# 遗留系统封装说明

## 封装使用说明

### 0.1 JDK版本

需要确保JDK版本为1.8，否则会报Endpoint类的ClassNotFoundException异常

### 0.2 本地调用方式

先在Initiator类中将服务通过Endpoint进行发布。

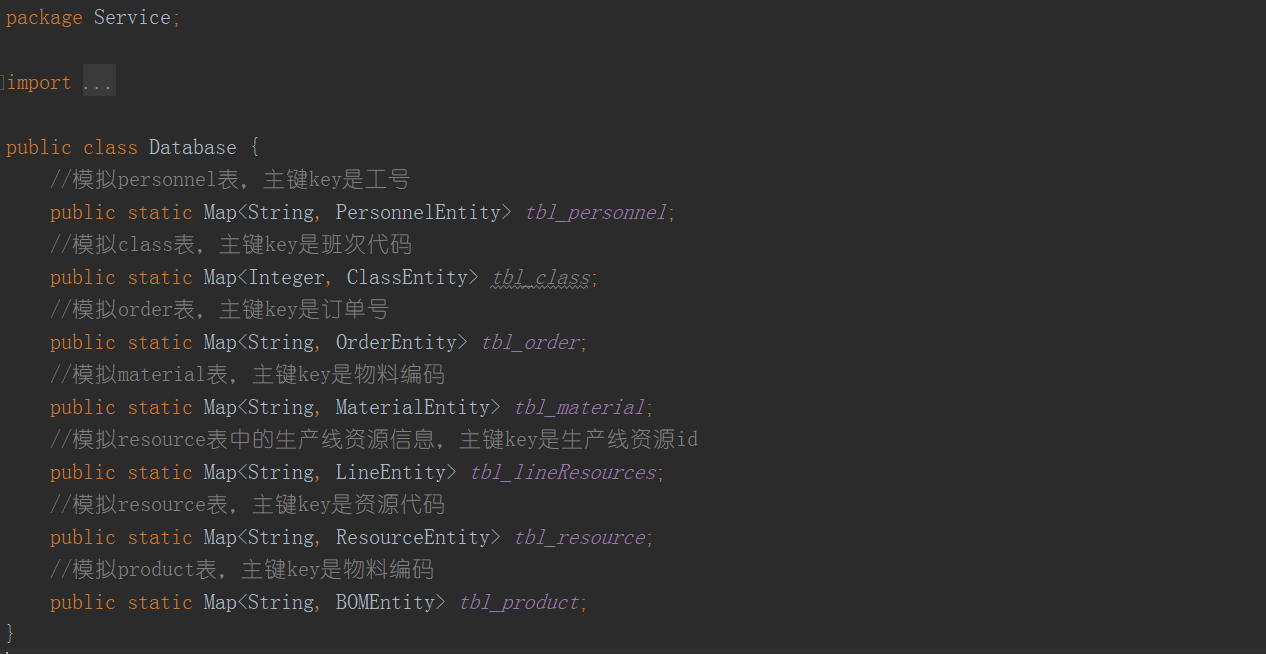
然后在本地通过创建\*Service类型的服务对象对服务进行调用，具体见**[3. 服务调用 & 测试结果](#_服务调用 & 测试结果)。**

### 0.3 客户端调用方式

首先需要确保客户端类实例化之前，相应的服务在服务端已经发布，这样在客户端就能通过服务地址获取WSDL文档，进而结合QName来创建服务接口的具体服务实例，然后调用其中的服务。在测试客户端程序调用时，需要通过@Before在每次单元测试前，将服务发布出去，然后获取到调用相应服务所需要的客户端对象，同时为了确保服务不会重复发布而导致绑定异常出现，@Before的代码中加入了相关判断，具体见[4. 客户端代码 & 测试结果](#_客户端代码 & 测试结果)。

### 0.4 数据模拟方式

创建一个Database类，在其中通过静态类型的Map对象来模拟数据库中的表。



### 0.5 Xslx表格的读取

读取过程中为了保证可扩展性，将数据为空，但表头不为空的内容作为空字符串读入，从而保证在未来xslx表格发生变更时，能够更容易的修改代码并进行扩展。

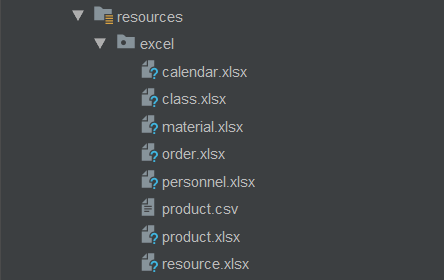
详见[util.ExcelReadUtil.java](LegacySystem/src/main/java/Service/util/ExcelReadUtil.java)。

### 0.6 编译说明

在读取Xlsx表格文件的过程中，使用了相对路径，而该路径的获取是针对编译产生的class文件进行的，所以如果运行过程中出现因为读取文件导致的NullPointerException异常，可以将target文件夹删除，并重新编译运行。

### 0.7 数据表拆分

将原始多表xlsx拆分成多个xlsx并放在resources.excel文件夹下，从而能够更好的和Database类中模拟的数据库表Map对象进行映射。

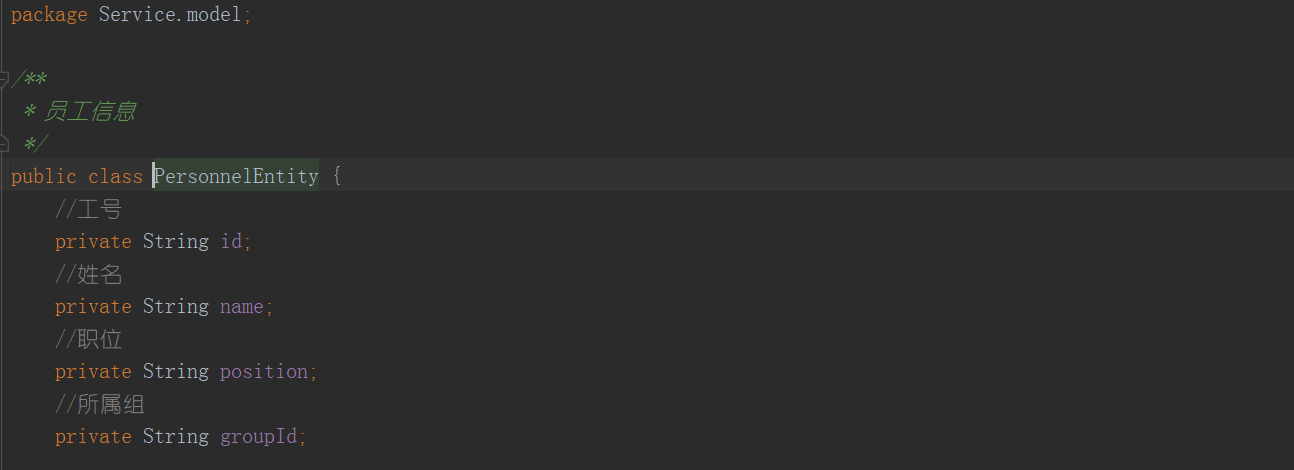


### 0.8 开发方式

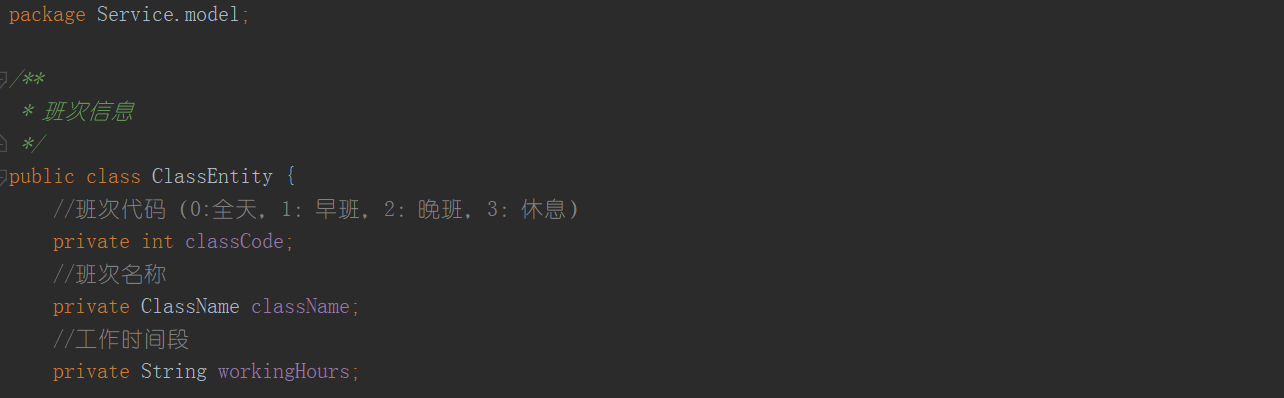
开发采用Start-from-Java的开发方式。通过先行开发Java Web Service，通过@WebService注释来让JWS来自动生成相应的WSDL文档。

## 数据模型

### 员工信息



### 班次信息



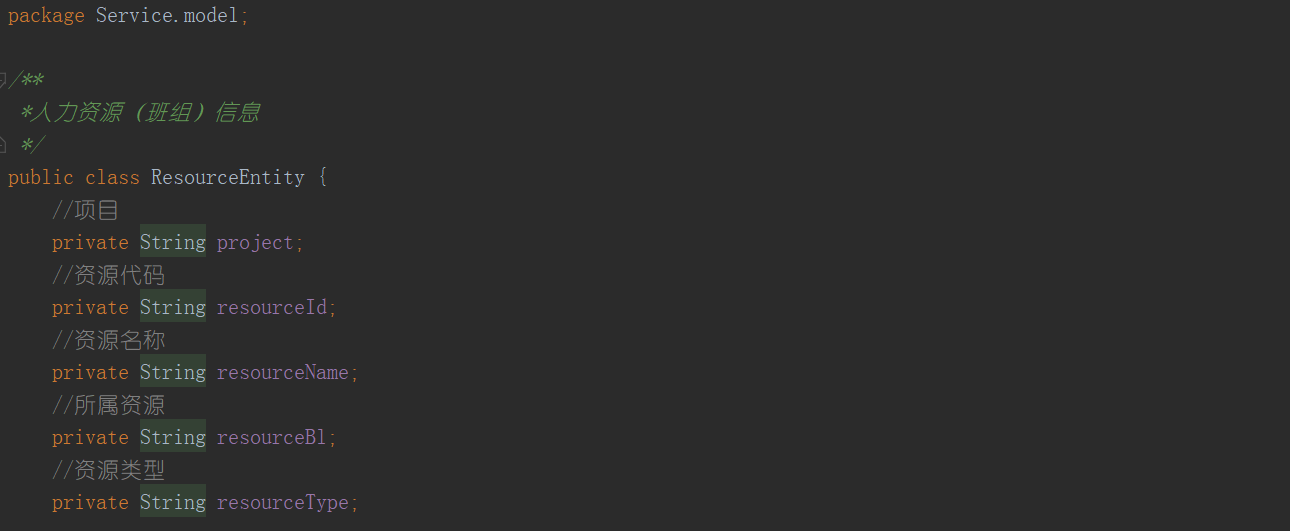
* + 1. 班次名称



### 生产线信息



### 人力资源班组信息



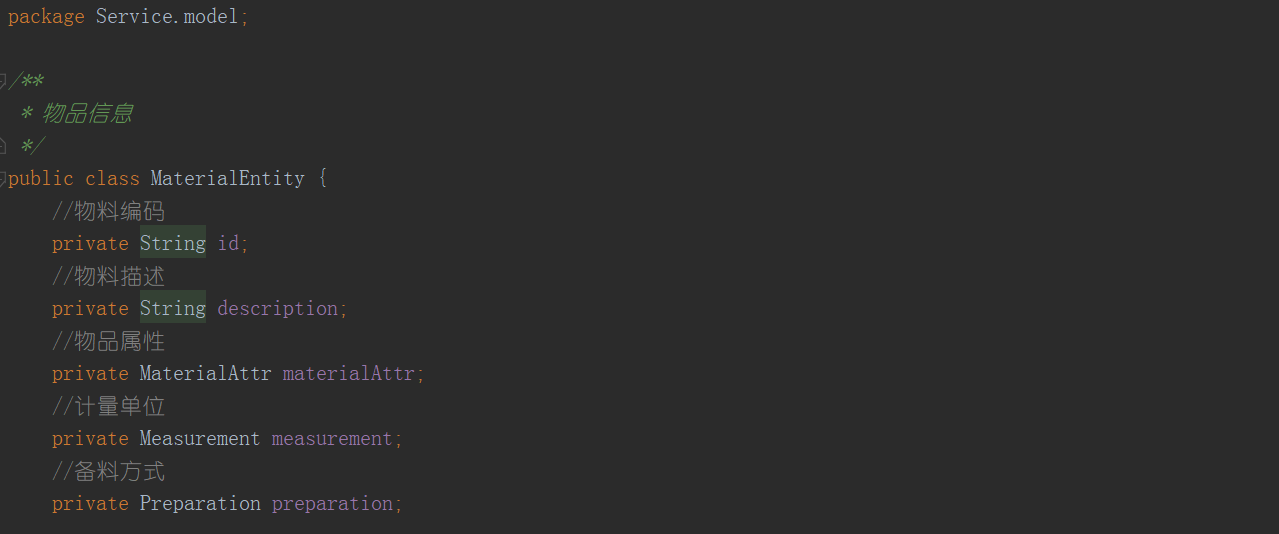
### 人力资源排班信息



* + 1. 工作日



### 物品信息



* + 1. 物品属性



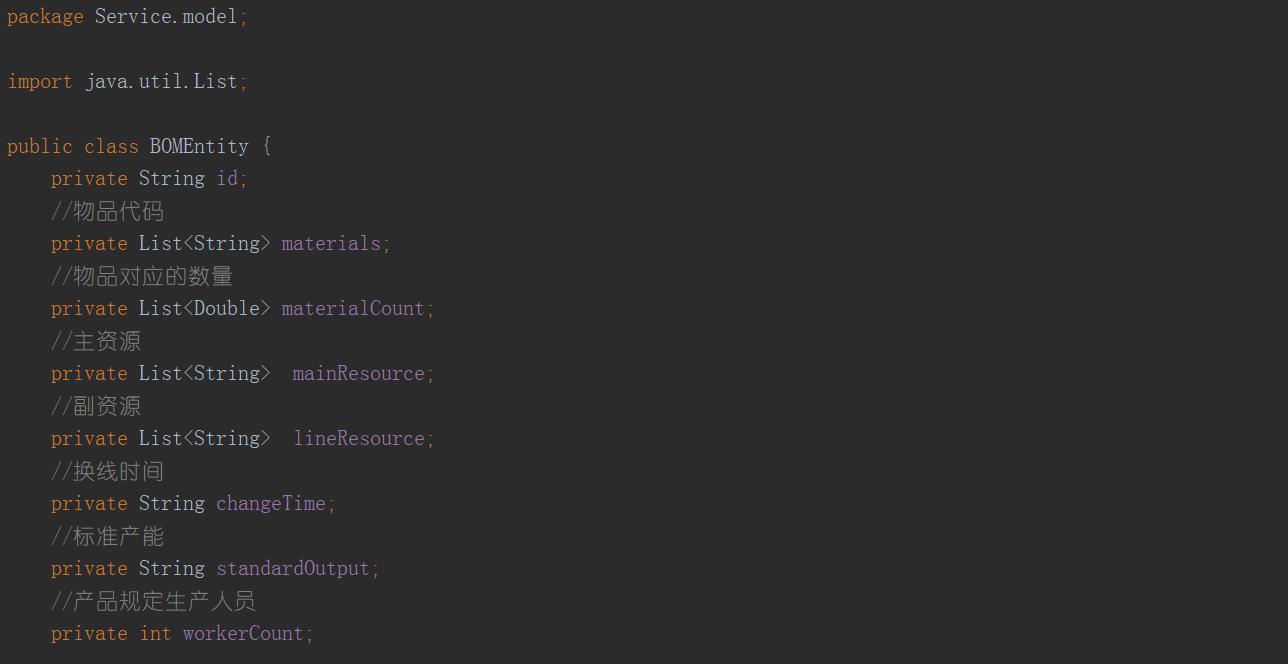
* + 1. 计量单位



* + 1. 准备方式



### BOM信息



### 订单信息



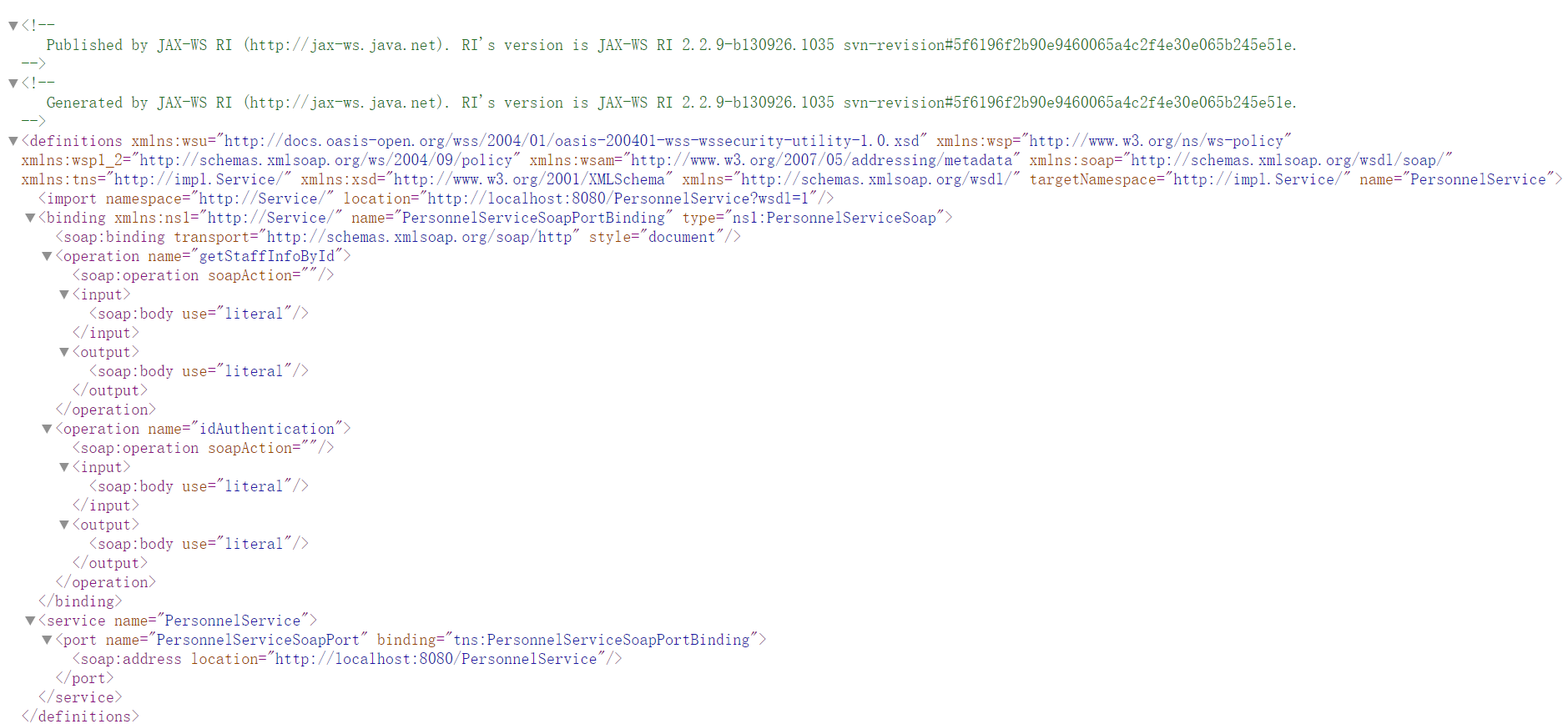
## 遗留系统服务接口 & WSDL

### 人事系统

2.1.1 服务接口



2.1.2 WSDL文档

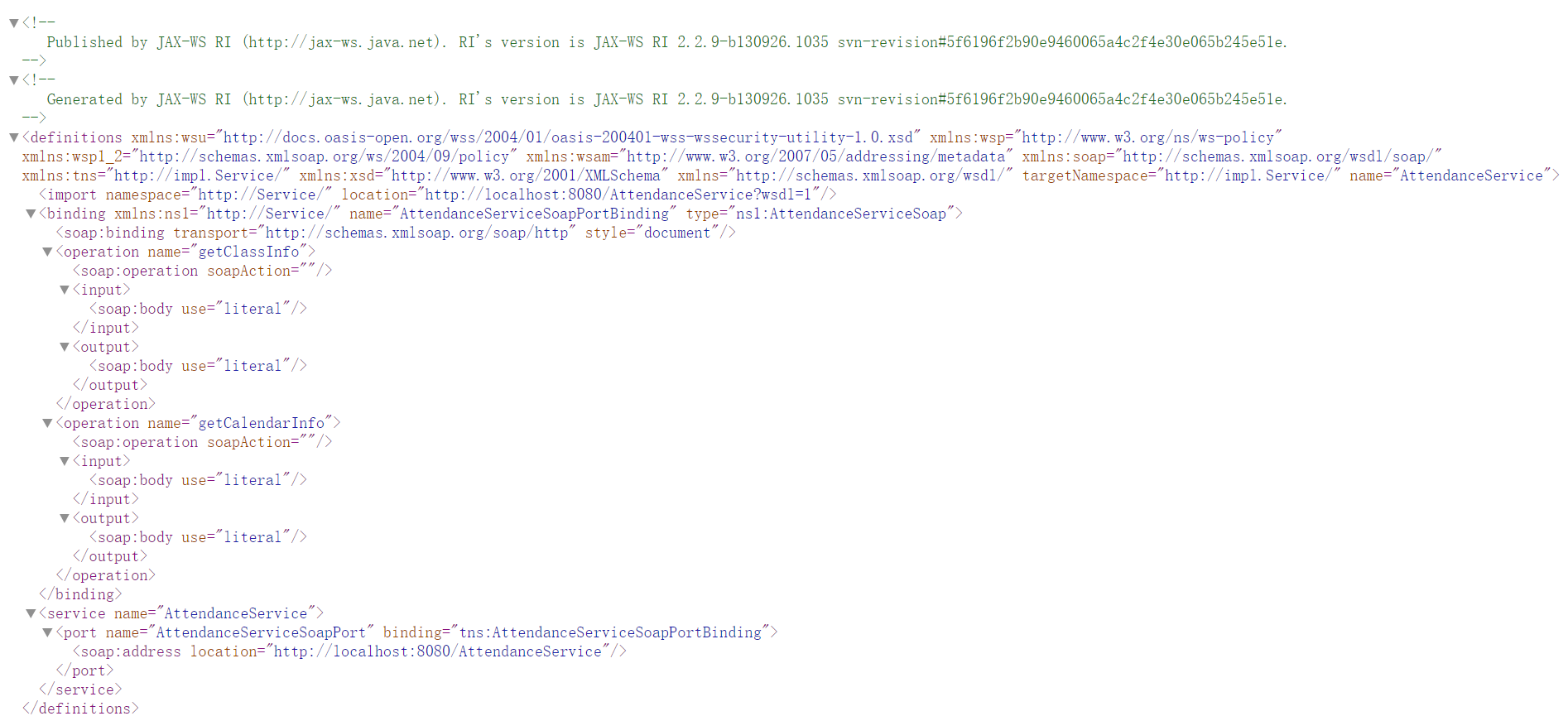


### 考勤系统

* + 1. 服务接口

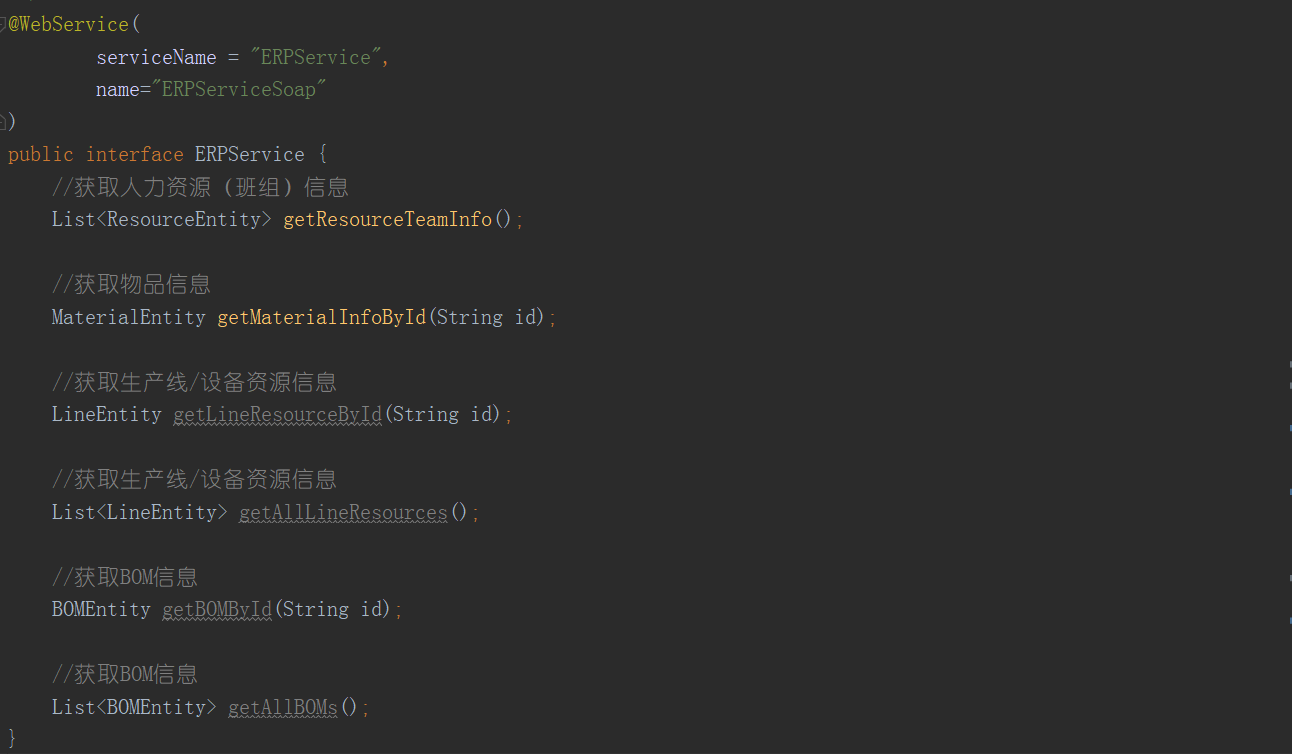


* + 1. WSDL文档



### ERP系统

* + 1. 服务接口

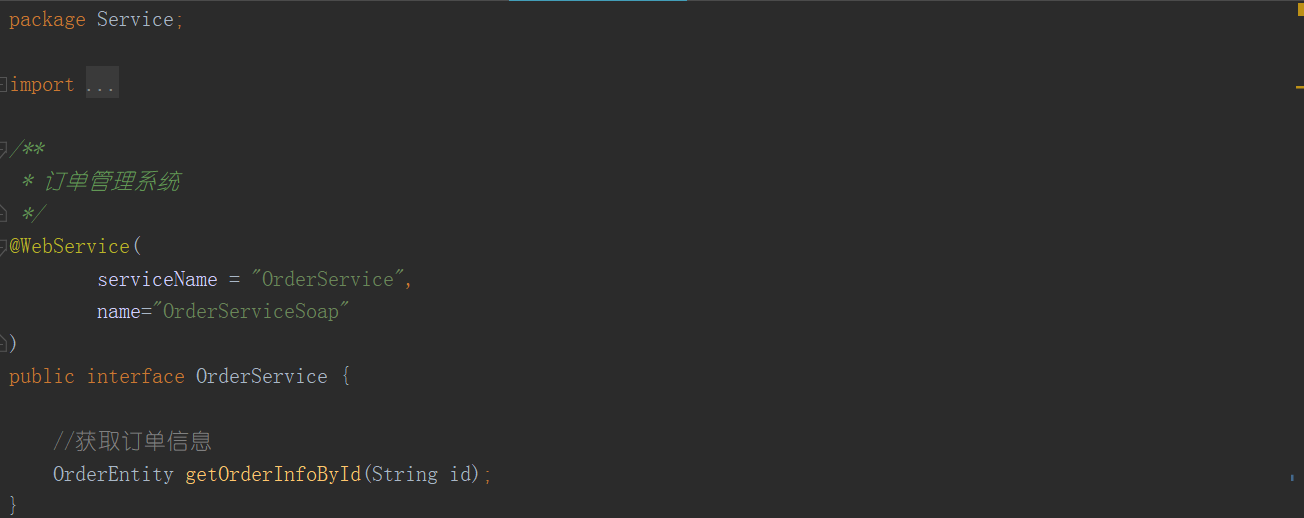


* + 1. WSDL文档



### 订单管理系统

* + 1. 服务接口



* + 1. WSDL文档



## 服务调用 & 测试结果

### 人事系统

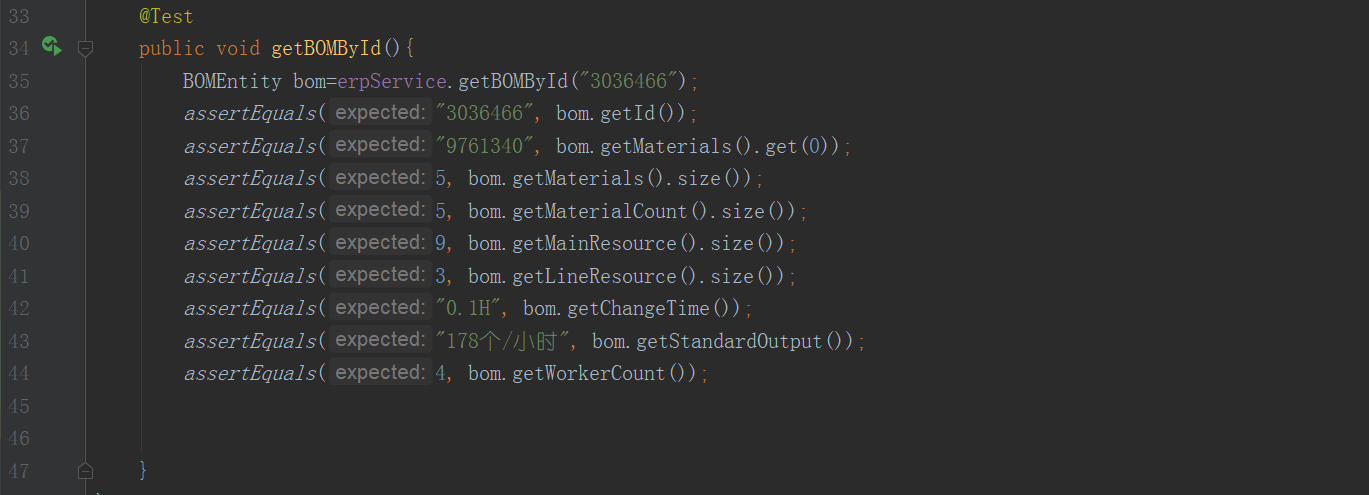


### 考勤系统



### ERP系统





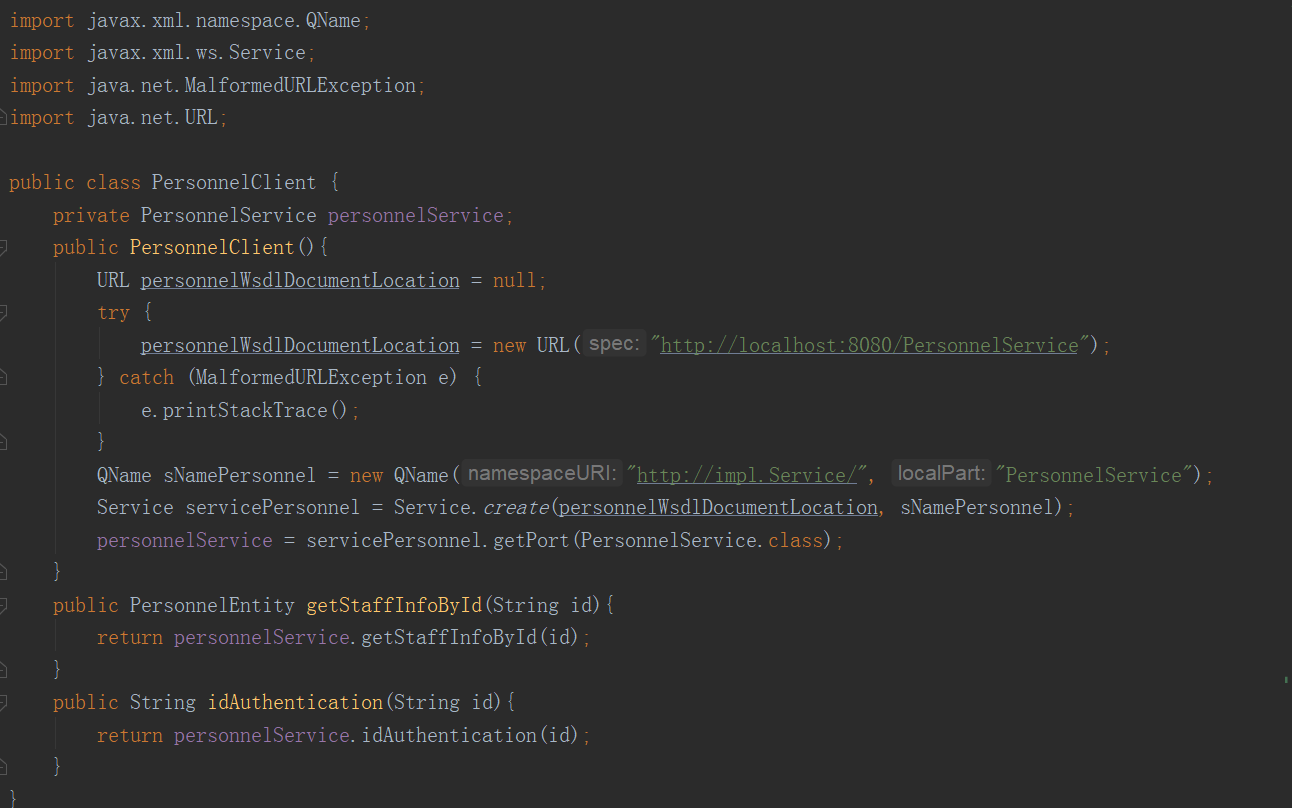
### 订单管理系统



## 客户端代码 & 测试结果

### 人事系统

* + 1. 客户端程序



* + 1. 客户端测试



### 考勤系统

* + 1. 客户端程序

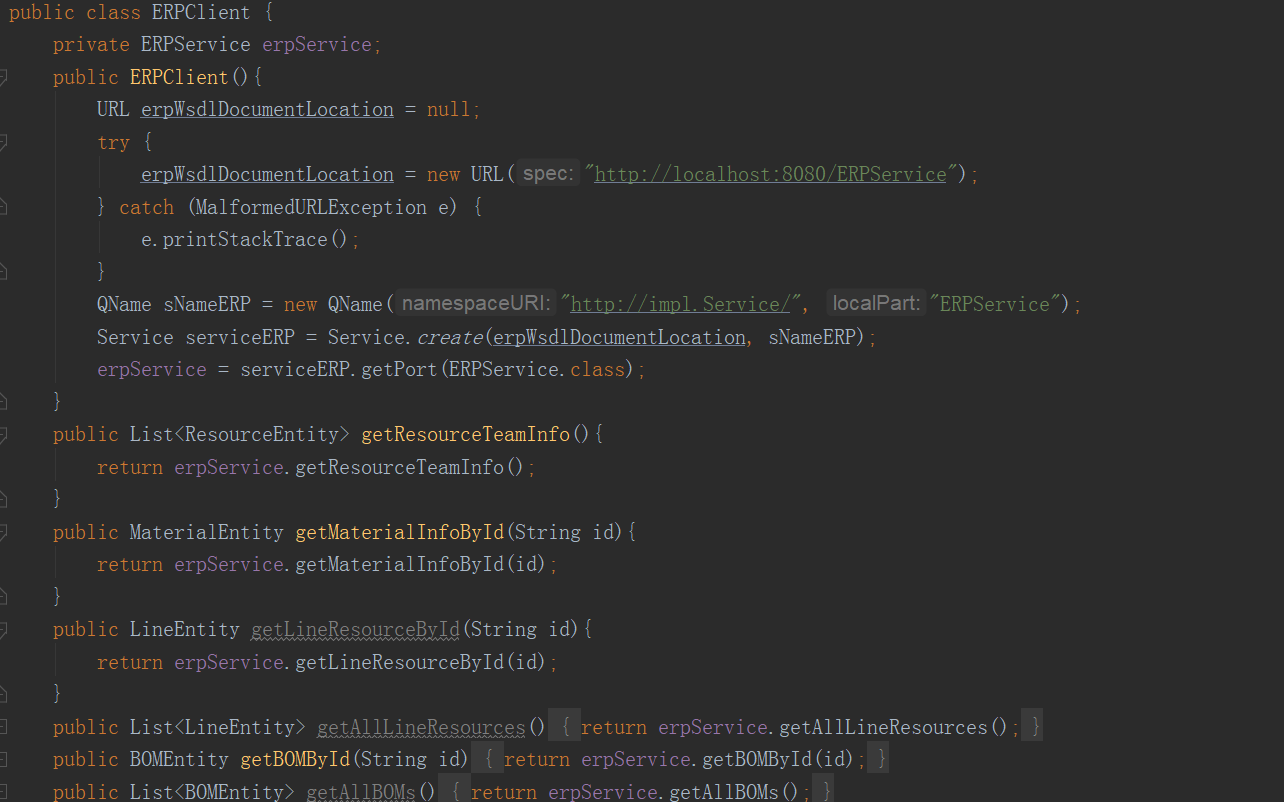


* + 1. 客户端测试



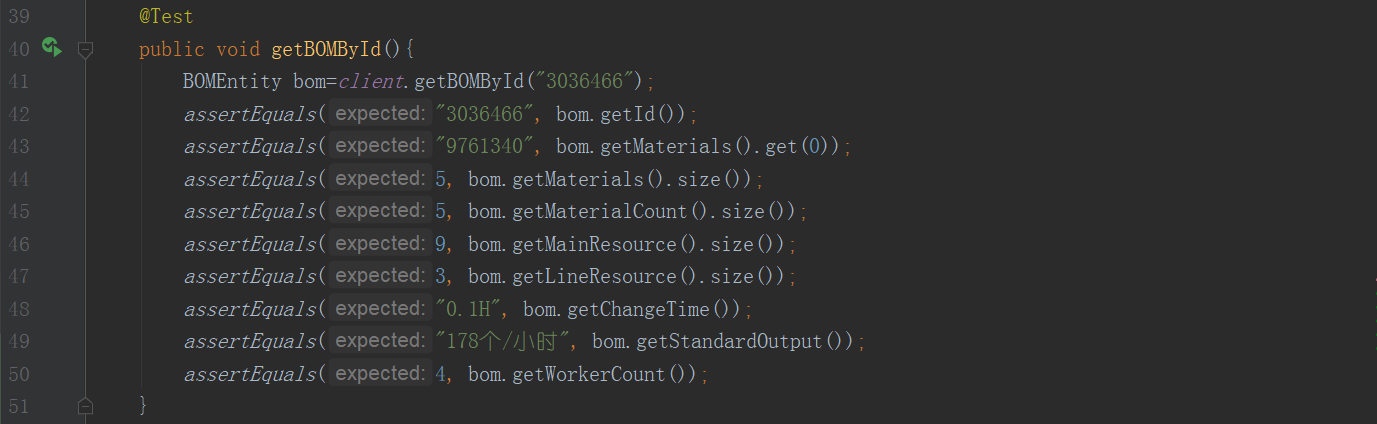
### ERP系统

* + 1. 客户端程序



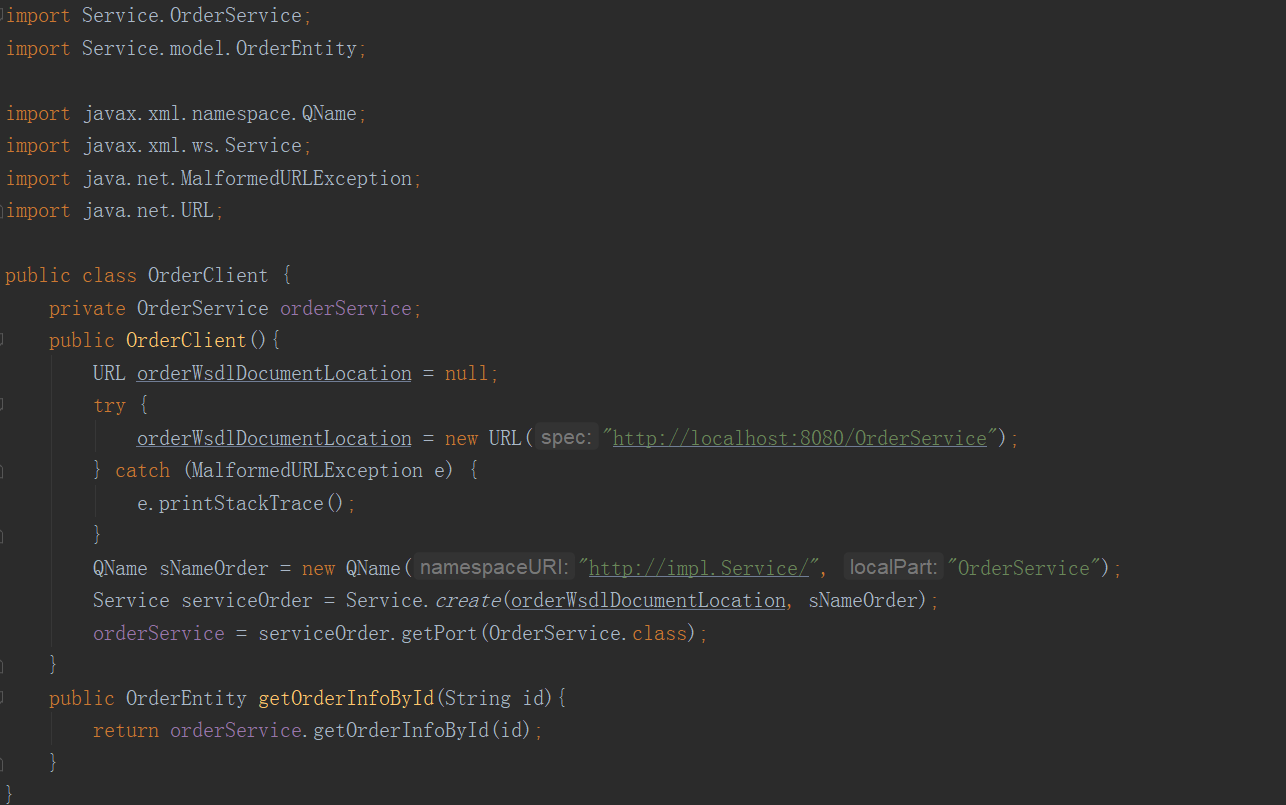
* + 1. 客户端测试





### 订单管理系统

* + 1. 客户端程序



* + 1. 客户端测试

