The Pragmatic Programmer

from journeyman
to master

Andrew Hunt
David Thomas

What others in the trenches say about *The Pragmatic Programmer*. . .

"The cool thing about this book is that it's great for keeping the programming process fresh. *[The book]* helps you to continue to grow and clearly comes from people who have been there."

► **Kent Beck,** author of *Extreme Programming Explained: Embrace Change*

"I found this book to be a great mix of solid advice and wonderful analogies!"

► **Martin Fowler,** author of *Refactoring* and *UML Distilled*

"I would buy a copy, read it twice, then tell all my colleagues to run out and grab a copy. This is a book I would never loan because I would worry about it being lost."

► **Kevin Ruland,** Management Science, MSG-Logistics

"The wisdom and practical experience of the authors is obvious. The topics presented are relevant and useful. . . . By far its greatest strength for me has been the outstanding analogies—tracer bullets, broken windows, and the fabulous helicopter-based explanation of the need for orthogonality, especially in a crisis situation. I have little doubt that this book will eventually become an excellent source of useful information for journeymen programmers and expert mentors alike."

► **John Lakos,** author of *Large-Scale C++ Software Design*

Pragmatic Programmer, The: From Journeyman to Master, Portable Documents

Table of Contents

CONTENTS

FOREWORD

PREFACE

1 A PRAGMATIC PHILOSOPHY

1. The Cat Ate My Source Code
2. Software Entropy
3. Stone Soup and Boiled Frogs
4. Good-Enough Software
5. Your Knowledge Portfolio
6. Communicate!

2 A PRAGMATIC APPROACH

7. The Evils of Duplication
8. Orthogonality
9. Reversibility
10. Tracer Bullets
11. Prototypes and Post-it Notes
12. Domain Languages
13. Estimating

3 THE BASIC TOOLS

14. The Power of Plain Text
15. Shell Games

Table of Contents

16. Power Editing
17. Source Code Control
18. Debugging
19. Text Manipulation
20. Code Generators

4 PRAGMATIC PARANOIA

21. Design by Contract
22. Dead Programs Tell No Lies
23. Assertive Programming
24. When to Use Exceptions
25. How to Balance Resources

5 BEND, OR BREAK

26. Decoupling and the Law of Demeter
27. Metaprogramming
28. Temporal Coupling
29. Its Just a View
30. Blackboards

6 WHILE YOU ARE CODING

31. Programming by Coincidence
32. Algorithm Speed
33. Refactoring
34. Code Thats Easy to Test
35. Evil Wizards

7 BEFORE THE PROJECT

36. The Requirements Pit
37. Solving Impossible Puzzles
38. Not Until Youre Ready

Table of Contents

39. The Specification Trap
40. Circles and Arrows

8 PRAGMATIC PROJECTS

41. Pragmatic Teams
42. Ubiquitous Automation
43. Ruthless Testing
44. Its All Writing
45. Great Expectations
46. Pride and Prejudice

Appendices

A: RESOURCES

Professional Societies
Building a Library
Internet Resources
Bibliography

B: ANSWERS TO EXERCISES

INDEX