

**Game Development Framework**

**Submitted By:**

**Asad Mehmood , 2020-CS-10**

**Submitted To:**

**Sir. Laeeq Khan Niazi**

**For fulfillment of**

**CS 162 Object Oriented Programming**

**Department of Computer Science**

**University of Engineering and Technology, Lahore**

**Problem Statement:**

* Add falling functionality, moving by keyboard keys, petrol movement and jumping movement for different objects (picturebox) such as enemies and other players.

**Solution (In Object oriented programming):**

**Demerits of the previous solution**

In the previous approach, solution was not interfaced. Classes and their functions were not handled with interface there was no restriction for the class.

**2nd Approach**

**Code:**

**Form1 Code:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

public partial class Form1 : Form

{

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

public Form1()

{

InitializeComponent();

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

}

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

{

GameObject player1Object = new GameObject(player1 , 5 , new LeftMovement());

GameObject player2Object = new GameObject(player2 , 6 , new RightMovement());

GameObject player3Object = new GameObject(player3, 6, new MoveWithKeyboard(this));

addObjects(player1Object);

addObjects(player2Object);

addObjects(player3Object);

}

private void addObjects(GameObject obj)

{

gameObjectList.Add(obj);

}

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

{

foreach(GameObject obj in gameObjectList)

{

obj.update();

}

}

}

}

**GameObject Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

class GameObject

{

PictureBox pictureBox;

int gravity;

Movement real\_movement;

public GameObject(PictureBox gameObj , int g , Movement movement)

{

pictureBox = gameObj;

gravity = g;

this.real\_movement = movement;

}

public void update()

{

real\_movement.moveObject(pictureBox, gravity);

}

}

}

**Movement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

class Movement

{

public virtual void moveObject(PictureBox pictureBox, int gravity)

{

}

}

}

**LeftMovement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

class LeftMovement : Movement

{

public override void moveObject(PictureBox pictureBox,int gravity)

{

pictureBox.Left -= gravity;

}

}

}

**RightMovement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

class RightMovement : Movement

{

public override void moveObject(PictureBox pictureBox,int gravity)

{

pictureBox.Left += gravity;

}

}

}

**MoveWithKeyboard Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework

{

class MoveWithKeyboard : Movement

{

Form1 form;

PictureBox pictureBox;

int gravity;

public MoveWithKeyboard(Form1 form)

{

this.form = form;

form.KeyDown += new KeyEventHandler(keyholder);

}

public override void moveObject(PictureBox pictureBox, int gravity)

{

this.pictureBox = pictureBox;

this.gravity = gravity;

}

private void keyholder(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Left)

{

this.pictureBox.Left -= gravity;

}

else if (e.KeyCode == Keys.Right)

{

this.pictureBox.Left += gravity;

}

}

}

}

**Solution of the previous problem:**

Now the new approach for this is that we have used interface. Now classes are restricted to inherit their interfaced parent class and define the function created in interfaced class. With this approach, we have also added two more features Petrol movement and Jumping Movement.

**Code:**

**Form1 Code:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

public partial class Form1 : Form

{

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

public Form1()

{

InitializeComponent();

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

}

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

{

GameObject player1Object = new GameObject(player1, 5, new LeftMovement());

GameObject player2Object = new GameObject(player2, 6, new RightMovement());

GameObject player3Object = new GameObject(player3, 6, new MoveWithKeyboard(this));

GameObject player4Object = new GameObject(player4, 6, new PetrolMovement());

GameObject player5Object = new GameObject(player5, 6, new Jumping(this));

addObjects(player1Object);

addObjects(player2Object);

addObjects(player3Object);

addObjects(player4Object);

addObjects(player5Object);

}

private void addObjects(GameObject obj)

{

gameObjectList.Add(obj);

}

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

{

foreach (GameObject obj in gameObjectList)

{

obj.update();

}

}

}

}

**GameObject Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

interface IGameObject

{

void update();

}

class GameObject : IGameObject

{

PictureBox pictureBox;

int gravity;

Movement real\_movement;

public GameObject(PictureBox gameObj, int g, Movement movement)

{

pictureBox = gameObj;

gravity = g;

this.real\_movement = movement;

}

public void update()

{

real\_movement.moveObject(pictureBox, gravity);

}

}

}

**Movement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

interface Movement

{

void moveObject(PictureBox pictureBox, int gravity);

}

}

**LeftMovement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

class LeftMovement : Movement

{

public void moveObject(PictureBox pictureBox, int gravity)

{

pictureBox.Left -= gravity;

}

}

}

**RightMovement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

class RightMovement : Movement

{

public void moveObject(PictureBox pictureBox, int gravity)

{

pictureBox.Left += gravity;

}

}

}

**MoveWithKeyboard Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

class MoveWithKeyboard : Movement

{

Form1 form;

PictureBox pictureBox;

int gravity;

public MoveWithKeyboard(Form1 form)

{

this.form = form;

form.KeyDown += new KeyEventHandler(keyholder);

}

public void moveObject(PictureBox pictureBox, int gravity)

{

this.pictureBox = pictureBox;

this.gravity = gravity;

}

private void keyholder(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Left)

{

this.pictureBox.Left -= gravity;

}

else if (e.KeyCode == Keys.Right)

{

this.pictureBox.Left += gravity;

}

else if (e.KeyCode == Keys.Up)

{

this.pictureBox.Top -= gravity;

}

else if (e.KeyCode == Keys.Down)

{

this.pictureBox.Top += gravity;

}

}

}

}

**PetrolMovement Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

class PetrolMovement : Movement

{

int verticalSpeed = 3;

public void moveObject(PictureBox pictureBox, int gravity)

{

pictureBox.Top += verticalSpeed;

if (pictureBox.Top < 81 || pictureBox.Top > 380)

{

verticalSpeed = -verticalSpeed;

}

}

}

}

**Jumping Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game\_Framework\_4

{

class Jumping : Movement

{

Form1 form;

PictureBox pictureBox;

int gravity;

int getJumping;

public Jumping(Form1 form)

{

this.form = form;

form.KeyDown += new KeyEventHandler(keyholder);

}

public void moveObject(PictureBox pictureBox, int gravity)

{

this.pictureBox = pictureBox;

this.gravity = gravity;

}

private void keyholder(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Space)

{

getJumping = this.pictureBox.Top;

this.pictureBox.Top = this.pictureBox.Top - 40;

}

this.pictureBox.Top = getJumping;

}

}

}

**Merit of the solution:**

With this approach, we basically have made interface in classes. This has made the classes to define the

Function created in interface classes. With this we have made our framework more efficient and classes are more punctual to define the interface function. Also, with the addition of two more movement Petrol and Jumping Movement, objects now just get the Movement object and start moving in the direction they get.