Before you can learn to recognize what's wrong, you must learn to recognize what's right

devblogs.microsoft.com/oldnewthing/20060710-12

July 10, 2006

Raymond Chen

Sometimes, when I'm debugging a problem, I'll ignore a particular thread and somebody will ask, "What is that thread doing, and how did you know to ignore it?" My reply: "I have no idea what it's doing, but whatever it's doing, it's normal." Tess has been running an excellent series of posts on debugging the CLR, but one of the most important lessons is where you learn about things to ignore when debugging an ASP.NET hang. Hangs and deadlocks are tricky to debug because there is no unhandled exception that says, "Look at me, I'm a bug!" Instead, the program just grinds to a halt and you have to go spelunking to figure out why. What you are looking for is anything out of the ordinary, but in order to recognize what that is, you first need to know what is ordinary. So do that. Run your program, let it do its thing, then break in with the debugger and take a look around. Make a note of what you see. Those are things that are going on when nothing is wrong. *This is what your program looks like when it is running normally*. Now that you know what normal operations look like, you can recognize the abnormal stuff.

Note that you don't even have to know what all those normal things are. For example, when I connect to a process with the debugger, I often find threads lying around which are waiting inside RPC or the kernel thread pool. I don't know what they are doing, but since they are always there, I don't pay much attention to them.

Raymond Chen

Follow