Introduction

This book is about finding fulfillment and happiness in your career. Fulfillment and happiness don't (often) come by chance. They require thought, intention, action, and a willingness to change course when you've made mistakes. This book lays out a strategy for planning and creating a radically successful career (and, therefore, life) in software development.

The book is also about cultivating the desire to live a remarkable life. Strangely, we don't all set out on the quest to lead remarkable lives when we start our careers. Most of us are content to go with the flow. Our expectations have been lowered for us by the media and by our friends, acquaintances, and family members. So, leading a remarkable life is something you have to discover as even being a reasonable goal. It's not obvious.

Most people spend far more of their waking adulthood working than doing anything else. According to a 2006 survey by the U.S. Bureau of Labor Statistics,[1] average Americans spend half of their waking time at work. Leisure and sports are a distant 15 percent of waking time spent. The facts show that our lives basically *are* our work.

If your life is primarily consumed by your work, then loving your work is one of the most important keys to loving your *life*. Challenging, motivating, rewarding work is more likely to make you want to get up in the morning than dull, average tasks. Doing your job well means that the activity you do for 50 percent of your available time is something you're good at. Conversely, if you don't do your job well, a large amount of your time will be spent feeling inadequate or guilty over not performing at your best.

1. http://www.bls.gov/tus/charts/

Ultimately, we're all looking for happiness. Once we have our basic human needs like food and shelter taken care of, most of our goals are geared toward finding happiness. Sadly, our *activities* are often mismatched to that one overarching goal. This is because we as humans get bogged down in the means and forget about the end.

I might be happier if I had more money. I might be happier if I got more and better recognition for my accomplishments. I might be happier if I were promoted in my company or I became famous. But what if I were poor and had a trivial job but I was really happy? Is that possible? If it were, should I be looking for more money? Or a better job?

Maybe not. What's certain is that, with the focused goal of happiness as a primary motivator, we can make better decisions about the smaller steps we take to achieve that goal. A higher salary might actually be desirable and lead toward happiness. But if you take your eyes off the primary goal, you can find yourself driving toward a higher salary at the *expense* of your happiness. It sounds ridiculous, but I've done it. And so have you. Think about it.

Throughout this book, I'm going to give you advice that I hope will lead you to a happier and more rewarding career (and thereby to a happier life). You might make more money if you follow this advice. You might gain more recognition or even become famous. But please don't forget that these are not the goals. They're a means to an end.

Failure Is Off the Radar!

One of the major steps along the road to creating a remarkable career for myself was, ironically, writing the first edition of this book. The book used to be called *My Job Went to India (And All I Got Was This Lousy Book): 52 Ways to Save Your Job*. It had a picture on the cover of a guy holding a sign that said "Will Code for Food." It was funny, and its title and shocking red cover were meant to play on the Western world's fears that their jobs were going to be outsourced to low-cost offshore programming teams.

The problem, though, is that it painted the wrong picture. The truth of the matter is, if you need to "save" your job, I can't help you. This book isn't about struggling to maintain the level of mediocrity required not to get fired. It's about being *awesome*. It's about *winning*. You don't win

a race by trying not to lose. And you don't win at life by trying not to suck. Fortunately, the content of the book has never been about trying not to suck. I can't think that way, and neither should you.

I remember the exact moment when I decided that my career would be remarkable. I'd been coasting through jobs sort of how I coasted through high school, the part of college I finished, and my brief and somewhat mediocre career as a professional saxophonist. Because of some combination of luck and natural talent, I managed to come across a healthy amount of success along the way—enough that it landed me a well-paying job as a respected member of the technical staff of one of the world's "most admired" companies. But I was just *getting by*, and I knew it.

One evening after work, while browsing through the local bookstore, I came across Kent Beck's *Extreme Programming Explained* [Bec00] on the new releases shelf. The subtitle of the book was *Embrace Change*. The idea of change has always been appealing to me. I have a tiny attention span that had, up until that point, manifested itself as a series of fast job changes—hopping from one company to the next. The idea of a "software development methodology" sounded atrociously boring and management-tinged, but I figured if it involved lots of change, it might be something I could push at work to avoid getting bored and feeling like I needed to find a new job.

Picking up this book turned out to be a really lucky whim. I started reading the book, and I couldn't put it down. After devouring its content, I hit the Internet and read everything I could about the ideas of Extreme Programming (XP). I was sufficiently moved by those ideas that I went to our chief information officer and attempted to sell him on the idea. He and his staff were convinced, and as part of the Extreme Programming adoption deal, he sent a large group of us to Object Mentor's Extreme Programming Immersion course.

Extreme Programming Immersion was *the* place to go if you wanted to learn about XP. It was like getting a backstage pass to a weeklong concert put on by our favorite rock stars. Being in that room with those people actually made me a *lot* smarter. It made me more creative. And, when it was over, I was really, really sad. I couldn't imagine going back to my cubicle and beating my head against the mediocrity I had grown accustomed to at work.

My co-worker Steve, who contributed the essay you'll find on page 177, and I came to the same conclusion. The only way to find yourself around *those* people as often as possible is to *become* one of those people. In other words, if I wanted to be around people who brought me up a level or two when I interacted with them, there wasn't a company I could apply to work at or a college course I could sign up for. I just had to identify what it meant to be one of those people and do what it took. So, I announced to Steve that I was going to become one of those people.

That was *the* turning point of my career. I somehow forgot it until years later when Steve reminded me of the conversation. I had told him about the fact that I had, for the first time, been invited to give a keynote speech at a conference. I was blown away that anyone would ask *me* of all people to not only speak but to deliver one of the main addresses to a software conference. I had indeed become one of those people I had aspired to become.

I did all of this without a formal education in computer programming. I was a musician before becoming a computer programmer. I went to college to study music. Since musicians don't benefit much from college degrees, I chose to avoid any class that didn't help me be a better musician. This means I left the university with more credits than required for any degree but still a few years worth of actual class time before I could graduate. In that way, I'm unqualified to be a professional software developer—at least if you look at the typical requirements for a software engineering position on the job market.

But, though I'm unqualified to be a typical software developer, my background as a musician gave me one key insight that ultimately allowed me to skip the step of being a typical software developer (who wants to be typical, anyway?). *Nobody* becomes a musician because they want to get a job and lead a stable and comfortable life. The music industry is too cruel an environment for this to be a feasible plan. People who become professional musicians *all* want to be *great*. At least when starting out, greatness is binary in the music world. A musician wants to either be great (and famous for it!) or not do it at all.

I'm often asked why it is that there are so many good musicians who are also good software developers. That's the reason. It's not because the brain functions are the same or that they're both detail-oriented or

both require creativity. It's because a person who wants to be great is *far* more likely to become great than someone who just wants to do their job. And even if we can't all be Martin Fowler, Linus Torvalds, or the Pragmatic Programmers, setting a high target makes it likely that we'll at least land somewhere far above average.

You Own It

Most people follow everyone else's plan but their own. To start differentiating yourself, all you have to do is stop and take a good look at your career. You need to be following *your* plan for you—not theirs.

How do you come up with this plan? Software is a business. As software developers, we are businesspeople. Our companies don't employ us because they love us. They never have, and they never will. That's not the job of a business. Businesses don't exist so we can have a place to go every day. The purpose of a business is to make money. To excel at a company, you're going to have to understand how you fit into the business's plan to make money.

As we'll explore later, keeping you employed costs your company a significant amount of money. Your company is *investing* in you. Your challenge is to become an obviously good investment. You will start to judge your own performance in terms of the business value you bring to the organization or customer who is employing you.

Think of your career as if it is the life cycle of a product that you are creating. That product is made up of you and your skills. In this book, we'll look at four facets that a business must focus on when designing, manufacturing, and selling a product. And we'll see how these four facets can be applied to our careers:

- **Choose your market.** Pick the technologies and the business domains you focus on consciously and deliberately. How do you balance risk and reward? How do supply and demand factor into the decision?
- **Invest in your product.** Your knowledge and skills are the cornerstone of your product. Properly investing in them is a critical part of making yourself marketable. Simply knowing how to program in Visual Basic or Java isn't good enough anymore. What other skills might you need in the new economy?

- **Execute.** Simply having employees with a strong set of skills does not pay off for a company. The employees have to *deliver*. How do you keep up the delivery pace without driving yourself into the dirt? How do you know you're delivering the *right* value for the company?
- **Market!** The best product in history will not actually get purchased if nobody knows it exists. How do you get find recognition in both your company and the industry as a whole without "sucking up"?

New Edition

This book is a second edition of the book originally titled *My Job Went to India (And All I Got Was This Lousy Book): 52 Ways to Save Your Job*. The goal of the second edition was to focus more closely on what the original book's real intent was: to create a remarkable career. In doing so, I not only created a new, more positive title, but I added new content as well.

David Heinemeir Hansson, the creator of Ruby on Rails and partner in 37signals, contributed a new foreword.

Each section contains one or more essays written by people I've encountered or worked with whose careers are truly remarkable. The essays provide insights into the decisions these innovators, developers, managers, and entrepreneurs have made along the path to success. They also underscore the fact that the techniques outlined here aren't just idealistic suggestions applicable only in a perfect environment. They're real things that real people can do and accomplish.

Some of the original tips have been removed, and several new tips have been added. The entire last section from the original, called "If You Can't Beat 'Em" was removed. New tips were added throughout the book that reflect new lessons I've learned since the first edition was published.

Some new "Act on It" sections have been added to tips held over from the previous edition.

This introduction and the ending have been replaced to reflect the book's clearer focus on the goal of a remarkable career.

The goal of this book is to give you a systematic way of building a remarkable career in software development. We will walk through specific examples and present a set of actions that you can take *right now* that will have both short-term and long-term positive effects.

And, like I said before, we're not going to talk about how to save your job. If you currently find yourself feeling afraid about losing your job, the steps you'll take to build a remarkable career will remove that fear. Remarkable software developers don't languish. They don't find themselves fruitlessly searching for work. So, don't worry. Stay focused on winning, and the fear of losing will be forever a memory.