

Accessing Natural from Java

Contents

1. Introduction	2
2. Prerequisites	2
3. Java.....	2
3.1. Apache Axis2	2
4. Natural	2
5. Discovery.....	2
6. Accessing Web Service with Java	7
6.1. Initial Setup	7
6.2. Building the generated Code	9
7. Writing the Code.....	10
7.1. Simple Java Program	10
7.2. Example using Java Swing	12
7.3. Building the Code	12
7.4. Running the code	12
8. Conclusion.....	13

1. Introduction

In this tutorial we will show you how to build a Java application to access Natural via the SOA Gateway.

2. Prerequisites

As a prerequisite you must have installed Natural and Java.

It is assumed you already have a SOA Gateway server and Control Centre installed. See [here](#) for more info about installing the SOA Gateway.

3. Java

To build and run a Java application, you will need a Java-compatible IDE. There are many available to download, such as NetBeans, JCreator, and IntelliJ. For the purposes of this tutorial, we are going to use Eclipse. The main reason for this is that you are already using Eclipse to run the SOA Gateway control centre, all you need to do is open the Java perspective, by clicking Window, Open Perspective, Java.

3.1. Apache Axis2

Apache Axis2 is an open source web service framework. It can be used to generate Java classes from a WSDL file. We can then use these classes to invoke our SOA Gateway web service.

Download Axis2 from [here](#). You should choose the “Standard Binary Distribution”. Save this file to a well known location, and extract. For example save to C:\Axis2\ and extract in this folder.

4. Natural

Natural is a programming language designed to simplify the implementation of business solutions. It takes a very pragmatic and non-theoretical approach to common programming tasks such as database access.

Natural is similar to JAVA in that it is a semi-interpreted language; Natural source programs are “compiled” into a platform independent byte code (object) format which is then executed (interpreted) by a platform dependent runtime environment. While originally developed for mainframe computers, Natural has been available on Windows and UNIX platforms for several years now.

This tutorial is based on the Natural Community Edition which can be downloaded from [Natural download site](#).

When Natural is installed successfully load the objects used for this tutorial. Details on this can be found [here](#). The sample objects are suitable for a Windows/Unix/Linux environment.

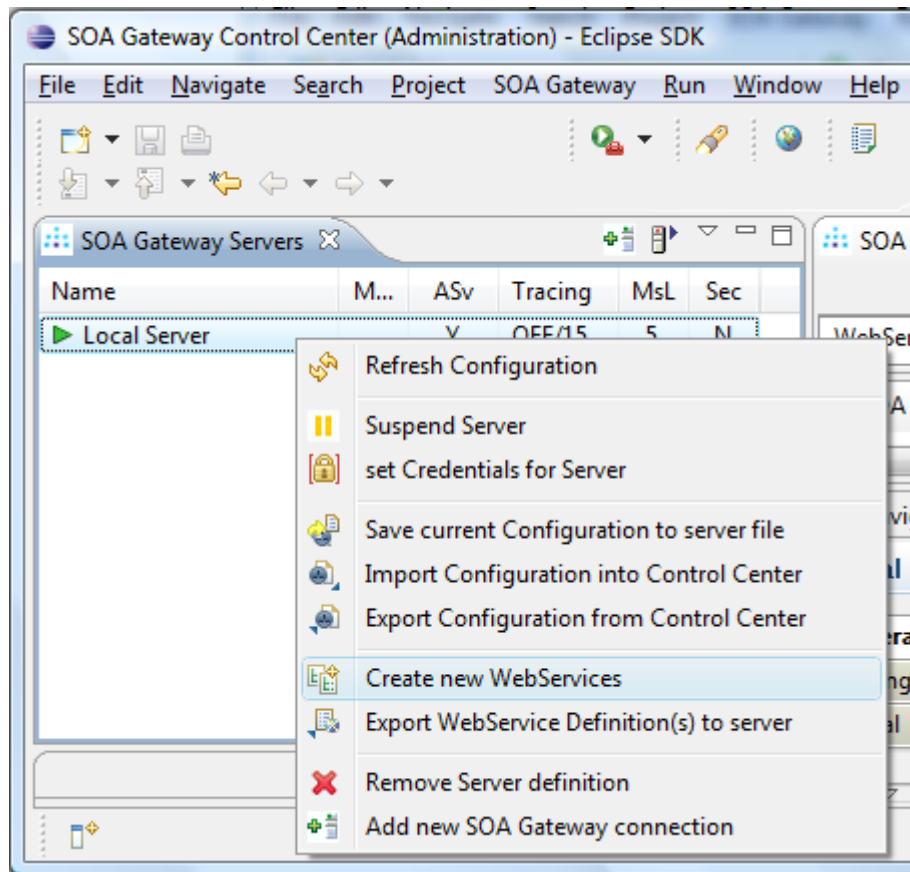
5. Discovery

In this section we’ll show you how to create a web services for one of the programs. This web service can be used by InfoPath (and many others) to give you direct real-time access to your Natural program.

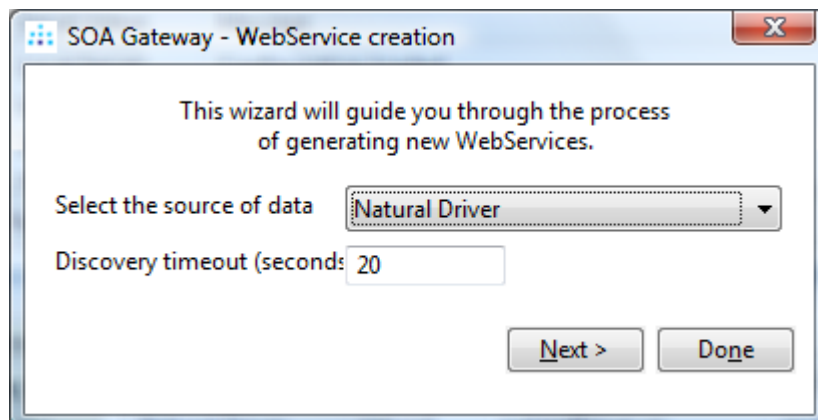
Web Service Creation using SOA Gateway

Start your SOA Gateway Control Centre. See [here](#) for an introduction to the Control Centre.

In your servers view, right click the entry which represents your local SOA Gateway Server. Select “Create new WebServices”.

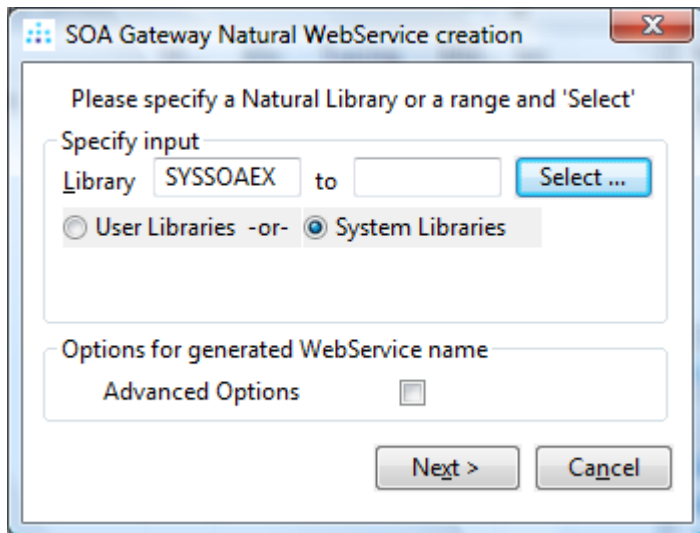


From the next dialog, choose “Natural Driver”. If you do not see have a Natural Driver in the list, see how to create one [here](#).

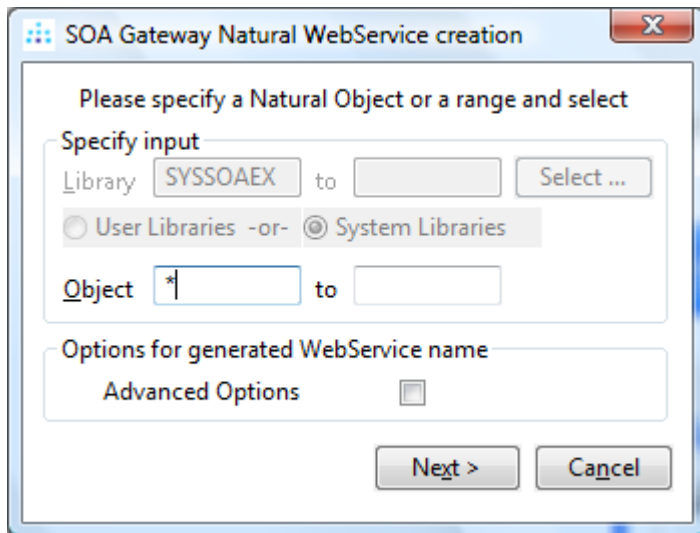


Click Next.

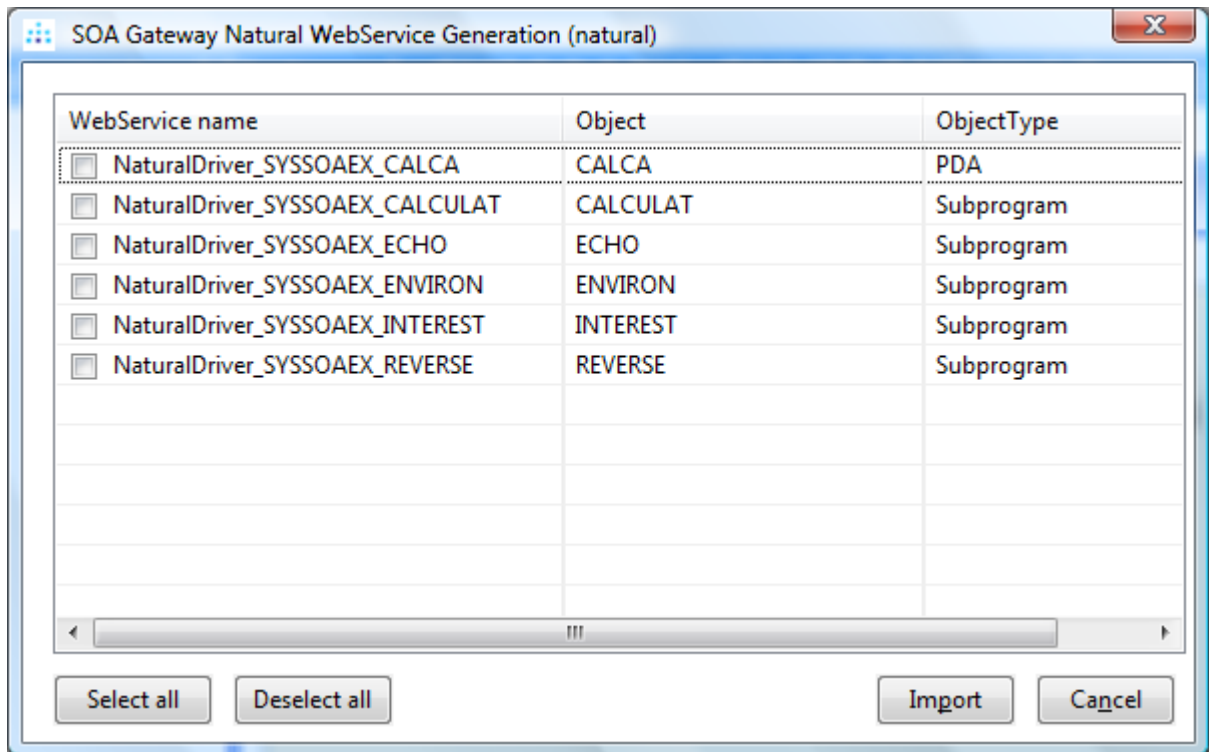
Enter “SYSSOAEX” for the Library name, select “System Libraries”, click “Next >”



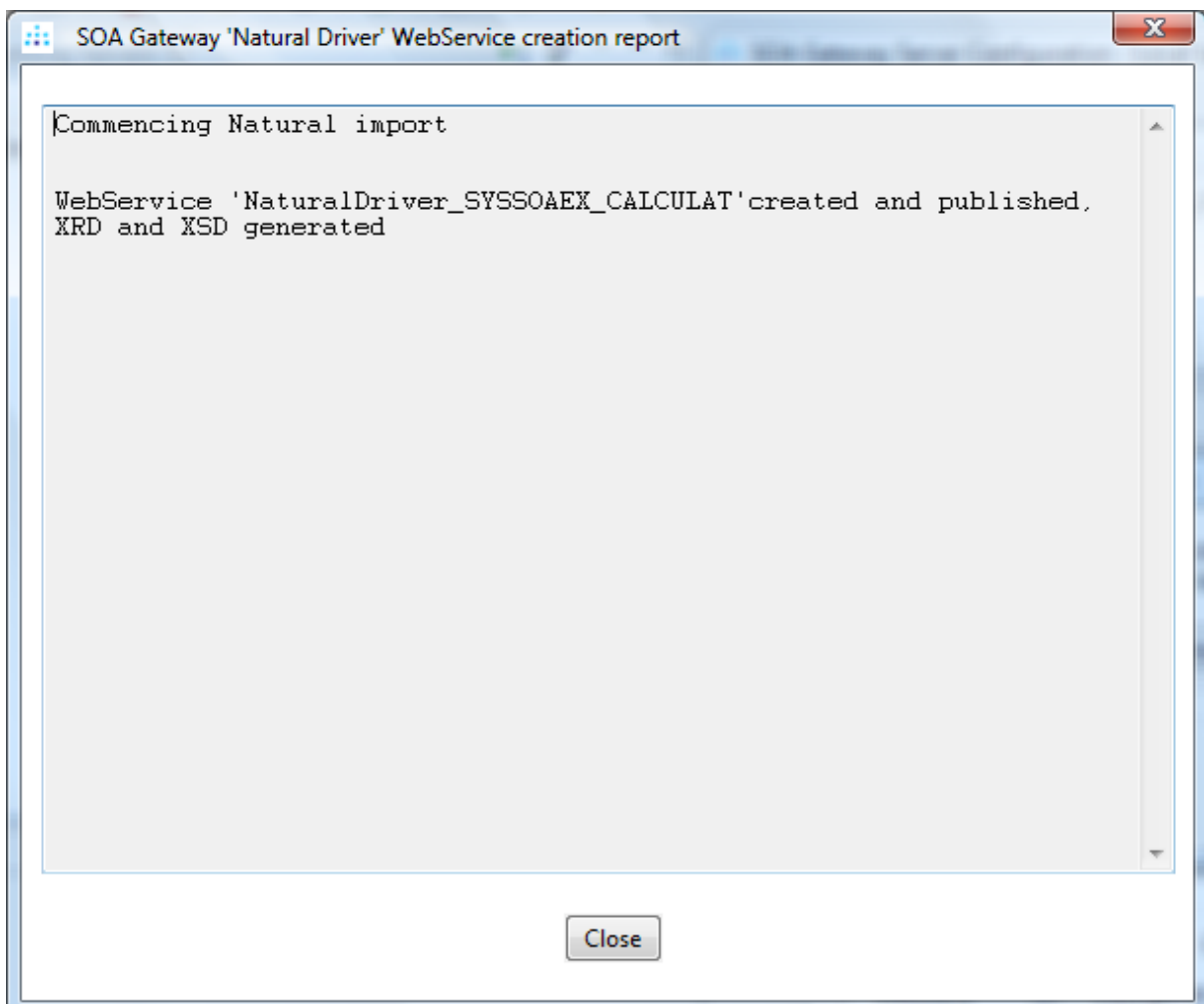
Put an asterisk (*) in the “Object” field and click “Next >”



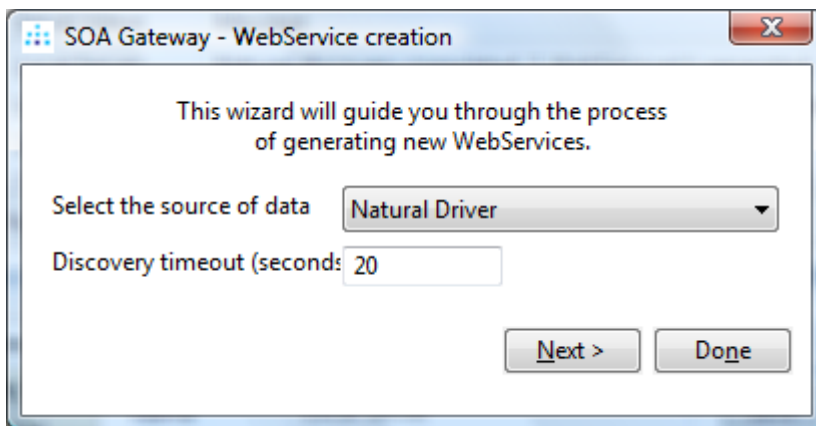
A list of possible WebServices is presented:



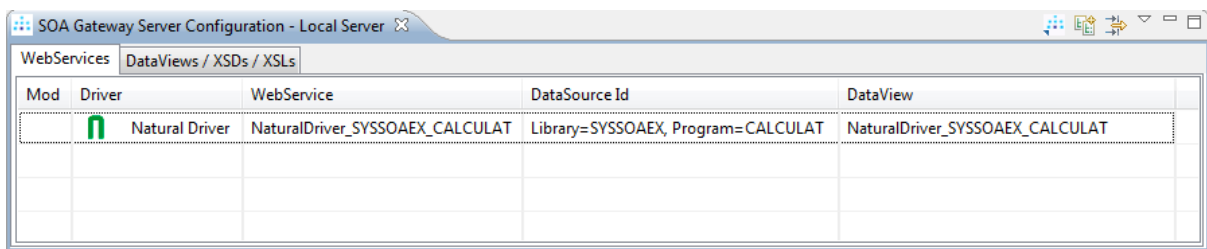
Select the “Natural_SYSSOAEX_CALCULAT” WebService and click “Import”.



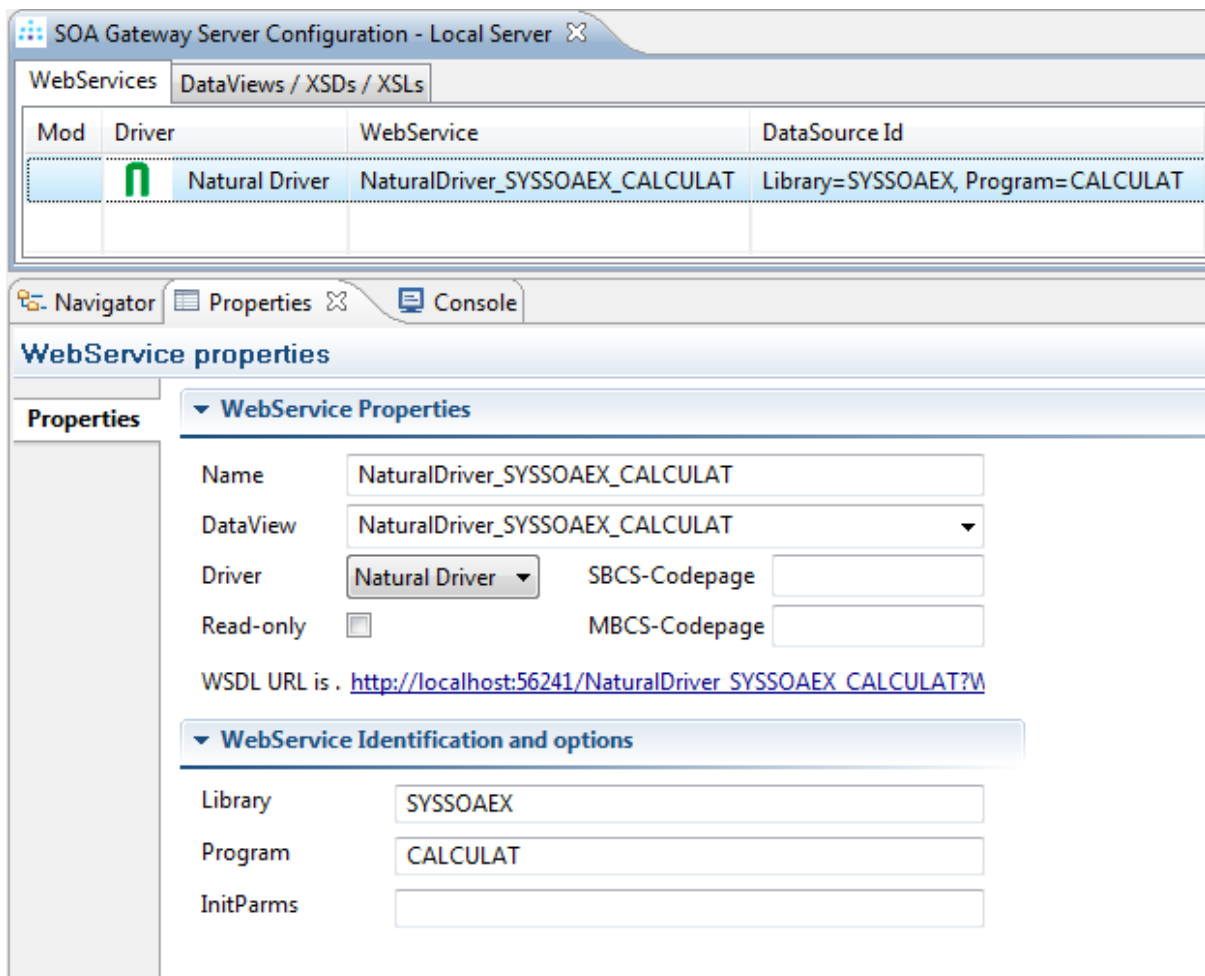
Select Close.



Select Done.



Click the WebService in the Configuration View. The Properties View displays the details.



Web Service Description Language (WSDL) is a standard, XML-based language that is used to describe a Web Service. The WSDL itself is usually interpreted by a web service client, such as InfoPath.

As WSDL is XML-based, it will open in your browser of choice. To see the WSDL click on the link at 'WSDL URL is' in the WebService Properties window.

This WSDL is the starting point for using Web Services, and can be used time and again by different web service clients.

6. Accessing Web Service with Java

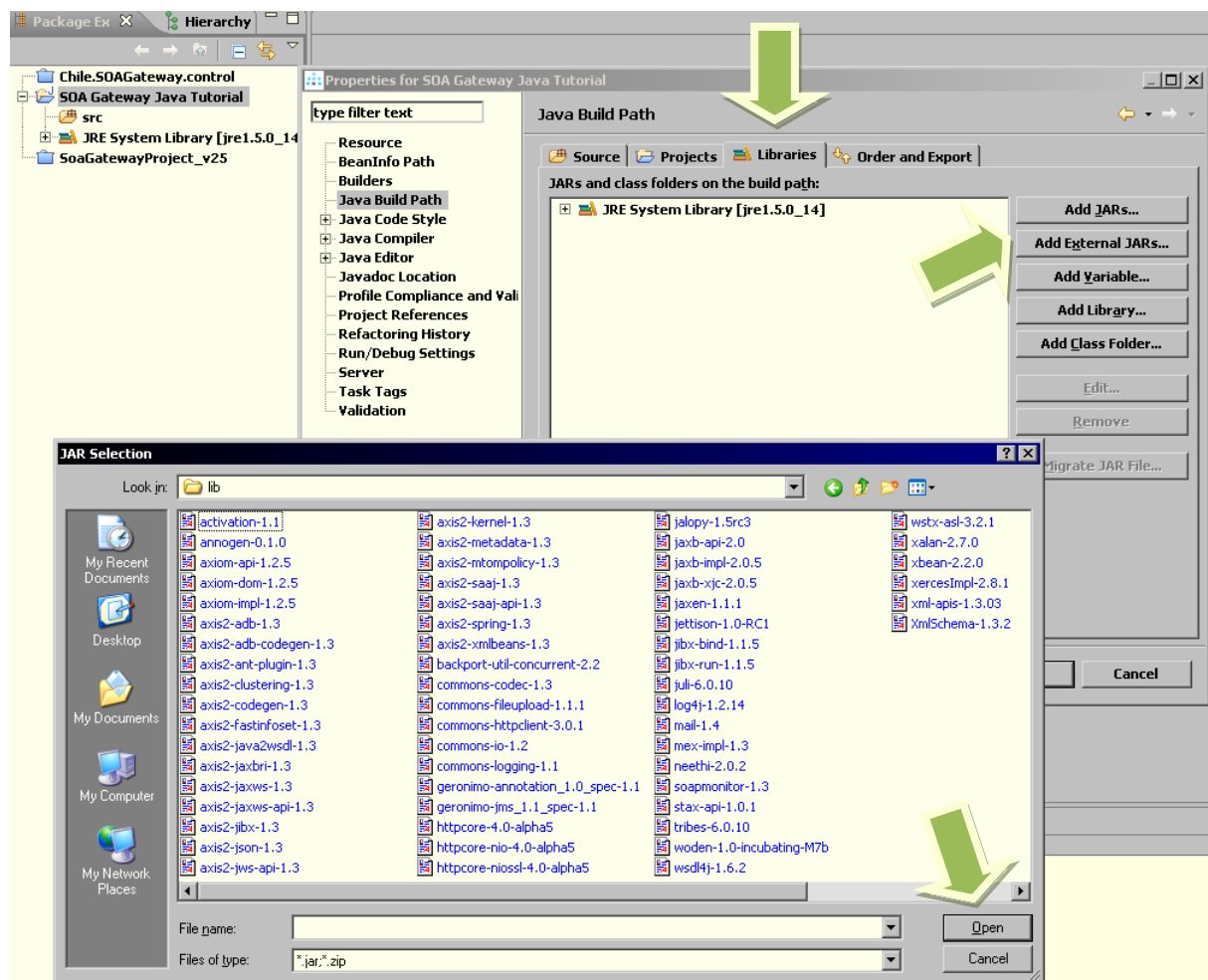
Java is an object-orientated programming language developed by Sun Microsystems. Its syntax is typically based around C++, but has fewer low-level APIs. Java programs are usually compiled into byte-code which can be run on any machines which run a Java Virtual Machine.

6.1. Initial Setup

In your Eclipse IDE, open a Java perspective. Select Window, Open Perspective, Java.

Create a new java project, by selecting File, New, Java Project. Name the project "SOA Gateway Java Tutorial". Click Finish.

Click File, Properties to view your project properties. Select “Java Build Path”, the “Libraries” tab, and click “Add External Jars”. From the pop-up that appears, traverse the “lib” folder in the Axis2 distribution you downloaded earlier. Select **all** these jar files, and click “Open”.



Click OK.

Open a DOS box, and change to the Axis2 bin directory. For example

```
C:\Users\John> cd \
```

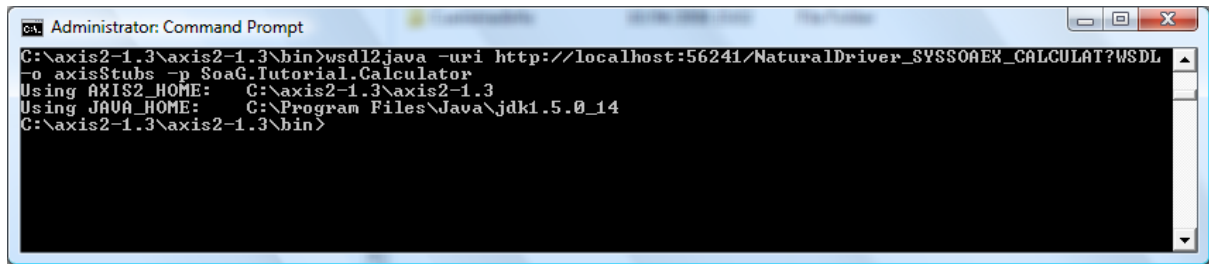
```
C:\>cd axis2-1.3\axis2-1.3\bin
```

We want to use 1 of the Web Services we’ve created which is the calculator web service.

Using the “wsdl2Java” program, Axis2 will generate Java code from the WSDL which we can use to call the SOA Gateway web service.

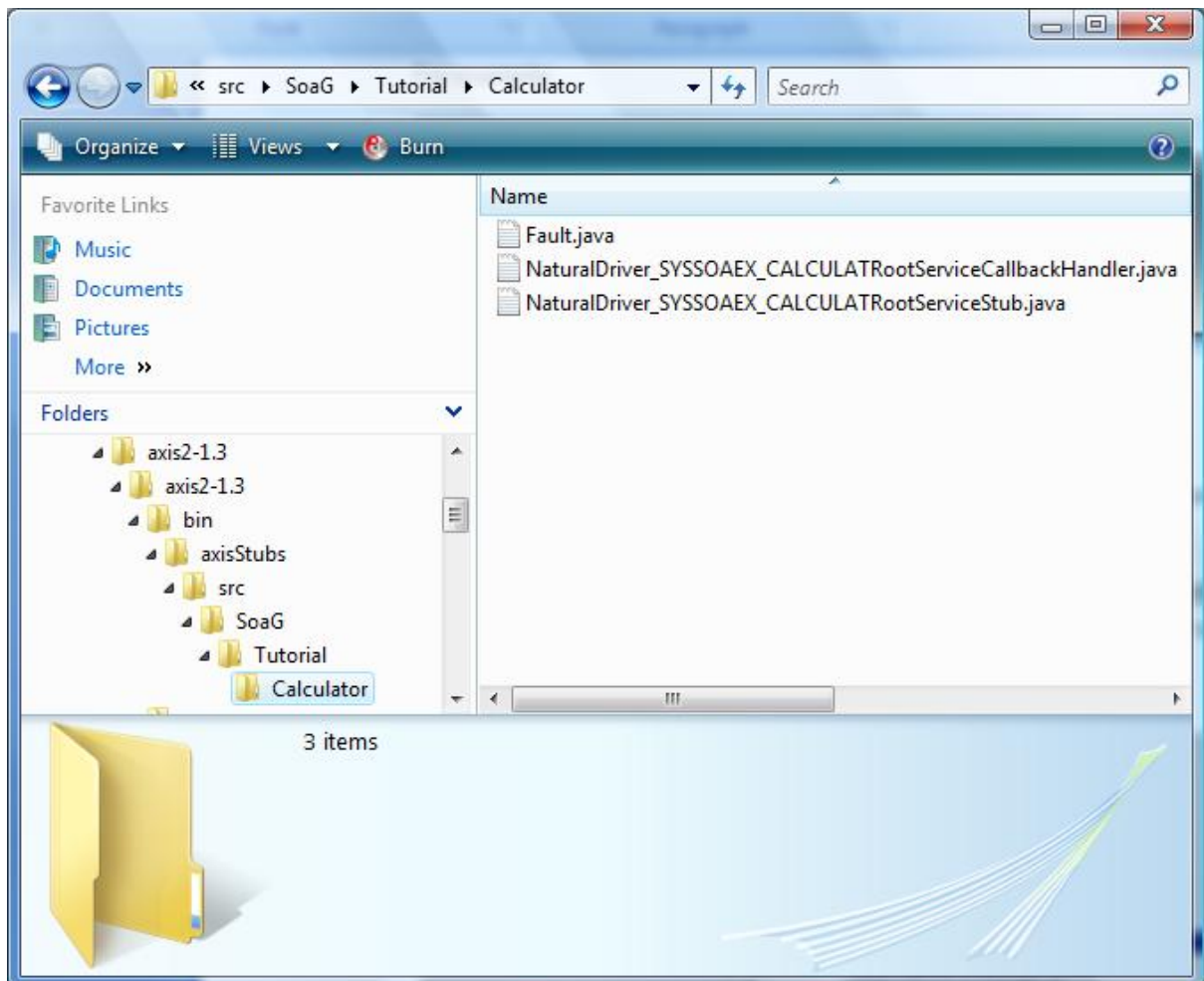
Run the following command:

`wsd12java -uri http://localhost:56241/NaturalDriver_SYSSOAEX_CALCULAT?WSDL -o axisStubs -p SoaG.Tutorial.Calculator:`



```
Administrator: Command Prompt
C:\axis2-1.3\axis2-1.3\bin>wsdl2java -uri http://localhost:56241/NaturalDriver_SYSSOAEX_CALCULAT?WSDL
-o axisStubs -p SoaG.Tutorial.Calculator
Using AXIS2_HOME: C:\axis2-1.3\axis2-1.3
Using JAVA_HOME: C:\Program Files\Java\jdk1.5.0_14
C:\axis2-1.3\axis2-1.3\bin>
```

This is what wsd12java generated:

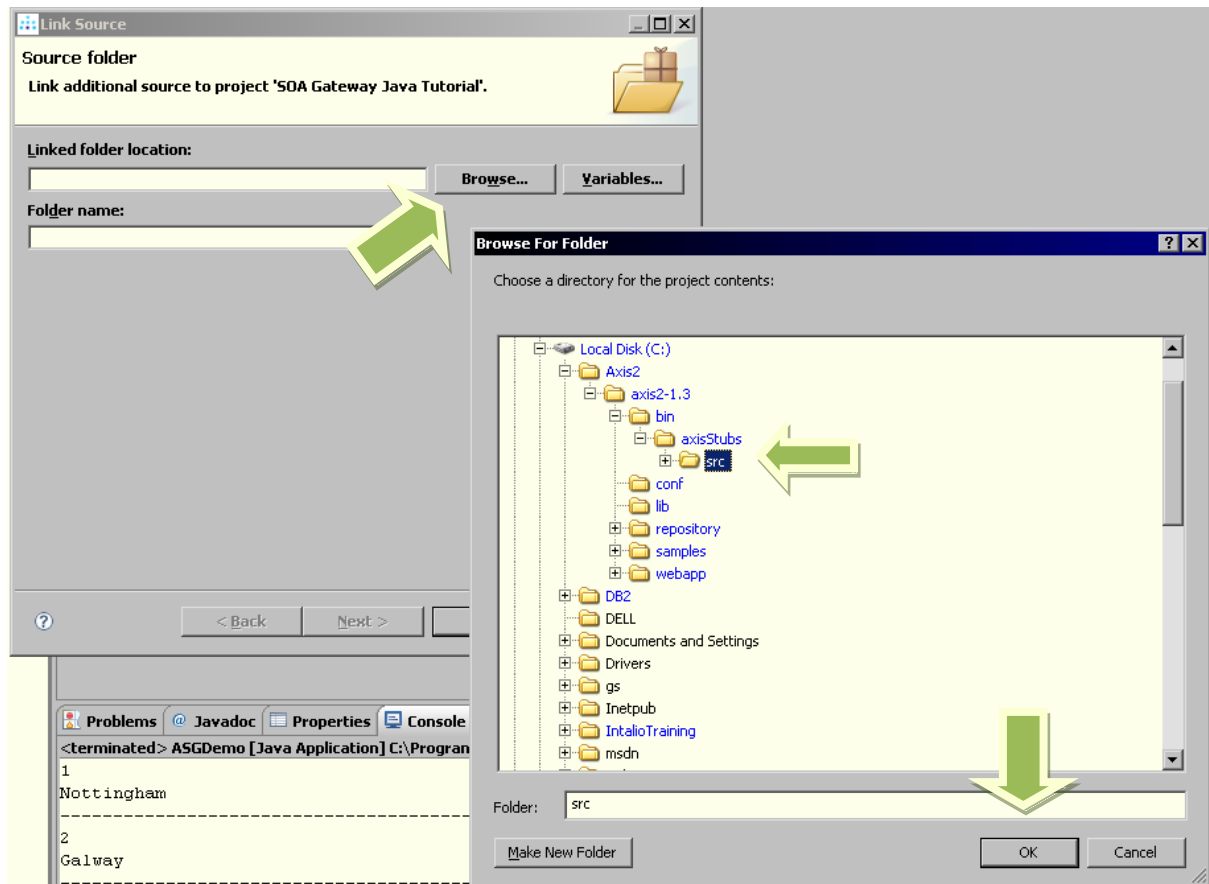


6.2. Building the generated Code

We will use the Eclipse IDE to build the generated code for us. To do this we need to add the “axisStubs” folder to our “SOA Gateway Java Tutorial” Project.

To do this, right click the “SOA Gateway Java Tutorial” project. Select “Build Path” and “Link Source”. Browse to the location of the “axisStubs”. Select the “src” folder and click OK.

For example:



After clicking OK, in the “Folder Name” field, enter AxisStubs. Click Finish.

7. Writing the Code

7.1. Simple Java Program

In this section we’ll create a simple java program that calls the SOA Gateway web service generated above. Under the SOA Gateway Java Tutorial project, right click the “src” folder. Select “New”, “File”, and name your file “simpleCalc.java”. Copy the source code from [here](#).

This is the code:

```
import SoaG.Tutorial.Calculator.*;
import
SoaG.Tutorial.Calculator.NaturalDriver_SYSSOAEX_CALCULATRootServiceStub.*;

public class simpleCalc {
    public static void main(String[] args) {
        try {
            NaturalDriver_SYSSOAEX_CALCULATRootServiceStub stub = new
NaturalDriver_SYSSOAEX_CALCULATRootServiceStub();
            NaturalDriver_SYSSOAEX_CALCULATRoot_type1 root = new
NaturalDriver_SYSSOAEX_CALCULATRoot_type1();
            NaturalDriver_SYSSOAEX_CALCULATGroup_type1 group = new
NaturalDriver_SYSSOAEX_CALCULATGroup_type1();
            InvokeInputElement input = new InvokeInputElement();
```

```

group.setOPERAND_1(556);
group.setOPERAND_2(4);
group.setOPERATION("DIV");
root.setNaturalDriver_SYSSOAEX_CALCULATGroup(group);
input.setNaturalDriver_SYSSOAEX_CALCULATRoot(root);

InvokeOutputElement output = null;
output = stub.soag_invoke(input, null, null);

System.out.println(output.getNaturalDriver_SYSSOAEX_CALCULATRoot().getNaturalDriver_SYSSOAEX_CALCULATGroup().getFUNCTION_RESULT());
}
catch (Exception e){
e.printStackTrace();
}
}
}

```

As you can see we are providing the following parameters to the Natural Calculator web service:

```

group.setOPERAND_1(556);
group.setOPERAND_2(4);
group.setOPERATION("DIV");

```

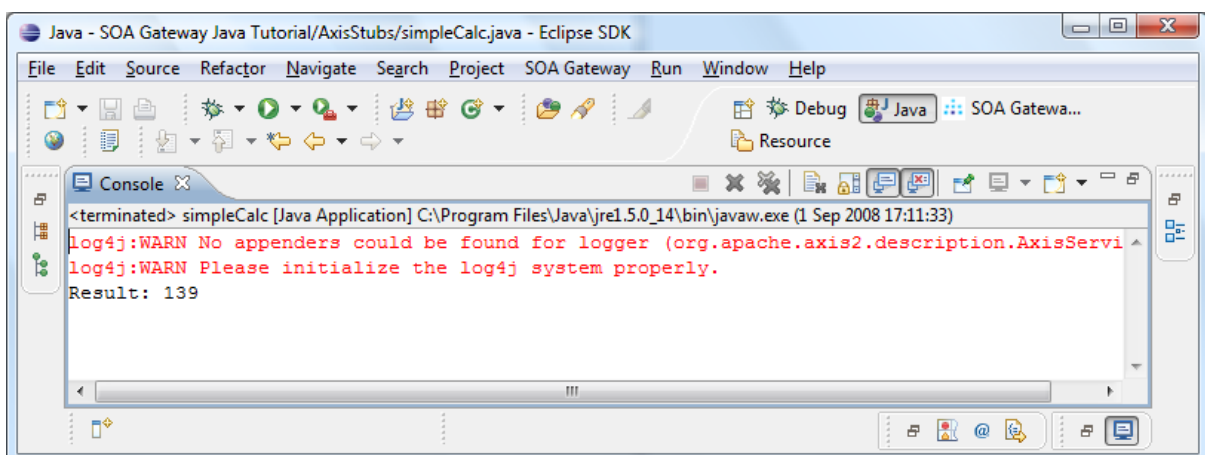
i.e. divide 556 by 4.

Hit Ctrl+S to save the file. The IDE will automatically build the file, and display errors and warnings in the “Problems” view.

The final thing to do is to run your program. Again this can be done from within the Eclipse IDE. Firstly, right-click the “simpleCalc.java” file. Then select “Run As”, “Java Application”.

The results of the program will appear in the “Console” view.

E.g. Result is 139.



7.2.Example using Java Swing

In this section we'll create a more complex example which makes use of Java's Graphical components, called "swing".

Right-click the "src" folder in your SOA Gateway Java Tutorial project. Select, "New", "File", and enter "guiCalc.java". Click Finish.

The complete code can be copied from [here](#).

Here all of the operations available to the web service can be called.

7.3.Building the Code

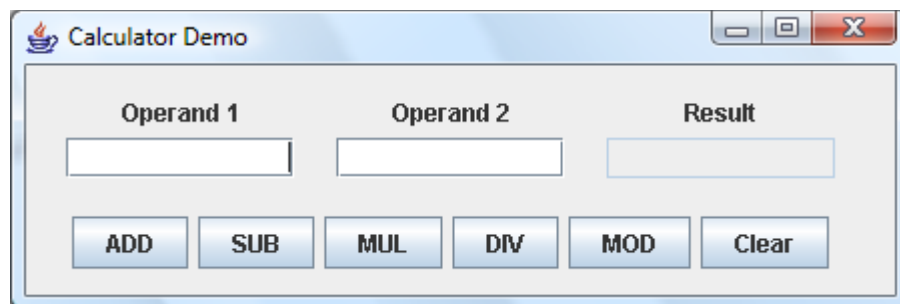
Hopefully, all is well, but in the case of errors or if you wish to change the code, try the following:

- Obviously misspellings are often the cause of compilation errors. Ensure that all object names, and variable names are spelt correctly.
- The Eclipse IDE has a neat trick where you can type the first few letters of an object, and by hitting Control + Space, it will bring up the suggested object names. It may give you an indication of a misspelt object name.
- Similarly, if you need the name of a member of an object, type the object name, followed by a dot (".") and Control+Space. The list of available proposals should appear.

7.4.Running the code

To run your program, right-click "guiCalc.java" and select "Run As", and "Java Application".

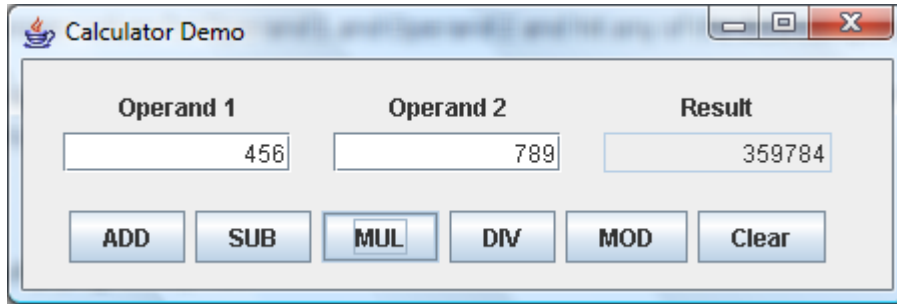
The program will appear on the screen:



Enter proper values for Operand 1 and Operand 2 and hit any of the desired operations.

The program will access the SOA Gateway web service to calculate the result from the Natural calculator program.

E.g. Multiplying 456 by 789:



8. Conclusion

This tutorial shows how to access Natural from Java using the SOA Gateway. As you can see, you have built a powerful application that uses a Web Service to retrieve information in real-time.