The EAX Unified Interface

The EAX Unified Interface ensures that the Environmental Audio features of an EAX application can be supported on any PC platform, whether or not it includes a Creative sound card with the latest drivers installed. The Unified Interface looks for a previous version of EAX or compatible standard, and automatically translates EAX calls to the most capable interface found, preserving as much as possible of the Environmental Audio features implemented in the application (including environment reverberation and reflections, as well as occlusion and obstruction effects). It is not necessary to utilize the EAX Unified interface to support EAX in your application, however the advantage of doing so, is simply to help provide a larger installed base of EAX capable systems.

The current version of the Unified interface provides automatic translation of EAX 2.0 parameters into EAX 1.0 parameters.

The EAX Unified Interface is an exact duplicate of the Direct Sound interface. Any application that currently uses the Direct Sound 7 or Direct Sound 8 interfaces can easily be altered to utilize the EAX Unified Interface by simply changing how it initializes and retrieves the Direct Sound interface. No other changes to the code base of the application are needed. While the Direct Sound 7 (or 8) interfaces are retrieved from the DSOUND.DLL, the EAX Unified Interface is retrieved from the EAX.DLL.

The EAX Unified Interface can be used to obtain either a Direct Sound 7 interface or a Direct Sound 8 interface. There are two alternative methods for obtaining one of these interfaces.

Initializing EAX.DLL using *load-time dynamic linking*

The EAX.DLL exports two functions - EAXDirectSoundCreate and EAXDirectSoundCreate8 in much the same way as the DSOUND.DLL exports DirectSoundCreate and, if using Direct X version 8, the DirectSoundCreate8 function.

The EAXDirectSoundCreate function takes the same parameters as DirectSoundCreate and returns the same type of interface (Direct Sound 7). In addition, the EAXDirectSoundCreate8 function takes the same parameters as DirectSoundCreate8 and returns the same type of interface (Direct Sound 8).

To utilize either of these functions you will need to link your application with the EAX.LIB file included with the EAX SDK. Once you have done this, it is essential that the EAX.DLL be present in either the \windows\system directory, or the directory in which your executable exists. If the EAX.DLL is not present, the operating system will throw up a message box informing the user of the missing component and cease to run the application. When using this method of creating the EAX Unified Interface you will need to be sure to either,

1. Install the EAX.DLL into the same directory as your application using your installer

Or

2. Install the EAX.DLL using the ‘redist’ installer supplied with the EAX SDK.

For a Direct Sound 7 application, the initialization code would look something like this: -

```
LPDIRECTSOUND        lpDS = NULL;

// Initialize Direct Sound using the EAXDirectSoundCreate method
if(FAILED(hr = EAXDirectSoundCreate(NULL, &lpDS, NULL)))
{
                printf("Failed the EAXDirectSoundCreate function");
        return false;
}
```

For a Direct Sound 8 application, the code would like something like this: -

```
LPDIRECTSOUND8       lpDS8 = NULL;

// Initialize Direct Sound using the EAXDirectSoundCreate8 method
if (FAILED(hr = EAXDirectSoundCreate8(NULL, &lpDS8, NULL)))
{
        printf("Failed the EAXDirectSoundCreate8 function");
        return false;
}
```

Initializing EAX.DLL using *CoCreateInstance*

The alternative method for using the EAX.DLL is to use the CoCreateInstance function. CoCreateInstance will create a single uninitialized object of the class associated with a specified CLSID. The EAX.DLL provides 2 CLSID GUIDs each defined in the EAX.H file. The CLSID_EAXDirectSound identifier can return a Direct Sound 7 interface (IID_IDirectSound), whereas the CLSID_EAXDirectSound8 identifier can return either a Direct Sound 7 interface or a Direct Sound 8 interface (IID_IDirectSound8).

To utilize this method, the EAX.DLL will need to be installed via the redist installer supplied with the EAX SDK. This installer will take care of checking version numbers (to avoid overwriting a newer version of the EAX.DLL) and of registering the DLL for COM runtime linking. Note: this installer can be launched from your own installer, and has an optional silent mode of operation.

When retrieving a Direct Sound interface using this method, the Direct Sound “Initialize” function must be called before the co-operative level can be set (exactly the same as if you used the CoCreateInstance method to load the DSOUND.DLL).

The following code fragment will initialize Direct Sound 7 via the EAX Unified Interface retrieved using the CoCreateInstance function.

```
LPDIRECTSOUND        lpDS = NULL;

CoInitialize(NULL);

if (FAILED(hr = CoCreateInstance(CLSID_EAXDirectSound, NULL,
        CLSCTX_INPROC_SERVER, IID_IDirectSound, (void**)&lpDS)))
{
        printf("Failed to CoCreateInstance for EAXDirectSound\n");
        return false;
}

if (FAILED(hr = lpDS->Initialize(NULL)))
{
        printf("Failed to initialize Direct Sound\n");
        return false;
}

printf("Successfully initialized Direct Sound\n");

…
```

Alternatively, the Direct X 8 version of the code would look like this: -

```
LPDIRECTSOUND8       lpDS8 = NULL;

CoInitialize(NULL);

if (FAILED(hr = CoCreateInstance(CLSID_EAXDirectSound8, NULL,
        CLSCTX_INPROC_SERVER, IID_IDirectSound8, (void**)&lpDS8)))
{
        printf("Failed to CoCreateInstance for CLSID_EAXDirectSound8\n");
        return false;
}

printf("CoCreateInstance for EAXDirectSound8 succeeded\n");

if (FAILED(hr = lpDS8->Initialize(NULL)))
{
        printf("Failed to Initialize Direct Sound 8\n");
        return false;
}

…
```

Since this method of initialization does not require linking with the EAX.LIB file, the above code fragment can be modified to fall back to using the DSOUND.DLL directly in the event that the EAX.DLL could not be initialized. This could happen if the EAX.DLL were removed from the system, or if it was not installed correctly. In either case, the Direct Sound interface could be retrieved through the DSOUND.DLL without need for change in any of the application code as both the Direct Sound interface and the EAX Unified Interface are identical to the application.

The following modification of the above DX 7 code fragment will handle these situations.

```
LPDIRECTSOUND        lpDS = NULL;

CoInitialize(NULL);

if (FAILED(hr = CoCreateInstance(CLSID_EAXDirectSound, NULL,
        CLSCTX_INPROC_SERVER, IID_IDirectSound, (void**)&lpDS)))
{
        // Failed initialization of the EAX.DLL, try DirectSound directly
        if (FAILED(hr = CoCreateInstance(CLSID_DirectSound, NULL,
                CLSCTX_INPROC_SERVER, IID_IDirectSound, (void**)&lpDS)))
        {
                printf("Failed to CoCreateInstance of EAXDirectSound and DirectSound\n");
                return false;
        }
}

if (FAILED(hr = lpDS->Initialize(NULL)))
{
        printf("Failed to initialize Direct Sound\n");
        return false;
}

printf("Successfully initialized Direct Sound\n");
```