**Guardians of Galaxy**

Contents

[Game Description: 2](#_Toc106650705)

[Characters Description: 2](#_Toc106650706)

[Rules and Instruction: 2](#_Toc106650707)

[Goal of Game: 3](#_Toc106650708)

[Features of Framework: 3](#_Toc106650709)

[Sequence Diagram: 3](#_Toc106650710)

[Class Diagram: 5](#_Toc106650711)

[How to use Framwork: 5](#_Toc106650712)

[Movements 6](#_Toc106650713)

[Firing 7](#_Toc106650714)

[Collision 8](#_Toc106650715)

[Full code 10](#_Toc106650716)

# 

# Game Description:

I make “Guardians of Galaxy” Game. In Which Player Have three Lives. Proper Health System for Both Enemies and Player. And Proper Score System is Implemented. In this Game 7 Types of Small Enemies Implemented and 2 Types of Big (Main) Enemies Implemented. The Goal Of the Game is That You have to steal the Infinity Stone from Enemies.

# Characters Description:

In Game there is one Player, and 9 types of Enemies. In 9 Enemies, two are Main Enemies and 7 are small Enemies whose Protect the Infinity Stone From Player. And Start Firing if One of the Main Enemy is die.

* Player
* 2 Main Enemies
* 7 Small Enemies

# Rules and Instruction:

* Left Arrow Key (Left Move)
* Right Arrow Key (Right Move)
* Up Arrow Key (Up Move)
* Down Arrow Key (Down Move)
* Space Key (Fire)
* Enemies Fire (Randomly)

You can Play this Game according to Given Rules. In Game if You Save Your life try to Away from Enemies and Enemies Fire. They can Damage you. And Game Over After three Player Lives.

# Goal of Game:

The goal of Game is To Steal Infinity Stone Form Enemies.

# Features of Framework:

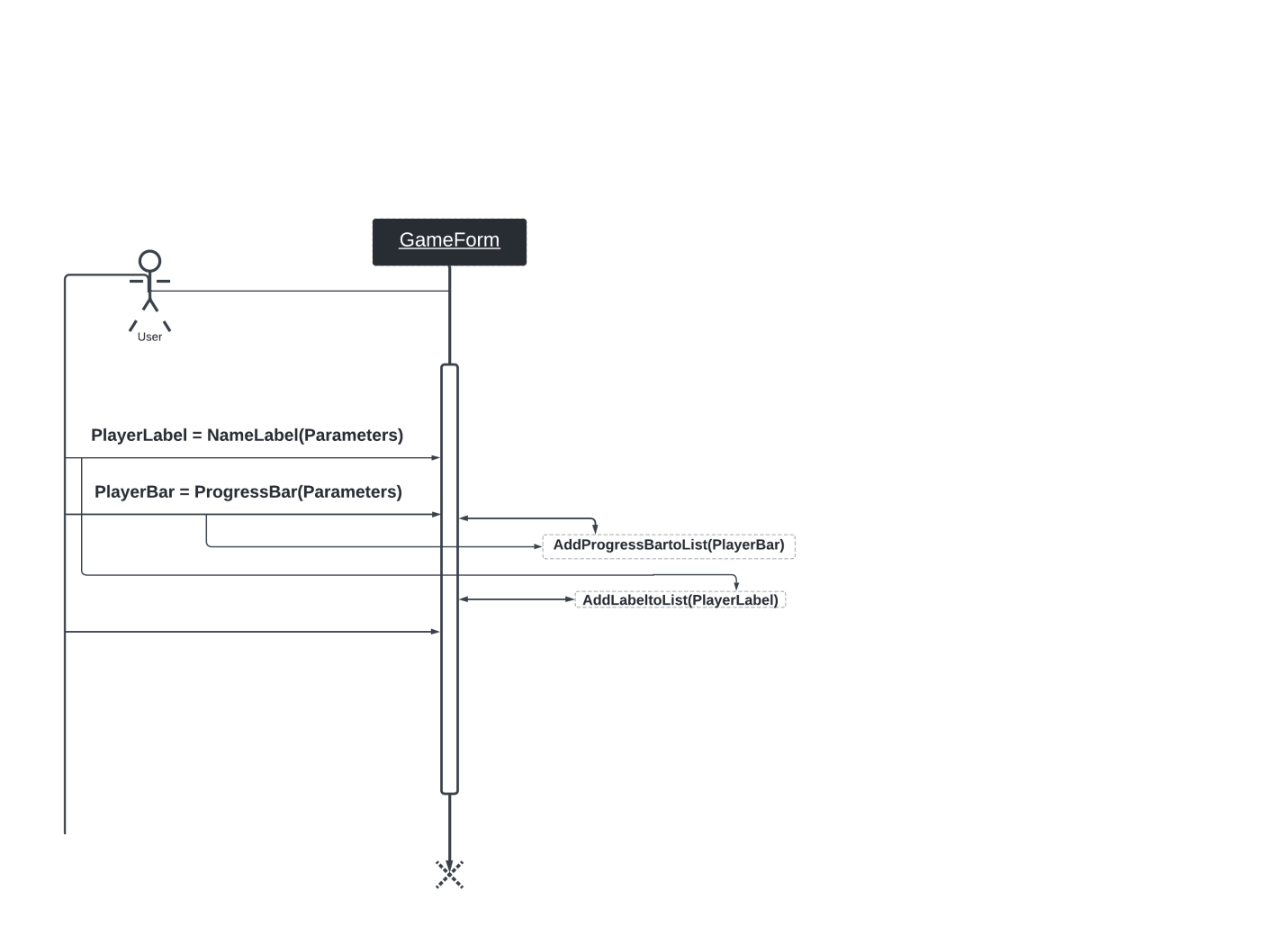
* Movement
* Collision
* Firing

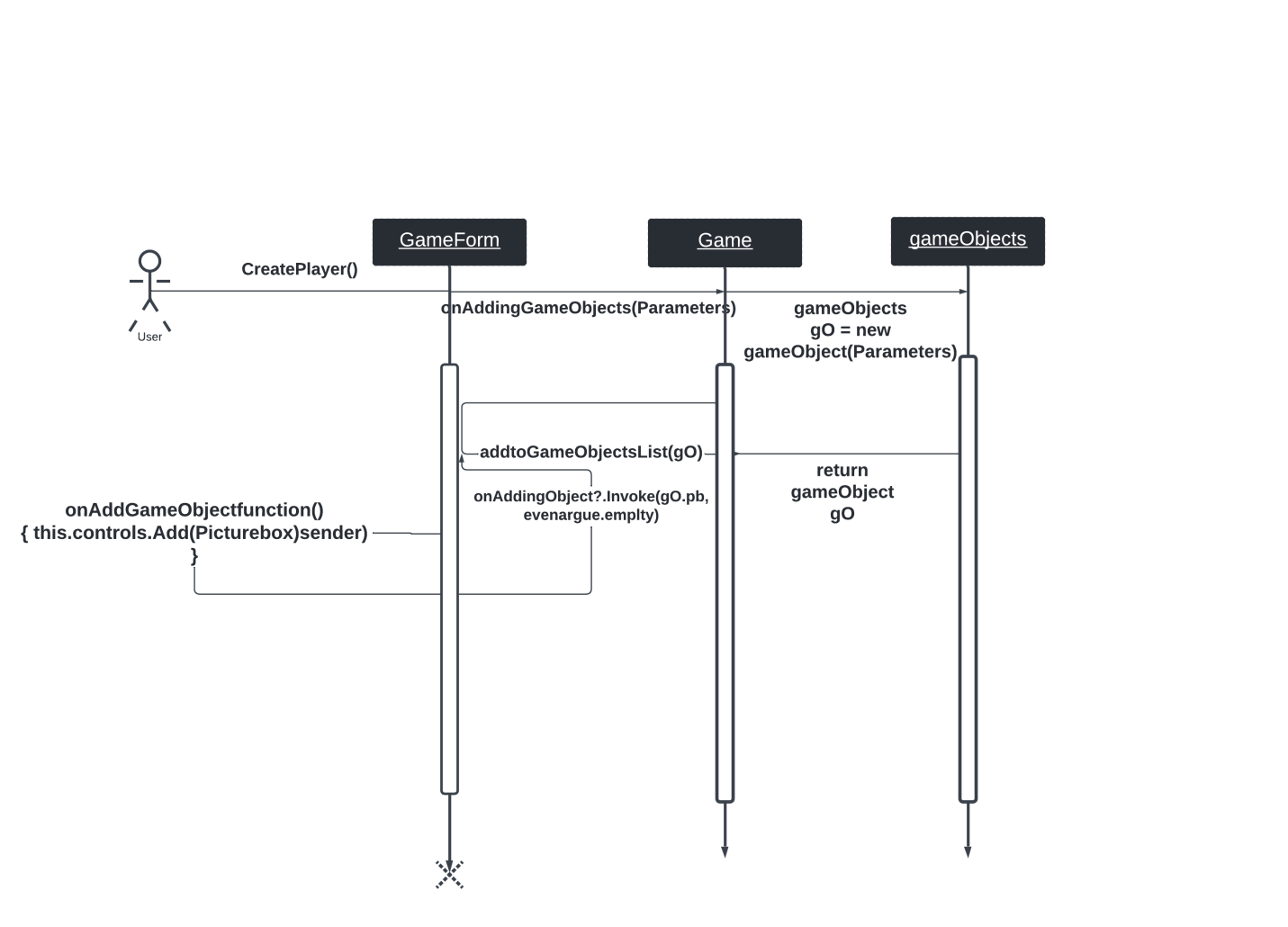
You Can Also Extent the Functionality of Framework.

* For Example You Want to Make Your Separate Class of Unique Movement First You have to Create Class and then You Inherit this Class With IMovement that can Help You to Extent Functionality of Movement according to You Requirements
* For Example You Want to Make Your Separate Class of Unique Fire, then First Make a Class and Then Inherit Your Class with IFire Class that I Implement in My framework and in this way you would be able to extend the functionality of Fire Framework and Use it according to Your Way.
* For Example You Want to Perform different type Action on Your Collisions In My framework I provide you “Enemy Fire to Player Collision”, “Player fire to Enemy Collision”, Enemy fire to Player Collision”, “Player or Enemy Collision”, and many more. You have Excess to Make Your own action on that Collisions. First Make You separate Class to type of Collision action and then inherit your Class with ICollision that I provided you in my Framework.

# Sequence Diagram:

**Form1:**

****

****

# Class Diagram:

//

# How to use Framwork:

If You Want to use my framework then first add it into your references. And then include their Libraries in your Form. And then You Access my Framework. And You Also Extend the Functionality of My Framework. So lets start how you can use my framework First if You Want To Make an Object then make it as given below:

“Game gO = new Game();

gO.AddGameObject(image,Top,Left,Size,ObjectType,IMovement);”

There is need to Pass these type of Parameters to in Sequence if You Add Object. Make the Game Object Global in Your Form and then You are Able to Access Limited Functions of Game Class.

# Movements

* **For Example if You want Add Movement** then in Previous example Make constructor of that Movement Class at the Place IMovement and then Pass the Required Parameters in Sequence then Your Object automatically access the My Movement Functionality and start to Move in the Direction whose class you can Access

For Example:

gO.AddGameObjects(Properties.Resources.Enemy, 150, 10, new Size(150, 120), ObjectType.BigEnemy1, new Horizontal(5, "left", Boundary));

if You Want to Move Horizontal then Follow this Pattern.Note I Pass Values According to My Need You Also Pass Values According to You Need.

There is different Movement Like:

* Horizontal
* Vertical
* Up
* No Move
* You can Also Make You separate Movement by using My framework.
* For Example You Want to Make Your Separate Class of Unique Movement First You have to Create Class and then You Inherit this Class With IMovement that can Help You to Extent Functionality of Movement according to You Requirements . But When You Inherit IMovement Class then You need to Make Move Function in Your Class as I Make:

public Point Move(Point Location)

{

if(Location.X <0)

{

Direction = "right";

}

else if(Location.X + offset > Boundary.X)

{

Direction = "left";

}

if(Direction == "left")

{

Location.X -= Speed;

}

if(Direction == "right")

{

Location.X += Speed;

}

return Location;

}

I Make this Function according My Need You can Also Change this According to Your Need.

# Firing

* **Now Move to firing** If You Want To fire then first You Want To Make an KeyDown event of Your Form and Where Write a condition of Space Press as I Written:

if (e.KeyCode == Keys.Space)

{

Createbullet();

}

And in Createbullet Function I Write this:

gameObjects Player = gO.GiveCurrentPlayer("Player");

if (Player != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserRed01,Player.PictureBox.Location.Y, Player.PictureBox.Location.X + 37, new Size(15, 25), ObjectType.Fire, new UpFire(10, Boundary));

}

When You Want fire from Your Player You Need To Require Player CurrentLocation so that’s Why I Make Function

GiveCurrentPlayer(string Character);

If You Want Fire form Player then Pass Player string to function then it can returns you Current Player Object then You can easily access you Player Location and Set to You Fire Movement.

In the Same Way You can Also Make Fire Form enemy You Need To Strictly add the Object Type of Your Object its can Help You to Detect You Object;

For Example: for Fire write in this way while making object in Parameters

**“ObjectType.Fire”**

**“ObjectType.Player”**

**“Objecttype.BigEnemy1”**

**“Objecttype.Bigenemy2”**

* For Example You Want to Make Your Separate Class of Unique Fire, then First Make a Class and Then Inherit Your Class with IFire Class that I Implement in My framework and in this way you would be able to extend the functionality of Fire Framework and Use it according to Your Way.

In this You when You Inherit IFIre to Your Class You Need to Impement the Functions of IFire to You Fire Movement Class as Given Below:

public PictureBox Move(IGame game,PictureBox pb)

{

Point Location = pb.Location;

if(Location.Y <=0)

{

game.FireDeleteAction(pb);

}

Location.Y-= Speed;

pb.Location = Location;

return pb;

}

There is “game.FireDeleteAction(pb);” is Function that Implement in IGame and Game that Invoke when You Object Going to Out of Boundary and You Can Handle it in your Form As Given Below:

gO.onRemoveFire += new EventHandler(onDeleteFire);

onRemoveFire already Implement in Game Class You Just Need to Call it

and then onDeleteFire Function is Given Below:

private void onDeleteFire(object sender, EventArgs e)

{

gO.RemovePlayerFormList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

# Collision

* If You Want To Access My Collison Class then first add Library and then move to next Procedure Like:

I make Many Classes Name Like:

PlayerCollision : ICollision

EnemyfireColllision : ICollision

EnemyorPlayerFireAction : ICollision

PlayerCollision : ICollision

and many More that can Help You to Detect Your Collsion and Perform Action that You Want to Perform

if Want detect Collision then after making All Objects Make Collision Objects Like this:

Collisions c = new Collisions(ObjectType.Player, ObjectType.BigEnemy1, new PlayerCollision());

Collisions c1 = new Collisions(ObjectType.Player, ObjectType.Bigenemy2, new PlayerCollision());

Collisions c2 = new Collisions(ObjectType.Fire, ObjectType.BigEnemy1, new Firecollision());

Collisions c3= new Collisions(ObjectType.Fire, ObjectType.Bigenemy2, new Firecollision());

Collisions c4 = new Collisions(ObjectType.enemyfire, ObjectType.Player, new EnemyfireColllision());

Collisions c5 = new Collisions(ObjectType.enemyfire, ObjectType.Fire, new EnemyorPlayerFireAction());

gO.AddCollision(c);

gO.AddCollision(c1);

gO.AddCollision(c2);

gO.AddCollision(c3);

gO.AddCollision(c4);

gO.AddCollision(c5);

In Which every Objects Have Different Functionality of Collision You Can Also raise You Own Event to Perform Action there are different event in Game Class That Help You To Invoke Like this:

gO.onPlayerDie += new EventHandler(RemovePlayer);

gO.onRemoveFire += new EventHandler(onDeleteFire);

gO.onEnemyDie += new EventHandler(onDeleteEnemy);

gO.onEnemyDie2 += new EventHandler(onDeleteEnemy2);

gO.onEnemyfireDelete += new EventHandler(DeleteEnemyFire);

and You can Perform Different Action Like this:

private void onDeleteEnemy2(object sender, EventArgs e)

{

if (count1 >= 1 && count1 <= 10)

{

enemy2Bar.Value = enemy2Bar.Value - 10;

}

if (count1 == 10)

{

BigEnemyDie = true;

gO.RemoveEnemyFromList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

count1++;

}

I Write Conditions according to My Need to Delete Enemy from List You can Also Delete this in Your Way there are different Functions of Deleting Objects are available Like:

* RemoveEnemyFireFromList(PictureBox pb);
* RemovePlayerFormList(PictureBox pb);
* RemoveEnemyFireFromList(PictureBox pb);
* RemovePlayerFormList(PictureBox pb);
* For Example You Want to Perform different type Action on Your Collisions In My framework I provide you “Enemy Fire to Player Collision”, “Player fire to Enemy Collision”, Enemy fire to Player Collision”, “Player or Enemy Collision”, and many more. You have Excess to Make Your own action on that Collisions. First Make You separate Class to type of Collision action and then inherit your Class with ICollision that I provided you in my Framework. If You Inherit ICollision Class You need to Impletement its Function Like this :

public void PerformAction(IGame game, gameObjects source1, gameObjects source2)

{

gameObjects Player = new gameObjects();

gameObjects fire = new gameObjects();

if (source1.Otype1 == ObjectType.Player)

{

Player = source1;

}

if(source2.Otype1 == ObjectType.Player)

{

Player = source2;

}

if(source1.Otype1 == ObjectType.enemyfire)

{

fire = source1;

}

if(source2.Otype1 == ObjectType.enemyfire)

{

fire = source2;

}

game.PlayerDieAction(Player.PictureBox);

game.EnemeyfireDelete(fire.PictureBox);

}

It Can Help You To Implement You Actions.

**Hope You Like this**

# Full code

**IMovements Class:**

public interface IMovement

{

Point Move(Point Location);

}

**Horizontal Movements:**

public class Horizontal : IMovement

{

private int speed;

private string direction;

private Point boundary;

private int offset = 20;

public Horizontal(int speed, string direction, Point boundary)

{

Speed = speed;

Direction = direction;

Boundary = boundary;

}

public int Speed { get => speed; set => speed = value; }

public string Direction { get => direction; set => direction = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public Point Move(Point Location)

{

if(Location.X <0)

{

Direction = "right";

}

else if(Location.X + offset > Boundary.X)

{

Direction = "left";

}

if(Direction == "left")

{

Location.X -= Speed;

}

if(Direction == "right")

{

Location.X += Speed;

}

return Location;

}

}

}

**Vertical Movement:**

public class Vertical : IMovement

{

private int speed;

private string direction;

private Point boundary;

private int offset = 100;

public Vertical(int speed, string direction, Point boundary)

{

Speed = speed;

Direction = direction;

Boundary = boundary;

}

public int Speed { get => speed; set => speed = value; }

public string Direction { get => direction; set => direction = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public Point Move(Point Location)

{

if (Location.Y <= 0)

{

Direction = "down";

}

if (Location.Y + offset >= Boundary.Y)

{

Direction = "up";

}

if (Direction == "up")

{

Location.Y -= Speed;

}

if (Direction == "down")

{

Location.Y += Speed;

}

return Location;

}

}

**KeyBoardMovement:**

public class Keyboard : IMovement

{

private int speed;

private string KeyAction;

private Point boundary;

private int offset = 100;

public Keyboard(int speed, Point boundary)

{

Speed = speed;

Boundary = boundary;

KeyAction = null;

}

public int Speed { get => speed; set => speed = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public string KeyAction1 { get => KeyAction; set => KeyAction = value; }

public void KeyPressedByME(Keys KeyCode)

{

if(KeyCode == Keys.Up)

{

KeyAction = "up";

}

if(KeyCode == Keys.Down)

{

KeyAction = "down";

}

if(KeyCode == Keys.Left)

{

KeyAction = "left";

}

if(KeyCode == Keys.Right)

{

KeyAction = "right";

}

}

public Point Move(Point Location)

{

if (KeyAction != null)

{

if (KeyAction == "left")

{

if (Location.X > 0)

{

Location.X -= Speed;

}

}

if (KeyAction == "right")

{

if (Location.X + offset <= Boundary.X)

{

Location.X += Speed;

}

}

if (KeyAction == "up")

{

if (Location.Y > 0)

{

Location.Y -= Speed;

}

}

if (KeyAction == "down")

{

if (Location.Y < Boundary.Y)

{

Location.Y += Speed;

}

}

KeyAction = null;

}

return Location;

}

}

**Up Movements:**

private int speed;

private Point boundary;

public Up(int speed, Point boundary)

{

Speed = speed;

Boundary = boundary;

}

public int Speed { get => speed; set => speed = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public Point Move(Point Location)

{

if (Location.Y >= 0)

{

Location.Y-=Speed;

}

return Location;

}

}

**Fire**

**UpFire:**

public class UpFire : IFire

{

private int speed;

private Point boundary;

public UpFire(int speed, Point boundary)

{

Speed = speed;

Boundary = boundary;

}

public int Speed { get => speed; set => speed = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public PictureBox Move(IGame game,PictureBox pb)

{

Point Location = pb.Location;

if(Location.Y <=0)

{

game.FireDeleteAction(pb);

}

Location.Y-= Speed;

pb.Location = Location;

return pb;

}

}

**Down fire:**

public class DownFire : IFire

{

private int speed;

private Point boundary;

private int offset = 100;

public DownFire(int speed, Point boundary)

{

Speed = speed;

Boundary = boundary;

}

public int Speed { get => speed; set => speed = value; }

public Point Boundary { get => boundary; set => boundary = value; }

public PictureBox Move(IGame game, PictureBox pb)

{

Point Location = pb.Location;

if (Location.Y + offset >= Boundary.Y)

{

game.FireDeleteAction(pb);

}

Location.Y += Speed;

pb.Location = Location;

return pb;

}

}

**IFire:**

public interface IFire

{

PictureBox Move(IGame game,PictureBox Pb);

}

**Collision**

**ICollision:**

public interface ICollision

{

void PerformAction(IGame game, gameObjects source1, gameObjects source2);

}

**Collision:**

public class Collisions

{

private ObjectType g1;

private ObjectType g2;

private ICollision behavior;

public Collisions(ObjectType g1, ObjectType g2, ICollision behavior)

{

G1 = g1;

G2 = g2;

Behavior = behavior;

}

public ObjectType G1 { get => g1; set => g1 = value; }

public ObjectType G2 { get => g2; set => g2 = value; }

public ICollision Behavior { get => behavior; set => behavior = value; }

}

**PlayerfireCollison:**

public class Firecollision : ICollision

{

public void PerformAction(IGame game, gameObjects source1, gameObjects source2)

{

gameObjects Enemy = new gameObjects();

gameObjects fire = new gameObjects();

if (source1.Otype1 == ObjectType.BigEnemy1 || source1.Otype1 == ObjectType.Bigenemy2)

{

Enemy = source1;

}

if(source2.Otype1 == ObjectType.BigEnemy1 || source2.Otype1 == ObjectType.Bigenemy2)

{

Enemy = source2;

}

if(source1.Otype1 == ObjectType.Fire)

{

fire = source1;

}

if(source2.Otype1 == ObjectType.Fire)

{

fire = source2;

}

if (Enemy.Otype1 == ObjectType.BigEnemy1)

{

game.Enemy1DieAction(Enemy.PictureBox);

}

if(Enemy.Otype1 == ObjectType.Bigenemy2)

{

game.Enemy2DieAction(Enemy.PictureBox);

}

game.FireDeleteAction(fire.PictureBox);

}

}

**EnemyorPlayerfirecollision:**

public class EnemyorPlayerFireAction : ICollision

{

public void PerformAction(IGame game, gameObjects source1, gameObjects source2)

{

gameObjects PlayerFire = new gameObjects();

gameObjects Enemyfire = new gameObjects();

if (source1.Otype1 == ObjectType.Player)

{

PlayerFire = source1;

}

if (source2.Otype1 == ObjectType.Player)

{

PlayerFire = source2;

}

if (source1.Otype1 == ObjectType.enemyfire)

{

Enemyfire = source1;

}

if (source2.Otype1 == ObjectType.enemyfire)

{

Enemyfire = source2;

}

game.FireDeleteAction(PlayerFire.PictureBox);

game.EnemeyfireDelete(Enemyfire.PictureBox);

}

}

**EnemyFirecollision:**

public class EnemyorPlayerFireAction : ICollision

{

public void PerformAction(IGame game, gameObjects source1, gameObjects source2)

{

gameObjects PlayerFire = new gameObjects();

gameObjects Enemyfire = new gameObjects();

if (source1.Otype1 == ObjectType.Player)

{

PlayerFire = source1;

}

if (source2.Otype1 == ObjectType.Player)

{

PlayerFire = source2;

}

if (source1.Otype1 == ObjectType.enemyfire)

{

Enemyfire = source1;

}

if (source2.Otype1 == ObjectType.enemyfire)

{

Enemyfire = source2;

}

game.FireDeleteAction(PlayerFire.PictureBox);

game.EnemeyfireDelete(Enemyfire.PictureBox);

}

}

**PlayerCollision:**

public void PerformAction(IGame game, gameObjects source1 , gameObjects source2)

{

gameObjects player;

if(source1.Otype1 == ObjectType.Player)

{

player = source1;

}

else

{

player = source2;

}

game.PlayerDieAction(player.PictureBox);

}

**GameForm 1:**

public partial class GameForm : Form

{

Game gO;

Point Boundary;

ProgressBar Playerbar;

ProgressBar enemy1Bar;

ProgressBar enemy2Bar;

Label PlayerLbl;

Label enemy1Lbl;

Label enemy2Lbl;

Label PlayerLives;

Label PlayerLiveName;

Label ScoreLabel;

Label Scorecount;

bool BigEnemyDie = false;

public GameForm()

{

InitializeComponent();

}

int count;

int count1;

int EnemyCount;

int PlayerCount;

public ProgressBar ProgressBar(int left, int top)

{

ProgressBar Playerbar = new ProgressBar();

Playerbar.Left = left;

Playerbar.Top = top;

Playerbar.Width = 200;

Playerbar.Height = 20;

Playerbar.Value = 100;

Playerbar.Show();

Playerbar.BackColor = Color.Red;

return Playerbar;

}

public Label NameLabel(string Name,string LblName,int Left,int Top,int width, int height,Color color)

{

Label Player = new Label();

Player.Name = Name;

Player.Text = LblName;

Player.Left = Left;

Player.Top = Top;

Player.Width = width;

Player.Height = height;

Player.BackColor = Color.Transparent;

Player.ForeColor = color;

return Player;

}

public void AddLabel(Label lbl)

{

this.Controls.Add(lbl);

}

public void AddProgressBar(ProgressBar bar)

{

this.Controls.Add(bar);

}

private void Form1\_Load(object sender, EventArgs e)

{

gO = new Game();

Playerbar = ProgressBar(this.Left+80, this.Top + 10);

PlayerLbl = NameLabel("PlayerLBL", "Player Health", this.Left, this.Top + 10,80,20, Color.GreenYellow);

enemy1Bar = ProgressBar(this.Left+80, this.Top + 35);

enemy1Lbl = NameLabel("EnemeyLbl", "Enemy1 Health", this.Left, this.Top + 35,80,20,Color.GreenYellow);

enemy2Bar = ProgressBar(this.Left + 80, this.Top + 60);

enemy2Lbl = NameLabel("EnemeyLbl", "Enemy2 Health", this.Left, this.Top + 60,80,20, Color.GreenYellow);

ScoreLabel = NameLabel("ScoreLbl", "SCORE",this.Right - 300,this.Top+10,100,20,Color.GreenYellow);

Scorecount = NameLabel("ScoreCountLbl", "0", this.Right - 250, this.Top + 10, 100, 20, Color.Red);

PlayerLiveName = NameLabel("LiveLbl", "PLAYER LIVES", this.Right - 150, this.Top + 10, 100, 20, Color.GreenYellow);

PlayerLives = NameLabel("LiveCountLbl", "3", this.Right - 60, this.Top + 10, 100, 20, Color.Red);

AddLabel(PlayerLbl);

AddProgressBar(Playerbar);

AddLabel(enemy1Lbl);

AddProgressBar(enemy1Bar);

AddLabel(enemy2Lbl);

AddProgressBar(enemy2Bar);

AddLabel(ScoreLabel);

AddLabel(Scorecount);

AddLabel(PlayerLives);

AddLabel(PlayerLiveName);

Scorecount.BringToFront();

PlayerLives.BringToFront();

gO.onAddObjects += new EventHandler(onAddgameObjects);

gO.onAddFire += new EventHandler(onAddingFire);

gO.onPlayerDie += new EventHandler(RemovePlayer);

gO.onRemoveFire += new EventHandler(onDeleteFire);

gO.onAddingSpace += new EventHandler(onAddingSpace);

gO.onEnemyDie += new EventHandler(onDeleteEnemy);

gO.onEnemyDie2 += new EventHandler(onDeleteEnemy2);

gO.onEnemyfireDelete += new EventHandler(DeleteEnemyFire);

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

CreatSpace();

CreatePlayer();

CreateEnemies();

Collisions c = new Collisions(ObjectType.Player, ObjectType.BigEnemy1, new PlayerCollision());

Collisions c1 = new Collisions(ObjectType.Player, ObjectType.Bigenemy2, new PlayerCollision());

Collisions c2 = new Collisions(ObjectType.Fire, ObjectType.BigEnemy1, new Firecollision());

Collisions c3= new Collisions(ObjectType.Fire, ObjectType.Bigenemy2, new Firecollision());

Collisions c4 = new Collisions(ObjectType.enemyfire, ObjectType.Player, new EnemyfireColllision());

Collisions c5 = new Collisions(ObjectType.enemyfire, ObjectType.Fire, new EnemyorPlayerFireAction());

gO.AddCollision(c);

gO.AddCollision(c1);

gO.AddCollision(c2);

gO.AddCollision(c3);

gO.AddCollision(c4);

gO.AddCollision(c5);

}

private void DeleteEnemyFire(object sender, EventArgs e)

{

gO.RemoveEnemyFireFromList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

private void onDeleteEnemy2(object sender, EventArgs e)

{

if (count1 >= 1 && count1 <= 10)

{

enemy2Bar.Value = enemy2Bar.Value - 10;

}

if (count1 == 10)

{

BigEnemyDie = true;

gO.RemoveEnemyFromList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

count1++;

}

private void onDeleteEnemy(object sender, EventArgs e)

{

if(count >= 1 && count<=10)

{

enemy1Bar.Value = enemy1Bar.Value - 10;

}

if (count == 10)

{

BigEnemyDie = true;

gO.RemoveEnemyFromList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

count++;

}

private void onAddingSpace(object sender, EventArgs e)

{

this.Controls.Add((PictureBox)sender);

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Space)

{

Createbullet();

}

else

{

gO.KeyPressed(e.KeyCode);

}

}

private void GameTimer\_Tick(object sender, EventArgs e)

{

gO.Update();

if(EnemyCount == 5)

{

CreateEnemyBullet();

EnemyCount = 0;

}

EnemyCount++;

}

private void onAddgameObjects(object sender, EventArgs e)

{

PictureBox pb = (PictureBox)sender;

this.Controls.Add(pb);

}

private void RemovePlayer(object sender, EventArgs e)

{

PlayerCount++;

if (PlayerCount >= 1 && PlayerCount <3)

{

CreatePlayer();

}

if (PlayerCount == 1)

{

PlayerLives.Text = "2";

Playerbar.Value -= 30;

}

if(PlayerCount == 2)

{

Playerbar.Value -= 30;

PlayerLives.Text = "1";

}

if(PlayerCount == 3)

{

Playerbar.Value -= 40;

PlayerLives.Text = "0";

}

this.Controls.Remove((PictureBox)sender);

}

private void onDeleteFire(object sender, EventArgs e)

{

gO.RemovePlayerFormList((PictureBox)sender);

this.Controls.Remove((PictureBox)sender);

}

private void onAddingFire(object sender, EventArgs e)

{

this.Controls.Add((PictureBox)sender);

}

public void CreatSpace()

{

gO.AddRaodObjects(GardiensOfGlaxy.Properties.Resources.Bakground, -Top, 0, new Size(this.Width, this.Height), new SpaceMovement(10,Boundary));

gO.AddRaodObjects(GardiensOfGlaxy.Properties.Resources.Bakground, 957, 0, new Size(this.Width, this.Height), new SpaceMovement(10, Boundary));

}

public void CreateEnemies()

{

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top, this.Width / 2 - 90, new Size(70, 60), ObjectType.enemy1, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.Enemy, 150, 10, new Size(150, 120), ObjectType.BigEnemy1, new Horizontal(5, "left", Boundary));

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.Enemy2, 250, this.Width, new Size(150, 120), ObjectType.Bigenemy2, new Horizontal(5, "right", Boundary));

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.Diamond2, this.Top, this.Width / 2, new Size(50, 50), ObjectType.Diamond, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top, this.Width / 2 + 90, new Size(70, 60), ObjectType.enemy2, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top + 40, this.Width / 2 + 150, new Size(70, 60), ObjectType.enemy3, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top + 40, this.Width / 2 - 150, new Size(70, 60), ObjectType.enemy4, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top, this.Width / 2 + 200, new Size(70, 60), ObjectType.enemy5, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top, this.Width / 2 - 200, new Size(70, 60), ObjectType.enemy6, new NoMove());

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.SmallEnemies, this.Top + 40, this.Width / 2, new Size(70, 60), ObjectType.enemy7, new NoMove());

}

public void CreatePlayer()

{

gO.AddGameObjects(GardiensOfGlaxy.Properties.Resources.PlayerShip\_\_2\_,this.Height-100, this.Width / 2, new Size(100, 90), ObjectType.Player, new Keyboard(10, Boundary));

}

public void CreateEnemyBullet()

{

gameObjects enemy = gO.GiveCurrentPlayer("Enemy1");

gameObjects enemy2 = gO.GiveCurrentPlayer("Enemy2");

gameObjects Senemy1 = gO.GiveCurrentPlayer("Senemy1");

gameObjects Senemy2 = gO.GiveCurrentPlayer("Senemy2");

gameObjects Senemy3 = gO.GiveCurrentPlayer("Senemy3");

gameObjects Senemy4 = gO.GiveCurrentPlayer("Senemy4");

gameObjects Senemy5 = gO.GiveCurrentPlayer("Senemy5");

gameObjects Senemy6 = gO.GiveCurrentPlayer("Senemy6");

gameObjects Senemy7 = gO.GiveCurrentPlayer("Senemy7");

if (enemy != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue14, enemy.PictureBox.Bottom, enemy.PictureBox.Location.X + 70, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (enemy2 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue14, enemy2.PictureBox.Bottom, enemy2.PictureBox.Location.X + 70, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (BigEnemyDie == true)

{

if (Senemy1 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy1.PictureBox.Bottom, Senemy1.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (Senemy2 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy2.PictureBox.Bottom, Senemy2.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (Senemy3 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy3.PictureBox.Bottom, Senemy3.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (Senemy4 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy4.PictureBox.Bottom, Senemy4.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (Senemy5 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy5.PictureBox.Bottom, Senemy5.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

if (Senemy6 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy6.PictureBox.Bottom, Senemy6.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

if (Senemy7 != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserBlue15, Senemy7.PictureBox.Bottom, Senemy7.PictureBox.Location.X + 30, new Size(15, 25), ObjectType.enemyfire, new DownFire(10, Boundary));

}

}

}

}

public void Createbullet()

{

gameObjects Player = gO.GiveCurrentPlayer("Player");

if (Player != null)

{

gO.AddFireObject(GardiensOfGlaxy.Properties.Resources.laserRed01, Player.PictureBox.Location.Y, Player.PictureBox.Location.X + 37, new Size(15, 25), ObjectType.Fire, new UpFire(10, Boundary));

}

}

}