package game\_shooting;

//インポート////////////////////////////////////////////////////////

import java.awt.Color;

import java.awt.Graphics;

import java.awt.Graphics2D;

import java.awt.Image;

import java.awt.Rectangle;

import java.awt.Toolkit;

import java.awt.event.KeyAdapter;

import java.awt.event.KeyEvent;

import java.net.URL;

import javax.swing.JFrame;

///////////////////////////////////////////////////////////////////

//メインクラス//////////////////////////////////////////////////////

public class game\_shooting extends JFrame {

//それぞれのサイズ

private final static int **FRAME\_SIZE**=600;

private final static int **PLAYER\_SIZE**=20;

private final static int **BEEM\_SIZE**=4;

///////////////////////////////////////////////////////////

private static int **drow\_beem\_triger**=0;

private static int[] **MONSTER\_XR**= {60,180,300,420,540};

private static int[] **MONSTER\_XL**= {20,140,260,380,500};

private static int **MONSTER\_Y**=50;

private static int **MONSTER\_COUNT**=0;

private static int[] **MONSTER\_DEATH**= {0,0,0,0,0};

static double **BEEM\_PLAYER\_X**=0;

private static int **START**=1;

private static int[] **END**=new int[5];

private static int **FINISH**=0;

private static int **GAMEOVER**=0;

private static int[] **BAN\_STOP**= {0,0,0,0,0};

private static int **BAN\_X**=0,**BAN\_Y**=0,**PRINT\_BAN**=0;

///////////////////////////////////////////////////////////

//画像ファイルパス取得////////////////////////////////////////

URL URL1=this.getClass().getResource("back-ground.jpg");

URL URL2=this.getClass().getResource("start.jpg");

URL URL3=this.getClass().getResource("ban.png");

URL URL4=this.getClass().getResource("gameover.png");

URL URL5=this.getClass().getResource("finish.png");

URL URL6=this.getClass().getResource("monster.jpg");

//////////////////////////////////////////////////////////

//画像を変数に格納//////////////////////////////////////////////////

Image BACKGROUND\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL1);

Image START\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL2);

Image BAN\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL3);

Image GAMEOVER\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL4);

Image FINISH\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL5);

Image MONSTER\_IMG = Toolkit.**getDefaultToolkit**().getImage(URL6);

////////////////////////////////////////////////////////////////

//////////図形描画//////////////////////////////////////////////

Rectangle player = new Rectangle(**FRAME\_SIZE**/2, 570,

**PLAYER\_SIZE**\*2, **PLAYER\_SIZE**);//時期のボディ部分（横長）

Rectangle player2 = new Rectangle((int)player.getX()+12, 554,

**PLAYER\_SIZE**-4, **PLAYER\_SIZE**-4);//自機の砲撃筒部分

Rectangle beem = new Rectangle((int)player.getX()+18, 554,

**BEEM\_SIZE**, **BEEM\_SIZE**);//ビームの四角形描画

///////////////////////////////////////////////////////////////

//メイン////////////////////////////////////////////////////////

public static void main(String[] args) {

// **TODO** 自動生成されたメソッド・スタブ

//コンストラクタ

new game\_shooting();

}

////////////////////////////////////////////////////////////////

//コンストラクタ//////////////////////////////////////////////////////

public game\_shooting(){

//フレームサイズでフレームセット

setSize(**FRAME\_SIZE**,**FRAME\_SIZE**);

//フレームタイトル

setTitle("シューティングゲーム");

//閉じるボタンが押された時終了

setDefaultCloseOperation(JFrame.**EXIT\_ON\_CLOSE**);

//キーの入力のリスナー

addKeyListener(new Key());

//フレーム出現

setVisible(true);

}

////////////////////////////////////////////////////////////////////

//GUI上に表示////////////////////////////////////////////////////////

public void paint(Graphics g) {

g.drawImage(getScreen(), 0, 0, this);

}

private Image getScreen() {

Image screen = createImage(**FRAME\_SIZE**, **FRAME\_SIZE**);

Graphics2D g = (Graphics2D)screen.getGraphics();

GameEnd gameend=new GameEnd();

BanTime bantime=new BanTime();

//drawImage(Imageオブジェクト, x座標, y座標, 横幅, 縦幅, 監視コンポーネント)

//背景表示

g.drawImage(BACKGROUND\_IMG, 0, 0,600,600, this);

//モンスター表示、削除

for(int i=0;i<5;i++) {

if(**MONSTER\_COUNT**==0) {

if(**MONSTER\_DEATH**[i]==0) g.drawImage(MONSTER\_IMG, **MONSTER\_XR**[i], **MONSTER\_Y**,40,40, this);

}

if(**MONSTER\_COUNT**==1) {

if(**MONSTER\_DEATH**[i]==0)g.drawImage(MONSTER\_IMG, **MONSTER\_XL**[i], **MONSTER\_Y**,40,40, this);

}

//敵が破壊された時の座標取得

if(**BAN\_STOP**[i]==1) {

**BAN\_STOP**[i]=2;

if(**MONSTER\_COUNT**==0&&**BAN\_X**==0&&**BAN\_Y**==0) {

**BAN\_X**= **MONSTER\_XR**[i];

**BAN\_Y**=**MONSTER\_Y**;

}

if(**MONSTER\_COUNT**==1&&**BAN\_X**==0&&**BAN\_Y**==0) {

**BAN\_X**=**MONSTER\_XL**[i];

**BAN\_Y**=**MONSTER\_Y**;

}

**PRINT\_BAN**=1;

}

}

//敵が破壊された時の爆発画像表示

if(**PRINT\_BAN**==1) {

g.drawImage(BAN\_IMG,**BAN\_X**,**BAN\_Y** ,40,40, this);

bantime.start();

}

// 自機描画

g.setColor(Color.**black**);//カラーセット

g.draw(player);

g.draw(player2);

// ビーム描画

if(**drow\_beem\_triger**>=1) {

g.setColor(Color.**red**);

g.draw(beem);

}

// スタート画面

if(**START**==1) {

g.drawImage(START\_IMG, 0, 0,600,600, this);

g.drawString("Enterキーでスタート！！", 230, 500);

}

// ゲーム敗北画面

if(**GAMEOVER**==1) {

g.drawImage(GAMEOVER\_IMG, 0, 0,600,600, this);

gameend.start();

}

// ゲーム勝利画面

if(**FINISH**==1) {

g.drawImage(FINISH\_IMG, 0, 0,600,600, this);

gameend.start();

}

return screen;

}

///////////////////////////////////////////////////////////////////////////

//キー入力判定クラス//////////////////////////////////////////////////////////

class Key extends KeyAdapter{

public void keyPressed(KeyEvent e) { //KeyPressedじゃないとだめ

int threadOn=0;

Beem beems = new Beem();

switch (e.getKeyCode()) {

//左キー押した時の処理

case KeyEvent.**VK\_LEFT**:

if(player.getX()>20) {

player.setLocation((int)player.getX() - 10, (int)player.getY());

player2.setLocation((int)player2.getX()-10, (int)player2.getY());

if(**drow\_beem\_triger**==0) {

beem.setLocation((int)beem.getX()-10,554);

}

}

break;

//右キー押した時の処理

case KeyEvent.**VK\_RIGHT**:

if(player.getX()<560) {

player.setLocation((int)player.getX() + 10, (int)player.getY());

player2.setLocation((int)player2.getX()+10, (int)player2.getY());

if(**drow\_beem\_triger**==0) {

beem.setLocation((int)beem.getX()+10,554);

}

}

break;

//上キー押した時の処理

case KeyEvent.**VK\_SPACE**:

**drow\_beem\_triger**+=1;

if(**drow\_beem\_triger**==1) beems.start();

break;

//Enterキーを押した時の処理

case KeyEvent.**VK\_ENTER**:

**START**=0;

threadOn=1;

break;

}

//スタート画面でエンター押された時に走らせるクラス

if(threadOn==1) {

Monster monsters=new Monster();

Hantei hantei=new Hantei();

monsters.start();

hantei.start();

Repaint re =new Repaint();

re.start();

threadOn+=1;

}

}

}

//////////////////////////////////////////////////////////////////////////////

//ビーム（攻撃）の処理・・・thread継承して並行動作///////////////////////////////////

class Beem extends Thread {

public void run() {

**BEEM\_PLAYER\_X**=player.getX()+18;

while(beem.getY()>=0) {

try {

Thread.**sleep**(2);//2マイクロ秒停止

beem.setLocation((int)**BEEM\_PLAYER\_X**, (int)beem.getY()-1);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

beem.setLocation((int)player.getX()+18,554);

**drow\_beem\_triger**=0;

}

}

/////////////////////////////////////////////////////////////////////////////

//敵の処理・・・thread継承して並行動作////////////////////////////////////////////

class Monster extends Thread{

public void run() {

int count=0;

while(true) {

try {

Thread.**sleep**(2000);

count+=1;

//モンスター左側表示

if(count%2==1) {

**MONSTER\_COUNT**=1;

}

//モンスター右側表示

else {

**MONSTER\_COUNT**=0;

}

//モンスターを下げる

if(count%5==0) {

**MONSTER\_Y**+=40;

}

//敵が自機に迫りきった時の処理

if(**MONSTER\_Y**==530) {

**GAMEOVER**=1;

}

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

/////////////////////////////////////////////////////////////////////////////////

//当たり判定などの処理/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

class Hantei extends Thread{

public void run() {

while(true) {

//if(start==0) {

try {

Thread.**sleep**(1);

for(int i=0;i<5;i++) {

//敵にビームが当たった時

if(**MONSTER\_COUNT**==0&&(int)**BEEM\_PLAYER\_X**>=**MONSTER\_XR**[i]&&(int)**BEEM\_PLAYER\_X**<=(**MONSTER\_XR**[i]+36)&&beem.getY()<(**MONSTER\_Y**+40)) {

**MONSTER\_DEATH**[i]=1;

**END**[i]=1;

**BAN\_STOP**[i]=1;

}

//敵にビームが当たった時

else if(**MONSTER\_COUNT**==1&&(int)**BEEM\_PLAYER\_X**>=**MONSTER\_XL**[i]&&(int)**BEEM\_PLAYER\_X**<=(**MONSTER\_XL**[i]+36)&&beem.getY()<(**MONSTER\_Y**+40)) {

**MONSTER\_DEATH**[i]=1;

**END**[i]=1;

**BAN\_STOP**[i]=1;

}

if(**MONSTER\_DEATH**[i]==1) {

**MONSTER\_DEATH**[i]=2;

}

}

//敵を全て倒した時

if(**END**[0]==1&&**END**[1]==1&&**END**[2]==1&&**END**[3]==1&&**END**[4]==1) **FINISH**=1;

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//ゲームが終わった時の画面を表示させておく時間を経過させるクラス///////////////////////////////

class GameEnd extends Thread{

public void run() {

try {

Thread.**sleep**(3000);

System.**exit**(0);//システム終了

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

////////////////////////////////////////////////////////////////////////////////////

//敵が破壊された時の画像表示の時間と表示を決める変数の処理///////////////////////////////////

class BanTime extends Thread{

public void run() {

try {

Thread.**sleep**(50);

**BAN\_X**=0;

**BAN\_Y**=0;

**PRINT\_BAN**=0;

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

////////////////////////////////////////////////////////////////////////////////////

//画面を更新させる処理/////////////////////////////////////////////////////////////////

class Repaint extends Thread {

public void run() {

while(true) {

repaint();//表示を更新

}

}

}

/////////////////////////////////////////////////////////////////////////////////////

}