PetCircleController.Java

package com.zlz.animalfamily.controller;

import com.github.pagehelper.PageInfo;

import com.zlz.animalfamily.common.Status;

import com.zlz.animalfamily.common.SystemConstant;

import com.zlz.animalfamily.domain.\*;

import com.zlz.animalfamily.pojo.\*;

import com.zlz.animalfamily.service.PetCircleService;

import com.zlz.animalfamily.service.SuperService;

import com.zlz.animalfamily.util.FileServerClientUtil;

import com.zlz.animalfamily.util.FormatUtil;

import org.apache.commons.lang.StringUtils;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.\*;

import org.springframework.web.multipart.MultipartFile;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.Date;

import java.util.List;

/\*\*

\* 该类用于控制宠物圈

\* @author mmmgdzl

\* @since 2019/3/28

\*/

@Controller

public class PetCircleController {

@Autowired

private PetCircleService petCircleService;

@Autowired

private SuperService superService;

/\*\*

\* 宠物圈显示最热门条数

\*/

private final int PET\_CIRCLE\_HOT\_TOP\_NUM = 5;

/\*\*

\* 获取宠物圈信息列表

\*/

@GetMapping("/petCircle")

@ResponseBody

public Result getPetCircleList(PetCircle petCircle,

@RequestParam(defaultValue = "1") Integer currentPage,

@RequestParam(defaultValue = "9") Integer pageSize) {

petCircle.setPcState(Status.ACTIVE.getNum());

PetCircleExample petCircleExample = petCircleService.transformPetCircleToPetCircleExample(petCircle);

//时间逆序

petCircleExample.setOrderByClause("pc\_publish\_time DESC");

//执行查询

List<PetCircle> petCircles = petCircleService.selectPetCircleList(petCircleExample, currentPage, pageSize);

//获取统计数据

PageInfo<PetCircle> pageInfo = new PageInfo<>(petCircles);

//执行渲染

List<LayuiPetCircle> layuiPetCircleList = petCircleService.renderPetCircles(petCircles);

//执行封装

PageBean<LayuiPetCircle> pageBean = new PageBean<>();

pageBean.setData(layuiPetCircleList);

pageBean.setCurrentPage(currentPage);

pageBean.setPageSize(pageSize);

pageBean.setTotalNum((int)pageInfo.getTotal());

pageBean.setTotalPage(pageInfo.getPages());

return Result.OK(pageBean);

}

/\*\*

\* 获取宠物圈信息列表

\*/

@GetMapping("/layuiPetCircle")

@ResponseBody

public LayUIResult<LayuiPetCircle> getLayuiPetCircleList(PetCircle petCircle,

@RequestParam(defaultValue = "1") Integer page,

@RequestParam(defaultValue = "9") Integer limit) {

petCircle.setPcState(Status.ACTIVE.getNum());

PetCircleExample petCircleExample = petCircleService.transformPetCircleToPetCircleExample(petCircle);

//时间逆序

petCircleExample.setOrderByClause("pc\_publish\_time DESC");

//执行查询

List<PetCircle> petCircles = petCircleService.selectPetCircleList(petCircleExample, page, limit);

//获取统计数据

PageInfo<PetCircle> pageInfo = new PageInfo<>(petCircles);

//执行渲染

List<LayuiPetCircle> layuiPetCircleList = petCircleService.renderPetCircles(petCircles);

//执行封装

return new LayUIResult<>(0, (int)pageInfo.getTotal(), layuiPetCircleList);

}

/\*\*

\* 跳转到宠物圈详情页面

\*/

@GetMapping("/petCircle/{id}")

public String toPetCircleInfo(@PathVariable Integer id, Model data, HttpSession session) {

//根据id获取宠物圈信息

PetCircle petCircleByPcId = petCircleService.getPetCircleByPcId(id);

//渲染数据

LayuiPetCircle layuiPetCircle = petCircleService.renderPetCircle(petCircleByPcId);

//获取宠物圈发布者用户的个人简介

String introduce = superService.getUserByUId(petCircleByPcId.getPcUser()).getuIntroduce();

if(StringUtils.isBlank(introduce))

introduce = "该用户很懒,没有写个人简介~~";

//获取热度最高的5条宠物圈

List<PetCircle> hotPetCircle = petCircleService.getHotPetCircle(PET\_CIRCLE\_HOT\_TOP\_NUM);

//获取是否喜爱petCircleSearch

//获取登陆用户

Object attribute = session.getAttribute(SystemConstant.LOGIN\_USER);

int userId;

if(attribute == null) {

userId = (int)session.getCreationTime();

} else {

userId = ((AfUser)attribute).getuId();

}

boolean isLike = petCircleService.checkLikeStatus(userId, id);

//获取是否收藏

if(attribute != null) {

boolean isCollect = petCircleService.checkCollectStatus(((AfUser)attribute).getuId(), id);

data.addAttribute("isCollect", isCollect);

}

//将数据放入

data.addAttribute("petCircleInfo", layuiPetCircle);

data.addAttribute("hotPetCircle", hotPetCircle);

data.addAttribute("userIntroduce", introduce);

data.addAttribute("isLike", isLike);

return "pet\_circle\_info";

}

/\*\*

\* 添加一条宠物圈信息

\*/

@PostMapping("/protect/petCircle")

@ResponseBody

public Result addPetCircle(String title, MultipartFile titleImg, String content, HttpSession session) {

//将图片上传

FileData upload = FileServerClientUtil.upload(titleImg);

//创建资源对象

PetCircle petCircle = new PetCircle();

petCircle.setPcTitle(title);

petCircle.setPcTitleImg(upload.getSrc());

petCircle.setPcDescribe(content);

petCircle.setPcDescribeSmall(FormatUtil.getSmallDescribe(content));

//设置默认值

petCircle.setPcCollectNum(0);

petCircle.setPcLikeNum(0);

petCircle.setPcState(Status.ACTIVE.getNum());

petCircle.setPcPublishTime(new Date());

petCircle.setPcUser(((AfUser)session.getAttribute(SystemConstant.LOGIN\_USER)).getuId());

//执行插入

petCircleService.insertPetCircle(petCircle);

return Result.OK();

}

/\*\*

\* 获取宠物圈评论列表

\*/

@GetMapping("/petCircleComment")

@ResponseBody

public Result getPetCircleCommentList(Integer petCircleId,

@RequestParam(defaultValue = "1") Integer currentPage,

@RequestParam(defaultValue = "5") Integer pageSize) {

//获取宠物圈评论列表

PetCircleCommentExample petCircleCommentExample = new PetCircleCommentExample();

PetCircleCommentExample.Criteria criteria = petCircleCommentExample.createCriteria();

criteria.andPcomPetCircleEqualTo(petCircleId);

criteria.andPcomStateEqualTo(Status.ACTIVE.getNum());

//设置时间逆序

petCircleCommentExample.setOrderByClause("pcom\_time DESC");

//执行查询

List<PetCircleComment> petCircleComments = petCircleService.selectPetCircleCommentList(petCircleCommentExample, currentPage, pageSize);

//获取分析数据

PageInfo<PetCircleComment> pageInfo = new PageInfo<>(petCircleComments);

//执行渲染

List<LayuiPetCircleComment> layuiPetCircleComments = petCircleService.renderPetCircleComments(petCircleComments);

PageBean<LayuiPetCircleComment> pageBean = new PageBean<>();

pageBean.setData(layuiPetCircleComments);

pageBean.setCurrentPage(currentPage);

pageBean.setPageSize(pageSize);

pageBean.setTotalNum((int)pageInfo.getTotal());

pageBean.setTotalPage(pageInfo.getPages());

return Result.OK(pageBean);

}

/\*\*

\* 添加一条宠物圈评论

\*/

@PostMapping("/protect/petCircleComment")

@ResponseBody

public Result addPetCircleComment(PetCircleComment petCircleComment, HttpSession session) {

//设置属性

petCircleComment.setPcomState(Status.ACTIVE.getNum());

petCircleComment.setPcomBuilds(petCircleService.getMaxPcomBuilds(petCircleComment.getPcomPetCircle())+1);

petCircleComment.setPcomTime(new Date());

petCircleComment.setPcomUser(((AfUser)session.getAttribute(SystemConstant.LOGIN\_USER)).getuId());

//执行添加

petCircleService.insertPetCircleComment(petCircleComment);

return Result.OK();

}

/\*\*

\* 点赞控制

\*/

@RequestMapping("/controlPetCircleLike")

@ResponseBody

public Result controlLike(Integer petCircleId, HttpSession httpSession) {

//获取登陆用户

Object attribute = httpSession.getAttribute(SystemConstant.LOGIN\_USER);

int userId;

if(attribute == null) {

userId = (int)httpSession.getCreationTime();

} else {

userId = ((AfUser)attribute).getuId();

}

//接收点赞处理后的点赞数

int i = petCircleService.controlLike(userId, petCircleId);

//设置返回结果 msg: 点赞数 data:本人点赞状态

return Result.build(200, i+"", petCircleService.checkLikeStatus(userId, petCircleId));

}

/\*\*

\* 收藏控制

\*/

@RequestMapping("/protect/controlPetCircleCollect")

@ResponseBody

public Result controlCollect(Integer petCircleId, HttpSession session) {

AfUser afUser = (AfUser) session.getAttribute(SystemConstant.LOGIN\_USER);

//执行收藏控制

int num = petCircleService.controlCollect(afUser.getuId(), petCircleId);

//设置返回结果 msg: 收藏数 data:本人收藏状态

return Result.build(200, num+"", petCircleService.checkCollectStatus(afUser.getuId(), petCircleId));

}

/\*\*

\* 完全更新索引

\*/

@RequestMapping("/super/updatePetCircleSearch")

@ResponseBody

public Result doUpdateSearch() {

return petCircleService.updateAllPetCircleToSearch();

}

/\*\*

\* 执行索引

\*/

@RequestMapping("/petCircleSearch")

@ResponseBody

public Result doPetCircleSearch(String searchString) {

List<PetCircle> petCircles = petCircleService.doPetCircleSearch(searchString, 1, 9);

List<LayuiPetCircle> layuiPetCircleList = petCircleService.renderPetCircles(petCircles);

PageBean<LayuiPetCircle> pageBean = new PageBean<>();

pageBean.setData(layuiPetCircleList);

pageBean.setCurrentPage(1);

pageBean.setPageSize(9);

pageBean.setTotalNum(petCircles.size());

pageBean.setTotalPage(1);

return Result.OK(pageBean);

}

}

## PetCircleService.Java

package com.zlz.animalfamily.service;

import com.zlz.animalfamily.domain.LayimPetCircleCollect;

import com.zlz.animalfamily.domain.LayuiPetCircle;

import com.zlz.animalfamily.domain.LayuiPetCircleComment;

import com.zlz.animalfamily.domain.Result;

import com.zlz.animalfamily.pojo.\*;

import java.util.List;

public interface PetCircleService {

/\*\*

\* 添加一条宠物圈

\*/

void insertPetCircle(PetCircle petCircle);

/\*\*

\* 将PetCircle对象转换为查询对象

\*/

PetCircleExample transformPetCircleToPetCircleExample(PetCircle petCircle);

/\*\*

\* 根据查询对象查询(不包含大类型)

\*/

List<PetCircle> selectPetCircleList(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize);

/\*\*

\* 根据查询对象查询(包含大类型)

\*/

List<PetCircle> selectPetCircleListWithBLOBs(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize);

/\*\*

\* 根据查询对象统计

\*/

int countPetCircle(PetCircleExample petCircleExample);

/\*\*

\* 渲染一条PetCircle对象

\*/

LayuiPetCircle renderPetCircle(PetCircle petCircle);

/\*\*

\* 渲染一组PetCircle对象

\*/

List<LayuiPetCircle> renderPetCircles(List<PetCircle> petCircles);

/\*\*

\* 根据id获取PetCircle对象

\*/

PetCircle getPetCircleByPcId(int pcId);

/\*\*

\* 获取热度top宠物圈

\*/

List<PetCircle> getHotPetCircle(int topNum);

/\*\*

\* 将PetCircleComment对象转换为PetCircleComment查询对象

\*/

PetCircleCommentExample transformPetCircleCommentToPetCircleCommentExample(PetCircleComment petCircleComment);

/\*\*

\* 根据PetCircleComment查询对象查询

\*/

List<PetCircleComment> selectPetCircleCommentList(PetCircleCommentExample petCircleCommentExample, Integer currentPage, Integer pageSize);

/\*\*

\* 根据查询对象统计PetCircleComment

\*/

int countPetCircleComment(PetCircleCommentExample petCircleCommentExample);

/\*\*

\* 根据宠物圈id获取评论数

\*/

int countPetCircleCommentByPetCircleId(int petCircleId);

/\*\*

\* 渲染一条PetCircleComment对象

\*/

LayuiPetCircleComment renderPetCircleComment(PetCircleComment petCircleComment);

/\*\*

\* 渲染一组PetCircleComment对象

\*/

List<LayuiPetCircleComment> renderPetCircleComments(List<PetCircleComment> petCircleComments);

/\*\*

\* 根据宠物圈评论id获取宠物圈评论

\*/

PetCircleComment getPetCircleCommentByPcId(int pcId);

/\*\*

\* 获取某条宠物圈的最大评论楼数

\*/

int getMaxPcomBuilds(int petCircleId);

/\*\*

\* 添加一条宠物圈评论

\*/

void insertPetCircleComment(PetCircleComment petCircleComment);

/\*\*

\* 添加/删除点赞

\* @return 返回处理后的点赞数

\*/

int controlLike(int userId, int petCircleId);

/\*\*

\* 获取是否已点赞

\*/

boolean checkLikeStatus(int userId, int petCircleId);

/\*\*

\* 添加/删除收藏

\* @return 返回处理后的收藏数

\*/

int controlCollect(int userId, int petCircleId);

/\*\*

\* 获取是否已收藏

\*/

boolean checkCollectStatus(int userId, int petCircleId);

/\*\*

\* 获取用户收藏对象

\*/

PetCircleCollect getPetCircleCollect(int userId, int petCircleId);

/\*\*

\* 将所有宠物圈信息更新到索引列表

\*/

Result updateAllPetCircleToSearch();

/\*\*

\* 执行宠物圈全文检索

\*/

List<PetCircle> doPetCircleSearch(String searchString, Integer currentPage, Integer pageSize);

/\*\*

\* @author zkz

\* @since 2019/4/13

\* 获取用户收藏的宠物圈

\*/

List<PetCircleCollect> getPetCircleCollect(String userId,Integer currentPage, Integer pageSize);

/\*\*@author zkz

\* 将宠物圈收藏对象转化为查询对象

\*since 2019/4/13

\* \*/

List<LayimPetCircleCollect> renderPetCircleCollect(List<PetCircleCollect> petCircleCollects,List<String> titleLists,List<String> petCircleImgs);

}

## PetCircleServiceImpl.java

PetCircleServiceImpl

package com.zlz.animalfamily.service.impl;

import com.github.pagehelper.PageHelper;

import com.zlz.animalfamily.common.Status;

import com.zlz.animalfamily.domain.\*;

import com.zlz.animalfamily.mapper.\*;

import com.zlz.animalfamily.pojo.\*;

import com.zlz.animalfamily.repository.PetCircleRepository;

import com.zlz.animalfamily.service.PetCircleService;

import com.zlz.animalfamily.service.SuperService;

import com.zlz.animalfamily.util.FormatUtil;

import com.zlz.animalfamily.util.RenderUtil;

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

import org.springframework.data.redis.core.RedisTemplate;

import org.springframework.stereotype.Service;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

/\*\*

\* 该类提供宠物圈相关服务

\* @author mmmgdzl

\* @since 2019/3/28

\*/

@Service

public class PetCircleServiceImpl implements PetCircleService {

@Autowired

private RedisTemplate<Object, Flag> flagRedisTemplate;

@Autowired

private PetCircleRepository petCircleRepository;

@Autowired

private PetCircleMapper petCircleMapper;

@Autowired

private PetCircleCommentMapper petCircleCommentMapper;

@Autowired

private PetCircleCollectMapper petCircleCollectMapper;

@Autowired

private ExtraPetCircleCommentMapper extraPetCircleCommentMapper;

@Autowired

private SuperService superService;

@Autowired

private ExtraCollectMapper extraPetCircleCollectMapper;

@Override

public void insertPetCircle(PetCircle petCircle) {

petCircleMapper.insert(petCircle);

}

@Override

public PetCircleExample transformPetCircleToPetCircleExample(PetCircle petCircle) {

PetCircleExample petCircleExample = new PetCircleExample();

PetCircleExample.Criteria criteria = petCircleExample.createCriteria();

//设置查询条件

if(petCircle != null) {

if(petCircle.getPcTitle() != null)

criteria.andPcTitleLike(petCircle.getPcTitle());

if(petCircle.getPcState() != null)

criteria.andPcStateEqualTo(petCircle.getPcState());

if(petCircle.getPcUser() != null)

criteria.andPcUserEqualTo(petCircle.getPcUser());

}

return petCircleExample;

}

@Override

public List<PetCircle> selectPetCircleList(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize) {

//执行分页

if(currentPage != null && pageSize != null) {

PageHelper.startPage(currentPage, pageSize);

}

return petCircleMapper.selectByExample(petCircleExample);

}

@Override

public List<PetCircle> selectPetCircleListWithBLOBs(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize) {

//执行分页

if(currentPage != null && pageSize != null) {

PageHelper.startPage(currentPage, pageSize);

}

return petCircleMapper.selectByExampleWithBLOBs(petCircleExample);

}

@Override

public int countPetCircle(PetCircleExample petCircleExample) {

return petCircleMapper.countByExample(petCircleExample);

}

@Override

public LayuiPetCircle renderPetCircle(PetCircle petCircle) {

LayuiPetCircle layuiPetCircle = new LayuiPetCircle();

//执行渲染

//自动渲染基础属性

RenderUtil.renderNormal(petCircle, layuiPetCircle);

//渲染其他属性

layuiPetCircle.setPcPublishTime(FormatUtil.formatTime(petCircle.getPcPublishTime()));

layuiPetCircle.setPcUser(superService.getUserByUId(petCircle.getPcUser()).getuName());

Status num = Status.getByNum(petCircle.getPcState());

layuiPetCircle.setPcState(num.getStatusName());

return layuiPetCircle;

}

@Override

public List<LayuiPetCircle> renderPetCircles(List<PetCircle> petCircles) {

List<LayuiPetCircle> layuiPetCircleList = new ArrayList<>(petCircles.size());

for(PetCircle petCircle : petCircles) {

layuiPetCircleList.add(renderPetCircle(petCircle));

}

return layuiPetCircleList;

}

@Override

public PetCircle getPetCircleByPcId(int pcId) {

return petCircleMapper.selectByPrimaryKey(pcId);

}

@Override

public List<PetCircle> getHotPetCircle(int topNum) {

PetCircleExample petCircleExample = new PetCircleExample();

PetCircleExample.Criteria criteria = petCircleExample.createCriteria();

criteria.andPcStateEqualTo(Status.ACTIVE.getNum());

petCircleExample.setOrderByClause("pc\_like\_num+pc\_collect\_num\*2 DESC, pc\_publish\_time DESC");

return selectPetCircleList(petCircleExample, 1, topNum);

}

@Override

public PetCircleCommentExample transformPetCircleCommentToPetCircleCommentExample(PetCircleComment petCircleComment) {

PetCircleCommentExample petCircleCommentExample = new PetCircleCommentExample();

PetCircleCommentExample.Criteria criteria = petCircleCommentExample.createCriteria();

//设置查询条件

if(petCircleComment != null) {

if(petCircleComment.getPcomBuilds() != null)

criteria.andPcomBuildsEqualTo(petCircleComment.getPcomBuilds());

if(petCircleComment.getPcomMessage() != null)

criteria.andPcomMessageLike("%" + petCircleComment.getPcomMessage() + "%");

if(petCircleComment.getPcomPetCircle() != null)

criteria.andPcomPetCircleEqualTo(petCircleComment.getPcomPetCircle());

if(petCircleComment.getPcomReplyCommentId() != null)

criteria.andPcomReplyCommentIdEqualTo(petCircleComment.getPcomReplyCommentId());

if(petCircleComment.getPcomUser() != null)

criteria.andPcomUserEqualTo(petCircleComment.getPcomUser());

if(petCircleComment.getPcomState() != null)

criteria.andPcomStateEqualTo(petCircleComment.getPcomState());

else

criteria.andPcomStateNotEqualTo(Status.DELETE.getNum());

}

return petCircleCommentExample;

}

@Override

public List<PetCircleComment> selectPetCircleCommentList(PetCircleCommentExample petCircleCommentExample, Integer currentPage, Integer pageSize) {

//设置分页

if(currentPage != null && pageSize != null)

PageHelper.startPage(currentPage, pageSize);

return petCircleCommentMapper.selectByExample(petCircleCommentExample);

}

@Override

public int countPetCircleComment(PetCircleCommentExample petCircleCommentExample) {

return petCircleCommentMapper.countByExample(petCircleCommentExample);

}

@Override

public int countPetCircleCommentByPetCircleId(int petCircleId) {

PetCircleCommentExample petCircleCommentExample = new PetCircleCommentExample();

PetCircleCommentExample.Criteria criteria = petCircleCommentExample.createCriteria();

criteria.andPcomPetCircleEqualTo(petCircleId);

criteria.andPcomStateEqualTo(Status.ACTIVE.getNum());

return countPetCircleComment(petCircleCommentExample);

}

@Override

public LayuiPetCircleComment renderPetCircleComment(PetCircleComment petCircleComment) {

LayuiPetCircleComment layuiPetCircleComment = new LayuiPetCircleComment();

//自动渲染

RenderUtil.renderNormal(petCircleComment, layuiPetCircleComment);

//渲染其他属性

layuiPetCircleComment.setPcomTime(FormatUtil.formatTime(petCircleComment.getPcomTime()));

layuiPetCircleComment.setPcomUser(superService.getUserByUId(petCircleComment.getPcomUser()).getuName());

layuiPetCircleComment.setPcomState(Status.getByNum(petCircleComment.getPcomState()).getStatusName());

//如果为回复楼层

if(petCircleComment.getPcomReplyCommentId() != null)

layuiPetCircleComment.setPcomReplyCommentId(getPetCircleCommentByPcId(petCircleComment.getPcomReplyCommentId()).getPcomBuilds());

return layuiPetCircleComment;

}

@Override

public List<LayuiPetCircleComment> renderPetCircleComments(List<PetCircleComment> petCircleComments) {

List<LayuiPetCircleComment> layuiPetCircleCommentList = new ArrayList<>(petCircleComments.size());

for(PetCircleComment petCircleComment : petCircleComments) {

layuiPetCircleCommentList.add(renderPetCircleComment(petCircleComment));

}

return layuiPetCircleCommentList;

}

@Override

public PetCircleComment getPetCircleCommentByPcId(int pcId) {

return petCircleCommentMapper.selectByPrimaryKey(pcId);

}

@Override

public int getMaxPcomBuilds(int petCircleId) {

Integer maxPcomBuilds = extraPetCircleCommentMapper.getMaxPcomBuilds(petCircleId);

if(maxPcomBuilds == null)

return 0;

else

return maxPcomBuilds;

}

@Override

public void insertPetCircleComment(PetCircleComment petCircleComment) {

petCircleCommentMapper.insert(petCircleComment);

}

@Override

public int controlLike(int userId, int petCircleId) {

String name = "pet-circle-like:" + userId + "-" + petCircleId;

Boolean aBoolean = flagRedisTemplate.hasKey(name);

PetCircle petCircle = new PetCircle();

petCircle.setPcId(petCircleId);

if(aBoolean) {

flagRedisTemplate.delete(name);

//点赞数-1

petCircle.setPcLikeNum(getPetCircleByPcId(petCircleId).getPcLikeNum() - 1);

} else {

flagRedisTemplate.opsForValue().set(name, new Flag());

//点赞数+1

petCircle.setPcLikeNum(getPetCircleByPcId(petCircleId).getPcLikeNum() + 1);

}

petCircleMapper.updateByPrimaryKeySelective(petCircle);

return petCircle.getPcLikeNum();

}

@Override

public boolean checkLikeStatus(int userId, int petCircleId) {

String name = "pet-circle-like:" + userId + "-" + petCircleId;

//从缓存中读取

return flagRedisTemplate.hasKey(name);

}

@Override

public int controlCollect(int userId, int petCircleId) {

//获取当前状态

boolean currentStatus = checkCollectStatus(userId, petCircleId);

//设置更新对象

PetCircle petCircle = new PetCircle();

petCircle.setPcId(petCircleId);

if(currentStatus) {

//已收藏则删除收藏记录

petCircleCollectMapper.deleteByPrimaryKey(getPetCircleCollect(userId, petCircleId).getPccId());

} else {

//未收藏则添加一条宠物圈收藏记录

PetCircleCollect petCircleCollect = new PetCircleCollect();

petCircleCollect.setPccPcId(petCircleId);

petCircleCollect.setPccTime(new Date());

petCircleCollect.setPccUserId(userId);

//执行插入

petCircleCollectMapper.insert(petCircleCollect);

}

//获取对应宠物圈

PetCircle petCircleByPcId = getPetCircleByPcId(petCircleId);

//已收藏则-1 未收藏则+1

int num = petCircleByPcId.getPcCollectNum() + (currentStatus ? -1 : 1);

petCircle.setPcCollectNum(num);

//更新宠物圈收藏个数

petCircleMapper.updateByPrimaryKeySelective(petCircle);

return num;

}

@Override

public boolean checkCollectStatus(int userId, int petCircleId) {

PetCircleCollect petCircleCollect = getPetCircleCollect(userId, petCircleId);

return petCircleCollect != null;

}

@Override

public PetCircleCollect getPetCircleCollect(int userId, int petCircleId) {

PetCircleCollectExample petCircleCollectExample = new PetCircleCollectExample();

petCircleCollectExample.createCriteria().andPccUserIdEqualTo(userId).andPccPcIdEqualTo(petCircleId);

//执行分页 只查一条

PageHelper.startPage(1, 1);

//执行查询

List<PetCircleCollect> petCircleCollects = petCircleCollectMapper.selectByExample(petCircleCollectExample);

if(petCircleCollects.size() == 0)

return null;

else

return petCircleCollects.get(0);

}

@Override

public Result updateAllPetCircleToSearch() {

//获取所有宠物圈信息

List<PetCircle> petCircles = petCircleMapper.selectByExampleWithBLOBs(new PetCircleExample());

//清空此type下的文档

petCircleRepository.deleteAll();

//批量提交

petCircleRepository.save(petCircles);

return Result.OK();

}

@Override

public List<PetCircle> doPetCircleSearch(String searchString, Integer currentPage, Integer pageSize) {

List<PetCircle> petCircles = petCircleRepository.findByPcDescribeLikeOrderByPcPublishTime(searchString);

return petCircles;

}

@Override

public List<PetCircleCollect> getPetCircleCollect(String userId,Integer currentPage, Integer pageSize) {

//设置分页

if(currentPage != null && pageSize != null)

PageHelper.startPage(currentPage, pageSize);

return extraPetCircleCollectMapper.selectPetCicleCollectByUserId(userId);

}

@Override

public List<LayimPetCircleCollect> renderPetCircleCollect(List<PetCircleCollect> petCircleCollects,List<String> titleLists,List<String> petCircleImgs) {

List<LayimPetCircleCollect> layimPetCircleCollects=new ArrayList<>(petCircleCollects.size());

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

if(petCircleCollects.get(i)!=null&&titleLists.get(i)!=null&&petCircleImgs.get(i)!=null) {

layimPetCircleCollects.add(renderpetCircleCollect(petCircleCollects.get(i), titleLists.get(i),petCircleImgs.get(i)));

}

}

return layimPetCircleCollects;

}

private LayimPetCircleCollect renderpetCircleCollect(PetCircleCollect petCircleCollect,String title,String petImg) {

LayimPetCircleCollect layimPetCircleCollect=new LayimPetCircleCollect();

//自动渲染

RenderUtil.renderNormal(petCircleCollect,layimPetCircleCollect);

//渲染其他属性

layimPetCircleCollect.setPcTitle(title);

layimPetCircleCollect.setPetCircleImg(petImg);

return layimPetCircleCollect;

}

}

package com.zlz.animalfamily.service.impl;

import com.github.pagehelper.PageHelper;

import com.zlz.animalfamily.common.Status;

import com.zlz.animalfamily.domain.\*;

import com.zlz.animalfamily.mapper.\*;

import com.zlz.animalfamily.pojo.\*;

import com.zlz.animalfamily.repository.PetCircleRepository;

import com.zlz.animalfamily.service.PetCircleService;

import com.zlz.animalfamily.service.SuperService;

import com.zlz.animalfamily.util.FormatUtil;

import com.zlz.animalfamily.util.RenderUtil;

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

import org.springframework.data.redis.core.RedisTemplate;

import org.springframework.stereotype.Service;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

/\*\*

\* 该类提供宠物圈相关服务

\* @author mmmgdzl

\* @since 2019/3/28

\*/

@Service

public class PetCircleServiceImpl implements PetCircleService {

@Autowired

private RedisTemplate<Object, Flag> flagRedisTemplate;

@Autowired

private PetCircleRepository petCircleRepository;

@Autowired

private PetCircleMapper petCircleMapper;

@Autowired

private PetCircleCommentMapper petCircleCommentMapper;

@Autowired

private PetCircleCollectMapper petCircleCollectMapper;

@Autowired

private ExtraPetCircleCommentMapper extraPetCircleCommentMapper;

@Autowired

private SuperService superService;

@Autowired

private ExtraCollectMapper extraPetCircleCollectMapper;

@Override

public void insertPetCircle(PetCircle petCircle) {

petCircleMapper.insert(petCircle);

}

@Override

public PetCircleExample transformPetCircleToPetCircleExample(PetCircle petCircle) {

PetCircleExample petCircleExample = new PetCircleExample();

PetCircleExample.Criteria criteria = petCircleExample.createCriteria();

//设置查询条件

if(petCircle != null) {

if(petCircle.getPcTitle() != null)

criteria.andPcTitleLike(petCircle.getPcTitle());

if(petCircle.getPcState() != null)

criteria.andPcStateEqualTo(petCircle.getPcState());

if(petCircle.getPcUser() != null)

criteria.andPcUserEqualTo(petCircle.getPcUser());

}

return petCircleExample;

}

@Override

public List<PetCircle> selectPetCircleList(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize) {

//执行分页

if(currentPage != null && pageSize != null) {

PageHelper.startPage(currentPage, pageSize);

}

return petCircleMapper.selectByExample(petCircleExample);

}

@Override

public List<PetCircle> selectPetCircleListWithBLOBs(PetCircleExample petCircleExample, Integer currentPage, Integer pageSize) {

//执行分页

if(currentPage != null && pageSize != null) {

PageHelper.startPage(currentPage, pageSize);

}

return petCircleMapper.selectByExampleWithBLOBs(petCircleExample);

}

@Override

public int countPetCircle(PetCircleExample petCircleExample) {

return petCircleMapper.countByExample(petCircleExample);

}

@Override

public LayuiPetCircle renderPetCircle(PetCircle petCircle) {

LayuiPetCircle layuiPetCircle = new LayuiPetCircle();

//执行渲染

//自动渲染基础属性

RenderUtil.renderNormal(petCircle, layuiPetCircle);

//渲染其他属性

layuiPetCircle.setPcPublishTime(FormatUtil.formatTime(petCircle.getPcPublishTime()));

layuiPetCircle.setPcUser(superService.getUserByUId(petCircle.getPcUser()).getuName());

Status num = Status.getByNum(petCircle.getPcState());

layuiPetCircle.setPcState(num.getStatusName());

return layuiPetCircle;

}

@Override

public List<LayuiPetCircle> renderPetCircles(List<PetCircle> petCircles) {

List<LayuiPetCircle> layuiPetCircleList = new ArrayList<>(petCircles.size());

for(PetCircle petCircle : petCircles) {

layuiPetCircleList.add(renderPetCircle(petCircle));

}

return layuiPetCircleList;

}

@Override

public PetCircle getPetCircleByPcId(int pcId) {

return petCircleMapper.selectByPrimaryKey(pcId);

}

@Override

public List<PetCircle> getHotPetCircle(int topNum) {

PetCircleExample petCircleExample = new PetCircleExample();

PetCircleExample.Criteria criteria = petCircleExample.createCriteria();

criteria.andPcStateEqualTo(Status.ACTIVE.getNum());

petCircleExample.setOrderByClause("pc\_like\_num+pc\_collect\_num\*2 DESC, pc\_publish\_time DESC");

return selectPetCircleList(petCircleExample, 1, topNum);

}

@Override

public PetCircleCommentExample transformPetCircleCommentToPetCircleCommentExample(PetCircleComment petCircleComment) {

PetCircleCommentExample petCircleCommentExample = new PetCircleCommentExample();

PetCircleCommentExample.Criteria criteria = petCircleCommentExample.createCriteria();

//设置查询条件

if(petCircleComment != null) {

if(petCircleComment.getPcomBuilds() != null)

criteria.andPcomBuildsEqualTo(petCircleComment.getPcomBuilds());

if(petCircleComment.getPcomMessage() != null)

criteria.andPcomMessageLike("%" + petCircleComment.getPcomMessage() + "%");

if(petCircleComment.getPcomPetCircle() != null)

criteria.andPcomPetCircleEqualTo(petCircleComment.getPcomPetCircle());

if(petCircleComment.getPcomReplyCommentId() != null)

criteria.andPcomReplyCommentIdEqualTo(petCircleComment.getPcomReplyCommentId());

if(petCircleComment.getPcomUser() != null)

criteria.andPcomUserEqualTo(petCircleComment.getPcomUser());

if(petCircleComment.getPcomState() != null)

criteria.andPcomStateEqualTo(petCircleComment.getPcomState());

else

criteria.andPcomStateNotEqualTo(Status.DELETE.getNum());

}

return petCircleCommentExample;

}

@Override

public List<PetCircleComment> selectPetCircleCommentList(PetCircleCommentExample petCircleCommentExample, Integer currentPage, Integer pageSize) {

//设置分页

if(currentPage != null && pageSize != null)

PageHelper.startPage(currentPage, pageSize);

return petCircleCommentMapper.selectByExample(petCircleCommentExample);

}

@Override

public int countPetCircleComment(PetCircleCommentExample petCircleCommentExample) {

return petCircleCommentMapper.countByExample(petCircleCommentExample);

}

@Override

public int countPetCircleCommentByPetCircleId(int petCircleId) {

PetCircleCommentExample petCircleCommentExample = new PetCircleCommentExample();

PetCircleCommentExample.Criteria criteria = petCircleCommentExample.createCriteria();

criteria.andPcomPetCircleEqualTo(petCircleId);

criteria.andPcomStateEqualTo(Status.ACTIVE.getNum());

return countPetCircleComment(petCircleCommentExample);

}

@Override

public LayuiPetCircleComment renderPetCircleComment(PetCircleComment petCircleComment) {

LayuiPetCircleComment layuiPetCircleComment = new LayuiPetCircleComment();

//自动渲染

RenderUtil.renderNormal(petCircleComment, layuiPetCircleComment);

//渲染其他属性

layuiPetCircleComment.setPcomTime(FormatUtil.formatTime(petCircleComment.getPcomTime()));

layuiPetCircleComment.setPcomUser(superService.getUserByUId(petCircleComment.getPcomUser()).getuName());

layuiPetCircleComment.setPcomState(Status.getByNum(petCircleComment.getPcomState()).getStatusName());

//如果为回复楼层

if(petCircleComment.getPcomReplyCommentId() != null)

layuiPetCircleComment.setPcomReplyCommentId(getPetCircleCommentByPcId(petCircleComment.getPcomReplyCommentId()).getPcomBuilds());

return layuiPetCircleComment;

}

@Override

public List<LayuiPetCircleComment> renderPetCircleComments(List<PetCircleComment> petCircleComments) {

List<LayuiPetCircleComment> layuiPetCircleCommentList = new ArrayList<>(petCircleComments.size());

for(PetCircleComment petCircleComment : petCircleComments) {

layuiPetCircleCommentList.add(renderPetCircleComment(petCircleComment));

}

return layuiPetCircleCommentList;

}

@Override

public PetCircleComment getPetCircleCommentByPcId(int pcId) {

return petCircleCommentMapper.selectByPrimaryKey(pcId);

}

@Override

public int getMaxPcomBuilds(int petCircleId) {

Integer maxPcomBuilds = extraPetCircleCommentMapper.getMaxPcomBuilds(petCircleId);

if(maxPcomBuilds == null)

return 0;

else

return maxPcomBuilds;

}

@Override

public void insertPetCircleComment(PetCircleComment petCircleComment) {

petCircleCommentMapper.insert(petCircleComment);

}

@Override

public int controlLike(int userId, int petCircleId) {

String name = "pet-circle-like:" + userId + "-" + petCircleId;

Boolean aBoolean = flagRedisTemplate.hasKey(name);

PetCircle petCircle = new PetCircle();

petCircle.setPcId(petCircleId);

if(aBoolean) {

flagRedisTemplate.delete(name);

//点赞数-1

petCircle.setPcLikeNum(getPetCircleByPcId(petCircleId).getPcLikeNum() - 1);

} else {

flagRedisTemplate.opsForValue().set(name, new Flag());

//点赞数+1

petCircle.setPcLikeNum(getPetCircleByPcId(petCircleId).getPcLikeNum() + 1);

}

petCircleMapper.updateByPrimaryKeySelective(petCircle);

return petCircle.getPcLikeNum();

}

@Override

public boolean checkLikeStatus(int userId, int petCircleId) {

String name = "pet-circle-like:" + userId + "-" + petCircleId;

//从缓存中读取

return flagRedisTemplate.hasKey(name);

}

@Override

public int controlCollect(int userId, int petCircleId) {

//获取当前状态

boolean currentStatus = checkCollectStatus(userId, petCircleId);

//设置更新对象

PetCircle petCircle = new PetCircle();

petCircle.setPcId(petCircleId);

if(currentStatus) {

//已收藏则删除收藏记录

petCircleCollectMapper.deleteByPrimaryKey(getPetCircleCollect(userId, petCircleId).getPccId());

} else {

//未收藏则添加一条宠物圈收藏记录

PetCircleCollect petCircleCollect = new PetCircleCollect();

petCircleCollect.setPccPcId(petCircleId);

petCircleCollect.setPccTime(new Date());

petCircleCollect.setPccUserId(userId);

//执行插入

petCircleCollectMapper.insert(petCircleCollect);

}

//获取对应宠物圈

PetCircle petCircleByPcId = getPetCircleByPcId(petCircleId);

//已收藏则-1 未收藏则+1

int num = petCircleByPcId.getPcCollectNum() + (currentStatus ? -1 : 1);

petCircle.setPcCollectNum(num);

//更新宠物圈收藏个数

petCircleMapper.updateByPrimaryKeySelective(petCircle);

return num;

}

@Override

public boolean checkCollectStatus(int userId, int petCircleId) {

PetCircleCollect petCircleCollect = getPetCircleCollect(userId, petCircleId);

return petCircleCollect != null;

}

@Override

public PetCircleCollect getPetCircleCollect(int userId, int petCircleId) {

PetCircleCollectExample petCircleCollectExample = new PetCircleCollectExample();

petCircleCollectExample.createCriteria().andPccUserIdEqualTo(userId).andPccPcIdEqualTo(petCircleId);

//执行分页 只查一条

PageHelper.startPage(1, 1);

//执行查询

List<PetCircleCollect> petCircleCollects = petCircleCollectMapper.selectByExample(petCircleCollectExample);

if(petCircleCollects.size() == 0)

return null;

else

return petCircleCollects.get(0);

}

@Override

public Result updateAllPetCircleToSearch() {

//获取所有宠物圈信息

List<PetCircle> petCircles = petCircleMapper.selectByExampleWithBLOBs(new PetCircleExample());

//清空此type下的文档

petCircleRepository.deleteAll();

//批量提交

petCircleRepository.save(petCircles);

return Result.OK();

}

@Override

public List<PetCircle> doPetCircleSearch(String searchString, Integer currentPage, Integer pageSize) {

List<PetCircle> petCircles = petCircleRepository.findByPcDescribeLikeOrderByPcPublishTime(searchString);

return petCircles;

}

@Override

public List<PetCircleCollect> getPetCircleCollect(String userId,Integer currentPage, Integer pageSize) {

//设置分页

if(currentPage != null && pageSize != null)

PageHelper.startPage(currentPage, pageSize);

return extraPetCircleCollectMapper.selectPetCicleCollectByUserId(userId);

}

@Override

public List<LayimPetCircleCollect> renderPetCircleCollect(List<PetCircleCollect> petCircleCollects,List<String> titleLists,List<String> petCircleImgs) {

List<LayimPetCircleCollect> layimPetCircleCollects=new ArrayList<>(petCircleCollects.size());

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

if(petCircleCollects.get(i)!=null&&titleLists.get(i)!=null&&petCircleImgs.get(i)!=null) {

layimPetCircleCollects.add(renderpetCircleCollect(petCircleCollects.get(i), titleLists.get(i),petCircleImgs.get(i)));

}

}

return layimPetCircleCollects;

}

private LayimPetCircleCollect renderpetCircleCollect(PetCircleCollect petCircleCollect,String title,String petImg) {

LayimPetCircleCollect layimPetCircleCollect=new LayimPetCircleCollect();

//自动渲染

RenderUtil.renderNormal(petCircleCollect,layimPetCircleCollect);

//渲染其他属性

layimPetCircleCollect.setPcTitle(title);

layimPetCircleCollect.setPetCircleImg(petImg);

return layimPetCircleCollect;

}

}

## Common.java

package com.zlz.animalfamily.common;

/\*\*

\* 状态枚举类

\* @author mmmgdzl

\* @since 2019/3/28

\*/

public enum Status {

DISABLE(0, "不可用"), ACTIVE(1, "可用"), DELETE(2, "删除");

private int num;

private String statusName;

Status(int num, String statusName) {

this.num = num;

this.statusName = statusName;

}

public static Status getByNum(int num) {

for(Status \_enum : Status.class.getEnumConstants())

if(\_enum.getNum() == num)

return \_enum;

return null;

}

public static Status getByStatusName(String statusName) {

for(Status \_enum : Status.class.getEnumConstants())

if(\_enum.getStatusName().equals(statusName))

return \_enum;

return null;

}

public int getNum() {

return num;

}

public String getStatusName() {

return statusName;

}

}

Result.java

package com.zlz.animalfamily.domain;

public class Result {

private Integer code;

private String msg;

private Object data;

/\*\*

\* 返回码为200的成功结果

\*/

public static Result OK() {

Result result = new Result();

result.code = 200;

return result;

}

/\*\*

\* 返回码为200的带数据的成功结果

\*/

public static Result OK(Object data) {

Result result = new Result();

result.code = 200;

result.data = data;

return result;

}

/\*\*

\* 创建自定义结果

\*/

public static Result build(Integer code, String msg) {

Result result = new Result();

result.code = code;

result.msg = msg;

return result;

}

/\*\*

\* 创建自定义结果

\*/

public static Result build(Integer code, Object data) {

Result result = new Result();

result.code = code;

result.data = data;

return result;

}

/\*\*

\* 创建自定义结果

\*/

public static Result build(Integer code, String msg, Object data) {

Result result = new Result();

result.code = code;

result.msg = msg;

result.data = data;

return result;

}

public Integer getCode() {

return code;

}

public void setCode(Integer code) {

this.code = code;

}

public String getMsg() {

return msg;

}

public void setMsg(String msg) {

this.msg = msg;

}

public Object getData() {

return data;

}

public void setData(Object data) {

this.data = data;

}

}

­­PageBean.Java

package com.zlz.animalfamily.domain;

import java.util.List;

public class PageBean<T> {

//当前页码

private Integer currentPage;

//总页数

private Integer totalPage;

//总条数

private Integer totalNum;

//每页显示条数

private Integer pageSize;

//当前页面主题

private String type;

//用于存放查询条件的对象,也可作他用

private T obj;

//用于存放列表数据

private List<T> data;

public Integer getCurrentPage() {

return currentPage;

}

public void setCurrentPage(Integer currentPage) {

this.currentPage = currentPage;

}

public Integer getTotalPage() {

return totalPage;

}

public void setTotalPage(Integer totalPage) {

this.totalPage = totalPage;

}

public Integer getTotalNum() {

return totalNum;

}

public void setTotalNum(Integer totalNum) {

this.totalNum = totalNum;

}

public Integer getPageSize() {

return pageSize;

}

public void setPageSize(Integer pageSize) {

this.pageSize = pageSize;

}

public T getObj() {

return obj;

}

public void setObj(T obj) {

this.obj = obj;

}

public List<T> getData() {

return data;

}

public void setData(List<T> data) {

this.data = data;

}

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

}

## PetCircleMapper.Java

package com.zlz.animalfamily.mapper;

import com.zlz.animalfamily.pojo.PetCircle;

import com.zlz.animalfamily.pojo.PetCircleExample;

import java.util.List;

import org.apache.ibatis.annotations.Param;

public interface PetCircleMapper {

int countByExample(PetCircleExample example);

int deleteByExample(PetCircleExample example);

int deleteByPrimaryKey(Integer pcId);

int insert(PetCircle record);

int insertSelective(PetCircle record);

List<PetCircle> selectByExampleWithBLOBs(PetCircleExample example);

List<PetCircle> selectByExample(PetCircleExample example);

PetCircle selectByPrimaryKey(Integer pcId);

int updateByExampleSelective(@Param("record") PetCircle record, @Param("example") PetCircleExample example);

int updateByExampleWithBLOBs(@Param("record") PetCircle record, @Param("example") PetCircleExample example);

int updateByExample(@Param("record") PetCircle record, @Param("example") PetCircleExample example);

int updateByPrimaryKeySelective(PetCircle record);

int updateByPrimaryKeyWithBLOBs(PetCircle record);

int updateByPrimaryKey(PetCircle record);

}

## PetCircle.java

package com.zlz.animalfamily.pojo;

import com.fasterxml.jackson.annotation.JsonProperty;

import org.springframework.data.annotation.Id;

import org.springframework.data.elasticsearch.annotations.Document;

import java.util.Date;

@Document(indexName = "animal-family", type = "pet-circle")

public class PetCircle {

@Id

private Integer pcId;

private String pcTitle;

private String pcTitleImg;

private Date pcPublishTime;

private Integer pcUser;

private Integer pcLikeNum;

private Integer pcCollectNum;

private Integer pcState;

private String pcDescribeSmall;

private String pcDescribe;

public Integer getPcId() {

return pcId;

}

public void setPcId(Integer pcId) {

this.pcId = pcId;

}

public String getPcTitle() {

return pcTitle;

}

public void setPcTitle(String pcTitle) {

this.pcTitle = pcTitle == null ? null : pcTitle.trim();

}

public String getPcTitleImg() {

return pcTitleImg;

}

public void setPcTitleImg(String pcTitleImg) {

this.pcTitleImg = pcTitleImg == null ? null : pcTitleImg.trim();

}

public Date getPcPublishTime() {

return pcPublishTime;

}

public void setPcPublishTime(Date pcPublishTime) {

this.pcPublishTime = pcPublishTime;

}

public Integer getPcUser() {

return pcUser;

}

public void setPcUser(Integer pcUser) {

this.pcUser = pcUser;

}

public Integer getPcLikeNum() {

return pcLikeNum;

}

public void setPcLikeNum(Integer pcLikeNum) {

this.pcLikeNum = pcLikeNum;

}

public Integer getPcCollectNum() {

return pcCollectNum;

}

public void setPcCollectNum(Integer pcCollectNum) {

this.pcCollectNum = pcCollectNum;

}

public Integer getPcState() {

return pcState;

}

public void setPcState(Integer pcState) {

this.pcState = pcState;

}

public String getPcDescribeSmall() {

return pcDescribeSmall;

}

public void setPcDescribeSmall(String pcDescribeSmall) {

this.pcDescribeSmall = pcDescribeSmall == null ? null : pcDescribeSmall.trim();

}

public String getPcDescribe() {

return pcDescribe;

}

public void setPcDescribe(String pcDescribe) {

this.pcDescribe = pcDescribe == null ? null : pcDescribe.trim();

}

}