山东大学软件学院

面向对象开发技术课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202000300125 | 姓名： 贾星宇 | | 班级：2020级5班 |
| 实验题目：五子棋(WEB版) | | | |
| 实验学时：16 | | 实验日期：到2022年6月14日 | |
| 实验目的：  《面向对象技术》是计算机科学与技术、软件工程专业学生的一门必修课程。面向对象技术是当前最重要的程序设计思想与方法，利用抽象性、封装性、继承性、多态性等特性帮助人们开发出模块化、复用性强、易修改、可扩展的软件系统和程序。本实验通过若干个备选的相对复杂、独立的题目，使学生在实验过程中全面、系统的熟悉面向对象技术的基本概念和设计技术，分析、设计一个相对完整的应用，并使用面向对象编程语言进行实现，通过实验掌握使用面向对象技术进行程序设计、开发的思想和技能。 | | | |
| 硬件环境：  宿主机：lntel(R) Core(TM) i5-10210U CPU @ 1.60G Hz 2 .1 1 G Hz | | | |
| 软件环境：  Windows 10 家庭中文版  WebStrom 2021.3、IntelliJ IDEA 2021.3 | | | |
| 实验步骤与内容：   1. 分析实验要求   【题目】  五子棋(网络版/WEB版)：设计程序，完成下述功能：在大于9条线的方形网格棋盘上，放置黑白两色棋子；两人各执一色，通过网络对弈；可以在任何两条线的交点上放置棋子；同色棋子，先同行、同列或者同队角线上5颗连成一条线者获胜。  【分析】  1.此题目要求为实现web版本的五子棋，因此应该摒弃在同一个页面用同一个棋局进行下棋的情景，应实现两个人各面对一个棋局，利用两个网页甚至两个设备进行对战。因此首先应该具有多用户登录以及创建棋局的功能，以实现多对用户同时下棋的功能  2.在两对用户开始下棋的过程中，应该实现非自己棋局不能下棋、及时查看到对方下的棋子以及对方的状态。为了体现面向对象思想，将每个棋子视作一个对象并存入数据库中，利用数据库的读写操作进行通信。  3.扩展功能：求和以及再来一局。由于每个棋子都是一个对象，所以考虑复用上述功能，当用户申请求和时，相当于下了一个“求和”棋子；当用户申请再来一局时，相当于下了一个“再来一局”棋子。   1. 考虑实现方法   由于要实现web版本的可满足多用户同时对战的功能，最终选择了前后端B/S架构编程。后端利用springboot框架+mybatisplus实现，前端用vue框架+vuetify实现。   1. 后端代码结构：   建立三个类：  Player类：代表不同玩家，有属性playerid单独标识玩家以及playername表示玩家名称  Game类：代表不同棋局，gameid单独标识棋局，player1表示发起棋局的玩家；player2表示客方玩家；isUsing属性表示此棋局是否已经开始，如果开始则不会显示在游戏大厅中。  Piece类：代表棋子信号。Pieceid唯一地指示一个棋子，gameid表示此棋子属于的棋局id；player表示此棋子是由哪一位棋手下的；px和py表示棋子放置在棋盘上的位置。同时，也可以用px与py代表以下信号：  //0,0 :求和  //-1,-1:同意求和  //-2,-2:不同意求和  //-3,-3:请求再来一局  //-4,-4:同意再来一局  //-5,-5:不同意再来一局  //-11,-11:棋局输赢已定  //-10,-10:游戏结束  三个控制器： playerController中有负责建立新玩家或者返回从前玩家的功能  gameController中有创建棋局、其他玩家进入棋局、退出房间以及搜索所有房间、搜索特定房间的功能  pieceController中有布置新棋子、查找所有棋子、更改特定棋子、删除特定棋子、删除所有棋子、判断是否满足五个棋子等功能。  两个mapper接口playerMapper和PieceMapper接口，负责对数据库的增删改查。   1. 前端页面 2. 初始未登录页面  1. 登陆成功后页面  1. 创建棋局后  1. 其他玩家会显示存在的未开始的棋局  1. 开始进行棋子对战  1. 判断输赢  1. 逻辑提示：   正常下棋时，下棋方可以点击任意位置并进行无数次改变，棋子会根据点击位置的变化转到相应位置；同时，当决定下的位置时点击选定，对方便可以看到此棋子，此棋子位置也不可以再次改变。  当未轮到本方下棋时，点击棋盘不会有相应的棋子出现  当一方率先达到五子连珠，系统会进入结束状态，双方无法下棋，可以选择再来一局或者退出游戏，同时满足条件的五子会用亮蓝色标出  在下棋过程中棋手可以点击求和。当点击求和后，另一方会收到求和请求，如果同意求和，棋局将进入结束状态，可以选择是否再来一局或者退出游戏；如果对方不同意求和，则此棋局会继续进行  当棋局在结束状态时，可以点击再来一局。此时对方的页面会弹出相应的请求。如果同意再来一局，会将所有关于此棋局的数据刷新，回到初始状态；如果对方不同意再来一局，此局棋结束，双方自动离场  任意时刻某方点击退出游戏时，此棋局结束，另一方认为赢，同时另一方会离场。随后可以再次创建新的棋局。  6.【小组分工】  小组成员共三人：贾星宇 杨钰润 易格名  经过讨论，最终决定贾星宇（本人）制作五子棋项目，杨钰润与易格名共同制作音视频播放器项目。  因此在实验报告的分工上，贾星宇完成五子棋的实验报告，杨钰润和易格名完成播放器的实验报告，最终打包上传。 | | | |
| 结论分析与体会：  此五子棋页面花费约一周的时间，在本学期初期完成。演示前又花费时间温故了一下此程序，最后顺利演示并写出了此实验报告。通过此次课程设计，我对面向对象编程的方法以及web技术有了更深入的了解，同时也清晰的认识到了自己学习内容的肤浅和不足之处，并决定利用假期时间深入地了解各框架的运行原理，并用最底层的java代码以及sql语言写出属于自己的低耦合高内聚的理想项目。  在设计初始，遇到的困难问题之一便是如何实现两游戏玩家的通信。为了体现面向对象的思想，最终决定利用数据库共享内存的方式进行通信：  在游戏首界面，前端函数会重复读取数据库中的棋局，如果有棋局被创建并存入了数据库，则会被后端代码读取到之后传入前端。随后前端进行判定，如果此局棋缺人，则会显示在首页面上，供其他玩家点击进入对战。如果此棋局两个玩家已经集齐，则不会显示在主页上。  随后进入棋局，棋局为标准棋盘（15×15），因此为了表示每个棋子，给每个棋子属性px和py表示它的横纵坐标。同时，有gameid属性表示棋子是否属于此棋局。在进入棋局后前端会不断发送请求获得当前的所有棋子，如果当前棋子属于该局棋并且为正常的坐标，则会显示在棋盘上。同时根据上一个棋子属于哪一位玩家来判定此刻应该是哪位玩家下棋。  同时，利用≤0的棋子坐标表示各种信号：  //0,0 :求和  //-1,-1:同意求和  //-2,-2:不同意求和  //-3,-3:请求再来一局  //-4,-4:同意再来一局  //-5,-5:不同意再来一局  //-11,-11:棋局输赢已定  //-10,-10:游戏结束  至此，基本功能得以实现。  路漫漫其修远兮，吾将上下而求索。  【前端代码】  //home.vue  <template>  <div class="background">     <div v-show="isNotLogedIn" style="font-family: 华文隶书;font-size: 30px">  <v-btn style="font-size: 30px;position: absolute;left: 2%;top:1%;background-color:#f1e58e;" @click="logInPlayer">  登录  </v-btn>  </div>  <div v-show="!isNotLogedIn" style="font-size: 30px;position: absolute;left: 5%;top:1%;font-family: 华文隶书;">  欢迎，<span>{{playerName}}</span>  <br>  <v-btn style="background-color:#f1e58e;font-size: 25px" @click="createGame">  创建棋局  </v-btn>  <v-btn style="background-color:#f1e58e;font-size: 25px" @click="userExit">  退出登录  </v-btn>  </div>   <div style="font-family: 华文隶书;position:absolute;left: 15%;top: 13%;width: 70%;height: 60%;overflow: auto">   <div v-for="theGame in game" :key="theGame.gameId" v-show="theGame.isUsing===0">  <v-chip style="font-size: 35px;margin-top: 3%;margin-left: 5%;background-color:#f1e58e;height: 55px;">  {{theGame.player1}} 的棋局  </v-chip>  <v-btn fab style="background-color:#f1e58e; position:relative;left: 5%;margin-top: 3%;" @click="playWithOthers(theGame.player1)">  <img src="../assets/beginGame.png" title="与此人游戏" width="20%">  </v-btn>  </div>   </div>    *<!-- 展示提醒-->* <div style="position: absolute;bottom: 4%;right: 0%">  <v-snackbar :timeout="3000" :value="true" absolute centered right outlined color="red accent-2" v-model="isShowRed">  {{content}}  </v-snackbar>  <v-snackbar :timeout="3000" :value="true" absolute centered right color="orange darken-2" v-model="isShowYellow">  {{content}}  </v-snackbar>  <v-snackbar :timeout="3000" :value="true" absolute centered right outlined color="success" v-model="isShowGreen">  {{content}}  </v-snackbar>  </div>       </div> </template>  <script> import *swal* from 'sweetalert'; import request from "@/utils/request";    export default {  name: 'Home',  data(){  return{  isNotLogedIn:true,  playerName:'',  game:[],  searchRoomInterval:'',  ifClickPlayBeforeLog:false,   content:'',isShowRed:false,isShowYellow:false,isShowGreen:false,   }  },   methods:{   seleteRoom(){  this.searchRoomInterval = *setInterval*(()=>{  this.getAllGame();  },1000);  },   turnToRoom(){  this.$router.push('/room');  },   *//刷新时操作* judgeIfLoged(){  this.getAllGame();  var name = sessionStorage.getItem('playerName');  if (name===null||name==='') this.isNotLogedIn = true;  else {  this.playerName = name;  this.isNotLogedIn = false;  }  },   *//用户登录* logInPlayer(theOtherPlayer){  *swal*("你的名字", {  content: "input",  icon: "info",  })  .then((value) => {   if (value !== null){  this.playerName=value;  sessionStorage.setItem('playerName',value);   request.post("/api/player/savePlayer",{  playerName:this.playerName,  playTime:0,  winTime:0,  }) .then( (response) => {  if (response.code==="0"){  this.content=response.msg;  this.isShowGreen=true;  }else{  this.content=response.msg;  this.isShowYellow = true;  }  this.isNotLogedIn=false;  *//如果是在未登陆前点击登录引起登录，则进入游戏* if (this.ifClickPlayBeforeLog === true) this.playWithOthers(theOtherPlayer);  })  .catch((error) => {  console.log(error);  });  }  });  },   *//退出登录* userExit(){  this.isNotLogedIn = true;  sessionStorage.setItem('playerName','');  this.playerName = '';  },   *//得到已有的棋局* getAllGame(){  request.post("/api/game/getAllGame",{   }) .then( (response) => {  console.log(response)  this.game = response;  })  .catch((error) => {  console.log(error);  });  },   *//自己开一局* createGame(){  request.post("/api/game/createGame",{  player1:this.playerName,  player2:'',  }) .then( (response) => {  console.log(response)  if (response.code==="0"){  *clearInterval*(this.searchRoomInterval);  this.$router.push('/room');  }else{  this.content=response.msg;  this.isShowRed = true;  }  })  .catch((error) => {  console.log(error);  });  },   *//与他人对战* playWithOthers(theOtherPlayer){  if (this.isNotLogedIn === true){  this.content = '请先登录！'  this.isShowYellow = true;  this.ifClickPlayBeforeLog = true;  this.logInPlayer(theOtherPlayer);  }else{  request.post("/api/game/playWithOthers",{  player2:this.playerName,  player1:theOtherPlayer,  }) .then( (response) => {  console.log(response)  if (response.code==="0"){  *clearInterval*(this.searchRoomInterval);  this.$router.push('/room');  }else{  this.content=response.msg;  this.isShowRed = true;  }  })  .catch((error) => {  console.log(error);  });   }  },         },   mounted() {  this.judgeIfLoged();  this.seleteRoom();     },    }   </script>   <style scoped>  .background {  background: url("../assets/helloBackground.png");  background-size: 100% 100%;  height: 100%;  position: fixed;  width: 100% }  .fade-enter{  opacity: 0; } .fade-enter-active{  transition: opacity 1s; }  .fade-leave-to{  opacity: 0; } .fade-leave-active{  transition: opacity 1s; }    </style>  //room.vue  <template> <div class="background">   <v-btn style="background-color:#f1e58e;font-size: 25px;font-family: 华文隶书;position: absolute;right: 2%;bottom: 5%;" @click="back">  退出游戏  </v-btn>     <v-btn style="background-color:#f1e58e;font-size: 25px;font-family: 华文隶书;position: absolute;right: 25%;bottom: 15%;"  :disabled="isDisabledConfirmButton" @click="confirmPiece">  选定  </v-btn>  <v-btn style="background-color:#f1e58e;font-size: 25px;font-family: 华文隶书;position: absolute;right: 18%;bottom: 15%;"  :disabled="isDisabledConfirmButton" @click="queryPeace">  求和  </v-btn> *<!-- <v-btn style="background-color:#f1e58e;font-size: 25px;font-family: 华文隶书;position: absolute;right: 11%;bottom: 15%;"--> <!-- :disabled="isDisabledConfirmButton">--> <!-- 悔棋--> <!-- </v-btn>-->* <v-btn style="background-color:#f1e58e;font-size: 25px;font-family: 华文隶书;position: absolute;right: 14%;bottom: 5%;"  v-show="isFinished" @click="quaryPlayAgain">  再来一局  </v-btn>   *<!-- 显示玩家-->* <div style="font-family: 华文隶书;font-size:40px;position:absolute;right: 0%;top: 13%;width: 30%;height: 20%;overflow: auto">  <img src="../assets/player1.png" title="玩家1" width="20%">  <span style="margin-left: 10%;">{{player1}}</span>  <span v-show="this.playerName===this.player1">（我）</span>  </div>  <div style="font-family: 华文隶书;font-size:50px;position:absolute;right: 10%;top: 38%;">  V S  </div>  <div style="font-family: 华文隶书;font-size:40px;position:absolute;right: 0%;top: 52%;width: 30%;height: 20%;overflow: auto">  <img src="../assets/player2.png" title="玩家2" width="20%">  <v-progress-circular  :size="50" color="amber" indeterminate style="margin-left: 10%"  v-show="!isAnotherPlayerEnter"  ></v-progress-circular>  <div v-show="!isAnotherPlayerEnter">  等待其他玩家加入…  </div>  <span style="margin-left: 10%;">{{player2}}</span>  <span v-show="this.playerName===this.player2">（我）</span>  </div>  *<!-- 五子棋盘-->* <div style="height: 750px;width: 750px;position: absolute;left: 170px;top: 35px;">  <table style="background-color:lightgoldenrodyellow;">  <tr v-for="i in 15" :key="i" >  <td v-for="j in 15" :key="j">  <div style="height: 47px;width: 47px;position:relative;text-align: center;"  @click="clickPiece(i,j)">  <v-btn  :id="i+','+j"  fab style="width: 30px;height: 30px;position: absolute;left: 20%;top: 20%;">  </v-btn>   <img v-if="i===8&&j===8||i===3&&j===3||i===3&&j===12||i===12&&j===3||i===12&&j===12"  src="../assets/lineWithPosition.png" width="50px" height="50px">  <img v-else src="../assets/line.png" width="50px" height="50px">  </div>  </td>  </tr>  </table>   </div>                            *<!-- 展示提醒-->* <div style="position: absolute;bottom: 4%;right: 12%">  <v-snackbar :timeout="3000" :value="true" absolute centered right outlined color="red accent-2" v-model="isShowRed">  {{content}}  </v-snackbar>  <v-snackbar :timeout="3000" :value="true" absolute centered right color="orange darken-2" v-model="isShowYellow" >  {{content}}  </v-snackbar>  <v-snackbar :timeout="6000" absolute centered right color="green" v-model="isShowGreen" v-show="isShowGreen">  {{content}}  </v-snackbar>  </div>  </div> </template>  <script> import *swal* from 'sweetalert'; import request from "@/utils/request";   export default {  name: "Room",   data(){  return{  content:'',isShowRed:false,isShowYellow:false,isShowGreen:false,   isAnotherPlayerEnter:false,isCanIGo:false,isDisabledConfirmButton:true,isFinished:false,   player1:'',player2:'',playerName:'',allPieces:[],winPieces:[],   playerInterval:'',pieceInterval:'',   nowChoosedPieceId:'',                     }  },   methods:{  selectPlayer(){  this.getPlayer();  this.playerInterval = *setInterval*(()=>{  this.getPlayer();  },6000);  },   selectPieces(){  this.pieceInterval = *setInterval*(()=>{  this.getNextPiece();  this.judgeIfMyTurn();  },500);    },   hideAllPiece(){  for (var i=1;i<=15;i++)  for (var j=1;j<=15;j++){  document.getElementById(i+','+j).hidden = true;  }  },     back(){  *clearInterval*(this.playerInterval);*//先停止刷新  clearInterval*(this.pieceInterval);  *swal*("确定退出游戏吗？", {  buttons: {  exit:{  text:"确定退出",  },  noExit:{  text:"取消",  },  },  })  .then((value) => {  if (value==='exit'){  request.post("/api/game/existRoom",{  player1:this.player1,  player2:this.player2,  }) .then( (response) => {  console.log(response);  this.$router.push('/');  })  .catch((error) => {  console.log(error);  });   }else if(value==='noExit'){  this.selectPlayer();  this.selectPieces();  }  });    },   getPlayer(){  *//得到玩家12，本地玩家放到player1中* request.post("/api/game/getPlayer",{  player1:sessionStorage.getItem('playerName'),  }) .then( (response) => {  console.log(response)  *//对方中途离开* if (response===''){  *clearInterval*(this.playerInterval);  *clearInterval*(this.pieceInterval);  this.isAnotherPlayerEnter = false;  var existPlayer = this.playerName===this.player1?this.player2:this.player1;  if (existPlayer!==this.playerName){  *swal*({  title: existPlayer+"退出棋局",  icon: "success",  button:{  text:"确定",  value:"confirm"  },  }).then(value => {  console.log(value)  this.$router.push('/');  });  }  }else{  *//否则* this.player1 = response.player1;  this.player2 = response.player2;  if (this.player2 !== ''){*//如果第二个人加入* this.isAnotherPlayerEnter = true;  }  }   })  .catch((error) => {  console.log(error);  });  },    *//选择一个点* clickPiece(i,j){  if (this.isCanIGo === true){  if (document.getElementById(i+','+j).hidden === true){  this.isDisabledConfirmButton=false;  *//前面选的点消失，换成现在选的点* if (this.nowChoosedPieceId === ''){  this.nowChoosedPieceId = i+','+j;  document.getElementById(this.nowChoosedPieceId).hidden = false;  if (this.playerName===this.player1)document.getElementById(this.nowChoosedPieceId).style.backgroundColor='white';  else document.getElementById(this.nowChoosedPieceId).style.backgroundColor='black';  }else{  document.getElementById(this.nowChoosedPieceId).hidden = true;  this.nowChoosedPieceId = i+','+j;  document.getElementById(this.nowChoosedPieceId).hidden = false;  if (this.playerName===this.player1)document.getElementById(this.nowChoosedPieceId).style.backgroundColor='white';  else document.getElementById(this.nowChoosedPieceId).style.backgroundColor='black';  }  }  }  },   *//确定选择此点* confirmPiece(){  request.post("/api/piece/addOnePiece",{  player:this.playerName,  pX:this.nowChoosedPieceId.split(',')[0],  pY:this.nowChoosedPieceId.split(',')[1],  }) .then( (response) => {  console.log(response);  this.isDisabledConfirmButton = true;  this.isCanIGo = false;  this.judgeIfFiveWin(this.nowChoosedPieceId.split(',')[0],this.nowChoosedPieceId.split(',')[1])  this.nowChoosedPieceId = '';*//防止下次放棋子的时候有这个* })  .catch((error) => {  console.log(error);  });  },         getAllPiece(){  request.post("/api/piece/getAllPiece",{  player:this.playerName,  }) .then( (response) => {  console.log(response);  this.allPieces = response;  for (var i=0;i<response.length;i++){  var thePieceId = response[i].pX+','+response[i].pY;  if (document.getElementById(thePieceId).hidden===true){  document.getElementById(thePieceId).hidden=false;  if (response[i].player===this.player1)  document.getElementById(thePieceId).style.backgroundColor='white';  else document.getElementById(thePieceId).style.backgroundColor='black';  }   }   })  .catch((error) => {  console.log(error);  });  },   *//0,0 :求和  //-1,-1:同意求和  //-2,-2:不同意求和  //-3,-3:请求再来一局  //-4,-4:同意再来一局  //-5,-5:不同意再来一局  //-11,-11:棋局输赢已定  //-10,-10:游戏结束* getNextPiece(){  request.post("/api/piece/getAllPiece",{  player:this.playerName,  }) .then( (response) => {  console.log(response);  if (response.length===0)return;  else{  this.allPieces = response;  var thePieceId = response[response.length-1].pX+','+response[response.length-1].pY;    *//如果求和* if (thePieceId==='0,0'){  if (response[response.length-1].player!==this.playerName){  *swal*("对方求和，是否同意？", {  buttons: {  agree:{text:"同意",},  disagree:{text:"不同意",},  },  })  .then((value) => {  if (value==='agree'){  *//从0，0改为-1，-1* request.post("/api/piece/changePiece",{  pX:0,  pY:-1,  }) .then( (response) => {  console.log(response);   })  .catch((error) => {  console.log(error);  });  }else if(value==='disagree'){  *//从0，0改为-2，-2* request.post("/api/piece/changePiece",{  pX:0,  pY:-2,  }) .then( (response) => {  console.log(response);   })  .catch((error) => {  console.log(error);  });  }  });  }  }else if (thePieceId==='-1,-1'){  if (response[response.length-1].player===this.playerName){  request.post("/api/piece/deletePiece",{  pX:-1,  pY:-1,  }) .then( (response) => {  console.log(response);  *swal*("对方同意求和！", {buttons: {agree:{text:"确定",},},});  *//结束标志：-10，-10* request.post("/api/piece/addOnePiece",{  player:this.playerName,  pX:-10,  pY:-10,  }) .then( (response) => {  console.log(response);  this.isFinished = true;  this.isCanIGo = false;  this.isDisabledConfirmButton = true;  })  .catch((error) => {  console.log(error);  });  })  .catch((error) => {  console.log(error);  });  }  }else if (thePieceId==='-2,-2'){  if (response[response.length-1].player===this.playerName){  request.post("/api/piece/deletePiece",{  pX:-2,  pY:-2,  }) .then( (response) => {  console.log(response);  *swal*("对方不同意求和！", {buttons: {agree:{text:"确定",},},})  })  .catch((error) => {  console.log(error);  });  }  }*//结束标志：-10，-10* else if (thePieceId==='-10,-10'){   this.isFinished = true;  this.isCanIGo = false;  this.isDisabledConfirmButton = true;     }else if (thePieceId==='-3,-3'){  if (response[response.length-1].player!==this.playerName){   *swal*("对方请求再来一局，是否同意？", {  buttons: {  agree:{text:"同意",},  disagree:{text:"不同意",},  },  })  .then((value) => {  if (value==='agree'){  *//从-3，-3改为-4，-4* request.post("/api/piece/changePiece",{  pX:-3,  pY:-4,  }) .then( (response) => {  console.log(response);  *clearInterval*(this.playerInterval);  *clearInterval*(this.pieceInterval);   *setTimeout*(()=>{  request.post("/api/piece/deleteAllPieces",{  player:this.playerName,  }) .then( (response) => {  console.log(response);  location.reload();  })  .catch((error) => {  console.log(error);  });  },1500);   })  .catch((error) => {  console.log(error);  });  }else if(value==='disagree'){  *//从-3，-3改为-5，-5* request.post("/api/piece/changePiece",{  pX:-3,  pY:-5,  }) .then( (response) => {  console.log(response);   this.content='准备离场……'  this.isShowGreen = true;  *clearInterval*(this.playerInterval);  *clearInterval*(this.pieceInterval);  *setTimeout*(()=>{  request.post("/api/game/existRoom",{  player1:this.player1,  player2:this.player2,  }) .then( (response) => {  console.log(response);  this.$router.push('/');  })  .catch((error) => {  console.log(error);  });  },1200)    })  .catch((error) => {  console.log(error);  });  }  });   }  }else if (thePieceId==='-4,-4'){  if (response[response.length-1].player===this.playerName){  request.post("/api/piece/deletePiece",{  pX:-4,  pY:-4,  }) .then( (response) => {  console.log(response);  *swal*("对方同意，棋局再次开始！", {buttons: {agree:{text:"确定",},},});   *setTimeout*(()=>{  location.reload();  },1600)   })  .catch((error) => {  console.log(error);  });  }  }else if (thePieceId==='-5,-5'){  if (response[response.length-1].player===this.playerName){  request.post("/api/piece/deletePiece",{  pX:-2,  pY:-2,  }) .then( (response) => {  console.log(response);   *clearInterval*(this.playerInterval);  *clearInterval*(this.pieceInterval);  *swal*("对方离场，此局游戏结束！", {buttons: {agree:{text:"确定",},},}).then(()=>{  this.$router.push('/');*//此时对方已经把数据库删了，本方不需要删除* })  })  .catch((error) => {  console.log(error);  });  }  }else if (thePieceId==='-11,-11'){   this.isFinished = true;  this.isCanIGo = false;  this.isDisabledConfirmButton = true;    if (this.winPieces.length===0){  request.post("/api/piece/judgeIfFiveWin",{  player:(this.playerName===this.player1?this.player2:this.player1),  pX:response[response.length-2].pX,  pY:response[response.length-2].pY,  }) .then( (response1) => {  console.log(response1);  this.winPieces = response1;  })  .catch((error) => {  console.log(error);  });  }        *swal*((response[response.length-1].player===this.playerName?"你":"对方")+"赢了！", {button: {agree:{text:"确定",},},});    *setTimeout*(()=>{  console.log(this.winPieces)  for (var i=0;i<this.winPieces.length;i++){  var theId = this.winPieces[i].pX+','+this.winPieces[i].pY;  document.getElementById(theId).style.backgroundColor='#00fdf3';  }  request.post("/api/piece/changePiece",{  player:this.playerName,  pX:-11,  pY:-10,  }) .then( (response) => {  console.log(response);  })  .catch((error) => {  console.log(error);  });  },700);   }     else if (document.getElementById(thePieceId).hidden===true){*//显示下了的棋子* document.getElementById(thePieceId).hidden=false;  if (response[response.length-1].player===this.player1)  document.getElementById(thePieceId).style.backgroundColor='white';  else document.getElementById(thePieceId).style.backgroundColor='black';  }  }    })  .catch((error) => {  console.log(error);  });  },   *//判断是否轮到我* judgeIfMyTurn(){   *//不是求和悔棋结束等特殊情况，棋局上有棋* if (this.allPieces.length!==0&&this.allPieces[this.allPieces.length-1].player!==this.playerName&&  this.allPieces[this.allPieces.length-1].pX>0){  this.isCanIGo = true;  this.content = '轮到你了！';  this.isShowGreen = true;  }else if (this.allPieces.length!==0){  this.isCanIGo = false;*//防止刷新后的客方可以放* }  },         *//求和：player是求和的人，x=0，y=0提出，-1，-1同意，-2，-2不同意* queryPeace(){  request.post("/api/piece/addOnePiece",{  player:this.playerName,  pX:0,  pY:0,  }) .then( (response) => {  console.log(response);   })  .catch((error) => {  console.log(error);  });  },       *//悔棋：player是悔棋的人，x=-，y=-      //请求再来一局,x=-3,y=-3* quaryPlayAgain(){  request.post("/api/piece/addOnePiece",{  player:this.playerName,  pX:-3,  pY:-3,  }) .then( (response) => {  console.log(response);  })  .catch((error) => {  console.log(error);  });  },     *//判断输赢* judgeIfFiveWin(x,y){  request.post("/api/piece/judgeIfFiveWin",{  player:this.playerName,  pX:x,  pY:y,  }) .then( (response) => {  console.log(response);   if (response.length>=5){*//说明赢了* this.winPieces = response;  *setTimeout*(()=>{  request.post("/api/piece/addOnePiece",{  player:this.playerName,  pX:-11,  pY:-11,  }) .then( (response) => {  console.log(response);  this.isFinished = true;  this.isCanIGo = false;  this.isDisabledConfirmButton = true;   })  .catch((error) => {  console.log(error);  });  },1200)   }     })  .catch((error) => {  console.log(error);  });  },         },   mounted() {  *clearInterval*(this.pieceInterval);  *clearInterval*(this.playerInterval);   this.playerName = sessionStorage.getItem('playerName');*//设置playerName* this.selectPlayer();*//不断刷新player1和player2* this.selectPieces();  this.getAllPiece();  *setTimeout*(()=>{*//前面selectPlayer执行完一次得到player1，2后，判断使用者是1还是2，让客方（2）先走* if (this.playerName===this.player2&&this.allPieces.length===0){*//如果游戏已经开始* this.isCanIGo = true;  this.content = '你是客方，请先放置棋子！'  this.isShowGreen = true;  }  this.judgeIfMyTurn();   },1000)  this.hideAllPiece();*//先隐藏所有棋子* },  } </script>  <style scoped> .background {  background: url("../assets/gameBackground.png");  background-size: 100% 100%;  height: 100%;  position: fixed;  width: 100% }  .fade-enter{  opacity: 0; } .fade-enter-active{  transition: opacity 1s; }  .fade-leave-to{  opacity: 0; } .fade-leave-active{  transition: opacity 1s; }  .fadenum{  animation:fadenum 1s 1; }  @keyframes fadenum{  0%{opacity: 0;}  100%{opacity: 1;} }  .fadenumR{  animation:fadenumR 1s 1;  height: 600px; }  @keyframes fadenumR{  0%{opacity: 0;}  100%{opacity: 1;} }  .barContent{  font-family: 华文隶书;  font-size: 30px; }   </style>  【后端代码】  //pieceMapper：  package com.example.student\_arrangement.mapper;  import com.baomidou.mybatisplus.core.mapper.BaseMapper; import com.example.student\_arrangement.entity.Game; import com.example.student\_arrangement.entity.Piece; import org.apache.ibatis.annotations.\*; import org.springframework.stereotype.Repository;  @Mapper @Repository public interface PieceMapper extends BaseMapper<Piece> {  @Insert("INSERT INTO `chess`.`piece` (`gameId`, `player`, `pX`, `pY`) VALUES (#{gameId}, #{player}, #{pX}, #{pY})")  public void insertPiece(@Param("gameId")int gameId,@Param("player")String player,@Param("pX")int pX,@Param("pY")int pY);   @Select("SELECT \* FROM chess.piece")  public Piece[] selectAll();   @Select("SELECT \* FROM chess.piece where gameId = #{gameId}")  public Piece[] selectByGameId(@Param("gameId")int gameId);   @Select("SELECT \* FROM chess.piece where (`pX` = #{opX} and `pY` = #{opY})")  public Piece selectByPxy(@Param("opX")int opX,@Param("opY")int opY);   @Select("SELECT \* FROM chess.piece where (`pX` = #{opX} and `pY` = #{opY} and `player` = #{player})")  public Piece selectByPxyAndPlayer(@Param("opX")int opX,@Param("opY")int opY,@Param("player")String player);   @Delete("DELETE FROM `chess`.`piece` WHERE (`gameId` = #{gameId})")  public void deleteByGameId(@Param("gameId")int gameId);   @Update("UPDATE `chess`.`piece` SET `pX` = #{pX}, `pY` = #{pY} WHERE (pieceId = #{pieceId})")  public void changePieceById(@Param("pX")int pX,@Param("pY")int pY,@Param("pieceId")int pieceId);   @Delete("DELETE FROM `chess`.`piece` WHERE (`pX` = #{opX} and `pY` = #{opY})")  public void deleteByPosition(@Param("opX")int opX,@Param("opY")int opY);   @Delete("DELETE FROM `chess`.`piece` WHERE (gameId = #{gameId})")  public void deleteAllPieceById(@Param("gameId")int gameId);   @Select("SELECT \* FROM chess.piece where (`pX` = #{pX} and `player` = #{player})")  public Piece[] selectByPxAndPlayer(@Param("pX")int pX,@Param("player")String player);   @Select("SELECT \* FROM chess.piece where (`pY` = #{pY} and `player` = #{player})")  public Piece[] selectByPyAndPlayer(@Param("pY")int pY,@Param("player")String player); }  //playerMapper：  package com.example.student\_arrangement.mapper;  import com.baomidou.mybatisplus.core.mapper.BaseMapper; import com.example.student\_arrangement.entity.Player; import org.apache.ibatis.annotations.Insert; import org.apache.ibatis.annotations.Mapper; import org.apache.ibatis.annotations.Param; import org.apache.ibatis.annotations.Select; import org.springframework.stereotype.Repository;  @Mapper @Repository public interface PlayerMapper extends BaseMapper<Player> { *// @Insert("INSERT INTO `shujvjiegou`.`nodes` (`nodeid`, `x`, `y`) VALUES (#{nodeid}, #{x}, #{y})") // public void insertNode(@Param("nodeid")int nodeid, @Param("x")String x, @Param("y")String y);* @Select("SELECT \* FROM chess.player where playerName = #{playerName}")  public Player selectByName(@Param("playerName")String playerName);   @Insert("INSERT INTO `chess`.`player` (`playerName`, `playTime`, `winTime`) VALUES (#{playerName},#{playTime},#{winTime})")  public void insertPlayer(@Param("playerName")String playerName,@Param("playTime")int playTime,@Param("winTime")int winTime);   }  //playerController：  import javax.annotation.Resource;  @RestController @CrossOrigin   @RequestMapping(value = "/player") public class PlayerController {  @Resource  PlayerMapper playerMapper;   @RequestMapping(value = "/savePlayer",method = RequestMethod.*POST*)  public Result<?> savePlayer(@RequestBody Player player){  Player thePlayer = playerMapper.selectByName(player.getPlayerName());  if (ObjectUtils.*isEmpty*(thePlayer)){  playerMapper.insertPlayer(player.getPlayerName(),player.getPlayTime(),player.getWinTime());  return Result.*srror*("0","欢迎你，"+player.getPlayerName()+"。 开始享受五子棋吧~");  }else {  return Result.*srror*("0","欢迎你，"+player.getPlayerName()+"!");  }  }  }  //gameController：  package com.example.student\_arrangement.controller;  import com.baomidou.mybatisplus.core.toolkit.ObjectUtils; import com.example.student\_arrangement.common.Result; import com.example.student\_arrangement.entity.Game; import com.example.student\_arrangement.mapper.GameMapper; import com.example.student\_arrangement.mapper.PieceMapper; import org.springframework.web.bind.annotation.\*;  import javax.annotation.Resource;  @RestController @CrossOrigin   @RequestMapping(value = "/game") public class GameController {  @Resource  GameMapper gameMapper;  @Resource  PieceMapper pieceMapper;   @RequestMapping(value = "/createGame",method = RequestMethod.*POST*)  public Result<?> createGame(@RequestBody Game game){  Game theGame = gameMapper.selectByPlayer12(game.getPlayer1(),game.getPlayer2());  if(ObjectUtils.*isEmpty*(theGame)){  gameMapper.insertGame(game.getPlayer1(),game.getPlayer2(),0);  return Result.*srror*("0","开场成功！");  }else{  return Result.*srror*("1","重复开场，开场失败！");  }  }   @RequestMapping(value = "/getAllGame",method = RequestMethod.*POST*)  public Game[] getAllGame(@RequestBody Game game){  return gameMapper.selectAll();  }   @RequestMapping(value = "/playWithOthers",method = RequestMethod.*POST*)  public Result<?> playWithOthers(@RequestBody Game game){  if (!game.getPlayer1().equals("")&&!game.getPlayer2().equals("")){  gameMapper.updateAnotherPlayer(game.getPlayer1(),game.getPlayer2());  return Result.*srror*("0","进入！");  }else{  return Result.*srror*("1","进入失败！");  }  }   @RequestMapping(value = "/existRoom",method = RequestMethod.*POST*)  public Result<?> existRoom(@RequestBody Game game){  pieceMapper.deleteByGameId(gameMapper.selectByPlayer1OR2(game.getPlayer1()).getGameId());  gameMapper.deleteByPlayer12(game.getPlayer1(),game.getPlayer2());  return Result.*success*();  }   @RequestMapping(value = "/getPlayer",method = RequestMethod.*POST*)  public Game getPlayer(@RequestBody Game game){  return gameMapper.selectByPlayer1OR2(game.getPlayer1());  }      }  //pieceController：  package com.example.student\_arrangement.controller;  import com.baomidou.mybatisplus.core.toolkit.ObjectUtils; import com.example.student\_arrangement.common.Result; import com.example.student\_arrangement.entity.Game; import com.example.student\_arrangement.entity.Piece; import com.example.student\_arrangement.mapper.GameMapper; import com.example.student\_arrangement.mapper.PieceMapper; import org.springframework.web.bind.annotation.\*;  import javax.annotation.Resource;  @RestController @CrossOrigin   @RequestMapping(value = "/piece") public class PieceController {  @Resource  PieceMapper pieceMapper;  @Resource  GameMapper gameMapper;   @RequestMapping(value = "/addOnePiece",method = RequestMethod.*POST*)  public Result<?> addOnePiece(@RequestBody Piece piece){  Game theGame = gameMapper.selectByPlayer1OR2(piece.getPlayer());  pieceMapper.insertPiece(theGame.getGameId(),piece.getPlayer(),piece.getpX(),piece.getpY());  return Result.*success*();  }   @RequestMapping(value = "/getAllPiece",method = RequestMethod.*POST*)  public Piece[] getAllPiece(@RequestBody Piece piece){  if (gameMapper.selectByPlayer1OR2(piece.getPlayer())!=null)  return pieceMapper.selectByGameId(gameMapper.selectByPlayer1OR2(piece.getPlayer()).getGameId());  else return null;  }   @RequestMapping(value = "/changePiece",method = RequestMethod.*POST*)  public Result<?> changePiece(@RequestBody Piece piece){*//x是原值，y是改后值* if (pieceMapper.selectByPxy(piece.getpX(),piece.getpX())==null)return Result.*success*();  pieceMapper.changePieceById(piece.getpY(),piece.getpY(),  pieceMapper.selectByPxy(piece.getpX(),piece.getpX()).getPieceId());   return Result.*success*();  }   @RequestMapping(value = "/deletePiece",method = RequestMethod.*POST*)  public Result<?> deletePiece(@RequestBody Piece piece){*//x是原值，y是改后值* pieceMapper.deleteByPosition(piece.getpX(),piece.getpY());  return Result.*success*();  }   @RequestMapping(value = "/deleteAllPieces",method = RequestMethod.*POST*)  public Result<?> deleteAllPieces(@RequestBody Piece piece){  pieceMapper.deleteAllPieceById(gameMapper.selectByPlayer1OR2(piece.getPlayer()).getGameId());  return Result.*success*();  }   @RequestMapping(value = "/judgeIfFiveWin",method = RequestMethod.*POST*)  public Piece[] judgeIfFiveWin(@RequestBody Piece piece){   *//横向* Piece[] xPiece = pieceMapper.selectByPxAndPlayer(piece.getpX(),piece.getPlayer());  if (xPiece.length>=5){  xPiece = binSortY(xPiece);  int[] xArr = new int[xPiece.length];  for (int i=0;i<xPiece.length;i++)  xArr[i] = xPiece[i].getpY();  if(ifHaveContinueFive(xArr,0)) {  Piece[] result = new Piece[5];  for (int j=0;j<5;j++)result[j] = xPiece[begin++];  flag = 1;  begin = -1;  return result;  }  }     *//纵向* Piece[] yPiece = pieceMapper.selectByPyAndPlayer(piece.getpY(),piece.getPlayer());  if (yPiece.length>=5){  Piece [] temp = binSortX(yPiece);  for (int i=0;i<temp.length;i++)yPiece[i] = temp[i];  int[] yArr = new int[yPiece.length];  for (int i=0;i<yPiece.length;i++)  yArr[i] = yPiece[i].getpX();  if(ifHaveContinueFive(yArr,0)) {  Piece[] result = new Piece[5];  for (int j=0;j<5;j++)result[j] = yPiece[begin++];  flag = 1;  begin = -1;  return result;  }  }     *//斜率为-1* int kX = piece.getpX();  int kY = piece.getpY();  Piece positiveKPieceTemp[] = new Piece[15];  int ind1 = 0;  for (;kX>0&&kY>0;kX--,kY--){  Piece thePiece = pieceMapper.selectByPxyAndPlayer(kX,kY,piece.getPlayer());  if (thePiece!=null)positiveKPieceTemp[ind1++] = thePiece;  }   kX = piece.getpX()+1;  kY = piece.getpY()+1;  for (;kX<=15&&kY<=15;kX++,kY++){  Piece thePiece = pieceMapper.selectByPxyAndPlayer(kX,kY,piece.getPlayer());  if (thePiece!=null)positiveKPieceTemp[ind1++] = thePiece;  }   Piece positiveKPiece[] = new Piece[ind1];  for (int i=0;i<ind1;i++)positiveKPiece[i] = positiveKPieceTemp[i];  if (ind1>=5){   Piece temp[] = binSortY(positiveKPiece);  int [] kpArrX = new int[ind1];  for (int i=0;i<ind1;i++){  kpArrX[i] = temp[i].getpX();  }  if(ifHaveContinueFive(kpArrX,0)) {  Piece[] result = new Piece[5];  for (int j=0;j<5;j++)  result[j] = temp[begin++];  flag = 1;  begin = -1;  return result;  }  }    *//斜率为1* kX = piece.getpX();  kY = piece.getpY();  Piece negativeKPieceTemp[] = new Piece[15];  int ind2 = 0;  for (;kX>0&&kY>0;kX--,kY++){  Piece thePiece = pieceMapper.selectByPxyAndPlayer(kX,kY,piece.getPlayer());  if (thePiece!=null)negativeKPieceTemp[ind2++] = thePiece;  }   kX = piece.getpX()+1;  kY = piece.getpY()-1;  for (;kX<=15&&kY<=15;kX++,kY--){  Piece thePiece = pieceMapper.selectByPxyAndPlayer(kX,kY,piece.getPlayer());  if (thePiece!=null)negativeKPieceTemp[ind2++] = thePiece;  }   Piece negativeKPiece[] = new Piece[ind2];  for (int i=0;i<ind2;i++)negativeKPiece[i] = negativeKPieceTemp[i];  if (ind2>=5){    Piece temp[] = binSortY(negativeKPiece);  int [] knArrY = new int[ind2];  for (int i=0;i<ind2;i++){  knArrY[i] = temp[i].getpY();   }  if(ifHaveContinueFive(knArrY,0)){  Piece[] result = new Piece[5];  for (int j=0;j<5;j++){  result[j] = temp[begin++];  }   flag = 1;  begin = -1;  return result;  }  }    return null;  }   int flag = 1;  int begin = -1;  public boolean ifHaveContinueFive(int []arr,int index){  if (flag==5){  return true;  }  else if (index==arr.length-1)return false;  else {  if (arr[index++]+1 == arr[index]){  if (begin==-1)begin = index-1;  flag++;  return ifHaveContinueFive(arr,index);  }else return ifHaveContinueFive(arr,index);  }  }   public Piece[] binSortX(Piece[] arr){  Piece[] bin = new Piece[15];  for (int i=0;i<15;i++)bin[i] = null;  for (int i=0;i<arr.length;i++)  bin[arr[i].getpX()] = arr[i];  Piece[] result = new Piece[arr.length];  int ind = 0;  for (int i=0;i<15;i++){  if (bin[i]!=null)result[ind++] = bin[i];  }  return result;  }   public Piece[] binSortY(Piece[] arr){  Piece[] bin = new Piece[15];  for (int i=0;i<15;i++)bin[i] = null;  for (int i=0;i<arr.length;i++)  bin[arr[i].getpY()] = arr[i];  Piece[] result = new Piece[arr.length];  int ind = 0;  for (int i=0;i<15;i++){  if (bin[i]!=null)result[ind++] = bin[i];  }  return result;  }      } | | | |