

Consumo de servicios Web con Apache Axis2 desde un cliente Java

En este ejercicio se creará un cliente de escritorio en Java usando la plataforma de desarrollo Eclipse que consuma el Servicio Web desarrollado con Apache Axis 2.

Pasos a seguir

El entorno Eclipse con el plugin WTP, al igual que para la creación de Servicios Web, ofrece un entorno que permite crear clientes que consuman dichos servicios de manera sencilla. El único problema es que los clientes de momento, solamente pueden ser creados en un proyecto Web dinámico, con lo que se incluyen elementos innecesarios en el proyecto cuando se quiere una aplicación cliente de escritorio. Este método de creación es descrito en el siguiente enlace.

<http://www.eclipse.org/webtools/community/tutorials/BottomUpAxis2WebService/tutorial.html>

Para ejecutar un cliente generado con el procedimiento anterior desde una consola (de forma externa al entorno de Eclipse) ejecutar:

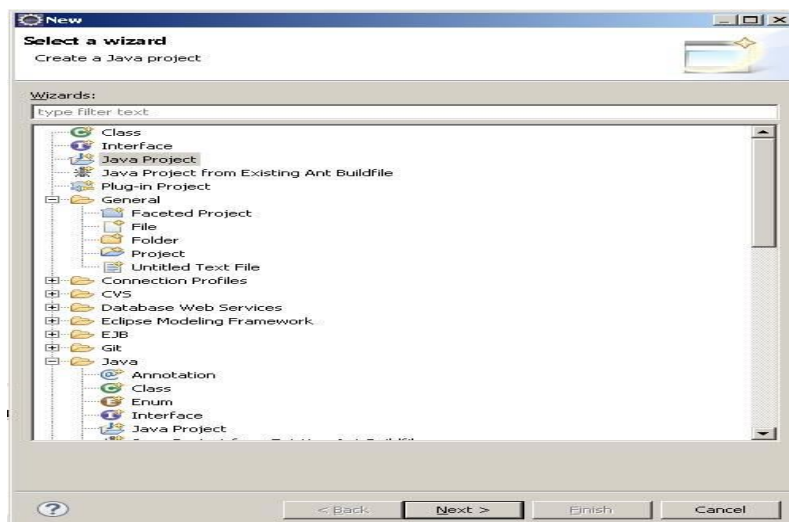
```
C:> java -Djava.ext.dirs=\axis2_home\lib clase_cliente
```

De esta forma incluirá todas las librerías necesarias de AXIS2 para ejecutar el cliente sin tenerlas que incluir en el CLASSPATH.

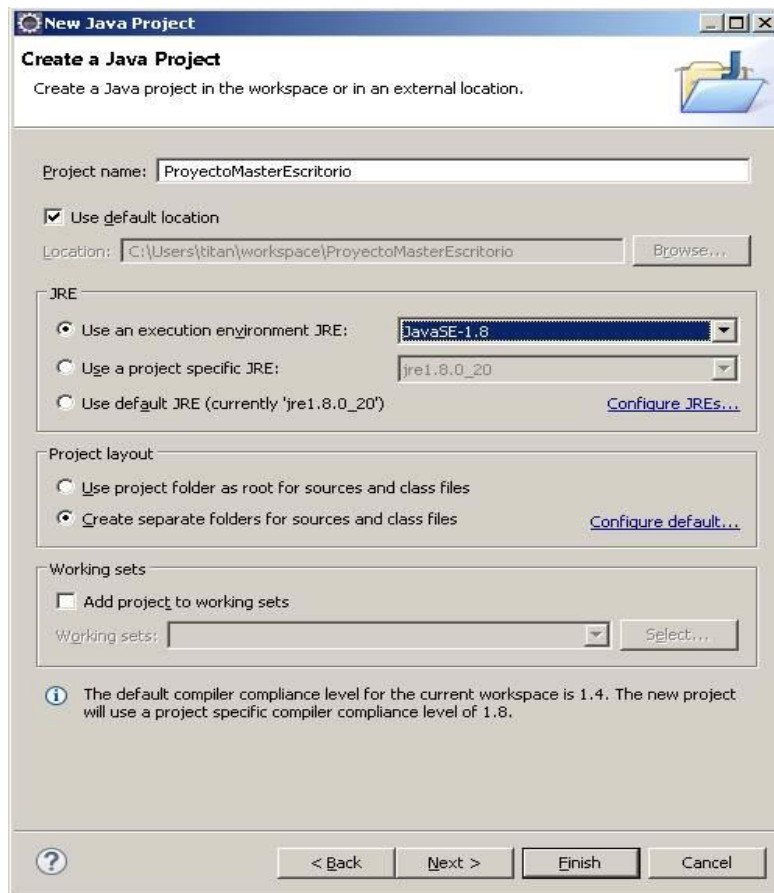
En el ejercicio propuesto se muestra una forma de poder crear aplicaciones de escritorio sobre un proyecto Java que consuman los Servicios Web desde Eclipse. Para ello se usará la herramienta de Apache AXIS 2 *WSDL2Java*.

1. Creación de un Proyecto Java

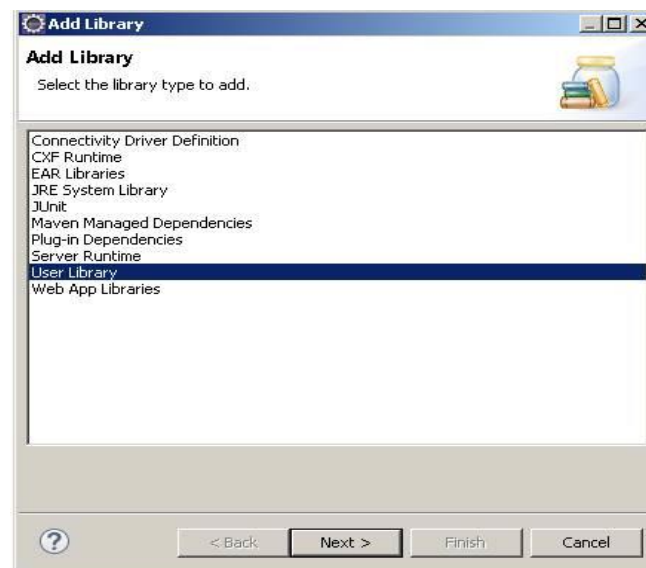
- 1.1. Ejecutamos desde el menú *File* → *New* → *Other...*
- 1.2. En la ventana que aparece a continuación, seleccionamos la carpeta *Java* → *Java Project* y pulsamos *Next*



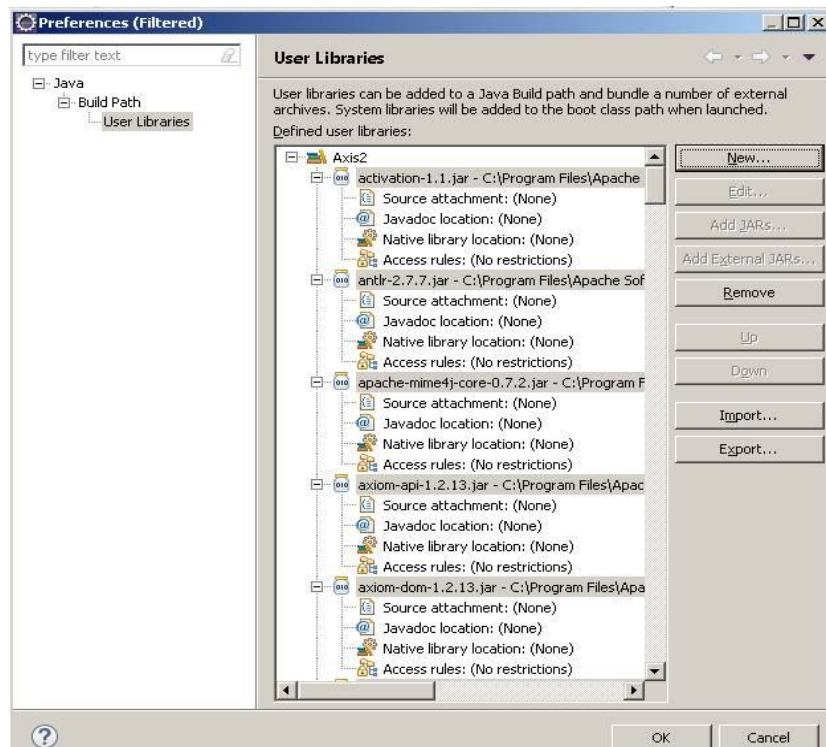
- 1.3. A continuación, introducimos el nombre del proyecto y aceptamos las opciones que aparecen por defecto y pulsamos *Next*.



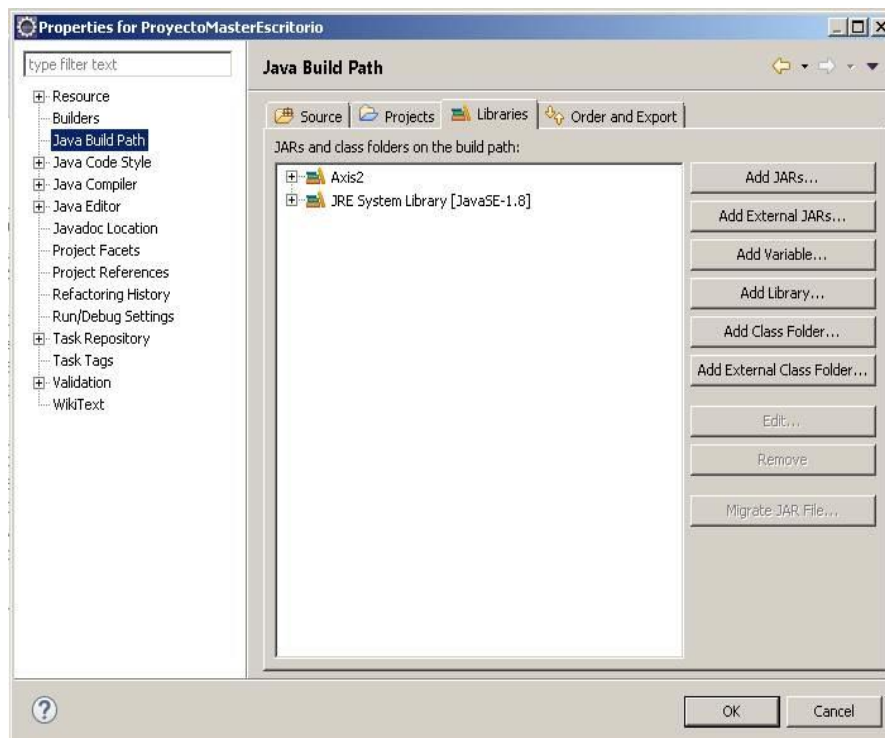
- 1.4. A continuación debemos añadir las librerías de AXIS 2. Para ello seleccionamos la pestaña *Libraries* y pulsamos en *Add Library* y seleccionamos *User Library* y pulsamos en *Next*.



- 1.5. Creamos una nueva librería *New...* Para ello clicamos sobre User Libraries → *New...* que llamaremos Axis2, la seleccionamos y añadimos los archivos *.jar ubicados en *RUTA_AXIS2/lib* con la opción *Add Jars...* Posteriormente pulsamos *Ok*.




- 1.6. Pulsamos *Finish* en la ventana anterior y ya hemos añadido las librerías necesarias de Apache AXIS 2.



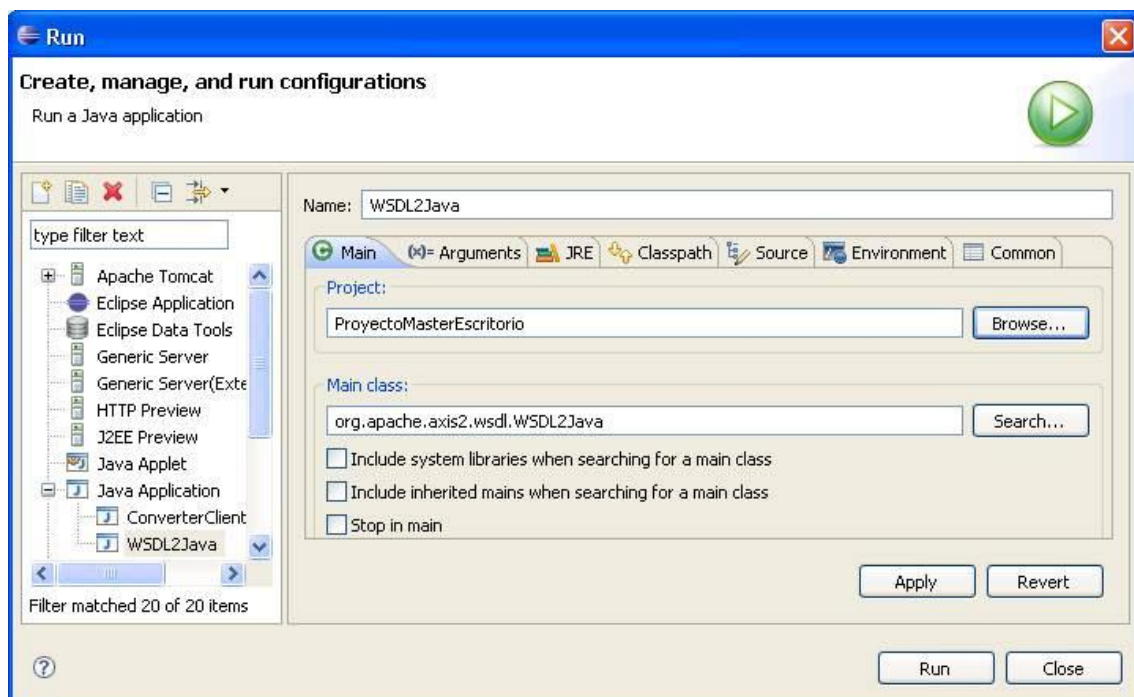
2. Integración de WSDL2Java en Eclipse

2.1. En el menú pulsamos *Run* → *Run configuration*

2.2. En la ventana que aparece a continuación seleccionamos en el panel de la izquierda el item que indica *Java Application* y pulsamos en el icono situado en la parte superior para crear una nueva configuración. 

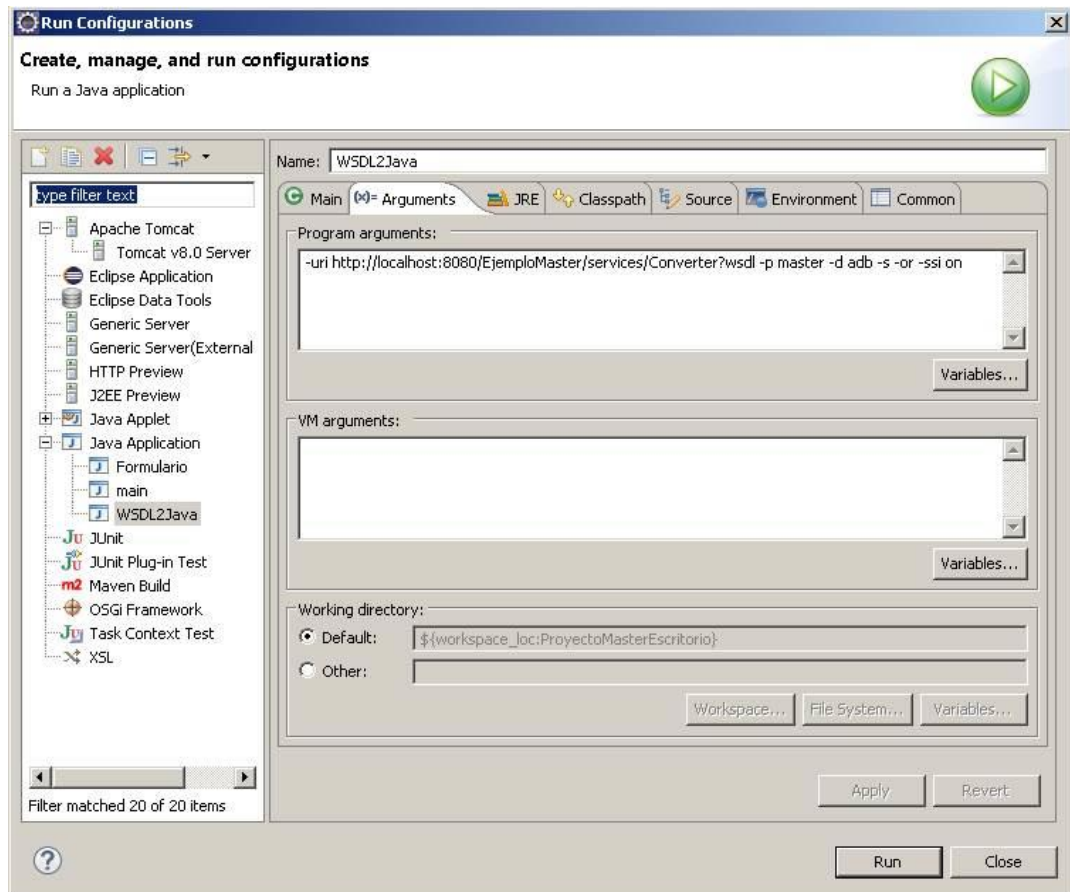
2.3. Introducimos el nombre de la configuración *WSDL2Java*, asociamos el proyecto *Java* y como clase principal indicamos *org.apache.axis2.wsdl.WSDL2Java*

Nota: Recordad que al Servicio Web le hemos llamado EjemploMaster en lugar de ProyectoMaster.



2.4. En la misma ventana pulsamos en la pestaña *(x)=Arguments* y en el apartado *Program arguments* introducimos la siguiente línea

```
-uri http://localhost:8080/ProyectoMaster/services/Converter?wsdl -p master -d  
adb -s -or -ssi on
```



Si os aparece un exception de Method Not Found, repasar bien la URL introducida, ya que ese error ocurre cuando el método no está alojado en services.

2.5. Pulsamos en *Apply* y posteriormente en *Run*

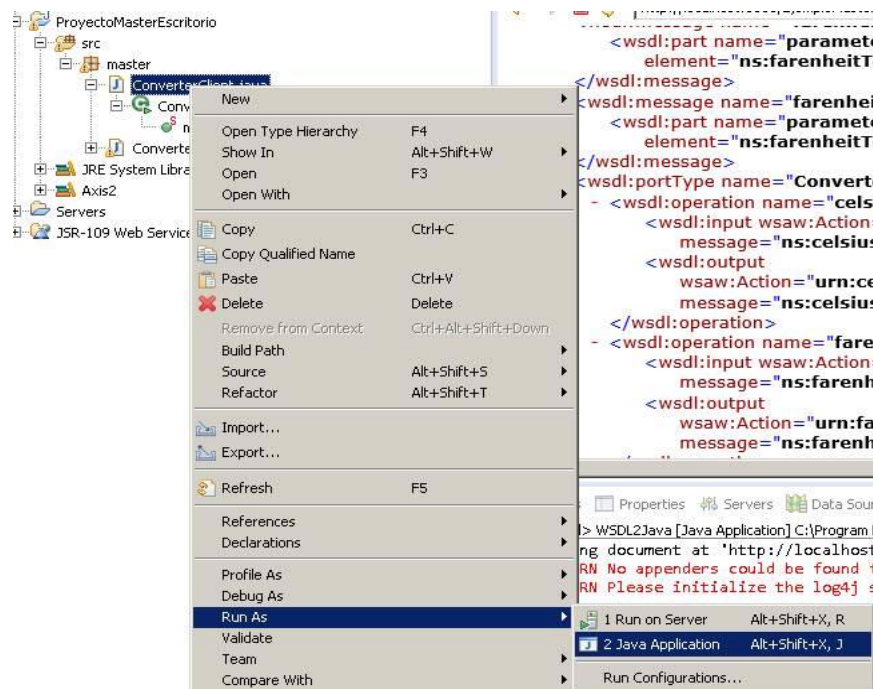
2.6. Sobre el proyecto pulsando *F5* veremos que se ha generado la clase proxy para acceder al Servicio Web indicado en *Programs arguments*.



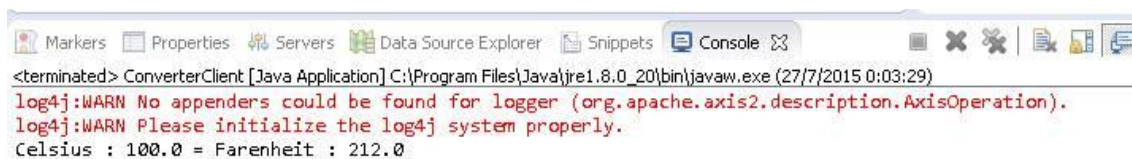
2.7. Ahora podemos importar en el proyecto la clase cliente *ConverterClient.java* que utiliza la clase proxy para acceder al servicio. El file *ConvertClient.java* está en la carpeta cliente de los recursos que se os han facilitado.



2.8. Seleccionamos la clase *ConverterClient.java* en el proyecto y pulsando el botón derecho seleccionamos *Run As → Java Application*.



2.9. En la consola inferior de Eclipse podemos comprobar la ejecución.



NOTA: El aviso que aparece se debe a que Apache Axis 2 utiliza log4j como sistema de log y en este caso no encuentra el archivo de configuración. Para resolverlo podemos copiar de *AXIS2_HOME/conf* el archivo *log4j.properties* a la carpeta *Bin* del proyecto cliente.

