

S.I.W.S

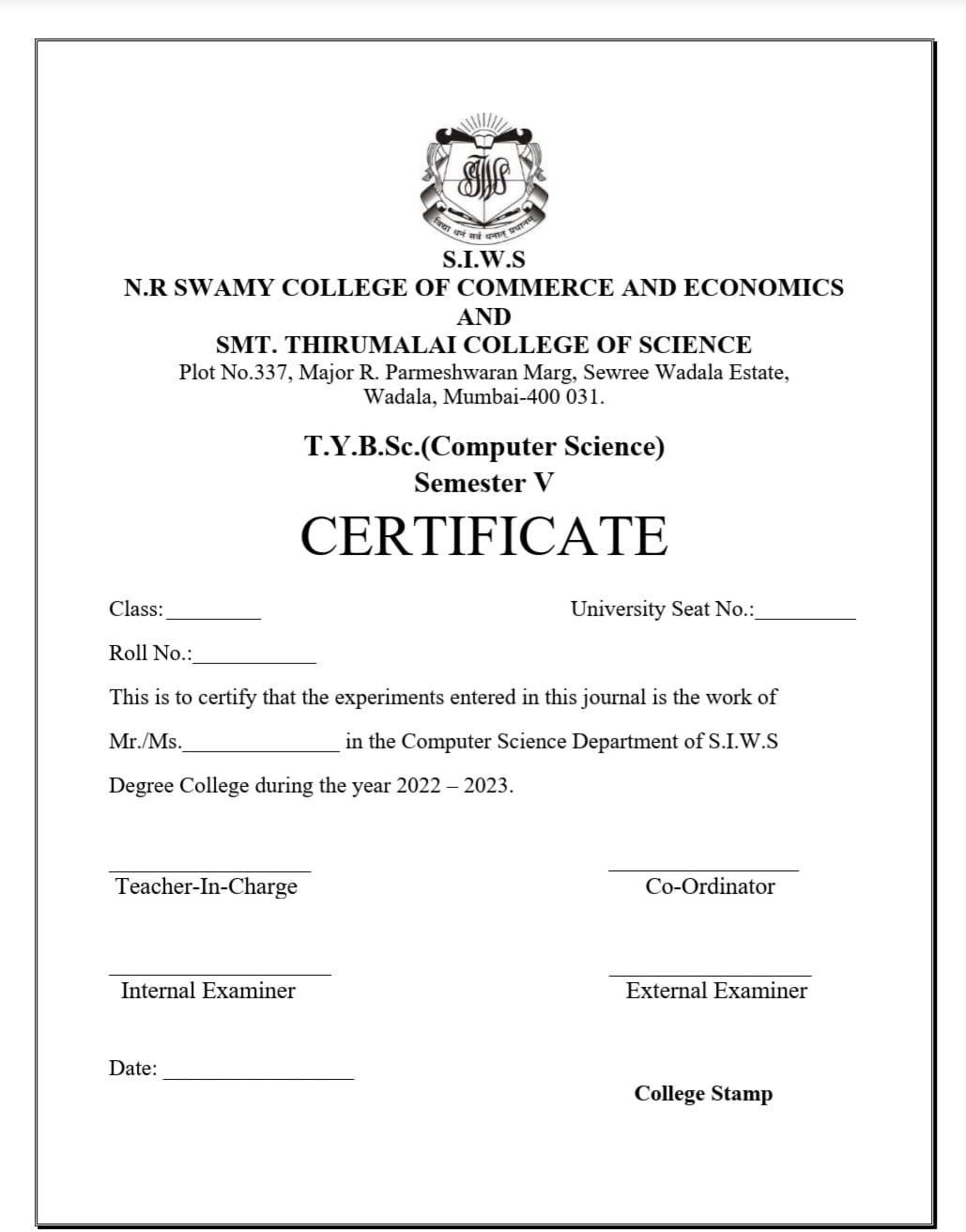
N.R SWAMY COLLEGE OF COMMERCE AND ECONOMICS AND SMT.THIRUMALAI COLLEGE OF SCIENCE.

**GAME PROGRAMMING**

JAISWAR ROHIT MANOJ

T.Y.Bsc C.S. 39041

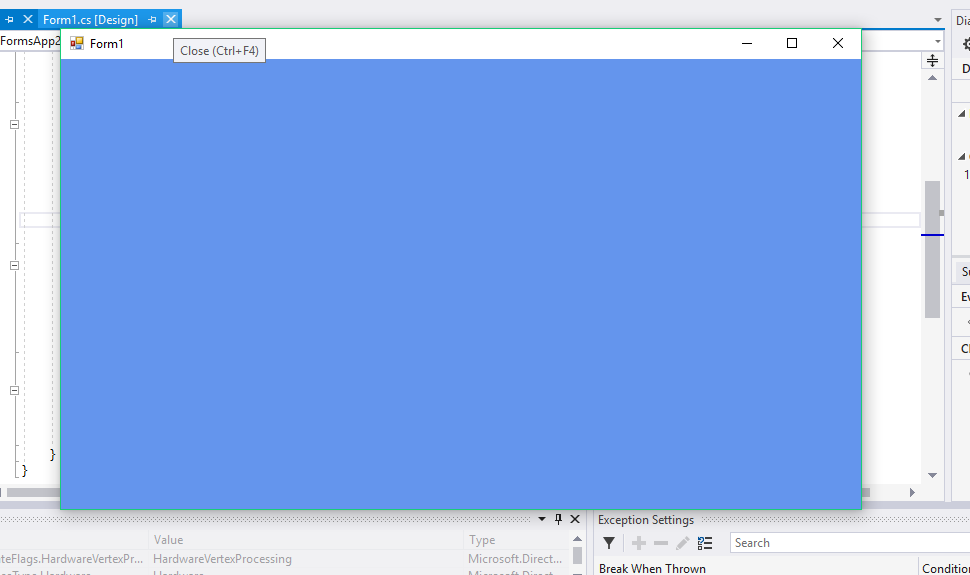
Year 2022-2023



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SR.NO** | **DATE** | **TITLE** | **PG.NO** | **SIGN** | |
| **1** | **26/08/22** | **Practical No:1**  **Setup DirectX 11, Window Framework and Initialize Direct3D Device** |  |  | |
| **2** | **26/08/22** | **Practical No:2**  **Buffers, shaders and HSL(Draw a triangle using Direct 3D 11)** |  |  | |
| **3** | **09/09/22** | **Practical No:3**  **Texturing**  **(Texture the triangle using Direct3D 11)** |  |  | |
| **4** | **09/09/22** | **Practical No:4**  **Lightining**  **(Programmable Diffuse Lightning using Direct 3D 11)** |  |  | |
| **5** | **16/09/22** | **Practical No:5**  **Specular Lighting**  **(Programmable spot lightining using DirectX)** |  |  | |
| **6** | **16/09/22** | **Practical No:6**  **Spotlight using unity** |  |  | |
| **7** | **23/09/22** | **Practical No: 7**  **2D UFO using unity** |  |  | |
| **8** | **23/09/22** | **Practical No:8**  **Space shooter using unity** |  |  |

**INDEX**

**Output of the code:**



**PRACTICAL NO 1**

**Aim:**

**Input of the code:**

In this practical we are just learning the window framework and initializing a Direct3D device.

**Step 1:**

Create new project, and select “Windows Forms Application”, select .NET Framework as 2.0 in Visuals C#.

Right Click on properties Click on open click on build Select Platform Target and Select x86.

**Step 2:**Click on View Code of Form 1.

**Step 3:**

Go to Solution Explorer, right click on project name, and select Add Reference. Click on Browse and select the given .dll files which are “Microsoft.DirectX”, “Microsoft.DirectX.Direct3D”, and “Microsoft.DirectX.DirectX3DX”.

**Step 4:**

Go to Properties Section of Form, select Paint in the Event List and enter as Form1\_Paint.

**Step 5:**

Edit the Form’s C# code file. Namespace must be as same as your project name.

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Text;

usingSystem.Windows.Forms;

usingMicrosoft.DirectX;

using Microsoft.DirectX.Direct3D;

namespace GP\_P1

{

public partial class Form1 : Form

{

Microsoft.DirectX.Direct3D.Device device;

public Form1()

{

InitializeComponent();

InitDevice();

}

public void InitDevice()

{

PresentParameterspp = new PresentParameters();

pp.Windowed = true;

pp.SwapEffect = SwapEffect.Discard;

device = new Device(0, DeviceType.Hardware, this,

CreateFlags.HardwareVertexProcessing, pp);

}

private void Render()

{

device.Clear(ClearFlags.Target, Color.Orange, 0, 1);

device.Present();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Render();

}

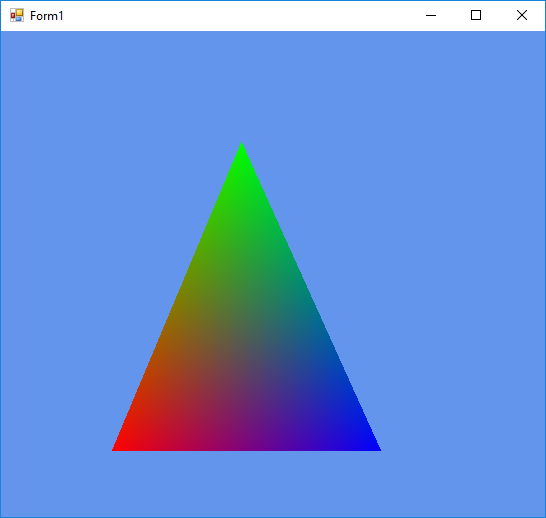
}

}

**Step 6:** Click on Start. And here is the output. We have initialized 3D Device.

**Conclusion:**

**Output of the code:**



**PRACTICAL NO 2**

**Aim:**

**Input of the code:**

**Solution:**

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Text;

usingSystem.Windows.Forms;

usingMicrosoft.DirectX;

using Microsoft.DirectX.Direct3D;

namespace GP\_P2

{

public partial class Form1 : Form

{

Microsoft.DirectX.Direct3D.Device device;

public Form1()

{

InitializeComponent();

InitDevice();

}

private void InitDevice()

{

PresentParameterspp = new PresentParameters();

pp.Windowed = true;

pp.SwapEffect = SwapEffect.Discard;

device = new Device(0, DeviceType.Hardware, this,

CreateFlags.HardwareVertexProcessing, pp);

}

private void Render()

{

CustomVertex.TransformedColored[]

vertexes = new CustomVertex.TransformedColored[3];

vertexes[0].Position = new Vector4(240, 110, 0, 1.0f);//first point

vertexes[0].Color = System.Drawing.Color.FromArgb(0, 255, 0).ToArgb();

vertexes[1].Position = new Vector4(380, 420, 0, 1.0f);//second point

vertexes[1].Color = System.Drawing.Color.FromArgb(0, 0, 255).ToArgb();

vertexes[2].Position = new Vector4(110, 420, 0, 1.0f);//third point

vertexes[2].Color = System.Drawing.Color.FromArgb(255, 0, 0).ToArgb();

device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1.0f, 0);

device.BeginScene();

device.VertexFormat = CustomVertex.TransformedColored.Format;

device.DrawUserPrimitives(PrimitiveType.TriangleList, 1, vertexes);

device.EndScene();

device.Present();

}

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

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Render();

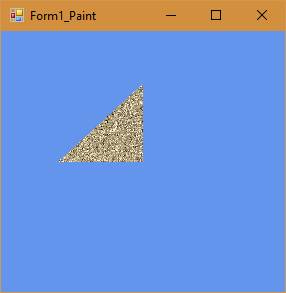
}

}

}

**Conclusion:**

**Output of the code:**



**PRACTICAL NO 3**

**Aim:**

**Input of the code:**

**Solution:**

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Text;

usingSystem.Windows.Forms;

usingMicrosoft.DirectX;

using Microsoft.DirectX.Direct3D;

namespace Gp\_prac3

{

public partial class Form1 : Form

{

private Microsoft.DirectX.Direct3D.Device device;

privateCustomVertex.PositionTextured[]

vertex = new CustomVertex.PositionTextured[3];

private Texture texture;

public Form1()

{

InitializeComponent();

InitDevice();

}

private void InitDevice()

{

PresentParameterspp = new PresentParameters();

pp.Windowed = true;

pp.SwapEffect = SwapEffect.Discard;

device=newDevice(0,DeviceType.Hardware,this,CreateFlags.HardwareVertexProcessing, pp);

device.Transform.Projection= Matrix.PerspectiveFovLH(3.14f/4,device.Viewport.Width/ device.Viewport.Height, 1f, 1000f);

device.Transform.View = Matrix.LookAtLH(new Vector3(0, 0, 20), new Vector3(),

new Vector3(0, 1, 0));

device.RenderState.Lighting = false;

vertex[0] = new CustomVertex.PositionTextured(new Vector3(0, 0, 0), 0, 0);

vertex[1] = new CustomVertex.PositionTextured(new Vector3(5, 0, 0), 0, 1);

vertex[2] = new CustomVertex.PositionTextured(new Vector3(0, 5, 0),-1, 1);

texture=new Texture (device,new Bitmap ("E:\\TYCS\\images\\img1.jpg"), 0,

Pool.Managed );

}

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

{ }

private void Form1\_Paint(Object sender, PaintEventArgs e)

{

device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1, 0);

device.BeginScene();

device.SetTexture(0,texture);

device.VertexFormat = CustomVertex.PositionTextured.Format;

device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3, vertex);

device.EndScene();

device.Present();

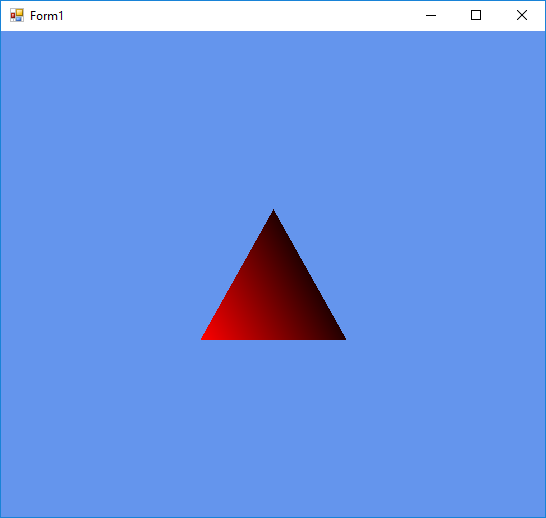
}

}

}

**Conclusion:**

**Output of the code:**



**PRATICAL NO 4**

**Aim:**

**Input of the code:**

**Solution:**

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Text;

usingSystem.Windows.Forms;

usingMicrosoft.DirectX;

using Microsoft.DirectX.Direct3D;

namespace GP\_P2

{

public partial class Form1 : Form

{

private Microsoft.DirectX.Direct3D.Device device;

privateCustomVertex.PositionNormalColored[]

vertex = new CustomVertex.PositionNormalColored[3];

public Form1()

{

InitializeComponent();

InitDevice();

}

public void InitDevice()

{

PresentParameterspp = new PresentParameters();

pp.Windowed = true;

pp.SwapEffect = SwapEffect.Discard;

device=newDevice(0,DeviceType.Hardware,this,CreateFlags.HardwareVertexProcessing, pp);

device.Transform.Projection=Matrix.PerspectiveFovLH(3.14f/4, device.Viewport.Width / device.Viewport.Height, 1f, 1000f);

device.Transform.View = Matrix.LookAtLH(new Vector3(0, 0, 10), new

Vector3(), new Vector3(0, 1, 0));

device.RenderState.Lighting = false;

vertex[0] = new CustomVertex.PositionNormalColored(new Vector3(0, 1, 1), new Vector3(1, 0, 1), Color.Red.ToArgb());

vertex[1] = new CustomVertex.PositionNormalColored(new Vector3(-1, -1, 1), new Vector3(1, 0, 1), Color.Red.ToArgb());

vertex[2] = new CustomVertex.PositionNormalColored(new Vector3(1, -1, 1), new Vector3(-1, 0, 1), Color.Red.ToArgb());

device.RenderState.Lighting = true;

device.Lights[0].Type = LightType.Directional;

device.Lights[0].Diffuse = Color.Plum;

device.Lights[0].Direction = new Vector3(0.8f, 0, -1);

device.Lights[0].Enabled = true;

}

public void Render()

{

device.Clear(ClearFlags.Target, Color.CornflowerBlue, 1, 0);

device.BeginScene();

device.VertexFormat = CustomVertex.PositionNormalColored.Format;

device.DrawUserPrimitives(PrimitiveType.TriangleList, vertex.Length / 3, vertex);

device.EndScene();

device.Present();

}

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

{

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Render();

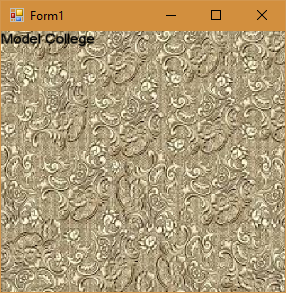
}

}

}

**Conclusion:**

**Output of the code:**



**PRACTICAL NO 5**

**Aim:**

**Input of the code:**

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Text;

usingSystem.Windows.Forms;

usingMicrosoft.DirectX;

using Microsoft.DirectX.Direct3D;

namespace GP\_P5\_Loading\_Model

{

public partial class Form1 : Form

{

Microsoft.DirectX.Direct3D.Device device;

Microsoft.DirectX.Direct3D.Texture texture;

Microsoft.DirectX.Direct3D.Font font;

public Form1()

{

InitializeComponent();

InitDevice();

InitFont();

LoadTexture();

}

private void InitFont()

{

System.Drawing.Font f = new System.Drawing.Font("Arial", 16f,

FontStyle.Regular);

font = new Microsoft.DirectX.Direct3D.Font(device, f);

}

private void LoadTexture()

{

texture = TextureLoader.FromFile(device,"E:\\TYCS\\images\\img1.jpg",400, 400, 1, 0, Format.A8B8G8R8, Pool.Managed, Filter.Point, Filter.Point,

Color.Transparent.ToArgb());

}

private void InitDevice()

{

PresentParameterspp = new PresentParameters();

pp.Windowed = true;

pp.SwapEffect = SwapEffect.Discard;

device = new Device(0, DeviceType.Hardware, this,

CreateFlags.HardwareVertexProcessing, pp);

}

private void Render()

{

device.Clear(ClearFlags.Target, Color.CornflowerBlue, 0, 1);

device.BeginScene();

using (Sprite s = new Sprite(device))

{

s.Begin(SpriteFlags.AlphaBlend);

s.Draw2D(texture, new Rectangle(0, 0, 0, 0), new Rectangle(0, 0,

device.Viewport.Width, device.Viewport.Height), new Point(0, 0), 0f, new

Point(0, 0), Color.White);

font.DrawText(s, "Model College", new Point(0, 0), Color.Black);

s.End();

}

device.EndScene();

device.Present();

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

Render();

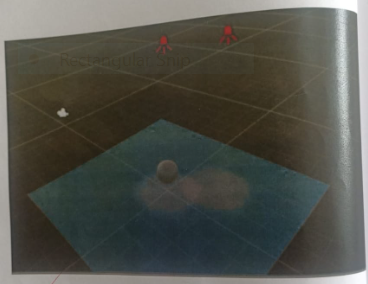
}

}

}

**Conclusion:**

**Output of the code:**



**PRACTICAL NO 6**

**Aim:**

**Input of the code:**

**Solution:**

**Steps:**

* Add a plane using Game Object
* Add a sphere using Game Object
* Change sphere Y axis position to “2”, X=0, Z=0
* Select Directional light and disable it
* Add a Spot Light using Game Object ->light
* Change Spot lights position to X=0 Y=3 Z=0
* Change Spot plane’s Scale to X=5 Y=5 Z=5 8.Create a duplicate Spot light by right clicking on it
* Change the color of light
* Save scene or your work will vanish
* Go to project window right click>create>c # script 12. Code for NewBehaviourScript.cs
* Add Newly created script to spot lights
* Click on Play button

**Program:**

**NewBehaviourScript.cs**

Using System.Collections;

Using System.Collections.Generic;

Using UnityEngine;

Public class NewBehaviourScript:MonoBehaviour

{

//Use this for initialization

Void Start()

{

}

//Update is called once per frame void Update()

{

Transform.RotateAround(Vector3.zero,Vector3.up,40\*Time.deltaTime);

}

}

**Conclusion :**

**Output of the code:**



**PRACTICAL NO 7**

**Aim:**

**Input of the code:**

* Step 1: Open unity software and create a new project.

Choose the 2d option, create the project.

* Go to window button and open asset store.
* Click on Unity essential, then go to sample projects
* You see 2d UFO tutorial package then open it.
* Import the package in software
* Save your scene. (Crtl+s )
* Step 2: Setting up the field
* Then click on sprites.
* This is your 2d UFO sprites.
* Drag background to hierarchy.
* Then you see right corners inspector button click on it.
* Sprite Renderer box is created.
* Same as it is UFO. Drag to hierarchy.
* You can change the name.
* Then you see sorting layer in sprite renderer, then set that layer background to background and UFO to player.
* Step 3: Controlling the player.
* Click on UFO and add component
* Click on 2d physics
* Click on rigidbody
* Then you see in inspector rigidbody box created
* We need to create a script for moving our UFO
* Click on add component and create
* Name the script.
* That script drag into your asset scripts
* Open the script
* Write that script into it.
* Then test your game.
* Then your UFO fall down because of gravity scale
* Go to rigidbody2d and gravity scale 1 to change 0.
* Then your UFO speed is so slow then go to your script and add that 2 lines
* “ Public float speed ;
* Go to inspector and see your rigidbody 2d is updated with speed.
* Set your speed
* Step 4 : Adding Collision.
* Go to add component and type circle collider this is for UFO.
* This is the circle collider.
* This the Radius 2.15 for UFO collider.
* Then we use as same for our background.
* Go to add component and go to 2D physics and choose box collider.
* Then go to your scene and 1inch down you see shaded button click on this and change to wireframe.
* Then set you x axis offset size 14.3 and y axis is zero. Box collider size for x axis 3.3 and y axis is 31.64.
* Then copy that component and paste component.
* See diagram above.
* And same as it y axis -14.3 for offset and x is zero
* Same as size collider y axis is 3.3 and x axis is 31.64
* Step 5: Following the player with camera
* That is a simple for camera
* We need to create script for camera control
* That is our script.
* And that script drag to player script
* Step 6: Creating a collectables objects.
* Then you see drag a UFO to hierarchy same as it go to sprites an take pick-ups object and drag it.
* Click on pick up and see inspector click on sprite render set sorting layer to pick-ups
* Then deactivate player object
* Then go to add component and add again circle collider box for our pick objects.
* Set radius to 0.94.
* We need to rotate our object or animate, then create a new scripts
* Then pickup object in hierarchy to drag into see this.
* Then we need to create a game object
* Go to game object and create empty click on it.
* Then our new object is created and drag into 1st pick-up in new game object.
* Then create duplicate our object
* Go to edit and duplicate
* Set your objects in game scene your own mind
* Step 7: picking a collectables object.
* We create pick-ups object then we need change our script in player module.
* Then go inspector see circle collider box the right corner you see setting button and book manual. Click on book manual
* The search onTriggerEnter2d and click on it.
* Copy the code and paste our UFO script.
* Below the rb2d.AddForce.
* Then delete this 2

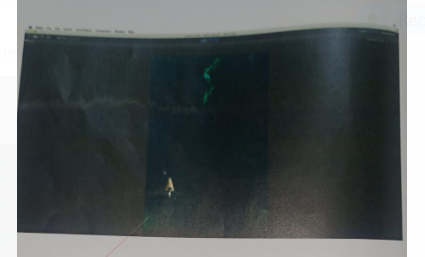
🡪lines 1st public bool

🡪2nd characterInQuicksand.

* The type that code in script.
* Then select untagged in inspector and change into pickups.
* Then go to circle collider body and activated is trigger.
* Add component rigiddbody 2d and activated kinematic field on
* Step 8: Collecting object count and Displaying score
* Go to player script again and add the code.
* “ Private in count ;”
* Add “ count =0 ; ”
* Count = count+1;
* Create gaming UI object
* Rename the UI object.
* Press the f button then you see new text then go to setting and reset coordinate
* Then click on anchor and shift+ alt click.
* Hold the that key and click to that square
* Set x axis 10 and y axis -10.
* Add this code for UFO for count text.

**Conclusion:**

**Output of the code:**



**PRACTICAL NO 8**

**Aim:**

**Input of the code:**

Step 1: **Open unity software create a new project.**

* Go to file-> New Project->Click on create new project ->set the location (By clicking on Set Button

)-> Give the name to your project As Space Shooter->Click on save Button->Click on Create Button.

* Go to window button and open asset store.
* Click on Unity essential, then go to sample projetcs
* Import the package in software

The Scene will look after Importing the Asset-> Select All Packages->Click on Import.

* Save the Asset is created your scene. (Crtl+s)
* Save the scene inside the Asset directory make new folder as \_Scene. Give the name to your scene as main the save it.
* Now we will set the build target to our scene
* Go to file->Build Setting Web Player->Click on Switch Platform
* Now we need to fill the build detail.
* Go to file->Project setting->Player->

Set the resolution:

* Go to Inspector window(Right middle corner of the scene)->Resolution-
* >Change the resolution width as 600 AND height as 900.
* Drag the game view to the top.
* Now save the layout**:** Choose layout->click on save layout->Give the name as Space Shooter->click on save button.

**STEP 2:** Setting Up the player Game Object

* Go to scene view->Add the player Ship Model From mode Directory->Drag the vehicle player ship from model directory to hierarchy
* Go to edit ->choose frame selected
* Now Ho To Hierarchy->double click on Vehicle ship->Rename it as player-> press Enter Key
* Add New Component into the Inspector window
* Click on Add Component Tab->Select physics->Rigid Body
* Go to Right Body Component->Deselect use Gravity->
* Add New Component into the Inspector Window
* Click on Add Component Tab->Select Physics->Select Capsule Collider
* Now change the capsule collider Direction to Z-axis->change the radius and Height also->
* Go to Add component button again->Physics->Mesh collider->Replace
* Go to mesh collider component inside the Inspector window->turn off the mesh render tab
* Open the model->Select the mesh Asset->Drag it in to the Mesh Slot on the Mesh on the mesh collider->Turn On the Mesh Renderer tab->Select Is Trigger tab
* Go to->Asset->Prefab->VFX->Engines->Drag it into the player Hierarchy
* You Can Change the position of Ship By Changing the Tab View .

**Step:3:**

**Setting up the main camera and lighting for the scene**

* Click on the main camera from hierarchy->Go to Inspector window->go to transform component->Click on Reset Tab->Change the rotation of x-axis as 90->go to camera view->click on the project-> Set it as orthographic->Se the size to 10
* Go to camera->Change the clear flags to solid color->Change the background color as Black
* Setting up the light
* Go to edit->Render setting->change the Ambient light 0.0.0.0(Black in color)
* Go to hierarchy->click on create->Directory light->Rename it with main light->Reset the light position->set the rotation x axis as 20->y axis as-115->
* Select the main light from hierarchy->go to edit menu->Select duplicate->Rename the duplicate as fill light
* Go to Inspector ->reset the rotation of fill light
* Change the light Intensity as 0.05->Change the Colors->Change the rotations x-axis as 5 and y- axis as 125-Duplicatthe fill light->Rename it with Rim Light->Deselect the RimLight FromInspector Window->selectRim Light From Hierarchy->Reset the transform
* Add Empty gameobject to the scene->press Shift+Ctrl+N-> Rename it with Lighting->Reset the transform Component

**Step 4: Adding the background**

* Click on player in the hierarchy->go to transform component ->deselect Player tab
* Create the quad to hold the background Image
* Go to Hierarchy->click on Create->Select Quad->Rename it as Background->reset the game object
* Change the rotat ion of background x-axis as 90 Go to Mesh Collider->Remove
* Now Add The Texture To Our Background: Go To Project ->Assets->Texture->Select Nebula
* Reset the scale of Quad-> set the Scale x as 15

Change the Shedder: go to Shedder ->Unlit->Texture

Click on background in the hierarchy->go to inspector window->change the y position to -10

**Step 5: Moving the player**

* Go to asset->click on create->folder->give the name as Script->press Enter->Now we will create a new script to our player ship->Click on player in the hierarchy->Go to inspector window ->Click on Add component Tab-> select New Script->Give the Name to script as Player Controller->Click on Create and Add Button. The Script is created in c-sharp .
* NowDragthePlayer Controller Script Into the Script Folder
* Open the script folder to view
* Select the script->Click on Open
* The Script will Open in the mono Developer code window Remove all the sample code from the script
* Set x-min value as 6->x-max value as 6->z-min value as 4-.z-mzx value as-4
* From the Inspector Set x- In the Inspector Window do the following settings

->Set x-min Value as 6

->x-max value as -6

->z-min value as 4

->z-max value as -4

->Adjust the tilt value

->Set the speed to the ship

**Step 6: Creating Shots**

* Now Will Create Shots to Our Player
* Now Create new Empty Game Object In the Hierarchy->Press Shift+Cntrl->Give the name to the Game Object (Bolt)
* Click on Bolt From Hierarchy->Reset the game object of Transform to Origin
* Create Quad From Hierarchy->Rename it as VFX-> Reset the Transform position to Origin
* Now Drag The VFX Game Object Into the Bolt-

>Change x-axis position as 90

* Go to Asset->Texture->select FX Lazer
* Go to material folder in the asset->Click on Create tab->Choose Material->give the name as Fx-bolt-orange
* Now will add the Texture Into the Material

Go to Inspector Window ->Click on fx-bolt-orange->click on Select->click on the Texture U want to add into material

* Now go to material->drag the vfx-Bolt-Orange on to the scene.
* Go to Inspector window->fs-bolt-orange->Shader-.Mobile->particles->Additive select Bolt from Hierarchy->Go to inspector->Click on Add Component ->physic->Rigid body->Deselect Use gravity Tab

* Now Go to hierarchy->Click on VFX-> Go to inspector->Select->Mesh Renderer tab
* In Inspector Window->Go to mesh Collider->Click on Setting Button->Click on Remove Component
* In the hierarchy Window->click on Bolt->Go to inspector Window->click on Add Component Button->Select Physics->Capsule Collider
* Go to Capsule Collider Tab in to inspector window ->Change the radius and 0.03->Height 0.58->Direction as Z
* Click on Is trigger tab in collide
* Click on Bolt in the hierarchy-<Go to inspector window->Click on Add Component->Click on New Script ->Give the Name to the Script as Mover->Press Enter
* Go toAssets-> Move the Script File into The Script Folder->open the Mover Script->
* Write the following Lines of Code into the Mover Script->Save the Script and come to the unity Window
* Drag the Bolt Game Object From Hierarchy to the Asset Prefab->Set the Script Speed As 20 Inspector Window.
* Turn Of the Maximize on play Button on the Scene->Click on the Play Button->Drag the Bold Into the Hierarchy To see the How Ship is running

**Step 7: Shot Spawn**

* Select player from hierarchy->go to inspector window->Go to player Controller Script->Click on Setting Tab->Select Edit Script Option
* Add Code into the Script
* Create New Empty Game Object->Name it as Shot Spawn
* Drag the Shot Spawn Game Object into The player
* Now go to player Controller Script And add some lines Of Code

**Program:**

**Done\_BGScroller.cs**

using UnityEngine;

using System.Collections;

Public class Done\_BGScroller : MonoBehaviour

{

public float scrollspeed;

public float tilesizeZ;

private Vector3 startPosition;

void Start ()

{

startPosition = transform.position;

}

void Update ()

{

float newPosition = Mathf.Repeat(Time.time \* scrollspeed, tile

SizeZ);

transform.position = startPosition + Vector3.forward \* newPosi

tion;

}

}

**Done\_DestroyByBoundry.cs**

using UnityEngine;

using System.Collections;

public class Done\_DestroyByBoundary : MonoBehaviour

{

void OntriggerExit (Collider other)

{

destroy(other.gameObject);

}

}

**Done\_DestroyByContact.cs**

using UnityEngine;

Using System.Collections;

public class Done\_DestroyByContact : MonoBehaviour

{

public GameObject explosion;

public GameObject playerExplosion;

public int scoreValue;

private Done\_GameController gameControllers;

void Start ()

{

Gameobject gameControllerobject = GameObject .FindGameObjectWait

hTag ("Gamecontroller");

if (gameControllerObject != null)

{

gameController = gameControllerObject.GetComponent <Done\_G

ameController>();

}

If (gameController == null)

{

Debug. Log ("Cannot find 'GameController’ script”);

}

}

void OnTriggerEnter (Collider other)

{

if (other.tag == “Boundary” || other.tag == "Enemy")

{

return;

}

if (explosion != null)

{

Instantiate(explosion, transform.position, transform.rotat

ion);

}

if (other.tag == "Player")

{

Instantiate(playerExplosion, other.transform.position, oth

er.transform. rotation) ;

gameController.Gameover();

}

gameController.AddScore(scoreValue) ;

Destroy (other.gameObject);

Destroy (gamedOject)

}

}

**Done\_DestroyByTime.cs**

using unityEngines ;

using System.collections;

public class Done\_DestroyByTime :ManoBenaviour

{

(public float 1ifetine;

void start ()

{

Destroy (gameObject, lifetime);

}

}

**Done\_EvasiveManeuver.cs**

using UnityEngine;

using System.Collections;

public class Done\_EvasiveManeuver : MonoBehaviour

{

public Done\_Boundary boundary;

public float title;

public float dodge;

public float smoothing;

public Vector2 startWait;

public Vector2 maneuverTime;

public Vector2 maneuverWait;

private float currentSpeed;

private float targetManeuver;

void Start ()

{

currentspeed = GetComponent <Rigidbody>().velocity.z;

StartCoroutine(Evade());

}

IEnumerator Evade ()

{

yield return new WaitForSeconds(Random.range(startWait.x,startWait.y));

while (true)

{

targetManeuver = Random.Range (1, dodge) \* -

Mathf.Sign (transform. position.x);

yield return new WaitForSecond(Random.Range(maneuverTim

e.x, maneuverTime.y));

targetManeuver = 0;

yieldreturnnew WaitForSecond(Randow.Range(maneuverWait.x,maneuverWait.y));

}

)

vold FixedUpdate ()

{

float newmaneuver = Mathf.MoveTowards (GetComponent<Rigidbody>

().velocity.x, targetManeuver, smoothing \* Time.deltaTime);

GetComponent<Rigidbody>().velocitynew Vector3(newManeuver,0.0f,currentSpeed);

GetComponent<Rigidbody>().position = new vector3

(

Mathf.Clamp(GetConponent<Rigidbody>().position.x, boundary

x.Min, boundary.xMax),

0.0f,

Mathf.Clamp (GetComponent<Rigidbody>().position.z,boundary

.zMin, boundary.zMax)

);

GetComponent<Rigidbody>().rotation = Quaternion.Euler (0, 0, G

etComponent<Rigidbody>().velocity.x \* -tilt);

}

}

**Done\_GameController.cs**

using UnityEngine;

using UnityEngine.SceneManagement;

using System.Collections;

public class Done\_GameController : MonoBehaviour

{

public GameObject[] hazards;

public Vector3 spawnValues;

public int hazardcount;

public float spawnWait;

public float startwait;

public float waveWait;

public GUIText scoreText;

public GUIText restartText;

Public GUIText gameOverText;

Private bool gameOver;

Private bool restart;

private int score;

void Start ()

{

gameove= false;

restart = false;

restartText.text = "";

gameOverText.text = "";

UpdateScore ();

StartCoroutine (SpawnWaves ());

}

void Update ()

{

if (restart) ;

(

if (Input.GetkeyDown (KeyCode.R));

{

SceneManager.LoadScene(SceneManager.GetActivescene().b

uildIndex);

}

}

}

IEnumerator SpawnWaves ()

{ |

yield return new WaitForSeconds (startWait);

while (true)

{

for (inti=0;i<hazardCount; i++)

{

GameObject hazard = hazards [Random.Range (0, hazards.Length)];

Vector3 spawnPosition = new Vector3 (Random.Range (-

spawnValues.x, spawnValues.x), spawnValues.y, spawnValues.z); |

Quaternion spawnRotation = Quaternion. identity;

Instantiate (hazard, spawnPosition, spawnRotation);

yield return new WaitForSeconds (spawnWait) ;

}

yield return new WaitForSeconds (waveWait);

if (gameOver)

{

restartText.text = "Press 'R' for Restart";

restart = true;

break;

}

}

}

public void AddScore(int newScorevalue)

{

score+=newscoreValue;

Updatescore();

}

void updatescore ()

{

scoreText.text="score:"+score;

}

public void GameOver ()

{

gameoverText.text="Game Over!";

gameOver = true;

}

}

**Done\_Mover.cs**

using UnityEngine;

using System.Collections;

public class Done\_Mover : MonoBehaviour

{

public float speed;

void Start ()

{

GetComponent<Rigidbody>().velocity = transform.forward \* speed;

}

}

**Done\_PlayerController.cs**

Using UnityEngine;

using System.Collections;

[System.Serializable]

public class Done\_Boundary

{

Public float xmin, xMax, zMin, zMax;

}

public class Done Playercontroller:MonoBehaviour

{

public float speed;

public float tilt; .

public Done\_Boundary boundary;

public GameObject shot;

public Transform shotSpawn;

public float fireRate;

private float nextFire;

void Update ()

{

if (Input.GetButton("Firel") && Time.time > nextFire)

{

nextFire = Time.time + fireRate;

Instantiate(shot, shotSpawn.position, shotSpawn.rotation);

GetComponent<AudioSource>().Play ();

}

}

void FixedUpdate ()

{

float moveHorizontal = Input.GetAxis ("Horizontal");

float moveVertical = Input.Getaxis ("Vertical");

Vector3 movement = new Vector3 (moveHorizontal,0.0f, moveVertical);

movement \* speed;

GetComponent<Rigidbody>().velocity=movement\*speed;

GetComponent<Rigidbody>().position = new Vector3

(

Mathf.Clamp (GetComponent<Rigidbody>().position.x, boundar

y.xMin,boundary.xMax),

0.0f,

Mathf.Clamp (GetComponent<Rigidbody>().position.z, boundar

y.zMin, boundary.zMax)

);

Getcomponent <Rigidbody>().rotation=Quaternion.Euler(0.0f,0.0f,

GetComponent <Rigidbody>().velocity.x \* -tilt);

**Done\_RandomRotator.cs**

using UnityEngine;

using System.Collections;

public class Done\_RandomRotator : MonoBehaviour

{

public float tumble;

void Start ()

{

GetComponent<Rigidbody>().angularvelocity = Random.insideunits

phere \* tumble;

}

}

**Done\_WeaponController.cs**

Using UnityEngine;

using System.Collections;

public class Done\_WeaponController : MonoBehaviour

{

public GameObject shot;

public Transform shotSpawn;

public float fireRate;

public float delay;

void Start ()

{

InvokeRepeating ("Fire", delay, fireRate);

}

void Fire ()

{

Instantiate(shot, shotSpawn.position, shotSpawn.rotation);

GetComponent <AudioSource>().Play();

}

}

**Conclusion**: