How-To Build and Use Web Services with JDeveloper

Web Services provide client neutral access to data and other services. JDeveloper allows you to create different types of Web Services quickly and easily....

In this tutorial, you will create 4 different Web Services: a POJO Annotation-Driven service, a Declaratively-Driven POJO service, a service for existing WSDL, and an EJB service. The focus of these scenarios is to demonstrate and test Java EE web services. In particular this means JAX-WS (Java API for XML Web Services) and annotation handling. JAX-WS enables you to enter annotations directly into the Java source without the need for a separate XML deployment descriptor.

At the end of the tutorial you create an ADF Client application that consumes the web services you created.

Purpose	Duration	Application
This tutorial shows you how to build and consume Web Services. The tutorial shows several end-to-end scenarios for creating web services. After you develop several web services, you create a client application that uses those services.	4 hours	■ Download
To see the complete application you will create, click the Download button to download a zip of the solution application, and then unzip it in a workspace folder of your choice.		

Part 1: Building a POJO Annotation-Driven Service

In this first part of the tutorial, you install the required lab files, start JDeveloper, and open the startup application and project.

Step 1: Getting Ready

- 1. Download the <u>lab starter files</u> and save the webservice.zip file in a temporary folder (such as d:\Temp.)
- 2. Using WinZip of whatever zip utility you have, unzip the webservice zip into a folder of you choice. In this tutorial, we used C:\JDeveloper\mywork.
- 3. Start JDeveloper by selecting Start > Programs > <JDEVELOPER_HOME> > OracleHome > Oracle JDeveloper Studio > Oracle JDeveloper Studio

If a dialog box opens asking if you would like to import preferences from a previous JDeveloper installation, click **NO**.

4. If prompted for a Role, select Studio Developer.

🕐 Select Role 🛛 💌
Select the role that matches your requirements. You can also change roles using the Roles page in preferences.
Role:
Studio Developer (All Features) Includes all features.
Customization Developer Configures the product for customizing metadata.
Database Developer Includes only features for core database development.
Java Developer Includes only features for core Java development.
Java EE Developer Includes only features for core Java EE development.
Always prompt for role selection on startup
OK Cancel

- 5. If the **Tip of the Day** window opens, click **Close**.
 6. You should now see the JDeveloper IDE. Close the **Start page** by hovering your mouse over the tab and clicking the **X** on the tab.

Oracle JDeveloper 12c					
Eile Edit View Application Refactor ● ● ● ● ● ●	Search Navigate Build	<u>R</u> un Tea <u>m</u> ▷ - ② -	<u>T</u> ools <u>W</u> ind	low <u>H</u> elp	
Applications ×	3 Start Page ×				
New Application			LOPER		
Open Application		Learn &	Explore		Get Started
		Mb = 5 ¹ = 11=			
	C.	what's new		Featured Tu	ıtorials
	F	Release Notes		Getting Starte	ed with the JDeveloper IDE
	S	Samples & Demo	15	Developing Ri	ich Web Applications with Oracle ADF
				Building and U	Jsing Web Services

7. Select the **Applications** window tab and click **Open Application** (alternatively, you can select **File > Open**)



8. In the **Open Application** dialog box, locate the **Web Service** folder where you unzipped the WebService.zip file and select **WebService.jws**.

🕜 Open Appli	cation(s)		X
Location:	C:\JDeveloper\mywork\webservice		
Â	Annotation		
Work	WebService.jws		
Home			
Desktop			
Documents			
	File Name: WebService.jws		
8	File Type: Application files (*.jws)		•
Help		Open Car	icel

9. Click Open

10. If your are prompted to migrate the application, click **Yes**.

The Applications window should look like this:

Applications ×
🔁 WebService 🔹 💌
I Projects Q Q - Y - E -
- Application Resources
A Data Controls
I Data Controis
+ Recent Files

Step 2: Adding a Plain Old Java Object (POJO) to contain a Web Service Method

In this section you start with a project that contains plain old Java classes and add an annotated method that you publish as a web service.

Web service annotation is a feature of Java EE 6 which takes complexity out of creating and deploying Web Services. Web service annotation allows you to define web services from within a POJO. This feature of Java EE eliminates the need for complex configuration of the web service and the web server. Java EE introspects the deployed classes a creates the web server configuration on-the-fly. This frees up the developer to concentrate more on the service rather than the tedious details of deployment.

- 1. In the Applications window, expand the Annotation project nodes to show the POJO classes:
 - o Dept.java describes the department structure
 - Emp. java describes the employee structure
 - o MyCompany.java populates information about departments and employees

Applications			×
🔁 WebService			• •
Projects	Q 82 -	7-	<u>-</u> -
Annotation			
Application Resources			
± Data Controls			
		51000000000	*

2. In the Applications window, double-click **MyCompany.java** to edit it.

Applications		×
🔁 WebService		
Projects	Q & - 7-	3 <u>5</u> -
Annotation	I	
± Application Resources		000000000000
± Data Controls		
± Recent Files		\$

3. Add an **@WebService** annotation after the import statements. The IDE will prompt you to select the import for the WebService class. Select **javax.jws.WebService** from the popup. This annotation denotes that the class contains a method to be used by a web service.

? st	Start Page 🛛 📴 <i>MyCompany.java</i> 🐣								
Q.= (Find	2	1	A	{}	423	5	4	5
	package annotation;								
	<pre>import java.util.ArrayList; import java.util.Collection;</pre>								
	<pre>import javax.jws.WebService;</pre>								
?]WebService								
	<pre>public class MyCompany { public MyCompany() { Collection<dept> depts = Collection<emp> myEmp10 =</emp></dept></pre>	ne = 1	ew I new	rra Arr	yList ayLis	: <dej st<ei< th=""><th>pt>() mp>()</th><th>););</th><th></th></ei<></dej 	pt>() mp>()););	

4. In the margin of the editor, click **Quick Hint** (light bulb icon) and select the **Configure project for web services** option.

? s	itart Page 🗡 📴 <i>MyCompany.java</i> 🐣
Q,-	Find 🔁 🌽 🦧 🖓 👼 🗟 📮 📮 🕽 🐘 🏠 🛅 🚟
	package annotation;
	<pre>import java.util.ArrayList;</pre>
	<pre>import java.util.Collection;</pre>
	<pre>import javax.jws.WebService;</pre>
?	@WebService
4	Configure web.xml for this web service
4	Suppress "Add servlet entry for this webservice class" By JDeveloper Name (SuppressWarnings Annotation)
4	Configure Project for Web Services
4	Suppress "Annotation Project Configuration" By JDeveloper Name (SuppressWarnings Annotation)
	<pre>Emp a = new Emp(1, "Larry"); myEmp10.add(a); a = new Emp(2, "Ken"); myEmp10.add(a);</pre>

5. In the **Select Deployment Platform** dialog box, ensure that **Java EE 6**, with support for JAX-WS Annotations is selected.



 Click OK. This step adds the javax.jws.WebService import statement to the Java class if it is not already there and creates a web.xml file. The Applications window should look like the following:

Notice that the icon for MyCompany.java class is changed to represent a WebService class, and the web.xml file has been added to your project.



- 7. Click Save All to save your work.
- 8. In the Code Editor, scroll to the bottom of the class and add the following code statements:

```
public Dept getDeptInfo (int id) {
    for (Dept a: this. getMyDepts() ) {
        if (a.getId() == id) {
            return a;
        }
    }
    return null;
}
```

This loop returns information about all employees working in a specific department. The code in the editor window should look like:



9. Create a second annotation before the getDeptInfo() method. The annotation signifies this is the method to be exposed from the web service. Add a blank line above the getDeptInfo() method, and start typing @webMethod. Code insight pops up up a list of available syntaxes. Select webMethod from the list.

```
public Collection<Dept> getMyDepts() {
     return myDepts;
           }
    public boolean addEmployeeToDept(Emp emp, int deptid) {
               //TODO write some logic here
               System.out.println("Here we'll be adding an employee to " +deptid);
               return true;
      Select import for WebMethod... (Alt-Enter)
 0
      @WebMethod
          public Dept getDeptInfo (int id) {
    for (Dept a: this. getMyDepts() ) {
    if (a.getId() == id) {
           return a;
           }
           }
           return null;
           3
      }
  MyCompany -> getDeptInfo(int) >>
Source History
```

10. If suggested, press [Alt]+[Enter] to add the **import javax.jws.WebMethod**; statement (although this statement may be added automatically.)

The class should now look like the following:



- 11. Click Save All to save your work.
- 12. You can use the Properties window to modify the characteristics of the class. In the menu bar, select **Window > Properties** and it will open as a tab in the bottom portion of the IDE. Note: If the Properties window opens in a different part of the IDE, you can drag its tab and drop it on the bottom panel if you would rather work with it there.



13. To display the properties of the MyCompany class in the **Properties** window, select the **Source** tab at the bottom of the Structure window, then select the top level **MyCompany** class name.

MyCompany.java - Structure ×	MyCompany - Propertie	s × Messages - Log	
🖓 🖥 📰 🔝 🖕 🛳	Q Find		?
annotation	± Java		-
	Policies		_
Same and the second sec	🗆 JAX-WS		_
	···· Web Services ·······		
getMyDepts() : Collection <dept></dept>	Name:	MyCompany	
	Service Name:	MyCompanyService	
······{123 ⊖ myDepts : Collection <dept></dept>	Port Name:	MyCompanyPort	
	Endpoint Interface:		
	WSDL Location:		
	Target Namespace:	http://annotation/	
	··· Binding ·····		
Severe Design	Binding Type:	<default>SOAP 11 Binding</default>	
Source Design	Descurrent Obular	cdefendes Demonstration	-

- The Properties window displays a few expandable nodes. Expand the JAX-WS node and notice that the Service Name has the word 'Service' appended to the class name.
 Change the Service Name to MyCompanyWS. Notice that the class reflects the name change.

MyCompany - Propertie	es 🐣 Messages - Log	
Q Find		?
🗄 Java		
Policies		
🗆 JAX-WS		
Web Services		
Name:	MyCompany	
Service Name:	MyCompanyWS	
Port Name:	MyCompanyPort	
Endpoint Interface:		
WSDL Location:		
Target Namespace:	http://annotation/	
Binding		
Binding Type:	<default>SOAP 11 Binding</default>	-
Designed Chiles	cale for the Decomposit Ob the	

16. Click Save All I to save your work.

You have now created a POJO Web Service. In this next section, you will test you Web Service.

Step 3: Testing a Web Service

In this section you compile, deploy and test the web service using the HTTP Analyzer. JDeveloper includes a web service testing mechanism called the HTTP Analyzer. When you use the HTTP analyzer to test web services, JDeveloper compiles and deploys the service to the integrated web server. It then invokes the analyzer, allowing you to send and receive values from the web service.

Before testing the web service, check that your web browser settings are correct. Select Tools >
 Preferences and then scroll down the list on the left to select the Web Browser and Proxy
 page. On the Proxy Settings tab, ensure that the No Proxy is selected, then click OK.

Search	Web Browser and Proxy
Http Analyzer	Web Browsers Proxy Settings Internet Files
···· Issues	
···· JavaScript Editor	
···· JSP and HTML Visual Editor	Use System Default Proxy Settings
Maven	 Use <u>A</u>utomatic Configuration Script
···· Merge	Script:
··· Mouseover Popups	O Manual Davies California
··· News	O Manual Proxy Settings
··· Oracle Cloud	Host: Port: 80
··· Profiler	No Proxy for: 0.1[[::1]]JGALLUS-LAP.st-users.us.orade.com[JGALLUS-LAP]192.168.
··· Resource Bundle	Example: localbastik aracle comiti27.0.0.1
Run	Example, localitosi, toracle.com/127.0.0.1
··· Shortcut Keys	Provy Server Requires Authentication
··· Swing GUI Builder	
···· Task Tags	User Name:
···· TopLink	Pass <u>w</u> ord:
··· UML	
··· Usage Reporting	Test Proxy
···· Versioning	
Web Browser and Proxy	
··· WS Policy Store	
XML Schemas	

2. In the Applications window, **right-click** the MyCompany.java node and in the context menu, select **Test Web Service**.

🕐 Oracle JDeveloper 12c - WebService.jws : Annotation.jpr				
<u>File Edit View Application</u>	Refa <u>c</u> tor <u>S</u> earch <u>N</u> avigate <u>B</u> uild <u>R</u> un Tea <u>m T</u> ools <u>M</u>	<u>V</u> indow <u>H</u> elp		
💁 - 🔁 🗒 🕄 👒 🕅	🔾 - 🛇 - 🔒 📥 🕍 🕌 🕨 - 🀲 -	Q- Search		
Applications	× 🖸 🕐 Start Page × ট MyCompany.java			
	Image: Second	<pre>epts(Collection<dept> myDe myDepts; Dept> getMyDepts() { ;; }</dept></pre>		
'⊡ ⊡⊡ WEB-INF :	Open Delete Exclude Project Content Deploy	EmployeeToDept(Emp emp, i) ome logic here ntln("Here we'll be adding		
	Find Usages Ctrl+Alt-U Image: Make Ctrl+Shift-F9 Image: Rebuild Alt+Shift-F9	64:11		
MyCompany.java - Structure	▶ <u>R</u> un 攀 Debug	3		
annotation 	Reformat Alt+Shift-F Organize Imports Ctrl+Alt-O Refactor Imports			
WyCompany() WyCompany	Compare Wit <u>h</u>			
	Create RESTful Ser <u>v</u> ice Show WSDL for We <u>b</u> Service Annotations Gener <u>a</u> te WSDL and Add to Project Create Client for Web <u>S</u> ervice Annotations <u>T</u> est Web Service			
	Create Data Control			
Source Design	web service Properties	1		
C:\JDeveloper\mywork\webservice\Ar	notation\src\annotation\MyCompany.java			

This option invokes the integrated WebLogic Server, deploys the service, and then starts the analyzer. It may take a few seconds to start WebLogic Server if you are running it for the first time. If this is the first time you test a service, Windows may ask you about blocking content. Allow the content to be displayed.

3. The top portion of the **HTTP Analyzer** editor window displays the URL for the web service, the WSDL URL, and the exposed Operations. Select the MyCompanyPort.getDeptInfo(,) operation from the list.

🕄 Start Page 🔺 ট MyCom	npany.java 🛛 📃 HTTP Analyzer :	Unsent Mes	ssage ×	
🖩 🗿 🔺 🔻 🖂 🥔 🖩	L == 00			
URL: http://localhost:	7101/WebService-Annotation-contex	kt-root/MyC	CompanyPort	•
WSDL URL: http://localhost:	7101/WebService-Annotation-contex	ct-root/MyC	CompanyPort?WSDL	Select WSDL
Operations: MyCompanyPor	t.getDeptInfo(,) Credential	s: <no ci<="" td=""><td>redential></td><td><u>N</u>ew</td></no>	redential>	<u>N</u> ew
Request H SOAP Hea arg0 : in Send R	WS panyPort IEmployeeToDept(,) Deptinfo(,) MyDepts(,) MyDepts(,)	<u>×</u> ;	Response HTTP Headers Request is being edited	

The bottom portion of the analyzer is split into two areas: Request and Response. The request area shows all the arguments from the exposed method (in this case, only one argument.) When the web service is executed, the Response area shows the results.

Request HTTP Headers	+ - X
🗆 parameters	
arg0 : int	
Send Request	
SOAP Structure HTTP Content REST Structure Hex	Content Raw Messag

4. In the Request area, enter a department number value (10, 20 or 30) in the arg0 field.

⊞ Request HTTP Heade ■	ers	+ - X	
± SOAP Headers			Re
parameters	1		-
arg0 : int 20			
Send Request	🧳 <u>C</u> lear Request		
SOAP Structure HTTP Con	tent REST Structure Hex	Content Raw Messag	ge

5. In the toolbar area of the analyzer, click **Send Request**, or click the Send Request

button	🔀 Send Request	below the argument.
0 010011		bolon alganiona

Request HTTP	Headers	- + - ×
SOAP Headers		
parameters		
arg0 : int	20	
Cond Deque	st 🛛 🥔 Clear Request	

6. The analyzer sends the request to the service, returning after a few seconds the information about employees working in the specified department.

🗄 Request HTTP Headers 🛛 🕂 👻	Response HTTP Head	ders 🛷 200 OK	1
SOAP Headers	parameters		_
parameters	🖃 return		-
arg0 : int 20	employees : Arr	ay	
	id	3	
🖂 Send Request 🧳 Clear Request	name	Gary	
	salary	0.0	
	employees		_
	id	4	
	name	Jeff	
	salary	0.0	
SOAP Structure HTTP Content REST Structure Hex Content Raw Message	🖃 employees		-

7. Click the HTTP Content tab at the bottom of the editor to look at the xml code.

🗈 Request HTTP Headers 🛛 🖶 👻	Response HTTP Headers
<pre><?xml version = '1.0' encoding = 'UTF-8'?> <env:envelope <s:body="" http:="" schemas.xmlsoap.org="" soa;="" xmlns:env="http://schemas.xmlsoap.org/s</td><td><pre><?xml version = '1.0' encoding = 'UTF-8'?> <S:Envelope xmlns:S="></env:envelope></pre>	
	<name>Jeff</name> <salary>0.0</salary>
SOAD Structure HTTP Content DEST Structure Hav Content Daw Massage	*(C) +

8. Click the **Raw Message** tab at the bottom of the editor for another presentation of the code.



9. Click the **SOAP Structure** tab at the bottom of the editor, and then in the top part of the HTTP Analyzer, click the **WSDL URL** link.

?) Start Page × B MyCompany.java × B Emp.java × III HTTP Ana Image: The start Page × I	lyzer : 29 ×		ė	-
URL: http://localhost:7101/WebService-Annotation-context-root/MyC	ompanyPort		•	-
WSDL URL: http://localhost:7101/WebService-Annotation-context-root/MyC	ompanyPort?WSDL		Select WSDL	
Operations: MyCd http://localhost:7101/WebService-Annotation-contex WSDL	d-root/MyCompanyPort?	-	<u>N</u> ew	
🗄 Request HTTP Headers 🛛 👘 👻	Response HTTP Head	lers	🖋 200 OK	-
∃ SOAP Headers	parameters			
□ parameters	return			
arg0 : int 20	employees : Arra	ау		
	employees			
	id	3		
🖂 Send Request 🧳 Qlear Request	name	Gary		
	salary	0.0		
	employees			-
	id	4		
	name	Jeff		
	salary	0.0		
SOAP Structure HTTP Content REST Structure Hex Content Raw Message	🖃 employees			v

10. This opens the visual editor for the web service. In the Port Types panel, expand the **getDeptInfo** > output > getDeptInfoResponse nodes.

Port Types	🕂 💥 🕞 Bindings / Partner	Link Types 🛉 👻 🖂 Service	s 🌵 🕽
MyCompany MyCompany AddEmployeeToDept SetMyDepts GetDeptInfo GetUptInfoResponse GetUp	oResponse parameters	PortBinding Image PortBinding Image PortBinding Image PortBinding Image PortBinding Image PortBinding	yCompanyWS ∳ MyCompanyPort

11. To the left of the Port Types panel, click the small **Plus** sign at the top of **Messages** to show message contents.



A new graphical representation shows the flow for any message you select.

Message Messages Messages	Port Types	Bindings / Partner Link Types MyCompanyPortBinding Soap:body - document ddEmployeeToDept setMyDepts getDeptInfo getMyDepts
getMyDeptsResponse	5	



12. Right-click any tab in the editor window and select the **Close All** option.

13. Collapse the Annotation project node in the Applications window.



Courtesy: https://docs.oracle.com/cd/E53569_01/tutorials/tut_web_services/tut_web_services.html

Modified: 2021.10.14.7.34.PM

Dököll Solutions, Inc.