Don’t keep track of information you don’t need

devblogs.microsoft.com/oldnewthing/20090216-00

February 16, 2009

Raymond Chen

This is sort of an extreme corollary to *Don’t save anything you can recalculate*. Sure, it sounds like such an obvious principle, but many people fail to understand its consequences.

Let’s look at the principle again. *Don’t keep track of information you don’t need*. I remember being asked to look at a customer’s program, and one thing that struck me was that the program had a bazillion different flag variables that it spent a lot of time setting and clearing. Here’s an oversimplified example:

```
void CConfiguration::ShowDialog(HWND hwnd)
{
  m_fShowingDialog = true;
  DoModal(hwnd);
  m_fShowingDialog = false;
  m_fDialogHasBeenShown = true;
}
```

A naïve reading of the code would lead one to believe that the main purpose of the program was to set and clear flags! Upon closer inspection, I found that most of the flag variables were write-only. Although the code went to great pains to update the flags as it ran, there was no code that actually cared whether the flag was set or not.

I asked the customer liaison about this, thinking that maybe I missed something. Maybe the variables were being used in a subtle way that I failed to appreciate. Maybe they were used during debugging.

The customer liaison got back to me. “When I asked them about it, they just said, ‘Ah, yes.’ Apparently that code was written by one of their guys named Bob. I guess he likes to set and clear flags a lot.”

But is there any code that actually checks the values of those flags?

“Probably not. He just set and cleared those flags because he figured maybe someday he might need them.”

Indeed, once in a while, he actually tested one of the dozens of flags he spent most of his time setting and resetting. But the vast majority of the flags were write-only. (And since most of the flags went unused, I suspect that many of them weren't even accurate.)

Obviously, this is an extreme case of write-only variables, but it illustrates the point. Next time, we'll look at one of the consequences of the principle of not keeping track of information you don't need.

Raymond Chen

Follow