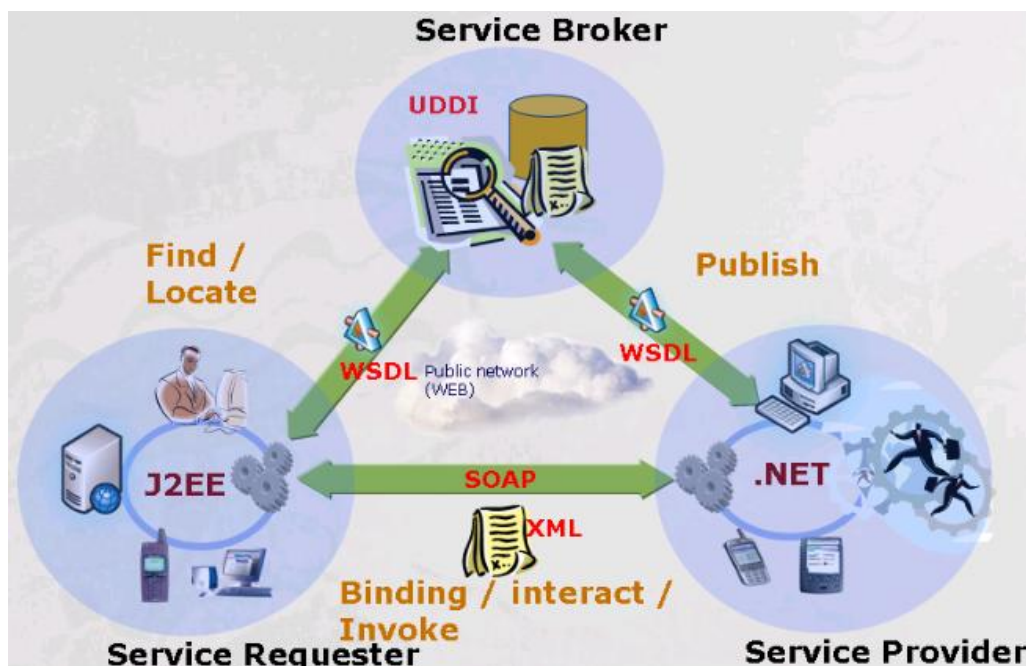


하나씩 쉽게 따라 해보는 IBM WebSphere Application Server(WAS) v7 – 9

이정운 (juwlee@kr.ibm.com)

하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈 그 아홉번째 이야기 Web Service 입니다.(이제 아홉수 ^^&) Web Service 란 표준화된 XML message를 통해 network상에서 접근 가능한 연산들의 집합을 기술하는 Interface로 정의되며 XML 기술을 기반으로 기존의 웹 환경을 이용한 분산컴퓨팅을 가능케 함으로써 웹을 통한 시스템 통합을 용이하게 할 수 있는 기술입니다. 즉, 인터넷과 표준 Interface 인 XML 을 이용하여 이기종 시스템간의 연동을 가능하게 하는 기반기술입니다.

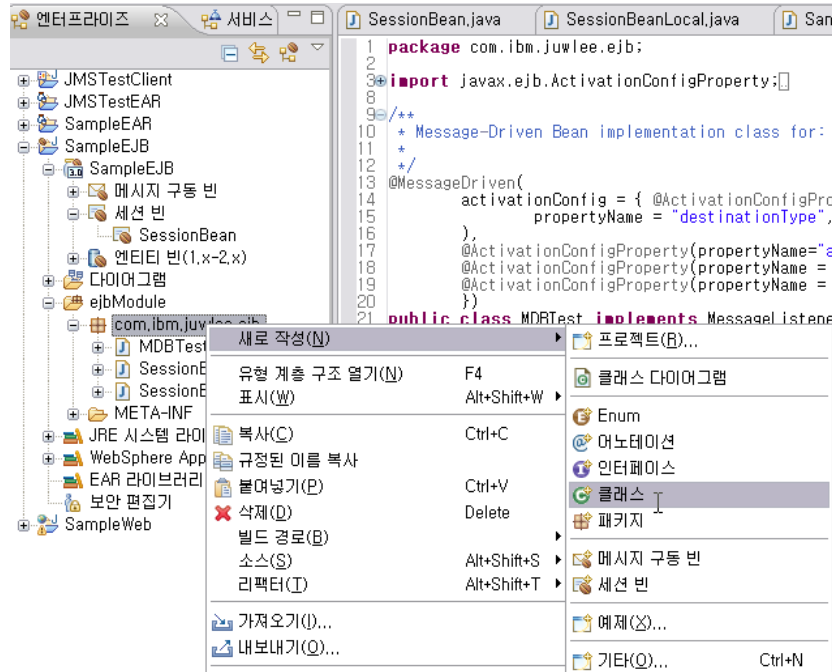
조금 어렵나요? 좀 더 쉽게 설명드리자면 사전에 해당 서버는 WSDL 형식으로 필요한 서비스를 오픈합니다. 그러면, 그 서비스를 사용하고자 하는 사용자는 WSDL에서 제공하는 Interface 를 준수하여 XML 형식으로 되어 있는 SOAP 이라는 프로토콜 형식으로 해당 서버에 Message 를 인터넷 상의 서버에 전달합니다. 이렇게만 하면 원하는 서비스가 이루어지는 구조입니다. 이때, 중요한 것은 Java 프로그램을 이용해서 Web Service 를 만드는 것인데 J2EE 5.0 규약에서는 이를 위해서 JAX-WS 라는 표준을 제공해 줍니다. JAX-WS(Java API for XML Web Services)는 웹 서비스를 생성하는 자바 API로써, Java EE의 일부입니다. 다른 Java EE의 자바 API와 같이, JAX-WS는 Java SE 5에서 도입된 어노테이션을 사용하여 웹 서비스 클라이언트 및 서버 모듈의 개발 및 배포를 쉽게 도와주는 역할을 합니다.



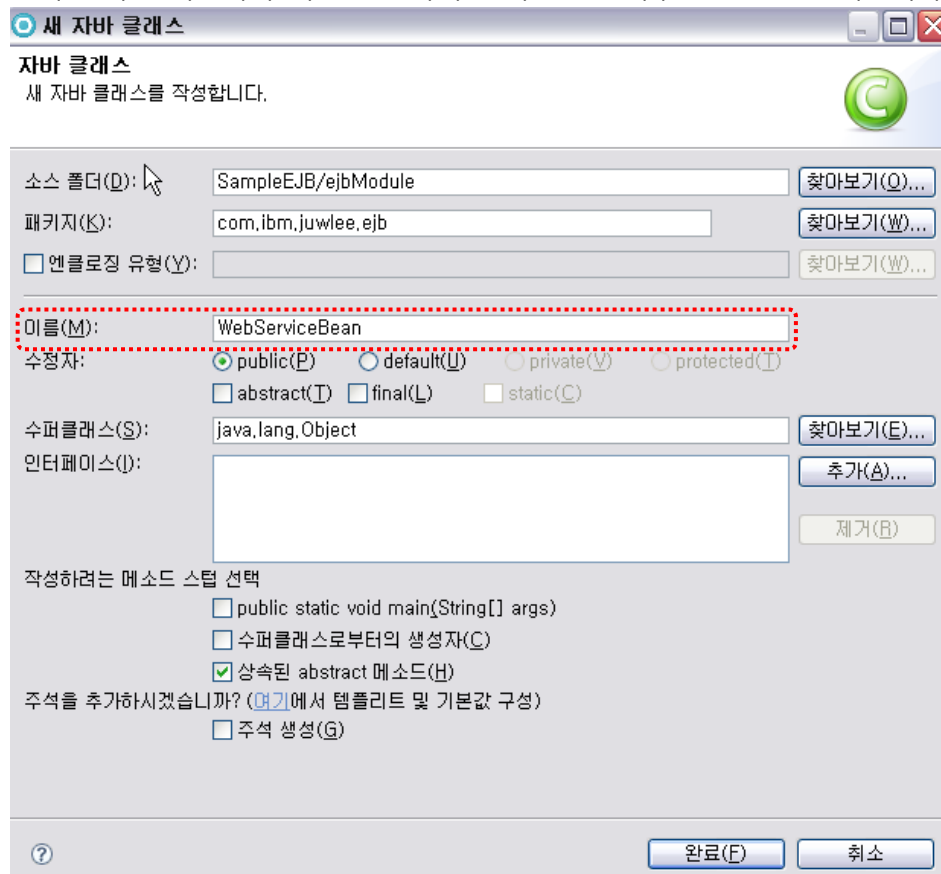
강의 전에, Web Service 를 만드는 방법에는 두가지가 있습니다. WSDL 을 먼저 만들고 나서 그것을 이용해 만드는 방법과 Java 클래스를 먼저 만들고 그 클래스를 활용하여 WSDL 과 Web Service를 만드는 방법이 있습니다. 이번 강좌에서는 개발이 편리한 Java 클래스를 먼저 만들고 JAX-WS 를 이용해서 Web Service 를 만드는 방법으로 진행하도록 하겠습니다. 그럼 본격적으로 강좌를 시작해 보겠습니다.

Part 1. Web Service 어플리케이션 개발

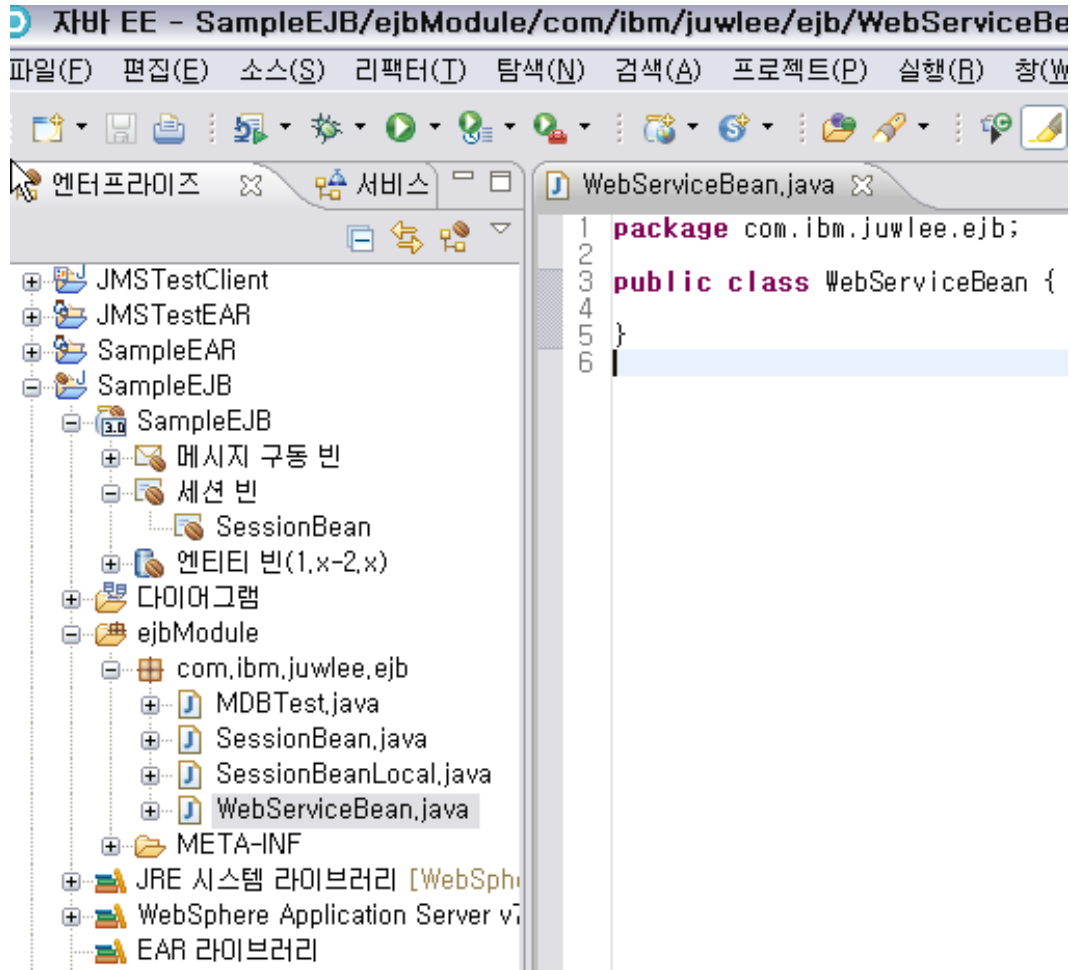
Web Service 어플리케이션을 개발하기 위하여 RAD v7.5 를 시작합니다. 그리고 이전에 작업하였던 SampleEAR 어플리케이션을 조금 수정하여 Web Service 어플리케이션을 작성하도록 하겠습니다. 이미 만들었던 EJB 패키지에서 마우스 우클릭 하여 새로 작성 > 클래스를 선택합니다.



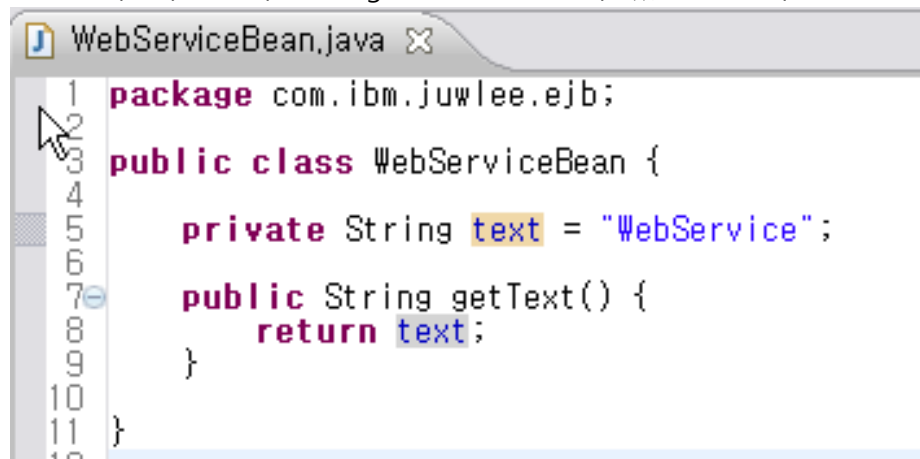
클래스 작성 마법사가 나오면 원하시는 이름을 넣어주고 완료를 선택합니다.



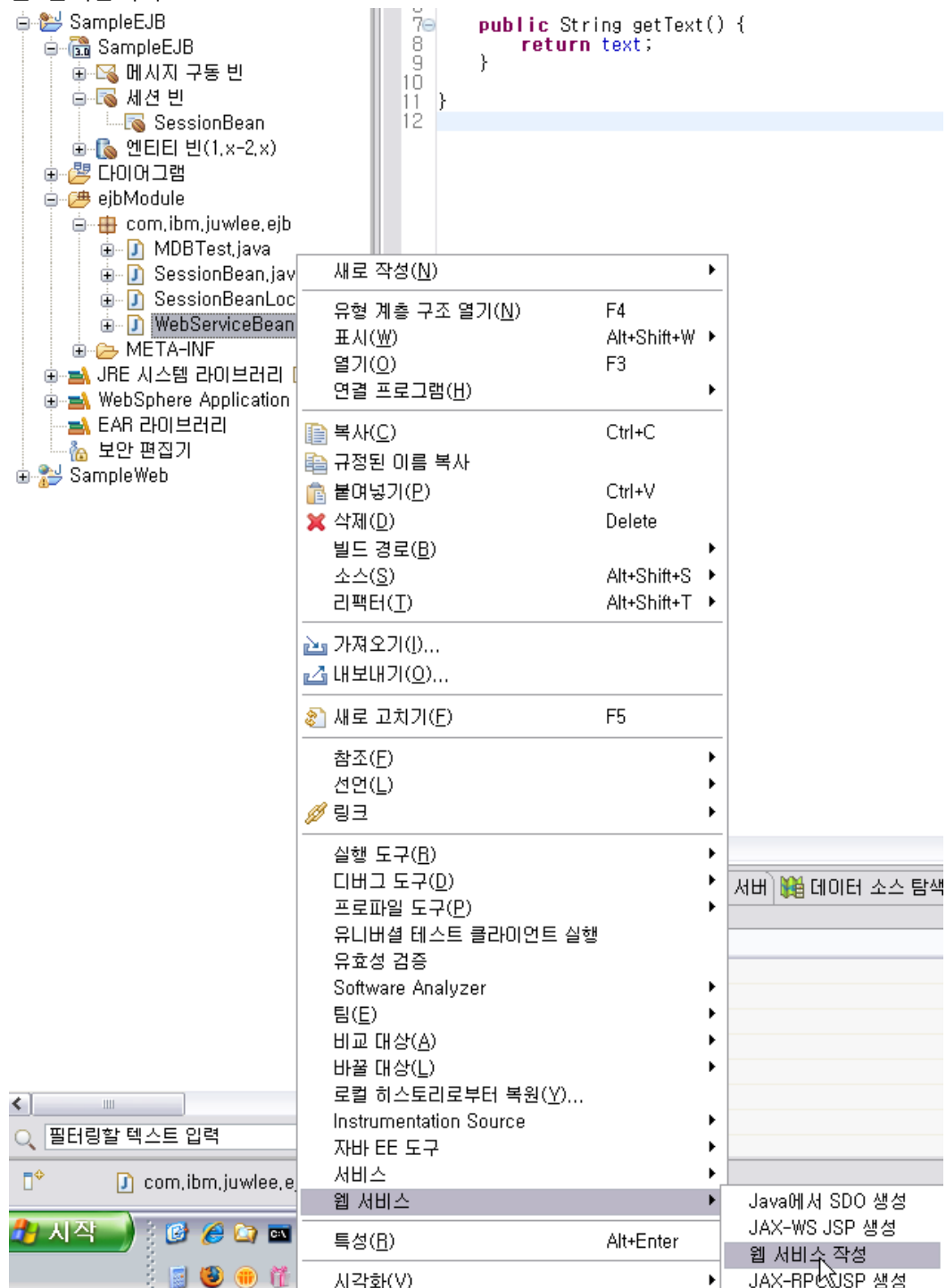
마법사를 완료하면 하단의 그림처럼 새롭게 Java 클래스가 생성된 것을 확인할 수 있습니다.



Java 클래스가 생성되면 String 을 return 받을 수 있는 샘플 메소드를 생성합니다.



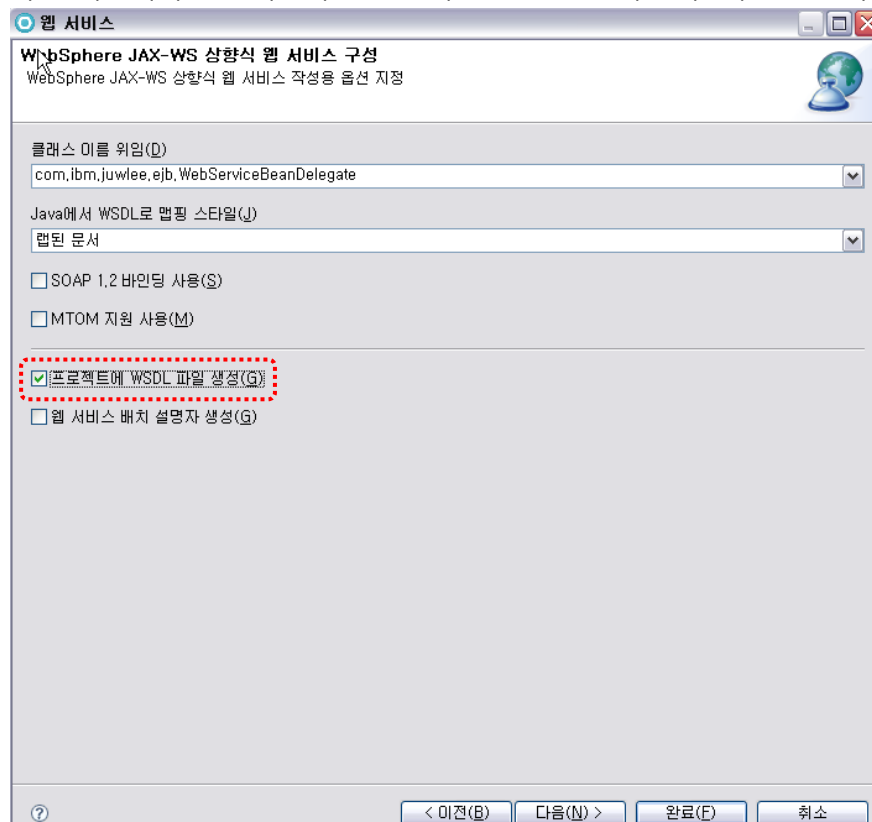
간단한 Java 클래스 생성이 완료되었으면 이제 그 생성된 Java 클래스를 통하여 Web Service 를 생성하도록 하겠습니다. 해당 Java 클래스에서 마우스 우 클릭하여 웹 서비스 > 웹 서비스 생성 을 선택합니다.



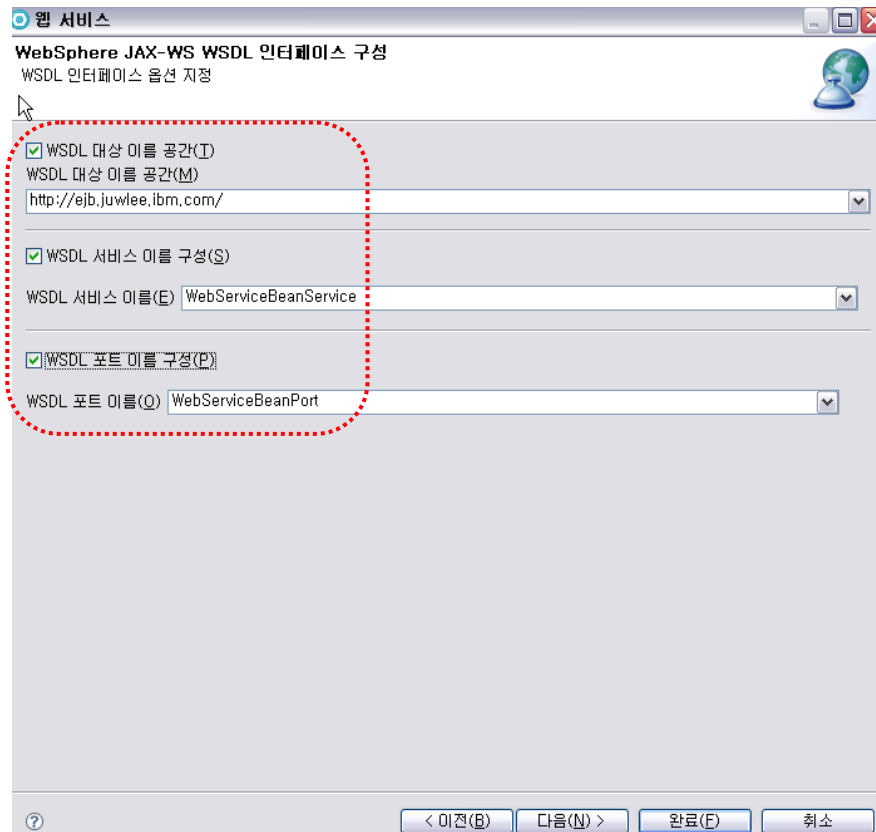
웹 서비스 생성 마법사가 나타나면 구성을 확인하고 서비스 단계를 배치로 맞춘 후 다음을 클릭합니다. (현재 RAD 와 WAS 서버를 연결하지 않았으므로 배치만 한채 파일로 배포할 예정입니다.)



다음 화면에서 프로젝트에 WSDL 파일 생성을 선택한 후 다음을 클릭합니다.



WSDL 인터페이스 구성에서 기본 값을 선택하고 다음을 클릭합니다.



웹 서비스

WebSphere JAX-WS WSDL 인터페이스 구성

WSDL 인터페이스 옵션 지정

☒ WSDL 대상 이름 공간(I)
WSDL 대상 이름 공간(M)

☒ WSDL 서비스 이름 구성(S)
WSDL 서비스 이름(E)

☒ WSDL 포트 이름 구성(P)
WSDL 포트 이름(O)

< 이전(B) 다음(N) > 완료(F) 취소

웹 서비스 공개 부분은 현재 필요하지 않으므로 선택하지 말고 마법사를 완료합니다.



웹 서비스

웹 서비스 공개

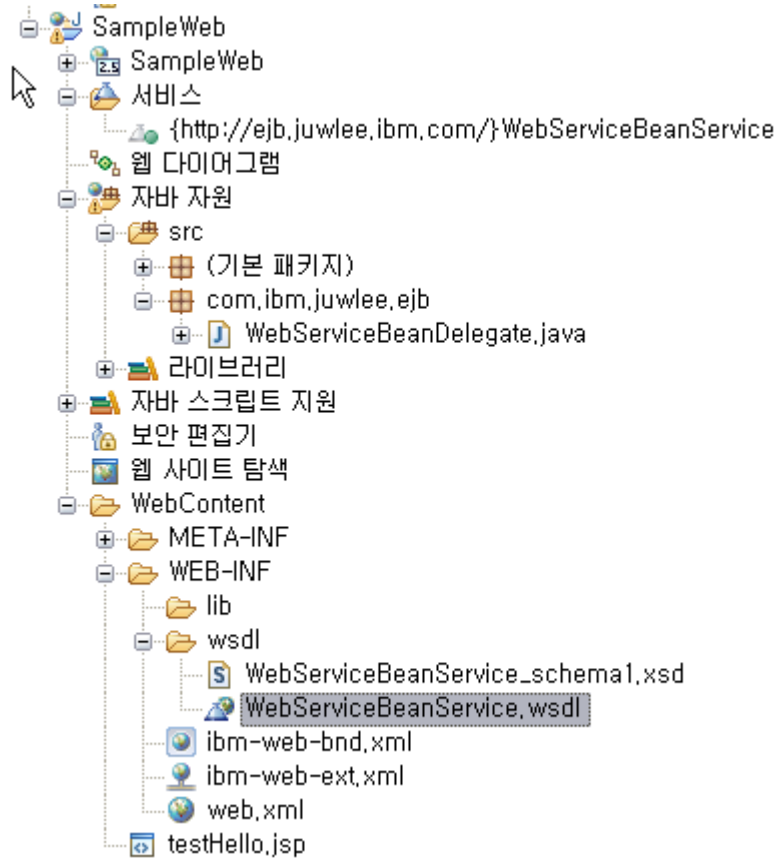
웹 서비스 공개

☐ 웹 서비스 탐색기를 실행하여 단위 테스트 UDDI 레지스트리에 웹 서비스 공개

☐ 웹 서비스 탐색기를 실행하여 UDDI 레지스트리에 웹 서비스 공개(L)

< 이전(B) 다음(N) > 완료(F) 취소

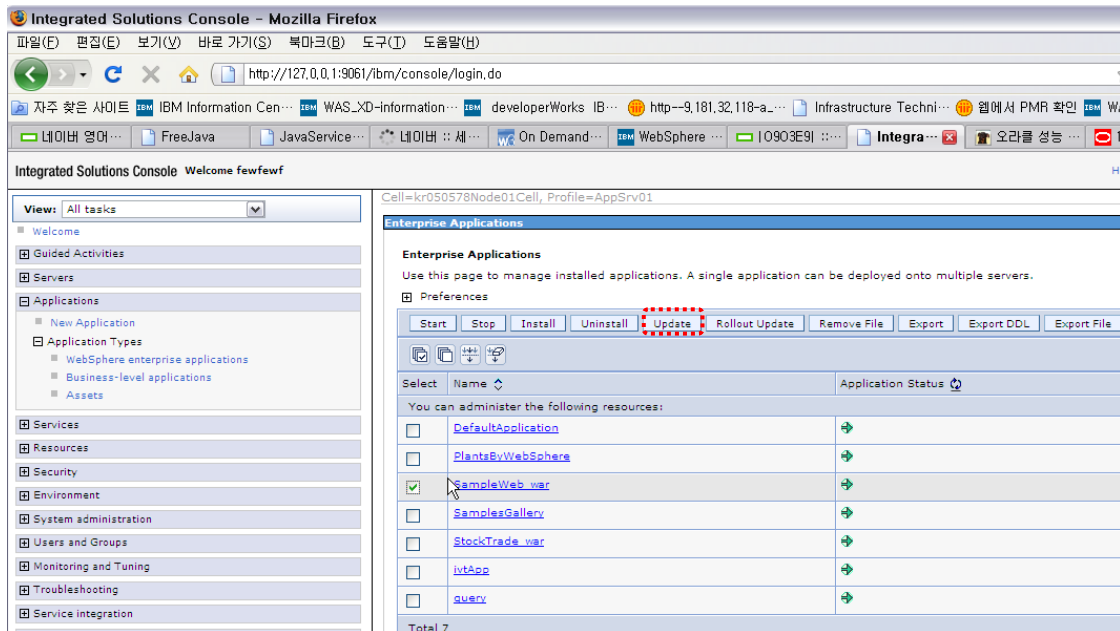
웹 서비스 생성 마법사를 완료하면 RAD 에서 생성된 서비스와 Servlet 파일 wsdl 파일등을 확인 하 실 수 있습니다.



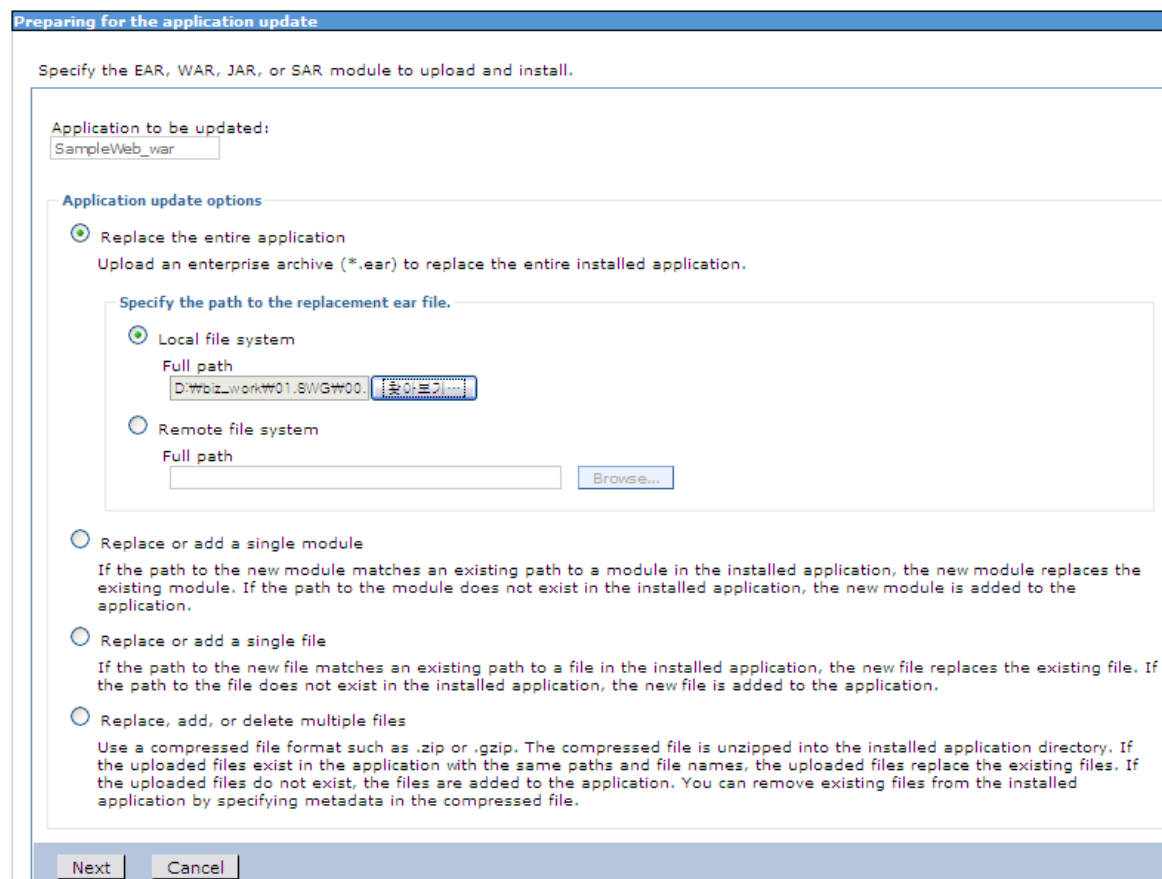
이렇게 하면 Java 클래스를 생성하고 그것을 이용해서 JAX-WS 방식으로 Web Service 생성을 완료하신 것입니다. 이제 실제 Web Service 테스트를 위하여 EAR 파일로 반출하고 WAS 서버에 배포해보는 단계를 진행하도록 하겠습니다.

Part 2. Web Service 어플리케이션 배포

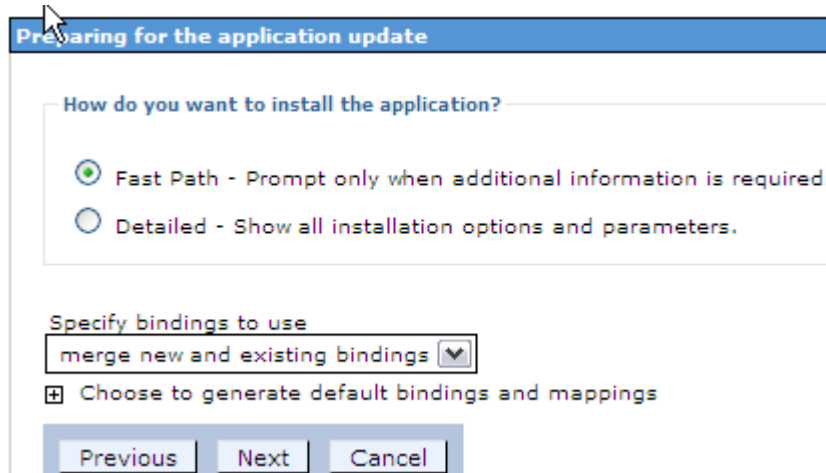
Web Service 어플리케이션 배포도 EAR 로 만들어진 어플리케이션 패키지를 WAS 에 배포하는 방법과 동일합니다. (따라서, 간단하게 서술하겠습니다.) Sample 어플리케이션을 선택하고 update 버튼을 클릭합니다.



마법사에서 업데이트할 EAR 파일을 선택하고 다음을 클릭합니다.



다음으로 간단 설치를 누르고 다음을 클릭합니다.



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.

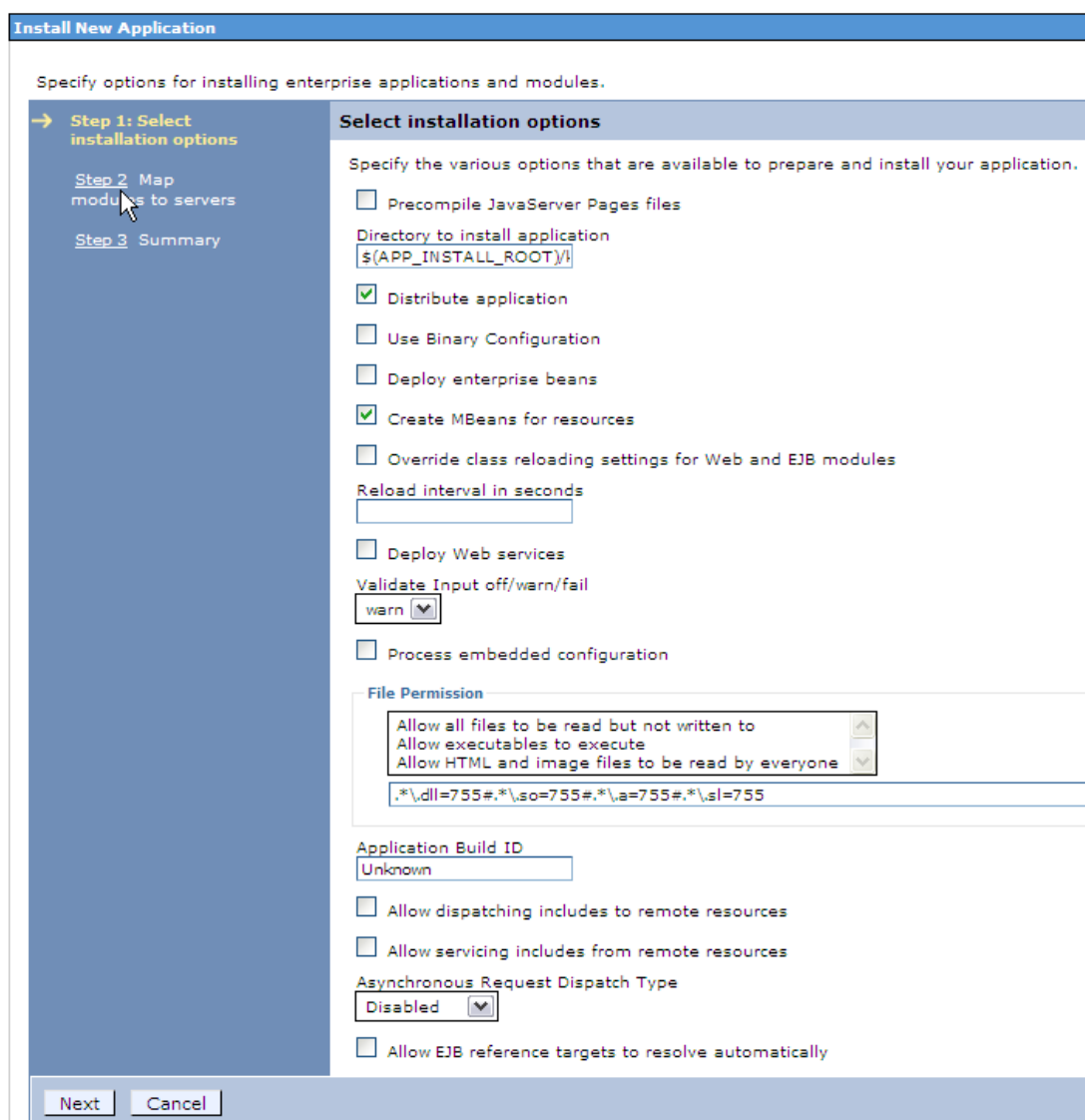
Specify bindings to use

merge new and existing bindings ▼

☒ Choose to generate default bindings and mappings

Previous Next Cancel

설치 마법사의 옵션들을 확인하고 다음을 클릭합니다.



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)/l

☒ 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#.*\sl=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 매핑을 확인하고 다음을 클릭합니다.

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:

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

마지막으로 설치 요약 정보를 확인하고 이상이 없으면 완료를 클릭합니다.

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2 Map modules to servers

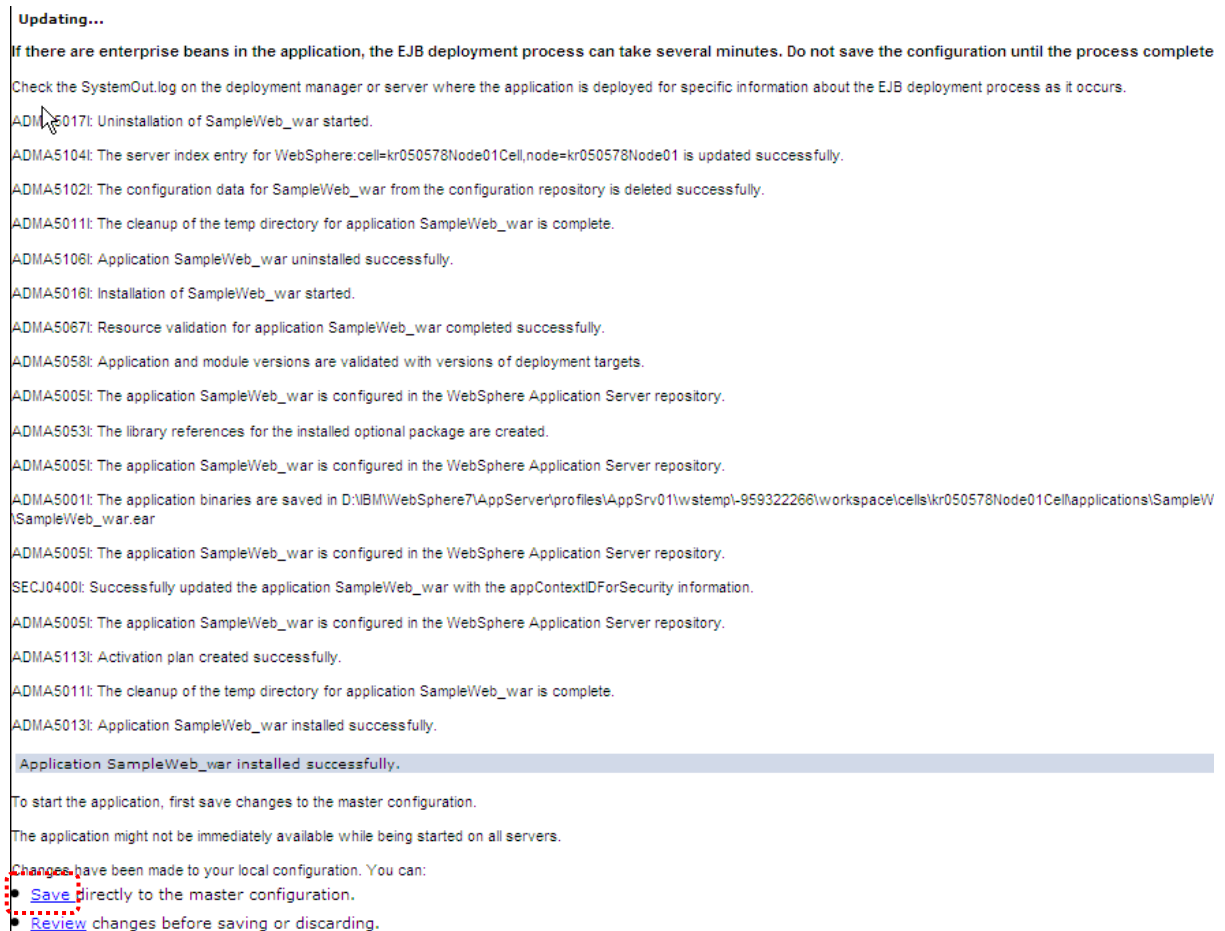
→ Step 3: Summary

Summary

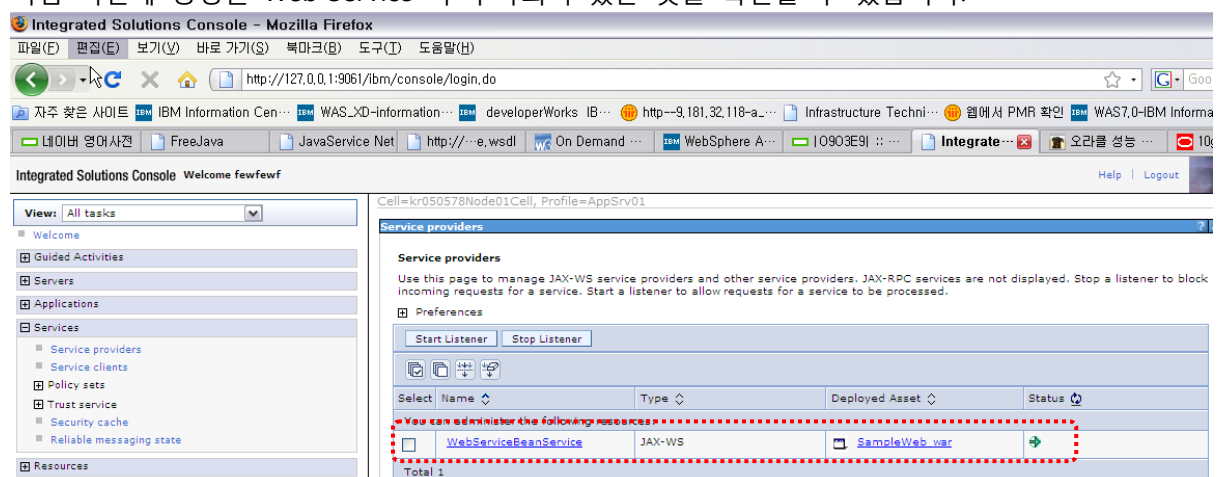
Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	\$(APP_INSTALL_ROOT)/kr050578Node01Cell
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

설치가 진행되는 것을 확인하고 정상적이면 Save 를 클릭하고 종료 합니다.



해당 어플리케이션 설치가 제대로 되었다면 이전처럼 Application 메뉴에서 해당 어플리케이션을 확인할 수 있지만 이에 덧붙여 관리콘솔의 Services > Service providers 메뉴를 클릭하면 하단처럼 이번에 생성된 Web Service 가 추가되어 있는 것을 확인할 수 있습니다.



추가된 Web Service 이름을 클릭하여 세부 정보를 확인해 봅니다.

Service providers

[Service providers](#) > **WebServiceBeanService**

Use this page to manage policy sets and bindings or to access additional information for this service provider.

Configuration

General Properties

Service provider

{http://ejb.juwlee.ibm.com/}WebServiceBeanService

Additional Properties

- WSDL document
- Application: [SampleWeb_war](#)
- Module: [SampleWeb_war](#)

Policy Set Attachments

Attach a policy set to the service, endpoints, or operations. Access the Policy Sharing link to allow clients to acquire the provider policy. Complete the attachment by providing system-specific configuration when you assign the appropriate binding.

Preferences

Attach Policy Set ▼ Detach Policy Set Assign Binding ▼

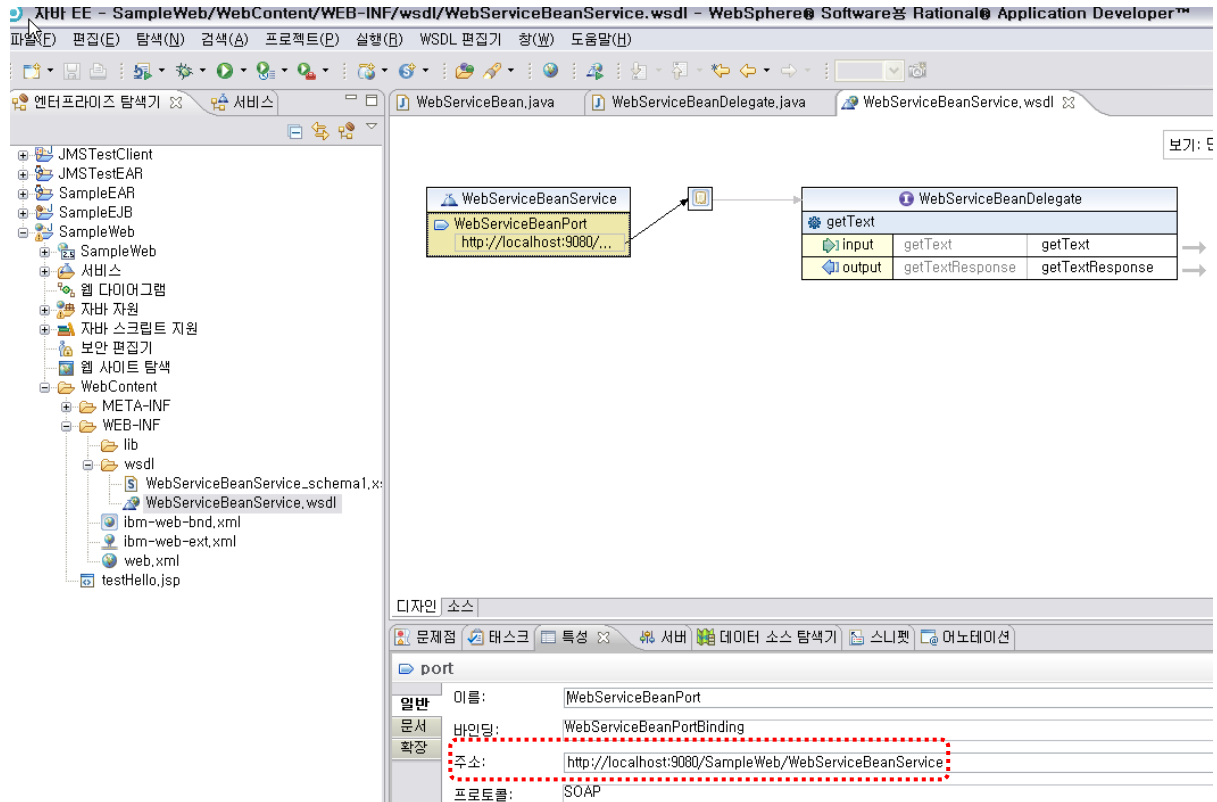
Select	Service/Endpoint/Operation	Attached Policy Set	Binding	Policy Sharing
You can administer the following resources:				
<input type="checkbox"/>	WebServiceBeanService	None	Not applicable	Not applicable
<input type="checkbox"/>	WebServiceBeanPort	None	Not applicable	Not applicable
<input type="checkbox"/>	getText	None	Not applicable	Not applicable
Total 3				

여기까지 오시면 만들었던 Web Service 어플리케이션을 WAS에 정상적으로 배포하신 것입니다.

Part 3. Web Service 어플리케이션 테스트 (SoapUI 활용)

Web Service 어플리케이션을 테스트 하기 위해서는 해당 주소를 알아야하는데 RAD 툴을 이용해서 이전에 만든 WSDL 파일을 보면 하단처럼 확인이 가능합니다.

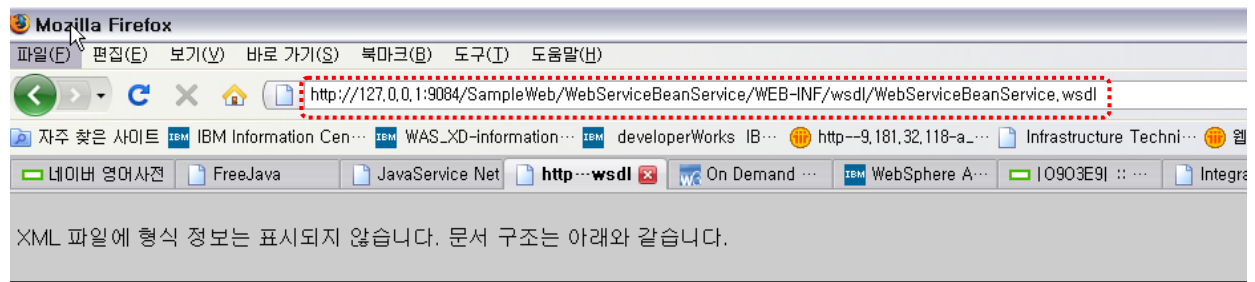
(WSDL 은 어떻게 서비스를 호출해야 하는지에 대한 Interface 이기 때문입니다.)



WSDL 에서 확인한 주소를 웹 브라우저에 입력하여 서비스가 제대로 구동 되었는지 확인합니다.
(단, port 는 실질적으로 서비스가 가능한 port 로 입력해야 합니다.)



위에서 확인한 URL 뒤에 "/wsdl" 을 추가하면 하단 처럼 오픈된 서비스 Interface 인 WSDL을 확인하실 수 있습니다.("/wsdl"을 입력하고 엔터를 누르면 URL이 하단처럼 자동으로 변환됩니다.)



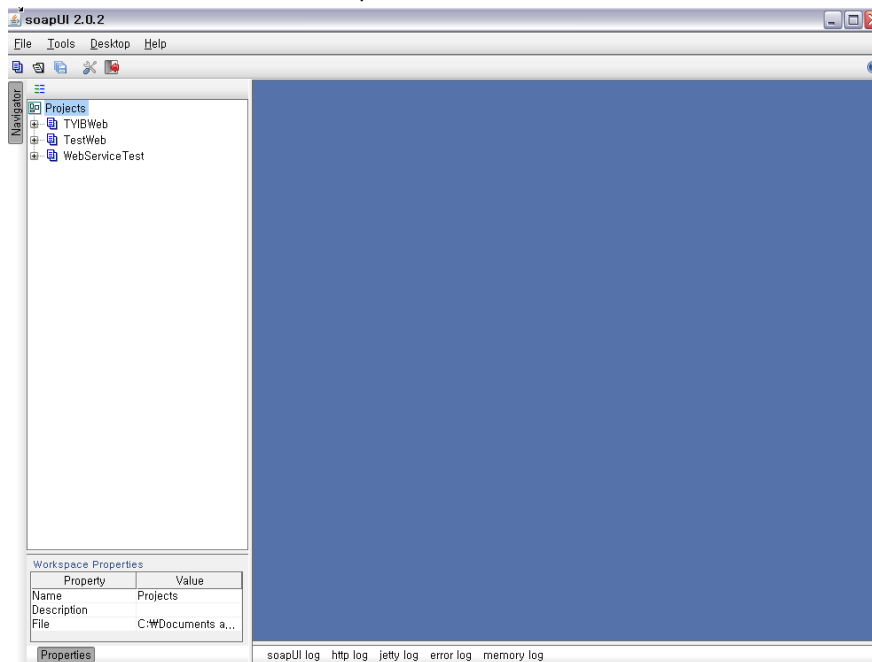
```

- <definitions name="WebServiceBeanService" targetNamespace="http://ejb.juwlee.ibm.com/">
  - <types>
    - <xsd:schema>
      - <xsd:import namespace="http://ejb.juwlee.ibm.com/" schemaLocation="WebServiceBeanService_schema1.xsd"/>
    - </xsd:schema>
  - </types>
  - <message name="getText">
    - <part name="parameters" element="tns:getText" />
  - </message>
  - <message name="getTextResponse">
    - <part name="parameters" element="tns:getTextResponse" />
  - </message>
  - <portType name="WebServiceBeanDelegate">
    - <operation name="getText">
      - <input message="tns:getText" />
      - <output message="tns:getTextResponse" />
    - </operation>
  - </portType>
  - <binding name="WebServiceBeanPortBinding" type="tns:WebServiceBeanDelegate">
    - <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    - <operation name="getText">
      - <soap:operation soapAction="" />
      - <input>
        - <soap:body use="literal"/>
      - </input>
      - <output>
        - <soap:body use="literal"/>
      - </output>
    - </operation>
  - </binding>
  - <service name="WebServiceBeanService">
    - <port name="WebServiceBeanPort" binding="tns:WebServiceBeanPortBinding">
      - <soap:address location="http://127.0.0.1:9084/SampleWeb/WebServiceBeanService"/>
    - </port>
  - </service>
- </definitions>

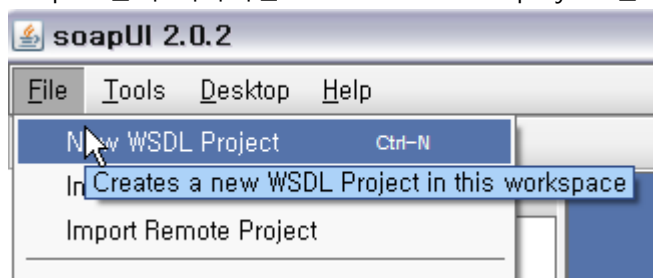
```

WSDL 이 오픈되는 것을 확인했으면 실제 Web Service 를 테스트 해보기 위하여 SoapUI 툴을 실행합니다. 이전에 설명드렸지만 Web Service 의 장점은 WSDL 로 Interface 를 오픈하고 SOAP 프로토콜을 통해서 표준 XML 메시지를 이용하여 원하는 서비스를 이용할 수 있는 것입니다. 즉 SOAP 메시지를 전달할 수 있는 무언가가 필요한데 간단한 테스트를 위해서 freeware인 SoapUI 툴을 활용하는 것입니다. 따라서 SOAP 메시지를 전달할 수 있는 툴이면 Web Service 테스트에는 무리가 없습니다.

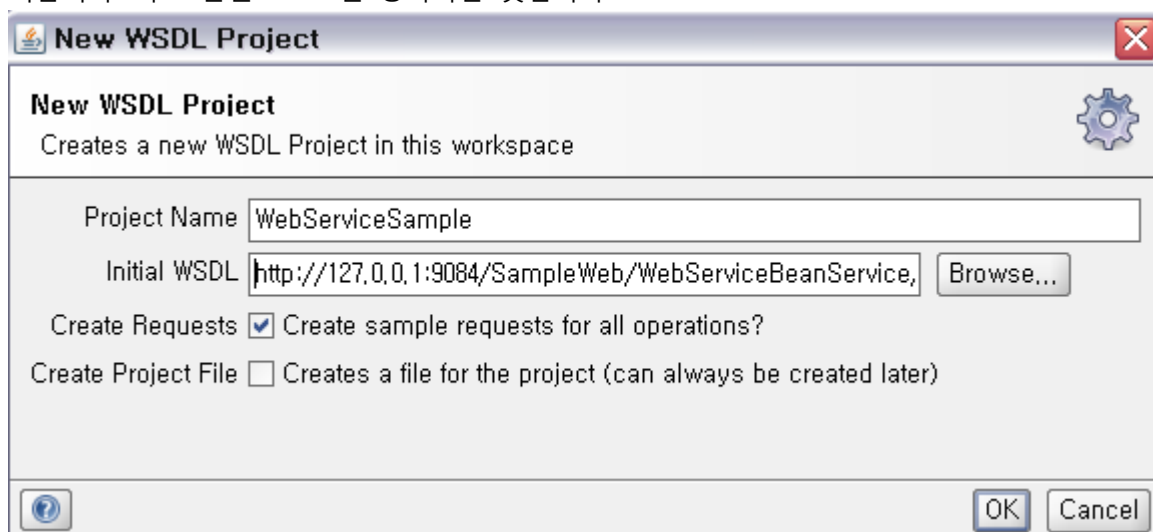
이번 강좌에서는 간단한 SoapUI 툴을 실행합니다.



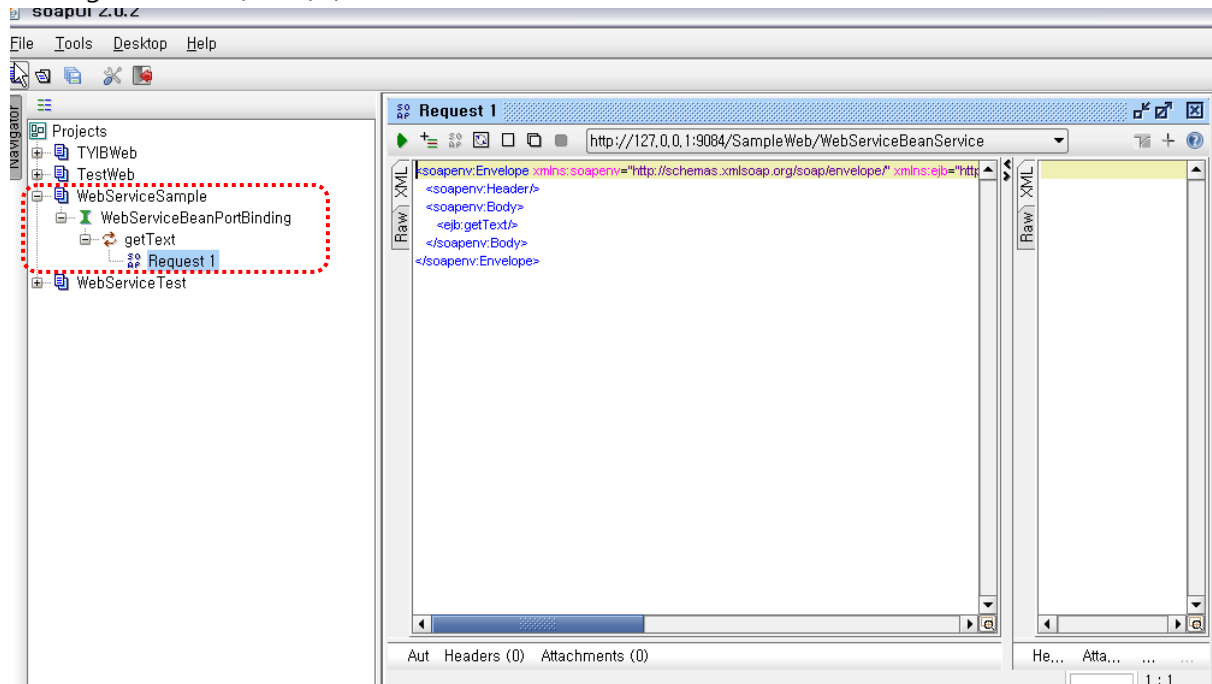
SoapUI 툴이 시작되면 File > New WSDL project 를 선택합니다.



New WSDL Project 마법사가 나오면 프로젝트 이름을 적고 위에서 WSDL 을 호출했던 주소를 입력합니다. 즉 오픈된 WSDL을 등록하는 것입니다.

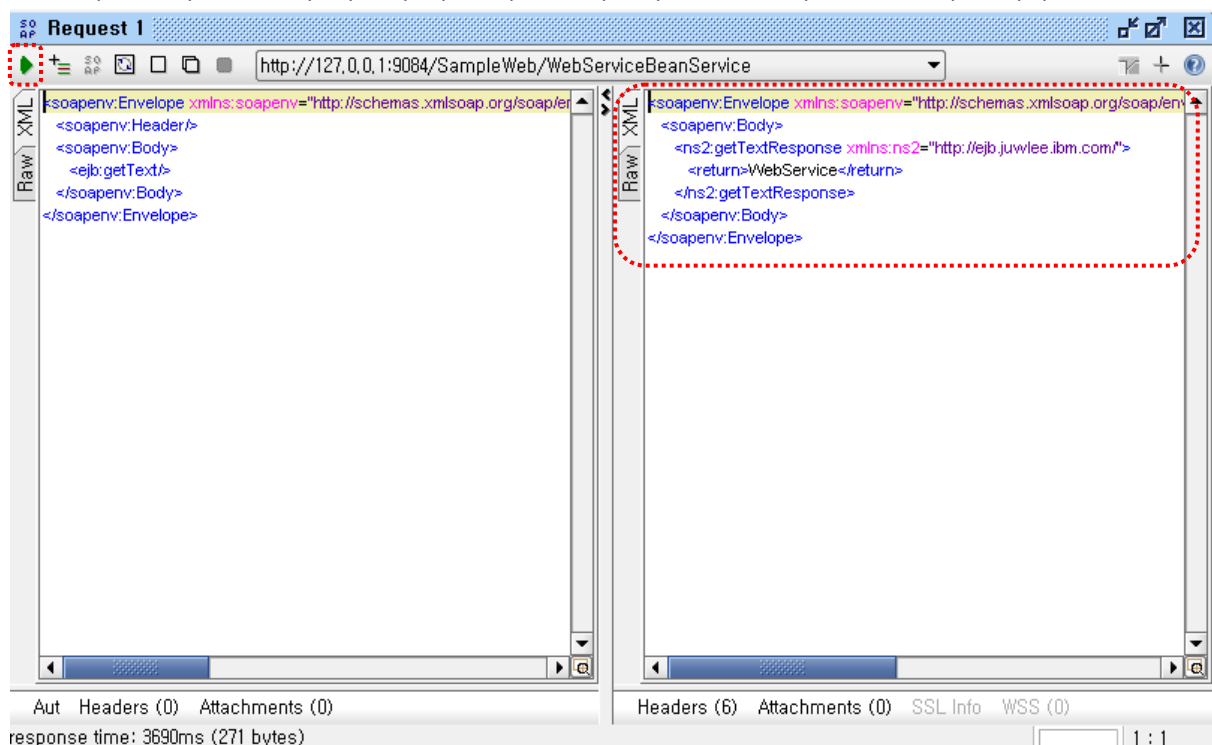


이렇게 하면 SoapUI 툴이 WSDL 을 해석하여 자동으로 request 를 보낼 수 있는 XML 형식의 Message 를 만들어 줍니다.



해당 request 의 왼쪽 위의 녹색 삼각형이 Send 버튼입니다. 이를 클릭하여 실제로 SOAP 메시지를 이용하여 Web Service 를 호출합니다.

호출에 성공하면 오른쪽 화면에 하단 처럼 결과값이 SOAP 형태로 반환받게 됩니다.



여기까지 잘 따라오셨다면 늘 얘기하는 것이지만 Web Service 생성과 배포, 그리고 테스트를 잘 완료하신 것 입니다. 수고하셨습니다. 지금 까지 간단하게 하나씩 쉽게 따라 해보는 IBM WAS v7 시리즈 강의를 진행하였습니다. 일하는 틈틈이 만드느라 시간은 오래 걸렸지만 이 자료를 가지고 IBM WAS v7 에 쉽게 다가갈 수 있는 기회가 될 수 있다면 정말로 좋을 것 같습니다. 여기까지 따라와 주신 분들 모두 수고 많으셨고.... 이제 IBM WAS v7 의 대략적인 기능들을 사용하실 수 있으니 좀 더 고급 레벨로 고고씽 해보시길 바라겠습니다. 그럼 IBM WAS v7 참 쉽조라는 말을 할 수 있기를 바라며 이만 줄이도록 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

참고 2) Information Center – Web Service

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

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