An amusing story about a practical use of the null garbage collector

devblogs.microsoft.com/oldnewthing/20180228-00

February 28, 2018

Raymond Chen

Some time ago, I noted that the null garbage collector is a valid garbage collector if the amount of RAM available to the runtime is greater than the total memory requirements of the program. This insight lets us make some statements about what you can assume from garbage-collected languages.

But it also can be looked at as rules for deciding when you can simply ignore the memory leaks in your program and just use a null garbage collector. Here's one developer who learned about an application of this principle:

```
From: k...@rational.com (Kent Mitchell)
Subject: Re: Does memory leak?
Date: 1995/03/31

Norman H. Cohen (nco...@watson.ibm.com) wrote:
: The only programs I know of with deliberate memory leaks are those whose
: executions are short enough, and whose target machines have enough
: virtual memory space, that running out of memory is not a concern.
: (This class of programs includes many student programming exercises and
: some simple applets and utilities; it includes few if any embedded or
: safety-critical programs.)

This sparked an interesting memory for me.  I was once working with a
customer who was producing on-board software for a missile.  In my analysis
of the code, I pointed out that they had a number of problems with storage
leaks.  Imagine my surprise when the customers chief software engineer said
"Of course it leaks".  He went on to point out that they had calculated the
amount of memory the application would leak in the total possible flight time
for the missile and then doubled that number.  They added this much
additional memory to the hardware to "support" the leaks.  Since the missile
will explode when it hits its target or at the end of its flight, the
ultimate in garbage collection is performed without programmer intervention.

--
Kent Mitchell                    | One possible reason that things aren't
Technical Consultant             | going according to plan is .....
Rational Software Corporation    | that there never *was* a plan!
```

Raymond Chen

Follow