package com.sdocean.dataQuery.dao;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Echarts;

import com.sdocean.common.model.HchartsLineData;

import com.sdocean.common.model.HchartsServieModel;

import com.sdocean.common.model.PlotLine;

import com.sdocean.common.model.YAxis;

import com.sdocean.dataQuery.model.GraphModel;

import com.sdocean.device.dao.DeviceDao;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.dictionary.dao.WaterQualityStandardDao;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.indicator.dao.IndicatorDao;

import com.sdocean.indicator.model.IndicatorModel;

import com.sdocean.main.dao.MainTenanceDao;

import com.sdocean.main.model.MainTainModel;

import com.sdocean.metadata.dao.MetadataTableDao;

import com.sdocean.metadata.model.MetadataTable;

import com.sdocean.station.model.StationModel;

import com.sdocean.warn.dao.DeviceAlarmDao;

import com.sdocean.warn.dao.RangeDataDao;

import com.sdocean.warn.model.DeviceAlarmModel;

import com.sdocean.warn.model.RangeDataModel;

@Component

public class GraphQueryDao extends OracleEngine{

@Resource

MetadataTableDao tableDao;

@Resource

IndicatorDao indicatorDao;

@Resource

DeviceDao deviceDao;

@Resource

WaterQualityStandardDao wasDao;

@Resource

RangeDataDao rangeDataDao;

@Resource

MainTenanceDao mainTenanceDao;

@Resource

DeviceAlarmDao deviceAlarmDao;

public List<HchartsServieModel> getEcharts4Graph(StationModel station,GraphModel model){

//初始化时间格式

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

//初始化返回结果

List<HchartsServieModel> list = new ArrayList<>();

//根据起始时间判断出需要查询的表名的集合

List<MetadataTable> tables = tableDao.getTables4Meta(station,model.getBeginDate(), model.getEndDate(), 1);

//得到查询条件中的设备的集合

List<DeviceModel> devices = model.getDevices();

//便利设备集合

for(DeviceModel device:devices){

//得到查询条件中的参数的集合

List<IndicatorModel> indicators = device.getIndicators();

//根据站点\设备\参数,依次查询出数据结果

for(IndicatorModel indicator:indicators){

//查询当前站点\设备的报警码,并匹配参数的上下线

List<DeviceAlarmModel> deviceAlarmList = deviceAlarmDao.getDeviceAlarmDataListByStationDeviceIndicator(

station.getId(), device.getId(), indicator.getCode(), model.getBeginDate(), model.getEndDate());

StringBuffer ydataSql = new StringBuffer("");

if(deviceAlarmList!=null&&deviceAlarmList.size()>0) {

ydataSql.append(" case ");

for(DeviceAlarmModel deviceAlarm:deviceAlarmList) {

ydataSql.append(" when collect\_time between ");

}

ydataSql.append(" else data end as ydata ");

}else {

ydataSql.append(" data as ydata");

}

//根据站点\设备\参数查询出应该符合的数据范围

RangeDataModel ed = rangeDataDao.getRangeDataByStationDeviceIndicator(station.getId(),device.getId(), indicator.getCode());

HchartsServieModel servie = new HchartsServieModel();

servie.setResult(indicator);

//开始拼接查询数据的SQL语句

StringBuffer sql = new StringBuffer();

//定义SELECT语句

StringBuffer selectSql = new StringBuffer("");

selectSql.append(" select collect\_time as xtime,data as ydata");

StringBuffer whereSql = new StringBuffer(" where 1=1");

//添加查询条件

whereSql.append(" and wpid = ").append(station.getId());

whereSql.append(" and deviceid = ").append(device.getId());

whereSql.append(" and indicator\_code='").append(indicator.getCode()).append("'");

whereSql.append(" and collect\_time>='").append(model.getBeginDate()).append("'");

whereSql.append(" and collect\_time<='").append(model.getEndDate()).append("'");

if(ed!=null) {

whereSql.append(" and data >=").append(ed.getMinData());

whereSql.append(" and data <=").append(ed.getMaxData());

}

//屏蔽掉运维数据

List<MainTainModel> mainList = mainTenanceDao.getMainTainList4StationDeviceIndicator(station.getId(),

device.getId(), indicator.getCode(), model.getBeginDate(), model.getEndDate());

for(MainTainModel main:mainList){

whereSql.append(" and !(collect\_time >= '").append(main.getBeginTime()).append("' ");

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

whereSql.append(" and collect\_time <= '").append(main.getEndTime()).append("'");

}

whereSql.append(")");

}

//定义排序语句

StringBuffer ordersql = new StringBuffer(" order by xtime desc");

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

MetadataTable table = tables.get(i);

String tableName = table.getTableName();

StringBuffer fromSql = new StringBuffer(" from ");

fromSql.append(tableName);

//将每个表的sql语句添加到总语句中

sql.append(selectSql).append(fromSql).append(whereSql);

if(i<tables.size()-1){

sql.append(" union all ");

}

}

sql.append(ordersql);

//初始化单个参数的结果集

List<HchartsLineData> lds = new ArrayList<>();

System.out.println(sql.toString());

lds = this.queryObjectList(sql.toString(), HchartsLineData.class);

List<double[]> resu = new ArrayList<>();

for(HchartsLineData ld:lds){

try {

Date da = sdf.parse(ld.getXtime());

double[] res = {da.getTime(),ld.getYdata()};

resu.add(res);

} catch (ParseException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

servie.setData(resu);

list.add(servie);

}

}

return list;

}

}

package com.sdocean.dataQuery.action;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Echarts;

import com.sdocean.common.model.HchartsServieModel;

import com.sdocean.common.model.SelectTree;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.dataQuery.model.GraphModel;

import com.sdocean.dataQuery.service.GraphQueryService;

import com.sdocean.dictionary.service.WaterQualityStandardService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.indicator.service.IndicatorService;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationService;

@Controller

public class GraphQueryAction {

private static Logger log = Logger.getLogger(GraphQueryAction.class);

@Autowired

IndicatorService indicatorService;

@Autowired

GraphQueryService graphService;

@Autowired

private ConfigInfo info;

@Autowired

StationService stationService;

@Autowired

WaterQualityStandardService waterQualityStandardService;

@RequestMapping("info\_graphquery.do")

public ModelAndView info\_graphquery(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/dataquery/graphquery\_init");

DateFormat beginDf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

DateFormat df = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

Calendar calendar = Calendar.getInstance();

calendar.setTime(new Date());

//默认开始时间为一个月以前

calendar.add(Calendar.MONTH, -2);

// 添加默认开始时间

String time1 = beginDf.format(calendar.getTime()) ;

// 添加默认结束时间

String time2 = df.format(new Date());

// 存放参数

mav.addObject("beginDate", time1);

mav.addObject("endDate", time2);

return mav;

}

/\*

\* 为查询条件初始化

\*/

@RequestMapping(value="graphquery\_init.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String graphquery\_init(HttpServletRequest request,

HttpServletResponse response){

DataQueryModel model = new DataQueryModel();

HttpSession session = request.getSession();

StationModel station = (StationModel) session.getAttribute("station");

model.setStationId(station.getId());

DateFormat beginDf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

Calendar calendar = Calendar.getInstance();

calendar.setTime(new Date());

//设置结束时间

String endDate = beginDf.format(calendar.getTime());

//设置开始时间

calendar.add(Calendar.DATE, -20);

String beginDate = beginDf.format(calendar.getTime());

model.setBeginDate(beginDate);

model.setEndDate(endDate);

//根据站点获得该站点下的参数列表

List<SelectTree> indicatorTree = indicatorService.getIndicators4StationDevice4Show(station);

model.setIndicatorTree(indicatorTree);

return JsonUtil.toJson(model);

}

/\*

\* 数据评价中综合趋势展示图标

\*/

@RequestMapping(value="graphShow4echarts.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String graphShow4echarts(@ModelAttribute("model") GraphModel model,HttpServletRequest request,

HttpServletResponse response) {

//对参数进行初始化

graphService.modelInfo(model);

HttpSession session = request.getSession();

StationModel station = (StationModel) session.getAttribute("station");

List<HchartsServieModel> list= graphService.getEcharts4Graph(station, model);

return JsonUtil.toJson(list);

}

}

package com.sdocean.warn.service;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.SelectTree;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.dao.StationCommDao;

import com.sdocean.station.dao.StationDeviceDao;

import com.sdocean.station.model.StationDeviceComm;

import com.sdocean.station.model.StationDeviceModel;

import com.sdocean.station.model.StationInfo;

import com.sdocean.station.model.StationModel;

import com.sdocean.warn.dao.DeviceAlarmDao;

import com.sdocean.warn.model.DeviceAlarmConfig;

import com.sdocean.warn.model.DeviceAlarmModel;

@Service

@Transactional(rollbackFor = Exception.class, propagation = Propagation.REQUIRED)

public class DeviceAlarmService {

@Resource

private DeviceAlarmDao deviceAlarmDao;

/\*

\* 为人员管理的查询结果添加表头

\*/

public List<UiColumn> getCols4DeviceAlarmList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col0 = new UiColumn("id", "id", false, "\*");

UiColumn col1 = new UiColumn("stationId", "stationId", false, "\*");

UiColumn col2 = new UiColumn("站点名称", "stationName", true, "\*");

UiColumn col3 = new UiColumn("deviceId", "deviceId", false, "\*");

UiColumn col6 = new UiColumn("设备名称", "deviceName", true, "\*");

UiColumn col7 = new UiColumn("configId", "configId", false, "\*");

UiColumn col8 = new UiColumn("报警类型", "configName", true, "\*");

UiColumn col9 = new UiColumn("报警码", "alarmData", true, "\*");

UiColumn col10 = new UiColumn("开始时间", "beginTime", true, "\*");

UiColumn col11 = new UiColumn("结束时间", "endTime", true, "\*");

cols.add(col0);

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

cols.add(col9);

cols.add(col10);

cols.add(col11);

return cols;

}

/\*

\* 读取设备报警码表

\*/

public List<DeviceAlarmModel> getDeviceAlarmList(DeviceAlarmModel model,List<StationModel> stations){

return deviceAlarmDao.getDeviceAlarmList(model, stations);

}

/\*

\* 添加设备报警码表

\*/

public Result saveNewDeviceAlarm(DeviceAlarmModel model){

return deviceAlarmDao.saveNewDeviceAlarm(model);

}

/\*

\* 修改站点设备配置

\*/

public Result saveChangeDeviceAlaram(DeviceAlarmModel model){

return deviceAlarmDao.saveChangeDeviceAlaram(model);

}

/\*

\* 删除设备报警码

\*/

public Result deleteDeviceAlarm(DeviceAlarmModel model){

return deviceAlarmDao.deleteDeviceAlarm(model);

}

/\*

\* 获得报警类型列表

\*/

public List<DeviceAlarmConfig> getDeviceAlarmConfigList(DeviceAlarmConfig model){

return deviceAlarmDao.getDeviceAlarmConfigList(model);

}

}

package com.sdocean.warn.model;

import java.util.List;

import com.sdocean.dictionary.model.PublicModel;

import com.sdocean.station.model.StationModel;

public class WarnValueModel {

private int id;

private int wpId;

private String wpName;

private int deviceId;

private String deviceName;

private String indicatorCode;

private String indicatorName;

private String beginDate;

private String endDate;

private int type;

private String typeName;

private Double value;

private String collect\_time;

private int operate; //0:未操作 1:已操作

private String operateName;

private String flag; //备注

private String remark;

private Double minConfig; //保留配置中的最低下限

private Double maxConfig; //保留配置中的最高上限

private String unitName;

private String ids;

//

private List<StationModel> stationList;

private List<PublicModel> typeList;

public String getIds() {

return ids;

}

public void setIds(String ids) {

this.ids = ids;

}

public List<StationModel> getStationList() {

return stationList;

}

public void setStationList(List<StationModel> stationList) {

this.stationList = stationList;

}

public List<PublicModel> getTypeList() {

return typeList;

}

public void setTypeList(List<PublicModel> typeList) {

this.typeList = typeList;

}

public String getOperateName() {

return operateName;

}

public void setOperateName(String operateName) {

this.operateName = operateName;

}

public String getUnitName() {

return unitName;

}

public void setUnitName(String unitName) {

this.unitName = unitName;

}

public String getWpName() {

return wpName;

}

public void setWpName(String wpName) {

this.wpName = wpName;

}

public String getDeviceName() {

return deviceName;

}

public void setDeviceName(String deviceName) {

this.deviceName = deviceName;

}

public String getIndicatorName() {

return indicatorName;

}

public void setIndicatorName(String indicatorName) {

this.indicatorName = indicatorName;

}

public String getTypeName() {

return typeName;

}

public void setTypeName(String typeName) {

this.typeName = typeName;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getWpId() {

return wpId;

}

public void setWpId(int wpId) {

this.wpId = wpId;

}

public int getDeviceId() {

return deviceId;

}

public void setDeviceId(int deviceId) {

this.deviceId = deviceId;

}

public String getIndicatorCode() {

return indicatorCode;

}

public void setIndicatorCode(String indicatorCode) {

this.indicatorCode = indicatorCode;

}

public String getBeginDate() {

return beginDate;

}

public void setBeginDate(String beginDate) {

this.beginDate = beginDate;

}

public String getEndDate() {

return endDate;

}

public void setEndDate(String endDate) {

this.endDate = endDate;

}

public int getType() {

return type;

}

public void setType(int type) {

this.type = type;

}

public Double getValue() {

return value;

}

public void setValue(Double value) {

this.value = value;

}

public String getCollect\_time() {

return collect\_time;

}

public void setCollect\_time(String collect\_time) {

this.collect\_time = collect\_time;

}

public int getOperate() {

return operate;

}

public void setOperate(int operate) {

this.operate = operate;

}

public String getFlag() {

return flag;

}

public void setFlag(String flag) {

this.flag = flag;

}

public String getRemark() {

return remark;

}

public void setRemark(String remark) {

this.remark = remark;

}

public Double getMinConfig() {

return minConfig;

}

public void setMinConfig(Double minConfig) {

this.minConfig = minConfig;

}

public Double getMaxConfig() {

return maxConfig;

}

public void setMaxConfig(Double maxConfig) {

this.maxConfig = maxConfig;

}

}package com.sdocean.warn.dao;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Result;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.station.dao.StationDao;

import com.sdocean.station.model.StationModel;

import com.sdocean.users.model.SysUser;

import com.sdocean.warn.model.Warn4FirstModel;

import com.sdocean.warn.model.WarnModel;

import com.sdocean.warn.model.WarnValueModel;

@Component

public class WarnDao extends OracleEngine {

@Resource

StationDao stationDao;

/\*

\* 为首页读取预警信息

\*/

public Warn4FirstModel getWarn4First(Long warnType,StationModel watchPoint,String beginDate){

Warn4FirstModel warn = new Warn4FirstModel();

String typeSql = "";

String warnTypeString = "";

if(warnType==0){//如果为0则表示预警信息

warnTypeString = "预警信息";

typeSql = " a.max\_value as upper,a.min\_value as lower";

}else{ //如果为1则表示告警信息

warnTypeString = "告警信息";

typeSql = " a.max\_value as upper,a.min\_value as lower";

}

//开始拼接SQL语句

StringBuffer sql = new StringBuffer("");

sql.append(" select a.collect\_time as collecttime,d.title,e.logo as unitname,a.value as data,");

sql.append(" if(a.operate=0,'未处理','已处理') as operate,");

sql.append(typeSql);

sql.append(" from aiot\_warn\_alarm\_value a,dm\_indicator d,g\_unit e");

sql.append(" where a.indicatorCode = d.code and d.unitid = e.id");

sql.append(" and a.collect\_time >='").append(beginDate).append("'");

sql.append(" and a.wpId =").append(watchPoint.getId());

sql.append(" and a.type = ").append(warnType);

sql.append(" order by collect\_time desc");

sql.append(" limit 0,3");

List<WarnModel> warnModels = null;

warnModels = this.queryObjectList(sql.toString(), WarnModel.class);

warn.setWarnType(warnTypeString);

warn.setWarnModels(warnModels);

return warn;

}

/\*

\* 根据条件查询出预警告警配置列表

\* 根据当前登录人员查询权限内的站点

\*/

public List<WarnModel> getWarnList(WarnModel model,SysUser user){

List<WarnModel> list = new ArrayList<WarnModel>();

//获得该人员权限下的所有站点的集合

List<StationModel> stations = stationDao.getStations4User(user);

//拼接INSQL语句

StringBuffer insql = new StringBuffer("0");

for(StationModel station:stations){

insql.append(",").append(station.getId());

}

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.wp\_id as wpId,b.title as wpName,a.indicator\_code as indicatorCode,c.title as indicatorName,");

sql.append(" a.warn\_lower as warnLower,a.warn\_upper as warnUpper,a.alarm\_lower as alarmLower,a.alarm\_upper as alarmUpper");

sql.append(" from aiot\_warn\_alarm\_config a,aiot\_watch\_point b,dm\_indicator c");

sql.append(" where a.wp\_id = b.id and b.isactive = 1");

sql.append(" and a.indicator\_code = c.code and c.isactive = 1");

sql.append(" and a.wp\_id in (").append(insql).append(")");

//添加查询条件

if(model!=null&&model.getWpId()>10000){

sql.append(" and a.wp\_id =").append(model.getWpId());

}

list = this.queryObjectList(sql.toString(), WarnModel.class);

return list;

}

/\*

\* 修改预警告警配置

\*/

public Result saveWarnChange(WarnModel model){

Result result = new Result();

//初始化返回结果

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("修改成功");

//判断该条配置是否冲突

StringBuffer checksql = new StringBuffer();

checksql.append(" select count(1) from aiot\_warn\_alarm\_config where wp\_id =? and indicator\_code =? and id <> ?");

Object[] checkParams = new Object[]{

model.getWpId(),model.getIndicatorCode(),model.getId()

};

int count = 0;

try {

count = this.queryForInt(checksql.toString(), checkParams);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("验证唯一性时失败");

return result;

}

if(count>0){

result.setResult(result.FAILED);

result.setMessage("已经包含同样的配置");

return result;

}

StringBuffer sql = new StringBuffer("");

sql.append(" update aiot\_warn\_alarm\_config set wp\_id=?,indicator\_code=?,warn\_lower=?,warn\_upper=?,alarm\_lower=?,alarm\_upper=? where id=?");

Object[] params = new Object[]{

model.getWpId(),model.getIndicatorCode(),model.getWarnLower(),

model.getWarnUpper(),model.getAlarmLower(),model.getAlarmUpper(),model.getId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("修改失败,请重试");

}

return result;

}

/\*

\* 保存新增预警告警配置

\*/

public Result saveNewWarn(WarnModel model){

Result result = new Result();

//初始化返回结果

result.setDotype(result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("新增成功");

StringBuffer sql = new StringBuffer("");

sql.append(" insert into aiot\_warn\_alarm\_config(wp\_id,indicator\_code,warn\_lower,warn\_upper,alarm\_lower,alarm\_upper)");

sql.append(" values(?,?,?,?,?,?) on duplicate key update warn\_lower=values(warn\_lower),warn\_upper=values(warn\_upper),alarm\_lower=values(alarm\_lower),alarm\_upper=values(alarm\_upper)");

Object[] params = new Object[]{

model.getWpId(),model.getIndicatorCode(),model.getWarnLower(),

model.getWarnUpper(),model.getAlarmLower(),model.getAlarmUpper()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("新增失败,请重试");

}

return result;

}

/\*

\* 删除选中的预警告警配置

\*/

public Result deleWarnSetting(WarnModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.DELETE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("删除成功");

//拼接删除语句

StringBuffer sql = new StringBuffer("");

sql.append(" delete from aiot\_warn\_alarm\_config where id in (").append(model.getIds()).append(")");

int res = 0 ;

try {

res = this.update(sql.toString(), null);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("删除失败");

}

return result;

}

/\*

\* 展示条件内的预警告警信息

\*/

public List<WarnValueModel> getWarnValueRows(WarnValueModel model){

List<WarnValueModel> list = new ArrayList<WarnValueModel>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.type,e.value as typeName,a.value,a.collect\_time,a.operate,case a.operate when 0 then '未操作' when 1 then '已操作' end as operateName,");

sql.append(" a.flag,a.remark,a.wpId,b.title as wpName,a.deviceId,c.name as deviceName,");

sql.append(" a.indicatorCode,d.title as indicatorName,a.min\_value as minConfig,a.max\_value as maxConfig,");

sql.append(" g.logo as unitName");

sql.append(" from aiot\_warn\_alarm\_value a,aiot\_watch\_point b,device\_catalog c,sys\_public e,");

sql.append(" dm\_indicator d left join g\_unit g on d.unitId = g.id and g.isactive = 1");

sql.append(" where a.wpid = b.id and b.isactive = 1");

sql.append(" and a.deviceId = c.id and a.indicatorCode = d.code and d.isactive = 1");

sql.append(" and a.type = e.classid and e.parentcode = '0008'");

//添加查询条件

if(model!=null){

//添加站点条件

if(model.getWpId()>10000){

sql.append(" and a.wpid = ").append(model.getWpId());

}

//添加类型条件

if(model.getType()>0){

sql.append(" and a.type = ").append(model.getType());

}

//添加时间条件

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

sql.append(" and a.collect\_time >= '").append(model.getBeginDate()).append("'");

}

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

sql.append(" and a.collect\_time <= '").append(model.getEndDate()).append("'");

}

if(model.getOperate()>=0){

sql.append(" and a.operate = ").append(model.getOperate());

}

}

//增加排序

sql.append(" order by a.collect\_time desc ");

list = this.queryObjectList(sql.toString(), WarnValueModel.class);

return list;

}

/\*

\* 为预警告警信息添加操作以及备注

\*/

public Result saveOperaValue(WarnValueModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("操作成功");

StringBuffer sql = new StringBuffer();

sql.append(" update aiot\_warn\_alarm\_value set operate=?,flag=?,remark=? where id =?");

Object[] params = new Object[]{

model.getOperate(),model.getFlag(),model.getRemark(),model.getId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("操作失败");

}

return result;

}

/\*

\* 读取预警告警中的未操作的

\*/

public Integer getNum4WarnValue(StationModel model){

Integer num = 0;

StringBuffer sql = new StringBuffer();

sql.append(" select count(1) from aiot\_warn\_alarm\_value where operate = 0");

num = this.queryForInt(sql.toString(), null);

return num;

}

/\*

\* 预警告警信息中批量操作

\*/

public Result updateAllOperaValue(WarnValueModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("批量修改成功");

//开始执行操作

StringBuffer sql = new StringBuffer();

sql.append(" update aiot\_warn\_alarm\_value set");

sql.append(" operate=").append(model.getOperate()).append(",");

sql.append(" flag='").append(model.getFlag()).append("',");

sql.append(" remark='").append(model.getRemark()).append("'");

sql.append(" where id in (").append(model.getIds()).append(")");

int res = 0;

try {

res = this.update(sql.toString(), null);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("批量修改失败");

}

return result;

}

}package com.sdocean. warn.dao;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Result;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.station.dao.StationDao;

import com.sdocean.station.model.StationModel;

import com.sdocean.users.model.SysUser;

import com.sdocean.warn.model.RangeDataModel;

import com.sdocean.warn.model.Warn4FirstModel;

import com.sdocean.warn.model.WarnModel;

import com.sdocean.warn.model.WarnValueModel;

@Component

public class RangeDataDao extends OracleEngine {

/\*\*

\* 得到查询条件下的站点的错误数据配置

\*/

public List<RangeDataModel> getRangeDataList(RangeDataModel model){

//初始化返回结果

List<RangeDataModel> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.stationid,b.title as stationname,a.deviceid,c.name as devicename,");

sql.append(" a.indicatorcode,d.title as indicatorname,a.minData,a.maxData");

sql.append(" from aiot\_range\_data a,aiot\_watch\_point b,dm\_indicator d,device\_catalog c");

sql.append(" where a.stationid = b.id and a.indicatorcode = d.code");

sql.append(" and a.deviceid = c.id");

sql.append(" and a.deviceid = c.id");

//添加查询条件

if(model!=null&&model.getStationId()>0){

sql.append(" and a.stationid = ").append(model.getStationId());

}

//添加模糊查询条件

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

sql.append(" and (d.code like '%").append(model.getIndicatorCode()).append("%'");

sql.append(" or d.title like '%").append(model.getIndicatorCode()).append("%'");

sql.append(" )");

}

//添加排序

sql.append(" order by b.ordercode,d.ordercode");

list = this.queryObjectList(sql.toString(), RangeDataModel.class);

return list;

}

/\*

\* 新增错误数据配置

\*/

public Result saveNewRangeData(RangeDataModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("新增成功");

StringBuffer sql = new StringBuffer("");

sql.append(" insert into aiot\_range\_data(stationid,indicatorcode,minData,maxData,deviceid)");

sql.append(" values(?,?,?,?,?) on duplicate key update minData=values(minData),maxData=values(maxData)");

Object[] param = new Object[]{

model.getStationId(),model.getIndicatorCode(),model.getMinData(),model.getMaxData(),model.getDeviceId()

};

try {

this.update(sql.toString(), param);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("新增失败");

}

return result;

}

/\*

\* 修改错误数据配置

\*/

public Result updateRangeData(RangeDataModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("修改成功");

StringBuffer sql = new StringBuffer();

sql.append(" update aiot\_range\_data set stationid=?,indicatorcode=?,minData=?,maxData=?,deviceid=? where id=?");

Object[] param = new Object[]{

model.getStationId(),model.getIndicatorCode(),model.getMinData(),model.getMaxData(),model.getDeviceId(),model.getId()

};

try {

this.update(sql.toString(), param);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("新增失败");

}

return result;

}

/\*

\* 删除错误数据配置

\*/

public Result deleRangeData(RangeDataModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.DELETE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("删除成功");

StringBuffer sql = new StringBuffer();

sql.append(" delete from aiot\_range\_data where id=?");

Object[] param = new Object[]{

model.getId()

};

try {

this.update(sql.toString(), param);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("删除失败");

}

return result;

}

/\*

\* 根据站点 参数查询出阈值范围

\*/

public RangeDataModel getRangeDataByStationDeviceIndicator(int stationId,int deviceId,String indicatorCode){

//初始化返回结果

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.stationid,b.title as stationname,a.deviceid,c.name as deviceName,");

sql.append(" a.indicatorcode,d.title as indicatorname,a.minData,a.maxData");

sql.append(" from aiot\_range\_data a,aiot\_watch\_point b,dm\_indicator d,device\_catalog c");

sql.append(" where a.stationid = b.id and a.indicatorcode = d.code");

sql.append(" and a.deviceid = c.id");

sql.append(" and a.stationid = ").append(stationId);

sql.append(" and a.deviceid = ").append(deviceId);

sql.append(" and a.indicatorCode='").append(indicatorCode).append("'");

sql.append(" limit 1");

RangeDataModel ed = this.queryObject(sql.toString(), RangeDataModel.class);

return ed;

}

/\*

\* 根据站点以及参数,获得该站点内所有能监测该参数的设备的量程范围

\*/

public List<RangeDataModel> getRangeDataListByStationDeviceIndicator(int stationId,int deviceId,String indicatorCode){

List<RangeDataModel> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.stationid,b.title as stationname,a.deviceid,c.name as deviceName,");

sql.append(" a.indicatorcode,d.title as indicatorname,a.minData,a.maxData");

sql.append(" from aiot\_range\_data a,aiot\_watch\_point b,dm\_indicator d,device\_catalog c");

sql.append(" where a.stationid = b.id and a.indicatorcode = d.code");

sql.append(" and a.deviceid = c.id");

if(stationId>0) {

sql.append(" and a.stationid = ").append(stationId);

}

if(deviceId>0) {

sql.append(" and a.deviceid = ").append(deviceId);

}

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

sql.append(" and a.indicatorCode='").append(indicatorCode).append("'");

}

list = this.queryObjectList(sql.toString(), RangeDataModel.class);

return list;

}

}

package com.sdocean.warn.dao;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import org.springframework.security.core.userdetails.User;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.SelectTree;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.role.model.RoleModel;

import com.sdocean.station.model.StationDeviceComm;

import com.sdocean.station.model.StationDeviceModel;

import com.sdocean.station.model.StationInfo;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.model.StationTypeModel;

import com.sdocean.users.model.SysUser;

import com.sdocean.warn.model.DeviceAlarmConfig;

import com.sdocean.warn.model.DeviceAlarmModel;

@Component

public class DeviceAlarmDao extends OracleEngine {

/\*

\* 读取设备报警码表

\*/

public List<DeviceAlarmModel> getDeviceAlarmList(DeviceAlarmModel model,List<StationModel> stations){

List<DeviceAlarmModel> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.stationid,c.title as stationname,a.deviceid,");

sql.append(" d.name as devicename,a.configid,b.title as configname,");

sql.append(" a.alarmdata,a.begintime,a.endtime");

sql.append(" from aiot\_device\_alarm a,aiot\_device\_alarm\_config b,");

sql.append(" aiot\_watch\_point c,device\_catalog d");

sql.append(" where a.configid = b.id");

sql.append(" and a.stationid = c.id and c.isactive = 1");

sql.append(" and a.deviceid = d.id");

//添加查询条件

if(model!=null&&model.getStationId()>0) {

sql.append(" and a.stationid = ").append(model.getStationId());

}else {

sql.append(" and a.stationid in (0");

for(StationModel station:stations) {

sql.append(",").append(station.getId());

}

sql.append(")");

}

//添加排序

sql.append(" order by c.ordercode,d.ordercode");

list = this.queryObjectList(sql.toString(), DeviceAlarmModel.class);

return list;

}

/\*

\* 添加设备报警码表

\*/

public Result saveNewDeviceAlarm(DeviceAlarmModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(Result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.SUCCESS);

result.setMessage("新增成功");

//初始化开始结束时间

if(model!=null) {

if(model.getBeginTime()==null||model.getBeginTime().length()<1) {

model.setBeginTime("2000-01-01 01:01:01");

}

if(model.getEndTime()==null||model.getEndTime().length()<1) {

model.setEndTime("2030-01-01 01:01:01");

}

}

//判断时间段与数据库现有的记录时间是否冲突

StringBuffer csql = new StringBuffer("");

csql.append(" select count(1) from aiot\_device\_alarm");

csql.append(" where stationid =").append(model.getStationId());

csql.append(" and deviceid =").append(model.getDeviceId());

csql.append(" and configid =").append(model.getConfigId());

csql.append(" and begintime <='").append(model.getEndTime()).append("'");

csql.append(" and endtime >='").append(model.getBeginTime()).append("'");

int cres = 0;

try {

cres = this.queryForInt(csql.toString(), null);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("检查时间规则失败");

return result;

}

if(cres>0){

result.setResult(Result.FAILED);

result.setMessage("时间设置不正确");

return result;

}

//添加数据

StringBuffer sql = new StringBuffer("");

sql.append(" insert into aiot\_device\_alarm(stationid,deviceid,configid,alarmdata,begintime,endtime)");

sql.append(" values(?,?,?,?,?,?)");

Object[] params = new Object[]{

model.getStationId(),model.getDeviceId(),model.getConfigId(),

model.getAlarmData(),model.getBeginTime(),model.getEndTime()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("新增失败");

return result;

}

return result;

}

/\*

\* 修改站点设备配置

\*/

public Result saveChangeDeviceAlaram(DeviceAlarmModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(Result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.SUCCESS);

result.setMessage("修改成功");

//初始化开始结束时间

if(model!=null) {

if(model.getBeginTime()==null||model.getBeginTime().length()<1) {

model.setBeginTime("2000-01-01 01:01:01");

}

if(model.getEndTime()==null||model.getEndTime().length()<1) {

model.setEndTime("2030-01-01 01:01:01");

}

}

//判断时间段与数据库现有的记录时间是否冲突

StringBuffer csql = new StringBuffer("");

csql.append(" select count(1) from aiot\_device\_alarm");

csql.append(" where stationid =").append(model.getStationId());

csql.append(" and deviceid =").append(model.getDeviceId());

csql.append(" and configid =").append(model.getConfigId());

csql.append(" and begintime <='").append(model.getEndTime()).append("'");

csql.append(" and endtime >='").append(model.getBeginTime()).append("'");

csql.append(" and id <> ").append(model.getId());

int cres = 0;

try {

cres = this.queryForInt(csql.toString(), null);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("检查时间规则失败");

return result;

}

if(cres>0){

result.setResult(Result.FAILED);

result.setMessage("时间设置不正确");

return result;

}

//添加数据

StringBuffer sql = new StringBuffer("");

sql.append(" update aiot\_device\_alarm set stationid=?,deviceid=?,configid=?,alarmdata=?,begintime=?,endtime=? where id=?");

Object[] params = new Object[]{

model.getStationId(),model.getDeviceId(),model.getConfigId(),

model.getAlarmData(),model.getBeginTime(),model.getEndTime(),model.getId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("修改失败");

return result;

}

return result;

}

/\*

\* 删除设备报警码

\*/

public Result deleteDeviceAlarm(DeviceAlarmModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(Result.DELETE);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.SUCCESS);

result.setMessage("删除成功");

StringBuffer sql = new StringBuffer("");

sql.append(" delete from aiot\_device\_alarm where id = ").append(model.getId());

int res = 0;

try {

res = this.update(sql.toString(), null);

} catch (Exception e) {

// TODO: handle exception

result.setResult(Result.FAILED);

result.setMessage("删除失败");

}

return result;

}

/\*

\* 获得报警类型列表

\*/

public List<DeviceAlarmConfig> getDeviceAlarmConfigList(DeviceAlarmConfig model){

List<DeviceAlarmConfig> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select id,code,title,ordercode,remark");

sql.append(" from aiot\_device\_alarm\_config");

sql.append(" where 1=1");

//添加查询条件

if(model!=null&&model.getId()>0) {

sql.append(" and id = ").append(model.getId());

}

sql.append(" order by ordercode");

list = this.queryObjectList(sql.toString(), DeviceAlarmConfig.class);

return list;

}

/\*

\* 根据站点设备获得该设备的报警类型列表

\*/

public List<DeviceAlarmModel> getDeviceAlarmListByStationDevice(DeviceAlarmModel model){

List<DeviceAlarmModel> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.stationid,c.title as stationname,a.deviceid,");

sql.append(" d.name as devicename,a.configid,b.title as configname,");

sql.append(" a.alarmdata,a.begintime,a.endtime");

sql.append(" from aiot\_device\_alarm a,aiot\_device\_alarm\_config b,");

sql.append(" aiot\_watch\_point c,device\_catalog d");

sql.append(" where a.configid = b.id");

sql.append(" and a.stationid = c.id and c.isactive = 1");

sql.append(" and a.deviceid = d.id");

//添加查询条件

if(model!=null&&model.getStationId()>0) {

sql.append(" and a.stationid = ").append(model.getStationId());

}

if(model!=null&&model.getDeviceId()>0) {

sql.append(" and a.deviceid = ").append(model.getDeviceId());

}

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

sql.append(" and a.endTime >= '").append(model.getBeginTime()).append("'");

}

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

sql.append(" and a.begintime <= '").append(model.getEndTime()).append("'");

}

//添加排序

sql.append(" order by c.ordercode,d.ordercode");

list = this.queryObjectList(sql.toString(), DeviceAlarmModel.class);

return list;

}

/\*

\* 设备超量程后,读取量程数值

\*/

public List<DeviceAlarmModel> getDeviceAlarmDataListByStationDeviceIndicator(

int stationId,int deviceId,String indicatorCode,String beginTime,String endTime){

List<DeviceAlarmModel> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.stationid,a.deviceid,a.configid,a.alarmdata,");

sql.append(" a.begintime,a.endtime,b.maxdata");

sql.append(" from aiot\_device\_alarm a,aiot\_range\_data b");

sql.append(" where a.stationid = b.stationid");

sql.append(" and a.deviceid = b.deviceid");

sql.append(" and a.configid = 1");

sql.append(" and a.stationid = ").append(stationId);

sql.append(" and a.deviceid = ").append(deviceId);

sql.append(" and a.endtime >= '").append(beginTime).append("'");

sql.append(" and a.begintime <= '").append(endTime).append("'");

list = this.queryObjectList(sql.toString(), DeviceAlarmModel.class);

return list;

}

}package com.sdocean.warn.action;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.PublicModel;

import com.sdocean.dictionary.service.PublicService;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationService;

import com.sdocean.users.model.SysUser;

import com.sdocean.warn.model.WarnModel;

import com.sdocean.warn.model.WarnValueModel;

import com.sdocean.warn.service.WarnService;

@Controller

public class WarnAction {

@Resource

WarnService warnService;

@Resource

OperationLogService logService;

@Resource

StationService stationService;

@Resource

PublicService publicService;

@RequestMapping("info\_warnsetting.do")

public ModelAndView handleRequestInternal(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/warn/warnSettingInfo");

return mav;

}

@RequestMapping(value="showWarnSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showWarnSetting(@ModelAttribute("model") WarnModel model,HttpServletRequest request,

HttpServletResponse response){

//获得当前的人员信息

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

PageResult result = new PageResult();

//为查询结果添加表头

List<UiColumn> cols = warnService.getCols4List();

result.setCols(cols);

//为查询结果添加结果

List<WarnModel> rows = warnService.getWarnList(model, user);

result.setRows(rows);

return JsonUtil.toJson(result);

}

@RequestMapping(value="saveWarnChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveWarnChange(@ModelAttribute("model") WarnModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = warnService.saveWarnChange(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="saveNewWarn.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewWarn(@ModelAttribute("model") WarnModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = warnService.saveNewWarn(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="deleWarnSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String deleWarnSetting(@ModelAttribute("model") WarnModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = warnService.deleWarnSetting(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 跳转到预警告警信息页面

\*/

@RequestMapping("info\_warnvalue.do")

public ModelAndView info\_warnvalue(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/warn/warnValueInfo");

return mav;

}

/\*

\* 初始化预警告警信息页面

\*/

@RequestMapping(value="init\_warnvalue.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String init\_warnvalue(HttpServletRequest request,

HttpServletResponse response){

//获得当前的人员信息

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

WarnValueModel model = new WarnValueModel();

//根据人员权限初始化站点列表

List<StationModel> stationList = stationService.getStations4User(user);

List<StationModel> stations = new ArrayList<StationModel>();

StationModel allStation = new StationModel();

allStation.setId(0);

allStation.setTitle("全部站点");

stations.add(allStation);

stations.addAll(stationList);

//初始化类型列表

List<PublicModel> publics = publicService.getPublicsByParent("0008");

List<PublicModel> typeList = new ArrayList<PublicModel>();

PublicModel type = new PublicModel();

type.setClassId(0);

type.setValue("全部");

typeList.add(type);

typeList.addAll(publics);

//初始化查询时间

DateFormat beginDf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

Calendar calendar = Calendar.getInstance();

calendar.setTime(new Date());

//设置结束时间

String endDate = beginDf.format(calendar.getTime());

//设置开始时间

calendar.add(Calendar.MONTH, -1);

String beginDate = beginDf.format(calendar.getTime());

model.setStationList(stations);

model.setTypeList(typeList);

model.setBeginDate(beginDate);

model.setEndDate(endDate);

return JsonUtil.toJson(model);

}

/\*

\* 展示预警告警信息

\*/

@RequestMapping(value="showWarnValues.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showWarnValues(@ModelAttribute("model") WarnValueModel model,HttpServletRequest request,

HttpServletResponse response){

//获得当前的人员信息

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

PageResult result = new PageResult();

//为查询结果添加表头

List<UiColumn> cols = warnService.getCols4WarnValue();

result.setCols(cols);

//为查询结果添加结果

List<WarnValueModel> rows = warnService.getWarnValueRows(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存操作信息

\*/

@RequestMapping(value="saveOperaValue.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveOperaValue(@ModelAttribute("model") WarnValueModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = warnService.saveOperaValue(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 预警告警信息批量修改操作信息

\*/

@RequestMapping(value="updateAllOperaValue.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String updateAllOperaValue(@ModelAttribute("model") WarnValueModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = warnService.updateAllOperaValue(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

}

package com.sdocean.warn.action;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.PublicModel;

import com.sdocean.dictionary.service.PublicService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationService;

import com.sdocean.users.model.SysUser;

import com.sdocean.warn.model.RangeDataModel;

import com.sdocean.warn.model.WarnModel;

import com.sdocean.warn.model.WarnValueModel;

import com.sdocean.warn.service.RangeDataService;

import com.sdocean.warn.service.WarnService;

@Controller

public class RangeDataAction {

@Resource

RangeDataService rangeDataService;

@Resource

OperationLogService logService;

@Resource

StationService stationService;

@Resource

PublicService publicService;

@RequestMapping("info\_rangevalue.do")

public ModelAndView info\_errorvalue(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/warn/rangeDataInfo");

return mav;

}

@RequestMapping(value="showRangeData.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showRangeData(@ModelAttribute("model") RangeDataModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果添加表头

List<UiColumn> cols = rangeDataService.getCols4RangeDataList();

result.setCols(cols);

//为查询结果添加结果

List<RangeDataModel> rows = rangeDataService.getRangeDataList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

@RequestMapping(value="updateRangeData.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String updateRangeData(@ModelAttribute("model") RangeDataModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = rangeDataService.updateRangeData(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="saveNewRangeData.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewRangeData(@ModelAttribute("model") RangeDataModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = rangeDataService.saveNewRangeData(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="deleRangeData.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String deleRangeData(@ModelAttribute("model") RangeDataModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = rangeDataService.deleRangeData(model);

//保存操作记录

logService.saveOperationLog(result, request);

return result.getMessage();

}

}package com.sdocean.warn.action;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

import com.sdocean.warn.model.DeviceAlarmConfig;

import com.sdocean.warn.model.DeviceAlarmModel;

import com.sdocean.warn.service.DeviceAlarmService;

@Controller

public class DeviceAlarmAction {

@Resource

DeviceAlarmService deviceAlarmService;

@Resource

OperationLogService logService;

@RequestMapping("info\_devicealarm.do")

public ModelAndView info\_station(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/warn/deviceAlarmInfo");

return mav;

}

/\*

\* 展示站点设备配置列表

\*/

@RequestMapping(value="showDeviceAlarmList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showDeviceAlarmList(@ModelAttribute("model") DeviceAlarmModel model,HttpServletRequest request,

HttpServletResponse response){

//获得该用户的站点权限

HttpSession session = request.getSession();

List<StationModel> stations = (List<StationModel>) session.getAttribute("stations");

PageResult page = new PageResult();

List<UiColumn> cols = deviceAlarmService.getCols4DeviceAlarmList();

List<DeviceAlarmModel> rows = deviceAlarmService.getDeviceAlarmList(model,stations);

page.setCols(cols);

page.setRows(rows);

return JsonUtil.toJson(page);

}

/\*

\* 保存新增的站点设备配置

\*/

@RequestMapping(value="saveNewDeviceAlarm.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewDeviceAlarm(@ModelAttribute("model") DeviceAlarmModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = deviceAlarmService.saveNewDeviceAlarm(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 保存修改的站点设备配置

\*/

@RequestMapping(value="saveChangeDeviceAlaram.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveChangeDeviceAlaram(@ModelAttribute("model") DeviceAlarmModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = deviceAlarmService.saveChangeDeviceAlaram(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 删除站点设备配置

\*/

@RequestMapping(value="deleteDeviceAlarm.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String deleteDeviceAlarm(@ModelAttribute("model") DeviceAlarmModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = deviceAlarmService.deleteDeviceAlarm(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="getDeviceAlarmConfigList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDeviceAlarmConfigList(@ModelAttribute("model") DeviceAlarmConfig model,HttpServletRequest request,

HttpServletResponse response){

List<DeviceAlarmConfig> list = deviceAlarmService.getDeviceAlarmConfigList(model);

return JsonUtil.toJson(list);

}

}

package com.sdocean.sms.service;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.page.model.UiColumn;

import com.sdocean.sms.dao.SmsDao;

import com.sdocean.sms.model.SmsLog;

import com.sdocean.sms.model.SmsMouldModel;

import com.sdocean.sms.model.SmsSettingModel;

import com.sdocean.users.model.SysUser;

@Service

@Transactional(rollbackFor = Exception.class, propagation = Propagation.REQUIRED)

public class SmsService {

@Resource

private SmsDao smsDao;

/\*

\* 为人员管理的查询结果添加表头

\*/

public List<UiColumn> getCols4TypeList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

//UiColumn col1 = new UiColumn("stationId", "stationId", false, "\*");

//UiColumn col2 = new UiColumn("stationName", "stationName", true, "\*");

UiColumn col3 = new UiColumn("userId", "userId", false, "\*");

UiColumn col6 = new UiColumn("用户", "userName", true, "\*");

UiColumn col7 = new UiColumn("手机号", "telephone", true, "\*");

UiColumn col8 = new UiColumn("周一", "monValue", false, "\*");

UiColumn col9 = new UiColumn("周二", "tuesValue", false, "\*");

UiColumn col10 = new UiColumn("周三", "wedValue", false, "\*");

UiColumn col11 = new UiColumn("周四", "thurValue", false, "\*");

UiColumn col12 = new UiColumn("周五", "friValue", false, "\*");

UiColumn col13 = new UiColumn("周六", "saturValue", false, "\*");

UiColumn col14 = new UiColumn("周日", "sunValue", false, "\*");

UiColumn col15 = new UiColumn("周一", "mon", true, "\*");

UiColumn col16 = new UiColumn("周二", "tues", true, "\*");

UiColumn col17 = new UiColumn("周三", "wed", true, "\*");

UiColumn col21 = new UiColumn("周四", "thur", true, "\*");

UiColumn col18 = new UiColumn("周五", "fri", true, "\*");

UiColumn col19 = new UiColumn("周六", "satur", true, "\*");

UiColumn col20= new UiColumn("周日", "sun", true, "\*");

UiColumn col22= new UiColumn("间隔发送时间", "betweenTime", true, "\*");

UiColumn col27= new UiColumn("是否接收短信", "isactiveName", true, "\*");

UiColumn col23= new UiColumn("上午起始时间", "ambegin", true, "\*");

UiColumn col24= new UiColumn("上午截止时间", "amend", true, "\*");

UiColumn col25= new UiColumn("下午起始时间", "pmbegin", true, "\*");

UiColumn col26= new UiColumn("下午截止时间", "pmend", true, "\*");

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

cols.add(col9);

cols.add(col10);

cols.add(col11);

cols.add(col12);

cols.add(col13);

cols.add(col14);

cols.add(col15);

cols.add(col16);

cols.add(col17);

cols.add(col18);

cols.add(col19);

cols.add(col20);

cols.add(col21);

cols.add(col22);

cols.add(col23);

cols.add(col24);

cols.add(col25);

cols.add(col26);

cols.add(col27);

return cols;

}

/\*

\* 查询某站点下需要接受短信的人员列表

\*/

public List<SmsSettingModel> getUsers4SmsSetting(SmsSettingModel model,SysUser user){

return smsDao.getUsers4SmsSetting(model, user);

}

/\*

\* 初始化用户下的短信配置

\*/

public SmsSettingModel infoUserSetting(SysUser user){

return smsDao.infoUserSetting(user);

}

/\*

\* 保存用户短信配置

\*/

public Result saveUserSetting(SmsSettingModel model){

return smsDao.saveUserSetting(model);

}

/\*

\* 展示短信模板

\*/

public List<UiColumn> getCols4MouldList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*");

UiColumn col2 = new UiColumn("stationId", "stationId", false, "\*");

UiColumn col3 = new UiColumn("站点", "stationName", true, "\*");

UiColumn col6 = new UiColumn("type", "type", false, "\*");

UiColumn col7 = new UiColumn("模板类型", "typeName", true, "\*");

UiColumn col8 = new UiColumn("模板", "mould", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

return cols;

}

/\*

\* 为展示模板提供结果

\*/

public List<SmsMouldModel> getSmsMoulds4Rows(SmsMouldModel model){

return smsDao.getSmsMoulds4Rows(model);

}

/\*

\* 对短信发送模板修改

\*/

public Result saveSmsMouldChange(SmsMouldModel model){

return smsDao.saveSmsMouldChange(model);

}

/\*

\* 保存新增的短信发送模板

\*/

public Result saveNewSmsMould(SmsMouldModel model){

return smsDao.saveNewSmsMould(model);

}

/\*

\* 展示短信发送记录

\*/

public List<UiColumn> getCols4LogList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*");

UiColumn col2 = new UiColumn("userId", "userId", false, "\*");

UiColumn col3 = new UiColumn("用户名", "userName", true, "\*");

UiColumn col6 = new UiColumn("手机号", "telephone", true, "\*");

UiColumn col7 = new UiColumn("短信", "message", true, "\*");

UiColumn col8 = new UiColumn("ifsend", "ifsend", false, "\*");

UiColumn col9 = new UiColumn("发送状态", "ifSendName", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

cols.add(col9);

return cols;

}

/\*

\* 查询短信发送记录,

\* 增加模糊查询

\*/

public List<SmsLog> getSmsLogs(SmsLog model){

return smsDao.getSmsLogs(model);

}

/\*

\* 保存短信发送配置

\*/

public Result saveSmsStationSetting(SmsSettingModel model){

return smsDao.saveSmsStationSetting(model);

}

}**package** com.sdocean.sms.model;

**public** **class** SmsSettingModel {

**private** **int** stationId;

**private** String stationName;

**private** **int** userId;

**private** String userIds;

**private** String userName;

**private** String telephone;

**private** **int** mon;

**private** String monValue;

**private** **int** tues;

**private** String tuesValue;

**private** **int** wed;

**private** String wedValue;

**private** **int** thur;

**private** String thurValue;

**private** **int** fri;

**private** String friValue;

**private** **int** satur;

**private** String saturValue;

**private** **int** sun;

**private** String sunValue;

**private** **int** betweenTime;

**private** **int** isactive;

**private** String isactiveName; //是否接收短信 1接收,0 不接受

**private** String ambegin;

**private** String amend;

**private** String pmbegin;

**private** String pmend;

**private** String type;

**public** String getUserIds() {

**return** userIds;

}

**public** **void** setUserIds(String userIds) {

**this**.userIds = userIds;

}

**public** String getType() {

**return** type;

}

**public** **void** setType(String type) {

**this**.type = type;

}

**public** **int** getIsactive() {

**return** isactive;

}

**public** **void** setIsactive(**int** isactive) {

**this**.isactive = isactive;

}

**public** String getIsactiveName() {

**return** isactiveName;

}

**public** **void** setIsactiveName(String isactiveName) {

**this**.isactiveName = isactiveName;

}

**public** String getPmbegin() {

**return** pmbegin;

}

**public** **void** setPmbegin(String pmbegin) {

**this**.pmbegin = pmbegin;

}

**public** String getAmbegin() {

**return** ambegin;

}

**public** **void** setAmbegin(String ambegin) {

**this**.ambegin = ambegin;

}

**public** String getAmend() {

**return** amend;

}

**public** **void** setAmend(String amend) {

**this**.amend = amend;

}

**public** String getPmend() {

**return** pmend;

}

**public** **void** setPmend(String pmend) {

**this**.pmend = pmend;

}

**public** String getTelephone() {

**return** telephone;

}

**public** **void** setTelephone(String telephone) {

**this**.telephone = telephone;

}

**public** **int** getStationId() {

**return** stationId;

}

**public** **void** setStationId(**int** stationId) {

**this**.stationId = stationId;

}

**public** String getStationName() {

**return** stationName;

}

**public** **void** setStationName(String stationName) {

**this**.stationName = stationName;

}

**public** **int** getUserId() {

**return** userId;

}

**public** **void** setUserId(**int** userId) {

**this**.userId = userId;

}

**public** String getUserName() {

**return** userName;

}

**public** **void** setUserName(String userName) {

**this**.userName = userName;

}

**public** **int** getMon() {

**return** mon;

}

**public** **void** setMon(**int** mon) {

**this**.mon = mon;

}

**public** String getMonValue() {

**return** monValue;

}

**public** **void** setMonValue(String monValue) {

**this**.monValue = monValue;

}

**public** **int** getTues() {

**return** tues;

}

**public** **void** setTues(**int** tues) {

**this**.tues = tues;

}

**public** String getTuesValue() {

**return** tuesValue;

}

**public** **void** setTuesValue(String tuesValue) {

**this**.tuesValue = tuesValue;

}

**public** **int** getWed() {

**return** wed;

}

**public** **void** setWed(**int** wed) {

**this**.wed = wed;

}

**public** String getWedValue() {

**return** wedValue;

}

**public** **void** setWedValue(String wedValue) {

**this**.wedValue = wedValue;

}

**public** **int** getThur() {

**return** thur;

}

**public** **void** setThur(**int** thur) {

**this**.thur = thur;

}

**public** String getThurValue() {

**return** thurValue;

}

**public** **void** setThurValue(String thurValue) {

**this**.thurValue = thurValue;

}

**public** **int** getFri() {

**return** fri;

}

**public** **void** setFri(**int** fri) {

**this**.fri = fri;

}

**public** String getFriValue() {

**return** friValue;

}

**public** **void** setFriValue(String friValue) {

**this**.friValue = friValue;

}

**public** **int** getSatur() {

**return** satur;

}

**public** **void** setSatur(**int** satur) {

**this**.satur = satur;

}

**public** String getSaturValue() {

**return** saturValue;

}

**public** **void** setSaturValue(String saturValue) {

**this**.saturValue = saturValue;

}

**public** **int** getSun() {

**return** sun;

}

**public** **void** setSun(**int** sun) {

**this**.sun = sun;

}

**public** String getSunValue() {

**return** sunValue;

}

**public** **void** setSunValue(String sunValue) {

**this**.sunValue = sunValue;

}

**public** **int** getBetweenTime() {

**return** betweenTime;

}

**public** **void** setBetweenTime(**int** betweenTime) {

**this**.betweenTime = betweenTime;

}

}

**package** com.sdocean.sms.model;

**public** **class** SmsMouldModel {

**private** **int** id;

**private** **int** stationId;

**private** String stationName;

**private** **int** type;

**private** String typeName;

**private** String mould;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **int** getStationId() {

**return** stationId;

}

**public** **void** setStationId(**int** stationId) {

**this**.stationId = stationId;

}

**public** String getStationName() {

**return** stationName;

}

**public** **void** setStationName(String stationName) {

**this**.stationName = stationName;

}

**public** **int** getType() {

**return** type;

}

**public** **void** setType(**int** type) {

**this**.type = type;

}

**public** String getTypeName() {

**return** typeName;

}

**public** **void** setTypeName(String typeName) {

**this**.typeName = typeName;

}

**public** String getMould() {

**return** mould;

}

**public** **void** setMould(String mould) {

**this**.mould = mould;

}

}

**package** com.sdocean.sms.model;

**public** **class** SmsLog {

**private** **int** id;

**private** **int** userId;

**private** String userName;

**private** String telephone;

**private** String title;

**private** String message;

**private** String sendTime;

**private** String createTime;

**private** **int** ifSend;

**private** String ifSendName;

**private** String error;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **int** getUserId() {

**return** userId;

}

**public** **void** setUserId(**int** userId) {

**this**.userId = userId;

}

**public** String getUserName() {

**return** userName;

}

**public** **void** setUserName(String userName) {

**this**.userName = userName;

}

**public** String getTelephone() {

**return** telephone;

}

**public** **void** setTelephone(String telephone) {

**this**.telephone = telephone;

}

**public** String getTitle() {

**return** title;

}

**public** **void** setTitle(String title) {

**this**.title = title;

}

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** String getSendTime() {

**return** sendTime;

}

**public** **void** setSendTime(String sendTime) {

**this**.sendTime = sendTime;

}

**public** String getCreateTime() {

**return** createTime;

}

**public** **void** setCreateTime(String createTime) {

**this**.createTime = createTime;

}

**public** **int** getIfSend() {

**return** ifSend;

}

**public** **void** setIfSend(**int** ifSend) {

**this**.ifSend = ifSend;

}

**public** String getIfSendName() {

**return** ifSendName;

}

**public** **void** setIfSendName(String ifSendName) {

**this**.ifSendName = ifSendName;

}

**public** String getError() {

**return** error;

}

**public** **void** setError(String error) {

**this**.error = error;

}

}

package com.sdocean.sms.dao;

import java.util.ArrayList;

import java.util.List;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Result;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.role.model.RoleModel;

import com.sdocean.sms.model.SmsLog;

import com.sdocean.sms.model.SmsMouldModel;

import com.sdocean.sms.model.SmsSettingModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.users.model.SysUser;

import freemarker.template.utility.StringUtil;

@Component

public class SmsDao extends OracleEngine{

/\*

\* 查询某站点下需要接受短信的人员列表

\*/

public List<SmsSettingModel> getUsers4SmsSetting(SmsSettingModel model,SysUser user){

List<SmsSettingModel> list = new ArrayList<SmsSettingModel>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.userid,' 默认' as userName,'' as telephone,");

sql.append(" a.betweentime,date\_format(a.ambegin,'%H:%m:%d') as ambegin,date\_format(a.amend,'%H:%m:%d') as amend,");

sql.append("date\_format(a.pmbegin,'%H:%m:%d') as pmbegin,date\_format(a.pmend,'%H:%m:%d') as pmend,");

sql.append(" a.mon,a.tues,a.wed,a.thur,a.fri,a.satur,a.sun,");

sql.append(" case a.isactive when 1 then '接收' else '不接收' end as isactiveName");

sql.append(" from sys\_sms\_userrole a");

sql.append(" where a.userid = 0");

sql.append(" union all");

sql.append(" select a.userid,b.realname as userName,b.telephone,");

sql.append(" a.betweentime,date\_format(a.ambegin,'%H:%m:%d') as ambegin,date\_format(a.amend,'%H:%m:%d') as amend,");

sql.append("date\_format(a.pmbegin,'%H:%m:%d') as pmbegin,date\_format(a.pmend,'%H:%m:%d') as pmend,");

sql.append(" a.mon,a.tues,a.wed,a.thur,a.fri,a.satur,a.sun,");

sql.append(" case a.isactive when 1 then '接收' else '不接收' end as isactiveName");

sql.append(" from sys\_sms\_userrole a,sys\_user b");

sql.append(" where a.userid = b.id and b.isactive = 1");

//添加查询条件

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

sql.append(" and (b.username like '%").append(model.getUserName()).append("%'");

sql.append(" or b.realname like '%").append(model.getUserName()).append("%')");

}

//添加排序

sql.append(" order by userName");

list = this.queryObjectList(sql.toString(), SmsSettingModel.class);

return list;

}

/\*

\* 初始化用户下的短信配置

\*/

public SmsSettingModel infoUserSetting(SysUser user){

SmsSettingModel result = new SmsSettingModel();

//读取发送规则配置

StringBuffer rolesql = new StringBuffer("");

rolesql.append(" select userid,date\_format(ambegin,'%H:%m:%d') as ambegin,date\_format(amend,'%H:%m:%d') as amend,date\_format(pmbegin,'%H:%m:%d') as pmbegin,date\_format(pmend,'%H:%m:%d') as pmend,");

rolesql.append("mon,tues,wed,thur,fri,satur,sun,betweentime,isactive from sys\_sms\_userrole where userid =").append(user.getId());

result = this.queryObject(rolesql.toString(), SmsSettingModel.class);

if(result==null){

StringBuffer role0sql = new StringBuffer("");

role0sql.append(" select userid,date\_format(ambegin,'%H:%m:%d') as ambegin,date\_format(amend,'%H:%m:%d') as amend,date\_format(pmbegin,'%H:%m:%d') as pmbegin,date\_format(pmend,'%H:%m:%d') as pmend,");

role0sql.append(" mon,tues,wed,thur,fri,satur,sun,betweentime,isactive from sys\_sms\_userrole where userid = 0");

result = this.queryObject(role0sql.toString(), SmsSettingModel.class);

}

return result;

}

/\*

\* 保存用户短信配置

\*/

public Result saveUserSetting(SmsSettingModel model){

//对返回结果进行初始化

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("保存成功");

//保存配置信息

//保存发送时间规则设置

StringBuffer roleSql = new StringBuffer("");

roleSql.append(" insert into sys\_sms\_userrole(userid,ambegin,amend,pmbegin,pmend,mon,tues,wed,thur,fri,satur,sun,betweentime,isactive)");

roleSql.append(" values(?,?,?,?,?,?,?,?,?,?,?,?,?,?)");

roleSql.append(" on duplicate key update ambegin=values(ambegin),amend=values(amend),pmbegin=values(pmbegin),");

roleSql.append(" pmend=values(pmend),mon=values(mon),tues=values(tues),wed=values(wed),thur=values(thur),fri=values(fri),satur=values(satur),sun=values(sun),");

roleSql.append(" betweentime=values(betweentime),isactive=values(isactive)");

int roleres = 0;

Object[] roleParams = new Object[]{

model.getUserId(),model.getAmbegin(),model.getAmend(),model.getPmbegin(),

model.getPmend(),model.getMon(),model.getTues(),model.getWed(),model.getThur(),

model.getFri(),model.getSatur(),model.getSun(),model.getBetweenTime(),model.getIsactive()

};

try {

roleres = this.update(roleSql.toString(), roleParams);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("保存失败");

return result;

}

return result;

}

/\*

\* 查询符合条件的短信模板列表

\*/

public List<SmsMouldModel> getSmsMoulds4Rows(SmsMouldModel model){

List<SmsMouldModel> list = new ArrayList<SmsMouldModel>();

if(model!=null&&model.getStationId()<10000){

//首先添加默认站点stationid = 0

StringBuffer first = new StringBuffer("");

first.append(" select a.id,a.stationid,'默认站点' as stationName,a.type,c.value as typename,a.mould");

first.append(" from sys\_sms\_mould a,sys\_public c");

first.append(" where c.parentcode = '0010' and a.type = c.classid");

first.append(" and a.stationid = 0");

//添加查询条件

if(model.getType()>0){

first.append(" and a.type =").append(model.getType());

}

list = this.queryObjectList(first.toString(), SmsMouldModel.class);

}

//查询非默认站点的列表

List<SmsMouldModel> nlist = new ArrayList<SmsMouldModel>();

StringBuffer sql = new StringBuffer();

sql.append(" select a.id,a.stationid,b.title as stationName,a.type,c.value as typename,a.mould");

sql.append(" from sys\_sms\_mould a ,aiot\_watch\_point b,sys\_public c");

sql.append(" where a.stationid = b.id and b.isactive = 1 and c.parentcode = '0010' and a.type = c.classid");

//添加查询条件

if(model!=null&&model.getStationId()>10000){

sql.append(" and a.stationid = ").append(model.getStationId());

}

if(model!=null&model.getType()>0){

sql.append(" and a.type = ").append(model.getType());

}

nlist = this.queryObjectList(sql.toString(), SmsMouldModel.class);

list.addAll(nlist);

return list;

}

/\*

\* 对短信发送模板修改

\*/

public Result saveSmsMouldChange(SmsMouldModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("修改成功");

//验证station以及type唯一性

StringBuffer checksql = new StringBuffer("");

checksql.append(" select count(1) from sys\_sms\_mould where stationid =").append(model.getStationId());

checksql.append(" and type = ").append(model.getType()).append(" and id <>").append(model.getId());

int chect =0;

try {

chect = this.queryForInt(checksql.toString(), null);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("站点类型唯一性检查时失败");

return result;

}

if(chect>0){

result.setResult(result.FAILED);

result.setMessage("违反站点,类型唯一性原则");

return result;

}

//提交数据

StringBuffer sql = new StringBuffer("");

sql.append(" update sys\_sms\_mould set stationid=?,type=?,mould=? where id=?");

Object[] params = new Object[]{

model.getStationId(),model.getType(),model.getMould(),model.getId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("修改失败");

}

return result;

}

/\*

\* 保存新增的短信发送模板

\*/

public Result saveNewSmsMould(SmsMouldModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("新增成功");

//验证station以及type唯一性

StringBuffer checksql = new StringBuffer("");

checksql.append(" select count(1) from sys\_sms\_mould where stationid =").append(model.getStationId());

checksql.append(" and type = ").append(model.getType());

int chect =0;

try {

chect = this.queryForInt(checksql.toString(), null);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("站点类型唯一性检查时失败");

return result;

}

if(chect>0){

result.setResult(result.FAILED);

result.setMessage("违反站点,类型唯一性原则");

return result;

}

//提交数据

StringBuffer sql = new StringBuffer("");

sql.append(" insert into sys\_sms\_mould(stationid,type,mould) values(?,?,?)");

sql.append(" on duplicate key update stationId=values(stationid),type=values(type),mould=values(mould)");

Object[] params = new Object[]{

model.getStationId(),model.getType(),model.getMould()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("新增失败");

}

return result;

}

/\*

\* 查询短信发送记录,

\* 增加模糊查询

\*/

public List<SmsLog> getSmsLogs(SmsLog model){

List<SmsLog> list = new ArrayList<>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.userid,b.realname as username,a.telephone,a.title,a.message,a.sendtime,");

sql.append(" a.createtime,a.ifsend,case a.ifsend when 0 then '未发送' else '发送' end as ifSendName,a.error");

sql.append(" from sys\_sms a,sys\_user b");

sql.append(" where a.userid = b.id");

//增加模糊查询条件

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

sql.append(" and (b.realname like '%").append(model.getUserName()).append("%' ");

sql.append(" or a.telephone like '%").append(model.getUserName()).append("%' )");

}

//增加排序

sql.append(" order by createtime desc ");

list = this.queryObjectList(sql.toString(), SmsLog.class);

return list;

}

/\*

\* 保存短信发送配置

\*/

public Result saveSmsStationSetting(SmsSettingModel model){

//初始化返回结果

Result result = new Result();

result.setDotype(Result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.SUCCESS);

result.setMessage("保存成功");

//删除该站点,该类型的相关短信接收配置

StringBuffer dsql = new StringBuffer("");

dsql.append(" delete from sys\_sms\_station\_user ");

dsql.append(" where stationid = ").append(model.getStationId());

dsql.append(" and type = ").append(model.getType());

int dres = 0;

try {

dres = this.update(dsql.toString(), null);

} catch (Exception e) {

e.printStackTrace();

result.setResult(Result.FAILED);

result.setMessage("删除原有数据时失败");

return result;

}

//添加短信接收配置

StringBuffer asql = new StringBuffer("");

asql.append(" insert into sys\_sms\_station\_user(stationid,userid,type)");

asql.append(" values(0,0,0)");

String userIds = model.getUserIds();

String[] users = userIds.split(",");

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

asql.append(",(").append(model.getStationId()).append(",").append(users[i]).append(",").append(model.getType()).append(")");

}

asql.append(" on duplicate key update userid = values(userid)");

int ares = 0;

try {

ares = this.update(asql.toString(), null);

} catch (Exception e) {

e.printStackTrace();

result.setResult(Result.FAILED);

result.setMessage("添加短信接收配置时失败");

}

return result;

}

}

package com.sdocean.sms.action;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.menu.model.CurrMenu;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.sms.model.SmsLog;

import com.sdocean.sms.model.SmsMouldModel;

import com.sdocean.sms.model.SmsSettingModel;

import com.sdocean.sms.service.SmsService;

import com.sdocean.users.model.SysUser;

@Controller

public class SmsAction {

@Resource

SmsService smsService;

@Resource

OperationLogService logService;

@RequestMapping("info\_smssetting.do")

public ModelAndView info\_smssetting(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/sms/smsSettingInfo");

return mav;

}

@RequestMapping("info\_smssetting4user.do")

public ModelAndView info\_smssetting4user(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/sms/smsSetting4userInfo");

CurrMenu menu = new CurrMenu();

menu.setCurl("info\_smssetting4user.do");

HttpSession session = request.getSession();

session.setAttribute("currMenu", menu);

return mav;

}

/\*

\* 跳转到短信发送记录页面

\*/

@RequestMapping("info\_smslog.do")

public ModelAndView info\_smslog(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/sms/smsLogInfo");

return mav;

}

/\*

\* 展示短信配置中,某站点的人员列表

\*/

@RequestMapping(value="showUsers4SmsSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showUsers4SmsSetting(@ModelAttribute("model") SmsSettingModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult page = new PageResult();

//获得当前的人员信息

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

//得到展示表格的表头

List<UiColumn> cols = smsService.getCols4TypeList();

//得到展示表格的结果集

List<SmsSettingModel> rows = smsService.getUsers4SmsSetting(model, user);

page.setCols(cols);

page.setRows(rows);

return JsonUtil.toJson(page);

}

/\*

\* 初始化当前用户的短信配置

\*/

@RequestMapping(value="getSmsSetting4User.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getSmsSetting4User(HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

SysUser model = (SysUser) session.getAttribute("user");

SmsSettingModel setting = new SmsSettingModel();

setting = smsService.infoUserSetting(model);

return JsonUtil.toJson(setting);

}

/\*

\* 初始化用户下的短信配置

\*/

@RequestMapping(value="infoUserSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String infoUserSetting(@ModelAttribute("model") SysUser model,HttpServletRequest request,

HttpServletResponse response){

SmsSettingModel setting = new SmsSettingModel();

setting = smsService.infoUserSetting(model);

return JsonUtil.toJson(setting);

}

/\*

\* 保存用户短信配置

\*/

@RequestMapping(value="saveUserSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveUserSetting(@ModelAttribute("model") SmsSettingModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = smsService.saveUserSetting(model);

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 跳转到短信模板设置页面

\*/

@RequestMapping("info\_smsmould.do")

public ModelAndView info\_smsmould(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/sms/smsMouldInfo");

return mav;

}

/\*

\* 展示模板列表

\*/

@RequestMapping(value="showSmsMouldList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showSmsMouldList(@ModelAttribute("model") SmsMouldModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult page = new PageResult();

//得到展示表格的表头

List<UiColumn> cols = smsService.getCols4MouldList();

//得到展示表格的结果集

List<SmsMouldModel> rows = smsService.getSmsMoulds4Rows(model);

page.setCols(cols);

page.setRows(rows);

return JsonUtil.toJson(page);

}

/\*

\* 保存短信模板修改

\*/

@RequestMapping(value="saveSmsMouldChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveSmsMouldChange(@ModelAttribute("model") SmsMouldModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = smsService.saveSmsMouldChange(model);

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 保存短信模板新增

\*/

@RequestMapping(value="saveNewSmsMould.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewSmsMould(@ModelAttribute("model") SmsMouldModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = smsService.saveNewSmsMould(model);

logService.saveOperationLog(result, request);

return result.getMessage();

}

@RequestMapping(value="saveSmsSetting4User.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveSmsSetting4User(@ModelAttribute("model") SmsSettingModel model,HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

model.setUserId(user.getId());

Result result = new Result();

result = smsService.saveUserSetting(model);

return result.getMessage();

}

/\*

\* 得到短信发送记录列表

\*/

@RequestMapping(value="getSmsLogs.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getSmsLogs(@ModelAttribute("model") SmsLog model,HttpServletRequest request,

HttpServletResponse response){

PageResult page = new PageResult();

List<UiColumn> cols = smsService.getCols4LogList();

List<SmsLog> rows = smsService.getSmsLogs(model);

page.setCols(cols);

page.setRows(rows);

return JsonUtil.toJson(page);

}

/\*

\* 跳转到短信发送配置

\*/

@RequestMapping("info\_smsstation.do")

public ModelAndView info\_smsstation(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/sms/smsStationInfo");

return mav;

}

package com.sdocean.security;

import java.util.Collection;

import java.util.Iterator;

import org.springframework.security.access.AccessDecisionManager;

import org.springframework.security.access.AccessDeniedException;

import org.springframework.security.access.ConfigAttribute;

import org.springframework.security.authentication.InsufficientAuthenticationException;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.GrantedAuthority;

public class MyAccessDecisionManager implements AccessDecisionManager {

public void decide(Authentication authentication, Object object, Collection<ConfigAttribute> configAttributes) throws AccessDeniedException, InsufficientAuthenticationException {

Authentication authentication1= authentication;

if(configAttributes == null) {

return;

}

//所请求的资源拥有的权限(这里一个资源只对应一个权限)

Iterator<ConfigAttribute> iterator = configAttributes.iterator();

while(iterator.hasNext()) {

ConfigAttribute configAttribute = iterator.next();

//访问所请求资源所需要的权限

String needPermission = configAttribute.getAttribute();

//用户所拥有的权限authentication

for(GrantedAuthority ga : authentication.getAuthorities()) {

if(needPermission.equals(ga.getAuthority())) {

return;

}

}

}

//没有权限

throw new AccessDeniedException(" 没有权限访问！ ");

}

public boolean supports(ConfigAttribute attribute) {

// TODO Auto-generated method stub

return true;

}

public boolean supports(Class<?> clazz) {

// TODO Auto-generated method stub

return true;

}

}package com.sdocean.security;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import org.springframework.security.access.SecurityMetadataSource;

import org.springframework.security.access.intercept.AbstractSecurityInterceptor;

import org.springframework.security.access.intercept.InterceptorStatusToken;

import org.springframework.security.web.FilterInvocation;

import org.springframework.security.web.access.intercept.FilterInvocationSecurityMetadataSource;

public class MySecurityFilter extends AbstractSecurityInterceptor implements Filter {

//与applicationContext-security.xml里的myFilter的属性securityMetadataSource对应，

//其他的两个组件，已经在AbstractSecurityInterceptor定义

private FilterInvocationSecurityMetadataSource securityMetadataSource;

@Override

public SecurityMetadataSource obtainSecurityMetadataSource() {

return this.securityMetadataSource;

}

public void doFilter(ServletRequest request, ServletResponse response,

FilterChain chain) throws IOException, ServletException {

FilterInvocation fi = new FilterInvocation(request, response, chain);

invoke(fi);

}

private void invoke(FilterInvocation fi) throws IOException, ServletException {

// object为FilterInvocation对象

//super.beforeInvocation(fi);源码

//1.获取请求资源的权限

//执行Collection<ConfigAttribute> attributes = SecurityMetadataSource.getAttributes(object);

//2.是否拥有权限

//this.accessDecisionManager.decide(authenticated, object, attributes);

InterceptorStatusToken token = super.beforeInvocation(fi);

try {

fi.getChain().doFilter(fi.getRequest(), fi.getResponse());

} finally {

super.afterInvocation(token, null);

}

}

public FilterInvocationSecurityMetadataSource getSecurityMetadataSource() {

return securityMetadataSource;

}

public void setSecurityMetadataSource(FilterInvocationSecurityMetadataSource securityMetadataSource) {

this.securityMetadataSource = securityMetadataSource;

}

public void init(FilterConfig arg0) throws ServletException {

// TODO Auto-generated method stub

}

public void destroy() {

// TODO Auto-generated method stub

}

@Override

public Class<? extends Object> getSecureObjectClass() {

//下面的MyAccessDecisionManager的supports方面必须放回true,否则会提醒类型错误

return FilterInvocation.class;

}

}package com.sdocean.security;

import java.util.ArrayList;

import java.util.Collection;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.security.access.ConfigAttribute;

import org.springframework.security.access.SecurityConfig;

import org.springframework.security.web.FilterInvocation;

import org.springframework.security.web.access.intercept.FilterInvocationSecurityMetadataSource;

import com.sdocean.resources.dto.Resources;

import com.sdocean.resources.service.ResourcesManager;

//1 加载资源与权限的对应关系

public class MySecurityMetadataSource implements FilterInvocationSecurityMetadataSource {

//由spring调用

public MySecurityMetadataSource(ResourcesManager resourcesManager) {

this.resourcesManager = resourcesManager;

loadResourceDefine();

}

private ResourcesManager resourcesManager;

private static Map<String, Collection<ConfigAttribute>> resourceMap = null;

public void setResourcesDao(ResourcesManager resourcesManager) {

this.resourcesManager = resourcesManager;

}

public Collection<ConfigAttribute> getAllConfigAttributes() {

// TODO Auto-generated method stub

return null;

}

public boolean supports(Class<?> clazz) {

// TODO Auto-generated method stub

return true;

}

//加载所有资源与权限的关系

private void loadResourceDefine() {

if(resourceMap == null) {

resourceMap = new HashMap<String, Collection<ConfigAttribute>>();

List<Resources> resources = this.resourcesManager.findAll();

for (Resources resource : resources) {

Collection<ConfigAttribute> configAttributes = new ArrayList<ConfigAttribute>();

//以权限名封装为Spring的security Object

ConfigAttribute configAttribute = new SecurityConfig(resource.getAUTHKEY());

configAttributes.add(configAttribute);

resourceMap.put(resource.getResUrl(), configAttributes);

}

}

// Set<Entry<String, Collection<ConfigAttribute>>> resourceSet = resourceMap.entrySet();

// Iterator<Entry<String, Collection<ConfigAttribute>>> iterator = resourceSet.iterator();

}

//返回所请求资源所需要的权限

public Collection<ConfigAttribute> getAttributes(Object object) throws IllegalArgumentException {

String requestUrl = ((FilterInvocation) object).getRequestUrl();

if(resourceMap == null) {

loadResourceDefine();

}

return resourceMap.get(requestUrl);

}

}package com.sdocean.security;

import java.util.Collection;

import java.util.HashSet;

import java.util.Set;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.userdetails.User;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import com.sdocean.users.model.SysUser;

import com.sdocean.users.service.UsersManager;

public class MyUserDetailServiceImpl implements UserDetailsService {

private UsersManager usersManager;

public void setUsersManager(UsersManager usersManager) {

this.usersManager = usersManager;

}

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

SysUser users = this.usersManager.getUsersByAccount(username);

if(users == null) {

throw new UsernameNotFoundException(username);

}

Collection<GrantedAuthority> grantedAuths = obtionGrantedAuthorities(users);

boolean enables = true;

boolean accountNonExpired = true;

boolean credentialsNonExpired = true;

boolean accountNonLocked = true;

User userdetail = new User(users.getUserName(), users.getPassword(), enables, accountNonExpired, credentialsNonExpired, accountNonLocked, grantedAuths);

return userdetail;

}

//取得用户的权限

private Set<GrantedAuthority> obtionGrantedAuthorities(SysUser user) {

Set<GrantedAuthority> authSet = new HashSet<GrantedAuthority>();

/\*List<Authority> authoritysByUserAccount = this.authorityManager.getAuthoritysByUserAccount(user);

for (Authority authority : authoritysByUserAccount) {

authSet.add(new SimpleGrantedAuthority(authority.getAuthKey()));

}\*/

return authSet;

}

}package com.sdocean.security;

import java.io.IOException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.commons.lang.StringUtils;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.authentication.AuthenticationServiceException;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.AuthenticationException;

import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.log.service.SysLoginLogService;

import com.sdocean.users.model.SysUser;

import com.sdocean.users.service.UsersManager;

public class MyUsernamePasswordAuthenticationFilter extends UsernamePasswordAuthenticationFilter{

public static final String VALIDATE\_CODE = "validateCode";

public static final String USERNAME = "username";

public static final String PASSWORD = "password";

private UsersManager usersManager;

public void setUsersManager(UsersManager usersManager) {

this.usersManager = usersManager;

}

private SysLoginLogService loginLogService;

public void setLoginLogService(SysLoginLogService loginLogService){

this.loginLogService = loginLogService;

}

@Override

public Authentication attemptAuthentication(HttpServletRequest request, HttpServletResponse response) throws AuthenticationException {

if (!request.getMethod().equals("POST")) {

throw new AuthenticationServiceException("Authentication method not supported: " + request.getMethod());

}

//检测验证码

// checkValidateCode(request);

String username = obtainUsername(request);

String password = obtainPassword(request);

//验证用户账号与密码是否对应

username = username.trim();

SysUser users = this.usersManager.getUsersByAccount(username);

if(users == null || !users.getPassword().equals(password)) {

throw new AuthenticationServiceException("用户名或者密码错误！");

}

//将用户信息存入到session中

HttpSession session = request.getSession();

session.setAttribute("user", users);

UsernamePasswordAuthenticationToken authRequest = new UsernamePasswordAuthenticationToken(username, password);

// 允许子类设置详细属性

setDetails(request, authRequest);

//loginLogService.saveSysLoginLog(request);

return this.getAuthenticationManager().authenticate(authRequest);

}

protected void checkValidateCode(HttpServletRequest request) {

HttpSession session = request.getSession();

String sessionValidateCode = obtainSessionValidateCode(session);

//让上一次的验证码失效

session.setAttribute(VALIDATE\_CODE, null);

String validateCodeParameter = obtainValidateCodeParameter(request);

if (StringUtils.isEmpty(validateCodeParameter) || !sessionValidateCode.equalsIgnoreCase(validateCodeParameter)) {

throw new AuthenticationServiceException("验证码错误！");

}

}

private String obtainValidateCodeParameter(HttpServletRequest request) {

Object obj = request.getParameter(VALIDATE\_CODE);

return null == obj ? "" : obj.toString();

}

protected String obtainSessionValidateCode(HttpSession session) {

Object obj = session.getAttribute(VALIDATE\_CODE);

return null == obj ? "" : obj.toString();

}

@Override

protected String obtainUsername(HttpServletRequest request) {

Object obj = request.getParameter(USERNAME);

return null == obj ? "" : obj.toString();

}

@Override

protected String obtainPassword(HttpServletRequest request) {

Object obj = request.getParameter(PASSWORD);

return null == obj ? "" : obj.toString();

}

}package com.sdocean.security;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.Authentication;

import org.springframework.security.web.authentication.SavedRequestAwareAuthenticationSuccessHandler;

import org.springframework.security.web.savedrequest.RequestCache;

public class SavedRequestAwareAuthenticationSuccessHandlerText extends SavedRequestAwareAuthenticationSuccessHandler{

@Override

public void onAuthenticationSuccess(HttpServletRequest request,

HttpServletResponse response, Authentication authentication)

throws ServletException, IOException {

// TODO Auto-generated method stub

super.onAuthenticationSuccess(request, response, authentication);

}

@Override

public void setRequestCache(RequestCache requestCache) {

// TODO Auto-generated method stub

super.setRequestCache(requestCache);

}

}

package com.sdocean.role.service;

import java.util.ArrayList;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.page.model.UiColumn;

import com.sdocean.role.dao.RoleDao;

import com.sdocean.role.model.RoleModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class RoleService {

@Autowired

private RoleDao roleDao;

/\*

\* 获得条件内的角色列表

\*/

public List<RoleModel> getRoleList(RoleModel role){

return roleDao.getRoleList(role);

}

/\*

\* 获得角色查询的表头

\*/

public List<UiColumn> getCols4List(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*");

UiColumn col2 = new UiColumn("角色名称", "name", true, "\*");

UiColumn col3 = new UiColumn("状态", "isactiveName", true, "\*");

UiColumn col4 = new UiColumn("备注", "remark", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col4);

return cols;

}

/\*

\* 保存角色-人员-菜单

\*

\*/

public Result saveRoleUserMenu(RoleModel role){

return roleDao.saveRoleUserMenu(role);

}

/\*

\* 保存角色-人员-菜单

\*

\*/

public Result saveRoleUserStation(RoleModel role){

return roleDao.saveRoleUserStation(role);

}

/\*

\* 保存新建角色

\*/

public Result saveAddRole(RoleModel role){

return roleDao.saveAddRole(role);

}

/\*

\* 保存编辑角色

\*/

public Result saveEditRole(RoleModel role){

return roleDao.saveEditRole(role);

}

/\*

\* 删除选中的角色

\*/

public Result deleRole(RoleModel role){

return roleDao.deleRole(role);

}

/\*

\* 保存首页角色权限

\*/

public Result saveFirstMenuRole(RoleModel role){

return roleDao.saveFirstMenuRole(role);

}

}

/\*

\* 保存短信发送配置

\*/

@RequestMapping(value="saveSmsStationSetting.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveSmsStationSetting(@ModelAttribute("model") SmsSettingModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = smsService.saveSmsStationSetting(model);

logService.saveOperationLog(result, request);

return JsonUtil.toJson(result);

}

}

package com.sdocean.dictionary.model;

public class PublicModel {

private int id;

private String parentCode;

private int classId;

private String classCode;

private String className;

private String value;

private String remark;

private String orderCode;

private String ids;

public String getIds() {

return ids;

}

public void setIds(String ids) {

this.ids = ids;

}

public String getOrderCode() {

return orderCode;

}

public void setOrderCode(String orderCode) {

this.orderCode = orderCode;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getParentCode() {

return parentCode;

}

public void setParentCode(String parentCode) {

this.parentCode = parentCode;

}

public int getClassId() {

return classId;

}

public void setClassId(int classId) {

this.classId = classId;

}

public String getClassCode() {

return classCode;

}

public void setClassCode(String classCode) {

this.classCode = classCode;

}

public String getClassName() {

return className;

}

public void setClassName(String className) {

this.className = className;

}

public String getValue() {

return value;

}

public void setValue(String value) {

this.value = value;

}

public String getRemark() {

return remark;

}

public void setRemark(String remark) {

this.remark = remark;

}

}

package com.sdocean.dictionary.model;

public class UnitGroupModel {

private int id;

private String code;

private String name;

private String description;

private int isactive;

private String isactiveName;

public String getIsactiveName() {

return isactiveName;

}

public void setIsactiveName(String isactiveName) {

this.isactiveName = isactiveName;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public int getIsactive() {

return isactive;

}

public void setIsactive(int isactive) {

this.isactive = isactive;

}

}

package com.sdocean.dictionary.model;

public class UnitModel {

private int id;

private String code;

private int pid;

private int groupId;

private String groupName;

private String name;

private String description;

private String logo;

private int isactive;

private String isactiveName;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public int getPid() {

return pid;

}

public void setPid(int pid) {

this.pid = pid;

}

public int getGroupId() {

return groupId;

}

public void setGroupId(int groupId) {

this.groupId = groupId;

}

public String getGroupName() {

return groupName;

}

public void setGroupName(String groupName) {

this.groupName = groupName;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getLogo() {

return logo;

}

public void setLogo(String logo) {

this.logo = logo;

}

public int getIsactive() {

return isactive;

}

public void setIsactive(int isactive) {

this.isactive = isactive;

}

public String getIsactiveName() {

return isactiveName;

}

public void setIsactiveName(String isactiveName) {

this.isactiveName = isactiveName;

}

}

package com.sdocean.dictionary.model;

public class UnitModel {

private int id;

private String code;

private int pid;

private int groupId;

private String groupName;

private String name;

private String description;

private String logo;

private int isactive;

private String isactiveName;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public int getPid() {

return pid;

}

public void setPid(int pid) {

this.pid = pid;

}

public int getGroupId() {

return groupId;

}

public void setGroupId(int groupId) {

this.groupId = groupId;

}

public String getGroupName() {

return groupName;

}

public void setGroupName(String groupName) {

this.groupName = groupName;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getLogo() {

return logo;

}

public void setLogo(String logo) {

this.logo = logo;

}

public int getIsactive() {

return isactive;

}

public void setIsactive(int isactive) {

this.isactive = isactive;

}

public String getIsactiveName() {

return isactiveName;

}

public void setIsactiveName(String isactiveName) {

this.isactiveName = isactiveName;

}

}

package com.sdocean.dictionary.action;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.PublicModel;

import com.sdocean.dictionary.service.PublicService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

@Controller

public class PublicAction {

private static Logger log = Logger.getLogger(PublicAction.class);

@Autowired

private PublicService publicService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_public.do")

public ModelAndView info\_public(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/dictionary/publicInfo");

return mav;

}

/\*

\* 通过parentCode查询到公共代码列表

\*/

@RequestMapping(value="getPublicList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getPublicList(@ModelAttribute("model") PublicModel model,HttpServletRequest request,

HttpServletResponse response){

List<PublicModel> publics = publicService.getPublicsByParent(model.getParentCode());

return JsonUtil.toJson(publics);

}

/\*

\* 为公共代码管理查询结果

\*/

@RequestMapping(value="showPublics.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showPublics(@ModelAttribute("model") PublicModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = publicService.getCols4List();

result.setCols(cols);

List<PublicModel> rows = publicService.getPublics(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改的公共代码

\*/

@RequestMapping(value="savePublicChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String savePublicChange(@ModelAttribute("model") PublicModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = publicService.savePublicChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增的公共代码

\*/

@RequestMapping(value="saveNewPublic.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewPublic(@ModelAttribute("model") PublicModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = publicService.saveNewPublic(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 删除选中的编码

\*/

@RequestMapping(value="delePublics.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String delePublics(@ModelAttribute("model") PublicModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = publicService.delePublic(model.getIds());

logService.saveOperationLog(result,request);

return result.getMessage();

}

}

package com.sdocean.dictionary.action;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.UnitModel;

import com.sdocean.dictionary.service.UnitService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

@Controller

public class UnitAction {

@Resource

UnitService unitService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_unit.do")

public ModelAndView info\_unit(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/dictionary/unitInfo");

return mav;

}

@RequestMapping(value="showUnitList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showUnitList(@ModelAttribute("model") UnitModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = unitService.getCols4GroupList();

result.setCols(cols);

//为查询结果增加列表

List<UnitModel> rows = unitService.getUnitList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 获得单位列表

\*/

@RequestMapping(value="getUnitList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getUnitList(@ModelAttribute("model") UnitModel model,HttpServletRequest request,

HttpServletResponse response){

List<UnitModel> rows = unitService.getUnitList(model);

return JsonUtil.toJson(rows);

}

/\*

\* 保存修改信息

\*/

@RequestMapping(value="saveUnitChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveUnitChange(@ModelAttribute("model") UnitModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = unitService.saveUnitChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增信息

\*/

@RequestMapping(value="saveNewUnit.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewUnit(@ModelAttribute("model") UnitModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = unitService.saveNewUnit(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

}

package com.sdocean.dictionary.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.UnitGroupModel;

import com.sdocean.dictionary.service.UnitGroupService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

@Controller

public class UnitGroupAction {

@Resource

UnitGroupService unitGroupService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_unit\_group.do")

public ModelAndView info\_unit\_group(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/dictionary/unitGroup");

return mav;

}

@RequestMapping(value="showGroupList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showGroupList(@ModelAttribute("model") UnitGroupModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = unitGroupService.getCols4GroupList();

result.setCols(cols);

//为查询结果增加列表

List<UnitGroupModel> rows = unitGroupService.getUnitGroups(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改信息

\*/

@RequestMapping(value="saveGroupChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveGroupChange(@ModelAttribute("model") UnitGroupModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = unitGroupService.saveGroupChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增信息

\*/

@RequestMapping(value="saveNewGroup.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewGroup(@ModelAttribute("model") UnitGroupModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = unitGroupService.saveNewGroup(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 获得有效的单位组的集合

\*/

@RequestMapping(value="getUnitGroups.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getUnitGroups(HttpServletRequest request,

HttpServletResponse response){

List<UnitGroupModel> list = new ArrayList<UnitGroupModel>();

list = unitGroupService.getUnitGroups();

return JsonUtil.toJson(list);

}

}

package com.sdocean.dictionary.action;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.dictionary.model.WaterQualityStandardModel;

import com.sdocean.dictionary.service.WaterQualityStandardService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

@Controller

public class WaterQualityStandardAction {

@Resource

WaterQualityStandardService standardService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_water\_level.do")

public ModelAndView info\_water\_level(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/dictionary/waterLevelInfo");

return mav;

}

@RequestMapping(value="getStandardList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStandardList(@ModelAttribute("model") WaterQualityStandardModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = standardService.getCols4GroupList();

result.setCols(cols);

//为查询结果增加列表

List<WaterQualityStandardModel> rows = standardService.getStandardList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改信息

\*/

@RequestMapping(value="saveQualityChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveQualityChange(@ModelAttribute("model") WaterQualityStandardModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = standardService.saveQualityChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增信息

\*/

@RequestMapping(value="saveNewQuality.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewQuality(@ModelAttribute("model") WaterQualityStandardModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = standardService.saveNewQuality(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 删除选中的水质等级

\*/

@RequestMapping(value="deleWaterQulity.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String deleWaterQulity(@ModelAttribute("model") WaterQualityStandardModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = standardService.deleWaterQulity(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

}

package com.sdocean.device.service;

import java.util.ArrayList;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.SelectTree;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.device.dao.DeviceDao;

import com.sdocean.device.model.DeviceGroup;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class DeviceService {

@Autowired

private DeviceDao deviceDao;

/\*

\* 为公共代码管理提供表头

\*/

public List<UiColumn> getCols4List(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", true, "\*");

UiColumn col2 = new UiColumn("设备名称", "name", true, "\*");

UiColumn col3 = new UiColumn("设备编码", "code", true, "\*");

UiColumn col4 = new UiColumn("设备模型", "deviceModel", true, "\*");

UiColumn col5 = new UiColumn("介绍", "brief", true, "\*");

UiColumn col6 = new UiColumn("明细", "detail", true, "\*");

UiColumn col19 = new UiColumn("orderCode", "orderCode", true, "\*");

UiColumn col10 = new UiColumn("保留位数", "pointNum", true, "\*");

UiColumn col12 = new UiColumn("维护周期(天)", "mainnum", true, "\*");

UiColumn col11 = new UiColumn("监测参数", "indicatorNames", true, "\*");

UiColumn col13 = new UiColumn("groupId", "groupId", false, "\*");

UiColumn col14 = new UiColumn("设备分类", "groupName", true, "\*");

UiColumn col15 = new UiColumn("typeId", "typeId", false, "\*");

UiColumn col16 = new UiColumn("设备类型", "typeName", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col4);

cols.add(col5);

cols.add(col19);

cols.add(col10);

cols.add(col12);

cols.add(col11);

cols.add(col13);

cols.add(col14);

cols.add(col15);

cols.add(col16);

return cols;

}

/\*

\* 查询符合条件的设备列表

\*/

public List<DeviceModel> getDevices(DeviceModel model){

return deviceDao.getDevices(model);

}

/\*

\* 保存修改的设备信息

\*/

public Result saveDeviceChange(DeviceModel model){

return deviceDao.saveDeviceChange(model);

}

/\*

\* 保存新增的设备

\*/

public Result saveNewDevice(DeviceModel model){

return deviceDao.saveNewDevice(model);

}

/\*

\* 获得当前站点所关联的设备列表

\*/

public List<SelectTree> getDeviceListByStation(StationModel model){

return deviceDao.getDeviceListByStation(model);

}

/\*

\* 获得当前站点下的设备的列表

\*/

public List<DeviceModel> getDevices4Station(StationModel model){

return deviceDao.getDevices4Station(model);

}

/\*

\* 获得当前站点下有展示权限的设备的列表

\*/

public List<DeviceModel> getDevices4Station4Show(StationModel model){

return deviceDao.getDevices4Station4Show(model);

}

/\*

\* 为站点组列表提供表头

\*/

public List<UiColumn> getCols4GroupList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", true, "\*");

UiColumn col2 = new UiColumn("分类名称", "title", true, "\*");

UiColumn col3 = new UiColumn("分类编码", "code", true, "\*");

UiColumn col6 = new UiColumn("明细", "detail", true, "\*");

UiColumn col19 = new UiColumn("排序码", "ordercode", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col19);

return cols;

}

/\*

\* 查询所有符合条件的设备组

\*/

public List<DeviceGroup> getDeviceGroupList(DeviceGroup model){

return deviceDao.getDeviceGroupList(model);

}

/\*

\* 新增设备组

\*/

public Result addDeviceGroup(DeviceGroup model){

return deviceDao.addDeviceGroup(model);

}

/\*

\* 修改设备组

\*/

public Result saveDeviceGroup(DeviceGroup model){

return deviceDao.saveDeviceGroup(model);

}

}

package com.sdocean.device.model;

public class DeviceGroup {

private int id;

private String code;

private String title;

private String detail;

private String ordercode;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getDetail() {

return detail;

}

public void setDetail(String detail) {

this.detail = detail;

}

public String getOrdercode() {

return ordercode;

}

public void setOrdercode(String ordercode) {

this.ordercode = ordercode;

}

}

package com.sdocean.device.model;

import java.util.List;

import com.sdocean.indicator.model.IndicatorModel;

public class DeviceModel {

private int id;

private String name;

private String code;

private String deviceModel;

private String brief;

private String detail;

private String orderCode;

private String indicatorIds;

private String indicatorNames;

private List<IndicatorModel> indicators;

private Integer interval;

private Integer pointNum; //小数点保留位数

private int mainnum; //例行维护周期

private int groupId;

private String groupName; //设备分组 device\_group表

private int typeId;

private String typeName; //设备类型 1是传感器 2是其他

public int getGroupId() {

return groupId;

}

public void setGroupId(int groupId) {

this.groupId = groupId;

}

public String getGroupName() {

return groupName;

}

public void setGroupName(String groupName) {

this.groupName = groupName;

}

public int getTypeId() {

return typeId;

}

public void setTypeId(int typeId) {

this.typeId = typeId;

}

public String getTypeName() {

return typeName;

}

public void setTypeName(String typeName) {

this.typeName = typeName;

}

public int getMainnum() {

return mainnum;

}

public void setMainnum(int mainnum) {

this.mainnum = mainnum;

}

public Integer getPointNum() {

return pointNum;

}

public void setPointNum(Integer pointNum) {

this.pointNum = pointNum;

}

public Integer getInterval() {

return interval;

}

public void setInterval(Integer interval) {

this.interval = interval;

}

public List<IndicatorModel> getIndicators() {

return indicators;

}

public void setIndicators(List<IndicatorModel> indicators) {

this.indicators = indicators;

}

public String getIndicatorIds() {

return indicatorIds;

}

public void setIndicatorIds(String indicatorIds) {

this.indicatorIds = indicatorIds;

}

public String getIndicatorNames() {

return indicatorNames;

}

public void setIndicatorNames(String indicatorNames) {

this.indicatorNames = indicatorNames;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getDeviceModel() {

return deviceModel;

}

public void setDeviceModel(String deviceModel) {

this.deviceModel = deviceModel;

}

public String getBrief() {

return brief;

}

public void setBrief(String brief) {

this.brief = brief;

}

public String getDetail() {

return detail;

}

public void setDetail(String detail) {

this.detail = detail;

}

public String getOrderCode() {

return orderCode;

}

public void setOrderCode(String orderCode) {

this.orderCode = orderCode;

}

}

package com.sdocean.device.dao;

import java.util.ArrayList;

import java.util.List;

import org.springframework.stereotype.Component;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.SelectTree;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.device.model.DeviceGroup;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.dictionary.model.PublicModel;

import com.sdocean.frame.dao.OracleEngine;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.indicator.model.IndicatorModel;

import com.sdocean.log.model.SysLoginLogModel;

import com.sdocean.menu.model.CurrMenu;

import com.sdocean.menu.model.SysMenu;

import com.sdocean.role.model.RoleModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.users.model.SysUser;

@Component

public class DeviceDao extends OracleEngine{

/\*

\* 查询所有符合条件的设备组

\*/

public List<DeviceGroup> getDeviceGroupList(DeviceGroup model){

List<DeviceGroup> list = new ArrayList<DeviceGroup>();

StringBuffer sql = new StringBuffer("");

sql.append(" select id,code,title,detail,ordercode");

sql.append(" from device\_group ");

sql.append(" where 1=1");

//添加查询条件

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

sql.append(" and (code like '%").append(model.getCode()).append("%' or title like '%").append(model.getCode()).append("%')");

}

//添加排序条件

sql.append(" order by ordercode");

list = this.queryObjectList(sql.toString(), DeviceGroup.class);

return list;

}

/\*

\* 新增设备组

\*/

public Result addDeviceGroup(DeviceGroup model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("新增成功");

StringBuffer sql = new StringBuffer("");

sql.append(" insert into device\_group(code,title,detail,ordercode)");

sql.append(" values(?,?,?,?)");

Object[] params = new Object[]{

model.getCode(),model.getTitle(),model.getDetail(),model.getOrdercode()

};

try {

this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("新增失败，请重试");

}

return result;

}

/\*

\* 修改设备组

\*/

public Result saveDeviceGroup(DeviceGroup model){

//初始化返回结果

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("修改成功");

StringBuffer sql = new StringBuffer("");

sql.append(" update device\_group set code=?,title=?,detail=?,ordercode=?");

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

Object[] params = new Object[]{

model.getCode(),model.getTitle(),model.getDetail(),model.getOrdercode(),model.getId()

};

try {

this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("修改失败，请重试");

}

return result;

}

/\*

\* 查询符合条件的设备列表

\*/

public List<DeviceModel> getDevices(DeviceModel model){

List<DeviceModel> list = new ArrayList<DeviceModel>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.name,a.code,a.deviceModel,a.brief,a.detail,a.ordercode,");

sql.append(" group\_concat(m.title SEPARATOR '|') as indicatorNames,");

sql.append(" group\_concat(m.indicatorid) as indicatorIds,pointnum,mainnum,");

sql.append(" a.groupid,g.title as groupname,a.typeid,case a.typeid when 1 then '传感器' else '其他' end as typename");

sql.append(" from device\_catalog a left join (select b.catalogId,b.indicatorid,c.title");

sql.append(" from device\_catalog\_indicator b,dm\_indicator c");

sql.append(" where b.indicatorid = c.id and c.isactive = 1) m");

sql.append(" on a.id = m.catalogid ,device\_group g");

sql.append(" where a.groupid = g.id");

//添加查询条件

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

sql.append(" and ( a.name like '%").append(model.getCode()).append("%' or");

sql.append(" a.code like '%").append(model.getCode()).append("%') ");

}

if(model!=null&&model.getGroupId()>0){

sql.append(" and a.groupid = ").append(model.getGroupId());

}

sql.append(" group by a.id");

//添加排序

sql.append(" order by ordercode ");

list = this.queryObjectList(sql.toString(), DeviceModel.class);

return list;

}

/\*

\* 根据设备ID,查询设备信息

\*/

public DeviceModel getDeviceByDeviceId(int deviceId){

DeviceModel device = new DeviceModel();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.name,a.code,a.deviceModel,a.brief,a.detail,a.ordercode,");

sql.append(" group\_concat(m.title SEPARATOR '|') as indicatorNames,");

sql.append(" group\_concat(m.indicatorid) as indicatorIds,pointnum,mainnum");

sql.append(" from device\_catalog a left join (select b.catalogId,b.indicatorid,c.title");

sql.append(" from device\_catalog\_indicator b,dm\_indicator c");

sql.append(" where b.indicatorid = c.id and c.isactive = 1) m");

sql.append(" on a.id = m.catalogid");

sql.append(" where a.id =").append(deviceId);

sql.append(" limit 1");

device = this.queryObject(sql.toString(), DeviceModel.class);

return device;

}

/\*

\* 保存修改的设备信息

\*/

public Result saveDeviceChange(DeviceModel model){

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("修改成功");

//验证code 是否唯一

StringBuffer checkSql = new StringBuffer("");

checkSql.append(" select count(1) from device\_catalog where code ='").append(model.getCode()).append("' and id <>").append(model.getId());

int cou = 0;

try {

cou = this.queryForInt(checkSql.toString(), null);

} catch (Exception e) {

e.printStackTrace();

result.setResult(result.FAILED);

result.setMessage("验证唯一性时失败");

return result;

}

if(cou>0){

result.setResult(result.FAILED);

result.setMessage("CODE重复,违反唯一性原则");

return result;

}

//开始保存数据

StringBuffer sql = new StringBuffer("");

sql.append(" update device\_catalog set name=?,code=?,devicemodel=?,brief=?,detail=?,ordercode=?,pointNum=?,mainnum=?,groupid=?,typeid=? where id =?");

Object[] params = new Object[]{

model.getName(),model.getCode(),model.getDeviceModel(),

model.getBrief(),model.getDetail(),model.getOrderCode(),model.getPointNum(),model.getMainnum(),model.getGroupId(),model.getTypeId(),model.getId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("保存失败");

return result;

}

this.saveDeviceIndicator(model);

return result;

}

/\*

\* 保存新增的设备

\*/

public Result saveNewDevice(DeviceModel model){

Result result = new Result();

result.setDotype(result.UPDATE);

result.setModel(JsonUtil.toJson(model));

result.setResult(result.SUCCESS);

result.setMessage("新增成功");

//验证code 是否唯一

StringBuffer checkSql = new StringBuffer("");

checkSql.append(" select count(1) from device\_catalog where code ='").append(model.getCode()).append("' ");

int cou = 0;

try {

cou = this.queryForInt(checkSql.toString(), null);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("验证唯一性时失败");

return result;

}

if(cou>0){

result.setResult(result.FAILED);

result.setMessage("CODE重复,违反唯一性原则");

return result;

}

//开始保存数据

StringBuffer sql = new StringBuffer("");

sql.append(" insert into device\_catalog(name,code,devicemodel,brief,detail,ordercode,pointnum,mainnum,groupid,typeid) values(?,?,?,?,?,?,?,?,?,?)");

Object[] params = new Object[]{

model.getName(),model.getCode(),model.getDeviceModel(),

model.getBrief(),model.getDetail(),model.getOrderCode(),model.getPointNum(),model.getMainnum(),model.getGroupId(),model.getTypeId()

};

int res = 0;

try {

res = this.update(sql.toString(), params);

} catch (Exception e) {

result.setResult(result.FAILED);

result.setMessage("新增失败");

return result;

}

//获得保存的ID值\

String last = "SELECT LAST\_INSERT\_ID()";

int lastId = this.queryForInt(last, null);

model.setId(lastId);

this.saveDeviceIndicator(model);

return result;

}

public void saveDeviceIndicator(DeviceModel model){

//删除设备的参数

StringBuffer deleSql = new StringBuffer("");

deleSql.append(" delete from device\_catalog\_indicator where catalogid = ").append(model.getId());

this.update(deleSql.toString(), null);

String ids = model.getIndicatorIds();

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

StringBuffer val = new StringBuffer("(0,0)");

for(String id:idStrings){

val.append(",(").append(model.getId()).append(",").append(id).append(")");

}

StringBuffer insertSql = new StringBuffer("");

insertSql.append(" insert into device\_catalog\_indicator(catalogid,indicatorid) values").append(val);

insertSql.append(" ON DUPLICATE KEY UPDATE indicatorid = indicatorid ");

this.update(insertSql.toString(), null);

}

/\*

\* 获得当前站点所关联的设备列表

\*/

public List<SelectTree> getDeviceListByStation(StationModel model){

List<SelectTree> list = new ArrayList<SelectTree>();

//添加设备列表的首层

SelectTree first = new SelectTree();

first.setId("0");

first.setName("设备");

first.setIsExpanded(true);

first.setIsActive(true);

//获得所有有效的设备的列表

List<SelectTree> children = new ArrayList<SelectTree>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.name, case when b.aiot\_watch\_point\_id is null then 'false' else 'true' end as selected");

sql.append(" from device\_catalog a");

sql.append(" left join map\_awp\_device\_catalog b on a.id = b.device\_catalog\_id");

sql.append(" and b.aiot\_watch\_point\_id = ").append(model.getId());

children = this.queryObjectList(sql.toString(), SelectTree.class);

//将设备列表添加的首层的child中

first.setChildren(children);

//将首层添加到结果集中

list.add(first);

return list;

}

/\*

\* 获得当前站点下的设备的列表

\*/

public List<DeviceModel> getDevices4Station(StationModel model){

List<DeviceModel> list = new ArrayList<DeviceModel>();

StringBuffer sql = new StringBuffer("");

sql.append(" select distinct a.id,a.name,a.code,a.deviceModel,a.brief,a.detail,a.ordercode");

sql.append(" from device\_catalog a,map\_awp\_device\_catalog b");

sql.append(" where a.id = b.device\_catalog\_id");

//添加查询条件

if(model!=null&&model.getId()>0){

sql.append(" and b.aiot\_watch\_point\_id =").append(model.getId());

}

sql.append(" order by a.ordercode");

list = this.queryObjectList(sql.toString(), DeviceModel.class);

return list;

}

/\*

\* 获得当前站点下有展示权限的设备的列表

\*/

public List<DeviceModel> getDevices4Station4Show(StationModel model){

List<DeviceModel> list = new ArrayList<DeviceModel>();

StringBuffer sql = new StringBuffer("");

sql.append(" select a.id,a.name,a.code,a.deviceModel,a.brief,a.detail,a.ordercode");

sql.append(" from view\_show\_station\_device a");

sql.append(" where a.stationid =").append(model.getId());

sql.append(" order by a.ordercode");

list = this.queryObjectList(sql.toString(), DeviceModel.class);

return list;

}

/\*

\*根据deviceid以及indicators

\*/

public DeviceModel getDeviceByid(String deviceId,String indicatorids){

DeviceModel model = new DeviceModel();

//查询出设备

StringBuffer devicesql = new StringBuffer("");

devicesql.append(" select a.id,a.name,a.code,a.deviceModel,a.brief,a.detail,a.ordercode");

devicesql.append(" from device\_catalog a where a.id = ").append(deviceId);

model = this.queryObject(devicesql.toString(), DeviceModel.class);

//查询出参数列表

StringBuffer indicatorSql = new StringBuffer("");

indicatorSql.append(" select a.id,a.code,a.title,a.groupId,a.unitId,a.description,a.isactive");

indicatorSql.append(" from dm\_indicator a where a.id in (").append(indicatorids).append(")");

List<IndicatorModel> indicators = new ArrayList<IndicatorModel>();

indicators = this.queryObjectList(indicatorSql.toString(), IndicatorModel.class);

if(indicators!=null&&indicators.size()>0){

model.setIndicators(indicators);

}

return model;

}

/\*

\* 读取站点内设备读取数据的时间间隔

\*/

public DeviceModel getInterval4device(int stationid,DeviceModel device){

StringBuffer sql = new StringBuffer("");

sql.append("select max(betweentime) from aiot\_station\_device\_comm where stationid in (0, ").append(stationid).append(") and deviceid in (0").append(device.getId()).append(")");

Integer inter = this.queryForInt(sql.toString(), null);

if(inter==null){

inter = 0;

}

device.setInterval(inter);

return device;

}

}

package com.sdocean.device.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.SelectTree;

import com.sdocean.device.model.DeviceGroup;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.device.service.DeviceService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

@Controller

public class DeviceAction {

private static Logger log = Logger.getLogger(DeviceAction.class);

@Autowired

private DeviceService deviceService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

/\*

\* 跳转到设备组页面

\*/

@RequestMapping("info\_device\_group.do")

public ModelAndView info\_device\_group(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/device/deviceGroupInfo");

return mav;

}

/\*

\* 查询设备分组列表

\*/

@RequestMapping(value="showDeviceGroupList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showDeviceGroupList(@ModelAttribute("model") DeviceGroup model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = deviceService.getCols4GroupList();

result.setCols(cols);

List<DeviceGroup> rows = deviceService.getDeviceGroupList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 获得设备分组列表

\*/

@RequestMapping(value="getDeviceGroupList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDeviceGroupList(@ModelAttribute("model") DeviceGroup model,HttpServletRequest request,

HttpServletResponse response){

List<DeviceGroup> rows = deviceService.getDeviceGroupList(model);

return JsonUtil.toJson(rows);

}

/\*

\* 保存新增的设备分组

\*/

@RequestMapping(value="addDeviceGroup.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String addDeviceGroup(@ModelAttribute("model") DeviceGroup model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = deviceService.addDeviceGroup(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存修改的设备分组

\*/

@RequestMapping(value="saveDeviceGroup.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveDeviceGroup(@ModelAttribute("model") DeviceGroup model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = deviceService.saveDeviceGroup(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

@RequestMapping("info\_device.do")

public ModelAndView info\_public(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/device/deviceInfo");

return mav;

}

/\*

\* 跳转到设备监控页面

\*/

@RequestMapping("info\_deviceWatch.do")

public ModelAndView info\_deviceWatch(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getPageVision()+"/device/deviceWatch");

return mav;

}

/\*

\* 为设备管理查询结果

\*/

@RequestMapping(value="showDevices.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showDevices(@ModelAttribute("model") DeviceModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = deviceService.getCols4List();

result.setCols(cols);

List<DeviceModel> rows = deviceService.getDevices(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改的公共代码

\*/

@RequestMapping(value="saveDeviceChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveDeviceChange(@ModelAttribute("model") DeviceModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = deviceService.saveDeviceChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增的公共代码

\*/

@RequestMapping(value="saveNewDevice.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewDevice(@ModelAttribute("model") DeviceModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = deviceService.saveNewDevice(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增的公共代码

\*/

@RequestMapping(value="getDeviceListByStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDeviceListByStation(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

List<SelectTree> list = new ArrayList<SelectTree>();

list = deviceService.getDeviceListByStation(model);

return JsonUtil.toJson(list);

}

/\*

\* 获得当前站点下的所有的设备的列表

\*/

@RequestMapping(value="getDevices4Station.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDevices4Station(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

List<DeviceModel> list = new ArrayList<DeviceModel>();

list = deviceService.getDevices4Station(model);

return JsonUtil.toJson(list);

}

/\*

\* 获得当前站点下有展示权限的设备的列表

\*/

@RequestMapping(value="getDevices4Station4Show.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDevices4Station4Show(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

List<DeviceModel> list = new ArrayList<DeviceModel>();

list = deviceService.getDevices4Station4Show(model);

return JsonUtil.toJson(list);

}

/\*

\* 查询所有的设备列表

\*/

@RequestMapping(value="getDevices.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDevices(HttpServletRequest request,

HttpServletResponse response){

DeviceModel model = new DeviceModel();

List<DeviceModel> rows = deviceService.getDevices(model);

return JsonUtil.toJson(rows);

}

}

package com.sdocean.dataQuery.service;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.model.AiotModel;

import com.sdocean.dataQuery.model.DataChangeModel;

import com.sdocean.dataQuery.model.DataImportModel;

import com.sdocean.dataQuery.model.DataQueryDataModel;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.firstpage.model.MetaFirstPage;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.metadata.dao.MetadataDao;

import com.sdocean.metadata.dao.MetadataTableDao;

import com.sdocean.metadata.model.MetadataModel;

import com.sdocean.metadata.model.MetadataTable;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class DataQueryService {

@Autowired

private DataQueryDao dataQueryDao;

@Autowired

private MetadataDao metaDataDao;

@Autowired

private MetadataTableDao tableDao;

/\*

\* 为实时数据添加表头

\*/

public List<UiColumn> getCols4DataQuery(DataQueryModel model){

return dataQueryDao.getCols4DataQuery(model);

}

/\*

\* 为实时数据查询提供结果

\*/

public List<Map<String, Object>> getRows4DataQuery(DataQueryModel model){

return dataQueryDao.getRows4DataQuery(model);

}

/\*

\* 查询出当前站点信息的最后实时数据

\* 为首页展示

\*/

public List<MetaFirstPage> getData4FirstPage(StationModel station){

//dataQueryDao.queryData();

return dataQueryDao.getData4FirstPage(station);

}

/\*

\* 为数据修改查询结果提供表头

\*/

public List<UiColumn> getCols4DataChange(DataChangeModel model){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*", false, "number");

UiColumn col2 = new UiColumn("时间", "collect\_time", true, "\*", true, "");

//col2.setCellFilter("data:\"yyyy-MM-dd HH:mm:ss.0\"");

UiColumn col3 = new UiColumn("数值", "data", true, "\*", true, "number");

cols.add(col1);

cols.add(col2);

cols.add(col3);

return cols;

}

/\*public static void main(String[] args) {

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*", false, "number");

UiColumn col2 = new UiColumn("时间", "collect\_time", true, "\*", true, "");

//col2.setCellFilter("data:\"yyyy-MM-dd HH:mm:ss.0\"");

UiColumn col3 = new UiColumn("数值", "data", true, "\*", true, "number");

cols.add(col1);

cols.add(col2);

cols.add(col3);

}\*/

/\*

\* 为数据修改的查询提供结果

\*/

public List<MetadataModel> getResult4DataChangeshow(DataChangeModel model){

return dataQueryDao.getResult4DataChangeshow(model);

}

/\*

\* 保存修改

\*/

public Result saveChangeData(DataChangeModel model){

return dataQueryDao.saveChangeData(model);

}

/\*

\* 保存数据导入

\*/

public Result saveImportData(DataImportModel model){

Result result = new Result();

result.setDotype(Result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.FAILED);

result.setMessage("导入数据成功");

List<DataImportModel> list = new ArrayList<DataImportModel>();

try {

list = (List<DataImportModel>) JsonUtil.fromJsons(model.getImportString(), DataImportModel.class);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("数据错误");

return result;

}

if(list==null||list.size()<1){

result.setResult(Result.FAILED);

result.setMessage("数据错误");

return result;

}

//将其他形式的元数据转换成标准元数据

List<MetadataModel> metas = new ArrayList<MetadataModel>();

metas = metaDataDao.changeMetadata(list, 1, "csvImport");

//整合数据采集时间

for(MetadataModel meta:metas){

meta = metaDataDao.changeNewMetadata(meta);

StationModel station = new StationModel();

station.setId(meta.getWpId());

//将标准元数据存入到基础元数据表中

MetadataTable metaTable = tableDao.getOneTable(station, meta.getCollect\_time(), 1);

try {

metaDataDao.saveMetaData(metaTable.getTableName(), meta);

} catch (Exception e) {

// TODO: handle exception

result.setResult(Result.FAILED);

return result;

}

//将标准元数据存入到综合元数据表中

MetadataTable synTable = tableDao.getOneTable(station, meta.getCollect\_time(), 2);

try {

metaDataDao.saveSynData(synTable.getTableName(), meta);

} catch (Exception e) {

// TODO: handle exception

result.setResult(Result.FAILED);

result.setMessage("导入综合元数据表错误");

return result;

}

}

return result;

}

/\*

\* 获得某个设备的表头

\*/

public String getCols4DataQuery(int deviceid){

return dataQueryDao.getCols4DataQuery(deviceid);

}

/\*

\* 获得某个设备的内容

\*/

public List<Map<String, Object>> getRows4DataQuery(int deviceid){

return dataQueryDao.getRows4DataQuery(deviceid);

}

/\*

\* 获得某个设备的图片合集

\*/

public List<DataQueryDataModel> getImages4DataQuery(int deviceid){

return dataQueryDao.getImages4DataQuery(deviceid);

}

public List<AiotModel> getAiotDatas(int deviceid){

return dataQueryDao.getAiotDatas(deviceid);

}

}

package com.sdocean.dataQuery.service;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.model.AiotModel;

import com.sdocean.dataQuery.model.DataChangeModel;

import com.sdocean.dataQuery.model.DataImportModel;

import com.sdocean.dataQuery.model.DataQueryDataModel;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.firstpage.model.MetaFirstPage;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.metadata.dao.MetadataDao;

import com.sdocean.metadata.dao.MetadataTableDao;

import com.sdocean.metadata.model.MetadataModel;

import com.sdocean.metadata.model.MetadataTable;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class DataQueryService {

@Autowired

private DataQueryDao dataQueryDao;

@Autowired

private MetadataDao metaDataDao;

@Autowired

private MetadataTableDao tableDao;

/\*

\* 为实时数据添加表头

\*/

public List<UiColumn> getCols4DataQuery(DataQueryModel model){

return dataQueryDao.getCols4DataQuery(model);

}

/\*

\* 为实时数据查询提供结果

\*/

public List<Map<String, Object>> getRows4DataQuery(DataQueryModel model){

return dataQueryDao.getRows4DataQuery(model);

}

/\*

\* 查询出当前站点信息的最后实时数据

\* 为首页展示

\*/

public List<MetaFirstPage> getData4FirstPage(StationModel station){

//dataQueryDao.queryData();

return dataQueryDao.getData4FirstPage(station);

}

/\*

\* 为数据修改查询结果提供表头

\*/

public List<UiColumn> getCols4DataChange(DataChangeModel model){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*", false, "number");

UiColumn col2 = new UiColumn("时间", "collect\_time", true, "\*", true, "");

//col2.setCellFilter("data:\"yyyy-MM-dd HH:mm:ss.0\"");

UiColumn col3 = new UiColumn("数值", "data", true, "\*", true, "number");

cols.add(col1);

cols.add(col2);

cols.add(col3);

return cols;

}

/\*public static void main(String[] args) {

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*", false, "number");

UiColumn col2 = new UiColumn("时间", "collect\_time", true, "\*", true, "");

//col2.setCellFilter("data:\"yyyy-MM-dd HH:mm:ss.0\"");

UiColumn col3 = new UiColumn("数值", "data", true, "\*", true, "number");

cols.add(col1);

cols.add(col2);

cols.add(col3);

}\*/

/\*

\* 为数据修改的查询提供结果

\*/

public List<MetadataModel> getResult4DataChangeshow(DataChangeModel model){

return dataQueryDao.getResult4DataChangeshow(model);

}

/\*

\* 保存修改

\*/

public Result saveChangeData(DataChangeModel model){

return dataQueryDao.saveChangeData(model);

}

/\*

\* 保存数据导入

\*/

public Result saveImportData(DataImportModel model){

Result result = new Result();

result.setDotype(Result.ADD);

result.setModel(JsonUtil.toJson(model));

result.setResult(Result.FAILED);

result.setMessage("导入数据成功");

List<DataImportModel> list = new ArrayList<DataImportModel>();

try {

list = (List<DataImportModel>) JsonUtil.fromJsons(model.getImportString(), DataImportModel.class);

} catch (Exception e) {

result.setResult(Result.FAILED);

result.setMessage("数据错误");

return result;

}

if(list==null||list.size()<1){

result.setResult(Result.FAILED);

result.setMessage("数据错误");

return result;

}

//将其他形式的元数据转换成标准元数据

List<MetadataModel> metas = new ArrayList<MetadataModel>();

metas = metaDataDao.changeMetadata(list, 1, "csvImport");

//整合数据采集时间

for(MetadataModel meta:metas){

meta = metaDataDao.changeNewMetadata(meta);

StationModel station = new StationModel();

station.setId(meta.getWpId());

//将标准元数据存入到基础元数据表中

MetadataTable metaTable = tableDao.getOneTable(station, meta.getCollect\_time(), 1);

try {

metaDataDao.saveMetaData(metaTable.getTableName(), meta);

} catch (Exception e) {

// TODO: handle exception

result.setResult(Result.FAILED);

return result;

}

//将标准元数据存入到综合元数据表中

MetadataTable synTable = tableDao.getOneTable(station, meta.getCollect\_time(), 2);

try {

metaDataDao.saveSynData(synTable.getTableName(), meta);

} catch (Exception e) {

// TODO: handle exception

result.setResult(Result.FAILED);

result.setMessage("导入综合元数据表错误");

return result;

}

}

return result;

}

/\*

\* 获得某个设备的表头

\*/

public String getCols4DataQuery(int deviceid){

return dataQueryDao.getCols4DataQuery(deviceid);

}

/\*

\* 获得某个设备的内容

\*/

public List<Map<String, Object>> getRows4DataQuery(int deviceid){

return dataQueryDao.getRows4DataQuery(deviceid);

}

/\*

\* 获得某个设备的图片合集

\*/

public List<DataQueryDataModel> getImages4DataQuery(int deviceid){

return dataQueryDao.getImages4DataQuery(deviceid);

}

public List<AiotModel> getAiotDatas(int deviceid){

return dataQueryDao.getAiotDatas(deviceid);

}

}

package com.sdocean.dataQuery.service;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.dao.MouldDao;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.dataQuery.model.MouldModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.page.model.NgColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class MouldService {

@Autowired

private MouldDao mouldDao;

/\*

\* 获得当前站点的水质评价模板

\*/

public MouldModel getMouldByStationId(MouldModel model){

return mouldDao.getMouldByStationId(model);

}

/\*

\* 保存当前站点的水质评价模板

\*/

public Result saveMouldSetting(MouldModel model){

return mouldDao.saveMouldSetting(model);

}

/\*

\* 根据站点,获得水质评价模板

\*/

public MouldModel getMouldByStation(StationModel model){

return mouldDao.getMouldByStation(model);

}

}

package com.sdocean.dataQuery.service;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.dao.MouldDao;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.dataQuery.model.MouldModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.page.model.NgColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class MouldService {

@Autowired

private MouldDao mouldDao;

/\*

\* 获得当前站点的水质评价模板

\*/

public MouldModel getMouldByStationId(MouldModel model){

return mouldDao.getMouldByStationId(model);

}

/\*

\* 保存当前站点的水质评价模板

\*/

public Result saveMouldSetting(MouldModel model){

return mouldDao.saveMouldSetting(model);

}

/\*

\* 根据站点,获得水质评价模板

\*/

public MouldModel getMouldByStation(StationModel model){

return mouldDao.getMouldByStation(model);

}

}

package com.sdocean.dataQuery.service;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.dao.MouldDao;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.dataQuery.model.MouldModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.page.model.NgColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class MouldService {

@Autowired

private MouldDao mouldDao;

/\*

\* 获得当前站点的水质评价模板

\*/

public MouldModel getMouldByStationId(MouldModel model){

return mouldDao.getMouldByStationId(model);

}

/\*

\* 保存当前站点的水质评价模板

\*/

public Result saveMouldSetting(MouldModel model){

return mouldDao.saveMouldSetting(model);

}

/\*

\* 根据站点,获得水质评价模板

\*/

public MouldModel getMouldByStation(StationModel model){

return mouldDao.getMouldByStation(model);

}

}

package com.sdocean.dataQuery.service;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.dataQuery.dao.DataQueryDao;

import com.sdocean.dataQuery.dao.MouldDao;

import com.sdocean.dataQuery.model.DataQueryModel;

import com.sdocean.dataQuery.model.MouldModel;

import com.sdocean.firstpage.model.LastMetaData;

import com.sdocean.page.model.NgColumn;

import com.sdocean.station.model.StationModel;

@Service

@Transactional(rollbackFor=Exception.class, propagation=Propagation.REQUIRED)

public class MouldService {

@Autowired

private MouldDao mouldDao;

/\*

\* 获得当前站点的水质评价模板

\*/

public MouldModel getMouldByStationId(MouldModel model){

return mouldDao.getMouldByStationId(model);

}

/\*

\* 保存当前站点的水质评价模板

\*/

public Result saveMouldSetting(MouldModel model){

return mouldDao.saveMouldSetting(model);

}

/\*

\* 根据站点,获得水质评价模板

\*/

public MouldModel getMouldByStation(StationModel model){

return mouldDao.getMouldByStation(model);

}

}

package com.sdocean.dataQuery.model;

public class AiotModel {

private int id;

private int deviceid;

private String indicatorcode;

private String collecttime;

private String data;

private String unit;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getDeviceid() {

return deviceid;

}

public void setDeviceid(int deviceid) {

this.deviceid = deviceid;

}

public String getIndicatorcode() {

return indicatorcode;

}

public void setIndicatorcode(String indicatorcode) {

this.indicatorcode = indicatorcode;

}

public String getCollecttime() {

return collecttime;

}

public void setCollecttime(String collecttime) {

this.collecttime = collecttime;

}

public String getData() {

return data;

}

public void setData(String data) {

this.data = data;

}

public String getUnit() {

return unit;

}

public void setUnit(String unit) {

this.unit = unit;

}

}

package com.sdocean.dataQuery.model;

import com.sdocean.device.model.DeviceModel;

import com.sdocean.indicator.model.IndicatorModel;

import com.sdocean.page.model.PageResult;

import com.sdocean.station.model.StationModel;

public class DataChangeModel {

private int id;

private int stationId;

private StationModel station;

private int deviceId;

private DeviceModel device;

private String indicatorCode;

private IndicatorModel indicator;

private String indicatorIds;

private PageResult page;

private String collect\_time;

private double newData;

private double oldData;

private String beginDate;

private String endDate;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public StationModel getStation() {

return station;

}

public void setStation(StationModel station) {

this.station = station;

}

public DeviceModel getDevice() {

return device;

}

public void setDevice(DeviceModel device) {

this.device = device;

}

public IndicatorModel getIndicator() {

return indicator;

}

public void setIndicator(IndicatorModel indicator) {

this.indicator = indicator;

}

public String getBeginDate() {

return beginDate;

}

public void setBeginDate(String beginDate) {

this.beginDate = beginDate;

}

public String getEndDate() {

return endDate;

}

public void setEndDate(String endDate) {

this.endDate = endDate;

}

public PageResult getPage() {

return page;

}

public void setPage(PageResult page) {

this.page = page;

}

public String getIndicatorIds() {

return indicatorIds;

}

public void setIndicatorIds(String indicatorIds) {

this.indicatorIds = indicatorIds;

}

public int getStationId() {

return stationId;

}

public void setStationId(int stationId) {

this.stationId = stationId;

}

public int getDeviceId() {

return deviceId;

}

public void setDeviceId(int deviceId) {

this.deviceId = deviceId;

}

public String getIndicatorCode() {

return indicatorCode;

}

public void setIndicatorCode(String indicatorCode) {

this.indicatorCode = indicatorCode;

}

public String getCollect\_time() {

return collect\_time;

}

public void setCollect\_time(String collect\_time) {

this.collect\_time = collect\_time;

}

public double getNewData() {

return newData;

}

public void setNewData(double newData) {

this.newData = newData;

}

public double getOldData() {

return oldData;

}

public void setOldData(double oldData) {

this.oldData = oldData;

}

}

package com.sdocean.dataQuery.model;

public class DataImportModel {

private String importString;

private int stationId;

private int deviceId;

private String indicatorCode;

private String collect\_time;

private Double data;

public String getCollect\_time() {

return collect\_time;

}

public void setCollect\_time(String collect\_time) {

this.collect\_time = collect\_time;

}

public Double getData() {

return data;

}

public void setData(Double data) {

this.data = data;

}

public String getImportString() {

return importString;

}

public void setImportString(String importString) {

this.importString = importString;

}

public int getStationId() {

return stationId;

}

public void setStationId(int stationId) {

this.stationId = stationId;

}

public int getDeviceId() {

return deviceId;

}

public void setDeviceId(int deviceId) {

this.deviceId = deviceId;

}

public String getIndicatorCode() {

return indicatorCode;

}

public void setIndicatorCode(String indicatorCode) {

this.indicatorCode = indicatorCode;

}

}

package com.sdocean.dataQuery.model;

import java.util.List;

import java.util.Map;

import com.sdocean.page.model.UiColumn;

public class DataQueryData {

private List<UiColumn> cols;

private List<Map<String, Object>> rows;

private List<DataQueryDataModel> images;

private List<AiotModel> aiots;

public List<AiotModel> getAiots() {

return aiots;

}

public void setAiots(List<AiotModel> aiots) {

this.aiots = aiots;

}

public List<UiColumn> getCols() {

return cols;

}

public void setCols(List<UiColumn> cols) {

this.cols = cols;

}

public List<Map<String, Object>> getRows() {

return rows;

}

public void setRows(List<Map<String, Object>> rows) {

this.rows = rows;

}

public List<DataQueryDataModel> getImages() {

return images;

}

public void setImages(List<DataQueryDataModel> images) {

this.images = images;

}

}

package com.sdocean.dataQuery.model;

import java.util.List;

import java.util.Map;

import com.sdocean.page.model.UiColumn;

public class DataQueryData {

private List<UiColumn> cols;

private List<Map<String, Object>> rows;

private List<DataQueryDataModel> images;

private List<AiotModel> aiots;

public List<AiotModel> getAiots() {

return aiots;

}

public void setAiots(List<AiotModel> aiots) {

this.aiots = aiots;

}

public List<UiColumn> getCols() {

return cols;

}

public void setCols(List<UiColumn> cols) {

this.cols = cols;

}

public List<Map<String, Object>> getRows() {

return rows;

}

public void setRows(List<Map<String, Object>> rows) {

this.rows = rows;

}

public List<DataQueryDataModel> getImages() {

return images;

}

public void setImages(List<DataQueryDataModel> images) {

this.images = images;

}

}

package com.sdocean.dataQuery.model;

import java.util.List;

import java.util.Map;

import com.sdocean.page.model.UiColumn;

public class DataQueryData {

private List<UiColumn> cols;

private List<Map<String, Object>> rows;

private List<DataQueryDataModel> images;

private List<AiotModel> aiots;

public List<AiotModel> getAiots() {

return aiots;

}

public void setAiots(List<AiotModel> aiots) {

this.aiots = aiots;

}

public List<UiColumn> getCols() {

return cols;

}

public void setCols(List<UiColumn> cols) {

this.cols = cols;

}

public List<Map<String, Object>> getRows() {

return rows;

}

public void setRows(List<Map<String, Object>> rows) {

this.rows = rows;

}

public List<DataQueryDataModel> getImages() {

return images;

}

public void setImages(List<DataQueryDataModel> images) {

this.images = images;

}

}

package com.sdocean.sms.model;

public class SmsSettingModel {

private int stationId;

private String stationName;

private int userId;

private String userIds;

private String userName;

private String telephone;

private int mon;

private String monValue;

private int tues;

private String tuesValue;

private int wed;

private String wedValue;

private int thur;

private String thurValue;

private int fri;

private String friValue;

private int satur;

private String saturValue;

private int sun;

private String sunValue;

private int betweenTime;

private int isactive;

private String isactiveName; //是否接收短信 1接收,0 不接受

private String ambegin;

private String amend;

private String pmbegin;

private String pmend;

private String type;

public String getUserIds() {

return userIds;

}

public void setUserIds(String userIds) {

this.userIds = userIds;

}

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public int getIsactive() {

return isactive;

}

public void setIsactive(int isactive) {

this.isactive = isactive;

}

public String getIsactiveName() {

return isactiveName;

}

public void setIsactiveName(String isactiveName) {

this.isactiveName = isactiveName;

}

public String getPmbegin() {

return pmbegin;

}

public void setPmbegin(String pmbegin) {

this.pmbegin = pmbegin;

}

public String getAmbegin() {

return ambegin;

}

public void setAmbegin(String ambegin) {

this.ambegin = ambegin;

}

public String getAmend() {

return amend;

}

public void setAmend(String amend) {

this.amend = amend;

}

public String getPmend() {

return pmend;

}

public void setPmend(String pmend) {

this.pmend = pmend;

}

public String getTelephone() {

return telephone;

}

public void setTelephone(String telephone) {

this.telephone = telephone;

}

public int getStationId() {

return stationId;

}

public void setStationId(int stationId) {

this.stationId = stationId;

}

public String getStationName() {

return stationName;

}

public void setStationName(String stationName) {

this.stationName = stationName;

}

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public int getMon() {

return mon;

}

public void setMon(int mon) {

this.mon = mon;

}

public String getMonValue() {

return monValue;

}

public void setMonValue(String monValue) {

this.monValue = monValue;

}

public int getTues() {

return tues;

}

public void setTues(int tues) {

this.tues = tues;

}

public String getTuesValue() {

return tuesValue;

}

public void setTuesValue(String tuesValue) {

this.tuesValue = tuesValue;

}

public int getWed() {

return wed;

}

public void setWed(int wed) {

this.wed = wed;

}

public String getWedValue() {

return wedValue;

}

public void setWedValue(String wedValue) {

this.wedValue = wedValue;

}

public int getThur() {

return thur;

}

public void setThur(int thur) {

this.thur = thur;

}

public String getThurValue() {

return thurValue;

}

public void setThurValue(String thurValue) {

this.thurValue = thurValue;

}

public int getFri() {

return fri;

}

public void setFri(int fri) {

this.fri = fri;

}

public String getFriValue() {

return friValue;

}

public void setFriValue(String friValue) {

this.friValue = friValue;

}

public int getSatur() {

return satur;

}

public void setSatur(int satur) {

this.satur = satur;

}

public String getSaturValue() {

return saturValue;

}

public void setSaturValue(String saturValue) {

this.saturValue = saturValue;

}

public int getSun() {

return sun;

}

public void setSun(int sun) {

this.sun = sun;

}

public String getSunValue() {

return sunValue;

}

public void setSunValue(String sunValue) {

this.sunValue = sunValue;

}

public int getBetweenTime() {

return betweenTime;

}

public void setBetweenTime(int betweenTime) {

this.betweenTime = betweenTime;

}

}

package com.sdocean.sms.service;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Propagation;

import org.springframework.transaction.annotation.Transactional;

import com.sdocean.common.model.Result;

import com.sdocean.page.model.UiColumn;

import com.sdocean.sms.dao.SmsDao;

import com.sdocean.sms.model.SmsLog;

import com.sdocean.sms.model.SmsMouldModel;

import com.sdocean.sms.model.SmsSettingModel;

import com.sdocean.users.model.SysUser;

@Service

@Transactional(rollbackFor = Exception.class, propagation = Propagation.REQUIRED)

public class SmsService {

@Resource

private SmsDao smsDao;

/\*

\* 为人员管理的查询结果添加表头

\*/

public List<UiColumn> getCols4TypeList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

//UiColumn col1 = new UiColumn("stationId", "stationId", false, "\*");

//UiColumn col2 = new UiColumn("stationName", "stationName", true, "\*");

UiColumn col3 = new UiColumn("userId", "userId", false, "\*");

UiColumn col6 = new UiColumn("用户", "userName", true, "\*");

UiColumn col7 = new UiColumn("手机号", "telephone", true, "\*");

UiColumn col8 = new UiColumn("周一", "monValue", false, "\*");

UiColumn col9 = new UiColumn("周二", "tuesValue", false, "\*");

UiColumn col10 = new UiColumn("周三", "wedValue", false, "\*");

UiColumn col11 = new UiColumn("周四", "thurValue", false, "\*");

UiColumn col12 = new UiColumn("周五", "friValue", false, "\*");

UiColumn col13 = new UiColumn("周六", "saturValue", false, "\*");

UiColumn col14 = new UiColumn("周日", "sunValue", false, "\*");

UiColumn col15 = new UiColumn("周一", "mon", true, "\*");

UiColumn col16 = new UiColumn("周二", "tues", true, "\*");

UiColumn col17 = new UiColumn("周三", "wed", true, "\*");

UiColumn col21 = new UiColumn("周四", "thur", true, "\*");

UiColumn col18 = new UiColumn("周五", "fri", true, "\*");

UiColumn col19 = new UiColumn("周六", "satur", true, "\*");

UiColumn col20= new UiColumn("周日", "sun", true, "\*");

UiColumn col22= new UiColumn("间隔发送时间", "betweenTime", true, "\*");

UiColumn col27= new UiColumn("是否接收短信", "isactiveName", true, "\*");

UiColumn col23= new UiColumn("上午起始时间", "ambegin", true, "\*");

UiColumn col24= new UiColumn("上午截止时间", "amend", true, "\*");

UiColumn col25= new UiColumn("下午起始时间", "pmbegin", true, "\*");

UiColumn col26= new UiColumn("下午截止时间", "pmend", true, "\*");

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

cols.add(col9);

cols.add(col10);

cols.add(col11);

cols.add(col12);

cols.add(col13);

cols.add(col14);

cols.add(col15);

cols.add(col16);

cols.add(col17);

cols.add(col18);

cols.add(col19);

cols.add(col20);

cols.add(col21);

cols.add(col22);

cols.add(col23);

cols.add(col24);

cols.add(col25);

cols.add(col26);

cols.add(col27);

return cols;

}

/\*

\* 查询某站点下需要接受短信的人员列表

\*/

public List<SmsSettingModel> getUsers4SmsSetting(SmsSettingModel model,SysUser user){

return smsDao.getUsers4SmsSetting(model, user);

}

/\*

\* 初始化用户下的短信配置

\*/

public SmsSettingModel infoUserSetting(SysUser user){

return smsDao.infoUserSetting(user);

}

/\*

\* 保存用户短信配置

\*/

public Result saveUserSetting(SmsSettingModel model){

return smsDao.saveUserSetting(model);

}

/\*

\* 展示短信模板

\*/

public List<UiColumn> getCols4MouldList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*");

UiColumn col2 = new UiColumn("stationId", "stationId", false, "\*");

UiColumn col3 = new UiColumn("站点", "stationName", true, "\*");

UiColumn col6 = new UiColumn("type", "type", false, "\*");

UiColumn col7 = new UiColumn("模板类型", "typeName", true, "\*");

UiColumn col8 = new UiColumn("模板", "mould", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

return cols;

}

/\*

\* 为展示模板提供结果

\*/

public List<SmsMouldModel> getSmsMoulds4Rows(SmsMouldModel model){

return smsDao.getSmsMoulds4Rows(model);

}

/\*

\* 对短信发送模板修改

\*/

public Result saveSmsMouldChange(SmsMouldModel model){

return smsDao.saveSmsMouldChange(model);

}

/\*

\* 保存新增的短信发送模板

\*/

public Result saveNewSmsMould(SmsMouldModel model){

return smsDao.saveNewSmsMould(model);

}

/\*

\* 展示短信发送记录

\*/

public List<UiColumn> getCols4LogList(){

List<UiColumn> cols = new ArrayList<UiColumn>();

UiColumn col1 = new UiColumn("id", "id", false, "\*");

UiColumn col2 = new UiColumn("userId", "userId", false, "\*");

UiColumn col3 = new UiColumn("用户名", "userName", true, "\*");

UiColumn col6 = new UiColumn("手机号", "telephone", true, "\*");

UiColumn col7 = new UiColumn("短信", "message", true, "\*");

UiColumn col8 = new UiColumn("ifsend", "ifsend", false, "\*");

UiColumn col9 = new UiColumn("发送状态", "ifSendName", true, "\*");

cols.add(col1);

cols.add(col2);

cols.add(col3);

cols.add(col6);

cols.add(col7);

cols.add(col8);

cols.add(col9);

return cols;

}

/\*

\* 查询短信发送记录,

\* 增加模糊查询

\*/

public List<SmsLog> getSmsLogs(SmsLog model){

return smsDao.getSmsLogs(model);

}

/\*

\* 保存短信发送配置

\*/

public Result saveSmsStationSetting(SmsSettingModel model){

return smsDao.saveSmsStationSetting(model);

}

}

package com.sdocean.station.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.google.gson.JsonIOException;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.common.service.CommonService;

import com.sdocean.common.service.ZTreeService;

import com.sdocean.company.model.CompanyModel;

import com.sdocean.company.service.CompanyService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.login.action.LoginAction;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.region.model.RegionModel;

import com.sdocean.region.service.RegionService;

import com.sdocean.role.model.RoleModel;

import com.sdocean.station.model.PlatformModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.PlatformService;

import com.sdocean.station.service.StationService;

import com.sdocean.users.model.SysUser;

@Controller

public class PlatformAction {

private static Logger log = Logger.getLogger(PlatformAction.class);

@Resource

PlatformService platformService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

/\*

\* 跳转到平台编辑页面

\*/

@RequestMapping("info\_platform.do")

public ModelAndView info\_platform(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/platform/platformInfo");

return mav;

}

/\*

\* 展示平台列表

\*/

@RequestMapping(value="showPlatformList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showPlatformList(@ModelAttribute("model") PlatformModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = platformService.getCols4PlatformList();

result.setCols(cols);

//为查询结果增加列表

List<PlatformModel> rows = platformService.getPlatformList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 获得平台列表

\*/

@RequestMapping(value="getPlatformList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getPlatformList(@ModelAttribute("model") PlatformModel model,HttpServletRequest request,

HttpServletResponse response){

List<PlatformModel> rows = platformService.getPlatformList(model);

return JsonUtil.toJson(rows);

}

/\*

\* 新增平台

\*/

@RequestMapping(value="addPlatform.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String addPlatform(@ModelAttribute("model") PlatformModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = platformService.addPlatform(model);

return result.getMessage();

}

/\*

\* 修改平台

\*/

@RequestMapping(value="savePlatform.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String savePlatform(@ModelAttribute("model") PlatformModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = platformService.savePlatform(model);

return result.getMessage();

}

/\*

\* 删除平台

\*/

@RequestMapping(value="delePlatform.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String delePlatform(@ModelAttribute("model") PlatformModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = platformService.delePlatform(model);

return result.getMessage();

}

}

package com.sdocean.station.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.google.gson.JsonIOException;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.common.service.CommonService;

import com.sdocean.common.service.ZTreeService;

import com.sdocean.company.model.CompanyModel;

import com.sdocean.company.service.CompanyService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.login.action.LoginAction;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.region.model.RegionModel;

import com.sdocean.region.service.RegionService;

import com.sdocean.role.model.RoleModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationService;

import com.sdocean.users.model.SysUser;

@Controller

public class StationAction {

private static Logger log = Logger.getLogger(StationAction.class);

@Resource

StationService stationService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@Autowired

private CompanyService companyService;

@Autowired

private ZTreeService ztreeService;

@Autowired

private RegionService regionService;

@RequestMapping("info\_station.do")

public ModelAndView info\_station(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/station/stationInfo");

return mav;

}

@RequestMapping(value="showStationList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showStationTypes(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = stationService.getCols4TypeList();

result.setCols(cols);

//为查询结果增加列表

List<StationModel> rows = stationService.getStationList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改站点信息

\*/

@RequestMapping(value="saveStaionChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveStaionChange(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = stationService.saveStaionChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增站点信息

\*/

@RequestMapping(value="saveNewStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewStation(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = stationService.saveNewStation(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 获得ZTREE展示列表

\*/

@RequestMapping(value="getStationList4ZTree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationList4ZTree(@ModelAttribute("role") RoleModel role,HttpServletRequest request,

HttpServletResponse response){

List<ZTreeModel> result = new ArrayList<ZTreeModel>();

result = stationService.getStationListByRole(role);

return JsonUtil.toJson(result);

}

/\*

\* 获得session中保存的station

\*/

@RequestMapping(value="getStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation(HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

StationModel station = (StationModel) session.getAttribute("station");

return JsonUtil.toJson(station);

}

/\*

\* 获得当前用户权限下的站点列表

\*/

@RequestMapping(value="getStationByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationByUser(HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> list = stationService.getStations4User(user);

return JsonUtil.toJson(list);

}

@RequestMapping(value="showStationList4Sms.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showStationList4Sms(HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = stationService.getStationCols4Sms();

result.setCols(cols);

//为查询结果增加列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> rows = stationService.getStations4User(user);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 获得组织机构下的站点设备列表,

\* 作为载体的参数,不接受跳转

\*/

@RequestMapping(value="getStationDevice4ComanyZtree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationDevice4ComanyZtree(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

if(cdevices==null||cdevices.size()<1){

z.setUrl(ztree.getUrl()+z.getId());

}

for(ZTreeModel c:cdevices){

c.setUrl(ztree.getUrl()+c.getId());

}

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得组织机构的站点列表

\*/

@RequestMapping(value="getStation4ComanyZtreeByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation4ComanyZtreeByUser(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

station.setUrl(ztree.getUrl()+station.getId());

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得组织机构的站点\1级设备\2级设备的树形菜单

\*/

@RequestMapping(value="getStationDevicesTree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationDevicesTree(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

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

//设置点击站点时的跳转列表

station.setUrl(ztree.getStationURL()+station.getId());

}

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

//设置点击一级设备时的跳转列表

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

z.setUrl(ztree.getDeviceURL()+z.getId());

}

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

for(ZTreeModel c:cdevices){

//设置点击二级设备时的跳转列表

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

c.setUrl(ztree.getDeviceChildURL()+c.getId());

}

}

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得地区树下的站点列表

\*/

@RequestMapping(value="getStation4RegionZtreeByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation4RegionZtreeByUser(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表得到对应的地区列表

List<RegionModel> regions = new ArrayList<>();

List<RegionModel> regs = new ArrayList<>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

regions.addAll(regionService.getRegionListById(station.getRegion\_id()));

station.setUrl(ztree.getUrl()+station.getId());

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

for(RegionModel region:regions){

if(!regs.contains(region)){

regs.add(region);

}

}

//将地区机构列表以及站点列表转换成ZTREE的形式

List<ZTreeModel> regztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

try {

regztrees = ztreeService.changeModel2ZTree(regs, "id", "pid", "text",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "region\_id", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(regztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

}

package com.sdocean.station.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.google.gson.JsonIOException;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.common.service.CommonService;

import com.sdocean.common.service.ZTreeService;

import com.sdocean.company.model.CompanyModel;

import com.sdocean.company.service.CompanyService;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.login.action.LoginAction;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.region.model.RegionModel;

import com.sdocean.region.service.RegionService;

import com.sdocean.role.model.RoleModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationService;

import com.sdocean.users.model.SysUser;

@Controller

public class StationAction {

private static Logger log = Logger.getLogger(StationAction.class);

@Resource

StationService stationService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@Autowired

private CompanyService companyService;

@Autowired

private ZTreeService ztreeService;

@Autowired

private RegionService regionService;

@RequestMapping("info\_station.do")

public ModelAndView info\_station(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/station/stationInfo");

return mav;

}

@RequestMapping(value="showStationList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showStationTypes(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = stationService.getCols4TypeList();

result.setCols(cols);

//为查询结果增加列表

List<StationModel> rows = stationService.getStationList(model);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 保存修改站点信息

\*/

@RequestMapping(value="saveStaionChange.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveStaionChange(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = stationService.saveStaionChange(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 保存新增站点信息

\*/

@RequestMapping(value="saveNewStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewStation(@ModelAttribute("model") StationModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = new Result();

result = stationService.saveNewStation(model);

logService.saveOperationLog(result,request);

return result.getMessage();

}

/\*

\* 获得ZTREE展示列表

\*/

@RequestMapping(value="getStationList4ZTree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationList4ZTree(@ModelAttribute("role") RoleModel role,HttpServletRequest request,

HttpServletResponse response){

List<ZTreeModel> result = new ArrayList<ZTreeModel>();

result = stationService.getStationListByRole(role);

return JsonUtil.toJson(result);

}

/\*

\* 获得session中保存的station

\*/

@RequestMapping(value="getStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation(HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

StationModel station = (StationModel) session.getAttribute("station");

return JsonUtil.toJson(station);

}

/\*

\* 获得当前用户权限下的站点列表

\*/

@RequestMapping(value="getStationByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationByUser(HttpServletRequest request,

HttpServletResponse response){

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> list = stationService.getStations4User(user);

return JsonUtil.toJson(list);

}

@RequestMapping(value="showStationList4Sms.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showStationList4Sms(HttpServletRequest request,

HttpServletResponse response){

PageResult result = new PageResult();

//为查询结果增加表头

List<UiColumn> cols = stationService.getStationCols4Sms();

result.setCols(cols);

//为查询结果增加列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> rows = stationService.getStations4User(user);

result.setRows(rows);

return JsonUtil.toJson(result);

}

/\*

\* 获得组织机构下的站点设备列表,

\* 作为载体的参数,不接受跳转

\*/

@RequestMapping(value="getStationDevice4ComanyZtree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationDevice4ComanyZtree(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

if(cdevices==null||cdevices.size()<1){

z.setUrl(ztree.getUrl()+z.getId());

}

for(ZTreeModel c:cdevices){

c.setUrl(ztree.getUrl()+c.getId());

}

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得组织机构的站点列表

\*/

@RequestMapping(value="getStation4ComanyZtreeByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation4ComanyZtreeByUser(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

station.setUrl(ztree.getUrl()+station.getId());

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得组织机构的站点\1级设备\2级设备的树形菜单

\*/

@RequestMapping(value="getStationDevicesTree.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationDevicesTree(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表获得对应的组织机构列表

List<CompanyModel> comps = new ArrayList<CompanyModel>();

List<ZTreeModel> comztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

comps.addAll(companyService.getComListByCode(station.getCompanyId()));

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

//设置点击站点时的跳转列表

station.setUrl(ztree.getStationURL()+station.getId());

}

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

//设置点击一级设备时的跳转列表

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

z.setUrl(ztree.getDeviceURL()+z.getId());

}

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

for(ZTreeModel c:cdevices){

//设置点击二级设备时的跳转列表

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

c.setUrl(ztree.getDeviceChildURL()+c.getId());

}

}

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

List<CompanyModel> coms = new ArrayList<CompanyModel>();

//将重复的组织机构去掉

for(CompanyModel com:comps){

if(!coms.contains(com)){

coms.add(com);

}

}

//将组织机构列表以及站点列表转换成ztree的形式

try {

comztrees = ztreeService.changeModel2ZTree(coms, "code", "pcode", "shortName",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "companyId", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(comztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

/\*

\* 获得地区树下的站点列表

\*/

@RequestMapping(value="getStation4RegionZtreeByUser.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation4RegionZtreeByUser(HttpServletRequest request,

HttpServletResponse response,ZTreeModel ztree){

String syspath = "http://"+request.getHeader("host")+request.getContextPath();

List<ZTreeModel> ztrees = new ArrayList<ZTreeModel>();

//获得用户权限下的站点的列表

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

List<StationModel> stations = stationService.getStations4User(user);

//根据站点列表得到对应的地区列表

List<RegionModel> regions = new ArrayList<>();

List<RegionModel> regs = new ArrayList<>();

List<ZTreeModel> deviceztrees = new ArrayList<ZTreeModel>();

for(StationModel station:stations){

regions.addAll(regionService.getRegionListById(station.getRegion\_id()));

station.setUrl(ztree.getUrl()+station.getId());

List<ZTreeModel> devices = stationService.getDeviceZTree4Station(station);

for(ZTreeModel z:devices){

List<ZTreeModel> cdevices = stationService.getDeviceZTree4Device(z,station.getId());

z.setChildren(cdevices);

}

deviceztrees.addAll(devices);

}

for(RegionModel region:regions){

if(!regs.contains(region)){

regs.add(region);

}

}

//将地区机构列表以及站点列表转换成ZTREE的形式

List<ZTreeModel> regztrees = new ArrayList<ZTreeModel>();

List<ZTreeModel> stationztrees = new ArrayList<ZTreeModel>();

try {

regztrees = ztreeService.changeModel2ZTree(regs, "id", "pid", "text",true,"","","",syspath);

stationztrees = ztreeService.changeModel2ZTree(stations, "id", "region\_id", "title",true,"url",ZTreeModel.getSelf(),"icon",syspath);

ztrees.addAll(regztrees);

ztrees.addAll(stationztrees);

ztrees.addAll(deviceztrees);

} catch (Exception e) {

// TODO: handle exception

e.printStackTrace();

}

return JsonUtil.toJson(ztrees);

}

}

package com.sdocean.station.action;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationDeviceModel;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationDeviceService;

@Controller

public class StationDeviceAction {

@Resource

StationDeviceService stationDeviceService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_stationDevice.do")

public ModelAndView info\_station(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/station/stationDeviceInfo");

return mav;

}

/\*

\* 展示站点设备配置列表

\*/

@RequestMapping(value="showStationDeviceList.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String showStationDeviceList(@ModelAttribute("model") StationDeviceModel model,HttpServletRequest request,

HttpServletResponse response){

//获得该用户的站点权限

HttpSession session = request.getSession();

List<StationModel> stations = (List<StationModel>) session.getAttribute("stations");

PageResult page = new PageResult();

List<UiColumn> cols = stationDeviceService.getCols4StationDeviceList();

List<StationDeviceModel> rows = stationDeviceService.getStationDeviceList(model,stations);

page.setCols(cols);

page.setRows(rows);

return JsonUtil.toJson(page);

}

/\*

\* 保存新增的站点设备配置

\*/

@RequestMapping(value="saveNewStationDevice.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveNewStationDevice(@ModelAttribute("model") StationDeviceModel model,HttpServletRequest request,

HttpServletResponse response){

//判断创建时间

if(model!=null&&(model.getCreateTime()==null||model.getCreateTime().length()<1)){

Date now = new Date();

SimpleDateFormat sdf=new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

model.setCreateTime(sdf.format(now));

}

Result result = stationDeviceService.saveNewStationDevice(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 保存修改的站点设备配置

\*/

@RequestMapping(value="saveChangeStationDevice.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveChangeStationDevice(@ModelAttribute("model") StationDeviceModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = stationDeviceService.saveChangeStationDevice(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 删除站点设备配置

\*/

@RequestMapping(value="deleteStationDevice.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String deleteStationDevice(@ModelAttribute("model") StationDeviceModel model,HttpServletRequest request,

HttpServletResponse response){

Result result = stationDeviceService.deleteStationDevice(model);

//保存操作日志

logService.saveOperationLog(result, request);

return result.getMessage();

}

/\*

\* 根据站点ID，设备ID或的站点设备配置

\*/

@RequestMapping(value="getStationDeviceByWidDid.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStationDeviceByWidDid(@ModelAttribute("model") StationDeviceModel model,HttpServletRequest request,

HttpServletResponse response){

StationDeviceModel result = stationDeviceService.getStationDeviceByWidDid(model);

return JsonUtil.toJson(result);

}

}

package com.sdocean.station.action;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.servlet.ModelAndView;

import com.sdocean.common.model.Result;

import com.sdocean.common.model.ZTreeModel;

import com.sdocean.frame.model.ConfigInfo;

import com.sdocean.frame.util.JsonUtil;

import com.sdocean.log.service.OperationLogService;

import com.sdocean.page.model.PageResult;

import com.sdocean.page.model.UiColumn;

import com.sdocean.station.model.StationDeviceIndicator;

import com.sdocean.station.model.StationModel;

import com.sdocean.station.service.StationDeviceIndicatorService;

import com.sdocean.users.model.SysUser;

@Controller

public class StationDeviceIndicatorAction {

@Resource

StationDeviceIndicatorService sdiService;

@Resource

OperationLogService logService;

@Autowired

private ConfigInfo info;

@RequestMapping("info\_stationIndicator.do")

public ModelAndView info\_station(HttpServletRequest request,

HttpServletResponse response)throws Exception{

ModelAndView mav = new ModelAndView("/"+info.getManageVision()+"/station/stationIndicator");

return mav;

}

@RequestMapping(value="getDeiviceIndicatorsByStation.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getDeiviceIndicatorsByStation(@ModelAttribute("model") StationDeviceIndicator model,HttpServletRequest request,

HttpServletResponse response){

List<ZTreeModel> list = new ArrayList<ZTreeModel>();

list = sdiService.getDeiviceIndicatorsByStation(model.getStationId());

return JsonUtil.toJson(list);

}

/\*

\* 获得用户权限下的站点的列表

\*/

@RequestMapping(value="getStation4DeviceIndicator.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String getStation4DeviceIndicator(@ModelAttribute("model") StationDeviceIndicator model,HttpServletRequest request,

HttpServletResponse response){

PageResult page = new PageResult();

List<UiColumn> cols = sdiService.getCols4Stations();

List<StationModel> stations = new ArrayList<StationModel>();

HttpSession session = request.getSession();

SysUser user = (SysUser) session.getAttribute("user");

stations = sdiService.getStation4DeviceIndicator(user.getId(), model.getStationName());

page.setCols(cols);

page.setRows(stations);

return JsonUtil.toJson(page);

}

/\*

\* 保存站点设备参数

\*/

@RequestMapping(value="saveStationDeviceIndicator.do", method = RequestMethod.POST,produces = "application/json;charset=UTF-8")

@ResponseBody

public String saveStationDeviceIndicator(@ModelAttribute("model") StationDeviceIndicator model,HttpServletRequest request,

HttpServletResponse response){

Result result = sdiService.saveStationDeviceIndicator(model);

logService.saveOperationLog(result, request);

return JsonUtil.toJson(result.getMessage());

}

}

<%@ page contentType="text/html; charset=UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>

<!DOCTYPE html>

<html >

<c:set var="ctx" value="${pageContext.request.contextPath}"/>

<head>

<meta charset="utf-8" />

<title>${system.systemName }</title>

<meta name="keywords" content="入海污染源" />

<script language="javascript">

var dooropenfunction= function (){

if(document.getElementById("door\_left").width>2&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

document.getElementById("door\_left").width-=2;

document.getElementById("door\_right").width-=2;

}else{

dooropenfunction=null;

}

}

var doorclosfunction = function(){

if(document.getElementById("door\_left").width<26&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

document.getElementById("door\_left").width+=2;

document.getElementById("door\_right").width+=2;

}

else{

doorclosfunction=null;

}

}

function showdoor(){

if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

window.setInterval(dooropenfunction, 200);

}else if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

window.setInterval(doorclosfunction, 200);

}

}

</script>

<style>

.kongzhimianban-imgnew{

width:250px;

height:230px;

background-image:url(img/donghua1.jpg);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

.kongzhimianban-img{

width:250px;

height:230px;

background-image:url(img/donghua2.jpg);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

#door\_left{

margin-top: 66px;

margin-left: 146px;

}

#door\_right{

margin-top: 66px;

margin-left:-3px;

float: right;

margin-right: 20px;

}

#demo {

background: #FFF;

overflow:hidden;

border: 0;

width: 20px;

height:8px;

margin-left: 110px;

margin-top: -2px;

}

#indemo {

float: left;

width: 800%;

}

#demo1 {

float: left;

}

#demo2 {

float: left;

}

#demo1 img{

margin-top:-11px;

height:8px;

width:20px;

}

#demo2 img{

margin-top:-4px;

}

</style>

</head>

<body ng-app="myApp" ng-controller="customersCtrl">

<%@ include file="../common/top.jsp" %>

<link rel="stylesheet" href="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.css" />

<script type="text/javascript" src="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.js"></script>

<script type="text/javascript" src="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.tpl.js"></script>

<script src="${ctx }/manage/js/device/deviceInfo.js"></script>

<div class="main-container" id="main-container">

<script type="text/javascript">

try{ace.settings.check('main-container' , 'fixed')}catch(e){}

</script>

<div class="main-container-inner">

<%@ include file="../common/leftMenu4device.jsp" %>

<div class="main-content" >

<div class="breadcrumbs" id="breadcrumbs">

<script type="text/javascript">

try{ace.settings.check('breadcrumbs' , 'fixed')}catch(e){}

</script>

<ul class="breadcrumb">

<li>

<i class="icon-home home-icon"></i>

<a href="#">${currMenu.pMenuName }</a>

</li>

<li>

<a href="#">${currMenu.cMenuName }</a>

</li>

</ul><!-- .breadcrumb -->

</div>

<div class="container">

<div class="row">

<div class="col-md-12 col-sm-12 col-sx-12 map-main">

<div class="panel panel-default">

<div class="panel-body panel-body-margin" >

<div class="row">

<div class="col-md-9 col-sm-9 map-right ">

<div class="panel panel-default panel-default-right">

<ul id="myTab" class="nav nav-tabs nav-tabs-tongji ">

<li class="">

<a href="#activites" data-toggle="tab" data-original-title="">自动控制</a>

</li>

<li class="active">

<a href="#signups" data-toggle="tab" data-original-title="">手动控制</a>

</li>

</ul>

<div class="panel-body panel-body-tongji">

<section class="todo">

<fieldset class="todo-list">

<div id="myTabContent" class="tab-content">

<div class="tab-pane fade" id="activites">

<div class="row">

<div class="col-md-8 col-sm-8 kongzhi-height-left">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">控制面板</h3>

</div>

<div class="panel-body">

<div class="row">

<div class="col-md-7 row-kongzhimianban kongzhimianban-img ">

</div>

<div class="col-md-5 row-kongzhimianban">

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">进水泵</span>

<select class="form-control input-sm" disabled>

<option value="">浅层</option>

<option value="">深层</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">泵方向</span>

<select class="form-control input-sm " disabled>

<option value="">进水</option>

<option value="">出水</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">门状态</span>

<select class="form-control input-sm " disabled>

<option value="">开</option>

<option value="">关</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon input-group-addon-control " >泵控制</span>

<select class="form-control input-sm" disabled>

<option value="">开</option>

<option value="">关</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">油电源</span>

<select class="form-control input-sm " disabled>

<option value="">开</option>

<option value="">关</option>

</select>

</div>

</div>

<div class="col-md-12">

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>潜水管清洗</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>深水管清洗</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>排空气</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>校准</button>

</div>

</div>

</div>

</div>

</div>

</div>

<div class="col-md-4 col-sm-4 kongzhi-height-right">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">时间间隔参数</h3>

</div>

<div class="panel-body">

<div class="input-group ">

<span class="input-group-addon ">物理参数</span>

<select class="form-control input-sm" disabled>

<option value="">20min</option>

<option value="">30min</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon ">营养盐</span>

<select class="form-control input-sm" disabled>

<option value="">20min</option>

<option value="">30min</option>

</select>

</div>

<div class="table-responsive ">

<table class="table no-margin">

<thead>

<tr>

<th>#</th>

<th>排水</th>

<th>进水</th>

<th>进水选择</th>

</tr>

</thead>

<tbody>

<tr>

<td>浅层</td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

</tr>

<tr>

<td>深层</td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

</tr>

</tbody>

</table>

</div>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">更新</button>

</div>

</div>

</div>

</div>

<div class="col-md-12">

<div class="panel-body center-align-text">

<span >数据浏览</span>

<select >

<option value="">浅层</option>

<option value="">深层</option>

</select>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新水质探头</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新油探头</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新营养盐标准品</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新营养盐样品</span>

</label>

<button type="button" class="btn btn-xs btn-default pilianggengxin " data-original-title="" disabled>自动批量更新</button>

</div>

</div>

<div class="col-md-12">

<table>

<tbody><tr>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">水质参数</h3>

</div>

<div class="panel-body">

<table>

<tbody>

<tr>

<td>蓝绿藻：</td>

<td>XXXX</td>

</tr>

<tr>

<td>叶绿素：</td>

<td>XXXX</td>

</tr>

<tr>

<td>电导率：</td>

<td>XXXX</td>

</tr>

<tr>

<td>溶解氧</td>

<td>XXXX</td>

</tr>

<tr>

<td>PH值：</td>

<td>XXXX</td>

</tr>

<tr>

<td>温度：</td>

<td>XXXX</td>

</tr>

<tr>

<td>浊度：</td>

<td>XXXX</td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">油参数</h3>

</div>

<div class="panel-body canshu-table">

<table>

<tbody>

<tr>

<td>原油：</td>

<td>16.9</td>

</tr>

<tr>

<td>成品油：</td>

<td>79.9</td>

</tr>

<tr>

<td>温度：</td>

<td>5.6.C</td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">营养盐标准品</h3>

</div>

<div class="panel-body canshu-table">

<table>

<tbody>

<tr>

<td>硝酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>磷酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>氨：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>硅酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">营养盐样品</h3>

</div>

<div class="panel-body canshu-table">

<table>

<tbody><tr>

<td>硝酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>磷酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>氨：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>硅酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

</tr><tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">天气参数</h3>

</div>

<div class="panel-body canshu-table">

<table>

<tbody>

<tr>

<td>空气温度：</td>

<td>23</td>

</tr>

<tr>

<td>太阳辐照度：</td>

<td>22</td>

</tr>

<tr>

<td>雨量：</td>

<td>22ml</td>

</tr>

<tr>

<td>相对湿度：</td>

<td>3</td>

</tr>

<tr>

<td>电池电压：</td>

<td>12.6</td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

</tr>

</tbody></table>

</div>

</div>

</div>

<div class="tab-pane fade active in" id="signups">

<div class="row">

<div class="col-md-8 col-sm-8 kongzhi-height-left">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">控制面板</h3>

</div>

<div class="panel-body">

<div class="row">

<div id="kongzhiimg" class="col-md-7 row-kongzhimianban kongzhimianban-img">

<img src="images/device/door-left.png" id="door\_left" width="27px" height="35px" />

<img src="images/device/door-right.png" id="door\_right" width="27px" height="35px" />

<div id="demo">

<div id="indemo">

<div id="demo1">

<img src="../live/img/dianji1.png" border="0" />

</div>

<div id="demo2"></div>

</div>

</div>

<script>

var speed=10;

var tab=document.getElementById("demo");

var tab1=document.getElementById("demo1");

var tab2=document.getElementById("demo2");

tab2.innerHTML=tab1.innerHTML;

var fadongjistart=null;

fadongjistart =function (){

if(document.getElementById("bengkongzhi").options[document.getElementById("bengkongzhi").selectedIndex].value=="kai"){

if(tab2.offsetWidth-tab.scrollLeft<=0)

{

tab.scrollLeft-=tab1.offsetWidth

}

else{

tab.scrollLeft++;

}

}

}

function begin(){

if(document.getElementById("bengkongzhi").options[document.getElementById("bengkongzhi").selectedIndex].value=="guan"){

fadongjistart=null;

}

window.setInterval(fadongjistart,speed);

}

function bengfangxiang(){

if(document.getElementById("bengkaiguan").options[document.getElementById("bengkaiguan").selectedIndex].value=="jinshui"){

var str=document.getElementById( "kongzhiimg").className;

document.getElementById( "kongzhiimg").className=str.substring(0,str.length-3);

}else if(document.getElementById("bengkaiguan").options[document.getElementById("bengkaiguan").selectedIndex].value=="chushui"){

document.getElementById( "kongzhiimg").className=document.getElementById( "kongzhiimg").className+"new";

}

}

</script>

</div>

<div class="col-md-5 row-kongzhimianban">

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">进水泵</span>

<select class="form-control input-sm">

<option value="">浅层</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">泵方向</span>

<select id="bengkaiguan" onChange="bengfangxiang()" class="form-control input-sm">

<option value="jinshui">进水</option>

<option value="chushui">出水</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">门状态</span>

<select id="menzhuangtai" onChange="showdoor()" class="form-control input-sm input-sm">

<option value="guan">关</option>

<option value="kai">开</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">泵控制</span>

<select id="bengkongzhi" onChange="begin()" class="form-control input-sm">

<option value="guan">关</option>

<option value="kai">开</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">油电源</span>

<select class="form-control input-sm">

<option value="">开</option>

<option value="">关</option>

</select>

</div>

</div>

<div class="col-md-12">

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">潜水管清洗</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">深水管清洗</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">排空气</button>

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">校准</button>

</div>

</div>

</div>

</div>

</div>

</div>

<div class="col-md-4 col-sm-4 kongzhi-height-right">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">时间间隔参数</h3>

</div>

<div class="panel-body">

<div class="input-group ">

<span class="input-group-addon ">物理参数</span>

<select class="form-control input-sm">

<option value="">20min</option>

<option value="">30min</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon ">营养盐</span>

<select class="form-control input-sm">

<option value="">20min</option>

<option value="">30min</option>

</select>

</div>

<div class="table-responsive ">

<table class="table no-margin">

<thead>

<tr>

<th>#</th>

<th>排水</th>

<th>进水</th>

<th>进水选择</th>

</tr>

</thead>

<tbody>

<tr>

<td>浅层</td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

</tr>

<tr>

<td>深层</td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

<td><input class="form-control form-control-kongzhi" type="text"></td>

</tr>

</tbody>

</table>

</div>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">更新</button>

</div>

</div>

</div>

</div>

<div class="col-md-12">

<div class="panel-body center-align-text">

<span >数据浏览</span>

<select >

<option value="">浅层</option>

<option value="">深层</option>

</select>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新水质探头</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新油探头</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新营养盐标准品</span>

</label>

<label class="checkbox-inline">

<input class="todo-list-cb" type="checkbox">

<span class="todo-list-mark"></span>

<span class="todo-list-desc">更新营养盐样品</span>

</label>

<button type="button" class="btn btn-xs btn-default pilianggengxin" data-original-title="">自动批量更新</button>

</div>

</div>

<div class="col-md-12" >

<table >

<tr>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">水质参数</h3>

</div>

<div class="panel-body">

<table >

<tbody>

<tr>

<td>蓝绿藻：</td>

<td>XXXX</td>

</tr>

<tr>

<td>叶绿素：</td>

<td>XXXX</td>

</tr>

<tr>

<td>电导率：</td>

<td>XXXX</td>

</tr>

<tr>

<td>溶解氧</td>

<td>XXXX</td>

</tr>

<tr>

<td>PH值：</td>

<td>XXXX</td>

</tr>

<tr>

<td>温度：</td>

<td>XXXX</td>

</tr>

<tr>

<td>浊度：</td>

<td>XXXX</td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">油参数</h3>

</div>

<div class="panel-body canshu-table">

<table >

<tbody>

<tr>

<td>原油：</td>

<td>16.9</td>

</tr>

<tr>

<td>成品油：</td>

<td>79.9</td>

</tr>

<tr>

<td>温度：</td>

<td>5.6.C</td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">营养盐标准品</h3>

</div>

<div class="panel-body canshu-table">

<table >

<tbody>

<tr>

<td>硝酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>磷酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>氨：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>硅酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">营养盐样品</h3>

</div>

<div class="panel-body canshu-table">

<table >

<tr>

<td>硝酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>磷酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>氨：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td>硅酸盐：</td>

<td>0.9MG/L</td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<td> </td>

<td> </td>

</tr>

<tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">天气参数</h3>

</div>

<div class="panel-body canshu-table">

<table >

<tbody>

<tr>

<td>空气温度：</td>

<td>23</td>

</tr>

<tr>

<td>太阳辐照度：</td>

<td>22</td>

</tr>

<tr>

<td>雨量：</td>

<td>22ml</td>

</tr>

<tr>

<td>相对湿度：</td>

<td>3</td>

</tr>

<tr>

<td>电池电压：</td>

<td>12.6</td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="">手动更新</button>

</div>

</div>

</div>

</td>

</tr>

</table>

</div>

</div>

</div>

</div>

</fieldset>

</section>

</div>

</div>

</div>

</div>

</div>

</div>

<div class="panel-footer clearfix">

</div>

</div>

</div>

</div>

</div>

</div>

</body>

<script language="javascript">

var dooropenfunction= function (){

if(document.getElementById("door\_left").width>2&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

document.getElementById("door\_left").width-=2;

document.getElementById("door\_right").width-=2;

}else{

dooropenfunction=null;

}

}

var doorclosfunction = function(){

if(document.getElementById("door\_left").width<26&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

document.getElementById("door\_left").width+=2;

document.getElementById("door\_right").width+=2;

}

else{

doorclosfunction=null;

}

}

function showdoor(){

if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

window.setInterval(dooropenfunction, 200);

}else if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

window.setInterval(doorclosfunction, 200);

}

}

</script>

<style>

.kongzhimianban-imgnew{

width:250px;

height:230px;

background-image:url(img/donghua1.jpg);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

.kongzhimianban-img{

width:250px;

height:230px;

background-image:url(${ctx }/images/device/donghua2.jpg);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

.kongzhimianban-img2{

width:250px;

height:230px;

background-image:url(${ctx }/images/device/shexiangtou.png);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

#door\_left{

margin-top: 66px;

margin-left: 146px;

}

#door\_right{

margin-top: 66px;

margin-left:-3px;

float: right;

margin-right: 20px;

}

#demo {

background: #FFF;

overflow:hidden;

border: 0;

width: 20px;

height:8px;

margin-left: 110px;

margin-top: -2px;

}

#indemo {

float: left;

width: 800%;

}

#demo1 {

float: left;

}

#demo2 {

float: left;

}

#demo1 img{

margin-top:-11px;

height:8px;

width:20px;

}

#demo2 img{

margin-top:-4px;

}

</style>

</html>

<%@ page contentType="text/html; charset=UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>

<!DOCTYPE html>

<html >

<c:set var="ctx" value="${pageContext.request.contextPath}"/>

<head>

<meta charset="utf-8" />

<title>${system.systemName }</title>

<meta name="keywords" content="入海污染源" />

</head>

<body ng-app="myApp" ng-controller="customersCtrl">

<%@ include file="../common/top.jsp" %>

<link rel="stylesheet" href="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.css" />

<script type="text/javascript" src="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.js"></script>

<script type="text/javascript" src="${ctx }/resources/angular-select-tree/angular-multi-select-tree-0.1.0.tpl.js"></script>

<script src="${ctx }/manage/js/device/deviceInfo.js"></script>

<div class="main-container" id="main-container">

<script type="text/javascript">

try{ace.settings.check('main-container' , 'fixed')}catch(e){}

</script>

<div class="main-container-inner">

<%@ include file="../common/leftMenu4device.jsp" %>

<div class="main-content" >

<div class="breadcrumbs" id="breadcrumbs">

<script type="text/javascript">

try{ace.settings.check('breadcrumbs' , 'fixed')}catch(e){}

</script>

<ul class="breadcrumb">

<li>

<i class="icon-home home-icon"></i>

<a href="#">${currMenu.pMenuName }</a>

</li>

<li>

<a href="#">${currMenu.cMenuName }</a>

</li>

</ul><!-- .breadcrumb -->

</div>

<div class="col-md-12 col-sm-12 col-sx-12 map-main">

<div class="panel panel-default">

<div class="row">

<div class="col-md-11 col-sm-11 map-right ">

<section class="todo">

<fieldset class="todo-list">

<div class="tab-pane fade active in" id="signups">

<div class="row">

<div class="col-md-8 col-sm-8 kongzhi-height-left">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">潜标</h3>

</div>

<div class="panel-body">

<div class="row">

<div id="kongzhiimg" class="col-md-7 row-kongzhimianban kongzhimianban-img">

</div>

<div class="col-md-5 row-kongzhimianban">

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">电量</span>

<select class="form-control input-sm">

<option value="">正常</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">电压</span>

<select id="bengkaiguan" onChange="bengfangxiang()" class="form-control input-sm">

<option value="chushui">12V</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">急停</span>

<select id="menzhuangtai" onChange="showdoor()" class="form-control input-sm input-sm">

<option value="guan">关</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">姿态</span>

<select id="bengkongzhi" onChange="begin()" class="form-control input-sm">

<option value="guan">正常</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">运行</span>

<select class="form-control input-sm">

<option value="">正常</option>

</select>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</fieldset>

</section>

</div>

</div>

</div>

</div>

<div class="col-md-12 col-sm-12 col-sx-12 map-main">

<div class="panel panel-default">

<div class="row">

</div>

</div>

</div>

<div class="col-md-12 col-sm-12 col-sx-12 map-main">

<div class="panel panel-default">

<div class="row">

<div class="col-md-11 col-sm-11 map-right ">

<section class="todo">

<fieldset class="todo-list">

<div class="tab-pane fade active in" id="signups">

<div class="row">

<div class="col-md-8 col-sm-8 kongzhi-height-left">

<div class="panel panel-default kongzhi-panel-default">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">CTD</h3>

</div>

<div class="panel-body">

<div class="row">

<div id="kongzhiimg" class="col-md-7 row-kongzhimianban kongzhimianban-img2">

</div>

<div class="col-md-5 row-kongzhimianban">

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">电压</span>

<select class="form-control input-sm">

<option value="">12V</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">面板</span>

<select id="bengkaiguan" onChange="bengfangxiang()" class="form-control input-sm">

<option value="chushui">20</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">急停</span>

<select id="menzhuangtai" onChange="showdoor()" class="form-control input-sm input-sm">

<option value="guan">关</option>

</select>

</div>

<div class="input-group ">

<span class="input-group-addon input-group-addon-control ">供电</span>

<select id="bengkongzhi" onChange="begin()" class="form-control input-sm">

<option value="guan">正常</option>

</select>

</div>

<div class="input-group">

<span class="input-group-addon input-group-addon-control">运行</span>

<select class="form-control input-sm">

<option value="">正常</option>

</select>

</div>

</div>

</div>

</div>

</div>

</div>

<div class="col-md-4" >

<table >

<tr>

<td>

<div class="panel panel-default panel-default-table">

<div class="panel-heading clearfix panel-heading-control">

<h3 class="panel-title panel-title-control">CTD</h3>

</div>

<div class="panel-body">

<table>

<tbody>

<tr>

<td>温度：</td>

<td>XXXX</td>

</tr>

<tr>

<td>湿度：</td>

<td>XXXX</td>

</tr>

<tr>

<td>风速</td>

<td>XXXX</td>

</tr>

<tr>

<td>风向：</td>

<td>XXXX</td>

</tr>

<tr>

<td>温度：</td>

<td>XXXX</td>

</tr>

<tr>

<td colspan="2"> 上次更新时间：</td>

</tr>

<tr>

<td colspan="2"> 2014-3-2 12:00:00</td>

</tr>

</tbody>

</table>

<div class="panel-body center-align-text">

<button type="button" class="btn btn-xs btn-default btn-default-control" data-original-title="" disabled>手动更新</button>

</div>

</div>

</div>

</td>

<td>

<div class="panel panel-default panel-default-table">

</div>

</td>

</tr>

</table>

</div>

</div>

</div>

</fieldset>

</section>

</div>

</div>

</div>

</div>

</div>

</div>

</body>

<script language="javascript">

var dooropenfunction= function (){

if(document.getElementById("door\_left").width>2&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

document.getElementById("door\_left").width-=2;

document.getElementById("door\_right").width-=2;

}else{

dooropenfunction=null;

}

}

var doorclosfunction = function(){

if(document.getElementById("door\_left").width<26&&document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

document.getElementById("door\_left").width+=2;

document.getElementById("door\_right").width+=2;

}

else{

doorclosfunction=null;

}

}

function showdoor(){

if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="kai"){

window.setInterval(dooropenfunction, 200);

}else if(document.getElementById("menzhuangtai").options[document.getElementById("menzhuangtai").selectedIndex].value=="guan"){

window.setInterval(doorclosfunction, 200);

}

}

</script>

<style>

.kongzhimianban-imgnew{

width:250px;

height:230px;

background-image:url(img/donghua1.jpg);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

.kongzhimianban-img{

width:250px;

height:230px;

background-image:url(${ctx }/images/device/qianbiao.png);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

.kongzhimianban-img2{

width:250px;

height:230px;

background-image:url(${ctx }/images/device/shexiangtou.png);

background-size: 250px 230px;

margin-left: 10px;

margin-bottom: -5px;

}

#door\_left{

margin-top: 66px;

margin-left: 146px;

}

#door\_right{

margin-top: 66px;

margin-left:-3px;

float: right;

margin-right: 20px;

}

#demo {

background: #FFF;

overflow:hidden;

border: 0;

width: 20px;

height:8px;

margin-left: 110px;

margin-top: -2px;

}

#indemo {

float: left;

width: 800%;

}

#demo1 {

float: left;

}

#demo2 {

float: left;

}

#demo1 img{

margin-top:-11px;

height:8px;

width:20px;

}

#demo2 img{

margin-top:-4px;

}

</style>

</html>

jQuery(**function**($) {

$('.date-picker').datepicker({autoclose:**true**}).next().on(ace.click\_event, **function**(){

$(**this**).prev().focus();

});

});

**var** myApp = angular.module('myApp',['ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection',

'ngDialog','ui.bootstrap','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal,$timeout){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

//开始分页信息

$scope.gridOptions = {

enableRowSelection:**true**,

rowTemplate: "<div ng-dblclick=\"grid.appScope.onDblClick(row)\" ng-repeat=\"(colRenderIndex, col) in colContainer.renderedColumns track by col.colDef.name\" class=\"ui-grid-cell\" ng-class=\"{ 'ui-grid-row-header-cell': col.isRowHeader }\" ui-grid-cell ></div>",

appScopeProvider: {

onDblClick : **function**(row) {

mtype = 2; //2 代表编辑对话框

pub = row.entity;

$scope.showEdit();

}

},

multiSelect:**false**,

modifierKeysToMultiSelect :**false**,

noUnselect:**true**,

enableRowSelection: **true**,

enableRowHeaderSelection:**false**,

headerRowHeight: 50,

paginationPageSizes: [25, 50, 100],

paginationPageSize: 25,

enableColumnResizing:**true**,

enableGridMenu: **true**,

enableSelectAll: **true**,

exporterCsvFilename: '功能区划列表',

exporterCsvLinkElement: angular.element(document.querySelectorAll(".custom-csv-link-location")),

onRegisterApi: **function**(gridApi){

$scope.gridApi = gridApi;

gridApi.selection.on.rowSelectionChanged($scope,**function**(row){

pub = row.entity;

});

}

};

$scope.query=**function**(){

$(".scbtn").attr('disabled',"true");

**var** pData = $scope.u;

$http({

method:'POST',

url:'getgCompanyList.do',

params:pData})

.success(**function**(response){

$(".scbtn").removeAttr("disabled");

row = response.rows;

col = response.cols;

$scope.gridOptions.columnDefs = col;

$scope.gridOptions.data = row;

});

};

$scope.query();

//删除功能

$scope.dele=**function**(){

**var** id = "0";

**if**(pub!=**null**){

id = pub.id;

}

console.log(id);

**if**(id=="0"){

alert("请选择要删除的功能区划");

**return**;

}

**if**(confirm("确定要删除选中的记录吗?")){

$(".scbtn").attr('disabled',"true");

$scope.u = {id:id};

**var** pData = $scope.u;

$http({

method:'POST',

url:'deleCompany.do',

params:pData,

headers: { 'Content-Type': 'application/json; charset=UTF-8'}

})

.success(**function**(response){

$(".scbtn").removeAttr("disabled");

alert(response);

$scope.query();

});

}

};

//点击新增按钮,弹出新增对话框

$scope.addUser = **function**(){

mtype = 1; //1 代表新增对话框

$scope.showEdit();

};

//弹出编辑框

$scope.showEdit = **function**(){

**var** modalInstance = $modal.open({

templateUrl: 'popupTmpl.html',

controller: ModalInstanceCtrl

});

modalInstance.opened.then(**function**(){//模态窗口打开之后执行的函数

console.log('modal is opened');

});

modalInstance.result.then(**function** (result) {

alert(result);

$scope.query();

console.log(result);

}, **function** (reason) {

console.log(reason);//点击空白区域，总会输出backdrop click，点击取消，则会暑促cancel

});

};

//弹出页面打开后的操作

**var** ModalInstanceCtrl = **function** ($scope, $modalInstance,$http) {

$http.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

//得到当前公司的父类树

$scope.showTree = **function**(){

$http({

method:'POST',

url:'showCompanyTree4Pcode.do',

params:sparam,

headers: { 'Content-Type': 'application/json; charset=UTF-8'}

})

.success(**function**(response){

$scope.pcodelist = response;

});

};

$scope.isactivelist = [{

name: '启用',

value: 1

}, {

name: '禁用',

value: 0

}];

**var** sparam = "";

**if**(mtype == 2){

$scope.m = pub;

sparam = {

code:pub.code,

pcode:pub.pcode

};

$scope.showTree();

}**else**{

$scope.m={

isactive:$scope.isactivelist[0].value

};

sparam={

code:"",

pcode:""

};

$scope.showTree();

}

$scope.save = **function**(){

**var** dataItem = "";

angular.forEach($scope.device, **function** ( item ) {

dataItem =item.id;

});

$scope.m.pcode = dataItem;

**var** param = $scope.m;

//如果执行修改操作时

**if**(mtype==2){

$http({

method:'POST',

url:'saveCompanyChange.do',

params:param,

headers: { 'Content-Type': 'application/json; charset=UTF-8'}

})

.success(**function**(response){

$modalInstance.close(response);

});

}

**if**(mtype==1){

$http({

method:'POST',

url:'saveNewCompany.do',

params:param,

headers: { 'Content-Type': 'application/json; charset=UTF-8'}

})

.success(**function**(response){

$modalInstance.close(response);

});

}

**return**;

};

};

});

**var** myApp = angular.module('myApp',['textAngular','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

myApp.controller('customersCtrl',**function**($scope,$http){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

//初始化站点列表

**var** dparam ="";

$http({

method:'POST',

url:'getStationByUser.do',

params:dparam})

.success(**function**(response){

$scope.stationList = response;

$scope.u={

stationId:$scope.stationList[0].id

}

$scope.query();

});

//根据站点查询出当前站点的水质模板

$scope.query=**function**(){

$(".scbtn").attr('disabled',"true");

**var** qparam = $scope.u;

$http({

method:'POST',

url:'getMouldByStationId.do',

params:qparam})

.success(**function**(response){

$(".scbtn").removeAttr("disabled");

$scope.u=response;

});

};

//保存当前站点的水质评价模板

$scope.save=**function**(){

**var** sparam = $scope.u;

$http({

method:'POST',

url:'saveMouldSetting.do',

params:sparam})

.success(**function**(response){

alert(response);

});

};

});**var** myApp = angular.module('myApp',['ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection'

,'ngDialog','ui.bootstrap','ui.grid.edit','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

$scope.u={};

//$scope.title = "";

//查询条件初始化

**var** stationParam = "";

$http({

method:'POST',

url:'dataChange\_info.do',

params:stationParam})

.success(**function**(response){

$scope.u=response;

});

//开始分页信息

$scope.gridOptions = {

enableRowSelection:**true**,

rowTemplate: "<div ng-dblclick=\"grid.appScope.onDblClick(row)\" ng-repeat=\"(colRenderIndex, col) in colContainer.renderedColumns track by col.colDef.name\" class=\"ui-grid-cell\" ng-class=\"{ 'ui-grid-row-header-cell': col.isRowHeader }\" ui-grid-cell ></div>",

appScopeProvider: {

onDblClick : **function**(row) {

}

},

multiSelect:**false**,

modifierKeysToMultiSelect :**false**,

noUnselect:**true**,

enableRowSelection: **true**,

enableRowHeaderSelection:**false**,

headerRowHeight: 50,

paginationPageSizes: [25, 50, 100],

paginationPageSize: 25,

enableColumnResizing:**true**,

enableGridMenu: **true**,

enableSelectAll: **true**,

exporterCsvFilename: '综合查询',

exporterCsvLinkElement: angular.element(document.querySelectorAll(".custom-csv-link-location")),

onRegisterApi: **function**(gridApi){

$scope.gridApi = gridApi;

gridApi.edit.on.afterCellEdit($scope,**function**(rowEntity, colDef, newValue, oldValue){

$scope.save(rowEntity, newValue, oldValue);

});

}

};

$scope.save = **function**(rowEntity,newValue,oldValue){

**var** collect\_time = rowEntity.collect\_time;

**if**(collect\_time==""){

alert("请填写采集时间");

**return** ;

}

**if**(confirm("是否确认修改")){

**var** param={

id:rowEntity.id,

stationId:$scope.u.stationId,

deviceId:$scope.u.deviceId,

indicatorCode:$scope.u.indicatorCode,

collect\_time:rowEntity.collect\_time,

newData:newValue,

oldData:oldValue

};

$http({

method:'POST',

url:'saveChangeData.do',

params:param})

.success(**function**(response){

alert(response);

console.log(response);

});

}

};

$scope.addRow = **function**(){

$scope.gridOptions.data.unshift({

id:0,

collect\_time:'',

data:0

});

};

$scope.query = **function**(){

**var** endDate = $('#endDate').val();

**var** startDate = $('#startDate').val();

**if**(ifBeginLaterEnd(startDate,endDate)){

$(".scbtn").removeAttr("disabled");

**return**;

};

**if**($scope.u.indicatorIds==**null**||$scope.u.indicatorIds==""){

alert("请选择参数");

**return** ;

}

**var** indicatorIds = "0#0";

angular.forEach($scope.u.indicatorIds, **function** ( item ) {

indicatorIds = indicatorIds +","+item.id;

});

**var** queryParam = {

stationId:$scope.u.stationId,

indicatorIds:indicatorIds,

beginDate:startDate,

endDate:endDate

};

console.log(queryParam);

$(".scbtn").attr('disabled',"true");

$http({

method:'POST',

url:'showDataChangeInfo.do',

params:queryParam})

.success(**function**(response){

$scope.u.stationId = response.stationId;

$scope.u.deviceId = response.device.id;

$scope.u.indicatorCode = response.indicator.code;

$(".scbtn").removeAttr("disabled");

row = response.page.rows;

col = response.page.cols;

$scope.gridOptions.columnDefs = col;

$scope.gridOptions.data = row;

});

};

setTimeout(**function** (){

$scope.query();

}, 1000);

});

**var** myApp = angular.module('myApp',['ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection'

,'ngDialog','ui.bootstrap','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

$scope.u={};

//$scope.title = "";

//查询条件初始化

**var** stationParam = "";

$http({

method:'POST',

url:'dataChange\_info.do',

params:stationParam})

.success(**function**(response){

$scope.u=response;

});

//开始分页信息

$scope.gridOptions = {

enableRowSelection:**true**,

rowTemplate: "<div ng-dblclick=\"grid.appScope.onDblClick(row)\" ng-repeat=\"(colRenderIndex, col) in colContainer.renderedColumns track by col.colDef.name\" class=\"ui-grid-cell\" ng-class=\"{ 'ui-grid-row-header-cell': col.isRowHeader }\" ui-grid-cell ></div>",

appScopeProvider: {

onDblClick : **function**(row) {

}

},

multiSelect:**false**,

modifierKeysToMultiSelect :**false**,

noUnselect:**true**,

enableRowSelection: **true**,

enableRowHeaderSelection:**false**,

headerRowHeight: 50,

paginationPageSizes: [25, 50, 100],

paginationPageSize: 25,

enableColumnResizing:**true**,

enableGridMenu: **true**,

enableSelectAll: **true**,

exporterCsvFilename: '综合查询',

exporterCsvLinkElement: angular.element(document.querySelectorAll(".custom-csv-link-location")),

};

$scope.query = **function**(){

**var** endDate = $('#endDate').val();

**var** startDate = $('#startDate').val();

**if**(ifBeginLaterEnd(startDate,endDate)){

$(".scbtn").removeAttr("disabled");

**return**;

};

**var** indicatorIds = "";

angular.forEach($scope.u.indicatorIds, **function** ( item ) {

indicatorIds = item.id;

});

**var** queryParam = {

stationId:$scope.u.stationId,

indicatorIds:indicatorIds,

beginDate:startDate,

endDate:endDate

};

$(".scbtn").attr('disabled',"true");

$http({

method:'POST',

url:'showdataChangeLog.do',

params:queryParam})

.success(**function**(response){

$(".scbtn").removeAttr("disabled");

row = response.page.rows;

col = response.page.cols;

$scope.gridOptions.columnDefs = col;

$scope.gridOptions.data = row;

});

};

setTimeout(**function** (){

$scope.query();

}, 1000);

});

**var** myApp = angular.module('myApp',['ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection'

,'ngDialog','ui.bootstrap','ui.grid.edit','multi-select-tree','ui.grid.importer']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

$scope.u={};

$scope.data = [];

//开始分页信息

$scope.gridOptions = {

enableRowSelection:**true**,

rowTemplate: "<div ng-dblclick=\"grid.appScope.onDblClick(row)\" ng-repeat=\"(colRenderIndex, col) in colContainer.renderedColumns track by col.colDef.name\" class=\"ui-grid-cell\" ng-class=\"{ 'ui-grid-row-header-cell': col.isRowHeader }\" ui-grid-cell ></div>",

appScopeProvider: {

onDblClick : **function**(row) {

}

},

multiSelect:**false**,

modifierKeysToMultiSelect :**false**,

noUnselect:**true**,

enableRowSelection: **true**,

enableRowHeaderSelection:**false**,

headerRowHeight: 50,

paginationPageSizes: [25, 50, 100],

paginationPageSize: 25,

enableColumnResizing:**true**,

enableGridMenu: **true**,

enableSelectAll: **true**,

exporterCsvFilename: '综合查询',

exporterCsvLinkElement: angular.element(document.querySelectorAll(".custom-csv-link-location")),

data: 'data',

importerDataAddCallback: **function** ( grid, newObjects ) {

$scope.data = $scope.data.concat( newObjects );

},

onRegisterApi: **function**(gridApi){

$scope.gridApi = gridApi;

gridApi.edit.on.afterCellEdit($scope,**function**(rowEntity, colDef, newValue, oldValue){

});

}

};

$scope.clear=**function**(){

$scope.data=[];

$scope.gridOptions.columnDefs = [];

$scope.gridOptions.data = [];

};

**var** needs=[{name:"collect\_time",type:"string"},

{name:"stationId",type:"number"},

{name:"deviceId",type:"number"},

{name:"indicatorCode",type:"string"},

{name:"data",type:"number"}];

//保存表单信息

$scope.save = **function**(){

**var** ifright = 1;

**var** columnDefs = $scope.gridOptions.columnDefs;

angular.forEach(needs,**function**(need){

**var** ifhave = 0;

angular.forEach(columnDefs,**function**(col){

**if**(col.name==need.name){

ifhave = 1;

}

});

**if**(ifhave==0){

alert("缺少"+need.name+"元素");

ifright = 0;

**return**;

}

});

**if**(ifright==0){

**return**;

}

**var** param = {

importString:$scope.data

};

$http({

method:'POST',

url:'saveImportData.do',

params:param})

.success(**function**(response){

console.log(response);

alert(response);

});

};

$scope.csvdown = **function**(){

console.log("333333333333");

**var** param = {

};

$http({

method:'POST',

url:'doc/datamodel.csv',

params:param})

.success(**function**(response){

});

};

});

**var** myApp = angular.module('myApp',['ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection'

,'ngDialog','ui.bootstrap','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal,i18nService){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

$scope.checkB={yunwei:**true**,yuzhi:**true**,alarm:**true**};

//$scope.title = "";

//查询条件初始化

**var** stationParam = "";

$http({

method:'POST',

url:'dataquery\_info.do',

params:stationParam})

.success(**function**(response){

$scope.u = response;

//根据获得的STATION,查询出当前站点包含的设备列表

**var** deviceParam = {id:$scope.u.stationId};

$http({

method:'POST',

url:'getDevices4Station4Show.do',

params:deviceParam})

.success(**function**(response){

$scope.deviceList = response;

$scope.u.deviceId = $scope.deviceList[0].id;

$scope.query();

});

});

//开始分页信息

$scope.gridOptions = {

enableRowSelection:**true**,

rowTemplate: "<div ng-dblclick=\"grid.appScope.onDblClick(row)\" ng-repeat=\"(colRenderIndex, col) in colContainer.renderedColumns track by col.colDef.name\" class=\"ui-grid-cell\" ng-class=\"{ 'ui-grid-row-header-cell': col.isRowHeader }\" ui-grid-cell ></div>",

appScopeProvider: {

onDblClick : **function**(row) {

mtype = 2; //2 代表编辑对话框

station = row.entity;

$scope.showEdit();

}

},

/\*columnDefs: [

{ displayName: 'name',field:"collect\_time",isible:true, width: "\*" },

{ displayName: 'ph',field:"pH",isible:true, width: "\*", enableCellEdit: false,

cellClass: function(grid, row, col, rowRenderIndex, colRenderIndex) {

if (grid.getCellValue(row,col) >= 9) {

return 'blue';

} else if (grid.getCellValue(row,col) < 9) {

return 'pink';

}

}

}

],\*/

multiSelect:**false**,

modifierKeysToMultiSelect :**false**,

noUnselect:**true**,

enableRowSelection: **true**,

enableRowHeaderSelection:**false**,

headerRowHeight: 50,

paginationPageSizes: [25, 50, 100],

paginationPageSize: 25,

enableColumnResizing:**true**,

enableGridMenu: **true**,

enableSelectAll: **true**,

exporterCsvFilename: '实时数据',

exporterCsvLinkElement: angular.element(document.querySelectorAll(".custom-csv-link-location")),

onRegisterApi: **function**(gridApi){

$scope.gridApi = gridApi;

gridApi.selection.on.rowSelectionChanged($scope,**function**(row){

firstpage = row.entity;

});

}

};

//查询数据

$scope.query = **function**(){

**var** endDate = $('#endDate').val();

**var** startDate = $('#startDate').val();

**if**(ifBeginLaterEnd(startDate,endDate)){

$(".scbtn").removeAttr("disabled");

**return**;

};

$(".scbtn").attr('disabled',"true");

$scope.u.beginDate = startDate;

$scope.u.endDate = endDate;

//获得复选框的值

**var** queryType = "";

**if**($scope.checkB.yunwei){

queryType += "1";

}**else**{

queryType += "0";

}

**if**($scope.checkB.yuzhi){

queryType += "1";

}**else**{

queryType += "0";

}

**if**($scope.checkB.alarm){

queryType += "1";

}**else**{

queryType += "0";

}

$scope.u.queryType = queryType;

**var** queryParam = $scope.u;

$http({

method:'POST',

url:'showDataQueryInfo.do',

params:queryParam})

.success(**function**(response){

$(".scbtn").removeAttr("disabled");

row = response.rows;

col = response.cols;

angular.forEach(col, **function** ( item ) {

**if**(item.field!="collect\_time"){ //排除第一排的时间参数

item.cellClass = **new** Function('grid','row','col','rowRenderIndex','colRenderIndex',item.cellClassName);

}

});

$scope.gridOptions.columnDefs = col;

$scope.gridOptions.data = row;

});

};

});

**var** myApp = angular.module('myApp',['highcharts-ng']);

myApp.controller('customersCtrl',**function**($scope,$http,$timeout){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

detail();

**var** updateClock = **function**(){

$timeout(**function**(){

updateClock();

detail();

$scope.showStand();

},600000);

};

updateClock();

**function** gotoWarnAlarm(type) {

location.href = "${ctx}/warn.do?type=" + type;

}

$(**function**() {

//初始化菜单树高度

//初始化站点

(**function**() {

})();

});

**function** detail() {

**var** airContent = **new** Array();

**var** firContent = **new** Array();

**var** warnContent = **new** Array();

**var** sysContent = **new** Array();

**var** polluContent = **new** Array();

**var** mainContent = **new** Array();

**var** standard = '';

getJSON(

'getlastdataNow.do?',

{},

**false**,

**function**(res) {

//开始编写实时数据部分

**var** lastData = res.Datas;

**for**(**var** p **in** lastData){

/\* airContent.push('<div class="widget-header " >'); \*/

**var** lastTime = lastData[p].lastTime;

**var** pointNum = lastData[p].pointNum;

**if**(lastTime == **null**){

lastTime = ' ';

}

airContent.push('<div class="col-xs-12 col-sm-6 ">');

airContent.push('<div class="widget-box">');

airContent.push('<div class="widget-header header-color-yle">');

airContent.push('<h5 class="bigger lighter">');

airContent.push('<i class="icon-table"></i>');

airContent.push(lastData[p].deviceName+' 更新时间：'+lastTime);

airContent.push('</h5>');

airContent.push('</div>');

airContent.push('<table class="table table-striped table-bordered table-hover">');

airContent.push('<tbody>');

**var** rdata = lastData[p].MetaDatas;

console.log(rdata);

**var** n = 0;

**for**(**var** m **in** rdata){

**if**(m%2==0){

airContent.push('<tr>');

**if**(rdata[m].unitName.length<1){

**if**(rdata[m].mdata<=0||rdata[m].mdata==401.21){

airContent.push('<td style="width: 50%">'+rdata[m].indicatorTitle+':<span >-</span></td>');

}**else**{

airContent.push('<td style="width: 50%">'+rdata[m].indicatorTitle+':<span >'

+(rdata[m].mdata).toFixed(pointNum)+'</span></td>');

}

}**else**{

**if**(rdata[m].mdata<=0||rdata[m].mdata==401.21){

airContent.push('<td style="width: 50%">'+rdata[m].indicatorTitle+':<span >-</span>('+rdata[m].unitName+')</td>');

}**else**{

airContent.push('<td style="width: 50%">'+rdata[m].indicatorTitle+':<span >'

+(rdata[m].mdata).toFixed(pointNum)+'</span>('+rdata[m].unitName+')</td>');

}

}

}**else**{

**if**(rdata[m].unitName.length<1){

**if**(rdata[m].mdata<=0||rdata[m].mdata==401.21){

airContent.push('<td >'+rdata[m].indicatorTitle+':<span >-</span></td>');

}**else**{

airContent.push('<td >'+rdata[m].indicatorTitle+':<span >'

+(rdata[m].mdata).toFixed(pointNum)+'</span></td>');

}

}**else**{

**if**(rdata[m].mdata<=0||rdata[m].mdata==401.21){

airContent.push('<td >'+rdata[m].indicatorTitle+':<span ">-</span>('+rdata[m].unitName+')</td>');

}**else**{

airContent.push('<td >'+rdata[m].indicatorTitle+':<span ">'

+(rdata[m].mdata).toFixed(pointNum)+'</span>('+rdata[m].unitName+')</td>');

}

}

airContent.push('</tr>');

}

n = m;

}

**if**(n>0&&n%2==0){

airContent.push('<td ><span style="font-size:15px;"></span></td>');

airContent.push('</tr>');

}

airContent.push('</tbody>');

airContent.push('</table>');

airContent.push('</div>');

airContent.push('</div>');

}

});

$("#datashow").html(airContent.join(""));

$("#warnshow").html(warnContent.join(""));

$("#standshow").html(firContent.join(""));

$("#systemshow").html(sysContent.join(""));

$("#pollushow").html(polluContent.join(""));

$("#mainshow").html(mainContent.join(""));

}

jQuery(**function**($) {

$('#simple-colorpicker-1').ace\_colorpicker({pull\_right:**true**}).on('change', **function**(){

**var** color\_class = $(**this**).find('option:selected').data('class');

**var** new\_class = 'widget-header';

**if**(color\_class != 'default') new\_class += ' header-color-'+color\_class;

$(**this**).closest('.widget-header').attr('class', new\_class);

});

// scrollables

$('.slim-scroll').each(**function** () {

**var** $this = $(**this**);

$this.slimScroll({

height: $this.data('height') || 100,

railVisible:**true**

});

});

/\*\*$('.widget-box').on('ace.widget.settings' , function(e) {

e.preventDefault();

});\*/

// Portlets (boxes)

$('.widget-container-span').sortable({

connectWith: '.widget-container-span',

items:'> .widget-box',

opacity:0.8,

revert:**true**,

forceHelperSize:**true**,

placeholder: 'widget-placeholder',

forcePlaceholderSize:**true**,

tolerance:'pointer'

});

});

});

**function** loadChart(data){

**var** step = 1;

**var** size = data.xtimes.length;

**if**(size>7){

step = Math.ceil(size/7);

}

$("#container").html("");

$('<div class="chart" class="span10" style="height: 308px; margin: 0 auto">>')

.appendTo('#container')

.highcharts({

//设置标题

title: {

text: '',

align: 'right',

x: -80,

verticalAlign: 'bottom',

y: -80

},

//去掉版权

credits:{

enabled:**false**

},

options: {

chart: {

type: 'line',

zoomType: 'x'

}

},

//当鼠标滑过时显示的文字,没有实现

tooltip: {

animation:**true**,

formatter: **function** () {

**var** result = "";

**if**(**this**.y==1){

result = "一类";

}**else** **if**(**this**.y==2){

result = "二类";

}**else** **if**(**this**.y==3){

result = "三类";

}**else** **if**(**this**.y==4){

result = "四类";

}**else** **if**(**this**.y==5){

result = "五类";

}**else** **if**(**this**.y==6){

result = "劣五类";

}

**return** '' + **this**.x +

'为<b>' + result + '</b>';

}

},

series: [{

data: data.ydatas,

//隐藏底部的series1

showInLegend: **false**

}],

xAxis: {

categories:data.xtimes,

labels: {

step:step

}

},

yAxis:{

max:6,

//设置纵坐标标题

title:{

enabled:**false**

},

labels:{

formatter: **function** () {

**var** result = "";

**if**(**this**.value==1){

result = "一类";

}**else** **if**(**this**.value==2){

result = "二类";

}**else** **if**(**this**.value==3){

result = "三类";

}**else** **if**(**this**.value==4){

result = "四类";

}**else** **if**(**this**.value==5){

result = "五类";

}**else** **if**(**this**.value==6){

result = "劣五类";

}

**return** result;

}

}

}

});

};

**var** myApp = angular.module('myApp',['ngDialog','ui.bootstrap','multi-select-tree']);

**var** row;

**var** col;

**var** pub;

**var** mtype;

**var** station;

**var** myyChart;

**var** myyChart2;

**var** myyChart3;

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$modal){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

//$scope.title = "";

//查询条件初始化

**var** stationParam = "";

$http({

method:'POST',

url:'graphquery\_init.do',

params:stationParam})

.success(**function**(response){

$scope.u=response;

});

$scope.query = **function**(){

**var** endDate = $('#endDate').val();

**var** startDate = $('#startDate').val();

**if**(ifBeginLaterEnd(startDate,endDate)){

$(".btn-primary").removeAttr("disabled");

**return**;

};

$scope.u.beginDate = startDate;

$scope.u.endDate = endDate;

**if**($scope.u.indicatorIds==**null**||$scope.u.indicatorIds==""){

alert("请选择参数");

**return** ;

}

**var** indicatorIds = "0#0";

**var** i = 0;

angular.forEach($scope.u.indicatorIds, **function** ( item ) {

indicatorIds = indicatorIds +","+item.id;

i = i + 1;

});

**if**(i>3){

alert("最多支持三个参数同时比较,请重试");

**return** ;

}

$(".btn-primary").attr('disabled',"true");

**var** queryParam = {

indicatorIds:indicatorIds,

beginDate:$scope.u.beginDate,

endDate:$scope.u.endDate

};

$http({

method:'POST',

url:'graphShow4echarts.do',

params:queryParam})

.success(**function**(response){

$(".btn-primary").removeAttr("disabled");

showData(response);

});

};

setTimeout(**function** (){

$scope.query();

}, 1000);

});

**function** showData(res){

**var** yaxs = [];

**var** yax = 0;

**var** series = **new** Array();

$.each(res, **function** (i, dataset) {

**var** result = dataset.result;

**var** dataList = dataset.data;

**var** list = dataList;

**var** servie = {

name: result.title,

type: 'spline',

yAxis: yax,

tooltip: {

valueSuffix: ' '+result.unitName

},

unit:result.unitName,

cropThreshold:100000,//设置能够缩放的

// Define the data points. All series have a dummy year

// of 1970/71 in order to be compared on the same x axis. Note

// that in JavaScript, months start at 0 for January, 1 for February etc.

data:list

};

**var** yaxi = { // Primary yAxis

labels: {

format: '{value}'+result.unitName,

style: {

color: Highcharts.getOptions().colors[yax]

}

},

title: {

text: result.title,

style: {

color: Highcharts.getOptions().colors[yax]

}

}

};

**if**(yax>0){

yaxi.opposite = **true**;//设置纵坐标的位置在右边

}

yax = yax + 1;

yaxs.push(yaxi);

series.push(servie);

});

**var** chart = Highcharts.chart('container', {

plotOptions:{

series:{

turboThreshold:1000000//set it to a larger threshold, it is by default to 1000

},

spline: {

marker: {

enabled: **true**

}

}

},

chart: {

/\*type: 'area',

zoomType: 'x',

panKey: 'shift'\*/

type: 'spline',

zoomType: 'xy', //鼠标缩放方式

panning: **true**,

/\*marginLeft: 40, // Keep all charts left aligned\*/

spacingTop: 20,

spacingBottom: 20,

backgroundColor: '#ffffff'

},

title: {

text: '综合趋势参数对比',

align: 'center',

margin: 0,

x: 30

},

credits: { //版权信息

enabled: **false**

},

legend: { //图例说明

layout: 'vertical',

align: 'left',

x: 80,

verticalAlign: 'top',

y: 55,

floating: **true**,

backgroundColor: (Highcharts.theme && Highcharts.theme.legendBackgroundColor) || '#FFFFFF'

},

xAxis: {

type: 'datetime',

dateTimeLabelFormats:{

second: '%m-%d %H:%M:%S',

minute: '%m-%d %H:%M',

hour: '%m-%d %H',

day: '20%y-%m-%d',

week: '%e. %m',

month: '20%y-%m',

year: '20%Y'

},

crosshair: **true**,

title: {

text: **null**

}

},

yAxis:yaxs,

tooltip: {

shared: **true**,

headerFormat: '<b>{point.x:20%y-%m-%d %H:%M}</b><br>',

pointFormatter:**function**(){

**return** '<span style="color: '+ **this**.series.color + '">\u25CF</span> '+

**this**.series.name+': <b>'+ **this**.y +'</b>'+**this**.series.userOptions.unit+'<br/>'

}

},

series: series

});

}

**var** myApp = angular.module('myApp',['highcharts-ng','ui.grid', 'ui.grid.exporter','ui.grid.pagination',

'ui.grid.resizeColumns','ui.grid.selection',

'ngDialog','ui.bootstrap','multi-select-tree']);

myApp.controller('customersCtrl',**function**($scope,$http,ngDialog,$timeout,$modal){

//添加一下内容防止输出乱码

$http.defaults.headers.post['Content-Type'] = 'application/json;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

//初始化统计口径列表

$scope.activelist = [{

name: '按月统计',

value: 1

}, {

name: '按周统计',

value: 2

}, {

name: '按日统计',

value: 3

}];

$scope.collectType = 3;

$scope.queryData = **function**(){

$(".scbtn").attr('disabled',"true");

**var** endDate = $('#endDate').val();

**var** startDate = $('#startDate').val();

**if**(ifBeginLaterEnd(startDate,endDate)){

$(".scbtn").removeAttr("disabled");

**return**;

};

**var** collectType = $scope.collectType;

**var** stationId = $('#stationId').val();

**var** zparams={

endDate:endDate,

beginDate:startDate,

statType:collectType,

};

$http({

method:'POST',

url:'showStat.do',

params:zparams})

.success(**function**(res){

$(".scbtn").removeAttr("disabled");

**var** headerHeight=$("#header").outerHeight();

**var** windowHeight=$(window).outerHeight();

$("#oneCharts").css({ height: (windowHeight-headerHeight\*2)/2});

$("#lineCharts").css({ height: (windowHeight-headerHeight\*2)/2});

loadCharts(res);

$scope.createTable(res,'queryContainer','','');

});

};

$scope.queryData();

//编辑表格信息

$scope.createTable = **function**(result,id,title,flag){

resultData = result;

**var** dtime = "天数";

//console.debug(resultData.tableContents);

**if**(result.statType=='1'){

dtime = "月份数";

}**else** **if**(result.statType=='2'){

}

**var** htmlStr = **new** Array();

htmlStr.push('<h4 style="text-align: center;"><strong>水质等级统计</strong></h4>');

htmlStr.push('<table class="table table-striped" style="width: 100%">');

htmlStr.push("<thead>");

htmlStr.push("<tr class='info'>");

htmlStr.push("<th>水质等级</th>");

htmlStr.push("<th>天数</th>");

htmlStr.push("<th>首要污染因子</th>");

htmlStr.push("<th>占比</th>");

htmlStr.push("</tr>");

htmlStr.push("</thead>");

**for**(**var** i **in** result.datas){

htmlStr.push("<tr>");

htmlStr.push("<td>"+result.datas[i].name+"</td>");

htmlStr.push("<td>"+result.datas[i].y+"</td>");

htmlStr.push("<td>"+result.datas[i].firstThing+"</td>");

htmlStr.push("<td>"+Math.round(100\*result.datas[i].y/result.datas[i].statcount)+"%"+"</td>");

htmlStr.push("</tr>");

}

htmlStr.push("</table>");

$("#"+id).html(htmlStr.join(""));

};

$scope.exportData=**function**(){

**var** modalInstance = $modal.open({

templateUrl: 'popupTmpl.html',

controller: ModalInstanceCtrl

});

modalInstance.opened.then(**function**(){//模态窗口打开之后执行的函数

console.log('modal is opened');

});

modalInstance.result.then(**function** (result) {

}, **function** (reason) {

console.log(reason);//点击空白区域，总会输出backdrop click，点击取消，则会暑促cancel

});

};

//弹出页面打开后的操作

**var** ModalInstanceCtrl = **function** ($scope, $modalInstance,$http) {

$http.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded;charset=utf-8';

$http.defaults.headers.post['Accept'] = 'application/json, text/javascript, \*/\*; q=0.01';

$http.defaults.headers.post['X-Requested-With'] = 'XMLHttpRequest';

$scope.print = **function**(){

$("div#printArea").printArea();

};

};

});

**function** excelData(){

//通过开始时间结束时间导出EXCEL

**var** startDate = $('#startDate').val();

**var** endDate = $('#endDate').val();

**var** url='${ctx}/exportDataStand.do'+'?startDate='+startDate+'&endDate='+endDate;

window.open(url, "下载Excel", "height=100,width=400,top=0,left=0,toolbar=no,menubar=no,scrollbars=no, resizable=no,location=no, status=no");

};

**function** loadCharts(data){

**var** piedata = data.datas;

console.log(piedata);

**var** step = 1;

**var** size = data.xtimes.length;

**if**(size>10){

step = Math.ceil(size/10);

}

$("#piecontainer").html("");

//生成饼状图

$('<div class="chart">')

.appendTo('#piecontainer')

.highcharts({

//设置标题

title: {

text: '',

align: 'right',

x: -80,

verticalAlign: 'bottom',

y: -80

},

//去掉版权

credits:{

enabled:**false**

},

options: {

chart: {

type: 'line',

zoomType: 'x'

}

},

//当鼠标滑过时显示的文字,没有实现

tooltip: {

animation:**true**,

formatter: **function** () {

**return** **this**.key +'的次数为'+

'<b>' + **this**.y + '</b>';

/\*return '该水质等级的天数为'+this.y;\*/

}

},

series: [{

data: piedata,

type:'pie',

//隐藏底部的series1

showInLegend: **false**

}]

});

//生成折线图

$("#linecontainer").html("");

$('<div class="chart" >')

.appendTo('#linecontainer')

.highcharts({

chart: {

//这里设置缩放

zoomType: 'x'

},

//设置标题

title: {

text: '',

align: 'left',

margin: 0,

x: 30

},

//去掉版权

credits: {

enabled: **false**

},

options: {

chart: {

type: 'line',

zoomType: 'x'

}

},

xAxis: {

categories:data.xtimes,

labels: {

step:step,

staggerLines: 1,

format: '{value}'

}

},

yAxis:{

max:6,

//设置纵坐标标题

title:{

enabled:**false**

},

plotLines:[{

color:'red', //线的颜色，定义为红色

dashStyle:'solid', //默认值，这里定义为实线

value:4, //定义在那个值上显示标示线，这里是在x轴上刻度为3的值处垂直化一条线

width:2 //标示线的宽度，2px

}],

labels:{

formatter: **function** () {

**var** result = "";

**if**(**this**.value==1){

result = "一类";

}**else** **if**(**this**.value==2){

result = "二类";

}**else** **if**(**this**.value==3){

result = "三类";

}**else** **if**(**this**.value==4){

result = "四类";

}**else** **if**(**this**.value==5){

result = "五类";

}**else** **if**(**this**.value==6){

result = "劣五类";

}

**return** result;

}

}

},

//当鼠标滑过时显示的文字,没有实现

tooltip: {

formatter: **function** () {

**var** result = "";

**if**(**this**.y==1){

result = "一类";

}**else** **if**(**this**.y==2){

result = "二类";

}**else** **if**(**this**.y==3){

result = "三类";

}**else** **if**(**this**.y==4){

result = "四类";

}**else** **if**(**this**.y==5){

result = "五类";

}**else** **if**(**this**.y==6){

result = "劣五类";

}

**return** '' + **this**.x +

'为<b>' + result + '</b><br/>首要污染因子为<b>'+**this**.key+'</b>';

},

/\*positioner: function () {

return {

x: this.chart.chartWidth - this.label.width, // right aligned

y: -1 // align to title

};

},\*/

borderWidth: 0,

backgroundColor: 'none',

pointFormat: '{point.y}',

headerFormat: '',

shadow: **false**,

style: {

fontSize: '15px'

},

},

series: [{

data: data.ydatas,

type:'line',

//隐藏底部的series1

showInLegend: **false**,

fillOpacity: 0.3,

tooltip: {

valueSuffix: ' '

}

}],

exporting: {

filename: '水质等级走势'

}

});

};