Code First Approach for Creating SOAP Web Service

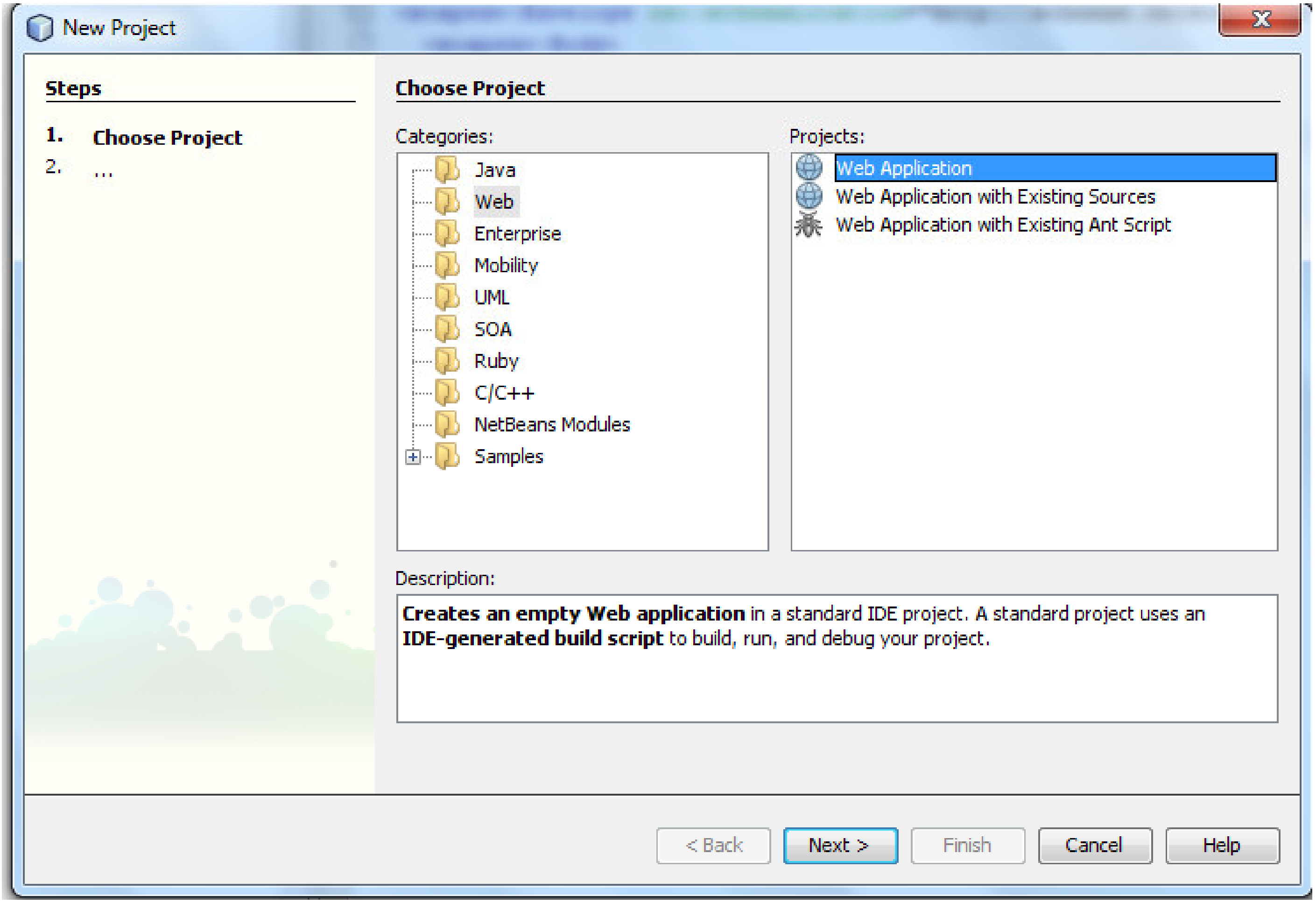
Prof. (Dr.) Vipul K. Dabhi
Assoc. Professor,
Department of Information Technology,
D. D. University

Approaches for creating SOAP web-service

- Code First Approach
  - Writing source code for web service and then WSDL File
- Contract First Approach
  - Creating Web-service based on given WSDL File

Creating Web Service using Code First Approach

- **Create Web Application**
  - **Give it name Arithmetic**



New Project

Steps

1. Choose Project
2. ...

Choose Project

Categories:

Projects:

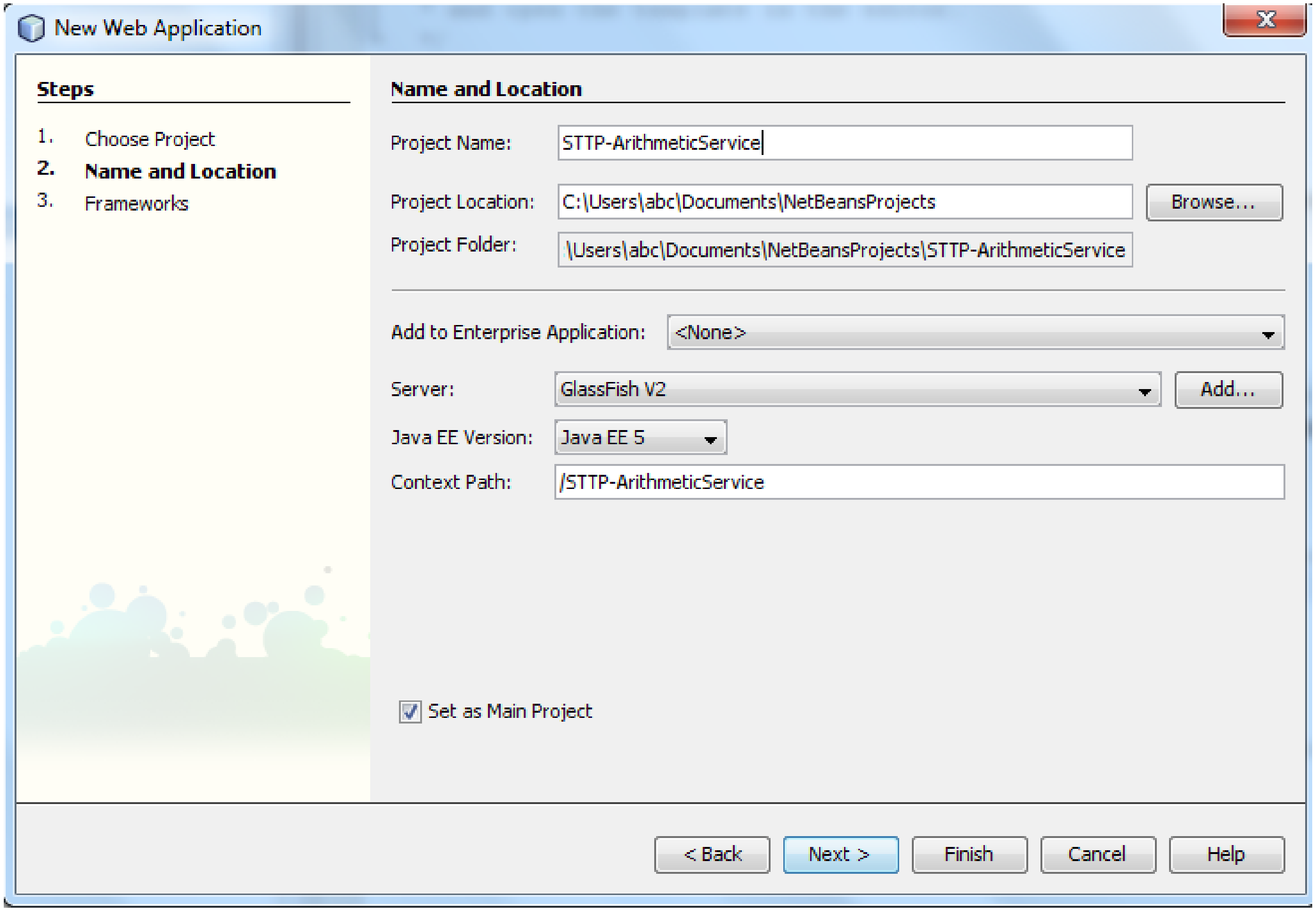
Web Application
Web Application with Existing Sources
Web Application with Existing Ant Script

Description:

Creates an empty Web application in a standard IDE project. A standard project uses an **IDE-generated build script** to build, run, and debug your project.

Creating Web Service using Code First Approach

- **Create Web Application**
- **Give it name STTP-ArithmeticService**



Project Name: STTP-ArithmeticService

Project Location: C:\Users\abc\Documents\NetBeansProjects

Project Folder: \Users\abc\Documents\NetBeansProjects\STTP-ArithmeticService

Add to Enterprise Application: <None>

Server: GlassFish V2

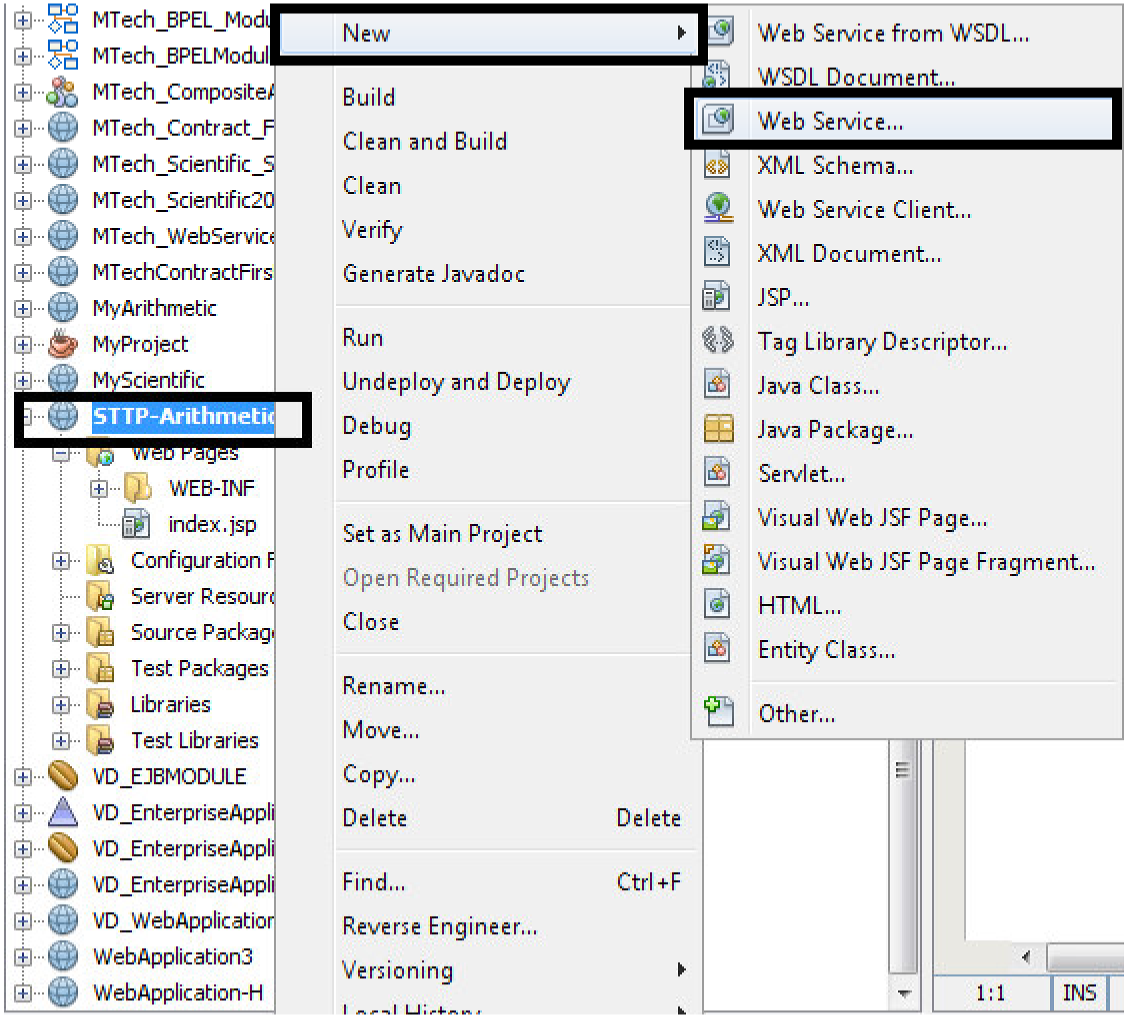
Java EE Version: Java EE 5

Context Path: /STTP-ArithmeticService

☑ Set as Main Project

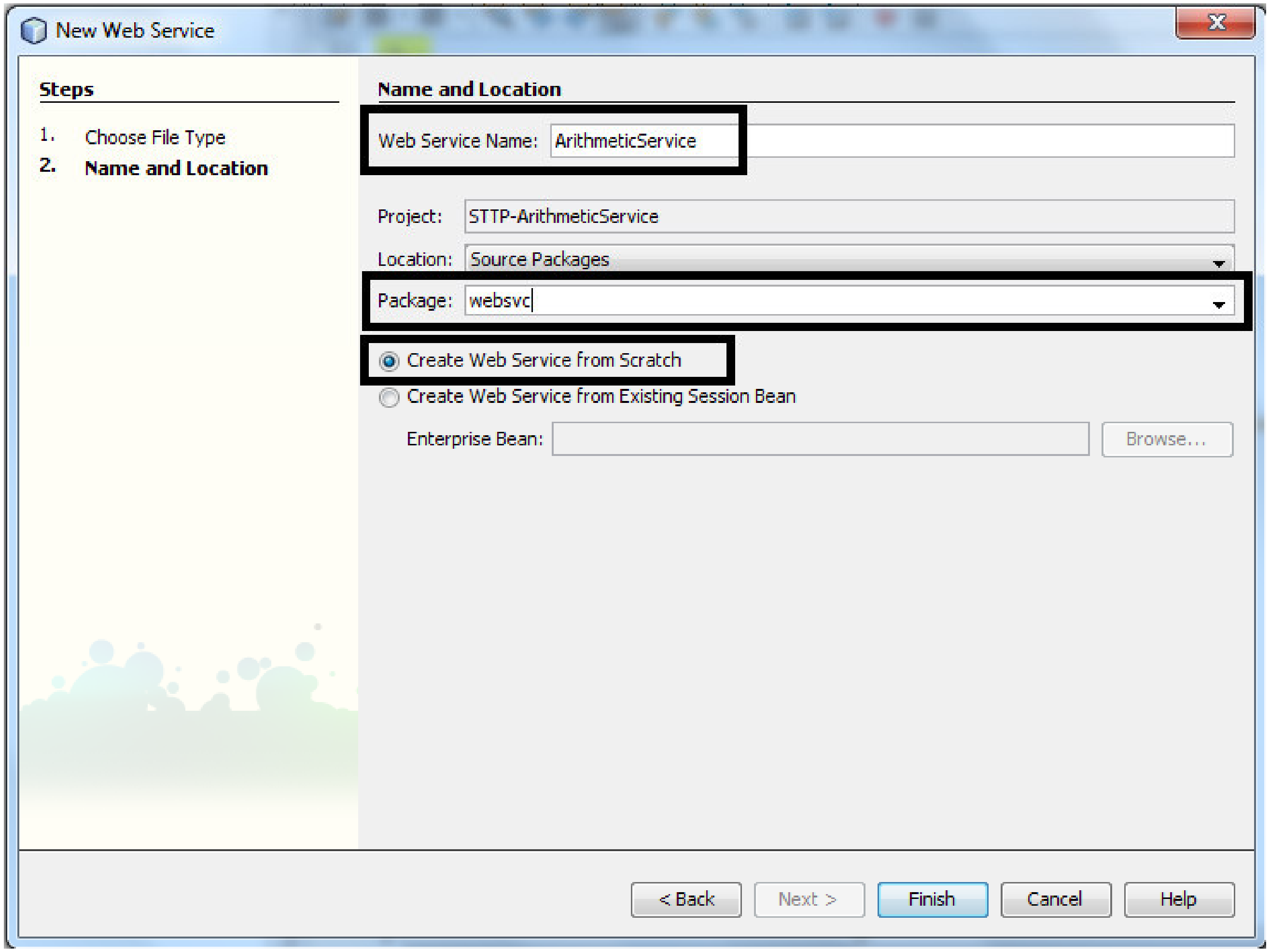
Creating Web Service using Code First Approach

- **Right click on project folder and select New-> Web Service**



Creating Web Service using Code First Approach

- **Provide Web Service Name, Package Name**



Name and Location

Web Service Name: ArithmeticService

Project: STTP-ArithmeticService

Location: Source Packages

Package: websvc

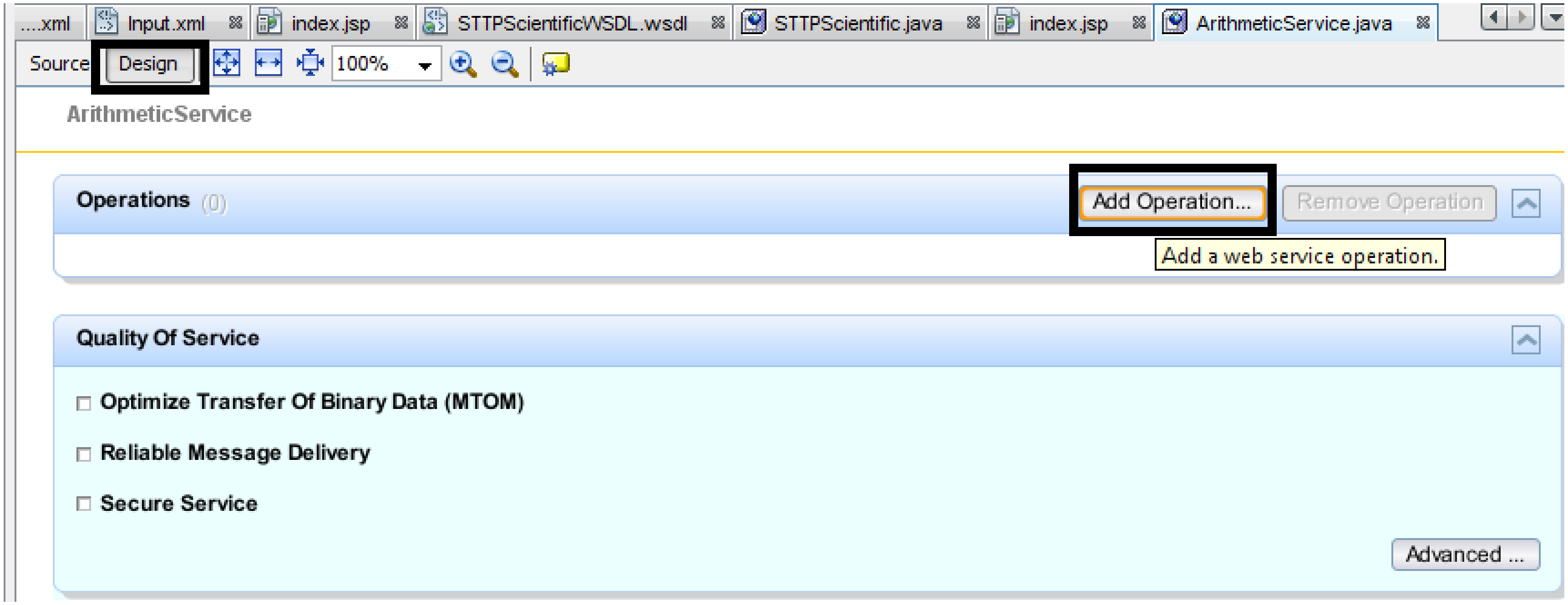
● Create Web Service from Scratch

○ Create Web Service from Existing Session Bean

Enterprise Bean:

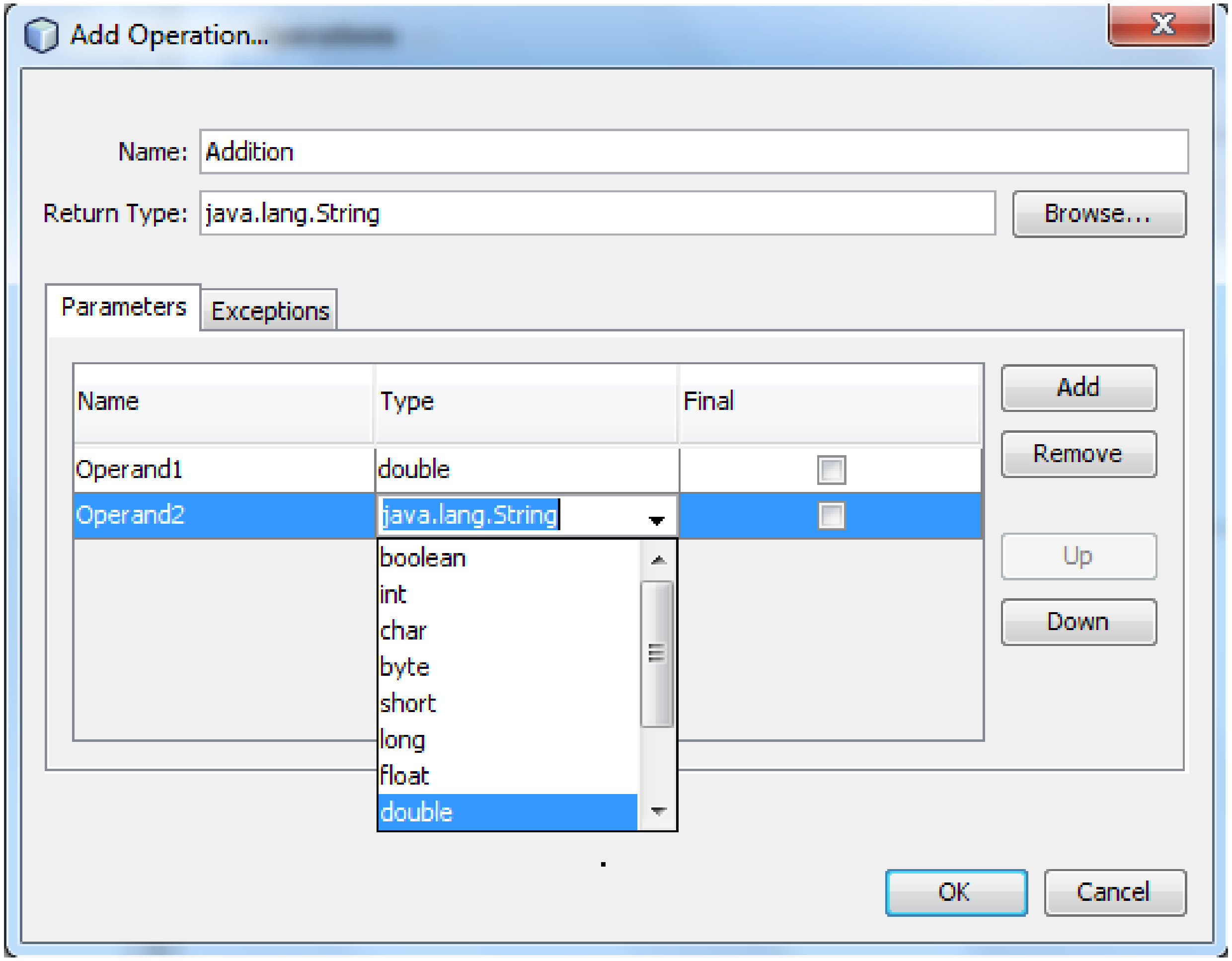
Creating Web Service using Code First Approach

- **Go to Design View -> Click Add Operation**



Creating Web Service using Code First Approach

Provide Following Details: Operation Name, Parameters and Return Type



Add Operation...

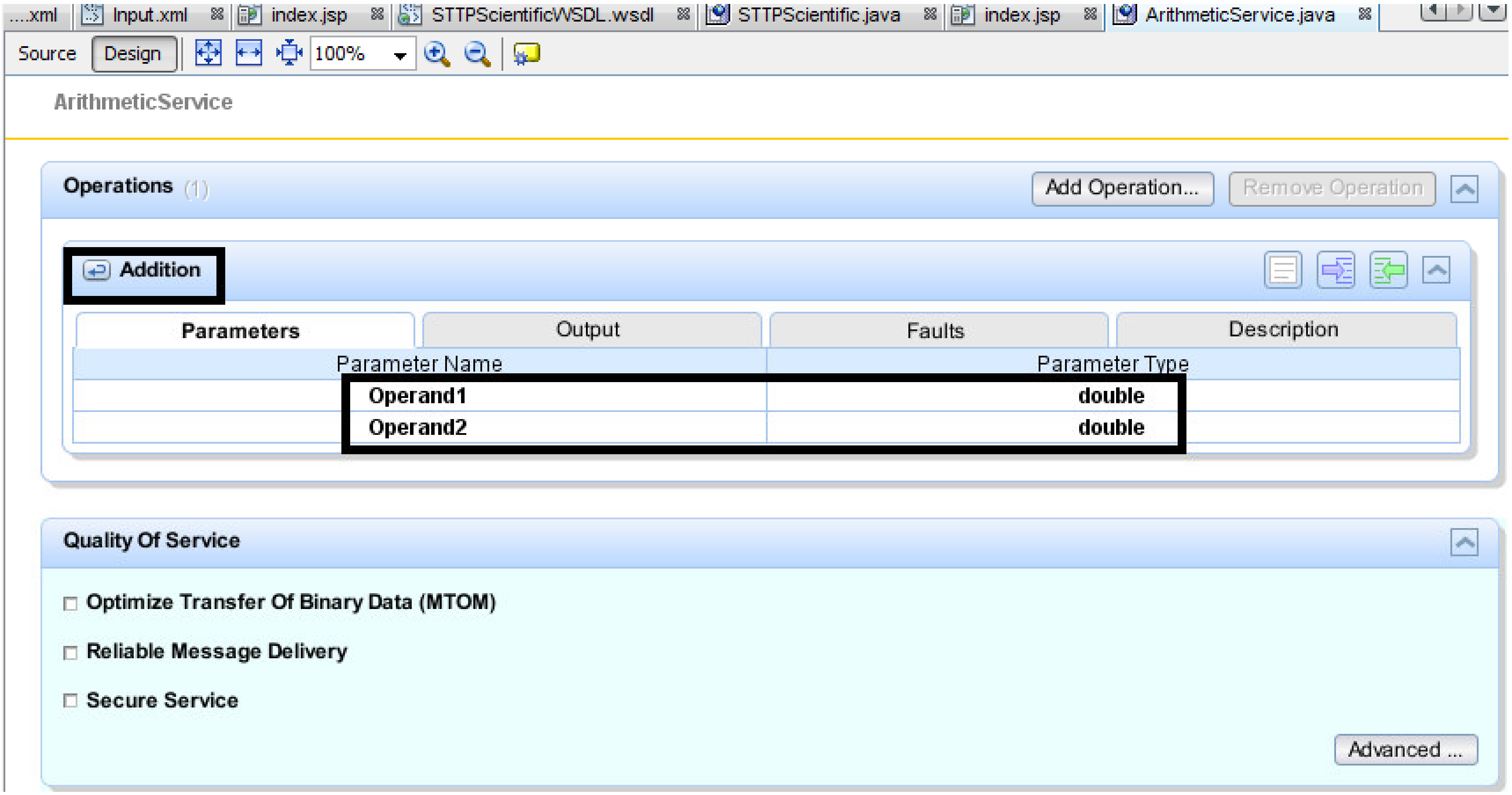
Name: Addition

Return Type: java.lang.String

Name	Type	Final
Operand1	double	□
Operand2	java.lang.String	□

Creating Web Service using Code First Approach

Note that the operation is added under the Web Service



Parameter Name	Parameter Type
Operand1	double
Operand2	double

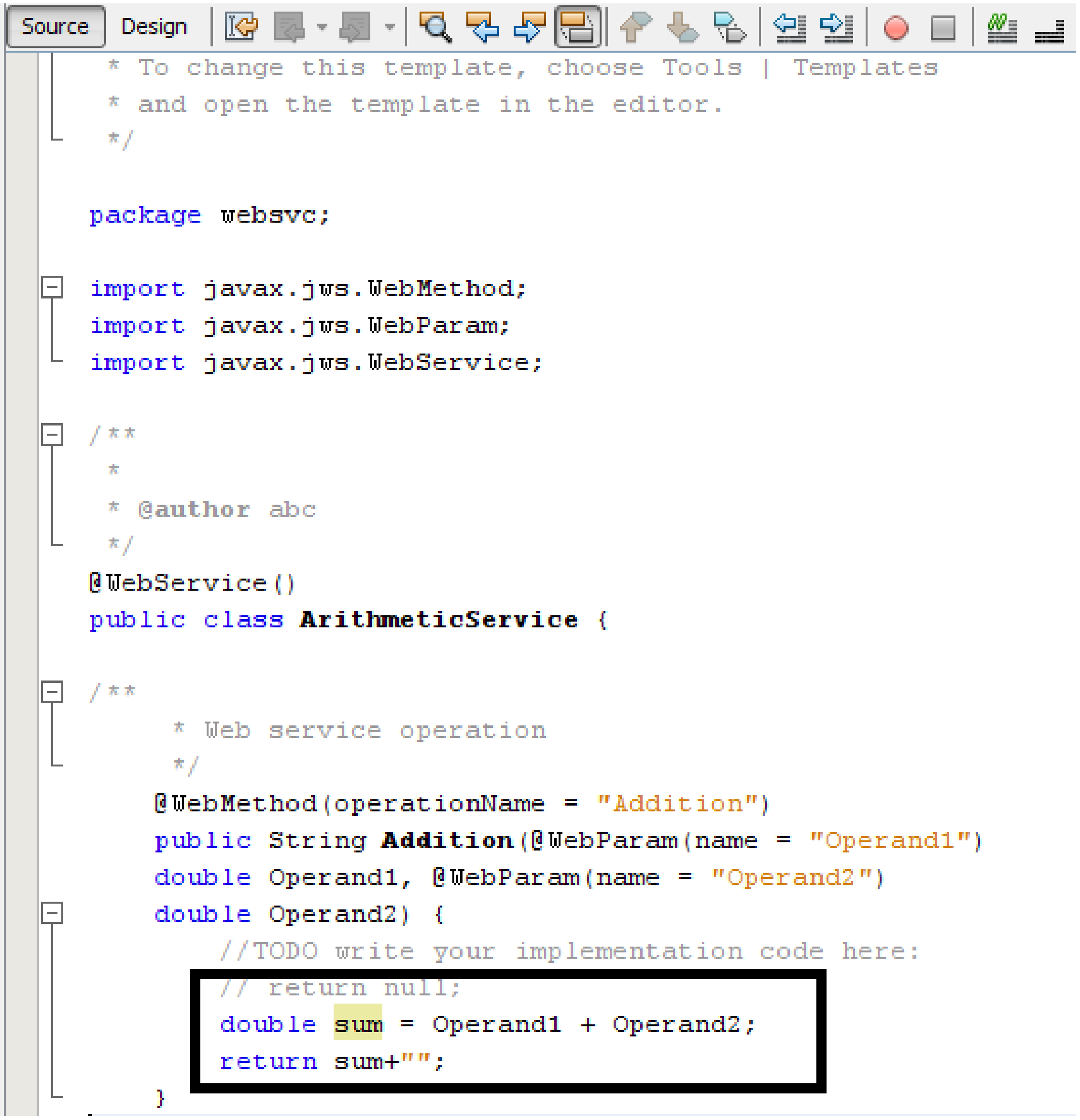
□ Optimize Transfer Of Binary Data (MTOM)

□ Reliable Message Delivery

□ Secure Service

Creating Web Service using Code First Approach

Go to the Source View of Web Service and Modify the Code of Addition Operation

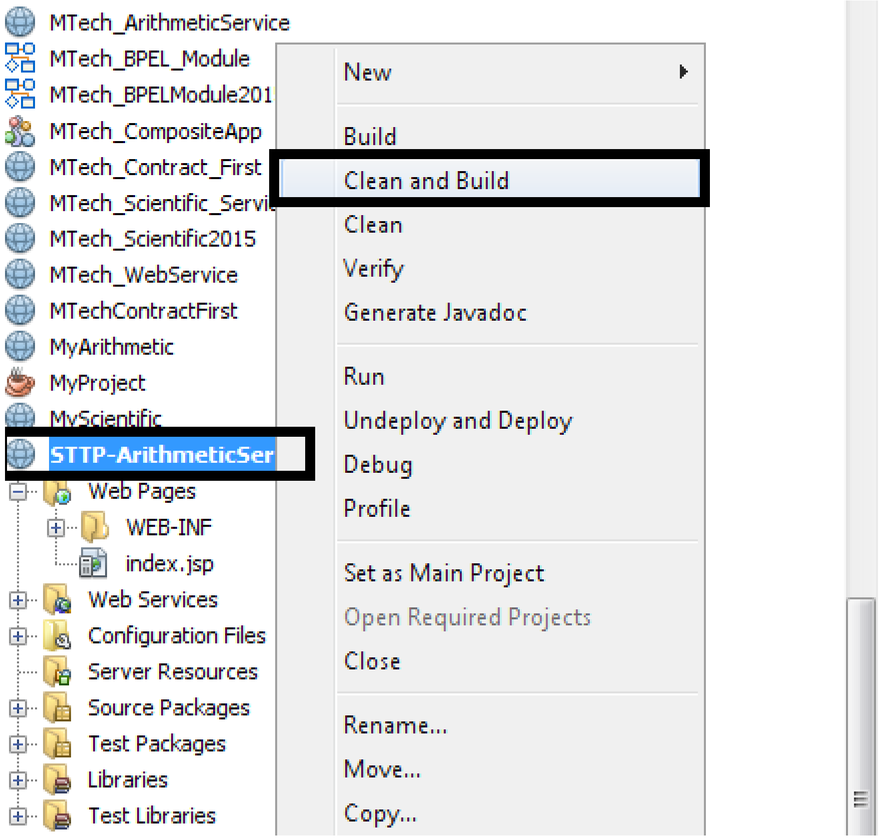


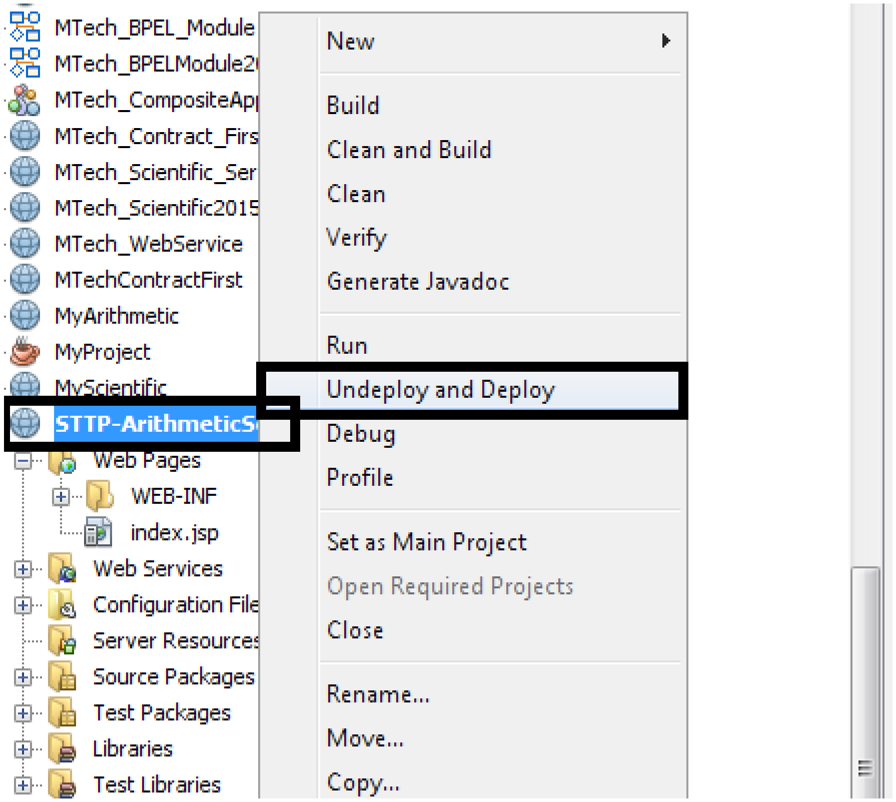
```
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package websvc;

import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;

/**
 *
 * @author abc
 */
@WebService()
public class ArithmeticService {

/**
     * Web service operation
     */
    @WebMethod(operationName = "Addition")
    public String Addition(@WebParam(name = "Operand1")
    double Operand1, @WebParam(name = "Operand2")
    double Operand2) {
        //TODO write your implementation code here:
        // return null;
        double sum = Operand1 + Operand2;
        return sum+"";
    }
```

Creating Web Service using Code First Approach

Clean and Build the Project

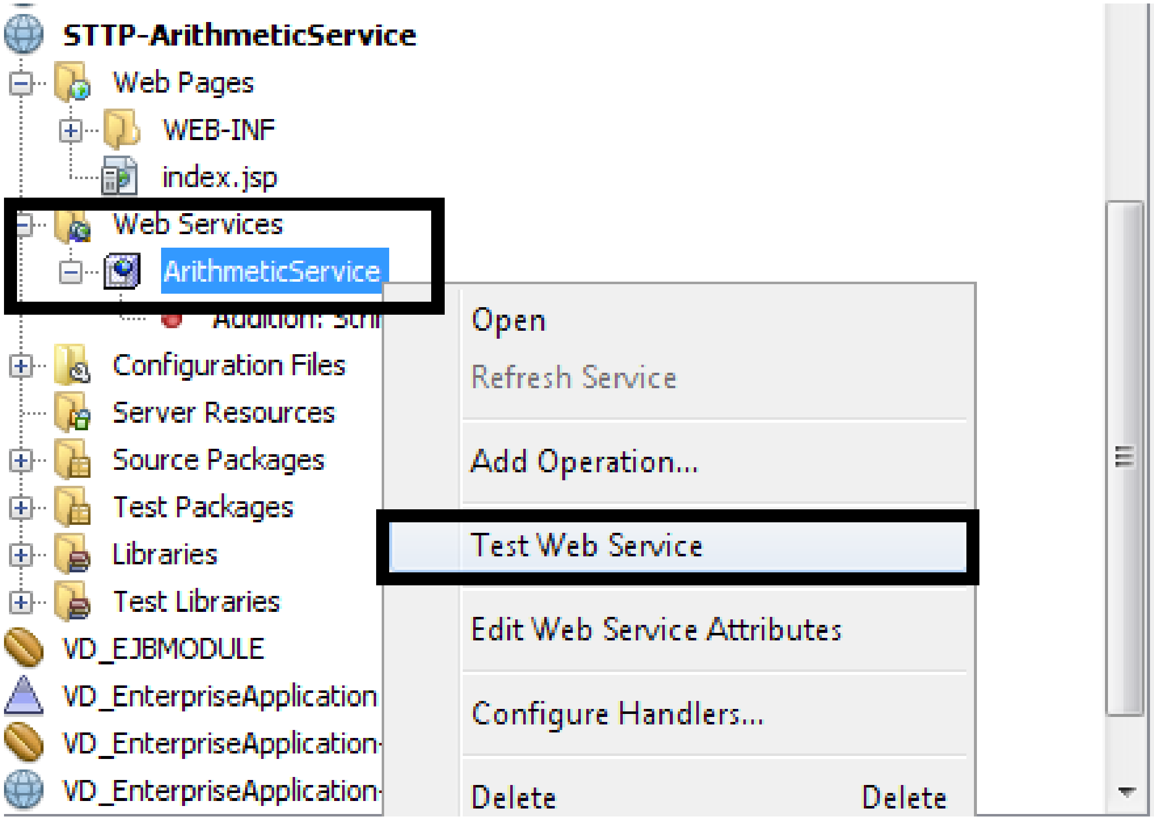
Deploy the Project





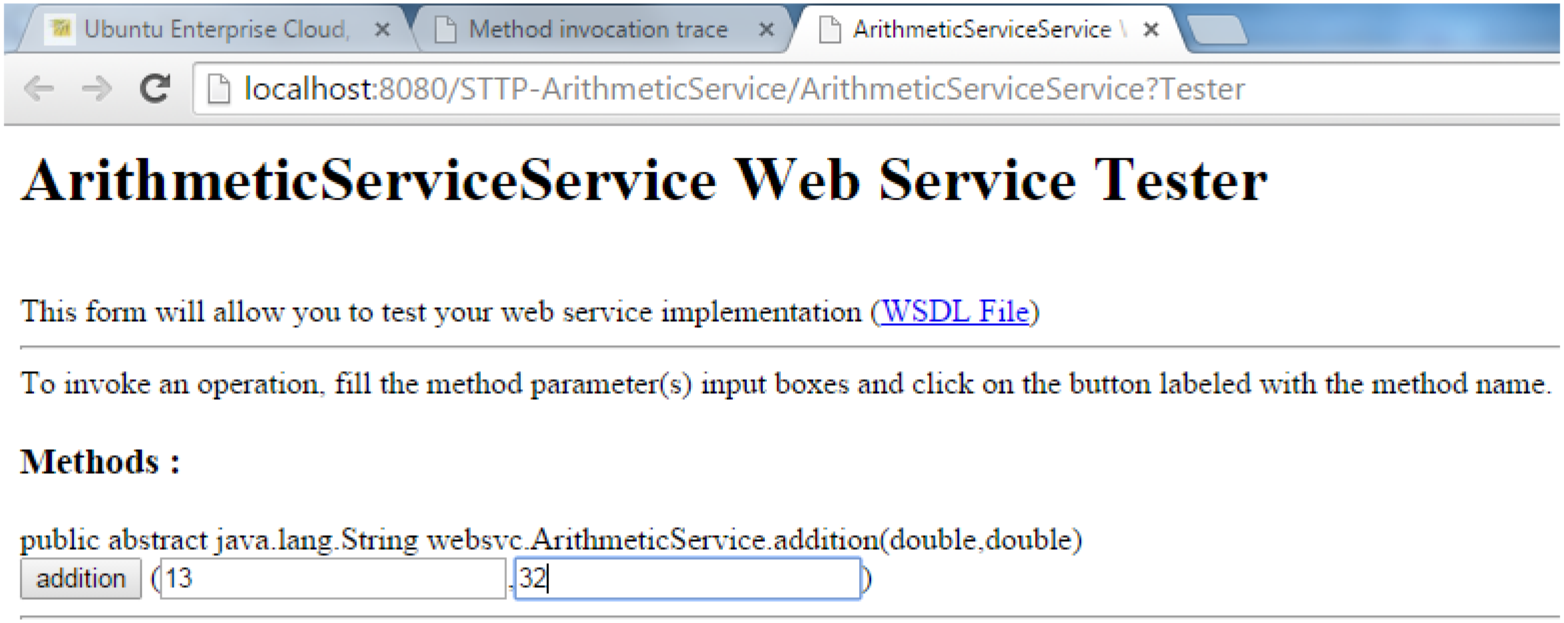
Creating Web Service using Code First Approach

Select Project -> Web Services -> ArithmeticService -> Test Web Service



Creating Web Service using Code First Approach

Test Web Service



Ubuntu Enterprise Cloud,
Method invocation trace
ArithmeticServiceService \
localhost:8080/STTP-ArithmeticService/ArithmeticServiceService?Tester

ArithmeticServiceService Web Service Tester

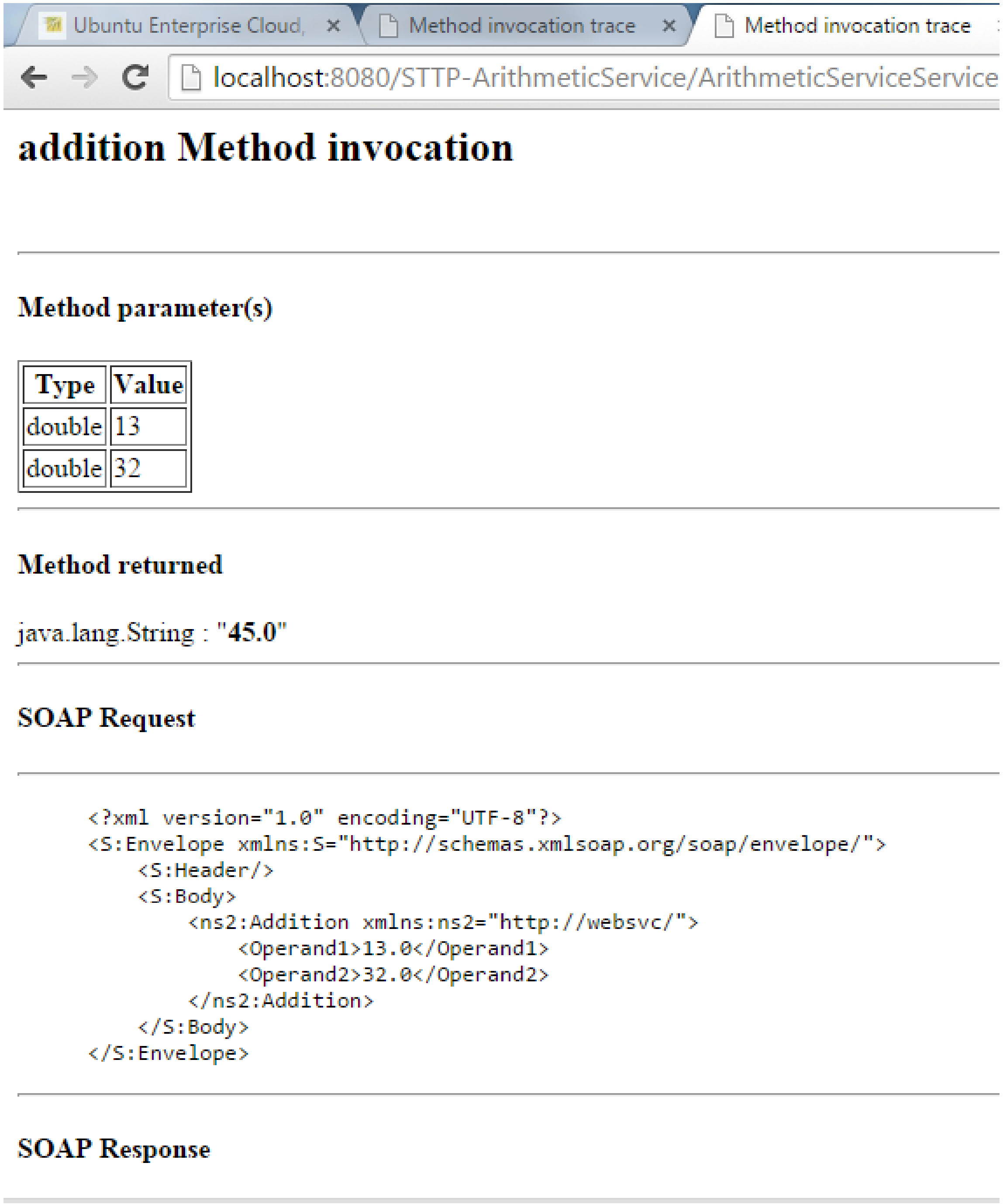
This form will allow you to test your web service implementation (WSDL File)

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

public abstract java.lang.String websvc.ArithmeticService.addition(double,double)

addition (13 ,32)



addition Method invocation

Method parameter(s)

Type	Value
double	13
double	32

Method returned

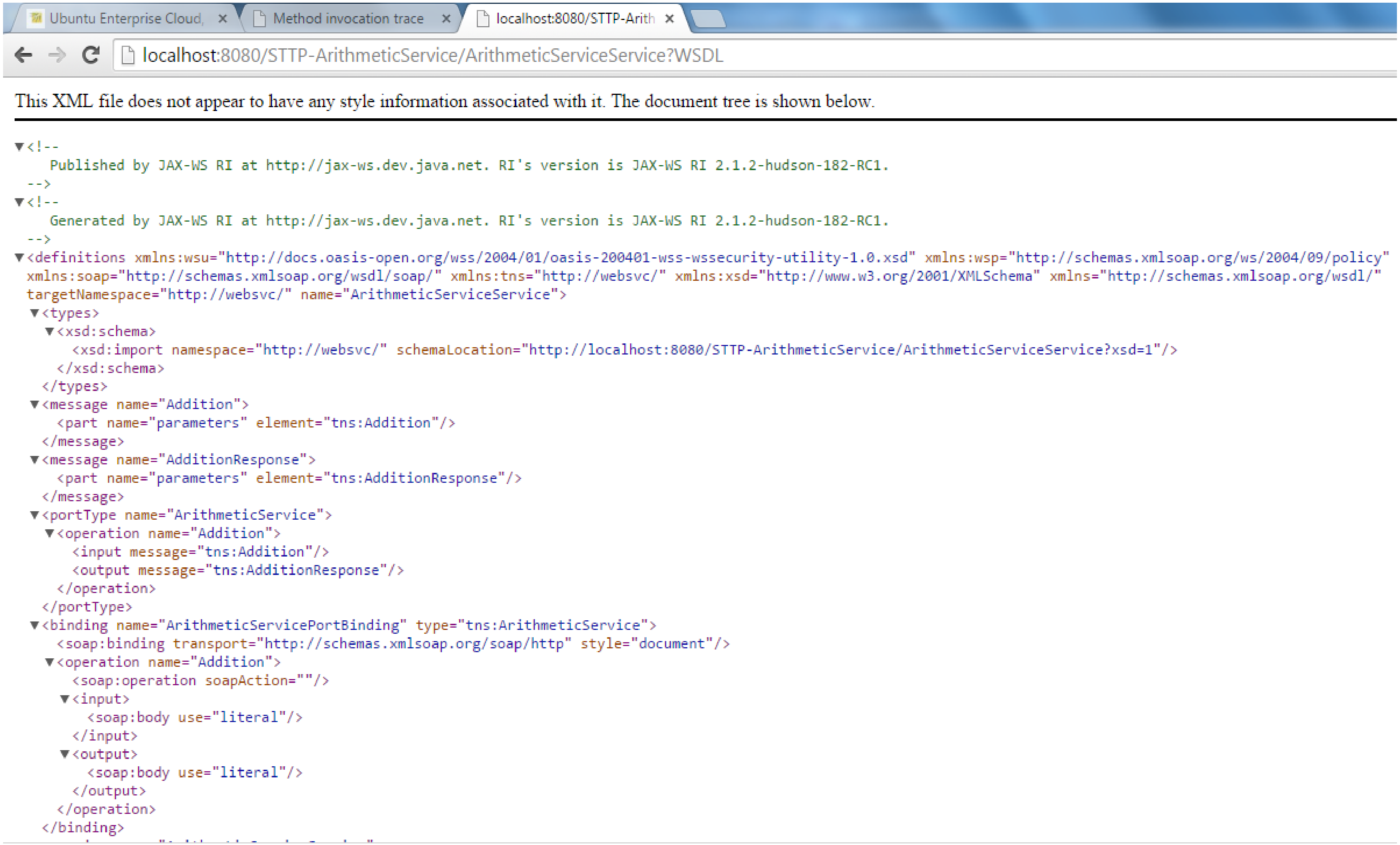
java.lang.String : "**45.0**"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:Addition xmlns:ns2="http://websvc/">
            <Operand1>13.0</Operand1>
            <Operand2>32.0</Operand2>
        </ns2:Addition>
    </S:Body>
</S:Envelope>
```

SOAP Response

Look at WSDL File

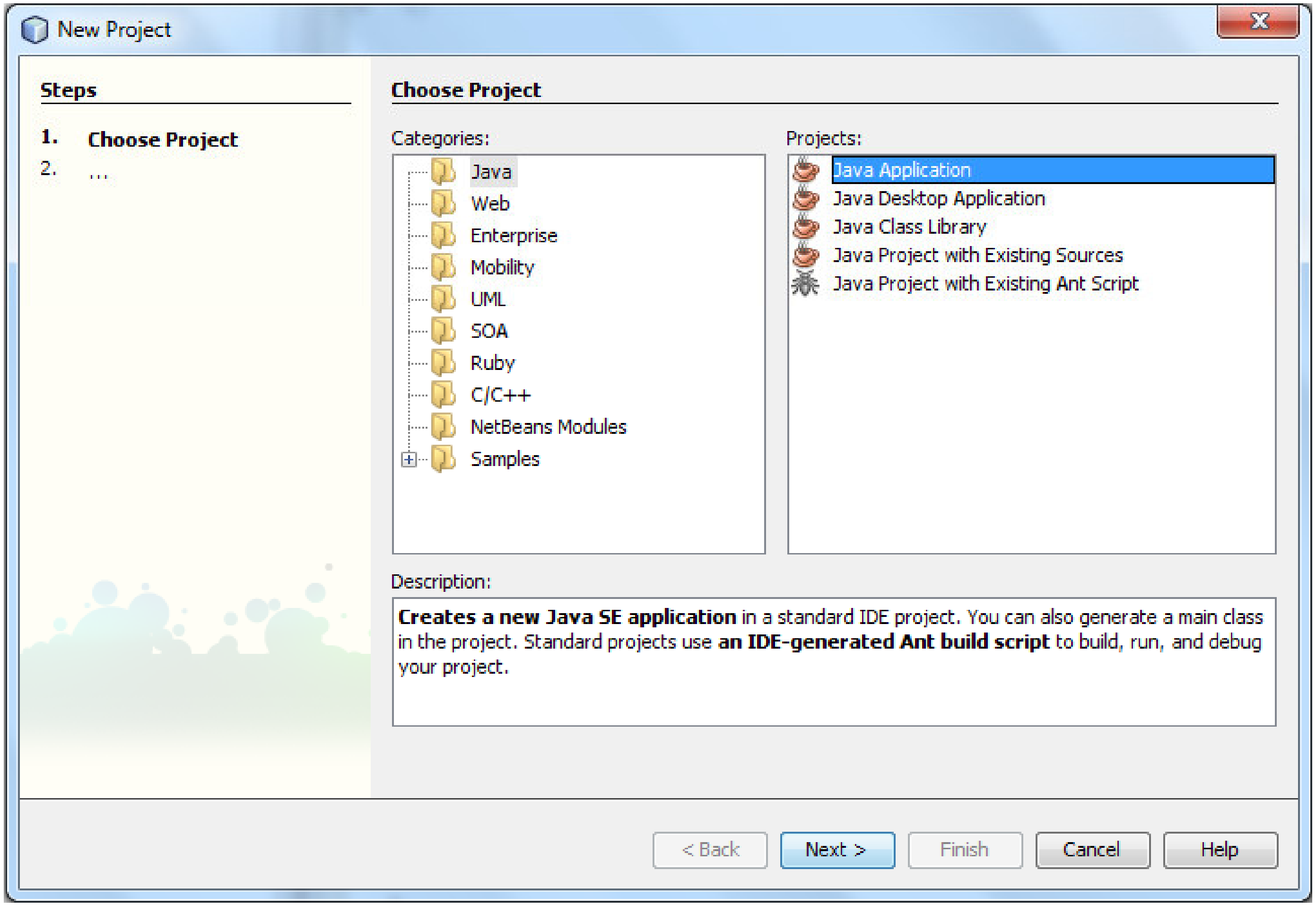

Ubuntu Enterprise Cloud,
Method invocation trace
localhost:8080/STTP-Arith
localhost:8080/STTP-ArithmeticService/ArithmeticServiceService?WSDL

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<!--
    Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.2-hudson-182-RC1.
 -->
▼<!--
    Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.2-hudson-182-RC1.
 -->
▼<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://websvc/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/"
 targetNamespace="http://websvc/" name="ArithmeticServiceService">
  ▼<types>
    ▼<xsd:schema>
        <xsd:import namespace="http://websvc/" schemaLocation="http://localhost:8080/STTP-ArithmeticService/ArithmeticServiceService?xsd=1"/>
     </xsd:schema>
   </types>
  ▼<message name="Addition">
     <part name="parameters" element="tns:Addition"/>
   </message>
  ▼<message name="AdditionResponse">
     <part name="parameters" element="tns:AdditionResponse"/>
   </message>
  ▼<portType name="ArithmeticService">
    ▼<operation name="Addition">
        <input message="tns:Addition"/>
        <output message="tns:AdditionResponse"/>
     </operation>
   </portType>
  ▼<binding name="ArithmeticServicePortBinding" type="tns:ArithmeticService">
     <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    ▼<operation name="Addition">
        <soap:operation soapAction=""/>
      ▼<input>
          <soap:body use="literal"/>
        </input>
      ▼<output>
          <soap:body use="literal"/>
        </output>
     </operation>
   </binding>
   <service name="ArithmeticServiceService">
```

Building Web-Service Client

Create a New Java Project



New Project

Steps

1. **Choose Project**
2. ...

Choose Project

Categories:

- Java
- Web
- Enterprise
- Mobility
- UML
- SOA
- Ruby
- C/C++
- NetBeans Modules
- Samples

Projects:

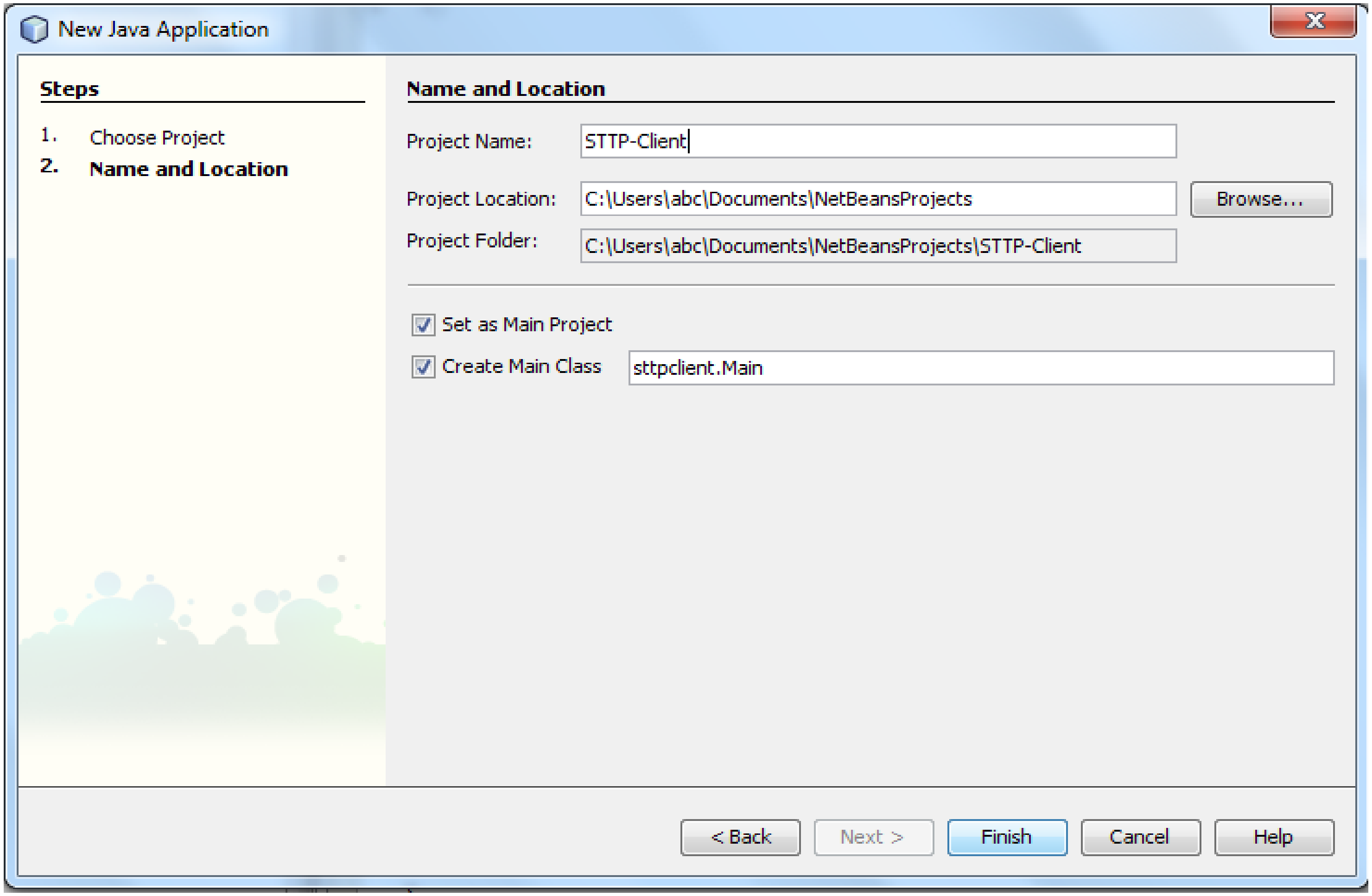
- Java Application
- Java Desktop Application
- Java Class Library
- Java Project with Existing Sources
- Java Project with Existing Ant Script

Description:

Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

Building Web-Service Client

Give Project Name – STTP-Client



New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: STTP-Client

Project Location: C:\Users\abc\Documents\NetBeansProjects

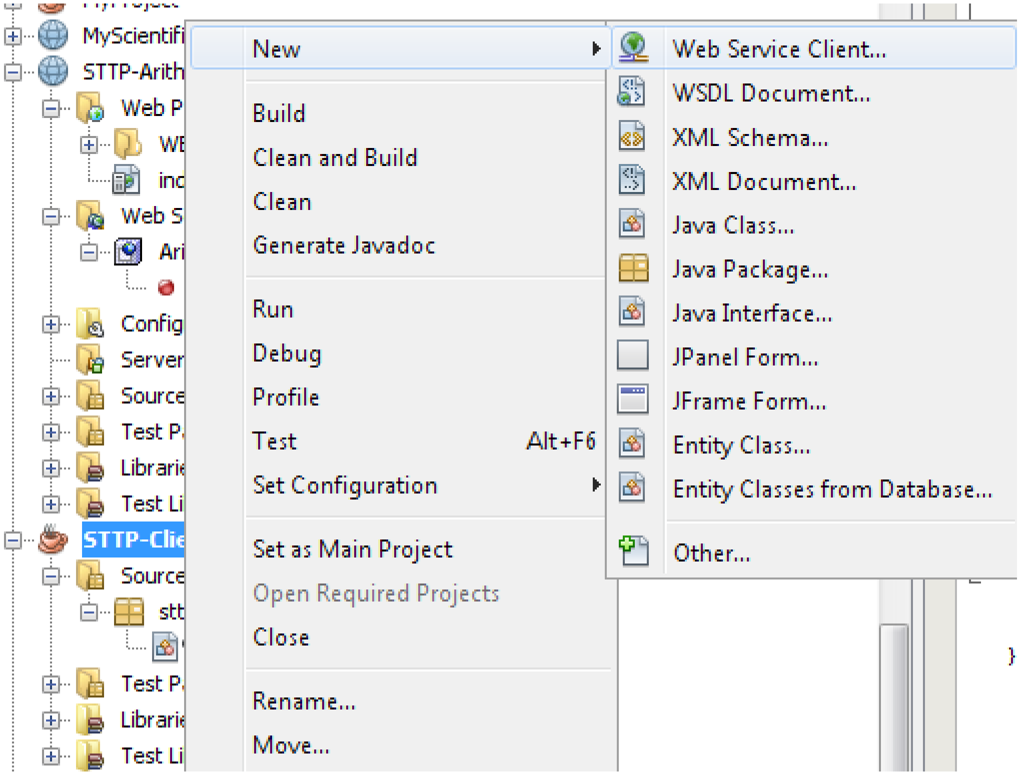
Project Folder: C:\Users\abc\Documents\NetBeansProjects\STTP-Client

☑ Set as Main Project

☑ Create Main Class sttpclient.Main

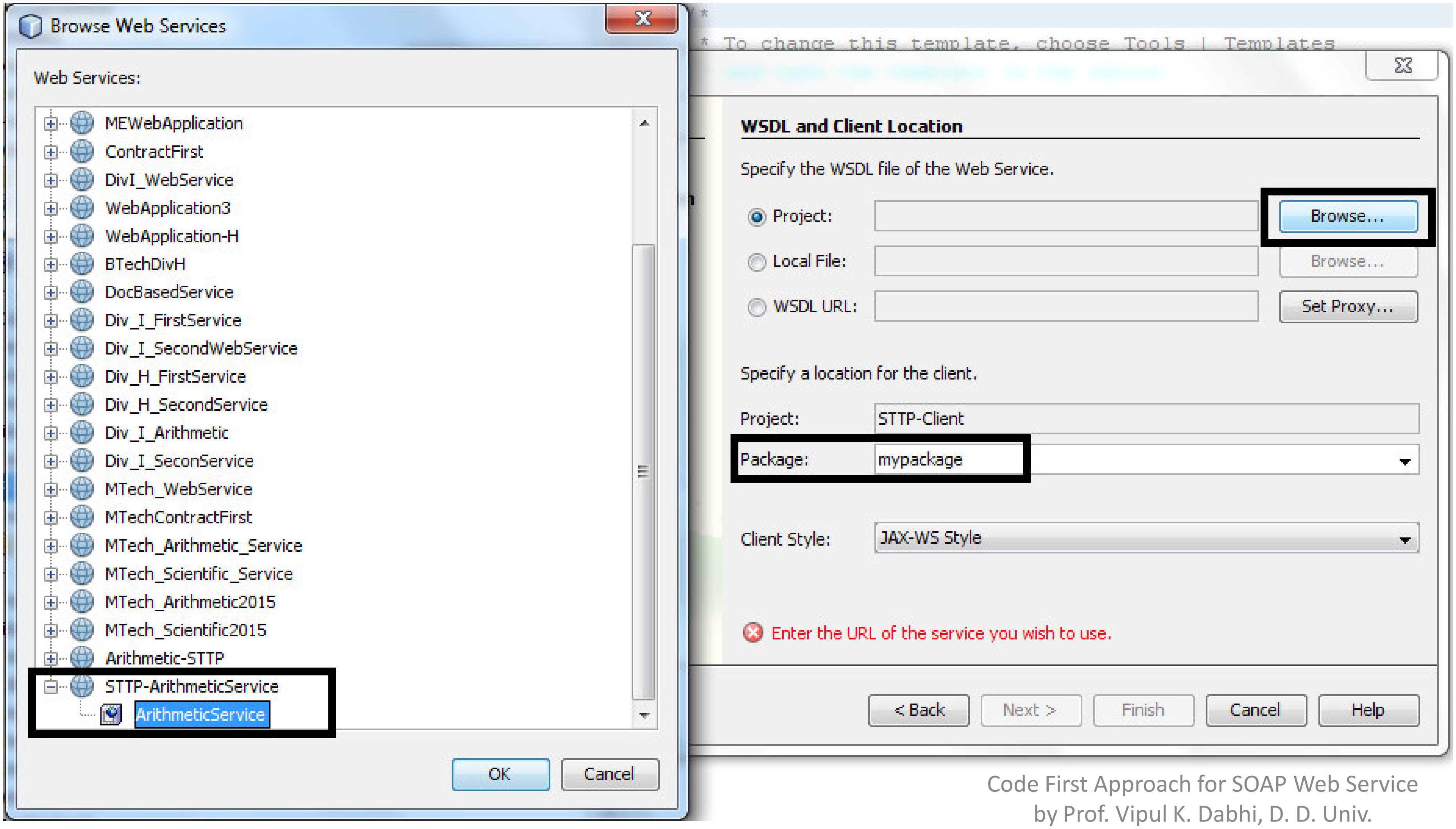
Building Web-Service Client

Select Project -> New -> Web Service Client



Building Web-Service Client

Specify WSDL Location and Package Name



Web Services:

- MEWebApplication
- ContractFirst
- DivI_WebService
- WebApplication3
- WebApplication-H
- BTechDivH
- DocBasedService
- Div_I_FirstService
- Div_I_SecondWebService
- Div_H_FirstService
- Div_H_SecondService
- Div_I_Arithmetic
- Div_I_SeconService
- MTech_WebService
- MTechContractFirst
- MTech_Arithmetic_Service
- MTech_Scientific_Service
- MTech_Arithmetic2015
- MTech_Scientific2015
- Arithmetic-STTP
- STTP-ArithmeticService
  - ArithmeticService

* To change this template, choose Tools | Templates

WSDL and Client Location

Specify the WSDL file of the Web Service.

● Project:

○ Local File:

○ WSDL URL:

Specify a location for the client.

Project: STTP-Client

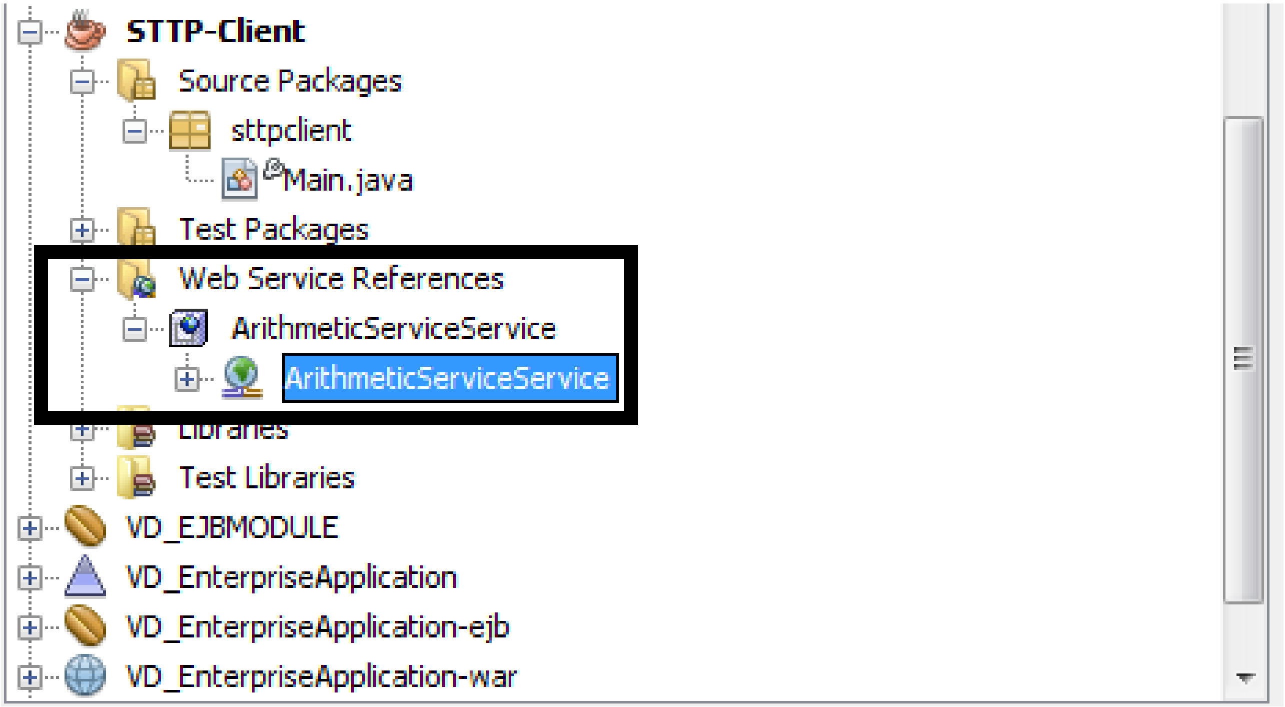
Package: mypackage

Client Style: JAX-WS Style

Enter the URL of the service you wish to use.

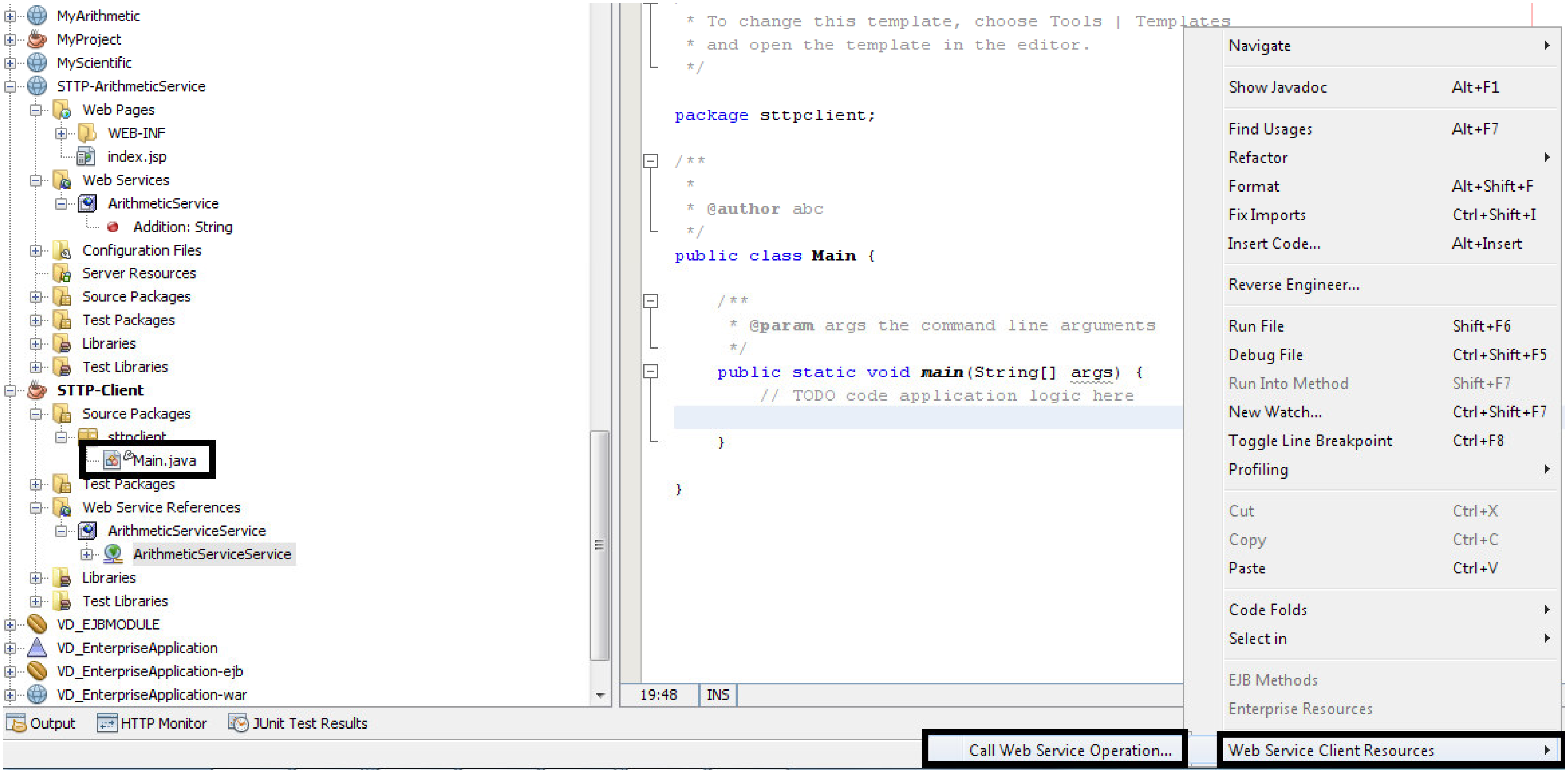
Building Web-Service Client

The Web Service Reference gets Added into the project



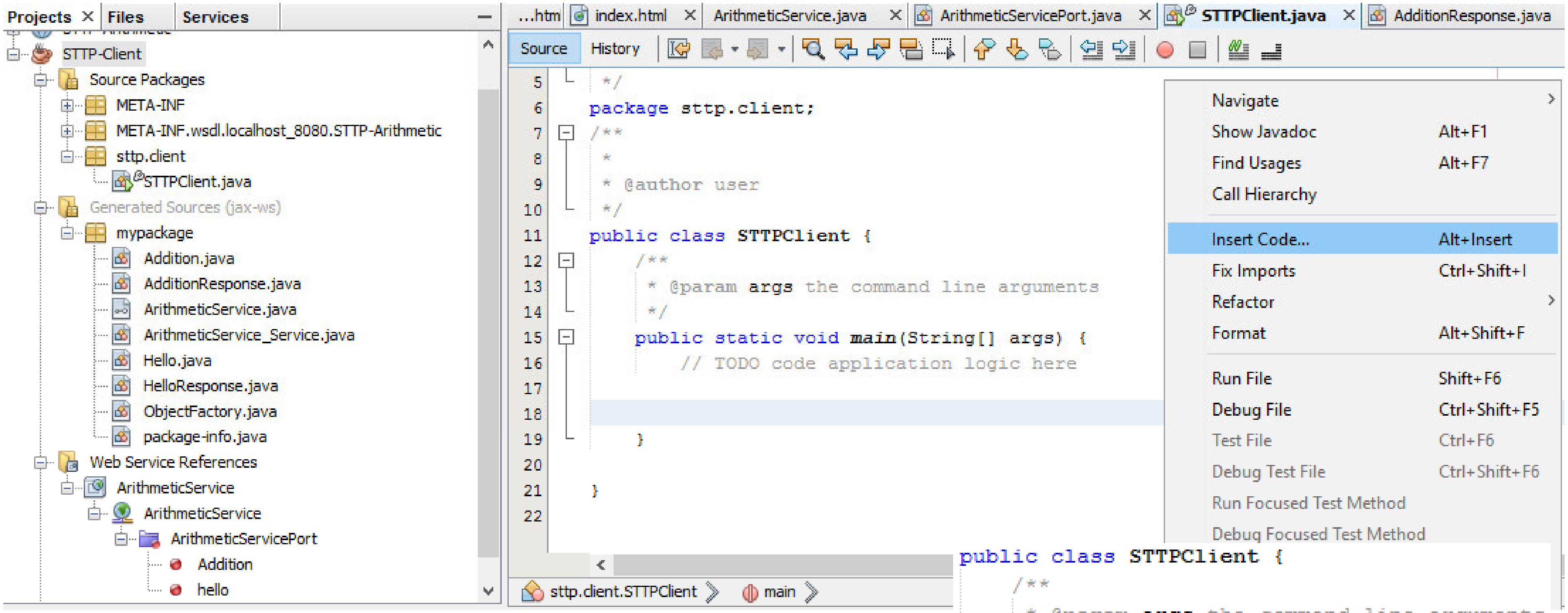
Building Web-Service Client

Go to source code of Main.java – Right click -> Web Service Client Resources -> Call Web Service Operation

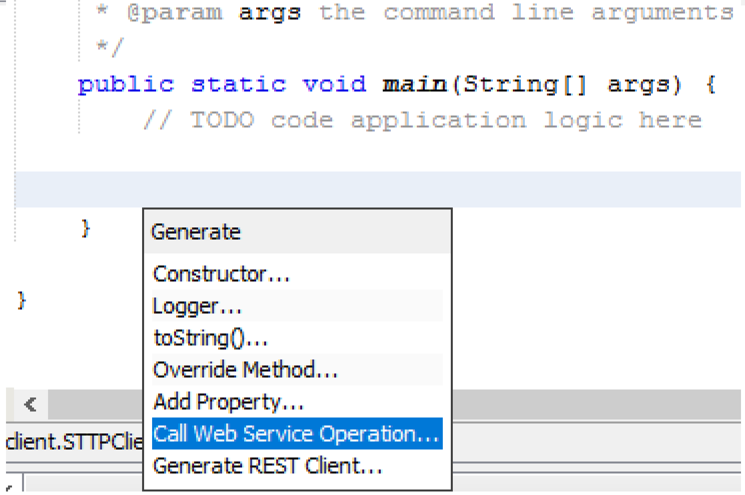


```
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package sttpclient;

/**
 *
 * @author abc
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here

    }

}
```

Building Web-Service Client



```
 */
package sttp.client;
/**
 *
 * @author user
 */
public class STTPClient {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here


    }

}
```



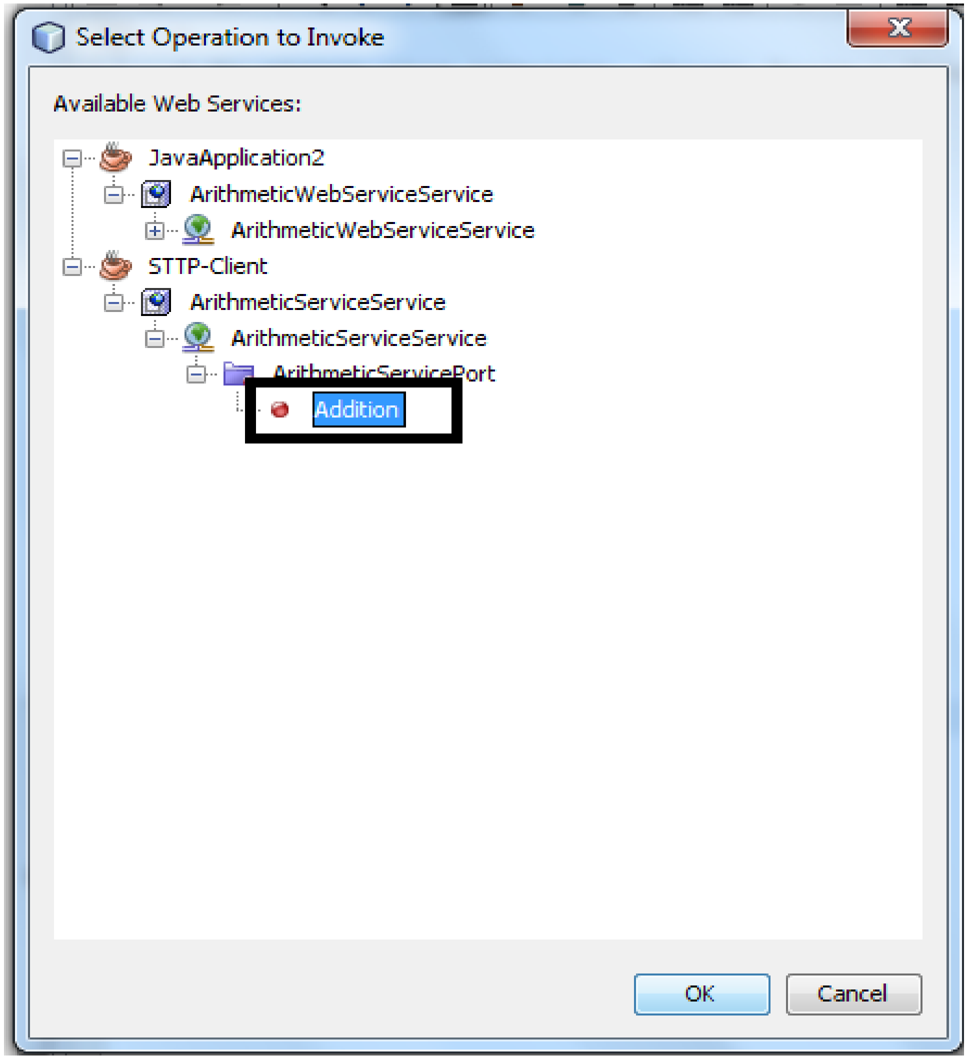
```
public class STTPClient {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here

    }
}
```

Netbeans 8.2:

Go to source code of STTPClient.java – Right click -> Insert Code -> Call Web Service Operation

Building Web-Service Client

Select STTP-Client Project. Select Operation Addition.



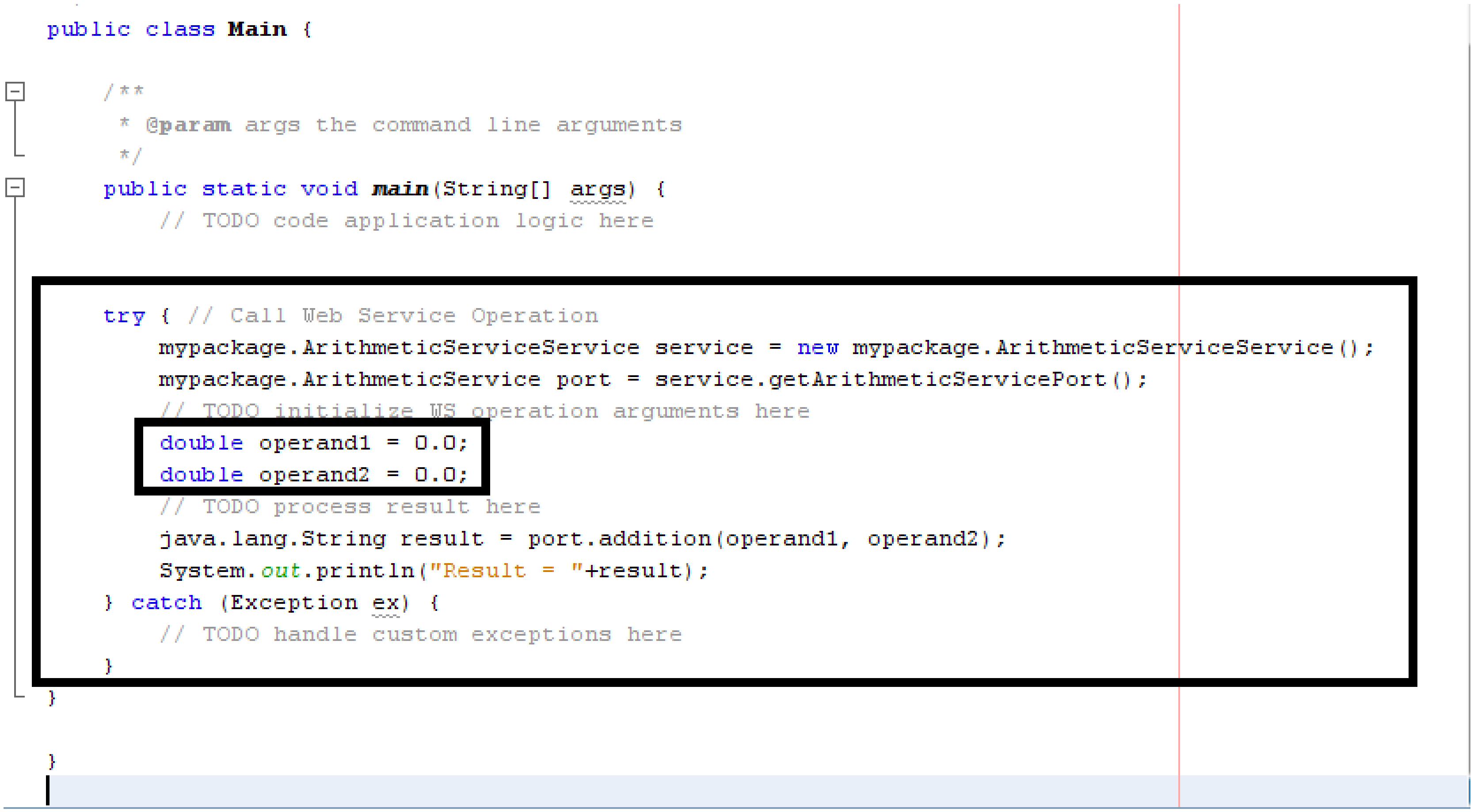
Select Operation to Invoke

Available Web Services:

- JavaApplication2
  - ArithmeticWebServiceService
    - ArithmeticWebServiceService
- STTP-Client
  - ArithmeticServiceService
    - ArithmeticServiceService
      - ArithmeticServicePort
        - Addition

Building Web-Service Client

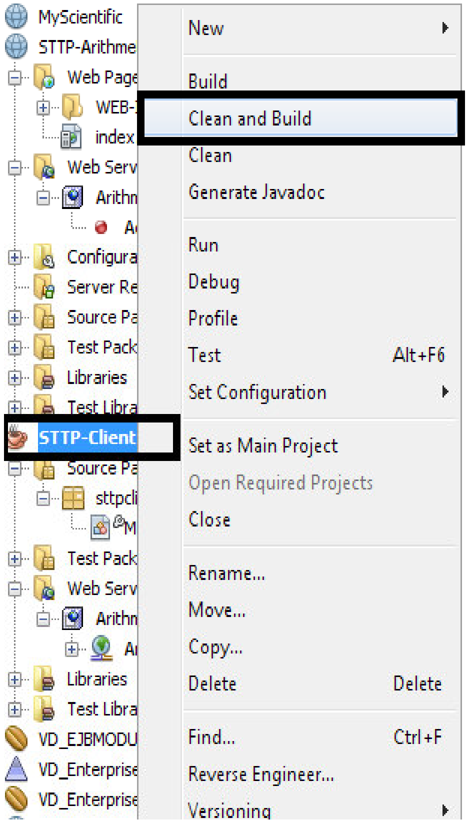
Notice that the code for calling Addition operation gets automatically added into source code of Main.java

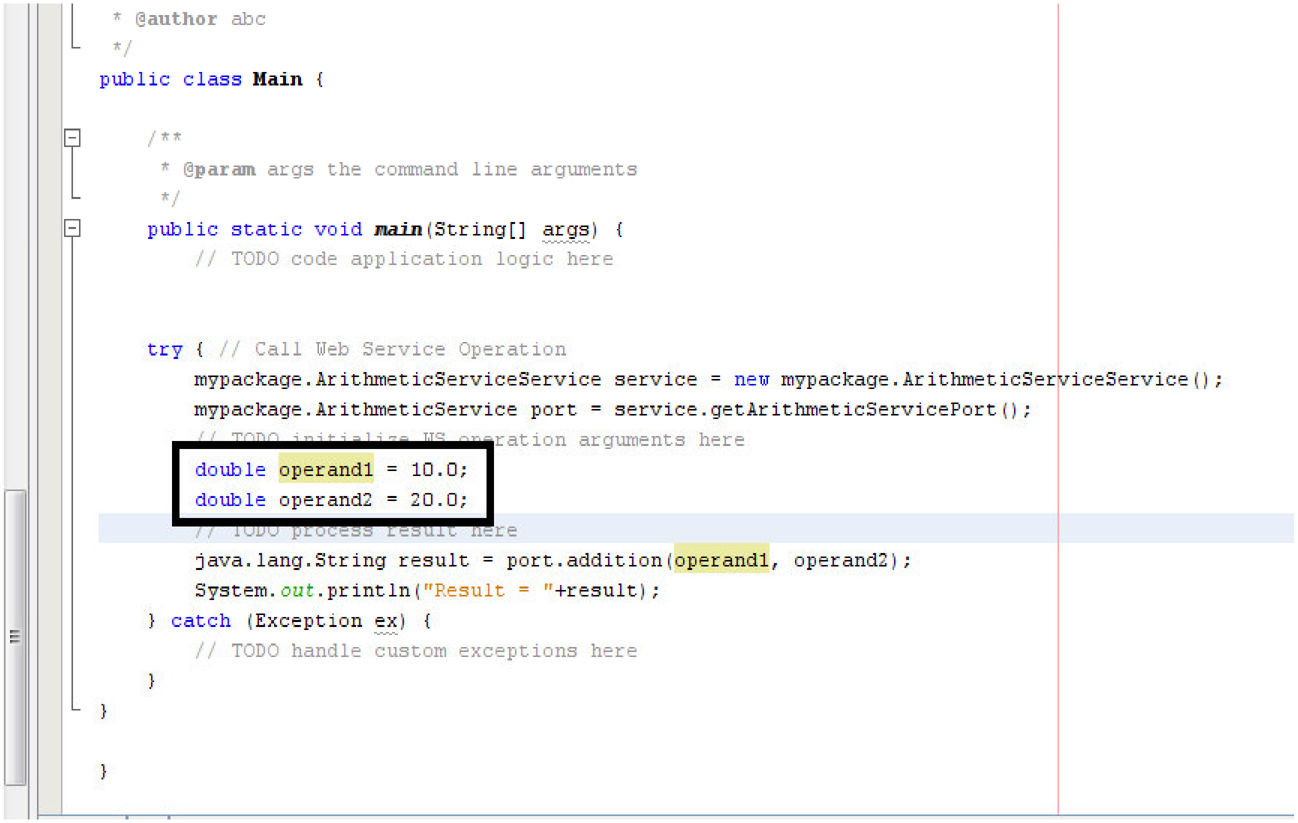


```
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here

    try { // Call Web Service Operation
        mypackage.ArithmeticServiceService service = new mypackage.ArithmeticServiceService();
        mypackage.ArithmeticService port = service.getArithmeticServicePort();
        // TODO initialize WS operation arguments here
        double operand1 = 0.0;
        double operand2 = 0.0;
        // TODO process result here
        java.lang.String result = port.addition(operand1, operand2);
        System.out.println("Result = "+result);
    } catch (Exception ex) {
        // TODO handle custom exceptions here
    }
}

}
```

Building Web-Service Client

Change the values of parameters in source code of Main.java. Clean and Build Application.



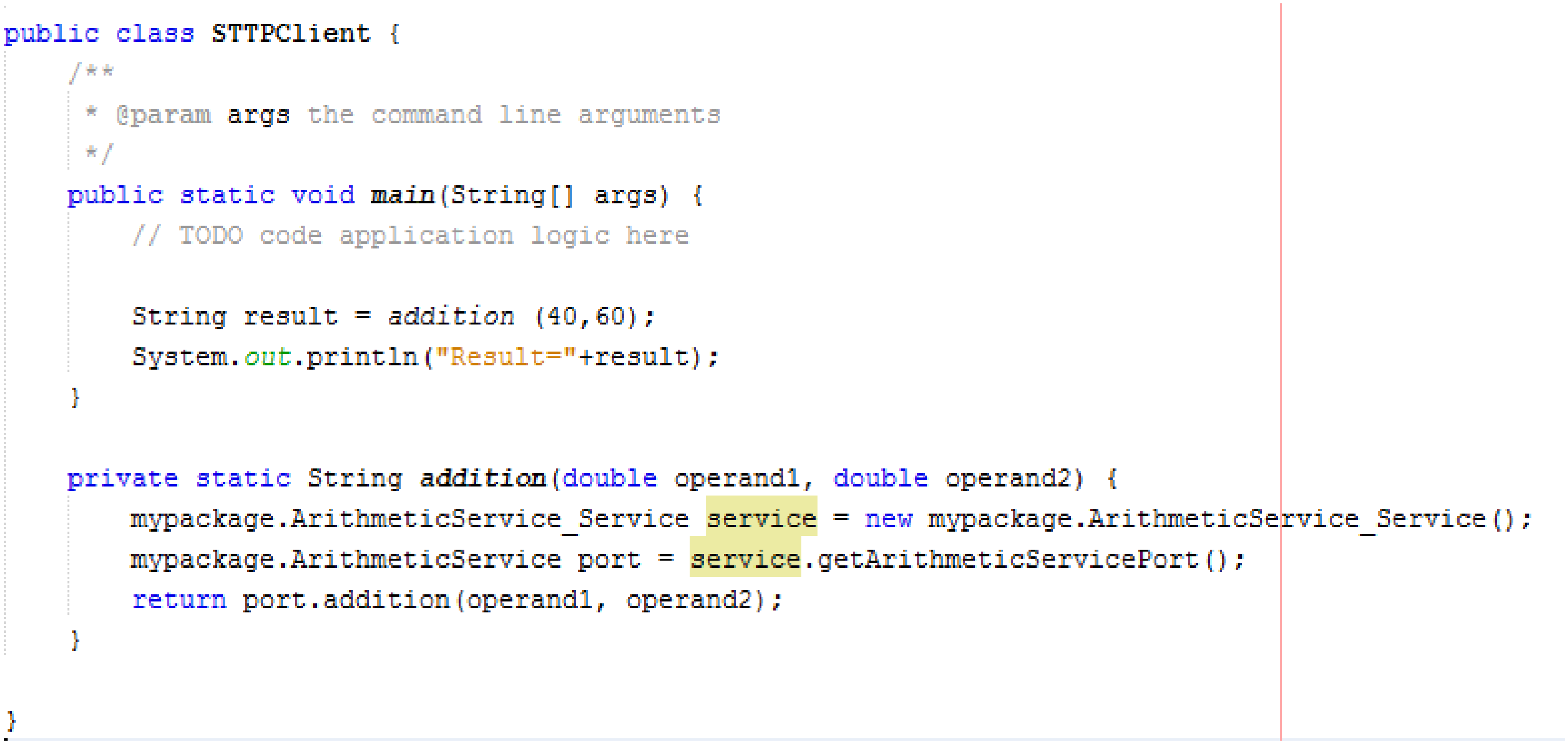


```
 * @author abc
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here


    try { // Call Web Service Operation
        mypackage.ArithmeticServiceService service = new mypackage.ArithmeticServiceService();
        mypackage.ArithmeticService port = service.getArithmeticServicePort();
        // TODO initialize WS operation arguments here
        double operand1 = 10.0;
        double operand2 = 20.0;
        // TODO process result here
        java.lang.String result = port.addition(operand1, operand2);
        System.out.println("Result = "+result);
    } catch (Exception ex) {
        // TODO handle custom exceptions here
    }
}

}
```

Building Web-Service Client

Netbeans 8.2

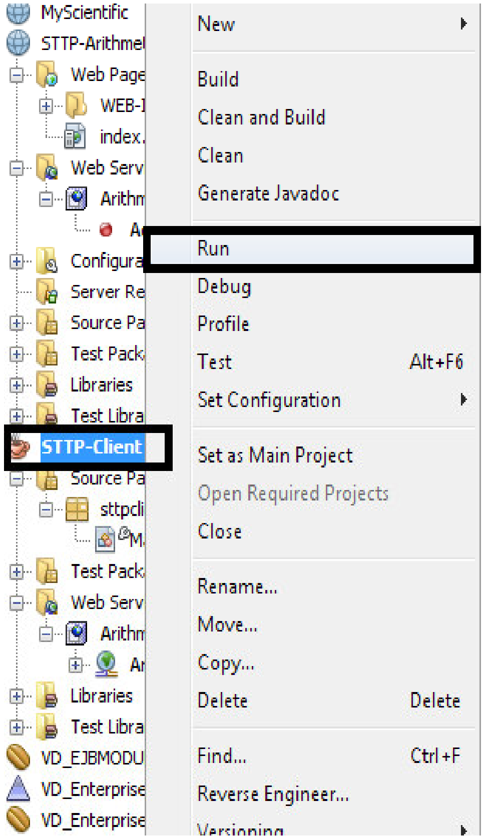
Notice that the code for calling addition operation gets automatically added into source code of STTPClient.java

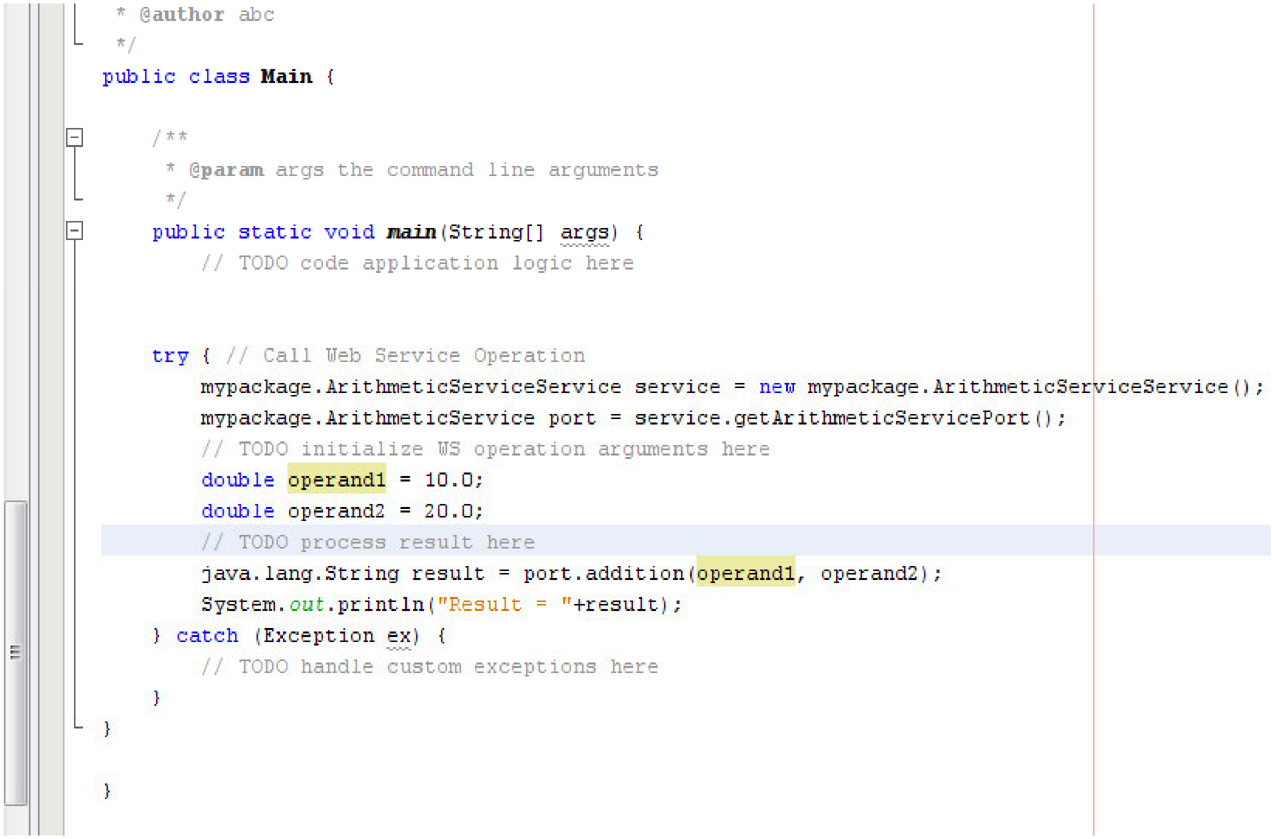


```java
public class STTPClient {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here

        String result = addition (40,60);
        System.out.println("Result="+result);
    }

    private static String addition(double operand1, double operand2) {
        mypackage.ArithmeticService_Service service = new mypackage.ArithmeticService_Service();
        mypackage.ArithmeticService port = service.getArithmeticServicePort();
        return port.addition(operand1, operand2);
    }

}
```

Building Web-Service Client

Run the Application.

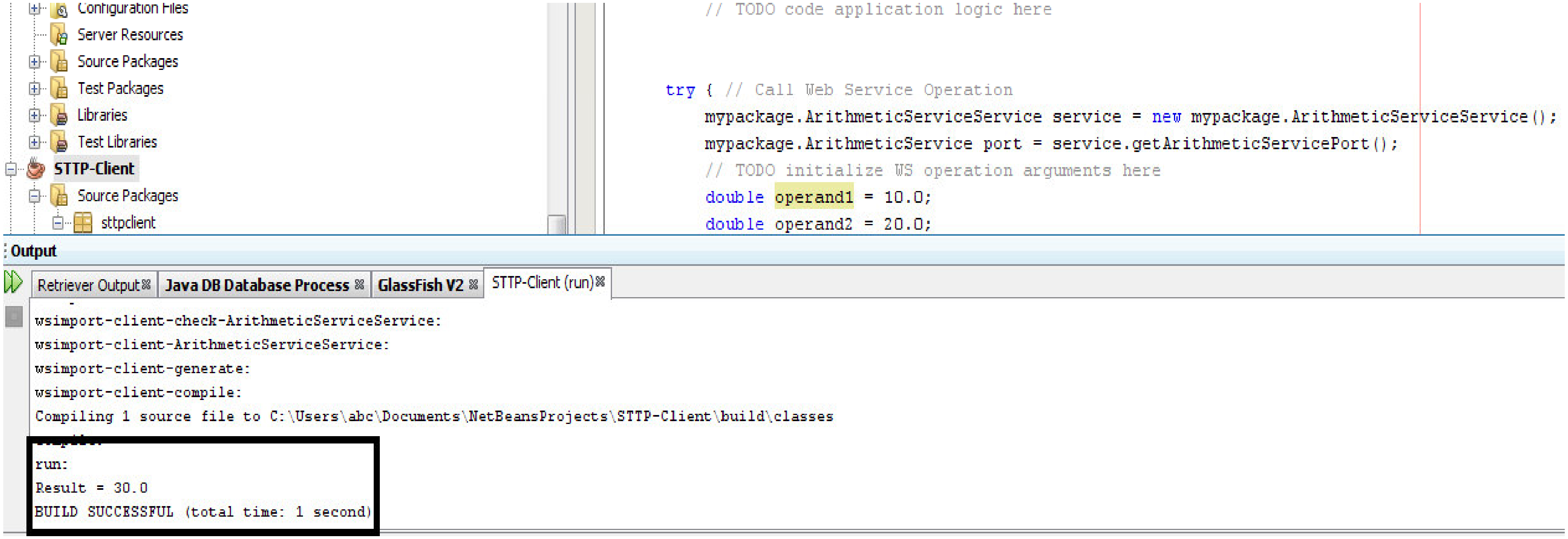




```
 * @author abc
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here


        try { // Call Web Service Operation
            mypackage.ArithmeticServiceService service = new mypackage.ArithmeticServiceService();
            mypackage.ArithmeticService port = service.getArithmeticServicePort();
            // TODO initialize WS operation arguments here
            double operand1 = 10.0;
            double operand2 = 20.0;
            // TODO process result here
            java.lang.String result = port.addition(operand1, operand2);
            System.out.println("Result = "+result);
        } catch (Exception ex) {
            // TODO handle custom exceptions here
        }
    }

}
```

Building Web-Service Client

Observe the output in Console.



```
        // TODO code application logic here


    try { // Call Web Service Operation
        mypackage.ArithmeticServiceService service = new mypackage.ArithmeticServiceService();
        mypackage.ArithmeticService port = service.getArithmeticServicePort();
        // TODO initialize WS operation arguments here
        double operand1 = 10.0;
        double operand2 = 20.0;
```

```
wsimport-client-check-ArithmeticServiceService:
wsimport-client-ArithmeticServiceService:
wsimport-client-generate:
wsimport-client-compile:
Compiling 1 source file to C:\Users\abc\Documents\NetBeansProjects\STTP-Client\build\classes
compile:
run:
Result = 30.0
BUILD SUCCESSFUL (total time: 1 second)
```