# 项目说明

### 项目名

1. 公有云系统
2. **项目内容**
3. 公有云系统：

**项目地址：**

https://github.com/zhicloud/web\_develop/tree/master/cloud\_service/manage\_portal/op

**技术架构：**

Spring+SpringMvc+Mybatis+Mysql+EasyUI

**部署运行环境：**

Linux6.4+JDK7+TOMCAT7

**项目配置文件：**

/coreApplicationContext.xml：Spring核心配置文件，配置Spring相关选项；

/log4j.properties：日志相关配置文件；

/META-INF/app-properties.xml：系统全局属性配置文件，主要在系统管理页使用， 运营商部分地方也有涉及；

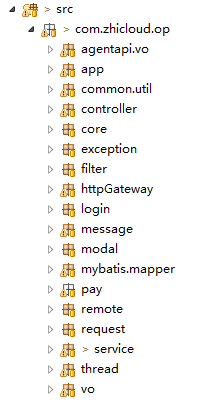
/META-INF/privilege-config.xml：系统权限配置文件，主要用于系统管理页的权限管 理；

/META-INF/regions.xml：各地域httpgateway回调地址的配置文件；

/WEB-INF/web.xml：TOMCAT配置文件，配置servlet、filter等；

/WEB-INF/webApplicationContext.xml：配置所有controller类及文件上传相关属性。

**包结构及主要作用：**



说明：所有包的共有目录为com.zhicloud.op，所以下面叙述中将省略这个目录。

agentapi.vo：主要包含提供给桌面组或第三方的一些API的VO类；

app.authorization：主要是系统权限的分类，便于权限的控制，对应于 /META-INF/privilege-config.xml权限配置文件中的权限；

app.helper：包含多种业务的帮助类，如账单结算、价格计算、登录信息等；

app.listener：包含各种监听类，大部分随着tomcat启动而启动，实现各种后台数 据的获取，状态的查询及缓存等；

app.listener.task：包含上面各监听任务的具体实现类；

app.pool：各种资源池或主机信息的缓存；

app.pool.addressPool：项目启动时，从后台获取地址资源池的信息缓存于此；

app.pool.device:项目启动时，从后台获取设备信息缓存于此；

app.pool.diskImagePool：缓存磁盘镜像信息；

app.pool.host：缓存主机备份和重装的相关信息；

app.pool.hostMonitorInfoPool：缓存主机监控信息；

app.pool.isoImagePool：缓存iso镜像信息；

app.pool.network：缓存VPC(专属云)相关信息；

app.pool.portPool：缓存端口资源池相关信息；

app.pool.rule：缓存智能路由例外规则信息；

app.pool. serverInfoPool：缓存后台服务的相关信息；

app.pool.serviceInfoPool：缓存云服务器相关信息；

app.propeties：包含操作属性文件的一些工具类；

commen.util：系统的各种工具类，包括日期、字符串、数字、线程、XML等工具 类；

commen.util.comstant：包括短息配置、邮件配置和树形结构的节点类；

commen.util.json：JSON相关工具类；

controller：包含各个权限(终端用户、代理商、运营商和系统管理员)和一些特殊的 控制类；

core：仅包含一个获取Spring容器中bean的工厂类；

exception：包含自定义异常及相关管理的类；

filter：包含异常拦截、安全登录拦截等类；

httpGateway：用于和python交互获取各种后台信息；

login：用于保存登录信息；

message：邮件和短信的一些模版信息；

modal：仅包含一个树结构的节点类；

mybatis.mapper：包含Mybatis中所有绑定映射语句的接口及配置文件；

pay：包含支付宝和银联支付相关功能的实现；

remote：方法远程调用相关功能；

service：包含所有业务逻辑类及接口；

service.constant：包含系统的各种变量表；

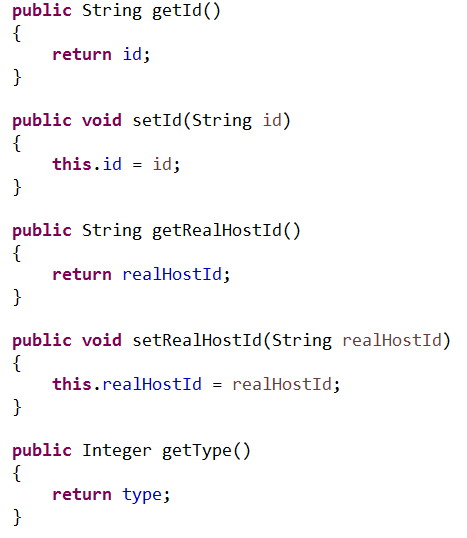
service.impl：各个业务接口的具体实现类；

vo：包含所有实体类；

**典型功能开发实例：**

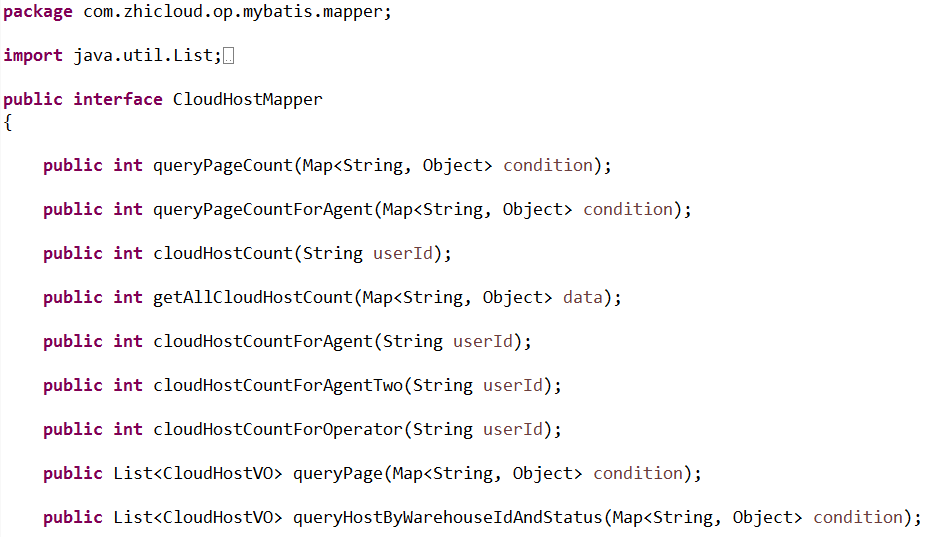
终端用户云主机管理:

首先编写相应的VO类，主要用到CloudHostVO.java，由于是贫血模型应用架构，类中仅提供相应的setter和getter方法，如下图(忽略XML化的标签)：



由于类太长，这里只截取部分属性。

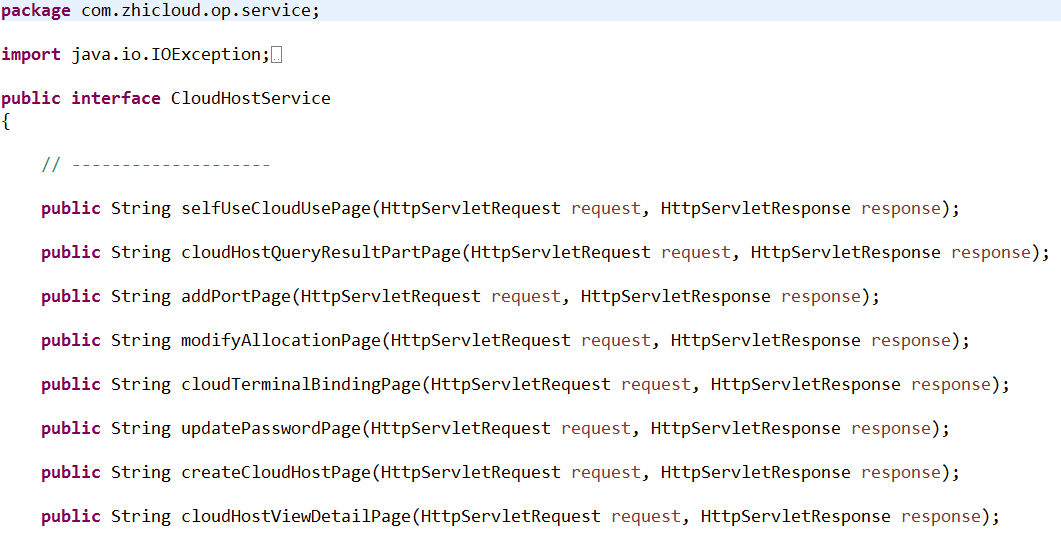
然后编写Mybatis映射的接口类及配置文件，如下图：





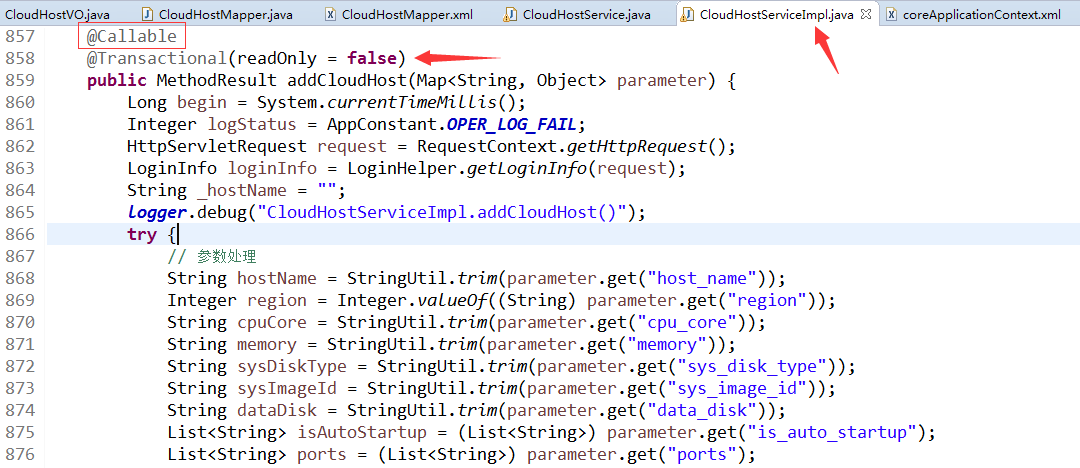
以上分别是CloudHostMapper.java和CloudHostMapper.xml。

接着是业务逻辑类即service，由于是基于接口编程，所以有要提供接口类及其实现类，如下图：





实现类CloudHostServiceImpl.java中提供类注解@Transactional(readOnly = true)，标注所 有方法为只读，当方法有写需求时再单个标记为可写@Transactional(readOnly = false);



值得注意的是，所有方法都要加上@callable注解才能被外部所调用。

之后实现类要加入Spring容器管理，这里没有使用注解，而是在Spring的核心配置文 件中配置的(当然使用注解也是可以的)：

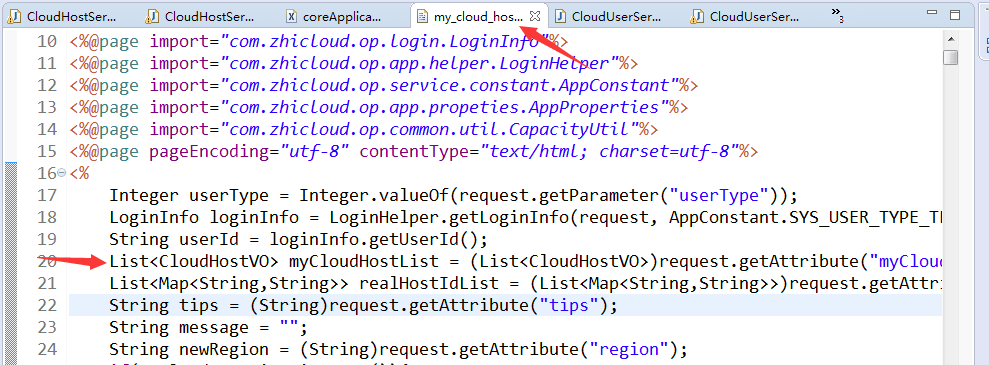


如此，后端代码就全部准备好了，最后是前端页面的调用。当点击页面的“我的云主机”按纽时，后台会获取当前用户的所有云主机传递给前端：





然后页面获取所有数据再迭代显示：







页面中的所有操作都是向后端发送请求然后返回JSON数据。

**关于项目中的图表：**

使用的是百度的Echarts：<http://echarts.baidu.com/，这里有完整的示例及文档。>

**关于数据库：**

开发用的数据库地址：172.18.10.18：3306，数据库：zcop，用户名：cdms，密码：cdms$250