

**Course Experiment Report**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course:** | Java Language | | | | | | |
|  |  | | | | | | |
| **Semester:** | 1-18th | **week** | 2nd | **year** | | 1st | **term** |
|  |  |  |  |  | |  |  |
| **Major:** | Software Engineering | | | | | **Class:** | 2019 |
|  |  | | | | |  |  |
| **Student name:** | 温长锟 | | **Student No.:** | | 222020321062106 | | |
|  |  | |  | |  | | |
| **Teacher:** | Wang Xiaomeng | | | | | | |

College of Computer and Information Science

|  |  |  |  |
| --- | --- | --- | --- |
| Project | Exp7 GUI | | |
| Time | 2021.12.16 | Type | □Verification □Design □Synthetical |
| 1. Answer the questions  (1) What else can be improved in the program?  First, this push box game can add a few more levels. We can set a variable to store scores and set a multi level.  Second, we can set a step-by-step customs clearance button. After clicking this button through event binding, our GUI interface can enable it to show how to clear customs correctly step by step  Third, we can use JDBC to link the database, save the status, set the user name and password, so that the game can save the game records of each user  The fourth point may be to join the combat system. By changing the original click event to keyboard input, one keyboard can support two users.  Fifth, after adding rich functions, we may try to separate the front and back ends and publish it on the LAN for everyone to play   1. How to develop a map editing program for this game?   Maybe we can set up a GUI interface for editing map and bind its editing operations through click or keyboard input events to achieve the desired results   1. All Codes   1.  import javafx.scene.image.Image; import javafx.scene.image.ImageView;  public abstract class Element extends ImageView {  //图像元素类型  public final static int *MAN* = 0;  public final static int *BOX* = 1;  public final static int *WALL* = 2;  public final static int *TARGET* = 3;  public final static int *BACKGROUND* = 4;   protected Element(Image img) {  super(img);  //设置展示尺寸  this.setFitHeight(MapPane.CELL\_SIZE);  this.setFitWidth(MapPane.CELL\_SIZE);  } }  2.  import javafx.scene.image.Image;   public class Box extends MovingElement {    public Box(Image img) {  super(img);   }  }  3.  import javafx.scene.image.Image;  public class Man extends MovingElement {  public Man(Image manImg) {  super(manImg);  } }  4.  import javafx.scene.image.Image;  public class Wall extends Element {  public Wall(Image img) {  super(img);  } }  5.  import javafx.scene.image.Image;  public class Wall extends Element {  public Wall(Image img) {  super(img);  } }  6.  import javafx.scene.image.Image;  public class MovingElement extends Element {  protected MovingElement(Image img) {   super(img);  }   public void left() {  this.setX(this.getX() - MapPane.CELL\_SIZE);  }   public void right() {  this.setX(this.getX() + MapPane.CELL\_SIZE);  }   public void up() {  this.setY(this.getY() - MapPane.CELL\_SIZE);  }   public void down() {  this.setY(this.getY() + MapPane.CELL\_SIZE);  } }  7.  import com.sun.glass.ui.Screen; import javafx.scene.image.Image; import javafx.scene.image.ImageView; import javafx.scene.layout.\*;  import java.io.File; import java.io.IOException; import java.util.ArrayList; import java.util.List; import java.util.Scanner;  public class MapPane extends Pane {  //大小   public final static int *CELL\_SIZE* = 64;  // 图像陈列   private Image[] icons;  // 所有的逻辑存储空间  private ImageView[][] map;  // 控制   private Man man;    private List<ImageView> targets;  // 数量  private int xlength;  private int ylength;    // 建   public MapPane(Image[] iconList, File mapFile) {  icons = iconList;  // 载入   this.loadMap(mapFile);  // 设置背景  Background bg = new Background(  new BackgroundImage(  iconList[Element.BACKGROUND],  BackgroundRepeat.REPEAT,  BackgroundRepeat.REPEAT,  BackgroundPosition.DEFAULT,  new BackgroundSize(  BackgroundSize.AUTO,  BackgroundSize.AUTO,  true, true, false, false)  )  );  setBackground(bg);  }   // 更新数据   public void loadMap(File mapFile) {  try (Scanner input = new Scanner(mapFile)) {  // 获取长度  String[] items = input.nextLine().split(",");  int xlen = Integer.*parseInt*(items[0]);  int ylen = Integer.*parseInt*(items[1]);  xlength = xlen;  ylength = ylen;  // 初始化与清楚   this.map = new ImageView[xlen][ylen];  targets = new ArrayList<ImageView>();  this.getChildren().clear();  // 读文件，加载元素   while (input.hasNextLine()) {  // 读信息   items = input.nextLine().split(",");  int x = Integer.*parseInt*(items[0]);  int y = Integer.*parseInt*(items[1]);  int type = Integer.*parseInt*(items[2]);  // 创建元素   Element e = null;  switch (type) {  case Element.MAN:  e = this.man = new Man(icons[Element.MAN]);  break;  case Element.BOX:  e = new Box(icons[Element.BOX]);  break;  case Element.WALL:  e = new Wall(icons[Element.WALL]);  break;  case Element.TARGET:  e = new Target(icons[Element.TARGET]);  break;  default:  break;  }  // 设置位置，展示元素   e.setX(x \* *CELL\_SIZE*);  e.setY(y \* *CELL\_SIZE*);  getChildren().add(e);  // 所有元素的  if (e instanceof Target) {  targets.add(e);  } else {  this.map[x][y] = e;  }  }  } catch (IOException e) {  e.printStackTrace();  }  }   // 判断状态   public boolean judge() {  boolean win = true;  ImageView img;  for (ImageView target : targets) {  img = target;  int x = (int) (img.getX() / *CELL\_SIZE*);  int y = (int) (img.getY() / *CELL\_SIZE*);  if (!(map[x][y] instanceof Box)) {   win = false;  break;  }  }  return win;  }   // 移动方向   public void moveManUp() {  int manX = (int) (man.getX() / *CELL\_SIZE*);  int manY = (int) (man.getY() / *CELL\_SIZE*);  if (manY > 0) {  if (map[manX][manY - 1] == null) {  man.up();  map[manX][manY] = null;  map[manX][manY - 1] = man;  } else if (map[manX][manY - 1] instanceof Box) {  System.*out*.println("hahah");  if (manY - 1 > 0 && map[manX][manY - 2] == null) {  Box b = (Box) map[manX][manY - 1];  b.up();  map[manX][manY - 2] = b;  man.up();  map[manX][manY] = null;  map[manX][manY - 1] = man;  }  }  }  }   public void moveManDown() {  int manX = (int) (man.getX() / *CELL\_SIZE*);  int manY = (int) (man.getY() / *CELL\_SIZE*);  if (manY < ylength - 1) {  if (map[manX][manY + 1] == null) {  man.down();  map[manX][manY] = null;  map[manX][manY + 1] = man;  } else if (map[manX][manY + 1] instanceof Box) {  System.*out*.println("hahah");  if (manY + 2 < ylength && map[manX][manY + 2] == null) {  Box b = (Box) map[manX][manY + 1];  b.down();  map[manX][manY + 2] = b;  man.down();  map[manX][manY] = null;  map[manX][manY + 1] = man;  }  }  }  }   // 左方向   public void moveManLeft() {  int manX = (int) (man.getX() / *CELL\_SIZE*);  int manY = (int) (man.getY() / *CELL\_SIZE*);  if (manX > 0) {  if (map[manX - 1][manY] == null) {  man.left();  map[manX][manY] = null;  map[manX - 1][manY] = man;  } else if (map[manX - 1][manY] instanceof Box) {  System.*out*.println("hahah");  if (manX - 1 > 0 && map[manX - 2][manY] == null) {  Box b = (Box) map[manX - 1][manY];  b.left();  map[manX - 2][manY] = b;  man.left();  map[manX][manY] = null;  map[manX - 1][manY] = man;  }  }  }  }   // 右方向   public void moveManRight() {  int manX = (int) (man.getX() / *CELL\_SIZE*);  int manY = (int) (man.getY() / *CELL\_SIZE*);  if (manX < xlength - 1) {  if (map[manX + 1][manY] == null) {  man.right();  map[manX][manY] = null;  map[manX + 1][manY] = man;  } else if (map[manX + 1][manY] instanceof Box) {  System.*out*.println("hahah");  if (manX + 2 < xlength && map[manX + 2][manY] == null) {  Box b = (Box) map[manX + 1][manY];  b.right();  map[manX + 2][manY] = b;  man.right();  map[manX][manY] = null;  map[manX + 1][manY] = man;  }  }  }  } }  8.  import javafx.application.Application; import javafx.scene.Scene; import javafx.scene.control.Alert; import javafx.scene.control.Button; import javafx.scene.control.Label; import javafx.scene.control.ToolBar; import javafx.scene.image.Image; import javafx.scene.layout.VBox; import javafx.stage.Stage;  import java.io.File; import java.util.ArrayList; import java.util.List;  public class Sokoban extends Application {  // 文件路径   private final String mapDir = "maps";  private final String imgDir = "imgs";  // 其文建  private List<File> mapFiles;  // 当前水品   private int currentLevel = 0;    private MapPane currentMap = null;     public void loadMapFiles() {  mapFiles = new ArrayList<File>();  File dir = new File(mapDir);  for (File f : dir.listFiles()) {  mapFiles.add(f);  }  }   public void start(Stage primaryStage) throws Exception {  loadMapFiles();  VBox vb = new VBox();  ToolBar tb = new ToolBar();  Label label = new Label("Current Level:" + currentLevel);  // 按钮   Button preBtn = new Button("Previous");  preBtn.setOnAction(e -> {  if (currentLevel > 0) {  currentLevel--;  currentMap.loadMap(mapFiles.get(currentLevel));  label.setText("Current Level:" + currentLevel);  primaryStage.sizeToScene();  }  });  // 按钮事件  Button nextBtn = new Button("Next");  nextBtn.setOnAction(e -> {  if (currentLevel < 3) {  currentLevel++;  currentMap.loadMap(mapFiles.get(currentLevel));  label.setText("Current Level:" + currentLevel);  primaryStage.sizeToScene();  }  });  // 按钮重置事件   Button resetBtn = new Button("Reset");  resetBtn.setOnAction(e -> {  currentLevel = 0;  currentMap.loadMap(mapFiles.get(currentLevel));  label.setText("Current Level:" + currentLevel);  primaryStage.sizeToScene();  });  tb.getItems().addAll(label, preBtn, nextBtn, resetBtn);    Alert a = new Alert(Alert.AlertType.CONFIRMATION);  a.setHeaderText("You have won");  a.setContentText("Next level?");  a.setOnCloseRequest(e -> {  if (currentLevel < mapFiles.size() - 1) {  currentLevel++;  currentMap.loadMap(mapFiles.get(currentLevel));  label.setText("Current Level:" + currentLevel);  primaryStage.sizeToScene();  }  });    Image[] icons = new Image[5];  icons[Element.MAN] = new Image("file:" + imgDir + "/man.png");  icons[Element.BOX] = new Image("file:" + imgDir + "/box.png");  icons[Element.WALL] = new Image("file:" + imgDir + "/wall.png");  icons[Element.TARGET] = new Image("file:" + imgDir + "/target.png");  icons[Element.BACKGROUND] = new Image("file:" + imgDir + "/floor.png");  currentMap = new MapPane(icons, mapFiles.get(currentLevel));  vb.getChildren().addAll(tb, currentMap);  Scene scene = new Scene(vb);  scene.setOnKeyPressed(e -> {  switch (e.getCode()) {  case A:  currentMap.moveManLeft();  break;  case D:  currentMap.moveManRight();  break;  case W:  currentMap.moveManUp();  break;  case S:  currentMap.moveManDown();  break;  default:  break;  }  if (currentMap.judge()) {  a.show();  }  });  primaryStage.sizeToScene();  primaryStage.setResizable(false);  primaryStage.setScene(scene);  primaryStage.getIcons().add(new Image("file:" + imgDir + "/man.png"));  primaryStage.setTitle("Sokoban");  primaryStage.show();  }   public static void main(String[] args) {  launch(args);  } } | | | |

|  |  |  |
| --- | --- | --- |
| Evaluation | Code Correctness (60%): |  |
| Experience (40%): |  |
| Score： | |