package com.bizlogic.qms.action;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.text.DecimalFormat;

import org.apache.poi.hssf.usermodel.HSSFCell;

import org.apache.poi.hssf.usermodel.HSSFRow;

import org.apache.poi.hssf.usermodel.HSSFSheet;

import org.apache.poi.hssf.usermodel.HSSFWorkbook;

import org.apache.poi.poifs.filesystem.POIFSFileSystem;

import com.bizlogic.base.front.action.BaseUserEditAction;

import com.bizlogic.common.Element;

public class AnhuiliFamilyUserImportSaveAction extends BaseUserEditAction{

protected Element createElement() {

return new Element();

}

protected Class getElementClass() {

return Element.class;

}

@Override

protected boolean prepareSave(Element element) {

logger.info("用户开始:", "start ++++++++++++++++++++++++++++++++++++++++++++++++++==");

DecimalFormat df = new DecimalFormat("#");

String dropSql = "drop table t\_anhuili\_family\_user";

try{

super.getDefaultManager().executeSQL(dropSql);

}catch(Exception e){}

String createSql ="create table t\_anhuili\_family\_user("

+" id VARCHAR2(100) PRIMARY KEY,"

+" benhuzhuzhi VARCHAR2(100),"

+" yuhuzhuguanxi VARCHAR2(100),"

+" xingming VARCHAR2(100),"

+" shifouhuzhu VARCHAR2(100),"

+" xingbie VARCHAR2(100),"

+" chushengriqi VARCHAR2(100),"

+" hukouxingzhi VARCHAR2(100),"

+" shenfenzhenghaoma VARCHAR2(100),"

+" wenhuachengdu VARCHAR2(100),"

+" lanxidianhua VARCHAR2(100),"

+" shifourenhufenli VARCHAR2(100)"

+" )";

super.getDefaultManager().executeSQL(createSql);

for(int j=1;j<=56;j++){

if(j<=28){

File file = new File("C:/Users/Administrator/Desktop/安慧里社区表1/"+j+".xls");

// 构造 XSSFWorkbook 对象，strPath 传入文件路径

POIFSFileSystem fs = null;

try {

fs = new POIFSFileSystem(new FileInputStream(file));

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

// 创建工作簿

HSSFWorkbook workBook = null;

try {

workBook = new HSSFWorkbook(fs);

} catch (IOException e) {

e.printStackTrace();

}

HSSFSheet sheet0 = workBook.getSheetAt(0);

HSSFRow row = null;

HSSFCell cell = null;

for (int i = 15; i < sheet0.getPhysicalNumberOfRows(); i++)

{

String benhuzhuzhi ="";

String yuhuzhuguanxi = "";

String xingming = "";

String shifouhuzhu = "";

String xingbie = "";

String chushengriqi="";

String hukouxingzhi="";

String shenfenzhenghaoma = "";

String wenhuachengdu = "";

row = sheet0.getRow(i);

//本户住址

cell = row.getCell(1);

if(cell != null){

try{

benhuzhuzhi = df.format(cell.getNumericCellValue());

}catch(Exception e){

benhuzhuzhi = cell.getStringCellValue();

}

if("NULL".equals(benhuzhuzhi))

benhuzhuzhi = null;

}

//与户主关系

cell = row.getCell(2);

if(cell != null){

try{

yuhuzhuguanxi = df.format(cell.getNumericCellValue());

}catch(Exception e){

yuhuzhuguanxi = cell.getStringCellValue().trim();

}

if("NULL".equals(yuhuzhuguanxi))

yuhuzhuguanxi = null;

}

int z = yuhuzhuguanxi.indexOf("户主");

if(z >= 0){

shifouhuzhu = "1";

}else{

shifouhuzhu = "0";

}

//姓名

cell = row.getCell(3);

if(cell != null){

xingming = cell.getStringCellValue().trim();

if("NULL".equals(xingming))

xingming = null;

}

if("0".equals(benhuzhuzhi) && "0".equals(yuhuzhuguanxi))

break;

//性别

cell = row.getCell(4);

if(cell != null){

xingbie = cell.getStringCellValue();

if(xingbie.equals("NULL"))

xingbie = "";

else

{

if("男".equals(xingbie))

xingbie="0";

if("女".equals(xingbie))

xingbie="1";

}

}else{

//没有设置性别的设置成 2

xingbie="2";

}

//出生日期

cell = row.getCell(5);

if(cell != null){

try{

chushengriqi = cell.getStringCellValue();

try{

if(!"NULL".equals(chushengriqi)){

chushengriqi = chushengriqi.substring(0, chushengriqi.length());

}

}catch(Exception ee){

chushengriqi = "1111-11-11";

}

}catch(Exception e){

chushengriqi = cell.getDateCellValue().toString();

try{

chushengriqi = chushengriqi.substring(0, chushengriqi.length());

}catch(Exception e2){

chushengriqi = "2222-02-22";

}

}

if("NULL".equals(chushengriqi))

chushengriqi = null;

}

//户口性质

cell = row.getCell(6);

if(cell != null){

try{

hukouxingzhi = df.format(cell.getNumericCellValue());

}catch(Exception e){

hukouxingzhi = cell.getStringCellValue();

}

if("NULL".equals(hukouxingzhi))

hukouxingzhi = null;

}

//身份证号码

cell = row.getCell(15);

if(cell != null){

try{

shenfenzhenghaoma = df.format(cell.getNumericCellValue());

}catch(Exception e){

shenfenzhenghaoma = cell.getStringCellValue();

}

if(shenfenzhenghaoma.equals("NULL"))

shenfenzhenghaoma = null;

}

//文化程度

cell = row.getCell(16);

if(cell != null){

try{

wenhuachengdu = df.format(cell.getNumericCellValue());

}catch(Exception e){

wenhuachengdu = cell.getStringCellValue();

}

if("NULL".equals(wenhuachengdu))

wenhuachengdu = null;

}

super.getDefaultManager().executeSQL("insert into t\_anhuili\_family\_user values(SYS\_GUID(),?,?,?,?,?,?,?,?,?,?,?)",

benhuzhuzhi , yuhuzhuguanxi, xingming, shifouhuzhu, xingbie, chushengriqi,

hukouxingzhi,shenfenzhenghaoma,wenhuachengdu,"","0");

}

}else{

File file = new File("C:/Users/Administrator/Desktop/安慧里社区表2/"+j+".xls");

// 构造 XSSFWorkbook 对象，strPath 传入文件路径

POIFSFileSystem fs = null;

try {

fs = new POIFSFileSystem(new FileInputStream(file));

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

// 创建工作簿

HSSFWorkbook workBook = null;

try {

workBook = new HSSFWorkbook(fs);

} catch (IOException e) {

e.printStackTrace();

}

HSSFSheet sheet0 = workBook.getSheetAt(0);

HSSFRow row = null;

HSSFCell cell = null;

for (int i = 21; i < sheet0.getPhysicalNumberOfRows(); i++)

{

String benhuzhuzhi ="";

String yuhuzhuguanxi = "";

String xingming = "";

String shifouhuzhu = "";

String xingbie = "";

String chushengriqi="";

String hukouxingzhi="";

String shenfenzhenghaoma = "";

String wenhuachengdu = "";

String lianxidianhua = "";

row = sheet0.getRow(i);

//本户住址

cell = row.getCell(1);

if(cell != null){

try{

benhuzhuzhi = df.format(cell.getNumericCellValue());

}catch(Exception e){

benhuzhuzhi = cell.getStringCellValue();

}

if("NULL".equals(benhuzhuzhi))

benhuzhuzhi = null;

}

//与户主关系

cell = row.getCell(2);

if(cell != null){

try{

yuhuzhuguanxi = df.format(cell.getNumericCellValue());

}catch(Exception e){

yuhuzhuguanxi = cell.getStringCellValue().trim();

}

if("NULL".equals(yuhuzhuguanxi))

yuhuzhuguanxi = null;

}

int z = yuhuzhuguanxi.indexOf("户主");

if(z >= 0){

shifouhuzhu = "1";

}else{

shifouhuzhu = "0";

}

//姓名

cell = row.getCell(3);

if(cell != null){

xingming = cell.getStringCellValue().trim();

if("NULL".equals(xingming))

xingming = null;

}

if("0".equals(benhuzhuzhi) && "0".equals(yuhuzhuguanxi))

break;

//性别

cell = row.getCell(4);

if(cell != null){

xingbie = cell.getStringCellValue();

if(xingbie.equals("NULL"))

xingbie = "";

else

{

if("男".equals(xingbie))

xingbie="0";

if("女".equals(xingbie))

xingbie="1";

}

}else{

//没有设置性别的设置成 2

xingbie="2";

}

//出生日期

cell = row.getCell(14);

if(cell != null){

try{

chushengriqi = cell.getStringCellValue();

try{

if(!"NULL".equals(chushengriqi)){

chushengriqi = chushengriqi.substring(0, chushengriqi.length());

}

}catch(Exception ee){

chushengriqi = "1111-11-11";

}

}catch(Exception e){

chushengriqi = cell.getDateCellValue().toString();

try{

chushengriqi = chushengriqi.substring(0, chushengriqi.length());

}catch(Exception e2){

chushengriqi = "2222-02-22";

}

}

if("NULL".equals(chushengriqi))

chushengriqi = null;

}

//户口性质

cell = row.getCell(5);

if(cell != null){

try{

hukouxingzhi = df.format(cell.getNumericCellValue());

}catch(Exception e){

hukouxingzhi = cell.getStringCellValue();

}

if("NULL".equals(hukouxingzhi))

hukouxingzhi = null;

}

//身份证号码

cell = row.getCell(11);

if(cell != null){

try{

shenfenzhenghaoma = df.format(cell.getNumericCellValue());

}catch(Exception e){

shenfenzhenghaoma = cell.getStringCellValue();

}

if(shenfenzhenghaoma.equals("NULL"))

shenfenzhenghaoma = null;

}

//联系电话

cell = row.getCell(16);

if(cell != null){

try{

lianxidianhua = df.format(cell.getNumericCellValue());

}catch(Exception e){

lianxidianhua = cell.getStringCellValue();

}

if(lianxidianhua.equals("NULL"))

lianxidianhua = null;

}

super.getDefaultManager().executeSQL("insert into t\_anhuili\_family\_user values(SYS\_GUID(),?,?,?,?,?,?,?,?,?,?,?)",

benhuzhuzhi , yuhuzhuguanxi, xingming, shifouhuzhu, xingbie, chushengriqi,

hukouxingzhi,shenfenzhenghaoma,wenhuachengdu,lianxidianhua,"1");

}

}

}

logger.info("人口基本信息导入:", "end ++++++++++++++++++++++++++++++++++++++++++++++++++==");

logger.info("导入成功:", "end ++++++++++++++++++++++++++++++++++++++++++++++++++==");

System.out.println("导入成功！");

super.getSession().setAttribute("scuess", "导入成功！");

return true;

}

@Override

protected void prepareShow() {

super.prepareShow();

}

}

package com.bizlogic.qms.action;

import java.util.ArrayList;

import java.util.List;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.Company;

import com.bizlogic.base.front.action.BaseUserEditAction;

import com.bizlogic.common.Element;

import com.bizlogic.qms.bean.BuildStructure;

import com.bizlogic.qms.bean.BuildType;

import com.bizlogic.qms.bean.Building;

import com.bizlogic.qms.bean.BuildingType;

import com.bizlogic.qms.bean.ManageMode;

import com.bizlogic.qms.bean.Use;

public class BuildingEditAction extends BaseUserEditAction{

//楼院id

private String id;

//街道、社区id

private String companyId;

//建筑结构id

private String buildStructureId;

//建筑类型id

private String buildTypeId;

//楼院类型id

private String buildingTypeId;

//管理方式id

private String manageModeId;

//大类用途id

private String bigClassUseId;

//小类用途id

private String smallClassUseId;

//街道、社区列表

private List<Company> companyList;

//建筑结构列表

private List<BuildStructure> buildStructureList;

//建筑类型列表

private List<BuildType> buildTypeList;

//楼院类型列表

private List<BuildingType> buildingTypeList;

//管理方式列表

private List<ManageMode> manageModeList;

//大类用途列表

private List<Use> bigClassUseList;

//小类用途列表

private List<Use> smallClassUseList;

protected void prepareShow(){

Building building = (Building)element;

if(id != null){

building = super.getDefaultManager().get(Building.class, id);

if(building.getCompany() != null)

companyId = building.getCompany().getId();

if(building.getBuildStructure() != null)

buildStructureId = building.getBuildStructure().getId();

if(building.getBuildType() != null)

buildTypeId = building.getBuildType().getId();

if(building.getBuildingType() != null)

buildingTypeId = building.getBuildingType().getId();

if(building.getManageMode() != null)

manageModeId = building.getManageMode().getId();

if(building.getBigClassUse() != null)

bigClassUseId = building.getBigClassUse().getId();

if(building.getSmallClassUse() != null)

smallClassUseId = building.getSmallClassUse().getId();

}

/\*\*

\* 社区工作人员时判断searchCompanyId

\*/

String userid = this.getVisit().getId();

if(userid.indexOf("sq\_")>=0){

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' and c.id='"+this.getVisit().getCompany().getId()+"' order by c.id");

}else{

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' order by c.id");

}

buildStructureList = super.getDefaultManager().findAll("from BuildStructure");

buildTypeList = super.getDefaultManager().findAll("from BuildType");

buildingTypeList = super.getDefaultManager().findAll("from BuildingType");

manageModeList = super.getDefaultManager().findAll("from ManageMode");

bigClassUseList = super.getDefaultManager().findAll("from Use u where u.parentId is null");

if(bigClassUseId!=null && bigClassUseId.compareTo("")!=0){

smallClassUseList = super.getDefaultManager().findAll("from Use a where a.parentId.id =? order by a.id ",bigClassUseId);

}else{

smallClassUseList = new ArrayList();

}

}

protected boolean prepareSave(Element element){

Building building = (Building)element;

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

building.setId(id);

}else{

Object maxid = super.getDefaultManager().getObjectSQL("select max(t.building\_id) from t\_building t");

String gudinghao = maxid.toString().substring(0, 13);

String louhao = maxid.toString().substring(13, maxid.toString().length());

String newlouhao = getBuildingNum(louhao);

String s = gudinghao+newlouhao;

building.setId(s);

}

if (StringUtils.isNotBlank(companyId)) {

building.setCompany((Company)super.getDefaultManager().get(Company.class, companyId));

}else{

building.setBuildType(null);

}

if (StringUtils.isNotBlank(buildStructureId)) {

building.setBuildStructure((BuildStructure)super.getDefaultManager().get(BuildStructure.class, buildStructureId));

}else{

building.setBuildStructure(null);

}

if (StringUtils.isNotBlank(buildTypeId)) {

building.setBuildType((BuildType)super.getDefaultManager().get(BuildType.class, buildTypeId));

}else{

building.setBuildType(null);

}

if (StringUtils.isNotBlank(buildingTypeId)) {

building.setBuildingType((BuildingType)super.getDefaultManager().get(BuildingType.class, buildingTypeId));

}else{

building.setBuildingType(null);

}

if (StringUtils.isNotBlank(manageModeId)) {

building.setManageMode((ManageMode)super.getDefaultManager().get(ManageMode.class, manageModeId));

}else{

building.setManageMode(null);

}

if (StringUtils.isNotBlank(bigClassUseId)) {

building.setBigClassUse((Use)super.getDefaultManager().get(Use.class, bigClassUseId));

}else{

building.setBigClassUse(null);

}

if (StringUtils.isNotBlank(smallClassUseId)) {

building.setSmallClassUse((Use)super.getDefaultManager().get(Use.class, smallClassUseId));

}else{

building.setSmallClassUse(null);

}

return super.prepareSave(element);

}

//将楼院编号加1

public String getBuildingNum(String str){

StringBuffer s1 = new StringBuffer();

StringBuffer s2 = new StringBuffer();

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

if(i<7){

s1.append(str.charAt(i));

}else{

s2.append(str.charAt(i));

}

}

int n = new Integer(s2.toString());

n++;

String s3 = String.valueOf(n);

int cha = s3.length()-s2.length();

String rerurnstr = s1.toString().substring(cha, s1.toString().length());

return rerurnstr+s3;

}

protected Class getElementClass()

{

return Building.class;

}

public String getId() {

return id;

}

public String getBuildStructureId() {

return buildStructureId;

}

public String getBuildTypeId() {

return buildTypeId;

}

public String getBuildingTypeId() {

return buildingTypeId;

}

public String getManageModeId() {

return manageModeId;

}

public String getBigClassUseId() {

return bigClassUseId;

}

public String getSmallClassUseId() {

return smallClassUseId;

}

public void setId(String id) {

this.id = id;

}

public void setBuildStructureId(String buildStructureId) {

this.buildStructureId = buildStructureId;

}

public void setBuildTypeId(String buildTypeId) {

this.buildTypeId = buildTypeId;

}

public void setBuildingTypeId(String buildingTypeId) {

this.buildingTypeId = buildingTypeId;

}

public void setManageModeId(String manageModeId) {

this.manageModeId = manageModeId;

}

public void setBigClassUseId(String bigClassUseId) {

this.bigClassUseId = bigClassUseId;

}

public void setSmallClassUseId(String smallClassUseId) {

this.smallClassUseId = smallClassUseId;

}

public List<BuildStructure> getBuildStructureList() {

return buildStructureList;

}

public List<BuildType> getBuildTypeList() {

return buildTypeList;

}

public List<BuildingType> getBuildingTypeList() {

return buildingTypeList;

}

public List<ManageMode> getManageModeList() {

return manageModeList;

}

public List<Use> getBigClassUseList() {

return bigClassUseList;

}

public List<Use> getSmallClassUseList() {

return smallClassUseList;

}

public void setBuildStructureList(List<BuildStructure> buildStructureList) {

this.buildStructureList = buildStructureList;

}

public void setBuildTypeList(List<BuildType> buildTypeList) {

this.buildTypeList = buildTypeList;

}

public void setBuildingTypeList(List<BuildingType> buildingTypeList) {

this.buildingTypeList = buildingTypeList;

}

public void setManageModeList(List<ManageMode> manageModeList) {

this.manageModeList = manageModeList;

}

public void setBigClassUseList(List<Use> bigClassUseList) {

this.bigClassUseList = bigClassUseList;

}

public void setSmallClassUseList(List<Use> smallClassUseList) {

this.smallClassUseList = smallClassUseList;

}

public String getCompanyId() {

return companyId;

}

public void setCompanyId(String companyId) {

this.companyId = companyId;

}

public List<Company> getCompanyList() {

return companyList;

}

public void setCompanyList(List<Company> companyList) {

this.companyList = companyList;

}

}

package com.bizlogic.qms.action;

import java.io.ByteArrayInputStream;

import java.io.ByteArrayOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.util.List;

import org.apache.poi.hssf.usermodel.HSSFCell;

import org.apache.poi.hssf.usermodel.HSSFCellStyle;

import org.apache.poi.hssf.usermodel.HSSFRow;

import org.apache.poi.hssf.usermodel.HSSFSheet;

import org.apache.poi.hssf.usermodel.HSSFWorkbook;

import org.apache.poi.ss.usermodel.Font;

import org.apache.poi.ss.util.CellRangeAddress;

import com.bizlogic.base.Company;

import com.bizlogic.common.action.BaseAction;

import com.bizlogic.qms.bean.Building;

import com.bizlogic.qms.bean.Person;

public class BuildingExportAction extends BaseAction{

//选取导出的列

private boolean company;

private boolean buildType;

private boolean buildingType;

private boolean manageMode;

private boolean bigClassUse;

private boolean smallClassUse;

private boolean unitQuantity;

private boolean upstairsQuantity;

private boolean coordinateX;

private boolean coordinateY;

private boolean buildingAddress;

private boolean genericName;

private boolean hasAccess;

private boolean downstairsQuantity;

private boolean hasList;

private boolean listCount;

private boolean useArea;

private boolean groundBuildArea;

private boolean hasHouseProperty;

private boolean propertyUnit;

private boolean managementUnit;

private boolean managementPhone;

private boolean roomCount;

private boolean households;

//社区编号

private String companyId;

public String execute() throws Exception

{

return super.execute();

}

public String getFileName() {

String excelName = "";

try {

//获得社区对象

Company comp = super.getDefaultManager().get(Company.class, companyId);

excelName = new String((comp.getName()+"楼院数据.xls").getBytes("gbk"), "ISO8859\_1");

} catch (Exception e) {

}

return excelName;

}

public InputStream getExcelFile() {

//创建excel工作簿

HSSFWorkbook workbook = new HSSFWorkbook();

//获得社区对象

Company comp = super.getDefaultManager().get(Company.class, companyId);

//创建一页

HSSFSheet sheet = workbook.createSheet(comp.getName()+"楼院情况");

HSSFCellStyle style0 = workbook.createCellStyle();

//设置边框

style0.setBorderTop(HSSFCellStyle.BORDER\_THIN);

style0.setBorderLeft(HSSFCellStyle.BORDER\_THIN);

style0.setBorderRight(HSSFCellStyle.BORDER\_THIN);

style0.setBorderBottom(HSSFCellStyle.BORDER\_THIN);

//边框颜色

style0.setRightBorderColor((short) 00000);

//内容在单元格中间显示

style0.setVerticalAlignment(HSSFCellStyle.VERTICAL\_CENTER);

//内容居中显示

style0.setAlignment(HSSFCellStyle.ALIGN\_CENTER);

Font font = workbook.createFont();

//设置粗体

font.setBoldweight((short) 10);

//字体颜色

font.setColor((short) 00000);

//字体大小

font.setFontHeightInPoints((short) 12);

font.setFontName("宋体");

style0.setFont(font);

//标题样式

HSSFCellStyle style1 = workbook.createCellStyle();

//设置边框

style1.setBorderTop(HSSFCellStyle.BORDER\_THIN);

style1.setBorderLeft(HSSFCellStyle.BORDER\_THIN);

style1.setBorderRight(HSSFCellStyle.BORDER\_THIN);

style1.setBorderBottom(HSSFCellStyle.BORDER\_THIN);

//边框颜色

style1.setRightBorderColor((short) 00000);

//内容在单元格中间显示

style1.setVerticalAlignment(HSSFCellStyle.VERTICAL\_CENTER);

//内容居中显示

style1.setAlignment(HSSFCellStyle.ALIGN\_CENTER);

Font font1 = workbook.createFont();

font1.setBoldweight((short) 10);//设置粗体

font1.setColor((short) 00000);//字体颜色

font1.setFontHeightInPoints((short) 18);//字体大小

font1.setFontName("黑体");//设置字体

style1.setFont(font1);

//标题样式

HSSFCellStyle style2 = workbook.createCellStyle();

//设置边框

style2.setBorderTop(HSSFCellStyle.BORDER\_THIN);

style2.setBorderLeft(HSSFCellStyle.BORDER\_THIN);

style2.setBorderRight(HSSFCellStyle.BORDER\_THIN);

style2.setBorderBottom(HSSFCellStyle.BORDER\_THIN);

//边框颜色

style2.setRightBorderColor((short) 00000);

//内容在单元格中间显示

style2.setVerticalAlignment(HSSFCellStyle.VERTICAL\_CENTER);

//内容居中显示

style2.setAlignment(HSSFCellStyle.ALIGN\_CENTER);

Font font2 = workbook.createFont();

font2.setBoldweight((short) 10);//设置粗体

font2.setColor((short) 00000);//字体颜色

font2.setFontHeightInPoints((short) 13);//字体大小

font2.setFontName("黑体");

style2.setFont(font2);

int x = 0;

sheet.setColumnWidth(x++, 2000);

if(company){

sheet.setColumnWidth(x++, 8000);

}

if(buildType){

sheet.setColumnWidth(x++, 4000);

}

if(buildingType){

sheet.setColumnWidth(x++, 4000);

}

if(manageMode){

sheet.setColumnWidth(x++, 4000);

}

if(bigClassUse){

sheet.setColumnWidth(x++, 4000);

}

if(smallClassUse){

sheet.setColumnWidth(x++, 4000);

}

if(unitQuantity){

sheet.setColumnWidth(x++, 4000);

}

if(upstairsQuantity){

sheet.setColumnWidth(x++, 4000);

}

if(coordinateX){

sheet.setColumnWidth(x++, 4000);

}

if(coordinateY){

sheet.setColumnWidth(x++, 4000);

}

if(buildingAddress){

sheet.setColumnWidth(x++, 8000);

}

if(genericName){

sheet.setColumnWidth(x++, 7000);

}

if(hasAccess){

sheet.setColumnWidth(x++, 4000);

}

if(downstairsQuantity){

sheet.setColumnWidth(x++, 4000);

}

if(hasList){

sheet.setColumnWidth(x++, 4000);

}

if(listCount){

sheet.setColumnWidth(x++, 4000);

}

if(useArea){

sheet.setColumnWidth(x++, 4000);

}

if(groundBuildArea){

sheet.setColumnWidth(x++, 4000);

}

if(hasHouseProperty){

sheet.setColumnWidth(x++, 4000);

}

if(propertyUnit){

sheet.setColumnWidth(x++, 4000);

}

if(managementUnit){

sheet.setColumnWidth(x++, 4000);

}

if(managementPhone){

sheet.setColumnWidth(x++, 4000);

}

if(roomCount){

sheet.setColumnWidth(x++, 4000);

}

if(households){

sheet.setColumnWidth(x++, 4000);

}

//设置合并单元格（0行到0行，0列到10列进行合并）== 标题

sheet.addMergedRegion(new CellRangeAddress(0,0,0,x-1));

//第0行

HSSFRow row0 = sheet.createRow(0);

//0行0列值

HSSFCell cell0 = row0.createCell(0);

cell0.setCellValue(comp.getName()+"楼院情况");

cell0.setCellStyle(style1);

int flag = 1;

//第一行

HSSFRow row1 = sheet.createRow(1);

HSSFCell cell1 = row1.createCell(0);

cell1.setCellValue("序号");

cell1.setCellStyle(style2);

cell1 = row1.createCell(flag);

if(company){

cell1.setCellValue("所属社区");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(buildType){

cell1.setCellValue("建筑类型");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(buildingType){

cell1.setCellValue("楼院类型");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(manageMode){

cell1.setCellValue("管理方式");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(bigClassUse){

cell1.setCellValue("大类用途");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(smallClassUse){

cell1.setCellValue("小类用途");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(unitQuantity){

cell1.setCellValue("单元数");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(upstairsQuantity){

cell1.setCellValue("楼上层数");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(coordinateX){

cell1.setCellValue("坐标X");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(coordinateY){

cell1.setCellValue("坐标Y");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(buildingAddress){

cell1.setCellValue("楼院地址");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(genericName){

cell1.setCellValue("通用名称");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(hasAccess){

cell1.setCellValue("有无门禁");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(downstairsQuantity){

cell1.setCellValue("楼下层数");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(hasList){

cell1.setCellValue("有无电梯");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(listCount){

cell1.setCellValue("电梯数量");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(useArea){

cell1.setCellValue("总使用面积");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(groundBuildArea){

cell1.setCellValue("地上建筑面积");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(hasHouseProperty){

cell1.setCellValue("有无房产证");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(propertyUnit){

cell1.setCellValue("产权单位");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(managementUnit){

cell1.setCellValue("物流管理单位");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(managementPhone){

cell1.setCellValue("物流联系电话");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(roomCount){

cell1.setCellValue("房间数");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

if(households){

cell1.setCellValue("户数");

cell1.setCellStyle(style2);

cell1 = row1.createCell(++flag);

}

List<Building> buildingList = super.getDefaultManager().findAll("from Building b " +

"where b.company.id='"+companyId+"' ");

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

Building building = buildingList.get(i);

//获取行

HSSFRow row = sheet.getRow(i+2);

//如果没有是空，进行创建

if(row == null )

row = sheet.createRow(i+2);

int flagA = 0;

//编号

HSSFCell cell = row.createCell(0);

cell.setCellStyle(style0);

cell.setCellValue(i+1);

//所属社区

if(company){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getCompany() != null && building.getCompany().getName() != null){

cell.setCellValue(building.getCompany().getName());

}else{

cell.setCellValue("");

}

}

//楼名称

if(buildType){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getBuildType()!= null && building.getBuildType().getName() != null){

cell.setCellValue(building.getBuildType().getName());

}else{

cell.setCellValue("");

}

}

//楼院类型

if(buildingType){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getBuildingType()!= null && building.getBuildingType().getName() != null){

cell.setCellValue(building.getBuildingType().getName());

}else{

cell.setCellValue("");

}

}

//管理方式

if(manageMode){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getManageMode()!= null && building.getManageMode().getName() != null){

cell.setCellValue(building.getManageMode().getName());

}else{

cell.setCellValue("");

}

}

//大类用途

if(bigClassUse){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getBigClassUse()!= null && building.getBigClassUse().getName() != null){

cell.setCellValue(building.getBigClassUse().getName());

}else{

cell.setCellValue("");

}

}

//小类用途

if(smallClassUse){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getSmallClassUse()!= null && building.getSmallClassUse().getName() != null){

cell.setCellValue(building.getSmallClassUse().getName());

}else{

cell.setCellValue("");

}

}

//单元数

if(unitQuantity){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getUnitQuantity()>0){

cell.setCellValue(building.getUnitQuantity());

}else{

cell.setCellValue("");

}

}

//楼上层数

if(upstairsQuantity){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getUpstairsQuantity()>0){

cell.setCellValue(building.getUpstairsQuantity());

}else{

cell.setCellValue("");

}

}

//坐标X

if(coordinateX){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getCoordinateX() != null){

cell.setCellValue(building.getCoordinateX());

}else{

cell.setCellValue("");

}

}

//坐标Y

if(coordinateY){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getCoordinateY() != null){

cell.setCellValue(building.getCoordinateY());

}else{

cell.setCellValue("");

}

}

//楼院地址

if(buildingAddress){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getBuildingAddress()!= null){

cell.setCellValue(building.getBuildingAddress());

}else{

cell.setCellValue("");

}

}

//通用名称

if(genericName){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getGenericName()!= null){

cell.setCellValue(building.getGenericName());

}else{

cell.setCellValue("");

}

}

//有无门禁

if(hasAccess){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getHasAccess() == 0){

cell.setCellValue("无");

}else if(building.getHasAccess() == 1){

cell.setCellValue("有");

}else{

cell.setCellValue("");

}

}

//楼下层数

if(downstairsQuantity){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getDownstairsQuantity()>0){

cell.setCellValue(building.getDownstairsQuantity());

}else{

cell.setCellValue("");

}

}

//有无电梯

if(hasList){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getHasList() == 0){

cell.setCellValue("无");

}else if(building.getHasList() == 1){

cell.setCellValue("有");

}else{

cell.setCellValue("");

}

}

//电梯数量

if(listCount){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getListCount()>0){

cell.setCellValue(building.getListCount());

}else{

cell.setCellValue("");

}

}

//总使用面积

if(useArea){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getUseArea()>0){

cell.setCellValue(building.getUseArea());

}else{

cell.setCellValue("");

}

}

//地上建筑面积

if(groundBuildArea){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getGroundBuildArea()>0){

cell.setCellValue(building.getGroundBuildArea());

}else{

cell.setCellValue("");

}

}

//有无房产证

if(hasHouseProperty){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getHasHouseProperty() == 0){

cell.setCellValue("无");

}else if(building.getHasHouseProperty() == 1){

cell.setCellValue("有");

}else{

cell.setCellValue("");

}

}

//产权单位

if(propertyUnit){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getPropertyUnit() != null){

cell.setCellValue(building.getPropertyUnit());

}else{

cell.setCellValue("");

}

}

//物流管理单位

if(managementUnit){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getManagementUnit() != null){

cell.setCellValue(building.getManagementUnit());

}else{

cell.setCellValue("");

}

}

//物流联系电话

if(managementPhone){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getManagementPhone() != null){

cell.setCellValue(building.getManagementPhone());

}else{

cell.setCellValue("");

}

}

//房间数

if(roomCount){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getRoomCount()>0){

cell.setCellValue(building.getRoomCount());

}else{

cell.setCellValue("");

}

}

//户数

if(households){

cell = row.createCell(++flagA);

cell.setCellStyle(style0);

if(building.getHouseholds()>0){

cell.setCellValue(building.getHouseholds());

}else{

cell.setCellValue("");

}

}

}

ByteArrayOutputStream baos = new ByteArrayOutputStream();

try {

workbook.write(baos);

} catch (IOException e) {

e.printStackTrace();

}

byte[] ba = baos.toByteArray();

ByteArrayInputStream bais = new ByteArrayInputStream(ba);

return bais;

}

//set、get方法

public boolean isCompany() {

return company;

}

public void setCompany(boolean company) {

this.company = company;

}

public boolean isBuildType() {

return buildType;

}

public void setBuildType(boolean buildType) {

this.buildType = buildType;

}

public boolean isBuildingType() {

return buildingType;

}

public void setBuildingType(boolean buildingType) {

this.buildingType = buildingType;

}

public boolean isManageMode() {

return manageMode;

}

public void setManageMode(boolean manageMode) {

this.manageMode = manageMode;

}

public boolean isBigClassUse() {

return bigClassUse;

}

public void setBigClassUse(boolean bigClassUse) {

this.bigClassUse = bigClassUse;

}

public boolean isSmallClassUse() {

return smallClassUse;

}

public void setSmallClassUse(boolean smallClassUse) {

this.smallClassUse = smallClassUse;

}

public boolean isUnitQuantity() {

return unitQuantity;

}

public void setUnitQuantity(boolean unitQuantity) {

this.unitQuantity = unitQuantity;

}

public boolean isUpstairsQuantity() {

return upstairsQuantity;

}

public void setUpstairsQuantity(boolean upstairsQuantity) {

this.upstairsQuantity = upstairsQuantity;

}

public boolean isCoordinateX() {

return coordinateX;

}

public void setCoordinateX(boolean coordinateX) {

this.coordinateX = coordinateX;

}

public boolean isCoordinateY() {

return coordinateY;

}

public void setCoordinateY(boolean coordinateY) {

this.coordinateY = coordinateY;

}

public boolean isBuildingAddress() {

return buildingAddress;

}

public void setBuildingAddress(boolean buildingAddress) {

this.buildingAddress = buildingAddress;

}

public boolean isGenericName() {

return genericName;

}

public void setGenericName(boolean genericName) {

this.genericName = genericName;

}

public boolean isHasAccess() {

return hasAccess;

}

public void setHasAccess(boolean hasAccess) {

this.hasAccess = hasAccess;

}

public boolean isDownstairsQuantity() {

return downstairsQuantity;

}

public void setDownstairsQuantity(boolean downstairsQuantity) {

this.downstairsQuantity = downstairsQuantity;

}

public boolean isHasList() {

return hasList;

}

public void setHasList(boolean hasList) {

this.hasList = hasList;

}

public boolean isListCount() {

return listCount;

}

public void setListCount(boolean listCount) {

this.listCount = listCount;

}

public boolean isUseArea() {

return useArea;

}

public void setUseArea(boolean useArea) {

this.useArea = useArea;

}

public boolean isGroundBuildArea() {

return groundBuildArea;

}

public void setGroundBuildArea(boolean groundBuildArea) {

this.groundBuildArea = groundBuildArea;

}

public boolean isHasHouseProperty() {

return hasHouseProperty;

}

public void setHasHouseProperty(boolean hasHouseProperty) {

this.hasHouseProperty = hasHouseProperty;

}

public boolean isPropertyUnit() {

return propertyUnit;

}

public void setPropertyUnit(boolean propertyUnit) {

this.propertyUnit = propertyUnit;

}

public boolean isManagementUnit() {

return managementUnit;

}

public void setManagementUnit(boolean managementUnit) {

this.managementUnit = managementUnit;

}

public boolean isManagementPhone() {

return managementPhone;

}

public void setManagementPhone(boolean managementPhone) {

this.managementPhone = managementPhone;

}

public boolean isRoomCount() {

return roomCount;

}

public void setRoomCount(boolean roomCount) {

this.roomCount = roomCount;

}

public boolean isHouseholds() {

return households;

}

public void setHouseholds(boolean households) {

this.households = households;

}

public String getCompanyId() {

return companyId;

}

public void setCompanyId(String companyId) {

this.companyId = companyId;

}

}

package com.bizlogic.qms.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.Company;

import com.bizlogic.base.Department;

import com.bizlogic.base.User;

import com.bizlogic.base.front.action.BaseUserListAction;

import com.bizlogic.common.action.helper.ListInfo;

import com.bizlogic.qms.bean.BuildStructure;

import com.bizlogic.qms.bean.BuildType;

import com.bizlogic.qms.bean.Building;

import com.bizlogic.qms.bean.BuildingType;

import com.bizlogic.qms.bean.HasAccess;

import com.bizlogic.qms.bean.HasHouseProperty;

import com.bizlogic.qms.bean.HasList;

import com.bizlogic.qms.bean.ManageMode;

import com.bizlogic.qms.bean.Use;

public class BuildingListAction extends BaseUserListAction{

//楼院名称（地址）搜索词

private String searchName;

//所属社区查询编号

private String companyId ;

//建筑结构查询编号

private String buildStructureId;

//建筑类型查询编号

private String buildTypeId;

//楼院类型查询编号

private String buildingTypeId;

//管理方式查询编号

private String manageModeId;

//大类用途查询编号

private String bigClassUseId;

//小类用途查询编号

private String smallClassUseId;

//有无电梯查询id

private String hasListId;

//有无门禁查询id

private String hasAccessId;

//有无房产证查询id

private String hasHousePropertyId;

//街道、社区列表

private List<Company> companyList;

//建筑结构列表

private List<BuildStructure> buildStructureList;

//建筑类型

private List<BuildType> buildTypeList;

//楼院类型

private List<BuildingType> buildingTypeList;

//管理方式列表

private List<ManageMode> manageModeList;

//大类用途列表

private List<Use> bigClassUseList;

//小类用途列表

private List<Use> smallClassUseList;

//有无电梯列表

private List<HasList> hasListList;

//有无门禁列表

private List<HasAccess> hasAccessList;

//有无房产证列表

private List<HasHouseProperty> hasHousePropertyList;

public List<Company> getCompanyList() {

String userid = this.getVisit().getId();

if(userid.indexOf("sq\_")>=0){

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' and c.id='"+this.getVisit().getCompany().getId()+"' order by c.id");

}else{

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' order by c.id");

}

return companyList;

}

public List<BuildStructure> getBuildStructureList() {

buildStructureList = super.getDefaultManager().findAll("from BuildStructure b order by b.id");

return buildStructureList;

}

public List<BuildType> getBuildTypeList() {

buildTypeList = super.getDefaultManager().findAll("from BuildType b order by b.id");

return buildTypeList;

}

public List<BuildingType> getBuildingTypeList() {

buildingTypeList = super.getDefaultManager().findAll("from BuildingType b order by b.id");

return buildingTypeList;

}

public List<ManageMode> getManageModeList() {

manageModeList = super.getDefaultManager().findAll("from ManageMode m order by m.id");

return manageModeList;

}

public List<Use> getBigClassUseList() {

return bigClassUseList = super.getDefaultManager().findAll("from Use u where u.parentId is null order by u.id");

}

public List<Use> getSmallClassUseList() {

if(bigClassUseId!=null && bigClassUseId.compareTo("")!=0){

smallClassUseList = super.getDefaultManager().findAll("from Use a where a.parentId.id =? order by a.id ",bigClassUseId);

return smallClassUseList;

}

else{

return new ArrayList();

}

}

public List<HasList> getHasListList() {

List<HasList> hlist = new ArrayList<HasList>();

HasList h1 = new HasList();

HasList h2 = new HasList();

h1.setListId(0);

h1.setName("无电梯");

hlist.add(h1);

h2.setListId(1);

h2.setName("有电梯");

hlist.add(h2);

hasListList = hlist;

return hasListList;

}

public List<HasAccess> getHasAccessList() {

List<HasAccess> haccess = new ArrayList<HasAccess>();

HasAccess h1 = new HasAccess();

HasAccess h2 = new HasAccess();

h1.setAccessId(0);

h1.setName("无门禁");

haccess.add(h1);

h2.setAccessId(1);

h2.setName("有门禁");

haccess.add(h2);

hasAccessList = haccess;

return hasAccessList;

}

public List<HasHouseProperty> getHasHousePropertyList() {

List<HasHouseProperty> hhouse = new ArrayList<HasHouseProperty>();

HasHouseProperty h1 = new HasHouseProperty();

HasHouseProperty h2 = new HasHouseProperty();

h1.setHousePropertyId(0);

h1.setName("无房产证");

hhouse.add(h1);

h2.setHousePropertyId(1);

h2.setName("有房产证");

hhouse.add(h2);

hasHousePropertyList = hhouse;

return hasHousePropertyList;

}

@Override

protected long doGetCount(ListInfo listInfo) {

//清楚session

super.getSession().removeAttribute("floorId");

super.getSession().removeAttribute("buildingId");

/\*\*

\* 社区工作人员时判断searchCompanyId

\*/

if(StringUtils.isEmpty(companyId)){

String userid = this.getVisit().getId();

if(userid.indexOf("sq\_")>=0){

companyId=this.getVisit().getCompany().getId();

}

}

sqlCondition = this.getConditions(searchName,companyId,buildStructureId,buildTypeId,buildingTypeId,manageModeId,

hasListId,hasAccessId,hasHousePropertyId,bigClassUseId,smallClassUseId);

return super.getDefaultManager().getCount("from Building a where 1=1 "+ sqlCondition);

}

@Override

protected List doSearch(ListInfo listInfo) {

List list = super.getDefaultManager().find("from Building a where 1=1 "+ sqlCondition +" order by a.lineNumber,a.buildingAddress",

listInfo.getFirstNo(), listInfo.getMaxNo());

return list;

}

public String delete() {

String[] ids = getSelectedIds();

if (ids != null) {

for (String id : ids) {

//先删除楼户

super.getDefaultManager().executeSQL("delete from t\_room t where t.building\_id=?", id);

//再删除楼层

super.getDefaultManager().executeSQL("delete from t\_floor t where t.building\_id=?", id);

//再删除楼院

deleteObjectById(id);

}

}

return SUCCESS;

}

/\*\*

\* 获得查询条件

\*/

protected String getConditions(String searchName,String companyId,String buildStructureId,String buildTypeId,

String buildingTypeId,String manageModeId,String hasListId,String hasAccessId,String hasHousePropertyId,

String searchBigClassUseId,String searchSmallClassUseId){

String conditionsStr = "";

if(searchName!=null && searchName.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildingAddress like '%"+ searchName.trim() +"%'";

}

if(companyId!=null && companyId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.company.id ='"+ companyId +"'";

}

if(buildStructureId!=null && buildStructureId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildStructure.id ='"+ buildStructureId +"'";

}

if(buildTypeId!=null && buildTypeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildType.id ='"+ buildTypeId +"'";

}

if(buildingTypeId!=null && buildingTypeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildingType.id ='"+ buildingTypeId +"'";

}

if(manageModeId!=null && manageModeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.manageMode.id ='"+ manageModeId +"'";

}

if(hasListId!=null && hasListId.compareTo("")!=0){

if(hasListId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasList=1";

}else if(hasListId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasList=0";

}

}

if(hasAccessId!=null && hasAccessId.compareTo("")!=0){

if(hasAccessId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasAccess=1";

}else if(hasAccessId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasAccess=0";

}

}

if(hasHousePropertyId!=null && hasHousePropertyId.compareTo("")!=0){

if(hasHousePropertyId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasHouseProperty=1";

}else if(hasHousePropertyId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasHouseProperty=0";

}

}

if(searchBigClassUseId!=null && searchBigClassUseId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.bigClassUse.id ='"+ searchBigClassUseId +"'";

}

if(searchSmallClassUseId!=null && searchSmallClassUseId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.smallClassUse.id ='"+ searchSmallClassUseId +"'";

}

return conditionsStr;

}

/\*\*

\* ajax请求

\* @throws IOException

\*/

public void ajaxMethod() throws IOException {

JSONObject res = new JSONObject();

//Response编码设置

super.getResponse().setCharacterEncoding("utf-8");

if (StringUtils.equals(act, "get\_small\_class\_use\_list")) {

super.getRequest().getQueryString();

String bigId = super.getParameterValue("bigId");

List<Use> dl = new ArrayList<Use>();

if (StringUtils.isNotBlank(bigId)) {

dl = super.getDefaultManager().findAll("from Use u where u.parentId.id=? ",bigId);

}

JSONObject[] contents = new JSONObject[dl.size()];

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

contents[i] = new JSONObject();

contents[i].put("value", dl.get(i).getId());

contents[i].put("text", dl.get(i).getName());

}

res.put("content", contents);

}

//结果输出

PrintWriter out = this.getResponse().getWriter();

out.println(res.toString());

out.flush();

out.close();

}

protected Class getImplementClass()

{

return Building.class;

}

public String getSearchName() {

return searchName;

}

public void setSearchName(String searchName) {

this.searchName = searchName;

}

public String getBuildStructureId() {

return buildStructureId;

}

public void setBuildStructureId(String buildStructureId) {

this.buildStructureId = buildStructureId;

}

public String getBuildTypeId() {

return buildTypeId;

}

public void setBuildTypeId(String buildTypeId) {

this.buildTypeId = buildTypeId;

}

public String getBuildingTypeId() {

return buildingTypeId;

}

public void setBuildingTypeId(String buildingTypeId) {

this.buildingTypeId = buildingTypeId;

}

public String getBigClassUseId() {

return bigClassUseId;

}

public void setBigClassUseId(String bigClassUseId) {

this.bigClassUseId = bigClassUseId;

}

public String getManageModeId() {

return manageModeId;

}

public void setManageModeId(String manageModeId) {

this.manageModeId = manageModeId;

}

public String getSmallClassUseId() {

return smallClassUseId;

}

public void setSmallClassUseId(String smallClassUseId) {

this.smallClassUseId = smallClassUseId;

}

public String getHasListId() {

return hasListId;

}

public void setHasListId(String hasListId) {

this.hasListId = hasListId;

}

public String getHasAccessId() {

return hasAccessId;

}

public String getHasHousePropertyId() {

return hasHousePropertyId;

}

public void setHasAccessId(String hasAccessId) {

this.hasAccessId = hasAccessId;

}

public void setHasHousePropertyId(String hasHousePropertyId) {

this.hasHousePropertyId = hasHousePropertyId;

}

public String getCompanyId() {

return companyId;

}

public void setCompanyId(String companyId) {

this.companyId = companyId;

}

}

package com.bizlogic.qms.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.front.action.BaseUserListAction;

import com.bizlogic.common.action.helper.ListInfo;

import com.bizlogic.qms.bean.Floor;

import com.bizlogic.qms.bean.Person;

import com.bizlogic.qms.bean.Room;

public class BuildingRoomShowAction extends BaseUserListAction{

//楼编号

private String buildingId;

//返回html字符串

private String buildingRoomStr = "";

@Override

protected long doGetCount(ListInfo listInfo) {

//楼编号

buildingId = super.getParameterValue("buildingId");

if(buildingId!=null && buildingId.compareTo("")!=0){

super.getSession().setAttribute("buildingId", buildingId);

}else{

buildingId = (String)super.getSession().getAttribute("buildingId");

}

return 0;

}

@Override

protected List doSearch(ListInfo listInfo) {

//获得楼层数量

List floorList = super.getDefaultManager().findAll("from Floor a where a.building.id=? order by a.lineNumber desc ",buildingId);

buildingRoomStr += "<table cellpadding='0' cellspacing='0' class='list\_tb' style='width : 500px;'>";

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

Floor floor = (Floor)floorList.get(i);

if(i%2==0){

buildingRoomStr += "<tr>";

}else{

buildingRoomStr += "<tr class='colorstlye'>";

}

List roomList = super.getDefaultManager().findAll("from Room a where a.status=1 and a.floor.id=? order by a.roomNum ",floor.getId());

for(int j=0;j<roomList.size();j++){

Room room = (Room)roomList.get(j);

//查看该户居住人的数量

Long l = super.getDefaultManager().getCount("from Person a where a.room.id=? ",room.getId());

String nullStr = "(<font color=red>空</font>)";

if(l>0){

nullStr="";

}

if(room.getFloor().getLineNumber()<0){

buildingRoomStr += "<td style='text-align:center'>"

+"<a href='javascript:void(0);' onclick=javascript:getRoomView('"+room.getId()+"'); ><strong>"

+room.getRoomNum()+"(地下)</strong>"+nullStr+"</a>"

+"</td>";

}else{

buildingRoomStr += "<td style='text-align:center'>"

+"<a href='javascript:void(0);' onclick=javascript:getRoomView('"+room.getId()+"'); ><strong>"

+room.getRoomNum()+"</strong>"+nullStr+"</a>"

+"</td>";

}

}

buildingRoomStr += "</tr>";

}

buildingRoomStr += "</table>";

return null;

}

/\*\*

\* ajax请求

\* @throws IOException

\*/

public void ajaxMethod() throws IOException {

JSONObject res = new JSONObject();

//Response编码设置

super.getResponse().setCharacterEncoding("utf-8");

if (StringUtils.equals(act, "get\_room\_view")) {

String str = "";

super.getRequest().getQueryString();

String roomId = super.getParameterValue("roomId");

Room room = super.getDefaultManager().get(Room.class, roomId);

str += "<div class='content\_tab\_outstyle'>";

str += "<p class='addition\_info\_yellow'>楼户详细信息</p>";

str += "<table cellpadding='0' cellspacing='0' class='content\_tab' >";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>房屋地址</th>";

str += "<td width='35%'>"+room.getRoomAddress()+"</td>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>所属社区</th>";

str += "<td width='35%'>"+room.getBuilding().getCompany().getName()+"</td>";

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>所属楼院</th>";

str += "<td>"+room.getBuilding().getBuildingAddress()+"</td>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>所属楼层</th>";

str += "<td>"+room.getFloor().getFloorName()+"</td>";

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>房间门牌号</th>";

str += "<td>"+room.getRoomNum()+"</td>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>居室类型</th>";

if(room.getRoomType()!=null){

str += "<td>"+room.getRoomType().getName()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>建筑面积</th>";

if(room.getBuildArea() == 0){

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}else{

str += "<td>"+room.getBuildArea()+"</td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>使用面积</th>";

if(room.getUseArea() == 0){

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}else{

str += "<td>"+room.getUseArea()+"</td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>规划用途</th>";

if(room.getPalnUse()!=null){

str += "<td>"+room.getPalnUse().getName()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>实际用途</th>";

if(room.getActualUse()!=null){

str += "<td>"+room.getActualUse().getName()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>产权人</th>";

if(room.getHouseHolder()!=null){

str += "<td>"+room.getHouseHolder()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>实际使用人</th>";

if(room.getActualUser()!=null){

str += "<td>"+room.getActualUser()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>家庭编码</th>";

if(room.getHomeNum()!=null){

str += "<td>"+room.getHomeNum()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>户主编码</th>";

if(room.getHouseHolderNum()!=null){

str += "<td>"+room.getHouseHolderNum()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>id</th>";

if(room.getXxid()!=null){

str += "<td>"+room.getXxid()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>使用状态</th>";

if(room.getUseStatus()!=null){

str += "<td>"+room.getUseStatus().getName()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>产权方类型</th>";

if(room.getHolderType()!=null){

str += "<td>"+room.getHolderType().getName()+"</td>";

}else{

str += "<td><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th><font color='#FF0000' style='font-weight:normal;'></font>&nbsp;</th>";

str += "<td>&nbsp;</td>";

str += "</tr>";

str += " </table>";

List personList = super.getDefaultManager().findAll("from Person a where a.room.id=? order by a.personRelationType.id",roomId);

str += "<p class='addition\_info\_yellow'>住户情况</p>";

str += "<table cellpadding='0' cellspacing='0' class='list\_tb\_yellow' >";

str += "<tr>";

str += "<th width='10%'><strong>姓名</strong></th>";

str += "<th width='10%'><strong>与户主关系</strong></th>";

str += "<th width='10%'><strong>性别</strong></th>";

str += "</tr>";

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

Person person = (Person)personList.get(i);

str += "<tr>";

str += "<td align='left' style='text-align:center'>"+

"<a href='javascript:personShow("+'"' + roomId +'"'+","+'"' + person.getId() +'"'+");'><strong>"+person.getName()+"</strong></a>"

+"</td>";

if(person.getPersonRelationType()!=null){

if("户主".equals(person.getPersonRelationType().getName()) ){

str += "<td align='left' style='text-align:center'>本人</td>";

}else{

str += "<td align='left' style='text-align:center'>"+person.getPersonRelationType().getName()+"</td>";

}

}else{

str += "<td align='left' style='text-align:center'></td>";

}

if(person.getGender() ==0){

str += "<td align='left' style='text-align:center'>男</td>";

}else{

str += "<td align='left' style='text-align:center'>女</td>";

}

str += "</tr>";

}

str += "</table>";

str += "<br>";

str += "</div>";

res.put("returnStr", str);

}

//获取person信息

if (StringUtils.equals(act, "get\_person\_view")) {

String str = "";

String personId = super.getParameterValue("personId");

String roomId = super.getParameterValue("roomId");

Person person = super.getDefaultManager().get(Person.class, personId);

str += "<div class='content\_tab\_outstyle'>";

str += "<p class='addition\_info\_yellow'>人员详细信息</p>";

str += "<table cellpadding='0' cellspacing='0' class='content\_tab' >";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>姓名</th>";

str += "<td width='35%'>"+person.getName()+"</td>";

//与户主关系

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>与户主关系</th>";

if(person.getHousehold() == 1){

str += "<td width='35%'>户主</td>";

}else if(person.getPersonRelationType() != null){

str += "<td width='35%'>"+person.getPersonRelationType().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>性别</th>";

if(person.getGender() == 0){

str += "<td width='35%'>男</td>";

}else if(person.getGender() == 1){

str += "<td width='35%'>女</td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>出生日期</th>";

if(person.getBirthday() != null){

str += "<td width='35%'>"+person.getBirthday()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>婚姻状况</th>";

if(person.getMarriage() != null){

str += "<td width='35%'>"+person.getMarriage().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>电话</th>";

if(person.getMobile() != null){

str += "<td width='35%'>"+person.getMobile()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>国家</th>";

if(person.getCountry() != null){

str += "<td width='35%'>"+person.getCountry().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>民族</th>";

if(person.getNation() != null){

str += "<td width='35%'>"+person.getNation().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>证件类型</th>";

if(person.getCertificateType() != null){

str += "<td width='35%'>"+person.getCertificateType().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>证件号码</th>";

if(person.getIdentityCardTypeNumber() != null){

str += "<td width='35%'>"+person.getIdentityCardTypeNumber()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>籍贯</th>";

if(person.getHometown() != null){

str += "<td width='35%'>"+person.getHometown()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>户口所在地</th>";

if(person.getAccountLocation() != null){

str += "<td width='35%'>"+person.getAccountLocation()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>户别</th>";

if(person.getPersonAccountType() != null){

str += "<td width='35%'>"+person.getPersonAccountType().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>户籍状况</th>";

if(person.getPersonAccountStatus() != null){

str += "<td width='35%'>"+person.getPersonAccountStatus().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>工作单位</th>";

if(person.getWorkunit() != null){

str += "<td width='35%'>"+person.getWorkunit()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>所在社区</th>";

if(person.getCompany() != null){

str += "<td width='35%'>"+person.getCompany().getName()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "</tr>";

str += "<tr>";

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font>家庭地址</th>";

if(person.getHomeAddress() != null){

str += "<td width='35%'>"+person.getHomeAddress()+"</td>";

}else{

str += "<td width='35%'><font color='red'><strong>未登记</strong></font></td>";

}

str += "<th width='15%'><font color='#FF0000' style='font-weight:normal;'></font></th>";

str += "<td width='35%'></td>";

str += "</tr>";

str += "</table>";

str += "<br>";

str += "<input type='button' class='pop\_btnTask' value='返回楼户' onclick=javascript:backToFamily('"+roomId+"');return false;'/>";

str += "</div>";

res.put("returnStr", str);

}

//结果输出

PrintWriter out = this.getResponse().getWriter();

out.println(res.toString());

out.flush();

out.close();

}

protected Class getImplementClass()

{

return Room.class;

}

public String getBuildingId() {

return buildingId;

}

public void setBuildingId(String buildingId) {

this.buildingId = buildingId;

}

public String getBuildingRoomStr() {

return buildingRoomStr;

}

public void setBuildingRoomStr(String buildingRoomStr) {

this.buildingRoomStr = buildingRoomStr;

}

}

package com.bizlogic.qms.action;

import java.util.ArrayList;

import java.util.List;

import com.bizlogic.base.front.action.BaseUserEditAction;

import com.bizlogic.qms.bean.Building;

public class BuildingShowAction extends BaseUserEditAction{

//楼层列表

private List buildingFloorList;

protected Class getElementClass()

{

return Building.class;

}

protected void prepareShow(){

Building building = (Building)element;

buildingFloorList = super.getDefaultManager().findAll("from Floor a where a.building=? order by a.atFloorNum asc", building);

}

public List getBuildingFloorList() {

return buildingFloorList;

}

public void setBuildingFloorList(List buildingFloorList) {

this.buildingFloorList = buildingFloorList;

}

}

package com.bizlogic.qms.action;

import java.util.ArrayList;

import java.io.FileNotFoundException;

import java.util.List;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.Company;

import com.bizlogic.base.front.action.BaseUserEditAction;

import com.bizlogic.common.Element;

import com.bizlogic.qms.bean.BuildStructure;

import com.bizlogic.qms.bean.BuildType;

import com.bizlogic.qms.bean.Building;

import com.bizlogic.qms.bean.BuildingType;

import com.bizlogic.qms.bean.Community;

import com.bizlogic.qms.bean.ManageMode;

import com.bizlogic.qms.bean.Use;

public class CommunityBuildingEditAction extends BaseUserEditAction{

//楼院id

private String id;

//街道、社区id

private String companyId;

//建筑结构id

private String buildStructureId;

//建筑类型id

private String buildTypeId;

//楼院类型id

private String buildingTypeId;

//管理方式id

private String manageModeId;

//大类用途id

private String bigClassUseId;

//小类用途id

private String smallClassUseId;

//街道、社区列表

private List<Company> companyList;

//建筑结构列表

private List<BuildStructure> buildStructureList;

//建筑类型列表

private List<BuildType> buildTypeList;

//楼院类型列表

private List<BuildingType> buildingTypeList;

//管理方式列表

private List<ManageMode> manageModeList;

//大类用途列表

private List<Use> bigClassUseList;

//小类用途列表

private List<Use> smallClassUseList;

protected void prepareShow(){

Building building = (Building)element;

if(id != null){

building = super.getDefaultManager().get(Building.class, id);

if(building.getCompany() != null)

companyId = building.getCompany().getId();

if(building.getBuildStructure() != null)

buildStructureId = building.getBuildStructure().getId();

if(building.getBuildType() != null)

buildTypeId = building.getBuildType().getId();

if(building.getBuildingType() != null)

buildingTypeId = building.getBuildingType().getId();

if(building.getManageMode() != null)

manageModeId = building.getManageMode().getId();

if(building.getBigClassUse() != null)

bigClassUseId = building.getBigClassUse().getId();

if(building.getSmallClassUse() != null)

smallClassUseId = building.getSmallClassUse().getId();

}

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' order by c.id");

buildStructureList = super.getDefaultManager().findAll("from BuildStructure");

buildTypeList = super.getDefaultManager().findAll("from BuildType");

buildingTypeList = super.getDefaultManager().findAll("from BuildingType");

manageModeList = super.getDefaultManager().findAll("from ManageMode");

bigClassUseList = super.getDefaultManager().findAll("from Use u where u.parentId is null");

if(bigClassUseId!=null && bigClassUseId.compareTo("")!=0){

smallClassUseList = super.getDefaultManager().findAll("from Use a where a.parentId.id =? order by a.id ",bigClassUseId);

}else{

smallClassUseList = new ArrayList();

}

}

protected boolean prepareSave(Element element){

Building building = (Building)element;

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

building.setId(id);

}else{

Object maxid = super.getDefaultManager().getObjectSQL("select max(t.building\_id) from t\_building t");

String gudinghao = maxid.toString().substring(0, 13);

String louhao = maxid.toString().substring(13, maxid.toString().length());

String newlouhao = getBuildingNum(louhao);

String s = gudinghao+newlouhao;

building.setId(s);

}

if (StringUtils.isNotBlank(companyId)) {

building.setCompany((Company)super.getDefaultManager().get(Company.class, companyId));

}else{

building.setBuildType(null);

}

if (StringUtils.isNotBlank(buildStructureId)) {

building.setBuildStructure((BuildStructure)super.getDefaultManager().get(BuildStructure.class, buildStructureId));

}else{

building.setBuildStructure(null);

}

if (StringUtils.isNotBlank(buildTypeId)) {

building.setBuildType((BuildType)super.getDefaultManager().get(BuildType.class, buildTypeId));

}else{

building.setBuildType(null);

}

if (StringUtils.isNotBlank(buildingTypeId)) {

building.setBuildingType((BuildingType)super.getDefaultManager().get(BuildingType.class, buildingTypeId));

}else{

building.setBuildingType(null);

}

if (StringUtils.isNotBlank(manageModeId)) {

building.setManageMode((ManageMode)super.getDefaultManager().get(ManageMode.class, manageModeId));

}else{

building.setManageMode(null);

}

if (StringUtils.isNotBlank(bigClassUseId)) {

building.setBigClassUse((Use)super.getDefaultManager().get(Use.class, bigClassUseId));

}else{

building.setBigClassUse(null);

}

if (StringUtils.isNotBlank(smallClassUseId)) {

building.setSmallClassUse((Use)super.getDefaultManager().get(Use.class, smallClassUseId));

}else{

building.setSmallClassUse(null);

}

return super.prepareSave(element);

}

//将楼院编号加1

public String getBuildingNum(String str){

StringBuffer s1 = new StringBuffer();

StringBuffer s2 = new StringBuffer();

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

if(i<7){

s1.append(str.charAt(i));

}else{

s2.append(str.charAt(i));

}

}

int n = new Integer(s2.toString());

n++;

String s3 = String.valueOf(n);

int cha = s3.length()-s2.length();

String rerurnstr = s1.toString().substring(cha, s1.toString().length());

return rerurnstr+s3;

}

protected Class getElementClass()

{

return Building.class;

}

public String getId() {

return id;

}

public String getBuildStructureId() {

return buildStructureId;

}

public String getBuildTypeId() {

return buildTypeId;

}

public String getBuildingTypeId() {

return buildingTypeId;

}

public String getManageModeId() {

return manageModeId;

}

public String getBigClassUseId() {

return bigClassUseId;

}

public String getSmallClassUseId() {

return smallClassUseId;

}

public void setId(String id) {

this.id = id;

}

public void setBuildStructureId(String buildStructureId) {

this.buildStructureId = buildStructureId;

}

public void setBuildTypeId(String buildTypeId) {

this.buildTypeId = buildTypeId;

}

public void setBuildingTypeId(String buildingTypeId) {

this.buildingTypeId = buildingTypeId;

}

public void setManageModeId(String manageModeId) {

this.manageModeId = manageModeId;

}

public void setBigClassUseId(String bigClassUseId) {

this.bigClassUseId = bigClassUseId;

}

public void setSmallClassUseId(String smallClassUseId) {

this.smallClassUseId = smallClassUseId;

}

public List<BuildStructure> getBuildStructureList() {

return buildStructureList;

}

public List<BuildType> getBuildTypeList() {

return buildTypeList;

}

public List<BuildingType> getBuildingTypeList() {

return buildingTypeList;

}

public List<ManageMode> getManageModeList() {

return manageModeList;

}

public List<Use> getBigClassUseList() {

return bigClassUseList;

}

public List<Use> getSmallClassUseList() {

return smallClassUseList;

}

public void setBuildStructureList(List<BuildStructure> buildStructureList) {

this.buildStructureList = buildStructureList;

}

public void setBuildTypeList(List<BuildType> buildTypeList) {

this.buildTypeList = buildTypeList;

}

public void setBuildingTypeList(List<BuildingType> buildingTypeList) {

this.buildingTypeList = buildingTypeList;

}

public void setManageModeList(List<ManageMode> manageModeList) {

this.manageModeList = manageModeList;

}

public void setBigClassUseList(List<Use> bigClassUseList) {

this.bigClassUseList = bigClassUseList;

}

public void setSmallClassUseList(List<Use> smallClassUseList) {

this.smallClassUseList = smallClassUseList;

}

public String getCompanyId() {

return companyId;

}

public void setCompanyId(String companyId) {

this.companyId = companyId;

}

public List<Company> getCompanyList() {

return companyList;

}

public void setCompanyList(List<Company> companyList) {

this.companyList = companyList;

}

}

package com.bizlogic.qms.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.Company;

import com.bizlogic.base.front.action.BaseUserListAction;

import com.bizlogic.common.action.helper.ListInfo;

import com.bizlogic.qms.bean.BuildStructure;

import com.bizlogic.qms.bean.BuildType;

import com.bizlogic.qms.bean.Building;

import com.bizlogic.qms.bean.BuildingType;

import com.bizlogic.qms.bean.HasAccess;

import com.bizlogic.qms.bean.HasHouseProperty;

import com.bizlogic.qms.bean.HasList;

import com.bizlogic.qms.bean.ManageMode;

import com.bizlogic.qms.bean.Use;

public class CommunityBuildingListAction extends BaseUserListAction{

//楼院名称（地址）搜索词

private String searchName;

//所属社区查询编号

private String companyId = "110105009005";

//建筑结构查询编号

private String buildStructureId;

//建筑类型查询编号

private String buildTypeId;

//楼院类型查询编号

private String buildingTypeId;

//管理方式查询编号

private String manageModeId;

//大类用途查询编号

private String bigClassUseId;

//小类用途查询编号

private String smallClassUseId;

//有无电梯查询id

private String hasListId;

//有无门禁查询id

private String hasAccessId;

//有无房产证查询id

private String hasHousePropertyId;

//街道、社区列表

private List<Company> companyList;

//建筑结构列表

private List<BuildStructure> buildStructureList;

//建筑类型

private List<BuildType> buildTypeList;

//楼院类型

private List<BuildingType> buildingTypeList;

//管理方式列表

private List<ManageMode> manageModeList;

//大类用途列表

private List<Use> bigClassUseList;

//小类用途列表

private List<Use> smallClassUseList;

//有无电梯列表

private List<HasList> hasListList;

//有无门禁列表

private List<HasAccess> hasAccessList;

//有无房产证列表

private List<HasHouseProperty> hasHousePropertyList;

public List<Company> getCompanyList() {

companyList = super.getDefaultManager().findAll("from Company c where c.id <> '110105009' order by c.id");

return companyList;

}

public List<BuildStructure> getBuildStructureList() {

buildStructureList = super.getDefaultManager().findAll("from BuildStructure b order by b.id");

return buildStructureList;

}

public List<BuildType> getBuildTypeList() {

buildTypeList = super.getDefaultManager().findAll("from BuildType b order by b.id");

return buildTypeList;

}

public List<BuildingType> getBuildingTypeList() {

buildingTypeList = super.getDefaultManager().findAll("from BuildingType b order by b.id");

return buildingTypeList;

}

public List<ManageMode> getManageModeList() {

manageModeList = super.getDefaultManager().findAll("from ManageMode m order by m.id");

return manageModeList;

}

public List<Use> getBigClassUseList() {

return bigClassUseList = super.getDefaultManager().findAll("from Use u where u.parentId is null order by u.id");

}

public List<Use> getSmallClassUseList() {

if(bigClassUseId!=null && bigClassUseId.compareTo("")!=0){

smallClassUseList = super.getDefaultManager().findAll("from Use a where a.parentId.id =? order by a.id ",bigClassUseId);

return smallClassUseList;

}

else{

return new ArrayList();

}

}

public List<HasList> getHasListList() {

List<HasList> hlist = new ArrayList<HasList>();

HasList h1 = new HasList();

HasList h2 = new HasList();

h1.setListId(0);

h1.setName("无电梯");

hlist.add(h1);

h2.setListId(1);

h2.setName("有电梯");

hlist.add(h2);

hasListList = hlist;

return hasListList;

}

public List<HasAccess> getHasAccessList() {

List<HasAccess> haccess = new ArrayList<HasAccess>();

HasAccess h1 = new HasAccess();

HasAccess h2 = new HasAccess();

h1.setAccessId(0);

h1.setName("无门禁");

haccess.add(h1);

h2.setAccessId(1);

h2.setName("有门禁");

haccess.add(h2);

hasAccessList = haccess;

return hasAccessList;

}

public List<HasHouseProperty> getHasHousePropertyList() {

List<HasHouseProperty> hhouse = new ArrayList<HasHouseProperty>();

HasHouseProperty h1 = new HasHouseProperty();

HasHouseProperty h2 = new HasHouseProperty();

h1.setHousePropertyId(0);

h1.setName("无房产证");

hhouse.add(h1);

h2.setHousePropertyId(1);

h2.setName("有房产证");

hhouse.add(h2);

hasHousePropertyList = hhouse;

return hasHousePropertyList;

}

@Override

protected long doGetCount(ListInfo listInfo) {

//清楚session

super.getSession().removeAttribute("floorId");

super.getSession().removeAttribute("buildingId");

sqlCondition = this.getConditions(searchName,companyId,buildStructureId,buildTypeId,buildingTypeId,manageModeId,

hasListId,hasAccessId,hasHousePropertyId,bigClassUseId,smallClassUseId);

return super.getDefaultManager().getCount("from Building a where 1=1 "+ sqlCondition);

}

@Override

protected List doSearch(ListInfo listInfo) {

List list = super.getDefaultManager().find("from Building a where 1=1 "+ sqlCondition +" order by a.lineNumber,a.buildingAddress",

listInfo.getFirstNo(), listInfo.getMaxNo());

return list;

}

public String delete() {

String[] ids = getSelectedIds();

if (ids != null) {

for (String id : ids) {

//先删除楼户

super.getDefaultManager().executeSQL("delete from t\_room t where t.building\_id=?", id);

//再删除楼层

super.getDefaultManager().executeSQL("delete from t\_floor t where t.building\_id=?", id);

//再删除楼院

deleteObjectById(id);

}

}

return SUCCESS;

}

/\*\*

\* 获得查询条件

\*/

protected String getConditions(String searchName,String companyId,String buildStructureId,String buildTypeId,

String buildingTypeId,String manageModeId,String hasListId,String hasAccessId,String hasHousePropertyId,

String searchBigClassUseId,String searchSmallClassUseId){

String conditionsStr = "";

if(searchName!=null && searchName.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildingAddress like '%"+ searchName.trim() +"%'";

}

if(companyId!=null && companyId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.company.id ='"+ companyId +"'";

}

if(buildStructureId!=null && buildStructureId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildStructure.id ='"+ buildStructureId +"'";

}

if(buildTypeId!=null && buildTypeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildType.id ='"+ buildTypeId +"'";

}

if(buildingTypeId!=null && buildingTypeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.buildingType.id ='"+ buildingTypeId +"'";

}

if(manageModeId!=null && manageModeId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.manageMode.id ='"+ manageModeId +"'";

}

if(hasListId!=null && hasListId.compareTo("")!=0){

if(hasListId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasList=1";

}else if(hasListId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasList=0";

}

}

if(hasAccessId!=null && hasAccessId.compareTo("")!=0){

if(hasAccessId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasAccess=1";

}else if(hasAccessId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasAccess=0";

}

}

if(hasHousePropertyId!=null && hasHousePropertyId.compareTo("")!=0){

if(hasHousePropertyId.equals("1")){

conditionsStr = conditionsStr+ " and a.hasHouseProperty=1";

}else if(hasHousePropertyId.equals("0")){

conditionsStr = conditionsStr+ " and a.hasHouseProperty=0";

}

}

if(searchBigClassUseId!=null && searchBigClassUseId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.bigClassUse.id ='"+ searchBigClassUseId +"'";

}

if(searchSmallClassUseId!=null && searchSmallClassUseId.compareTo("")!=0){

conditionsStr = conditionsStr+ " and a.smallClassUse.id ='"+ searchSmallClassUseId +"'";

}

return conditionsStr;

}

/\*\*

\* ajax请求

\* @throws IOException

\*/

public void ajaxMethod() throws IOException {

JSONObject res = new JSONObject();

//Response编码设置

super.getResponse().setCharacterEncoding("utf-8");

if (StringUtils.equals(act, "get\_small\_class\_use\_list")) {

super.getRequest().getQueryString();

String bigId = super.getParameterValue("bigId");

List<Use> dl = new ArrayList<Use>();

if (StringUtils.isNotBlank(bigId)) {

dl = super.getDefaultManager().findAll("from Use u where u.parentId.id=? ",bigId);

}

JSONObject[] contents = new JSONObject[dl.size()];

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

contents[i] = new JSONObject();

contents[i].put("value", dl.get(i).getId());

contents[i].put("text", dl.get(i).getName());

}

res.put("content", contents);

}

//结果输出

PrintWriter out = this.getResponse().getWriter();

out.println(res.toString());

out.flush();

out.close();

}

protected Class getImplementClass()

{

return Building.class;

}

public String getSearchName() {

return searchName;

}

public void setSearchName(String searchName) {

this.searchName = searchName;

}

public String getBuildStructureId() {

return buildStructureId;

}

public void setBuildStructureId(String buildStructureId) {

this.buildStructureId = buildStructureId;

}

public String getBuildTypeId() {

return buildTypeId;

}

public void setBuildTypeId(String buildTypeId) {

this.buildTypeId = buildTypeId;

}

public String getBuildingTypeId() {

return buildingTypeId;

}

public void setBuildingTypeId(String buildingTypeId) {

this.buildingTypeId = buildingTypeId;

}

public String getBigClassUseId() {

return bigClassUseId;

}

public void setBigClassUseId(String bigClassUseId) {

this.bigClassUseId = bigClassUseId;

}

public String getManageModeId() {

return manageModeId;

}

public void setManageModeId(String manageModeId) {

this.manageModeId = manageModeId;

}

public String getSmallClassUseId() {

return smallClassUseId;

}

public void setSmallClassUseId(String smallClassUseId) {

this.smallClassUseId = smallClassUseId;

}

public String getHasListId() {

return hasListId;

}

public void setHasListId(String hasListId) {

this.hasListId = hasListId;

}

public String getHasAccessId() {

return hasAccessId;

}

public String getHasHousePropertyId() {

return hasHousePropertyId;

}

public void setHasAccessId(String hasAccessId) {

this.hasAccessId = hasAccessId;

}

public void setHasHousePropertyId(String hasHousePropertyId) {

this.hasHousePropertyId = hasHousePropertyId;

}

public String getCompanyId() {

return companyId;

}

public void setCompanyId(String companyId) {

this.companyId = companyId;

}

}

package com.bizlogic.qms.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.front.action.BaseUserListAction;

import com.bizlogic.common.Element;

import com.bizlogic.common.action.helper.ListInfo;

import com.bizlogic.qms.bean.Building;

public class CommunityBuildingMapAction extends BaseUserListAction{

//社区建筑信息

private String communityBuildingInfoStr = "";

protected long doGetCount(ListInfo listInfo) {

return 0;

}

protected List doSearch(ListInfo listInfo) {

long buildingCount = super.getDefaultManager().getCount("from Building a where a.company=? ", this.getVisit().getCompany());

communityBuildingInfoStr += "<table width='250px' border='0'>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td colspan='2' width='240' align='center' valign='middle' bgcolor='#fff000'> <strong>统计</strong></td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >楼院合计:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong>"+buildingCount+"</strong></font>栋</td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >住宅楼:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong> 未统计 </strong></font></td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >写字楼:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong> 未统计 </strong></font></td>";

communityBuildingInfoStr += "</tr>";

long personCount = super.getDefaultManager().getCount("from Person a where a.company=? ", this.getVisit().getCompany());

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >总人口:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong>"+personCount+"</strong></font>人</td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >常住人口人口:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong> 未统计 </strong></font></td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "<tr>";

communityBuildingInfoStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >流动人口:</td>";

communityBuildingInfoStr += "<td width='143' align='center' valign='middle'><font color='red'><strong> 未统计 </strong></font></td>";

communityBuildingInfoStr += "</tr>";

communityBuildingInfoStr += "</table>";

return null;

}

/\*\*

\* ajax请求

\* @throws IOException

\*/

public void ajaxMethod() throws IOException {

JSONObject res = new JSONObject();

//Response编码设置

super.getResponse().setCharacterEncoding("utf-8");

//获得系统一级菜单

if (StringUtils.equals(act, "get\_building")) {

String returnStr = "";

String buildingId = super.getParameterValue("buildingId");

Building building = super.getDefaultManager().get(Building.class, buildingId);

if(building!=null){

returnStr += "<table width='250px' border='0'>";

returnStr += "<tr>";

returnStr += "<td colspan='2' width='240' align='center' valign='middle' bgcolor='#fff000'>"

+"<a href='buildingEdit.action?id="+building.getId()+"'><strong>"

+building.getGenericName()+"</strong></a></td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >所属社区:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getCompany().getName()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >楼院地址:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getBuildingAddress()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >单元数:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getUnitQuantity()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >楼上层数:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getUpstairsQuantity()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >楼下层数:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getDownstairsQuantity()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >有无门禁:</td>";

if(building.getHasAccess() == 0){

returnStr += "<td width='143' align='center' valign='middle'> 无 </td>";

}else{

returnStr += "<td width='143' align='center' valign='middle'> 有 </td>";

}

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >有无电梯:</td>";

if(building.getHasList() == 0){

returnStr += "<td width='143' align='center' valign='middle'> 无 </td>";

}else{

returnStr += "<td width='143' align='center' valign='middle'> 有 </td>";

}

returnStr += "</tr>";

if(building.getHasList() == 1){

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >电梯数量:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getListCount()+"</td>";

returnStr += "</tr>";

}

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >总使用面积:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getUseArea()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >地上建筑面积:</td>";

returnStr += "<td width='143' align='center' valign='middle'>"+building.getGroundBuildArea()+"</td>";

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >有无房产证:</td>";

if(building.getHasHouseProperty() == 0){

returnStr += "<td width='143' align='center' valign='middle'> 无 </td>";

}else{

returnStr += "<td width='143' align='center' valign='middle'> 有 </td>";

}

returnStr += "</tr>";

returnStr += "<tr>";

returnStr += "<td width='97' align='right' valign='middle' bgcolor='#b8def3' >建筑结构:</td>";

if(building.getBuildStructure()!=null){

returnStr += "<td width='143' align='center' valign='middle'>"+building.getBuildStructure().getName()+"</td>";

}else{

returnStr += "<td width='143' align='center' valign='middle'></td>";

//结果输出

PrintWriter out = this.getResponse().getWriter();

out.println(res.toString());

out.flush();

out.close();

}

/\*\*

\* className

\*/

protected Class getImplementClass()

{

return Element.class;

}

public String getCommunityBuildingInfoStr() {

return communityBuildingInfoStr;

}

public void setCommunityBuildingInfoStr(String communityBuildingInfoStr) {

this.communityBuildingInfoStr = communityBuildingInfoStr;

}

}

-----END------