
PRE-REQUISITE INTRODUCTION

(Don't skip this, you're going to need it.)



I presume you just picked up this book because you are a computer programmer and are intrigued by the notion of professionalism. You should be. Professionalism is something that our profession is in dire need of.

I'm a programmer too. I've been a programmer for 42¹ years; and in that time—*let me tell you*—I've seen it all. I've been fired. I've been lauded. I've been a team leader, a manager, a grunt, and even a CEO. I've worked with brilliant

1. Don't Panic.

programmers and I've worked with slugs.² I've worked on high-tech cutting-edge embedded software/hardware systems, and I've worked on corporate payroll systems. I've programmed in COBOL, FORTRAN, BAL, PDP-8, PDP-11, C, C++, Java, Ruby, Smalltalk, and a plethora of other languages and systems. I've worked with untrustworthy paycheck thieves, and I've worked with consummate professionals. It is that last classification that is the topic of this book.

In the pages of this book I will try to define what it means to be a professional programmer. I will describe the attitudes, disciplines, and actions that I consider to be essentially professional.

How do I know what these attitudes, disciplines, and actions are? Because I had to learn them the hard way. You see, when I got my first job as a programmer, professional was the last word you'd have used to describe me.

The year was 1969. I was 17. My father had badgered a local business named ASC into hiring me as a temporary part-time programmer. (Yes, my father could do things like that. I once watched him walk out in front of a speeding car with his hand out commanding it to "Stop!" The car stopped. Nobody said "no" to my Dad.) The company put me to work in the room where all the IBM computer manuals were kept. They had me put years and years of updates into the manuals. It was here that I first saw the phrase: "This page intentionally left blank."

After a couple of days of updating manuals, my supervisor asked me to write a simple Easycoder³ program. I was thrilled to be asked. I'd never written a program for a real computer before. I had, however, inhaled the Autocoder books, and had a vague notion of how to begin.

The program was simply to read records from a tape, and replace the IDs of those records with new IDs. The new IDs started at 1 and were incremented by

2. A technical term of unknown origins.

3. Easycoder was the assembler for the Honeywell H200 computer, which was similar to Autocoder for the IBM 1401 computer.

1 for each new record. The records with the new IDs were to be written to a new tape.

My supervisor showed me a shelf that held many stacks of red and blue punched cards. Imagine that you bought 50 decks of playing cards, 25 red decks, and 25 blue decks. Then you stacked those decks one on top of the other. That's what these stacks of cards looked like. They were striped red and blue, and the stripes were about 200 cards each. Each one of those stripes contained the source code for the subroutine library that the programmers typically used. Programmers would simply take the top deck off the stack, making sure that they took nothing but red or blue cards, and then put that at the end of their program deck.

I wrote my program on some coding forms. Coding forms were large rectangular sheets of paper divided into 25 lines and 80 columns. Each line represented one card. You wrote your program on the coding form using block capital letters and a #2 pencil. In the last 6 columns of each line you wrote a sequence number with that #2 pencil. Typically you incremented the sequence number by 10 so that you could insert cards later.

The coding form went to the key punchers. This company had several dozen women who took coding forms from a big in-basket, and then "typed" them into key-punch machines. These machines were a lot like typewriters, except that the characters were punched into cards instead of printed on paper.

The next day the keypunchers returned my program to me by inter-office mail. My small deck of punched cards was wrapped up by my coding forms and a rubber band. I looked over the cards for keypunch errors. There weren't any. So then I put the subroutine library deck on the end of my program deck, and then took the deck upstairs to the computer operators.

The computers were behind locked doors in an environmentally controlled room with a raised floor (for all the cables). I knocked on the door and an operator austerey took my deck from me and put it into another in-basket inside the computer room. When they got around to it, they would run my deck.

The next day I got my deck back. It was wrapped in a listing of the results of the run and kept together with a rubber band. (We used *lots* of rubber bands in those days!)

I opened the listing and saw that my compile had failed. The error messages in the listing were very difficult for me to understand, so I took it to my supervisor. He looked it over, mumbled under his breath, made some quick notes on the listing, grabbed my deck and then told me to follow him.

He took me up to the keypunch room and sat at a vacant keypunch machine. One by one he corrected the cards that were in error, and added one or two other cards. He quickly explained what he was doing, but it all went by like a flash.

He took the new deck up to the computer room and knocked at the door. He said some magic words to one of the operators, and then walked into the computer room behind him. He beckoned for me to follow. The operator set up the tape drives and loaded the deck while we watched. The tapes spun, the printer chattered, and then it was over. The program had worked.

The next day my supervisor thanked me for my help, and terminated my employment. Apparently ASC didn't feel they had the time to nurture a 17-year-old.

But my connection with ASC was hardly over. A few months later I got a full-time second-shift job at ASC operating off-line printers. These printers printed junk mail from print images that were stored on tape. My job was to load the printers with paper, load the tapes into the tape drives, fix paper jams, and otherwise just watch the machines work.

The year was 1970. College was not an option for me, nor did it hold any particular enticements. The Viet Nam war was still raging, and the campuses were chaotic. I had continued to inhale books on COBOL, Fortran, PL/1, PDP-8, and IBM 360 Assembler. My intent was to bypass school and drive as hard as I could to get a job programming.

Twelve months later I achieved that goal. I was promoted to a full-time programmer at ASC. I, and two of my good friends, Richard and Tim, also 19, worked with a team of three other programmers writing a real-time accounting system for a teamster's union. The machine was a Varian 620i. It was a simple mini-computer similar in architecture to a PDP-8 except that it had a 16-bit word and two registers. The language was assembler.

We wrote every line of code in that system. And I mean *every* line. We wrote the operating system, the interrupt heads, the IO drivers, the *file system* for the disks, the overlay swapper, and even the relocatable linker. Not to mention all the application code. We wrote all this in 8 months working 70 and 80 hours a week to meet a hellish deadline. My salary was \$7,200 per year.

We delivered that system. And then we quit.

We quit suddenly, and with malice. You see, after all that work, and after having delivered a successful system, the company gave us a 2% raise. We felt cheated and abused. Several of us got jobs elsewhere and simply resigned.

I, however, took a different, and very unfortunate, approach. I and a buddy stormed into the boss' office and quit together rather loudly. This was emotionally very satisfying—for a day.

The next day it hit me that I did not have a job. I was 19, unemployed, with no degree. I interviewed for a few programming positions, but those interviews did not go well. So I worked in my brother-in-law's lawnmower repair shop for four months. Unfortunately I was a lousy lawnmower repairman. He eventually had to let me go. I fell into a nasty funk.

I stayed up till 3 AM every night eating pizza and watching old monster movies on my parents' old black-and-white, rabbit-ear TV. Only some of the ghosts where characters in the movies. I stayed in bed till 1 PM because I didn't want to face my dreary days. I took a calculus course at a local community college and failed it. I was a wreck.

My mother took me aside and told me that my life was a mess, and that I had been an idiot for quitting without having a new job, and for quitting so emotionally, and for quitting together with my buddy. She told me that you never quit without having a new job, and you always quit calmly, coolly, and alone. She told me that I should call my old boss and beg for my old job back. She said, “You need to eat some humble pie.”

Nineteen-year-old boys are not known for their appetite for humble pie, and I was no exception. But the circumstances had taken their toll on my pride. In the end I called my boss and took a big bite of that humble pie. And it worked. He was happy to re-hire me for \$6,800 per year, and I was happy to take it.

I spent another eighteen months working there, watching my Ps and Qs and trying to be as valuable an employee as I could. I was rewarded with promotions and raises, and a regular paycheck. Life was good. When I left that company, it was on good terms, and with an offer for a better job in my pocket.

You might think that I had learned my lesson; that I was now a professional. Far from it. That was just the first of many lessons I needed to learn. In the coming years I would be fired from one job for carelessly missing critical dates, and nearly fired from still another for inadvertently leaking confidential information to a customer. I would take the lead on a doomed project and ride it into the ground without calling for the help I knew I needed. I would aggressively defend my technical decisions even though they flew in the face of the customers’ needs. I would hire one wholly unqualified person, saddling my employer with a huge liability to deal with. And worst of all, I would get two other people fired because of my inability to lead.

So think of this book as a catalog of my own errors, a blotter of my own crimes, and a set of guidelines for you to avoid walking in my early shoes.