Introduction

NEXTSTEP™ Development Tools and Techniques describes the essential tools for developing a NEXTSTEP applicationÐthese tools include the Project Builder, Interface Builder™, Terminal, and Edit applications, miscellaneous developer applications, and the GNU C compiler, preprocessor, and debugger. The manual is part of a collection of manuals called the *NEXTSTEP Developer's Library*.

This manual assumes you're familiar with the standard NEXTSTEP user interface. Experience using a variety of NEXTSTEP applications would also be helpful. Some topics that are discussed here aren't covered in detail; instead, you're referred to a generally available book on the subject, or to an on-line source of the information.

A version of this manual is stored on-line in the NeXT™Digital Library (which is described in the *User's Guide*). The Digital Library also contains release notes, which provide last-minute information about the latest release of the software.

How This Manual is Organized

The first 14 chapters of this manual concentrate on the tools used in building a NEXTSTEP application. The last four chapters contain step-by-step instructions for creating several simple applications, thereby providing a hands-on overview of the application development process.

- Chapter 1, ªPutting Together a NeXT Application,º provides an overview of the tools and techniques that you'll use to assemble a working application. The tools introduced in this chapter are discussed in greater detail in other chapters of this manual.
- Chapter 2, ªThe Project Builder Application,º describes the central control point for application development in NEXTSTEP. Project Builder helps you with each stage of application development, from inception to installation.
- Chapter 3, ªThe Interface Builder Application,º describes the tool that lets you assemble your application's user interface (and other parts) from predefined building blocks, and lets you create new building blocks of your own design.
- Chapter 4, ªThe Edit Application,º describes the NEXTSTEP text editor you'll be using to edit and debug your application's source files.
- Chapter 5, ªThe Terminal Application,º describes the application you'll use to interact with a UNIX®shell from the NEXTSTEP workspace.
- Chapter 6, ªThe Icon Builder Application,º describes a simple graphic editor for creating and editing application icons.
- Chapter 7, ªThe DBModeler Application,º describes an application for building data models based on the structure of an existing relational database. The resulting models can be used in Interface Builder to construct applications that access the data in the database.
- Chapter 8, ªThe MallocDebug Application,º describes an application for measuring the dynamic memory usage of the applications you develop.

- Chapter 9, ªThe Process Monitor Application,º describes an application that lets you examine running processes, and pause or kill any of the processes and applications running on your computer.
- Chapter 10, ªThe PostScript®Previewers Yap and pft,º describes two tools: an application for developers who want to write and test PostScript code, and a shell-based interface to the PostScript Window Server.
- Chapter 11, ªThe GNU C Compiler,º describes GNU CC, the ANSI-standard C compiler used on NeXT computers. The chapter also describes how to compile a C program using the GNU compiler.
- Chapter 12, ªThe GNU C Preprocessor,º describes the macro preprocessor that's used to transform your C program or application before actual compilation. The chapter provides information about standard and precompiled header files, macros, and conditionals. It also lists the options that can be used with the **cpp** (C preprocessor) command.
- Chapter 13, ªThe GNU Source-Level Debugger,º describes GDB, the primary tool you'll use to debug the applications that you develop.
- Chapter 14, ªMach Object Files,º describes the format of Mach object (also known as Mach-O) files, which NeXT computers use instead of the UNIX 4.3BSD **a.out** format.
- Chapter 15, ªBuilding a Simple Application,º provides a tutorial introduction to the process of application development in NEXTSTEP. It gives you an introduction to Project Builder and Interface Builder, while showing you some of the basic features of the Application Kit.
- Chapter 16, ªBuilding a One-Button Calculator,º continues the tutorial introduction to the major development tools in NEXTSTEP and gives you further insight into object-oriented programming with the Application Kit™.
- Chapter 17, ªBuilding a Text Editor Using Multiple Nib Files,º shows how Interface Builder and the Application Kit are used to tackle more advanced issues of object-oriented application design.
- Chapter 18, ªBuilding a Custom Palette,º the final tutorial in the series, shows you how Interface Builder itself can be modified to include the objects and tools you find most useful.

Conventions

Syntax Notation

Where this manual shows the syntax of a function, command, or other programming element, the use of bold, italic, square brackets, and ellipsis has special significance, as described here.

Bold denotes words or characters that are to be taken literally (typed as they appear). *Italic* denotes words that represent something else or can be varied. For example, the syntax

print *expression*

means that you follow the word **print** with an expression.

Square brackets [ ] mean that the enclosed syntax is optional, except when they're bold **[ ]**, in which case they're to be taken literally. The exceptions are few and will be clear from the context. For example,

pointer [*filename*]

means that you type a pointer with or without a file name after it, but

[*receiver message***]**

means that you specify a receiver and a message enclosed in square brackets.

Ellipsis (...) indicates that the previous syntax element may be repeated. For example:

Syntax	**Allows**
pointer ...	One or more pointers
pointer [, *pointer*] ...	One or more pointers separated by commas
pointer [*filename* ...]	A pointer optionally followed by one or more file names
pointer [, *filename*] ...	A pointer optionally followed by a comma and one or more file names separated by commas

Special Characters

In general, notation like

Alternate-x

represents the character you get when you hold down the Alternate key while typing x. Because the modifier keys Alternate, Command, and Control interpret the case of letters differently, their notation is somewhat different:

Notation	**Meaning**
Alternate-x	Hold down Alternate while typing lowercase x.
Alternate-X	Hold down Alternate while typing uppercase X (with either Shift or Alpha Lock).
Alternate-Shift-x	Same as Alternate-X.
Command-d	Hold down Command while typing lowercase d; if Alpha Lock is on, pressing the D key will still produce lowercase d when Command is held down.
Command-Shift-D	Hold down Command and Shift while pressing the D key. Alpha Lock won't work for producing uppercase D in this case.
Control-X	Hold down Control while pressing the X key, with or without Shift or Alpha Lock (case doesn't matter with Control).

Notes and Warnings

Note: Paragraphs like this contain incidental information that may be of interest to curious readers but can safely be skipped.

Warning: Paragraphs like this are extremely important to read.