

## Partial Contents

- Chapter 1: What Does It Mean to Be Object-Oriented, Anyway?
  - 1.1 Encapsulation
  - 1.2 Information/Implementation Hiding
  - 1.3 State Retention
  - 1.4 Object Identity
  - 1.5 Messages
  - 1.6 Classes
  - 1.7 Inheritance
  - 1.8 Polymorphism
  - 1.9 Genericity
- Chapter 2: Object-Orientation—Who Ordered That?
- Chapter 3: The Basic Notation for Classes and Methods
- Chapter 4: Inheritance and Aggregation Diagrams
- Chapter 5: The Object-Communication Diagram
- Chapter 6: State-Transition Diagrams
- Chapter 7: Additional OODN Diagrams
- Chapter 8: Encapsulation and Concurrence
- Chapter 9: Domains, Encumbrance, and Cohesion
- Chapter 10: Properties of Classes and Subclasses
- Chapter 11: The Perils of Inheritance and Polymorphism
- Chapter 12: Class Interfaces
- Appendix A: Checklist for an Object-Oriented Design Walkthrough
- Appendix B: The Object-Oriented Design Owner's Manual
- Appendix C: Blitz Guide to Object-Oriented Terminology

## What Every Programmer Should Know . . .

"Beautifully written introduction to the fundamental ideas, notation, terminology, criteria, and principles of object-oriented design. Written for every programmer and, indeed, a must-read for every programmer involved with object-oriented programming. A classic programmer's text, delightfully brimming with clarity and elegance."

—Carlene Char, *Computer Book Review*

"Regardless of your level of expertise, the principles will undoubtedly spark more than one discovery. They are Page-Jones's lasting contribution to our collective understanding of object orientation and software development in general.

" . . . surprisingly simple, elegant, and revealing . . . Rest assured that the book's explanations will make your job as a designer and programmer easier and more satisfying.

" . . . fresh breeze across the object-oriented landscape."

—Andrew J. Wozniwicz, *Windows Tech Journal*

" . . . vintage Page-Jones . . . Meilir is a gifted teacher. . . a relentlessly pragmatic focus based on real-world experience . . . compact examples to guide the developer, whether novice or old hand, toward better object-oriented software solutions . . .

" . . . Who could ask for more?"

—Larry L. Constantine, *from the foreword*

"Some programmers don't think they're doing design when they program, but whenever you write code, you're always doing design, either explicitly or implicitly. One of the aims of this book is to make programmers explicitly aware of the design patterns that they're creating in their code."

—Meilir Page-Jones, *DHQ*, Vol. V, No. 2

## About the Author



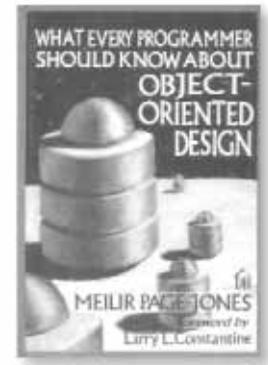
Meilir Page-Jones is president and senior consulting methodologist at Wayland Systems, and has extensive experience in training, consulting, and hands-on system development. He is also the author of *Practical Project Management*, published by Dorset House. Visit [www.waysys.com](http://www.waysys.com).

ORIGINAL EDITION

# What Every Programmer Should Know About Object-Oriented Design

by Meilir Page-Jones  
foreword by Larry L. Constantine

Winner of a Software Development  
Productivity Award



ISBN: 0-932633-31-5  
©1995 392 pages hardcover  
\$50.95 (incl. \$6.00 for UPS in US)

With exceptional clarity, Page-Jones teaches the fundamentals of object-oriented design in a language-independent way. Techniques for designing and building with objects are illustrated with abundant examples, elaborated with do's and don'ts discussions, and reinforced with meaningful exercises and answers.

Revised for UML in a second edition—entitled *Fundamentals of Object-Oriented Design in UML*—this award-winning first edition remains one of the most understandable books on O.O. design.

Programmers, designers, and managers who want to learn object-oriented design without learning UML will especially find this book useful. The diagrams are presented in the author's OODN notation and the brief code examples are given in an easily recognizable hybrid of C++, Smalltalk, and Eiffel.

The chief concern of the text is design—not notation. "Some programmers don't think they're doing design when they program," the author comments, "but whenever you write code, you're always doing design. . . ."

If you haven't joined the UML bandwagon—or if you're ready to jump off!—explore this experienced teacher's lessons on object-oriented design with this highly acclaimed, readable text.

"illustrates exactly what the criteria are for good objects. It also reveals that the criteria are much more relevant than the coding language, diagramming techniques, or object methodology for determining system goodness and object success. . . . a book for developers who are sure they have already figured out everything there is to know about objects."

—Roland Racko, *Software Development*

Read more about this book at  
<http://www.dorsethouse.com/books/weps.html>