JAVA응용프로그래밍

주차 과제

담당교수 : 천인국

제출일 : 2021. 04. 06

학과 : 컴퓨터 공학과

20194073 손정민

1. 프로그래밍 설계

추가하고자 하는 기능

1 적이 미사일을 발사하는 기능

2 외계인 (미사일을 맞으면 사라짐)이 랜덤하게 움직임

구현방법.

1. 적도 미사일을 발사하는 방법

적의 미사일 arraylist를 만든다.

우리편의 우주선과 충돌하는지를 계속 확인한다.

1. 외계인의 랜덤한 움직임 구현방법
2. 움직임

위 아래로 움직이다 일정범위를 벗어나면 튕긴다. 좌우로는 일정한 속도로 움직이고 위 아래로는 시간에 따라 속도를 바꿔 곡선으로 움직이는 것처럼 보이게 한다.

1. 미사일에 맞았을 경우

안보이게 만들고 리스트에서 지운다.

* 1. 코드 – UFO 클래스

import java.awt.Image;

import java.util.ArrayList;

import javax.swing.ImageIcon;

public class ufo extends Sprite {

private int mx=2, my=0;

private boolean up = true, speed = true;

int dx, dy=3;

public ufo(int x) {

super(x, 100);

loadImage("C:\\Users\\손정민\\Desktop\\학교\\3-1\\JAVA\\2주차\\src\\ufo.png");

}

public void move() {

x += mx;

y += my;

if (speed) {

my+=1;

if(my>10) {speed=false;}

}

else{

my-=1;

if(my<-10) {speed=true;}

}

if (x < 0 || x>890) {

mx=-mx;

}

}

}

* 1. 코드 – 수정된 우주선 코드(Spaceship.java)

import java.awt.Image;

import java.awt.event.KeyEvent;

import java.util.ArrayList;

import javax.swing.ImageIcon;

public class Spaceship extends Sprite {

private int dx;

private int dy;

private ArrayList<ufo> ufoList;

private ArrayList<Missile> m;

private ArrayList<Missile> Um;

private int i = 0;

public Spaceship(int x, int y) {

super(x, y);

loadImage("C:\\Users\\손정민\\Desktop\\학교\\3-1\\JAVA\\2주차\\src\\ship.png");

getImageDimensions();

m= new ArrayList<Missile>();

Um= new ArrayList<Missile>();

ufoList= new ArrayList<ufo>();

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

System.out.println("ufoList 가 추가됨 ");

ufoList.add(new ufo(200\*i));

}

}

public void move() {

x += dx;

y += dy;

}

public ArrayList<Missile> getMissile() {

return m;

}

public ArrayList<Missile> getUMissile() {

return Um;

}

public ArrayList<ufo> getUfo() {

return ufoList;

}

public void fire() {

m.add(new Missile(x, y,true));

}

public void U\_fire(int i) {

Um.add(new Missile(ufoList.get(i).getX(),ufoList.get(i).getY(),false));

i+=1;

if(i>5) {i-=5;}

}

public void keyPressed(KeyEvent e) {

int key = e.getKeyCode();

if (key == KeyEvent.VK\_SPACE) {

fire();U\_fire(i);

}

if (key == KeyEvent.VK\_LEFT) {

dx = -1;

}

if (key == KeyEvent.VK\_RIGHT) {

dx = 1;

}

if (key == KeyEvent.VK\_UP) {

dy = -1;

}

if (key == KeyEvent.VK\_DOWN) {

dy = 1;

}

}

public void keyReleased(KeyEvent e) {

int key = e.getKeyCode();

if (key == KeyEvent.VK\_LEFT) {

dx = 0;

}

if (key == KeyEvent.VK\_RIGHT) {

dx = 0;

}

if (key == KeyEvent.VK\_UP) {

dy = 0;

}

if (key == KeyEvent.VK\_DOWN) {

dy = 0;

}

}

}

* 1. 코드 – 수정된 보더 코드 (Boarder.java)

import java.awt.Color;

import java.awt.Graphics;

import java.awt.Graphics2D;

import java.awt.Toolkit;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyAdapter;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import java.util.ArrayList;

import javax.swing.JPanel;

import javax.swing.Timer;

public class Board extends JPanel implements ActionListener, KeyListener {

private Timer timer;

private Spaceship ship;

private final int DELAY = 10;

private int ny, ux, mx;

public Board() {

addKeyListener(this);

setFocusable(true);

setBackground(Color.BLACK);

ship = new Spaceship(500, 500);

timer = new Timer(DELAY, this);

timer.start();

}

@Override

public void paintComponent(Graphics g) {

super.paintComponent(g);

Graphics2D g2d = (Graphics2D) g;

if(ship.visible) {

g2d.drawImage(ship.getImage(), ship.getX(), ship.getY(), this);

}

if( ship.getMissile() != null ) {

for(Missile m : ship.getMissile() ) {

g2d.drawImage(m.getImage(), m.getX(), m.getY(), this);

}

}

if( ship.getUMissile() != null ) {

for(Missile m : ship.getUMissile() ) {

g2d.drawImage(m.getImage(), m.getX(), m.getY(), this);

}

}

if( ship.getUfo() != null ) {

for(ufo u : ship.getUfo() ) {

g2d.drawImage(u.getImage(), u.getX(), u.getY(), this);

}

}

else

System.out.println("없음.");

Toolkit.getDefaultToolkit().sync();

}

@Override

public void actionPerformed(ActionEvent e) {

ship.move();

if( ship.getMissile() != null ) {

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

ship.getMissile().get(i).move();

}

}

if( ship.getUMissile() != null ) {

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

ship.getUMissile().get(i).move();

}

}

if( ship.getUfo() != null ) {

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

ship.getUfo().get(i).move();

}

//지금의 y위치를 저장

if (ship.getUfo().size() != 0) {

ny=ship.getUfo().get(0).getY();

if( ship.getMissile() != null ) {

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

if(ship.getMissile().get(i).getY()>ny && ship.getMissile().get(i).getY()<(ny+80)) {

for(int j=0; j<ship.getUfo().size();j++) {

ux=ship.getUfo().get(j).getX();

mx=ship.getMissile().get(i).getX();

if(mx>ux && mx<(ux+100)) {

ship.getUfo().get(j).visible= false;

ship.getUfo().remove(j);

}

}

}

}

}

}

}

if( ship.getUMissile() != null ) {

for(Missile m : ship.getUMissile() ) {

if(ship.getX()<m.getX()&&ship.getX()>(m.getX()+40)) {

if(ship.getY()>(m.getY()-30)&&ship.getY()>(m.getY()+40)) {

ship.visible=false;

}

}

}

}

repaint();

}

@Override

public void keyReleased(KeyEvent e) {

ship.keyReleased(e);

}

@Override

public void keyPressed(KeyEvent e) {

ship.keyPressed(e);

}

@Override

public void keyTyped(KeyEvent arg0) {

}

}

* 1. 코드 – 수정된 미사일 코드 (Missile.java)

import java.awt.Image;

import javax.swing.ImageIcon;

public class Missile extends Sprite {

private final int MISSILE\_SPEED = 2;

int dx = 1, dy = 1;

boolean w;

public Missile(int x, int y, boolean s) {

super(x, y);

w=s;

loadImage("C:\\Users\\손정민\\Desktop\\학교\\3-1\\JAVA\\2주차\\src\\missile.png");

}

public void move() {

if(w) {

y -= MISSILE\_SPEED;

if (y < 0) {

visible = false;

}

}

else {

y += MISSILE\_SPEED;

if (y > 850) {

visible = false;

}

}

}

}

1. 실행결과

-파일추가

1. 코드해설(추가된 부분)
2. UFO 객체

좌우로는 일정하게 움직이도록 mx는 항상 일정한 값을 유지한다. 하지만 벽에 닫으면 튕기게 벽에 닫은 경우 방향을 바꾸도록 한다.

위 아래로 움직이는 경우에는 속도가 일정하게 증가하고, 일정하게 감소하도록 한다.

증가하고 감소하는 시점은 절대값으로 10이 넘어가면 바뀌도록 한다.

1. SpaceShip 클래스

미사일은 이 클래스에서 모든 ufo의 미사일을 공통적으로 생성하도록한다.

사용자의 미사일이 생성됨과 동시에 미사일을 발사하도록 하자.

미사일에 새로운 인수 중 true나 false를 전달하여 미사일이 어느 방향으로 나아가는지 전달한다.

1. Board 클래스

새롭게 추가된 미사일 클래스도 화면에 표시되고 움직이도록 이벤트에 추가 한다.

1. Missile 클래스

생성할 때 ture나 false를 인수로 받아 위로 움직일지 아래로 움직일지 결정한다.

1. 코드평가

게임이라 충돌이 일어나는 경우가 많은데, 충돌과 관련된 함수가 없어 일일이 적는 게 어렵다.

다음에 이 코드를 더 발전시키게 된다면 충돌과 관련된 함수를 생성하는 것이 좋아 보인다.