OBJECT-ORIENTED PROGRAMMING: AN INTRODUCTION

OBJECT-ORIENTED PROGRAMMING: AN INTRODUCTION

Greg Voss

Osborne McGraw-Hill
Berkeley New York St. Louis San Francisco
Auckland Bogotá Hamburg London Madrid
Mexico City Milan Montreal New Delhi Panama City
Paris São Paulo Singapore Sydney
Tokyo Toronto

9350153

Osborne McGraw-Hill
2600 Tenth Street
Berkeley, California 94710
U.S.A.

For information on translations or book distributors outside of the U.S.A., please write to Osborne McGraw-Hill at the above address.

Object-Oriented Programming: An Introduction

Copyright © 1991 by McGraw-Hill, Inc. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

34567890 DOC 998765432

ISBN 0-07-881682-3

Figure 1-1, "Evolution of structure in architecture," is a computer-generated drawing after Fig. 0-16 in *Art Through the Ages, Fourth Edition,* by Helen Gardner (New York: Harcourt, Brace & World, Inc., 1959).

Part opener art is by Janie Wooldridge.

Information has been obtained by Osborne McGraw-Hill from sources believed to be reliable. However, because of the possibility of human or mechanical error by our sources, Osborne McGraw-Hill, or others, Osborne McGraw-Hill does not guarantee the accuracy, adequacy, or completeness of any information and is not responsible for any errors or omissions or the results obtained from use of such information.

039012 93

Publisher

Kenna S. Wood

Acquisitions Editor

Jeffrey Pepper

Associate Editor

Vicki Van Ausdall

Project Editor

Judith Brown

Copy Editor

Paul Medoff

Proofreading Coordinators

Bob O'Keefe
Ann Krueger Spivack
Kelly Barr

Proofreaders

K.D. Sullivan
Mick Arellano

Artists

Janie Wooldridge
Greg Allen

Indexer

Sharon Hilgenberg

Computer Designers

Judy Wohlfrom
Mickey Salinaro

Cover Design

Bay Graphics Design, Inc.

ACKNOWLEDGMENTS

As always, the more creative ideas must be credited to Melissa Milich, who has an eye and an ear for the funny things that most of us never see or hear. Melissa also edited, proofed, and typed thousands of strange little words—without complaint—and kept me laughing through it all. She never ceases to amaze me.

Thanks to the folks at Osborne: Judith Brown for never-ending patience as production editor; Vicki Van Ausdall and Emily Rader for manuscript preparation and revision; Ann Pharr for keeping us all in touch; and Jeff Pepper for keeping me on target and giving me rope when I needed it.

Bruce Clift played a critical role as developmental editor and Microsoft Windows guru. Not only did he read and critique several drafts of the manuscript, Bruce translated most of the ObjectWindows example programs for Chapter 14 into C++. Jeff Hsu read every word and program of a manuscript that grew to astronomical proportions before his very eyes. He has no idea how much I depended on his technical commentary and review. Readers can thank Paul Medoff for improved readability of the text and elimination of many errors that would have resulted in frustration and misunderstanding.

I thank Janie Wooldridge for her wonderful drawings and Judith Brown for insisting that they had a place in this book. Greg Allen lightened a heavy discussion of encapsulation with his slovenly cook cartoon.

此为试读，需要完整PDF请访问：www.ertongbook.com

Not all of the contributions to this book were technical. Cindy Borjon made sure the manuscripts got to Osborne on time. Priscilla kept me from swearing and made sure I got my exercise. Once again David, Martha, and the little Intersimones opened their home to us in time of need. I thank members of the Santa Cruz City Council and municipal offices for their support. Louis Rittenhouse and Sharon Saldavia were knights in shining armor.

Most of all I would like to thank the Santa Cruz Police Department. Without their protection and professional services this book would never have been completed. They gave us the spirit and backbone to stand up to a nightmare. Particularly I want to thank Mike Dunbaugh, Jeanne Morningstarr, Randy Harris, Todd Dickson, and the undercover officers who helped a neighborhood say no to drugs. Chief Bassett, you have reason to be proud.

PREFACE

This book introduces the concepts of object-oriented programming using a variety of languages. The principles of object-oriented programming are language independent. In fact, object-oriented programming can be practiced, although awkwardly, in traditional languages such as ANSI C or standard Pascal. This book, however, uses only languages that provide special features designed to make object-oriented programming easy, natural, and reliable. The languages covered include C++, object-oriented Pascal, Actor, and Smalltalk.

GOALS OF THE BOOK

A great deal of confusion has resulted from inadequate explanations of object-oriented programming. Programming language features have been emphasized almost exclusively while larger structural issues of object-oriented systems have been ignored. Some programmers have been led to believe that the principles of structured programming are obsolete and have been replaced by object-oriented programming. Today, more than

9350153

ever, the principles of structured programming hold firm. Object-oriented programming supplements these principles. The expert programmer will merely add the tools of object-oriented programming to the professional toolbox, and through experience and training know how to select the appropriate tool for the job at hand.

As much as possible the exercises in this book convey realistic and exciting object-oriented application programs that prepare you for problems currently being explored in industry and education.

Starting with the concepts of object-oriented programming the book shows complete working example programs that are made easier to build and understand precisely because they use object-oriented programming techniques. Seeing how the general concepts of encapsulation, inheritance, polymorphism, and message passing simplify program construction provides the motivation for study of language features. Not until the general concepts are thoroughly discussed in the context of working programs does the book go on to a detailed discussion of the manner in which various languages provide features to support the object-oriented style of programming.

In the second part of the book the implementation of object-oriented language features is compared and contrasted in C++, object-oriented Pascal, Actor, and Smalltalk. In addition to language features, the significant features of programming environments are discussed, including the presentation of powerful programming tools introduced by object-oriented programming environments like Smalltalk, which invented browsers, inspectors, and debuggers to provide development tools that were as sophisticated as the new concepts introduced by objects.

The third part of the book is devoted to a specific type of class library called an application framework. Application frameworks show the power that object-oriented structure can give to users by making code reuse possible.

Every effort has been made to make example programs as realistic as possible. For this reason the initial programs are larger than those you will typically see used to introduce object-oriented programming. The size and complexity of the initial programs is not accidental. Realistic programs of moderate size provide better illustrations of object-oriented programming principles.

93201239

PROGRAMMING LANGUAGES, ENVIRONMENTS, AND TOOLS

As mentioned, this book covers four popular object-oriented programming languages: C++, object-oriented Pascal, Actor, and Smalltalk. The main operating environments for both program development and program execution include MS-DOS and Windows. Programs developed for Smalltalk can run almost without modification on the Macintosh and under UNIX X-Windows, as well as under MS-DOS in graphics mode and under Windows and the OS/2 Presentation Manager.

Much of the code presented for C++ will run under any operating environment that supports the standard iostream library. However some of the lower-level graphics output requires the support of graphics primitives to draw lines and ellipses. All modern graphics environments support such primitives. To make the programs as portable as possible, an abstract screen class is developed and used for graphics output. All you need to do to get the programs to run in your specific environment is define the line and circle drawing methods inside the abstract screen class.

As a specific example, complete implementations of the abstract graphics screen class are provided for the Zortech C++ Flash Graphics library, for the Borland BGI, and for Microsoft Windows. Adapting this class to run under X-Windows should be a project that will take you from an hour to a half-day, depending on your level of familiarity with X-Windows.

The specific language products used for development include:

- Borland C++ and Turbo C++ (all versions)
- Zortech C++
- Digitalk Smalltalk/V 286
- Digitalk Smalltalk/V Windows
- Turbo Pascal (5.5 and later)
- Turbo Pascal for Windows
- Microsoft QuickPascal
- Actor from the Whitewater Group

Other language products were also used to develop example programs. These include standard tools for Microsoft Windows, such as the Software Development Toolkit (SDK) and add-on class libraries designed to enhance the basic products just listed. These include

- C++ Views
- Turbo Vision
- ObjectWindows
- Whitewater Resource Toolkit
- Microsoft Windows 3.0
- Microsoft Windows SDK

C++ Views is an application framework and class library by CNS, Inc. for developing Microsoft Windows applications. C++ Views works with both Zortech C++ and Borland C++. Turbo Vision is an application framework and class library developed by Borland that works with Turbo C++, Borland C++, and Turbo Pascal. Turbo Vision provides support for character-oriented window systems (COWS) running under MS-DOS.

ObjectWindows is an application framework and class library developed by the Whitewater Group in cooperation with Borland International. Object-
Windows runs under Microsoft Windows and works with Borland C++, Turbo Pascal for Windows, and Actor. ObjectWindows uses a class hierarchy and class naming conventions that are nearly identical to Turbo Vision. The Microsoft Windows SDK is required only if you are using Zortech C++. Borland C++ provides its own version of the Microsoft Windows standard library, Windows-compatible linker, header file, and resource compiler. Actor and Smalltalk/V Windows do not require the SDK. The Whitewater Resource Toolkit is handy for developing resources such as menus, dialog boxes, and bitmapped images such as icons and cursors. The Whitewater Resource Toolkit comes with Actor, Borland C++, and Turbo Pascal for Windows.

You do not need to have all these products to follow this book. In fact you do not need to have any software if you are simply trying to learn the basic concepts of object-oriented programming. If you are shopping for languages or tools for an object-oriented programming project, this is a good place to start. You'll get a solid feeling for what it is like to use all of

the products just mentioned. By the time you are through with the book you should be able to adapt any program you find to the desired target language by seeing the different approaches each of these products takes to solving a similar problem.

All program listings have been submitted to the appropriate compiler or interpreter to verify their correctness. However, you may be using a different implementation of one of these languages than was used for the development of this book. This is only likely to be a problem with Smalltalk other than Digitalk (that is, Parc Place System's Smalltalk) and with very special-case issues for object-oriented Pascal.

Note that the term used in this book is *object-oriented Pascal*, not Object Pascal. Object Pascal is a specific product of Apple Computer, originally called *Clascal*. Both Borland and Microsoft followed the basic implementation of Apple's Object Pascal when they extended standard Pascal to support object-oriented programming language features. The term *object-oriented Pascal* is used in the broadest sense to include Turbo Pascal 5.5 (and later), Microsoft's QuickPascal, and Apple's Object Pascal.

WHAT YOU SHOULD KNOW

You should have a basic familiarity with at least one procedural language such as C or Pascal. If you have worked with the newer varieties of BASIC, which support the definition of data structures and named subroutines, you will do fine. You should be familiar with basic program structure such as conditional statements, loops, and callable subroutines, as well as basic input and output mechanisms for printing strings and reading input from the keyboard. A good understanding of structural programming would be helpful, but is not essential.

APPLICATION FRAMEWORKS

Application frameworks are a special type of class library used in the quick construction of application programs. Application frameworks provide

此为试读,需要完整PDF请访问: www.ertongbook.com

menus and dialogs, and use built-in event-driven input to simplify the construction of window-oriented programs. Application frameworks are presented that use character-oriented windows in MS-DOS as well as the Microsoft Windows graphical user interface. The third part of this book should be approached as an example of how object-oriented programming can extend power and flexibility to programmers through class libraries. Class libraries for standard C, Pascal, and FORTRAN can extend power, but seldom flexibility. Because standard libraries do not support inheritance and polymorphism, they cannot be extended and specialized by the programmer. Application frameworks are more than an example of the power of class libraries. Application frameworks will change forever the way you program systems that require sophisticated user interfaces. You should read this section as an example of things to come and as proof that continuing to design your own user interface code is an obsolete practice.

OBJECT-ORIENTED PROGRAMMING AND WINDOWS

Windows and objects are natural companions. This is true whether you are talking about a character-oriented window system for MS-DOS or a graphical user interface such as Windows. Windows are by definition complex data structures manipulated through a variety of standard operations. In other words, windows are objects.

The windows theme recurs frequently throughout this book—it is a good approach for pulling together a diverse set of languages and topics under one umbrella. The windows theme also guides and unifies the example programs to demonstrate a natural progression of basic concepts. By the time you have worked through the examples you should find it easier to approach any window system or application framework, even if the window system is not explicitly object-oriented. It is surprising how much this approach can prepare you to understand the inner workings of Microsoft Windows and the SDK, even if you are programming in C.

BIBLIOGRAPHY

Booch, Grady. *Object-Oriented Design with Applications.* Redwood City, CA: Benjamin/Cummings, 1991.

Cox, Brad J. *Object-Oriented Programming: An Evolutionary Approach.* Reading, MA: Addison-Wesley, 1986.

Ellis, Margaret A., and Stroustrup, Bjarne. *The Annotated C++ Reference Manual.* Reading, MA: Addison-Wesley, 1990.

Franz, Marty. *Object-Oriented Programming Featuring Actor.* Glenview, IL: Scott, Foresman, 1990.

Fuller, R. Buckminster. *Synergetics: Explorations in the Geometry of Thinking.* New York: Macmillan, 1975.

Goldberg, Adele, and Robson, David. *Smalltalk-80: The Language and Its Implementation.* Reading, MA: Addison-Wesley, 1983.

Hansen, Tony L. *The C++ Answer Book.* Reading, MA: Addison-Wesley, 1990.

Meyer, Bertrand. *Object-Oriented Software Construction.* New York: Prentice Hall, 1988.

Peterson, Gerald E., ed. *Object-Oriented Computing.* Vols. 1 and 2. Washington, DC: IEEE Computer Society Press, 1987.

Petzold, Charles. *Programming Windows.* Redmond, WA: Microsoft Press, 1990.

Schmucker, Kurt J. *Object-Oriented Programming for the Macintosh.* Hasbrouk Heights, NJ: Hayden, 1986.

Stroustrup, Bjarne. *The C++ Programming Language.* Reading, MA: Addison-Wesley, 1986.

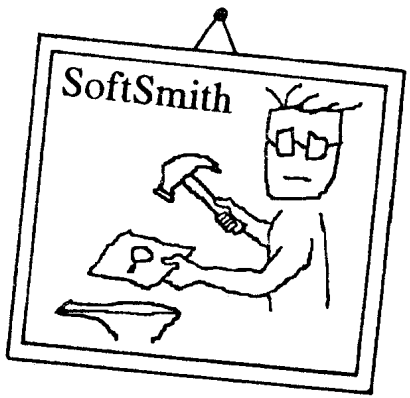
WindowFrame

by Greg Voss and Bruce Clift

Build your own MS-Windows application framework with this booklet/software package.

Regular price: $119.95
Special offer: $ 49.95*

(*Limited to owners of *OOP an Introduction*)



Cut through the MS-Windows learning curve. ***The WindowFrame Construction Toolkit*** leads you through development of a C++ application framework for Microsoft Windows. Organizes MS-Windows into easy-to-use, intuitive, classes. Works with any C++ compiler that generates Windows application programs, including Borland, Zortech, and TopSpeed C++.

100+ Page Construction Manual
See OOP techniques in action:

- Simplified event handling
- Automated window registration
- Callbacks to C or C++ functions
- Encapsulation of window features
- Customize through inheritance

C++ Source Code Disk
Application framework features:

- Complete source for framework
- Complete example programs
- Dialog boxes and menus
- Container class library
- Programmer's editor

Please send _____ copies of ***The WindowFrame Construction Toolkit.*** ($49.95 each, postpaid. Tax included. Foreign orders add $10 shipping/handling and use a check from a US bank or a credit card.)

Name

Address

City State Zip

IBM PC Diskette Size (check one): 5-1/4"_____ 3-1/2"_____.
Method of Payment: Check_____ Visa_____ MC_____.

Credit Card Number Expiration date

Signature Telephone

SoftSmith Publishing **P.O. Box 1124** **Soquel, Ca 95073**

Get the Disk!

Companion Disk for

Object-Oriented Programming: An Introduction

$12.95

Source code for example programs

Complete source code for all programs in the book. Includes makefiles and project file lists.

Additional ObjectWindows examples

Advanced examples of the Borland/ Whitewater application framework. Examples are in Actor, Pascal, and C++.

Smalltalk/V Windows

Now Smalltalk works with MS-Windows. As a prototyping tool Smalltalk is unsurpassed. See Smalltalk/V Windows examples in action.

Advanced graphics examples

Rotate three dimensional shapes. Source in C++, Pascal, Smalltalk, and Actor.

WindowFrame demo programs

WindowFrame is a separate product available from SoftSmith. This disk includes sample programs demonstrating the convenience and power of the *WindowFrame* application framework.

($12.95 each, postpaid. Tax included. Foreign orders add $10 shipping/ handling and use a check from a US bank or a credit card.)

Please send me _____ copies of the companion disk for *Object-Oriented Programming: An Introduction.*

Name _____

Address _____

City _____ State _____ Zip _____

IBM PC Diskette Size (check one): 5-1/4"_____ 3-1/2"_____

Method of Payment: Check_____ Visa_____ MC_____

Credit Card Number _____ Expiration date _____

Signature _____ Telephone _____

SoftSmith Publishing **P.O. Box 1124** **Soquel, Ca 95073**

CONTENTS

此为试读,需要完整PDF请访问: www.ertongbook.com