package com.bizlogic.mz.action;

import java.io.File;

import java.util.ArrayList;

import java.util.List;

import com.bizlogic.base.AttachmentValue;

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

import com.bizlogic.base.helper.AttachmentHelper;

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

public class AddAllowanceAttachmentAction extends BaseUserListAction{

/\*\*

\* 附件manager

\*/

private AttachmentHelper attachmentHelper;

//上传文件

private File uploadify;

//文件名称

private String uploadifyFileName;

//文件类型

private String uploadifyContentType;

private long serial = 0;

private String allowanceReapetId;

@SuppressWarnings("deprecation")

public String uploadFile() throws Exception {

attachmentHelper.init(super.getDefaultManager());

AttachmentValue value = new AttachmentValue();

String prefix = uploadifyFileName.substring(uploadifyFileName.lastIndexOf(".")+1);

String contentType = getFileContentTypeStr(prefix);

value.setContentType(contentType);

value.setFileName(uploadifyFileName);

value.setType("ALLOWANCE\_REPEAT\_ATTACHMENT");

value.setName(uploadifyFileName.substring(0,uploadifyFileName.lastIndexOf(".")));

value.setSize(uploadify.length());

value.setUploadFile(uploadify);

value.setObjectId(allowanceReapetId);

value.setRealName(System.currentTimeMillis() + (serial++) + "." + getFileExtName(value.getFileName()));

value.setStatusId("1010");

//保存附件

attachmentHelper.saveAttachment(value);

super.getResponse().getWriter().print(uploadifyFileName+"上传成功");

return null; //这里不需要页面转向，所以返回空就可以了

}

private String getFileExtName(String filename)

{

int pos = filename.lastIndexOf(".");

if (pos > 0)

{

return filename.substring(pos + 1);

} else

{

return "";

}

}

private String getFileContentTypeStr(String typeStr){

String returnStr = null;

if(typeStr!=null && typeStr.equals("pdf")){

returnStr = "application/pdf";

}

if(typeStr!=null && typeStr.equals("htm")){

returnStr = "text/html";

}

if(typeStr!=null && typeStr.equals("mht")){

returnStr = "message/rfc822";

}

if(typeStr!=null && typeStr.equals("txt")){

returnStr = "text/plain";

}

if(typeStr!=null && typeStr.equals("html")){

returnStr = "text/html";

}

if(typeStr!=null && typeStr.equals("xls")){

returnStr = "application/vnd.ms-excel";

}

if(typeStr!=null && typeStr.equals("doc")){

returnStr = "application/msword";

}

if(typeStr!=null && typeStr.equals("ppt")){

returnStr = "application/vnd.ms-powerpoint";

}

if(typeStr!=null && typeStr.equals("pptx")){

returnStr = "application/vnd.openxmlformats-officedocument.presentationml.presentation";

}

if(typeStr!=null && typeStr.equals("docx")){

returnStr = "application/vnd.openxmlformats-officedocument.wordprocessingml.document";

}

if(typeStr!=null && typeStr.equals("xlsx")){

returnStr = "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet";

}

if(typeStr!=null && typeStr.equals("avi")){

returnStr = "video/avi";

}

if(typeStr!=null && typeStr.equals("wmv")){

returnStr = "video/x-ms-wmv";

}

if(typeStr!=null && typeStr.equals("swf")){

returnStr = "application/x-shockwave-flash";

}

if(typeStr!=null && typeStr.equals("exe")){

returnStr = "application/x-shockwave-flash";

}

if(typeStr!=null && typeStr.equals("jpg")){

returnStr = "pic/jpg";

}

return returnStr;

}

protected long doGetCount(ListInfo listInfo)

{

return 0;

}

protected List doSearch(ListInfo listInfo)

{

return new ArrayList();

}

public File getUploadify() {

return uploadify;

}

public void setUploadify(File uploadify) {

this.uploadify = uploadify;

}

public String getUploadifyFileName() {

return uploadifyFileName;

}

public void setUploadifyFileName(String uploadifyFileName) {

this.uploadifyFileName = uploadifyFileName;

}

public AttachmentHelper getAttachmentHelper() {

return attachmentHelper;

}

public void setAttachmentHelper(AttachmentHelper attachmentHelper) {

this.attachmentHelper = attachmentHelper;

}

public String getUploadifyContentType() {

return uploadifyContentType;

}

public void setUploadifyContentType(String uploadifyContentType) {

this.uploadifyContentType = uploadifyContentType;

}

public long getSerial() {

return serial;

}

public void setSerial(long serial) {

this.serial = serial;

}

public String getAllowanceReapetId() {

return allowanceReapetId;

}

public void setAllowanceReapetId(String allowanceReapetId) {

this.allowanceReapetId = allowanceReapetId;

}

}

package com.bizlogic.mz.action;

import java.io.File;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import com.bizlogic.base.AttachmentValue;

import com.bizlogic.base.User;

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

import com.bizlogic.base.helper.AttachmentHelper;

import com.bizlogic.common.Element;

import com.bizlogic.mz.bean.AllowanceAndPersonType;

import com.bizlogic.mz.bean.AllowanceChange;

import com.bizlogic.mz.bean.AllowanceRepeatChecked;

import com.bizlogic.mz.bean.CacheFamily;

import com.bizlogic.mz.bean.CacheFamilyUser;

import com.bizlogic.mz.bean.Family;

import com.bizlogic.mz.bean.FamilyChange;

import com.bizlogic.mz.bean.FamilyChangeType;

import com.bizlogic.mz.bean.FamilyUser;

import com.bizlogic.mz.bean.FamilyUserWelfareItem;

import com.bizlogic.mz.bean.FamilyWelfareItem;

import com.bizlogic.mz.bean.WelfareItem;

public class AllowanceChangeEditAction extends BaseUserEditAction{

private String familyId;

//家庭变更时间

private Date changeDate;

//是否继续享受低保

private int isAllow;

//附件上传----start

private AttachmentHelper attachmentHelper;

private File[] fileAttachment;

private String[] fileAttachmentContentType;

private String[] fileAttachmentFileName;

private long serial = 0;

//附件列表

private List<AttachmentValue> attachmentList;

private Family family;

private CacheFamily cacheFamily;

//家庭成员列表

private List<FamilyUser> familyUserList;

//缓存家庭成员列表

private List cacheFamilyUserList;

//低保变更时选择的享受低保的家人

private String[] checkAllowanceUser;

//填写家庭福利列表

private List<FamilyUserWelfareItem> familyUserWelfareItemList;

//已选择的福利展示

private List <FamilyWelfareItem> familyWelfareItemList;

//低保复审单编号

private String allowanceRepeatId;

//变更办理时间

private Date changeEditDate;

//福利选择框

private String welfareItemStr;

//选择的享受福利项id

private String[] welfareItemIds;

//福利表单填写展示表单字符串

private String welfareStr;

//填写框校验id

private String validatorIds;

//操作表单的标志，editFlag值为save时表示保存表单，editFlag值为edit是表示修改表单，editFlag值为submit时表示提交表单

private String editFlag;

//低保复审时需要隐藏的人员id

private String familyUserIds;

//享受低保类别编号

private String allowanceTypeId;

//人员类别编号

private String personTypeId;

//享受低保类别选择列表

private List<AllowanceAndPersonType> allowanceTypeList;

//人员类别选择列表

private List<AllowanceAndPersonType> personTypeList;

// protected Element createElement()

// {

// AllowanceChange allowanceChange = (AllowanceChange)element;

// allowanceChange.setCreateUser((User)getVisit());

// allowanceChange.setCreateDate(new Date());

// return allowanceChange;

// }

protected void prepareShow(){

AllowanceChange allowanceChange = (AllowanceChange)element;

if(allowanceChange.getId() == null){

changeDate = new Date();

changeEditDate = new Date();

}else{

changeDate = allowanceChange.getChangeDate();

changeEditDate = allowanceChange.getCreateDate();

}

//低保金、粮油补助金、教育救助金、司法救助、住房救助、慈善救助、生活困难补助

String strSql = " w.id=10270001 or w.id=10270002 or w.id=10270003 or w.id=10270006 or w.id=10270007 or w.id=10270008 or w.id=10270013 order by w.id ";

String cacheFamilyId = "";

welfareStr = "";

validatorIds = "";

//变更表单和低保复审编号都为空，表明新添加变更单

if(allowanceChange.getId() == null && allowanceRepeatId == null && !"".equals(allowanceRepeatId)){

family = super.getDefaultManager().get(Family.class, familyId);

//家庭人员列表

familyUserList = super.getDefaultManager().findAll("from FamilyUser a where a.family.id=? and a.status=1 order by a.personRelationType.id", familyId);

//填写福利列表

// familyUserWelfareItemList = new ArrayList<FamilyUserWelfareItem>();

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

// //默认列出低保金、粮油补助金、教育救助金额填写框，默认隐藏教育救助金填写框

// List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 or a.id=10270003 or a.id=10270007 or a.id=10270008 or a.id=10270006 order by a.id");

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

// FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

// familyUserWelfareItem.setFamilyUser(familyUserList.get(i));

// familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

// familyUserWelfareItem.setStatus(1);

// familyUserWelfareItemList.add(familyUserWelfareItem);

// }

// }

if(family.getAllowanceAndPersonType() != null && family.getAllowanceAndPersonType().getParent() != null){

allowanceTypeId = family.getAllowanceAndPersonType().getParent().getId();

}

if(family.getAllowanceAndPersonType() != null){

personTypeId = family.getAllowanceAndPersonType().getId();

}

//享受低保类别

allowanceTypeList = super.getDefaultManager().findAll("from AllowanceAndPersonType t where t.parent is null");

//人员类别

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

personTypeList = super.getDefaultManager().findAll("from AllowanceAndPersonType t where t.parent.id=?",allowanceTypeId);

}else{

personTypeList = new ArrayList();

}

//多选框

welfareItemStr = "";

//默认列表

List<WelfareItem> list = super.getDefaultManager().findAll("from WelfareItem a where a.status=? order by a.id", 1);

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

WelfareItem item = (WelfareItem)list.get(i);

String welfareId = item.getId();

String welfareName = item.getName();

//查询当前编辑人员是否已经享受该项福利

List familyWelfareList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? and a.welfareItem=?", cacheFamilyId,item);

if(!"10270004".equals(welfareId) && !"10270005".equals(welfareId) && !"10270009".equals(welfareId) && !"10270010".equals(welfareId) && !"10270011".equals(welfareId) && !"10270012".equals(welfareId)){

if(familyWelfareList.size() > 0){

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' /> " + welfareName+" ";

}else{

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' /> " + welfareName+" ";

}

}

}

//福利填写表格

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem w where " + strSql);

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr id='head'><th>姓名\\福利类别</th>");

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

str.append("<th id='"+welfareItemList.get(i).getId()+"'>"+welfareItemList.get(i).getName()+"</th>");

}

str.append("</tr>");

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

str.append("<tr id='"+familyUserList.get(i).getId()+"'>");

str.append("<td>"+familyUserList.get(i).getName()+"</td>");

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

validatorIds = validatorIds + "," + familyUserList.get(i).getId()+welfareItemList.get(j).getId();

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(j).getId()+"'><input size='10' onFocus='validator(this.id)' name='"+familyUserList.get(i).getId()+welfareItemList.get(j).getId()+"' id='"+familyUserList.get(i).getId()+welfareItemList.get(j).getId()+"' value='"+0.00+"'/> 元/月</td>");

}

str.append("</tr>");

}

str.append("</table>");

str.append("<table cellpadding='0' cellspacing='0' class='content\_tab'>");

str.append("<tr>");

str.append("<th width='15%'>添加附件</th>");

str.append("<td colspan='3'>");

str.append("<input type='file' name='fileAttachment' size='50' onchange='addFileAttachment(this)'/>&nbsp;<br>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

welfareStr = str.toString();

//变更单编号不为空，低保复审编号为空，表明查看该变更单，或者查看已保存变更单

}else if(allowanceChange.getId() != null && allowanceRepeatId == null && !"".equals(allowanceRepeatId)){

cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceChange.getId());

//附件列表

attachmentList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime", allowanceChange.getId());

//等于1表示该变更单提交后查看，等于0表示保存后查看变更单

if(allowanceChange.getStatus() == 1){

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=? and a.status=1", cacheFamily.getId());

//成功变更后的福利展示

familyUserWelfareItemList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.cacheFamily.id=? order by a.familyUser.personRelationType.id,a.cacheFamilyUser.id,a.welfareItem.id", cacheFamily.getId());

//该家庭享受福利项展示

familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? order by a.welfareItem.id", cacheFamily.getId());

//福利展示表格

List<FamilyWelfareItem> familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem f where f.cacheFamily.objectId=? order by f.welfareItem.id", allowanceChange.getId());

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr><th>姓名\\福利类别</th>");

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

str.append("<th>"+familyWelfareItemList.get(i).getWelfareItem().getName()+"</th>");

}

str.append("</tr>");

//该低保单对应的缓存家人列表

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1 order by a.personRelationType.id", allowanceChange.getId());

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

str.append("<tr>");

str.append("<td>"+cacheFamilyUserList.get(i).getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUserList.get(i).getId());

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

str.append("<td align='right' style='text-align:right'>"+familyUserWelfareList.get(j).getMonthAmount()+" 元/月</td>");

}

str.append("</tr>");

}

str.append("</table>");

welfareStr = str.toString();

}else{

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=?", cacheFamily.getId());

//多选框

welfareItemStr = "";

//默认列表

List<WelfareItem> list = super.getDefaultManager().findAll("from WelfareItem a where a.status=? order by a.id", 1);

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

WelfareItem item = (WelfareItem)list.get(i);

String welfareId = item.getId();

String welfareName = item.getName();

//查询当前编辑人员是否已经享受该项福利

List familyWelfareList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? and a.welfareItem=?", cacheFamily.getId(),item);

if(!"10270004".equals(welfareId) && !"10270005".equals(welfareId) && !"10270009".equals(welfareId) && !"10270010".equals(welfareId) && !"10270011".equals(welfareId) && !"10270012".equals(welfareId)){

if(familyWelfareList.size() > 0){

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' /> " + welfareName+" ";

}else{

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' /> " + welfareName+" ";

}

}

}

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem w where " + strSql);

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr id='head'><th>姓名\\福利类别</th>");

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

str.append("<th id='"+welfareItemList.get(i).getId()+"'>"+welfareItemList.get(i).getName()+"</th>");

}

str.append("</tr>");

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

CacheFamilyUser cacheFamilyUser = (CacheFamilyUser)cacheFamilyUserList.get(i);

str.append("<tr id='"+cacheFamilyUser.getId()+"'>");

str.append("<td>"+cacheFamilyUser.getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUser.getId());

for(int x = 0; x<welfareItemList.size(); x++){

int flag = 0;

double money = 0.00;

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

String welId = familyUserWelfareList.get(j).getWelfareItem().getId();

if(welId.equals(welfareItemList.get(x).getId())){

flag ++;

money = familyUserWelfareList.get(j).getMonthAmount();

break;

}

}

validatorIds = validatorIds + "," + cacheFamilyUser.getFamily().getId()+welfareItemList.get(x).getId();

if(flag > 0){

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(x).getId()+"'><input onFocus='validator(this.id)' name='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' id='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' value='"+money+"' size='10'/> 元/月</td>");

}else{

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(x).getId()+"'><input onFocus='validator(this.id)' name='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' id='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' value='"+money+"' size='10'/> 元/月</td>");

}

}

str.append("</tr>");

}

str.append("</table>");

str.append("<table cellpadding='0' cellspacing='0' class='content\_tab'>");

str.append("<tr>");

str.append("<th width='15%'>添加附件</th>");

str.append("<td colspan='3'>");

str.append("<input type='file' name='fileAttachment' size='50' onchange='addFileAttachment(this)'/>&nbsp;<br>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

welfareStr = str.toString();

}

//变更单编号为空，低保复审编号不为空，表明填写低保复审单后，进入变更单

}else if(allowanceChange.getId() == null && allowanceRepeatId != null && !"".equals(allowanceRepeatId)){

cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatId);

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=? and a.status=1", cacheFamily.getId());

//多选框

welfareItemStr = "";

//默认列表

List<WelfareItem> list = super.getDefaultManager().findAll("from WelfareItem a where a.status=? order by a.id", 1);

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

WelfareItem item = (WelfareItem)list.get(i);

String welfareId = item.getId();

String welfareName = item.getName();

//查询当前编辑人员是否已经享受该项福利

List familyWelfareList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? and a.welfareItem=?", cacheFamilyId,item);

if(!"10270004".equals(welfareId) && !"10270005".equals(welfareId) && !"10270009".equals(welfareId) && !"10270010".equals(welfareId) && !"10270011".equals(welfareId) && !"10270012".equals(welfareId)){

if(familyWelfareList.size() > 0){

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' /> " + welfareName+" ";

}else{

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' /> " + welfareName+" ";

}

}

}

if(isAllow == 1){

//福利填写表格

familyUserIds = "";

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem w where " + strSql);

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr id='head'><th>姓名\\福利类别</th>");

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

str.append("<th id='"+welfareItemList.get(i).getId()+"'>"+welfareItemList.get(i).getName()+"</th>");

}

str.append("</tr>");

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

CacheFamilyUser cacheFamilyUser = (CacheFamilyUser)cacheFamilyUserList.get(i);

familyUserIds = familyUserIds + "," + cacheFamilyUser.getFamilyUser().getId();

str.append("<tr id='"+cacheFamilyUser.getFamilyUser().getId()+"'>");

str.append("<td>"+cacheFamilyUser.getName()+"</td>");

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

validatorIds = validatorIds + "," + cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(j).getId();

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(j).getId()+"'><input onFocus='validator(this.id)' size='10' name='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(j).getId()+"' id='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(j).getId()+"'> 元/月</td>");

}

str.append("</tr>");

}

str.append("</table>");

str.append("<table cellpadding='0' cellspacing='0' class='content\_tab'>");

str.append("<tr>");

str.append("<th width='15%'>添加附件</th>");

str.append("<td colspan='3'>");

str.append("<input type='file' name='fileAttachment' size='50' onchange='addFileAttachment(this)'/>&nbsp;<br>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

welfareStr = str.toString();

}

//变更单编号不为空，低保复审编号不为空，表明填写低保复审单后，进入变更单，变更单保存没有提交

}else if(allowanceChange.getId() != null && allowanceRepeatId != null && !"".equals(allowanceRepeatId) && allowanceChange.getStatus() == 0){

cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatId);

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=? and a.status=1", cacheFamily.getId());

AllowanceRepeatChecked allowanceRepeatChecked = super.getDefaultManager().get(AllowanceRepeatChecked.class, allowanceRepeatId);

attachmentList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime,a.id", allowanceRepeatChecked.getAllowanceChange().getId());

//多选框

welfareItemStr = "";

//默认列表

List<WelfareItem> list = super.getDefaultManager().findAll("from WelfareItem a where a.status=?", 1);

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

WelfareItem item = (WelfareItem)list.get(i);

String welfareId = item.getId();

String welfareName = item.getName();

//查询当前编辑人员是否已经享受该项福利

List familyWelfareList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? and a.welfareItem=?", cacheFamily.getId(),item);

if(!"10270004".equals(welfareId) && !"10270005".equals(welfareId) && !"10270009".equals(welfareId) && !"10270010".equals(welfareId) && !"10270011".equals(welfareId) && !"10270012".equals(welfareId)){

if(familyWelfareList.size() > 0){

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' /> " + welfareName+" ";

}else{

welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' /> " + welfareName+" ";

}

}

}

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem w where " + strSql);

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr id='head'><th>姓名\\福利类别</th>");

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

str.append("<th id='"+welfareItemList.get(i).getId()+"'>"+welfareItemList.get(i).getName()+"</th>");

}

str.append("</tr>");

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

CacheFamilyUser cacheFamilyUser = (CacheFamilyUser)cacheFamilyUserList.get(i);

str.append("<tr id='"+cacheFamilyUser.getId()+"'>");

str.append("<td>"+cacheFamilyUser.getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUser.getId());

for(int x = 0; x<welfareItemList.size(); x++){

int flag = 0;

double money = 0.00;

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

String welId = familyUserWelfareList.get(j).getWelfareItem().getId();

if(welId.equals(welfareItemList.get(x).getId())){

flag ++;

money = familyUserWelfareList.get(j).getMonthAmount();

break;

}

}

validatorIds = validatorIds + "," + cacheFamilyUser.getFamily().getId()+welfareItemList.get(x).getId();

if(flag > 0){

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(x).getId()+"'><input onFocus='validator(this.id)' name='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' id='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' value='"+money+"' size='10'/> 元/月</td>");

}else{

str.append("<td align='left' style='text-align:left' id='"+welfareItemList.get(x).getId()+"'><input onFocus='validator(this.id)' name='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' id='"+cacheFamilyUser.getFamilyUser().getId()+welfareItemList.get(x).getId()+"' value='"+money+"' size='10'/> 元/月</td>");

}

}

str.append("</tr>");

}

str.append("</table>");

str.append("<table cellpadding='0' cellspacing='0' class='content\_tab'>");

str.append("<tr>");

str.append("<th width='15%'>添加附件</th>");

str.append("<td colspan='3'>");

str.append("<input type='file' name='fileAttachment' size='50' onchange='addFileAttachment(this)'/>&nbsp;<br>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

welfareStr = str.toString();

//从复审单进入查看变更单

}else if(allowanceChange.getId() != null && allowanceRepeatId != null && !"".equals(allowanceRepeatId) && allowanceChange.getStatus() == 1){

cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatId);

//附件列表

attachmentList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime,a.id", allowanceChange.getId());

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=? and a.status=1", cacheFamily.getId());

//成功变更后的福利展示

familyUserWelfareItemList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.cacheFamily.id=? order by a.familyUser.personRelationType.id,a.cacheFamilyUser.id,a.welfareItem.id", cacheFamily.getId());

//该家庭享受福利项展示

familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? order by a.welfareItem.id", cacheFamily.getId());

//福利展示表格

List<FamilyWelfareItem> familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem f where f.cacheFamily.objectId=? order by f.welfareItem.id", allowanceRepeatId);

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr><th>姓名\\福利类别</th>");

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

str.append("<th>"+familyWelfareItemList.get(i).getWelfareItem().getName()+"</th>");

}

str.append("</tr>");

//该低保单对应的缓存家人列表

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1 order by a.personRelationType.id", allowanceRepeatId);

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

str.append("<tr>");

str.append("<td>"+cacheFamilyUserList.get(i).getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUserList.get(i).getId());

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

str.append("<td align='right' style='text-align:right'>"+familyUserWelfareList.get(j).getMonthAmount()+" 元/月</td>");

}

str.append("</tr>");

}

str.append("</table>");

welfareStr = str.toString();

}

}

protected boolean prepareSave(Element element){

AllowanceChange allowanceChange = (AllowanceChange)element;

if("submit".equals(editFlag)){

if(allowanceChange.getId() == null){

allowanceChange.setFamily(super.getDefaultManager().get(Family.class, familyId));

allowanceChange.setCreateUser((User)getVisit());

}

// allowanceChange.setCreateDate(new Date());

allowanceChange.setCreateDate(getDate(changeEditDate));

//1表示提交，已生效状态

allowanceChange.setStatus(1);

allowanceChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

}else if("save".equals(editFlag)){

if(allowanceChange.getId() == null){

allowanceChange.setFamily(super.getDefaultManager().get(Family.class, familyId));

allowanceChange.setCreateUser((User)getVisit());

}

// allowanceChange.setCreateDate(new Date());

allowanceChange.setCreateDate(getDate(changeEditDate));

//0表示保存，未生效状态

allowanceChange.setStatus(0);

allowanceChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

}else if("editCheck".equals(editFlag)){

allowanceChange.setCreateDate(getDate(changeEditDate));

allowanceChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

}else if("editWelfare".equals(editFlag)){

}

return super.prepareSave(element);

}

public String save(){

AllowanceChange allowanceChange = (AllowanceChange)element;

super.save();

if("submit".endsWith(editFlag)){

//选择福利的id

String sqlStr = "";

if(welfareItemIds != null){

//将福利添加到家庭福利表中

for (String itemId : welfareItemIds) {

sqlStr = sqlStr + " or a.id= " + itemId;

}

}

//allowanceRepeatId 为空，表明是家庭变更，不为空则表明是复审变更

if(allowanceRepeatId == null || "".equals(allowanceRepeatId)){

//查找该变更单是否已经存在缓存家庭

List<CacheFamily> cacheFamilyList = super.getDefaultManager().findAll("from CacheFamily a where a.objectId=?", allowanceChange.getId());

//如果变更没有缓存家庭，则说明是新建复审单，需要添加缓存家庭跟缓存家庭人员

if(cacheFamilyList.size() < 1){

//isAllow为2，表示审核不通过

if(isAllow == 2){

//如果复审不通过，修改该家庭为非低保家庭

super.getDefaultManager().executeSQL("update t\_mz\_family t set t.allowance\_family=0 where t.family\_id=?", allowanceChange.getFamily().getId());

//将该低保单列表中的人员更改为非低保人员

super.getDefaultManager().executeSQL("update t\_mz\_family\_user t set t.allowance\_family\_user=0 where t.family\_id=?", allowanceChange.getFamily().getId());

}else{

//先将家庭成员全部变更为非低保人

super.getDefaultManager().executeSQL("update t\_mz\_family\_user t set t.allowance\_family\_user = 0 where t.family\_id=?", allowanceChange.getFamily().getId());

//再将低保变更单中选中的家庭成员变为低保人

if(checkAllowanceUser != null){

for (String familyUserId : checkAllowanceUser) {

super.getDefaultManager().executeSQL("update t\_mz\_family\_user t set t.allowance\_family\_user=1 where t.family\_user\_id=?", familyUserId);

}

}

}

//添加家庭缓存数据

super.getDefaultManager().executeSQL("insert into t\_mz\_cache\_family"

+" select "

+" SYS\_GUID(),t.family\_id,'"+allowanceChange.getId()+"',t.person\_id,t.family\_head\_name,t.live\_address,t.account\_address,t.user\_quntity,t.total\_income,"

+" t.average\_income,t.status,t.allowance\_family,t.create\_user\_id,t.create\_date,t.update\_user\_id,t.company\_id,t.update\_date,t.description,t.allowance\_and\_person\_type\_id,"+null+" "

+" from "

+" t\_mz\_family t "

+" where t.status = 1 "

+" and t.family\_id='"+allowanceChange.getFamily().getId()+"'");

Object obj = super.getDefaultManager().getObjectSQL("select t.cache\_family\_id from t\_mz\_cache\_family t where t.object\_id=?", allowanceChange.getId());

//添加家庭人员缓存数据

super.getDefaultManager().executeSQL("insert into t\_mz\_cache\_family\_user"

+" select "

+" SYS\_GUID(),'"+(String)obj+"',t.family\_user\_id,t.family\_id,t.person\_relation\_type\_id,t.name,t.sex,t.age,t.nation\_id,t.id\_num,t.income,t.person\_id,"

+" t.identity\_type\_id,t.nature\_id,t.workunit,t.company\_address,t.health\_type\_id,t.account\_type\_id,t.family\_type\_id,"

+" t.home\_address,t.account\_location,"+0+",t.create\_user\_id,t.create\_date,t.update\_user\_id,t.company\_id,t.update\_date,"

+" t.description,t.allowance\_family\_user,t.apply\_success\_date,t.repeat\_date,t.disability,t.disability\_effective\_date,"

+" t.disability\_welfare,t.mobile,t.birthday,t.zip\_code,t.marriage\_id "

+" from "

+" t\_mz\_family\_user t "

+" where t.status = 1"

+" and t.family\_id='"+allowanceChange.getFamily().getId()+"'");

//将低保变更单中选中的家庭成员变为选中状态

if(checkAllowanceUser != null){

for (String familyUserId : checkAllowanceUser) {

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user t set t.status=1 where t.family\_user\_id=? and t.cache\_family\_id = (select a.cache\_family\_id from t\_mz\_cache\_family a where a.object\_id=?)",familyUserId,allowanceChange.getId());

}

}

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1", allowanceChange.getId());

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

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

}else if(isAllow == 2){

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

familyUserWelfareItem.setStatus(1);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceChange.getFamily().getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceChange.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//保存家庭变更记录

FamilyChange familyChange = new FamilyChange();

familyChange.setFamily(allowanceChange.getFamily());

//设置缓存家庭

CacheFamily cacheFamily = (CacheFamily) super.getDefaultManager().getObject("from CacheFamily c where c.objectId=?", allowanceChange.getId());

familyChange.setCacheFamily(cacheFamily);

//设置变更类别为申请类别

FamilyChangeType familyChangeType = super.getDefaultManager().get(FamilyChangeType.class, "1002");

familyChange.setFamilyChangeType(familyChangeType);

//设置申请单编号

familyChange.setAllowanceChange(allowanceChange);

familyChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

familyChange.setStatus(1);

super.getDefaultManager().save(familyChange);

//将最后一次变更对应的家庭福利变为有效状态

FamilyChange fc = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", fc.getCacheFamily().getId());

}else{

//将缓存家庭人员状态变为0

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user a set a.status=0 where a.cache\_family\_id in"

+" (select b.cache\_family\_id from t\_mz\_cache\_family b where b.object\_id=?) ", allowanceChange.getId());

String cacheFamilyUserIds = "";

//将申请单中选中的家庭成员变为选中状态

if(checkAllowanceUser != null){

for (String cacheFamilyUserId : checkAllowanceUser) {

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user t set t.status=1 where t.cache\_family\_user\_id=?",cacheFamilyUserId);

cacheFamilyUserIds = cacheFamilyUserIds + "," + cacheFamilyUserId;

}

}

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceChange.getId());

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

//删除之前保存的人员福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceChange.getFamily().getId());

CacheFamily ca = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", element.getId());

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_welfare\_item a where a.cache\_family\_id=?", ca.getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceChange.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//保存家庭变更记录

FamilyChange familyChange = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

familyChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

super.getDefaultManager().save(familyChange);

//将最后一次变更对应的家庭福利变为有效状态

FamilyChange fc = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", fc.getCacheFamily().getId());

}

}else{//当allowanceRepeatId 不为空时，说明是复审

//复审单

AllowanceRepeatChecked allowanceRepeatChecked = super.getDefaultManager().get(AllowanceRepeatChecked.class,allowanceRepeatId);

//如果该低保复审单没有家庭福利则需要添加

if(allowanceRepeatChecked.getAllowanceChange() == null){

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1", allowanceRepeatId);

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

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//当isAllow为1时说明复审通过，为2时说明复审未通过

if(isAllow == 1){

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

}else if(isAllow == 2){

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

familyUserWelfareItem.setStatus(1);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceRepeatChecked.getFamily().getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatChecked.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//当isAllow为1时说明复审通过，为2时说明复审未通过

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//更新低保复审中的低保变更编号

allowanceRepeatChecked.setAllowanceChange(allowanceChange);

super.getDefaultManager().save(allowanceRepeatChecked);

//将最后一次变更对应的家庭福利变为有效状态

CacheFamily caf = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatChecked.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", caf);

}else{//当该低保复审单已经存在福利时

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceRepeatId);

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

//删除之前保存的人员福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceRepeatChecked.getFamily().getId());

CacheFamily ca = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatId);

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_welfare\_item a where a.cache\_family\_id=?", ca.getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where a.id=10270001 or a.id=10270002 "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatId);

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//将最后一次变更对应的家庭福利变为有效状态

CacheFamily caf = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatChecked.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", caf);

}

}

}else if("save".equals(editFlag)){

//选择福利的id

String sqlStr = " ";

if(welfareItemIds != null){

int i = 0;

//将福利添加到家庭福利表中

for (String itemId : welfareItemIds) {

i++;

if(i == 1){

sqlStr = sqlStr + " a.id= " + itemId;

}else{

sqlStr = sqlStr + " or a.id= " + itemId;

}

}

}

//allowanceRepeatId 为空，表明是家庭变更，不为空则表明是复审变更

if(allowanceRepeatId == null || "".equals(allowanceRepeatId)){

//查找该变更单是否已经存在缓存家庭

List<CacheFamily> cacheFamilyList = super.getDefaultManager().findAll("from CacheFamily a where a.objectId=?", allowanceChange.getId());

//如果变更没有缓存家庭，则说明是新建复审单，需要添加缓存家庭跟缓存家庭人员

if(cacheFamilyList.size() < 1){

//添加家庭缓存数据

super.getDefaultManager().executeSQL("insert into t\_mz\_cache\_family"

+" select "

+" SYS\_GUID(),t.family\_id,'"+allowanceChange.getId()+"',t.person\_id,t.family\_head\_name,t.live\_address,t.account\_address,t.user\_quntity,t.total\_income,"

+" t.average\_income,t.status,t.allowance\_family,t.create\_user\_id,t.create\_date,t.update\_user\_id,t.company\_id,t.update\_date,t.description,t.allowance\_and\_person\_type\_id,"+null+" "

+" from "

+" t\_mz\_family t "

+" where t.status = 1 "

+" and t.family\_id='"+allowanceChange.getFamily().getId()+"'");

Object obj = super.getDefaultManager().getObjectSQL("select t.cache\_family\_id from t\_mz\_cache\_family t where t.object\_id=?", allowanceChange.getId());

//添加家庭人员缓存数据

super.getDefaultManager().executeSQL("insert into t\_mz\_cache\_family\_user"

+" select "

+" SYS\_GUID(),'"+(String)obj+"',t.family\_user\_id,t.family\_id,t.person\_relation\_type\_id,t.name,t.sex,t.age,t.nation\_id,t.id\_num,t.income,t.person\_id,"

+" t.identity\_type\_id,t.nature\_id,t.workunit,t.company\_address,t.health\_type\_id,t.account\_type\_id,t.family\_type\_id,"

+" t.home\_address,t.account\_location,"+0+",t.create\_user\_id,t.create\_date,t.update\_user\_id,t.company\_id,t.update\_date,"

+" t.description,t.allowance\_family\_user,t.apply\_success\_date,t.repeat\_date,t.disability,t.disability\_effective\_date,"

+" t.disability\_welfare,t.mobile,t.birthday,t.zip\_code,t.marriage\_id "

+" from "

+" t\_mz\_family\_user t "

+" where t.status = 1"

+" and t.family\_id='"+allowanceChange.getFamily().getId()+"'");

//将低保变更单中选中的家庭成员变为选中状态

if(checkAllowanceUser != null){

for (String familyUserId : checkAllowanceUser) {

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user t set t.status=1 where t.family\_user\_id=? and t.cache\_family\_id = (select a.cache\_family\_id from t\_mz\_cache\_family a where a.object\_id=?)",familyUserId,allowanceChange.getId());

}

}

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceChange.getId());

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

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceChange.getFamily().getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceChange.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//将最后一次变更对应的家庭福利变为有效状态

List<FamilyChange> familyChangeList = super.getDefaultManager().findAll("from FamilyChange t where t.family.id=? order by t.changeDate desc", allowanceChange.getFamily().getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", familyChangeList.get(0).getCacheFamily().getId());

//保存家庭变更记录

FamilyChange familyChange = new FamilyChange();

familyChange.setFamily(allowanceChange.getFamily());

//设置缓存家庭

CacheFamily cacheFamily = (CacheFamily) super.getDefaultManager().getObject("from CacheFamily c where c.objectId=?", allowanceChange.getId());

familyChange.setCacheFamily(cacheFamily);

//设置变更类别为申请类别

FamilyChangeType familyChangeType = super.getDefaultManager().get(FamilyChangeType.class, "1002");

familyChange.setFamilyChangeType(familyChangeType);

//设置申请单编号

familyChange.setAllowanceChange(allowanceChange);

familyChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

familyChange.setStatus(1);

super.getDefaultManager().save(familyChange);

}else{

//将缓存家庭人员状态变为0

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user a set a.status=0 where a.cache\_family\_id in"

+" (select b.cache\_family\_id from t\_mz\_cache\_family b where b.object\_id=?) ", allowanceChange.getId());

//将申请单中选中的家庭成员变为选中状态

if(checkAllowanceUser != null){

for (String familyUserId : checkAllowanceUser) {

super.getDefaultManager().executeSQL("update t\_mz\_cache\_family\_user t set t.status=1 where t.cache\_family\_user\_id=?",familyUserId);

}

}

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceChange.getId());

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

//删除之前保存的人员福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceChange.getFamily().getId());

CacheFamily ca = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", element.getId());

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_welfare\_item a where a.cache\_family\_id=?", ca.getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceChange.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//修改家庭变更记录

FamilyChange familyChange = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

familyChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

super.getDefaultManager().save(familyChange);

//将最后一次变更对应的家庭福利变为有效状态

FamilyChange fc = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", fc.getCacheFamily().getId());

}

}else{//当allowanceRepeatId 不为空时，说明是复审

//复审单

AllowanceRepeatChecked allowanceRepeatChecked = super.getDefaultManager().get(AllowanceRepeatChecked.class,allowanceRepeatId);

//allowanceRepeatChecked.getAllowanceChange() 为空，表明该复审单没有填写福利变更单

if(allowanceRepeatChecked.getAllowanceChange() == null){

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceRepeatId);

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

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//当isAllow为1时说明复审通过，为2时说明复审未通过

if(isAllow == 1){

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

}else if(isAllow == 2){

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceRepeatChecked.getFamily().getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatChecked.getId());

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(0);

//当isAllow为1时说明复审通过，为2时说明复审未通过

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//更新低保复审中的低保变更编号

allowanceRepeatChecked.setAllowanceChange(allowanceChange);

super.getDefaultManager().save(allowanceRepeatChecked);

//将最后一次变更对应的家庭福利变为有效状态

CacheFamily caf = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatChecked.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", caf);

}else{//该低保单复审单已经存在福利

//添加人员福利

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=?", allowanceRepeatId);

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

//删除之前保存的人员福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyUserWelfareItem familyUserWelfareItem = new FamilyUserWelfareItem();

familyUserWelfareItem.setFamilyUser(cacheFamilyUserList.get(i).getFamilyUser());

familyUserWelfareItem.setCacheFamilyUser(cacheFamilyUserList.get(i));

familyUserWelfareItem.setWelfareItem(welfareItemList.get(j));

//从页面上获取金额

String str = cacheFamilyUserList.get(i).getFamilyUser().getId()+welfareItemList.get(j).getId();

String money = super.getRequest().getParameter(str);

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

familyUserWelfareItem.setTotalAmount((double)0);

//保存时状态为0，未生效，提交后状态为1，生效

familyUserWelfareItem.setStatus(0);

super.getDefaultManager().save(familyUserWelfareItem);

}

}

//添加家庭福利，先将之前的福利全部变为无效状态

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item a set a.status=0 where a.family\_id=?", allowanceRepeatChecked.getFamily().getId());

CacheFamily ca = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatId);

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_welfare\_item a where a.cache\_family\_id=?", ca.getId());

List<WelfareItem> welfareItemList = super.getDefaultManager().findAll("from WelfareItem a where "+sqlStr+" order by a.id");

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

FamilyWelfareItem familyWelfareItem = new FamilyWelfareItem();

familyWelfareItem.setFamily(allowanceChange.getFamily());

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatId);

familyWelfareItem.setCacheFamily(cacheFamily);

familyWelfareItem.setWelfareItem(welfareItemList.get(j));

familyWelfareItem.setStatus(1);

//该isAllow的值是从低保变更表单提交到action的值，当isAllow为1时说明继续享受低保，为2时说明不再享受低保

if(isAllow == 1){

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.id=? and t.status=1 and t.welfareItem.id=?",cacheFamily.getId(), welfareItemList.get(j).getId());

if(sum != null){

familyWelfareItem.setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.setTotalAmount(0.00);

}

}else if(isAllow == 2){

familyWelfareItem.setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem);

}

//将最后一次变更对应的家庭福利变为有效状态

CacheFamily caf = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", allowanceRepeatChecked.getId());

super.getDefaultManager().executeSQL("update t\_mz\_family\_welfare\_item t set t.status=1 where t.cache\_family\_id=?", caf);

}

}

}else if("editCheck".equals(editFlag)){

FamilyChange familyChange = new FamilyChange();

//allowanceRepeatId长度小于1，表示家庭变更单进行修改，否则是低保复审单对应的变更单进行修改

if(allowanceRepeatId.length() < 1){

//修改家庭变更记录

familyChange = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceChange.id=?", allowanceChange.getId());

}else{

familyChange = (FamilyChange)super.getDefaultManager().getObject("from FamilyChange t where t.allowanceRepeatChecked.id=?", allowanceRepeatId);

}

//当复审没有填写完福利变更单，但是流程没有完成，则不更新家庭变更记录

if(familyChange != null){

familyChange.setChangeDate(new java.sql.Date(changeDate.getTime()));

super.getDefaultManager().save(familyChange);

}

}else if("editWelfare".equals(editFlag)){

AllowanceRepeatChecked allowanceRepeat = (AllowanceRepeatChecked)super.getDefaultManager().getObject("from AllowanceRepeatChecked t where t.allowanceChange.id=?", allowanceChange.getId());

String objectId = "";

//如果allowanceRepeat.getAllowanceChange() == null,则表明修改家庭变更福利，否则，是修改低保复审的福利

if(allowanceRepeat == null){

objectId = allowanceChange.getId();

}else{

objectId = allowanceRepeat.getId();

}

familyUserWelfareItemList = super.getDefaultManager().findAll("from FamilyUserWelfareItem f where f.cacheFamilyUser.cacheFamily.objectId=? order by f.familyUser.personRelationType.id,f.welfareItem.id",objectId);

//修改家庭人员福利

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

FamilyUserWelfareItem familyUserWelfareItem = (FamilyUserWelfareItem)familyUserWelfareItemList.get(i);

//从页面上获取金额

String money = super.getRequest().getParameter(familyUserWelfareItem.getId());

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

familyUserWelfareItem.setMonthAmount(Double.parseDouble(money));

}else{

familyUserWelfareItem.setMonthAmount(0.00);

}

super.getDefaultManager().save(familyUserWelfareItem);

}

//修改家庭福利(低保金和粮油补助)

List<FamilyWelfareItem> familyWelfareItem = super.getDefaultManager().findAll("from FamilyWelfareItem w where w.cacheFamily.objectId=?",objectId);

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

Object sum = super.getDefaultManager().getObject("select sum(t.monthAmount) from FamilyUserWelfareItem t where t.cacheFamilyUser.cacheFamily.objectId=? and t.status=1 and t.welfareItem.id=?",objectId, familyWelfareItem.get(i).getWelfareItem().getId());

if(sum != null){

familyWelfareItem.get(i).setTotalAmount(Double.parseDouble(sum.toString()));

}else{

familyWelfareItem.get(i).setTotalAmount(0.00);

}

super.getDefaultManager().save(familyWelfareItem.get(i));

}

}

this.setId(element.getId());

// 添加附件

if (fileAttachment != null)

this.addAttachment(allowanceChange);

return SUCCESS;

}

protected Class getElementClass()

{

return AllowanceChange.class;

}

//添加附件

public String addAttachment(AllowanceChange allowanceChange)

{

//传入DAO

attachmentHelper.init(super.getDefaultManager());

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

{

AttachmentValue value = new AttachmentValue();

value.setContentType(fileAttachmentContentType[i]);

value.setFileName(fileAttachmentFileName[i]);

value.setType("ALLOWANCE\_CHANGE\_ATTACHMENT");

value.setName(fileAttachmentFileName[i].substring(0,fileAttachmentFileName[i].lastIndexOf(".")));

value.setSize(fileAttachment[i].length());

value.setUploadFile(fileAttachment[i]);

value.setObjectId(id);

value.setRealName(System.currentTimeMillis() + (serial++) + "." + attachmentHelper.getFileExtName(value.getFileName()) );

value.setCreateTime(new Date());

value.setCreateUser((User)getVisit());

attachmentHelper.saveAttachment(value);

}

return SUCCESS;

}

private Date getDate(Date date){

//获得当前时间Sring类型的时分秒

String sfm = getCurrentTime();

sfm = sfm.substring(sfm.length() - 8, sfm.length());

String ymd = DateToString(date);

String d = ymd + " " + sfm;

Date da = StingToDate(d);

return da;

}

/\*\*

\* 将Date 类型 格式的日期转换成 String 类型 yyyy-MM-dd HH:mm:ss

\*

\* @param param

\* @return

\*/

private static String DateToString(Date param) {

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

return sdf.format(param);

}

/\*\*

\* 将String 类型 格式的日期转换成 Date 类型

\*

\* @param param

\* @return

\*/

private static Date StingToDate(String param) {

Date date = null;

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

try {

date = sdf.parse(param);

} catch (ParseException ex) {

}

return date;

}

//返回当前时间的String类型

private static String getCurrentTime() {

return getCurrentTime("yyyy-MM-dd HH:mm:ss");

}

private static String getCurrentTime(String pattern) {

SimpleDateFormat sdf = new SimpleDateFormat(pattern);

return sdf.format(Calendar.getInstance().getTime());

}

public String getFamilyId() {

return familyId;

}

public void setFamilyId(String familyId) {

this.familyId = familyId;

}

public Date getChangeDate() {

return changeDate;

}

public List<FamilyUser> getFamilyUserList() {

return familyUserList;

}

public void setChangeDate(Date changeDate) {

this.changeDate = changeDate;

}

public void setFamilyUserList(List<FamilyUser> familyUserList) {

this.familyUserList = familyUserList;

}

public AttachmentHelper getAttachmentHelper() {

return attachmentHelper;

}

public File[] getFileAttachment() {

return fileAttachment;

}

public String[] getFileAttachmentContentType() {

return fileAttachmentContentType;

}

public String[] getFileAttachmentFileName() {

return fileAttachmentFileName;

}

public List<AttachmentValue> getAttachmentList() {

return attachmentList;

}

public void setAttachmentHelper(AttachmentHelper attachmentHelper) {

this.attachmentHelper = attachmentHelper;

}

public void setFileAttachment(File[] fileAttachment) {

this.fileAttachment = fileAttachment;

}

public void setFileAttachmentContentType(String[] fileAttachmentContentType) {

this.fileAttachmentContentType = fileAttachmentContentType;

}

public void setFileAttachmentFileName(String[] fileAttachmentFileName) {

this.fileAttachmentFileName = fileAttachmentFileName;

}

public void setAttachmentList(List<AttachmentValue> attachmentList) {

this.attachmentList = attachmentList;

}

public Family getFamily() {

return family;

}

public void setFamily(Family family) {

this.family = family;

}

public CacheFamily getCacheFamily() {

return cacheFamily;

}

public void setCacheFamily(CacheFamily cacheFamily) {

this.cacheFamily = cacheFamily;

}

public String[] getCheckAllowanceUser() {

return checkAllowanceUser;

}

public void setCheckAllowanceUser(String[] checkAllowanceUser) {

this.checkAllowanceUser = checkAllowanceUser;

}

public List<FamilyUserWelfareItem> getFamilyUserWelfareItemList() {

return familyUserWelfareItemList;

}

public void setFamilyUserWelfareItemList(

List<FamilyUserWelfareItem> familyUserWelfareItemList) {

this.familyUserWelfareItemList = familyUserWelfareItemList;

}

public List<CacheFamilyUser> getCacheFamilyUserList() {

return cacheFamilyUserList;

}

public String getAllowanceRepeatId() {

return allowanceRepeatId;

}

public void setAllowanceRepeatId(String allowanceRepeatId) {

this.allowanceRepeatId = allowanceRepeatId;

}

public void setCacheFamilyUserList(List cacheFamilyUserList) {

this.cacheFamilyUserList = cacheFamilyUserList;

}

public int getIsAllow() {

return isAllow;

}

public void setIsAllow(int isAllow) {

this.isAllow = isAllow;

}

public Date getChangeEditDate() {

return changeEditDate;

}

public void setChangeEditDate(Date changeEditDate) {

this.changeEditDate = changeEditDate;

}

public String getWelfareItemStr() {

return welfareItemStr;

}

public void setWelfareItemStr(String welfareItemStr) {

this.welfareItemStr = welfareItemStr;

}

public String[] getWelfareItemIds() {

return welfareItemIds;

}

public void setWelfareItemIds(String[] welfareItemIds) {

this.welfareItemIds = welfareItemIds;

}

public List<FamilyWelfareItem> getFamilyWelfareItemList() {

return familyWelfareItemList;

}

public void setFamilyWelfareItemList(

List<FamilyWelfareItem> familyWelfareItemList) {

this.familyWelfareItemList = familyWelfareItemList;

}

public String getWelfareStr() {

return welfareStr;

}

public void setWelfareStr(String welfareStr) {

this.welfareStr = welfareStr;

}

public String getValidatorIds() {

return validatorIds;

}

public String getEditFlag() {

return editFlag;

}

public void setEditFlag(String editFlag) {

this.editFlag = editFlag;

}

public String getFamilyUserIds() {

return familyUserIds;

}

public void setFamilyUserIds(String familyUserIds) {

this.familyUserIds = familyUserIds;

}

public String getAllowanceTypeId() {

return allowanceTypeId;

}

public String getPersonTypeId() {

return personTypeId;

}

public void setAllowanceTypeId(String allowanceTypeId) {

this.allowanceTypeId = allowanceTypeId;

}

public void setPersonTypeId(String personTypeId) {

this.personTypeId = personTypeId;

}

public List<AllowanceAndPersonType> getAllowanceTypeList() {

return allowanceTypeList;

}

public List<AllowanceAndPersonType> getPersonTypeList() {

return personTypeList;

}

public void setAllowanceTypeList(List<AllowanceAndPersonType> allowanceTypeList) {

this.allowanceTypeList = allowanceTypeList;

}

public void setPersonTypeList(List<AllowanceAndPersonType> personTypeList) {

this.personTypeList = personTypeList;

}

}

package com.bizlogic.mz.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import net.sf.json.JSONObject;

import org.apache.commons.lang.StringUtils;

import com.bizlogic.base.AttachmentValue;

import com.bizlogic.base.Company;

import com.bizlogic.base.User;

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

import com.bizlogic.common.Element;

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

import com.bizlogic.mz.bean.AllowanceAndPersonType;

import com.bizlogic.mz.bean.AllowanceChange;

import com.bizlogic.mz.bean.AllowanceRepeatChecked;

import com.bizlogic.mz.bean.CacheFamily;

import com.bizlogic.mz.bean.CacheFamilyUser;

import com.bizlogic.mz.bean.FamilyUser;

import com.bizlogic.mz.bean.FamilyUserWelfareItem;

import com.bizlogic.mz.bean.FamilyWelfareItem;

public class AllowanceChangeListAction extends BaseUserListAction{

//选择查看低保单选择条件

private String searchAllowanceStatus = "wait";

//身份证号查询人员

private String searchIdNumber;

//查看待审核、已审核通过低保单列表

private List<Element> selectAllowanceChangeList;

public List<Element> getSelectAllowanceChangeList() {

return getProcessStatusList();

}

@Override

protected long doGetCount(ListInfo listInfo) {

sqlCondition = this.getConditions(searchName,searchAllowanceStatus);

return super.getDefaultManager().getCount("from AllowanceChange a "

+" where "

+" a.id not in (select b.allowanceChange.id from AllowanceRepeatChecked b where b.allowanceChange is not null) "

+sqlCondition);

}

@Override

protected List doSearch(ListInfo listInfo) {

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

list = super.getDefaultManager().find("from AllowanceChange a "

+" where "

+" a.id not in (select b.allowanceChange.id from AllowanceRepeatChecked b where b.allowanceChange is not null) "

+sqlCondition,

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

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

List l = super.getDefaultManager().findAll("from FamilyUser a where a.family=? and a.status=1 order by a.personRelationType.id ", list.get(i).getFamily());

AllowanceRepeatChecked allowanceRepeatChecked = (AllowanceRepeatChecked)super.getDefaultManager().getObject("from AllowanceRepeatChecked t where t.allowanceChange.id=?", list.get(i).getId());

String familyUserStr = "";

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

FamilyUser u = (FamilyUser)l.get(j);

familyUserStr += u.getName()+";&nbsp;&nbsp;";

}

list.get(i).setFamilyUserStr(familyUserStr);

}

return list;

}

/\*\*

\* 获得查询条件

\*/

protected String getConditions(String name,String allowanceChangeStatus)

{

String conditionsStr = "";

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

name = name.replaceAll(" ", "");

conditionsStr = conditionsStr+ " and a.family.id in (select a.family.id from FamilyUser a where a.name like '%"+ name.trim() +"%'" + ") ";

}

//获取登陆人

User visitUser = (User)getVisit();

Company parent = visitUser.getCompany().getParent();

//社区人员登录

if(parent != null){

String companyId = visitUser.getCompany().getId();

conditionsStr = conditionsStr + " and a.status=1 and a.family.company.id='" + companyId + "' ";

}else{

if("wait".equals(allowanceChangeStatus)){

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

}else if("success".equals(allowanceChangeStatus)){

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

}

}

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");

//修改变更单内容

if(StringUtils.equals(act, "edit\_allowance\_change\_content")){

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

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

List<AttachmentValue> attachmentList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime,a.id", allowanceChangeId);

AllowanceChange allowanceChange = super.getDefaultManager().get(AllowanceChange.class, allowanceChangeId);

StringBuffer str = new StringBuffer();

//修改审核内容

if("checkContent".equals(grade)){

// //福利多选框

// String welfareItemStr = "";

// //默认列表

// List<WelfareItem> list = super.getDefaultManager().findAll("from WelfareItem a where a.status=?", 1);

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

// WelfareItem item = (WelfareItem)list.get(i);

// String welfareId = item.getId();

// String welfareName = item.getName();

// //查询当前编辑人员是否已经享受该项福利

// List familyWelfareList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.objectId=? and a.welfareItem=?", allowanceChangeId,item);

// if(!"10270004".equals(welfareId) && !"10270005".equals(welfareId) && !"10270009".equals(welfareId) && !"10270010".equals(welfareId) && !"10270011".equals(welfareId) && !"10270012".equals(welfareId)){

// if(familyWelfareList.size() > 0){

// //低保金和粮油补助默认选中不可更改

// if("10270001".equals(welfareId) || "10270002".equals(welfareId)){

// welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' disabled/> " + welfareName+" ";

// }else{

// welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' /> " + welfareName+" ";

// }

// }else{

// //低保金和粮油补助默认选中不可更改

// if("10270001".equals(welfareId) || "10270002".equals(welfareId)){

// welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' checked='checked' disabled/> " + welfareName+" ";

// }else{

// welfareItemStr += "<input type='checkbox' name='welfareItemIds' value='" + welfareId + "' /> " + welfareName+" ";

// }

// }

// }

// }

//

str.append(grade+",<table cellpadding='0' cellspacing='0' class='content\_tab'>");

// str.append("<tr>");

// str.append("<th>选择福利<font color='red'>\*</font></th>");

// str.append("<td>");

// str.append(welfareItemStr);

// str.append("</td>");

// str.append("</tr>");

str.append("<tr>");

str.append("<th>生效日期<font color='red'>\*</font></th>");

str.append("<td>");

str.append("<input type='text' name='changeDate' value='"+parse(allowanceChange.getChangeDate())+"' readonly='readonly' id='changeDate' class='Wdate' onclick='WdatePicker();'/>");

str.append("</td>");

str.append("</tr>");

str.append("<tr>");

str.append("<th>办理日期<font color='red'>\*</font></th>");

str.append("<td>");

str.append("<input type='text' name='changeEditDate' value='"+parse(allowanceChange.getCreateDate())+"' readonly='readonly' id='changeEditDate' class='Wdate' onclick='WdatePicker();'/>");

str.append("</td>");

str.append("</tr>");

str.append("<tr>");

str.append("<th width='15%'>审核意见<font color='red'>\*</font></th>");

str.append("<td colspan='3'>");

str.append("<textarea name='element.description' cols='80' rows='5' id='description'>"+allowanceChange.getDescription()+"</textarea></td>");

str.append("</tr>");

str.append("<tr>");

str.append("<th width='15%'>附件列表</th>");

str.append("<td colspan='3'>");

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

str.append("<div class='delete\_pic' id='"+attachmentList.get(i).getId()+"'>");

str.append("<img src='../images/folders.png' title='模板附件'>");

str.append("<a href='attachmentDownload.action?id="+attachmentList.get(i).getId()+"' target='blank'>"+attachmentList.get(i).getName()+".jpg</a>");

str.append("<img src='../images/no\_passbtn.gif' title='删除' onclick='javascript:deleteAttachmentById("+'"' + attachmentList.get(i).getId() +'"'+");return false;'/>");

str.append("</div><br>");

}

str.append("</td>");

str.append("</tr>");

str.append("<tr>");

str.append("<th>添加附件</th>");

str.append("<td colspan='3'>");

str.append("<input type='file' name='fileAttachment' size='50' value='' id='theForm\_fileAttachment' onchange='EditAddFileAttachment(this)'/><br>");

str.append("</td>");

str.append("</tr>");

str.append("<tr>");

str.append("<td colspan='4'>");

str.append("<p class='up\_btn'>");

str.append("<button onclick='javascript:editSave("+'"' + "editCheck" +'"'+");return false;'>保存修改</button>");

str.append("<button onclick='window.location.href=window.location.href;return false;'>返回</button>");

str.append("</p>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

res.put("message", str.toString());

}else if("welfareContent".equals(grade)){

String validatorIds = "";

AllowanceRepeatChecked allowanceRepeat = (AllowanceRepeatChecked)super.getDefaultManager().getObject("from AllowanceRepeatChecked t where t.allowanceChange.id=?", allowanceChangeId);

List<FamilyWelfareItem> familyWelfareItemList = new ArrayList<FamilyWelfareItem>();

List<CacheFamilyUser> cacheFamilyUserList = new ArrayList<CacheFamilyUser>();

//如果allowanceRepeat.getAllowanceChange() == null,则表明修改家庭变更福利，否则，是修改低保复审的福利

if(allowanceRepeat == null){

familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem f where f.cacheFamily.objectId=? order by f.welfareItem.id", allowanceChangeId);

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1 order by a.personRelationType.id", allowanceChangeId);

}else{

familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem f where f.cacheFamily.objectId=? order by f.welfareItem.id", allowanceRepeat.getId());

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1 order by a.personRelationType.id", allowanceRepeat.getId());

}

str.append("welfareContent,<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr><th>姓名\\福利类别</th>");

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

str.append("<th>"+familyWelfareItemList.get(i).getWelfareItem().getName()+"</th>");

}

str.append("</tr>");

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

str.append("<tr>");

str.append("<td>"+cacheFamilyUserList.get(i).getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUserList.get(i).getId());

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

validatorIds = validatorIds + "," + familyUserWelfareList.get(j).getId();

str.append("<td align='left' style='text-align:left'><input name='"+familyUserWelfareList.get(j).getId()+"' id='"+familyUserWelfareList.get(j).getId()+"' value='"+familyUserWelfareList.get(j).getMonthAmount()+"' size='12'/> 元/月</td>");

}

str.append("</tr>");

}

str.append("<tr>");

str.append("<td colspan='4'>");

str.append("<p class='up\_btn'>");

str.append("<button onclick='javascript:editSave("+'"' + "editWelfare" +'"'+");return false;'>保存修改</button>");

str.append("<button onclick='window.location.href=window.location.href;return false;'>返回</button>");

str.append("</p>");

str.append("</td>");

str.append("</tr>");

str.append("</table>");

res.put("message", str.toString());

}

}

//根据低保类别查找人员类别

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

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

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

if (StringUtils.isNotBlank(allowanceTypeId)) {

dl = super.getDefaultManager().findAll("from AllowanceAndPersonType u where u.parent.id=? ",allowanceTypeId);

}

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();

}

public String delete() {

String[] ids = getSelectedIds();

if (ids != null) {

for (String aid : ids) {

AllowanceChange allowanceChange = super.getDefaultManager().get(AllowanceChange.class, aid);

//福利变更对应缓存家庭

CacheFamily cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily t where t.objectId=?", aid);

if(cacheFamily != null){

//福利变更对应缓存家庭人员

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser t where t.cacheFamily.id=?", cacheFamily.getId());

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

//删除家庭人员的福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_user\_welfare\_item t where t.cache\_family\_user\_id=?", cacheFamilyUserList.get(i).getId());

//删除缓存人员

super.getDefaultManager().execute("delete from CacheFamilyUser t where t.id=?", cacheFamilyUserList.get(i).getId());

}

//删除家庭福利

super.getDefaultManager().executeSQL("delete from t\_mz\_family\_welfare\_item t where t.cache\_family\_id=?", cacheFamily.getId());

//删除缓存家庭

super.getDefaultManager().execute("delete from CacheFamily t where t.id=?", cacheFamily.getId());

}

//删除变更单

super.getDefaultManager().execute("delete from AllowanceChange t where t.id=?", aid);

}

}

return SUCCESS;

}

public static String parse(Date param) {

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

return sdf.format(param);

}

public List<Element> getProcessStatusList() {

List list = new ArrayList();

Element lt1 = new Element();

Element lt2 = new Element();

lt1.setId("wait");

lt1.setName("待提交");

lt2.setId("success");

lt2.setName("审核完成");

list.add(lt1);

list.add(lt2);

return list;

}

protected Class getImplementClass()

{

return AllowanceChange.class;

}

public String getSearchAllowanceStatus() {

return searchAllowanceStatus;

}

public String getSearchIdNumber() {

return searchIdNumber;

}

public void setSearchAllowanceStatus(String searchAllowanceStatus) {

this.searchAllowanceStatus = searchAllowanceStatus;

}

public void setSearchIdNumber(String searchIdNumber) {

this.searchIdNumber = searchIdNumber;

}

}

package com.bizlogic.mz.action;

import com.bizlogic.common.action.BaseAction;

import com.bizlogic.mz.bean.Family;

import com.bizlogic.mz.bean.FamilyUser;

import com.bizlogic.mz.bean.FamilyWelfareItem;

import com.bizlogic.mz.bean.FamilyWelfareItemLog;

import java.io.ByteArrayInputStream;

import java.io.ByteArrayOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.UnsupportedEncodingException;

import java.math.BigDecimal;

import java.text.SimpleDateFormat;

import java.util.Calendar;

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;

public class AllowanceCivilExportExcelAction extends BaseAction{

private String grantDate;

public String execute() throws Exception

{

return super.execute();

}

public String getFileName() {

try {

return new String(("低保发放表.xls困难.xls").getBytes("gbk"), "ISO8859\_1");

} catch (UnsupportedEncodingException e) {

return "低保发放表.xls困难.xls";

}

}

public InputStream getExcelFile() {

//创建excel工作簿

HSSFWorkbook workbook = new HSSFWorkbook();

//创建一页

HSSFSheet sheet = workbook.createSheet("城镇低保");

//设置列宽

sheet.setColumnWidth(0, 3000);

sheet.setColumnWidth(1, 4000);

sheet.setColumnWidth(2, 6000);

sheet.setColumnWidth(3, 3000);

sheet.setColumnWidth(4, 5000);

sheet.setColumnWidth(5, 5000);

sheet.setColumnWidth(6, 4000);

sheet.setColumnWidth(7, 4000);

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();

//设置粗体

font1.setBoldweight((short) 12);

//字体颜色

font1.setColor((short) 00000);

//字体大小

font1.setFontHeightInPoints((short) 16);

font1.setFontName("宋体");

style1.setFont(font1);

/\*\*

\* 签字、制表日志样式

\*/

HSSFCellStyle style2 = workbook.createCellStyle();

//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 font2 = workbook.createFont();

//设置粗体

font2.setBoldweight((short) 12);

//字体颜色

font2.setColor((short) 00000);

//字体大小

font2.setFontHeightInPoints((short) 12);

font2.setFontName("宋体");

style2.setFont(font2);

List list = super.getDefaultManager().findAll("from Family a where a.status=1 and a.allowanceFamily=1 and a.allowanceAndPersonType.parent.id=10141002 ");

//设置合并单元格（0行到0行，0列到8列进行合并）== 标题

sheet.addMergedRegion(new CellRangeAddress(0,0,0,7));

//设置合并单元格（0行到0行，0列到8列进行合并）== 合计

sheet.addMergedRegion(new CellRangeAddress(list.size()+3,list.size()+3,0,1));

//设置合并单元格（0行到0行，0列到8列进行合并）== 注解

sheet.addMergedRegion(new CellRangeAddress(list.size()+4,list.size()+4,0,7));

//获取当前年月

String date = getCurrentDate();

String year = date.substring(0, 4);

String month = date.substring(date.indexOf("-")+1, date.length());

int y = new Integer(year);

int m = new Integer(month);

if(m < 12){

m = m + 1;

}else{

y = y + 1;

m = 1;

}

//第0行

HSSFRow row0 = sheet.createRow(0);

//0行0列值

HSSFCell cell0 = row0.createCell(0);

cell0.setCellValue("城镇低保人员"+y+"年"+m+"月生活补助发放表");

cell0.setCellStyle(style1);

//第一行

HSSFRow row1 = sheet.createRow(1);

HSSFCell cell1 = row1.createCell(0);

cell1.setCellValue("单位名称：");

cell1.setCellStyle(style0);

cell1 = row1.createCell(1);

cell1.setCellValue("");

cell1.setCellStyle(style0);

cell1 = row1.createCell(2);

cell1.setCellValue("亚运村");

cell1.setCellStyle(style0);

cell1 = row1.createCell(3);

cell1.setCellValue("");

cell1.setCellStyle(style0);

cell1 = row1.createCell(4);

cell1.setCellValue("");

cell1.setCellStyle(style0);

cell1 = row1.createCell(5);

cell1.setCellValue("");

cell1.setCellStyle(style0);

cell1 = row1.createCell(6);

cell1.setCellValue("资金来源：");

cell1.setCellStyle(style0);

cell1 = row1.createCell(7);

cell1.setCellValue("财政拨款");

cell1.setCellStyle(style0);

//第二行

HSSFRow row2 = sheet.createRow(2);

HSSFCell cell2 = row2.createCell(0);

cell2.setCellValue("编号");

cell2.setCellStyle(style0);

cell2 = row2.createCell(1);

cell2.setCellValue("持证人姓名");

cell2.setCellStyle(style0);

cell2 = row2.createCell(2);

cell2.setCellValue("人员类别");

cell2.setCellStyle(style0);

cell2 = row2.createCell(3);

cell2.setCellValue("人口数");

cell2.setCellStyle(style0);

cell2 = row2.createCell(4);

cell2.setCellValue("保障金（元）");

cell2.setCellStyle(style0);

cell2 = row2.createCell(5);

cell2.setCellValue("粮油帮困（元）");

cell2.setCellStyle(style0);

cell2 = row2.createCell(6);

cell2.setCellValue("合计");

cell2.setCellStyle(style0);

cell2 = row2.createCell(7);

cell2.setCellValue("领款人签字");

cell2.setCellStyle(style0);

// grantDate = getCurrentDate();

grantDate = "2013-05";

BigDecimal total = new BigDecimal(0.00);

double money = 0.00;

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

Family family = (Family)list.get(i);

FamilyUser familyUser = (FamilyUser)super.getDefaultManager().getObject("from FamilyUser t " +

" where " +

" t.personRelationType.id=10142001 " +

" and t.status=1 " +

" and t.family.id=?", family.getId());

//有效低保金

FamilyWelfareItem familyWelfareItem1 = (FamilyWelfareItem)super.getDefaultManager().getObject(

" from FamilyWelfareItem t " +

" where t.family.id=? " +

" and t.status=1 " +

" and t.welfareItem.id=10270001 ", familyUser.getFamily().getId());

//有效粮油补贴

FamilyWelfareItem familyWelfareItem2 = (FamilyWelfareItem)super.getDefaultManager().getObject(

"from FamilyWelfareItem t " +

" where t.family.id=? " +

" and t.status=1 " +

" and t.welfareItem.id=10270002 ", familyUser.getFamily().getId());

FamilyWelfareItem familyWelfareItem3 = new FamilyWelfareItem();

//其他

familyWelfareItem3 = (FamilyWelfareItem)super.getDefaultManager().getObject(

"from FamilyWelfareItem t " +

" where " +

" t.family.id=? " +

" and t.status=1 " +

" and t.welfareItem.id=10279999 ", familyUser.getFamily().getId());

//获取行

HSSFRow row = sheet.getRow(i+3);

//如果没有是空，进行创建

if(row == null )

row = sheet.createRow(i+3);

//编号

HSSFCell cell = row.createCell(0);

cell.setCellStyle(style0);

cell.setCellValue(i+1);

//持证人姓名

cell = row.createCell(1);

cell.setCellStyle(style0);

if(familyUser.getName() != null){

cell.setCellValue(familyUser.getName());

}else{

cell.setCellValue("");

}

//人员类别

cell = row.createCell(2);

cell.setCellStyle(style0);

cell.setCellValue(familyUser.getFamily().getAllowanceAndPersonType().getName());

cell.setCellType(HSSFCell.CELL\_TYPE\_STRING);

//人口数

cell = row.createCell(3);

cell.setCellStyle(style0);

long userCount = super.getDefaultManager().getCount("from FamilyUser " +

" a where a.family=? " +

" and a.allowanceFamilyUser=1 " +

" and a.family.allowanceAndPersonType.parent.id=10141002 " +

" and a.status=1",familyUser.getFamily());

cell.setCellValue(userCount);

cell.setCellType(HSSFCell.CELL\_TYPE\_STRING);

//低保金

cell = row.createCell(4);

cell.setCellStyle(style0);

cell.setCellType(HSSFCell.CELL\_TYPE\_NUMERIC);

BigDecimal bd = null;

if(familyWelfareItem1 != null){

bd = new BigDecimal(""+familyWelfareItem1.getTotalAmount());

if(familyWelfareItem3 != null &&familyWelfareItem3.getTotalAmount() != null){

bd = bd.add(new BigDecimal(""+familyWelfareItem3.getTotalAmount()));

}

cell.setCellValue(bd.doubleValue());

}else{

cell.setCellValue(0.00);

}

//粮油补助

cell = row.createCell(5);

cell.setCellStyle(style0);

cell.setCellType(HSSFCell.CELL\_TYPE\_NUMERIC);

if(familyWelfareItem2 != null){

cell.setCellValue((double)familyWelfareItem2.getTotalAmount());

}else{

cell.setCellValue(0.00);

}

//合计

cell = row.createCell(6);

cell.setCellStyle(style0);

cell.setCellType(HSSFCell.CELL\_TYPE\_NUMERIC);

if(familyWelfareItem2 != null && familyWelfareItem2.getTotalAmount() != null){

BigDecimal one2 = new BigDecimal(""+familyWelfareItem2.getTotalAmount());

money = total.add(bd).add(one2).doubleValue();

}else{

money = bd.doubleValue();

}

cell.setCellValue(money);

//领款人签字

cell = row.createCell(7);

cell.setCellStyle(style0);

cell.setCellValue("");

cell.setCellType(HSSFCell.CELL\_TYPE\_STRING);

}

String strSql = " (a.allowance\_and\_person\_type\_id=10141012 or a.allowance\_and\_person\_type\_id=10141022 or a.allowance\_and\_person\_type\_id=10141032) ";

//查询条件

String sqlCondition = this.getConditions("10141002","2013-02");

//人员循环完毕后第一行

HSSFRow rowx1 = sheet.createRow(list.size()+3);

HSSFCell cellx1 = rowx1.createCell(0);

cellx1.setCellValue("合计");

cellx1.setCellStyle(style0);

cellx1 = rowx1.createCell(1);

cellx1.setCellValue("");

cellx1.setCellStyle(style0);

cellx1 = rowx1.createCell(2);

cellx1.setCellValue("");

cellx1.setCellStyle(style0);

//人员数合计

long userCount = super.getDefaultManager().getCount("from FamilyUser a where a.allowanceFamilyUser=1 and a.family.allowanceAndPersonType.parent.id=10141002 and a.status=1");

cellx1 = rowx1.createCell(3);

cellx1.setCellValue(userCount);

cellx1.setCellStyle(style0);

//低保金合计

Object allowanceMoney = super.getDefaultManager().getObjectSQL("select sum(t.total\_amount) " +

" from t\_mz\_family\_welfare\_item t " +

" where " +

" t.status=1 " +

" and (t.welfare\_item\_id=10270001 or t.welfare\_item\_id=10279999) " +

" and t.family\_id in " +

" (select a.family\_id " +

" from t\_mz\_family a " +

" where " + strSql +

" and a.status=1 " +

" and a.allowance\_family=1)");

// //低保金合计

// Object allowanceMoney = super.getDefaultManager().getObjectSQL("select " +

// " sum(t.grant\_amount) " +

// " from t\_mz\_family\_welfare\_item\_log t " +

// " where " +

// " to\_char(t.grant\_date,'yyyy-mm')=? " +

// " and t.item\_id in " +

// "(select " +

// " a.item\_id " +

// " from t\_mz\_family\_welfare\_item a " +

// " where (a.welfare\_item\_id=10270001 or a.welfare\_item\_id=10279999) " +

// " and a.family\_id in" +

// "(select " +

// " b.family\_id " +

// " from t\_mz\_family b " +

// " where "+strSql +

// " and b.status=1 " +

// " and b.allowance\_family=1))",grantDate);

cellx1 = rowx1.createCell(4);

if(allowanceMoney != null ){

cellx1.setCellValue(allowanceMoney.toString());

}else{

cellx1.setCellValue(0.00);

}

cellx1.setCellStyle(style0);

//粮油补贴合计

Object grainOilMoney = super.getDefaultManager().getObjectSQL("select sum(t.total\_amount) " +

" from t\_mz\_family\_welfare\_item t " +

" where t.status=1 " +

" and t.welfare\_item\_id=10270002 " +

" and t.family\_id in " +

" (select a.family\_id from t\_mz\_family a " +

" where " + strSql +

" and a.status=1 " +

" and a.allowance\_family=1)");

// //粮油补贴合计

// Object grainOilMoney = super.getDefaultManager().getObjectSQL("select " +

// " sum(t.grant\_amount) " +

// " from t\_mz\_family\_welfare\_item\_log t " +

// " where " +

// " to\_char(t.grant\_date,'yyyy-mm')=? " +

// " and t.item\_id in " +

// "(select " +

// " a.item\_id " +

// " from t\_mz\_family\_welfare\_item a " +

// " where a.welfare\_item\_id=10270002 " +

// " and a.family\_id in" +

// "(select " +

// " b.family\_id " +

// " from t\_mz\_family b " +

// " where "+strSql +

// " and b.status=1 " +

// " and b.allowance\_family=1))",grantDate);

cellx1 = rowx1.createCell(5);

if(grainOilMoney != null){

cellx1.setCellValue(grainOilMoney.toString());

}else{

cellx1.setCellValue(0.00);

}

cellx1.setCellStyle(style0);

//总合计

cellx1 = rowx1.createCell(6);

//总合计

BigDecimal allowanceM = new BigDecimal(""+0);

if(allowanceMoney != null){

allowanceM = new BigDecimal(""+allowanceMoney);

}

if(grainOilMoney != null){

BigDecimal grainOilM = new BigDecimal(""+grainOilMoney);

money = total.add(allowanceM).add(grainOilM).doubleValue();

}else{

money = new Double(allowanceM.toString());

}

cellx1.setCellValue(String.valueOf(money));

cellx1.setCellStyle(style0);

cellx1 = rowx1.createCell(7);

cellx1.setCellValue("");

cellx1.setCellStyle(style0);

//人员循环完毕后第二行

HSSFRow rowx2 = sheet.createRow(list.size()+4);

HSSFCell cellx2 = rowx2.createCell(0);

cellx2.setCellValue("注：此项支出根据人员类别记入“城市居民最低生活保障费”下的明细科目");

cellx2.setCellStyle(style2);

//人员循环完毕后第三行

HSSFRow rowx3 = sheet.createRow(list.size()+6);

HSSFCell cellx3 = rowx3.createCell(0);

cellx3.setCellValue("主管领导签字：");

cellx3.setCellStyle(style2);

cellx3 = rowx3.createCell(5);

cellx3.setCellValue("审核人签字：");

cellx3.setCellStyle(style2);

//人员循环完毕后第四行

HSSFRow rowx4 = sheet.createRow(list.size()+8);

HSSFCell cellx4 = rowx4.createCell(0);

cellx4.setCellValue("制表人签字：");

cellx4.setCellStyle(style2);

cellx4 = rowx4.createCell(5);

cellx4.setCellValue("制表时间：");

cellx4.setCellStyle(style2);

cellx4 = rowx4.createCell(6);

cellx4.setCellValue(getCDate());

cellx4.setCellStyle(style2);

ByteArrayOutputStream baos = new ByteArrayOutputStream();

try {

workbook.write(baos);

} catch (IOException e) {

e.printStackTrace();

}

byte[] ba = baos.toByteArray();

ByteArrayInputStream bais = new ByteArrayInputStream(ba);

return bais;

}

/\*\*

\* 获得查询条件

\*/

protected String getConditions(String allowanceTypeId,String date)

{

String conditionsStr = "";

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

conditionsStr = conditionsStr+ " and t.welfare\_item\_id = "+ allowanceTypeId +" ";

}

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

conditionsStr = conditionsStr+ " and to\_char(t.grant\_date,'yyyy-MM')='"+date+"'";;

}

return conditionsStr;

}

//根据指定格式获取当前日期

public static String getCurrentDate() {

return getCurrentTime("yyyy-MM");

}

public static String getCDate() {

return getCurrentTime("yyyy-MM-dd");

}

public static String getCurrentTime(String pattern) {

SimpleDateFormat sdf = new SimpleDateFormat(pattern);

return sdf.format(Calendar.getInstance().getTime());

}

public String getGrantDate() {

return grantDate;

}

public void setGrantDate(String grantDate) {

this.grantDate = grantDate;

}

}

package com.bizlogic.mz.action;

import java.io.File;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import com.bizlogic.base.AttachmentValue;

import com.bizlogic.base.User;

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

import com.bizlogic.base.helper.AttachmentHelper;

import com.bizlogic.common.Element;

import com.bizlogic.mz.bean.AllowanceChange;

import com.bizlogic.mz.bean.AllowanceRepeatChecked;

import com.bizlogic.mz.bean.AllowanceRepeatLog;

import com.bizlogic.mz.bean.AllowanceRepeatStatus;

import com.bizlogic.mz.bean.CacheFamily;

import com.bizlogic.mz.bean.CacheFamilyUser;

import com.bizlogic.mz.bean.Family;

import com.bizlogic.mz.bean.FamilyChange;

import com.bizlogic.mz.bean.FamilyChangeType;

import com.bizlogic.mz.bean.FamilyUser;

import com.bizlogic.mz.bean.FamilyUserWelfareItem;

import com.bizlogic.mz.bean.FamilyWelfareItem;

public class AllowanceRepeatCheckEditAction extends BaseUserEditAction{

//是否通过审核

private int isAllow;

private String familyId;

//附件上传----start

private AttachmentHelper attachmentHelper;

private File[] fileAttachment;

private String[] fileAttachmentContentType;

private String[] fileAttachmentFileName;

private long serial = 0;

//附件列表

private List<AttachmentValue> attachmentList;

private List<AttachmentValue> attachmentAllowanchengeList;

private Family family;

private CacheFamily cacheFamily;

//家庭成员列表

private List<FamilyUser> familyUserList;

//缓存家庭成员列表

private List cacheFamilyUserList;

//低保变更时选择的享受低保的家人

private String[] checkAllowanceUser;

//填写家庭福利列表

private List<FamilyUserWelfareItem> familyUserWelfareItemList;

//复审单中的家庭福利展示

private List<FamilyWelfareItem> familyWelfareItemList;

private AllowanceChange allowanceChange;

//流程状态

private String statusId;

//复审日期

private Date repeatCheckDate;

//复审办理时间

private Date repeatEditDate;

//社保所审核日期

private Date socialSecurityDate;

//街道民政科审核日期

private Date companyDate;

//人员福利展示

private String welfareStr;

//编辑状态：保存(save)、提交(submit)、修改(edit)

private String editFlag;

//填写变更单标记，当fillChangeForm值为y时，即为填写变更表

private String fillChangeForm;

//操作日志

private List<AllowanceRepeatLog> allowanceRepeatLogList;

//低保复审社区审核权限

private boolean allowanceRepeatCheck;

//低保复审社保所审核权限

private boolean allowanceRepeatSocialCheck;

//低保复审街道民政科审核权限

private boolean allowanceRepeatCompanyCheck;

protected Element createElement()

{

AllowanceRepeatChecked allowanceRepeatChecked = new AllowanceRepeatChecked();

//初始值

allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "1010"));

return allowanceRepeatChecked;

}

protected void prepareShow(){

AllowanceRepeatChecked allowanceRepeatChecked = (AllowanceRepeatChecked)element;

if(allowanceRepeatChecked.getId() != null){

allowanceRepeatLogList = super.getDefaultManager().findAll("from AllowanceRepeatLog a where a.allowanceRepeatChecked.id=? order by a.operateTime", allowanceRepeatChecked.getId());

}

if(allowanceRepeatChecked.getCreateTime() == null){

repeatEditDate = new Date();

}else{

repeatEditDate = allowanceRepeatChecked.getCreateTime();

}

if(allowanceRepeatChecked.getSocialSecurityDate() == null){

socialSecurityDate = new Date();

}else{

socialSecurityDate = allowanceRepeatChecked.getSocialSecurityDate();

}

if(allowanceRepeatChecked.getCompanyDate() == null){

companyDate = new Date();

}else{

companyDate = allowanceRepeatChecked.getCompanyDate();

}

if(allowanceRepeatChecked.getId() == null){

family = super.getDefaultManager().get(Family.class, familyId);

//从家庭人员列表

familyUserList = super.getDefaultManager().findAll("from FamilyUser a where a.family.id=? and a.status=1", familyId);

}else{

cacheFamily = (CacheFamily)super.getDefaultManager().getObject("from CacheFamily a where a.objectId=?", allowanceRepeatChecked.getId());

//缓存家庭成员列表

cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.id=? and a.status=1", cacheFamily.getId());

//成功变更后的家庭福利展示

familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem a where a.cacheFamily.id=? order by a.welfareItem.id", cacheFamily.getId());

//成功变更后的家庭人员福利展示

familyUserWelfareItemList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.cacheFamily.id=? order by a.familyUser.personRelationType.id,a.cacheFamilyUser.id,a.welfareItem.id", cacheFamily.getId());

//附件列表

attachmentList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime,a.id", allowanceRepeatChecked.getId());

//当复审单提交以后查看复审单

if(allowanceRepeatChecked.getStatus() == 1){

allowanceChange = allowanceRepeatChecked.getAllowanceChange();

//附件列表

attachmentAllowanchengeList = super.getDefaultManager().findAll("from AttachmentValue a where a.objectId=? order by a.createTime,a.id", allowanceRepeatChecked.getAllowanceChange().getId());

welfareStr = "";

//人员福利展示

List<FamilyWelfareItem> familyWelfareItemList = super.getDefaultManager().findAll("from FamilyWelfareItem f where f.cacheFamily.objectId=? order by f.welfareItem.id", allowanceRepeatChecked.getId());

StringBuffer str = new StringBuffer();

str.append("<table cellpadding='0' cellspacing='0' class='list\_tb'>");

str.append("<tr><th>姓名\\福利类别</th>");

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

str.append("<th>"+familyWelfareItemList.get(i).getWelfareItem().getName()+"</th>");

}

str.append("</tr>");

//该低保单对应的缓存家人列表

List<CacheFamilyUser> cacheFamilyUserList = super.getDefaultManager().findAll("from CacheFamilyUser a where a.cacheFamily.objectId=? and a.status=1 order by a.personRelationType.id", allowanceRepeatChecked.getId());

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

str.append("<tr>");

str.append("<td>"+cacheFamilyUserList.get(i).getName()+"</td>");

//家庭人员对应的福利列表

List<FamilyUserWelfareItem> familyUserWelfareList = super.getDefaultManager().findAll("from FamilyUserWelfareItem a where a.cacheFamilyUser.id=? order by a.welfareItem.id", cacheFamilyUserList.get(i).getId());

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

str.append("<td align='right' style='text-align:right'>"+familyUserWelfareList.get(j).getMonthAmount()+" 元/月</td>");

}

str.append("</tr>");

}

str.append("</table>");

welfareStr = str.toString();

}

}

String userId = getVisit().getId();

allowanceRepeatCheck = this.getUserRight(userId, "ALLOWANCE\_REPEAT\_CHECK");

allowanceRepeatSocialCheck = this.getUserRight(userId, "ALLOWANCE\_REPEAT\_SOCIAL\_CHECK");

allowanceRepeatCompanyCheck = this.getUserRight(userId, "ALLOWANCE\_REPEAT\_COMOMAPNY\_CHECK");

}

protected boolean prepareSave(Element element){

AllowanceRepeatChecked allowanceRepeatChecked = (AllowanceRepeatChecked)element;

Family family = new Family();

if(allowanceRepeatChecked.getId() == null){

family = super.getDefaultManager().get(Family.class, familyId);

}else{

family = allowanceRepeatChecked.getFamily();

}

//保存复审单

if("submit".equals(editFlag)){

//状态1010，表明社区审核提交

if("1010".equals(statusId)){

if(isAllow == 1){

//审核通过后，将状态改为社保所审核

allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "1020"));

}else if(isAllow == 2){

//0表示复审未通过

allowanceRepeatChecked.setResultStatus(0);

//记录审核不通过时的状态

allowanceRepeatChecked.setEndStatus(statusId);

allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "8888"));

}

allowanceRepeatChecked.setFamily(family);

// allowanceRepeatChecked.setCheckDate(getDate(repeatCheckDate));

allowanceRepeatChecked.setCreateUser((User)getVisit());

// allowanceRepeatChecked.setCreateTime(new Date());

allowanceRepeatChecked.setCreateTime(getDate(repeatEditDate));

//设置状态为0，表明复审单保存状态，未填写待遇审批表

allowanceRepeatChecked.setStatus(0);

//状态1011，表明退回复审单提交

}else if("1011".equals(statusId)){

if(isAllow == 1){

//将记录不通过的状态清空

allowanceRepeatChecked.setCreateUser((User)getVisit());

allowanceRepeatChecked.setEndStatus(null); allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "1020"));

}else if(isAllow == 2){ allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "8888"));

}

//状态1020，表明复审单社保所提交

}else if("1020".equals(statusId)){

if(isAllow == 1){

allowanceRepeatChecked.setSocialSecurityUser((User)getVisit()); allowanceRepeatChecked.setSocialSecurityDate(getDate(socialSecurityDate));

if("".equals(fillChangeForm) || fillChangeForm == null){ allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "1030"));

}else if("y".equals(fillChangeForm)){

}

}else if(isAllow == 2){ allowanceRepeatChecked.setSocialSecurityDate(getDate(socialSecurityDate));

allowanceRepeatChecked.setEndStatus(statusId); allowanceRepeatChecked.setAllowanceRepeatStatus(super.getDefaultManager().get(AllowanceRepeatStatus.class, "1011"));

}

}else if("1030".equals(statusId)){

if(isAllow == 1){ allowanceRepeatChecked.setCompanyUser((User)getVisit());

allowanceRepeatChecked.setResultStatus(1);

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