ERP 系统项目笔记

第 1 章

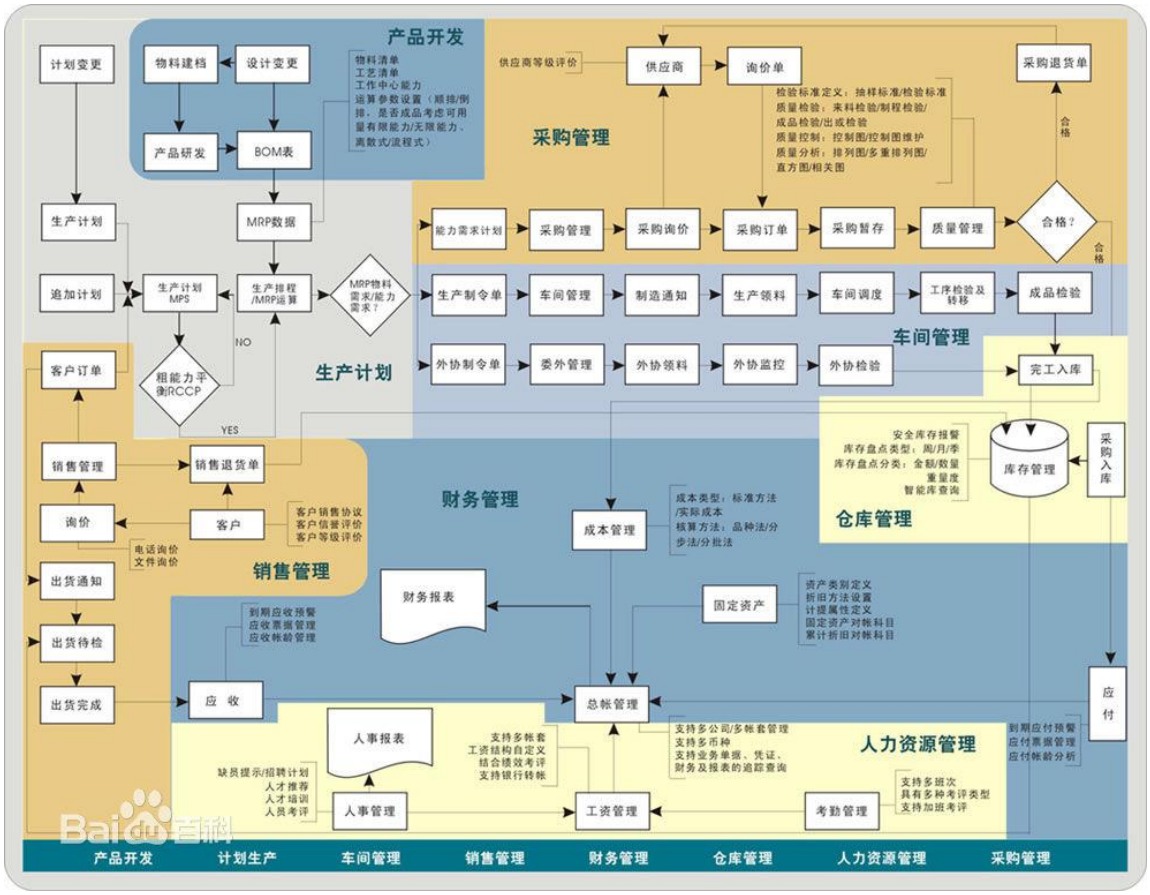
软件设计与环境搭建

传智播客

一、ERP 简介

（一）什么是 ERP

ERP 系统是企业资源计划(Enterprise Resource Planning)的简称，是指建立在信息技术基础上，以系统化的管理思想，为企业决策层及员工提供决策运行手段的管理平台。ERP 系统集信息技术与先进管理思想于一身，成为现代企业的运行模式，反映时代对企业合理调配资源，最大化地创造社会财富的要求，成为企业在信息时代生存、发展的基石。它对于改善企业业务流程、提高企业核心竞争力具有显著作用。ERP 行业人才稀缺成为 SAP 发展的制约因素之一，鉴于此，国内的 ERP 培训行业逐渐开始发展。



我们课程中的 ERP 系统定位的行业为商贸企业

这里我们需要熟悉以下基本概念---

供应商：你的卖家，提供给你原材料的商家

客户：　你的买家，你要销售的对象

采购：　买东西

销售：　卖东西

订单：　采购订单　销售订单

（二）项目演示

登陆页面 login.html　　用户名密码均为 admin

二、软件项目常识

软件开发流程：

需求分析--->软件设计--->编码---->测试---->上线---->系统维护

（一）需求分析与需求规格说明书

所谓"需求分析"，是指对要解决的问题进行详细的分析，弄清楚问题的要求，包括需要输入什么数据，要得到什么结果，最后应输出什么。可以说，在软件工程当中的“需求分析”就是确定要计算机“做什么”，要达到什么样的效果。可以说需求分析是做系统之前必做的。

需求规格说明书的编制是为了使用户和软件开发者双方对该软件的初始规定有一个共同的理解， 使之成为整个开发工作的基础。包含硬件、功能、性能、输入输出、接口需求、警示信息、保密安全、数据与数据库、文档和法规的要求。

（ 请大家课后阅读《蓝云 ERP 需求规格说明书》了解需求 ）

（二）软件设计与软件设计文档

软件设计是从软件需求规格说明书出发，根据需求分析阶段确定的功能设计软件系统的整体结构、划分功能模块、确定每个模块的实现算法以及编写具体的代码，形成软件的具体设计方案。

软件设计文档分为《概要设计》和《详细设计》

概要设计：系统模块划分、网络拓扑图、用例图

详细设计：类图、时序图、类清单、方法清单、接口清单、表结构文档

（三）UML 语言

统一建模语言（UML，UnifiedModelingLanguage）是面向对象软件的标准化建模语言。UML 因其简单、统一的特点，而且能表达软件设计中的动态和静态信息，目前已成为可视化建模语言的工业标准。

UML 从考虑系统的不同角度出发，定义了用例图、**类图**、对象图、状态图、**活动图**、序列图、协作图、构件图、部署图等 9 种图。这些图从不同的侧面对系统进行描述。系统模型将这些不同的侧面综合成一致的整体，便于系统的分析和构造。尽管 UML 和其它开发工具还会设计出许多派生的视图，但上述这些图和其它辅助性的文档是软件开发人员所见的最基本的构造。

（四）设计工具 PowerDesigner(PD)

PowerDesigner(PD)最初由 Xiao-Yun Wang（王晓昀）在 SDP Technologies 公

司开发完成。是 Sybase 的企业建模和设计解决方案，采用模型驱动方法，将业务与 IT 结合起来，可帮助部署有效的企业体系架构，并为研发生命周期管理提供强大的分析与设计技术。PowerDesigner 独具匠心地将多种标准数据建模技术（UML、业务流程建模以及市场领先的数据建模）集成一体，并与 .NET、WorkSpace、PowerBuilder、Java™、Eclipse 等主流开发平台集成起来，从而为传统的软件开发周期管理提供业务分析和规范的数据库设计解决方案。

演示：使用 PD 完成“活动图”的绘制

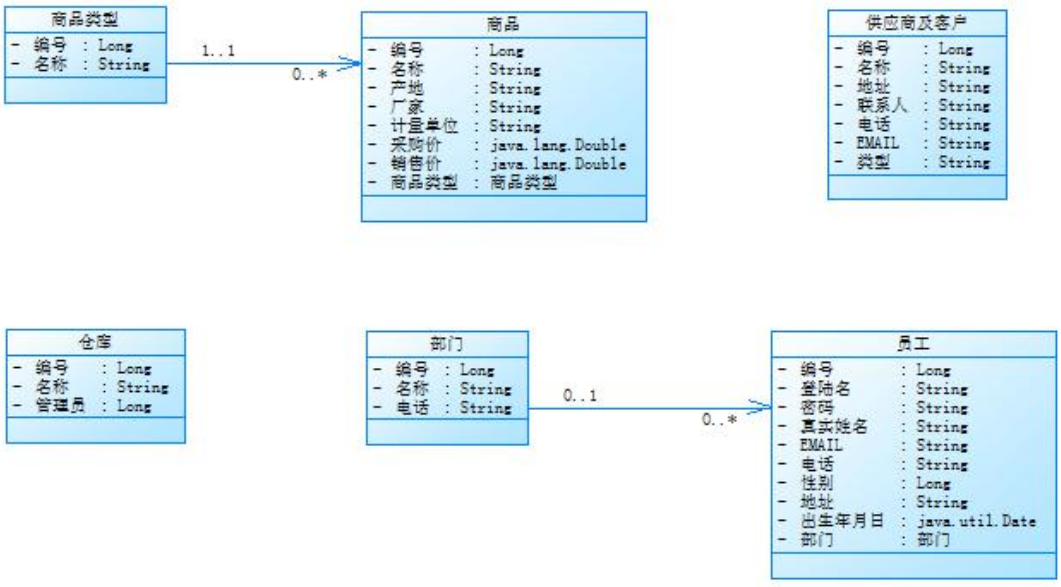
活动图就是描述业务流程的图

三、软件设计

（一）类图设计（实体类）

所谓类图是用于描述类的名称属性方法以及类与类之间关系的 UML 图。

演示：使用 PD 完成“类图”的绘制



（二）数据库设计

1.部门表（DEP）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
name	varchar(30)	必填	名称
tele	varchar(30)		电话

2.员工表（EMP）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
username	varchar(15)	必填	登陆名
pwd	varchar(32)		密码
name	varchar(30)		真实姓名
gender	number		性别
email	varchar(255)		电子邮箱
tele	varchar(30)		电话
address	varchar(255)		地址
birthday	date		出生年月日
depuuid	number		部门编号

3.商品类型表（GOODSTYPE）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
name	varchar(30)	必填	名称

4.商品表（GOODS）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
name	varchar(30)	必填	商品名称
origin	varchar(30)		产地
producer	varchar(30)		厂商
unit	varchar(30)		计量单位
inprice	number(8,2)		进货价
outprice	number(8,2)		销售价
goodstypeuuid	number		商品类型 ID

5.仓库表（STORE）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
name	varchar(30)	必填	名称
empuuid	number		库管员 ID

6.供应商及客户表（SUPPLIER）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
name	varchar(30)	必填	名称
address	varchar(100)		地址
contact	varchar(30)		联系人
tele	varchar(30)		电话
email	varchar(100)		电子邮箱
type	number		类型

7.订单表（ORDERS）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
createtime	date		创建日期
checktime	date		审核日期
starttime	date		确认日期
endtime	date		结束日期
type	char(1)		类型
creater	number		创建人
checker	number		审核人
starter	number		确认人
ender	number		结束人

supplieruuid	number		供应商 ID
totalmoney	number		总金额
state	char(1)		状态

8.订单明细表（ORDERDETAIL）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
goodsuuid	number		商品 ID
goodsname	varchar(50)		商品名称
price	number(10,2)		价格
num	number		数量
money	number(10,2)		金额
endtime	date		结束日期
ender	number		结束人
storeuuid	number		仓库 ID
state	char(1)		状态
ordersuuid	number		订单 ID

9.商品仓库库存表（STOREDETAIL）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
storeuuid	number		仓库 ID

goodsuuid	number		商品 ID
num	date		库存数量

10.商品仓库库存操作记录（STOREOPER）

字段名	类型(位数)	是否必填	说明
uuid	number	必填	主键
empuuid	number		员工 ID
opertime	date		操作时间
storeuuid	number		仓库 ID
goodsuuid	number		商品 ID
num	number		操作数量
type	char(1)		类型

（三）命名规范

1.系统分层及包命名规范

（1）实体类层

cn.itcast.erp.entity

（2）数据访问层

接口　cn.itcast.erp.dao

类　　cn.itcast.erp.dao.impl

（3）业务逻辑层

接口　cn.itcast.erp.biz

类　　cn.itcast.erp.biz.impl

（4）action 层

cn.itcast.erp.action

2.类与接口命名规范

（1）实体类　表名称，首字母大写　dep　Dep

（2）数据访问层接口　IDepDao

（3）数据访问类　DepDao

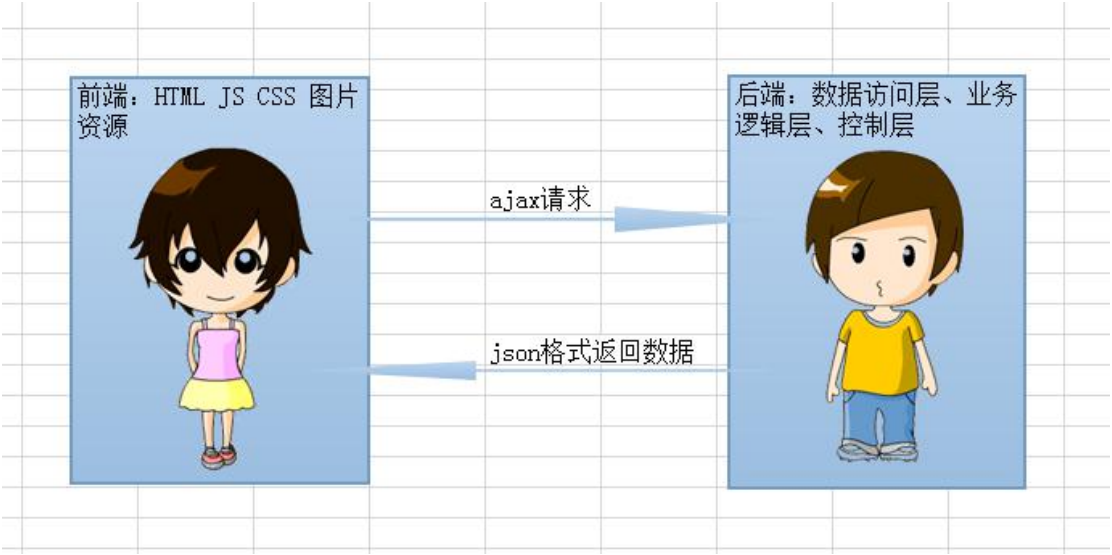
（4）业务逻辑层接口　IDepBiz

（5）业务逻辑类　DepBiz

（6）action 类　DepAction

四、系统架构- SSH2+ easyui

（一）流行的前后端开发



前端：包括 HTML 、JS 、CSS 、图片 ,作用是展示数据和采集数据

后端：数据访问层、业务逻辑层、控制层（action），作用处理业务逻辑、进行数据存储和读取。

前后端开发的优点：

（1）分工明确，有利于提高开发速度

（2）项目更换开发语言，工作量较少。

（3）增强用户的体验。

前端：所关心的问题是如何展示数据和如何采集用户输入的数据

后端：数据的逻辑处理

（二）前端- EasyUI

对于企业及的开发项目，或是网站的后台部分，我们可以使用前端框架来实现，现在目前比较主流的前端框架有 easyUI　BUI　miniUI 等。其中 easyUI 在企业开发中市场份额最大，应用最广泛，使用起来也比较容易上手。我们在本次项目中就采用 easyUI 作为前端框架。

简介：easyui 是一种基于 jQuery 的用户界面插件集合。使用 easyui 你不需要写很多代码，你只需要通过编写一些简单 HTML 标记，就可以定义用户界面。

easyui 是个完美支持 HTML5 网页的完整框架。节省您网页开发的时间和规模，很简单但功能强大的。

我们在初学 easyUI 时，可以用到哪学到哪，我们在项目进展过程中或以查文档的方式，循序渐进地掌握 easyUI。

推荐学习网址：http://www.jeasyui.net

（三）后端- SSH2 集成框架

SSH2 即 struts2 +spring +hibernate

Struts2： Struts 2 是 Struts 的下一代产品，是在 struts 1 和 WebWork 的技术基础上进行了合并的全新的 Struts 2 框架。其全新的 Struts 2 的体系结构与 Struts 1 的体系结构差别巨大。Struts 2 以 WebWork 为核心，采用拦截器的机制来处理用户的请求，这样的设计也使得业务逻辑控制器能够与 ServletAPI 完全脱离开，所以 Struts 2 可以理解为 WebWork 的更新产品。

Spring： Spring 是一个开源框架，Spring 是于 2003 年兴起的一个轻量级的 Java 开发框架，由 Rod Johnson 在其著作 Expert One-On-One J2EE Development and Design 中阐述的部分理念和原型衍生而来。它是为了解决企业应用开发的复杂性而创建的。

（配置方式）事务处理--（声明式事务）

Hibernate: Hibernate 是一个开放源代码的对象关系映射框架，它对 JDBC 进行了非常轻量级的对象封装，使得 Java 程序员可以随心所欲的使用对象编程思维来操纵数据库。

五、数据库-ORACLE

（一）表空间的创建

```
create tablespace 表空间名称
```

```
datafile 'c:\数据文件名称.dbf'
size 100m
autoextend on
next 10m
```

（二）用户与表的创建

表空间里建用户 数据库对象都是建立在用户下的。

1、创建用户并赋权

创建用户 erpuser ,用来存储我们 ERP 系统的表

```
创建用户：
create user erpuser
identified by itcast
default tablespace 表空间名称
赋权：
grant dba to erpuser
```

2、创建表

登陆 erpuser ,创建表

六、开发工具及环境准备

开发工具采用 Eclipse (版本 Mars.2)

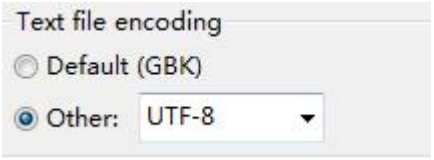
JDK1.7.0.72

应用服务器：TOMCAT 7.0.52

Maven 3.3.9

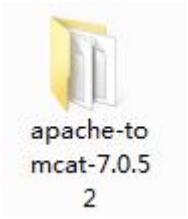
（一）安装 JDK1.7 及 Eclipse 初始环境配置

设定字符集为 UTF-8

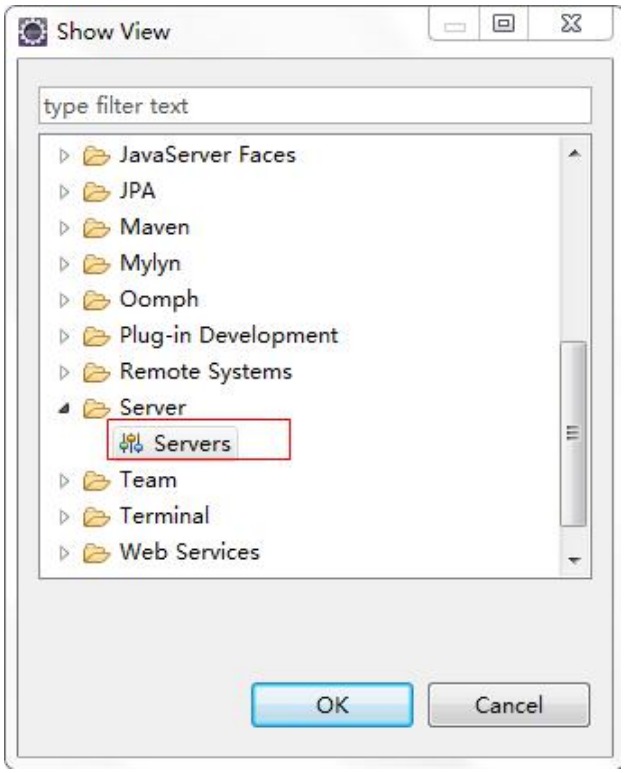


（二）解压 Tomcat 并在 Eclipse 中配置 Tomcat

（1）将 TOMCAT 解压到 D 盘根目录



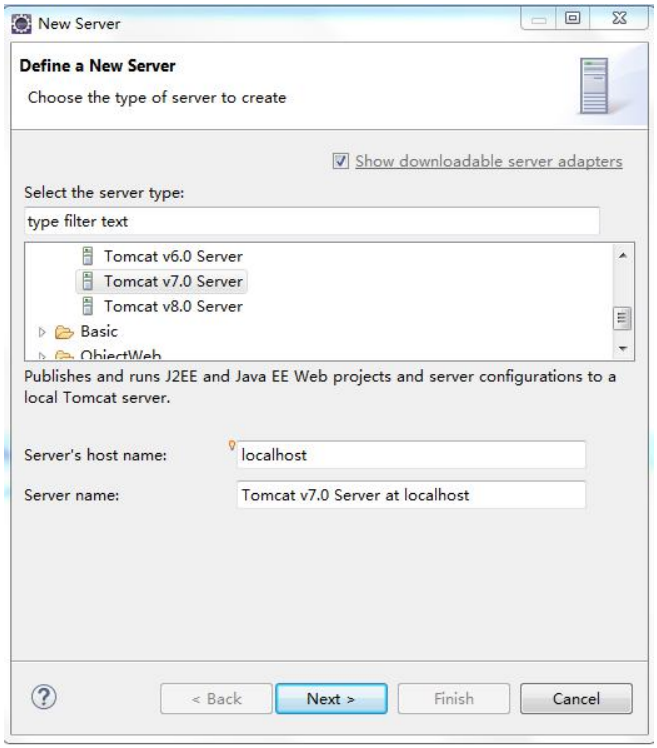
（2）打开菜单 window-- showView other



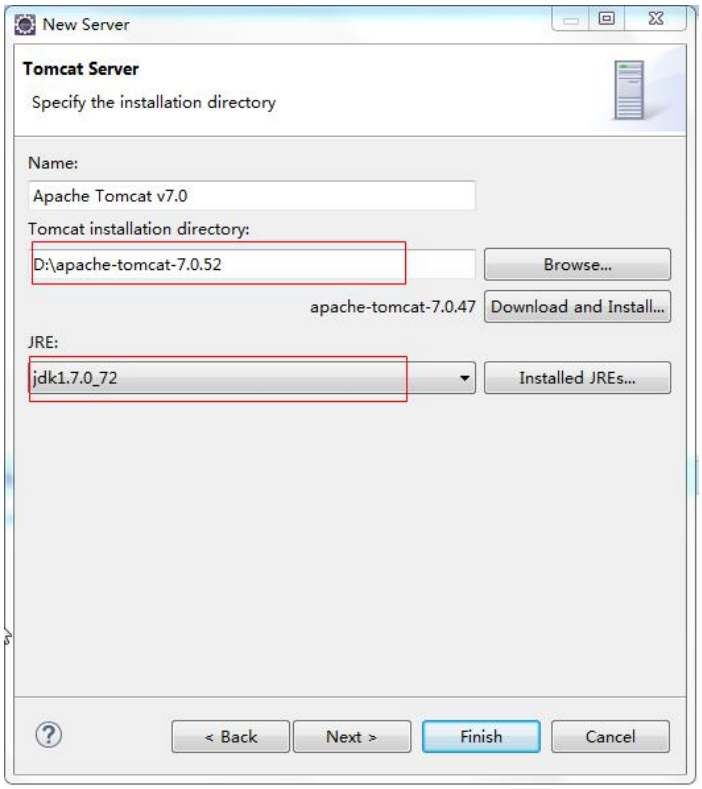
选择 servers OK

(3) 配置 TOMCAT 目录

点击上边的连接

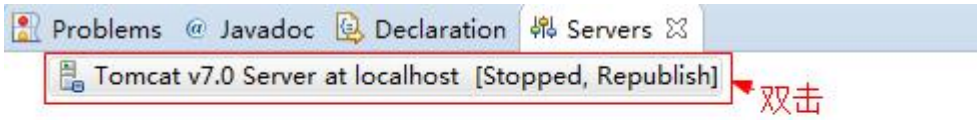


选择 APACH 下的 TOMCAT7.0

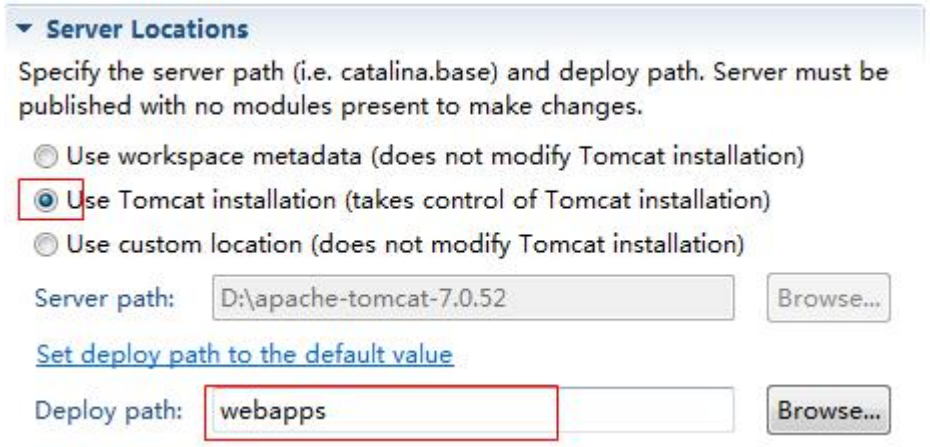


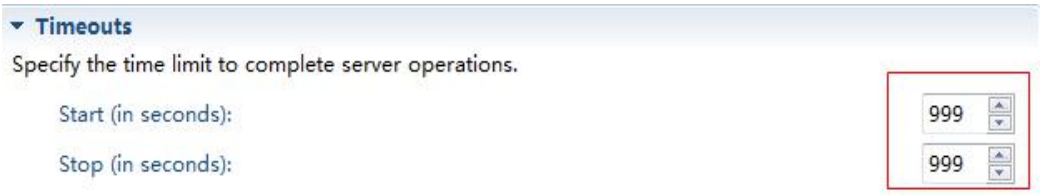
指定目录后，选择 JRE ,然后 Finish

（4）修改 TOMCAT 配置



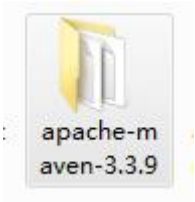
修改以下配置





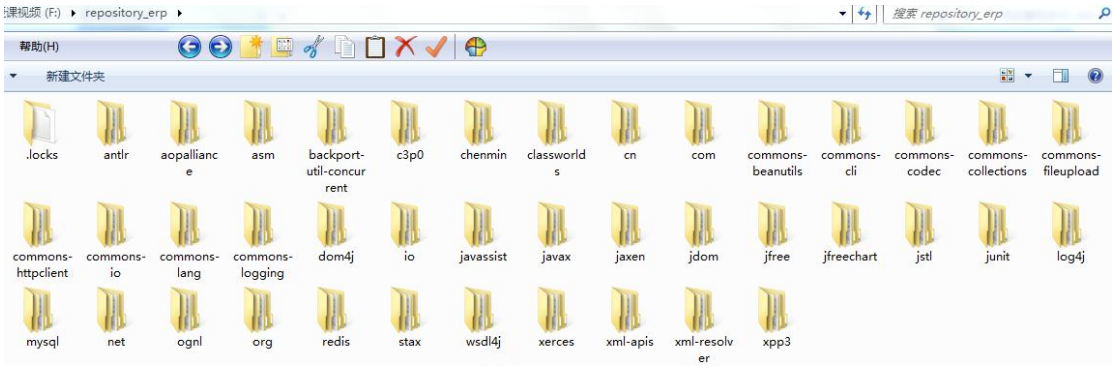
（三）搭建 Maven 环境

（1）解压 Maven 3.3.9 到 D 盘根目录



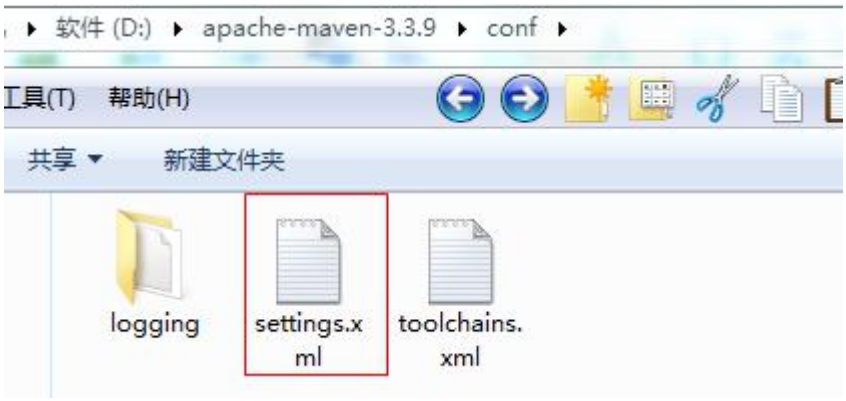
（2）配置环境变量

（3）解压本地仓库



解压本地仓库到 F 盘

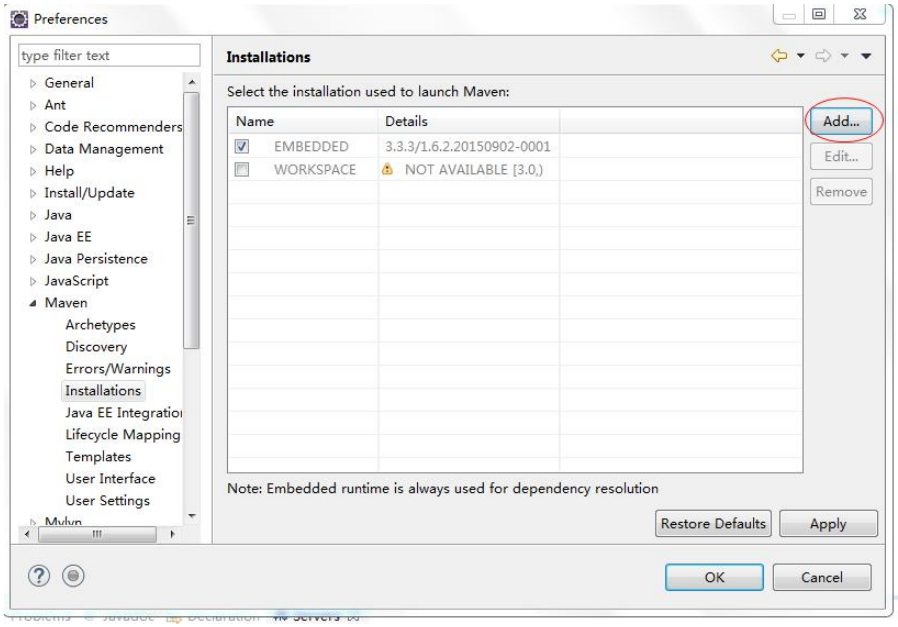
配置 maven 下的 conf 下的 settings.xml

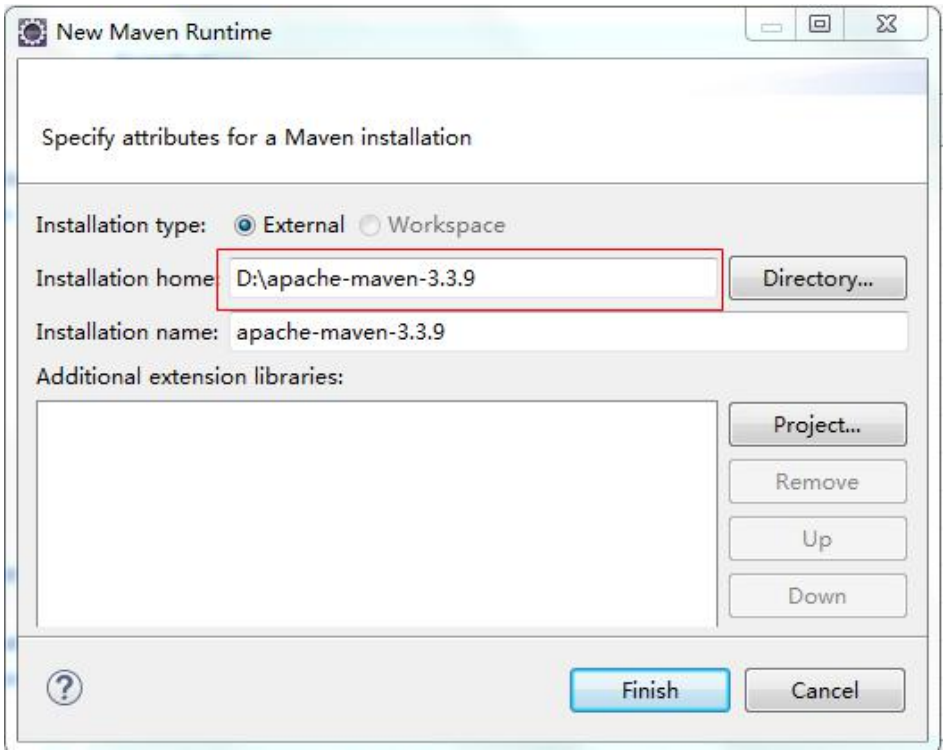


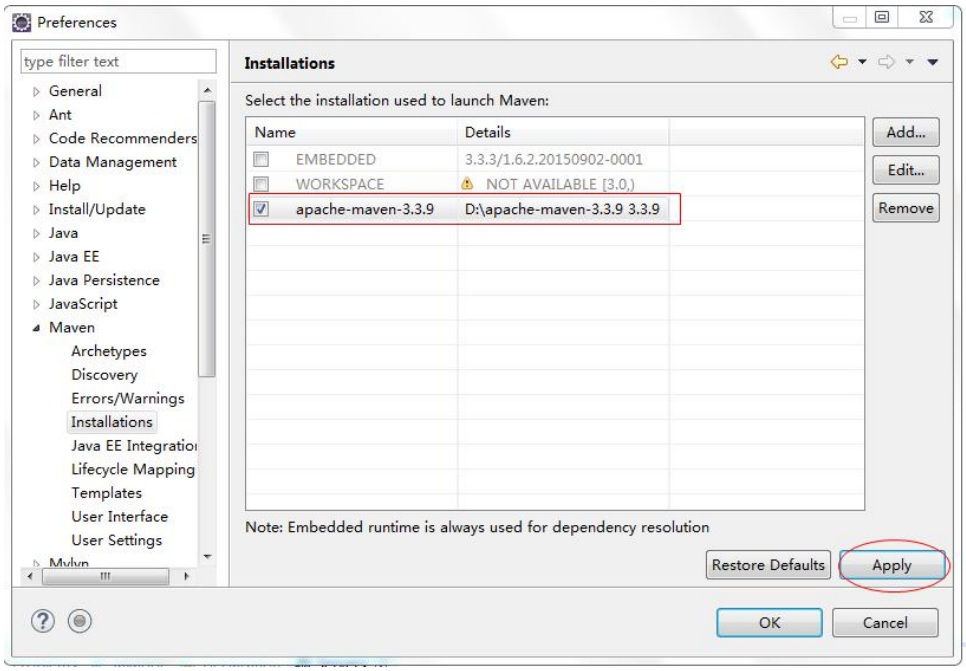
配置本地仓库目录

```
<localRepository>F:\repository_erp</localRepository>
```

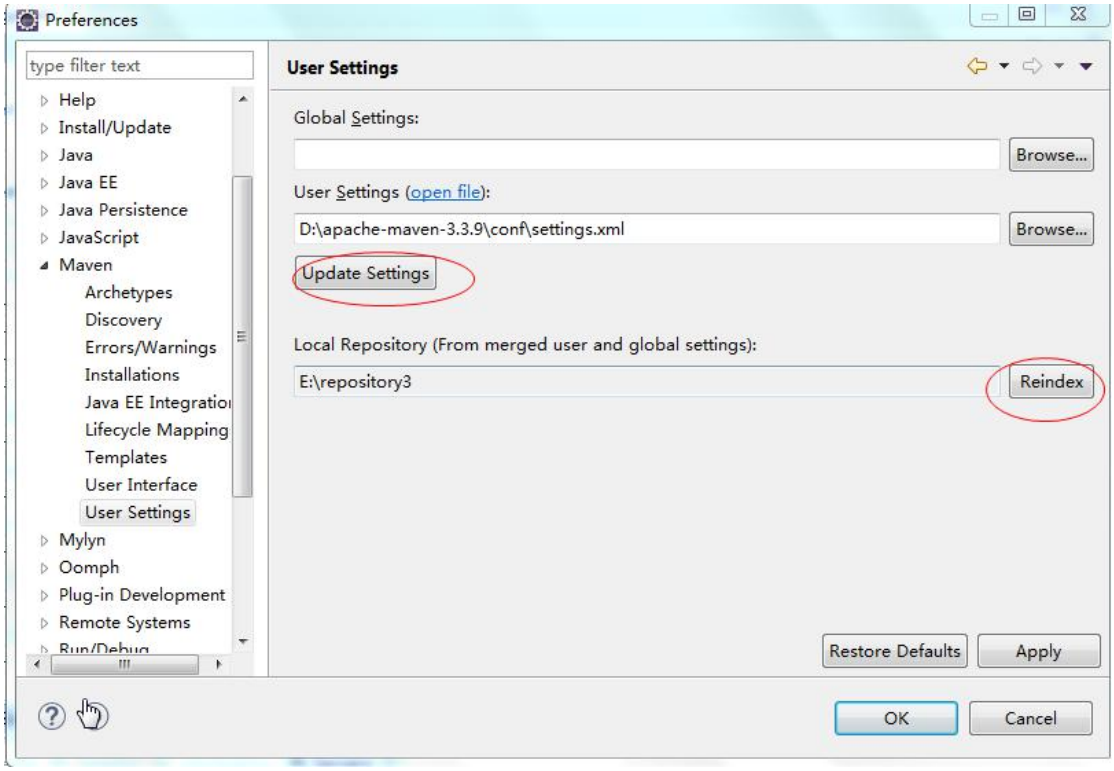
（4）在 Eclipse 中配置 Maven


Preferences
type filter text
General
Ant
Code Recommenders
Data Management
Help
Install/Update
Java
Java EE
Java Persistence
JavaScript
Maven
Archetypes
Discovery
Errors/Warnings
Installations
Java EE Integratio
Lifecycle Mapping
Templates
User Interface
User Settings
Installations
Select the installation used to launch Maven:
Name
Details
EMBEDDED
3.3.3/1.6.2.20150902-0001
WORKSPACE
NOT AVAILABLE [3.0,)
Add...
Edit...
Remove
Note: Embedded runtime is always used for dependency resolution
Restore Defaults
Apply
OK
Cancel

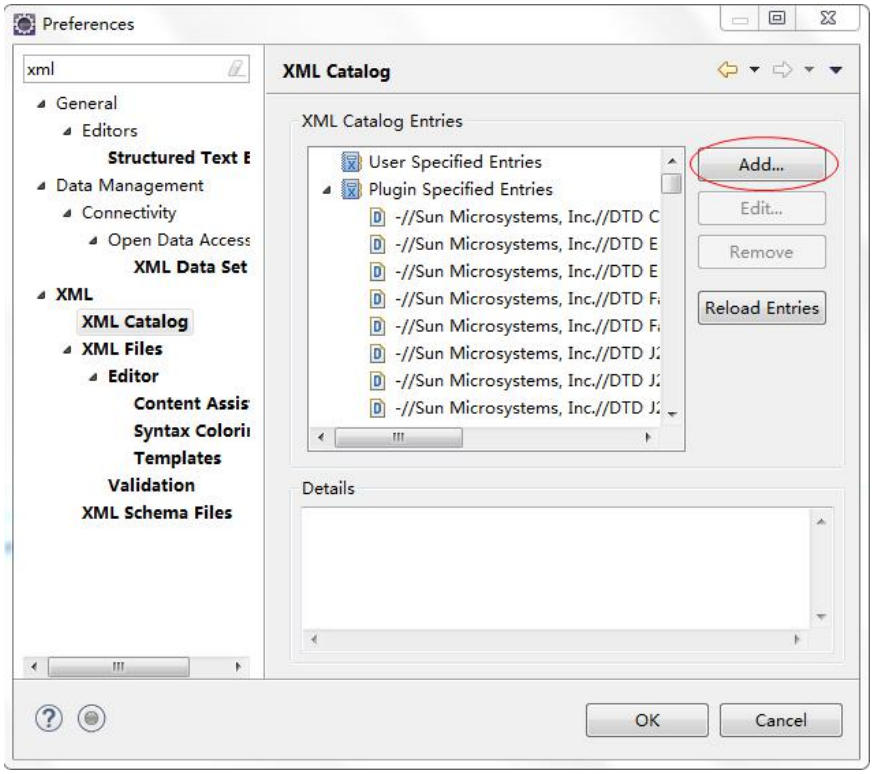

New Maven Runtime
Specify attributes for a Maven installation
Installation type:
External
Workspace
Installation home:
D:\apache-maven-3.3.9
Directory...
Installation name:
apache-maven-3.3.9
Additional extension libraries:
Project...
Remove
Up
Down
Finish
Cancel



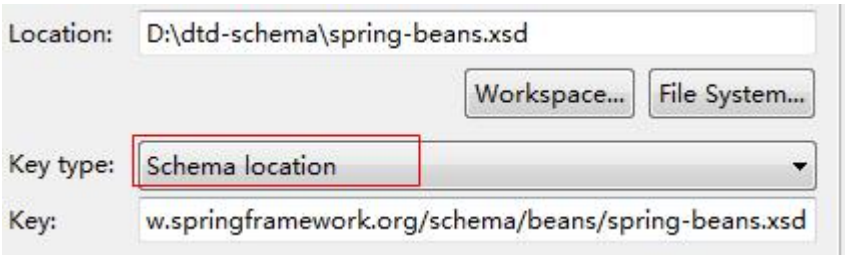
配置 user settings



（四）配置离线约束

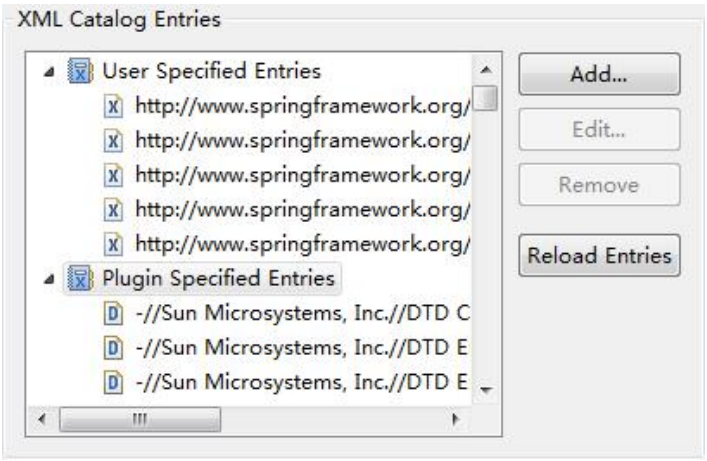


（1）配置 XSD 约束

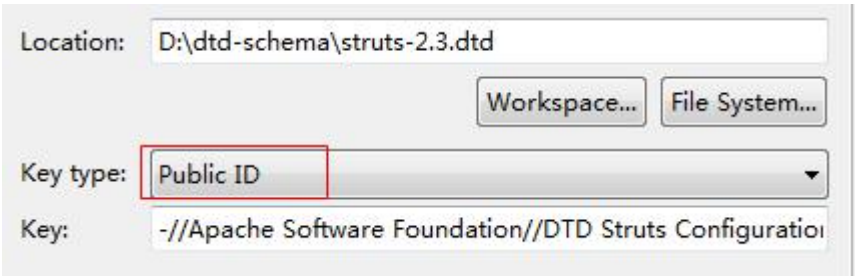


将下面的 5 个 XSD 都配置完成

http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context/spring-context.xsd
http://www.springframework.org/schema/tx/spring-tx.xsd
http://www.springframework.org/schema/aop/spring-aop.xsd
http://www.springframework.org/schema/tool/spring-tool.xsd



（2）DTD 配置



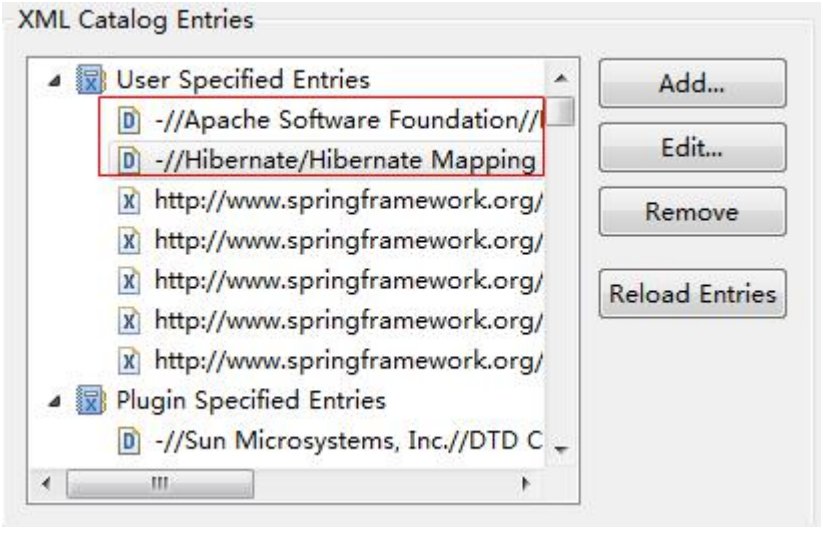
struts-2.3.dtd 的 PUBLICID:

-//Apache Software Foundation//DTD Struts Configuration 2.3//EN

---

hibernate-mapping-3.0.dtd 的 PUBLICID:

-//Hibernate/Hibernate Mapping DTD 3.0//EN

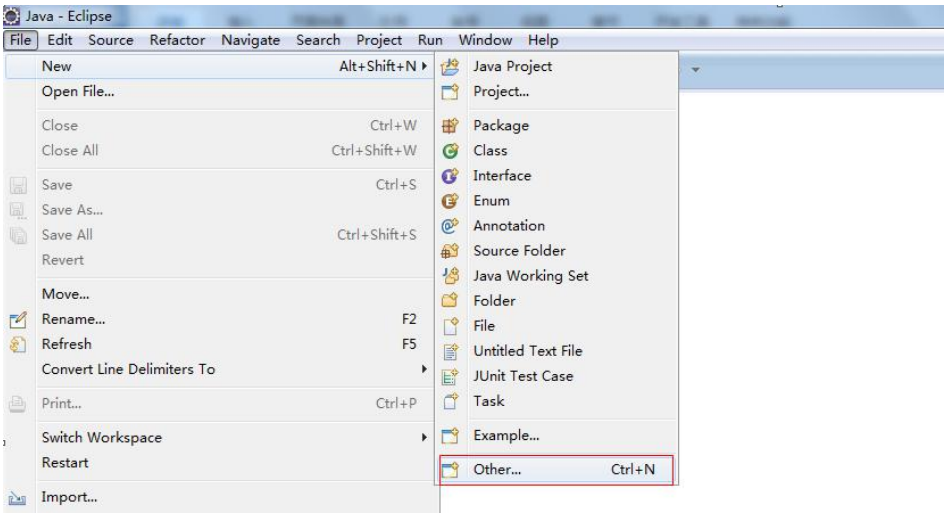


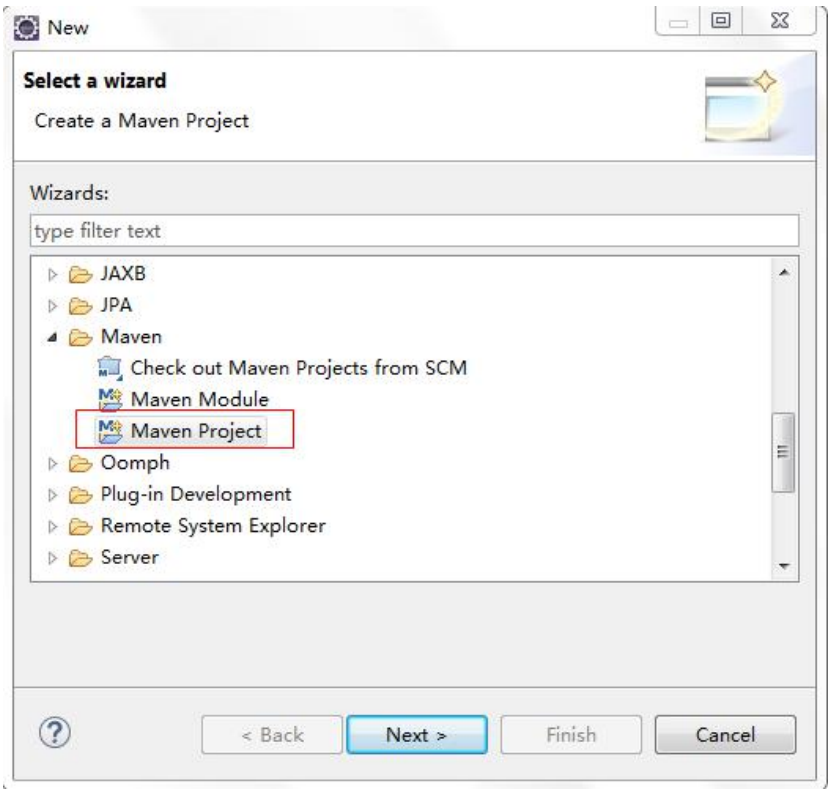
配置完成

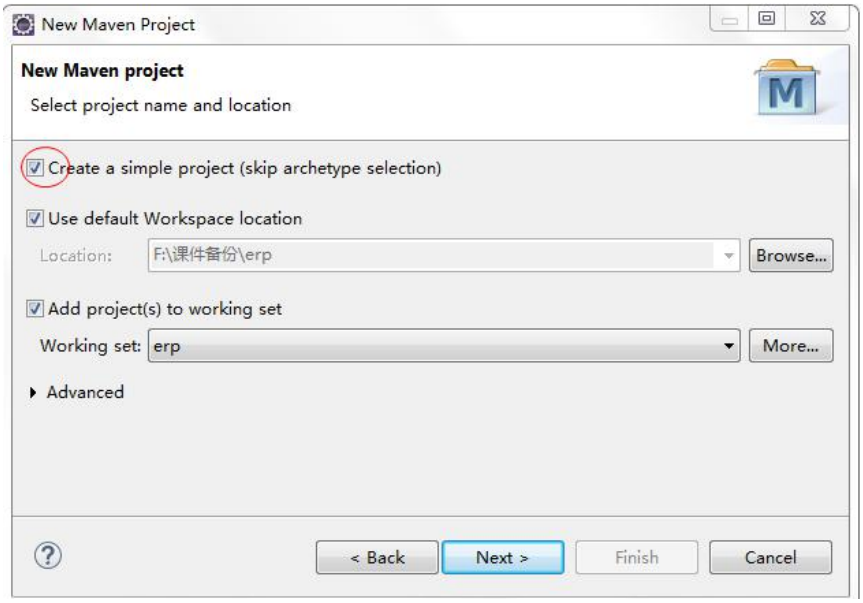
七、SSH2 框架搭建

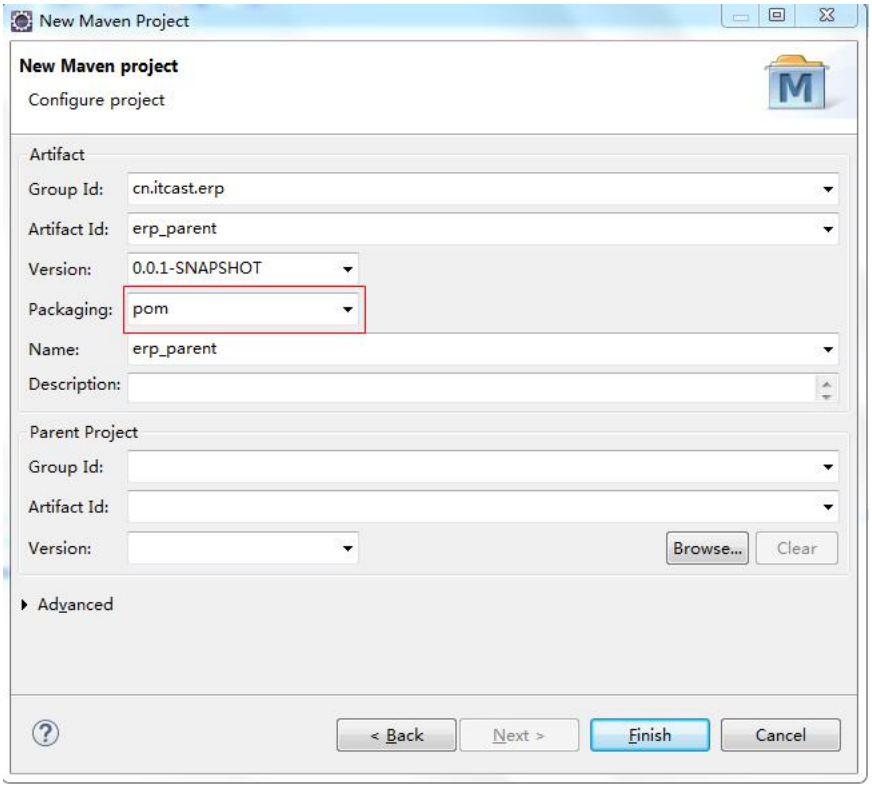
（一）创建 Maven 父工程

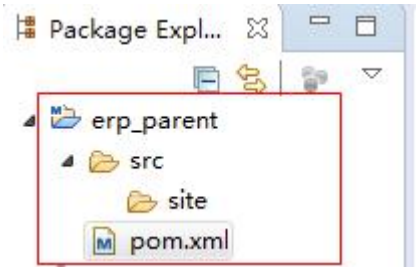
（1）创建 Maven 父工程(erp_parent)







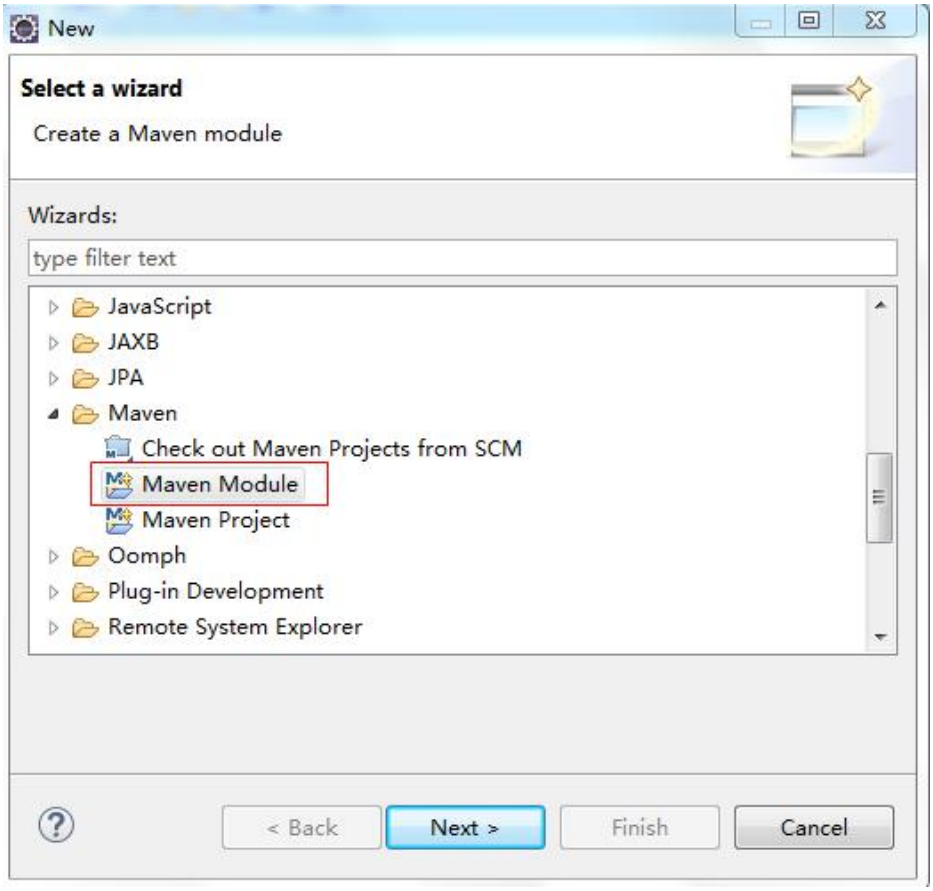


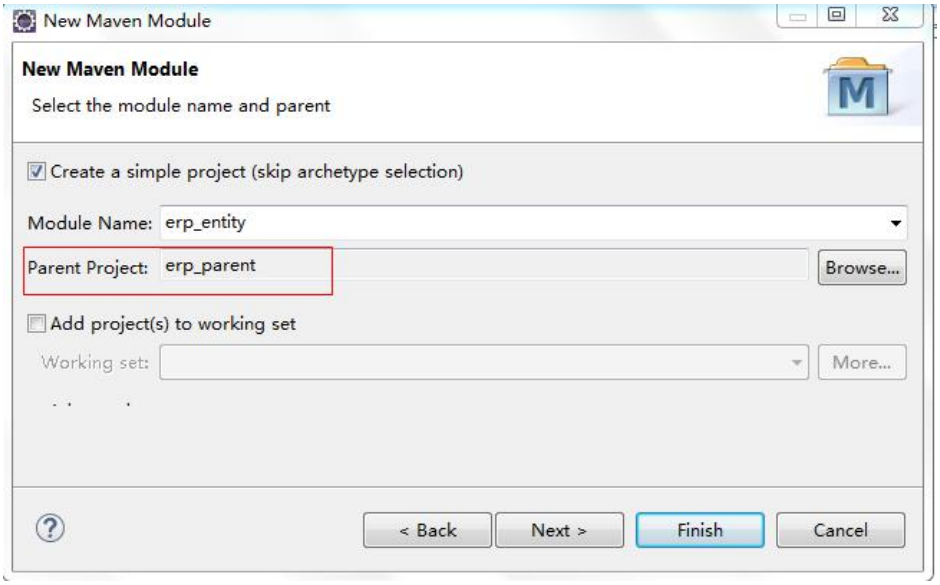


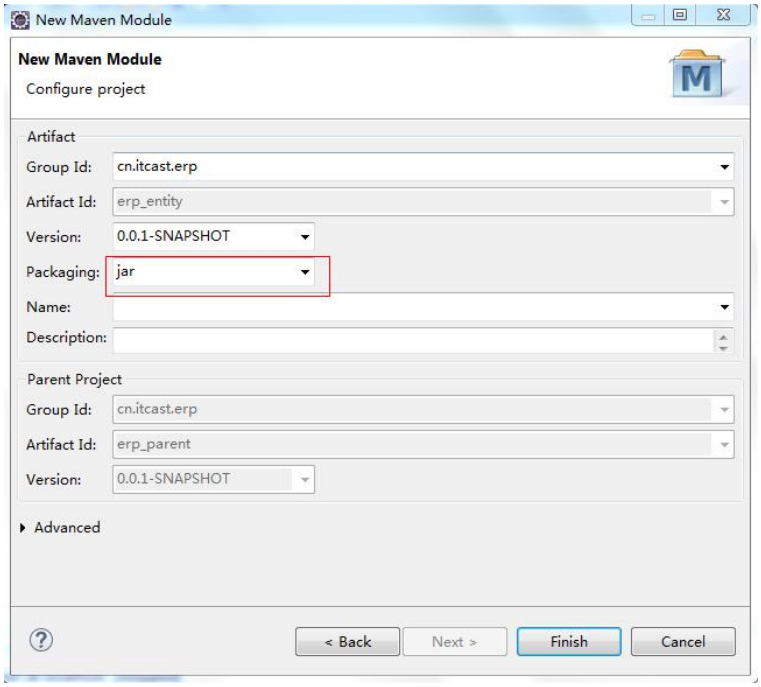
工程创建完成，将 项目必备下\配置文件 下的 pom.xml 复制到此工程

（二）创建 Maven 子模块

选择菜单 file -- new -- other







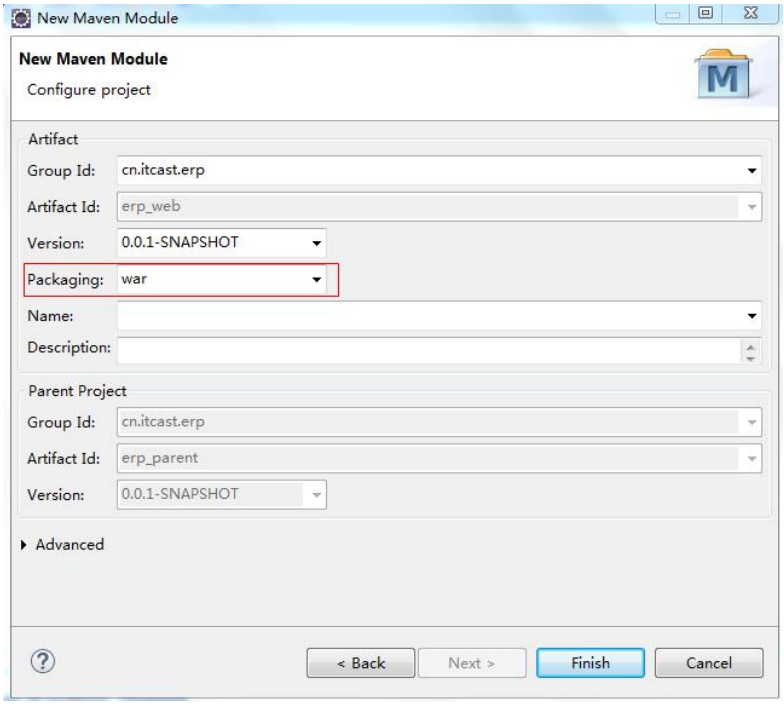
一共需要建立 4 个子模块

erp_entity　存放实体包

erp_dao　存放数据访问接口及实现类

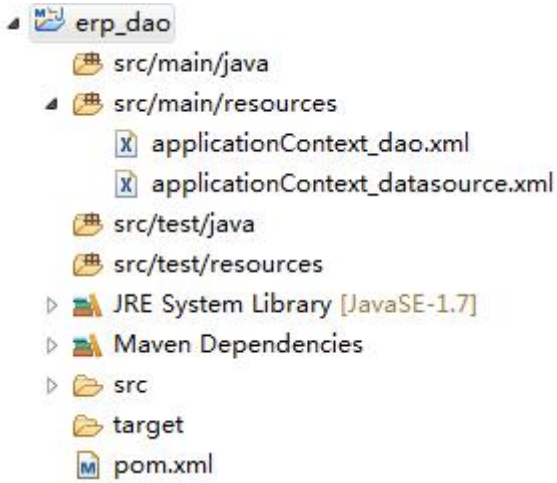
erp_biz　存放业务逻辑层接口及实现类

erp_web　存放 action 类代码和前端代码 （注意 此模块的 packaging 选择 war ）

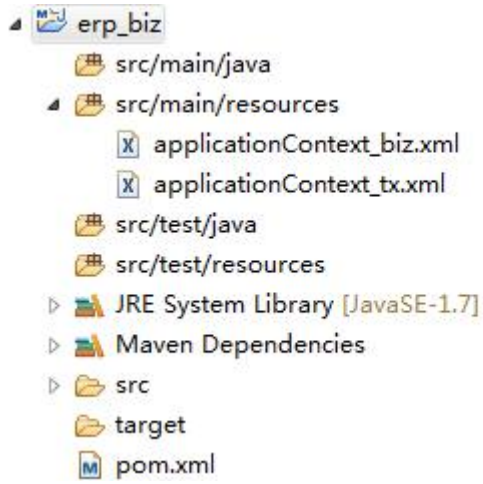


在 erp_web 工程中展开 src \main \webapp 目录 建立 WEB-INF 文件夹，并将 web.xml 拷贝到文件夹中。

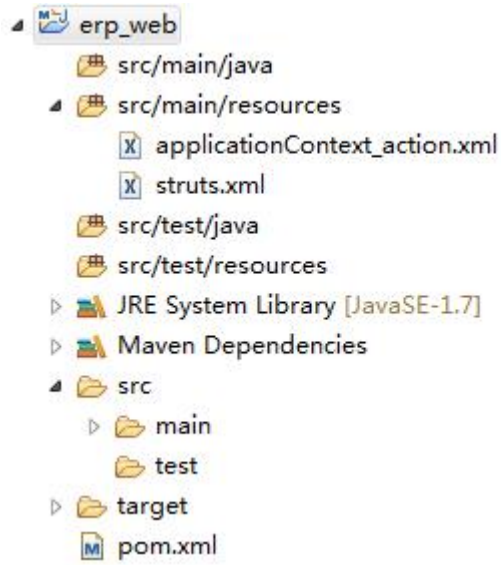
将 spring 配置文件中的 applicationContext_datasource.xml 和 applicationContext_dao.xml 拷贝到 erp_dao 工程的 src/main/resources 下。



将 spring 配置文件中的 applicationContext_tx.xml 和 applicationContext_biz.xml 拷贝到 erp_biz 工程的 src/main/resources 下。



将 struts.xml 和 spring 配置文件中的 applicationContext_action.xml 拷贝到 erp_web 工程的 src/main/resources 下。



（三）建立 Maven 子模块之间的依赖

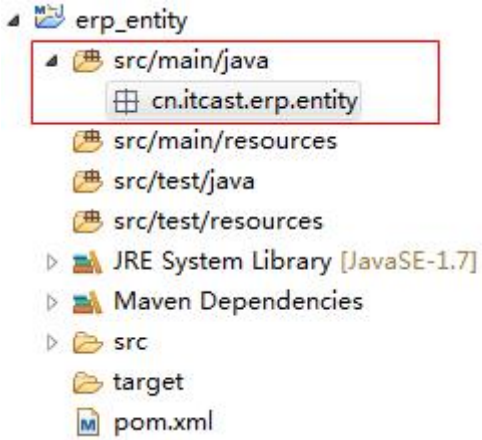
各模块的依赖关系如下：

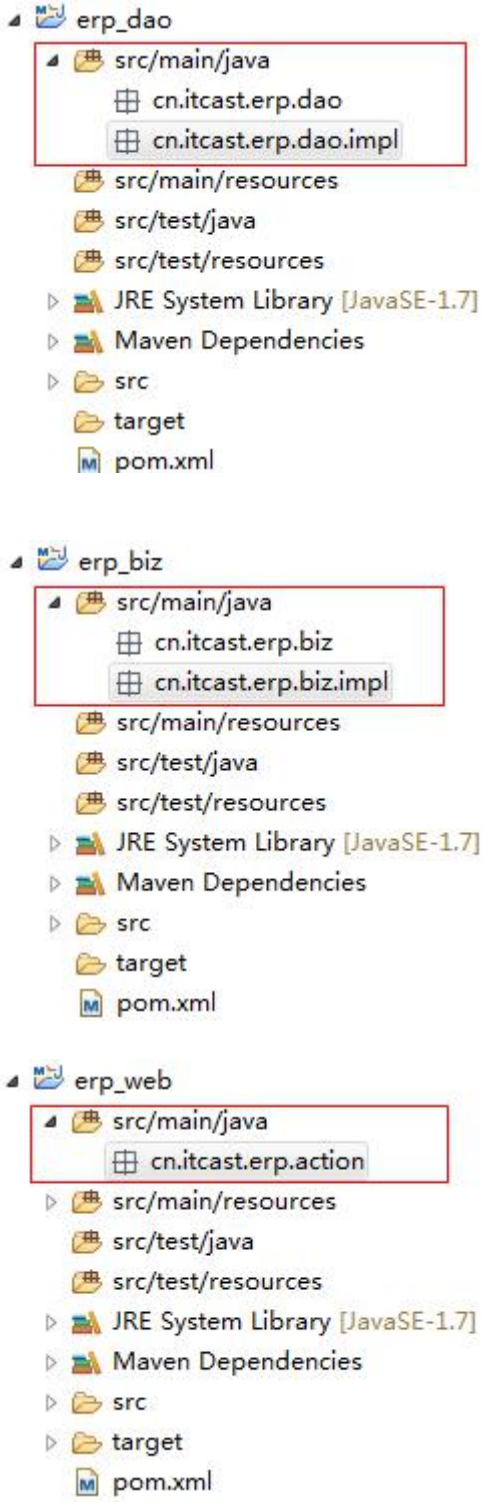
erp_dao 依赖 erp_entity

erp_biz 依赖 erp_dao

erp_web 依赖 erp_biz

（四）在各模块中创建包





（五）修改 Spring 配置文件

修改 erp_dao 工程的 applicationContext_datasource.xml

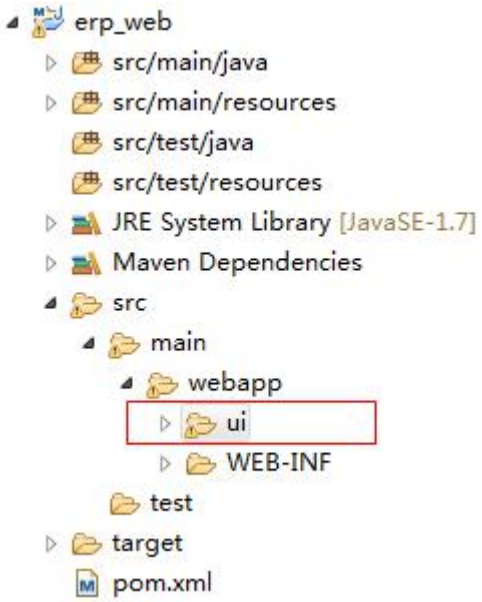
```
<property name="mappingLocations">
    <value>classpath:cn/itcast/erp/entity/*.hbm.xml</value>
</property>

<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@192.168.80.10:1521:ORCL"/>
    <property name="username" value="erpuser"/>
    <property name="password" value="itcast"/>
</bean>
```

修改 erp_biz 工程的 applicationContext_tx.xml

```
<aop:config>
    <!-- 切面 -->
    <aop:pointcut id="serviceMethod" expression="execution(* cn.itcast.erp.biz.impl.*.*(..))"/>
    <aop:advisor pointcut-ref="serviceMethod" advice-ref="advice" />
</aop:config>
```

（六）添加 easyUI 到 erp_web 工程

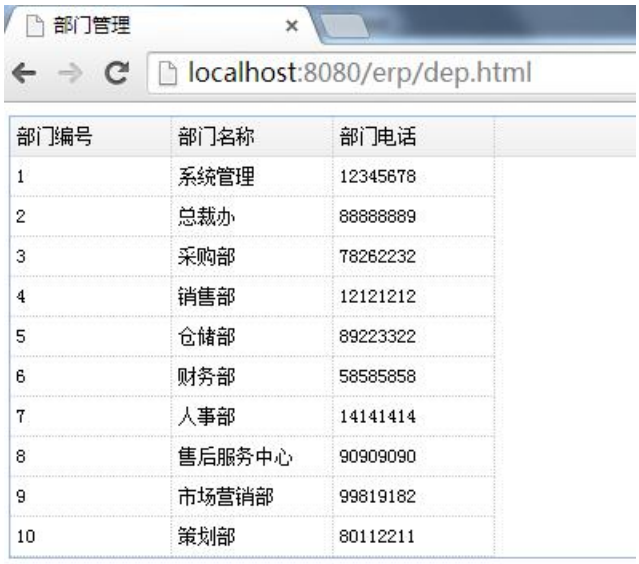


将项目必备文件夹中的 ui 文件夹拷贝到 erp_web 工程的 src/main/webapp 下

八、【部门管理】实现列表查询

（一）需求

实现部门列表的查询，最终效果如下：



部门编号	部门名称	部门电话
1	系统管理	12345678
2	总裁办	88888889
3	采购部	78262232
4	销售部	12121212
5	仓储部	89223322
6	财务部	58585858
7	人事部	14141414
8	售后服务中心	90909090
9	市场营销部	99819182
10	策划部	80112211

（二）后端代码编写

后端代码的目标是完成 action ,地址栏输入 action 地址可以看到列表的 json

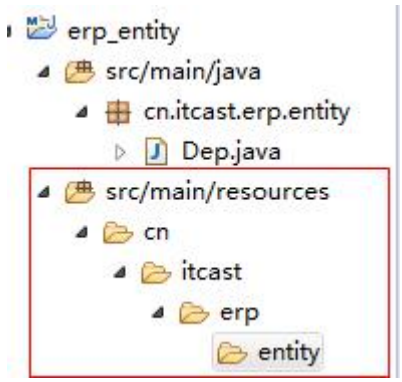
（1）编写实体类

在 erp_entity 工程中的 cn.itcast.erp.entity 包下建立 Dep 类

```
package cn.itcast.erp.entity;
/**
 * 部门实体类
 * @author Administrator
 *
 */
public class Dep {
    private Long uuid;//部门编号
    private String name;//部门名称
    private String tele;//部门电话

    public Long getUuid() {
        return uuid;
    }
    public void setUuid(Long uuid) {
        this.uuid = uuid;
    }
    public String getName() {
        return name;
```

```
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getTele() {
        return tele;
    }
    public void setTele(String tele) {
        this.tele = tele;
    }
}
```



（2）编写映射文件

在 erp_entity 工程的 resources 中创建包 cn.itcast.erp.entity ，并在此目录下创建映射文件 dep.hbm.xml ：

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD
3.0//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
    <class name="cn.itcast.erp.entity.Dep" table="dep">
        <id name="uuid">
        </id>
        <property name="name"></property>
        <property name="tele"></property>
    </class>
```

```
</hibernate-mapping>
```

（3）编写数据访问层接口

在 erp_dao 工程中的 cn.itcast.erp.dao 包下创建 IDepDao 接口

```
package cn.itcast.erp.dao;
import java.util.List;
import cn.itcast.erp.entity.Dep;
/**
 * 部门数据访问接口
 * @author Administrator
 *
 */
public interface IDepDao {

    /**
     * 查询全部列表
     * @return
     */
    public List<Dep> getList();
}
```

（4）编写数据访问层实现类

在 erp_dao 工程中的 cn.itcast.erp.dao.impl 包下创建 DepDao 类

```
package cn.itcast.erp.dao.impl;
import java.util.List;
import org.springframework.orm.hibernate5.support.HibernateDaoSupport;
import cn.itcast.erp.dao.IDepDao;
import cn.itcast.erp.entity.Dep;
/**
 * 部门数据访问类
 * @author Administrator
 *
 */
public class DepDao extends HibernateDaoSupport implements IDepDao {

    /**
     * 查询全部列表
     */
```

```
    public List<Dep> getList() {
        // TODO Auto-generated method stub
        return (List<Dep>) getHibernateTemplate().find("from Dep");
    }
}
```

（5） 编写业务逻辑接口

在 erp_biz 的 cn.itcast.erp.biz 包下创建 IDepBiz 接口

```
package cn.itcast.erp.biz;

import java.util.List;

import cn.itcast.erp.entity.Dep;

public interface IDepBiz {

    /**
     * 查询全部列表
     * @return
     */
    public List<Dep> getList();
}
```

（6） 编写业务逻辑实现类

在 erp_biz 的 cn.itcast.erp.biz.impl 包下创建 DepBiz 类

```
package cn.itcast.erp.biz.impl;

import java.util.List;

import cn.itcast.erp.biz.IDepBiz;
import cn.itcast.erp.dao.IDepDao;
import cn.itcast.erp.entity.Dep;
/**

 * 业务逻辑类

 * @author Administrator
 *
```

```
 */
public class DepBiz implements IDepBiz {

	private IDepDao depDao;

	public void setDepDao(IDepDao depDao) {
		this.depDao = depDao;
	}

	/**
	 * 查询全部列表
	 */
	public List<Dep> getList() {
		return depDao.getList();
	}

}
```

（7）编写 action

在 erp_web 工程下的 cn.itcast.erp.action 包下创建 DepAction 类

```
package cn.itcast.erp.action;
import java.io.IOException;
import java.util.List;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts2.ServletActionContext;
import com.alibaba.fastjson.JSON;
import cn.itcast.erp.biz.IDepBiz;
import cn.itcast.erp.entity.Dep;
/**
 * 部门 Action
 * @author Administrator
 *
 */
public class DepAction {

    private IDepBiz depBiz;

    public void setDepBiz(IDepBiz depBiz) {
```

```
        this.depBiz = depBiz;
    }
    public void list(){
        List<Dep> list = depBiz.getList();
        String jsonString = JSON.toJSONString(list);
        HttpServletResponse response = ServletActionContext.getResponse();
        response.setCharacterEncoding("UTF-8");
        try {
            response.getWriter().print(jsonString);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

我们在 action 中，调用业务逻辑返回列表，转换为 json 格式数据，这里我们采用的是目前最为流行的 fastJSON，它可是由阿里巴巴开发的呦。

（8）配置文件 注入，测试运行

applicationContext.xml

```
<bean id="depDao" class="cn.itcast.erp.dao.impl.DepDao">
    <property name="sessionFactory" ref="sessionFactory"></property>
</bean>
<bean id="depBiz" class="cn.itcast.erp.biz.impl.DepBiz">
    <property name="depDao" ref="depDao"></property>
</bean>
<bean id="depAction" class="cn.itcast.erp.action.DepAction">
    <property name="depBiz" ref="depBiz"></property>
</bean>
```

struts2.xml

```
<struts>
    <package name="default" namespace="/" extends="struts-default">
        <action name="dep_*" class="depAction" method="{1}"></action>
    </package>
</struts>
```

测试运行：

（三）前端页面编写

（1）创建页面 dep.html,引入样式表和 js 文件

```
<link rel="stylesheet" type="text/css" href="ui/themes/default/easyui.css">
<link rel="stylesheet" type="text/css" href="ui/themes/icon.css">
<script type="text/javascript" src="ui/jquery.min.js"></script>
<script type="text/javascript" src="ui/jquery.easyui.min.js"></script>
<script type="text/javascript" src="ui/locale/easyui-lang-zh_CN.js"></script>
```

（2）数据表格（dataGrid）的简单用法

在 body 中添加以下内容：

```
<table id="grid"></table>
```

注意：我们的表格构建方法要放入匿名函数内

```
$(function(){

    $('#grid').datagrid({
        url:'dep_list.action',
        columns:[[
            {field:'uuid',title:'部门编号',width:100},
            {field:'name',title:'部门名称',width:100},
            {field:'tele',title:'部门电话',width:100}
        ]],
        singleSelect:true
    });


});
```

我们这里主要学习 datagrid 最常用的属性

url ：远程 action 的地址

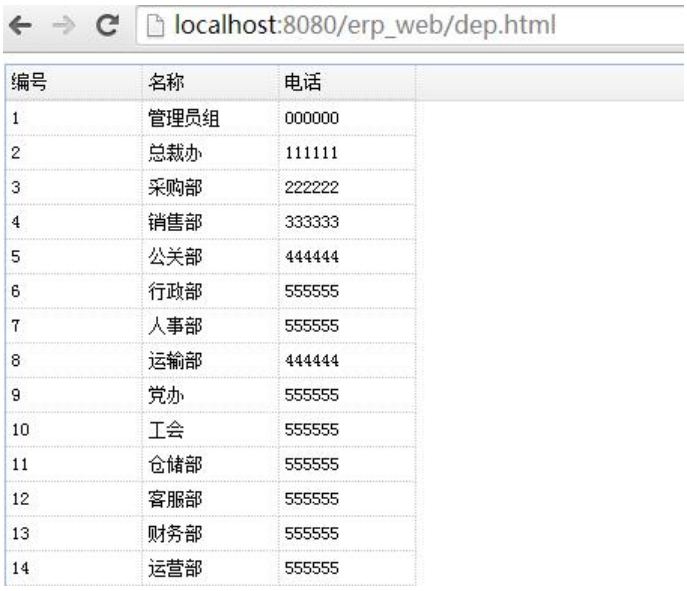
columns ：列定义

singleSelect：为 true 表示单行选中

列属性：

属性名称	属性值类型	描述
title	string	列标题文本。
field	string	列字段名称。
width	number	列的宽度。如果没有定义，宽度将自动扩充以适应其内容。

测试运行：



主要任务：

1. 数据库建立表空间、创建用户、执行建表 SQL　(客户端安装配置可选)
2. 约束配置
3. 搭建框架
4. 部门列表实现
5. 类图、活动图绘制（可选）