Do you know when your destructors run? Part 2.

devblogs.microsoft.com/oldnewthing/20040521-00

May 21, 2004

Raymond Chen

Continuing from yesterday, here's another case where you have to watch your destructors. Yesterday's theme was destructors that run at the wrong time. Today, we're going to see destructors that don't run at all!

Assume there's an ObjectLock class which takes a lock in its constructor and releases it in its destructor.

```
DWORD ThreadProc(LPVOID p)
{
  ... do stuff ...
  ObjectLock lock(p);
  ... do stuff ...
  return 0;
}
```

Pretty standard stuff. The first batch of stuff is done without the lock, and the second batch is done inside the lock. When the function returns, the lock is automatically released.

But suppose somebody adds a little code to this function like this:

```
DWORD ThreadProc(LPVOID p)
{
  ... do stuff ...
  ObjectLock lock(p);
  ...
  if (p->cancelled) ExitThread(1);
  ...
  return 0;
}
```

The code change was just to add an early exit if the object was cancelled.

But when does that ObjectLock destructor run?

It runs at the `return` statement, since that's when the `lock` goes out of scope. In particular, it is **not** run before you call `ExitThread`.

Result: You left an object locked permanently.

You can imagine how variations on this code could lead to resource leaks or other problems.

Raymond Chen

Follow