Course 3DC7: 3D-Computer Graphics with C++/MFC 7.0
Chapter C3: The DirectDrawMesh Project

based on: **http://msdn.microsoft.com/library/default.asp?url/code/list/directx.asp**
Branch into: `Graphics and Multimedia -> DirectX -> SDK Documentation -> DirectX 9.0 (C++) -> DirectX Graphics -> Programming Guide -> Tutorials, Samples, Tools, and Tips -> Tutorials -> Tutorial 6: Using Meshes`

Download the DirectX 9.0 Software Development Kit

Falls Sie bereits Teile und/oder alte Komponenten von DirectX auf Ihrem Rechner haben, dann deinstallieren Sie diese zuerst.
Dann downloaden Sie das `Software Development Kit` (186MB!) von
http://msdn.microsoft.com/library/default.asp?url=/downloads/list/directx.asp
und installieren es **in einem eigenen Verzeichnis C:\DXSDK oder D:\DXSDK**.

Projekt ddmesh1 mit leerem Fenster

`Microsoft Visual Studio.NET starten`
`File – New – Project – Project Types: Visual C++ Projects, Templates: MFC Application`
Name: `ddmesh1`
Location: `C:\temp`
Button `OK` unten Mitte klicken
Links unter `Overview` auf `Application Type` klicken.
Schritt1: **`Single document`** einschalten
Schritt 2: **`Document/View architecture support`** Checkbox **ausschalten**
Schritt 3: `Finish`
Schritt 4: Klicken Sie im Hauptmenu von VS.NET auf den Menupunkt `View` und dann auf den Unterpunkt `Class View Ctrl+Shift+C`.
Sie sehen im rechten Bereich des VS.NET-Hauptfensters das Unterfenster `Class View - ddmesh1`.
An dessen unteren Rand die Registerkarte `Class View` klicken.
Jetzt sehen Sie oben die Zeile `+ ddmesh1` , klicken Sie mit der rechten Maustaste auf `ddmesh1`.
Es öffnet sich ein Kontextmenü. Klicken Sie ganz unten auf `Properties`. Es öffnet sich ein Fenster `ddmesh1 Property Pages`.
Stellen Sie im Kopf von `ddmesh1 Property Pages` die `Configuration:` auf `Debug`.
4a: Auf der linken Seite der `ddmesh1 Property Pages` öfnnen Sie den Eintrag `Linker`. Klicken Sie auf den Untereintrag `Input`.
4b: Auf der rechten Seite der `ddmesh1 Property Pages` erscheint eine Tabelle. In deren erster Zeile steht `Additional Dependencies`.
4c: Dahinter in das freie Feld in der zweiten Spalte schreiben Sie `d3d8.lib d3dx8.lib` .
4d: In der dritten Zeile des gleichen Fensters steht `Ignore Specific Library`.
4e: Dahinter in das freie Feld in der zweiten Spalte schreiben Sie `libci.lib`.
Stellen Sie im Kopf von `ddmesh1 Property Pages` die `Configuration:` auf `Release`.
Wiederholen Sie die Schritte 4a bis 4e und verlassen Sie dann die `ddmesh1 Property Pages` mit `OK`.

Schritt 5: Kopieren Sie folgende beiden Dateien ins Verzeichnis `C:\temp\ddmesh1`: **`Tiger.x`** und **`Tiger.bmp`**.
Schritt 6: Probeweise ausführen, beenden.

Code

Klicken Sie im Hauptmenu von VS.NET auf den Menupunkt `View` und dann auf den Unterpunkt `Class View` `Ctrl+Shift+C`.
Sie sehen im rechten Bereich des VS.NET-Hauptfensters das Unterfenster `Class View - ddmesh1`.
An dessen unteren Rand die Registerkarte `Class View` klicken.
Sie sehen im Unterfenster `Class View - ddmesh1` die Zeile `+ ddmesh1` , klicken Sie das Pluszeichen.
Sie sehen u.a. `+ CMainFrame`, klicken Sie das Pluszeichen.
Sie sehen u.a. `PreCreateWindow(CREATESTRUCT& cs)`, doppelklicken Sie diese Funktion.
Sie editieren jetzt das File `MainFrm.cpp`, Schreiben Sie dort in die Funktion `PreCreateWindow(CREATESTRUCT& cs)` anstelle der beiden Kommentarzeilen:

```
cs.cx = 700;
cs.cy = 700;
```

Doppelklicken Sie im Unterfenster `Class View - ddmesh1` die Zeile `CChildView`.
Sie editieren jetzt das File `ChildView.h`.
Includen Sie im Kopf von `ChildView.h`, am besten unterhalb der Zeile `#pragma once`:

```
#include < d3d8.h >
#include < d3dx8mesh.h >
```

Deklarieren Sie am Anfang (am besten gleich unter der ersten geschweiften Klammer) von `class CChildView : public CWnd` folgende Variablen:

```
private:
   LPDIRECT3D8           d3dpointer;
   LPDIRECT3DDEVICE8     d3dDevicepointer;
   D3DDISPLAYMODE        d3ddm;
   D3DPRESENT_PARAMETERS d3dpp;
   HRESULT               error;
   LPD3DXBUFFER          d3dID3DXBufferpointer;
   LPD3DXMESH            d3dMeshpointer;
   float                 angle_x, angle_y, angle_z;
```

Klicken Sie im Klassenbaum mit der rechten Maustaste auf `CChildView`.
Es öffnet sich ein Kontextmenü. Klicken Sie ganz unten im Kontextmenü auf `Properties`.
Es öffnet sich unter dem `Class View - ddmesh1` - Fenster ein `Properties` - Fenster mit der Überschrift `CChildView VCCodesClass`.
Im Toolbar dieses Fensters klicken Sie rechts neben dem gelben Blitz auf das `Messages` - Symbol, das aussieht wie ein weißer Topf mit blauem Deckel.
Sie sehen nun die Liste der Windows-Messages von `CChildView`.
Klicken Sie auf `WM_CREATE` und dann auf das schwarze Abwärtsdreieck rechts in dieser Zeile und dann auf `Add OnCreate`.
Klicken Sie auf `WM_LBUTTONDOWN` und dann auf das schwarze Abwärtsdreieck rechts in dieser Zeile und dann auf `Add OnLButtonDown`.
Klicken Sie auf `WM_TIMER` und dann auf das schwarze Abwärtsdreieck rechts in dieser Zeile und dann auf `Add OnTimer`.

Programmieren Sie die eben generierten Funktionen bis sie so aussehen:

```
int CChildView::OnCreate(LPCREATESTRUCT lpCreateStruct)
{  if (CWnd ::OnCreate(lpCreateStruct) == -1) return -1;
   d3dpointer = Direct3DCreate8( D3D_SDK_VERSION );
   d3dpointer->GetAdapterDisplayMode( D3DADAPTER_DEFAULT, &d3ddm );
   memset( &d3dpp, 0, sizeof(d3dpp) );
   d3dpp.Windowed = TRUE;
   d3dpp.SwapEffect = D3DSWAPEFFECT_DISCARD;
   d3dpp.BackBufferFormat = d3ddm.Format;
   d3dpp.EnableAutoDepthStencil = TRUE;
   d3dpp.AutoDepthStencilFormat = D3DFMT_D16;
   return 0;
}

void CChildView::OnLButtonDown(UINT nFlags, CPoint point)
{  HWND hWnd = CChildView::m_hWnd;
   d3dpointer->CreateDevice(
       D3DADAPTER_DEFAULT, D3DDEVTYPE_HAL, hWnd,
       D3DCREATE_SOFTWARE_VERTEXPROCESSING,
       &d3dpp, &d3dDevicepointer );
   unsigned long      NumMaterials;
   D3DXMATERIAL*      d3dxMaterials = NULL;
   D3DMATERIAL8       MeshMaterial;
   LPDIRECT3DTEXTURE8 MeshTexture = NULL;
   error = D3DXLoadMeshFromX( "tiger.x", D3DXMESH_SYSTEMMEM,
                              d3dDevicepointer, NULL,
                              &d3dID3DXBufferpointer, &NumMaterials,
                              &d3dMeshpointer );
   if (error!=D3D_OK) {MessageBox(NULL,"Load Tiger.x failed",MB_OK); return;}
   d3dxMaterials = (D3DXMATERIAL*)d3dID3DXBufferpointer->GetBufferPointer();
   MeshMaterial = d3dxMaterials->MatD3D;
   MeshMaterial.Ambient = MeshMaterial.Diffuse;
  error = D3DXCreateTextureFromFile( d3dDevicepointer, "tiger.bmp", &MeshTexture );
   if (error!=D3D_OK) {MessageBox(NULL,"Load Tiger.bmp failed",MB_OK); return;}
   d3dID3DXBufferpointer->Release();
   d3dDevicepointer->SetRenderState( D3DRS_AMBIENT, 0xFFFFFFFF );
   d3dDevicepointer->SetRenderState( D3DRS_ZENABLE, TRUE );
   SetTimer( 1, 1, NULL );
   angle_x = angle_y = angle_z = 0.f;
   d3dDevicepointer->SetMaterial( &MeshMaterial );
   d3dDevicepointer->SetTexture( 0, MeshTexture );
}

void CChildView::OnTimer(UINT nIDEvent)
{  D3DXMATRIX matWorld; D3DXMatrixIdentity( &matWorld );
   D3DXMATRIX matx ; D3DXMatrixIdentity( &matx  );
   D3DXMATRIX maty ; D3DXMatrixIdentity( &maty  );
   D3DXMATRIX matz ; D3DXMatrixIdentity( &matz  );
   D3DXMATRIX matView ; D3DXMatrixIdentity( &matView );
   D3DXMATRIX matProj; D3DXMatrixIdentity( &matProj );
   float pi = 3.14159f;
   if      ( angle_y < 2.f*pi ) D3DXMatrixRotationY( &maty, angle_y+=0.01 );
   else if ( angle_x < 2.f*pi ) D3DXMatrixRotationX( &matx, angle_x+=0.02 );
   else if ( angle_z < 2.f*pi ) D3DXMatrixRotationZ( &matz, angle_z+=0.06 );
   else
   {  D3DXMatrixRotationX( &matx, angle_x+=0.04 );
      D3DXMatrixRotationX( &matx, angle_x+=0.02 );
      D3DXMatrixRotationZ( &matz, angle_z+=0.01 );
   }
```

```
   D3DXMatrixMultiply ( &matWorld, &matx, &matWorld );
   D3DXMatrixMultiply ( &matWorld, &maty, &matWorld );
   D3DXMatrixMultiply ( &matWorld, &matz, &matWorld );
   if ( angle_x > 20.f*pi ) angle_x = angle_y = angle_z = 0.f;
   D3DXMatrixTranslation( &matView, .0f, .0f, -1.f );
   D3DXMatrixScaling( &matView,0.5f, 0.5f, 0.5f);
   D3DXMatrixLookAtLH( &matView, &D3DXVECTOR3( 0.0f, 3.0f,-5.0f ),
                       &D3DXVECTOR3( 0.0f, 0.0f, 0.0f ),
                       &D3DXVECTOR3( 0.0f, 1.0f, 0.0f ) );
   D3DXMatrixPerspectiveFovLH( &matProj, D3DX_PI/4, 1.0f, 1.0f, 100.0f );
   d3dDevicepointer->Clear( 0, NULL,
                            D3DCLEAR_TARGET|D3DCLEAR_ZBUFFER,
                            D3DCOLOR_XRGB(0,0,255), 1.0f, 0 );
   d3dDevicepointer->SetTransform(D3DTS_WORLD, &matWorld );
   d3dDevicepointer->SetTransform(D3DTS_VIEW , &matView  );
   d3dDevicepointer->SetTransform(D3DTS_PROJECTION, &matProj  );
   d3dDevicepointer->BeginScene();
   d3dMeshpointer->DrawSubset( 0 );
   d3dDevicepointer->EndScene();
   d3dDevicepointer->Present( NULL, NULL, NULL, NULL );
}
```

Programmieren Sie nun die Funktion `void CChildView::OnPaint()` bis sie so aussieht:

```
void CChildView::OnPaint()
{  CPaintDC dc(this);
   dc.TextOut(100,100,"Click left mouse button here!" );
}
```

Programmieren Sie nun den Destructor `CChildView::~CChildView()` bis er so aussieht:

```
CChildView::~CChildView()
{  if ( d3dDevicepointer != NULL ) d3dDevicepointer->Release();
   if ( d3dpointer != NULL ) d3dpointer->Release();
}
```

Ausführen

Experimente

(1) Variieren Sie die Startwinkel.
(2) Variieren Sie die Winkelzuwächse (Drehungsgeschwindigkeiten).
(3) Führen Sie die Anfangsdrehungen zweimal aus.
(4) Verlängern Sie die Drehungsphase bis zum zurücksetzen aller Winkel auf 0.
(5) Variieren Sie die Position des Betrachters in `D3DXMatrixLookAtLH`.
(6) Sehen Sie sich die Datei `Tiger.x` mit einem Texteditor an.
(7) Sehen Sie sich die Datei `Tiger.bmp` mit einem Bildeditor an.
(8) Lesen Sie den Link: **http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndrive/html/directx11192002.asp**
(9) Lesen Sie den Link: **http://msdn.microsoft.com/library/default.asp?url/code/list/directx.asp**
Branch into: `Graphics and Multimedia -> DirectX -> SDK Documentation -> DirectX 9.0 (C++) -> DirectX Graphics -> Programming Guide -> Tutorials, Samples, Tools, and Tips -> Tutorials -> Tutorial 6: Using Meshes`