Get that job at Facebook

Par Carlos Bueno, vendredi 20 juillet 2012, 21:14

Interviewing for a technical job is hard, and so is being the interviewer. You want to get that engineering job at Facebook, and we want to hire the best people (you!). Knowing what to expect on both sides can go a long way toward making the process work better.

Preparation is important. Hiring, here and across the industry, is like shooting a few protons into the very large space of your life experience, hoping to get enough information back to say that yes, this specimen is definitely made of elemental awesomium. We're trying to build a picture of your abilities as a professional and a colleague from a handful of data points.

How we do it
Every role is different and we are always tweaking things, so don't freak out if your experience is different from what's laid out here. Typically it will start with an email or a phone call from a recruiter. Perhaps they found you online, or you applied directly, or a friend recommended you. We also have an intern program, where we invite talented students to work with us for a few months.

After talking with a recruiter and passing the basic hurdles, a coordinator will schedule you for a phone screen or an initial in-person interview. If the feedback is good, we'll invite you for a longer series of interviews at our office. If *that* goes well, we will make you an offer. Yay!

Technical interviews tend to follow the same general pattern: talk about your past work, some interactive coding, answering anything you're curious about, and selling you on the idea of coming to work for us.

Phone screen / Onsite
The initial screening interview is a 45 minute talk with a potential coworker. The idea is to slot you with someone in your general area of expertise. They will explain who they are and what they do, ask you about interesting things from your resume, your skills, motivation, interests, and so on.

The bulk of the time is spent on coding exercises. The interviewer will send you a link to a collaborative editor and ask you to solve some programming problems. More about that in a sec.

Onsite Loop
The loop is several interviews back-to-back on the same day, usually with a lunch break. Instead of coding in a text editor, you will likely be asked to write code on a whiteboard. And of course there will be time to ask the interviewer anything you want.

What we look for
These traits are not all we look for, nor all we care about. But here are some of the things that interviewers base their decisions on:

Fit: We're looking for your ability to understand and explain complex ideas. We look for a healthy level of enthusiasm, curiosity and motivation. Interviewers are evaluating you as a potential colleague. We have a ridiculous ratio of users to engineers and ship code five days a week. We want people who know how to make a large impact, who can move quickly, make bold choices, and be transparent about what they are doing.

Generalism: We need many kinds of specialists, but we also look for people who can fill other roles in a pinch. This means understanding the "stack" above and below your area of expertise. Bonus points for having multiple areas of expertise. It's not unusual for someone at Facebook to work on machine learning, then move on to web performance, build and maintain a new backend tool, then spend a year on photos.

Architecture: Can you arrive at an answer in the face of unusual constraints? We want to see how well you can visualize the entire problem and solution space. We also want to see how much you've thought about Facebook in particular and some of the unique problems we face. How would you architect a world-wide video distribution system? Or Facebook chat?

Coding: We don't ask puzzle-type questions, where you need to know a trick. Even so, the coding questions you get asked may sound contrived. This is because they are contrived, in the sense of being designed for a special purpose. They have to be simple enough to explain in a few minutes and solvable in 10 to 30 minutes. But they must also require knowledge, skill, and concentration to solve.

Good coding problems are fractal in nature. They can be extended arbitrarily to gauge the depth of your knowledge. For example, you might be asked to solve a problem any way you want. Then you'll be asked to solve it again in constant space or sub-linear time.

Incidentally, the ability to give total focus to a problem, no matter how basic it sounds at first, is something we pay close attention to. How you attack a problem is at least as important as the answer.

We will ask you to do a *lot* of coding during the interview process, because programming ability tends to correlate strongly with how well people perform as employees. We even have a large set of take-home questions. It can't hurt to check them out and maybe solve a couple before you even submit your resume.

How to Prepare

Steve Yegge of Google wrote an excellent post about interview prep a few years ago. If you haven't read it, go read it. If you have, go read it again. The tips Yegge lists are very good, though I have never seen anyone bring their own whiteboard markers. I'll paraphrase some of it here and add some more:

Take your time preparing. Do code katas and practice interviewing with friends. Try solving the interview questions on our site. See our tech talks to get a feel for how we do things and the scale of problems we're trying to solve.

For phone screens, make sure you are in a quiet place with a good internet connection. Headphones are handy. I forgot this during my first interview with Facebook, and had to type code while keeping the phone jammed between shoulder and ear, like a nerdy T-Rex.

Practice writing code in a simple text editor without syntax highlighting or completion macros. Don't let little surprises throw you off your stride during the interview.

Impress us with your mastery of whatever language you're best at. Don't use a language you know less well because it's trendy or you think it will please the interviewer. This is a very common pitfall.

More generally, skills on your resume are fair game. If it says "expert in X," we will try to schedule you with a proven expert in X, so be prepared. If you are not, leave it off. I'd rather have a short list of the things you're awesome at than pages of everything you've ever done.

A good skill to cultivate is the ability to change your point of view at will. Sometimes you'll encounter a problem that seems like it should have an elegant solution, but in fact must be brute-forced or approximated. If you're stuck on a problem, try to think of any way to solve it, no matter how clumsy or inefficient. Then improve on it. Getting something that works is better than nothing.

Hard training makes for an easy battle. Brush up on techniques that you may not use every day, but are very useful when you need them: recursion, graph theory, tree traversal, combinatorial problems, etc.

You might be asked to implement some well-known library functions. Knowing at least a little bit about how things work under the hood is highly recommended.

Another kind of coding question might be parsing some data format or mini-language. Aside from CS nerd-points, these problems exercise your ability to reason about edge cases and handle lots of state in your head.

Give feedback. We regularly survey candidates about the interview process and take feedback seriously.

Ask questions! Take advantage of the time to ask your interviewer about working life, bootcamp, the interview process itself, how the company is organized, or really anything at all. I recently spent a few minutes at the end of an interview chatting about power efficiency in our datacenters. The candidate was genuinely curious and I did my best to answer. Always remember that you are interviewing us as much as the other way around.

Above all, relax! And if you are on the fence about applying to Facebook, do it. I've worked at companies of all kinds, from two-person startups to billion-dollar government projects. Facebook has the resources and leverage of a large company, but as an engineer you have freedom and responsibility far beyond the typical. That's how we punch above our weight. It's an amazing combination that you don't find very often. Check out facebook.com/careers.

Carlos Bueno, an engineer at Facebook, knows how to move Mt. Fuji.