**INFO TO GO**

- Job interviews are designed to reveal your strengths and weaknesses—just like any other test.
- Use the test strategy outlined in this article to prepare. It covers things to do before, during, and after an interview.

# When You're Tested

by Jonathan Bach

SOME PEOPLE THINK TESTERS BREAK SOFTWARE, but I think it's more accurate to say that we look for what's already broken. When it's delivered to us, we start the process of revealing that it's only as good as it's programmed to be.

People are like that, too. Place us in any situation and we reveal information about our programming. No one breaks us, but a situation may reveal more weaknesses than strengths, making us *appear* broken.

Job interviews are notorious for this. We sit in front of the interviewer and it feels as if all our weaknesses are highlighted on their clipboard. We're being tested, and at risk are our pride, our reputation, and our financial future.

There's even more pressure in a bad economy. According to the Bureau of Labor Statistics, the national unemployment rate for 2002 was around six percent.

I was part of these statistics last May. None of my seven years of testing, test leadership, test management, and consulting seemed to matter to anyone. I sent thirty résumés to thirty different companies, and heard nothing. I began to feel like version 5.0 in an 8.0 world.

So when an interview opportunity came, the pressure was on. I felt like all the software I'd ever tested—about to be delivered in front of a panel of Ph.Ds with state-of-the-art laser scanning machines able to map every flaw within me.

But wait…in my testing career so far, testers aren't Ph.Ds, and there are no scanning lasers. It has always been a human process—people using their expertise and imagination to reveal problems. I would likely be tested by a human, and hey, I'm a human tester, too!

To do well in my interview, all I had to do was apply the same testing principles to myself that I had used when testing software. If I focused on discovering my weaknesses before the interview, it would prepare me for if and when they were found during the interview.

Suddenly, I was not obsolete software, but a solid program speckled with minor nervousness bugs that made me look more flawed than I really was. So I came up with the following test strategy for myself—things to think about before, during, and after the interview.

## Before the Interview

**Scrutinize your résumé.** One source of nervousness is fearing that your résumé will be picked apart. It may, or it may not, but the only thing to worry about is if you can't explain something confidently and succinctly when asked. If there's something on there that you can't talk meaningfully about, remove it.

**Practice—test something in fifteen minutes using a test heuristic.** It's a safe bet that they're going to ask you to "test this." Test this chair, this flashlight, this pen, this notebook, this window blind, this SQE mouse pad you got at the last testing conference.

Heuristics can help. Here is one I learned from fellow tester (and brother) James Bach, and it's a useful way to think quickly about basic elements of software, but it also can be used to test inanimate objects. It's called "San Francisco Depot" (SFDPO):

**S**tructure—What testable things is this comprised of?
**F**unction—What is this thing meant to do? What problem is it meant to solve?
**D**ata—What kinds of input can it handle?
**P**latform—What does this thing depend on in order to fulfill its purpose?
**O**perations—How is this thing meant to be used?

**Prepare a list of questions for the interviewer.** Seems like common sense, but it's more important in test interviews because questions are a tester's best tools. They're meant to provoke information, so if you have no questions for the interviewer, they may think you won't seek information on the job either. So test the job as if it's software. Does it meet your requirements? For example, is overtime allowed? Is it required? Are you designing tests or just running ones that somebody else wrote? Are you just one of two testers in a project of one hundred developers?

**Remember that technical skill isn't everything.** You'll face scrutiny, yes, but your strengths will be evaluated, too. Other candidates may be more qualified, but that doesn't mean they're automatic hires. For example, some of the best testers I know do not have computer science degrees, and sometimes the ability to think, communicate, and learn quickly supersedes requirements for technical skill.

## During the Interview

**Ask any questions you've prepared.** They usually ask if you have any questions at the end of the interview. I ask mine up front. No sense in wasting time. You're testing them, too.

**Stay centered.** Being centered means being aware of how you're being provoked. If you're talking fast to match their pace, slow down. Talking fast sounds impressive, but it's not a skill testers need to have. If they seem annoyed that you're taking too much time on an exercise, it may not be because of speed, but lack of agility—how active your imagination is when designing tests. Then again, their over-critical attitude may be an illusion to provoke you. Some interviewers purposely exaggerate their attitudes—frowning, belligerence, being argumentative—to see how you react to conflict.

**When given an object to test, think out loud.** Proclaim assumptions you have, ask clarifying questions, ask for more context, or even for hints. Note the properties of the object (also known as a dimensional analysis)—color, weight, size, buoyancy, even conjectures about specific gravity or molecule density may not be important, but list them anyway. If there is a whiteboard, use it. List your observations and test ideas. It may take more courage for introverts, but it always seems to impress the interviewer (it impressed me when I was one) and is very telling—showing confidence and openness. If there isn't a whiteboard, use a pad of paper and move it close so they can see what you're writing and drawing.

**If given a logic problem, don't worry about getting the answer right.** Logic problems are usually given to see how you think, not to see if you know the answer. Think of them as opportunities to show how well you can announce your observations and conjectures. If you're feeling stuck, admit it. The ultimate defense against scrutiny is to think out loud and to admit your limitations. That gives them confidence that when on the job, you won't drown when splashed with a cup of cold water. (Test your logical problem-solving skills right now with the exercise in the sidebar below.)

**Evaluate their testing of you.** You know that old anti-nervousness trick about imagining them in their underwear? Well here's another one along the same lines: *You may be more expert about how to test your skills than they are.* You could be the best person or the worst person for the job, but they're the ones who have to invent the right set of tests to reveal the truth, and they have to do it quickly, so how did they do? What have they missed? Did you just get away with a vague answer? Since you're the developer, you know where your weaknesses are. If they're not finding them, that may give you some confidence that you're a better tester than they are.

### Test Yourself

**0, 1, 1, 2, 3, 5, 8, __**

**What number comes next in the series?**

***Answer:*** *The next number in the series is 13. This pattern of numbers is referred to as the Fibonacci Series. You add the first two numbers to get the third, then add the second and third numbers to get the fourth, etc.*

## After the Interview

**Don't assume that they are test experts or that they even understand testing.** Sometimes, the people doing the interview were asked to do it by their boss. I've seen recruiters "fail" a candidate who had great testing skills but didn't give the correct answers to the logic problems listed on the interviewer's clipboard.

**Don't take it personally if you don't hear back from them.** There could be a lot of reasons you didn't get the job that have nothing to do with how well you did. Sometimes there's a last-minute decision to suspend interviews and not hire anyone for a few months, or ever.

**Consider that fate has other plans for you.** EasyPadPro isn't the word processor of choice for the *New York Times,* but it could be perfect for the *Jonsville Gazette.* During one interview, as I was answering a question about how I would test a chair, the interviewer suddenly looked up from his email (which he was checking as I gave my answer) and whipped one of those beanbag stress balls hard against the door behind me. Why nerves of steel were a requirement for that job, I'll never know, but something tells me I'm glad I didn't get called back for that job.

**Remember that some flaws or weaknesses don't matter.** Ever been part of a test team that knowingly shipped with bugs? You know that there are a lot of reasons for that, but one is that those bugs aren't likely to affect customers enough to degrade their opinion of quality. Customers will forgive flaws that don't minimize the benefits they get. The same is true about interviewers.

**Think of each interview as practice for the next one.** Until you're hired for the job you want, don't give up.

## After All

Like software, humans are only as good as their programming. Life is a series of tests that reveal information about who we are, and how we interpret that information is important. But ultimately, we are not only just software and tester, but developer, program manager, and customer—deciding what's important to fix, what's too costly to fix, and what we can live with.

You may not find all your weaknesses before the interview, but you may find the important ones. With that out of the way, you might drive to the interview more eager to show off the features you were built for. STQE

*Jonathan Bach (jbtestpilot@hotmail.com) was hired after a month on unemployment, on the last day of May 2002. He is a contract tester for Volt Computer Services onsite at Microsoft in Redmond, WA.*

***STQE* magazine is produced by Software Quality Engineering.**