Assignment - 7

Implementing the solution:

Step 1: Choosing a container:

You can either deploy your web service in a web container.

- Choose **File** > **New Project** (Ctrl-Shift-N on Linux and Windows).
- Select **Web Application** from the Java **Web** category.
- Name the project *CalculatorWSApplication*. ● Select a location for the project. Click **Next**.
- Select the server **[Glassfish / Tomcat]** and **Java EE** version and click **Finish.**

Step 2: Creating a Web Service from a Java Class:

- Right-click the *CalculatorWSApplication* node and **choose New** > **Web Service**.
- Name the web service *CalculatorWS* and type *org.me.calculator* in Package.
- Keep "**Create Web Service from Scratch**" check box selected.
- If you are creating a Java EE project on GlassFish, select "**Implement Web Service as a Stateless Session Bean"**.
- Click **Finish**. The Projects window displays the structure of the new web service and the source code is shown in the editor area.

Step 3: Adding an Operation to the Web Service:

- Find the web service's node in the Projects window. Right-click that node. A context menu opens.
- Click **Add Operation** in either the visual designer or the context menu. The Add Operation dialog opens.
- In the upper part of the **Add Operation dialog box**, type **add** in **Name** and type **int** in the **Return Type** drop-down list.
- In the lower part of the Add Operation dialog box, click **Add** and create a parameter of type **int** named **i**. Click **Add** again and create a parameter of type **int** called **j**.
- Click **OK** at the bottom of the Add Operation dialog box. You return to the editor.
- Remove the **default hello operation**, either by deleting the hello() method in the **source code** or by selecting the hello operation in the **visual designer** and clicking **Remove Operation**.
- Click **Source** menu and view the generated code.
- In the editor, extend the skeleton **add operation.** Add the following: int k = i + j;
  return k; (instead of return 0)

Step 4: Deploying and Testing the Web Service:

Once you deploy a web service to a server, you can use the IDE to open the server's test client. The GlassFish server provides test clients whereas in Tomcat Web Server, there is no test client.

- Right-click the **project** and choose **Deploy**. The IDE starts the application server, builds the application, and deploys the application to the server.
- In the IDE's **Projects** tab, expand the **Web Services** node of the *CalculatorWSApplication* project. Right-click the *CalculatorWS* node, and choose **Test Web Service.**
- The IDE opens the tester page in the browser, if you deployed a web application to the GlassFish server.

Step 5: Consuming the Web Service:

Once the web service is deployed, you need to **create a client** to make use of the web service's **add** method. Here, you can create three types of clients: a Java class in a Java SE application, a servlet, and a JSP page in a web application.

Client 1: Java Class in Java SE Application ●

Choose **File** > **New Project**.

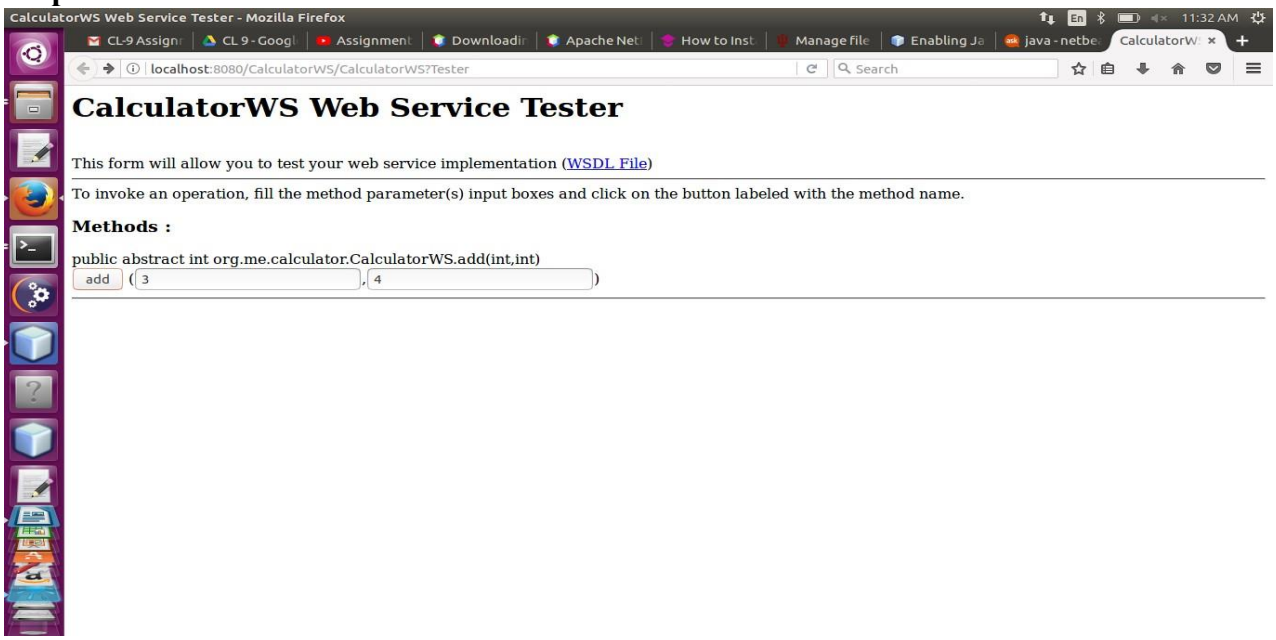
- Select **Java Application** from the **Java** category.
- Name the project CalculatorWS_Client_Application.
- Keep "**Create Main Class selected**" and accept all other default settings. Click **Finish**.
- Right-click the CalculatorWS_Client_Application node and choose **New** > **Web Service Client**. The New Web Service Client wizard opens.
- Select "Project as the **…** source. Click on **Browse** button. Browse to the CalculatorWS web service in the **CalculatorWSApplication** project.
- When you have selected the web service, click **OK**.
- Do not select **a package** name. Leave this field empty. Keep the other settings at default and click **Finish**.
- The Projects window displays the new **web service client**, with a node for the **add** method that is created
- Double-click your **main class** so that it opens in the Source Editor. Drag the **add** node below the main() method.
- In the main() method body, **write the** code that initializes values for i and j, calls add(), and prints the result.

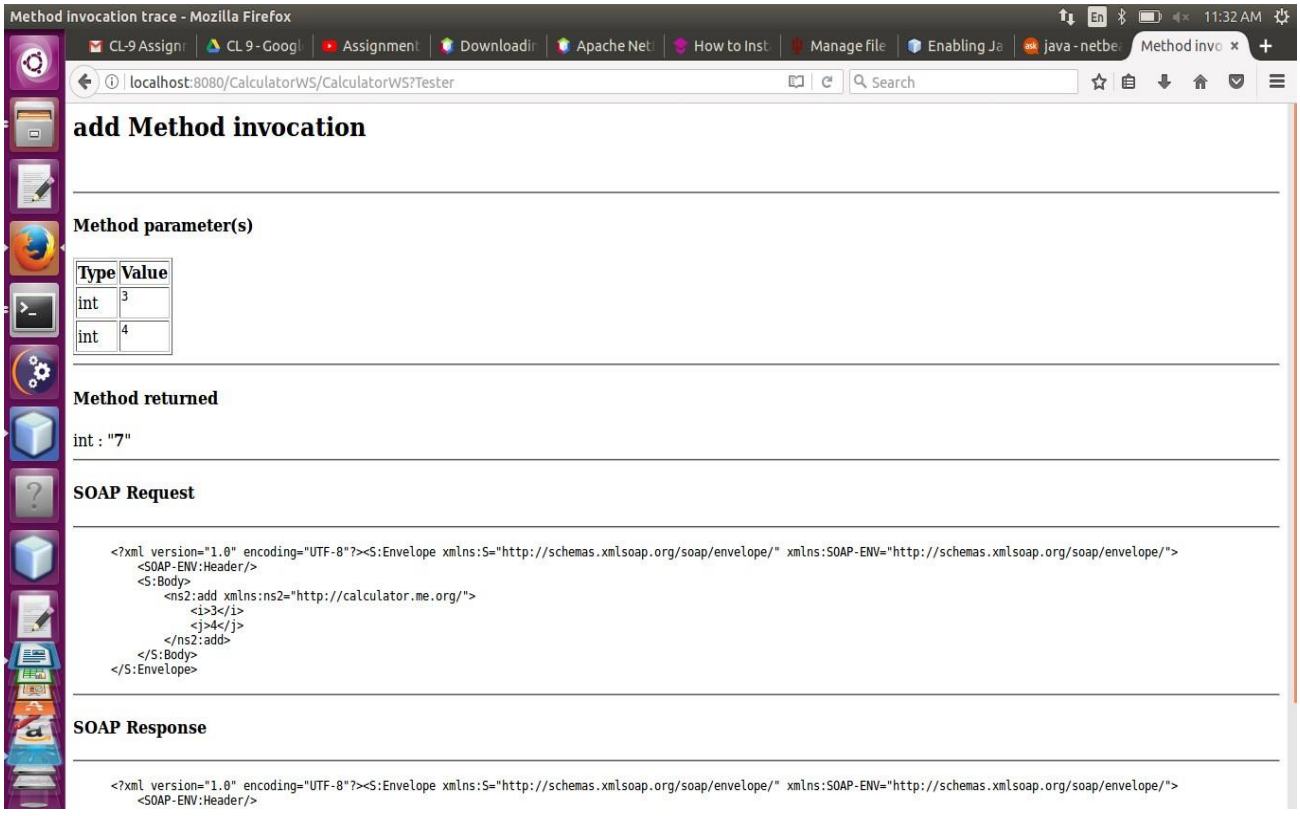
```
try { int i=3; int j=4; int result = add(i,j);
        System.out.println("Result =" +
result); } catch (Exception ex) {
          System.out.println("Exception =" + ex);
```

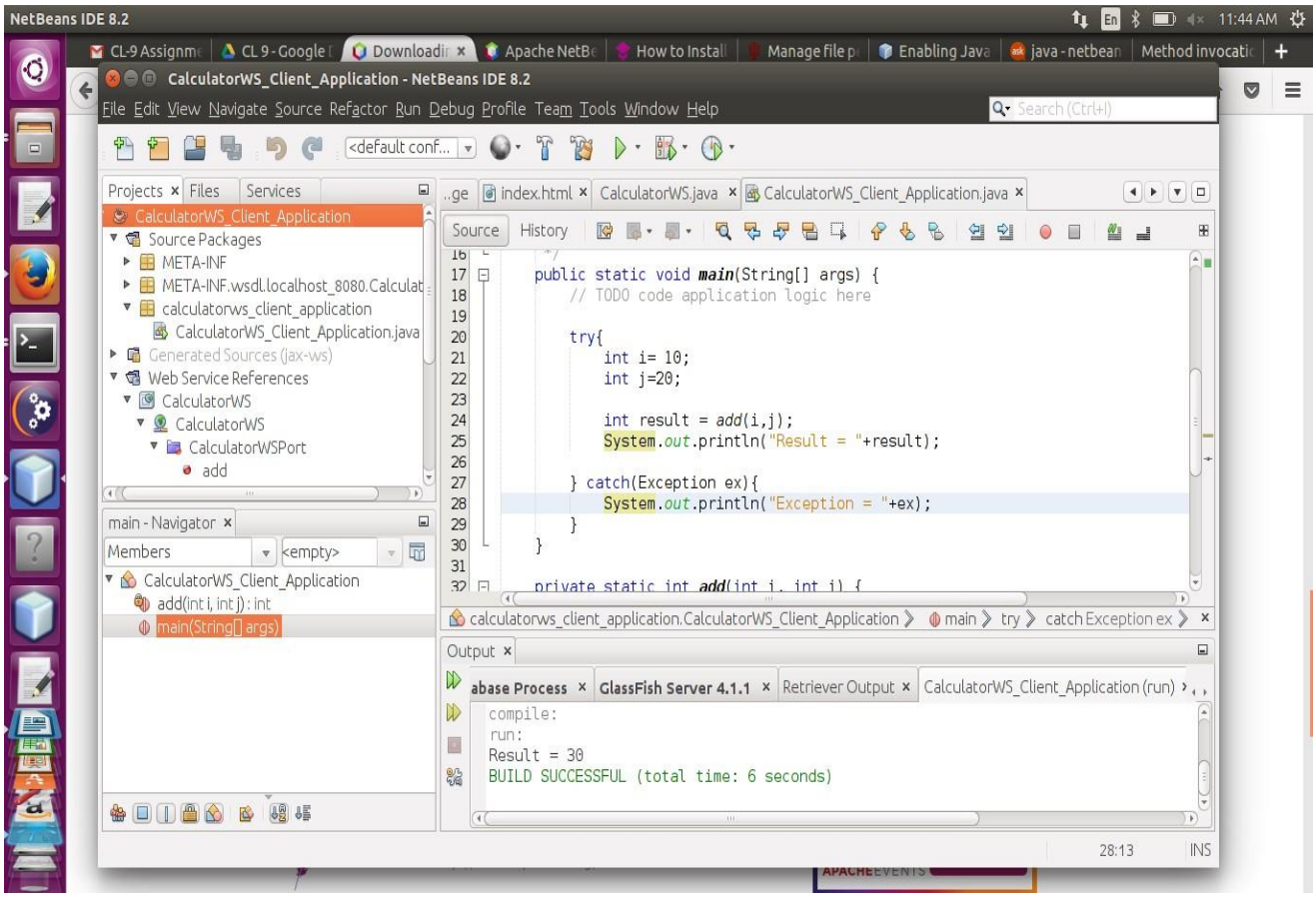
Compiling and Executing the solution:

Right Click on the Project node and Choose Run.

Output:






NetBeans IDE 8.2
11:44 AM
CalculatorWS_Client_Application - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Projects
Files
Services
CalculatorWS_Client_Application
Source Packages
META-INF
calculatorws_client_application
CalculatorWS_Client_Application.java
Generated Sources (jax-ws)
Web Service References
CalculatorWS
CalculatorWSPort
add
main - Navigator
Members
CalculatorWS_Client_Application
add(int i, int j) : int
main(String[] args)
index.html
CalculatorWS.java
CalculatorWS_Client_Application.java
Source
History
public static void main(String[] args) {
// TODO code application logic here
try{
int i= 10;
int j=20;
int result = add(i,j);
System.out.println("Result = "+result);
} catch(Exception ex){
System.out.println("Exception = "+ex);
}
}
calculatorws_client_application.CalculatorWS_Client_Application
main
try
catch Exception ex
Output
GlassFish Server 4.1.1
Retriever Output
CalculatorWS_Client_Application (run)
compile:
run:
Result = 30
BUILD SUCCESSFUL (total time: 6 seconds)
28:13
INS