What is the HINSTANCE passed to CreateWindow and RegisterClass used for?

devblogs.microsoft.com/oldnewthing/20050418-59

April 18, 2005

Raymond Chen

One of the less-understood parameters to the `CreateWindow` function and the `RegisterClass` function is the `HINSTANCE` (either passed as a parameter or as part of the `WNDCLASS` structure).

The window class name is not sufficient to identify the class uniquely. Each process has its own window class list, and each entry in the window class list consists of an instance handle and a class name. For example, here's what the window class list might look like if a program has two DLLs, both of which register a class name "MyClass", passing the DLL's handle as the `HINSTANCE`.

	HINSTANCE	Class name
1.	USER32.DLL	Static
2.	USER32.DLL	Button
3.	USER32.DLL	Listbox
4.	USER32.DLL	Combobox
5.	USER32.DLL	Edit
6.	A.DLL	MyClass
7.	B.DLL	MyClass

When it comes time to create a window, each module then passes its own `HINSTANCE` when creating the window, and the window manager uses the combination of the instance handle and the class name to look up the class.

```
CreateWindow("MyClass", ..., hinstA, ...); // creates class 6
CreateWindow("MyClass", ..., hinstB, ...); // creates class 7
CreateWindow("MyClass", ..., hinstC, ...); // fails
```

This is why it is okay if multiple DLLs all register a class called "MyClass"; the instance handle is used to tell them apart.

There is an exception to the above rule, however. If you pass the `CS_GLOBALCLASS` flag when registering the class, then the window manager will ignore the instance handle when looking for your class. All of the USER32 classes are registered as global. Consequently, all of the following calls create the USER32 edit control:

```
CreateWindow("edit", ..., hinstA, ...);
CreateWindow("edit", ..., hinstB, ...);
CreateWindow("edit", ..., hinstC, ...);
```

If you are registering a class for other modules to use in dialog boxes, you need to register as `CS_GLOBALCLASS`, because as we saw earlier the internal `CreateWindow` call performed during dialog box creation to create the controls passes the dialog's `HINSTANCE` as the `HINSTANCE` parameter. Since the dialog instance handle is typically the DLL that is creating the dialog (since that same `HINSTANCE` is used to look up the template), failing to register with the `CS_GLOBALCLASS` flag means that the window class lookup will not find the class since it's registered under the instance handle of the DLL that provided the class, not the one that is using it.

In 16-bit Windows, the instance handle did other things, too, but they are no longer relevant to Win32.

A common mistake is to pass the `HINSTANCE` of some other module (typically, the primary executable) when registering a window class. Now that you understand what the `HINSTANCE` is used for, you should be able to explain the consequences of registering a class with the wrong `HINSTANCE`.

Raymond Chen

Follow