**Man Hunter Game**

**Session 2023 – 2027**

**Submitted by:**

Malik Sohaib Ahmad 2023-CS-71

**Supervised by:**

Ma’am Maida Shahid and Prof. Laeeq Khan

**Course:**

CSC-103L Object Oriented Programming

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Table of Contents**

* Description:

Contributes toward CS

Why Shooting

* Characters

Player

Enemy(Vertical)

Enemy(Horizontal)

Enemy (Zig Zag)

* **Description:**

Man Hunter is a user friendly game. It is built for the purpose to learn OOP and How to implement it into games. This can serve as a base for learning how to develop games.

* + - **Contribution towards CS:**

Man Hunter teaches us the concepts of OOP. In this we implement Inhertiance, Interfaces and many other concepts of OOP.

* + - **Why Man Hunter:**

As this game is most related to space shooter and all the concepts of OOP are being Implemented. Another reason for choosing this project is due lack of expertise in game development.

.

* + - **What to Expect:**

This game serves as a base for the journey to develop more games based on the same Principles.

* **User Type:**
  + - **Player:**

In this game the player is able to move with the help of arrow keys. He can also shoot with the help of space bar.

* + - **Enemy (Vertical):**

In this game the Vertical Enemy is able to move Vertically. He can also shoot after a fixed interval of Time.

* + - **Enemy (Horizontal):**

In this game the Horizontal Enemy is able to move Horizontally. He can also shoot after a fixed interval of Time.

.

* + - **Enemy (Zig Zag):**

In this game the Zig Zag Enemy is able to move Zig Zag. He can also shoot after a fixed interval of Time.

* **Wireframe:**





using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

namespace Backend.BL

{

public class Game

{

List<GameObject> gameObjectsList;

Form Container;

Collision Collision ;

public Game(Form Conatiner)

{

gameObjectsList = new List<GameObject>();

this.Container = Conatiner;

Collision = new Collision();

}

public void AddGameObjects(GameObject myobject)

{

GameObject gameObject = myobject;

this.Container.Controls.Add(gameObject.getPictureBox());

gameObjectsList.Add(gameObject);

}

public void removeGameObjects(GameObject myobject)

{

GameObject gameObject = myobject;

this.Container.Controls.Remove(gameObject.getPictureBox());

gameObjectsList.Remove(gameObject);

}

public void Update()

{

List<GameObject> objectsToRemove = new List<GameObject>();

foreach (GameObject go in gameObjectsList)

{

go.Update();

Collision.CheckCollisionOfPlayer(gameObjectsList, this.Container, objectsToRemove);

}

foreach (GameObject go in gameObjectsList)

{

go.Update();

Collision.CheckCollisionOfEnemy(gameObjectsList, this.Container, objectsToRemove);

}

foreach (GameObject objToRemove in objectsToRemove)

{

gameObjectsList.Remove(objToRemove);

this.Container.Controls.Remove(objToRemove.getPictureBox());

}

}

public void PlayerFire(Image Img)

{

int Top = 0;

int Left=0;

Point Boundary = new System.Drawing.Point(this.Container.Width, this.Container.Height);

for(int i=0;i<gameObjectsList.Count;i++)

{

if (gameObjectsList[i].GetObjectType()==ObjectType.player)

{

Top = gameObjectsList[i].GetTop();

Left = gameObjectsList[i].GetLeft();

Top -= 50;

Left += 40;

break;

}

}

GameObject gameObject = new GameObject(this.Container, Img, Left, Top, new PlayerBullet(5, Boundary), ObjectType.playerBullet);

this.Container.Controls.Add(gameObject.getPictureBox());

gameObjectsList.Add(gameObject);

}

public void DeletePlayerFire()

{

for (int i = 0; i < gameObjectsList.Count; i++)

{

if (gameObjectsList[i].GetObjectType() == ObjectType.playerBullet)

{

GameObject gameObject = gameObjectsList[i];

int Y = gameObjectsList[i].GetTop();

if (Y == this.Container.Top)

{

this.Container.Controls.Remove(gameObject.getPictureBox());

gameObjectsList.Remove(gameObject);

}

}

}

}

public void EnemyFire(Image Img)

{

Point Boundary = new System.Drawing.Point(this.Container.Width, this.Container.Height);

for (int i=0;i<gameObjectsList.Count;i++)

{

if (gameObjectsList[i].GetObjectType()==ObjectType.enemy)

{

int Top = gameObjectsList[i].GetTop();

int Left = gameObjectsList[i].GetLeft();

Top += 120;

Left += 5;

GameObject gameObject = new GameObject(this.Container,Img,Left,Top,new Bullet(5,Boundary),ObjectType.enemyBullet);

this.Container.Controls.Add(gameObject.getPictureBox());

gameObjectsList.Add(gameObject);

}

}

}

public void DeleteFire()

{

for(int i=0; i<gameObjectsList.Count;i++)

{

if (gameObjectsList[i].GetObjectType()==ObjectType.enemyBullet)

{

GameObject gameObject = gameObjectsList[i];

int Y = gameObjectsList[i].GetTop();

if(Y<= 0)

{

this.Container.Controls.Remove(gameObject.getPictureBox());

gameObjectsList.Remove(gameObject);

}

}

}

}

}

}

using Backend.Interface;

using System.Drawing;

using System.Windows.Forms;

namespace Backend.BL

{

public class GameObject

{

private readonly IMotion Motion;

private readonly PictureBox PB;

private readonly ObjectType objectType;

private readonly Form Conatiner;

private readonly IMotion move;

public GameObject(PictureBox pb, IMotion motion)

{

PB = pb;

Motion = motion;

}

public GameObject(Form Conatiner, Image image, int left, int top, IMotion motion, ObjectType objectType)

{

this.Conatiner = Conatiner;

PB = new PictureBox

{

Image = image,

Left = left,

Top = top,

BackColor = Color.Transparent

};

Motion = motion;

this.objectType = objectType;

if (objectType == ObjectType.enemy)

{

PB.Width = 62;

PB.Height = 92;

PB.SizeMode = PictureBoxSizeMode.StretchImage;

}

else if (objectType == ObjectType.player)

{

PB.Width = 95;

PB.Height = 105;

PB.SizeMode = PictureBoxSizeMode.StretchImage;

}

else if (objectType == ObjectType.enemyBullet)

{

PB.Height = 62;

PB.Width = 30;

PB.SizeMode = PictureBoxSizeMode.StretchImage;

}

else if (objectType == ObjectType.playerBullet)

{

PB.Width = 20;

PB.Height = 50;

PB.SizeMode = PictureBoxSizeMode.StretchImage;

}

this.Conatiner.Controls.Add(PB);

}

public void Update()

{

PB.Location = Motion.Move(PB.Location);

}

public PictureBox getPictureBox()

{

return PB;

}

public PictureBox GetEnmies()

{

return objectType == ObjectType.enemy ? PB : null;

}

public ObjectType GetObjectType()

{

return objectType;

}

public int GetLeft()

{

return PB.Left;

}

public int GetTop()

{

return PB.Top;

}

public void SetLeft(int val)

{

PB.Left = val;

}

public void SetTop(int val)

{

PB.Top = val;

}

}

}