Java Project report

 ——打砖块

15107018张浩楠 17103012天敏如

17103004金泽龙 17103014文善姬

Project report

Project——打砖块

简介：打砖块游戏是利用小球将一个个砖块打没的游戏。

研究内容：研究分为四个部分。

1. 游戏的开始键和横板移动命令
2. 砖块位置的设定和填充颜色
3. 小球的移动指令和速度的确定
4. 判断小球的反弹和当砖块全打没时的初始化重新开始

第一部分的研究：

privateJLabellabel;定义一个标签。

label=newJLabel("按空格键开始")。

内容为空格键开始，addKeyListener(this);定义一个键盘监听器。

**super**("打砖块");//\*

Container pane=getContentPane();

//设置空白面板容器

label=**new** JLabel("按空格键开始");

//标签

label.setHorizontalAlignment(JLabel.***CENTER***);

//水平

label.setVerticalAlignment(JLabel.***BOTTOM***);

//垂直

pane.add(label);

**if** (e.getKeyCode() ==e.***VK\_SPACE***) {

**if**(renew){

greenBallThread=**new** BallThread(**this**);

bouncing = **true**;//可以打掉砖块

greenBallThread.start();

label.setVisible(**false**);//使标签变得不可见

}

renew=**false**;//按空格不可重新开始

}

移动命令：方向的左键与右键

同开始命令原理，如果键入左键，横版向左移动20个单位（Rx为横板坐标），如果小球还在运动，当横板移到最左侧（Rx=0），不能再向左移动，则横板靠在最左侧（Rx=0），键入右键也是如此。

**if**(e.getKeyCode()==e.***VK\_LEFT***){

Rx=Rx-20;

**if**(bouncing){

**if**(Rx<0){

Rx=0;

}

} **else**{

**if**(Rx<0){

Rx=0;

} **else**{

x=x-20;

ball.ball\_x=x;

}

}

repaint();

}

**if**(e.getKeyCode()==e.***VK\_RIGHT***){

Rx=Rx+20;

**if**(bouncing){

**if**(Rx+80>600){

Rx=520;

}

} **else**{

**if**(Rx+80>600){

Rx=520;

} **else**{

x=x+20;

ball.ball\_x=x;

}

}

repaint();

}

}

第二部分的研究：

brick[0]=newBrick(0,60,50,20);设置砖块坐标，ball=newBall(150,450,10,10);和小球的坐标

if(brick[i].visible==true)判断砖块存在，用switch语句，逐个对砖块填充颜色，最后四行代码是分别对小球和横板颜色坐标的定义。

brick[1]=**new** Brick(50,60,50,20);

brick[2]=**new** Brick(100,60,50,20);

brick[3]=**new** Brick(150,60,50,20);

brick[4]=**new** Brick(200,60,50,20);

brick[5]=**new** Brick(250,60,50,20);

brick[6]=**new** Brick(0,90,50,20);

.............

**for** (**int** i=0;i<=35;i++){

**if**(brick[i].visible==**true**){

**switch**(i){

**case** 0 :g.setColor(Color.***blue***);

**break**;

**case** 1 :g.setColor(Color.***cyan***);

**break**;

**case** 2 :g.setColor(Color.***gray***);

**break**;

**case** 3 :g.setColor(Color.***green***);

**break**;

**case** 4 :g.setColor(Color.***magenta***);

**break**;

**case** 5 :g.setColor(Color.***yellow***);

**break**;

**case** 6 :g.setColor(Color.***white***);

**break**;

**case** 7 :g.setColor(Color.***black***);

**break**;

............

}

第三部分的研究：

先定义小球的速度。

**while**(**true**){

**try**{

Thread.*currentThread*().*sleep*(50);//定义小球速度

}

**catch**(InterruptedException exception){

System.***err***.println(exception.toString());//在屏幕上打印

}//打印小球的动作

判断小球坐标和横板坐标，当小球落在横板坐标之内，小球反弹，小球横坐标和纵坐标都以一个随机值改变后运动。

**if**(x+5>Rx&&x+5<Rx+80&&y+10>=Ry){//小球在横板的坐标的范围内

yUp=**false**;

xDx=(**int**)(Math.*random*()\*5+2);

//小球坐标增量

yDy=(**int**)(Math.*random*()\*5+2);

}

第四部分的研究：

判断小球横坐标如果在增加，小球横坐标=小球原横坐标+小球横坐标增量，否则小球横坐标=小球原横坐标-小球横坐标增量；纵坐标同理。打到左右墙壁时也同理。

**if**(y<=0){

y=0;

ball.ball\_y=y;

yUp=**true**;

xDx=(**int**)(Math.*random*()\*5+2);

yDy=(**int**)(Math.*random*()\*5+2);

当小球落下去时则重新玩。

**else** **if**(y>=MAX\_Y-15){

//yDy=(int)(Math.random()\*5+2);

yUp=**false**;

**break**;

}

如果所有砖块都不可见，则重新玩。

**for** (i=0;i<=35&&brick[i].visible==**false**;i++){//所有的球都不可见的时候

//则重新玩

}

**if**(i==36){

**break**;

}

//

}

最后的重新开始初始化。

renew=**true**;

//初始化

bouncing=**false**;

**for** (**int** i=0;i<=35;i++){

brick[i].visible=**true**;

}

xUp=**true**;

yUp=**false**;

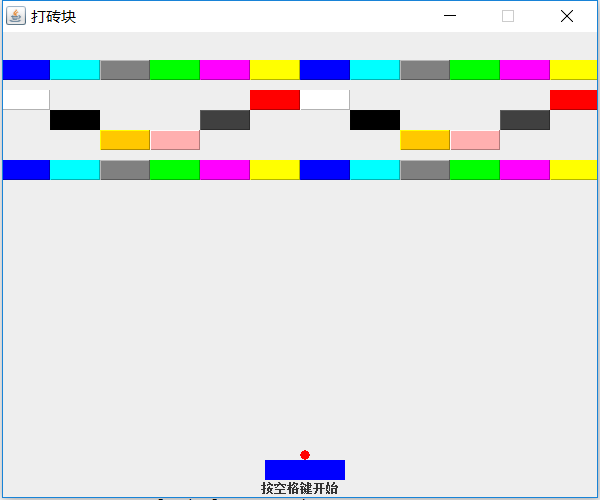
xDx=2;

yDy=2;

x=300;

y=450;

运行结果



总结

通过这次的课程设计，它使我们更加深刻的体会到Java语言程序编写过程的严谨与细致，还有无论做任何事情都要有耐心。本次编程使我们进一步巩固了Java语言的基本知识，提高了编程能力和分析实际问题解决实际问题的能力，体会到了团结互助的精神。

参考文献：

线程的调用

<https://blog.csdn.net/jiesa/article/details/72845441>

随机数

<https://blog.csdn.net/qq_36868342/article/details/73478112>

键盘监听器

<https://www.cnblogs.com/KeenLeung/archive/2012/05/27/2520657.html>

class BallThread extends Thread线程的实现方法

<https://www.cnblogs.com/kevinfuture/p/4281623.html>

创建空白容器 Container pane=getContentPane();

<https://blog.csdn.net/xuefeideren/article/details/41174677>

如何实现画面可视化

<https://blog.csdn.net/qq_34228570/article/details/72774088>

改变组件位置setLocation(int x, int y)：

<https://blog.csdn.net/u011235543/article/details/51892500/>

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import javax.swing.event.\*;

class BallThread extends Thread{

private HitBrick m;

BallThread(HitBrick a){

//super();

m=a;

}

public void run(){

m.move();

m.repaint();

}

}//小球的运动轨迹

class Brick{

Rectangle rect=null;//创建一个空白界面

//长方形对象，砖块按钮的位置和宽高

int brick\_x,brick\_y;

//按扭的左上角坐标

int brick\_width,brick\_height;

//按扭的宽和高

Boolean visible;//\*

public Brick(int x,int y,int w,int h)

{

brick\_x=x;

brick\_y=y;

brick\_width=w;

brick\_height=h;

visible=true;

rect=new Rectangle(x,y,w,h);

//创建长方形对象---砖块按钮的位置和宽高。

}

}

class Ball{

Rectangle rect=null;

int ball\_x,ball\_y;

int ball\_width,ball\_height;

public Ball(int x,int y,int w,int h){

ball\_x=x;

ball\_y=y;

ball\_width=w;

ball\_height=h;

rect=new Rectangle(x,y,w,h);

}

}

public class HitBrick extends JFrame implements KeyListener{

private BallThread greenBallThread;

//控制小球的线程

private Boolean xUp,yUp,bouncing;

private int x,y,xDx,yDy;

//小球坐标,增量

private final int MAX\_X=600,MAX\_Y=500;

private Boolean renew;

private JLabel label;

private int Rx,Ry;

//横板坐标

private Brick brick[]=new Brick[36];

//砖块

private Ball ball;

//小球

public HitBrick(){

super("打砖块");//\*

Container pane=getContentPane();

//设置空白面板容器

label=new JLabel("按空格键开始");

//标签

label.setHorizontalAlignment(JLabel.CENTER);

//水平

label.setVerticalAlignment(JLabel.BOTTOM);

//垂直

pane.add(label);

//向面板里添加标签

xUp=true;

//横坐标可以移动

yUp=false;

//纵坐标不可以移动

xDx=2;

yDy=2;

x=300;

//小球坐标

y=450;

Rx=265;

//横板坐标

Ry=460;

renew=true;

bouncing=false;//\*

addKeyListener(this);

//键盘监听器

brick[0]=new Brick(0,60,50,20);

//砖块坐标

brick[1]=new Brick(50,60,50,20);

brick[2]=new Brick(100,60,50,20);

brick[3]=new Brick(150,60,50,20);

brick[4]=new Brick(200,60,50,20);

brick[5]=new Brick(250,60,50,20);

brick[6]=new Brick(0,90,50,20);

brick[7]=new Brick(50,110,50,20);

brick[8]=new Brick(100,130,50,20);

brick[9]=new Brick(150,130,50,20);

brick[10]=new Brick(200,110,50,20);

brick[11]=new Brick(250,90,50,20);

brick[12]=new Brick(0,160,50,20);

brick[13]=new Brick(50,160,50,20);

brick[14]=new Brick(100,160,50,20);

brick[15]=new Brick(150,160,50,20);

brick[16]=new Brick(200,160,50,20);

brick[17]=new Brick(250,160,50,20);

brick[18]=new Brick(300,60,50,20);

brick[19]=new Brick(350,60,50,20);

brick[20]=new Brick(400,60,50,20);

brick[21]=new Brick(450,60,50,20);

brick[22]=new Brick(500,60,50,20);

brick[23]=new Brick(550,60,50,20);

brick[24]=new Brick(300,90,50,20);

brick[25]=new Brick(350,110,50,20);

brick[26]=new Brick(400,130,50,20);

brick[27]=new Brick(450,130,50,20);

brick[28]=new Brick(500,110,50,20);

brick[29]=new Brick(550,90,50,20);

brick[30]=new Brick(300,160,50,20);

brick[31]=new Brick(350,160,50,20);

brick[32]=new Brick(400,160,50,20);

brick[33]=new Brick(450,160,50,20);

brick[34]=new Brick(500,160,50,20);

brick[35]=new Brick(550,160,50,20);

ball=new Ball(300,450,10,10);

//球的坐标

setSize(MAX\_X,MAX\_Y);

//窗口大小

setResizable(false);//窗口的大小不可调

setVisible( true );//让程序变成可视化

//可视化

}

public void keyPressed(KeyEvent e) {

if (e.getKeyCode() ==e.VK\_SPACE) {

if(renew){

greenBallThread=new BallThread(this);

bouncing = true;//可以打掉砖块

greenBallThread.start();

label.setVisible(false);//使标签变得不可见

}

renew=false;//按空格不可重新开始

}

if(e.getKeyCode()==e.VK\_LEFT){

Rx=Rx-20;

if(bouncing){

if(Rx<0){

Rx=0;

}

} else{

if(Rx<0){

Rx=0;

} else{

x=x-20;

ball.ball\_x=x;

}

}

repaint();

}

if(e.getKeyCode()==e.VK\_RIGHT){

Rx=Rx+20;

if(bouncing){

if(Rx+80>600){

Rx=520;

}

} else{

if(Rx+80>600){

Rx=520;

} else{

x=x+20;

ball.ball\_x=x;

}

}

repaint();

}

}

public void keyReleased (KeyEvent e) {

}

public void keyTyped (KeyEvent e){

}

public void paint(Graphics g){

super.paint(g);

ball.rect.setLocation(x,y);

if(bouncing){

for (int i=0;i<=35;i++){

if(brick[i].visible==true){

switch(i){

case 0 :g.setColor(Color.blue);

break;

case 1 :g.setColor(Color.cyan);

break;

case 2 :g.setColor(Color.gray);

break;

case 3 :g.setColor(Color.green);

break;

case 4 :g.setColor(Color.magenta);

break;

case 5 :g.setColor(Color.yellow);

break;

case 6 :g.setColor(Color.white);

break;

case 7 :g.setColor(Color.black);

break;

case 8 :g.setColor(Color.orange);

break;

case 9 :g.setColor(Color.pink);

break;

case 10 :g.setColor(Color.darkGray);

break;

case 11 :g.setColor(Color.red);

break;

case 12 :g.setColor(Color.blue);

break;

case 13 :g.setColor(Color.cyan);

break;

case 14 :g.setColor(Color.gray);

break;

case 15 :g.setColor(Color.green);

break;

case 16 :g.setColor(Color.magenta);

break;

case 17 :g.setColor(Color.yellow);

break;

case 18 :g.setColor(Color.blue);

break;

case 19 :g.setColor(Color.cyan);

break;

case 20 :g.setColor(Color.gray);

break;

case 21 :g.setColor(Color.green);

break;

case 22 :g.setColor(Color.magenta);

break;

case 23 :g.setColor(Color.yellow);

break;

case 24 :g.setColor(Color.white);

break;

case 25 :g.setColor(Color.black);

break;

case 26 :g.setColor(Color.orange);

break;

case 27 :g.setColor(Color.pink);

break;

case 28 :g.setColor(Color.darkGray);

break;

case 29 :g.setColor(Color.red);

break;

case 30 :g.setColor(Color.blue);

break;

case 31 :g.setColor(Color.cyan);

break;

case 32 :g.setColor(Color.gray);

break;

case 33 :g.setColor(Color.green);

break;

case 34 :g.setColor(Color.magenta);

break;

case 35 :g.setColor(Color.yellow);

break;

}

g.fill3DRect(brick[i].brick\_x,brick[i].brick\_y,brick[i].brick\_width,brick[i].brick\_height,true);//利用3d画出砖块

}

}

g.setColor(Color.red);

g.fillOval(x, y, 10, 10);

g.setColor(Color.blue);

g.fillRect(Rx,Ry,80,20);

} else{

for (int i=0;i<=35;i++){

switch(i){

case 0 :g.setColor(Color.blue);

break;

case 1 :g.setColor(Color.cyan);

break;

case 2 :g.setColor(Color.gray);

break;

case 3 :g.setColor(Color.green);

break;

case 4 :g.setColor(Color.magenta);

break;

case 5 :g.setColor(Color.yellow);

break;

case 6 :g.setColor(Color.white);

break;

case 7 :g.setColor(Color.black);

break;

case 8 :g.setColor(Color.orange);

break;

case 9 :g.setColor(Color.pink);

break;

case 10 :g.setColor(Color.darkGray);

break;

case 11 :g.setColor(Color.red);

break;

case 12 :g.setColor(Color.blue);

break;

case 13 :g.setColor(Color.cyan);

break;

case 14 :g.setColor(Color.gray);

break;

case 15 :g.setColor(Color.green);

break;

case 16 :g.setColor(Color.magenta);

break;

case 17 :g.setColor(Color.yellow);

break;

case 18 :g.setColor(Color.blue);

break;

case 19 :g.setColor(Color.cyan);

break;

case 20 :g.setColor(Color.gray);

break;

case 21 :g.setColor(Color.green);

break;

case 22 :g.setColor(Color.magenta);

break;

case 23 :g.setColor(Color.yellow);

break;

case 24 :g.setColor(Color.white);

break;

case 25 :g.setColor(Color.black);

break;

case 26 :g.setColor(Color.orange);

break;

case 27 :g.setColor(Color.pink);

break;

case 28 :g.setColor(Color.darkGray);

break;

case 29 :g.setColor(Color.red);

break;

case 30 :g.setColor(Color.blue);

break;

case 31 :g.setColor(Color.cyan);

break;

case 32 :g.setColor(Color.gray);

break;

case 33 :g.setColor(Color.green);

break;

case 34 :g.setColor(Color.magenta);

break;

case 35 :g.setColor(Color.yellow);

break;

}

g.fill3DRect(brick[i].brick\_x,brick[i].brick\_y,brick[i].brick\_width,brick[i].brick\_height,true);

}

g.setColor(Color.red);

g.fillOval(x, y, 10, 10);

g.setColor(Color.blue);

g.fillRect(Rx, Ry, 80, 20);

}

}

public void move(){

while(true){

try{

Thread.currentThread().sleep(25);//定义小球速度

}

catch(InterruptedException exception){

System.err.println(exception.toString());//在屏幕上打印

}//打印小球的动作

for (int i=0;i<=35;i++){

if(ball.rect.intersects(brick[i].rect)&&brick[i].visible){//如果球碰到可见的砖块时

brick[i].visible=false;//砖块不可见

yUp=!yUp;

//打到球不可见

}

}

if(x+5>Rx&&x+5<Rx+80&&y+10>=Ry){//小球在横板的坐标的范围内

yUp=false;

xDx=(int)(Math.random()\*5+2);

//小球坐标增量

yDy=(int)(Math.random()\*5+2);

}

if(xUp==true){

x+=xDx;

//小球左右移动坐标改变

} else{

x-=xDx;

}

if(yUp==true){

y+=yDy;

} else{

y-=yDy;

}

if(y<=0){

y=0;

ball.ball\_y=y;

yUp=true;

xDx=(int)(2+Math.random()\*5);

yDy=(int)(2+Math.random()\*5);

} else if(y>=MAX\_Y-15){

//yDy=(int)(2+Math.random()\*5);

yUp=false;

break;

}

if(x<=0){

x=0;

ball.ball\_x=x;

xUp=true;

xDx=(int)(2+Math.random()\*5);

yDy=(int)(2+Math.random()\*5);

} else if(x>=MAX\_X-10){

x=MAX\_X-10;

ball.ball\_x=x;

xDx=(int)(2+Math.random()\*5);

yDy=(int)(2+Math.random()\*5);

xUp=false;

}//在两墙反弹

ball.rect.setLocation(ball.ball\_x,ball.ball\_y);//球的实时位置

repaint();//刷新

int i;

//如果所有砖块都不可见

for (i=0;i<=35&&brick[i].visible==false;i++){//所有的球都不可见的时候

//则重新玩

}

if(i==36){

break;

}

//

}

renew=true;

//初始化

bouncing=false;

for (int i=0;i<=35;i++){

brick[i].visible=true;

}

xUp=true;

yUp=false;

xDx=2;

yDy=2;

x=300;

y=450;

Rx=265;

Ry=460;

//

repaint();

repaint();

label.setVisible(true);

}

public static void main(String[] args) {

HitBrick mar=new HitBrick();

}

}