**Practical No. 1:**

**Aim: Setup DirectX 11, Window Framework and Initialize Direct3D Device**

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

**Step 1:**

1. Create new project, and select “Windows Forms Application”, select .NET Framework as 2.0 in Visuals C#.
2. 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();

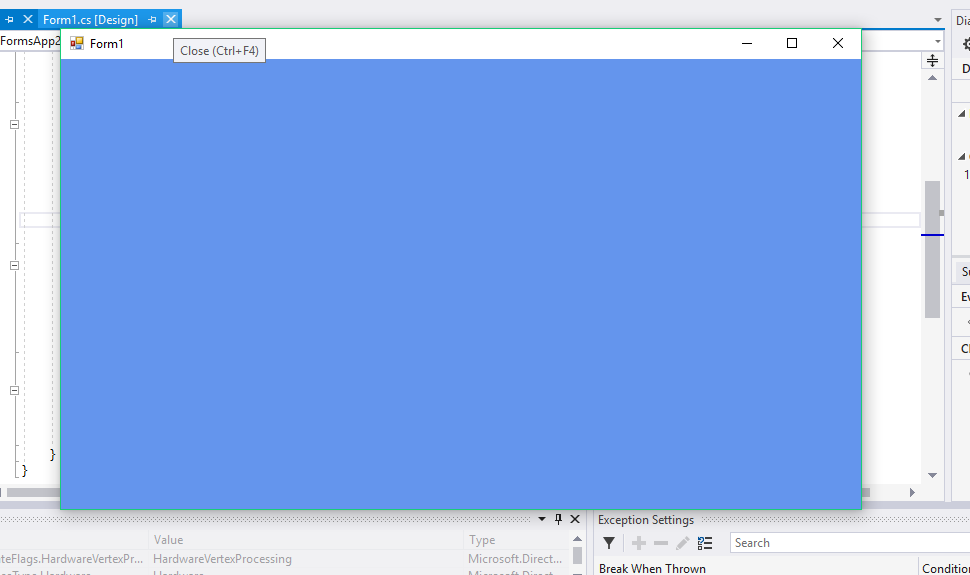
}

}

}

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

**Output:**



**Practical No. 2:**

**Aim: Draw a triangle using Direct3D 11**

**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)

{

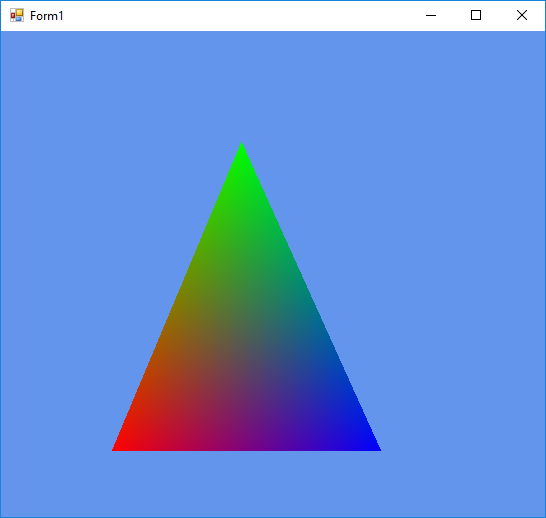
Render();

}

}

}

**Output:**



**Practical No. 3:**

**Aim: Texture the triangle using Direct3D 11**

**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 = new Device(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();

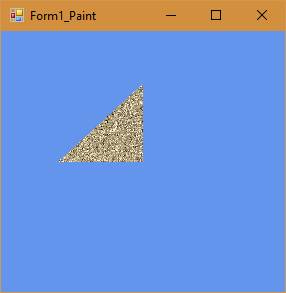
device.Present();

}

}

}

**Output:**



**Practical No. 4:**

**Aim: Programmable Diffuse Lightning using Direct3D 11**

**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 = new Device(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)

{

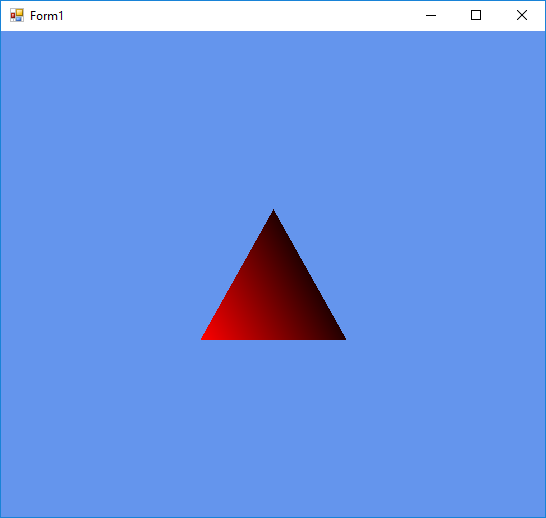
Render();

}

}

}

**Output:**



**Practical No. 5:**

**Aim: Loading models into DirectX 11 and rendering**

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();

}

}

}

**Output:**

