登录板块

登录界面

登录界面是当今比较主流的界面，用户名和密码正确便可登录进游戏，在登录框底下加入注册的超链接。通过jsp输入用户名密码，发送表单到servlet，Servlet调用实现方法，在数据库中运用select，查询全部的用户名和密码如果，与表单中的用户名和密码相匹配则成功，然后响应到登录界面。外观上运用星空背景加ACG人物，还有可以动的星星图形的点缀，能够吸引更多玩家尝试登录框为半透明框，鼠标移上去有过渡的效果并且显示蓝色边框。

核心代码：

DBconn.*init*();

ResultSet rs = DBconn.*selectSql*("select \* from user where Nickname='"+Nickname+"' and Password='"+Password+"'");

System.***out***.println(rs);

**if**(rs.next()){

**do** {

**if**(rs.getString("Nickname").equals(Nickname) && rs.getString("Password").equals(Password)){

// flag = true;

user.setID(rs.getInt("UserID"));

user.setName(rs.getString("Nickname"));

user.setPwd(rs.getString("Password"));

user.setDate(rs.getString("RegisterDate"));

user.setEmail(rs.getString("Email"));

}

}**while**(rs.next());

}

**else** {user=**null**;}

DBconn.*closeConn*();

注册界面

排行榜板块

战绩板块

修改信息板块

留言板块的实现

留言界面以及回复界面

流言界面具有简易操作的特点，任何人只要点击留言按钮，输入问题的标题以及具体内容，点击发送便可以成功留言。后端会将内容通过insert语句加进数据库并且每条留言在数据库中会生成唯一的留言序号QID，在外观上连接css文件，将留言栏设成半透明的并在周围配上可移动的集合图形，鼠标移到留言框中还会有过渡效果，以及橙色边框。暗色背景也不会显得刺眼，整体来说赏心悦目。回复界面整体类似于留言界面，在回复是以QID为外码，因为一条回复只对应一个留言，并且自己生成唯一的回复序号RID。

核心代码：

DBconn.*init*();

**if**(!(user.getQuestion().equals("")||user.getText().equals(""))){

**int** i =DBconn.*addUpdDel*("insert into messege(QID,Nickname,Question,Text,Date) "+"values('"+user.getQID()+"','"+user.getName()+"','"+user.getQuestion()+"','"+user.getText()+"','"+user.getDate2()+"')");

**if**(i>0){

flag = **true**;

}

}

DBconn.*closeConn*();

查看留言界面

点击查看留言的按钮，后端会在数据库中执行select语句调用数据库查询全部的留言内容。在jsp中调用Servlet，Servlet在执行实现方法，最后响应jsp将所有数据显示出来，外观上连接css文件，整个列表以蓝色为主色调，按钮设置为果冻状比较个性化。在周围配上可移动的集合图形，鼠标移到列表界面中还会有过渡效果，以及深蓝色边框。背景配了钢笔的背景，衬托出全部留言的主题。在操作列表中可以点击查看我的留言，我的留言界面实现方法类似。在查看界面中加了分页功能，需要另写一个函数得出问题的总数count值，并且在select查询时加上limit限制以实现分页功能。

核心代码：

List<User> list = **new** ArrayList<User>();

**try** {

DBconn.*init*();

ResultSet rs = DBconn.*selectSql*("select \* from question limit "+limit+",8");

**while**(rs.next()){

User user = **new** User();

user.setQID(rs.getInt("QID"));

user.setName(rs.getString("Nickname"));

user.setQuestion(rs.getString("Question"));

user.setText(rs.getString("Text"));

user.setDate2(rs.getString("Date"));

list.add(user);

}

DBconn.*closeConn*();

**return** list;

删除留言功能

玩家可以通过点击删除按钮删除自己的留言而，管理员则有权限删除全部玩家的留言，这里只需要设置grant的值限定删除按钮的出现位置。点击删除时将QID传值到Servlet，之后调用实现方法，在数据库中运用delete语句删除留言。

核心代码：

DBconn.*init*();

String sql = "delete from messege where QID="+QID;

**int** i =DBconn.*addUpdDel*(sql);

**if**(i>0){

flag = **true**;

}

DBconn.*closeConn*();

更改规则模块实现

管理员界面相比于玩家界面显得简陋一些，以实现功能为主。点击规则的按钮即可看到当前规则的具体描述。管理员可以更改任意一项数值，更改完后点击右侧更新按钮即可。Jsp将值传进更改规则的servlet中，然后servlet调用实现方法，实现在数据库中写update，将数据库更新。之后再响应显示规则的servlet，列出新的规则表。

核心代码：

DBconn.*init*();

String sql ="update rulest set Startstep ='"+Startstep

+"' , Endstep ='"+Endstep

+"' , eachS ='"+eachS

+"'where PID = "+PID;

**int** i =DBconn.*addUpdDel*(sql);

五子棋算法

ai算法的核心就是“打分”，下完每一步棋都会对所有没有落子的地方打分数，分数越高威胁就越大。对此，需要设2个数组分别表示对于玩家的棋子来说周围的分数以及对于ai的棋子来说周围的分数。设为AIScore[11]与PlayerScore[11]。此外还需要设置一个分数表，例如一颗子20分、2颗连子40分之类，以此类推，至于为什么设置可见下文。

首先玩家落子落在a[2][2]位置，此时player[2]=player[7]=player[11]=1，此时对于player[2]来说，a[2][0]、a[2][1]、a[2][3]、a[2][4]都是可能让玩家胜利的点，因此PlayerScore[2][0]、PlayerScore[2][1]、PlayerScore[2][3]、PlayerScore[2][4]都可以加20分，同理对于player[7]与player[11]也是一样，PlayerScore[0][0]、PlayerScore[1][1]、PlayerScore[3][3]、PlayerScore[4][4]、PlayerScore[0][2]、PlayerScore[1][2]、PlayerScore[3][2]、PlayerScore[4][2]都变成了20分。如图1-4所示。

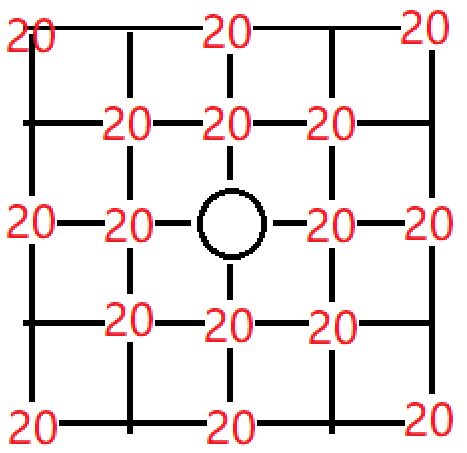


图1-4

为了更好的演示，这里继续让玩家下子，假设下在了a[2][3]的位置，由于2子连珠有40分，此时在player[2]方式下，PlayerScore[2][0]、PlayerScore[2][1]、PlayerScore[2][4]都会在原来的基础上加40分，变为60分。在别的方式下相应位置仍加20分。如图1-5所示。

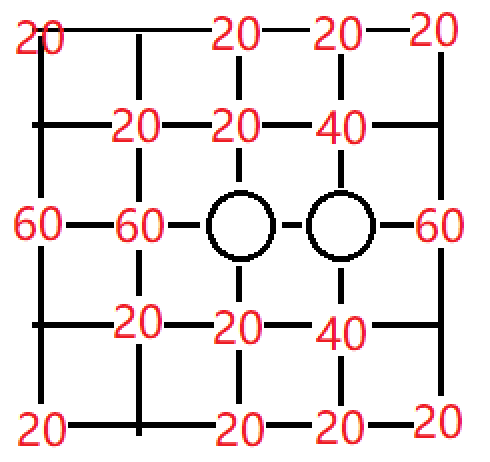
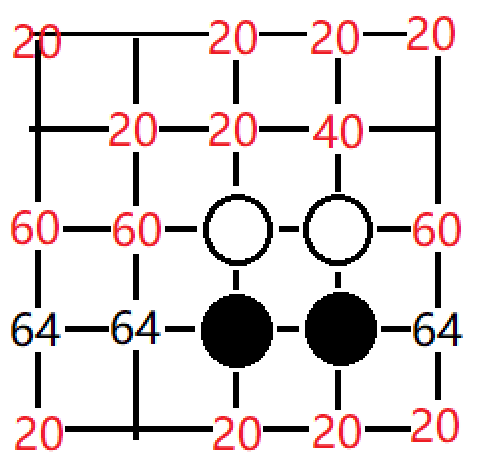


图1-5

假设现在为ai的落子时间，经过计算得到有3个空位的分数为60分为整个棋盘最高的点，因此在三者中选其一即可。以此类推，最终计算机总是会阻止玩家五子连珠的路径。

现在解决了怎么阻止玩家获胜，接下来需要设计ai怎么才能让自己获胜。其实只需要将玩家的分数表加上一定分数即可，例如一个子22分，2个连子42分。下面看图1-6。



这里可以看到现在玩家和ai都下了2个子，假设现在是ai落子，此时可以看到AIScore数组中有3个64分的点，而玩家数组中最高只有60分的点，此时可以将ai看成进攻模式，优先连自己的子。可以在3个64分中的任选一个落子。

因此，如果ai与玩家都有一个4子连珠，ai不会先去处理玩家的4子连珠而是优先完成自己的五子连珠。这样ai让自己获胜的方式就解决了。