Appearing to succeed is a valid form of undefined behavior, but it's still undefined

devblogs.microsoft.com/oldnewthing/20110929-00

September 29, 2011

Raymond Chen

A customer requested a clarification on the MSDN documentation for the `HeapFree` function.

> The MSDN documentation says that if the `lpMem` parameter is `NULL`, then the behavior is undefined. Is this true?

As explicitly stated in MSDN, the behavior is undefined. Observe that the annotation on the `lpMem` parameter is `__in`, which means that the parameter must be a non-`NULL` value provided by the caller. (If `NULL` were permitted, the annotation would have been `__in_opt`.)

Undefined behavior means that *anything can happen*. The program might crash immediately. It might crash five minutes later. It might send email to your boss saying that you screwed up and then read you Vogon poetry. Or maybe not.

MSDN says don't do it, so don't do it.

The customer explained why they were interested in knowing more information about undefined behavior:

> We were interested because there is a mismatch between the semantics of a function we are implementing (where `NULL` is valid and ignored) and the function `HeapFree` we are using as the implementation. It looks like Windows Vista returns `TRUE` if you pass `NULL`.

If there is a mismatch in semantics between the function you are implementing and the function you are calling, it is your responsibility as the programmer to bridge the gap. The customer didn't say what function they were implementing, but I'm guessing it was something like `operator delete`. Since your function accepts `NULL` but `HeapFree` doesn't, it is your responsibility to filter out `NULL` parameters.

```
void operator delete(void* ptr) throw ()
{
 if (ptr != NULL)
  HeapFree(CustomHeap, 0, ptr);
}
```

This concept goes by the fancy name of the Adapter Pattern. The less fancy name is wrapper function.

And the value returned by `HeapFree` on Windows Vista is irrelevant. Pretending to succeed is a valid form of undefined behavior, because *anything* qualifies as undefined behavior.

(Of course, you can't assume that returning `TRUE` will always be the result of triggering undefined behavior. After all, if you could rely on it, then it wouldn't be undefined any more!)

Raymond Chen

Follow