



**JISHOU　UNIVERSITY**

**软件学院课程设计**

|  |  |
| --- | --- |
| **题 目：** | 斗地主 |
|  |  |
| **作 者：** | 周广宇 |
| **学 号：** | 2018401227 |
| **课程名称：** | Java程序设计基础 |
| **专业年级：** | 2018级 |
| **指导老师：** | 聂明 |
| **完成时间：** | 2020年7月5日 |

**吉首大学软件学院制**

设计文档格式要求如下：

1.【页面设置】中的“页边距”设置，上下左右为“2 厘米”。

【页面设置】中的“版式”设置，页眉和页脚为“1 厘米”。

2.全文字体设置，中文为“宋体”，英文为“Times New Roman”。

3.全文【行距】设置，“多倍行距 1.25倍”。

4.一级标题字体设置，“小二号宋体，粗体，居中对齐，间距段前段后各1行”。

5.二级标题字体设置，“小三号宋体，粗体，左对齐，无缩进”。

6.三级标题字体设置，“小四号宋体，粗体，左对齐，无缩进”。

7.正文字体设置，“小四号宋体，两端对齐，段首缩进2字符”。

8.图片、图片编号与名称格式设置，“居中对齐，无缩进”。

9.表格、表格编号与名称格式设置，“居中对齐，无缩进”。

10.代码格式“五号Times New Roman，左对齐”。

上述格式说明在文档撰写完成后自行删除。

**目 录**

[斗地主设计与实现 1](#_Toc44836025)

[1 项目概述 1](#_Toc44836026)

[1.1 项目描述 1](#_Toc44836027)

[1.2 项目需求 1](#_Toc44836028)

[2 系统设计与实现 4](#_Toc44836029)

[2.1 实现系统功能所采覆盖知识点 4](#_Toc44836030)

[2.2 读者信息管理模块 5](#_Toc44836031)

[3 系统测试 7](#_Toc44836032)

[4 项目特色 9](#_Toc44836033)

# 斗地主设计与实现

# 1 项目概述

## 项目描述

斗地主，属于基础类扑克游戏。起源和流行于湖北十堰房县一带，玩法简单，娱乐性强，老少皆宜。据传在万恶的旧社会，地主横行乡里，无恶不做，人们为了发泄对地主的痛恨，常常在一天的劳作之后，一家人关起门来“斗地主”。

斗地主对抗性和配合性都很强。该游戏由三个人玩(四人也能玩)，用一副牌共54张牌。每局牌有一个玩家是“地主”，地主为一方，其余两家农民为另一方，双方对战，地主的目标是(以合法的出牌方式)先出完手里所有的牌，而农民的目标是在地主出完牌以前，先出完手里所有的牌。先出完牌的人所代表的一方获胜。

## 1.2 项目需求

### 1.2.1 功能需求

本项目分为两种玩家，地主和平民，地主比平民多三张牌（地主牌），项目分为AI和玩家

本项目至少完成以下的功能或模块：

1）登录注册界面：玩家进入游戏需要登录自己的账号，如果没有账号，可以进行注册

2）菜单界面：玩家进入菜单页面时，可以开始游戏和查看帮助等

3）发牌：将52张扑克牌随机发放，留三张地主牌

4）抢地主：进行抢地主争夺，抢夺地主可收下三张地主牌

5）计时器：对出牌时间进行控制

6）强制出牌：超出时间，需要强制出一张牌，如果要不起，自动选择不要

7）判断牌型：判断出牌是否规范

8）比较大小：判断能否出牌

9）AI：电脑走牌

10）游戏积分：对游戏积分进行计算

### 1.2.2 用例分析

下面使用总体用例图可以更加清晰的表达整个系统功能，总体用例图如图1-1所示。



图1-1 总体用例图

### 1.2.3 系统流程图

斗地主实现了包括记录，胜率，积分等数据的同步，用户的个人数据存储于数据库，用户可以直接进行登录，不需要重新注册。

在游戏启动时，会有一个登录界面，如果玩家有账号密码，可以直接进行登录，如果没有注册过账号，则可以点击注册按钮进行注册，填写相关信息后，再行登录。登录后将跳转到游戏主界面。

主界面直接分为三大模块。

创建游戏模块提供创建新的游戏，他可以开始游戏，进行重新发牌，玩家在游戏过程中，可以选择重新开始

进行游戏模块提供游戏时的操作功能，游戏进行时，可以叫地主，出牌，过牌等行为，其中，叫地主包括抢地主，和不要地主，玩家游戏时，抢到地主将会获得额外的三张地主牌。

在出牌阶段，能判断牌型，看是否符合出牌的规则。在出牌阶段，会有出牌倒计时，在时间结束时，会自动出牌或者过牌

个人页面模块则提供了当前用户相关信息的展示，包括积分、胜率率等。可以看到自己游戏的痕迹。留言板记录游戏吐槽语，如图1-2所示。



图1-2 系统流程图

### 1.2.3 数据需求

服务端采用MySQL数据库，使用MyBatis作为ORM框架。根据实体设计，共有八张数据表。数据库实体的E-R图如图1-3所示：



图1-3 服务端数据库实体E-R图

表设计如下：

表1-1 Login表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 说明 | 类型 | 长度 | 可否为空 | 主键 |
| 01 | user | 用户名 | varchar | 11 | 否 | 是 |
| 02 | password | 密码 | varchar | 32 | 否 | 否 |

表 1-2 User表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 说明 | 类型 | 长度 | 可否为空 | 主键 |
| 01 | user | 用户名 | varchar | 11 | 否 | 是 |
| 02 | name | 昵称 | varchar | 32 | 是 | 否 |
| 03 | sex | 性别 | varchar | 8 | 是 | 否 |
| 04 | logtime | 注册时间 | data | 20 | 否 | 否 |

表1-3 User-Game表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 说明 | 类型 | 长度 | 可否为空 | 主键 |
| 01 | user | 用户名 | varchar | 11 | 否 | 是 |
| 02 | number | 场次编号 | varchar | 11 | 否 | 否 |
| 03 | livel | 胜率 | double | 11 | 否 | 否 |
| 04 | score | 积分 | int | 32 | 否 | 否 |

表 1-4 Game表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 说明 | 类型 | 长度 | 可否为空 | 主键 |
| 01 | number | 场次编号 | varchar | 11 | 否 | 否 |
| 02 | getscore | 得分 | int | 32 | 否 | 否 |
| 03 | btime | 开始时间 | data | 20 | 否 | 否 |
| 04 | otime | 结束时间 | data | 20 | 否 | 否 |

# 2 系统设计与实现

## 2.1 实现系统功能所采覆盖知识点

本系统实现了综合办公、科研机构、科研人员、科研项目等个11个模块，各模块功能实现所采用技术见表2-1所示。

表2-1 系统功能及所采用技术一览表

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **功能点** | **基本知识点** | **扩展知识点** |
| 1 | 注册 | 1.JavaIO  2.Java类集框架  3.Java数据库编程  4.DAO设计模式  5.图形界面  6.数据内容  7.API文档  8.自动测试 | 1.泛型 |
| 2 | 登录 | 2.多线程 |
| 3 | 场地预约 | 3.单例模式 |
| 4 | 搜索订单 | 4.第三方库 |
| 5 | 提交订单 | 5.代码管理 |
| 6 | 用户管理 |  |
| 7 | 场地管理 |  |
| 8 | 月订单份额 |  |

## 2.2 读者信息管理模块

**2.2.1 界面设计与实现**

登录和注册界面，主界面，留言板。具体的实现界面如图2-1、图2-2所示。



图2-1 注册界面

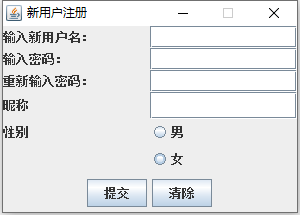


图2-2 登录界面



图2-3 主界面



图2-4 留言板界面

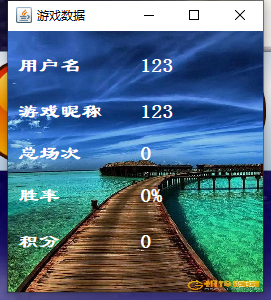


图2-5 游戏数据界面



图2-6 游戏界面

**2.2.2 类设计与实现**

（主要介绍使用了哪些类，并介绍类中重要的方法，并绘制类图，参考如下）

读者信息管理模块类和方法如表2-1所示。

表2-1 读者信息管理模块类和方法列表

|  |  |
| --- | --- |
| 类名称 | 说明 |
| UserAddIFrame | 添加读者信息界面类 |
| UserModiAndDelIFrame | 修改与删除读者信息界面类 |
| Dao | 数据库访问类 |
| Dao中的方法 | |
| public static int Insertoperator (参数省略) | 添加读者信息 |
| public static List selectuser() | 选择读者信息 |
| public static int Deluser(int id) | 删除读者信息 |
| public static int Updateuser(参数省略) | 修改读者信息 |



图2-3 读者信息管理类图

# 3 系统测试

（使用Junit测试核心模块，复制代码，例如如下代码）

雇员增加、修改、删除等功能Junit测试代码如下：

public class IEmpServiceTest {

@Test

public void testInsert() {

Emp vo = new Emp();

vo.setEmpno(8888);

vo.setEname("施工2");

vo.setJob("保洁3");

vo.setHiredate(new Date());

vo.setSal(3.0);

vo.setComm(130.0);

try {

TestCase.assertTrue(ServiceFactory.getIEmpServiceInstance().insert(vo));

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void testUpdate() {

Emp vo = new Emp();

vo.setEmpno(8888);

vo.setEname("施工7");

vo.setJob("保洁7");

vo.setHiredate(new Date());

vo.setSal(1.0);

vo.setComm(110.0);

try {

TestCase.assertTrue(ServiceFactory.getIEmpServiceInstance().update(vo));

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void testDelete() {

Set<Integer> all = new HashSet<Integer>() ;

all.add(8888) ;

try {

TestCase.assertTrue(ServiceFactory.getIEmpServiceInstance().delete(all));

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void testGet() {

try {

TestCase.assertNotNull(ServiceFactory.getIEmpServiceInstance().get(7369));

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void testList() {

try {

TestCase.assertTrue(ServiceFactory.getIEmpServiceInstance().list().size() > 0);

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void testListSplit() {

try {

Map<String, Object> map = ServiceFactory.getIEmpServiceInstance().listSplit("ename", "S", 1, 10);

List<Emp> allEmps = (List<Emp>) map.get("allEmps") ;

Integer empCount = (Integer) map.get("empCount") ;

TestCase.assertTrue(allEmps.size() > 0 && empCount > 0 );

} catch (Exception e) {

e.printStackTrace();

}

}

}

# 4 项目特色

至少写3条以上自己认为有特色地方，可以从功能或者技术方面描述。