Onsite Interview Tips for Facebook

For the on-site, you generally have 4 to 5 interviews – at least 2 strictly focused on coding, 1 on design, and 1 on conversation/coding.

Coding (2)

You will have two interviews that focus very heavily on coding, and will be similar to the initial interviews you had via SKYPE/phone. These do tend to be a bit more challenging, but the engineers are looking for accurate, bug- free, fast, and well-thought-out code. They'll be looking for your thought process, so be sure to provide a narrative as you go through the code. As before, you're welcome to code in whatever language you feel most comfortable, but choosing one that is going to assist in getting an optimal solution in the most speedy and efficient manner is key.

In addition to reviewing the CS fundamentals, these tips may be helpful:
-Understand the problem you have to solve. It is okay to ask for clarifications or to talk through the problem.
-Think about different algorithms and algorithmic techniques (sorting, divide- and-conquer, dynamic programming/memorization, recursion)
-Think about data structures, particularly the ones used most often (Array, Stack/Queue, Hashset/Hashmap/Hashtable/Dictionary, Tree/Binary Tree, Heap, Graph, Bloom Filter, etc.,)
-Sometimes modifying the problem or thinking about it in smaller pieces may be helpful
-Practice coding on a whiteboard

Design (1)

There will be one interview that focuses on design. These interviews focus on systems – think distributed systems and APIs – very focused on building/implementing a structure/product. One example of a question: How would you build a chat system that handles millions of concurrently connected users? Be sure to be very thorough in your explanation, we are generally looking for a boxes and arrows diagram on the whiteboard.

A couple of things to focus on in this interview:
-Communication is key, you will be steering the conversation and it will be up to you to understand the problem and ask clarifying questions
-Our engineers will be focusing on your familiarity with complex systems.

Some topics you should be familiar with:

-Concurrency (threads, deadlock, starvation, consistency, coherence)
-Networking (IPC, TCP/IP)
-Abstraction (understanding how OS, filesystem, and database works)
-Real-world performance (relative performance RAM, disk, your network, SSD)
-Availability and Reliability (durability, understanding how things can fail) -Data storage (RAM vs. durable storage, compression, byte sizes)
-CAP Theorem -byte math *Note that we're not looking for you to be an expert in ALL of these, but you should know enough of them to weigh design considerations and know when to consult an expert*

- For practice:
  -Work with a fellow engineer on mock design sessions.
  -Dig into the implementation and performance of an open source system, understand things like how the system stores data on disk and how it compacts data
  -Be familiar with how databases and operating systems work
  -Practice on a whiteboard

Conversation/Coding (1)

This interview will be with one of our leadership engineers and will consist of a combination of two aspects: a conversation and coding. During the conversation, the engineer will be asking questions about your technical background, what you're looking for at Facebook, and why FB specifically (vs. other companies). Some questions might include: what you've been doing in the past, why FB, what you would improve/change at FB, where you foresee making an impact, etc. The second part of the interview will focus on coding – the engineer will provide a coding question (similar to your other coding interview).

Lunch

There will be a 45-minute lunch to break up your day – the lunch will most likely be with one of the engineers. Feel free to be very candid with the engineer – they will not be providing feedback, but are there to answer any questions/concerns you may not have asked during your interview. Also, take this time to breathe :)