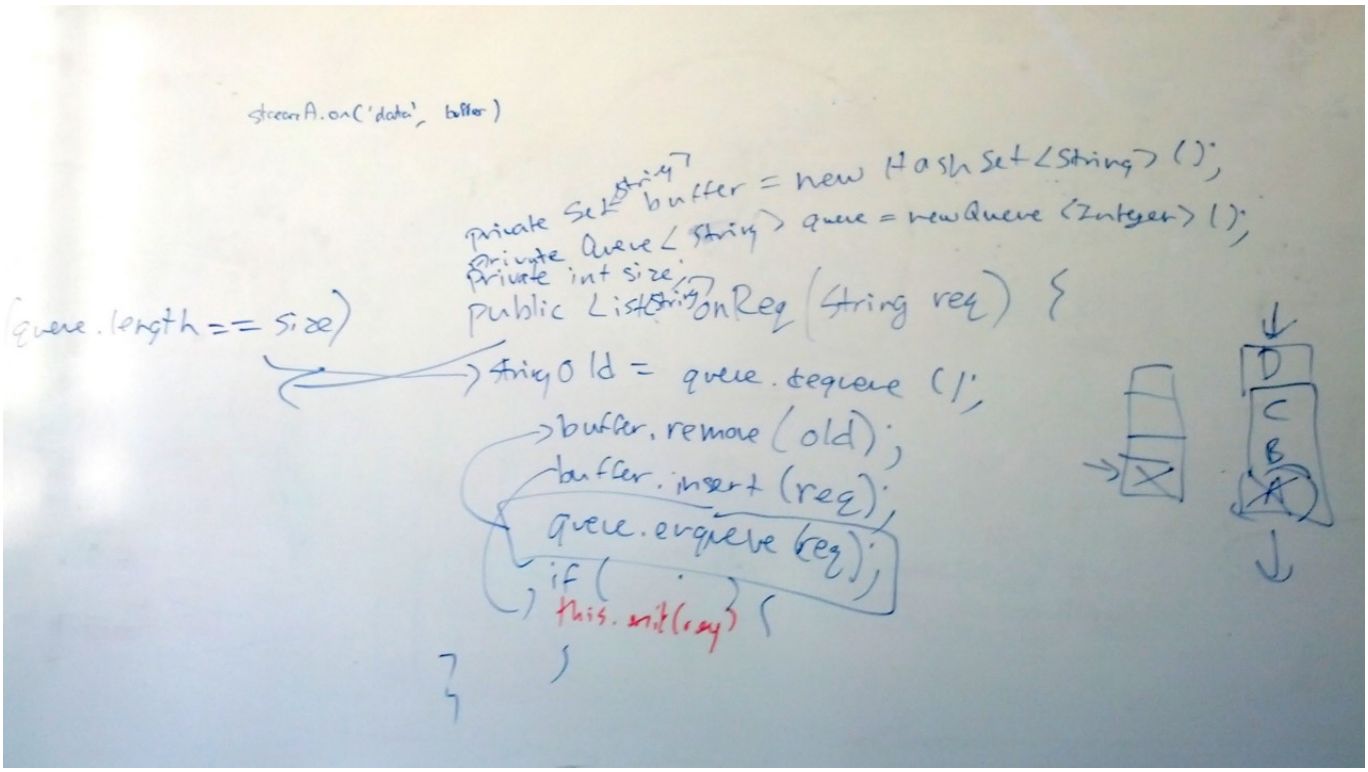
Writing by Dan Pupius

Follow 337 Followers About



Whiteboarding

Dan Pupius Nov 20, 2012 · 4 min read

If you are interviewing for a programming job it is almost inevitable that you will be asked to do some coding on a whiteboard. While there has been some criticism of the practice, it is widely employed by the tech giants — Google, Facebook, Microsoft, Amazon — and startups alike.

In this post I'm not commenting on the validity of this style of interview (though I do personally find whiteboard coding a useful way to see how candidates work through

problems) but instead pointing out some common pitfalls that can be avoided, given that this is how tech companies interview.

Background

To give some context, I worked at Google for 6 years and did just shy of 200 interviews. Google interviews are notoriously code heavy and the hiring committee will often contact interviewers if there aren't code snippets in the feedback.

I've interviewed people who have gone on to make really valuable contributions. I've also interviewed people who didn't have the right skill-set, aptitude, or experience. The hardest type of interview is always somewhere between these two extremes.

Then earlier this year I switched jobs and interviewed at several silicon valley companies and startups, so I had the chance to be on the other side of the process.

Given that tech firms will err on the side of not hiring (wanting to avoid false-positives), you want to stack the odds in your favor and give them no doubts. These points might seem obvious, but they are problems I've seen come up time and time again.

Slow Down

Slow-down, chill-out, and try to write as neatly as possible. This will help you stay organized and focussed and will also helps the interviewer understand whether the code actually does what it is meant to.

Ask questions

Many interview questions are intentionally vague; the interviewer wants to see how you handle uncertainty and what assumptions you make. If you are implementing an algorithm, what are the constraints? Memory, runtime, size of inputs, etc.

As you are answering the question talk through what you are doing and what your thought process is. The interview is as much about the how you get to the answer as the answer itself.

Code how you'd normally code

One of the most common mistakes I saw was candidates getting lost in their answer. Hopefully you wouldn't write a 50-line function in "real" life, so don't do it in an

interview. In many ways it is even more important to break up your code when working on the whiteboard; you don't get to cut-and-paste, and in the heat of an interview it is easy to lose track of state.

Check your work

When you think you are done, check your work.

Most errors will happen at the termination cases. The best way to do this is to take a minute to run through the code with some sample input. It's not unlikely you'll have missed a case or introduced an off-by-one error, your interviewer won't mind you taking time to do this, in fact, they'll probably be impressed.

Also, check how your code handles unusual inputs and edge cases? Think about how you would unit test the code and write them up on the board.

Practice

When preparing for an interview you'll likely be brushing up on algorithms and reading about the latest technologies and best practices. You may as well take an hour and code something up on a whiteboard (or even on paper if you don't have a whiteboard).

Think about how you'd implement a JQuery helper or a method from your favorite language's standard library. Search the web for sample questions—ignoring any site that talks about manhole covers.

When you think you're done, take your code and see if it actually runs. Learn from what didn't work.

Research

Presumably you want to work at the company you are interviewing at, so try and seem interested and enthusiastic about what they do.

You'll undoubtedly get asked about the company's products and "why you want to work here". It *can* be hard to come up with good answers on the spot, so research the company and figure out what you'd say ahead of time.

Have fun

More than anything, try to have fun. Be curious and enjoy the challenge. You'll get more out of process and probably come across better all the more because of it. As well as seeing whether you are capable, the interviewer is also looking to see if they would *want* to work with you.

These things may seem obvious and simple, and it may seem unfair that they have an impact on a candidate's prospects. But it is hard to differentiate between someone who can't code very well at all, and someone who just can't code well on the whiteboard. So do the best to stack the odds in your favor and don't let the whiteboard get in the way.

Engineering Interviewing Technique Help