Trying to avoid double-destruction and inadvertently triggering it

devblogs.microsoft.com/oldnewthing/20091111-00

November 11, 2009

Raymond Chen

We saw some time ago the importance of artificially bumping an object's reference count during destruction to avoid double-destruction. However, one person's attempt to avoid this problem ended up triggering it.

```
ULONG MyObject::Release()
{
 LONG cRef = InterlockedDecrement(&m_cRef);
 if (cRef > 0) return cRef;
 m_cRef = MAXLONG; // avoid double-destruction
 delete this;
 return 0;
}
```

The explanation for the line `m_cRef = MAXLONG` was that it was done to avoid the double-destruction problem if the object receives a temporary `AddRef/Release` during destruction.

While it's true that you should set the reference count to an artificial non-zero value, choosing `MAXLONG` has its own problem: integer overflow.

Suppose that during the object's destruction, the reference count is temporarily incremented twice and decremented twice.

Action	m_cRef
Just before call to Release()	1
InterlockedDecrement	0
m_cRef = MAXLONG	2147483647
destructor does temporary `AddRef()`	−2147483648 (integer overflow)
destructor does temporary `AddRef()`	−2147483647

destructor does temporary `Release()`	−2147483648
since `m_cRef` < 0, we re-destruct	

Sure, choosing a huge `DESTRUCTOR_REFCOUNT` means that you have absolutely no chance of decrementing the reference count back to zero prematurely. However, if you choose a value too high, you introduce the risk of *incrementing* the reference count so high that it overflows.

That's why the most typical values for `DESTRUCTOR_REFCOUNT` are 1, 42, and 1000. The value 1 is really all you need to avoid double-destruction. Some people choose 42 because it's cute, and other people choose 1000 because it's higher than any "normal" refcount, so it makes it easier to spot during debugging. But even then, the "high" value of 1000 still leaves room for over two billion `AddRef()`s before overflowing the reference count.

On the other hand, if you choose a value like `MAXLONG` or `MAXDWORD`, then you're taking something that previously never happened (reference count integer overflow) and turning it into an almost certainty.

Raymond Chen

Follow