

特别说明

此资料来自豆丁网(<http://www.docin.com/>)

您现在所看到的文档是使用**下载器**所生成的文档

此文档的原件位于

<http://www.docin.com/p-51696638.html>

感谢您的支持

抱米花

<http://blog.sina.com.cn/lotusbaob>

webService 简单示例

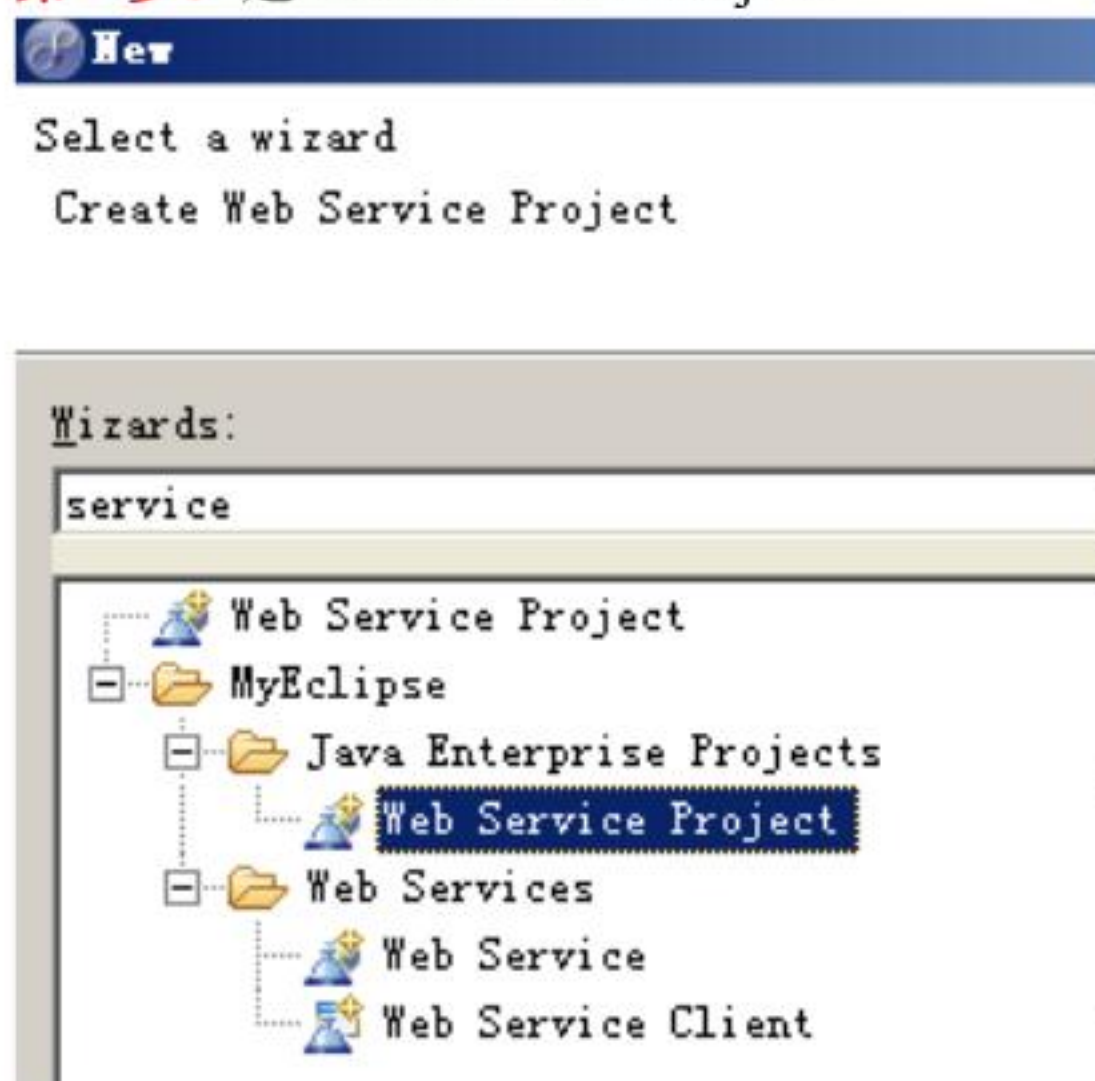
webService 客户端开发【一】

目的：使用 webServices 技术从提供服务端获取信息

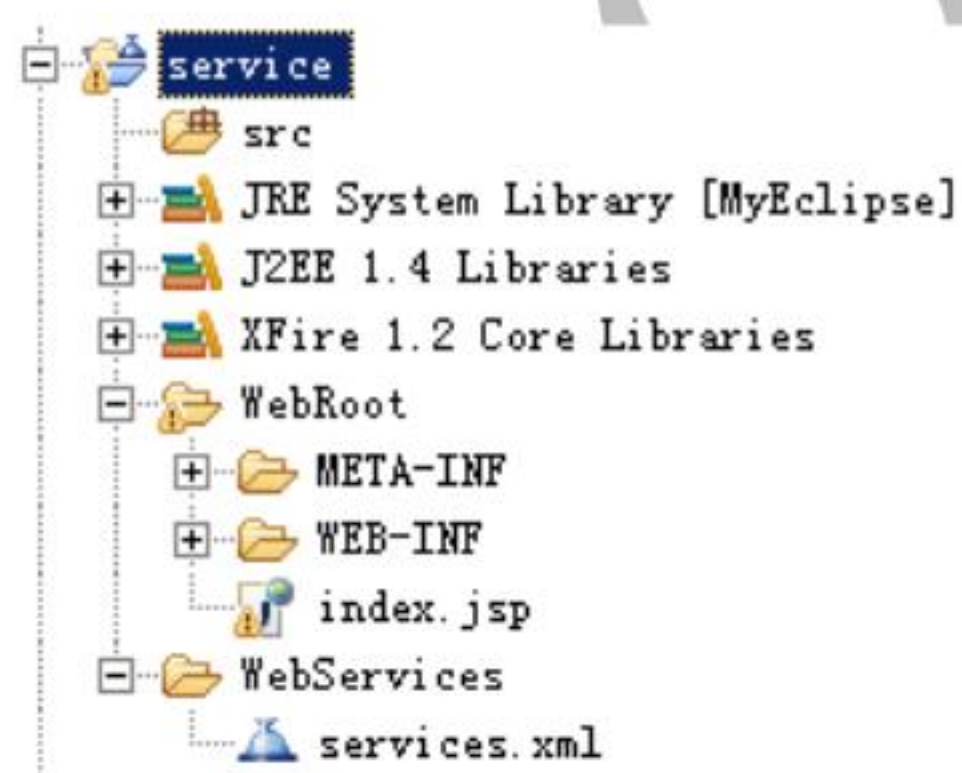
开发思路：创建 web service 服务器端；创建客户端 (C/S ; B/S 均可)进行对服务器端的调用

服务器端：

第一步：建立 Web Service Project



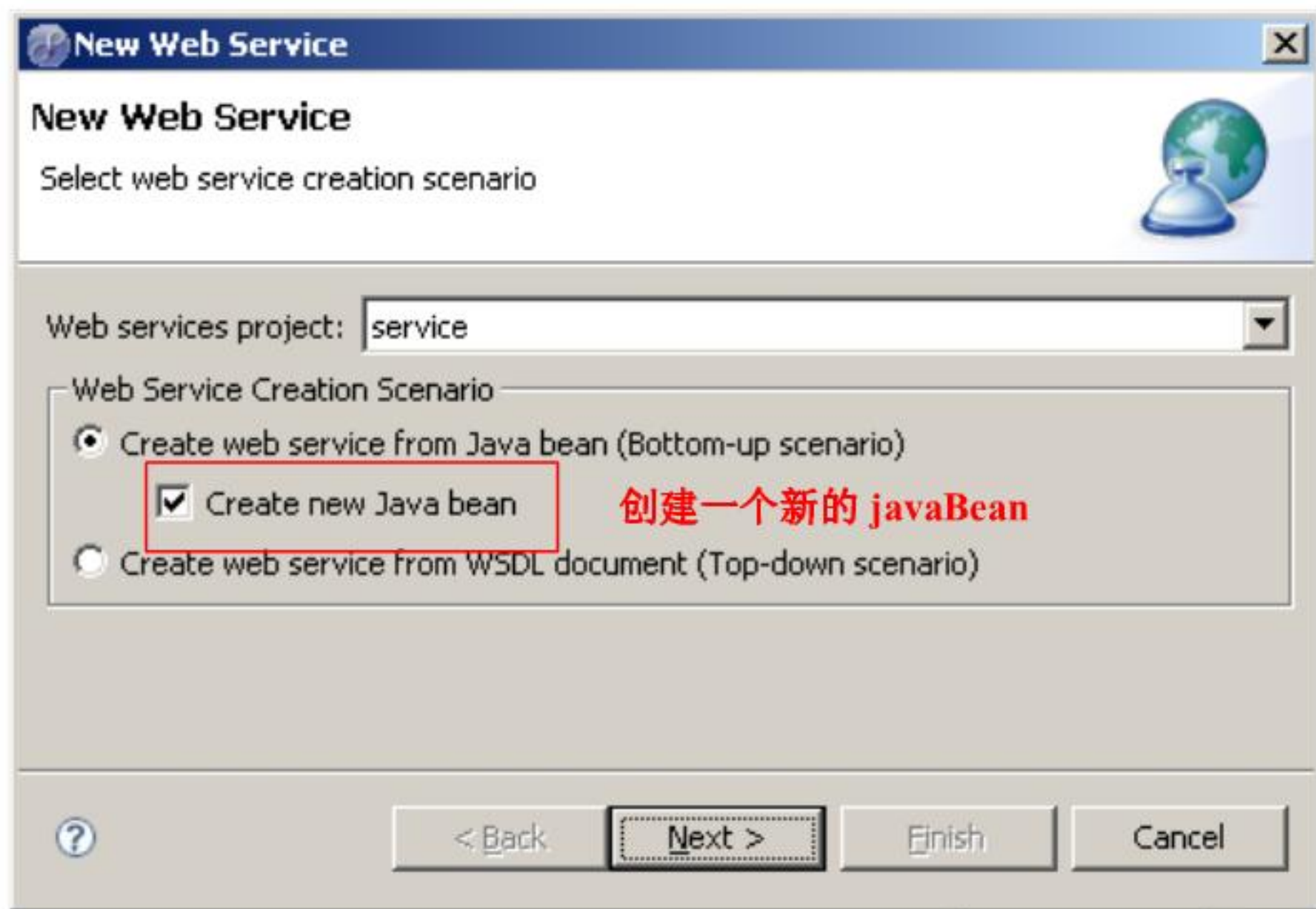
建立工程之后：



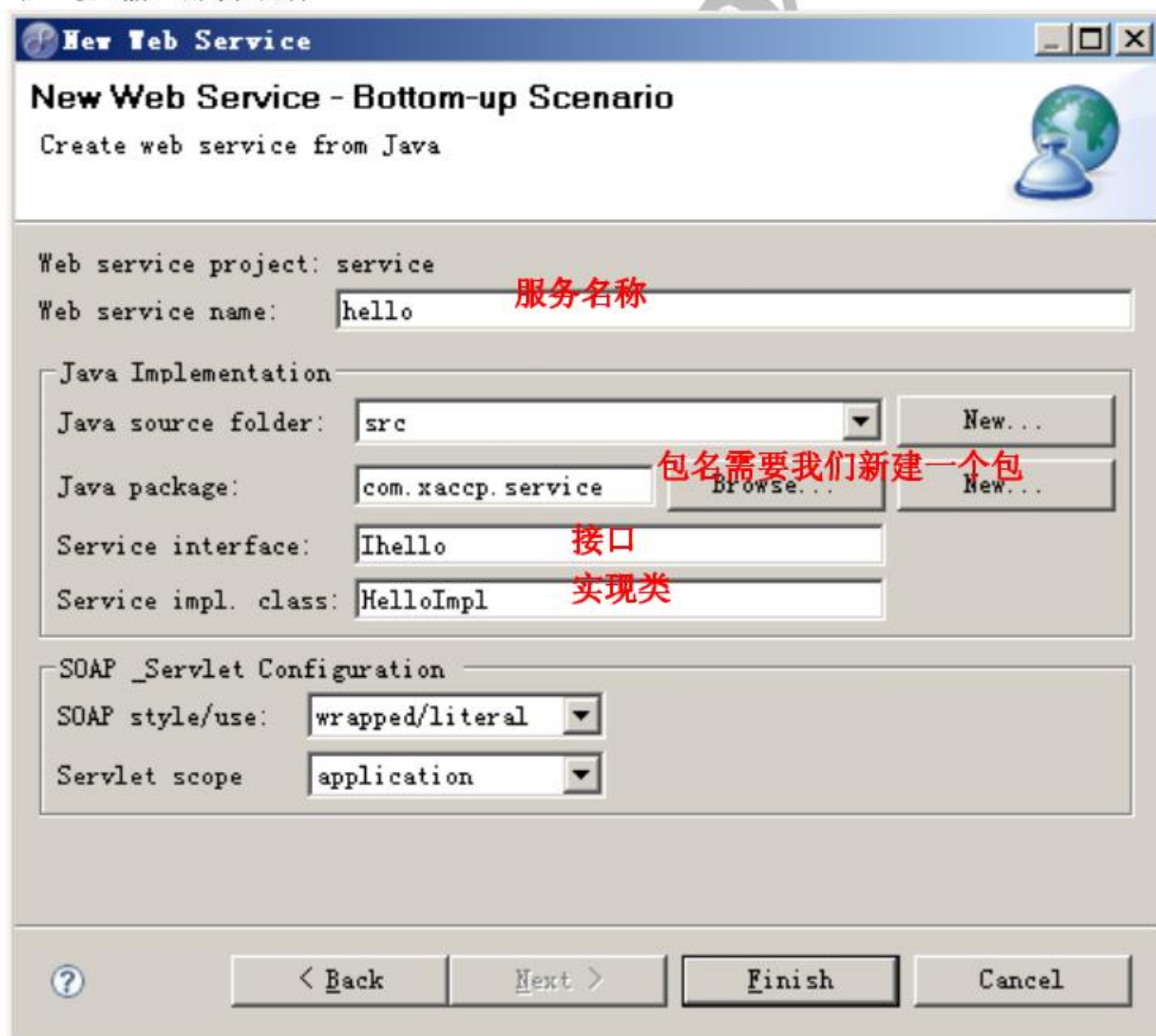
比一般的 web 工程多了一个 webServices 目录, 该目录中有一个 services.xml 文件, 是服务配置文件。

第二步：新建一个 Web Service





下一步:输入服务名称



更改生成的代码(更改为我们想要的方法, 更改接口和实现类):

```
public interface IHello {  
  
    public String sayHello(String name);  
  
}
```



```
public class HelloImpl implements IHello {  
  
    public String sayHello(String name) {  
        return "Hello "+name;  
    }  
  
}
```

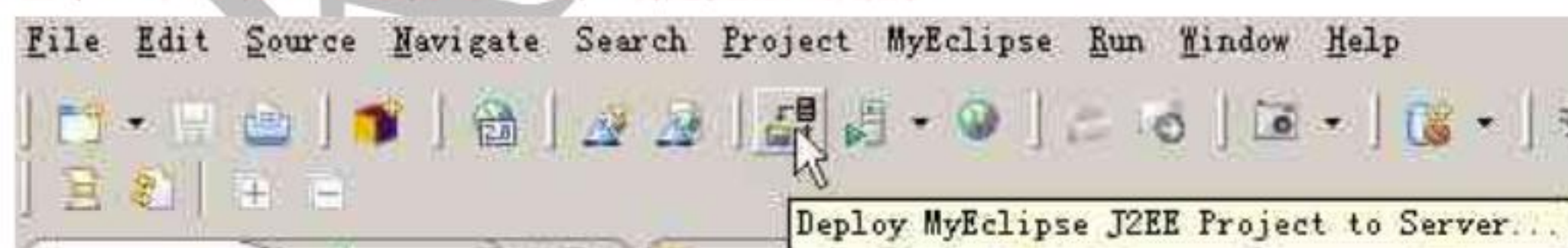
察看 web.xml 文件:

```
<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>
```

察看 service.xml

```
<service>  
    <name>Hello</name> ← 服务名称, 它会被客户端应用程序所使用  
    <serviceClass>com.xaccp.service.IHello</serviceClass> ← <name>元素所绑定的  
    <implementationClass>                                服务接口  
        com.xaccp.service.HelloImpl ← 服务接口的实现类  
    </implementationClass>  
    <style>wrapped</style>  
    <use>literal</use>  
    <scope>application</scope>  
</service></beans>
```

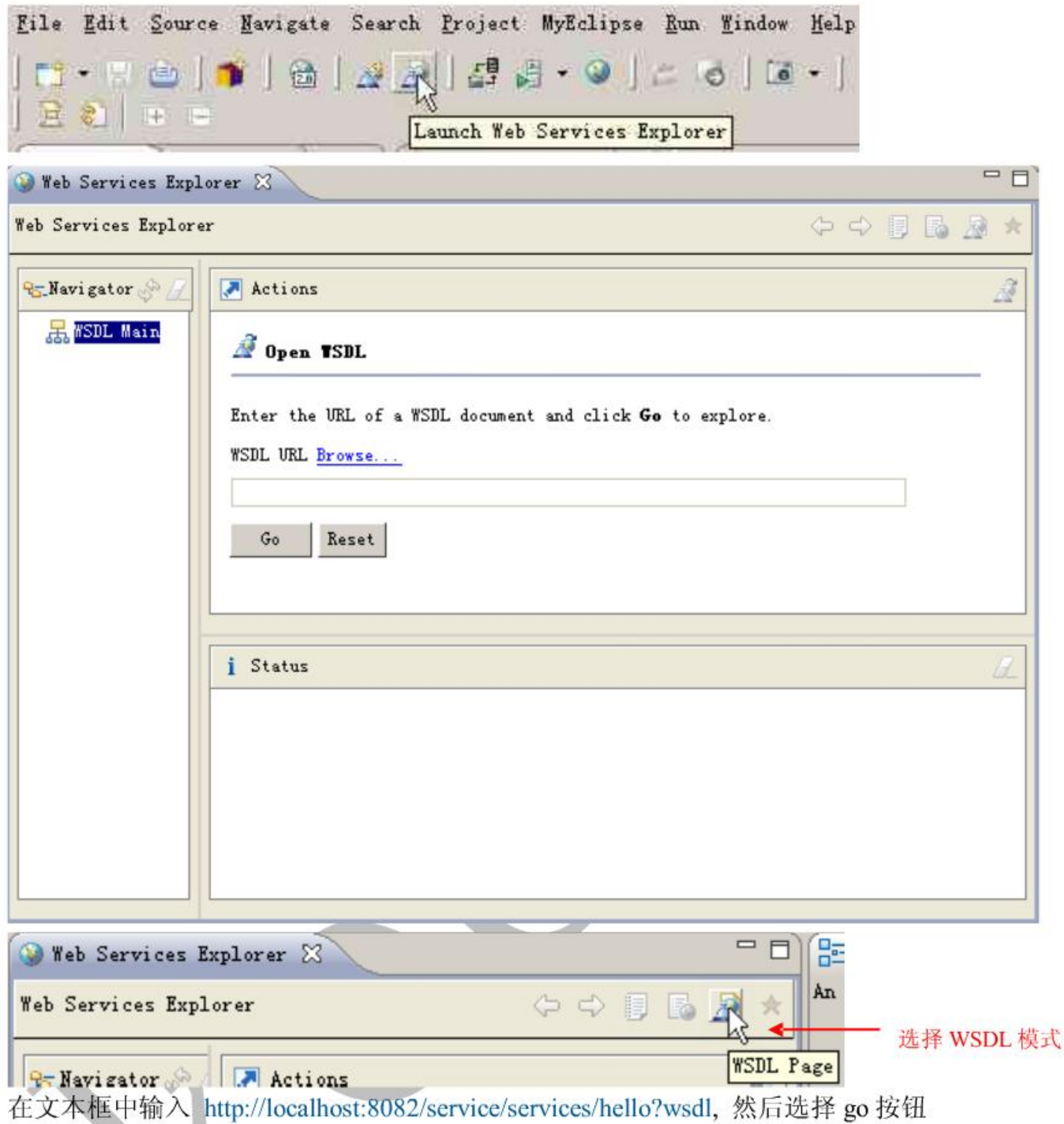
部署到 Tomcat 服务器, 然后启动服务 (注意: 这点很重要, 如果我们不部署到 web 容器中, 不启动 web 服务那么客户端是无法调用的)

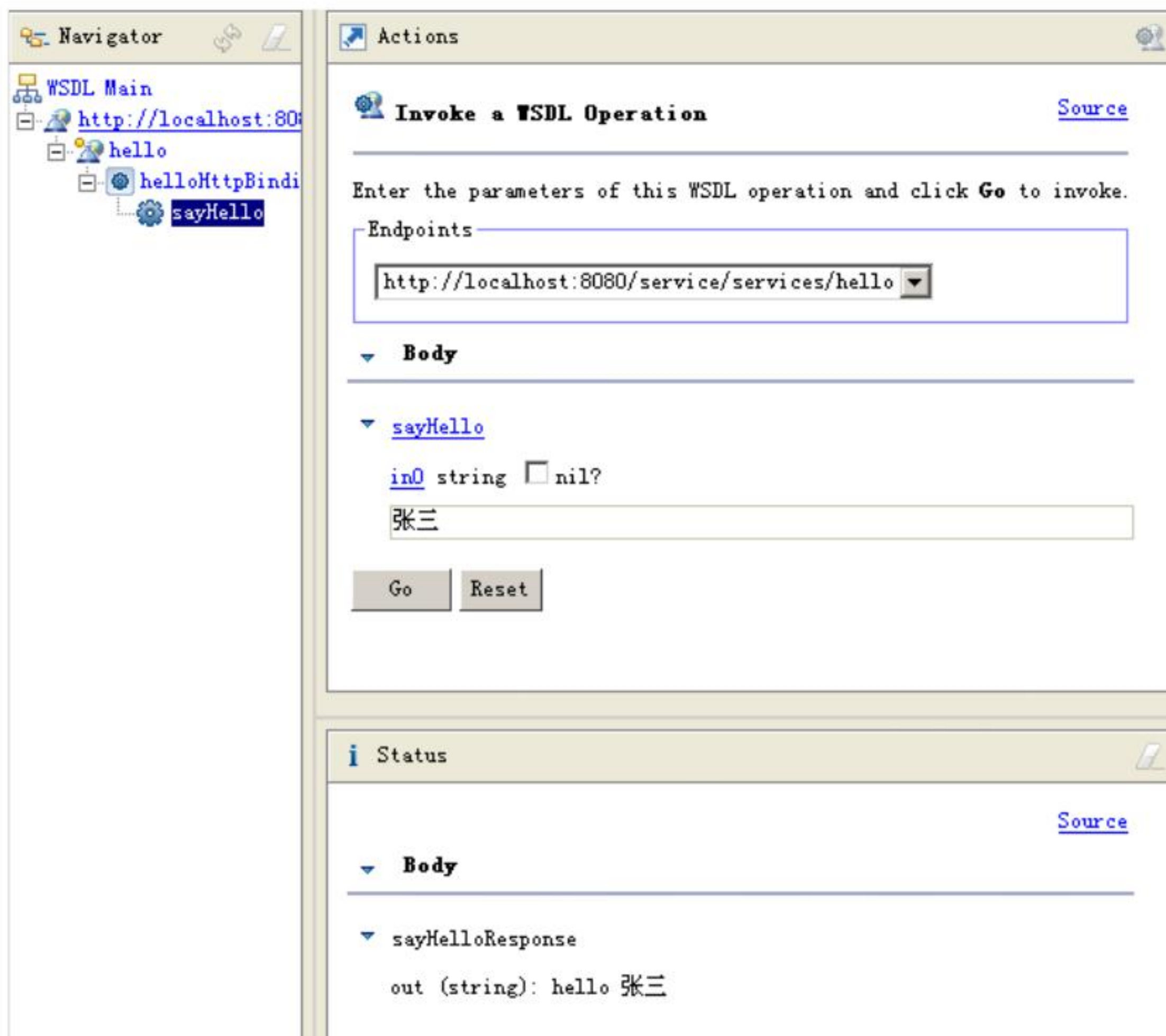


为了确认我们的 web 服务是否已经正常工作, 需要进行测试, 在地址栏中输入:



我们还可以通过 MyEclipse 提供的 web service 浏览器进行浏览





测试成功！至此我们的服务器端服务开发完毕！

webService 客户端开发【二】

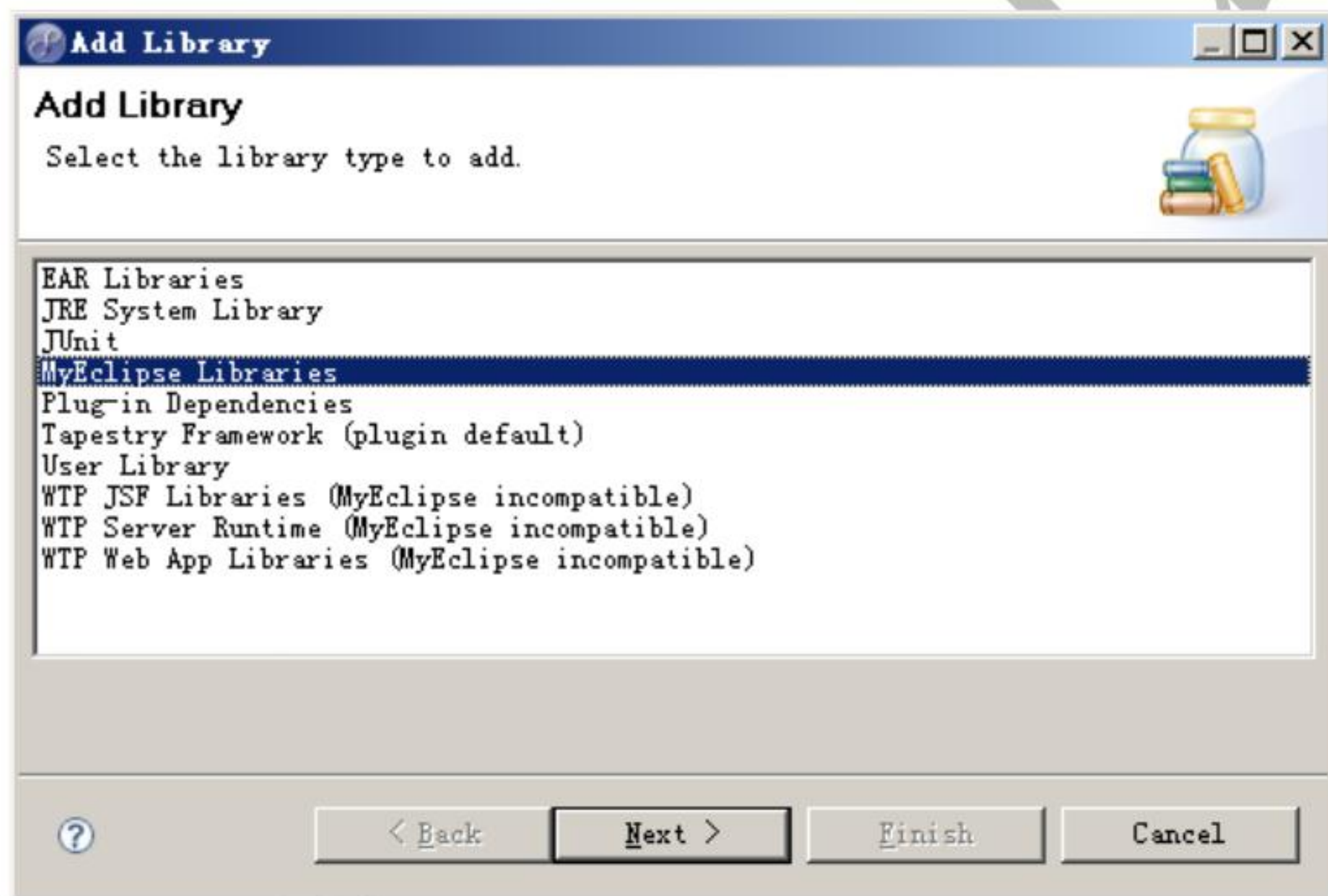
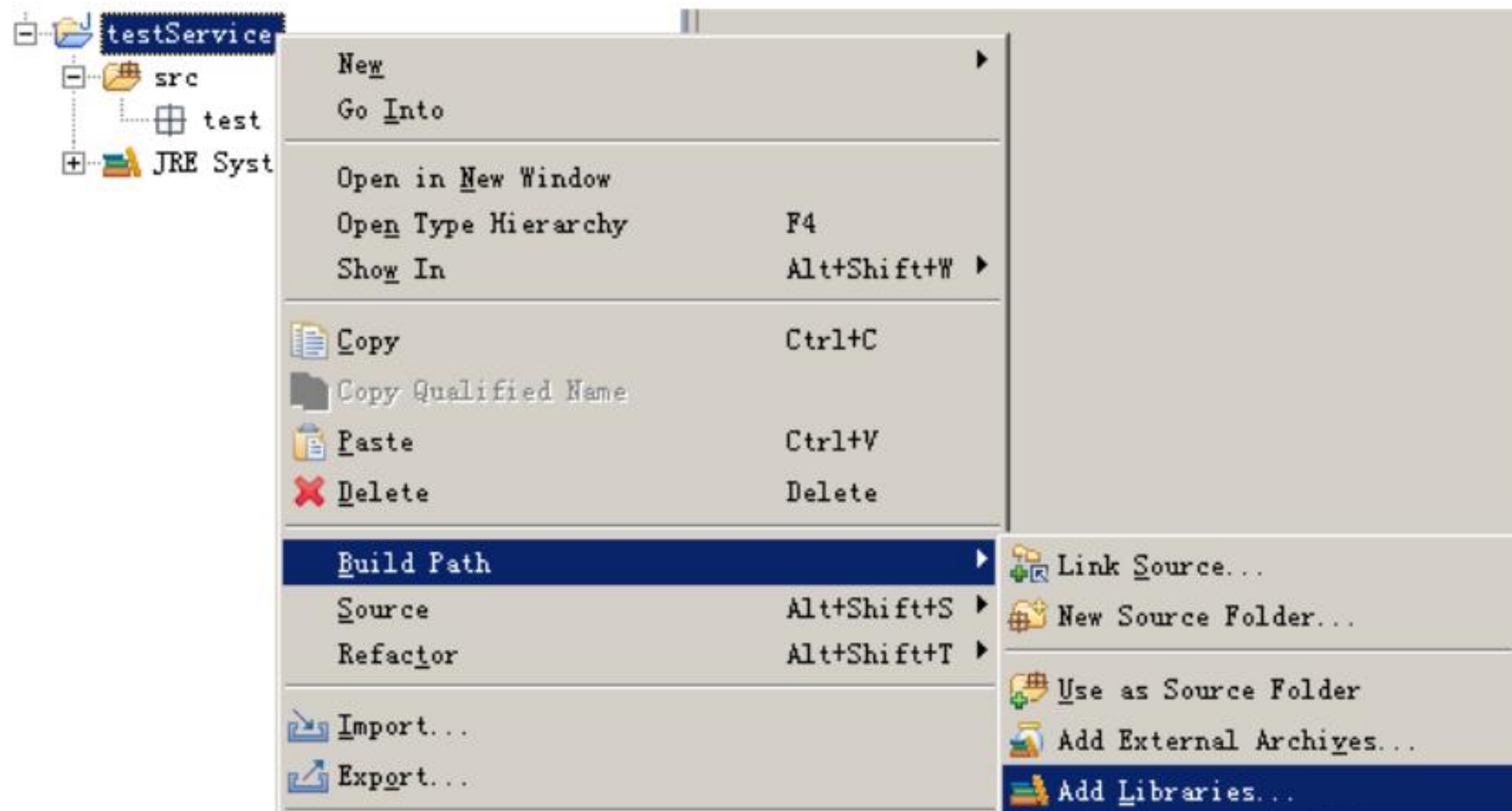
客户端：

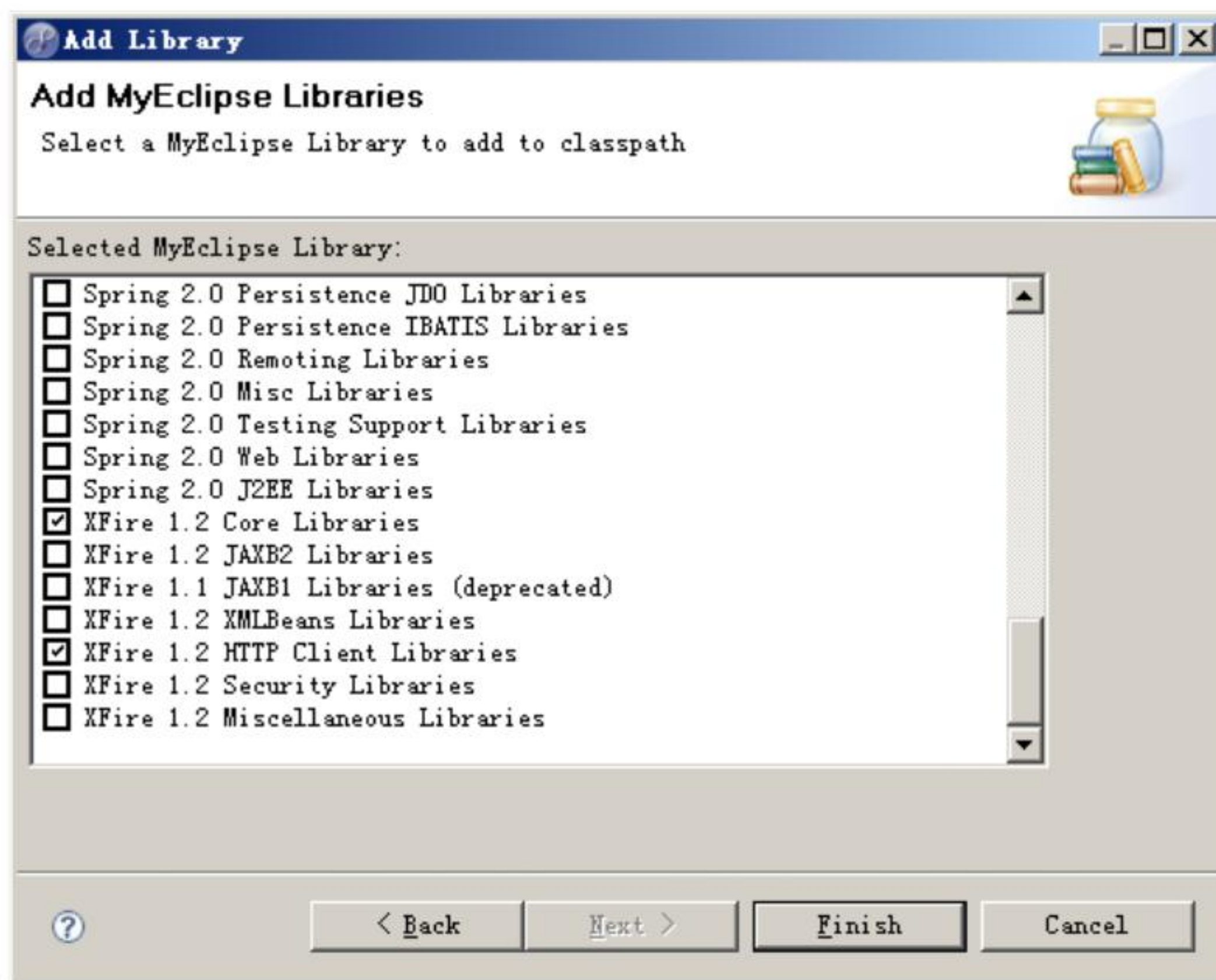
创建 webService 客户端应用程序

1. 新建一个 java 工程，web 工程也可以（Net 的也可以）



2. 添加 jar 包，因为需要使用到 XFire,所以需要将 XFire 的库文件加入





将 webservice 提供的接口加入到当前工程中



3. 创建 webservice 客户端

创建一个 TestService 类并添加 main 方法:


```
import java.net.MalformedURLException;
import org.codehaus.xfire.XFire;
import org.codehaus.xfire.XFireFactory;
import org.codehaus.xfire.client.XFireProxyFactory;
import org.codehaus.xfire.service.Service;
import org.codehaus.xfire.service.binding.ObjectServiceFactory;
import com.xaccp.service.Ihello;
public class TestService {
    public static void main(String[] args) {
        //创建服务的元数据
        Service serviceModel=new ObjectServiceFactory().create(Ihello.class);
        //创建服务代理
        XFire xfire=XFireFactory.newInstance().getXFire();
        XFireProxyFactory factory=new XFireProxyFactory(xfire);
        //服务地址
        String serviceURL="http://localhost:8080/service/services/hello";
        try {
            //服务代理通过服务元数据和服务地址 取得服务实例
            Ihello helloservice=(Ihello) factory.create(serviceModel,serviceURL);
            String serviceResponse=helloservice.sayHello("张三");
            System.out.println("服务返回的结果是: "+serviceResponse);
        } catch (MalformedURLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

输出:



The screenshot shows a console window titled 'Console' with a close button. It displays the output of a Java application: '<terminated> TestService [Java Applic' followed by '服务返回的结果是: hello 张三'.

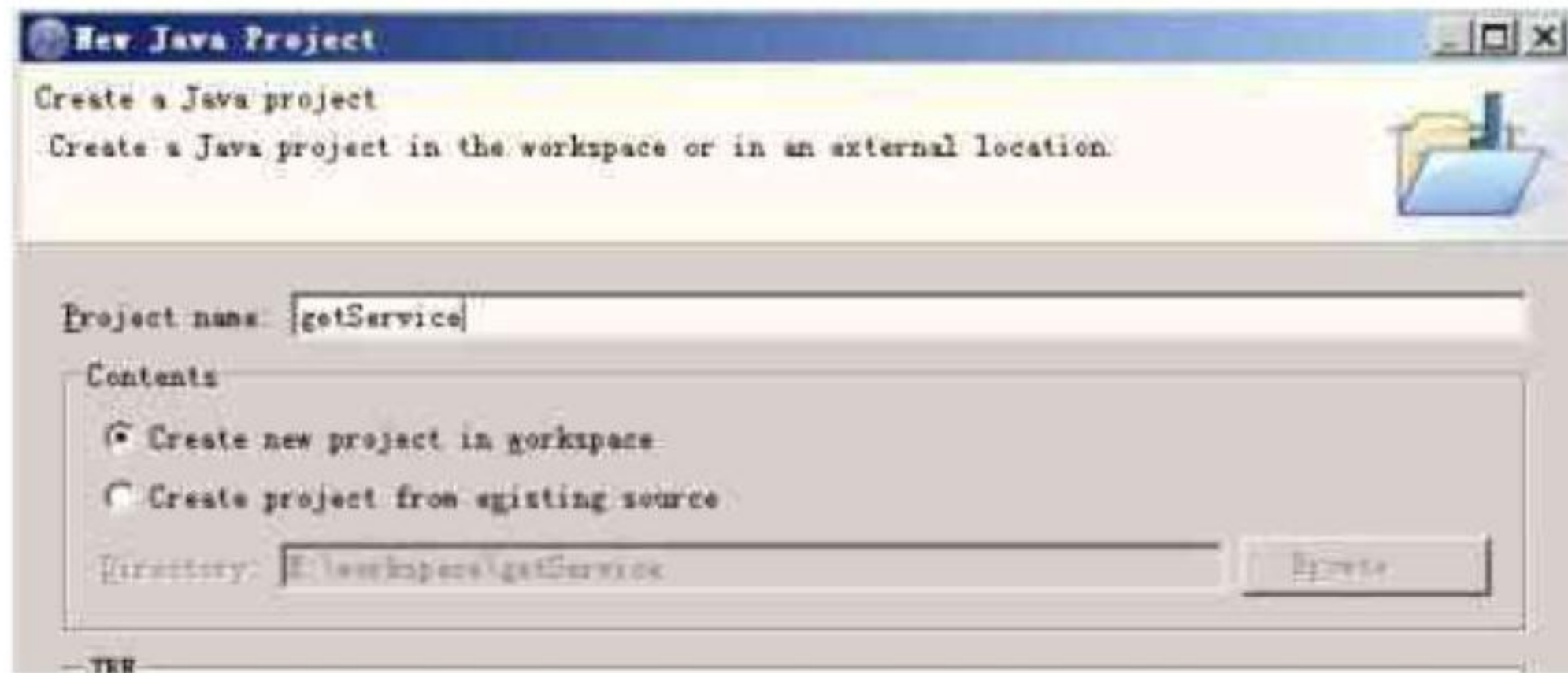
在 web 工程中采用同样的方法来调用 webservice 提供的服务

至此我们简单的 Web Service 示例开发完毕! 希望达到举一反三的效果!

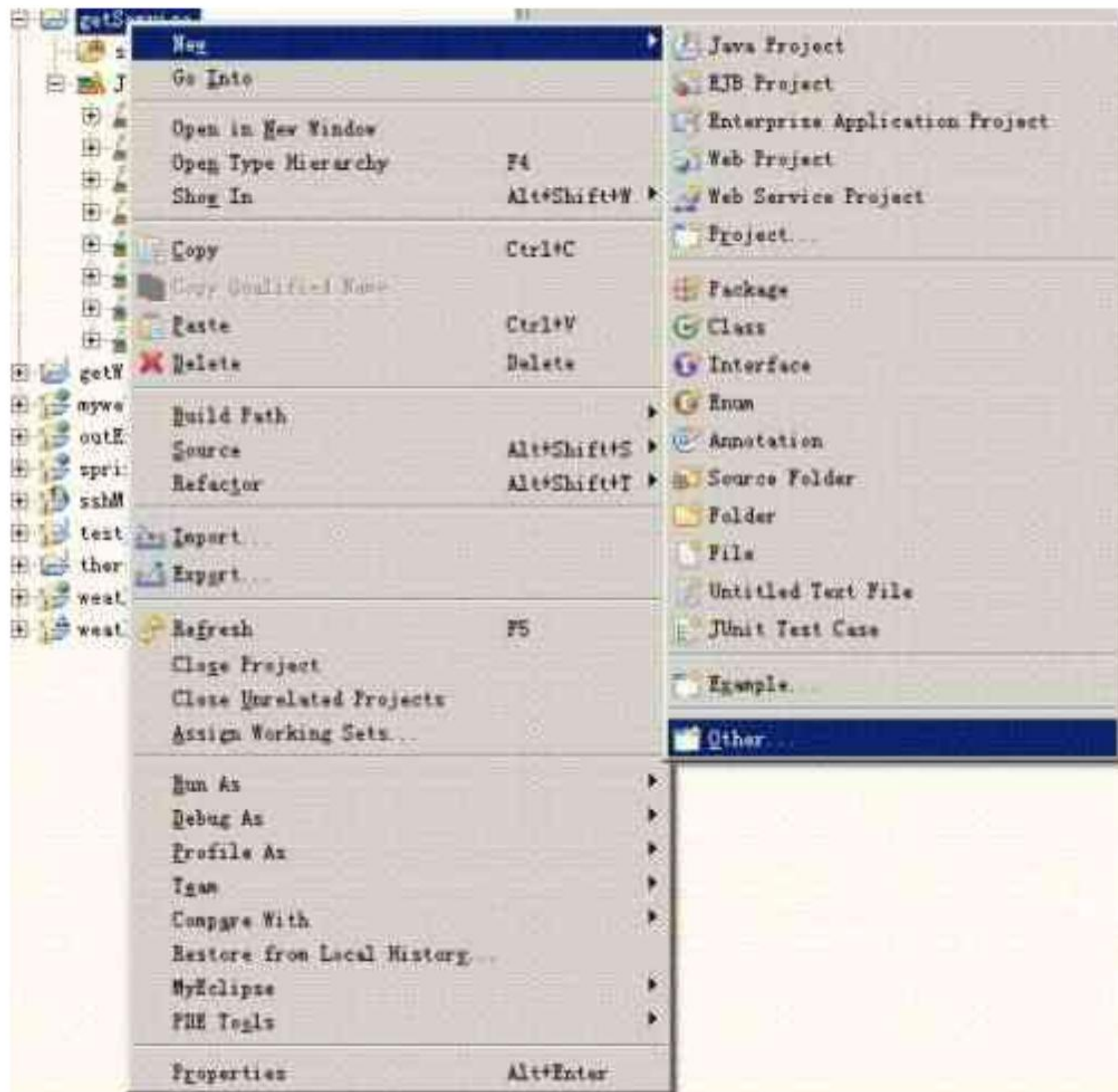
WebService 客户端开发【三】

在上面调用的时候代码很多并不很好记, 那么在介绍给大家一种方法

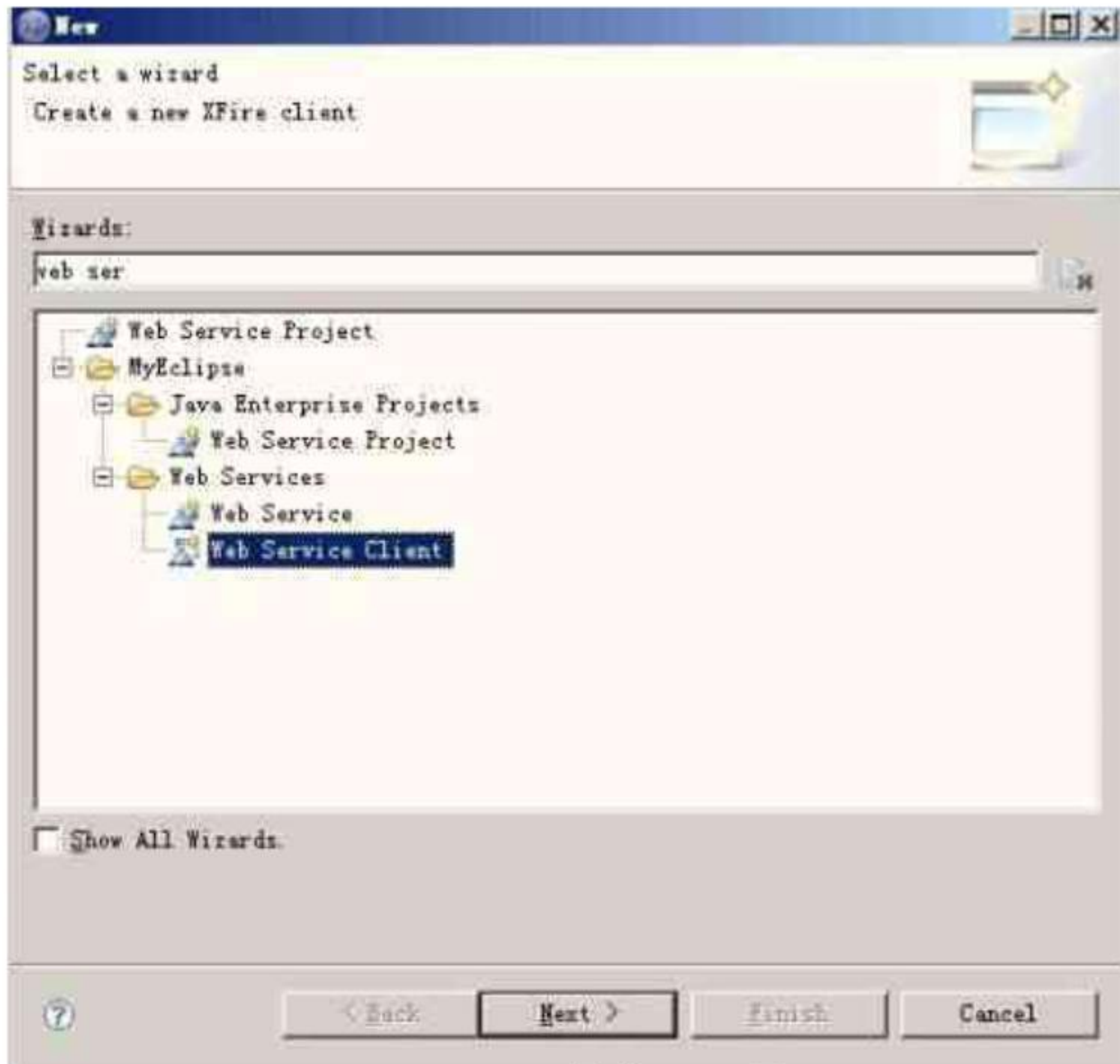
首先新建一个客户端工程 我们这里新建一个 java project



我们选择工程 点击右键 选择 new 选择 Other



选择 web service Client

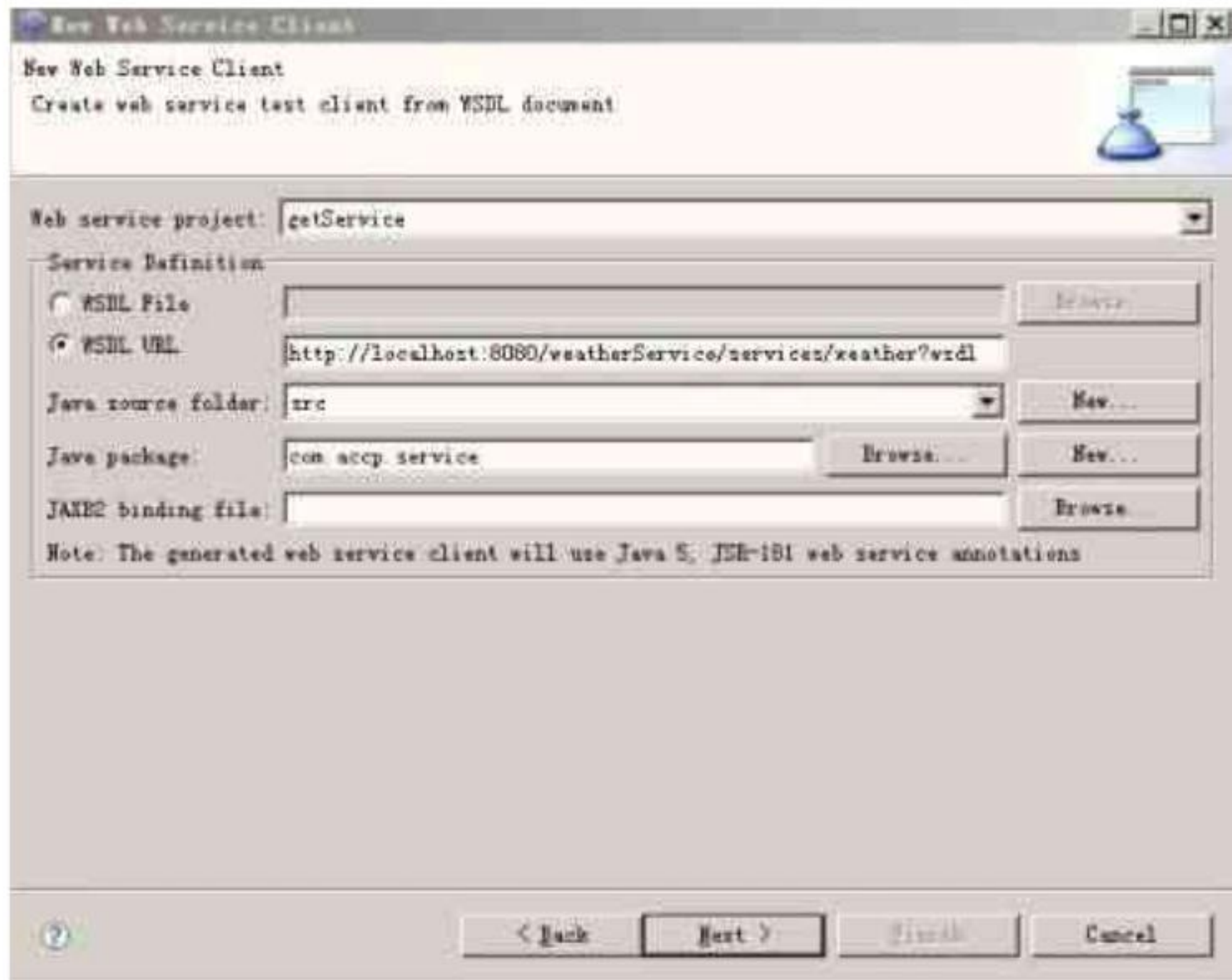


选择 web service project

选择 WSDL URL 填写服务地址

添加 java project

选择 Next 继而完成



看看我们的工程是不是多了很多的东西

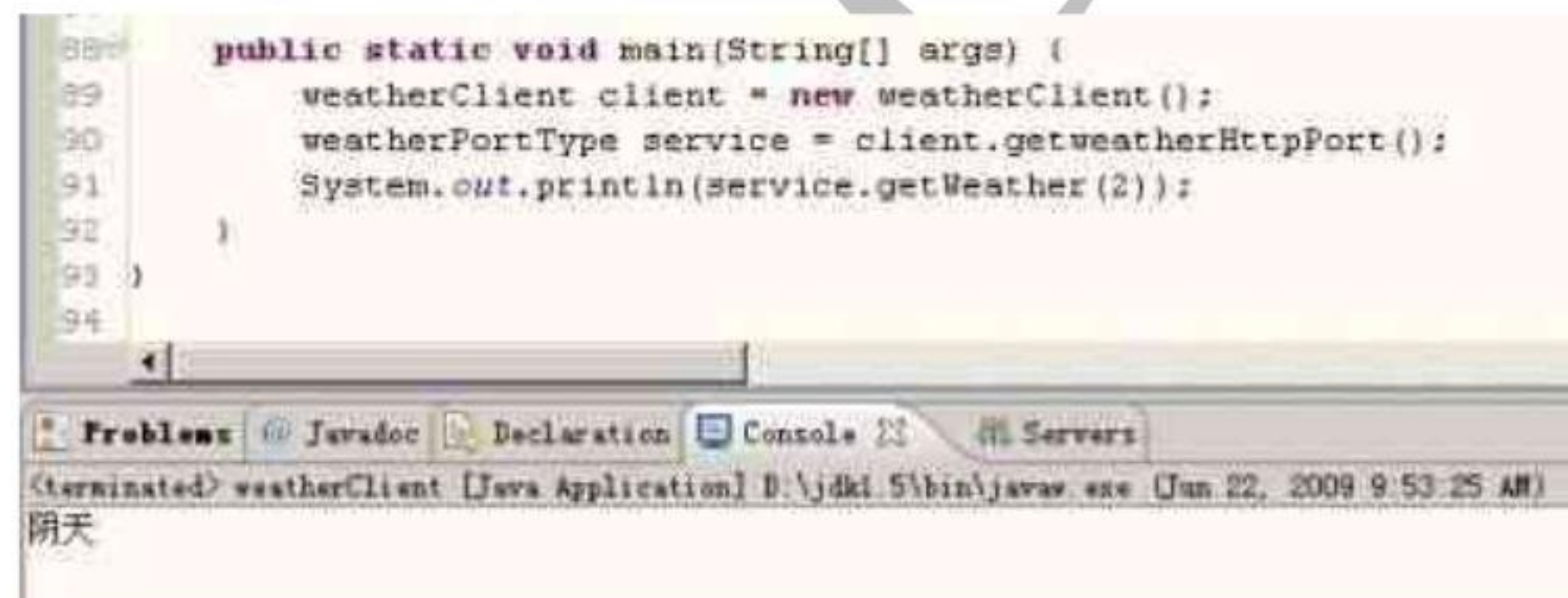


找一个以*Client 结尾的 java 类 并打开它

找到这个类中的 main 方法，在这里我们就可以进行对服务的调用

```
public static void main(String[] args) {  
  
    weatherClient client = new weatherClient();  
  
    //create a default service endpoint  
    weatherPortType service = client.getweatherHttpPort();  
  
    //TODO: Add custom client code here  
    //  
    //service.yourServiceOperationHere();  
  
    System.out.println("test client completed");  
    System.exit(0);  
}
```

编写代码 并运行



The screenshot shows an IDE window with a Java file. The code is as follows:

```
88 public static void main(String[] args) {  
89     weatherClient client = new weatherClient();  
90     weatherPortType service = client.getweatherHttpPort();  
91     System.out.println(service.getWeather(2));  
92 }  
93 }  
94
```

Below the code editor, the 'Console' tab is active, showing the output of the program:

```
<terminated> weatherClient [Java Application] D:\jdk1.5\bin\javaw.exe (Jun 22, 2009 9:53:25 AM)  
阴天
```

Ok 测试成功