

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：JDevloper

编程语言：Java

API：Java API for Web Services（JAX-WS）

构建工具：Ant

1. 实验目的

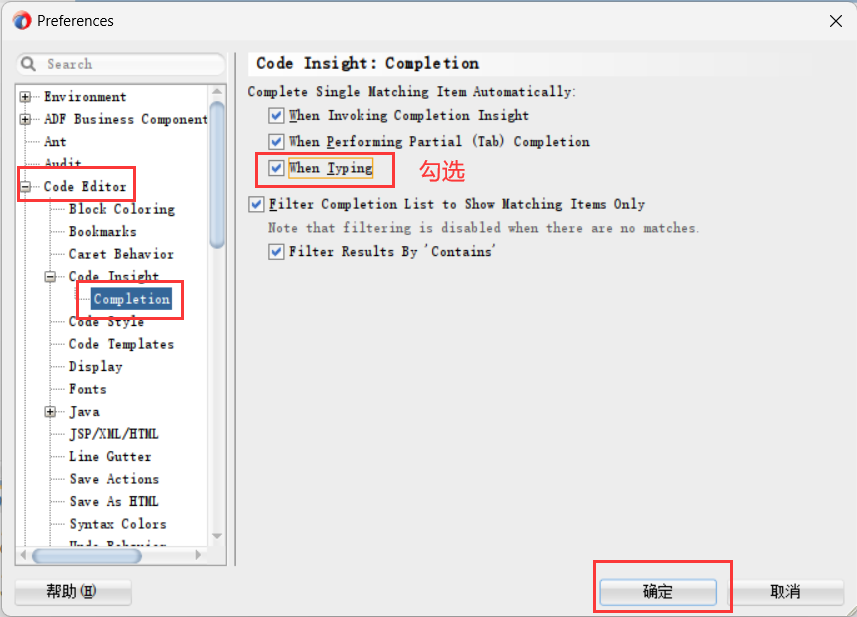
使用JAX-WS和相关工具来创建和发布SOAP web service

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。

1. 实验内容和步骤

** 创建 SOAP web service 的步骤；**

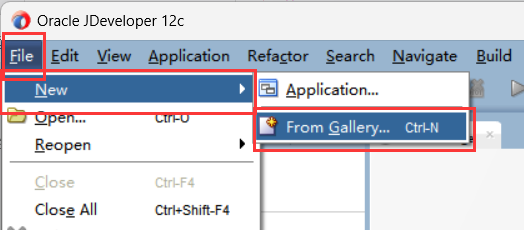
1. 概述
2. JDeveloper 的代码补全功能设置（可选）



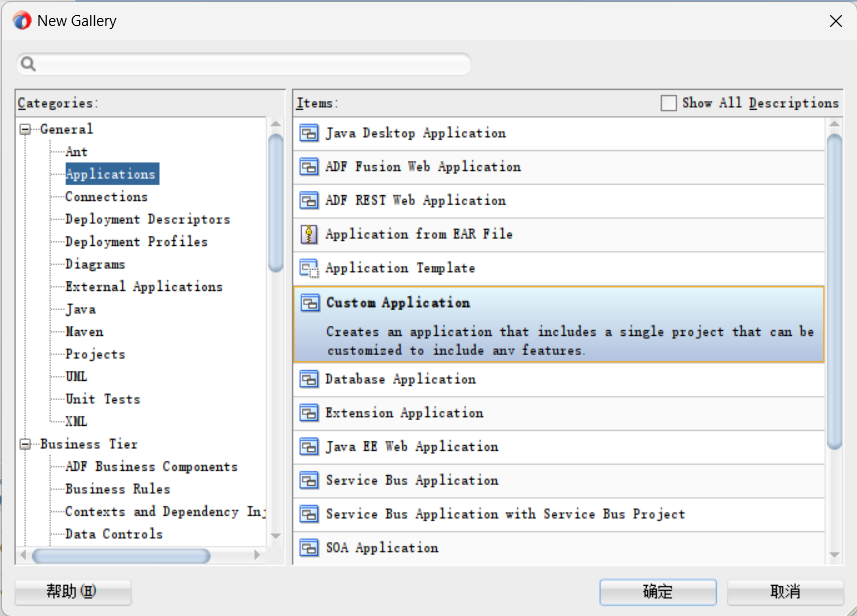
1. 使用 JDeveloper 创建 web services
   1. 创建一个 web service

创建一个 Plain Old Java Object（POJO）对象

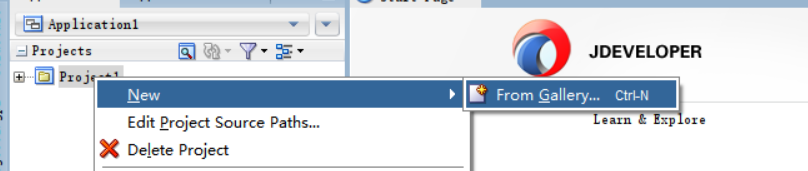
在JDevloper中创建一个空项目

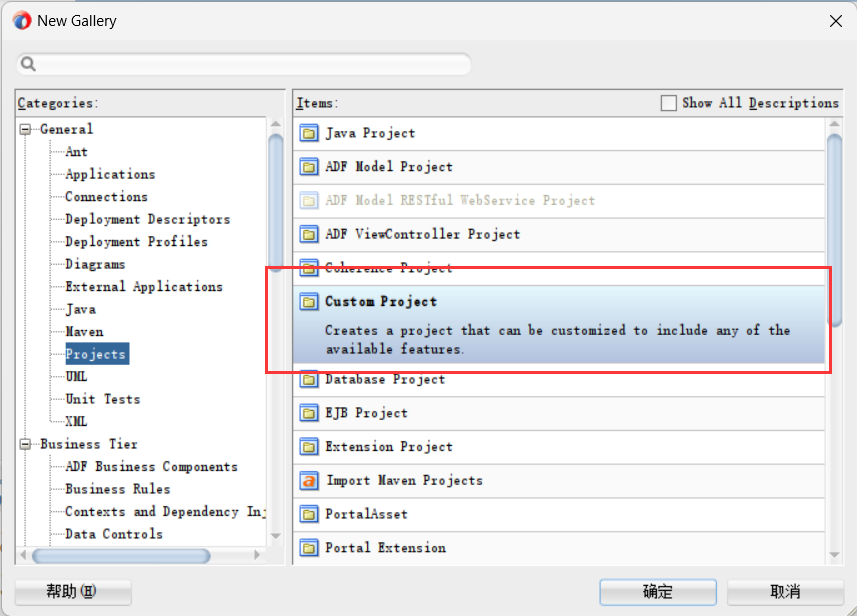


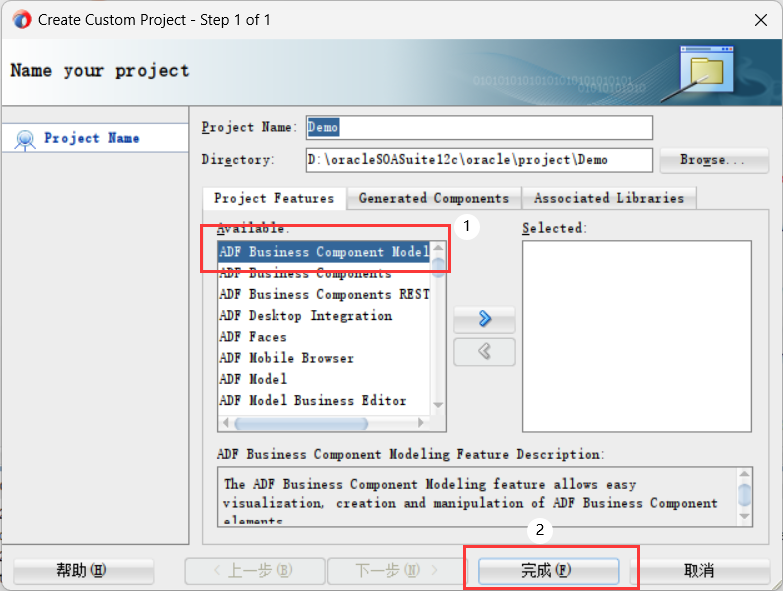
先创建一个application



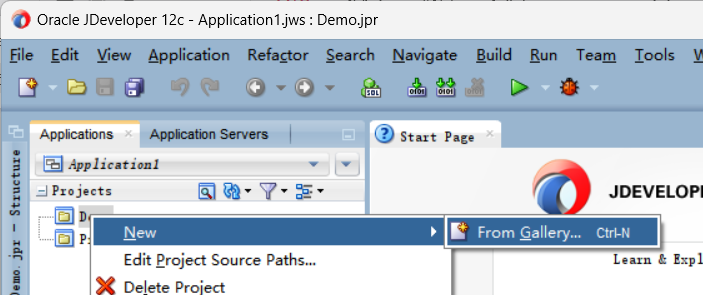
在application下创建一个project

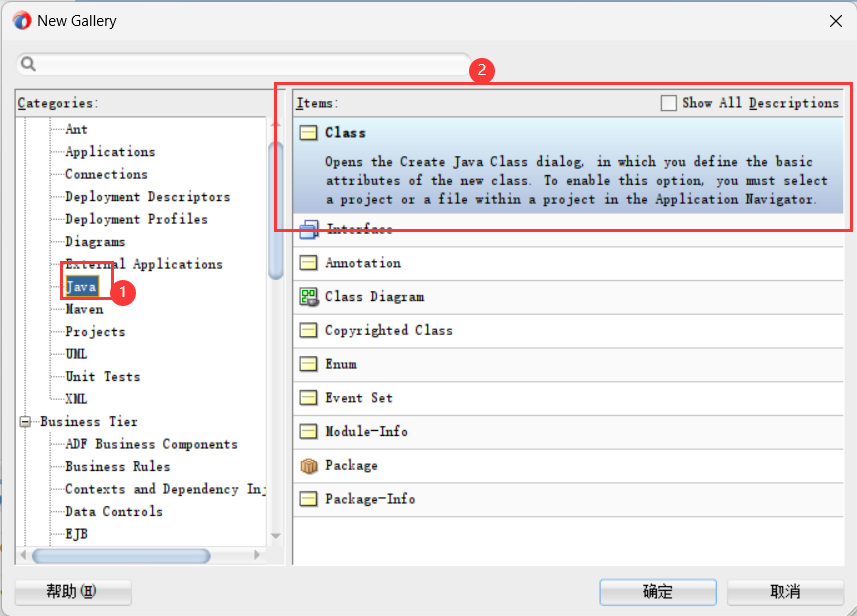


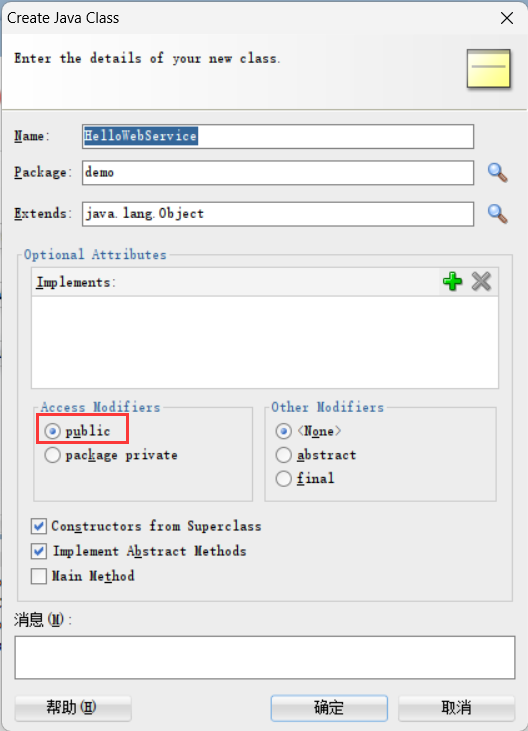




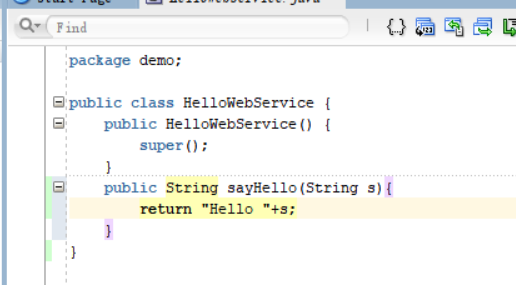
在新的project下创建一个java类



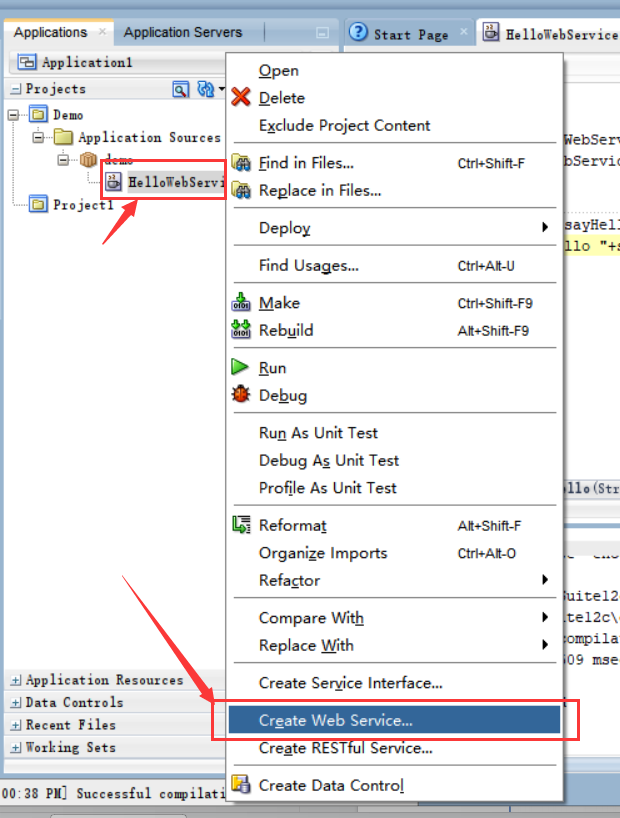




创建新的方法sayHello输入字符串s并返回”Hello “+s用于后续测试

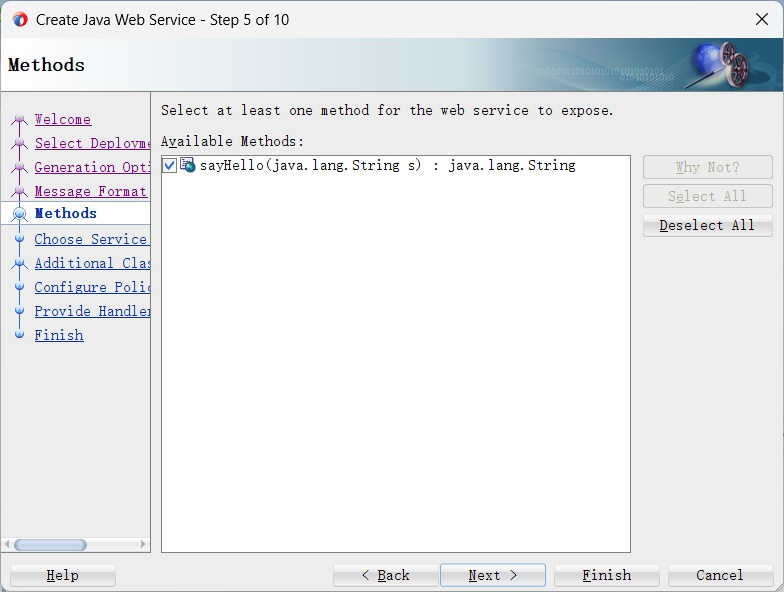


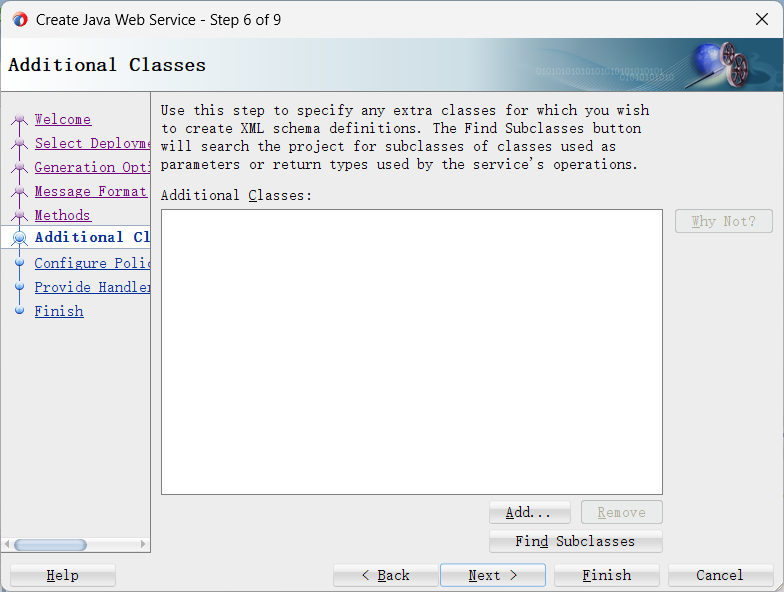
创建 web service 类



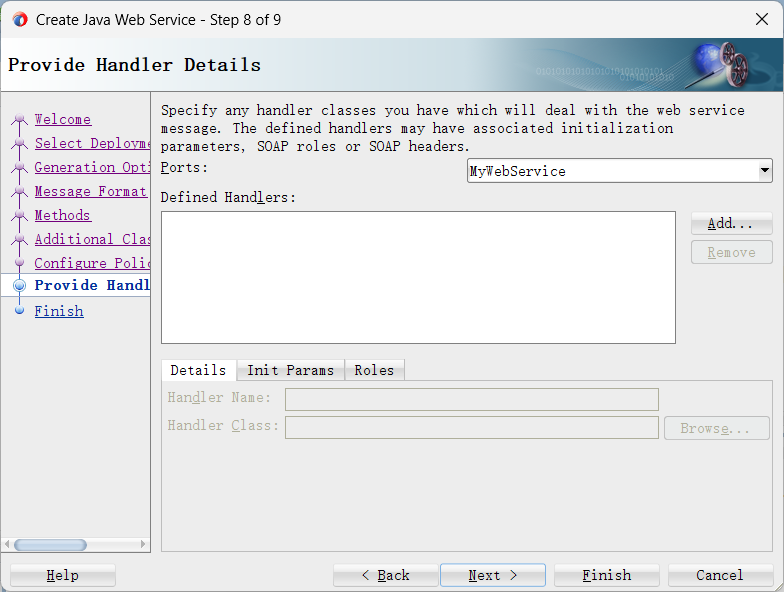


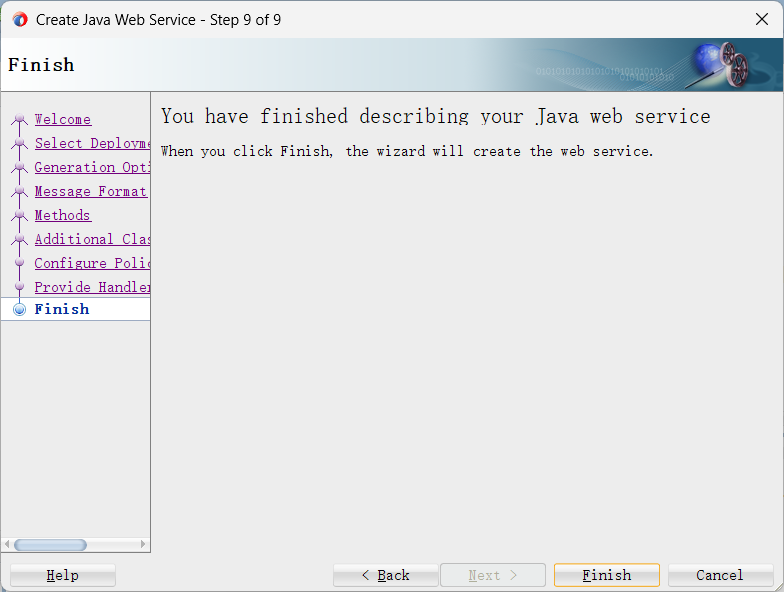




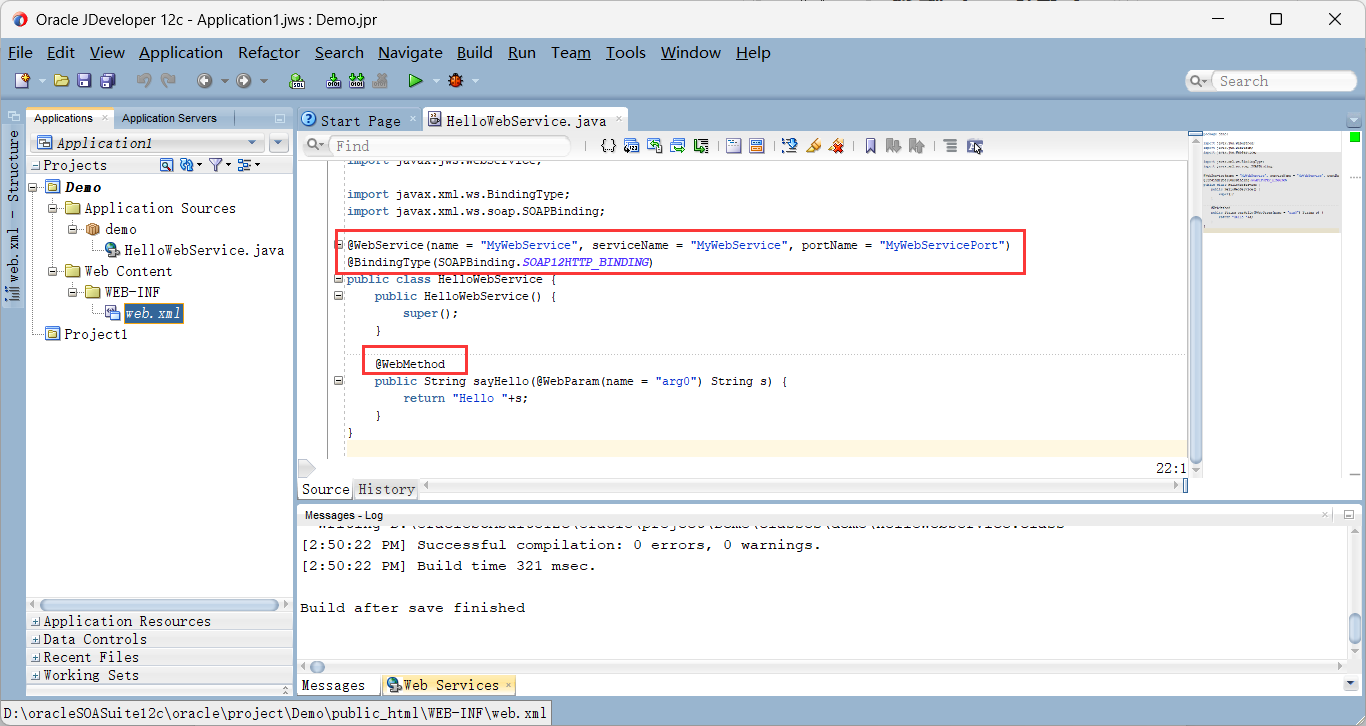








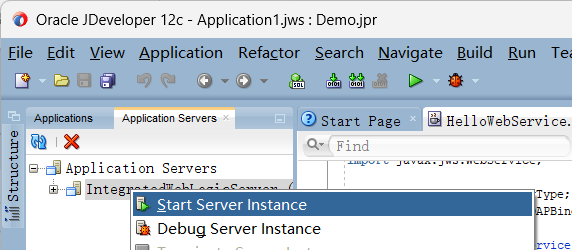
原 java 类中添加了新的注释 annotations：@WebService、@BindingType、@WebMethod 和@WebParam。



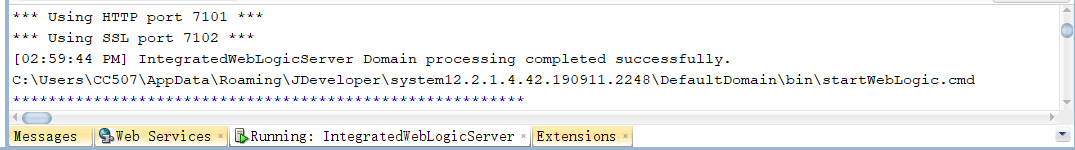
** 测试所创建的 web service 是否成功；**

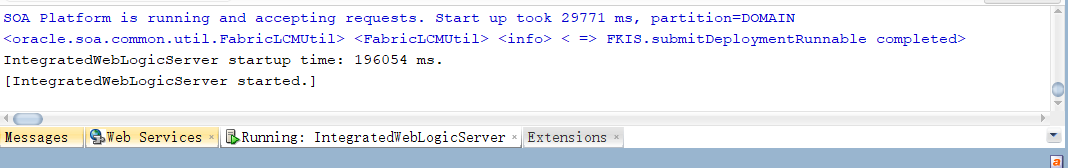
* 1. 测试 web service

步骤一：启动 IntegratedWebLogicServer

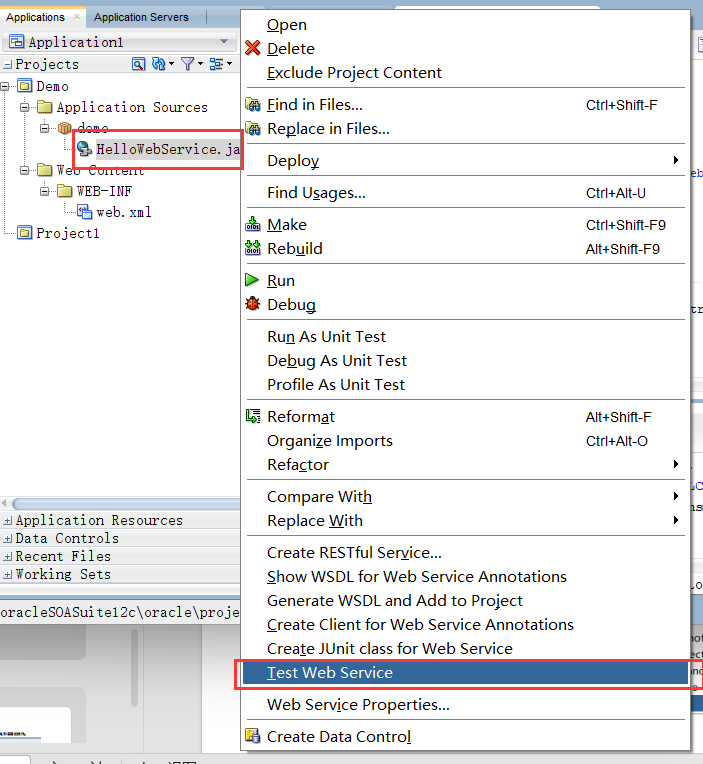


显示IntegratedWebLogicServer started表示成功启动

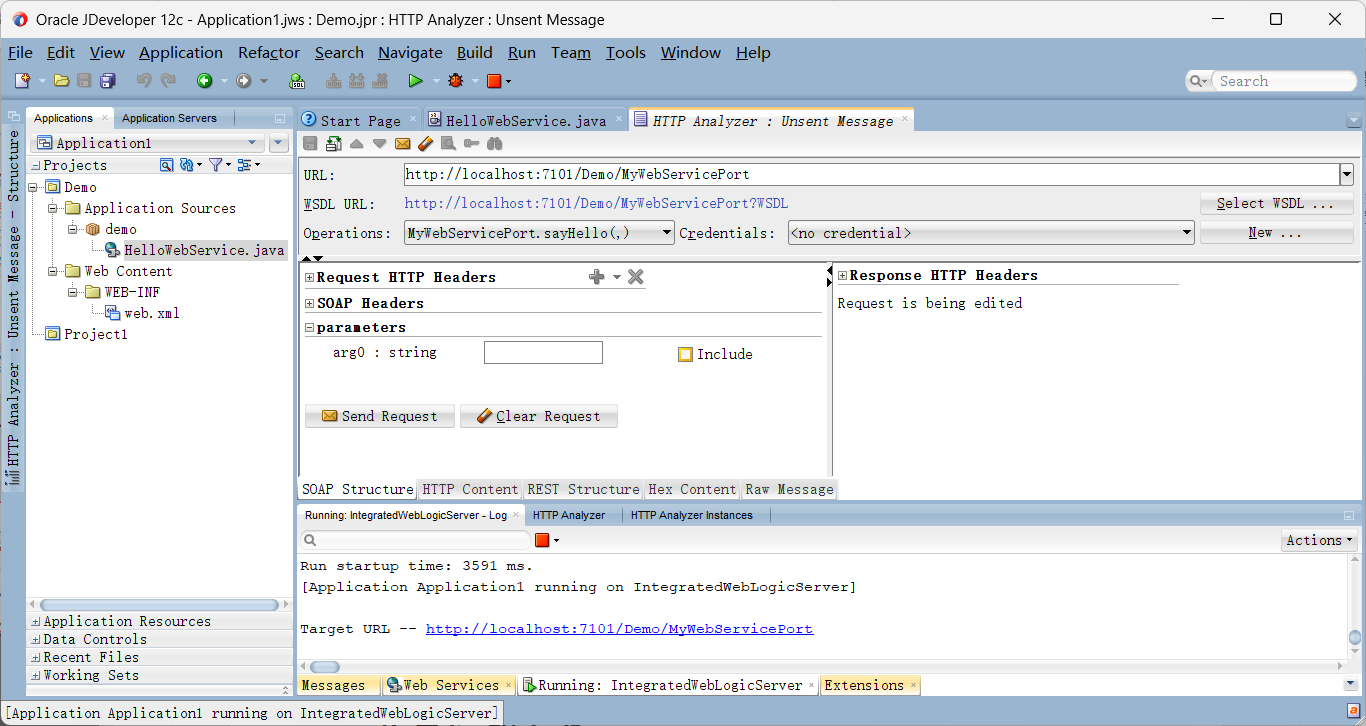




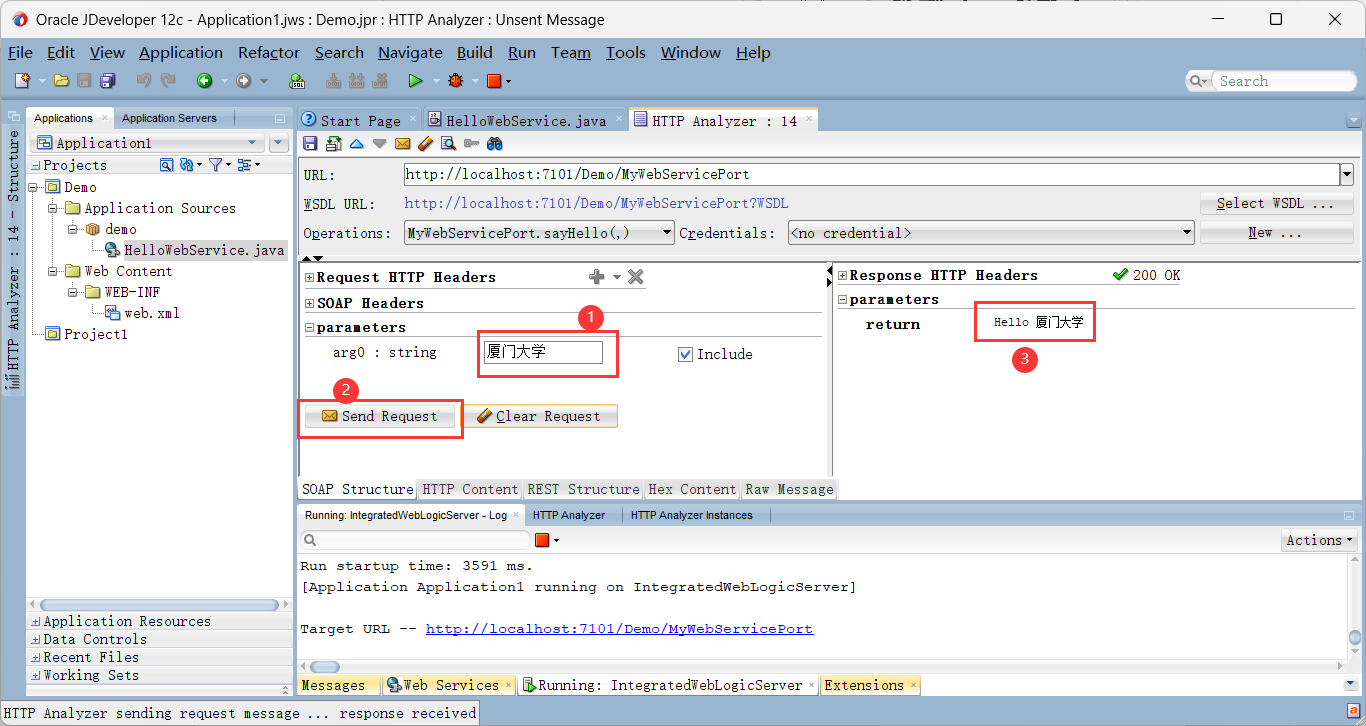
步骤二：测试 web service



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

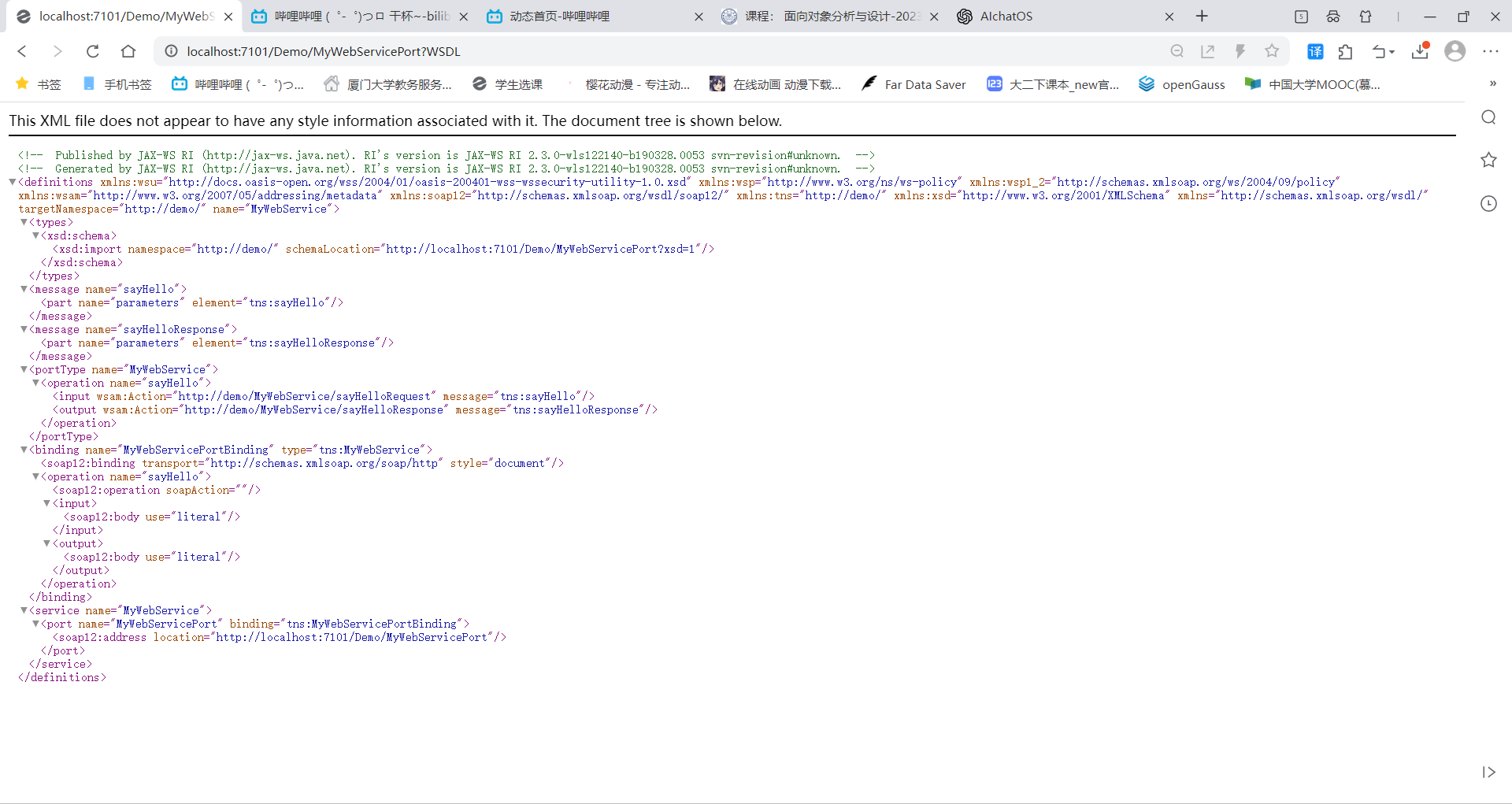
XML-RPC（XML Remote Procedure Call）是一种RPC（Remote Procedure Call）通信协议，使用HTTP作为传输协议，将请求和响应数据封装在XML格式之中。XML-RPC Web服务可以使用多种编程语言实现，传输数据体积相对较小，具有适应性强和易于实现等优点。但在安全性方面存在一些缺陷，例如采用明文传输、不支持数字证书等规范引入的安全问题。

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

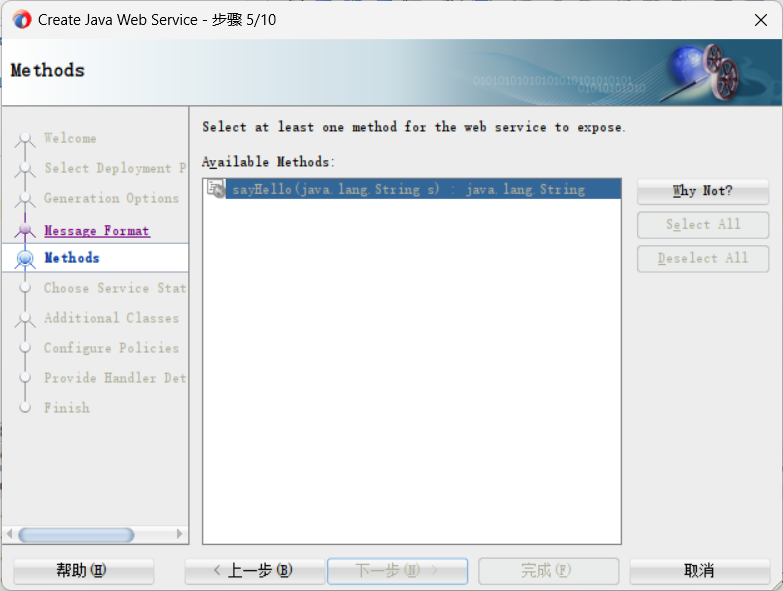
service元素定义了Web服务的名称以及包含的端点（即提供Web服务的地址）。在本例中，定义了一个名为"MyWebService"的服务，该服务包含了一个名为"MyWebServicePort"的端点，连接到"http://localhost:7101/Demo/MyWebServicePort"地址。

1. 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。

1. 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)