



面向服务的体系结构实验报告

实验名称：	实验二：创建 SOAP Web Services
实验日期：	2023/10/8
实验地点：	文宣楼 B311
提交日期：	2023/10/8

学号：	32420212202930
姓名：	陈澄
专业年级：	软工 2021 级
学年学期：	2023-2024 学年第一学期

1. 实验环境

Windows 10 (64 位) + Oracle SOA Suite 12.2.1.4.0(64 位);

开发工具: JDeveloper

编程语言: Java

API: Java API for Web Services (JAX-WS)

构建工具: Ant

2. 实验目的

使用 JAX-WS 和相关工具来创建和发布 SOAP web service

理解并分析 HTTP Analyzer 编辑器中显示的 web service 的 URL,

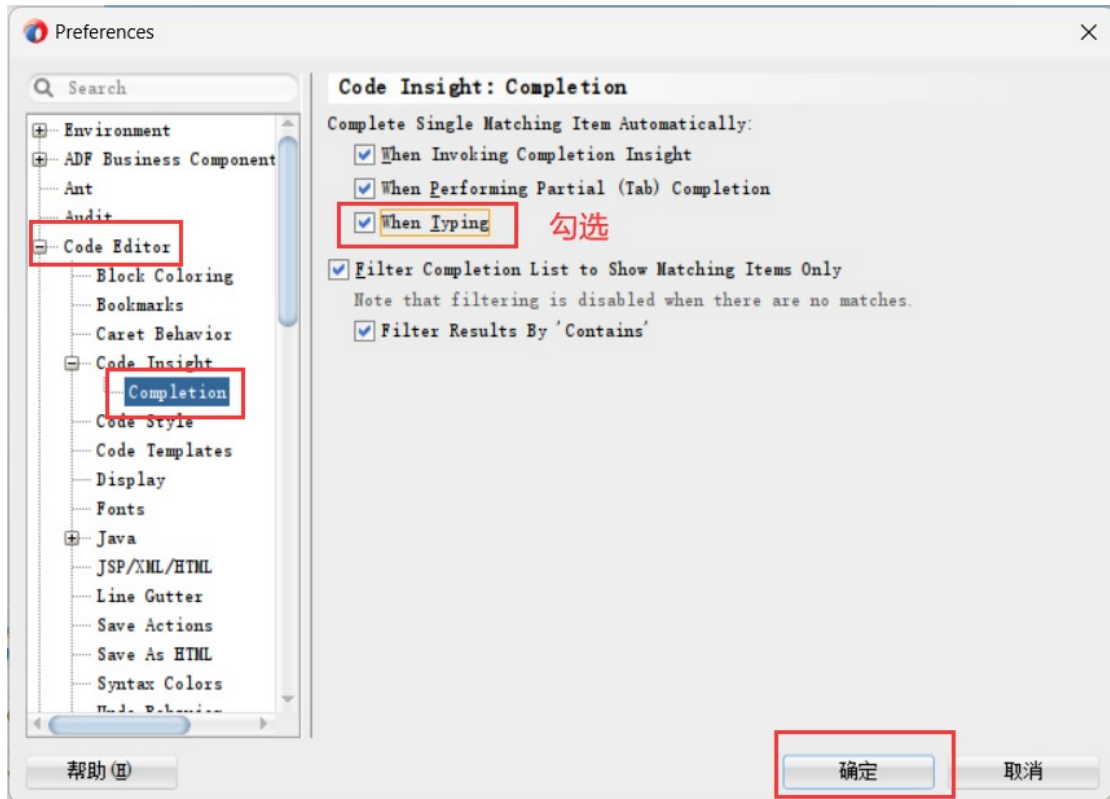
WSDL URL 和暴露的操作。

3. 实验内容和步骤

• 创建 SOAP web service 的步骤;

1. 概述

2. JDeveloper 的代码补全功能设置 (可选)

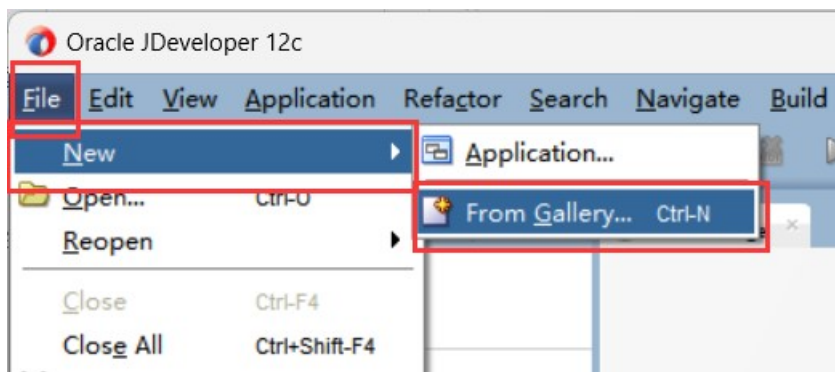


3. 使用 JDeveloper 创建 web services

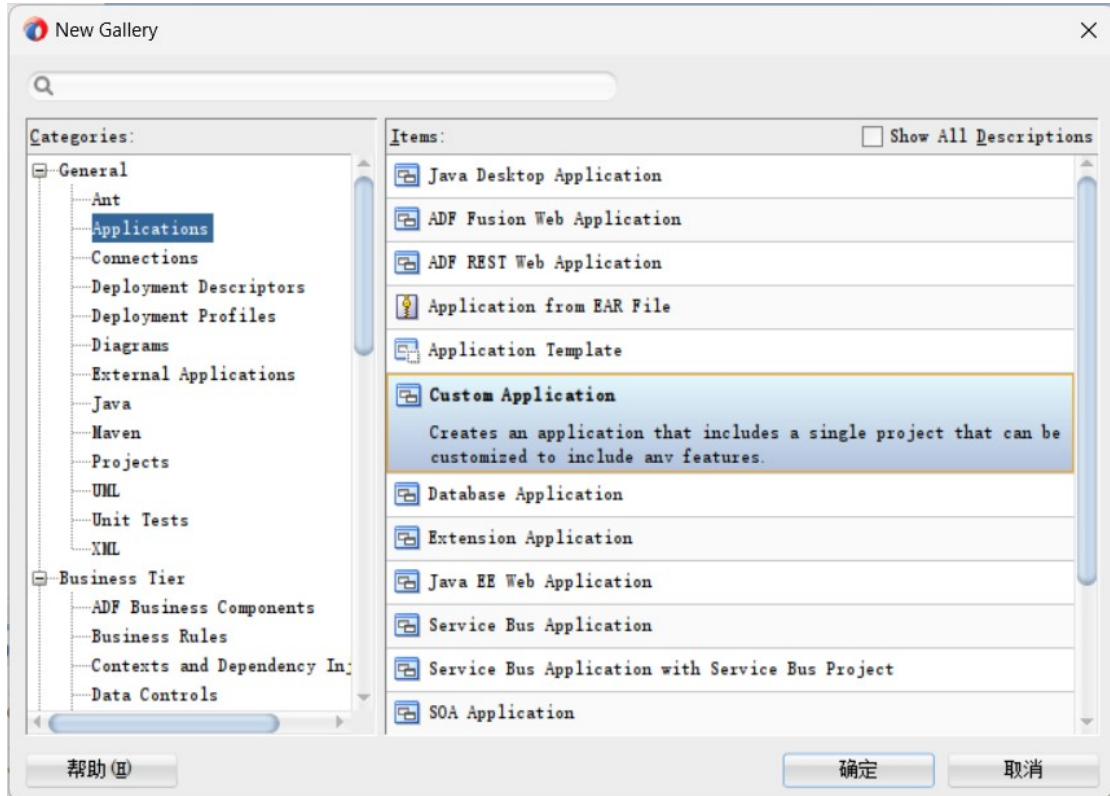
3.1 创建一个 web service

创建一个 Plain Old Java Object (POJO) 对象

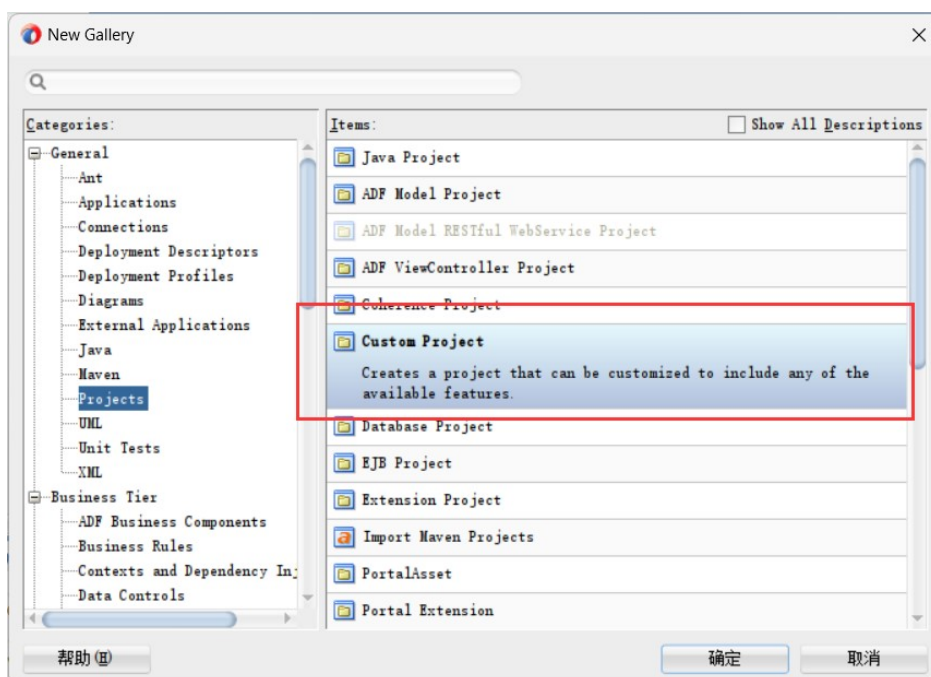
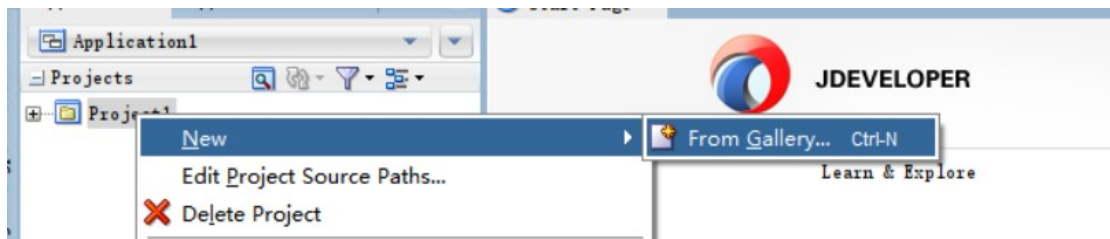
在 JDeveloper 中创建一个空项目

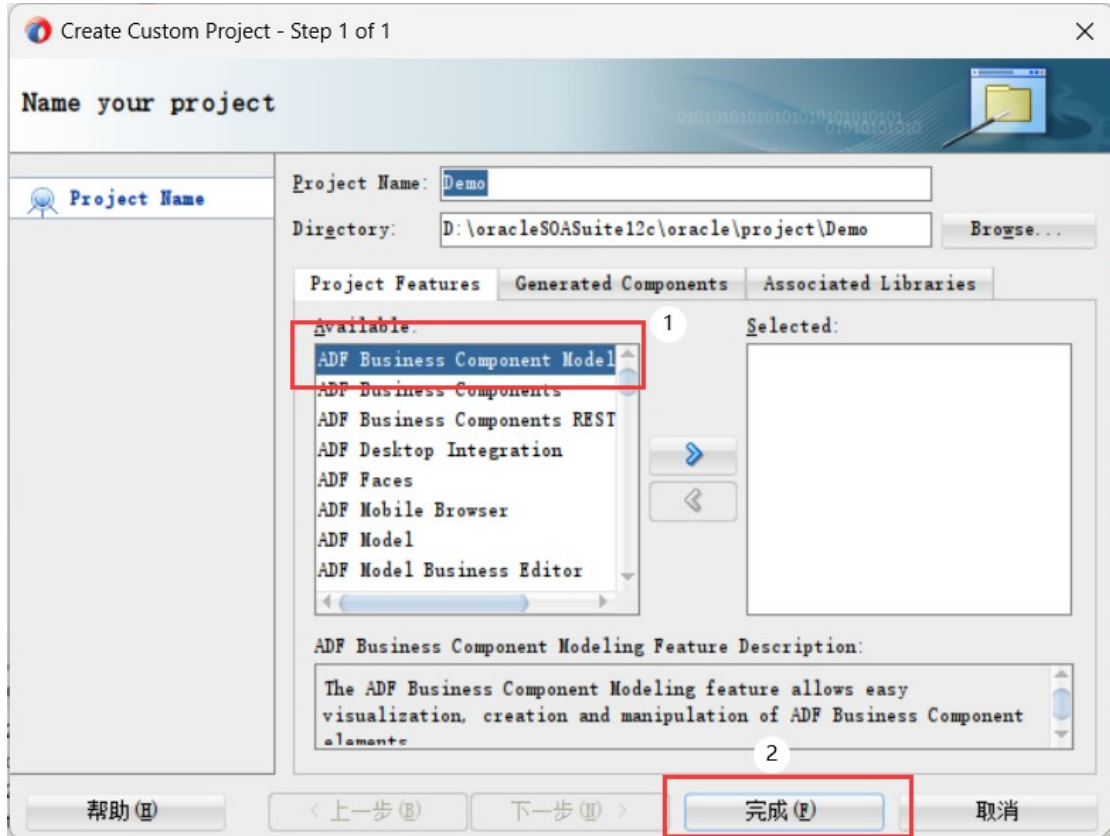


先创建一个 application

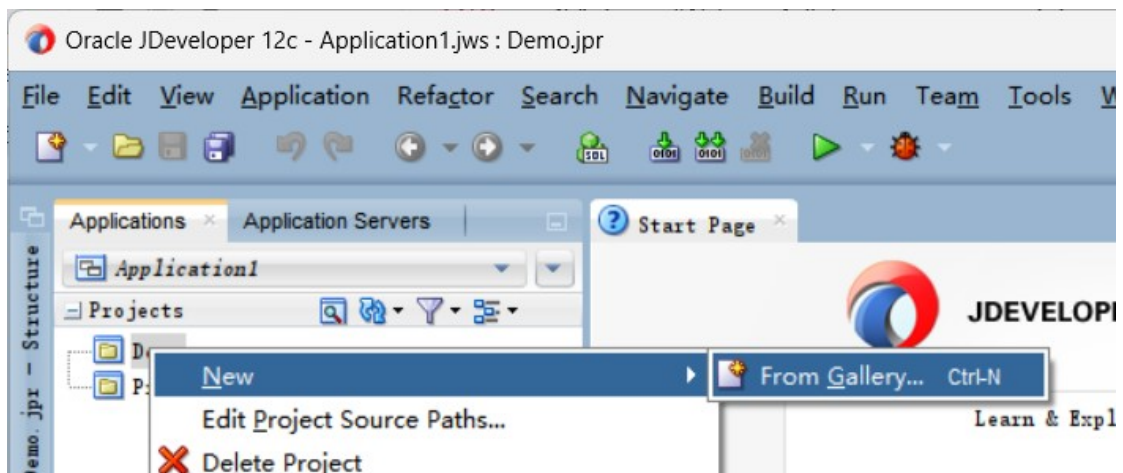


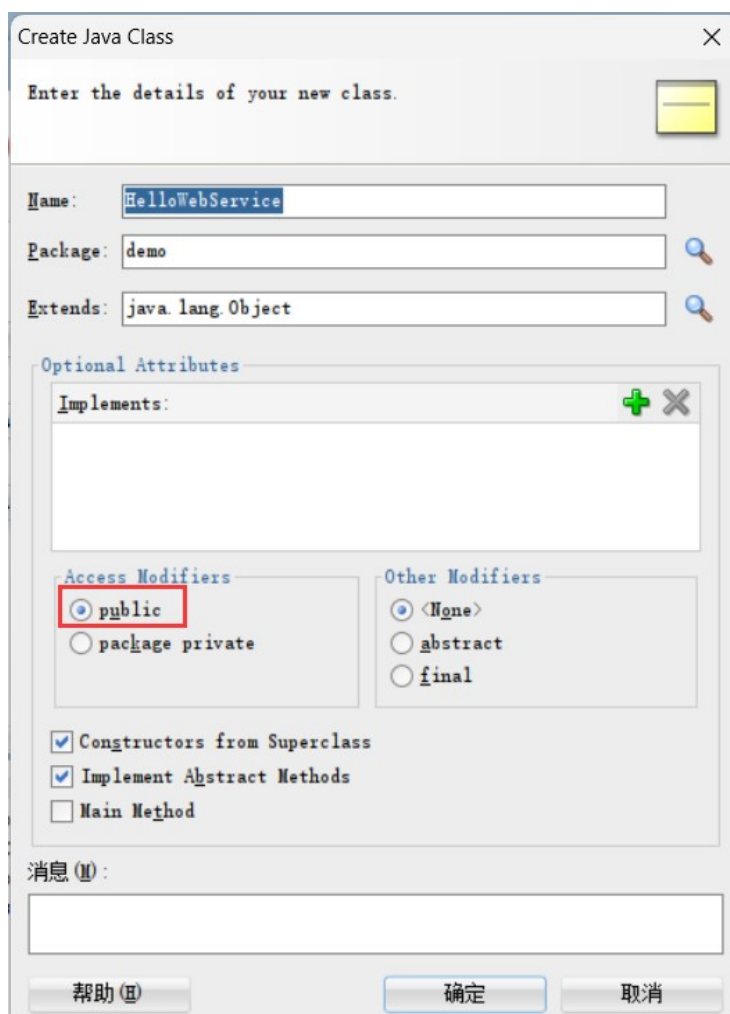
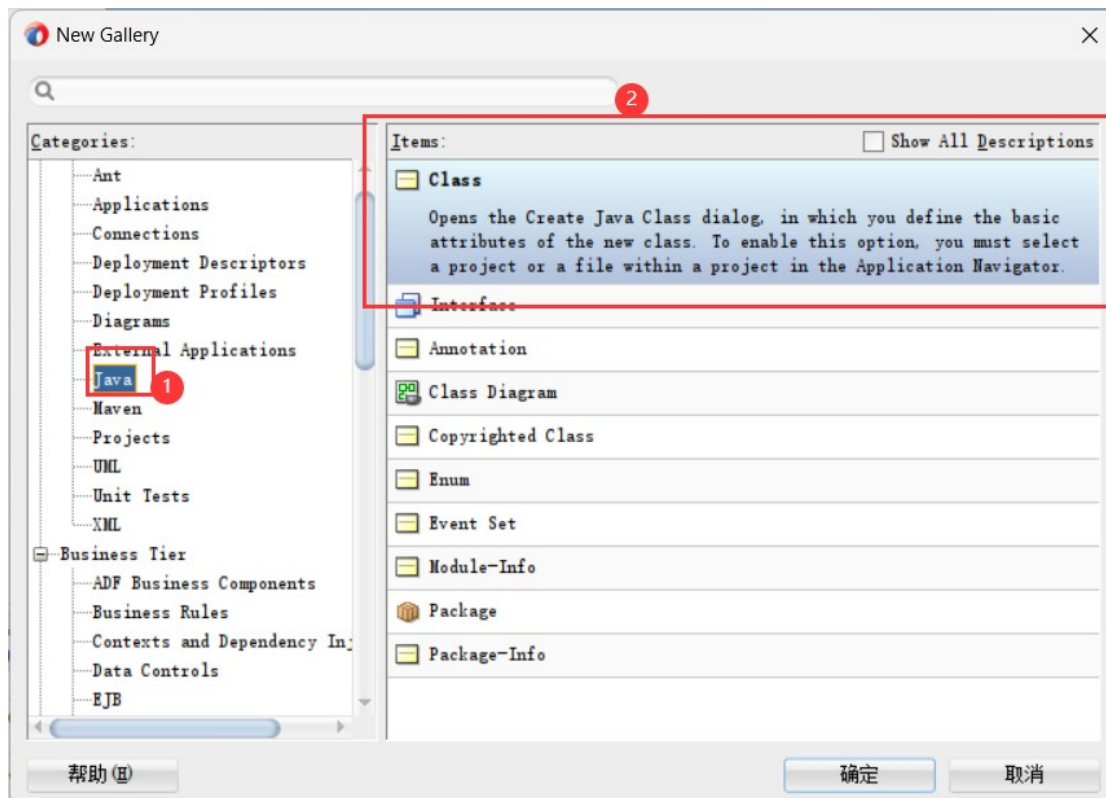
在 application 下创建一个 project



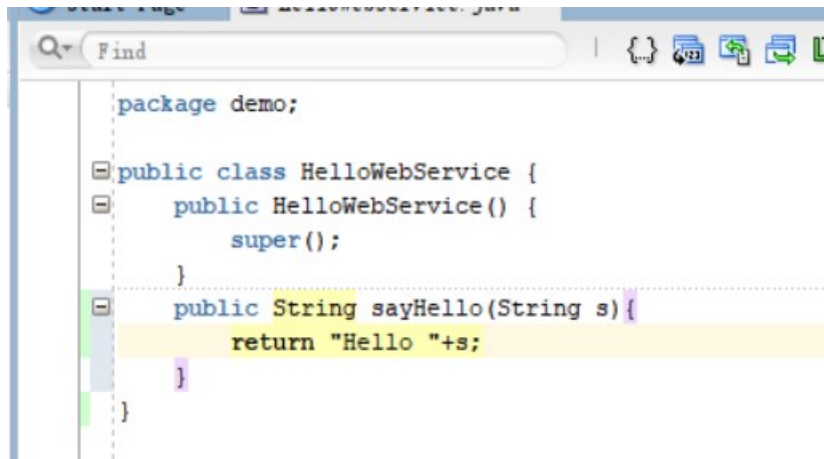


在新的 project 下创建一个 java 类





创建新的方法 sayHello 输入字符串 s 并返回” Hello “+s 用于后续测试

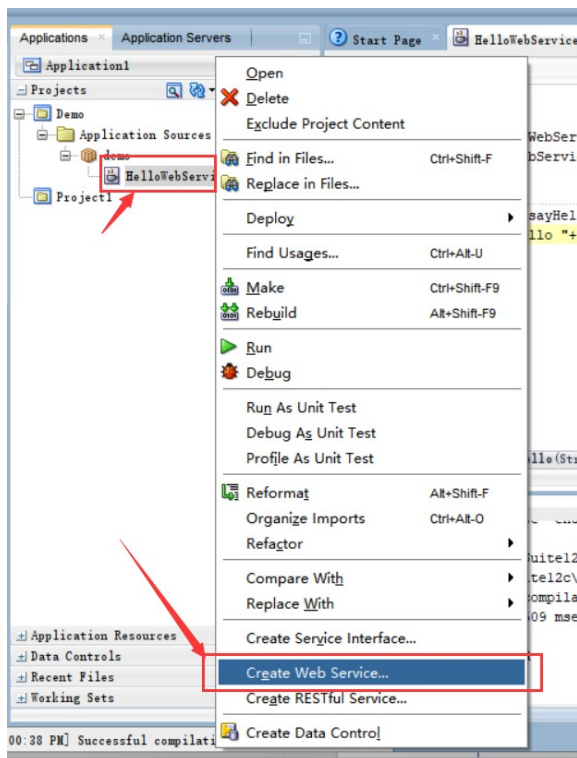


```
package demo;

public class HelloWebService {
    public HelloWebService() {
        super();
    }

    public String sayHello(String s) {
        return "Hello "+s;
    }
}
```

创建 web service 类



Create Java Web Service - 步骤 3/11

Generation Options

Select or type the Java class or Stateless Session EJB you want to publish (Note: If you type a name for which no class exists, you will be offered the option to have the wizard generate a default implementation class with that name).

Component To Publish:

demo.HelloWebService

Web Service Name:

MyWebService

Port Name:

MyWebServicePort

☐ Add SEI

Help (H) < 上一步 (B) 下一步 (N) > 完成 (F) 取消

Create Java Web Service - 步骤 4/10

Message Format

Use the controls on this page to choose the binding option and the settings that control the structure of the SOAP messages transmitted to and from the web service.

☐ SOAP 1.1 Binding

☒ SOAP 1.2 Binding

SOAP Message Format: Document/Wrapper

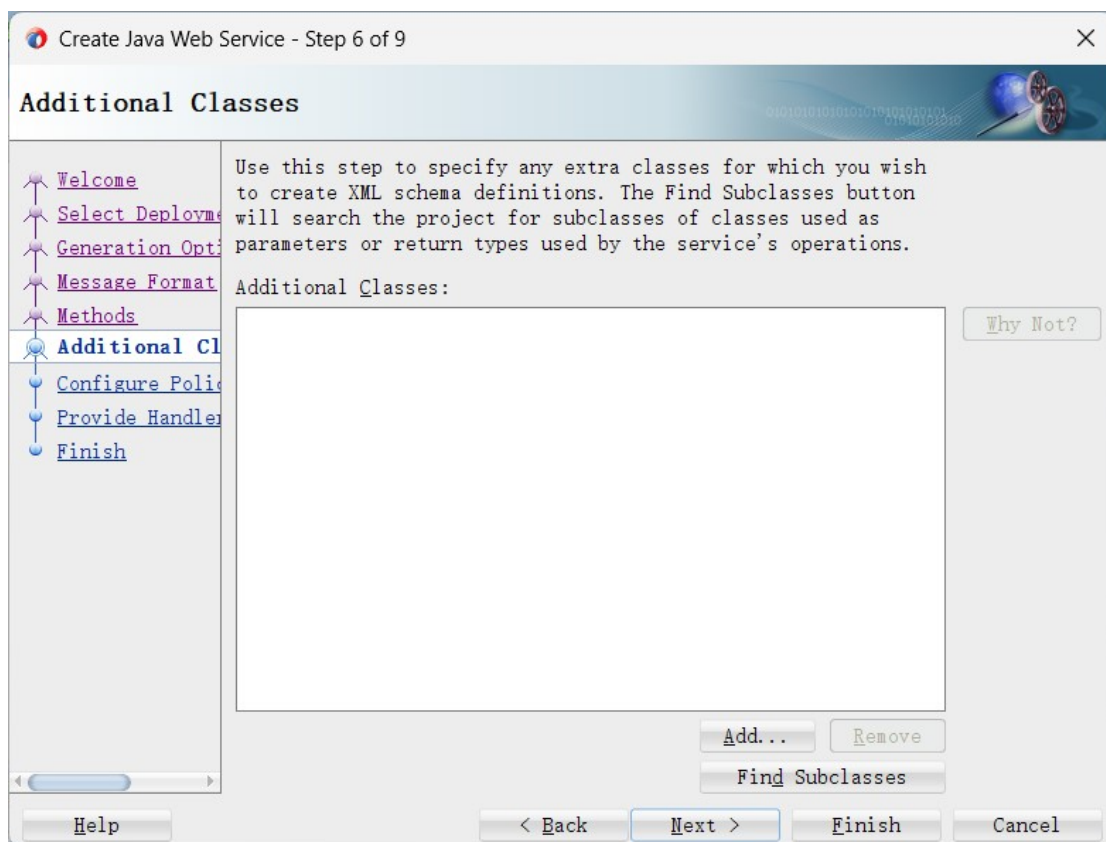
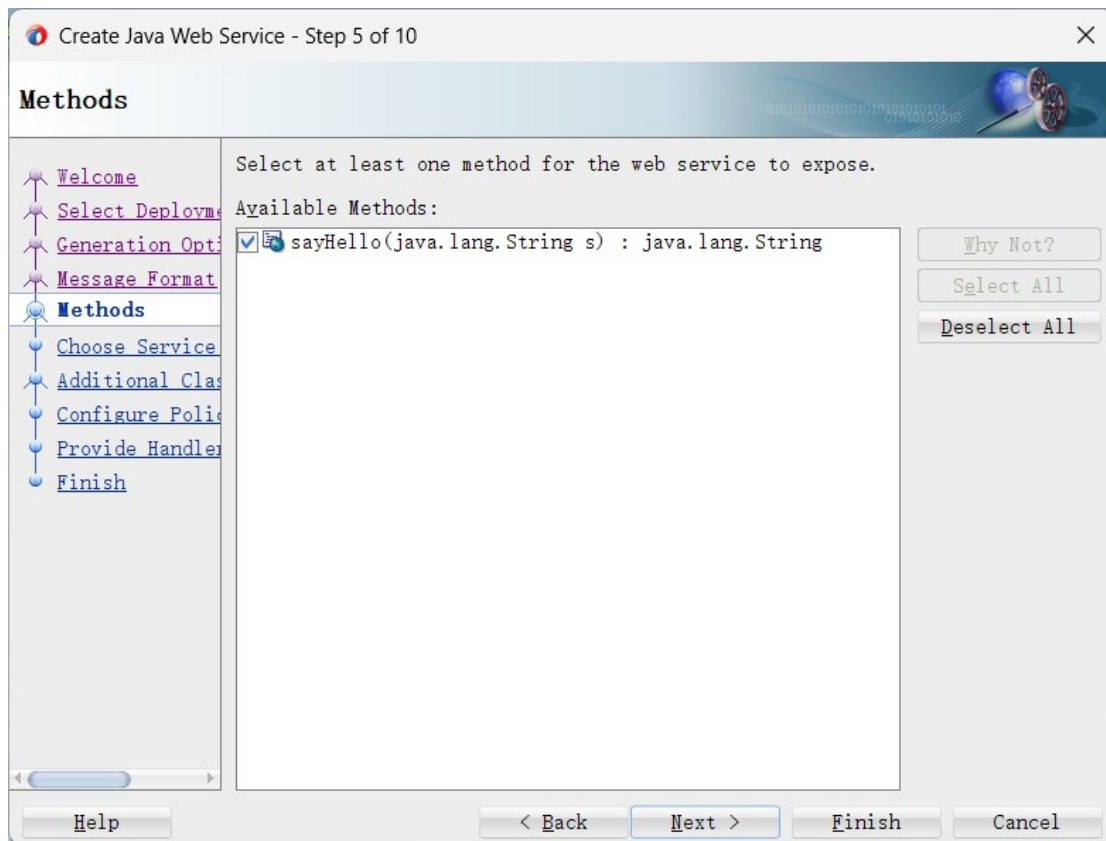
Binary encoding style for this service is MTOM. Enable MTOM to write MTOM annotation for this service.

☐ Enable MTOM

Select to enable Fast Infoset for this service

☐ Enable Fast Infoset


Help (H) < 上一步 (B) 下一步 (N) > 完成 (F) 取消



Create Java Web Service - Step 7 of 9

×

Configure Policies



Welcome

Select Deployment

Generation Options

Message Format

Methods

Additional Class Files

Configure Policies

Provide Handler Details

Finish

In this page you decide the security policy option to use for the web service. Based on the selection the appropriate policy configuration page will be shown.

☐ QWSM Policies
 ☐ WLS Policies
 ☒ No Policies

Help

< Back

Next >


Finish

Cancel

Create Java Web Service - Step 8 of 9

×

Provide Handler Details



Welcome

Select Deployment

Generation Options

Message Format

Methods

Additional Class Files

Configure Policies

Provide Handler Details

Finish

Specify any handler classes you have which will deal with the web service message. The defined handlers may have associated initialization parameters, SOAP roles or SOAP headers.

Ports:

Defined Handlers:

Add...

Remove

Details

Init Params

Roles

Handler Name:

Handler Class:

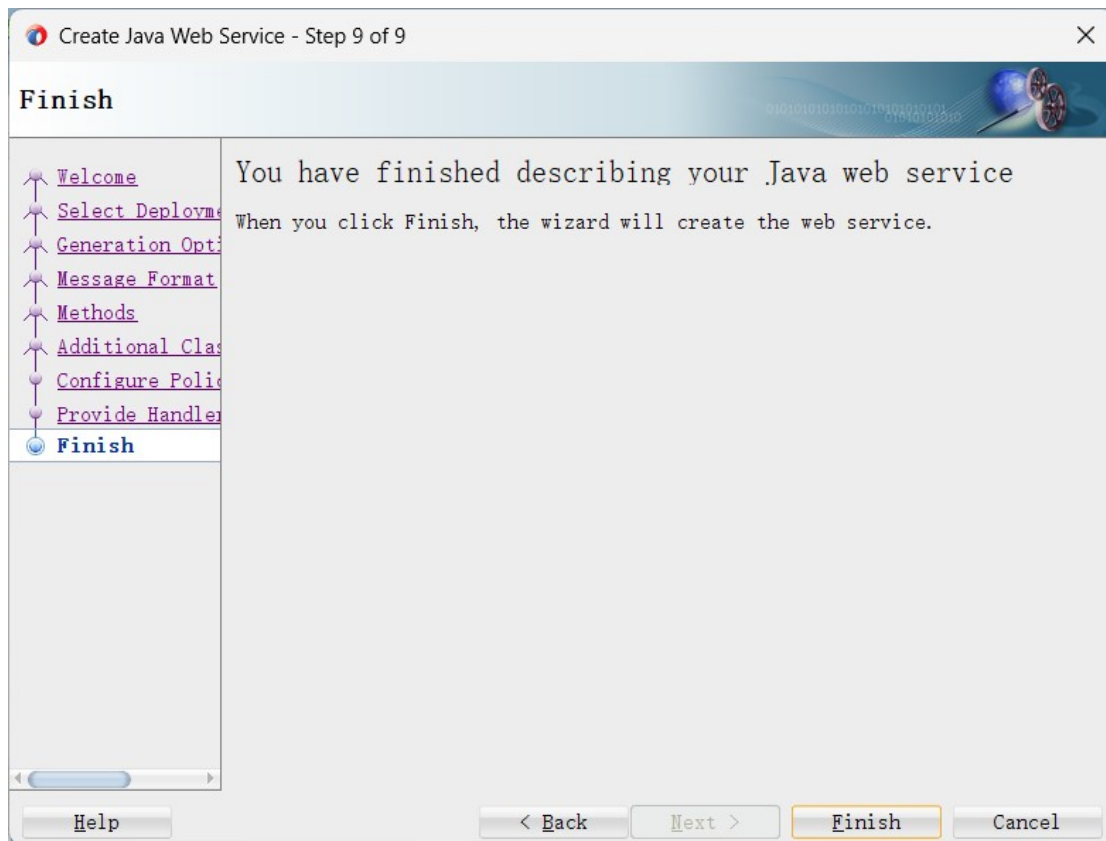
Help

< Back

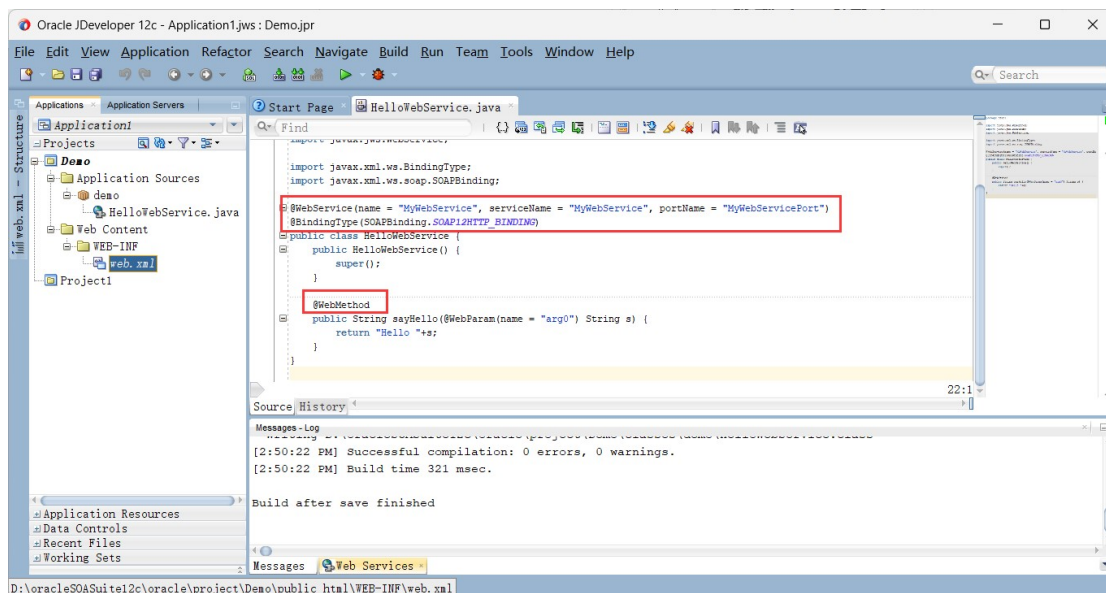
Next >

Finish

Cancel



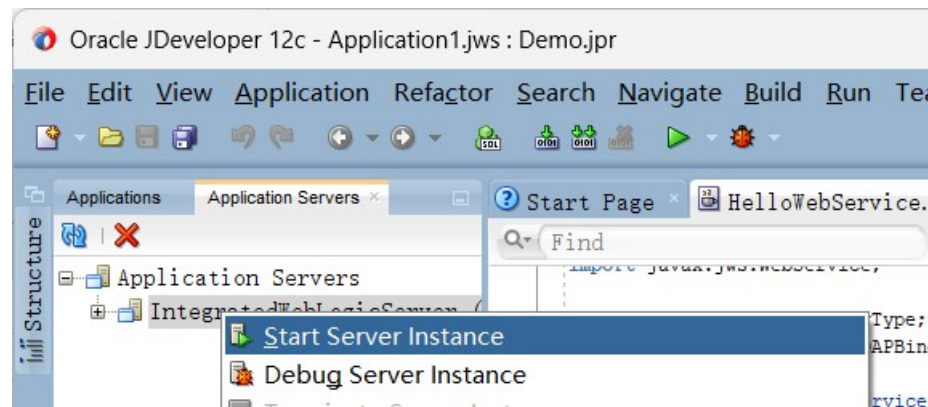
原 java 类中添加了新的注释 annotations: @WebService、@BindingType、@WebMethod 和@WebParam。



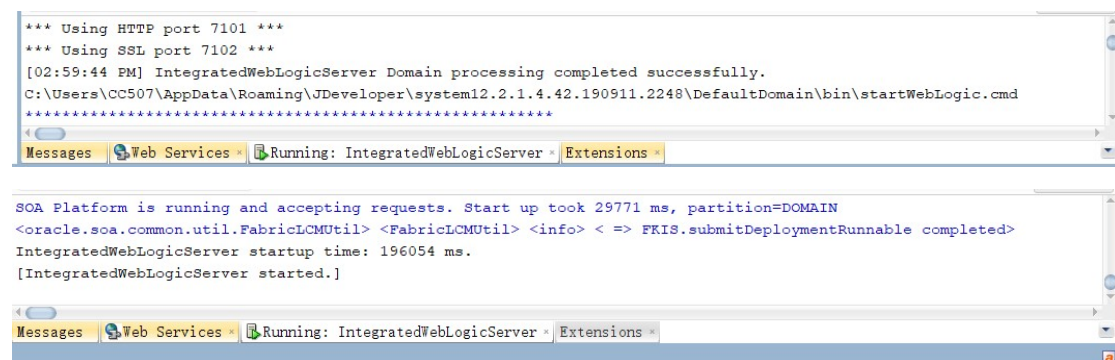
- 测试所创建的 web service 是否成功;

3.2 测试 web service

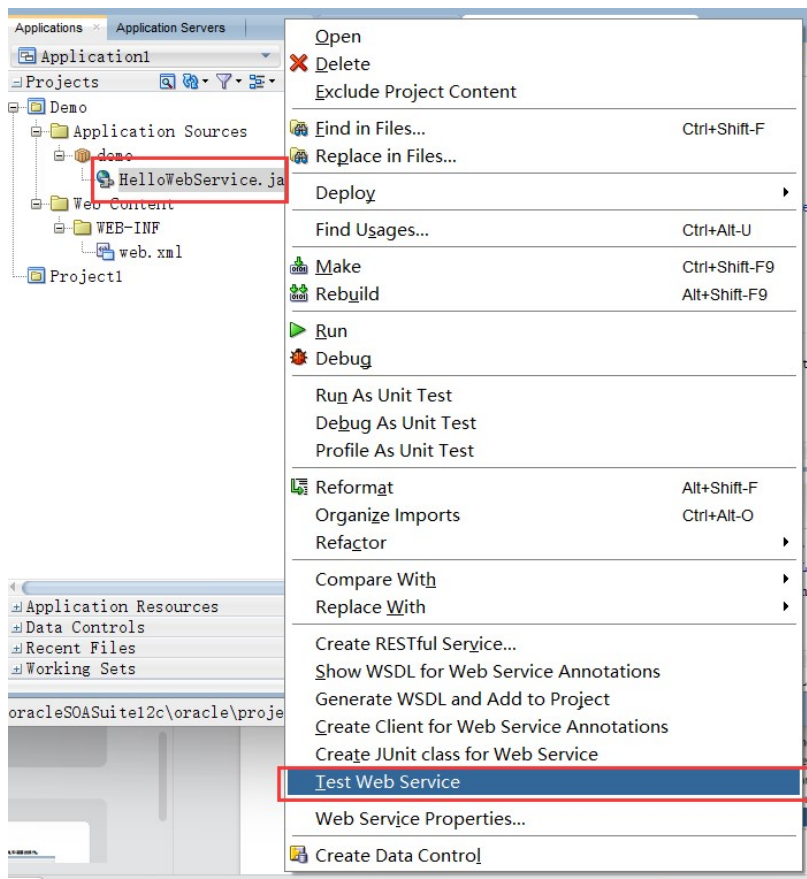
步骤一：启动 IntegratedWebLogicServer



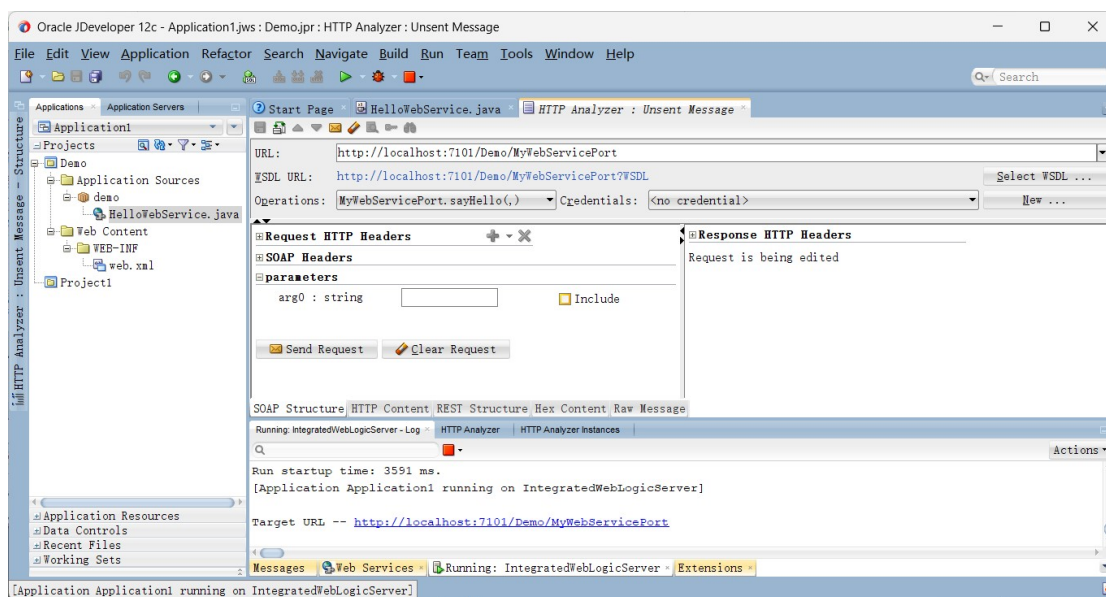
显示 IntegratedWebLogicServer started 表示成功启动



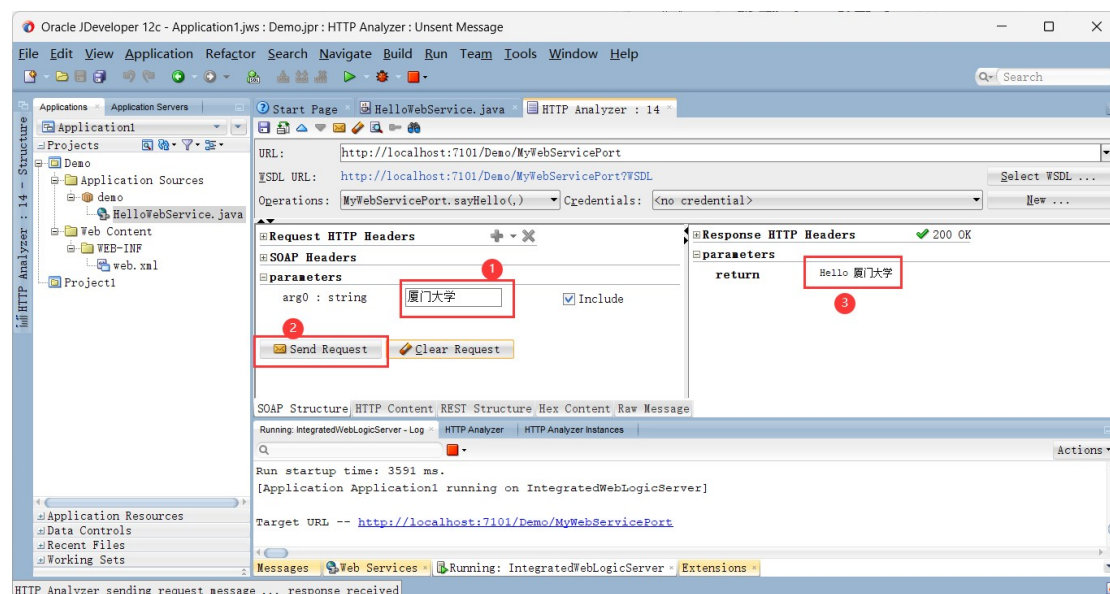
步骤二：测试 web service



在 HTTP Analyzer 编辑器中显示了 web service 的 URL，WSDL URL 和暴露的操作



在请求区域，在 arg0 字段输入名字（如，厦门大学）点击【send request】，在响应区域就可以看到结果



至此，自底向上创建 web service 的方法就结束了。

- **web service 有哪些类型？列出它们的主要异同点；**

1. SOAP Web 服务

SOAP (Simple Object Access Protocol) 是基于 XML 的协议，用于在 Web 环境中进行信息交换。SOAP Web 服务使用 WSDL (Web Services Description Language) 描述语言来定义服务接口，支持多种标准化协议如 HTTP、FTP 等，并可支持广泛的平台和编程语言。SOAP Web 服务具有较高的可靠性和安全性，但也因其基于 XML 的文本格式而导致处理效率较低。

2. RESTful Web 服务

REST (Representational State Transfer) 是一种轻量级的架构风格，常用于 Web 服务的设计和开发。RESTful Web 服务基于 HTTP 协议，使用标准化的 HTTP 方法如 GET、POST、PUT 和 DELETE 来实现对资源的 CRUD (Create、Read、Update、Delete) 操作。RESTful Web 服务具有简单易用、高效、扩展性强等特点，但需要依赖于 URI

(Uniform Resource Identifier) 来定义资源和请求方式。

3. XML-RPC Web 服务

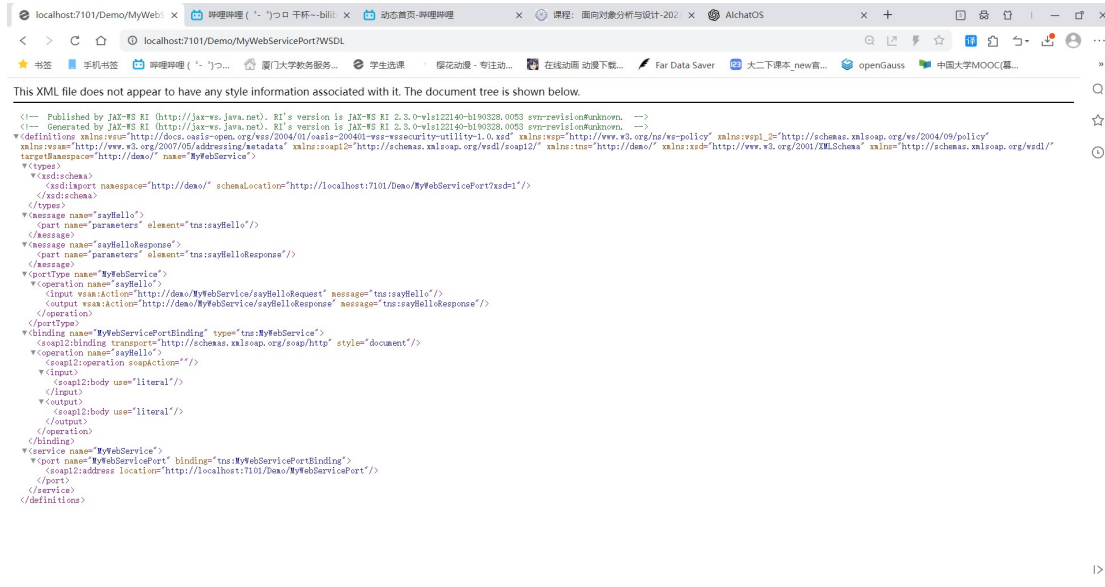
XML-RPC (XML Remote Procedure Call) 是一种 RPC (Remote Procedure Call) 通信协议，使用 HTTP 作为传输协议，将请求和响应数据封装在 XML 格式之中。XML-RPC Web 服务可以使用多种编程语言实现，传输数据体积相对较小，具有适应性强和易于实现等优点。但在安全性方面存在一些缺陷，例如采用明文传输、不支持数字证书等规范引入的安全问题。

4. JSON-RPC Web 服务

JSON-RPC (JavaScript Object Notation Remote Procedure Call) 是一种 RPC 通信协议，使用 JSON (JavaScript Object Notation) 作为封装格式，将请求和响应数据序列化后进行传输。JSON-RPC Web 服务具有轻量级、易于解析、可读性高等特点，常用于 Web 应用程序和移动应用 API 的开发。同时，与 XML-RPC 类似，JSON-RPC Web 服务也存在安全性方面的弱点。

总体来说，SOAP Web 服务比较重量级和复杂，可靠性和安全性较高，适用于企业级应用程序，而 RESTful Web 服务更加灵活、简单，用于构建轻量级应用程序和服务端 API 接口；XML-RPC 和 JSON-RPC Web 服务则适用于传输小量数据的服务端组件，如微服务、嵌入式设备等。

- **在文本编辑器或浏览器中打开 WSDL 文档，分析 WSDL 的各部分内容**



1. targetNamespace

targetNamespace 指定了 XML 文档中的命名空间，用于标识该 Web 服务所属的命名空间。在本例中，targetNamespace 为“http://demo/”。

2. types

types 元素可用于定义 Web 服务中使用的数据类型。在本例中，types 包含一个 xsd:schema 元素，用于导入名为

“MyWebServicePort?xsd=1”的 XSD 模式。

3. message

message 元素用于定义 Web 服务中请求和响应所使用的消息。在本例中，定义了两个消息，一个是请求消息“sayHello”，另一个是响应消息“sayHelloResponse”。

4. portType

portType 元素定义了 Web 服务端点中可用的操作及其输入、输出消息。在本例中，portType 定义了一个操作“sayHello”，该操作具有一个输入消息“sayHello”和一个输出消息“sayHelloResponse”。

5. binding

binding 元素用于定义 Web 服务端点的消息格式和协议细节，即绑定端口类型到消息格式和协议上。在本例中，binding 定义了一个名为“MyWebServicePortBinding”的绑定，采用 soap12 格式，并定义了一个操作“sayHello”，该操作具有一个输入消息和一个输出消息。

6. service

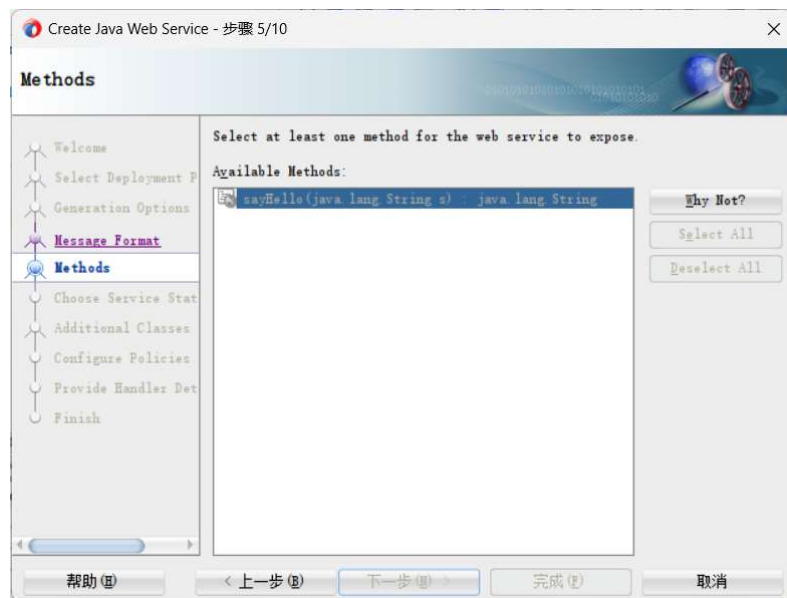
service 元素定义了 Web 服务的名称以及包含的端点（即提供 Web 服务的地址）。在本例中，定义了一个名为“MyWebService”的服务，该服务包含了一个名为“MyWebServicePort”的端点，连接到“http://localhost:7101/Demo/MyWebServicePort”地址。

4. 实验总结(完成的工作、对实验的认识、遇到的问题及解决方法)

通过本次实验，学会了使用 JDeveloper 创建一个 SOAP 类型的 Web Service，以及如何测试创建的 Web Service 是否正确运行，也了解了各种不同 Web Service 类型之间的差别以及优缺点，并且学会了分析解析 WSDL 文档的各部分内容。

5. 问题及解决方法

创建 Webservice 时方法无法选择



解决方法：使用低版本 java(1.8)