Off the Trails

by Michael Bolton

ISTOCKPHOTO

It was a busy conference, with testers milling about the hotel hallways, moving from session to session. It was also a tester's dream come true, for there, at the top of the hallway, were two computers—Windows- and Internet Explorer-based information kiosks, each with its own screen, mouse, and full keyboard. This was like candy for a pair of enthusiastic testers like James Bach and me. We began by surveying the systems. Apparently we had very limited rights to do anything other than to operate the browser. Default Windows programs like Accessories were unavailable. Internet Explorer's File/Close menu option was grayed out. Control-Alt-Delete brought up a message saying that the Task Manager was disabled due to an administration policy. We observed that the service provider had prevented access to most of the Internet—the only sites accessible were those associated with the hotel itself, the chain to which it belonged, and some of the major airlines.

Conference participants had to walk past the kiosks to get to the presentation rooms. During the breaks, a small crowd would gather around the machines as we tested, and people would eagerly shout suggestions as to what to try next. It was fun, and the audience had some great ideas: "Where else can you go? Try clicking that link! Try right clicking! Have you tried to open a command window? What if you put CMD.EXE in the address line?" Most of our discoveries were about activities that we were prevented from doing, but we learned some other things, too. At one point, I noticed that the times on the two displays were different. "These browsers are running on different boxes," I said.

"Are you sure? How do you know?" James asked.

"If they were running on the same machine, they'd show the same time," I replied.

"True," James said. "Did you notice that until now we hadn't really considered the question of whether they were running on the same machine or different ones?" A good point; the box is especially black when you can't see the box at all.

I had to give a presentation, so I reluctantly left my self-appointed testing assignment. I returned an hour later, and as I approached the kiosks, James was grinning.

"Check this out," he said. "Put your hands on either side of the keyboard—nowhere in particular—and spread out your fingers. Now, just flap your hands lightly on the keys, as though you were playing the congas, but not hard." I did. To my astonishment, after a few drumbeats, the image on the screen seemed to rotate counterclockwise ninety degrees. The browser was suddenly sideways. Wha…? "That's how I found it. It took me a minute to figure out why it's happening," said James. "You try."

I looked at my hands. The left was over Shift, Alt, Ctrl, Caps Lock, Tab, A, and Z. The right covered Ctrl, Shift, Alt, the Windows key, and the cursor navigation keys. I tried pressing Tab, which didn't seem to have an effect. The cursor keys—up, down, left, right? Nope. Ctrl with those keys? No. Alt with those keys? No. Combinations? Ctrl-Alt-Down? *Yes*. When I pressed Ctrl-Alt-Down, the display suddenly flipped upside down. Ctrl-Alt-Left rotated the image such that the top was on the left, Ctrl-Alt-Right put the top on the right, and Ctrl-Alt-Up set the screen right-side up. For a few moments we amused ourselves by showing the effect to the other testers. Then I began to wonder what might happen if we changed strategy and toured the keyboard systematically.

The Windows key brings up the Start menu, but in combination with other keys, it provides a little-known set of shortcuts. By default, Windows-A doesn't seem to do anything. Windows-B brings up the taskbar—think Windows-*B*ar. Windows C doesn't do anything. Windows-D minimizes the applications on the screen to expose the *D*esktop. Windows-E brings up *E*xplorer. Windows-F presents the Windows *F*ind (Search) feature. At the kiosk, I walked through the alphabet from Windows-A, but all of these combinations seemed to be disabled. I was losing enthusiasm for my systematic tour, until I got to Windows-L, for *L*ogin or *L*ock—and there, to my astonishment, was the Login screen.

At least it was passworded. I considered trying to guess the password—maybe it was some combination of "Admin" and the hotel name, maybe combined with a 1 or a 0. Going through all of the possibilities felt like too much work, and lunchtime was approaching—and besides, there was that button in the lower right corner of the screen that said "Turn off computer." I considered the potential consequences for a moment, then I clicked the button. The machine rebooted. As it was starting up, I pressed the F8 key and got access to the command prompt. I now owned the machine. Time to stop.

I rebooted and left the system alone. To my relief, the Windows logo appeared, Internet Explorer came up, and the kiosk software started inside the browser. Once it did, we noted the support number for the company that created and maintained the kiosk. We had to believe that the service provider's programmers and technicians didn't know about this vulnerability or had forgotten to change the key combination to something more obscure. James called the company to alert it to the vulnerability. The first-level support technician didn't understand the implications of what James was reporting, but her supervisor did and took the report seriously.

This is an example of what we can find with freestyle exploration. We had been working implicitly under a very open-ended mission: Learn something about this system starting from scratch. We designed and executed tests in parallel with interpretation of test results and with learning. We created questions and answered them on the fly. Sometimes—as in the case of discovering that the two systems were on different machines—we obtained the answer before we had framed the corresponding question.

Not knowing from the outset where we were headed, we used an approach that we've found effective for all forms of test design: *Start anywhere you like.* Every journey begins with a single step. For the very first step of a freestyle exploratory session, we can choose something that might be interesting, fun, familiar, or useful; something that we might want to learn about; or something that we could infer might be important to some client. Start with a question, a conjecture, a specifically defined task, or a completely unmotivated activity and see where it takes us. Did we learn anything? Did we observe something surprising? Did we have a new idea? Did we suddenly recognize a new risk? Is there something here that's worth recording for future reference or that might be a point of departure for a more specifically chartered session later on? What previous knowledge did we bring to the table? What do we know now that we didn't know before? What skills might we want to practice? When we report our findings, how does our client react? If we wanted to do something like this next time, what would we do differently?

Whatever we discover, we may choose to focus on it, explore it, and learn something about it, or we may choose to refocus, step off in a different direction, and investigate something new. Note that we can follow this process with anything that we might wish to test or review—a document, a prototype, a flowchart, a whiteboard diagram, a new feature, a unit, a complete program, or a test idea itself.

Would we test an entire product using *only* freestyle exploration? Of course not. A good test strategy is driven primarily by risk. Many problems in software products are familiar, and we can be enormously productive by anticipating those problems and seeking them out, aided by risk lists, coverage outlines, and other forms of checklists. Yet we must also foster the discovery of new risks, problems that we haven't anticipated. We can't ever be sure of what we're going to find, and sometimes we can't recognize what we're looking for until we see it. There are at least two ways of addressing that problem. One is to remain alert, alternating between focusing and defocusing, switching between random and systematic actions as we follow our planned routes through the program. Another is to wander every now and then—quite deliberately—away from the plans that we've set for ourselves and see what we find hiding in the bushes when we're off the trails. {end}

Whether testing a product for business or for pleasure, what problems did you find incidentally—or accidentally?

Follow the link on the StickyMinds.com homepage to join the conversation.

We deliver web applications

www.railsware.com/be-agile

railsware