

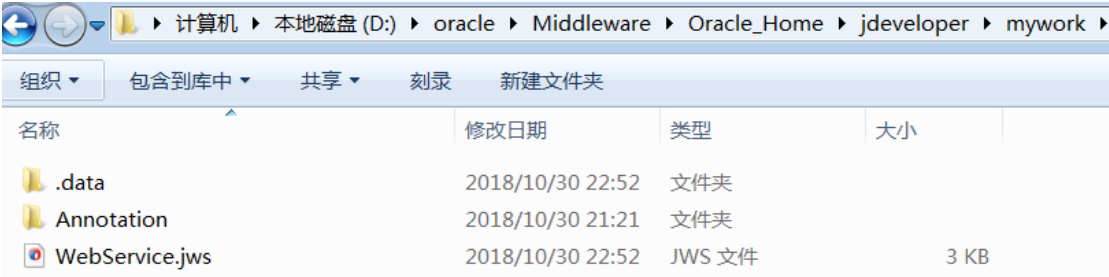
# 目录

第一部分： 建立一个 POJO Annotation-driven 服务.....1

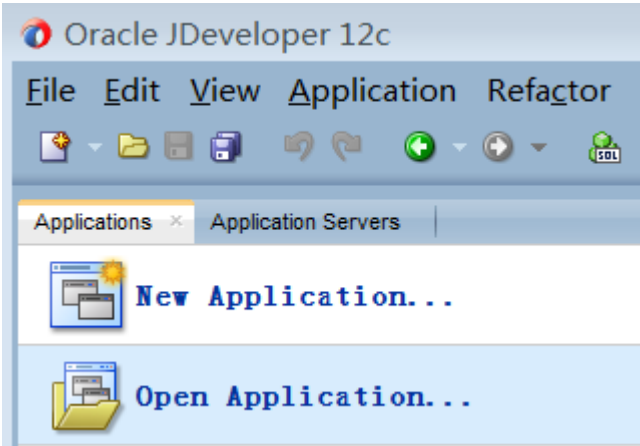
## 第一部分： 建立一个 POJO Annotation-driven 服务

### 步骤一：准备

- 1. 下载并解压 webservice.zip 文件到本地文件夹，本例为：  
D:\oracle\Middleware\Oracle\_Home\jdeveloper\mywork  
即



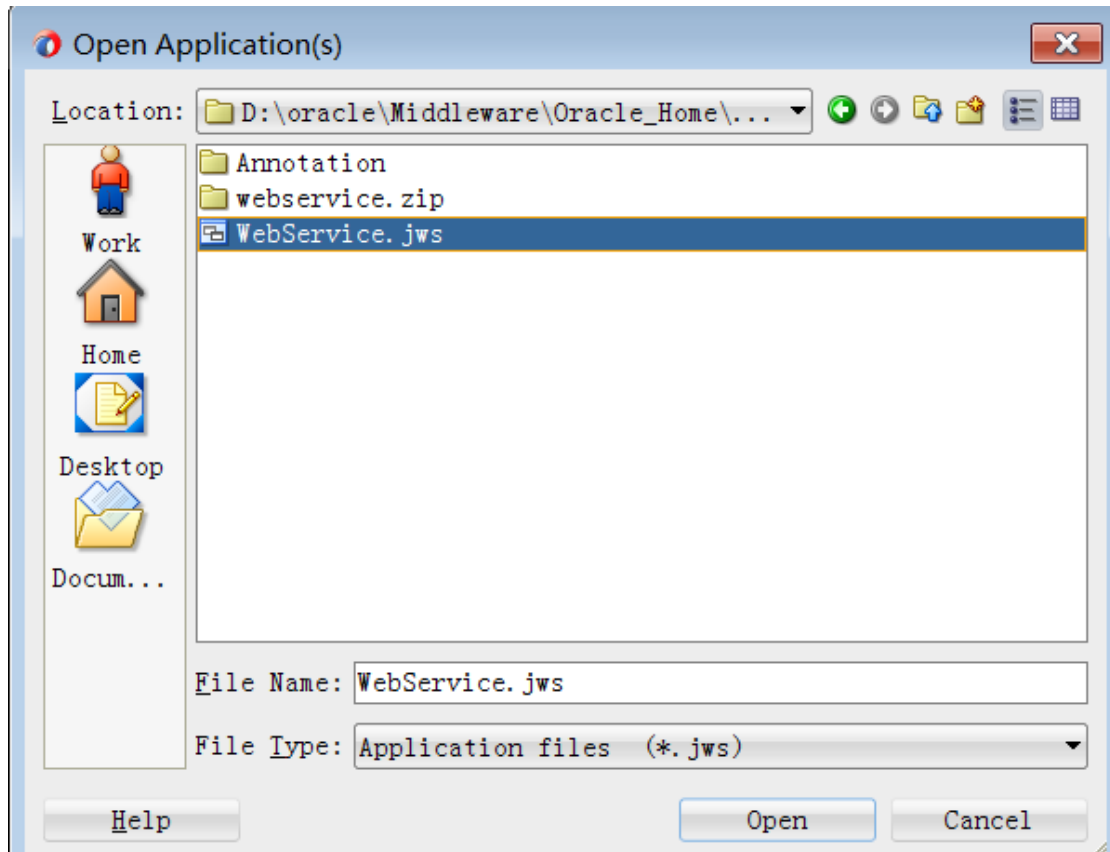
- 注：名为 mywork 的文件夹原先并不存在，是自己建立的，也可用其他文件夹名。
- 2. 启动 Oracle JDeveloper studio，studio 角色为 studio developer
  - 3. 选择 application 窗口标签并选择 open application（或 FILE->OPEN）



Open application...

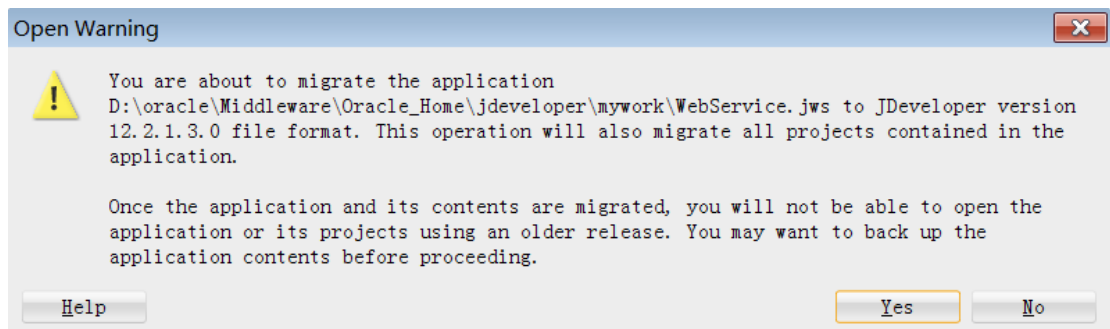
- 4. 找到解压 webservice.zip 文件后指定的文件夹，此处为  
D:\oracle\Middleware\Oracle\_Home\jdeveloper\mywork\webservice.jws  
即：

计算机 > 本地磁盘 (D:) > oracle > Middleware > Oracle_Home > jdeveloper > mywork				
组织	包含到库中	共享	刻录	新建文件夹
名称	修改日期	类型	大小	
.data	2018/10/30 22:52	文件夹		
Annotation	2018/10/30 21:21	文件夹		
WebService.jws	2018/10/30 22:52	JWS 文件	3 KB	

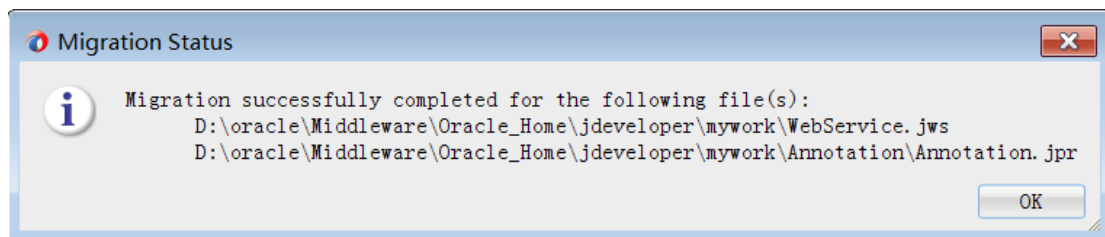
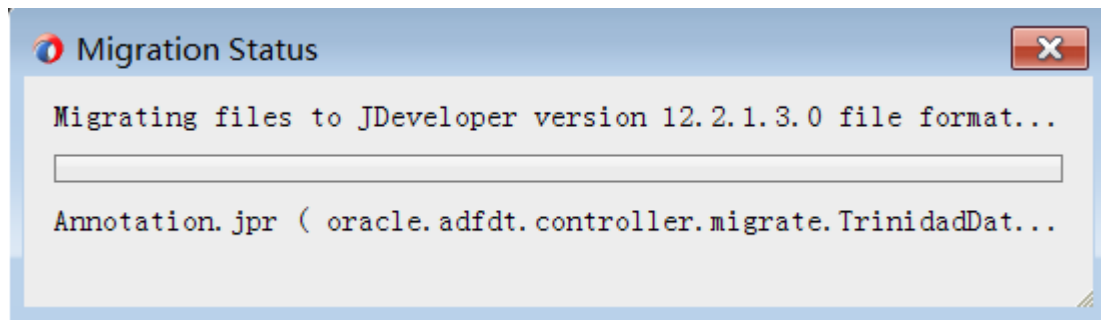


Open

5. 出现警告:

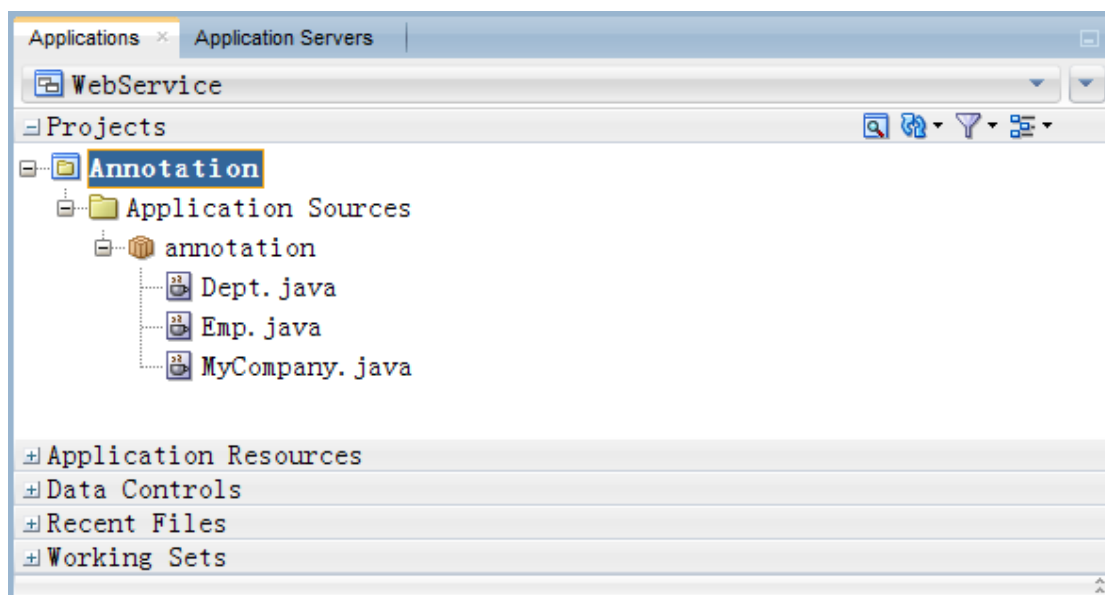


Yes



Ok

Applications window look like this:

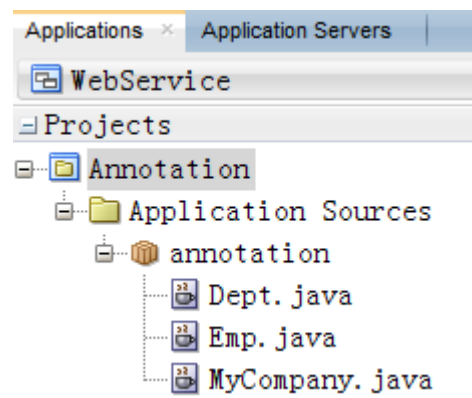


步骤二：添加一个 plain old java object (POJO)来包含一个 web service 方法

[Adding a Plain Old Java Object \(POJO\) to contain a Web Service Method](#)

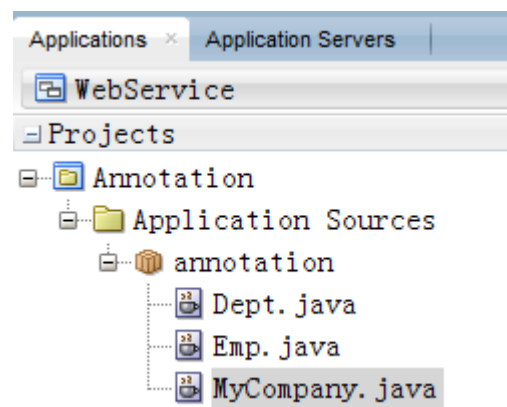
任务：

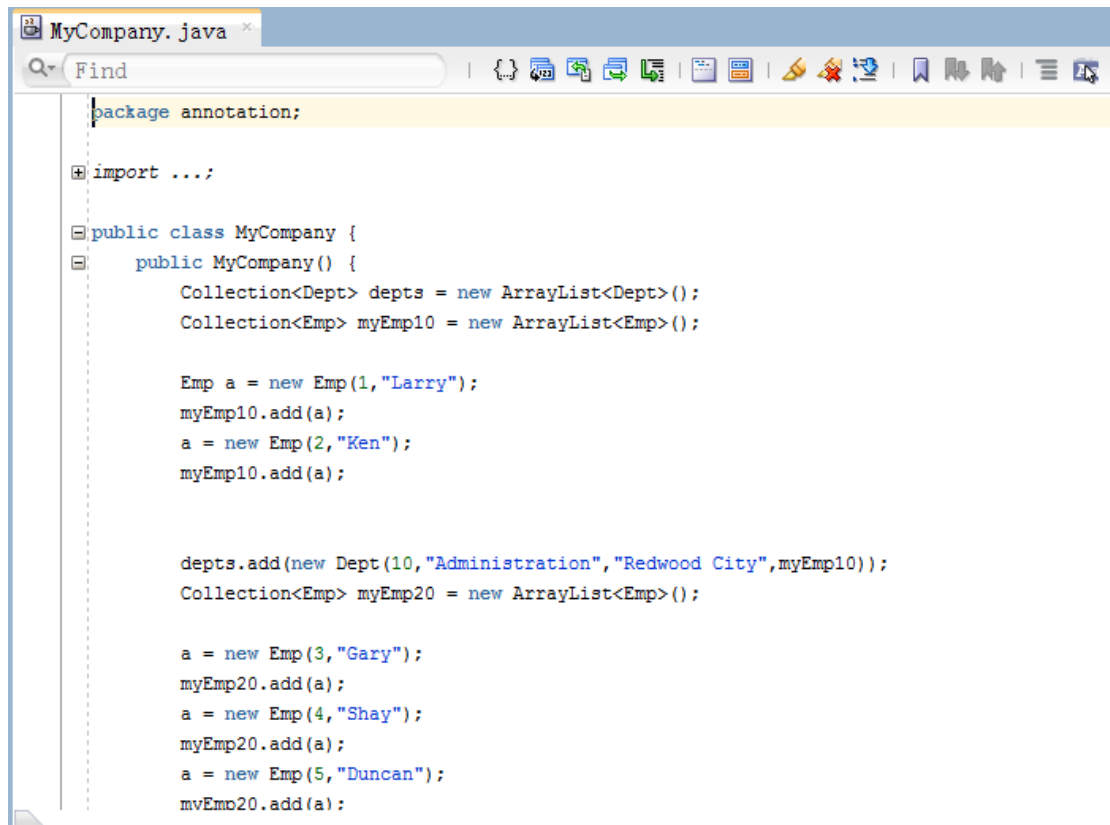
1. 在 Applications 窗口依次展开 Annotation 项目，显示 POJO 类：



- Dept.java: 描述 department 结构
- Emp.java: 描述 employee 结构
- MyCompany.java: populates information about departments and employees

2. 双击 MyCompany.java 进行编辑





The screenshot shows a Java IDE window titled 'MyCompany.java'. The code is as follows:

```
package annotation;

import ...;

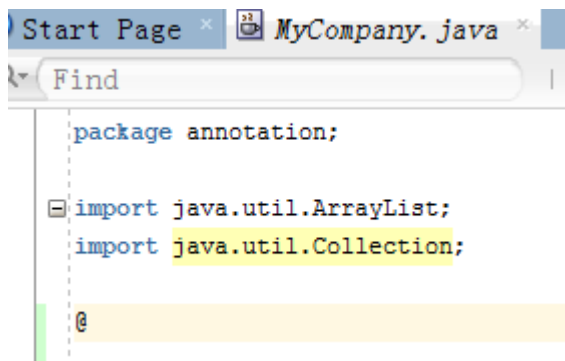
public class MyCompany {
    public MyCompany() {
        Collection<Dept> depts = new ArrayList<Dept>();
        Collection<Emp> myEmp10 = new ArrayList<Emp>();

        Emp a = new Emp(1, "Larry");
        myEmp10.add(a);
        a = new Emp(2, "Ken");
        myEmp10.add(a);

        depts.add(new Dept(10, "Administration", "Redwood City", myEmp10));
        Collection<Emp> myEmp20 = new ArrayList<Emp>();

        a = new Emp(3, "Gary");
        myEmp20.add(a);
        a = new Emp(4, "Shay");
        myEmp20.add(a);
        a = new Emp(5, "Duncan");
        myEmp20.add(a);
    }
}
```

3. 在 import 语句后添加@WebService annotation

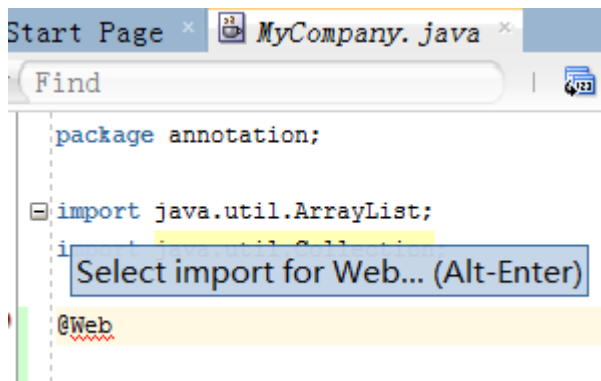


The screenshot shows the same IDE window. The import statements are highlighted in yellow:

```
package annotation;

import java.util.ArrayList;
import java.util.Collection;

@
```



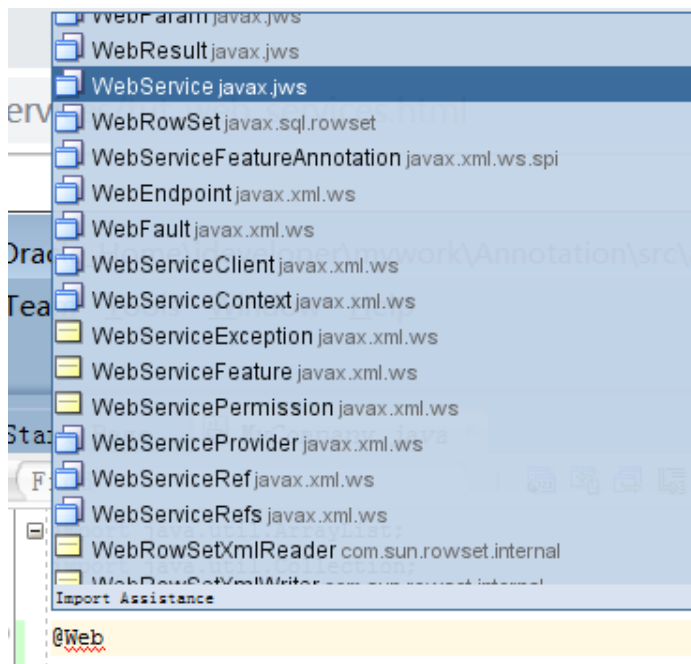
The screenshot shows the IDE window with a popup menu open over the '@' symbol. The popup text is 'Select import for Web... (Alt-Enter)'. Below the popup, the text '@Web' is visible.

```
package annotation;

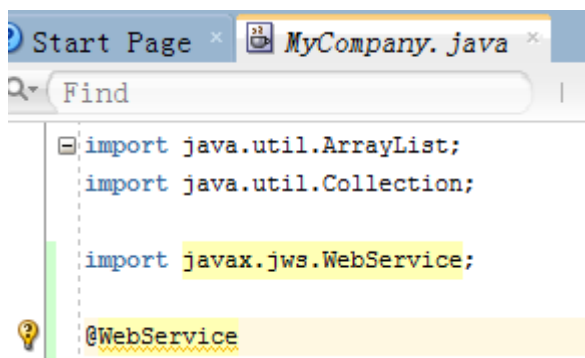
import java.util.ArrayList;
import java.util.Collection;

@Web
```


输入@Web时 JDeveloper 会自动弹出“select import for web...”，按 ALT-ENTER 组合键，出现下图。选择 WebService javax.jws。

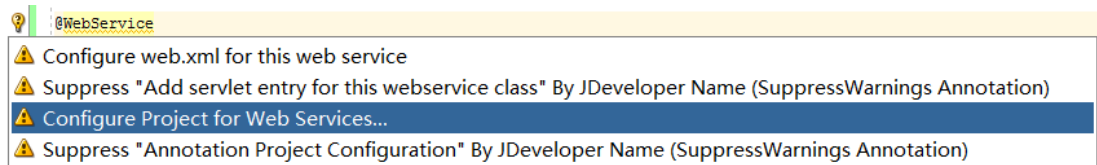


选择后的结果为下图。

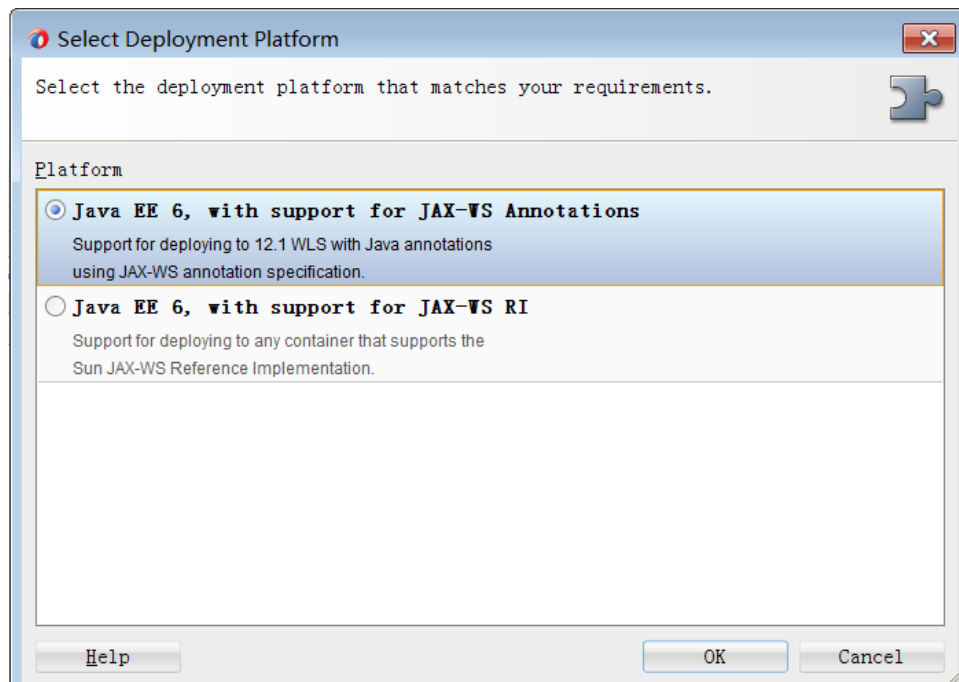


该 annotation 表明这个类包含了一个用于 web service 的方法。

- 鼠标左键点击  `@WebService` 左边的灯泡图标，弹出下图并选择“configure project for web services...”:

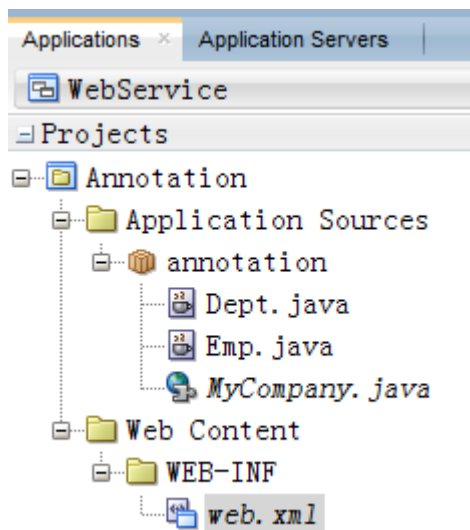


- 出现 select deployment platform 对话框，选择 “java EE 6, with support for JAX-WS annotations”。




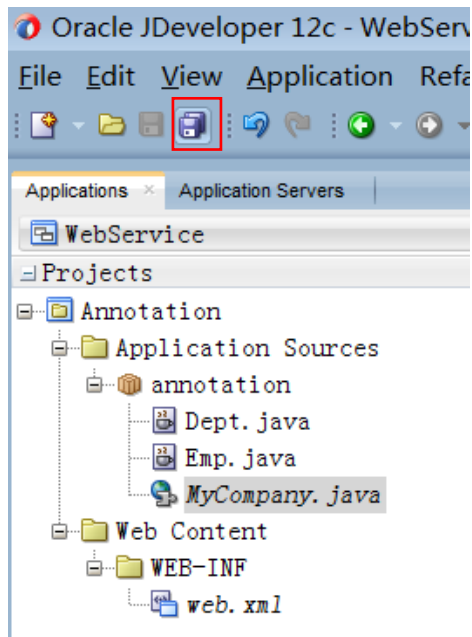
Ok

- 上面这一步是添加 `javax.jws.webservice import` 语句到 java 类（如果不存在的话），并创建一个 `web.xml` 文件。在 JDeveloper 的左侧 Applications 标签栏可以看到下图：



上图中的 `MyCompany.java` 类的图标已经改变为表示 `WebService` 类，同时项目中添加了 `web.xml` 文件。

- 点击 “save all  ”保存。

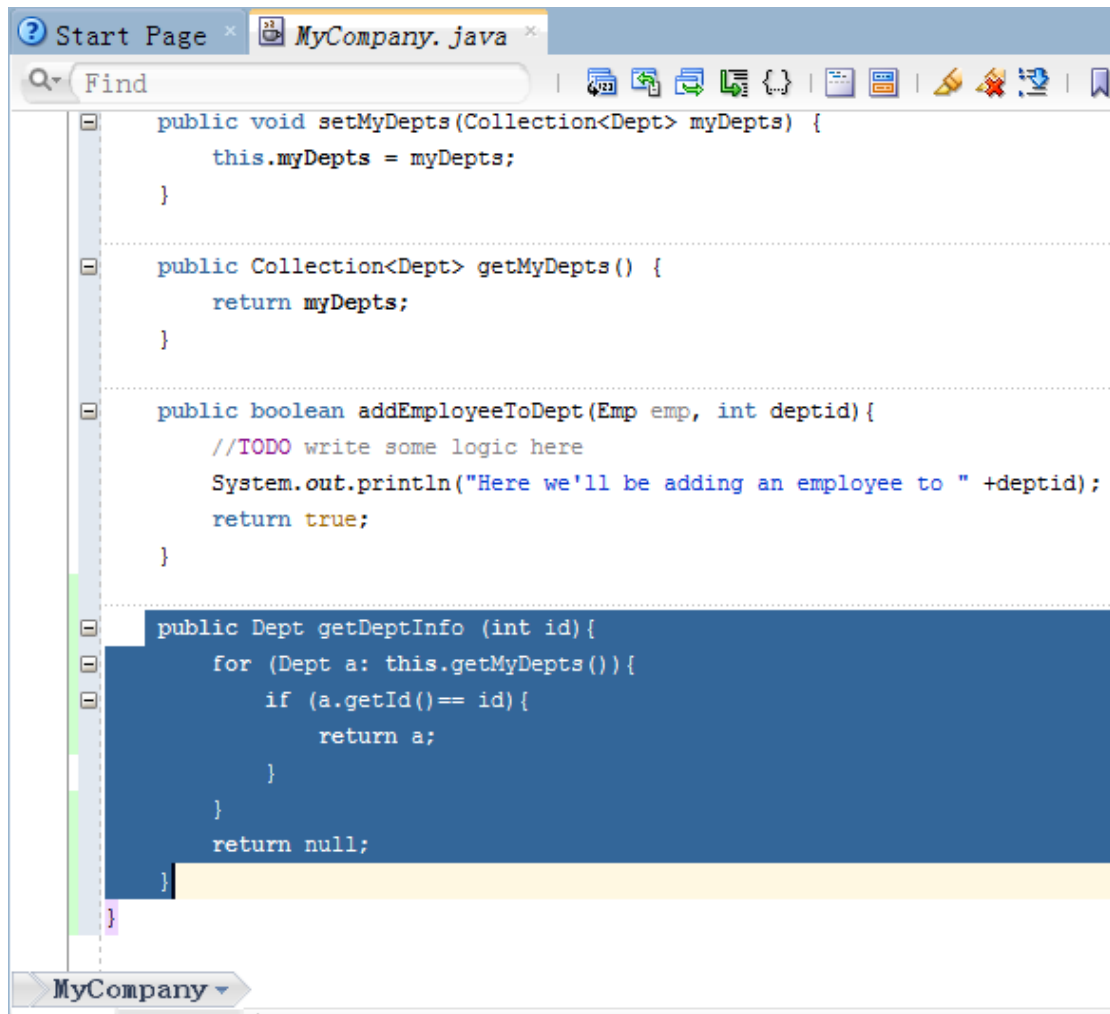


8. 在 code editor，滚动到类的最底部添加以下代码：

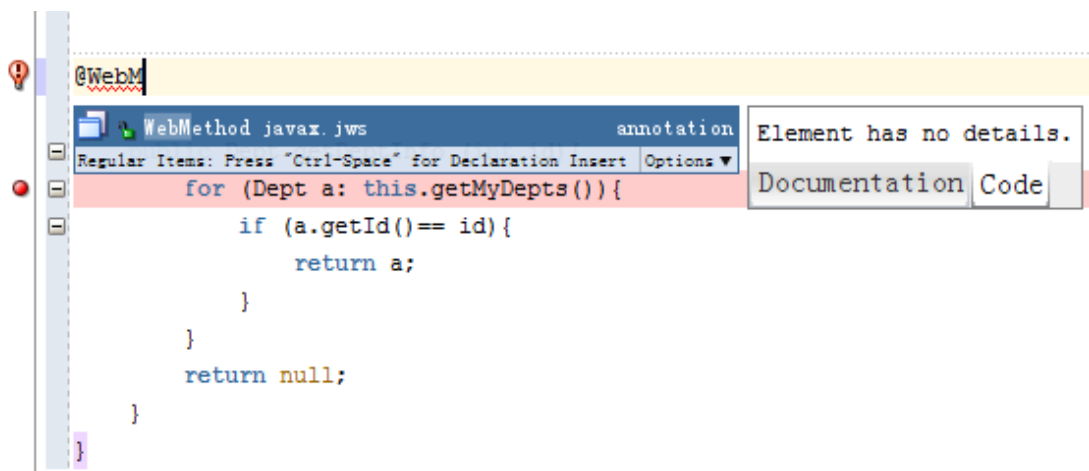
```
public Dept getDeptInfo (int id) {  
    for (Dept a: this. getMyDepts() ) {  
        if (a.getId() == id) {  
            return a;  
        }  
    }  
    return null;  
}
```

添加代码后的界面：

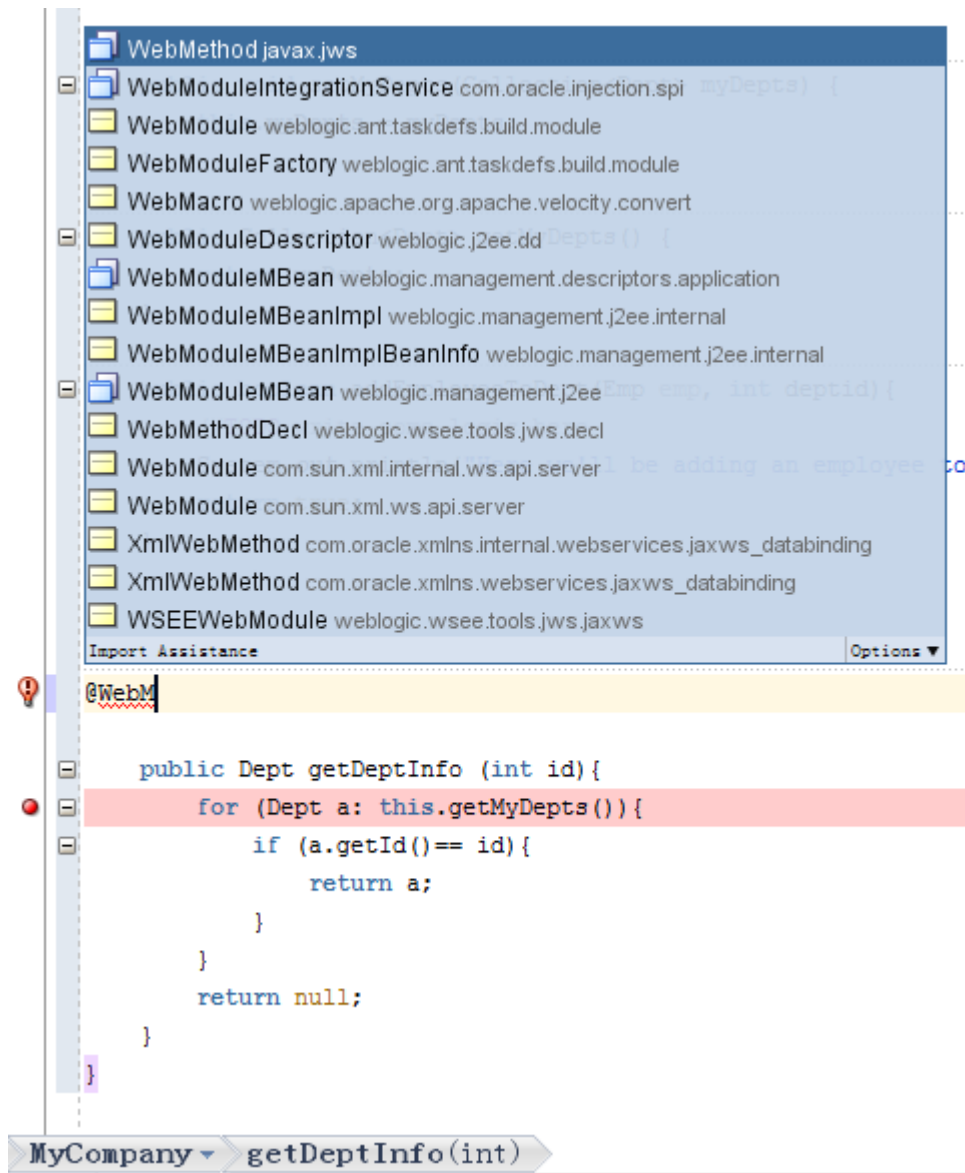




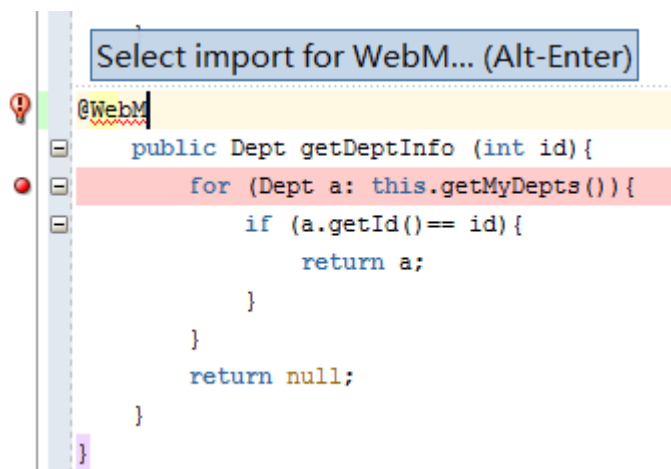
9. 在 getDeptInfo()方法前创建第二个 annotation, 这个 annotation signifies this is the method to be exposed from the web service. (担心中文表达不恰当, 保留原来的英文说法)。
- 在 getDeptInfo()方法的上面添加一个空行 → 键入: @WebMethod →在弹出可用语法列表中选择 WebMethod javax.jws



或



或在下面情形同时按“Alt-Enter”，再选择。



- 新添加的 import javax.xml.ws.WebMethod 会出现在之前添加的 import javax.xml.ws.WebService 之前，下图：

```
package annotation;

import java.util.ArrayList;
import java.util.Collection;

import javax.jws.WebMethod;
import javax.jws.WebService;

@WebService

public class MyCompany {
    public MyCompany() {
        Collection<Dept> depts = new ArrayList<Dept>();
        Collection<Emp> myEmp10 = new ArrayList<Emp>();
    }
}

....

@WebMethod

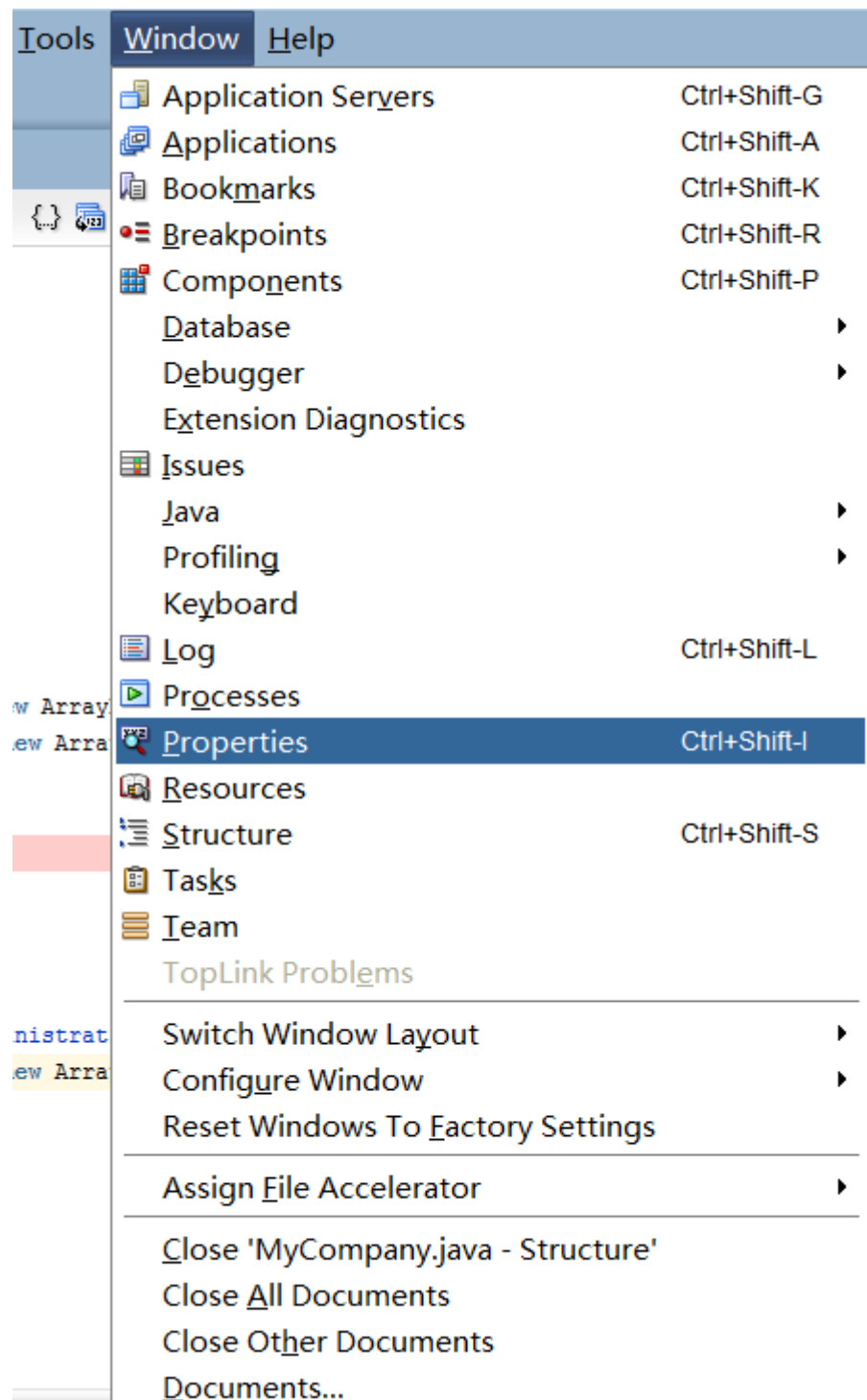
public Dept getDeptInfo (int id){
    for (Dept a: this.getMyDepts()){
        if (a.getId()== id){
            return a;
        }
    }
}

MyCompany ▾ MyCompany()
```

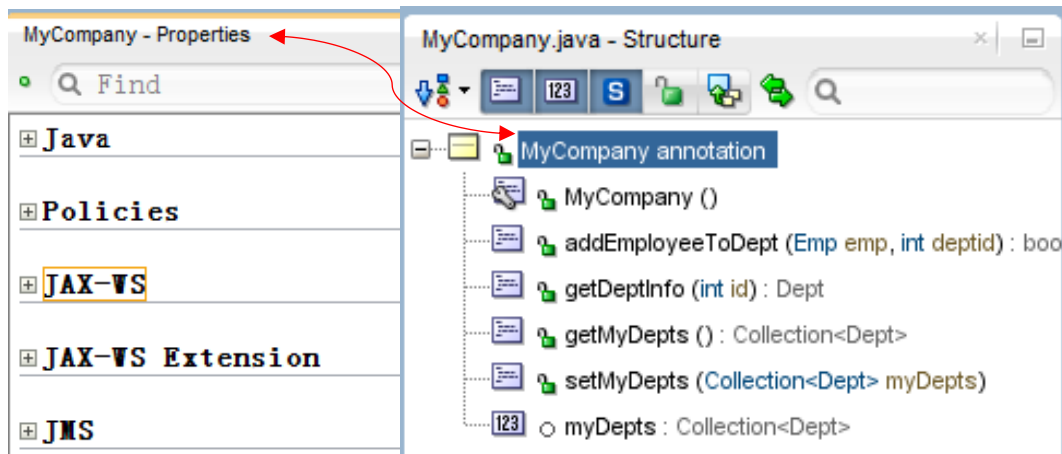
10. 点击“Save All” 图标以保存上述操作。

11. MyCompany 类性质的修改。

在菜单栏选择 “Window→Properties”。

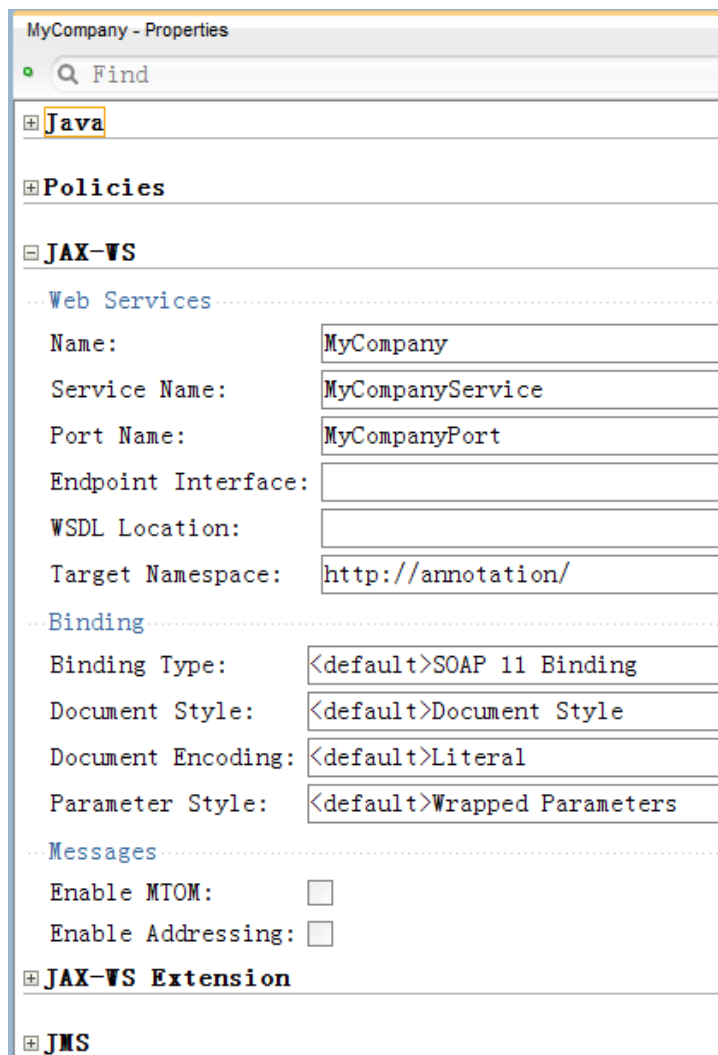


弹出“MyCompany-Properties”窗口，



上左图的 MyCompany-Properties 对应上右图的 MyCompany annotation 内容。

12. 在“MyCompany-Properties”窗口，展开 “JAX-WS”结点，下图。



13. 把上图 Service Name 中的值 MyCompanyService 改为 MyCompanyWS。下图

## JAX-WS

Web Services	
Name:	MyCompany
Service Name:	MyCompanyWS
Port Name:	MyCompanyPort
Endpoint Interface:	
WSDL Location:	
Target Namespace:	http://annotation/
Binding	
Binding Type:	<default>SOAP 11 Binding
Document Style:	<default>Document Style
Document Encoding:	<default>Literal
Parameter Style:	<default>Wrapped Parameters
Messages	
Enable MTOM:	<input type="checkbox"/>
Enable Addressing:	<input type="checkbox"/>

14. 点击 Save All 图标保存。

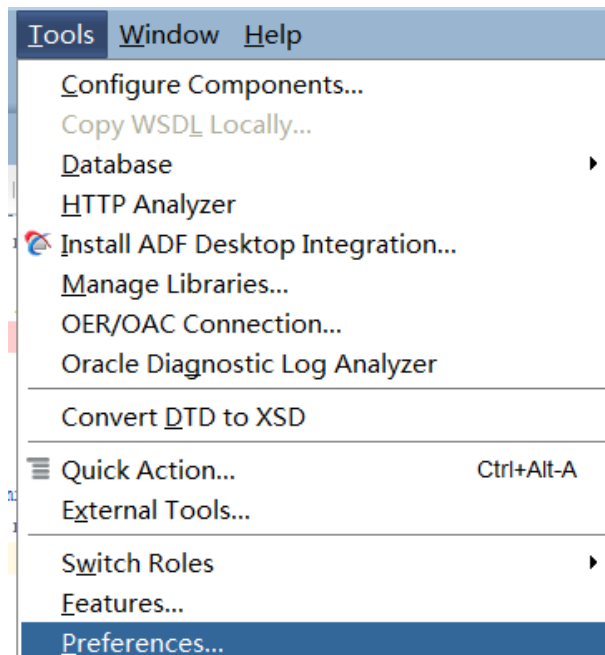
15. 这样就创建了一个 POJO web service。下一步将测试该 web service。

## 步骤三：测试 web service

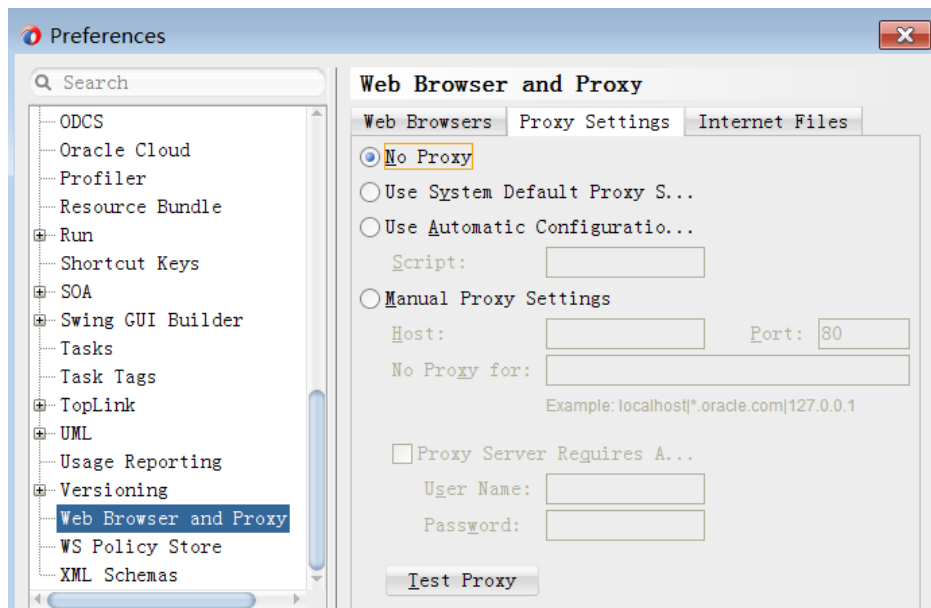
任务：编译、配置和测试 web service。

- JDeveloper 包含一个用于测试 web service 的机制（称为 HTTP Analyzer）。当使用 HTTP Analyzer 来测试 web service 时，JDeveloper 会自动编译和配置 web service 到集成的 web server 中。JDeveloper 调用 analyzer，允许进行发送和接收来自 web service 的值。

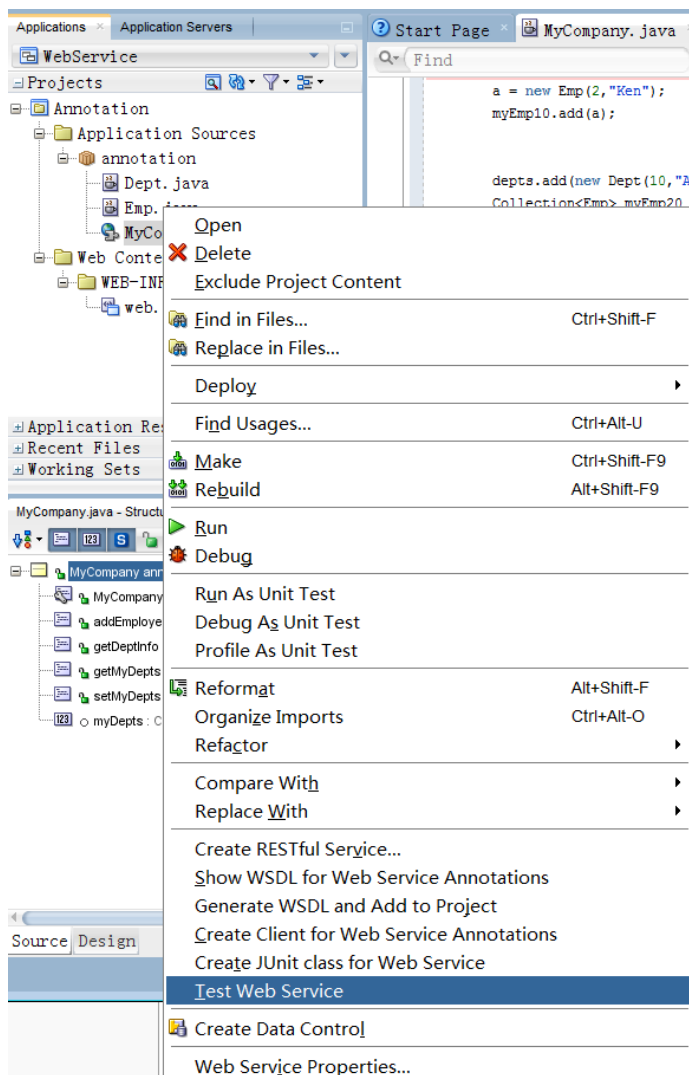
### 1. Web 浏览器设置



依次执行：Web browser and proxy -> proxy setting->no proxy.

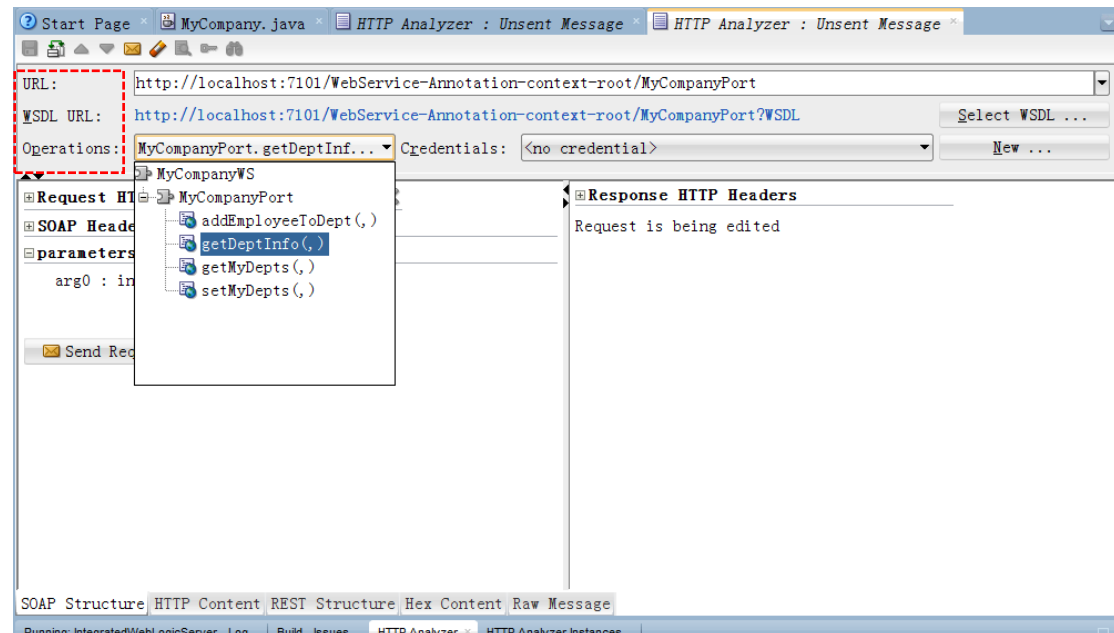


2. 在 application 窗口，右键点击 MyCompany.java 节点，选择 TEST WEB SERVICE.

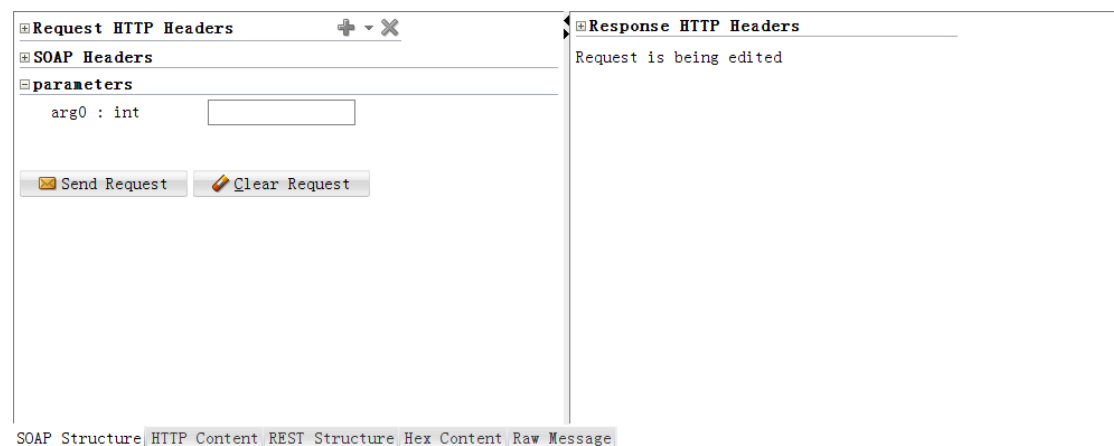


JDeveloper 将调用 integrated weblogic server（首次运行可能需要一些时间），配置服务，然后启动 analyzer。

在 HTTP Analyzer 编辑器窗口的顶部显示 web service 的 URL，WSDL URL，和 operations。在 operations 的下拉菜单中选择 MyCompanyPort.getDeptInfo(,)。



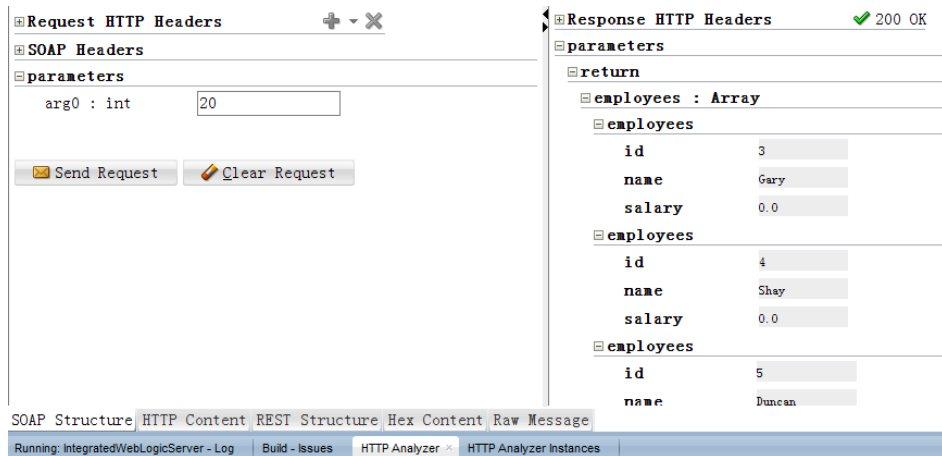
在 HTTP Analyzer 编辑器窗口的底部左边显示 request，右边显示 response，下图。



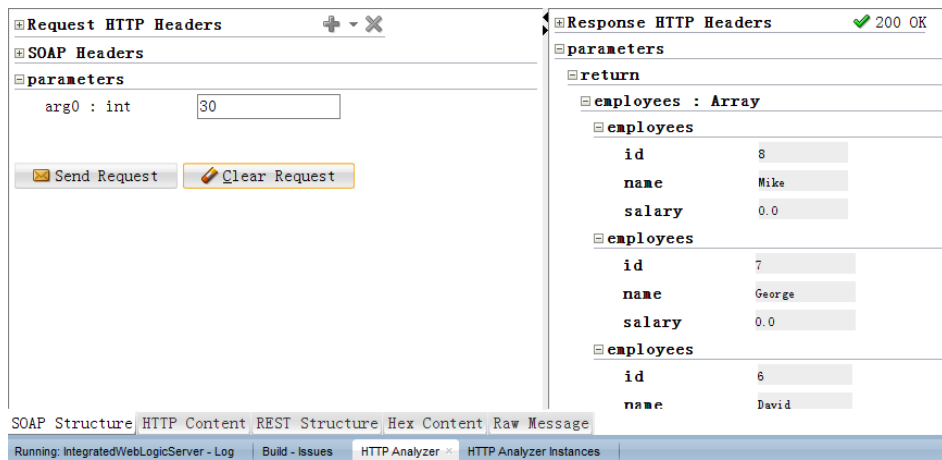
Request 窗口显示 exposed 方法的所有变量（本例中只有一个变量 arg0）。执行 web service 后右边 response 显示执行结果。

3. 在 arg0:int 输入框内键入 20→点击“send request”→在 response 窗口可看到结果，下图：

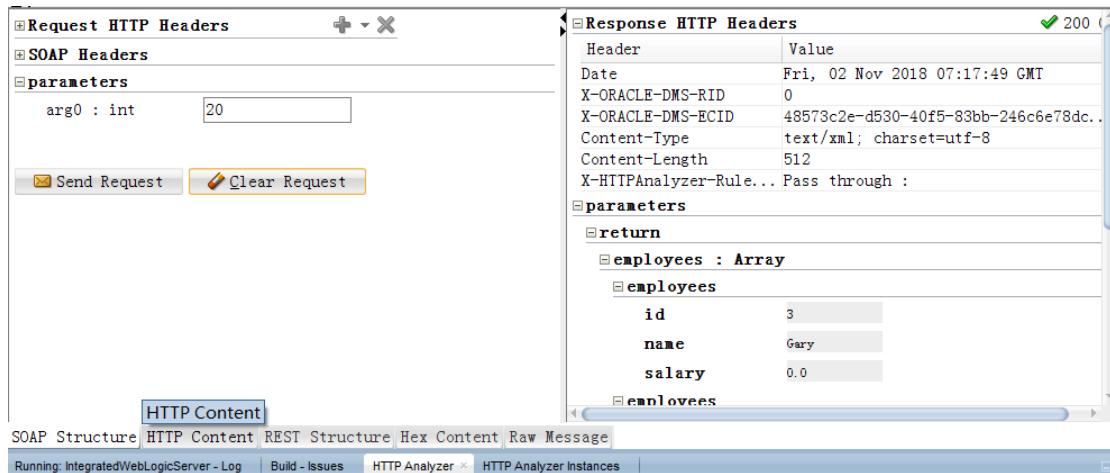




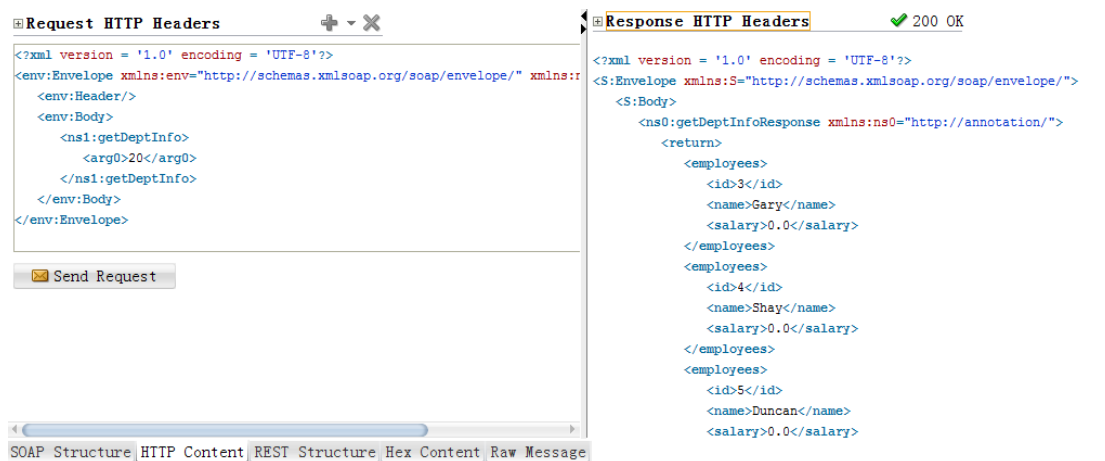
输入 30 的结果。



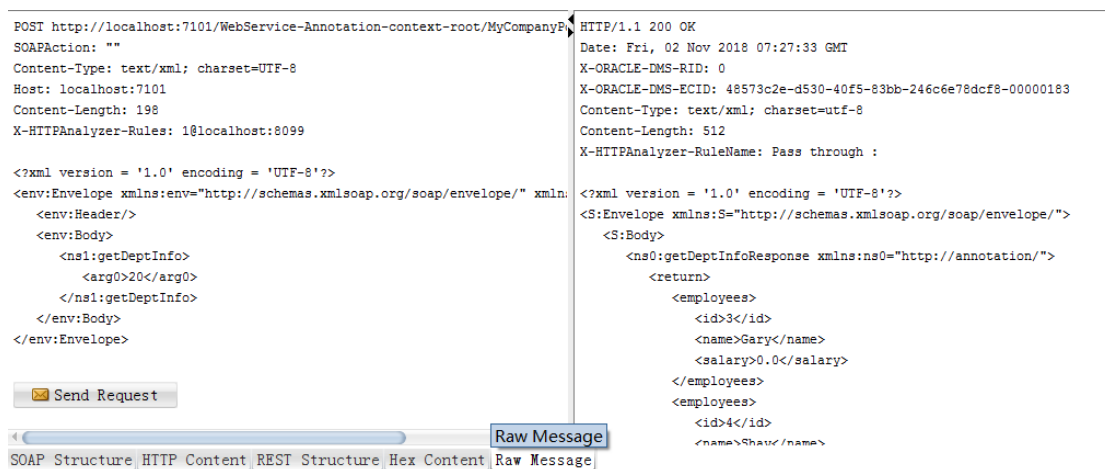
4. 点击下图底部的 HTTP content 标签



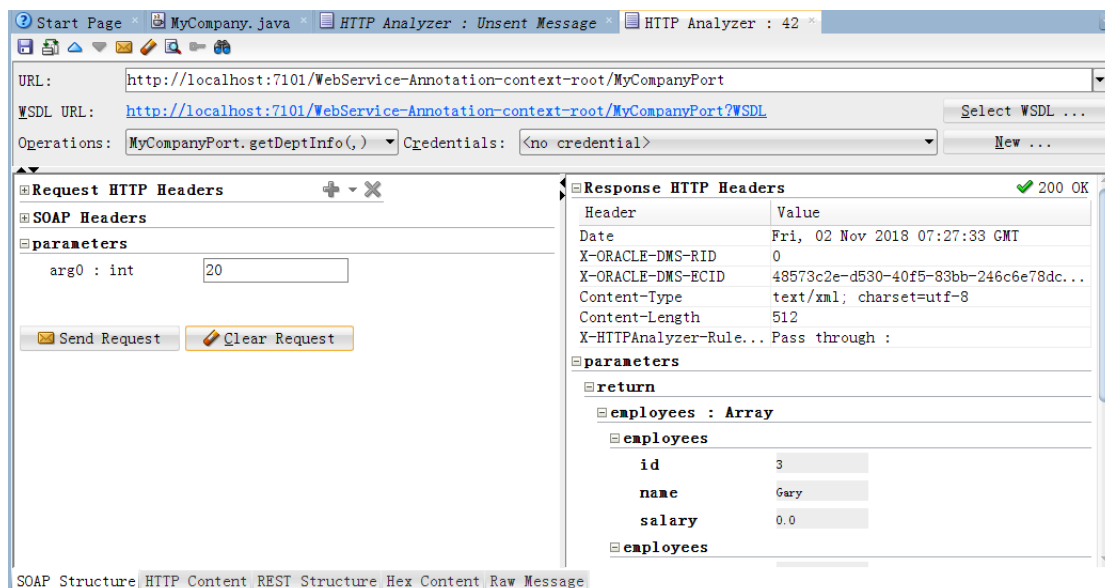
可以看到 xml 代码:



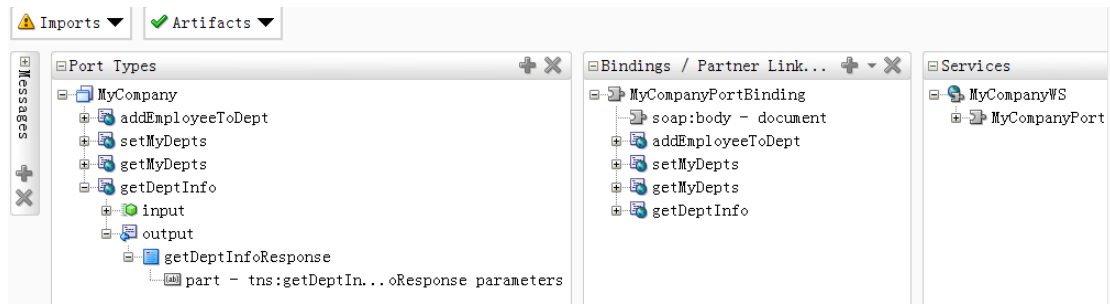
5. 点击底部的 raw message 标签，可以看到其他格式的代码：



6. 点击底部的 SOAP structure 标签返回。

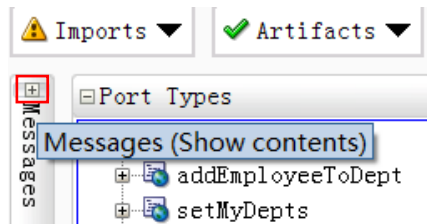


单击上面的 WSDL URL 链接，将打开一个新的 web service 可视编辑器，下图：

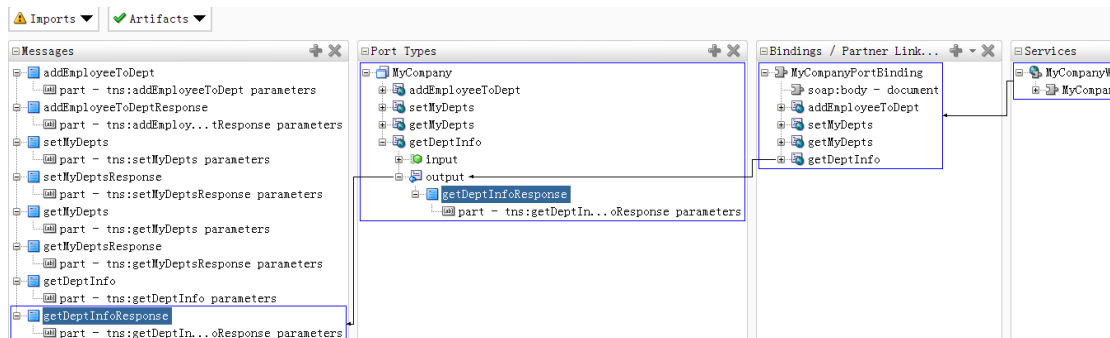


在上图中依次展结点 `getDeptInfo`→`output`→ `getDeptInfoResponse`

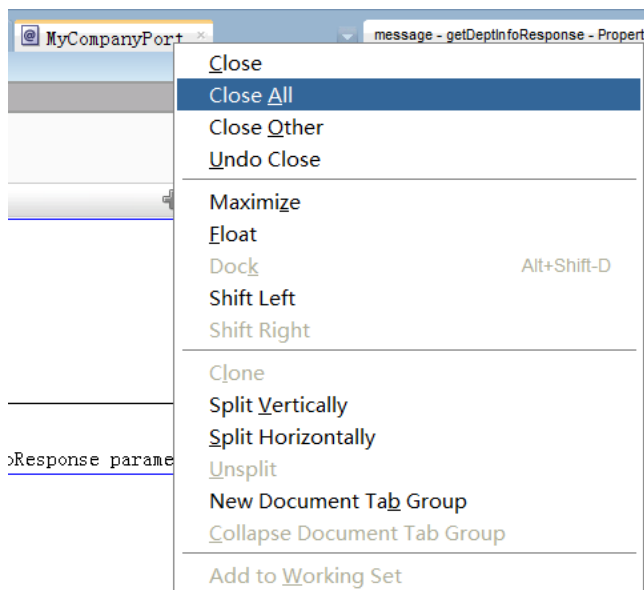
7. 点击 port types 面板的左侧 messages 上面的加号以显示消息内容。



得到下图：



8. 右击 MyCompanyPort 标签→选择 close all



9. 在 application 窗口收缩 annotation 项目结点。

