import java.util.List;

import org.apache.commons.lang.StringUtils;

import org.apache.log4j.Logger;

import com.jfinal.core.Controller;

import com.jfinal.upload.UploadFile;

import com.yawa.common.AppService;

import com.yawa.common.ConstantItems;

import com.yawa.util.Des;

public class AppController extends Controller {

public static final Logger logger = Logger.getLogger(AppController.class);

private static final long serialVersionUID = 4164978694728909371L;

/\*\*

\* APP 接口处理

\*/

public void actProcess() {

List<UploadFile> uploadFiles = null;

try{ uploadFiles = getFiles(); }catch(Exception e){}

try {

String key = getPara("key");

String appData = getPara("appData");

if(StringUtils.isNotEmpty(appData)){

appData = Des.desEncode(appData, key);

System.out.println(("appData------------:" + appData));

}

renderJson(new AppService().processAppData(getRequest(),appData,uploadFiles));

} catch (Exception e) {

logger.info(e.getMessage());

renderJson(AppService.createResponseMessage(ConstantItems.CODE\_COMMON\_BUSFAILED,

"系统处理异常：" + e.getMessage()));

}

}

}

import net.sf.json.JSONObject;

import org.apache.commons.logging.Log;

import org.apache.commons.logging.LogFactory;

import com.yawa.core.model.Dict;

import com.yawa.model.APPLedgerInspect;

import com.yawa.model.Account;

import com.yawa.model.Alert;

import com.yawa.model.AppLaws;

import com.yawa.model.Asset;

import com.yawa.model.Clients;

import com.yawa.model.Distrb;

import com.yawa.model.Home;

import com.yawa.model.AppInspect;

import com.yawa.model.AppNew;

import com.yawa.model.APPOdors;

import com.yawa.model.Offer;

import com.yawa.model.Common;

import com.yawa.model.Control;

import com.yawa.model.Customer;

import com.yawa.model.FeedBack;

import com.yawa.model.Leader;

import com.yawa.model.Ledger;

import com.yawa.model.Message;

import com.yawa.model.Release;

import com.yawa.model.Tip;

import com.yawa.model.Visit;

import com.yawa.model.Water;

public class BusiManager {

private static BusiManager bm = null;

static Log logger = LogFactory.getLog(BusiManager.class);

private BusiManager() {

}

public static BusiManager getInstance() {

if (bm == null) {

synchronized (BusiManager.class) {

if (bm == null) {

bm = new BusiManager();

}

}

}

return bm;

}

private void validateLogin(DataPkg datapkg) throws Exception{

if(StringUtils.isEmpty(datapkg.getDeviceId()))

throw new Exception("手机串号不能为空，请重新登录！");

if(StringUtils.isEmpty(datapkg.getToken()))

throw new Exception("尚未获取访问令牌，请重新登录！");

Mobile dbMobile = datapkg.getDbMobile();

if(dbMobile == null)

throw new Exception("尚未登记手机信息，请重新登录！");

String dbToken = dbMobile.getStr("token");

if(!datapkg.getToken().equals(dbToken))

throw new Exception("登录令牌不匹配，请重新登录！");

}

private DataPkg exceptPackage(Exception e){

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_BUSFAILED);

rtdatapkg.setRetmsg(e.getMessage());

return rtdatapkg;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* public methods start \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 登录校验

public DataPkg validateToken(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.validateToken(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 发送短信

public DataPkg sendMsg(DataPkg datapkg) {

return Message.me.sendMsg(datapkg);

}

// 发送邮件

public DataPkg sendEmail(DataPkg datapkg) {

return Message.me.sendEmail(datapkg);

}

// 获取当前公司

public DataPkg getCompanys(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getCompanys(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 站点字母分页列表

public DataPkg getPageSites(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getPageSites(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 储罐分页列表

public DataPkg getPageTanks(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getPageTanks(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取站点储罐信息

public DataPkg getSiteTanks(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getSiteTanks(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取储罐信息

public DataPkg getTanks(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getTanks(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取附件列表

public DataPkg getAttaches(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getAttaches(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 上传附件

public DataPkg saveAttaches(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.saveAttaches(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除附件

public DataPkg revAttach(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.revAttach(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取字典

public DataPkg getDicts(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("字典列表查询成功！");

JSONObject jsonObj = datapkg.getInParam();

String groups = null;

try{ groups = jsonObj.getString("groups"); }catch(Exception e){}

if(groups == null){

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_BUSFAILED);

rtdatapkg.setRetmsg("未获取字典组编码！");

return rtdatapkg;

}

try{

rtdatapkg.setOutParam(Dict.me.getGroups(groups));

}catch(Exception e){

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_BUSFAILED);

rtdatapkg.setRetmsg("客户详情查询失败:" + e.getMessage());

}

return rtdatapkg;

}

// 获取所有地区

public DataPkg getAreas(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getAreas(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取公司等级列表

public DataPkg listCompanys(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.listCompanys(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 登录登出 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 登录系统

public DataPkg login(DataPkg datapkg) {

try {

return Account.me.login(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 登出系统

public DataPkg logout(DataPkg datapkg) {

return Account.me.logout(datapkg);

}

// 推送绑定

public DataPkg bindUser(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.bindUser(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 解除绑定

public DataPkg unBind(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.unBind(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 首页 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 首页广告栏

public DataPkg getHomeBanner(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Home.me.getBanner(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 主页菜单

public DataPkg getMenus(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Home.me.getMenus(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 级别菜单

public DataPkg getLevelMenus(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Home.me.getLevelMenus(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 是否包含权限

public DataPkg hasRight(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Home.me.hasRight(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 个人中心 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 个人资料

public DataPkg getUser(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.getUser(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 信息保存

public DataPkg saveUser(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.saveUser(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 修改密码

public DataPkg changePass(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.changePass(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 上传头像

public DataPkg saveAvatar(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Account.me.saveAvatar(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 新闻通知 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 新闻资讯分页列表

public DataPkg getPageNews(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppNew.me.getPageNews(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取新闻详情

public DataPkg getArticle(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppNew.me.getArticle(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 法律法规 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 分页获取法律专栏信息

\* @param datapkg

\* @return

\*/

public DataPkg getPageLaws(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppLaws.me.getPageLaws(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 获取法律专栏详情

\* @param datapkg

\* @return

\*/

public DataPkg getLaw(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppLaws.me.getLaw(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 气源分页列表

public DataPkg getPageSources(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Offer.me.getPageSources(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取液源详情

public DataPkg getGasSource(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Offer.me.getGasSource(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 领导视图 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 领导数据

public DataPkg getLeaderData(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Leader.me.getLeaderData(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 装车统计

public DataPkg getLoading(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Leader.me.getLoading(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 卸车统计

public DataPkg getUnLoading(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Leader.me.getUnLoading(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 客户用气 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 客户用气

public DataPkg getUsed(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Customer.me.getUsed(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 客户日用气

public DataPkg getDailyUsed(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Customer.me.getDailyUsed(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 用气测算 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 获取用户站点列表

public DataPkg getUserSites(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Customer.me.getUserSites(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取站点测算数据

public DataPkg getReckonData(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Customer.me.getReckonData(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存计算数据

public DataPkg saveCalculate(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Customer.me.saveCalculate(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 液位监控 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 查看液位

public DataPkg getScales(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Water.me.getPageScales(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 查看液位

public DataPkg getPageScales(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Water.me.getPageScales(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 意见反馈 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 新增意见反馈

public DataPkg addFeedBack(DataPkg datapkg) {

try {

validateLogin(datapkg);

return FeedBack.me.addFeedBack(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 意见反馈列表

public DataPkg getPageFeedBack(DataPkg datapkg) {

try {

validateLogin(datapkg);

return FeedBack.me.getPageFeedBack(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取意见反馈详情

public DataPkg getFeedBack(DataPkg datapkg) {

try {

validateLogin(datapkg);

return FeedBack.me.getFeedBack(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 站点巡检 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 获取巡检列表

public DataPkg getPageInspects(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getPageInspects(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取巡检类别

public DataPkg getSiteItems(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getSiteItems(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取巡检项目

public DataPkg getInspectItems(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getInspectItems(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取巡检详情

public DataPkg getInspect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getInspect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取待解决问题

public DataPkg getTrackErrors(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getTrackErrors(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取问题跟踪列表

public DataPkg getTrackList(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.getTrackList(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 上传巡检信息

public DataPkg saveInspect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.saveInspect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 解决遗留问题

public DataPkg saveTrack(DataPkg datapkg) {

try {

validateLogin(datapkg);

return AppInspect.me.saveTrack(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 储罐充装 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 选择公司车辆

public DataPkg getCars(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.getCars(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取充装列表

public DataPkg getPageDistrbs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.getPageDistrbs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取巡检详情

public DataPkg getDistrb(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.getDistrb(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 槽车装液发布

public DataPkg saveCaoDistrb(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.saveCaoDistrb(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存装液信息

public DataPkg saveAukDistrb(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.saveAukDistrb(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取卸液详情

public DataPkg getFilling(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.getFilling(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存卸液信息

public DataPkg saveFilling(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Distrb.me.saveFilling(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 储罐监控 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 储罐列表

public DataPkg getList(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getList(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 数据列表

public DataPkg getDatas(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getDatas(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 查询阀门列表

public DataPkg getPageValves(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getPageValves(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 储罐详情

public DataPkg getTankAttrs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getTankAttrs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 杜瓦瓶监控

public DataPkg getDewar(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getDewar(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 是否是卫星站

public DataPkg isSatellite(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.isSatellite(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取用气列表

public DataPkg getSiteUsers(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getSiteUsers(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取用户数据

public DataPkg getCsrData(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getCsrData(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 远程控制界面

public DataPkg getValves(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getValves(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 查看是否开阀成功

public DataPkg getValveAttr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getValveAttr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 阀门操作控制

public DataPkg valveClick(DataPkg datapkg){

try {

validateLogin(datapkg);

return Control.me.valveClick(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 站点紧急停止

public DataPkg eStop(DataPkg datapkg){

try {

validateLogin(datapkg);

return Control.me.eStop(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取日志记录

public DataPkg getPageRecords(DataPkg datapkg){

try {

validateLogin(datapkg);

return Control.me.getPageRecords(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取小时记录

public DataPkg getPageHours(DataPkg datapkg){

try {

validateLogin(datapkg);

return Control.me.getPageHours(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 设置中心 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 告警规则分页查询

public DataPkg getRules(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Alert.me.getRules(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取告警详情

public DataPkg getRule(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Alert.me.getRule(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取设备属性

public DataPkg getGroupAttrs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Common.me.getGroupAttrs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除告警规则

public DataPkg revRule(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Alert.me.revRule(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 编辑修改规则

public DataPkg saveRule(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Alert.me.saveRule(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 内部调测 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public DataPkg getDevices(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Control.me.getDevices(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 推送消息 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 获取消息列表

public DataPkg getPageTips(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.getPageTips(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取未读消息数

public DataPkg getNewTips(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.getNewTips(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取消息详情

public DataPkg getTip(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.getTip(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 消息读取

public DataPkg readTip(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.readTip(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 消息操作

public DataPkg tipHandler(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.tipHandler(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取推送规则

public DataPkg getSetting(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.getSetting(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存推送规则

public DataPkg saveSetting(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Tip.me.saveSetting(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 站点发布 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// 站点详情

public DataPkg site(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.showSite(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 修改站点

public DataPkg getSite(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getSite(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 站点发布

public DataPkg saveSite(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveSite(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取远传盒子

public DataPkg getCollect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getCollect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存远传设备

public DataPkg saveCollect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveCollect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除远传设备

public DataPkg revCollect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.revCollect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 修改储罐

public DataPkg getTank(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getTank(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取LNG型号列表

public DataPkg getTankModels(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getTankModels(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取LNG型号

public DataPkg getTankModel(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getTankModel(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存储罐

public DataPkg saveTank(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveTank(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除储罐

public DataPkg revTank(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.revTank(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 采集设备型号下拉框

public DataPkg listDevices(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.listDevices(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取控制版本

public DataPkg getDeviceTokens(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getDeviceTokens(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取采集设备对应属性类别

public DataPkg getDeviceItems(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getDeviceItems(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取设备属性信息

public DataPkg deviceAttrs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.deviceAttrs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 采集设备查看

public DataPkg findDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.findDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 采集设备编辑

public DataPkg getDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存采集设备

public DataPkg saveDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除采集

public DataPkg revDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.revDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 采集设备查看

public DataPkg getDeviceAttrs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getDeviceAttrs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取将要修改的属性信息

public DataPkg getDeviceAttr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getDeviceAttr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取接口线路信息

public DataPkg getLines(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getLines(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 获取寄存器地址列表

public DataPkg getRaddrs(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getRaddrs(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 采集设备属性保存

public DataPkg saveDeviceAttr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveDeviceAttr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除采集

public DataPkg revAttr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.revAttr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 选择用户信息

public DataPkg getCsr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.getCsr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 保存用户信息

public DataPkg saveCsr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.saveCsr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

// 删除用户信息

public DataPkg revCsr(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Release.me.revCsr(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*客户管理\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：客户管理模块

\* 功能：分页列表

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg getPageClients(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.getPageClients(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：获取客户详情

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg getClient(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.getClient(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：客户分类数量

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg getCountClients(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.getCountClients(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：保存客户信息

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg saveClient(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.saveClient(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：获取联系人信息

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg getContact(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.getContact(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：保存联系人

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg saveContact(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.saveContact(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：删除联系人

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg rmvContact(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.rmvContact(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户管理模块

\* 功能：选择负责人

\* @author WangYong

\* @param datapkg

\* @return

\*/

public DataPkg getUsers(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Clients.me.getUsers(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*客户拜访模块\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：客户拜访

\* 功能：获取拜访列表

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg getPageVisits(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.getPageVisits(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户拜访

\* 功能：新增或保存拜访

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg saveVisit(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.saveVisit(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户拜访

\* 功能：新增或保存指定拜访

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg saveAppoint(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.saveAppoint(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户拜访

\* 功能：获取拜访

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg getVisit(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.getVisit(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户拜访

\* 功能：获取指定拜访

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg getAppoint(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.getAppoint(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：客户拜访

\* 功能：删除拜访记录

\* @author lmc

\* @param datapkg

\* @return

\*/

public DataPkg rmvVisit(DataPkg datapkg){

try {

validateLogin(datapkg);

return Visit.me.rmvVisit(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*销售插件\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：销售插件

\* 功能：查询评估的客户列表

\* 高昂(2017-6-2)

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getPageClientsAssess(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Asset.me.getPageClientsAssess(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：销售插件

\* 功能：客户的评估记录列表页

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getPageAssess(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Asset.me.getPageAssess(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：销售插件

\* 功能：根据id查询评估记录

\* 高昂(2017-6-2)

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getAssess(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Asset.me.getAssess(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：销售插件

\* 功能：评估储罐并保存

\* 高昂(2017-6-2)

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg saveAssess(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Asset.me.saveAssess(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*台账管理\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：台账模块

\* 功能：查询台账分类列表

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerTypes(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getLedgerTypes(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：查找类型设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getTypeLedgers(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getTypeLedgers(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：删除设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg revLedger(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.revLedger(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：编辑设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedger(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getLedger(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：保存设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg saveLedger(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.saveLedger(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：获取设备类型

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getEquiptype(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getEquiptype(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：获取生产公司

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getTypeManufators(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getTypeManufators(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：获取公司型号

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerModels(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getLedgerModels(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账模块

\* 功能：获取台账采集设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return Ledger.me.getLedgerDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*台账巡检模块\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：台账巡检模块

\* 功能：获取首页信息

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getPageLedgerInsp(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.getPageLedgerInsp(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：获取巡检台账列表

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getSiteLedgers(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.getSiteLedgers(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：获取附属设备列表

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerAttached(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.getLedgerAttached(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：获取检查规范

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerStandard(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.getLedgerStandard(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：保存巡检记录

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg saveLedgerInspctItems(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.saveLedgerInspctItems(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：获取站点列表

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getLedgerSites(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.getLedgerSites(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：台账巡检模块

\* 功能：删除巡检记录

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg revLedgerSiteInspect(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPLedgerInspect.me.revLedgerSiteInspect(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*\*\*\*\*\*\*加臭器管理模块\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* 模块：加臭器管理模块

\* 功能：加臭器首页列表展示

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getPageOdors(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPOdors.me.getPageOdors(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：加臭器管理模块

\* 功能：获取加臭器巡检详情

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getOdor(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPOdors.me.getOdor(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：加臭器管理模块

\* 功能：根据站点选择加臭器设备

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg getOdorDevice(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPOdors.me.getOdorDevice(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：加臭器管理模块

\* 功能：保存加臭器巡检

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg saveOdor(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPOdors.me.saveOdor(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

/\*\*

\* 模块：加臭器管理模块

\* 功能：删除加臭器巡检

\* @author WangYong

\* @param datapkg

\* @return DataPkg

\*/

public DataPkg rmvOdor(DataPkg datapkg) {

try {

validateLogin(datapkg);

return APPOdors.me.rmvOdor(datapkg);

} catch (Exception e) {

return exceptPackage(e);

}

}

}

package com.yawa.model;

import java.util.Date;

import java.util.Hashtable;

import java.util.List;

import java.util.Map;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.Company;

import com.yawa.core.model.Dict;

import com.yawa.core.model.LogApp;

import com.yawa.core.model.Mobile;

import com.yawa.core.model.Token;

import com.yawa.core.model.User;

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

import com.yawa.util.CodeUtil;

import com.yawa.util.ToolKit;

import com.yawa.util.model.Attach;

public class Account {

public static final Account me = new Account();

private Hashtable<String, String> getCache(Integer uid){

Hashtable<String, String> htTable = new Hashtable<String, String>();

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

if(user != null){

htTable.put("user", user.getStr("name"));

Company company = Company.me.findById(user.getInt("company"));

if(company != null)

htTable.put("group", company.getStr("code"));

List<Attach> attaches = Attach.me.getAttaches(com.yawa.util.model.ConstantItems.ATTACH\_AVATAR, uid);

if(attaches != null && attaches.size() > 0)

htTable.put("url", attaches.get(0).getStr("url"));

}

return htTable;

}

/\*\*

\* 登录验证

\* @param datapkg

\* @return

\* @throws Exception

\*/

public DataPkg validateToken(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

User user = User.me.findById(datapkg.getOperID());

// boolean loginPermit = Mobile.me.isAuthorized(user.getInt("id"), datapkg.getDeviceId());

// if(!loginPermit)

// throw new Exception("您的手机未被允许登录此账号！");

JSONObject jsonObj = datapkg.getInParam();

Mobile.me.updateMobile(jsonObj);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

htTable.put("tip", Tip.me.getNewTips(datapkg.getDeviceId(), datapkg.getOperID()));

htTable.put("setting", Tip.me.getSetting(datapkg).getOutParam());

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

/\*\*

\* 保存头像

\* @param datapkg

\* @return

\*/

public DataPkg saveAvatar(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("头像上传成功!");

Attach.me.removeFile(com.yawa.util.model.ConstantItems.ATTACH\_AVATAR,

datapkg.getOperID());

Attach.me.saveFile(com.yawa.util.model.ConstantItems.ATTACH\_AVATAR,

datapkg.getOperID(), datapkg.getUploadFiles());

rtdatapkg.setOutParam(getCache(datapkg.getOperID()));

return rtdatapkg;

}

public DataPkg bindUser(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Mobile mobile = datapkg.getDbMobile();

if(mobile != null)

mobile.set("isBind", 1)

.update();

return rtdatapkg;

}

public DataPkg unBind(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Mobile mobile = datapkg.getDbMobile();

if(mobile != null)

mobile.set("isBind", 0)

.update();

return rtdatapkg;

}

/\*\*

\* App登录

\* @param datapkg

\* @return

\* @throws Exception

\*/

public DataPkg login(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("账户登录成功！");

String account = "";

String password = "";

try{ account = jsonObj.getString("account"); }catch(Exception e){}

try{ password = jsonObj.getString("password"); }catch(Exception e){}

if(StringUtils.isEmpty(account.trim()))

throw new Exception("用户名不能为空！");

if(StringUtils.isEmpty(password.trim()))

throw new Exception("登录密码不能为空！");

// 更新手机信息

Mobile mobile = Mobile.me.updateMobile(jsonObj);

// User user = User.me.getUser(account, password);

User user = User.me.getUser(account);

Validate.me.accountValidate(user);

boolean loginPermit = Mobile.me.isAuthorized(user.getInt("id"), datapkg.getDeviceId());

// if(!loginPermit)

// throw new Exception("您的手机未被允许登录此账号！");

String token = ToolKit.getToken();

mobile.set("userId", user.getInt("id"))

.set("userName", user.getStr("name"))

.set("token", token)

.update();

Token.me.saveToken(user.getInt("id"), token, datapkg.getDeviceId());

// 记录登录日志

LogApp.me.Log(datapkg.getRequest(),user.getInt("id"),"login","用户账号登录系统！",user.getInt("id"), token);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

htTable.put("token", token);

htTable.put("cache", getCache(user.getInt("id")));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

/\*\*

\* 账户注销

\* @param datapkg

\* @return

\*/

public DataPkg logout(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("账户注销成功！");

Mobile.me.clear(datapkg.getDeviceId());

Token token = Token.me.getToken(datapkg.getToken());

if(token != null){

token.set("state", 0)

.set("invalid\_time", new Date())

.update();

}

LogApp logApp = new LogApp();

logApp.set("userId", datapkg.getOperID())

.set("token", datapkg.getToken())

.set("ipaddress", datapkg.getRequest().getRemoteAddr())

.set("busitype", "logout")

.set("remarks", "用户账号退出系统！")

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

.save();

return rtdatapkg;

}

/\*\*

\* 修改密码

\* @param datapkg

\* @return

\* @throws Exception

\*/

public DataPkg changePass(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("密码修改成功！");

JSONObject jsonObj = datapkg.getInParam();

String oldPass = "";

String newPass = "";

try{ oldPass = jsonObj.getString("oldPass"); }catch(Exception e){}

try{ newPass = jsonObj.getString("newPass"); }catch(Exception e){}

User user = User.me.getUserByPass(datapkg.getOperID(), oldPass);

if (user == null)

throw new Exception("账户原密码错误！");

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

throw new Exception("账户已无效或锁定！");

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

.update();

return rtdatapkg;

}

/\*\*

\* 获取当前登录用户

\* @param datapkg

\* @return

\*/

public DataPkg getUser(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("获取信息成功！");

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

User user = User.me.findFirst("select t.name, t.account, t.gender,a.name gender\_name,"

+ " t.birth\_date, t.vtype, b.name vtype\_name, t.tele\_phone,t.email from base\_users t"

+ " left join (select code,name from dict where GROUP\_CODE = 'BASE.GENDER' and type = 'I') a on t.gender = a.code"

+ " left join (select code,name from dict where GROUP\_CODE = 'USER.VTYPE' and type = 'I') b on t.vtype = b.code"

+ " where t.id = ?", datapkg.getOperID());

htTable.put("user", user);

htTable.put("gender", Dict.me.getGroupItems("BASE.GENDER"));

htTable.put("vtype", Dict.me.getGroupItems("USER.VTYPE"));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

/\*\*

\* 保存用户资料

\* @param datapkg

\* @return

\*/

@SuppressWarnings("unchecked")

public DataPkg saveUser(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("用户资料保存成功！");

User user = User.me.findById(datapkg.getOperID());

user.setAttrs((Map<String,Object>)jsonObj)

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

.update();

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.Hashtable;

import java.util.LinkedHashMap;

import java.util.List;

import org.apache.commons.lang.StringUtils;

import net.sf.json.JSONObject;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.Company;

import com.yawa.core.model.User;

import com.yawa.util.model.Attach;

public class Home {

public static final Home me = new Home();

public String getUrl(Integer company){

List<Attach> attachs = Attach.me.getAttaches(com.yawa.util.model.ConstantItems.APP\_COMPANY, company);

if(attachs.size() > 0)

return attachs.get(0).getStr("url");

Company yawa = Company.me.getByCode(com.yawa.util.model.ConstantItems.CY\_SYSTEM);

attachs = Attach.me.getAttaches(com.yawa.util.model.ConstantItems.APP\_COMPANY, yawa.getInt("id"));

if(attachs.size() > 0)

return attachs.get(0).getStr("url");

return null;

}

public DataPkg getBanner(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

User user = User.me.findById(datapkg.getOperID());

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

if(vrcom == null)

vrcom = user.getInt("company");

String url = getUrl(vrcom);

if(StringUtils.isNotEmpty(url))

htTable.put("banner", url);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg getMenus(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

List<Record> menus = Db.find("select distinct c.code, c.name, c.image, c.url,c.opened, c.message"

+ " from app\_user\_roles a"

+ " left join app\_role\_menus b on a.roleId = b.roleId"

+ " left join app\_menus c on b.menuId = c.id"

+ " where c.state = 1 and c.category = 'home' and a.userId = ? order by c.sort asc", datapkg.getOperID());

htTable.put("menus", menus);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg getLevelMenus(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

List<Record> menus = Db.find("select distinct g.name group\_name, c.code, c.name, c.image, c.url,c.opened, c.message"

+ " from app\_user\_roles a"

+ " left join app\_role\_menus b on a.roleId = b.roleId"

+ " left join app\_menus c on b.menuId = c.id"

+ " left join (select code, name from dict where group\_code = 'MENU.GROUPS' and type = 'I') g on c.category = g.code"

+ " where c.state = 1 and c.category != 'home' and a.userId = ? order by g.code,c.sort asc", datapkg.getOperID());

HashMap<String, List<Record>> hashMap = new LinkedHashMap<String, List<Record>>();

for(Record menu: menus){

String group = menu.getStr("group\_name");

List<Record> data = new ArrayList<Record>();

if(hashMap.containsKey(group))

data = hashMap.get(group);

data.add(menu);

hashMap.put(group, data);

}

rtdatapkg.setOutParam(hashMap);

return rtdatapkg;

}

public boolean hasRight(String code, Integer user){

boolean hasRight = false;

if(StringUtils.isNotEmpty(code)){

Long cnt = Db.queryLong("select count(\*) from app\_user\_roles t, app\_role\_rights t1, app\_rights t2"

+ " where t.roleId = t1.roleId and t1.rightId = t2.id and t2.code = ? and t.userId = ?", code, user);

hasRight = cnt > 0;

}

return hasRight;

}

public DataPkg hasRight(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

String code = "";

try{ code = jsonObj.getString("code"); }catch(Exception e){}

htTable.put("right", hasRight(code, datapkg.getOperID()));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

}

package com.yawa.model;

import net.sf.json.JSONObject;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.New;

public class AppNew {

public static final AppNew me = new AppNew();

/\*\*

\* 获取分页列表

\* @param datapkg

\* @return

\*/

public DataPkg getPageNews(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("新闻列表查询成功！");

Integer page = 1;

Integer limit = 10;

try{ page = jsonObj.getInt("page"); }catch(Exception e){}

rtdatapkg.setOutParam(New.me.getPages(datapkg.getRequest(), null, null, page,limit));

return rtdatapkg;

}

public DataPkg getArticle(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Long id = null;

try{ id = jsonObj.getLong("id"); }catch(Exception e){}

if(id == null)

throw new Exception("入参不能为空！");

rtdatapkg.setOutParam(New.me.getArticle(datapkg.getRequest(), id));

return rtdatapkg;

}

}

package com.yawa.model;

import net.sf.json.JSONObject;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.Law;

public class AppLaws {

public static final AppLaws me =new AppLaws();

/\*\*

\* 获取法律专栏分页列表

\* @param datapkg

\* @return

\*/

public DataPkg getPageLaws(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("新闻列表查询成功！");

Integer page = 1;

Integer limit = 10;

try{ page = jsonObj.getInt("page"); }catch(Exception e){}

rtdatapkg.setOutParam(Law.me.getPages(datapkg.getRequest(), null, null, page,limit));

return rtdatapkg;

}

/\*\*

\* 获取法律专栏详情

\* @param datapkg

\* @return

\* @throws Exception

\*/

public DataPkg getLaw(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Integer id = null;

try{ id = jsonObj.getInt("id"); }catch(Exception e){}

if(id == null)

throw new Exception("入参不能为空！");

rtdatapkg.setOutParam(Law.me.getLaw(datapkg.getRequest(), id));

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.Hashtable;

import net.sf.json.JSONObject;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.tank.model.GasSource;

import com.yawa.tank.model.GasSupplier;

public class Offer {

public static final Offer me = new Offer();

// 液源分页列表

public DataPkg getPageSources(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("查询成功！");

Integer page = 1;

Integer limit = 50;

String field = "";

String sort = "";

try{ page = jsonObj.getInt("page"); }catch(Exception e){}

try{ field = jsonObj.getString("field"); }catch(Exception e){}

try{ sort = jsonObj.getString("sort"); }catch(Exception e){}

rtdatapkg.setOutParam(GasSource.me.getPages(null,null, field, sort, page,limit));

return rtdatapkg;

}

// 液源详情信息

public DataPkg getGasSource(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("查询成功！");

Integer id = jsonObj.getInt("id");

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

htTable.put("data", GasSource.me.getSource(id));

htTable.put("suppliers", GasSupplier.me.getOffer(id));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.Date;

import java.util.Hashtable;

import org.apache.commons.lang.StringUtils;

import net.sf.json.JSONObject;

import com.jfinal.plugin.activerecord.Db;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.User;

import com.yawa.util.DateTools;

public class Leader {

public static final Leader me = new Leader();

/\*\*

\* 领导视图首页数据

\* @param datapkg

\* @return

\*/

public DataPkg getLeaderData(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

User user = User.me.findById(datapkg.getOperID());

if(user != null){

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

// 当前时间

htTable.put("now", DateTools.formatDate(new Date()));

// 本日用气

htTable.put("dload", Db.queryDouble("select ifnull(round(sum(tr.value - ifnull(b.ext\_flow,0)),2),0) from sites a"

+ " left join rpt\_hour\_llj b on a.id = b.site\_id and b.stat\_date = date\_format(now(), '%Y%m%d00')"

+ " left join tank\_attrs tr on b.device\_id = tr.device\_id and tr.attr\_id like '41%'"

+ " where a.state = 1 and a.company = ?", company));

// 本月用气

htTable.put("mload", Db.queryDouble("select ifnull(round(sum(tr.value - ifnull(b.ext\_flow,0)),2),0) from sites a"

+ " left join rpt\_hour\_llj b on a.id = b.site\_id and b.stat\_date = date\_format(now(), '%Y%m0100')"

+ " left join tank\_attrs tr on b.device\_id = tr.device\_id and tr.attr\_id like '41%'"

+ " where a.state = 1 and a.company = ?", company));

// 本日卸车

htTable.put("dunload", Db.queryDouble("select ifnull(round(sum(fill\_sum),2),0) from tank\_fillings"

+ " where date\_format(arrive\_time, '%Y%m%d') = date\_format(now(), '%Y%m%d') and company = ?", company));

// 本月卸车

htTable.put("munload", Db.queryDouble("select ifnull(round(sum(fill\_sum),2),0) from tank\_fillings"

+ " where date\_format(arrive\_time, '%Y%m') = date\_format(now(), '%Y%m') and company = ?", company));

}

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

/\*\*

\* 查看车辆装液记录

\* @param datapkg

\* @return

\*/

public DataPkg getLoading(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Date date = new Date();

String stat = "";

String category = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

try{ category = jsonObj.getString("category"); }catch(Exception e){}

User user = User.me.findById(datapkg.getOperID());

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

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

if(category.equals("month")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年");

Double monthMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from sites a left join rpt\_hour\_llj b on a.id = b.site\_id where a.state = 1 and a.company = ? and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m')) t ", company);

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月') label,"

+ " date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') stat,ifnull(round(sum(b.total\_used),2),0) used,"

+ " concat(round(sum(b.total\_used) / " + monthMax + " \* 100,2), '%') rate,"

+ " date\_format(now(),'%Y-%m') = date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') currt"

+ " from sites a left join rpt\_hour\_llj b on a.id = b.site\_id"

+ " where a.state = 1 and a.company = ? and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') order by 1 desc", company));

}else if(category.equals("day")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年MM月");

Double dayMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from sites a left join rpt\_hour\_llj b on a.id = b.site\_id where a.state = 1 and a.company = ? and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d')) t ", company);

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') stat,ifnull(round(sum(b.total\_used),2),0) used,"

+ " concat(round(sum(b.total\_used) / " + dayMax + " \* 100,2), '%') rate"

+ " from sites a left join rpt\_hour\_llj b on a.id = b.site\_id"

+ " where a.state = 1 and a.company = ? and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') order by 1 desc", company));

}

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg getUnLoading(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Date date = new Date();

String stat = "";

String category = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

try{ category = jsonObj.getString("category"); }catch(Exception e){}

User user = User.me.findById(datapkg.getOperID());

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

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

if(category.equals("month")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年");

Double monthMax = Db.queryDouble("select max(loading) loading from (select round(sum(`fill\_sum`), 2) loading from tank\_fillings a"

+ " where date\_format(arrive\_time, '%Y年') = ? and a.company = ? group by date\_format(arrive\_time, '%Y-%m')) t", stat, company);

htTable.put("data", Db.find("select date\_format(arrive\_time, '%Y年%m月') label,date\_format(arrive\_time, '%Y-%m') stat,ifnull(round(sum(`fill\_sum`), 2),0) used,"

+ " concat(round(sum(`fill\_sum`) / " + monthMax + " \* 100,2), '%') rate,"

+ " date\_format(now(),'%Y-%m') = date\_format(arrive\_time, '%Y-%m') currt from tank\_fillings a"

+ " where date\_format(arrive\_time, '%Y年') = ? and a.company = ? group by date\_format(arrive\_time, '%Y-%m') order by 1 desc", stat, company));

}else if(category.equals("day")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年MM月");

Double dayMax = Db.queryDouble("select max(loading) loading from (select round(sum(`fill\_sum`),2) loading from tank\_fillings a"

+ " where date\_format(arrive\_time, '%Y年%m月') = ? and a.company = ? group by date\_format(arrive\_time, '%Y-%m-%d')) t", stat, company);

htTable.put("data", Db.find("select date\_format(arrive\_time, '%m-%d') stat,round(sum(`fill\_sum`),2) used,"

+ " concat(round(sum(`fill\_sum`) / " + dayMax + " \* 100,2), '%') rate,"

+ " date\_format(now(),'%Y-%m-%d') = date\_format(arrive\_time, '%Y-%m-%d') currt from tank\_fillings a"

+ " where date\_format(arrive\_time, '%Y年%m月') = ? and a.company = ? group by date\_format(arrive\_time, '%m-%d') order by 1 desc", stat, company));

}

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.Date;

import java.util.Hashtable;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.LogApp;

import com.yawa.util.DateTools;

public class Customer {

public static final Customer me = new Customer();

// 客户用气查询

public DataPkg getUsed(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Date date = new Date();

String stat = "";

String category = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

try{ category = jsonObj.getString("category"); }catch(Exception e){}

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

if(category.equals("month")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年");

Double monthMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m')) t", datapkg.getOperID());

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月') label,"

+ " date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') stat,"

+ " round(sum(b.total\_used),2) used,concat(round(sum(b.total\_used) / " + monthMax + " \* 100,2), '%') rate,"

+ " date\_format(now(),'%Y-%m') = date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') currt "

+ " from base\_user\_sites a, rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') order by 1 desc", datapkg.getOperID()));

}else if(category.equals("day")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年MM月");

Double dayMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d')) t ", datapkg.getOperID());

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') stat,"

+ " date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') label,"

+ " round(ifnull(sum(b.total\_used), 0),2) used,concat(round(ifnull(sum(b.total\_used),0) / " + dayMax + " \* 100,2), '%') rate"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') order by 1 desc", datapkg.getOperID()));

}

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

// 储罐日用气

public DataPkg getDailyUsed(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

String stat = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

Double dayMax = Db.queryDouble("select max(t.gasUsed) from (select sum(b.total\_used) gasUsed from base\_user\_sites a,rpt\_hour\_llj b, site\_device c, tanks d"

+ " where find\_in\_set(b.site\_id, a.sites) and b.device\_id = c.id and find\_in\_set(d.id, c.tank\_ids)"

+ " and c.state = 1 and d.state = 1 and a.userid = ? and a.relaType = '20'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') = ? group by d.id) t", datapkg.getOperID(), stat);

htTable.put("data", Db.find("select d.name, b.total\_used,concat(round(ifnull(sum(b.total\_used),0) / " + dayMax + " \* 100,2), '%') rate"

+ " from base\_user\_sites a,rpt\_hour\_llj b, site\_device c, tanks d"

+ " where find\_in\_set(b.site\_id, a.sites) and b.device\_id = c.id and find\_in\_set(d.id, c.tank\_ids)"

+ " and c.state = 1 and d.state = 1 and a.userid = ? and a.relaType = '20'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') = ? group by d.id", datapkg.getOperID(), stat));

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg getUserSites(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setOutParam(Db.find("select b.id, b.name from base\_user\_sites a, sites b"

+ " where find\_in\_set(b.id, a.sites) and a.relaType = '20' and b.state = 1 and a.userId = ?", datapkg.getOperID()));

return rtdatapkg;

}

public DataPkg getReckonData(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Integer site = null;

try{ site = jsonObj.getInt("site"); }catch(Exception e){}

if(site == null)

throw new Exception("未获取站点标识");

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

Record rd = Db.findFirst("select sum(b.weight) value, sum(c.average) avg\_used, sum(b.supply\_days) forcast"

+ " from site\_device a left join data\_tanks b on a.id = b.device\_id left join data\_lljs c on a.id = c.device\_id"

+ " where a.site\_id = ?", site);

htTable.put("left", rd.getDouble("value"));

htTable.put("avgUsed", rd.getDouble("avg\_used"));

htTable.put("supplyDay",rd.getDouble("forcast"));

rd = Db.findFirst("select max(a.sum\_used) topUsed,sum(a.unload) charge,sum(a.sum\_used) sumUsed"

+ " from report\_tank\_date a, tanks b where a.tank\_id = b.id"

+ " and str\_to\_date(a.stat\_date, '%Y-%m-%d') >= date\_sub(now(),INTERVAL 3 month) and b.site\_id = ?", site);

htTable.put("topUsed", rd.getStr("topUsed"));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg saveCalculate(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Integer site = null;

Object data = null;

try{ site = jsonObj.getInt("site"); }catch(Exception e){}

try{ data = jsonObj.get("data"); }catch(Exception e){}

if(site != null)

LogApp.me.Log(datapkg.getRequest(), site, "calculate", data.toString(),

datapkg.getOperID(), datapkg.getToken());

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.Date;

import java.util.Hashtable;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.LogApp;

import com.yawa.util.DateTools;

public class Customer {

public static final Customer me = new Customer();

// 客户用气查询

public DataPkg getUsed(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Date date = new Date();

String stat = "";

String category = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

try{ category = jsonObj.getString("category"); }catch(Exception e){}

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

if(category.equals("month")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年");

Double monthMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m')) t", datapkg.getOperID());

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月') label,"

+ " date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') stat,"

+ " round(sum(b.total\_used),2) used,concat(round(sum(b.total\_used) / " + monthMax + " \* 100,2), '%') rate,"

+ " date\_format(now(),'%Y-%m') = date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') currt "

+ " from base\_user\_sites a, rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y-%m') order by 1 desc", datapkg.getOperID()));

}else if(category.equals("day")){

if(StringUtils.isEmpty(stat))

stat = DateTools.formatDate(date ,"yyyy年MM月");

Double dayMax = Db.queryDouble("select max(t.gas) from (select round(sum(b.total\_used),2) gas"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d')) t ", datapkg.getOperID());

htTable.put("data", Db.find("select date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') stat,"

+ " date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') label,"

+ " round(ifnull(sum(b.total\_used), 0),2) used,concat(round(ifnull(sum(b.total\_used),0) / " + dayMax + " \* 100,2), '%') rate"

+ " from base\_user\_sites a,rpt\_hour\_llj b where a.userid = ? and a.relaType = '20' and find\_in\_set(b.site\_id, a.sites) and b.stat\_date like '%00'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') like '" + stat + "%'"

+ " group by date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%m-%d') order by 1 desc", datapkg.getOperID()));

}

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

// 储罐日用气

public DataPkg getDailyUsed(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

String stat = "";

try{ stat = jsonObj.getString("stat"); }catch(Exception e){}

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

Double dayMax = Db.queryDouble("select max(t.gasUsed) from (select sum(b.total\_used) gasUsed from base\_user\_sites a,rpt\_hour\_llj b, site\_device c, tanks d"

+ " where find\_in\_set(b.site\_id, a.sites) and b.device\_id = c.id and find\_in\_set(d.id, c.tank\_ids)"

+ " and c.state = 1 and d.state = 1 and a.userid = ? and a.relaType = '20'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') = ? group by d.id) t", datapkg.getOperID(), stat);

htTable.put("data", Db.find("select d.name, b.total\_used,concat(round(ifnull(sum(b.total\_used),0) / " + dayMax + " \* 100,2), '%') rate"

+ " from base\_user\_sites a,rpt\_hour\_llj b, site\_device c, tanks d"

+ " where find\_in\_set(b.site\_id, a.sites) and b.device\_id = c.id and find\_in\_set(d.id, c.tank\_ids)"

+ " and c.state = 1 and d.state = 1 and a.userid = ? and a.relaType = '20'"

+ " and date\_format(date\_sub(str\_to\_date(b.stat\_date, '%Y%m%d%H'),INTERVAL 1 DAY),'%Y年%m月%d日') = ? group by d.id", datapkg.getOperID(), stat));

htTable.put("stat", stat);

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg getUserSites(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setOutParam(Db.find("select b.id, b.name from base\_user\_sites a, sites b"

+ " where find\_in\_set(b.id, a.sites) and a.relaType = '20' and b.state = 1 and a.userId = ?", datapkg.getOperID()));

return rtdatapkg;

}

public DataPkg getReckonData(DataPkg datapkg) throws Exception {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Integer site = null;

try{ site = jsonObj.getInt("site"); }catch(Exception e){}

if(site == null)

throw new Exception("未获取站点标识");

Hashtable<String, Object> htTable = new Hashtable<String, Object>();

Record rd = Db.findFirst("select sum(b.weight) value, sum(c.average) avg\_used, sum(b.supply\_days) forcast"

+ " from site\_device a left join data\_tanks b on a.id = b.device\_id left join data\_lljs c on a.id = c.device\_id"

+ " where a.site\_id = ?", site);

htTable.put("left", rd.getDouble("value"));

htTable.put("avgUsed", rd.getDouble("avg\_used"));

htTable.put("supplyDay",rd.getDouble("forcast"));

rd = Db.findFirst("select max(a.sum\_used) topUsed,sum(a.unload) charge,sum(a.sum\_used) sumUsed"

+ " from report\_tank\_date a, tanks b where a.tank\_id = b.id"

+ " and str\_to\_date(a.stat\_date, '%Y-%m-%d') >= date\_sub(now(),INTERVAL 3 month) and b.site\_id = ?", site);

htTable.put("topUsed", rd.getStr("topUsed"));

rtdatapkg.setOutParam(htTable);

return rtdatapkg;

}

public DataPkg saveCalculate(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

JSONObject jsonObj = datapkg.getInParam();

Integer site = null;

Object data = null;

try{ site = jsonObj.getInt("site"); }catch(Exception e){}

try{ data = jsonObj.get("data"); }catch(Exception e){}

if(site != null)

LogApp.me.Log(datapkg.getRequest(), site, "calculate", data.toString(),

datapkg.getOperID(), datapkg.getToken());

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.List;

import net.sf.json.JSONObject;

import com.jfinal.plugin.activerecord.Db;

import com.jfinal.plugin.activerecord.Record;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.Company;

import com.yawa.core.model.User;

import com.yawa.util.model.Page;

public class Water {

public static final Water me = new Water();

// 查看液位

public DataPkg getPageScales(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("列表数据查詢成功");

Integer page = 1;

Integer limit = 50;

try{ page = jsonObj.getInt("page"); }catch(Exception e){}

User user = User.me.findById(datapkg.getOperID());

Integer root = Company.me.getRootCompany(user.getInt("company"));

Company company = Company.me.findById(root);

String filter = "";

if(!Company.me.isSystem(company))

filter += " and find\_in\_set(" + root + ", e.path)";

String sql = "select count(1) from sites a left join tanks b on a.id = b.site\_id left join data\_tanks c on find\_in\_set(b.id, c.tank\_ids)"

+ " left join data\_cards d on find\_in\_set(b.id, d.tank\_ids) left join base\_companys e on a.company = e.id"

+ " where a.state = 1 and b.state = 1 and (c.id is not null or d.id is not null) " + filter;

Long count = Db.queryLong(sql);

List<Record> records = Db.find("select a.id, b.id tank, b.name, b.cube,"

+ "(case when c.id is not null then 'plc' else 'card' end) classify,"

+ "(case when c.id is not null then concat(c.rate,'%') when d.id is not null then concat(d.rate,'%') end) rate,"

+ "(case when timestampdiff(second, c.update\_time,now()) > 300 or timestampdiff(second, d.update\_time,now()) > 300 then 'off' else 'on' end) line,"

+ "(case when c.rate < 20 or d.rate < 20 then 'alert'"

+ " when (c.rate >= 20 and c.rate < 50) or (d.rate >= 20 and d.rate < 50) then 'warn'"

+ " when c.rate >= 50 or d.rate >= 50 then 'info' end) class,"

+ "(case when c.id is not null and c.supply\_days is not null then c.supply\_days"

+ " when c.id is not null then '未用气'"

+ " when d.id is not null and d.supply\_days is not null then d.supply\_days"

+ " when d.id is null then '为用气' end) cnt,"

+ " c.scale, c.volume, c.weight, d.device\_id, d.card\_gqzl, d.card\_yqzl, d.card\_syql, d.card\_ffyq"

+ " from sites a left join tanks b on a.id = b.site\_id left join data\_tanks c on find\_in\_set(b.id, c.tank\_ids) "

+ " left join data\_cards d on find\_in\_set(b.id, d.tank\_ids) left join base\_companys e on a.company = e.id"

+ " where a.state = 1 and b.state = 1 and (c.id is not null or d.id is not null) " + filter

+ " order by e.id asc limit ?,?", (page - 1) \* limit,limit);

rtdatapkg.setOutParam(new Page<Record>(page, limit, count, records));

return rtdatapkg;

}

}

package com.yawa.model;

import java.util.Date;

import java.util.List;

import java.util.Map;

import org.apache.commons.lang.StringUtils;

import net.sf.json.JSONObject;

import com.jfinal.plugin.activerecord.Db;

import com.yawa.common.ConstantItems;

import com.yawa.common.DataPkg;

import com.yawa.core.model.Fedback;

import com.yawa.core.model.User;

import com.yawa.util.model.Page;

public class FeedBack {

public static final FeedBack me = new FeedBack();

@SuppressWarnings("unchecked")

public DataPkg addFeedBack(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("意见反馈成功!");

JSONObject jsonObj = datapkg.getInParam();

User user = User.me.findById(datapkg.getOperID());

Fedback fb = new Fedback();

fb.set("channel", "app")

.set("state", 1)

.set("createdate", new Date())

.setAttrs((Map<String,Object>)jsonObj);

fb.save(fb, user);

return rtdatapkg;

}

public DataPkg getPageFeedBack(DataPkg datapkg) {

DataPkg rtdatapkg = new DataPkg();

JSONObject jsonObj = datapkg.getInParam();

rtdatapkg.setRetcode(ConstantItems.CODE\_COMMON\_SUCCESS);

rtdatapkg.setRetmsg("意见列表查询成功！");

Integer page = 1;

String type = "";

Integer limit = 10;

try{ page = jsonObj.getInt("page"); }catch(Exception e){}

try{ type = jsonObj.getString("type"); }catch(Exception e){}

String filter = "";

if(StringUtils.isNotEmpty(type))

filter = " and t.`type` = '" + type + "'";

String sql = "select count(1) from base\_info\_fedback t where t.channel = 'app'" +

" and t.creator = ? " + filter;

Long count = Db.queryLong(sql, datapkg.getOperID());

List<Fedback> feds = Fedback.me.find("select t.id,t1.name typename,t.content,t.state, t2.name statename,"

+ " t.reply,t.createdate, t.updatedate from base\_info\_fedback t"

+ " left join (select code,name from dict where group\_code = 'FEDBACK.TYPE' and type = 'I') t1 on t.`type` = t1.code"

+ " left join (select code,name from dict where group\_code = 'FEDBACK.STATE' and type = 'I') t2 on t.`state` = t2.code"

+ " where t.channel = 'app' and t.creator = ? " + filter + " order by t.id desc limit ?,?", datapkg.getOperID(), (page - 1) \* limit,limit);

rtdatapkg.setOutParam(new Page<Fedback>(page, limit, count, feds));

return rtdatapkg;

}

END