package com.yawa.base.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.plugin.activerecord.Db;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Company;

import com.yawa.core.model.Menu;

import com.yawa.core.model.User;

import com.yawa.model.Tab;

import com.yawa.util.SecurityContextUtil;

import org.apache.commons.lang.time.DateFormatUtils;import java.util.Date;

import java.util.List;

@Before(SessionInterceptor.class)

public class HomeController extends Controller {

public void index() {

setAttr("records", Db.find("select t.id,t.name,sum(case when t1.id is null then 0 else 1 end) cnt"

+ " from base\_companys t left join (select \* from tanks where state = 1) t1"

+ " on t.id = t1.company where t.state = 1 group by t.id order by t.id asc"));

render("home.jsp");

}

public void getUserRights() {

Integer user = SecurityContextUtil.getLoginUserId(getRequest());

renderJson(User.me.getGroupRights(user, "home"));

}

public void menu() {

User user = SecurityContextUtil.getLoginUser(getRequest());

setAttr("menus", Menu.me.getLoginMenus(user.getInt("id")));

render("menu.jsp");

}

public void leader() {

User user = SecurityContextUtil.getLoginUser(getRequest());

Integer company = user.getInt("company");

setAttr("user", User.me.getTotal(company));

//首页监控图

setAttr("montht", DateFormatUtils.format(new Date(), "yyyy-MM"));

setAttr("comList",Company.me.getSubComs(user.getInt("company")));

//首页监控图

render("leader.jsp");

}

/\*\*

\* 获取取得条数的sql

\*/

public String sqlToCntSql(String sql){

return "select count(1) cnt from ("+sql+") tbfcnt";

}

public void first() {

User user = SecurityContextUtil.getLoginUser(getRequest());

int userId = user.get("id");

String account = user.get("account"); List<Tab> lte = Tab.me.getUserTab(userId,1);

List<Tab> lts = Tab.me.getUserTab(userId,2); StringBuffer sqle = new StringBuffer();

sqle.append("select id,name title,DATE\_FORMAT(create\_time,'%Y/%m/%d %H:%i:%s') timestr from base\_enterprise where state = 1 and type = 1 and not( 1=1 ");

for(Tab te : lte){

String tabId = te.get("tab\_id").toString();

sqle.append("and FIND\_IN\_SET('");

sqle.append(tabId);

sqle.append("',ifnull(tab\_pass,'')) ");

}

sqle.append(") ");

String ecntinner = sqle.toString();

sqle.append("order by create\_time desc limit 5 "); StringBuffer sqls = new StringBuffer();

sqls.append("select id,name title,DATE\_FORMAT(create\_time,'%Y/%m/%d %H:%i:%s') timestr from base\_enterprise where state = 1 and type = 2 and not( 1=1 ");

for(Tab ts : lts){

String tabId = ts.get("tab\_id").toString();

sqls.append("and FIND\_IN\_SET('");

sqls.append(tabId);

sqls.append("',ifnull(tab\_pass,'')) ");

}

sqls.append(") ");

String scntinner = sqls.toString();

sqls.append("order by create\_time desc limit 5 "); //setAttr("daishenqiye",Db.find("select id,name title,DATE\_FORMAT(create\_time,'%Y/%m/%d %H:%i:%s') timestr from base\_enterprise where state = 1 and type = 1 order by create\_time desc limit 5 "));

setAttr("daishenqiye",Db.find(sqle.toString()));

//setAttr("daishenzhandian",Db.find("select id,name title,DATE\_FORMAT(create\_time,'%Y/%m/%d %H:%i:%s') timestr from base\_enterprise where state = 1 and type = 2 order by create\_time desc limit 5"));

setAttr("daishenzhandian",Db.find(sqls.toString()));

setAttr("task",Db.find("select id,title,DATE\_FORMAT(starttime,'%Y/%m/%d %H:%i:%s') timestr from base\_inspection\_task where stat = 0 and inspector = ? order by endtime desc limit 5",account));

setAttr("news",Db.find("select id,title,url,DATE\_FORMAT(create\_date,'%Y/%m/%d %H:%i:%s') timestr from news where state = 2 order by update\_date desc limit 5"));

setAttr("notice",Db.find("select id,title,DATE\_FORMAT(ndate,'%Y/%m/%d %H:%i:%s') timestr from notice order by ndate desc limit 5"));

setAttr("file",Db.find("select id,oriname title,DATE\_FORMAT(fdate,'%Y/%m/%d %H:%i:%s') timestr from shares where busi = 'share' order by fdate desc limit 5"));

setAttr("study",Db.find("select id,descript title,DATE\_FORMAT(fdate,'%Y/%m/%d %H:%i:%s') timestr from shares where busi = 'study' order by fdate desc limit 5")); setAttr("daishenqiyecnt",

Db.findFirst(sqlToCntSql(ecntinner)).get("cnt"));

setAttr("daishenzhandiancnt",

Db.findFirst(sqlToCntSql(scntinner)).get("cnt"));

setAttr("renwuguanlicnt",

Db.findFirst("select count(1) cnt from base\_inspection\_task where stat = 0 and inspector = ? ",account).get("cnt"));

render("first.jsp");

}

}

package com.yawa.base.controller;import java.util.List;

import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.yawa.core.interceptor.LoginInterceptor;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Company;

import com.yawa.core.model.Menu;

import com.yawa.core.model.User;

import com.yawa.core.model.virtual.Validate;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.Attach;

import com.yawa.util.model.ConstantItems;

import com.yawa.util.model.ResponseData;/\*\*

\* 登录操作

\* @author wrz

\*/

public class LoginController extends Controller {

@Before(LoginInterceptor.class)

public void login() {

render("login.jsp");

}

@Before(POST.class)

public void login\_captcha() {

renderJson(Validate.me.login\_captcha(getRequest(), getPara("account")));

}

/\*\*

\* 用户登陆认证

\* @return

\*/

@Before(POST.class)

public void login\_validate() {

try{

renderJson(Validate.me.login\_validate(getRequest(), getPara("captcha"),

getPara("account"), getPara("password")));

}catch(Exception e){

renderJson(new ResponseData(false,"系统处理失败：" + e.getMessage()));

}

} /\*\*

\* 登录成功后显示主页

\*/

@Before(SessionInterceptor.class)

public void index() {

User user = SecurityContextUtil.getLoginUser(getRequest());

setAttr("user", user);

List<Attach> attach = Attach.me.getAttaches(ConstantItems.ATTACH\_AVATAR, user.getInt("id"));

if(attach.size() > 0)

setAttr("attach", attach.get(0));

Integer company = user.getInt("company");

if(company != null)

setAttr("company", Company.me.findById(company));

setAttr("shortCuts", Menu.me.getShortCuts(user.getInt("id")));

render("index.jsp");

}

package com.yawa.core.controller;import java.util.Date;

import org.apache.commons.lang.RandomStringUtils;

import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Company;

import com.yawa.core.model.Dict;

import com.yawa.core.model.Org;

import com.yawa.core.model.Role;

import com.yawa.core.model.User;

import com.yawa.core.model.virtual.Captcha;

import com.yawa.core.model.virtual.Validate;

import com.yawa.util.CodeUtil;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.ResponseData;@Before(SessionInterceptor.class)

public class UserController extends Controller { /\*\*

\* 个人设置

\*/

public void settings() {

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

setAttr("user", user);

setAttr("genders", Dict.me.getGroupItems("BASE.GENDER"));

setAttr("vtypes", Dict.me.getGroupItems("USER.VTYPE"));

setAttr("org", Org.me.findFirst("select \* from base\_orgs where id = ?", user.getInt("org\_id")));

render("settings.jsp");

} /\*\*

\* 修改密码

\*/

public void password() {

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

setAttr("user", user);

render("password.jsp");

} /\*\*

\* 信息管理首页

\* @param request

\*/

public void index() {

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

setAttr("classify", user.getStr("classify"));

setAttr("company", Company.me.getById(user.getInt("company")));

render("index.jsp");

}

/\*\*

\* 通讯录

\*/

public void mailList() {

User user = SecurityContextUtil.getLoginUser(getRequest());

setAttr("company", Company.me.getById(user.getInt("company")));

render("mailList.jsp");

}

/\*\*

\* 系统权限授权

\*/

public void grant() {

setAttr("id", getParaToInt("id"));

render("grant.jsp");

}

/\*\*

\* 客户端授权

\*/

public void app() {

setAttr("id", getParaToInt("id"));

render("app.jsp");

}

/\*\*

\* 查询人员信息分页列表

\*/

public void getPages() {

Integer company = getParaToInt("company");

if(company == null){

User user = SecurityContextUtil.getLoginUser(getRequest());

company = user.getInt("company");

}

renderJson(User.me.getPages(getPara("name"), getPara("account"),

getPara("tel"), company, getParaToInt("page"), getParaToInt("limit")));

} /\*\*

\* 功能：用户角色授权查询

\*/

public void getUserRoles() {

Integer id = getParaToInt("id");

User user = User.me.findById(id);

User login = SecurityContextUtil.getLoginUser(getRequest());

renderJson(User.me.getUserRoles(id, user.getInt("company"), login.getInt("company")));

}

/\*\*

\* 功能：展示所有权限，用户可以选择权限

\*/

public void getUserRights() {

String roleIds = getPara("roleIds");

renderJson(User.me.getUserRights(getParaToInt("id"), roleIds));

}

/\*\*

\* 用户授权显示菜单

\*/

public void getRoleMenus() {

String roleIds = getPara("roleIds");

renderJson(Role.me.getRoleMenus(roleIds));

} /\*\*

\* 手机端角色列表

\*/

public void getUserAppRoles() {

Integer id = getParaToInt("id");

renderJson(User.me.getUserAppRoles(id));

}

/\*\*

\* 人员编辑页面

\*/

public void edit() {

Integer id = getParaToInt("id");

Integer company = getParaToInt("company");

User login = SecurityContextUtil.getLoginUser(getRequest());

User user = new User();

if (id != null){

user = User.me.findById(getParaToInt("id"));

company = user.getInt("company");

}else{

if(company == null)

company = login.getInt("company");

user.set("company", company);

}

setAttr("user", user);

setAttr("genders", Dict.me.getGroupItems("BASE.GENDER"));

setAttr("vtypes", Dict.me.getGroupItems("USER.VTYPE"));

setAttr("orgs", Org.me.find("select \* from base\_orgs where state = 1 and company = ?", company));

setAttr("companys", Company.me.getSubComs(login.getInt("company")));

render("edit.jsp");

}

/\*\*

\* 登录获取验证码

\* @return

\*/

@Before(POST.class)

public void pass\_captcha() {

User user = User.me.findById(getParaToInt("id"));

renderJson(Captcha.me.sendCaptcha(getRequest(), user, "password", "密码修改"));

}

/\*\*

\* 修改密码

\*/

public void modifyPass(){

String captcha = getPara("captcha");

if(!SecurityContextUtil.isCaptchaValid(getRequest())){

new ResponseData(false, "验证码已过期，请重新获取!");

return;

}

if(!SecurityContextUtil.validateCaptcha(getRequest(), captcha)){

renderJson(new ResponseData(false, "验证码错误，请重新输入!"));

return;

}

Integer id = getParaToInt("id");

User u = User.me.findById(id);

renderJson(Validate.me.modifyPass(getRequest(), u, getPara("old\_password"),

getPara("new\_password"), getPara("confirm\_password")));

}

/\*\*

\* 个人设置

\*/

public void modifyUser(){

User user = getModel(User.class, "user");

if(user != null)

user.set("update\_date", new Date())

.update();

renderJson(new ResponseData(true, "信息修改成功！ "));

}

/\*\*

\* 重置密码

\*/

public void resetPass(){

String salt = RandomStringUtils.randomAlphanumeric(10);

String pass = RandomStringUtils.randomAlphanumeric(8);

User user = User.me.findById(getParaToInt("id"));

if(user == null){

renderJson(new ResponseData(false, "未获取相关用户记录！ "));

return;

}

user.set("salt", salt)

.set("state", 1)

.set("error\_cnt", 0)

.set("password", CodeUtil.md5(pass + salt))

.set("update\_date", new Date())

.update();

renderJson(new ResponseData(true, "用户【" + user.getStr("name") + "】密码重置为：【" + pass + "】！ "));

}

/\*\*

\* 用户解锁

\*/

public void unlock(){

User user = User.me.findById(getParaToInt("id"));

if(user == null){

renderJson(new ResponseData(false, "未获取相关用户记录！ "));

return;

}

user.set("state", 1)

.set("error\_cnt", 0)

.set("update\_date", new Date())

.update();

renderJson(new ResponseData(true, "用户【" + user.getStr("name") + "】解锁成功！ "));

}

/\*\*

\* 用户角色授权

\*/

public void saveRoleRights() {

try {

Integer user = getParaToInt("user");

String roleids = getPara("roleids");

String rightids = getPara("rightids");

User.me.saveUserRoles(user, roleids);

User.me.saveUserRights(user, rightids);

} catch (Exception e) {

renderJson(new ResponseData(false, "保存发生异常，异常操作如下: " + e.getMessage()));

return;

}

renderJson(new ResponseData(true, "用户系统授权成功!"));

}

/\*\*

\* 客户端角色授权

\*/

public void saveAppRoleRights() {

try {

Integer user = getParaToInt("user");

String roleIds = getPara("roleIds");

User.me.saveAppUserRole(user, roleIds);

String addIds = getPara("addIds");

String revIds = getPara("revIds");

User.me.saveAppUserModules(user, addIds, revIds);

} catch (Exception e) {

renderJson(new ResponseData(false, "保存发生异常，异常操作如下: " + e.getMessage()));

return;

}

renderJson(new ResponseData(true, "用户客户端授权成功!"));

}

/\*\*

\* 保存用户所选快捷方式

\*/

public void saveShortCuts() {

try {

User user = SecurityContextUtil.getLoginUser(getRequest());

String[] shortCuts = getParaValues("shortCuts");

if (user != null)

if(null != shortCuts && shortCuts.length > 0){

String menuIds = "";

for(int i = 0;i < shortCuts.length;i++)

menuIds = menuIds + shortCuts[i] + ",";

User.me.saveShortCuts(user.getInt("id"), menuIds);

}

} catch (Exception e) {

renderJson(new ResponseData(false, "保存发生异常，异常操作如下: " + e.getMessage()));

return;

}

renderJson(new ResponseData(true, "保存成功!"));

}

/\*\*

\* 删除用户

\*/

public void delete() {

User user = User.me.findById(getParaToInt("id"));

if (user != null) {

try {

user.set("state", 0)

.set("update\_date", new Date())

.update();

} catch (Exception e) {

renderJson(new ResponseData(false, "删除发生异常，异常操作如下：" + e.getMessage()));

return;

}

}

renderJson(new ResponseData(true, "记录删除成功!"));

} /\*\*

\* 新增/修改用户

\*/

public void update() {

try {

User user = getModel(User.class, "user");

if (user != null) {

boolean hasAccount = User.me.hasAccount(user);

if(hasAccount){

renderJson(new ResponseData(false, "用户登录账号已经存在，请重新填写！"));

return;

}

boolean hasPhone = User.me.hasPhone(user);

if(hasPhone){

renderJson(new ResponseData(false, "用户手机号已经存在，请重新填写！"));

return;

}

if (user.get("id") == null) {

String salt = RandomStringUtils.randomAlphanumeric(10);

String pass = RandomStringUtils.randomAlphanumeric(8);

user.set("salt", salt)

.set("password",CodeUtil.md5(pass + salt))

.set("state", 1)

.set("creator",SecurityContextUtil.getLoginUserId(getRequest()))

.set("create\_date", new Date())

.save();

renderJson(new ResponseData(true, "用户新增成功，初始密码为【" + pass + "】!"));

} else{

user.set("update\_date", new Date())

.update();

renderJson(new ResponseData(true, "用户信息修改成功!"));

}

}

} catch (Exception e) {

renderJson(new ResponseData(false, "保存发生异常，异常操作如下: " + e.getMessage()));

}

}

}

/\*\*

\* 登出处理

\* @param request

\* @return

\*/

@Before(SessionInterceptor.class)

public void logOut() {

SecurityContextUtil.invalidateSession(getRequest());

redirect("/login");

}

}

package com.yawa.base.controller;import java.util.List;

import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.yawa.core.interceptor.LoginInterceptor;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Company;

import com.yawa.core.model.Menu;

import com.yawa.core.model.User;

import com.yawa.core.model.virtual.Validate;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.Attach;

import com.yawa.util.model.ConstantItems;

import com.yawa.util.model.ResponseData;/\*\*

\* 登录操作

\* @author wrz

\*/

public class LoginController extends Controller {

/\*\*

\* 如果已经登录，则直接跳转至首页

\*/

@Before(LoginInterceptor.class)

public void login() {

render("login.jsp");

}

/\*\*

\* 登录获取验证码

\* @return

\*/

@Before(POST.class)

public void login\_captcha() {

renderJson(Validate.me.login\_captcha(getRequest(), getPara("account")));

}

/\*\*

\* 用户登陆认证

\* @return

\*/

@Before(POST.class)

public void login\_validate() {

try{

renderJson(Validate.me.login\_validate(getRequest(), getPara("captcha"),

getPara("account"), getPara("password")));

}catch(Exception e){

renderJson(new ResponseData(false,"系统处理失败：" + e.getMessage()));

}

} /\*\*

\* 登录成功后显示主页

\*/

@Before(SessionInterceptor.class)

public void index() {

User user = SecurityContextUtil.getLoginUser(getRequest());

setAttr("user", user);

List<Attach> attach = Attach.me.getAttaches(ConstantItems.ATTACH\_AVATAR, user.getInt("id"));

if(attach.size() > 0)

setAttr("attach", attach.get(0));

Integer company = user.getInt("company");

if(company != null)

setAttr("company", Company.me.findById(company));

setAttr("shortCuts", Menu.me.getShortCuts(user.getInt("id")));

render("index.jsp");

}

/\*\*

\* 登出处理

\* @param request

\* @return

\*/

@Before(SessionInterceptor.class)

public void logOut() {

SecurityContextUtil.invalidateSession(getRequest());

redirect("/login");

}

}

package com.yawa.core.model.virtual;import com.jfinal.plugin.activerecord.Db;

import com.yawa.core.model.Base;

import com.yawa.core.model.User;

import com.yawa.util.CodeUtil;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.ToolKit;

import com.yawa.util.model.ResponseData;

import org.apache.commons.lang.RandomStringUtils;

import org.apache.commons.lang.StringUtils;import javax.servlet.http.HttpServletRequest;

import java.util.Date;public class Validate { public static final Validate me = new Validate(); /\*进行登录校验\*/

public ResponseData login\_validate(HttpServletRequest request, String captcha, String account, String password) {

User user = null;

if (null != captcha && !captcha.equals("null")) {

if (!SecurityContextUtil.validateCaptcha(request, captcha))

return new ResponseData(false, "验证码错误，请重新输入!");

if (!SecurityContextUtil.isCaptchaValid(request))

return new ResponseData(false, "验证码已过期，请重新获取!");

}

try {

user = User.me.getLoginUser(account);

} catch (Exception e) {

return new ResponseData(false, "数据库错误，请联系管理员!");

}

if (user == null)

return new ResponseData(false, "用户名或密码错误，请重新输入!");

if (user.getInt("state") == 0)

return new ResponseData(false, "账户已无效，请联系管理员!");

if (user.getInt("state") == 2)

return new ResponseData(false, "登录错误次数过多，账户已锁定，请联系管理员解锁!");

String encrypt = ToolKit.md5(password + user.getStr("salt"));

if (!encrypt.equals(user.getStr("password"))) {

Integer error\_cnt = user.getInt("error\_cnt") + 1;

String value = Base.me.getValue("login\_cnt");

if (StringUtils.isNotEmpty(value)) {

Integer log\_cnt = Integer.parseInt(value);

if (error\_cnt >= log\_cnt) {

user.set("error\_cnt", error\_cnt)

.set("state", 2)

.set("update\_date", new Date())

.update();

return new ResponseData(false, "登录错误次数达到" + error\_cnt + "次，账户锁定，请联系管理员解锁!");

}

}

user.set("error\_cnt", error\_cnt)

.set("update\_date", new Date())

.update();

return new ResponseData(false, "用户名或密码错误，请重新输入!");

}

user.set("error\_cnt", 0)

.set("update\_date", new Date())

.update();

SecurityContextUtil.setLoginUser(request, user);

return new ResponseData(true, "登录成功", user);

} public ResponseData login\_validate(HttpServletRequest request, String account, String password, String dId, boolean isAuto) {

User user = null;

try {

user = User.me.getLoginUser(account);

} catch (Exception e) {

return new ResponseData(false, "数据库错误，请联系管理员!");

}

if (user == null)

return new ResponseData(false, "用户名或密码错误，请重新输入!");

if (user.getInt("state") == 0)

return new ResponseData(false, "账户已无效，请联系管理员!");

if (user.getInt("state") == 2)

return new ResponseData(false, "登录错误次数过多，账户已锁定，请联系管理员解锁!");

if (!isAuto) {

password = ToolKit.md5(password + user.getStr("salt"));

if (!password.equals(user.getStr("password"))) {

Integer error\_cnt = user.getInt("error\_cnt");

String value = Base.me.getValue("login\_cnt");

if (StringUtils.isNotEmpty(value)) {

Integer log\_cnt = Integer.parseInt(value);

if (error\_cnt >= log\_cnt) {

user.set("error\_cnt", error\_cnt).set("state", 2).set("update\_date", new Date()).update();

return new ResponseData(false, "登录错误次数已达到" + error\_cnt + "次，账户锁定，请联系管理员解锁!");

}

}

user.set("error\_cnt", error\_cnt)

.set("update\_date", new Date())

.update();

return new ResponseData(false, "用户名或密码错误，请重新输入!");

}

} else {

if (Db.find("SELECT \* FROM ngmsg.base\_users WHERE dId = ? AND account=?", dId, account).size() < 1) {

SecurityContextUtil.invalidateSession(request);

return new ResponseData(false, "非法登陆");

}

}

user.set("error\_cnt", 0)

.set("update\_date", new Date())

.set("dId", dId)

.update();

SecurityContextUtil.setLoginUser(request, user);

return new ResponseData(true, "登录成功", user);

}

/\*\*

\* 发送登录验证码

\*

\* @param account

\* @return

\*/

public ResponseData login\_captcha(HttpServletRequest request, String account) {

User user = null;

try {

user = User.me.getLoginUser(account);

} catch (Exception e) {

return new ResponseData(false, "系统错误，请联系管理员!");

}

if (user == null)

return new ResponseData(false, "账户错误，无法发送验证码!");

if (user.getInt("state") != 1)

return new ResponseData(false, "账户已无效或锁定，请联系管理员!");

return Captcha.me.sendCaptcha(request, user, "login", "账户登录");

} /\*\*

\* 密码修改

\*

\* @param request

\* @param u

\* @param oldPass

\* @param newPass

\* @param confPass

\* @return

\*/

public ResponseData modifyPass(HttpServletRequest request, User u, String oldPass,

String newPass, String confPass) {

if (u == null)

return new ResponseData(false, "未找到用户记录，请联系管理员!");

if (u.getInt("state") == 0)

return new ResponseData(false, "账户已无效，请联系管理员!");

if (u.getInt("state") == 2)

return new ResponseData(false, "登录错误次数过多，账户已锁定，请联系管理员解锁!");

String encrypt = CodeUtil.md5(oldPass + u.getStr("salt"));

if (!encrypt.equals(u.getStr("password")))

return new ResponseData(false, "原密码错误，请重新输入!");

if (!newPass.equals(confPass))

return new ResponseData(false, "新密码与确认密码不一致！");

if (newPass.length() < 6)

return new ResponseData(false, "密码长度必须大于6位！");

if (newPass.length() > 12)

return new ResponseData(false, "密码长度必须小于12位！");

String salt = RandomStringUtils.randomAlphanumeric(10);

u.set("salt", salt)

.set("state", 1)

.set("error\_cnt", 0)

.set("password", CodeUtil.md5(newPass + salt))

.set("update\_date", new Date())

.update();

return new ResponseData(true, "密码修改成功！");

}

}

package com.yawa.util.model;import java.io.File;

import java.io.IOException;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.apache.commons.io.FileUtils;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.jfinal.plugin.activerecord.Record;

import com.jfinal.upload.UploadFile;

import com.yawa.util.ToolKit;/\*\*

\* 附件表

\* @author wrz

\*

\*/

@SuppressWarnings("serial")

public class Attach extends Model<Attach> {

public static final Attach me = new Attach();

/\*\*

\* 获取附件信息

\* @param attach\_type

\* @param attach\_id

\* @return

\*/

public List<Attach> getAttaches(String attach\_type, Integer attach\_id) {

return find("select a.\*,cast(concat((select value from app\_base where code = 'access\_path'),'/',a.attach\_type,'/',a.attach\_id,'/',a.store\_name) as char) url"

+ " from attaches a where attach\_type = ? and attach\_id = ? order by id asc", attach\_type,attach\_id);

} /\*\*

\* 保存附件信息

\* @param attach\_type

\* @param attach\_id

\* @param uploadFiles

\*/

public void saveFile(String attach\_type, Long attach\_id, List<UploadFile> uploadFiles) {

Record recd = Db.findFirst("select value from app\_base where code = 'attach\_store'");

if(recd != null){

String baseDir = recd.getStr("value");

Date nowDate = new Date();

Map<String, Object> paraMap = new HashMap<String, Object>();

paraMap.put("attach\_type", attach\_type);

paraMap.put("attach\_id", attach\_id);

paraMap.put("create\_date", nowDate);

for(int i = 0;i < uploadFiles.size();i++){

File file = uploadFiles.get(i).getFile();

String storeName = ToolKit.getStoreName(nowDate, file.getName(), i);

Attach attach = new Attach();

attach.setAttrs(paraMap)

.set("size", file.length())

.set("upload\_name", file.getName())

.set("store\_name", storeName)

.save();

File destDir = new File(baseDir,attach\_type + "/" + attach\_id);

if(!destDir.exists())

destDir.mkdirs();

File destFile = new File(destDir,storeName);

try {

FileUtils.moveFile(file, destFile);

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

}

package com.yawa.home.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.model.Car;@Before(SessionInterceptor.class)

public class CarMapController extends Controller {

/\*\*

\* 地图监控

\*/

public void index() {

render("index.jsp");

}

/\*\*

\* 获取车辆列表

\*/

public void getCars() {

renderJson(Car.me.getCars());

}

/\*\*

\* 获取车辆信息定位

\*/

public void findCar() {

renderJson(Car.me.findById(getParaToInt("id")));

}

/\*\*

\* 获取车辆轨迹记录

\*/

public void getCarTracks() {

Integer id = getParaToInt("id");

renderJson(Car.me.getCarTracks(id));

}

}

package com.yawa.model;import java.util.List;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.jfinal.plugin.activerecord.Record;@SuppressWarnings("serial")

public class Car extends Model<Car> {

public static final Car me = new Car(); /\*\*

\* 获取车辆列表

\* @return

\*/

public List<Car> getCars() {

return find("select \* from cars where state = 1");

}

/\*\*

\* 获取车辆轨迹记录

\* @source remote\_cars

\* @param car

\* @return

\*/

public List<Record> getCarTracks(Integer id) {

String sql = "";

if(id != null)

sql = " and c.id = " + id;

return Db.find("select c.carid,r.dtuid,r.longitude,r.latitude,r.height,r.speed,r.direction,r.ctime"

+ " from cars c,car\_dtu\_relas cr, tank.remote\_cars r"

+ " where c.carid = cr.carid and cr.dtuid = r.dtuid"

+ " and c.state = 1 and r.state = 1" + sql

+ " and r.create\_time >= date\_sub(now(),interval 2 day)"

+ " order by r.id desc");

}

}

package com.yawa.home.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Company;

import com.yawa.core.model.User;

import com.yawa.model.Tank;

import com.yawa.util.SecurityContextUtil;@Before(SessionInterceptor.class)

public class ScaleController extends Controller { /\*\*

\* 模块：液位监控

\* 功能：跳转液位监控页面

\*/

public void index() {

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

setAttr("classify", user.getStr("classify"));

setAttr("company", Company.me.getById(user.getInt("company")));

render("index.jsp");

}

/\*\*

\* 模块：液位监控

\* 功能：储罐列表

\*/

public void getScales() {

Integer company = getParaToInt("company");

if(company == null){

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

company = user.getInt("company");

}

renderJson(Tank.me.getScales(getPara("name"),company));

}

}

package com.yawa.core.model;import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.util.model.Page;

import org.apache.commons.lang.StringUtils;import java.util.ArrayList;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;@SuppressWarnings("serial")

public class User extends Model<User> { public static final User me = new User(); public User getUserById(int id){

return findFirst("select \* from base\_users where id = ?",id);

} /\*\*

\* 根据账户和密码查找用户，登录时使用

\*

\* @param account

\* @param password

\* @return

\*/

public User getLoginUser(String account, String password) {

return findFirst("select \* from base\_users"

+ " where account = ? and password = md5(concat(?,salt))",

account, password);

}

/\* 根据手机查找用户 \*/

public User getLoginUser(int tele\_phone) {

return findFirst("select \* from ngmsg.base\_users where tele\_phone = ?", tele\_phone+"");

} /\*\*

\* 根据账户查找用户，发送验证码

\*

\* @param account

\* @return

\*/

public User getLoginUser(String account) {

return findFirst("select \* from base\_users where account = ?", account);

} /\*\*

\* 根据用户ID查找同公司下的用户(同级以及下级公司或部门)

\* @param id

\* @return

\*/

public List<User> getUserInOneCompany(int id){

return find("select a.\* from base\_users a,base\_companys b,base\_users c where a.org\_id = b.id and FIND\_IN\_SET(c.org\_id,b.path) and c.id = ?",id);

} /\*\*

\* 模块：角色管理

\* 功能：展示公司角色列表

\* @return

\*/

public List<User> getUsersByCompany(Integer company) {

String sql = "select \* from base\_users where company = ?";

return find(sql,company);

} /\*\*

\* 根据密码和id获取用户，用户校验原密码时使用

\*

\* @param id

\* @param password

\* @return

\*/

public User getUserByPass(Integer id, String password) {

return findFirst("select \* from base\_users where id = ?"

+ " and password = md5(concat(?,salt))", id, password);

} /\*\*

\* 过滤条件

\*

\* @param name

\* @return

\*/

private String filterCondition(String name, String account, String tel, Integer company) {

String sql = "";

if (StringUtils.isNotEmpty(name))

sql += " and t.name like '%" + name + "%'";

if (StringUtils.isNotEmpty(account))

sql += " and t.account like '%" + account + "%'";

if (StringUtils.isNotEmpty(tel))

sql += " and t.tele\_phone like '%" + tel + "%'";

if (company != null)

sql += " and t.company = " + company;

return sql;

} /\*\*

\* 判断用户账号是否重复

\*

\* @param user

\* @return

\*/

public boolean hasAccount(User user) {

String sql = "select count(1) cnt from base\_users where state = 1 and account = ?";

if (user.getInt("id") != null)

sql += " and id <> " + user.getInt("id");

return findFirst(sql, user.getStr("account")).getLong("cnt") > 0;

} /\* 判断手机账号是否重复 \*/

public boolean hasPhone(User user) {

//language=MySQL

String sql = "select count(1) cnt from ngmsg.base\_users where state =1 and tele\_phone = ?";

if (user.getInt("id") != null)

sql += " and id <> " + user.getInt("id");

return findFirst(sql,user.getStr("tele\_phone")).getLong("cnt")>0; }

/\* 更改用户名 \*/

public boolean updateName(User user,String name) { Record record = new Record();

record.set("id",user.get("id"));

record.set("name",name);

return Db.update("base\_users", record); }

/\* 更改用户头像 \*/

public boolean uploadHeader(User user,String header) {

Record record = new Record();

record.set("id",user.get("id")+"");

record.set("header",header);

return Db.update("base\_users", record);

}

/\* 更改用户签名 \*/

public boolean updateSignature(User user,String signature) {

Record record = new Record();

record.set("id",user.get("id"));

record.set("signature",signature);

return Db.update("base\_users", record);

}

/\* 更改用户密码 \*/

public boolean changePassword(User user,String oldpsw,String password) {

String sql = "select count(1) cnt from ngmsg.base\_users where account = ? and password = md5(concat(?,salt))"; if (findFirst(sql, user.getStr("account"),oldpsw).getLong("cnt") > 0) {

Record record = new Record();

record.set("id", user.get("id"));

record.set("password", password);

return Db.update("base\_users", record);

} else {

return false;

}

}

/\*\*

\* 分页列表

\*

\* @param name

\* @param page

\* @param limit

\* @return

\*/

public Page<User> getPages(String name, String account, String tel,

int company, Integer page, Integer limit) {

String sql = "select count(1) from ngmsg.base\_users t where t.state <> 0";

String cdtSQL = filterCondition(name, account, tel, company);

Long count = Db.queryLong(sql + cdtSQL);

sql = "select t.\*,org.name org\_name"

+ " from base\_users t"

+ " left join base\_orgs org on t.org\_id = org.id"

+ " where t.state <> 0 and org.state = 1" + cdtSQL;

sql += " order by t.id asc limit ?,?";

return new Page<User>(page, limit, count, find(sql, (page - 1) \* limit, limit));

} /\*\*

\* 模块：用户管理

\* 功能：用户系统角色授权,展示角色多选框

\*

\* @param user

\* @return

\*/

public Map<String, List<Role>> getUserRoles(Integer user, Integer ucompany, Integer logCompany) {

List<Role> roles = Role.me.find("select a.\*,"

+ " (select count(\*) from base\_user\_roles where userId = ? and roleId = a.id) checked"

+ " from (select r.\*,c.name company\_name from base\_roles r,base\_companys c"

+ " where r.company = c.id and r.state = 1 and c.state = 1 and (c.id = ? or c.id = ?)"

+ " union"

+ " select r.\*,c.name company\_name from base\_user\_roles ur,base\_roles r,base\_companys c"

+ " where ur.roleId = r.id and r.company = c.id and r.state = 1 and c.state = 1 and ur.userId = ?) a"

+ " order by company,id asc", user, ucompany, logCompany, user);

Map<String, List<Role>> roleMap = new LinkedHashMap<String, List<Role>>();

for (Role role : roles) {

String company = role.getStr("company\_name");

List<Role> subRoles = new ArrayList<Role>();

if (roleMap.containsKey(company))

subRoles = roleMap.get(company);

subRoles.add(role);

roleMap.put(company, subRoles);

}

return roleMap;

} /\*\*

\* 模块：用户管理

\* 功能：用户客户端角色授权,展示角色多选框

\*

\* @param user

\* @return

\*/

public List<Record> getUserAppRoles(Integer user) {

List<Record> roles = Db.find("select r.\*,"

+ "(select count(\*) from ngmsg.app\_user\_roles where userId = ? and roleId = r.id) checked"

+ " from ngmsg.app\_roles r where state = 1 order by id asc", user);

for (Record recd : roles) {

List<Module> modules = Module.me.find("SELECT m.\*,"

+ " (SELECT count(1) FROM ngmsg.app\_user\_modules WHERE userId = ? AND find\_in\_set(m.id,add\_modules)) added,"

+ " (SELECT count(1) FROM ngmsg.app\_user\_modules WHERE userId = ? AND find\_in\_set(m.id,rev\_modules)) reved"

+ " FROM ngmsg.app\_modules m,ngmsg.app\_role\_modules rm"

+ " WHERE m.state = 1 AND m.id = rm.moduleId AND rm.roleId = ?"

+ " ORDER BY m.sort ASC", user, user, recd.getInt("id")); recd.set("modules", modules);

}

return roles;

} /\*\*

\* 模块：用户管理

\* 功能：系统角色授权，展示未授权权限

\*

\* @param roles 角色列表

\* @return

\*/

public List<Right> getUserRights(Integer user, String roles) {

return Right.formatRights(Right.me.find("select t.\*,"

+ " (select count(\*) from base\_role\_rights where find\_in\_set(roleId, ?) and rightId = t.id) roleRight, "

+ " (select count(\*) from base\_user\_rights where userId = ? and rightId = t.id) userRight"

+ " from base\_rights t where state = 1 order by t.sort asc", roles, user));

} /\*\*

\* 功能：根据用户和权限组编码获取用户权限

\*

\* @param user

\* @param code

\* @return

\*/

public List<Right> getGroupRights(Integer user, String code) {

return Right.me.find("select r.\* from base\_rights r,base\_rights p"

+ " where r.parent = p.id and p.code = ? and r.state = 1 and p.state = 1"

+ " and r.id in( select rightId from base\_user\_rights where userId = ? union"

+ " select rightId from base\_user\_roles a,base\_role\_rights b"

+ " where a.roleId = b.roleId and a.userId = ?) order by r.sort asc", code, user, user);

} /\*\*

\* 保存用户角色关系

\*

\* @param roleids(,号分割)

\*/

public void saveUserRoles(Integer user, String roleids) {

if (StringUtils.isEmpty(roleids))

Db.update("delete from base\_user\_roles where userId = ?", user);

else {

Db.update("delete from base\_user\_roles where userId = ? and roleId not in"

+ "(select id from base\_roles where find\_in\_set(id,?))", user, roleids);

Db.update("insert into base\_user\_roles(userId,roleId)"

+ " select ? userId,id roleId from base\_roles where find\_in\_set(id,?)"

+ " and id not in(select roleId from base\_user\_roles where userId = ?)", user, roleids, user);

}

} /\*\*

\* 模块：用户管理

\* 功能：保存用户角色对应关系

\*

\* @param user

\* @param rightids

\*/

public void saveUserRights(Integer user, String rightids) {

if (StringUtils.isEmpty(rightids))

Db.update("delete from base\_user\_rights where userId = ?", user);

else {

Db.update("delete from base\_user\_rights where userId = ? and rightId not in"

+ "(select id from base\_rights where find\_in\_set(id,?))", user, rightids);

Db.update("insert into base\_user\_rights(userId,rightId)"

+ " select ? userId,id rightId from base\_rights where find\_in\_set(id,?)"

+ " and id not in(select rightId from base\_user\_rights where userId = ?)", user, rightids, user);

}

} /\*\*

\* 模块：用户管理

\* 功能：保存手机角色与用户关系

\*

\* @param user

\*/

public void saveAppUserRole(Integer user, String roleIds) {

if (StringUtils.isEmpty(roleIds))

Db.update("delete from app\_user\_roles where userId = ?", user);

else {

Db.update("delete from app\_user\_roles where userId = ? and roleId not in"

+ "(select id from app\_roles where find\_in\_set(id,?))", user, roleIds);

Db.update("insert into app\_user\_roles(userId,roleId)"

+ " select ? userId,id roleId from app\_roles where find\_in\_set(id,?)"

+ " and id not in(select roleId from app\_user\_roles where userId = ?)", user, roleIds, user);

}

} /\*\*

\* 模块：用户管理

\* 功能：保存用户与模块关系

\*

\* @param user

\* @param addIds

\* @param revIds

\*/

public void saveAppUserModules(Integer user, String addIds, String revIds) {

Db.update("delete from app\_user\_modules where userId = ?", user);

if (StringUtils.isNotEmpty(addIds) || StringUtils.isNotEmpty(revIds)) {

Db.update("insert into app\_user\_modules(userId,add\_modules,rev\_modules)"

+ " values(?,?,?)", user, addIds, revIds);

}

} /\*\*

\* 保存用户所选快捷菜单

\*

\* @param user

\* @param ids

\*/

public void saveShortCuts(Integer user, String ids) {

if (StringUtils.isEmpty(ids))

Db.update("delete from log\_menu\_clicks where userId = ?", user);

else {

Db.update("delete from log\_menu\_clicks where userId = ? and menuId not in"

+ "(select id from base\_menus where find\_in\_set(id,?))", user, ids);

Db.update("insert into log\_menu\_clicks(userId,menuId,clicknum)"

+ " select ? userId,id menuId,0 clicknum from base\_menus where find\_in\_set(id,?)"

+ " and id not in(select menuId from log\_menu\_clicks where userId = ?)", user, ids, user);

}

} /\*\*

\* 获取公司用户数

\*

\* @param company

\*/

public Long getTotal(Integer company) {

return Db.queryLong("select count(1) from base\_users t,base\_companys t1"

+ " where t.company = t1.id and t.state <> 0 and find\_in\_set(?, t1.path)", company);

} /\*\*

\* 根据组织ID 获取巡检人

\* @param orgid

\* @return

\*/

public List<User> getXJR(int orgid){

return find("SELECT account,name FROM base\_users WHERE org\_id = ?",orgid);

}}

package com.yawa.home.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.model.Tank;import java.util.List;/\*\*

\* 站点分布控制器

\*/

@Before(SessionInterceptor.class)

public class TankMapController extends Controller { /\*\*

\* 模块：站点分布 功能：跳转站点分布页面

\*/

public void index() {

render("index.jsp");

} /\*\*

\* 模块：站点分布 功能：展示公司储罐地图分布

\*/

public void getTanks() {

String company = getPara("company");

List<Tank> lt = Tank.me

.find("select t.id,t.name,t.location from base\_enterprise t"

+ " where t.type = 2 and parent = ? /\*tanks\*/ ",

company);

for(Tank tk : lt){

String location = tk.get("location");

String[] at = location.split(",");

if(at.length!=2){

continue;

}

String lonlat = at[1]+","+at[0];

tk.set("location",lonlat);

}

renderJson(lt);

}

}

package com.yawa.model;import java.util.List;import org.apache.commons.lang.StringUtils;import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.util.model.Page;@SuppressWarnings("serial")

public class Tank extends Model<Tank> { public static final Tank me = new Tank(); /\*\*

\* 模块：储罐配置 功能：展示储罐分页列表

\*/

public Page<Tank> getPages(String name, Integer company, Integer page,

Integer limit) {

String filter = "";

if (company != null)

filter += " and s.company = " + company;

if (StringUtils.isNotEmpty(name))

filter += " and t.name like '%" + name + "%'";

String sql = "select count(1) from tanks t "

+ " left join sites s on t.site\_id = s.id"

+ " left join base\_companys c on s.company = c.id"

+ " where t.state = 1" + filter;

Long count = Db.queryLong(sql);

sql = "select t.id,t.name,class.name classify,a.path\_name,t.address,tm.capacity cube,t.water,"

+ " (select group\_concat(name) from dict where group\_code = 'TANK.DEVICE' and find\_in\_set(code, t.device)) device,"

+ " tm.model, t.create\_date from tanks t"

+ " left join base\_areas a on t.area\_id = a.id"

+ " left join (select code,name from dict where group\_code = 'TANK.CLASSIFY' and type = 'I') class on t.classify = class.code"

+ " left join tank\_model tm on t.model = tm.id"

+ " left join sites s on t.site\_id = s.id"

+ " left join base\_companys c on s.company = c.id"

+ " where t.state = 1"

+ filter

+ " order by t.id asc limit ?,?";

return new Page<Tank>(page, limit, count, find(sql, (page - 1) \* limit,

limit));

} /\*\*

\* 模块：监控列表 功能：查询监控列表信息

\*

\* @return

\*/

public List<Record> getMonitors(Integer company) {

String filter = "";

if (company != null)

filter = " and v.company in (select id from base\_companys where find\_in\_set("

+ company + ",path))";

String sql = "select v.id,cast(volume as char) volumeo,v.name,v.scale\_rate,v.volume,d.tank\_scale,d.pressure,d.ext\_press\_1,d.ext\_press\_2,d.ext\_temp,"

+ " d.ext\_flow,d.flow\_bk,d.flow\_gk,date\_format(d.create\_date,'%m%d %H:%i:%s') create\_date,v.alert\_state,ta.field,ta.sound,ta.sound\_field,"

+ " (case when avl.value is null or rtd.avgUsed = 0 then '~' else format((avl.value / rtd.avgUsed),2) end) forcast,"

+ " (select color from tank\_section where category = 'scales' and code = v.scale\_level ) scale\_color,"

+ " (case when v.plc\_state is null and v.llj\_state is null then '#949FB1'"

+ " when v.plc\_state is null then '#008000'"

+ " when v.alert\_state = 1 then '#800000' else (select color from tank\_section where category = 'work\_state' and code = v.plc\_state) end) back\_color,"

+ " (case when v.llj\_state = 'work' or v.llj\_state is null then 'white' end) font\_color"

+ " from view\_tank\_verify v"

+ " left join tank\_dynamic d on v.id = d.tank\_id"

+ " left join (select tank\_id,value from tank\_attrs where attr\_id = '66') avl on v.id = avl.tank\_id"

+ " left join (select tank\_id,group\_concat(b.field) field,sum(a.sound) sound,group\_concat(case when a.sound = 1 then b.field end) sound\_field"

+ " from tank\_alerts a,core\_attrs b"

+ " where a.attr\_id = b.attr\_id and a.state = 0 and b.field is not null group by a.tank\_id) ta on v.id = ta.tank\_id"

+ " left join (select tank\_id,avg(sum\_used) / 1.43 avgUsed from report\_tank\_date t"

+ " where stat\_date >= date\_format(date\_sub(now(), interval (select value from app\_base where code = 'expect\_days') day),'%Y-%m-%d')"

+ " group by tank\_id) rtd on v.id = rtd.tank\_id"

+ " where 1=1 " + filter + " order by v.alert\_state desc";

return Db.find(sql);

} /\*\*

\* 模块：首页 功能：当前公司储罐数量

\*

\* @param company

\* @return

\*/

public Long getTotal(Integer company) {

return Db

.queryLong(

"select count(1) from sites t,tanks t1,sites t2,base\_companys t3"

+ " where t.id = t1.site\_id and t1.site\_id = t2.id and t2.company = t3.id and t.state = 1 and t1.state = 1"

+ " and find\_in\_set(?, t3.path)", company);

} /\*\*

\* 液罐详情

\*

\* @param id

\* @return

\*/

public Tank getById(Integer id) {

String sql = "select t.\*,ar.path\_name"

+ " from tanks t left join base\_areas ar on t.area\_id = ar.id"

+ " where t.id = ?";

return findFirst(sql, id);

} /\*\*

\* 模块：首页领导视图 功能：获取公司储罐数量

\*

\* @param int1

\* @return

\*/

public List<Record> getTankCharts(Integer company) {

return Db

.find("select t2.id,t2.name,count(\*) cnt from tanks t,sites t1,base\_companys t2"

+ " where t.site\_id = t1.id and t1.company = t2.id and t.state = 1 and t1.state = 1"

+ " and find\_in\_set(?, t2.path) group by t2.id,t2.name",

company);

} /\*\*

\* 模块：液位监控 功能：获取储罐液位数据

\*

\* @param company

\* @return

\*/

public List<Record> getScales(String name, Integer company) {

String filter = "";

if (StringUtils.isNotEmpty(name))

filter = " and t.name like '%" + name + "%'";

return Db

.find("select t.id,t.name,t1.tank\_scale,t.alert\_state,"

+ " CONVERT(concat((100-scale\_rate) \* 0.64 + 18,'%') USING utf8) scale\_rate,t1.pressure,t1.ext\_flow from view\_tank\_verify t"

+ " left join tank\_dynamic t1 on t.id = t1.tank\_id"

+ " where t.company = ?" + filter, company);

}

}

package com.yawa.inspection.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.model.\*;

import org.apache.commons.lang.StringUtils;

import sun.misc.BASE64Decoder;import javax.servlet.ServletOutputStream;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;/\*\*

\* Created by bluster on 2017/5/14.

\*/

@Before(SessionInterceptor.class)

public class InspectionController extends Controller{

public void taskDetail(){

String noframe = "false";

if(StringUtils.isNotEmpty(getPara("noframe"))){

noframe = "true";

}

setAttr("noframe",noframe);

Integer id = getParaToInt("id");

InspectionTask it = InspectionTask.me.getInspectionTask(id);

String entId = it.get("enterprise");

String stnId = it.get("station");

Enterprise enterprise = Enterprise.me.getEnterprise(entId);

Enterprise station = Enterprise.me.getEnterprise(stnId);

setAttr("station",station);

setAttr("enterprise",enterprise);

setAttr("task",it);

List<InspectionTaskDetail> lTaskDetail = InspectionTaskDetail.me.getTaskDetailsByTaskId(id);

List<InspectionType> lType = InspectionType.me.getListInspectionType();

Map<Integer,List<InspectionTaskDetail>> mlTaskDetail = new HashMap<Integer,List<InspectionTaskDetail>>();

for(InspectionType iType : lType){

Integer typeId = iType.get("id");

//String typeTitle = iType.get("title");

mlTaskDetail.put(typeId,new ArrayList<InspectionTaskDetail>());

}

for(InspectionTaskDetail taskDetail:lTaskDetail){

Integer typeId = taskDetail.get("type");

mlTaskDetail.get(typeId).add(taskDetail);

}

setAttr("mlTaskDetail",mlTaskDetail);

setAttr("lType",lType); List<InspectionTaskImg> taskImgList = InspectionTaskImg.me.getTaskImgSmallByTask(id);

setAttr("taskImgList",taskImgList);

//setAttr("mTypeTitle",mTypeTitle);

render("taskDetail.jsp");

} public void getImgBig() {

// String enterprise\_id = getPara("enterprise\_id");

// String field\_code = getPara("field\_code");

try {

Integer id = getParaToInt(0);

InspectionTaskImg timg = InspectionTaskImg.me.getTaskImgBigById(id);

String imgStr = timg.get("img\_big");

// imgStr = imgStr.substring(23);

imgStr = imgStr.substring(imgStr.indexOf(',')+1);

BASE64Decoder decoder = new BASE64Decoder();// Base64解码

byte[] bytes = decoder.decodeBuffer(imgStr);

for (int i = 0; i < bytes.length; ++i) {

if (bytes[i] < 0) {// 调整异常数据

bytes[i] += 256;

}

}

// 生成jpeg图片 ServletOutputStream sos = getResponse().getOutputStream();

sos.write(bytes);

sos.flush();

sos.close();

} catch (Exception e) {

e.printStackTrace();

}

renderNull();

}

}

package com.yawa.model;import com.jfinal.plugin.activerecord.Model;import java.util.List;/\*\*

\* Created by bluster on 2017/5/15.

\*/

@SuppressWarnings("serial")

public class InspectionTaskImg extends Model<InspectionTaskImg> {

public static final InspectionTaskImg me = new InspectionTaskImg();

public List<InspectionTaskImg> getTaskImgSmallByTask(int taskId){

return find("select id,task\_id,img from base\_inspection\_task\_img where task\_id = ? ",taskId);

}

public List<InspectionTaskImg> getTaskImgByTask(int taskId){

return find("select id,task\_id,img,img\_big from base\_inspection\_task\_img where task\_id = ? ",taskId);

}

public InspectionTaskImg getTaskImgSmallById(int Id){

return findFirst("select id,task\_id,img from base\_inspection\_task\_img where id = ? ",Id);

}

public List<InspectionTaskImg> getTaskImgBigByTask(int taskId){

return find("select id,task\_id,img\_big from base\_inspection\_task\_img where task\_id = ? ",taskId);

}

public InspectionTaskImg getTaskImgBigById(int Id){

return findFirst("select id,task\_id,img\_big from base\_inspection\_task\_img where id = ? ",Id);

}

}

package com.yawa.news.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.yawa.core.model.New;

import com.yawa.model.Notice;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.ResponseData;

import org.apache.commons.lang.StringUtils;import java.io.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.UUID;/\*\*

\*

\* @author wwl

\*

\*/

//@Before(SessionInterceptor.class)

public class NewsCKController extends Controller { /\*\*

\* 查看新闻内容

\*/

public void chakan() {

//累计点击量

String id = getPara("id");

if(null!=id && !"".equals(id)){

New.me.upClicks(id);

}

String fname = getPara("fname");

render(fname+".html");

}}

package com.yawa.core.model;import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.yawa.util.model.ConstantItems;

import com.yawa.util.model.Page;import java.util.List;@SuppressWarnings("serial")

public class New extends Model<New> { public static final New me = new New(); /\*\*

\* 创建新闻

\*

\* @param title

\* @param content

\* @param url

\*/

public int addnews(String title, String content, String url, int uid, String oname, String nname, String ftype, String img) {

return Db.update("insert into news(title,content,url,clicks,creator,create\_date,update\_date,state,oname,nname,ftype,img) values (?,?,?,0,?,NOW(),NOW(),2,?,?,?,?)", title, content, url, uid, oname, nname, ftype, img);

} /\*\*

\* 修改新闻

\*

\* @param title

\* @param content

\* @param url

\* @param id

\* @return

\*/

public int editnews(String title, String content, String url, int id, String oname, String nname, String ftype) {

String sql = "update news set title = ?,content = ?,url = ? ";

if (null != oname && !"".equals(oname)) {

sql += " ,oname = '" + oname + "'";

}

if (null != nname && !"".equals(nname)) {

sql += " ,nname = '" + nname + "'";

}

if (null != ftype && !"".equals(ftype)) {

sql += " ,ftype = '" + ftype + "'";

}

sql += " where id=?";

return Db.update(sql, title, content, url, id);

} /\*\*

\* 更新点击量

\*

\* @param id

\*/

public void upClicks(String id) {

Db.update("update news set clicks = clicks+1 where id=?", id);

} /\*\*

\* 删除新闻

\*

\* @param id

\* @return

\*/

public int deleteNews(int id) {

return Db.update("delete from news where id = ?", id);

} /\*\*

\* 新闻列表

\*

\* @param bt

\* @param zt

\* @return

\*/

public List<New> getList(String bt, String zt) {

String sql = "select id,title,content,url,clicks,creator,create\_date,state from news where 1=1";

if (null != bt && !"".equals(bt)) {

sql += " and title like '%" + bt + "%'";

}

if (null != zt && !"".equals(zt)) {

sql += " and state = " + zt;

}

sql += " order by create\_date desc ";

return find(sql);

} /\*新闻列表\*/

public List<New> getList() {

//language=MySQL

String sql = "SELECT \* FROM ngmsg.news ORDER BY create\_date DESC ";

return find(sql);

} /\*新闻列表 \*/

public List<New> getList(String top) {

//language=MySQL

String sql = "select \* from ngmsg.news ORDER BY create\_date DESC LIMIT " + top; return find(sql);

} /\*新闻列表 \*/

public List<New> getListCollect(String collects, String top) {

//language=MySQL

String sql = "select \* from ngmsg.news WHERE id IN ("+collects+") ORDER BY create\_date DESC LIMIT " + top; return find(sql);

} public List<New> getInfo(int id) {

//language=MySQL

String sql = "SELECT \* FROM ngmsg.news WHERE id = ?";

return find(sql, id);

} /\*\*

\* 新闻分页列表

\*

\* @return

\*/

public List<New> getPageNews(Integer company, Integer page, Integer limit) {

String filter = "";

if (company != null) {

filter = " and a.company = " + company;

}

return find("select a.id,a.title,a.summary,"

+ " concat((select value from app\_base where code = 'access\_path'),'/',b.attach\_type,'/',a.id,'/',b.store\_name) url"

+ " from app\_news a "

+ " left join (select attach\_id,attach\_type,store\_name from attaches where attach\_type = ?) b on a.id = b.attach\_id"

+ " where a.state = 'R'" + filter + " order by a.create\_date desc limit ?,?",

ConstantItems.APP\_NEWS, (page - 1) \* limit, limit);

}

/\*\*

\* 获取新闻详情

\*

\* @param id

\* @return

\*/

public New getById(Integer id) {

return findFirst("select a.id,a.title,a.summary,a.content,b.name creator,"

+ " date\_format(a.release\_date,'%Y-%m-%d %H:%i:%s') release\_date"

+ " from app\_news a left join base\_users b on a.creator = b.id"

+ " where a.id = ?", id);

} /\*\*

\* 新增分页列表

\*

\* @param page

\* @param limit

\* @return

\*/

public Page<New> getPages(Integer company, Integer page, Integer limit) {

String filter = "";

if (company != null) {

filter = " and t.company = " + company;

}

String sql = "select count(1) from app\_news t where t.state = 'R'" + filter;

Long count = Db.queryLong(sql);

return new Page<New>(page, limit, count, getPageNews(company, page, limit));

}

}

package com.yawa.news.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.jfinal.upload.UploadFile;

import com.yawa.common.PageService;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.New;

import com.yawa.model.Notice;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.ResponseData;

import org.apache.commons.lang.StringUtils;import javax.servlet.ServletOutputStream;

import java.io.\*;

import java.net.URLEncoder;

import java.nio.channels.FileChannel;

import java.util.ArrayList;

import java.util.List;

import java.util.UUID;/\*\*

\*

\* @author wwl

\*

\*/

@Before(SessionInterceptor.class)

public class NewsController extends Controller { public void index() {

String biaoti = getPara("biaoti");

String zhuangtai = getPara("zhuangtai");

setAttr("biaoti",biaoti);

setAttr("zhuangtai",zhuangtai);

if("审批驳回".equals(zhuangtai)){

zhuangtai = "0";

}else if("审批中".equals(zhuangtai)){

zhuangtai = "1";

}else if("审批通过".equals(zhuangtai)){

zhuangtai = "2";

}else{

zhuangtai = "";

}

PageService<New> ps = new PageService<New>();

setAttr("newList",ps.getPagedList(this,New.me.getList(biaoti,zhuangtai),"pageCnt","pageNo"));

render("index.jsp");

} /\*\*

\* 跳转到新增页面

\*/

public void addnews() {

render("addnews.jsp");

}

/\*\*

\* 跳转到修改页面

\*/

public void toUpdate() {

String id = getPara("id");

String[] info = id.split("\_");

New n = New.me.getInfo(Integer.parseInt(info[0])).get(0);

String title = "<h1>"+n.get("title")+"</h1>";

String neirong = getFileContent(info[1]);

neirong = neirong.replace(title,"");

setAttr("neirong",neirong);//文件内容

setAttr("news",n);

render("updatenews.jsp");

} /\*\*

\* 查看新闻内容

\*/

/\*public void chakan() {

String fname = getPara("fname");

render(fname+".html");

}\*/ /\*\*

\* 添加新闻

\* @return

\*/

@Before(POST.class)

public void addNewFile(){

String title = getPara("title");

String content = getPara("content");

String text = getPara("neirong");

int uid = SecurityContextUtil.getLoginUserId(getRequest());

UUID uuid = UUID.randomUUID();

boolean bool = createFile(uuid+"",text); String img = getPara("img");

System.out.println("--------");

System.out.println(img);

System.out.println("--------");

if(bool){

//新增上传附件

String oname = getPara("oname");

String nname = getPara("nname");

String ftype = getPara("ftype");

//创建数据记录

int ret = New.me.addnews(title,content,uuid.toString(),uid,oname,nname,ftype,img);

if(ret>0){

renderJson(new ResponseData(true, "提交成功!"));

}else{

renderJson(new ResponseData(false, "操作失败!"));

}

} } public void looknews(){

Integer id = getParaToInt("id");

try {

New n = New.me.getInfo(id).get(0);

setAttr("news",n);

}catch(Exception ex){

ex.printStackTrace();

} render("looknews.jsp");

} public void downAttr(){

try {

String fpath = getRequest().getSession().getServletContext().getRealPath("/")+"fileUpload";

Integer id = getParaToInt("id");

//Shares s = Shares.me.getShares(id);

New n = New.me.getInfo(id).get(0);

String saveName = n.get("nname");

saveName += ("."+n.get("ftype"));

String oriName = n.get("oname"); getResponse().setContentType("multipart/form-data");

String filename = URLEncoder.encode(oriName, "UTF-8");

getResponse().setHeader("Content-Disposition", "attachment;filename="+filename); ServletOutputStream sos=getResponse().getOutputStream();

InputStream fi= new FileInputStream(new File(fpath+"/"+saveName));

byte[] buffer=new byte[1024];

int bytesRead;

while(-1 != (bytesRead = fi.read(buffer, 0, buffer.length))) {

sos.write(buffer,0,bytesRead);

}

sos.close();

fi.close();

} catch (Exception e) {

}

renderNull();

}

/\*\*

\* 上传附件

\*/

public void shangchuan(){

List<UploadFile> lf = this.getFiles();

//List<String> oldfile = new ArrayList<String>();

//List<String> newfile = new ArrayList<String>();

for(UploadFile file : lf){//为改为多文件上传提供方便，虽然现在是单个文件上传

//System.out.println(file.getOriginalFileName());

String o\_name = file.getOriginalFileName();

//oldfile.add(o\_name);

String type = o\_name.substring(o\_name.lastIndexOf(".")+1);

String n\_name = UUID.randomUUID().toString();

//newfile.add(n\_name);

if(writeFile(file,n\_name+"."+type)){

renderJson(new ResponseData(true, o\_name+","+n\_name+","+type));

}else{

renderJson(new ResponseData(false, "上传失败!"));

}

}

} /\*\*

\* 编辑新闻内容

\*/

@Before(POST.class)

public void editNewFile(){

String title = getPara("title");

String content = getPara("content");

String text = getPara("neirong");

String fname = getPara("fname");

String id = getPara("id");

//删除

boolean bool = delFile(fname);

if(!bool){//删除失败就创建新的文件

UUID uuid = UUID.randomUUID();

fname = uuid.toString();

}

//新建

bool = createFile(fname+"",text);

if(bool){

//新增上传附件

String oname = getPara("oname");

String nname = getPara("nname");

String ftype = getPara("ftype");

//修改数据记录

int ret = New.me.editnews(title,content,fname,Integer.parseInt(id),oname,nname,ftype);

if(ret>0){

renderJson(new ResponseData(true, "修改成功!"));

}else{

renderJson(new ResponseData(false, "操作失败!"));

}

}

} /\*\*

\* 创建文件

\* @param fileName

\* @param filecontent

\* @return

\*/

public boolean createFile(String fileName,String filecontent){

Boolean bool = false;

File file =new File(getRequest().getSession().getServletContext().getRealPath("/")+"newsJSP");

//如果文件夹不存在则创建

if(!file .exists() && !file .isDirectory())

{

file .mkdir();

}

//文件路径+名称+文件类型

String filenameTemp = getRequest().getSession().getServletContext().getRealPath("/")+"newsJSP\\"+fileName+".html";

file = new File(filenameTemp);

try {

//如果文件不存在，则创建新的文件

if(!file.exists()){

file.createNewFile();

bool = true;

System.out.println("success create file,the file is "+filenameTemp);

//创建文件成功后，写入内容到文件里

writeFileContent(filenameTemp, filecontent);

}

} catch (Exception e) {

e.printStackTrace();

} return bool;

} /\*\*

\* 向文件中写入内容

\* @param filepath 文件路径与名称

\* @param newstr 写入的内容

\* @return

\* @throws IOException

\*/

private boolean writeFileContent(String filepath,String newstr) throws IOException{

Boolean bool = false;

String filein = newstr+"\r\n";//新写入的行，换行

String temp = ""; FileInputStream fis = null;

InputStreamReader isr = null;

BufferedReader br = null;

FileOutputStream fos = null;

PrintWriter pw = null;

try {

File file = new File(filepath);//文件路径(包括文件名称)

//将文件读入输入流

fis = new FileInputStream(file);

isr = new InputStreamReader(fis);

br = new BufferedReader(isr);

StringBuffer buffer = new StringBuffer(); //文件原有内容

for(int i=0;(temp =br.readLine())!=null;i++){

buffer.append(temp);

// 行与行之间的分隔符 相当于“\n”

buffer = buffer.append(System.getProperty("line.separator"));

}

buffer.append(filein); fos = new FileOutputStream(file);

pw = new PrintWriter(fos);

pw.write(buffer.toString().toCharArray());

pw.flush();

bool = true;

} catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}finally {

//不要忘记关闭

if (pw != null) {

pw.close();

}

if (fos != null) {

fos.close();

}

if (br != null) {

br.close();

}

if (isr != null) {

isr.close();

}

if (fis != null) {

fis.close();

}

}

return bool;

} /\*\*

\* 读取文件内容

\* @param fname

\* @return

\*/

public String getFileContent(String fname)

{

String fileContent = "";

try

{

File f = new File(getRequest().getSession().getServletContext().getRealPath("/")+"newsJSP\\"+fname+".html");

if(f.isFile()&&f.exists())

{

InputStreamReader read = new InputStreamReader(new FileInputStream(f),"GB2312");

BufferedReader reader=new BufferedReader(read);

String line;

while ((line = reader.readLine()) != null)

{

fileContent += line;

}

read.close();

}

} catch (Exception e)

{

e.printStackTrace();

}

return fileContent;

} /\*\*

\* 删除文件

\* @param fname 文件名称

\* @return

\*/

public boolean delFile(String fname){

Boolean bool = false;

String filenameTemp = getRequest().getSession().getServletContext().getRealPath("/")+"newsJSP\\"+fname+".html";

File file = new File(filenameTemp);

try {

if(file.exists()){

file.delete();

bool = true;

}

} catch (Exception e) {

// TODO: handle exception

}

return bool;

}

/\*\*

\* 删除新闻

\* @return

\*/

@Before(POST.class)

public void deleteNews(){

int index = 0;

String ids = getPara("ids");

if(ids.length()>0){

String[] id = ids.split(",");

for(String s : id){

String[] str = s.split("\_");

//删除数据

int ret = New.me.deleteNews(Integer.parseInt(str[0]));

//删除文件

if(ret>0){

delFile(str[1]);

}else{

index++;

}

}

if(index == 0){

renderJson(new ResponseData(true, "操作成功!"));

}else{

renderJson(new ResponseData(false, "有"+index+"条数据删除失败!"));

}

}

} //-------------------工具类--------------------------- public boolean writeFile(UploadFile ufile,String fname){

String fpath = getRequest().getSession().getServletContext().getRealPath("/")+"fileUpload";

File file1 =new File(fpath);

//如果文件夹不存在则创建

if(!(file1 .exists() && file1 .isDirectory()))

{

file1 .mkdir();

}

File file = ufile.getFile();

File t = new File(fpath+"/"+fname);

try {

t.createNewFile();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

return false;

}

FileInputStream fi = null; FileOutputStream fo = null; FileChannel in = null; FileChannel out = null; try { fi = new FileInputStream(file); fo = new FileOutputStream(t); in = fi.getChannel();// 得到对应的文件通道 out = fo.getChannel();// 得到对应的文件通道 in.transferTo(0, in.size(), out);// 连接两个通道，并且从in通道读取，然后写入out通道 } catch (IOException e) { e.printStackTrace(); } finally { try { fi.close(); in.close(); fo.close(); out.close(); } catch (IOException e) { e.printStackTrace();

return false;

} }

return true;

}

//----------------------------------------------------------------//

public void notice() {

//User user = User.me.findById(SecurityContextUtil

// .getLoginUserId(getRequest()));

String qrydate = getPara("qrydate");

String startQryDate = getPara("startqrydate");

String endQryDate = getPara("endqrydate");

List<String> lf = new ArrayList<String>();

setAttr("notices", Notice.me.getNotices(qrydate,startQryDate,endQryDate));

render("notice.jsp");

} public void newnotice(){

String noticeid = getPara("noticeid");

String busi = "new";

if(StringUtils.isNotEmpty(noticeid)){

busi = "edit";

}

setAttr("busi",busi);

String title = "";

String content = ""; if("edit".equals(busi)){//查找对应记录，进行展示

setAttr("notice",Notice.me.getNotice(noticeid));

}

render("newnotice.jsp");

} public void addNotice(){

try{

if(Notice.me.addNotice(getPara("title"),getPara("content"),getPara("color"))==1){

renderJson(new ResponseData(true, "监管通知新增成功!"));

}

else{

renderJson(new ResponseData(false, "监管通知新增失败!"));

}

}catch(Exception ex){

renderJson(new ResponseData(false, "监管通知新增失败!"));

}

} public void editNotice(){

try{

if(Notice.me.editNotice(getPara("title"),getPara("content"),getPara("color"),getPara("noticeid"))==1){

renderJson(new ResponseData(true, "监管通知编辑成功!"));

}

else{

renderJson(new ResponseData(false, "监管通知编辑失败!"));

}

}catch(Exception ex){

renderJson(new ResponseData(false, "监管通知编辑失败!"));

}

}

}

package com.yawa.task.controller;import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.jfinal.ext.interceptor.POST;

import com.yawa.common.PageService;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Org;

import com.yawa.core.model.User;

import com.yawa.model.\*;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.ResponseData;

import org.apache.commons.lang.StringUtils;

import sun.misc.BASE64Decoder;import javax.servlet.ServletOutputStream;

import java.util.List;/\*\*

\*

\* @author wwl

\* 巡检任务控制器

\*/

@Before(SessionInterceptor.class)

public class TaskController extends Controller { /\*\*

\*

\*/

public void index() {

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

SupervisionTask st = new SupervisionTask();

st.set("executer",user.get("id"));

st.set("creator",user.get("id"));

st.set("title",getPara("title"));

st.set("stat",getPara("stat"));

List<SupervisionTask> lst = SupervisionTask.me.getListSupervisionTask(st);

for(SupervisionTask task : lst){ int executer\_id = 0;

try {

executer\_id = task.get("executer");

} catch (Exception e) { }

User executer = User.me.getUserById(executer\_id);

task.put("executername",executer.get("name"));

task.put("executerphone",executer.get("tele\_phone"));

int stat = Integer.parseInt(task.getStr("stat"));

String statname = "";

switch (stat){

case 0:

statname = "待处理";

break;

case 1:

statname = "处理中";

break;

case 2:

statname = "处理完成";

break;

default:

statname = "未知";

break;

}

task.put("statname",statname); }

PageService<SupervisionTask> ps = new PageService<SupervisionTask>();

setAttr("taskList",ps.getPagedList(this,lst,"pageCnt","pageNo"));

render("index.jsp");

}

public void inspectionIndex() {

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

//InspectionTask it = getModel(InspectionTask.class,"it");

InspectionTask it = new InspectionTask();

String noframe = getPara("noframe");

if(StringUtils.isNotEmpty(noframe)&&"true".equals(noframe)){

setAttr("noframe","true");

}else{

setAttr("noframe","false");

}

it.set("title",getPara("title"));

it.set("stat",getPara("stat"));

it.set("station",getPara("station"));

List<InspectionTask> lit = InspectionTask.me.getListInspectionTask(it);

for(InspectionTask inspTask : lit){

String enterprise\_id = inspTask.getStr("enterprise");

String stations\_id = inspTask.getStr("station");

String inspector\_account = inspTask.getStr("inspector");

Enterprise ent = Enterprise.me.getEnterprise(enterprise\_id);

List<Enterprise> lstation = Enterprise.me.getEnterpriseListByInStr(stations\_id);

User inspector = User.me.getLoginUser(inspector\_account);

inspTask.put("starttime",inspTask.get("starttime").toString().substring(0,10));

inspTask.put("endtime",inspTask.get("endtime").toString().substring(0,10));

inspTask.put("enterprise\_name",ent.get("name"));

inspTask.put("inspector\_name",inspector.get("name"));

StringBuffer sbStationName = new StringBuffer();

for(int i = 0 ; i < lstation.size() ; i++){

if(i!=0){

sbStationName.append(",");

}

sbStationName.append((lstation.get(i)).getStr("name"));

}

inspTask.put("station\_name",sbStationName.toString());

String stat = inspTask.get("stat").toString();

String statName = "";

if(stat.equals("0")){

statName = "待处理";

}else if(stat.equals("1")){

statName = "巡检完成";

}else if(stat.equals("2")){

statName = "逾期";

}else if(stat.equals("3")){

statName = "待复查";

}

inspTask.put("stat\_name",statName);

}

PageService<InspectionTask> ps = new PageService<InspectionTask>();

setAttr("listInsp",ps.getPagedList(this,lit,"pageCnt","pageNo")); render("inspectionIndex.jsp");

}

public void inspectionTaskList() {

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

//InspectionTask it = getModel(InspectionTask.class,"it");

InspectionTask it = new InspectionTask();

String noframe = getPara("noframe");

if(StringUtils.isNotEmpty(noframe)&&"true".equals(noframe)){

setAttr("noframe","true");

}else{

setAttr("noframe","false");

}

String statr = getPara("stat");

if(StringUtils.isEmpty(statr)){

statr = "1,3";

}

it.set("title",getPara("title"));

it.set("stat",statr);

it.set("station",getPara("station"));

List<InspectionTask> lit = InspectionTask.me.getListInspectionTask(it);

for(InspectionTask inspTask : lit){

String enterprise\_id = inspTask.getStr("enterprise");

String stations\_id = inspTask.getStr("station");

String inspector\_account = inspTask.getStr("inspector");

Enterprise ent = Enterprise.me.getEnterprise(enterprise\_id);

List<Enterprise> lstation = Enterprise.me.getEnterpriseListByInStr(stations\_id);

User inspector = User.me.getLoginUser(inspector\_account);

inspTask.put("starttime",inspTask.get("starttime").toString().substring(0,10));

inspTask.put("endtime",inspTask.get("endtime").toString().substring(0,10));

inspTask.put("enterprise\_name",ent.get("name"));

inspTask.put("inspector\_name",inspector.get("name"));

StringBuffer sbStationName = new StringBuffer();

for(int i = 0 ; i < lstation.size() ; i++){

if(i!=0){

sbStationName.append(",");

}

sbStationName.append((lstation.get(i)).getStr("name"));

}

inspTask.put("station\_name",sbStationName.toString());

String stat = inspTask.get("stat").toString();

String statName = "";

if(stat.equals("0")){

statName = "待处理";

}else if(stat.equals("1")){

statName = "巡检完成";

}else if(stat.equals("2")){

statName = "逾期";

}else if(stat.equals("3")){

statName = "待复查";

}

inspTask.put("stat\_name",statName);

}

PageService<InspectionTask> ps = new PageService<InspectionTask>();

setAttr("listInsp",ps.getPagedList(this,lit,"pageCnt","pageNo")); render("inspectionTaskList.jsp");

}

public void addInspection() {

int uid = SecurityContextUtil.getLoginUserId(getRequest());

//以组织树显示 上下级

//List<User> inspectorList = User.me.getUserInOneCompany(uid);

//setAttr("inspectorList",inspectorList);

List<Org> olist = Org.me.getlist(uid);

setAttr("olist",olist);

//获取当前登录用户所在集团片区的企业列表

List<Enterprise> lent = Enterprise.me.getEnterpriseList(1,uid);

setAttr("enterpriseList",lent); render("addInspection.jsp");

}

/\*\*

\* 根据组织ID 获取巡检人列表

\*/

@Before(POST.class)

public void getXJR(){

String oid = getPara("id");

List<User> xjr = User.me.getXJR(Integer.parseInt(oid));

renderJson(xjr);

} public void addSupervision(){

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

List<User> executorList = User.me.getUserInOneCompany(user.getInt("id"));

setAttr("executorList",executorList);

render("addSupervision.jsp");

}

public void viewSupervision(){

Integer id = getParaToInt("id");

if(id == null){

renderText("id为空，失败");

return;

}

SupervisionTask st = SupervisionTask.me.getSupervisionTask(id);

Integer executerId = st.get("executer");

User executer = User.me.findById(executerId);

st.put("executerName",executer.get("name"));

setAttr("task",st);

render("viewSupervision.jsp");

}

public void editSupervision(){

Integer id = getParaToInt("id");

if(id == null){

renderText("id为空，失败");

return;

}

SupervisionTask st = SupervisionTask.me.getSupervisionTask(id);

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

List<User> executorList = User.me.getUserInOneCompany(user.getInt("id"));

setAttr("executorList",executorList);

setAttr("task",st);

render("editSupervision.jsp");

}

public void openInspection(){

String readOnly = getPara("readOnly");

Integer id = getParaToInt("id");

if(id == null){

renderText("id为空，失败");

return;

}

InspectionTask it = InspectionTask.me.getInspectionTask(id);

String inspector\_account = it.get("inspector");

User inspector = User.me.getLoginUser(inspector\_account); if(StringUtils.isNotEmpty(readOnly)&&readOnly.equals("true")){

setAttr("readOnly","true");

int inspector\_id = inspector.get("id");

InspectionTaskHistory ith = InspectionTaskHistory.me.getInspectionTaskHistoryByCreatorIdAndTaskId(inspector\_id,id);

setAttr("taskHistory",ith);

}

else{

setAttr("readOnly","false");

if(!it.get("stat").toString().equals("0")){

renderText("只有在待审核状态才能编辑");

return;

}

} setAttr("inspector",inspector);

setAttr("task",it); //组织树列表

int uid = SecurityContextUtil.getLoginUserId(getRequest());

List<Org> olist = Org.me.getlist(uid);

setAttr("olist",olist);

//需要知道选中巡检人所在的组织ID

int oid = inspector.get("org\_id");

setAttr("orgId",oid);

//当前组织下的巡检人列表

List<User> xjr = User.me.getXJR(oid);

setAttr("xjr",xjr);

String creatorAccount = it.get("creator");

int creatorId = User.me.getLoginUser(creatorAccount).get("id");

List<User> inspectorList = User.me.getUserInOneCompany(creatorId); User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest())); setAttr("inspectorList",inspectorList); String enterprise\_id = it.get("enterprise");

String station\_id = it.get("station"); Enterprise enterprise = Enterprise.me.getEnterprise(enterprise\_id);

Enterprise station = Enterprise.me.getEnterprise(station\_id); setAttr("enterprise",enterprise);

setAttr("station",station); //List<Enterprise> lent = Enterprise.me.getEnterpriseList();

//setAttr("enterpriseList",lent); render("openInspection.jsp");

} public void editInspectionTask(){

try{

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

InspectionTask it = new InspectionTask();

int id = getParaToInt("id");

it.set("id",id);

String title = getPara("title");

if(StringUtils.isNotEmpty(title))

it.set("title",title);

String starttime = getPara("starttime");

if(StringUtils.isNotEmpty(starttime))

it.set("starttime",starttime);

String endtime = getPara("endtime");

if(StringUtils.isNotEmpty(endtime))

it.set("endtime",endtime);

String inspector = getPara("inspector");

if(StringUtils.isNotEmpty(inspector))

it.set("inspector",inspector); if(it.update()){

renderJson(new ResponseData(true, "巡检任务编辑成功!"));

}

else{

renderJson(new ResponseData(false, "巡检任务编辑失败!")); }

}catch(Exception ex){

renderJson(new ResponseData(false, "巡检任务编辑失败!"));

}

}

/\*\*

\* 编辑督办任务

\*/

public void editSupervisionTask(){

try {

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

SupervisionTask st = new SupervisionTask();

st.set("id",getParaToInt("id"));

st.set("title", getPara("title"));

st.set("content", getPara("content"));

st.set("createtime", getPara("createtime"));

st.set("executer", getPara("executer"));

st.set("stat", getPara("stat"));

if (st.update()) {

renderJson(new ResponseData(true, "督办任务修改成功!"));

} else {

renderJson(new ResponseData(false, "督办任务修改失败!"));

}

}catch(Exception ex){

ex.printStackTrace();

renderJson(new ResponseData(false, "督办任务修改失败!"));

}

}

/\*\*

\* 新增督办任务

\*/

public void addNewSupervisionTask(){

try {

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

SupervisionTask st = new SupervisionTask();

st.set("title", getPara("title"));

st.set("content", getPara("content"));

st.set("createtime", getPara("createtime"));

st.set("executer", getPara("executer"));

st.set("creator", user.get("id"));

st.set("stat", getPara("stat"));

if (st.save()) {

renderJson(new ResponseData(true, "督办任务新增成功!"));

} else {

renderJson(new ResponseData(false, "督办任务新增失败!"));

}

}catch(Exception ex){

ex.printStackTrace();

renderJson(new ResponseData(false, "督办任务新增失败!"));

}

}

/\*\*

\* 新增监管任务

\*/

public void addNewInspectionTask(){

try{

User user = User.me.findById(SecurityContextUtil

.getLoginUserId(getRequest()));

String stations[] = StringUtils.split(getPara("stations"),",");

for(String station : stations) {

InspectionTask it = new InspectionTask();

it.set("title",getPara("title"));

it.set("starttime",getPara("starttime"));

it.set("endtime",getPara("endtime"));

it.set("enterprise",getPara("enterprise"));

it.set("remark",getPara("remark"));

it.set("stat",getPara("stat"));

it.set("inspector",getPara("inspector"));

it.set("station", station);

it.set("creator", user.get("account"));

if (it.save()) {

// int task = InspectionTask.me.getLastInsertId();

// List<InspectionOption> lio = InspectionOption.me.getListInspectionOption();

// for (InspectionOption io : lio) {

// int option = io.getInt("id");

// InspectionTaskDetail itd = new InspectionTaskDetail();

// itd.set("result", 0);

// itd.set("option", option);

// itd.set("task", task);

// itd.save();

// }

renderJson(new ResponseData(true, "巡检任务新增成功!"));

}else{

renderJson(new ResponseData(false, "巡检任务新增失败!"));

}

} }catch(Exception ex){

renderJson(new ResponseData(false, "巡检任务新增失败!"));

}

} public void supervisionHistory() {

//User user = User.me.findById(SecurityContextUtil

// .getLoginUserId(getRequest()));

Integer id = getParaToInt("id");

setAttr("histories", SupervisionTaskHistory.me.getSupervisionTaskHistories(id));

setAttr("images",SupervisionImg.me.getSupervisionImgs(id.toString()));

render("supervisionHistory.jsp");

} public void getImgBig() {

// String enterprise\_id = getPara("enterprise\_id");

// String field\_code = getPara("field\_code");

try {

Integer id = getParaToInt(0);

String imgStr = SupervisionImg.me.getSupervisionImg(id).get("img\_big");

// imgStr = imgStr.substring(23);

imgStr = imgStr.substring(imgStr.indexOf(',')+1);

BASE64Decoder decoder = new BASE64Decoder();// Base64解码

byte[] bytes = decoder.decodeBuffer(imgStr);

for (int i = 0; i < bytes.length; ++i) {

if (bytes[i] < 0) {// 调整异常数据

bytes[i] += 256;

}

}

// 生成jpeg图片 ServletOutputStream sos = getResponse().getOutputStream();

sos.write(bytes);

sos.flush();

sos.close();

} catch (Exception e) {

e.printStackTrace();

}

renderNull();

}

}

package com.yawa.model;import com.jfinal.plugin.activerecord.Model;

import org.apache.commons.lang.StringUtils;import java.util.ArrayList;

import java.util.List;/\*\*

\* Created by user1 on 2017/4/26.

\*/

@SuppressWarnings("serial")

public class SupervisionTask extends Model<SupervisionTask> {

public static final SupervisionTask me = new SupervisionTask(); public List<SupervisionTask> getListSupervisionTask(SupervisionTask cond) {

StringBuffer sql = new StringBuffer("SELECT id,title,content,createtime,creator,stat,executer FROM base\_supervision\_task WHERE 1=1 ");

List<String> largs = new ArrayList<String>();

if (cond != null) { if (StringUtils.isNotEmpty(cond.getStr("title"))) {

sql.append(" and title like concat(?,'%') ");

largs.add(cond.getStr("title"));

}

if (cond.get("creator") != null && cond.get("executer") != null) {

sql.append(" and (creator = ? or executer = ?) ");

largs.add(String.valueOf(cond.get("creator")));

largs.add(String.valueOf(cond.get("executer")));

} else if (cond.get("creator") != null) {

sql.append(" and creator = ? ");

largs.add(cond.getStr("creator"));

} else if (cond.get("executer") != null) {

sql.append(" and executer = ? ");

largs.add(cond.getStr("executer"));

}

if (StringUtils.isNotEmpty(cond.getStr("stat"))) {

sql.append(" and stat = ? ");

largs.add(cond.getStr("stat"));

}

if (StringUtils.isNotEmpty(cond.getStr("id"))) {

sql.append(" and id = ? ");

largs.add(cond.getStr("id"));

}

Object aargs[] = new Object[largs.size()];

for (int i = 0; i < largs.size(); i++) {

aargs[i] = largs.get(i);

}

return find(sql.toString(), aargs);

} else

return find(sql.toString());

} public SupervisionTask getSupervisionTask(Integer id) {

String sql = "select id,title,content,createtime,creator,stat,executer from base\_supervision\_task where id = ?";

return findFirst(sql, id);

} public String getLastInsertId() {

return findFirst("select LAST\_INSERT\_ID() id /\* base\_supervision\_task \*/").get("id").toString();

}

}

package com.yawa.core.controller;import java.util.Date;

import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Area;

import com.yawa.core.model.Company;

import com.yawa.core.model.User;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.ToolKit;

import com.yawa.util.model.ResponseData;@Before(SessionInterceptor.class)

public class ComController extends Controller {

/\*\*

\* 地区管理首页

\*/

public void index() {

render("index.jsp");

}

/\*\*

\* 修改公司信息

\*/

public void edit() {

Integer id = getParaToInt("id");

Company company = new Company();

if(id != null)

company = Company.me.findById(id);

setAttr("parent", Company.me.getParentComs(id));

setAttr("areas", Area.me.getAreas(null,""));

setAttr("company", company);

render("edit.jsp");

}

/\*\*

\* 所有公司选择树形结构

\*/

public void getComs() {

renderJson(Company.me.getComs(getPara("id"),null));

}

/\*\*

\* 用户所属公司结构树形结构

\*/

public void getUserComs() {

User user = User.me.findById(SecurityContextUtil.getLoginUserId(getRequest()));

if(user.getStr("classify").equals("user"))

renderJson(Company.me.getComs(getPara("id"), user.getInt("company")));

else

renderJson(Company.me.getComs(getPara("id"),null));

}

/\*\*

\* 公司详情

\*/

public void getById() {

renderJson(Company.me.getById(getParaToInt("id")));

}

/\*\*

\* 更新公司信息

\*/

public void update() {

try{

String[] areas = getParaValues("company.area");

Company com = getModel(Company.class,"company");

if(com != null){

com.set("area", ToolKit.join(areas, ','));

if(com.getInt("id") == null)

com.set("state", 1)

.set("creator", SecurityContextUtil.getLoginUserId(getRequest()))

.set("create\_date", new Date())

.save();

else

com.set("update\_date", new Date())

.update();

if(com.getInt("parent") == null){

com.set("level", 1)

.set("path", com.getInt("id").toString());

}else{

Company parent = Company.me.findById(com.getInt("parent"));

com.set("level", parent.getInt("level") + 1)

.set("path", parent.getStr("path") + "," + com.getInt("id"));

}

com.update();

}

}catch(Exception e){

renderJson(new ResponseData(false,"操作发生异常，异常操作如下: " + e.getMessage()));

return;

}

renderJson(new ResponseData(true,"公司信息更新成功!"));

}

/\*\*

\* 删除公司信息

\*/

public void delete() {

Company company = Company.me.findById(getPara("id"));

if(company != null){

try{

company.set("state", 0)

.set("update\_date", new Date())

.update();

}catch(Exception e){

renderJson(new ResponseData(false,"操作发生异常，异常操作如下: " + e.getMessage()));

return;

}

}

renderJson(new ResponseData(true,"公司删除成功!"));

}

}

package com.yawa.core.controller;import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import com.jfinal.aop.Before;

import com.jfinal.core.Controller;

import com.yawa.core.interceptor.SessionInterceptor;

import com.yawa.core.model.Menu;

import com.yawa.core.model.User;

import com.yawa.util.SecurityContextUtil;

import com.yawa.util.model.ResponseData;@Before(SessionInterceptor.class)

public class MenuController extends Controller{

/\*\*

\* 菜单管理首页

\*/

public void index() {

render("index.jsp");

}

/\*\*

\* 添加快捷菜单

\*/

public void shortCuts() {

User u = SecurityContextUtil.getLoginUser(getRequest());

setAttr("user", u);

setAttr("menus", Menu.me.getAllShortCuts(u.getInt("id")));

render("shortCuts.jsp");

}

/\*\*

\* 菜单编辑页面

\*/

public void edit() {

Integer id = getParaToInt("id");

Integer parent = getParaToInt("parent");

Menu menu = new Menu();

if(id != null){

menu = Menu.me.findById(id);

menu.put("parent\_name","");

}else if(parent != null){

menu.set("parent", parent);

}

List<Menu> roots = new ArrayList<Menu>();

if(menu.getInt("parent") != null){

menu.put("parent\_name", Menu.me.findById(menu.getInt("parent")).getStr("name"));

roots = Menu.me.getRootMenus(null);

}else{

roots = Menu.me.getRootMenus(menu.getInt("id"));

}

setAttr("menu", menu);

setAttr("roots", roots);

render("edit.jsp");

}

/\*\*

\* 所有菜单列表JSON串

\*/

public void getMenus() {

renderJson(Menu.me.getMenus());

}

/\*\*

\* 获取菜单对应的角色

\*/

public void getMenuRoles() {

renderJson(Menu.me.getMenuRoles(getParaToInt("id")));

}

/\*\*

\* 新增/修改菜单

\*/

public void update() {

try{

Menu menu = getModel(Menu.class,"menu");

if(menu != null){

menu.set("state", 1).set("update\_date", new Date());

if(menu.getInt("id") == null){

Integer creator = SecurityContextUtil.getLoginUserId(getRequest());

menu.set("creator", creator);

menu.set("create\_date", new Date());

menu.save();

Menu.me.saveLoginMenu(creator, menu.getInt("id"));

}else

menu.update();

}

}catch(Exception e){

renderJson(new ResponseData(false,"菜单保存失败!"));

return;

}

renderJson(new ResponseData(true,"菜单保存成功!"));

}

/\*\*

\* 删除菜单

\*/

public void delete() {

Menu menu = Menu.me.findById(getParaToInt("id"));

if(menu != null){

try{

menu.set("state", 0).set("update\_date", new Date()).update();

}catch(Exception e){

renderJson(new ResponseData(false,"菜单操作失败!"));

return;

}

}

renderJson(new ResponseData(true,"菜单删除成功!"));

}

}

package com.yawa.model;import java.util.List;import org.apache.commons.lang.StringUtils;import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Model;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.util.model.Page;@SuppressWarnings("serial")

public class Tank extends Model<Tank> { public static final Tank me = new Tank(); /\*\*

\* 模块：储罐配置 功能：展示储罐分页列表

\*/

public Page<Tank> getPages(String name, Integer company, Integer page,

Integer limit) {

String filter = "";

if (company != null)

filter += " and s.company = " + company;

if (StringUtils.isNotEmpty(name))

filter += " and t.name like '%" + name + "%'";

String sql = "select count(1) from tanks t "

+ " left join sites s on t.site\_id = s.id"

+ " left join base\_companys c on s.company = c.id"

+ " where t.state = 1" + filter;

Long count = Db.queryLong(sql);

sql = "select t.id,t.name,class.name classify,a.path\_name,t.address,tm.capacity cube,t.water,"

+ " (select group\_concat(name) from dict where group\_code = 'TANK.DEVICE' and find\_in\_set(code, t.device)) device,"

+ " tm.model, t.create\_date from tanks t"

+ " left join base\_areas a on t.area\_id = a.id"

+ " left join (select code,name from dict where group\_code = 'TANK.CLASSIFY' and type = 'I') class on t.classify = class.code"

+ " left join tank\_model tm on t.model = tm.id"

+ " left join sites s on t.site\_id = s.id"

+ " left join base\_companys c on s.company = c.id"

+ " where t.state = 1"

+ filter

+ " order by t.id asc limit ?,?";

return new Page<Tank>(page, limit, count, find(sql, (page - 1) \* limit,

limit));

} /\*\*

\* 模块：监控列表 功能：查询监控列表信息

\*

\* @return

\*/

public List<Record> getMonitors(Integer company) {

String filter = "";

if (company != null)

filter = " and v.company in (select id from base\_companys where find\_in\_set("

+ company + ",path))";

String sql = "select v.id,cast(volume as char) volumeo,v.name,v.scale\_rate,v.volume,d.tank\_scale,d.pressure,d.ext\_press\_1,d.ext\_press\_2,d.ext\_temp,"

+ " d.ext\_flow,d.flow\_bk,d.flow\_gk,date\_format(d.create\_date,'%m%d %H:%i:%s') create\_date,v.alert\_state,ta.field,ta.sound,ta.sound\_field,"

+ " (case when avl.value is null or rtd.avgUsed = 0 then '~' else format((avl.value / rtd.avgUsed),2) end) forcast,"

+ " (select color from tank\_section where category = 'scales' and code = v.scale\_level ) scale\_color,"

+ " (case when v.plc\_state is null and v.llj\_state is null then '#949FB1'"

+ " when v.plc\_state is null then '#008000'"

+ " when v.alert\_state = 1 then '#800000' else (select color from tank\_section where category = 'work\_state' and code = v.plc\_state) end) back\_color,"

+ " (case when v.llj\_state = 'work' or v.llj\_state is null then 'white' end) font\_color"

+ " from view\_tank\_verify v"

+ " left join tank\_dynamic d on v.id = d.tank\_id"

+ " left join (select tank\_id,value from tank\_attrs where attr\_id = '66') avl on v.id = avl.tank\_id"

+ " left join (select tank\_id,group\_concat(b.field) field,sum(a.sound) sound,group\_concat(case when a.sound = 1 then b.field end) sound\_field"

+ " from tank\_alerts a,core\_attrs b"

+ " where a.attr\_id = b.attr\_id and a.state = 0 and b.field is not null group by a.tank\_id) ta on v.id = ta.tank\_id"

+ " left join (select tank\_id,avg(sum\_used) / 1.43 avgUsed from report\_tank\_date t"

+ " where stat\_date >= date\_format(date\_sub(now(), interval (select value from app\_base where code = 'expect\_days') day),'%Y-%m-%d')"

+ " group by tank\_id) rtd on v.id = rtd.tank\_id"

+ " where 1=1 " + filter + " order by v.alert\_state desc";

return Db.find(sql);

} /\*\*

\* 模块：首页 功能：当前公司储罐数量

\*

\* @param company

\* @return

\*/

public Long getTotal(Integer company) {

return Db

.queryLong(

"select count(1) from sites t,tanks t1,sites t2,base\_companys t3"

+ " where t.id = t1.site\_id and t1.site\_id = t2.id and t2.company = t3.id and t.state = 1 and t1.state = 1"

+ " and find\_in\_set(?, t3.path)", company);

} /\*\*

\* 液罐详情

\*

\* @param id

\* @return

\*/

public Tank getById(Integer id) {

String sql = "select t.\*,ar.path\_name"

+ " from tanks t left join base\_areas ar on t.area\_id = ar.id"

+ " where t.id = ?";

return findFirst(sql, id);

}

public List<Record> getTankCharts(Integer company) {

return Db

.find("select t2.id,t2.name,count(\*) cnt from tanks t,sites t1,base\_companys t2"

+ " where t.site\_id = t1.id and t1.company = t2.id and t.state = 1 and t1.state = 1"

+ " and find\_in\_set(?, t2.path) group by t2.id,t2.name",

company);

}

public List<Record> getMonitors(Integer company) {

String filter = "";

if (company != null)

filter = " and v.company in (select id from base\_companys where find\_in\_set("

+ company + ",path))";

String sql = "select v.id,cast(volume as char) volumeo,v.name,v.scale\_rate,v.volume,d.tank\_scale,d.pressure,d.ext\_press\_1,d.ext\_press\_2,d.ext\_temp,"

+ " d.ext\_flow,d.flow\_bk,d.flow\_gk,date\_format(d.create\_date,'%m%d %H:%i:%s') create\_date,v.alert\_state,ta.field,ta.sound,ta.sound\_field,"

+ " (case when avl.value is null or rtd.avgUsed = 0 then '~' else format((avl.value / rtd.avgUsed),2) end) forcast,"

+ " (select color from tank\_section where category = 'scales' and code = v.scale\_level ) scale\_color,"

+ " (case when v.plc\_state is null and v.llj\_state is null then '#949FB1'"

+ " when v.plc\_state is null then '#008000'"

+ " when v.alert\_state = 1 then '#800000' else (select color from tank\_section where category = 'work\_state' and code = v.plc\_state) end) back\_color,"

+ " (case when v.llj\_state = 'work' or v.llj\_state is null then 'white' end) font\_color"

+ " from view\_tank\_verify v"

+ " left join tank\_dynamic d on v.id = d.tank\_id"

+ " left join (select tank\_id,value from tank\_attrs where attr\_id = '66') avl on v.id = avl.tank\_id"

+ " left join (select tank\_id,group\_concat(b.field) field,sum(a.sound) sound,group\_concat(case when a.sound = 1 then b.field end) sound\_field"

+ " from tank\_alerts a,core\_attrs b"

+ " where a.attr\_id = b.attr\_id and a.state = 0 and b.field is not null group by a.tank\_id) ta on v.id = ta.tank\_id"

+ " left join (select tank\_id,avg(sum\_used) / 1.43 avgUsed from report\_tank\_date t"

+ " where stat\_date >= date\_format(date\_sub(now(), interval (select value from app\_base where code = 'expect\_days') day),'%Y-%m-%d')"

+ " group by tank\_id) rtd on v.id = rtd.tank\_id"

+ " where 1=1 " + filter + " order by v.alert\_state desc";

return Db.find(sql);

}

END