Course 3D_MDX: 3D-Graphics with Managed DirectX 9.0
Chapter C7: Comments to the "Graphics Card" Project

---

namespaces

`using System;` //Home of the base class of all classes "`System.Object`" and of all primitive data types such as `Int32, Int16, double, string`.
`using System.Windows.Forms;` //Home of the "`Form`" class (base class of `Form1)` and its method `Application.Run`.
`using System.Drawing;` //Home of the "`Graphics`" class and its drawing methods such as `DrawStirng, DrawLine, DrawRectangle, FillClosedCurve` etc.
`using System.Text;` // Home of the `StringBuilder`-class.
`using Microsoft.DirectX.Direct3D;` //Graphics application programming interface (API) with models of 3-D objects and hardware acceleration.

Start and global variables: `public class Form1 : Form`

//We derive our window `Form1` from the class `Form`, which is contained in the `System.Windows.Forms` namespace.
The constructor `Form1()` contains the complete functionality of the program.

`static void Main() { Application.Run( new Form1() ); }` //Create an instance of `Form1` and ask the operating system to start it as main window of our program.

```
string s1 = "", s2 = "";
```

`StringBuilder s = new StringBuilder();` //The `StringBuilder`-object `s` will contain the complete output of the program since the method `StringBuilder.Append("something")` is much faster than any concatenation as: `String + "something"`.

```
TextBox TB = new TextBox(); //Space to display text.
TB.Size = ClientSize = new Size( 400, 800 );
TB.Font = new System.Drawing.Font( "Courier New", 8 );
TB.Multiline = true; //Allow line wrap.
TB.ScrollBars = ScrollBars.Both; //Add a vertical and a horizontal ScrollBar.
Controls.Add( TB ); //Connect the TextBox to Form1.
```

```
if ( Manager.Adapters.Count > 1 ) //How many graphics cards ?
s.Append( "You have two graphics cards or one dual monitor card.\r\n" +
          "The infos are about your primary card.\r\n\r\n" );
AdapterInformation ai = Manager.Adapters[0]; //Investigate the 1st graphics card. Ignore all others.
```

//For details about `AdapterInformation` see: **www.c-unit.com/tutorials/mdirectx/?t=51**

```
s.Append( "Graphic Board: " + ai.Information.Description + "\r\n" );
s.Append( "Driver : " + ai.Information.DriverName + "\r\n" );
s.Append( "Driver Vers. : " + ai.Information.DriverVersion + "\r\n\r\n" );
s.Append( "current display mode:\r\n" );
s.Append( ai.CurrentDisplayMode.Width .ToString() + "\tx " );
s.Append( ai.CurrentDisplayMode.Height.ToString() + "\t" );
s.Append( ai.CurrentDisplayMode.Format.ToString() + "\r\n\r\n" );
s.Append( s1 + "possible display modes:\r\n" );
```

```
foreach ( DisplayMode dm in ai.SupportedDisplayModes) //List of possible monitor resolutions
s2 = string.Format( "{0}\tx {1}\t{2}\r\n", dm.Width, dm.Height, dm.Format );
//Sample line: 1280 x 1024 X8R8G8B8.
if ( s2 != s1 ) { s.Append( s2 ); s1 = s2; } //Some DisplayModes just differ from RefreshRate, don't show them.
//Recommended experiment: Reduce this line to: s.Append( s2 ); and see what happens.
```

```
s.Append( "\r\n\r\nCapabilities of this graphics card:\r\n\r\n" ); //Title line for the next paragraph.
```

```
Caps caps = Manager.GetDeviceCaps( 0, DeviceType.Hardware ); //Details see: <b
s1 = caps.DeviceCaps.ToString(); //Make a long string of all DeviceCaps.
s1 = s1.Replace("Caps: ", "Caps:\r\n" ); //If the string contains a substring "Caps: ", insert a carriage return and a line feed character.
s1 = s1.Replace( "\n", "\r\n"); //Insert a carriage return character in front of each line feed character.
s.Append( s1 ); //Append to the global StringBuilder-object.
```

```
TB.Text = s.ToString(); //Convert the StringBuilder-object to String and show it.
```