What's up with Code Analysis rule CA2202: Do not dispose objects multiple times?

devblogs.microsoft.com/oldnewthing/20181126-00

November 26, 2018

Raymond Chen

You thought you'd be able to make it through the entire year without suffering through a CLR Week. And you almost made it.

But here it is. Ha ha. There's no theme for this week; it's just a collection of random CLR-related articles.

We begin with one of the more frustrating Code Analysis rules. Rule CA2202: Do not dispose objects multiple times.

Officially, the rule is to defend against buggy objects that do not implement `Dispose` correctly.

But I have another rationale for this warning: Disposing an object multiple times means that you don't really know who is in charge of the object. Calling `Dispose` means, "I'm finished, and I promise not to use it any more." If two people call `Dispose`, then the second one thought he could still use it but he can't because somebody else disposed it!

It's like saying "Last person to leave the room turns off the lights." Suppose there are two people in the room. One person leaves the room and turns out the lights. The second person is right behind and also turns out the lights. Turning off the lights is safe to do twice (the second time does nothing). But it means that the first person turned off the lights too soon. What if the second person wasn't directly behind? The first person just turned off the lights while there's still somebody in the room!

After the first `Dispose`, the object is dead. But there's still code that's using it, as evidenced by the fact that that other code also tries to `Dispose` it. If the other code knew that the object was dead, it wouldn't bother disposing it. You don't dispose dead objects, after all.

In my opinion, that is what rule CA2202 is trying to tell you. You lost track of the object's useful lifetime.

Raymond Chen

Follow