Apple II
File Type Notes

Developer Technical Support

File Type: $42 (66)
Auxiliary Type: All

Full Name: File Type Descriptors
Short Name: File Type Names

Written by: Matt Deatherage July 1989

Files of this type contain File Type Descriptor tables.

Introduction

As applications continue to be assigned file type and auxiliary type combinations, the task of user file identification becomes more complex. If someone has the list in "About File Type Notes" memorized, simply displaying the file type and auxiliary type in hexadecimal is a suitable way of identification. However, few people have memorized this list. Programs such as the Finder have a need for this information in machine-readable form—not just a list of ASCII strings describing file types, but a way to take a given file's file type and auxiliary type and turn it into a string which describes the file. The Finder is not alone in this need, as parts of command shells, and sometimes entire programs, exist simply to identify files.

Developer Technical Support (DTS) has taken this opportunity to create a data structure that may be used by the Finder and any other application wishing to identify files. By using a separate file, the file identifiers can be updated between System Software releases, at the discretion of DTS, by releasing new descriptor files. Other applications may use the same file without having to reinvent the wheel. Furthermore, the multiple-file structure introduced and suggested in this Note allows developers to ship File Type Descriptor files with their software that allow the Finder and other applications to properly identify these files without a new release of the Finder (much as developers can supply their own Finder icons).

Note: Updated files, if and when released, will not result in changes being made to the System Software. The files as shipped with the System Software will remain unchanged until the next System Software release. Developer Technical Support reserves the right to release updated files for the convenience of those who wish to use them.

The File Format

The file's format is designed to give full and fast access to any file descriptor string, while still remaining flexible enough to grow and be indicative of new features. Each file has three main parts: a header, an index, and the strings.

The Header

The file begins with a header which describes all the entries in the file:

+000	Version	**Word**	Version number. This is toolbox style, so revision x.y would be stored as $0xyy. x is the major revision number; when this value changes, it means that previously-written code will not be able to read this file. yy is the minor version number; when it changes, there are new fields but the old ones are in the same order. This Note describes version 1.0 of the File Type Descriptor format.
+002	Flags	**Word**	This word is all the flags words from all the records combined using a binary `OR` instruction. The flags word for each entry indicates the type of entry it contains (see the section "The Index Entries"). A particular bit in this word will be set if there exists a record in the file where the corresponding bit in the flags word is set. For example, bit 14 will be clear in this word if no index entry has bit 14 set.
+004	NumEntries	**Word**	The number of entries in this file.
+006	SpareWord	**Word**	Reserved for the application's use. The Finder calculates a value for each file and stores it in this field when the file is read into memory. This should be zero in the file on disk.
+008	IndexRecordSize	**Word**	The number of bytes in each index record.
+010	OffsetToIdx	**Word**	Offset, from the beginning of the file, to the first index entry. This field is provided so that other fields may be added to the header at a later date without breaking existing programs.

The Index Entries

The descriptions for each file type and auxiliary type assignment are pointed to by index entries for each string. If there is a string in the file that should be displayed for a particular assignment, there will be an index entry for it. If there is not an entry in **any** of the loaded files (see the section "Having More Than One File Type Descriptor File"), the string for file type $0000, auxiliary type $00000000 should be displayed.

The index contains one index entry for every file type and auxiliary type assignment or range (see below) in the descriptor file. All index entries in a given file are the same length (given in the header) so fast binary-searching algorithms may be performed. Their format is as follows for files with major version 1:

+000	Filetype	**Word**	The file type that should match for the string to which this index entry points.
+002	Auxtype	**Long**	The auxiliary type that should match for the string to which this index entry points.

+006	Flags	**Flag Word**	A word, defined bit-wise, indicating the type of match this entry contains. The following definitions apply if the bit in question is set: Bit 15: This record matches this file type and any auxiliary type. This bit would be set, for example, for a record for file type $FF (ProDOS 8 application). Bit 14: This record matches this auxiliary type and any file type. Bit 13: This record is the beginning of a range of file types and auxiliary types to match this string. Any file type and auxiliary type combination falling linearly between this record and the record with the same offset and bit 12 set should be given this string by default if no specific match is found. Bit 12: This record is the end of a range of file types and auxiliary types to match this string. Any file type and auxiliary type combination falling linearly between the record with the same offset and bit 13 set and this record should be given this string by default if no specific match is found. Bits 11-0: Reserved, must be set to zero when creating files.

Note: A range uses the file type and auxiliary type combined as a six-byte value, with the file type bytes as most significant. For example, file type $15, auxiliary type $4000 would fall in the range that starts with file type $13, auxiliary type $0800 and ends with file type $17, auxiliary type $2000

+008	Offset	**Word**	The offset, from the beginning of the file, to the Pascal string matching the description in this index entry. Note that more than one index entry can point to the same string.

The Strings

Since each index entry contains an offset to a string, it seems only logical that somewhere in the file is a string for each index entry. Apple recommends that the strings be placed in an array at the end of the index for most efficient use of space and ease in creating the file.

General Truths

So programs using File Type Descriptor files or resources don't have to construct all information about them each time they are opened, certain characteristics will be true of all files. The following are characteristics which will always be true for files or resources with major revision number $01:

- The strings describing the files must each be no more than 30 characters long.
- The entries must always be sorted primarily by ascending file type and secondarily by ascending auxiliary type.
- Records that match file types or auxiliary types generically (for example, file type $FF and any auxiliary type) must contain zeroes for the wildcard field. A descriptor for ProDOS 8 application files should have file type $00FF, auxiliary type $00000000 and bit 15 set in the flags word. This record should be before any specific match for a file that has file type $FF and auxiliary type $0000, if

such a record were to exist. Similarly, records which match a certain auxiliary type and any file type should appear before records which match file type $00 and that auxiliary type.

- The index entry marking the beginning of a range and the index entry marking the end of a range must not have any other index entries between them.
- Range index entries may not have bit 14 set.

Having More Than One File Type Descriptor File

More than one File Type Descriptor file may exist in a given directory. However, only one file may exist in a given directory with any auxiliary type from $00000000 to $000000FF. These files are provided by Apple Computer, Inc. and should **not** be altered by anything containing carbon atoms. Future implementations of System Software reserve the right to assume undocumented properties about File Type Descriptor files with auxiliary types smaller than $00000100. Editing of the strings in these files is not necessary, since other files may contain strings to override the ones in these files.

There is no such restriction on auxiliary types of $00000100 or greater.

To provide flexibility in changing file descriptions, applications should build in the capability to use as many File Type Descriptor files as are present. Files created by third-parties must follow the following two rules:

- The auxiliary type must not be lower than $00000100. Auxiliary types below $0100 are reserved for Apple.

- The File Type Descriptors must not contain a match for file type $00 and auxiliary type $0000. Such a descriptor contains the string to display for a file that does not match any other index entry. This entry must only be contained in the File Type Descriptor with auxiliary type zero.

A file with auxiliary type zero **must** exist. Others should be searched in order of descending auxiliary type, with $FFFFFFFF having highest priority. (This is why no file must contain a match for file type $00 and auxiliary type $0000 except the Apple-supplied one; otherwise, no searching would ever be done beyond the offending file.) In this way, strings in the Apple-supplied files may be superseded by other strings, without replacing or altering the Apple-supplied file (a feat that would be difficult anyway, due to the offset nature of the file structure).

Program Use of More Than One File

Applications should search the directory for as many of the given files as can be found. If none is found with auxiliary type $0000, then the application should behave as if no files were found. When searching for a description, a separate search can be done on each file, stopping when a match is found. The search algorithm should return the “unknown” string when no specific match is found in the Apple-supplied file, so the search process will always return **some** string. An application should never run out of File Type Descriptors before finding a match.

Adding a File

Developers who wish to ship their own File Type Descriptor file with their product may contact Developer Technical Support for assistance in creating the file.

Memory Considerations

An application (especially a ProDOS 8 application) may not wish to spend valuable memory on files for file identification purposes, especially if directory listings are not an important part of the application. Since all offsets in the File Type Descriptor files are offsets from the beginning of the file, they may also be used with the ProDOS 8 or GS/OS `SetMark` call. Disk-based searches will obviously be much slower, but could be used for special instances (such as printing complete directories of disks as opposed to displaying them, or for specific functions that identify files).

About the Finder's Implementation

In Apple IIGS System Software 5.0, the Finder uses File Type Descriptor files. The Finder's implementation is somewhat limited, as this is a first pass at this new standard. The following implementation notes apply to Finder 1.3:

- The Finder looks for the File Type Descriptor files in the Icons directory of the boot disk (pathname *:Icons). It does not look in other directories or on other disks.
- Up to 30 File Type Descriptor files will be loaded.
- Two File Type Descriptor files are shipped with System Software 5.0. The first, FType.Main, contains file type descriptions for a small subset of file types, and no specific auxiliary types. This file will be loaded on machines with 512K or less of memory. The second file, FType.Aux, contains the rest of the descriptions shipped with System Software 5.0, as listed in "About Apple II File Type Notes" for July, 1989, and this file will be loaded in addition to the first on machines with more than 512K of memory. FType.Main has auxiliary type $0000; FType.Aux has auxiliary type $0001. The Finder does not depend on the names, but on the auxiliary types and file types.
- If the Finder cannot find any File Type Descriptor files in the *:Icons directory, it will terminate with fatal system error $4242. If it can not find a File Type Descriptor file with auxiliary type $0000, it will terminate with fatal system error $4243.
- The Finder will only use File Type Descriptor files with major version number $01. Also, the file will not be used if it any bits in the flags word of the header other than bit 15 are set, or if the spare word in the header is not zero, or if there are zero entries in the file. The Finder's search algorithm is fast, but currently does not handle special index entries other than for a given file type and any auxiliary type.

Further Reference

- About Apple II File Type Notes