On partially-constructed objects, additional remarks, not as interesting

devblogs.microsoft.com/oldnewthing/20130808-00

August 8, 2013

Raymond Chen

Don’t worry. Our long national nightmare of CLR week is almost over.

I had originally planned to write an article about partially-constructed objects, but in the time since I queued up the topic (back in November 2005) to the time I got around to writing it up, I found that Joe Duffy had already written it for me!

On partially-constructed objects

Read it.

Okay, here are some follow-up remarks.

One place where people get caught out by partially-constructed objects is when they try to maintain a cache of objects (perhaps with a little bit of `WeakReference` action) and stash the objects into the cache before they are fully constructed:

```
class SomeClass {
 public SomeClass(...) {
  cache.Add(this);
  AdditionalConstructionWork();
 }
}
```

If the `AdditionalConstructionWork` takes an exception, then you end up with a partially-constructed object in your cache. (Mind you, you had one all along, but now it’s a persistent condition as opposed to a transient one.)

You might think to fix the problem by reordering the operations:

```
class SomeClass {
 public SomeClass(...) {
  AdditionalConstructionWork();
  // add to cache only after construction ran to completion
  cache.Add(this);
 }
}
```

But that still doesn't work once you have derived classes:

```
class Derived : SomeClass {
 public Derived(...) : base(...) {
  AdditionalConstruction(); // oops, what if this throws?
 }
}
```

The base constructor runs first, it successfully constructs the base object, and then puts it in the cache. And then the derived constructor runs and encounters an exception. You're back in the same boat with a partially-constructed object in the cache.

You want to wait until the object is fully constructed because you add it to your cache.

```
class SomeClass {
 static public SomeClass Create(...) {
  SomeClass c = new SomeClass(...);
  Register(c);
  return c;
 }
 protected static void Register(SomeClass c) { cache.Add(c); }
 protected SomeClass(...) { ... }
}
class Derived : SomeClass {
 static public Derived Create(...) {
  Derived d = new Derived(...);
  Register(d);
  return d;
 }
 public Derived(...) : base(...) { ... }
}
```

Raymond Chen

Follow