The history of passing a null pointer as the key name to RegOpenKeyEx

devblogs.microsoft.com/oldnewthing/20210723-00

July 23, 2021

Raymond Chen

For decades, the documentation for the `RegOpenKeyEx` function said

> The *lpSubKey* parameter can be **NULL** only if *hKey* is one of the predefined keys.

This statement was true when it was written.

In 1992.

For Windows NT 3.5, the behavior of the `RegOpenKeyEx` function was revised so that passing **NULL** as the *lpSubKey* is equivalent to passing an empty string.

Nobody updated the documentation to reflect this.

As a result, from 1994 to 2021, the documentation for `RegOpenKeyEx` called out a special case that was no longer a special case.

Here's what changed:

RegOpenKeyEx(key, subkey, …)			
key	**subkey**	**Windows NT 3.1**	**Windows NT 3.5 and later**
Predefined key	`NULL`	Refreshes hive, returns same key	
Other key	`NULL`	`ERROR_BADKEY`	Returns unique handle to same key
Any key	non-`NULL`	Returns unique handle to subkey	

The `ERROR_BADKEY` case was removed. Instead, what you get is a unique handle to the same underlying key.

Note that the `RegOpenKey` function behaves differently when you pass `NULL` or an empty string as the subkey name:

RegOpenKey(key, subkey, …)		
key	**subkey**	**Result**
Predefined key	`NULL` or empty string	Refreshes hive, returns same key
Other key	`NULL` or empty string	Returns same key
Any key	non-empty string	Returns unique handle to subkey

The fact that it returns the same key back when the subkey is `NULL` or empty makes the function difficult to use because the handle might need to be closed, or might not, depending on whether the subkey is a non-empty string.

Some of the outcomes are labeled *Refreshes hive*. What does that mean?

If you pass a predefined key to `RegCloseKey` or trigger one of the outcomes marked *Refreshes hive*, then the registry key associated with the root of the predefined key is closed, and when all of the outstanding subkeys from that hive are closed, the hive is unloaded. Meanwhile, any future references to the predefined key will go back and reload the hive.

Ironically, one of the things that counts as a reference to a predefined key is closing it! This means that if you call `RegCloseKey` twice on a predefined key, the first time will close the hive reference from the root key, and then the second time will reload the hive, only to close it imediately.

Basically, you shouldn't try to close a predefined key. It just creates a lot of work for no net effect.

One rare case where there is a net effect is where you are closing `HKEY_CURRENT_USER` or `HKEY_CLASSES_ROOT` from a service that impersonates. Recall that these keys are problematic when impersonating because they load the registry hive associated with the user being impersonated, making the hive available to all threads (not just the one doing the impersonation), and it remains available even after the impersonation reverts.

A totally hacky way to clear out the hive left over from impersonation is to close it explicitly, but now you're using a global solution for a local problem. The predefined keys are applicable to the entire process, but you are trying to clean up your thread. If two threads are impersonating, they will step all over each other. You really should be using `RegOpen-CurrentUser` or using `RegOpenUserClassesRoot` to access the registry hive that corresponds to the user being impersonated.

Raymond Chen

Follow