EARLY MACINTOSH TECHNICAL INFORMATION

INSIDE MACINTOSH WRITER'S GUIDE

COMMENT

INSTRUCTIONS FOR APPLE'S "INSIDE MACINTOSH" WRITERS ON HOW TO WRITE EACH CHAPTER

AUTHOR

APPLE COMPUTER

DATE

26 FEBRUARY 1985

SOURCE

DAVID T CRAIG • JANUARY 2004

MACINTOSH USER EDUCATION

Inside Macintosh Writer's Guide /TOOLBOX/GUIDE

Modification History:	First Draft	Pamela Stanton-Wyman	6/82
	Revised and Expanded	Caroline Rose	9/10/82
	Minor Revision	Caroline Rose	10/8/82
	Renamed and Expanded	Caroline Rose	1/5/83
	Revised and Expanded	Caroline Rose	3/30/83, 5/2/83, 7/13/83, 11/15/83, 2/20/84, 5/30/84, 9/17/84, 2/26/85

ABSTRACT

This document presents an outline of the Inside Macintosh manual and describes the writing conventions being followed in it.

2 Inside Macintosh Writer's Guide

CONTENTS

PREFACE

The Inside Macintosh manual will contain all the information programmers need to create applications for the Macintosh, except for information that depends on the development system being used. It will document the Macintosh User Interface Toolbox and the Operating System, as well as RAM-based software such as the Macintosh Packages.

There are important interdependencies and similarities among the various parts of the Toolbox and between the Toolbox and the Operating System. To underscore this, we've constructed a formal structure under which to organize and write Inside Macintosh. Without this structure, we risk duplication of material and lack of continuity and connection between the various parts of the documentation; with it, we maintain the integrity of the individual parts while making them interconnected and easy to use.

Regarding our general approach to the manual: We want it to be not only useful for reference but also a good source of learning; thus it should be more "tutorial" and "handholding" than a strict reference manual. This is often a difficult balance to achieve. We know our audience consists of educated programmers but we still want to give them the information in small, gradual doses. To this end, we do the following, particularly in our Toolbox documentation:

- We defer the detailed technical discussions, making our initial overviews quite general and saving the details till later wherever feasible.

- We defer mentioning specific routine names until just before the section that discusses them in detail, concentrating on the concepts and ideas in earlier sections.

- We underplay field names because most of the time the user need not be concerned with them at all.

- We try to avoid duplicating technical details.

- We don't have a lot of cross-references, which tend to break up the flow of a discussion and are more suited to a strict reference manual. (Cross-references are more common in the more reference-like parts of the manual, such as the descriptions of routines.)

- We make our headings "functional" wherever we can.

We also try to keep in mind what most of the readers will want to know, and present that first, or at least isolate the more esoteric information with notes that it isn't for most readers. This applies, for example, to details about data structures that can be accessed by routines, and information about defining your own types of windows or controls. Thus the Control Manager documentation is written primarily with the user of standard control types in mind, and the Window Manager documentation is heavily oriented toward users of document windows.

When the advanced programmers get to the section on "customizing", they can, with our help, retrofit that information into what they read before.

OUTLINE OF INSIDE MACINTOSH

Inside Macintosh will consist of three volumes:

- Volume I: the introduction to the entire manual; the "road map" to everything; the User Interface Guidelines; introductory information about memory management and about using assembly language; and the documentation of the Toolbox and other high-level software (such as the Standard File Package)

- Volume II: the documentation of the Operating System and other low-level software (such as the Disk Initialization Package)

- Volume III: the Finder interface, a description of the hardware, the summaries, the appendices, and a comprehensive glossary

Each volume will begin with the following:

{ Disclaimer and other Pubs Information as required; credits to writers }

{ Title page }

Table of contents

{ All first- and second-level headings in the manual }

Each volume will end with a comprehensive index and a comment sheet.

The outlines of the three volumes follow, each on a separate page for easier reading and comparison.

INSIDE MACINTOSH MANUAL: VOLUME I

INSIDE MACINTOSH MANUAL: VOLUME II

6 Inside Macintosh Writer's Guide

INSIDE MACINTOSH MANUAL: VOLUME III

Preface
The Finder Interface
Macintosh Hardware
Summary
Appendices
   Appendix A: Result Codes
   Appendix B: Routines that May Move or Purge Memory
   Appendix C: System Traps
   Appendix D: Global Variables
Glossary
Index

Outline for Individual Chapters

The chapters describing the individual parts of the Toolbox and Operating System are meant for sequential reading as well as for reference. They follow the outline presented below.

Table of contents

{ All headings in the chapter }

About This Chapter

{ A brief introduction to the subject and this chapter; what knowledge is assumed }

About <the Subject>

{ Brief description of the subject, benefits and highlights of features (general, without mentioning specific data types or routines, or any terms they wouldn't already know from the User Interface Guidelines); how they relate to User Interface }

{ sections describing the concepts and data types }

{ The highest-level headings should be somewhat functional rather than strictly reference--for example, "The Editing Environment: Edit Record" rather than "EditRecord Data Type". Don't divide into sections like "Data Types" and "Constants and Variables"; they will be much more meaningful if discussed in some appropriate context, and a list will be provided in the summary at the end.

It's OK to sprinkle these sections with references to specific routines, but place the emphasis on the concepts. See also "More on Data Types", below. }

{ any other sections that seem useful and of general interest }

Using <the Subject>

{ Details about using the subject, such as the order in which you'd most likely do things; mention specific data types or routines. Ties together the next section's individual descriptions. }

<The Subject> Routines

{ Describe the routines; subdivide logically. Don't combine the discussions of two routines; there should be one discussion per routine. Don't begin a routine description with a partial sentence, as in "Initializes...". }

{ sections not of general interest, but rather for advanced

programmers; for example, "Defining Your Own Windows" }

Summary of <the Subject>

{ First, under separate subheads, any constants, data structures, and routines in the Pascal interface (with the routines subdivided logically as in the "Routines" subsection); then any related information for Pascal programmers or of general interest (such as result codes); then the subhead "Assembly-Language Information" with summary information for assembly-language programmers only (constants; offsets into data structures, each under a heading like "Menu Record Data Structure"; special macro names for calling routines). End the assembly-language summary information with "Variables", giving the size of each global variable. }

If there's any information just for assembly-language programmers, it should usually be put in an "assembly-language note" where relevant in the body of the manual. Whenever the trap macro for a Toolbox routine is other than an underline followed by the routine name, say so in an assembly-language note, as in the following example:

Assembly-language note: The macro you invoke to call GetMenu from assembly-language is named _GetRMenu.

In some cases, such as if the material is very lengthy or would otherwise disrupt the body text too much, information for assembly-language programmers should go in a separate "Notes for Assembly-Language Programmers" section (just before the summary at the end). If the information is general, it belongs in the "Using Assembly Language" section in Volume I.

More on Data Types

The following paragraphs present a consistent approach to describing data structures.

Where a structure contains a lot of fields that contain internally used information that the reader need not be concerned with (and would probably be confused by), consider not showing the exact structure at all. If, for example, the remaining fields can all be set with routines, there's a strong case for not showing the exact structure.

If you do show the exact structure:

- Make it clear where appropriate that the programmer can use routines to manipulate the fields (that is, indirect access rather than direct) and need not be concerned about certain ("internal") fields at all. A couple of Macintosh programmers have requested this because they feel the structures can look awfully imposing

and complex at times.

- After each field, put a very brief comment that briefly describes the field. (This will be especially useful in the summary, for quick reference.)

- Explain the fields in their order of appearance when convenient, but don't feel constrained to do so. (Sometimes the programmers group or order fields in a way that's not the best for purposes of documentation.)

- In the explanations of fields, don't underline each field name.

- Don't mention for each field which routine you would use to set it, get its value, or whatever. This gets tedious, is covered in the routine descriptions themselves, is distracting to first-time readers, and is not especially useful for reference purposes (since readers referring back will probably look at the logical subdivisions of the routine calls).

CONVENTIONS

This section lists some of the conventions being followed in Inside Macintosh. They augment (and in some cases override) those presented in the Macintosh Style Sheet for User Manuals and the glossary of the Macintosh Writer's Style Guide. If something isn't listed here, look it up in that glossary or the Style Sheet.

(note)
Consistent terminology and information are of course important between chapters. Be sure to read existing chapters on subjects related to yours. If an existing chapter needs to be changed in view of new information, be sure to let its author know.

When describing a routine whose name doesn't follow one of these conventions, use the conventional term anyway, and relate it to the term used in the name if you feel this is necessary for clarity. For example, if a routine were named CursorFlash, you would say it 'makes the cursor blink ("flash")'. If you can't possibly relate the routine name to the conventional term, don't worry about it.

Word Usage

Use, for example, "bits Ø-6", not "bits Ø through 6" or "bits 6-Ø".

Use "Lisa Workshop Assembler", not "Lisa Assembler".

The system or an application "starts up", not "is started up".

Refer to the "size", not the "length", of a data structure, with some exceptions: The length of a string is the number of characters it currently contains, whereas its size is the maximum number of characters plus 1. "Length" may also be appropriate to use for other variable-length entities; in general, use "size" wherever reasonable.

In assembly-language notes, refer to the "address of" a routine, not a "pointer to" it.

Use "in the heap", not "on the heap".

Don't use "unit" when referring to a part of the Toolbox or Operating System.

Use "midnight", not "12:ØØ AM".

Use "diacritical mark", not "diacritical" alone.

Don't use "launch" to mean starting up an application in general, because there's a specific routine named Launch that does a special kind of startup.

Use "Macintosh Technical Support", not "your software coordinator".

Use "release" rather than "deallocate".

Use "in ROM" rather than "in the ROM".

Don't use "interface" as a verb.

Use "global constant" or "global variable" rather than "system global" (when referring to them individually).

Use "Command key" for the key labeled with the symbol that looks like a freeway interchange.

"Package" has special meaning, so don't use it loosely (as in, for example, "text editing package").

Treat "data" as a singular, not a plural, noun.

Don't use the masculine gender when the person can be of either sex. If you must, use "he or she", but make every effort to avoid such constructions.

Use "graphics" (the noun) before "procedures", but "graphic" (the adjective) before such nouns as "object" and "operation". Don't use "graphic" as a noun.

Use "cursor", not "pointer" (the end-user term). This convention exists to avoid confusion with the other, technical meaning of "pointer"; in general, however, use the end-user terms wherever such confusion does not result.

Don't use "dots" when you mean "bits" or "pixels".

Use "boldfacing" or "boldface" rather than "emboldening", and "bold" rather than "emboldened".

Use "top" instead of "upper" and "bottom" instead of "lower" in, for example, "top right corner".

Windows are stacked from "front" to "back", not from "top" to "bottom".

Don't use "input" or "output" as a verb.

Use "application" program (programmer, programming) rather than "applications". Likewise, use "system" program (etc.) rather than "systems".

Use "boundary rectangle", not "bounding rectangle" or "bounding box". Use this term only when referring to the bounds field of a QuickDraw bit map, not when referring to the "bBox" field of a shape. Call the latter a "rectangle that encloses the shape" ("enclosing rectangle" for short) or "that defines the shape"; use no special term.

Use "resulting" rather than "resultant".

Use "invert" or "highlight" as a transitive verb, not an intransitive one. A procedure may, for example "invert" bits, but a bit "is inverted" (rather than "inverts"). It's OK to use "unhighlight", but use it only when necessary.

Use "empty" rectangle (or region, or whatever), not "null".

Use "character style", not "typestyle" or "typeface attribute". Note that the singular "character style" means the set of stylistic variations, and may be the empty set ("plain" text), only one variation (such as "underlined"), or more than one (such as "bold italic underlined").

Use "NIL" where appropriate; your audience will know what it means.

Don't use "desk" when you mean "desktop".

Use "Return" or "Return character", not "return character" or "carriage return". Similarly, use "Tab" or "Tab character", not "tab character".

Use "space" or "space character" (if necessary for clarity), not "blank".

Use "blink", not "flash" (for "to be repeatedly inverted").

Use "routine" when it can be either a procedure or a function; use the more specific word when possible.

Clarify "truncate" by saying where; do not assume "at the end".

Spelling

"3 1/2-inch" disk, not "3-1/2 inch"

"assembly-language" (adj.), not "assembly language"

"auto-key" event

"base line" (n.), not "baseline"

"bit map", not "bitmap" or "bitMap"

"BOOLEAN" for the reserved word in Pascal, otherwise "Boolean"

"desktop" (adj. or n.), not "desk top"

"disk-inserted" event, not "disk inserted"

"double-click" (v. or n.), not "double click"

"gray", not "grey"

"highlight", not "hilite" or "highlite" or "hilight"

"key-down" event or "key-up" event, not "key down" or "key up"

"menu bar", not "menubar"

"mouse-down" event or "mouse-up" event, not "mouse down" or "mouse up"

"on-line" or "off-line", not "online" or "offline"

"on-screen" or "off-screen", not "onscreen" or "offscreen"

"QuickDraw", not "Quickdraw"

"startup" (adj. or n.), not "start-up"

"text editing" (adj.), not "text-editing (for example, "text editing routines")

"Toolbox", not "ToolBox"

Avoid using words beginning with "non" that aren't in the dictionary, but if you do: Usually there should be no hyphen after "non", as there aren't in such words that are in the dictionary.

Spell out "one" through "ten" in, for example, "four fields", "eight bits", "four-field record", and "eight-bit field", but not in, for example, "bit 5", or "a field containing 1" (the value).

Don't spell out the value "Ø" but do spell out "zero-length" and "nonzero".

Capitalization

Capitalize the first letter and other selected letters of the names listed below. ("Selected" generally means the first letter of what would be a separate word in normal English, as in "ExitToShell".)

- routine names
- data type names except when they're used as nouns to refer to objects, as in "the grafPort" (where the type is GrafPort)
- file names (for example, "ToolEqu.Text", not "TOOLEQU.TEXT")
- names of global variables, except for those declared in the Pascal interface to QuickDraw

Except at the beginning of a sentence: Don't capitalize constants or field names; if the name begins with an acronym that's usually in all capitals, put the entire acronym in lowercase (for example, "eofMarker", not "eOFMarker"). At the beginning of a sentence, capitalize the first letter or, in the case of an acronym, the entire acronym (for example, "EOFMarker contains...").

Capitalize the first letter of the first word of a table heading or the first word of a sentence following a colon, unless the word is a constant or field name.

In assembly-language code, capitalize the first letter of label names and every letter of instruction names.

Capitalize "Figure" in references to illustrations (as in "see Figure 3"), and capitalize figure captions like section headings.

Capitalize "Operating System".

Capitalize "Macintosh User Interface Guidelines".

Underlining

Underline exact manual titles. (Use quotes around chapter titles, unless the chapter can be referred to informally as, for example, the Control Manager chapter.)

Underline column headings in tables.

When emphasizing a word or words a sentence, don't underline; use bold. To make something bold, precede it with <open-apple>-<escape> <shift-Q> and end it with <open-apple>-<escape> <shift-R>. (Notice that <open-apple>-<escape> displays a check mark.)

Don't underline individual field names in discussions of data structures.

For words you're going to include in the glossary, underline them once and only once in the body of the manual. You don't have to do this the very first time you use the term; you might do it near the first full discussion of what the term means.

Don't underline any words that you aren't going to include in the glossary. The reader should see the underline as a "flag" indicating this is a special term, one that will be defined in the glossary. For example, quotes and not underline would be appropriate around something like "flag" in the preceding sentence.

Glossary Conventions

Alphabetize the glossary (and the index) word-by-word, not letter-by-letter; for example, "print record" precedes "printing port". Slashes should be ignored, but hyphens should be treated as spaces (rather than ignored, as some style guides recommend). Also, alphabetize numbers before letters, in numerical order.

Don't define individual types of handles, as in:

xxxxx handle: A pointer to a master pointer to a xxxxx.

(One general definition of "handle" suffices.)

Most glossary items will have the format:

term: Partial sentence.

These formats are also OK:

term: Partial sentence; partial sentence.
term: Partial sentence. Full sentence.

Use this format only when necessary:

term: Full sentence.

But don't use this format:

term: Partial sentence; full sentence.

Index Conventions

Alphabetize as indicated above under "Glossary Conventions".

For every routine whose trap macro has a different name than in Pascal, there should be an index entry for the trap name as well, but without the initial underscore. For example:

SetCtlMin function mm
SetMinCtl function mm

Where there are both high-level and low-level routines that do the same thing, as in the File Manager, sometimes the high-level Pascal name is the same as the low-level macro name, as for GetVol, and sometimes it's not, as for GetVolInfo--yet the "Using..." section always refers to the routine in general by the macro name. The indexing scheme we'll use is illustrated in the following example:

```
GetVInfo function  nn
GetVol function
  high-level  xx
  low-level  yy
GetVolInfo function
  high-level  nn
  low-level  zz
PBGetVInfo function  zz
PBGetVol function  yy
```

Punctuation

When you use "--", precede and follow it with <open-apple>-<left-pointing cursor key> <space>. On Qume printer output, it will give the appearance of no spaces around the "--" yet Script will still recognize word breaks there. This avoids problems with underlining and line breaking. You can do the same after a single "-" if you want, or after something like the "/" in "input/output".

In constructs such as "ARRAY[Ø..9]", do not include a space after "ARRAY".

Put comma and period after a closing quote mark (as in this file), not before it.

Don't precede a colon by a space in data structures or routine calls, except for the colon before the data type returned by a function. In data structures, line up data types following the field names (leave one space after the longest field name).

In the declaration of a routine, don't put a space after the comma between parameters of the same type.

When you show a declaration, include the initial reserved word (for example, TYPE or CONST), and don't forget the ending semicolon.

Don't put a semicolon at the end of the last line of an example of one or more executable Pascal statements.

Follow a colon with only one space in, for example, "VAR foo: Type".

Illustrations

Refer to ("call out") every illustration, preferably just preceding it.

Use 1Ø-point Helvetica for callouts within an illustration. This font and font size were once standard but no longer are. Caroline Rose has a MacPaint disk with a font named "HelveticalØ" on it; the 12-point size of this font is actually 1Ø-point. Some characters look wrong and need fixing up, like the hyphen/minus sign.

Use 12-point Chicago wherever the text is supposed to be in the system font and system font size.

Leave only as much vertical space as needed for each illustration. The most common size is 3 inches, or 18 lines.

Observe a maximum width of 6 inches, and a maximum length of about 8 1/2 inches.

Keep all the illustrations for a particular subject on one disk. Name them Fig1 through FigN, where N is the number of the last illustration.

Always keep backups of your illustrations!

Other Conventions

To assist with the eventual incorporation of your document into the final comprehensive manual, isolate your summary and glossary into separate files named, for example, Menus.S.Text and Menus.G.Text, respectively.

In a chapter on a particular part of the Toolbox or Operating System, when you refer to a procedure in another part of the system, precede it with "the <other part> procedure"; ditto for a function.

Enclose your temporary comments in " *** " (a pair, like brackets). It's important to include such a comment where something will change before the documentation becomes final.

Use contractions, for a more informal style. (Don't use "mustn't", though.)

FORMATTING: SCRIPT MACROS

The standard Script macros for Macintosh technical documentation are in /Pro/Toolbox/MS.Text on Caroline Rose's ProFile. A brief summary of the most commonly used macros follows. See the file itself for more information.

Macro	Description
^pp	Paragraph break; starts a 72-character wide, filled, single-spaced paragraph
^stTEXT	Highest-level heading, entire line underlined above and below
^sbText	Heading a level below "^st", entire line underlined below
^s2Text	Heading a level below "^sb", text underlined below
^fo	Footer; should follow rather than precede the ^st or ^sb at the beginning of a file
^bu	Begins a "bulleted" paragraph
^nu N	Begins a paragraph numbered N
^ds	Begins an indented, unfilled paragraph ("diagram start"); usually followed by an ^ne directive
^ed	Ends an indented, unfilled paragraph ("end diagram")
^ipnote or ^ha	Starts an indented, filled paragraph preceded by "(note)"
^ipwarning or ^ey	Starts an indented, filled paragraph preceded by "(warning)"
^ei	Ends indentation; usually not needed (except after a bulleted item or indented paragraph at the end of a file)
^bx	Starts a "boxed" note, indented, filled, and preceded by a continuous line
^as	Starts a boxed note beginning with "Assembly-language note: "
^eb	Ends a boxed note, with a continuous line
^il N,Text,X,X+2	Leaves space for illustration N, title Text, X lines high (X+2 lines needed)
^pr PROCEDURE...	Starts a routine description (PROCEDURE or FUNCTION followed by the parameters on **one** line)
^ia	Input arrow for OS parameters
^oa	Output arrow for OS parameters
^io	Input/output arrow for OS parameters

The ^as macro is set up so that one-line assembly-language notes won't be split at the end of a page. For an assembly-language note of two lines, insert ^ne 8 before the ^as, and for three lines, ^ne 9. To prevent the text following a ^bx directive from being split across pages, precede the directive with ^ne followed by the number of lines in the text plus 6.

In addition, the tilde character (SHIFT `) is interpreted as a "sticky space" and back slash is interpreted as a comma. One good use of the tilde is before or after punctuation that isn't underlined in a heading, so it will be underlined. You must use back slash to separate parameters of the same type in a ^pr macro, and to include a comma in the text of a heading.

CONCERNS FOR THE FINAL MANUAL

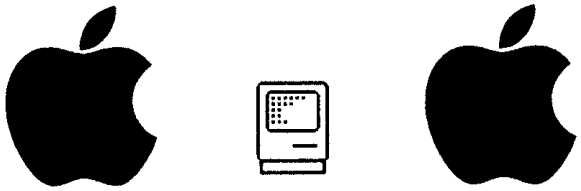
The designations "(note)" and "(warning)" will need to be replaced by an appropriate graphic.

The assembly-language notes will be graphically improved, perhaps printed on a gray background or highlighted in some other way so that Pascal programmers can easily spot and skip them.

We'll need a way of handling tables beyond what we have now. Currently we just let the table fall where it may and precede it by an appropriate "^ne" directive, so there may be a lot of blank space at the bottom of a page preceding the table. Eventually we'll probably want to have some tables set off like illustrations, with a "Table n" designation. This has to be given more thought.

The font(s) we use in the manual **must** clearly distinguish the letter "O" from the number "Ø", and the letters "I" and "l" from each other and from the number "1".

We'll have a special, monospaced "computer voice" font for showing examples, excerpts from the interface/equate files, and other similar indented material. We won't use such a font in prose, however.



END OF DOCUMENT