Gettin' Your Head Straight

A koder with attitude, KV ANSWERS

YOUR QUESTIONS.

MISS MANNERS HE AIN'T.

Kode Vicious is hungry. He sustains himself on your questions from the software development trenches (and lots of beer). Without your monthly missives, KV is like a fish out of water, or a scientist without a problem to solve. So please, do your part to keep him sane (or at least free from psychotic episodes), occupied, and useful. Send your koding conundrums and other questions to kv@acmqueue.com, and if we print your letter, we'll thank you with a nifty orange mug and an incredibly bright LED flashlight keychain.

Dear KV,

One of the biggest problems I have is memory. Not the RAM in my computer, but the wet squishy stuff in my head. It seems that no matter how many signs I put up around my cube, nor how often I turn off all the annoying instant messaging clients I need to use for work, I can't get through more than 15 minutes of work without someone interrupting me, and then I lose my train of thought. If this happens when I'm reading e-mail, that's not a problem, but when working on code—in particular when debugging a difficult problem in code—this makes my life very difficult. What tricks are there to being able to maintain a train of thought without moving up the side of a mountain?

Bugged while Debugging

Dear Bugged,

I really ought to learn not to read my mail during breakfast, because the less I think about the wet squishy thing in your head, the longer I'll be able to keep my beer and cornflakes down, and we all know that breakfast is the most important meal of the day!

Memory is a funny thing. Well, some memories are,

Got a question for Kode Vicious? E-mail him at kv@acmqueue.com—if you dare! And if your letter appears in print, he may even send you a Queue coffee mug, if he's in the mood. And oh yeah, we edit letters for content, style, and for your own good!

while others are more in the horrifying genre and are actually best left exactly where you forgot them. It does amaze me how dense

one's coworkers, loved ones, and children can be to one's state of concentration. I have tried signs, headphones, turning my back to everyone, closing the door to my office (when I was lucky enough to have one), and one time I tried flares. All of these were to no avail. People invariably ignored the signs (they couldn't be referring to *them*—their problems were important!) and knocked on the door. The problem with the flares was twofold. One was the fire regulations and the other was that they burned out and had to be replaced. I gave up on trying to remain uninterrupted in an office environment.

What you really want is a way to remember what you were doing when you were doing it, and debugging is one of the best examples of this. Nothing is more annoying than to come back to a problem you were working on and not remember what you had already tried. Having to redo your experiments and measurements again is a hair-tearing experience, which is why I started shaving my head years ago—it's far easier than tearing the hair out. What is important is to realize that you *are* doing experiments and taking measurements, and that will lead you to your answer.

Real scientists, as opposed to lame hacks who claim to be scientists, know how to formulate ideas—called *hypotheses*—and test them. They write down each hypothesis, then describe the experiment and the results. They keep all of this data in logbooks or notebooks. One of the earliest examples of a logbook being used in this way by computer scientists was when Grace Murray Hopper recorded the first bug, a moth found in a relay of the Harvard Mark I computer, in the logbook that was kept in the machine room with the computer. Gone are the days when you can debug a program by pulling a moth from a relay, and, well, good riddance—I mean, ick! It is still a fine example of how one should proceed when trying to figure out a complex problem.

Speaking of examples, I will present you with one of

my own from a kernel debugging session I was recently trapped—er, I mean involved—in.

Figure 1 shows a couple of pages from my debugging notebook, with some unnecessary details removed. While the details are unimportant—unless you, too, are adding IPv6 support to a new IPsec protocol stack, which I bet you're not—there are a few points to note about what I have recorded.

The first is that I included the output of all the commands I ran—in fact, far more than is shown in this example. Keeping the output around for re-reading later is like making sure you have all your specimens in specimen jars. As we all know, rerunning the same test on a given piece of software does not always give the same results, and thanks to Murphy's law, the more you need the test to be repeatable, the less likely it will be. Save your output.

The second point to consider is that I record and keep all the hypotheses, including the incorrect ones. Attempting to re-create your state of mind during a previous debugging session, without the use of mind-altering substances, is even more difficult than rerunning the test code and getting the same results. There is nothing to be embarrassed about when you get one of your educated guesses wrong; that's how science works. You make a guess, based on what you know or think you know at the time, and then you test it. Keep all your hypotheses.

Lastly, I explain what I fixed. Just because your test passes or your code doesn't crash doesn't mean that you have completed your debugging. It might be that your fix hides the problem, until some other issue causes the bug to resurface. If you don't remember what you did to fix a problem, then when you see the same problem again you won't remember how you fixed it, and that is going to suck. Take notes on the cure, as well as the disease.

Plenty of other things can go into a debugging log, and they need not be kept in electronic form. KV actually has a friend who likes to take notes with quill pens, which he makes himself. KV seems to have many such eccentrics as friends and hasn't a clue why.

The important thing isn't so much how you keep the log, but that you keep enough relevant data for the debugging log to be useful to you over the long term. The fact that disks are cheap, well, that just works to your advantage.

KV

KODE VICIOUS, known to mere mortals as George V. Neville-Neil, works on networking and operating system code for fun and profit. He also teaches courses on various subjects related to programming. His areas of interest are code spelunking, operating systems, and rewriting your bad code (OK, maybe not that last one). He earned his bachelor's degree in computer science at Northeastern University in Boston, Massachusetts, and is a member of ACM, the Usenix Association, and IEEE. He is an avid bicyclist and traveler who has made San Francisco his home since 1990.

© 2007 ACM 1542-7730/07/0700 \$5.00