How do I forward an exported function to an ordinal in another DLL?

devblogs.microsoft.com/oldnewthing/20121116-00

November 16, 2012

Raymond Chen

The syntax for specifying that requests to import a function from your DLL should be forwarded to another DLL is

```
; A.DEF
EXPORTS
 Dial = B.Call
```

This says that if somebody tries to call `Dial()` from `A.DLL`, they are really calling `Call()` in `B.DLL`. This forwarding is done in the loader. Normally, when a client links to the function `A!Dial`, the loader says, "Okay, let me get the address of the `Dial` function in `A.DLL` and store it into the `__imp__Dial` variable." It's the logical equivalent of

```
client::__imp__Dial = GetProcAddress(hinstA, "Dial");
```

When you use a forwarder, the loader sees the forwarder entry and says, "Whoa, I'm not actually supposed to get the function from `A.DLL` at all! I'm supposed to get the function `Call` from `B.DLL`!" So it loads `B.DLL` and gets the function `Call` from it.

```
hinstB = LoadLibrary("B.DLL");
client::__imp__Dial = GetProcAddress(B, "Call");
```

(Of course, the loader doesn't actually do it this way, but this is a good way of thinking about it.)

But what if the function `Call` was exported by ordinal? How do you tell the linker, "Please create a forwarder entry for `Dial` that forwards to function 42 in `B.DLL`?"

I didn't know, but I was able to guess.

Back in the days of 16-bit Windows, there were two ways to obtain the address of a function exported by ordinal. The first way is the way most people are familiar with:

```
FARPROC fp = GetProcAddress(hinst, MAKEINTRESOURCE(42));
```

The second way uses an alternate formulation, passing the desired ordinal as a string prefixed with the number-sign:

```
FARPROC fp = GetProcAddress(hinst, "#42");
```

You can hide a number inside a string by using `MAKEINTRESOURCE`, and you can hide a string inside a number by using the '#' character.

Given that the number sign has been used in the past to hide a number inside a string, I figured it was worth a shot to see if the loader carried this convention forward. (No pun intended.)

```
; A.DEF
EXPORTS
 Dial = B.#1
```

Hey, check it out. It works.

Sometimes a little knowledge of history actually helps you solve problems in the present day.

Raymond Chen

Follow