

## 实验三：SOA 组合应用的创建与部署

### 1. 实验环境

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

### 2. 实验目的

- 掌握创建、部署和测试 SOA 组合应用的方法

### 3. 实验要求

- 完成任务并提交实验报告。实验报告内容以下任务的方法：
  - 创建一个 SOA 应用；为 SOA 应用增加一个 BPEL 过程；编辑一个现有的 BPEL 过程；部署和测试 SOA 组合应用
- 实验提交截至日期：2023 年 10 月 27 日星期五

### 4. 实验内容

- 见以下操作

### 5. 问题及解决方法

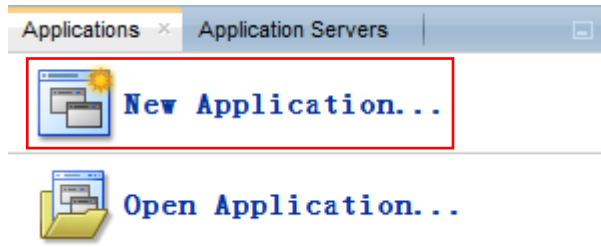
## 目录

任务一：创建一个 SOA 应用 .....	3
任务二：增加一个 BPEL 过程 .....	6
任务三：编辑一个 BPEL 过程 .....	8
任务四：组合服务的部署 .....	15
任务五：测试组合服务 .....	19

# 任务一：创建一个 SOA 应用

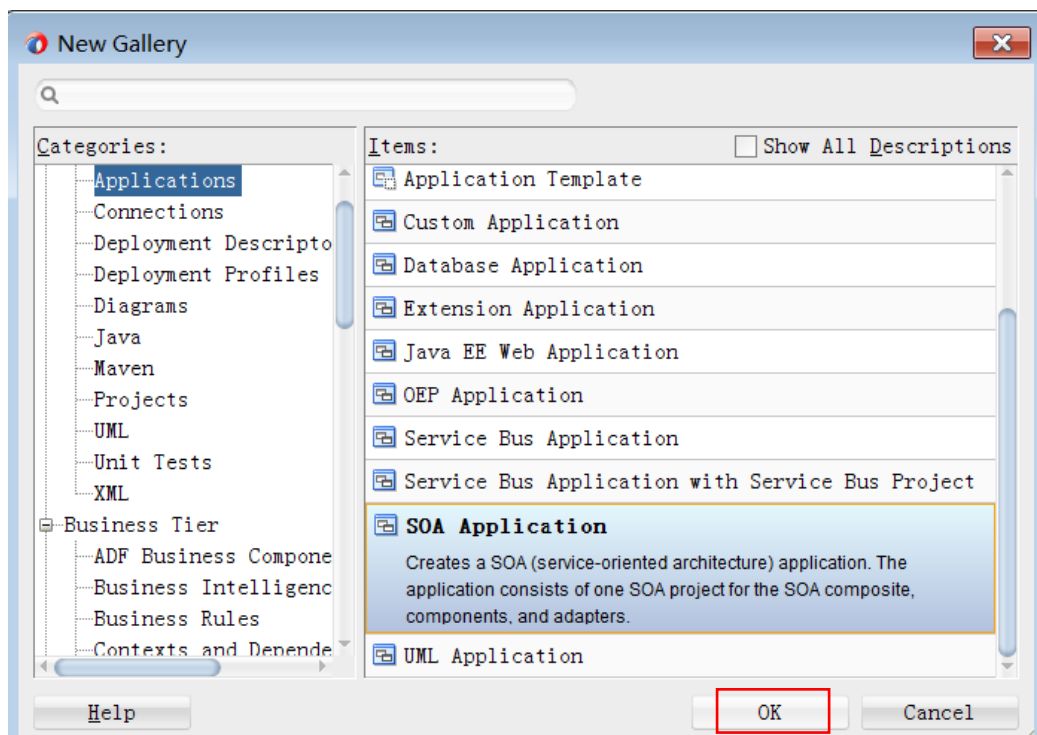
---

1. 打开 Oracle JDeveloper 12c, 点击 Applications 标签, 选择 New Application.

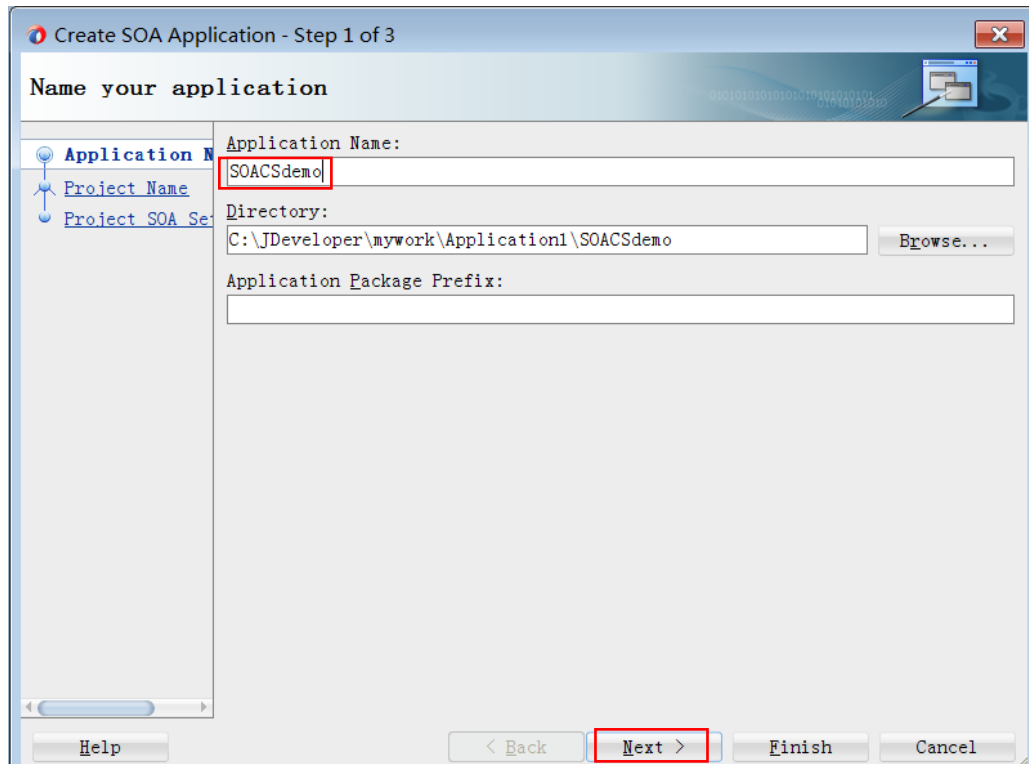


出现 New Gallery 页面。

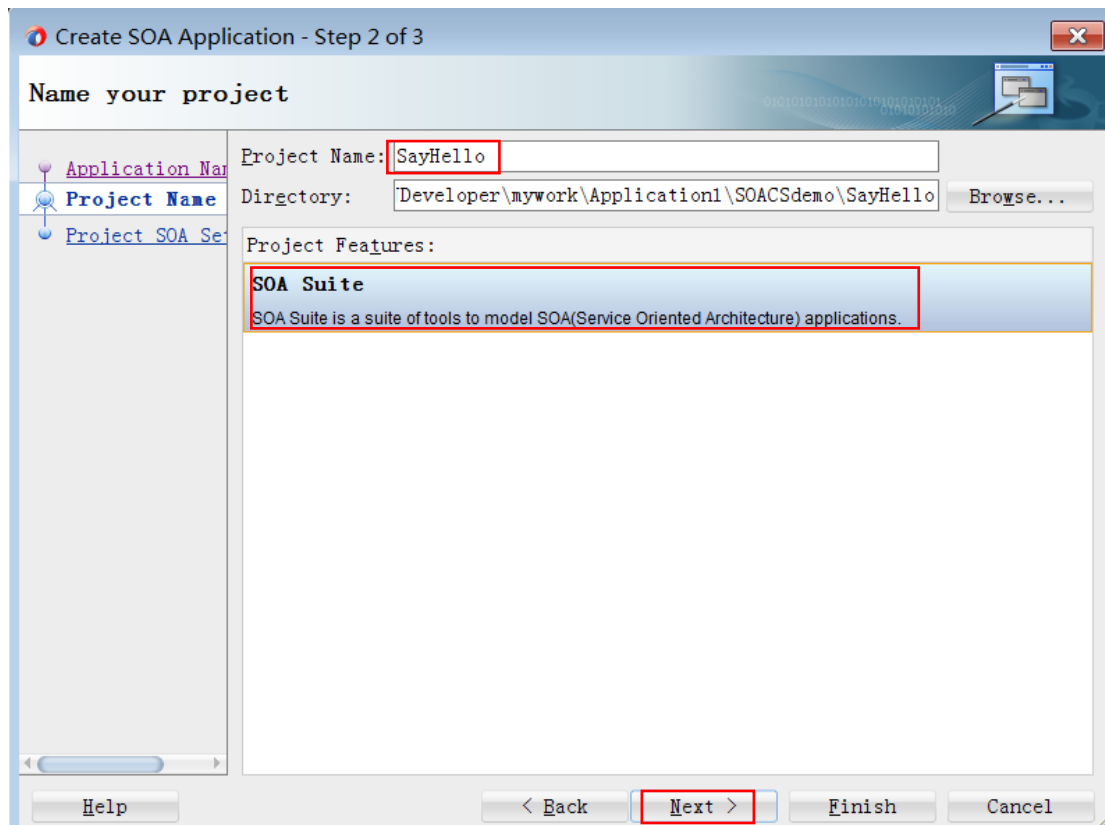
2. 选择 SOA Application->OK.



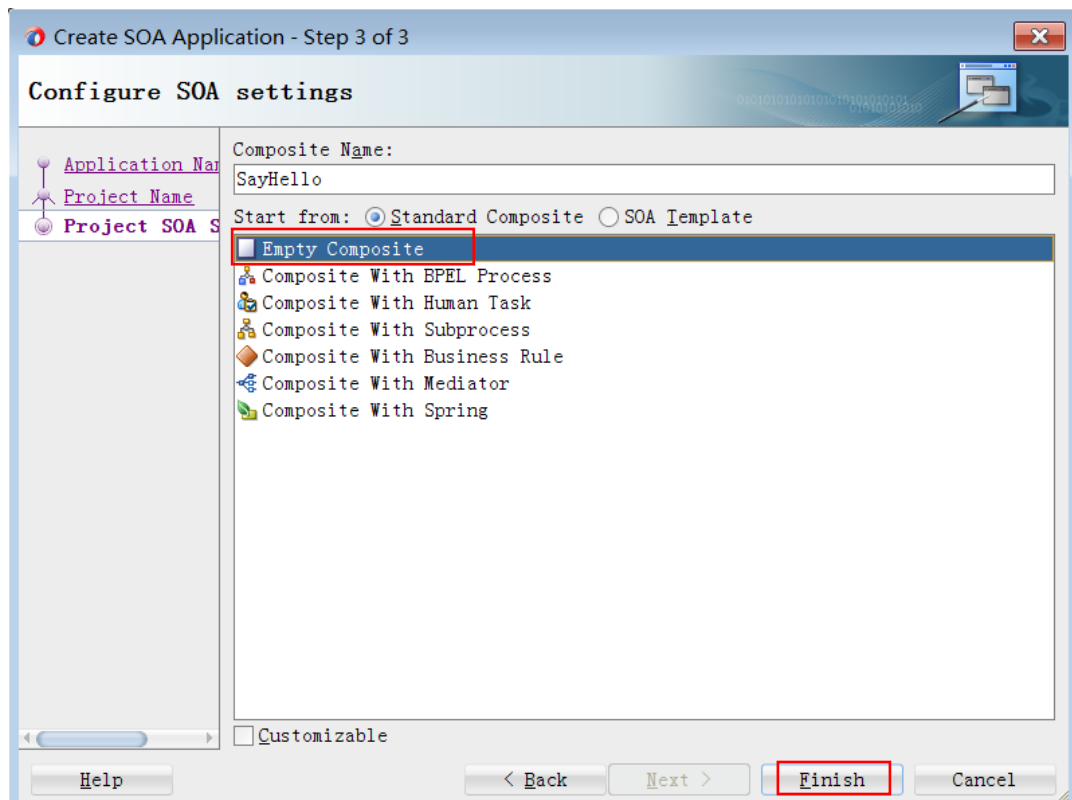
3. 在 Create SOA Application-Step 1 of 3 页面, Application Name 填写: SOACSDemo, 点击 Next.



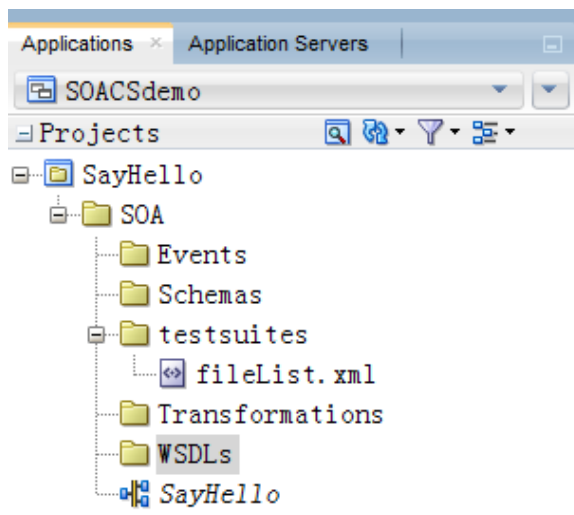
4. 在 **Project Name** 填写: **SayHello**; 在 **Project Features** 选择 **SOA Suite**, 点击 **Next**



5. 选择 **Empty Composite**，点击 **Finish**。

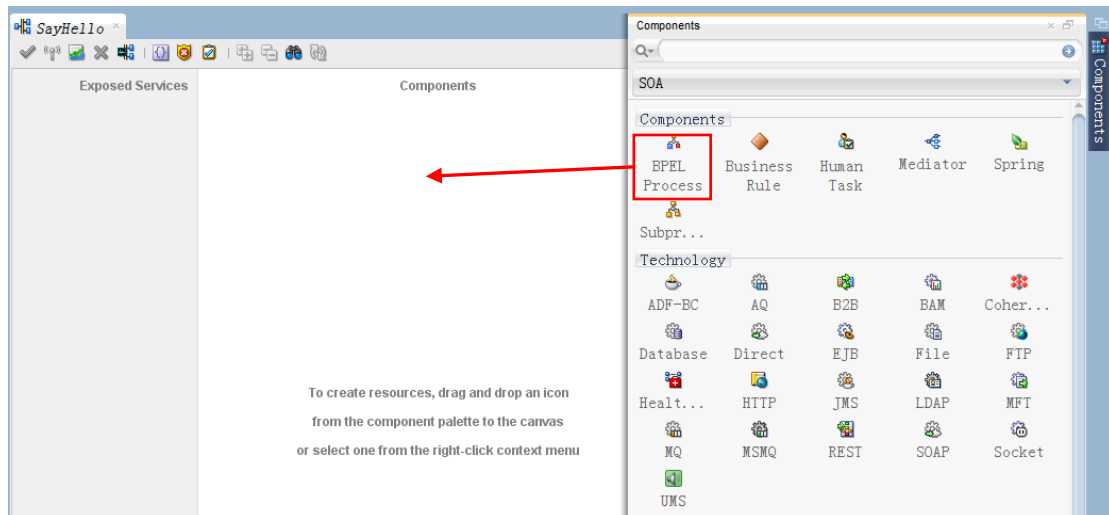


6. 在 Application 标签上，检查刚创建的 **SOACSDemo** 应用文件。



## 任务二：增加一个 BPEL 过程

1. 在 SayHello 页，从右边的 Components 面板里拖拽一个 BPEL 过程组件到 Components 泳道，出现 Create BPEL Process 页面。



2. 各域填写或选择如下内容之后，点击 OK.

**Create BPEL Process**

**BPEL Process**

A BPEL process is a service orchestration, based on the BPEL specification, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

☒ BPEL 2.0 Specification ☐ BPEL 1.1 Specification

General In Memory SOA

Name: MakeGreeting

Namespace: http://xmlns.oracle.com/SOACSDemo/SayHello/MakeGreeting

Directory: veloper\mywork\Application1\SOACSDemo\SayHello\SOA\BPEL

Template Type: ☒ Web Service ☐ REST Service ☐ No Service

Template: Synchronous BPEL Process

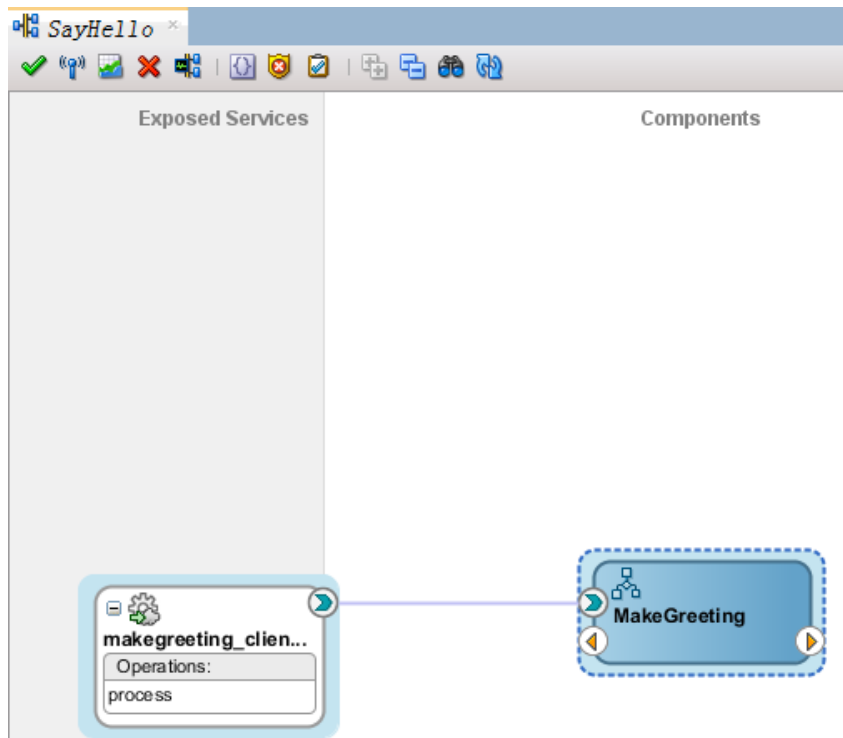
Service Name: Asynchronous BPEL Process

Input: le.com/SOACSDemo/SayHello/MakeGreeting}process

Output: SOACSDemo/SayHello/MakeGreeting}processResponse

Help OK Cancel

3. 在 SayHello 页，检查 **MakeGreeting** 和 **makegreeting\_client\_ep** 组件。

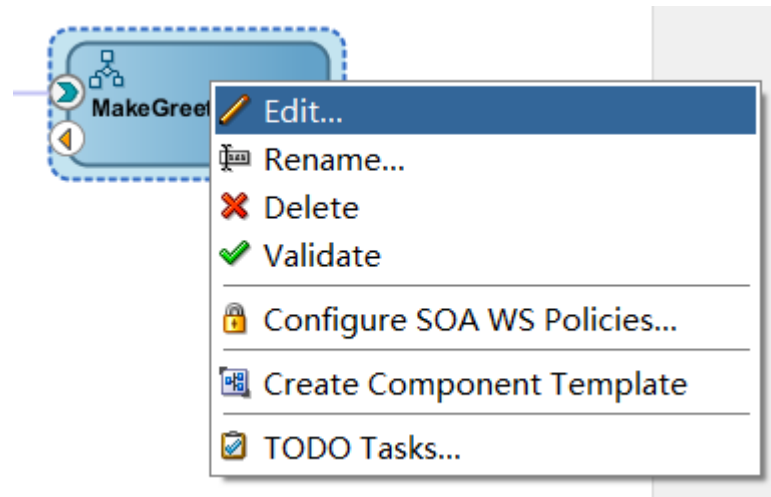


## 任务三：编辑一个 BPEL 过程

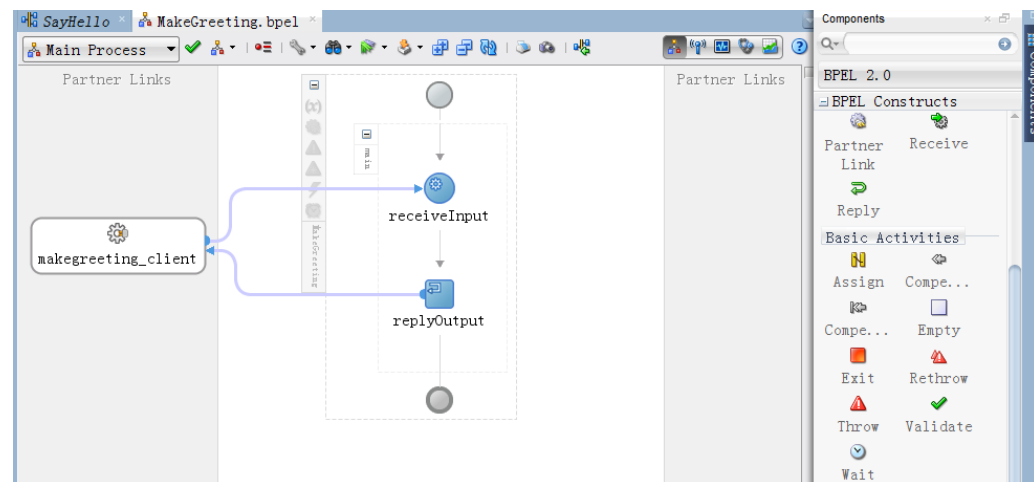
### 1. 增加一个 Assign 活动

添加一个 BPEL 过程之后，我们可以给它添加 Assign 活动。

- 1) 在 SayHello 页面，右击 MakeGreeting BPEL 组件，选择 Edit。

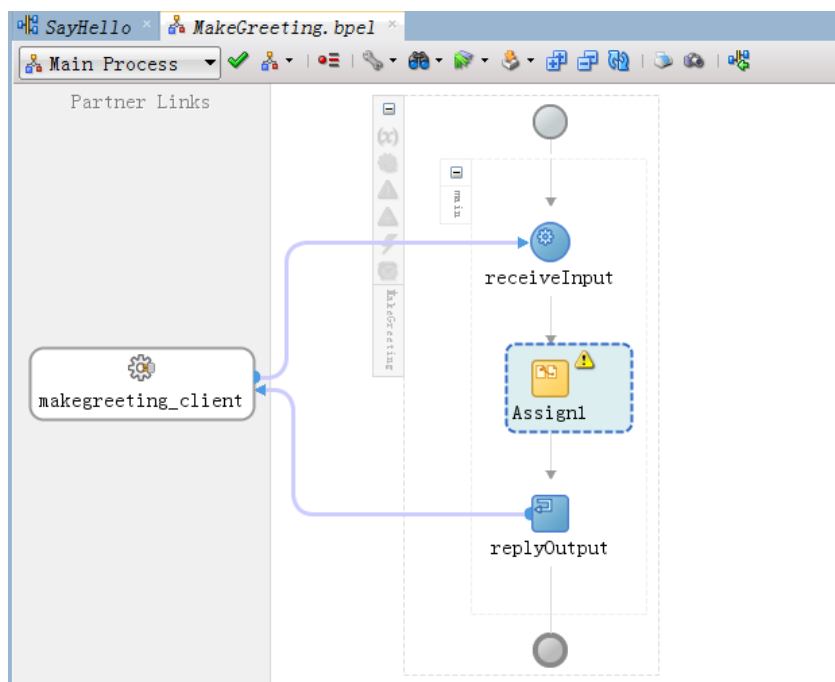
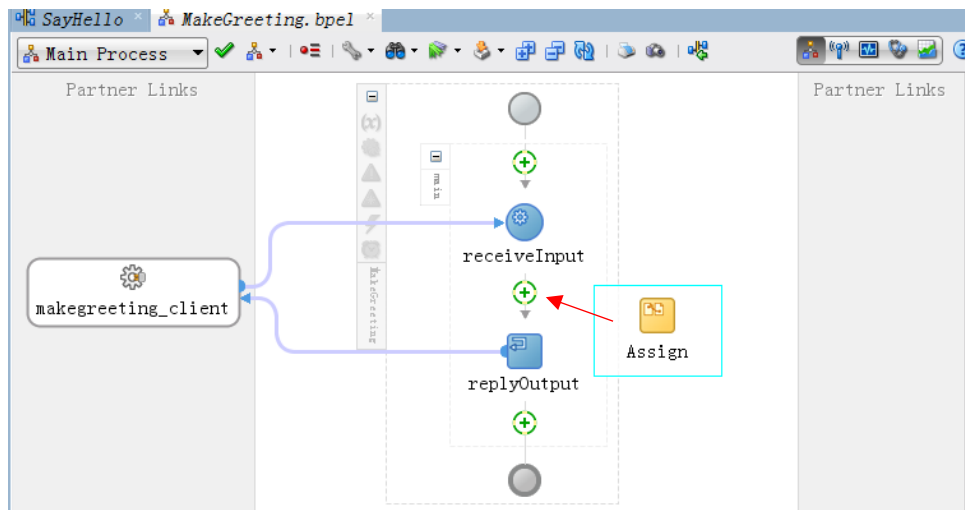


出现 MakeGreeting.bpel 页面。



- 2) 从 Basic Activities 面板拖拽 Assign 活动到主过程序列的 **receiveInput** 和 **replyOutput** 中间。



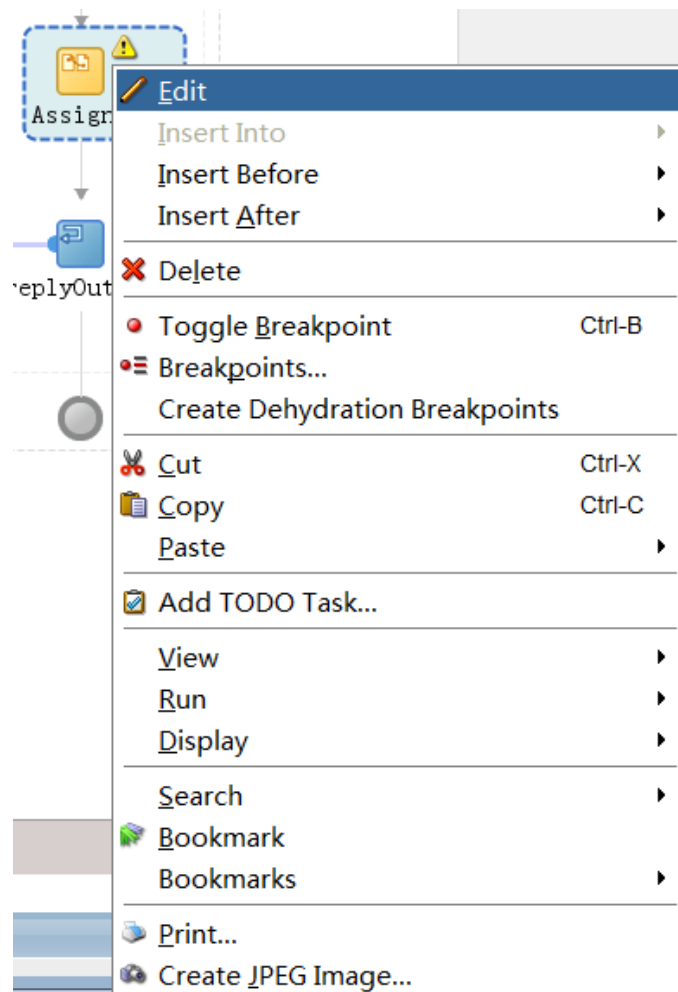


默认 Assign1 会显示一个警告图标!。

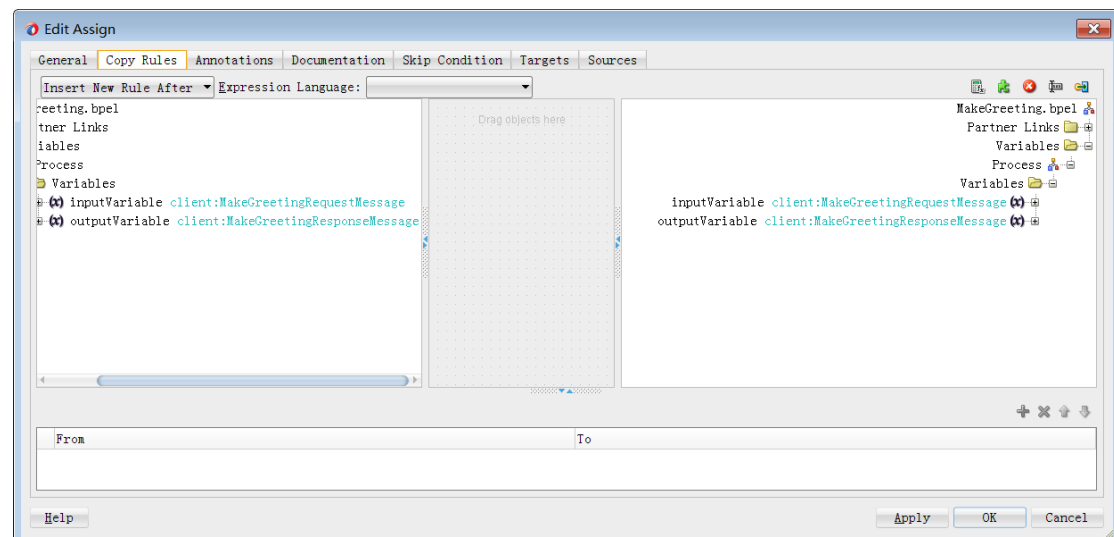
## 2.编辑一个 Assign 活动

添加一个 Assign 活动后，我们可以编辑它来建立输出消息，步骤如下：

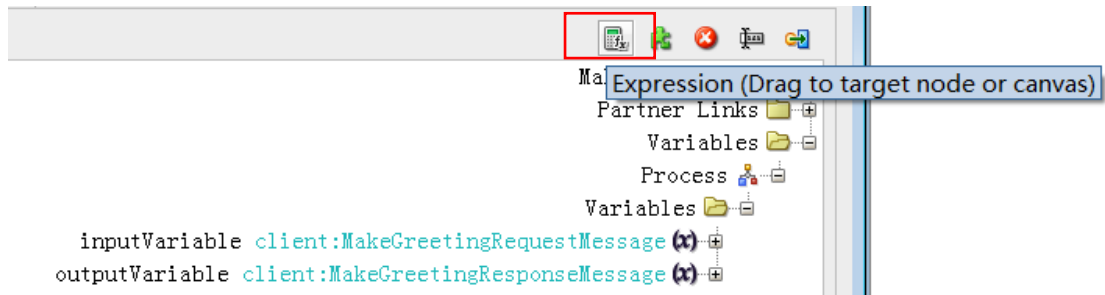
- 1) 右击 Assign1 活动，选择 Edit。



出现 Edit Assign 页面。

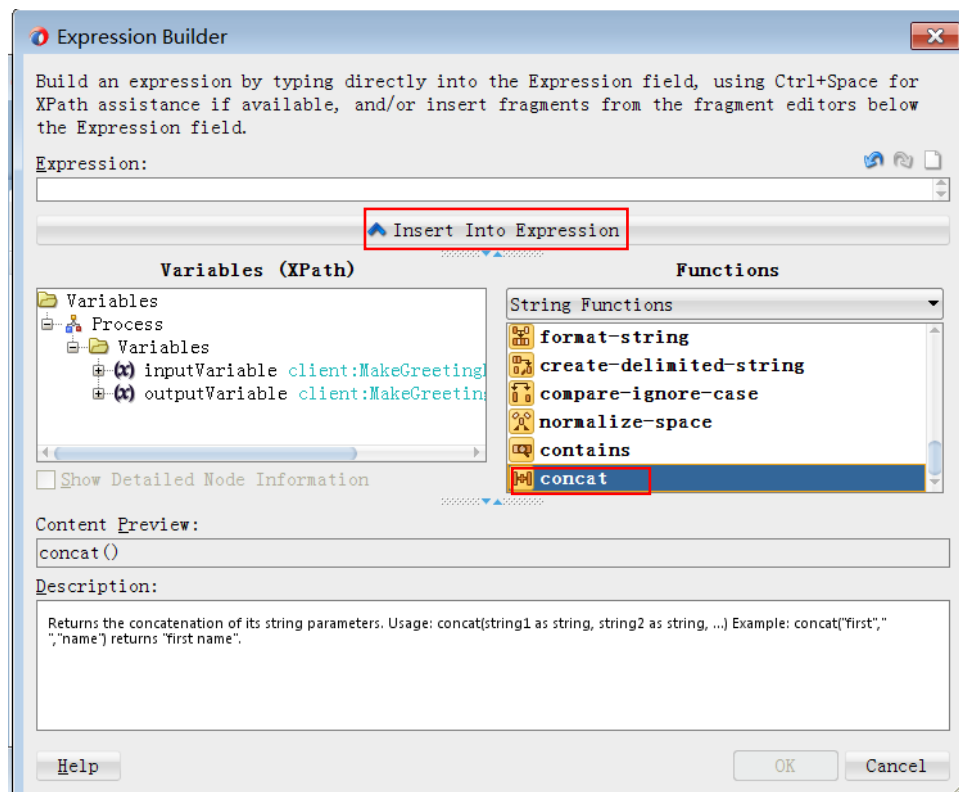


2) 点击 Expression 图标。

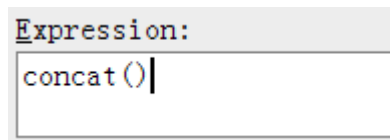


出现 Expression Builder 页面。

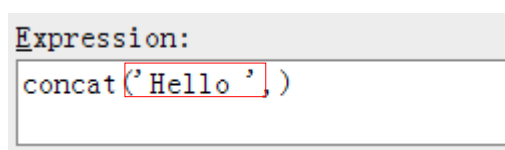
3) 在 String Functions 面板中选择 concat 函数->点击 Insert Info Expression。



在 Expression 面板出现 concat()函数。



4) 在 Expression 面板里修改 concat()函数为 concat("Hello ",)



**//重要说明!** 上面的 concat("Hello ",)里面应是双引号, 不是单引号! 请改正!

5) 在 Expression 面板, 让光标在逗号之后处于活动状态, 扩展 variables(XPath)面板中的

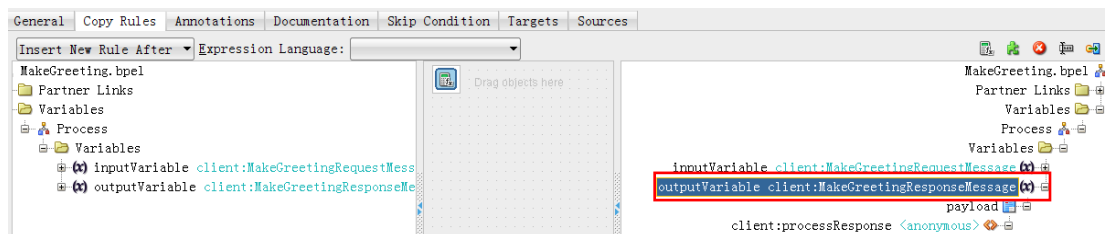
inputVariable->payload->client:process, 选择 client:input string, 点击 Insert Into Expression. 在 Expression 面板上可验证表达式



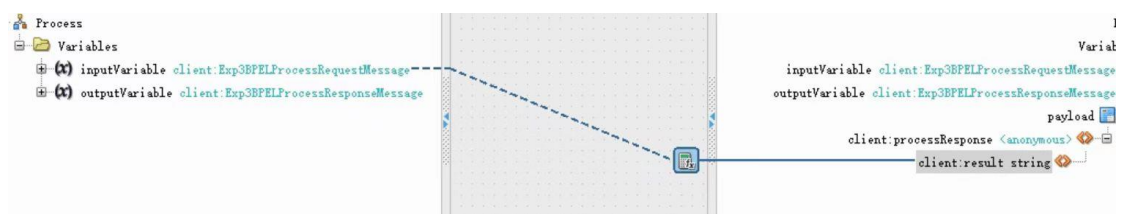
6) 在 Expression 面板上可验证表达式, 点击 OK, 出现 Edit Assign 页面。

7) 在 Edit Assign 页面右边的面板中, 扩展

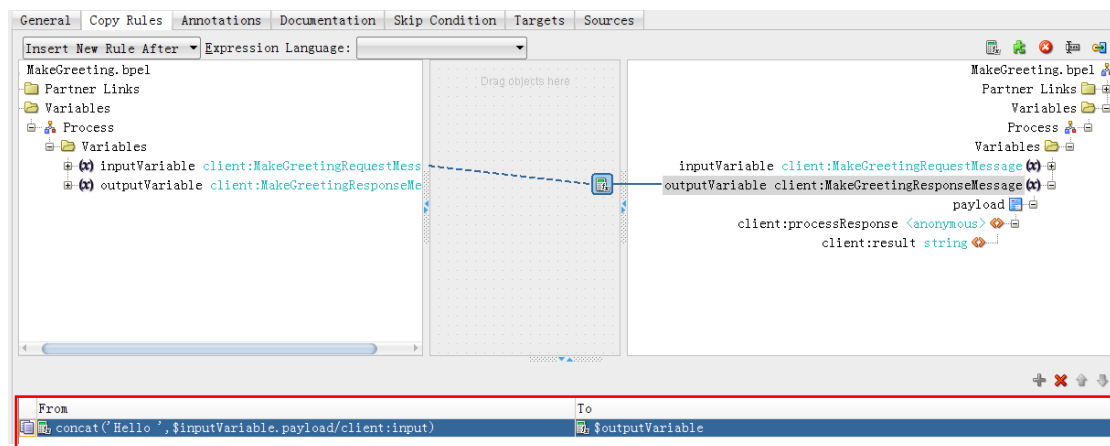
outputVariable client:MakeGreetingResponseMessage(x),  
payload, client:processResponse



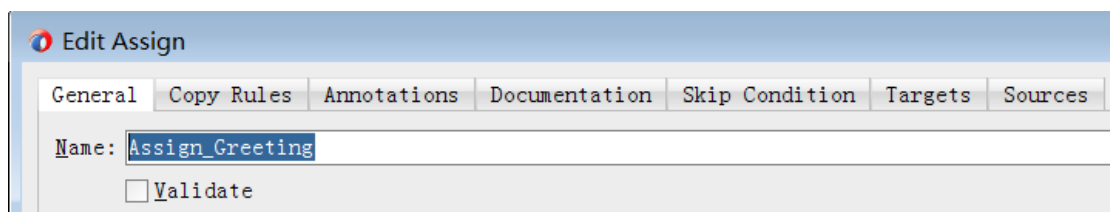
拖拽中间的功能到 client:result string



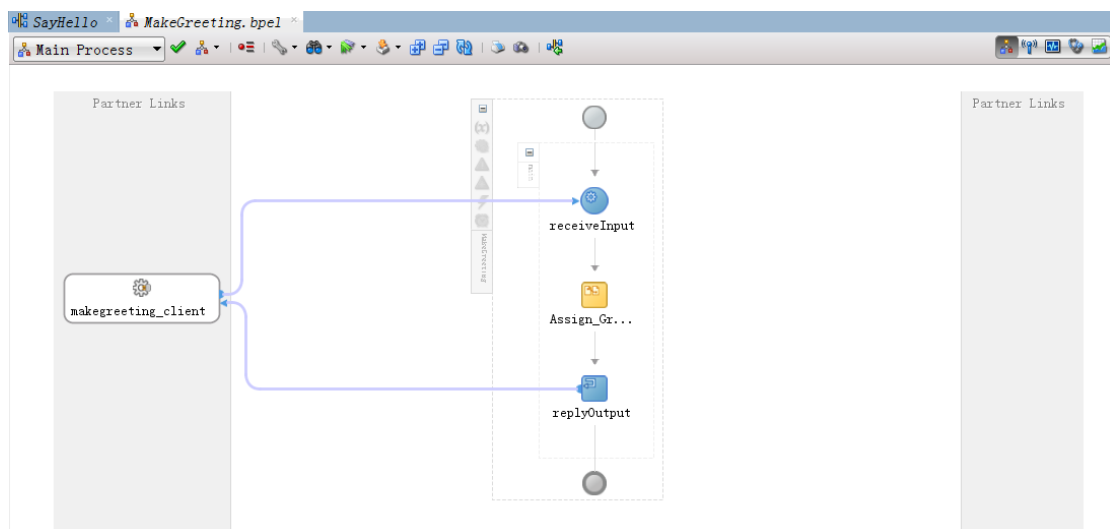
功能连接 client:input string from the inputVariable and the client:result string from the outputVariable.



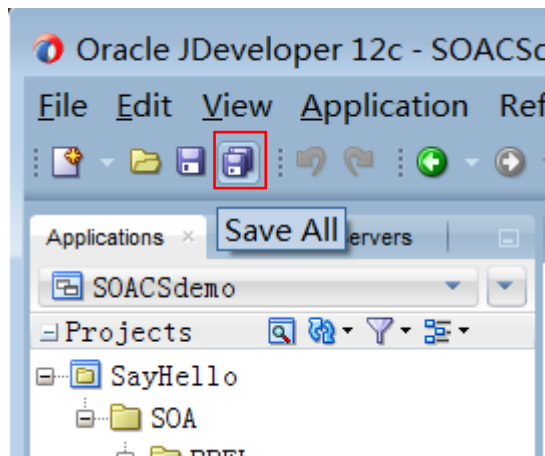
8) 在 Edit Assign 页面，选择 General 标签，Name 填写为 Assign\_Greeting。



点击 OK 后出现 MakeGreeting.bpel 页面, 验证 MakeGreeting.bpel 页面包含 Assign activity。

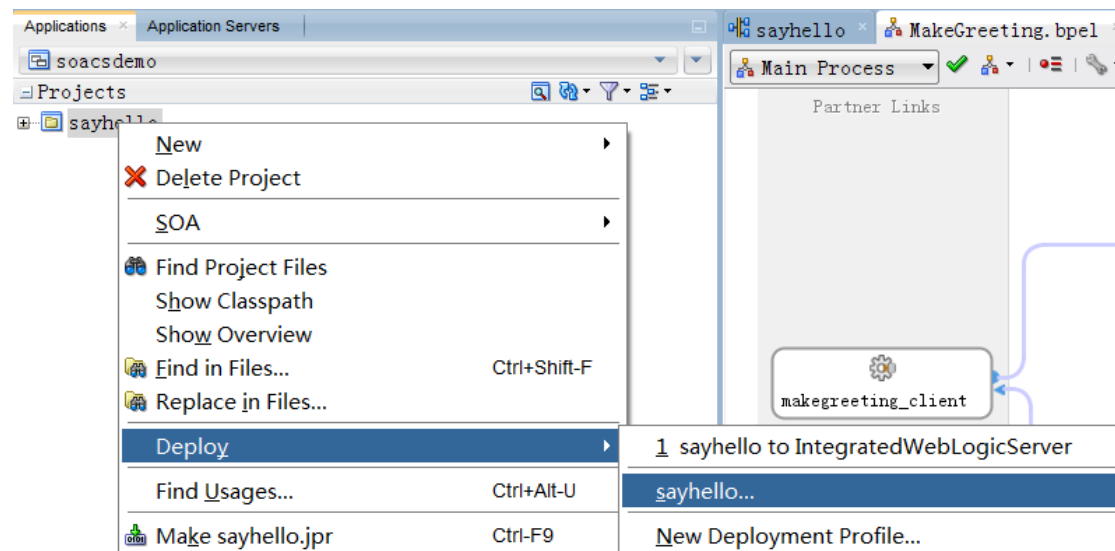


9) 在 JDeveloper 主菜单上点击 Save All

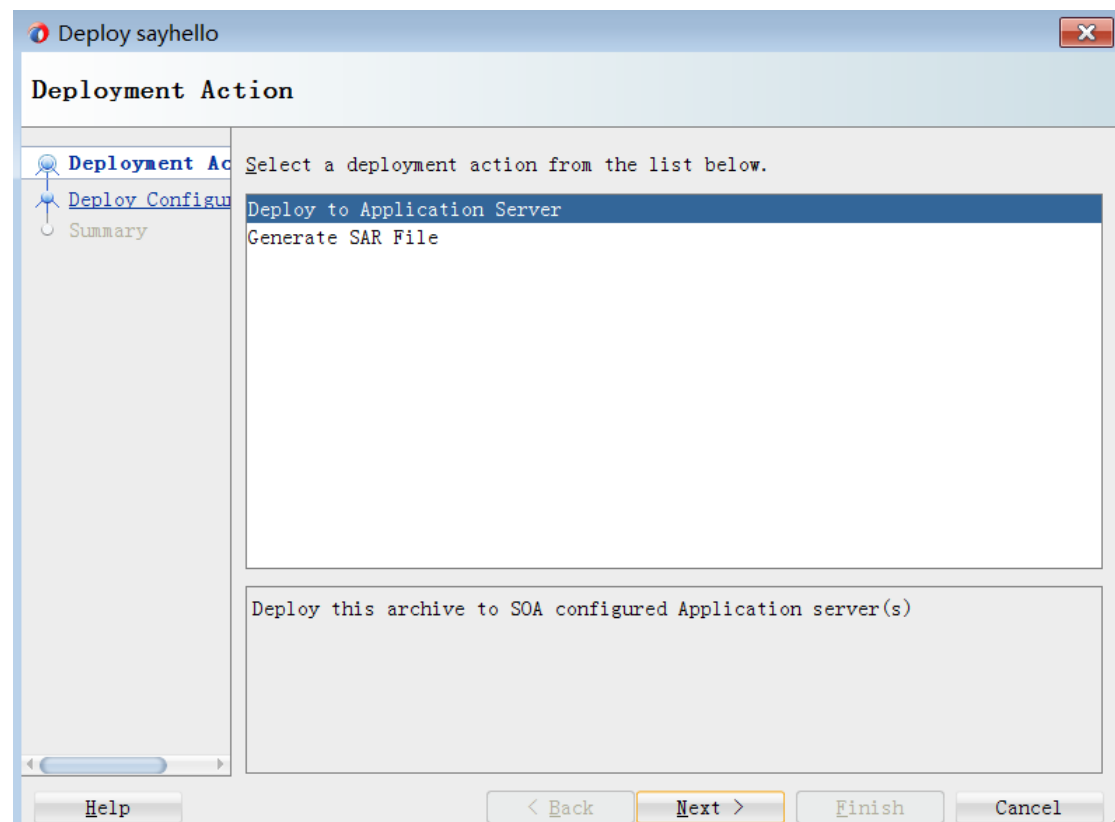


## 任务四：组合服务的部署

1. 依次点击：Application > Projects > sayhello > deploy > sayhello...



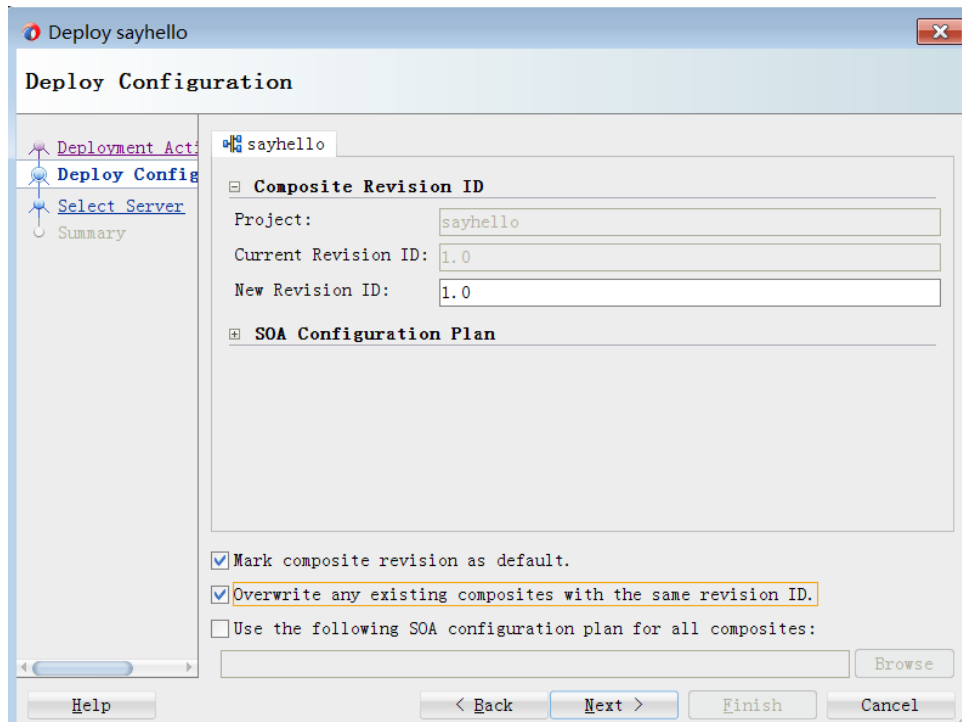
2. 出现 deployment action 页面。这步是指定组合服务部署到哪个应用服务器上。



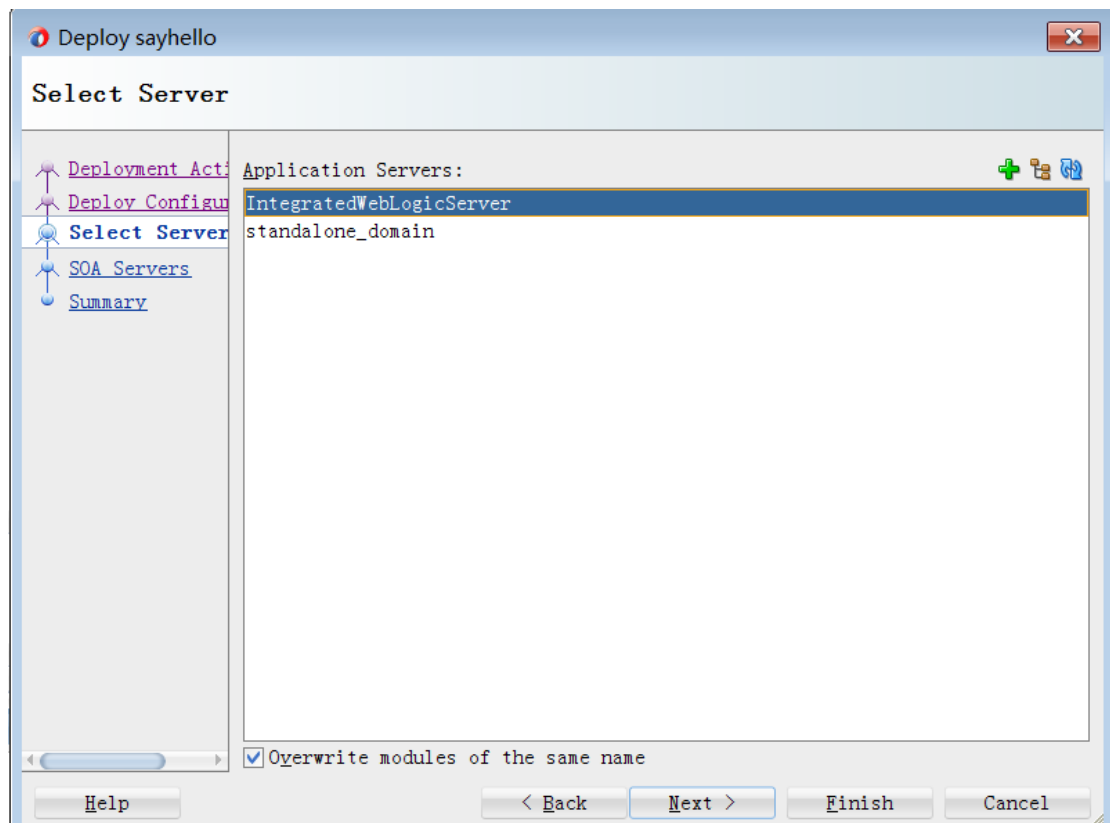
点击 next

3. 出现 deploy configuration 页面，选择相应的值。勾选“overwrite any existing composites”

with the same revision ID"

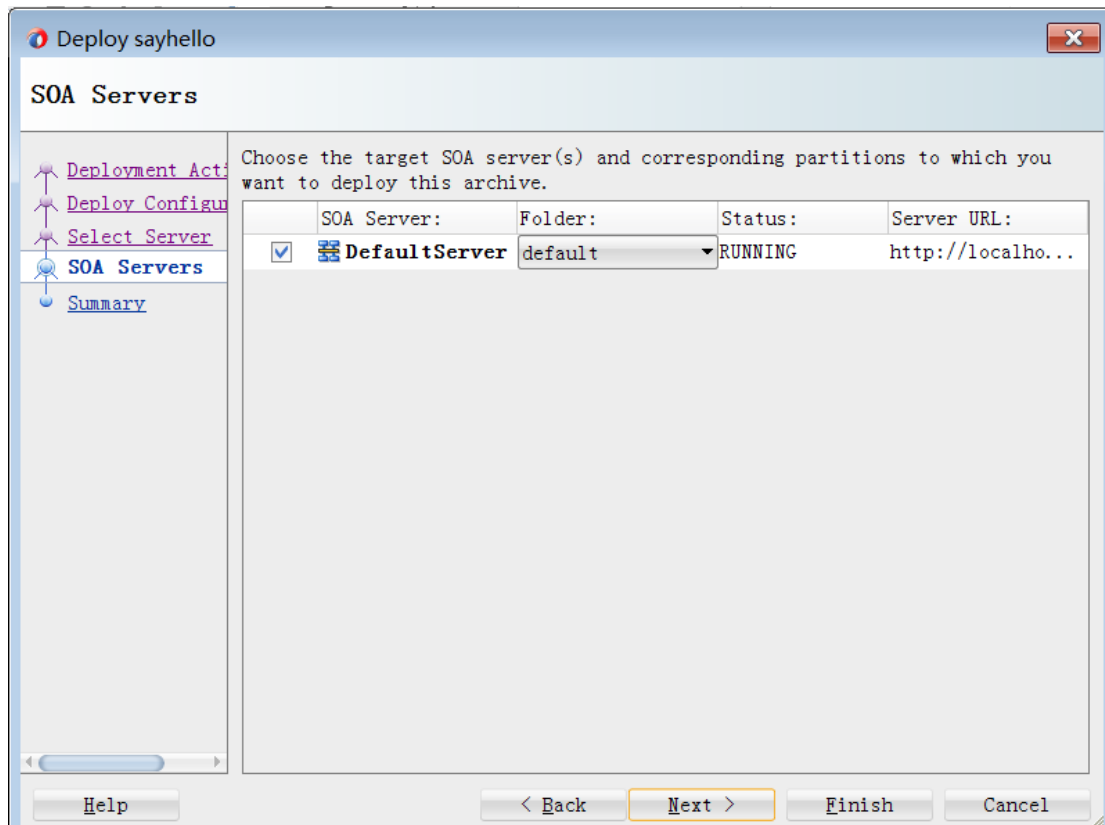


4. 进入 select server 页面。此处可以发现已安装的所有应用服务器列表。选择 Integratedweblogicserver。如果启动的是 standalone\_domain, 也可选择



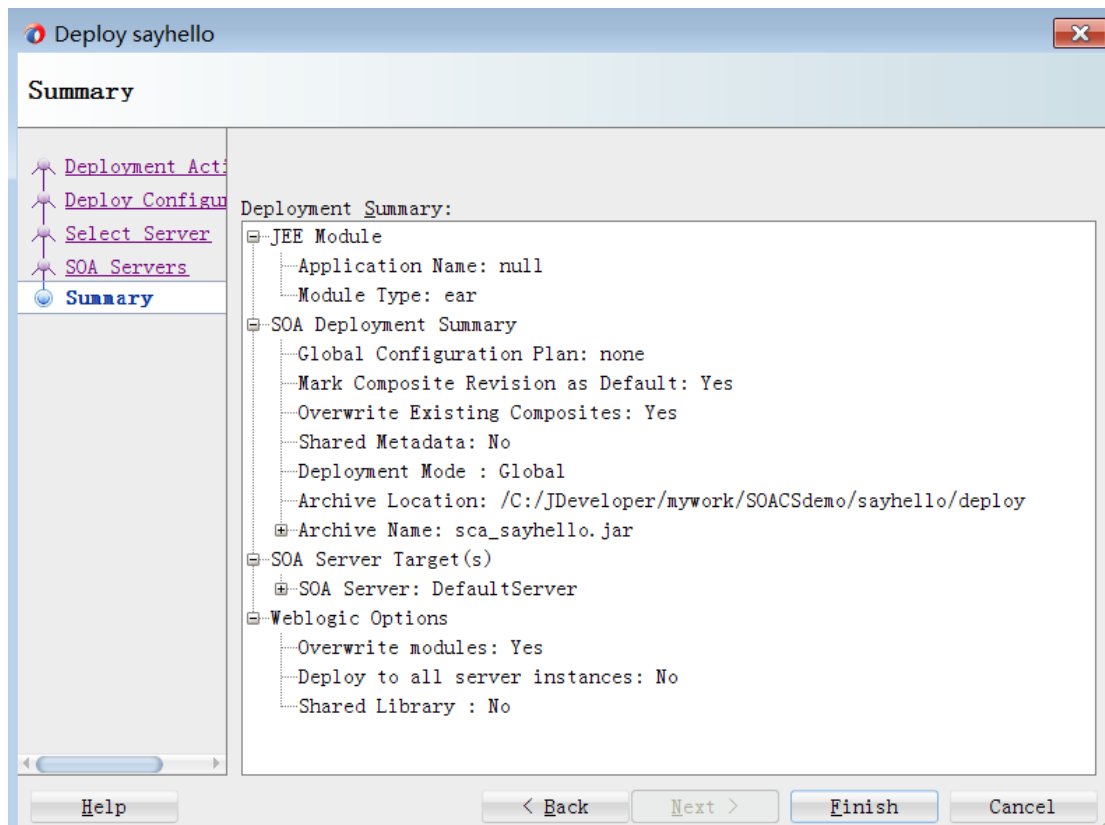
5. 进入 SOA server 页面。





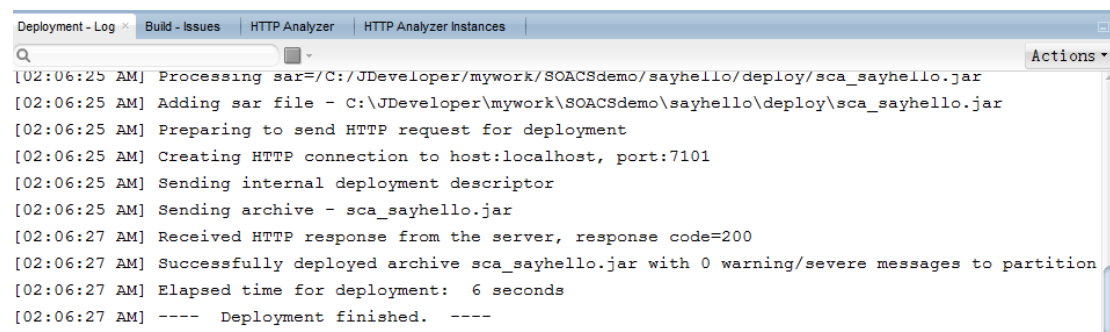
点击 next

6. 进入 summary 页面。



点击 finish 进行部署。注意：可能在部署过程中会出现一些问题，需要逐个解决。

部署需要一些时间，成功后会在部署日志出现一下页面。



The screenshot shows a web browser window with tabs for 'Deployment - Log', 'Build - Issues', 'HTTP Analyzer', and 'HTTP Analyzer Instances'. The 'Deployment - Log' tab is active, displaying a log of deployment steps. The log entries are as follows:

```
[02:06:25 AM] Processing sar=C:/JDeveloper/mywork/SOACSDemo/sayhello/deploy/sca_sayhello.jar
[02:06:25 AM] Adding sar file - C:\JDeveloper\mywork\SOACSDemo\sayhello\deploy\sca_sayhello.jar
[02:06:25 AM] Preparing to send HTTP request for deployment
[02:06:25 AM] Creating HTTP connection to host:localhost, port:7101
[02:06:25 AM] Sending internal deployment descriptor
[02:06:25 AM] Sending archive - sca_sayhello.jar
[02:06:27 AM] Received HTTP response from the server, response code=200
[02:06:27 AM] Successfully deployed archive sca_sayhello.jar with 0 warning/severe messages to partition
[02:06:27 AM] Elapsed time for deployment: 6 seconds
[02:06:27 AM] ---- Deployment finished. ----
```

## 任务五：测试组合服务

两种测试方式：1.在 jdev 里面测试；2.通过企业管理器 EM 测试。注：因第一种方式测试出现一些问题，后改用第二种方式测试成功。以下介绍第二种方式。

1. 在浏览器地址栏打开 em：

<http://localhost:7101/em>

注意：这个地址就是部署时的服务器地址，本例中在任务四之 5：SOA server。

2. 登录进入目标导航



3. 找到【已部署的组合】标签



SOA 文件夹是用于帮助您管理较大部署的组的逻辑分组。下列 SOA 组合修订已部署到此文件夹中。

组合	状态	模式	已部署 ?
● SayHello [1.0]	↑	活动	2018-12-13 20:07:30
● sayhello [1.0]	↑	活动	2018-12-14 2:06:26

- 打开第二个 sayhello[1.0]（因为第一个 SayHello[1.0]在部署时出现了一些问题，第二个是部署成功的，通过组合名可以区分）



- 点击【测试】标签

soa-infra

SOA 基础结构

SOA 基础结构 > default (SOA 文件夹) > sayhello (组合) > 测试 Web 服务

### 测试 Web 服务

使用此页可以测试任何 WSDL 或 WADL, 包括不在场中的 WSDL 或 WADL。要测试 Web 服务, 请输入 WSDL 或 WADL 并单击“对 WSDL 或 WADL 进行语法分析”。当该页用 WSDL 或 WADL 类型。指定所有输入参数, 然后单击“测试 Web 服务”。

WSDL 或 WADL

对 WSDL 或 WADL 进行语法分析

用于 WSDL 或 WADL 访问的 HTTP 基本验证选项

服务

makegreeting\_client\_ep

端口

MakeGreeting\_pt

操作

process

端点 URL

编辑 端点 URL

请求

响应

安全性

服务质量

HTTP 标头

其他测试选项

输入参数

树视图

启用验证

加载有效负载

选择文件

未选择任何文件

保存有效负载

#### SOAP 正文

查看

分离

名称	类型	值
* payload	payload	
* input	string	<div>Michael</div>

6. 在页面底部点击【测试 web 服务】，同时【响应】标签出现值。

请求

响应

测试状态 已成功接收请求。

响应时间 (毫秒) 148

树视图

已生成测试实例。 启动跟踪

名称	类型	值
payload	payload	
result	string	Hello Michael

请求

响应

测试 Web 服务

21