB 프로그램을 개발해 보고 IBM WAS에 배포하여 운영하는 것을 진행해 보도록 하겠습니다. EJB 프로그램이란 Enterprise Java Bean 으로서 기업 환경에 적합하도록 만들어진 Java 프로그램으로 Container 가 보안, 트랜잭션, 동시성등 기업환경의 어플리케이션에서 필요한 여러가지 처리들을 할 수 있습니다. 좀 더 복잡하다고 말씀을 드렸긴하지만 EJB 규칙을 이해하고 이 형식을 잘 지키면 나머지는 일반 Java 소스 개발과 거의 동일하니 너무 긴장하지는 않으셔도 됩니다. 그럼 강좌를 진행해 볼까요.

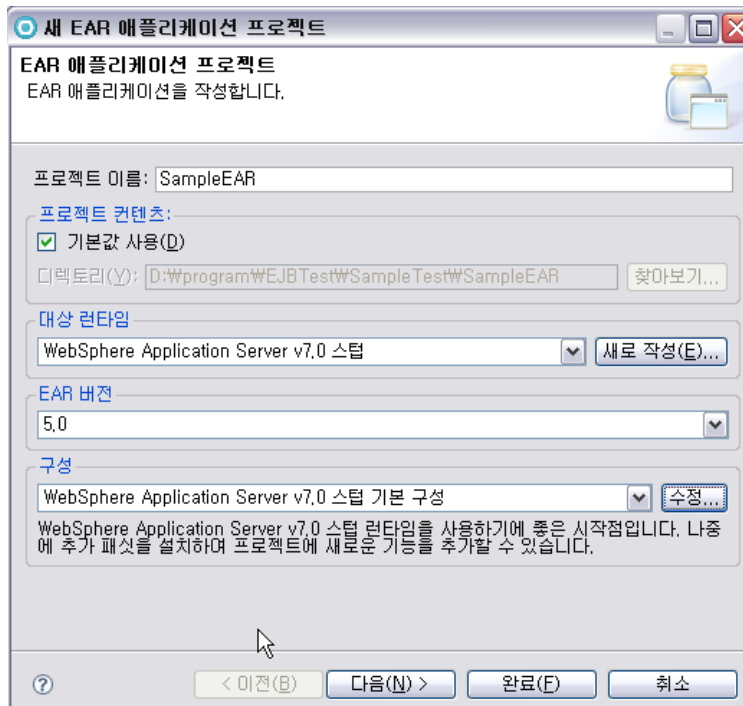
(현재 강좌의 EJB 개발에서는 EJB 3.0 을 기준으로 작성하였습니다.)

Part 1. EJB 개발

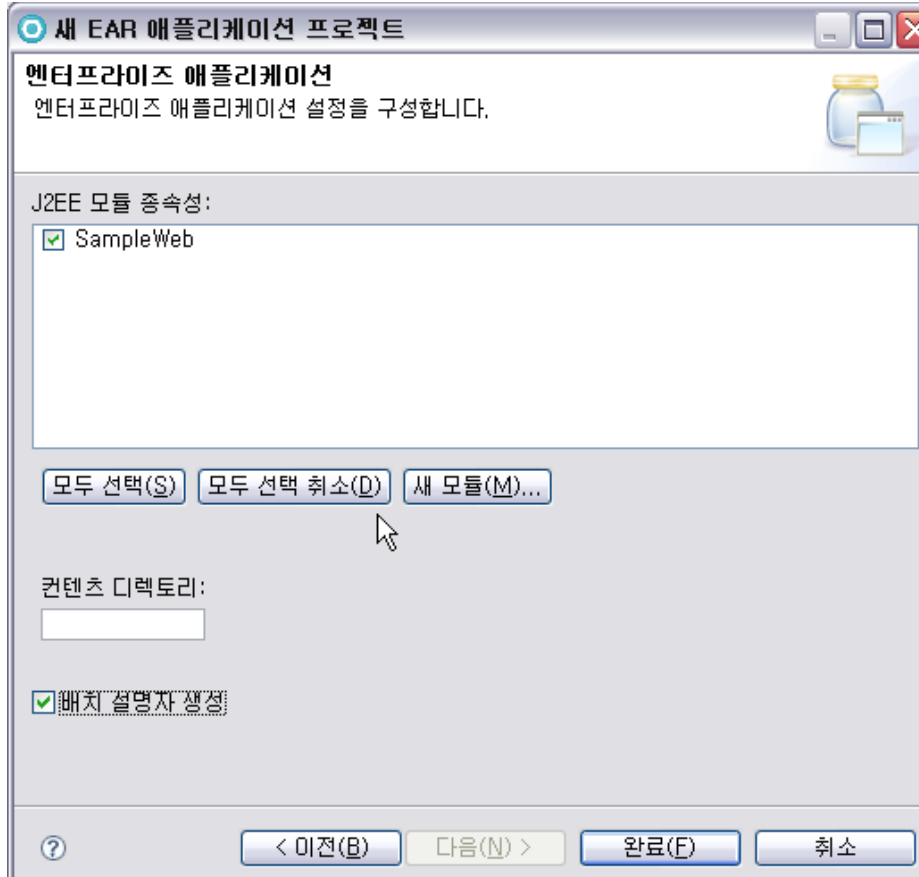
자, EJB 개발을 하기 위하여 RAD v7.5 를 실행합니다. (본 강좌는 이전 강좌에서 작성된 동적 웹프로젝트에 새롭게 EJB 프로젝트를 추가하는 방식으로 진행하도록 하겠습니다.) EJB를 수행하고 배포하기 위해서는 EAR 패키지 형식으로 프로젝트를 만들어야 합니다. 즉, 엔터프라이즈 어플리케이션 프로젝트를 만드셔야 합니다. 지금 이 강좌에서는 새롭게 엔터프라이즈 어플리케이션 프로젝트를 만드는 것이 아니라 이전 강좌에서 만들었던 동적 웹 프로젝트를 활용해서 작업하는 것으로 진행을 하도록 하겠습니다. 기존 프로젝트 화면을 연후 새롭게 엔터프라이즈 어플리케이션 프로젝트를 만들기 위하여 파일 > 새로작성 > 엔터프라이즈 어플리케이션 프로젝트를 선택합니다.



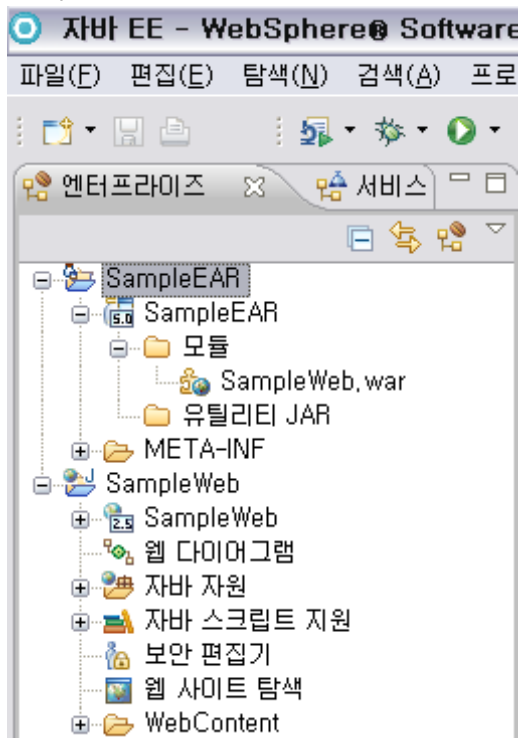
EAR 어플리케이션 프로젝트 마법사 화면이 나오면 프로젝트 이름과 사용할 EAR 버전등을 입력한 후 다음을 클릭합니다.



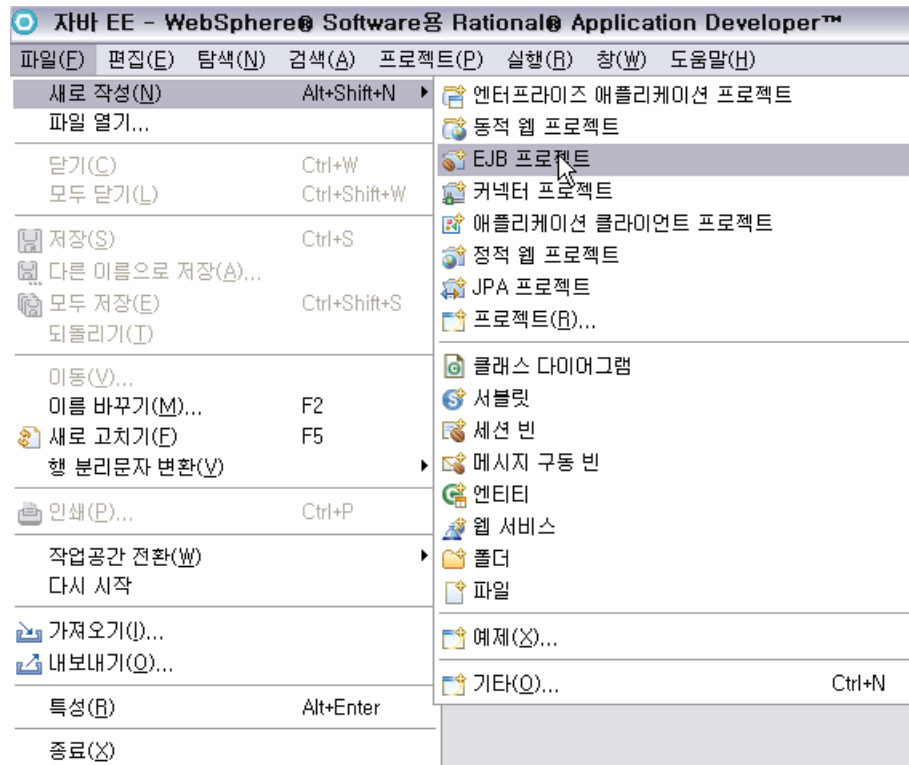
기존에 만들어진 SampleWeb 동적 웹 프로젝트를 추가한 후에 완료를 클릭합니다.



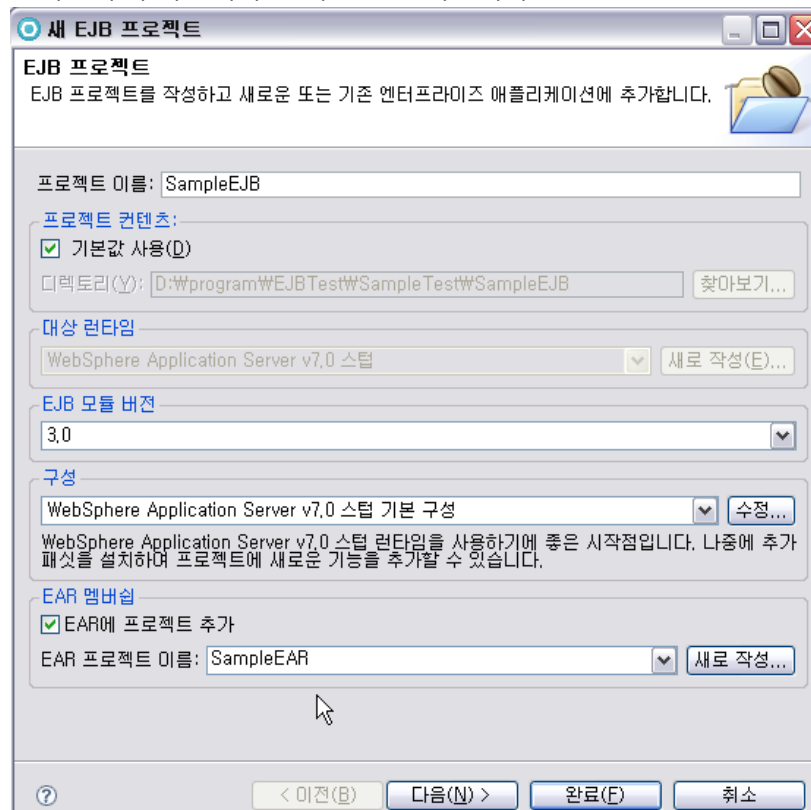
완료를 하시게 되면 하단의 그림처럼 EAR 프로젝트가 새롭게 생성되고 그 안의 모듈로 SampleWeb 동적 웹 프로젝트가 추가되어 있는 것을 확인하실 수 있습니다.



이제 EAR 어플리케이션 프로젝트라는 겹데기를 만들었으니 실제로 안에서 동작할 EJB 프로젝트를 작성하기 위하여 파일 > 새로작성 > EJB 프로젝트를 선택합니다.



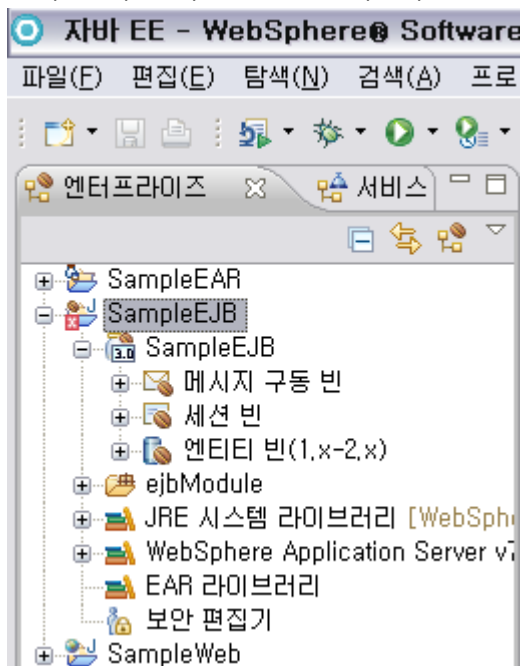
EJB 프로젝트 마법사에서 마찬가지로 이름과 EJB 버전을 선택하고 해당 EJB 프로젝트를 EAR 프로젝트에 추가선택하고 다음을 클릭합니다.



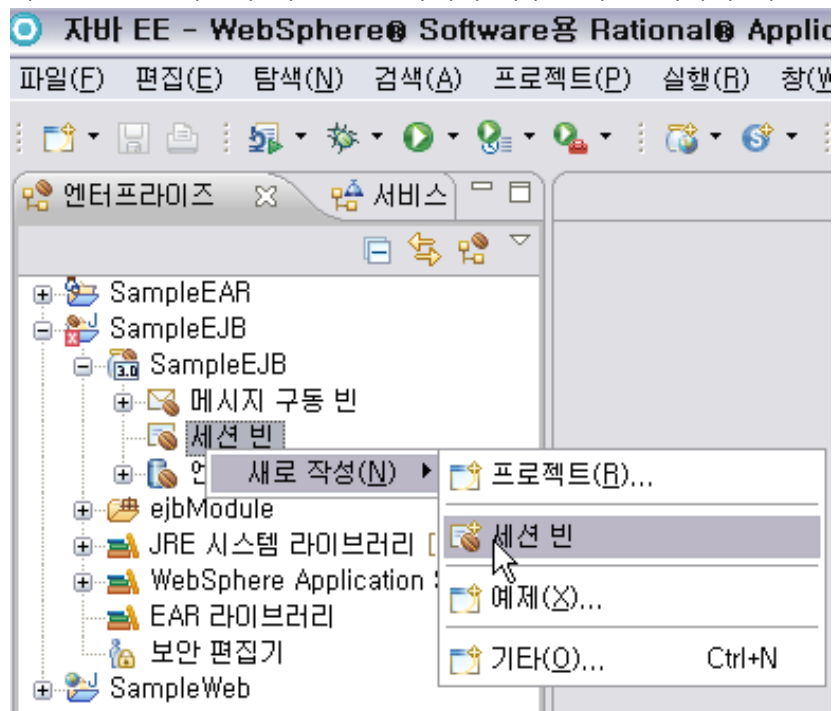
클라이언트 모듈 작성의 체크 박스에 체크되어 있는 것을 해제하고 완료를 누릅니다.



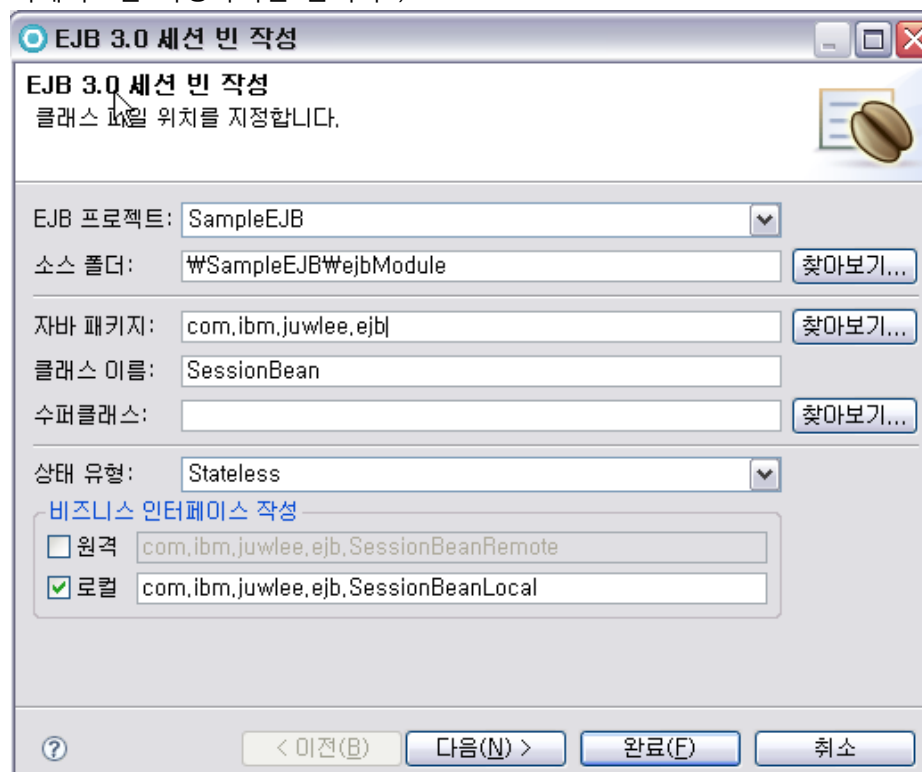
그러면, 하단 처럼 EJB 프로젝트가 생성된 것을 확인하실 수 있습니다.



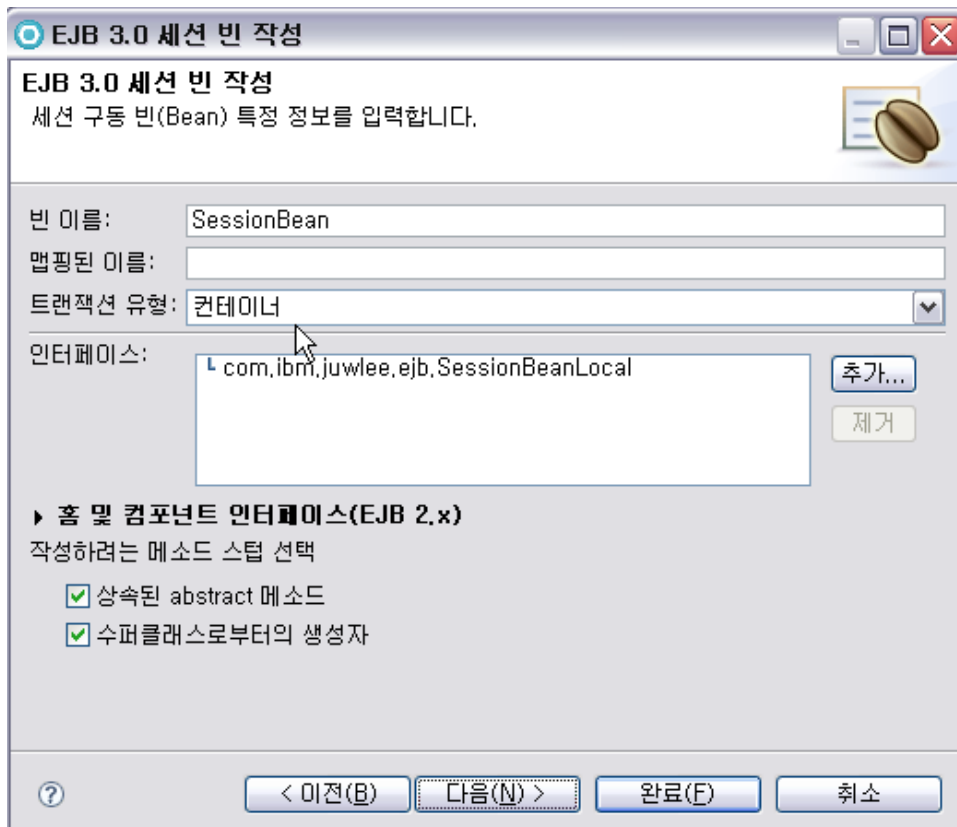
이제 실제로 EJB 프로젝트에서 호출되어 사용될 Session Bean 클래스를 만들어 보도록 하겠습니다. EJB 프로젝트의 세션 빈 항목에서 마우스 우 클릭하여 새로 작성 > 세션 빈을 선택합니다.



EJB 3.0 세션 빈 마법사가 나타나면 하단처럼 자바 패키지를 만들어주고 클래스 이름을 원하는 대로 입력하여 줍니다. 또한, 비즈니스 인터페이스를 작성하도록 체크박스를 체크합니다. (이번 강좌에서는 WAR 웹 프로젝트에서 같이 들어있는 EJB 프로젝트를 호출할 것이므로 로컬 인터페이스만 작성하시면 됩니다.)



Session Bean 의 인터페이스를 확인하고 완료를 눌러 생성을 마무리합니다.



EJB 3.0 세션 빈 작성
세션 구동 빈(Bean) 특정 정보를 입력합니다.

빈 이름: SessionBean

맵핑된 이름:

트랜잭션 유형: 컨테이너

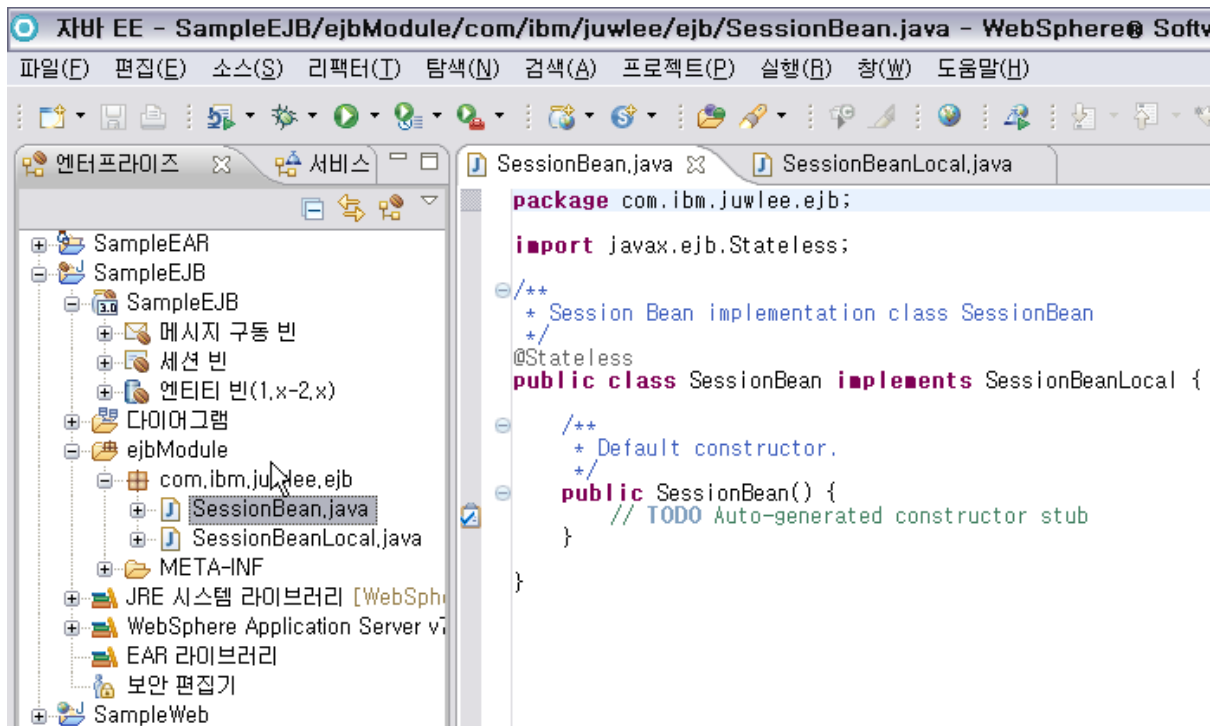
인터페이스: com.ibm.juwlee.ejb.SessionBeanLocal

홈 및 컴포넌트 인터페이스(EJB 2.x)
작성하려는 메소드 스텝 선택

- ☒ 상속된 abstract 메소드
- ☒ 수퍼클래스로부터의 생성자

< 이전(B) 다음(N) > 완료(F) 취소

Session Bean 을 생성하게 되면 아래 처럼 ejbModule 안에 실질적으로 java 패키지와 지정된 이름의 Java class 와 로컬 인터페이스가 생성된 것을 확인하실 수 있습니다.



자바 EE - SampleEJB/ejbModule/com.ibm.juwlee.ejb/SessionBean.java - WebSphere® Soft

파일(F) 편집(E) 소스(S) 리팩터(I) 탐색(N) 검색(A) 프로젝트(P) 실행(R) 창(W) 도움말(H)

엔터프라이즈 서비스 SessionBean.java SessionBeanLocal.java

```

package com.ibm.juwlee.ejb;

import javax.ejb.Stateless;

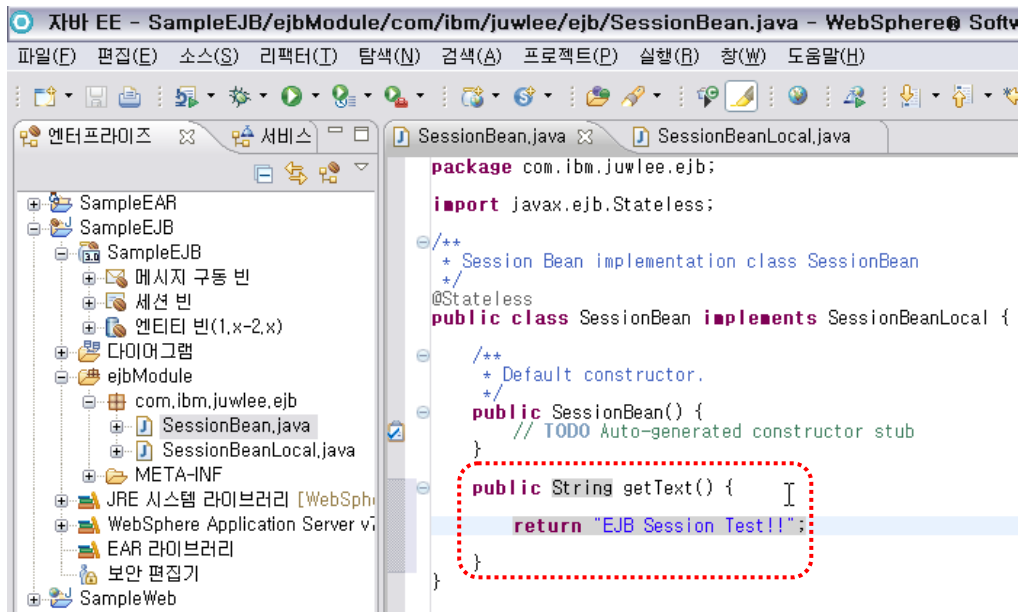
/**
 * Session Bean implementation class SessionBean
 */
@Stateless
public class SessionBean implements SessionBeanLocal {

    /**
     * Default constructor.
     */
    public SessionBean() {
        // TODO Auto-generated constructor stub
    }

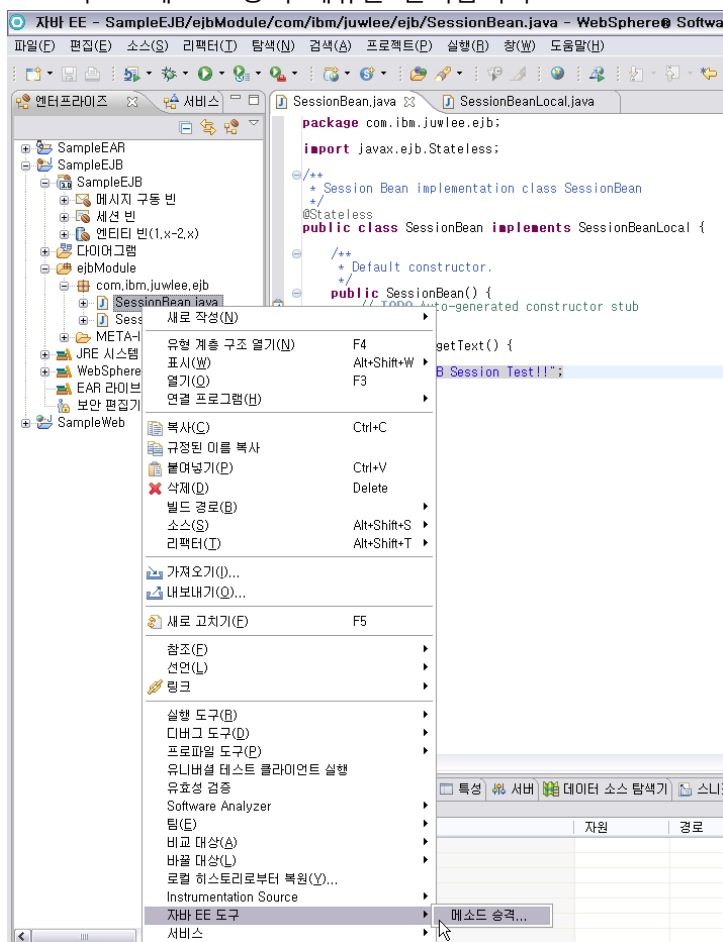
}

```

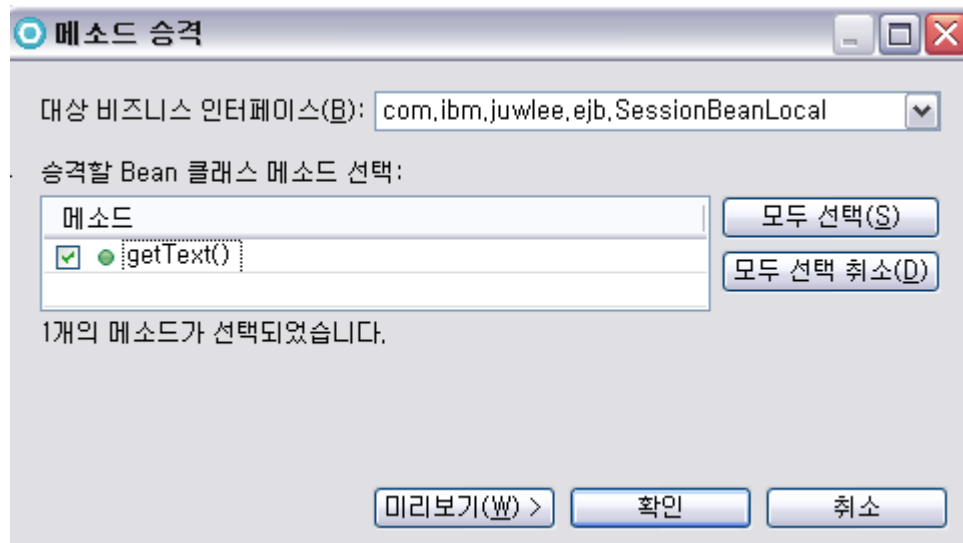
Session Bean 클래스에 하단에서 처럼 String 값을 반환하는 간단한 메소드를 만들어 줍니다.
(이 메소드는 향후 테스트 용도로 사용할 예정입니다.)



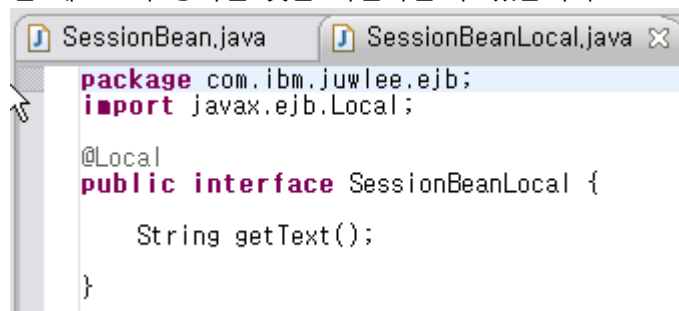
Session Bean 클래스에서 메소드를 만들었으면 EJB 규약에 따라 이 메소드는 반드시 로컬 인터페이스에도 등록되어야 합니다. 이를 위하여 Session Bean 클래스에서 마우스 우 클릭하여 자바 EE 도구 > 메소드 승격 메뉴를 선택합니다.



메소드 승격 마법사가 나타나면 방금 만들었던 `getText()` 메소드를 선택하고 확인을 누릅니다.



메소드 승격이 마무리 되면 하단에서 처럼 로컬 인터페이스에서 Session Bean 클래스에서 만들었던 메소드가 등록된 것을 확인하실 수 있습니다.



여기까지의 작업으로 간단한 샘플 EJB Session bean 클래스 작성을 성공적으로 수행하였습니다.

Part 2. EJB 호출 소스 작성

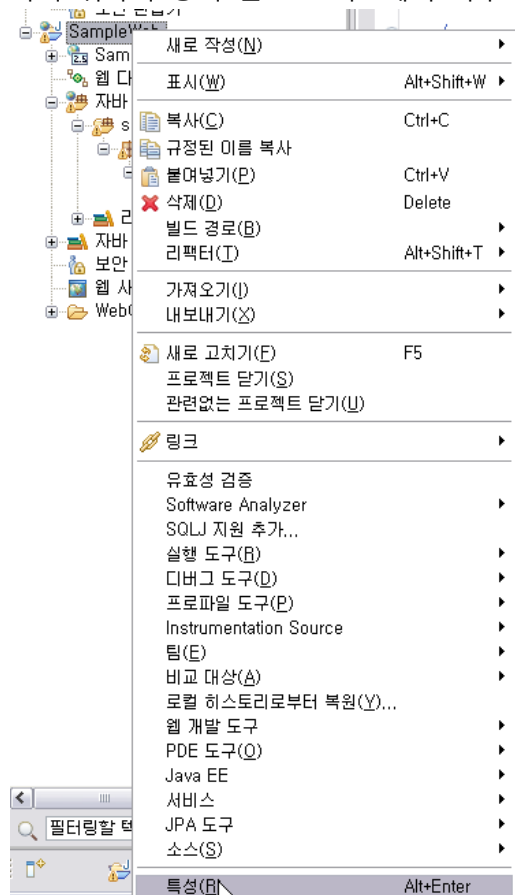
위의 단계까지 간단한 EJB Session Bean 클래스를 만들었으니 이제 실제로 EJB 를 look up 하고 호출하기 위한 소스를 WAR 동적 웹 프로젝트에 작성해보도록 하겠습니다.

(웹 브라우저로 결과를 바로 확인하기 위하여 동적 웹 프로젝트에 호출 소스를 넣는 것이고 원하는 곳이라면 어디서든 EJB 를 호출 가능합니다.)

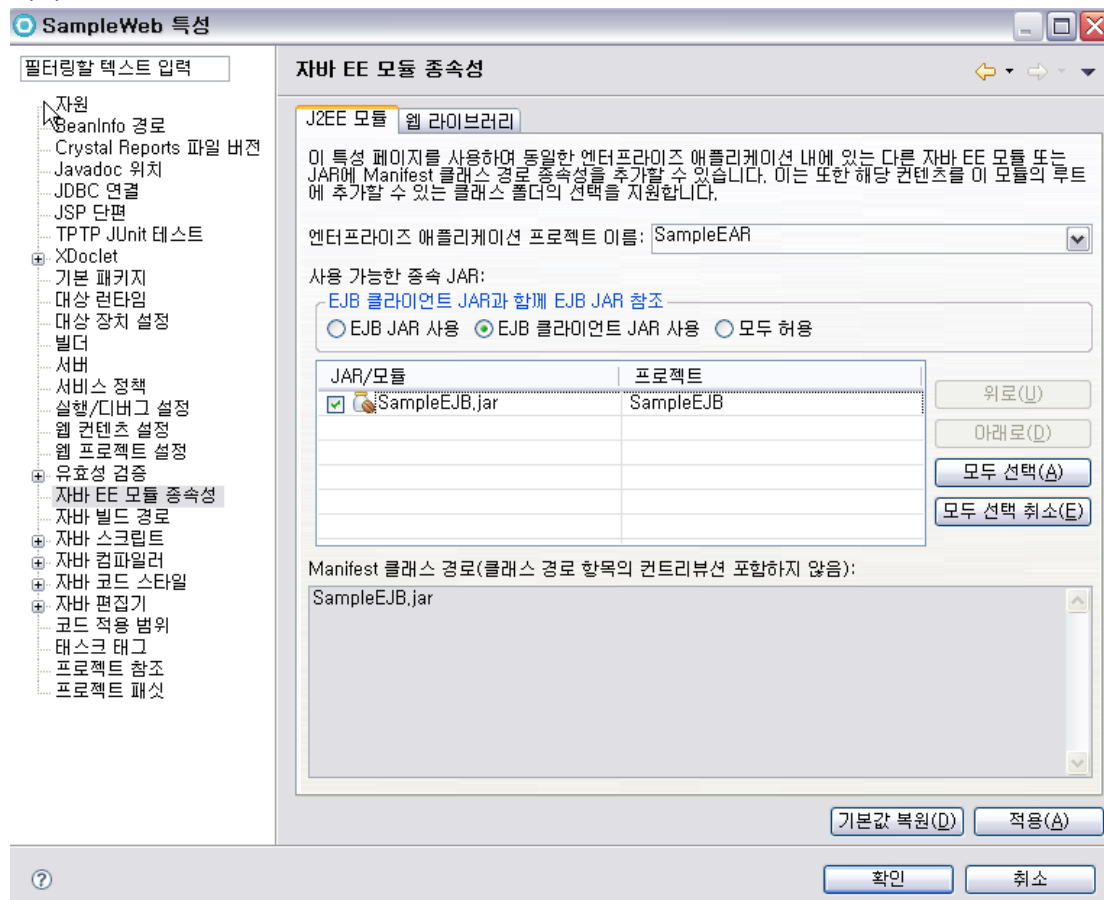
이전에 만들었던 SampleServlet 소스로 이동하여 EJB 호출부분을 아래 처럼 추가합니다.

```
3 import java.io.IOException;
4 import java.io.PrintWriter;
5
6 import javax.servlet.ServletException;
7 import javax.servlet.http.HttpServlet;
8 import javax.servlet.http.HttpServletRequest;
9 import javax.servlet.http.HttpServletResponse;
10
11 /**
12  * Servlet implementation class SampleServlet
13  */
14
15 public class SampleServlet extends HttpServlet {
16     private static final long serialVersionUID = 1L;
17
18     @EJB(name="ejb/SessionBean", beanInterface=SessionBeanLocal.class)
19     private SessionBeanLocal ejb3Session;
```

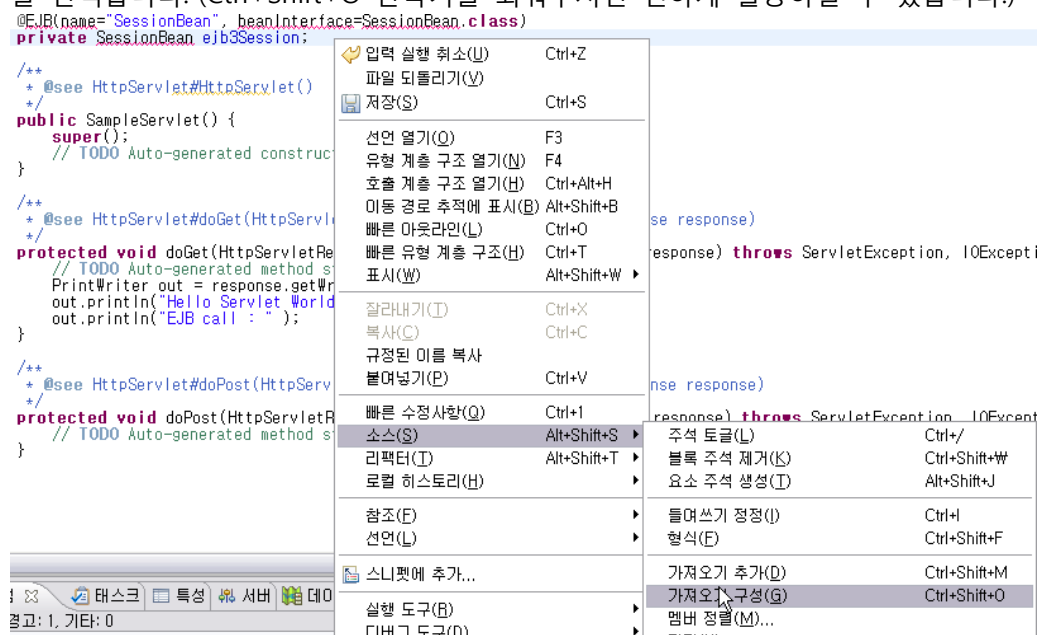
위의 소스만 넣으면 Error 표시가 나는데 그 이유는 EJB 프로젝트가 현재 동적 웹 프로젝트에서 참조할 수 없기 때문에 해당 class 를 인식 못해서 발생하는 것입니다. EJB 프로젝트 참조를 추가하기 위하여 동적 웹 프로젝트에서 마우스 우 클릭하여 특성을 선택합니다.



특성 창이 나오면 자바 EE 모듈 종속성을 선택하여 EJB 모듈의 체크박스를 체크하고 확인을 누릅니다.



참조가 된 EJB 모듈의 해당 class 를 인식하기 위해서 import 구문을 추가해 주어야 합니다. 이를 위하여 직접 소스에서 만들어주어도 되지만 소스에서 마우스 우 클릭하여 소스 > 가져오기 구성 을 선택합니다. (Ctrl+Shift+O 단축키를 외워두시면 편하게 활용하실 수 있습니다.)



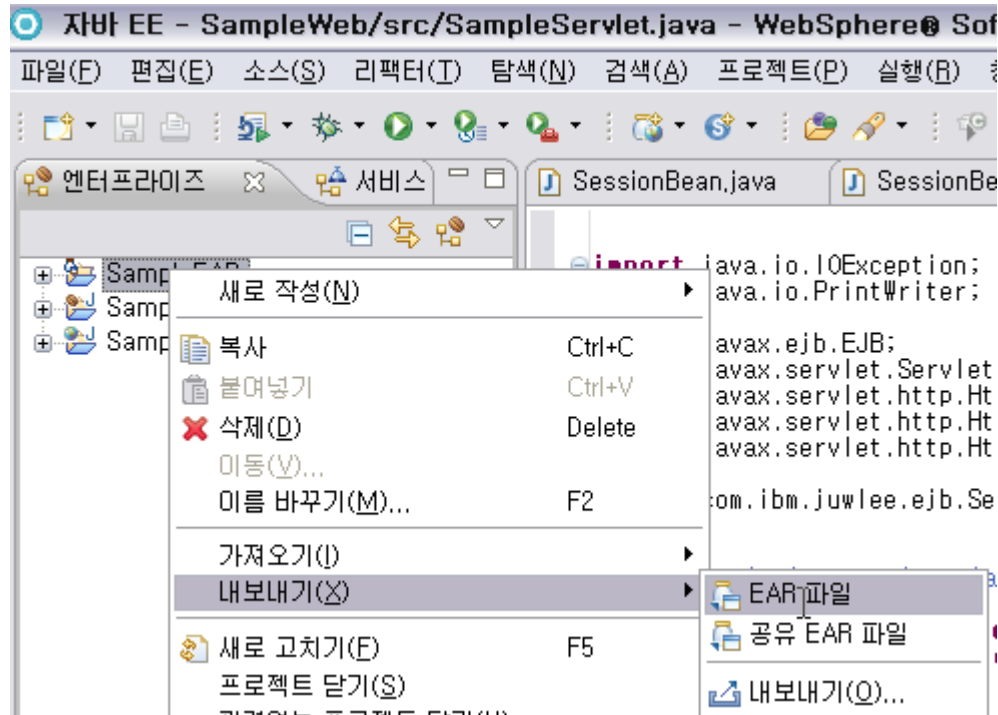
가져오기 구성을 선택하면 자동으로 필요한 클래스들이 import 되고 Error 가 사라진 것을 확인할 수 있습니다. Error 가 사라지면 실제 EJB 클래스의 동작을 확인하기 위하여 doGet() 메소드 안에서 화면으로 EJB Session Bean 클래스의 메소드를 호출하는 구문을 넣어 줍니다.

```
3 import java.io.IOException;
4 import java.io.PrintWriter;
5
6 import javax.ejb.EJB;
7 import javax.servlet.ServletException;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 import com.ibm.juwlee.ejb.SessionBeanLocal;
13
14 /**
15  * Servlet implementation class SampleServlet
16  */
17
18 public class SampleServlet extends HttpServlet {
19     private static final long serialVersionUID = 1L;
20
21     @EJB(name="ejb/SessionBean", beanInterface=SessionBeanLocal.class)
22     private SessionBeanLocal ejb3Session;
23
24     /**
25      * @see HttpServlet#HttpServlet()
26      */
27     public SampleServlet() {
28         super();
29         // TODO Auto-generated constructor stub
30     }
31
32     /**
33      * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
34      */
35     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36         // TODO Auto-generated method stub
37         PrintWriter out = response.getWriter();
38         out.println("Hello Servlet World!!!");
39         out.println("EJB call : " + ejb3Session.getText());
40     }
41 }
```

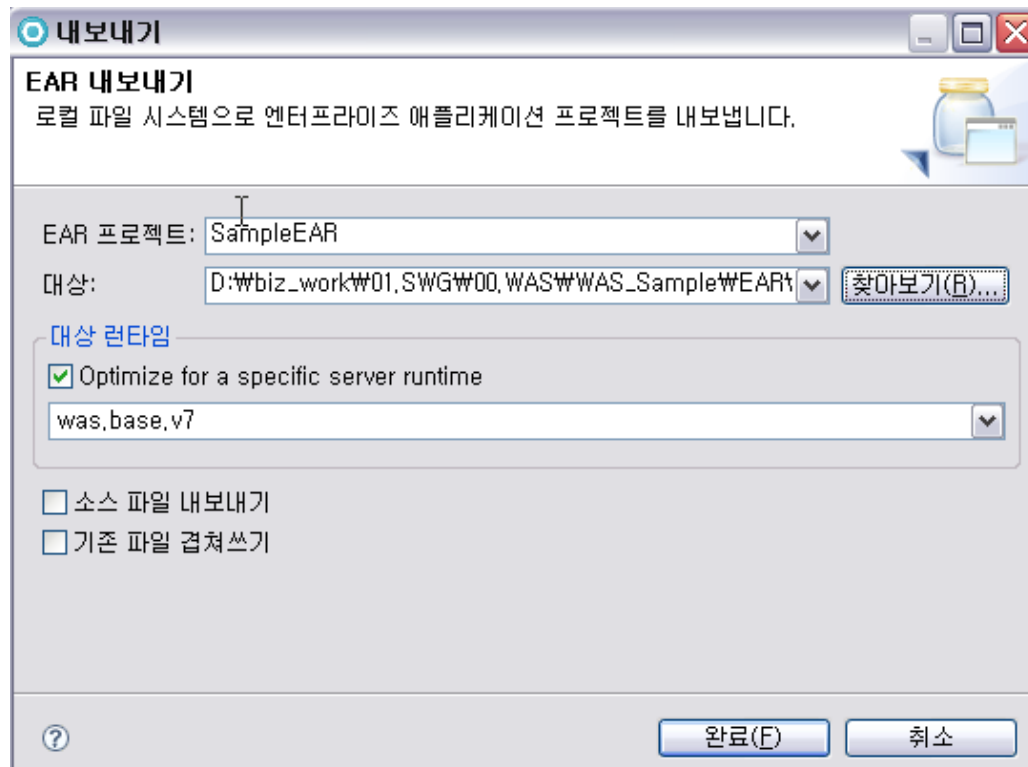
위와 같이 소스를 작성하시면 동적 웹 프로젝트에서 EJB 호출 부분의 소스 작성이 완성되었습니다.

Part 3. EJB 어플리케이션 배포

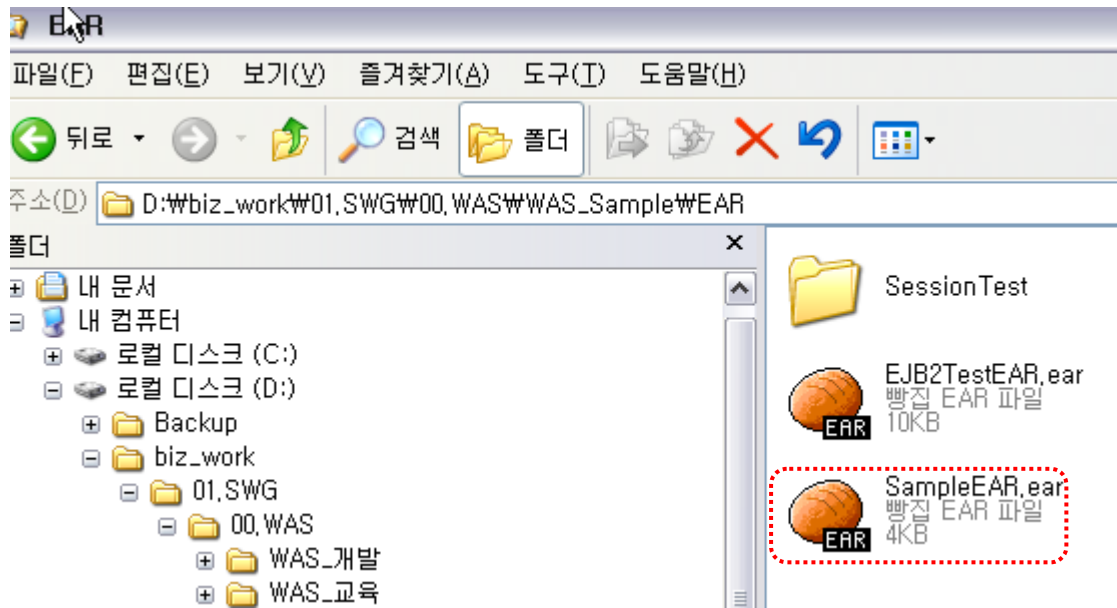
소스 작성이 완료되었으므로 이제 해당 프로젝트를 EAR 패키지 형태로 반출한 후에 IBM WAS 에 배포, 설치하는 작업을 진행하시면 됩니다. 본 작업은 이전 강좌의 배포와 동일 하므로 간략하게 만 설명드리겠습니다. 우선 EAR 엔터프라이즈 어플리케이션 프로젝트에서 마우스 우 클릭하여 내 보내기 > EAR 파일을 선택합니다.



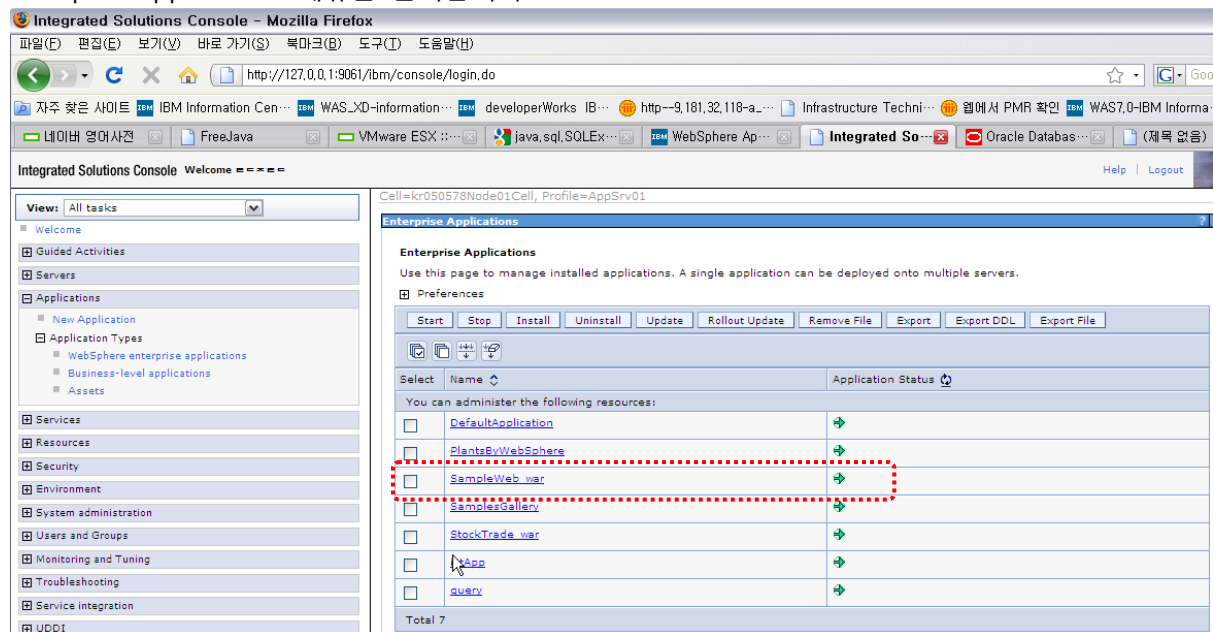
EAR 파일을 내보낼 위치를 선택하고 완료를 누르시면 EAR 파일의 반출이 완료됩니다.



EAR 파일의 반출이 완료되면 이전에 지정된 폴더에서 해당 EAR 파일을 확인하실 수 있습니다.



이제 실제 배포를 위하여 IBM WAS v7 의 관리콘솔로 들어가서 Applications > WebSphere enterprise applications 메뉴를 클릭합니다.



이전에 배포된 WAR 패키지 형태의 동적 웹 어플리케이션을 확인하실 수 있는데 이를 업데이트 해서 EAR 어플리케이션을 반영하도록 강화를 진행하겠습니다. 해당 어플리케이션을 선택하고 업데이트(Update) 버튼을 클릭합니다.

업데이트 마법사가 나타나면 찾아보기를 클릭하여 배포할 EAR 파일을 선택하고 다음을 누릅니다.

Specify the EAR, WAR, JAR, or SAR module to upload and install.

Application to be updated:
SampleWeb_war

Application update options

☒ Replace the entire application
Upload an enterprise archive (*.ear) to replace the entire installed application.

Specify the path to the replacement ear file.

☒ Local file system
Full path

☐ Remote file system
Full path

☐ Replace or add a single module
If the path to the new module matches an existing path to a module in the installed application, the new module replaces the existing module. If the path to the module does not exist in the installed application, the new module is added to the application.

☐ Replace or add a single file
If the path to the new file matches an existing path to a file in the installed application, the new file replaces the existing file. If the path to the file does not exist in the installed application, the new file is added to the application.

☐ Replace, add, or delete multiple files
Use a compressed file format such as .zip or .gzip. The compressed file is unzipped into the installed application directory. If the uploaded files exist in the application with the same paths and file names, the uploaded files replace the existing files. If the uploaded files do not exist, the files are added to the application. You can remove existing files from the installed application by specifying metadata in the compressed file.

업데이트 준비 마법사의 진행 방법을 선택하고 다음을 누릅니다.

Preparing for the application update

How do you want to install the application?

☒ Fast Path - Prompt only when additional information is required.

☐ Detailed - Show all installation options and parameters.

Specify bindings to use
merge new and existing bindings ▼

☒ Choose to generate default bindings and mappings

설치 마법사의 여러 옵션들을 확인하고 다음을 누릅니다.

Install New Application

Specify options for installing enterprise applications and modules.

Step 1: Select installation options

[Step 2: Map modules to servers](#)

[Step 3: Summary](#)

Select installation options

Specify the various options that are available to prepare and install your application.

☐ Precompile JavaServer Pages files

Directory to install application
`$(APP_INSTALL_ROOT)/i`

☒ Distribute application

☐ Use Binary Configuration

☐ Deploy enterprise beans

☒ Create MBeans for resources

☐ Override class reloading settings for Web and EJB modules

Reload interval in seconds

☐ Deploy Web services

Validate Input off/warn/fail
`warn`

☐ Process embedded configuration

File Permission

Allow all files to be read but not written to
Allow executables to execute
Allow HTML and image files to be read by everyone

`*.*dll=755#.*.*.so=755#.*.*.a=755#.*.*.l=755`

Application Build ID
`Unknown`

☐ Allow dispatching includes to remote resources

☐ Allow servicing includes from remote resources

Asynchronous Request Dispatch Type
`Disabled`

☐ Allow EJB reference targets to resolve automatically

[Next](#) [Cancel](#)

서버 맵핑정보를 확인하고 다음을 누릅니다 .이때, 지난 WAR 프로젝트와는 다르게 SampleEJB.jar 가 추가되어 있는 것을 확인하실 수 있습니다.

Specify options for installing enterprise applications and modules.

[Step 1: Select installation options](#)

→ Step 2: Map modules to servers

[Step 3: Summary](#)

Map modules to servers

Specify targets such as application servers or clusters of application servers where you want to install the modules that are contained in your application. Modules can be installed on the same application server or dispersed among several application servers. Also, specify the Web servers as targets that serve as routers for requests to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated, based on the applications that are routed through.

Clusters and servers:
`WebSphere:cell=kr050578Node01Cell,node=kr050578Node01,server=server1` [Apply](#)

Select	Module	URI	Server
<input type="checkbox"/>	SampleEJB.jar	SampleEJB.jar,META-INF/ejb-jar.xml	WebSphere:cell=kr050578Node01Cell,node=kr050578Node01,server=server1
<input type="checkbox"/>	SampleWeb	SampleWeb.war,WEB-INF/web.xml	WebSphere:cell=kr050578Node01Cell,node=kr050578Node01,server=server1

[Previous](#) [Next](#) [Cancel](#)

마지막으로 요약정보를 확인하고 완료를 누릅니다.

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2 Map modules to servers

→ Step 3: Summary

Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	\$(APP_INSTALL_ROOT)/kr050578Node0
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Create MBeans for resources	Yes
Override class reloading settings for Web and EJB modules	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	No
File Permission	.*\,dll=755#.*\,so=755#.*\,a=755#.*\,sl=755
Application Build ID	Unknown
Allow dispatching includes to remote resources	No
Allow servicing includes from remote resources	No
Asynchronous Request Dispatch Type	Disabled
Allow EJB reference targets to resolve automatically	No
Application name	SampleWeb_war
Cell/Node/Server	Click here

Previous

Finish

Cancel

완료를 누르면 실제 배포작업이 진행되는 것을 확인하실 수 있고 배포가 완료되면 저장(Save) 를 누릅니다. 이렇게 되면 EAR 어플리케이션 배포 작업이 마무리 되었습니다.

Updating...

If there are enterprise beans in the application, the EJB deployment process can take several minutes. Do not save the configuration until the process completes.

Check the SystemOut.log on the deployment manager or server where the application is deployed for specific information about the EJB deployment process as it occurs.

ADMA5017: Installation of SampleWeb_war started.

ADMA5104: The server index entry for WebSphere:cell=kr050578Node01Cell,node=kr050578Node01 is updated successfully.

ADMA5102: The configuration data for SampleWeb_war from the configuration repository is deleted successfully.

ADMA5011: The cleanup of the temp directory for application SampleWeb_war is complete.

ADMA5106: Application SampleWeb_war uninstalled successfully.

ADMA5016: Installation of SampleWeb_war started.

ADMA5067: Resource validation for application SampleWeb_war completed successfully.

ADMA5058: Application and module versions are validated with versions of deployment targets.

ADMA5005: The application SampleWeb_war is configured in the WebSphere Application Server repository.

ADMA5053: The library references for the installed optional package are created.

ADMA5005: The application SampleWeb_war is configured in the WebSphere Application Server repository.

ADMA5001: The application binaries are saved in D:\IBM\WebSphere7\AppServer\profiles\AppSrv01\wstempl-859749752\workspace\cells\kr050578Node01Cell\applications\SampleWeb_

ADMA5005: The application SampleWeb_war is configured in the WebSphere Application Server repository.

SECJ0400: Successfully updated the application SampleWeb_war with the appContextIDForSecurity information.

ADMA5005: The application SampleWeb_war is configured in the WebSphere Application Server repository.

ADMA5113: Activation plan created successfully.

ADMA5011: The cleanup of the temp directory for application SampleWeb_war is complete.

ADMA5013: Application SampleWeb_war installed successfully.

Application SampleWeb_war installed successfully.

To start the application, first save changes to the master configuration.

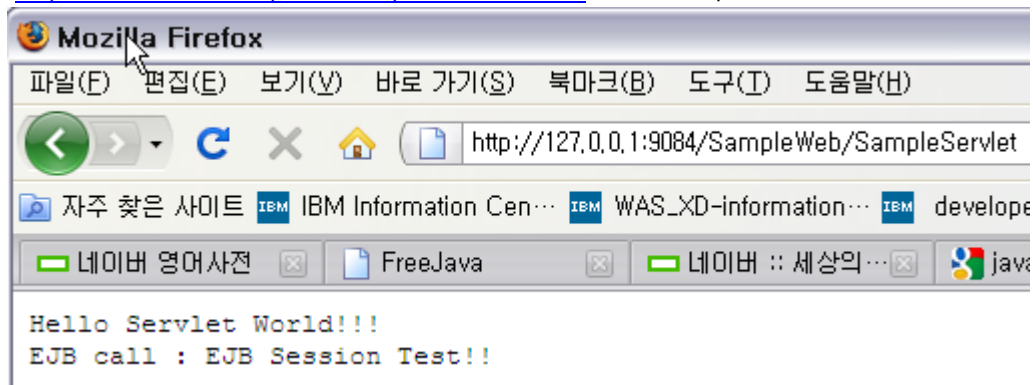
The application might not be immediately available while being started on all servers.

Changes have been made to your local configuration. You can:

- [Save](#) directly to the master configuration.
- [Review](#) changes before saving or discarding.

EAR 어플리케이션 배포작업이 완료되었으므로 웹 브라우저를 이용해서 제대로 동작하는지 테스트 해보도록 하겠습니다. 이전 강좌의 Servlet 호출 URL 을 입력합니다.

([http://127.0.0.1:9084\(port\)/SampleWeb\(Context root\)/SampleServlet\(Servlet 명\)](http://127.0.0.1:9084(port)/SampleWeb(Context root)/SampleServlet(Servlet 명)))



위의 화면과 같은 결과가 나온다면 작성하신 EJB Session bean 클래스 호출에 성공하신 것 입니다. 축하드립니다. 이제 EJB 소스 작성과 EAR 어플리케이션 작성 및 배포 테스트까지 완료하신 것입니다. 참 쉽죠^^&...그럼 이제 다음 강좌로 넘어가 볼까요...휘리릭~~~

참고 1) IBM Information Center for WebSphere Application Server v7

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.multipiplatform.doc/info/welcome_nd.html

참고 2) IBM Information Center for WebSphere Application Server v7

- Administering applications and their environment

<http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.doc/info/ae/ae/welc6topmanaging.html>

참고 3) IBM RAD v7.5 Information Center

<http://publib.boulder.ibm.com/infocenter/radhelp/v7r5/index.jsp>