

Re: Deployment from MDX Sample Framework

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.directx.managed/2005-02/00>

From: Kev (Kev_at_discussions.microsoft.com)

Date: 02/03/05

Date: Wed, 2 Feb 2005 18:21:02 -0800

Hi Phil – thanks for the quick response.

"are you sure the proper .NET framework version and MDX version is deployed already on the target machine"

Pretty sure, all Machines run .NET 1.1 with DirectX9.0c installed. DXDiag shows no problems.

In addition (and importantly I think) I have successfully deployed MDX applications to all test machines before, but only using my own D3D Framework, and then only if "SoftwareVertexProcessing" was selected on the present params by the user. Choosing HardwareVertexProcessing would cause a null reference exception without fail on all tested machines (1 * Win2K low spec card, 2 * XP, one with high Spec GeForce Card & one with TNT2).

"what does the debug runtime output stream tell you".

I'm sorry – I'm quite new at this. What is that? How would I see it? If you mean the error message, it told me that "CreateDevice" failed with an error number which translated to "Not available, Device does not support the queried technique".

"its strange that your app has a dependency on the samples".

You think? I'm not so sure, I imported the cs files from the "C:\DXSDK90C\Samples\Managed\Common" folder into my project and I depend the framework on it to get D3D up & running and make the following calls to get it running:

```
// Show the cursor and clip it when in fullscreen  
sampleFramework.SetCursorSettings(true, true);
```

```
// Init the sample framework and create the desired window and  
// direct3D device for the application. Calling each of these functions is  
// optional  
// but they allow you to set several options that control the behavior  
// of the sample framework
```

```
sampleFramework.Initialize(false, true, true);
sampleFramework.CreateWindow("Test App");
sampleFramework.CreateDevice(0, true, Framework.DefaultSizeWidth,
Framework.DefaultSizeHeight, TestEngine);
```

```
// Pass control to the sample framework for handling the message pump and
// dispatching render calls. The sample framework will call your FrameMove
// and FrameRender callback when there is idle time between handling
// windows messages
sampleFramework.MainLoop();
```

As for the dependancy, you may notice here that I'm using "Initialize(false, true, true);" the second "true" asks the framework to support the standard button presses, which means F2 brings up the "Device settings screen" where the user can change the D3D setup. The UI "controls" on that screen are rendered using a dds file, the dds file is located by a method in the framework called "FindMediaFile"..... the upshot is that the method will fail if you do not deploy the dds file in question with your app. The method will near enough search your entire hard drive for the needed file but if it ain't there, it ain't there! This is definate dependancy the SampleFramework has on the SDK installation (unless you do something about it).

Of course I find this out by digging around and so, naturally, am wondering if there are some other "hidden" set of things that an application that uses the sample framework needs to be deployed with in order to work. Other things to consider.

Or is all that just a total red herring and this is purley a matter that a device cannot be created for some reason – for example, Device Caps says that the host machine supports HardwareVertexProcessing & a pure device but when you call it, it fails?

"does it always work on a machine with the SDK installed but not on a machine without the SDK?"

I don't know the answer to that as I only have the SDK on my dev machine. I do have a download URL if anyone with the SDK is interested – its around 2MB though I'm afraid – I was ambitious enough to start texuring stuff, never imagined the deploy would fail.....

www.victory-road.co.uk/download/setupFrame1.msi

I'd be grateful for any assistance anyone can offer!

"Phil Taylor" wrote:

```
> are you sure the proper .NET framework version and MDX version is deployed
> already on the target machine, before the app deployment process occurs? if
> so, what does the debug runtime output stream tell you? the debug runtime
> output always has info for failed calls...
>
```

> also, its strange that your app has a dependency on the samples, as
> evidenced by the error...does it always work on a machine with the SDK
> installed but not on a machine without the SDK? so do you have another
> dependency on the SDK?
>
>
>
> "Kev" <Kev@discussions.microsoft.com> wrote in message
> news:AD66A13C-2B48-4084-B66C-480005148886@microsoft.com...
> > I hope someone can help me. I'm using the Managed DX sample framework and
> > have created a setup project of a simple Direct3D program.
> >
> > This installs & run fine on my dev machine but not on others, the
> > initialisation of Direct3D Fails. The error is always the same:
> >
> > ERROR at Microsoft.Samples.DirectX.UtilityToolkit.Framework.....
> >
> > I looked up the error number and found that it means, "Not available,
> > Device
> > does not support the queried technique".
> >
> > This is the same on Win2K, WinXP.
> >
> > I've noticed that the framework code depends on
> > "Media\UI\dxutcontrols.dds"
> > for rendering the UI for buttons etc so have included this (and the other
> > UI
> > files) in my msi but still no success, testing has showed that this would
> > only cause an error when the UI is displayed anyway – not at startup.
> >
> > Has anyone else had deployment problems? Share some tips? Does anyone know
> > what the problem likely is? Is there something about the sample framework
> > I
> > should be aware of when attempting deployment? I've run into problems with
> > deployment before from my own 3d framework – always specifying Hardware
> > Vertex Processing would cause a "Null Reference" exception, even on
> > machines
> > with GeForce FX cards ...
> >
> > I should also say that I'm supporting the IDeviceCreation Interface, here
> > is
> > some code from that.
> >
> > public bool IsDeviceAcceptable(Caps caps, Format adapterFormat, Format
> > backBufferFormat, bool windowed)
> > {
> > // Skip back buffers that do not support Alpha blending
> > if (!Manager.CheckDeviceFormat(caps.AdapterOrdinal, caps.DeviceType,
> > adapterFormat,
> > Usage.QueryPostPixelShaderBlending, ResourceType.Textures,
> > backBufferFormat))

```
> > return false;
> >
> > // Skip devices that do not support at least one light!
> > if (caps.MaxActiveLights == 0)
> > return false;
> >
> > return true;
> > }
> >
> > public void ModifyDeviceSettings(DeviceSettings settings, Caps caps)
> > {
> > // This application is designed to work on a pure device by not using
> > // any Get methods, so create a pure device if supported and using HWVP
> > if ( (caps.DeviceCaps.SupportsPureDevice) &&
> > ((settings.BehaviorFlags & CreateFlags.HardwareVertexProcessing) != 0) )
> > settings.BehaviorFlags |= CreateFlags.PureDevice;
> > }
> >
> >
> > I'm totally at a loss with trying to deploy managed DirectX applications –
> > please someone enlighten me to the secrets!
> >
> >
>
>
>
```