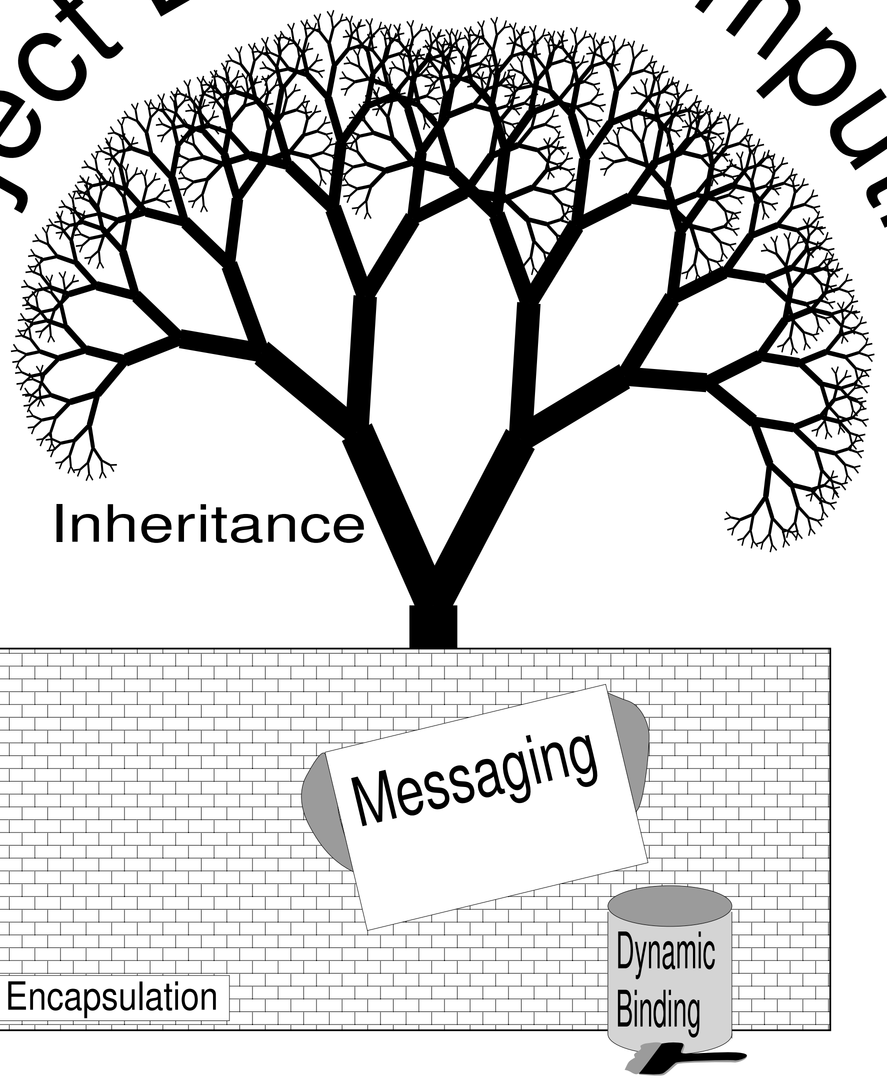
Object Based Computing



A NeXTstep Tutorial and Cookbook

Dan McCreary

Version 1.5

Object Based Computing

A NextStep Tutorial and Cookbook

Dan McCreary
Systems Engineer, NeXT Inc.
2626 E 82nd Ave. Suite 325
Minneapolis, MN 55425
Dan_McCreary@NeXT.com

Version 1.5

Please feel free to use any part of this early version (pre 1.0) of the textbook in any classes you are developing. I only ask that you give me feedback as to how useful it is and where it needs improvement before you tell others. The more people use it, the more feedback we get and the better the final product will be.

If we attain the blessings of the appropriate people within NeXT it is possible we will attempt to publish a later version of these materials through a publisher. Until that version is available users may make as many copies as you feel are necessary. After a 1.0 version is available the quality will be high enough that users will feel it is worth their while to go out to purchase a published copy.

All these materials are designed to be used with the NeXT Digital Librarian. This allows quick searching for key words. To use it just drag the folder containing these materials directly into the Digital Librarian window. See the on-line documentation for more information.

Please do not make changes to this documents without clearly stating in the page headers that changes have been made.

Please feel free to send me a printed copy with red ink, an on line copy with you comments in *16 point italic font* **or just fill out a reviewers form in the "DOC" sub-directory. I will attempt to produce a new version of this text every few months until the text is finished.**

ISBN 0-000-000000-0

ABCDEFGHIJ-DO-89

Preface

This book evolved from my work teaching classes in object based computing to faculty members who purchased NeXT systems around the midwest. I started with a half day seminar in which I tried to allow people to uncover the immense potential of the systems they had purchased. Many of them purchased NeXT systems because of its superior publishing tools such as FrameMaker or its advanced analytical tools such as Mathematica. Others found it to be one of the first strong multi-tasking environments which was also was easy to use. Unfortunately, these features overshadowed what I feel is the NeXT systems greatest strength: its ability to allow the user to quickly create and manipulate abstractions of the world around them. My task was to show people in a half day that the NeXT system also had the most revolutionary application development environment ever created.

In the summer of 1989, I heard a presentation by Dr. Ed Barboni of Allegheny College. He saw the NextStep environment had far reaching implications: not just for programmers but for anyone who needed to learn to manipulate information and create value from that information. We shared a vision that object based computing was not just a programming environment, it was was a core technology of the information age. Barboni's vision was that to be competitive, Allegheny College needed to teach the skills required to create and manipulate abstractions and that NextStep was the best environment to teach those skills. Many of these ideas have been greatly influenced by Shoshana Zuboff's text *"In The Age of a Smart Machine"*[1].

The more I read, the more I started to realize that I was on to something really big. But the ideas came from diverse sources and I was having a hard time putting all my thoughts into a clear presentations I could give to faculty at a small college. I was also being asked to attempt to explain these concepts to

[1]Shoshana Zuboff, *In the Age of the Smart Machine: The Future of Work and Power* (Basic Books Inc., 1988)

groups of MIS Managers at fortune 500 companies who couldn't spare more then an hour from their busy schedules. I reasoned that creating a book would help me clarify my thoughts and allow me to condense the essential information into a short presentation.

A second reason for creating a book was the realization that while there are only a few simple concepts needed to create NextStep applications, there was little material on object based computing targeted at non-technical people. There was a great deal of information for graduate students who could spend hours in a library doing literature searches, but each textbook or article had a large number of assumptions as to what background the reader had. Ideally, a book on object based computing should minimize these assumptions if it is to reach the widest audience.

A third reason was to give people the appropriate background to get greater value from advanced classes such as NeXT developers class. Many of these classes assume people have an understanding of object oriented programming and event based programming. This book attempts to introduce these concepts to people who have a minimum programming background.

One of the most exciting aspects of this book is the fact that NeXT users will actually have the ability to create and integrate powerful and useful applications with the material in this book. Most of the introductory programs are less then a dozen lines long, and each chapter introduces few new concepts at one time. The cookbook recipes have been chosen to build the readers vocabulary and give ideas of how their objects influence the way they create mental abstractions of the world around them.

Target Audience

The book has been designed to be used as a textbook for freshman college students and advanced high-school students. It assumes the reader has a programming background which includes a description of flow control (for,if-then-else, subroutine calls), as well as the basic data types used (integer, float, strings). Although exposure to advanced data structures (linked lists, trees etc) is helpful, it is not required.

I have been told the introductory chapters have been very useful for managers of MIS departments and college deans responsible for integrating technology into the curriculum. There are very few places in the first five chapters which require a programming background. Many of the recipes can be used without an understanding of the UNIX "shell".

Language Independence

The central concepts in this book are how to create objects, extend objects, integrate objects and communicate between objects using message statements. These concepts are language independent. Because every NeXT comes with an compiler and development environment which uses messages we use those tools in the examples programs. This language called Objective C. Objective C is a super-set of the C language which is the default language used on most workstations. It is exactly like C with the addition of the message statement. Previous knowledge of the C language is very helpful but not required for most of the programs in the first half of the text. The essential aspects of object based computing are not bound to one specific language, so techniques learned in this text can also be applied to many other programming systems that use messaging. The C compiler used on the NeXT is based on the GNU C compiler which has many ANSI extensions which give it superior type checking. This allows programmers to catch errors early in the development cycle. This GNU compiler and the NeXT extensions are available at no cost under the GNU license.

Structure

This book is designed as a cookbook. After a short introduction to the theoretical foundations it gives recipes for creating a few simple programs. And just like learning to cook in the kitchen, you can learn to create more complicated programs to match your tastes by simply combining the various components of the simple recipes. The way many people learn to program is not by reading textbooks on the theory of programming but by taking examples from books and from other people and extending them to solve the problems we are interested in. Each recipe will introduce a few new concepts. After the foundation chapters (1-5) some recipes are independent small programs, but many also build on previous recipes. We recommend reading chapters 1-5 approximately in the order they appear in the book. People who are already familiar with object based computing and NextStep may go directly to the later chapters. The overall dependency structure of the book is given in figure 1.

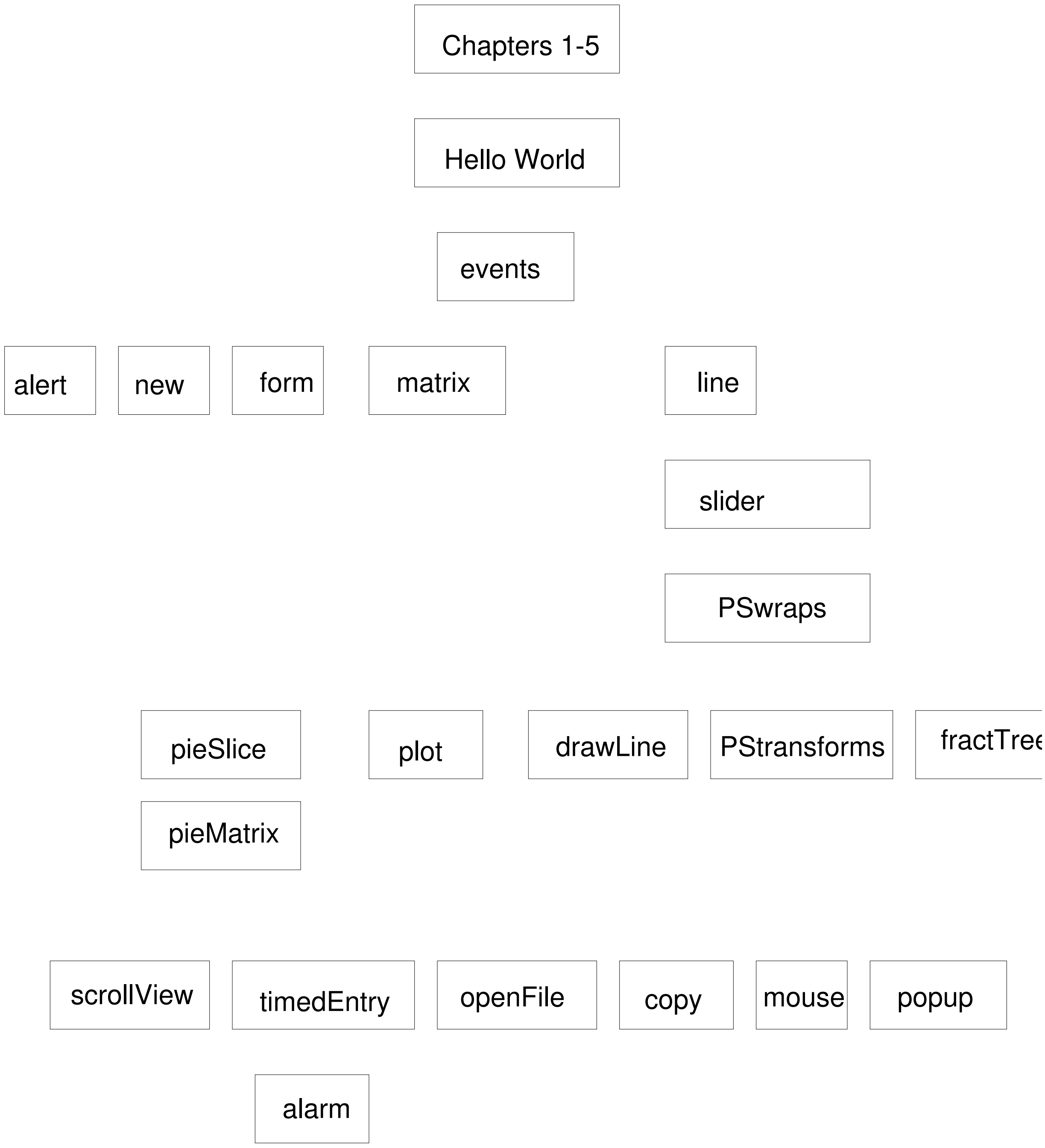


Figure 1: Dependency Of Chapters

The first half of the textbook (Part I) can be used by any person wanting a general overview of an advanced object oriented programming system. The second half of the book is most helpful for people who have access to a NeXT computer system.

Graphical Feedback

One of the goals of this text is to quickly get the reader to the stage where they can draw a line on the screen. Once they can do this and hook up controls which change the drawing they get immediate graphical feedback. This feedback gives immediate positive reinforcement as they reach each stage.

Exercises

At the end of each chapter is a set of exercises. They begin with some very simple exercises which can be used to illustrate the major points of each chapter. Exercises with an asterisk following them are questions designed to encourage the reader extrapolate the information given in each section and integrate it with other chapters. These exercises can also take a considerable amount of additional time.

Chapter by Chapter Summary

I have struggled with the order of materials for this book for many hours. The dilemma is "Should the tedious but essential theory part go first or will this scare off too many people?" In the end I have chosen to mix in bit of "doing but not fully understanding" as the second chapter before I trudge through the more theoretical chapters which follow.

1. Introduction

This chapter covers the "why" of object base programming. It give a historical perspective of our movement into the information age and how object based computing features provide benefits for information managers. This chapter is important because it provides a background for several questions which will be answered later in the text.

Part I: The Foundations of Object Oriented Programming

2. A Tour of the Interface Builder

Our goal is to create a simple program which has many of the objects used in creating user interfaces. This includes buttons, sliders, a box with forms, scrolling text, and sound. After we place these objects on the screen we will connect them together and modify their initial conditions. The first five chapters of this book will be to explain what we are really doing and how we will extend and integrate these objects.

3. Hello World: Creating a New Object

In this chapter we create a simple program which has one button. When pressed the button will print out the message: "Hello World" in a UNIX shell. This is often one of the most confusing sections to get through because it involves approximately twenty mouse events. The theory behind our actions this chapter is described in detail in the next chapter.

4. Encapsulation and Inheritance

In the last chapter we created a "black box" around an object and allow events such as a button being pressed to execute a section of program code within this object. This section discusses the reason for encapsulation and the advantages it creates for the developer and users of objects. We then discuss the relationships of object classes in an inheritance tree structure.

5. Messaging and Dynamic Binding

Once we have learned how to create objects we need a flexible way to allow objects to communicate. This chapter discusses the message statement and how dynamic binding allows us to direct messages in a flexible manner. It discusses the concepts behind connection based programming and how modern systems create relationships based on connections rather then location of objects.

6. Events

In this exercise we create two buttons which send messages to a object that displays a single integer. One button increments the integer and the other decrements the integer. This we will use to change the internal state of an object.

7. Initializing Objects

When we create new objects we have control over their initial states by controlling the factory that produces the objects. This example uses the previous example but adds a section which allows us to initialize the state of the object.

8. Simple Drawing

This chapter allows us to integrate inheritance and drawing into a simple program which draws a line on the screen. We then show how we can add a slider to the previous program and have the slider update a instance variable which in turn will change the way we draw a line to the screen.

9. More Drawing

This chapter demonstrates how we initialize data within a view as well as show how we can create "wraps" for files with PostScript commands. Examples are presented for creating pie charts as fractle trees.

10. Overview of the Application Kit

This chapter gives a broad overview of the Application Kit supplied as part of NeXTstep. Each of the general classes is briefly described and some samples cases of where the classes are used is discussed.

Part II: Building to your NextStep Vocabulary

Part II expands on the introductory concepts learned earlier by giving the reader a tour of additional Application Kit objects.

11. Using the Pasteboard

A sample program which allows the user to copy graphics into the pasteboard. They can then be pasted directly into the WriteNow word processor or used in other programs that can include encapsulated PostScript.

12. Speaker and Listener: Communicating With Other Applications

This chapter shows how to send and receive messages to other standard application kit objects. Examples for adding services.

13. Scrolling Views

This examples shows how we can easily add scrolling knobs to a view. We can then manipulate views which are larger then the the screen area they are displayed in.

14. Text Objects

The Text object offers a rich varity of ways to manipulate text in your application. This chapter covers basic functionality of the Text object in a simple Text editor as well as some of the details needed to manage text files.

15. Using Sound

The SoundEditor program is used as an example of integrating sound into an application.

16. Timed Entries

A simple stopwatch is used to demonstrate that not all events must come directly from the mouse or keyboard. Timed entries cause events to occur at regular intervals. Using timed entries is important for any objects that include animation. The stopwatch has just two buttons (start and stop) and displays the number of seconds that have elapsed in 1/10 second intervals.

Exercise: A Chronograph

17. Monitoring Mouse Motion

This chapter demonstrates how to monitor mouse events over a view. The program we build will display the x and y location of the mouse when we press the mouse button inside a view object.

18. Animation

This chapter will demonstrate programs that draw with motion.

Part II: Extending Accessibility: Building Custom Palettes

If you build a really fantastic object, you can share the object with other programmers using the information presented up to this point. But for every programmer out there there could be 100 non-programmers that might be able to use you object if it could be accesed from a "point-and-shoot" environment. This chapter covers the background you need to make your objects accessible through Interface Builder.

19. Working with Streams

An example of how to use the Stream library to read and write to files.

19. Archiving Objects: Typed Streams

After Streams are mastered we will need to used typed streams to read and write data to memory.

19. Building your own Palettes

A step by step description of the process of making objects accessible through the palette interface. The gauges and PieChart examples are used.

20. Building a Complete Program

This chapter covers some of the details necessary for complete program including assigning an icon to the application, assigning an icon for data files, creating on-line help and using multiple nib files. A Disk Usage analizer which creates PieCharts program is used as the example program.

Acknowledgements

First and foremost I must thank my wife Nancy, for her support, encouragement, and detailed editing. It was usually our time together that suffered because of because of this project.

I am deeply indebted to my colleges within NeXT that made this book possible. First, I would like to thank Greg Smedsrud who made me realize that a good salesman is a good teacher and a good teacher is a good salesman. A great deal of thanks goes to Dave Stutz who's overheads expanded my realm of thought, whose initial "toy" examples gave me enough confidence to keep trying, and who allowed me to use his phrase "Object Based Computing" as the title for this book. Berry Silverman also helped with early comments on typography. I would like to thank Bob Longo who taught me that balance is what gets us through in the long run. And finally to Steve Jobs, whose wisdom made computers of the 1990s available to the masses in 1988.

I also want to thank the faculty, staff and students at the institutions around the country who have worked with me in the creation and refinement of this book. Faculty at Gustavus Adolphus College, Allegany College and Rose Hullman deserve special mention. Without their feedback the final product would still show many of the early the rough edges.