University of Westminster
Faculty of Science and Technology

Module code and title: 5COSC004W-Client Service Architecture Tutorial Manual	
Tutorial title	**Getting acquainted with the concepts and procedures of the Coding Exercise Two (Building a real Client/Server from a Dummy Client)**
Tutorial type	**Guided and indepenent and non-marked**
Week 07	**14/03/19**

Contents

Learning Goals

This tutorial focuses on two main learning goals:

- to get acquainted with the the concepts of the second Coding Exercise (create a client/server from a dummy client)

It is divided into two separate sections, the student will perform the first task (1-X) following the instructions of the tutor, and then, will complete the other tasks independently.

TASKS to be Performed under the instruction of the Tutor (from Task 1 to Task 10)

1) Start Netbeans in your system. If Netbeans is not present in your system, use AppsAnywhere to launch it: (https://www.westminster.ac.uk/sites/default/public-files/general-documents/Using%20AppsAnywhere.pdf)
2) Given the dummy client in Figure 1, you can observe the usual testing connection test. The dummy client is connected to a dummy server that exposes two methods:

a. Boolean isConnected() that returns true if the server is running, and,
b. String sendMessage(String arg) that returns a String which contains arg.

```
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args)
    {
        DummyClient client  =   new DummyClient();
        client.executeTest();

    }

    private void executeTest()
    {
        System.out.println("[CLIENT] - Executing Test...");
        if(server.isConnectet())
        {
            System.out.println("[CLIENT] - Server is connected...");
            System.out.println("[CLIENT] - Server has returned " + server.sendMessage("hello !"));
        }
        else
        {
            System.out.println("[CLIENT] - Server is NOT connected, test Failed");

        }

        System.out.println("[CLIENT] - Test Completed");
    }

}
```

Figure 1, Dummy Client

3) Now, you have to create a REAL server that implements these functions and a real client to test it.
4) Create a Web Application to host the Web Service. (Figure 2 and Figure 3)

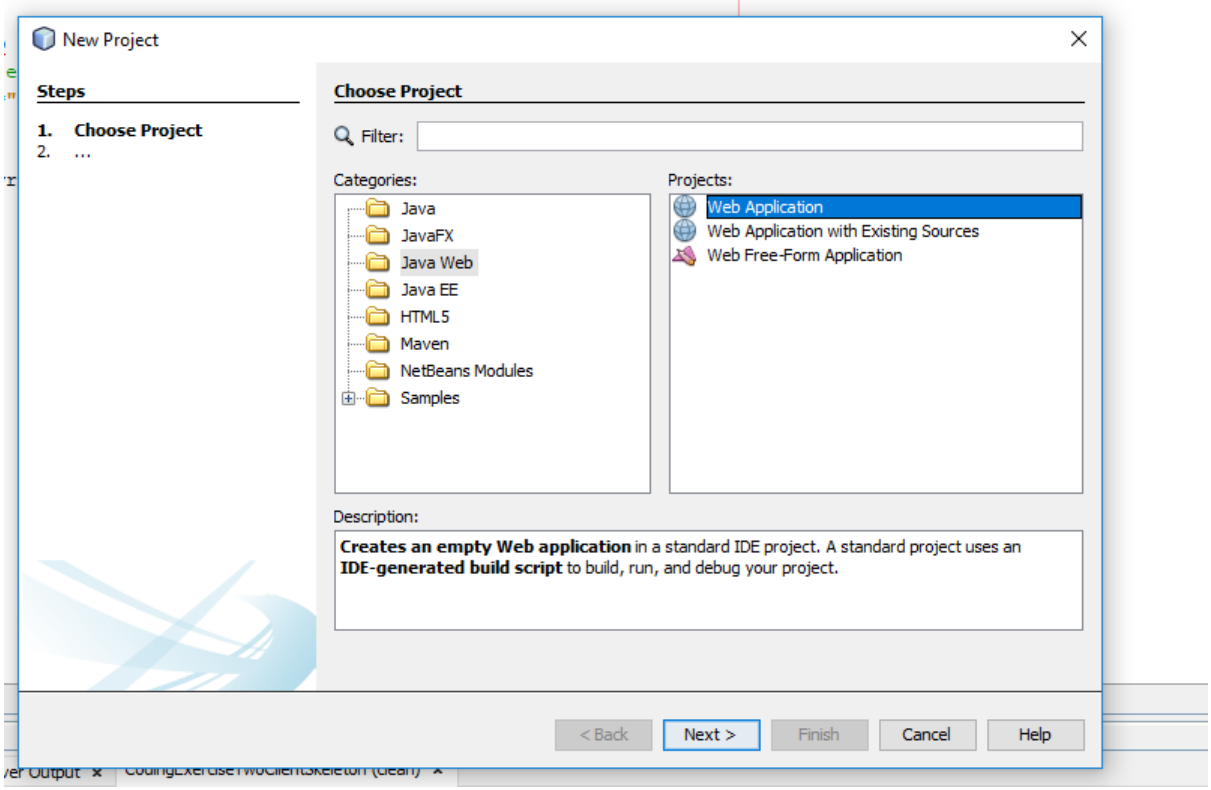

Figure 2, Create Web Application (A)

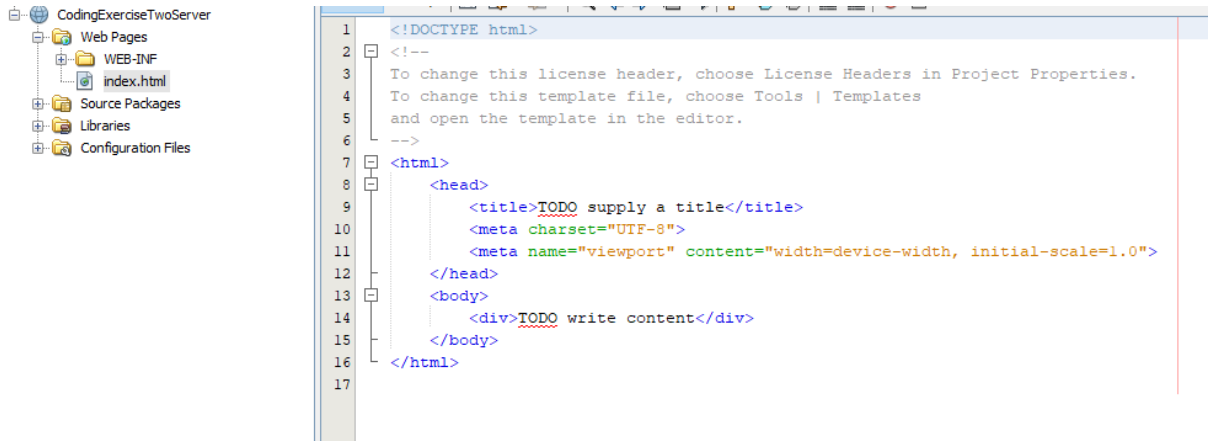

Figure 3, Create Web Application (B)

5) Create a server package in the source code. Figure 4

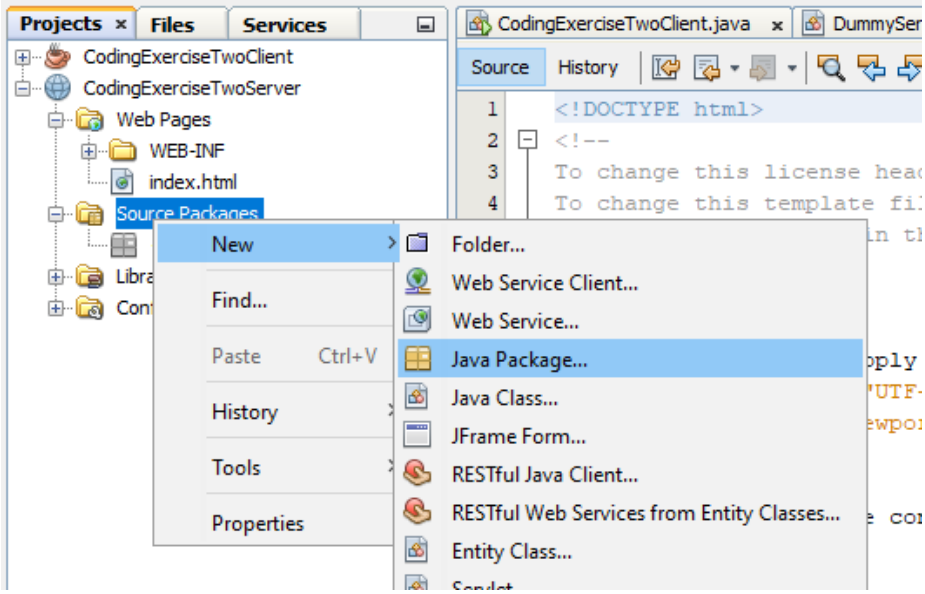

Figure 4, Create server package

6) Create a Web Service. (Figure 5 and Figure 6)

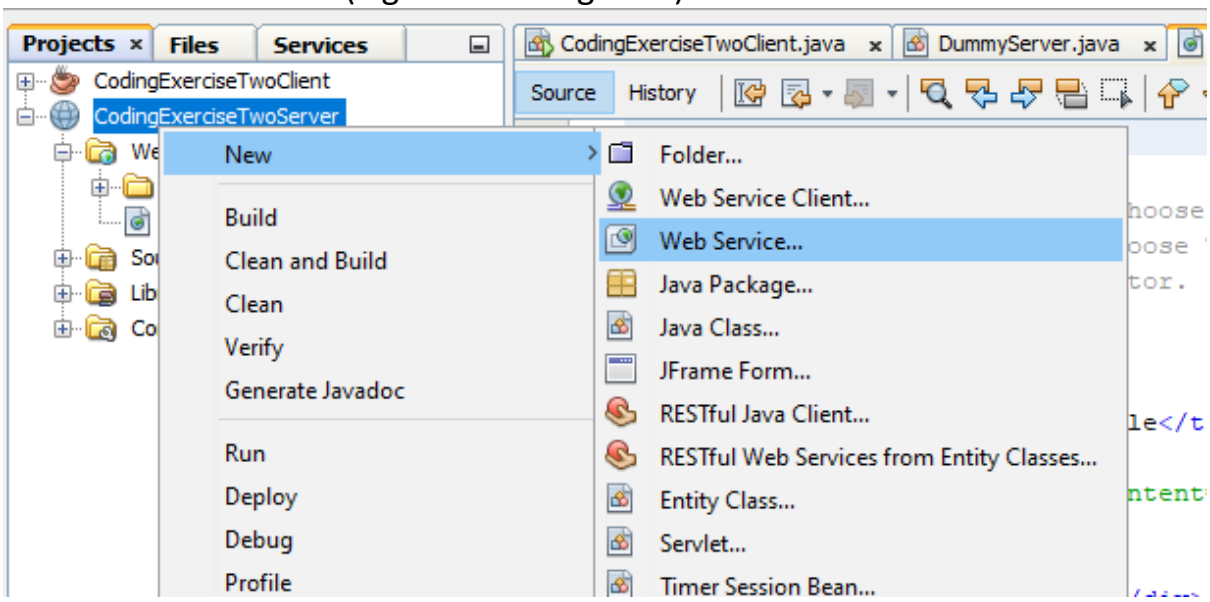

Figure 5, Create Web Service (A)

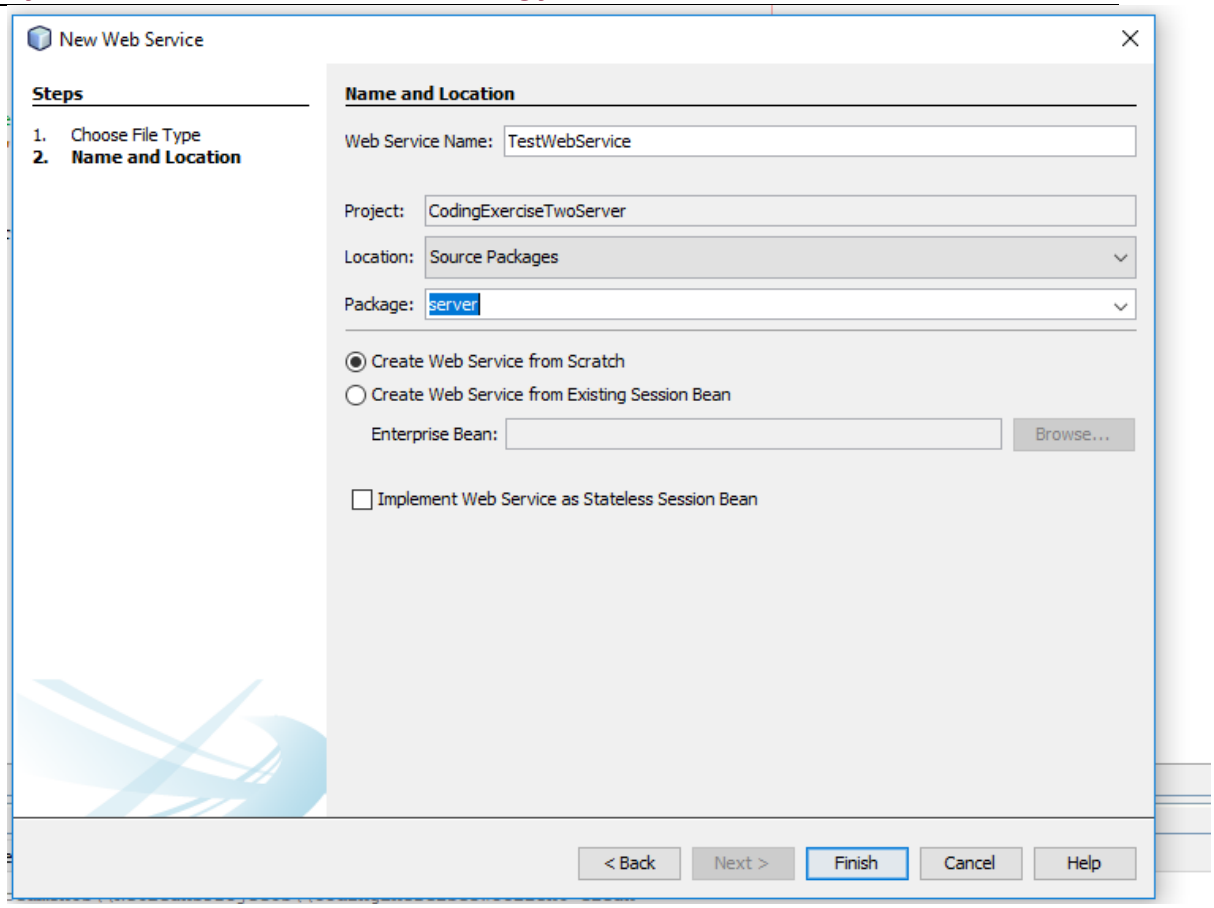

Figure 6, Create Web Service (B)

7) Add the two methods to the Web Service

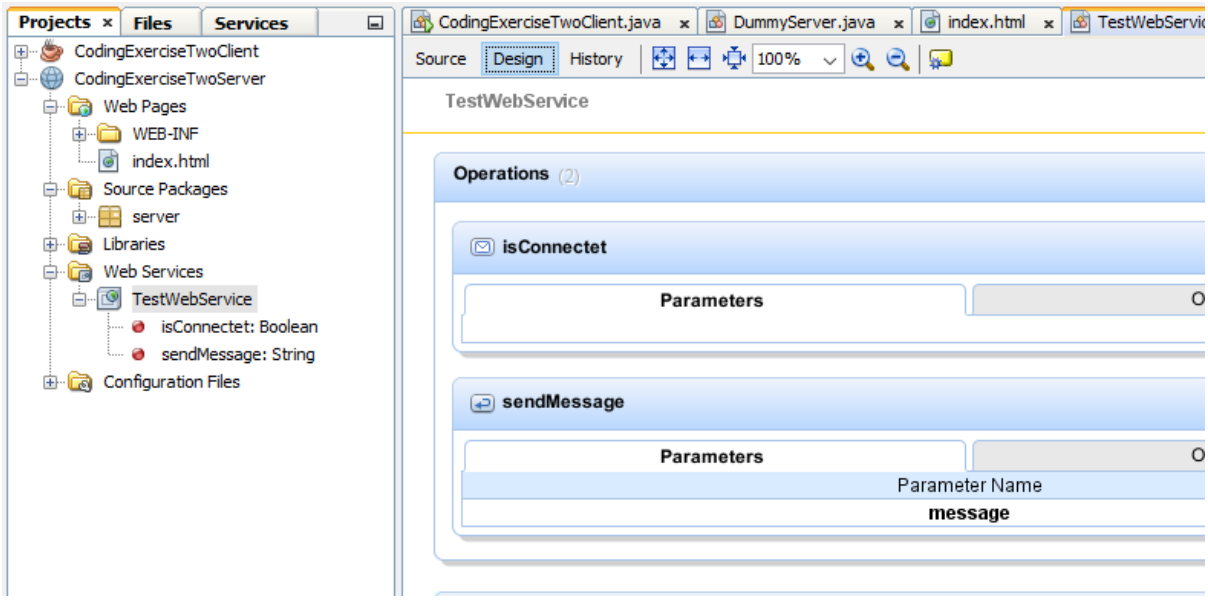

Figure 7, Adding Methods to the Web Service

8) Add the logic (Figure 9) into the empty method skeletons created by NetBeans (Figure 8)

```
import javax.jws.WebParam;

/**
 *
 * @author Gab
 */
@WebService(serviceName = "TestWebService")
public class TestWebService {

    /**
     * Web service operation
     */
    @WebMethod(operationName = "isConnectet")
    public Boolean isConnectet() {
        //TODO write your implementation code here:
        return null;
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "sendMessage")
    public String sendMessage(@WebParam(name = "message") String message) {
        //TODO write your implementation code here:
        return null;
    }
}
```

Figure 8, Empty Methods Skeletons created by Netbeans

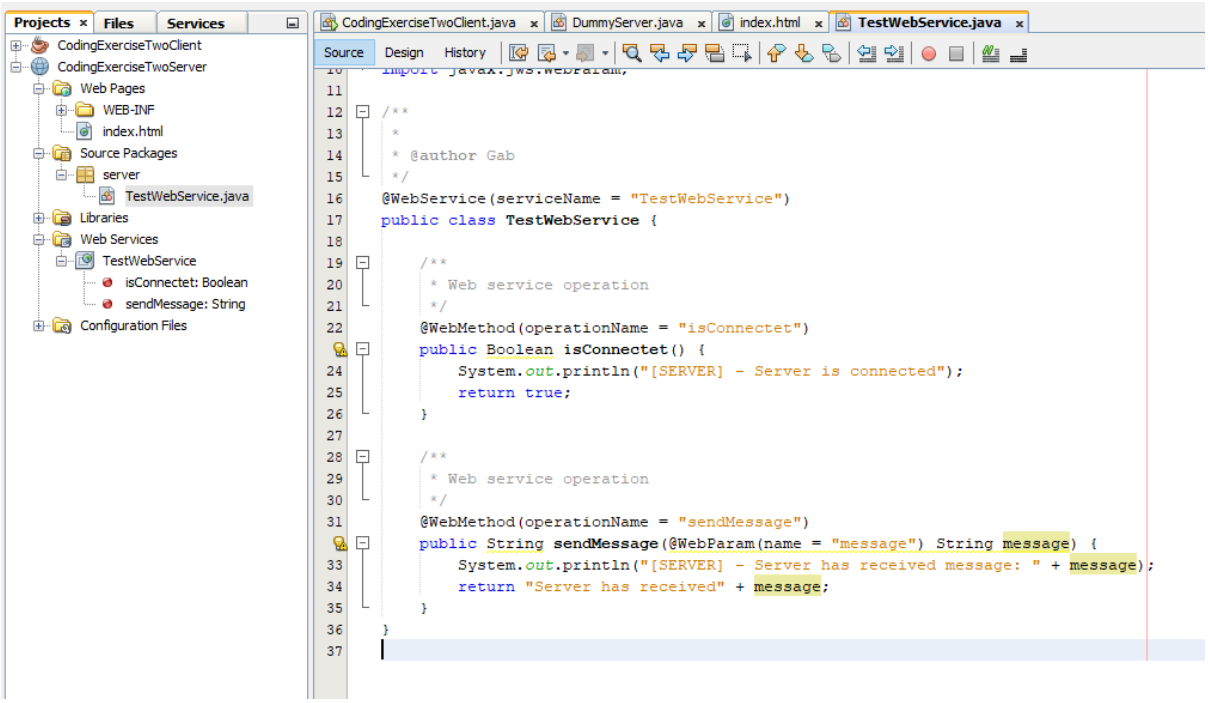

Figure 9, Adding logic to the empty methods.

9) Deploy (Figure 10) and test the Web Service (Figure 11 and Figure 12).

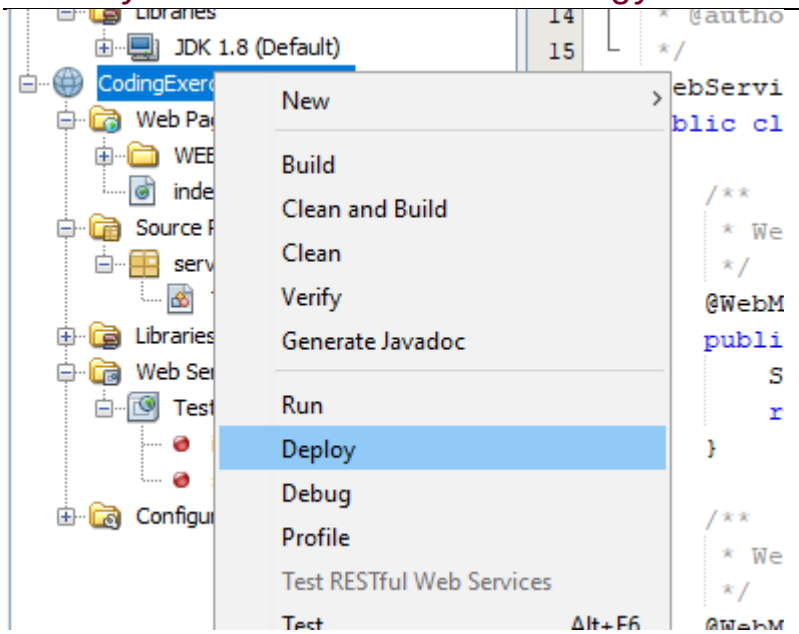

Figure 10, Deploy the Web Service

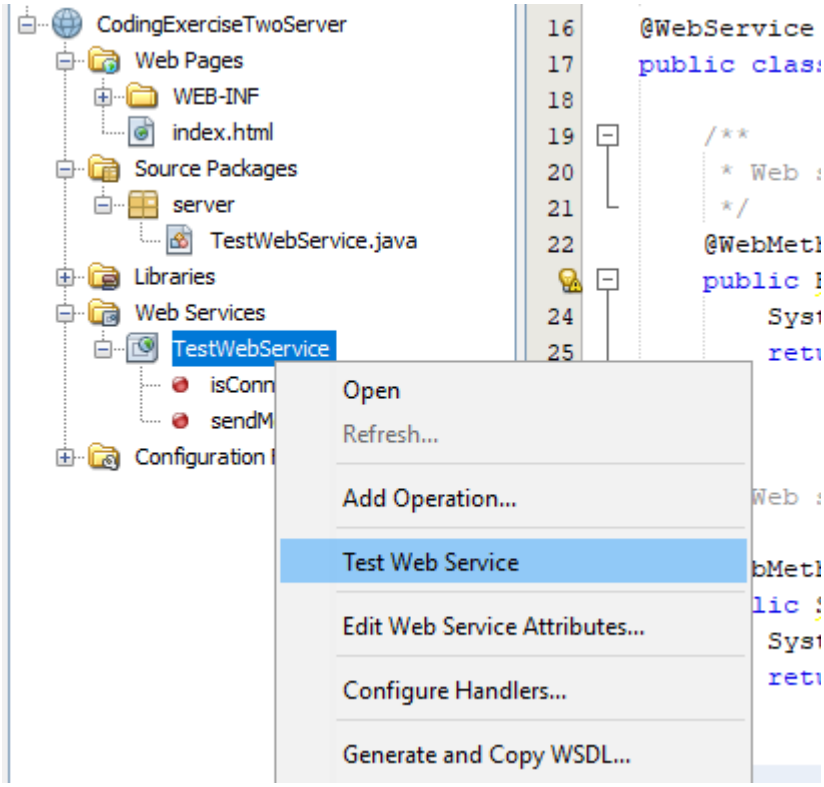

Figure 11, Test the Web Service (A)

Figure 12, Test the Web Service (B)

10) Create a Java Client for the web service you have just created (The steps are detailed in Tutorial 7.A

TASKS to BE PERFORMED Independently be the student (from Task 11 to Task 14) (Formative Assessment)

11) Modify the testConnection method so that the client can send its id (as a string) and that is returned from the method (e.g. Connection from client ….. succeded)
12) Modify the Server Class so that the server has a name (as a stgring) so that the returned string from the method testConnection is (e.g. Server….. : Connection from client ….. succeded)
13) Modify the Server Class so that the server can add a time stamp to method testConnection (e.g. [Date and Time] - Server….. : Connection from client ….. succeded)
14) Modify the Server and Client class so that the server raises an exception in the method sendMessage that is thrown when the argument is null.