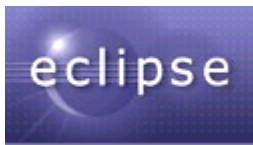


## 하나씩 쉽게 따라 해보는 IBM WebSphere Application Server(WAS) v7 – 3

이정운 ([juwlee@kr.ibm.com](mailto:juwlee@kr.ibm.com))

하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈 벌써 그 세 번째 이야기를 할 때가 왔네요. 그 세 번째 이야기는 바로 개발입니다. 개발이라, WAS를 강의하다 뭔가 생똥 맞은 주제가 갑자기 튀어나왔다는 생각이 드시나요? 그러나 이전에 설명드린 것처럼 WAS 는 기본적으로 JSP, Servlet, EJB 같은 Java 프로그램이 돌아갈 수 있는 Container, 엔진입니다. 이말은 즉, WAS 만으로 무엇인가를 할 수는 없고 반드시 돌려야 하는, 운영되어야 하는 J2EE 프로그램이 있어야만 실제로 무언가가 이루어 지고 서비스를 할 수 있다는 것입니다.

지금까지 첫 번째, 두 번째 강좌를 통해서 IBM WAS v7를 잘 설치했다면 이제는 이렇게 설치된 WAS 위에 올릴 어플리케이션을 작성하고 IBM WAS 에 배포하여 실제 구동을 시켜야합니다. 이때, Java 어플리케이션을 직접 Text editor 를 이용해서 개발할 수도 있지만 요즘에는 개발을 쉽게 해 줄수 있는 다양한 툴이 나와서 쉽고 빠르게 개발할 수 있게 도와주고 있습니다. 그 대표적인 툴 이 바로 많이들 사용하시는 Open Source 인 Eclipse 입니다.



그러나 Eclipse 에 대한 자료나 강좌는 인터넷에서 아주 많이, 쉽게 구할 수 있기 때문에 여기서 까지 굳이 다룰 필요는 없다고 생각합니다.

그렇기 때문에 이 강좌에서 사용하는 툴은 바로 RAD v7.5 입니다. RAD(Rational Application Developer) 는 IBM 에서 판매하는 상용 툴로서 J2EE 5.0 스펙을 준수하는 어플리케이션의 개발을 지원하고 오픈 소스인 Eclipse 기반이라, Eclipse 를 사용하실 줄 아는 분들이라면 사용하시는데 전혀 문제가 없이 더 막강한 기능들을 느낄 수 있을 것입니다. 특히 IBM WAS v7 을 위해서는 RAD v7.5 for WebSphere 를 제공하고 WebSphere 특징을 제대로 활용할 수 있는 개발이 가능하게 해줍니다. 그럼 설명은 이만 마치고 강좌를 진행하도록 하겠습니다.

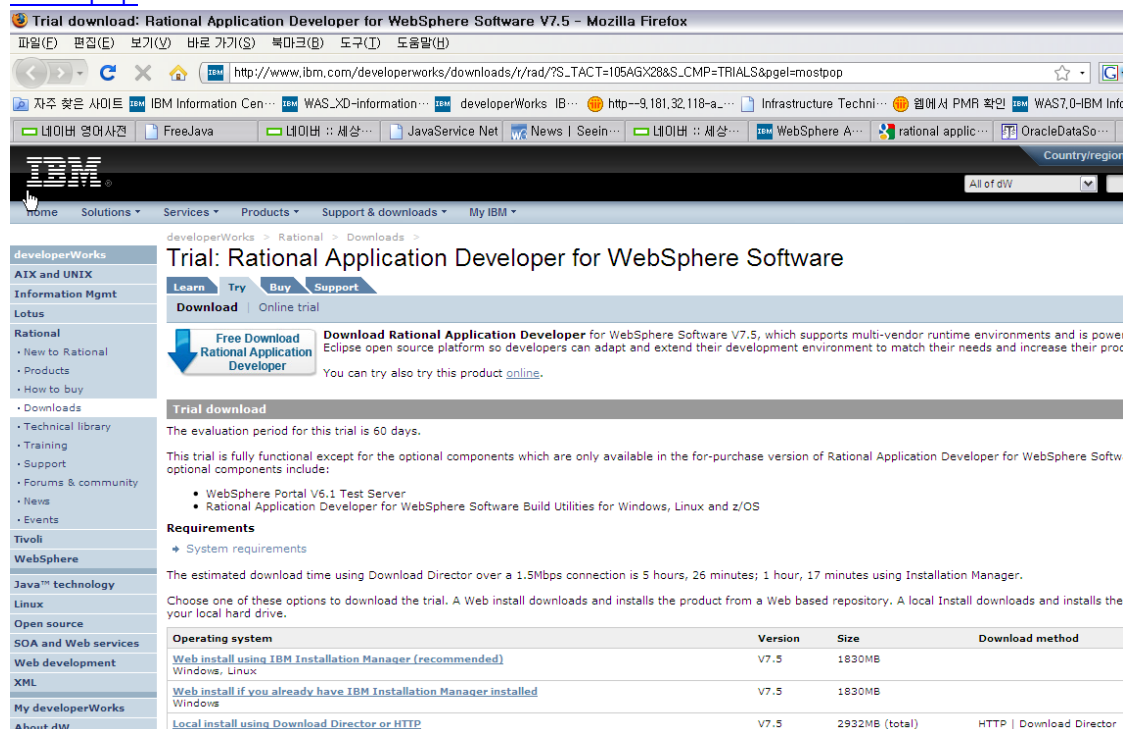
## Part 1. RAD v7.5 설치

RAD v7.5 는 이전에 이미 이야기 했지만 돈을 주고 사야하는 상용 툴입니다. 그러나 IBM WAS 를 구매하시게 되면 RADAD(Rational Application Developer Assembly&Deployment) v7.5 를 번들로 제공해드립니다. RAD의 일부기능을 제한적으로 사용할 수 있는 버전이지만 기본적인 어플리케이션 개발과 배포부분에서는 문제 없이 사용할 수 있습니다. 다만, 이를 다운받으려면 고객 번호가 있어야 하므로 우선 테스트해 볼 수 있는 환경을 구축하기 위하여 Trial version을 다운로드 받아서 사용해 보도록 하겠습니다.

(Trial version 은 60일 사용제한이 걸린 제품입니다.)

RAD v7.5 Trial version 을 다운 받기 위해서는 아래의 링크에 접속하셔서 다운받으시면 됩니다.

[http://www.ibm.com/developerworks/downloads/r/rad/?S\\_TACT=105AGX28&S\\_CMP=TRIALS&pgel=mostpop](http://www.ibm.com/developerworks/downloads/r/rad/?S_TACT=105AGX28&S_CMP=TRIALS&pgel=mostpop)



**Trial download: Rational Application Developer for WebSphere Software V7.5 - Mozilla Firefox**

developerWorks > Rational > Downloads >

### Trial: Rational Application Developer for WebSphere Software

**Download** | Online trial

**Free Download Rational Application Developer**

You can try also try this product [online](#).

**Trial download**

The evaluation period for this trial is 60 days.

This trial is fully functional except for the optional components which are only available in the for-purchase version of Rational Application Developer for WebSphere Software. optional components include:

- WebSphere Portal V6.1 Test Server
- Rational Application Developer for WebSphere Software Build Utilities for Windows, Linux and z/OS

**Requirements**

- System requirements

The estimated download time using Download Director over a 1.5Mbps connection is 5 hours, 26 minutes; 1 hour, 17 minutes using Installation Manager.

Choose one of these options to download the trial. A Web install downloads and installs the product from a Web based repository. A local Install downloads and installs the product to your local hard drive.

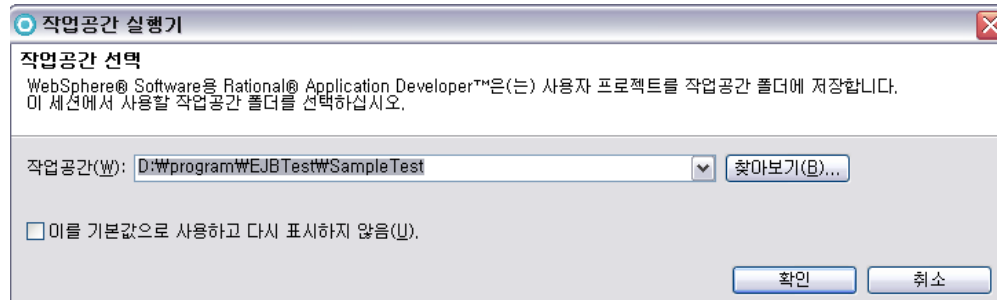
Operating system	Version	Size	Download method
Web install using IBM Installation Manager (recommended) Windows, Linux	V7.5	1830MB	
Web install if you already have IBM Installation Manager installed Windows	V7.5	1830MB	
Local install using Download Director or HTTP Windows, Linux	V7.5	2932MB (total)	HTTP   Download Director

RAD v7.5 를 다운받았으면 설치를 진행합니다.

(RAD v7.5 도 IBM 에서 판매하는 툴이므로 설치 마법사를 통해서 한 번에 쉽게 설치할 수 있습니다. 따라서, 별도로 Screen shot 및 가이드는 생략하도록 하겠습니다.)

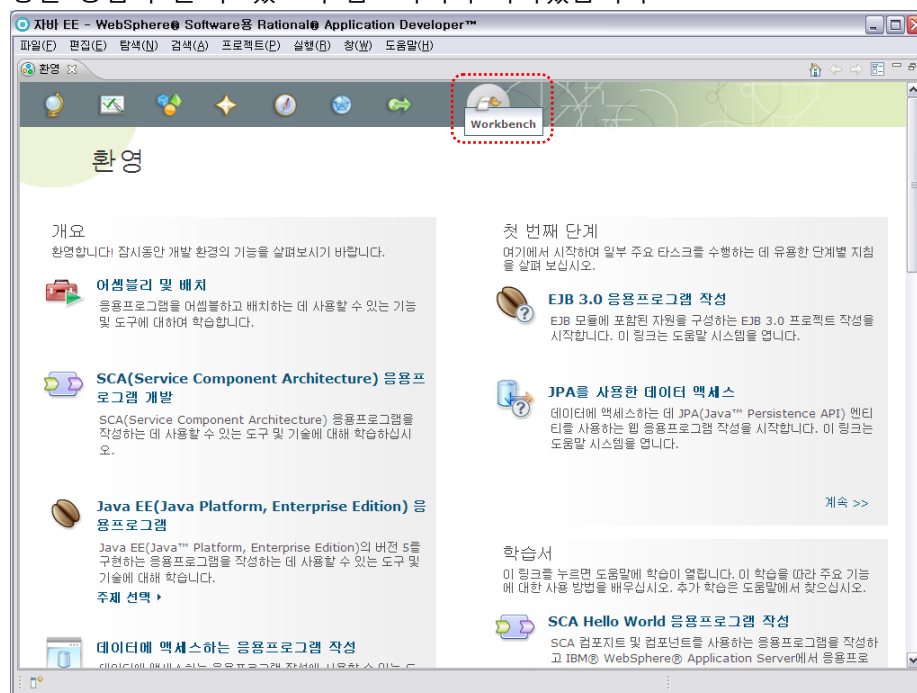
## Part 2. JSP, Servlet 개발

설치를 완료하셨다고 가정하고 RAD v7.5 를 실행합니다. (개발을 해본 경험이 있으신 분들인 이번 part 를 pass 하시고 배포부분을 보셔도 무방합니다. 또한, RAD 를 설치하는 것이 귀찮다고 생각하시는 분들은 그냥 Eclipse 를 사용해서 이 강좌를 따라해도 거의 똑같이 따라하실 수 있습니다.) RAD v7.5 를 실행하시면 처음에는 하단과 같은 작업공간을 설정하는 화면이 나옵니다. 작업공간이라는 것은 RAD 개발툴의 개발하는 소스들이 실제 저장되는 공간을 말하는 것이며 원하시는 위치를 적어주시면 됩니다.



확인 버튼을 누르면 하단과 같은 환영메세지와 RAD v7.5 가 실행됩니다. 그럼 오른쪽 위에 있는 workbench 를 클릭하여 실제 작업모드로 들어갑니다.

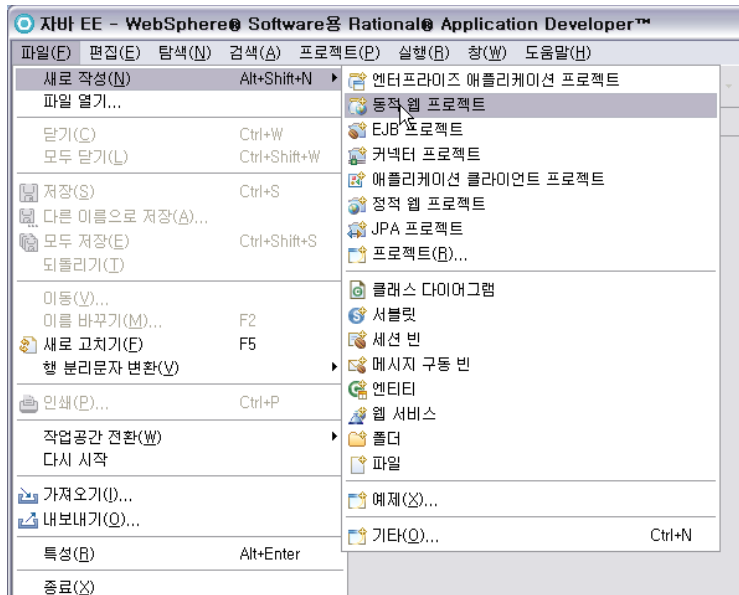
(RAD와 개발에 관심있으신 분들은 RAD에서 제공하는 여러 학습서와 샘플을 활용해 보시는 것도 좋은 방법이 될 수 있으니 참고하시기 바라겠습니다.



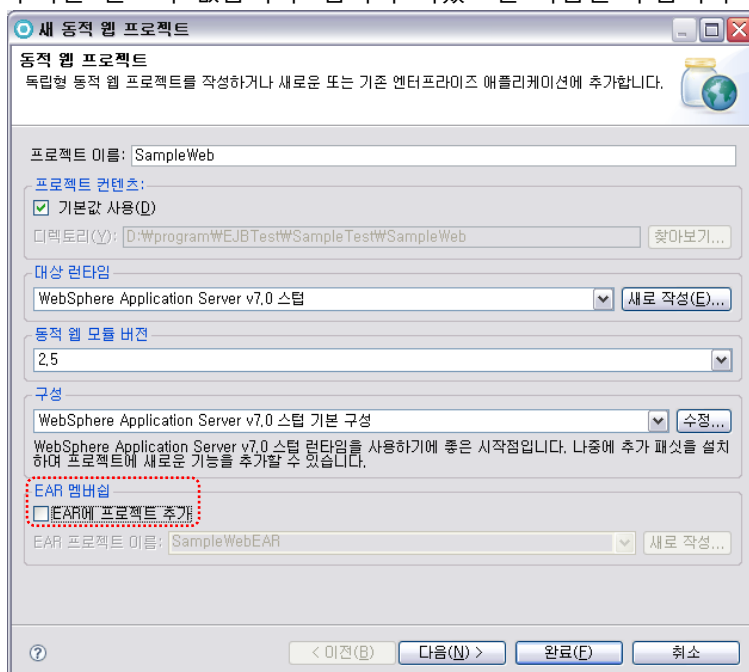
RAD v7.5 가 실행되었으므로 이제 개발을 할 수 있는 환경이 준비되었습니다. 처음 접해보는 분들을 위한 강좌라 어려운 EJB보다는 접근하기 쉬운 JSP와 Servlet 개발과 배포, 실행에 대해서 먼저 살펴보도록 하겠습니다. JSP와 Servlet 을 돌리기 위해서는 동적 웹 프로젝트를 작성해야 하며 향후 이 동적 웹프로젝트는 WAR 형태의 파일로 묶여서 IBM WAS v7에 배포되어 실행됩니다. 그럼 이 부분을 하나 하나 진행해 보도록 하겠습니다. (본 강좌에서는 개발에 대한 자세한 이론은

배제하도록 하겠습니다. 아시겠지만 본 강좌는 IBM WAS 를 설명하기 위한 용도로 만들어졌고 개발까지 다루기에는 scope 이 너무 넓기 때문입니다.)

개발을 진행하기 위해서는 우선 새로운 프로젝트를 만들어야 합니다. 프로젝트는 개발을 위한 하나의 작업공간이 됩니다. 그럼 새로운 프로젝트를 만들기 위하여 파일 > 새로작성 > 동적 웹프로젝트를 선택합니다.

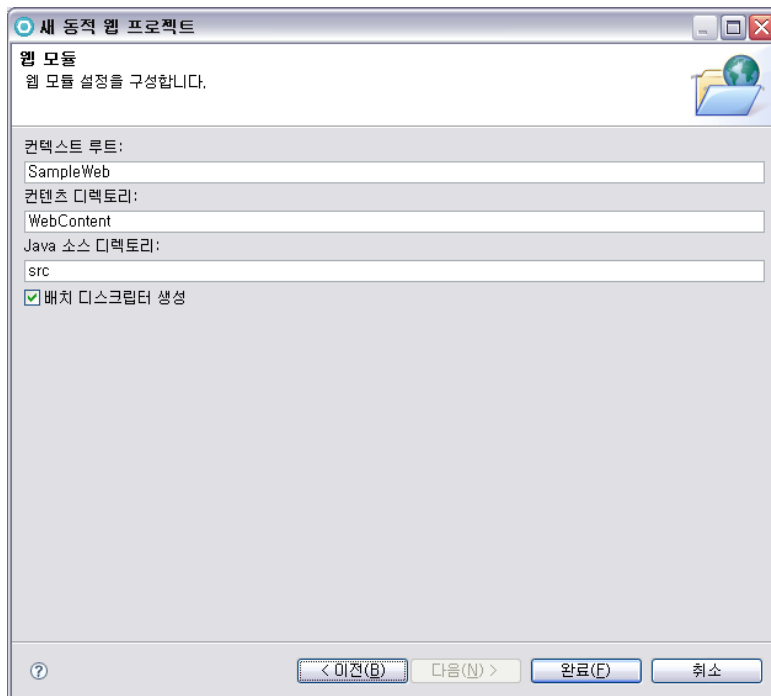


동적 웹 프로젝트는 웹 환경에서 많이 사용되는 JSP 와 Servlet 을 돌릴 수 있는 어플리케이션 작업공간을 만들 수 있는 프로젝트 이며 WAR(Web Archive) 형태의 패키지로 만들 수 있습니다. 동적 웹 프로젝트 마법사가 나오면 사용하고자 하는 프로젝트 이름을 넣어주고 EAR 프로젝트에 추가 부분을 disable 해줍니다. 이번 강좌에서는 WAR 만 만들 것 이므로 굳이 EAR에 프로젝트를 추가할 필요가 없습니다. 입력이 되었으면 다음을 누릅니다.

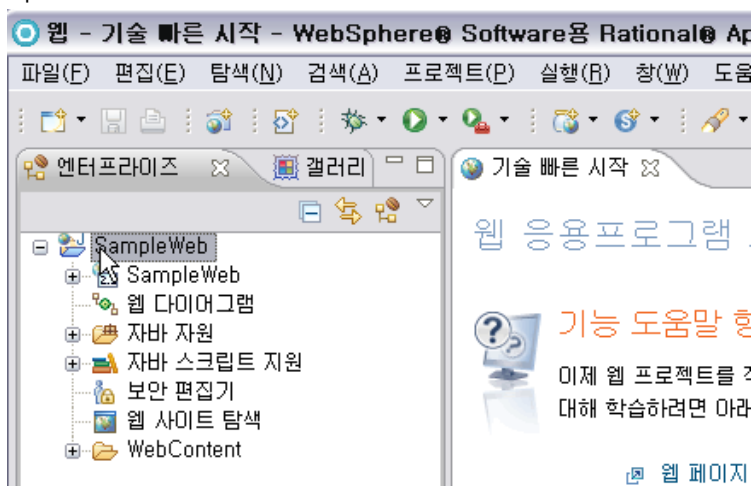


웹 모듈 설정을 확인한 후 완료 버튼을 누릅니다. 여기서 중요한 부분은 컨텍스트 루트(Context root)입니다. 컨텍스트 루트는 말 그대로 해당 내용의 루트입니다. 예를 들어 <http://127.0.0.1/>까지는 기본 루트이고 그 이후에 나올 수 있는 것이 컨텍스트 루트입니다. 위의 예에서는 '/'이 컨텍스트 루트가 되며 <http://127.0.0.1/SampleWeb> 이렇게 되면 'SampleWeb'이 컨텍스트 루트가 됩니다. 이 후 이 WAR에서 만들어지는 모든 Servlet 및 JSP는 해당 컨텍스트 루트 기준으로 찾을 수 있습니다. (예를 들어 TestServlet을 만들었다면 <http://127.0.0.1/SampleWeb/TestServlet>으로 요청을 전송하고 찾을 수 있습니다.)

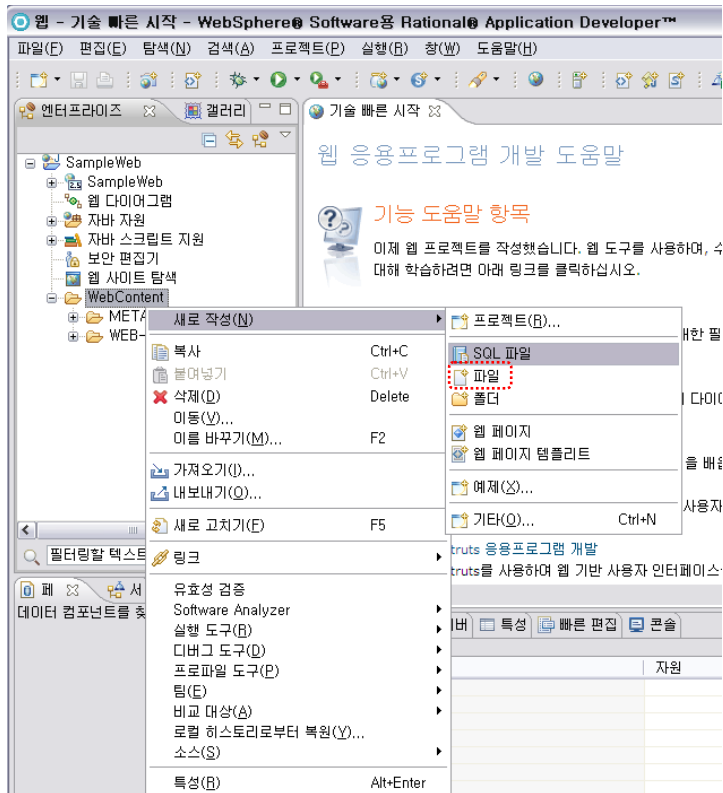
WAS에서는 어플리케이션의 요청이 들어오면 이 컨텍스트 루트를 기준으로 라우트 하기 때문에 다른 어플리케이션의 컨텍스트 루트와 중복만 되지 않으면 편하신 방법으로 작성하시면 됩니다.



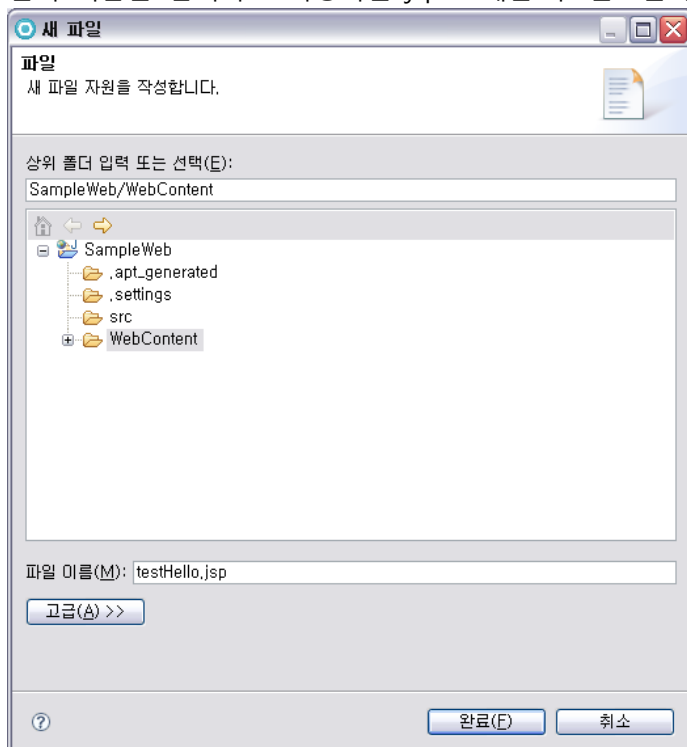
완료를 누르게 되면 기존에 주어진 이름으로 동적 웹 프로젝트가 생성된 것을 RAD 화면에서 확인할 수 있습니다. 또한 웹 프로젝트를 위한 모든 폴더 구성이 만들어진 것도 확인할 수 있습니다.



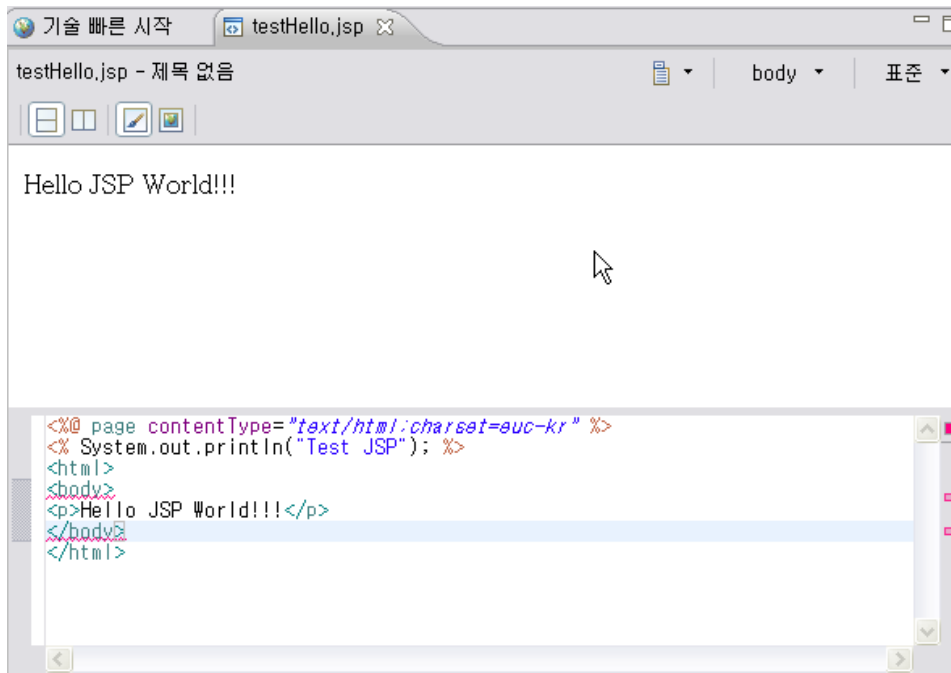
이제 동적 웹 프로젝트가 만들어졌으면 작업공간과 환경은 만들어졌고 실제로 운영될 수 있는 샘플 JSP 어플리케이션을 작성해 보도록 하겠습니다. 동적 웹 프로젝트의 WebContent 를 클릭한 후 마우스 오른쪽 버튼을 누릅니다.



나타난 팝업화면에서 새로 작성 > 파일을 선택합니다. 새 파일 마법사가 나타나는데 원하시는 파일의 이름을 입력하고 확장자를 jsp로 해준 후 완료를 클릭합니다.

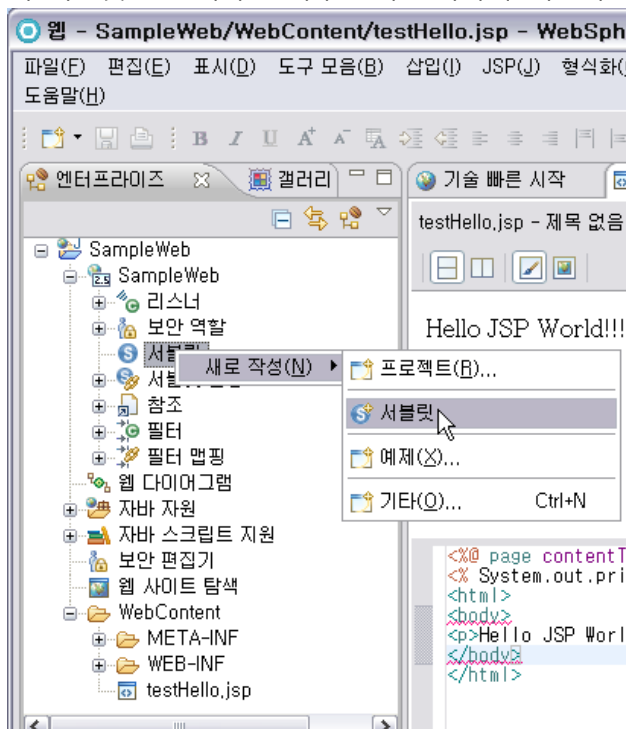


완료를 하게되면 파일 창이 생성되는데 JSP 실행시 화면에 출력할 간단한 문구를 하단 처럼 입력 합니다.



이와 같은 방식으로만 하시면 쉽게 JSP 개발을 하실 수 있습니다. 단, JSP 규약에 따라서 맞추어서 개발을 해주어야 한다는 전제가 있지만 어쨌든 특별한 설정 같은 것들이 없어서 쉽게 개발할 수 있습니다. 어때요 쉽죠..^^&

이전 단계까지 JSP 개발이 끝났으니 이제는 Servlet 개발을 살펴보도록 하겠습니다. 왼쪽 메뉴에서 서블릿을 선택하고 마우스 우클릭하여 새로작성 > Servlet 을 선택합니다.



서블릿 작성 마법사가 나오면 해당 서블릿 클래스 이름을 원하시는데로 넣어주고 다음을 누릅니다.

**서블릿 작성**

클래스 파일 위치를 지정합니다.

웹 프로젝트: SampleWeb

소스 폴더: \\SampleWeb\\src 찾아보기...

자바 패키지: 찾아보기...

**클래스 이름: SampleServlet**

수퍼클래스: javax.servlet.http.HttpServlet 찾아보기...

☐ 기존 서블릿 클래스 또는 JSP 사용

클래스 이름: SampleServlet 찾아보기...

? < 이전(B) 다음(N) > 완료(F) 취소

배치 디스크립터 정보를 확인한 후 완료를 눌러 작성을 마무리 합니다. 여기서 URL 매핑이란 실제로 어떤 URL 호출되었을 때 해당하는 Servlet 클래스를 호출할지 정의하는 내용입니다. 즉, 나중에 웹 브라우저에서 호출할 때 이 Servlet 매핑 이름으로 호출하게 되니 기억해두시기 바랍니다.

**서블릿 작성**

서블릿 배치 디스크립터 특정 정보를 입력하십시오.

이름: SampleServlet

설명: ?

초기화 매개변수:

이름	값	설명

추가... 편집... 제거

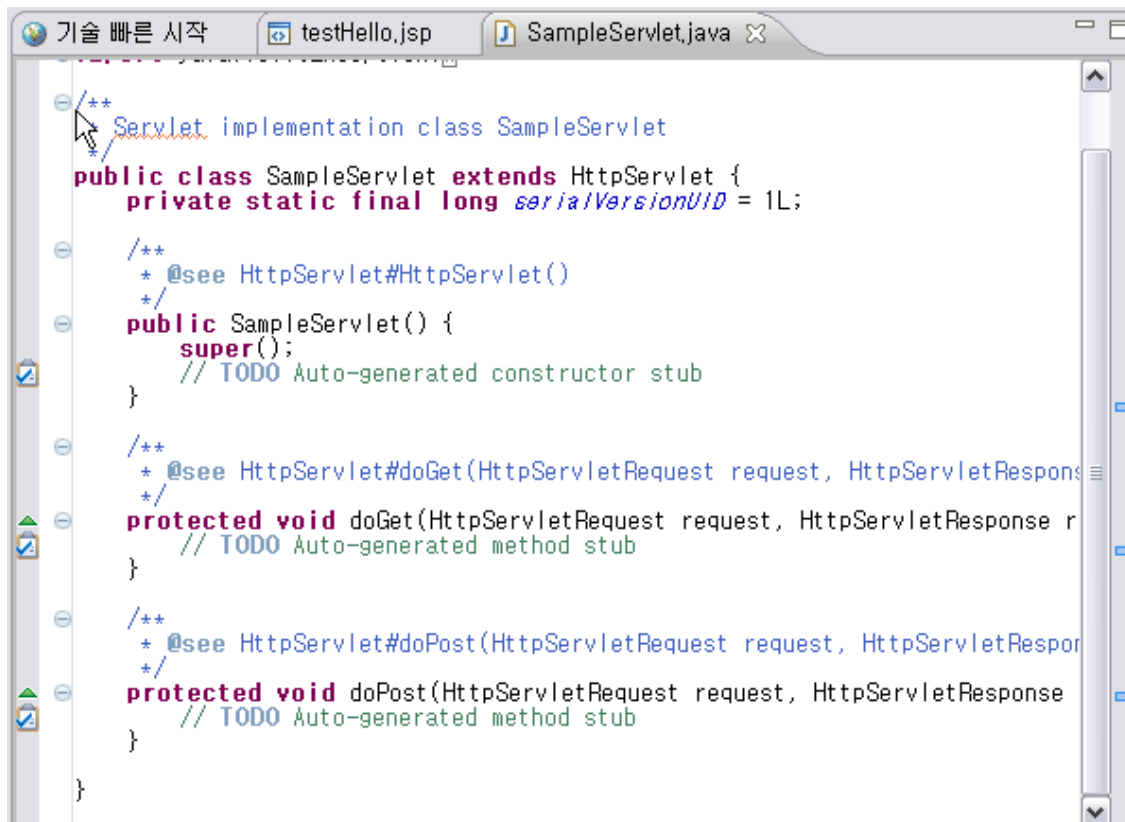
**URL 매핑: /SampleServlet**

추가... 편집... 제거

? < 이전(B) 다음(N) > 완료(F) 취소



서블릿 작성 마법사를 완료하게 되면 동적으로 Servlet 의 기본 틀이 만들어집니다. 즉, Servlet 에서 반드시 필요한 구문들이 다 들어가 있는 소스가 아래처럼 자동으로 생성됩니다. Text Editor 가 아닌 툴을 사용할 때의 가장 큰 장점은 필요한 소스를 자동으로 생성해 준다는 것입니다.



```
/**
 * Servlet implementation class SampleServlet
 */
public class SampleServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public SampleServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub
    }
}
```

여기서 필요한 부분에 소스만 추가하여 간단한 출력이 나오는 Servlet 샘플을 작성할 수 있습니다.

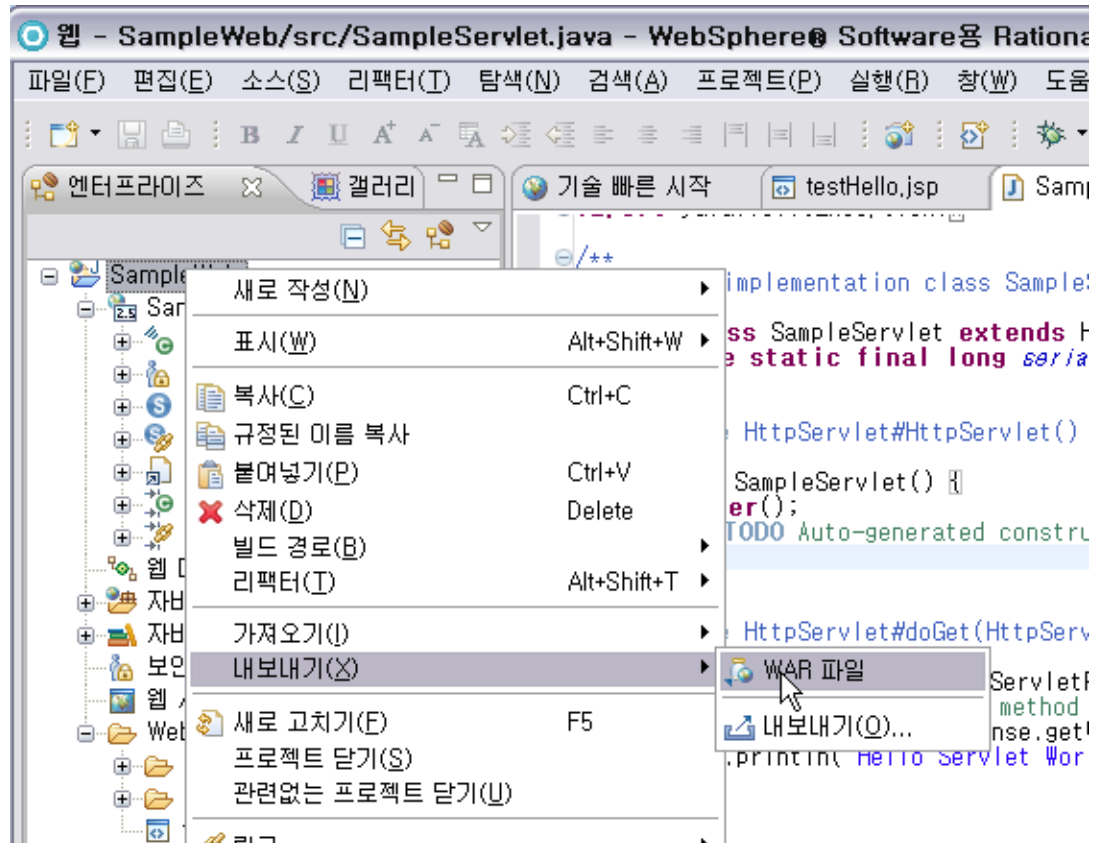


```
/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) {
    // TODO Auto-generated method stub
    PrintWriter out = response.getWriter();
    out.println("Hello Servlet World!!!");
}

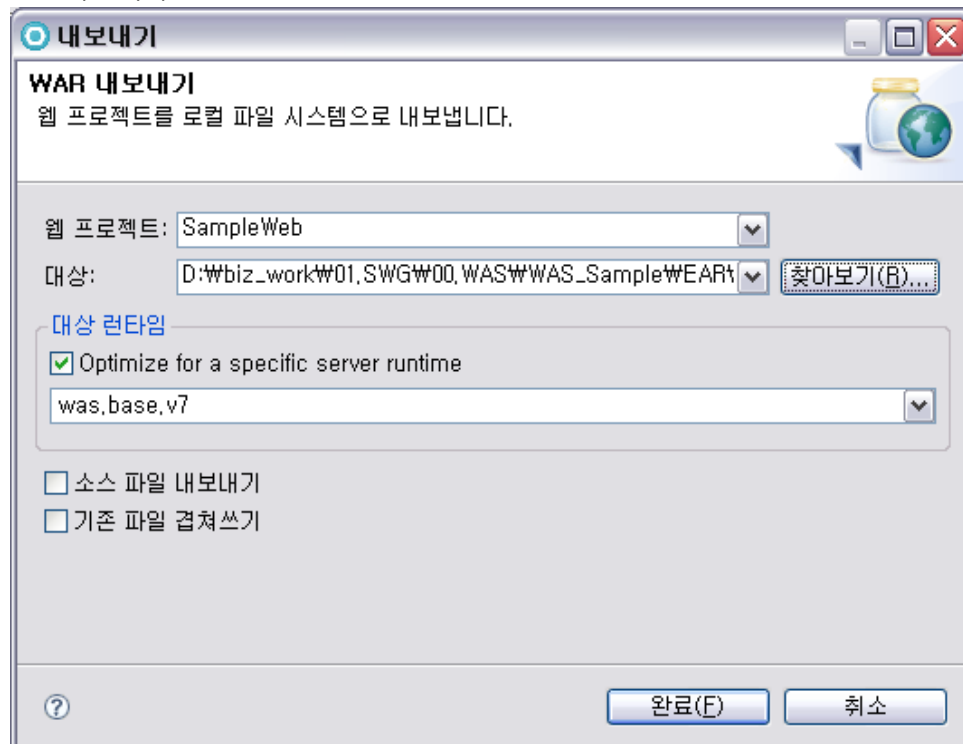
/**
 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
 */
protected void doPost(HttpServletRequest request, HttpServletResponse response) {
    // TODO Auto-generated method stub
}
```

간단하게 Servlet 을 설명드리자면 Servlet 은 호출되면 Get 방식일 경우 초기화 이후 자동으로 doGet() 메소드가 호출되므로 이렇게 하시면 화면에 문구를 출력하는 간단한 샘플 Servlet 작성을 완료하실 수 있습니다. (소스를 보시면 바로 아시겠지만 Post 방식일 경우에는 doPost() 가 호출됩니다.) 여기까지 원하시는 대로 JSP와 Servlet 을 만들었기 때문에 기본적으로 돌릴 수 있는 소스는 완성이 되었습니다. 이제 이 소스를 WAR 파일 형태의 패키지로 묶고 실제 IBM WAS v7 에 배포하는 일이 남아 있습니다.

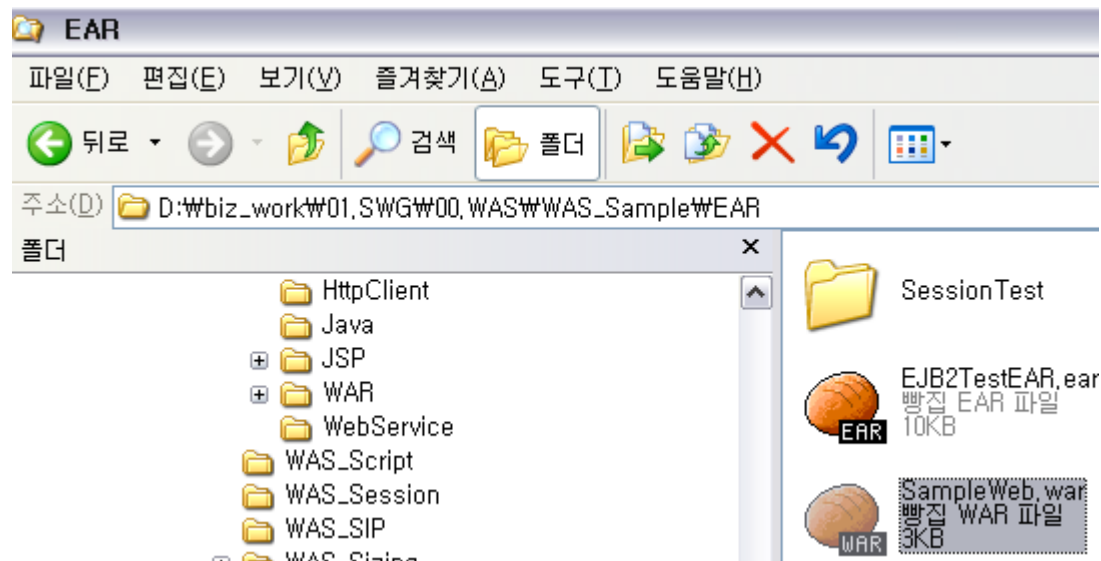
동적 웹 프로젝트 이름에서 마우스 우 클릭하여 내보내기 > WAR 파일을 선택합니다.



WAR 파일을 내보낼 디렉토리 위치를 선택한 후에 완료 버튼을 누르면 성공적으로 WAR 파일이 만들어집니다.

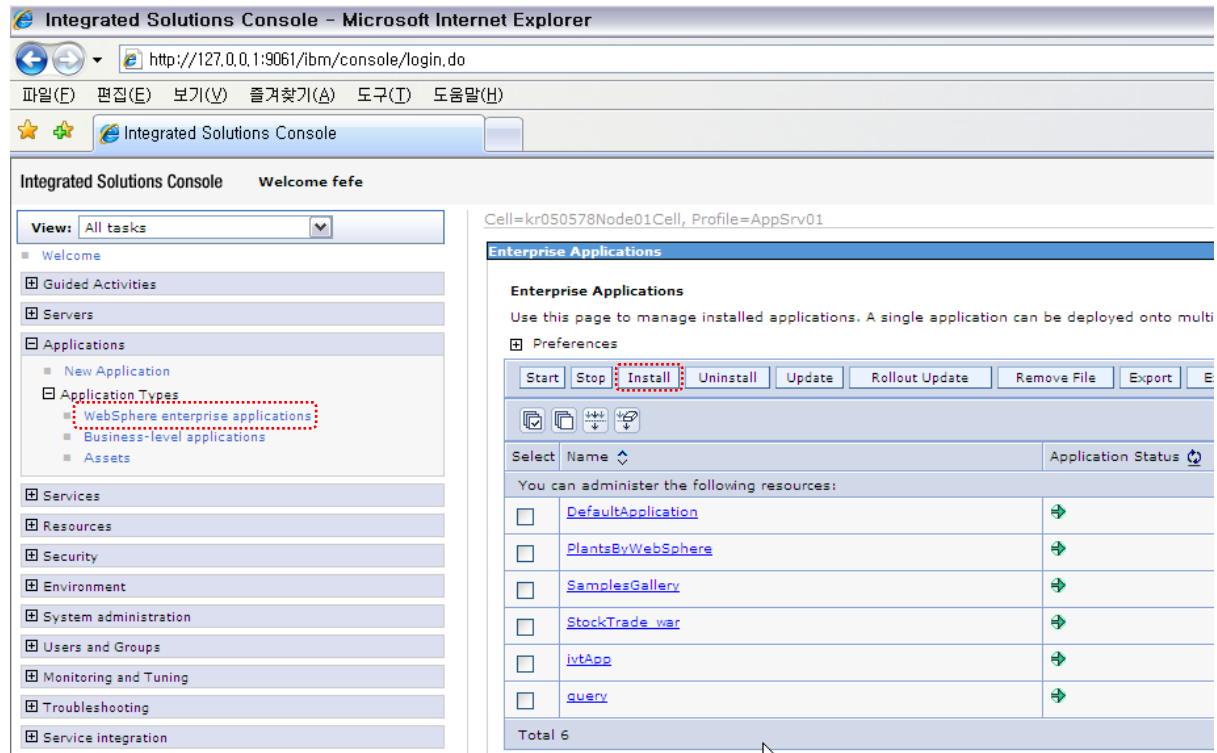


탐색기를 통해서 WAR 파일 내보내기로 지정된 폴더를 확인해 보시면 하단처럼 WAR 파일을 확인하실 수 있습니다. 해당 파일을 확인하였다면 개발 완료, 이제 WAS 에 배포할 일만 남았습니다. 자 자 힘을 내세요..^^&;

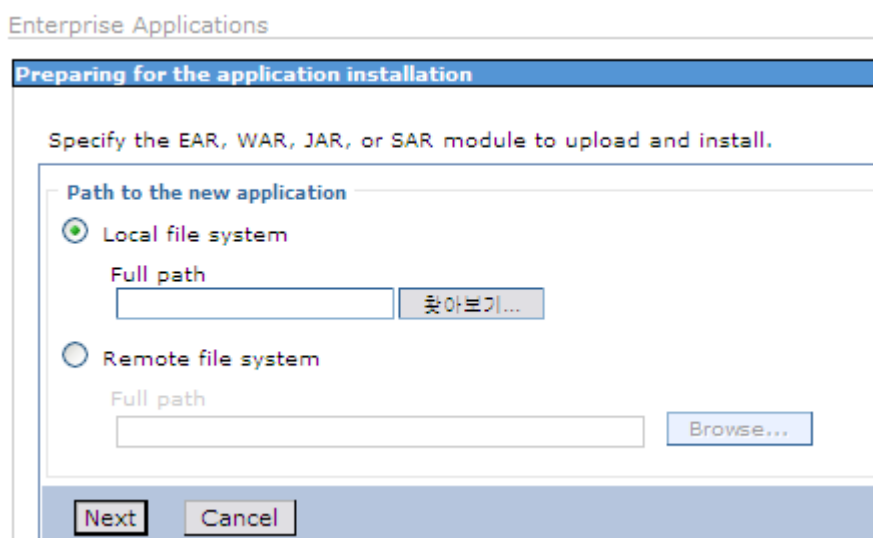


### Part 3. 어플리케이션을 IBM WAS v7 에 배포

어플리케이션을 다 작성했으면 이제 실제로 그 어플리케이션을 동작 시키기 위하여 J2EE 엔진인 WAS 위에 배포하고 설치하는 일이 남았습니다. WAS 를 시작하여 관리콘솔로 들어갑니다. 그리고 관리콘솔의 메뉴에서 Applications > WebSphere enterprise applications 를 선택합니다.



그러면 위와 같이 기존에 자동으로 디플로이된 샘플어플리케이션을 확인할 수 있습니다. 이제 이전 단계에서 만들었던 어플리케이션을 배포하기 위하여 설치(Install) 버튼을 누릅니다. 어플리케이션 인스톨 준비창이 나오면 찾아보기를 클릭하여 이전에 추출했던 WAR 파일을 선택하고 다음을 누릅니다



어플리케이션 설치를 어떤 방식으로 할 것인가에 대한 질문이 나오는데 간단하게 설치하기 위하여 빠른 설치(Fast path) 를 선택하고 다음을 누릅니다.

- Fast Path : 빠른 설치를 위해서 반드시 필요한 옵션만 선택할 수 있는 설치 방법입니다.
- Detailed : 설치에 사용될 수 있는 모든 옵션들을 선택할 수 있게 하는 설치 방법입니다.

#### Enterprise Applications

Preparing for the application installation

How do you want to install the application?

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

☐ Detailed - Show all installation options and parameters.

☒ Choose to generate default bindings and mappings

Previous Next Cancel

어플리케이션 설치 마법사가 나타나면 여러 옵션을 입력할 수 있지만 간단한 테스트용 어플리케이션의 배포이기 때문에 특별한 내용의 입력없이 확인만 한번 하고 다음을 누릅니다.

Step 1: Select installation options

Step 2: Map modules to servers

Step 3: Map context roots for Web modules

Step 4: 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

☒ Distribute application

☐ Use Binary Configuration

☐ Deploy enterprise beans

Application name

SampleWeb\_war

☒ 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#.\*\.\*s=755

Application Build ID

Unknown

☐ Allow dispatching includes to remote resources

☐ Allow servicing includes from remote resources

Business level application name

Create New BLA

Asynchronous Request Dispatch Type

Disabled

☐ Allow EJB reference targets to resolve automatically

Next Cancel

다음으로 어플리케이션을 서버에 매핑하는 화면이 나오는데 현재는 하나의 서버만이 설치되어 있는 환경이므로 바로 다음으로 넘어갑니다. 간단히 이 메뉴에 설명을 드리자면 WAS 에서는 어플리케이션을 서버와 매핑을 합니다. 그러면 해당 서버에서 해당 어플리케이션이 구동됩니다.

Specify options for installing enterprise applications and modules.

Step 1 Select installation options  
→ Step 2: Map modules to servers  
★ Step 3 Map context roots for Web modules  
Step 4 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:

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

마법사에서 컨텍스트 루트를 설정하는 화면이 나오면 원하는 컨텍스트 루트를 입력하고 다음을 누릅니다.

Specify options for installing enterprise applications and modules.

Step 1 Select installation options  
Step 2 Map modules to servers  
→ Step 3: Map context roots for Web modules  
Step 4 Summary

### Map context roots for Web modules

Context root defined in the deployment descriptor can be edited.

Web module	URI	Context Root
SampleWeb	SampleWeb.war,WEB-INF/web.xml	<input type="text" value="/SampleWeb"/>

마지막으로 요약정보를 확인하여 이상이 없으면 완료를 누릅니다.

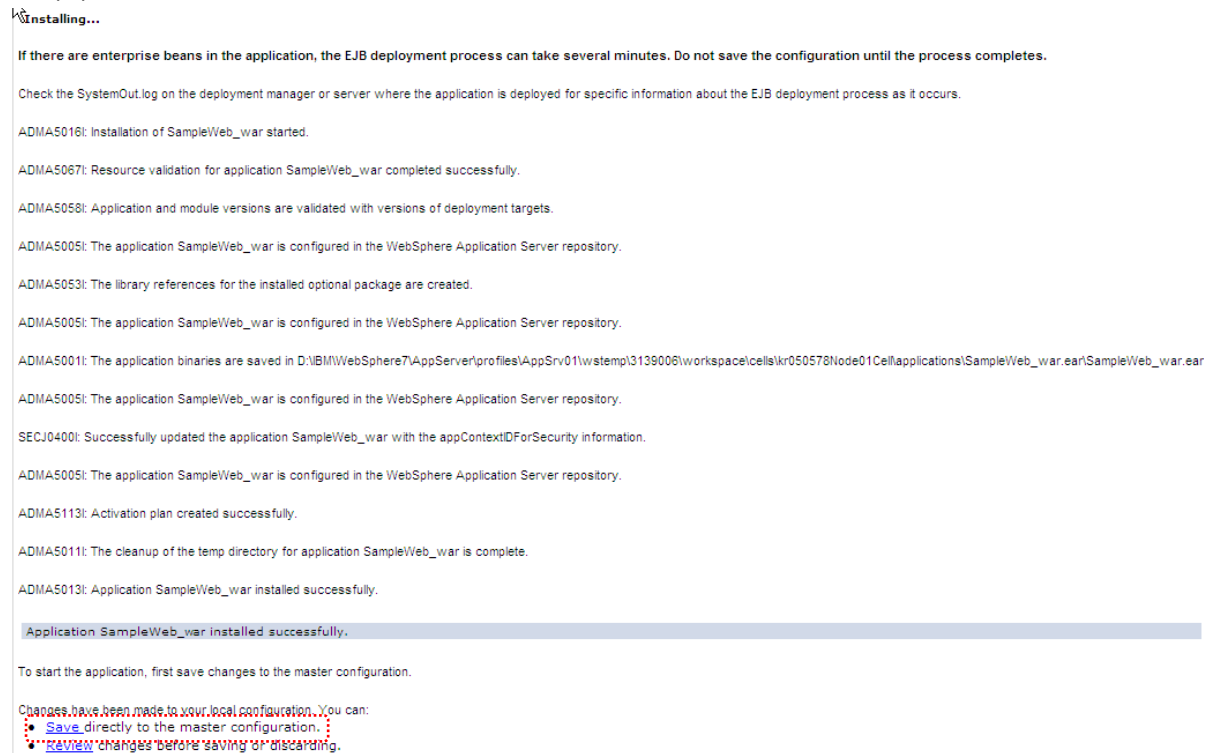
Step 1 Select installation options  
Step 2 Map modules to servers  
Step 3 Map context roots for Web modules  
→ Step 4: Summary

### Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Application name	SampleWeb_war
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	.*\,dl=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
Business level application name	
Asynchronous Request Dispatch Type	Disabled
Allow EJB reference targets to resolve automatically	No
Cell/Node/Server	<a href="#">Click here</a>

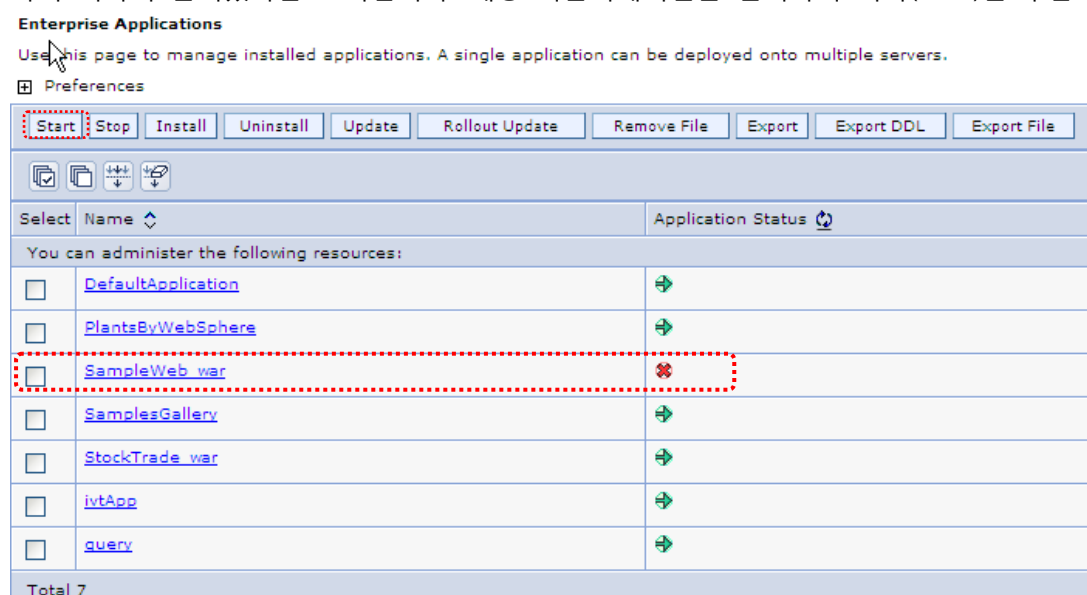
완료가 눌러지면 어플리케이션 설치가 아래 화면에서 처럼 실제로 진행되며 진행현황이 표시 됩니다.



설치 진행이 이상없이 완료되면 저장(Save) 을 클릭하여 어플리케이션 설치 정보를 마스터 저장소에 저장합니다.

(지금은 노드 한 곳에 서버 한대만 설치된 환경이지만 여러 개의 노드로 분산된 환경인 경우에는 동기화 작업을 반드시 수행해 주어야 하며, 그 때 실제 어플리케이션들이 각 노드로 반영됩니다.)

어플리케이션 리스트화면에서 보면 이번에 설치된 어플리케이션만 빨간색 X로 표시되는데 그것은 아직 시작이 안되었다는 표시입니다. 해당 어플리케이션을 선택하여 시작(Start)을 누릅니다.



어플리케이션 시작이 성공적으로 완료되면 아래와 같은 화면을 보실 수 있습니다.



Messages

Application SampleWeb\_war on server server1 and node kr050578Node01 started successfully.

### Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple servers.

#### Preferences

StartStopInstallUninstallUpdateRollout UpdateRemove FileExportExport DDLExport File

Select

Name

Application Status

You can administer the following resources:

<input type="checkbox"/>	<a href="#">DefaultApplication</a>	
<input type="checkbox"/>	<a href="#">PlantsByWebSphere</a>	
<input type="checkbox"/>	<a href="#">SampleWeb_war</a>	
<input type="checkbox"/>	<a href="#">SamplesGallery</a>	
<input type="checkbox"/>	<a href="#">StockTrade_war</a>	
<input type="checkbox"/>	<a href="#">jvtApp</a>	
<input type="checkbox"/>	<a href="#">query</a>	

Total 7

여기까지 잘 따라오셨다면 이번 강좌의 가장 큰 부분인 어플리케이션 배포를 성공적으로 완료하신 것입니다. 이제는 마무리로 제대로 어플리케이션이 동작하고 있는지 웹 브라우저를 이용해서 실제 테스트를 진행해 보도록 하겠습니다.



#### Part 4. 어플리케이션 테스트

지금까지 RAD v7.5 로 만든 어플리케이션을 IBM WAS v7 에 제대로 배포했다면 실제로 이 어플리케이션이 제대로 동작하는지 확인해볼 필요가 있습니다. 사용하시는 웹 브라우저를 열고 아래의 URL을 입력합니다.

http://127.0.0.1:9084(WC\_default\_port)/SampleWeb(컨텍스트 루트)/testHello.jsp (호출하는 JSP)



위와 같은 화면이 나오면 정상적으로 어플리케이션이 시작했고 해당 JSP 도 잘 수행되어서 화면에 문구가 출력이 된 것입니다. 이전에 JSP 를 작성할 때 System.out.println() 구문을 통하여 시스템의 출력으로도 메시지를 출력하게 했으므로 WAS의 SystemOut.log 파일을 확인합니다. SystemOut.log 는 WAS 에서 출력되는 모든 Log 와 어플리케이션에서 출력되는 모든 Log 를 확인할 수 있는 파일입니다.

(이전 강좌와 동일하게 설치했다면 아래 위치에서 SystemOut.log를 확인 하실 수 있습니다.

D:\IBM\WebSphere7\AppServer\profiles\AppSrv01\logs\server1 )

```
3589 [09. 6. 18 17:30:02:531 KST] 00000020 AdminHelper A ADMN1009I: SampleWeb_war 응용프로그램을 시작하려고 시도했
3590 [09. 6. 18 17:30:03:734 KST] 00000020 CompositionUn A WSVR0190I: BLA WebSphere:blaname=SampleWeb_war에서 작성
3591 [09. 6. 18 17:30:04:500 KST] 00000020 ApplicationMg A WSVR0200I: SampleWeb_war 응용프로그램을 시작하는 중
3592 [09. 6. 18 17:30:04:500 KST] 00000020 ApplicationMg A WSVR0204I: 응용프로그램: SampleWeb_war 응용프로그램 빌드
3593 [09. 6. 18 17:30:04:937 KST] 00000020 webapp I com.ibm.ws.webcontainer.webapp.WebGroupImpl WebGroup SRVE01
3594 [09. 6. 18 17:30:05:203 KST] 00000020 WASSessionCor I SessionContextRegistry getSessionContext SESN0176I: 응용프
3595 [09. 6. 18 17:30:05:312 KST] 00000020 webcontainer I com.ibm.ws.wswebcontainer.VirtualHost addWebApplication SRV
3596 [09. 6. 18 17:30:05:359 KST] 00000020 ApplicationMg A WSVR0221I: SampleWeb_war 응용프로그램이 시작됨
3597 [09. 6. 18 17:30:05:546 KST] 00000020 CompositionUn A WSVR0191I: BLA WebSphere:blaname=SampleWeb_war에서 작성
3598 [09. 6. 18 17:34:15:093 KST] 0000001f SystemOut O Test JSP
3599 [09. 6. 18 17:34:15:093 KST] 0000001f SystemOut O Test JSP
```

위의 메시지까지 확인하셨다면 성공적으로 JSP 테스트가 완료되었습니다. 그럼 다음단계로 넘어가서 하단의 URL을 입력하여 Servlet 테스트를 진행합니다.

http://127.0.0.1:9084(WC\_default\_port)/SampleWeb(컨텍스트 루트)/SampleServlet(호출하고자 하는 Servlet 매핑)



위의 화면처럼 결과가 나온다면 Servlet 테스트도 무사히 완료된 것입니다. 지금까지 RAD v7.5 설치 및 어플리케이션을 제작해보았고, 이를 다시 WAR 파일로 패키징하여 WAS에 배포와 실제 JSP, Servlet 수행 테스트를 진행했습니다. 어때요 쉬우신가요 ? 아님, 그래도 아직 어려우신가요 ? 처음에 말씀드린대로 이 강좌는 기본적인 것들만 설명드리며 Jsp나 Servlet 관련 개발을 원하신다면 개발 관련 공부를 따로 하셔야 합니다. 여기서는 해당 어플리케이션을 어떻게 WAS 위에서 돌릴 것인가에 대해 중점적으로 말씀드렸습니다. 즉, WAS 를 사용하는 사용자 관점에 맞추어진 강좌이며 최대한 쉽게 설명하도록 노력하고 있습니다. 그럼 이번 강좌도 이걸로 마무리하고 다음 강좌에서 계속해서 진행하도록 하겠습니다. 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](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>

※이 자료의 저작권은 작성자에게 있으며 유포는 자유로이 허용되나 상업적으로 이용은 금합니다.