Assignment No.1

Implement multi-threaded client/server Process communication using RMI

Client.java

```
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
public class Client
{
        private Client() {}
         public static void main(String[] args) {
           try {
             // Getting the registry
             Registry registry = LocateRegistry.getRegistry(null);

             // Looking up the registry for the remote object
             Hello stub = (Hello) registry.lookup("Hello");

             // Calling the remote method using the obtained object
             stub.printMsg();

             // System.out.println("Remote method invoked");
           } catch (Exception e) {
             System.err.println("Client exception: " + e.toString());
             e.printStackTrace();
           }
```

```
        }
}
```

Hello.java

```
import java.rmi.Remote;

import java.rmi.RemoteException;

public interface Hello extends Remote

{

        void printMsg() throws RemoteException;

}
```

ImplExample.java

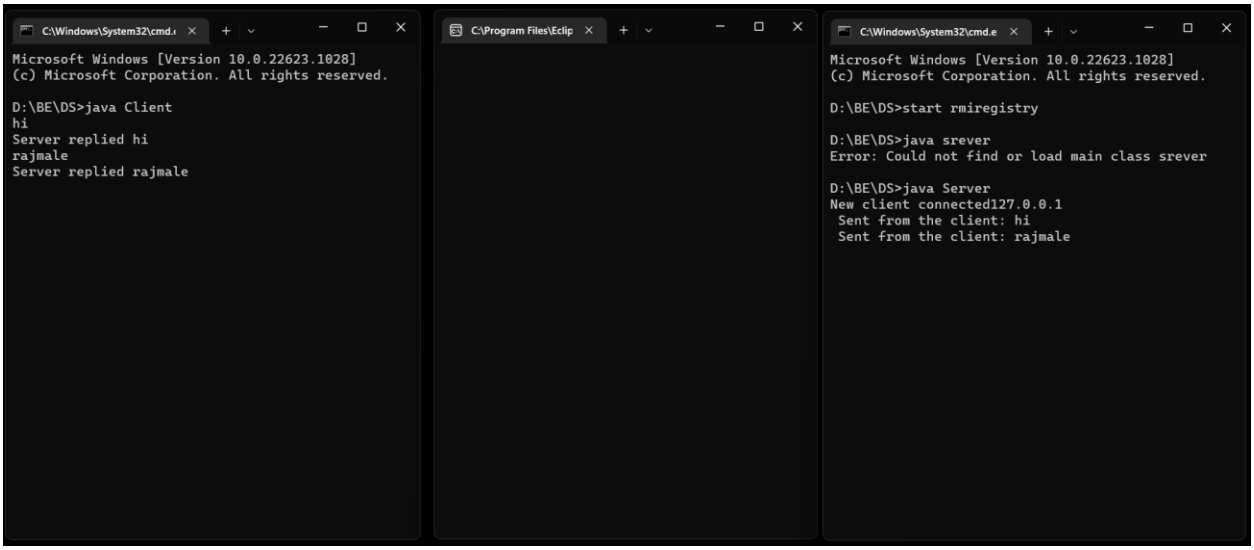
```
public class ImplExample
{
        public void printMsg() {
            System.out.println("This is an example RMI program");
          }

}
```

Server.java

```
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class Server extends ImplExample
{
        public Server() {}
          public static void main(String args[]) {
            try {
              // Instantiating the implementation class
              ImplExample obj = new ImplExample();
```

```
            // Exporting the object of implementation class
            // (here we are exporting the remote object to the stub)
            Hello stub = (Hello) UnicastRemoteObject.exportObject((Remote) obj, 0);

            // Binding the remote object (stub) in the registry
            Registry registry = LocateRegistry.getRegistry();

            registry.bind("Hello", stub);
            System.err.println("Server ready");
          } catch (Exception e) {
            System.err.println("Server exception: " + e.toString());
            e.printStackTrace();
          }
        }

}
```

Output:



Assignment No.2

Develop any distributed application using CORBA to demonstrate object brokering.

Steps:

1. Create a new folder C:\Calc in C:\ directory.

2. Create Calc.idl file using Notepad. Save it with in C:\Calc folder with idl as extension. Paste the following into the idl file:

```
module WssCalculator
{
interface Calc
{

//Performs the Calculations:ADD/SUB/MUL/DIV
long calculate(in long operator,in long num1,in long num2);

//The Server EXITS when the Client prompts it to do so
oneway void shutdown();

};
};
```

3. This file will define language neutral definition for Remote Interface called Calc with specified Methods/Functions (Note: It can be Java or C++ or any OOP based object).

4. Next, compile the .idl file, open the command prompt and change the directory to C:\Calc. Now enter the following command:

```
idlj -fall Calc.idl
```

The -fall specifies create binding for both client as well as server.
This will create the folder C:\Calc\WssCalculator and the following java files within it:
_CalcStub.java
Calc.java
CalcHelper.java
CalcHolder.java
CalcOperations.java
CalcPOA.java

5. Now a java file named CalcServer is created in C:\Calc with the following code:

```
//Importing all the packages and classes

//Import the package which contains the Server Skeleton
import WssCalculator.*;

//Import the below two packages to use the Naming Service
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;

//Import this package to run the CORBA Application
import org.omg.CORBA.*;//

//Import the below to Classes for inheriting Portable Server
import org.omg.PortableServer.*;
import org.omg.PortableServer.POA;

//Initiate the ORB using the class Properties
import java.util.Properties;

//Perform the Input-Output functionalities
import java.io.*;
import java.util.*;


//Write the Servant class
//It inherits the general CORBA utilities generated by the Compiler

class Calcserverimpl extends CalcPOA

{

//orb variable is used to invoke the shutdown()
private ORB orb;

public void setORB(ORB orb_val)

{

orb = orb_val;

}


//Declaring and Implementing the required method
public int calculate(int a,int b,int c)
```

```
{

//ADDITION
if(a==43)
{
return (b+c);
}

//SUBTRACTION
else if(a==45)
{
return (b-c);
}

//MULTIPLICATION
else if(a==42)
{
return (b*c);
}

//DIVISION
else if(a==47)
{
return (b/c);
}

//DEFAULT
else
{
return 0;
}

}

//Closing the server
 public void shutdown()
{
 orb.shutdown(false);
}

}//end of the servant class


public class CalcServer

{
```

```
public static void main(String args[])

{

try

{

//Create and Initialize the ORB object
//init() allows to set the properties at run time


ORB orb=ORB.init(args,null);


//Obtain the initial Naming Context
//Obtain an initial object reference to the name server


//orb retrieves the reference to the Root POA
//Activate the POA Manager
//activate() causes the POAs to process the client requests


POA rootpoa=POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
rootpoa.the_POAManager().activate();


//The server instantiates the servant objects
//The servant performs the operations defined in the idlj interface


Calcserverimpl simpl=new Calcserverimpl();
simpl.setORB(orb);


//Get the object reference associated with the servant
//narrow() is used to cast CORBA obj ref to its proper type


org.omg.CORBA.Object ref = rootpoa.servant_to_reference(simpl);
Calc href=CalcHelper.narrow(ref);

//Obtain the initial Naming Context
//Obtain an object reference to the Name Server
```

```
org.omg.CORBA.Object objRef=orb.resolve_initial_references("NameService");

//Narrow the objref to its proper type

NamingContextExt ncRef=NamingContextExtHelper.narrow(objRef);

//Register the Servant with the Name Server

String name = "Calc";

//NameComponent array contains the path to Calc

NameComponent path[]=ncRef.to_name(name);

//Pass the path and the servant object to the Naming Service
//Bind the servant object to Calc

ncRef.rebind(path,href);

System.out.println("The SERVER is READY");
System.out.println("The SERVER is WAITING to receive the CLIENT requests");

//run() is called by the main thread
//run() enables the ORB to perform work using the main thread
//the server waits until an invocation comes from the ORB

orb.run();

}

catch (Exception e)

{
```

```
System.err.println("ERROR: " + e);
e.printStackTrace(System.out);

}


//This statement is executed when the Client wishes to discontinue
System.out.println("The Server Exits");

}//end of main()

}//end of CalcServer()
```

6. Create another file C:\Calc\CalcClient.java with the following code in it:

```
//Import all the important packages

//Import the package which contains the Client Stub
import WssCalculator.*;

//Import the below two packages to use the Naming Service
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;

//Import this package to run the CORBA Applicaion
import org.omg.CORBA.*;

//Import to perform Input-Output functionalities
import java.io.*;
import java.util.*;



public class CalcClient

{

static Calc cimpl;

public static void main(String args[])

{

try
```

```
{

//Declaring and initializing the variables

int dec=1;
int i=0;
int j=0;
int k=0;
int result=0;
int x=1;
char c='x';
char d='y';
char f='z';
String abc="vas";


//Create and Initialize the ORB object
//init() allows to set properties at run time

ORB orb=ORB.init(args,null);

//ORB helps the Client to locate the actual services which it needs
//COS Naming Service helps the client to do so


//Obtain the initial Naming Context
//Obtain an object reference to the name server

org.omg.CORBA.Object objRef=orb.resolve_initial_references("NameService");


//Narrow the objref to its proper type

NamingContextExt ncRef=NamingContextExtHelper.narrow(objRef);


//Identify a String to refer the Naming Service to Calc object

String name="Calc";

//Get a reference to the CalcServer and Narrow it to Calc object

cimpl=CalcHelper.narrow(ncRef.resolve_str(name));
System.out.println("Obtained a handle on the server object");
```

```
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));


while(x==1)

{

System.out.println("Enter the string:");
abc=br.readLine();

//Separate the input string into separate characters

c=abc.charAt(0);
d=abc.charAt(1);
f=abc.charAt(2);

//Get the ASCII value of the Operator

i=(int)c;

//Get the Integer values of the other two characters

j=Character.getNumericValue(d);
k=Character.getNumericValue(f);

result=cimpl.calculate(i,j,k);
System.out.println("The result of the operation is "+result);

System.out.println("Enter 1 to continue and 0 to exit ");
x=Integer.parseInt(br.readLine());
}

//If the Client wants to discontinue
cimpl.shutdown();

}


catch(Exception e)

{

System.out.println("ERROR : " + e) ;
e.printStackTrace(System.out);

}
```

}//end of main()

}//end of class

7. Compile both the files by entering the following commands:

javac CalcServer.java WssCalculator/*.java

javac CalcClient.java WssCalculator/*.java

ignore the notes.

8. Now, to run the application: start orbd.exe in current directory on port 1050 by entering the following code:

start orbd -ORBInitialPort 1050 -ORBInitialHost localhost

This will cause one CMD to open with the name orbd.exe. Minimize it and ignore the firewall pop up (if any).

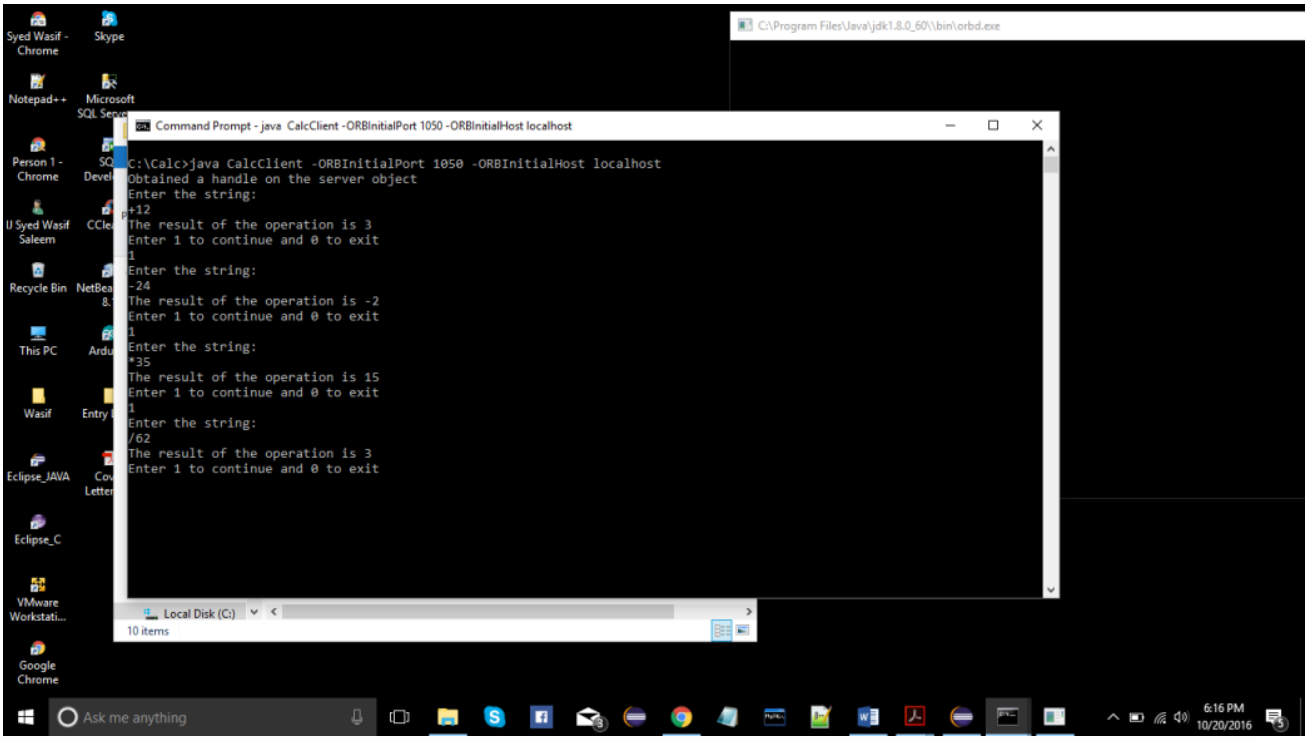
9. Next, run the CalcServer first by:

java CalcServer -ORBInitialPort 1050 -ORBInitialHost localhost

10. Run the CalcClient in another CMD window (change the directory to C:\Calc) using:

java CalcClient -ORBInitialPort 1050 -ORBInitialHost localhost

The display is:



11. To exit, close the CalcClient cmd window first. Then close the CalcServer cmd window. To close orbd.exe cmd window, do not use close button. Instead click Ctrl+C while cdm is on focus. This will close orbd.exe.

Assignment No.3

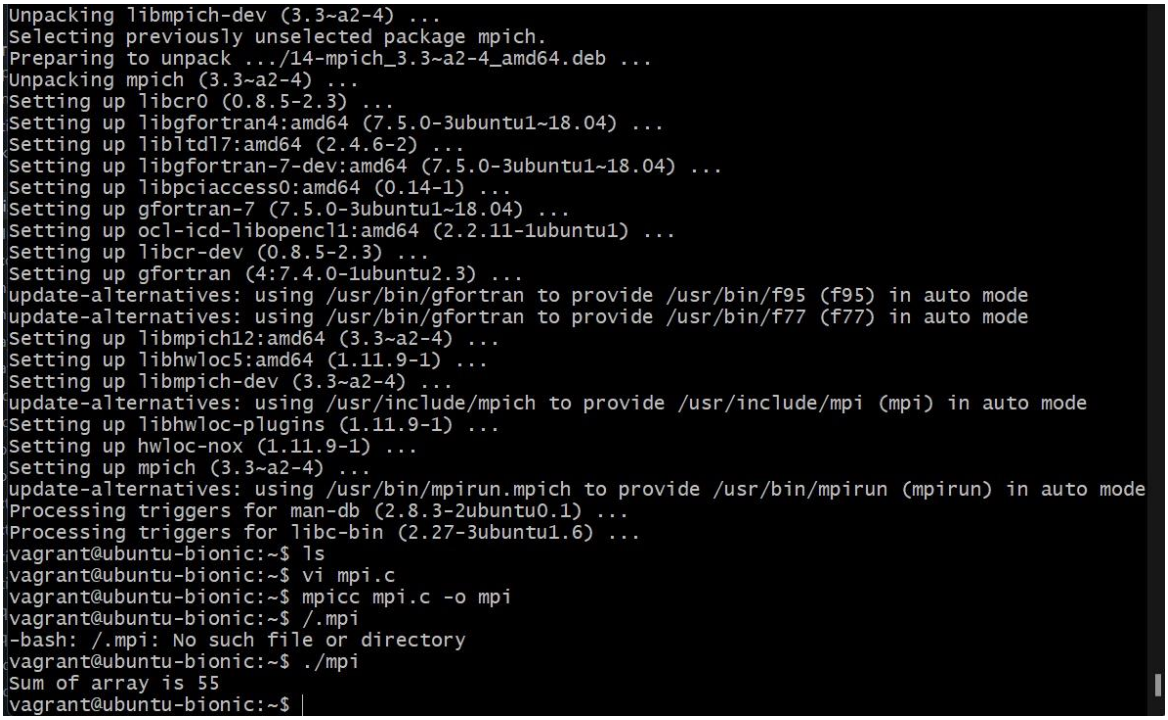
Develop a distributed system, to find sum of N elements in an array by distributing N/n elements to n number of processors MPI or OpenMP. Demonstrate by displaying the intermediate sums calculated at different processors.

```
#include <mpi.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

#define n 10

int a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
int a2[1000];

int main(int argc, char *argv[])
{

   int pid, np, element_per_process, n_element_received;
   MPI_Status status;
   MPI_Init(&argc, &argv);

   MPI_Comm_rank(MPI_COMM_WORLD, &pid);
   MPI_Comm_size(MPI_COMM_WORLD, &np);

   if (pid == 0)
```

```
{

    int index, i;
    element_per_process = (n / np);

    if (np > 1)
    {

        for (int i = 0; i < np - 1; i++)
        {

            index = i * element_per_process;

            MPI_Send(&element_per_process, 1, MPI_INT, i, 0, MPI_COMM_WORLD);
            MPI_Send(&a[index], 1, MPI_INT, i, 0, MPI_COMM_WORLD);
        }
        index = i * element_per_process;
        int element_left = n - index;

        MPI_Send(&element_left, 1, MPI_INT, i, 0, MPI_COMM_WORLD);
        MPI_Send(&a[index], element_left, MPI_INT, i, 0, MPI_COMM_WORLD);
    }

    int sum = 0;
    for (int i = 0; i < element_per_process; i++)
```

```
        {


            sum += a[i];

        }

        int temp;

        for (int i = 1; i < np; i++)


        {

            MPI_Recv(&temp, 1, MPI_INT, MPI_ANY_SOURCE, 0, MPI_COMM_WORLD,
&status);

            int sender = status.MPI_SOURCE;

            sum += temp;

        }

        printf("Sum of array is %d \n", sum);

    }


    else

    {


        MPI_Recv(&n_element_received, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, &status);

    }

    return 0;

}
```

Output:


Unpacking libmpich-dev (3.3~a2-4) ...
Selecting previously unselected package mpich.
Preparing to unpack .../14-mpich_3.3~a2-4_amd64.deb ...
Unpacking mpich (3.3~a2-4) ...
Setting up libcr0 (0.8.5-2.3) ...
Setting up libgfortran4:amd64 (7.5.0-3ubuntu1~18.04) ...
Setting up libltdl7:amd64 (2.4.6-2) ...
Setting up libgfortran-7-dev:amd64 (7.5.0-3ubuntu1~18.04) ...
Setting up libpciaccess0:amd64 (0.14-1) ...
Setting up gfortran-7 (7.5.0-3ubuntu1~18.04) ...
Setting up ocl-icd-libopencl1:amd64 (2.2.11-1ubuntu1) ...
Setting up libcr-dev (0.8.5-2.3) ...
Setting up gfortran (4:7.4.0-1ubuntu2.3) ...
update-alternatives: using /usr/bin/gfortran to provide /usr/bin/f95 (f95) in auto mode
update-alternatives: using /usr/bin/gfortran to provide /usr/bin/f77 (f77) in auto mode
Setting up libmpich12:amd64 (3.3~a2-4) ...
Setting up libhwloc5:amd64 (1.11.9-1) ...
Setting up libmpich-dev (3.3~a2-4) ...
update-alternatives: using /usr/include/mpich to provide /usr/include/mpi (mpi) in auto mode
Setting up libhwloc-plugins (1.11.9-1) ...
Setting up hwloc-nox (1.11.9-1) ...
Setting up mpich (3.3~a2-4) ...
update-alternatives: using /usr/bin/mpirun.mpich to provide /usr/bin/mpirun (mpirun) in auto mode
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for libc-bin (2.27-3ubuntu1.6) ...
vagrant@ubuntu-bionic:~$ ls
vagrant@ubuntu-bionic:~$ vi mpi.c
vagrant@ubuntu-bionic:~$ mpicc mpi.c -o mpi
vagrant@ubuntu-bionic:~$ /.mpi
-bash: /.mpi: No such file or directory
vagrant@ubuntu-bionic:~$ ./mpi
Sum of array is 55
vagrant@ubuntu-bionic:~$

Assignment No.4

Write a program to implement Berkeley Clock Synchronization.

```
import datetime
import socket
import threading
import time
from functools import reduce

from dateutil import parser

# data structure used to store client address and clock data
client_data = {}


def startReceivingClockTime(connector, address):
    while True:
        clock_time_string = connector.recv(1024).decode()
        clock_time = parser.parse(clock_time_string)
        clock_time_diff = datetime.datetime.now() - clock_time
        client_data[address] = {
            "clock_time": clock_time,
            "time_difference": clock_time_diff,
            "connector": connector
        }

        print("Client Data updated with: " + str(address), end="\n\n")
        time.sleep(5)


def startConnecting(master_server):
    while True:
        master_slave_connector, addr = master_server.accept()
        slave_address = str(addr[0]) + ":" + str(addr[1])
        print(slave_address + " got connected successfully")
        current_thread = threading.Thread(
            target=startReceivingClockTime, args=(master_slave_connector, slave_address))
        current_thread.start()


def synchronizeAllClocks():
    while True:
        print("New synchronization cycle started.")
```

```
        print("Number of clients to be synchronized: " + str(len(client_data)))
        if len(client_data) > 0:
            average_clock_difference = getAverageClockDiff()
            for client_addr, client in client_data.items():
                try:
                    synchronized_time = datetime.datetime.now() + average_clock_difference
                    client['connector'].send(str(synchronized_time).encode())
                except Exception as e:
                    print(
                        "Something went wrong while sending synchronized time through " +
str(client_addr))
        else:
            print("No client data. Synchronization not applicable.")
        print("\n\n")
        time.sleep(5)


def getAverageClockDiff():
    current_client_data = client_data.copy()
    time_difference_list = [client['time_difference']
                            for client in current_client_data.values()]
    sum_of_clock_difference = reduce(
        lambda x, y: x + y, time_difference_list, datetime.timedelta(0))
    average_clock_difference = sum_of_clock_difference / len(client_data)
    return average_clock_difference


def initiateClockServer(port=8080):
    global master_server
    master_server = socket.socket()
    master_server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    print("Socket at master node created successfully\n")
    master_server.bind(('', port))
    master_server.listen(10)
    print("Clock server started...\n")
    print("Starting to make connections...\n")
    master_thread = threading.Thread(
        target=startConnecting, args=(master_server,))
    master_thread.start()
    print("Starting synchronization parallelly...\n")
    sync_thread = threading.Thread(target=synchronizeAllClocks)
    sync_thread.start()


if __name__ == '__main__':
```

initiateClockServer(port=8080)

Output:

```
New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.
```

Assignment No.5

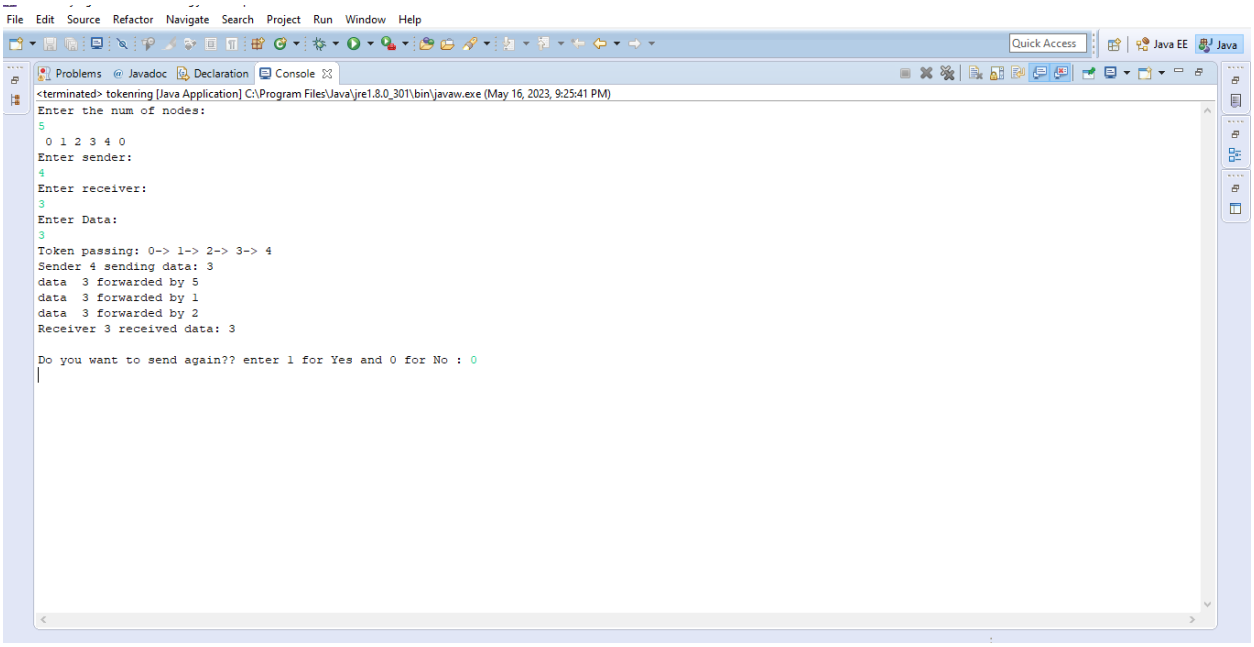
Implement token ring based mutual exclusion algorithm

```
import java.io.*;
import java.util.*;
public class tokenring
{
        public static void main(String args[]) throws Throwable {
      Scanner scan = new Scanner(System.in);
      System.out.println("Enter the num of nodes:");
      int n = scan.nextInt();
      int m = n - 1;
      // Decides the number of nodes forming the ring
      int token = 0;
      int ch = 0, flag = 0;
      for (int i = 0; i < n; i++) {
         System.out.print(" " + i);
      }
      System.out.println(" " + 0);
      do{
         System.out.println("Enter sender:");
         int s = scan.nextInt();
         System.out.println("Enter receiver:");
```

```
int r = scan.nextInt();
System.out.println("Enter Data:");
int a;
a = scan.nextInt();
System.out.print("Token passing:");
for (int i = token, j = token; (i % n) != s; i++, j = (j + 1) % n) {
   System.out.print(" " + j + "->");
}
System.out.println(" " + s);
System.out.println("Sender " + s + " sending data: " + a);
for (int i = s + 1; i != r; i = (i + 1) % n) {
   System.out.println("data  " + a + " forwarded by " + i);
}
System.out.println("Receiver " + r + " received data: " + a +"\n");
token = s;
do{
   try {
      if( flag == 1)
                  System.out.print("Invalid Input!!...");
      System.out.print("Do you want to send again?? enter 1 for Yes and 0 for No : ");
      ch = scan.nextInt();
      if( ch != 1 && ch != 0 )
                  flag = 1;
      else
                  flag = 0;
```

```
            } catch (InputMismatchException e){
                System.out.println("Invalid Input");
            }
        }while( ch != 1 && ch != 0 );
    }while( ch == 1 );
  }
}
```

Output:



Assignment No.6

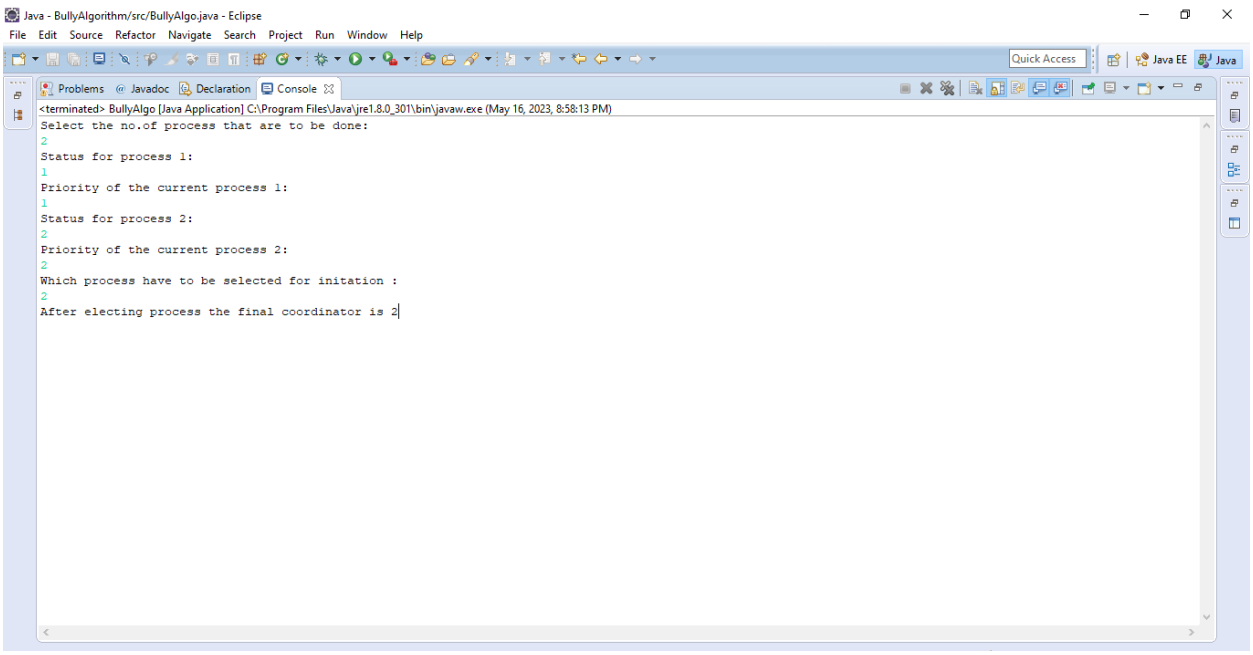
Implement Bully and Ring algorithm for leader election.

BullyAlgo.java

```
import java.io.*;
import java.util.Scanner;
public class BullyAlgo
{
	static int noofprocess;
	static int priority[] = new int[100];
	static int S[] = new int[100];
	static int C;

	   // main() method of the class
	public static void main(String args[]) throws IOException
	   {
	   // Taking input for the number of processes from the user
	System. out.println("Select the no.of process that are to be done:");

		// Object creation for scanner class
		Scanner sc = new Scanner(System.in);
		noofprocess = sc.nextInt();

	int i;

		// Setting priorities for the process in the for loop
	for(i = 0; i<noofprocess; i++)
		{
	System.out.println("Status for process "+(i+1)+":");
	S[i] = sc.nextInt();
	System.out.println("Priority of the current process "+(i+1)+":");
	priority[i] = sc.nextInt();
		}

	System.out.println("Which process have to be selected for initation :");
	int e = sc.nextInt();

	sc.close();

		// call electProcess() method
	electProcess(e);
```

```
System.out.println("After electing process the final coordinator is "+C);
    }

    // create electProcess() method
static void electProcess(int e)
    {
e = e - 1;
C = e + 1;

for(int i = 0; i<noofprocess; i++)
        {
    if(priority[e]<priority[i])
            {
System.out.println("Election message is sent from "+(e+1)+" to "+(i+1));
if(S[i]==1)
    electProcess(i+1);
            }
        }
    }
}
```

Output:



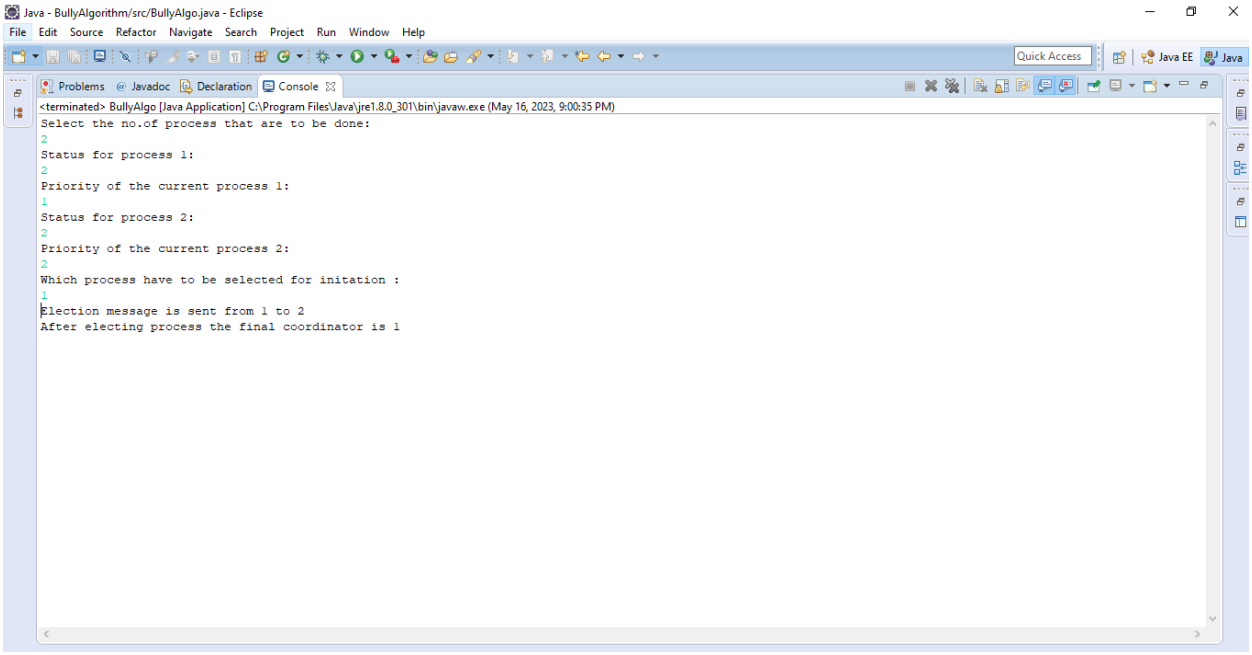
Ring.java

```
import java.util.Scanner;
class Process{
public int id;
public boolean active;

public Process(int id){
this.id=id;
active=true;
}

}
public class Ring
{
	int noOfProcesses;
	Process[] processes;
	Scanner sc;

	public Ring(){
	sc=new Scanner(System.in);
	}
	public void initialiseRing(){
	System.out.println("Enter no of processes");
	noOfProcesses=sc.nextInt();
	processes = new Process[noOfProcesses];
	for(int i=0;i<processes.length;i++){
	processes[i]= new Process(i);
	}
	}

	public int getMax(){
	int maxId=-99;
	int maxIdIndex=0;
	for(int i=0;i<processes.length;i++){
	if(processes[i].active && processes[i].id>maxId){
	maxId=processes[i].id;
	maxIdIndex=i;

	}
	}
	return maxIdIndex;
	}
	public void performElection(){
```

```
        System.out.println("Process no "+processes[getMax()].id+" fails");
        processes[getMax()].active=false;
        System.out.println("Election Initiated by");
        int initiatorProcesss=sc.nextInt();

        int prev = initiatorProcesss;
        int next = prev+1;

        while(true){
        if(processes[next].active){
        System.out.println("Process "+processes[prev].id+" Pass
Election("+processes[prev].id+") to "+processes[next].id);
        prev=next;
        }

        next = (next+1)%noOfProcesses;
        if(next == initiatorProcesss){
        break;
        }
        }

        System.out.println("Process "+ processes[getMax()].id +" becomes coordinator");
        int coordinator = processes[getMax()].id;

        prev = coordinator;
        next =(prev+1)%noOfProcesses;

        while(true){

        if(processes[next].active)
        {
        System.out.println("Process "+ processes[prev].id +" pass Coordinator("+coordinator+ ")
message to process "+processes[next].id );
        prev = next;
        }
        next = (next+1) % noOfProcesses;
        if(next == coordinator)
        {
        System.out.println("End Of Election");
        break;
        }
        }

        }

        public static void main(String arg[]){
```

```
Ring r= new Ring();
r.initialiseRing();
r.performElection();
}

}
```

Output:



Assignment No.7

Create a simple web service and write any distributed application to consume the web service.

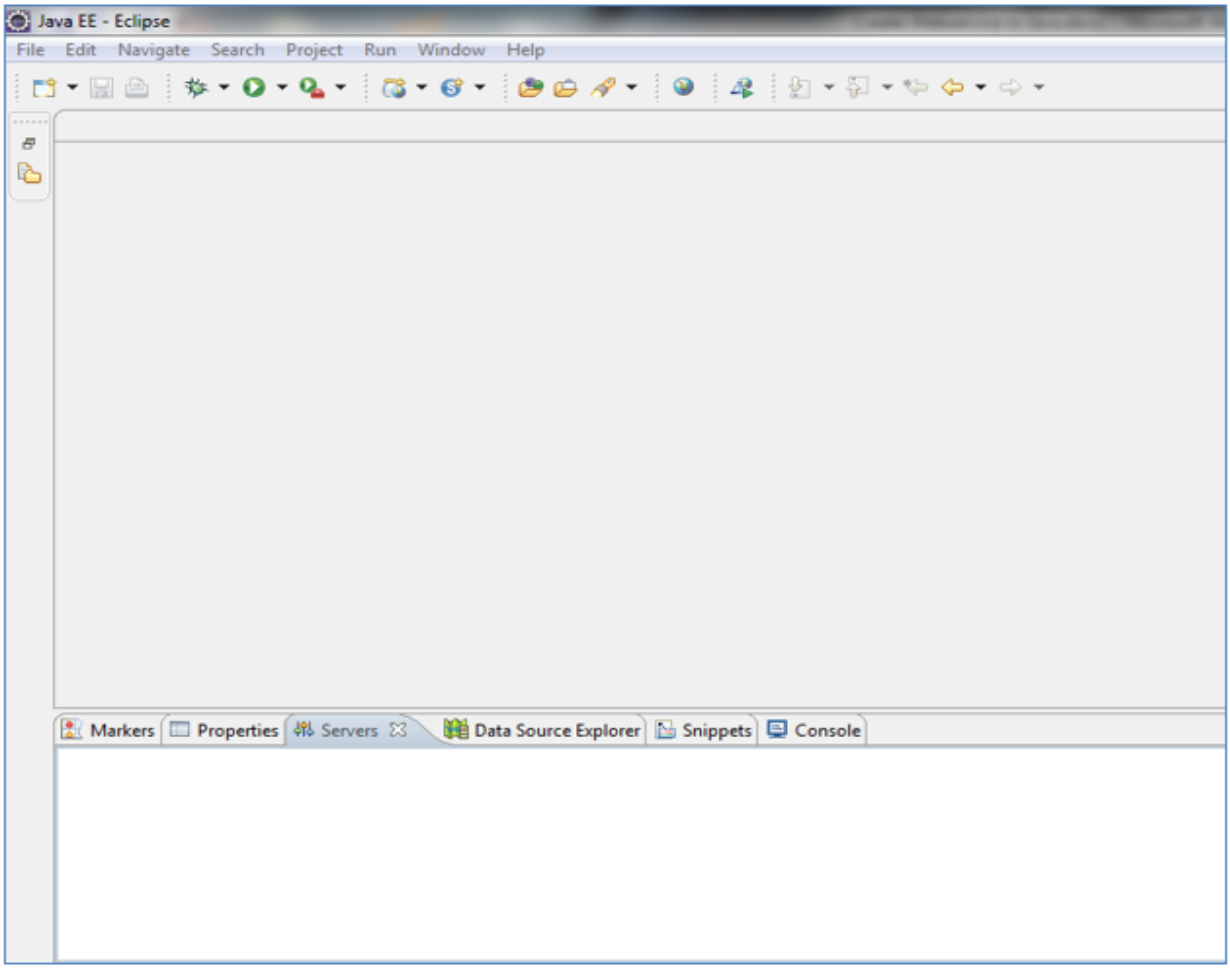
Code:

```
package com.tutorial.ws;

public class WebAdder
{
	public int addition (int nm1, int nm2)
	{
		return nm1+nm2;
	}
}
```

How to Create Webservice in Java?

Here we will use Eclipse IDE for this. You just need to follow step by step as mentioned below with screenshots to create a simple web service in java.

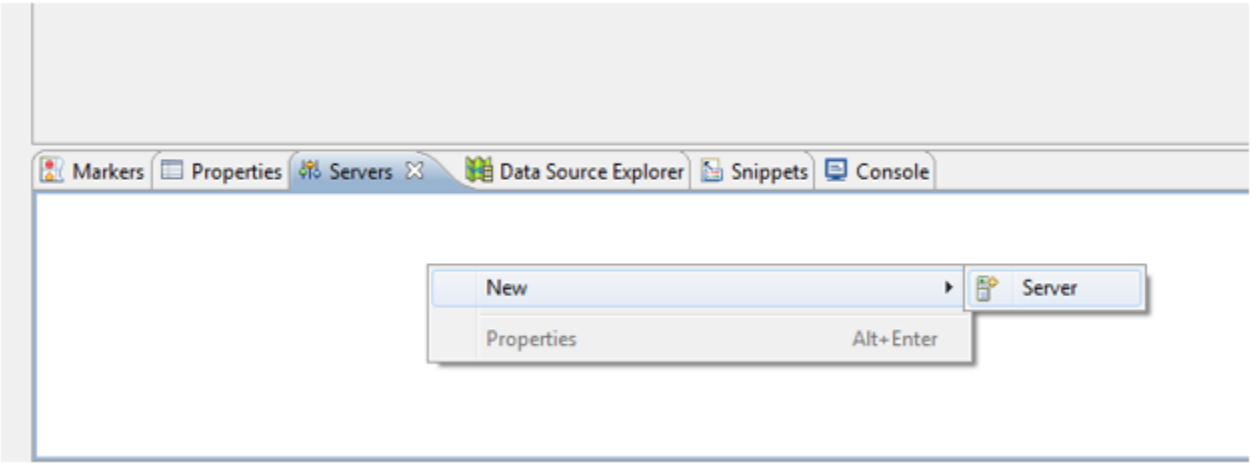
Step 1: Open eclipse >

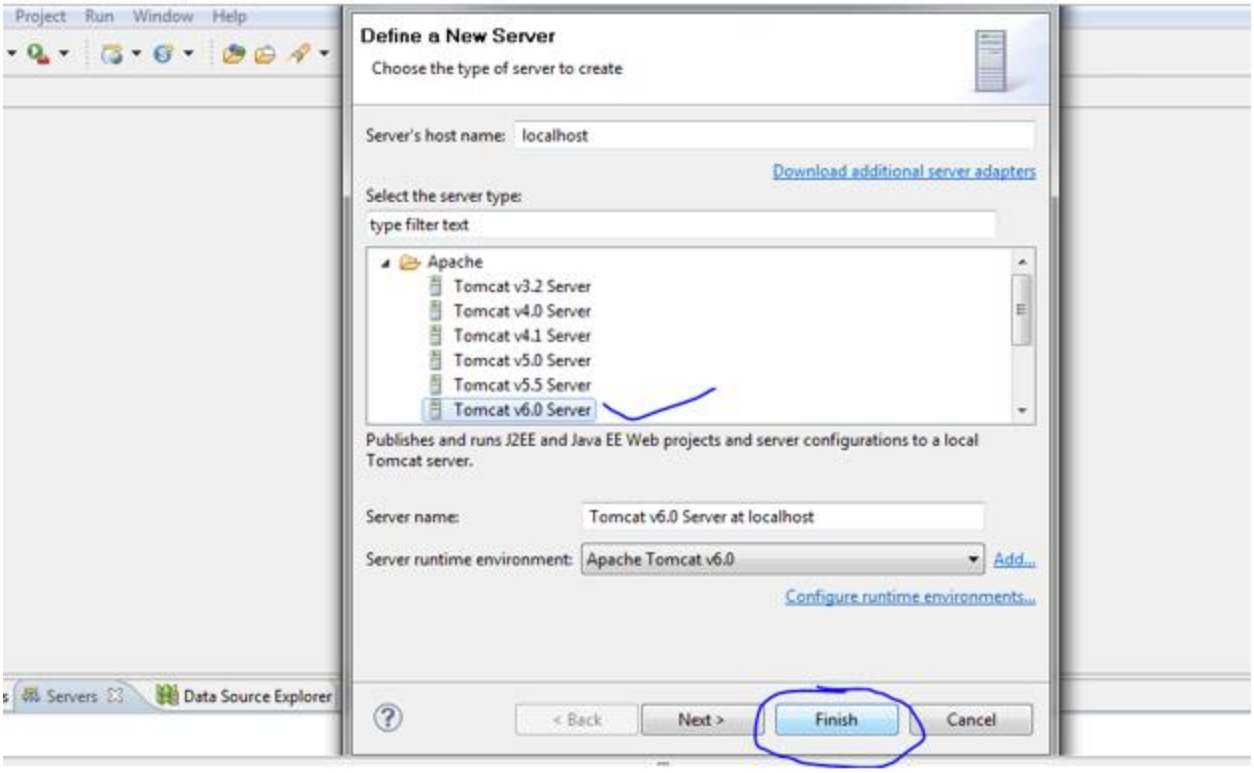


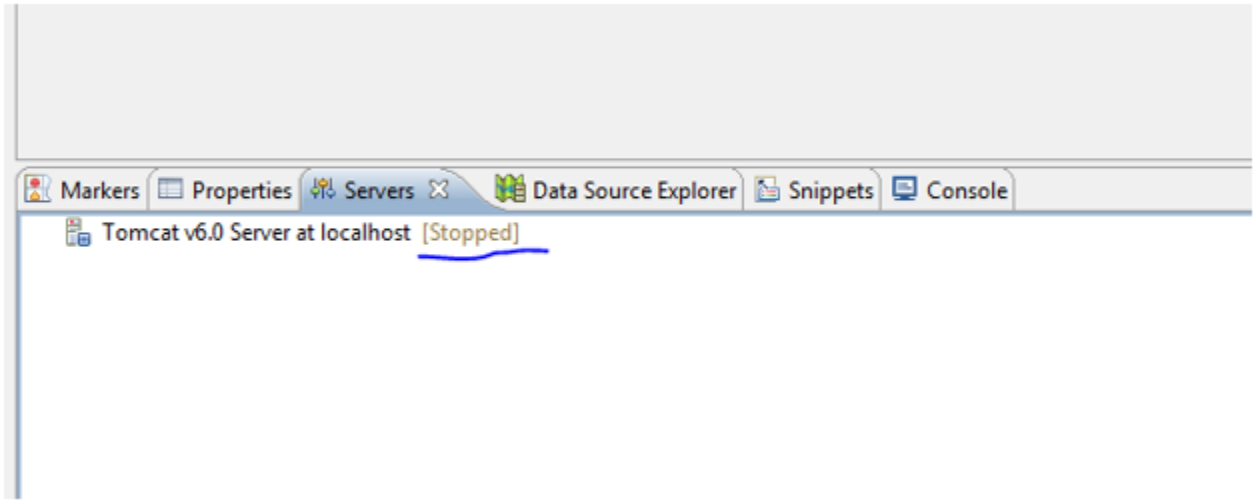
On the server tab, add one server (here I am using tomcat) in which you will run your web services.

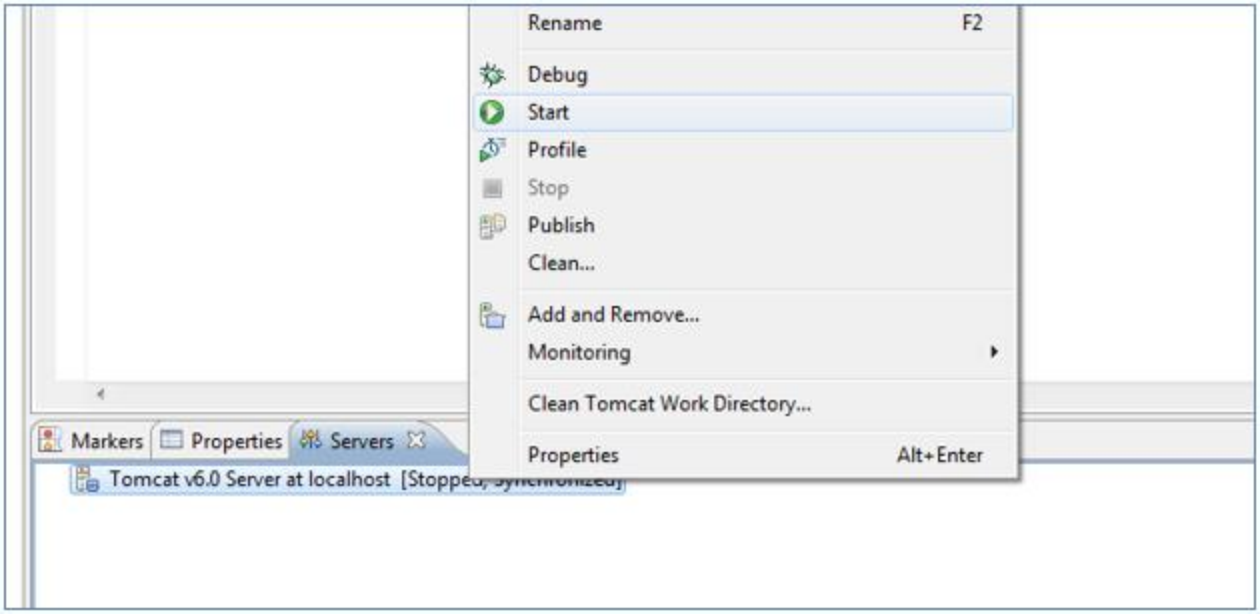
Step 2: Right-click on server tab> New> Server> Select Apache tomcat 6> Finish.

After that, you can see the server is created in the “Stopped” state; we will start the servers before running our application.



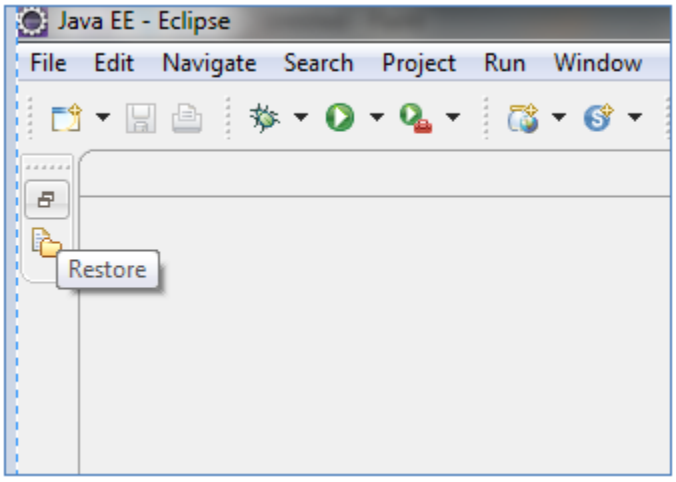

Project Run Window Help
Define a New Server
Choose the type of server to create
Server's host name: localhost
Download additional server adapters
Select the server type:
type filter text
Apache
Tomcat v3.2 Server
Tomcat v4.0 Server
Tomcat v4.1 Server
Tomcat v5.0 Server
Tomcat v5.5 Server
Tomcat v6.0 Server
Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server.
Server name: Tomcat v6.0 Server at localhost
Server runtime environment: Apache Tomcat v6.0
Add...
Configure runtime environments...
Servers
Data Source Explorer
< Back
Next >
Finish
Cancel

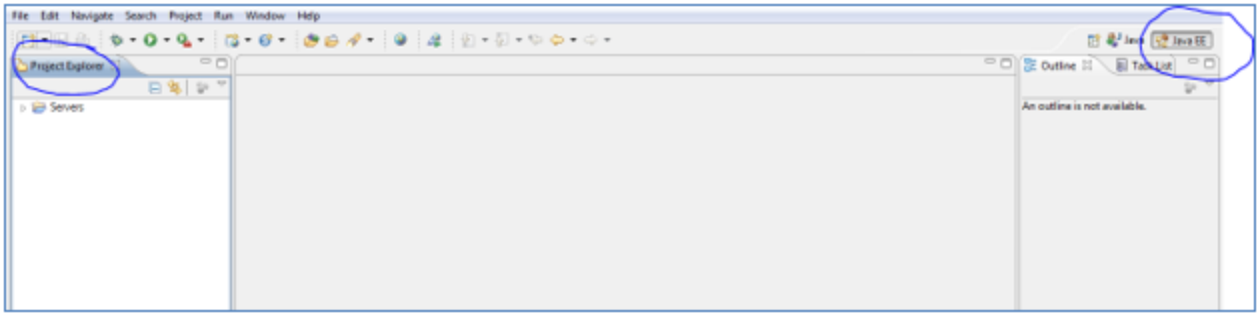

Markers
Properties
Servers
Data Source Explorer
Snippets
Console
Tomcat v6.0 Server at localhost [Stopped]



Now we will start our server.

Step 3: Right-click on tomcat > start



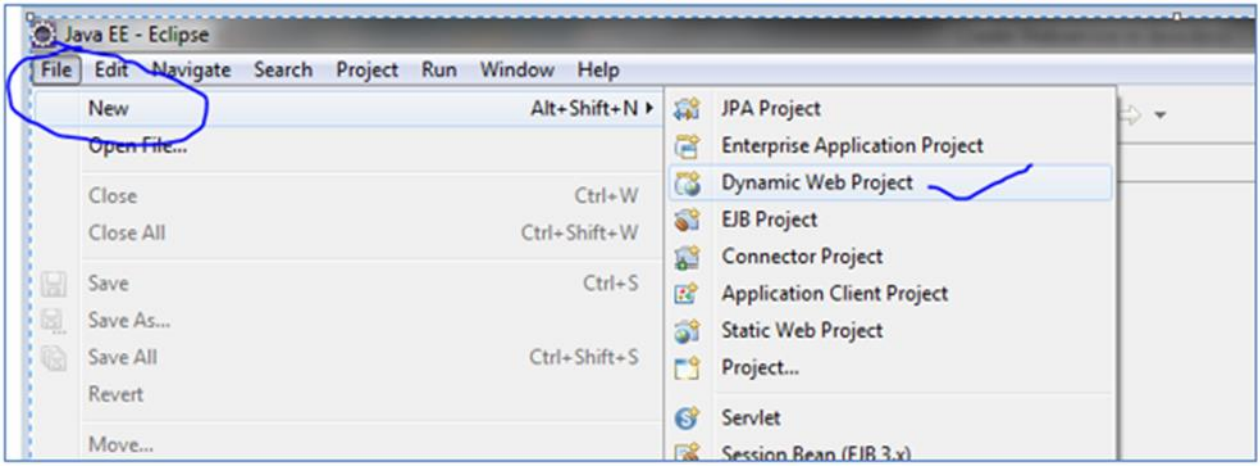


Now we will open project explorer for java EE.

Step 4: Click on restore > you will see project explorer

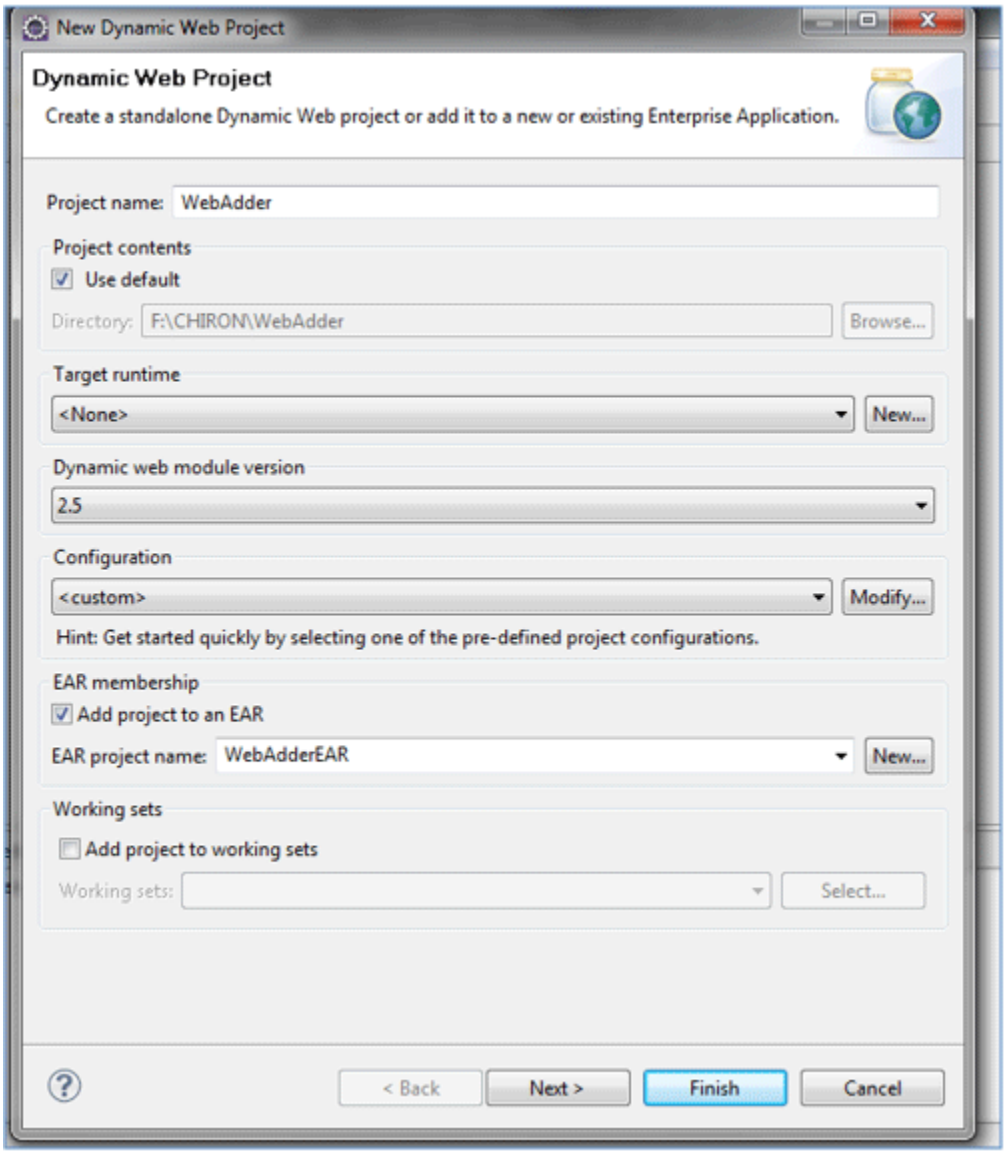
Now we will create our web service.

Step 5: File>New>Dynamic Web Project

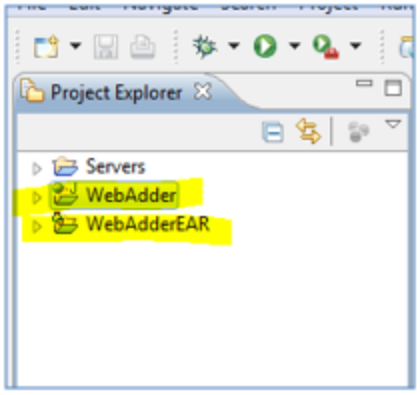


Could you give it a name (here, WebAdder)? Follow the below picture and do exactly the same as what is in there.

Step 6: Click on next> Finish

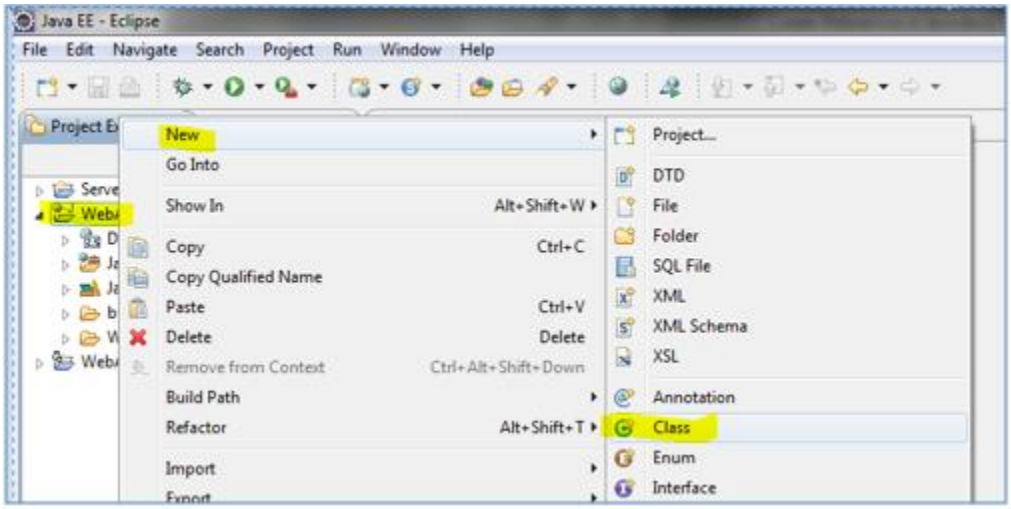


You will see your project is added as below

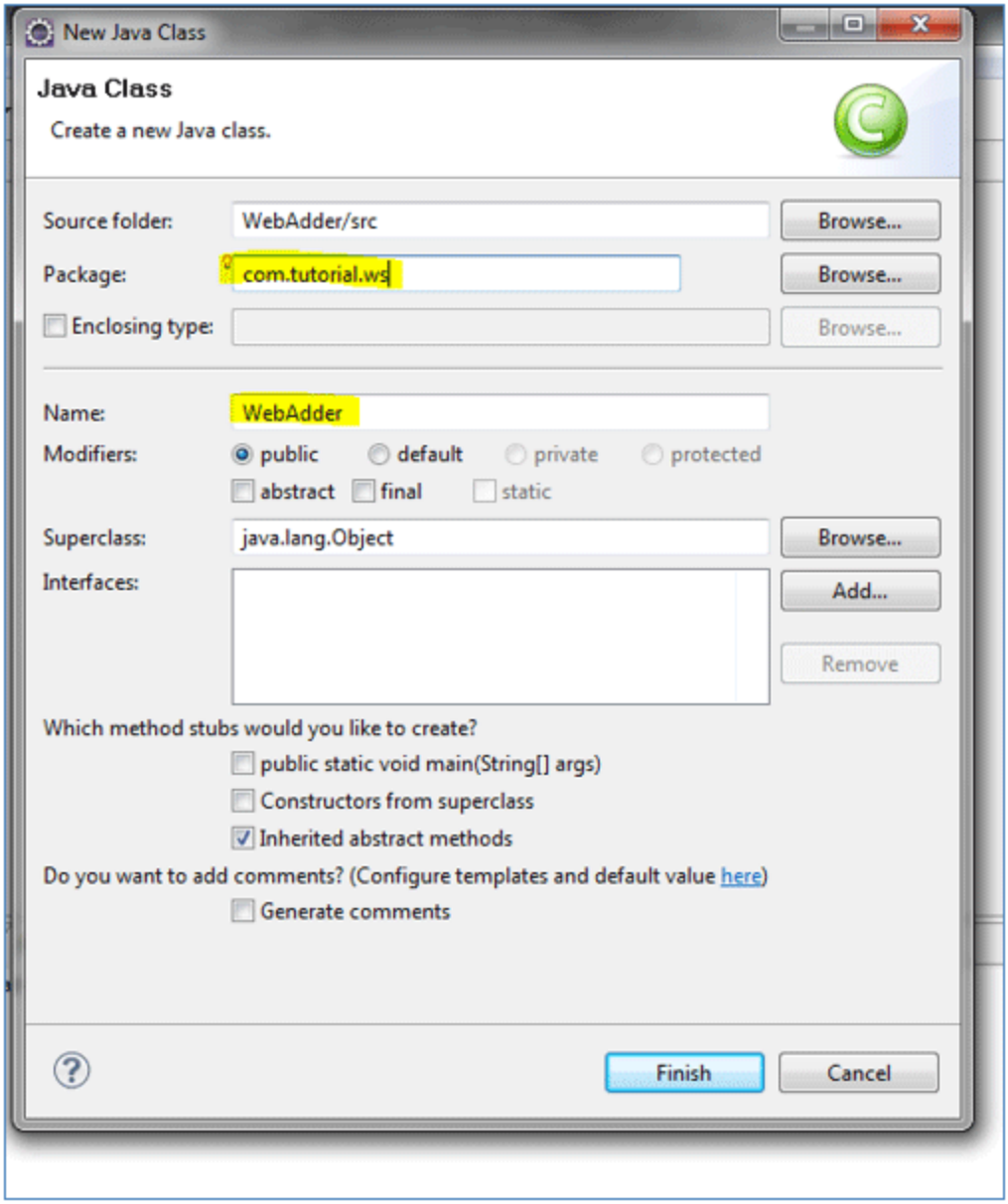


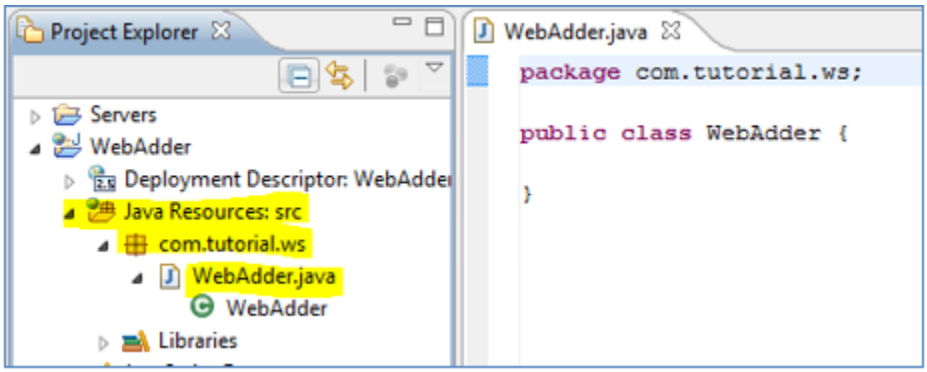
Now create a class.

Step 7: Right-click on “WebAdder”> New> Class



Step 8: Give it a name along with package name as per below picture > FInish.


New Java Class
Java Class
Create a new Java class.
Source folder:
WebAdder/src
Browse...
Package:
com.tutorial.ws
Browse...
Enclosing type:
Browse...
Name:
WebAdder
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
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? (Configure templates and default value here)
Generate comments
Finish
Cancel


Project Explorer
Servers
WebAdder
Deployment Descriptor: WebAdder
Java Resources: src
com.tutorial.ws
WebAdder.java
WebAdder
Libraries
WebAdder.java
package com.tutorial.ws;
public class WebAdder {
}

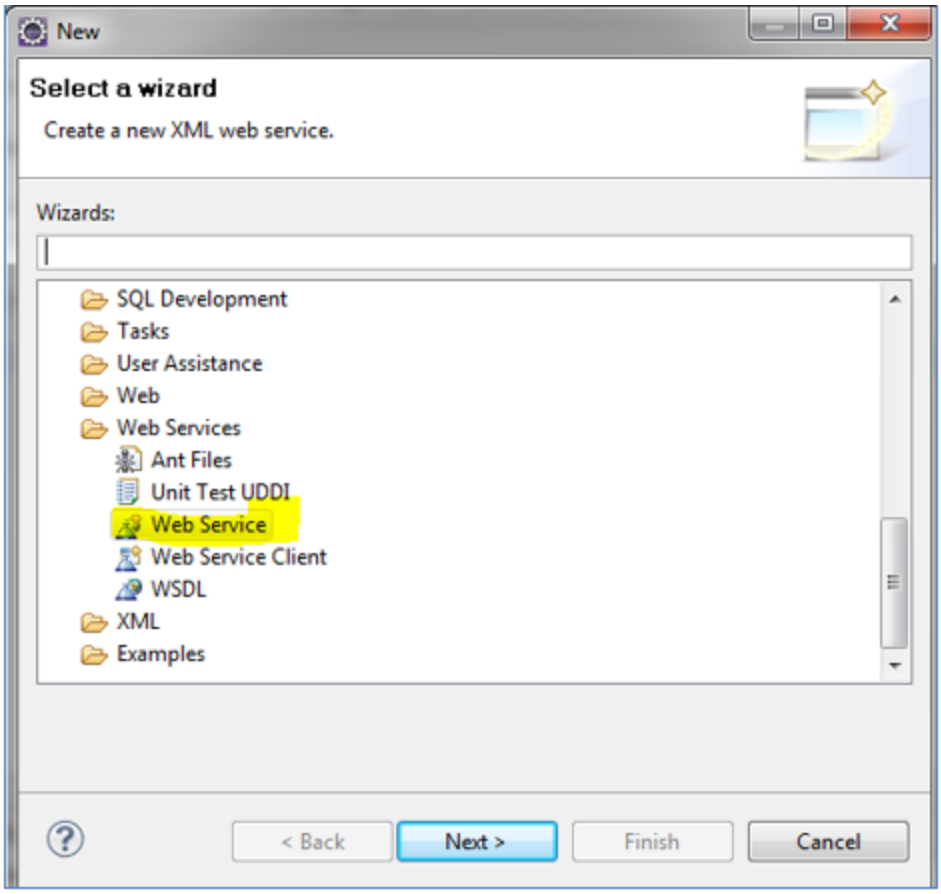
This class will be acting as a web service, so whatever methods we will write inside this class will act as web service methods.

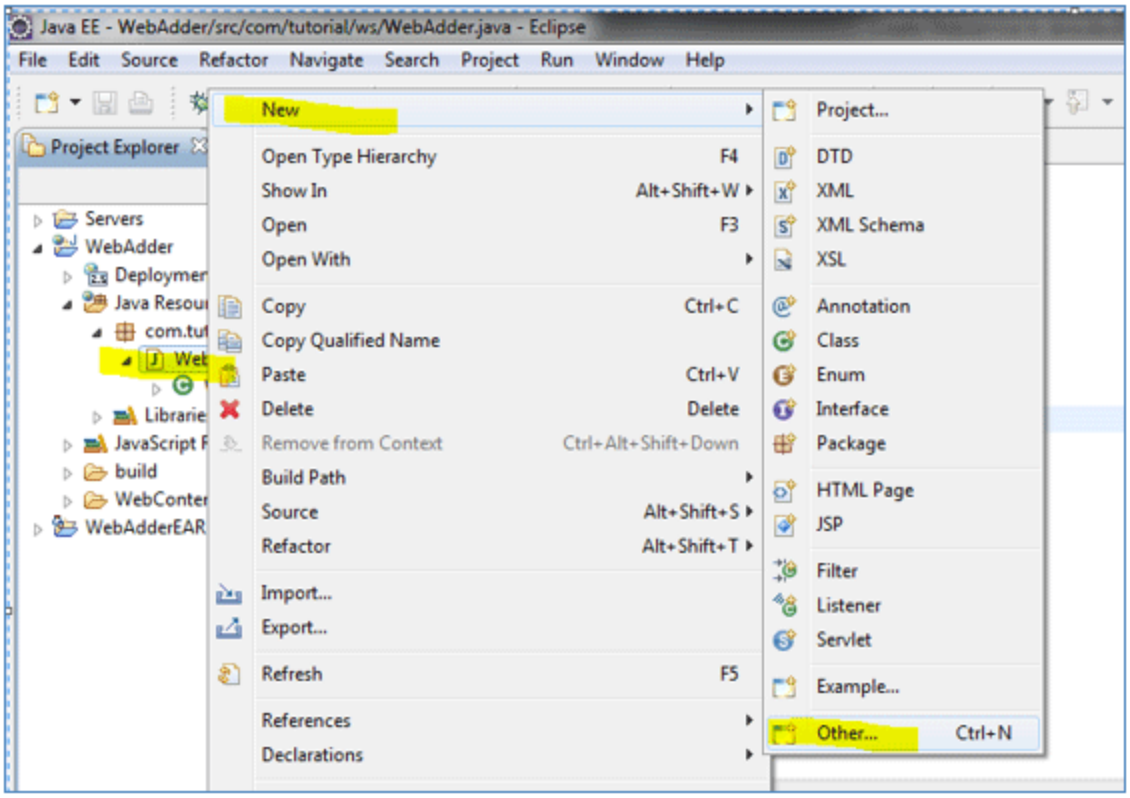
Step 9: Below is the code:

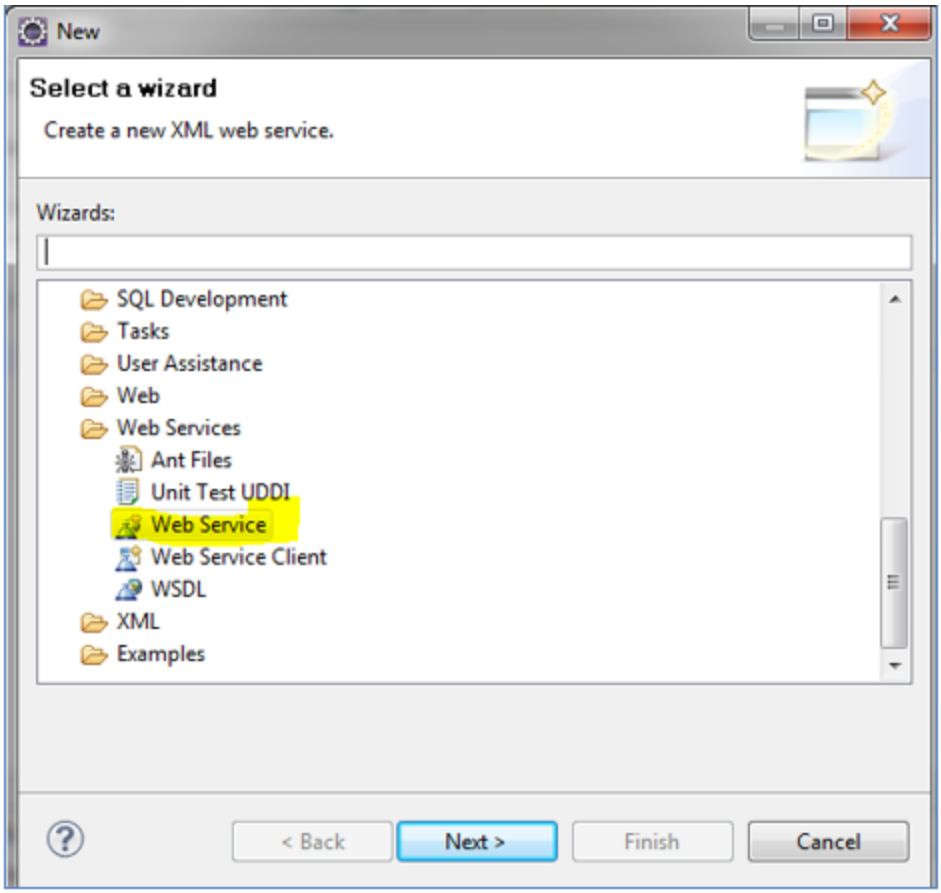
```
package com.tutorial.ws;
public class WebAdder {
public int addition (int nm1, int nm2){
return nm1+nm2;
}
// above method just doing addition
}
```

Now we want to make the class and method as a web service; for this we need to do the following:

Step 10: Right-click on WebAdder class> New>Other> Webservice>Next

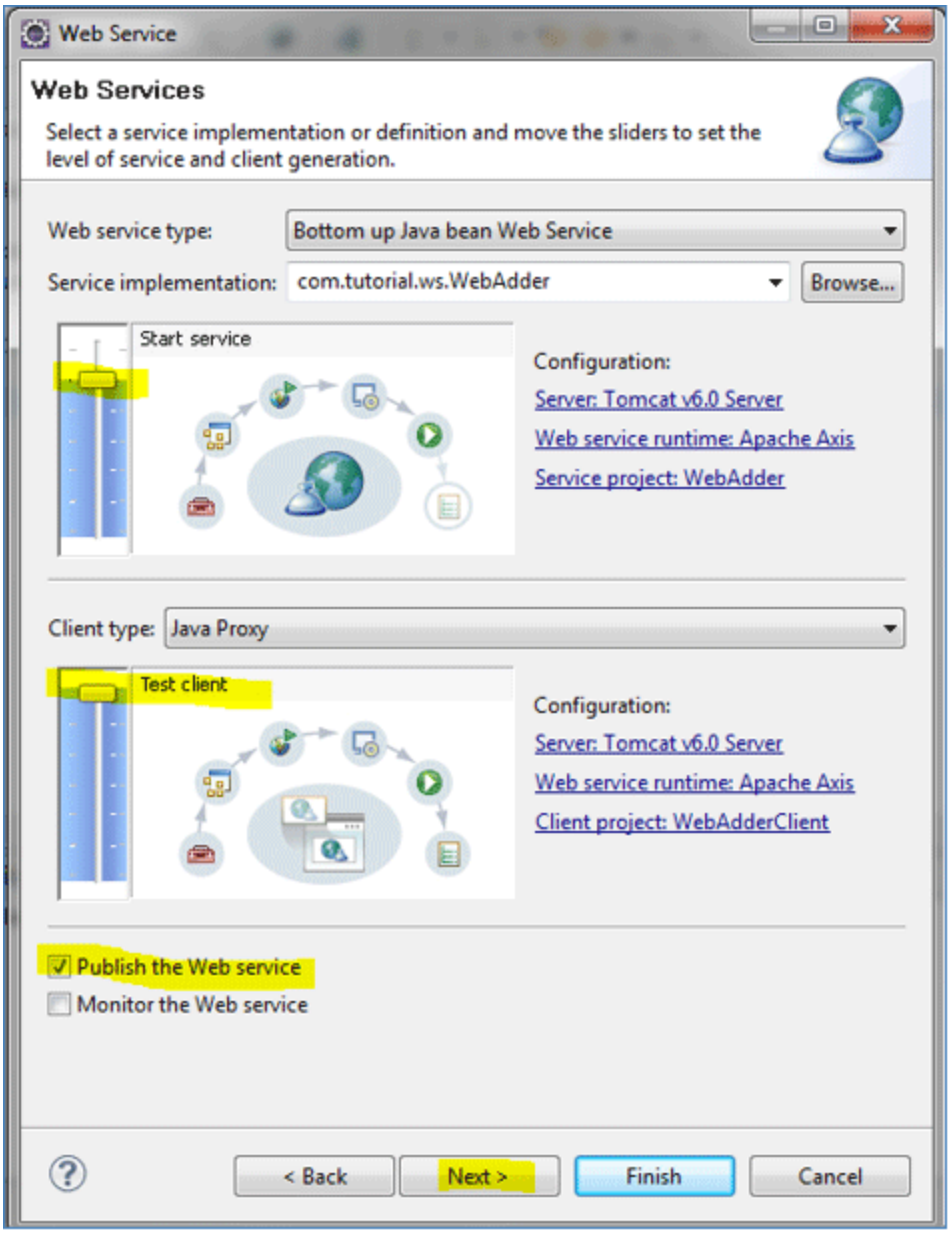

New
Select a wizard
Create a new XML web service.
Wizards:
SQL Development
Tasks
User Assistance
Web
Web Services
Ant Files
Unit Test UDDI
Web Service
Web Service Client
WSDL
XML
Examples
< Back
Next >
Finish
Cancel


Java EE - WebAdder/src/com/tutorial/ws/WebAdder.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Servers
WebAdder
New
Open Type Hierarchy F4
Show In Alt+Shift+W
Open F3
Open With
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
References
Declarations
Project...
DTD
XML
XML Schema
XSL
Annotation
Class
Enum
Interface
Package
HTML Page
JSP
Filter
Listener
Servlet
Example...
Other... Ctrl+N



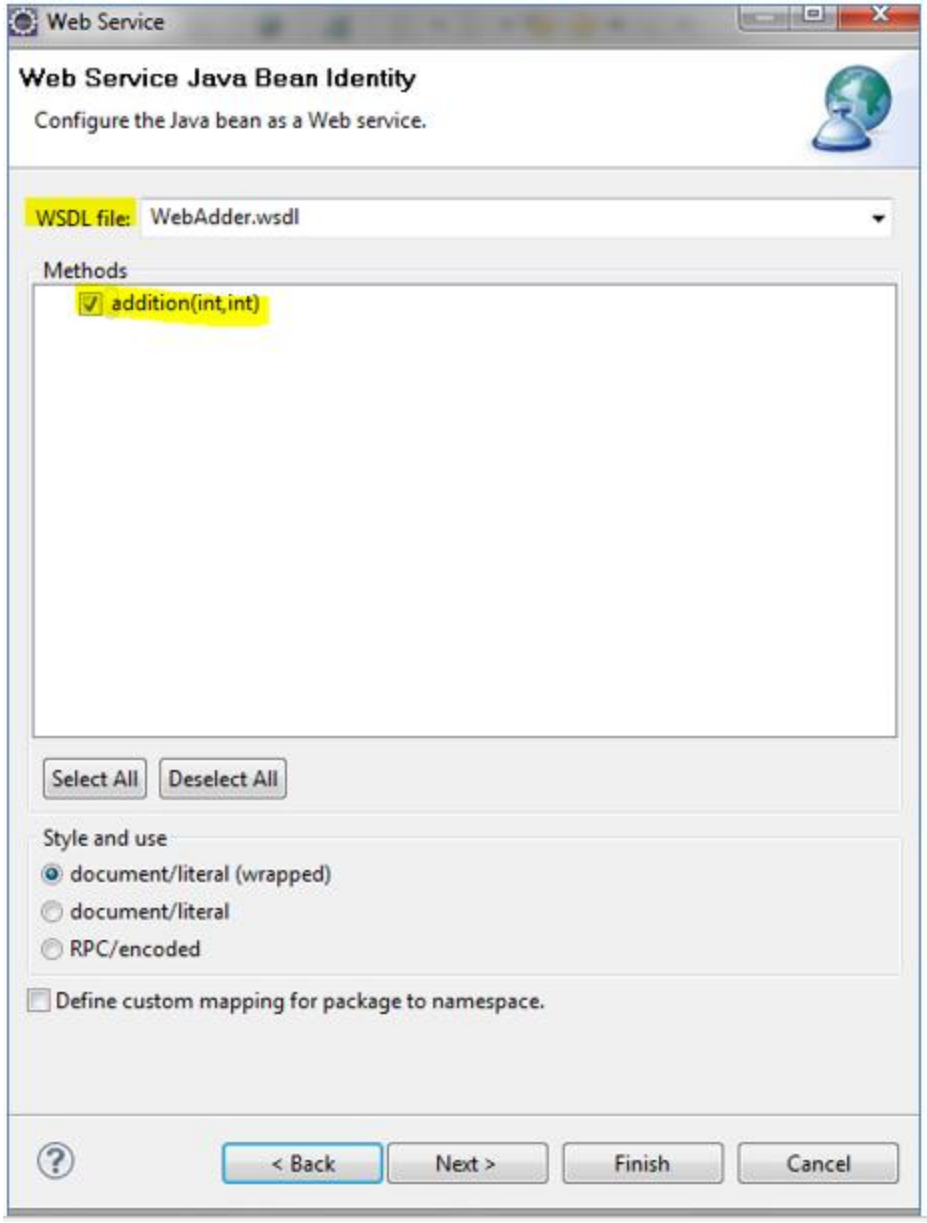
Here in the below screenshot, please pay attention to the highlighted portions; we are here creating clients also to test our web service. Follow the exact things as mentioned in the highlighted portion of the below screenshot and click on “Next.”

Step 11: Click on Next> “Finish.”

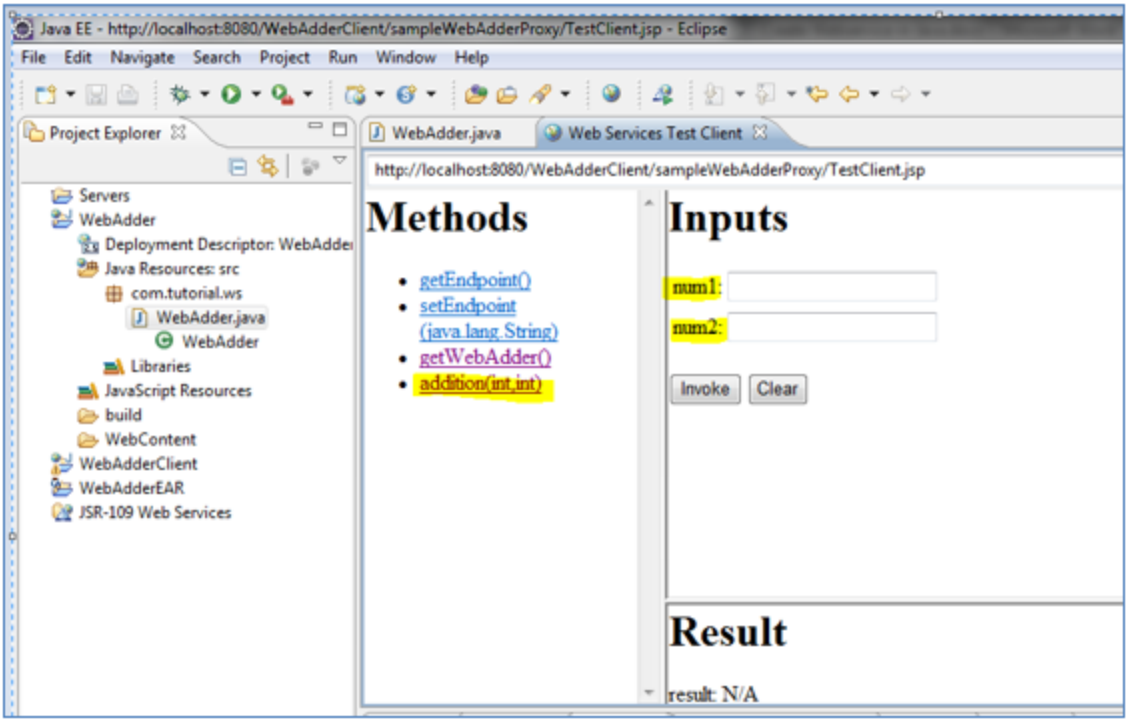


On the next page, as in the below screenshot, you can see that, addition () method becomes a web service method that will go in the WSDL document.

Step 12: Now you can see, your client is running on the server, where you need to click on your web method called “addition”,

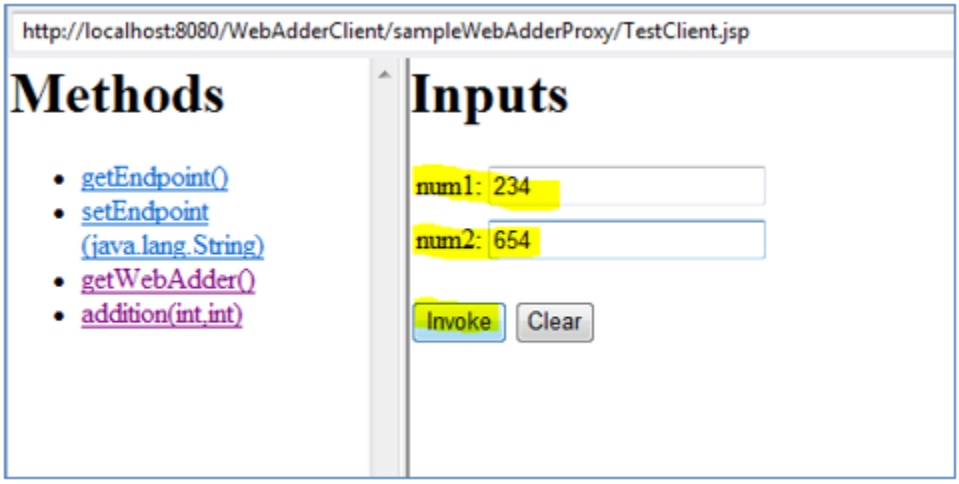


Step 13: Then, you will see that two fields will come, num1 and num2, to provide user input. Refer to the below screenshot:

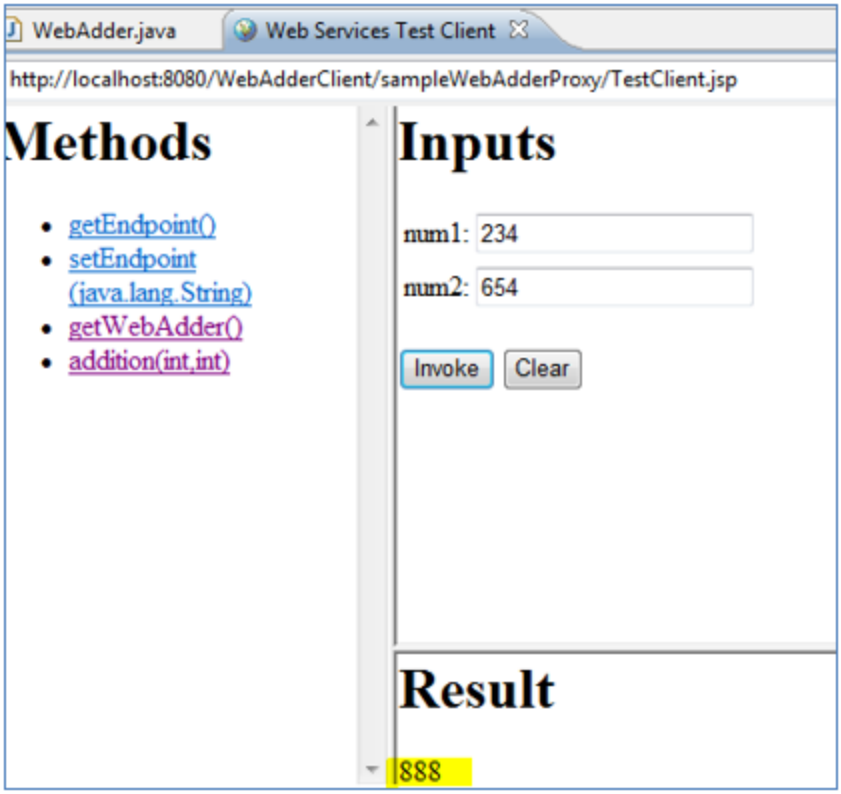


tep

14: Give some random integer values to get the result.



Step 15: After clicking on the "invoke" button, you will see the result. Congrats, you have completed building your first web service in java!



Program link:

https://www.theserverside.com/video/Step-by-step-SOAP-web-services-example-in-Java-using-Eclipse.

Output:

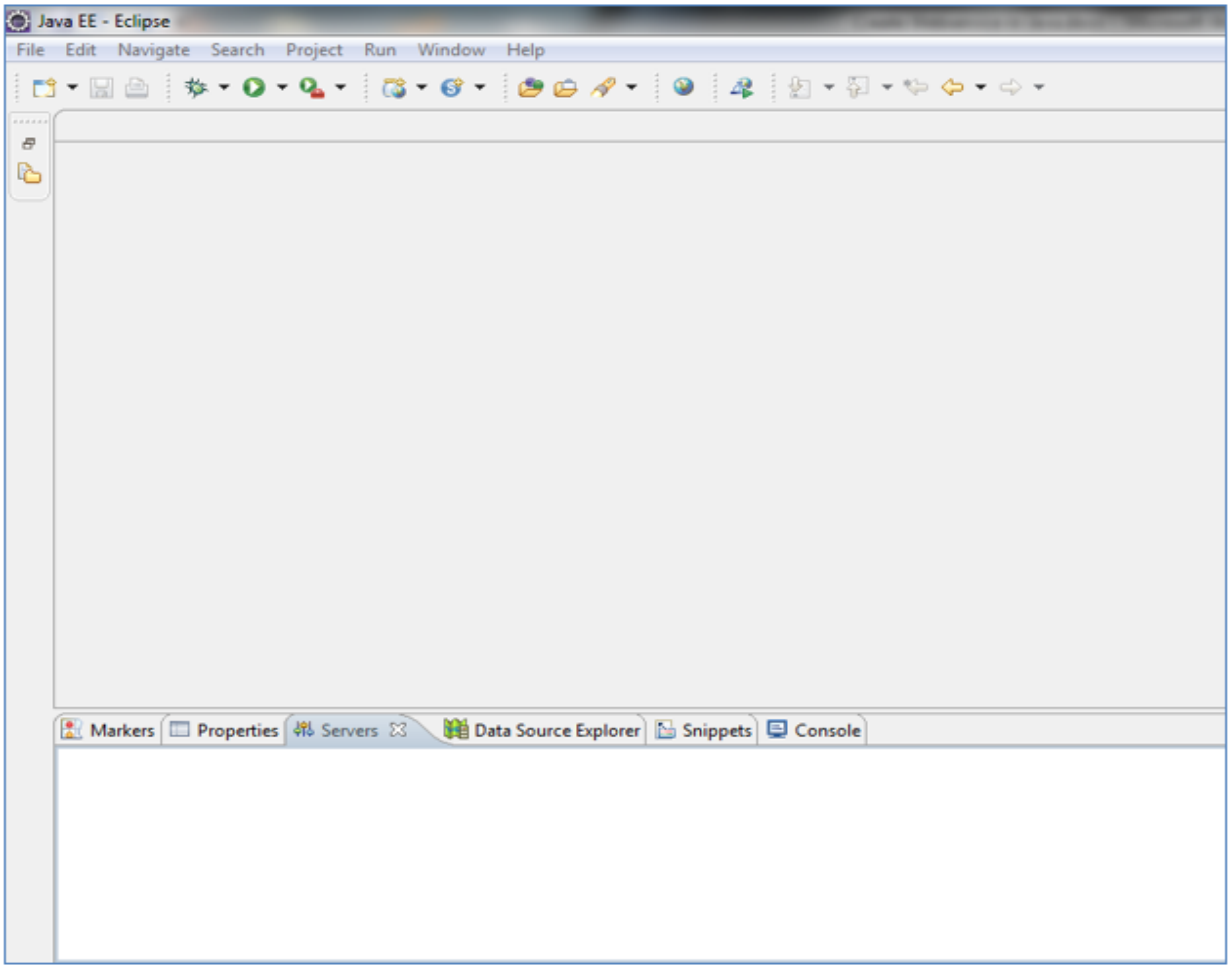
Practical no.7

How to Create Webservice in Java?

Here we will use Eclipse IDE for this. You just need to follow step by step as mentioned below

with screenshots to create a simple web service in java.

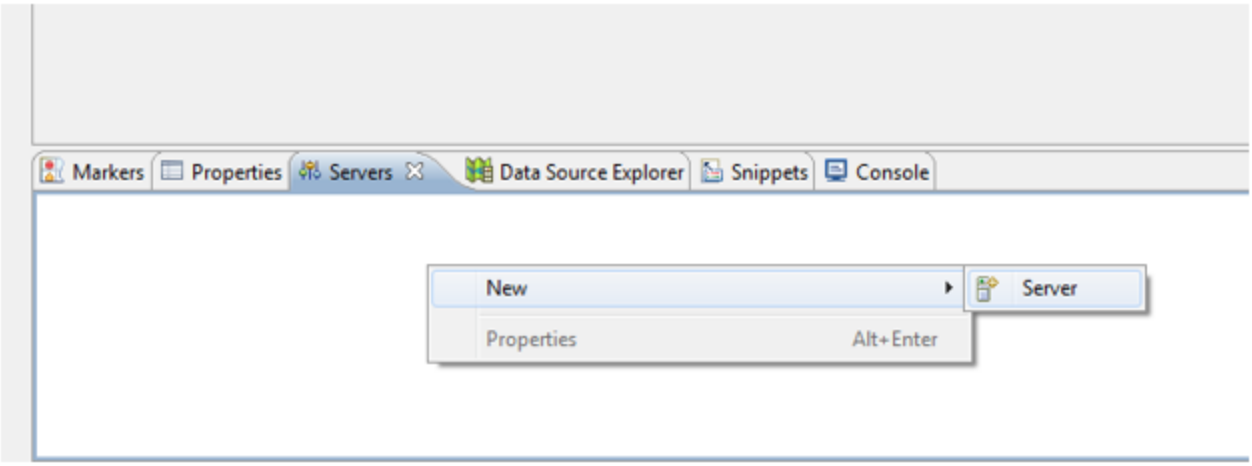
Step 1: Open eclipse >

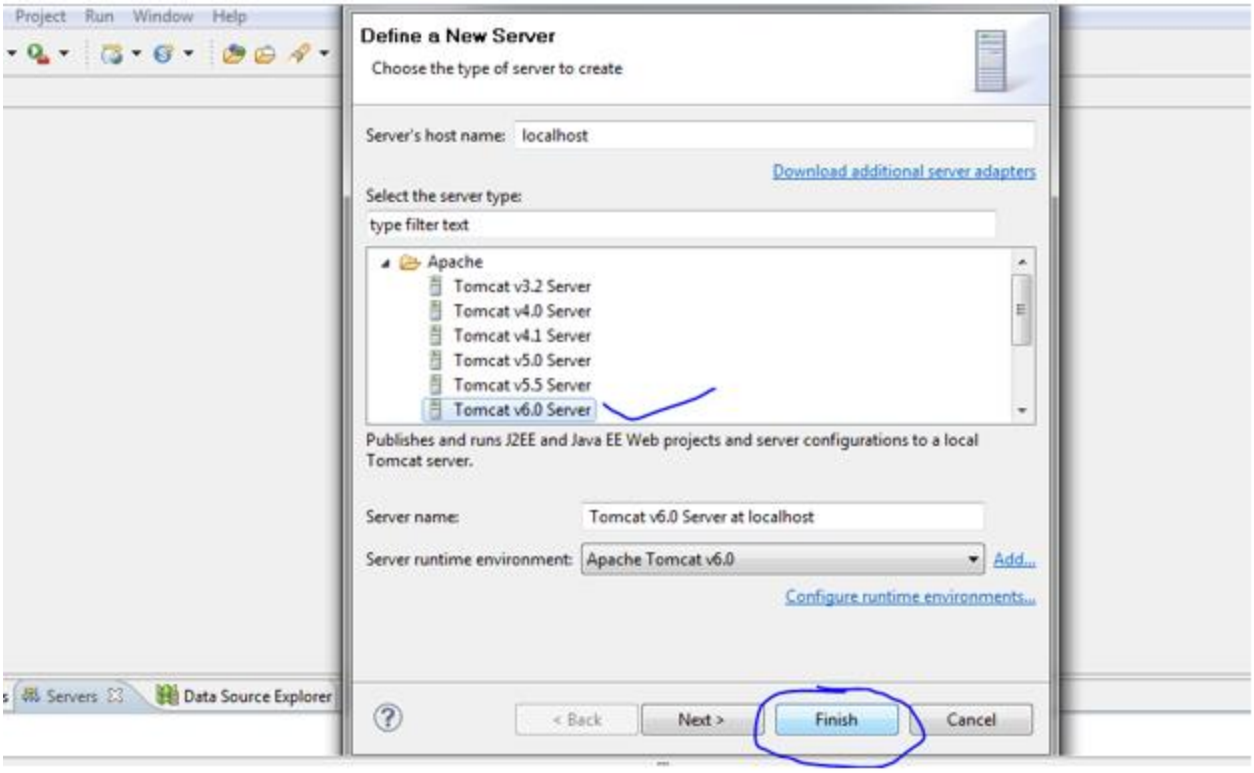


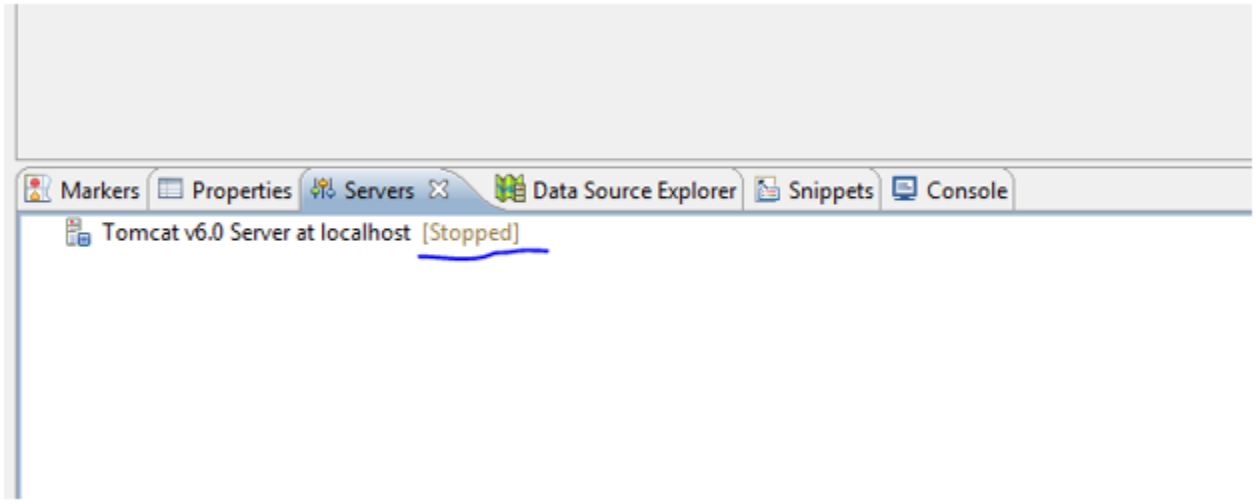
On the server tab, add one server (here I am using tomcat) in which you will run your web services.

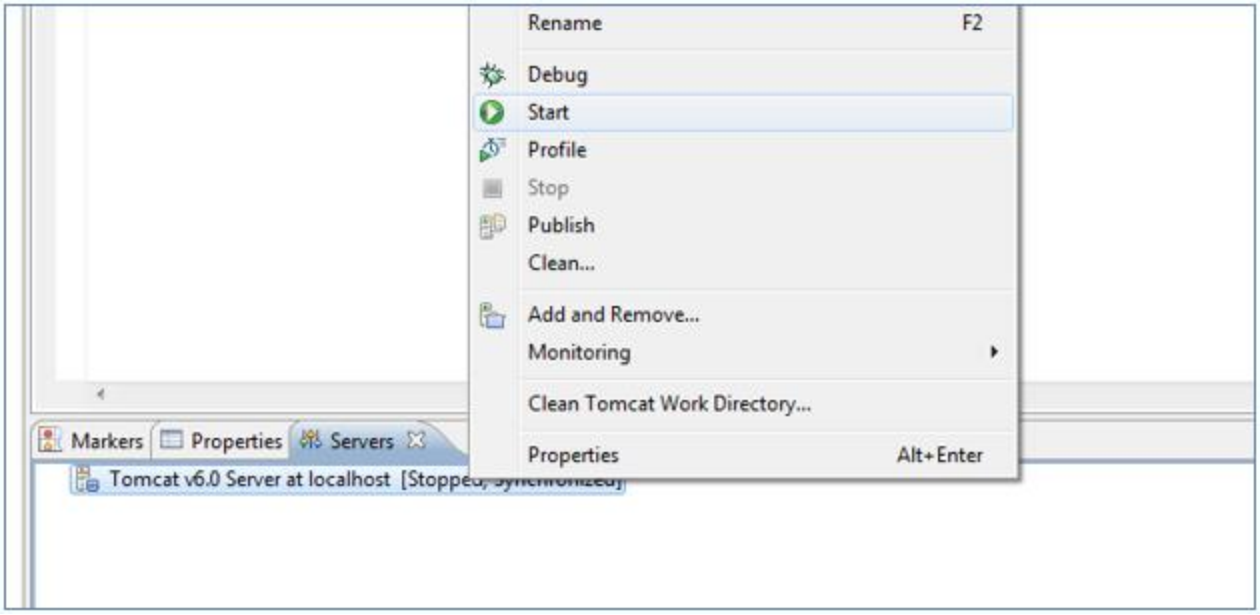
Step 2: Right-click on server tab> New> Server> Select Apache tomcat 6> Finish.

After that, you can see the server is created in the “Stopped” state; we will start the servers before running our application.


Markers
Properties
Servers
Data Source Explorer
Snippets
Console
New
Server
Properties
Alt+Enter

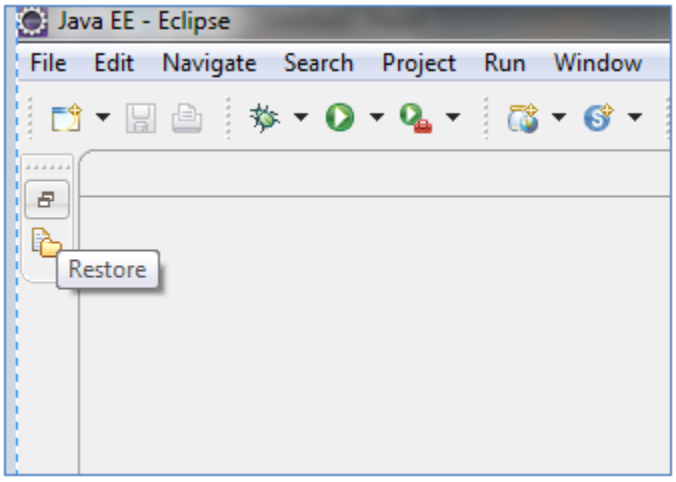

Project Run Window Help
Define a New Server
Choose the type of server to create
Server's host name: localhost
Download additional server adapters
Select the server type:
type filter text
Apache
Tomcat v3.2 Server
Tomcat v4.0 Server
Tomcat v4.1 Server
Tomcat v5.0 Server
Tomcat v5.5 Server
Tomcat v6.0 Server
Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server.
Server name: Tomcat v6.0 Server at localhost
Server runtime environment: Apache Tomcat v6.0
Add...
Configure runtime environments...
Servers
Data Source Explorer
< Back
Next >
Finish
Cancel

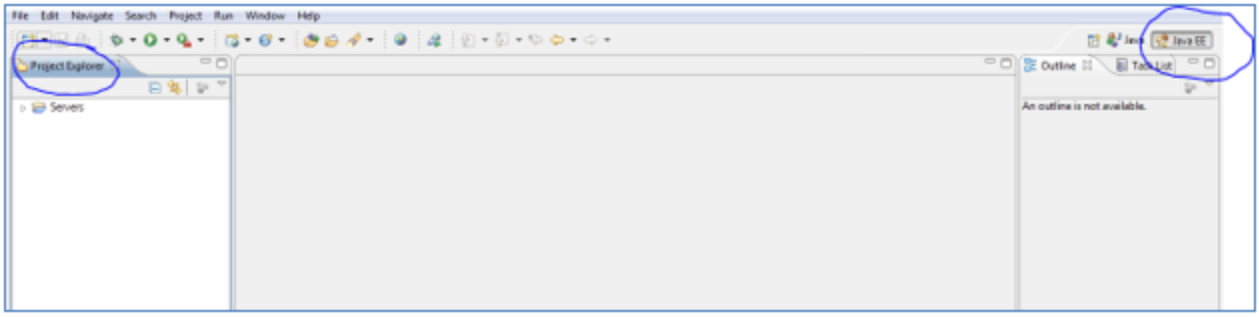

Markers
Properties
Servers
Data Source Explorer
Snippets
Console
Tomcat v6.0 Server at localhost [Stopped]



Now we will start our server.

Step 3: Right-click on tomcat > start



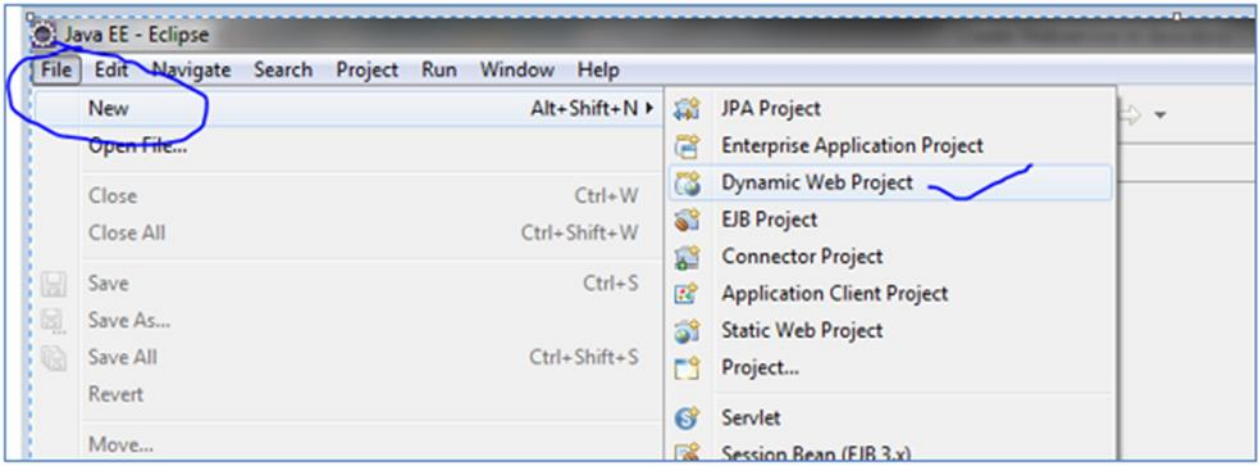


Now we will open project explorer for java EE.

Step 4: Click on restore > you will see project explorer

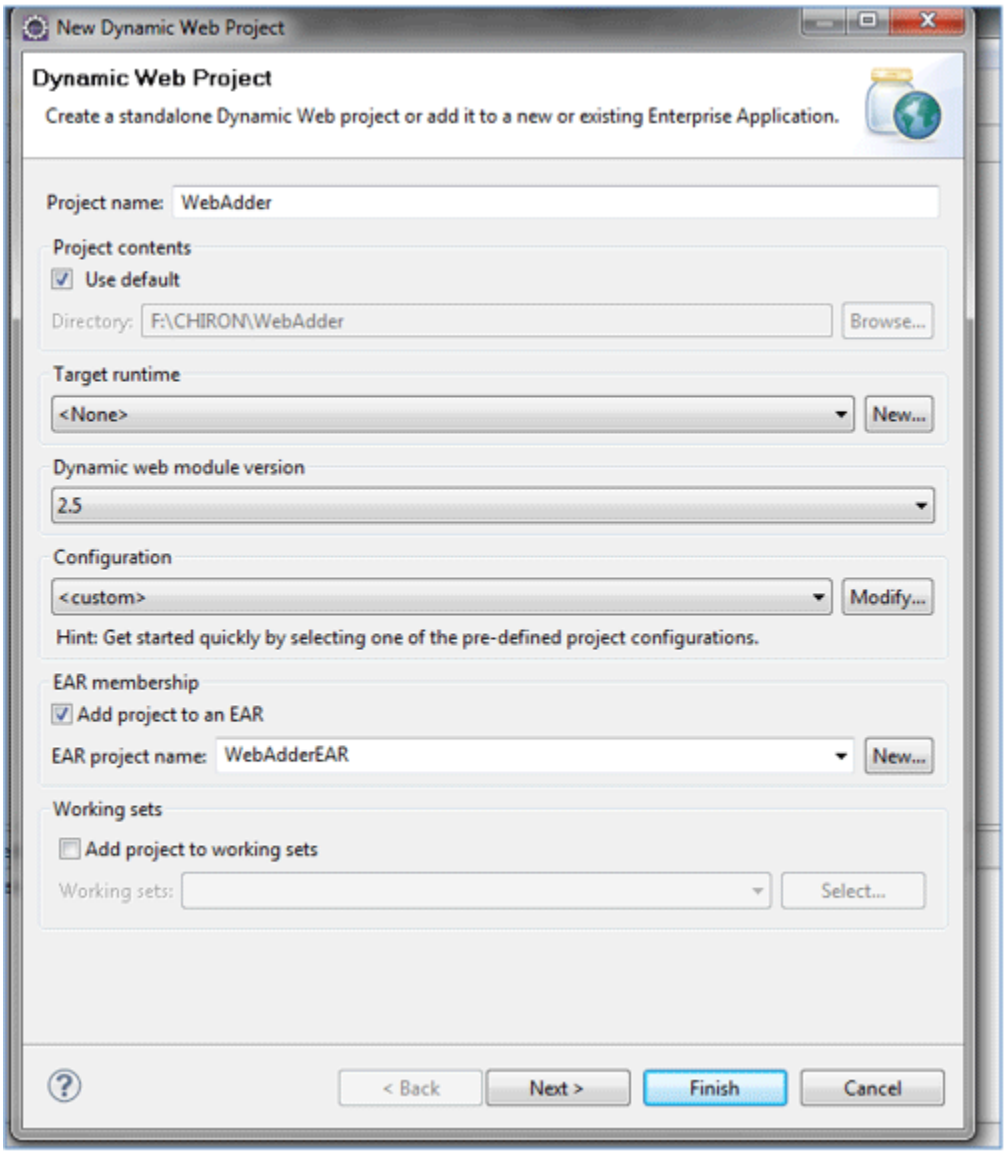
Now we will create our web service.

Step 5: File>New>Dynamic Web Project

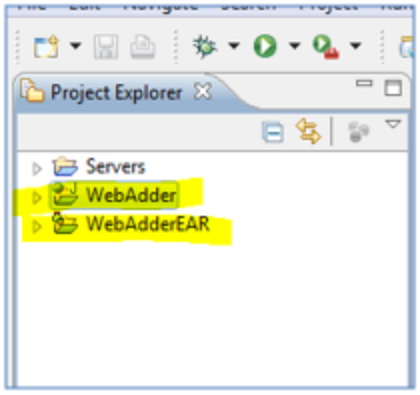


Could you give it a name (here, WebAdder)? Follow the below picture and do exactly the same as what is in there.

Step 6: Click on next> Finish

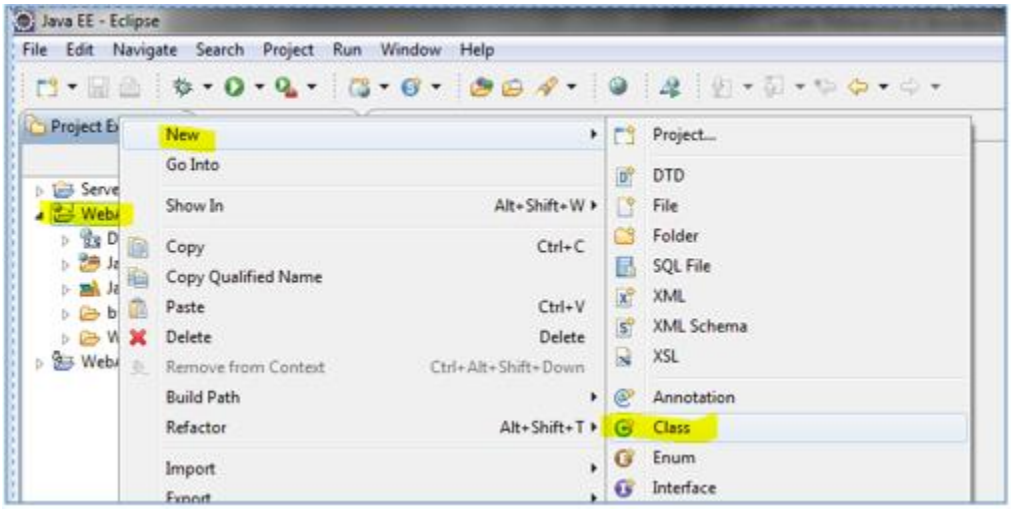


You will see your project is added as below

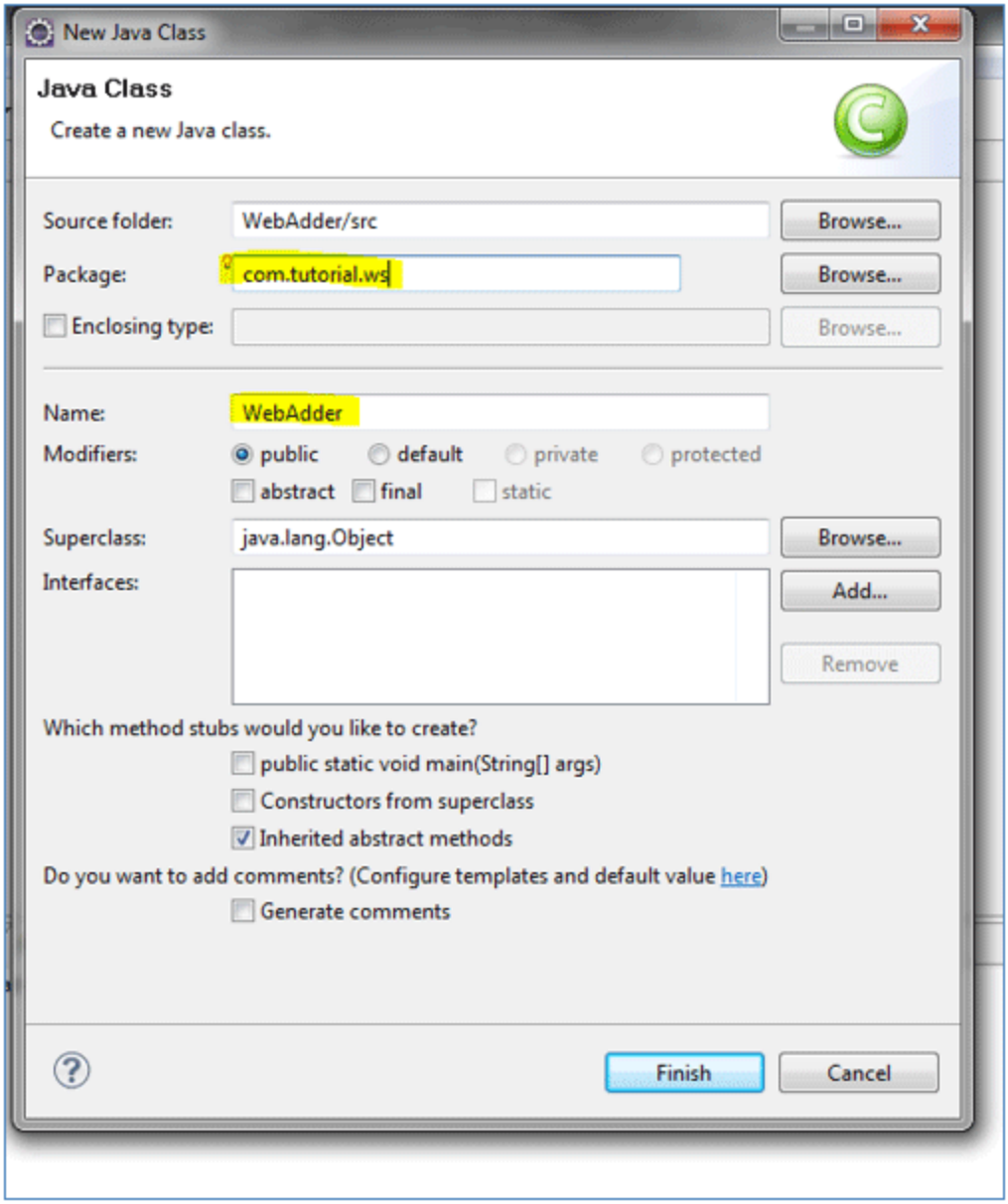


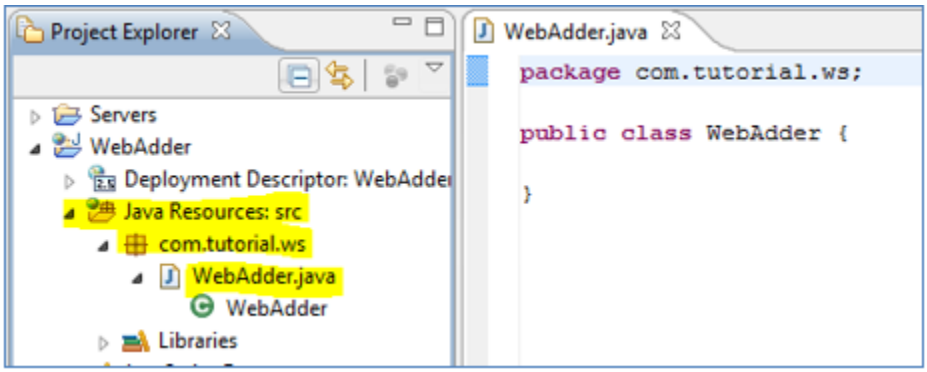
Now create a class.

Step 7: Right-click on “WebAdder”> New> Class



Step 8: Give it a name along with package name as per below picture > FInish.


New Java Class
Java Class
Create a new Java class.
Source folder:
WebAdder/src
Browse...
Package:
com.tutorial.ws
Browse...
Enclosing type:
Browse...
Name:
WebAdder
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
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? (Configure templates and default value here)
Generate comments
Finish
Cancel


Project Explorer
Servers
WebAdder
Deployment Descriptor: WebAdder
Java Resources: src
com.tutorial.ws
WebAdder.java
WebAdder
Libraries
WebAdder.java
package com.tutorial.ws;
public class WebAdder {
}

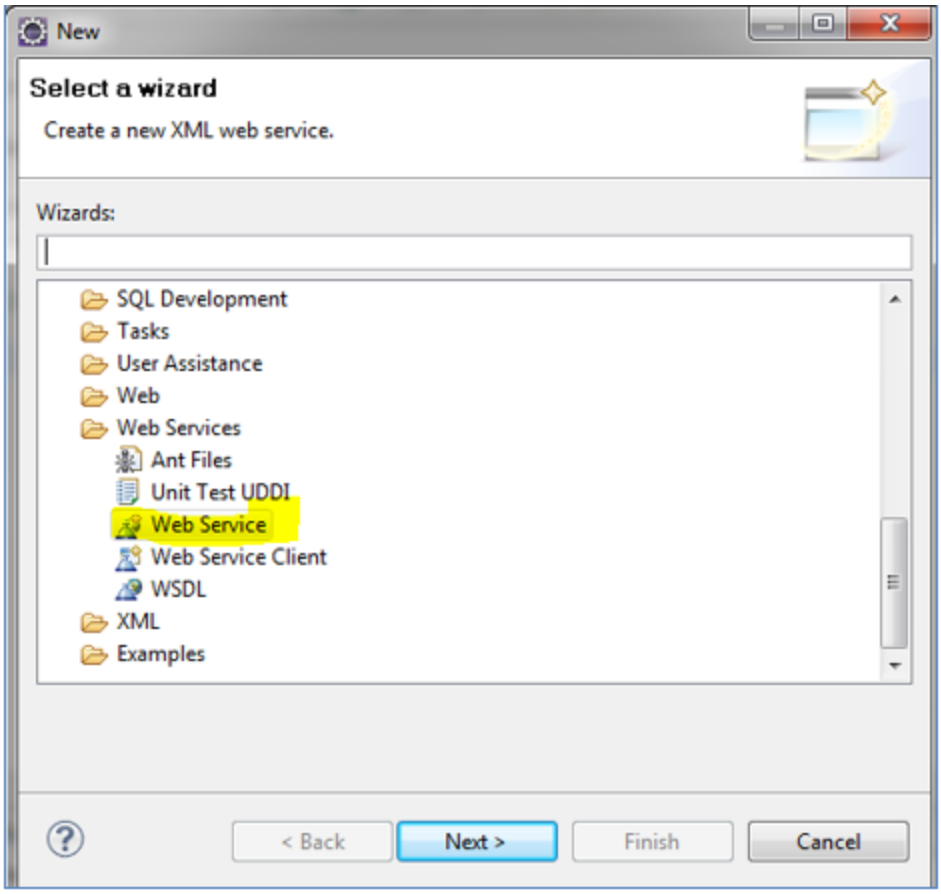
This class will be acting as a web service, so whatever methods we will write inside this class will act as web service methods.

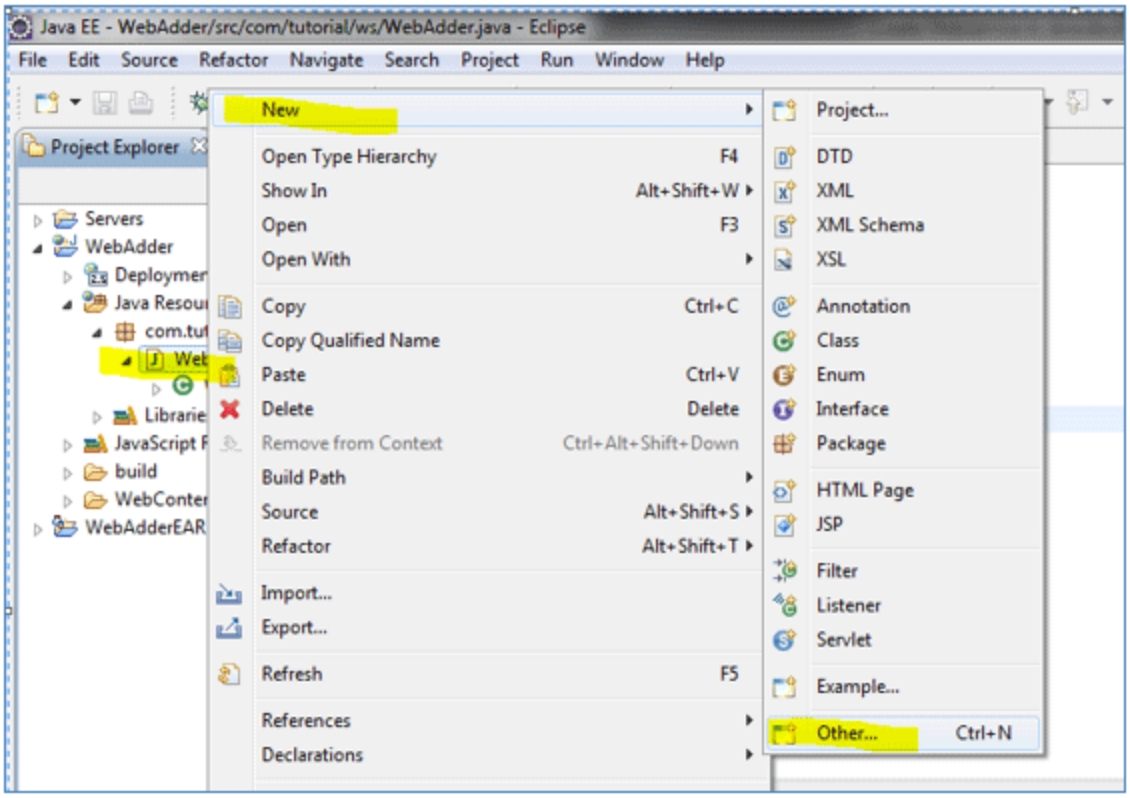
Step 9: Below is the code:

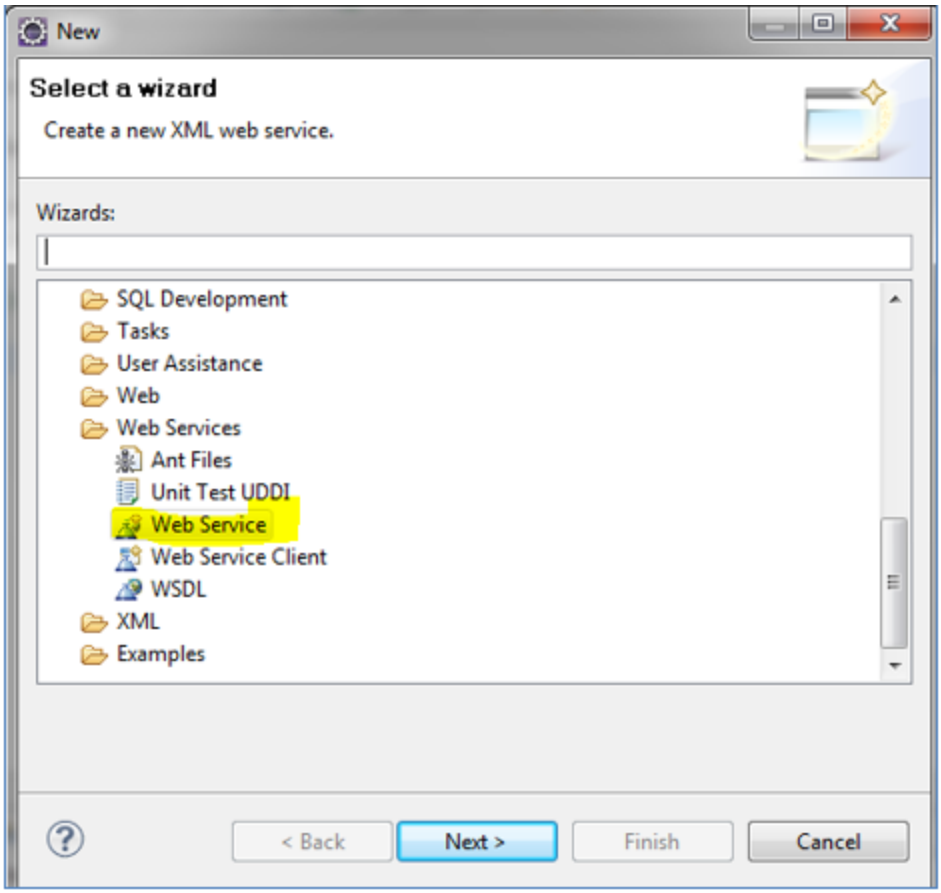
```
package com.tutorial.ws;
public class WebAdder {
public int addition (int nm1, int nm2){
return nm1+nm2;
}
// above method just doing addition
}
```

Now we want to make the class and method as a web service; for this we need to do the following:

Step 10: Right-click on WebAdder class> New>Other> Webservice>Next

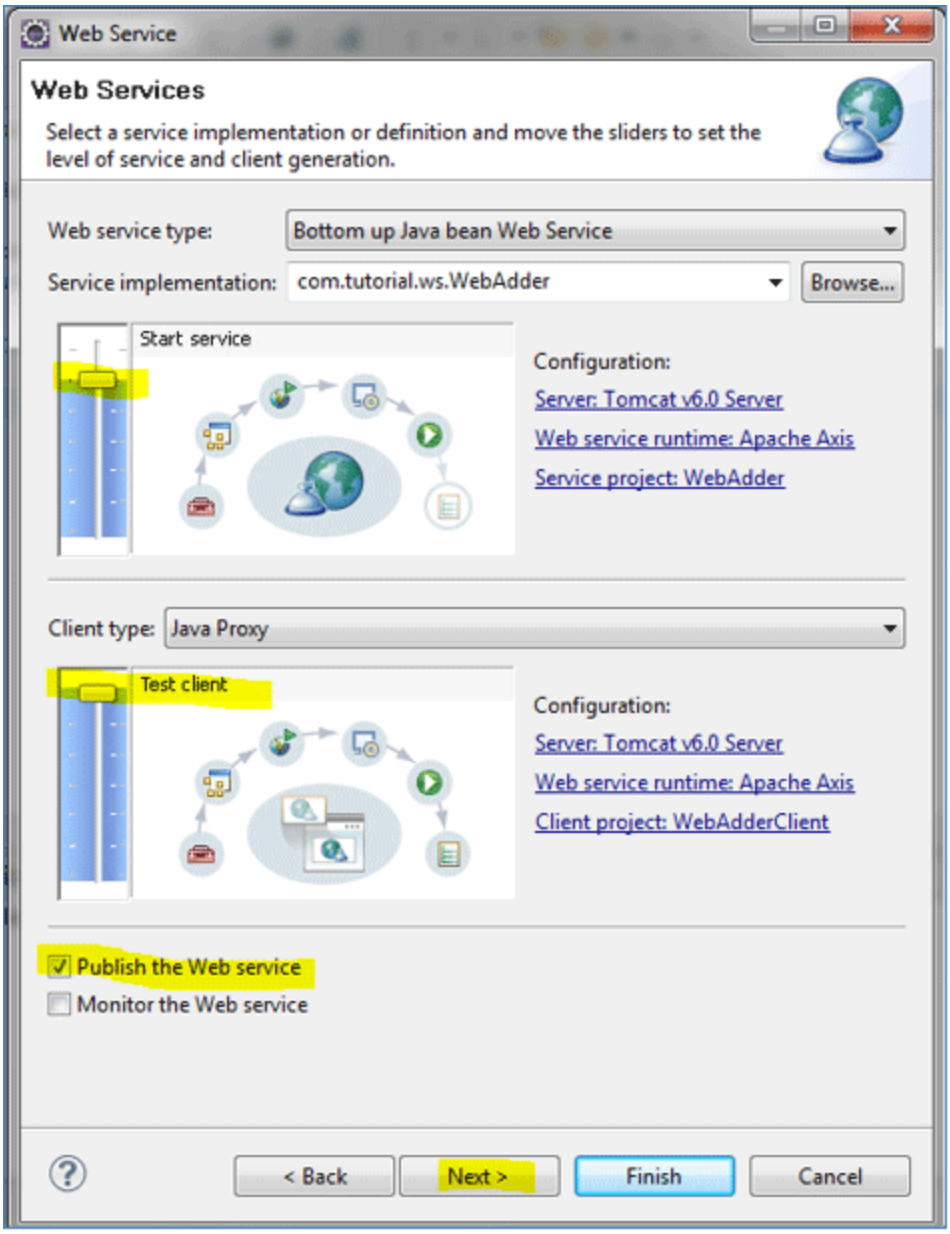

New
Select a wizard
Create a new XML web service.
Wizards:
SQL Development
Tasks
User Assistance
Web
Web Services
Ant Files
Unit Test UDDI
Web Service
Web Service Client
WSDL
XML
Examples
< Back
Next >
Finish
Cancel


Java EE - WebAdder/src/com/tutorial/ws/WebAdder.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Servers
WebAdder
New
Open Type Hierarchy F4
Show In Alt+Shift+W
Open F3
Open With
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
References
Declarations
Project...
DTD
XML
XML Schema
XSL
Annotation
Class
Enum
Interface
Package
HTML Page
JSP
Filter
Listener
Servlet
Example...
Other... Ctrl+N



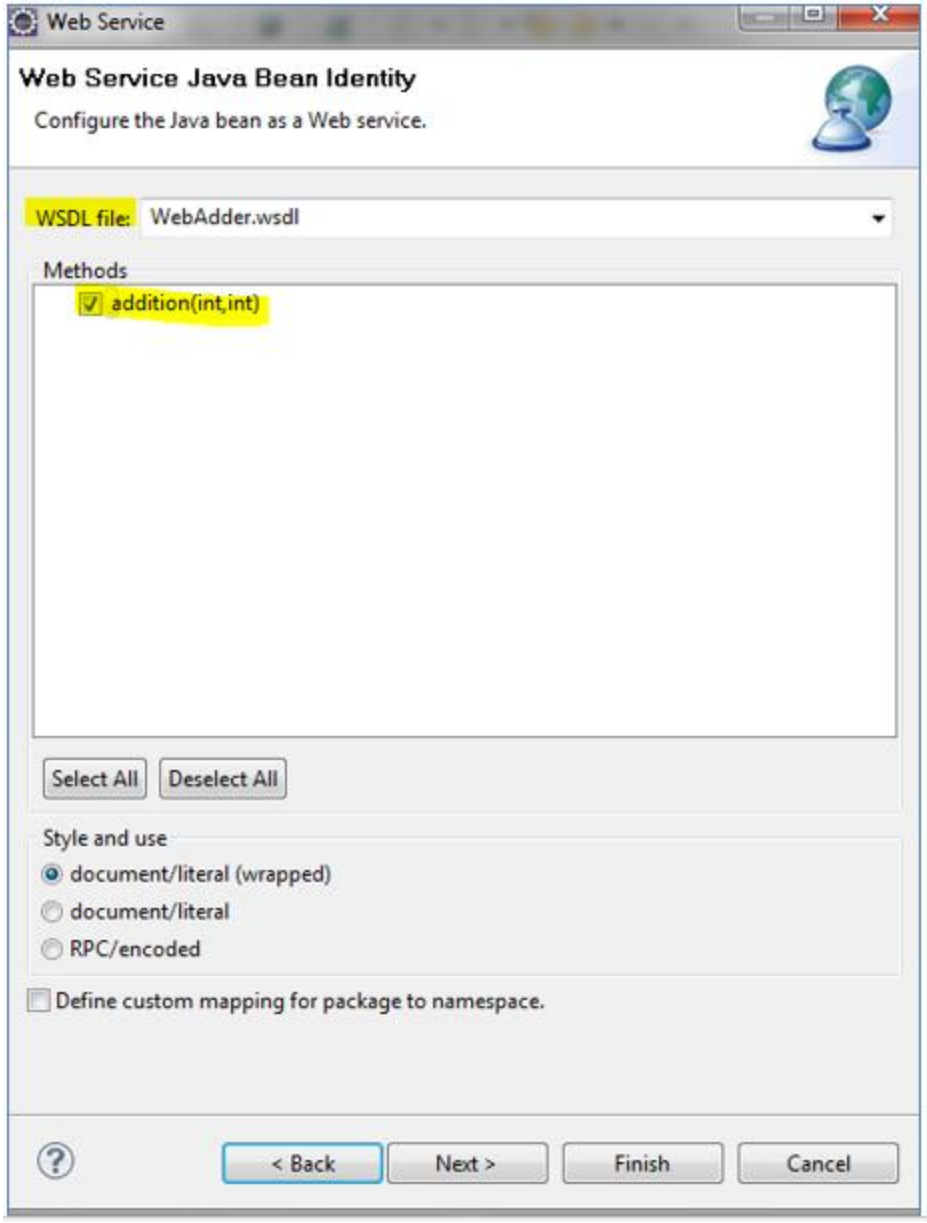
Here in the below screenshot, please pay attention to the highlighted portions; we are here creating clients also to test our web service. Follow the exact things as mentioned in the highlighted portion of the below screenshot and click on “Next.”

Step 11: Click on Next> “Finish.”

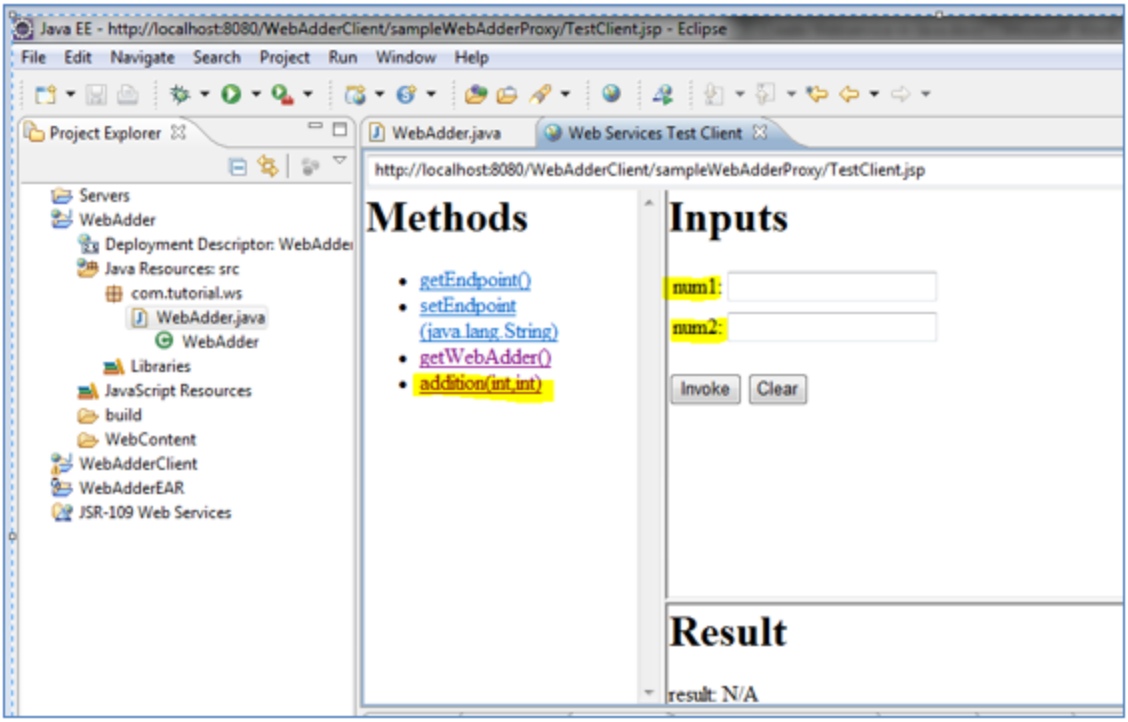


On the next page, as in the below screenshot, you can see that, addition () method becomes a web service method that will go in the WSDL document.

Step 12: Now you can see, your client is running on the server, where you need to click on your web method called “addition”,

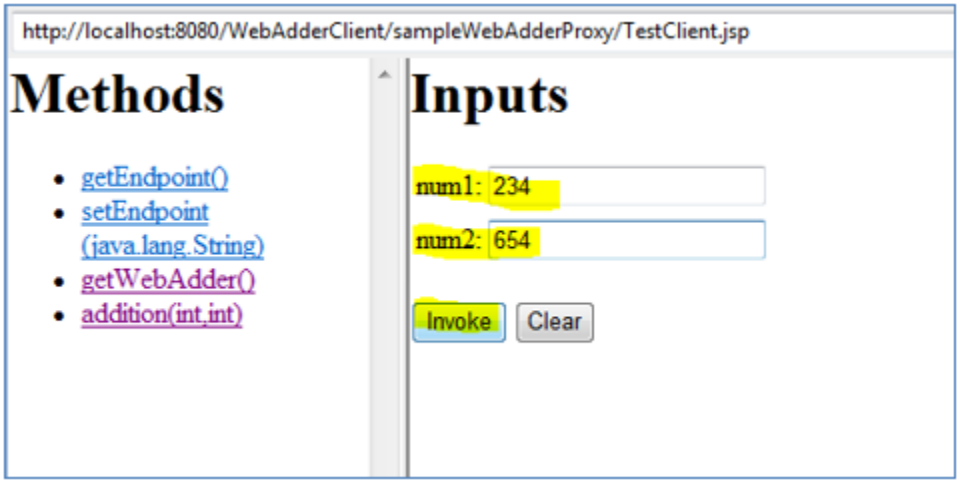


Step 13: Then, you will see that two fields will come, num1 and num2, to provide user input. Refer to the below screenshot:

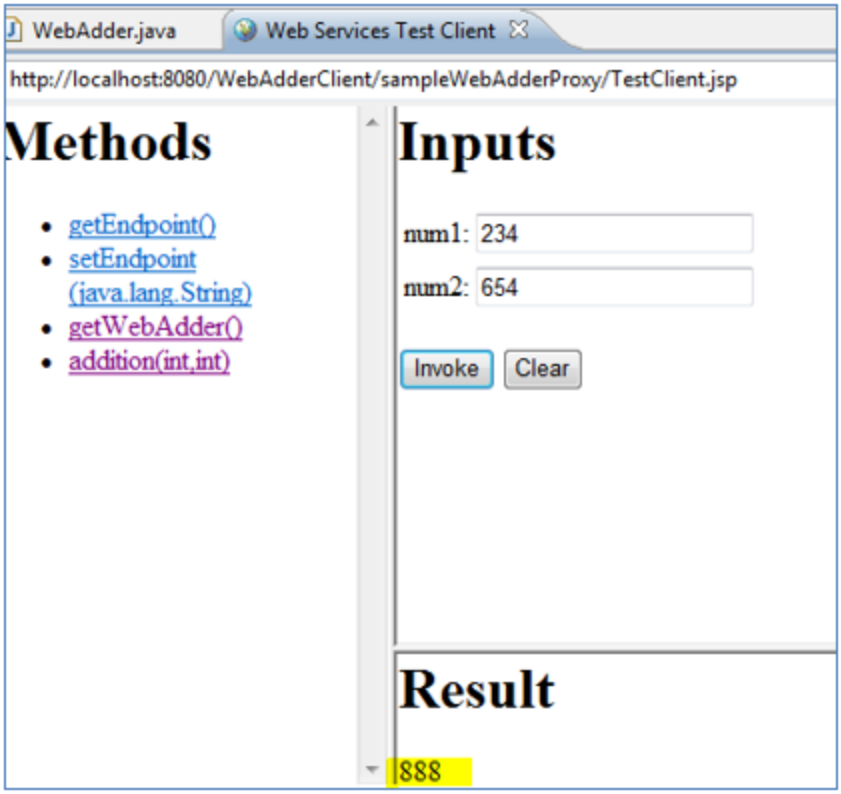


tep

14: Give some random integer values to get the result.



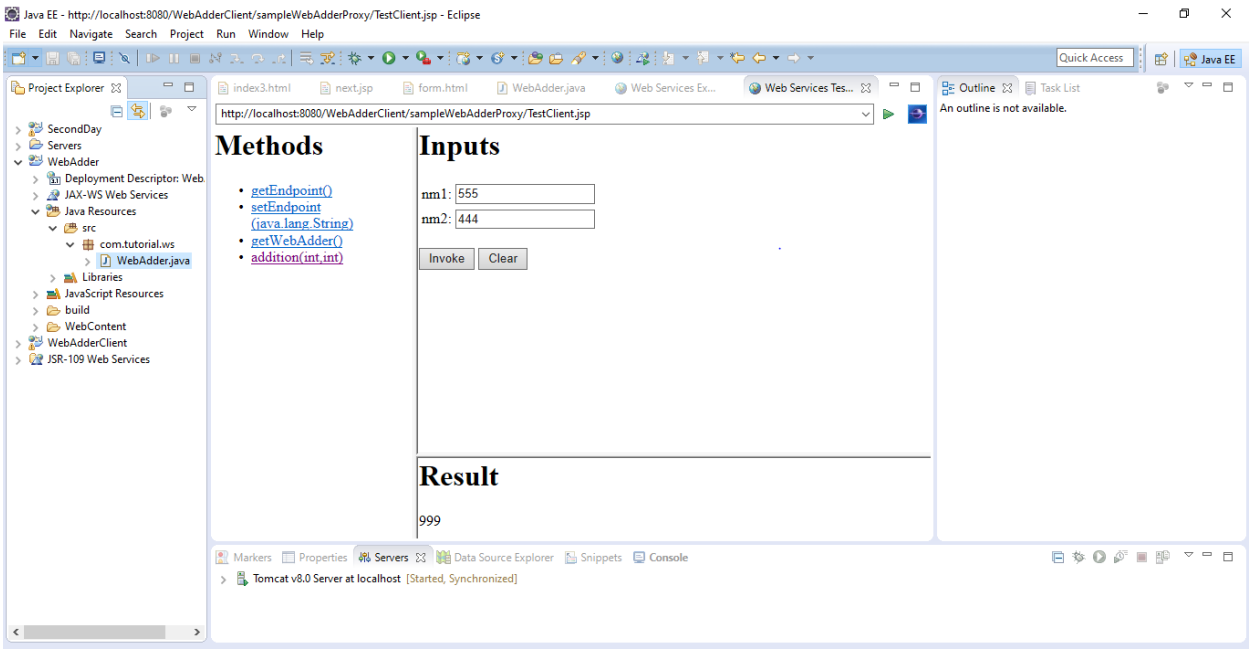
Step 15: After clicking on the "invoke" button, you will see the result. Congrats, you have completed building your first web service in java!



Program link:

https://www.theserverside.com/video/Step-by-step-SOAP-web-services-example-in-Java-using-Eclipse.

Output:


Java EE - http://localhost:8080/WebAdderClient/sampleWebAdderProxy/TestClient.jsp - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
SecondDay
Servers
WebAdder
Deployment Descriptor: Web
JAX-WS Web Services
Java Resources
src
com.tutorial.ws
WebAdder.java
Libraries
JavaScript Resources
build
WebContent
WebAdderClient
JSR-109 Web Services
index3.html
next.jsp
form.html
WebAdder.java
Web Services Ex...
Web Services Tes...
http://localhost:8080/WebAdderClient/sampleWebAdderProxy/TestClient.jsp
Methods
getEndpoint()
setEndpoint (java.lang.String)
getWebAdder()
addition(int,int)
Inputs
nm1: 555
nm2: 444
Invoke
Clear
Result
999
Outline
Task List
An outline is not available.
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
Tomcat v8.0 Server at localhost [Started, Synchronized]