Course 3D_MDX: 3D-Graphics with Managed DirectX 9.0
Chapter C3: Cylinder with Texture

Projekt texture1

Diese Übungsaufgabe ist eine kurze, übersichtliche Fassung eines Direct3D-Tutorials von Microsoft: `Tutorial4`. Sie finden es unter `C:\DXSDK\Samples\Managed\Direct3D\Tutorials\Tutorial5`.

Main Menu nach dem Start von VS 2005: `File -> New Project... -> Templates: Windows Application`
`Name: texture1 -> Location: C:\temp -> Create directory for solution:` ausschalten `-> OK`
Löschen Sie die Files `Program.cs` und `Form1.Designer.cs` und den Inhalt von `Form1.cs`, wie es in den Kapiteln 2DCisC1 bis 2DCisC4 beschrieben wurde.

Falls das `Solution Explorer -` Fenster nicht schon offen ist, öffnen Sie es über das Hauptmenü: `View -> Solution Explorer`.
Im `Solution Explorer -` Fenster klicken Sie auf das Pluszeichen vor `texture1`. Es öffnet sich ein Baum. Ein Ast heißt "`References`". Klicken Sie mit der **rechten** Maustaste auf `References` und dann mit der **linken** Maustaste auf `Add Reference...`. Es öffnet sich eine `Add Reference` Dialog Box. Scrollen Sie abwärts, bis Sie den Component Name: `Microsoft.DirectX Version 1.0.2902.0` sehen.
Markieren Sie durch Linksklick diese Referenz und (bei gedrückter der Strg-Taste) die weiter unten stehende Referenz `Microsoft.DirectX.Direct3D Version 1.0.2902.0`. Verlassen Sie die `Add Reference` Dialog Box mit `OK`.
Kontrollieren Sie, ob jetzt im Solution Explorer Fenster unter `texture1 -> References` (unter anderen) die beiden Referenzen `Microsoft.DirectX` und `Microsoft.DirectX.Direct3D` stehen.

Form1, OnResize, OnTimer

Schreiben in das leere Codefenster `Form1.cs` folgenden Code:

```
using System;
using System.Drawing;
using System.Windows.Forms;
using Microsoft.DirectX;
using Microsoft.DirectX.Direct3D;

public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  static Device device = null;
  static float xAngle, yAngle, zAngle;
  static Vector3 xAxis = new Vector3( 1, 0, 0 );
  static Vector3 yAxis = new Vector3( 0, 1, 0 );
  static Vector3 zAxis = new Vector3( 0, 0, 1 );
  VertexBuffer      vertexBuffer;
  Bitmap  bmp     = null;
  Texture texture = null;
  const Int32   N = 100; //N must be an even no. 6, 8, 10, etc
  CustomVertex.PositionNormalTextured[] vv = new CustomVertex.PositionNormalTextured[N];
  Timer myTimer = new Timer();

  public Form1()
  { MenuItem miRead = new MenuItem( "Read", new EventHandler( MenuFileRead ) );
    MenuItem miExit = new MenuItem( "Exit", new EventHandler( MenuFileExit ) );
    MenuItem miFile = new MenuItem( "File", new MenuItem[] { miRead, miExit } );
    Menu = new System.Windows.Forms.MainMenu( new MenuItem[] { miFile } );
```

```
  try { bmp = (Bitmap)Image.FromFile( "C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp" ); }
  catch { try { //Delete this inner try-catch clause if you have no Internet connection running.
    String s = "http://www.miszalok.de/Images/tiger.bmp";
    System.Net.WebRequest  webreq = System.Net.WebRequest.Create( s );
    System.Net.WebResponse webres = webreq.GetResponse();
    System.IO.Stream       stream = webres.GetResponseStream();
    bmp = (Bitmap)Image.FromStream( stream );
  } catch {}; }; // end of inner and outer try-catch clauses
  Text = "D3DTexture:    Use the File menu to read new textures !";
  //TriangleStrip forming a cylinder
  //radius = 1; axis = Z-axis; top = 1; bottom = -1; => height = 2;
  //in order to see the vertices, replace the the TriangleStrip by a LineStrip in OnTimer(...)
  float arcus_increment = (float)(2.0 * Math.PI / (N-2)); //cylinder angular increment
  float    tu_increment = (float)(1.0           / (N-2)); //texture horiz. increment
  Vector3 v = new Vector3();
  for (int i = 0; i < N; i++)
  { float arcus = i * arcus_increment;
    v.X = (float)Math.Cos( arcus );
    v.Y = (float)Math.Sin( arcus );
    if ( i%2 == 0 ) v.Z =  1f;
    else            v.Z = -1f; //zigzag between top and bottom
    vv[i].Position = v; //vertex = (cos,sin,+1) or (cos,sin,-1)
    v.Z =  0;           //cylinder normals have no Z-component
    vv[i].Normal   = v; //normal = (cos,sin,0)
    vv[i].Tu = i * tu_increment; //horizontal texture position
    if ( i%2 == 0 ) vv[i].Tv = 0f;
    else            vv[i].Tv = 1f; //vertical zigzag on texture image
  }
  myTimer.Tick += new EventHandler( OnTimer );
  myTimer.Interval = 1;
  ClientSize = new Size( 400, 300 ); //Calls OnResize( ... )
}

protected override void OnResize( System.EventArgs e )
//Whenever the window changes we have to initialize Direct3D from scratch
{ myTimer.Stop();// stop the timer during initialization
  try
  { //get information from the operating system about its current graphics properties
    PresentParameters presentParams = new PresentParameters();
    //we have to set two flags
    presentParams.Windowed = true;                 //no full screen display
    presentParams.SwapEffect = SwapEffect.Discard; //no swap buffer
    presentParams.EnableAutoDepthStencil = true;   //with depth buffer
    presentParams.AutoDepthStencilFormat = DepthFormat.D16; //16 bit depth
    //Create a new D3D-device that serves as canvas.
    if ( device != null ) device.Dispose(); //free the old canvas if any
    device = new Device( 0, DeviceType.Hardware, this,
                         CreateFlags.SoftwareVertexProcessing, presentParams );
    //Create a white material.
    Material mtrl = new Material();
    mtrl.Diffuse = mtrl.Ambient = Color.White;
    device.Material = mtrl;
    //Create a single, white, directional, diffuse light source and a gray ambient light.
    //Many lights may be active at a time. (Notice: Each one slows down the render process.)
    device.Lights[0].Type = LightType.Directional;
    device.Lights[0].Diffuse = System.Drawing.Color.White;
    device.Lights[0].Direction = new Vector3( 0, 1, 1 );
    device.Lights[0].Enabled = true; //turn it on
    //Finally, turn on some ambient light that scatters and lights the object evenly
    device.RenderState.Ambient = System.Drawing.Color.FromArgb( 0x202020 );
    //setup texture
    if ( texture != null ) texture.Dispose();
    if ( bmp != null) texture = Texture.FromBitmap( device, bmp, 0, Pool.Managed );
    device.SetTexture( 0, texture );
    //set up the transformation of world coordinates into camera or view space
    device.Transform.View = Matrix.LookAtLH(
      new Vector3( 0f, 0f,-4f ),   //eye point 4.0 in front of the canvas
      new Vector3( 0f, 0f, 0f ),   //camera looks at point 0,0,0
      new Vector3( 0f, 1f, 0f ) ); //worlds up direction is the y-axis
    //set up the projection transformation using 4 parameters:
    //1.: field of view = 45 degrees; 2.: aspect ratio = heigth / width = 1 = square window;
    //3.: near clipping distance = 0; 4.: far clipping distance = 10;
    device.Transform.Projection = Matrix.PerspectiveFovLH((float)Math.PI/4, 1f, 1f, 10f );
```

```
      device.RenderState.CullMode = Cull.None;
      device.RenderState.Lighting = true;
      //set up the property that fills the triangle with colors
      device.VertexFormat = CustomVertex.PositionNormalTextured.Format;
      //create a new vertex buffer and connect it to the device
      if ( vertexBuffer != null ) vertexBuffer.Dispose(); //free the old vertexBuffer if any
      vertexBuffer = new VertexBuffer( typeof(CustomVertex.PositionNormalTextured), N,
                                       device, 0,
                                       CustomVertex.PositionNormalTextured.Format,
                                       Pool.Default );
      //copy the coordinates and colors of "vv" into the vertex buffer
      vertexBuffer.SetData( vv, 0, LockFlags.None );
      device.SetStreamSource( 0, vertexBuffer, 0 );
      myTimer.Start();//start the timer again
    }
    catch (DirectXException) { MessageBox.Show( "Could not initialize Direct3D." ); return;
}
  }

  protected static void OnTimer( Object myObject, EventArgs myEventArgs )
  { if (device == null) return;
    //throw the old image away
    device.Clear( ClearFlags.Target | ClearFlags.ZBuffer, Color.Gray, 1f, 0 );
    //rotate with 3 angular velocities
    xAngle += 0.1f;
    yAngle += 0.02f;
    zAngle += 0.02f;
    device.Transform.World  = Matrix.RotationAxis( xAxis, xAngle );
    device.Transform.World *= Matrix.RotationAxis( yAxis, yAngle );
    device.Transform.World *= Matrix.RotationAxis( zAxis, zAngle );
    //draw on the canvas
    device.BeginScene();
       device.DrawPrimitives( PrimitiveType.TriangleStrip, 0, N-2 );
       //Experiment: Replace the TriangleStrip by a LineStrip as follows:
       //device.DrawPrimitives( PrimitiveType.LineStrip, 0, N-2 );
    device.EndScene();
    device.Present(); //show the canvas
  }

  void MenuFileRead( object obj, EventArgs ea )
  { try
    { OpenFileDialog dlg = new OpenFileDialog();
      dlg.Filter = "images |*.bmp;*.gif;*.jpeg;*.jpg;*.png;*.tif;*.tiff;*.bmp|" +
                   "Windows Bitmap (*.bmp)|*.bmp|" +
                   "Graphics Interchange Format (*.gif)|*.gif|" +
                   "Joint Photographic Experts Group (*.jpg)|*.jpg;*.jpeg|" +
                   "Portable Network Graphics (*.png)|*.png|" +
                   "Tag Image File Format (*.tif)|*.tif;*.tiff|" +
                   "All files (*.*)|*.*";
      if ( dlg.ShowDialog() != DialogResult.OK ) return;
      bmp = (Bitmap)Image.FromFile( dlg.FileName );
      OnResize( null );
    }
    catch
    { MessageBox.Show ( "Cannot read", "D3DTexture", MessageBoxButtons.OKCancel ); }
  }

  void MenuFileExit( object obj, EventArgs ea )
  { Application.Exit(); }

}
```

Klicken Sie `Debug` -> `Start Without Debugging Ctrl F5`. Erproben Sie das Programm durch ziehen an allen Fensterrändern.

Embedded Resource = Eingebaute Daten

Unser Programm braucht beim Starten das Bild `"tiger.bmp"` und im Konstruktor `public Form1()` sucht es zunächst auf der Festplatte unter `"C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp"` und, wenn dort nichts zu finden ist, im Internet unter `"http://www.miszalok.de/Images/tiger.bmp"`.
Problem: Wenn beides nicht funktioniert, startet unser Textur-Programm ohne Textur.
Lösung: Bauen Sie `"tiger.bmp"` in das Programm fest ein:
1. Wenn es nicht schon offen ist, öffnen Sie das `Solution Explorer`-Fenster und klicken mit der rechten Maustaste auf den Ast `texture1`. Im Kontextmenü klicken Sie `Add`. Im nächsten Kontextmenü klicken Sie `Add Existing Item...`. Im sich jetzt öffnenden `File Dialog` klicken sie ganz unten am Ende der `Files of type:`-Textbox auf das Dreieck und wählen `Image Files (*.bmp, ... )`. Dann suchen Sie auf Ihrer Festplatte unter `"C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp"` oder sonstwo nach `tiger.bmp` und binden es ein. `tiger.bmp` muss jetzt als Ast von `texture1` im `Solution Explorer`-Fenster stehen. Beim Rechtsklick auf diesen Ast erhalten Sie ein Kontextmenü. Klicken Sie auf dessen letzte Zeile `Properties`. Im Eigenschaftenfenster ändern Sie die Eigenschaft `Build Action` von `Content` auf `Embedded Resource`.
2. Löschen Sie im Konstruktor `public Form1()` folgenden Block von Zeilen:

```
try { bmp = (Bitmap)Image.FromFile( "C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp"
); }
catch { try { //Delete this inner try-catch clause if you have no Internet
connection running.
String s = "http://www.miszalok.de/Images/tiger.bmp";
System.Net.WebRequest webreq = System.Net.WebRequest.Create( s );
System.Net.WebResponse webres = webreq.GetResponse();
System.IO.Stream stream = webres.GetResponseStream();
bmp = (Bitmap)Image.FromStream( stream );
} catch {}; }; // end of inner and outer try-catch clause<
```

und ersetzen Sie das enstandene Loch durch eine einzige Zeile:

```
bmp = new Bitmap( typeof( Form1 ), "texture1.tiger.bmp" ); //read the embedded
resource
```

Jetzt enthält das ausführbare `texture1.exe` einen festen Bestandteil, nämlich die initiale Textur.
Vorteil: `"tiger.bmp"` ist beim Starten immer da, unabhängig von der Festplattenstruktur und vom Internet.
Nachteil: Das ausführbare `texture1.exe` (bisher 20 KB) schwillt um die Bildgröße (=60 KB) auf 80 KB an.

Weitere Aufgaben

1. Lesen Sie beliebige Bilder zur Laufzeit ein. Eine Bildersammlung finden Sie unter `C:\DXSDK\Samples\Media\.`
2. Ersetzen Sie in der 5.-letzte Zeile von `protected static void OnTimer( Object myObject, EventArgs myEventArgs )` den Bezeichner `TriangleStrip` durch `LineStrip`.
3. Ersetzen Sie im Konstruktor `public Form1()` in der Zeile `float tu_increment = (float)(1.0 / (N-2));` die `1.0` durch eine `2.0`, dann `3.0` etc.
4. Ersetzen Sie im Konstruktor `public Form1()` in der Zeile `else vv[i].Tv = 1f;` die `1f` durch eine `2f`, dann `3f` etc.
5. Bremsen Sie die Animation durch hoch setzen von `myTimer.Interval = 100;` im Konstruktor.
6. Beschleunigen Sie die Animation durch hoch setzen von `xAngle, yAngle` und/oder `zAngle` in `OnTimer`.
7. Drehen Sie die Röhre um nur eine Achse in `OnTimer`.
8. Geben Sie dem diffusen Licht eine andere Farbe.
9. Geben Sie dem ambienten Licht eine andere Farbe.
10. Ändern Sie die Position der Lichtquelle: nur von oben, nur von unten, nur von links etc.
11. Treten Sie mit dem Augenpunkt weiter zurück von `( 0f, 0f,-4f )` nach `( 0f, 0f,-100f )` und näher heran mit `( 0f, 0f,-2f )`.
12. Lesen Sie die Kommentare und versuchen Sie, den Sinn der Befehle zu verstehen.
13. Sie finden Erklärungen und Kommentare zu dieser Übung unter **http://msdn.microsoft.com/library/default.asp**.
Verzweigen Sie in dem Baum auf der linken Seite auf: Win32 and COM Development → Graphics and Multimedia → DirectX → SDK Documentation → DirectX 9.0 SDK Update (February 2006) Managed → DirectX SDK → Introducing DirectX 9.0 → Direct3D Graphics → Getting Started with Direct3D → Direct3D Tutorials → Tutorial 5: Using Texture Maps.