飞机大战小游戏实验报告

[一、 实验目的 2](#_Toc44276895)

[二、 实验环境 2](#_Toc44276896)

[三、 实现功能 2](#_Toc44276897)

[四、 实验内容 3](#_Toc44276898)

[1. 开始模块 3](#_Toc44276899)

[2. 发射子弹 6](#_Toc44276900)

[3. 积分模块 9](#_Toc44276901)

[4. 游戏操作： 12](#_Toc44276904)

[5. 防触碰： 14](#_Toc44276904)

飞机大战小游戏实验报告

## 一、实验目的

了解如何将程序导入Linux系统中，并在Linux环境下运行，熟悉Linux系统操作环境。

利用eclipse实现小游戏运行。

## 二、实验环境

Linux虚拟机

Java配置：Jdk

开发工具：eclipse

## 三、功能实现

飞机大战主要需要我方飞机和敌方飞机，还有子弹，特殊nPC，开始背景，结束背景，以及背景音乐。我方飞机可以随意移动，敌方飞机无规律出现。游戏玩家通过鼠标移动控制飞机移动，我方飞机在游戏开始时就一直能发射子弹，当我方子弹碰到敌方飞机时，敌方飞机消失。当玩家飞机碰到敌方子弹时，生命值减一，直到玩家飞机生命值为一时，游戏结束。

## 四、实验内容

### 1.开始模块





private int clearCompletedRows(TetrisPiece piece) {

TetrisBoard board = this.getBoard();

for(int i = 0; i < TetrisConstants.FOUR\_BLOCKS; i++) {

int rowY = piece.getBlockY(i);

if(board.checkRowCompleted(rowY)) {

this.markRowCompleted(rowY, true);

}

}

int numClearedRows = 0;

for(int y = TetrisConstants.HEIGHT - 1; y >= 0; y--) {

if(numClearedRows > 0) {

board.dropRow(y, numClearedRows);

}

if(this.isRowCompleted(y)) {

numClearedRows++;

this.markRowCompleted(y, false);

}

}

for(int i = 0; i < numClearedRows; i++) {

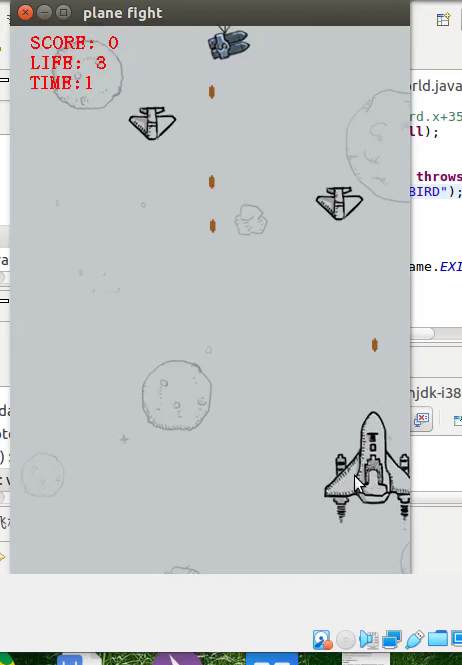
board.clearRow(i);

}

return numClearedRows;

}

### 2.发射子弹



public void rotate(int pivotX, int pivotY, boolean rotateDirection) {

if(TetrisConstants.ROTATION\_TYPE\_TOGGLE == this.rotationType) {

// 如果翻转类型为TOGGLE

rotateDirection = this.rotationToggle; //判断翻转方向

this.rotationToggle = !this.rotationToggle; // 如果与canRotate匹配，

中心点方块将被使用

}

// 重构四个小块，每个都转动

for(int i = 0; i < TetrisConstants.FOUR\_BLOCKS; i++) {

int blockX = this.getBlockX(i);

int blockY = this.getBlockY(i);

// 左旋转：交换x和y坐标，x坐标取反

// 右旋转：交换x和以坐标，y坐标取反

int dx = blockY - pivotY;

int dy = blockX - pivotX;

if(rotateDirection) {

// 如向左旋转

dx \*= -1;

} else {

// 如向右旋转

dy \*= -1;

}

int rotateX = pivotX + dx;

int rotateY = pivotY + dy;

this.setBlockCoords(i, rotateX, rotateY);//得到翻转后的新坐标

}

}

public boolean checkRowCompleted(int rowY) {

for(int x = 0; x < TetrisConstants.WIDTH; x++) {

if(TetrisConstants.BLOCK\_EMPTY == this.getBlockType(x, rowY)) {

// 选中目标，发射子弹

return false;

}

}

return true;

}

private void init() {

this.board = new TetrisBoard();

this.gameCanvas = new TetrisCanvas(this);

this.activePiece = new TetrisPiece();

this.completedRows = new boolean[TetrisConstants.HEIGHT]; //初始化数组

this.hiScore = this.openAndReadHiScore();

this.nextPieceType = TetrisConstants.UNINITIALIZED;

this.rand = new Random();

// 设置exit/pause/resume命令

this.setuPCommands();

this.gameCanvas.addCommand(this.exitCommand);

// 显示title屏

this.setGameState(TetrisConstants.TITLE\_STATE);

}

### 3.积分模块

public void drawCount(Graphics g){

g.setColor(Color.white);

Font font = new Font("宋体", 1, 20);

g.setFont(font);

g.drawString("积分:" + this.score, 0, 470);

}

实现积分的增长：

if (burst != null)

burst.draw(g\_off);

for (int i = 0; i < list.size(); i++) {

Role chara1 = (Role) list.get(i);

if (chara1.isDead()) {

if (chara1 instanceof Enemy)

{

chara1.drawBurst(g\_off);

this.battle.score++;

}

list.remove(i);

}

}

全屏爆炸时，计算炸死几个敌人增长分数：

if(GamePanel.*skillCount*>0){

for (int i = 0; i < GamePanel.*list*.size(); i++) {

Role chara1 = (Role) GamePanel.*list*.get(i);

if(!(chara1 instanceof Battle) && chara1.x>0 && chara1.y>0

&& !(chara1 instanceof BossA) && !(chara1 instanceof BossB)

&& !(chara1 instanceof BossC)){

GamePanel.*list*.remove(i);

if(chara1 instanceof Enemy){

this.score++;

}

实现积分的输出：

private void gameOver() {

GamePanel.*skillCount* = 10;

if (Key.*enter*) {

*gameMode* = 12;

stage = 1;

} else {

g\_off.setColor(Color.*white*);

Font font = new Font("黑体", 1, 28);

g\_off.setFont(font);

FontMetrics fontMetrics = getFontMetrics(font);

g\_off.drawString("Game Over", (450 - fontMetrics

.stringWidth("Game Over")) / 2, (500 + fontMetrics

.getHeight()) / 2 - 50);

String score = "Score: " + this.battle.score;

g\_off.drawString(score, (450 - fontMetrics

.stringWidth(score)) / 2, (500 + fontMetrics

.getHeight()) / 2 - 20);

if (15 <= current % 50)

g\_off.drawString("请按 回车", (450 - fontMetrics

.stringWidth("请按 回车")) / 2,

(500 + fontMetrics.getHeight()) / 2 + 100);

}

}

### 4.游戏操作

public void move() {

oldx = x;

oldy = y;

if (Key.*left*) {

if (Key.*xkey*)

x -= (double) speed / 4D;

else

x -= speed;

if (x <= 0.0F)

x = 0.0F;

}

if (Key.*right*) {

if (Key.*xkey*)

x += (double) speed / 4D;

else

x += speed;

if (x + WIDTH >= (float) *app*.getWidth())

x = (float) *app*.getWidth() - WIDTH;

}

if (Key.*down*) {

if (Key.*xkey*)

y += (double) speed / 4D;

else

y += speed;

if (y + HEIGHT >= (float) *app*.getHeight())

y = (float) *app*.getHeight() - HEIGHT;

}

if (Key.up) {

if (Key.*xkey*)

y -= (double) speed / 4D;

else

y -= speed;

if (y <= 0.0F)

y = 0.0F;

}

### 5.防触碰



public boolean checkHit(Role chara) {

if ((chara instanceof EnemyA) || (chara instanceof EnemyB)|| (chara

instanceof EnemyC) || (chara instanceof EnemyShot)) {

if ((x + WIDTH) - 14F > chara.x && x + 14F < chara.x +

chara.WIDTH

&& (y + HEIGHT) - 12F > chara.y

&& y + 12F < chara.y + chara.HEIGHT) {

//如果碰到敌人,敌人死亡

chara.dead();

//如果碰到子弹血量减少

if (chara instanceof EnemyBeam){

power--;

}

power -= 50;

if (power <= 0) {

dead();

//绘制爆炸图片

GamePanel.*burst* = new Burst(x, y);

}

return true;

}

## 五、实验总结：

通过这次Linux 大作业程序编写，我了解到Linux环境下如何运行代码，熟悉了Linux系统的环境，也发现了很多错误，学到了很多东西，感谢老师一学期的知识讲授。