Developing a Framework for Platformer Type Game

Using C# and WinForms

Learning Objective

Develop a platformer game using win form and Object Oriented Programming.

Assessment

This project carry 20% marks toward the Theory Course. 40% marks toward the Lab Cours.

How to Pass the Project:CODING

1. Project Task shall be assigned daily basis. You need to complete the task using git hub repository and through daily commits.

How to Pass the Project:Documentation

2. Every step require a documentation that include

- Clearly mentioned Problem statement.
- Procedural Solution
    - Its sudo code/Algorithm
    - Demerit of the Solution
    - code
- OOP.
    - Design Decision if multiple OOP Way exist
    - UML Diagram
    - Merit/DeMerit
    - Code.

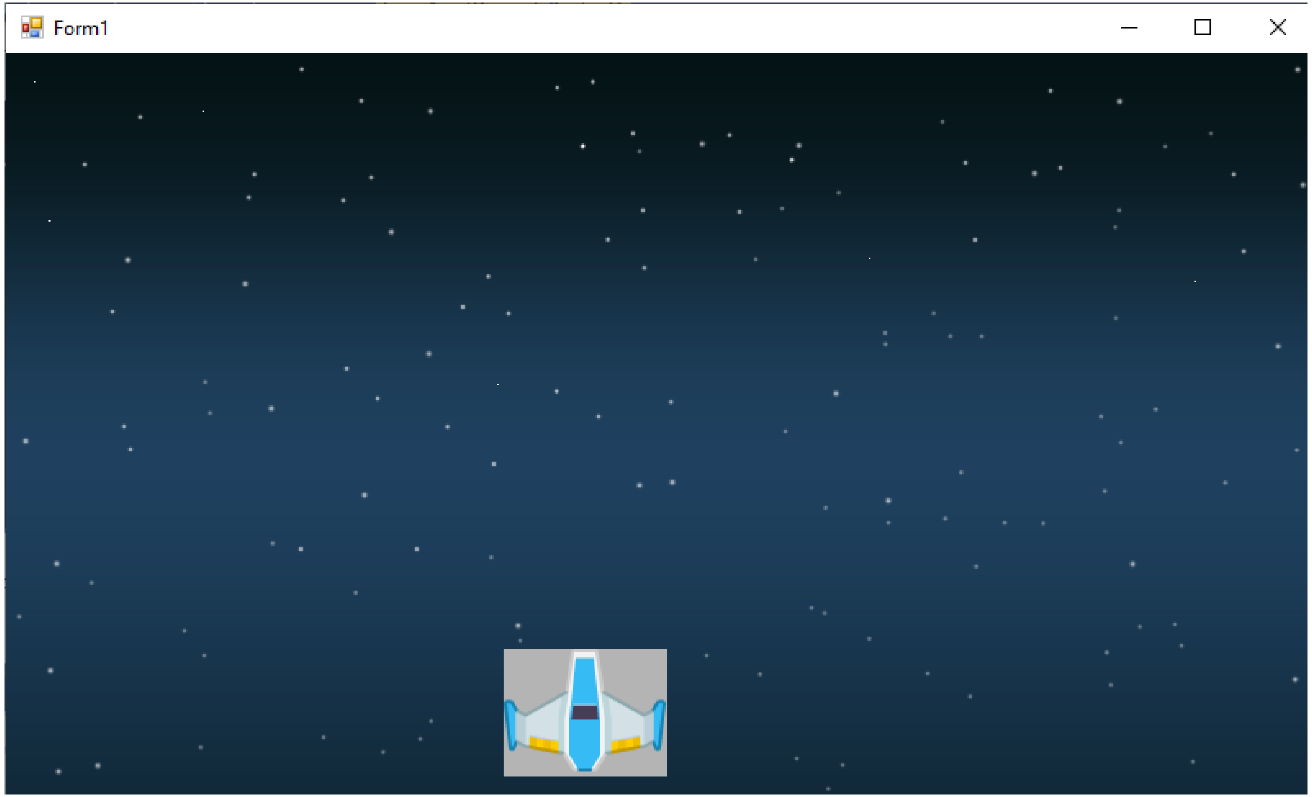
Setting up Environment.

Let make our view bit attractive.

- Setup the background of the form.
- Add picture Box that shows the main player

Note: you need to have png for the background and main player.

Setting up Environment.



Problem: Taking Asynchronous Input

Now you need to make the playership move with Arrow Keys.

We have already used EZInput. So You need to install NUGET Package.

Problem: Game Loop

Where to write code for KeyBoard Input ?

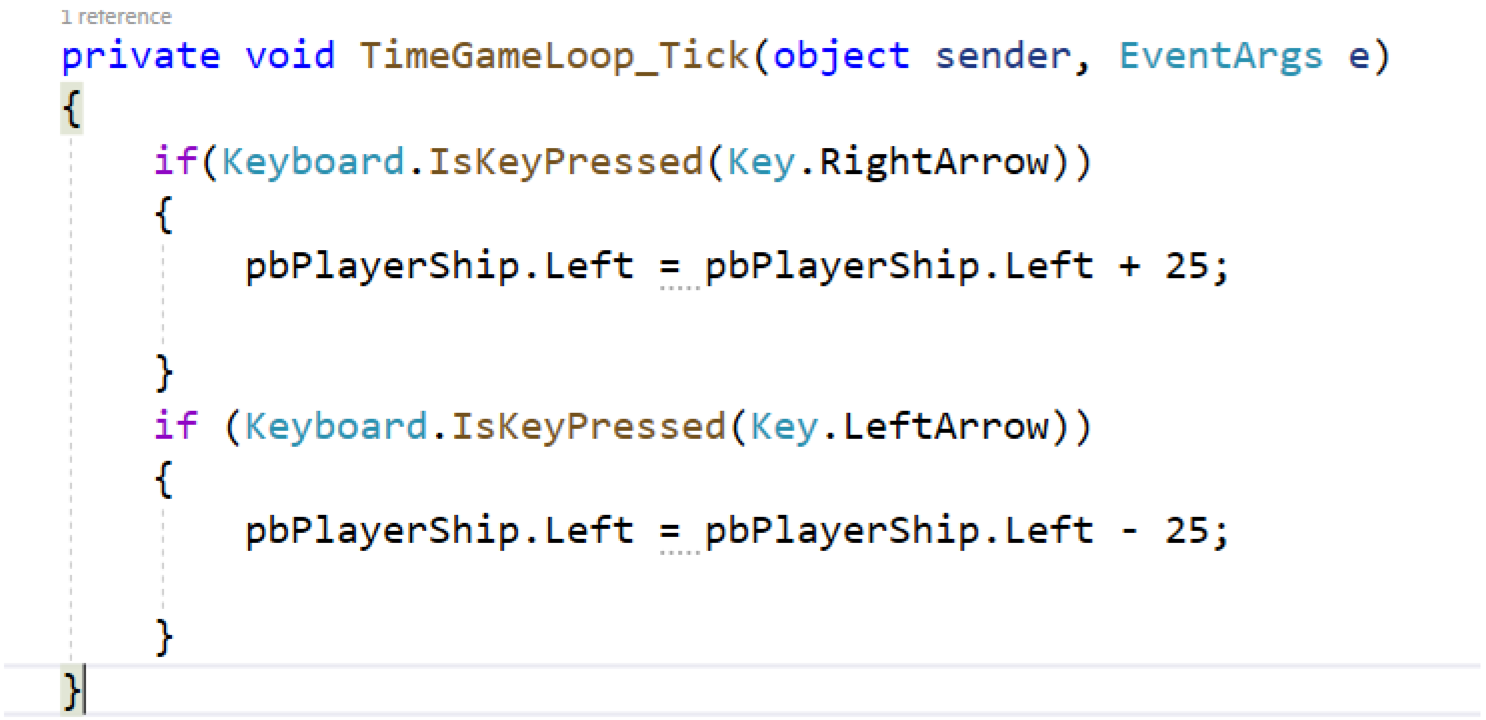
Problem: Game Loop

Where to write code for KeyBoard Input ?

We need to take a continuous input from the user. Therefore, we need to add timer control which will run its event after some time.

Therefore, we shall add a timer, set it interval time to 500 milliseconds and write the code inside the Tick Event of the timer

Problem: Game Loop



```
1 reference
private void TimeGameLoop_Tick(object sender, EventArgs e)
{
    if(Keyboard.IsKeyPressed(Key.RightArrow))
    {
        pbPlayerShip.Left = pbPlayerShip.Left + 25;

    }
    if (Keyboard.IsKeyPressed(Key.LeftArrow))
    {
        pbPlayerShip.Left = pbPlayerShip.Left - 25;

    }
}
```

Problem: Game Loop

We have implemented the player ship movement but there are two problems in it

1. Picture box showing filled area
2. Game is jittering.

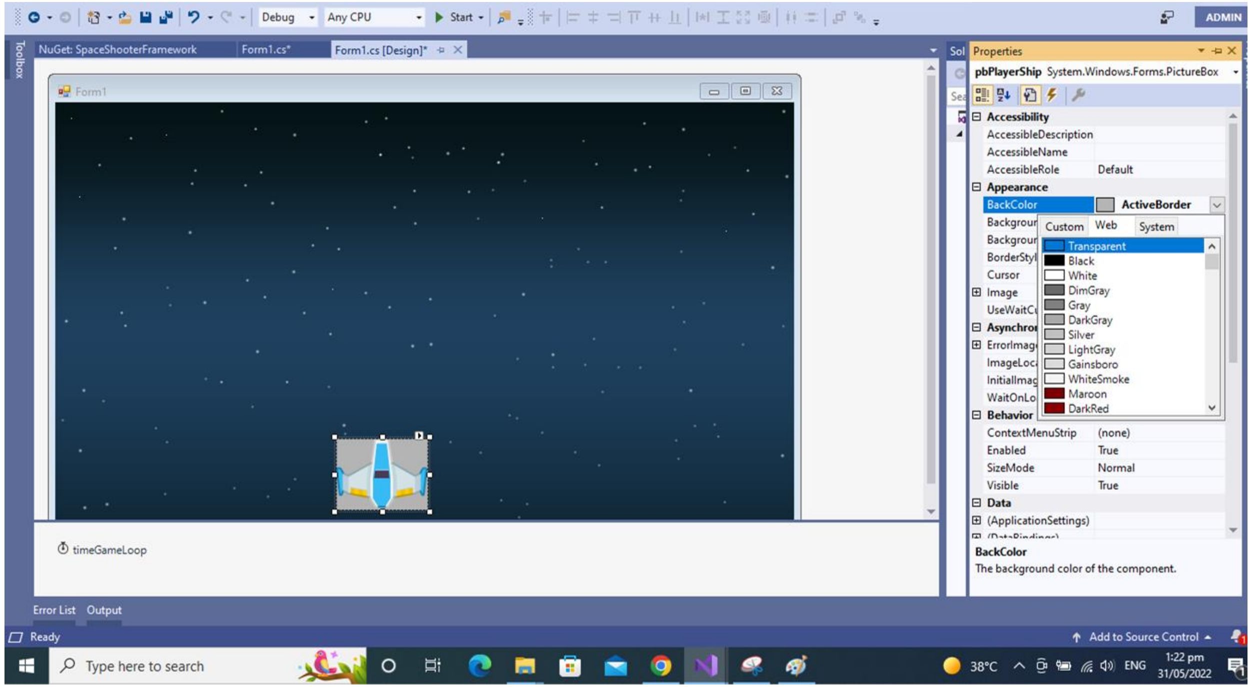
Problems

We have implemented the player ship movement but there are two problems in it

1. Picture box showing filled area
2. Game is jittering.

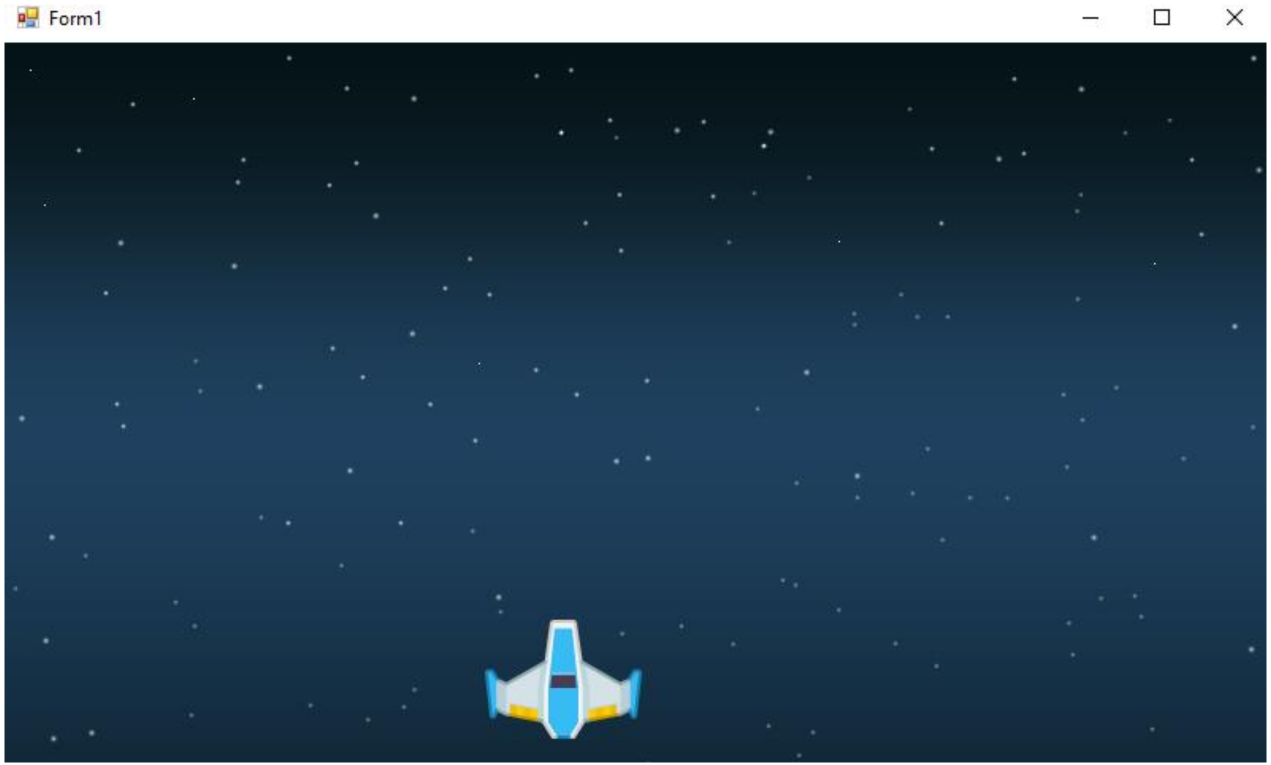
Problem: Transparent Background

Set the background property of image box to **Transparent**



Problem: Game Jitter

Set the form doubleBuffer true to improve the rendering of the game.



Add Firing Behaviour.

How to add fire behaviour for the playership.

Add Firing Behaviour.

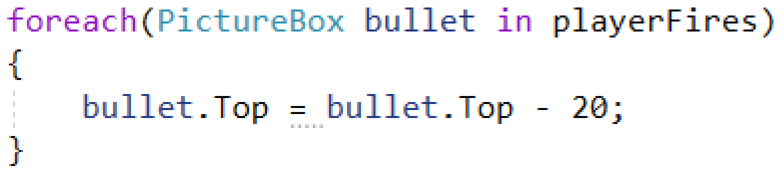
On spacebar press, we can create a pictureBox at runtime setup it location to the exactly the middle of the player ship and add into the list of fires so later on these fires may start moving.

Setting up Fires.

In main gameLoop, add following code.

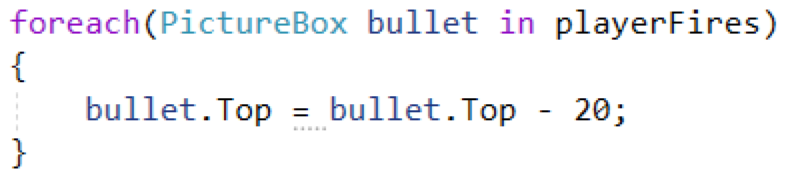
```
if (Keyboard.IsKeyPressed(Key.Space)) {
        Image fireImage = Properties.Resources.laserBlue01;
        PictureBox pbFire = new PictureBox();
        pbFire.Image = fireImage;
        pbFire.Width = fireImage.Width;
        pbFire.Height = fireImage.Height;
        pbFire.BackColor = Color.Transparent;
        pbFire.Left = pbPlayer.Left + (pbPlayer.Width / 2) - 5;
        pbFire.Top = pbPlayer.Top;
        playerFires.Add(pbFire);
        this.Controls.Add(pbFire);
 }
```

Moving fire Behaviour

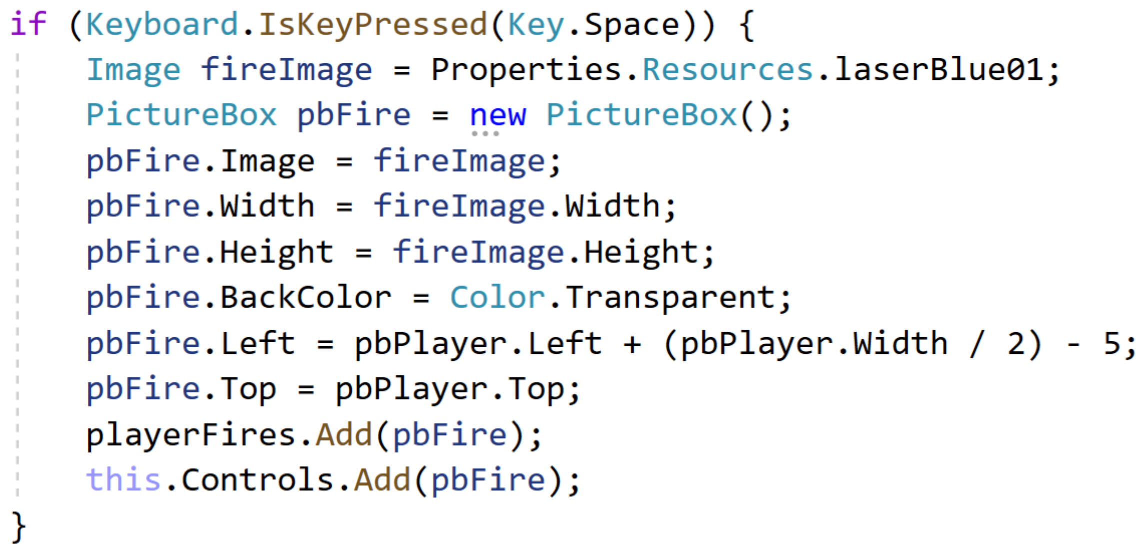


```
foreach(PictureBox bullet in playerFires)
{
    bullet.Top = bullet.Top - 20;
}
```

Can you Highlight problem.



```
foreach(PictureBox bullet in playerFires)
{
    bullet.Top = bullet.Top - 20;
}
```



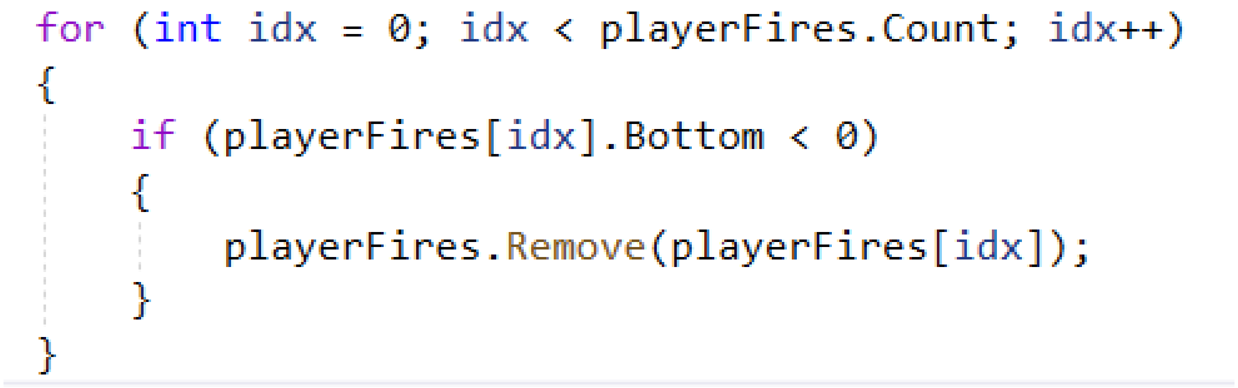
```
if (Keyboard.IsKeyPressed(Key.Space)) {
    Image fireImage = Properties.Resources.laserBlue01;
    PictureBox pbFire = new PictureBox();
    pbFire.Image = fireImage;
    pbFire.Width = fireImage.Width;
    pbFire.Height = fireImage.Height;
    pbFire.BackColor = Color.Transparent;
    pbFire.Left = pbPlayer.Left + (pbPlayer.Width / 2) - 5;
    pbFire.Top = pbPlayer.Top;
    playerFires.Add(pbFire);
    this.Controls.Add(pbFire);
}
```

Can you Highlight problem.

Here, we are adding fires and moving even if they are out of context (screen), they live in memory and game loop keep processing it. Therefore, we need to remove them from list so they will be collected by gc.

Removing Unnecessary Bullets

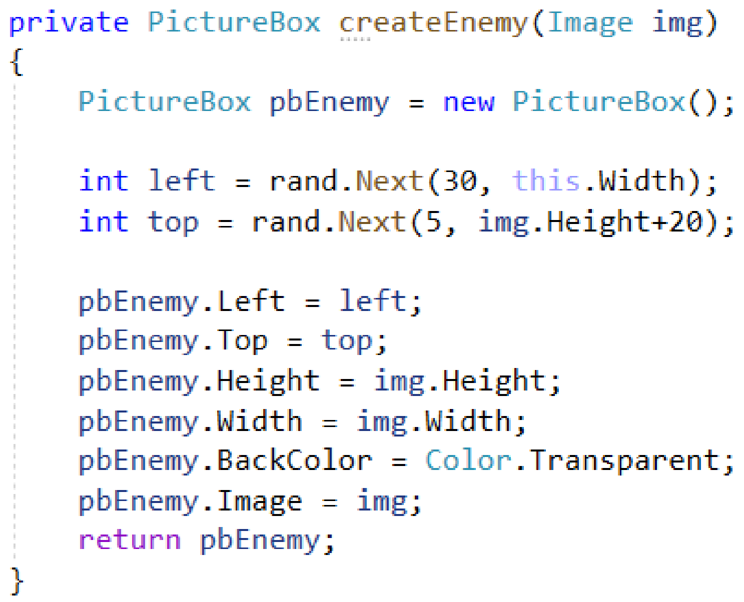
We can not use the foreach loop to remove from the current list. So we have to do it with traditional loop.



```
for (int idx = 0; idx < playerFires.Count; idx++)
{
    if (playerFires[idx].Bottom < 0)
    {
        playerFires.Remove(playerFires[idx]);
    }
}
```

Creating Enemies:

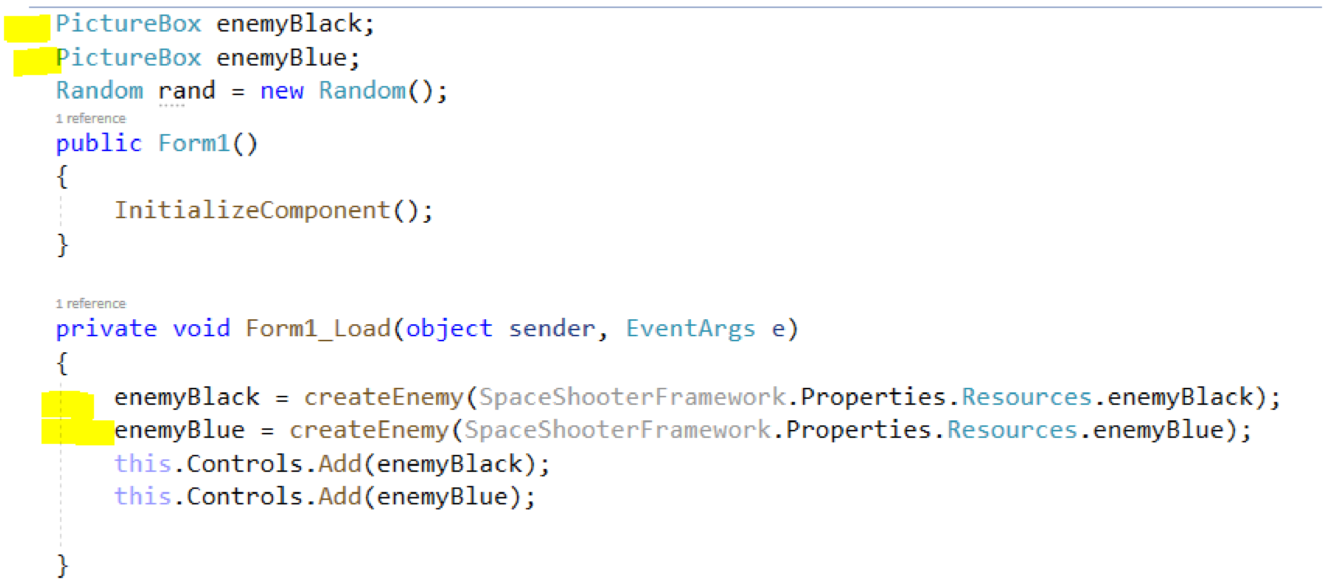
We have to create multiple enemies therefore we can make create Enemy Function



```
private PictureBox createEnemy(Image img)
{
    PictureBox pbEnemy = new PictureBox();

    int left = rand.Next(30, this.Width);
    int top = rand.Next(5, img.Height+20);

    pbEnemy.Left = left;
    pbEnemy.Top = top;
    pbEnemy.Height = img.Height;
    pbEnemy.Width = img.Width;
    pbEnemy.BackColor = Color.Transparent;
    pbEnemy.Image = img;
    return pbEnemy;
}
```

Adding Enemies:

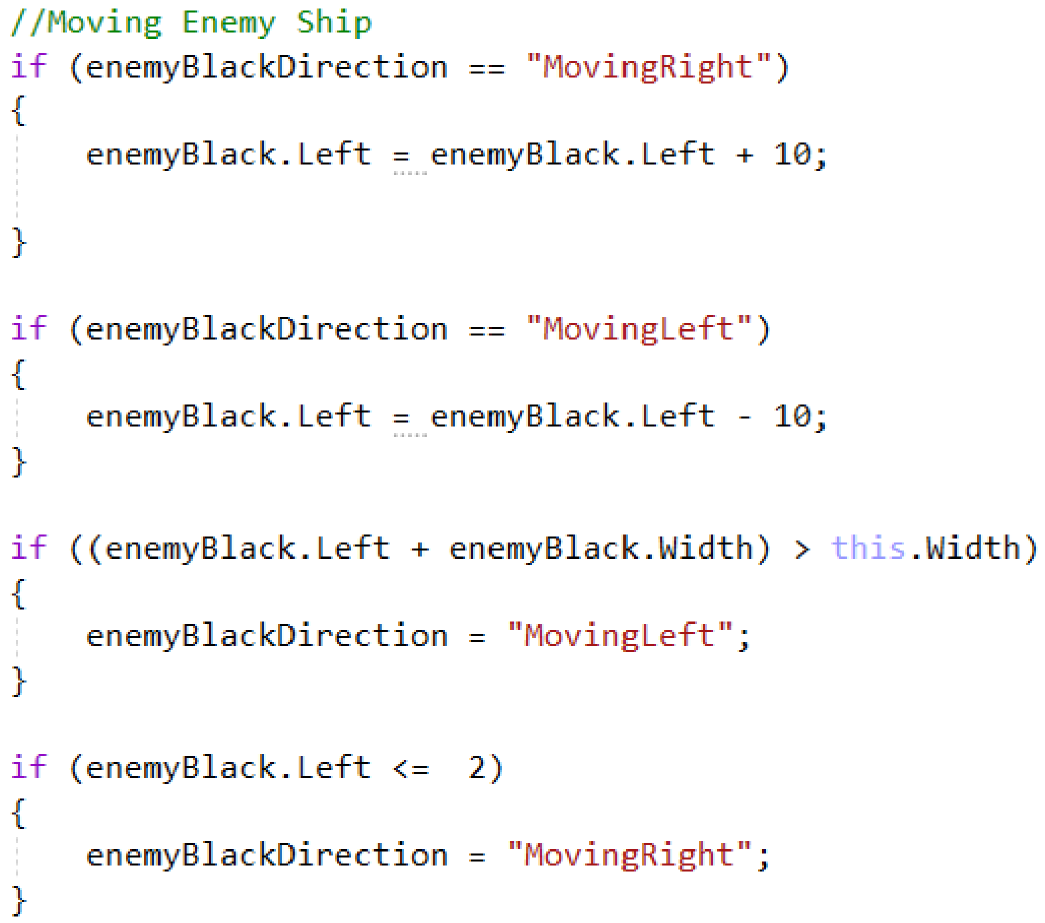
At the moment, we call the function from load form event to create two enemies.



```
PictureBox enemyBlack;
PictureBox enemyBlue;
Random rand = new Random();
1 reference
public Form1()
{
    InitializeComponent();
}

1 reference
private void Form1_Load(object sender, EventArgs e)
{
    enemyBlack = createEnemy(SpaceShooterFramework.Properties.Resources.enemyBlack);
    enemyBlue = createEnemy(SpaceShooterFramework.Properties.Resources.enemyBlue);
    this.Controls.Add(enemyBlack);
    this.Controls.Add(enemyBlue);

}
```

Moving Enemy

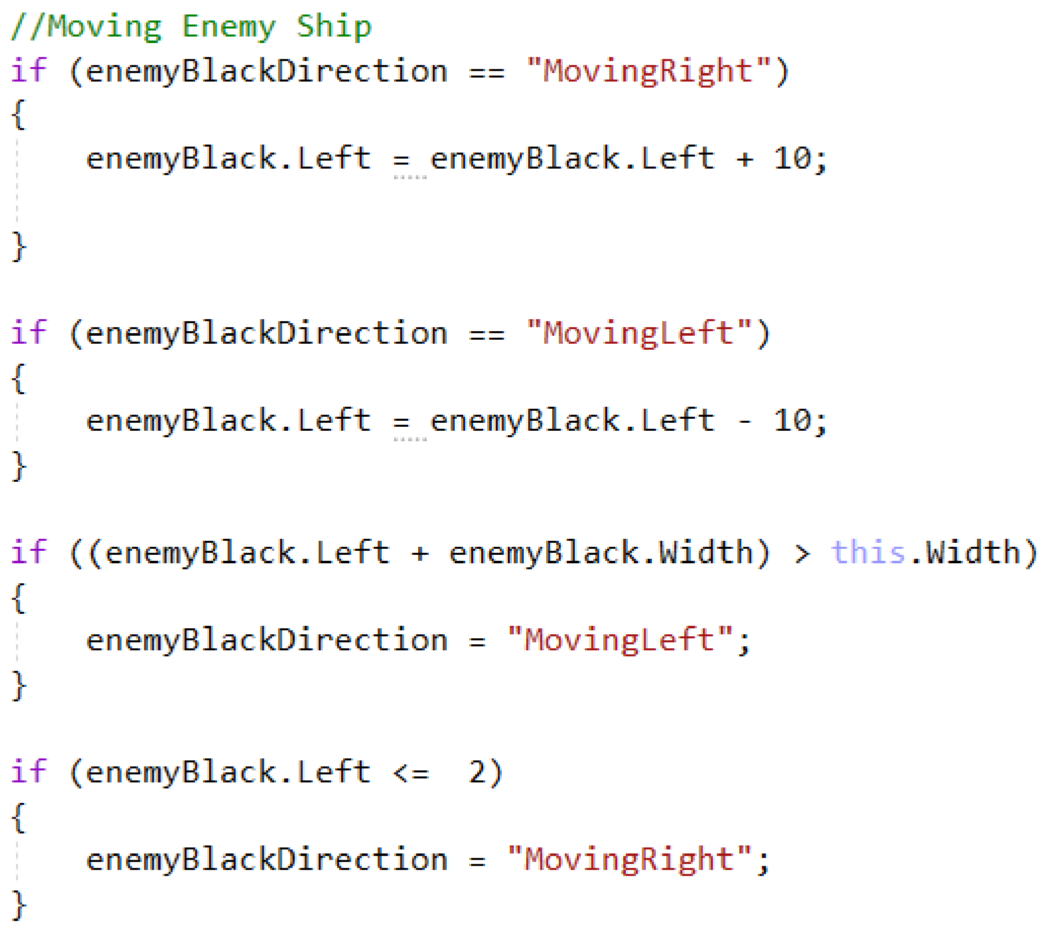
We want to move enemy from left to right and then right to left automatically. Let’s write the code



```
//Moving Enemy Ship
if (enemyBlackDirection == "MovingRight")
{
    enemyBlack.Left = enemyBlack.Left + 10;

}

if (enemyBlackDirection == "MovingLeft")
{
    enemyBlack.Left = enemyBlack.Left - 10;
}

if ((enemyBlack.Left + enemyBlack.Width) > this.Width)
{
    enemyBlackDirection = "MovingLeft";
}

if (enemyBlack.Left <=  2)
{
    enemyBlackDirection = "MovingRight";
}
```

Moving Enemy

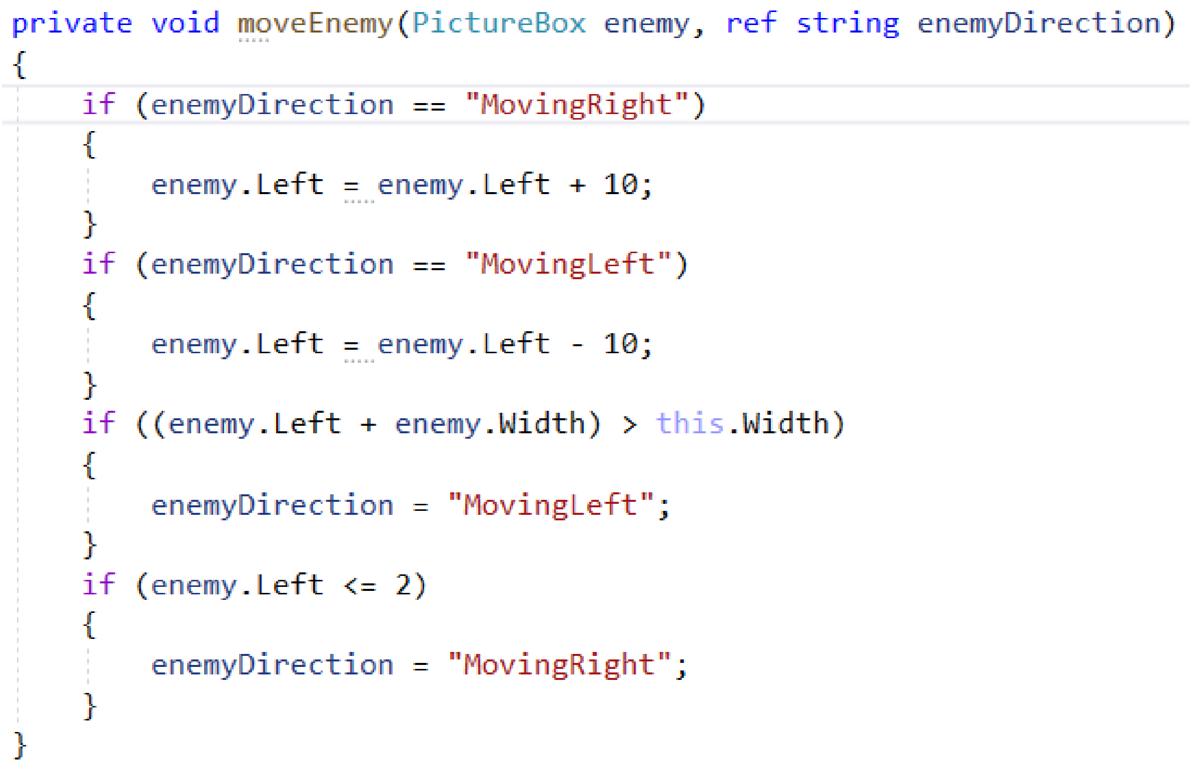
The code is for one enemy how to make it for two enemy but at the moment we do not want to use the classes ?



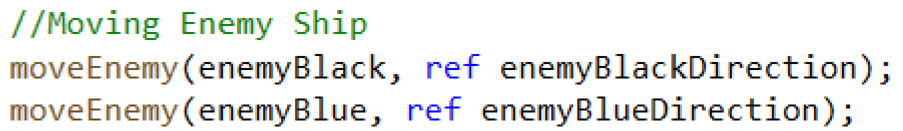
```
//Moving Enemy Ship
if (enemyBlackDirection == "MovingRight")
{
    enemyBlack.Left = enemyBlack.Left + 10;


}

if (enemyBlackDirection == "MovingLeft")
{
    enemyBlack.Left = enemyBlack.Left - 10;
}

if ((enemyBlack.Left + enemyBlack.Width) > this.Width)
{
    enemyBlackDirection = "MovingLeft";
}

if (enemyBlack.Left <=  2)
{
    enemyBlackDirection = "MovingRight";
}
```

Moving Enemy

We can create a function that will modify enemy generically.



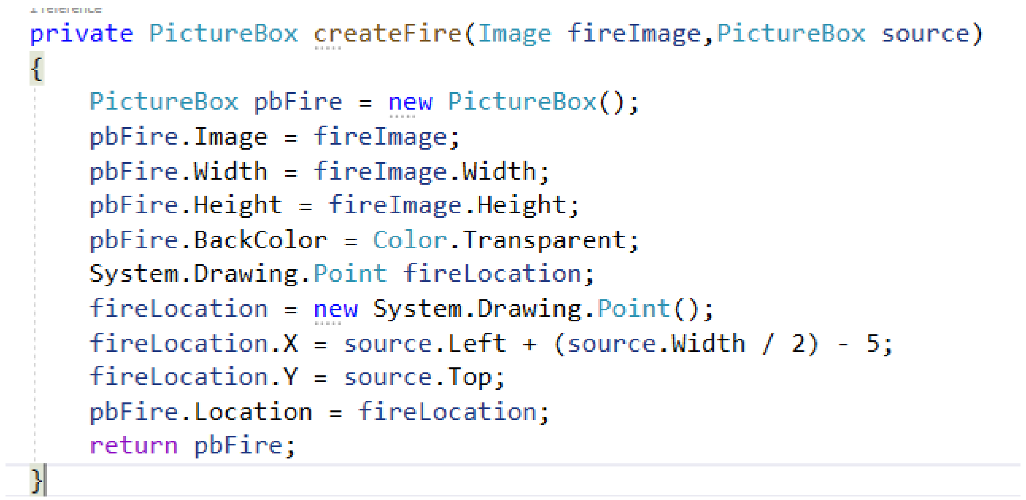
```
private void moveEnemy(PictureBox enemy, ref string enemyDirection)
{
    if (enemyDirection == "MovingRight")
    {
        enemy.Left = enemy.Left + 10;
    }
    if (enemyDirection == "MovingLeft")
    {
        enemy.Left = enemy.Left - 10;
    }
    if ((enemy.Left + enemy.Width) > this.Width)
    {
        enemyDirection = "MovingLeft";
    }
    if (enemy.Left <= 2)
    {
        enemyDirection = "MovingRight";
    }
}
```



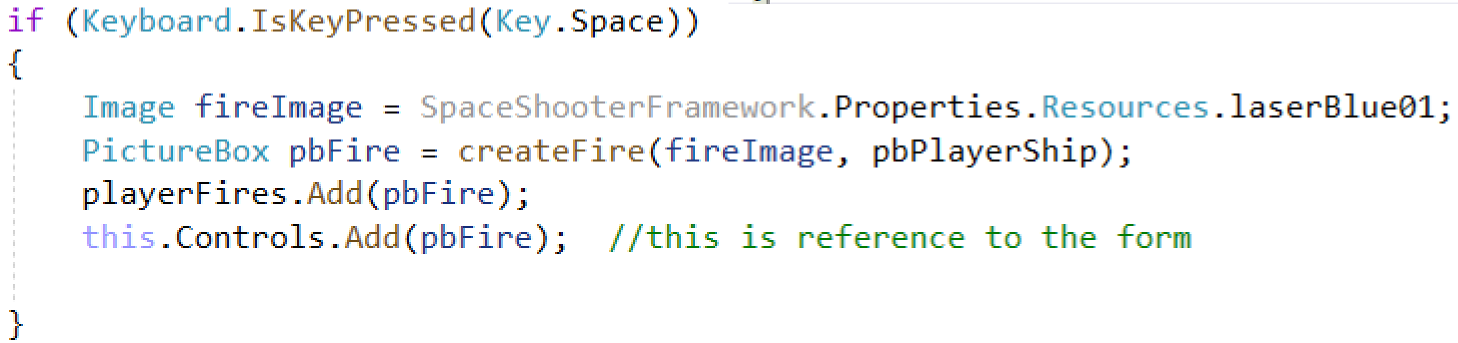
```
//Moving Enemy Ship
moveEnemy(enemyBlack, ref enemyBlackDirection);
moveEnemy(enemyBlue, ref enemyBlueDirection);
```

Making Fire Reusable

We need fire functionality for enemy as well therefore we have to make a separate function to create fire to reuse it.

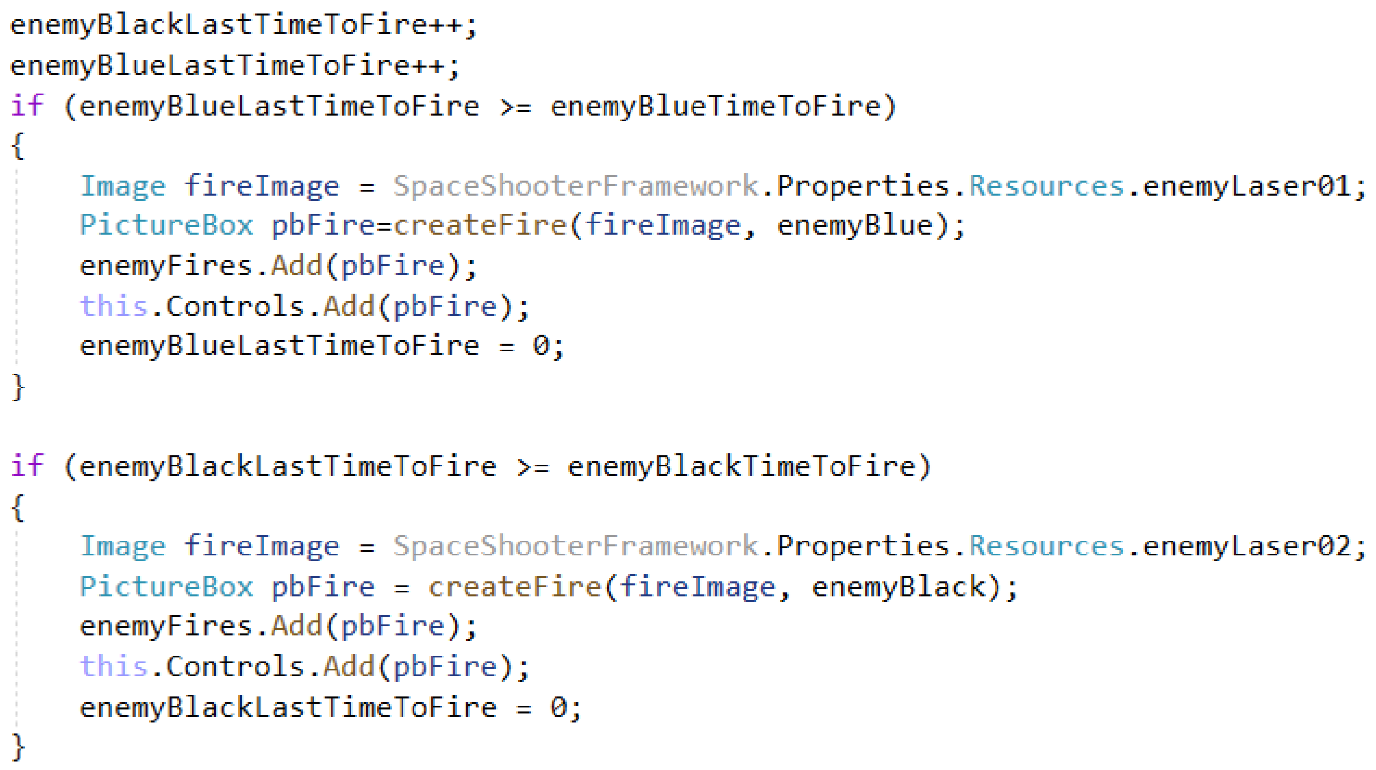


```
private PictureBox createFire(Image fireImage,PictureBox source)
{
    PictureBox pbFire = new PictureBox();
    pbFire.Image = fireImage;
    pbFire.Width = fireImage.Width;
    pbFire.Height = fireImage.Height;
    pbFire.BackColor = Color.Transparent;
    System.Drawing.Point fireLocation;
    fireLocation = new System.Drawing.Point();
    fireLocation.X = source.Left + (source.Width / 2) - 5;
    fireLocation.Y = source.Top;
    pbFire.Location = fireLocation;
    return pbFire;
}
```



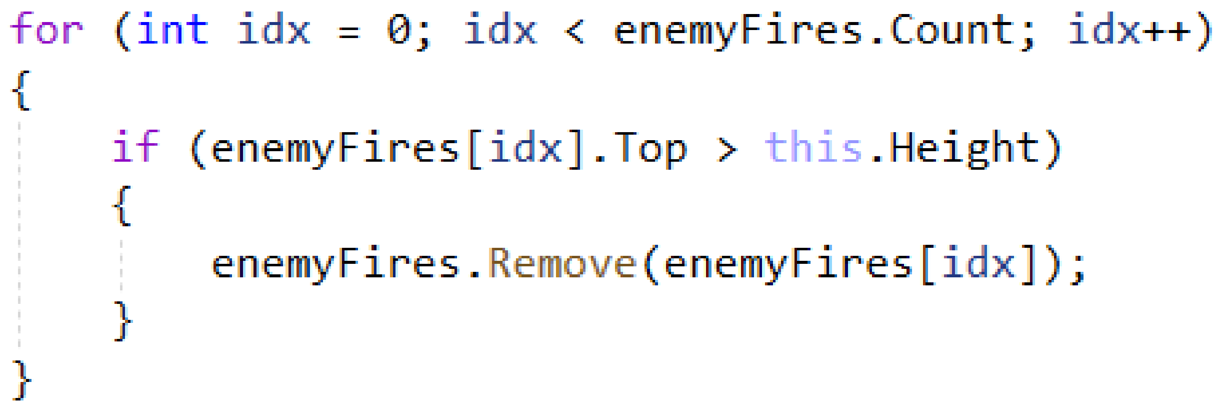
```
if (Keyboard.IsKeyPressed(Key.Space))
{
    Image fireImage = SpaceShooterFramework.Properties.Resources.laserBlue01;
    PictureBox pbFire = createFire(fireImage, pbPlayerShip);
    playerFires.Add(pbFire);
    this.Controls.Add(pbFire);  //this is reference to the form

}
```

Creating Enemy Fire



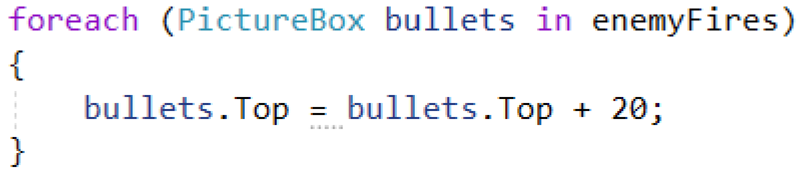
```
enemyBlackLastTimeToFire++;
enemyBlueLastTimeToFire++;
if (enemyBlueLastTimeToFire >= enemyBlueTimeToFire)
{
    Image fireImage = SpaceShooterFramework.Properties.Resources.enemyLaser01;
    PictureBox pbFire=createFire(fireImage, enemyBlue);
    enemyFires.Add(pbFire);
    this.Controls.Add(pbFire);
    enemyBlueLastTimeToFire = 0;
}

if (enemyBlackLastTimeToFire >= enemyBlackTimeToFire)
{
    Image fireImage = SpaceShooterFramework.Properties.Resources.enemyLaser02;
    PictureBox pbFire = createFire(fireImage, enemyBlack);
    enemyFires.Add(pbFire);
    this.Controls.Add(pbFire);
    enemyBlackLastTimeToFire = 0;
}
```

Removing the Fires from the List



```
for (int idx = 0; idx < enemyFires.Count; idx++)
{
    if (enemyFires[idx].Top > this.Height)
    {
        enemyFires.Remove(enemyFires[idx]);
    }
}
```

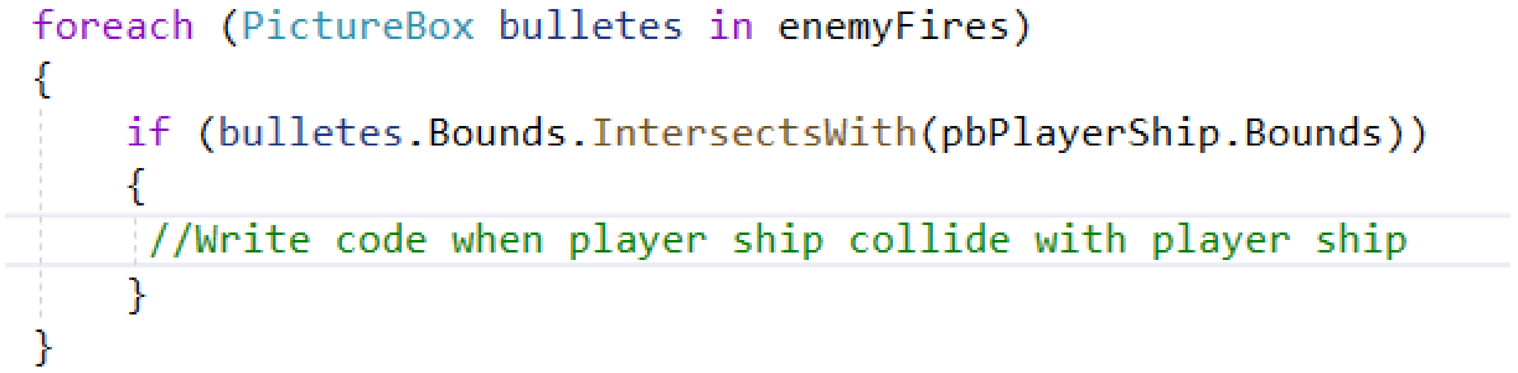
Moving Enemy Fires



```
foreach (PictureBox bullets in enemyFires)
{
    bullets.Top = bullets.Top + 20;
}
```

Identify the Collision

Now, we need to write the code that identify the collision between the enemy bullets and the player ship. How to do that ?



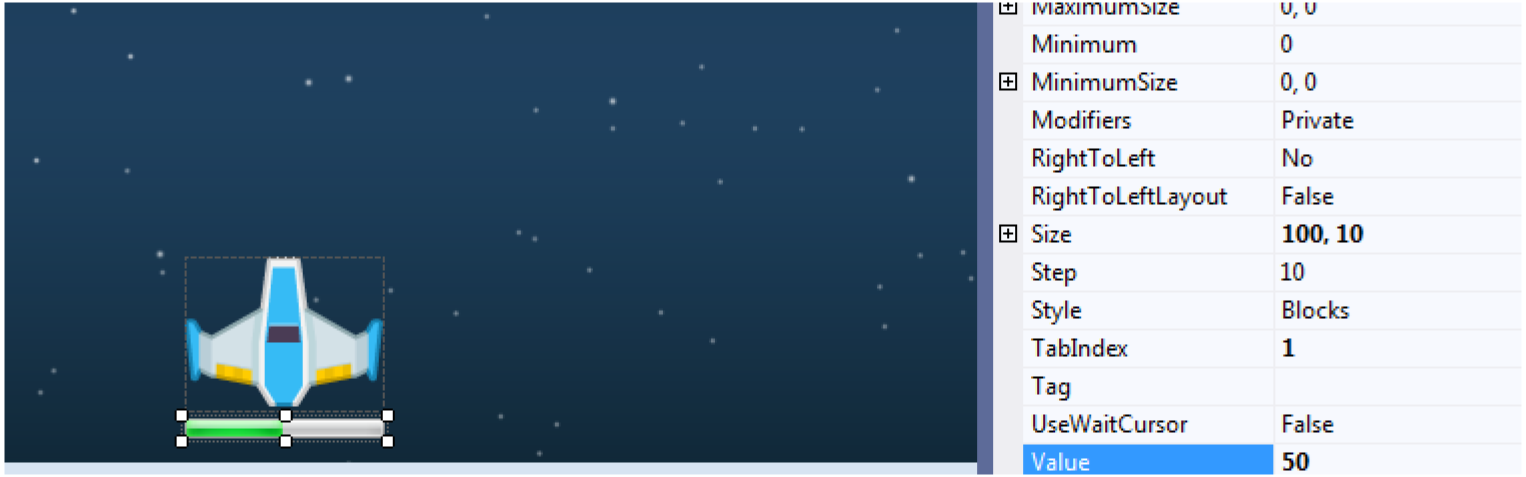
```
foreach (PictureBox bulletes in enemyFires)
{
    if (bulletes.Bounds.IntersectsWith(pbPlayerShip.Bounds))
    {
     //Write code when player ship collide with player ship
    }
}
```

Player Health

We can show health to user with help of the progress bar control. We add the control right under the ship and move it with the playership

Player Health

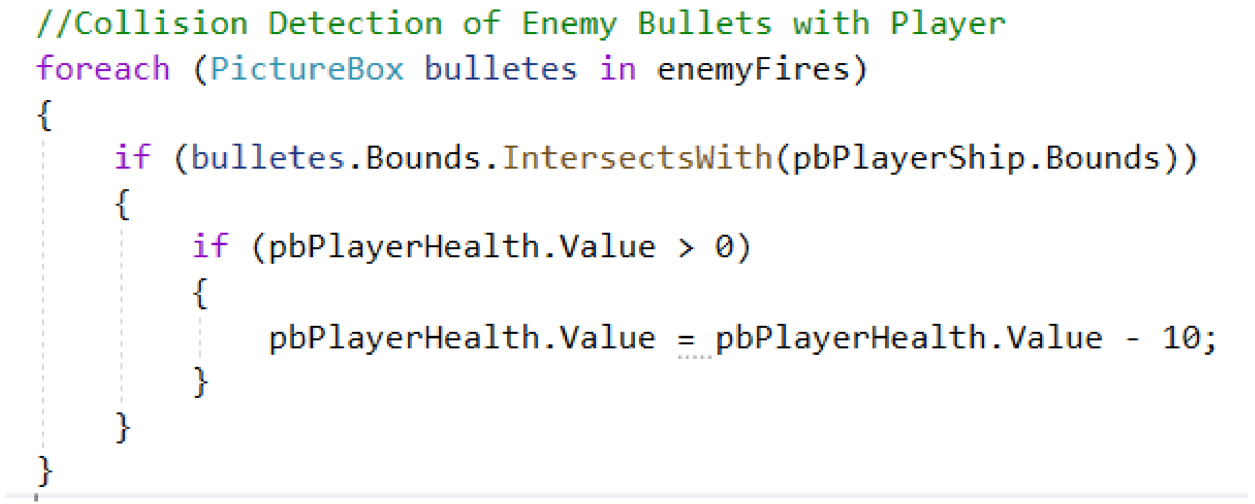
There is an attribute of progress bar name value which shows how much progress bar should be filled



MaximumSize	0, 0
Minimum	0
⊞ MinimumSize	0, 0
Modifiers	Private
RightToLeft	No
RightToLeftLayout	False
⊞ Size	**100, 10**
Step	10
Style	Blocks
TabIndex	**1**
Tag	
UseWaitCursor	False
Value	**50**

Player Health

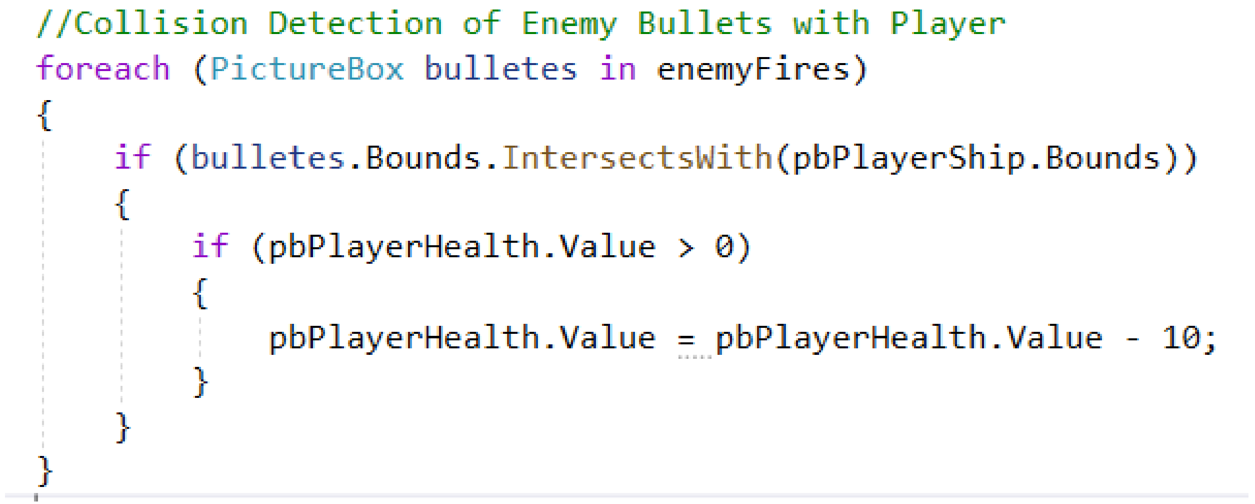
So on collision, we shall **decrease the player health** by 10 points till the value of the progress bar is not zero



```
//Collision Detection of Enemy Bullets with Player
foreach (PictureBox bulletes in enemyFires)
{
    if (bulletes.Bounds.IntersectsWith(pbPlayerShip.Bounds))
    {
        if (pbPlayerHealth.Value > 0)
        {
            pbPlayerHealth.Value = pbPlayerHealth.Value - 10;
        }
    }
}
```

Player Health: Game Over

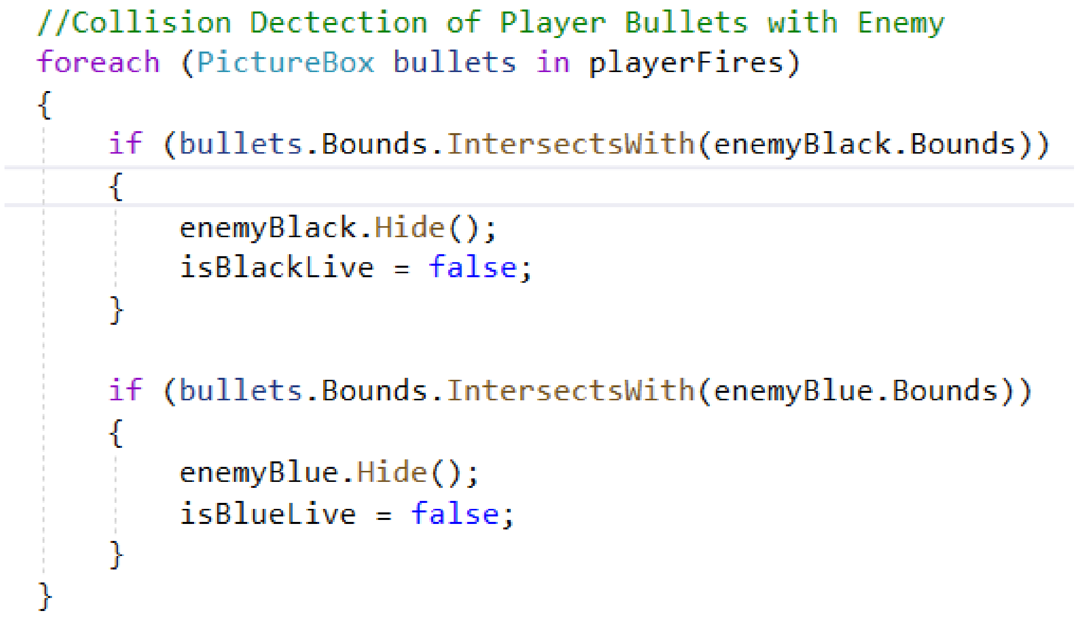
We need to game over when the health of the player get zero



```
//Collision Detection of Enemy Bullets with Player
foreach (PictureBox bulletes in enemyFires)
{
    if (bulletes.Bounds.IntersectsWith(pbPlayerShip.Bounds))
    {
        if (pbPlayerHealth.Value > 0)
        {
            pbPlayerHealth.Value = pbPlayerHealth.Value - 10;
        }
    }
}
```

Destroy Enemy:

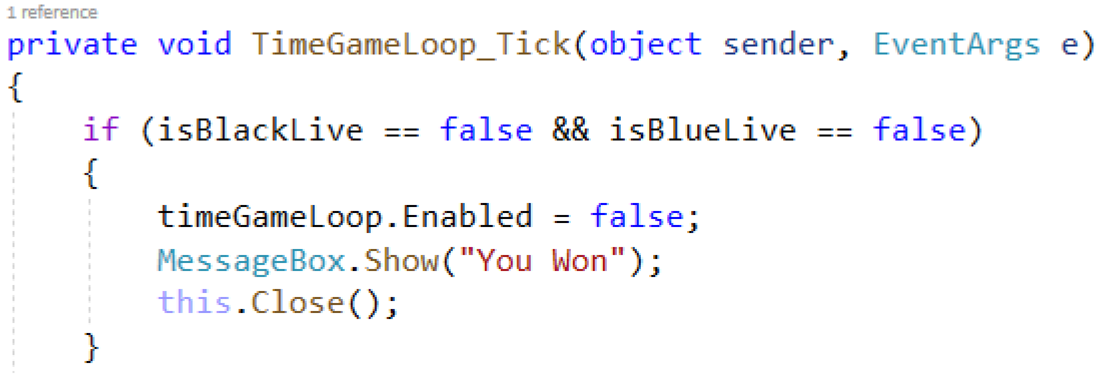
Similarly, we can destroy enemy when player bullet hit it. For simplicity we are destroying enemy at single bullet.



```
//Collision Dectection of Player Bullets with Enemy
foreach (PictureBox bullets in playerFires)
{
    if (bullets.Bounds.IntersectsWith(enemyBlack.Bounds))
    {
        enemyBlack.Hide();
        isBlackLive = false;
    }

    if (bullets.Bounds.IntersectsWith(enemyBlue.Bounds))
    {
        enemyBlue.Hide();
        isBlueLive = false;
    }
}
```

Destroy Enemy: Game Win

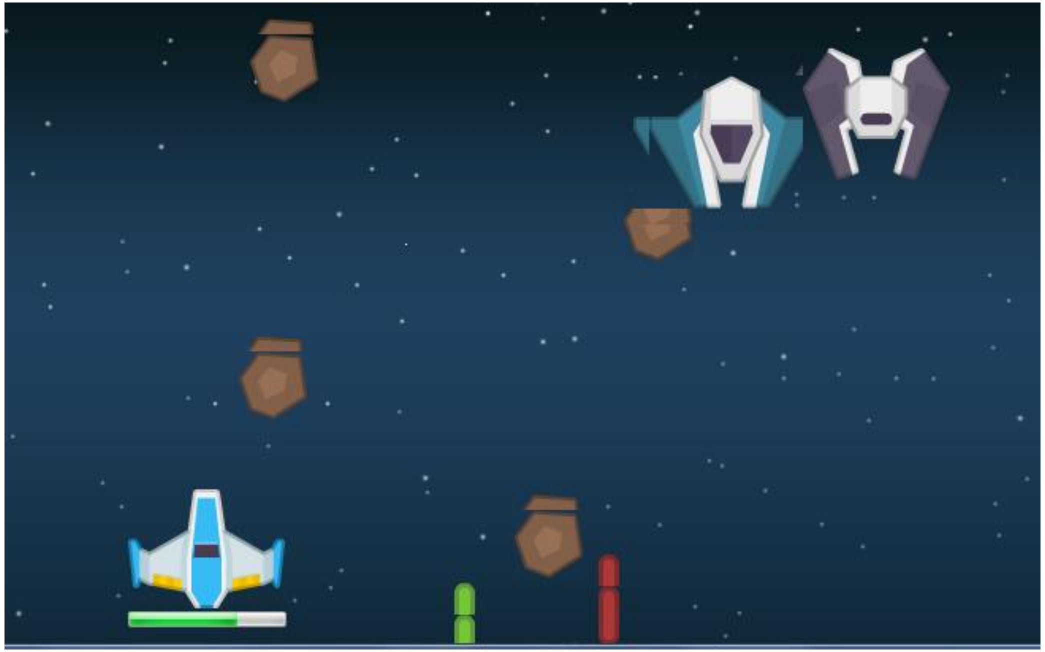
When all enemies are destroyed game should be won by the user. For that we can write the code.



```
1 reference
private void TimeGameLoop_Tick(object sender, EventArgs e)
{
    if (isBlackLive == false && isBlueLive == false)
    {
        timeGameLoop.Enabled = false;
        MessageBox.Show("You Won");
        this.Close();
    }
```

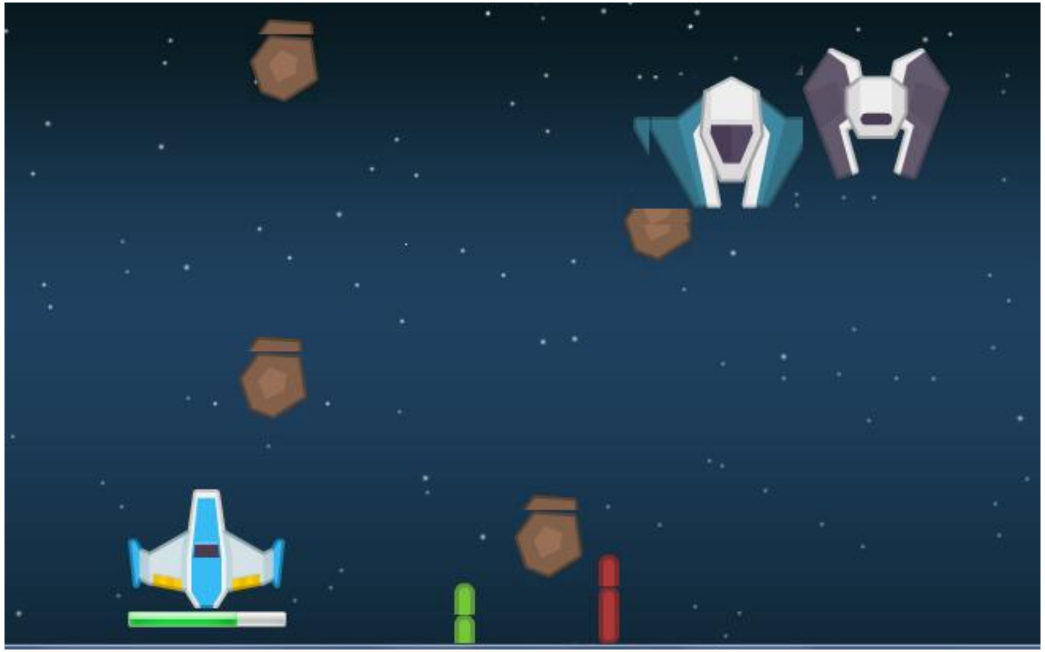
Adding Falling Meteoroid

Now we make our game a bit more interesting to add falling meteoroid.



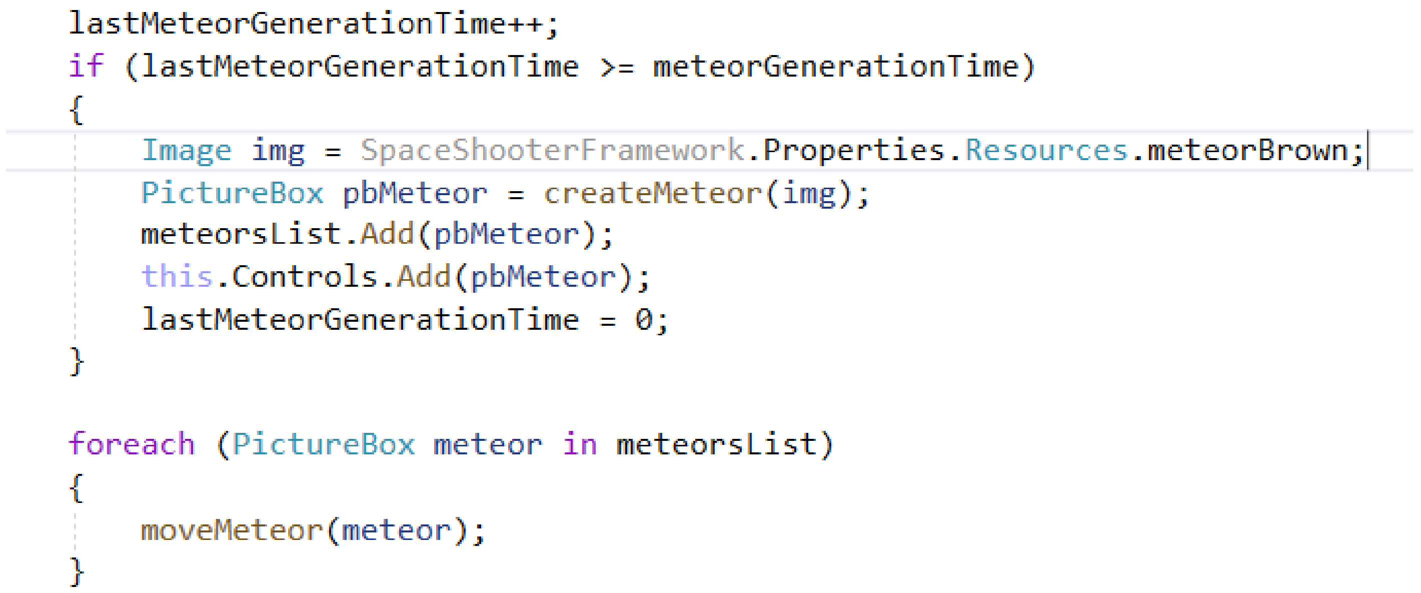
Adding Falling Meteoroid

We want to generate meteorite after some time and want to generate these randomly.



Adding Falling Meteoroid

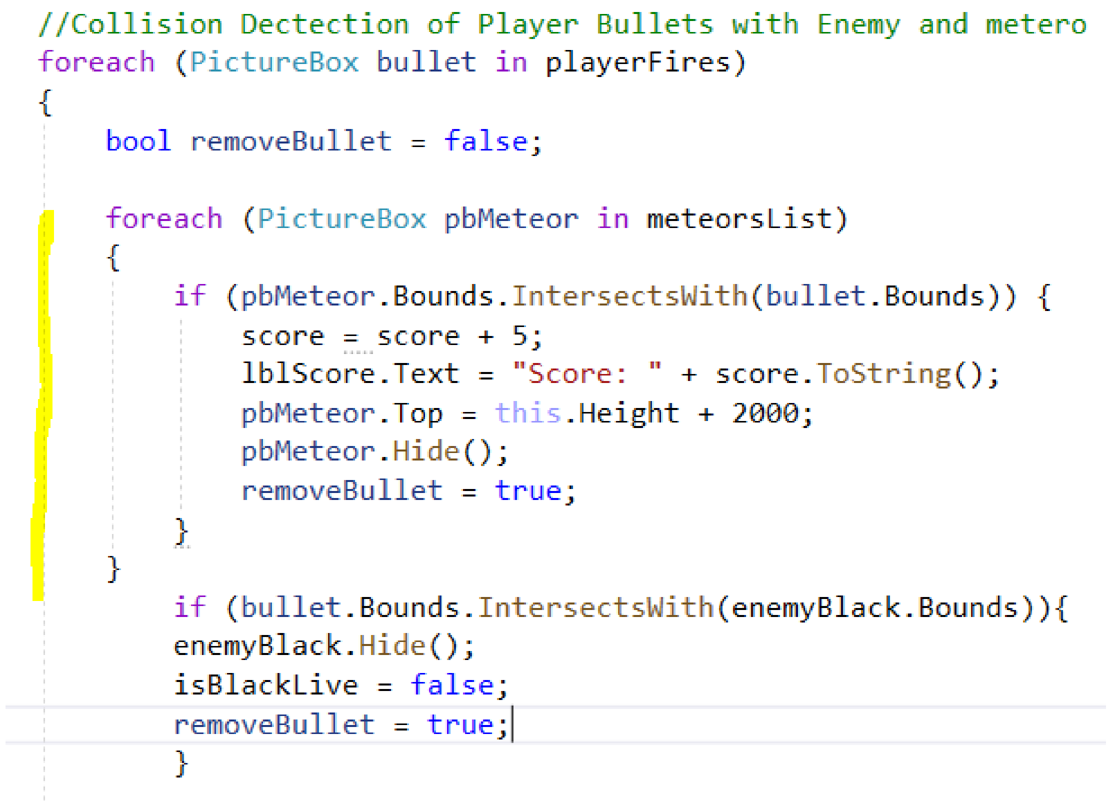
We want to generate meteorite after some time and want to generate these randomly.



```
lastMeteorGenerationTime++;
if (lastMeteorGenerationTime >= meteorGenerationTime)
{
    Image img = SpaceShooterFramework.Properties.Resources.meteorBrown;
    PictureBox pbMeteor = createMeteor(img);
    meteorsList.Add(pbMeteor);
    this.Controls.Add(pbMeteor);
    lastMeteorGenerationTime = 0;
}

foreach (PictureBox meteor in meteorsList)
{
    moveMeteor(meteor);
}
```

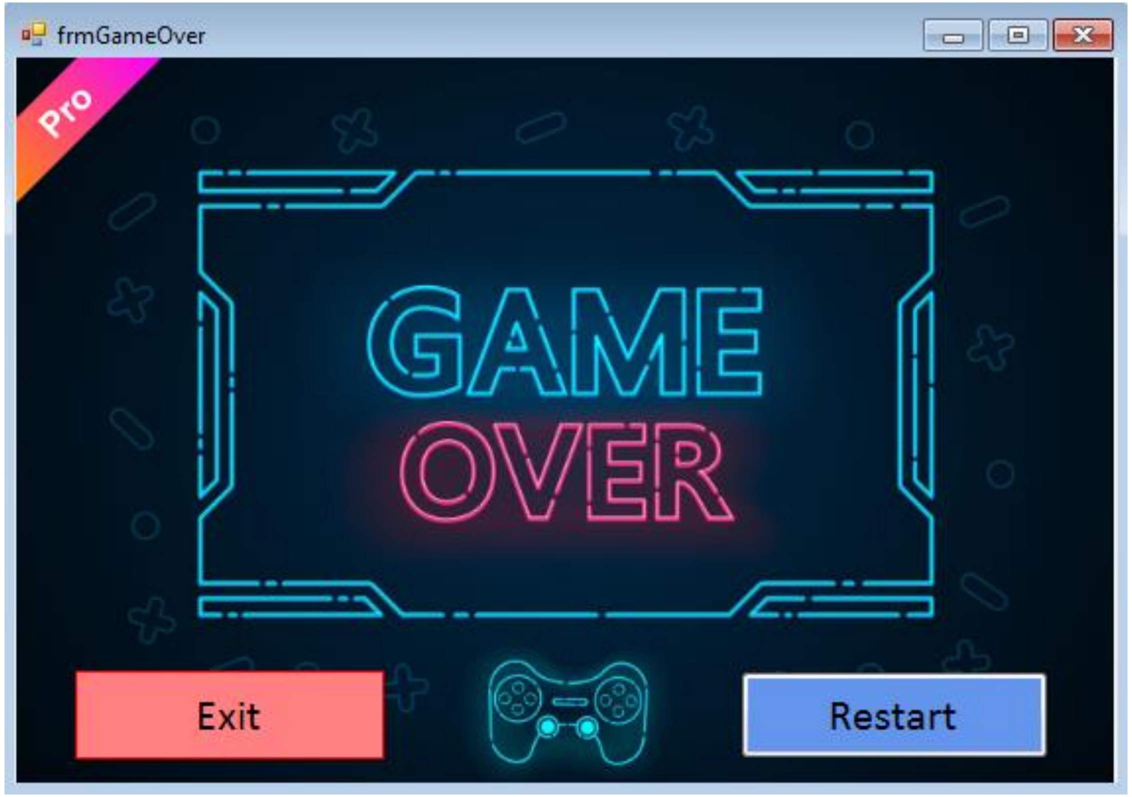
Adding Scoring Behaviour

Now, if the player's bullet hit the meteoroid, we want to add 5 to score. Also, once bullet is hit to meteoroid it should be hide and removed from the list.



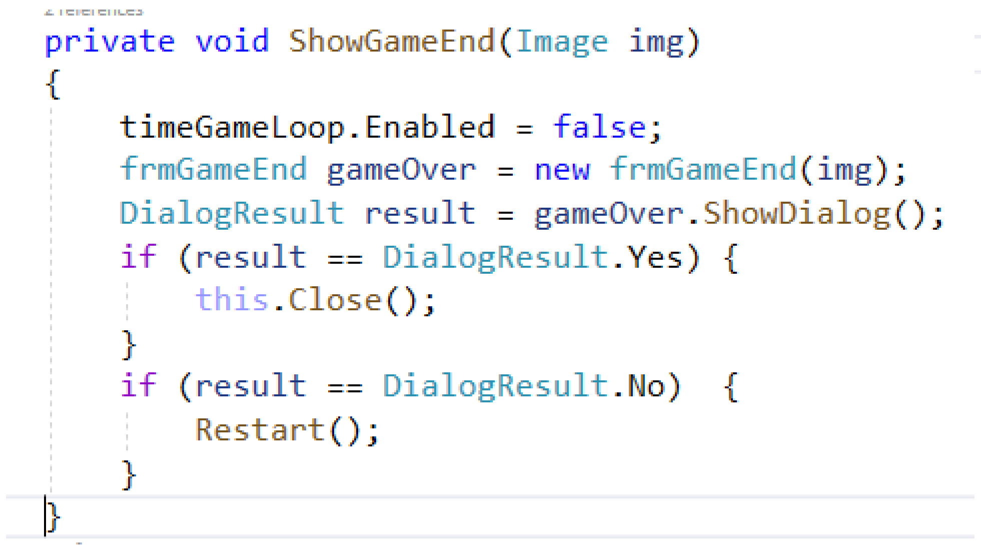
```
//Collision Dectection of Player Bullets with Enemy and metero
foreach (PictureBox bullet in playerFires)
{
    bool removeBullet = false;

    foreach (PictureBox pbMeteor in meteorsList)
    {
        if (pbMeteor.Bounds.IntersectsWith(bullet.Bounds)) {
            score = score + 5;
            lblScore.Text = "Score: " + score.ToString();
            pbMeteor.Top = this.Height + 2000;
            pbMeteor.Hide();
            removeBullet = true;
        }
    }
        if (bullet.Bounds.IntersectsWith(enemyBlack.Bounds)){
        enemyBlack.Hide();
        isBlackLive = false;
        removeBullet = true;
        }
```

Now, we add the Game End Screen

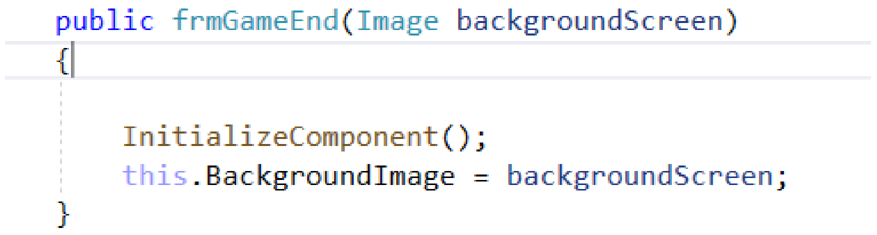


Show Game End Screen

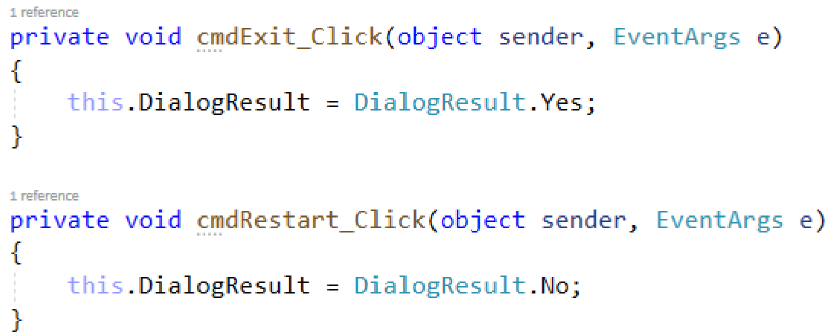
We use dialogue box to show the game screen and use the DialogResult to decide what option user has chosen



```
private void ShowGameEnd(Image img)
{
    timeGameLoop.Enabled = false;
    frmGameEnd gameOver = new frmGameEnd(img);
    DialogResult result = gameOver.ShowDialog();
    if (result == DialogResult.Yes) {
        this.Close();
    }
    if (result == DialogResult.No)  {
        Restart();
    }
}
```



```
public frmGameEnd(Image backgroundScreen)
{

    InitializeComponent();
    this.BackgroundImage = backgroundScreen;
}
```

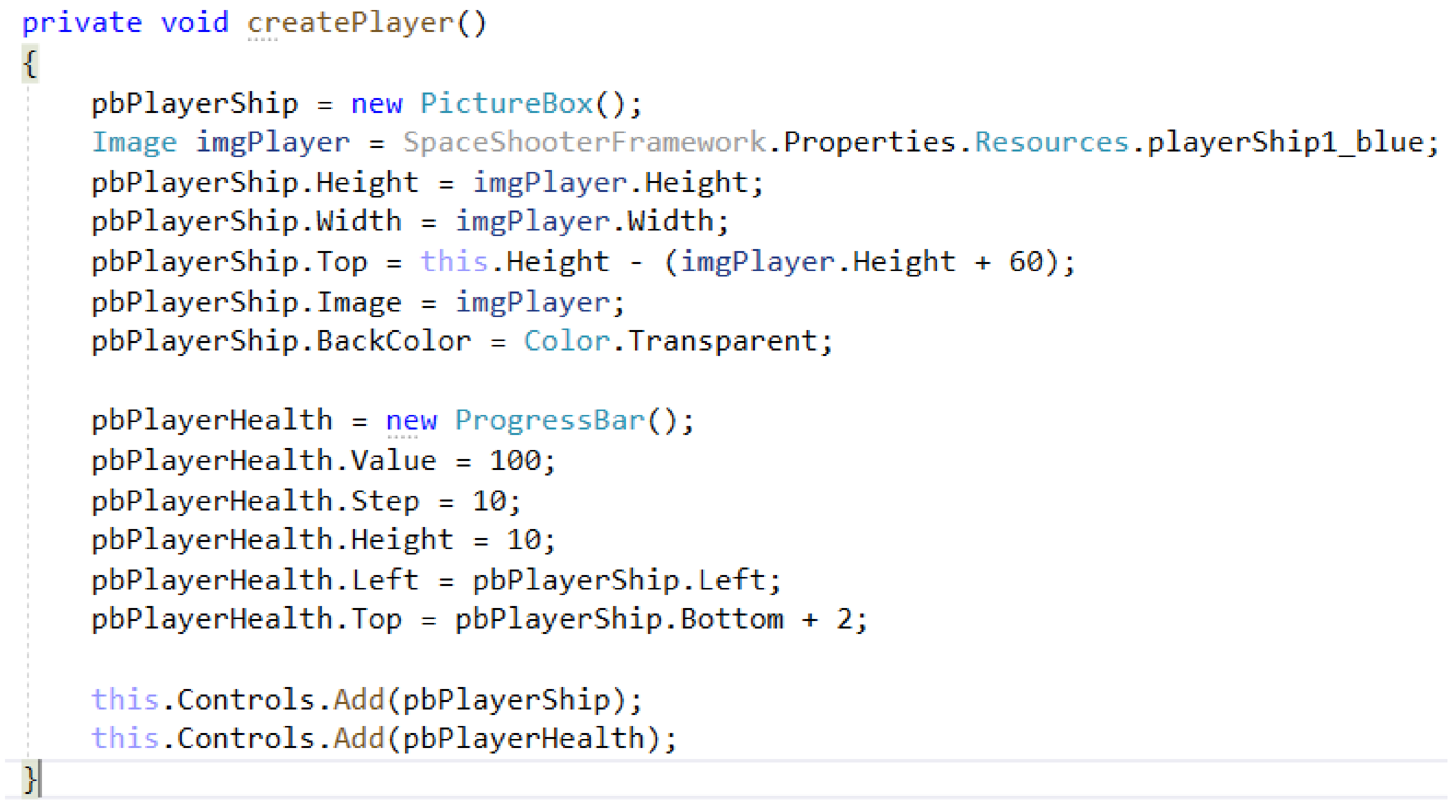


```
1 reference
private void cmdExit_Click(object sender, EventArgs e)
{
    this.DialogResult = DialogResult.Yes;
}

1 reference
private void cmdRestart_Click(object sender, EventArgs e)
{
    this.DialogResult = DialogResult.No;
}
```

Creating Playership Dynamically.

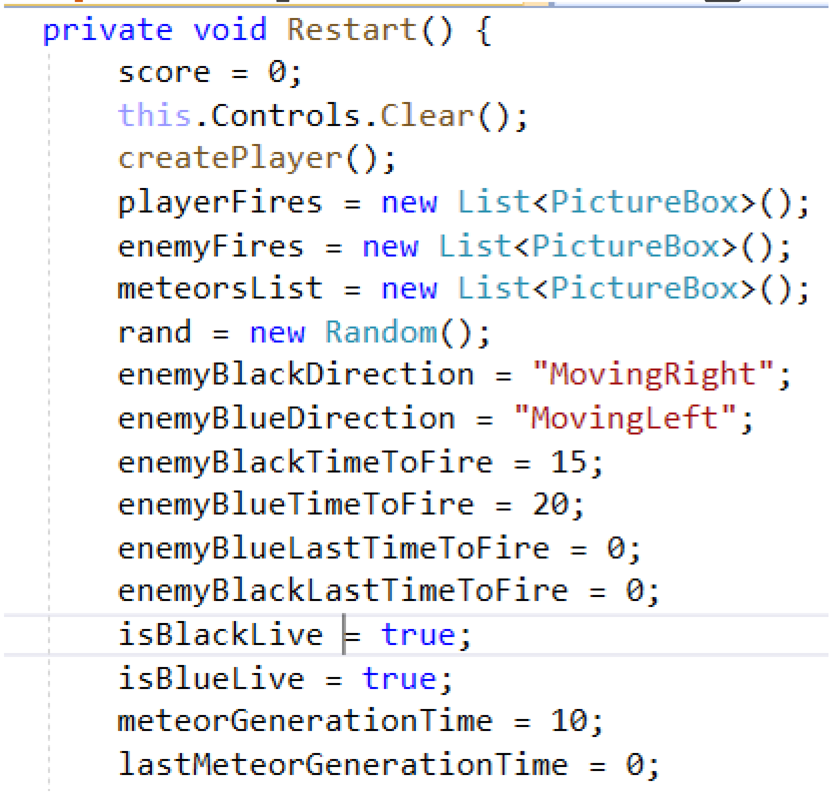
Before, looking into the code how to restart the game, we need to create player dynamically.
At the moment the player ship is created at design time but we have to create player ship through code so game can be restarted.

Creating Playership Dynamically.

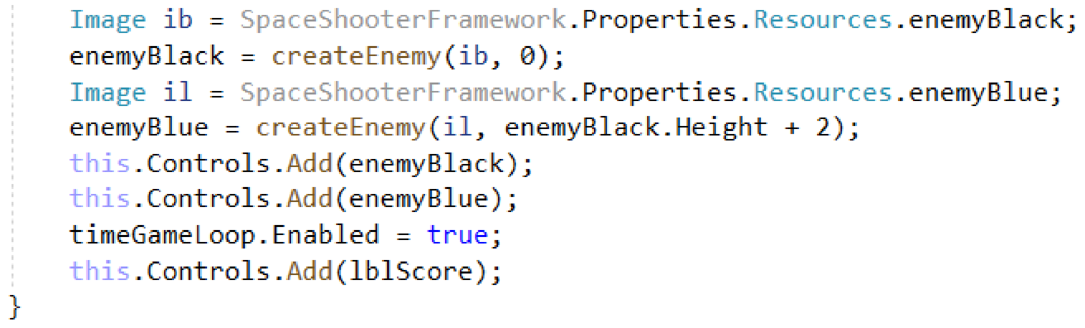


```
private void createPlayer()
{
    pbPlayerShip = new PictureBox();
    Image imgPlayer = SpaceShooterFramework.Properties.Resources.playerShip1_blue;
    pbPlayerShip.Height = imgPlayer.Height;
    pbPlayerShip.Width = imgPlayer.Width;
    pbPlayerShip.Top = this.Height - (imgPlayer.Height + 60);
    pbPlayerShip.Image = imgPlayer;
    pbPlayerShip.BackColor = Color.Transparent;

    pbPlayerHealth = new ProgressBar();
    pbPlayerHealth.Value = 100;
    pbPlayerHealth.Step = 10;
    pbPlayerHealth.Height = 10;
    pbPlayerHealth.Left = pbPlayerShip.Left;
    pbPlayerHealth.Top = pbPlayerShip.Bottom + 2;

    this.Controls.Add(pbPlayerShip);
    this.Controls.Add(pbPlayerHealth);
}
```

Implementing Restart.



```
private void Restart() {
    score = 0;
    this.Controls.Clear();
    createPlayer();
    playerFires = new List<PictureBox>();
    enemyFires = new List<PictureBox>();
    meteorsList = new List<PictureBox>();
    rand = new Random();
    enemyBlackDirection = "MovingRight";
    enemyBlueDirection = "MovingLeft";
    enemyBlackTimeToFire = 15;
    enemyBlueTimeToFire = 20;
    enemyBlueLastTimeToFire = 0;
    enemyBlackLastTimeToFire = 0;
    isBlackLive = true;
    isBlueLive = true;
    meteorGenerationTime = 10;
    lastMeteorGenerationTime = 0;
```



```
    Image ib = SpaceShooterFramework.Properties.Resources.enemyBlack;
    enemyBlack = createEnemy(ib, 0);
    Image il = SpaceShooterFramework.Properties.Resources.enemyBlue;
    enemyBlue = createEnemy(il, enemyBlack.Height + 2);
    this.Controls.Add(enemyBlack);
    this.Controls.Add(enemyBlue);
    timeGameLoop.Enabled = true;
    this.Controls.Add(lblScore);
}
```