5 Secrets to a Programmer's DREAM Career

INVENTIVE

Miguel Gonzalez
Devin Rose
Andrew Siemer

5 secrets to a programmers DREAM career

To DREAM big is a matter of choice. To reach that DREAM is a matter of discipline.

Andrew Siemer, Miguel A. Gonzalez and Devin Rose

This book is for sale at
http://leanpub.com/programmers-dream-career

This version was published on 2018-12-03

Tweet This Book!

Please help Andrew Siemer, Miguel A. Gonzalez and Devin Rose by spreading the word about this book on Twitter!

The suggested tweet for this book is:

I just bought #DreamCareer by @DevSpringboard @asiemer @techdevman

The suggested hashtag for this book is #DreamCareer.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#DreamCareer

Contents

Introduction

Most people ask us "What inspired you to start the Developer-Springboard.com website and write such a book?" To which we usually reply "you did". This book is a collection of our experiences and observations over time. All three of the authors have worked in the programming industry for quite a while.

Two of us (Miguel and Andy) have worked at more than one job at a time continuously for around 20 years. Always moving from one exciting project to the next. Generally being invited to go to the next place by a friend we worked with previously or by someone who heard we were likely available and could use our talents.

In moving around so much we had a lot of exposure to people that never moved around. The other author of this book, Devin, was such a developer. Having worked at one company straight out of college for more than 10 years, he was experiencing everything Miguel and I were seeing as we moved around.

Then we all came together.

Devin lived just down the road from my farm on his own farm. I was butchering 100 chickens that day and had a sign out on the road for passers by to get some of our farm fresh chickens. He was passing by and drove up. I could tell from how he spoke and the things we were talking about that we were going to hit it off. Then I found out that he too was a programmer. We became fast friends.

Eventually I found myself suggesting to Devin that he really shouldn't keep working at one place for so long. That he was

missing experience, career opportunities, happiness at work, and – more money. After talking back and forth about this for 6 months or so Devin was convinced. He started looking for a new gig. And found a hot Austin start up doing a job that he had never done before. Exactly the fresh perspective he needed. After working there for 8 months or so I was able to get Devin to move again to work directly with me. Again doing something new and exciting. Each time he moved, he took on new challenges, kept a smile, and made more pay.

Looking back, all three of us have had several such stories in our past. A willingness to help others. Experience that taught us something colleges and the industry wasn't teaching. And exposure to many young folks just starting their career.

Devin and I had both written books before. At lunch, some of our young programming friends got an ear full of career advice from us. They were complaining about this and that. But felt like they had no power to change anything. After listening intently they told us we should write this book.

So we did.

Help Us Make This Book Better

The intent of this book was to get the information out quickly. And to self publish it. While we have done our best to edit this book and review this book just like a big publishing company would, this book can always be better. Please take a minute to tell us when you find something that doesn't feel quite right. This could be a mis-spelling, something out of order, an unclear point being made, etc.

Please feel free to post your findings on our forum. Or if you want to start a conversation with the group do that too.

https://groups.google.com/a/developerspringboard.com/d/forum/dream-career

Or you can email us at dream-career@developerspringboard.com.

Please enjoy the book!

Foreword

About the Authors

Miguel A. Gonzalez

Miguel has designed, developed and managed dozens of software projects for clients in a wide variety of industries including emergency management, media, insurance, real estate, logistics, marketing, education, and retail. In the process, he has gained valuable experience building highly available, high bandwidth applications. He has a passion for robust, precise and efficient software development. His recent focus has been in leading the development of distributed, scalable applications for several top-tier clients.

After obtaining his B.A. in Computer Science at the University of Texas at Austin, Miguel began his career as a software developer designing portable window-based client applications. Since then, he has worked for numerous companies, such as the United States Chemical Weapons Program, Hearst, Belo Interactive, Wachovia, and DELL. His work has included heavy .NET utilization in C#, as well as many other languages and technologies.

Miguel is a father of 4 – 3 girls and 1 boy. He enjoys spending his leisure time surrounded by friends and family.

Andrew T. Siemer

Since 1998 Andy has been a self-educated software engineer that specializes in large scale data driven web applications primarily focused on social networking and ecommerce platforms. His technology focus has revolved around Microsoft offerings but has

always lived in a world where you use the right tool for the job – and Microsoft isn't the only tool maker! Admittedly, his focus has been on the .NET stack ever since it first shipped in its beta form. Andy tends to spend quite a bit of time with his head in the clouds – specifically Azure and Amazon. Andy has done work for Dell, HP, Fox Interactive, MySpace, American Idol, Callaway Golf, 3M, Emerson, LampsPlus, RackSpace, OTX, Guidance Software, and many more.

Andy has worked at nearly all levels in his career. He started as an Access DBA automating a manufacturing facility and quickly found himself upside down in a web dev role making that system accessible to the entire company. He has been a team lead, manager, director, and VP, too. After leaving the role of Chief Architect for dell.com, Andy now spends the majority of his time as the Chief Architect for Clear Measure where he attempts to keep his finger on the pulse of all the software projects they are building and managing.

Andy is a member of the ASP Insiders group. He also manages the Azure Austin user group. Andy is currently a Microsoft Virtual Technology Specialist and Azure Advisor.

As an active contributor to the developer world, Andy has published three technical books with Packt Publishing (ASP.NET 3.5 Social Networking, ASP.NET 4 Social Networking, and ASP.NET MVC 2 Cookbook). He is also currently working to self publish a farming book on Fodder with LeanPub. Andy has also done quite a bit of public speaking from California, to Texas, to New York. He is also helping Clear Measure build and deliver day long hands on deep dive courses. And is putting together the first Clear Measure conference – "MeasureUP".

Prior to rubbing the letters off of his keyboards, Andy served in the US Army from 1994 to 1998 in the airborne infantry. He spent most of his time in 2nd Ranger Battalion and a then a short stint in a recon unit.

Andy is the father of 6, owns and operates a small farm near Austin Texas – Friendly Pastures, and is happily married to a great lady who strangely still puts up with his always busy and bigger than life nature. He loves to shoot guns, drive jeeps as near to vertical as possible, do wacky obstacle courses and group activities, and has recently started to get into motocross. And when time permits he still does the occasional side job from time to time. Friend him somewhere – his smile is infectious!

Devin S. Rose

Devin's an average Joe developer and believes that others like him should be able to program great software.

He got his degree in electrical engineering but couldn't take many more Maxwell's equations so went into software. After programming a Tetris clone with a guy from Norway, he got a job at National Instruments and worked on C++ projects in the test and measurement world. He wrote code for real-time operating systems and Windows and Linux desktop environments.

Though COM is a bad word now, he developed COM-like interfaces in C++ and taught others how to componentize their C++ apps. He worked on an object-oriented database before that fell out of fashion and developed straight-up RPC code between processes on embedded computers.

On the side he missed the After Dark screensaver collection, especially the Marbles one, so he wrote a clone of it and added black holes to it, as well as multiple active bouncy balls.

For seven years he worked on the LabVIEW graphical programming language, then he went to a startup and developed a cross-platform mobile app using Xamarin. His most recent step has been to work at Clear Measure and develop web backends with Web API and complementary technologies.

Devin has written three books: *Farm Flop*, a book chronicling his misadventures in hobby farming; *The Protestant's Dilemma* and *If Protestantism Is True*, two best-selling books in the Catholic apologetics space.

Devin was a farmer but hurt his back and limped back to the city, where he lives with his wife and two children.

Welcome to Your Programmer's DREAM Career

When you graduate from college, or when you start on your first software development job, no one gives you a guide on how to navigate your programming career.

Instead, what most of us do is put our head down, make our fingers punch out code–of whatever quality–at the keyboard, and look up perhaps four years later to see what is going on in the software world around us. Or, in my case, it was *fourteen* years later.

We programmers have technically oriented minds, sharp intellects that can understand how to make a rigidly calculating computer do what we want it to do. We have those smarts, but we don't always have the intuition or broader awareness of what we need to do at our jobs and in our lives when we are away from the coding itself, in order to be successful in our careers for the long haul.

That is why we wrote this manual. It is the guide we wished that we had had when we were starting out. It is the guide I wished someone had given to me when I was seven years into my software career. At least I would have saved seven more years of laboring away on products using outdated technologies.

Learn from our mistakes. Benefit from wisdom we gained through much bumbling around in the dark.

What is the DREAM?

DREAM stands for Discover, Refine, Establish, Advance, and Master.

We divided this book into these sections, with each one containing chapters devoted to the fundamental things you need to understand in order to make your career a stellar one. That may translate into: getting paid more money, getting promotions more quickly, finding more fulfilling jobs at better companies, becoming well-known in the software world, speaking at conferences and so on.

We will first help you to **Discover** who you are. Every person is different and has different values, goals in life, personalities, and tendencies. These traits are usually not accidental, and you can use the positive aspects of your personality to focus your career's direction and accelerate it. What sounds like the best life imaginable to your programmer friend may bore you to death or even repel you completely. That's okay; both of you will take different directions based on your life goals. The important thing is *knowing* what those goals are–knowing who you are–so that you can act in an intentional way to realizing them.

After you discover who you are, we will help you **Refine** your focus. What areas of software technology should you seek to learn and work in? What track is right for you: management or programming? And are you more of an architect or engineer? When you answer these questions you will be able to formulate clear goals about where to work, which track to take, and how to avoid becoming stale on irrelevant technologies.

Once you have refined your career's focus, you can begin to **Establish** yourself. You want to start building your own personal brand, one that retains its value even if you leave the company you currently work for and move on to another one. You want to become known in your own right in the software development industry, as it will open doors for you and make getting through

those doors easier. We will also teach you to continuously update your resume, improve your interviewing skills, and try out different roles to broaden your experience. Establishing yourself will also include finding one or more mentors and looking for people that you can be a mentor to.

You will then begin to **Advance** your career. For many of us, this is where we spend the majority of our productive lives. We will show you key principles that you need to internalize in order to advance your career to the next level and the next level beyond that. Some of these will come naturally to you; others will be difficult. Advancing your career involves increasing in technical skill, but even more importantly it involves how you build relationships, your attitude, work-culture considerations, and how you approach your software tasks.

Our ultimate goal for you is to advance you to the **Master** stage. While you are always working on the previous stages, you will reach the point where you are taking the steps to master various aspects of your career. What if you wake up one day and realize that your actual skills don't meet what your job title says you can do? What do you need to be doing as a middle-aged developer to ensure you remain valuable? What productivity skills do you need to remove distractions from your work life? And finally, how can you work on your own side project, create your own product, or start your own business, so that you are no longer dependent on a company giving you a paycheck.

The DREAM Team

Who are we to write this book? We are three guys: Miguel, Andrew, and Devin, with a combined total of fifty years of software development experience. We started near the ground floor of programming–though after punch cards!–and have worked on everything from real-time operating systems in C++ to load balancing

web servers.

We've been software developers, staff engineers, senior engineers, principal engineers, lead architects, chief architects, and worn several other hats in addition. One of us has had seventeen jobs in twenty years, while another stayed at the same company for over thirteen years. Andrew picks up things quickly, while Devin plods along methodically until he understands something. Andrew never went to college; Devin and Miguel have four year degrees from good computer science schools.

In short, our experiences are varied, and that allows us to speak to a wide range of topics. The common denominator for all of us is that we realized at some point in our careers–Devin later than the other guys–that we needed to actively manage our careers, set goals, learn good core values and live them, in order to become what we wanted to be.

We are pilgrims on the same journey as you, but we have made realizations that most other developers have not. And we want to share that wisdom with you in the following chapters.

As you read, jot down notes or send us an email. We want to hear from you, good and bad. You can find us at http://developerspringboard.com[1]. Without further ado, let us begin!

[1] http://developerspringboard.com

Devin's Three Career Mistakes to Avoid

Remember the old days, when you would go to work out of college and stay at the same company for forty years? They'd give you a watch and a pension, and you would be set for the rest of your life. Those days are long gone. And yet, somehow I never got that memo. I worked at the same company out of college for almost 14 years. And it was a mistake to do so.

But before I tell you why, let me give you a little background. I was a star student, straight As through high school, full scholarship to a good school studying electrical engineering. I graduated but never loved electrical; my heart was in developing software. So out of college, I landed a job at a good tech company in the scientific and engineering area. But soon after starting, I made my first mistakes.

The Blinders

I started out my work on a C++ project. It was pretty cool actually, a utility app that internal product groups plugged into to get their hardware to show up in it. I learned a good deal on this project, but I put the blinders on technology-wise. I wasn't keeping track of what was going on in the world around me.

This was from 2001 to 2006 or so, when the web was continuing to evolve at a break-neck pace. "Web programming?" I scoffed. "That's not even real programming. This hardcore C++ embedded and desktop application programming is the real stuff." Ah, foolish Devin. How wrong you were. While I was cutting my teeth on interesting problems, I was also getting very narrow in my skillset:

desktop app development in C++ using antiquated UI technologies like Win32/GDI and MFC.

The takeaway here is to make sure you are continually learning and keeping track of what is happening in the world outside your office and even your company. Work on side projects to play around with new technologies. Read blogs, and if you are a Luddite like me, force yourself to buy the new technology gadget at least once every two years.

Dinosaur Syndrome

I worked on my first project for over 6 years. And I finished my time at the company with a second project that I labored on for over 7 years. Yes, a combined total of over 13 years at one company essentially only working on two things.

I thought I was loyal to the company and that the company would be loyal to me. But when the rubber met the road, the company wasn't willing to budge an inch to show me that they cared that I had invested so much of my time and thought with them. I took it personally at first but not anymore. The company had grown in my time there, quadrupling in size to almost 10,000 people. When you get that big you are not a close-knit group; you're a medium to large sized big-corp. The little guy doesn't matter anymore, nor do you have to act like he does. Should you? Yes, but that's a story for another day.

Bottom line is: don't work at a company for 13 years. Unless by the end of it you are the CTO or CEO and make millions of dollars per year, get out of there after two to four years, especially for your first company. Level up to at least Staff Software Engineer and jump ship. Try a small company if you've been at a large one, or vice-versa. **You are in the growing and gaining experience phase in your career, so think diversity of experience over going super**

deep and long at one place and with one technology. I wished that someone had told me this when I was three years into my first job, the one I remained at for far too long.

Perception Problems

I used to brag that other companies had politics but that the company I worked at was a pure meritocracy. I was serious and had been told that, and I believed it for about ten years. Then I started to realize it was not a pure meritocracy, and that politics did exist (duh). I had not played the game or been wise on how to make sure that I was working on projects with good visibility, doing things that senior management saw and rewarded. Instead, I thought that if I just kept doing a good job they would realize the value of it and recompense me.

Meanwhile, smarter coworkers were constantly getting the boss's attention and demoing little things to them. Even things that I and others had worked to make happen but didn't think to brag about and show demos of. Hey, it's a team thing right? Right and wrong. There's nothing wrong with showing what you've done, and we as programmers tend to do it less than we should.

Also, I realized after some time that I had gained a perception as someone who put in his hours but didn't go above and beyond. I was late in noticing this and worked hard to show that I could make a powerful impact on the company, bringing new development methodologies that were proven in the software craft but that we had ignored. Too late: the company tech culture was built to look skeptically at outside ideas, even proven ones, and people like me who pushed them were dinged for not being good culture fits.

Perception influences reality. If that perception is objectively accurate (a true perception), then that is a good thing. But when the perception is false, erroneous, it leads to bad decisions and actions.

Be aware, be mindful, of how you are perceived at the company, by your supervisors, and by your coworkers. Not to suck up or something like that, but to understand if a false perception is being built up that you need to counter, or a true but critical one is developing that you can change about yourself.

New Trails

I left that job and worked at a startup developing their cross-platform mobile app using Xamarin. Then I left that job and work at a custom engineering company where I've implemented a Web API backend for an iPad app and web client, using NServiceBus, SignalR, CQRS, TDD, and many other good practices and technologies. I've learned more in a year than I did in the last five years at my old job. Avoid my mistakes: be nimble, externally aware and self-aware, and bypass the long slog that I went through to get to a good place in your knowledge and career.

Miguel's Career Hindsight

When I look back at the choices I have made in the almost 17 years since I graduated from college, I sometimes question if I knew what I was doing early in my career. Actually, I take that back. There is no doubt about it…I really didn't know what I was doing back then. I had been working since I was very young, but I had never worked in a professional environment until after I graduated from college. In school, I was taught several programming languages, how to use them to build software and how to interpret business requirements, but I was never coached on any of the light skills that are necessary for making a career out of what I had learned.

Frankly, I really didn't know what it meant to be a professional software developer or what I had to do to make a good career out of it. Had I known then what I know now, I would have done some things differently. I made so many mistakes early on and I missed out on a lot of great opportunities because I was just not focused or looking out for all of the career cues that I now know exist.

Fortunately, I was able to overcome this shortsightedness early enough that I was able to recognize some of these cues and use them to guide my career choices in a positive direction. One of my goals with this book, and of Developer Springboard, is to help others identify these cues early on in their careers. In doing so, I hope I can help those that are new to the industry become good software developers without having to make all of the mistakes that I, and many of my peers, have made.

In this book we will dig deeper into some of the mistakes that I and the other guys have made, because I was not focused, as well as some of the things that I've identified as important cues to look out

for in your career. I think we can all learn from my experience and apply those lessons to our careers to make better decisions and not miss out on the opportunities that we are presented with every day.

Jack of All Trades - Master of Some: Andy's Path Forward

I noticed a pattern in my life early on. I tended to go deep into a topic in a very short amount of time. It never mattered what the topic was, just that I had some interest in it. If I was interested in it, I was all in on the spot. This is true even now. But my life was never lived with a plan – just an interest in some topic that led to another interest.

I had a father that I remember doing everything himself. I always saw him tinkering on something. He didn't spend much time glued to the TV. And he always had the tools or books or whatever he needed to work with that topic. It seemed that he knew everything related to a given topic. To him, these topics spanned construction, all sorts of fishing, almost every war, airplanes, plants, and chemistry.

By day, he had a PhD in plant sciences and ran a company running various experiments to make plants bigger, better, healthier, pest and disease resistant, while growing faster. He did a lot of research trying to make a difference in the agricultural world. He flew all over the place (often times in his own plane with him as the pilot) working with people from all over the world. I like to tell a story about him contributing the size, flavor, and seed-less-ness of grapes as you now know them. When I was a kid grapes were tiny blah seeded things. Dad loved grapes and worked to make them better.

One of his knacks that fascinated me was his recall. Someone might ask him a question about a report he wrote and he could tell you all about what he wrote 15 years ago up in the storage at the office

in a box buried 3 rows back on the bottom. This binder. That page. This paragraph. Yep – there it is. How did he do that? This led to him eventually being a very talented "professional witness" in agricultural legal trials.

I am not sure my recall capabilities are quite as good as his. I forget my birthday for Christ's sake! But I have always been able to skim through magazines and books cover to cover. I never left the information with full recall of a topic but I could always leave with the vocabulary in hand and enough information to be dangerous.

Having seen my dad do anything he set out to, I have always felt empowered to dive in and just get it done. I had built an entire "club house" around the age of 12. And by club house I mean 8'x8'x14' building complete with sleeping loft, door, windows, wired for plugs and lighting (plugged into my mom's house – she hated that) where I spent a lot of time! My mom had one of my Dad's friends come inspect it to ensure it was safe. He said it was "Up to code! How did he do that?"

I had family issues growing up which caused me to be an interesting kid. By the second grade, I was already being kicked out of school. My mom sent me to live with my dad when I was 7 – she couldn't handle me. I don't know how my dad managed to raise me without seeing the inside of a jail! Among a plethora of other issues, I had eventually found a way to get kicked out of high school a couple months into my 10th grade year. Out of the school district that is. I was forced into home school. Dad took me to work with him every day at 7 and home again at 6. I had to sit in the library and do nothing but school. No talking. No help. But in my sophomore year, I had jumped ahead in school. By the end of my junior year, I was done with high school. My senior year was a blast. I had two PE classes, 2 wood shops, 1 metal shop, and 1 auto shop (and a couple half semester things I couldn't take until my senior year). With access to all these great facilities I was building swords, cross bows, knives – thankfully my shpp teachers got to know me well and were okay with this crazy kid. But it was clear that my parents

were done being responsible for me. Where to now?

Off to the Army I go. Like most of my life to this point, I had bumbled from one interest to the next. Why not do that in the Military? I tested very high and was told I could have any job I wanted. They sold me on 11X-ray. I would get to shoot and blow things up. Sign me up. They even "gave me" an airborne contract. Don't trust recruiters – they are there to fill quotas. 11X-ray is an open ended infantry contract. Grunt. Oops!

Duped into a life as a worker bee, I went through basic and AIT without much problem. Earned my PFC quickly. Showed an aptitude to lead. Off to airborne school. Did well enough. Got recruited into the RIP program – ranger indoctrination program (now called RASP – ranger assessment and selection program). Got through that. Showed up to ranger battalion where they looked at me and immediately decided I could carry heavy shit – weapon's squad for me! It was a fun period in my life. I have never regretted it in the least. I learned that while I can suffer through anything you might throw at me, I needed to work with my head, not my back.

Once I got out of the Military, I tried "work". I attempted to hang wire for an industrial electrician. Having just gotten out of a world where shit rolls downhill, I quickly realized that I was not going to cut it in a trade where someone that is happy doing the same thing every day for 40 years is telling me how to do something inefficient – back to school. ITT Tech! I started in a CAD program and had my first real introduction to computers. I had touched them a couple of times in previous years but never in any way that sparked my true interest. Now, I was sitting in front of a computer every day.

Talk about a never ending stream of learning! I loved the world of computers. Other than learning command line access to CAD programs, how to hand draw blue prints (right up my alley), and draw in Auto CAD, I was learning daily about networking, computer internals, and how to make this thing go faster. I quickly

started to collect dead computer chassis and bring them back to life. I got a job as an in-store tech support guy for HP where I stood in BestBuy and Fry's to talk to customers as an HP rep. Every day I learned something I didn't know I was missing. Someone showed me HTML in notepad. Another guy showed me javascript. I had friends that could "fdisk" their computers – what's that? Quickly I found myself in a part-time network admin gig (not that I knew how to do that job yet) and started to figure things out. Each day I was getting further away from enjoying sitting in a class learning as slow as the teacher would trickle information to us.

My first networking gig was with a company that built shutters which they managed on a magnetic white board. This was insanely inefficient to me. Someone mentioned I should look into making a database to manage that. What's that? OK. But entering data directly into a database wasn't going to work. Build a web site. OK. We need servers to run this stuff. OK. A great first playground for me.

All the while going to school where I had two teachers that I interacted with quite a bit. One of them hated that I was so all over the place. I built video games in Dd studio max and some world building program (when we were just supposed to model something simple). Or I would go deep into photoshop. You name it – if the assignment was simple I would go over the top in five different directions. This guy pulled me out of the class (in the middle of class) to go for a walk. We walked around the parking lot. He asked me "what do you want to do when you grow up?" I replied that I had no idea, I liked a lot of things. He told me that I would eventually have to focus on being really good at just one thing to make it in this world. A "Jack of all trades, master of none, had no place in this world." This didn't sit well with me as I tended to be very scattered and really enjoyed doing all sorts of things – not just one thing.

The other teacher pulled me aside in a similar fashion. He saw that I was tinkering in the web world already and suggested that I

should go learn a "real programming language". His suggestion was that I learn Visual Basic. I giggle at this now of course. I bought a book on PERL (oops). He was a GIS and big data guy before the term big data existed. He saw where my mind was and steered me in the right direction. I got my associates degree at ITT in CAD with a 3.8 but didn't complete my bachelor's degree. I didn't think I needed school anymore – LET'S DO THIS! This pattern of constantly bumping into what I didn't know pushed me to learn more and more. I almost always had three jobs running at one time. Tech support, networking, cable pulling, LAN configuration, phone lines, IVR, VOIP, server builds, and building out data centers. I was very hardware oriented for the first couple of years. At some point, I bridged over to building data driven dynamic web sites. I was doing AJAX before the term existed. I started with simple commerce websites plus managing the data center, then social networking, then big commerce. Then big systems piqued my interest in general.

To this day, there is nothing that I enjoy more than cracking open a new topic. From rebuilding a car to fabricating a greenhouse, to building pole barns, wiring up IoT widgets to the cloud, building a pig farm, to building big distributed systems, I have made a career of learning how to learn. A career of being a technology generalist. A jack of all trades, **master of some**.

Which leads me to why I am writing this book and donating my time to the Developer Springboard. While my path to where I am now was very unplanned and fragmented, yours certainly doesn't have to be. I have worked at almost all levels of the technology industry. There are so many soft skills that you can learn from my experiences that will immediately be beneficial to you going down the right path. I am very passionate about sharing this with you and learning what I don't yet know in this career management space.

My wife is now cringing at my new hobby for which I have to go deep.

Discover

Discover Who You Are

"This above all: to thine own self be true" – Polonius, from Shakespeare's *Hamlet*

Shakespeare gives good advice, but how can you be true to yourself if you don't know yourself? You would think it would be easy to know ourselves; after all, we are around ourselves all the time. Yet paradoxically, nothing is more difficult. We are too close to ourselves to see objectively.

Fortunately we can get help in discovering who we are, our strengths and weaknesses, our talents and tendencies. We can look to our own experiences first, then to various inventories and tests, and finally to what others have told us about ourselves (good and bad). These three directions of input will help us know ourselves better.

Experience

Look back at your life from childhood to now. Have you been someone who others are naturally attracted to? Did you always rise to be the team captain or leader? Did you enjoy doing that? Were you someone who sought out the outcasts and loners? Did you prefer to be left alone and do your own thing? Are you the life of the party or someone who only feels comfortable with people you know really well? Can you learn quickly and solve problems without much difficulty, or do you have to methodically plod along exploring different routes until you understand the concepts?

What's important here is doing some introspection to think about the big arcs in your personality. Some foundational part of you is

baked into your genetic code and soul, but you have built upon that foundation in unique ways based on your environment and free will. We will call that foundational part your temperament and perhaps your talents in addition. But your *personality* is the sum total of your experiences and choices, your virtues and your faults as well. While you may tend toward fading into the background because your temperament is introverted, you may have chosen and learned to make yourself step out sometimes. Keep your experiences in mind as you go through the next two sections; they will help to confirm or filter your discoveries about yourself.

In The Book

In the book we go through various types of personality inventories that you can take to assess who you are, and who the people are that you work with. We go through temperament, strength finders, disc, and Myers-Briggs. Then we also dig into how to interact with others once you have this information in hand. Listening to others can tell you a lot - if you are so inclined.

Devin's strengths are Relator, Input, Connectedness, and Belief.

After taking the strengths finder and reading about each of these, they all fit me quite well. I was particularly struck by the strength of input. I had always knows I collected things and ideas and people's opinions or perspectives, but never thought it was a strength or had a name. Of course, the dark side of input is analysis paralysis, a fault that I have had to be on guard against for a long time. But that can be monitored and kept in check, and then the strength of input shines, because you are able to take in the best that others have thought about and apply it to your problem.

The First Leg on the Journey

No test or inventory can tell you everything about yourself. These ideas merely help you take the first step toward self-discovery. We are complex creatures full of contradictions, facets, and hidden aspects. Similarly, while others may have valuable insights about you to share, they cannot get inside your heart and truly know what makes you tick. So use all three strategies and triangulate to find the true center point within yourself.

As we move forward with launching your career, these things you have discovered about yourself will help inform the choices you make, the jobs and roles you will be best at, and the way you can maximize your strengths in any situation. The next step in discovering who you are is identifying your core values in life. We will tackle this important subject in the following chapter.

Learning your core values

Your core values are foundational. They are the concepts that you live by on a daily basis. They generally drive your everyday decisions and how you live your life. If you can't lie for any reason – you have honesty buried deep in your soul. If you would drop anything to serve your fellow man you are an overwhelmingly caring person. If you seek money over anything else you might be greedy.

Leadership, and the drive to take charge, is a core value that may guide most of your daily decisions. Be aware that core values are ingrained in you and not likely to change over time. That being said, you can actively target new core values through training. You may want to be a leader and you can take classes to become a better one.

However, other things, like creativity, are hard to learn if you don't already have the knack for it. Understanding the five most consistent core values that you don't waver on can help you choose the industry you want to work in, the type of people you would be most successful working with, and the type of work environment you might consider seeking.

In The Book

In this section we dig through a few core values that will help you in any job you might have through out your career. We discuss what each one is, how it applies to you. And then discuss application in the day to day.

Discover Your Priorities

After you get a good idea of who you are, but before you can set out on a career path, you need to understand what your life's priorities are. We all have different ideas of success, differing motivations, and differing constraints in our life. Maybe you had a child at age 18 and are a single parent. Maybe you are a double-income no kids couple in your late thirties whose only real financial worry is how many vacations you can go on this year. Those circumstances will influence–and possibly even dictate–the career direction you choose.

Success

Ask yourself a simple question: What is your primary measure of success at your work? Do you want to be *wealthy*? Do you want to make the highest salary or consulting rate possible? if this describes you then you want to go where the money is and level yourself up rapidly to get more and more money at each new job.

Or maybe you want to work on projects that make a difference, locally or globally. You want to change the world and do something that matters. If so, then *where* you work will be vitally important. You may be willing to get paid half of your possible salary but work for a non-profit that helps underprivileged children, or that helps police find women being trafficked. The bottom line is that you want to make a big impact on humanity and that is your driving purpose.

I've also met people who want to work on *interesting problems*. They don't care much what it is as long as it is complex and satisfying. They salivate at debugging a multi-threaded deadlock

involving semaphores, mutexes, and thread-local storage. They want to work at one of the few companies that still make compilers because implementing new optimizations sends them into a titter.

Other people desire to become *famous or well-known in the industry*. They want to speak at conferences and have popular blogs. They want to be known as the guy who implemented the awesome thing that everyone uses. They want to write the definitive work on X programming language or Y library. This desire will guide you to find a company or project–even open-source–where you can do something big and make a name for yourself. Nerd adulation calls to you.

A final measure of success is *freedom*. You want to work so that you have the freedom to live your life as you want. Freedom means different things to different people. My brother-in-law is a hotshot fire fighter in New Mexico and California in the summer, then collects unemployment checks and skis all winter. Your idea of freedom may mean not having to clock into a job everyday, or it may mean being able to go on nice vacations whenever you want, or do mission trips to orphanages in Africa. Whatever freedom looks like to you, you want to work to enable you not to work in some significant way.

In The Book

In the book we will discuss how to identify what priorities are important to you. Then we will get into how to set and adhere to your priorities along the way.

Refine

Focus Your Technical Skills

As a programmer seeking to one day be considered an expert in your field, you must decide what technical areas you want to focus on. Focusing on one area means you don't focus on another. It's a reality of life that you need not fear to embrace. The good news is, you get to choose the technical areas that interest you most.

Platforms

Not too long ago, choosing a platform generally meant choosing a desktop operating system to develop on, with Windows being the 800 lb. gorilla. Cross-platform libraries were developed and came to maturity, and a stasis of sorts was reached.

Then the web came into its own and mobile followed, and suddenly the tables were upended again and all development was done in proprietary silos. Now, new cross-platform tools and frameworks are beginning to bridge the gap and make it so that you don't have to commit yourself to choosing just one or two platforms and re-implementing your code multiple times. You now get to choose from a wide area of places to focus your skills.

Do you want to do web-related programming? Client-side or server-side? Do you want to do mobile development? Android or iOS platform? Or maybe you want to use C# and Xamarin and write cross-platform mobile apps across both platforms and Windows Phone as well.

Ask yourself: *where is the world heading?*Â Web and mobile are here to stay, while big desktop applications continue to dwindle

in number and importance. You don't want to be cut adrift on a shrinking iceberg of stale technology that leaves you with irrelevant expertise. If you can make a bet with your skills, bet on web and mobile over desktop.

In The Book

In the book we will discuss how to navigate an ever changing landscape of programming languages and technologies.

Climb the Right Ladder

Choosing a set of technologies to master can be very important, but so can understanding what role your personality and strengths best fit at a particular job. Most companies offer a broad array of technical career roles to chose from. At a high level, do you want to be a programmer or manage programmers? If you want to remain in a hands-on role developing software, do you want to follow the architect or engineer track? There are many ladders for you to climb and you need to choose the one that aligns best with your interests and strengths. Fortunately, you can often make the leap from one ladder to another without too much difficulty. You are not stuck forever on one of them if your desires change over time.

The Technical Climb

Most companies offer a technical track for software developers that may branch into a few sub-tracks as you move along. Typically the levels are engineer, staff engineer, senior engineer, principal, chief, distinguished engineer, and then fellow or CTO. Not all companies have (or need) all these levels, but this categorization is common.

Somewhere up the track, usually at the principal level, the track splits into *architect* vs. *engineer* sub-tracks. Architects excel at high-level design and structure. While they also are often excellent coders and deep in technical knowledge. They are capable of understanding customer needs at a high level and architecting solutions for them. For example, the architect of a large website may be juggling in his or her mind the website design itself, the web back-end, the scaling and deployment stories, and even specific technologies used in each part (database types, buses, messaging

frameworks, and so on).

In The Book

In the book we will look at a story of the student becoming the teacher. We also talk about the rise to managing people instead of writing code.

Where Do You Excel?

As you try out different roles or progress in your career, you should try to identify the conditions under which you work best. This will make for a better overall work experience and increase your likelihood of success. For me, this proved to be extremely useful. You see, after several years of trail and error, I discovered that I am an engineer and not an architect. More importantly, I found that I do best when working under a strong architect or engineering lead. This is why I now look for positions where an existing strong technical leader has already blazed the trail.

Unlike me, you may discover that you excel most when you can do the trail-blazing yourself. You might not like having to follow in other people's footsteps. On the other hand, you may find that you enjoy working closely with other people doing pair programming and test-driven development. All of these are important markers to take note of so you can seek out positions with companies where you can excel. Of course, you need to balance this out with the need to push yourself beyond your comfort zone. Sometimes you'll need to take a role that stretches you outside your boundaries. Doing so can be a catalyst for personal growth.

In the end, the worst outcome isn't failure. Instead, it is having never tried to do something in the first place. As you navigate your career, be mindful of which track you are on and whether it is

the right one for you. Should you consider jumping to a different ladder? Are you making steady progress up your current ladder? Are you finding fulfillment in the technical track you have chosen? These are all questions you should ask yourself regularly as you refine your career goals.

Setting Clear Goals

For the sake of argument, let's say that you've been working as a software developer for a while and you've been able to advance in your career. You've gotten a few raises and maybe a promotion or two. If someone were to ask you how your career is going, what would you answer? Better yet, how would you answer? You might say that it's going well considering that you've been able to advance financially and have been promoted to a more senior position. Is this a good enough answer? Are promotions and raises the only measures of success and failure for our careers? If you've been following the DREAM[2] principles and have taken the time to **DISCOVER** who you are and **REFINE** your goals, then you have a lot more data points at your disposal to answer these kind of questions. Most importantly, you have the career goals that you've established and can use to quantify your success and measure how much progress you have, or have not, made towards them.

If someone were to ask you how your career is going, what would you answer? Better yet, how would you answer?

As I've mentioned before, when I came out of college I didn't really have an idea of what I was going to do with my career. I was not really prepared to be a professional software developer at all. Nothing in my youth or schooling had prepared me for the real world. In fact, for the first few years after school, I just did whatever was in front of me and took every opportunity I was presented with without much regard to career path or any other personal goals.

[2] http://www.developerspringboard.com/vision/

At first glance, you could say that this was working well for me. I was progressing rather quickly at work and I had become pretty good at my job. In the first 5 years or so, I had switched jobs once, recieved several raises and was promoted into a management position. To a certain extent, these minor successes had blinded me to the shortsightedness of my career choices. It took me a while to realize that, although I was advancing, I wasn't really doing well in terms of my career.

You see, the first job I took after college consisted mostly of writing back-end applications in Perl. Then, I switched over to a job working at a small Cold Fusion shop. Neither of these programming languages were what I considered cutting edge and, in fact, I could see that there was no future in them for me. Yes, I was gainfully employed, but I was losing a lot of ground with respect to the changes that were happening in the software development industry. There were new programming languages, tools, technologies and process that were taking off and I wasn't getting any exposure to them. This realization hit me like a ton of bricks. I knew that if I didn't do something about it quickly I would soon become pigeon holed into the niche programmer that I was becoming and it would be really difficult to regain all the ground that I had already lost.

I realized that if I did not set these goals for my career I ran the risk of digging myself into a hole that I would not be able to climb out of.

So, what happened? ...

In The Book

In the book we will finish the story and see how this path was corrected.

Establish

Mastering the Five Keys to a Good Interview

We've all been there many times in our careers–the dreaded interview! We've sat in someone's office nervously answering questions, trying not to say the wrong thing or sound like we don't know what we're talking about.

Interviewing can be a very difficult thing to do well and it is something that doesn't come naturally to most of us. It's another one of those soft skills that we are not taught in school. Instead, we only get better at it after we fumble our way through many nerve racking interviews hoping that we don't do or say something that messes them up completely.

I've probably been interviewed forty times and have interviewed at least one hundred people in my career. There are a few guidelines I have come up with that I've found help make the experience a lot less stressful, both when I interview for a position or when I interview a candidate. I've found that doing these things makes a huge positive impact on the interviewing experience and increases the interviewee's chances of getting the job.

Be Prepared

The first thing you should do is prepare yourself for the interview. This means different things for different industries and even different companies but there are a few things you can do to make sure you are ready to interview with a potential employer.

Most importantly, you should research the company (and the key people that work there) so that you understand what it is they do and make sure that you are addressing the things that are important to them during your interview.

Most importantly, you should research the company (and the key people that work there) so that you understand what it is they do and make sure that you are addressing the things that are important to them during your interview. You want your conversation and answers to be relevant to the interviewer so that they resonate with him/her, the company and the position that you are applying for. A focused answer, that is relevant to the potential employer, will make a bigger impact than a generic one that might speak more to situations and circumstances that were important to previous employers.

Remember, you have a plethora of resources at your disposal. All of the effort you put into learning about the company you are interviewing with will pay off when you're in front of the interviewer answering questions. You should start off by visiting to the company's website and reading through their vision statement, news articles, product offering and "about us"Â pages. These could provide you with a valuable insight into the company's history, the products or services they offer and the key people that work there.

Secondly, you can search the internet for articles that speak about the company, its employees and competitors. This gives you an outside perspective of the company that can help you understand how it fits into theÂ market and/or industry. You can also utilize the network of people around you to help research your potential employer. Of course, this assumes that your job search is not a secret or that there are people in your network that you can trust to help you in confidence. If so, you should ask them if they know of the company you are interviewing with. If they do, any insight they can give you will help focus your interactions with

that company. Maybe they can give you some information aboutÂ their development processes, the technologies that they use and any tools, technologies or processes that they struggle with.

All of this information can be used to focus your conversation and show the interviewer that you've done your homework and are seriously interested in the opportunity. It shows initiative and desire…two things that are extremely valuable in our industry.

In The Book

In the book we will take a look at the other four keys to a good interview: Being honest, being relaxed, being thorough, and being…

Thou Shalt Blog

When I left my first programming job after almost fourteen years there, I couldn'tÂ take any of the code with me. In fact, I couldn't take anything I had done with me. Think about that: you worked for a place for a third of your entire career, and you own none of it.

When I went looking for another job, I had no public presence in the software world. No one knew who I was, nor could I point to a book or blog or even a twitter account that demonstrated I was passionate about software development. I was starting at ground zero in terms of looking for work, with only the skills in my head to help me find something.

Become a Blogger

Blogging establishes your presence. It puts a stake in the ground and declares "I care about the software craft." Care enough to write stuff down about it to share with others.

Your blog is the key part of your overall platform, which may eventually include a facebook page and twitter account, a podcast, books and articles that you write, speaking and training that you do, and so on. But it starts with a website that is your home base on the internet. Here's the cut-to-the-chase guide for programmer's on how to start a blog.

In The Book In the book we will walk you through step by step how to set up a blog. And more importantly how to actually be a blogger. If it were easy everyone would do it. We will show you how to make it easy.

Follow a Mentor to Grow, Become a Mentor to Succeed

If you see anyone that has done much with their life they usually have a plan. And they usually have one or more people they can use to "phone a friend" in a pinch. I can list a great many of these people that have helped me move along in my world. Some directly. Most indirectly. But having someone to follow and coach you is very powerful.

What is a mentor

A mentor is a person or friend who guides a less experienced person by building trust and modeling positive behaviors. An effective mentor understands that his or her role is to be dependable, engaged, authentic, and tuned into the needs of the mentee (yes, that's a word).

In my first real job, where i was hired to do software development full time not just as one of many non-technical responsibilities, I worked with a guy that was a fire fighter AND a perl developer. He had built his own forum in perl and was busy porting that system to another system. But no doubt about it, perl was his thing. He would tell you that rather than learn a new language he wanted to be perfect at perl. At this time we were doing AJAX and Web 2.0

sites but neither of those names existed. .NET was just starting up. I was in ASP, ColdFusion, and SQL Server. JavaScript was not quite yet popular. IE was broken.

This outlook on technology wasn't a positive approach and I didn't look to him to teach me in that way. But he did teach me quite a lot about all the things that sat on the periphery of the systems I would build going forward. At that time I had no idea that there was a role for DBA, and a role for web developer, and a role for IT. See, in that job I was responsible for the membership website. Not just the writing of code, but from the DNS to the database and everything in between. This firefighter was my mentor in that he made sure I didn't focus on just writing code. He taught me to always look at and try to understand the bigger picture. The code is one small part in a very big system.

In The Book

In the book we will finish this story. Then we will take a look at whether or not you should have a mentor and what is required to have a mentor. We will also look at reasons you should have an example to follow. And ultimately we will drive you to become someone's mentor.

Be a Portfolio Manager Not Just an Employee

Mentoring and being mentored is valuable. But while you are doing that, and while your are working at your job, you need to be actively thinking about your career goals and how you can continue to make progress them.

As soon as you do something, log it on your resume. It is easy to remove something from your resume when you need to send less to someone. But it is very difficult to work on a project for a few years and then recall in a split second what it was you did when it is time to polish your resume.

Beyond the Resume

But being a "portfolio manager" isn't just about maintaining a resume is it? If you watch the career of a successful movie star and compare it with a movie star that isn't so successful – what is one of the key differences between them? Picking the right next film to act in. There are actors that take movies as they get them and do them all. This is a hit or miss strategy – but definitely not a plan. Other actors read through a script and are very choosy about the next film they take on. Even if the love the idea of a movie – they will actively turn a script down if it doesn't align with their goals for their career.

When I went head-down at my job for fourteen years, I only updated my resume *one time* during that entire period. And the only reason I did that was because my wife wanted to move to Washington D.C., so I applied for a job with a Microsoft subsidiary out there.

When it came time to really look for a new job, my resume was woefully out of date, and I couldn't remember what all I had worked on over the years. I sketched out some highlights and tried to play up my best work, but I had done a horrible job at managing my own brand.

– Devin

In The Book

In the book we will take a look at what it means to be a portfolio manager, your resume, and your career.

Make Time For Your Projects

What do you do when you don't know how to do something? Or you have something you want to do but don't have the time to do it?

The answer is easy but difficult: you simply do it.

Everyone has twenty-four hours in the day. To accomplish your goals of leveling up with your skills, learning new technologies, or working on side projects that you want to turn into your main income, you must make the time to work on it.

Make Time Stop

Most of us have to work full-time, requiring at least nine hours per day when counting commute time and extra hours. If you have a family to care for, you also need to spend time with them, caring for your children and spending time with them.

In The Book

In the book we will show you how to make time stop. Get control of your time. And some tools to manage your priorities so that you can meet your career goals.

Advance

You've got a job! Perhaps it isn't quite the job you wanted, but this one is yours. Celebrate. Now, let's work on getting you down the path towards your goals. Always be iterating over your goals and where you are. Keep your eyes on the definition of your DREAM career.

In this section, we are going to take a look at how you can maneuver a company in a successful manner, whether it is your first job, or your thirty-first job! We will do this by examining a handful of core values and how you can use them in navigating the job you currently have. Living by your personal core values is the one thing you can control at all times. And if you identified a company that claims to live by a set of core values similar to your own you have won half the battle!

Strive to Always Be Awesome

This sounds like something everyone would automatically think to do. But being awesome is actually hard to do if you don't put any thought into what it means to be awesome. Like most things right? But before we dig into this too much, let's be sure that you clearly understand what I mean by awesome. Awesome is not "saving a file as a pdf". That is now AWE-some.

Awesome is: *extremely impressive or daunting; inspiring great admiration, apprehension, or fear*

So then, a person doing their job merely as expected *can't* be considered awesome. You need to be "extremely impressive and inspiring great admiration". In order to be considered anything, you have to be able to be measured against a certain baseline. AWE-some is at the top. Which means there must be a measureable bottom or middle.

For this reason we will discuss AWE-some in terms of ten points. Like a tip at a restaurant, you start with X. And we will whittle it down as your performance strays away from AWE-some. Measure yourself each day. See how awesome you are today. How about tomorrow? Can you keep ten points most of the time? Track these metrics. Keep a log. See where you fail to be awesome most consistently and then make an action plan to correct that. Now let's look at some ways to measure AWESOME!

Deadlines

Did you miss a deadline? This might happen when you said "I'll look into that". But then you forgot to look into it. Or you looked into it but you never returned a result. Now the person you are working with is being taught that you are unreliable. You are not an effective business partner in their eyes.

Deduct a point!

How might you fix this? I use workflowy, trello, and asana to manage my work. Workflowy allows me to keep a running bulleted list that syncs everywhere for things I am managing. Trello is used to manage projects: tasks that need to be managed across more than just me. And asana is used to manage things that are important to the company I work for. Items with due dates.

Forgot To Give Status

Did your peer, boss, client, or customer have to come to you for a status update? This means you didn't communicate out the status of something that was expected of you. Or you are running late on a deliverable and never communicated that late status. In the end your communications skills are lacking because someone else is being forced to track the work that you should be tracking and reporting on yourself.

Deduct a point!

How might you fix this? Keep a list of touch points that need to be managed. If a client has asked you to do something but it can't be done right this second, tell them when you will get back to them. As soon as you fall behind on a task, tell them. If you are almost done, tell them. Most people don't complain because of over communication.

Anticipating Questions

When working with others, be sure to give an appropriate level of information. When someone has to ask you a question they run the risk of looking stupid. Most people don't want to look stupid in which case they may not ask the question at all. Now, because you didn't provide enough information, and the recipient of the information you did provide left someone with a question, there is potential miscommunication occurring or coming in the future.

Deduct a point!

How might you fix this? Come to meetings prepared. Ensure that you have all the answers to questions. But also offer up more information than is requested. Pause before ending a meeting to think about if there is anything else worth mentioning.

In The Book

In the book we discuss many other points, their importance, and how to fix them where you are currently failing.

Don’t Take Everything Seriously

I am amazed at how so many people live with their serious face on at all times. It isn’t as if we are launching rockets or doing brain surgery!

…if you are launching rockets for a living, keep your serious face on…

In my world, we build line-of-business applications most of the time. These applications may take form in all sorts of different ways. Currently we are working on iOS tablet apps, Web Api apps, web sites, distributed loan processing systems, and distributed property management systems, all with mission critical data behind them. And we have several active projects in the pipeline. I have some level of responsibility for every single one of these projects. And you will very rarely ever catch me without a smile, a joke, and a “Good morning!” Yes, even at lunch time!

In The Book

More in the book!

We were trained to run towards the explosions and bullets and evil doers. We looked forward to walking into the gas chamber to see who tosses their cookies first. A thirty-mile walk was something we did now and then for fun. Jumping out of air planes before they landed was a common occurrence. Or sliding down a fifty foot long fast rope with a hundred pounds of gear from a helicopter to the

top of a multistory building while only being attached to the rope by the strength of your hands. Or navigating the Panama shipping channels in a small rubber boat as fast as possible between shipping vessels that were many stories tall. All of those are moments of real pressure. Not adding some fancy feature to your project by Friday really isn’t the end of the world.

You have no reason to not be happy! Smile constantly! Live out loud in a positive manner.

In The Book

More in the book!

Be happy.

In most cases where your team is feeling pressure, take a breath, take a step back, and review how you got to where you are at this moment. In almost every case where I am working a late night, ordering pizza in to keep my team happy while toiling away on overtime, missing their families while they continue to bang away at their keyboard, I realize that this situation is the result of the three Ps.

Piss Poor Planning

I learned in the military that the three Ps expression was actually just a part of the story!

It is actually a five Ps story: "Piss Poor Planning on your Part shouldn't constitute an emergency on my Part!"

Watch your project every day. If you see it slipping, immediately be a man and step up. Speak with the best interest in mind and help someone take the reins way ahead of time. Everyone will ultimately be grateful.

– Andy

Master

Don’t Believe Your Own Hype

Throughout our careers we’re told that we’re awesome.

Everyone we interact with tells us that we’re magicians making software appear out of nowhere. Every day we perform super human feats of software development. Maybe it’s because we work for people that are not very technically inclined. What we do on a daily basis is nothing short of magic to them. Or maybe it’s because these people were trying to motivate us to do better.

In any case, sooner or later, some of us start believing the hype. We begin to think that we might in fact be some sort of code slinging wizard. Every year, or so, we get a decent raise and a huge bonus. Eventually we start getting promoted: first a senior engineer, then team lead, next principal engineer and eventually architect. This only helps to reinforce the belief that we are in fact the gods of the programming universe.

All of this continues for several years and we keep buying into the idea that we are the awesome-est of the awesome oozing awesome sauce on everything we touch. By this time, our head has grown so much that we can barely squeeze it through the door of our swanky corner office.

In The Book

In the book we will continue this story. One of my favorites.

Living the DREAM

Have you mastered the DREAM already? Do we have nothing more to share with you?

By no means! This is the end of the beginning. On each of these topics we have much more to say (and to hear). Some developers will be far along on all the phases, and others just starting out on Discover. Some may be stuck in Advance, wondering how they can break out of the plateau they've been stuck on for years. Others may be well into the Mastering of all their skills and experience.

Much of what we do as programmers is non-technical. We *think* that we are valued for just our coding ability and technical know-how, but really much more goes into it: our talents, interests, passion, personality, temperament, negotiation tactics, and many more soft skills that you cannot sufficiently learn from any book. But a book can give you the nudge in the right direction, the insight that you didn't or couldn't think of yourself, and if you act on that, you can improve.

Discovering who we are is a life-long journey. Refining our goals and targets likewise stretches across decades, and may change over time as well. Establishing ourselves is also never completely done; more relationships can be made, blog posts written, courses created, and social media reach increased. Advancing in our career is the phase that most of us can spend productive time in across many axes. There are always better ways to conduct ourselves and technical skills to deepen and broaden. And Mastering our career is a phase that we can always set another goal in.

Our goal is to help you reach your goals. We have each learned these lessons the hard way. And we believe that we can help you avoid much of the pain while enjoying the gain that comes from living the programmer DREAM.

So make sure you are following our blog and subscribed to our newsletter at www.developerspringboard.com[3].

We want to hear from you as well. What did we miss that you were hoping we would cover? What frustration are you feeling in your career right now? How can we help you excel and thrive as a developer?

Shoot us a message at programmers@developerspringboard.com[4]

Or if you know exactly who you want to reach, you can find us:

Miguel: miguel@developerspringboard.com[5] Andrew: andrew@developerspringboa
Devin: devin@developerspringboard.com[7]

Thanks for reading and best of luck in your current endeavors!

[3] http://www.developerspringboard.com
[4] mailto:programmers@developerspringboard.com
[5] mailto:miguel@developerspringboard.com
[6] mailto:andrew@developerspringboard.com
[7] mailto:devin@developerspringboard.com