Άσκηση 8η

Ο σκοπός της 8ης άσκησης είναι η υλοποίηση της διαδραστικής συμπεριφοράς του παιχνιδιού. Συγκεκριμένα, είναι η επέκταση της 7ης και 6ης άσκησης έτσι ώστε *α)* το διαστημόπλοιο του χρήστη να ρίχνει πυρά στο αντίπαλο διαστημόπλοιο, *β)* το αντίπαλο διαστημόπλοιο να ρίχνει πυρά στο διαστημόπλοιο του χρήστη, *γ)* υλοποιεί μια συνθήκη τερματισμού του παιχνιδιού. Μπορείτε να επεκτείνετε το παιχνίδι κατά την αρέσκειά σας.

Υπενθυμίζουμε ότι η 8η άσκηση είναι η τρίτη από μία σύνθετη εργασία τριών ασκήσεων (6η, 7η και 8η) που έχουν ως στόχο το σχεδιασμό και την υλοποίηση ενός παιχνιδιού τύπου *arcade spaceship shooting game*. Σημειώστε ότι ενδεικτικός κώδικας για την άσκηση θα συζητηθεί και υπάρχει στις διαφάνειες της εβδομάδας.

Ζητούμενα της 8ης Άσκησης:

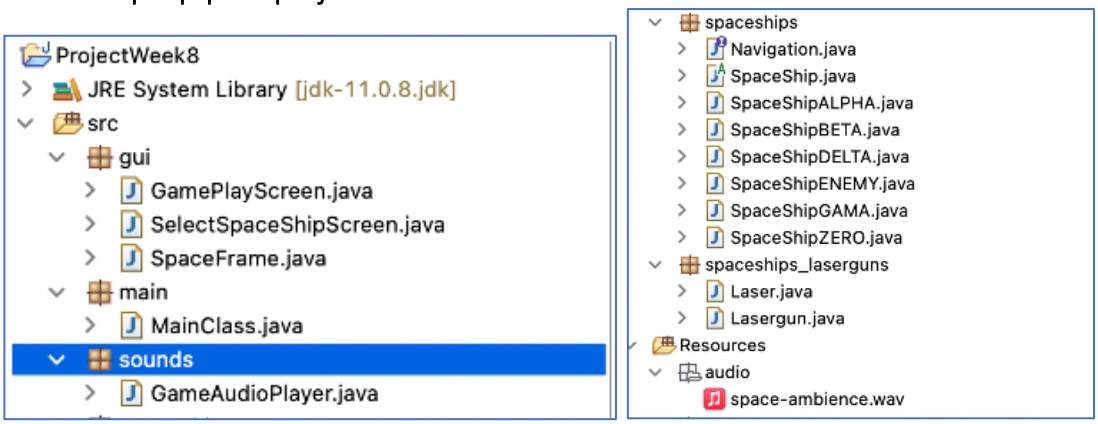
Α. Για τη συμπεριφορά του εχθρικού διαστημοπλοίου, υλοποιήστε μία γεννήτρια τυχαίων αριθμών θα μετακινεί το εχθρικό διαστημόπλοιο και θα ρίχνει διαστημικό λέιζερ στο διαστημόπλοιο του χρήστη.

Β. Για την υλοποίηση του διαστημικού λέιζερ, υλοποιήστε ένα καινούριο πακέτο ***`spaceships_laserguns.`*** Το λέιζερ του διαστημοπλοίου του χρήστη ενεργοποιείται με το κουμπί *space* στο πληκτρολόγιο. Μπορείτε να χρησιμοποιήσετε χρώμα του λέιζερ κατά την αρέσκειά σας όμως κάθε διαστημόπλοιο πρέπει να έχει διαφορετικό χρώμα στο λείζερ του. Για κάθε λέιζερ που φεύγει από το διαστημόπλοιο, μπορείτε να δημιουργήσετε ένα αντικείμενο της κλάσης ***`Laser`***, το οποίο θα προστίθεται σε μία συνδεδεμένη λίστα. Απεικονίστε τις δέσμες των λέιζερ χρησιμοποιώντας την ***`DrawLine`***.

Γ. Προσθέστε στην εφαρμογή σας ήχο με σκοπό να βελτιώσετε την εμπειρία του τελικού χρήστη. Για αυτό το λόγο υπάρχει ένα αρχείο ***space-ambience.wav*** στο **eclass** το οποίο θέλουμε να παίζει καθόλη τη διάρκεια του παιχνιδιού στο background.

Δ. Επιλέξετε τις συνθήκες τερματισμού του παιχνιδιού. Για παράδειγμα θα μπορούσε ένα διαστημόπλοιο να υλοποιεί μια συνθήκη η οποία θα δήλωνε πόσες φορές πρέπει να χτυπηθεί από το εχθρικό διαστημόπλοιο για να σταματήσει η ύπαρξή του.

Ενδεικτική δομή του project:



Ενδεικτικός κώδικας της Laser:

```
package spaceships;
import main.MainClass;
class Laser{
    int x;
    int y;
    Laser(int x, int y){
        this.x=x+MainClass.spaceShipWidth/2;
        this.y=y;
    }
}
```

Ενδεικτικός κώδικας της Lasergun:

```
package spaceships_laserguns;
import java.awt.Color;
import java.util.LinkedList;

public class Lasergun {
    public Color lasercolor;
    public LinkedList<Laser> laserShootersLinkedList=new LinkedList<Laser>();
    public Lasergun(Color lasercolor){
        this.lasercolor=lasercolor;
    }
    public void fire(int x, int y) {
        laserShootersLinkedList.add(laserShootersLinkedList.size(),new Laser(x,y));

    }
}
```

Ενδεικτικός κώδικας της SpaceShip:

```
package spaceships;
import java.util.LinkedList;
abstract public class SpaceShip implements Navigation,  LaserGuns{
    public LinkedList<Laser> laserShootersLinkedList=new LinkedList<Laser>();
    protected String SpaceShipName;
    protected int horPace;
    protected int verPace;
    protected int xCoord;
    protected int yCoord;
    protected ImageIcon SpaceShipImageIcon;


    public ImageIcon getIcon() {return SpaceShipImageIcon;}
    public int getX() {return xCoord;}
    public int getY() {return yCoord;}

    public void fire(int x, int y) {
        laserShootersLinkedList.add(laserShootersLinkedList.size(),new Laser(x,y));
    }
```

Ενδεικτικός κώδικας της GameAudioPlayer:

```
package sounds;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.Clip;
import javax.sound.sampled.FloatControl;

import main.MainClass;
public class GameAudioPlayer {
    private  AudioInputStream audioInputStream;
    Clip clip;
    public GameAudioPlayer(){
            loadSpaceAmbient();
   }
    void loadSpaceAmbient() {
        try{
        audioInputStream =
                AudioSystem.getAudioInputStream(MainClass.class.getResource("../audio/space-ambience.wav"));
                clip = AudioSystem.getClip();
                clip.open(audioInputStream);
                clip.loop(Clip.LOOP_CONTINUOUSLY);


                FloatControl gainControl =
                    (FloatControl) clip.getControl(FloatControl.Type.MASTER_GAIN);
                gainControl.setValue(-15.0f); // Reduce volume by 15 decibels.


            clip.start();
        }
        catch (Exception ex) {System.out.println(ex);}

    }
}
```

Ενδεικτικός κώδικας της GamePlayScreen:

```
GamePlayScreen(){
    addKeyListener(this);
    this.setVisible(true);
    this.setBackground( Color.BLACK );
    createDaemon();
}
private void createDaemon() {
    Timer timer = new Timer();
    TimerTask task = new monitorDeamonGame();
    timer.schedule(task, 100, 100);
}
    class monitorDeamonGame extends TimerTask{
      public void run(){repaint();}
    }
 @Override
 public void paintComponent (Graphics g) {
        super.paintComponent(g);
        enemySpaceShip.huntUserSpaceShip(userSpaceShip);
        userSpaceShip.getIcon().paintIcon(this, g, userSpaceShip.getX(), userSpaceShip.getY());
        enemySpaceShip.getIcon().paintIcon(this, g, enemySpaceShip.getX(), enemySpaceShip.getY());
        showLaserShootings(g);
 }

private void showLaserShootings(Graphics g) {
    userSpaceShip.gun.laserShootersLinkedList.forEach((tmp) -> {
        g.setColor(userSpaceShip.gun.lasercolor);
        g.drawLine(tmp.x, tmp.y, tmp.x, tmp.y-15);
        tmp.y=tmp.y-15; //move the line up
    });
    enemySpaceShip.gun.laserShootersLinkedList.forEach((tmp) -> {
        g.setColor(enemySpaceShip.gun.lasercolor);
        g.drawLine(tmp.x, tmp.y, tmp.x, tmp.y+15);
        tmp.y=tmp.y+15;
    });
}
```