# ERP项目笔记 Day-11

## CXF框架

1. WebService与CXF简介
   1. WebService

WebService是一种**跨编程语言**和**跨操作系统**平台的**远程调用**技术。

**跨编程语言和跨操作平台** 就是说服务端程序采用java编写，客户端程序则可以采用其他编程语言编写，反之亦然！跨操作系统平台则是指服务端程序和客户端程序可以在不同的操作系统上运行。

**远程调用** 就是一台计算机a上的一个程序可以调用到另外一台计算机b上的一个对象的方法，譬如，银联提供给商场的pos刷卡系统，商场的POS机转账调用的转账方法的代码其实是跑在银行服务器上。再比如，amazon，天气预报系统，淘宝网，校内网，百度等把自己的系统服务以WebService服务的形式暴露出来，让第三方网站和程序可以调用这些服务功能，这样扩展了自己系统的市场占有率。

**服务端**：把公司内部系统的业务方法发布成WebService服务，供远程他人调用

**客户端**：调用别人发布的WebService服务

**常见的远程调动技术**：

1. Socket 套接字 TCP/IP UDP
2. WebService
3. http 调用
4. RMI( 远程方法调用 ) Hessian 框架（二进制RPC协议传输数据）

**WebService 的特点**:

1. 跨平台，跨语言
2. W3C(万维网联盟)制定的标准
3. 可以穿透防火墙（因为 soap 协议是基于 HTTP 协议）

**SOAP 协议**（简单对象访问协议Simple Object Access Protocol）：

WebService通过HTTP协议发送请求和接收结果时，发送的请求内容和结果内容都采用XML格式封装，并增加了一些特定的HTTP消息头，以说明HTTP消息的内容格式，这些特定的HTTP消息头和XML内容格式就是SOAP协议

SOAP协议 = HTTP协议 + XML数据格式

**WSDL**

(Web Services Description Language)就是基于XML的语言，用于描述Web Service及其函数、参数和返回值。它是WebService客户端和服务器端都能理解的标准格式。因为是基于XML的，所以WSDL既是机器可阅读的，又是人可阅读的，这将是一个很大的好处。

一些最新的开发工具既能根据你的Web service生成WSDL文档，又能导入WSDL文档，生成调用相应WebService的代理类代码。

* 1. CXF

CXF，apache 下的 WebService 的开源框架。

Apache CXF = Celtix + Xfire，开始叫 Apache CeltiXfire，后来更名为 Apache CXF了，以下简称为 CXF。Apache CXF 是一个开源的 web Services 框架，CXF 帮助您构建和开发 web Services ，它支持多种协议，比如：SOAP1.1,1,2、XML/HTTP、REST HTTP 或者 CORBA。

灵活的部署：可以运行有 Tomcat,Jboss,weblogic,Jetty(内置)上面。

1. CXF入门Demo
   1. 需求分析

我们这里是实现一个简单的天气预报的案例。注意，我们不是做真正的天气预报，而是一个假的信息。传递城市名称，如果是北京返回雾霾，否则返回晴。

我们开发需要开发服务端和客户端。

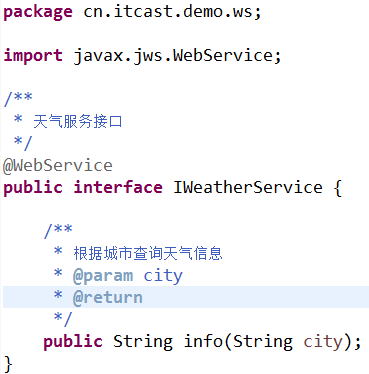
* 1. 服务端开发

1. 创建maven工程，名称为cxfserver，打包方式为war
2. Pom.xml添加依赖以及相关插件



1. 编写服务接口和服务类

接口:

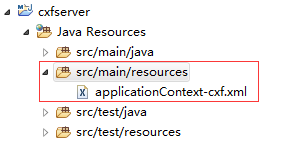


实现类：

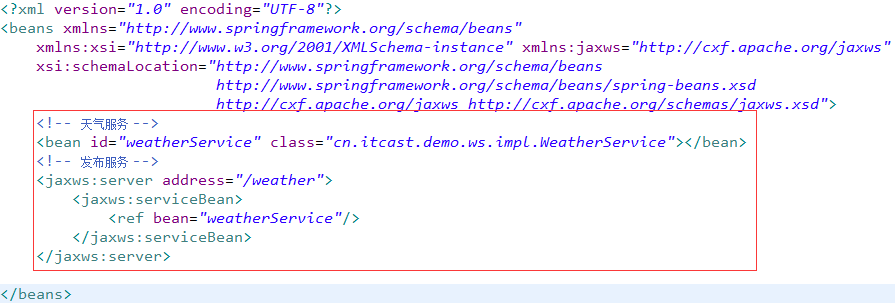


1. 编写applicationContext-cxf.xml文件

在 资料\cxf目录下的applicationContext\_cxf.xml复制到项目的配置目录下



修改applicationContext-cxf.xml，加上以下内容：



1. 配置web.xml文件

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xmlns=*"http://java.sun.com/xml/ns/javaee"*  xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"* version=*"2.5"*>    <!-- 启动spring容器 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:applicationContext\_cxf.xml</param-value>  </context-param>  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <!-- 加载cxf servlet -->  <servlet>  <servlet-name>cxf</servlet-name>  <servlet-class>org.apache.cxf.transport.servlet.CXFServlet</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>cxf</servlet-name>  <url-pattern>/ws/\*</url-pattern>  </servlet-mapping>    </web-app> |

1. 测试运行

浏览器输入：<http://localhost:9090/ws/ws/weather?wsdl>



这个内容就是 WSDL 文档，相当与 webservice 的使用说明书

我们可以看到这里还import了另一个xml



打开它内容如下：

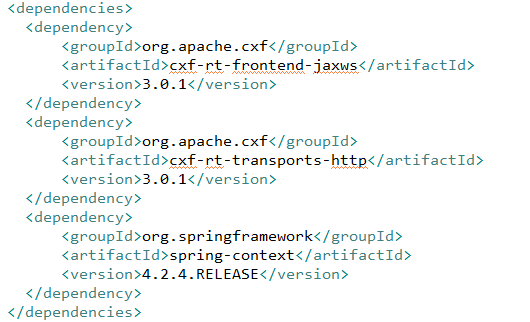


1. WSDL介绍



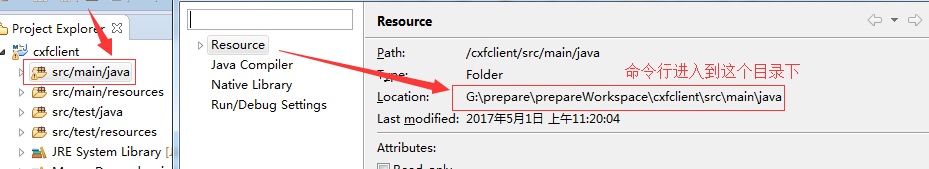
* 1. 客户端开发

1. 创建maven工程(jar) cxfclient
2. 添加依赖和插件



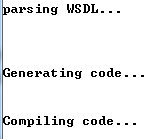
1. 根据 WSDL 生成本地代码

命令行进入项目目录src/main/java目录下：

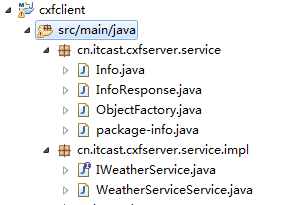


运行:





生成的代码：



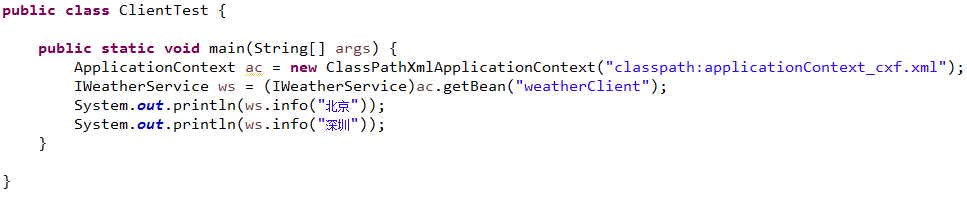
1. 编写 spring(cxf)配置文件applicationContext\_cxf.xml

复制：资料\cxf\applicationContext\_cxf.xml到项目的资源配置目录下

配置applicationContext\_cxf.xml:



1. 编写测试代码调用远程服务



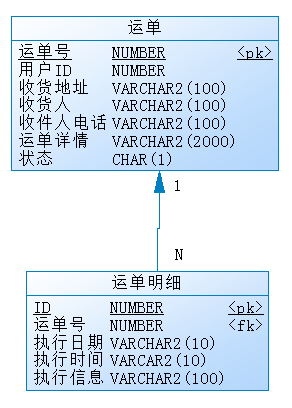
1. 《红日物流BOS系统》开发（服务端）
   1. 需求与思路分析

我们需要开发《红日物流 BOS 系统》，并提供两个功能。

1. 可以通过运单号查询该运单的运单详情
2. 可以在线下单预约

最终由 ERP 系统调用这两个功能

* 1. 数据库设计



运单状态： 0：待发 1：在途 2：结束

数据库：mysql, 数据库名称：redsun

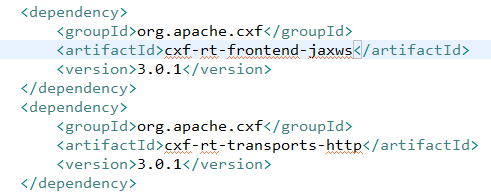
* 1. 系统开发

使用《代码生成器》生成代码。 因为《红日物流 BOS 系统》会与《蓝云 ERP》同时运行，所以需要将《红日物流 BOS 系统》的 pom.xml 中的 8080 端口改为9090

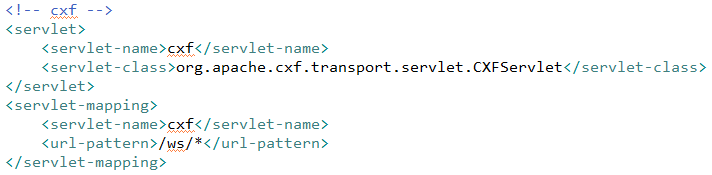
* 1. 整合CXF框架开发WS
     1. SSH2整合CXF框架

1. 引入 CXF 框架

在redsun\_parent中的pom.xml中添加cxf的两个依赖：

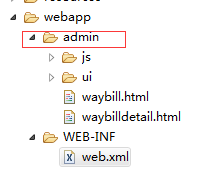


1. 修改 web.xml，添加 CXFServlet

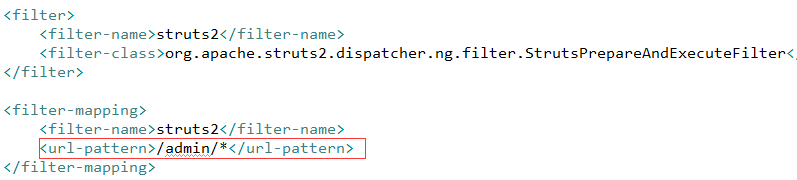


1. 将原有 html 页面和相关资源移动到 admin 目录下

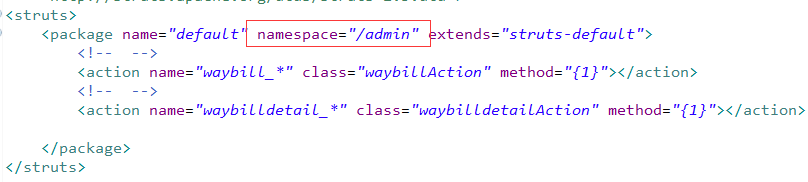
在webapp目录下创建admin目录，并将所有相关资源移动到这个目录下：



1. 修改 web.xml 中 struts2 过滤器的映射目录为/admin/\*



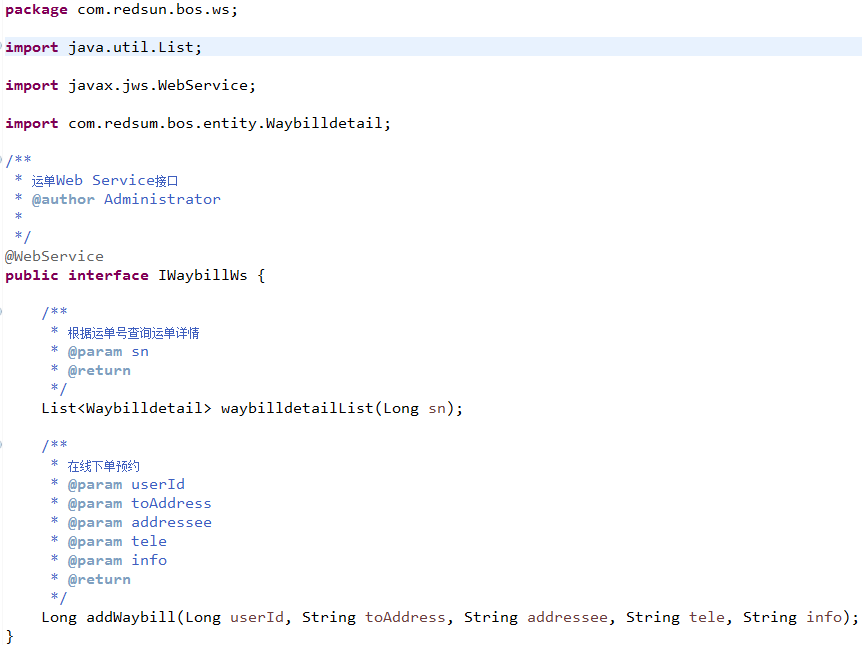
1. 修改 struts.xml



* + 1. 代码实现

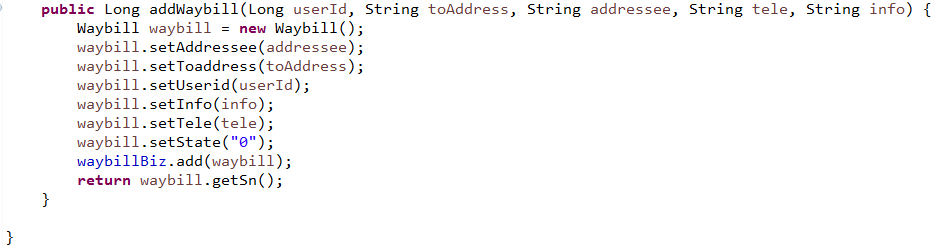
在redsun\_web中创建接口包：com.redsun.bos.ws, com.redsun.bos.ws.impl

1. 创建IWaybillWs接口

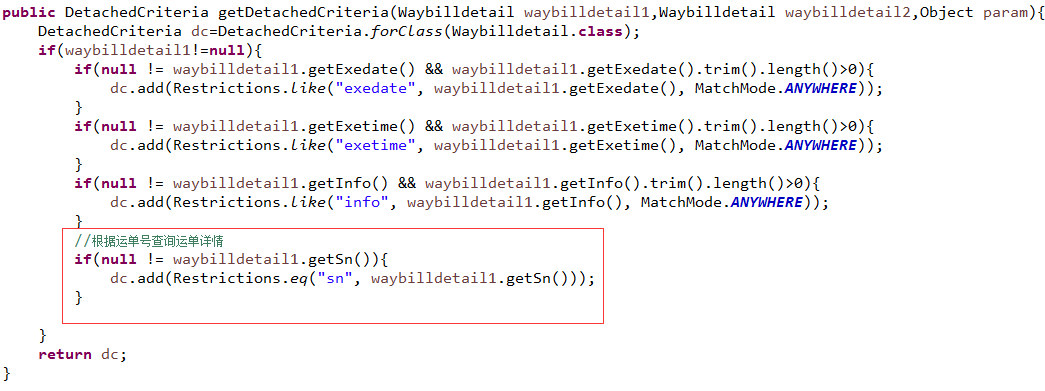


1. 创建WaybillWS实现类





1. 修改WaybilldetailDao，添加查询条件



* + 1. 发布服务

创建applicationContext\_cxf.xml文件：



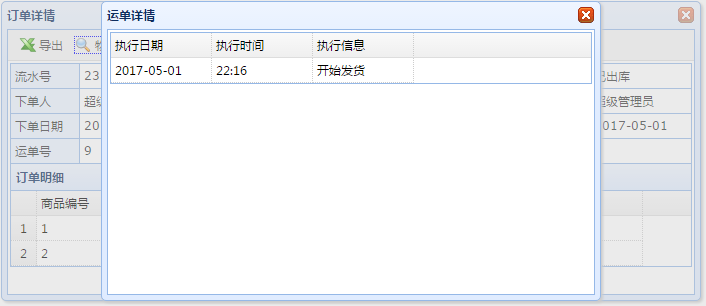
1. ERP调用《红日物流BOS》系统（客户端）
   1. 思路分析

引入 CXF 框架，生成本地代码，调用远程的服务,

功能 1：获取指定运单的运单详情:



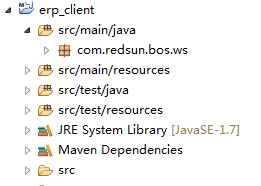
点击物流详情，弹出下面的窗口



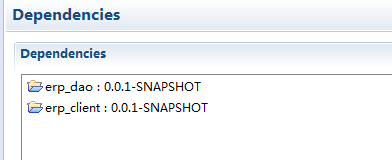
功能 2：出库后的订单自动提交物流信息到远程物流系统

* 1. 创建子模块

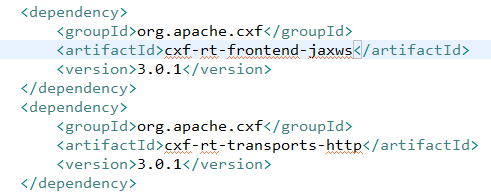
在 ERP 工程中创建子模块 erp\_client 用于存放生成的 webservice 本地代码修改



erp\_biz工程的 pom.xml文件, 配置erp\_biz依赖erp\_client 模块。更新Maven工程。



引入 CXF 框架

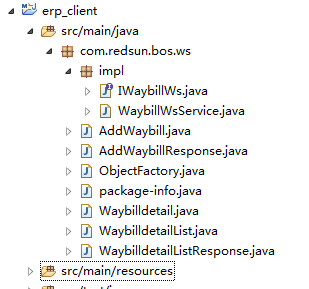


* 1. 客户端实现
     1. 生成代码

进入 erp\_client 工程下的 src\main\java 目录 执行命令

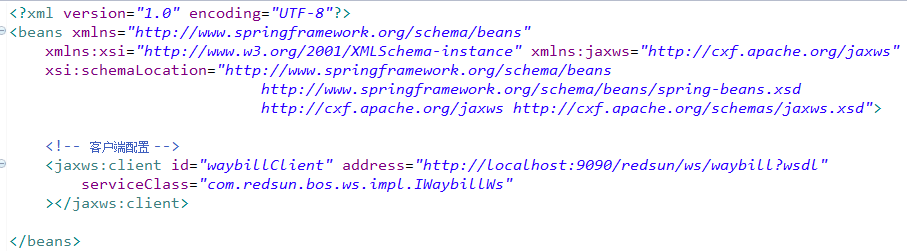


刷新工程后目录如下：



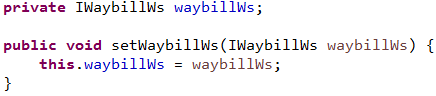
* + 1. 配置客户端

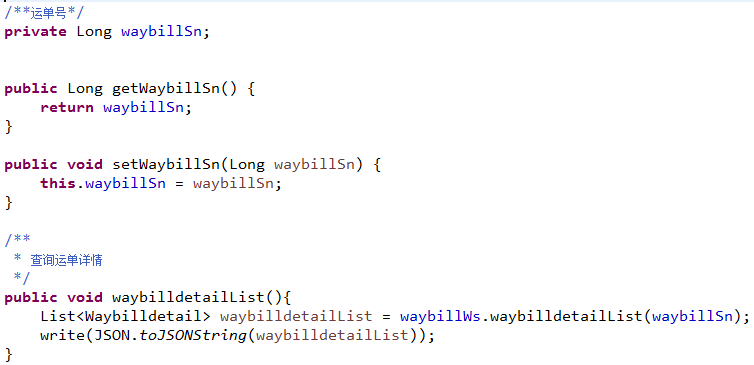
创建applicationContext\_cxf.xml文件：



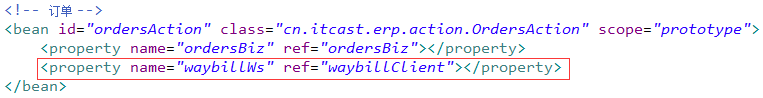
* 1. 根据运单号查询运单详情

1. 在 orders 表 waybillsn (运单号)字段录入测试数据
2. 在 OrdersAction 中调用 web 服务实现根据运单号查询运单详情的功能



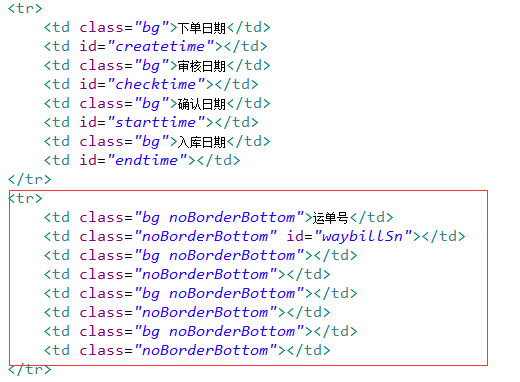


1. 修改 applicationContext\_action.xml

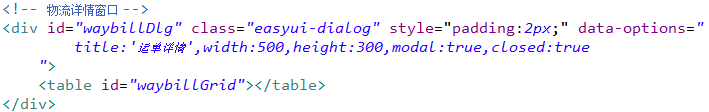


1. 编写前端代码完成运单查询

订单详情窗口，添加一行，用于显示运单号：

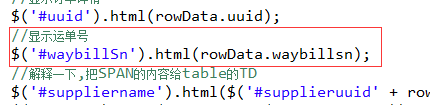


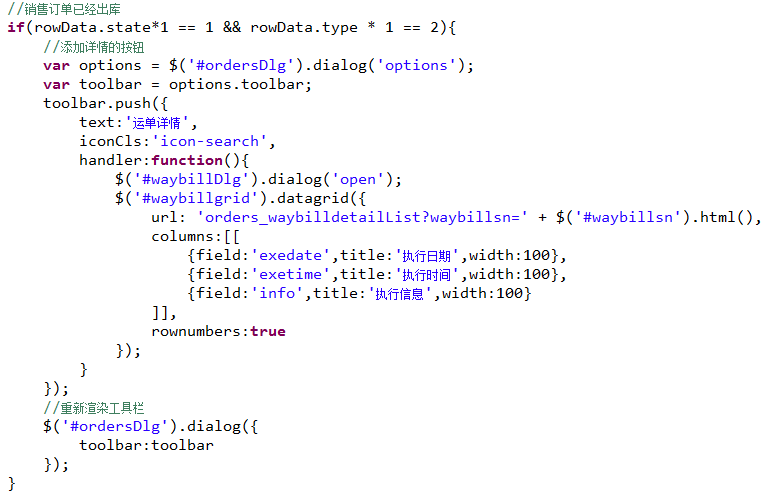
添加物流详情窗口：



修改orders.js在双击打开订单详情中添加

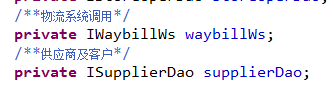
运单号的显示：



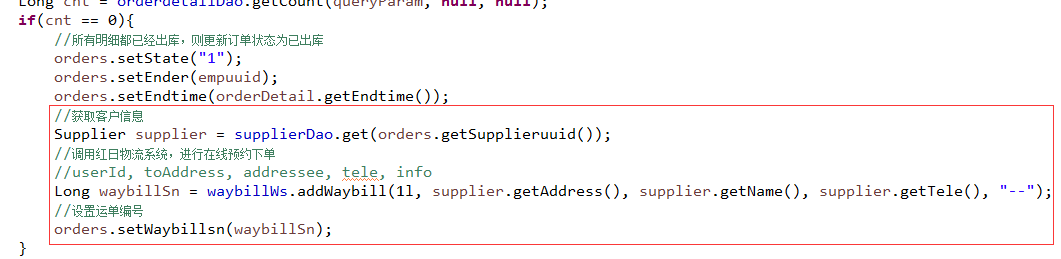


* 1. 销售出库自动提交物流订单

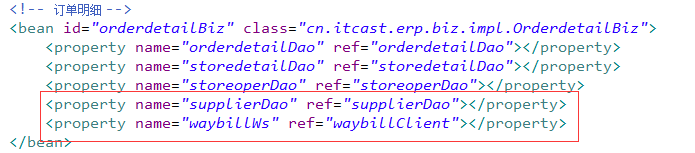
1. 在 OrderDetailBiz 中添加属性，同时生成setter



1. 修改 OrderDetailBiz 的 doOutStore 方法 ,在 count==0 的条件中添加以下代码



1. 修改 applicationContext\_biz.xml



1. 测试：进行销售出库，如果出库后，物流系统的 waybill 表中能够添加记录表示成功。