# Why do registry keys have a default value?

devblogs.microsoft.com/oldnewthing/20080118-00

January 18, 2008

Raymond Chen

In addition to all the named values you can create underneath a registry key with the `RegSetValueEx` function, there is also the so-called *default value* which you obtain by passing `NULL` or a pointer to a null string as the `lpValue`. This default value is also the value set and retrieved when you call `RegSetValue` and `RegQueryValue`. What's the deal with this default value? The original 16-bit registry didn't have named values. All it had were keys, and associated with each key was a single piece of data: a string. The functions that operated on this data were `RegSetValue` and `RegQueryValue`, which explains why those functions (1) don't have a `lpValue` parameter and (2) set and retrieve only string data. Because back in the 16-bit world, that's all you had. In the conversion to Win32, the registry gained new capabilities, such as storing data in formats beyond simple strings, and storing multiple pieces of data under a single key, using a name to distinguish them. What used to be called simply "the value of a registry key" (for since there was only one, there was no need to give it a name) now goes by the special name *the default value*: It's the value whose name is null.

There's nothing particularly special about the default value aside from its unusual name. A named value need not exist, and if it exists, the data type could be anything. Similarly, the default value need not exist, and its type can be anything. At this point, it's just a value with a strange name.

Raymond Chen

**Follow**