Using Java (Axis2) wrapper classes

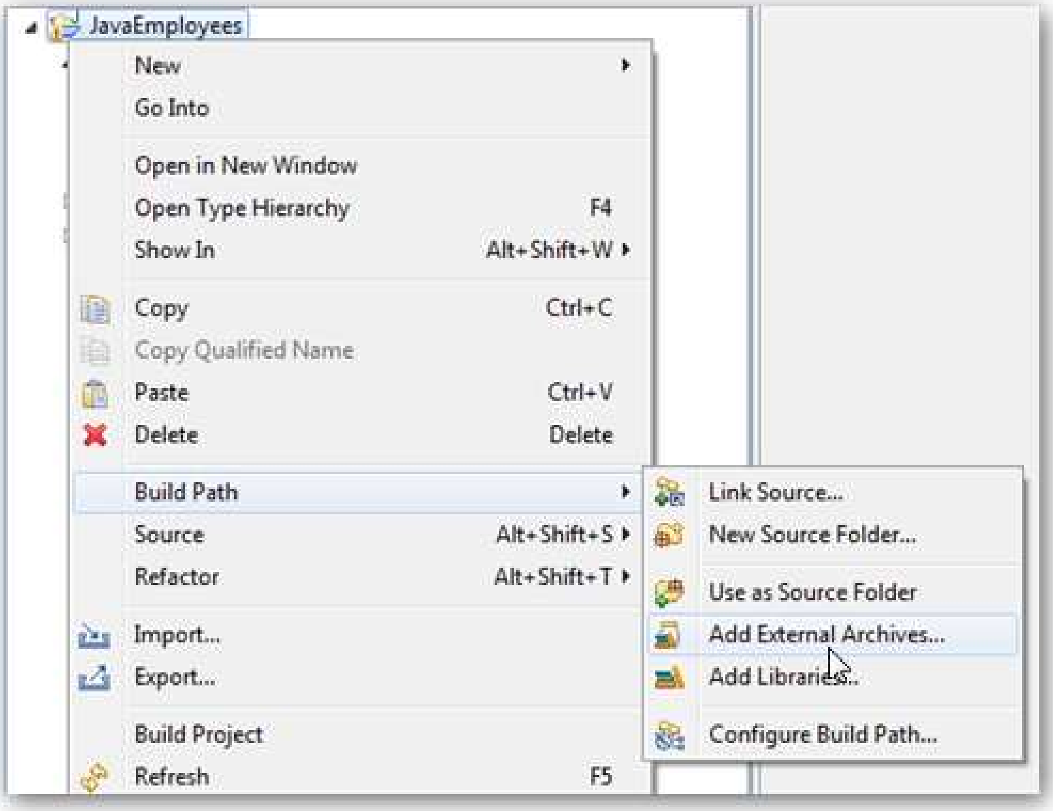
Tutorial: Generate a Java wrapper for the "Employees" file

Java wrapper/stub classes are generated using the Apache Axis2 feature WSDL2Java.

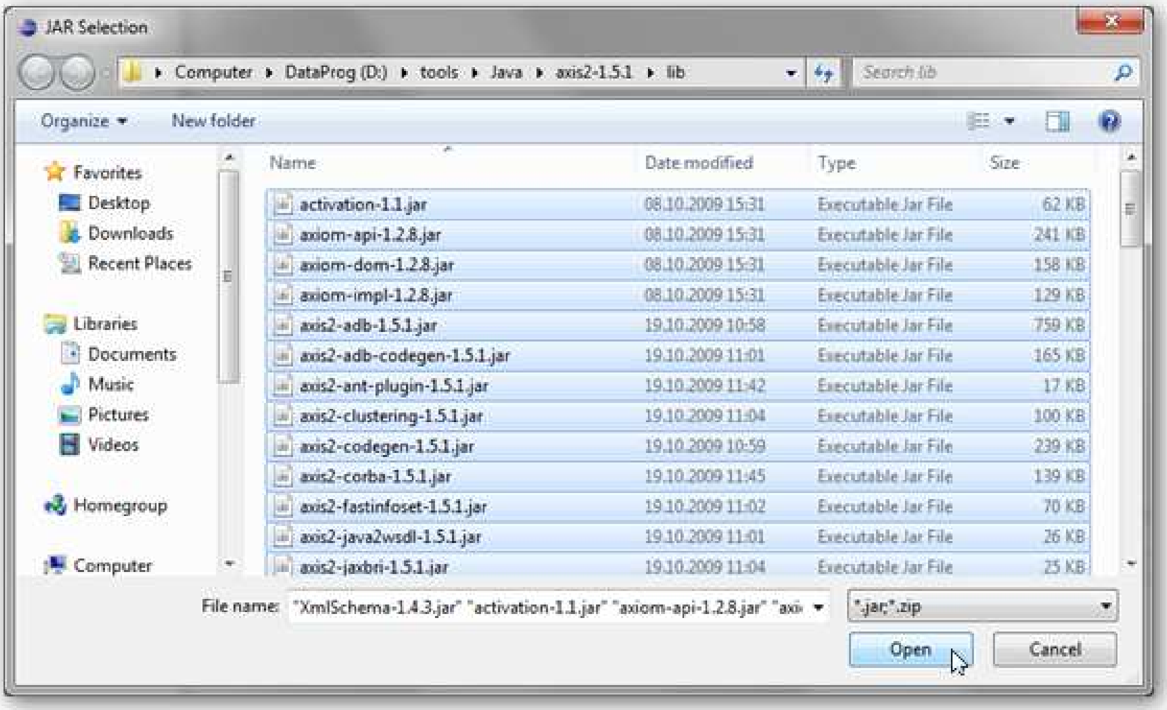
If you do not have it already, download and install the latest Axis2 kit.

These are the steps required to generate the Java wrapper classes for the "adabas_EmployeesMini" DataView supplied with SOA Gateway:

1. Create a new Java-project (refer to Getting started with Eclipse), name it "JavaEmployees"
2. Right-click the "JavaEmployees" project folder, select "Build Path", then "Add External Archives..:"



Add all .jar files from the axis2 "lib" directory to the project’s Build-Path.



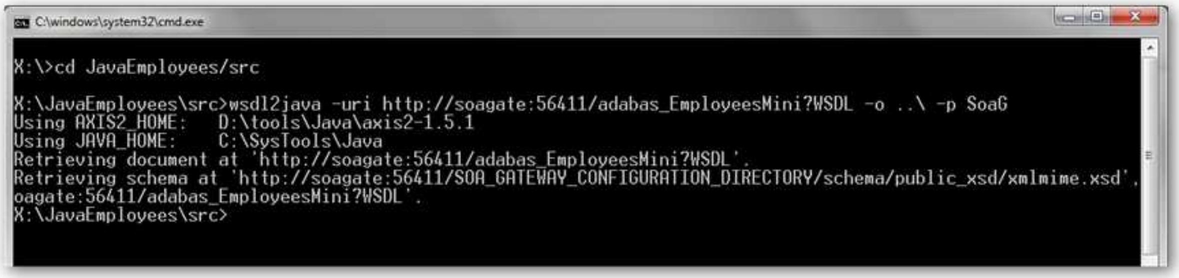
Name	Date modified	Type	Size
activation-1.1.jar	08.10.2009 15:31	Executable Jar File	62 KB
axiom-api-1.2.8.jar	08.10.2009 15:31	Executable Jar File	241 KB
axiom-dom-1.2.8.jar	08.10.2009 15:31	Executable Jar File	158 KB
axiom-impl-1.2.8.jar	08.10.2009 15:31	Executable Jar File	129 KB
axis2-adb-1.5.1.jar	19.10.2009 10:58	Executable Jar File	759 KB
axis2-adb-codegen-1.5.1.jar	19.10.2009 11:01	Executable Jar File	165 KB
axis2-ant-plugin-1.5.1.jar	19.10.2009 11:42	Executable Jar File	17 KB
axis2-clustering-1.5.1.jar	19.10.2009 11:04	Executable Jar File	100 KB
axis2-codegen-1.5.1.jar	19.10.2009 10:59	Executable Jar File	239 KB
axis2-corba-1.5.1.jar	19.10.2009 11:45	Executable Jar File	139 KB
axis2-fastinfoset-1.5.1.jar	19.10.2009 11:02	Executable Jar File	70 KB
axis2-java2wsdl-1.5.1.jar	19.10.2009 11:01	Executable Jar File	26 KB
axis2-jaxbri-1.5.1.jar	19.10.2009 11:04	Executable Jar File	25 KB

File name: "XmlSchema-1.4.3.jar" "activation-1.1.jar" "axiom-api-1.2.8.jar" "axi

.jar;.zip

3. Open a command prompt (aka "DOS box"), change to the "JavaEmployees\src" directory and run the following command

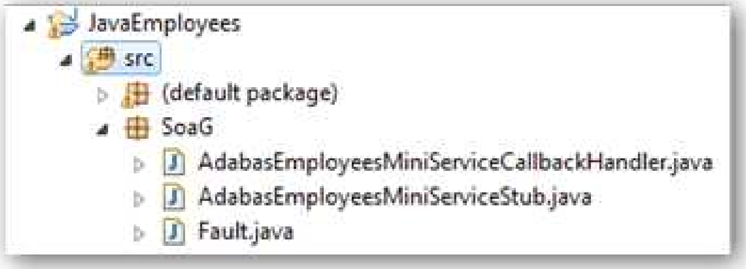
   ```
   wsdl2java -uri
   http://<yourserver>:<yourport>/adabas_EmployeesMini?WSDL -o ..\ -p SoaG
   ```



```
X:\>cd JavaEmployees/src

X:\JavaEmployees\src>wsdl2java -uri http://soagate:56411/adabas_EmployeesMini?WSDL -o ..\ -p SoaG
Using AXIS2_HOME:   D:\tools\Java\axis2-1.5.1
Using JAVA_HOME:    C:\SysTools\Java
Retrieving document at 'http://soagate:56411/adabas_EmployeesMini?WSDL'.
Retrieving schema at 'http://soagate:56411/SOA_GATEWAY_CONFIGURATION_DIRECTORY/schema/public_xsd/xmlmime.xsd',
oagate:56411/adabas_EmployeesMini?WSDL'.
X:\JavaEmployees\src>
```

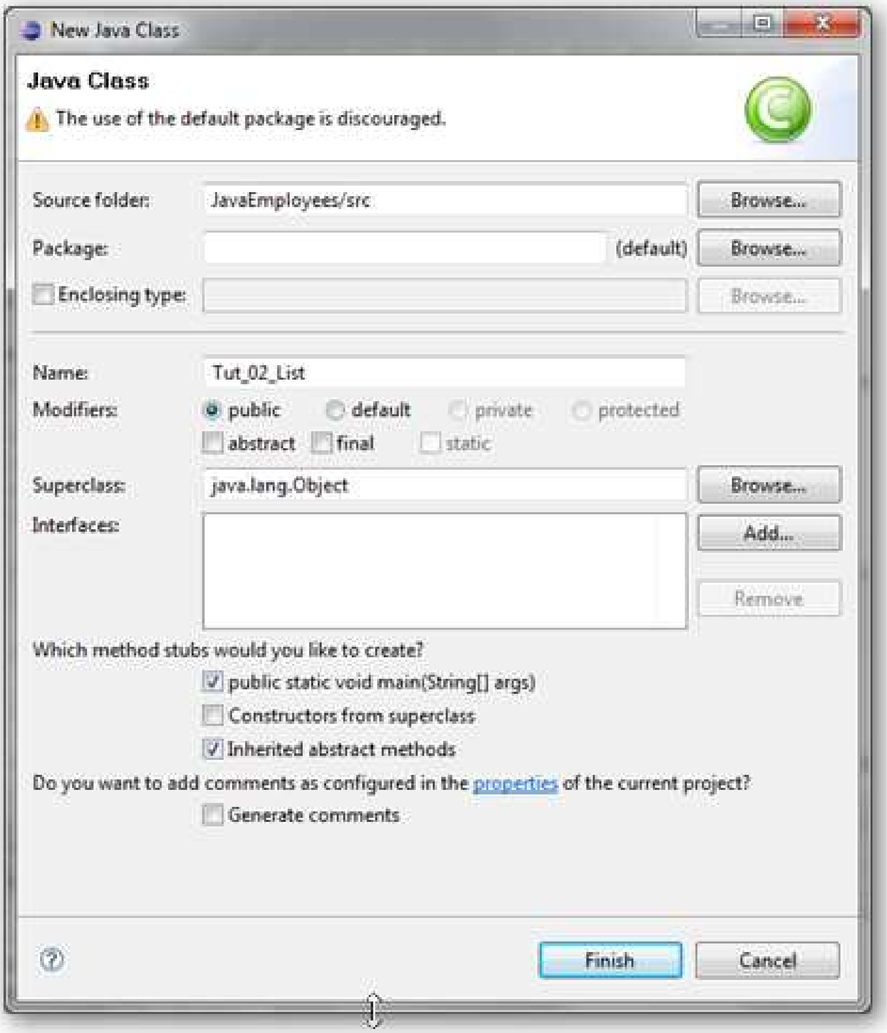
The following items are generated from a SOA Gateway WSDL:

- A "Stub" class implementing all types and operations (ports / bindings)
- A CallbackHandler - a stub class (not used in this tutorial) providing hooks for client-side extensions to the generated result- and error handlers.
- A Fault class



```
JavaEmployees
  src
    (default package)
    SoaG
      AdabasEmployeesMiniServiceCallbackHandler.java
      AdabasEmployeesMiniServiceStub.java
      Fault.java
```

4. Add a new Java class named "Tut_02_List" to the project (**File** -> **New** -> **Class**), opt to create a "main" method, click "**Finish**".



New Java Class

Java Class

The use of the default package is discouraged.

Source folder: JavaEmployees/src

Package: (default)

□ Enclosing type:

Name: Tut_02_List

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

□ Constructors from superclass

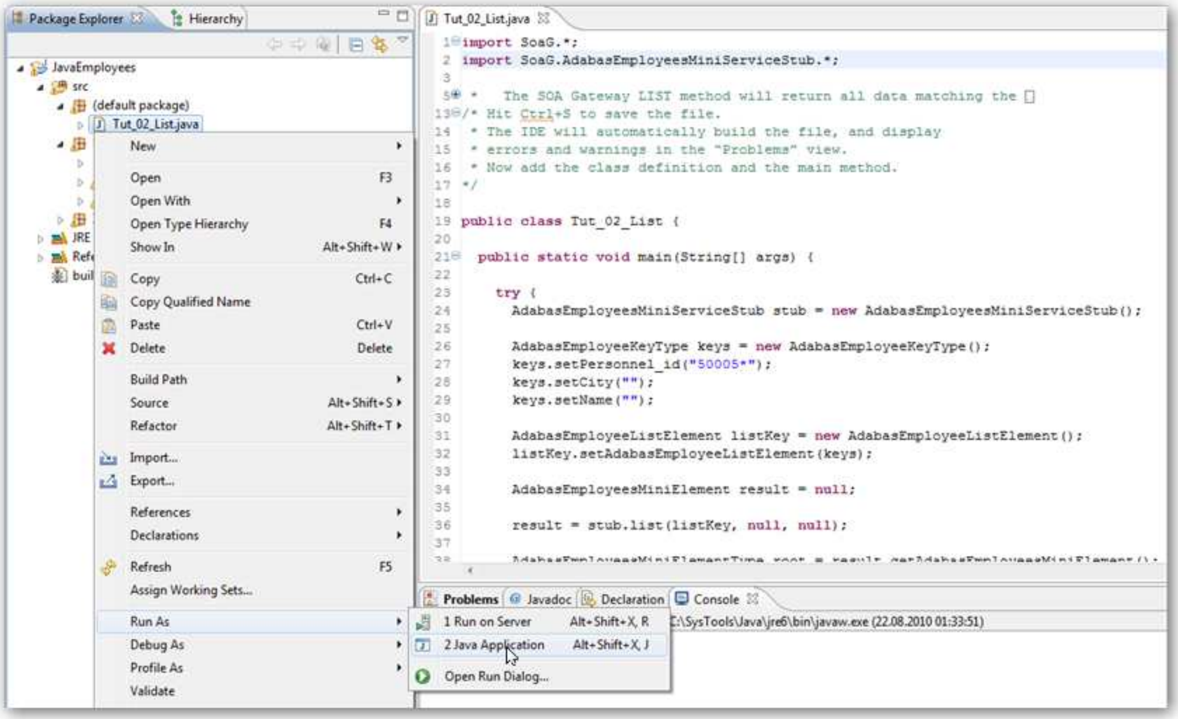
☑ Inherited abstract methods

Do you want to add comments as configured in the properties of the current project?

□ Generate comments

5. Remove the generated code from the newly added class entirely, use (paste) the code from Tut_02_List.java to create your first test program accessing ADABAS data via SOA Gateway.

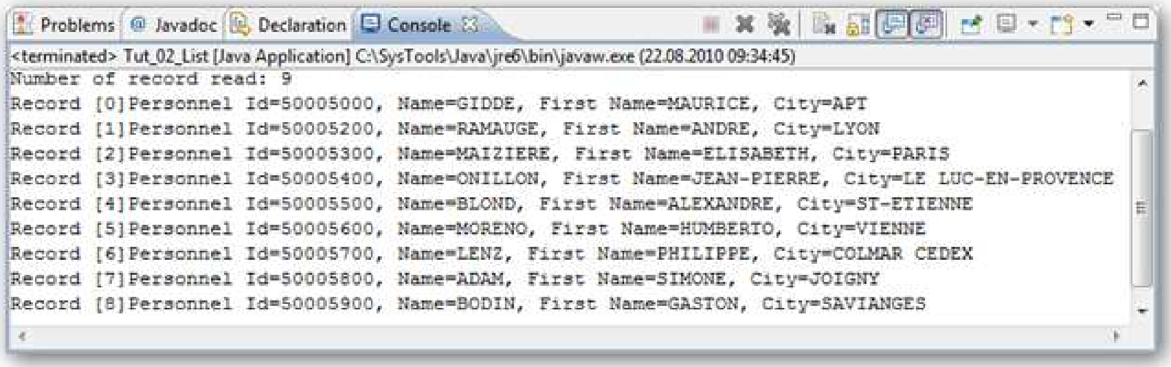
6. Run the program as a "Java Application"



```
import SoaG.*;
import SoaG.AdabasEmployeesMiniServiceStub.*;

 *   The SOA Gateway LIST method will return all data matching the 
/* Hit Ctrl+S to save the file.
 * The IDE will automatically build the file, and display
 * errors and warnings in the "Problems" view.
 * Now add the class definition and the main method.
*/

public class Tut_02_List {

  public static void main(String[] args) {

    try {
      AdabasEmployeesMiniServiceStub stub = new AdabasEmployeesMiniServiceStub();

      AdabasEmployeeKeyType keys = new AdabasEmployeeKeyType();
      keys.setPersonnel_id("50005*");
      keys.setCity("");
      keys.setName("");

      AdabasEmployeeListElement listKey = new AdabasEmployeeListElement();
      listKey.setAdabasEmployeeListElement(keys);

      AdabasEmployeesMiniElement result = null;

      result = stub.list(listKey, null, null);

      AdabasEmployeesMiniElementType root = result.getAdabasEmployeesMiniElement();
```

```
C:\SysTools\Java\jre6\bin\javaw.exe (22.08.2010 01:33:51)
```

7. The output appears in the "Console" window:



```
<terminated> Tut_02_List [Java Application] C:\SysTools\Java\jre6\bin\javaw.exe (22.08.2010 09:34:45)
Number of record read: 9
Record [0]Personnel Id=50005000, Name=GIDDE, First Name=MAURICE, City=APT
Record [1]Personnel Id=50005200, Name=RAMAUGE, First Name=ANDRE, City=LYON
Record [2]Personnel Id=50005300, Name=MAIZIERE, First Name=ELISABETH, City=PARIS
Record [3]Personnel Id=50005400, Name=ONILLON, First Name=JEAN-PIERRE, City=LE LUC-EN-PROVENCE
Record [4]Personnel Id=50005500, Name=BLOND, First Name=ALEXANDRE, City=ST-ETIENNE
Record [5]Personnel Id=50005600, Name=MORENO, First Name=HUMBERTO, City=VIENNE
Record [6]Personnel Id=50005700, Name=LENZ, First Name=PHILIPPE, City=COLMAR CEDEX
Record [7]Personnel Id=50005800, Name=ADAM, First Name=SIMONE, City=JOIGNY
Record [8]Personnel Id=50005900, Name=BODIN, First Name=GASTON, City=SAVIANGES
```

8. This sample selects all "Employees" records with a personnel-id of 50005nnn, you may want to experiment varying the key data, this is easily done by modifying the properties passed to the generated classes. E.g. try the following to list all records for "Employees" whose names start "SMI", living in cities with names starting "D".

```
keys.setPersonnel_id("");

keys.setName("SMI*");

keys.setCity("D*");
```

The following Java/Axis2 tutorial programs are available:

Tutorial	**What it does**
Tut_01a_Get.java	GET a single record by Personnel Id
Tut_01b_GetByISN.java	GET a single record by ISN
Tut_02_List.java	LIST some records
Tut_03a_SelectSimple.java	SELECT by a range of Personnel IDs
Tut_03b_SelectSorted.java	SELECT by a range of Personnel IDs and sort by Name
Tut_03c_SelectConversational.java	SELECT by multiple ranges of Personnel IDs, returned in "chunks" of 20 records each
Tut_04a_AddUpdateDelete.java	ADD + UPDATE + DELETE in "autocommit" mode
Tut_04b_AddTransactional.java	ADD in a transactional context