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

Project texture1

This chapter is a summary of a Direct3D-Tutorial from Microsoft: `Tutorial5`. You find the tutorial here: `C:\DXSDK\Samples\Managed\Direct3D\Tutorials\Tutorial5`.

Main Menu after starting VS 2005: `File` → `New Project...` → `Templates: Windows Application Name: texture1` → `Location: C:\temp` → `Create directory for solution:` switch it off → `OK`
Delete the files `Program.cs` and `Form1.Designer.cs` and the content of `Form1.cs`, as described in the chapters 2DCisC1 to 2DCisC4.

If You can't find a `Solution Explorer`-window, open it via the main menu: `View` → `Solution Explorer`.
Inside the `Solution Explorer`-window click the plus-sign in front of `texture1`. A tree opens. Look for the branch "`References`". **Right**-click `References` and **left**-click `Add Reference...`. An `Add Reference` dialog box opens. Scroll down to the component name: `Microsoft.DirectX Version 1.0.2902.0`.
Highlight this reference by a left-click and (holding the `Strg`-key pressed) the reference `Microsoft.DirectX.Direct3D Version 1.0.2902.0` somewhere below. Quit the `Add Reference` dialog box with `OK`.
Check if both references `Microsoft.DirectX` and `Microsoft.DirectX.Direct3D` are now visible inside the Solution Explorer window underneath `texture1` → `References`.

Form1, OnResize, OnTimer

Write the following code to `Form1.cs`:

```
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.
    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 TriangleStrip by 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 rendering.)
    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
      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(); }

}
```

Click Debug → `Start Without Debugging Ctrl F5`. Try to drag all window borders.

Embedded Resource

This program needs at first an image `"tiger.bmp"` and the constructor `public Form1()` looks for it at first on the hard disk under `"C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp"` and when it can't be found (DirectX SDK not installed or installed in another directory), it looks for it on the internet: `"http://www.miszalok.de/Images/tiger.bmp"`.
Problem: When both attempts fail, the program starts without texture.
Solution: Build in `"tiger.bmp"` as an inseparable part of the program:
1. If not already visible, open the `Solution Explorer`-window and right-click the branch `texture1`.
A context menu appears and You click `Add`. In the next context menu click `Add Existing Item...`.
In the file dialog window `File Dialog` click at the lowest end of the `Files of type:`-TextBox on the triangle and choose `Image Files (*.bmp, ... )`.
Now investigate Your hard disk (`"C:\\DXSDK\\Samples\\Media\\Tiger\\tiger.bmp"` ?)
in order to find `tiger.bmp` and bind it to `texture1`.
"`tiger.bmp`" must now be listed as one of the branches of `texture1` of the `Solution Explorer`-tree.
Right click this branch to obtain a context menu. Click its last line `Properties`.
In the property window change `Build Action` from `Content` to `Embedded Resource`.
2. Inside the constructor `public Form1()` You can now delete a block of lines:

```
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.
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<
```

Replace this complete block by one single line:

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

Now the executable `texture1.exe` contains its initial texture `tiger.bmp` as an integral part of the file.
Advantage: You can be sure that `"tiger.bmp"` is present at start. It doesn't depend neither on the content and structure of a hard disk nor on the presence of an internet connection anymore.
Disadvantage: The file `texture1.exe` (formerly 20 KB) inflates by the size of the image (=60 KB) to 80 KB.

Exercises

1. Read arbitrary images at run time via the `File`-menu. You can find an image collection under `C:\DXSDK\Samples\Media\`.
2. In the 5th next to the last line of `protected static void OnTimer( Object myObject, EventArgs myEventArgs )` replace `TriangleStrip` by `LineStrip` and try out how the vertex framework changes when You vary the constant `N` (in the 10th line of the class declaration of `Form1`) from 6 to 8, to 12, to 48 etc. to 100.
3. Inside the constructor `public Form1()` in the line `float tu_increment = (float)(1.0 / (N-2));` replace `1.0` by `2.0` and by `3.0` etc.
4. Inside the constructor `public Form1()` in the line `else vv[i].Tv = 1f;` replace `1f` by `2f` and by `3f` etc.
5. Slow the animation down by increasing `myTimer.Interval = 100;` inside the constructor.
6. Accelerate the animation by increasing `xAngle, yAngle` and/or `zAngle` in `OnTimer(..)`.
7. Rotate the cylinder around just one axis in `OnTimer(..)`.
8. Light the cylinder with another diffuse color.
9. Light the cylinder with another ambient color.
10. Vary the position of the light source: from top only, from bottom only, from left only etc.
11. Step back with the eye point from `( 0f, 0f,-4f )` to `( 0f, 0f,-100f )` and get closer with `( 0f, 0f,-2f )`.
12. Read the comments **C3DCisC2_Comment.htm** and try to understand the sense of the code lines.
13. You can find more explanations and comments about this chapter here:
http://msdn.microsoft.com/library/default.asp.
Caution: Mozilla Firefox doesn't correctly display the tree on the left side. Use the Internet Explorer here !
Click trough the tree on the left side:
Win32 and COM Development → Graphics and Multimedia → DirectX → SDK Documentation → DirectX SDK Managed → DirectX SDK → Introducing DirectX 9.0 → Direct3D Graphics → Getting Started with Direct3D → Direct3D Tutorials → Tutorial 5: Using Texture Maps.