# Why does INVALID_HANDLE_VALUE cast through a LONG_PTR first?

devblogs.microsoft.com/oldnewthing/20250326-00

Lyubomir Lyubomirov March 26, 2025


Raymond Chen

The definition of INVALID_HANDLE_VALUE is

```
#define INVALID_HANDLE_VALUE ((HANDLE)(LONG_PTR)-1)
```

Why is there an extra cast of `LONG_PTR`? Why not just go straight to the final answer?

```
#define INVALID_HANDLE_VALUE ((HANDLE)-1)
```

One clue is that 32-bit Windows did use that simpler definition.

That clue tells us that the answer has something to do with 64-bit programming.

The issue is that casting a 32-bit integer to a 64-bit pointer involves two operations: Upgrading the 32-bit value to a 64-bit value, and then treating that 64-bit value as a pointer. The C and C++ languages leave the “interpreting an integer as a pointer” as an implementation-defined operation, but at least if the integer is a 64-bit integer and the pointer is a 64-bit pointer, there is only one reasonable implementation, and that is just to take the integer bit pattern and use it as a pointer.

The problem is that if the integer is a 32-bit integer and the pointer is a 64-bit pointer, then multiple implementations might make sense. One might be to zero-extend the 32-bit value to a 64-bit value, which might be preferred on systems that do zero extension naturally. Another would be to sign-extend the 32-bit value to a 64-bit value, which might be preferred on systems that do sign extension naturally. Or it might be to extend based on the signed-ness of the 32-bit value, which might be preferred on systems that can do both types of extensions with equal facility.

Since there are multiple reasonable but conflicting choices that could be made by an implementation, we should avoid casting a 32-bit integer to a 64-bit pointer because we’re not sure what we’re going to get.

What we want is a pointer-sized value that has all bits set. So we start with the 32-bit signed integer `-1` and cast it to a `LONG_PTR`, which makes it a pointer-sized signed integer. Then we cast that pointer-sized integer to a pointer, which in any reasonable implementation would be a reinterpretation with no change in value.

In other words, the extra cast forces the compiler down the desired path.

| | | |
|---|---|---|
| | `FFFFFFFF` | `-1` |
| | ↓ | sign extension required by language rules |
| `FFFFFFFF` | `FFFFFFFF` | `(LONG_PTR)-1` |
| ↓ | | only reasonable implementation |
| `FFFFFFFF` | `FFFFFFFF` | `(HANDLE)(LONG_PTR)-1` |

If we had omitted the cast to `LONG_PTR`, then we would lose control over the upper 32 bits.

| | | |
|---|---|---|
| | `FFFFFFFF` | `-1` |
| | ↓ | no control over how 32-bit value becomes a 64-bit value |
| `????????` | `FFFFFFFF` | `(HANDLE)-1` |

As a general rule, therefore, casts between pointers and 32-bit integers are mediated by 64-bit integers so that we can control whether the values are sign-extended or zero-extended. We saw a little while ago that different processors have different preferences on how 32-bit values are extended to 64-bit values, so we can't leave this up to chance.

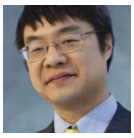
Once this rule is in place, we can activate compiler warnings that trigger when you perform casts between 32-bit integers and 64-bit pointers. These are usually indications of old 32-bit code that is not 64-bit ready. In the case that the integer is a variable or a function return value, the fix is to upgrade the variable or function to return a 64-bit integer so that the upper 32 bits of pointers are not lost. In the case that the integer is just being smuggled inside a pointer, insert an explicit cast to a 64-bit value to indicate that you don't actually have a pointer at all; you just want to create a pointer with a specific bit pattern.

Category

Old New Thing

## Author

Raymond Chen



Raymond has been involved in the evolution of Windows for more than 30 years. In 2003, he began a Web site known as The Old New Thing which has grown in popularity far beyond his wildest imagination, a development which still gives him the heebie-jeebies. The Web site spawned a book, coincidentally also titled The Old New Thing (Addison Wesley 2007). He occasionally appears on the Windows Dev Docs Twitter account to tell stories which convey no useful information.

## 1 comment

Join the discussion.

Sign in

Newest

L.L.

It’s so simple that I’m surprised you have to explain it.

## Stay informed

Get notified when new posts are published.