

Oracle® Java CAPS Database Binding Component Tutorial

License Restrictions Warranty/Consequential Damages Disclaimer

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

Warranty Disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

Hazardous Applications Notice

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group in the United States and other countries.

Third Party Content, Products, and Services Disclaimer

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Using the Database Binding Component	5
Tutorial Plan	6
Database Binding Component Project in a Nutshell	7
Starting the GlassFish V2 Application Server	10
▼ To Start the GlassFish V2 Application Server From NetBeans IDE	10
Creating a BPEL Module Project For Table Type Operations	11
▼ To Create a BPEL Module Project	12
Connecting to a MySQL Database	14
▼ To Connect to a MySQL Database	14
Starting the GlassFish Admin Console	18
▼ To Start the Admin Console	18
Setting Up Database Access	19
Integrating a JDBC Driver	19
Creating a JDBC Connection Pool	20
▼ To Create a JDBC Connection Pool	20
▼ To Create a JDBC Resource	23
Creating a WSDL Document For Type : DATABASE	24
▼ To Create a WSDL Document : dbWSDL	25
▼ To Select the Database Connection	27
▼ To Edit the SQL Code	30
Source View	33
Creating a WSDL Document For Type SOAP	34
▼ To Create a WSDL Document : SOAPWSDL	34
Creating a BPEL Process	41
▼ To Create a BPEL Process	42
▼ To Add a Partner Link	44
▼ To Add a Web Service and Basic Activities	45
▼ To Edit Web Service : Receive1	49

▼ To Edit the Web Service : Invoke1	51
▼ To Edit the Web Service : Reply1	55
▼ To Edit the Basic Activities : Assign1	57
▼ To Edit the Basic Activities : Assign2	59
Validating BPEL	61
▼ To Invoke Explicit Validation	61
Design View : Notifications	62
The Design View	62
Creating the Composite Application Project	64
▼ To Create the Composite Application Project	64
Deploying and Testing the Composite Application	72
▼ To Deploy the Composite Application	72
▼ To Test the Composite Application	74
Debug the Test Case	78
Creating a BPEL Module Process Using Prepared Statements	79
Creating a BPEL Module Project Using Procedures	82
Creating a BPEL Module Project Using SQL File	84

Using the Database Binding Component

The Database BC provides a comprehensive solution for configuring and connecting to databases that support JDBC from within a JBI environment. In this tutorial, you will create a SOA project using BPEL and the Database Binding Component, create a Composite Application, and then deploy and test the application.

The topics listed here provides information about Understanding the Database Binding Component (BC).

What You Need to Know

These topics provide information you need to know before using the Database Binding Component.

- [“Tutorial Plan” on page 6.](#)
- [“Database Binding Component Project in a Nutshell” on page 7.](#)

What You Need to Do

These topics provide instructions to complete a Database Binding Component sample Project. It also helps to test the Binding Component on the system and demonstrate how it is implemented.

- [“To Start the GlassFish V2 Application Server From NetBeans IDE” on page 10.](#)
- [“To Create a BPEL Module Project” on page 12.](#)
- [“To Connect to a MySQL Database” on page 14.](#)
- [“To Start the Admin Console” on page 18.](#)
- [“To Integrate a JDBC Driver” on page 20.](#)
- [“To Create a JDBC Connection Pool” on page 20.](#)
- [“To Create a JDBC Resource” on page 23.](#)
- [“To Create a WSDL Document : dbWSDL” on page 25.](#)
- [“To Select the Database Connection” on page 27.](#)
- [“To Edit the SQL Code” on page 30.](#)
- [“To Create a WSDL Document : SOAPWSDL” on page 34.](#)

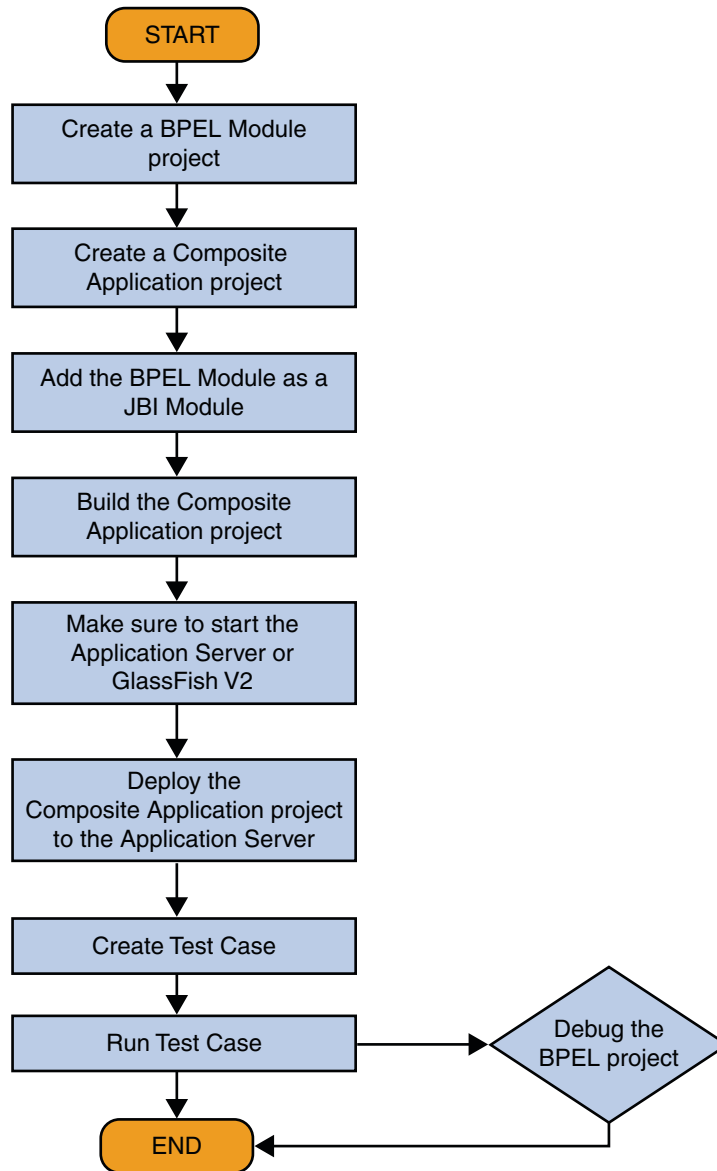
- “To Create a BPEL Process” on page 42.
- “To Add a Partner Link” on page 44.
- “To Add a Web Service and Basic Activities” on page 45.
- “To Edit Web Service : Receive1” on page 49.
- “To Edit the Web Service : Invoke1” on page 51.
- “To Edit the Web Service : Reply1” on page 55.
- “To Edit the Basic Activities : Assign1” on page 57.
- “To Edit the Basic Activities : Assign2” on page 59.
- “To Invoke Explicit Validation” on page 61.
- “To Create the Composite Application Project” on page 64.
- “To Deploy the Composite Application” on page 72.
- “To Test the Composite Application” on page 74.
- “To Debug the Test Case” on page 79.

Tutorial Plan

Below is a summary of the steps that this tutorial takes you through to create a Composite Application using the Database BC.

1. Create a BPEL Module project using the New Project wizard.
2. Create Database BC and SOAP BC WSDL documents for the BPEL Module.
3. Create a Composite Application project.
4. Add the BPEL Module project (*.jar) as a JBI Module to the Composite Application project.
5. Build the Composite Application project and make sure that the Application Server is started.
6. Deploy the Composite Application project to the Application Server.
7. Create Test Case.
8. Run the Test Case.
9. (Optional) Debug the BPEL process.

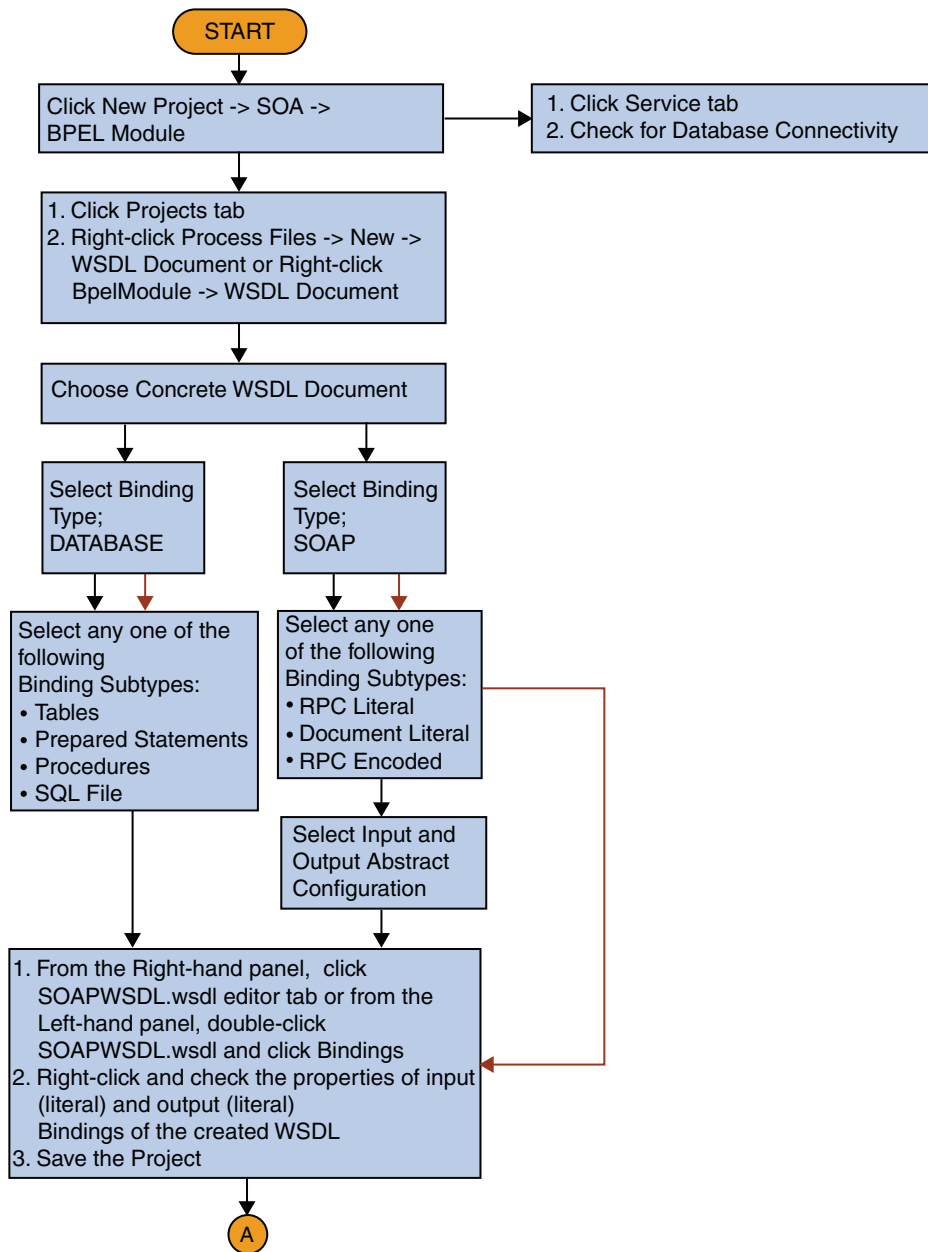
This is performed when the Test Case fails.



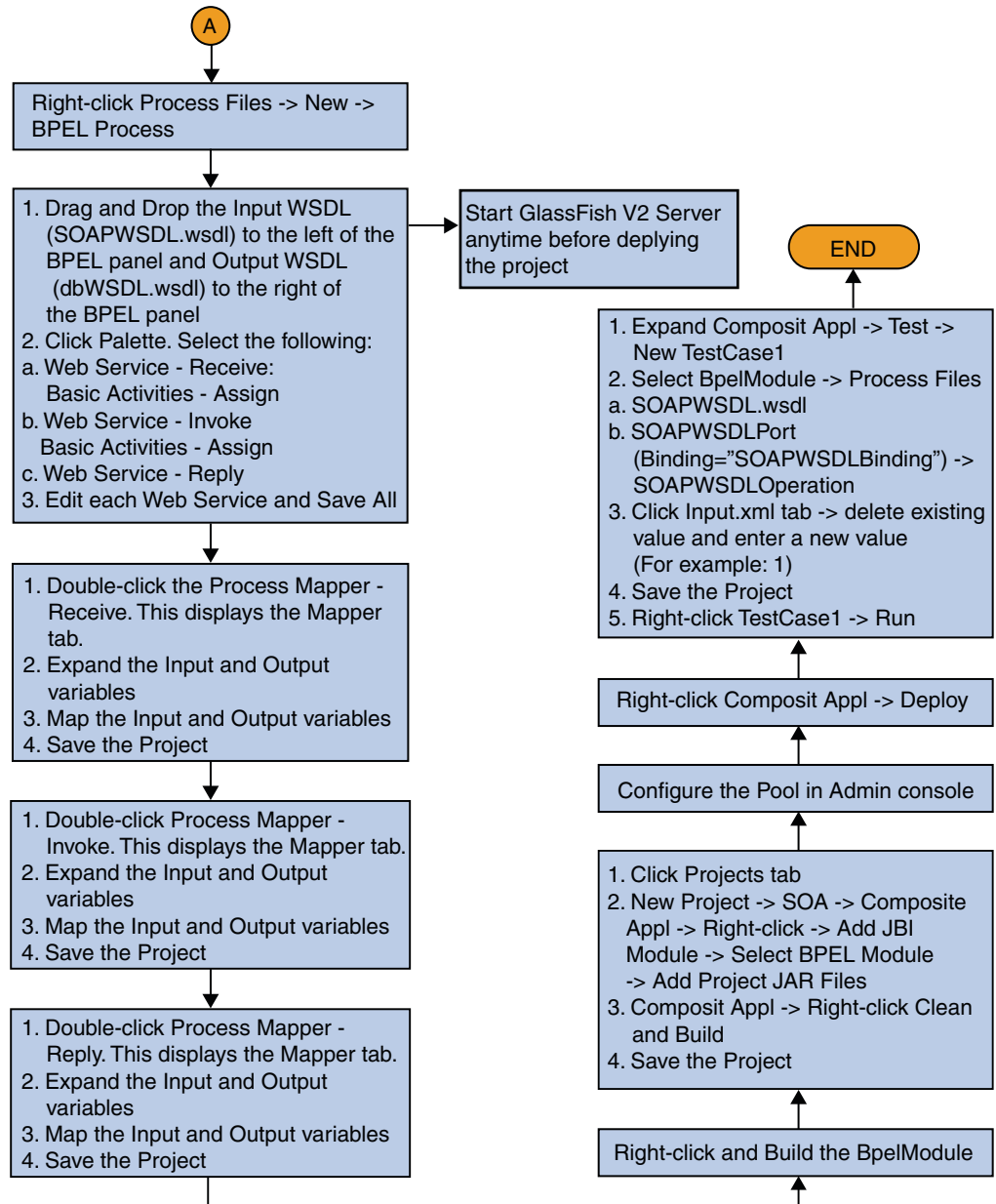
Database Binding Component Project in a Nutshell

The illustration explains the entire process.

Flowchart 1



Flowchart 1 (Continued)

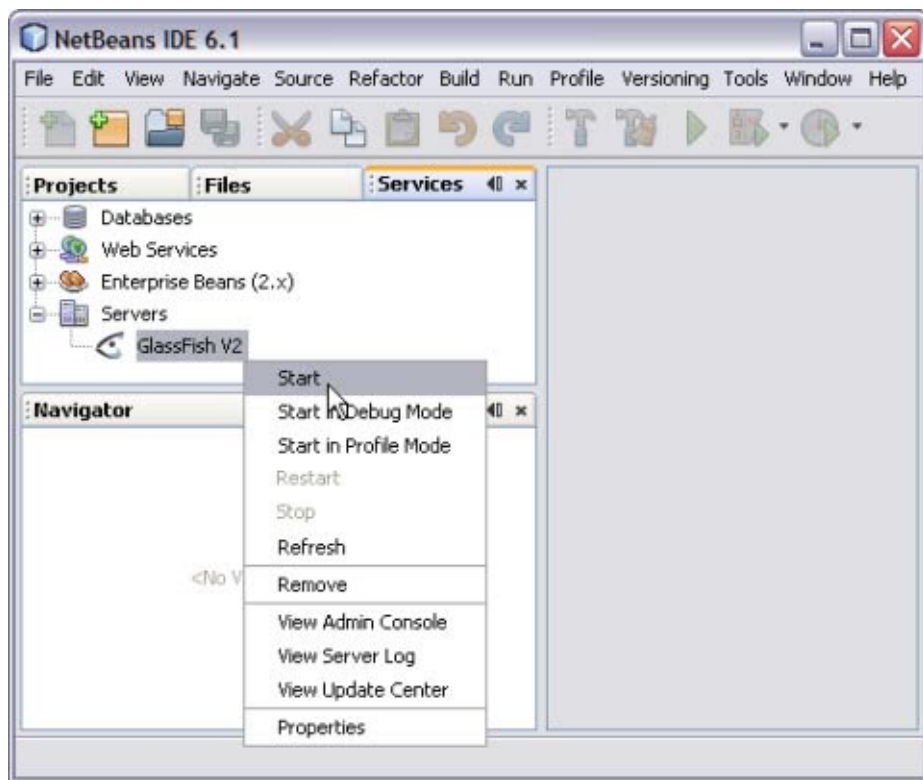


Starting the GlassFish V2 Application Server

Follow the procedure to start the GlassFish V2 Application Server.

▼ To Start the GlassFish V2 Application Server From NetBeans IDE

- 1 Choose Window —> Services, if the Services tab is not visible.
- 2 Click Services tab and expand the Servers node.
The Servers node should contain a GlassFish V2 subnode.
- 3 Right-click the GlassFish V2 node and select Start.

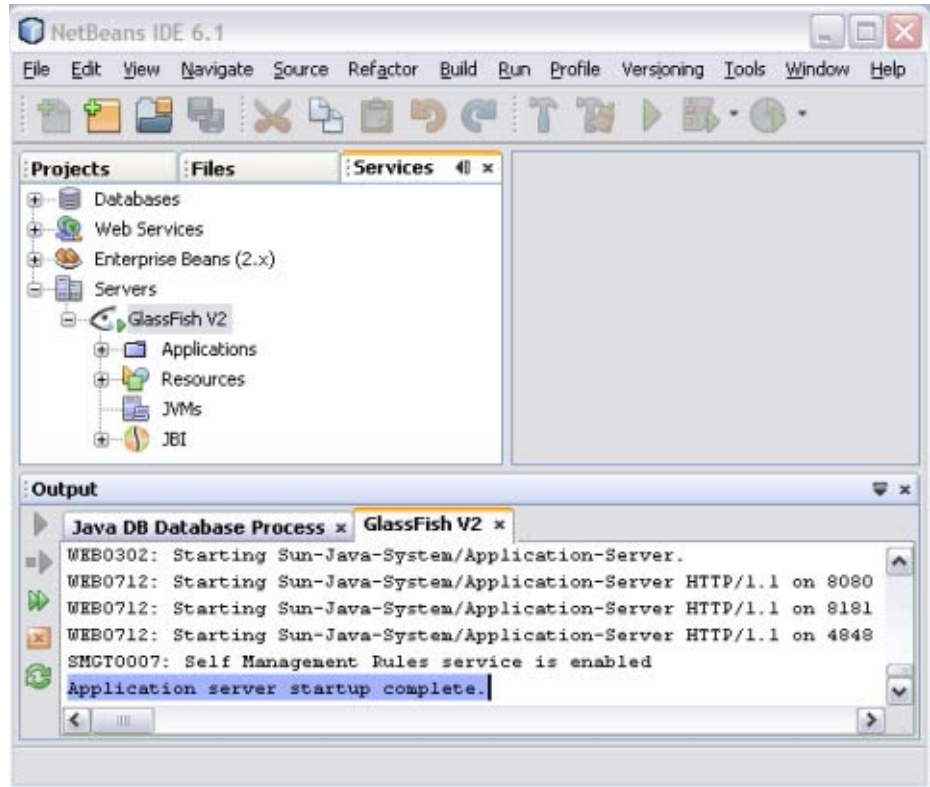


The Output window displays logging information about the application startup.

The following message appears in the Output console indicating the application server is listening.

Application server startup complete

Note – A green arrow badge on the GlassFish Application Server node indicates the server is listening.



Creating a BPEL Module Project For Table Type Operations

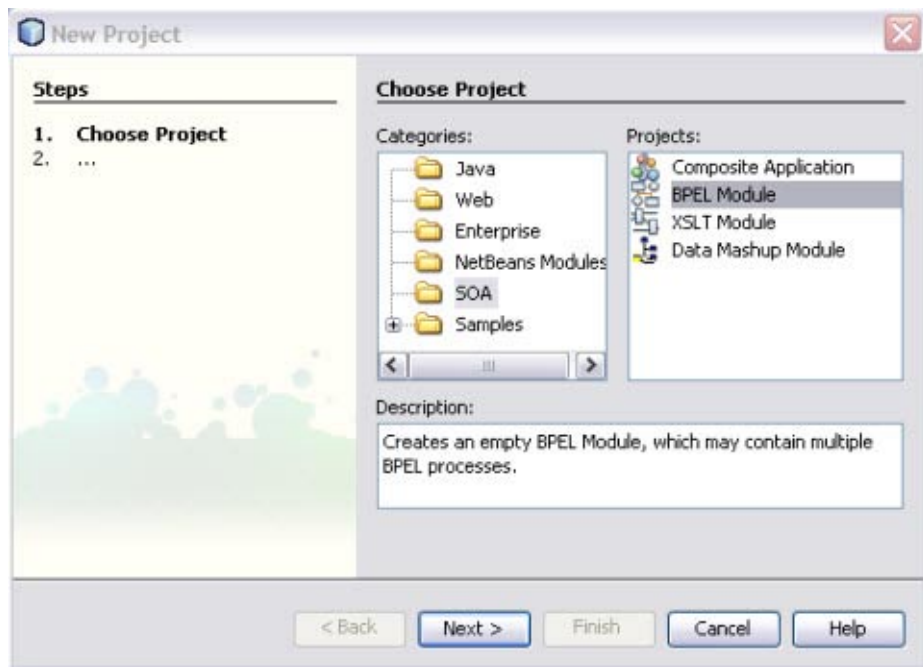
A table represents a database table. It consists of fields and methods. This allows you to perform query, update, insert, and delete SQL operations in a table.

- Fields correspond to the columns of a table
- Methods are the operations that you can apply

Create a BPEL Module project with a name **BpelModuleSample**.

▼ To Create a BPEL Module Project

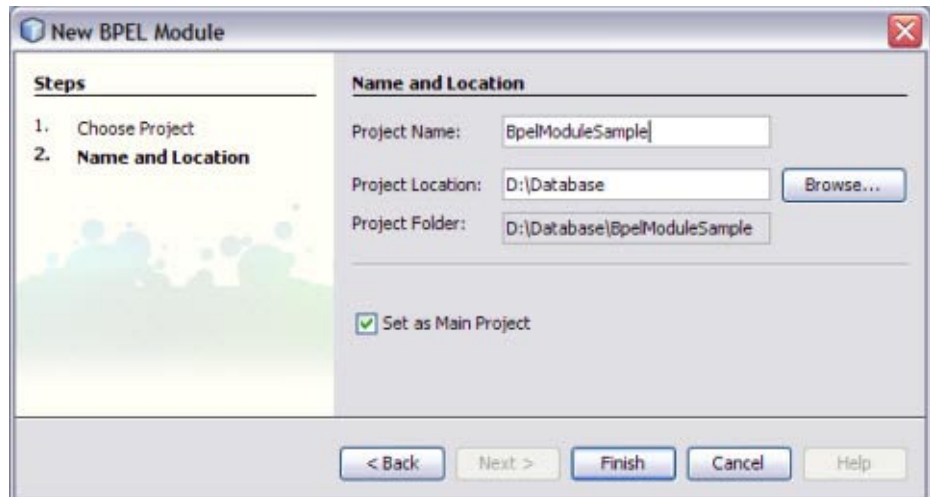
- 1 Choose File —> New Project from the main menu.
This opens the New Project wizard.
- 2 Select the SOA node from the Categories list.
- 3 Select the BPEL Module node from the Projects list.



- 4 Click Next.
- 5 In the Project Name field, enter BpelModuleSample.

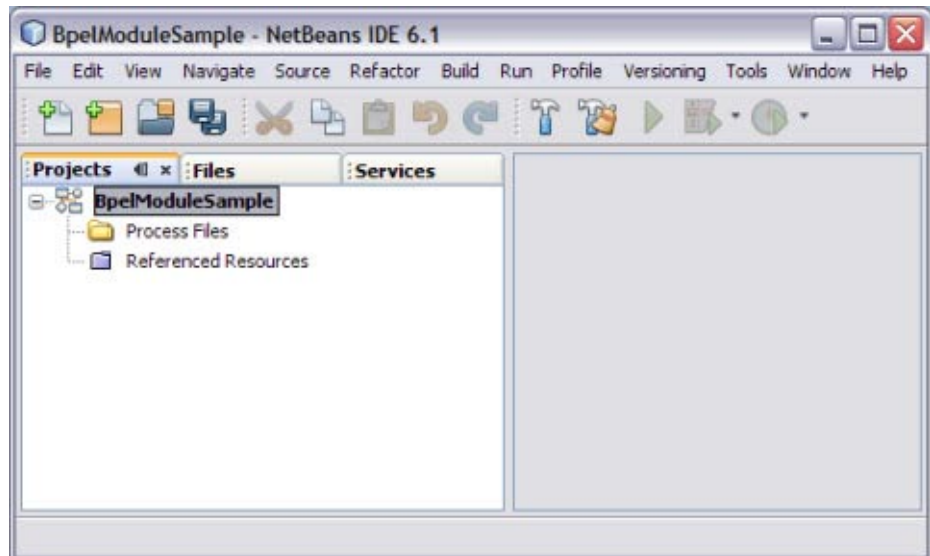
6 Click Browse to navigate to the project location field.

The IDE stores the project files in this location. You can leave the default value.



7 Click Finish.

The Projects window now contains a project node for a BPEL Module project.



8 Click Save All.

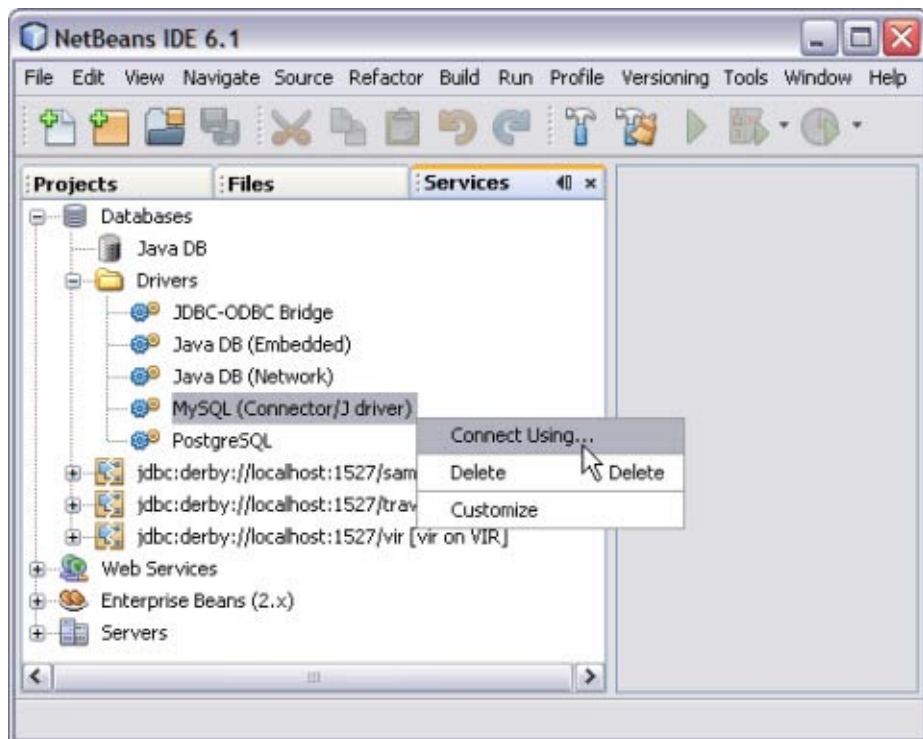
Connecting to a MySQL Database

Connections to databases are managed using database drivers, which enable applications written in different programming languages to interact with the database management system. Java CAPS include the MySQL Native (`mysql-connector-java-5.1.5-bin.jar`), which is a Java implementation of the JDBC API, and communicates directly with the MySQL Driver.

▼ To Connect to a MySQL Database

- 1 Click Services tab.
- 2 Expand the Drivers node from the Database Explorer. Right-click the MySQL (Connector/J driver) and select Connect Using.

The New Database Connection dialog box is displayed.



- 3 In the Basic Setting tab, enter the Database's URL in the corresponding text field.

The URL identifies the type and location of a database server. The format is `jdbc:mysql://hostname:port/database_name`.

4 Enter User Name and Password.

Note – Select the Remember password option. This is optional.



5 Click OK to accept the credentials.

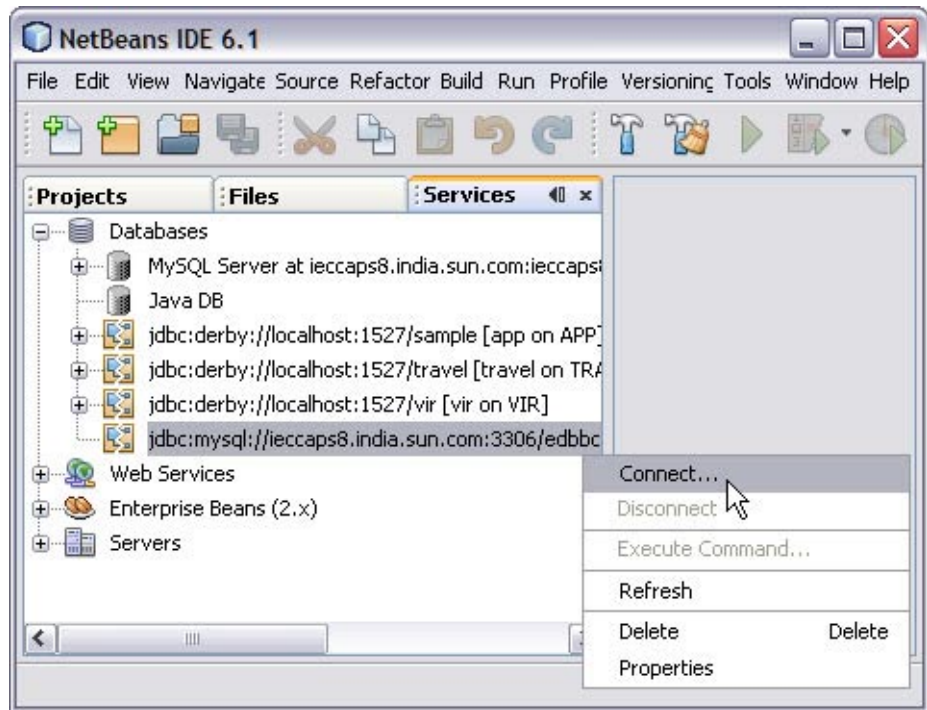
This displays a new Connection node in the Database Explorer under the Databases node.



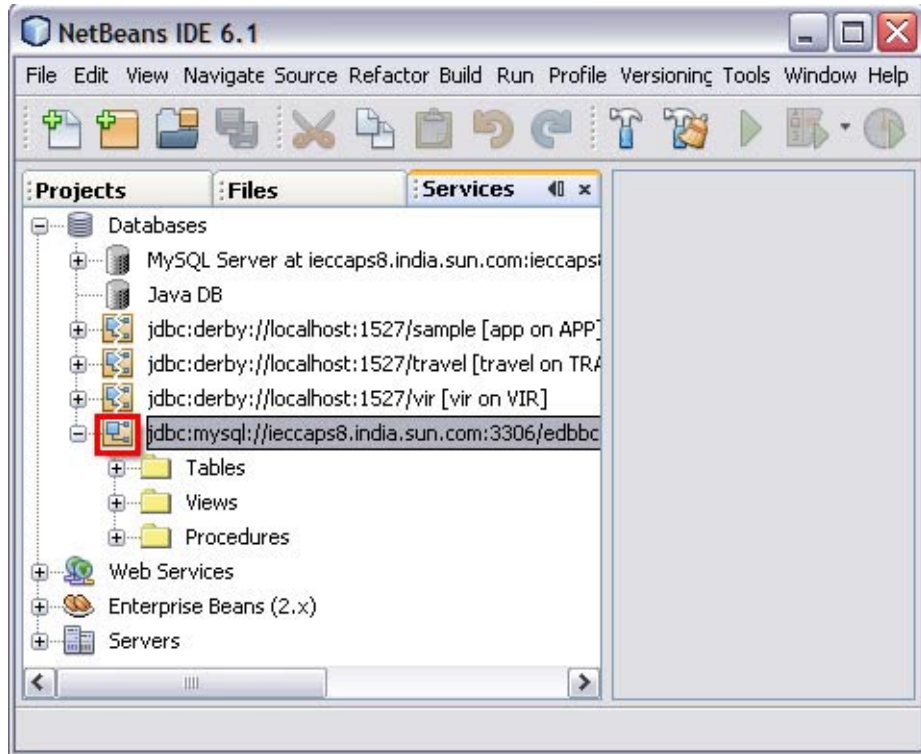
6 Click OK to accept the default schema.

7 Right-click the MySQL Database URL in the Services window (Ctrl-5). Click Connect.

This opens the New Database Connection dialog box.



The illustration shows that the Database Connection is enabled.



You are now connected to the MySQL RDBMS in the IDE.

Starting the GlassFish Admin Console

You define the database connections for your project using the GlassFish Admin Console.

▼ To Start the Admin Console

- 1 Open a web browser and enter the following URL:

`http://hostname:port/login.jsf`

Note – The default port is **4848**.

- 2 Enter the User Name and Password.

Note – The default username is admin; the password was specified during installation.

3 Click Login.



Sun Java System Application Server
Admin Console

User Name:

Password:

Copyright © 2007 Sun Microsystems, Inc. All rights reserved. Unpublished - rights reserved under the Copyright Laws of the United States. Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties. Portions may be derived from Berkeley BSD systems, licensed from U. of CA. Sun, Sun Microsystems, the Sun logo, Java, Jini, Netra, Solaris, Sun Ray and Sun Java System Application Server are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Setting Up Database Access

Follow these steps to create and configure a database connection:

1. Make the driver's JAR file accessible to the domain's server instance. See [“To Integrate a JDBC Driver” on page 20](#).
2. Create a connection pool for the database. See [“To Create a JDBC Connection Pool” on page 20](#).
3. Create a JDBC resource that points to the connection pool. See [“To Create a JDBC Resource” on page 23](#).

Integrating a JDBC Driver

A JDBC driver translates an application's JDBC calls into the protocol of the database server.

▼ To Integrate a JDBC Driver

- 1 Make the driver accessible to the common class loader.
 - a. Copy the driver's JAR and ZIP files into the *JavaCAPS_Home\appserver\lib*.
 - b. Restart the application server.
- 2 Make the driver accessible to the system class loader.

Creating a JDBC Connection Pool

When creating the pool with the Admin Console, you are actually defining the aspects of a connection to a specific database. A JDBC connection pool is a group of reusable connections for a particular database. Because creating each new physical connection is time consuming, the server maintains a pool of available connections to increase performance. When an application requests a connection, it obtains one from the pool. When an application closes a connection, the connection is returned to the pool.

Before creating the pool, you must first install and integrate the JDBC driver. See [“To Integrate a JDBC Driver” on page 20](#).

When creating the Create Connection Pools, certain data specific to the JDBC driver and the database vendor must be specified. Before proceeding, gather the following information:

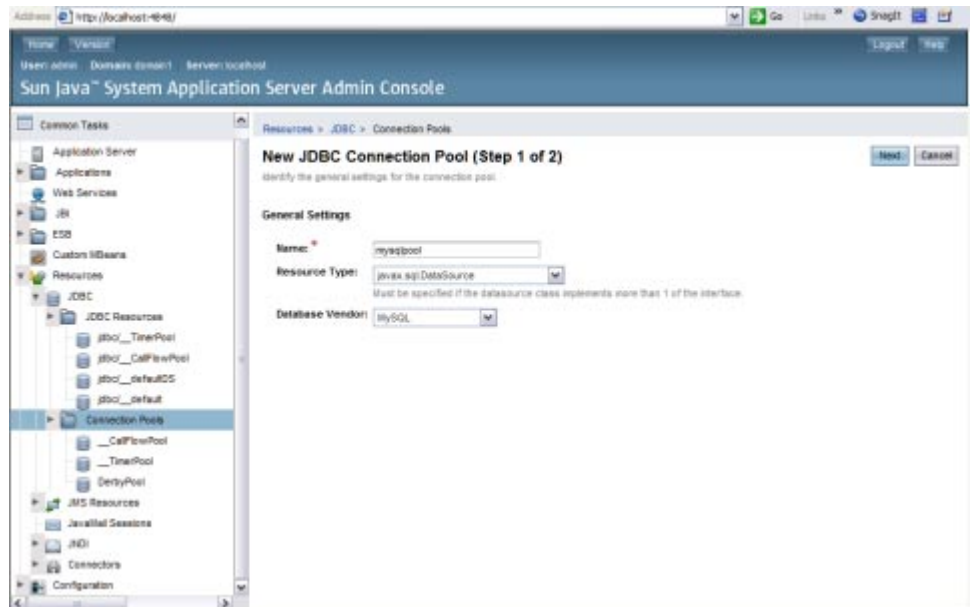
- Database vendor name
- Resource type (select one of the following options):
 - **javax.sql.DataSource** (local transactions only)
 - **java.sql.ConnectionPoolDataSource** (local transactions, possible performance improvements)
 - **javax.sql.XADataSource** (global transactions)
- Data source class name
- Required properties, such as the database name (URL), user name, and password

▼ To Create a JDBC Connection Pool

- 1 Select the **Common Tasks** node, then click **Resources** —> **JDBC** —> **Connection Pool**.

Note – Select **New** to create a new connection pool from the **New Connection Pool** page.

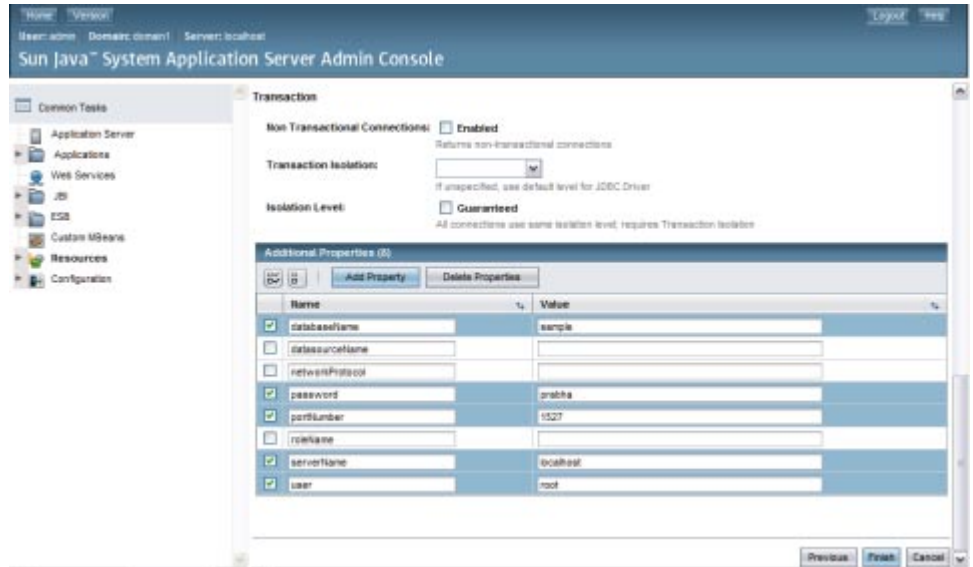
- 2 Specify the General Settings as follows:
 - a. **Name:** Specify a Name for the pool.
 - b. **Resource Type:** Select a Resource Type from the drop-down menu.
See the list of choices above.
 - c. **Database Vendor:** Select a vendor from the list provided in the drop-down menu.



- 3 Click Next.
- 4 Specify the additional General Settings as follows:
 - **Datasource Classname:** If the JDBC driver has a Datasource class for the resource type and database vendor specified in the previous page, then the value of the Datasource Classname field is provided. An example for MySQL is `com.mysql.jdbc.jdbc2.optional.MysqlDataSource`.
 - **Description:** A text description of the connection pool.



- 5 **Specify the Pool Settings.**
The values are entered, by default.
- 6 **Specify the Connection Validation.**
The values are entered, by default.
- 7 **Specify the Transaction Isolation settings.**
The values are entered, by default.
- 8 **Add the required properties in the Additional Properties table, such as database name (URL), user name, and password. Enter the following.**
 - a. `databaseName`
 - b. `password`
 - c. `portNumber`
 - d. `serverName`
 - e. `user`



9 Click Finish.

▼ To Create a JDBC Resource

Applications get a database connection from a connection pool by looking up a data source on the Java Naming and Directory Interface (JNDI) tree and then request a connection. The connection pool associated with the datasource provides the connection to the application.

Before You Begin Before creating a JDBC resource, first create a JDBC connection pool. See [“To Create a JDBC Connection Pool” on page 20](#).

- 1 Expand the Resources, then JDBC path node.
- 2 Select the JDBC Resources node.
- 3 Click New from the JDBC Resources page.
- 4 Specify the Resource settings as follows:
 - a. **JNDI Name: Specify a unique name.**

The JNDI name organizes and locates components within a distributed computing environment similarly to the way that card catalogs organize and represent locations of books in a library. Consequently, the JNDI name becomes an important method of accessing the JDBC resource. By convention, the name begins with the jdbc/string.

For example: jdbc/payrolldb.

Note – Remember to give a forward slash.

- b. **Pool Name:** Choose the connection pool to be associated with the new JDBC resource.
 - c. **Description:** Type a short description of the resource.
 - d. **Status:** If you want the resource to be unavailable, deselect the Enabled checkbox. By default, the resource is available (enabled) as soon as it is created.
- 5 Click OK.

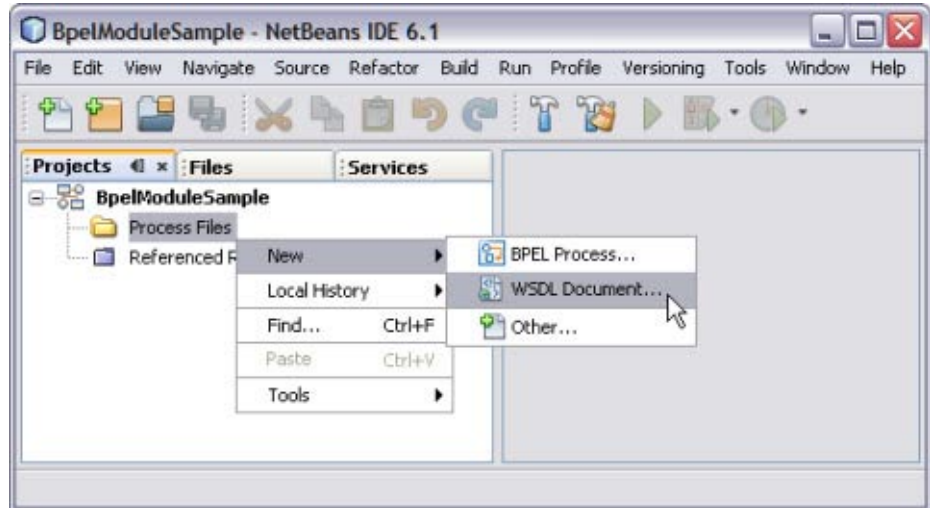


Creating a WSDL Document For Type : DATABASE

In this section, you add a WSDL document, to the BPEL Module project and then use the Partner view of the WSDL editor to configure the components of the WSDL document.

▼ To Create a WSDL Document : dbWSDL

- 1 Expand the project node in the Projects window.
The New WSDL Document wizard appears.
- 2 Right-click the project node or Process Files node. Select New —> WSDL Document.



- 3 In the File Name field, enter dbWSDL.wsdl.
- 4 Select Concrete WSDL Document.
 - **binding**
Defines the message format and protocol details for a port type. For JDBC Binding Component, this is always a message containing a database operation defined using the JDBC API.
 - **service**
Indicates which binding to use and how to access the database resource. For Database Binding Component, the address element (<jdbc:address>) specifies the JNDI name for the database resource.
- 5 In the Binding field, select Database from the drop-down list.
- 6 Choose any one of the following Types from the drop-down list.
 - Table
 - Prepared Statements
 - Procedures

- SQL File

7 Choose Type — Table from the drop-down list.

The screenshot shows the 'New WSDL Document' dialog box with the 'Steps' pane on the left and the 'Name and Location' pane on the right. The 'Steps' pane lists six steps, with the second step, 'Name and Location', highlighted. The 'Name and Location' pane contains the following fields and options:

- File Name:** dbWSDL
- Project:** BpelModuleSample
- Folder:** src (with a 'Browse...' button)
- Created File:** D:\Database\BpelModuleSample\src\dbWSDL.wsdl
- Target Namespace:** http://j2ee.netbeans.org/wsdl/BpelModuleSample/dbWSDL
- WSDL Type:**
 - ☐ Abstract WSDL Document
 - ☒ Concrete WSDL Document
- Binding:** DATABASE (dropdown menu)
- Type:** Table (dropdown menu)

At the bottom of the dialog are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

8 Click Next.

9 Select the URL from the drop-down list.

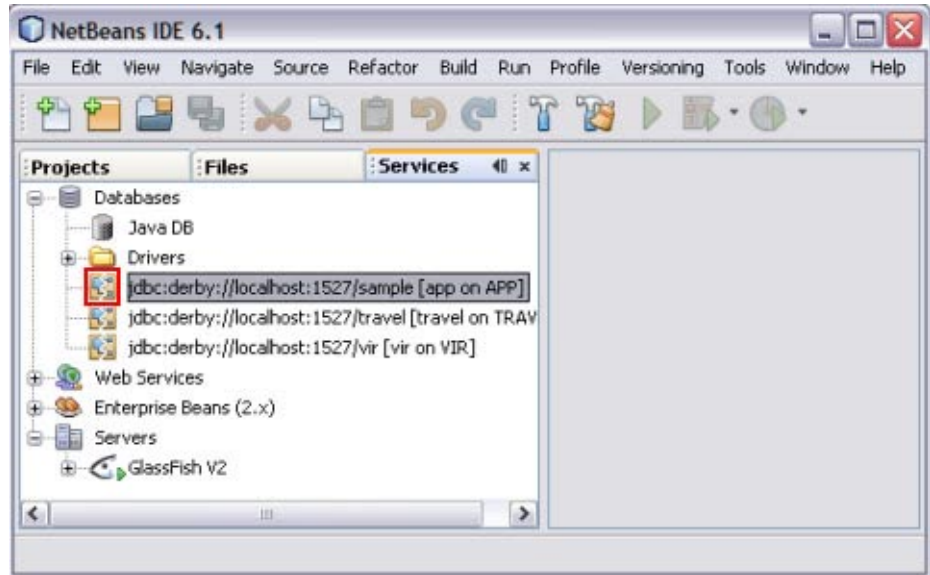
The screenshot shows the 'New WSDL Document' dialog box with the 'Steps' pane on the left and the 'Select one of the following' pane on the right. The 'Steps' pane lists six steps, with the third step, 'Select one of the following', highlighted. The 'Select one of the following' pane contains the following fields and options:

- URL :** A dropdown menu with the text '<Select an Item from the list>' and a list of three items:
 - jdbc:derby://localhost:1527/sample [app on APP]
 - jdbc:derby://localhost:1527/travel [travel on TRAVEL]
 - jdbc:derby://localhost:1527/vir [vir on VIR]

At the bottom of the dialog are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

You find the MySQL Database is not configured and is not available in the list.

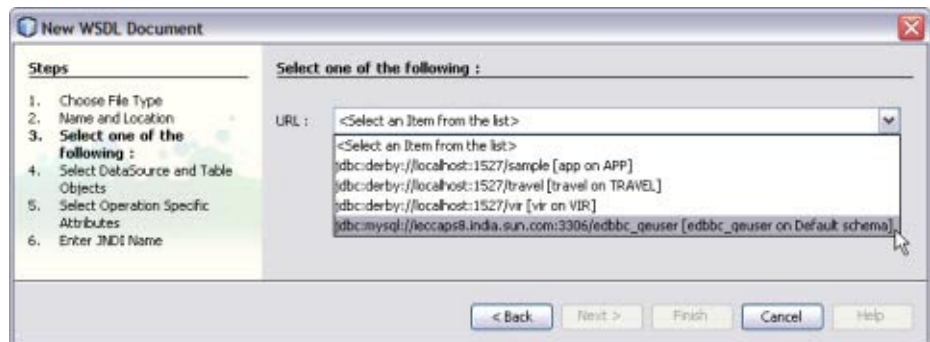
Note – A broken icon indicates that the database is not connected.



▼ To Select the Database Connection

Before You Begin See [“To Create a WSDL Document : dbWSDL”](#) on page 25 to Create the WSDL Document.

- 1 Select the URL from the drop-down list.

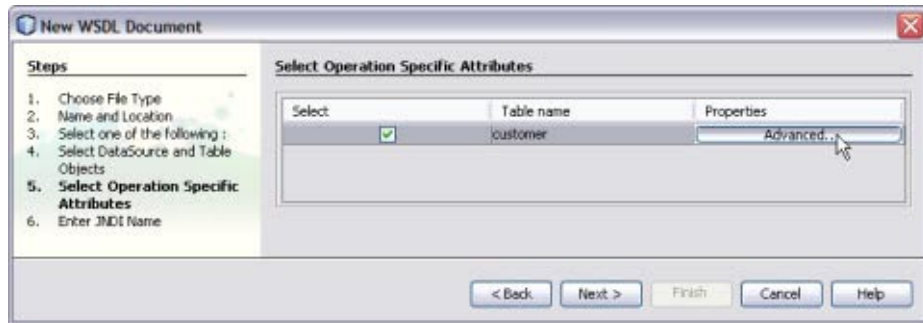


- 2 Click Next.

- 3 Select the table from the DataSource and Table Objects.
 - 4 Click the right arrow to move the table from Available Tables to Selected Tables.
Click the left arrow to remove the table from the list of Selected Tables.
-

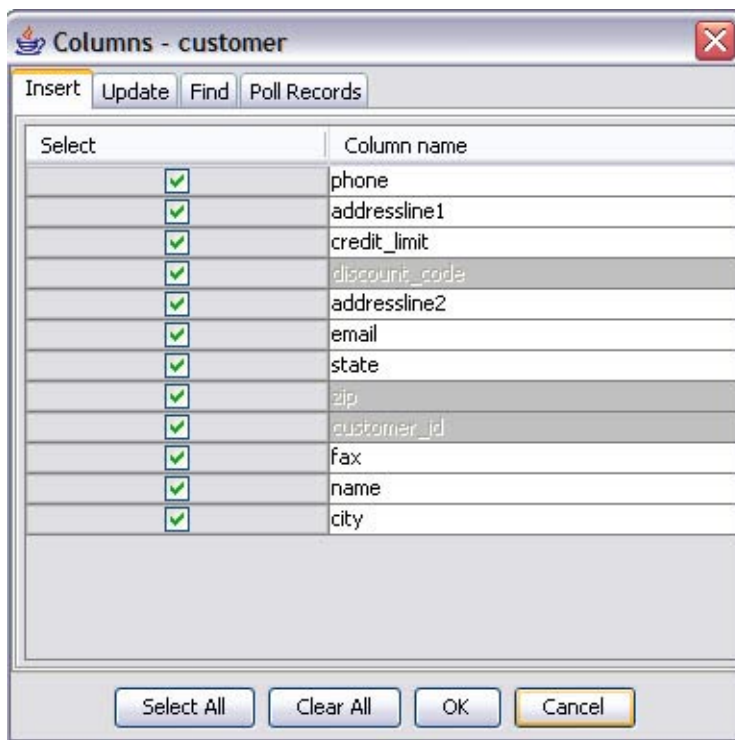
Note – You can select and move one table at a time.

- 5 Click Next.
- 6 Select Operation Specific Attributes.



- 7 Click Advanced. Select the required Column name.
You can Insert, Update, Find, and Poll Records.
-

Note – As the ID is unique, they cannot be unchecked. Hence, these are greyed out.



Note – Based on the requirement, you can check and deselect the Column names.

- a. Choose **Select All** to select all the Column names.
 - b. Choose **Clear All** to clear all the Column names.
- 8 Click **OK**.
 - 9 Click **Next**.
 - 10 Enter an appropriate **JNDI Resource Name**.
 - 11 Click **Finish**.

Observe the following:

- In the Projects window, the IDE adds a dbWSDL.wsdl node under the Process Files node along with the .xsd file.

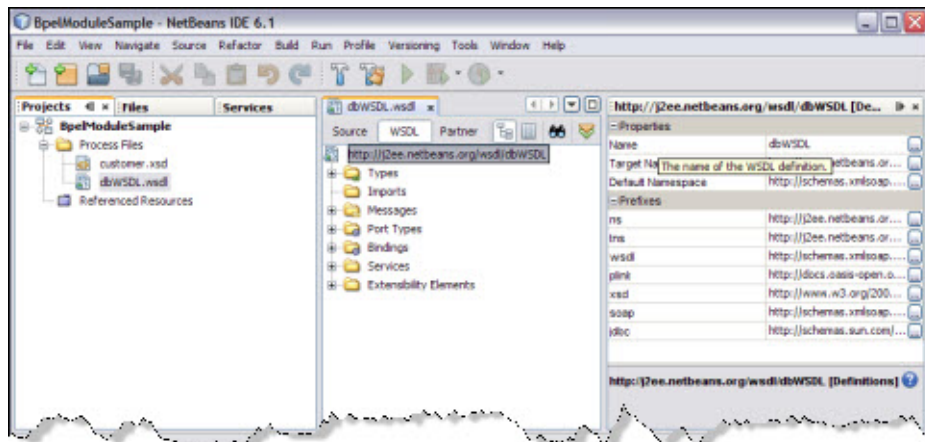
- The dbWSDL.wsdl file is open in the WSDL editor.

The WSDL editor has three views: Source, WSDL, and Partner.

- The Properties window is open.

Note – If you do not see the Properties window, choose Window —> Properties.

- The Navigator window is open.



▼ To Edit the SQL Code

- 1 Double-click the created WSDL Document, dbWSDL.wsdl.

The WSDL Editor is opened.

- 2 Expand Bindings —> binding PortType="jdbcPortType".

For instance, you want to perform a find operation.

- 3 Select and expand find from the list.

Expand inputFind and click jdbc:input.

4 Select sql from Properties.

Modify the SQL Statement.

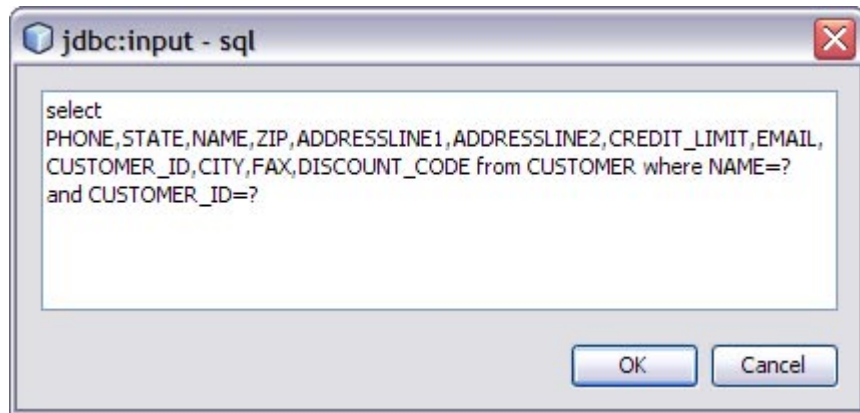
5 Click the ellipsis.

The jdbc:input — sql Editor appears.

Make the necessary changes.

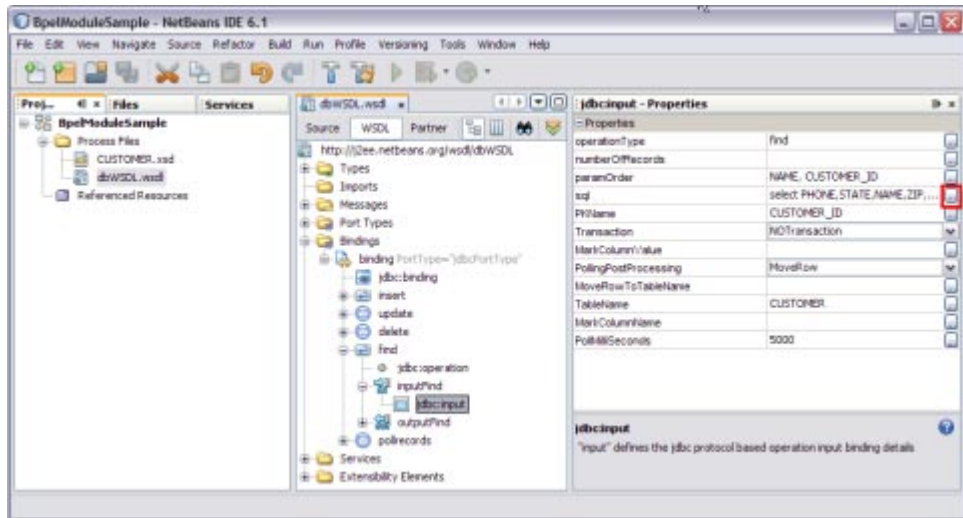
For example:

```
select PHONE, STATE, NAME, ZIP, ADDRESSLINE1, ADDRESSLINE2, CREDIT_LIMIT,  
EMAIL, CUSTOMER_ID, CITY, FAX, DISCOUNT_CODE  
from CUSTOMER where NAME=? and CUSTOMER_ID=?
```

**6 Click OK.**

7 Click the paramOrder from Properties and enter the values.

For example, you want to find for NAME and CUSTOMER_ID. Enter the parameter order of the result.



See Also WSDL View

- In the WSDL view of the WSDL Editor, the WSDL file appears as a tree component or a series of columns.
- The WSDL view has two subviews: tree view and column view. To switch between the subviews, use the buttons in the WSDL Editor toolbar.
- The main nodes in the WSDL view correspond to the major elements in a WSDL file.
 - **Types:** This node enables you to import XML schemas and to add inline schemas.
 - **Imports:** This node enables you to import WSDL files.
 - **Messages:** This node enables you to create, edit, and delete messages.
 - **Port Types:** This node enables you to create, edit, and delete port types.
 - **Bindings:** This node enables you to create, edit, and delete bindings.
 - **Services:** This node enables you to create, edit, and delete services.
 - **Extensibility Elements:** This node enables you to add the following extensibility elements: partner link types, properties, and property aliases.

Some of the nodes in the WSDL view allow you to add extension attributes.

- Right-click the node and choose Add Extension Attribute.
- Specify the name and namespace in the Add Extension Attribute dialog box.
- Specify the value from the Properties window after adding the attribute.
- Right-click the node and choose Remove Attributes to delete the attribute.

- Right-click a node and choose Go To —> Source.

The Source view appears with the cursor positioned at the beginning of the component's block.

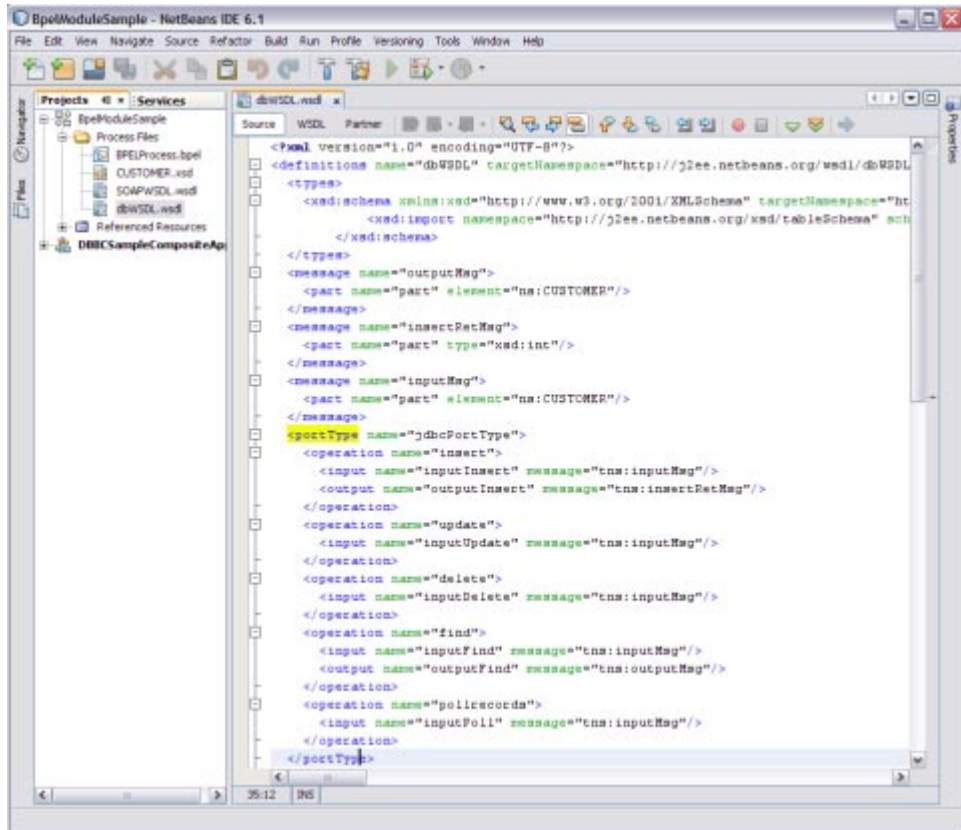
Source View

In the Source view, the underlying XML source code appears. You can directly edit the XML.

1. The top of the Source Editor has a tab for each open document. Each tab shows the name of the document.

Note – If the document has been modified and has not been saved, then an asterisk (*) appears after the name. You can right-click a tab to access various commands.

2. A toolbar is located at the top of the Source Editor window.
3. Source code displayed in the Source Editor is syntactically colored.
4. The Source Editor status line is located beneath the horizontal scroll bar. To toggle between insert mode and overwrite mode, use the Insert key.



Creating a WSDL Document For Type SOAP

In this section, you add a WSDL document to the BPEL Module project. Later, you can use the Partner view of the WSDL editor to configure the components of the WSDL document.

When creating or editing a WSDL file, you may be prompted to select a binding type and a binding subtype. A binding contains protocol and data format information for the operations and messages of a port type. For more information, see step 6.

▼ To Create a WSDL Document : SOAPWSDL

- 1 **Expand the project node in the Projects window.**
The New WSDL Document wizard appears.
- 2 **Right-click the node or Process Files node. Choose New —> WSDL Document.**

3 In the File Name field, enter SOAPWSDL.wsdl.

4 Select Concrete WSDL Document.

- **binding**

Defines the message format and protocol details for a port type. For JDBC Binding Component, this is always a message containing a database operation defined using the JDBC API.

- **service**

Indicates which binding to use and how to access the database resource. For Database Binding Component, the address element (<jdbc:address>) specifies the JNDI name for the database resource.

5 In the Binding field, select SOAP from the drop-down list.

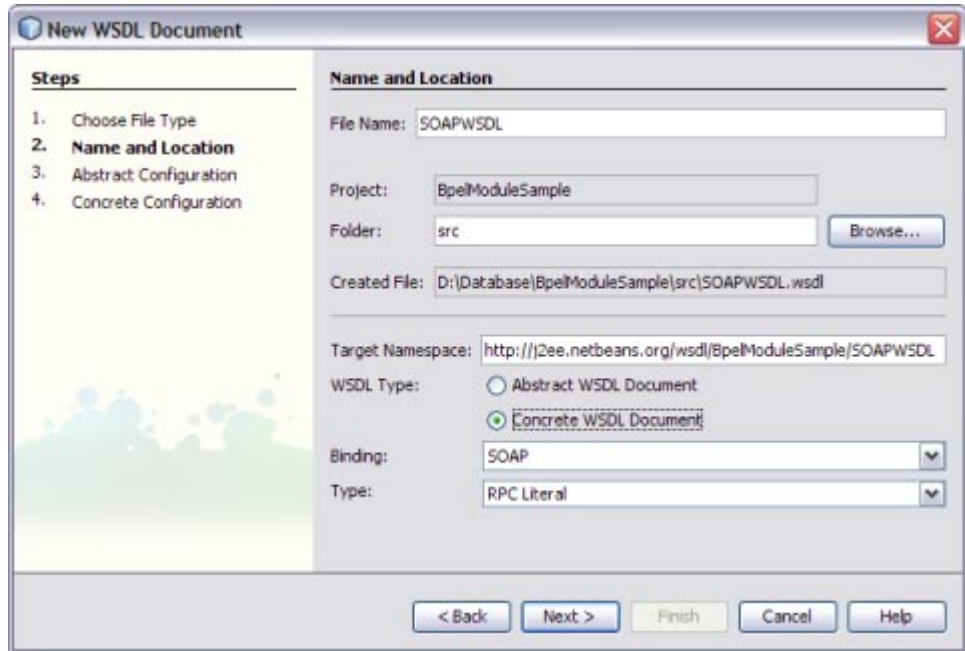
The Binding — SOAP is selected and displayed, by default.

6 Select any one of the following Type.

If you select the SOAP binding type, then you must select any one of the following binding subtypes:

- **RPC Literal:** The operations are RPC oriented (that is, messages contain parameters and return values). Each message part references a concrete schema definition by using the element or type attribute.
- **Document Literal:** The operations are document oriented (that is, messages contain one or more documents). Each message part references a concrete schema definition by using the element or type attribute.
- **RPC Encoded:** The operations are RPC oriented (that is, messages contain parameters and return values). Each message part references an abstract type by using the type attribute.

7 Choose Type — RPC Literal from the drop-down list.



8 Click Next.

This action displays the New WSDL Document dialog box.

9 Choose the Operation Type from the drop-down list.

The WSDL Editor is used to create, edit, and delete port types.

The WSDL Editor supports the following categories of operations:

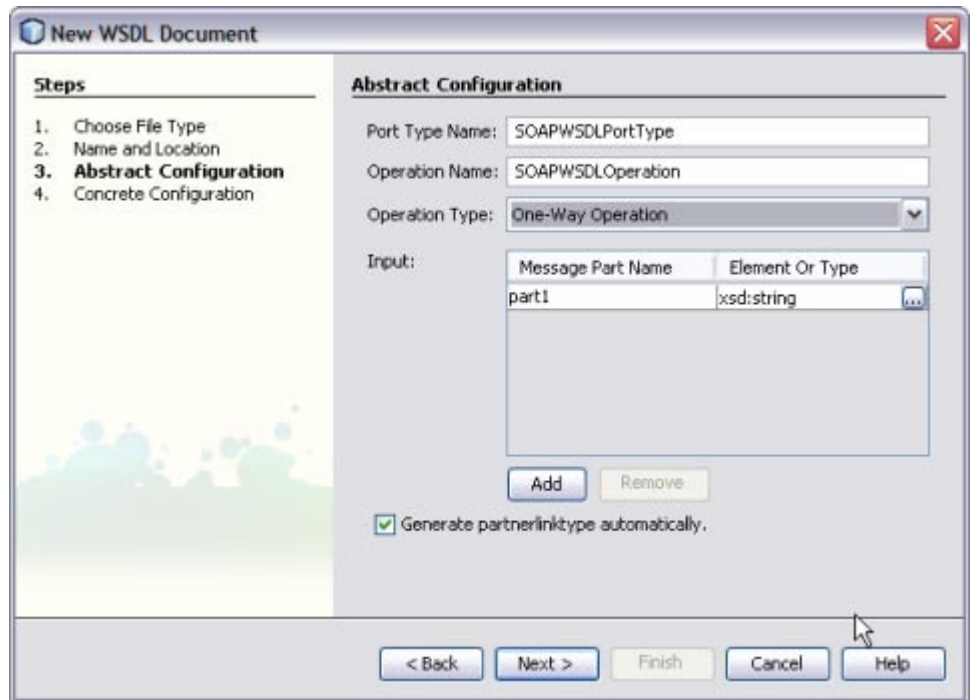
- **Request-Response Operation:** The operation receives a message as input, and sends a message as output.
- **One-Way Operation:** The operation can either receive or send a message as input.

Each message contains one or more logical parts. Specify the name and the type of content for each part.

Tip – If you change the name of a port type or operation, then the WSDL Editor renames all occurrences in the same file.

- a. Right-click the component node to rename all occurrences in associated XSD, WSDL, and BPEL files.
- b. Choose Refactor —> Rename.

When Operation Type : One-Way Operation. The illustration is as shown.



Note –

- a. Click the ellipsis to select any Element or Type.
- b. Enter both the Input and Output Message Part Names for Request-Response Operation.
The default value of the Input and Output Message Part Names is xsd:string.
- c. Click Add to another Message Part Name as the Input.
The checkbox **Generate partnerlinktype automatically** is selected, by default.

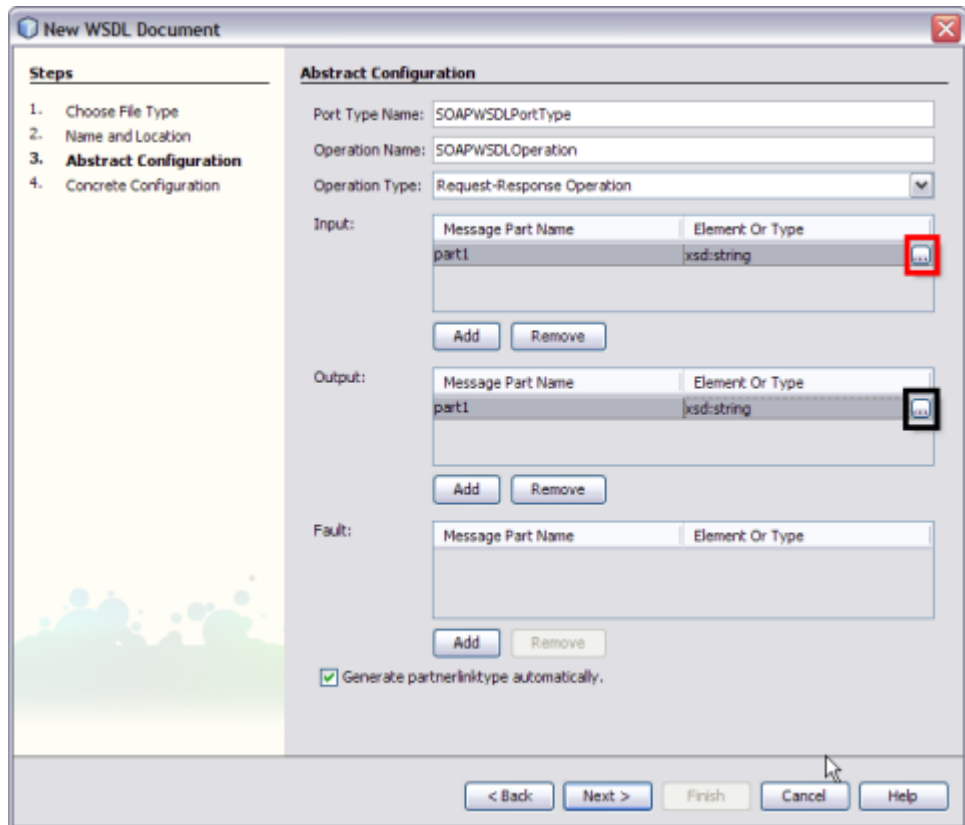
Note – Click Remove to delete the Message Part Name as the Input.

See Steps 11 through 15.

10 Select Operation Type : Request-Response from the drop-down list.

Note – Port Type Name and Operation Name are populated from the previous wizard.

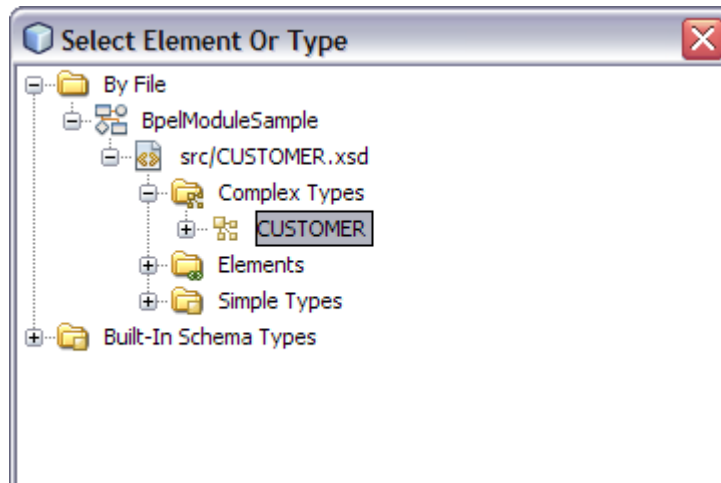
11 Click the ellipsis to select the Element or Type.



The Select Element or Type dialog box is displayed.

12 Expand By File.

- 13 Select the Complex Types for both the Input and Output Message Part Name.**
For example, CUSTOMER.



14 Click OK.

The New WSDL Document is populated with the Input and Output Element Part Names.

The screenshot shows the 'New WSDL Document' dialog box, specifically the 'Abstract Configuration' step. The 'Steps' pane on the left lists four steps: 1. Choose File Type, 2. Name and Location, 3. **Abstract Configuration**, and 4. Concrete Configuration. The main area is titled 'Abstract Configuration' and contains the following fields and controls:

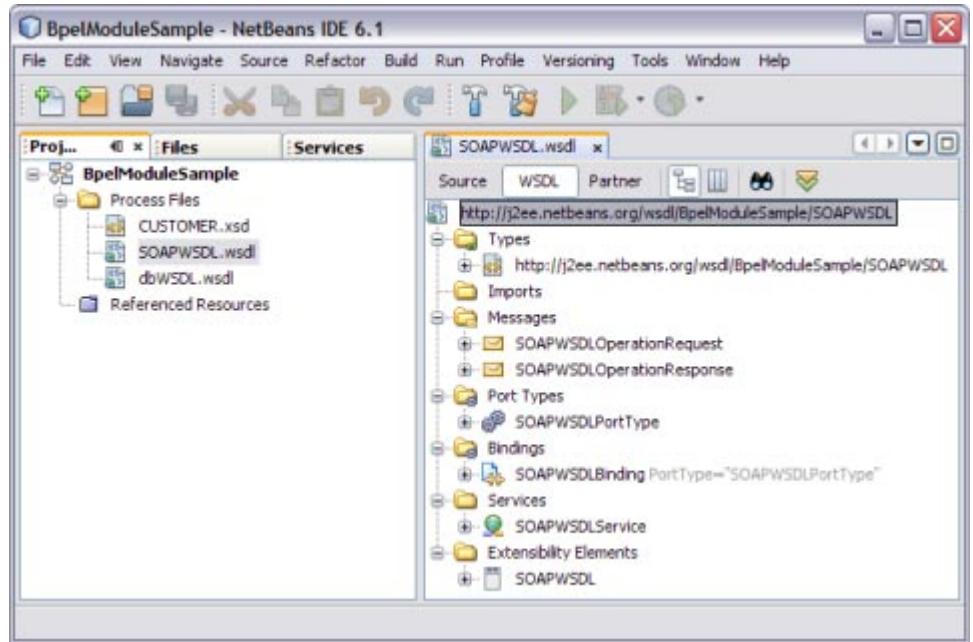
- Port Type Name:** SOAPWSDLPortType
- Operation Name:** SOAPWSDLOperation
- Operation Type:** Request-Response Operation (dropdown menu)
- Input:** A table with two columns: 'Message Part Name' and 'Element Or Type'. It contains one entry: 'part1' and 'ns:CUSTOMER'. Below the table are 'Add' and 'Remove' buttons.
- Output:** A table with two columns: 'Message Part Name' and 'Element Or Type'. It contains one entry: 'part1' and 'ns:CUSTOMER'. Below the table are 'Add' and 'Remove' buttons.
- Fault:** A table with two columns: 'Message Part Name' and 'Element Or Type'. It is currently empty. Below the table are 'Add' and 'Remove' buttons.
- ☒ Generate partnerlinktype automatically.

At the bottom of the dialog are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

15 Click Next.

Verify the Concrete Configuration.

16 Click Finish.

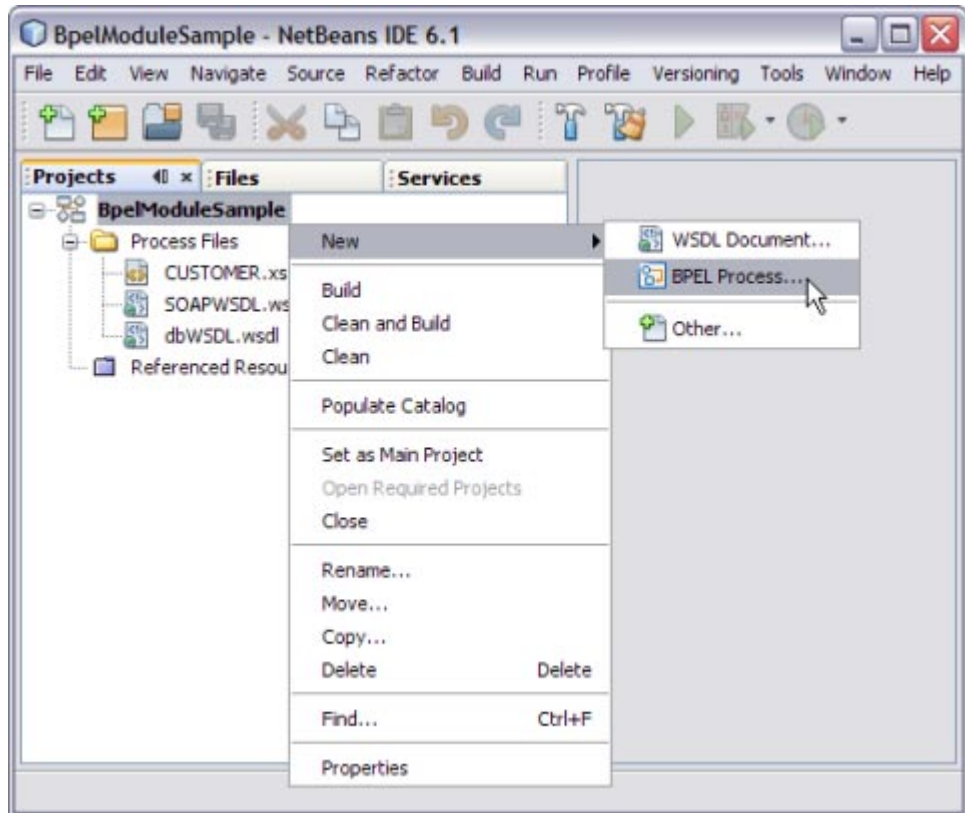


Creating a BPEL Process

In this section, you add a BPEL process file, for example, BPELProcess.bpel. Add a partner link and three activities to the BPEL process file.

▼ To Create a BPEL Process

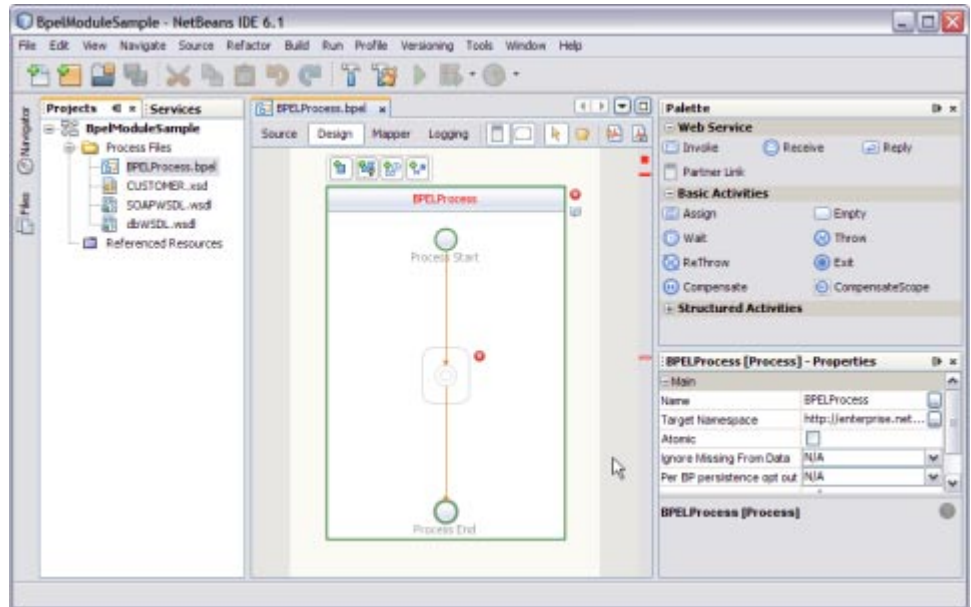
- 1 Expand the BPEL Module project node in the Projects window.
- 2 Right-click the BPEL Module project name or Process Files node. Choose New —> BPEL Process.



This opens the New BPEL Process wizard.

- 3 In the File Name field, enter BPELProcess.

4 Click Finish.

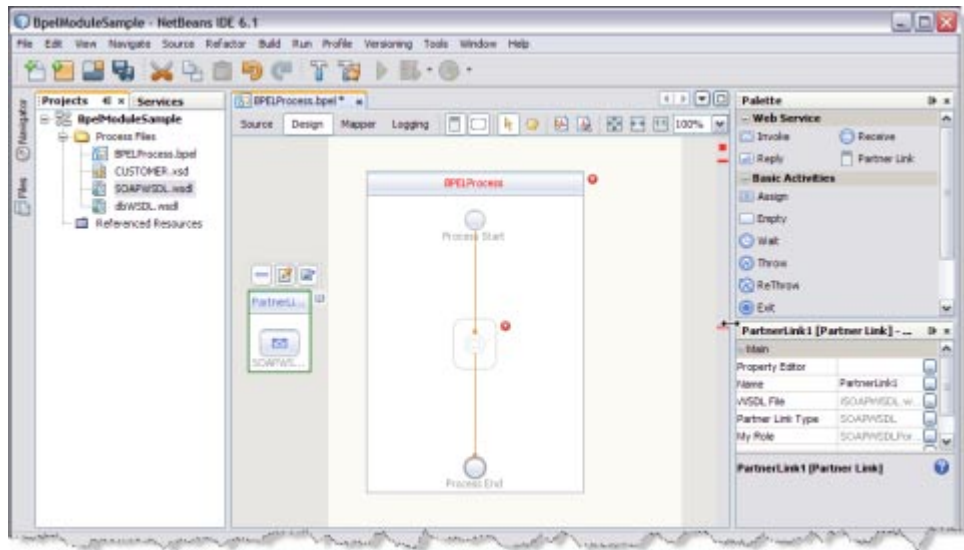


Note –

- In the Projects window, the IDE adds a BPELProcess.bpel node under the Process Files node.
- The BPELProcess.bpel file is open in the BPEL Designer.
- The Properties window is open.
- If you do not see the Properties window, choose Window —> Properties.
- The Navigator window is open showing the BPEL Logical View of the BPEL Process document.

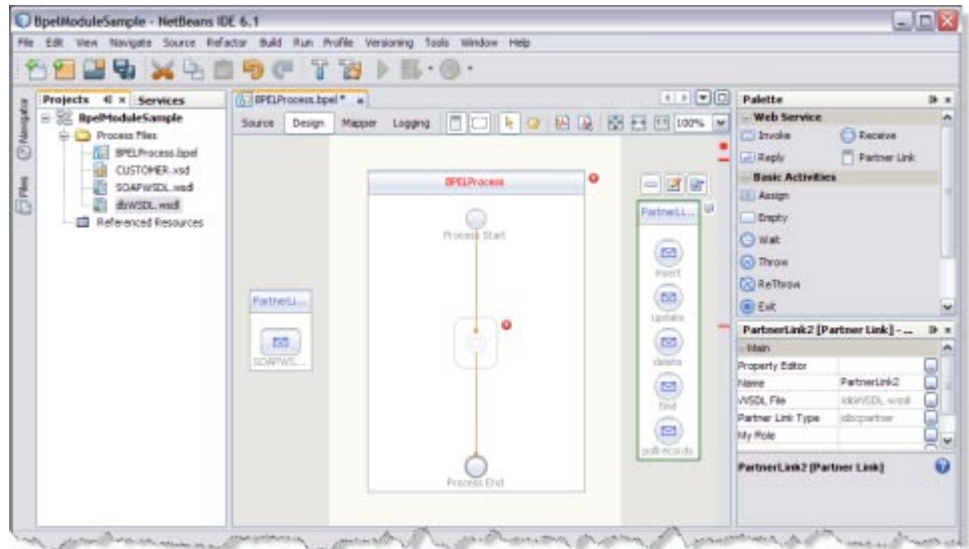
▼ To Add a Partner Link

- 1 Select the Partner Link from the Projects tab (for example, SOAPWSDL.wsdl).
This is the Input WSDL.
- 2 Drag and drop the SOAP WSDL Document to the left panel of the design area.



- 3 Select the Partner Link (dbWSDL.wsdl) from the Projects tab.
This is the Output WSDL.

- 4 Drag and drop the database WSDL Document (dbWSDL.wsdl) to the right panel of the design area.



▼ To Add a Web Service and Basic Activities

Drag and Drop the following Web Services:

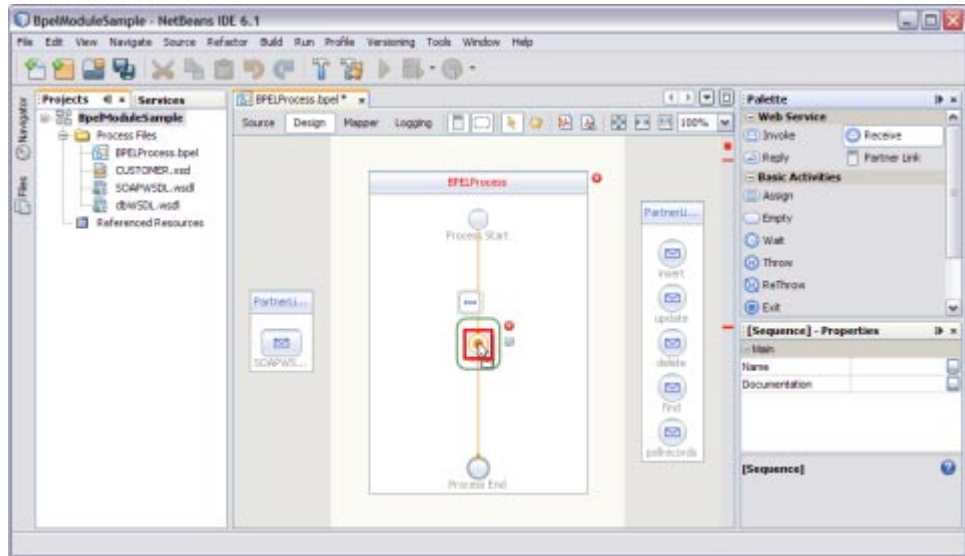
- Receive
- Invoke
- Reply

Drag and Drop the Basic Activities : Assign1 and Assign2.

- 1 Select the Web Service : Receive in the Web Service section of the Palette.

- 2 **Drag the selection to the BPELProcess box in the design area between the Process Start and the Process End activities.**

The IDE provides the visual clues to show an appropriate location to drop the selection.



This action places a Web Service Receive called Receive1 in the Design view.

- 3 **Select the Basic Activities — Assign in the Basic Activities section of the Palette.**

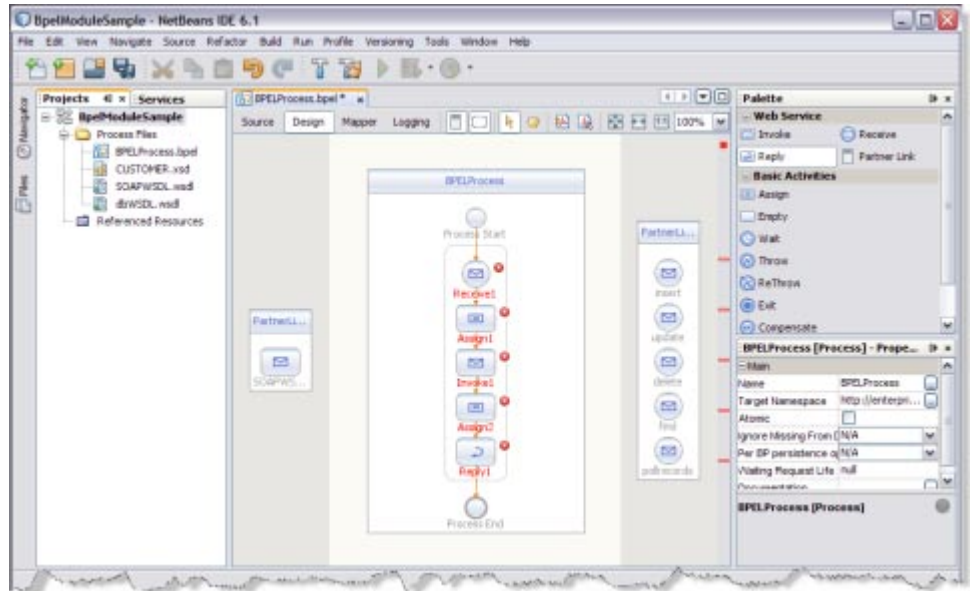
This action places a Assign activity called Assign1 in the Design view.

- 4 **Drag the selection to the BPELProcess box in the design area, between the Process Start and the Process End activities.**

Note – Repeat steps 1 through 4 to select Invoke1, Assign2, and Reply1.

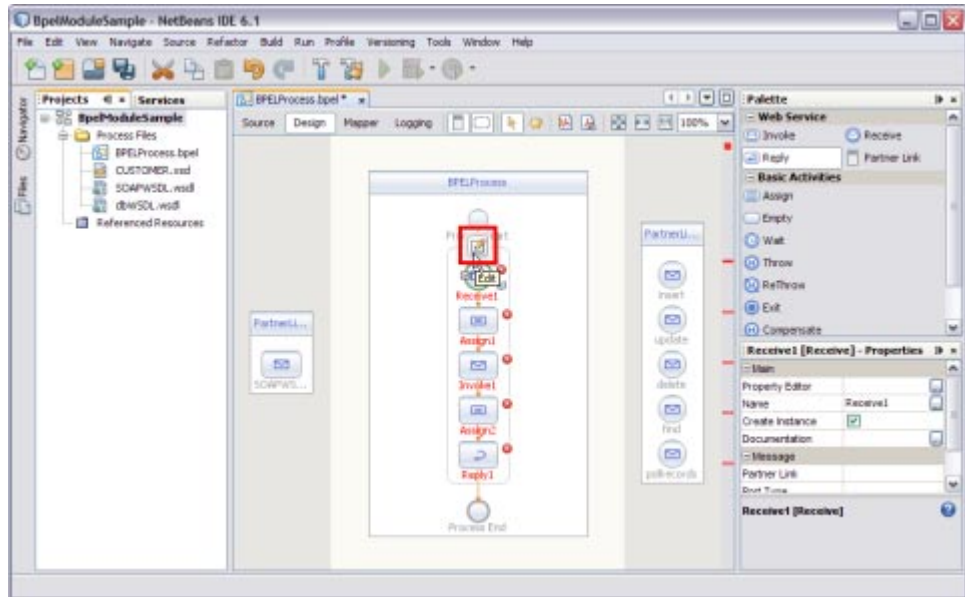
Choose the following:

- Select the Web Service : Invoke and Basic Activities : Assign.
- Select the Web Service ; Reply.



Note – In the diagram, a red cross next to an element means that the element has not passed validation and the output contains errors. Edit each Sequence to pass validation.

The icon symbolizes that the Web Services can be edited.



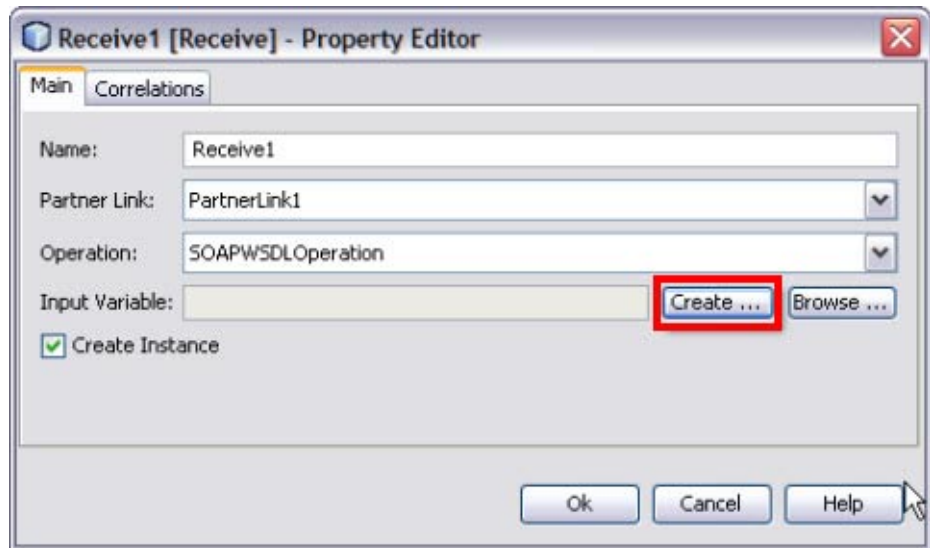
▼ To Edit Web Service : Receive1

- 1 Click **Web Service — Receive1** and click **Edit**.

This opens the Receive1 [Receive] - Property Editor.

- 2 Select the properties from the **Main** tab. Select **PartnerLink1** from the drop-down list.

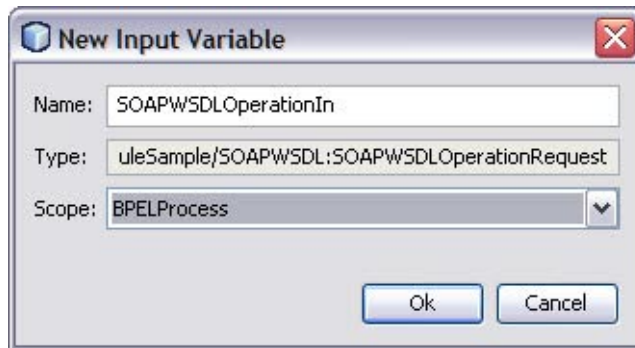
The IDE fills in the Operation field with SOAPWSDLOperation.



- 3 Create a new input variable.

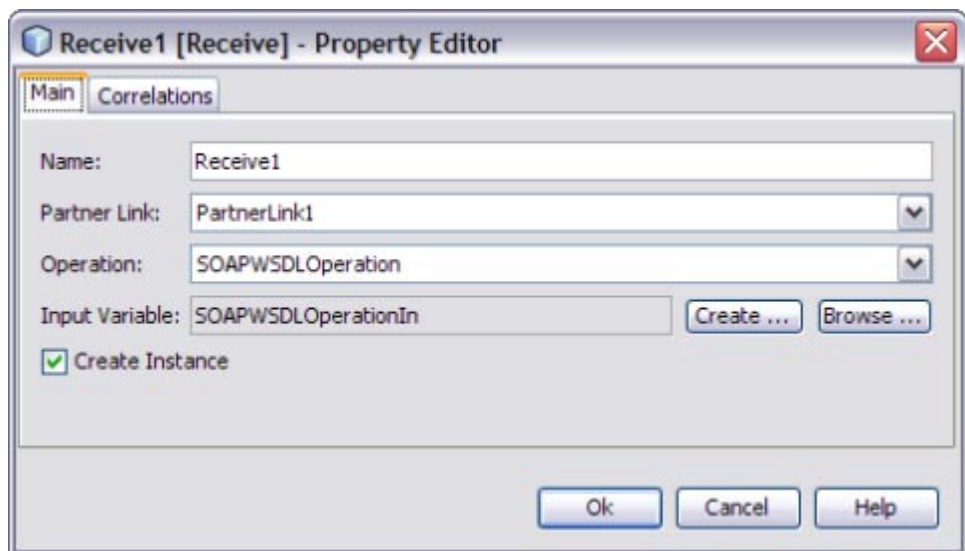
Perform the following:

- Click the **Create** button next to the **Input Variable** field.
This opens the **New Input Variable** dialog box.
- The **Name**, **Type**, and **Scope** are displayed, by default.
You can also change the value in the **Name** field.
- Click **OK**.



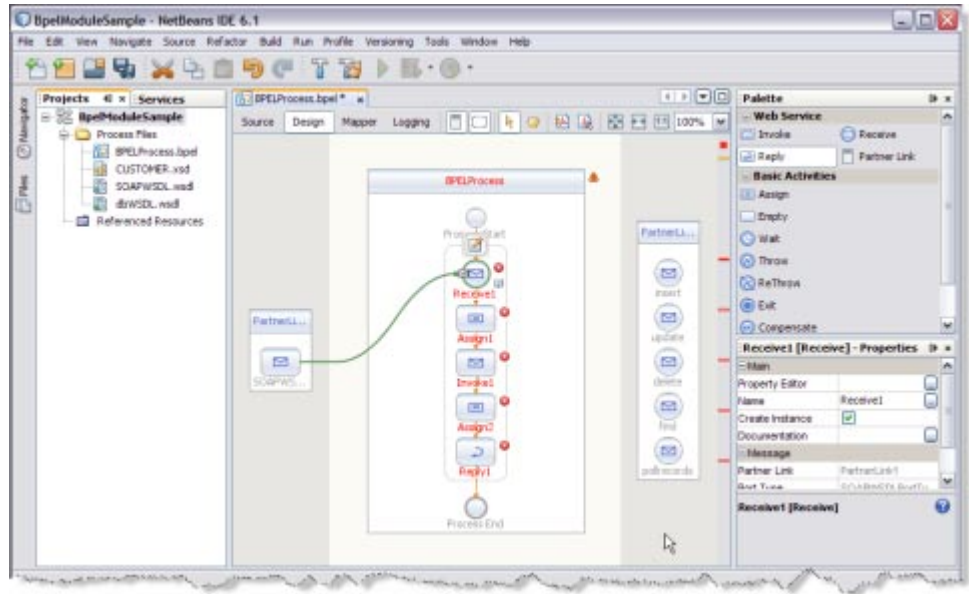
Note – All the fields are populated with the assigned values.

Input Variable — SOAPWSDLOperationIn



- 4 Click OK to close the Receive1 [Receive] - Property Editor.

5 Click Save All.



▼ To Edit the Web Service : Invoke1

1 Click Web Service — Invoke1 and click Edit.

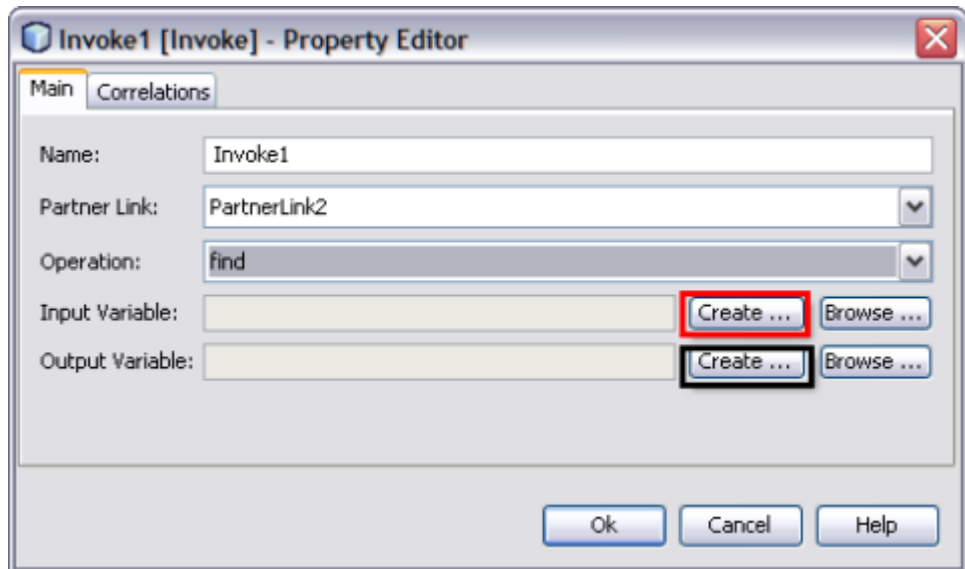
This opens the Invoke1 [Invoke] - Property Editor.

2 Select the properties from the Main tab. Select PartnerLink2 from the drop-down list.

Select any one of the Operations from the drop-down list.

- Insert
- Update
- Delete
- Find
- Pollrecords

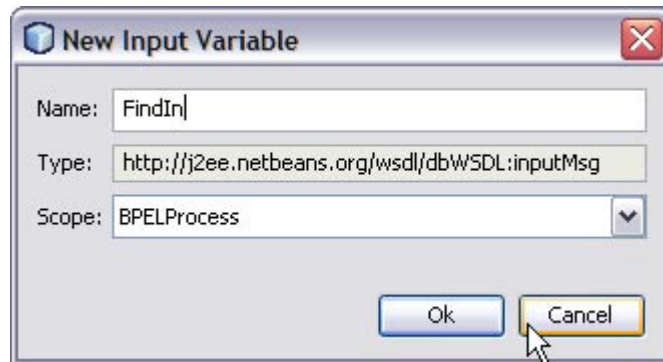
Select Operation — find from the drop-down list. The IDE fills in the Operation field.



3 Follow the steps to create a new input and an output variable.

a. Click the Create button next to the Input Variable field.

This opens the New Input Variable dialog box.



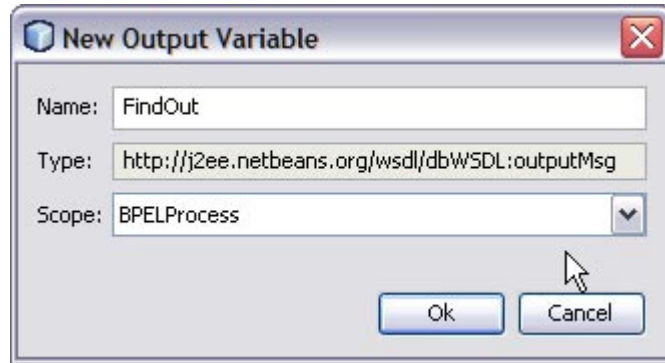
The Name, Type, and Scope are displayed, by default.

You can also change the value in the Name field.

b. Click OK to close the New Input Variable dialog box.

- c. Click the **Create** button next to the **Output Variable** field.

This opens the New Output Variable dialog box.



The Name, Type, and Scope are displayed, by default.

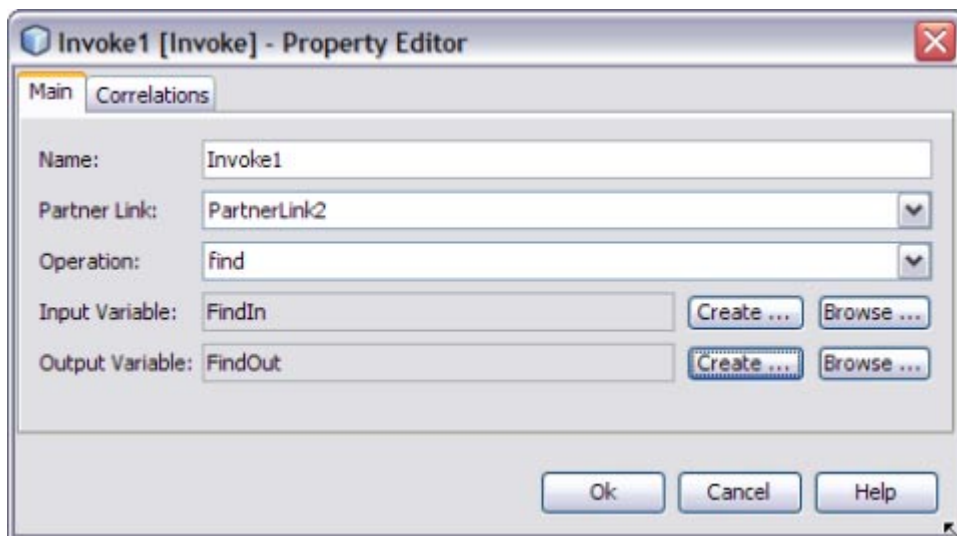
You can also change the value in the Name field.

- d. Click **OK** to close the **New Output Variable** dialog box.

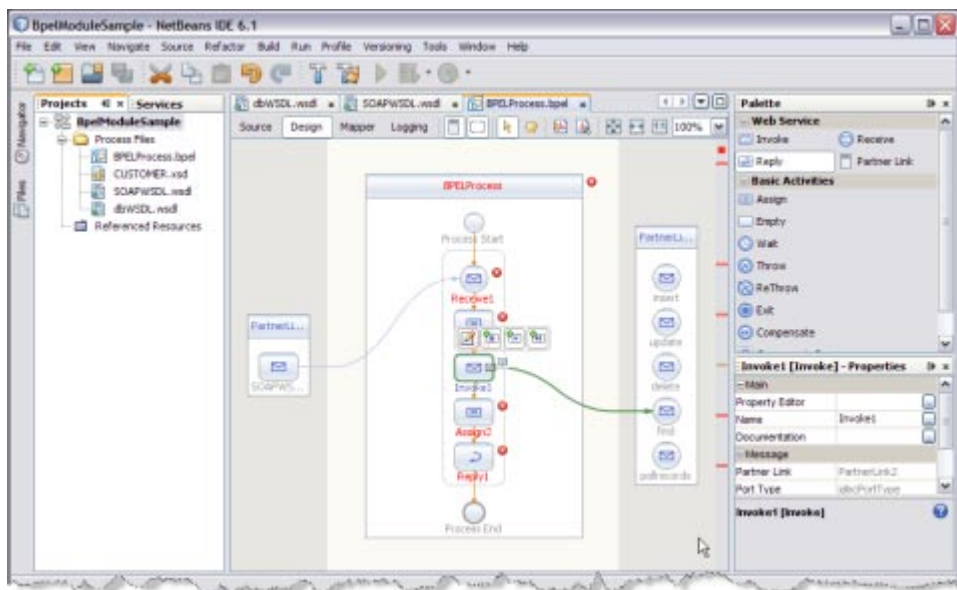
Note – All the fields are populated with the assigned values.

Select the following Variables:

- Input Variable : FindIn
 - Output Variable : FindOut
-



- 4 Click OK to close the Invoke1 [Invoke] - Property Editor.
- 5 Save the project.



▼ To Edit the Web Service : Reply1

- 1 Click Web Service : Reply1. Click Edit.

This opens the Reply1 [Reply] - Property Editor.

- 2 Select the properties from the Main tab. Select PartnerLink1 from the drop-down list.

The IDE fills in the Operation field with SOAPWSDLOperation.

Reply1 [Reply] - Property Editor

Main Correlations

Name: Reply1

Partner Link: PartnerLink1

Operation: SOAPWSDLOperation

☒ Normal Response

Output Variable: Create ... Browse ...

☐ Fault Response

Fault Name: Choose ...

Fault Variable: Create ... Browse ...

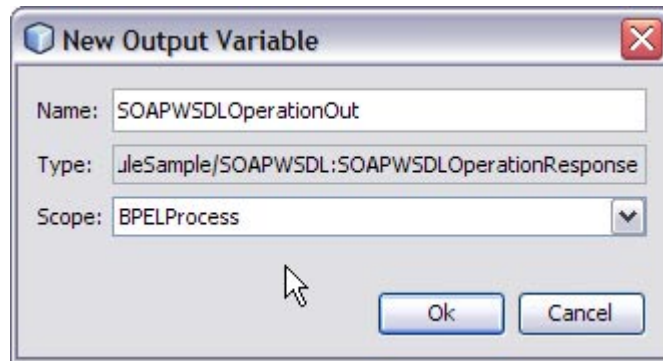
Ok Cancel Help

- 3 Follow the steps to create a New Output Variable.

- a. Make sure to select the Normal Response radio button.

- b. Click the **Create** button next to the **Output Variable** field.

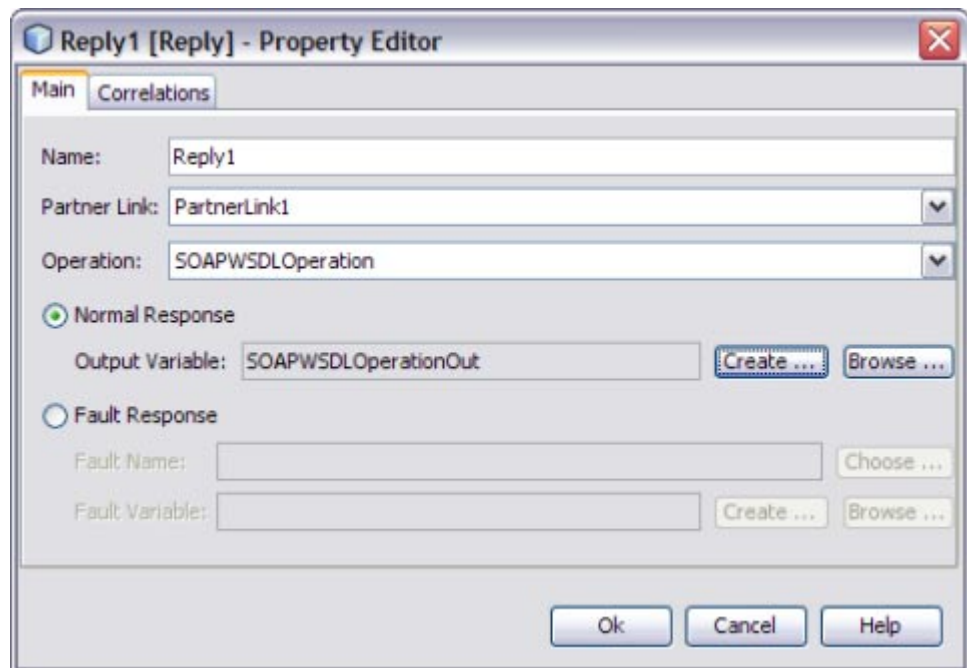
This opens the New Output Variable dialog box.



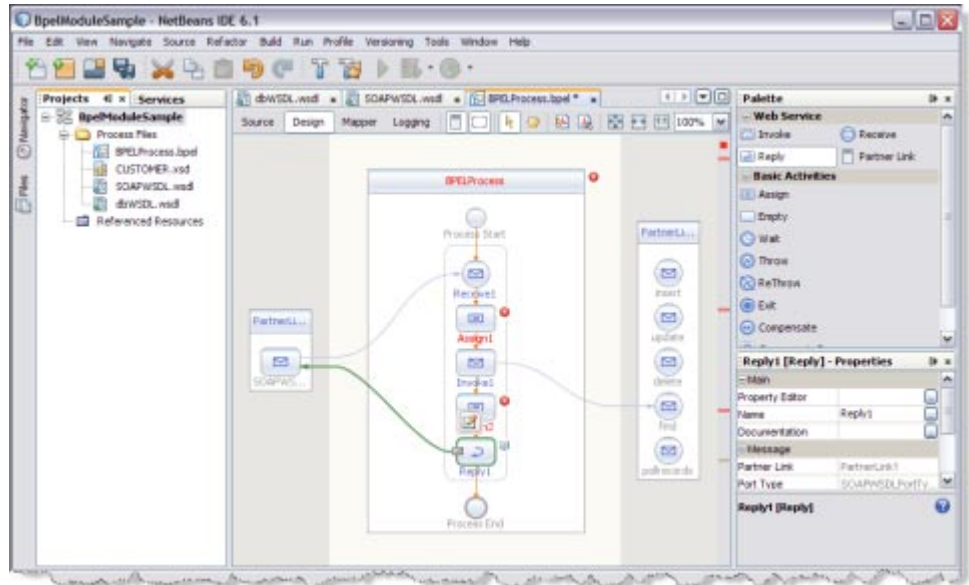
- c. Change the value in the **Name** field. This is optional.

Reply1 is displayed, by default.

- d. Click **OK**.



- 4 Click OK to close the Reply1 [Reply] - Property Editor.

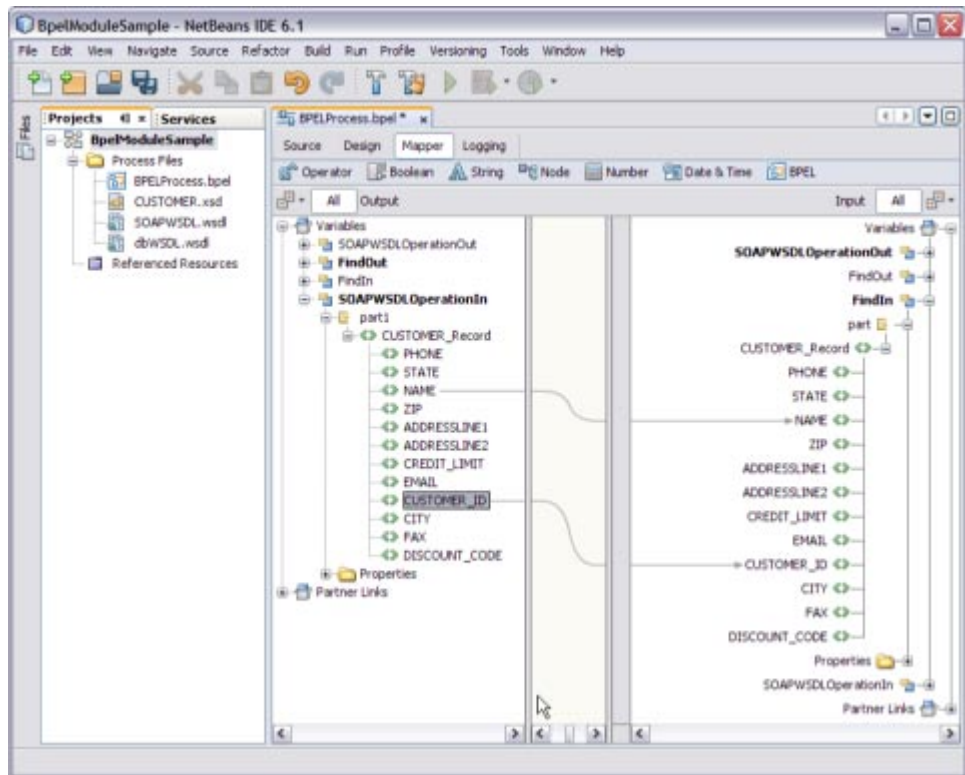


▼ To Edit the Basic Activities : Assign1

- 1 Double-click the Basic Activity : Assign1.
This displays the BPEL Mapper window.
- 2 Expand the node in the Source tree pane (the left pane) of the BPEL Mapper under Output —> Variables (for example, SOAPWSDLOperationIn).
A part1 node appears under the SOAPWSDLOperationIn node.
- 3 Expand the node in the Destination tree pane (the right pane) of the BPEL Mapper under Input —> Variables (for example, FindIn).
A part node appears under the FindIn node.
- 4 Select the node in the Source tree pane.
For example, SOAPWSDLOperationIn — part1 — CUSTOMER_Record.
- 5 Drag the selection and map it to the node in the Destination tree pane.
For example, FindIn — part — CUSTOMER_Record.

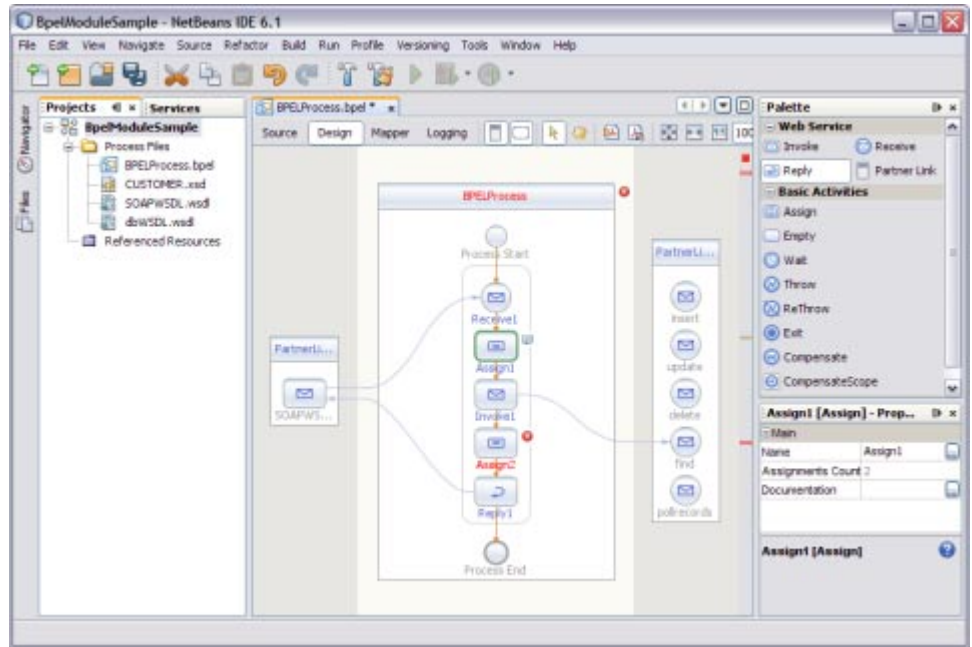
Map the following Variables:

- a. NAME — NAME
- b. CUSTOMER_ID — CUSTOMER_ID



6 Click the Design tab.

Note – A red icon marked against Basic Activities — Assign1 is not shown.



▼ To Edit the Basic Activities : Assign2

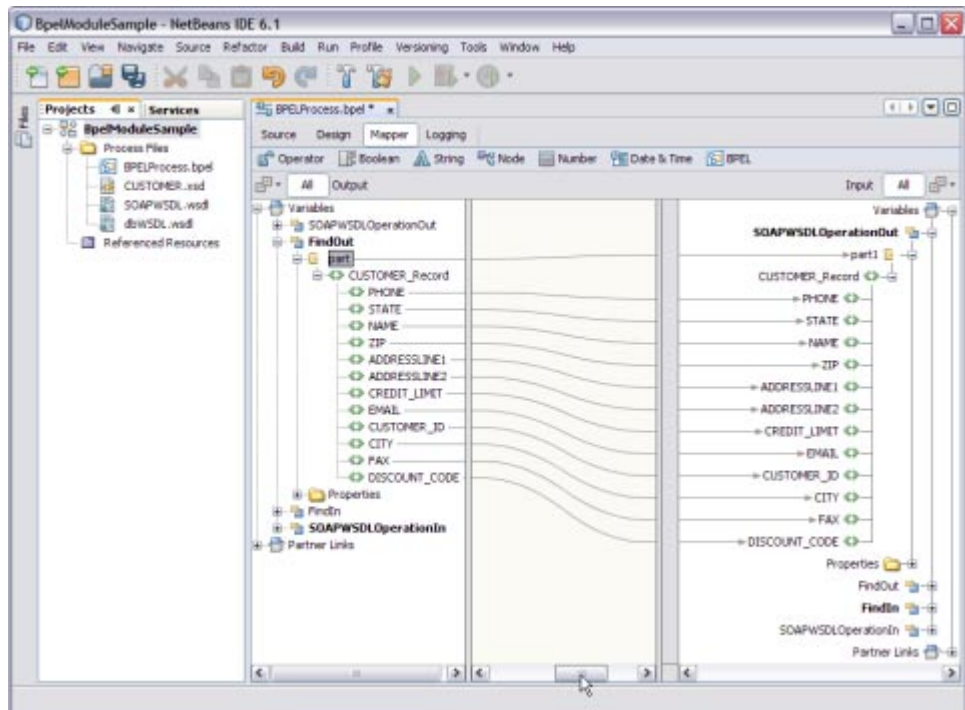
- 1 **Double-click the Basic Activity : Assign2.**
This displays the BPEL Mapper window.
- 2 **Expand the node in the Source tree pane (the left pane) of the BPEL Mapper under Output — Variables (for example, FindOut).**
- 3 **Expand the node in the Destination tree pane (the right pane) of the BPEL Mapper under Input — Variables (for example, SOAPWSLOptionOut).**
- 4 **Select the node in the Source tree pane.**
For example, FindOut : part : CUSTOMER_Record.
- 5 **Drag the selection to the node in the Destination tree pane.**
For example, SOAPWSLOptionOut : part1 : CUSTOMER_Record

Map the following variables:

- PHONE : PHONE
- STATE : STATE
- NAME : NAME

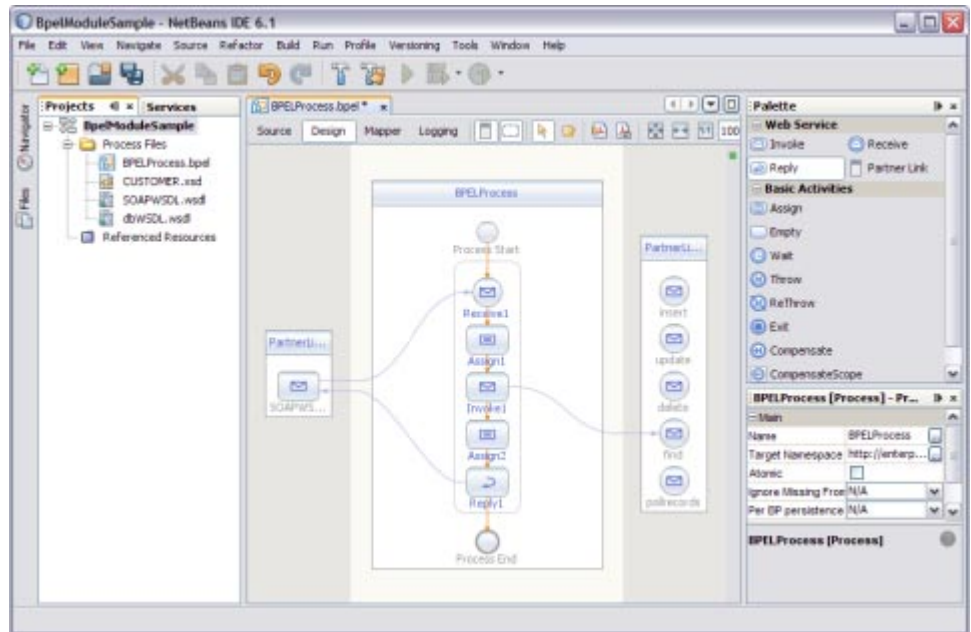
- ZIP : ZIP
- ADDRESSLINE1 — ADDRESSLINE1
- ADDRESSLINE2 — ADDRESSLINE2
- CREDIT_LIMIT — CREDIT_LIMIT
- EMAIL — EMAIL
- CUSTOMER_ID — CUSTOMER_ID
- CITY — CITY
- FAX — FAX
- DISCOUNT_CODE — DISCOUNT_CODE

Finally, map part : part1.



6 Click the Design tab.

The final output is as shown in the illustration.



7 Right-click the project node and select Clean and Build.

The following message is displayed.

BUILD SUCCESSFUL (total time: 2 seconds).

8 Click Save All.

Validating BPEL

The BPEL Designer has a built-in BPEL code validation functionality that helps create well-formed, valid and standard-compliant code. The code is checked for errors and is notified if validation fails.

▼ To Invoke Explicit Validation

To invoke explicit validation, perform any one of the following:

- 1 Right-click the source to invoke the pop-up menu in the Source view and choose Validate XML (Alt-Shift-F9).**

- 2 Click the **Validate XML** button (Alt-Shift-F9) on the Editor toolbar in the Design view.

Design View : Notifications

The user is notified about validation errors or success in the Output window, in the Design view, and in the Navigator.

The Design View

The Design view shows the results of both real-time and explicit validation in call-out windows on the diagram and the error stripe.

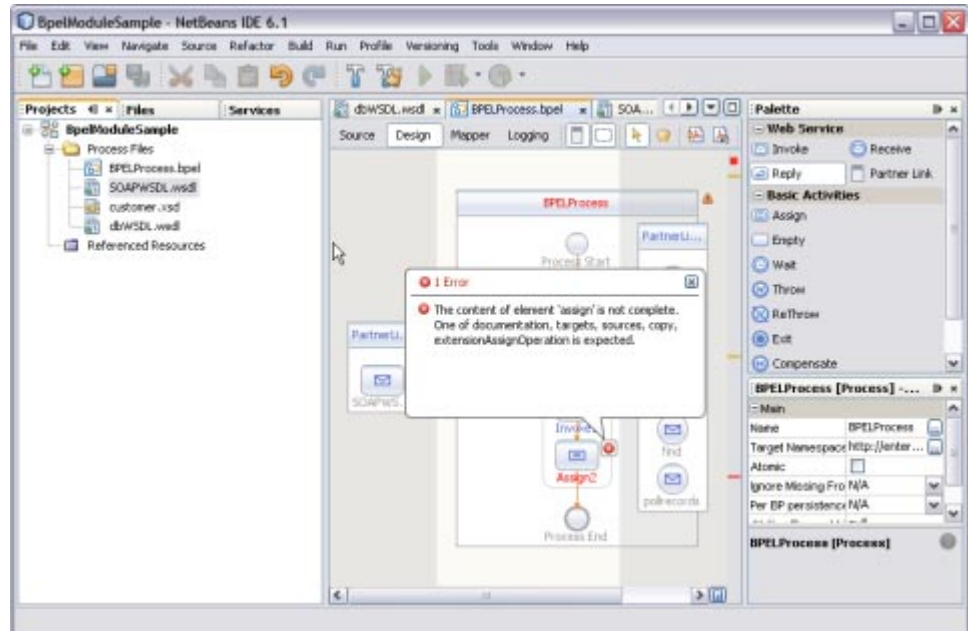
In the illustration,

Note – A red cross next to an element on the diagram means that the element has not passed validation and the output contains errors.

A yellow triangle with an exclamation mark means that the element has not passed validation and the output contains warnings.

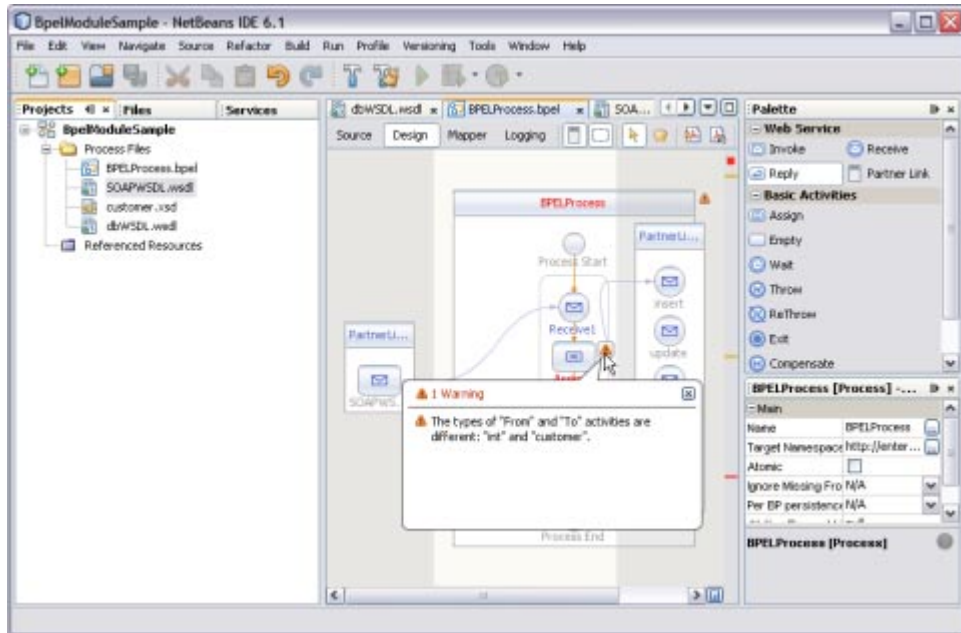
A red cross in the Design view means there are both errors and warnings.

If you click the cross or the triangle, a call-out window appears with a list of errors and/or warnings.



The call-out window includes messages related to validation in accordance with all the criteria listed above. Messages related to the real-time validation are constantly updated.

In the Design view, validation results are also shown by the error stripe, which is a strip to the right of the scroll bar that contains red marks if some elements have not passed validation. The error stripe represents the entire diagram, not just the portion that is currently displayed. You can immediately see if the BPEL process contains errors without having to scroll through the entire diagram. You can click a red mark to jump to the element that causes problems. If no errors are detected, the small square in the error stripe is green.



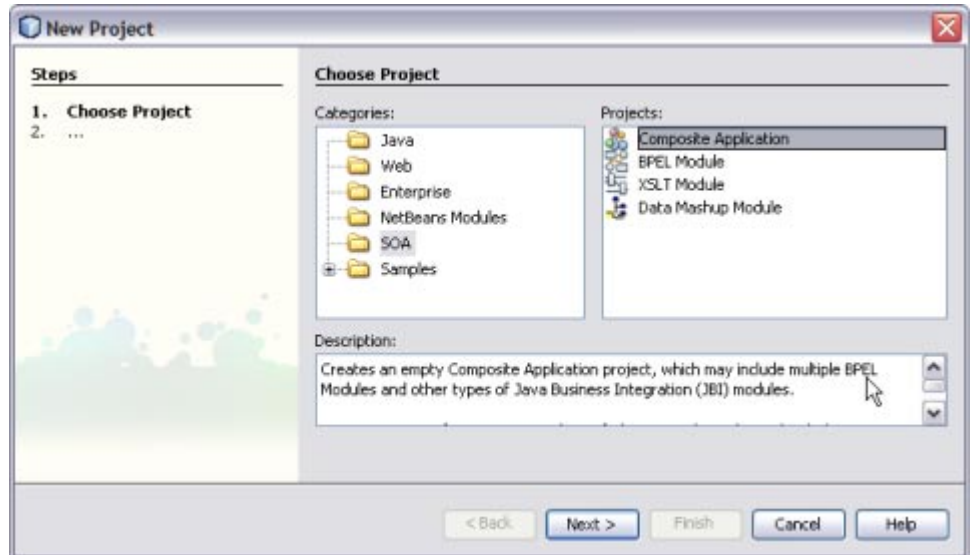
Creating the Composite Application Project

Add the JBI module to the BPEL Module project before deploying the Composite Application. Deploying the project makes the service assembly available to the application server, thus allowing its service units to run.

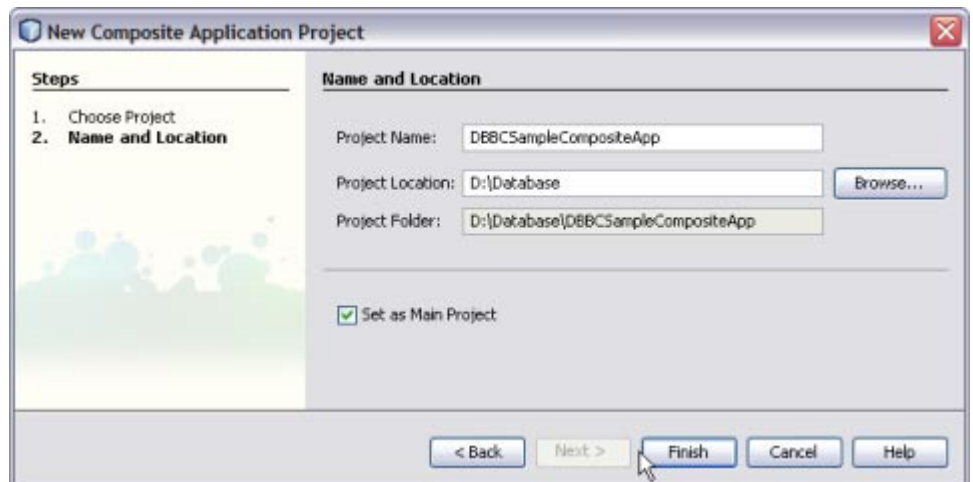
▼ To Create the Composite Application Project

- 1 Choose **File** —> **New Project** from the main menu.
This opens the New Project wizard.
- 2 Select the **SOA** node from the **Categories** list.
- 3 Select the **Composite Application** node from the **Projects** list.

- 4 Click Next.

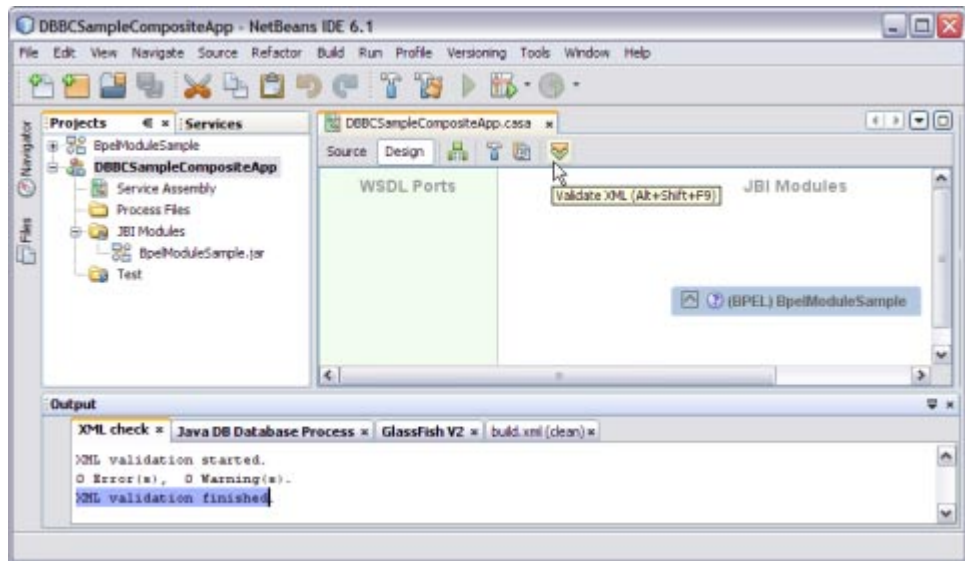


- 5 In the Project Name field, enter DBBCSampleCompositeApp.



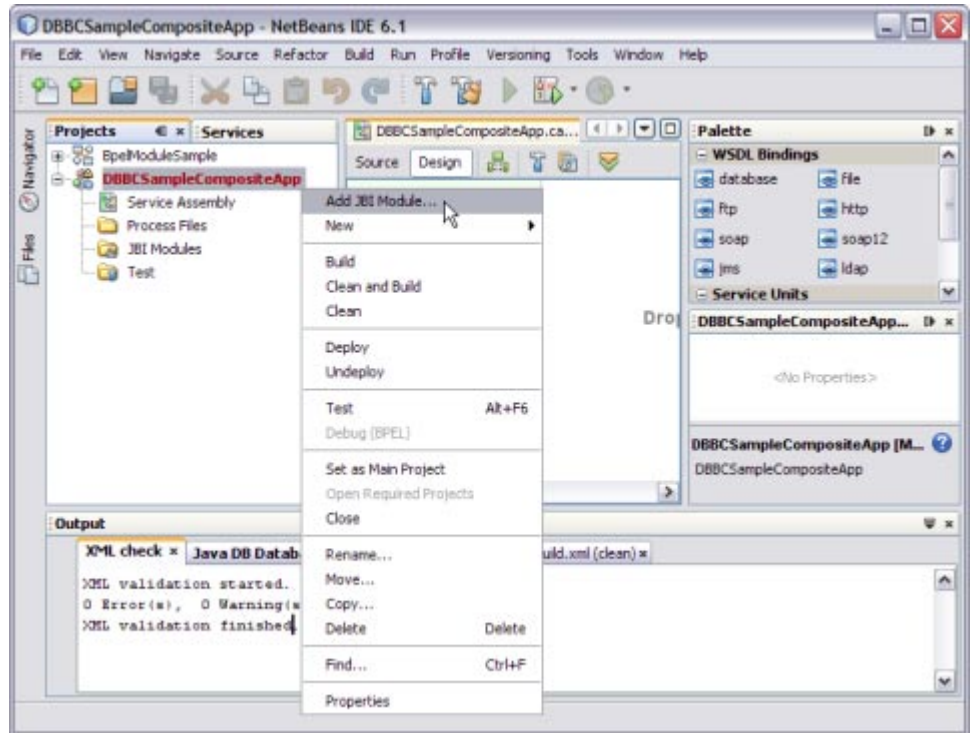
6 Click Finish.

The Projects window now contains a project node for a Composite Application project called DBBCSampleCompositeApp.



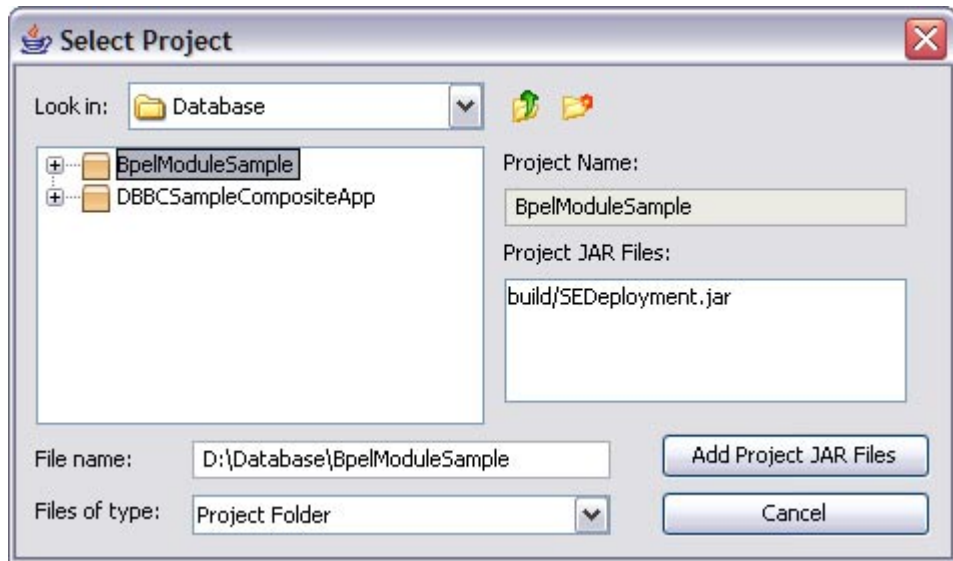
7 Right-click either the Composite Application Project node or expand the node and select JBI Modules.

8 Select Add JBI Module.

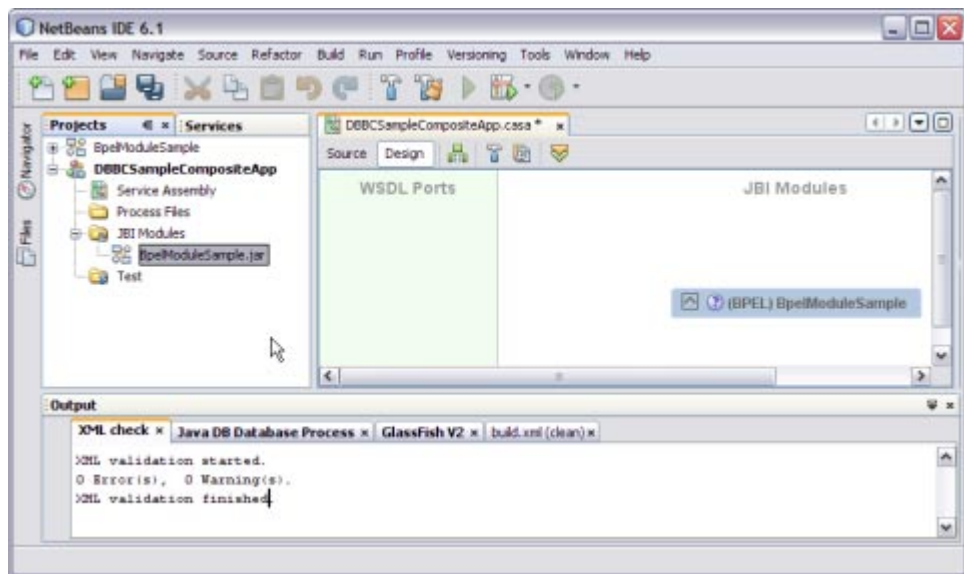


9 Select the BpelModuleSample Project. Click Add Project JAR Files.

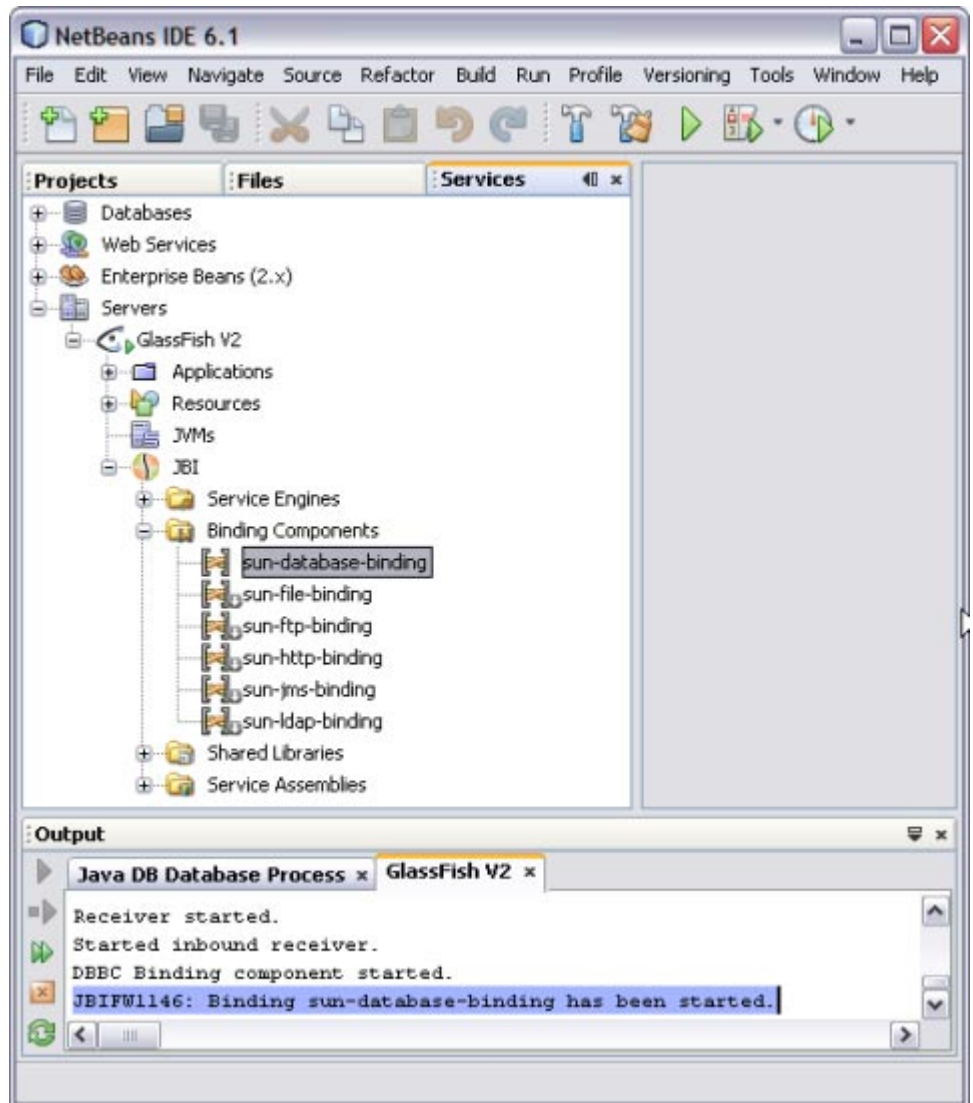
The Project JAR Files is build/SEDeployment.jar.



The BpelModuleSample.jar file is added into the project.



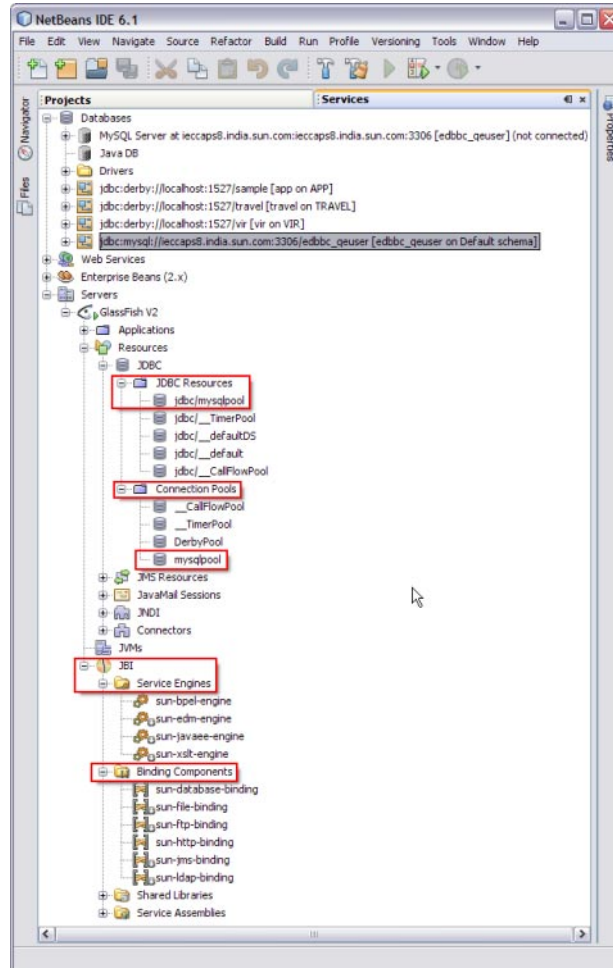
10 Click Save All.



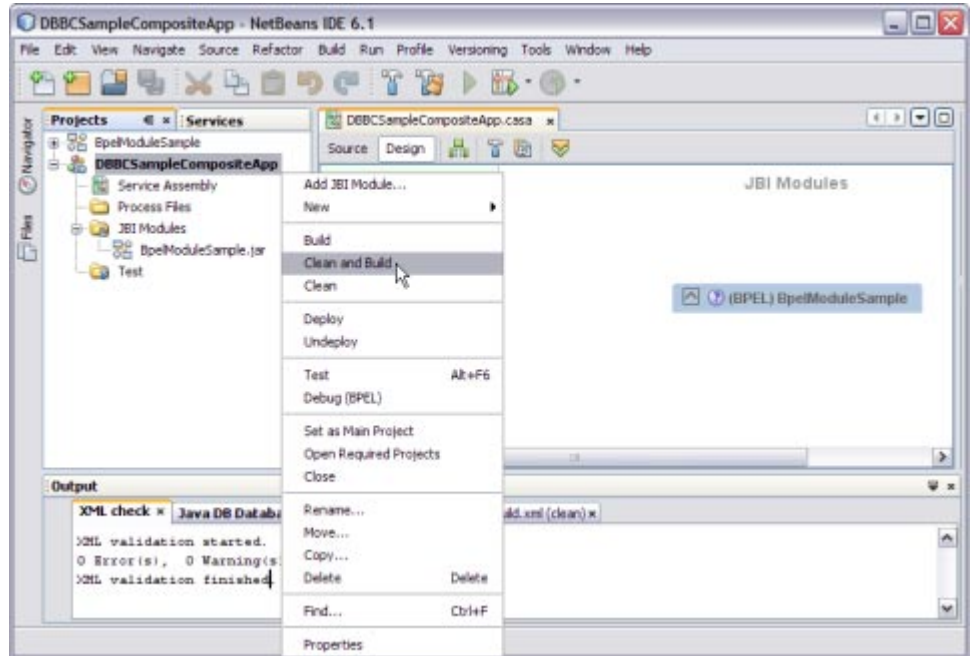
A detailed illustration is as shown on the deployed JBI Components.

- a. Expand IDE, open the Services window, expand the GlassFish V2 node, and expand the JBI node.
- b. Right-click sun-database-binding and click Start.

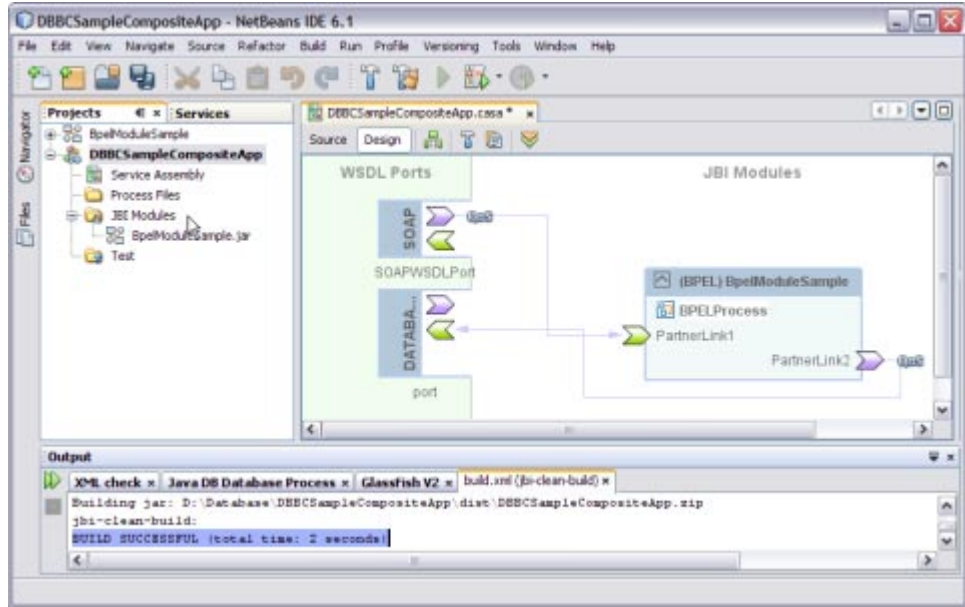
Oracle Java CAPS Database Binding Component Tutorial • March 2011



- 11 Right-click the DBBCSampleCompositeApp Composite Application project node and select Clean and Build.



The CASA Editor displays the build of DBBCSampleCompositeApp.



A message is displayed in the Output window:

BUILD SUCCESSFUL (total time: 2 seconds)

- 12 Click Save All.

Deploying and Testing the Composite Application

In this section, you deploy the Composite Application project and then test the deployed application. Start the GlassFish Application Server before deploying the project. See [“Starting the GlassFish V2 Application Server”](#) on page 10.

▼ To Deploy the Composite Application

Troubleshooting

- Stop the GlassFish V2 Application Server and Restart to ensure the connection.
- Logout of the GlassFish Admin Console and Login.
- Go to Resources —> JDBC —> Connection Pools —> Choose the defined Connection Pool.
- Click the Connection Pool.

The Edit Connection Pool screen is displayed.

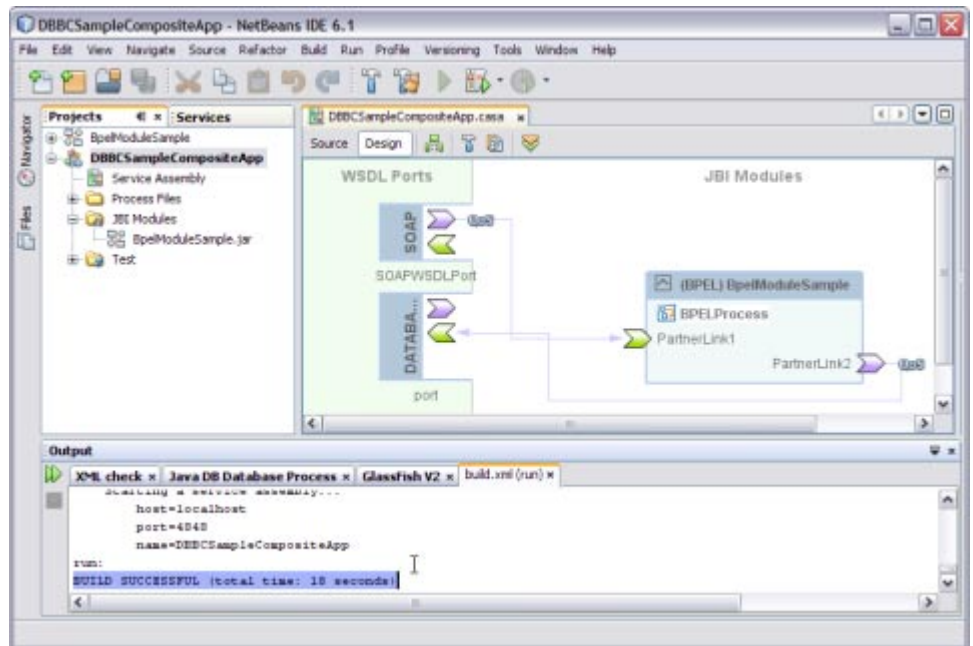
- Ping the Connection Pool.

The following confirmation message appears.

Ping Successful



- 1 Select the project node in the Projects window.
For example, DBBCSampleCompositeApp
- 2 Right-click and choose Deploy.



After successful deployment of the project the following message appears in the Output window:

BUILD SUCCESSFUL (total time: 18 seconds)

▼ To Test the Composite Application

- 1 Click Test.
- 2 Right-click and select New Test Case.
- 3 Enter the Test Case Name.
- 4 Click Next.
- 5 Select the WSDL Document.
 - a. Select the following:
For example, SOAPWSDL.wsdl from BpelModuleSample : Process Files.

Note – One WSDL document must be selected.

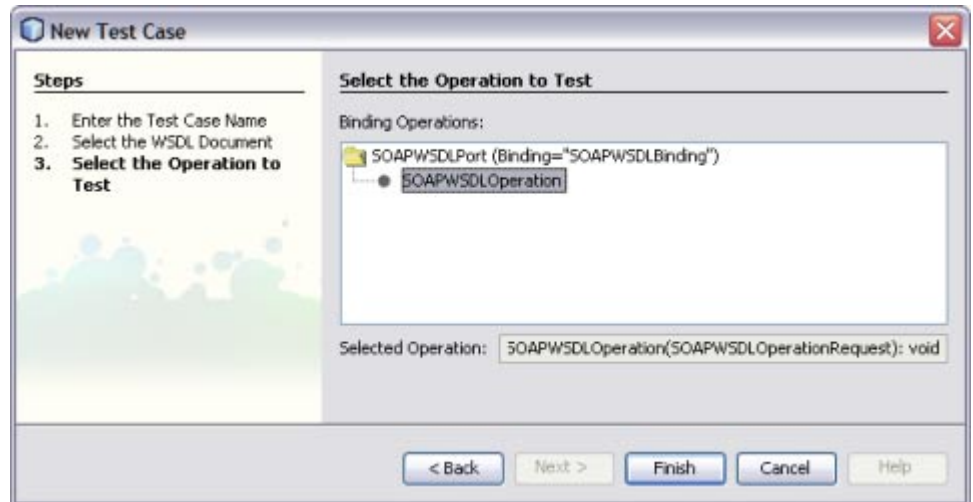


- b. Click Next.

c. **Select the following:**

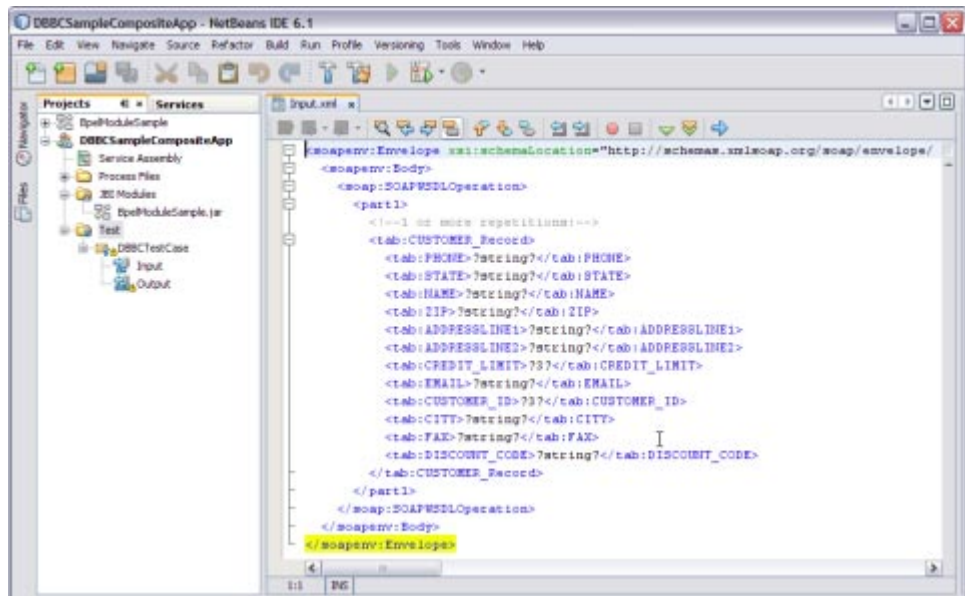
SOAPWSDLOperation from SOAPWSDLPort (Binding="SOAPWSDLBinding")

Note – One operation to text must be selected.



6 Click Finish.

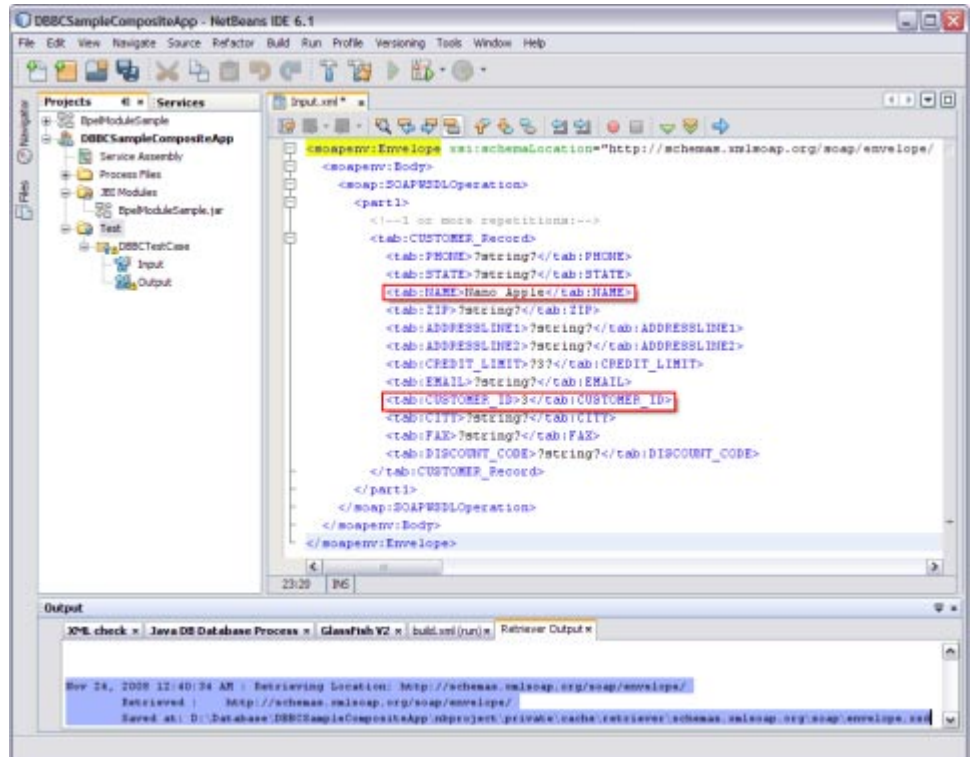
The Source Code is as shown.



7 Enter the string values.

For example:

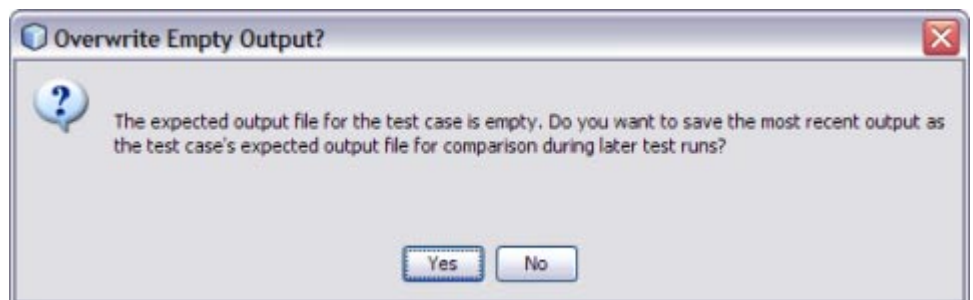
- NAME — Nano Apple
- CUSTOMER_ID — 3



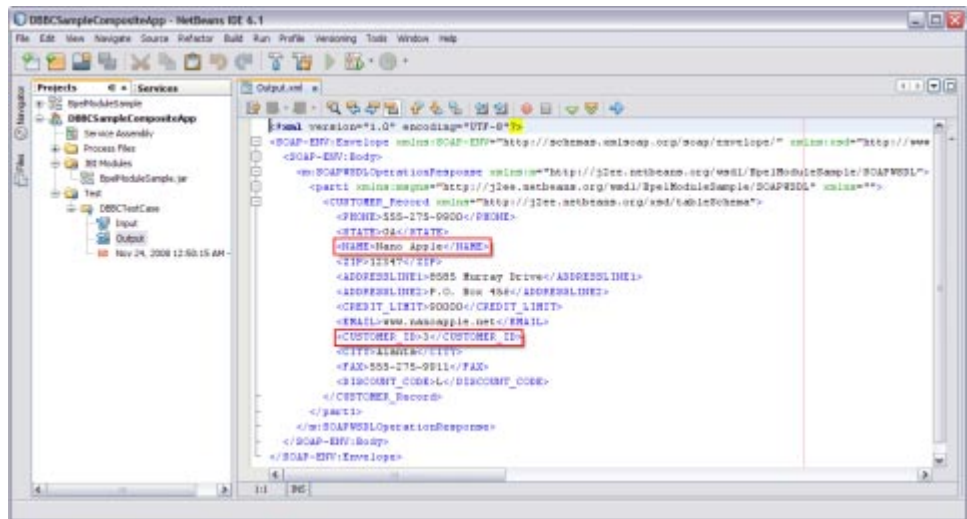
When you enter the string values in the Input.xml window, the output console displays Retrieve.xml. This window displays the location to save the data.

8 Right-click the Test Case and click Run.

A confirmation box to overwrite the Output is displayed. Click Yes to overwrite.



The Output is as shown in the illustration.



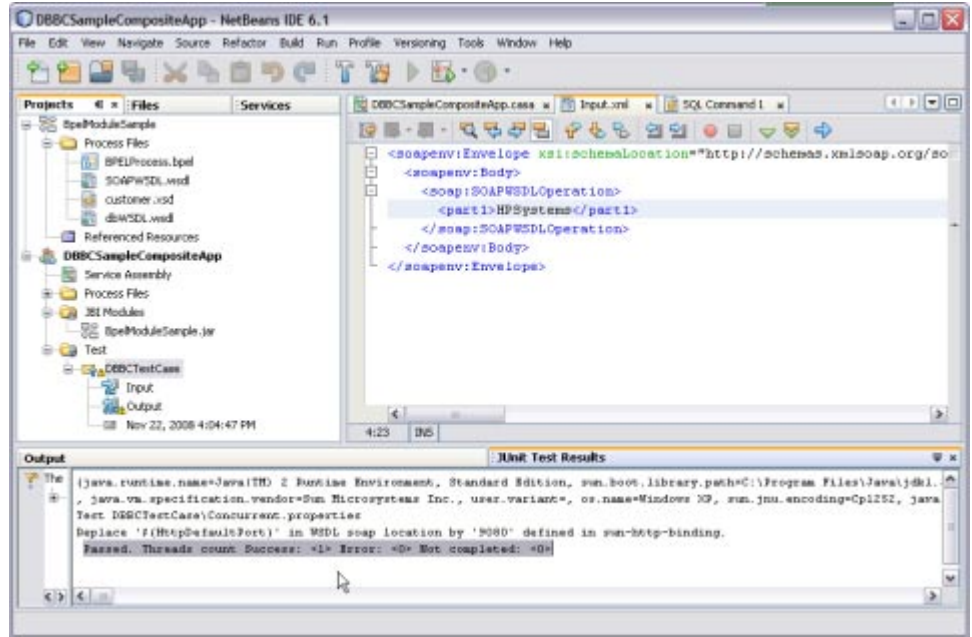
Debug the Test Case

You can debug the test case.

▼ To Debug the Test Case

- Right-click on the Test Case and select Debug.

A message is displayed as shown in the illustration.



For a demonstration, see

<http://wiki.open-esb.java.net/Wiki.jsp?page=DatabaseBindingComponentFlightDetails>.

Creating a BPEL Module Process Using Prepared Statements

A Prepared Statement represents a SQL statement that has been compiled. Prepared statements can be used to perform insert, update, delete and query operations. Prepared statements provide the ability to create queries that are more secure, have better performance, and are more convenient to write. They come in two flavors: bound parameter prepared statements, and bound result prepared statements.

Prepared statements allow query templates to be created and then stored on the database server. When a query needs to be made, data to fill in the template is sent to the database server, and a complete query is formed and then executed.

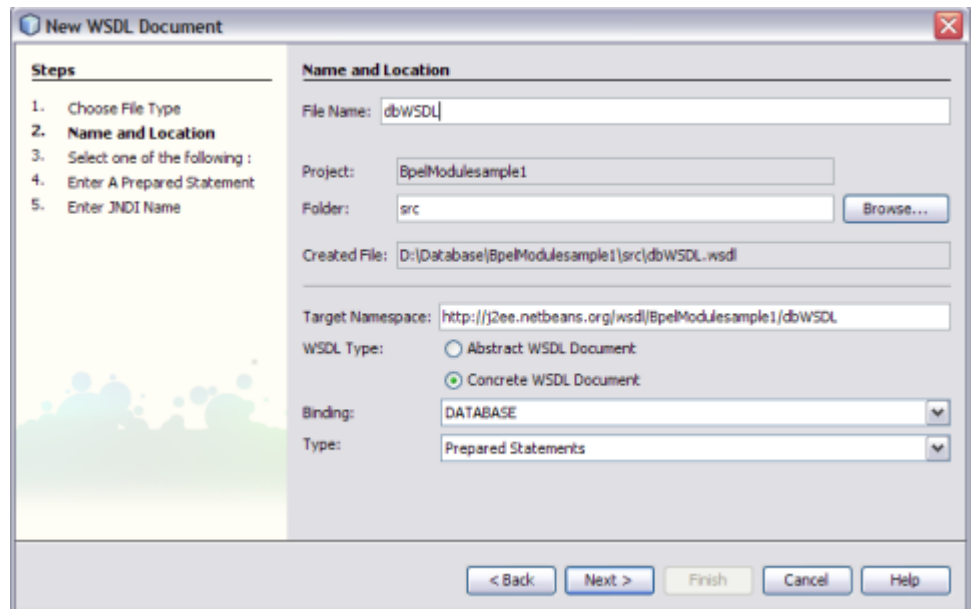
The basic process for creating and using bound parameter prepared statements is simple. A query template is created and sent to the database server. The database server receives the query template, validates it to ensure that it is well-formed, parses it to ensure that it is meaningful,

and stores it in a special buffer. It then returns a special handle that can later be used to reference the prepared statement. When a query needs to be made, data to fill in the template is sent to the database server, and then a complete query is formed and then executed.

This process has some very important behaviors wrapped up in it. The body of the query is only sent to the database server once. On requests to execute the query, only the data to fill in the template needs to be delivered to the database server. Most of the work required to validate and parse the query only needs to be done a single time, instead of each time that the query is executed. Additionally, for queries that contain a small amount of data, the overhead of sending the query is greatly reduced.

For steps, see the following links:

1. [“Creating a WSDL Document For Type : DATABASE” on page 24](#)
 - a. **Select Type — Prepared Statement.**



- b. **Enter a valid Prepared Statement.**
- c. **Click Discover Parameters to view types for the parameters.**
- d. **Click Discover Resultset to discover the Resultset types.**

Steps

1. Choose File Type
2. Name and Location
3. Select one of the following :
4. **Enter A Prepared Statement**
5. Enter JNDI Name

Enter A Prepared Statement

Prepared Statement Editor

Enter a valid prepared statement

Select * from CUSTOMER where CUSTOMER_ID=? and NAME=?

Discover Parameters

Parameters

Parameter Name	Parameter Type
param1	int
param2	java.lang.String

Discover Resultset

Resultset

Column Name	Column Type
CUSTOMER_ID	int
DISCOUNT_CODE	java.lang.String
ZIP	java.lang.String
NAME	java.lang.String
ADDRESSLINE1	java.lang.String
ADDRESSLINE2	java.lang.String
CITY	java.lang.String
STATE	java.lang.String

Status:

< Back Next > Finish Cancel Help

- e. Click Next.
 - f. Enter JNDI Name.
 - g. Click Finish.
2. “Creating a WSDL Document For Type SOAP” on page 34.
 3. “Creating a BPEL Process” on page 41.
 4. “Validating BPEL” on page 61.
 5. “Design View : Notifications” on page 62.
 6. “Creating the Composite Application Project” on page 64.
 7. “Deploying and Testing the Composite Application” on page 72.

For a demonstration, see

<http://wiki.open-esb.java.net/Wiki.jsp?page=DatabaseBindingComponentPollScenario>.

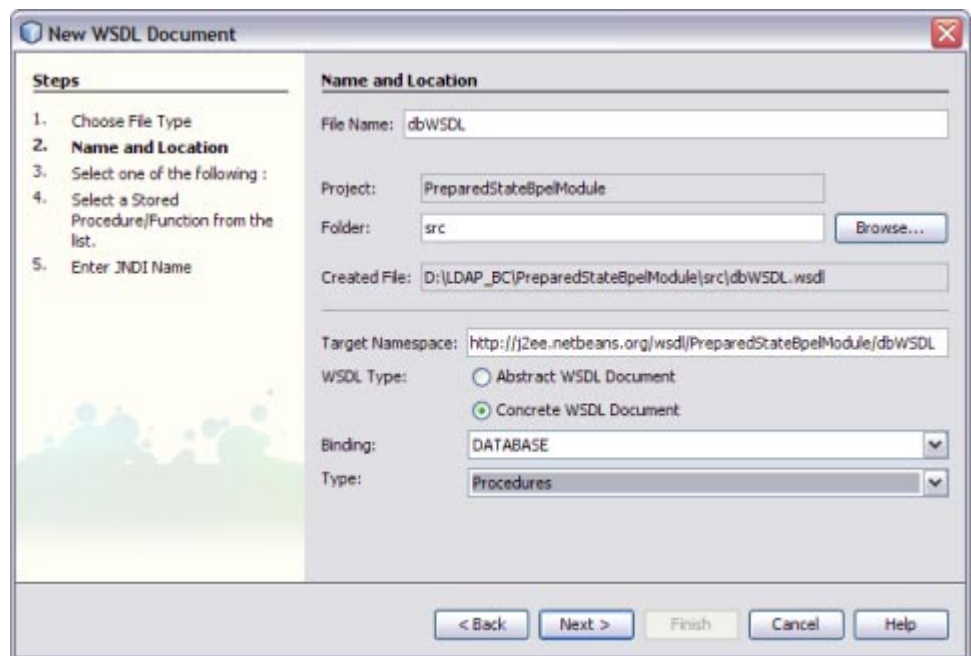
Creating a BPEL Module Project Using Procedures

A Procedure represents a database stored procedure. Fields correspond to the arguments of a stored procedure while methods are the operations that you can apply. It allows you to execute a stored procedure.

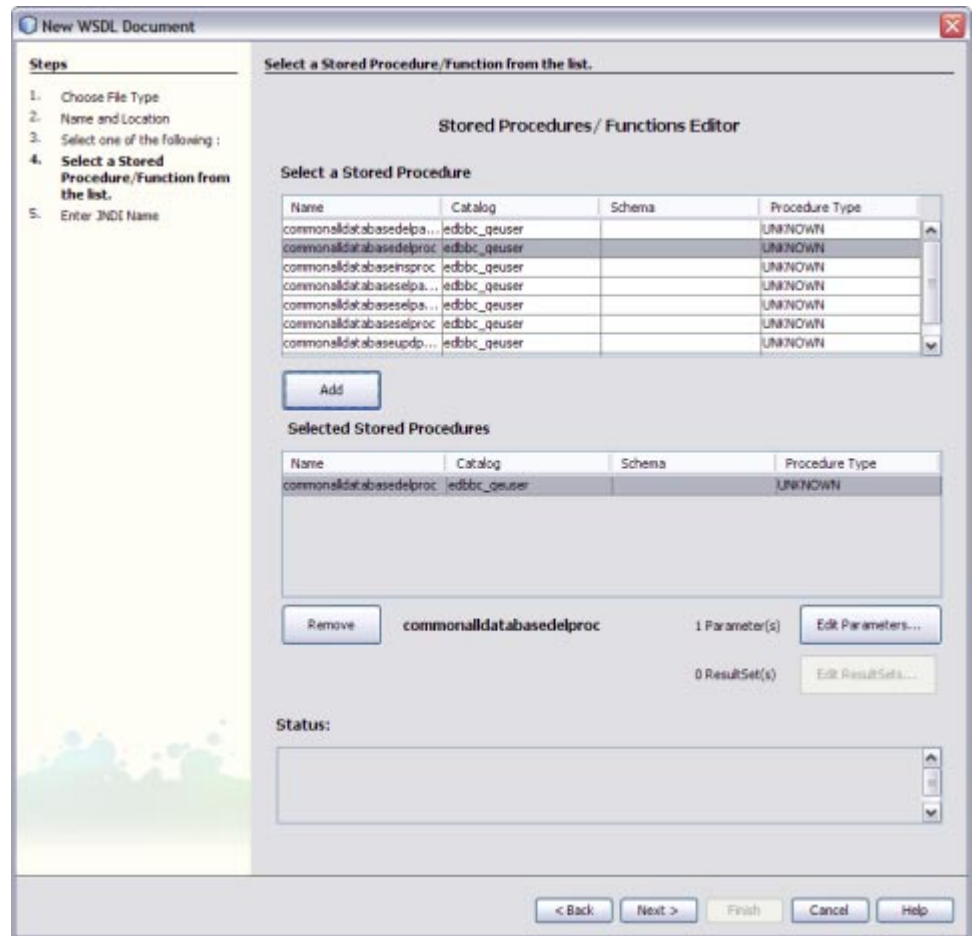
Note – Select a Procedure from the list and add the stored procedure to the Selected Stored Procedures console.

For the procedure, see the following links:

1. [“Creating a WSDL Document For Type : DATABASE” on page 24](#)
 - a. **Select Type — Procedures**



- b. **Click Next.**
- c. **Select the URL from the drop-down list.**
- d. **Select a Stored Procedure and click Add to move the selected Stored Procedure to the Selected Stored Procedure panel.**



- e. Click Next.
 - f. Enter JNDI Name.
 - g. Click Finish.
2. “Creating a WSDL Document For Type SOAP” on page 34
 3. “Creating a BPEL Process” on page 41.
 4. “Validating BPEL” on page 61.
 5. “Design View : Notifications” on page 62.
 6. “Creating the Composite Application Project” on page 64.
 7. “Deploying and Testing the Composite Application” on page 72.

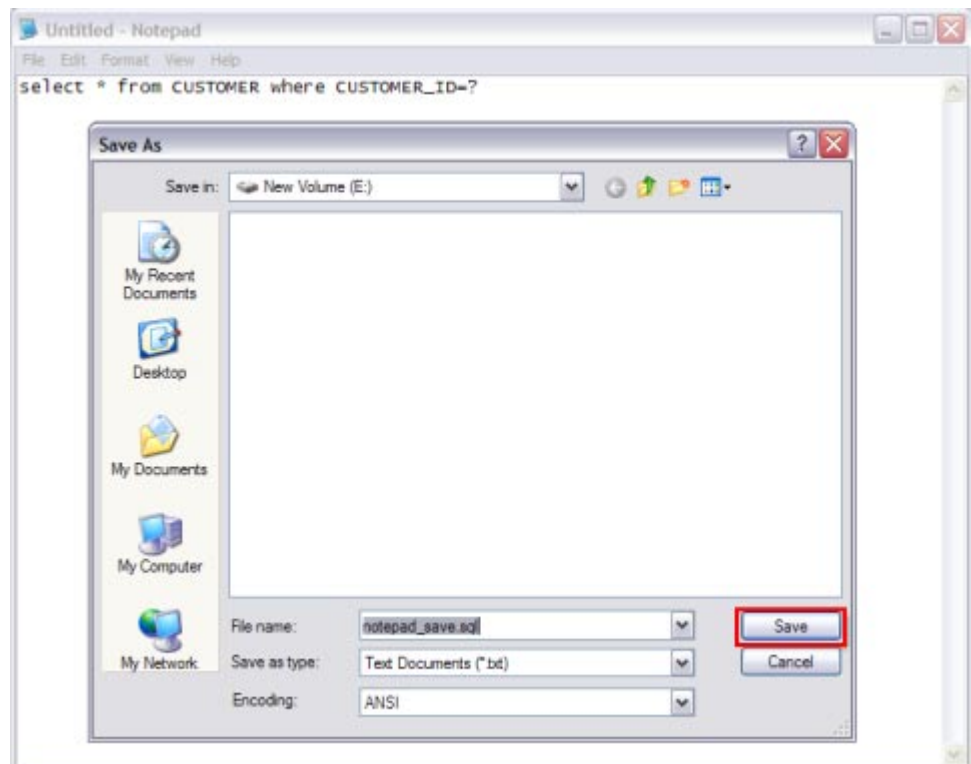
For a demonstration, see
<http://wiki.open-esb.java.net/Wiki.jsp?page=DatabaseComponentStoredProcedure>.

Creating a BPEL Module Project Using SQL File

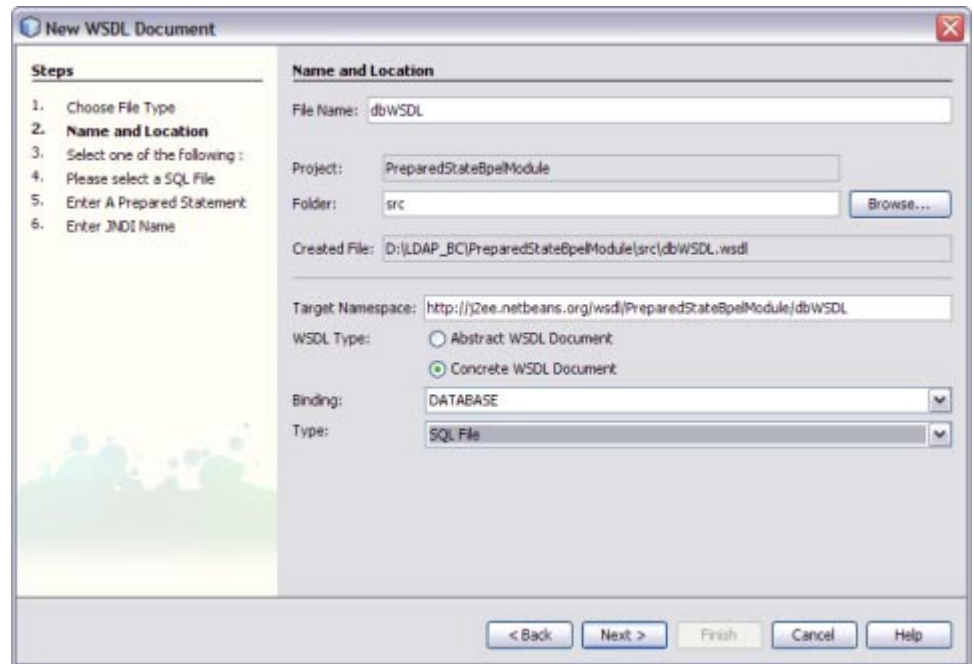
This is the script that contains the data definition statements for the MySQL database. The file filter for the file open dialog window defaults to *.sql so you should only be able to view files with the .sql extension.

For procedure, see the following links:

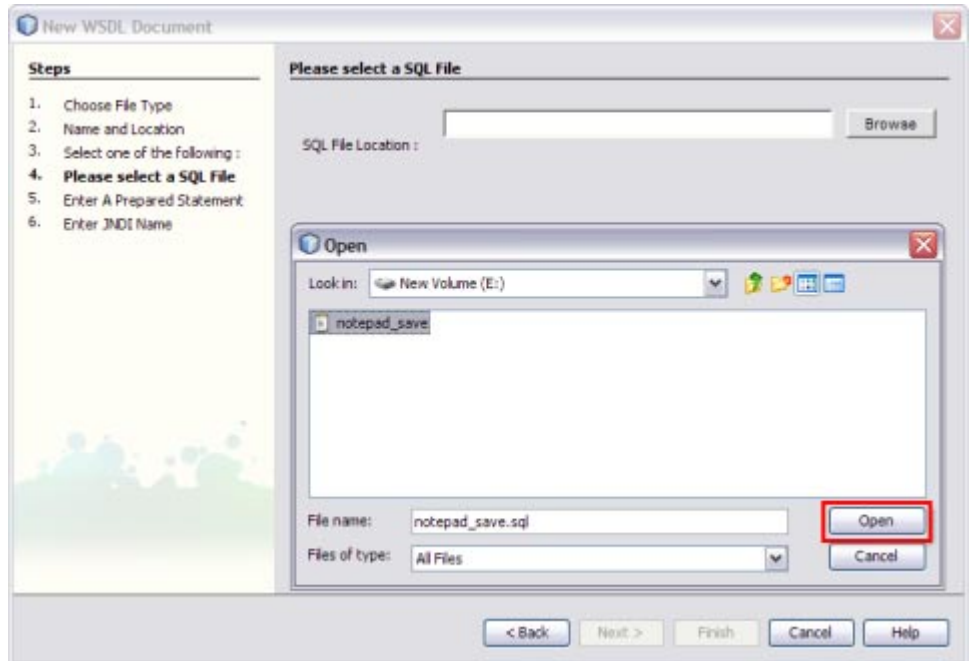
1. [“Creating a WSDL Document For Type : DATABASE” on page 24.](#)
 - a. **Create a .sql file using Notepad.**
 - b. **Save the file in a folder location.**



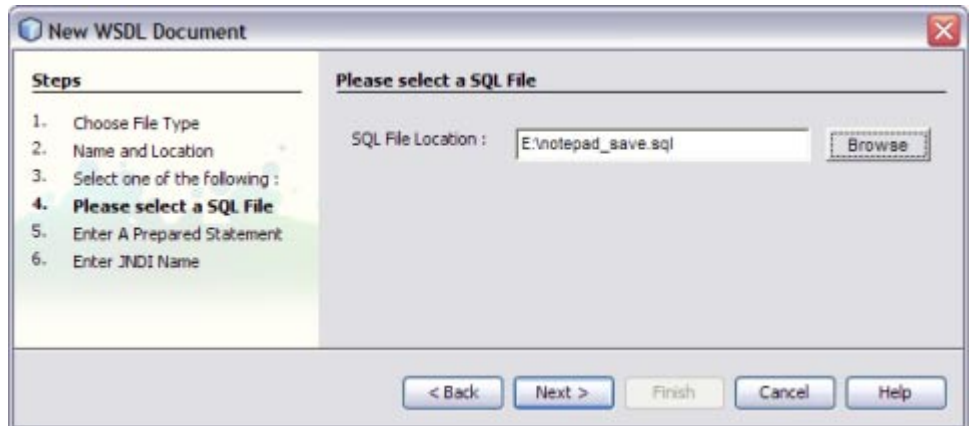
- c. **Choose Type — SQL File.**



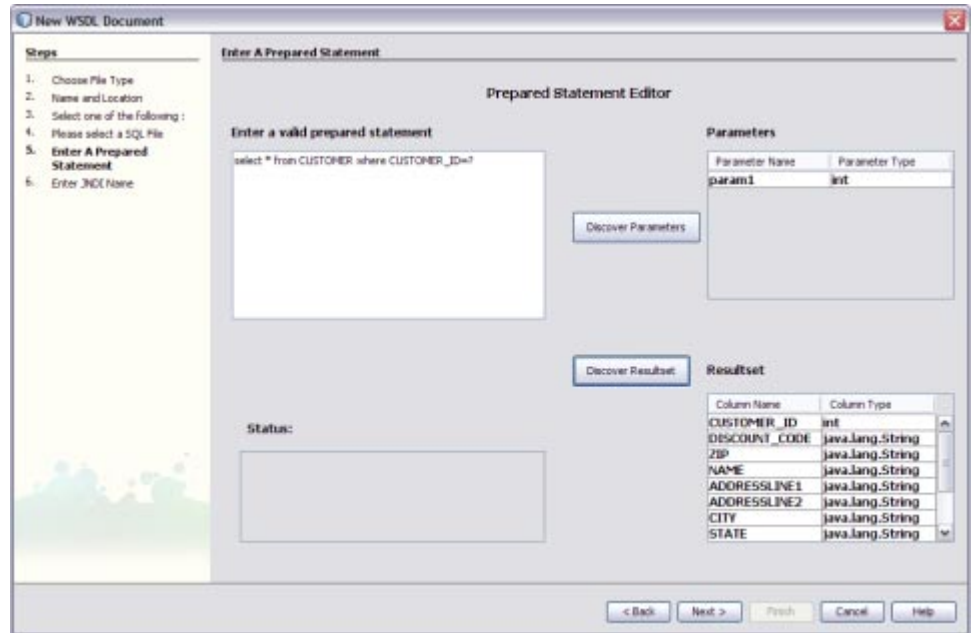
- d. Click Browse to select the saved .sql file.



The selected file is displayed as shown in the Figure.



- e. Click Discover Parameters to view types for the parameters.
- f. Click Discover Resultset to discover the Resultset types.



- g. **Click Next.**
 - h. **Enter JNDI Name.**
 - i. **Click Finish.**
2. “Creating a WSDL Document For Type SOAP” on page 34.
 3. “Creating a BPEL Process” on page 41.
 4. “Validating BPEL” on page 61.
 5. “Design View : Notifications” on page 62.
 6. “Creating the Composite Application Project” on page 64.
 7. “Deploying and Testing the Composite Application” on page 72.

For a demonstration, see

<http://wiki.open-esb.java.net/Wiki.jsp?page=BuildSampleProjectSQLFileDatabaseWSDL>.

