

An Interview with Tom Speirs

Extending the Desktop to Games – GameEx

Most of the articles in MSDN magazine tend to be focused on the more serious topics such as web services, security and performance tuning and testing. As something a little bit different to lighten your reading load this issue, we sat down with Tom Speirs, creator of GameEx, and discussed with him the joys of designing something that is intended from the outset to be entertaining.

Rather than attempting to describe GameEx ourselves, we asked Speirs to give us a description. Speirs said “GameEx is a front-end application for command line game emulators. Initially I wrote it for MAME.”

MAME stands for Multiple Arcade Machine Emulator, and is the most popular game emulator available. It currently supports more than 3000 unique games, and the numerous authors pride themselves on the most accurate emulation possible for each one. For those of you who are not sure what an arcade machine game is – think the classics – Galaga, Pacman, Space Invaders through to more modern titles such as Metal Slug and Raiden Fighters.

Speirs continued on, “it basically acts as a GUI that allows you to browse, filter, organise and launch games for classic games systems. It also supports the use of remote controls, gamepads

and even Arcade controllers.”

There are many other front-end applications for MAME and other game emulators available for download all over the web, but he considers GameEx as a special case that stands out from the crowd. Speirs says “GameEx is a little special, as it is designed to work as a seamless plug-in for Windows Media Center Edition. It can also be used in custom arcade cabinets as it allows full operation without the need for a mouse or keyboard.”

“It doesn’t need a normal Windows GUI either – much like Windows MCE itself. In addition, it has several HTPC (Home Theatre PC) type features that allow playing of videos and a fully functional music jukebox. All of these features can also be operated with the remote control or gamepad.”

GameEx not only runs on Windows Media Center Edition, but it borrows from the look and feel of that edition of Windows. Speirs claimed that he emulated MCE “because it’s just about the best GUI I have ever seen. And I also originally designed it to be a plug-in for MCE so it made sense to use the same kind of interface.”

When you run GameEx for the first time, you’ll find a solid application that takes over the entire system. It performs extremely well too, so you might think that it uses low-level unmanaged code to achieve its results. We asked Tom what technologies were



behind the application, which he revealed as being managed code, with the majority of GameEx written in Visual Basic .NET and accessing the DirectX SDK.

“I discovered from looking at the DirectX SDK that Visual Basic .NET actually ran the samples at a much better speed than Visual Basic 6. I thought that would be enough power for GameEx, so I chose to code in VB.NET using the Visual Studio .NET IDE,” Speirs said.

He continued on, “As a seasoned programmer/analyst and being business oriented in my ‘other’ life, I am focused on results and getting things done to a deadline, rather than using languages that give the best speed or performance. Having the application written in Visual Basic .NET meant that the project was quick to develop, and is still quick to enhance.”

It seems that his combination of VB.NET and DirectX has also meant that the supposed performance hit that many people claim comes in using managed code isn’t obvious – if it’s there at all.

Speirs said it was a new challenge for him to create GameEx, as it used parts of DirectX that he hadn’t looked at before.

“I’d already used DirectDraw and DirectSound in C programs, but never looked at Direct3D. DirectDraw is quite easy to use and I just didn’t want to face the learning curve of Direct3D.” This statement confirms his commitment to choosing the best and easiest tools for the job to get it done rather than wasting too much time on looking around. “I thought that DirectDraw would be enough for what I wanted to do with the only extra things that would have been nice to have being things like hardware alpha blending and rotation. I didn’t feel that was really needed for the project.”

We thought that coding for Windows Media Center Edition may have presented some interesting issues during development, but when we asked him if he thought developing for MCE was harder than for other editions of Windows, Speirs said “No, not at all. GameEx does not require MCE and runs on all modern versions of Windows. It is a completely separate application to Windows Media Center Edition but just happens to have a

similar look and feel, and supports similar features such as the remote control.”

This gives people who are running other versions of Windows the opportunity to easily customise their main user interface to look like the simple and elegant Windows Media Center Edition, but he admitted to not supporting one aspect of MCE that could have made his development easier but would have disallowed it on other editions of Windows; “It supports applications running in the same process, but that’s limited to web applications only. I investigated coding GameEx like that but decided I wanted my application to be visually stunning and with web applications, this wasn’t possible.”

Tom is obviously proud of his program, and so we took the chance to ask him some more pointed questions about the technical aspects of GameEx. He was more than willing to walk through sections of his code and discuss components that he was really happy with.

“I really like the way GameEx controls MCE, MAME and other emulators by checking if their process is running, and closing them accordingly. In Visual Basic 6 we had the SendKeys and AppActivate commands to control applications. While these still exist in Visual Basic .NET, we now have the ability to monitor and control applications processes with the Diagnostics namespace.”

Speirs provided some useful code for anyone wanting to follow in his footsteps of gaining control over Windows Media Center Edition. He revealed that when GameEx first starts up he detects if MCE is open with the following code snippet (code listing 1):

```
If UBound(Diagnostics.Process.GetProcessesByName("ehshell")) >= 0 Then
    Dim proc As Process
    For Each proc In Diagnostics.Process.GetProcessesByName("ehshell")
        proc.CloseMainWindow()
        MediaCenterWasOpen= True
    Next
End If
```

Code Listing 1



Corresponding code is included when GameEx is shutdown to restart MCE (code listing 2):

```
Dim ehshell As Integer

If MediaCenterWasOpen= True Then
    ehshell = Shell(""" & GetWinPath() & "\ehome\ehshell.exe""", _
        AppWinStyle.MaximizedFocus, False)
End If
```

Code Listing 2

Another neat feature of GameEx is that it has its own screen-saver. After a few minutes of inactivity, GameEx runs through the games list of MAME, displaying a minute or two of gameplay for each one until the user returns to the computer. Speirs revealed that he achieved that through the Process.CloseMainWindow method along with CurrentThread.Sleep(). “In addition, I check Process.MainWindowTitle to see if MAME and any other applications I am interested in are running by their window caption.”

“I also found the threading features of Visual Basic .NET incredibly easy to use, and stable as well,” he continued on, “I create a thread to download album art for the selected track in the Jukebox,” which allows the main process of GameEx to continue without a performance hit.

One feature did cause Speirs some headaches, but the final result is impressive; “Something I wanted to do was support MAME videos, so you could see the game play of a MAME game while it was selected. I searched everywhere in the documentation and online for information on how to do this, but couldn’t find the solution.”

“The problem is that Managed DirectX does support the playing of videos, but it only allows you to output the video to a Windows control, form or a Direct3D texture. As I was using DirectDraw, what I really needed was the ability to Blit the video output to a DirectDraw surface.”

The solution was to create a simple C++ DLL that had access to the full Windows API for media playback. It has simple methods to open and then play a video file, using the BitBlt API call to Blit the current video frame on to a DirectDraw surface HDC. “Once I saw how this was done, I added the option to alpha blend the output – something I wasn’t going to do originally – and finetuned the DLL to play the video in a thread.”

“Getting the HDC of a DirectDraw Surface object is easy,” Speirs says. If anyone needs to do this, all they need is the following few

lines of Visual Basic .NET code:

```
Dim hdc As IntPtr
hdc = surfacevideo.GetDc()
Functions.DrawNewFrame(hdc, surfacevideo.SurfaceDescription.Width, sur-
facevideo.SurfaceDescription.Height,surfacesnap2.Height,Alpha)
ReleaseDc(hdc)
```

Code Listing 3

Once he decided to use C++ to do the Blitting, even that became straightforward. The final C++ code for the DrawNewFrame function looks like this:

```
m_pDDSurface7->GetDC(&hdc2);
if (alpha!=0)
{
    blend.BlendOp=AC_SRC_OVER;
    blend.BlendFlags=0;
    blend.SourceConstantAlpha=alpha;
    AlphaBlend(backHDC, 0, 0, width, height,hdc2, 0, 0, width, height,
    blend);
}
else
{
    BitBlt(backHDC, 0, 0, width, height,hdc2,m_rSrcRect.left, m_rSrcRect.
top, SRCCOPY);
}
m_pDDSurface7->ReleaseDC(hdc2);
```

Code Listing 4

“All I need to do is pass in an Alpha value to have the code automatically alpha blend the video out to the surface – otherwise the video playback uses the standard BitBlt command.”

We finished the interview by asking him what he thinks of the outcome. Speirs said that “although the project is a hobby and for entertainment use, I enhanced my skillset a great deal. The project is also quite popular amongst both MAME and Windows MCE communities and I get a lot of positive feedback. I am still actively developing and enhancing the project – it’s helped me become very proficient with Visual Basic .NET while boosting my confidence with C++.” Sounds like a win all round for Speirs.

GameEx is one of the most updated front-end applications currently on the web, with new builds coming out at least weekly and usually more frequently. With each new release comes new features and in between the time this interview was conducted and it going to print, a number of cool additions have been added to the application.

For more information on GameEx, including the latest build, go to <http://gameex.net>.