

Les Services WEB

TP N°1: Développement d'un Service Web, Publication et Invocation en utilisant Java 1.7, AXIS2 1.6.4, Tomcat 7.0 et Eclipse Mars.

SOMMAIRE

I- Objectifs :	3
II- Outils utilisés :	3
III- Développement	3
A- Configuration de l'environnement de développement	3
A-1 Installation de la JDK version 1.7	3
A-2 Installation de Eclipse Mars 4.5.1	6
A-3 Configuration de Java (JDK et JRE) au niveau de Eclipse Mars	8
A-4 Installation de Tomcat 7.0	10
A-5 Création d'une Bibliothèque Utilisateur AXIS2 au niveau de Eclipse Mars	12
A-6 Configuration de Tomcat dans Eclipse	15
B- Développement du Service WEB (L'approche Bottom-Up)	19
C-Développement du Service Web client	32

I- Objectifs :

- Configurer Axis2 version 1.6.4 au niveau de Eclipse Mars ;
- Développer le Service Web en adoptant l'approche Bottom-Up (code first);
- Générer les Skelton (côté Serveur) ;
- Déployer le Service Web au niveau de Tomcat en utilisant Axis2;
- Générer automatiquement le fichier WSDL (La couche Définition dans l'architecture des services Web) ;
- Générer le fichier d'archive AAR ;
- Développer le Web Service Client et Générer les Stubs (côté Client);
- Invoker le Service Web.

II- Outils utilisés :

Dans ce TP, nous allons utiliser les outils suivants :

- ✓ Eclipse Mars 4.5.1 (comme IDE) ;
- ✓ Java 1.7 ou 1.8 ;
- ✓ Apache Tomcat 7.0 ;
- ✓ Apache Axis 2 version 1.6.4 (Binary Distribution + WAR Distribution)

Ces outils sont dans le dossier **/Dossier Services Web/tools** :

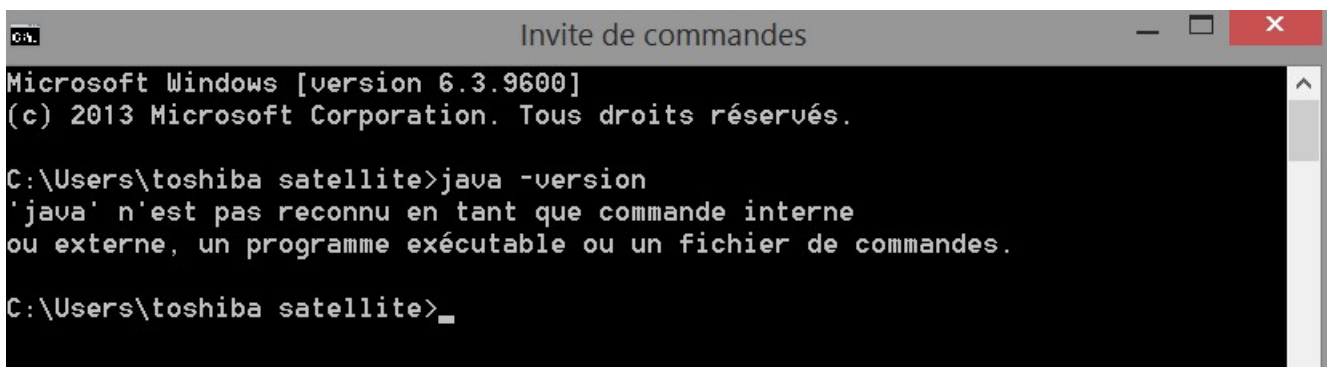
II- Développement

A- Configuration de l'environnement de développement

A-1 Installation de la JDK version 1.7

La première chose qu'il faut faire est de vérifier si java 1.7 est installée sur votre machine. Pour ceci, lancer la commande DOS **java -version**.

Si java, n'est pas installée, le message suivant sera affiché :

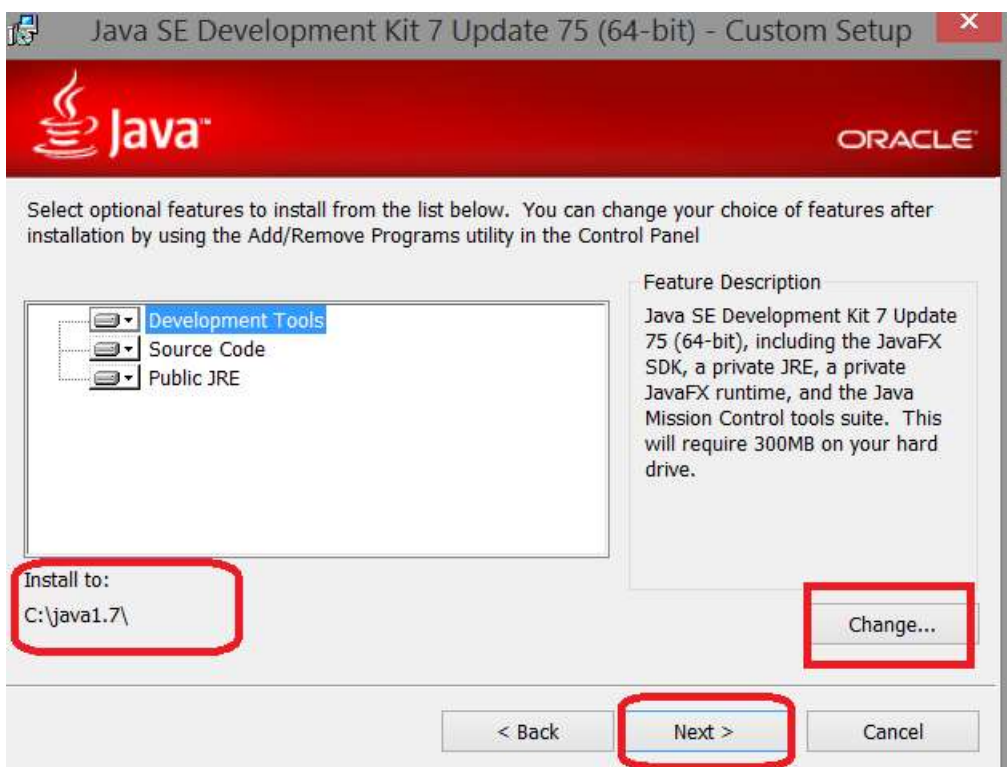


Suivre les étapes suivantes pour installer JDK 1.7 :

- Cliquer sur le fichier **jdk-7u75-windows-x64.exe**. La fenêtre suivante sera affichée :



- Cliquer sur Next. L'écran suivant sera affiché :



- Cliquer sur **Change...** pour préciser le dossier dans lequel sera installé Java. Cliquer ensuite sur **Next** pour installer Java 1.7. L'écran suivant sera affiché afin de configurer le chemin d'installation de la JRE :



- Choisir le dossier d'installation, puis cliquer sur **Next**. Java 1.7 sera finalement installé et l'écran suivant sera affiché :



- Cliquer sur **Close** pour terminer.

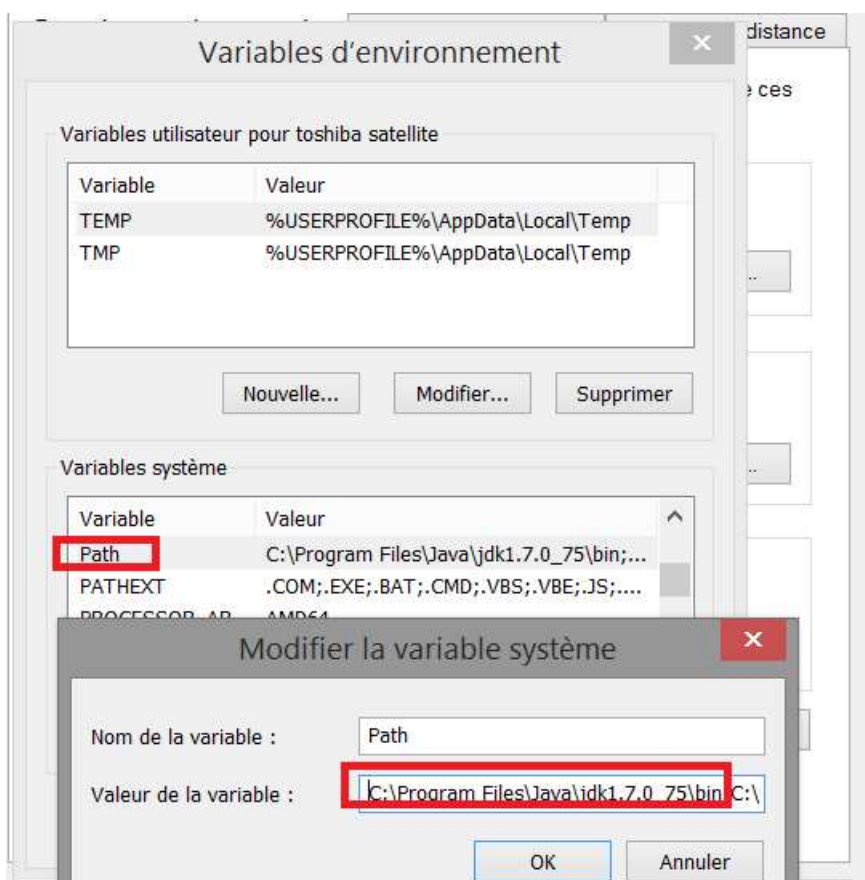
Pour vérifier si Java 1.7 est installée, lancer la commande DOS **java -version**, le résultat devrait être :

```
Microsoft Windows [version 6.3.9600]
(c) 2013 Microsoft Corporation. Tous droits réservés.

C:\Users\toshiba satellite>java -version
java version "1.7.0_75"
Java(TM) SE Runtime Environment (build 1.7.0_75-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.75-b04, mixed mode)

C:\Users\toshiba satellite>
```

- Ajouter le chemin **C:\java1.7\bin** (le chemin du dossier que vous avez choisi au début de l'installation) au début de la variable d'environnement **path** :



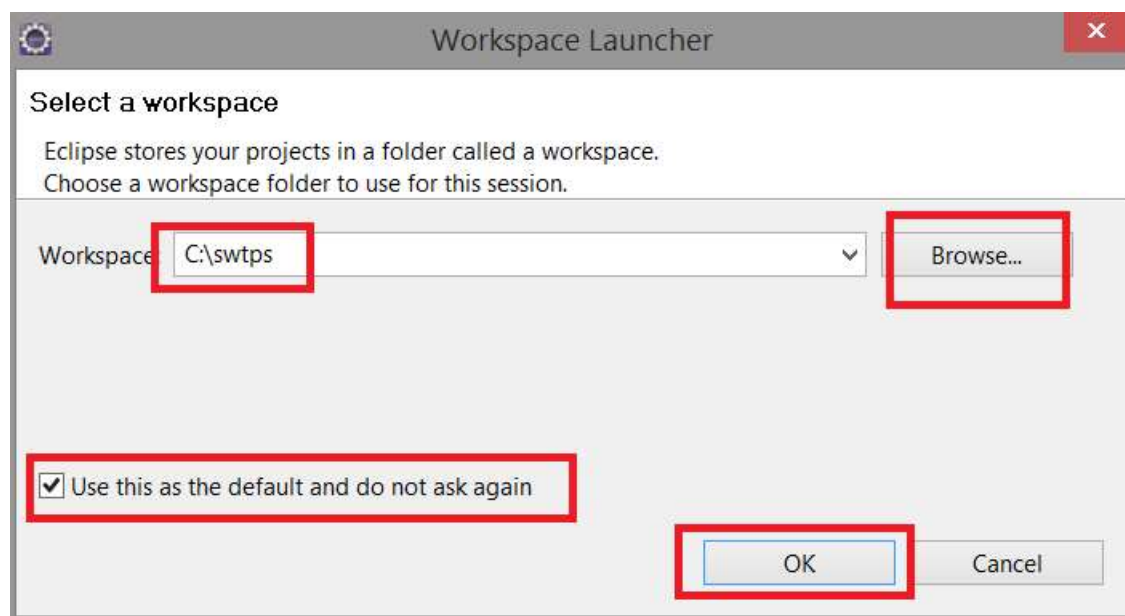
A-2 Installation de Eclipse Mars 4.5.1

Noter que Eclipse ne s'installe pas. Pour le lancer, il suffit de lancer son exe suivant :

C:\TI31320200A (C:) > formations > Dossier Services Web > tools > eclipse Kepler

Nom	Modifié le	Type	Taille
configuration	19/02/2016 21:19	Dossier de fichiers	
dropins	24/02/2014 06:05	Dossier de fichiers	
features	06/02/2016 19:37	Dossier de fichiers	
p2	06/02/2016 19:37	Dossier de fichiers	
plugins	06/02/2016 19:38	Dossier de fichiers	
readme	06/02/2016 19:38	Dossier de fichiers	
.eclipseproduct	06/02/2016 19:31	Fichier ECLIPSEPR...	1 Ko
artifacts.xml	06/02/2016 19:31	Document XML	245 Ko
eclipse.exe	06/02/2016 19:31	Application	305 Ko
eclipse.ini	06/02/2016 19:31	Paramètres de con...	1 Ko
eclipsesec.exe	06/02/2016 19:31	Application	18 Ko
epl-v10.html	06/02/2016 19:31	Chrome HTML Do...	17 Ko
notice.html	06/02/2016 19:31	Chrome HTML Do...	10 Ko

L'interface de **Eclipse Mars** suivante sera affichée :

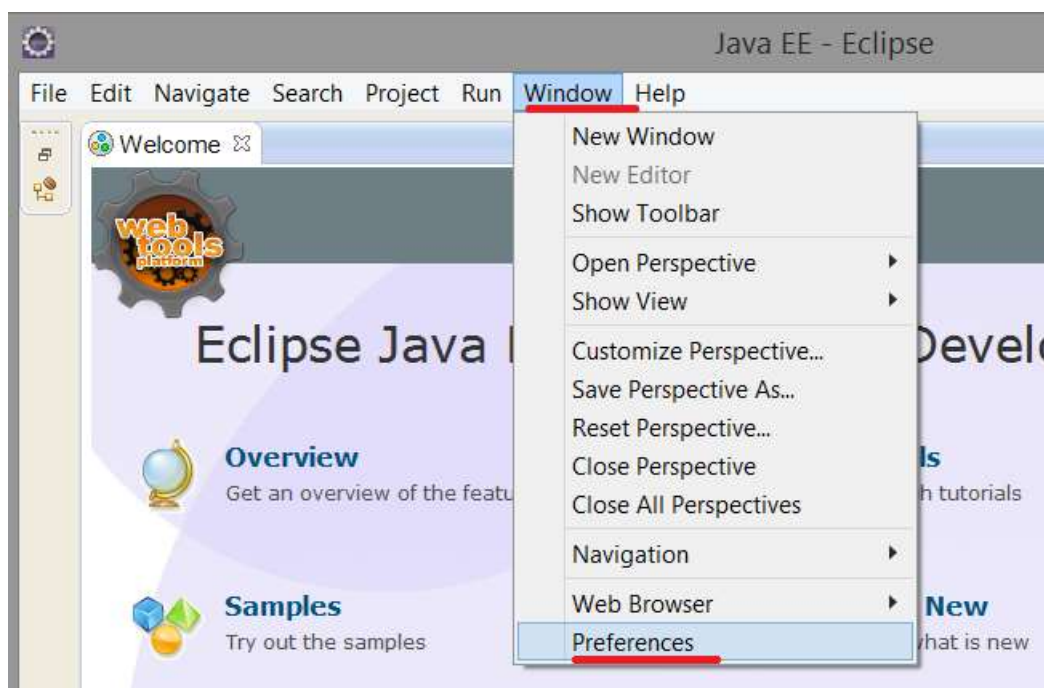


- Choisir le dossier dans lequel Eclipse créera les projets (votre **workspace**). Ensuite, cliquer sur **OK**. L'interface générale de **Eclipse Mars** est la suivante :

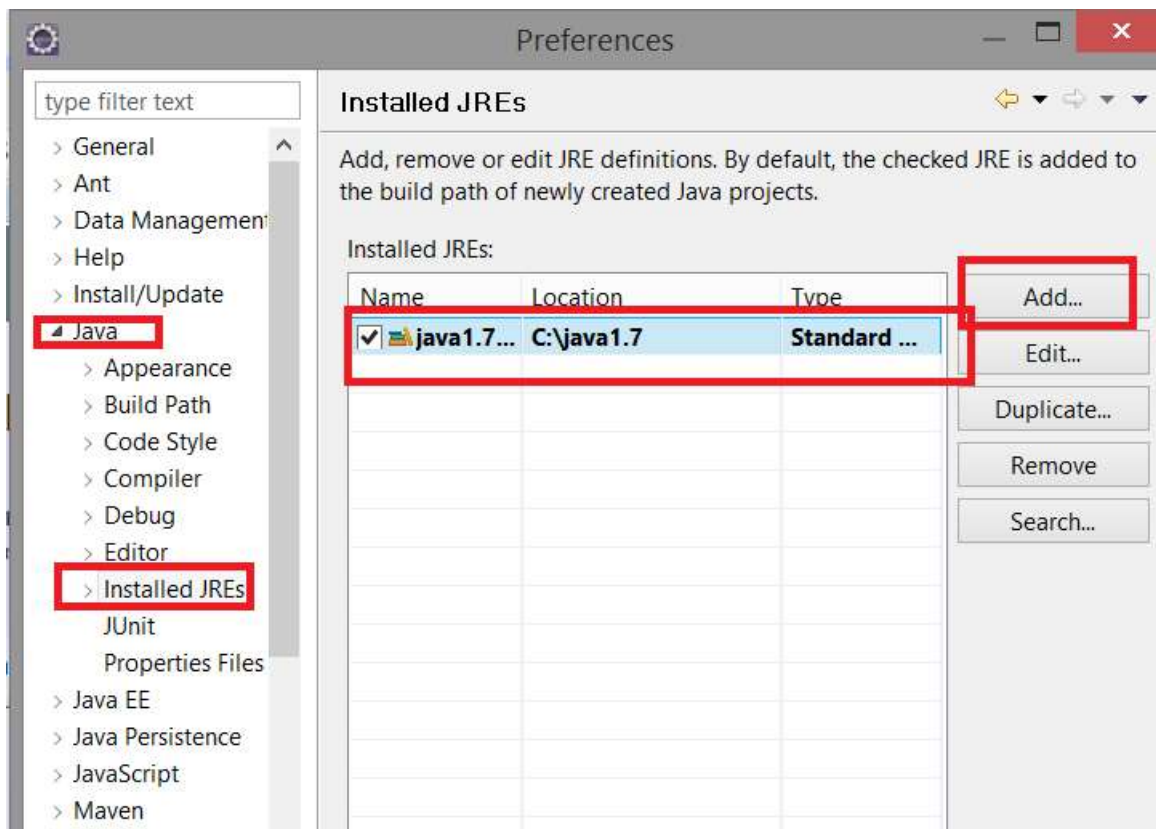


A-3 Configuration de Java (JDK et JRE) au niveau de Eclipse Mars

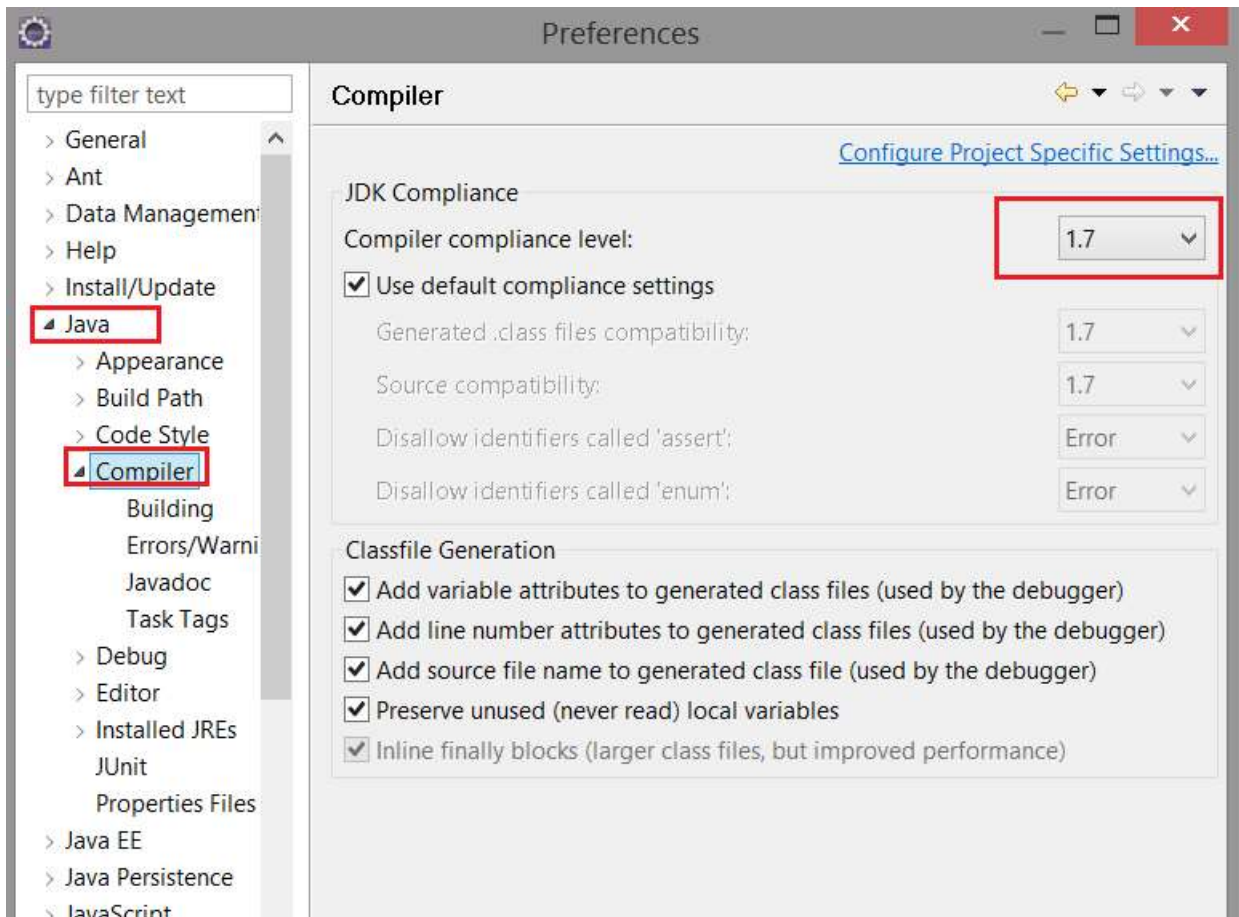
- Cliquer sur le menu suivant :



La fenêtre suivante sera affichée :

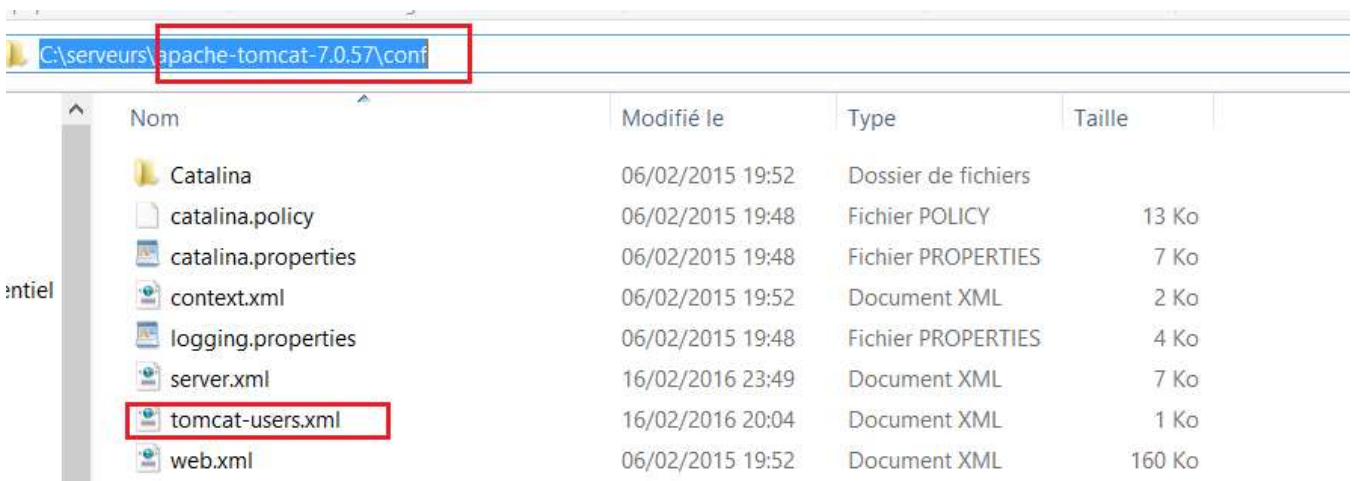


- Pour ajouter une autre JRE, il suffit de cliquer sur le bouton **Add** et préciser le chemin de l'installation de votre JRE.
- Pour configurer la version du compilateur, cliquer sur le menu suivant et préciser la version 1.7 :



A-4 Installation de Tomcat 7.0

Noter que Tomcat 7.0 ne s'installe pas. La première chose qu'il faut faire est de configurer la liste des utilisateurs. Editer le fichier **tomcat-users.xml** :



- Copier les lignes suivantes :

```
<?xml version="1.0" encoding="UTF-8"?>
<tomcat-users>
  <role rolename="manager-gui"/>
  <role rolename="manager-script"/>
```

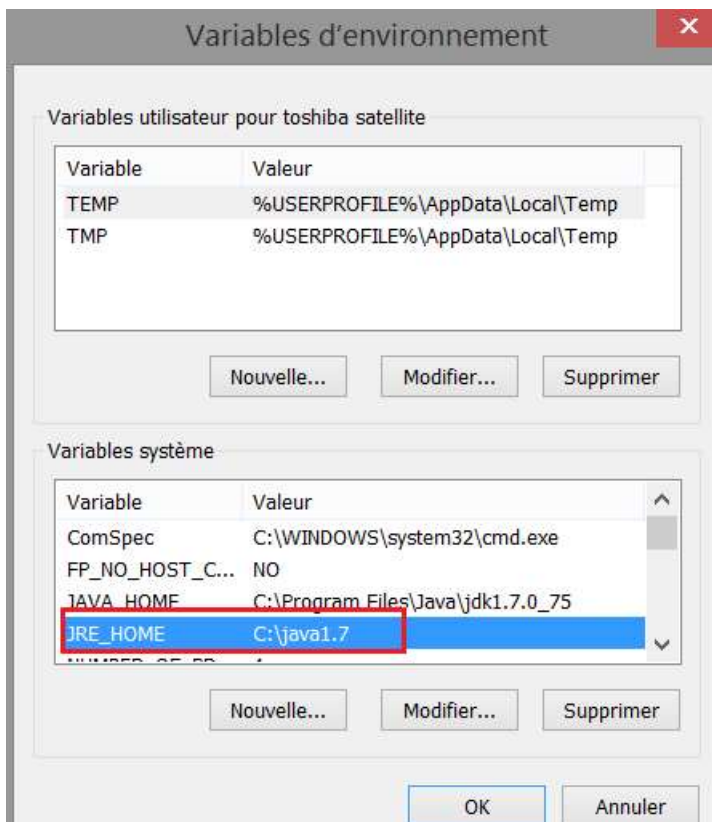
```

<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<role rolename="tomcat"/>
<role rolename="admin"/>
<user password="tomcat" roles="tomcat" username="tomcat"/>
<user password="admin" roles="admin,manager-gui,manager-
script,manager-jmx,manager-status" username="admin"/>
</tomcat-users>

```

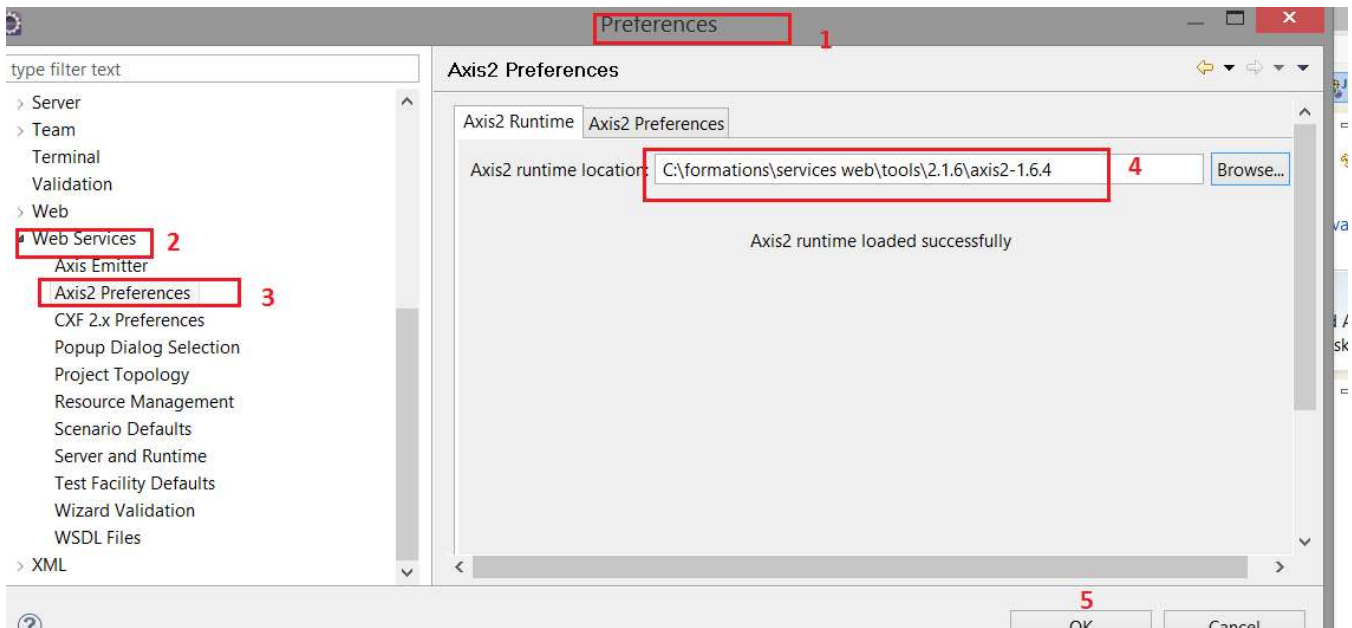
Ici, le compte **admin** dispose de tous les droits. Nous en aurons besoin afin d'administrer les ressources à déployer au niveau de **tomcat**.

- Créer la variable d'environnement **JAVA_HOME** comme suit :



A-4 Configuration de AXIS 2 version 1.6.4 dans Eclipse

Noter que **Axis2** version **1.6.4** est une implémentation du standard de communication **SOAP**. Afin de configurer **Axis2** au niveau de Eclipse, cliquer sur le menu **Preferences** et préciser le chemin de la distribution de **Axis2** comme le montre la fenêtre suivante :



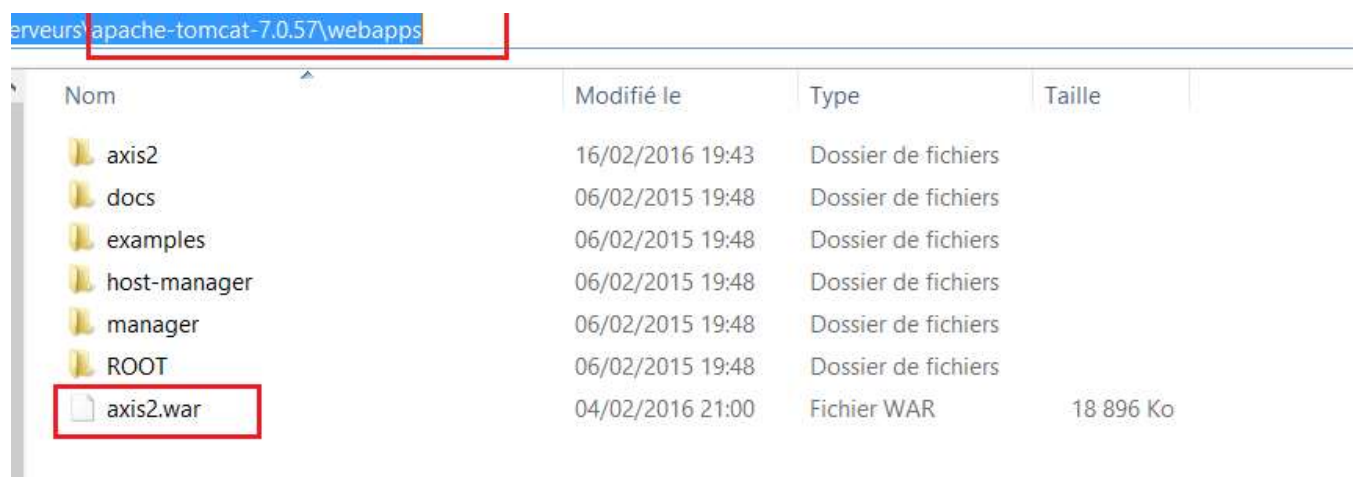
- Cliquer ensuite sur le bouton **OK**.

A-4 Déployer Axis2 dans Tomcat

Pour déployer **Axis2** au niveau de **Tomcat**, il suffit de copier le fichier **axis2.war** suivant :

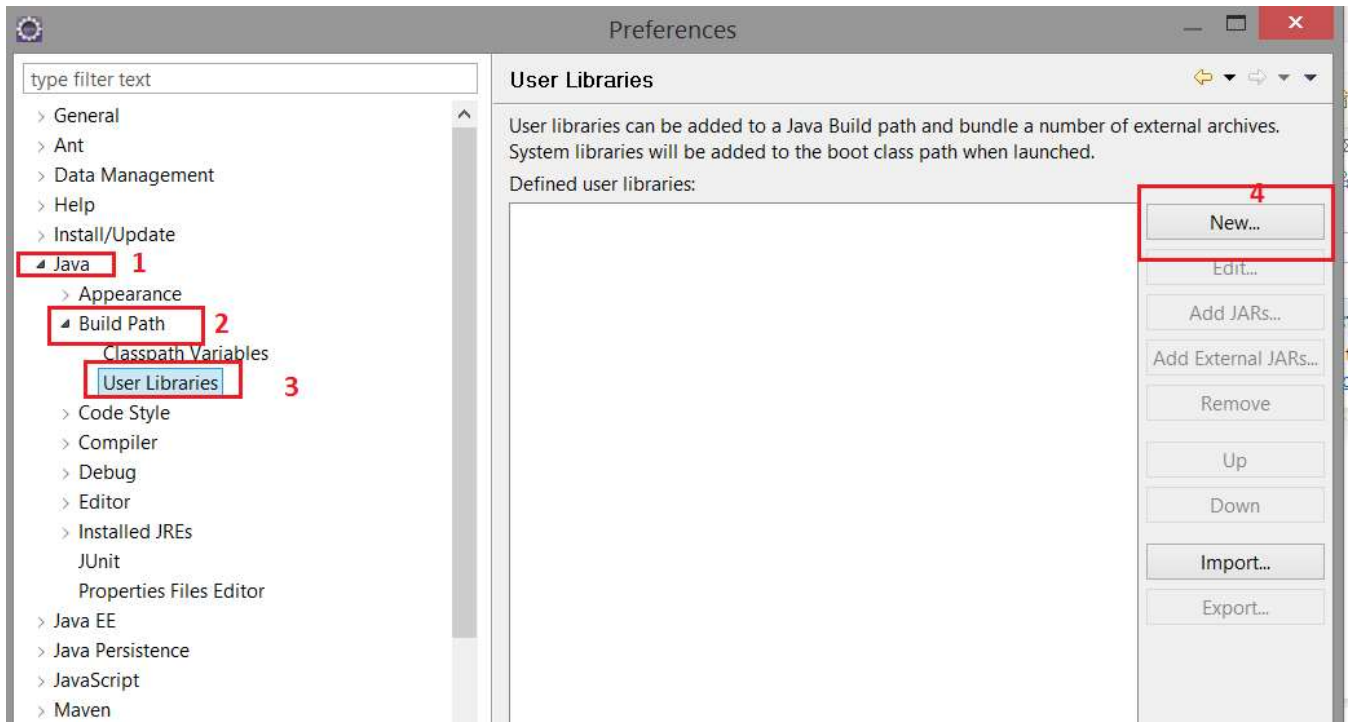


et le mettre dans le dossier **C:\serveurs\apache-tomcat-7.0.57\webapps** de Tomcat :

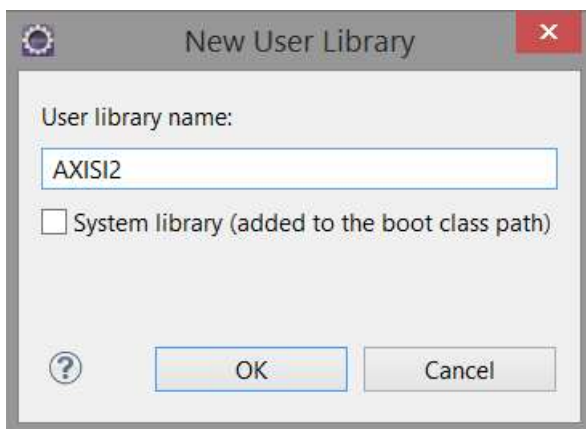


A-5 Création d'une Bibliothèque Utilisateur AXIS2 au niveau de Eclipse Mars

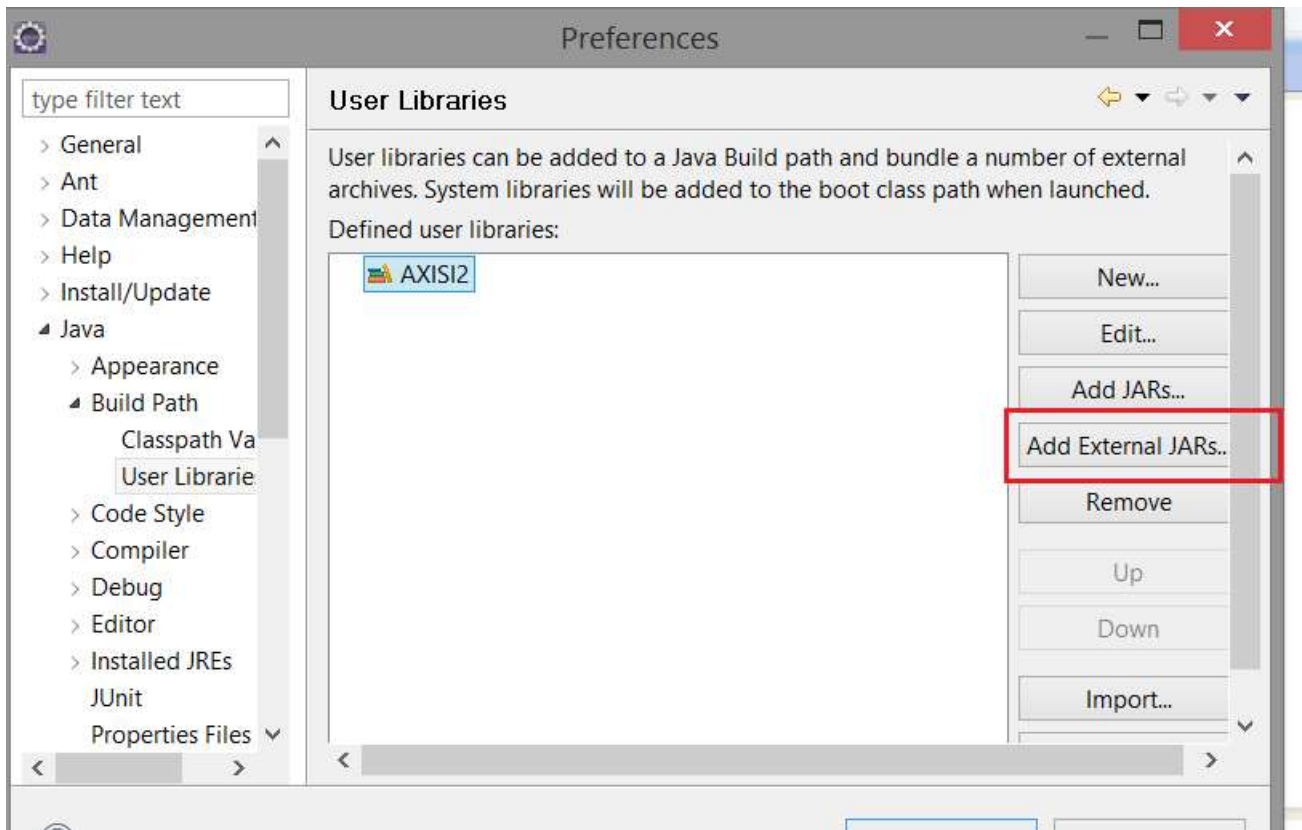
- Cliquer sur le menu Préférence de Eclipse. L'écran suivant sera affiché :



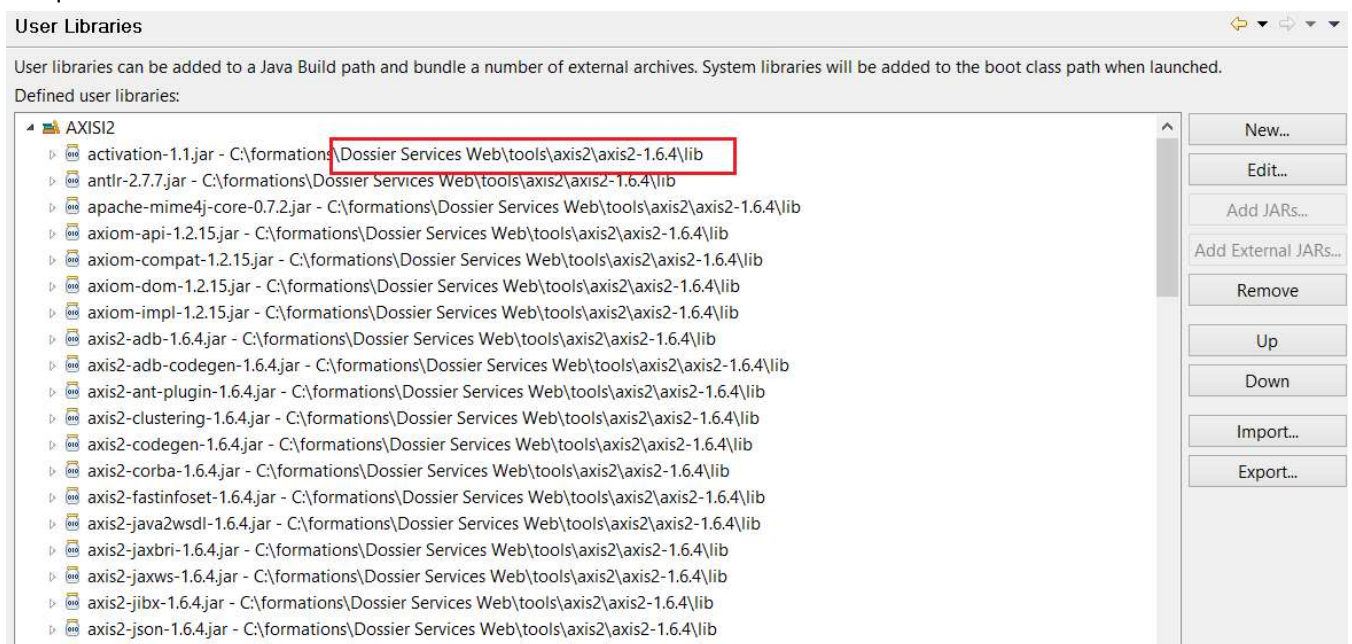
- Cliquer sur **New** et créer votre **User Library AXIS2** comme suit :



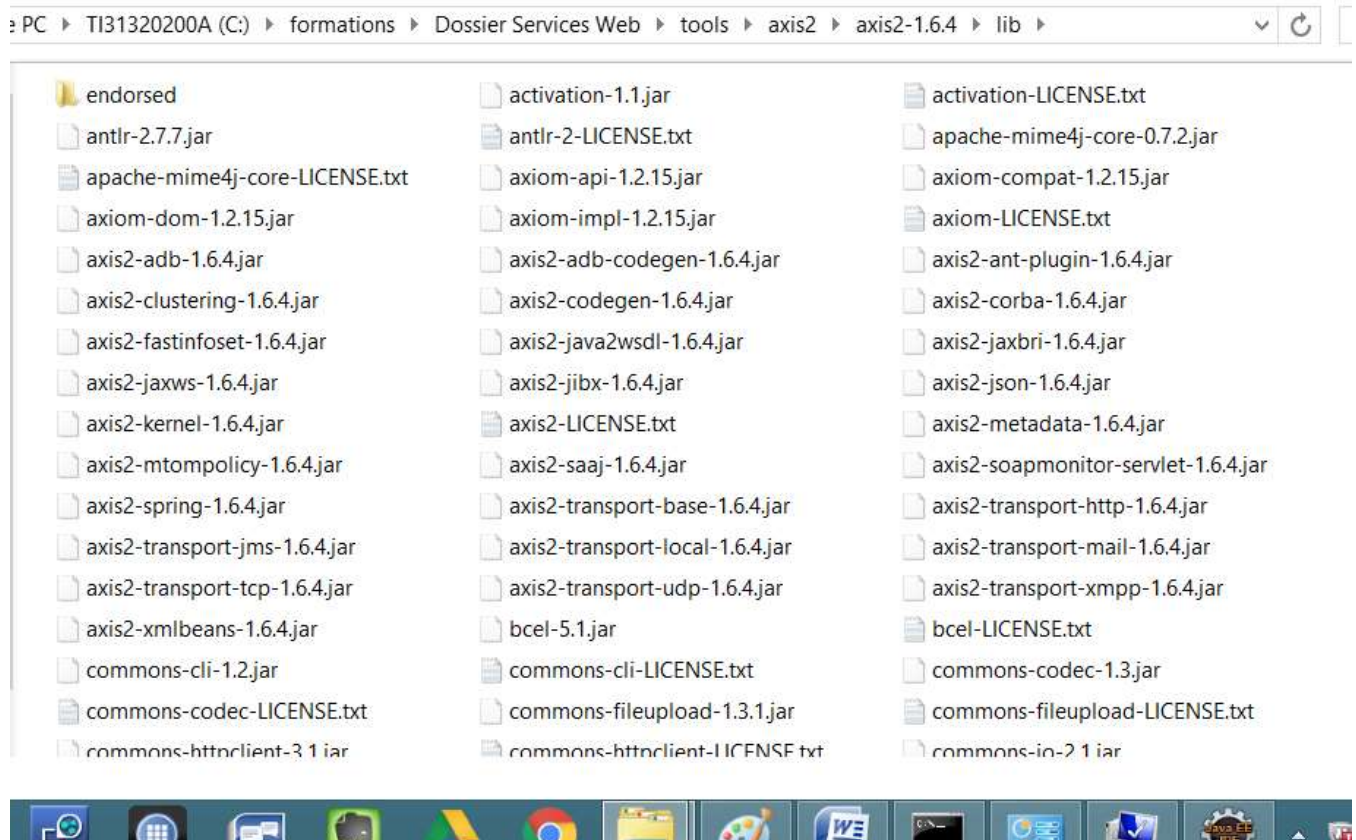
- Cliquer sur **OK**. L'écran suivant sera affiché :



- Cliquer sur **Add External JARs**. L'écran suivant sera affiché :

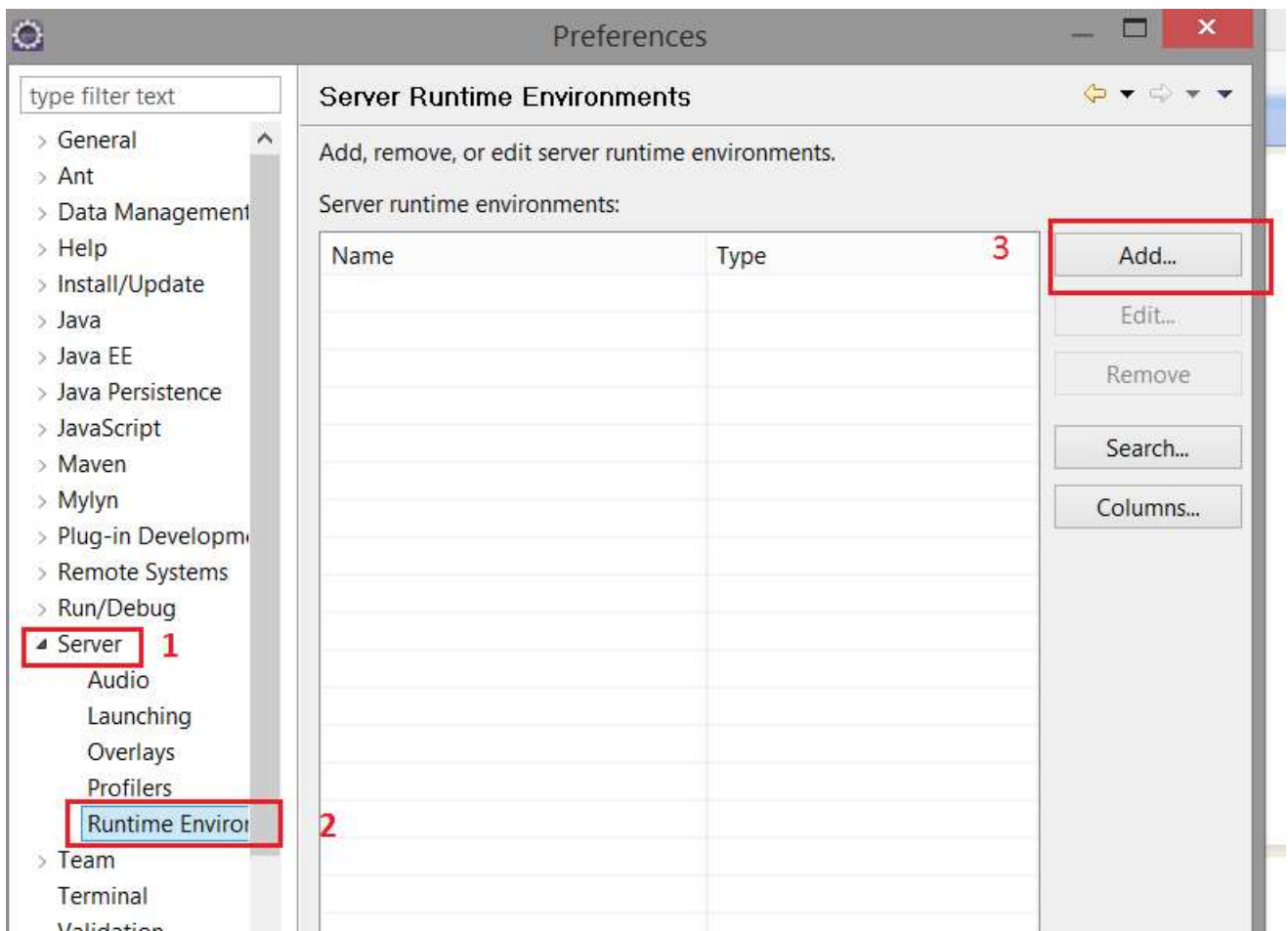


Le chemin des librairies de **Axis2** sont dans le dossier :
/Dossier Services Web/tools/axis2/axis2-1.6.4/lib :

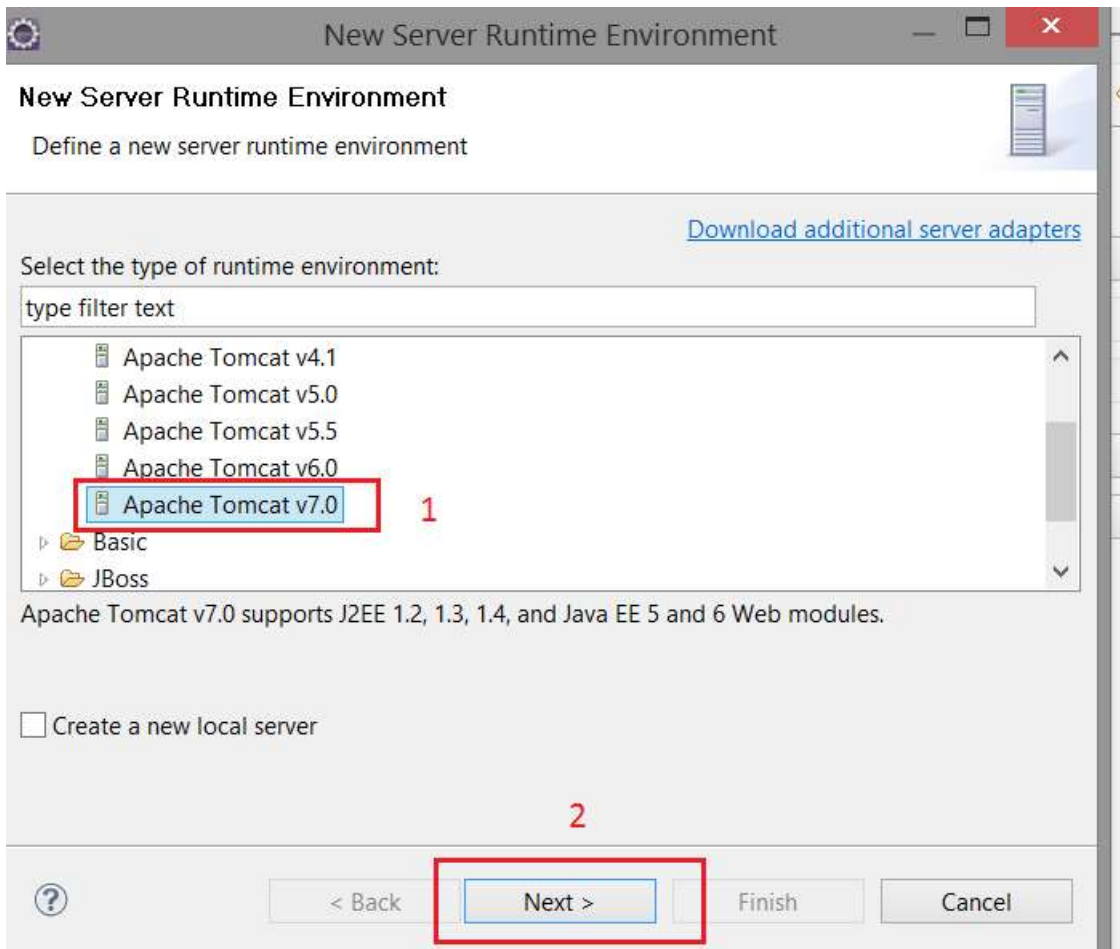


A-6 Configuration de Tomcat dans Eclipse

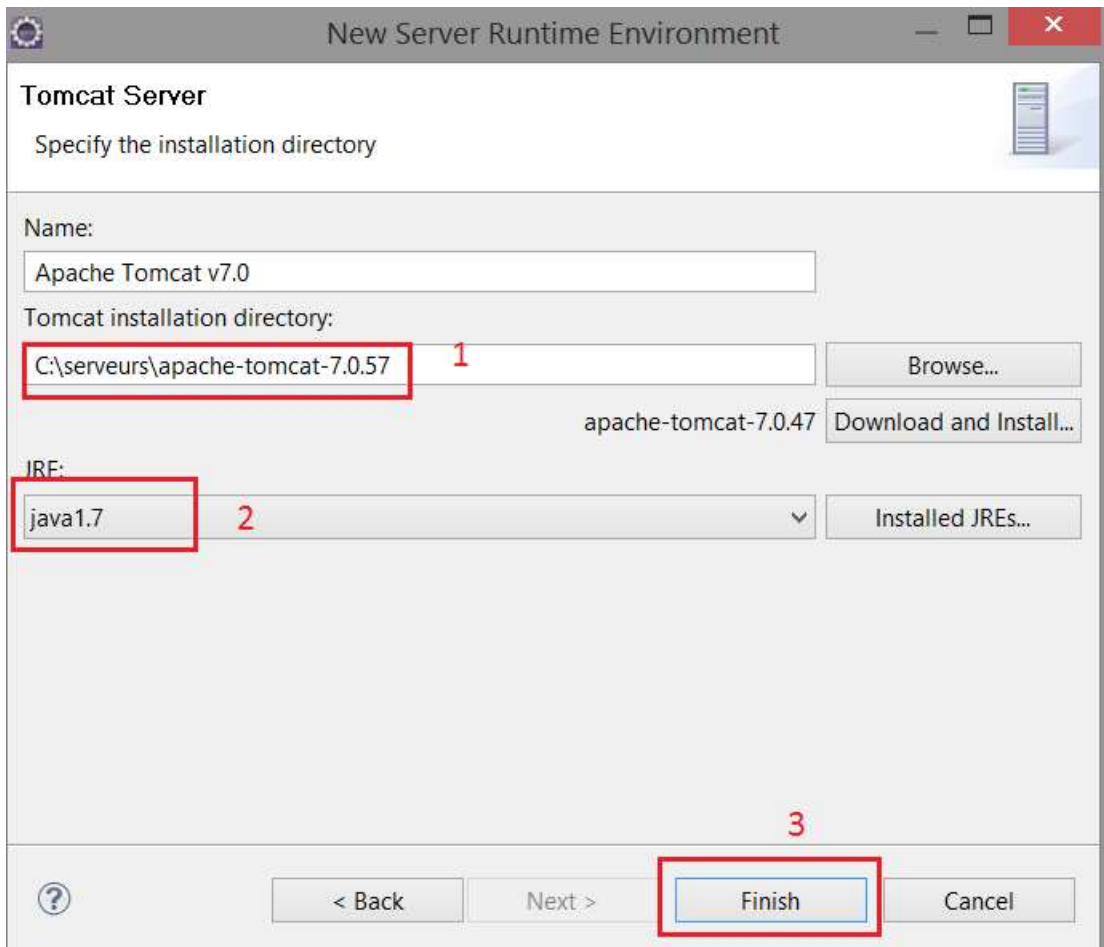
- Cliquer sur **Preferences**. Ensuite, cliquer sur **Server ==> Runtime Environment** comme le montre la fenêtre suivante :



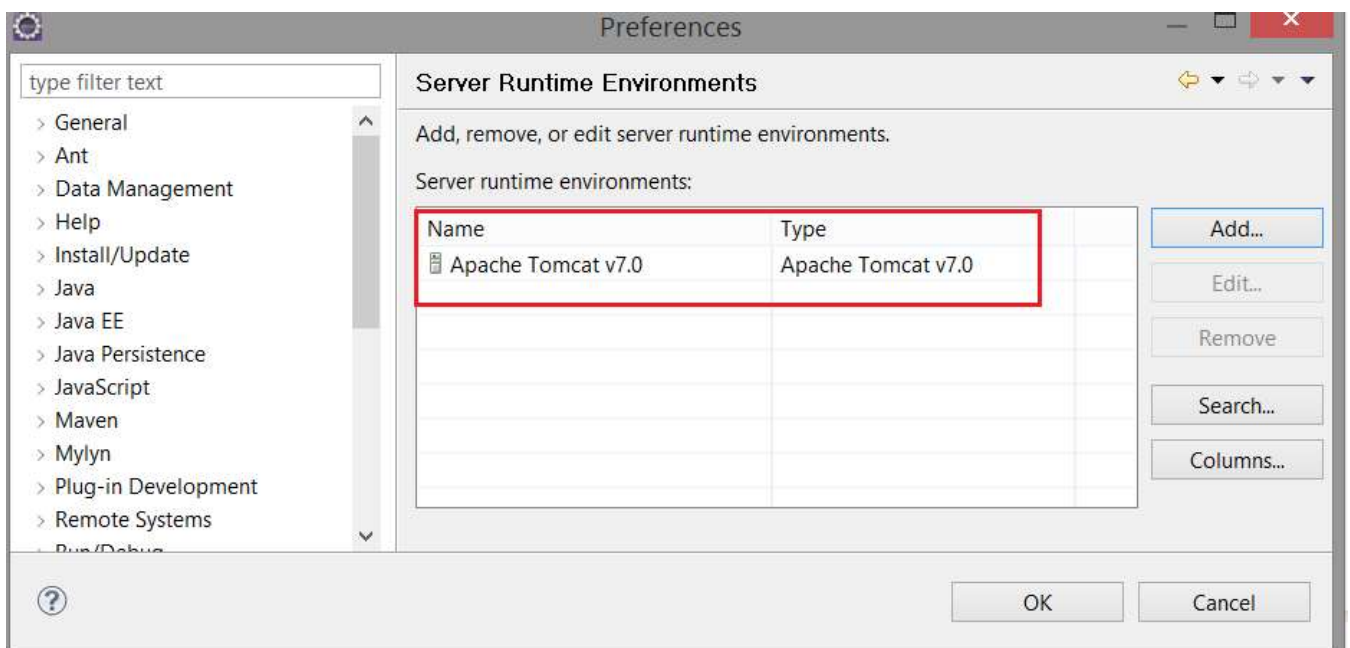
- Cliquer sur **Add**. L'écran suivant sera affiché :



- Choisir Apache **Tomcat v7.0**, puis cliquer sur **Next>**. L'écran suivant sera affiché :



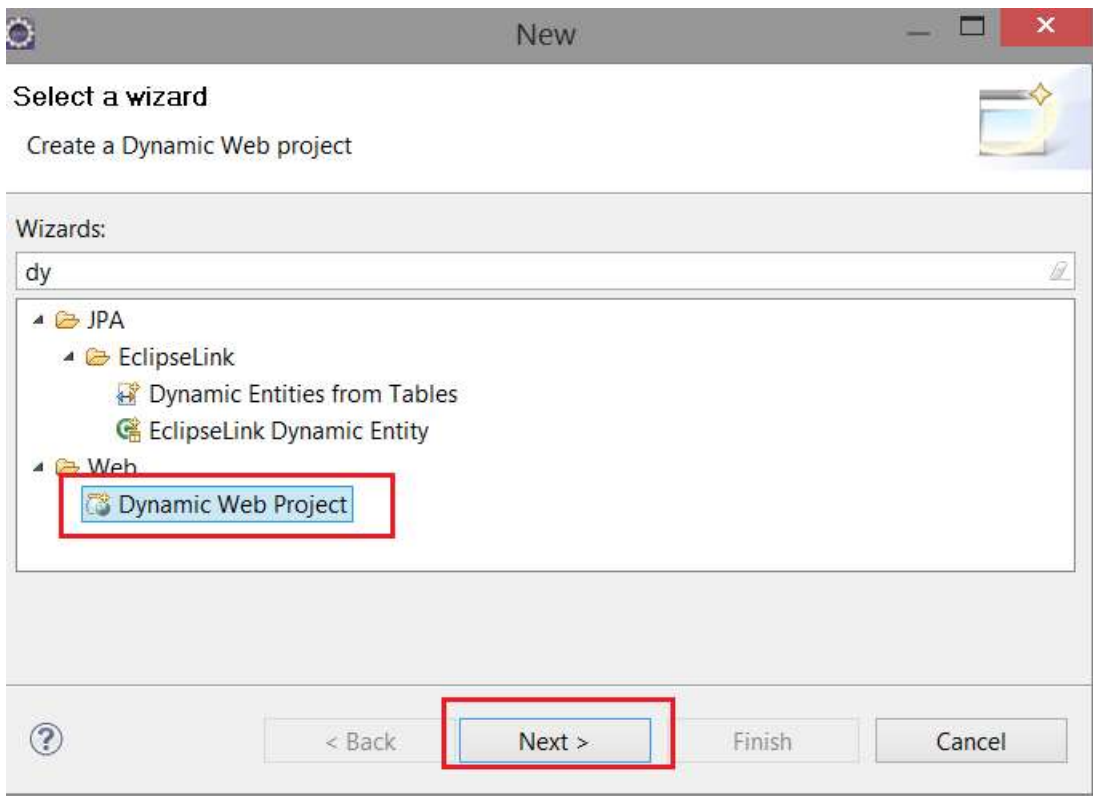
- Préciser le chemin de votre serveur **Tomcat** et préciser la **JRE** qui sera utilisé par **Tomcat** puis cliquer sur **Finish**.
Tomcat sera ajouté au niveau de Eclipse comme suit :



B- Développement du Service WEB (L'approche Bottom-Up)


Maintenant que l'environnement de développement est bien mis en place, nous allons commencer le développement de notre Service Web.

- Créer un nouveau projet web "**Dynamic Web Project**" comme suit :



donner comme nom à votre premier projet : MyFirstWebService

- Cliquer sur **Next>** :

 New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location

☒ Use default location

Location:

Target runtime

Apache Tomcat v7.0

Dynamic web module version

2.5

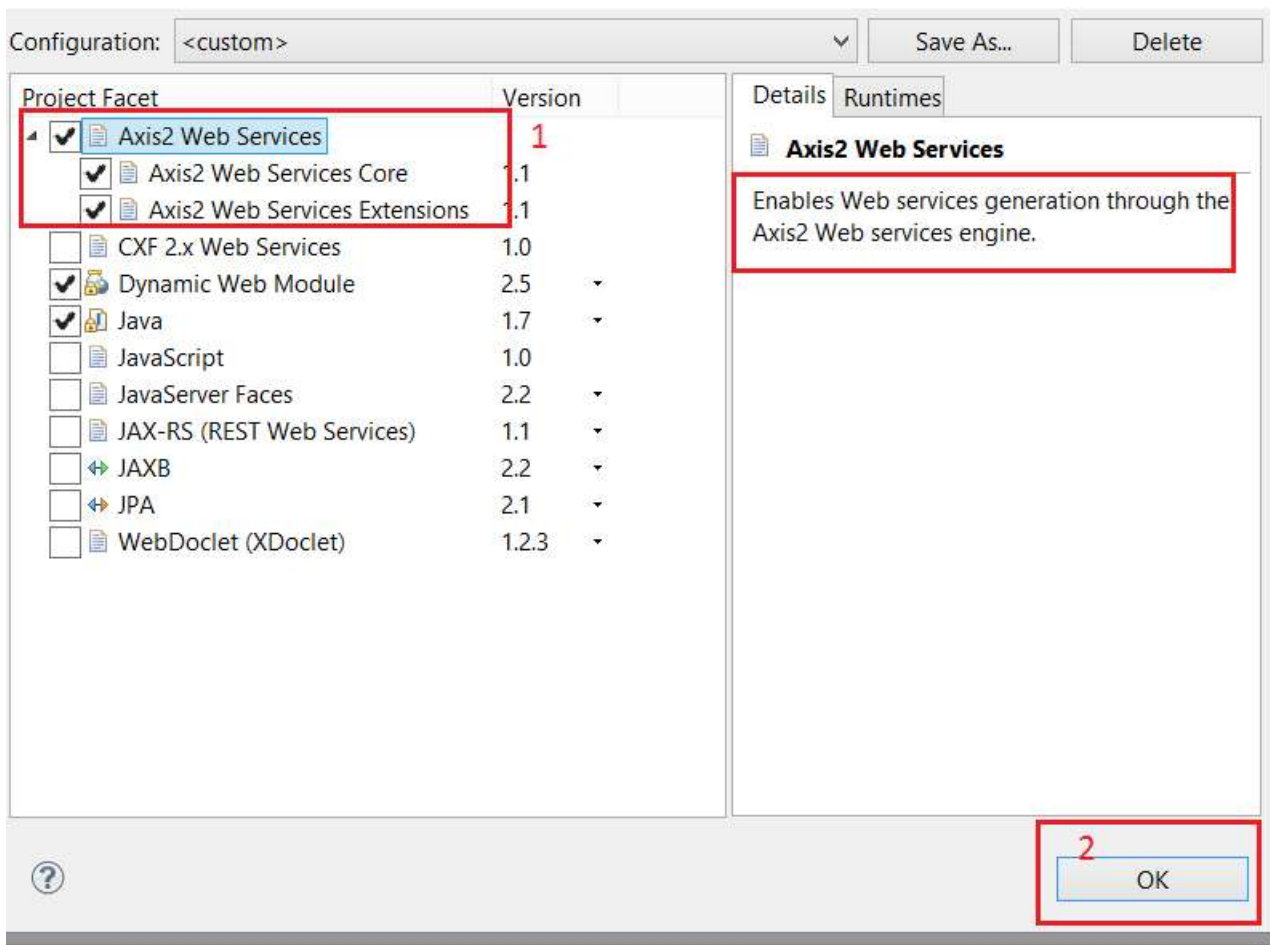
Configuration

<custom>

Hint: Get started quickly by selecting one of the pre-defined project configurations.

EAR membership

- Vérifier que la **Target Runtime** est bien **Apache Tomcat v7.0**.
- Cliquer sur le bouton **Modify ...**



- Cocher **Axis2 Web Services** puis cliquer sur OK.
- Cliquer ensuite sur **Next** puis sur **Finish**.
- Maintenant, créer la classe **Calculator** qui sera exposée au niveau de tomcat sous forme d'un Service Web.

Java Class
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ default ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

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

Le code de la classe **Calculator** est comme suit :

```
package ma.formation.sw;

public class Calculator {
    public double add(double a, double b) {
        return a+b;
    }

    public double subtract(double a, double b) {
        return a-b;
    }
    public double divide(double a, double b) {
        return a/b;
    }

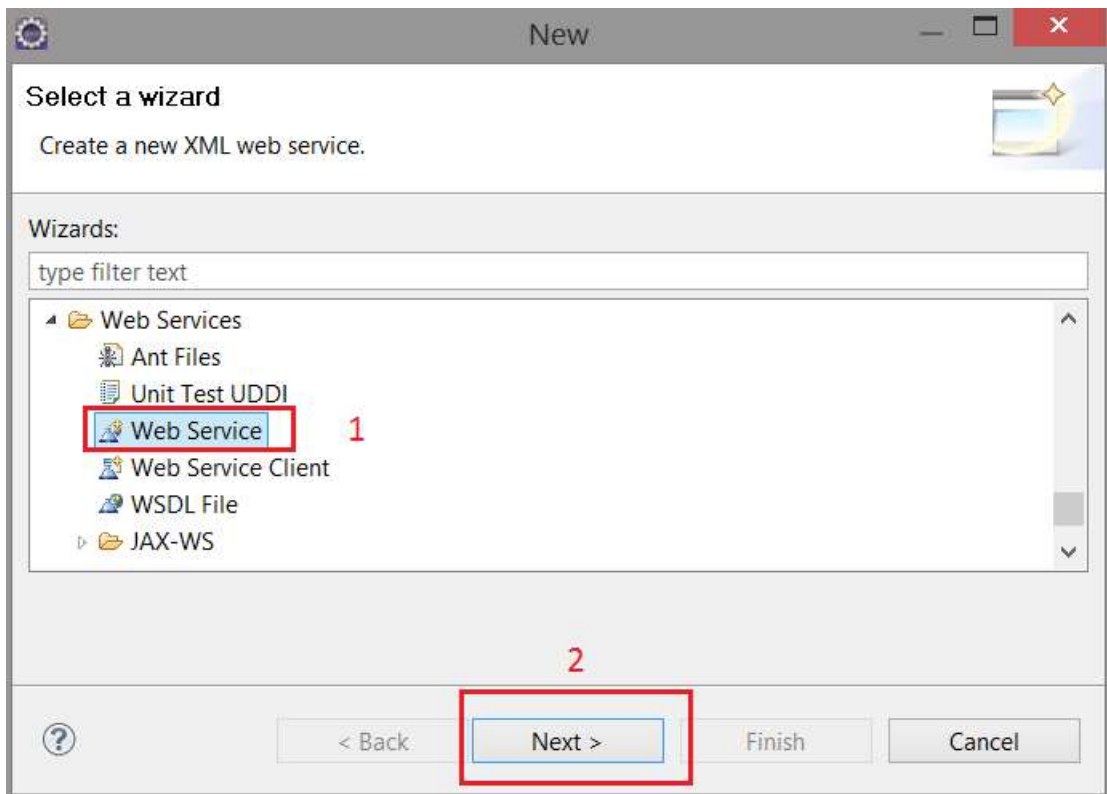
    public double multiply(double a, double b) {
        return a*b;
    }
}
```

La classe **Calculator** offre 04 services :

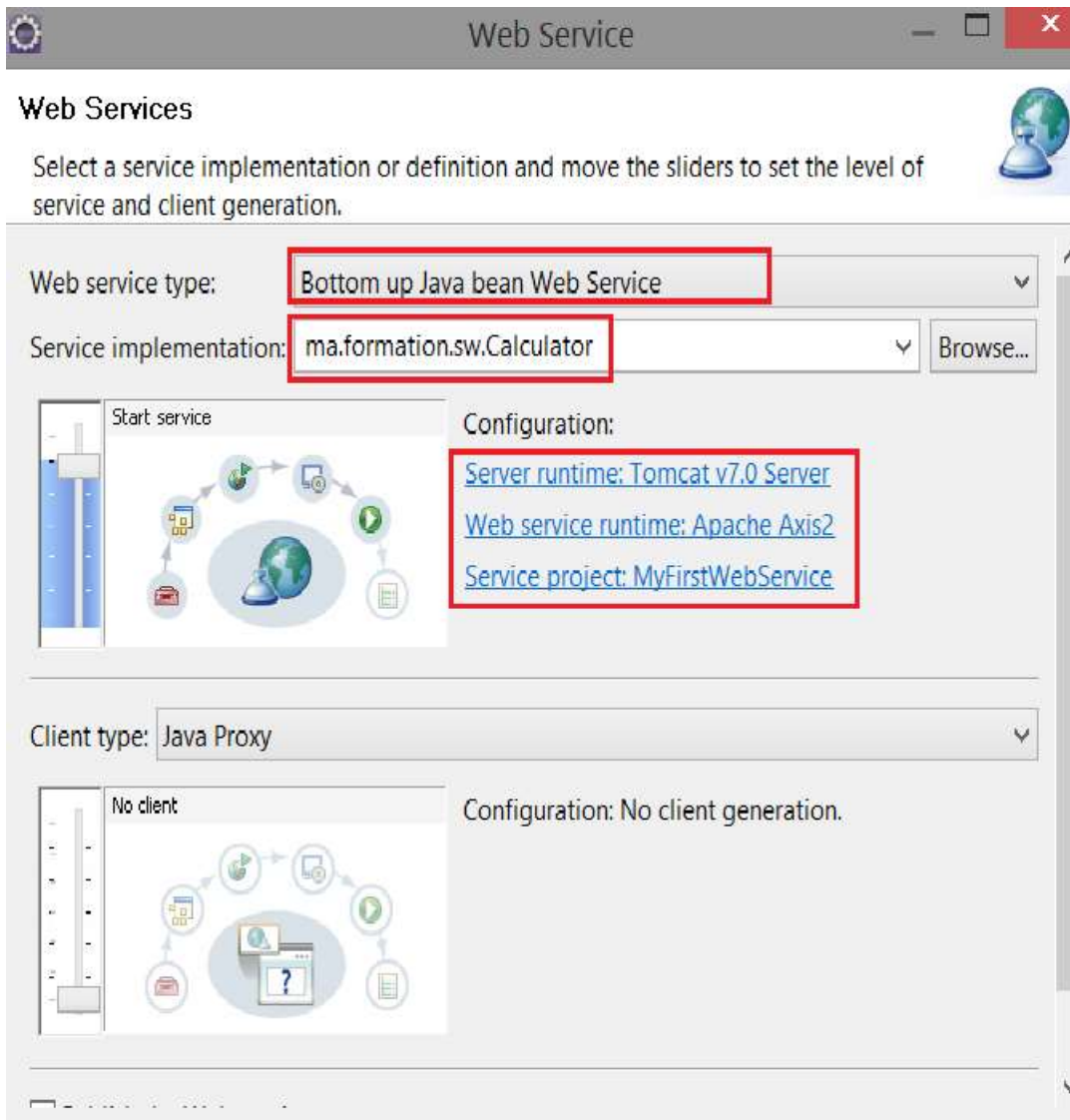
- ✓ add;
- ✓ subtract ;
- ✓ divide ;

✓ multiply.

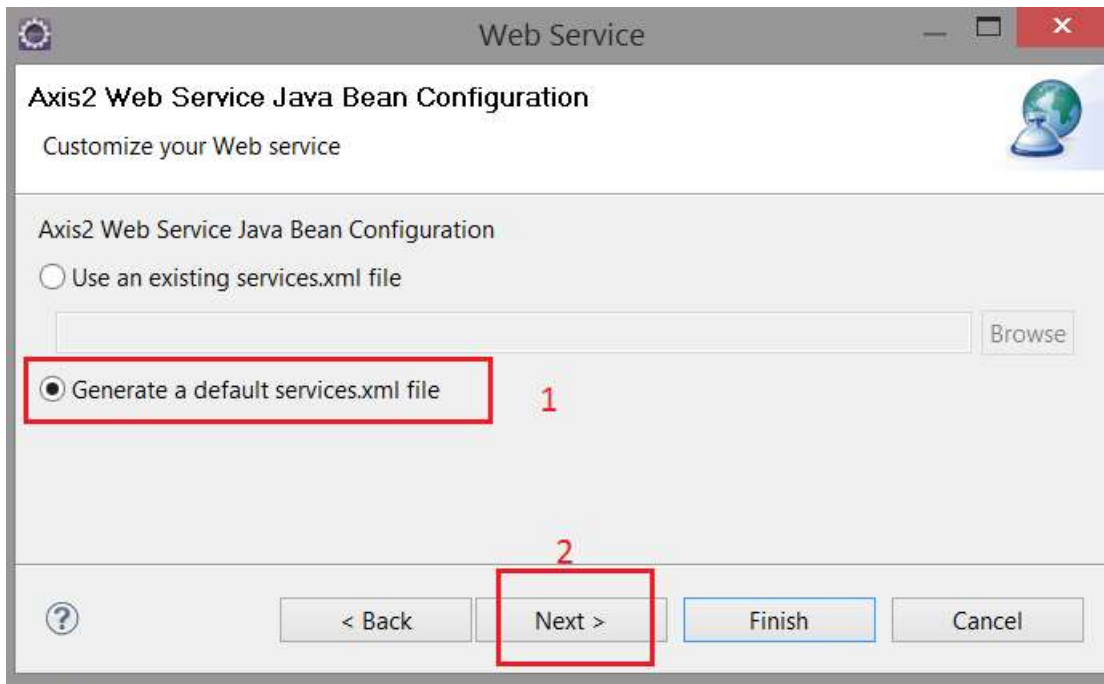
- Cliquer sur **File ==> New ==> Other...==> Web service** :



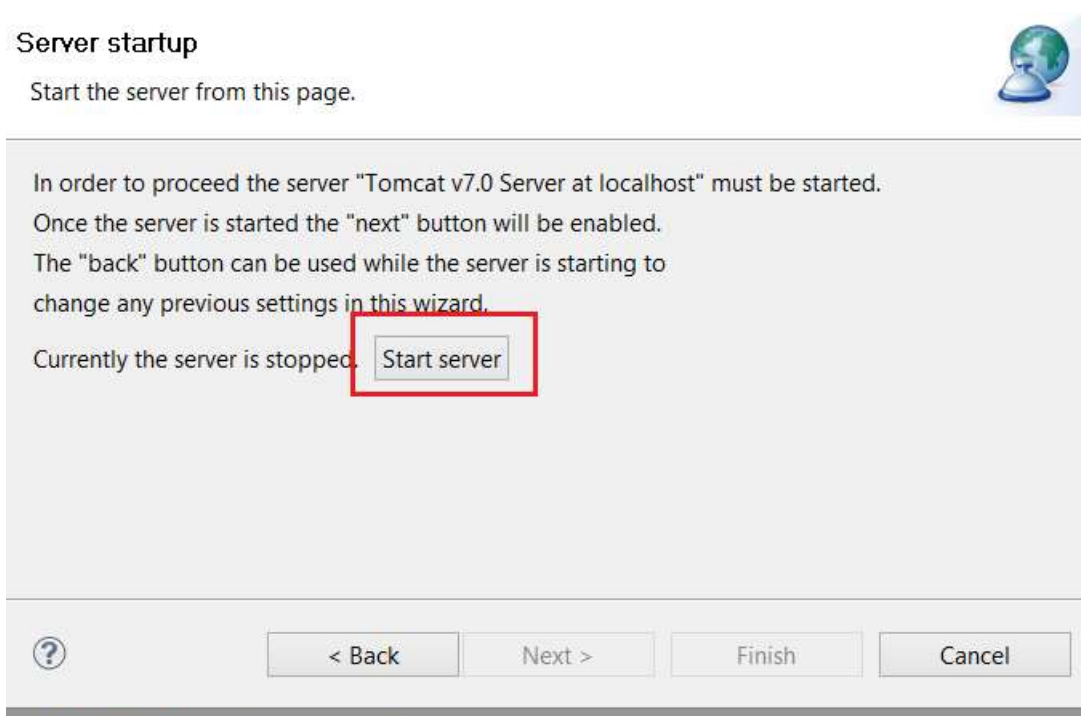
- Cliquer ensuite sur **Next>**. La fenêtre suivante est affichée :



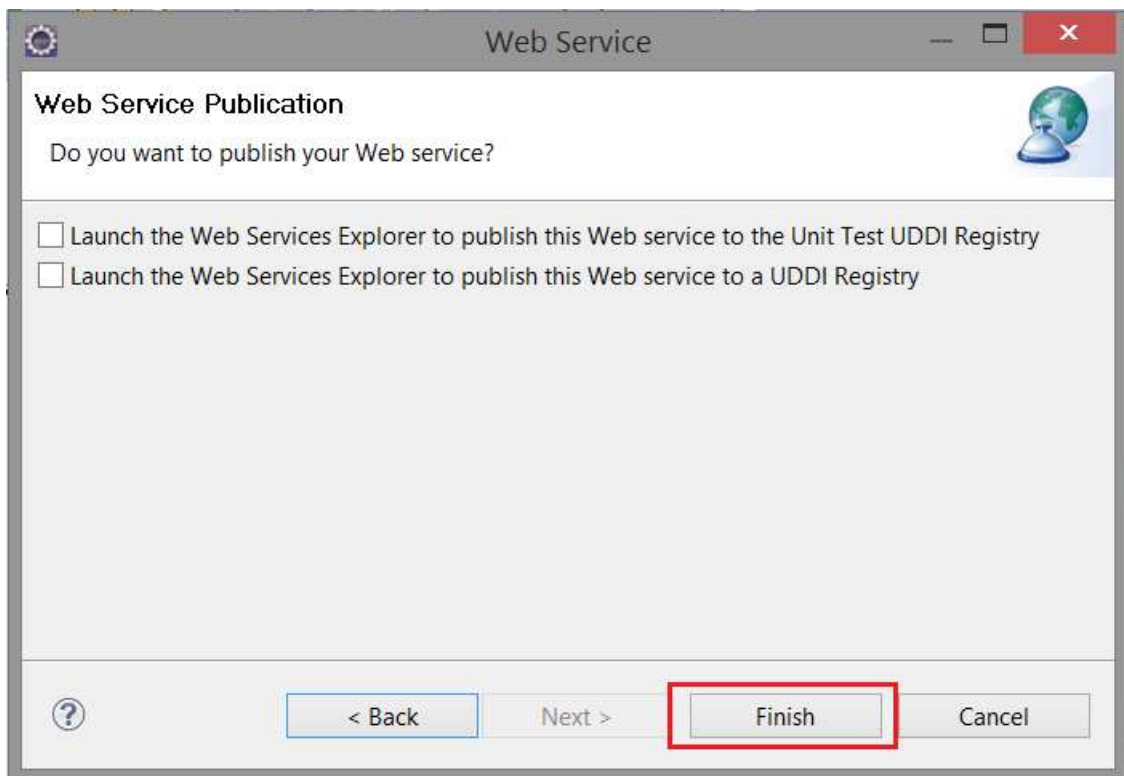
- Préciser l'approche "**Bottom-up**";
- Préciser le Server Runtime : **Tomcat 7** ;
- Préciser le Web service Runtime : Axis2 ;
- Préciser le Service Projet : **MyFirstWebService**.
- Cliquer ensuite sur **Next**. L'écran suivant sera affiché :



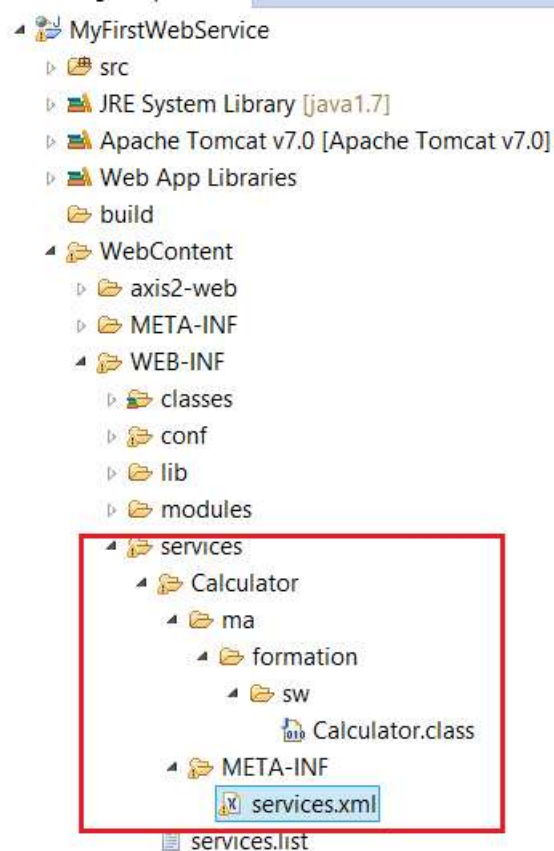
- Cliquer ensuite sur **Next>**. L'écran suivant sera affiché :



- Appuyer sur le bouton **Start server** puis cliquer sur **Next**. L'écran suivant sera affiché :



- Cliquer sur **Finish**. Vérifier que l'arborescence suivante a été créée automatiquement :



- Vérifier que le contenu du fichier services.xml:

```
services.xml
<?xml version="1.0" encoding="UTF-8"?>
<service name="Calculator">
  <Description>
    Please Type your service description here
  </Description>
  <messageReceivers>
    <messageReceiver mep="http://www.w3.org/2004/08/wsdl/in-only"
      class="org.apache.axis2.rpc.receivers.RPCInOnlyMessageReceiver" />
    <messageReceiver mep="http://www.w3.org/2004/08/wsdl/in-out"
      class="org.apache.axis2.rpc.receivers.RPCMessageReceiver" />
  </messageReceivers>
  <parameter name="ServiceClass" locked="false">ma.formation.sw.Calculator</parameter>
</service>
```

- Pour vérifier si le Service Web "**Calculator**" a été bien déployé par **Axis2** au niveau de **Tomcat**, lancer le lien **<http://localhost:8080/MyFirstWebService/services/listServices>** :



Available services

Version

Service Description : Version

Service EPR : <http://localhost:8080/MyFirstWeb Service/services/Version>

Service Status : Active

Available Operations

- getVersion

Calculator

Service Description : Please Type your service description here

Service EPR : <http://localhost:8080/MyFirstWeb Service/services/Calculator>

Service Status : Active

Available Operations

- add
- divide

- Pour visualiser le code WSDL décrivant le service web (**couche Définition** de l'architecture des Services Web), cliquer sur le lien **Calculator** ci-dessus :

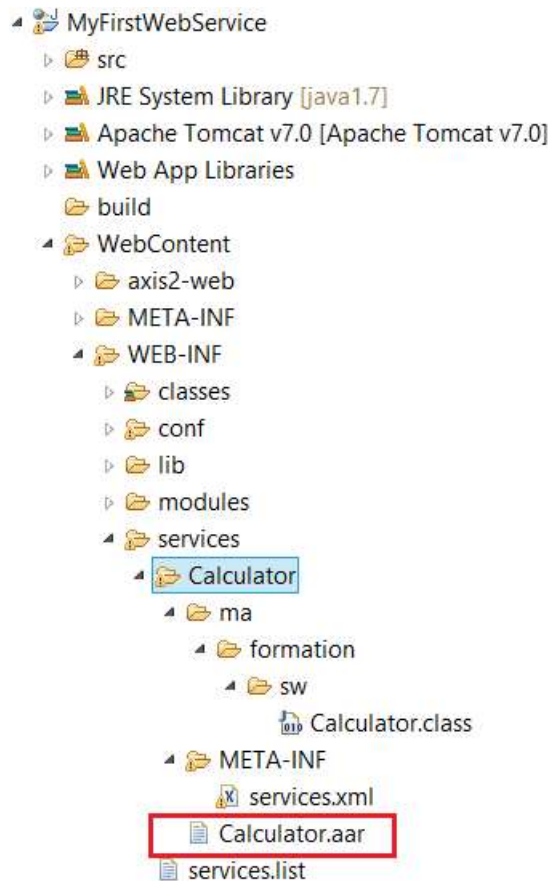
Le contenu du fichier WSDL est :

```
<?xml version='1.0' encoding='utf-8' ?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:ns1="http://org.apache.axis2/xsd" xmlns:ns="http://sw.formation.ma"
  xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
  targetNamespace="http://sw.formation.ma">
  <wsdl:documentation>Please Type your service description here</wsdl:documentation>
  <wsdl:types>...</wsdl:types>
  <wsdl:message name="addRequest">...</wsdl:message>
  <wsdl:message name="addResponse">...</wsdl:message>
  <wsdl:message name="divideRequest">...</wsdl:message>
  <wsdl:message name="divideResponse">...</wsdl:message>
  <wsdl:message name="multiplyRequest">...</wsdl:message>
  <wsdl:message name="multiplyResponse">...</wsdl:message>
  <wsdl:message name="subtractRequest">...</wsdl:message>
  <wsdl:message name="subtractResponse">...</wsdl:message>
  <wsdl:part name="parameters" element="ns:subtractResponse"/>
  </wsdl:message>
  <wsdl:portType name="CalculatorPortType">...</wsdl:portType>
  <wsdl:binding name="CalculatorSoap11Binding" type="ns:CalculatorPortType">...</wsdl:binding>
  <wsdl:binding name="CalculatorSoap12Binding" type="ns:CalculatorPortType">...</wsdl:binding>
  <wsdl:binding name="CalculatorHttpBinding" type="ns:CalculatorPortType">...</wsdl:binding>
  <wsdl:service name="Calculator">
    <wsdl:port name="CalculatorHttpSoap11Endpoint" binding="ns:CalculatorSoap11Binding">
      <soap:address location="http://localhost:8080/MyFirstWebService/services/Calculator.CalculatorHttpSoap11Endpoint/" />
    </wsdl:port>
    <wsdl:port name="CalculatorHttpSoap12Endpoint" binding="ns:CalculatorSoap12Binding">
      <soap12:address location="http://localhost:8080/MyFirstWebService/services/Calculator.CalculatorHttpSoap12Endpoint/" />
    </wsdl:port>
    <wsdl:port name="CalculatorHttpEndpoint" binding="ns:CalculatorHttpBinding">
      <http:address location="http://localhost:8080/MyFirstWebService/services/Calculator.CalculatorHttpEndpoint/" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

- Maintenant, et afin de ne pas utiliser Eclipse à chaque fois que vous voulez déployer votre Service Web, nous allons créer le fichier **.aar (Axis Archive Ressource)**. Pour ceci :

- Lancer l'invite de commande DOS ;
- Aller au chemin C:\swtps\MyFirstWebService\WebContent\WEB-INF\services\Calculator.
- Lancer la commande : **jar cvf Calculator.aar ma META-INF**. Le fichier **Calculator.aar** sera créé :

```
C:\swtps\MyFirstWebService\WebContent\WEB-INF\services\Calculator>jar cvf Calculator.aar ma META-INF
manifeste ajouté
ajout : ma/(entr  e = 0) (sortie = 0)(stockage : 0 %)
ajout : ma/formation/(entr  e = 0) (sortie = 0)(stockage : 0 %)
ajout : ma/formation/sw/(entr  e = 0) (sortie = 0)(stockage : 0 %)
ajout : ma/formation/sw/Calculator.class(entr  e = 664) (sortie = 318)(compression : 52 %)
entr  e META-INF/ ignor  e
ajout : META-INF/services.xml(entr  e = 502) (sortie = 241)(compression : 51 %)
C:\swtps\MyFirstWebService\WebContent\WEB-INF\services\Calculator>
```



- Dans la vue **Server** au niveau d'Eclipse, arrêter **Tomcat**.
- Lancer le fichier **startup.bat** (C:\serveurs\apache-tomcat-7.0.57\bin) :

```

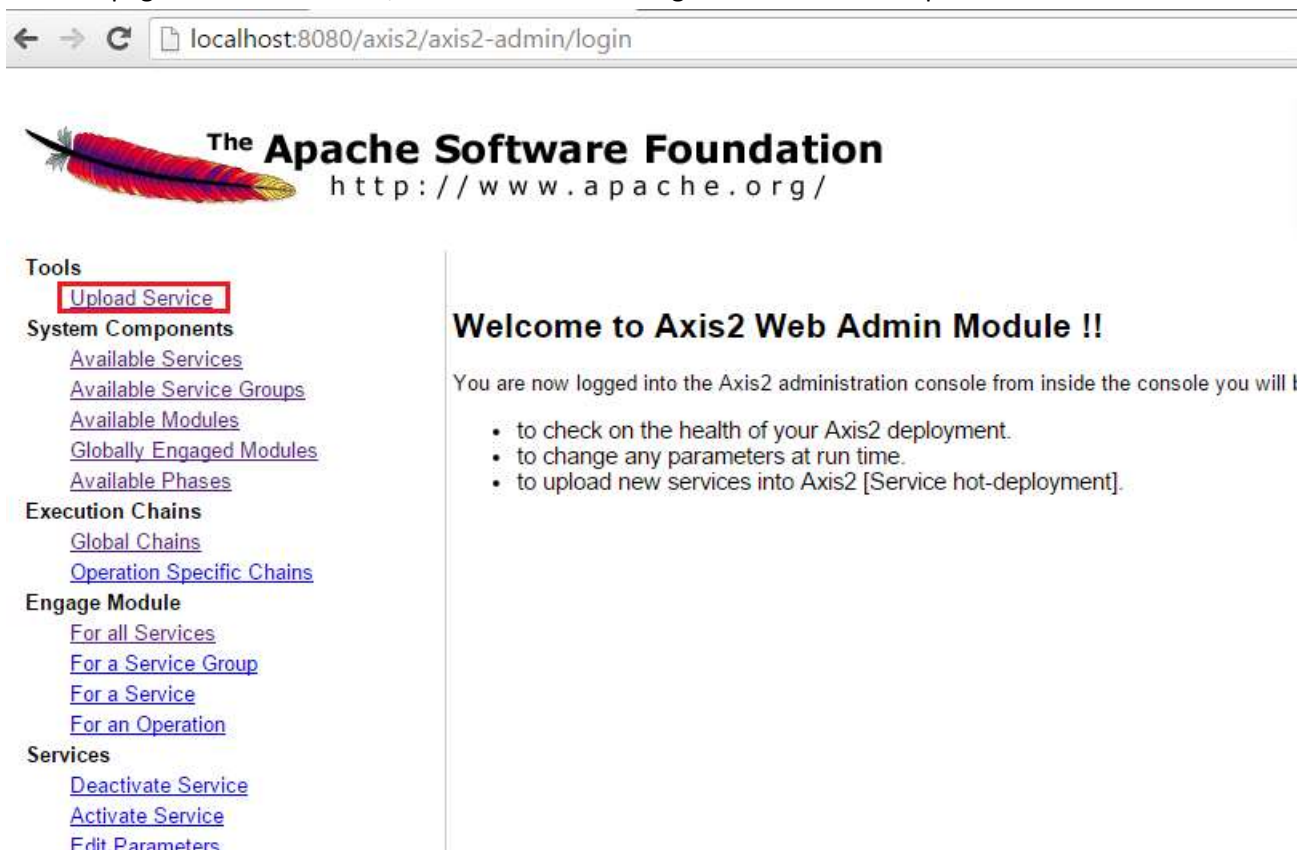
INFOS: D ploiement du r pertoire C:\serveurs\apache-tomcat-7.0.57\webapps\manager
de l'application web
f vr. 19, 2016 11:26:51 PM org.apache.catalina.startup.HostConfig deployDirector
y
INFOS: Deployment of web application directory C:\serveurs\apache-tomcat-7.0.57\
webapps\manager has finished in 70 ms
f vr. 19, 2016 11:26:51 PM org.apache.catalina.startup.HostConfig deployDirector
y
INFOS: D ploiement du r pertoire C:\serveurs\apache-tomcat-7.0.57\webapps\ROOT d
e l'application web
f vr. 19, 2016 11:26:51 PM org.apache.catalina.startup.HostConfig deployDirector
y
INFOS: Deployment of web application directory C:\serveurs\apache-tomcat-7.0.57\
webapps\ROOT has finished in 44 ms
f vr. 19, 2016 11:26:51 PM org.apache.coyote.AbstractProtocol start
INFOS: Starting ProtocolHandler ["http-apr-8080"]
f vr. 19, 2016 11:26:51 PM org.apache.coyote.AbstractProtocol start
INFOS: Starting ProtocolHandler ["ajp-apr-8009"]
f vr. 19, 2016 11:26:51 PM org.apache.catalina.startup.Catalina start
INFOS: Server startup in 5318 ms

```


- Lancer le lien : ***http://localhost:8080/axis2***. La page web Home de AXIS2 sera affichée :



- Cliquer ensuite sur le bouton **Administration** pour déployer le service web **Calculator.aar** :
- Dans la page d'authentification, saisir **admin** comme login et **admin** comme password.



- Cliquer sur le lien **Upload Service** ci-dessus :

[Upload Service](#)
[System Components](#)
[Available Services](#)
[Available Service Groups](#)
[Available Modules](#)
[Globally Engaged Modules](#)
[Available Phases](#)
[Execution Chains](#)
[Global Chains](#)
[Operation Specific Chains](#)
[Engage Module](#)
[For all Services](#)
[For a Service Group](#)
[For a Service](#)
[For an Operation](#)
[Services](#)

Upload an Axis Service Archive File

You can upload a packaged Axis2 service from this page in two small steps.

- Browse to the location and select the axis service archive file you wish to upload
- Click "Upload" button

Simple as that!

File Calculator.aar successfully uploaded

Service archive 1

2

Hot deployment of new service archives is enabled

Hot update of existing service archives is disabled

- Cliquer ensuite sur **Choisissez un fichier** pour choisir le fichier AAR à déployer.
- Cliquer ensuite sur le bouton **Upload**.
- Pour vérifier si le Service Web "**Calculator**" a été bien déployé par **AXIS2**, cliquer sur le lien **Available Services** :

[System Components](#)
[Available Services](#)
[Available Service Groups](#)
[Available Modules](#)
[Globally Engaged Modules](#)
[Available Phases](#)
[Execution Chains](#)
[Global Chains](#)
[Operation Specific Chains](#)
[Engage Module](#)
[For all Services](#)
[For a Service Group](#)
[For a Service](#)
[For an Operation](#)
[Services](#)
[Deactivate Service](#)
[Activate Service](#)
[Edit Parameters](#)
[Contexts](#)
[View Hierarchy](#)

Available Services

Version

Service Description : Version
 Service EPR : http://localhost:8080/axis2/services/Version
 Service Status : Active Actions : [Remove Service](#)

Engaged modules for the service

- logging :: [Disengage](#)
- jaxws :: [Disengage](#)
- addressing :: [Disengage](#)

Available operations

- getVersion

Engaged Modules for the Operation

- logging :: [Disengage](#)
- jaxws :: [Disengage](#)
- addressing :: [Disengage](#)

[Calculator](#)

- Lancer le lien **http://localhost:8080/axis2/services/listServices** :

Available services

Version

Service Description : Version

Service EPR : <http://localhost:8080/axis2/services/Version>

Service Status : Active

Available Operations

- getVersion

Calculator

Service Description : Please Type your service description here

Service EPR : <http://localhost:8080/axis2/services/Calculator>

Service Status : Active

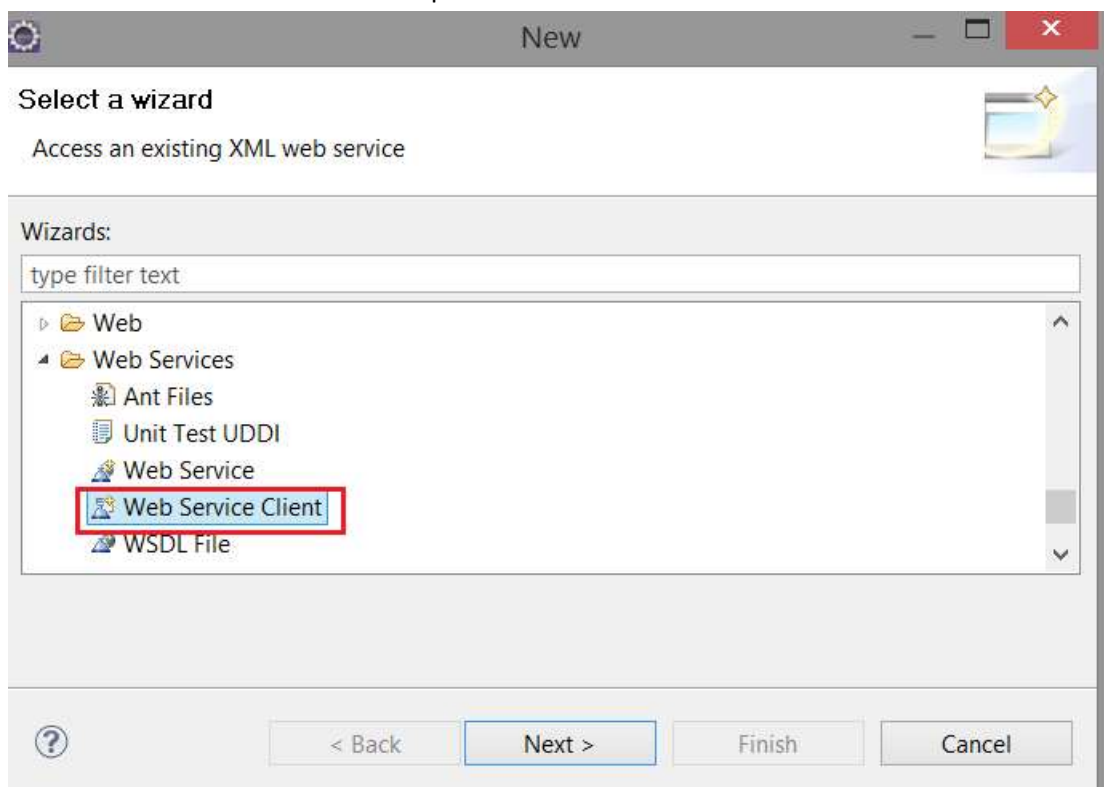
Available Operations

- add
- divide

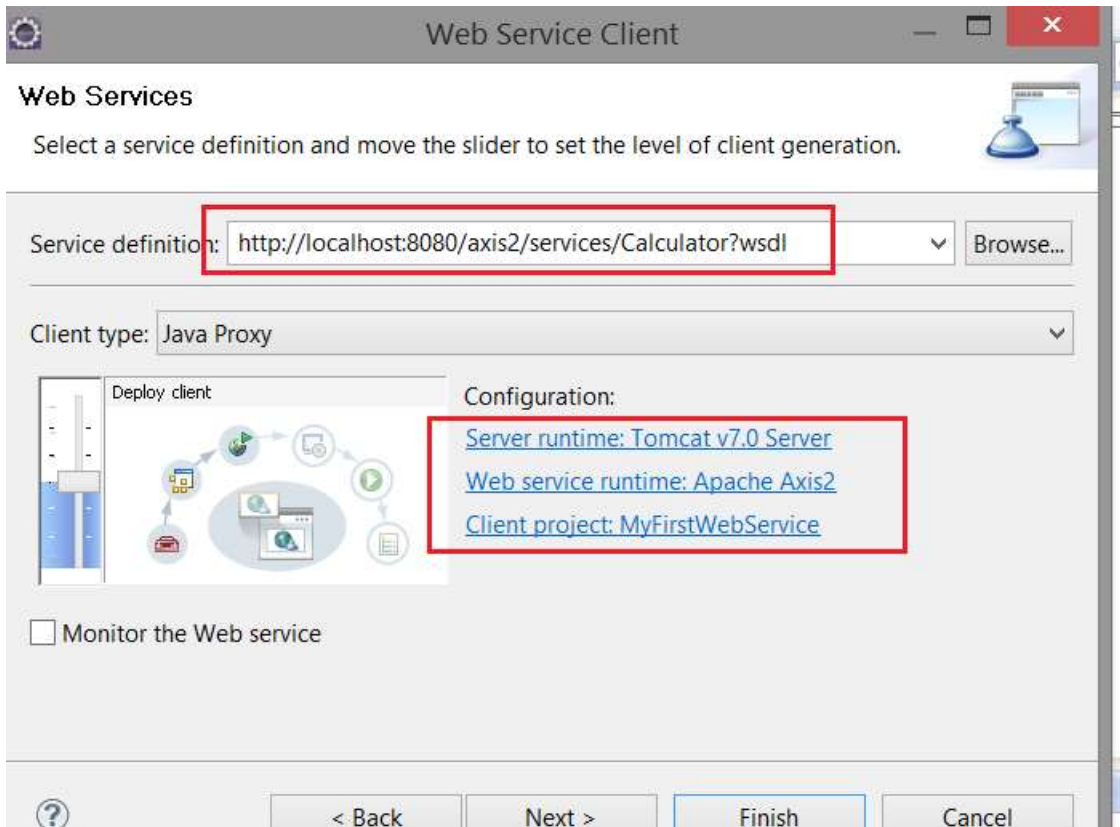
Noter l'URL du WSDL : ***<http://localhost:8080/axis2/services/Calculator?wsdl>***

C-Développement du Service Web client

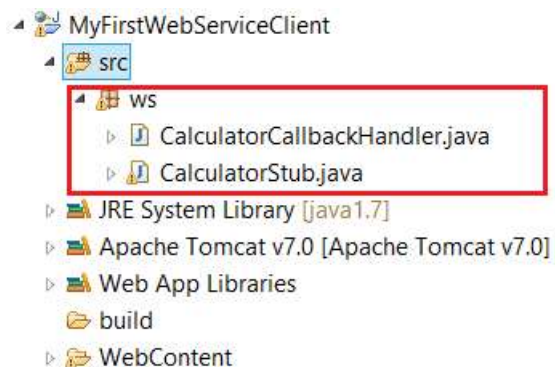
- Créer un projet Web Dynamic Project. Donner lui comme nom : ***MyFirstWebServiceClient***.
- Créer un Web Service Client en cliquant sur : ***Fichier ==> New ==> Other ...***



- Choisir Web Service Client et cliquer sur ***Next >***. La fenêtre suivante sera créée :



- Renseigner l'URL du WSDL ;
- Préciser le Server runtime ;
- Préciser le projet client et cliquer sur le bouton **Next >** puis sur **Finish**. Les deux classe Stubs seront créés :



- Créer la classe de test Test.java et tester les 04 opérations (add, subtract, divide, multiply) comme suit:

```
package test;

import java.rmi.RemoteException;

import org.apache.axis2.AxisFault;

import ws.CalculatorStub;
import ws.CalculatorStub.Add;
```

```

import ws.CalculatorStub.AddResponse;
import ws.CalculatorStub.Divide;
import ws.CalculatorStub.DivideResponse;
import ws.CalculatorStub.Multiply;
import ws.CalculatorStub.MultiplyResponse;
import ws.CalculatorStub.Subtract;
import ws.CalculatorStub.SubtractResponse;

public class Test {
    public static void main(String[] args) {
        double a = 12.45;
        double b = 5.7;
        try {
            CalculatorStub stub = new CalculatorStub();

            // Tester la méthode add
            Add add = new Add();
            add.setA(a);
            add.setB(b);
            AddResponse ar = stub.add(add);
            System.out.println(a + "+" + b + "= " + ar.get_return());

            // Tester la méthode subtract
            Subtract subtract = new Subtract();
            subtract.setA(a);
            subtract.setB(b);
            SubtractResponse sr = stub.subtract(subtract);
            System.out.println(a + "-" + b + "= " + sr.get_return());

            // Tester la méthode divide
            Divide divide = new Divide();
            divide.setA(a);
            divide.setB(b);
            DivideResponse dr = stub.divide(divide);
            System.out.println(a + "/" + b + "= " + dr.get_return());

            // Tester la méthode multiply
            Multiply multiply = new Multiply();
            multiply.setA(a);
            multiply.setB(b);
            MultiplyResponse mr = stub.multiply(multiply);
            System.out.println(a + "*" + b + "= " + mr.get_return());

        } catch (AxisFault e) {
            e.printStackTrace();
        } catch (RemoteException e) {
            e.printStackTrace();
        }
    }
}

```

Le résultat dans ce cas de test est :

Console 

12.45+5.7= 18.15

12.45-5.7= 6.749999999999999

12.45/5.7= 2.1842105263157894

12.45*5.7= 70.965