IT'S ONLY CRAZY UNTIL YOU DO IT

HOW I BECAME AN IOS DEVELOPER

BY

SEAN ALLEN

It's Only Crazy Until You Do It:
How I Became an iOS Developer

Special Thanks

First up is my partner on this book, my sister, Amanda Allen. Without her this would have never made the transition from idea to reality.

Next I'd like to thank Paul Hudson for all the guidance on what it takes to publish a book. He saved me from a ton of headaches.

I'd also like to thank my Mom for the last minute proofread. She found 16 errors in the 11th hour. Thanks Mom 😀.

And finally, I'd like to thank all of you. The community around my content has been nothing but supportive, grateful, and encouraging.

Thank you.

I wrote my first line of code at 33.

Let that sink in. Thirty three. After years of an unfulfilling career, and at an age when many people throw up their hands and say, “okay… I guess this is it. This is what I’m doing the rest of my life,” I said “fuck all that,” and went on a journey to find something exciting and fulfilling. I wanted to love going to work everyday.

Let me back up a bit and tell you how I got to that first line of code. My path to becoming a developer was not traditional, and if drawn on a piece of paper, looks more like a maze than a straight line.

I started my adult life in the military, serving in the US Navy on a nuclear ballistic submarine. After my service, I joined the family

business, insurance. Spoiler alert: insurance is not exciting. Sure, I was collecting a decent paycheck, living comfortably and enjoying the perks of a family business, but eventually that wasn't enough for me. So, at 29 years old, I quit the family business, enrolled as an adult student at Penn State University, and set my sights on Silicon Valley. Graduation day came and went, and next thing I knew I was driving cross-country to start what, in hindsight, would be the true beginning of my life.

There's more to the story. I'll get into a lot of the detail in this book, but before we do I want to manage your expectations. This is not a How-To book. This is not a step-by-step guide to becoming a developer. That doesn't exist. Everyone is unique and has different circumstances. We all wish we could go back in time with the knowledge we have now. You've heard the saying, "if I knew then, what I know now…" Well, that's the point of this book. I'm going to recap how I became an iOS developer with no prior coding experience, and provide insights with my current knowledge to help you along your journey.

I hope you find inspiration within these pages. Here I am, 37 years old, thriving in a career that wasn't even a thought, let alone a possibility, just over three years ago. Making this leap was the best decision I ever made. Fast forward to present day, and I've built apps for a few startups here in Silicon Valley and created a reputation within the iOS developer community. Perhaps the coolest thing I've done is meet all of you. Through my YouTube channel, I've met and have been able to help thousands of people trying to

accomplish the same thing I did: start a new career in something they love. What I've found is that so many people are scared to start something new or are intimidated by the world of software development. Some have taken the leap but are looking for help navigating the trials and tribulations of getting that first iOS developer job.

That's what this book is all about. I don't care if you're a kid in high school, a 40-something bored with the status quo, or whatever your story is – anyone can become a software developer. It only sounds crazy until you do it.

Chapter 1: The Labyrinth

I think it's important you hear a quick version of my backstory to provide the context necessary to understand how and why I made the decision to become an iOS developer and what led me to Silicon Valley.

After high school, I wasn't exactly sure what I wanted to do next. And in the absence of a set plan, I chose to join the US Navy. It runs in the family and my first day in the Navy was the same day my dad officially retired from it. I spent four years, from ages 18 to 22, working on a nuclear ballistic submarine. It's not as badass as it might sound; I spent a lot of time listening to whales while manning the sonar station and played a ton of video games and Texas Hold'Em.

I served on the USS West Virginia SSBN 736

Shortly before I enlisted in the Navy, my parents opened their own insurance agency in my hometown. So after my four years of service, I decided to join the family business. Looking back, I did that for all the wrong reasons. The money was good. The hours, even better. And the workload? Minimal. For an immature kid in his early twenties, it was kind of a dream job. *Until it wasn't.*

Pretty soon I was bored. I was going through the motions and not even giving it half of my effort. Around the time I turned 28, I finally realized that money – the main motivator for taking the job in the first place – wasn't worth slogging through mind-numbing work each and every day. It's a little sad to say that it took until I was 28, but better late than never, right? My priorities shifted. They say if you do what you love, you'll never work a day in your life. Well, I was ready to not work another day in my life.

The first step was going back to school. In exchange for four years of service the US Navy paid for me to get a college degree as part of the Montgomery GI Bill. If you've served in the US military and have not taken advantage of this, do it. Do it now. It's an amazing deal. While my experience may have been unique, that program ended up paying for my schooling and living expenses at the time. School and rent paid for? Yes, please. It was awesome. And it was critical in getting me to where I am now.

I was always an early adopter, and at the risk of sounding cocky I have a knack for experiencing a new technology, and realizing "yeah, this is going to be big". I remember getting made fun of for having an iPhone in early 2008. "Why do you need all that in a phone?!? That's way too expensive." I've always just said, "you'll see," and way more often than not I've been historically correct. And that's what got me thinking about Silicon Valley: if I'm digging deep, searching the soul for what I want to do with the rest of my life, I want to be involved in creating the future.

My first. The iPhone 3G

That's when it clicked. I wanted to be in the world of the Apples, the Googles, the Twitters, but I wanted to get in on the ground floor: startups. That's where I was going to find the passion I had been lacking. I wanted to get involved in the early stage funding process of these companies. Joining them as a software engineer never even crossed my mind. In my head, I had the same stereotype many of you may have had – I thought it was way too hard and that I could never do it. After all, I was almost 30, my time had passed to do something like that. So I set my sights on getting into the world of angel investing and venture capital. If you want to become an actor, moving to Los Angeles is a wise move. For that same reason, I needed to be in San Francisco.

I'm not sure if you've heard the rumor, but San Francisco is expensive. Like, crazy expensive. So before I could get there, I needed to save some money. That meant sacrifices. I moved in with my parents for a few months to cut down on expenses, I declined invitations to go out partying, and just really focused on saving my money for what really mattered. Eventually I was able to save up six months' worth of living expenses. *San Francisco* living expenses – that shit's no joke. But, that was all the runway I needed. I have a unique kind of confidence. There's confident, there's cocky, there's arrogant. Then there's me. I was so confident in my abilities to work my ass off and make something happen, that as soon as I had the money saved, I was bound for San Francisco. I didn't know a soul out there, knew nothing about the area, and didn't have a job lined up. I was going anyway. I knew what I wanted and I knew it was the best place for me to be to make it happen.

My job hunt started about a week after I arrived and had settled into San Francisco. I knew I wanted to get into angel investing and venture capital firms, but I quickly realized I didn't *really* know enough about startups. Reading TechCrunch, watching Shark Tank and HBO's Silicon Valley doesn't really give you the full picture. To really understand that life, I had to live it. So, I hopped on angelist.co and starting applying to small startups. About a month later, a small start-up called Playpass took a chance on me.

Due to my experience in sales, I got hired as the Growth Lead for the company. However, because of the nature of start-ups and the fact that it was only 5 people, I had to be much more than a Growth

Lead. You don't have just one job at a startup, there's a closet full of hats you have to put on each day. But the focus of my work was to get people onto our platform. Playpass was a service for recreational sports leagues, a place they could go to manage their league, from scheduling and taking payments, to posting the league standings. At the time, I was personally playing in a few recreational basketball leagues, so I was familiar with the need for this kind of service.

My desk at my first startup, Playpass

For the first five months at Playpass, I was obsessed with my job of growing the user base for the company. I completely immersed myself in reading articles on the internet, blogs, whatever I could get my hands on to better understand growth strategies. I was hooked.

One of the ways I attracted new users was utilizing Google SEO (Search Engine Optimization) and SEM (Search Engine Marketing) tactics to drive users to various landing pages based on what they were searching for. Believe it or not, developers don't get excited about creating a bunch of simple static landing pages (*who knew?*), so I didn't want to ask them to do it. They were busy building the core functionality of the company, so I didn't want to waste their time. The only way these pages were going to get made was for me to do it myself. I played around with HTML/CSS at first, researching and experimenting, so that I could be self-sufficient. That's when it happened. I caught the bug. Suddenly all I wanted to do was code and build things. For the first few months on the job, my spare time was spent researching growth strategies, but after I started playing around with the landing pages, that stopped. I couldn't stop thinking about writing code. Suddenly my time and energy was consumed by learning how to become a developer.

I continued working at Playpass while playing around with coding at home. Self-guided learning was cool, but I wanted the inside scoop too. I realized I'd be crazy if I didn't take advantage of the resources available to me at work. There I was, spending the majority of my time among developers. What I learned during this time was that developers love to talk about their work. It's the opposite of Fight Club. So I struck up conversations, "tell me what you do," "how do you do it?", "what are the challenges?" I also got a chance to sit in with company leadership during interviews for new developers. I was there for culture fit, not technical questions. This interesting because I got to know what companies look for in a developer, questions they ask, and hear the team's honest feedback

on what makes a candidate a fit. Oh, I also learned what a company is willing to offer in order to get high-caliber developers into their company. Hint: it's a lot. Going back to my earlier comments about confidence, after some of these discussions and interviews I started to realize, "I'm just as smart as these people, I can do this." From that point on, it wasn't even a question: I was going to become a developer.

Around this time, Apple made a huge announcement, and the timing could not have been more perfect: Swift was here, and everyone was a beginner. As I did when I first started learning, I completely immersed myself in all things Swift and iOS. I got in on the ground floor, Swift 1.0.

As my interest in software development grew, my time at Playpass was nearing it's end. The catalyst for me to quit was a crushing blow delivered to me by what I thought was a slam dunk client (cheesy pun intended). I've said it before, and I'll probably mention it again many more times, my extracurricular passion is basketball. I play as much as I can, at the YMCA, in tournaments around San Francisco, you name it. The tournaments I was playing in brought me to JamTown, a local gym and host of basketball tournaments. Playpass

was experimenting with a service that would go out and find players to fill these leagues/tournaments. JamTown was the perfect client for this and I saw a massive opportunity in front of me.

I was amped. Landing a client like JamTown was what Playpass desperately needed at the time. Soon, JamTown was brought on to our service and I began scouring the internet for players to bring to the gym. Because of my time playing basketball around the city, I had a large network of basketball contacts and therefore had a bit of a leg up on this. I tapped into that network and got 20 guys to sign up for the league. It was pumped. I spent hours cold-calling in order to do this, and the hard work paid off. But do you wanna know what happened? The league folded and I had to call each and every one of those 20 players to let them know we'd be refunding their money. It was embarrassing. It didn't look good for Playpass or myself.

The JamTown letdown was the catalyst for me leaving Playpass. It wasn't so much that I lost the account – that's a fact of life in any sales job.. The problem was that despite busting my ass, putting my name out there to bring guys in, and doing everything right by the client, it all fell through. It was totally out of my hands. And that's what really bothered me. I realized then I didn't want to do it anymore, and that sales was not for me. I wanted something that was one-on-one, me against the problem. I wanted something where I could turn out a product and be judged on my skills alone, not outside, uncontrollable factors. Coding was it for me.

I left Playpass to pursue becoming an iOS developer full time. A famous saying in the world of startups is that companies jump off a cliff and try to build the plane on the way down. Some do… some don't, and crash and burn. That's what I was doing with my life: time to become a developer or crash and burn.

Chapter 2: Why a bootcamp?

So here I am, unemployed and burning through savings. How do I become an iOS developer as fast as possible? I had a good amount of savings, but almost no knowledge of the industry, so there was really only one answer: a bootcamp. I applied to an iOS-specific bootcamp called Mobile Makers Academy based in San Francisco. More on that later, but now it's insight time. Looking back, would I do a bootcamp again? *Maybe*. Just the kind of concrete advice you were looking for, right?

Here's the thing: knowing what I know now, a bootcamp doesn't teach you anything out of the ordinary. Just the basics. The same basics that can easily be found online for free at a certain YouTube channel (👋) as well as other sites. If you understand basic programming concepts like arrays and for loops, as well as how to

create a UITableView, basic networking, passing data between UIViewControllers, and delegation… then it's probably not worth the upwards of $10,000 for a bootcamp for the knowledge.

However, it's not all about the course material. Accountability is a big deal. Sure, you can *say* you're going to self-study for eight hours a day, but we all know *very* few people would actually keep that up for eight weeks straight. Attending a bootcamp forces you to do this, accelerating your learning. So if you're on a time crunch like I was, then a bootcamp is a good way to go.

You'll also learn how to play nice with others. As a developer, working well within a team is vital. You'll plan out your projects, assign tasks, review pull-requests, handle merge conflicts, etc… It can be very difficult to get this valuable experience while self-studying, but it's a big part of the value in a bootcamp.

Lastly, a common question: did the fact that you were a bootcamp graduate carry any weight in landing your first job? Yes, absolutely. There's a big difference between "I've been studying on my own for eight weeks" and "I graduated from a bootcamp." From the employers perspective, they have no idea what kind of self-studying you've been doing. However, if you've graduated an intense eight week bootcamp, they know exactly what that entails and it signals commitment because they know you paid a pretty penny to do it.

So here's my verdict on if a bootcamp is right for you: if you're extremely committed, have the willpower to study on your own for eight hours a day, have confidence that you can create a good portfolio of apps, and believe that you can market yourself well to potential employers… then I wouldn't recommend going to a bootcamp. You can do it on your own. However, if you can reasonably afford it, you need that extra push, and you want to get the valuable team experience, then I think you'd benefit from a bootcamp.

Chapter 3: The Vacations

April 2015 was an exciting time. Swift was just a baby, I quit my first job in San Francisco, I got into Mobile Makers Academy, and the Golden State Warriors kicked off the franchise's most successful Playoff run in 40 years.

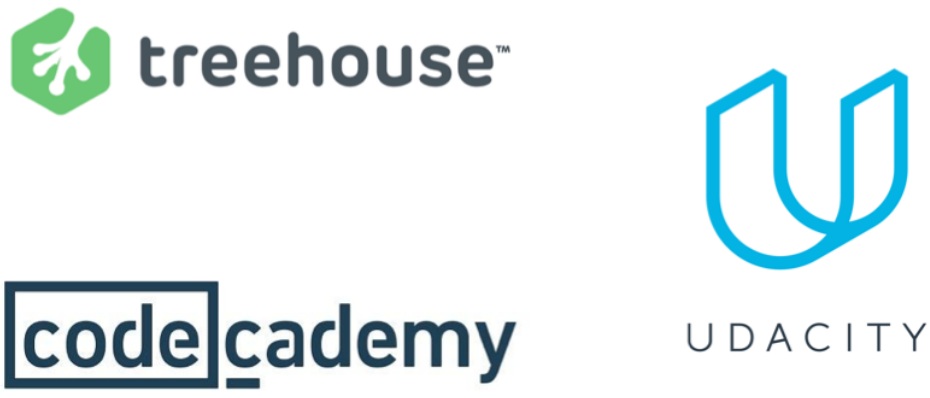
I didn't start Mobile Makers Academy until mid July, 2015 – three months after quitting my job. As I just mentioned, I quit my job at the beginning of April. If you do that math, that doesn't add up. Why quit in April?

The next cohort for the bootcamp began in May. Perfect, right? Nope. A family vacation that had been planned for almost two years was scheduled for June. Right in the middle of the cohort. My

family was celebrating my grandparents' 50th anniversary by going on an Alaskan cruise. In an eight week bootcamp, you can't exactly take two weeks off in the middle of it. So, I had to wait till the next cohort which began in July. No big deal. That'll give me an extra couple of months to study and prepare, right? Kind of.

Remember what I said about how *very* few people will self-study for eight hours a day for eight weeks straight? Yeah, that's what I told myself I would do as I prepared for the bootcamp. That didn't happen. Even though I was unemployed and had no good reason not to study my ass off, distractions got the better of me. That first Warriors championship run was a great time. The city of San Francisco was in a frenzy. Watch parties for every game. Looking back… it was actually pretty cool to be here for that. But it killed my studying. I only studied for two to three hours a day, if that. I was on an extended vacation that would come back to bite me in the ass.

What materials did I use to study during this time? I'm glad you asked. Swift was still very much a baby at this time. Swift's first major update, 1.2 had just been released. That meant that there was very little educational material available. Everyone was still learning. Even the material that was available wasn't fully fleshed out.



The sites I used at the time were Codecademy, Treehouse, and the Udacity Nanodegree. Again, Swift was new and these were the very first iteration of these courses. For that reason, it wouldn't be right for me to pass judgement on how those courses were. I'm sure each of their Swift material has come a *long* way since then.

But, I do want to talk about a trap that many, myself included, fall into. I call it the "paint-by-numbers" trap when following tutorials. When doing that you typically do exactly what they say, type what they type step-by-step, and end up with a working app. While you learn the basics by doing this, you don't really learn and retain the information. Have you ever driven somewhere using your GPS, only to realize that without it, you'd have *no idea* how to get to where you just arrived? It's like that. You got from point A to point B, but good luck repeating it on your own.

I say this only to caution you from getting a false sense of confidence. It happened to me. I would follow these tutorials, end up with a working app, and be like "Yup, I'm a developer now. I'm

awesome." As with anything in life, you're going to learn more by actually doing. It took me a long time to realize this. I wish I would have realized it before I started my bootcamp, but I didn't.

I was in for a rude awakening.

This chapter is a reformatted blog post I wrote the week after I graduated Mobile Makers Academy in September of 2015.

https://seanallen.co/posts/mobile-makers-review

Chapter 4: Mobile Makers Academy

Over the past two months I completed one of the hardest things I've ever done. I graduated from Mobile Maker Academy, which is an intense and immersive eight week bootcamp that teaches iOS development. I had been the Growth Lead for a small startup that developed and released an app, and after working closely with the engineers during that process, I was hooked.

Not only is it an amazing feeling to create something people use everyday, but the profession of an iOS developer is incredibly flexible. You can work for a large company, join a small start up, do some consulting and/or work on side projects. Combine that with the fact that there is always some cool new technology to learn (tvOS anyone?), I can't imagine ever being bored in this line of work. So, I chose to take some time off, spend a nice chunk of

money, and attended Mobile Makers Academy to accelerate the transition to becoming an iOS developer.

Aside from a little HTML/CSS and some online tutorials for iOS, I came into this bootcamp with no prior programming knowledge. It was a long, arduous journey but extremely rewarding at the same time. At first I felt completely lost and overwhelmed, but I found my groove midway through and finished strong. I ended up working with a great team on an app that I'm extremely proud of.

Some words of wisdom for anyone taking the plunge into Mobile Makers Academy or a similar bootcamp: Don't plan anything during this time so you can focus all your energy on the coursework, and kiss your nights and weekends goodbye.

What to expect at Mobile Makers Academy

Let's talk about cohort dynamics. A cohort is the group of people taking this bootcamp with you, and each one has a different

dynamic. Some have 20+ students and some are smaller, like mine, which only had eight. The dynamic of the cohort plays a major role in the overall experience. I can't speak to the advantages of a larger cohort, but in a smaller cohort you'll get a lot of one-on-one time with the instructors, which is invaluable.

My cohort spanned a wide age range. We had students getting ready to start college, students fresh out of college, and students in their mid-late 20's and early 30's that are changing careers. As you can imagine the 19 year-old who just finished high school and the 33 year-old changing careers have slightly different expectations and priorities when it comes to this class. There were students who wanted to learn and write code the correct way, and others that just wanted to get done quickly and write their code in a hacky, fragile way. Needless to say, those two styles clash when working together on a project.

Pay attention to the motivations of the people in your cohort. It's an interesting dynamic to navigate, but if you do it well you'll be much better off.

Weeks 1 through 4: What did I get myself into?

During the first week, I remember saying to myself many times, "I've never felt dumber". I'm a pretty smart guy... (humble too), but I was overwhelmed. It didn't help that five of the eight students had

a Computer Science degree or prior programming experience. They had quite the head start on me and I felt very behind from day one. I jumped in the deep end and had to learn to swim.

Every day of the first four weeks is structured the same. In the mornings we would receive a lesson on specific topics such as UITableViews, Core Data, delegation, networking, MapKit, etc… In the afternoons, we would build a small app based on the morning's topic.

We worked in pairs on the afternoon projects, and every Friday morning there is a solo assessment to test your abilities. That's pretty much the gist of the first half of the bootcamp.

Day 1

The morning was just like any other orientation. Introductions, overview of the course, Q&A, etc. After the orientation phase, we did a brief lesson on basic storyboard stuff and IBAction/IBOutlets. The afternoon challenge was to build a basic calculator that multiplied two numbers. It was pretty simple and what was expected for a first day challenge. Then we got to the "stretch". Which simply said "build a calculator". Doesn't sound too bad, right?

Let me take a second a talk about stretches. Each day's app had roughly 10 steps required to complete. After that were a couple stretches, which are steps designed to push our limits as developers. They included more advanced topics we haven't been taught and encourage "self-learning," (a.k.a. Googling the shit out of it).

Most were opened ended, like the one from day one that simply stated "build a calculator." How far do you take that? Do you completely duplicate the iOS Calculator on an iPhone? Have you seen that thing in landscape mode?!

That's what made the stretches great: they really separated the pack. Some didn't bother to do the stretches and some stayed extra late to put out their best possible work. After just a day or two you can easily point out the strongest and most dedicated students. I pinpointed some ideal teammates for the final project by day two.

After building our calculator app I remember thinking how much we take that thing for granted. It was surprisingly complex (for newbie developers). I'll never look at that thing the same again.

Rest of Week 1

The first thing you do each morning is a code review of the previous day's challenge app. It's basically a show-and-tell of your homework, and you discuss areas that you got hung up on. Depending on how the previous day went, you were either super

proud to show off your work, or embarrassed and dreading the code review. I certainly experienced both, although fortunately the proud days outweighed the embarrassed days by a good amount.

Day 1 – challenging, but doable.
Day 2 – a little more rough.
Day 3 – reality punched me in the face.

Academic settings have always come easy to me, so I came into this bootcamp expecting to be one of the top students. That all went out the window by day three. Because I was starting out so far behind the top students with CS degrees and prior experience, my priority was to just survive and improve my skills each week. Consider myself humbled.

Day four wasn't any better and I felt like I was getting further behind. I absolutely hated that feeling, so I subscribed to Ray Wenderlich and went through his tutorials any chance I could. This bolstered my learning and reinforced the basics. If you're getting into iOS development I highly recommend subscribing. I can't imagine a better way to spend $19/month as a junior iOS developer. I live on that site.

The Thursday challenge app is special because it's the weekend challenge. It's extra difficult, but you get 4 days to work on it (Thursday – Sunday). For this reason, having a great partner for the

weekend challenge is crucial. The first weekend challenge was to make a Tic-Tac-Toe game. Sounds easy enough…

Finishing the basic required steps actually was pretty easy. But then you get to those damn stretches… The main stretch was to create an AI (Artificial Intelligence) opponent that a human can play against.

Sure, you could just have the AI select a random available space on the board and call it a day. But again, how far does your group go with the stretches? We decided to create an AI that would be smart enough to be challenging, but not smart enough that you could never defeat it. Thinking through the priority of the logic the AI would perform wasn't too difficult:

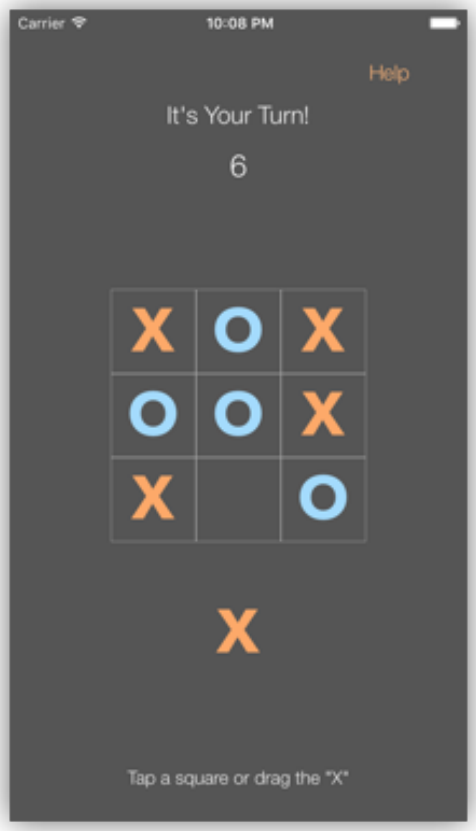
If AI can win, then win
If AI can't win, then block
If AI can't block, then take middle square
If AI can't take middle square, take random available square

This is an intermediate AI logic. You can create a perfect AI for Tic-Tac-Toe so that you'd never defeat it, but that's not very fun for the user. So, taking a random available square in the final step allows for mistakes to be made, and the AI to lose from time to time.

The logic is easy to follow. However, executing it in code is another story and that took us all weekend, but we did it. We were proud of

our Tic-Tac-Toe app and excited to show it off at Monday's code review.

It's fun to look back at that first week and reflect on how far I've come. It was touch and go for a bit there, but after the first week things started to click and it got better.



The AI is tough to beat!

Weekly Assessments

Every Friday morning we built an app as a test to see if we could execute what we learned during the week. Every other app you build is with a partner or small team. These weekly assessments are 100% solo so it's all on you to create the app. The first "practice"

assessment was a piece of cake, and not at all a reflection of what the other assessments would be like.

Based on this, I didn't take the second assessment as seriously as I should have and ended up with a 58%. Yikes. I had to resubmit the test with my corrections to get a passing grade. It was a rude awakening and I felt horrible. I don't think I've ever received a grade that low. That wasn't going to happen again, and I made sure to show improvement on each test going forward. My remaining three test scores were a 76%, 78% and finally, a 94% on the last (and hardest) one which covered Core Data.

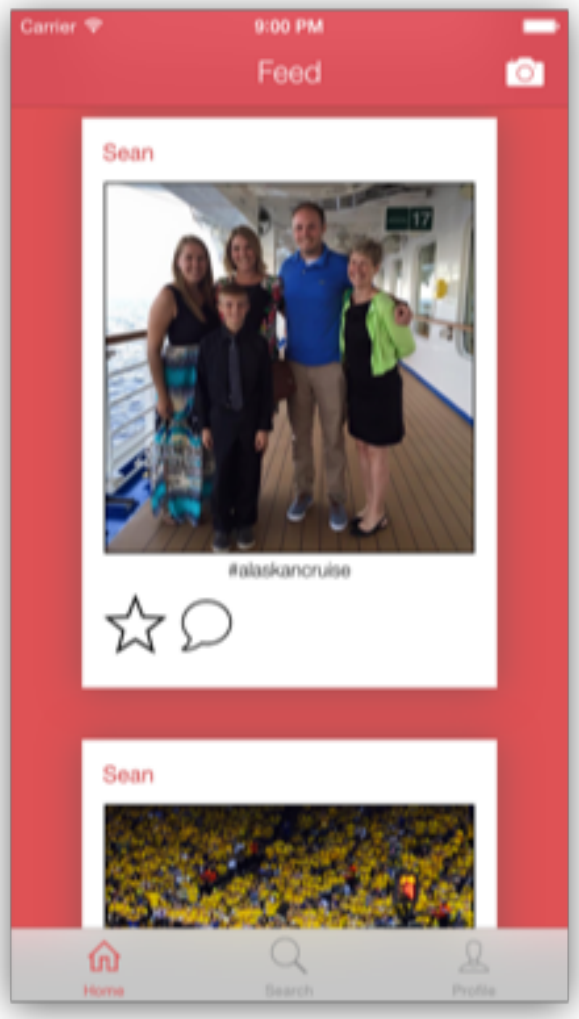
My assessment scores were a pretty accurate reflection of my learning experience at Mobile Makers. I started off completely lost and behind, things start to click in the middle, and by the end I felt pretty good about where I was at.

Week 5: Fakestagram

Week five is where the shift in the bootcamp begins. In the first four weeks we created simple apps, worked on them for a day and rotated partners. Rinse and repeat. Now, we formed a group of three and were tasked with recreating Instagram, and we had an entire week to do it. We called ours Fakestagram, because we're awesome at names. By this point in the course, I had my final project group

locked up and we worked together on Fakestagram as a trial run for the final project.

In Week 5 we touched on Swift for the first time. In the mornings we learned about Swift and the afternoons were spent working on Fakestagram. For only having a week to work on it, I think our Instagram clone turned out OK.



Week 6 through 8: The Fun Stuff

The final three weeks are where you build your portfolio showcase app. It can be whatever you want, and this is the time to show off what you can do. As soon as my partners and I came up with the

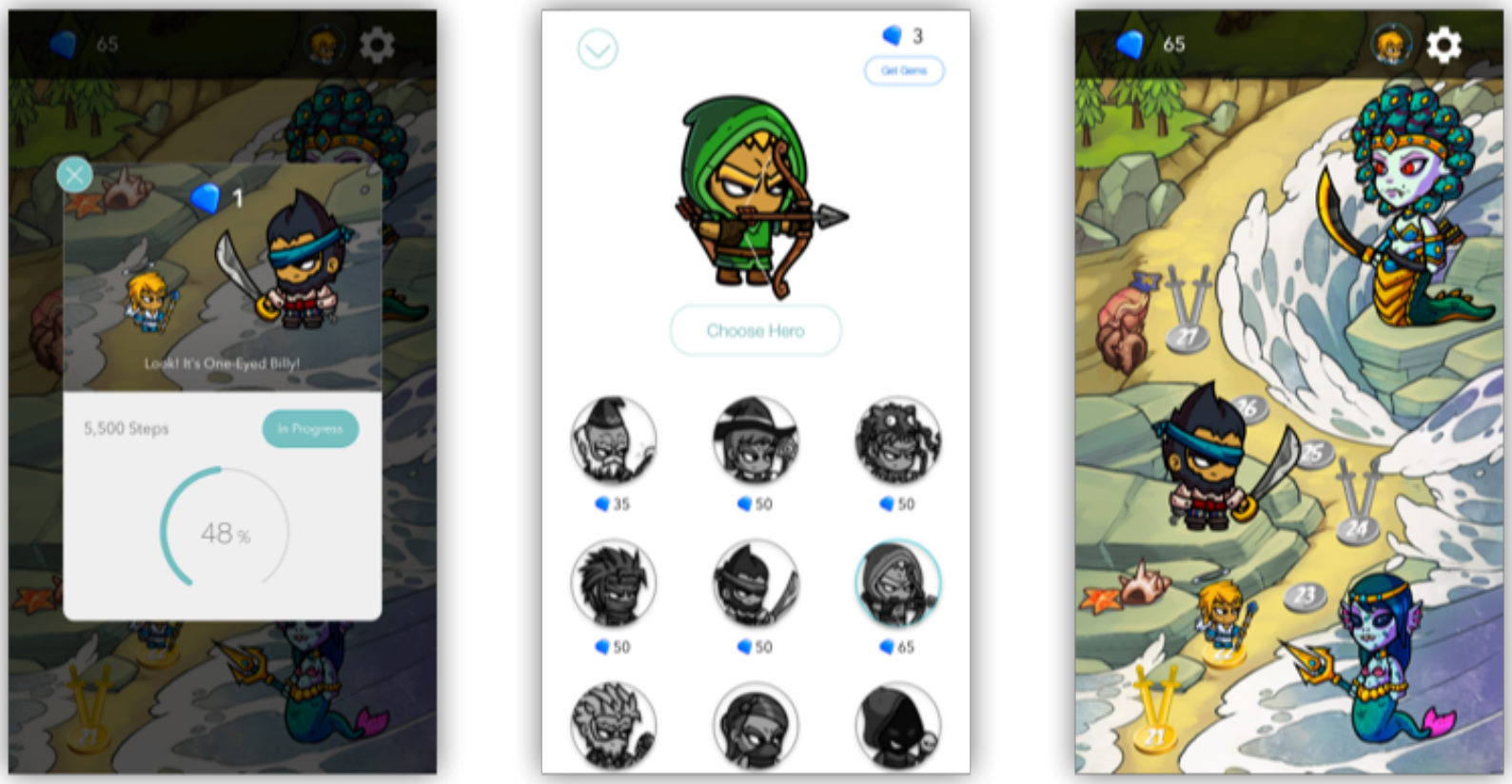
idea for FitHero, an app that makes a game of your daily step count, we were ecstatic. We loved the idea and couldn't wait to spend the next three weeks working on it. In fact, it wasn't even work – it was a blast.

There are no more official lessons. All day, every day you work on your app. During this time, we got experience working as a development team, using version control (GitHub) and agile development techniques. Even though this part of the bootcamp felt the easiest, I worked the hardest by far. 14-hour days, seven days a week was the norm during these final three weeks. That sounds rough, but when you're passionate about the project, time just slips away.

My team worked great together and our skillsets complemented each other very well. From the moment we came up with FitHero we looked at it is as a product we were going to ship rather than just a project to showcase in our portfolios. For that reason, we decided to split the work, focusing on what each team member was best at. My two partners focused on the back-end code and database (Parse), while I focused on the design, storyboard, user interface and user experience.

Here's a little bit about the app. FitHero gamified your Fitbit or Apple Health step counts, making staying active and healthy a lot more fun. A user completes a wide variety of quests such as walking 10,000 steps in a day, walk 5,000 steps in two hours, or walk 30,000

steps in 48 hours to progress on their journey. A user is rewarded for completing quests and defeating bosses by earning gems that allow them to purchase new heroes to play as. FitHero is a simple game that motivates people to get up and get moving.



Be a FitHero!

FitHero's backend was built on Parse, a service that simplified your backend database, similar to Firebase. Parse was acquired by Facebook, and then shutdown about a year later. Because we were busy with our separate careers, the FitHero team made the tough decision to not rebuild/migrate the entire backend for FitHero. Sadly, it is no longer on the app store.

I enjoyed my time at Mobile Makers, met some great people, and learned more in eight weeks than I ever thought I could. The funny

thing is, even though I learned a ton, above all it made me painfully aware of how much I didn't know. My eyes were opened to the incredible diversity of what can be learned in iOS development. My journey was just beginning and I felt confident I've gained the knowledge and tools to continue learning and to contribute to an iOS development team.

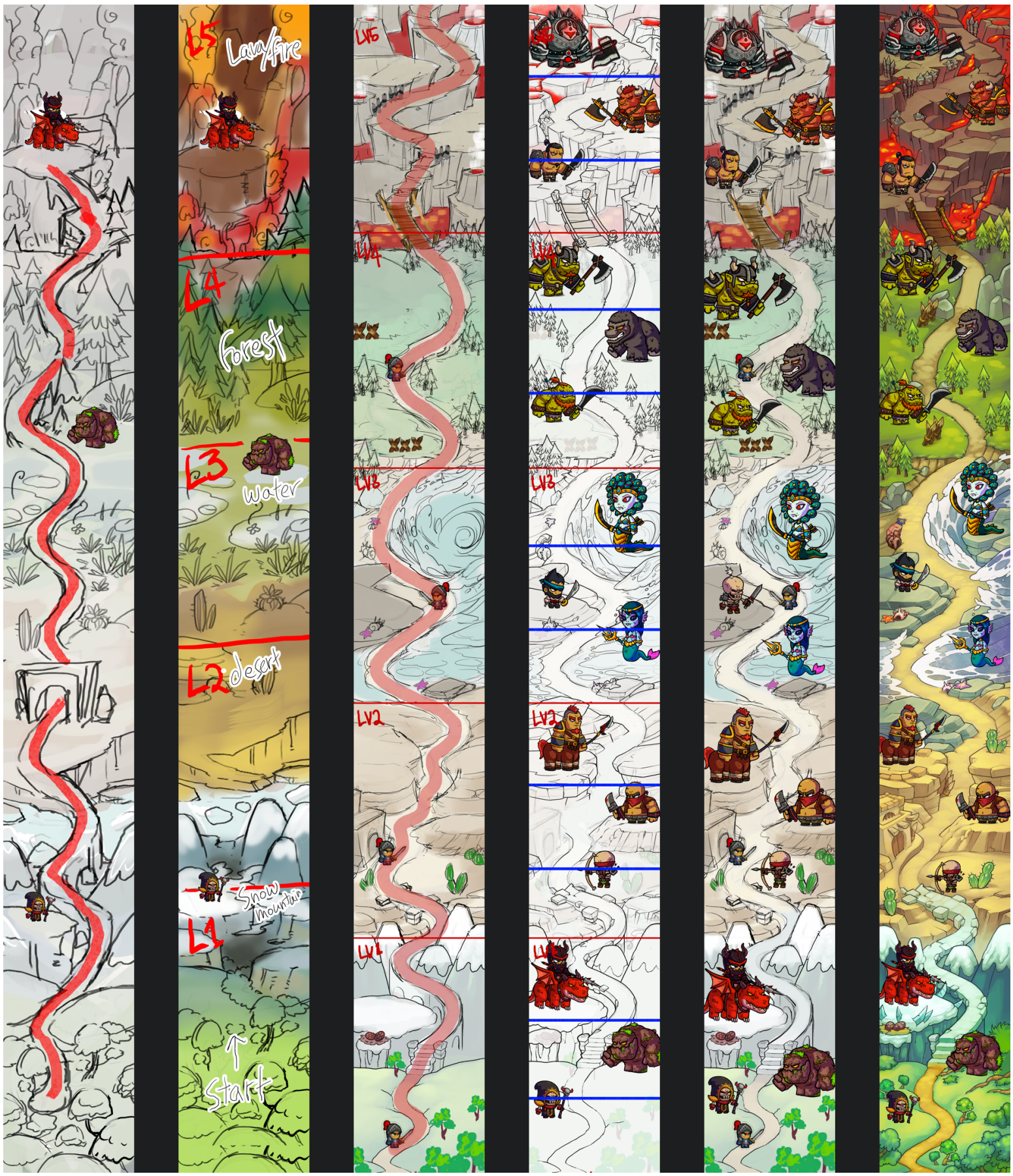
Mobile Makers Academy is no longer an iOS bootcamp available for the general public. They have since changed their business model to focus on teaching iOS development to high school students.

Chapter 5: FitHero

Since we only had 3 weeks to create FitHero from scratch, the app wasn't quite where we wanted it to be at the end of the bootcamp. My team and I decided to continue working together after graduation to get it there. We spent the next month adding features and polishing FitHero to a point we were proud of. This was going to be our showcase during the upcoming job hunt.

Let's talk more about my role in the creation of my first app. As a team of three, we split the responsibility based on our strengths. I focused on the designing and building the user interface and user experience (UI/UX), while my teammates focused on building the functionality of the app. Looking back, I regret this. I'm super interested in UI/UX so that came naturally. But it didn't challenge me. If I could do FitHero all over again, I would have worked on the functionality of the app and I would have come out of the bootcamp a much better developer.

If you're going to stick with your strength, it better be good. So let's talk about that design. I'm super proud of it. I show it off to this day. I had the vision for what it should look like and the illustrator we hired absolutely crushed it! Check out the progression of the design below. Yeah, she did that in four days.



FitHero is a sad story for me. I still love the idea. Especially now, with the launch of the Apple Watch Series 4. There's a market out there for the gamification of health and wellness. I've thought about making my own version of this app, but I'm prioritizing other projects at the moment. Someday. Someday I'll come back home.

I continue to be incredibly proud of FitHero. While it may have been short-lived, it's been impactful on my career. Often times I'll pull it up on my phone as a reminder of just how far I've come. I was inspired then, I was passionate then, and in moments of doubt tapping into that feeling again can help propel you forward. Keep that in mind as you go through this journey. You were bright-eyed and bushy-tailed when you started. Always keep that first success around as a reminder of why you're doing this.

RIP FitHero (for now).

Chapter 6: The Job Hunt

One thing to remember is software is never done. But while that's true, it was time to put FitHero aside and start looking for a job; time to put to test everything I'd learned and taught myself over the past few months.

I started my job search by casting a wide net. I applied to sixty – *sixty* – companies during this time. I wasn't confident in my skills, so I limited my search to mostly smaller companies, but also included some heavy hitters, like Uber, Twitter, and Airbnb. I figured, why not? As expected, I did not hear anything from those larger companies, but that's fine. I had to shoot my shot. Shooters shoot.

Impeccable form.

The site I would recommend most for developers looking to get their first job is AngelList (angel.co). This site specializes in job postings for smaller startups. I actually got my job at Playpass using AngelList, so I knew it was a good place to start. I also knew that the kind of company using this site would also be the kind of company willing to take a chance on someone like me. And so I began my search. For each of those sixty companies I applied to, I sent personalized messages. Just a flurry of them. It was like a full-time job. And from all the effort I got…three callbacks. Sixty personalized messages and applications resulted in three phone interviews. But hey, that's why you cast the net wide.

For these three companies I didn't even make it past the initial phone interview. Yeah. It was bad. It's funny to look back on now, because that initial phone screen is a gimme these days. But it's a good reminder that it wasn't always like that. If this situation sounds familiar… keep going. You'll breakthrough. At this point, things were looking pretty bleak, and I was starting to question whether this major life change was the right decision. But then a company randomly reached out to me: Breathometer.

As I'm sure some of you may have experienced, companies will mass email a list of potential candidates. Like Tinder – they swipe right on everything and hope one works out… or so I hear. When I first got the email that would change my life, my initial reaction was that it was spam. But then I read it: "we're interested in your diverse background." That, to me, said this was not spam. We've already talked about my background. It *is* diverse, so I thought this email was something different. You can imagine my excitement, especially after the initial job hunt was a dead end. So I responded to their email. Funnily enough, I later learned that this was, in fact, a mass copy/paste email to many candidates, but whatever – I got my opening and sprinted through it.

Before I moved out to San Francisco, I was obsessed with the show *Shark Tank*. For those of you not familiar, *Shark Tank* is a show focused on entrepreneurs pitching their ideas to well-known venture capitalists, in hopes of securing the funds needed to take their company to the next level. The whole reason I initially wanted to move to San Francisco was to get into the venture capital world, so I

was obsessed with this show. Because I was such a fan, I recognized the name Breathometer. The company was on the fifth season of *Shark Tank* and was the first company to get all five "Sharks" to invest. Those investments totaled one million dollars. I watched this episode as it aired, so it was kind of crazy interviewing with them – never mind the fact they were the only company interested at this point.

Shark Tank, season 5, episode 2, ABC, 27 Sep. 2013

After responding to Breathometer's email, I set up an initial phone interview with the VP of Software. The call went *really* well and I moved on to the next round, the technical phone screen. I had now passed my first initial phone screen. Again, that seems silly now, but we can't forget these milestones. I'm moving on to my first ever technical interview of any kind. I have no idea what to expect. This is about to be a shit show.

Having been through tons of technical phone screens at this point, I can honestly look back and say this was one of the easier ones. In a typical technical phone screen, you answer some iOS "warm up" questions. Things like explain Automatic Reference Counting or explain how optionals work. Basic stuff. Then, with the remaining 30-45 minutes, they'll present you with a random coding problem that you'll solve while walking the interviewer through your thought process. If that sounds stressful, that's because it is. It sucks and I hate them. Luckily…This was not that type of phone screen. I would have had zero chance.

Another type of phone screen, one that is much more preferred, is of the code review variety. This is where they ask you to pull up one of your projects and walk them through your code and discuss. Not just specific coding questions, but also why you chose certain design patterns or feature implementations. They want to hear how you think about solving problems. That's what this was. I had a chance.

Here's the downside. The interviewer specifically requested code that I wrote 100% by myself. And sadly, I didn't have much. Remember that "paint by numbers" tutorial trap I talked about earlier? Yeah, most of the projects I had were of that variety, so I couldn't claim it as *my* code. But I have FitHero, right? This is where focusing on design and UI bit me in the ass. The UI was primarily built using storyboards so the only code I had to show was a couple of animations. I hadn't worked on any of the game's logic or functionality. As you can imagine, this did not go well. I fumbled my way through the interview and showed what I could, but it was

really basic stuff. They weren't impressed and I could tell. It was awkward.

This interview took place on October 20, 2015. Three days had passed and I still hadn't heard anything. I later found out that, after this technical interview debacle, I was immediately dumped into the "No" pile.

Now, pay attention: the story I'm about to tell is perhaps the most important message in this entire book.

Initiative matters. The whole reason you're reading this book is because you're about to, or have recently, taken a leap. You have a passion and you want to follow it. Well, what good is passion without the gumption to make it happen?

Despite not hearing anything from Breathometer after a few days, and knowing in my gut I didn't pass the interview, I wasn't giving up. The fact that this was the only lead I had helped with that determination. It was my first ever technical interview, and I fucked it up. That's going to happen. But I knew I could be an asset on this team and I need to figure out a way to show that. So I wrote the following email, which changed my life.


Hey [redacted],

Thanks for taking the time to speak with me and review some code concerning the iOS Developer opportunity at Breathometer.

I'd like to offer to work on something useful for breathometer as a way to showcase my ability to get things done on my own and to a high standard. As a developer without a lot of experience, I completely understand that I need to prove my abilities more than the typical candidate. I'd love to get that chance.

Thanks again,

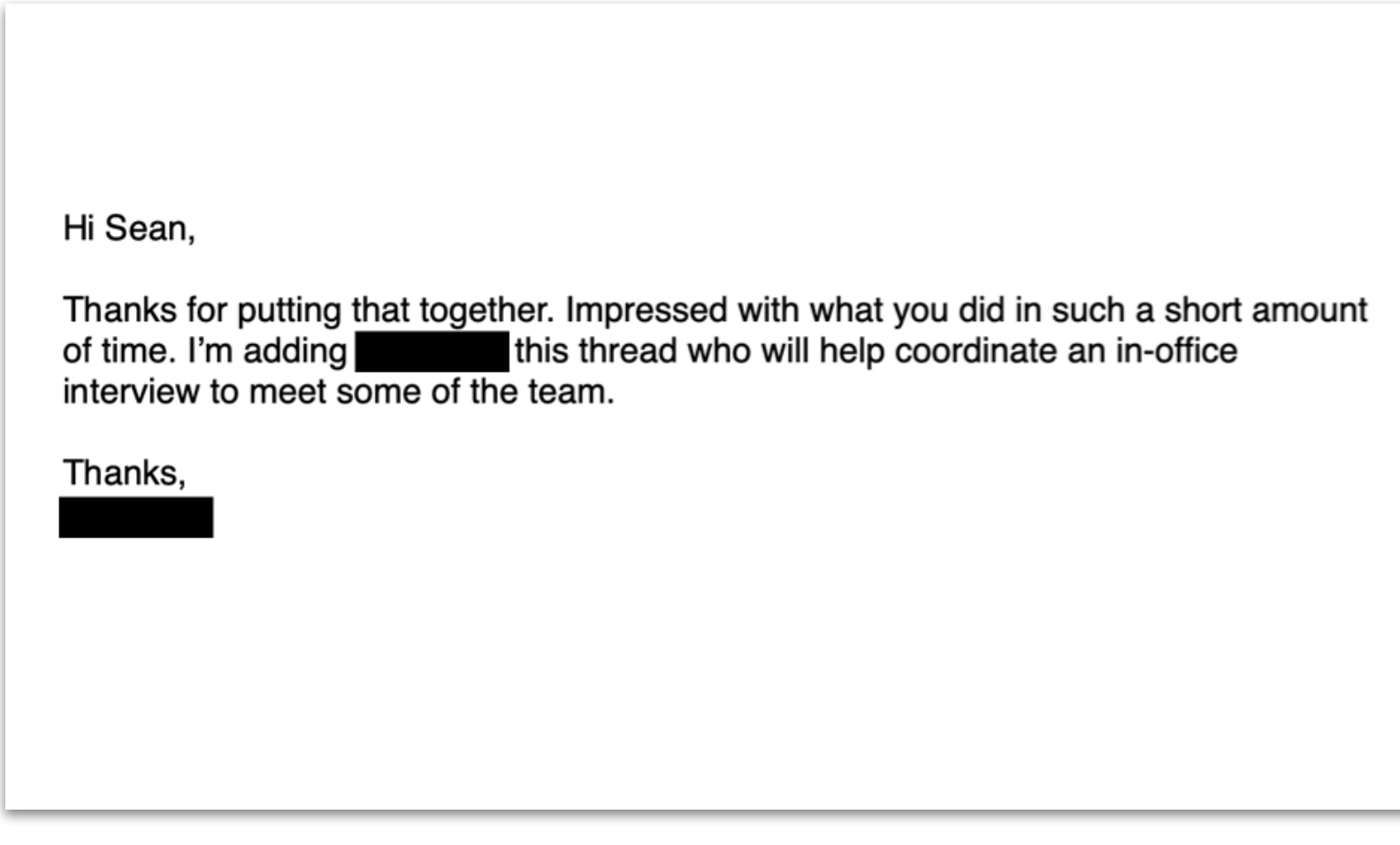
Sean Allen
seanallen.co

They responded the next day, a Friday. They were willing to give me a take home assignment to see what I could do.

Let's. Fucking. Go. I took a shot and it worked. This is my chance. Eminem's "Lose Yourself" was playing non-stop in my head. I spent that entire weekend immersed in this project. I'm talking 10 hours a day on Saturday and Sunday. This project was my only chance so I went all in. On Monday, I submitted my project. It was good. I was damn proud of it. I'm gaining confidence.

I hit send on my email at 1:54 pm; they responded by 2:19 pm. 25 minutes. Such a fast response didn't exactly fill me with confidence. Surely they couldn't have thoroughly reviewed it that fast. I figured they had just humored me, and this was the final nail in the coffin.

Nope: they were impressed and wanted to bring me in for an in-office interview to meet the team.



Hi Sean,

Thanks for putting that together. Impressed with what you did in such a short amount of time. I'm adding [redacted] this thread who will help coordinate an in-office interview to meet some of the team.

Thanks,
[redacted]

Did you catch the word 'interview' in that last sentence? Because I didn't. I was so pumped, I read it as "come down to the office and meet the team". I assumed I just had to meet everyone to pass the culture fit portion, and I was good to go. Idiot.

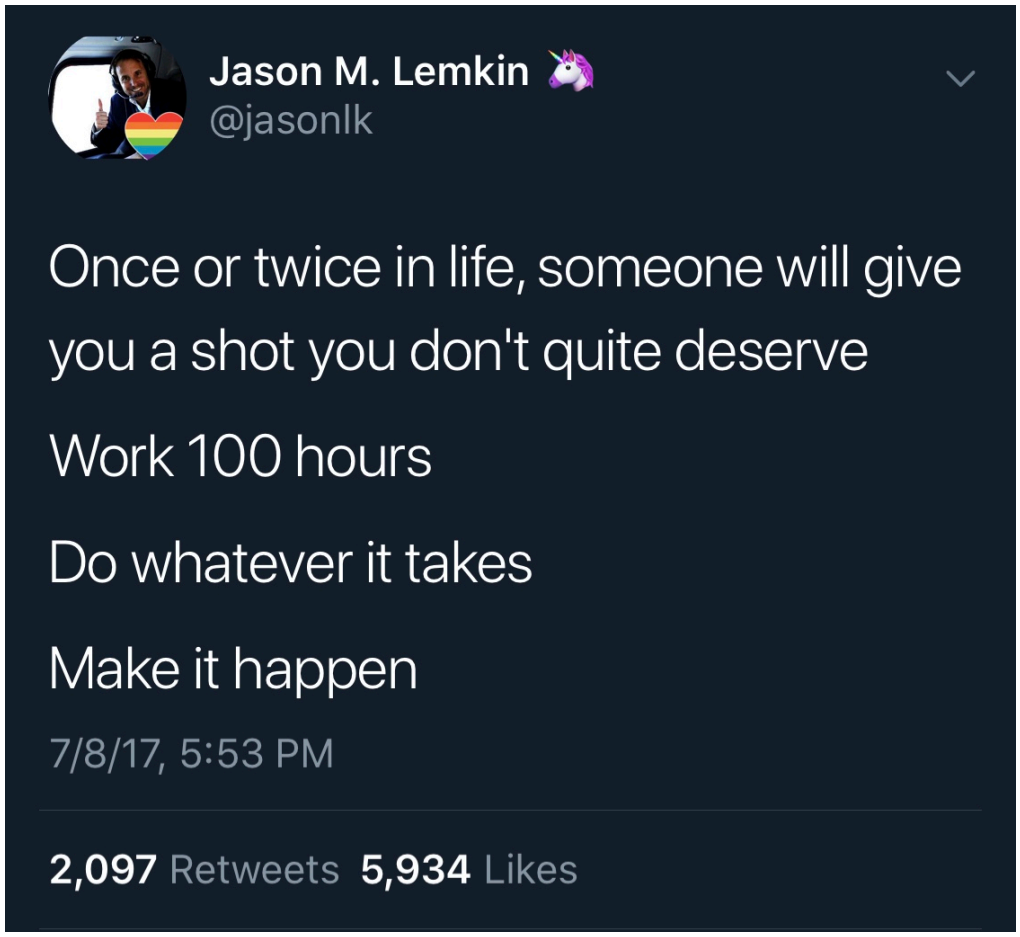
Are you ready to see naiveté at its finest? I strutted into the Breathometer offices full of confidence and excitement, only to be told this was my final onsite interview and I'll be meeting with five different people throughout the day. Two of which were technical interviews. That excitement and confidence went away real fucking quick. My stomach dropped. For those that have never done one of these final onsite interviews before… they are brutal. They typically

last four to six hours and you are interviewed by multiple team members, including leadership. Again, I wrongly assumed I had the job, and I was just meeting the team. So I did *zero* preparation. None. I literally walked into my first ever final onsite interview without studying for a single minute. Like… what?! Seriously? I had to idea what the hell I was doing.

Can you guess how this went? Wrong. It didn't go well. Oh, that's what you guessed? OK, then you're right. But it wasn't *all* bad, and that's the key. I had great discussions with the CTO, CEO and VP of Software. Those were the non-technical interviews and I crushed those. The two technical interviews were a train wreck. Luckily, I had impressed the right people in the non-technical interviews, and they decided to take a chance on me. I received my first iOS dev job offer a few days later. I literally teared up when I got it. I did it. I fucking did it. Holy shit. It wasn't pretty, but I did it. I made the jump from having never coded in my life to now being a full-time iOS developer. It took about seven months.

Here's the deal. I absolutely *did not* get hired because of my developer skills. They liked my initiative, diverse background, personality, attitude and maturity. Based on some of my discussions, it sounded like they had a poor experience with young developers in the past so my advanced age was actually an asset. To the older set reading this book, keep that in mind. You're not too old.

Sometimes I think I just got lucky. And maybe I did. But keep in mind, you don't get lucky by sitting around waiting for something to happen. Go get it.



This was my exact mindset at the time.

Chapter 7: Industry Initiation

Let me tell you a little bit about Breathometer. The product that got all five Sharks to invest was a personal breathalyzer you could hook up to your phone. Safety first, folks. Drink responsibly. After Shark Tank, the company started moving into other breath related devices.

First up was an oral health product called Mint. Mint was designed to track and manage oral health by measuring the amount of volatile sulfur compounds in your breath. Not that I really had a choice, but I wasn't excited about that product. As I was coming onboard, they were beginning to work on their third product called Burn. Now this is what I'm talking about. I was pumped for Burn and couldn't wait to work on it. During my interviews they told me I was being brought on for the sole purpose of working on Burn.

They needed a demo for the upcoming Consumer Electronics Show in Las Vegas, and that was to be my first project.

Burn was a device designed to help identify whether or not the user was burning stored fat. Anyone who has ever tried to lose weight and get in shape knows that often times, the scale does not tell the full story. If you're familiar with the ketogenic diet or ketosis, then Burn will make sense. I'm not a doctor, obviously, so this is a very high level description. When you're on a low carb or ketogenic diet, your body starts to produce ketones. One of these ketones, acetone, is small enough to escape the blood stream, get into your lungs, and therefore your breath. Burn would detect the level of acetone in your breath and therefore could tell how much stored fat your body was burning.

I loved this idea. I was excited for it. The Breathometer team knew that I was playing basketball all the time, so I was likely the most active in the company. I was a good guinea pig for the product. Shortly after starting at the company, I started a keto diet (low carb, high fat) so I could test the product and it's ability to determine if my body had entered ketosis. Being the guinea pig was pretty cool; I lost like thirty pounds in the first month alone. It was definitely too much, too fast. The diet combined with all that basketball… the weight just melted off. To see this first hand, go look back at some of my very first videos and see how weirdly skinny I look.

I joined Breathometer in November of 2015, when the company was gearing up for CES in January. The idea was to introduce Burn at the event. When I first walked through those doors in November, I thought I was going to hit the ground running on Burn and help the team get ready for CES. In reality, I walked into more of a bait and switch: I ended up working on Mint, a product I thought was doomed for failure and had no interest in for about 14 months before I had the chance to work on Burn. Of course, beggars can't be choosers and I'm thrilled I even had a developer job. However, knowing that there's a *much* better project on the horizon that you can never seem to get to was frustrating. It wasn't all bad, though. I learned a ton at Breathometer, both good and bad. Let's talk about it.

Aside from not loving the project I was working on, the main downside to my time here was the lack of senior developer mentorship. Breathometer was not a huge company with roughly 25 people, so the iOS team was small. Just myself and another iOS developer who didn't have much more experience than me. It was like the "blind leading the blind." There was a senior developer contracting for the company, but he was part-time and based in Israel so our interactions weren't as frequent as I would've liked. Also, there's no replacement for sitting next to someone and working with them each and every day.

That's my biggest piece of advice to developers looking for their first job. Make sure the team around you is great, and has at least one senior developer. Ideally, this senior will teach you about the art

of programming. I'm talking about the "why" we do things certain way. The type of knowledge that only comes from experience, not an online tutorial. Having a mentor early in your career is the best investment you can make. Invest early and the interest keeps compounding throughout your career.

Not having that senior developer wasn't all bad though. I was able to slip into a leadership role almost immediately. I wasn't an iOS lead by any means, but I was given a lot of responsibility very early in my career. I was in charge of the entire client side of the app. I was planning and leading our two week sprints. Oh, I was also the designer for the app.

As an Apple fanboy, amazing design is very important to me. I like to call myself a design enthusiast. By no means am I a professional designer, but I pay attention to that world and I know things. Breathometer didn't have an in-house designer, so they were contracting that out to this expensive design firm. Their work wasn't great (I'm being nice here). So, picture this: here I am a fresh new iOS developer super excited to be finally building apps professionally, and then here's this shitty design that would be embarrassing to show people. You know that question that comes up with friends and family. "What are you working on?"… "Uhhh… nothing!" Would have been my answer as I ran away from the conversation.

I couldn't let that happen. Even I could do a much better job. So, after my first week on the job, I took it upon myself to redesign the entire app over the weekend in Sketch. On Monday I showed it to the VP of Software. My motivation was to show them that I, a simple design enthusiast, could do a much better job than this firm, and that we should look elsewhere for our design work. That's not what happened. The VP of Software took my designs to the CEO, and 20 minutes later that firm was fired, and I was the designer. That wasn't my intention, but ok, I guess this is happening now.

One quick piece of advice before anyone gets any ideas. I wouldn't do this again. While I love design and have a blast creating screens in Sketch, I quickly learned being a designer for a company is a lot different than doing it on the side for fun. Design is subjective and I had input coming at me from all angles. The CTO says one thing, then the CEO comes in and says another… It sucked. Professional designers know how to handle this. I was not a professional designer. After all was said and done, the design ended up looking almost nothing like my initial vision. Be careful what you wish for.

Let's talk money. Before joining Breathometer, I had been making $60,000 per year as the Growth Lead at the small startup, Playpass. When I got my first developer job, I was hoping to make more than that. My initial offer at Breathometer was for $60,000 per year. They say you should never accept the first offer. Well, I'm dumb and jumped all over it. Remember my mindset at the time. This was the only company even remotely interested. I needed to get my foot in the door of this industry. I knew this salary would be temporary and

didn't want to risk anything by countering. I'm not saying this was the best strategic move… but it's my truth. (Yes, I understand that in many areas of world, $60k is a ton of money. But San Francisco is very expensive to live in, so the salaries compensate – every market around the world will be different.)

When I first got the job, I was fine with that initial salary because it was what I was making before, so my living expenses were already based on that. Fast forward few months, I was no longer happy with that amount. At this point, I was doing two jobs for less than market value for even just one of those jobs. In these few months I had undeniably proven my worth and value to the company. I was an asset. Something had to change. Taking on all the extra responsibility made negotiating a salary increase a piece of cake. It was undeniable I deserved it. Was just a matter of actually making it happen.

The VP of Software told me just one month after I started that he was going to recommend I get a pay bump to $80,000, while sheepishly admitting I was low-balled on my offer. However, the company couldn't do that until after their Series A round closed. We agreed on my new salary in December, and the Series A round of funding was expected to close in February. I had to wait a couple months, I didn't care. My good work was being recognized and the raise was coming soon. I'm pumped.

Once again, I was naive. At the time, this was my first rodeo. But I've since learned that a Series A timeline is fluid at best. February close date became March. March became April. April became May. I became frustrated. Finally, when the May close date became June, things came to a head. I told my boss my last day would be May 30th. I literally put in my two weeks notice. I had an offer on the table from another company for $90,000, and I was going to take it. I didn't intend for this to be a negotiation tactic. I'm a shitty negotiator. But that's exactly what it ended up happening. The VP of Software at the time went into a bit of a panic as he did *not* want to lose me. He asked "What do you want. I'll do what I can to make it happen".

Now I have some leverage. Being patient, although internally frustrated, paid off. Had I received my raise in February this would all be over and done with at $80,000. What do I want? "$120,000 per year", I said. I had nothing to lose, so I figured why the hell not ask for way more than I should. "Let me see what I can do,", was the response I got from my boss. The next day, he came back to me and said they couldn't do that, but maybe there's some non-monetary benefits we can discuss. What we landed on was $100,000 per year and the ability to work from home two days per week, with a little bonus. That bonus was that for one of those work from home days, it was understood that there would be "very low expectations" from me that day. This allowed me to work on various side projects as well as my budding contracting career at that time. It was a sweet deal. This is all happening six months removed from a dev bootcamp where I didn't know a thing about iOS development. I can't complain at all.

Here's the punchline. A lot of people will tell you to negotiate hard on your initial salary. I'm not saying I disagree with that, but it's not absolute. It's certainly possible to take less money to start, prove yourself, and make yourself irreplaceable. It's a long term plan, but in most cases the money will come. This is a marathon, not a sprint.

Breathometer started out a little rough, I'm not going to lie. It was a great introduction into this industry even though it was a total trial by fire. I spent about year and a half at this position and eventually got to work on Burn for my last few months there. Hell, I even went to the 2017 Consumer Electronics Show to present demos of Burn.

Presenting live demos of Breathometer's Burn at CES in 2017.

In February of 2017, I was lured away from Breathometer. What lured me away from this sweet deal? Facebook. An innocent little recruiter message from Facebook appeared in my LinkedIn inbox. They wanted me to interview. Turns out my initiation into this industry wasn't quite done. At Breathometer, I was on a high. I could do no wrong. Big fish, small pond. I was about to try my hand at the big leagues. And little did I know… I was about to get destroyed.

But that's a story for another time.

Epilogue

Thanks for reading and I hope you enjoyed hearing my story. Even more, I hope you gained insights and value from it.

But this was just the beginning. Over the next two and a half years, I got crushed in interviews all around Silicon Valley, started a YouTube channel, spent a year and a half freelancing, and ideally will land a job at a major company. Maybe I should do a part two…