Creating a Computer Games

The City of Liverpool College

Leandro Ruiz Boyle

17/05/2018

INDEX

Terms of reference

The purpose of this document is to prove the capability of creating, designing and testing a computer game.

For this purpose, there has been some research on the internet. Searching miscellaneous websites for each point of the assessment and selecting the most ad-hoc to the requirements. There is a major support of resources and lessons by coursing the module "Computer Games Design and Development" which helps the understanding of the subject, the opportunity to build and fell comfortable with the knowledge learned.

A work of design will be done previously to ensure that the time management is controlled and the code is structured is giving way to future improvements. Then the game will be tested in various wais: using black box testing and peer testing by a colleague. Finally a support documentation for the user.

Main Finding

Intro game

The Alien Loving Game is a 2D video game to play on any computer. The game is a player that tries to defend himself from many waves of attacks of aliens that come on each level. The player has a number of lives and needs to survive. To avoid losing all his lives the player has to kill the monsters that are attacking him by shooting a small heart at them. It the heart reaches the enemy the enemy will die and no longer will be able to damage the player. If an enemy reaches, the baseline where the player is the player will lose a life. Before he happens, the aim is to kill all the enemies before being killed.

The game has five levels to complete. Each of them is played in a different arena. Some levels are easier than others are. They progressively get harder to complete and more skills will be required to pass them. On each level, you will find that the enemies are different and they will move down faster or quicker from side to side. The first level is done so that anyone can pass it and get a little taste of the game, but the last level needs some practicing.

Objectives

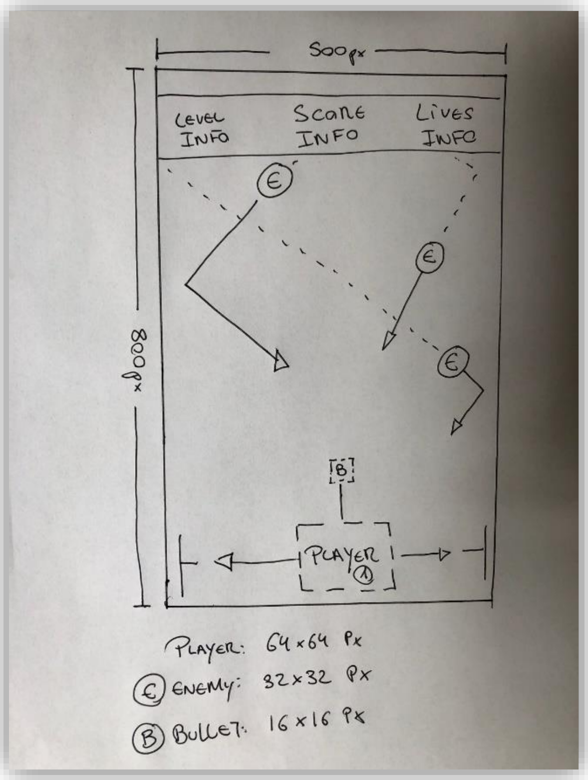
The task is to develop a video game by coding it from scratch. The approach on choosing what type of video game is going to develop, we first have considerate all skills known to be able to use it. It is important to approach this assessment to be able to feel comfortable with a programming language that the developer feels enough confidence to be able to deliver the game before the deadline. In this case I it is more secure to take an existing idea and convert it into a new version of the same type of game. The game developed will be based on the famous game "SPACE INVADERS" but with a twist of its own. There was the idea of developing a 3D game, or maybe a game more towards the physics aspect of the game. However, considering the time it is a more secure option to stick to a simple exercise, than a more complex time that could encounter many time problems.

There were many options to combine the attacks of the enemy with the attacks of the player; Have the enemies reaching the player by moving all in one block... at the end the best option that was chosen is having to kill all enemies that randomly move on the screen and approaches the baseline where the player will be shooting from. They start from the top of the screen and try to reach the bottom line.

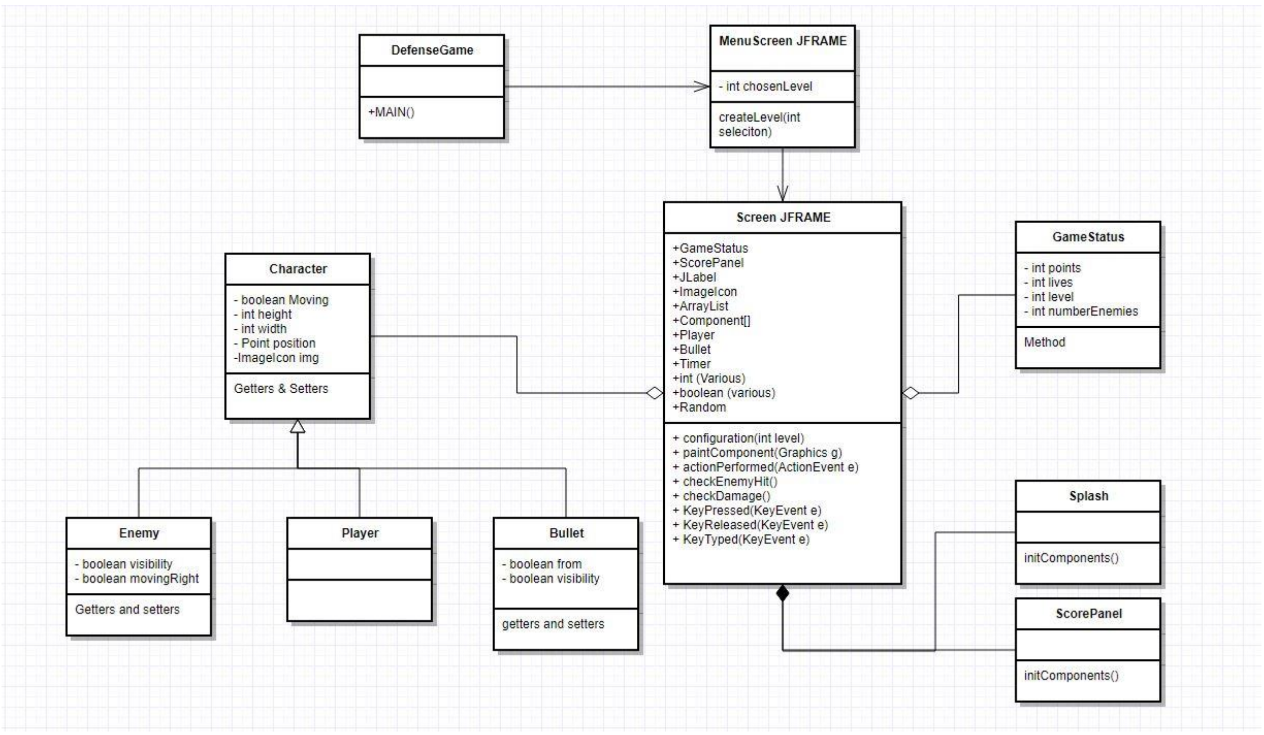
I would like to develop this game in an object-orientated design so that it is easier to scale the game to more levels, and making it easy towards adding functionality.

Design

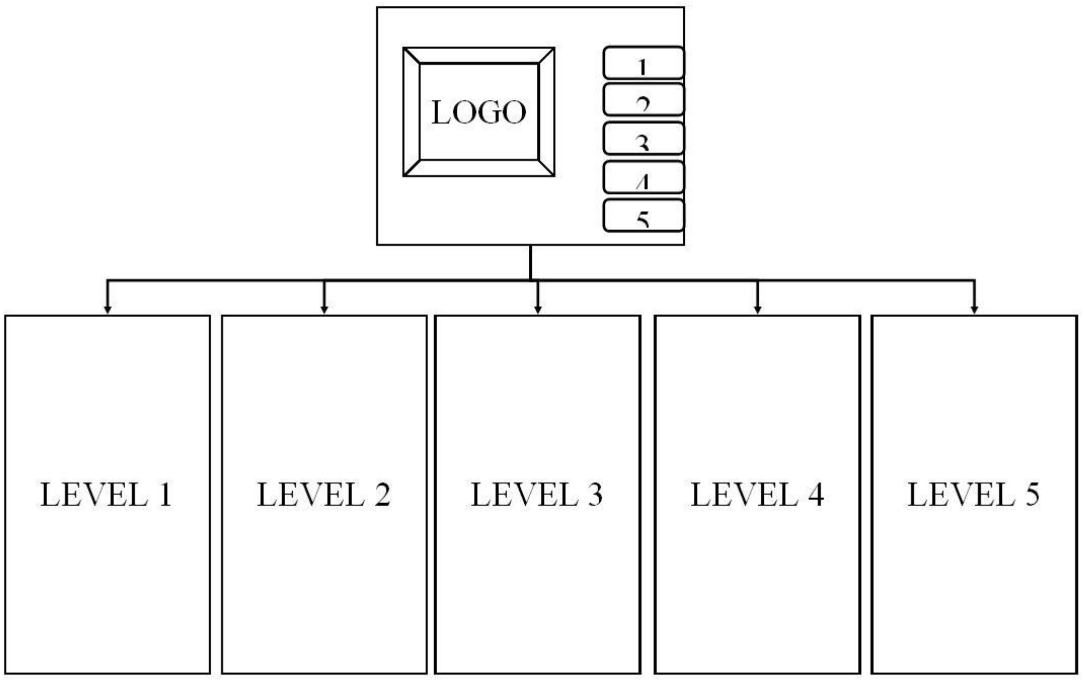
Draft



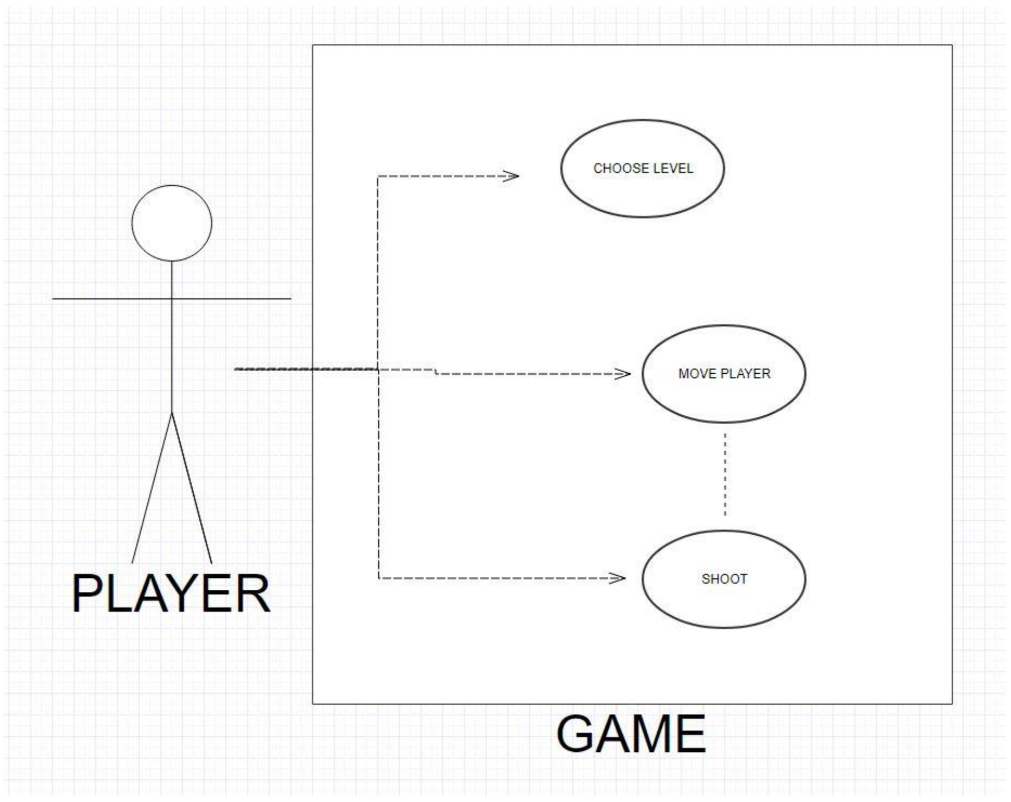
Class Structure



Flow Chart



Activity structure



The justification for the development package used.

The video game has been coded in JAVA programming language. I have chosen this programming language for various reasons.

The first reason I have chosen JAVA out of the hundreds of programming languages that there are is that JAVA is a language that I already know and I could tackle the task easier because I feel comfortable with it.

The second reason is to be able to be able to migrate the code to a web applet or maybe adapt it to an android phone by just reusing a huge amount of the code. This is an interesting point that might be on the bucket list after this course. It would only need to adapt to a touchscreen and set the improvements that will be discussed later on. This code will be able to run on any platform that is JAVA compatible. Laptops, computers, tablets, phones…

Third, JAVA is a language that has a plenty of documentation on handling graphics and this was helpful to be able to code while checking examples on the internet.

By choosing JAVA I have chosen to work with NETBEANS because it is an IDE that I find fast to create the layout, so I could focus more on the game and physics and less on the layout of the menu. I used the integrated layout designer. NetBeans has an easy debugger and many shortcuts to libraries methods.

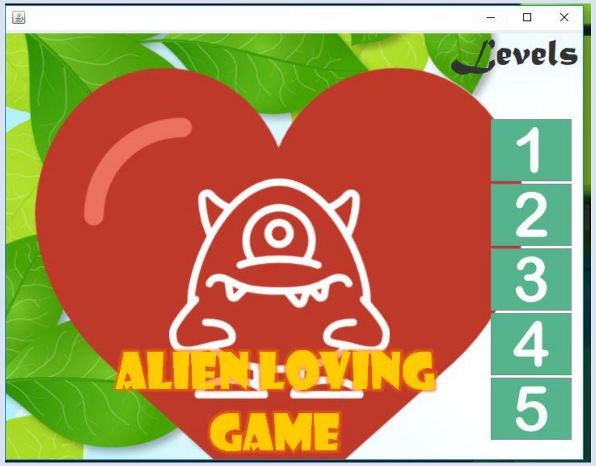
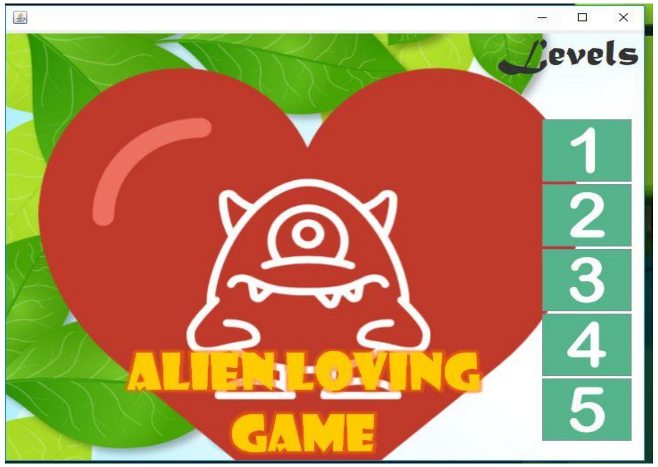
TEST PLAN

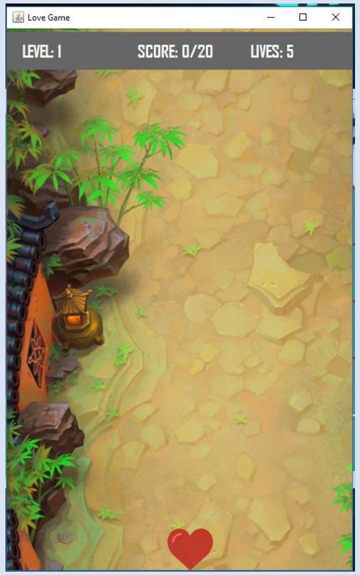
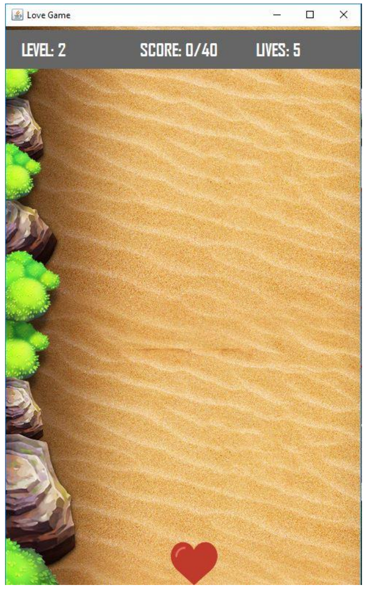
Test	Date	Purpose of test	Input(test data)	Expected Output
1	**10/05/2018**	Run the program	Run the programme with CMD “java class name”	The programme should run without any problem.
2	**10/05/2018**	Check if all layout or writing mistakes appear	No entry of data	There should be no mistakes.
3	**10/05/2018**	Click on the button 1	Mouse click	Level 1 should open correctly
4	**10/05/2018**	Click on the button 2	Mouse click	Level 2 should open correctly
5	**10/05/2018**	Click on the button 3	Mouse click	Level 3 should open correctly
6	**10/05/2018**	Click on the button 4	Mouse click	Level 4 should open correctly
7	**10/05/2018**	Click on the button 5	Mouse click	Level 5 should open correctly
8	**10/05/2018**	Close the game screen	Click on the window exit	The game should close and the game menu should still be on the show.
9	**10/05/2018**	Check if the player moves left	Press the left arrow on the keyboard once	The player should move 20 pixels to the left

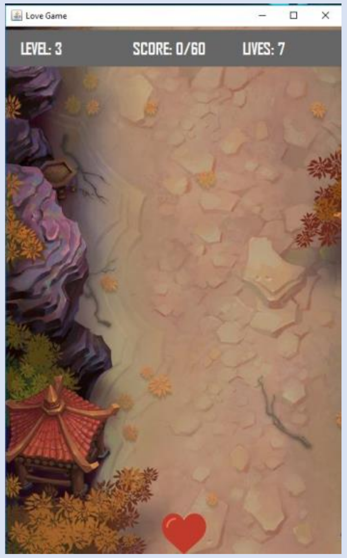
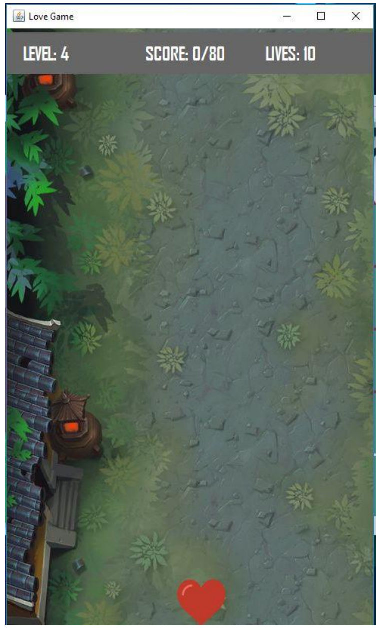
		One position		
10	**10/05/2018**	Check if the player moves right One position	Press the right arrow on the keyboard once	The player should move 20 pixels to the right
11	**10/05/2018**	Check if the player moves left While the button is pressed	Press the left arrow on the keyboard once and maintain	The player should move a to the left until it reaches the border or the button is released.
12	**10/05/2018**	Check if the player moves right While the button is pressed	Press the right arrow on the keyboard once and maintain	The player should move to the right until it reaches the border or the button is released.
13	**10/05/2018**	Check that the player doesn't leave the screen to the right	Force him to go out by using the right arrow on the keyboard	It shouldn't go out the screen
14	**10/05/2018**	Check that the player doesn't leave the screen to the left	Force him to go out by using the left arrow on the keyboard	It shouldn't go out the screen
15	**10/05/2018**	The bullet heart works	Press spacebar key from the keyboard	The bullet should run up the screen from the player's position
16	**10/05/2018**	Check enemies are killed	Shoot a bullet and hit an enemy	The enemy should stop and change its image for a dead enemy.
17	**10/05/2018**	Check the score label add one point when hitting an enemy	Shoot at an enemy	If an enemy is hit the score label should increase in 1

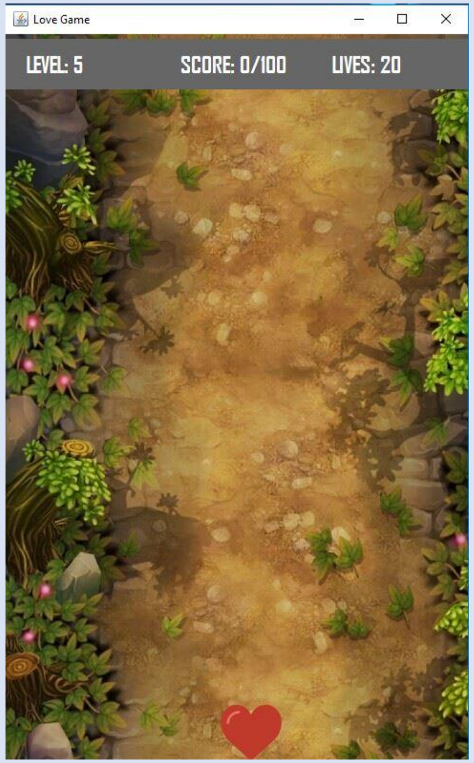
18	10/05/2018	Check enemies splash at the reach of the bottom line	Leave the enemies to reach the bottom line	The enemy should change the image to a splash image when reaching the bottom of the screen
19	10/05/2018	Check if lives decrease when bottom line has been reached	Leave the enemy to reach the bottom and check the change in the life label	The lives should decrease in one
20	10/05/2018	Check the splash appears when completing the game	Kill all the enemies	The game should show a message LEVEL COMPLETED
21	10/05/2018	Check the splash appears when being killed by the enemies	Let the enemies kill the player	The game should show a message of GAME OVER

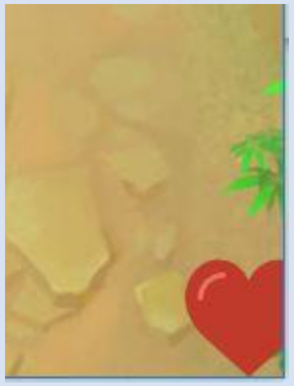
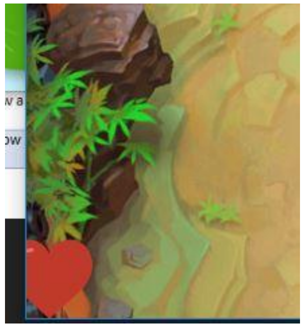
TEST LOG

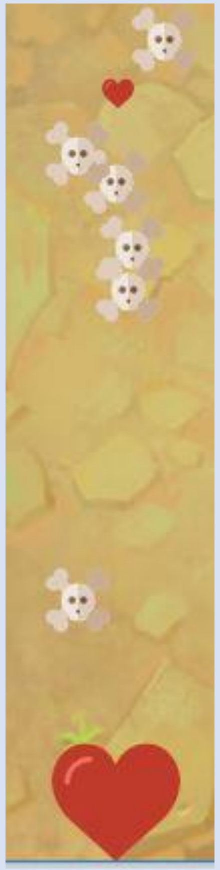
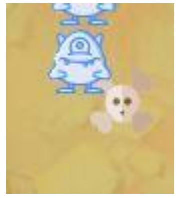
	Date	Actual Output	Success	Comments/Screenshots
1	10/05/2018	The programme opens without any problems. The result was the expected.	YES	
2	10/05/2018	The interface does not have any spelling mistakes, but it could improve. But the test was successful. The result was the expected.	YES	

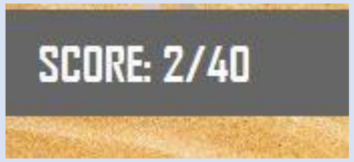
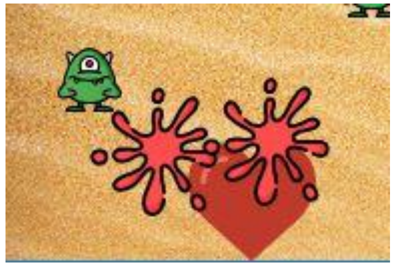
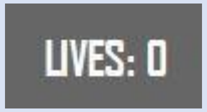
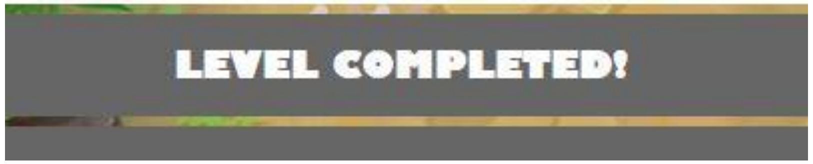
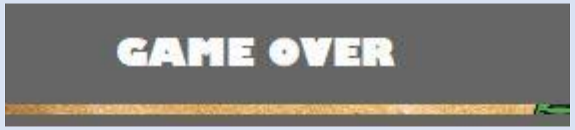
3	**10/05/2018**	Level one has correctly loaded The result was the expected.	YES	
4	**10/05/2018**	Level two has correctly loaded. The result was the expected.	YES	

5	**10/05/2018**	Level 3 has correctly loaded. The result was the expected.	YES	
6	**10/05/2018**	Level4 has correctly loaded. The result was the expected.	YES	

7	**10/05/2018**	Level 5 has correctly loaded. The result was the expected.	YES	
8	**10/05/2018**	When closing the game it returns to the main menu. The result was the expected.	YES	This test cannot be evidenced with a screenshot
9	**10/05/2018**	It correctly moves to a position 30 pixels to the left. The result was the expected.	YES	This test cannot be evidenced with a screenshot

10	10/05/2018	It correctly moves to a position 30 pixels to the right. The result was the expected.	YES	This test cannot be evidenced with a screenshot
11	10/05/2018	It correctly moves to a position 30 pixels to the left as many times while the key button Is pressed. The result was the expected.	YES	This test cannot be evidenced with a screenshot
12	10/05/2018	It correctly moves to a position 30 pixels to the left as many times while the key button Is pressed. The result was the expected.	YES	This test cannot be evidenced with a screenshot
13	10/05/2018	It only goes out a bit, but this is because the layer needs to be able to shoot from the very corner to not have blind spaces when attacking. The result was the expected.	YES	
14	10/05/2018	It only goes out a bit, but this is because the layer needs to be able to shoot from the very corner to not have blind spaces when attacking. The result was the expected.	YES	

15	**10/05/2018**	Has we can see in the evidence, the bullet was shouted and it moves up very quickly. The result was the expected.	YES	
16	**10/05/2018**	The enemies got hit change into the death picture. The result was the expected.	YES	

17	10/05/2018	The test was successful, two enemies were killed and two points were added to the scoreboard. The result was the expected.	YES	 SCORE: 2/40
18	10/05/2018	The enemies correctly changed into a splash image when reached the base line The result was the expected.	YES	
19	10/05/2018	The lives panel decreases when the enemies touch the bottom line. The result was the expected.	YES	 LIVES: 0
20	10/05/2018	The splash message appears with the correct message when completing the game surviving the enemy wave. The result was the expected.	YES	 LEVEL COMPLETED!
21	10/05/2018	The splash message appears with the correct message when completing the game NOT surviving the enemy wave. The result was the expected.	YES	 GAME OVER

Peer Testing

User Feedback: Andrew Laing

Alien Loving Game is beautifully designed, fun to play and can become quite addictive. The backgrounds, characters and fonts used work well together, creating a game world that I wanted to return to time and time.

The characters move smoothly enough on the screen and can be quite challenging to hit, especially in the later levels when they become more numerous.

It may have been a better idea to implement the bullets to travel until they either hit a character or exit the screen, because at present they disappear as soon as the spacebar is pressed again, which can make the task of judging when to fire to hit multiple moving targets difficult.

I also think that the side-to-side movement of the main character could become a little more responsive. At present, there is just enough lag between arrow key press and continuous movement to render quick direction changes impossible.

The main screen should have been created using the same portrait orientated rectangle shape as the level screens. All screens should also be run within a single window/frame, as this would make the game very suitable for adaptation to the Android framework.

Another slight worry for me was that I could open and attempt to play multiple level instances simultaneously. If screens were run within, a single window/frame I would also suggest that they allowed the player to return to the main screen with a key press after a level had been defeated or the player had lost the game.

The game could benefit from the addition of background music and sound effects. Character animations would also improve it. Overall, though, the game is fun, nice to look at, and with a few minor changes made to it could easily become a highly marketable Mobile phone application.

Installation guide

The file will be stored on your computer in any place you want to place it in. The normal form of saving it would be in the folder called "Programme Files" and under the name of the game. Then create a shortcut to the executable, as many other games work. By placing, the full directory that holds all the programme folders will be ok.

As we know this game has been built with JAVA so the procedure will be the same for most of the operating systems. Make sure you have installed the Java Virtual Machine. That would be one of the requirements, and update the Java version on the computer.

The executable is a file with a ".jar" extension. By doing double click on the shortcut it should be enough to be able to run the programme.

The game is very simple and the installation process would only be to decide where to place all the files and create the shortcut. No internal settings to be done.

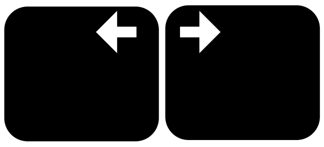
Any further questions or issues encountered in the installation process please contact the developer: Leandro Ruiz Boyle +44 751196545252 or email him any issue to leandroDeveloper@IT.co.uk

User Guide

How to play Alien Love Game?

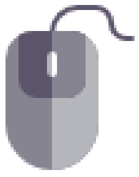
A one-player game needs a mouse and a keyboard to play with it. It is a very easy game to play and it all starts by choosing one of the levels as soon as the game has started. There are five levels to play. They progressively increase in difficulty having number one as the easiest and five the hardest to play. Players are free to choose any.

Once the game starts, the controls are.



Use the arrows left and right on the keyboard to move the player on the screen from side to side. If you keep, the button pressed down, the player will move larger distances and quicker.

Use the spacebar to fire the bullet to attack the enemies. Press the spacebar once and do not maintain it pressed down. Let the bullet move towards his target. If you press the space bar, again, the old bullet will disappear and a new one is fired.



Use the mouse to close the game and start a new one or choose levels.

The goal of the game is to try surviving the waves of enemies that will take your likes if they reach you. The player shall fire his bullets to avoid the enemies to reach where the player is. The player will have to hit the enemies in order to them to die.

Evaluation and Improvements

Evaluation

Personally, the development of this small video game has been a great experience to learn and understand how games are made and also understand how applications can be more dynamic by using the concepts of moving objects on a screen. I have gain experience by using new libraries and a new combination of tools. It has been an essential step in my programming education.

I believe that the game meets all requirements and therefore I am happy with the job delivered. It is always about meeting what the customer is asking for. In this case, it was a video game where a player could play and score points and could lose the game.

The actual programming took me about 3 days to programme de game. Although I will say that I spent a whole day designing it a trying to think ahead before coding it. Nevertheless, my inexperience clearly made it a waste of time. When the game was taken to code there were too many facts that were not considered, so I could not stick to my first design.

One of the main aims with this video game in terms of coding was to try to build it in a way that it could easily expand to more levels, manage the difficulty, of each individual levels... This was made so I could just change any aspect of the game very easily. It was one of the personal challenges to improve my own way of coding. I manage to organize the code by dividing it between the interface, and the game engine in two separate packages. Then also, I created even an abstract class to be able to work with the objects that were going to be moving on the screen.

By coding in JAVA this exercise has been easy to do in terms of having a lot of documentation and already known the language. There were many video tutorials on YouTube where I could get ideas and learn how to use certain libraries inside JAVA.

Test plan evaluation

The results of the testing were all ok. It was not a very deep testing on the application. It was tested to meet criteria and make sure the main functionality made it play the game. All tests were successful. Therefore, no more test needs doing or changes need to be done. The game should run ok.

I believe that if this assignment were for a customer, then maybe a white box testing would be great to try working and catching boundaries errors, it is important to check that all primitive data declared can hold a huge increasing of speed, number of characters, bullets…

Improvements

This game is open to plenty of improvements. Today's game market is a huge billionaire and the type of game that I have developed is more a type of game that you would play on a smartphone platform. Desktop games are normally big complex games, and smartphones are used to play also simple games.

Before tackling the improvements that my own think should take place, I should acknowledge that Andrew Laing has done a great job in giving a great feedback on what mistakes should be fixed or what improvements should be taken place.

The first improvement that I would suggest is to transfer it to android so that it could be played on a platform that will make the game more likable. Most of the code as mentioned before is made in JAVA so it will be easy to migrate it from one platform to another.

The second improvement would be to be able to save the game so that the players can continue their progress.

Thirdly indicate what the highest score was made at all levels to create a motivation for a player to try to beat their own records.

Another improvement could be by adding a boss character after the wave attack. This boss could have lived as well. Therefore, it would need more than a shot to kill him. The boss would also shoot at the player to make it more interesting.

Fifth, create more than 10 levels. This will make the game more interesting, you want the game to be played as much as possible. If the game ends quickly then the game will not be played as much.

It would be good to be able to shoot more than one bullet. On the demo, you have to wait until the bullet has done the damage or has disappeared to be able to shoot again. The improvement would be by letting shoot many bullets so more than one bullets are traveling through the screen at the same time.

Similar games have music in the game and sound effects when a game event occurs. This would definitely increase the emotions that the player would experience. Also if this is controlled by the player, so that the sound effects and music are used only if the player wants. This could be a great improvement.

Finally, I would suggest a personal preference of being able to swap the image of the enemies for a picture taken with the camera. Therefore, those friends can play and make the enemies customizable.

As said previously this improvement are options to set the game closer to the actual market standards. Although the graphics are the main thing to improve focus on the aspects that we have obtained from the "peer testing" that Andrew Laing.

Game source code

DefenseGame.java

```
package defensegame;

import views.MenuScreen;

/**
* The DefenseGame program implements an application that
* simply displays a Menu Screen to the standard output.
*
* @author  Leandro Ruiz Boyle
* @version 1.0
* @since   2018-05-10
*/
public class DefenseGame {
    public static void main(String[] args) {

        //We create an object of the Menu JFrame
        MenuScreen ms = new MenuScreen();
        ms.setLocationRelativeTo(null);
        ms.setVisible(true);
    }
}
```

GameStatus.java

```
package gameControl;
/**
 *  This class is a data structure to keep the game data.
 * @author Leandro Ruiz Boyle
 */
public class GameStatus {
   //atributes
    private int points = 0;
   private int lives;
   private int level;
   private int numberEnemies;


   //Constructors
```

```
    public GameStatus() {
    }
    public GameStatus(int lives, int level, int ne) {
        this.lives = lives;
        this.level = level;
        this.numberEnemies = ne;
   }

    //Getters and Setters
    public int getNumberEnemies() {
        return numberEnemies;
    }

    public void setNumberEnemies(int numberEnemies) {
        this.numberEnemies = numberEnemies;
    }

    public int getPoints() {
        return points;
    }

    public void setPoints(int points) {
        this.points = points;
    }

    public int getLives() {
        return lives;
    }

    public void setLives(int lives) {
        this.lives = lives;
    }

    public int getLevel() {
        return level;
    }

    public void setLevel(int level) {
        this.level = level;
    }
}
```

Bullet.java

```
package models;

import javax.swing.ImageIcon;
/**
 *  This class will show will instanciate the
 * bullet that the main player ahs shoted.
 *
 * @author Leandro Ruiz Boyle
 */
public class Bullet extends Character{
    //attributes
    private boolean from;
    private boolean visible = false;

    //Constructor
    public Bullet(boolean from,int x,int y) {
        this.from = from;
        if (from == true) {
            this.setImg(new ImageIcon("src\\icons\\LoveBullet.png"));
        }
        this.setPosition(x,y);
    }

    public boolean isFrom() {
        return from;
    }

    public void setFrom(boolean from) {
        this.from = from;
    }

    public boolean isVisible() {
        return visible;
    }

    public void setVisible(boolean visible) {
        this.visible = visible;
    }
}
```

Character.java

```
package models;
```

```
import java.awt.Point;
import javax.swing.ImageIcon;

abstract class Character {
    //attributes
    private boolean moving = true;
    private int height = 50;
    private int width = 50;
    private Point position = new Point(0,0);
    private ImageIcon img;

    //Getters and Setters
    public boolean isMoving() {
        return moving;
    }

    public void setMoving(boolean move) {
        this.moving = move;
    }

    public ImageIcon getImg() {
        return img;
    }

    public void setImg(ImageIcon img) {
        this.img = img;
    }

    public int getHeight() {
        return height;
    }

    public void setHeight(int height) {
        this.height = height;
    }

    public int getWidth() {
        return width;
    }

    public void setWidth(int width) {
        this.width = width;
    }
    public Point getPosition() {
        return position;
    }

    public void setPosition(int x, int y) {
        this.position = new Point(x,y);
    }

    @Override
    public String toString() {
        return "Character{" + "height=" + height + ", width=" + width + ", position=" + position + ", img=" + img + '}';
    }
}
```

Enemy.java

```
package models;

import javax.swing.ImageIcon;

/**
 *  This class creates an Enemy object
 *  to store all the information about an enemy of the game.
 * @author Leandro Ruiz Boyle
 */
public class Enemy extends Character {
    //Atributes
    private boolean visibility = false;
    private boolean movingRight = true;

    //Constructor
    public Enemy(int x, int y,ImageIcon img){

        setImg(img);
        setPosition(x, y);
    }

    //Getters and Setters
    public boolean isVisibility() {
        return visibility;
    }
    public void setVisibility(boolean visibility) {
        this.visibility = visibility;
    }

    public boolean isMovingRight() {
        return movingRight;
    }

    public void setMovingRight(boolean movingRight) {
        this.movingRight = movingRight;
    }
   }
```

Player.java

```
package models;

import javax.swing.ImageIcon;

/**
 *  This class creates an instance of the main player in the game.
 *
 * @author Leandro Ruiz Boyle
 */
public class Player extends Character {
    //contructor
    public Player(int x, int y){
        this.setPosition(x,y);
        this.setImg(new ImageIcon("src\\icons\\Player1.png"));
    }

}
```

MenuScreen.java

```
package views;

import javax.swing.JFrame;
/**
 *  This class creates an Panel where it will allow
 *  the user to choose a game level.
 *
 * @author Leandro Ruiz Boyle
 */
public class MenuScreen extends javax.swing.JFrame {

    public MenuScreen() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel6 = new javax.swing.JLabel();
        btn_five = new javax.swing.JButton();
        btn_one = new javax.swing.JButton();
        btn_two = new javax.swing.JButton();
        btn_three = new javax.swing.JButton();
        btn_four = new javax.swing.JButton();
        lbl_Background = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setMaximumSize(new java.awt.Dimension(730, 570));
        setMinimumSize(new java.awt.Dimension(730, 570));
        getContentPane().setLayout(null);

        jLabel6.setFont(new java.awt.Font("Matura MT Script Capitals", 0, 48)); // NOI18N
        jLabel6.setText("Levels");
        getContentPane().add(jLabel6);
        jLabel6.setBounds(550, 0, 160, 65);

        btn_five.setBackground(new java.awt.Color(255, 255, 0));
        btn_five.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/five.png"))); // NOI18N
        btn_five.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                btn_fiveMouseClicked(evt);
            }
        });
        getContentPane().add(btn_five);
        btn_five.setBounds(600, 430, 100, 77);
        btn_five.getAccessibleContext().setAccessibleName("btn_five");

        btn_one.setBackground(new java.awt.Color(255, 255, 0));
        btn_one.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/one.png"))); // NOI18N
        btn_one.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                btn_oneMouseClicked(evt);
            }
        });
        getContentPane().add(btn_one);
        btn_one.setBounds(600, 110, 100, 77);
        btn_one.getAccessibleContext().setAccessibleName("btn_one");

        btn_two.setBackground(new java.awt.Color(255, 255, 0));
        btn_two.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/two.png"))); // NOI18N
        btn_two.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                btn_twoMouseClicked(evt);
            }
        });
        getContentPane().add(btn_two);
        btn_two.setBounds(600, 190, 100, 77);
        btn_two.getAccessibleContext().setAccessibleName("btn_Two");

        btn_three.setBackground(new java.awt.Color(255, 255, 0));
        btn_three.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/three.png"))); // NOI18N
        btn_three.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                btn_threeMouseClicked(evt);
            }
        });
        getContentPane().add(btn_three);
        btn_three.setBounds(600, 270, 100, 77);
        btn_three.getAccessibleContext().setAccessibleName("btn_Three");

        btn_four.setBackground(new java.awt.Color(255, 255, 0));
        btn_four.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/four.png"))); // NOI18N
        btn_four.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                btn_fourMouseClicked(evt);
            }
        });
        getContentPane().add(btn_four);
        btn_four.setBounds(600, 350, 100, 77);
        btn_four.getAccessibleContext().setAccessibleName("btn_four");
```

```
        lbl_Background.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/Menu.png"))); // NOI18N
        lbl_Background.setText("jLabel1");
        lbl_Background.setMaximumSize(new java.awt.Dimension(600, 700));
        lbl_Background.setMinimumSize(new java.awt.Dimension(600, 700));
        lbl_Background.setPreferredSize(new java.awt.Dimension(600, 700));
        getContentPane().add(lbl_Background);
        lbl_Background.setBounds(0, 0, 730, 700);

        jLabel2.setText("jLabel2");
        getContentPane().add(jLabel2);
        jLabel2.setBounds(250, 160, 34, 14);

        pack();
    }// </editor-fold>

    private void btn_oneMouseClicked(java.awt.event.MouseEvent evt) {
      openScreen(1);
    }

    private void btn_twoMouseClicked(java.awt.event.MouseEvent evt) {
        openScreen(2);
    }

    private void btn_threeMouseClicked(java.awt.event.MouseEvent evt) {
       openScreen(3);
    }

    private void btn_fourMouseClicked(java.awt.event.MouseEvent evt) {
       openScreen(4);
    }

    private void btn_fiveMouseClicked(java.awt.event.MouseEvent evt) {
        openScreen(5);
    }

    /**
* This is a function that is creating a new game depending on the
* level selected by the player.
* @param int number of the level the user wants to platy
*/
public void openScreen(int number){
    JFrame f = new JFrame();
        f.add(new Screen(number));
        f.setSize(500, 800);
        f.setLocationRelativeTo(null);
        f.setTitle("Love Game");
        f.setVisible(true);
}


    // Variables declaration - do not modify
    private javax.swing.JButton btn_five;
    private javax.swing.JButton btn_four;
    private javax.swing.JButton btn_one;
    private javax.swing.JButton btn_three;
    private javax.swing.JButton btn_two;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel lbl_Background;
    // End of variables declaration
}
```

ScorePanel.java

```
package views;

/**
 *  This class creates a panel that will update the information about
 * the game. points level,...
 *
 * @author Leandro Ruiz Boyle
 */
public class ScorePanel extends javax.swing.JPanel {

    //constructor
    public ScorePanel() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPasswordField1 = new javax.swing.JPasswordField();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();

        jPasswordField1.setText("jPasswordField1");

        setBackground(new java.awt.Color(102, 102, 102));
        setForeground(new java.awt.Color(153, 153, 153));

        jLabel1.setFont(new java.awt.Font("Agency FB", 1, 24)); // NOI18N
        jLabel1.setForeground(new java.awt.Color(255, 255, 255));
        jLabel1.setText("Level: 0");

        jLabel2.setFont(new java.awt.Font("Agency FB", 1, 24)); // NOI18N
        jLabel2.setForeground(new java.awt.Color(255, 255, 255));
        jLabel2.setText("Score: 0");

        jLabel3.setFont(new java.awt.Font("Agency FB", 1, 24)); // NOI18N
        jLabel3.setForeground(new java.awt.Color(255, 255, 255));
        jLabel3.setText("Lives: 0");

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(30, 30, 30)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(61, 61, 61)
                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 129,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 30, Short.MAX_VALUE)
                .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(50, 50, 50))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
```

```
                    .addComponent(jLabel3, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)
                        .addComponent(jLabel2)
                        .addGap(0, 0, Short.MAX_VALUE)))
                .addContainerGap())
        );

        jLabel1.getAccessibleContext().setAccessibleName("lbl_Level");
        jLabel2.getAccessibleContext().setAccessibleName("lbl_Score");
        jLabel3.getAccessibleContext().setAccessibleName("lbl_Lives");
    }// </editor-fold>

    // Variables declaration - do not modify
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPasswordField jPasswordField1;
    // End of variables declaration
}
```

Screen.java

```
package views;

import gameControl.GameStatus;
import java.awt.Component;
import java.awt.Graphics;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.util.ArrayList;
import java.util.Random;
import javax.swing.ImageIcon;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.Timer;
import models.Bullet;
import models.Enemy;
import models.Player;
```

```
/**
 *  This class is the class that will hold most of the code .In this class we have
 * Physics, events, timmers,...
 *
 * @author Leandro Ruiz Boyle
 */
public class Screen extends JPanel implements ActionListener, KeyListener{
    // GLOBAL variables
    GameStatus gs;
    ScorePanel sp;
    Splash splash;
    JLabel lbl1,lbl2,lbl3,lblSplash;

    ImageIcon pImage,bImage,eImage,background;
    ArrayList<Enemy> enemiesList = new ArrayList<>();
    Component[] lbl_list;

    Player p;
    Bullet b;
    Timer t = new Timer(20,this);

    int speedPlayer = 30, speedBullet = 30, speedEnemyY = 1, speedEnemyX = 1;
    int pX = 225, pY = 700;
    int eX = 0, eY = 0;
    int count = 0;
    int enemyCount = 0;
    int NUMBER_OF_ENEMIES;
    boolean shooting = false, start = false, nextEnemy = false;
    Random randomGenerator;

    //Constructor
    public Screen(int level){
        splash = new Splash();
        splash.setVisible(false);
        lblSplash = (JLabel) splash.getComponent(0);
        gs = new GameStatus(5,1,10); //points , lives , level
        configuration(level);
        NUMBER_OF_ENEMIES = gs.getNumberEnemies();

        p = new Player(pX,pY);
        t.start();
        sp = new ScorePanel();
        this.add(splash);
        this.add(sp);
        addKeyListener(this);
        setFocusable(true);
        setFocusTraversalKeysEnabled(false);

        randomGenerator = new Random();
        for(int i = 0; i<NUMBER_OF_ENEMIES;i++){
            int temp = randomGenerator.nextInt(450);
            enemiesList.add(new Enemy(temp,eY,eImage));
        }
        lbl_list = sp.getComponents();
        lbl1 = (JLabel) lbl_list[0];
        lbl2 = (JLabel) lbl_list[1];
        lbl3 = (JLabel) lbl_list[2];

        lbl3.setText("LIVES: "+gs.getLives());
        lbl2.setText("SCORE: 0/"+NUMBER_OF_ENEMIES);
        lbl1.setText("LEVEL: "+gs.getLevel());


    }

    /**
    * This function sets the details specific for each level.
    * @param int level .
    */
    public void configuration(int level){

        switch(level){
            case 1:                              //LEVEL 1
                gs.setLevel(1);
                gs.setLives(5);
                gs.setNumberEnemies(20);
                speedEnemyX = 3;
                background = new ImageIcon("src\\icons\\background1.jpg");
                eImage = new ImageIcon("src\\icons\\Boss1.png");
                break;
            case 2:                              //LEVEL 2

                gs.setLevel(2);
                gs.setLives(5);
                gs.setNumberEnemies(40);
                speedEnemyX = 5;
                background = new ImageIcon("src\\icons\\background2.jpg");
                eImage = new ImageIcon("src\\icons\\Boss2.png");
                break;
            case 3:                                  //LEVEL 3

                gs.setLevel(3);
                gs.setLives(7);
                gs.setNumberEnemies(60);
                speedEnemyY = 2;
                speedEnemyX = 2;
                 background = new ImageIcon("src\\icons\\background3.jpg");
                 eImage = new ImageIcon("src\\icons\\Boss3.png");
                break;
            case 4:                                  //LEVEL 4

                gs.setLevel(4);
                gs.setLives(10);
                gs.setNumberEnemies(80);
                speedEnemyY = 2;
                speedEnemyX = 4;
                background = new ImageIcon("src\\icons\\background4.jpg");
                eImage = new ImageIcon("src\\icons\\Boss4.png");
                break;
            case 5:                                  //LEVEL 5

                gs.setLevel(5);
                gs.setLives(20);
                gs.setNumberEnemies(100);
                speedEnemyY = 3;
                speedEnemyX = 4;
                background = new ImageIcon("src\\icons\\background5.jpg");
                eImage = new ImageIcon("src\\icons\\Boss5.png");
                break;
        }
     }

      /**
    * This function draws all the components on the screen.
    * @param Graphics g .
    */
     @Override
     public void paintComponent(Graphics g) {
         //BACKGROUND DRAWING
        g.drawImage(background.getImage(), 0, 0, this);

        //PLAYER DRAWING
        pImage = p.getImg();
        g.drawImage(pImage.getImage(), pX, pY, this);

        //BULLET DRAWING
        if((shooting)&&(b.isVisible())){
           bImage = b.getImg();
           g.drawImage(bImage.getImage(), b.getPosition().x, b.getPosition().y, this);
        }


        //ENEMY DRAWING
        for(Enemy e: enemiesList){
            if(e.isVisibility()){
               eImage =  e.getImg();
               if(e.isMoving()){
                  e.setPosition(e.getPosition().x , e.getPosition().y+speedEnemyY);
               }
               if(e.isVisibility()){
                   g.drawImage(eImage.getImage(), e.getPosition().x, e.getPosition().y , this);
               }
            }
        }
      }

      /**
     * This listener gets triggered by any event. In this
     * case the main event that it controls is the ticking of the
     * @param ActionEvent e .
     */
      @Override
      public void actionPerformed(ActionEvent e) {


         repaint();
         if(count>250){
            checkEnemyHit();
         }
         checkDamage();

         //Move bullet one position up each tick of the timer
        if((shooting)&&(b.getPosition().y > -20)&&(b.isMoving())){
            b.setPosition(b.getPosition().x, b.getPosition().y-speedBullet);
        }else if(shooting){
            b.setVisible(false);
    }
        //This piece of code sets visible each new enemy
        if ((count > 250) && (count % (randomGenerator.nextInt(100-1) + 1) == 0)){
            enemiesList.get(enemyCount).setVisibility(true);
            if(enemyCount< enemiesList.size()-1){
                enemyCount++;
            }
        }

        count++;

        //This loop checks the enemy does not fall out of the screen.
        for(Enemy enemy: enemiesList){
            if(enemy.isMoving()){
                if(enemy.isMovingRight()){
                     enemy.setPosition(enemy.getPosition().x + speedEnemyX, enemy.getPosition().y); //MOVE RIGHT
                }
                if(!enemy.isMovingRight()){
                     enemy.setPosition(enemy.getPosition().x - speedEnemyX, enemy.getPosition().y);    //MOVE LEFT
                }
                if (enemy.getPosition().x <= 0){    //LEFT LIMIT BOUND
                    enemy.setMovingRight(true);
                }
                if (enemy.getPosition().x >= 450){    //RIGHT LIMIT BOUND
                    enemy.setMovingRight(false);
                }
            }
        }

        //GAME OVER SPLASH
        if(gs.getLives()==0){
                t.stop();
                splash.setVisible(true);

        }

        //CHECK IF ALL ENEMIES ARE KILLED
        boolean winner = true;
        for(Enemy checkE:enemiesList){
            if(checkE.isMoving()){
                winner = false;
            }
        }
         //WINNER SPLASH
        if(t.isRunning()&&winner){
            lblSplash.setText("LEVEL COMPLETED!");
            splash.setVisible(true);
        }
    }

   /**
   * This function checks if any enemy has been hit with the
   * bullet.
   */
    public void checkEnemyHit(){

        if(b != null){
            for (Enemy enemy:enemiesList){

                //Check on all enemies are not in the bullet area overlapping.
                if((b.getPosition().x+16>=enemy.getPosition().x)&&
                        (b.getPosition().x<=enemy.getPosition().x+32)&&
                        (b.getPosition().y+16>=enemy.getPosition().y)&&
                        (b.getPosition().y<=enemy.getPosition().y+32)&&
                        (b.isMoving())&&(enemy.isMoving())){

                        //if its hit change values of that specific enemy
                        enemy.setImg(new ImageIcon("src\\icons\\bones.png"));
                        enemy.setMoving(false);
                        b.setMoving(false);

                        //add a point
                        gs.setPoints(gs.getPoints()+1);
                        lbl2.setText("SCORE: "+gs.getPoints()+"/"+NUMBER_OF_ENEMIES);
                }
            }
        }
    }

     /**
   * This function checks if any enemy
   * reaches the baseline.
```

```
*/
 public void checkDamage(){                                        //HIT

     for(Enemy enemy:enemiesList){
         //Enemy reaches the baseline change the image to a splash
         if(enemy.getPosition().y+32>=p.getPosition().y){
             enemy.setImg(new ImageIcon("src\\icons\\splash.png"));
             enemy.setMoving(true);

         }
         //After a few ticks set the enemy to invisible. And take a life of the board
         if(enemy.getPosition().y+10>=p.getPosition().y){
             enemy.setMoving(false);
             if(enemy.isVisibility()){
                 gs.setLives(gs.getLives()-1);
                 lbl3.setText("LIVES: "+gs.getLives());
             }
             enemy.setVisibility(false);
         }
     }
 }

/**
* This lister runs the keyboard events.
*/
 @Override
 public void keyPressed(KeyEvent e) {
    int code = e.getKeyCode();

    //right  arrow presed on to move player right
    if(code == KeyEvent.VK_RIGHT){
         if(pX+65 < this.getWidth()){
             pX += speedPlayer;
             p.setPosition(pX,pY);
         }
    }
    //LEFT arrow pressed on to move player left
    if(code == KeyEvent.VK_LEFT){
        if(pX > 0){
            pX -= speedPlayer;
            p.setPosition(pX,pY);
        }
    }
```

```
        //Space bar to shoot a new bullet
        if(code == KeyEvent.VK_SPACE){
            b = new Bullet(true,pX,pY);
            b.setPosition(pX+25, pY);
            shooting = true;
            b.setVisible(true);
        }
    }

    //NOT USED
    @Override
    public void keyReleased(KeyEvent e) {}

    //NOT USED
    @Override
    public void keyTyped(KeyEvent e) {}
}
```

Splash.java

```
package views;
/**
 *  This class creates an Panel where it will announce
 *  if the player wins or looses.
 *
 * @author Leandro Ruiz Boyle
 */
public class Splash extends javax.swing.JPanel {

    public Splash() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();

        setBackground(new java.awt.Color(102, 102, 102));
        setAlignmentX(350.0F);
        setMaximumSize(new java.awt.Dimension(500, 51));
        setMinimumSize(new java.awt.Dimension(500, 51));
        setPreferredSize(new java.awt.Dimension(500, 51));

        jLabel1.setFont(new java.awt.Font("Gill Sans Ultra Bold", 1, 18)); // NOI18N
        jLabel1.setForeground(new java.awt.Color(255, 255, 255));
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("GAME OVER");

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap(48, Short.MAX_VALUE)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 391,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(61, 61, 61))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE, 29, Short.MAX_VALUE)
                .addContainerGap())
        );
    }// </editor-fold>


    // Variables declaration - do not modify
    private javax.swing.JLabel jLabel1;
    // End of variables declaration
}
```