Introduction Chapter 1

1 Chapter 1 Introduction

Sections

When you begin any enterprise, you must find a starting point. You set out from that starting point and acquire a basic vocabulary, a notion of boundaries and techniques, a sense of how things fit together and what is possible. For those who want to learn how to create OPENSTEP applications, this book provides a starting point.

With this book you become familiar with OPENSTEP application development not merely by reading but by *doing*. The book guides you through the creation of three applications of increasing complexity. Along the way it explains related concepts and issues. The techniques and concepts you learn in one tutorial lay the foundation for the more advanced techniques and concepts in the next tutorial.

The final chapter of the book tells you where to go for further information and where and how to find things, such as tools and documentation. It also tells you how to get NeXT products and services.

This book covers a lot of ground, although sometimes at only a summary level. Finishing this book makes you much better prepared to take on serious application development with OPENSTEP in general and the Enterprise Object Framework in particular.

Although the aim is primarily to educate, this book is also intended—for those interested in programming—to be fun.

. . . . . .

Some of you might be new to OPENSTEP. To learn more about OPENSTEP, the standard on which it's based, and OPENSTEP Developer, turn the page.

What is OPENSTEP?

OPENSTEP is NeXT Software's graphical, object-oriented user and development environment. It is based on the OpenStep standard and available on a variety of platforms. OPENSTEP is earning a growing reputation in the corporate world as the premier environment for developing and deploying mission-critical custom applications.

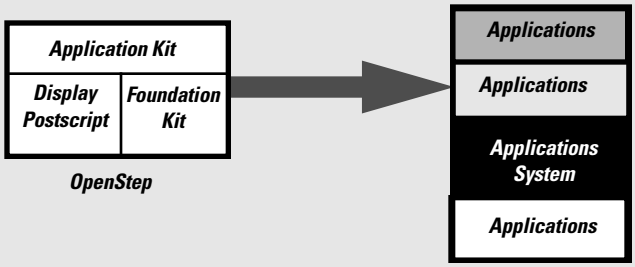
The two core components of the product are OPENSTEP User and OPENSTEP Developer.

OPENSTEP User is a user environment acclaimed for its intuitively navigable desktop and file manager. On it you can easily deploy your own OPENSTEP applications as well as those supplied by NeXT and third-party vendors. Intelligent networking, particularly NetInfo, makes it possible to install and upgrade OPENSTEP in a fraction of the time it takes other systems.

OPENSTEP Developer, NeXT's software-development environment, provides seamlessly integrated set of tools for building complex applications that can be deployed on heterogeneous client/server networks running not only OPENSTEP, but Portable Distributed Objects, Enterprise Objects Framework, and OpenStep-based software developed by other vendors.

OpenStep

OpenStep is the software industry's first open standard for object-oriented software development. It is an application programming interface (API) based on the fundamental NEXTSTEP object layer: the Application Kit, the Foundation Kit, and Display PostScript.



The OpenStep object layer allows corporate customers to create, evolve and deploy multi-tier, client/server business applications in a fraction of time it takes other methods.

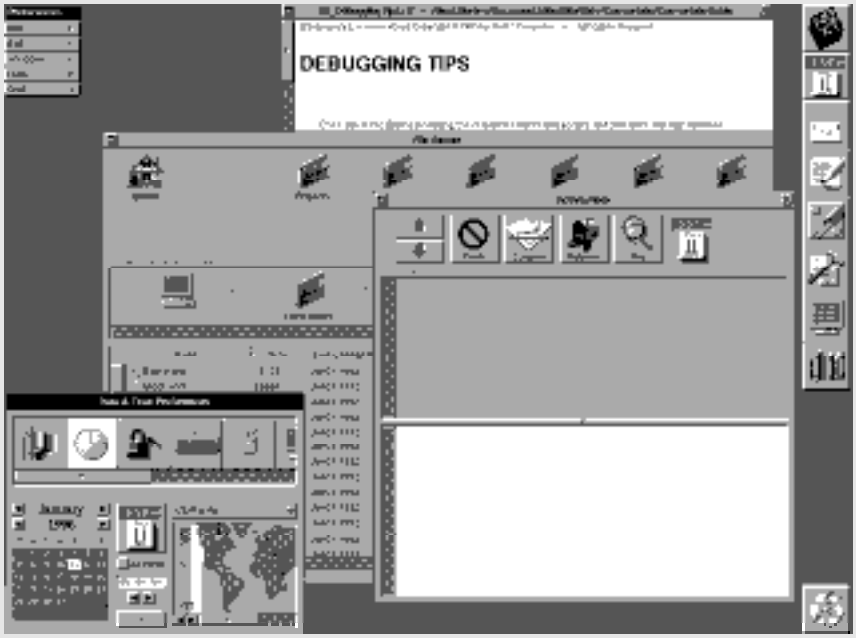
- Application Kit: APIs for user-interface objects and for essential application behavior, such as event handling
- Foundation Kit: APIs that define basic object behavior, that support object persistence and distribution, and that "objectify" collections, Unicode strings, and many other programmatic entities
- Display PostScript: APIs for PostScript drawing

What's in a Name?

"OPENSTEP" refers to the software product. "OpenStep" refers to the standard or specification on which the product is based, and by extension to the concepts expressed by the specification.

The OpenStep specification is available via anonymous ftp at ***ftp. next. com****.*

*The OPENSTEP user environment includes File Viewer (a file-system browser), Mail, Preferences, Edit, and other applications in **/NextApps**.*



Other Products from NeXT

- With **WebObjects** you can easily create dynamic websites. WebObjects applications provide a smart, interactive connection between corporate databases and customers on "The Web."
- **Enterprise Objects Framework** enables you to construct applications that use (and reuse) enterprise business objects, storing them in relational databases such as those from Oracle and Sybase.
- **Portable Distributed Objects (PDO)** allow objects in a single application to be distributed among a heterogeneous network of OpenStep clients and a broad range of servers.
- **D'OLE** brings the PDO object model to the Windows platform, giving Windows application developers the ability to make use of distributed-object technology.

In addition to software products, NeXT provides developer support and professional services, in particular the Object Experts program, an innovative on-site support and training program.

OPENSTEP— Platforms and Interoperability

Supported Platforms	Intel-based PCs, Sun SPARC workstations, NeXT's Motorola 68040-based computers.
CORBA	NeXT has licensed SunSoft's implementation of the CORBA standard (from the Object Management Group) and is committed to CORBA interoperability.

Power Programming With OPENSTEP Developer

OPENSTEP Developer 4.0 is a programming environment ideally suited for the rapid development of custom object-oriented applications deployable on a variety of computer architectures. It comprises an integrated set of development software, libraries, header files, tools, documentation, and other resources.

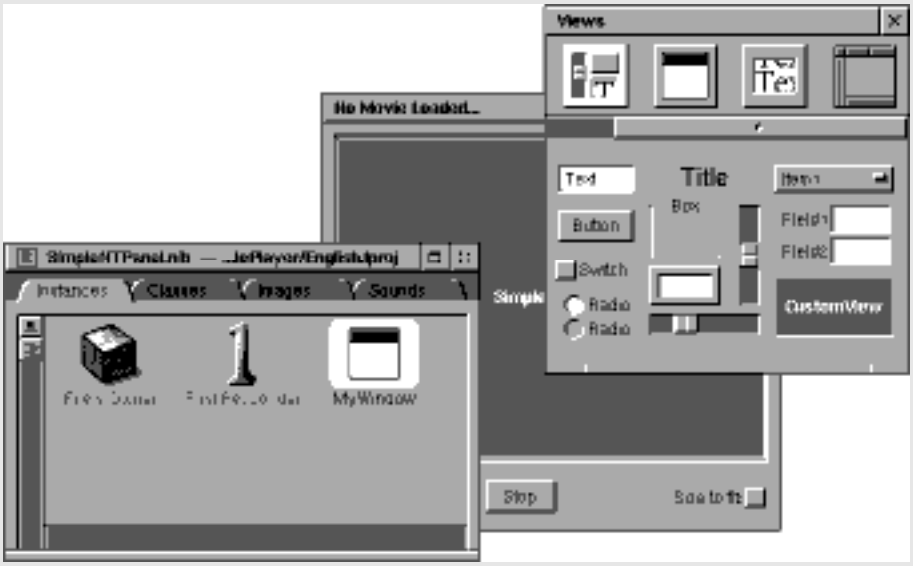
Project Builder is an application that manages software-development projects, and that orchestrates and streamlines the development process. It integrates a project browser, a full-featured code editor, language-savvy symbol recognition, sophisticated project search capabilities, header file and documentation access, build and debugging support, and a host of other features.

Interface Builder makes it easy to create application interfaces by dragging objects from palettes. Standard palettes hold an assortment of Application Kit objects. Custom palettes can include third-party objects as well as the developer's own objects. Interface Builder archives and restores elements of a user interface as objects—it doesn't "hardwire" them into the interface. Interface Builder helps to connect objects for messaging, and it assists in the definition of custom classes.

The main window of Project Builder combines a project browser with a code editor.

Iconic buttons above the browser let you access the application's Build, Project Find, and Loaded Files panels, as well as the project inspector and the graphical debugger.

When projects are indexed, Project Builder caches all symbols in memory and makes them instantly available upon request.



Interface Builder lets you craft user interfaces from palettes of ready-made objects, then store the interface in a file.

A palettes window contains assortments of standard Application Kit and (if installed) Enterprise Objects Framework objects as well as custom objects that you or third-party developers create.

A nib file window enables the initial definition of custom classes and facilitates the connection of these classes to objects on the interface. It also catalogs the image and sound resources used in the interface.

- **OpenStep Class Libraries.** Includes NeXT's implementation of the Application Kit, the Foundation Kit, and Display PostScript, with some extensions.
- **Objective-C.** An object-oriented programming language that is an extension of standard ANSI C, Objective-C is a simple and powerful language. It is easy to learn, yet elegant in its application to the problem domain. OPENSTEP projects can include Objective-C, C, and C++ source code.

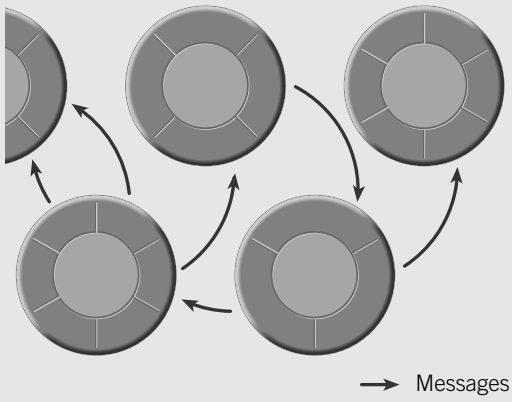
In addition, OPENSTEP Developer 4.0 offers a new version of the GNU C compiler, enhancements to C++ compilation, and GNU make technology.

- **Compiler and Library Technology.** Two important features of OPENSTEP Developer 4.0 for Mach are dynamic shared libraries and frameworks. Programs linked with a dynamic shared library share one copy of that library's routines, and are linked with only those modules they currently need. Frameworks assemble all library components in one place: executable code, header files, resources, and documentation. The executable code is in the form of a dynamic shared library. The Application Kit, Foundation, and Display PostScript are installed as frameworks.

The Advantage of Objects

Objects are the software equivalent of the Industrial Revolution. In the same way that modern factories assemble products out of prefabricated components rather than manufacture every product from scratch, object-orientation allows programmers to build complex software by reusing software components called objects. Specifically, objects lead to several measurable advantages:

Greater reliability. By breaking complex software projects into small, self-contained, and modular objects, object-orientation ensures that changes to one part of a software project will not adversely affect other portions of the software. Being small, each of these objects is a well-tested module of code, and so the overall reliability of the software increases.

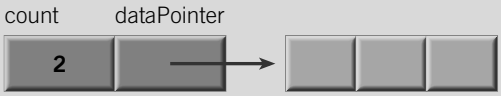


Greater maintainability. Since objects are modular and usually small (in terms of the overall code size of a project), bugs in code are easier to locate. Developers can also change the implementation of an object without causing havoc to other parts of an application.

Greater productivity through reuse. One of the principal benefits of object-orientation is reuse. Objects can be integrated into many applications. And through subclassing you can create specialized objects merely by adding the code unique to the new object. Objects of the new subclass inherit functionality from the superclass, reducing coding and promoting greater reliability.

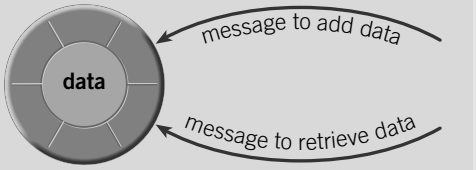
An Example

Object-oriented programming delivers its greatest benefits to large and complex programs. But its advantages can also be demonstrated with a simple data structure such as might be used in any application.



With procedural programming techniques, the application is directly responsible for data manipulation. One problem with this is illustrated in the picture above: It shows a data structure consisting of a **count** variable and a data pointer. Since the application directly manipulates the data, it has the opportunity to introduce inconsistencies. Here, it has added an item to the data, but has forgotten to increment the count; the **count** variable says there are still only two data elements when in fact there are three. The structure has become inconsistent and unreliable.

Another problem is that all parts of the application must have intimate knowledge about the structure of the data. If the allocation of data elements were changed from a statically allocated array to a dynamically allocated linked list, it would affect every part of the application that accesses, adds, or deletes elements from the list.



With an object-oriented programming paradigm, the application as a whole wouldn't directly manipulate the data structure; rather, that task is entrusted to a particular object. Since the application doesn't directly access the data, it can't introduce inconsistencies. Note also that it's possible to change the implementation of the object without breaking other parts of the application. For example, the data storage method could be changed to optimize performance. So long as the object responds to the same messages, other parts of the application are unaffected by internal implementation details.

The Advantage of OPENSTEP

Proven Technology. NeXT Software's technology has been evolving through 10 years and four major releases. During that time, it has been rigorously tested and iteratively refined. NeXT has an established track record in object technology, while it will be years before its major competitors can offer comparable technology of comparable maturity.

True Objects. OPENSTEP objects are truly objects—modular, autonomous, persistent, and distributable. They are not static entities, but can be bound dynamically at run time. When you drag an object from an Interface Builder palette, you're getting a real object and not an area painted on the screen with some code attached.

Portability. OPENSTEP is designed to foster both hardware portability and operating-system portability.

Simplified Client/Server Development. OPENSTEP Developer's integrated tool set simplifies the complex process of building distributed client/server applications.

Substantial Business Benefits. OPENSTEP's object-orientation helps managers to accelerate the introduction of new products and services that depend on new software. With OPENSTEP programmers can modify software quickly and assuredly to take advantage of evolving business opportunities. Through reusable object libraries, systems integrators can quickly customize a generic product to produce an individualized software solution for each client.

Don't Take Our Word For It

Here are a few comments on OPENSTEP and its predecessor, NEXTSTEP:

- Booz Allen & Hamilton's study of OPENSTEP development suggests that experienced developers could increase their productivity five to ten times.
- "Information is our business. That's why OPENSTEP succeeds here. The most important product we have is the quality of the service we provide: our timeliness, the effectiveness of our analysis and planning."
  Director, Software Engineering
  Fannie Mae
- "We would never have be able to do what we did on time or on budget if we had chosen any other solution but NeXT."
  Manager, IS Branch Automation
  Chrysler Financial Corporation
- "The great thing about object-oriented programming is that the longer you're at the game, the more benefits you derive. You can reuse objects you've created or add to objects to make them more robust. And NEXTSTEP is the best integrated computer platform on the market."
  MIS Manager,
  UBS Securities

In three years, Nicholas-Applegate Capital Management reengineered its business systems, enabling the company to manage its business growth from $4 billion to $14 billion in assets. Using object technology from NeXT, Nicholas-Applegate was able to develop an investment and trading environment that was flexible and able to expand as the company grew.