CTOS open

Programming Practices and Standards

Application Design

CTOS/Open
Programming Practices and Standards

Application Design

PRENTICE HALL
Englewood Cliffs, New Jersey 07632

Authors: *Alan Coleman* and *Margaret Morris*
Cover design: *April Bishop*
Page design: *Milena Martin-Arana*
Editorial/production supervision: *Mary Rottino*
Manufacturing buyers: *Kelly Behr and Susan Brunke*
Acquisitions editor: *Karen Gettman*

Published by Prentice-Hall, Inc.
A Division of Simon & Schuster
Englewood Cliffs, New Jersey 07632

The publisher offers discounts on this book when ordered in bulk quantities. For more information, write:

Special Sales/College Marketing
Prentice-Hall, Inc.
College Technical and Reference Division
Prentice Hall
Englewood Cliffs, New Jersey 07632

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN 0-13-194382-0

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

This manual describes recommended techniques and practices for writing portable applications that follow the CTOS/Open standard for the CTOS® operating system. This manual is a companion volume to the *CTOS/Open Application Programming Interface Specification*.

The CTOS/Open standard is a specification which is implemented by several CTOS-based operating systems. The standard is not itself an operating system. When this manual talks about CTOS/Open, it means the specification. When it talks about CTOS, or about CTOS-based operating systems, it means those operating systems that conform to the CTOS/Open standard.

The CTOS/Open standard is a common subset of features. The operating systems that comply with it generally have additional features beyond the ones included in the CTOS/Open standard.

Purpose of This Manual

This manual describes practices and standards programmers should follow to ensure that their applications are portable across the various implementations of the CTOS/Open standard. Following the recommendations in the standard can greatly simplify porting an application from one CTOS-based operating system to another.

This manual can also serve as a "programmer's primer" for those new to CTOS/Open. It provides many examples, showing how to write commonly-used operations. This manual does not provide an in-depth conceptual explanation of CTOS implementation internals. For more comprehensive conceptual information, see the manuals for the CTOS-based operating system you use.

Audience

This manual is intended for programmers who want to write portable applications for CTOS-based operating systems. Programmers who are experienced with these operating systems, and programmers new to them, should both find the recommendations and examples in this book useful.

Related Documentation

The following documents may be useful to programmers who read this book.

CTOS/Open Books

- *CTOS/Open Application Programming Interface Specification*

  This document describes the procedural interface definitions for each of the procedures supported by the CTOS/Open Standard. It defines a common set of system interfaces that is consistent across all CTOS/Open-compliant versions of CTOS. The specification is intended for use by software developers and Independent Software Vendors (ISVs) who want to write applications that conform to the CTOS/Open Standard and are therefore portable across vendor platforms. It assumes that the reader has experience writing applications under CTOS or under another operating system.

- *CTOS/Open Application Programming Interface Specification: Computer Graphics Interface (CGI)*

  This document describes each CGI operation and explains how to write your own CGI programs.

- *CTOS/Open Application Programming Interface Specification: Graphical User Interface (GUI)*

  This document introduces an eXtensible Virtual Toolkit (XVT) and describes its relationship to other windowing interfaces. It also reviews the current XVT operation set, which includes functions,

macros, constants, and types. This specification is a virtual API and is intended for programmers who want to write applications that run in several different window environments and work across all CTOS-based platforms.

- *CTOS/Open Application Programming Interface Specification: Networking Services*

  This document describes the standard Link Layer interface used under the CTOS operating system. It defines the common requests used by all Link Layers and the requests used by the Data Link Manager. This specification also provides the standard format for reporting Link Layer status, and documents Link Layer error messages/codes, event codes, and commands. The final draft of this manual will include information on the Transport Layer.

- *CTOS/Open Application Programming Interface Specification: Printing Services (GPS)*

  This document describes how to write applications that use the Generic Print System (GPS) or the Generic Print Access Method (GPAM).

- *CTOS/Open Programming Practices and Standards: User Interface Design*

  This document provides guidelines for designing graphical user interfaces for CTOS-based applications. It contains information about the user interface standard call Common User Access (CUA), which has become part of the public domain. This manual is a working draft.

- *Exploring CTOS*

  This book gives a conceptual overview of the CTOS architecture. Additionally, it explains the advantages of developing applications on a distributed, message-based operating system.

Intel Manuals

- *iAPX 286 Programmer's Reference Manual*

  This book describes the features and instruction set of the Intel 80286 microprocessor.

- *80386 Programmer's Reference Manual*

  This book describes the features and instruction set of the Intel 80386 microprocessor.

How to Use This Manual

This manual is organized by function. Individual chapters are devoted to each of the most commonly-used functional areas of CTOS. To use this manual, examine the chapter or chapters that apply to your project, and make use of the examples and recommendations you find there. To look up specific topics, use the index at the back of the manual.

The CTOS Naming Convention

The examples in this book follow a specific naming convention, which is designed to promote the readability of source code. The underlying principal of the naming convention is this: use explanatory prefixes and suffixes on all variables and procedure names. Elementary programming practice dictates that the names themselves should also be explanatory. These conventions are particularly important when programming in a language like C, which tends toward cryptic syntax.

Each variable takes the form <Prefix><Root>Name<Suffix>. Prefix, root, and suffix do not all have to be present. In fact, most variables use only a prefix and a root in combination with the variable name itself. The following table describes the CTOS naming convention.

Table ATM-1. CTOS Variable-Naming Convention

(Page 1 of 2)

Token	Meaning
PREFIXES:	
b	Byte. A character or unsigned 8-bit integer.
c	Count. A two-byte unsigned integer.
f	Flag. A one-byte flag. True = 0FFh, False = 0.
i	Index. A two-byte unsigned integer.
l	Literal. A constant.
n	Number. A two-byte unsigned integer. Same as Count.
o	Offset. A two-byte offset from a segment base address.
p	Pointer. A logical memory address. Consists of a two byte segment identifier and a two-byte offset.
q	Quad. A four-byte unsigned integer.
rg	Array. Usually used with another prefix. For example, the prefix "rgb" identifies an array of bytes.
s	Size. A two-byte unsigned integer.
sb	String. An array of bytes where the first byte is the size of the string.
w	Word. A two-byte integer.
cb	Count of bytes.
pb	Pointer to a string of bytes.

Table ATM-1. CTOS Variable-Naming Convention

(Page 2 of 2)

Token	Meaning
ROOTS:	
erc	Two-byte status code.
exch	Two-byte exchange number.
fh	Two-byte file handle.
lfa	Four-byte logical file address.
ra	Two-byte relative address. Synonymous with offset.
rq	Request block. Size varies.
sa	Two-byte segment identifier.
sn	Selector. Segment identifier for a protected-mode memory address.
sr	Paragraph number. Segment identifier for a real-mode memory address.
userNum	Two-byte user number.
SUFFIXES:	
Last	Largest allowable index in an array.
Max	Maximum size of an array or buffer (Max = Last + 1).
Ret	Indicates a variable whose value is set by a called procedure and returned to the current one.

For example, to define a data buffer using this naming convention, we can assign four variable names:

pBuffer. A pointer to the start of the buffer.

sBufferMax. The maximum size of the buffer.

sBufferDataRet. The size of the data actually written to the buffer, returned by the procedure that writes to the buffer.

psBufferDataRet. Address of sBufferDataRet. Passed to the procedure that writes to the buffer, telling that procedure where to return the value of sBufferDataRet.

Part I – The Essentials

Contents