الدرس الخامس

أهلا بكم في الدرس الخامس من سلسلة دروس تعلم ال 3D Xna السلسلة الأولى، في هذا الدرس سوف نتحدث عن التدوير و النقل.

سوف نقوم بجعل مثلثنا يدور ويتحرك. بما أننا نستخدم فضاء إحداثيات العالم، من السهل عمل ذلك. دعنا أولا نضيف متغير بإسم 'angle' إلى الصنف الخاص بنا من أجل تخزين زاوية التدوير الحالية. فقط قم بإضافة السطر التالي إلى المتغيرات.

```
private float angle = 0f;
```

الآن، يجب علينا زيادة هذا المتغير بقيمة 0.005f في كل إطار. الدالة Update هي المكان الأمثل لوضع هذا الكود، بما أنه يتم إستدعائها 60 مره في كل ثانية:

```
angle += 0.005f;
```

الآن و بما أن الزاوية تزداد بشكل تلقائي، كل ما نحتاجه هو أن نقوم بعمل تدوير لنظام الإحداثيات. أرجو أنك ما زلت تذكر من حصص الرياضيات أن هذا يتم بإستخدام مصفوفات التحويل ☺ لحسن الحظ، كل ما تحتاج لعمله هو أن تحدد محور التدوير و زاوية الدوران. كل الباقي يتم عمله بواسطة ال XNA!

التدوير يتم تخزينه في ما يسمى مصفوفة العالم، في مصفوفة بإسم worldMatrix. قم بإضافة الكود التالي إلى الدالة Draw، قبل إستدعاء ال effect.Begin:

```
Matrix worldMatrix = Matrix.CreateRotationY(3 * angle);
effect.Parameters["xWorld"].SetValue(worldMatrix);
```

السطر الأول يقوم بإنشاء مصفوفة العالم، التي تحتوي على التدوير حول محور Y. السطر الثاني يقوم بتمرير مصفوفة العالم إلى التأثير "effect"، التي تحتاجها لإتمام العمل. من الآن فصاعدا، كل شيئ سوف نقوم برسمه سوف يتم تدويره حول محور Y بالقيمة المخزنة حاليا في المتغير 'angle'!

عندما تقوم بتشغيل البرنامج، سوف ترى أن المثلث يدور حول النقطة (0,0,0)! هذا بالتأكيد بسبب أن محور Y يمر من هذه النقطة، لذا النقطة (0,0,0) هي النقطة الوحيدة في المثلث التي تبقى كما هي. الآن تخيل أننا نريد أن نجعل المثلث يدور حول مركزه. أحد الإحتمالات هي من خلال إعادة تعريف المثلث بحيث نجعل النقطة (0,0,0) هي مركز المثلث. الحل الأفضل هو من خلال تحريك (= نقل "Translate") المثلث قليلا إلى اليسار و إلى الأسفل، و بعدها تدويره. لعمل ذلك، ببساطة قم أولا بضرب مصفوفة العالم بمصفوفة النقل:

```
Matrix worldMatrix = Matrix.CreateTranslation(-20.0f/3.0f, -10.0f / 3.0f, 0) *
Matrix.CreateRotationZ(angle);
```

هذا سوف يحرك المثلث بحيث يصبح مركزه في نقطة الأصل الخاصة بالعالم (0,0,0). بعدها يتم تدوير المثلث حول تلك النقطة، على محور Z، ما يعطينا النتيجة المطلوبة.

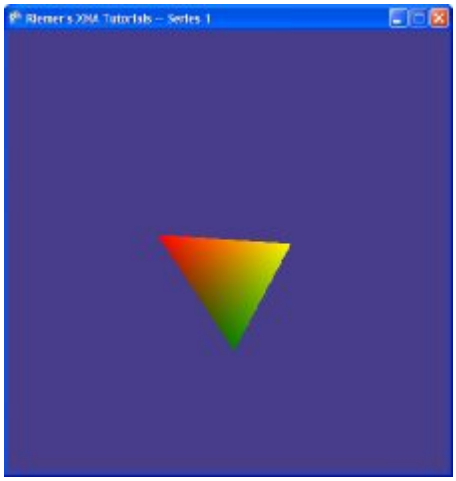
لاحظ ترتيب العمليات. حاول وضع عملية النقل **بعد** عملية التدوير. سوف تلاحظ أن المثلث أصبح يدور حول نقطة واحدة، و يتحرك لليسار و الأسفل. سبب ذلك أن في ضرب المصفوفات M1*M2 لا يساوي M2*M1!.

بإمكانك ببساطة تغيير الكود بحيث يصبح المثلث يدور حول محور X أو محور Y. تذكر أن أحدى نقاط المثلث لدينا قيمة إحدائي Z الخاص بها هو 5-، و هو ما يفسر عدم دوران المثلث بشكل متماثل في بعض الأحيان.

شيئ أعقد قليلا هو مصفوفة CreateFromAxisAngle، حيث يكون بإستطاعتك أن تحدد محور التدوير الخاص بك:

```
Vector3 rotAxis = new Vector3(3*angle, angle, 2*angle);
rotAxis.Normalize();
Matrix worldMatrix = Matrix.CreateTranslation(-20.0f/3.0f, -10.0f / 3.0f, 0) *
Matrix.CreateFromAxisAngle(rotAxis, angle);
```

هذا سوف يتيح لنا تدوير المثلث حول محور قابل للتغيير في أي وقت. السطر الأول يحدد المحور (الذي يتغير في كل إطار لأنه يعتمد على المتغير angle). السطر الثاني يقوم بعمل تطبيع "Normalization" للمحور، حيث يلزم لكي تعمل الدالة CreateFromAxisAngle بشكل صحيح (الدالة Normalize() تقوم بتغيير إحداثيات المتجه، بحيث تصبح المسافة بين المحور و النقطة (0,0,0) تساوي 1 بالضبط).



بإمكانك محاولة تطبيق التمرين التالي من أجل ممارسة ما قد تعلمته:

- حاول تدوير المثلث بزاوية 180 درجة حول الضلع السفلي له.

الكود حتى الآن:

```
using System;
using System.Collections.Generic;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.GamerServices;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Net;
using Microsoft.Xna.Framework.Storage;

namespace XNAtutorial
{
    public class Game1 : Microsoft.Xna.Framework.Game
```

```
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;
        GraphicsDevice device;
        Effect effect;
        VertexPositionColor[] vertices;
        VertexDeclaration myVertexDeclaration;

        Matrix viewMatrix;
        Matrix projectionMatrix;

        private float angle = 0f;

        public Game1()
        {
            graphics = new GraphicsDeviceManager(this);
            Content.RootDirectory = "Content";
        }

        protected override void Initialize()
        {
            graphics.PreferredBackBufferWidth = 500;
            graphics.PreferredBackBufferHeight = 500;
            graphics.IsFullScreen = false;
            graphics.ApplyChanges();
            Window.Title = "Riemer's XNA Tutorials -- Series 1";

            base.Initialize();
        }

        protected override void LoadContent()
        {
            device = graphics.GraphicsDevice;
            spriteBatch = new SpriteBatch(GraphicsDevice);

           effect = Content.Load<Effect> ("effects");
           SetUpVertices();
           SetUpCamera();
       }

       private void SetUpVertices()
       {
           vertices = new VertexPositionColor[3];

           vertices[0].Position = new Vector3(0f, 0f, 0f);
           vertices[0].Color = Color.Red;
           vertices[1].Position = new Vector3(10f, 10f, 0f);
           vertices[1].Color = Color.Yellow;
           vertices[2].Position = new Vector3(10f, 0f, -5f);
           vertices[2].Color = Color.Green;

           myVertexDeclaration = new VertexDeclaration(device,
VertexPositionColor.VertexElements);
       }

       private void SetUpCamera()
       {
           viewMatrix = Matrix.CreateLookAt(new Vector3(0, 0, 50), new Vector3(0, 0, 0),
new Vector3(0, 1, 0));
           projectionMatrix = Matrix.CreatePerspectiveFieldOfView(MathHelper.PiOver4,
device.Viewport.AspectRatio, 1.0f, 300.0f);
       }

       protected override void UnloadContent()
       {
       }

       protected override void Update(GameTime gameTime)
       {
           if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed)
               this.Exit();
```

```
            angle += 0.005f;

            base.Update(gameTime);
        }

        protected override void Draw(GameTime gameTime)
        {
            device.Clear(Color.DarkSlateBlue);
            device.RenderState.CullMode = CullMode.None;

            Vector3 rotAxis = new Vector3(3 * angle, angle, 2 * angle);
            rotAxis.Normalize();
            Matrix worldMatrix = Matrix.CreateTranslation(-20.0f/3.0f, -10.0f / 3.0f, 0)
* Matrix.CreateFromAxisAngle(rotAxis, angle);

            effect.CurrentTechnique = effect.Techniques["Colored"];
            effect.Parameters["xView"].SetValue(viewMatrix);
            effect.Parameters["xProjection"].SetValue(projectionMatrix);
            effect.Parameters["xWorld"].SetValue(worldMatrix);

            effect.Begin();
            foreach (EffectPass pass in effect.CurrentTechnique.Passes)
            {
                pass.Begin();

                device.VertexDeclaration = myVertexDeclaration;
                device.DrawUserPrimitives(PrimitiveType.TriangleList, vertices, 0, 1);

                pass.End();
            }
            effect.End();

            base.Draw(gameTime);
        }
    }
}
```