package edu.nuc.bean.help;

public class EntityForHQL {

private String hql;

private Object[] args;

public String getHql() {

return hql;

}

public void setHql(String hql) {

this.hql = hql;

}

public Object[] getArgs() {

return args;

}

public void setArgs(Object[] args) {

this.args = args;

}

public EntityForHQL() {

// TODO Auto-generated constructor stub

}

public EntityForHQL(String hql, Object[] args) {

this.hql = hql;

this.args = args;

}

}

package edu.nuc.bean.help;

public class EntityForHQL {

private String hql;

private Object[] args;

public String getHql() {

return hql;

}

public void setHql(String hql) {

this.hql = hql;

}

public Object[] getArgs() {

return args;

}

public void setArgs(Object[] args) {

this.args = args;

}

public EntityForHQL() {

// TODO Auto-generated constructor stub

}

public EntityForHQL(String hql, Object[] args) {

this.hql = hql;

this.args = args;

}

}

package edu.nuc.bean.help;

public class Filter {

private String columnName;

private String key;

public String getColumnName() {

return columnName;

}

public void setColumnName(String columnName) {

this.columnName = columnName;

}

public String getKey() {

return key;

}

public void setKey(String key) {

this.key = key;

}

}

package edu.nuc.bean.help;

public class Pager {

private int pageNum;

private int pageCount;

public int getPageNum() {

return pageNum;

}

public void setPageNum(int pageNum) {

this.pageNum = pageNum;

}

public int getPageCount() {

return pageCount;

}

public void setPageCount(int pageCount) {

this.pageCount = pageCount;

}

public Pager(int pageNum, int pageCount) {

super();

this.pageNum = pageNum;

this.pageCount = pageCount;

}

}

package edu.nuc.bean.help;

public class SQLProperties {

private String columnName;

private String eq;

private Object obj;

public SQLProperties() {

// TODO Auto-generated constructor stub

}

public String getColumnName() {

return columnName;

}

public void setColumnName(String columnName) {

this.columnName = columnName;

}

public String getEq() {

return eq;

}

public void setEq(String eq) {

this.eq = eq;

}

public Object getObj() {

return obj;

}

public void setObj(Object obj) {

this.obj = obj;

}

public SQLProperties(String columnName, String eq, Object obj) {

this.columnName = columnName;

this.eq = eq;

this.obj = obj;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="Comment" table="onlineshare.comment">

<id name="id" column="id" type="integer">

<generator class="increment"></generator>

</id>

<many-to-one name="user" class="UserInfo" column="userId" lazy="false"></many-to-one>

<many-to-one name="message" class="Message" column="messageId" lazy="false"></many-to-one>

<property name="content" column="content" type="string"></property>

<property name="del" column="del" type="boolean"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

import java.io.Serializable;

public class Comment implements Serializable{

/\*\*

\*

\*/

private int id;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

private static final long serialVersionUID = 1L;

private UserInfo user;

private Message message;

private String content = "";

private boolean del;

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public Message getMessage() {

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public String getContent() {

return content;

}

public void setContent(String content) {

this.content = content;

}

public boolean isDel() {

return del;

}

public void setDel(boolean del) {

this.del = del;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="Follow" table="onlineshare.follow">

<composite-id>

<key-many-to-one name="user" column="userId" class="UserInfo" foreign-key="id"></key-many-to-one>

<key-many-to-one name="follower" column="followerId" class="UserInfo" foreign-key="id"></key-many-to-one>

</composite-id>

<property name="del" column="del" type="boolean"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

import java.io.Serializable;

public class Follow implements Serializable{

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

private UserInfo user;

private UserInfo follower;

private boolean del;

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public UserInfo getFollower() {

return follower;

}

public void setFollower(UserInfo follower) {

this.follower = follower;

}

public boolean isDel() {

return del;

}

public void setDel(boolean del) {

this.del = del;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="Like" table="onlineshare.like">

<composite-id>

<key-many-to-one name="user" column="userId" class="UserInfo" foreign-key="id"></key-many-to-one>

<key-many-to-one name="message" column="messageId" class="Message" foreign-key="id"></key-many-to-one>

</composite-id>

<property name="del" column="del" type="boolean"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

import java.io.Serializable;

public class Like implements Serializable{

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

private UserInfo user;

private Message message;

public Like(UserInfo user) {

super();

this.user = user;

}

public Like() {

super();

// TODO Auto-generated constructor stub

}

private boolean del;

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public Message getMessage() {

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public boolean isDel() {

return del;

}

public void setDel(boolean del) {

this.del = del;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="Message" table="onlineshare.message">

<id name="id" column="id" type="integer">

<generator class="native"></generator>

</id>

<many-to-one name="user" class="UserInfo" column="userId" lazy="false"></many-to-one>

<property name="image" column="image" type="java.sql.Blob"></property>

<property name="videoName" column="videoName" type="string"></property>

<property name="imgOrVideo" column="imgOrVideo" type="boolean"></property>

<property name="text" column="text" type="string"></property>

<property name="time" column="time" type="timestamp"></property>

<property name="location" column="location" type="string"></property>

<property name="del" column="del" type="boolean"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

import java.sql.Blob;

import java.sql.Timestamp;

public class Message{

public Message() {

super();

// TODO Auto-generated constructor stub

}

private int id;

private UserInfo user;

private Blob image = null;

private String videoName = "";

private boolean imgOrVideo;

private String text = "";

private Timestamp time;

private String location = "";

private boolean del;

public Message(Blob image) {

this.image = image;

}

public Message(String videoName) {

this.videoName = videoName;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public Message(int id, UserInfo user, boolean imgOrVideo, String text) {

this.id = id;

this.user = user;

this.imgOrVideo = imgOrVideo;

this.text = text;

// this.time = time;

// this.location = location;

}

public Message(int id) {

this.id = id;

}

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public Blob getImage() {

return image;

}

public void setImage(Blob image) {

this.image = image;

}

public String getVideoName() {

return videoName;

}

public void setVideoName(String videoName) {

this.videoName = videoName;

}

public boolean isImgOrVideo() {

return imgOrVideo;

}

public void setImgOrVideo(boolean imgOrVideo) {

this.imgOrVideo = imgOrVideo;

}

public String getText() {

return text;

}

public void setText(String text) {

this.text = text;

}

public Timestamp getTime() {

return time;

}

public void setTime(Timestamp time) {

this.time = time;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

public boolean isDel() {

return del;

}

public void setDel(boolean del) {

this.del = del;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="Role" table="onlineshare.role">

<id name="id" column="id" type="integer">

<generator class="increment"></generator>

</id>

<property name="name" column="name" type="string"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

public class Role {

private int id;

private String name;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping package="edu.nuc.bean">

<class name="UserInfo" table="onlineshare.userinfo">

<id name="id" column="id" type="integer">

<generator class="increment"></generator>

</id>

<property name="name" column="name" type="string"></property>

<property name="username" column="username" type="string"></property>

<property name="password" column="password" type="string"></property>

<property name="email" column="email" type="string"></property>

<property name="phone" column="phone" type="string"></property>

<property name="photo" column="photo" type="java.sql.Blob"></property>

<property name="breif" column="breif" type="string"></property>

<property name="gender" column="gender" type="boolean"></property>

<property name="state" column="state" type="int"></property>

<many-to-one name="role" class="Role" column="roleId" lazy="false"></many-to-one>

<property name="del" column="del" type="boolean"></property>

</class>

</hibernate-mapping>

package edu.nuc.bean;

import java.sql.Blob;

public class UserInfo{

private int id;

private String name = "";

private String username;

private String password;

private String email = "";

private String phone = "";

private Blob photo;

private String breif = "";

private boolean gender;

private int state;

private Role role;

public Role getRole() {

return role;

}

public UserInfo(Blob photo) {

super();

this.photo = photo;

}

public UserInfo(int id) {

super();

this.id = id;

}

public UserInfo() {

super();

// TODO Auto-generated constructor stub

}

public void setRole(Role role) {

this.role = role;

}

private boolean del;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPhone() {

return phone;

}

public void setPhone(String phone) {

this.phone = phone;

}

public Blob getPhoto() {

return photo;

}

public void setPhoto(Blob photo) {

this.photo = photo;

}

public String getBreif() {

return breif;

}

public void setBreif(String breif) {

this.breif = breif;

}

public boolean isGender() {

return gender;

}

public void setGender(boolean gender) {

this.gender = gender;

}

public int getState() {

return state;

}

public void setState(int state) {

this.state = state;

}

public boolean isDel() {

return del;

}

public void setDel(boolean del) {

this.del = del;

}

}

package edu.nuc.common;

public class Constant {

public static final int DELETE = 123;

public static final int UPDATE = 124;

public static final int INSERT = 125;

public static final int RECOMMAND\_BY\_LIKE = 126;

public static final int RECOMMAND\_BY\_COMMENT = 127;

public static final String RECOMMAND\_MESSAGE\_CACHE = "recommandMessage";

public static final int RECOMMAND\_MESSAGE\_NUM = 15;

public static final int HOMEPAGE\_MESSAGENUM = 10;

public static final int ADMIN = 1;

}

package edu.nuc.common;

public class Tool {

}

package edu.nuc.dao.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.dao.ICommentDao;

import edu.nuc.dao.SysDao;

import edu.nuc.platform.exception.DaoException;

public class CommentDaoImpl extends SysDao implements ICommentDao {

public void insertComment(Comment comment) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(comment);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

public List<Comment> selectCommentsByMessageId(Message message, Pager pager)

throws DaoException {

String hql = "from Comment c where c.message.id=:messageId order by c.id desc";

List<Comment> comments = new ArrayList<Comment>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("messageId", message.getId());

comments = selectByHql(hql, pager, params);

return comments;

}

public List<Comment> selectFiveCommentsByMessId(Message message)

throws DaoException {

Pager pager = new Pager(0, 5);

return selectCommentsByMessageId(message, pager);

}

}

package edu.nuc.dao.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import edu.nuc.bean.Follow;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.common.Constant;

import edu.nuc.dao.IFollowDao;

import edu.nuc.dao.SysDao;

import edu.nuc.platform.exception.DaoException;

public class FollowDaoImpl extends SysDao implements IFollowDao {

public void addFollow(Follow follow) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(follow);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

public void removeFollow(Follow follow) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(follow);

eft.setOperate(Constant.DELETE);

exeUpdate(eft);

}

public boolean isFollow(Follow follow) throws DaoException {

String hql = "from Follow f where f.user.id=:userId and f.follower.id=:followerId";

List<Follow> follows = new ArrayList<Follow>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", follow.getUser().getId());

params.put("followerId", follow.getFollower().getId());

follows = selectByHql(hql, null, params);

return (follows!=null&&follows.size()>0);

}

}

package edu.nuc.dao.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import edu.nuc.bean.Like;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.common.Constant;

import edu.nuc.dao.ILikeDao;

import edu.nuc.dao.SysDao;

import edu.nuc.platform.exception.DaoException;

public class LikeDaoImpl extends SysDao implements ILikeDao {

public int countLikes(Message message) throws DaoException {

String hql = "from Like l where l.message.id=:messageId";

List<Like> likes = new ArrayList<Like>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("messageId", message.getId());

likes = selectByHql(hql, null, params);

return likes.size();

}

public void insertLike(Like like) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(like);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

public boolean isLike(UserInfo user, Message message) throws DaoException {

String hql = "from Like l where l.user.id=:userId and l.message.id=:messageId";

List<Like> likes = new ArrayList<Like>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId());

params.put("messageId", message.getId());

likes = selectByHql(hql, null, params);

return (likes!=null&&likes.size()>0);

}

}

package edu.nuc.dao.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.dao.IMessageDao;

import edu.nuc.dao.SysDao;

import edu.nuc.platform.exception.DaoException;

public class MessageDaoImpl extends SysDao implements IMessageDao {

@SuppressWarnings("unchecked")

public List<Message> selectMessagesByLikes(Pager pager) throws DaoException {

String hql = "select l.message from Like l where l.message.del=0 GROUP BY l.message.id ORDER BY COUNT(l.user.id) desc";

List<Message> messages = new ArrayList<Message>();

messages = selectByHql(hql, pager, null);

//selectByHql(hql, pager, null);

return messages;

}

@SuppressWarnings("unchecked")

public Message selectPicById(Message message) throws DaoException {

String hql = "select new Message(image) from Message m where m.id=:messageId";

List<Message> messages = new ArrayList<Message>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("messageId", message.getId());

messages = selectByHql(hql, null, params);

return messages.get(0);

}

@SuppressWarnings("unchecked")

public List<Message> selectUserMessages(UserInfo user, Pager pager)

throws DaoException {

String hql = "select new Message(m.id, m.user, m.imgOrVideo, m.text) from Message m where m.user.id=:userId and m.del=0 order by m.time desc";

List<Message> messages = new ArrayList<Message>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId());

messages = selectByHql(hql, pager, params);

return messages;

}

@SuppressWarnings("unchecked")

public List<Message> selectFollowingMessages(UserInfo user, Pager pager)

throws DaoException {

String hql = "select new Message(m.id, m.user, m.imgOrVideo, m.text) from Message m,Follow f where f.follower.id=:userId and m.user.id=f.user.id and m.del=0 order by m.time desc";

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId());

List<Message> messages = new ArrayList<Message>();

messages = selectByHql(hql, pager, params);

return messages;

}

public void insertMessage(Message message) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(message);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

public Message selectMessageById(Message message) throws DaoException {

String hql = "select new Message(m.id, m.user, m.imgOrVideo, m.text) from Message m where m.id=:messageId";

List<Message> messages = new ArrayList<Message>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("messageId", message.getId());

messages = selectByHql(hql, null, params);

return messages.get(0);

}

public void deleteMessage(Message message) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(message);

eft.setOperate(Constant.DELETE);

exeUpdate(eft);

}

public int selectMessagesCount(UserInfo user) throws DaoException {

String hql = "select count(m.id) from Message m";

Map<String, Object> params = null;

if (user != null && user.getUsername() != null && !"".equals(user.getUsername())) {

hql += " where m.user.username like :username";

params = new HashMap<String, Object>();

params.put("username", user.getUsername());

}

int count = Integer.parseInt(selectByHql(hql, null, params).get(0).toString());

return count;

}

public List<Message> selectMessages(Pager pager,UserInfo user) throws DaoException

{

String hql = "from Message m";

List<Message> messages = new ArrayList<Message>();

Map<String, Object> params = null;

if (user != null && user.getUsername() != null && !"".equals(user.getUsername())) {

hql += " where m.user.username like :username";

params = new HashMap<String, Object>();

params.put("username", "%"+user.getUsername()+"%");

}

hql+=" order by m.time desc";

messages = selectByHql(hql, pager, params);

return messages;

}

public void managerChangeMessages(List<Message> messages)

throws DaoException {

EntityForTx[] efts = new EntityForTx[messages.size()];

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

efts[i] = new EntityForTx();

efts[i].setOperate(Constant.UPDATE);

efts[i].setObject(messages.get(i));

}

exeUpdate(efts);

}

public Message selectMessageById2(Message message) throws DaoException {

String hql = "from Message m where m.id=:messageId";

List<Message> messages = new ArrayList<Message>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("messageId", message.getId());

messages = selectByHql(hql, null, params);

return messages.get(0);

}

public List<Message> selectMoreFollowingMessages(UserInfo user,

Pager pager, Message message) throws DaoException {

String hql = "select new Message(m.id, m.user, m.imgOrVideo, m.text) from Message m,Follow f where f.follower.id=:userId and m.user.id=f.user.id and m.del=0 and m.id>:messageId order by m.time desc";

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId());

params.put("messageId", message.getId());

List<Message> messages = new ArrayList<Message>();

messages = selectByHql(hql, pager, params);

return messages;

}

}

package edu.nuc.dao.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.dao.IUserInfoDao;

import edu.nuc.dao.SysDao;

import edu.nuc.platform.exception.DaoException;

public class UserInfoDaoImpl extends SysDao implements IUserInfoDao {

@SuppressWarnings("unchecked")

public UserInfo selectForLogin(UserInfo user) throws DaoException {

List<UserInfo> al = new ArrayList<UserInfo>();

String hql = "from UserInfo u where u.username=:username and u.password=:password and u.del=0";

Map<String, Object> params = new HashMap<String, Object>();

params.put("username", user.getUsername());

params.put("password", user.getPassword());

al = selectByHql(hql,null,params);

UserInfo loginUser = (al != null && al.size() > 0) ? al.get(0) : null;

return loginUser;

}

public UserInfo selectUserById(UserInfo user) throws DaoException {

List<UserInfo> al = new ArrayList<UserInfo>();

String hql = "from UserInfo u where u.id=:useId";

Map<String, Object> params = new HashMap<String, Object>();

params.put("useId", user.getId());

al = selectByHql(hql,null,params);

UserInfo userInfo = (al != null && al.size() > 0) ? al.get(0) : null;

return userInfo;

}

public void regist(UserInfo user) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(user);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

@SuppressWarnings("unchecked")

public UserInfo selectPicById(UserInfo user) throws DaoException {

String hql = "select new UserInfo(photo) from UserInfo u where u.id =:userId";

List<UserInfo> users = new ArrayList<UserInfo>();

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId());

users = selectByHql(hql, null, params);

return users.get(0);

}

public void uploadPhoto(UserInfo user) throws DaoException {

EntityForTx eft = new EntityForTx();

eft.setObject(user);

eft.setOperate(Constant.INSERT);

exeUpdate(eft);

}

public List<UserInfo> selectFollower(UserInfo user) throws DaoException {

List<UserInfo> followers = new ArrayList<UserInfo>();

String hql = "select u from UserInfo u,Follow f where u.id=f.follower.id and f.user.id=:userId";

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId() );

followers = selectByHql(hql,null,params);

return followers;

}

public List<UserInfo> selectIFollow(UserInfo user) throws DaoException {

List<UserInfo> iFollows = new ArrayList<UserInfo>();

String hql = "select u from UserInfo u,Follow f where u.id=f.user.id and f.follower.id=:userId";

Map<String, Object> params = new HashMap<String, Object>();

params.put("userId", user.getId() );

iFollows = selectByHql(hql,null,params);

return iFollows;

}

public List<UserInfo> selectUserByName(UserInfo user, Pager pager) throws DaoException {

List<UserInfo> users = new ArrayList<UserInfo>();

String hql = "from UserInfo u";

Map<String, Object> params = null;

if (user != null && user.getUsername() != null && !"".equals(user.getUsername())) {

hql += " where u.username like :username";

params = new HashMap<String, Object>();

params.put("username", "%"+user.getUsername()+"%");

}

users = selectByHql(hql,pager,params);

return users;

}

public int selectUserCount(UserInfo user) throws DaoException {

String hql = "select count(u.id) from UserInfo u";

Map<String, Object> params = null;

if (user != null && user.getUsername() != null && !"".equals(user.getUsername())) {

hql += " where u.username like :username";

params = new HashMap<String, Object>();

params.put("username", user.getUsername());

}

int count = Integer.parseInt(selectByHql(hql, null, params).get(0).toString());

return count;

}

public void managerChangeUsers(List<UserInfo> users)

throws DaoException {

EntityForTx[] efts = new EntityForTx[users.size()];

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

efts[i] = new EntityForTx();

efts[i].setOperate(Constant.UPDATE);

efts[i].setObject(users.get(i));

}

exeUpdate(efts);

}

public void updateInfo(UserInfo user) throws DaoException {

EntityForTx efts = new EntityForTx();

efts.setOperate(Constant.UPDATE);

efts.setObject(user);

exeUpdate(efts);

}

}

package edu.nuc.dao;

import java.util.List;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.DaoException;

public interface ICommentDao {

void insertComment(Comment comment) throws DaoException;

List<Comment> selectCommentsByMessageId(Message message, Pager pager) throws DaoException;

List<Comment> selectFiveCommentsByMessId(Message message) throws DaoException;

}

package edu.nuc.dao;

import edu.nuc.bean.Follow;

import edu.nuc.platform.exception.DaoException;

public interface IFollowDao

{

void addFollow(Follow follow) throws DaoException;

void removeFollow(Follow follow) throws DaoException;

boolean isFollow(Follow follow) throws DaoException;

}

package edu.nuc.dao;

import edu.nuc.bean.Like;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.platform.exception.DaoException;

public interface ILikeDao {

void insertLike(Like like) throws DaoException;

//if like is null, represent the user have not like the message

boolean isLike(UserInfo user, Message message) throws DaoException;

int countLikes(Message message) throws DaoException;

}

package edu.nuc.dao;

import java.util.List;

import javax.xml.registry.infomodel.User;

import com.sun.jmx.snmp.Timestamp;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.DaoException;

public interface IMessageDao {

void insertMessage(Message message) throws DaoException;

//user follower's messages

List<Message> selectFollowingMessages(UserInfo user,Pager pager) throws DaoException;

List<Message> selectMoreFollowingMessages(UserInfo user,Pager pager, Message message) throws DaoException;

// one's messages

List<Message> selectUserMessages(UserInfo user,Pager pager) throws DaoException;

//as recomend

List<Message> selectMessagesByLikes(Pager pager) throws DaoException;

Message selectPicById(Message message) throws DaoException;

Message selectMessageById(Message message) throws DaoException;

Message selectMessageById2(Message message) throws DaoException;

List<Message> selectMessages(Pager pager,UserInfo user) throws DaoException;

void deleteMessage(Message message) throws DaoException;

int selectMessagesCount(UserInfo user) throws DaoException;

void managerChangeMessages(List<Message> messages) throws DaoException;

}

package edu.nuc.dao;

import java.util.List;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.DaoException;

public interface IUserInfoDao

{

/\*\*

\* select userinfo for login

\* @param user must contains username and password

\*/

UserInfo selectForLogin(UserInfo user) throws DaoException;

void regist(UserInfo user) throws DaoException;

UserInfo selectPicById(UserInfo user) throws DaoException;

void uploadPhoto(UserInfo user) throws DaoException;

List<UserInfo> selectFollower(UserInfo user) throws DaoException;

List<UserInfo> selectIFollow(UserInfo user) throws DaoException;

List<UserInfo> selectUserByName(UserInfo user, Pager pager) throws DaoException;

UserInfo selectUserById(UserInfo user) throws DaoException;

int selectUserCount(UserInfo user) throws DaoException;

void managerChangeUsers(List<UserInfo> newUsers) throws DaoException;

void updateInfo(UserInfo user) throws DaoException;

}

package edu.nuc.dao;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import org.hibernate.Query;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import edu.nuc.bean.help.EntityForTx;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.platform.exception.DaoException;

public class SysDao {

private static SessionFactory sessionFactory;

public SessionFactory getSessionFactory() {

return sessionFactory;

}

@SuppressWarnings("static-access")

public void setSessionFactory(SessionFactory sessionFactory) {

this.sessionFactory = sessionFactory;

}

/\*\*

\* get entity info by it's id

\* @param id entity id in the database

\* @param c entity class type

\* @return entity model

\* @throws DaoException

\*/

@SuppressWarnings("unchecked")

public Object selectInfoById(Serializable id, Class c) throws DaoException {

Object obj = null;

Session session = sessionFactory.getCurrentSession();

Transaction tx = session.beginTransaction();

try {

tx.begin();

obj = session.get(c, id);

tx.commit();

} catch (Exception e) {

e.printStackTrace();

if (null != tx) {

tx.rollback();

}

throw new DaoException("Get entity info by id exception", e);

}

return obj;

}

@SuppressWarnings("unchecked")

public List selectByHql(String hql, Pager pager,Map<String,Object> params)

throws DaoException {

List al = new ArrayList();

Session session = sessionFactory.getCurrentSession();

Transaction tx = session.beginTransaction();

try {

tx.begin();

Query query = session.createQuery(hql);

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

{

query.setProperties(params);

}

if (pager != null) {

query.setFirstResult((pager.getPageNum() - 1)

\* pager.getPageCount());

query.setMaxResults(pager.getPageCount());

}

al = query.list();

tx.commit();

} catch (Exception e) {

e.printStackTrace();

if (null != tx) {

tx.rollback();

}

throw new DaoException("Get all exception", e);

}

return al;

}

public boolean exeUpdate(EntityForTx... entitys) throws DaoException {

Session session = sessionFactory.getCurrentSession();

Transaction tx = session.beginTransaction();

try {

tx.begin();

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

switch (entitys[i].getOperate()) {

case Constant.DELETE:

session.delete(entitys[i].getObject());

break;

case Constant.UPDATE:

session.update(entitys[i].getObject());

break;

case Constant.INSERT:

session.saveOrUpdate(entitys[i].getObject());

break;

}

}

tx.commit();

} catch (Exception e) {

e.printStackTrace();

if (null != tx) {

tx.rollback();

}

throw new DaoException("Execute operation exception", e);

}

return true;

}

}

package edu.nuc.platform.cache;

import java.util.ArrayList;

import java.util.List;

import freemarker.cache.StrongCacheStorage;

public class Cache {

private static StrongCacheStorage cacheTable;

public Cache()

{

if (cacheTable == null) {

cacheTable = new StrongCacheStorage();

}

}

public <T> List<T> applyCache(String key)

{

List<T> cacheBlock = new ArrayList<T>();

cacheTable.put(key, cacheBlock);

return cacheBlock;

}

public <T> List<T> getCache(String key)

{

if (cacheTable.get(key) != null) {

return (List<T>)cacheTable.get(key);

}

return null;

}

}

package edu.nuc.platform.exception;

public class DaoException extends Exception{

private static final long serialVersionUID = 5754691931087500492L;

public DaoException() {

super();

// TODO Auto-generated constructor stub

}

public DaoException(String message, Throwable cause) {

super(message, cause);

// TODO Auto-generated constructor stub

}

public DaoException(String message) {

super(message);

// TODO Auto-generated constructor stub

}

public DaoException(Throwable cause) {

super(cause);

// TODO Auto-generated constructor stub

}

}

package edu.nuc.platform.exception;

public class ServiceException extends Exception{

private static final long serialVersionUID = 9062234127038712178L;

public ServiceException() {

super();

// TODO Auto-generated constructor stub

}

public ServiceException(String message, Throwable cause) {

super(message, cause);

// TODO Auto-generated constructor stub

}

public ServiceException(String message) {

super(message);

// TODO Auto-generated constructor stub

}

public ServiceException(Throwable cause) {

super(cause);

// TODO Auto-generated constructor stub

}

}

package edu.nuc.service.impl;

import java.util.List;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.help.Pager;

import edu.nuc.dao.ICommentDao;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.ICommentService;

public class CommentServiceImpl implements ICommentService {

private ICommentDao commentDao;

public ICommentDao getCommentDao() {

return commentDao;

}

public void setCommentDao(ICommentDao commentDao) {

this.commentDao = commentDao;

}

public List<Comment> viewCommentsByMessageId(Message message, Pager pager)

throws ServiceException {

List<Comment> comments = null;

try {

comments = commentDao.selectCommentsByMessageId(message, pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return comments;

}

public void addComment(Comment comment) throws ServiceException {

try {

commentDao.insertComment(comment);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public List<Comment> getFiveComments(Message message) throws ServiceException {

List<Comment> comments = null;

try {

comments = commentDao.selectFiveCommentsByMessId(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return comments;

}

}

package edu.nuc.service.impl;

import edu.nuc.bean.Follow;

import edu.nuc.dao.IFollowDao;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.IFollowService;

public class FollowServiceImpl implements IFollowService {

private IFollowDao followDao;

public IFollowDao getFollowDao() {

return followDao;

}

public void setFollowDao(IFollowDao followDao) {

this.followDao = followDao;

}

public void addFollow(Follow follow) throws ServiceException {

try {

followDao.addFollow(follow);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public boolean isFollow(Follow follow) throws ServiceException {

try {

return followDao.isFollow(follow);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return false;

}

public void removeFollow(Follow follow) throws ServiceException {

try {

followDao.removeFollow(follow);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

package edu.nuc.service.impl;

import edu.nuc.bean.Like;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.dao.ILikeDao;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.ILikeService;

public class LikeServiceImpl implements ILikeService {

private ILikeDao likeDao;

public ILikeDao getLikeDao() {

return likeDao;

}

public void setLikeDao(ILikeDao likeDao) {

this.likeDao = likeDao;

}

public void addLike(Like like) throws ServiceException {

try {

likeDao.insertLike(like);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public boolean isLike(UserInfo user, Message message)

throws ServiceException {

// TODO Auto-generated method stub

try {

return likeDao.isLike(user, message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return false;

}

}

package edu.nuc.service.impl;

import java.io.IOException;

import java.io.InputStream;

import java.sql.Blob;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import edu.nuc.common.Constant;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.dao.ICommentDao;

import edu.nuc.dao.ILikeDao;

import edu.nuc.dao.IMessageDao;

import edu.nuc.dao.impl.MessageDaoImpl;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.IMessageService;

import edu.nuc.service.model.MessageItem;

public class MessageServiceImpl implements IMessageService {

public MessageServiceImpl() {

super();

// TODO Auto-generated constructor stub

}

private IMessageDao messageDao;

private ILikeDao likeDao;

private ICommentDao commentDao;

public MessageServiceImpl(MessageDaoImpl messageDaoImpl) {

this.messageDao = messageDaoImpl;

}

public ICommentDao getCommentDao() {

return commentDao;

}

public void setCommentDao(ICommentDao commentDao) {

this.commentDao = commentDao;

}

public ILikeDao getLikeDao() {

return likeDao;

}

public void setLikeDao(ILikeDao likeDao) {

this.likeDao = likeDao;

}

public IMessageDao getMessageDao() {

return messageDao;

}

public void setMessageDao(IMessageDao messageDao) {

this.messageDao = messageDao;

}

// public List<Message> viewFollowingMessage(UserInfo user,Pager pager) throws ServiceException

// {

// List<Message> messages = new ArrayList<Message>();

// if (user != null) {

// try {

// messages = messageDao.selectFollowerMessages(user, pager);

// } catch (Exception e) {

// throw new ServiceException(e);

// }

// }

//

//

// return messages;

// }

public List<MessageItem> viewFollowingMessage(UserInfo user,Pager pager) throws ServiceException

{

List<Message> messages = new ArrayList<Message>();

if (user != null) {

try {

messages = messageDao.selectFollowingMessages(user, pager);

} catch (Exception e) {

e.printStackTrace();

}

}

List<MessageItem> messageList = new ArrayList<MessageItem>();

for (Message message : messages) {

MessageItem messageItem = ConvertMessageToItem(user, message);

List<Comment> fiveComments = null;

try {

fiveComments = commentDao.selectFiveCommentsByMessId(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

messageItem.setFiveComments(fiveComments);

messageList.add(messageItem);

}

return messageList;

}

private MessageItem ConvertMessageToItem(UserInfo user,Message message) throws ServiceException

{

MessageItem messageItem = new MessageItem();

int likeNum;

try {

likeNum = likeDao.countLikes(message);

} catch (DaoException e) {

e.printStackTrace();

throw new ServiceException("");

}

boolean isViewerlike;

try {

isViewerlike = likeDao.isLike(user, message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

throw new ServiceException("");

}

Pager pager = new Pager(1, 5);

List<Comment> comments;

try {

comments = commentDao.selectCommentsByMessageId(message, pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

throw new ServiceException("");

}

messageItem.setMessage(message);

messageItem.setViewerLike(isViewerlike);

messageItem.setLikeNum(likeNum);

messageItem.setFiveComments(comments);

return messageItem;

}

public byte[] getPicByMessageId(Message message) throws ServiceException {

Blob img = null;

InputStream is = null;

if (message != null) {

try {

img = messageDao.selectPicById(message).getImage();

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

is = img.getBinaryStream();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

byte[] buffer = new byte[1024\*1024];

try {

is.read(buffer);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

int c = buffer.length;

return buffer;

}

public void uploadMessage(Message message) throws ServiceException {

if(message != null)

{

try {

messageDao.insertMessage(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

public List<Message> getRecommadMessage(Pager pager, int option)

throws ServiceException {

if (option == Constant.RECOMMAND\_BY\_LIKE) {

try {

return messageDao.selectMessagesByLikes(pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

return null;

}

public List<MessageItem> getUserMessage(UserInfo user, Pager pager)

throws ServiceException {

List<Message> messages = new ArrayList<Message>();

try {

messages = messageDao.selectUserMessages(user, pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

List<MessageItem> messageList = new ArrayList<MessageItem>();

for (Message message : messages) {

messageList.add(ConvertMessageToItem(user, message));

}

return messageList;

}

public MessageItem getMessageById(Message message,UserInfo user)

throws ServiceException {

Message m = null;

try {

m = messageDao.selectMessageById(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return ConvertMessageToItem(user,m);

}

public void removeMessage(Message message) throws ServiceException {

if(message != null)

{

try {

messageDao.deleteMessage(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

public List<Message> getMessages(Pager pager, UserInfo user) throws ServiceException {

try {

return messageDao.selectMessages(pager, user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

public int getMessagesCount(UserInfo user) throws ServiceException {

try {

return messageDao.selectMessagesCount(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return 0;

}

public void changeMessagesStatus(List<Message> messages, boolean status)

throws ServiceException {

List<Message> newMessages = new ArrayList<Message>();

for (Message message : messages) {

try {

Message newMessage = messageDao.selectMessageById2(message);

newMessage.setDel(true);

newMessages.add(newMessage);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

messageDao.managerChangeMessages(newMessages);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public List<MessageItem> viewMoreFollowingMessage(UserInfo user,

Pager pager, Message m) throws ServiceException {

List<Message> messages = new ArrayList<Message>();

if (user != null) {

try {

messages = messageDao.selectMoreFollowingMessages(user, pager,m);

} catch (Exception e) {

e.printStackTrace();

}

}

List<MessageItem> messageList = new ArrayList<MessageItem>();

for (Message message : messages) {

MessageItem messageItem = ConvertMessageToItem(user, message);

List<Comment> fiveComments = null;

try {

fiveComments = commentDao.selectFiveCommentsByMessId(message);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

messageItem.setFiveComments(fiveComments);

messageList.add(messageItem);

}

return messageList;

}

}

package edu.nuc.service.impl;

import java.io.IOException;

import java.io.InputStream;

import java.sql.Blob;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.dao.IUserInfoDao;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.IUserService;

public class UserServiceImpl implements IUserService {

private IUserInfoDao userDao;

public IUserInfoDao getUserDao() {

return userDao;

}

public void setUserDao(IUserInfoDao userDao) {

this.userDao = userDao;

}

public UserInfo login(UserInfo user) throws ServiceException {

UserInfo onlineUser = null;

try {

onlineUser = userDao.selectForLogin(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return onlineUser;

}

public void regist(UserInfo user) throws ServiceException {

try {

userDao.regist(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

throw new ServiceException();

}

}

public byte[] getPhotoById(UserInfo user) throws ServiceException {

Blob img = null;

InputStream is = null;

if (user != null) {

try {

img = userDao.selectPicById(user).getPhoto();

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

is = img.getBinaryStream();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

byte[] buffer = new byte[1024\*1024];

try {

is.read(buffer);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

int c = buffer.length;

return buffer;

}

public void uploadPhoto(UserInfo user) throws ServiceException {

try {

userDao.uploadPhoto(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public List<UserInfo> getFollower(UserInfo user) throws ServiceException {

try {

return userDao.selectFollower(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

public List<UserInfo> getIFollow(UserInfo user) throws ServiceException {

// TODO Auto-generated method stub

try {

return userDao.selectIFollow(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

public UserInfo getUserById(UserInfo user) throws ServiceException {

// TODO Auto-generated method stub

try {

return userDao.selectUserById(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return user;

}

public List<UserInfo> getUserByName(UserInfo user, Pager pager) throws ServiceException {

// TODO Auto-generated method stub

try {

return userDao.selectUserByName(user, pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

public List<UserInfo> getUser(UserInfo user, Pager pager) throws ServiceException {

try {

return userDao.selectUserByName(user, pager);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

public int getUserCount(UserInfo user) throws ServiceException {

try {

return userDao.selectUserCount(user);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return 0;

}

public void changeUsersStatus(List<UserInfo> users, boolean status)

throws ServiceException {

List<UserInfo> newUsers = new ArrayList<UserInfo>();

for (UserInfo user : users) {

try {

UserInfo newUser = userDao.selectUserById(user);

newUser.setDel(status);

newUsers.add(newUser);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

try {

userDao.managerChangeUsers(newUsers);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void updateInfo(UserInfo user) throws ServiceException {

try {

UserInfo newUser = userDao.selectUserById(user);

newUser.setName(user.getName());

newUser.setUsername(user.getUsername());

newUser.setPassword(user.getPassword());

newUser.setGender(user.isGender());

newUser.setEmail(user.getEmail());

newUser.setPhone(user.getPhone());

newUser.setBreif(user.getBreif());

userDao.updateInfo(newUser);

} catch (DaoException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

package edu.nuc.service.model;

import java.util.ArrayList;

import java.util.List;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

public class MessageItem {

private Message message;

private boolean isViewerLike; // weather the user like the message

private int likeNum;

private List<Comment> fiveComments = new ArrayList<Comment>();

public Message getMessage() {

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public boolean isViewerLike() {

return isViewerLike;

}

public void setViewerLike(boolean isViewerLike) {

this.isViewerLike = isViewerLike;

}

public int getLikeNum() {

return likeNum;

}

public void setLikeNum(int likeNum) {

this.likeNum = likeNum;

}

public List<Comment> getFiveComments() {

return fiveComments;

}

public void setFiveComments(List<Comment> fiveComments) {

this.fiveComments = fiveComments;

}

}

package edu.nuc.service;

import java.util.List;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.model.MessageItem;

public interface ICommentService {

List<Comment> viewCommentsByMessageId(Message message, Pager pager) throws ServiceException;

void addComment(Comment comment) throws ServiceException;

List<Comment> getFiveComments(Message message) throws ServiceException;

}

package edu.nuc.service;

import edu.nuc.bean.Follow;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

public interface IFollowService {

void addFollow(Follow follow) throws ServiceException;

void removeFollow(Follow follow) throws ServiceException;

boolean isFollow(Follow follow) throws ServiceException;

}

package edu.nuc.service;

import edu.nuc.bean.Like;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.platform.exception.DaoException;

import edu.nuc.platform.exception.ServiceException;

public interface ILikeService {

void addLike(Like like) throws ServiceException;

boolean isLike(UserInfo user, Message message) throws ServiceException;

}

package edu.nuc.service;

import java.util.List;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.model.MessageItem;

public interface IMessageService {

List<MessageItem> viewFollowingMessage(UserInfo user, Pager pager) throws ServiceException;

List<MessageItem> viewMoreFollowingMessage(UserInfo user, Pager pager, Message message) throws ServiceException;

byte[] getPicByMessageId(Message message) throws ServiceException;

void uploadMessage(Message message) throws ServiceException;

List<Message> getRecommadMessage(Pager pager, int option) throws ServiceException;

List<MessageItem> getUserMessage(UserInfo user, Pager pager) throws ServiceException;

MessageItem getMessageById(Message message,UserInfo user) throws ServiceException;

void removeMessage(Message message) throws ServiceException;

List<Message> getMessages(Pager pager, UserInfo user) throws ServiceException;

int getMessagesCount(UserInfo user) throws ServiceException;

void changeMessagesStatus(List<Message> messages, boolean status) throws ServiceException;

}

package edu.nuc.service;

import java.util.List;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.ServiceException;

public interface IUserService {

UserInfo login(UserInfo user) throws ServiceException;

void regist(UserInfo user) throws ServiceException;

byte[] getPhotoById(UserInfo user) throws ServiceException;

void uploadPhoto(UserInfo user) throws ServiceException;

List<UserInfo> getFollower(UserInfo user) throws ServiceException;

List<UserInfo> getIFollow(UserInfo user) throws ServiceException;

List<UserInfo> getUserByName(UserInfo user, Pager pager) throws ServiceException;

UserInfo getUserById(UserInfo user) throws ServiceException;

int getUserCount(UserInfo user) throws ServiceException;

void changeUsersStatus(List<UserInfo> users, boolean status) throws ServiceException;

void updateInfo(UserInfo user) throws ServiceException;

}

package edu.nuc.service;

import java.io.IOException;

import java.io.InputStream;

import java.sql.Blob;

import java.sql.SQLException;

import java.util.List;

import edu.nuc.bean.Message;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.dao.impl.MessageDaoImpl;

import edu.nuc.platform.cache.Cache;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.impl.MessageServiceImpl;

public class MessageCacheManager implements Runnable{

private static List<Message> messageCacheBlock;

private static Pager pager;

private static IMessageService messageService;

static {

messageService = new MessageServiceImpl(new MessageDaoImpl());

Cache cache = new Cache();

messageCacheBlock = cache.applyCache(Constant.RECOMMAND\_MESSAGE\_CACHE);

pager = new Pager(0, Constant.RECOMMAND\_MESSAGE\_NUM);

new Thread(new MessageCacheManager()).start();

}

public void run() {

while(true)

{

try {

Thread.sleep(10000);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

try {

refreshRecomandMessage();

} catch (ServiceException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

try {

Thread.sleep(25000);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

private void refreshRecomandMessage() throws ServiceException

{

if (messageCacheBlock == null) {

messageCacheBlock = new Cache().applyCache(Constant.RECOMMAND\_MESSAGE\_CACHE);

}

List<Message> recommandMessages = MessageCacheManager.messageService.getRecommadMessage(pager, Constant.RECOMMAND\_BY\_LIKE);

messageCacheBlock.addAll(recommandMessages);

if (messageCacheBlock.size() > Constant.RECOMMAND\_MESSAGE\_NUM) {

for (int i = 0; i < Constant.RECOMMAND\_MESSAGE\_NUM; i++) {

messageCacheBlock.remove(0);

}

}

}

public List<Message> getRecomandMessages() throws ServiceException

{

return messageCacheBlock;

}

public byte[] getRecomandPicReadom() throws ServiceException {

int size = messageCacheBlock.size();

byte[] buffer = new byte[1024\*1024];

double ra = Math.random()\*100;

int id = (int) (ra%size);

Message message = messageCacheBlock.get(id);

if (message != null) {

Blob img = null;

InputStream is = null;

img = message.getImage();

try {

is = img.getBinaryStream();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

try {

is.read(buffer);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

int c = buffer.length;

}

return buffer;

}

}

package edu.nuc.web.action;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.struts2.ServletActionContext;

import com.opensymphony.xwork2.ActionContext;

public class BaseAction {

protected HttpServletResponse getResponse() {

ActionContext ac = ActionContext.getContext();

HttpServletResponse response = (HttpServletResponse)

ac.get(org.apache.struts2.StrutsStatics.HTTP\_RESPONSE);

return response;

}

protected HttpServletRequest getRequest() {

ActionContext ac = ActionContext.getContext();

HttpServletRequest request = (HttpServletRequest)

ac.get(org.apache.struts2.StrutsStatics.HTTP\_REQUEST);

return request;

}

protected HttpSession getSession() {

HttpSession session = ServletActionContext.getRequest().getSession();

return session;

}

}

package edu.nuc.web.action;

import java.io.PrintWriter;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import com.opensymphony.xwork2.ActionContext;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.service.ICommentService;

import edu.nuc.web.manage.CommentManagement;

import edu.nuc.web.manage.MessageManagement;

public class CommentAction extends BaseAction{

private Comment comment;

private Message message;

private CommentManagement commentManagement;

public CommentManagement getCommentManagement() {

return commentManagement;

}

public void setCommentManagement(CommentManagement commentManagement) {

this.commentManagement = commentManagement;

}

public Message getMessage() {

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public Comment getComment() {

return comment;

}

public void setComment(Comment comment) {

this.comment = comment;

}

public void toAddCommentPage() throws Exception{

getSession().setAttribute("currentMessage", getMessage());

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"()");

out.close();

}

public void addComment() throws Exception{

HttpSession session = getSession();

Message message = (Message)session.getAttribute("currentMessage");

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

commentManagement.addComment(user, message, comment);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"()");

out.close();

}

public void getComments() throws Exception{

Message message = (Message)getSession().getAttribute("currentMessage");

String data = commentManagement.viewComments(message);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+data+")");

out.close();

}

}

package edu.nuc.web.action;

import java.io.PrintWriter;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import edu.nuc.bean.Like;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.service.ILikeService;

public class LikeAction extends BaseAction {

private ILikeService likeService;

private Message message;

public Message getMessage() {

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public ILikeService getLikeService() {

return likeService;

}

public void setLikeService(ILikeService likeService) {

this.likeService = likeService;

}

public void addLike() throws Exception{

HttpSession session = getSession();

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

Like like = new Like();

like.setMessage(message);

like.setUser(user);

if (!likeService.isLike(user, message)) {

likeService.addLike(like);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+message.getId()+")");

out.close();

}

}

public void removeLike() throws Exception{

HttpSession session = getSession();

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

// Like like = new Like();

// like.setMessage(message);

// like.setUser(user);

//

// if (likeService.isLike(user, message)) {

// likeService.addLike(like);

// String methodName = getRequest().getParameter("methodName");

//

// HttpServletResponse response = getResponse();

// response.setContentType("text/plain");

// PrintWriter out = response.getWriter();

// out.print(methodName+"("+message.getId()+")");

// out.close();

// }

}

}

package edu.nuc.web.action;

import java.io.File;

import java.io.OutputStream;

import java.io.PrintWriter;

import javax.servlet.http.HttpServletResponse;

import org.omg.CORBA.Request;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.web.manage.MessageManagement;

import edu.nuc.web.model.MessageForPage;

public class MessageAction extends BaseAction{

private Message message;

private File upload;

private String uploadContentType;

private String uploadFileName;

private MessageManagement messageManagement;

private UserInfo user;

private Pager pager;

private String ids;

public String getIds() {

return ids;

}

public void setIds(String ids) {

this.ids = ids;

}

public Pager getPager() {

return pager;

}

public void setPager(Pager pager) {

this.pager = pager;

}

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public MessageManagement getMessageManagement() {

return messageManagement;

}

public void setMessageManagement(MessageManagement messageManagement) {

this.messageManagement = messageManagement;

}

public File getUpload() {

return upload;

}

public void setUpload(File upload) {

this.upload = upload;

}

public String getUploadContentType() {

return uploadContentType;

}

public void setUploadContentType(String uploadContentType) {

this.uploadContentType = uploadContentType;

}

public String getUploadFileName() {

return uploadFileName;

}

public void setUploadFileName(String uploadFileName) {

this.uploadFileName = uploadFileName;

}

public Message getMessage() {

message = new Message();

message.setText(getUploadFileName());

return message;

}

public void setMessage(Message message) {

this.message = message;

}

public void viewHomePage() throws Exception{

UserInfo user = (UserInfo) getSession().getAttribute("user");

// user = new UserInfo();

// user.setId(1);

String json = messageManagement.getHomePageData(user);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public void viewHomePageMore() throws Exception{

UserInfo user = (UserInfo) getSession().getAttribute("user");

// user = new UserInfo();

// user.setId(1);

String json = messageManagement.getMoreHomePageData(user,message);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public void removeMessage() throws Exception

{

messageManagement.removeMessage(message);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+message.getId()+")");

out.close();

}

public void getPicById() throws Exception{

byte[] data = messageManagement.getPicById(message);

HttpServletResponse response = getResponse();

response.setContentType("image/\*");

OutputStream toClient = response.getOutputStream(); // �õ���ͻ���������������ݵĶ���

toClient.write(data); // �������

toClient.close();

}

public void getRecommandPicReadom() throws Exception{

byte[] data = messageManagement.getRecomandPicReadom();

HttpServletResponse response = getResponse();

response.setContentType("image/\*");

OutputStream toClient = response.getOutputStream(); // �õ���ͻ���������������ݵĶ���

toClient.write(data); // �������

toClient.close();

}

public void getRecomandMessage() throws Exception{

String json = messageManagement.getRecomandMessageIds();

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public void getRecommandPicById() throws Exception{

byte[] data = messageManagement.getRecomandPicById(message.getId());

HttpServletResponse response = getResponse();

response.setContentType("image/\*");

OutputStream toClient = response.getOutputStream(); // �õ���ͻ���������������ݵĶ���

toClient.write(data); // �������

toClient.close();

}

public void uploadMessage() throws Exception{

File uploadPic = getUpload();

UserInfo user = (UserInfo) getSession().getAttribute("user");

messageManagement.uploadMessage(user, getMessage(), uploadPic);

}

public void getMyMessage() throws Exception{

UserInfo user = (UserInfo) getSession().getAttribute("user");

String json = messageManagement.getUserMessage(user);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public void getUserMessage() throws Exception{

UserInfo manager = (UserInfo)getSession().getAttribute("manager");

UserInfo other = (UserInfo) getSession().getAttribute("other");

String json = messageManagement.getUserMessage(other);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public void getMessageById() throws Exception{

UserInfo user = (UserInfo) getSession().getAttribute("user");

String json = messageManagement.getMessageById(message, user);

String methodName = getRequest().getParameter("methodName");

HttpServletResponse response = getResponse();

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName+"("+json+")");

out.close();

}

public String managerGetMessage() throws Exception

{

UserInfo manager = (UserInfo)getSession().getAttribute("manager");

if (manager == null) {

return "managerLoginFaild";

}

String pageNuum = getRequest().getParameter("pager.pageNum");

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

int pageNum = Integer.parseInt(pageNuum);

if (pageNum <= 1) {

pager = new Pager(1,12);

}else

{

pager = new Pager(pageNum, 12);

}

}else

{

pager = new Pager(1,12);

}

MessageForPage mfg = messageManagement.getMessages(pager,user);

int pageCount = (mfg.getCount() / 12) + 1;

int nowFrom = (pager.getPageNum()-1) \* pager.getPageCount() + 1;

int nowTo = pager.getPageNum() >= pageCount ? mfg.getCount() : (nowFrom + 11);

int forward = (pager.getPageNum()-1)<=1?1:(pager.getPageNum()-1);

int next = (pager.getPageNum()+1)>=pageCount?pageCount:(pager.getPageNum()+1);

getRequest().setAttribute("mfg", mfg);

getRequest().setAttribute("nowFrom", nowFrom);

getRequest().setAttribute("nowTo", nowTo);

getRequest().setAttribute("pageCount", pageCount);

getRequest().setAttribute("forward", forward);

getRequest().setAttribute("next", next);

return "managerGetMessageSuccess";

}

public void managerStopMessage() throws Exception

{

messageManagement.changeMessagesStatus(ids,true);

}

public void managerStartMessage() throws Exception

{

messageManagement.changeMessagesStatus(ids,false);

}

}

package edu.nuc.web.action;

import java.io.File;

import java.io.FileInputStream;

import java.io.OutputStream;

import java.io.PrintWriter;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.hibernate.Hibernate;

import com.google.gson.Gson;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.common.Constant;

import edu.nuc.service.IUserService;

import edu.nuc.web.manage.UserManagement;

import edu.nuc.web.model.MessageForPage;

import edu.nuc.web.model.UserForJson;

import edu.nuc.web.model.UserInfoForPage;

public class UserAction extends BaseAction {

private UserInfo user;

private IUserService userService;

private File upload;

private String uploadContentType;

private String uploadFileName;

private UserManagement userManagement;

private Pager pager;

private String ids;

public String getIds() {

return ids;

}

public void setIds(String ids) {

this.ids = ids;

}

public Pager getPager() {

return pager;

}

public void setPager(Pager pager) {

this.pager = pager;

}

public UserManagement getUserManagement() {

return userManagement;

}

public void setUserManagement(UserManagement userManagement) {

this.userManagement = userManagement;

}

public IUserService getUserService() {

return userService;

}

public void setUserService(IUserService userService) {

this.userService = userService;

}

public UserInfo getUser() {

return user;

}

public void setUser(UserInfo user) {

this.user = user;

}

public void regist() throws Exception {

userService.regist(user);

userManagement.follow(user, user);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

out.print(methodName + "()");

out.close();

}

public void updateInfo() throws Exception{

UserInfo me = (UserInfo) getSession().getAttribute("user");

user.setId(me.getId());

userService.updateInfo(user);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/plain");

PrintWriter out = response.getWriter();

Gson gson = new Gson();

String json = gson.toJson(user);

out.print(methodName + "("+json+")");

out.close();

}

public void login() throws Exception {

UserInfo onlineUser = userService.login(user);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

String method = "";

if(onlineUser != null)

{

getSession().setAttribute("user", onlineUser);

Gson gson = new Gson();

String json = gson.toJson(user);

method = methodName+"("+json+")";

}else

{

method = methodName+"()";

}

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(method);

out.close();

}

public void getMyInfo() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

Gson gson = new Gson();

String json = gson.toJson(me);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "(" + json + ")");

out.close();

}

public void getUserInfo() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

UserInfo other = (UserInfo) getSession().getAttribute("other");

String json = userManagement.getUserDetail(me, other);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "(" + json + ")");

out.close();

}

public void getPhoto() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

byte[] data = userService.getPhotoById(me);

HttpServletResponse response = getResponse();

response.setContentType("image/\*");

OutputStream toClient = response.getOutputStream(); // �õ���ͻ���������������ݵĶ���

toClient.write(data); // �������

toClient.close();

}

public void getUserPhoto() throws Exception {

byte[] data = userService.getPhotoById(getUser());

HttpServletResponse response = getResponse();

response.setContentType("image/\*");

OutputStream toClient = response.getOutputStream(); // �õ���ͻ���������������ݵĶ���

toClient.write(data); // �������

toClient.close();

}

public void uploadPhoto() throws Exception {

UserInfo user = (UserInfo) getSession().getAttribute("user");

byte[] buffer = new byte[1024 \* 1024];

FileInputStream fis = null;

try {

File photo = getUpload();

fis = new FileInputStream(photo);

fis.read(buffer);

} catch (Exception e) {

e.printStackTrace();

} finally {

if (fis != null) {

fis.close();

}

}

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

Date date = new Date();

user.setPhoto(Hibernate.createBlob(buffer));

userService.uploadPhoto(user);

}

public void getIFollow() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

String json = userManagement.getIFollow(me);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "("+json+")");

out.close();

}

public void getMyFollower() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

String json = userManagement.getFollower(me);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "("+json+")");

out.close();

}

public void toOtherDetail() throws Exception {

if(getSession().getAttribute("other")!=null)

{

getSession().removeAttribute("other");

}

getSession().setAttribute("other", getUser());

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "()");

out.close();

}

public void getUserByName() throws Exception {

String json = userManagement.getUserByName(getUser());

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "("+json+")");

out.close();

}

public void follow() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

UserInfo other = (UserInfo)getSession().getAttribute("other");

userManagement.follow(me, other);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "()");

out.close();

}

public void unFollow() throws Exception {

UserInfo me = (UserInfo) getSession().getAttribute("user");

UserInfo other = (UserInfo)getSession().getAttribute("other");

userManagement.unFollow(me, other);

HttpServletResponse response = getResponse();

HttpServletRequest request = getRequest();

String methodName = request.getParameter("methodName");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.print(methodName + "()");

out.close();

}

public String managerLogin() throws Exception{

UserInfo onlineUser = userService.login(user);

if(onlineUser != null && onlineUser.getRole().getId() == Constant.ADMIN)

{

getSession().setAttribute("manager", onlineUser);

return "managerLoginSuccess";

}else

{

return "managerLoginFaild";

}

}

public void managerStopUser() throws Exception

{

userManagement.changeUserStatus(ids,true);

}

public void managerStartUser() throws Exception

{

userManagement.changeUserStatus(ids,false);

}

public String managerLogout() throws Exception

{

UserInfo manager = (UserInfo)getSession().getAttribute("manager");

if (manager == null) {

return "managerLogin";

} else

{

getSession().setAttribute("manager",null);

return "managerLogin";

}

}

public String managerGetUser() throws Exception

{

UserInfo manager = (UserInfo)getSession().getAttribute("manager");

if (manager == null) {

return "managerLoginFaild";

}

String pageNuum = getRequest().getParameter("pager.pageNum");

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

int pageNum = Integer.parseInt(pageNuum);

if (pageNum <= 1) {

pager = new Pager(1,12);

}else

{

pager = new Pager(pageNum, 12);

}

}else

{

pager = new Pager(1,12);

}

UserInfoForPage ufp = userManagement.getUser(user,pager);

int pageCount = (ufp.getCount() / 12) + 1;

int nowFrom = (pager.getPageNum()-1) \* pager.getPageCount() + 1;

int nowTo = pager.getPageNum() >= pageCount ? ufp.getCount() : (nowFrom + 11);

int forward = (pager.getPageNum()-1)<=1?1:(pager.getPageNum()-1);

int next = (pager.getPageNum()+1)>=pageCount?pageCount:(pager.getPageNum()+1);

getRequest().setAttribute("ufp", ufp);

getRequest().setAttribute("nowFrom", nowFrom);

getRequest().setAttribute("nowTo", nowTo);

getRequest().setAttribute("pageCount", pageCount);

getRequest().setAttribute("forward", forward);

getRequest().setAttribute("next", next);

return "managerGetUserSuccess";

}

public File getUpload() {

return upload;

}

public void setUpload(File upload) {

this.upload = upload;

}

public String getUploadContentType() {

return uploadContentType;

}

public void setUploadContentType(String uploadContentType) {

this.uploadContentType = uploadContentType;

}

public String getUploadFileName() {

return uploadFileName;

}

public void setUploadFileName(String uploadFileName) {

this.uploadFileName = uploadFileName;

}

}

package edu.nuc.web.manage;

import java.util.ArrayList;

import java.util.List;

import com.google.gson.Gson;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.service.ICommentService;

import edu.nuc.web.model.CommentForJson;

public class CommentManagement {

private Pager pager = new Pager(0, 20);

private ICommentService commentService;

public ICommentService getCommentService() {

return commentService;

}

public void setCommentService(ICommentService commentService) {

this.commentService = commentService;

}

public String viewComments(Message message) throws Exception

{

List<CommentForJson> commentList = new ArrayList<CommentForJson>();

List<Comment> comments = commentService.viewCommentsByMessageId(message, pager);

for (Comment comment : comments) {

commentList.add(convertComment(comment));

}

Gson gson = new Gson();

String json = gson.toJson(commentList);

return json;

}

public void addComment(UserInfo user, Message message, Comment comment) throws Exception

{

comment.setMessage(message);

comment.setUser(user);

commentService.addComment(comment);

}

private CommentForJson convertComment(Comment comment)

{

CommentForJson commentForJson = new CommentForJson();

commentForJson.setSenderName(comment.getUser().getName());

commentForJson.setContent(comment.getContent());

return commentForJson;

}

}

package edu.nuc.web.manage;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStream;

import java.sql.SQLException;

import java.sql.Timestamp;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.hibernate.Hibernate;

import com.google.gson.Gson;

import edu.nuc.bean.Comment;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.ICommentService;

import edu.nuc.service.IMessageService;

import edu.nuc.service.MessageCacheManager;

import edu.nuc.service.model.MessageItem;

import edu.nuc.web.model.CommentForJson;

import edu.nuc.web.model.MessageForJson;

import edu.nuc.web.model.MessageForPage;

public class MessageManagement {

private Pager pager = new Pager(0, edu.nuc.common.Constant.HOMEPAGE\_MESSAGENUM);

private IMessageService messageService;

private ICommentService commentService;

private MessageCacheManager messageCacheManagement;

private static Map<Integer, Message> mapCacheData = new HashMap<Integer, Message>();

public ICommentService getCommentService() {

return commentService;

}

public MessageCacheManager getMessageCacheManagement() {

return messageCacheManagement;

}

public void setMessageCacheManagement(

MessageCacheManager messageCacheManagement) {

this.messageCacheManagement = messageCacheManagement;

}

public void setCommentService(ICommentService commentService) {

this.commentService = commentService;

}

public IMessageService getMessageService() {

return messageService;

}

public void setMessageService(IMessageService messageService) {

this.messageService = messageService;

}

public MessageManagement() {

}

public String getHomePageData(UserInfo user) throws Exception {

List<MessageItem> messages = (List<MessageItem>) messageService

.viewFollowingMessage(user, pager);

List<MessageForJson> messageList = new ArrayList<MessageForJson>();

for (MessageItem messageItem : messages) {

Message message = messageItem.getMessage();

List<Comment> fiveComments = commentService

.getFiveComments(message);

messageItem.setFiveComments(fiveComments);

messageList.add(converMessage(messageItem));

}

Gson gson = new Gson();

String json = gson.toJson(messageList);

return json;

}

public String getMoreHomePageData(UserInfo user,Message m) throws Exception {

List<MessageItem> messages = (List<MessageItem>) messageService

.viewMoreFollowingMessage(user, pager,m);

List<MessageForJson> messageList = new ArrayList<MessageForJson>();

for (MessageItem messageItem : messages) {

Message message = messageItem.getMessage();

List<Comment> fiveComments = commentService

.getFiveComments(message);

messageItem.setFiveComments(fiveComments);

messageList.add(converMessage(messageItem));

}

Gson gson = new Gson();

String json = gson.toJson(messageList);

return json;

}

public byte[] getPicById(Message message) throws Exception {

byte[] data = messageService.getPicByMessageId(message);

return data;

}

public void uploadMessage(UserInfo user, Message message, File image)

throws Exception {

byte[] buffer = new byte[1024 \* 1024];

FileInputStream fis = null;

try {

fis = new FileInputStream(image);

fis.read(buffer);

} catch (Exception e) {

e.printStackTrace();

} finally {

if (fis != null) {

fis.close();

}

}

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

Date date = new Date();

message.setUser(user);

message.setImage(Hibernate.createBlob(buffer));

message.setTime(Timestamp.valueOf(sdf.format(date)));

messageService.uploadMessage(message);

}

public String getUserMessage(UserInfo user) throws Exception {

List<MessageItem> messages = (List<MessageItem>) messageService

.getUserMessage(user, pager);

List<MessageForJson> messageList = new ArrayList<MessageForJson>();

for (MessageItem messageItem : messages) {

Message message = messageItem.getMessage();

List<Comment> fiveComments = commentService

.getFiveComments(message);

messageItem.setFiveComments(fiveComments);

messageList.add(converMessage(messageItem));

}

Gson gson = new Gson();

String json = gson.toJson(messageList);

return json;

}

public String getRecomandMessageIds() throws ServiceException {

List<Message> messages = messageCacheManagement.getRecomandMessages();

List<Integer> ids = new ArrayList<Integer>();

for (Message message : messages) {

ids.add(message.getId());

mapCacheData.put(message.getId(), message);

}

Gson gson = new Gson();

String json = gson.toJson(ids);

return json;

}

public byte[] getRecomandPicById(int id) {

if (MessageManagement.mapCacheData == null) {

try {

getRecomandMessageIds();

} catch (ServiceException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

InputStream is = null;

Message message = MessageManagement.mapCacheData.get(id);

try {

is = message.getImage().getBinaryStream();

} catch (SQLException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

byte[] buffer = new byte[1024 \* 1024];

try {

is.read(buffer);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

int c = buffer.length;

return buffer;

}

private MessageForJson converMessage(MessageItem messageItem) {

MessageForJson message = new MessageForJson();

message.setSenderId(messageItem.getMessage().getUser().getId());

message.setMessageId(messageItem.getMessage().getId());

message.setUserName(messageItem.getMessage().getUser().getName());

message.setViewerLike(messageItem.isViewerLike());

message.setLikeNum(messageItem.getLikeNum());

message.setText(messageItem.getMessage().getText());

List<Comment> comments = messageItem.getFiveComments();

List<CommentForJson> commentList = new ArrayList<CommentForJson>();

for (Comment comment : comments) {

commentList.add(convertComment(comment));

}

message.setFiveComments(commentList);

message.setImageOrVideo(messageItem.getMessage().isImgOrVideo());

return message;

}

private CommentForJson convertComment(Comment comment) {

CommentForJson commentForJson = new CommentForJson();

commentForJson.setSenderName(comment.getUser().getName());

commentForJson.setContent(comment.getContent());

return commentForJson;

}

public byte[] getRecomandPicReadom() throws ServiceException {

return messageCacheManagement.getRecomandPicReadom();

}

public String getMessageById(Message message,UserInfo user) throws ServiceException

{

MessageItem m = messageService.getMessageById(message,user);

Gson gson = new Gson();

MessageForJson mfj = converMessage(m);

String json = gson.toJson(mfj);

return json;

}

public void removeMessage(Message message) throws ServiceException

{

messageService.removeMessage(message);

}

public MessageForPage getMessages(Pager pager, UserInfo user) throws ServiceException

{

List<Message> messages = messageService.getMessages(pager, user);

int count = messageService.getMessagesCount(user);

MessageForPage mfp = new MessageForPage();

mfp.setCount(count);

mfp.setMessages(messages);

return mfp;

}

public void changeMessagesStatus(String ids,boolean status) throws ServiceException

{

if (!ids.equals("")) {

ids.substring(0, ids.length()-1);

}

String[] idArray = ids.split(",");

List<Message> messages = new ArrayList<Message>();

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

Message message = new Message(Integer.parseInt(idArray[i]));

messages.add(message);

}

messageService.changeMessagesStatus(messages,status);

}

}

package edu.nuc.web.manage;

import java.util.ArrayList;

import java.util.List;

import com.google.gson.Gson;

import edu.nuc.bean.Follow;

import edu.nuc.bean.Message;

import edu.nuc.bean.UserInfo;

import edu.nuc.bean.help.Pager;

import edu.nuc.platform.exception.ServiceException;

import edu.nuc.service.IFollowService;

import edu.nuc.service.IUserService;

import edu.nuc.web.model.OtherForJson;

import edu.nuc.web.model.UserForJson;

import edu.nuc.web.model.UserInfoForPage;

public class UserManagement {

private IUserService userService;

private IFollowService followService;

public IUserService getUserService() {

return userService;

}

public void setUserService(IUserService userService) {

this.userService = userService;

}

public String getFollower(UserInfo user) throws Exception

{

List<UserInfo> userInfos = userService.getFollower(user);

List<UserForJson> userForJsons = new ArrayList<UserForJson>();

for (UserInfo userInfo : userInfos) {

userForJsons.add(convertUserInfo(userInfo));

}

Gson gson = new Gson();

String json = gson.toJson(userForJsons);

return json;

}

public String getIFollow(UserInfo user) throws Exception

{

List<UserInfo> userInfos = userService.getIFollow(user);

List<UserForJson> userForJsons = new ArrayList<UserForJson>();

for (UserInfo userInfo : userInfos) {

userForJsons.add(convertUserInfo(userInfo));

}

Gson gson = new Gson();

String json = gson.toJson(userForJsons);

return json;

}

public String getUserByName(UserInfo user) throws Exception

{

List<UserInfo> userInfos = userService.getUserByName(user,null);

List<UserForJson> userForJsons = new ArrayList<UserForJson>();

for (UserInfo userInfo : userInfos) {

userForJsons.add(convertUserInfo(userInfo));

}

Gson gson = new Gson();

String json = gson.toJson(userForJsons);

return json;

}

private UserForJson convertUserInfo(UserInfo user)

{

UserForJson ufj = new UserForJson();

ufj.setId(user.getId());

ufj.setUsername(user.getUsername());

ufj.setBreif(user.getBreif());

return ufj;

}

private OtherForJson convertOtherInfo(UserInfo user)

{

OtherForJson ofj = new OtherForJson();

ofj.setBreif(user.getBreif());

ofj.setGender(user.isGender());

ofj.setId(user.getId());

ofj.setUsername(user.getUsername());

return ofj;

}

public IFollowService getFollowService() {

return followService;

}

public void setFollowService(IFollowService followService) {

this.followService = followService;

}

public String getUserDetail(UserInfo me, UserInfo other)

{

UserInfo otherUser = null;

try {

otherUser = userService.getUserById(other);

} catch (ServiceException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

Follow follow = new Follow();

follow.setFollower(me);

follow.setUser(other);

boolean isFollow = false;

try {

isFollow = followService.isFollow(follow);

} catch (ServiceException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

OtherForJson otherUserInfo = convertOtherInfo(otherUser);

otherUserInfo.setFollow(isFollow);

Gson gson = new Gson();

String json = gson.toJson(otherUserInfo);

return json;

}

public void follow(UserInfo me, UserInfo other) throws ServiceException

{

Follow follow = new Follow();

follow.setFollower(me);

follow.setUser(other);

try {

followService.addFollow(follow);

} catch (ServiceException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public void unFollow(UserInfo me, UserInfo other) throws ServiceException

{

Follow follow = new Follow();

follow.setFollower(me);

follow.setUser(other);

try {

followService.removeFollow(follow);

} catch (ServiceException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public UserInfoForPage getUser(UserInfo user, Pager pager) throws ServiceException

{

List<UserInfo> users = userService.getUserByName(user, pager);

int count = userService.getUserCount(user);

UserInfoForPage ufp = new UserInfoForPage();

ufp.setCount(count);

ufp.setUserInfos(users);

return ufp;

}

public List<UserInfo> getUserByName(UserInfo user, Pager pager) throws ServiceException

{

return userService.getUserByName(user, pager);

}

public void changeUserStatus(String ids,boolean status) throws ServiceException

{

if (!ids.equals("")) {

ids.substring(0, ids.length()-1);

}

String[] idArray = ids.split(",");

List<UserInfo> users = new ArrayList<UserInfo>();

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

UserInfo user = new UserInfo(Integer.parseInt(idArray[i]));

users.add(user);

}

userService.changeUsersStatus(users,status);

}

}

package edu.nuc.web.model;

public class CommentForJson {

private String senderName;

private String content;

public String getSenderName() {

return senderName;

}

public void setSenderName(String senderName) {

this.senderName = senderName;

}

public String getContent() {

return content;

}

public void setContent(String content) {

this.content = content;

}

}

package edu.nuc.web.model;

import java.util.List;

public class MessageForJson {

private int messageId;

private int senderId;

private String userName;

private String text;

private List<CommentForJson> fiveComments;

private boolean isViewerLike;

private int likeNum;

private boolean imageOrVideo;//true for image

public boolean isImageOrVideo() {

return imageOrVideo;

}

public void setImageOrVideo(boolean imageOrVideo) {

this.imageOrVideo = imageOrVideo;

}

public int getMessageId() {

return messageId;

}

public void setMessageId(int messageId) {

this.messageId = messageId;

}

public int getSenderId() {

return senderId;

}

public void setSenderId(int senderId) {

this.senderId = senderId;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getText() {

return text;

}

public void setText(String text) {

this.text = text;

}

public List<CommentForJson> getFiveComments() {

return fiveComments;

}

public void setFiveComments(List<CommentForJson> fiveComments) {

this.fiveComments = fiveComments;

}

public boolean isViewerLike() {

return isViewerLike;

}

public void setViewerLike(boolean isViewerLike) {

this.isViewerLike = isViewerLike;

}

public int getLikeNum() {

return likeNum;

}

public void setLikeNum(int likeNum) {

this.likeNum = likeNum;

}

}

package edu.nuc.web.model;

import java.util.List;

import edu.nuc.bean.Message;

public class MessageForPage {

private int count;

private List<Message> messages;

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

public List<Message> getMessages() {

return messages;

}

public void setMessages(List<Message> messages) {

this.messages = messages;

}

}

package edu.nuc.web.model;

public class OtherForJson {

private int id;

private String username;

private String breif = "";

private boolean gender;

private boolean isFollow;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getBreif() {

return breif;

}

public void setBreif(String breif) {

this.breif = breif;

}

public boolean isGender() {

return gender;

}

public void setGender(boolean gender) {

this.gender = gender;

}

public boolean isFollow() {

return isFollow;

}

public void setFollow(boolean isFollow) {

this.isFollow = isFollow;

}

}

package edu.nuc.web.model;

public class UserForJson {

private int id;

private String username;

private String breif;

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getBreif() {

return breif;

}

public void setBreif(String breif) {

this.breif = breif;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

}

package edu.nuc.web.model;

import java.util.List;

import edu.nuc.bean.UserInfo;

public class UserInfoForPage {

List<UserInfo> userInfos;

int count;

public List<UserInfo> getUserInfos() {

return userInfos;

}

public void setUserInfos(List<UserInfo> userInfos) {

this.userInfos = userInfos;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

}

function getCommentsData(){

$.ajax({

type:"get",

//dataType:'json',

url:"http://192.168.0.102:8088/OnlineShare/getComments?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"getCommentsDatas",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function addComment(){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/addComment",

dataType : "jsonp",

data:getCommentQueryString(),

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"backToHomePage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getMyFollower(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getMyFollower?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showMyFollower",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getIFollow(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getIFollow?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showIFollow",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getUserByName(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getUserByName?id="+Math.random(),

dataType : "jsonp",

data:getQueryString(),

jsonp: "methodName",

jsonpCallback:"showUsers",

});

}

function toOtherDetail(id)

{

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/toOtherDetail",

dataType : "jsonp",

data:"user.id="+id,

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"getOtherInfo",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

hideLoading();

}

$(document).ready(function(){

$(".followMeBtn").bind("click",function(){

getMyFollower();

$(".followMeBtn").unbind();

});

$(".meFollowBtn").bind("click",function(){

getIFollow();

$(".meFollowBtn").unbind();

});

});

$(document).ready(function(){

alert();

$.ajax({

type:"post",

url:"getPicById?message.id=12",

});

});

getHomePageData();

function getHomePageData(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/viewHomePage?abc="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",

jsonpCallback:"initHomePage",

});

}

function getMoreHomePageData(id){

showLoading();

$.ajax({

type:"get",

//dataType:'json',

url:"http://192.168.0.102:8088/OnlineShare/viewHomePageMore?message.Id="+id+"&abc="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showMoreHomePage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function clickAddComment(messageId){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/toAddCommentPage?message.Id="+messageId,

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"getCommentsData",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function addLike(messageId,div){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/addLike?message.Id="+messageId,

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"plusLikeNum",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function login(){

showLoading();

var data = getLoginString();

loginGo(data);

}

function loginGo(data)

{

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/login?abc="+Math.random(),

dataType : "jsonp",

data:data,

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"jumpToHomePage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getMyInfo(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getMyInfo?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showMyInfo",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function updateInfo()

{

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/updateInfo",

dataType : "jsonp",

data:getUpdateString(),

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"backToMePage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function removeMessage(id)

{

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/removeMessage",

dataType : "jsonp",

data:"message.id="+id,

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"removeMessagePic",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getMyMessage(){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getMyMessage?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showMyMessage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getUserInfoForUpdate()

{

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getMyInfo?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showInfoForUpdate",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

$(document).ready(function(){

$(".meBtn").bind("click",function(){

getMyInfo();

$(".meBtn").unbind();

});

});

function getOtherInfo(){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getUserInfo?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showOtherInfo",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getOtherMessage(){

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/getUserMessage?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showOtherMessage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function unFollow()

{

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/unFollow",

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showUnFollow",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function follow()

{

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/follow",

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showFollowing",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getRecomandIds(){

showLoading();

$.ajax({

type:"get",

//dataType:'json',

url:"http://192.168.0.102:8088/OnlineShare/getRecomandMessage?id="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showRecomadMessage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

function getOneMessageById(id){

showLoading();

$.ajax({

type:"get",

//dataType:'json',

url:"http://192.168.0.102:8088/OnlineShare/getMessageById?message.id="+id+"&abc="+Math.random(),

dataType : "jsonp",

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"showOneMessagePage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

$(document).ready(function(){

$(".recomandBtn").bind("click",function(){

getRecomandIds();

$(".recomandBtn").unbind();

});

});

function regist(){

showLoading();

$.ajax({

type:"get",

url:"http://192.168.0.102:8088/OnlineShare/regist",

dataType : "jsonp",

data:getRegistString(),

jsonp: "methodName",//��������ڽ���callback���õ�function���Ĳ���

jsonpCallback:"jumpToLoginPage",//callback��function����

//success:function(json){},

//success:function(json){getData(json);},

//error:function(){alert("error");}

});

}

//1.tabId����id 2.oddColor��������ʽ 3.evenColorż������ʽ 4.hoverClass���������ʽ

function senfe(tabId, oddColor, evenColor, hoverClass) {

var table = $("#" + tabId);

$("#" + tabId + " tbody tr:odd td").addClass(oddColor);

$("#" + tabId + " tbody tr:even td").addClass(evenColor);

$("#" + tabId + " tbody tr").hover(

function () {

$(this).children("td").addClass(hoverClass);

},

function () {

$(this).children("td").removeClass(hoverClass);

}

);

}

//ȫѡ

function GetAllCheckBox(CheckAll) {

var items = document.getElementsByTagName("input");

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

if (items[i].type == "checkbox") {

items[i].checked = CheckAll.checked;

}

}

}

<%@ page language="java" import="java.util.\*" pageEncoding="utf-8"%>

<%@ taglib prefix="s" uri="/struts-tags" %>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href="<%=basePath%>">

<title>My JSP 'exception.jsp' starting page</title>

<meta http-equiv="pragma" content="no-cache">

<meta http-equiv="cache-control" content="no-cache">

<meta http-equiv="expires" content="0">

<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">

<meta http-equiv="description" content="This is my page">

<!--

<link rel="stylesheet" type="text/css" href="styles.css">

-->

</head>

<body>

<s:property value="exception.Message" /><br><br>

请联系管理员或<a href="" onclick="window.history.back();return false;">重试</a>

</body>

</html>

<%@ page language="java" import="java.util.\*" pageEncoding="utf-8"%>

<%@ taglib prefix="s" uri="/struts-tags" %>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE HTML>

<html style="overflow:hidden;">

<head>

<meta http-equiv="Content-Type" content="text/html" charset="utf-8" />

<title>OnlineShare</title>

<link href="css/CommonStyle.css" rel="stylesheet" type="text/css" />

<link href="css/PubilcStyle.css" rel="stylesheet" type="text/css" />

<link href="jsServer/Prompt/skin/theme/ymPrompt.css" rel="stylesheet" />

<script src="jsServer/Prompt/ymPrompt.js"></script>

<script src="jsServer/jquery-1.7.js"></script>

<script>

function showPage(src)

{

$("iframe").attr("src",src);

}

</script>

</head>

<body style="overflow:hidden;">

<center>

<!--allwidth -->

<div class="allwidth">

<div class="top">

<div class="top\_info"><span class="bigfont">OnlineShare </span> 后台管理&nbsp;v1.0</div>

<div class="user\_exit">

<span><a href="#" class="user\_name">管理员：${sessionScope.manager.username }</a></span><span><a href="managerLogout" class="btn\_exit">退 出</a></span>

</div>

</div>

<div class="top\_menu">

<a href="#" onclick="showPage('managerGetMessage');return;"><img src="images/icon/ico\_x16\_menu5.gif" />消息管理</a>|

<a href="#" onclick="showPage('managerGetUser');return;"><img src="images/icon/ico\_x16\_menu2.gif" />用户管理</a>|

<a href="#" ><img src="images/icon/ico\_x16\_menu1.gif" />系统设置</a>|

</div>

<div class="b\_body">

<iframe src="managerGetMessage" id="main" name="main" frameBorder=0 scrolling=auto width="100%" height="592px" ></iframe>

<div class="clear"></div>

</div>

<div class="clear"></div>

<div class="foot">&copy;版权归Steven所有</div>

</div>

<!--/allwidth -->

</center>

</body>

</html>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<link href="css/CommonStyle.css" rel="stylesheet" type="text/css" />

<link href="css/PubilcStyle.css" rel="stylesheet" type="text/css" />

<script src="jsServer/jquery-1.7.js"></script>

<script>

$(function () {

$(".b\_body").height($(window).height() - $(".top").height() - $(".foot").height() - 5);

})

$(window).resize(function () {

$(".b\_body").height($(window).height() - $(".top").height() - $(".foot").height() - 5);

});

</script>

</head>

<body>

<center>

<!--allwidth -->

<div class="allwidth">

<div class="b\_body" style="min-height:490px;">

<div class="login\_bigbox\_par">

<div class="login\_bigbox">

<div class="login\_box">

<div class="login\_title">OnlineShare登陆</div>

<form action="/OnlineShare/managerLogin" method="post">

<p><input type="text" name="user.username" class="txt\_box txt\_userName" /></p>

<p><input type="password" name="user.password" class="txt\_box txt\_pwd" /></p>

<p><input type="submit" value="登陆" class="btn\_blue"/></p>

</form>

</div>

</div>

</div>

</div>

</div>

<!--/allwidth -->

</center>

</body>

</html>

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<%@taglib prefix="cc" uri="http://java.sun.com/jsp/jstl/core"%>

<%@taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<meta http-equiv="Content-Type" content="text/html" charset="utf-8" />

<title>ICYRIVER网站后台</title>

<link href="css/CommonStyle.css" rel="stylesheet" type="text/css" />

<link href="css/PubilcStyle.css" rel="stylesheet" type="text/css" />

<script src="jsServer/jquery-1.7.js"></script>

<script src="jsServer/changerows.js"></script>

<script src="jsServer/My97DatePicker/WdatePicker.js"></script>

<script>

//搜索框隐藏和显示

function showSearch() {

if ($(".search\_box").css("display")=="block") {

$(".search\_box").slideUp();

} else {

$(".search\_box").slideDown();

}

}

//隐藏搜索框

function hideSearch() {

$(".search\_box").slideUp();

}

function search()

{

var name = $(".txt\_box").val();

window.location.href="/OnlineShare/managerGetMessage?user.username="+name;

}

function selectAll(cb)

{

$(":checkbox").attr("checked",cb.checked);

}

function stop()

{

var ids = "";

$(":checkbox").each(function(){

if($(this).attr("messageId")!=null&&$(this).attr("checked")=="checked")

{

ids+=$(this).attr("messageId");

ids+=',';

}

})

$.ajax({

type: "post",

url : "/OnlineShare/managerStopMessage",

data: 'ids='+ids,

success: function(){

$(":checkbox").each(function(){

if($(this).attr("messageId")!=null&&$(this).attr("checked")=="checked")

{

$(this).parents("tr").find("span").attr("class","circular\_red");

}

})

}

});

}

function start()

{

var ids = "";

$(":checkbox").each(function(){

if($(this).attr("messageId")!=null&&$(this).attr("checked")=="checked")

{

ids+=$(this).attr("messageId");

ids+=',';

}

})

$.ajax({

type: "post",

url : "/OnlineShare/managerStartMessage",

data: 'ids='+ids,

success: function(){

$(":checkbox").each(function(){

if($(this).attr("messageId")!=null&&$(this).attr("checked")=="checked")

{

$(this).parents("tr").find("span").attr("class","circular\_green");

}

})

}

});

}

</script>

</head>

<body>

<center>

<!--allwidth -->

<div class="allwidth">

<!--列表-->

<div class="list\_div">

<!--操作-->

<div class="operation\_div">

<div class="f\_left">

按用户查找

<input type="text" class="txt\_box" id="txt\_box"/> <input type="button" class="btn\_search" onclick="search()" value="搜 索" />

</div>

<div class="f\_right"><a href="#" onclick="start();return;" class="btn\_ok">启用所选</a><a href="#" class="btn\_del" onclick="stop();return;">删除所选</a></div>

<div class="clear"></div>

</div>

<!--/操作-->

<table border="0" cellspacing="0" class="mrTable" id="list">

<tr>

<th style="width: 5%;"><input type="checkbox" onclick="selectAll(this)" /></th>

<th style="width: 14%;">图片</th>

<th style="width:8%">发布人</th>

<th align="center" style="width: auto;">描述</th>

<th style="width: 12%;">发布时间</th>

<th style="width: 8%;">状态</th>

</tr>

<cc:forEach items="${requestScope.mfg.messages}" var="message" varStatus="vs">

<tr><td><input type="checkbox" class="cbvalue" messageId="${message.id }"/></td>

<td>

<img src="http://localhost:8088/OnlineShare/getPicById?message.id=${message.id }" width="140px" height="140px" />

</td>

<td>

${message.user.username }

</td>

<td class="align\_l">

${message.text }

</td>

<td class="color\_gray">

${message.time }

</td>

<td>

<cc:if test="${message.del=='false'}">

<span class="circular\_green" id="status" messageId="${message.id }"></span>

</cc:if>

<cc:if test="${message.del=='true'}">

<span class="circular\_red" id="status" messageId="${message.id }"></span>

</cc:if>

</td>

</tr>

</cc:forEach>

</table>

<!--分页-->

<div class="pageList">

<div class="f\_left">共${mfg.count}条&nbsp;&nbsp;当前${nowFrom}-${nowTo }</div>

<div class="f\_right">

<a href="/OnlineShare/managerGetMessage?user.username=${param['user.username']}&pager.pageNum=0" title="首页"><<</a><a href="/OnlineShare/managerGetMessage?user.username=${param['user.username']}&pager.pageNum=${forward}" title="上一页"><</a><a href="/OnlineShare/managerGetMessage?user.username=${param['user.username']}&pager.pageNum=${next}" title="下一页">></a><a href="/OnlineShare/managerGetMessage?user.username=${param['user.username']}&pager.pageNum=${pageCount}" title="尾页">>></a>

<input type="text" class="pageNumber" />页/共${pageCount}页 <input type="button" class="go\_page" title="跳转" />

</div>

<div class="clear"></div>

</div>

<!--/分页-->

<script language="javascript">

//奇偶行变色senfe("表格名称","奇数行背景f7feff","偶数行背景","鼠标经过背景"--,"点击后背景");

senfe("list", "oddColor", "evenColor", "hoverClass");

</script>

</div>

<!--/列表-->

</div>

<!--/allwidth -->

</center>

</body>

</html>

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<%@taglib prefix="cc" uri="http://java.sun.com/jsp/jstl/core"%>

<%@taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

<meta http-equiv="Content-Type" content="text/html" charset="utf-8" />

<title>ICYRIVER网站后台</title>

<link href="css/CommonStyle.css" rel="stylesheet" type="text/css" />

<link href="css/PubilcStyle.css" rel="stylesheet" type="text/css" />

<script src="jsServer/jquery-1.7.js"></script>

<script src="jsServer/changerows.js"></script>

<script src="jsServer/My97DatePicker/WdatePicker.js"></script>

<script>

//搜索框隐藏和显示

function showSearch() {

if ($(".search\_box").css("display")=="block") {

$(".search\_box").slideUp();

} else {

$(".search\_box").slideDown();

}

}

//隐藏搜索框

function hideSearch() {

$(".search\_box").slideUp();

}

function search()

{

var name = $(".txt\_box").val();

window.location.href="/OnlineShare/managerGetUser?user.username="+name;

}

function selectAll(cb)

{

$(":checkbox").attr("checked",cb.checked);

}

function stop()

{

var ids = "";

$(":checkbox").each(function(){

if($(this).attr("userId")!=null&&$(this).attr("checked")=="checked")

{

ids+=$(this).attr("userId");

ids+=',';

}

})

$.ajax({

type: "post",

url : "/OnlineShare/managerStopUser",

data: 'ids='+ids,

success: function(){

$(":checkbox").each(function(){

if($(this).attr("userId")!=null&&$(this).attr("checked")=="checked")

{

$(this).parents("tr").find("span").attr("class","circular\_red");

}

})

}

});

}

function start()

{

var ids = "";

$(":checkbox").each(function(){

if($(this).attr("userId")!=null&&$(this).attr("checked")=="checked")

{

ids+=$(this).attr("userId");

ids+=',';

}

})

$.ajax({

type: "post",

url : "/OnlineShare/managerStartUser",

data: 'ids='+ids,

success: function(){

$(":checkbox").each(function(){

if($(this).attr("userId")!=null&&$(this).attr("checked")=="checked")

{

$(this).parents("tr").find("span").attr("class","circular\_green");

}

})

}

});

}

</script>

</head>

<body>

<center>

<!--allwidth -->

<div class="allwidth">

<!--列表-->

<div class="list\_div">

<!--操作-->

<div class="operation\_div">

<div class="f\_left">

查找用户

<input type="text" class="txt\_box" id="txt\_box"/> <input type="button" class="btn\_search" onclick="search()" value="搜 索" />

</div>

<div class="f\_right"><a href="#" onclick="start();return;" class="btn\_ok">启用所选</a><a href="#" class="btn\_del" onclick="stop();return;">删除所选</a></div>

<div class="clear"></div>

</div>

<!--/操作-->

<table border="0" cellspacing="0" class="mrTable" id="list">

<tr>

<th style="width: 5%;"><input type="checkbox" onclick="GetAllCheckBox(this)" /></th>

<th style="width: 8%;">头像</th>

<th style="width:8%">用户名</th>

<th align="center" style="width: auto;">真实姓名</th>

<th style="width: 20%;">电子邮箱</th>

<th style="width: 8%;">状态</th>

</tr>

<cc:forEach items="${requestScope.ufp.userInfos}" var="userInfo" varStatus="vs">

<tr><td><input type="checkbox" class="cbvalue" userId="${userInfo.id }"/></td>

<td>

<img src="http://localhost:8088/OnlineShare/getUserPhoto?user.id=${userInfo.id }" width="80px" height="80px" />

</td>

<td>

${userInfo.username }

</td>

<td class="align\_l">

${userInfo.name }

</td>

<td class="color\_gray">

${userInfo.email }

</td>

<td>

<cc:if test="${userInfo.del=='false'}">

<span class="circular\_green" id="status" userId="${userInfo.id }"></span>

</cc:if>

<cc:if test="${userInfo.del=='true'}">

<span class="circular\_red" id="status" userId="${userInfo.id }"></span>

</cc:if>

</td>

</tr>

</cc:forEach>

</table>

<!--分页-->

<div class="pageList">

<div class="f\_left">共${ufp.count}条&nbsp;&nbsp;当前${nowFrom}-${nowTo }</div>

<div class="f\_right">

<a href="/OnlineShare/managerGetUser?user.username=${param['user.username']}&pager.pageNum=0" title="首页"><<</a><a href="/OnlineShare/managerGetUser?user.username=${param['user.username']}&pager.pageNum=${forward}" title="上一页"><</a><a href="/OnlineShare/managerGetUser?user.username=${param['user.username']}&pager.pageNum=${next}" title="下一页">></a><a href="/OnlineShare/managerGetUser?user.username=${param['user.username']}&pager.pageNum=${pageCount}" title="尾页">>></a>

<input type="text" class="pageNumber" />页/共${pageCount}页 <input type="button" class="go\_page" title="跳转" />

</div>

<div class="clear"></div>

</div>

<!--/分页-->

<script language="javascript">

//奇偶行变色senfe("表格名称","奇数行背景f7feff","偶数行背景","鼠标经过背景"--,"点击后背景");

senfe("list", "oddColor", "evenColor", "hoverClass");

</script>

</div>

<!--/列表-->

</div>

<!--/allwidth -->

</center>

</body>

</html>

package edu.nuc.onlinesharemobile;

import org.apache.cordova.DroidGap;

import android.os.Bundle;

public class MainActivity extends DroidGap

{

@Override

public void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

//super.setIntegerProperty( "splashscreen", R.drawable.splash );

super.loadUrl("file:///android\_asset/www/login.html", 1);

}

}