FUDAN-adweb

MyEclipse 创建基于 X-File 的 Web service 及调用实例详解

whh

2013-3-6

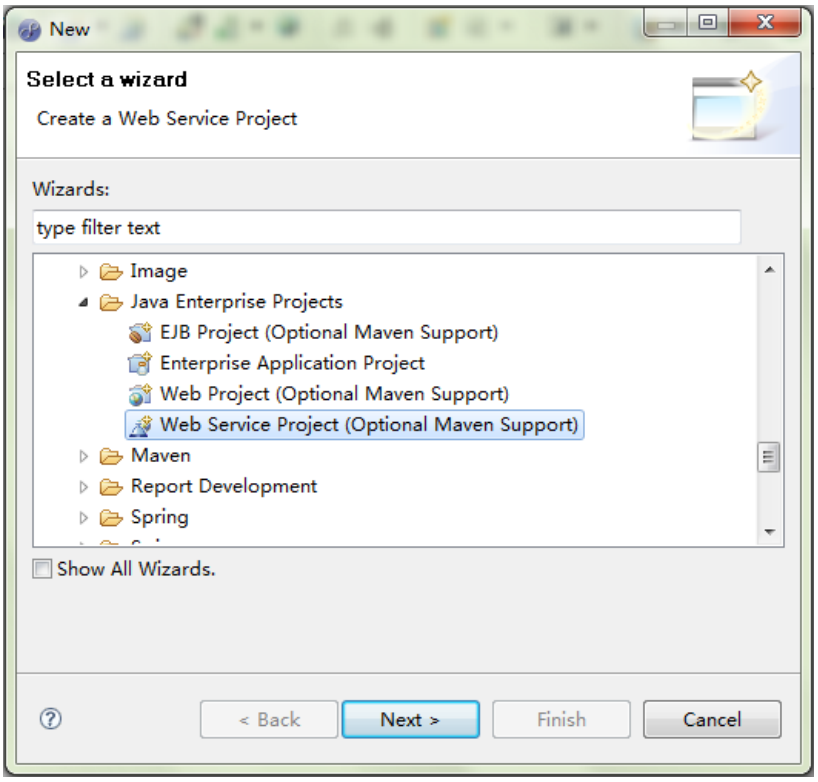
目录

1. 环境配置

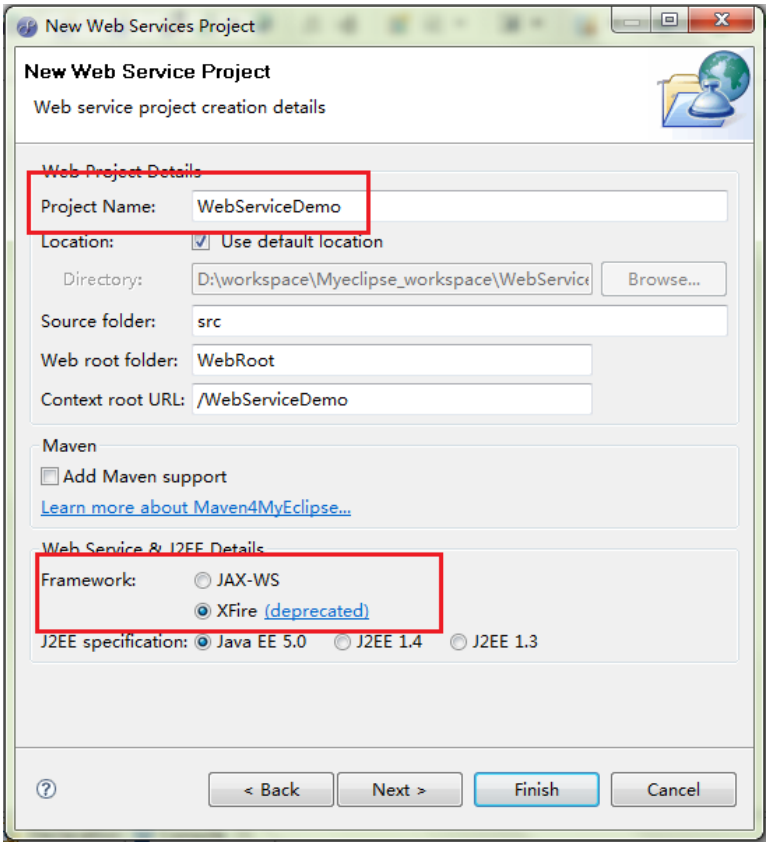
- MyEclipse 7.0
- JDK 1.6

2. 创建 Web service project

2.1 在 MyEclipse 中选择 File→New→Other，选择 Web Service Project。

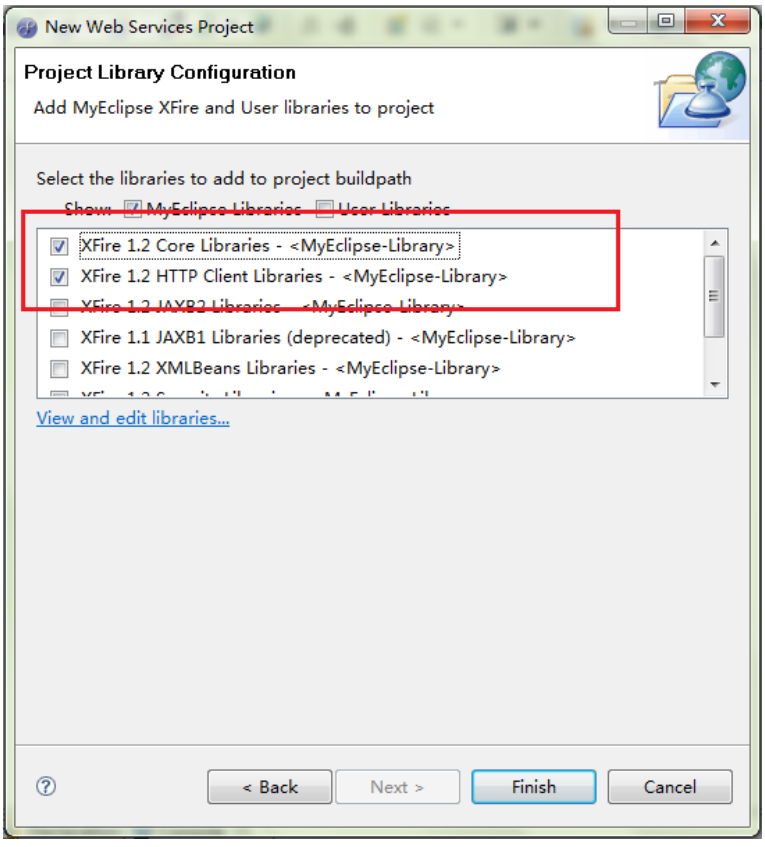


2.2 Project Name: WebServiceDemo; Framework: X-File; 其他选项默认。

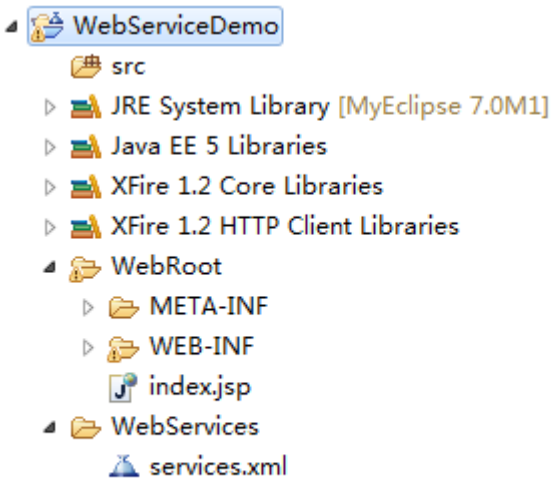


2.3 选择 next，保持默认值，选择 next。

2.4 Project Library Configuration，选择如下图，点击 finish



2.5 创建完成后看到的工程目录如下

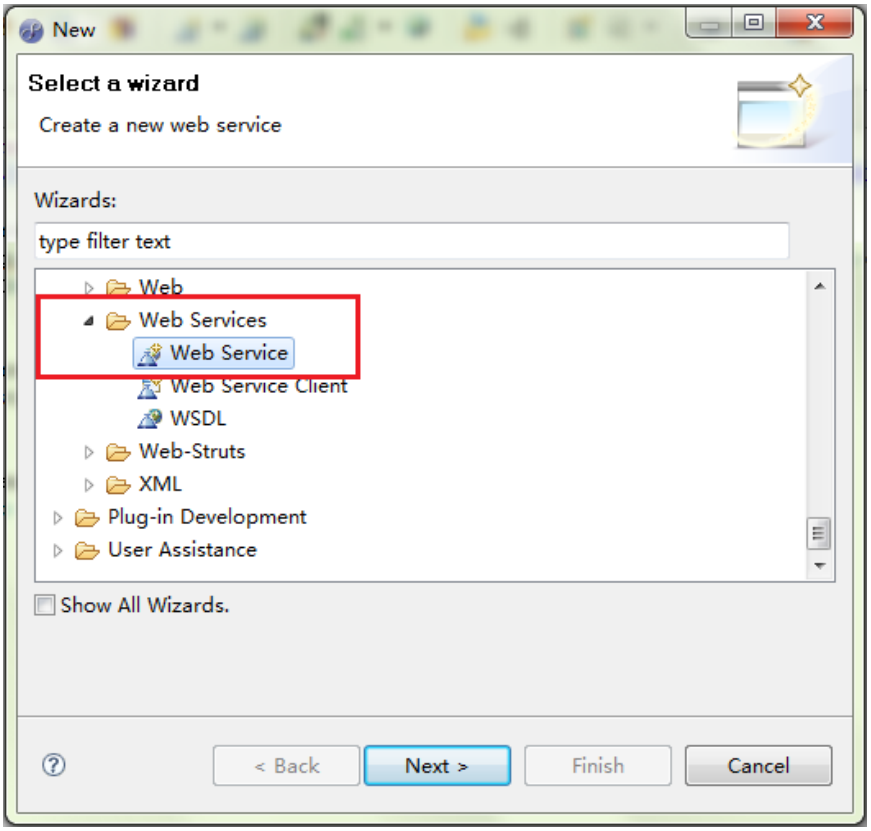


2.6 完成以上 5 步后，查看 WebRoot/WEB-INF/web.xml，这里指明了当遇到/services/*请求时，将选用 XFireConfigurableServlet 来处理

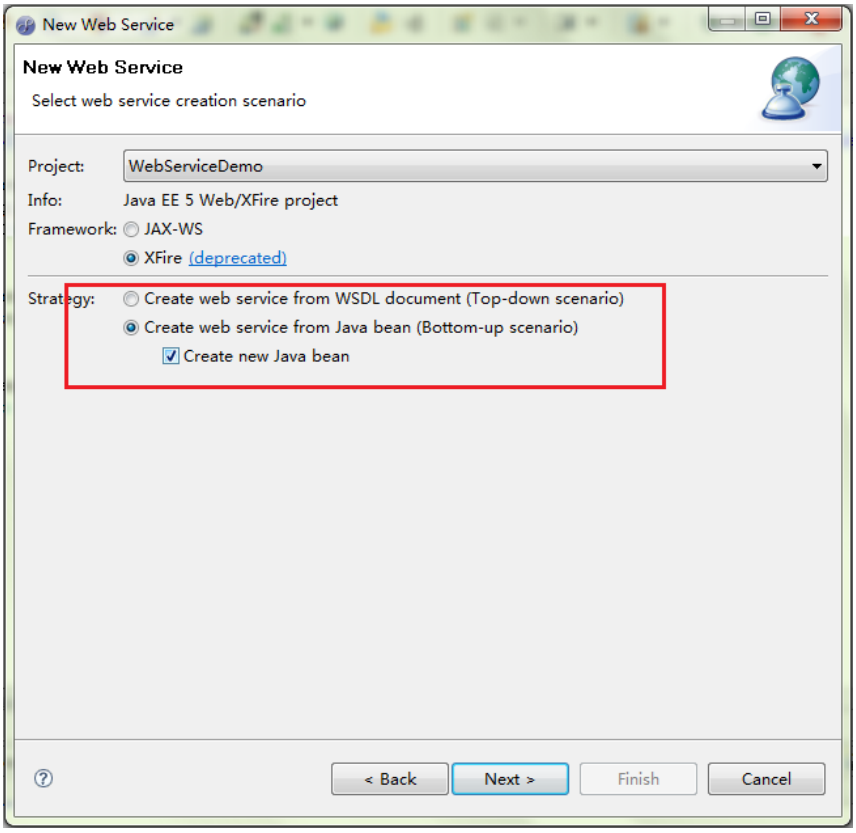
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 version="2.5"
 xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
 http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
  <servlet>
    <servlet-name>XFireServlet</servlet-name>
    <servlet-class>org.codehaus.xfire.transport.http.XFireConfigurableServlet</servlet-class>
    <load-on-startup>0</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>XFireServlet</servlet-name>
    <url-pattern>/services/*</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

3. 编写 Web service 服务端代码

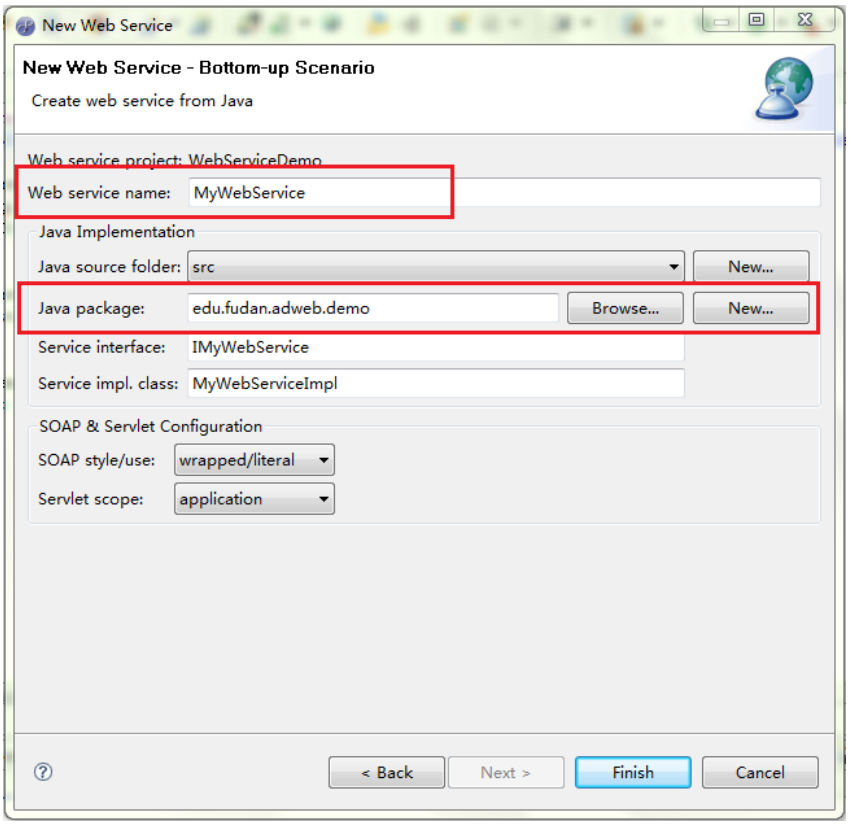
3.1 在刚刚创建的 WebServiceDemo 上右键，选择 New→Other→Web services→Web service，创建代码。



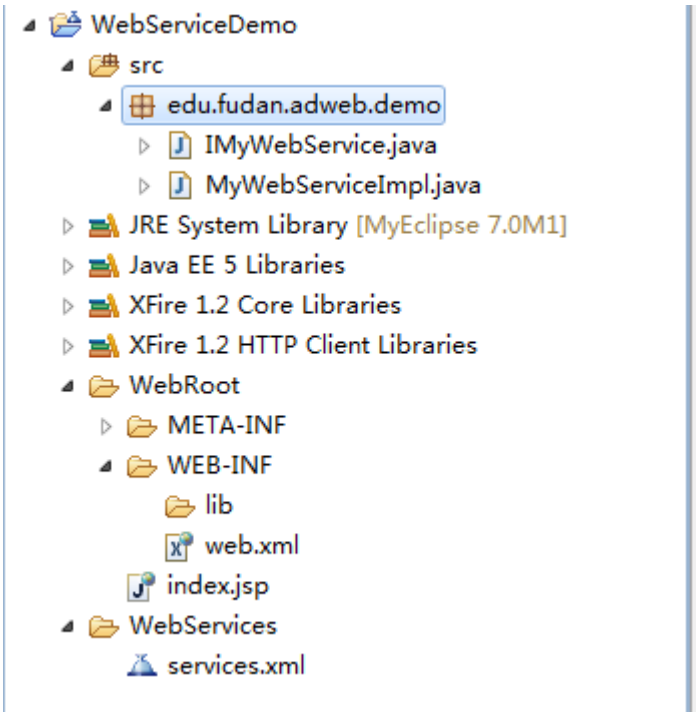
3.2 选择通过 JAVA CLASS 创建 Web Service。



3.3 创建 MyWebService，通过 New 创建 edu.fudan.adweb.demo package，向导将为工程自动创建对应的接口类与实现类。



工程目录如下：



3.4 创建成功后查看 WebServices/services.xml，发现配置文件多了以下内容：

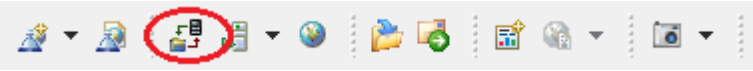
```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://xfire.codehaus.org/config/1.0">

    <service>
        <name>MyWebService</name>
        <serviceClass>edu.fudan.adweb.demo.IMyWebService</serviceClass>
        <implementationClass>
            edu.fudan.adweb.demo.MyWebServiceImpl
        </implementationClass>
        <style>wrapped</style>
        <use>literal</use>
        <scope>application</scope>
    </service></beans>
```

3.5 修改 MyWebServiceImpl.java 文件，实现方法内容。

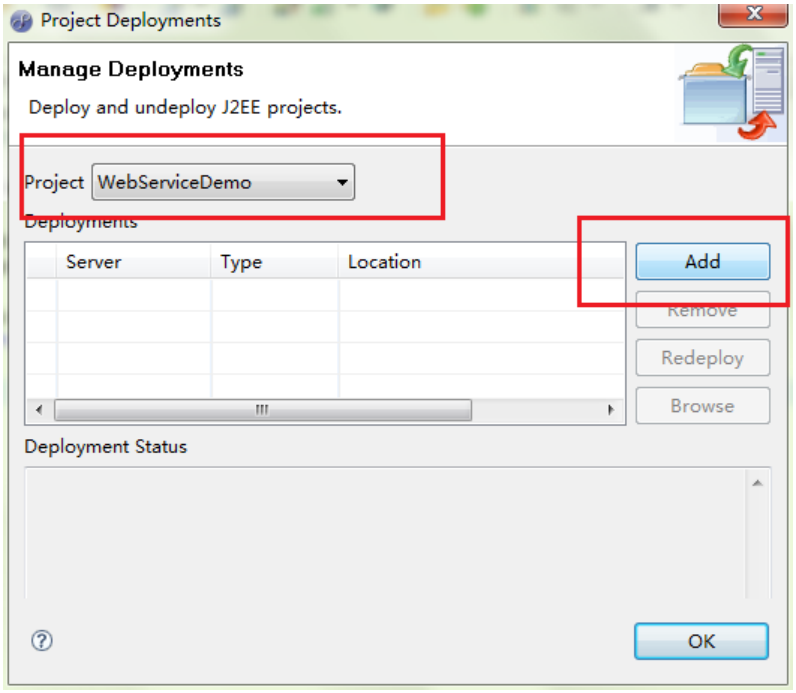
```java
package edu.fudan.adweb.demo;
//Generated by MyEclipse

public class MyWebServiceImpl implements IMyWebService {

    public String example(String message) {
        return getAuthor() + ":" + message;
    }
    private String getAuthor(){
        return "whh";
    }
}
```

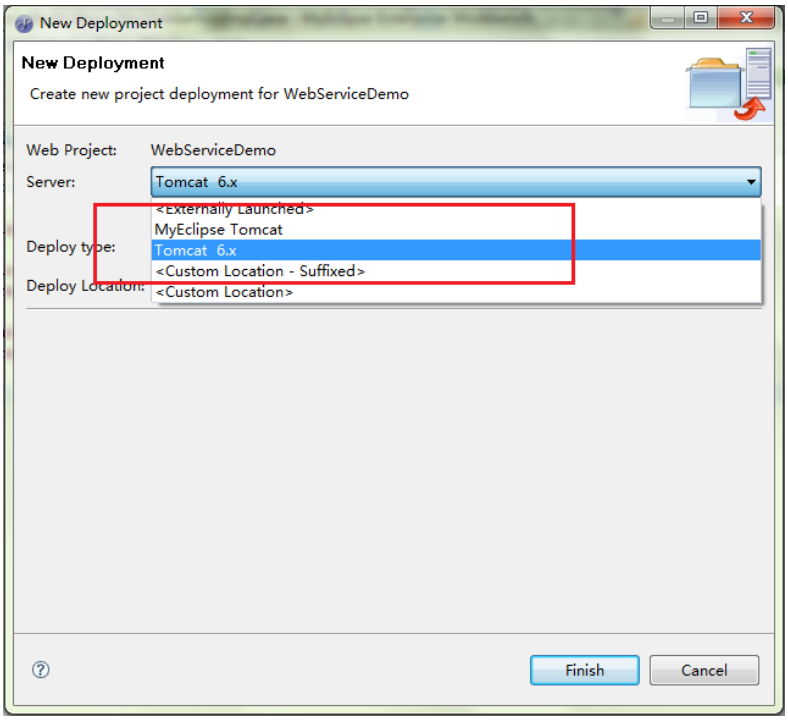
4. 部署并测试 WebSericeServer

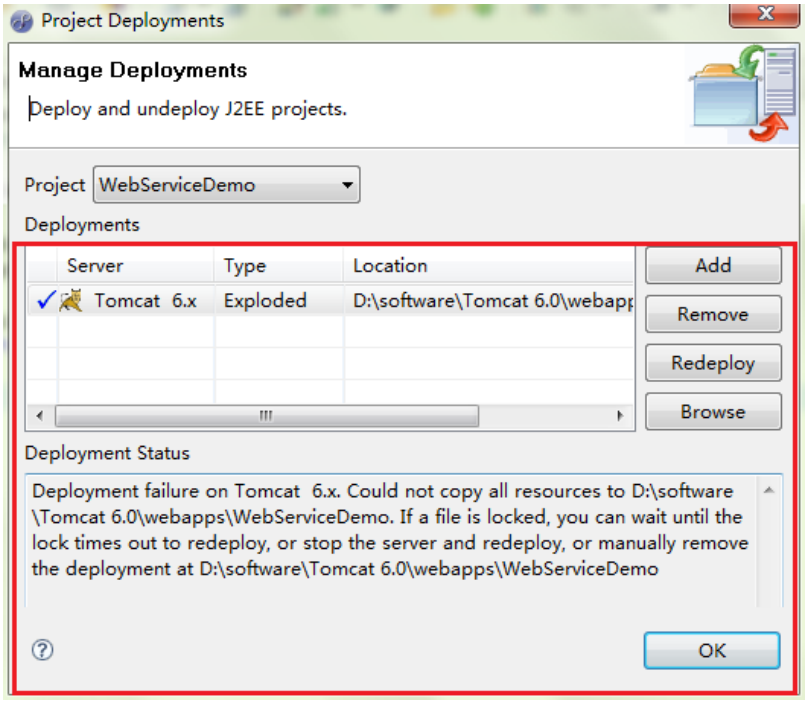
4.1. 部署 WebServiceServer



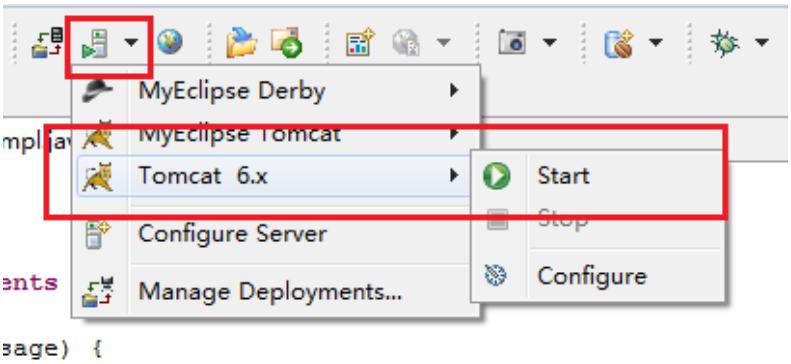
在部署向导中选择项目、服务器


Project Deployments
Manage Deployments
Deploy and undeploy J2EE projects.
Project
WebServiceDemo
Deployments
Server
Type
Location
Add
Remove
Redeploy
Browse
Deployment Status
OK


New Deployment
New Deployment
Create new project deployment for WebServiceDemo
Web Project:
WebServiceDemo
Server:
Tomcat 6.x
<Externally Launched>
MyEclipse Tomcat
Tomcat 6.x
<Custom Location - Suffixed>
<Custom Location>
Deploy type:
Deploy Location:
Finish
Cancel



4.2. 启动服务器

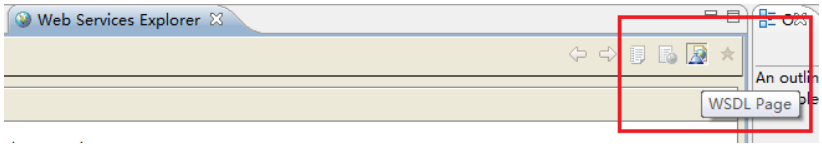


4.3. 测试 WebService

4.3.1 通过 MyEclipse 自带 Explorer 测试

MyEclipse 提供了一个 Web Service Explorer 来测试 Web Service

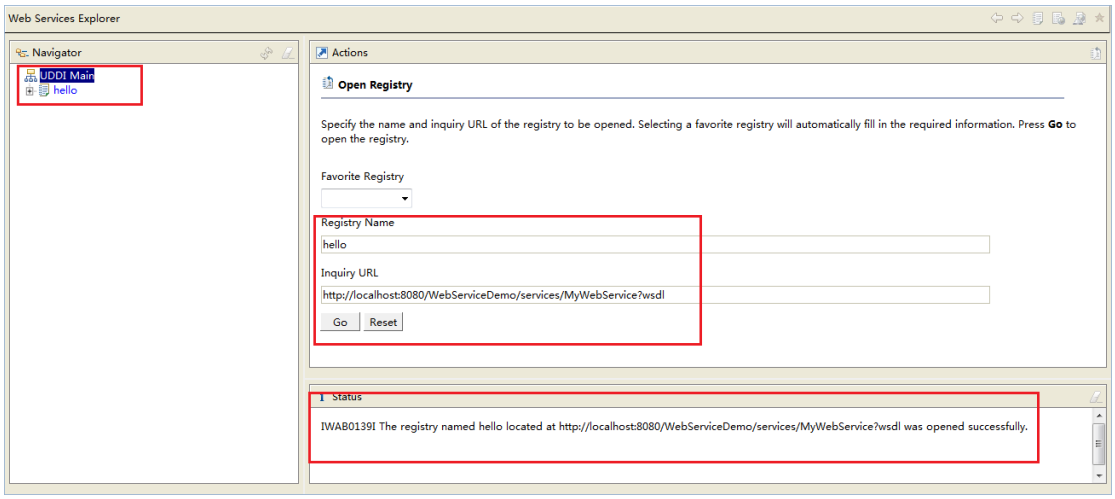
选择列表中的 Launch SOAP Web Services Explorer，选择 WSDL 模式



在 Explorer 中选择 WSDL Main，在右边的 Registry Name 中输入 hello(这是因为我们定义的 service 中 example 方法需要参数)，在 URL 中输入 http://localhost:8080/WebServiceDemo/services/MyWebService?wsdl，单击"Go"。

其中，8080 为之前选择的服务器的端口号，WebServiceDemo 为我们定义的 web service project 名字(见 2.2)，MyWebService 为我们创建的 Service 名字(见 3.3)。

若在 Status 中显示成功打开 WSDL 文件则测试成功



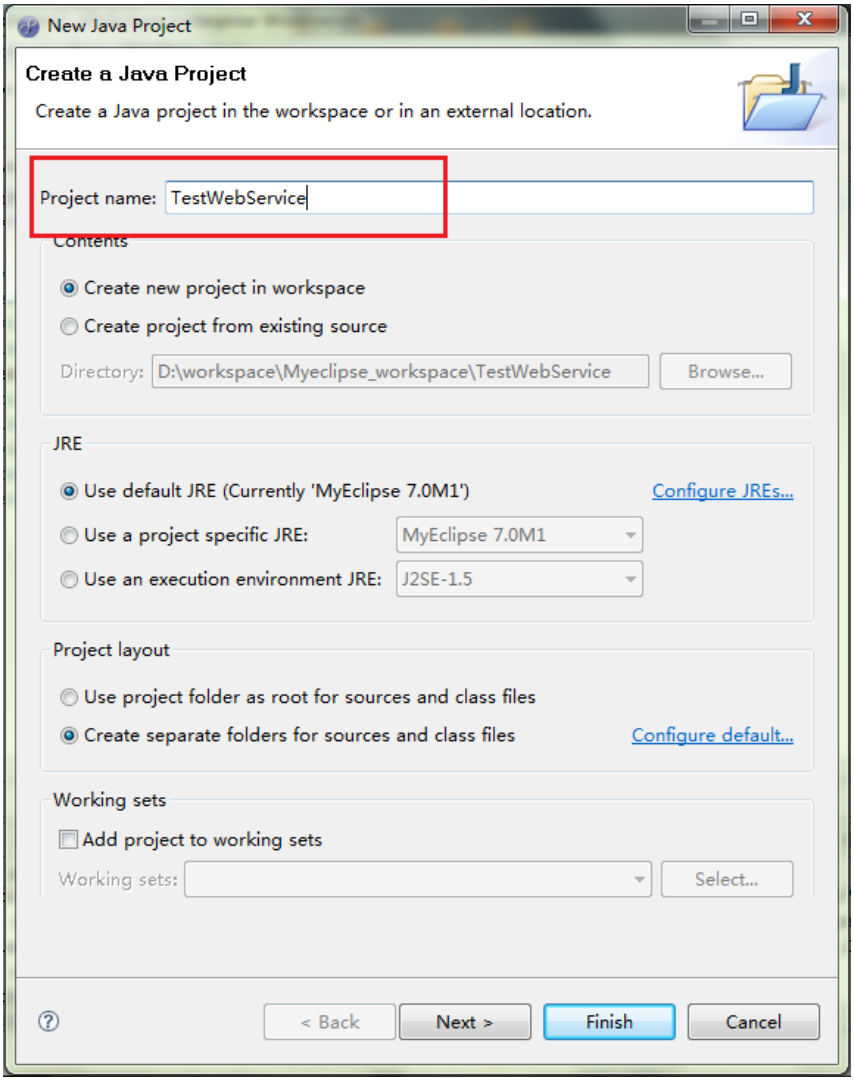
4.3.2 直接通过外部浏览器测试

在外部浏览器地址栏中输入：http://localhost:8080/WebServiceDemo/services/MyWebService?wsdl，打开如下则说明部署成功。

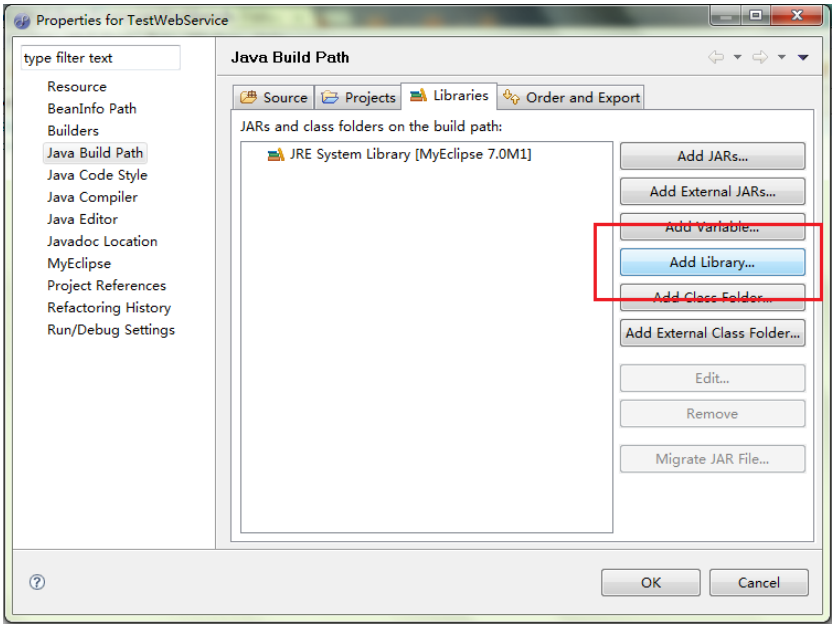
5. 创建 Web service 客户端

经过以上几步，我们已经成功发布了一个 WebService，现在需要创建一个客户端来调用该服务。调用 web service 的客户端可以是一个简单的 Java Project，也可以是另外一个 web service，在此步骤中我们创建一个 Java Project 来调用在以上步骤创建的 web service。

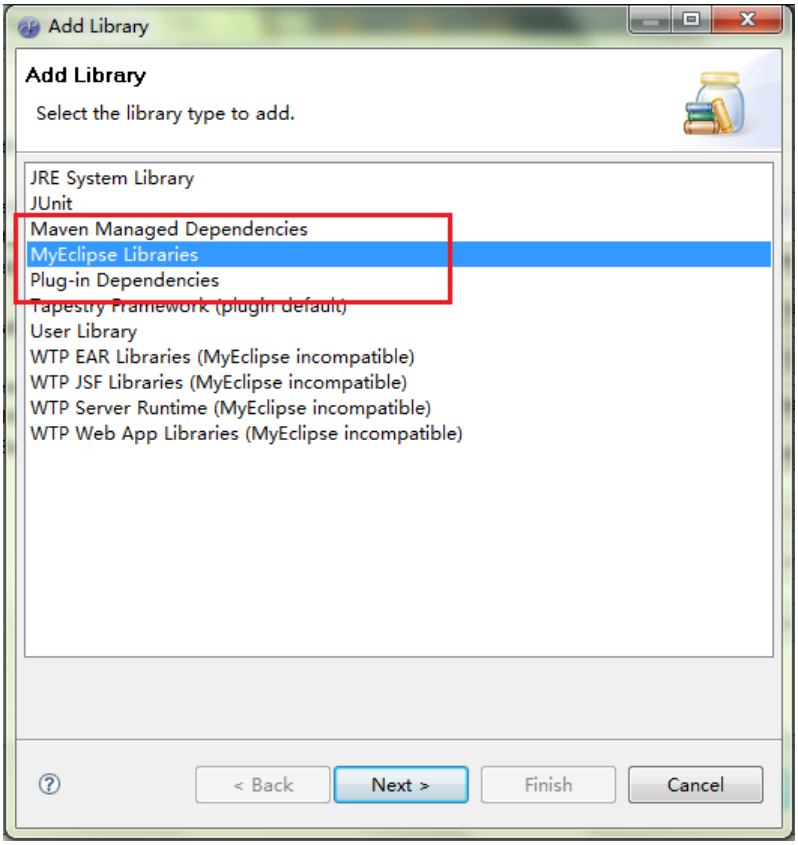
5.1 创建 Java Project：File→New→Java Project. Project Name: TestWebService



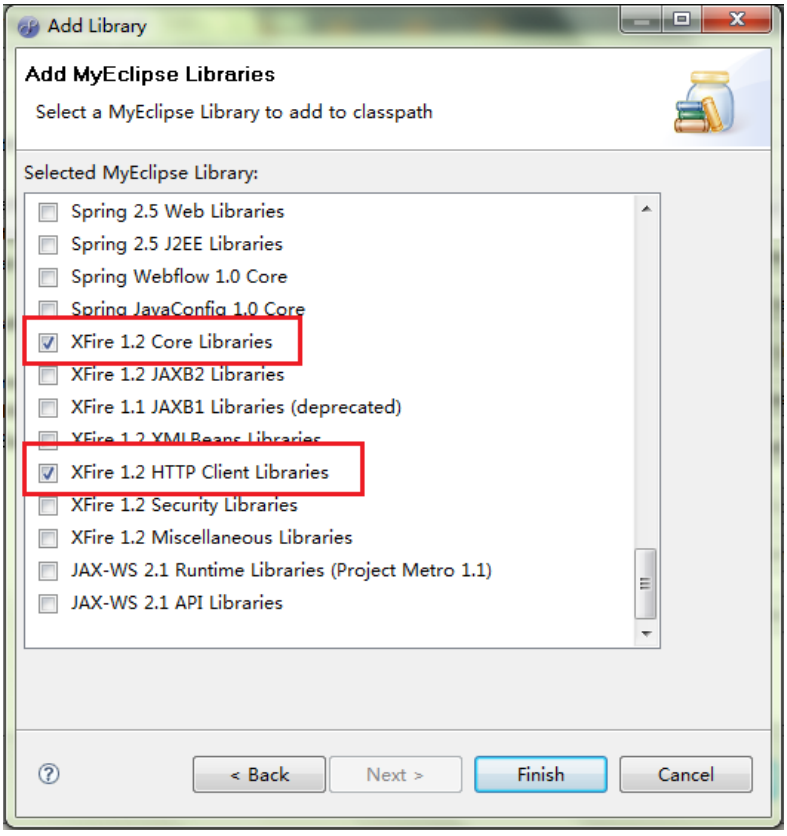
5.2 引入 MyEclipse Libraries，右键 TestWebServic→Build Path→Add Library



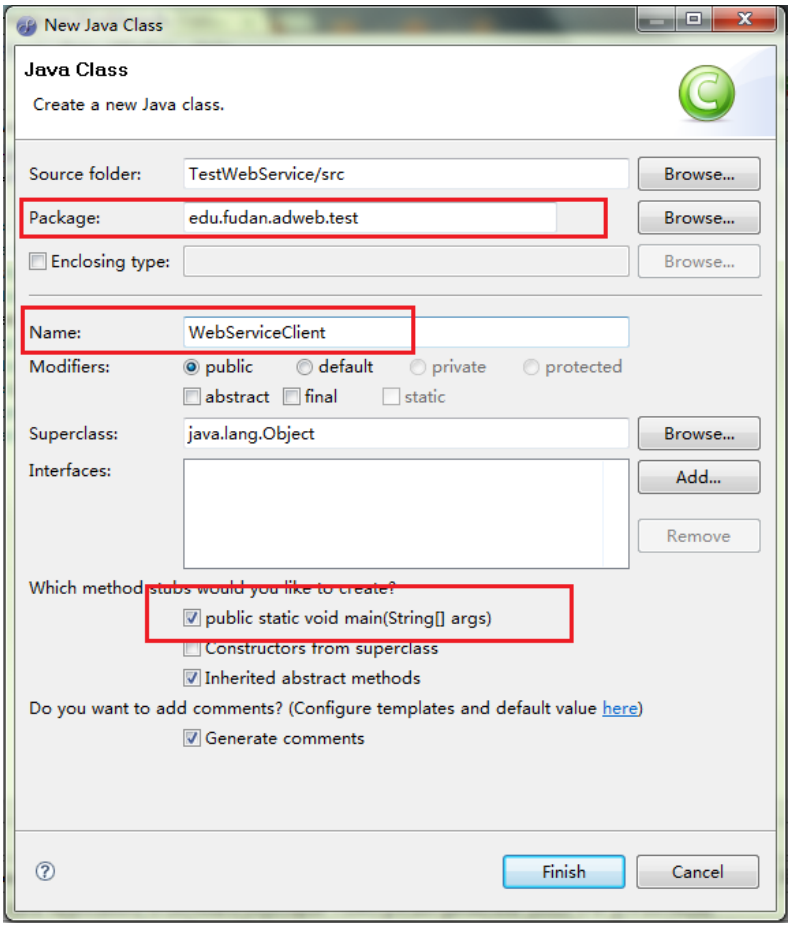
选择 MyEclipse Libraries



选择 XFile 1.2 Core Libraries 和 XFire 1.2 HTTP Client Libraries



5.3 创建 WebServiceClient.java

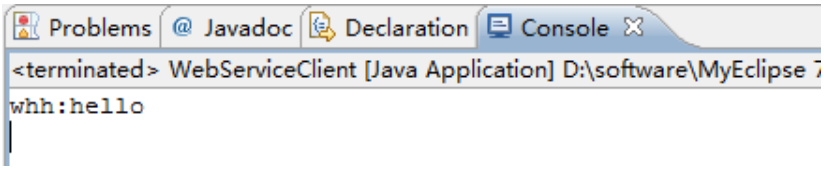



```
/**
package edu.fudan.adweb.test;
import java.net.MalformedURLException;
import java.net.URL;
import org.codehaus.xfire.client.Client;

/**
 * @author whh
 *
 */
public class WebServiceClient {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub

        try {
            Client client = new Client(new URL("http://localhost:8080/WebServiceDemo/services/MyWebService?wsdl"));
            Object[] results = client.invoke("example", new Object[] {"hello"});
            System.out.println((String) results[0]);
        } catch (MalformedURLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }

}
```

（其中方框中”example”为想要调用的 web service 中的方法名，”hello”为传入的参数。见 3.5）

5.4 在确认服务器启动的情况下，运行 HelloWebService.java，右键点击 Run As→Java

Application，在控制台中看到如下输出



至此，通过 Java Project 调用 web service 测试完成。

6. 附：Web service 传递自定义数据的实现

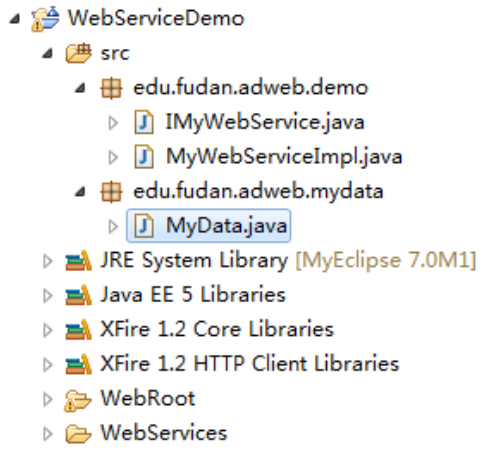
在以上过程中，web service 返回的为简单的 String，可以通过 5.3 中的 Client 直接调用并得到返回值，但是某些时候（在实际项目中）我们可能需要 web service 返回较复杂的值，这个时候需要通过生成客户端的方式来调用 web service。

在下面的实例中，我们让 web service 返回一个我们自定义的 MyData 数据的 ArrayList。

6.1 首先，我们在创建 MyWebService（见 3.3）后，另外创建一个 edu.fudan.adweb.mydata package 和 MyData 类，代码如下：

```
/**
 package edu.fudan.adweb.mydata;
/**
 * @author whh
 *
 */
public class MyData {
    private String name;
    private int ID;
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public void setID(int iD) {
        ID = iD;
    }
    public int getID() {
        return ID;
    }
}
```

此时工程目录为:



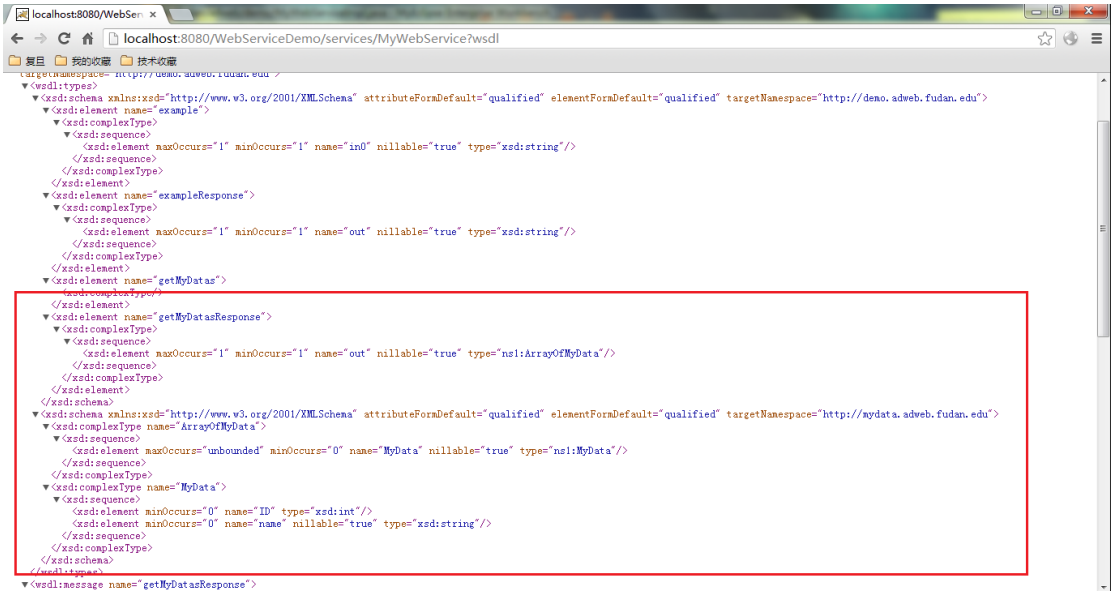
6.2 之后修改 IMyWebService.java 和 MyWebServiceImpl.java，增加 getMyDatas()方法。

```java
package edu.fudan.adweb.demo;

import java.util.ArrayList;

import edu.fudan.adweb.mydata.MyData;

//Generated by MyEclipse

public interface IMyWebService {

    public String example(String message);

    public ArrayList<MyData> getMyDatas();
}
```

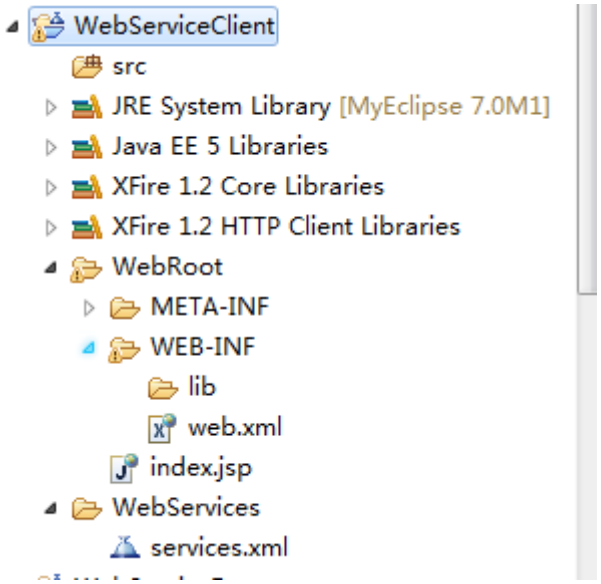
MyWebServiceImpl.java:

```java
package edu.fudan.adweb.demo;

import java.util.ArrayList;

import edu.fudan.adweb.mydata.MyData;
//Generated by MyEclipse

public class MyWebServiceImpl implements IMyWebService {

    public String example(String message) {
        return getAuthor() + ":" + message;
    }
    private String getAuthor(){
        return "whh";
    }
    public ArrayList<MyData> getMyDatas() {
        // TODO Auto-generated method stub\
        ArrayList<MyData> ret = new ArrayList<MyData>();

        MyData d1 = new MyData();
        d1.setID(1);
        d1.setName("data1");

        MyData d2 = new MyData();
        d2.setID(2);
        d2.setName("data2");

        ret.add(d1);
        ret.add(d2);

        return ret;
    }
}
```

6.3 重新部署后可以在 wsdl 文件中看到如下内容（方框中），可以看到我们定义的返回值。

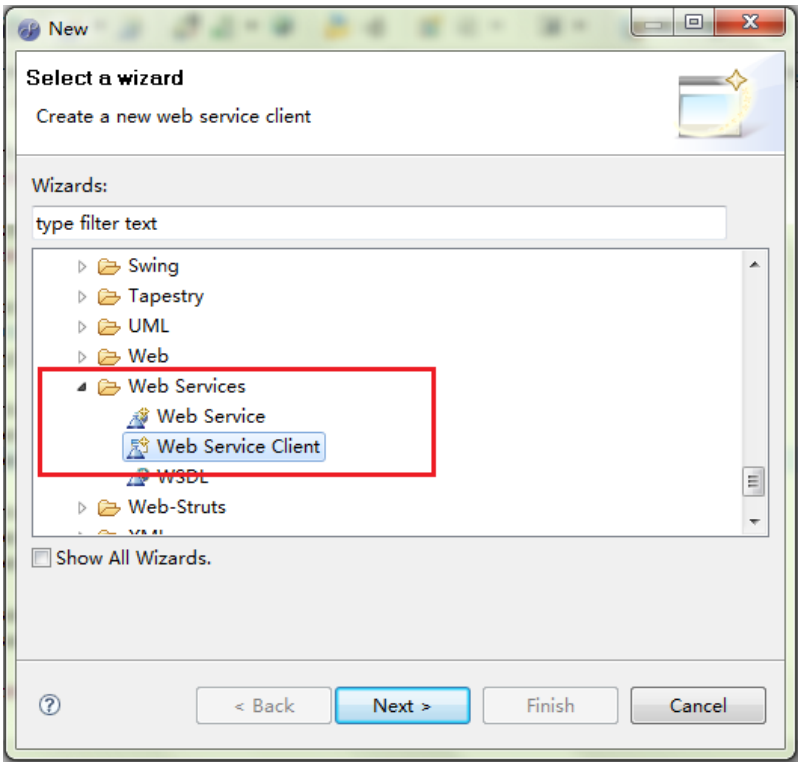


6.4 采用生成客户端的方式来调用 web service 时，不能够在 Java Project 中生成，我们需要另外建立一个 web service project（参考步骤 2.1 至 2.5），新建的 web service project 名字为 WebServiceClient.

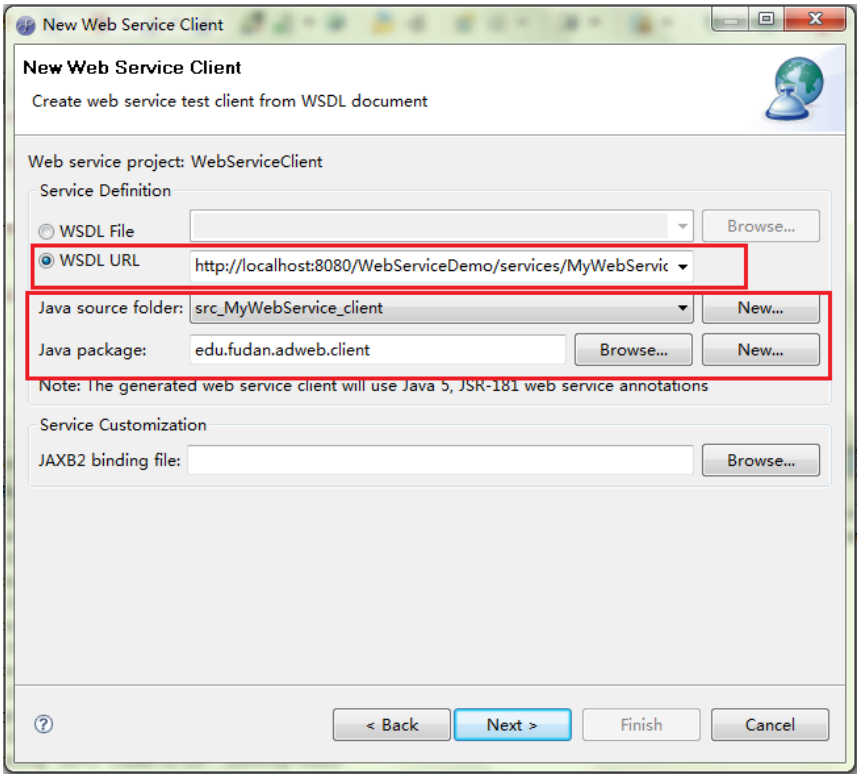


6.5 接下来我们需要通过 wsdl 文件生成一个客户端。（6.5 整个步骤中需要保证 Tomcat 服务器运行中）

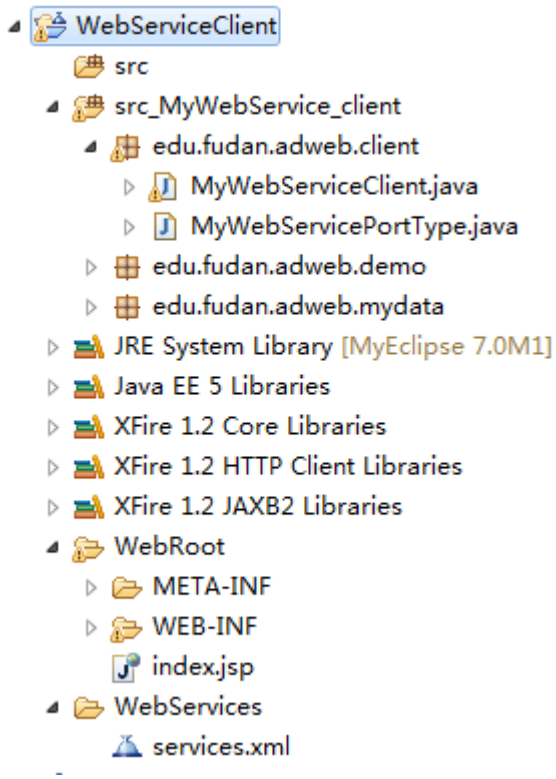
在刚刚创建的 WebServiceClient 上右键，选择 New→Other→Web services→Web service client，创建代码。



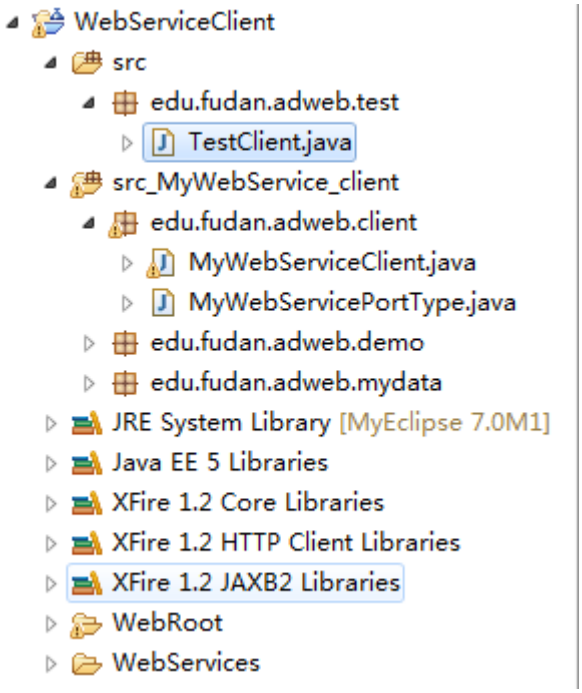
根据向导配置如下：WSDL URL 为之前发布的 web service 地址（和 4.3 种 URL 地址相同），java source folder 和 java package 都是点击 New 根据向导提示生成。



一路 next 后 finish，最后配置好后工程目录如下：（src_MyWebService_client 下文件均是自动生成的）



6.6 接下来，我们在 WebServiceClient src 文件夹下创建 edu.fudan.adweb.test package 和 TestClient.java 文件，内容如下：

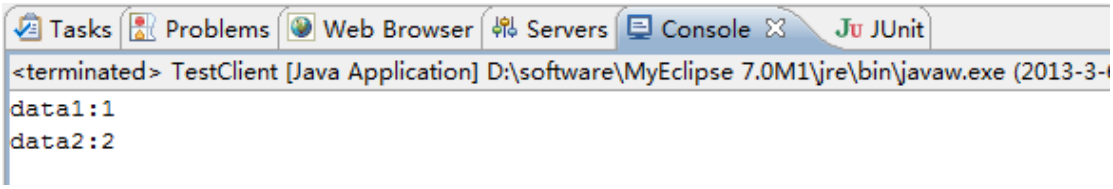


TestClient.java：

```java
/**
 */
package edu.fudan.adweb.test;

import java.util.ArrayList;
import java.util.List;

import edu.fudan.adweb.client.MyWebServiceClient;
import edu.fudan.adweb.client.MyWebServicePortType;
import edu.fudan.adweb.mydata.MyData;
/**
 * @author whh
 *
 */
public class TestClient {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        MyWebServiceClient client = new MyWebServiceClient();
        MyWebServicePortType service = client.getMyWebServiceHttpPort();

        List<MyData> datas = new ArrayList<MyData>();
        datas = service.getMyDatas().getMyData();

        for(MyData d : datas){
            System.out.println(d.getName().getValue() + ":" + d.getID());
        }
    }

}
```

其中，MyWebServiceClient 和 MyWebServicePortType 均是根据 wsdl 生成客户端时自动生成的文件。

运行 TestClient，控制台输出如下：

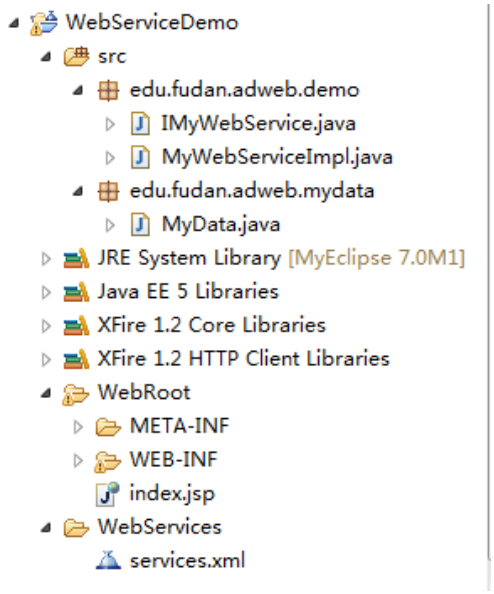


至此，根据 wsdl 文件生成客户端，传递自定义数据的测试完成。

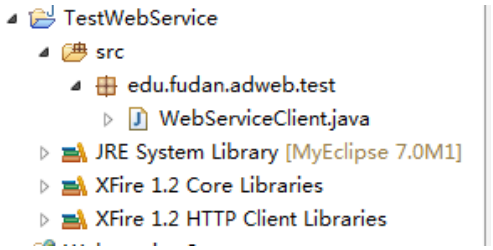
7. 附录：涉及到的工程的工程目录

整个过程中一共建立的三个工程，分别如下：

- Web Service Project：WebServiceDemo 实现了服务端代码，定义了自定义数据类型



- Java Project：TestWebService 实现了调用服务端的客户端代码



- Web Service Project：WebServiceClient 实现了通过 wsdl 文件自动生成客户端调用服务端的代码，接受了服务端传递过来的自定义数据。