Extracted from:

New Programmer's Survival Manual

Navigate Your Workplace, Cube Farm, or Startup

This PDF file contains pages extracted from *New Programmer's Survival Manual*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit http://www.pragprog.com .

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printer versions; the content is otherwise identical.

Copyright © 2010 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

The Pragmatic Programmers

New Programmer's Survival Manual

Navigate Your Workplace, Cube Farm, or Startup

Josh Carter

Edited by Susannah Davidson Pfalzer

New Programmer's Survival Manual

Navigate Your Workplace, Cube Farm, or Startup

Josh Carter

The Pragmatic Bookshelf
Dallas, Texas • Raleigh, North Carolina

Pragmatic Bookshelf

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic courses, workshops, and other products can help you and your team create better software and have more fun. For more information, as well as the latest Pragmatic titles, please visit us at http://pragprog.com.

The team that produced this book includes:

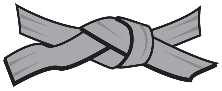
Susannah Pfalzer (editor)
Potomac Indexing, LLC (indexer)
Kim Wimpsett (copyeditor)
David J Kelly (typesetter)
Janet Furlow (producer)
Juliet Benda (rights)
Ellie Callahan (support)

Copyright © 2011 Pragmatic Programmers, LLC.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

Printed in the United States of America.
ISBN-13: 978-1-934356-81-4
Printed on acid-free paper.
Book version: P1.0—November 2011

For Daria and Genevieve.



[*Brown Belt*] You need to start with simply doing your job well. Then you move onto getting noticed for it.

Whatever role you have, you need to establish a positive reputation. That reputation magnifies the *visibility* of your role. Visibility is when people around the office know your name—"I've heard of Emma; she does awesome work." Not only does it stroke your ego, but it gives you influence to get the projects and roles you want down the road.

Visibility doesn't require a fancy job title; it's far more subtle. You, as a lowly programmer on the bottom of the org chart, can have visibility all the way up to the CEO. If you're working on a project the CEO has personal interest in, you have visibility. You could have stumbled into it by accident; this happened to me when I overhauled the user interface graphics for a product I was working on. The graphics were simply a hobby, not really part of my job, but the CEO *loved* the new look, and I was *in*.

You don't always get visibility by accident, yet pushing for visibility can backfire. It's something of a zen thing—people who overtly seek it come off looking like phonies. You know that guy in meetings who's always piping up just so he gets a word in and sounds smart, but really he sounds like a meathead instead? Don't be that guy.

The better approach is to let your work do the talking. First, while you're new in your programming job, strive to get some early wins. If you have any say in the matter, take on some tasks that you *know* you can deliver quickly and solidly. Follow the company coding style perfectly, write unit tests that prove functionality, and make it *really* good and deliver fast.

Your manager will notice, and he'll go bragging to *his* manager about making a good hire. "Our new programmer Emma has been here only a month, and her code went into test with zero defects. Clearly I'm a genius for hiring her." (OK, that's a little bit of an exaggeration, but only a little.)

Second, make a mark on the product where others will notice. Let's say you get stuck on bug patrol—a common

Don’t Use Duct Tape, Velcro Looks Better

The year was 1996, well before cell phones did anything more than make phone calls, and a company called Metricom was selling a radio modem with inexpensive data service. I bought one and started poking at it; it turns out its serial interface was dead simple.

I was working on the networking stack for our company’s handheld computer. It had a phone jack on it for connecting to a dial-up ISP. (Back then, dial-up was the norm.) I looked at our computer, looked at the radio modem, and it was blindingly obvious that these two were made for each other. So, I got some Velcro and stuck the two together.

It didn’t take long to attract attention. *Everyone* in the company wanted to check it out. At a conference a couple months later I wrote a small web server application and passed it around —people could edit a web page on-screen, and others could browse to it in real time. It was a hit.

If there’s one thing geeks can’t resist, it’s plugging stuff together. In this case, our handheld computer just *begged* to be untethered from the phone line, so even nongeeks could appreciate the coolness of this combo. I was still a lowly programmer, but everyone in the company knew my name from then on.

Industry Perspective: Making an Impression

The most important thing you as a new/freshman programmer can do is keep your head down and be *certain* you’re doing the task at hand. Any opinion, idea, or suggestion you may have regarding the workflow of the company, or the product you are on, should be withheld at least until after the first review cycle.

Any person straight out of school wants to make a big impression; you want to show the company that you’re worth your salt and said company is full of geniuses for hiring you. And that’s great. The problem is that trying to make this impression can throw you into over-reaction mode, especially if you have a highly competitive personality.

New employees can (and do) have great, fresh ideas. The problem is you don’t know where the corporate land mines are, you don’t yet have awareness of what factions are competing within the corporate climate, and you don’t have a good enough feel for the corporate culture.

Putting it another way, what gets you a raise and kudos at Google may well get you fired at Netflix.

> Nothing, and I mean *nothing*, speaks louder than doing the job you're assigned to the absolute best you possibly can.
>
> *—Mark "The Red" Harlan, engineering manager*

new-hire task, purportedly to help you "learn the code base," but really it's because the other programmers just don't want to fix the bugs themselves. Your first task is to fix some GUI text fields that don't validate correctly. Do that and check it in, and then go through and make sure the GUI widgets line up right—do some visual housekeeping that makes the interface look nicer than you found it. "Emma went through and polished this part of the GUI, and *wow*, it looks a lot better!"

Finally, as you get more credibility in the group and freedom to pick your tasks, pick some things that people in the company are passionate about. It could be as simple as tying a couple features together in a novel way, like adding an email gateway to a web application. "Emma made it so I can email an update to the tracking system without having to log in—what a great idea!" There's always *some* itch that people have wanted to scratch but haven't put in the effort.

Visibility may sound like shameless self-promotion. Yes, to a degree it is. But when the next job opening comes up or the next cool project is getting started, don't you want a shot? If your name has been circulating among the managers because of something cool you did, you're much more likely to get that shot.

And let's be honest, programmers love to build cool stuff. When you've built something cool, it's really fun to show it off. So, *show it off* and enjoy your moment in the spotlight.

Actions

Consider the work you have on your plate. Are there opportunities for early wins? Grab some tasks and truly nail them.

Looking ahead, can you make a dent in your product in a way that others will notice? It may be a shiny new feature, but it could also be fixing an annoying bug, too.