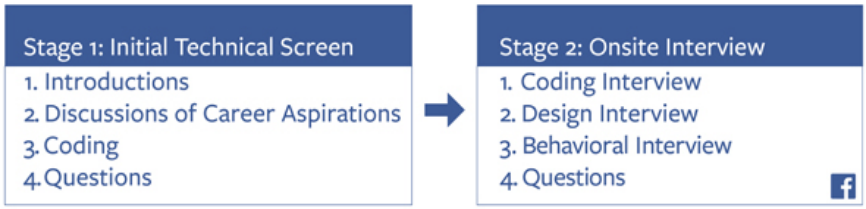
Preparing for your Software Engineering Interview at Facebook

By: Facebook Careers | June 27, 2016

If you're interviewing at Facebook or you're just curious about the process, we want to be transparent about what to expect so you feel well informed and have a positive interview experience. Three Facebook Software Engineers have broken down the stages of our Software Engineering interview process, covering the Initial Interview and Onsite Interview with many tips, links and insights to help you prepare and do your best.



STAGE 1: INITIAL TECHNICAL SCREEN

My name is Bosmat, I'm an engineering manager at Facebook in Menlo Park. I've been with Facebook since 2011 and I regularly interview engineering candidates. Our initial interview serves as a screening step to determine whether to continue with a full series of onsite interviews. This interview will be the first with a Facebook engineer and is primarily a coding interview. Below is some insight on what to expect, how to prepare and some tips for a successful interview.

What to Expect:

- **Introductions:** The interviewer will first introduce herself/himself and explain what they do at Facebook.
- **Career Aspirations:** For the next 5-10 minutes, the interviewer will ask questions about your experience and your career aspirations.
- **Coding:** The next 30-35 minutes will be spent on coding.

- This takes place in an online collaborative editor shared between you and the interviewer (or on the whiteboard if you do the initial interview in person).
- You are given one or more coding questions to complete in this editor. We ask questions that are short enough to explain in a few minutes and to solve in 10-30 minutes.
- In this section we try to understand your approach to problem solving.
- We typically don't ask trick or estimation questions (we don't care how many ping pong balls can be fit in Sea World).
- You could be asked to solve a problem in any way you choose, and then the interviewer could add further constraints or requirements.

- **Ask Us Anything:** The last 5 minutes is for questions. This is a great opportunity to get an insider's perspective directly from a Facebook engineer.

How to Prepare:

- **Invest time in preparing:** It's important for any engineer, even senior ones, to brush up on their interview skills, coding skills and algorithms. An interview is typically different from your day-to-day job. This is the first technical interview in the process, so any preparation for this interview will be beneficial for the next ones.
- **Practice answering many different coding questions:** Practice answering a coding question with the most efficient bug-free solution without using a compiler. A few resources that offer coding questions to use for practice: Careercup, Topcoder, Project Euler, or Facebook Code Lab.
- **Write code in a simple text editor:** In the interview you will write your code in a similar environment (like CoderPad) without syntax highlighting or auto-completion.
- **Practice by coding by hand:** Coding interviews will be done on a whiteboard. Practice some of the questions with a good old fashion pen and paper to help prepare.
- **Practice under time pressure:** You will have a limited time for the coding question, so it will be important to finish it in time. If possible, have a mock interview with a friend to simulate the interview experience.
- **Go over data structures, algorithms and complexity:** Be able to discuss the big-O complexity of your approaches. Don't forget to brush up on your data structures like lists, arrays, hash tables, hash maps, stacks, queues, graphs, trees, heaps. Also sorts, searches, and traversals (BFS, DFS). Also review recursion and iterative approaches.
- **Think about your 2-5 years career aspirations:** You will be asked to talk about your interest and your strengths as an engineer.
- **Prepare 1-2 questions to ask your interviewer:** There is 5 minutes at the end of the interview for this.
- **Suggested reading resources:** Cracking the Coding Interview, Introduction to Algorithms, Algorithms in C.

Tips for the Coding Interview:

- **Think out loud:** We pay a lot of attention to the way you solve problems, which can be as important as having the right answer. Thinking out loud gives the interviewer insight into your thinking process and can also help them follow along with your solution. Moreover, it allows them to give hints when needed.
- **Locate a good interview spot:**Choose a quiet place and ensure that you have good Internet connection and strong phone reception. Headphones will help with having your both hands free for coding.
- **Speak clearly:** Ensure you are speaking clearly and likewise, if you can't hear the interviewer clearly, let them know so they can accommodate! You don't want to waste the whole interview trying to understand each other.
- **Use the programming language you're best at:** It's important to write your solution correctly and in time, so use the language you are most familiar with.
- **Manage Your Time Effectively:** Spend some time figuring out the ideal solution to the question. Don't jump too quickly into brute forcing the first solution that comes in mind. If you can't find a better solution in a reasonable time, start writing a working solution, then iterate and improve it as you go. Some interviews end without any coding because the interviewee couldn't find the ideal solution. It's better to have non-optimal but working code than just an idea. Once you have a working solution, you can then try to improve its efficiency, code design or any other aspect of it.
- **Share your reasoning:** Make sure you can talk about your solution; you will probably be asked to explain them. Engineering is all about tradeoffs, so be prepared to discuss them.
- **Find and fix the bugs by yourself:** Don't wait for the interviewer to find them for you.
- **Use the hints you are given:** Usually, the interviewer knows the question well enough to know which hints will help you next if you get stuck.

STAGE 2: ONSITE INTERVIEW

My name is Brent, I'm a software engineer at Facebook Seattle. Interviewing with any company can be a nerve-racking process, and the best thing you can do to ensure your best possible outcome is to *prepare, prepare, prepare*. To help you prepare for your Facebook interview I've put together a few tips about what you can expect, how to study and tips for each type of interview.

As an interviewee for an engineering position at Facebook, you're going to have 4 or 5 interviews over the course of the day. These will be distributed across 3 different types of interviews:

1. The coding interview – where you'll solve some general coding questions.

2. The design interview – where you'll be asked to show off your design skills. The design question will be focused on either systems or product, depending on your background.

3. The behavioral interview – where you'll talk through your previous work experience, motivations, and a number of other behavioral questions.

Unless you've scheduled your interview for very early or very late in the day, someone from engineering or recruiting will take you to lunch. This will give you a chance to ask lots of questions of someone who isn't interviewing you.

1. The Coding Interview - What to Expect:

The coding interview is typically harder than the initial interview: we ask more difficult questions and have a more exacting evaluation. This interview is 45 minutes. Not all interviewers follow the exact same time breakdown, but the following is typical:

- **Introductions:** The first five minutes will be an introduction and possibly brief questions about your background.
- **Coding:** The next 30 minutes will be one or more coding problems.
- **Ask Us Anything:** We try to reserve the final five minutes for your questions for the interviewer. This part gives you a chance to learn more about Facebook from someone in engineering and gives your interviewer a chance to learn more about your interests.

How to Prepare:

If you haven't already, check out Bosmat's post about coding interview preparation above. She provides some great resources for coding interview preparation that are useful for both the initial screen and the onsite coding interviews.

2. The Design Interview - What to Expect:

The design interview is 45 minutes. These almost never involve coding - you'll spend the interview talking and drawing on the whiteboard. As with all interviews, the interviewer will typically save the last five minutes for your questions. The purpose of the interview is to assess the candidate's ability to solve a non-trivial engineering design problem. To that end, your interviewer will ask you a very broad design problem and evaluate your solution.

There are two types of design interviews: systems design and product design. I've outlined the specifics of the systems design interview and Dan, a software engineer at Facebook in Menlo Park, describes the product design interview below.

We try to match candidates to engineers with related expertise. Candidates come from all sorts of backgrounds: some build complex user interfaces, others build network libraries, others build platform APIs for third parties. We expect every candidate to be familiar with basic design principles, but we'll try to match you with a interviewer so that the conversation is as engaging as possible. We aim to have the conversation be based in an area slightly familiar to you, allowing you to demonstrate your design abilities on an unfamiliar problem, but in a space that is not completely foreign to you.

Systems Design Interview - How to Prepare:

- **Improving upon a design:** Think about and review the complex systems you've already designed. What would you change about your approach if you did it all over again? What worked well?
- **Designing from the ground up:** Think about how you'd design a system that Facebook (Or Twitter, Google, Uber, Dropbox, etc) already has. It's a good thought exercise to think through the complicated, high-scale systems that you already use every day. How would design it from the ground up?
- **Blog Up:** Read engineering blogs about approaches that work and false starts big companies have had along the way.
- **Start with requirements:** Your interviewer might ask: "How would you architect the backend for a messaging system?" Obviously this question is extremely vague. Where do you even start? You could start with some requirements:
  - How many users are we talking about here?
  - How many messages sent?
  - How many messages read?
  - What are the latency requirements for sender->receiver message delivery?
  - How are you going to store messages?
  - What operations does this data store need to support?
  - What operations is it optimized for?
  - How do you push new messages to clients? Do you push at all, or rely on a pull based model?
- **Ask Us Anything:** The last 5 minutes is for questions. This is a great opportunity to get an insider's perspective directly from a Facebook engineer.
- **Design tutorial:** This is a good online tutorial about design questions.

Product Design Interview - How to Prepare:

- **Reflect on your projects:** Think about the projects you've built. What was easy, and what was difficult?
- **Your interviewer might ask:** "Tell me how you'd design an email server." Some questions you might want to think about:
  - How do you store mail, especially as the system gets large enough that it won't fit on one machine?
  - How do you handle mailing lists with large numbers of recipients?
  - How do you handle people abusing the system for spam?
  - How do you guarantee the reliability of the system in the face of potential system failures?

- **A different interviewer might ask:** Tell me how you'd design a client-server API to build a rich document editor." Some questions you might want to think about:
  - How does the client request data on the document from the server, especially as the document gets large enough that we wouldn't want to download it in a single request?
  - How do we represent the rich document aspects like bold and italics in our API response?
  - How do we design the system so that new features can be added on the server without breaking older clients?
- **Start with requirements:** Your interviewer might ask: "How would you architect the backend for a messaging system?" Obviously this question is extremely vague. Where do you even start? You could start with some requirements:
  - How many users are we talking about here?
  - How many messages sent?
  - How many messages read?
  - What are the latency requirements for sender->receiver message delivery?
  - How are you going to store messages?
  - What operations does this data store need to support?
  - What operations is it optimized for?
  - How do you push new messages to clients? Do you push at all, or rely on a pull based model?

Tips for the Design Interviews:

- **Outline your high-level approach:** When you take your approach, outline it then think about how it can be broken down into subparts.
- **Identify your focus:** There's not enough time to discuss every detail of the design, find the interesting and hard problems to focus the discussion on.
- **Navigate the holistic and the details:** Move effortlessly from the goals to the high-level approach to the precise details and back again. A good solution covers both high level ideas as well as low level specifics. Talk about what components you'll use and how they fit together - "Responsibilities will be divided as so between Service A and Service B..." Also describe the implementation details - "A pub-sub queue makes sense here because…"
- **Explore the inherent tradeoffs:** In any engineering problem, you will need to make intelligent decisions about tradeoffs. A good solution compares and contrasts different approaches. It explores the tradeoffs present in any complex engineering problem and it makes intelligent, reasoned decisions about those tradeoffs.
- **Drive the discussion:** Part of the signal the interviewer hopes to gather is whether you've learned how to build large systems through hard experience. Your ability to anticipate and work around typical problems is part of that signal.
- **Make it a conversation:** Be sure to ask clarifying questions. But make sure you drive towards a good solution.

3. The Behavioral Interview - What to Expect:

The behavioral interview is actually part behavioral interview and part coding interview. The behavioral part is about you and your history, your resumé, and your motivation. The purpose of the behavioral interview is to assess whether the candidate will thrive in Facebook's peer-to-peer, minimal-process, unstructured engineering organization.

The coding part is a shorter version of the coding interviews above. We include a coding question in this interview to supplement the two coding interviews and get additional coding signal.

How to Prepare:

- **Know yourself:** Take the time to review your own resume as your interviewer will almost certainly ask about key events in your work history.
- **Motivation and collaboration:** Different interviewers will ask different questions. When I do a behavioral interview, I typically seek two signals in particular:
  - One, what is the candidate's motivation? Why do they want to work at Facebook? Why are they working as a software engineer?
  - Two, how do they collaborate with their peers? How do they resolve conflicts?
- **Have concrete examples or anecdotes:** Support each question with examples. Some typical behavioral interview questions are:
  - What were some of the best things you've built?

- What are you proud of?
- What could you have done better?
- What were some excellent collaborations you've had?
- Tell me about a time when you advocated for and pushed your own ideas forward despite opposition?
- How do you deal with conflict?
- How do you like to give and receive feedback?
- What kinds of technologies are you most excited about?
- Why Facebook?

Tips for the Interview:

- Familiarize yourself with our 5 core values (move fast, be bold, focus on impact, be open, and build social value). This is how we work together to make the world more open and connected. We look for people who believe in these values and practice them daily.
- Be yourself! Be open and honest about your successes and failures.
- Be humble and focus on team work, leadership and mentorship qualities.