I've held quite a number of different jobs: competitive swim coach, private school teacher, big 4 accounting consultant, buy side real estate financial analyst, etc.

So why and how did I ever start this road to software engineering? I think that I've always been predisposed, but have just never taken the full leap. My parents tell me that legos were the worlds best babysitter, "We'd put on a movie, bring in the tub of legos, and you'd build stuff for hours, happy as a clam". When I was about 12, around 1999, I ran our swim team's website (HTML in a text editor and ftp'ing files up to the server).

##The Trading Algorithm

In college, I wrote a trading algorithm in VBA pulling data from Yahoo finance. I remember just how "right" the whole thing felt. I described what I wanted the program to simulate, first in english, then in pseudocode, and then finally in VBA. I had never really programmed in my life, but the whole concept made sense. I learned about loops, if statements, functions, variable assignments, data structures and static types. Looking back, it was probably a terrible first language in some sense, but it definitely made picking up python 4 years later seem a lot easier than if I had been totally green.

##The Mock OMPA

I had been coaching competitive swimming full time for 3 years, and I had more free time than most normal people do in the winter months. Our swim league, of 9 teams and 2000 swimmers, had an online database of swim times and ranked swimmers in each event throughout the season. I had an idea: web scrape the data, waterfall it into a mock swim meet, score the meet, publish a mock swim meet to rank the teams like a BCS ranking system. I first put the problem into english, and then converted it to pseudocode. I continued to break the problem down into more modules, and more specific pseudocode. Thankfully, python practically reads like pseudocode! I taught myself python by implementing one module of pseudocode at a time. Each time I got stumped, I consulted the python documentation or stack overflow.

##www.SwimSolo.com

What finally pushed me to make the full leap towards to software engineering was a pet project much more ambitious than any of my previous ideas. "Why isn't there an app that intelligently builds actual swim workouts specific to the user?". The internet was filled with static libraries of pdf swim workouts, written as one size fits all. None of them had intervals aimed at exactly your speed or the exact type of workout that you wanted. My good friend, who was also a swim coach, agreed. We joined forces to make something. He would write the workout templates (1000s of them), and I would build the software that intelligently manipulated and selected the templates to build up a whole workout.

What we ended up with was a web app that does some pretty awesome things. SwimSolo takes user inputs (workout time, swimmer speed, endurance/sprint focus, stroke focus etc), and builds extremely specific and accurate "swim coach quality" workouts. I hacked an algorithm that takes just two sprint times (say your 25 freestyle and your 200 freestyle), and calculates all of your repeat intervals for any type of swim set. Whether its 10 x 100s easy, 20 x 100s anaerobic threshold, 8 x 200s aerobic endurance, the algorithm works! This combined with categorizing of set types combines for a lot of workout variety entirely from software.

The project taught me about: django, mysql, mandrill html emails, dust.js (client side templating), bootstrap (less.js), design (background, colors, logos etc), jQuery, AJAX, git, deploying etc.

In the end we found that the 40+ people who tried our site basically said, "the site looks awesome and has a cool technical concept, but I just don't need this sort of service on a daily basis". It turns out experienced swimmers are pretty good at writing their own workouts, and coaches also prefer to write their own workouts. I have no regrets though. The project was a fun way to learn as much as possible about web development.

## I'm going to Hack Reactor!

After finishing up much of the work on swimsolo.com, I decided that I wanted to learn more and really push the limits of what I can eventually build. I applied to a software engineering school in San Francisco called Hack Reactor. There are so many "learn to code" bootcamps popping up that that it's easy to become cynical on the whole idea of a coding school. However, Hack Reactor is different. They take people who already know some programming and ready them for the rigors of an actual software engineering career. Their program has the most hours of instruction, the best instructor ratio, and the best job placement rate that I could find.

I feel like I've been continuously learning this whole past year, and I love that. I hear software engineering is all about learning new things, and continuously reinventing your skill set. I cant' wait for school to start.